

Administering webMethods Broker

Version 9.6

April 2014

This document applies to webMethods Broker Version 9.6 and to all subsequent releases.

Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

Copyright © 2007-2019 Software AG, Darmstadt, Germany and/or Software AG USA Inc., Reston, VA, USA, and/or its subsidiaries and/or its affiliates and/or their licensors.

The name Software AG and all Software AG product names are either trademarks or registered trademarks of Software AG and/or Software AG USA Inc. and/or its subsidiaries and/or its affiliates and/or their licensors. Other company and product names mentioned herein may be trademarks of their respective owners.

Detailed information on trademarks and patents owned by Software AG and/or its subsidiaries is located at <http://softwareag.com/licenses>.

Use of this software is subject to adherence to Software AG's licensing conditions and terms. These terms are part of the product documentation, located at <http://softwareag.com/licenses> and/or in the root installation directory of the licensed product(s).

This software may include portions of third-party products. For third-party copyright notices, license terms, additional rights or restrictions, please refer to "License Texts, Copyright Notices and Disclaimers of Third Party Products". For certain specific third-party license restrictions, please refer to section E of the Legal Notices available under "License Terms and Conditions for Use of Software AG Products / Copyright and Trademark Notices of Software AG Products". These documents are part of the product documentation, located at <http://softwareag.com/licenses> and/or in the root installation directory of the licensed product(s).

Use, reproduction, transfer, publication or disclosure is prohibited except as specifically provided for in your License Agreement with Software AG.

Table of Contents

About this Guide.....	23
Document Conventions.....	23
Documentation Installation.....	24
Online Information.....	24
webMethods Broker.....	27
Overview.....	28
Publish-and-Subscribe Model.....	28
Producers and Consumers Are De-coupled.....	28
Interaction Is Asynchronous.....	29
Delivery Quality of Service.....	29
Broker's Relationship with Other webMethods Components.....	29
Broker Client APIs.....	31
webMethods Broker Architecture and Components.....	31
Broker Server.....	32
Brokers.....	33
Broker Monitor.....	34
Broker User Interface.....	35
Document Types.....	35
Client Groups.....	36
Clients (Client State Objects).....	37
Subscriptions.....	38
Queues.....	38
Storage Type.....	39
Queue Storage (QS).....	39
Territories.....	39
Territory Gateways.....	40
Clusters.....	41
Cluster Gateways.....	41
Broker Security Model.....	42
Working with Firewalls.....	43
Accessing webMethods Broker through a Firewall.....	43
Enable Full Core Dumps on an AIX System.....	44
Using My webMethods to Administer webMethods Broker.....	45
Managing Broker Using My webMethods.....	46
Selecting a Database.....	46
Using My webMethods with Earlier Versions of Broker.....	46
Who Can Use the Broker User Interface?.....	46
Extending Access to Other Users.....	47
Accessing the Broker User Interface.....	47

How My webMethods Interacts with Broker Servers.....	48
Logging Out of My webMethods.....	48
Adding Broker Servers to My webMethods.....	49
Adding an Individual Broker Server to My webMethods.....	49
Discovering Broker Servers on a Specified Broker Monitor Host Machine.....	50
Uploading a List of Broker Servers into My webMethods.....	51
Creating the Definitions File.....	51
Uploading the Definitions File to My webMethods.....	52
Removing Broker Servers from My webMethods.....	52
Configuring the Connection Parameters.....	53
Viewing and Editing the Connection Parameters.....	54
Using My webMethods with ACL-Protected Broker Servers.....	54
Broker Monitor.....	57
The Role of Broker Monitor.....	58
How Broker Monitor Starts Broker Servers.....	59
How Broker Monitor Monitors the State of Broker Servers.....	59
Status Messages Logged by Broker Monitor.....	59
Starting Broker Monitor.....	60
Starting Broker Monitor Automatically on Windows Systems.....	60
Starting Broker Monitor Manually on Windows Systems.....	60
Starting Broker Monitor Automatically on UNIX Systems.....	61
Starting Broker Monitor Manually on UNIX Systems.....	62
Stopping Broker Monitor.....	62
Stopping Broker Monitor on Windows.....	62
Stopping Broker Monitor on UNIX.....	63
How to Secure Broker Monitor Access.....	64
How to Determine Whether Broker Monitor Is Running.....	64
Checking on Windows.....	64
Checking on UNIX.....	64
Ports and Running Multiple Instances of Broker Monitor.....	65
Viewing the Port Setting for Broker Monitor.....	66
Changing the Port Setting for Broker Monitor.....	66
Configuring Broker Monitors to Start Automatically on UNIX Systems.....	67
Managing Broker Servers.....	69
Overview.....	70
Broker Data Directory.....	70
Broker Server Configuration File (awbroker.cfg).....	71
Starting Broker Server.....	71
Starting Broker Server from My webMethods.....	72
Starting Broker Server Using the broker_start Command.....	72
Starting Broker Server from Windows Control Panel.....	73
Stopping Broker Server.....	73
Stopping Broker Server from My webMethods.....	74
Stopping Broker Server Using the broker_stop Command.....	74

Stopping Broker Server from Windows Control Panel.....	75
Broker Server Communication Ports.....	76
Ports Used by Broker Server.....	76
Ports and Multiple Broker Servers.....	77
Viewing the Base Port Setting.....	77
Changing the Base Port Setting.....	78
Choosing the Storage Solution for Broker Server.....	79
Broker Server Memory.....	79
Controlling Queue Size for Volatile Documents.....	79
Limiting Memory Usage by Broker Server.....	80
Maximum Limit Notification.....	80
Values for the max-memory-size Parameter.....	80
The preallocate-memory Parameter.....	81
Selecting the Maximum Memory Size.....	81
Limitations of the Maximum Memory Size.....	82
Configuring the max-memory-size Parameter.....	83
Configuring Queue Storage.....	84
The Log File.....	84
The Storage Files.....	85
Combined and Separate Storage Sessions.....	86
Default Queue Storage Files Created by the Installer.....	87
Default File Names and Sizes Created by server_config.....	88
Modifying the Size of the Log File.....	88
Modifying the Size of a Storage File.....	90
Adding a Storage File.....	91
Viewing Queue Storage Utilization and Settings.....	94
Configuring the Cache Settings.....	95
Default Cache Settings.....	95
Selecting a Cache Size.....	96
Modifying the Cache Size.....	96
Viewing the Cache Settings.....	97
Configuring Broker Server to Use Asynchronous Write Mode.....	98
Viewing the Write-Mode Setting.....	99
Running Multiple Broker Servers on the Same Host Machine.....	100
Running Multiple Broker Servers in a Single webMethods Broker Environment.....	100
Running Multiple Instances of webMethods Broker on the Same Host.....	101
Creating a Broker Server with the Default Log and Storage Files.....	101
Creating a Broker Server that Specifies Size and Location of Files.....	103
Deleting a Broker Server.....	104
Viewing Status Information for a Broker Server.....	105
Broker Server License.....	108
Contents of License File.....	108
Viewing and Changing the License Using My webMethods.....	110
Viewing and Changing the License Using server_config.....	111
Monitoring Resource Utilization.....	111

Logging.....	112
The Broker Server Log.....	113
Log File Names and Locations.....	113
Message Types.....	113
Including Log Messages in Other Logging Systems.....	114
Configuring the Logging Behavior of Broker Server.....	114
Configuring the Logging Behavior of Broker Monitor.....	115
Viewing the Broker Server Log.....	116
Purging the Broker Server Log.....	117
The Messaging Log.....	117
Message Types.....	117
Log File Names and Locations.....	118
Continuous versus Daily Log Files.....	118
Viewing the Messaging Log.....	119
Configuring the Messaging Log.....	119
Purging the Messaging Log that Uses Daily Files.....	120
Purging the Messaging Log that Uses Continuous Files.....	120
Internet Protocol Support.....	121
Backing Up and Restoring a Broker Server.....	121
Metadata that Is Backed Up.....	121
When to Perform a Backup.....	122
Other Files That You Should Backup.....	122
Backing Up Territories and Gateways.....	123
Choosing the Correct Backup Procedure.....	123
Backing Up a Separate Storage Configuration.....	123
Files that Are Backed Up.....	124
Backing Up an ACL-Protected Broker Server.....	124
Restoring a Separate Storage Configuration.....	125
Resynchronizing a Restored Broker with its Territory.....	128
Backing Up a Combined Storage Configuration.....	129
Preparing for the Backup.....	129
Precautions and Alternatives.....	129
Restoring a Combined Storage Configuration.....	130
Pausing Document Publishing on the Broker Server.....	131
Configuring Broker Server for high throughput in high-latency and high-bandwidth networks.....	132
Configuring Streaming in Broker Servers.....	132
Configuring Parallel Channels in Broker Servers.....	134
Setting the Locale.....	135
Managing Brokers.....	137
Role of Broker.....	138
Creating a Broker.....	140
Deleting a Broker.....	141
Configuring a Default Broker.....	142

Determining Which Broker Is the Default.....	142
Designating a Default Broker.....	142
Using Document Logging.....	143
Determining Whether Document Logging Is Enabled.....	143
Checking the Length of the Document Logging Queue.....	143
Routing Documents to the Dead Letter Queue.....	144
The Dead Letter Client.....	144
Relationship to Dead Letter Queues Created by Client Applications.....	144
Activating the Dead Letter Queue.....	144
Viewing and Purging the Dead Letter Queue.....	145
Disabling the Dead Letter Queue.....	146
Viewing Status Information for a Broker.....	146
Viewing Basic Operating Statistics for the Broker.....	148
Managing Transactions.....	150
Viewing Running Transactions.....	150
About Configuring the Transaction Timeout Options.....	152
Configuring the Transaction Timeout Options.....	153
Setting the Recover Mode for XA Transactions.....	154
Manually Performing a Commit or Roll Back.....	154
Viewing and Purging Lost Transactions.....	155
Managing Document Types.....	157
Overview.....	158
What Are Documents and Document Types?.....	158
Creating Document Types.....	158
About Document Type Names.....	159
Document Type Properties.....	159
Time to Live.....	159
Document Type Validation.....	160
Document Type Storage Types.....	161
Client Group Storage Types and Document Type Storage Types.....	161
Document Field Information.....	162
Infosets Information.....	162
Viewing Document Types on a Broker.....	163
Viewing Document Type Configuration Information.....	164
Viewing Document Type Fields and Infosets.....	166
Viewing Which Clients Subscribed to a Document Type.....	167
Modifying Document Type Properties.....	167
Copying Document Types Between Brokers.....	169
Copying and Pasting Document Types.....	169
Deleting Document Types.....	170
Managing Client Groups.....	171
Overview.....	172
Defining a Client Group.....	172
Client Groups and Broker Security.....	172

Client Group Properties.....	173
Client Queue Storage Type.....	174
Queue Storage Type and Lifecycle.....	175
Queue Storage Type and Document Type.....	175
Client Lifecycle.....	175
Client Group Access Control Lists (ACLs).....	176
System-Defined Client Groups.....	176
admin Client Group.....	177
accessLabelAdapter Client Group.....	177
adapters Client Group.....	177
eventLog Client Group.....	178
Working with Client Groups.....	178
Searching for Client Groups.....	178
Creating a Client Group.....	179
Viewing and Editing Client Groups.....	180
Adding Document Type Permissions for Client Groups.....	184
Removing Document Type Permissions for Client Groups.....	184
Copying Client Groups between Brokers.....	184
Using Copy and Paste.....	185
Deleting a Client Group.....	185
Managing Clients.....	187
Overview.....	188
What Is a Client?.....	188
Client State.....	188
Shared State Clients.....	189
Shared State Order.....	189
Storage Type and Lifecycle.....	189
Viewing Clients on a Broker.....	190
Viewing Client Configuration Information.....	193
Viewing Client Statistics.....	196
Managing Subscriptions.....	199
Viewing Subscriptions.....	199
Deleting Subscriptions.....	200
Managing Sessions.....	200
Viewing Session Information.....	201
Viewing Detailed Session Information.....	202
About Disconnecting Sessions.....	204
Before Disconnecting a Session.....	204
Disconnecting a Session.....	204
Copying Clients between Brokers.....	205
Copying and Pasting Clients.....	205
Deleting Clients.....	206
Working with Test Clients.....	207
Overview.....	208

What Is a Test Client?.....	208
Viewing Test Clients on a Broker.....	208
Creating Test Clients.....	209
Creating a Test Client that Uses Basic Authentication.....	210
Creating a Test Client that Uses SSL Authentication.....	211
Creating a Test Client that Uses One-way SSL Authentication.....	212
Working in Test Client View.....	213
Managing Document Subscriptions for a Test Client.....	214
Creating Subscriptions for a Test Client.....	214
Creating or Editing a Filter.....	215
Deleting Subscriptions for a Test Client.....	215
Publishing Documents with a Test Client.....	216
Building an Outgoing Documents List.....	216
Adding Blank Documents.....	217
Loading Documents from a File.....	218
Copying and Pasting Documents in the Outgoing Documents List.....	219
Adding Comments to Documents.....	220
Publishing Documents with a Test Client.....	221
Before Publishing Documents with a Test Client.....	221
Publishing or Delivering Documents with a Test Client.....	221
Stopping Publishing by a Test Client.....	222
Retrieving Documents for a Test Client.....	223
Clearing the Incoming Documents List.....	224
Clearing the Test Client Queue.....	225
Saving Outgoing or Incoming Documents.....	225
Disconnecting a Test Client.....	226
Reconnecting Test Clients.....	227
Reconnecting a Test Client that Uses Basic Authentication.....	227
Reconnecting a Test Client that Uses SSL Authentication.....	228
Reconnecting a Test Client that Uses One-way SSL Authentication.....	228
Deleting a Test Client.....	229
Managing Client Queues.....	231
Overview.....	232
What Is a Client Queue?.....	232
Client Queue Ownership.....	232
Viewing Client Queue Statistics.....	233
Locking a Client Queue.....	233
Unlocking a Client Queue.....	234
Viewing Lock Information for a Queue.....	234
Browsing a Client Queue.....	235
Filtering Queue Contents while Browsing.....	236
Saving Client Queue Documents to a File.....	237
Inserting Documents into a Client Queue.....	238
Loading Documents from a File.....	239

Copying and Pasting Documents.....	240
Inserting Documents.....	240
Deleting Documents from a Client Queue.....	241
Clearing Client Queues.....	242
Before Clearing a Client Queue.....	242
Clearing a Client Queue.....	242
Proactively Deleting Documents from a Client Queue.....	243
Managing Forwarding Queues.....	245
Overview.....	246
What Is a Forwarding Queue?.....	246
Navigating to the Forwarding Queue Browser Page.....	246
Navigating to the Forwarding Queue for a Remote Broker in Your Territory.....	246
Navigating to the Forwarding Queue for a Remote Broker in Your Cluster.....	247
Navigating to the Forwarding Queue for a Remote Broker in a Remote Territory.....	247
Navigating to the Forwarding Queue for a Remote Broker in a Remote Cluster.....	247
Locking and Unlocking the Forwarding Queue.....	248
Browsing a Forwarding Queue.....	249
Filtering Forwarding Queue Contents while Browsing.....	251
Filtering Contents for a Remote Broker in Your Territory or Cluster.....	251
Filtering Contents for a Remote Broker in a Remote Territory or Cluster.....	252
Saving the Forwarding Queue Documents to a File.....	252
Inserting Documents into a Forwarding Queue.....	253
Loading Documents from a File.....	254
Copying and Pasting Documents.....	254
Inserting Documents.....	255
Deleting Documents from a Forwarding Queue.....	255
Proactively Deleting Documents from a Forwarding Queue.....	256
Forwarding Messages to a Remote Cluster or Territory.....	256
Using Topology View and Document Trace.....	259
Topology View of Brokers.....	260
Data Displayed in the Broker Topology View.....	260
Displaying the Broker Topology View.....	262
About Tracing Documents in Broker Topology View.....	263
Tracing Documents in the Topology View.....	263
Setting Your User Preferences for the Display Settings.....	264
Defining How to Display Data in the Broker Topology View.....	265
Defining the Brokers and Clients to Display in the Topology View.....	267
Editing Existing Node Groupings.....	269
Deleting Node Groupings.....	270
About Ordering the Node Groupings.....	270
Ordering Node Groupings.....	271
Defining the Documents to Include in Traces.....	271
Table View of Trace Information.....	272
Data Displayed in Trace View.....	272

Setting Your User Preferences for the Trace View.....	274
Running Document Traces in Trace View.....	274
Refreshing the Data in the Trace View.....	274
Stopping the Trace in the Trace View.....	274
Exporting the Data in the Trace View to a .csv File.....	275
Clearing the Trace Data.....	275
Topology View of Territories.....	275
Data Displayed in the Territory Topology View.....	276
Displaying the Territory Topology View.....	277
About Tracing a Document in the Territory Topology View.....	278
Tracing a Document in the Territory Topology View.....	278
Setting Your User Preferences for the Territory Topology View.....	279
Defining How to Display Data in the Territory Topology View.....	280
Identifying the Territories to Show in the Topology View.....	281
Editing Existing Node Groupings.....	283
Deleting Node Groupings.....	283
About Ordering the Node Groupings.....	283
Ordering Node Groupings.....	284
Viewing Documents that Are Included in Territory Topology Document Traces.....	285
Managing Broker Security.....	287
Overview.....	288
Data Protection.....	288
Broker Security Model.....	288
Authentication Identities.....	290
Access Control Lists (ACLs).....	290
Securing Broker System Components.....	291
Securing the Broker Server.....	291
Securing the Broker User Interface.....	292
Securing Broker Command-Line Utilities.....	292
Securing Client Groups.....	292
Securing Territories and Gateways.....	292
Securing Clusters and Cluster Gateways.....	293
Securing Broker Server Using Basic Authentication.....	293
Using the Basic Authentication Security Model.....	293
Enabling Basic Authentication.....	294
Basic Authentication Configuration File.....	294
Basic Authentication on UNIX.....	295
Basic Authentication Configuration Parameters.....	296
Default Values for Active Directory Parameters.....	305
Default Values for SunOne Parameters.....	306
Default Values for OpenLdap Parameters.....	306
Basic Authentication Alias.....	307
Disabling Basic Authentication Alias.....	307
Configuring Basic Authentication Identity for a Broker Server.....	308

Disabling the Basic Authentication Identity of a Broker Server.....	309
Securing Broker Server Using SSL.....	309
Broker Security Model and OpenSSL.....	309
Assigning SSL Identities for a Broker Server.....	309
One-Way and Two-Way SSL Authentication.....	310
SSL Authentication and Broker Port Usage.....	310
Broker Server Identity and SSL.....	311
Certificate Files.....	312
Keystore File.....	312
Keystore File Formats.....	313
Certificate Chains.....	313
Installing a Certificate.....	313
Truststore File.....	314
Truststore File Formats.....	314
How Broker Uses a Keystore and Truststore.....	315
Protecting Keystore and Truststore Files.....	315
Using SSL Encryption.....	316
OpenSSL Cryptography.....	316
Configuring SSL for Broker Server.....	317
Creating Keystores and Truststores.....	318
Loading Keystores and Truststores into My webMethods.....	318
Loading Keystores from the Machine on Which Broker is Administered.....	318
Loading Keystores on the Machine Hosting My webMethods Server.....	319
Loading Truststores from the Machine on Which Broker is Administered.....	319
Loading Truststores on the Machine Hosting My webMethods Server.....	319
Modifying Keystores and Truststores.....	320
Configuring an SSL Identity for a Broker Server.....	320
Disabling the SSL Identity for a Broker Server.....	322
Configuring SSL for Clients.....	322
Securing Broker Server Using CRL.....	323
Configuring Certificate Revocation Checking.....	323
Configuring CRL Checking For the Entire Certificate Chain.....	324
Securing Broker Server Using Basic Authentication Over SSL.....	324
Securing Broker Server Using FIPS.....	324
Starting Broker Server in FIPS mode.....	325
Access Control Lists.....	325
Authenticator vs. User ACLs.....	325
How User Rights Are Granted.....	326
Client Group ACLs.....	326
admin Client Group ACLs.....	328
Broker Server ACLs.....	328
Territory ACLs.....	329
Cluster ACLs.....	330
Territory Gateway ACLs.....	331
Cluster Gateway ACLs.....	332

Configuring Access Control Lists.....	333
About Configuring Broker Server ACLs.....	333
Adding Authenticator Names.....	333
Adding User Names.....	334
Disabling Broker Server ACLs.....	335
Deleting Authenticator Names.....	335
Granting Access Permissions to a New Client.....	336
About Configuring Client Group ACLs.....	337
Registering User Names.....	337
Registering Authenticator Names.....	338
About Controlling Which Brokers Can Join a Territory.....	339
Specifying Brokers Using User Name DNs.....	339
Specifying Brokers Using Authenticator Names.....	340
Deleting Brokers From a Territory ACL.....	340
About Controlling Which Brokers Can Join a Cluster.....	341
Specifying Brokers Using User Names.....	341
Specifying Brokers Using Authenticator Names.....	342
Deleting Brokers From a Cluster.....	343
Setting Permissions to Connect to Remote Brokers across a Territory Gateway.....	344
Configuring Broker Gateway ACLs Using User Names.....	344
Configuring Broker Gateway ACLs Using Authenticator Names.....	344
About Setting Permissions to Connect to Remote Brokers across a Cluster Gateway.....	345
Configuring Cluster Gateway ACLs Using Broker User Names.....	345
Configuring Cluster Gateway ACLs Using Broker Authenticator Names.....	345
ACLs and Migration Issues.....	346
Broker Security and My webMethods Server.....	346
Configuring a Basic Authentication Identity for the Broker User Interface Component.....	347
Configuring an SSL Identity for the Broker User Interface Component.....	348
Configuring for One-way SSL Authentication.....	349
Reconfiguring the Identity of the Broker User Interface Component.....	350
Disabling the Identity of User Interface Component to the Broker Server.....	350
Viewing the Identity Details of the Broker User Interface Component to the Broker Server.....	350
Converting Certificate Files.....	351
Broker Certificate Conversion Utility.....	351
Single Certificate Format.....	351
Trusted Roots and Truststore Files.....	351
Downloading the Broker Certificate Conversion Utility.....	352
Command Usage.....	352
Remarks.....	352
Managing Territories.....	353
Introduction to Territories.....	354
Why Implement a Territory?.....	354
How Brokers in a Territory Communicate with Each Other.....	355

Subscriptions Maintained by a Remote Broker Object.....	356
How Subscription Filters Are Handled in a Territory.....	356
The Queue Maintained by a Remote Broker.....	356
How Delivered Documents Are Handled in a Territory.....	357
How Brokers Synchronize Metadata.....	358
Exporting and Importing Territory Information.....	358
Securing a Territory.....	358
Leaving a Territory.....	358
Creating a Territory.....	359
Requirements Checklist.....	359
Reverse Lookup.....	360
Creating Territories.....	361
Adding Brokers to a Territory.....	362
Removing a Broker from a Territory.....	363
Deleting a Territory.....	363
Viewing a List of Known Territories.....	363
Viewing Information about a Remote Broker Object.....	364
Deleting a Remote Broker Object.....	366
Managing Territory Gateways.....	367
Overview.....	368
Territory Gateway Connections, Territory Gateways, and Territory Gateway Pairs.....	368
Territory Gateway Names.....	369
Connecting Multiple Territories with Territory Gateway Connections.....	369
Creating a Territory Gateway Connection.....	370
Creating a "One-Way" Territory Gateway.....	371
Configuring and Controlling Territory Gateway Behavior.....	372
Creating a Territory Gateway Pair if You Control Both Brokers.....	373
Defining the Territory Gateway Pair.....	373
Specifying which Documents Types Territories Can Exchange.....	374
Specifying Permissions for Receiving Individual Document Types.....	374
Specifying Permissions for Forwarding Individual Document Types.....	375
Using Client Groups to Specify Permissions on Document Types.....	376
Creating a Territory Gateway Pair if You Control Only One Broker.....	377
Defining One Side of a Territory Gateway Pair.....	378
Specifying which Documents the Territory Gateway Can Forward and Receive.....	378
Using Territory Gateway Filters.....	379
Filtering Documents.....	379
Filter Rules.....	380
Filter Operators.....	381
Viewing Information about a Territory Gateway.....	383
Viewing Information about Documents Flowing between the Gateways.....	386
Deleting a Territory Gateway Connection.....	387
Removing an Allowed Document Type from a Territory Gateway.....	388
Pausing and Resuming a Territory Gateway.....	389

Pausing a Territory Gateway.....	389
Resume a Territory Gateway.....	390
Preventing Connection Timeouts between Territories.....	390
Viewing and Modifying the Keep-Alive Setting.....	390
Configuring the Forwarding Mode.....	391
Refusing Document-Type Updates.....	392
Viewing or Configuring the Global Refuse Update Option.....	392
Viewing or Configuring the Refuse Update Option for an Individual Document Type.....	393
Using Broker Remote Publish.....	393
Broker in the Same Territory.....	395
Broker Peer of Local Territory Gateway.....	395
Broker in a Different Territory.....	395
Same as Delivering Broker.....	396
Managing Clusters.....	397
Introduction to Clusters.....	398
Why Implement a Cluster?.....	398
How Brokers in a Cluster Communicate with Each Other.....	399
How Delivered Documents Are Handled in a Cluster.....	399
How Brokers Synchronize Metadata.....	399
Exporting and Importing Cluster Information.....	400
Securing a Cluster.....	400
Leaving a Cluster.....	400
Creating Clusters.....	400
Requirements Checklist.....	400
Creating a Cluster.....	401
Adding Brokers to a Cluster.....	402
Removing a Broker from a Cluster.....	403
Deleting a Cluster.....	403
Viewing a List of Known Clusters.....	403
Managing Cluster Gateways.....	405
Overview.....	406
Cluster Gateway Connections, Cluster Gateways, Cluster Gateway Pairs, and Cluster Gateway Brokers.....	407
Cluster Gateway Names.....	408
Connecting Multiple Clusters with Cluster Gateways.....	408
Creating a Cluster Gateway.....	409
Creating a "One-Way" Cluster Gateway.....	410
Configuring and Controlling Cluster Gateway Behavior.....	411
Creating a Cluster Gateway when You Control Both Sides.....	412
Defining the Cluster Gateway Pair.....	412
Exchanging Document Types across a Cluster Gateway.....	413
Specifying Individual Document Types a Cluster Gateway Can Receive.....	413
Specifying Individual Document Types a Cluster Gateway Can Forward.....	414
About Using Client Groups to Specify Permissions on Document Types.....	415

Using Client Groups to Specify Permissions on Document Types.....	416
Creating a Cluster Gateway when You Control Only One Side.....	417
Defining One Side of a Cluster Gateway Pair.....	417
Specifying which Documents the Cluster Gateway Can Forward and Receive.....	418
Filtering Documents across a Cluster Gateway.....	418
Filtering Documents.....	419
Filter Rules.....	419
Filter Operators.....	420
Viewing Cluster Gateway Details.....	422
Editing Cluster Gateway Details.....	425
Viewing Document Details across a Cluster Gateway.....	425
Deleting a Cluster Gateway.....	427
Removing an Allowed Document Type from a Cluster Gateway.....	428
Pausing and Resuming a Cluster Gateway.....	429
Pausing a Cluster Gateway.....	429
Resuming a Cluster Gateway.....	430
Preventing Connection Timeouts between Clusters.....	430
Configuring the Forwarding Mode.....	431
Refusing Document Type Updates from a Remote Cluster Gateway.....	432
Blocking Document Types Globally.....	433
Blocking Document Types Individually.....	433
Using My webMethods with JMS.....	435
Overview.....	436
JMS and Traditional Broker Applications.....	436
webMethods Broker as a JMS Provider.....	437
JMS Messages.....	437
JMS Messaging Model.....	437
Pub-Sub Messaging.....	437
Point-to-Point Messaging.....	437
JMS Connections.....	438
Session.....	438
Message Consumer.....	438
Durable Subscribers.....	439
Message Delivery Mode.....	439
Message Acknowledgement Mode.....	440
Temporary Topics and Queues.....	440
Binding Administered Objects in JNDI.....	440
Using Basic Authentication with JMS.....	441
Using SSL with JMS.....	441
Managing JMS Clients: A Road Map.....	442
Managing JNDI Naming Directories.....	443
Listing the JNDI Providers.....	444
Viewing a JNDI Provider's Properties.....	444
Editing a JNDI Provider's Properties.....	445

Adding a JNDI Provider.....	446
Including additional JAR files required by the JNDI provider.....	448
Managing Connection Factories.....	449
Viewing the List of Connection Factories.....	449
Viewing the Detailed Information About Connection Factories.....	450
Adding Connection Factories.....	452
Modifying Connection Factories.....	454
Deleting Connection Factories.....	455
Managing Cluster Connection Factories.....	455
Viewing a List of Cluster Connection Factories.....	456
Viewing Detailed Information About Cluster Connection Factories.....	456
Adding Cluster Connection Factories.....	459
Load-Balancing Policies for Clusters.....	462
Editing Cluster Configuration.....	463
Modifying Cluster Connection Factories.....	465
Deleting Cluster Connection Factories.....	466
Managing Composite Cluster Connection Factories.....	466
Viewing Detailed Information About Composite Cluster Connection Factories.....	467
Adding Composite Cluster Connection Factories.....	468
Modifying Composite Cluster Connection Factories.....	469
Deleting Composite Cluster Connection Factories.....	470
Managing Destinations.....	470
Binding Destinations to JNDI.....	471
About Creating Destinations.....	472
Creating a JMS Topic From a Naming Directory (JNDI).....	472
Creating a JMS Topic Without Using a Naming Directory (JNDI).....	473
Creating a JMS Queue From a Naming Directory (JNDI).....	474
Creating a JMS Queue Without Using a Naming Directory (JNDI).....	474
Viewing and Editing Destinations.....	476
Deleting a Destination.....	477
Assigning Publish and Subscribe Permissions.....	477
Creating a Durable Subscription.....	478
Exporting and Importing Broker Metadata.....	483
Overview.....	484
Description of Functionality.....	484
Using Command-Line Utilities for Export and Import.....	484
Exporting Broker Data.....	484
Importing Broker Data.....	485
Automating Export and Import with Command Files.....	485
Replicating Broker Metadata.....	485
Building a Production System Using Export and Import.....	486
Importing and Exporting Territories and Clusters.....	486
Exporting Territories.....	487
Importing Territories.....	487

Exporting Territory Gateway Information.....	488
Importing Territory Gateway Information.....	489
Exporting Clusters.....	490
Importing Clusters.....	491
Exporting Cluster Gateway Information.....	491
Importing Cluster Gateway Information.....	492
Exporting, Importing, and Deploying Assets.....	495
Overview.....	496
Exporting and Importing Broker Server and Broker Information.....	496
Asset Types and Dependencies.....	497
Exporting Broker Server and Broker Configuration to a File.....	498
Selecting JNDI Assets for Export.....	500
Importing Broker Server and Broker Configuration from a File.....	501
Exporting and Importing Territory Gateways.....	502
Deploying Broker Assets.....	503
Building Assets for Deployment.....	504
Deploying Assets Using webMethods Deployer.....	504
Partial Deployment.....	505
Checkpoint and Roll Back Assets.....	505
Using Command Central to Manage Broker.....	507
About webMethods Broker Administration.....	508
Configuring Broker Server License.....	508
Configuring SSL in Broker Server.....	509
Retrieving Configuration Details of Broker Server Base Port.....	510
Pausing and Resuming Message Publishing in Broker Servers.....	511
Using the Administration Link of Broker Server.....	512
Configuring the Host and Port of My webMethods Server.....	512
Pre-requisites for Viewing the Broker Server Details Page in My webMethods.....	512
Viewing the Broker Server Details Page in My webMethods.....	513
Administering Broker Server Using My webMethods.....	513
About Administering Broker Server Using My webMethods.....	513
Configuring the Host and Port of My webMethods Server for webMethods Broker User Interface.....	513
Pre-requisites for Viewing the webMethods Broker User Interface in My webMethods.....	513
Viewing the webMethods Broker User Interface in My webMethods.....	514
webMethods Broker and the Command Line Interface.....	514
Commands that webMethods Broker Supports.....	514
Configuration Types that webMethods Broker Supports.....	515
Lifecycle Actions for Broker Server.....	516
Monitoring Run-time Statuses for webMethods Broker.....	517
Monitoring Run-time States for webMethods Broker.....	518
Monitoring webMethods Broker KPIs.....	519
Overview.....	519
Storage Utilization KPI.....	520

Marginal, Critical, and Maximum Values for Broker Server's Storage Utilization.....	520
Storage Utilization Display.....	520
Memory Utilization KPI.....	522
Marginal, Critical, and Maximum Values for Memory Utilization.....	522
Stalled Queues KPI.....	522
webMethods Broker Command-Line Utilities.....	525
Startup and Shutdown Utilities.....	526
startup.....	526
shutdown.....	526
Backup and Restore Utilities.....	526
server_conf_backup.....	527
server_conf_restore.....	528
Broker Server Configuration Utility.....	529
server_config add.....	531
server_config create.....	532
server_config delete.....	538
server_config help.....	538
server_config list.....	538
server_config relocate.....	539
server_config remove.....	540
server_config start.....	541
server_config stop.....	542
server_config stopall.....	542
server_config storage.....	543
server_config update.....	545
Broker Utilities.....	546
broker_buildall.....	547
broker_create.....	548
broker_delete.....	550
broker_load.....	552
broker_ping.....	554
broker_ping Return Codes.....	556
broker_save.....	558
broker_start.....	560
broker_status.....	561
broker_stop.....	562
Directing Command-line Utilities to Use a Non-Default JRE or JDK.....	564
Specifying the JVM Options in Command-line Utilities.....	565
webMethods Broker Storage Utility.....	567
Overview.....	568
Supported Broker Servers.....	568
Processing Modes.....	568
Processing Phases.....	568
Files on which the Utility Operates.....	569

Filenames and Locations for Separate Storage Sessions.....	569
Filenames and Locations for a Combined Storage Session.....	571
Running the Broker Storage Utility in Check Mode.....	572
Running the Broker Storage Utility in Fix Mode.....	573
server_qsck Command Reference.....	574
server_qsck check.....	574
server_qsck fix.....	575
server_qsck help.....	577
server_qsck -version.....	577
Broker SSL Distinguished Name Syntax.....	579
Overview.....	580
Distinguished Name Syntax.....	580
Using Distinguished Names with Broker.....	580
Managing Certificate Files with OpenSSL.....	581
Overview.....	582
Managing Certificate Files.....	582
Generating a Key and Certificate Request.....	582
Viewing the Certificate Request.....	582
Building a Keystore.....	583
Viewing a Certificate in a PKCS#12 File.....	583
Viewing a Certificate in a PEM File.....	583
Building a Trust Store.....	583
Using Access Labels.....	585
Overview.....	586
Types of Access Control.....	586
Per-Message Security.....	586
What Is an Access Label?.....	586
Using Access Labels.....	588
Acquiring an Access Label through Access Label Adapter.....	590
ALA Communication.....	591
Error Handling.....	592
The Access Label Process.....	593
Expiration of Labels.....	594
Behavior Across Broker Boundaries.....	595
Configuration Audit Logging.....	597
Overview.....	598
Audit Log Description and Functionality.....	598
Audit Log File Name and Location.....	598
Audit Log File Size and Cache Size.....	598
Audit Log Categories.....	599
Configure Audit Logging and Identify the Information to Log.....	601
Viewing the Audit Log.....	602

Editing the Audit Log Parameters in the Broker Server Configuration File.....	602
webMethods Broker Document Logging.....	605
webMethods Broker Document Logging.....	606
Logging Utility Client Groups and Clients.....	607
Client ID.....	608
Identifying a Logging Utility Client by Client ID.....	609
Setting Up Document Logging.....	609
Enable Document Logging.....	610
Identify Document Types to Log.....	610
Subscribe to Document Types from Other Brokers in the Territory or Cluster.....	611
Specify User-Defined Document IDs.....	611
Specify Number of Logged Documents to Give to Integration Server at One Time.....	612
Other Configuration Settings.....	613
Logging Utility Management.....	613
Managing the Logging Utility.....	613
Built-In Services.....	614
pub.loggingUtility.util:brokerLoggingLength.....	615
pub.loggingUtility.util:brokerLoggingPeek.....	615
pub.loggingUtility.util:brokerLoggingSingleExtract.....	616
Configuring My webMethods Server to Use the Embedded Database.....	617
Overview.....	618
Running Multiple My webMethods Server Instances.....	618
Configuring My webMethods Server to Use Derby Database.....	619
Configuring to use Derby when you install My webMethods Server.....	619
Configuring to use Derby after you install My webMethods Server.....	619
webMethods Broker Server Configuration Parameters.....	621
Introduction.....	622
Broker Server Configuration Parameters.....	622
Broker Monitor Configuration Parameters.....	637
Introduction.....	638
Broker Monitor Configuration Parameters.....	638

About this Guide

This guide provides information about how to administer and manage webMethods Broker. It describes how to create and manage Brokers on a Broker Server, set up access permissions, and monitor document traffic.

This guide is written primarily for the system administrator who is responsible for configuring and monitoring webMethods Broker.

webMethods software needs to be successfully installed before you can use the procedures described in this guide. For information about installing webMethods Broker, see *Installing webMethods and Intelligent Business Operations Products*.

This guide assumes you are familiar with the following:

- Terminology and basic operations of your operating system (OS)
- Basic concepts of webMethods Broker architecture

This guide also assumes that the systems on which you are running the webMethods Broker software meet or exceed the recommended system requirements outlined in the *webMethods and Intelligent Business Operations System Requirements* document on the Empower Product Support website.

Important: If you have a lower fix level installed, some of the features described in this document might not be available to you. For a cumulative list of fixes and features, see the latest fix readme on the Empower website at "<https://empower.softwareag.com>".

Document Conventions

Convention	Description
Bold	Identifies elements on a screen.
Narrowfont	Identifies storage locations for services on webMethods Integration Server, using the convention <i>folder.subfolder:service</i> .
UPPERCASE	Identifies keyboard keys. Keys you must press simultaneously are joined with a plus sign (+).

Convention	Description
<i>Italic</i>	Identifies variables for which you must supply values specific to your own situation or environment. Identifies new terms the first time they occur in the text.
Monospace font	Identifies text you must type or messages displayed by the system.
{ }	Indicates a set of choices from which you must choose one. Type only the information inside the curly braces. Do not type the { } symbols.
	Separates two mutually exclusive choices in a syntax line. Type one of these choices. Do not type the symbol.
[]	Indicates one or more options. Type only the information inside the square brackets. Do not type the [] symbols.
...	Indicates that you can type multiple options of the same type. Type only the information. Do not type the ellipsis (...).

Documentation Installation

You can download the product documentation using the Software AG Installer. The documentation is downloaded to a central directory named `_documentation` in the main installation directory (SoftwareAG by default).

Online Information

You can find additional information about Software AG products at the locations listed below.

If you want to...	Go to...
Access the latest version of product documentation.	Software AG Documentation website "http://documentation.softwareag.com"

If you want to...	Go to...
<p>Find information about product releases and tools that you can use to resolve problems.</p> <p>See the “Knowledge Center” to:</p> <ul style="list-style-type: none"> ■ Read technical articles and papers. ■ Download fixes and service packs (9.0 SP1 and earlier). ■ Learn about critical alerts. <p>See the “Products area” to:</p> <ul style="list-style-type: none"> ■ Download products. ■ Download certified samples. ■ Get information about product availability. ■ Access older versions of product documentation. ■ Submit feature/enhancement requests. 	<p>Empower Product Support website</p> <p>“https://empower.softwareag.com”</p>
<ul style="list-style-type: none"> ■ Access additional articles, demos, and tutorials. ■ Obtain technical information, useful resources, and online discussion forums, moderated by Software AG professionals, to help you do more with Software AG technology. ■ Use the online discussion forums to exchange best practices and chat with other experts. ■ Expand your knowledge about product documentation, code samples, articles, online seminars, and tutorials. ■ Link to external websites that discuss open standards and many web technology topics. ■ See how other customers are streamlining their operations with technology from Software AG. 	<p>Software AG Developer Community for webMethods</p> <p>“http://communities.softwareag.com/”</p>

1 webMethods Broker

■ Overview	28
■ Publish-and-Subscribe Model	28
■ Broker's Relationship with Other webMethods Components	29
■ webMethods Broker Architecture and Components	31
■ Broker Security Model	42
■ Working with Firewalls	43
■ Enable Full Core Dumps on an AIX System	44

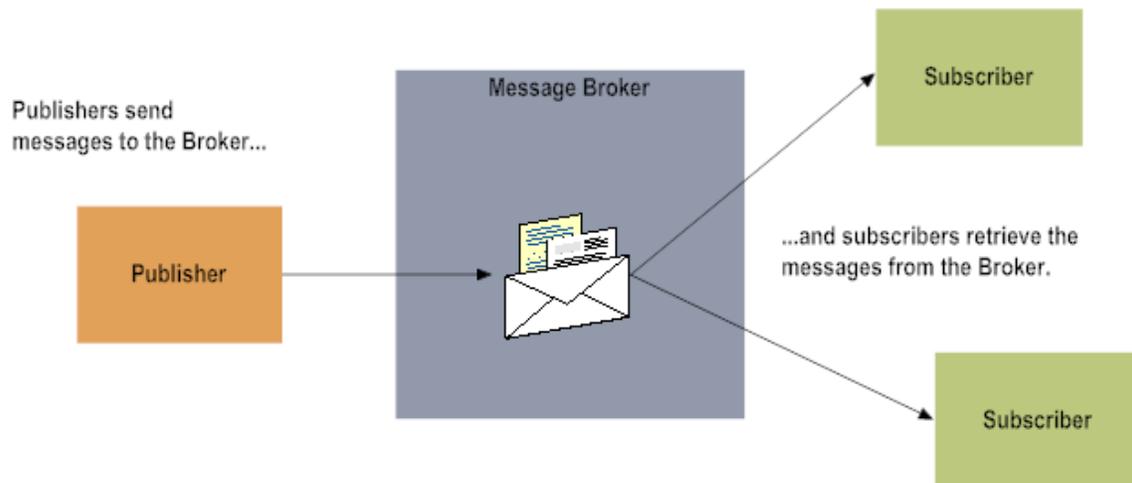
Overview

webMethods Broker is the primary component in what is referred to as the "message backbone" in a webMethods integration environment. Along with other webMethods components, webMethods Broker facilitates asynchronous, message-based integration using the *publish-and-subscribe* model.

Publish-and-Subscribe Model

The publish-and-subscribe model is a specific type of message-based solution in which applications exchange messages (called *documents* in webMethods) through a third entity called *Broker*. In solutions based on this model, applications that produce information (publishers) send the information to the Broker entity and applications that require the information (subscribers) connect to the Broker and retrieve the information they need.

Pub-Sub Application Model



The Broker entity temporarily stores the documents that it receives from publishers in a message queue. Subscribers connect to the Broker entity and fetch documents from the queue. Depending on the application, a subscriber might publish a response after it retrieves and processes a message, but it is not required to do so.

Producers and Consumers Are De-coupled

In the pub-sub model, information producers and consumers are *de-coupled*, meaning they do not interact with one another directly. Instead, each participant interacts only with the message and the Broker entity. This greatly reduces the complexity of an integration solution by eliminating the need to establish individual point-to-point connections among all of the participants and eliminating the need to manage

subscriptions and queues for all those publish points. It also produces solutions that are flexible and easy to extend, because you can easily add or remove producers and consumers without impacting existing participants.

Interaction Is Asynchronous

Participants in a pub-sub solution interact *asynchronously*. A program that produces information does not have to wait for the consumer to acknowledge receipt of that information. It simply publishes the information to the Broker and continues processing (also known as fire-and-forget). Consumers can connect to the message Broker and retrieve the information when they choose. (Which, under some circumstances, might be hours or even days after the document is published.)

Delivery Quality of Service

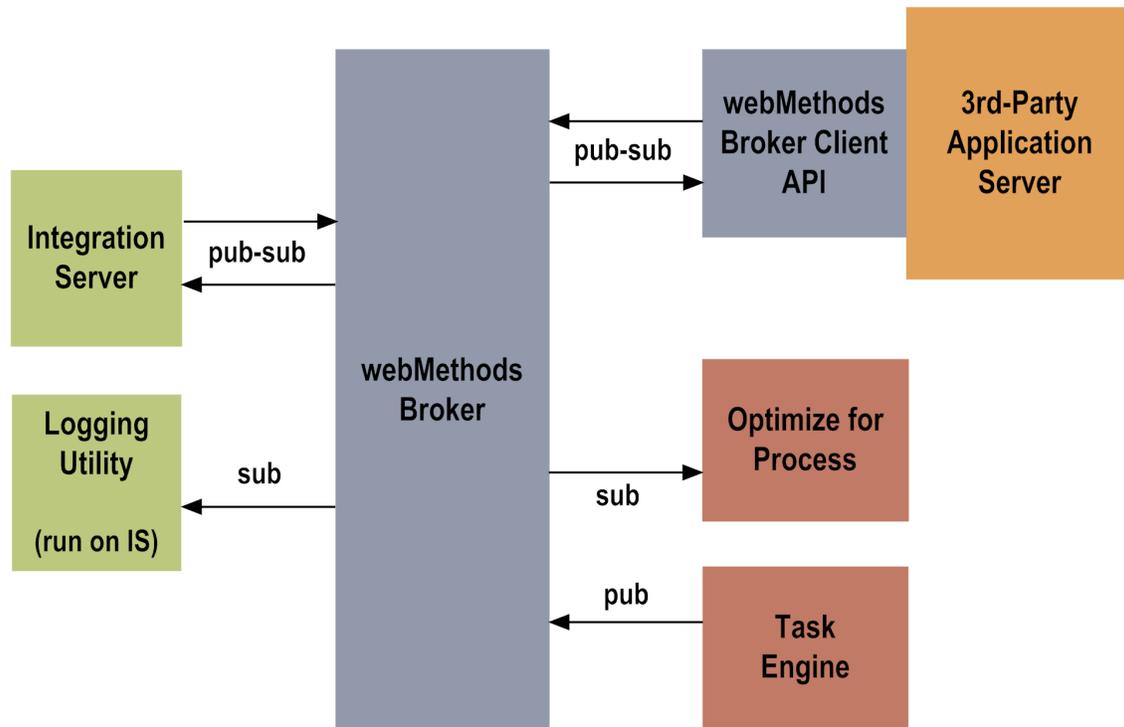
Message Brokers enable messages to be delivered to consumers in a volatile (at most once) or guaranteed (at least once) fashion. When you use webMethods Broker with an Integration Server, you can also specify exactly-once processing to ensure the end integration is executed once and only once. For information about exactly-once processing, see the *Publish-Subscribe Developer's Guide*.

Broker's Relationship with Other webMethods Components

webMethods Broker functions as the message Broker for pub-sub solutions that you develop with webMethods. As shown in the following diagram, Broker provides the pub-sub infrastructure for many webMethods components.

webMethods Broker with ASF support functions as a JMS provider. webMethods Broker supports JMS message producers and consumers (including message-driven beans hosted on supported application servers) to exchange messages. For information about developing pub-sub solutions using webMethods Broker, see *webMethods Broker Messaging Programmer's Guide*.

Many webMethods components interact with the Broker



Components that use webMethods Broker include:

- webMethods Integration Server

Integration Servers host solutions that use the pub-sub model to integrate applications. A pub-sub solution on an Integration Server is made up of *services* and *triggers*. Services are units of logic that publish information to the Broker. Triggers subscribe to documents and invoke services that process the documents. For information about developing pub-sub solutions using Integration Server, see the *Publish-Subscribe Developer's Guide*.

- webMethods Task Engine

webMethods Task Engine can collect task instance metrics in the run-time environment and send this information to webMethods Optimize for Process using the Broker. For information about using Broker for task analytics, see *webMethods Task Engine User's Guide*.

- webMethods Logging Utility

The webMethods Logging Utility is an optional utility that runs on Integration Server. The logging utility retrieves audit information that Broker publishes and moves that information to the central logging database. For information about using the webMethods Logging Utility, see the "[webMethods Broker Document Logging](#)" on page 605.

- webMethods Process Engine

The process run time facility within Integration Server uses webMethods Broker to trigger steps in a business process. For more information about the process run time facility, see *Administering webMethods Process Engine*.

Broker Client APIs

webMethods Broker provides the following APIs for developing your own client programs:

- The webMethods Broker Client API for JMS
- The webMethods Broker Client C# API
- The webMethods Broker Client Java API

webMethods Broker Architecture and Components

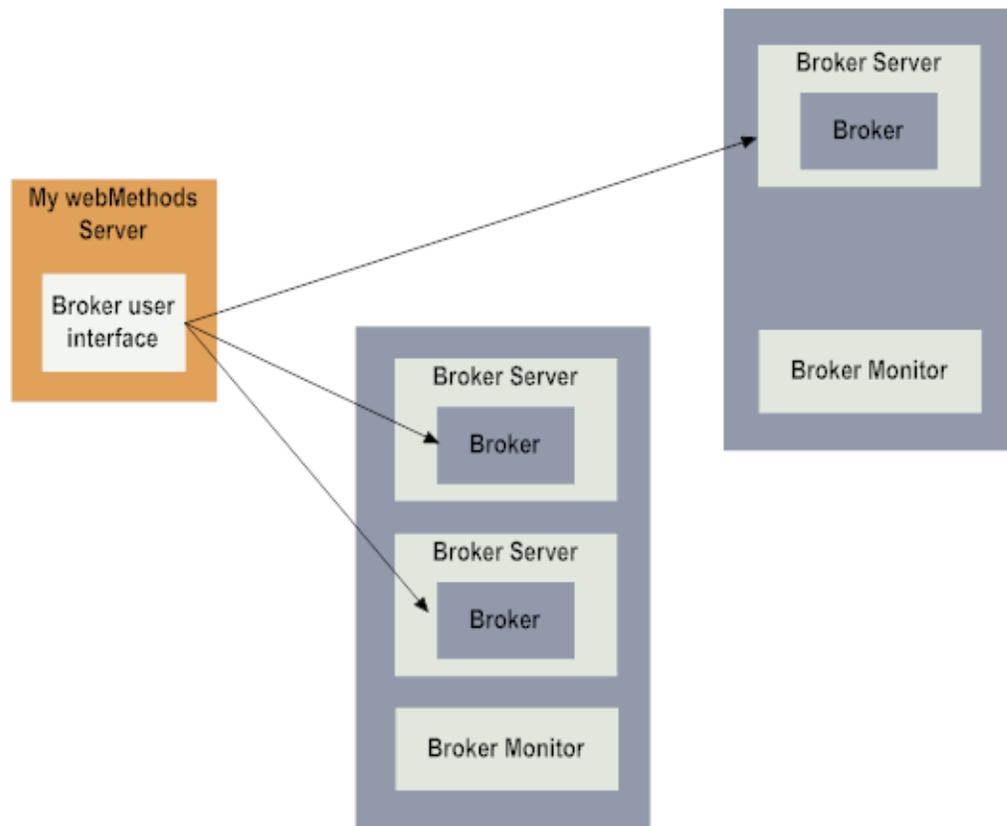
A webMethods Broker environment consists of two main components: *Broker Server*, the run-time component with which publishers and subscribers interact, and the *Broker user interface*, the administrative component that runs on My webMethods Server.

As shown in the following figure, one webMethods Broker environment can contain one or more Broker Servers. The Broker user interface connects to a Broker Server as an administrative client. You can use the same instance of the Broker user interface to administer multiple Broker Servers.

Any machine that hosts a Broker Server will also host a *Broker Monitor*. The Broker Monitor is automatically installed when you install Broker Server.

Broker Monitor monitors all of the Broker Servers running in the webMethods Broker environment. It will automatically attempt to restart any Broker Server that stops running in its environment.

Typical webMethods Broker Environment



A host machine can host multiple webMethods Broker environments. You can install and run more than one instance of webMethods Broker on the same host machine. Refer to the [“Running Multiple Instances of webMethods Broker on the Same Host”](#) on [page 101](#) for more information.

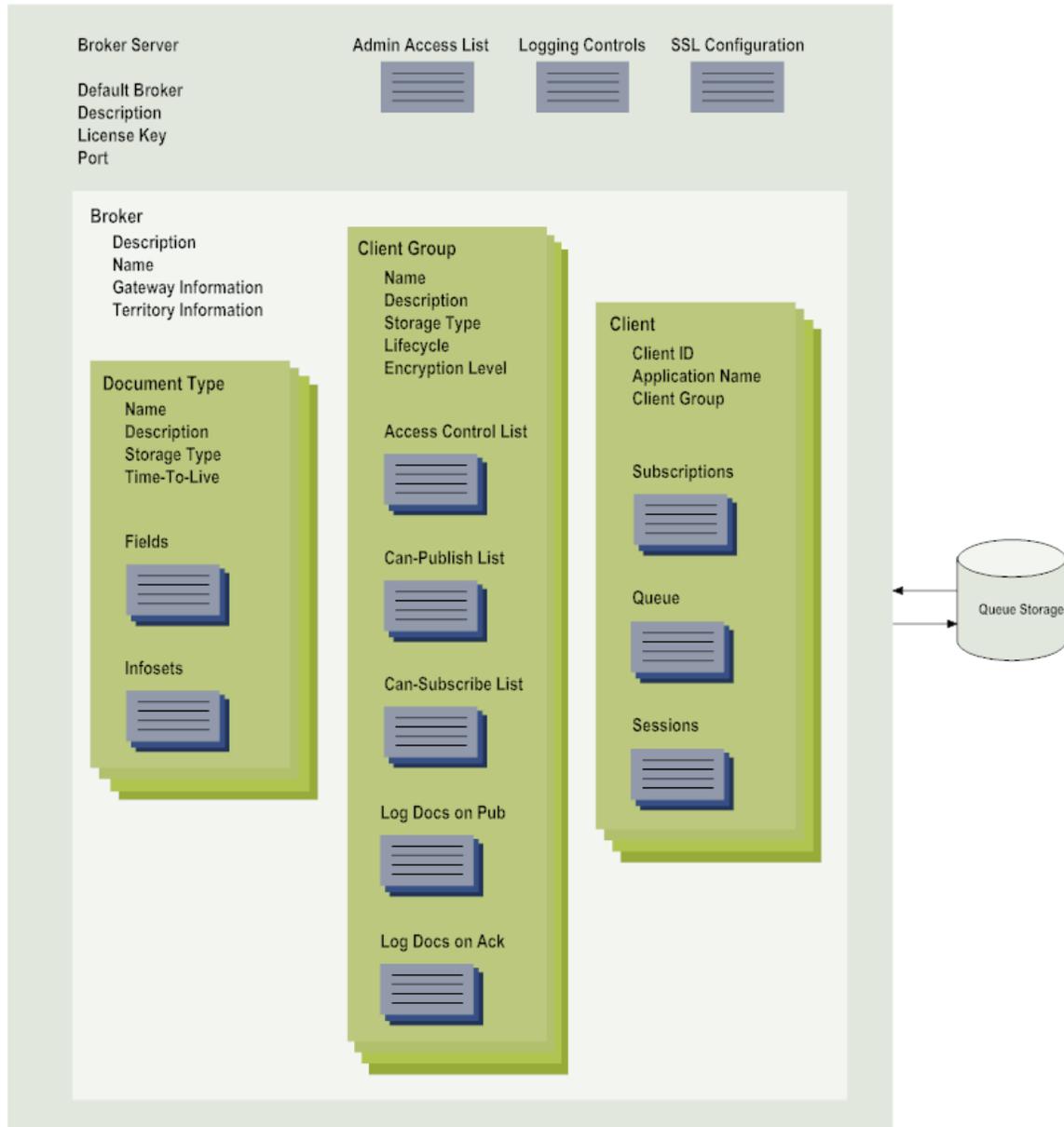
Broker Server

Broker Server is the container within which one or more Brokers reside. A Broker Server performs the communication-related work of receiving client requests, dispatching requests to the requested resource (which in this case, is a Broker), and returning responses to clients. It also manages memory and disk resources for all the Brokers that reside on it.

As an administrator, you will configure port assignments and security settings, specify memory settings, and allocate queue storage for a Broker Server. Generally, you have one Broker Server in a webMethods Broker environment, however, you can run multiple Broker Servers in the same webMethods Broker environment if you choose. If you do this, each Broker Server instance must have its own port and its own queue storage files.

The following depicts the key elements associated with a Broker Server.

Elements Associated with a Broker Server



For information about configuring and managing Broker Servers, see [“Managing Broker Servers”](#) on page 69.

Brokers

A Broker is an entity that resides on a Broker Server. When a client connects to Broker Server, the client specifies the Broker with which it wants to interact.

A Broker encompasses the following types of objects:

- *Document types*, which identify the kinds of documents that the Broker's clients can exchange.
- *Client Groups*, which define specific properties and permissions that Broker applies to clients.
- *Client State Objects*, which maintain information about the individual clients that use the Broker.

These objects are explained in more detail later in this chapter.

A Broker Server is installed with a single Broker, called "default." You can define additional Brokers on a Broker Server if necessary. For example, in the following scenarios, you will find it beneficial to configure multiple Brokers on the same Broker Server:

- If you have a development environment in which you need to test different application environments, you might create multiple Brokers on a single Broker Server instead of installing and configuring separate Broker Server for each individual Broker environment that you need to emulate.
- If you are using a Broker for publish-subscribe, and another Broker for JNDI, you can host both these Brokers on the same Broker Server. As both the publish-subscribe Broker and the JNDI Broker will be working together, hosting them on the same Broker Server makes it easy to manage the Broker Server instance.

When you configure multiple Brokers on a Broker Server, you can designate one of the Brokers to act as the *default Broker*. The default Broker is the one to which Broker Server will connect any client that does not explicitly specify the name of the Broker that it wants to use.

Following are the limitations when multiple Brokers are configured on the same Broker Server:

- Only the Broker Server runs as an *Operating System process*. The Brokers are just logical entities that reside on the Broker Server. All Brokers hosted on the Broker Server share the system resources such as disk space, CPU, and memory.
- If any of the Brokers require maintenance, all the other Brokers installed on the Broker Server are also affected.
- If you are using multiple Broker Servers, maintaining these Broker Servers will cause overhead as you will have to use multiple storage directories and ports.

For information about creating, configuring, and managing Brokers, see [“Managing Brokers” on page 137](#).

Broker Monitor

Broker Monitor is a program that executes along side Broker Server on the host machine in a webMethods Broker environment. The Broker Monitor is automatically installed when you install webMethods Broker.

Broker Monitor continually checks the state of the Broker Server and automatically attempts to restart it if it stops running.

Broker Monitor is also the program that starts a Broker Server. You usually do not start a Broker Server directly. Instead, you ask the Broker Monitor to start it for you.

Broker Monitor monitors (and starts) all Broker Servers in the webMethods Broker environment.

It is possible to have more than one Broker Monitor on a host machine if the host machine has more than one webMethods Broker environment. For each webMethods Broker environment there is only one Broker Monitor.

For more information about Broker Monitor, see [“ Broker Monitor ” on page 57](#).

Broker User Interface

You use the Broker user interface to configure, monitor, and manage one or more Broker Servers and the Brokers that they host.

The Broker user interface is a portal application that executes on My webMethods Server. It enables you to manage webMethods Broker from any browser-equipped computer in your organization's network.

For information about working with the Broker user interface, see [“Using My webMethods to Administer webMethods Broker ” on page 45](#).

Note: You can use the Broker user interface to perform most administrative tasks for webMethods Broker. Some tasks, however, must be performed using command-line utilities that reside on the Broker Server host machine. If a task can only be accomplished through a command-line utility, this guide will describe how to use that utility. For a list of the command-line utilities, see [“ webMethods Broker Command-Line Utilities ” on page 525](#).

Document Types

A document type is a schema-like definition that represents a kind of document that publishers and subscribers can exchange using the Broker. A document type has a name, a structure that consists of named fields, and a set of properties that determines how the Broker handles instances of that document type at run time.

A document type must be registered on the Broker before clients can publish instances of that type. The name of the document type, which must be unique within a Broker, is carried by all documents of its type. Subscribers indicate which documents they want to receive by subscribing to specific document types.

Document types are created when a developer uses Software AG Designer to define *publishable document types*. When a developer creates a publishable document type, the Integration Server to which that developer is connected automatically registers a

corresponding document type on the Broker. You can also use the Broker admin APIs to create document types.

Additional document types are created by Broker itself to support underlying system processes and by other webMethods components (such as Designer) that use the Broker's pub-sub infrastructure.

You use the Broker user interface in My webMethods to examine the document types that are registered on a Broker and to monitor the activity associated with each type. You can also use the Broker user interface to export a document type from one Broker and import it on another.

For information about managing document types, see [“Managing Document Types” on page 157](#).

Note: In older versions of webMethods Broker, document types were called *event types* and instances of document types were called *events*. You will still see these terms in documentation that relates to the underlying Broker API.

Client Groups

To connect to a Broker, a client program must declare the name of the *client group* to which it belongs. Conceptually, a client group represents a particular group or category of client (e.g., administrators, customers, Integration Server processes). On the Broker, a client group is represented by a client group object. This object has a name and a set of properties. The Broker applies these properties to clients that declare themselves to be members of that group.

A client states its group membership when it initially connects to a Broker. Once a client connects to a Broker and its client state object is created, the client cannot change its client group membership.

Some of the properties in a client group determine how the Broker interacts with a client. For example, the Lifecycle property determines whether the Broker maintains state information for a client after it disconnects. Other properties determine which document types a client is permitted to publish and to which document types it can subscribe. Additional properties specify which document types the client can log to the audit subsystem.

Client groups also play a key role in Broker security. If you enable SSL or basic authentication and require clients to authenticate themselves, the access control list in the client group determines which clients are authorized to connect as members of that particular group.

As an administrator, you will create and configure client groups using the Broker user interface. (You can also create client groups using the Broker admin API.) For more information about managing client groups, see [“Managing Client Groups” on page 171](#).

Clients (Client State Objects)

When a client initially establishes a connection with a Broker, the Broker generates a *client state object* for that client.

The client state object maintains the following information about the client:

- **Client ID.** An ID that uniquely identifies the client.
- **Application Name.** An optional label that client programs can use to identify the system or application to which they belong.
- **Client Group.** The name of the client group to which the client belongs.
- **Subscription List.** The list of document types this client wants to receive.
- **Queue.** The list of document instances that are currently enqueued for the client.
- **Sessions.** A client can optionally enable "shared-state" when it connects to a Broker. The shared-state property enables multiple clients to connect to the Broker using the same client state object. (Typically, a client application does this to enable multiple client processes to retrieve and process documents from the client's queue, thereby improving performance.) When shared state is enabled, the client state object contains a list of sessions. Each session identifies the IP address of a client that is currently connected to the Broker using that client state object. For example a trigger on a cluster of two Integration Servers appears as one client on the Broker (that is, they share the same client state object), but the client state object has two sessions, one for each Integration Server.

When a client disconnects from Broker, the disposition of its client state object is determined by the lifecycle property in the client group to which the client belongs.

If the lifecycle value is...	The Broker...
Explicit Destroy	Maintains the client state object after the client disconnects. The client state object continues to receive documents in its queue, even though the client program is not actively connected to the Broker. The client can subsequently reconnect (using the same client ID) and retrieve documents from the queue.
Destroy on Disconnect	Drops the client state object. When the client state object is dropped, the Broker loses all knowledge of that client.

You use the Broker user interface to examine the client state objects that represent the clients that are using the Broker. You can also use the Broker user interface to forcibly disconnect a client from a Broker and to instantiate a test client for diagnostic purposes.

For information about monitoring and managing Broker clients, see [“Managing Clients” on page 187](#).

Subscriptions

To receive documents, a client must register a subscription on the Broker. A subscription identifies the type of document the client wants to receive. If the client wants to receive only certain instances of a particular type, it can attach a filter to the subscription. When a subscription includes a filter, the subscriber receives only the documents that satisfy the filter criteria.

Subscriptions are registered by the client programs that interact with the Broker. When a developer creates a trigger on an Integration Server, for example, the Integration Server registers a subscription on the Broker for each document type associated with the trigger.

When Broker receives a document for which there are no subscribers, it puts the document in the dead letter queue (if one is configured). If a dead letter queue has not been configured (which is the installed behavior), the document is discarded.

You can use the Broker user interface to view the subscriptions for a client and to delete a subscription. You can also use the Broker user interface to discover which clients subscribe to a particular document type.

For more information, see [“Managing Subscriptions” on page 199](#) and [“Viewing Which Clients Subscribed to a Document Type” on page 167](#).

Queues

The Broker places documents that match a client's subscription in the client's queue. Each client has one queue, which is part of its client state object. A document remains in the queue until the client retrieves it (and acknowledges that it has retrieved the document successfully) or until the document expires. To reduce memory usage, volatile documents that have expired can be proactively deleted at regular intervals, based on the size of the queue, from the client queues and forward queues before the client tries to retrieve them.

When the Broker queues a document for a client, it does not actually place a copy of the document in the queue. The Broker maintains only one copy of a document instance. Queues belonging to the subscribers of that document's type contain pointers to that instance.

You can use the Broker user interface to view the list of documents in a client queue and to examine the content of the documents themselves. You can also use the Broker user interface to delete documents from a queue, move documents from one queue to another, and change the order of the documents in a queue.

For information about managing queues, see [“Managing Forwarding Queues” on page 245](#).

Storage Type

Documents published to Broker can be stored as *volatile documents* or *guaranteed documents*. Volatile documents are stored only in memory and are lost if the host machine loses power or the Broker Server is otherwise restarted. Guaranteed documents are persisted to disk so that they can be recovered in case of a power failure or a restart.

Whether the Broker treats a document as volatile or guaranteed depends on the storage type associated with the document's type and the storage requirements of the client (as specified by the "Queue Type" property in the client's associated client group).

For additional information about storage type, see [“Document Type Storage Types” on page 161](#) and [“Client Queue Storage Type” on page 174](#).

Queue Storage (QS)

Queue storage refers to the file storage that Broker Server uses to persist guaranteed documents and non-volatile Broker metadata (e.g., Broker definitions, document type definitions, client group definitions) to disk.

Queue storage consists of two files: a *log file*, which is a fixed-sized file that Broker can write data to quickly, and one or more *storage files*, into which data is ultimately stored using a logged-commit process. Queue storage is shared by all Brokers on a Broker Server.

Queue storage can be configured to operate in *combined mode* or *separate mode*. In combined mode, Broker metadata and run-time data are stored in the same log and storage files. In separate mode, the Broker uses two log files and two storage files. It stores metadata in one log/storage pair and run-time data in the other log/storage pair. When queue storage operates in separate mode, you can use the Broker Server's online backup utility to backup metadata while the Brokers on that server continue running.

As an administrator you can configure the size of the queue-storage files and monitor their usage. For information about managing queue storage, see [“Configuring Queue Storage” on page 84](#).

Territories

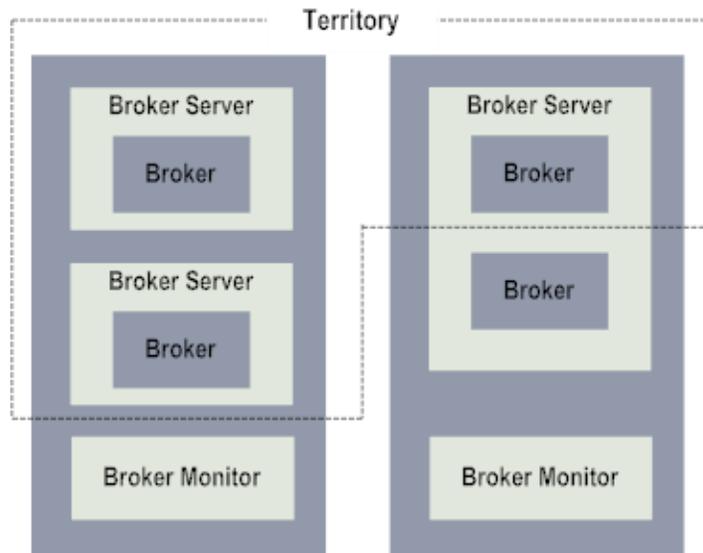
Brokers can be linked to form units known as *territories*. Brokers that form a territory function as a single logical Broker. Clients that connect to one Broker in a territory can automatically exchange documents with clients connected to any of the Brokers in the territory.

Territories support scalability by allowing you to segment traffic among multiple Brokers, while still treating the Brokers as a single operational unit.

When Brokers operate in a territory, they maintain the same set of document types and client group definitions. (These objects are initially synchronized when a Broker joins

a territory.) Any changes that occur to these objects on one Broker, are automatically propagated to the other Brokers in the territory.

The Brokers that are members of a territory can reside on any Broker Server. The following figure shows a territory that is made up of three Brokers on two Broker Servers.



You use the Broker user interface to form territories and to monitor traffic that flows among the Brokers that make up the territory.

For information about working with territories, see [“Managing Territories” on page 353](#).

Territory Gateways

Territory gateways are links that you establish between territories. A territory gateway enables clients in one territory to receive documents that are published in another territory. Territory gateways are often used to connect territories that are geographically distant or whose clients share only a limited number of document types.

Unlike territories, whose Brokers operate as a single unit and share the same document types and client groups, a gateway is simply a bridge that enables instances of specific document types to travel from one territory to another. (Message subscriptions are shared dynamically/implicitly in a territory and statically/explicitly across a gateway connection.)

You use the Broker user interface to create territory gateways and to specify which document types can travel across them.

For information about working with territory gateways, see [“Managing Territory Gateways” on page 367](#).

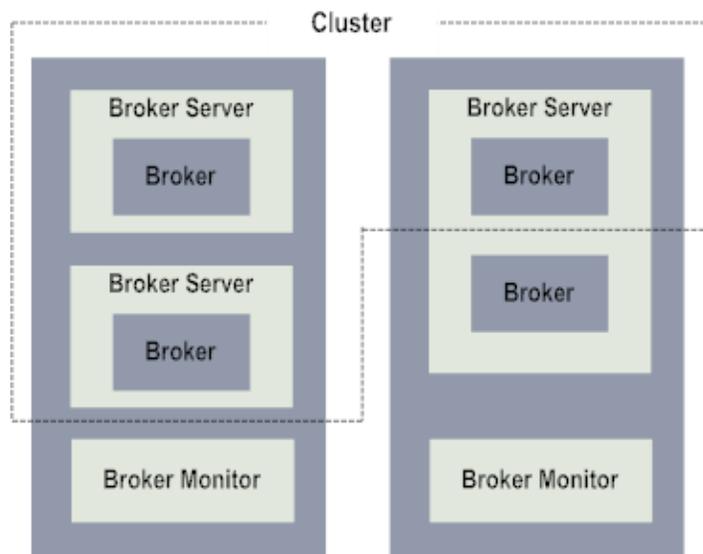
Clusters

Brokers can also be linked to form units known as *clusters*. Brokers that form a cluster function as a single logical Broker. All Brokers in a cluster maintain the same set of document types and client groups. However, each Broker supports its own set of clients. In a cluster, published documents are not forwarded to other Brokers (peers).

Clusters support client-side load balancing and failover by allowing you to use JMS clustering policies to distribute message traffic between multiple Brokers, while still treating the Brokers as a single operational unit.

When Brokers operate in a cluster, they maintain the same set of document types and client group definitions. (These objects are initially synchronized when a Broker joins a cluster.) Any changes that occur to these objects on one Broker, are automatically propagated to the other Brokers in the cluster.

The Brokers that are members of a cluster can reside on any Broker Server. The following figure shows a cluster made up of three Brokers on two Broker Servers.



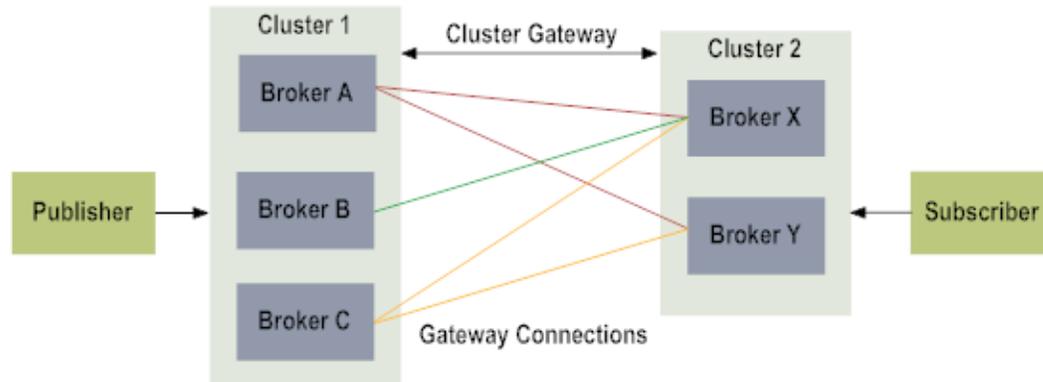
For information about working with clusters, see [“Managing Clusters” on page 397](#).

Cluster Gateways

A *cluster gateway* enables two clusters to exchange documents. Unlike clusters, whose constituent Brokers automatically share their document types and client groups, a cluster gateway requires an administrator to explicitly specify which document types the clusters are permitted to send and receive. Moreover, the clusters never share client group information across a gateway.

Conceptually, Broker cluster gateway is conceptually similar to a territory gateway except that multiple gateway connections are supported between two clusters. Multiple

gateway connections are required to support failover in case of Broker failure or network outage as well as to support multiple redundant paths for the multisend load balancing policy.



For information about working with cluster gateways, see [“Managing Cluster Gateways” on page 405](#).

Broker Security Model

You can secure webMethods Broker using basic authentication and SSL authentication. With basic authentication, clients must present the user name and password. With SSL authentication, clients must present digital certificates that verify their identity.

For additional security, you can encrypt the following connections configured with:

- Two-way SSL authentication
- One-way SSL authentication
- Basic authentication over one-way SSL

When you use SSL for encryption, Broker Server uses standard public/private key protocol to encrypt communications between it and the client, which ensures data confidentiality and integrity.

Broker supports certificate revocation checking by using *certificate revocation lists* (CRLs). A certificate revocation check against a list of revoked certificates provides a mechanism for protecting against using certificates that are compromised or no longer in effect.

Broker supports *Federal Information Processing Standards* (FIPS). In accordance with FIPS 140-2 Implementation Guidance, Section G.5, webMethods Broker uses an embedded FIPS 140-2-validated cryptographic module (Certificate #1051) running on all platforms that it supports.

Broker also supports access control lists (ACLs) which enable you to restrict access to resources on the Broker to specified clients. To enforce ACLs, you must enable basic authentication or SSL authentication, and clients must present credentials to prove their identity.

For information about configuring security, see [“Managing Broker Security” on page 287](#).

Working with Firewalls

You can implement webMethods Broker behind a firewall to help preserve the integrity of your network. To enable webMethods Broker to work through a commercial firewall, open the port used by the Broker Server through the firewall. Opening the Broker Server's port enables users outside the firewall to run your webMethods Broker-based applications without compromising on the network integrity. webMethods Broker administrator requires access to the Broker Server's port for webMethods Broker administration. It is recommended that you use SSL in conjunction with a firewall so that you prevent administrative access from a public network.

Accessing webMethods Broker through a Firewall

If webMethods Broker is behind a firewall, to administer webMethods Broker using My webMethods, My webMethods requires access to Broker Server port (default is 6849) and Broker Monitor port (default is 6850).

Depending on what capabilities you want to allow, open all or some of these webMethods Broker ports in your firewall.

Connection	Port
Non-SSL connection	Broker Server port. The default port is 6849.
One-way SSL connection	Broker Server port -1. For example, port 6848 if the Broker Server port is 6849.
Two-way SSL connection	Broker Server port -2. For example, port 6847 if the Broker Server port is 6849.
Parallel channel for Non-SSL connection	Broker Server port -3. For example, port 6846 if the Broker Server port is 6849.
Parallel channel for SSL connection	Broker Server port -4. For example, port 6845 if the Broker Server port is 6849.

Enable Full Core Dumps on an AIX System

On AIX systems, you must enable full core dumps. Using the AIX System Management Interface Tool (SMIT), run the command `smitty chgsys` and set `Enable full CORE dump` to `true`.

2 Using My webMethods to Administer webMethods Broker

■ Managing Broker Using My webMethods	46
■ Who Can Use the Broker User Interface?	46
■ Accessing the Broker User Interface	47
■ Logging Out of My webMethods	48
■ Adding Broker Servers to My webMethods	49
■ Removing Broker Servers from My webMethods	52
■ Configuring the Connection Parameters	53
■ Using My webMethods with ACL-Protected Broker Servers	54

Managing Broker Using My webMethods

You use the Broker user interface in My webMethods to manage and configure webMethods Broker. To access the Broker user interface, select the **Administration** function on the My webMethods Server home page and then click the **Messaging** tab.

Note: Although you can perform most administrative tasks using the Broker user interface, certain tasks require you to use the Broker command-line utilities. These utilities reside on the Broker Server host machine. This guide provides procedures for using these utilities when a task requires them. For a complete list of command-line utilities, see “[webMethods Broker Command-Line Utilities](#)” on page 525.

Selecting a Database

If you are using My webMethods Server 10.1 and earlier versions to run the Messaging user interface in My webMethods, then you can configure My webMethods Server using the embedded database, Apache Derby, instead of an external RDBMS. For more information about using the embedded Apache Derby database, see “[Configuring My webMethods Server to Use the Embedded Database](#)” on page 617.

Note: You cannot configure My webMethods Server using the embedded database, Apache Derby, in My webMethods Server 10.3 and later. For more information, see, *Administering My webMethods Server*.

If you use My webMethods to administer Software AG webMethods products other than webMethods Broker, you must use an external RDBMS. For complete information about configuring My webMethods Server to use an external RDBMS, see *Installing webMethods and Intelligent Business Operations Products*.

Using My webMethods with Earlier Versions of Broker

You can use the Broker user interface in My webMethods to manage Broker Servers running version 6.x or later. When you use it with earlier versions of Broker Server, certain features in the user interface might not be available. If the Broker Server does not support a particular feature, you will not be permitted to use that feature in My webMethods.

Who Can Use the Broker User Interface?

When you install the Broker user interface on My webMethods Server, only members of the "My webMethods Administrators" role can access it.

Extending Access to Other Users

To enable other users to access the Broker user interface (that is, users who are not members of My webMethods Administrators), you must take the following general steps on the My webMethods Server.

Note: You must be a member of My webMethods Administrators to perform these steps.

1. Define a role (or select an existing role) that represents the group of users who will be authorized to use the Broker user interface on this My webMethods Server.
2. Give this role access privileges to the Messaging function. If you want to restrict the role to certain activities within the Broker user interface, enable the specific Messaging subtasks that the users belonging to this role are permitted to perform.
3. To this role, add the users who will be permitted to use the Broker user interface.
4. For more information about creating roles and assigning access privileges to them, see the *Administering My webMethods Server* guide.

Accessing the Broker User Interface

To access the Broker user interface, you need the following:

- A Web browser.
- The URL of the My webMethods Server that hosts the Broker user interface.
- A user account on that My webMethods Server. Your account must have access privileges for the Messaging function.

Important: Because My webMethods uses JavaScript, you must ensure that your Web browser is set up to allow JavaScript to execute. For more information about how to verify that your Web browser settings allow JavaScript, see the help for your Web browser.

Note: The functionality available to you in the Broker user interface is based on the privileges assigned to your roles in My webMethods. If your roles do not have all Messaging privileges, the Broker user interface might not display pages required to perform some Broker actions. If a procedure instructs you to use an item that is not available, contact your system administrator about acquiring the proper privileges to allow you to perform the action.

To access the Broker user interface

1. In your Web browser, enter the URL where My webMethods Server is hosted.

Example: atlas:8080

- At the log on screen, enter your My webMethods user name and password and click **Login**. If you just installed My webMethods Server, you can use the default administrator account as follows:

For this parameter...	Enter...
User Name	Administrator
Password	manage

Important: To keep My webMethods Server secure, you should immediately change the default administrator password after you install My webMethods Server. For instructions about how to change the password, see *Administering My webMethods Server*.

- Click **Messaging** to access the Broker user interface.
 - If this is the first time you have used the Broker user interface, the interface displays an empty Broker Server List. You must populate this list with the Broker Servers that you want to manage. For procedures, see [“Adding Broker Servers to My webMethods” on page 49](#).
 - If you have already populated the Broker Server List, the Broker user interface will connect to each Broker Server in the list and displays its status.

How My webMethods Interacts with Broker Servers

The Broker user interface in My webMethods operates as a client of the Broker Server or Broker that you are using it to manage. When it connects to a Broker, it does so as a member of the Broker's "admin" client group.

When you display the list of clients associated with a Broker, that list will include a client that represents your own session on My webMethods (that is, it represents the connection that My webMethods has established with the Broker on your behalf). Clients that are generated by the Broker user interface in My webMethods have the application name "webMethods Messaging." The ID that appears beside the application name identifies to which My webMethods user the connection belongs.

Logging Out of My webMethods

When you log out of My webMethods, all connections that it made with Broker Servers and Brokers on your behalf are closed.

If you do not explicitly log out of My webMethods, your connections will remain in place until your browser session times out, or until you restart the My webMethods Server that hosts the Broker user interface.

To log out of My webMethods

Click the **Logout** link, which is located at the top of all My webMethods pages.

Adding Broker Servers to My webMethods

To manage a Broker Server, you must "add" the Broker Server to your *Broker Server List*. This list identifies the set of Broker Servers that you want to manage. When you open the Broker user interface, it automatically connects to the Broker Servers specified in your Broker Server List and displays their status.

Note: The Broker Server List you create is associated with your user account and appears only when you log on to My webMethods. Other administrators that use My webMethods to manage webMethods Broker must define their own Broker Server Lists.

You can add Broker Servers to your Broker Server List in the following ways:

- Specify the URL of an individual Broker Server.
- Discover all the Broker Servers on a specified host machine.
- Upload Broker Server definitions from a file.

After you add Broker Servers to your Broker Server List, you can optionally use the procedures in [“Configuring the Connection Parameters” on page 53](#) to specify the way in which you want the Broker user interface to connect to the Broker Servers.

Adding an Individual Broker Server to My webMethods

If you know the URL (including the port number) of the Broker Server that you want to add, use the following procedure to add it to your Broker Server List in My webMethods.

The Broker Server that you want to add must be running when you perform the following procedure.

To add an individual Broker Server to the Broker Server List

1. In My webMethods: **Messaging > Broker Servers > Servers**.
2. Click **Add**.
3. Specify the following:

In this field...	Specify...
Host Name	<p>The name of the host machine on which the Broker Server is installed.</p> <p>Note: Although you can specify a numeric IP address in this field, we recommend that you identify the Broker Server by its host name instead. The use of host names makes it easier to distinguish servers in the Broker Server List.</p> <p>Example: atlas</p>
Port	<p>The port number on which the Broker Server is configured to listen for client requests. You must specify a port number even if the Broker Server is using the default port.</p> <p>Example: 6849</p>

4. Click **Add**.

Discovering Broker Servers on a Specified Broker Monitor Host Machine

If you know the name of the Broker Monitor host machine on which one or more Broker Servers are running, you can use the following procedure to add the Broker Servers to the Broker Server List in My webMethods.

If multiple Broker Servers are running on the specified Broker Monitor host machine, this procedure enables you to add all of them to the Broker Server List in one step.

The Broker Monitors and the Broker Servers that you want to add must be running when you perform this procedure.

To discover and add Broker Servers running on a specified host machine

1. In My webMethods: **Messaging > Broker Servers > Servers**.
2. Click **Add**.
3. Select the **Discover Broker Server** tab.
4. Specify the name of the Broker Monitor host machine on which the Broker Servers are installed.

For example: atlas

Note: Although you can specify the host machine's numeric IP address, Software AG recommends that you identify it by name instead. The use of host names makes servers easier to distinguish in the Broker Server List.

5. Specify the port on which the Broker Monitor is runs. The Broker Server List will contain only the Broker Servers configured under the port you specify. If a port is not specified, the default port (6850) is used.

If you do not know the port number, you can look it up using the `server_config` utility. See [“Viewing the Port Setting for Broker Monitor ” on page 66](#) for more information.

6. Click **Discover**.
7. When the list of Broker Servers appears, select the servers that you want to add to your Broker Server List and click **Add to List**.

Uploading a List of Broker Servers into My webMethods

You can also add multiple Broker Servers to your Broker Server List in a single step by creating a file that defines the locations of the Broker Servers and importing the definitions into My webMethods. This procedure is especially convenient if multiple administrators will manage the same set of Broker Servers. It enables you to define a single set of definitions that each administrator can use to populate his or her Broker Server List.

To use this approach, you must create a definitions file and upload the definitions file into My webMethods.

Creating the Definitions File

To create the definitions file

1. Use a text editor to create a file that identifies the Broker Servers that you want to add to your Broker Server List. For each Broker Server, identify the host computer on which the Broker Server runs and the port number to which it is assigned. If a Broker Server uses the default port, you can omit the port number. If the Broker Server is monitored by a Broker Monitor on a non-default port, you must specify the port. Separate each entry with a carriage return (as shown below), a comma, a space, a tab, or a semicolon.

```
HostMachineName1:Broker Server port
HostMachineName2
HostMachineName3:Broker Server port
HostMachineName4
HostMachineName5:Broker Server port, Broker Monitor Port
HostMachineName6, Broker
Monitor Port
```

For example:

```
tokyo:9000
sanfrancisco
paris:7000
```

```
newyork  
london:8000, 6866  
hongkong,6868
```

2. Save the file as an ASCII text file. When you save the file, be sure to do the following:
 - Give the file a .txt extension. The file must have this extension for My webMethods to import it.
 - Save the file to a location that you can access from your browser.

Uploading the Definitions File to My webMethods

Use the following procedure to import the definitions into your Broker Server List.

Note: This procedure will prompt you to select which Broker Servers you want to add to your Broker Server List. It does not automatically add all of the Broker Servers in the file.

The Broker Servers that you want to add to your Broker Server List must be running when you perform this procedure.

To upload Broker Server definitions into My webMethods

1. In My webMethods: **Messaging > Broker Servers > Servers**.
2. Click **Add**.
3. Select the **Upload Broker Server** tab.
4. Use the **Browse** button to select the file that contains the Broker Server definitions.
5. Click **Upload**.
6. When the list of Broker Servers appears, select the servers that you want to add to your Broker Server List and click **Add to List**.

Removing Broker Servers from My webMethods

Use the following procedure to remove a Broker Server from your Broker Server List in My webMethods.

Note: Removing a Broker Server simply removes the server from the Broker Server List in My webMethods. It does not remove or uninstall the Broker Server from its host machine.

To remove a Broker Server from the Broker Server List

1. In My webMethods: **Messaging > Broker Servers > Servers**.
2. In the Broker Server List, select the check box beside the Broker Server that you want to remove.

3. Click **Delete**.

Configuring the Connection Parameters

You use the connection parameters to control how My webMethods connects to the Broker Servers that you want to manage. These parameters determine timeout periods, reporting intervals, and encryption requirements.

Note: The connection parameters are associated with *your* My webMethods user account. Other administrators that use My webMethods to manage Broker Servers must configure their own parameter settings.

Parameter Name	Description
How long to wait for a new connection to a Broker Server	<p>The period of time (in seconds) My webMethods waits to establish a connection with a Broker Server before it determines that the Broker Server is not responding.</p> <p>Servers that do not respond within the specified time are denoted with an  error icon in the Broker Server List.</p> <p>Min: 0</p> <p>Max: 2,147,483,647</p> <p>Default value: 30</p>
How long to wait for Broker Client connections	<p>The period of time (in seconds) My webMethods waits to establish a connection with an individual Broker before it determines that the Broker is not responding.</p> <p>Brokers that do not respond within the specified time period are denoted with an  error icon in the Broker List.</p> <p>Min: 0</p> <p>Max: 2,147,483,647</p> <p>Default value: 30</p>
Enable Statistical Polling	<p>When enabled, the Time interval between statistical refresh field is available for specifying a value.</p> <p>If you do not select Enable Statistical Polling, the Total Deliveries will be available instead of Recent Deliveries in the Broker Server Details page.</p>

Parameter Name	Description
Time interval between statistical refresh	<p>The period of time (in minutes) over which "recent" statistics are calculated. This period is also known as the <i>statistic collection interval</i>.</p> <p>This field is available for input only when Enable Statistical Polling is selected.</p> <p>For example, if you set this parameter to 10 minutes, the value in the Recent Deliveries field on the Broker page represents the number of documents received during the previous 10 minutes.</p> <p>Min: 0</p> <p>Max: 2,147,483,647</p> <p>Default value: 1</p> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> <p>Note: When you display a page that reports recent statistics, My webMethods displays the statistic's current value. These statistics are not updated unless you click Refresh to update the display.</p> </div>

Viewing and Editing the Connection Parameters

Use the following procedure to examine or set the connection parameters. When you modify the connection parameters, the changes you make take effect immediately.

To view or edit the connection parameters

1. In My webMethods: **Messaging > Settings**.
2. Select the **Connection** tab.
3. Modify the parameters as necessary, then click **Apply**. For a description of the connection parameters, see ["Configuring the Connection Parameters" on page 53](#).

Using My webMethods with ACL-Protected Broker Servers

If you are using My webMethods to manage a Broker Server that is secured with an access control list (ACL), you must assign an *identity* to the Broker user interface.

For basic authentication, an identity is composed of user name and password.

For SSL authentication, an identity is composed of:

- A signed certificate, stored in a password-protected certificate file called a *keystore*.

- The trusted root of the certificate issuer (or authenticator), stored in a certificate file called a *trust store*.

Note: The ACL configuration of a Broker Server is deleted when you edit the SSL identity of that Broker Server. You have to reconfigure the ACL to include the newly edited Broker Server.

For information about assigning a basic authentication identity to the Broker user interface in My webMethods, see [“Configuring a Basic Authentication Identity for the Broker User Interface Component” on page 347](#).

For information about assigning an SSL identity to the Broker user interface in My webMethods, see [“Configuring an SSL Identity for the Broker User Interface Component” on page 348](#).

Note: If you do not configure an identity for the Broker user interface, or if you specify an identity that does not have administrative permissions for a particular Broker or Broker Server, you can still view some information about that Broker or Broker Server. You will not, however, be able to modify any settings or see all information of the protected resource.

3 Broker Monitor

■ The Role of Broker Monitor	58
■ Starting Broker Monitor	60
■ Stopping Broker Monitor	62
■ How to Secure Broker Monitor Access	64
■ How to Determine Whether Broker Monitor Is Running	64
■ Ports and Running Multiple Instances of Broker Monitor	65
■ Configuring Broker Monitors to Start Automatically on UNIX Systems	67

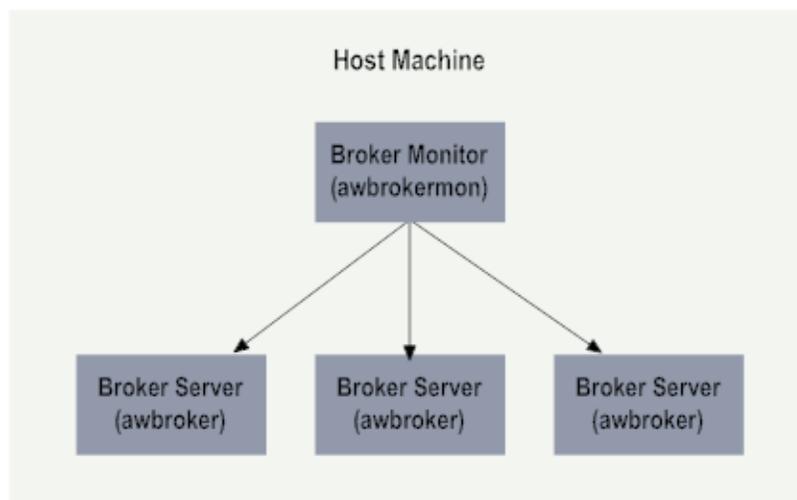
The Role of Broker Monitor

Broker Monitor (awbrokermom) is a separate process that runs on the machine that hosts Broker Server. It has three functions:

- It starts Broker Servers.
- It monitors the state of the Broker Servers running on the host machine and automatically attempts to restart them if they fail.
- It logs status messages about the Broker Servers.

During installation, Broker Monitor binds to a port and IP address on the host machine. Broker Monitor listens for requests from administrative clients (such as the Broker user interface in My webMethods) on the port to which it is bound. The default port of Broker Monitor is 6850.

Typical Broker Monitor installation monitors all Broker Servers on the host machine



You can have more than one instance of Broker Monitor on the host machine if the host machine has more than one instance of webMethods Broker installed. When running multiple instances of webMethods Broker, each installation is a stand-alone application with its own set of unique configuration and its own Broker Monitor. Broker Monitor only interacts with the Broker Servers in its own webMethods Broker environment. For more information, see [“Running Multiple Instances of webMethods Broker on the Same Host” on page 101](#) and [“Ports and Running Multiple Instances of Broker Monitor” on page 65](#).

For more information about binding Broker Monitor to an IP address, see [“Broker Security Model” on page 288](#).

How Broker Monitor Starts Broker Servers

The Broker Monitor configuration file (awbrokermon.cfg) points to the Broker Servers that reside on the host machine. When you start Broker Monitor, it reads this configuration file and automatically starts all of the Broker Servers that are identified in the file.

The Broker Monitor's configuration file is updated automatically when you install Broker Server on the host machine and when you define additional instances of Broker Server using the server_config utility. For detailed descriptions of the Broker Monitor configuration parameters, see “[Broker Monitor Configuration Parameters](#)” on page 637.

Platform	Location of awbrokermon.cfg file
UNIX	<i>webMethods Broker_directory/bin/awbrokermon.cfg</i>
Windows	<i>webMethods Broker_directory\bin\awbrokermon.cfg</i>

How Broker Monitor Monitors the State of Broker Servers

When Broker Monitor starts a Broker Server, it internally captures the process ID that the operating system assigns to the server process. Using this ID, Broker Monitor continually monitors the server's run state. If the server exits unexpectedly (that is, it is not stopped in a controlled way through the Broker user interface or the server_config utility), Broker Monitor automatically attempts to restart it.

To avoid the situation where Broker Monitor keeps restarting a Broker Server that is unable to stay up and running, Broker Monitor will not restart a Broker Server that has experienced three unexpected exits within a five minute period.

Status Messages Logged by Broker Monitor

Broker Monitor maintains a log file in which it records the following events for the Broker Servers that it monitors:

- Broker Server is launched (by Broker Monitor).
- Broker Server exits unexpectedly.
- Broker Server is stopped by an administrative action.

On Windows, this information is also written to the Windows event log and can be viewed through the Windows Event Viewer (for example, **Start > Settings > Control Panel > Administrative Tools > Event Viewer**).

On UNIX, the information is written by the syslog daemon. To understand the syslog daemon configuration and to determine the location of the log file, see the man page of "syslogd" for AIX, HP-UX, Solaris and Red Hat Enterprise Linux, or the man page of "syslog-ng" for SUSE Enterprise Server.

For information about viewing the log information generated by Broker Monitor and Broker Servers, see [“The Broker Server Log” on page 113](#).

Starting Broker Monitor

You can start Broker Monitor on both Windows and UNIX platforms either automatically or manually.

Starting Broker Monitor Automatically on Windows Systems

On Windows platforms, you generally configure Broker Monitor to run automatically when the host machine starts. The Software AG Installer registers Broker Monitor as a service with the following nomenclature:

```
Software AG Broker Monitor version (port_number) (instance_number)
```

For example, Software AG Broker Monitor *version* (9849) (2) is the service name for instance number 2 of Broker Monitor running on port 9849.

The Broker Monitor service starts automatically when the host machine starts. The port number in the service name is the port number specified in the awbrokermon.cfg file.

Starting Broker Monitor Manually on Windows Systems

If your Broker Monitor is configured for manual startup rather than automatic startup, or if Broker Monitor has been stopped and needs to be restarted, perform one of the following:

- Run the startup script.
- Start the Broker Monitor service.

To start Broker Monitor manually using the `startup` utility

1. Navigate to the `webMethods Broker_directory\bin` directory.
2. Run the following command:

```
.\startup.bat
```

Broker Monitor starts, and then Broker Monitor automatically starts all the Broker Servers to be monitored by it. If Broker Monitor is already running, then the `startup` script does nothing.

To start Broker Monitor service manually on Windows

1. From the Start menu, open Services as follows:

Settings > Control Panel > Administrative Tools > Services

Note: This command sequence varies by Windows version. If the sequence above does not work for your version, consult Windows help for the correct command.

2. Right-click the Software AG Broker Monitor *version (port_number) (instance_number)* service and click **Start**.

The status of the Broker Monitor switches to "Started" and Broker Monitor starts the Broker Servers that reside on the machine.

Starting Broker Monitor Automatically on UNIX Systems

By default, Broker Monitors run as applications that you start and shut down manually. If you want your Broker Monitors to start and shut down automatically when your system starts and shuts down, you must configure the Broker Monitors to run as daemons.

To configure Broker Monitor to start at system boot time on UNIX

To automatically start and stop Broker Monitor with the UNIX system, you must install the Broker Monitor start/kill script *webMethods Broker_directory/Broker/aw_brokerversion* to the boot system.

You must have root permissions to run the following commands.

- **AIX.** On AIX, you use the `mkitab` command. Read the man page for `mkitab` to get a full feature description.

The following example configures Broker Monitor to start with runlevel 3 and 5:

```
cp webMethods Broker_directory/Broker/aw_brokerversion /etc
mkitab "aw_brokerversion:53:wait:/etc/aw_brokerversion start > /dev/console"
```

- **HP-UX.** On HP-UX, you install the Broker Monitor start/kill script into the runlevel directories manually. The following example configures Broker Monitor to start/stop with HP-UX runlevel 3:

```
cp webMethods Broker_directory/Broker/aw_brokerversion /sbin/init.d
ln -s /sbin/init.d/aw_brokerversion /sbin/rc3.d/S45brokerversion # start
ln -s /sbin/init.d/aw_brokerversion /sbin/rc3.d/K45brokerversion # kill
```

Note: The number in S45... or K45... defines the order for the execution of multiple scripts within a runlevel directory and you can change it to any appropriate number. See the HP-UX man page for 'rc' to get the details.

- **Solaris or Linux.** On Solaris and Linux, you install the Broker Monitor start/kill script into the runlevel directories manually. The following example configures Broker Monitor to start/stop with HP-UX or Linux runlevel 3:

```
cp webMethods Broker_directory/Broker/aw_brokerverision /etc/init.d
ln -s /etc/init.d/aw_brokerverision /etc/rc3.d/S45brokerverision # start
ln -s /etc/init.d/aw_brokerverision /etc/rc3.d/K45brokerverision # kill
```

Note: The number in S45... or K45... defines the order for the execution of multiple scripts within a runlevel directory and you can change it to any appropriate number. See the Solaris man page for 'init.d' or the Linux man page for 'boot' to get the details.

Starting Broker Monitor Manually on UNIX Systems

If your Broker Monitor is configured for manual startup or if Broker Monitor has been stopped and needs to be restarted, perform one of the following to start it manually.

- Run the `startup` script.
- Start the Broker Monitor service.

To start Broker Monitor manually on UNIX using the `startup` utility

1. Navigate to the `webMethods Broker_directory/bin` directory.
2. Run the following command:

```
./startup.sh
```

Broker Monitor starts, and then Broker Monitor automatically starts all the Broker Servers to be monitored by it. If Broker Monitor is already running, then the `startup` script does nothing.

To start Broker Monitor manually on UNIX using the `start` script

1. Navigate to the directory that contains the Broker Monitor script.
2. Run the following command:

```
./S45brokerverision start
```

Where `S45brokerverision` specifies the file name of the start up script.

3. To confirm that Broker Monitor is running, enter the following command:

```
ps -ef | grep aw
```

If Broker Monitor is running, "awbrokermon" will appear in the process list.

Stopping Broker Monitor

Stopping Broker Monitor on Windows

When you stop Broker Monitor, you will also stop *all* the Broker Servers that it is monitoring. If you want to stop an individual Broker Server, see [“Stopping Broker Server” on page 73](#).

Perform one of the following to stop Broker Monitor on Windows:

- Run the `shutdown` script.
- Stop the Broker Monitor service.

To stop Broker Monitor manually on Windows using the `shutdown` utility

1. Navigate to the `webMethods Broker_directory\bin` directory.
2. Run the following command:

```
.\shutdown.bat
```

Stops all the Broker Servers being monitored by the Broker Monitor, and then stops the Broker Monitor.

To stop Broker Monitor service manually on Windows

1. From the Start menu, open Services as follows:

Settings > Control Panel > Administrative Tools > Services

Note: This command sequence varies by Windows version. If the sequence above does not work for your version, consult Windows help for the correct command.

2. Right-click the Software AG Broker Monitor *version (port_number)* (*instance_number*) service and click **Stop**.
3. Click **Yes** to confirm that you want to stop the Broker Servers that Broker Monitor is monitoring.

Stopping Broker Monitor on UNIX

When you stop Broker Monitor, you will also stop *all* the Broker Servers that it is monitoring. If you want to stop an individual Broker Server, see [“Stopping Broker Server” on page 73](#).

Perform one of the following to stop Broker Monitor on UNIX.

- Run the `shutdown` script.
- Stop the Broker Monitor service.

To stop Broker Monitor manually on UNIX using the `shutdown` utility

1. Navigate to the `webMethods Broker_directory/bin` directory.
2. Run the following command:

```
./shutdown.sh
```

Stops all the Broker Servers being monitored by the Broker Monitor, and then stops the Broker Monitor.

To stop Broker Monitor on UNIX

1. Navigate to the directory where the Broker Monitor script file resides.
2. Run the following command:

```
./S45brokerversion stop
```

Where *S45broker*version** specifies the file name of the stop script.

How to Secure Broker Monitor Access

Use the `monitor-allowed-client-ipaddress-list` configuration parameter available in the Broker Monitor's configuration file (`awbrokermon.cfg`) to restrict unauthorized clients from performing administrative tasks on the Broker Monitor and the corresponding Broker Servers. If you do not specify any IP address of the clients in the `monitor-allowed-client-ipaddress-list` configuration parameter, there is no restriction on any clients for accessing Broker Monitor. By default, no value is assigned to the `monitor-allowed-client-ipaddress-list` parameter.

For description of the `monitor-allowed-client-ipaddress-list` parameter, see “[Broker Monitor Configuration Parameters](#)” on page 637.

How to Determine Whether Broker Monitor Is Running

Checking on Windows

You can use the following procedure to determine whether Broker Monitor is running on Windows.

To check Broker Monitor on Windows

1. From the Start menu, open Services as follows:

Settings > Control Panel > Administrative Tools > Services

Note: This command sequence varies by Windows version. If the sequence above does not work for your version, consult Windows help for the correct command.

2. Verify that the status of the Software AG Broker Monitor *version* (*port_number*) (*instance_number*) service is "Started."

Checking on UNIX

You can use the following procedure to determine whether Broker Monitor is running on UNIX.

To check Broker Monitor on UNIX

Enter the following command:

```
ps -ef | grep aw
```

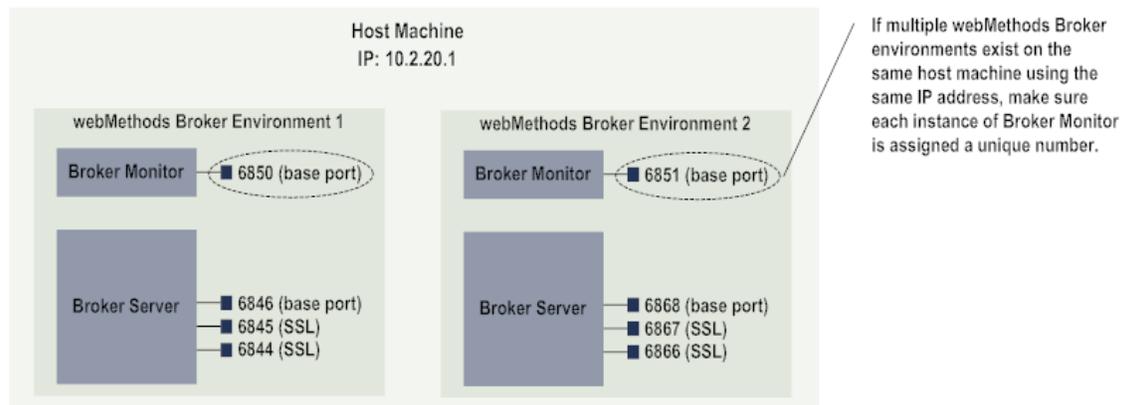
If Broker Monitor is running, "awbrokermon" will appear in the process list.

Ports and Running Multiple Instances of Broker Monitor

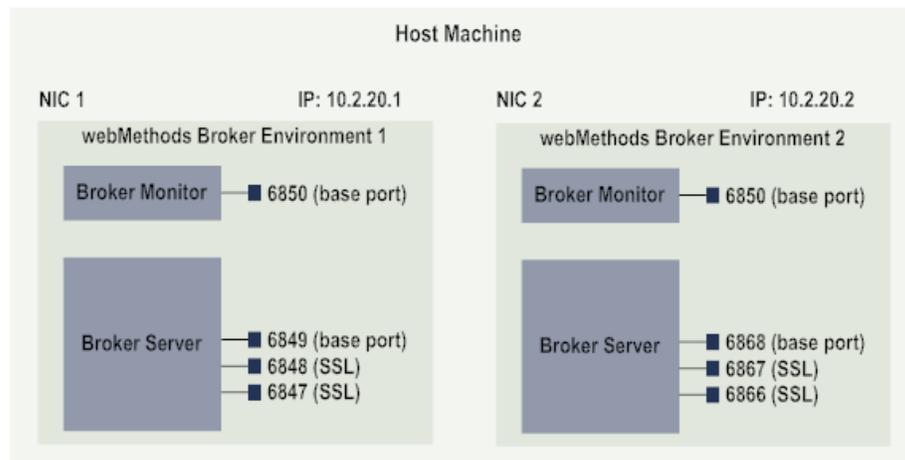
During installation, Broker Monitor binds to a port and IP address on the host machine. If a host machine supports virtual hosts/IPs and multiple NIC cards that reference a virtual host, the Broker Monitor can co-host multiple installations.

If you install multiple Broker environments on the same IP address on the same host machine, each Broker Monitor must be assigned a unique port. If, however, each Broker environment is installed on different IP addresses, the same Broker Monitor port can be used.

Two Broker environments on the same host machine with the same IP address



Two Broker environments on the same host machine with different IP addresses on the same port



Viewing the Port Setting for Broker Monitor

The port setting for Broker Monitor is a parameter in the Broker Monitor configuration file, `awbrokermon.cfg`. You can view the port setting by examining the `monitor-port` value in the configuration file with a text editor. If the port value is empty, then port 6850 is used.

On a Windows machine, you can also view the Broker Monitor port setting in the service name. Windows lists the Broker Monitor services in the following format:

```
Software AG Broker Monitor version (port_number) (instance_number)
```

Where:

- *version* is the version of webMethods Broker.
- *port_number* is the Broker Monitor port specified in the `awbrokermon.cfg` file.
- *instance_number* is the serial number of the Broker Monitor instance when multiple installations of Broker Server and Broker Monitor are running. Windows does not display the *instance_number* for the first instance of the Broker Monitor service. Subsequent instances of the service are numbered starting from (2).

Changing the Port Setting for Broker Monitor

When you have multiple installations of Broker Server and Broker Monitor running, the instance index is displayed along with the service names and port number. The instance index number starts with 1. The instance index of a Broker Server and that of the assigned Broker Monitor will be same. For example, Windows lists the services as follows:

```
Software AG Broker Monitor version (6850)
Software AG Broker Monitor version (7850) (2)
Software AG Broker Monitor version (9950) (3)
Software AG Broker Server version (6849)
```

```
Software AG Broker Server version (7849) (2)
Software AG Broker Server version (8849) (3)
Software AG Broker Server version (9949) (3)
```

In this example, the Broker Monitor on port 9950 monitors the Broker Servers on port 8849 and port 9949.

To change the port setting for Broker Monitor and update the service names

1. Run the following command to stop and remove the Broker Server and Broker Monitor Windows services:

```
awbrokermon -unconfig
```

Note: If you use `awbrokermon -stop` instead of `awbrokermon -unconfig`, Broker Monitor does not change the Windows service names.

2. Edit the value of `monitor-port` parameter in the `awbrokermon.cfg` file to change the port setting for Broker Monitor.
3. Run the following command to start and add the Broker Server and Broker Monitor Windows services:

```
awbrokermon -config
```

Note: If you use `awbrokermon -start` instead of `awbrokermon -config`, Broker Monitor does not change the Windows service names.

For example, after you change the Broker Monitor port from 6850 to 8850, Windows lists the services as following:

```
Software AG Broker Monitor version (7850) (2)
Software AG Broker Monitor version (8850)
Software AG Broker Monitor version (9950) (3)
Software AG Broker Server version (6849)
Software AG Broker Server version (7849) (2)
Software AG Broker Server version (8849) (3)
Software AG Broker Server version (9949) (3)
```

Configuring Broker Monitors to Start Automatically on UNIX Systems

By default, Broker Monitors run as applications that you start and shut down manually. If you installed webMethods Broker on a UNIX system, and you want your Broker Monitors to start and shut down automatically when your system starts and shuts down, you must configure the Broker Monitors to run as daemons.

1. Go to the webMethods Broker installation directory.
2. Copy each Broker Monitor startup script `aw_broker $version$` to the appropriate UNIX startup directory.
3. Replace the “aw” in `aw_broker $version$` with “ $Snumber$,” where $number$ is the run, or priority level (for example, `S45broker $version$`). If you configure multiple Broker Monitors to run as daemons, make the name for each Broker Monitor startup script

unique among the startup scripts in the UNIX startup directory (for example, *S45broker_{version}_1*, *S45broker_{version}_2*, and so on).

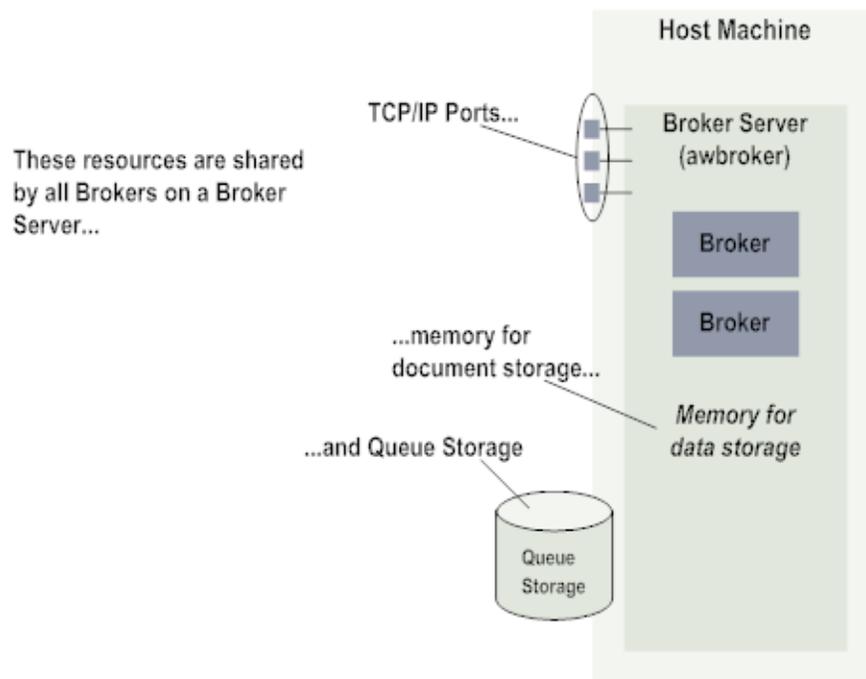
4 Managing Broker Servers

■ Overview	70
■ Starting Broker Server	71
■ Stopping Broker Server	73
■ Broker Server Communication Ports	76
■ Choosing the Storage Solution for Broker Server	79
■ Broker Server Memory	79
■ Configuring the max-memory-size Parameter	83
■ Configuring Queue Storage	84
■ Viewing Queue Storage Utilization and Settings	94
■ Configuring the Cache Settings	95
■ Configuring Broker Server to Use Asynchronous Write Mode	98
■ Running Multiple Broker Servers on the Same Host Machine	100
■ Creating a Broker Server with the Default Log and Storage Files	101
■ Creating a Broker Server that Specifies Size and Location of Files	103
■ Deleting a Broker Server	104
■ Viewing Status Information for a Broker Server	105
■ Broker Server License	108
■ Monitoring Resource Utilization	111
■ Logging	112
■ The Broker Server Log	113
■ The Messaging Log	117
■ Internet Protocol Support	121
■ Backing Up and Restoring a Broker Server	121
■ Pausing Document Publishing on the Broker Server	131
■ Configuring Broker Server for high throughput in high-latency and high-bandwidth networks ...	132
■ Setting the Locale	135

Overview

Broker Server is a container-like process (awbroker) that hosts one or more Brokers. Broker Server manages the communication, memory management, and queue storage functions for all Brokers that it hosts.

Resources managed by Broker Server and shared by the Brokers that it hosts



Broker Data Directory

Every Broker Server has its own data directory, which holds Broker Server's configuration file and log files. Frequently, the data directory also holds Broker Server's queue storage files, but often these files are placed on a separate (usually faster) storage device.

When you install Broker Server using the Software AG Installer, a single Broker Server is installed. The following table lists where the Broker Server data directory is located.

On this platform...	The default data directory is located here...
Windows	<i>webMethods Broker_directory\data\awbrokersversion \default\</i>
UNIX	<i>/var/opt/webMethods/awbrokersversion /default/</i>

If you define additional instances of Broker Server using the `server_config` utility, you must assign a separate data directory to each one.

Broker Server Configuration File (awbroker.cfg)

The Broker Server configuration file (`awbroker.cfg`) contains parameters that define a single Broker Server instance. The configuration file resides in Broker Server's data directory and supplies information such as Broker Server's license key location (in the license file), security settings, base port, and the location of its queue storage files. For detailed descriptions of the Broker Server configuration parameters, see “[webMethods Broker Server Configuration Parameters](#)” on page 621.

You can use My webMethods to display most parameters in the configuration file in the Broker user interface.

The following example illustrates the general content and format of the `awbroker.cfg` file.

```
file-encoding=escaped-unicode
license-key=C:\SoftwareAG\Broker\data\awbrokersversion\default\license.xml
port=6849
executable=C:\SoftwareAG\Broker\bin\awbroker.exe
version=9.6
log-alert=1
log-warning=1
log-info=1
syslog=-1
snmp=1
eventlog=1
internal=1
basic-auth-cfg-
file=C:\SoftwareAG\Broker\data\awbrokersversion\default\basicauth.cfg
session-
config=qs://C:\SoftwareAG\Broker\data\awbrokersversion\default\BrokerConfig.qs
session-
data=qs://C:\SoftwareAG\Broker\data\awbrokersversion\default\BrokerData.qs
```

Starting Broker Server

Broker Servers are usually started by Broker Monitor, which runs when the host machine boots. When Broker Monitor starts, it automatically starts all the Broker Servers that reside in the webMethods Broker environment.

Occasionally, you might want to start an individual Broker Server while Broker Monitor is running. For example, if you manually shut down a single Broker Server to perform a backup or an upgrade operation, you might want to restart that Broker Server without restarting Broker Monitor (which would have the unwanted effect of restarting all of the other Broker Servers in the webMethods Broker environment).

You can restart Broker Server by:

- Using My webMethods

- Using the `broker_start` utility
- Using the Windows control panel

Starting Broker Server from My webMethods

If your Broker Server appears in the Broker Server List, you can use the following procedure to restart it.

To use this procedure, Broker Monitor must be running on the host machine where Broker Server is installed. If Broker Monitor is not running, use the procedure in “[Starting Broker Monitor](#)” on page 60 to start Broker Monitor and Broker Server.

To manually start a Broker Server using My webMethods

1. In My webMethods: **Messaging > Broker Servers > Servers**.
2. In the **Broker Servers List**, select the check box beside the Broker Server that you want to start.
3. Click **Restart**.

Starting Broker Server Using the `broker_start` Command

If you need to start a Broker Server that does not appear in the Broker Server list, or if you want to start a Broker Server from the command line, you can use the `broker_start` utility.

To use this procedure, Broker Monitor must be running on the host machine where the Broker Server is installed. If Broker Monitor is not running, use the procedure in “[Starting Broker Monitor](#)” on page 60 to start Broker Monitor and the Broker Server.

To manually start a Broker Server from the command line (using `broker_start`)

1. On the machine where Broker Server is installed, navigate to the following directory:

`webMethods Broker_directory/bin`

2. Run the following command:

```
broker_start hostName:portNum -monitor_port monitorPort
```

Where...	Specifies...
<code>hostName</code>	The name of the host machine on which the Broker Server is installed. If <code>hostName</code> is omitted, localhost is used.
<code>portNum</code>	The Broker Server's base port number. If <code>portNum</code> is omitted, port 6849 is used.

Where...	Specifies...
<code>monitorPort</code>	The port number (default 6850) of the Broker Monitor that is monitoring the Broker Server.

The following examples show how you can use `broker_start`:

```
broker_start
broker_start localhost:6849

broker_start localhost:6845 -monitor_port 7850

broker_start atlas -monitor_port 8850

broker_start atlas.dev.mycompany.com:6845 -monitor_port 6850
```

For additional information about the `broker_start` utility, see [“broker_start” on page 560](#).

Starting Broker Server from Windows Control Panel

If the Broker Server is running under Windows and you have administrative access to that machine, you can restart the Broker Server service from the Windows control panel. This procedure will also start Broker Monitor if it is not already running.

Note: If this procedure starts Broker Monitor, it will start all Broker Servers in the webMethods Broker environment.

To manually start a Broker Server from the Windows control panel

1. From the Start menu in Windows, open Services as follows:

Settings > Control Panel > Administrative Tools > Services

Note: This command sequence varies by Windows version. If the sequence above does not work for your version, consult Windows help for the correct command.

2. Right-click the **webMethods Broker Server** service that you want to start and click **Start**.

Stopping Broker Server

If you want to shut down *all* Broker Servers that are running in a webMethods Broker environment, shut down Broker Monitor as described in, [“Stopping Broker Monitor” on page 62](#).

If you want to stop an individual Broker Server, without stopping Broker Monitor, use the procedures in this section.

When you stop a Broker Server, it immediately disconnects all clients and shuts down. Broker Server does not wait for in-process transmissions to complete before shutting

down. Volatile documents that Broker Server has in memory are lost. Destroy-on-disconnect clients are also discarded and are not restored when the Broker Server is restarted.

Explicit-destroy clients and their state (e.g., queues, subscriptions) are preserved and will be restored when you restart the Broker Server. Guaranteed documents that have been successfully persisted to the Queue Storage log file are preserved and will be placed in the appropriate client queues when the Broker Server restarts.

You can stop a Broker Server in the following ways:

- Using My webMethods
- Using the `broker_stop` utility
- Using the Windows Control Panel

Stopping Broker Server from My webMethods

If the Broker Server appears in your Broker Server List, you can use the following procedure to stop the server.

To stop a Broker Server using My webMethods

1. In My webMethods: **Messaging > Broker Servers > Servers**.
2. In the **Broker Servers List**, select the check box beside the Broker Server that you want to stop.
3. Click **Stop**.

Stopping Broker Server Using the `broker_stop` Command

If you want to stop a Broker Server from the command line, use the `broker_stop` command-line utility.

The `broker_stop` utility functions as an admin client and issues the stop request to the Broker Server through the network. If the Broker Server is protected by an access control list (ACL), you must include identity parameters that identify you as an authorized administrator of the Broker Server.

For additional information about the `broker_stop` command, see [“broker_stop” on page 562](#).

To stop a Broker Server from the command line (using `broker_stop`)

1. On the machine where Broker Server is installed, navigate to the following directory:

`webMethods Broker_directory/bin`

2. Run the following command:

```
broker_stop hostName :portNum-monitor_port monitorPort idParams
```

Where...	Specifies...
<i>hostName</i>	The name of the host machine on which the Broker Server is installed. If <i>hostName</i> is omitted, localhost is used.
<i>portNum</i>	The Broker Server's base port number. If <i>portNum</i> is omitted, port 6849 is used.
<i>monitorPort</i>	The port number (default 6850) of the Broker Monitor that is monitoring the Broker Server.
<i>idParams</i>	<p>Your identity as an authorized administrator of the Broker Server. You must include the following identity parameters if the Broker Server is protected by an ACL:</p> <pre>-certfile fileName</pre> <p>The fully-qualified name of the keystore file that contains your certificate.</p> <pre>-password password</pre> <p>Password for the certificate file. The utility will prompt you for the password if you omit this parameter from the command line.</p> <pre>-dn name</pre> <p>The distinguished name associated with the certificate that represents your identity. You can omit this parameter if the certificate file contains only one certificate.</p>

The following examples show how you can use `broker_stop`:

```
broker_stop
broker_stop atlas:6845

broker_stop atlas:6845 -monitor_port 8850

broker_stop atlas -certfile \keystores\myKeystore.cert -password oak11
-dn "CN=Broker Admin,OU=IT,O=webMethods,L=Santa Clara,ST=CA,C=US"

broker_stop atlas -monitor_port 8850 -certfile \keystores\myKeystore.cert
-password oak11 -dn "CN=Broker Admin,OU=IT,O=webMethods,L=Santa
Clara,ST=CA,C=US"
```

Stopping Broker Server from Windows Control Panel

If the Broker Server is running under Windows and you have administrative access to that machine, you can stop the Broker Server service from the Windows control panel.

To stop a Broker Server from the Windows control panel

1. From the Start menu, open Services as follows:

Settings > Control Panel > Administrative Tools > Services

Note: This command sequence varies by Windows version. If the sequence above does not work for your version, consult Windows help for the correct command.

2. Right-click the **webMethods Broker Server** service that you want to stop and click **Stop**.

Broker Server Communication Ports

When you install Broker Server or create a new Broker Server with the `server_config` utility, you specify the server's *base port*. Broker Server uses the base port for non-SSL communications. It uses the two ports immediately below the base port for SSL-based communications.

By default, port 6849 is the Broker Server base port. If you do not explicitly assign a base port when you install or create a Broker Server, it uses the default port 6849 for non-SSL requests, ports 6848 and 6847 for SSL requests, and 6846 and 6845 for parallel channels.

Ports Used by Broker Server

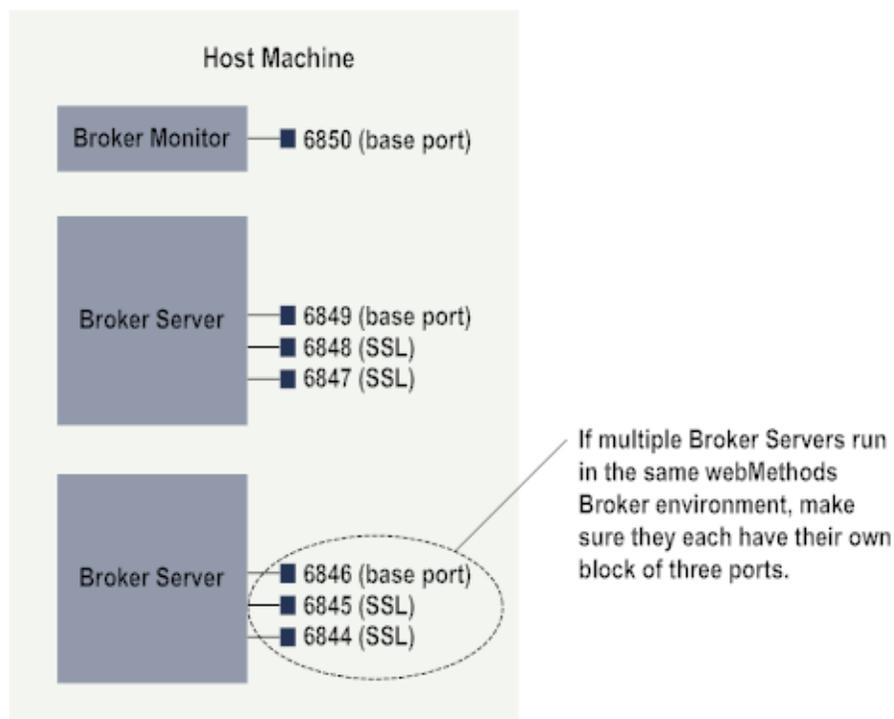
The following table describes the ports that Broker Server uses.

Port	Usage
Base Port	Non-SSL communications. The default is 6849.
Base Port -1	One-way (server-only) SSL authentication. Clients can optionally request encrypted transmissions on this port. When the default base port (6849) is used, the one-way SSL port runs at 6848.
Base Port -2	Two-way (client-and-server) SSL authentication. Clients can optionally request encrypted transmissions on this port. When the default base port (6849) is used, the two-way SSL port runs at 6847.
Base Port -3	Parallel channel for non-SSL communication. When the default base port (6849) is used, the port runs at 6846.

Port	Usage
Base Port -4	Parallel channel for SSL communication. When the default base port (6849) is used, the port runs at 6845.

Ports and Multiple Broker Servers

If you run multiple Broker Servers in the same webMethods Broker environment, you must ensure that each server has its own block of three ports and that none of the blocks interfere with each other or with the Broker Monitor port.



Viewing the Base Port Setting

The base port setting is a parameter in the Broker Server's configuration file (awbroker.cfg). You can view the base port setting in My webMethods or by examining the configuration file with a text editor.

Tip: If Broker Server is running on a Windows machine, the base port number also appears next to the Broker Server service name in the Windows Service list (e.g., Broker Server, port 6849).

To view the base port for a Broker Server

1. In My webMethods: **Messaging > Broker Servers > Servers.**

- Click the name of the server in the Broker Server List panel.

The Broker Server Details page displays the base port.

Changing the Base Port Setting

To change the base port assignment for a Broker Server, you must stop the Broker Server and change the port setting using the `server_config` utility.

Do not use the port on which Broker Monitor resides. The default port for Broker Monitor is 6850.

Note: On Windows, the port number is part of the service name. When you change the port number, the `server_config` utility attempts to update the service name. However, this action is not always successful. To switch the base port on a Windows machine, consider creating a new Broker Server with the new port number and then copying the data files (excluding `awbroker.cfg`) from the old Broker Server to the new Broker Server. For information about copying data files from one Broker Server to another, see [“Replicating Broker Metadata” on page 485](#).

To change the base port assignment of a Broker Server

- Stop the Broker Server.
- On the machine where Broker Server is installed, navigate to the following directory:

`webMethods Broker_directory/bin`

- Run the following command:

```
server_config update dataDir -p portNum
```

Where...	Specifies...
<code>dataDir</code>	The fully-qualified location of the Broker Server's data directory. Enclose the entire directory name in quotes if any portion of the name contains a space.
<code>portNum</code>	The new base port number.

For example, on Windows you could use the following:

```
server_config update "c:\SoftwareAG\Broker\data\awbrokersversion\My Broker" -p 6800
```

On UNIX, you could use the following:

```
server_config update /var/opt/webmethods/awbrokersversion/myBroker -p 6845
```

- Restart the Broker Server in port 6800 using the following command:

```
broker_start localhost:6800
```

Choosing the Storage Solution for Broker Server

Make sure Broker Server uses a storage solution that adheres to the following requirements.

- **Synchronous write persistence.** The storage solution must guarantee data write to durable and persistent storage when the synchronous write call returns.
- **Distributed file locking.** The storage solution must not assign the locks to two servers simultaneously. Broker Servers must be able to request and obtain an exclusive lock on the shared storage.
- **Unique write ownership.** The Broker Server process that has the file lock must be the only server process that can write to the file. Once the system transfers the lock to another server, the pending write requests queued by the previous owner must fail.
- **Write order.** The storage solution must write data blocks to the shared storage in the same order as they occur in the data buffer.

Note: Write order is not required if the storage solution supports synchronous writer persistence and the Broker Server configuration is the default synchronous write mode.

Broker Server Memory

Broker Server requires a substantial amount of memory to hold the documents that clients publish. Most of the memory is used to hold volatile documents, which exist only in memory. However, a certain portion of memory is used to cache guaranteed documents and other objects that are kept in queue storage.

To perform optimally, Broker Server needs enough physical memory to simultaneously hold:

- The maximum number of volatile documents that are awaiting delivery
- A fully loaded cache of guaranteed documents

If the operating system begins paging documents to virtual memory, the performance of the Broker Server drops significantly. Extreme out-of-memory conditions can cause the Broker Server to exit and restart.

Controlling Queue Size for Volatile Documents

You can proactively delete volatile documents from a queue by setting the "queue-cleanup-enable" and the "queue-cleanup-threshold" parameters in the awbroker.cfg file of each Broker Server instance. For complete information, see [“Proactively Deleting Documents from a Client Queue” on page 243](#).

Limiting Memory Usage by Broker Server

You can use the `max-memory-size` parameter (in `awbroker.cfg`) to prevent Broker Server from receiving more documents than it has the physical memory to store. This parameter specifies an upper memory limit. When this memory limit is reached, the Broker Server stops accepting documents.

When the `max-memory-size` parameter is set, Broker Server monitors the amount of memory that it is using for document storage. If the size of the incoming document plus the total amount of memory that is already "in use" exceeds the `max-memory-size` limit, Broker Server rejects the document and returns an "out of memory" error to the client.

If the `max-memory-size` is not specified (that is, this parameter is omitted from the Broker Server's configuration file), Broker Server does not monitor its memory usage and will accept documents even though there might not be sufficient memory to process them.

Maximum Limit Notification

When memory usage reaches 80% of the maximum specified in `max-memory-size`, Broker Server writes a warning in the journal log. If usage reaches 95% of the specified maximum, Broker Server issues an alert to the journal log. When memory usage returns to a level below 80%, Broker Server writes an informational message to denote that memory usage has dropped out of the danger zone.

To avoid cluttering the log with memory-related messages, the Broker Server does not issue these types of messages more than once an hour.

Values for the max-memory-size Parameter

The value of `max-memory-size` is an integer that represents the maximum amount of memory, in megabytes, that Broker Server can use for document storage. The default value of `max-memory-size` is the total memory of the system.

The following example sets the memory limit to 1.1 gigabytes:

```
max-memory-size=1100
```

The minimum value that you can set for `max-memory-size` is 50 megabytes. If you specify a value less than 50, Broker Server displays an alert, ignores the setting, and resets the value of `max-memory-size` to the default value.

If you change the value of `max-memory-size`, you must restart Broker Server to put the new value into effect.

The preallocate-memory Parameter

You can optionally enable the `preallocate-memory` parameter, which causes Broker Server to allocate the amount of memory specified in `max-memory-size` when it starts. If Broker Server cannot obtain the specified amount, it will immediately exit.

Note: The preallocate memory feature only works for a Broker Server running on a UNIX host machine. It does not work for Broker Servers running on Windows machines.

Setting the `preallocate-memory` parameter ensures that Broker Server can actually obtain the amount of memory that is specified by `max-memory-size`. If other applications besides Broker Server run on the same host machine or multiple instances of webMethods Broker run on the same machine and they use a significant amount of memory, we recommend that you preallocate the memory space to ensure that the memory is actually available to Broker Server. (Setting the `preallocate-memory` parameter will slow down the startup sequence slightly.)

Selecting the Maximum Memory Size

Although there are no hard and fast rules for setting the memory threshold, you can use the following formula as a guide. This formula will provide an approximate value that you can use as a starting point and refine through testing.

$$\text{MaxMemorySize} = \text{TotalMemory} - (\text{OS} + \text{OtherSoftware} + \text{BrokerProgram} + \text{SafetyMargin})$$

Where...	Is...
<i>TotalMemory</i>	The total amount of physical memory (RAM <i>not</i> virtual memory) on the machine.
<i>OS</i>	The amount of memory consumed by the operating system.
<i>OtherSoftware</i>	The amount of memory consumed by other software programs that run on the same host machine.
<i>BrokerProgram</i>	The amount of memory used by the Broker Server program code. 20 MB is a reasonable approximation for this value on a 32-bit processor. 40 MB on a 64-bit processor.
<i>SafetyMargin</i>	An extra amount that covers memory usage that might not be accounted for in any of the above.

Example:

Assuming the following values:

TotalMemory = 2000 MB (2 gigabytes)

OS = 800 MB

OtherSoftware = 100 MB

BrokerProgram = 20 MB

SafetyMargin = 100 MB

Calculation:

$MaxMemorySize = 2000 - (800 + 100 + 20 + 100)$

Result:

$MaxMemorySize = 980$

Important: This formula is meant to be used as a starting point for testing. The 'best' max-memory-size is specific to the characteristics of a particular environment and workload (for example, document size, mix of guaranteed and volatile documents). You can identify the right size only through experimentation.

Limitations of the Maximum Memory Size

The Broker memory limit feature monitors total memory usage for publishing operations by tracking all memory allocation calls. Using that information, the feature prevents calls for memory if the total Broker memory remaining is insufficient to service a publishing operation. However, because Broker does not track actual heap memory, there are certain circumstances under which the maximum memory limit feature may not be able to prevent the Broker from getting into an extreme out-of-memory condition and exiting.

- It is possible for the heap memory to become highly fragmented in such a way that:

$amount_memory_requested < (max_memory_size) - (current_total_memory_used)$

but the memory allocator is not able to find a contiguous block of memory large enough to satisfy *amount_memory_requested* for the publication. One way to prevent that scenario is by increasing the *SafetyMargin* when calculating:

$MaxMemorySize = TotalMemory - (OS + OtherSoftware + BrokerProgram + SafetyMargin)$

An increased *SafetyMargin* effectively sets aside contiguous memory chunks that can be used later to satisfy unexpectedly large publication requests. A rule of thumb is to have the *SafetyMargin* set at 40 to 100 percent of the *MaxMemorySize*.

- An incorrect *TotalMemory* value is used in calculating *MaxMemorySize*, where the *TotalMemory* value is larger than the actual memory available to the Broker process. This condition should be corrected by using the correct value for *TotalMemory*.

Configuring the max-memory-size Parameter

Use the following procedure to set the `max-memory-size` parameter in the Broker Server configuration file (`awbroker.cfg`). If you want Broker Server to allocate the specified amount of memory when it starts, enable the `preallocate-memory` parameter in the configuration file.

To configure the memory parameters

1. Stop the Broker Server.
2. On the machine where Broker Server is installed, locate the Broker Server's configuration file (`awbroker.cfg`) and make a backup copy.
3. Open the configuration file with a text editor and locate the `max-memory-size` parameter. If your configuration file does not include this parameter, add it to the file. For detailed descriptions of the Broker Server configuration parameters, see [“webMethods Broker Server Configuration Parameters” on page 621](#).

```
.
.
.
snmp=1
eventlog=1
internal=1
max-memory-size=500
```

4. Modify the value of the `max-memory-size` parameter to specify the maximum amount of memory, in megabytes, that Broker Server can allocate for document storage before it begins blocking incoming documents.

The following example sets the `max-memory-size` parameter to 1.2 gigabytes.

```
.
.
.
snmp=1
eventlog=1
internal=1
max-memory-size=1200
```

5. If you want Broker Server to allocate this amount of memory when it starts, include the `preallocate-memory` parameter and set its value to 1. If you do not want to preallocate this amount of memory, set the `preallocate-memory` parameter to 0 or omit it from the configuration file.

```
.
.
.
snmp=1
eventlog=1
internal=1
max-memory-size=1200
preallocate-memory=1
```

Important: If you enable the `preallocate-memory` parameter and Broker Server cannot obtain the amount of memory specified in `max-memory-size`, it will not start.

6. Save the configuration file.
7. Restart the Broker Server.

Configuring Queue Storage

Queue storage is the storage system that Broker Server uses to persist guaranteed documents and non-volatile metadata such as Broker definitions, document types, client groups, and explicit-destroy clients. All Brokers on a Broker Server share the same queue storage space.

Queue storage is composed of two types of files: a *log file* and one or more *storage files*.

The Log File

The log file is where Broker Server initially stores the data that it needs to persist to disk. When a client publishes a guaranteed document, Broker Server commits the document to the log file before it returns control to the client. If the Broker Server cannot successfully write the document to the log, it returns an error so the publisher knows that the document was not received successfully.

Broker Server is able to write to the log file very quickly, in part, because the file is a fixed size and is preallocated. When the log file becomes full, Broker Server transfers data from the log file to a storage file to make room for new, incoming data.

You select the size of the log file when you install Broker Server or create a Broker Server with the `server_config` utility. You can modify the size and location of the log file after it has been created, but you must stop and restart the Broker Server to do so. For procedures, see [“Modifying the Size of the Log File” on page 88](#).

The Software AG Installer automatically creates the log file in the Broker Server's data directory. If you use the `server_config` utility to create a Broker Server, you can place the log file elsewhere.

When you configure the size and location of the log file, keep the following points in mind:

- The maximum size for a log file is 2 GB on 32-bit platforms and 8 GB (as a practical limit) on 64-bit platforms.
- The log file for run-time data must be large enough to hold the largest document or batch of documents that a client will publish in a single transaction.
- The larger the log file, the longer it will take Broker Server to start. When Broker Server starts, it reads the entire log and moves items to long-term storage.

- Space for the entire log file is immediately allocated when you install or create the Broker Server.
- For optimal performance, place the log file on a fast device that is dedicated to the Broker Server and is separate from the storage files.

The Storage Files

Storage files are used for long-term storage of configuration data and for guaranteed documents that have not yet been retrieved by their subscribers. The Broker Server automatically moves data (in 8 MB chunks) from the log file to a storage file when the log becomes full.

Log data is also moved to long-term storage each time you start the Broker Server. This process, which is often referred to as "playing the log," effectively "empties" the entire log file by moving pending items to a storage file and purging items that are expired or have been acknowledged by all of their subscribers.

When you configure a storage file you specify its *reserve size* and its *maximum size*. The reserve size specifies the file's starting size (that is, the amount of space that is immediately allocated to the file). The maximum size specifies the size to which the storage file can grow.

A Broker Server can have multiple storage files. When you install Broker Server using the Software AG Installer, the Installer creates one storage file. You can create additional storage files using the `server_config` utility after you install Broker Server, however you must stop and restart the server to do this. For procedures, see [“Adding a Storage File” on page 91](#).

After you create a storage file, you can use the `server_config` utility to increase the file's reserve size or its maximum size. For procedures, see [“Modifying the Size of a Storage File” on page 90](#).

You cannot decrease the reserve or maximum size of a storage file, delete the storage file, or change the file's location after it is created.

When you configure the size and location of a storage file, keep these points in mind:

- The maximum size for a storage file is 32 gigabytes.
- The minimum reserve size is 16 megabytes.
- You can create up to 62 storage files for each session.
- For optimal performance, place storage files on fast devices that are dedicated to the Broker Server and are separate from the log file.
- If you use multiple storage files, give each storage file a unique name, even if you store them in different directories. For example, use:

```
data/myBrokerStore/datastore01.qs.stor  
data/myBrokerStore/datastore02.qs.stor
```

Do not use the following:

```
data/myBrokerStore01/datastore.qs.stor  
data/myBrokerStore02/datastore.qs.stor
```

Storage files that have the same name will not cause problems with the operation of queue storage, but they can cause problems if they ever need to be restored using the `server_conf_restore` utility.

Combined and Separate Storage Sessions

Broker Server can be configured to use *combined* or *separate* storage sessions.

When you use a combined session, Broker Server persists metadata and run-time data to the same set of queue storage files. When you use separate storage sessions, Broker Server persists metadata to one set of files (called the *configuration session*) and run-time data to another set of files (called the *data session*).

Metadata includes:

- Brokers
- Client groups
- Territories
- Clusters
- Gateways
- Explicit-destroy clients
- Access control lists
- Document types

Run-time data includes:

- Guaranteed documents
- Client queues
- Client queue statistics

By default, Broker Server is installed with separate storage sessions. If you use the `server_config` utility to create a new Broker Server, you can configure the new server to use separate or combined sessions.

The use of separate storage sessions is strongly recommended for production environments. This session type enables you to use the `server_conf_backup` utility to back up the Broker Server metadata without taking the server offline.

Important: After you install or create a Broker Server, its storage-session type cannot be changed. To switch session types, you must create a Broker Server with the new session type and import the old server's metadata data into the new Broker Server. For information about importing data from one Broker Server into another, see [“Replicating Broker Metadata” on page 485](#).

Default Queue Storage Files Created by the Installer

When you install a Broker Server, you choose a small, medium, or large configuration. The size you choose determines the size of the log and storage files that the Installer configures for each session.

Size	Session	File Type (Name) *	Default Sizes
Small	Run-Time	Log file (BrokerData.qs.log)	64 MB
		Storage file (BrokerData.qs.stor)	1 GB (max)/128 MB (reserve)
	Configuration	Log file (BrokerConfig.qs.log)	64 MB
		Storage file (BrokerConfig.qs.stor)	1 GB (max)/128 MB (reserve)
Medium (default)	Run-Time	Log file (BrokerData.qs.log)	256 MB
		Storage file (BrokerData.qs.stor)	4 GB (max)/512 MB (reserve)
	Configuration	Log file (BrokerConfig.qs.log)	256 MB
		Storage file (BrokerConfig.qs.stor)	4 GB (max)/512 MB (reserve)
Large	Run-Time	Log file (BrokerData.qs.log)	512 MB
		Storage file (BrokerData.qs.stor)	8 GB (max)/1 GB (reserve)
	Configuration	Log file (BrokerConfig.qs.log)	512 MB
		Storage file (BrokerConfig.qs.stor)	8 GB (max)/1GB (reserve)

* These files are created in the Broker Server data directory.

Default File Names and Sizes Created by server_config

When you create a Broker Server using the `server_config` utility, you specify whether you want the server to use separate or combined storage sessions. If you do not explicitly specify the size and location of the queue storage files, the utility creates the following set of default files in the Broker Server data directory.

The following table shows the defaults for separate-session configuration:

Session	Files Type (Name)	Default Sizes
Run-time	BrokerData.qs.log	256 MB
	BrokerData.qs.stor	4 GB (max)/512 MB (reserve)
Configuration	BrokerConfig.qs.log	256 MB
	BrokerConfig.qs.log	4 GB (max)/512 MB (reserve)

The following table shows the defaults for combined-session configuration:

Session	Files Type (Name)	Default Sizes
Combined	Broker.qs.log	256 MB
	Broker.qs.stor	4 GB (max)/512 MB (reserve)

Modifying the Size of the Log File

You can use the `server_config` utility to increase or decrease the size of the log file for a storage session. When you change the size of the log file, you actually create a new log file of the specified size. Before creating the new log file, the `server_config` utility empties the old file by applying the data it contains to the storage file.

When specifying the size of the new log file, be sure the file is large enough to hold the largest document, or batch of documents, that a client will publish in a single transaction. Remember that the log file is used by all Brokers on the Broker Server.

For best performance, the log file should be placed on a dedicated device that is not used by any other application or by the storage files.

The maximum size for a log file is 2 GB on 32-bit platforms and 8 GB (as a practical limit) on 64-bit platforms.

Note: If the existing log file contains a large amount of data, it might take several minutes for the `server_config` utility to empty the old log and create a new one.

To modify the size of a log file

1. Stop the Broker Server.
2. On the machine where Broker Server is installed, navigate to the following directory:
webMethods Broker_directory/bin
3. Run the following command. Type the entire command on one line.

```
server_config storage dataDir -session_type qs
-qs_log_file logFile logSize
```

Where...	Specifies...
<i>dataDir</i>	The fully-qualified location of the Broker Server's data directory. Enclose the entire directory name in quotes if any portion of the name contains a space.
<i>type</i>	<p>The storage session whose log file you want to modify.</p> <ul style="list-style-type: none"> ■ If you are modifying the log file for the configuration session or for a combined storage session, <i>type</i> is: <code>config</code> ■ If you are modifying the log file for the run-time data session, <i>type</i> is: <code>data</code>
<i>logFile</i>	<p>The fully-qualified name of the new log file.</p> <p>Note: By convention, the names of log files have ".qs.log" as an extension. Although you are not required to use this extension when you create a log file, we recommend you do so to make the log file easy to identify.</p> <p>If you are running a separate-session configuration, we recommend that you specify a name that indicates whether the log file belongs to the configuration session or the run-time session (e.g., <code>configData.qs.log</code> or <code>run-timeData.qs.log</code>).</p>
<i>logSize</i>	The size of the log file. Specify the size as a number followed by a K (kilobytes), M (megabytes), or G (gigabytes).

The following example illustrates the configuration for a combined session:

```
server_config storage
"C:\webMethods Broker_directory\Broker\data\awbrokersversion\my Broker"
-session_config qs
-qs_log_file "C:\webMethods Broker_directory\Broker\data\awbrokersversion\
my Broker\BrokerConfig01.qs.log" 20M
```

The following example illustrates the configuration for a run-time data session:

```
server_config storage
```

```
"C:\webMethods Broker_directory\Broker\data\awbrokersversion\my Broker"
-session_data qs
-qs_log_file
"C:\webMethods Broker_directory\Broker\data\awbrokersversion\
my Broker\BrokerData01.qs.log" 1G
```

Modifying the Size of a Storage File

You can use the `server_config` utility to increase the maximum size of a storage file or increase its reserve size. You cannot reduce the maximum size or the reserve size of a storage file.

The maximum size of a storage file is 32 gigabytes. If your storage file is already at the maximum size and you require more storage, you can create additional storage files. For procedures, see [“Adding a Storage File” on page 91](#).

Note: The queue storage system will seek a storage file that is not busy when it is moving an object from the log file to a storage file. Therefore, configuring multiple storage files can provide faster access than a single storage file.

To modify the size of a storage file

1. Stop the Broker Server.
2. On the machine where Broker Server is installed, navigate to the following directory:
`webMethods Broker_directory/bin`
3. Run the following command. Type the entire command on one line.

```
server_config storage dataDir -session_type qs
                -qs_storage_file storeFile storeSize reserveSize
```

Where...	Specifies...
<code>dataDir</code>	The fully-qualified location of the Broker Server's data directory. Enclose the entire directory name in quotes if any portion of the name contains a space.
<code>type</code>	The session whose storage file you want to modify. <ul style="list-style-type: none"> ■ If you are modifying the storage file for the configuration session or for a combined storage session, <code>type</code> is: <code>config</code> ■ If you are modifying the storage file for the run-time data session, <code>type</code> is: <code>data</code>
<code>storeFile</code>	The fully-qualified name of the storage file whose size you want to change. Enclose the entire directory name in quotes if any portion of the name contains a space.

Where...	Specifies...
<i>storeSize</i>	<p>The maximum size to which this storage file can grow. Specify the size as a number followed by a K (kilobytes), M (megabytes), or G (gigabytes).</p> <p>You must specify a value that is larger than the storage file's current maximum size setting.</p> <p><i>storeSize</i> must not exceed 32 gigabytes.</p>
<i>reserveSize</i>	<p>The total amount of storage that should be reserved for this storage file. For example, if the storage file is currently 64 megabytes and you want to allocate an additional 10 megabytes to the file, you would specify a <i>reserveSize</i> of 74 megabytes.</p> <p>Specify a number followed by a K (kilobytes), M (megabytes), or G (gigabytes).</p> <p><i>reserveSize</i> must be larger than the storage file's current size but less than <i>storeSize</i>.</p>

The following example illustrates a configuration session or combined session:

```
server_config storage
"C:\webMethods_Broker_directory\Broker\data\awbrokersversion\my Broker"
-session_config qs
  -qs_storage_file
C:\webMethods_Broker_directory\Broker\data\awbrokersversion\myBroker\
BrokerConfig01.qs.stor 512M 100M
```

The following example illustrates a run-time data session:

```
server_config storage
"C:\webMethods_Broker_directory\Broker\data\awbrokersversion\my Broker"
-session_data qs
  -qs_storage_file
C:\webMethods_Broker_directory\Broker\data\awbrokersversion\myBroker\
BrokerData01.qs.stor 1G 100M
```

Adding a Storage File

You can use the `server_config` utility to add one or more storage files to queue storage. A storage session can have a total of 62 storage files. Each file can be up to 32 gigabytes in size.

The queue storage system will seek a storage file that is not busy when it is moving an object from the log file to a storage file. Therefore, multiple storage files can provide faster access than a single storage file.

Note: If you add a storage file with a very large reserve size, it might take several minutes for the `server_config` utility to initialize the new file (up to 15 minutes for very large files).

To add one or more storage files to queue storage

1. Stop the Broker Server.
2. On the machine where Broker Server is installed, navigate to the following directory:

```
webMethods Broker_directory/bin
```

3. Run the following command. Type the entire command on one line.

Note that you can enter the `-qs_storage_file` parameter multiple times to add multiple storage files.

```
server_config storage dataDir -session_type qs
  -qs_storage_file storeFile
storeSize reserveSize
  [-qs_storage_file storeFile
storeSize reserveSize ...]
```

Where...	Specifies...
<i>dataDir</i>	The fully-qualified location of the Broker Server's data directory. Enclose the entire directory name in quotes if any portion of the name contains a space.
<i>type</i>	The session to which you want to add the storage file. <ul style="list-style-type: none"> ■ If you are adding the storage file to the configuration session or for a combined storage session, <i>type</i> is: <code>config</code> ■ If you are adding the storage file to the run-time data session, <i>type</i> is: <code>data</code>
<i>storeFile</i>	The fully-qualified name of the new storage file. Enclose the entire directory name in quotes if any portion of the name contains a space. <p>Note: By convention, the names of storage files have ".qs.stor" as an extension. Although you are not required to use this extension when you create a storage file, we recommend you do so to make the file easy to identify. If you are running a separate-session configuration, we recommend that you specify a name that indicates whether the storage file belongs to the configuration session or the data session (e.g., <code>configData.qs.stor</code> or <code>run-timeData.qs.stor</code>).</p>

Where...	Specifies...
<i>storeSize</i>	<p>The maximum size to which this storage file can grow. Specify the size as a number followed by a K (kilobytes), M (megabytes), or G (gigabytes).</p> <p><i>storeSize</i> must not exceed 32 gigabytes.</p> <p>Once specified, the maximum size of the storage file can be increased, but not reduced.</p>
<i>reserveSize</i>	<p>The total amount of storage that should be reserved for the storage file. (The utility will create a storage file of this size.)</p> <p>Specify a number followed by a K (kilobytes), M (megabytes), or G (gigabytes).</p> <p><i>reserveSize</i> must be at least 16M, but less than <i>storeSize</i> .</p> <p>Once specified, the reserve size for the storage file can be increased, but not reduced.</p>

The following example illustrates a configuration session or combined session:

```
server_config storage
C:\webMethods_Broker_directory\Broker\data\awbrokersversion\myBroker
-session_config qs
  -qs_storage_file
C:\webMethods_Broker_directory\Broker\data\awbrokersversion\myBroker\
BrokerConfig01.qs.stor 512M 100M
```

The following example illustrates a run-time data session:

```
server_config storage
C:\webMethods_Broker_directory\Broker\data\awbrokersversion\my Broker
-session_data qs
  -qs_storage_file
C:\webMethods_Broker_directory\Broker\data\awbrokersversion\myBroker\
BrokerData01
.qs.stor 1G 100M
```

The following example illustrates a run-time data session with multiple storage files:

```
server_config storage
C:\webMethods_Broker_directory\Broker\data\awbrokersversion\myBroker
-session_data qs
  -qs_storage_file
C:\webMethods_Broker_directory\Broker\data\awbrokersversion\myBroker\
BrokerData01.qs.stor 1G 100M
  -qs_storage_file
C:\webMethods_Broker_directory\Broker\data\awbrokersversion\myBroker\
BrokerData02
.qs.stor 1G 100M
  -qs_storage_file
C:\webMethods_Broker_directory\Broker\data\awbrokersversion\myBroker\
BrokerData03
.qs.stor 1G 100M
```

Viewing Queue Storage Utilization and Settings

You can use My webMethods to view the following information about queue storage:

- The size of the log file
- The size, location, and utilization of the storage files
- The maximum size to which each storage file can grow

To view the queue storage utilization and settings for a server

1. In My webMethods: **Messaging > Broker Servers > Servers**.
2. Click the server in the **Broker Server List**.
3. Click the **Utilization** tab to display the storage sessions associated with the Broker Server.
 - The resource list on this tab displays a top-level entry for each storage session. If the Broker Server uses combined sessions, the tab will display an entry for Runtime Data and Configuration Data (that is, metadata). If the Broker Server has a single session, it will have an entry for Configuration Data.
 - Under each top-level session entry, the tab displays the individual storage files associated with the storage session.
 - The value in the **Total Capacity** column indicates the maximum size to which the storage file can grow. In rows that represent the entire storage session, this column displays the combined total capacity of all storage files.
4. Click **Details**  to display the Broker ServerUtilization page. The **Storage** section on this page provides the following information about queue storage. If you are viewing details for the entire storage session, the values in the **Total Storage** section on this page represent the combined totals for all of the session's storage files.

Field	Description
File Size Limit	The maximum size to which the storage file can grow.
Used	The amount of data currently stored in the storage file.
Current File Size	The allocated size of the storage file.
Log File Size	The number of bytes currently allocated to the log file. (Displayed only when you view details for an entire storage session.)

Configuring the Cache Settings

Broker Server uses a portion of memory to cache guaranteed documents and other objects (for example, Broker metadata and explicit-destroy client queues and subscriptions) that Broker Server keeps in queue storage.

The `max_cache_size` setting restricts the amount of memory that Broker Server can use to cache data for queue storage. This parameter ensures that a certain amount of memory is available to retain queue storage objects in memory, but prevents the cache from depriving the Broker Server of the memory it needs to store volatile objects such as volatile documents and destroy-on-disconnect client queues and subscriptions.

For separate storage sessions, there are two cache settings, one for each storage session. For combined storage session, there is a single cache setting.

Default Cache Settings

The following table shows the installed default sizes assigned to cache. You can modify the cache size if you do not want to use the default setting; however, you cannot reduce it below the minimum size shown below.

The following table shows the cache size used for separate storage sessions:

Session	Cache Size
Configuration	<ul style="list-style-type: none"> ■ Default Size: 8 MB ■ Minimum Size: 8 MB
Run-time data	<ul style="list-style-type: none"> ■ Default Size: Size of the physical memory on the host machine or the value specified in the <code>max-memory-size</code> parameter, whichever is smaller. That is, <code>MIN (physicalMemorySize, max-memory-size)</code>. ■ Minimum Size: 64 MB

The following table shows the cache size used for combined storage sessions:

Session	Cache Size
Configuration	<ul style="list-style-type: none"> ■ Default Size: Size of the physical memory on the host machine or the value specified in the <code>max-memory-size</code> parameter, whichever is smaller. That is, <code>MIN (physicalMemorySize, max-memory-size)</code>. ■ Minimum Size: 64MB

Note: The actual size of the cache may exceed the specified maximum by up to 15% during peak loads. This 15% allowance gives Broker Server the leeway to expand the cache in order to accommodate an entire document.

Selecting a Cache Size

For most environments, the default settings are adequate. However, you might need to increase the cache size if your Broker Server receives very large guaranteed documents or if it receives many guaranteed docs simultaneously.

If you modify the cache setting, keep the following points in mind:

- A cache that is too large relative to the amount of physical memory on the Broker Server host machine might, when full, cause volatile data to begin paging to virtual storage.
- A cache that is sized too small reduces performance by increasing the percentage of guaranteed documents that Broker Server has to fetch from disk.

Always select a cache size that leaves an ample amount of physical memory for the storage of volatile documents and destroy-on-disconnect client information.

The right cache size is highly dependent on the characteristics of the traffic that it will receive. It is affected by factors such as the ratio of volatile to guaranteed documents, document size, arrival rate, and retrieval rate.

You will need to experiment to find a cache setting that suits your mix of documents. In general, larger cache settings are effective if your guaranteed documents are short-lived, meaning that subscribers will retrieve the documents promptly after they are published. For this case, you might even maintain a cache as large as your log file. If, on the other hand, guaranteed documents are likely to remain on Broker Server for lengthy periods before being retrieved, a small cache is generally more practical.

Modifying the Cache Size

To modify the cache size, you must modify the storage session parameters in the Broker Server configuration file (`awbroker.cfg`). For more information, see “[webMethods Broker Server Configuration Parameters](#)” on page 621.

As shown in the example below, the “`max_cache_size`” argument is a name-value pair that you append to the session URL as a query string. The value of this argument specifies the maximum size of the cache in megabytes.

```
session-config=qs:///var/opt/BrokerStorage/BrokerConfig.qs?max_cache_size=16  
session-data=qs:///var/opt/BrokerStorage/BrokerData.qs?async,max_cache_size=512
```

If the `max_cache_size` argument is omitted from the URL, the default cache size is used. See “[Default Cache Settings](#)” on page 95.

Note: Prior to version 6.5, a global parameter in the `awbroker.cfg` file was used to specify the size of the cache. This parameter (`storage-max-cache-size`) has

been deprecated. If present in the configuration file, it will be overridden by the value specified in `max_cache_size`.

To modify the cache size

1. Stop the Broker Server.
2. On the machine where Broker Server is installed, locate the Broker Server's configuration file (`awbroker.cfg`) and make a backup copy.
3. Open the configuration file with a text editor and locate the `session-config` and `session-data` parameters, which are underscored in the example below. (If you are running a combined storage session, your configuration file will contain only the `session-config` parameter.)

```
.
.
.
snmp=1
eventlog=1
internal=1
session-config=qs:///var/opt/BrokerStorage/BrokerConfig.qs
session-data=qs:///var/opt/BrokerStorage/BrokerData.qs
```

4. To specify the cache size for a storage session, append the "`max_cache_size=nnn`" name-value pair to the session's URL, where `nnn` specifies the cache size in megabytes.

Because the storage session is specified as a URL, you must include the "?" separator if `max_cache_size` is the first argument in the query string. If the URL already includes a query string, use the "," character to separate `max_cache_size` from the other arguments in the string. Both forms are shown in the example below.

```
.
.
.
snmp=1
eventlog=1
internal=1
session-config=qs:///var/opt/BrokerStorage/BrokerConfig.qs?max_cache_size=16
session-
data=qs:///var/opt/BrokerStorage/BrokerData.qs?async,max_cache_size=512
```

5. Save the configuration file.
6. Restart the Broker Server.

Viewing the Cache Settings

To determine the size of the cache on a Broker Server, you can examine the `session-config` and `session-data` parameters in the Broker Server configuration file (`awbroker.cfg`) or you can use the following procedure to view the cache settings using My webMethods.

To view the cache setting

1. In My webMethods: **Messaging > Broker Servers > Servers**.
2. In the **Broker Server List**, click the server whose cache setting you want to view.
3. Click the **Utilization** tab.
4. Click  **Details** for the storage session (that is, **Configuration Data**, **Run-Time Data**, or **Configuration and Run-time Data**) whose cache setting you want to view. (Be sure to click the icon for the top-level storage session, not an individual storage file.)
5. Examine the **Session URL** value in the **Storage Session** section of the **Server Utilization** tab.

The `max_cache_size` argument in the query string at the end of this URL specifies the maximum amount of memory (in megabytes) that can be used as cache. If the URL does not include the `max_cache_size` argument, the default cache size is in effect. See [“Default Cache Settings” on page 95](#).

Configuring Broker Server to Use Asynchronous Write Mode

Broker Server can be configured to write data to disk asynchronously. In this mode, Broker Server permits the operating system to cache its disk write operations in memory.

Enabling asynchronous write mode can significantly increase the performance of guaranteed documents. However, it also carries the risk of document loss in the event of a host system failure. If the host fails, any guaranteed documents that the operating system had written to cache, but had not yet committed to disk, are lost. If you require a high degree of reliability, you should not enable asynchronous write mode.

To enable the asynchronous write mode, you must modify the storage session parameters in the Broker Server configuration file (`awbroker.cfg`). If you are using separate storage sessions, you can enable asynchronous mode for the run-time session, the configuration session, or both sessions.

Use the `batchmode` argument to enable batching for the asynchronous disk writes. For information about configuring the batch mode, see [“webMethods Broker Server Configuration Parameters” on page 621](#).

To configure asynchronous-write mode

1. Stop the Broker Server.
2. On the machine where Broker Server is installed, locate the Broker Server's configuration file (`awbroker.cfg`) and make a backup copy of it.
3. Open the configuration file with a text editor and locate the `session-config` and `session-data` parameters, which are underscored in the example below. (If you are running a combined storage session, your configuration file will contain only the `session-config` parameter.)

```

.
.
.
snmp=1
eventlog=1
internal=1
session-config=qs:///var/opt/BrokerStorage/BrokerConfig.qs
session-data=qs:///var/opt/BrokerStorage/BrokerData.qs?max_cache_size=512

```

4. To specify the cache size for a storage session, append the "async" argument to the session's URL.

Because the storage session is specified as a URL, you must include the "?" separator if the `async` argument is the first argument in the query string. If the URL already includes a query string, use the "," character to separate the `async` argument from the other arguments in the string. Both forms are shown in the example below.

```

.
.
.
snmp=1
eventlog=1
internal=1
session-config=qs:///var/opt/BrokerStorage/BrokerConfig.qs?async
session-
data=qs:///var/opt/BrokerStorage/BrokerData.qs?max_cache_size=512,async

```

5. Save the configuration file.
6. Restart the Broker Server.

Viewing the Write-Mode Setting

To determine whether asynchronous write mode is enabled for the Broker Server, you can examine the `session-config` and `session-data` parameters in the Broker Server configuration file (`awbroker.cfg`), or you can use the following procedure to view the setting using My webMethods.

To view the write mode setting

1. In My webMethods: **Messaging > Broker Servers > Servers**.
2. In the **Broker Server List**, click the name of the server whose write-mode setting you want to view.
3. Click the **Utilization** tab.
4. In the **Details** column, click the  icon for the storage session (**Configuration Data** or **Run-Time Data**) whose write-mode setting you want to view.
5. Examine the **Session URL** value in the **Storage Session** section of the **Server Utilization** tab.

If the URL includes the `async` argument, asynchronous write mode is enabled for the storage session.

Running Multiple Broker Servers on the Same Host Machine

When you install webMethods Broker, the Software AG Installer creates a single instance of Broker Server on the host machine.

In some circumstances, you may want to run multiple instances of Broker Server in a single webMethods Broker environment or you may want to run multiple instances of webMethods Broker on a single host machine. webMethods Broker allows you to do both.

Running Multiple Broker Servers in a Single webMethods Broker Environment

Running multiple instances of Broker Server can be more advantageous than running multiple Brokers on a single Broker Server. For example, instead of hosting two Brokers on a single server, you might create two Broker Servers that each have a single Broker. By hosting the Brokers on individual Broker Servers, each Broker has a dedicated queue storage system with its own log file.

When you create a new instance of Broker Server, the `server_config` utility creates the following items for the new server:

- A data directory
- A configuration file
- A set of queue storage files

Additionally, it registers the new Broker Server with the Broker Monitor. If you are running on a Windows platform, the `server_config` utility will also register the new Broker Server as a Windows service.

When creating a new Broker Server, keep the following points in mind:

- Make sure the Broker Server's block of three ports does not interfere with the Broker Monitor port (the default port is 6850) or with ports used by other Broker Servers (or other applications) running on the host machine. For more information about port assignments, see "[Broker Server Communication Ports](#)" on page 76.
- To prevent documents from paging to virtual storage, ensure that the host machine is equipped with sufficient physical memory to support the volume of documents received by all Broker Server instances that it hosts.
- Each Broker Server on the machine requires access its own set of queue storage files. To prevent disk contention among the Broker Servers, queue storage files may need to be divided across different devices.

The Broker Server will start automatically after you create it unless you include the `-nstart` option in the `server_config` command.

Running Multiple Instances of webMethods Broker on the Same Host

You can run multiple instances of webMethods Broker on a single host machine. A webMethods Broker environment can run on each available IP address or port on the host.

One advantage to running multiple instances of webMethods Broker on a single host is for ease of management. For example, if the webMethods Broker environments share the same Broker Monitor port, you can discover and manage all of the Broker Servers running on that port in one Broker Server list through My webMethods.

There are several other reasons why you may want to run multiple instances of webMethods Broker on the same host machine. When you upgrade or change a Broker Server configuration on a production system, all Broker Servers in the same webMethods Broker environment must be stopped and restarted, disrupting the production system. However, when running multiple instances of webMethods Broker, each installation of webMethods Broker is a stand-alone application with its own set of unique configuration files and its own Broker Monitor so maintenance on one Broker Server does not affect the availability of another. Running multiple instances of webMethods Broker gives you the ability to upgrade or uninstall one instance of a Broker Server independently of the others.

When running multiple instances of webMethods Broker on the same host machine, keep the following points in mind:

- During installation, Broker Monitor binds to a port and IP address on the host machine. If multiple webMethods Broker environments exist on the same host machine using the same IP address, make sure each instance of Broker Monitor is assigned a unique port number. For more information about port assignments, see [“Ports and Running Multiple Instances of Broker Monitor”](#) on page 65.
- On UNIX platforms, each Broker Monitor instance requires its own start up script. Ensure that each start up script file is unique between each Broker Monitor instance. For more information about the start up script, see [“Starting Broker Monitor”](#) on page 60.

Creating a Broker Server with the Default Log and Storage Files

To create a new Broker Server, you use the `server_config` utility.

You can create a Broker Server that uses separate storage sessions for metadata and runtime data. This is the recommended queue storage configuration. If you want to create a Broker Server that uses a combined storage session, or use the `server_config` utility to create Broker Servers, see [“`server_config create`”](#) on page 532.

The following procedure creates a Broker Server with the default log file and storage files sizes. If you want to specify the size and location of the log and storage files, see [“Creating a Broker Server that Specifies Size and Location of Files” on page 103](#).

Note: Unlike the installation process, which creates a default Broker, the following procedure does not create any predefined Brokers on the new Broker Server. After the Broker Server is created, you must define one or more Brokers on the Broker Server. For procedures, see [“Creating a Broker ” on page 140](#).

To create a Broker Server with the default log and storage files

1. On the machine where Broker Server is installed, navigate to the following directory:

webMethods Broker_directory/bin

2. Run the following command.

```
server_config create dataDir -p port -k license-file-path
[-d description ] [-nostart] [-S]
```

Where...	Specifies...
<i>dataDir</i>	The fully-qualified location of the data directory for the new Broker Server. Enclose the entire directory name in quotes if any portion of the name contains a space. The path that you specify must already exist although the data directory itself does not. For example, if you specify the data directory, C:\BrokerServers\data \BrokerServerEast01, the underlined portion of this structure must already exist. If the specified data directory exists when you run the server_config utility, be certain it is empty.
<i>port</i>	The base port for this Broker Server. Be sure that this port and the two ports below it are available for this Broker Server to use.
<i>license-file-path</i>	The absolute path to the license file for the Broker Server. The location must have the valid license file in XML format as provided by Software AG.
<i>description</i>	An optional description for this Broker Server. This description appears when status information is displayed for the Broker Server and aids in identification of the server.

Where...	Specifies...
	Note: You do not have to provide a description when you create Broker Server. You can add it later using My webMethods.

- Include the `-nostart` parameter if you do not want Broker Monitor to automatically start the Broker Server after it is created.
- Include the `-s` parameter to run the utility in "silent mode." In this mode, the utility suppresses informational messages and writes warnings and error messages to stdout:

Example in Windows:

```
server_config create
"c:\webMethods_Broker_directory\Broker\data\awbrokersversion\My Broker"
-p 6800 -k c:\license_file_directory\new_license.xml -d "East Region Broker"
-nostart
```

Example in Unix:

```
server_config create /var/opt/webmethods/awbrokersversion/myBroker -p 6800
-k /tmp/licenses/new_license.xml -nostart -s
```

Creating a Broker Server that Specifies Size and Location of Files

Use the following procedure to create a Broker Server that specifies the size and location of the Broker Server log and storage files.

To create a Broker Server with specified log and storage files

1. On the machine where Broker Server is installed, navigate to the following directory:

`webMethods_Broker_directory/bin`

2. Run the following command.

```
server_config create dataDir -p port -k license-file-path
[-d description ]
-session_config qs
  -qs_log_file logFile logSize
  -qs_storage_file storeFile storeSize reserveSize
  [-qs_storage_file storeFile storeSize storeReserve ...]
-session_data qs
  -qs_log_file logFile logSize
  -qs_storage_file storeFile storeSize storeReserve
  [-qs_storage_file storeFile storeSize storeReserve ...]
[-nostart] [-S]
```

For parameter descriptions, see [“server_config create” on page 532](#).

The following example illustrates a single run-time storage file on Windows:

```
server_config create "c:\webMethods Broker_directory\Broker\data\
awbrokersversion\My Broker" -p 6800
-k c:\license_file_directory\new_license.xml -d "East Region Broker"
-session_config qs
  -qs_log_file F:\BrokerStore\myBroker\configData.qs.log 20M
  -qs_storage_file F:\BrokerStore\myBroker\configData01.qs.stor 10G 100M
-session_data qs
  -qs_log_file F:\BrokerStore\myBroker\runtimeData.qs.log 256M
  -qs_storage_file F:\BrokerStore\myBroker\runtimeData01.qs.stor
10G 100M
-nostart
```

The following example shows multiple run-time storage files on UNIX:

```
server_config create /var/opt/webmethods/awbrokersversion/myBroker -p 6800
-k /tmp/licenses/new_license.xml
-session_config qs
  -qs_log_file /var/opt/webmethods/myBroker/configData.qs.log 20M
  -qs_storage_file /var/opt/webmethods/myBroker/configData01.qs.stor
10G 100M
-session_data qs
  -qs_log_file /var/opt/webmethods/myBroker/runtimeData.qs.log 256M
  -qs_storage_file /var/opt/webmethods/myBroker/runtimeData01.qs.stor
10G 100M
  -qs_storage_file /var/opt/webmethods/myBroker/runtimeData02.qs.stor
10G 100M
-nostart -s
```

Deleting a Broker Server

You use the `server_config` utility to delete a Broker Server. When you delete a Broker Server, the `server_config` utility does the following:

- It deletes the Broker Server configuration file, queue storage files, and message log files.
- It removes the Broker Server from the Broker Monitor's configuration file.
- It removes the Broker Server from the Windows Service List (Windows only).

To delete a Broker Server

1. Stop the Broker Server.
2. On the machine where Broker Server is installed, navigate to the following directory:

```
webMethods Broker_directory/bin
```

3. Run the following command.

```
server_config delete dataDir [-f] [-S]
```

Where *dataDir* specifies the Broker Server's data directory. Enclose *dataDir* in quotes if any part of the directory path includes a space character.

- Include the `-f` parameter to suppress the confirmation message that the utility ordinarily issues before it deletes the Broker Server.

- Include the `-S` parameter to run the utility in "silent mode." In this mode, the utility suppresses informational messages and writes warnings and error messages to stdout.

See “[server_config delete](#)” on page 538 for a complete description of the parameters associated with this command.

4. Example:

```
server_config delete
"c:\webMethods Broker_directory\Broker\data\awbrokersversion\My Broker"
```

Viewing Status Information for a Broker Server

Use the following procedure to display basic status information about all the Broker Servers that you are managing and view detailed status information for a particular Broker Server.

To view status information

1. In My webMethods: **Messaging > Broker Servers > Servers**.

The **Broker Servers List** displays the following information:

Column	Description
	<ul style="list-style-type: none"> ■ The Broker Server is on line and operating normally.
	<ul style="list-style-type: none"> ● The Broker Server cannot be reached. This might occur because: <ul style="list-style-type: none"> ■ The Broker Server is not currently running. ■ The URL specified for this Broker Server is incorrect. ■ My webMethods was not able to complete the connection within the timeout period. To retry the connection, click Go on the Search tab.
	<ul style="list-style-type: none"> ▲ -The Broker Server is online, but one of the following abnormal conditions exist: <ul style="list-style-type: none"> ■ The license is expired or about to expire. ■ The Broker Server is running low on disk space. ■ The Broker Server currently has the maximum permitted number of connections.
<p>Note: You can hover over the  or  icon to obtain more information about the condition of the Broker Server.</p>	

Column	Description
Connected	<p>Indicates the state of the connection between My webMethods and Broker Server as follows:</p> <ul style="list-style-type: none"> ■ Yes. My webMethods has successfully connected to the Broker Server and the Broker Server is running. ■ Error. My webMethods could not establish a connection with the Broker Server. ■ In Progress. My webMethods is in the process of completing a connection with the Broker Server. ■ Not Started. My webMethods has not yet started to initiate a connection with this Broker Server. ■ Closed. My webMethods is in the process of ending its connection with this Broker Server.
Connections	The number of connections (SSL and non-SSL) that currently exist with this Broker Server.
SSL	Indicates the availability of SSL on the Broker Server. If this column displays "Not Configured," the Broker Server keystore and identity have not yet been specified and the server cannot communicate using SSL. If SSL is configured but the connection cannot be established, this column displays "Error". For information about configuring a Broker Server's SSL feature, see "Managing Broker Servers" on page 69 .
Version	The version of software installed on this Broker Server.

- Click the name of the Broker Server or click  **Details** to view the following details for a Broker Server.

Field	Description
Description	Description assigned to the Broker Server.
Status	Indicates the current state of the Broker Server. See description in step 1.
Basic Identity	Indicates the outbound credentials that enable the Broker Server to identify itself to other Brokers in the territory, cluster, or gateway.

Field	Description
SSL	Indicates the availability of SSL on the Broker Server. If this field says "Secure Sockets Layer needs to be configured," the Broker Server keystore and identity have not yet been specified and the server cannot communicate using SSL.
Access Control	Link used to configure access control for the Broker Server. For information about this field, see “Configuring Access Control Lists” on page 333.
FIPS Mode	<p>Indicates whether FIPS mode is enabled or disabled as follows:</p> <ul style="list-style-type: none"> ■ Enabled by OpenSSL version < Version number >. Indicates that FIPS mode is enabled on your Broker Server by OpenSSL, where <i><version number></i> indicates the version of OpenSSL. ■ Disabled. Unsupported Broker Server version. Indicates that FIPS mode is disabled because your version of Broker Server does not support FIPS. <div style="border: 1px solid gray; padding: 5px; margin: 10px 0;"> <p>Note: FIPS mode is only supported on Broker Server version 8.0 and later.</p> </div> <ul style="list-style-type: none"> ■ Disabled. Indicates that you did not enable FIPS although your Broker Server version supports FIPS.
Connections	The number of connections (SSL and non-SSL) that currently exist with this Broker Server.
Filter Collation Locale	<p>The locale that Broker Server will use when performing string comparisons during the evaluation of a subscription filter. The character order associated with a locale determines whether a character is greater than, less than, or equal to another character when two strings are compared.</p> <p>If you change the Filter Collation Locale, you must restart the Broker Server to put it into effect.</p>
Version	The version of software installed on the Broker Server.
License Expires	The time until which this Broker Server's license is valid.

Field	Description
Created	The date and time when this Broker Server was created.
Pause Publishing	Allows you to pause and resume publishing on the Broker Server.

Broker Server License

You require a license file to install Broker Server. During installation of Broker Server, you are prompted to specify the location of the license file. Therefore, please ensure that the license file is in a location that will be accessible during the installation, such as on the local file system.

Broker Server version 8.0 and later installations check for license information in a license file, while installations of earlier Broker Server versions use a license key. You use My webMethods to update the license file (for Broker Server version 8.0 and later) as well as the license key (for earlier versions of Broker Server).

The license key is a parameter in the awbroker.cfg file that enables Broker Server to operate in full capacity until a specified date and time. You receive the server's license-key in the form of an XML file (for example, licensePIF80.xml) from Software AG, when you obtain your license. The license key is one of the parameters that you use to configure a Broker Server at the time of installation.

When you start Broker Server, it checks for the presence of a license file at the location that the awbroker.cfg variable "license-key" points to.

- If the license-key file is missing or if it is invalid, Broker Server writes an error to the journal log and exits. You cannot start Broker Server if the license-key file is missing or invalid.
- If the license has expired, Broker Server will start, but it will not accept any documents that clients publish (clients can continue to retrieve documents that have already been published). Broker Server writes a message to the system log to indicate that it is running under an expired license. When Broker Server is operating with an expired license, the ▲ icon appears in the Broker Server list to indicate that Broker Server is not fully operational.

If the license expires while the Broker Server is running, the same sequence of steps occurs. That is, the Broker Server writes a message to the system log and begins rejecting all publish requests.

Contents of License File

A license file contains the sales, product, license, and Broker Server information.

The following table shows the sales information contained in the license file:

Attribute	Description
Serial Number	A ten-digit number with leading zero (0) for Broker Server.
License Key	A unique alphanumeric key, 32 characters long.
Customer ID	A unique alphanumeric ID that Software AG assigns each customer.
Customer Name	Name of the customer to whom Software AG distributed this license.
Contract Details	Any information specific to your license contract with Software AG.
License Type Details	The type of license you own (for example, External, Internal).

The following table shows the product information contained in the license file:

Attribute	Description
Expiration Date	<p>The date of expiration of your license. The license file displays:</p> <ul style="list-style-type: none"> ■ "Unlimited", if the license never expires. ■ The date of license expiration in either YYYY/MM/DD or DD/MM/YYYY format. ■ The number of days this license is valid.
OS	<p>The operating systems on which this license is supported. Multiple operating systems are separated by commas.</p> <p>If the license is supported on all operating systems that Broker Server is supported on, the license file displays "any".</p>
Product Code	A unique code assigned by Software AG to the Broker Server product (for example, PIF).
Product ID	A unique code assigned by Software AG to the Broker Server product (for example, PIF80FSET).

Attribute	Description
Product Name	Name of the product for which you obtained the license (for example, Software AGwebMethods Broker).
Product Version	Version number of the Software AG product (for example, 9.6).
Usage	Indicates whether the license is for production environment or test environment.

The following table shows the license information contained in the license file:

Attribute	Description
Price Unit	Price unit information.
Price Quantity	Price quantity information.

The following table shows the Broker Server information contained in the license file:

Attribute	Description
Product Code	A unique code assigned by Software AG to the Broker Server product (for example, PIF).
Product Version	Version number of Broker Server (for example, 9.6).

Viewing and Changing the License Using My webMethods

If Broker Server is running, you can use My webMethods to view and change the license.

To view or change the license using My webMethods

1. In My webMethods: **Messaging > Broker Servers > Servers**.
2. In the **Broker ServerList**, click the server whose license information you want to view.
3. Click the **License** tab.
4. If you want to update the license, click the **Change** button.
5. In the **Upload a New Server License File** dialog box, type the full path of the new license file, or click **Browse** to navigate to the file.
6. Click **Apply**.

Viewing and Changing the License Using `server_config`

If Broker Server cannot be started due to a missing or invalid license, you must use the `server_config` utility to change the license.

To change the license from the command line (using `server_config`)

1. Stop Broker Server.
2. On the machine where Broker Server is installed, navigate to the following directory:

```
webMethods Broker_directory/bin
```

3. Run the following command:

```
server_config update dataDir -k license-file-path
```

Where...	Specifies...
<i>dataDir</i>	The fully-qualified location of the Broker Server's data directory. Enclose the entire directory name in quotes if any portion of the name contains a space.
<code>license-file-path</code>	The absolute path to the license file for the Broker Server. The location must have the valid license file in XML format as provided by Software AG.

For example:

```
server_config update
"c:\webMethods Broker_directory\Broker\data\awbrokersversion\My Broker"
-k c:\license_file_directory\new_license.xml

server_config update /var/opt/webmethods/awbrokersversion/myBroker
-k /tmp/licenses/new_license.xml
```

4. Start Broker Server.

Monitoring Resource Utilization

Use the following procedure to examine statistics relating to CPU, memory, and queue storage utilization.

To display Broker Server resource utilization

1. In My webMethods: **Messaging > Broker Servers > Servers**.
2. In the **Broker Servers List**, click the server whose resource utilization you want to view.
3. Click the **Utilization** tab to display the following statistics.

Resource	Description
CPU	Used. The percentage of CPU time that this Broker Server has consumed on this host machine since the Broker Server was last started. <i>This statistic is not available if Broker Server is running on a Windows-based host machine.</i>
Virtual Memory	Total Capacity. The virtual memory capacity on the Broker Server host machine. Used. The amount of virtual memory currently in use by the Broker Server, and all the processes on the host machine.
Configuration Data	Total Capacity. The configured maximum size for each storage file associated with the configuration data session (for separate-storage configurations) or the combined storage session (for combined-storage configurations). Used. The amount of space in the storage files that is currently in use by this session.
Run-time Data	Total Capacity. The configured maximum size of the storage files associated with the run-time data session. Used. The amount of space in the storage files that is currently in use by this session.

- Click  **Detail** to display detailed information for any of the resources. For information about the details displayed for queue storage resources, see [“Viewing Queue Storage Utilization and Settings”](#) on page 94.

Logging

There are four logs associated with webMethods Broker:

- **The Broker Server log** records operational and error messages issued by Broker Monitor and Broker Server. You can refer to this log to find out when a Broker Server started running or stopped running. You can also refer to the log to examine error messages, warnings, and informational messages issued by a Broker Server while it was running. For information about configuring and viewing this log, see [“The Broker Server Log”](#) on page 113.
- **The Messaging log** records errors that My webMethods encounters when you use it to manage Broker Servers. You can refer to this log for details about error messages that you receive while using the Broker user interface. For information about configuring and viewing this log, see [“The Messaging Log”](#) on page 117.

- **The configuration audit log** records changes to configuration and security settings made to a Broker Server, Broker, client group, client, document type, territory and gateway. You can use the configuration audit log to maintain a trail of critical operations performed by users and to troubleshoot issues on the Broker Server. For information about configuring and viewing the configuration audit log, see [“Configuration Audit Logging” on page 597](#).
- **The document audit log** enables you to log published documents into the central audit database. For more information about using this logging facility, see [“ webMethods Broker Document Logging” on page 605](#).

The Broker Server Log

The Broker Server log receives messages from both the Broker Monitor and Broker Server. These messages represent significant events or unusual conditions that occur on the Broker Server. For example, when Broker Monitor starts a Broker Server or detects that Broker Server has stopped running, it writes a message to the log.

As an administrator, the information in the log is useful for troubleshooting purposes and for alerting you to abnormal operating conditions that arise on Broker Server.

Log File Names and Locations

The log consists of two files. Both files reside on the machine where Broker Server is installed. Broker Server writes to one file and Broker Monitor writes to the other. When you view the Broker Server log in My webMethods, you see entries from both files.

This process...	Writes its log entries to...
Broker Server	<ul style="list-style-type: none"> ■ <i>Broker Server Data directory \logmsgs</i> (Windows) ■ <i>Broker Server Data directory /logmsgs</i> (UNIX)
Broker Monitor	<ul style="list-style-type: none"> ■ <i>webMethods Broker_directory\bin\logmsgs.mon</i> (Windows) ■ <i>webMethods Broker_directory/bin/logmsgs.mon</i> (UNIX)

Message Types

Broker Server and Broker Monitor write three types of messages to the Broker Server log.

Type	Description
Alerts	Errors and conditions that require immediate administrative action (e.g., full disks, missing data files, a misconfigured network).
Warnings	Conditions that could potentially lead to an alert if ignored (e.g., low disk space, license expiration).
Information	General information (e.g., Broker Server started, Broker Server stopped).

You can configure Broker Monitor and Broker Server to specify which types of messages you want to include in the log. Additionally, when you view the log in My webMethods, you can filter the log by message type.

Including Log Messages in Other Logging Systems

Besides logging messages in the Broker Server log, Broker Monitor and Broker Server can optionally send their messages to the host machine's native logging facility (syslog on UNIX or the Event Log Service on Windows).

You can also configure Broker Server and Broker Monitor to send their messages as SNMP traps to other computers via the Simple Network Management Protocol (SNMP). To view these messages, you must have an SNMP management system configured with an SNMP viewer, such as Hewlett-Packard's OpenView.

To configure your SNMP viewer, you will need the Broker Server's management information base (MIB) file. This file (*activesw.mib*) is located in the *webMethods Broker_directory/lib* directory. To learn how to set up and configure your SNMP viewer, see the documentation for your system's management software.

Configuring the Logging Behavior of Broker Server

By default, Broker Server records all three message types to the Broker Server log and to the host machine's native logging facility. To modify this behavior, or to enable SNMP traps, you can use the following procedure.

Note: The following procedure configures the logging behavior of the Broker Server. Broker Monitor also generates entries in the Broker Server log. To configure Broker Monitor's logging behavior, see [“Configuring the Logging Behavior of Broker Monitor” on page 115](#).

To configure Broker Server logging behavior

1. In My webMethods: **Messaging > Broker Servers > Servers**.

2. In the **Broker ServerList**, click the server whose log you want to configure.
3. Click the **Server Log** tab and then click **Broker ServerLog Setting**.
4. Under **What to Log**, select the types of messages that you want Broker Server to include in the log. For a description of message types, see [“Audit Log Categories” on page 599](#).
5. Under **How to Log**, specify whether you want Broker Server to log entries to the host system's native logging system and/or an SNMP management system.

Note: Broker Server always logs messages to the Broker Server log. When you enable systems under **How to Log**, Broker Server will send log messages to the selected systems *in addition* to its own Broker Server log.

6. Click **OK**.

Configuring the Logging Behavior of Broker Monitor

By default, Broker Monitor records all three message types to the Broker Server log and to the host computer's native logging facility.

To modify this behavior or to enable logging to SNMP, you must edit the Broker Monitor configuration file (`awbrokermon.cfg`). The following table shows where the file resides for each platform:

On this platform...	The configuration file is located here...
Windows	<code>webMethods Broker_directory\bin\awbrokermon.cfg</code>
UNIX	<code>webMethods Broker_directory/bin/awbrokermon.cfg</code>

The parameters to configure the logging behavior of Broker Monitor are as follows:

Logging Parameter	Where...
<code>monitor-log-alert=logSetting</code>	<code>logSetting</code> is 0 (to disable) or 1 (to enable) the logging of alert messages. The default is 1.
<code>monitor-log-warning=logSetting</code>	<code>logSetting</code> is 0 (to disable) or 1 (to enable) the logging of warning messages. The default is 1.
<code>monitor-log-info=logSetting</code>	<code>logSetting</code> is 0 (to disable) or 1 (to enable) the logging of informational messages. The default is 1.

Logging Parameter	Where...
monitor-log-eventlog=logSetting	<i>logSetting</i> is 0 (to disable) or 1 (to enable) the logging of messages to the Windows Event Log Service. Default is 1 on Windows host machines.
monitor-log-syslog=logSetting	<i>logSetting</i> is 0 (to disable) or 1 (to enable) the logging of messages to the UNIX syslog. The default is 1 on UNIX host machines.
monitor-log-syslog-facility=logFacility	<i>logFacility</i> specifies the UNIX syslog facility to which messages are to be sent. The default is local5.
monitor-log-snmp=logSetting	<i>logSetting</i> is 0 (to disable) or 1 (to enable) the generation of SNMP traps. The default is 0.

Important: Add the logging parameters to the end of the awbrokermon.cfg file. Do *not* modify any of the non-logging parameters in the file.

The following is an example of a configuration file:

```
# This file is the configuration file for the webMethods Broker Monitor.
# It is maintained by the Broker Server configuration program.
# Do not edit this file manually!
server-6849.datadir=C:\SoftwareAG\Broker\data\awbrokersversion\default
server-6849.exec=C:\SoftwareAG\Broker\bin\awbroker.exe
server-6849.version=9.6
server-6849.port=6849
server-6845.datadir=C:\SoftwareAG\Broker\data\awbrokersversion\BrokerServerE01
server-6845.exec=C:\SoftwareAG\Broker\bin\awbroker.exe
server-6845.version=9.6
server-6845.port=6845

monitor-log-info=0
monitor-log-syslog=1
monitor-log-syslog-facility=local7
monitor-log-snmp=1
```

Tip: If you want to accept the default behavior for a parameter, simply omit the parameter from the configuration file. For example, if you want Broker Server to log alerts and warnings (the default behavior), but not informational messages, include the "monitor-log-info=0" parameter in the configuration file and omit the parameters for the other two message types.

Viewing the Broker Server Log

Use the following procedure to view the Broker Server log.

To view the Broker Server log

1. In My webMethods: **Messaging > Broker Servers > Servers**.
2. In the **Broker ServerList**, click the server whose log you want to view.
3. Click the **Server Log** tab to display the log entries.
 - When you view the **Server Log** tab, My webMethods displays the current log. The tab is not updated unless you click **Refresh** to update the display.
 - To display the messages for a particular period, select the time period from the **Display Log Entries** list and click **Refresh** to update the display.
 - For more information about a particular message, see the *webMethods Error Message Reference* on the Empower Product Support Web site. You can look up a message by its code number.

Purging the Broker Server Log

Use the following procedure to remove old entries from the Broker Server log.

To purge the Broker Server log

1. In My webMethods: **Messaging > Broker Servers > Servers**.
2. In the **Broker ServerList**, click the server whose log you want to purge.
3. Click the **Server Log** tab then click **Purge Log** to display the Purge Broker Server Log page.
4. From the **Purge Log Entries** list, select the range of messages that you want to remove from the log.
5. Click **Purge**.

The Messaging Log

When you use My webMethods to manage Broker Servers, errors and other operational messages that are generated by the Broker user interface are written to the Messaging log. You can refer to this log to obtain detailed information (including stack trace information) about errors that you experience in the Broker user interface.

Message Types

Each log message that the Broker user interface generates has a message type (also called a *log level*) of either Error or Verbose. In general, the message type indicates the severity or importance of the message.

You can configure the Broker user interface to specify which types of messages you want it to include in the log. Additionally, when you view the log, you can filter the display by message type.

Log File Names and Locations

The Messaging log resides on My webMethods Server. Each user account that uses the Broker user interface has its own Messaging log.

The Messaging log files are located in this directory on the My webMethods Server:

My webMethods Server_directory/Broker/logs

The log is made up of two files: an *activity* file and an *error* file. The activity file contains the log messages and the error file contains the stack traces associated with logged errors and exceptions.

The name of the file indicates the type of information it contains and the user account to which it belongs as follows.

```
userID.activity.xmluserID.error.xml
```

For example:

```
Administrator.activity.xml
Administrator.error.xml
yourUserID.activity.xml
yourUserID.error.xml
```

Continuous versus Daily Log Files

You can configure the Broker user interface to maintain the Messaging log as a *continuous* log or a *daily* log. When you use a continuous log, the Broker user interface maintains your log in a single pair of error and activity files (e.g., yourUserID.activity.xml and yourUserID.error.xml)

If you use a daily log, the Broker user interface opens a new pair of error and activity files each day. As shown below, the filename indicates date associated with the messages in the log.

```
userID.activityYYYYMMDD.xmluserID.errorYYYYMMDD.xml
```

For example:

```
Administrator.activity20060714.xml
Administrator.activity20060715.xml
Administrator.error20060714.xml
Administrator.error20060715.xml
yourUserID.activity20060610.xml
yourUserID.activity20060714.xml
yourUserID.error20060610.xml
yourUserID.error20060714.xml
```

Tip: Using daily logs makes it easier to purge (delete) old log entries. A continuous log cannot be purged without stopping your My webMethods Server. For additional information about purging the log, see [“Purging the Messaging Log that Uses Daily Files”](#) on page 120.

Viewing the Messaging Log

Use the following procedure to view the contents of your Messaging log.

To view the Messaging log

1. In My webMethods: **Messaging > Log**.
The **Messaging Log** tab displays entries that were generated in your Messaging log during the last day.
2. Use the following fields to specify which portion of the log you want to view and click **Refresh** to update the display.

Use this field...	To specify...
Display Log Entries within	The time period that you want to view.
Log Level	The types of messages that you want to view. You can specify either Error or Verbose. When you select a log level, the tab displays messages of the selected level and higher. For example, if you select Verbose , the tab will show messages of all types.

- When you view the Log page, My webMethods displays the log's current contents. The display is not updated unless you click **Refresh** to update the page.
- For more information about a particular message, see the *webMethods Error Message Reference* on the Empower Product Support Web site. You can look up a message by its ID number.

Configuring the Messaging Log

Use the following procedure to specify the types of messages that you want to include in your Messaging log and to specify whether you want to use continuous or daily log files. For more information about continuous and daily logging, see [“Continuous versus Daily Log Files” on page 118](#).

Note: Your configuration changes affect the Messaging log for your user account. If there are other administrators that use My webMethods to manage Broker Servers, they must configure the log for their accounts.

To configure the Messaging log

1. In My webMethods: **Messaging > Settings**.
2. Select the **Tool Log** tab.

3. In **Log Level**, set the level at which you want messages logged. The Broker user interface logs all messages that are of the selected level or higher.
4. Enable the **Log to Single File** option if you want to maintain continuous (that is, not daily) log files.
5. Click **Apply** to save your changes.

Purging the Messaging Log that Uses Daily Files

To purge the Messaging log, you must delete the associated log files from the file system. If you keep daily log files, you can simply delete the log files from previous days.

To purge a Message log that uses daily log files

1. On the machine where My webMethods Server is installed, navigate to the directory:
My webMethods Server_directory/Broker/logs
2. Locate and delete the log files for the user and time period that you want to purge. As shown below, the name of the log file will indicate the user to which the log belongs and the date on which it was created.

```
userID.activityYYYYMMDD.xml  
userID.errorYYYYMMDD.xml
```

For example:

```
yourUserID.activity20060610.xml  
yourUserID.error20060610.xml
```

- Be sure to delete the activity file and the corresponding error file for each daily log that you purge.
- Be sure to locate and delete all of the log files for the time period that you are purging. For example, if you want to purge your log of all messages issued before July 1, 2010, be sure to delete all the log files (for your account) whose names are marked 20060631 or earlier.

Purging the Messaging Log that Uses Continuous Files

If you use continuous files (that is, a single pair of activity and error files), you must stop My webMethods Server to purge the log.

To purge a Message Management log that uses continuous log files

1. Stop the My webMethods Server.
2. On the machine where My webMethods Server is installed, navigate to the directory:
My webMethods Server_directory/Broker/logs
3. Locate and delete the activity and error log files for the user whose log you want to purge. Be aware that this step deletes the entire log, including the most recent

entries. With continuous log files, there is no way to purge messages for a specified time period. The following shows what the names of the log files look like.

```
userID.activity.xmluserID.error.xml
```

A new, empty pair of log files will be created the next time the user accesses the Broker user interface.

Internet Protocol Support

webMethods Broker supports both IPv4 and IPv6. When you configure webMethods Broker for IPv6, specify the IP address of Broker Monitor and Broker Server in the IPv6 specified format (for example, [2a00:2000:4061:0:9152:35b:95ea:da8f]:5849). Note that webMethods Broker does not support SNMP protocol in an IPv6 environment.

Backing Up and Restoring a Broker Server

As with all critical data resources, the potential exists for a physical failure to leave Broker Server storage files in an inconsistent or non-readable state. In these situations, the method of recovery is to first attempt to correct the existing files using the Broker Storage Utility as described in “[webMethods Broker Storage Utility](#)” on page 567. If this approach fails, the next step is to replace the existing storage files with the most current backup.

This section describes how to backup the metadata associated with your Broker Server. The backup procedures in this section produce a snapshot of the storage files that you can use to restore a queue storage system that has been irreparably damaged.

Note: There are other reasons that you might want to create a copy of the Broker Server's metadata. For example, you might want to replicate some or all of the server's metadata on another machine. Or, if you are updating a set of client groups or document types, you might want to copy the objects beforehand in case you need to "back out" the changes at a later time. The backup and restore procedures in this section are not suitable for these purposes. To copy metadata for these purposes, use the procedures described in “[Managing Client Groups](#)” on page 171.

Metadata that Is Backed Up

The procedures described in this section back up the following metadata:

- Brokers
- Document types
- Client groups
- Territory Membership

- Cluster Membership
- Gateways
- Explicit-destroy client state (except queue contents)
- ACLs
- Broker Server SSL configuration

Run-time information (document instances and client queues) is not backed up.

When to Perform a Backup

You must create backups your Broker Server regularly. The frequency of these backups depends on the dynamic nature of your metadata. For example, if you have explicit-destroy clients whose subscriptions change often, you might want to perform routine backups more frequently than if client subscriptions are fairly static.

In addition to performing a regular backup, it is critical that you back up your Broker Server immediately after making any of the following changes:

- Adding a new Broker
- Adding, deleting, or modifying document types
- Adding, deleting, or modifying client groups
- Creating territories or changing their membership
- Changing ACL assignments
- Adding, deleting, or modifying a gateway
- Changing the keystore location, password, or Broker Server SSL identity

Other Files That You Should Backup

When you back up the Broker Server metadata, also back up the following files:

- Broker Server's configuration file (awbroker.cfg)
- Broker Monitor's configuration file (awbrokermon.cfg)
- Broker Server's keystore file (if the Broker Server is SSL-enabled)

Important: Always store the backup files in a location other than on the Broker Server host machine.

Backing Up Territories and Gateways

If a Broker Server hosts a Broker that is a member of a territory, it is critical that you back up *all of the* Broker Servers that participate in the territory after making any of the following types of changes:

- Modifying the membership of the territory
- Adding, deleting, or modifying document types
- Adding, deleting, or modifying client groups
- Changing territory ACL assignments
- Adding, deleting, or modifying a gateway in the territory

Backing up all of the Broker Servers in the territory ensures that you can restore any one of the Broker Servers and the participating Brokers on that server will be in sync with the rest of the territory.

If you restore a Broker that is not in sync with the other members of the territory, you will have to either:

- Resynchronize the Broker by removing it from the territory and adding it to the territory again (which will cause it to lose any gateways that reside on the Broker).
- Manually update the client groups and document types on the restored Broker to synchronize them with the rest of the territory.

Choosing the Correct Backup Procedure

The way in which you backup your Broker Server depends on whether it uses separate or combined queue storage sessions.

- If your Broker Server uses *separate* storage sessions, you can use the `server_conf_backup` utility to perform the backup as described in [“Backing Up a Separate Storage Configuration” on page 123](#).
- If your Broker Server uses a *combined* storage session, you must use the file system's copy command to perform the backup as described in [“Backing Up a Combined Storage Configuration” on page 129](#).

Backing Up a Separate Storage Configuration

If your Broker Server is configured to use separate queue storage sessions, you use the `server_conf_backup` utility to back up the queue storage files that contain the Broker Server's configuration data. You do not need to stop the Broker Server to use this utility. It is designed to run while the Broker Server is online.

Files that Are Backed Up

The backup utility copies the following files into a single compressed file.

Name	Description
BrokerConfig.qs	The catalog for the configuration storage session. The catalog records the names and sizes of the log and storage files.
*.qs.stor	The storage files for the configuration storage session.
BrokerConfig.qs.log	The log file for the configuration storage session. (The default name for this file is "Broker Config.qs.log," however its actual name will depend upon how it was specified when the Broker Server was created.)

Important: When creating the backup, the utility takes steps to ensure that the files are in a consistent state. Never attempt to back up these files using the file system's regular copy command. Always use the backup utility to back them up.

Backing Up an ACL-Protected Broker Server

To back up an ACL-protected Broker Server, you must be authorized to administer the Broker Server. Depending on whether you have configured basic authentication or SSL authentication, you must supply the credentials when you run the utility:

- For basic authentication, supply the user name and password.
- For SSL authentication, supply:
 - The name and location of the keystore file
 - The password for the keystore file
 - The distinguished name that identifies you as an authorized administrator of the Broker Server

To back up the Broker Server using the `server_conf_backup` utility

1. On the machine where Broker Server is installed, navigate to the following directory:
Software AG_directory/Broker/bin
2. Run the following command:

```
server_conf_backup backupFile -h host:port [-k keystore -p password
-tttruststore] [-bu basic-auth-user -bp basic-auth-password] [-e] [-o] [-v]
```

Include...	To...
-e	Encrypt transmissions between the <code>server_conf_backup</code> utility and the Broker Server you are backing up.
-o	Overwrite the backup file if it already exists.
-v	Enable verbose mode, which displays informational messages about the backup process.

For other parameter descriptions, see “[server_conf_backup](#)” on page 527.

The following example illustrates how to back up a non-protected Broker Server:

```
server_conf_backup /var/backups/Brokers/atlas6842/bkp01.zip -h localhost:6842 -v
```

The following example illustrates how to back up an ACL-protected SSL authenticated Broker Server:

```
server_conf_backup /var/backups/Brokers/atlas6842/bkp01.zip -h atlas:6842 -o -k /var/MyKeys.cert -p axl545j -t CN=hdavis,OU=IT,O=webmethods,L=Denver,ST=CO,C=US
```

The following example illustrates how to back up an ACL-protected basic authenticated Broker Server:

```
server_conf_backup /var/backups/Brokers/atlas6842/bkp01.zip -h atlas:6842 -o -bu user1 -bp password123
```

Restoring a Separate Storage Configuration

You use the `server_conf_restore` utility to restore the Broker Server's configuration storage session from a backup file that you created using the `server_conf_backup` utility.

Because there is no guarantee that the data in the existing run-time session is consistent with the metadata that you are restoring, you must reinitialize the run-time storage files before restoring the configuration session from the backup file. All client queues and guaranteed documents that are currently in the run-time storage session, will be lost.

To reinitialize the run-time storage session, you delete the existing files and create new, empty run-time storage files in their place. To do this, you must create a new Broker Server using exactly the same settings as your original Broker Server. This creates an initialized environment into which you can restore the configuration session from the backup file.

The following procedure describes the steps to do this. During this procedure, you will:

- Locate and delete all of the existing queue storage files.
- Create a new, empty Broker Server in the same data directory as the original Broker Server. (This step initializes the run-time storage session).

- Restore the configuration session from the backup file.

Important: If you are restoring an SSL-enabled Broker Server and you have modified the location, password, or content of the keystore file, you will need to restore the copy of the keystore file that you made when you created the backup file of the configuration session.

To restore the Broker Server using the `server_conf_restore` utility

1. Stop Broker Server.
2. Perform the following steps to obtain information about this Broker Server's queue storage system. You will use this information in later steps:
 - a. Navigate to the following directory:


```
webMethods Broker_directory/bin
```
 - b. Run the following command to display details about the existing queue storage system:


```
server_config storage dataDir -list
```

where `dataDir` is the path to Broker Server's data directory. Enclose the entire directory name in quotes if any portion of the name contains a space.
 - c. Print or otherwise record the information that the command displays. You will use this information later when you create new instances of these files.
3. Delete all of the existing queue storage files for this Broker Server (that is, the files ending ".qs," ".qs.stor," and ".qs.log"). Use the output from step 2c to locate these files.
4. Perform the following steps to prepare the Broker Server to accept the restored files.
 - a. Move the Broker Server's `awbroker.cfg` file out of Broker Server's data directory and into a different directory. (You cannot complete the next step if the `awbroker.cfg` file exists in the Broker Server's data directory.)
 - b. Use the "`server_config create`" command (shown below) to initialize Broker Server's queue storage system. The following points apply when you run this command:
 - You must create queue storage files using the exact names, locations, and sizes as the files described in the report generated in step 2c.
 - Be sure to include the `-nostart` option to prevent Broker Server from starting after you run this command.
 - If you do not know the Broker Server's port number or do not find the license file, get that information from the original `awbroker.cfg` file that you moved in step 4a.

```
server_config create dataDir -p port -k license-file-path -nostart
-session_config qs configSessionParameters
-session_data qs dataSessionParameters
```

For example:

```
server_config create /var/opt/webmethods/awbrokersversion/myBroker -p
6800 -
nostart
-k /tmp/licenses/new_license.xml
-session_config qs
-qs_log_file /var/opt/webmethods/myBroker/configData.qs.log 20M
-qs_storage_file /var/opt/webmethods/myBroker/configData01.qs.stor 10G
100M
-session_data qs
-qs_log_file /var/opt/webmethods/myBroker/runtimeData.qs.log 1000M
-qs_storage_file /var/opt/webmethods/myBroker/runtimeData01.qs.stor 10G
100M
-qs_storage_file /var/opt/webmethods/myBroker/runtimeData02.qs.stor 10G
100M
```

- c. For additional information about the `server_config create` utility, see [“Creating a Broker Server with the Default Log and Storage Files” on page 101](#) and the [“server_config create” on page 532](#).
5. Run the following command to restore the configuration queue storage files to the Broker Server:

```
server_conf_restore dataDir backupFile -o
```

Where...	Specifies...
<code>dataDir</code>	The Broker Server's data directory. Enclose the entire directory name in quotes if any portion of the name contains a space.
<code>backupFile</code>	The name of the backup file.

For more information about these parameters, see [“server_conf_restore” on page 528](#).

For example:

```
server_conf_restore /var/opt/webmethods/awbrokersversion/myBroker
/var/backups/Brokers/atlas6849/bkp01.zip -o
```

6. In a UNIX environment, change the permissions of the restored files:
 - a. Change the ownership of all files ending with “.qs”, “.qs.stor”, and “.qs.log” to bin (if Broker Server was installed by the root) or to the user ID that was used to install Broker Server.
For example: `chown bin *.qs*`
 - b. Change the group ownership for all files ending with “.qs”, “.qs.stor”, and “.qs.log” to allow bin access (if Broker Server was installed by root) or to a group in which the user that installed Broker Server is a member.
For example: `chgrp bin *.qs*`

7. If your original Broker Server was configured to use any of the following features, add the appropriate settings to the new awbroker.cfg file in the data directory.

Feature	For information see...
Cache settings	“Configuring the Cache Settings” on page 95
Max Memory Size	“Configuring the max-memory-size Parameter” on page 83
Async Write Mode	“Configuring Broker Server to Use Asynchronous Write Mode” on page 98

8. Restart the Broker Server.
9. If this Broker Server hosts a Broker that is part of a territory, review the information in the following section to determine whether you need to perform additional steps to synchronize the restored Broker with the rest of the territory.

Resynchronizing a Restored Broker with its Territory

If you have restored a Broker Server that hosts a Broker that is a member of a territory, you must ensure that the restored Broker is synchronized with the other Brokers in the territory. Review the following points and perform the steps that apply to your situation.

- If you know for certain that the backup was created after the last metadata change was made to the territory (that is, the restored Broker has exactly the same set of client groups and document types as the other Brokers in the territory), you do not need to take any additional steps.
- If you are not certain that the restored Broker has same set of client groups and document types as the rest of the territory and that Broker *does not host a gateway*, you can perform the following steps to resynchronize it with the territory.
 1. Remove the Broker from the territory as described in [“Removing a Broker from a Territory” on page 363](#).
 2. Add the Broker back to the territory as described in [“Adding Brokers to a Territory” on page 362](#).

This action causes the Broker to resynchronize itself with the territory. Its client groups and document types will be brought up to date. Additionally, the subscription lists that it maintains for the other Brokers in the territory will be regenerated.

- If you are not certain that the restored Broker has same set of client groups and document types as the territory and if the Broker *hosts a gateway*, you will need to manually update its set of client groups and document types to match those in the territory. One way to do this is to export the definitions from one of the Brokers in

the territory and import them into this Broker. For procedures, see [“Managing Client Groups” on page 171](#).

Backing Up a Combined Storage Configuration

If your Broker Server uses a combined storage session and it has no Brokers that belong to a territory, you can back up the queue storage files using the file system's copy command.

Note: If your Broker Server hosts Brokers that belong to a territory, you must back up your Broker Server's metadata by exporting it to an XML or ADL file. If any of the Brokers include gateways, be sure to export the gateway definitions, too (available in XML export only). For information about the export feature, see [“Exporting and Importing Broker Metadata” on page 483](#).

Preparing for the Backup

Because queue storage for a combined storage session contains both metadata and run-time data, any backup copy you create with the file system's copy command has the potential to capture run-time data as well as metadata. Therefore, this type of backup is only suitable in an environment where you can easily halt all client activity and drain the queues of all guaranteed documents. If you create a backup that captures any guaranteed documents, those documents will be resent if that backup is ever restored.

If you cannot put your Broker Server in this state, then you should back up your Broker Server's metadata by exporting it to an XML or ADL file. For information about the export feature, see [“Managing Client Groups” on page 171](#).

Precautions and Alternatives

Because of the potential for inadvertently including run-time data in the backup, backing up queue storage using the file system's copy command is strongly discouraged. Alternatively, consider migrating the Broker Server to a new Broker Server that uses separate storage sessions. Such a configuration allows you to use the `server_conf_backup` utility, which backs up just the Broker Server's metadata. This utility also enables you to perform the backup while the Broker Server is running.

To back up Broker Server metadata using the file system copy command

1. Disconnect clients and drain the queues of guaranteed documents.
2. Stop the Broker Server.
3. Perform the following steps to locate all of the queue storage files associated with the Broker Server's queue storage system.
 - a. Navigate to the following directory:
webMethods Broker_directory/bin

- b. Run the following command to display details about the existing queue storage system:

```
server_config storage dataDir -list
```

where *dataDir* is the path to the Broker Server's data directory. Enclose the entire directory name in quotes if any portion of the name contains a space.

4. Locate and copy all of the queue storage files identified in the output from step 3b. If your Broker Server uses the default set of storage files, you would locate and copy the following files:
 - BrokerConfig.qs
 - BrokerConfig.qs.log
 - BrokerConfig.qs.stor

Important: If the Broker Server has multiple storage files, be certain that you copy them all.

5. Restart the Broker Server.

Restoring a Combined Storage Configuration

Use the following procedure to restore the queue storage files for a combined session from a backup that you made using the file system's copy command.

To restore Broker Server metadata using the file system copy command

1. Stop the Broker Server.
2. Perform the following steps to locate all of the queue storage files associated with the Broker Server's queue storage system.
 - a. Navigate to the following directory:

```
Software AG_directory/Broker/bin
```

- b. Run the following command to display details about the existing queue storage system:

```
server_config storage dataDir -list
```

where *dataDir* is the path to the Broker Server's data directory. Enclose the entire directory name in quotes if any portion of the name contains a space.

3. Locate and delete all of the queue storage files for this Broker Server as identified in the output from step 2b. If your Broker Server uses the default names for queue storage files, you would delete the following files:
 - Broker.qs
 - Broker.qs.log
 - Broker.qs.stor

4. Copy the queue storage files from the backup files to their original locations.
5. In a UNIX environment, change the permissions of the restored files:
 - a. Change the ownership of all files ending with ".qs", ".qs.stor", and ".qs.log" to bin (if Broker Server was installed by the root) or to the user ID that was used to install Broker Server.

For example: `chown bin *.qs*`
 - b. Change the group ownership for all files ending with ".qs", ".qs.stor", and ".qs.log" to allow bin access (if Broker Server was installed by root) or to a group in which the user that installed Broker Server is a member.

Example: `chgrp bin *.qs*`
6. Restart the Broker Server.
7. If this Broker Server hosts a Broker that is part of a territory, review the information in [“Resynchronizing a Restored Broker with its Territory” on page 128](#) to determine whether you need to perform additional steps to synchronize the restored Broker with the rest of the territory.

Pausing Document Publishing on the Broker Server

You can temporarily halt the publishing of documents on the Broker Server. Use the Pause Publishing feature when it is necessary to clear the client document queues on the Broker Server. The document queues should be cleared before performing certain Broker Server maintenance tasks such as increasing queue storage, performing a backup, or upgrading the Broker Server.

When document publishing is paused on a Broker Server, the Broker Server refuses all incoming documents. The documents that exist in the publishing client queues continue to be consumed by the subscribing clients until the queues are emptied.

To clear all client queues on the Broker Server

1. In My webMethods: **Messaging > Broker Servers > Servers**.
2. Click the name of the server in the **Broker Server List**.
3. Click the **Pause Publishing** button.

The publishing clients refuse incoming documents until publishing is explicitly resumed.
4. To restart publishing to the client queues, click **Resume Publishing**.

Configuring Broker Server for high throughput in high-latency and high-bandwidth networks

When Broker Server uses a single TCP/IP connection for Broker to Broker communication, there can be delays in message transmission in high-bandwidth and high-latency networks (WAN). These delays occur because a single TCP/IP connection cannot scale up to fully use the available bandwidth.

To deal with these delays, Broker Server can be configured for streaming and/or remote Broker parallel channels.

Streaming improves the performance of Broker Server when the servers are connected across high-latency networks. When streaming is not enabled between Broker Servers, the messages are transferred using the request/reply mode. In this situation, one batch of documents are sent, and the subsequent transfer happens only after receiving an acknowledgement message from the receiver. Hence, the performance is affected due to the round-trip time (RTT). However, when you configure streaming in Broker Servers, documents are sent in multiple batches without waiting for an acknowledgement from the receiver, thereby improving the performance of the server.

Parallel channels enable a Broker Server to use more than one TCP/IP socket for Broker to Broker communication, thus increasing throughput at the connection level. Test results show linear increase in message throughput when a remote Broker is configured for 5 channels (default) and 10 channels.

When you configure Broker Server for streaming, the performance of the server improves while routing small documents. When you configure parallel channels in Broker Server, the performance of the server improves while routing large documents. A combination of parallel channels with streaming improves performance of the server in case of small and large documents.

Configuring Streaming in Broker Servers

Streaming is configured globally for a Broker Server. That is, when you enable streaming in a Broker Server, the streaming configuration is applied to all the Broker connections (including the connections within a territory, a cluster, and across a gateway) that are connected to another streaming enabled Broker Server.

Note: Streaming is supported only for Broker Server versions 8.2 and above. The streaming functionality is not backward compatible with Broker versions 8.0 or earlier.

To enable streaming in a Broker Server

1. Stop the Broker Server.

2. On the machine where Broker Server is installed, locate the Broker Server's configuration file (`awbroker.cfg`).
3. Open the configuration file with a text editor, and set the `rbroker-streaming-enabled` configuration parameter as: `rbroker-streaming-enabled=1`
4. To specify the batch size of acknowledgements, set the value of the `rbroker-streaming-window-size` configuration parameter.
5. Set the `connection-sndbuf-size` and the `connection-rcvbuf-size` configuration parameters for optimizing the TCP/IP connection buffering.

Important: To enable large buffer size in Broker, ensure that your operating system also supports large TCP/IP buffer size.

For example, on a HP-UX machine, to enable streaming on a gateway separated by high latency network, set the send/receive buffer size to 1 MB.

- Set the TCP/IP send and receive buffer sizes as:

```
ndd -set /dev/tcp tcp_xmit_hiwater_def 1048576
```

```
ndd -set /dev/tcp tcp_rcv_hiwater_def 1048576
```

Note: If you want to check the existing TCP settings on your HP-UX machine, use

```
ndd -get /dev/tcp tcp_xmit_hiwater_def
```

```
ndd -get /dev/tcp tcp_rcv_hiwater_def
```

- Set the `connection-sndbuf-size` and `connection-rcvbuf-size` configuration parameters in the `awbroker.cfg` file as:

```
connection-sndbuf-size=1048576
```

```
connection-rcvbuf-size=1048576
```

Note: For more information on configuring high TCP/IP buffer support on your host machine where Broker is installed, see *RFC 1323* and your operating system guide.

6. Save the configuration file.
7. Restart the Broker Server.

For more information about the configuration parameters such as `rbroker-streaming-ack-timeout`, `rbroker-streaming-enabled`, `rbroker-streaming-window-size`, `connection-sndbuf-size`, and `connection-rcvbuf-size` used for streaming, see “[webMethods Broker Server Configuration Parameters](#)” on page 621.

Configuring Parallel Channels in Broker Servers

The parallel channel messaging feature enables Broker Servers to use more than one TCP/IP socket for Broker to Broker communication.

Broker Server uses the *baseport -3* and *baseport -4* ports. These ports must be available for parallel channel messaging.

In the communicating Broker Servers, you need to enable parallel channel messaging, and specify the number of sockets required for Broker to Broker communication.

Note: The parallel channel messaging feature supports basic authentication in both UNIX and Windows systems. Supports SSL only on a UNIX system.

When you configure parallel channels in a Broker Server, streaming is enabled, if you have not explicitly disabled streaming by specifying `rbroker-streaming-enabled=0` in the `awbroker.cfg` file. On a high-bandwidth network, you can achieve high speed message transfer when you configure parallel channels with streaming.

If streaming is enabled when you enable parallel channel messaging, set the `rbroker-streaming-window-size`, `connection-sndbuf-size`, and `connection-rcvbuf-size` configuration parameters as required. For information about configuring streaming, see [“Configuring Streaming in Broker Servers” on page 132](#).

To configure parallel channels for communication between Broker Servers

1. Stop the Broker Server.
2. On the machine where Broker Server is installed, locate the Broker Server's configuration file (`awbroker.cfg`).
3. Open the configuration file with a text editor.
4. To enable parallel channels, specify:

```
rbroker-parallelchannel-enabled=1
```
5. To specify the message size or the batch size limit that the remote Broker must use while sending messages to other Brokers, set the value in the `rbroker-max-send-events-size` parameter. Setting this parameter according to your network bandwidth avoids message delivery delays. Adjust the message size or the batch size limit according to the receive Broker log file size. The default value is 16MB. Maximum value recommended is 64MB. Adjust the receiving Broker Server's runtime log file size as required.
6. To define the number of sockets the remote Broker can request for communicating with another Broker, assign the value to the `rbroker-parallelchannel-count` parameter.

Make sure that the number of parallel channels requested by both the communicating Brokers is the same. If the number of parallel channels requested by

the two communicating Brokers is not the same, then the default value of 5 parallel channels are used for message communication.

7. Save the configuration file.
8. Restart the Broker Server.

Setting the Locale

If you are going to run webMethods Broker in a locale other than U.S. English, set your system to the proper locale before installation. However, this setting only affects the way the Software AG Installer starts your Broker Server during installation. When you restart the Broker Server, the Broker Server will use the system locale.

On UNIX systems, if you want to set the Broker Server to run using a different locale, set the environment variables `LC_ALL` and `LANG` to that locale.

On Windows systems, use the region and language settings on the Control Panel to set the language for displaying the date, time, currency, and number. Make sure you copy these locale settings to the system accounts.

Restart the system after you change the system locale.

5 Managing Brokers

■ Role of Broker	138
■ Creating a Broker	140
■ Deleting a Broker	141
■ Configuring a Default Broker	142
■ Using Document Logging	143
■ Routing Documents to the Dead Letter Queue	144
■ Viewing Status Information for a Broker	146
■ Viewing Basic Operating Statistics for the Broker	148
■ Managing Transactions	150

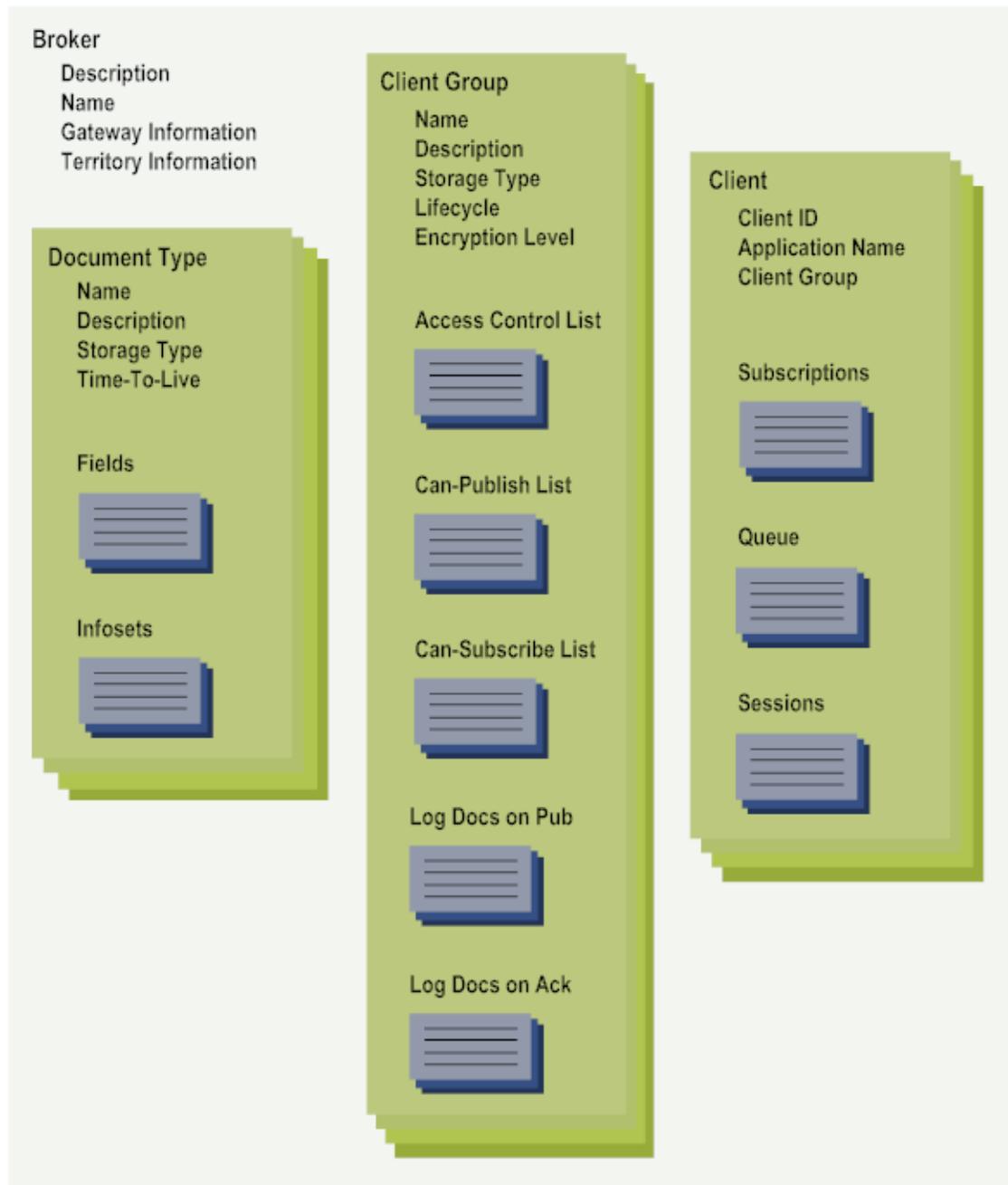
Role of Broker

Broker is an entity that resides on a Broker Server. When a client connects to Broker Server, it specifies the Broker with which it wants to interact.

Broker contains three key types of objects:

- Document types, which define the kinds of documents that clients of the Broker can exchange.
- Client groups, which are objects that represent a set of client properties and permissions. Any client that connects to a Broker must declare the client group to which it belongs. If the Broker Server on which the Broker is running is SSL enabled, an access control list (ACL) can be assigned to the client group. By restricting access to a client group, you restrict access to the Broker.
- Clients, which are state objects that represent the client programs that connect to the Broker. The state object maintains a client's list of subscriptions and its queue. If a client program connects to the Broker as an "explicit destroy" client, the Broker maintains the state object even if the client program is physically disconnected from the Broker.

A Broker contains document types, client groups, and client state objects



Although we tend to think of Brokers as functioning objects, they are actually data structures that encompass a collection of objects upon which processing threads within Broker Server operate. In this sense, a Broker is more like a container that enables you to partition the publish-and-subscribe capabilities of a Broker Server into separate domains or namespaces.

Creating a Broker

When you install Broker Server using the Software AG Installer, the Installer creates one Broker, called "default," on the Broker Server. You can use the following procedure to create additional Brokers on the Broker Server if necessary.

When you create Brokers, the JMS client groups (IS-JMS, JMSSClient, and EventMembers) and Broker objects (JMS::Temporary::Queue, JMS::Temporary::Topic, Event::WebM::EventSubscriptionMgmt::SubscriptionChange, and Event::WebM::Deployment) are created by default.

By default, the "can-publish" and "can-subscribe" permissions for Broker objects:

- JMS::Temporary::Queue and JMS::Temporary::Topic are assigned to IS-JMS and JMSSClient.
- Event::WebM::EventSubscriptionMgmt::SubscriptionChange is assigned to EventMembers.

You do not need to explicitly initialize the Brokers using the JMSAdmin command-line tool. However, you can use the initialize broker JMSAdmin tool to recreate the default JMS definitions on the currently connected Broker in the event the JMSSClient client group, or the JMS::Temporary::Queue and JMS::Temporary::Topic Broker objects were deleted using My webMethods after Broker creation.

You can also use this procedure to add a Broker to a Broker Server that you have created using the server_config utility. Unlike the installation process, the server_config utility does not install a default Broker on a new Broker Server. You must explicitly create the Broker yourself.

Note: Although you can create multiple Brokers on a Broker Server, this capability is provided mainly as a convenience for development organizations that need to develop and test against multiple Broker environments. Running multiple Brokers on a Broker Server is not necessary in a production environment.

To create a Broker

1. In My webMethods: **Messaging > Broker Servers**.
2. In the **Broker Servers List**, click the Broker Server on which you want to create the Broker.
3. Select the **Brokers** tab and click **Add Broker**.
4. Complete the following fields.

In this field...	Specify...
Broker Name	A name for the new Broker.

In this field...	Specify...
	<ul style="list-style-type: none"> ■ The name must be unique among all Brokers on the Broker Server. If the Broker will eventually become a member of a territory or a cluster, its name must also be unique among the members of the territory or cluster. ■ The name cannot contain the following characters: @ / : ■ The first character must not be #. ■ The name cannot exceed 255 characters.
Description	Optional. A description for the new Broker.

5. If you want this Broker to act as the default Broker, enable the **Default Broker** option. For additional information about the default Broker, see [“Configuring a Default Broker” on page 142](#).
6. Click **Add**.

Deleting a Broker

Use the following procedure to delete a Broker from a Broker Server. When you delete a Broker, you immediately disconnect and delete all clients that exist on it. You also permanently delete all of its client groups and document types.

If the Broker that you are deleting is a member of a Broker territory, My webMethods will remove it from the territory before deleting it. However, if at least one other member of the territory is not online when you delete the Broker, the Broker's "Remote Broker objects" will become orphaned. You will need to manually delete these objects when the other Brokers are brought back online. For procedures, see [“Deleting a Remote Broker Object” on page 366](#).

If you delete a Broker has been designated as the "default" Broker, the Broker Server *will not* automatically designate another Broker to function as a default. If your client applications are written connect to the default Broker (that is, your clients connect to Broker Server without explicitly specifying a Broker), you will need to designate an appropriate Broker to function as a default. For additional information about a default Broker, see [“Configuring a Default Broker” on page 142](#).

To delete a Broker

1. In My webMethods: **Messaging > Broker Servers**.
2. In the **Broker Servers List**, click the Broker Server on which the Broker resides.
3. Click the **Brokers** tab.

4. Select the check box beside the name of the Broker that you want to delete and click **Delete**.

Configuring a Default Broker

You can optionally assign a Broker to function as the *default Broker* for a Broker Server. The default Broker is the one to which a Broker Server will route client requests that are not addressed to a specific Broker.

When you install Broker Server using Software AG Installer, it installs one Broker, which it designates it as the default. You can change this designation if you install additional Brokers on the Broker Server.

A Broker Server is not required to have a default Broker. However, if your Broker clients are designed to connect to the default Broker (that is, they do not explicitly specify a Broker name when they connect to Broker Server), you must designate a default. Otherwise, requests from those clients will be rejected.

Determining Which Broker Is the Default

Use the following procedure to determine whether a default Broker has been designated for a Broker Server.

To determine which Broker is the default

1. In My webMethods: **Messaging > Broker Servers**.
2. In the **Broker Servers List**, click the Broker Server on which the Broker resides.
3. Click the **Brokers** tab.
4. Scan the **Default** column. The default Broker (if one has been designated for this Broker Server) will have **Yes** displayed in this column.

Designating a Default Broker

Use the following procedure to designate a Broker to act as the default Broker on a Broker Server.

To designate a default Broker

1. In My webMethods: **Messaging > Broker Servers**.
2. In the **Broker Servers List**, click the Broker Server on which the Broker resides.
3. Select the **Brokers** tab and click the Broker that you want to designate as the default.
4. Enable the **DefaultBroker** option and click **Apply**.
5. Click **Close** to return to the **Brokers** tab.

Using Document Logging

The document-logging facility enables you to optionally log instances of documents in the central audit database.

To use the document-logging facility, you must install and configure the Logging Utility on the Integration Server. Then you must take the following steps on the Broker:

- Enable the **Document logging** option on the Broker.
- Select the types of document that you want to log.
- Specify whether you want to log the documents when they are published or when they are received by subscribers (or both).

For complete procedures, see “[webMethods Broker Document Logging](#)” on [page 605](#).

Determining Whether Document Logging Is Enabled

Use the following procedures to determine whether the **Document logging** option is enabled on your Broker.

To determine whether Document logging is enabled on a Broker

1. In My webMethods: **Messaging > Broker Servers > Brokers**.
2. In the **Brokers List**, click the Broker whose logging option you want to check. If the Broker does not appear in the list, use the **Search** tab to locate it.
3. In the **Configuration** tab on the Broker Details page, check the state of the **Document logging** option.

Checking the Length of the Document Logging Queue

Use the following procedure to view the number of documents that are in the Document logging queue waiting to be retrieved by the Logging Utility.

To check the length of the Document logging queue

1. In My webMethods: **Messaging > Broker Servers > Brokers**.
2. In the **Brokers List**, click the Broker whose logging queue you want to examine. If the Broker does not appear in the list, use the **Search** tab to locate it.
3. On the Broker Details page, click the **Statistics** tab and check the **Document Log Length** value. This value represents the number of documents in the log queue awaiting retrieval by the Logging Utility.

Routing Documents to the Dead Letter Queue

The dead letter queue is a queue in which Broker places *dead letters*. A dead letter is a document for which the Broker has no subscribers. For example, if a client program publishes a document of type "Orders::ShippingNotice," and the Broker has no subscribers for that document type (or if it has filtered subscriptions for the document type, but the document instance does not match the conditions specified by the filters) it places the document in the dead letter queue. A Broker also sends to the dead letter queue delivered documents that are addressed to non-existent clients.

The Dead Letter Client

To activate the dead letter queue, you must configure the Broker's *dead letter client*. The dead-letter client is a system client that subscribes to dead letters. It is a member of the eventLog client group. This client group generates an explicit-destroy client with a guaranteed queue.

By default, the client ID for the Broker's dead-letter queue is "DefaultDLQ_", however, you can optionally append a suffix to this client ID when you activate the queue. For example, you might add the name of your Broker to the default client ID to form an ID such as "DefaultDLQ_BrokerEast01." You can also associate an application name with the dead letter client to make it easier to locate in the Broker user interface.

When you activate the dead letter queue, the Broker generates a (disconnected) client state object using the client ID that you specify. You can use My webMethods to examine or purge the dead letter queue. You can also use the Broker client API to develop a program that connects to the dead letter client's state object (using the client ID you assigned to the dead letter client) and retrieves the dead letter documents from the queue.

Relationship to Dead Letter Queues Created by Client Applications

The Broker's dead letter queue is the default queue for dead letters. It receives dead letters for all document types. Broker client applications can also create and maintain their own dead-letter queues. For example, an application that submits order documents might maintain a dead letter client to capture order documents for which there are no subscriptions.

The presence of an application-defined dead letter queue does not affect the behavior of the Broker's default dead letter queue. If the Broker receives a dead letter for which there exists an application-defined dead letter queue, it places the document in the application's dead-letter queue and in the default queue.

Activating the Dead Letter Queue

Use the following procedure to activate the dead letter queue on the Broker.

Note: Once you configure the client ID and application ID parameters for the dead letter queue, they cannot be changed. If you need to modify these parameters, you must delete the existing dead letter client and configure a new one.

To activate the dead letter queue

1. In My webMethods: **Messaging > Broker Servers > Brokers**.
2. In the **Brokers List**, click the Broker whose dead letter queue you want to activate. If the Broker does not appear in the list, use the **Search** tab to locate it.
3. On the Broker Details page, click the **Dead Letter Queue** tab and (optionally) complete the following fields. If these fields are non-editable, it indicates that the dead letter client has already been configured.

In this field...	Specify...
Client ID	<p>Optional. A suffix of one or more characters that, when appended to the "DefaultDLQ_" prefix, forms a client ID that is unique within the Broker.</p> <ul style="list-style-type: none"> ■ The ID cannot contain the following characters: @ / : ■ The entire ID cannot exceed 255 characters. <p>Examples:</p> <pre>DefaultDLQ_BrokerE01 DefaultDLQ_NoSubscribers</pre>
Application Name	<p>Optional. An application name to associate with this client ID. The application name appears when you display the list of clients in My webMethods. You can use this field to label the client in a way that distinguishes it from regular clients and makes it easy to locate in the client list.</p> <p>Examples:</p> <pre>Dead letter queue for BrokerE01 ADMIN-Dead letters</pre>

4. Click **Apply** to generate the dead letter client.

Viewing and Purging the Dead Letter Queue

To view the dead letter queue, use My webMethods to examine the queue associated with the dead letter client.

To view the dead letter queue

1. In My webMethods: **Messaging > Broker Servers > Clients**.

2. Use the following steps to locate the dead letter client:
 - a. Type the following search string in the **Keyword** field on the **Search** tab:
`defaultDLQ`
 - b. In **Server Name** and **BrokerName** fields, specify the Broker Server and Broker on which the dead letter queue resides.
 - c. Click **Go**.
3. Click the client ID associated with the dead letter queue for your Broker.

Note: If the search returns no client IDs, it means that a dead letter queue has not been configured for this Broker.

- To view the contents of the queue, click the **Browse Queue** tab. For information about using this tab, see [“Browsing a Client Queue” on page 235](#).
- To view statistics for the queue, click the **Statistics** tab.
- To purge the queue, click the **Statistics** tab and then click the **Clear Queue** button.

Disabling the Dead Letter Queue

To disable the dead letter queue, you must delete the dead letter client. Deleting the dead letter client immediately discards any documents that are contained in the dead letter queue and de-activates the dead letter feature.

To disable the dead letter queue

1. In My webMethods: **Messaging > Broker Servers > Clients**.
2. Use the following steps to locate the dead letter client:
 - a. Type the following search string in the **Keyword** field on the **Search** tab:
`defaultDLQ`
 - b. In **Server Name** and **BrokerName** fields, specify the Broker Server and Broker on which the dead letter queue resides.
 - c. Click **Go**.
3. In the **Client List**, select the check box beside the dead letter client ID.
4. Click **Delete**.

Viewing Status Information for a Broker

Use the following procedure to display basic status information for a Broker.

Note: You can also use the **Topology** and **Trace** tabs to display a graphic representation of Broker activity. For more information about these tab, see [“Topology View of Brokers” on page 260.](#)

To view status information

1. In My webMethods: **Messaging > Broker Servers > Brokers.**
2. The **Brokers List** displays the following status information about a Broker. If the Broker you want to examine does not appear in the list, use the **Search** tab to locate it.

Column	Description
Server Name	The Broker Server that hosts the Broker.
Connected	<p> Yes.My webMethods has successfully connected to the Broker.</p> <p> No.My webMethods could not establish a connection with the Broker. This might occur because your connection to this Broker has been closed or because Broker Server is not functioning normally.</p> <p> In Progress.My webMethods is in the process of establishing a connection with this Broker.</p> <p> Unknown.My webMethods experienced an unrecognized error when trying to establish a connection with this Broker.</p>
Clients	<p>The number of clients associated with this Broker. This number represents the total count of <i>client state objects</i> that are currently defined on the Broker.</p> <ul style="list-style-type: none"> ■ This number includes explicit-destroy clients that are in the disconnected state as well as clients that are physically connected to Broker. ■ A shared-state client counts as one connection regardless of how many clients are physically connected to it.
Territory	The name of the territory, if any, to which the Broker belongs.
Recent Deliveries or Total Deliveries	<ul style="list-style-type: none"> ■ Recent Deliveries displays the number of documents that clients have published or delivered to this Broker during the last statistics collection interval. By default, the statistics collection interval represents one minute, however, you can configure this interval in My webMethods. For procedures,

Column	Description
	<p>see the Time interval between statistical refresh parameter in “Configuring the Connection Parameters” on page 53.</p> <p>Note: Recent Deliveries data is available only if you have selected Enable Statistical Polling.</p> <ul style="list-style-type: none"> ■ Total Deliveries displays the total number of documents that clients have published or delivered to this Broker. <p>Note: Total Deliveries data is available only if you do not select Enable Statistical Polling.</p>
Default	Indicates whether the Broker functions as the default Broker for the Broker Server. A Broker Server can have zero or one default Brokers. For more information about default Brokers, see “Configuring a Default Broker ” on page 142.

- To display additional details about the Broker, click the name of the Broker.

Viewing Basic Operating Statistics for the Broker

Use the following procedure to examine statistics relating to the number of documents that the Broker has received from clients.

To view basic Broker statistics

- In My webMethods: **Messaging > Broker Servers > Brokers**.
- In the **Brokers List**, click the Broker whose statistics you want to view. If the Broker does not appear in the list, use the **Search** tab to locate it.
- Select the **Statistics** tab to display the following information:

Statistic	Description
RecentBrokerDeliveries	<p>Displays the number of documents this Broker has received during the last statistics collection interval.</p> <p>By default, the statistics collection interval represents one minute, however, you can configure the interval in My webMethods. For procedures, see the Time interval between statistical refresh parameter in “Configuring the Connection Parameters” on page 53.</p> <p>Note: RecentBrokerDeliveries data is available only if you have selected Enable Statistical Polling.</p>

Statistic	Description
Total Broker Deliveries	Displays the number of documents that this Broker has received since the Broker Server was last created.
Document Log Length	<p>Displays the number of documents in the Document logging queue waiting to be retrieved by the Logging Utility.</p> <p>For more information about the Document logging queue, see “Using Document Logging” on page 143.</p>
Broker Retry Queue	<p>Displays statistics for an internal queue that Broker uses to store requests that it must retry. You might be asked to provide these statistics to Software AG Global Support for troubleshooting purposes.</p> <ul style="list-style-type: none"> ■ Current Publishes displays the number of current publishes in the retry queue. ■ Current Documents displays the number of current documents in the retry queue. ■ Maximum Published displays the current maximum size for retry queue publishing. This number is not a constant, it increases according to the available storage space. ■ Maximum Documents displays the current maximum size for retry queue events. This number is not a constant, it increases according to the available storage space. ■ Reserved Total Publishes displays the total number of reserved publishes. ■ Reserved Guaranteed Publishes displays the number of guaranteed publishes. ■ Reserved Volatile Publishes displays the number of volatile publishes. ■ Reserved Guaranteed Documents displays the number of guaranteed documents. ■ Reserved Volatile Documents displays the number of volatile documents. ■ Next Operation Sequence Number displays the number of the next operation sequence.
Broker Trace Queue	Displays statistics for an internal queue that Broker uses to store activity events and trace events. You might be

Statistic	Description
	asked to provide these statistics to Software AG Global Support for troubleshooting purposes.
	<ul style="list-style-type: none"> ■ Highest Length displays the highest number of events in the trace queue. ■ Highest Length Time displays the time when Highest Length was last set. ■ Length displays the number of events in the trace queue. ■ Size displays the number of bytes of events in the trace queue. ■ Documents Queued displays the number of documents queued. ■ Last Document Enqueued displays the time when the last event was enqueued.

Managing Transactions

Transaction processing allows a Broker client to group the events it publishes as a single unit of work called a *transaction*. A transaction either completes successfully, is rolled back to some known earlier state, or it fails. Once all of the documents that make up a transaction have been published, delivered, or received, the Broker client ends the transaction. A transaction can be ended by committing the transaction or aborting the transaction.

The Broker's transaction manager coordinates and controls the transactions that are initiated by Broker clients. Transactions that run under the Broker transaction manager include transactions initiated by regular Broker transactional clients, as well as regular (local) and XA transactions that are initiated by the webMethods Broker Client API for JMS.

You use the transaction controls in My webMethods to monitor and manage transactions that are running under the transaction manager on the Broker. These controls allow you to monitor the activity of transactions as they execute and take action against transactions that do not appear to be running correctly.

Viewing Running Transactions

If a client has started a transaction on the Broker, that transaction appears on the **Transactions** tab. The transaction disappears from this tab when the client explicitly ends the transaction.

In many cases, transactions complete too quickly to be viewed in the **Transactions** tab. However, the list is useful for monitoring the state of long-running transactions and for spotting transactions that have become hung in the system.

To view transactions running on Broker

1. In My webMethods: **Messaging > Broker Servers > Brokers**.
2. In the **Brokers List**, click the Broker whose transactions you want to view. If the Broker does not appear in the list, use the **Search** tab to locate it.
3. On the Broker Details page, click the **Transactions** tab. The transaction list at the bottom of this tab displays transactions that are currently running on the Broker.

Column	Description
Transaction ID	The unique identification number (txid) that the Broker assigns to a transaction when it starts.
External ID	An identifier that an application can optionally assign to a transaction. Depending on the application, this value might be empty.
State	<p>The current state of the transaction, which will be one of the following:</p> <ul style="list-style-type: none"> ■ Open. The transaction is in progress and has not yet been prepared or committed. ■ End Success. The transaction has ended with the success flag and not yet been prepared. ■ End Failure. The transaction has ended with the failure flag and not yet been prepared. ■ Heuristic Commit. The transaction has been heuristically committed. ■ Heuristic Rollback. The transaction has been heuristically rolled back. ■ Prepared. The transaction has been prepared, but has not yet been committed or rolled back. ■ Committed. The transaction is in the process of being committed. ■ Rollback. The transaction is being rolled back.
Created	The time at which the transaction was started.

Column	Description
Published	The number of documents that the transactional client has published or delivered within the context of this transaction since the transaction started. (If the transactional client published other documents during this time under another transactional context, those documents would be reflected in a separate transaction entry.)
Acknowledged	The number of documents the transactional client has received and acknowledged within the context of this specific transaction since the transaction was started. (If the transactional client received other documents during this time under another transactional context, those documents would be reflected in a separate transaction entry.)

About Configuring the Transaction Timeout Options

For transaction processing, you can configure the Broker to monitor the length of time between stages of a transaction and to take a prescribed action if a transaction exceeds a specified period of time.

When a transaction exceeds the specified time limit (that is, when it expires), the Broker automatically performs a commit or roll back for the transaction. Whether the Broker performs a commit or roll back depends on how you have configured the transaction timeout options.

Transactions that expire and are completed by the Broker are considered to be *heuristically completed*, meaning that the decision to perform a commit or roll back did not come from the client. As required by the XA Specification, the Broker maintains a record of heuristically completed transactions in a log.

The Broker allows you to specify separate timeout limits for two stages of a transaction:

- **The pre-prepare stage.** The pre-prepare stage refers to the time interval between the point at which a transaction begins and the point at which it performs a prepare operation (for two-phase commit) or a commit or roll back operation (for single-phase transaction). If a transactional client does not issue a prepare, a commit, or a roll back request within the specified timeout period, the Broker automatically performs a roll-back operation and then terminates the transaction.

The pre-prepare timeout parameter applies to both two-phase (XA) transactional clients and single-phase (local) transactional clients. For two-phase transactions, the pre-prepare timeout represents the time limit imposed for the period between the start of the transaction and the receipt of the prepare request. For single-phase transactions, the timeout setting represents the time limit for the period between the start of the transaction and the receipt of a commit or roll back request.

You can specify an infinite pre-prepare period (that is, impose no timeout limit) by setting the pre-prepare timeout parameter to -1.

- **The post-prepare stage.** In the case of a transaction that uses a two-phase commit, the post-prepare stage refers to the period from the receipt of the prepare request until the initiation of the commit or roll back request. In the case of a single-phase commit, this timeout does not apply. If a transaction exceeds the specified timeout period, the Broker executes a commit or roll-back operation on behalf of the client and terminates the transaction. Whether the Broker executes a commit or a roll back depends on the way in which you configure the Broker's post-prepare timeout action parameter. The default is to commit the transaction.

The post-prepare timeout parameter does not apply to single-phase transactions.

You may specify an infinite post-prepare period (that is, impose no timeout limit) by setting the post-prepare timeout parameter to -1.

Configuring the Transaction Timeout Options

Use the following procedure to configure the transaction timeout options.

To configure the timeout settings

1. In My webMethods: **Messaging > Broker Servers > Brokers**.
2. In the **Brokers List**, click the Broker that you want to configure. If the Broker does not appear in the list, use the **Search** tab to locate it.
3. On the Broker Details page, click the **Transactions** tab and configure the following fields.

In this field...	Specify...
Pre-Prepare Timeout	The length of the pre-prepare timeout interval, in seconds, or -1 to set an infinite timeout interval.
Post-Prepare Timeout	The length of the post-prepare timeout interval, in seconds, or -1 to set an infinite timeout interval.
Post-Prepare Timeout Action	The action you want Broker to take if a transactional client exceeds the post-prepare timeout interval.

4. Click **OK**.

The new timeout settings will apply to transactions that begin after the pre-prepare timeout parameter has been changed or are prepared after the post-prepare timeout parameter has been changed.

Setting the Recover Mode for XA Transactions

The recovery mode parameter determines whether the Broker transaction manager allows clients to recover any recoverable transaction or only those transactions in which they participated.

Note: XA transactions can only be initiated by JMS clients.

To specify the recovery mode for XA transactions

1. In My webMethods: **Messaging > Broker Servers > Brokers**.
2. In the **Brokers List**, click the Broker that you want to configure. If the Broker does not appear in the list, use the **Search** tab to locate it.
3. On the Broker Details page, click the **Transactions** tab.
4. In the **Recover Mode** field, select one of the following:

Option	Description
Restricted	Clients can recover only transactions in which they participated.
Global	Clients can recover any transaction.

5. Click **OK**.

Manually Performing a Commit or Roll Back

You can force a transaction to terminate using the **Commit Transactions** and **Roll Back** buttons on the **Transactions** tab (these commands will not appear on the **Transaction** tab unless transactions are running).

You might force a commit or roll back to terminate a transaction instead of waiting for it to expire. If you have configured the Broker to execute transactions without an imposed time limit, you will need to use the commit and roll back commands to terminate hung transactions.

A transaction that you manually commit or roll back is considered a heuristically completed transaction (because the decision to commit or roll back was not made by the client) and is recorded in the lost transaction log.

To manually commit or roll back a transaction

1. In My webMethods, go to **Messaging > Broker Servers > Brokers**.
2. In the **Brokers List**, click the Broker on which the transaction is running. If the Broker does not appear in the list, use the **Search** tab to locate it.

3. On the Broker Details page, click the **Transactions** tab.
4. Select the check box beside the transaction that you want to rollback or commit.
5. Click **Roll Back** or **Commit**.

Viewing and Purging Lost Transactions

When a transaction ends heuristically, that is, when either the Broker or an administrator makes the decision to perform a commit or roll back for a transaction, that transaction is written to the Broker's lost transaction log.

On the Broker, a heuristically completed transaction is referred to as a *lost transaction*. Lost transactions are written to the Broker's lost transaction log, where they remain until they are explicitly purged.

The Broker Server periodically purges old lost transactions. You can configure the age, time when the transaction was heuristically completed, and the time interval between two automatic purge operations using the `forget-lost-transaction-timer-interval-seconds` and `forget-lost-transaction-time-days` server configuration parameters. For more information about the configuration parameters, see “[webMethods Broker Server Configuration Parameters](#)” on page 621.

Use the following procedure to purge a transaction from the lost transaction log. When you purge a transaction, the Broker releases that transaction's txid and discards all knowledge it had of the transaction.

To view and purge lost transactions

1. In My webMethods, go to **Messaging > Broker Servers > Brokers**.
2. In the **Brokers List**, click the Broker on which the transactions you want to view or purge are running. If the Broker does not appear in the list, use the **Search** tab to locate it.
3. On the Broker Details page, click the **Transactions** tab.
4. Click the **Lost Transaction** link to display the lost transaction log.
5. Select the check box beside the transactions that you want to purge and click **Delete**.

6 Managing Document Types

■ Overview	158
■ What Are Documents and Document Types?	158
■ Viewing Document Types on a Broker	163
■ Viewing Document Type Configuration Information	164
■ Viewing Document Type Fields and Infosets	166
■ Viewing Which Clients Subscribed to a Document Type	167
■ Modifying Document Type Properties	167
■ Copying Document Types Between Brokers	169
■ Deleting Document Types	170

Overview

This chapter describes how to manage and view information about document types on a Broker.

What Are Documents and Document Types?

Documents are messages that travel over a network from a publisher to a subscriber, through the Broker. The role of the Broker is to route documents between information producers (publishers) and information consumers (subscribers). The Broker receives, queues, and delivers documents.

Each document is an instance of a document type. A document type is a schema-like definition that describes the structure of a document that publishers and subscribers can exchange using the Broker. A document type has a unique name, a structure that consists of named fields, and a set of properties that determines how the Broker handles instances of that document type at run time. Clients indicate which documents they want to receive by subscribing to specific document types.

Creating Document Types

Document types can be created on Broker in one of the following ways.

- A developer uses Software AG Designer to define a document type and mark it as publishable. For each publishable document type created using Designer, Integration Server automatically registers a corresponding document type on the Broker. For more information about creating publishable document types, see the *Software AG Designer Online Help*.
- A developer can create a document type using the Broker client APIs. For more information, refer to one of the client API programming guides.
- A developer can add a JMS topic as a Broker document type through My webMethods. For more information about adding topics, see [“About Creating Destinations” on page 472](#).
- Other webMethods components add document types to support their underlying system processes.

A document type must exist on the Broker before clients can publish instances of that type. The name of the document type, which must be unique within a Broker (and among all Brokers with which that Broker interacts), is carried by all documents of its type.

Note: In previous versions of Broker, document types were called event types and instances of document types were called events. You may still see these terms in documentation that relates to the underlying Broker API.

About Document Type Names

A document type name consists of two components: a folder path and a base document type name.

folderName::subFolderName::documentTypeName

The folder path component can consist of one or more levels (subfolders). For example, for this fully qualified document type name:

`WesternRegion::Hardware::Sales::receiveCustOrder`

The base document type name would be `receiveCustOrder` and the full folder path would be `WesternRegion::Hardware::Sales`.

If the document type is created through Designer, the Broker document type name includes the preface "wm::is::" before the *folderName::documentTypeName* convention. For example, a document type named `employee:employeeInfo` on Designer would be named `wm::is::employee::employeeInfo` on the Broker. Other webMethods components that create document types will use similar naming notations. For more information about publishable document types in Designer, see the *Software AG Designer Online Help*.

Document Type Properties

Each document type has properties associated with it that determine how the Broker handles instances of that document type at run time. These properties include how long instances of the publishable document type should remain valid once they are published, document type validation, and the document type's storage type. You use My webMethods to modify these properties:

- **Time to live** specifies how long Broker maintains instances of this document type once they are published.
- **Validation** indicates whether Broker validates instances of this document type and if so, if the document can contain fields that are not defined in the document type.
- **Storage type** determines whether instances of this document type are stored in memory or on disk.

The following sections provide more information about these properties.

Time to Live

For volatile documents and guaranteed documents, the **Time to live** property of a document type determines how long instances of that document type remain valid after being published. The time to live commences when the Broker receives a document from a publishing client. If the time to live elapses before the client retrieves the document, the Broker discards the document the next time the client requests documents from its queue.

By default, Broker does not actively check if a document has exceeded its time to live. A document's time to live is checked only when a client retrieves the document. An expired document is discarded when the Broker attempts to deliver it to the client. This behavior can be changed for volatile documents only by allowing Broker to proactively delete volatile expired documents based on their combined size. For more information, see [“Proactively Deleting Documents from a Client Queue” on page 243](#).

The time to live can be useful for documents containing data that is updated regularly, such as stock quotes. If a document is generated for a quote every 20 seconds, the time to live can be set to 20. If a client does not retrieve the quote after 20 seconds, then it is discarded because a newer value will arrive soon.

When you set the time to live, you specify the number of seconds before the document expires. If you do not want the document to expire, you can specify a value of 0 (zero). For more information about setting the [Time to Live](#) value for a document type, see [“Modifying Document Type Properties” on page 167](#).

Note that changing the time to live for a document type using My webMethods can cause the document type on the Broker to become unsynchronized with the document type on the client application.

For example, suppose you use Designer to create a document type and then make the document type publishable. This creates a corresponding document type on the Broker. Suppose that when you created the document in Designer, you set the time to live to 30 seconds. If you use My webMethods to change the time to live for the document type to 15 seconds, the document type on the Broker is now unsynchronized with the document type on Integration Server (documents created via Designer are stored on Integration Server).

When it receives a document, Broker uses the time to live specified in the document type on the Broker. If a developer built an application that relied on the time to live value specified in the Integration Server document type, changing the document type time to live using My webMethods might unexpectedly (or negatively) impact the application.

Document Type Validation

When a Broker receives a document, it checks or validates the document contents against the associated document type. The validation level assigned to the document type determines whether Broker considers a document invalid if it contains fields that are not declared in the document type. The Broker will accept or discard the document based on the outcome of the validation.

There are three levels of document type validation:

- **Full.** The document can contain fields declared in the document type only. If the document contains fields that are not declared in the document type, Broker considers the document to be invalid.
- **Open.** The document can contain fields that are not declared in the document type. Fields that are defined in the document type must match the restrictions defined

in the document type. Fields not defined in the document type can exist in the published document, but these undefined fields will not be type validated.

- **None.** Broker does not validate instances of this document type. This is the default.

For information about changing the level of document type validation, see [“Modifying Document Type Properties” on page 167](#).

Note: Integration Server can also validate documents at publication time. To improve Broker (or Integration Server) performance, you may want to perform validation on either Integration Server or Broker, but not both.

Document Type Storage Types

The document storage type setting determines how instances of that document are persisted. Documents published to Broker can be stored as *volatile* documents or *guaranteed* documents.

- **Volatile documents** are stored in local memory only. These documents are lost if the Broker host experiences a service interruption or the Broker Server is restarted.

To reduce memory usage, volatile documents that have expired can be proactively deleted at regular intervals, based on the size of the queue, from the client queues and forward queues before the client tries to retrieve them. For more information, see [“Proactively Deleting Documents from a Client Queue” on page 243](#).

Volatile storage provides higher performance than guaranteed storage; however, there is a greater risk of losing documents if a hardware, software, or network failure occurs. All documents of a volatile document type and documents in a volatile client queue are lost when the Broker is shut down or when the computer restarts. This storage type is suited for documents that have a short life span or are not critical.

- **Guaranteed documents** are persisted to disk so that they can be recovered in the event of a power failure or a server restart. This is the default storage type for document types.

Guaranteed storage provides lower performance than volatile storage, but very little risk of losing events if a hardware, software, or network failure occurs. This type of storage is the safest, and is suited for documents that you do not want to lose.

Guaranteed storage has a fixed, preallocated size that can only be changed while the Broker is stopped. This size is a function of the document flow and of the size of the documents. The default guaranteed storage size is 32MB per document and 512MB total for all guaranteed, queued documents. You can increase the storage size by adding new storage files (see [“Adding a Storage File” on page 91](#)).

Client Group Storage Types and Document Type Storage Types

The ultimate determination of how a document is saved (in local memory only or to disk) is made by a combination of storage type settings:

- That of the *client group* to which the subscriber belongs

- That of the document's *document type*

The following table indicates whether Broker designates a document as volatile or guaranteed, depending on the above factors.

Client group storage type	Document type storage type	The Broker saves the document as...
Volatile	Volatile	Volatile
Volatile	Guaranteed	Volatile
Guaranteed	Volatile	Volatile
Guaranteed	Guaranteed	Guaranteed

For information about client group storage types, see [“Client Group Storage Types and Document Type Storage Types”](#). For information about how to assign storage properties to a document type, refer to the *Software AG Designer Online Help*.

Document Field Information

Document fields contain the data that your client applications use to exchange information. Document fields may contain a single value, a sequence of values with the same type, or a structure containing values of different types.

You can view the fields of a document type to learn more about what the document type does. Fields contain application-specific data that your application will set before publishing a document or will retrieve when processing a document. Each field has a name and a typed value; that is, each field is assigned a data type (e.g., int, string, byte) and values in that field must conform to that data type. You cannot edit field information in My webMethods.

Infosets Information

Infosets contain client application-specific information stored in the format of a Broker document. The infoset information is based on the application that created the client (for example, both JMS and Test Client store data in the infoset field), and the application uses the infoset to store specific configuration information for its own use. For more information about setting infoset information, refer to the appropriate Broker client API guide.

Viewing Document Types on a Broker

Using My webMethods, you can view a list of the document types on a Broker. You can display all the document types or you can filter the list of document types to see only those that meet specific criteria. For example, you might want to view only those document types that contain the word "Adapter" in their name.

To view documents types on a Broker

1. In My webMethods: **Messaging > Broker Servers > Document Types**.
2. On the Document Types page, in the text box on the **Search** tab, type the keywords contained in the document types that you want to find. For example, you might specify
 - The complete name of the document type (for example, Customer1::Order::Received)
 - A partial document type name (for example, Received or wm::is) to select all document types that contain the provided text.

If you leave the text box blank, My webMethods searches for all the document types on the specified Broker. For more information about using the search facility, see *Working with My webMethods*.

3. Select the Broker Server and Broker whose document types you want to view by doing the following:

In this field...	Specify...
Server Name	Fully-qualified name of the Broker Server: <i>hostname:port</i> If the Broker Server has the default port of 6849, only the Broker Server <i>hostname</i> will appear.
BrokerName	Name of the Broker as defined on the Broker Server.

4. Click **Go** to execute the search. My webMethods displays the search results in the **Document Type List** tab.

The following table describes the contents of the **Document Type List**.

Field	Description
Document Type	Fully qualified name of the document type in the following format: <i>folderName::documentTypeName</i>

Field	Description
Broker@Server	Name of the Broker Server on which the document type resides: <i>Broker ServerName@hostname:port</i>
Storage Type	Storage type of the document type as follows: <ul style="list-style-type: none"> ■ Guaranteed. Documents of this type are stored on disk. This is the default value. ■ Volatile. Documents of this type are stored in memory. For more information, see “Document Type Storage Types” on page 161.
Description	A brief description of the document type.
Created	Date and time the document type was created.

- You can view information for a specific document type by clicking on its name in this list of document types. For more information, see [“Viewing Document Type Configuration Information” on page 164.](#)

Note: To view an updated list of document types, click **Go** on the **Search** tab.

Viewing Document Type Configuration Information

For each document type that exists, Broker maintains configuration information specified at the time the document type was created. Broker also maintains statistics for the document type, such as the number of subscriptions to the document type and the total number of published instances of the document type.

You can use My webMethods to modify the description, time to live, and validation level for a document type. For more information about modifying these properties, see [“Modifying Document Type Properties” on page 167.](#)

To view document type configuration information

- In My webMethods: **Messaging > Broker Servers > Document Types.**
- In the **Document Type List**, click the document type for which you want to view configuration information. If the document type does not appear in the list, use the **Search** tab to locate it.
- On the Document Type Details page, click the **Configuration** tab to view additional information and statistics about a specific document type.

The following table describes the contents of the **Configuration** tab.

Field	Description
Description	A brief description of the document type. You can edit this field to modify the document type description. For more information about modifying document properties, see “Modifying Document Type Properties” on page 167 .
Time to Live (seconds)	<p>Determines how long instances of a document type remain valid after being published. A time to live value of "0" (zero) indicates that documents of that type will never expire.</p> <p>For more information about time to live, see “Time to Live” on page 159". For more information about modifying document properties, see “Modifying Document Type Properties” on page 167.</p>
Validation	<p>The document type validation level that Broker performs when it receives instances of this document type. The validation level can be one of the following:</p> <ul style="list-style-type: none"> ■ Full. Broker considers the document valid if all the fields in the document are declared in the document type and all the fields match the document type definition. <p>Broker considers the document invalid if it contains fields that are not declared in the document type.</p> <p>This is the strictest form of validation.</p> ■ Open. Broker considers the document valid if the fields declared in the document type match the document type definition. Broker allows the document to contain fields that are not declared in the document type. Broker does not validate these fields. ■ None. Broker does not perform validation for instances of this document type. <p>For information about validation, see “Document Type Validation” on page 160. For more information about modifying document properties, see “Modifying Document Type Properties” on page 167.</p>
Storage Type	<p>Indicates how the Broker stores instances of this document as follows:</p> <ul style="list-style-type: none"> ■ Guaranteed. Documents of this type are stored on disk. This is the default value. ■ Volatile. Documents of this type are stored in memory.

Field	Description
	For more information about how a document is saved (in local memory only or to disk), see “Client Group Storage Types and Document Type Storage Types” on page 161.
Created	The date and time the document type was created.
Last Modified	The date and time the document type was last modified.
System Defined	Indicates whether the document type is defined by the Broker as follows: <ul style="list-style-type: none"> ■ Yes. The document type is predefined in the system. You cannot edit or delete system defined document types. ■ No. The document type is not defined by the system.
Total Documents Published	Total number of instances of this document type that have been published to the Broker since the document type was created.
Last Published	Date and time an instance of this document type was last published.
Subscriptions	Number of subscriptions to this document type by clients on this Broker.
Forwards Received	The number of documents of this type received from a remote Broker in the same territory.
Last Forward	The date and time a document of this type was last forwarded from a remote Broker in the same territory.

- Click **Close** to return to the Document Types page.

Viewing Document Type Fields and Infosets

You can use My webMethods to view the fields and infosets for a specific document type. Fields contain the data that your client applications use to exchange information. Infosets contain client application-specific information stored in document format. For more information about document type data fields, see [“Document Field Information”](#) on page 162. For more information about infosets, see [“Infosets Information”](#) on page 162.

To view document type fields and infosets

1. In My webMethods: **Messaging > Broker Servers > Document Types**.
2. In the **Document Type List**, click the document type for which you want to view fields and infosets. If the document type does not appear in the list, use the **Search** tab to locate it.
3. On the Document Type Details page, click the **Fields** tab to view the fields defined in the document type.
4. On the Document Type Details page, click the **Infosets** tab to view the infoset information maintained for the document type.
5. Click **Close** to return to the Document Types page.

Viewing Which Clients Subscribed to a Document Type

You can use My webMethods to view which clients have subscribed to a specific document type.

To view which clients have subscribed to a document type

1. In My webMethods: **Messaging > Broker Servers > Document Types**.
2. In the **Document Type List**, click the document type for which you want to view the clients that have subscribed to it. If the document type does not appear in the list, use the **Search** tab to locate it.
3. On the Document Type Details page, click the **Clients** tab to view all the clients that have subscribed to the document type.

Modifying Document Type Properties

Using My webMethods, you can modify the following document type properties:

- Document type description
- Time to live value
- Validation level

Changing the time to live for a document type using My webMethods can cause the document type on the Broker to become unsynchronized with the document type on the client application. For complete information, see [“Time to Live” on page 159](#).

Note: You cannot edit system-defined document types.

Note: For information about adding and modifying JMS topics as document types, see [“JMS Messages” on page 437](#).

To modify document type properties

1. In My webMethods: **Messaging > Broker Servers > Document Types**.
2. In the **Document Type List**, click the document type that you want to modify. If the document type does not appear in the list, use the **Search** tab to locate it.
3. On the Document Type Details page, make sure the **Configuration** tab is selected.
4. Do one or more of the following to change properties for the selected document type.

In this field...	Enter...
Description	A brief description of the document type.
Time to Live	<p>The number of seconds that instances of this document type remain valid on the Broker. After the time to live elapses, the document expires and Broker discards it. If you do not want instances of this document type to expire, enter 0.</p> <p>For complete information about the time to live property, see “Time to Live” on page 159.</p> <div style="background-color: #f0f0f0; padding: 5px;"> <p>Note: Increasing or decreasing the time to live does not have an immediate effect on documents already in the client queues.</p> </div>
Validation	<p>The type of validation that you want the Broker to perform for instances of this document type. Select one of the following:</p> <ul style="list-style-type: none"> ■ Full. The document can contain fields declared in the document type only. If the document contains fields that are not declared in the document type, Broker considers the document to be invalid. This is the default. ■ Open. The document can contain fields that are not declared in the document type. Fields that are defined in the document type must match the restrictions defined in the document type. Fields not defined in the document type can exist in the published document, but these undefined fields will not be type validated. ■ None. Broker should not validate instances of this document type. <p>For more information, see “Document Type Validation” on page 160.</p>

5. Click **Apply**.

Copying Document Types Between Brokers

You can copy document types between any Brokers to which you have access in one of two ways:

- Copy and paste.
- Export and Import. When you use the export-import functionality, My webMethods places the exported document type information in a file. The administrator for the destination Broker then imports the file. To use the export-import functionality, you must have administrator access. For more information about using the export-import functionality, see [“Exporting and Importing Broker Metadata” on page 483](#).

Note: If a document type with the same name already exists on the Broker, My webMethods appends "_1" to the document type name. You cannot copy and paste a document type that already exists on the target Broker. However, you can import a document type that already exists on the target Broker. The Broker replaces the existing document type with the imported one.

Copying and Pasting Document Types

When you copy and paste a document type, you make a duplicate copy of an existing document type on another Broker. That is, copying and pasting documents between Brokers does not create a new document type.

To copy and paste document types between Brokers

1. In My webMethods: **Messaging > Broker Servers > Document Types**.
2. In the **Document Type List**, select the check box next to the name of the document types that you want to copy. If the document type does not appear in the list, use the **Search** tab to locate it.
3. Click **Copy**.
My webMethods copies the selected document types.
4. Click **Paste**. The Document Type List dialog box appears.
5. Under **Select Target to Paste**, select the **Broker Server** and **Broker** on which to paste the copied document types.
6. Click **Paste**. My webMethods places a copy of the document types on the selected Broker.

Deleting Document Types

Using My webMethods, you can delete document types on the Broker. You can only delete document types if:

- The document type has no subscribers.
- A client group does not have can-publish or can-subscribe permissions to the document type.
- The document type is user-defined. You cannot delete system-defined document types.

If you attempt to delete a document type for which there are subscribers, a document type to which a client group has can-publish or can-subscribe permissions, or a system-defined document type, My webMethods displays an error and prevents the deletion of the document type.

Note: If a client publishes a document for which there is not an associated document type on the Broker, Broker returns an error message to the client and publication fails.

For information about deleting Broker document types and their associated publishable Integration Server document types, see the *Publish-Subscribe Developer's Guide*.

To delete a Broker document type

1. In My webMethods: **Messaging > Broker Servers > Document Types**.
2. In the **Document Type List**, select the check box next to the name of the document types that you want to delete. If the document type does not appear in the list, use the **Search** tab to locate it.
3. Click **Delete**. My webMethods deletes the selected document type.

7 Managing Client Groups

■ Overview	172
■ Defining a Client Group	172
■ Client Groups and Broker Security	172
■ Client Group Properties	173
■ System-Defined Client Groups	176
■ Working with Client Groups	178

Overview

This chapter explains how to use webMethods Broker client groups to assign default properties to Broker clients, and describes the various settings for these properties. The chapter also describes the security features associated with client groups, such as Access Control Lists (ACLs) and SSL encryption, and describes the key procedures for configuring and managing client groups, such as how to create a client group, configure its publish and subscribe permissions, edit its properties, and copy client groups from one Broker to another.

Defining a Client Group

Every Broker client belongs to one *client group*, which you can use to define a particular class of client (such as administrators, customers, Integration Server processes). The settings for a client group belong to the following categories:

- **Client characteristics**, such as queue storage type and whether client information is saved after a disconnection. These are client group settings that are passed to all clients in the group.
- **Security-related** properties, which include ACLs, SSL encryption, and use of an access label. These settings determine which clients can join the client group, as well as other methods of protecting messages and message traffic.
- **Access to document types**, specified by lists of the document types available to the client group's members.

Client Groups and Broker Security

Client groups play a critical role in Broker security. After you configure Broker SSL authentication (see [“Managing Broker Security” on page 287](#)), you can define Access Control Lists (ACLs) for a client group. ACLs are lists of SSL distinguished names (DNs) or basic authentication aliases that restrict access to a Broker object (see [“Access Control Lists” on page 325](#)). When configured for a client group, ACLs specify the clients that have permission to join.

You can use client group ACLs to restrict which clients have permission to access a Broker document type. For more information, see [“Client Group ACLs” on page 326](#).

You can also use client groups to specify whether a client needs an access label to join. Access labels secure individual messages based on access privilege level, and are an advanced security feature of Broker. For more information, see [“Using Access Labels” on page 585](#).

Client Group Properties

The following table lists the client group properties and includes a brief description of how each works.

Property	Description
Client group description	Optional. A one-line description of the client group.
Client queue storage type and lifecycle	<p>Determine the default storage type for client queues and how Broker deals with a client's state upon disconnection (the properties are set as a pair).</p> <p>The possible settings are:</p> <ul style="list-style-type: none"> ■ Volatile (destroy on disconnect) ■ Guaranteed (explicit destroy) (the default setting) <p>For more information, see “Client Queue Storage Type” on page 174 and “Client Lifecycle” on page 175.</p>
Access Control List (ACL)	<p>Optional. Specify the basic authentication identity or the SSL identity of clients permitted to join the client group. There are two ACLs, and you can configure one or both:</p> <ul style="list-style-type: none"> ■ User ACL <ul style="list-style-type: none"> ■ For basic authentication, a list of basic authentication user names. ■ For SSL, a list of clients specified by their user distinguished names, or DNs. ■ Authenticator ACL <ul style="list-style-type: none"> ■ For basic authentication, a list of basic authentication directory aliases (for example, LDAP and ADSI). ■ For SSL, a list of certificate issuer DNs (authorization names). <p>For more information, see “Managing Broker Security” on page 287, “Client Group Access Control Lists (ACLs)” on page 176, and “Access Control Lists” on page 325.</p>
Encryption	Whether to use SSL encryption with a 128-bit key length. By default, encryption is enabled.

Property	Description
	To enable encryption for a client group, its Broker Server must have an SSL identity, and the current user must have permission to configure encryption. For more information, see “Using SSL Encryption” on page 316 .
Access label required	Whether clients must have an access label to join the client group. By default, no label is required. Access labels are an advanced security feature of Broker. For more information, see “Using Access Labels” on page 585 .
Can-publish types	List the document types that the client group's Broker clients are permitted to publish (or deliver). For more information, see “Adding Document Type Permissions for Client Groups” on page 184 .
Can-subscribe types	List all document types that Broker clients in the client group can subscribe to (or receive deliveries from). For more information, see “Adding Document Type Permissions for Client Groups” on page 184 .
Log publish types	List all the document types that Broker clients in the client group can log when published. For more information, see “Adding Document Type Permissions for Client Groups” on page 184 .
Log acknowledge types	List all the document types that Broker clients in the client group can log when received. For more information, see “Adding Document Type Permissions for Client Groups” on page 184 .

Client Queue Storage Type

Administrators need to select the storage mechanism that clients will use to queue documents. This selection is made at the client group level, and is dictated by factors such as:

- Trade-offs between safety and performance: the safer the storage mechanism, the slower the performance
- The importance of maintaining client data across sessions

You assign a queue storage type when you create a client group, and the value cannot be changed. You can set the storage type to volatile or guaranteed.

- **Volatile.** Documents are stored only in local memory and are lost if the Broker Host loses power or the Broker Server is restarted.
- **Guaranteed.** Documents are persisted to disk and can be recovered in case of a power failure or restart.

Queue Storage Type and Lifecycle

Setting the value of the queue storage type also sets the client lifecycle value (see [“Client Lifecycle” on page 175](#)). Selecting a storage type of **volatile** always sets the lifecycle value to **destroy on disconnect**; selecting a storage type of **guaranteed** always sets the lifecycle value to **explicit destroy**.

Queue Storage Type and Document Type

The final determination of how a document is stored on Broker (volatile or guaranteed storage) is determined by the client group storage type and document storage type. For more information, see [“Document Type Storage Types” on page 161](#). For information about assigning storage properties to a document type, see the *Software AG Designer Online Help*.

Client Lifecycle

The *lifecycle* property determines whether the Broker maintains state information for a client after a disconnection. The lifecycle settings are:

- **Explicit destroy.** Maintains the Broker client's state object when the client disconnects. The client continues to receive documents in its queue, even though it is not actively connected. Use this setting for applications that need to maintain state information in the Broker between connections; for example, if a network or system failure occurs.

For example, a webMethods Integration Server uses the explicit destroy lifecycle so that documents that update the database are not lost if the Integration Server is not running. When the Integration Server is not running, the Broker queues documents for it; the Integration Server retrieves the documents when it restarts.

- **Destroy on disconnect.** Broker automatically destroys client state object information whenever the connection between the client and Broker is terminated, thus losing all knowledge of the client. With this setting, the state of a Broker client exists for the duration of the client's connection to the Broker.

Use this setting for applications that do not need to maintain state information in the Broker between connections.

Note: If an explicit destroy client disconnects for a long period of time, documents can begin to accumulate in memory and potentially cause the Broker to run out of resources. Make sure that explicit destroy clients

retrieve their documents frequently, especially if their subscription volume is high.

Client Group Access Control Lists (ACLs)

You use client group ACLs to prevent unauthorized clients from joining a client group. Before joining an ACL-protected client group, a client must supply its SSL identity (SSL distinguished name, or DN). The client's identity is then checked against any client group ACLs, which contain lists of authorized clients. If a client's identity matches that in an ACL, it is granted permission to join the client group; if not, it is denied access.

Important: Client group ACLs only work if basic authentication or SSL authentication has been configured for the client group's Broker Server, and is enabled.

To protect your Broker data with a client group ACL

1. Determine the document types and logs whose information should be restricted.
2. Add those document types and logs to the client group you plan to secure (make sure those document types and logs have not been listed in any non-ACL-protected client groups).
3. Decide which clients will have access to those documents and logs.
4. Add the basic authentication identities or the SSL identities of these clients to the client group ACL.

Important: Always protect the admin client group with ACLs, or else any user can gain administrative access to any document type (see [“admin Client Group” on page 177](#)).

For general information about the use of ACLs in Broker, refer to [“Access Control Lists” on page 325](#). For specific information about client group ACLs, refer to [“Client Group ACLs” on page 326](#) and [“About Configuring Client Group ACLs” on page 337](#).

System-Defined Client Groups

Broker creates several system-defined client groups. You can make only limited changes to these groups:

- You cannot modify any of their properties; that is, you cannot change the description, name, lifecycle or storage type for these client groups.
- You cannot delete system-defined client groups, except for the eventLog client group.

You can modify the document types belonging to the can-publish, can-subscribe, log-publish, and log-acknowledge lists for system-defined client groups.

admin Client Group

Members of the admin client group have full permissions and administrative privileges for a given Broker. Each Broker has one admin client group. Clients belonging to the admin client group can:

- Add any document type to any client group's can-publish and can-subscribe lists.
- Create client groups.
- Browse and modify any client's queue.
- Modify any Broker setting.
- Configure the territories or local gateways to which the Broker belongs.

Generally, you should use membership in the admin client group to perform Broker administrative functions. As such, it is not necessary for admin client information to be saved between sessions; clients belonging to the admin client group have a queue storage setting of **volatile** and a lifecycle setting of **destroy on disconnect**.

It is strongly recommended that you restrict membership of the admin client group to only those users who should have administrative privileges, otherwise, any client could join the admin client group and gain full administrative access to the Broker. To limit access to the admin client group, configure its ACLs after you have configured the Broker Server for basic authentication or SSL.

For more information, see [“admin Client Group ACLs” on page 328](#) and [“About Configuring Client Group ACLs” on page 337](#).

accessLabelAdapter Client Group

The accessLabelAdapter client group corresponds to an advanced security feature called the Access Label Adapter (ALA). The ALA is a C or Java API program using a client on the system-defined accessLabelAdapter client group.

Access labels allow you to prevent a Broker client from accessing a specific document regardless of whether its client group is authorized to access the document type to which the document belongs.

The accessLabelAdapter client group has a `destroy on disconnect` lifecycle and a `volatile` storage type. You can modify the access control list (ACL) of the client group so that ALAs can authenticate using basic authentication or SSL; otherwise, you cannot modify the client group.

For more information, see [“Using Access Labels” on page 585](#).

adapters Client Group

This system-defined client group is for the 4.x Adapter Developer Kit (ADK), used in previous versions of Broker. The adapters client group has `volatile` queue storage and

a `destroy` on `disconnect` lifecycle. The adapter developer kit is deprecated for this release of Broker.

eventLog Client Group

This client group can subscribe to all document types defined within the Broker. If your installation is not using the document logging utility, you should delete this client group for security reasons. For more information, see [“Deleting a Client Group” on page 185](#).

If there are customized document logging utilities that require the eventLog client group, you can secure it with an ACL. For more information, see [“About Configuring Client Group ACLs” on page 337](#).

The eventLog client group is used by a Broker's dead letter queue facility, which receives messages for which there are no subscribers (for more information, see [“Routing Documents to the Dead Letter Queue” on page 144](#)).

The eventLog client group is deprecated for this release of Broker.

Working with Client Groups

The following sections describe the most commonly used procedures for working with Broker client groups:

- Searching for client groups
- Creating client groups
- Viewing information about client groups
- Editing client group settings
- Assigning publish and subscribe permissions
- Copying a client group from one Broker to another
- Deleting a client group

Searching for Client Groups

Use the procedure outlined below to search for one or more client groups based on these criteria:

- Name of a client group, or text string included in the client group name. Note that you can use wild card characters such as `*` (for 0 to n instances of any character) and `?` (for a single instance of any character).
- The Broker Server and Broker to which the client group or groups belong.

For more information about using the My webMethods search feature, refer to the *Working with My webMethods* guide.

To search for a client group

1. In My webMethods: **Messaging > Broker Servers > Client Groups**.
2. Click the **Search** tab if it is not already displayed.
3. Select the Broker Server and Broker whose client groups you want to list by selecting:
 - The Broker Server from the **Server Name** list
 - The Broker from the **BrokerName** list
4. In the text box, type the keywords that are contained in the name of the client group you want to find. For example, you might specify:
 - The complete name of the client group (for example, "admin")
 - A partial name (for example, "prod") to select all client groups that contain the phrase

If you leave the text box blank, My webMethods searches for all client groups on the specified Broker. For more information about using the search facility, see the *Working with My webMethods* guide.

5. Click **Go** to execute the search.

For information about saving the search criteria for reuse, see the *Working with My webMethods* guide.

Creating a Client Group

When you create a client group, you are also configuring several default property settings for any client that will join the group. You must have a client group created and defined before you can add clients.

Following is a summary of the high-level steps to follow when creating a client group:

1. Assign a client group name, description, lifecycle, and client queue storage type to the client group. See [“Creating a Client Group” on page 179](#) below for instructions.
2. Determine the document types to which the client group can publish and subscribe. See [“Adding Document Type Permissions for Client Groups” on page 184](#) for instructions.
3. Optionally, configure ACLs for the client group. You must have configured the Broker Server for basic authentication or SSL authentication before you configure ACLs.

To create a client group

1. In My webMethods: **Messaging > Broker Servers > Client Groups**.
2. On the Client Groups page, click **Add Client Group**.

- On the Add Client Groups page, enter the following information.

Field	Description
Broker Server	Select the Broker Server for the client group from list.
Broker	Select the Broker to which you want to add the client group from the list.
Client Group Name	Enter a name for the new client group. For information about the naming restrictions for client groups, see the appendix "Parameter Naming Rules" in the <i>webMethods Broker Client Java API Programmer's Guide</i> .
Client Group Description	Optional. Enter a brief description of the client group.
Queue Storage Type (Lifecycle)	Determine the default storage type for client queues and how the Broker deals with a client's state upon disconnection (the properties are set as a pair). If you do not make a selection, Broker sets the default values to Guaranteed (explicit destroy) . For more information, see " Client Queue Storage Type " on page 174 and " Client Lifecycle " on page 175.

- Click **Add**.

Viewing and Editing Client Groups

You can display a list of client groups stored on a selected Broker Server and Broker, select a client group to view its settings, and edit certain settings.

To view the list of client groups, view client group details, and edit client group settings

- In My webMethods: **Messaging > Broker Servers > Client Groups**.
- Select the Broker Server and **Broker** whose client groups you want to list and click **Go**.

The Broker user interface displays all client groups stored on that Broker in the **Client Groups List**. Following are descriptions of the fields:

Fields	Description
Client Group	List of client groups belonging to the selected Broker Server and Broker.

Fields	Description
Broker@Server	Name of the Broker and Broker Server Host to which the client group belongs, in the format: <i>BrokerName@hostname:port</i>
Queue Type	Default storage type for client queues. Possible values are: <ul style="list-style-type: none"> ■ Guaranteed. Documents of this type are stored on disk (default). ■ Volatile. Documents of this type are stored in memory. For more information, see “Client Queue Storage Type” on page 174.
Lifecycle	Determines what the Broker does with a Broker client's state upon disconnection. Possible values are: <ul style="list-style-type: none"> ■ Explicit destroy. Maintains the client's state object when the client disconnects. ■ Destroy on disconnect. The Broker loses all knowledge of the client upon disconnection. For more information, see “Client Lifecycle” on page 175.
System Defined	A value of <code>Yes</code> indicates that the client group was defined by Broker. For more information, see “System-Defined Client Groups” on page 176.
Created	Date and time that the client group was created.

- For detailed information about a client group, or to edit a client group's settings, click its name. On the Client Group Details page is displayed, select the **Configuration** tab.

Note: You can only modify the settings for **Description**, **Encryption**, and **Access Label Required**.

Information	Description
Description	Description of the client group. You can edit the text in this field.
Lifecycle	Specifies what the Broker does with a Broker client's state when the Broker client disconnects. Possible values are:

Information	Description
	<ul style="list-style-type: none"> ■ Explicit destroy. Maintains the client's state object when the client disconnects. ■ Destroy on disconnect. The Broker loses all knowledge of the client upon disconnection. <p>For more information, see “Client Lifecycle” on page 175.</p>
Queue Type	<p>Default storage type for client queues. Possible values are:</p> <ul style="list-style-type: none"> ■ Guaranteed. Documents of this type are stored on disk (default). ■ Volatile. Documents of this type are stored in memory. <p>For information, see “Client Queue Storage Type” on page 174.</p>
Created	Date and time the client group was created.
Last Modified	Date and time the client group was last modified.
System Defined	Specifies whether the client group is system defined and whether client group properties can be modified.
Encryption	<p>Specifies whether message traffic between clients belonging to this client group and the Broker is encrypted (checked) or unencrypted (unchecked). By default, encryption is enabled.</p> <p>To enable encryption for a client group, its Broker Server must have an SSL identity, and the current user must have permission to configure encryption. For more information, see “Assigning SSL Identities for a Broker Server ” on page 309 and “Using SSL Encryption” on page 316.</p>
Access Label Required	<p>Specifies whether clients must have an access label to join the client group (the default is no access label required).</p> <p>Access labels are an advanced security feature of Broker. For more information, see “Using Access Labels” on page 585.</p>
Total Docs Published	Total number of documents that Broker clients in this group have published.

Information	Description
Last published	Date and time that a Broker client in this group last published a document.
Access control	<p>Provides status information about the ACLs for this client group and about permissions for the current user. Possible values are listed below:</p> <p>The following indicate that you can configure or view ACL settings:</p> <ul style="list-style-type: none"> ■ Configured. ACLs are configured and enabled for this client group. ■ Not configured. No ACLs have been configured for this client group, therefore, any clients are permitted to join. <p>The following indicate that you can configure ACLs, but must first configure basic authentication or SSL for other Broker components, as follows:</p> <ul style="list-style-type: none"> ■ Broker Server SSL required. You must establish an SSL identity to the Broker Server before you can configure ACLs for this client group. ■ Identity required. You must establish a basic authentication or an SSL identity for the Broker user interface component before you can configure ACLs for this client group. <p>The following indicates that you do not have permission to configure or view ACLs:</p> <ul style="list-style-type: none"> ■ Configured, but not accessible. An ACL is configured for this client group, but the current user does not have the permissions needed to view or reconfigure these settings.

4. Click **Apply** to save any client group settings you modified.
5. To view a list of the document types that Broker clients in the client group are able to access, do the following:
 - a. Select the **Can Publish** tab to view the document types that clients are able to deliver.
 - b. Select the **Can Subscribe** tab to view the document types to which clients are able to receive deliveries.
 - c. Select the **Log Publish** tab to view the document types that clients are able to log when a document is published.

- d. Select the **Log Acknowledge** tab to view the document types to which clients can make a log entry when a document is received.

Adding Document Type Permissions for Client Groups

You can specify the document types Broker clients can publish, subscribe to, and log, by assigning permissions to the following lists:

- Can publish
- Can subscribe
- Log publish (can log when published)
- Log acknowledge (can log when received)

The procedure for adding a document type to each of these lists is the same: select the client group, select a list, check the document types to add, repeat the last step for the next list. The steps are described below.

To add document type permissions for a client group

1. Perform the steps listed in [“Adding Document Type Permissions for Client Groups” on page 184](#).
Skip any steps for editing a client group.
2. Click **Add Document Type**.
3. On the Document Types page, check the document types whose permissions you want to add and click **Add**.

Removing Document Type Permissions for Client Groups

The steps for removing document type permissions for a client group are listed below.

To remove document type permissions for a client group

1. Perform the steps listed under [“Adding Document Type Permissions for Client Groups” on page 184](#).
Omit any steps for editing a client group.
2. Check the boxes of the document types to remove permissions and click **Delete**.

Copying Client Groups between Brokers

You can copy and paste client groups between any Brokers to which you have access. When you use the copy and paste functions, My webMethods copies and pastes the client group object to and from the clipboard.

If you do not have access to the target Broker, use the export-import functionality instead of copying and pasting. In that case, you export client group definitions from the source Broker, and the administrator for the target Broker imports the client group definitions.

In general, use copy and paste for simple objects, such as client groups, and importing and exporting for copying complex objects, such as a territory or a Broker.

For more information, see [“Managing Client Groups” on page 171](#).

Using Copy and Paste

The copy and paste function is available from any of the tabs on the Client Groups Search page.

To copy and paste a client group

1. In My webMethods: **Messaging > Broker Servers > Client Groups**.
2. Select the **Broker Server** and **Broker** of the client group you want to copy.
3. Click **Go**. If the client group does not appear in the list, use the **Search** tab to locate it (see [“Searching for Client Groups” on page 178](#)).
4. In the Search results, check the name of the client group you want to copy and then click **Copy**.
5. Click **Paste**. The **Select Target to Paste** dialog box appears.
6. Select the **Broker Server** and **Broker** (the target) on which to paste the copied client group.
7. Click **Paste**.

Deleting a Client Group

Use My webMethods to delete a client group.

Note: You cannot undo a client group deletion. You cannot delete system-defined client groups, except for the eventLog client group. For more information, see [“System-Defined Client Groups” on page 176](#).

To delete a client group

1. In My webMethods: **Messaging > Broker Servers > Client Groups**.
2. Select the **Broker Server** and **Broker** of the client group you want to delete.
3. Click **Go**. If the client group does not appear in the list, use the **Search** tab to locate it (see [“Searching for Client Groups” on page 178](#)).
4. On the Client Groups page, click the box next to the client groups you want to delete and then click **Delete**.

Note: You cannot delete a client group if Broker clients in that group are connected to the Broker. You must delete the Broker clients before deleting the client group.

8 Managing Clients

■ Overview	188
■ What Is a Client?	188
■ Viewing Clients on a Broker	190
■ Viewing Client Configuration Information	193
■ Viewing Client Statistics	196
■ Managing Subscriptions	199
■ Managing Sessions	200
■ Copying Clients between Brokers	205
■ Deleting Clients	206

Overview

This chapter explains how to view clients that exist on a Broker, examine client configuration information, and monitor client activity. This chapter also describes ways in which you can manage and modify these clients, such as changing document subscriptions, disconnecting sessions, or deleting a client.

For information about creating and naming clients, refer to the appropriate programming interface manual. For information about managing the contents of a client queue, see [“Managing Client Queues” on page 231](#). For information about using test clients, see [“Working with Test Clients” on page 207](#).

What Is a Client?

A client is an object on the Broker that represents the connection between a client program and the Broker. A client program creates a client to publish or retrieve documents. A client might be created by a client program (using JMS, C#, or Java APIs), webMethods Integration Server (triggers), and My webMethods (clients for performing administrative tasks).

A client program publishes documents to a Broker Server using its client. The Broker Server's publishing engine routes documents published by a client to other clients that subscribe to those documents.

A client program subscribes to documents by registering document type subscriptions with its client. Broker places the documents to which a client subscribes in the client's queue. A client program then retrieves the documents from its client queue.

For example, a customer-facing Web application might create a client to publish sales order documents. The back-end accounting system and the order fulfillment systems might create their own clients to subscribe to the sales order documents. Broker routes the documents published by the Web applications client to the clients for the accounting and order fulfillment systems.

A client program creates a client using one of the Broker APIs. For more information about creating a Broker client, see the appropriate programming guide.

Client State

Broker maintains state information for each client. State information includes the client ID, subscriptions, a queue of documents, sessions, and the client group to which the client belongs. For more information about state information maintained for a client, see [“Clients \(Client State Objects\)” on page 37](#).

Shared State Clients

Multiple client programs (or multiple threads within a client program) can share the same client if the client was configured as a shared state client when it was created. Sharing a client allows multiple client programs (or multiple threads within a client program), each using its own session, to process documents from a single client queue in parallel on a first-come, first-served basis.

Any changes to the state for a shared-state client affect all of the sessions using that client. For example, any changes to the document subscriptions or to the queue, such as clearing it, affect all the client sessions that connect to the shared-state client.

For more information about configuring clients to share a client state, see the appropriate programming guide.

Shared State Order

When you create a shared-state client, you can determine how the Broker distributes documents to the client sessions that share the client queue.

- **Publisher.** Broker guarantees that documents from each publisher are delivered to the client sessions in the order that the documents are published.

Note: If the receiving client session does not acknowledge the receipt of the document immediately, Broker delivers all subsequent documents from the same publisher to the same receiving client session. This continues until the receiving client acknowledges all the documents that it has received from the publisher.

- **None.** Broker delivers documents to client sessions in any order. Broker might deliver documents from a single publisher in an order that does not match the publishing order.

For more information about configuring a shared state order, see the appropriate programming guide.

Storage Type and Lifecycle

The storage type for a client determines whether the client's state information is stored in memory or on disk. The client lifecycle determines whether the Broker maintains a client's state object after the client disconnects from the Broker. The client group to which a client belongs determines the storage type and the lifecycle for the client. For more information about setting the storage type and lifecycle properties for a client group, see [“Client Queue Storage Type” on page 174](#) and [“Client Lifecycle” on page 175](#).

Viewing Clients on a Broker

Using My webMethods, you can view a list of the clients on a Broker. You can display all the clients or you can filter the list of clients to see only those that meet specific criteria. For example, you might want to view only those clients belonging to a particular client group.

Use an advanced search to search for specific clients by specifying detailed criteria. You perform the advanced search based on application name, category, client group, connection status, number of documents queued, and/or date of the last document retrieved.

Note: Clients with lifecycles of destroy on disconnect only appear on the **Client List** if an active session exists for that client.

To view clients on a Broker

1. In My webMethods: **Administration > Messaging > Broker Servers > Clients**.
2. To perform a keyword search, do the following:
 - a. In the **Keywords** text box on the **Keyword** tab, type the keywords contained in the client ID that you want to find. For example, you might specify:
 - The complete client ID.
 - A portion of the client ID to locate all the client IDs that contain the provided text.

If you leave the text box blank, My webMethods searches for all the clients on the specified Broker.
 - b. Select the Broker Server and Broker whose clients you want to view by doing the following:

Field	Description
Server Name	Fully-qualified name of the Broker Server: <i>hostname:port</i> If the Broker Server has the default port of 6849, only the Broker Server <i>hostname</i> will appear.
BrokerName	Name of the Broker as defined on the Broker Server.

3. To perform an advanced search, click the **Advanced** tab and do the following:
 - a. Select the Broker Server and Broker whose clients you want to view by doing the following:

Field	Description
Server Name	Fully-qualified name of the Broker Server: <i>hostname:port</i>
BrokerName	Name of the Broker as defined on the Broker Server.

- b. In the Filter panel, specify at least one filter. Select the field name, operator, and the value for the filter criteria.

Field Name	Operator	Value
Application Name	<input type="checkbox"/> Equal To	Type the application name of the client you would like to search.
Category	<input type="checkbox"/> Equal To	<input type="checkbox"/> Durable Subscriber <input type="checkbox"/> Queue <input type="checkbox"/> Test Client
Client Group	<input type="checkbox"/> Equal To	Select a client group from the list of defined client groups on the Broker.
Connection Status	<input type="checkbox"/> Equal To	<input type="checkbox"/> Connected <input type="checkbox"/> Not Connected
Number of Documents Queued	<input type="checkbox"/> Equal To <input type="checkbox"/> Greater Than <input type="checkbox"/> Less Than	Type a number to specify the document queue length of the client.
Last Document Retrieved	<input type="checkbox"/> After <input type="checkbox"/> Before <input type="checkbox"/> Never	Specify the date and time to check when the last document was retrieved from the client queue.

- c. To add a new filter click **Add**. To remove a filter, click **Delete** corresponding to the filter you want to delete.
4. Select the search condition from the **Search Condition** list.

5. Click **Search**. My webMethods displays the search results in the **Client List**.

The following table describes the contents of the **Client List**.

Field	Description
	<p>Indicates the connection status of the client as follows:</p> <ul style="list-style-type: none"> ■  The client is connected to the Broker Server. ■  The client is not connected to the Broker Server. <p>Displays  if the client queue is locked. For information about locking or unlocking a client queue, see “Locking a Client Queue” on page 233 and “Unlocking a Client Queue” on page 234.</p>
Client ID	A unique identifier for the client. When creating a client, you can specify a client ID or allow the Broker to generate a unique ID.
Application Name	Name of the application that created the client. The application name can be any string of characters and is specified when the client is created.
Client Group	Name of the client group to which the client belongs. The client group determines client properties such as storage type, lifecycle, and the document types that a client can publish or subscribe to. For more information about client groups, see “Managing Client Groups” on page 171 .
SSL Sessions	Indicates the number of sessions using SSL if an SSL connection is established between the client program and the Broker.
Docs Queued	Indicates the number of documents available for delivery in the client queue. For more information about managing the documents in a client queue, see “Managing Client Queues” on page 231 .
Last Retrieved	Indicates the date and time when the last document was retrieved from the client queue. Displays <i>Never</i> if no documents were retrieved from the client queue.
Note:	To view an updated list of clients, click Search .

Viewing Client Configuration Information

For each client that exists, Broker maintains configuration information specified at the time the client was created and information about current client queue state.

To view client configuration information

1. In My webMethods: **Messaging > Broker Servers > Clients**.
2. In the **Client List**, in the **Client ID** column, click the name of the client for which you want to view configuration information. If the client does not appear in the list, use the search functionality to locate it.

Note: If you selected a test client, My webMethods displays a **Test Client View** button in the **Client Information** tab. Click this button to switch to test client view. For more information about test clients, see [“Working with Test Clients” on page 207](#).

3. On the Client Details page, click the **Configuration** tab to view configuration information about the client.

The following table describes the contents of the **Configuration** tab.

Field	Description
Application Name	Name of the application that created the client. The application name can be any string of characters and is specified when the client is created.
Shared Document Order	Specifies whether or not Broker delivers documents to the receiving client program in the same order in which the documents are published. <div data-bbox="613 1423 1352 1556" style="background-color: #f0f0f0; padding: 5px; margin-top: 10px;"> <p>Note: This property is meaningful for shared-state clients only. If a client does not share state, Broker always delivers documents to the receiving client in the order in which the documents are published.</p> </div> <ul style="list-style-type: none"> ■ Publisher. Broker delivers documents from each publisher to the client programs that share the client state in the same order in which the documents are published. This is the default. ■ None. Broker does not enforce the by-publisher order when delivering documents to the client programs that share the client state.

Field	Description
State Sharing	<p>Indicates whether or not state sharing is enabled for this client. In a shared-state configuration, multiple client sessions share a single client queue.</p> <ul style="list-style-type: none"> ■ Enabled. Specifies that state sharing is enabled. Multiple clients, possibly residing on multiple hosts, share the same client state and can process documents in the client queue on a first-come, first-served basis. ■ Disabled. Specifies that state sharing for this client is disabled.
Created	Date and time this client was created.
Client Group	Client group to which the client belongs. Click the client group name to view the Client Group Details page for that client group. For more information about client groups, see “Managing Client Groups” on page 171 .
Infoset	The infoset containing the state or configuration information for a client. This is for pre-6.0 Brokers only.
Publish Sequence Number	<p>Specifies the last sequence number for published documents received by this client. A publishing client program assigns sequence numbers to documents to allow Broker to recognize and discard duplicate documents from the same publisher. A sequence number of 0 typically indicates that the publishing client does not use publish sequence numbers.</p> <p>For more information about sequence numbers, see the relevant programming guide.</p>
Forced Reconnect	<p>Specifies whether a client program can forcibly reconnect to a Broker even if it appears that the client program is already connected to the Broker. In some situations, a Broker might not recognize a lost client connection. For example, if you disconnect the machine on which the client program is running, the Broker may not detect the lost connection for some time and will consider the client program to still be connected.</p> <p>Forced reconnect is meaningful only for clients that do not share a client state. For shared-state clients, Broker always creates a new client session when processing a connection request.</p>

Field	Description
	<p>This applies to Broker versions 6.x and later.</p> <ul style="list-style-type: none"> ■ Yes. Indicates that Broker will accept the reconnection request from a client even if it appears that a connection already exists. If the client is currently connected, Broker disconnects the existing session before establishing a new connection. ■ No. Indicates that Broker will reject requests from this client to reconnect if it appears that a connection already exists.
Queue Locked	<p>Indicates whether or not the client queue is locked. For more information about locking client queues, see “Locking a Client Queue” on page 233.</p> <ul style="list-style-type: none"> ■ Yes. Indicates the queue is locked. You must lock a queue before you can edit the contents of the queue. When a queue is locked, new documents can be placed in the queue, but clients cannot retrieve documents from the queue. ■ No. Indicates the queue is unlocked.
Lock Held by Client ID	<p>Client ID for the client that locked the queue. If the queue is unlocked, this field is blank. If the queue is locked, click the client ID to view details about the client that holds the lock. For information about unlocking client queues, see “Unlocking a Client Queue” on page 234.</p>
Lock Held by Client Session	<p>Session ID for the session that holds the lock to the client queue. Click the session ID to view session information for the client that is holding the lock. For more information about client sessions, see “Managing Sessions” on page 200.</p>
Lock Held Since	<p>Date and time at which the client was locked.</p>
Access Label	<p>Access label value if one has been assigned to the client. For more information about access labels, see “Using Access Labels” on page 585.</p>

4. Click **Close** to return to the Clients page.

Viewing Client Statistics

My webMethods maintains statistics about a client including:

- Queue data, such as the number of documents in the queue, the number of unacknowledged documents in the queue, and the last time a document was placed in the queue
- Document information, such as the total number of documents a client has published or delivered
- Session information, such as the last time a client program established a session with the client and the time at which the client was created

These statistics can help you monitor the amount of activity for a client, determine if the client publishes and delivers the expected number of documents, and verify that client programs retrieve documents from the client queue. You might want to take corrective action based on these statistics. Statistics can also be useful for debugging purposes.

The statistics displayed on the **Statistics** tab reflect data gathered for all the sessions for a client since the client was created. If the client is a shared-state client, the **Statistics** tab includes data for all sessions.

To view client statistics

1. In My webMethods: **Messaging > Broker Servers > Clients**.
2. On the **Client List**, in the **Client ID** column, click the name of the client for which you want to view statistics. If the client does not appear in the list, use the search functionality to locate it.
3. On the Client Details page, click the **Statistics** tab. The **Statistics** tab groups data into three different categories: Client Queue, Documents, and Sessions.

The following table describes the information contained in the Client Queue section:

Field	Description
Length	<p>Total number of documents currently in the client queue, including documents not yet retrieved by client programs and documents retrieved but not yet acknowledged by client programs.</p> <p>Click the Clear Queue button to remove all documents from the queue, including retrieved but unacknowledged documents. For more information about clearing the client queue, see “Clearing Client Queues” on page 242.</p>

Field	Description
Deliverable Length	Total number of documents in the client queue available to client programs. This does not include documents retrieved but not yet acknowledged by the client program.
Unacknowledged Length	Total number of documents that client programs have retrieved from the client queue but have not yet acknowledged to the Broker. Broker removes documents from the client queue after receiving an acknowledgement only.
Size	Size of the queue in bytes.
Last Queued	Date and time that Broker last placed a document in the queue.
Last Retrieved	Date and time that a client program last retrieved a document from the queue.
Last Published	Date and time at which the client last published a document.
Last Delivery	Date and time at which the client last delivered a document.

4. The following table describes the information contained in the Documents section:

Field	Description
Total Retrieved	Total number of documents retrieved from the client queue by client programs.
Total Published	Total number of documents published by this client.
Total Delivered	Total number of documents delivered by this client.
Total Queued	Total number of documents placed in the queue since the client was created.

Field	Description
Highest in Queue	The most number of documents ever in the queue concurrently and the date and time at which that occurred.
Recent Deliveries or Total Deliveries	<ul style="list-style-type: none"> ■ Recent Deliveries displays the number of documents delivered by this client within the last statistics polling period. The value of the Time interval between statistical refresh field specifies the statistics collection interval. <p>Note: Recent Deliveries data is available only if you have selected Enable Statistical Polling.</p> <ul style="list-style-type: none"> ■ Total Deliveries displays the total number of documents delivered by this client. <p>Note: Total Deliveries data is available only if you do not select Enable Statistical Polling.</p>

5. The following table describes the information contained in the Sessions section:

Field	Description
Last Retrieved Session	Session identifier for the client session that most recently retrieved a document from this client queue.
Last Connected Session	Session identifier for the most recently established client session.
Last Disconnected Session	Session identifier for the most recent client session that disconnected from the Broker.
Last Connected Time	Date and time the most recent client session was established with the Broker.
Last Disconnected Time	Date and time that the last client session disconnected from the Broker.
Created Time	Date and time this client was created.

6. Click **Close** to return to the Clients page.

Managing Subscriptions

The document types to which a client subscribes determine the documents that Broker places in the client's queue. Using My webMethods, you can view and delete subscriptions for a client.

Note: If a guaranteed client has more than one thousand document type subscriptions, the Broker may become noticeably slower when handling new subscription requests made by the client.

Viewing Subscriptions

When you display the document types to which a client subscribes, you can also view any filters saved with the subscriptions. A filter specifies criteria that documents must meet before being placed in the client queue.

To view subscriptions for a client

1. In My webMethods: **Messaging > Broker Servers > Clients**.
2. On the **Client List** tab, in the **Client ID** column, click the name of the client for which you want to view document type subscriptions. If the client does not appear in the list, use the search functionality to locate it.
3. On the Client Details page, click the **Subscriptions** tab.

The following table describes the contents of the **Subscriptions** tab.

Field	Description
Document Type	Name of the document type to which the client subscribes. Click the document type name to view and modify document type details. For more information about document types, see “Managing Document Types” on page 157 .
Subscription ID	A unique identifier for this document type subscription. If a subscription ID was not specified when the client established a subscription to this document type, the Subscription ID will be 0. For more information about specifying a subscription ID, see the relevant programming guide.
Filter	The filter registered with the document type subscription. If a document to which this client subscribes does not meet the

Field	Description
	criteria specified in the filter, Broker discards the document. If this column is blank, the subscription does not have a filter.

- Click **Close** to return to the Clients page.

Deleting Subscriptions

You can delete any document type subscriptions for a client. You might want to delete a subscription to stop the unwanted or repeated delivery of documents from another client. (The other client might be publishing invalid instances of the document type or might be publishing more instances of the document than the current client can handle.)

To delete subscriptions for a client

- In My webMethods: **Messaging > Broker Servers > Clients**.
- On the **Client List** tab, in the **Client ID** column, click the name of the client for which you want to delete document type subscriptions. If the client does not appear in the list, use the search functionality to locate it.
- On the Client Details page, click the **Subscriptions** tab.
- For each document type subscription that you want to delete, select the check box next to the name of the document type.
- Click **Delete**.
- Click **Close** to return to the Clients page.

Managing Sessions

A session represents a connection between an active client program and a client. A client program typically has one active session to a client.

A client can have multiple concurrent sessions if the client is configured as a shared-state client. A shared-state client allows multiple sessions, possibly involving clients on multiple hosts, to share the same client state, including its client ID, subscriptions, and queue. By default, each client is configured to allow only one active session. For more information about shared-state clients, see [“Shared State Clients” on page 189](#).

Broker assigns each connection between a client program and the client object a unique session identifier.

Using My webMethods, you can monitor and manage sessions to a client by:

- Viewing session statistics, such as the time a client program last established a session with a client. For more information about session statistics, see [“Viewing Client Statistics” on page 196](#).

- Viewing a list of active sessions to the client.
- Viewing detailed information about a session including encryption information.
- Disconnecting a session.

The following sections provide more details about these activities.

Viewing Session Information

You can view a list of the active sessions between client programs and a client. A client can have multiple active sessions if the client is a shared-state client.

To view session information for a client

1. In My webMethods: **Messaging > Broker Servers > Clients**.
2. On the **Client List** tab, in the **Client ID** column, click the name of the client for which you want to view session information. If the client does not appear in the list, use the search functionality to locate it.
3. On the Client Details page, click the **Sessions** tab.

My webMethods displays the following information for each active session:

Field	Description
From	The IP address of the client application connecting to the client. Click this address to view detailed information about the client program that established this session. For more information about viewing detailed client information, see “Viewing Detailed Session Information” on page 202 .
ID	A numeric identifier for the session generated by the client.
Authentication	Indicates whether the client uses basic authentication or SSL authentication to connect to Broker. <ul style="list-style-type: none"> ■ Basic. Indicates that this session uses basic authentication to connect to Broker. ■ SSL. Indicates that this session uses SSL to connect to the Broker. ■ None. Indicates that the session is not using any authentication protocol for this connection with Broker.
Creation Time	Date and time the client program established the client session with the Broker.

Field	Description
Last Activity	Date and time the client program last made a request to the Broker.

- Click **Close** to return to the Client Details page.

Viewing Detailed Session Information

Broker maintains detailed information about each session, including:

- Client program information such as the version of the Broker client API used to connect to the client
- Encryption information

The following procedure describes how to access this information and provides descriptions of the detailed information maintained for each session.

To view detailed session information

- In My webMethods: **Messaging > Broker Servers > Clients**.
- On the **Client List** tab, in the **Client ID** column, click the name of the client for which you want to view detailed session information. If the client does not appear in the list, use the search functionality to locate it.
- On the Client Details page, click the **Sessions** tab.
- On the **Sessions** tab, click the IP address for the session for which you want to view detailed information. My webMethods displays the Client Session Detail page.

The **Session Details** tab groups client session information into two different categories: Platform Information and **Encryption and Authentication Information**. The following tables describe the information contained in each of these sections.

The data displayed Platform Information provides information about the client program that established the session with the client. The following table shows the Platform Information fields.

Field	Description
API Language	The Broker API (Java, C, or JMS) that the client program used to connect to the Broker.
API Language Version	The version of the Broker API used to connect to the Broker.
Hardware	Hardware on which the client program runs.

Field	Description
OS	Operating system on which the client program runs.

The fields under **Encryption and Authentication Information** display details about the encryption and authentication that the client program used to establish the session on the Broker. The following table shows the **Encryption and Authentication Information** fields.

Field	Description
Encryption	<p>Encryption level and version number for the encryption software that the client program uses.</p> <p>"None" indicates that the client application does not use encryption software.</p>
Authentication Protocol	<p>The authentication protocol the client program is using to connect to Broker.</p> <ul style="list-style-type: none"> ■ Basic. The client program is using basic authentication protocol. ■ SSL. The client program is using SSL protocol. ■ None. The client program is not using any authentication protocol.
User Name	<p>The user name used by the client program for authentication.</p> <ul style="list-style-type: none"> ■ Indicates the distinguished name if the client connects to Broker through SSL authentication. ■ Indicates the basic authentication username provided by the client program if it has connected to Broker through basic authentication.
Authenticator Name	<p>The authenticator name used by the client program for authentication.</p> <ul style="list-style-type: none"> ■ Indicates the distinguished name of the certification authority if the client connects to Broker through SSL authentication. ■ Indicates the authenticator alias specified in the basic authentication configuration file if the client connects to Broker through basic authentication.

About Disconnecting Sessions

You can disconnect an individual session between a client program and client. If the client is a shared-state client with multiple active sessions or client with an explicit destroy lifecycle, disconnecting a session preserves the client, including the client queue and its documents.

Before Disconnecting a Session

Keep the following points in mind before you disconnect a session:

- If you disconnect all the active sessions for a client, you effectively disconnect the client program from the Broker.
- If you disconnect the last session to a client with a destroy on disconnect lifecycle, Broker deletes the client.
- If you disconnect the last session for a client with an explicit destroy lifecycle, the client remains. The client program can reconnect to the client at a future time.
- Any transactions currently executing within the session will be affected. The way in which a transaction is impacted depends on the state of the transaction at the time of the disconnection. In general, any open transactions (transactions that are not completed or prepared), will be aborted.
- Any unacknowledged guaranteed documents that the client program retrieved from the client queue will be made available for redelivery when the client session is disconnected. A client program with an active session to the client can retrieve those documents again.

Note: If an explicit destroy client disconnects for a long period of time, documents can begin to accumulate in memory and potentially cause the Broker to run out of resources. Make sure that explicit destroy clients retrieve their documents frequently, especially if their subscription volume is high.

Disconnecting a Session

Use the following procedure to disconnect a session.

To disconnect a session

1. In My webMethods: **Messaging > Broker Servers > Clients**.
2. On the **Client List** tab, in the **Client ID** column, click the name of the client for which you want to view detailed session information. If the client does not appear in the list, use the search functionality to locate it.
3. On the Client Details page, click the **Sessions** tab.

4. On the **Sessions** tab, for each session that you want to disconnect, select the check box next to the IP address.
5. Click **Disconnect**.
My webMethods prompts you to confirm deleting the session.
6. Click **Disconnect** to disconnect the client session.

Copying Clients between Brokers

You can copy clients between any Brokers to which you have access in one of two ways:

- Copy and paste.
- Export and Import. When you use the export-import functionality, My webMethods places the exported client information in a file. The administrator for the destination Broker then imports the file. To use the export-import functionality, you must have administrator access. For more information about using the export-import functionality, see [“Exporting and Importing Broker Metadata” on page 483](#).

Copying and Pasting Clients

Keep the following points in mind when copying clients between Brokers:

- You must have access to the source and target Brokers.
- You can only copy a client to the target Broker if the client group to which the client belongs already exists on the target Broker. In turn, you can only copy a client group to a target Broker if the document types to which the client group has can-publish and can-subscribe permissions already exist on the target Broker.

To copy and paste clients between Brokers

1. In My webMethods: **Messaging > Broker Servers > Clients**.
2. On the **Client List** tab, in the **Client ID** column, select the check box next to the name of the each client that you want to copy. If the client does not appear in the list, use the search functionality to locate it.
3. Click **Copy**.
My webMethods copies the selected clients.
4. Click **Paste**. The Client List dialog box appears.
5. Under **Select Target to Paste**, select the **Broker Server** and **Broker** on which to paste the copied clients.
6. Click **Paste**.

Deleting Clients

You can delete clients that you do not need. When you delete a client, the client program disconnects from the Broker and the Broker destroys the client, including the client queue and any documents in the queue. This occurs regardless of the lifecycle of the client (destroy on disconnect or explicit destroy).

You can only delete a client with an unlocked client queue. The **Queue Locked** field on the **Configuration** tab indicates whether a client queue is locked or unlocked.

To delete a client

1. In My webMethods: **Messaging > Broker Servers > Clients**.
2. On the **Client List** tab, in the **Client ID** column, select the check box next to the name of each client that you want to delete. If the client does not appear in the list, use the search functionality to locate it.
3. Click **Delete**.

9 Working with Test Clients

■ Overview	208
■ What Is a Test Client?	208
■ Viewing Test Clients on a Broker	208
■ Creating Test Clients	209
■ Working in Test Client View	213
■ Managing Document Subscriptions for a Test Client	214
■ Publishing Documents with a Test Client	216
■ Retrieving Documents for a Test Client	223
■ Saving Outgoing or Incoming Documents	225
■ Disconnecting a Test Client	226
■ Reconnecting Test Clients	227
■ Deleting a Test Client	229

Overview

This chapter explains how to use test clients to diagnose errors or identify trouble spots for clients that publish and subscribe documents.

What Is a Test Client?

A test client is a function built in My webMethods that you can use to quickly create clients on the Broker Server. A test client can function as a client program running inside My webMethods and can act as a client on Broker Server.

In practical terms, a test client is a diagnostic tool that you can use to test the publishing and subscribing process. Sometimes, the publishing and subscription processes are developed independent of one another. In fact, the publishing and subscription might be created by different developers and might not be developed at the same time. Using a test client, you can publish and retrieve the published documents without building or writing one or more actual clients for this purpose.

You can use test clients to:

- Verify that a client receives the documents to which it has subscribed.
- Ensure that a filter saved with a subscription is valid and will allow only those documents that meet filter criteria to be placed in the client queue.
- Determine whether documents are getting stuck after publication or are being properly routed to the subscribers.

For example, suppose that a new adapter subscribes to a document type named `customerInfo`. You can use a test client to publish a `customerInfo` document and verify that the new adapter receives the document.

Test clients can be useful for diagnostic purpose when used in conjunction with the trace or topology features.

These features allow you to publish and then monitor a document routed by Broker from the publisher and distributed to subscribing clients. For more information about using the trace and topology features, see [“Using Topology View and Document Trace” on page 259](#).

Viewing Test Clients on a Broker

My webMethods displays test clients along with other clients in the **Clients List** on the Clients page. However, you might want to filter the **Clients List** to display only test clients.

To view test clients on a Broker

1. In My webMethods: **Messaging > Broker Servers > Clients**.
2. On the Clients page, click the **Advanced** tab.
3. Select the Broker Server and Broker whose clients you want to view by doing the following:

Field	Description
Server Name	Fully-qualified name of the Broker Server: <i>hostname:port</i>
BrokerName	Name of the Broker as defined on the Broker Server.

4. Under **Filter**, in the **Field Name** list, select **Category**.
5. In the **Value** list, select **Test Client**.
6. Click **Search**.

The **Client List** displays a list of all the test clients located on the specified Broker.

Creating Test Clients

When you create a test client, you must select the Broker Server and Broker on which the test client will be located. You also must determine the client group to which the test client will belong. The client group determines the following for a test client:

- The document types the test client can publish and subscribe.
- The lifecycle of the test client.
 - If the client group has a destroy on disconnect lifecycle, the test client is volatile. Broker will delete the test client when you log out of My webMethods or your session expires.
 - If the client group has an explicit destroy lifecycle, the test client is guaranteed. Broker maintains the test client and its client-state when your My webMethods session ends. Broker disconnects the test client when your session ends. You will need to reconnect the test client each time you log into My webMethods.

For more information about client lifecycle, see [“Client Lifecycle” on page 175](#).

- Whether the test client needs to provide the authorization information used by the test client's client group ACLs.

You can create a test client that uses either basic or SSL authentication. Proceed to one of the following sections to continue working with the new test client.

For information about...	See...
Working in test client view	“Working in Test Client View” on page 213
Managing document type subscriptions	“Managing Document Subscriptions for a Test Client” on page 214
Publishing documents with a test client	“Publishing Documents with a Test Client” on page 216
Retrieving documents with a test client	“Retrieving Documents for a Test Client” on page 223
Disconnecting a test client	“Disconnecting a Test Client” on page 226
Reconnecting a test client	“Reconnecting Test Clients” on page 227

Creating a Test Client that Uses Basic Authentication

Use the following procedure to create a test client that uses basic authentication.

To create a test client that uses basic authentication

1. In My webMethods: **Messaging > Broker Servers > Clients**.
2. In the **Client List**, click **Add Test Client**. My webMethods opens the Add Test Client dialog.
3. Provide the following information under **Client Configuration**.

In this field...	Specify...
Broker Server	Fully-qualified name of the Broker Server on which you want to create the test client in the format: <i>hostname:port</i> .
Broker	The Broker on which you want to create the test client.
Client Group	The client group to which you want to add the test client.

In this field...	Specify...
Client ID	The client ID that you want to assign to the test client. For information about naming restrictions for client identifiers, see the <i>webMethods Broker Client Java API Programmer's Guide</i> .

4. Click **Edit** to select the Test Client Authentication option.
5. In the **Test Client** field, select **Basic**.
6. Type the basic authentication user name in the **Username** field and password in the **Password** field.
7. If you want to specify the SSL and encryption settings along with basic authentication, perform the following:
 - a. In the **SSL Truststore** field, specify the absolute path and file name of the truststore file containing the trusted root for the SSL certificate stored in the specified keystore.
 - b. In the **SSL Truststore Type** field, select the file type of the truststore certificate file.
 - c. If you want to enable encryption, click the **Yes** option for **Encryption**.
8. Click **Create**. Broker Server creates the test client and returns you to the Clients page.

Creating a Test Client that Uses SSL Authentication

Use the following procedure to create a test client that uses SSL authentication.

To create a test client that uses SSL authentication

1. In My webMethods: **Messaging > Broker Servers > Clients**.
2. In the **Client List**, click **Add Test Client**. My webMethods opens the Add Test Client dialog.
3. Provide the following information under **Client Configuration**.

In this field...	Specify...
Broker Server	Fully-qualified name of the Broker Server on which you want to create the test client in the format: <i>hostname:port</i> .
Broker	The Broker on which you want to create the test client.
Client Group	The client group to which you want to add the test client.
Client ID	The client ID that you want to assign to the test client.

In this field...	Specify...
	For information about naming restrictions for client identifiers, see the <i>webMethods Broker Client Java API Programmer's Guide</i> .

4. Click **Edit** to select the Test Client Authentication option.
5. In the **Test Client** field, select **SSL**.
6. Perform the following:
 - a. In the **SSL Keystore** field, select the keystore file containing the SSL certificate to be used by the test client to join the client group.
 - b. In the **Password** field, type the password to the keystore file.
 - c. Click **Get User Names** to retrieve a list of distinguished names (DNs) from the keystore.
 - d. In the **User Name** list, select the user DN for the test client.
 - e. In the **SSL Keystore Type** field, select the file type of the client keystore.
 - f. In the **SSL Truststore** field, specify the absolute path and file name of the truststore file containing the trusted root for the SSL certificate stored in the specified keystore.
 - g. In the **SSL Truststore Type** field, select the file type of the truststore certificate file.
7. Click **Create**. Broker Server creates the test client and returns you to the Clients page.

Creating a Test Client that Uses One-way SSL Authentication

Use the following procedure to create a test client that uses one-way SSL authentication.

To create a test client that uses one-way SSL authentication

1. In My webMethods: **Messaging > Broker Servers > Clients**.
2. In the **Client List**, click **Add Test Client**. My webMethods opens the Add Test Client dialog.
3. Provide the following information under **Client Configuration**.

In this field...	Specify...
Broker Server	Fully-qualified name of the Broker Server on which you want to create the test client in the format: <i>hostname:port</i> .
Broker	The Broker on which you want to create the test client.

In this field...	Specify...
Client Group	The client group to which you want to add the test client.
Client ID	The client ID that you want to assign to the test client. For information about naming restrictions for client identifiers, see the <i>webMethods Broker Client Java API Programmer's Guide</i> .

4. Click **Edit** to select the Test Client Authentication option.
5. In the **Test Client** field, select **None**.
6. Perform the following:
 - a. In the **SSL Truststore** field, specify the absolute path and file name of the truststore file containing the trusted root for the SSL certificate.
 - b. In the **SSL Truststore Type** field, select the file type of the truststore certificate file.
7. Click **Create**. Broker Server creates the test client and returns you to the Clients page.

Working in Test Client View

Test client view is a set of pages and tabs in My webMethods that you use to build and manage a test client. In the test client view, you can:

- Disconnect or reconnect a test client.
- Create, view, and delete document type subscriptions of a test client.
- Publish or deliver documents with a test client.
- Retrieve documents received by the test client.

You can use the regular client view, called Broker client view, to manage and monitor the test client in the same way in which you would manage a regular client. For more information about managing clients, see [“Managing Clients”](#).

To switch between test client view and Broker client view

1. In My webMethods: **Messaging > Broker Servers > Clients**.
2. In the **Client List**, click the client ID of the test client with which you want to work. If the test client does not appear in the list, use the **Search** tab to locate it.
3. On the Client Details page, click **Test Client View** in the **Client Details Information** view.
My webMethods displays the Test Client Information view.
4. On the Client Details page, click **Broker Client View** in the Test Client Information view.

My webMethods displays the Client Details Information view.

Managing Document Subscriptions for a Test Client

When you create a test client to the test the subscription side of a publish-subscribe process, you need to establish document type subscriptions. The subscriptions you create should correspond to the document types published by the clients on the publishing side of the process. Although you need to work in the test client view to create subscriptions, you can view and delete subscriptions for a test client in either test client view or Broker client view.

Creating Subscriptions for a Test Client

As with any client, the client group to which a test client belongs determines the document types to which a test client can subscribe to or receive. A test client can subscribe to or receive any document types for which the client group has can-subscribe permissions. When a test client subscribes to a document type, Broker places all published instances of that document type in the test client's queue.

To create subscriptions for a test client

1. In My webMethods: **Messaging > Broker Servers > Clients**.
2. In the **Client List**, click the client ID of the test client for which you want to register document type subscriptions. If the test client does not appear in the list, use the **Search** tab to locate it.
3. On the Client Details page, click **Test Client View** in the Client Details Information view. My webMethods displays the Test Client Information view.
4. Click the **Subscriptions** tab. The **Subscriptions** tab lists all of the document types to which the client has can-subscribe permissions.
5. Select the check box next to each document type to which you want the test client to subscribe.
6. If you want to save a filter with the subscription, do the following:
 - a. Click  in the **Edit Filter** column. My webMethods displays the Test Client Document Type Filter page.
 - b. In the **Filter Expression** field, type the filter that you want to save with the subscription. Broker places only documents that meet the filter criteria in the test client queue. For information about filter syntax, see the *webMethods Broker Client Java API Programmer's Guide*.
 - c. Click **OK**.
7. Click **Subscribe**.

On the **Subscription** tab, My webMethods displays a "Yes" in the **Subscription** column for each document type to which the client subscribes.

Creating or Editing a Filter

You can create a filter to save with the document type subscription. A filter further refines a document type subscription because Broker only places documents whose contents meet the filter criteria in the client queue. You might create a filter for a test client to make sure that the filter is properly designed or that source documents are populated with the expected values.

To create or edit a filter

1. In My webMethods: **Messaging > Broker Servers > Clients**.
2. In the **Client List**, in the **Client ID** column, click the name of the test client for which you want to create document type subscriptions. If the test client does not appear in the list, use the **Search** tab to locate it.
3. On the Client Details page, click **Test Client View** in the Client Details Information view. My webMethods displays the Test Client Information view.
4. Click the **Subscriptions** tab.
5. Locate the document type for which you want to create or edit a filter.
6. Click  in the **Edit Filter** column. My webMethods displays the Test Client Document Type Filter page.
7. In the **Filter Expression** field, type the filter that you want to save with the subscription. Broker places only documents that meet the filter criteria in the test client queue. For information about filter syntax, see the *webMethods Broker Client Java API Programmer's Guide*.
8. Click **OK**. Broker saves the filter with the document type.

Deleting Subscriptions for a Test Client

When you delete a document type subscription for a test client, Broker stops placing documents of that type in the test client queue. You can delete a subscription for a test client using test client view or Broker client view. For more information about deleting a test client in Broker client view, see ["Deleting a Test Client"](#).

To delete subscriptions for a test client

1. In My webMethods: **Messaging > Broker Servers > Clients**.
2. In the **Client List**, click the client ID of the test client for which you want to delete document type subscriptions. If the test client does not appear in the list, use the **Search** tab to locate it.

3. On the Client Details page, click **Test Client View** in the Client Details Information view. My webMethods displays the Test Client Information view.
4. Click the **Subscriptions** tab.
5. For each current document type subscription that you want to delete, select the check box next to the document type name.
6. Click **Unsubscribe**. Broker removes the selected document type subscriptions.

Publishing Documents with a Test Client

Using a test client, you can publish documents to test document subscriptions registered by other clients. You might want to verify that Broker delivers a copy of a published document for each client that subscribes to that document type. You might also want to verify that a filter saved with a subscription is working properly, such that Broker only delivers documents that meet the criteria specified by the filter.

To publish documents with a test client, you need to do the following:

- Build a list of documents to publish.
- Publish the documents for all subscribers or deliver the documents to a specific client.

You use the **Outgoing Documents** tab to build and publish a set of documents for a test client.

Note: A test client can only publish documents for which the client has can-publish permissions. The client group to which a client belongs determines the document types for which a client has can-publish permissions. For more information about setting can-publish permission for a client group, see [“Adding Document Type Permissions for Client Groups” on page 184](#).

Building an Outgoing Documents List

To publish documents with a test client, you first need to build a list of documents to publish. The **Outgoing Documents** tab provides a staging area in which you can build this set of documents. You can build an outgoing documents list by doing one or more of the following:

- Adding blank documents
- Loading documents from a file
- Pasting documents copied from other locations in My webMethods

My webMethods maintains the outgoing documents list in its memory cache for the duration of your current session only. When you log out of My webMethods, your session is disconnected, or your session expires due to inactivity, My webMethods deletes the contents of the **Outgoing Documents** tab. If you want to use a set of documents

across multiple sessions, save the documents to a file. For more information about saving documents to a file, see [“Saving Outgoing or Incoming Documents” on page 225](#).

Adding Blank Documents

You can add empty documents to the outbound documents list one at a time. You can publish empty documents to verify that client subscriptions to that document type work and to trace the documents as Broker routes them.

Tip: If you want to add content to a document before publishing it, save the document to a file, use a text editor to modify the XML file, and then load the document into the outbound documents list. For information about saving documents, see [“Saving Outgoing or Incoming Documents” on page 225](#). For information about loading documents from a file, see [“Loading Documents from a File” on page 218](#).

To add an empty document

1. In My webMethods: **Messaging > Broker Servers > Clients**.
2. In the **Client List**, click the client ID of the test client for which you want to add empty documents to the outgoing documents list. If the test client does not appear in the list, use the **Search** tab to locate it.
3. On the Client Details page, click **Test Client View** in the Client Details Information view. My webMethods displays the Test Client Information view.
4. Click the **Outgoing Documents** tab.

Note: If the test client is not connected, the **Outgoing Documents** tab displays the message "Test client is not connected. Please reconnect the test client to get the information." For information about reconnecting a test client, see [“Reconnecting Test Clients” on page 227](#).

5. Click **Add Document** to display the Test Client - Add Documents dialog.
6. On the **Add Empty Document** tab, do the following
 - a. In the **Can Publish Document Type** list, select the document type for the empty document that you want to insert in the outgoing documents list.
 - b. Next to **Where to Add**, select the option that indicates whether you want to insert the documents at the beginning or end of the outgoing document list.

Note: After you insert documents, you can move them up or down in the outgoing documents list using the buttons in the **Move Up** and **Move Down** columns.

7. Click **Add**.

Loading Documents from a File

You can add documents to the outgoing documents list by loading documents from a file. You might have documents saved to a file if you want to use the same set of documents with multiple test clients or you want to use the same documents with a test client across multiple sessions.

You might also have documents saved to file if you want to add or edit document content and then publish the populated documents. When you publish populated documents, you can test subscription filters.

To load documents from a file

1. In My webMethods: **Messaging > Broker Servers > Clients**.
2. In the **Client List**, click the client ID of the test client for which you want to load documents in to the outgoing documents list. If the test client does not appear in the list, use the **Search** tab to locate it.
3. On the Client Details page, click **Test Client View** in the Client Details Information view. My webMethods displays the Test Client Information view.
4. Click the **Outgoing Documents** tab.

Note: If the test client is not connected, the **Outgoing Documents** tab displays the message "Test client is not connected. Please reconnect the test client to get the information." For information about reconnecting a test client, see ["Reconnecting Test Clients" on page 227](#).

5. Click **Add Document**. My webMethods displays the Test Client - Add Documents dialog.
6. Click the **Load Documents** tab.
7. In the **File Name** field, enter the path to the file containing the documents you want to add to the outgoing documents list. Click the **Browse** button to navigate to and select the file.
8. Next to **Where to Add**, select an option to indicate whether you want to insert the documents at the beginning or end of the outgoing document list.
9. Click **Load**.

If the test client does not have can-publish permissions for one or more of the documents in the file, My webMethods displays the following message:

```
Documents pasted. However, some documents were not pasted because
their document type didn't have can-publish permission.
```

Copying and Pasting Documents in the Outgoing Documents List

You can add documents to the outgoing documents list by pasting copied documents. You can copy documents from the outgoing documents list for this test client or another test client. You can also copy documents from the **Browse Queue** tab for a client.

Keep in mind that you can only paste documents that the client group to which the test client belongs has can-publish permissions.

To add documents to the outgoing documents list by copying and pasting

1. In My webMethods: **Messaging > Broker Servers > Clients**.
2. If you want to copy documents from the **Outgoing Documents** tab of a test client, do the following:
 - a. In the **Client List**, click the client ID of the test client for which you want to copy documents in the outgoing documents list. If the test client does not appear in the list, use the **Search** tab to locate it.
 - b. On the Client Details page, click **Test Client View** in the Client Details Information view. My webMethods displays the Test Client Information view.
 - c. Click the **Outgoing Documents** tab.
 - d. Select the check box next to each document that you want to copy.
 - e. Click **Copy**. Proceed to step 4.
3. If you want to copy documents from the **Browse Queue** tab for a client, do the following:
 - a. In the **Client List**, click the client ID of the client for which you want to copy documents in the **Browse Queue** tab. If the client does not appear in the list, use the **Search** tab to locate it.
 - b. On the Client Details page, click the **Browse Queue** tab.
 - c. Click **Start** to view the documents in the queue.
 - d. Select the check box next to each document that you want to copy.
 - e. Click **Copy**. Proceed to step 4.
4. Click **Clients** at the top of the current My webMethods page.
5. In the **Client List**, click the client ID for the test client in which you want to paste the copied documents.
6. Click **Test Client View**.
7. Click the **Outgoing Documents** tab.

Note: If the test client is not connected, the **Outgoing Documents** tab displays the message "Test client is not connected. Please reconnect the test client to

get the information." For information about reconnecting a test client, see [“Reconnecting Test Clients” on page 227](#).

8. Click **Add Document**.
9. Click the **Paste Documents** tab.
10. Next to **Where to Add**, select an option to indicate whether you want to insert the documents at the beginning or end of the outgoing document list.
11. Click **Paste**.

If the test client does not have can-publish permissions for one or more of the documents in the file, My webMethods displays the following message:

Documents pasted. However, some documents were not pasted because their document type didn't have can-publish permission.

Adding Comments to Documents

To help you identify documents in the **Outgoing Documents** tab and to assist you when examining documents received by a test client, you can add a comment to a document listed on the **Outgoing Documents** tab. For example, the **Outgoing Documents** tab might list several documents of the same type. You can add comments to each document to allow you to quickly differentiate the documents.

Note: When you save documents to file, comments are not saved with the documents.

To add a comment to a document

1. In My webMethods: **Messaging > Broker Servers > Clients**.
2. In the **Client List**, click the client ID of the test client for which you want to add comments to documents in to the outgoing documents list. If the test client does not appear in the list, use the **Search** tab to locate it.
3. On the Client Details page, click **Test Client View** in the Client Details Information view. My webMethods displays the Test Client Information view.
4. On the Test Client Information page, click the **Outgoing Documents** tab.
5. In the **Document Type** column, click the document type for which you want to add a comment. My webMethods opens the Document Details page.
6. In the **Document Details** dialog, enter the comment you want to add to the document in the **Comment** field.
7. Click **OK**. My webMethods displays the comment you added in the **Comments** column next to the document type.

Publishing Documents with a Test Client

You can publish documents with a test client to verify that client subscriptions and subscription filters work and to track a document as it flows through the Broker or between remote Brokers and gateways. When you publish documents with a test client, you can:

- Publish one or more documents on the outgoing documents list.
- Publish the same group of documents multiple times at specific intervals.
- Deliver documents to a specific client.

Before Publishing Documents with a Test Client

Keep the following points in mind before publishing documents with a test client:

- You can publish any documents listed on the **Outgoing Documents** tab.
- The test client publishes the documents in the same order in which the documents appear on the **Outgoing Documents** tab. Use the buttons in the **Move Up** and **Move Down** columns to change the order of the documents.
- A test client can publish only one set of documents at a time. If you want to use a test client to publish a different set of documents and the test client is still publishing an earlier set of documents, you must first stop publishing the old set. For information about stopping publishing by a test client, see [“Stopping Publishing by a Test Client” on page 222](#).
- You can only publish documents with a test client if the test client is connected and the test client queue is unlocked. For information about reconnecting a test client, see [“Reconnecting Test Clients” on page 227](#). For information about unlocking a client queue, see [“Unlocking a Client Queue” on page 234](#).

Publishing or Delivering Documents with a Test Client

Use the following procedure to publish or deliver documents with a test client.

To publish or deliver documents from a test client

1. In My webMethods: **Messaging > Broker Servers > Clients**.
2. In the **Client List**, select the test client for which you want to publish documents. If the test client does not appear in the list, use the **Search** tab to locate it.
3. On the Client Details page, in the Client Details Information view, click **Test Client View**.
4. On the Test Client Information page, click the **Outgoing Documents** tab.
5. Select the check box next to each document that you want to publish.

Note: Make sure the documents in the outgoing documents list appear in the order in which you want to publish the documents. Use the arrows in the **Move Up** and **Move Down** columns to reorder the documents, if necessary.

- Click **Publish**. My webMethods displays the Test Client - Publish Document dialog.
- In the Test Client - Publish Document dialog, click one of the following tabs to select the publish mode:

Click...	To...
Publish Documents	Publish the documents. Broker distributes each document to all of the clients that subscribe to that document type.
Deliver Documents	Deliver the selected documents to a specific client.

- If you selected **Deliver Documents** in step 7, next to the **Target Broker** field, select an option to indicate whether you want to deliver the selected documents to a client in the current Broker or in another Broker. Specify the client ID for the client to which you want to deliver the document.
- In the **Number of Times to Publish** field, specify the number of times you want Broker to publish the set of selected documents. The default is 1.
- In the **Delay between Publishing** field, specify the number of milliseconds the Broker should wait between publishing each set of documents. The default is 0 milliseconds.
- Click **Publish**.

You can trace published documents in topology view to monitor how documents flow between clients. For more information about tracing documents, see [“About Tracing Documents in Broker Topology View”](#) on page 263.

Stopping Publishing by a Test Client

At times, you might want to stop a test client before it completes publishing of all selected documents. For example, you might want to stop publishing if you notice that Broker is not routing any of the published documents to the correct subscribers. You might also want to stop publishing by a test client if you want to use the test client to publish another set of documents. A test client can only publish one set of documents at a time.

To stop publishing by a test client

- In My webMethods: **Messaging > Broker Servers > Clients**.
- In the **Client List**, select the test client for which you want to stop document publishing. If the test client does not appear in the list, use the **Search** tab to locate it.

3. On the Client Details page, in the Client Details Information view, click **Test Client View**.
4. On the Test Client Details page, click the **Outgoing Documents** tab.
5. Click **Stop Publish**.

Retrieving Documents for a Test Client

Broker places documents that match a test client's subscriptions in the test client queue. You use the Incoming Documents tab to retrieve and view these documents. By retrieving documents, you request documents from the client queue that Broker maintains for the test client. You can limit the number of documents retrieved with each request. By default, Broker retrieves up to 10 documents with each request. Broker stores the retrieved documents in the test client's memory cache.

Note: You can retrieve documents for a test client only if the client queue is unlocked. For information about unlocking a client queue, see [“Unlocking a Client Queue” on page 234](#).

To retrieve documents for a test client

1. In My webMethods: **Messaging > Broker Servers > Clients**.
2. In the **Client List**, select the test client for which you want to retrieve documents. If the test client does not appear in the list, use the **Search** tab to locate it.
3. On the Client Details page, in the Client Details Information view, click **Test Client View**.
4. On the Test Client Details page, click the **Incoming Documents** tab.
5. In the **Number of Documents to Retrieve** field, specify the maximum number of documents that you want to retrieve for this test client at one time.

Broker will retrieve up to the specified number. For example, if you specify five documents, but the client queue contains only four documents, Broker retrieves five documents. The default is 10 documents.

6. Click **Retrieve Documents**.
7. Take one or more of the following actions if you want to continue working with retrieved documents:

To...	Do this...
Retrieve more documents for the test client	Click Retrieve Documents .

To...	Do this...
Clear the list of incoming documents	Click Clear Documents .
Save retrieved documents to a file	Select the check box next to the documents you want to save and click Save to File .
View document details, including document content and any comments saved with the published document	Click the document name.

Clearing the Incoming Documents List

You can clear the list of documents retrieved for a test client at any time. Broker clears the list of incoming documents automatically when you log out of My webMethods or your session expires. However, you might find it easier to work with a test client if you clear the list of retrieved documents periodically.

Clearing the incoming documents list deletes the documents from the test client's cache. If you want to examine the documents later or reuse the documents with a test client, save the documents to a file before you clear the list. For information about saving documents to a file, see [“Saving Outgoing or Incoming Documents” on page 225](#).

Clearing the incoming documents list is different from clearing the test client queue. Clearing the incoming documents list removes documents already retrieved by the test client (and stored in the test client's memory cache). Clearing the test client queue removes documents that have not yet been retrieved by the test client. For more information about clearing a test client queue, see [“Clearing the Test Client Queue” on page 225](#).

Note: Clearing the incoming documents list removes all of the documents in the list.

To clear the incoming documents list for a test client

1. In My webMethods: **Messaging > Broker Servers > Clients**.
2. In the **Client List**, select the client ID for the test client for which you want to clear the incoming documents list. If the test client does not appear in the list, use the **Search** tab to locate it.
3. On the Client Details page, click **Test Client View** in the Client Details Information view.
4. On the Test Client Details page, click the **Incoming Documents** tab.

5. Click **Clear Documents**. My webMethods removes all the documents from the **Incoming Documents** tab.

Clearing the Test Client Queue

A test client queue contains all of the published and delivered documents that match the registered subscriptions for the test client. Documents remain in the test client's queue until you retrieve them using the **Incoming Documents** tab. However, you might want to remove documents from the test client's queue without retrieving them. For example, other clients, including other test clients, might have published documents that you are not interested in examining or retrieving.

To clear a test client queue

1. In My webMethods: **Messaging > Broker Servers > Clients**.
2. In the **Client List**, select the test client for which you want to empty the queue. If the test client does not appear in the list, use the **Search** tab to locate it.
3. On the Client Details page, click **Test Client View** in the Client Details Information view.
4. On the **Configuration** tab, click **Clear Queue**. My webMethods deletes all the documents in the test client queue. Notice that the **Queue Length** value is now 0.

Saving Outgoing or Incoming Documents

You can save any of the documents on the **Outgoing Documents** tab or the **Incoming Documents** tab to a file. By saving documents to a file, you can:

- Use the same set of documents with multiple test clients.
- Use documents across multiple sessions. When you log out of My webMethods or your session expires, Broker deletes the contents of the **Outgoing Documents** tab and the **Incoming Documents** tab.
- Add or modify content of the documents by editing the XML file that contains the saved documents. You can then add the populated documents to the **Outgoing Documents** tab for a test client.

Note: When saving a document that contains a large amount of binary data to an XML file, My webMethods encodes the binary data using base 64 encoding. Consequently, My webMethods might take a long time to save a document to an XML file.

To save documents in the outgoing or incoming documents list

1. In My webMethods: **Messaging > Broker Servers > Clients**.

2. In the **Client List**, click the client ID of the test client for which you want to save the outgoing or incoming documents list. If the test client does not appear in the list, use the **Search** tab to locate it.
3. On the Client Details page, click **Test Client View** in the Client Details Information view.
4. On the Test Client Details page, do one of the following:
 - To save documents on the outgoing documents list, click the **Outgoing Documents** tab.
 - To save documents in the incoming documents list, click the **Incoming Documents** tab.
5. Select the check box next to the documents that you want to save to a file.
6. Click **Save to File**. My webMethods displays the Export Documents dialog, which identifies the documents that will be saved to a file.
7. Click **Save to File**. The Internet browser prompts you to open the file or save the file to disk.
8. Select the option to save the file to disk. Enter a name and location for the exported documents and save the file.
9. On the Export Documents page, click **Back** to return to the Test Client Details page.

Disconnecting a Test Client

You can disconnect any active test client on the Broker. Before you disconnect a test client, consider the following points:

- When a test client is disconnected, you cannot modify subscriptions, publish documents, or retrieve documents for the test client. In fact, My webMethods deletes the contents of the outgoing documents list and the incoming documents list when you disconnect the test client. (My webMethods saves the outgoing and incoming documents lists in its memory cache.)
- If you disconnect a test client with a destroy on disconnect lifecycle, Broker deletes the test client.
- If a test client with an explicit destroy lifecycle is disconnected, Broker continues to place documents to which the test client subscribes in the test client's queue. You can retrieve the documents after you reconnect the test client.
- If a test client is in use by another user, you cannot disconnect the test client in test client view. Instead, switch to Broker client view and then disconnect the active session. For more information about disconnecting client sessions, see [“About Disconnecting Sessions” on page 204](#).
- My webMethods disconnects a client automatically when you log out of My webMethods and when your My webMethods session expires due to inactivity.

To disconnect a test client

1. In My webMethods: **Messaging > Broker Servers > Clients**.
2. In the **Client List**, click the client ID of the test client that you want to disconnect. If the test client does not appear in the list, use the **Search** tab to locate it.
3. On the Client Details page, in the Client Details Information view, click **Test Client View**. My webMethods displays the Test Client Information view.
4. On the **Configuration** tab, click **Disconnect**.
5. Click **Close** to return to the Clients page.

Reconnecting Test Clients

If you want to work with a test client that is currently disconnected, you first need to reconnect the test client. On the Test Client Details page, the **Subscriptions** tab, **Outgoing Documents** tab, and **Incoming Documents** tab display the following message if the test client is disconnected:

```
Test client is not connected. Please reconnect the test client to get the information.
```

You can reconnect test clients with an explicit destroy lifecycle only. When a test client with a destroy on disconnect lifecycle is disconnected, Broker deletes the test client.

Reconnecting a Test Client that Uses Basic Authentication

Use the following procedure to reconnect a test client that uses basic authentication.

To reconnect a test client that uses basic authentication

1. In My webMethods: **Messaging > Broker Servers > Clients**.
2. In the **Client List**, click the client ID for the test client that you want to reconnect. If the client does not appear in the list, use the **Search** tab to locate it.
3. On the Client Details page, in the Client Details Information view, click **Test Client View**.
4. On the **Configuration** tab, click **Reconnect**.
5. In the Test Client Authentication options, select **Basic**.
6. Type the basic authentication user name in the **Username** field and the password in the **Password** field.
7. If you want to specify the SSL and encryption settings along with basic authentication, perform the following:

- a. In the **SSL Truststore** field, specify the absolute path and file name of the truststore file containing the trusted root for the SSL certificate stored in the specified keystore.
 - b. In the **SSL Truststore Type** field, select the file type of the truststore certificate file.
8. Click **Reconnect**.

Reconnecting a Test Client that Uses SSL Authentication

Use the following procedure to reconnect a test client that uses SSL authentication.

To reconnect a test client that uses SSL authentication

1. In My webMethods: **Messaging > Broker Servers > Clients**.
2. In the **Client List**, click the client ID for the test client that you want to reconnect. If the client does not appear in the list, use the **Search** tab to locate it.
3. On the Client Details page, in the Client Details Information view, click **Test Client View**.
4. On the **Configuration** tab, click **Reconnect**.
5. On the Test Client Reconnect page, select **SSL**.
6. Perform the following:
 - a. In the **SSL Keystore** field, select the keystore file containing the SSL certificate to be used by the test client to join the client group.
 - b. In the **Password** field, type the password to the keystore file.
 - c. Click **Get User Names** to retrieve a list of distinguished names (DNs) from the keystore.
 - d. In the **User Name** list, select the user DN for the test client.
 - e. In the **SSL Truststore** field, specify the absolute path and file name of the truststore file containing the trusted root for the SSL certificate stored in the specified keystore. In the **SSL Truststore Type** field, select the file type of the truststore certificate file.
7. Click **Reconnect**.

Reconnecting a Test Client that Uses One-way SSL Authentication

Use the following procedure to reconnect a test client that uses one-way SSL authentication.

To reconnect a test client that uses SSL authentication

1. In My webMethods: **Messaging > Broker Servers > Clients**.

2. In the **Client List**, click the client ID for the test client that you want to reconnect. If the client does not appear in the list, use the **Search** tab to locate it.
3. On the Client Details page, in the Client Details Information view, click **Test Client View**.
4. On the **Configuration** tab, click **Reconnect**.
5. In the Test Client Reconnect dialog, select **None**.
6. Perform the following:
 - a. In the **SSL Truststore** field, select the truststore file containing the trusted root for the SSL certificate.
 - b. In the **SSL Truststore Type** field, select the file type of the truststore certificate file.
7. Click **Reconnect**.

Deleting a Test Client

You can delete a test client the same way in which you delete any client on the Broker. Broker continues to deliver documents to the test client as long as the test client exists. If you no longer need a test client, you should delete it. For more information about deleting clients, see [“Deleting Clients” on page 206](#).

Note: If a test client has a destroy on disconnect lifecycle, Broker deletes the test client when you log out of My webMethods or your session expires.

10 Managing Client Queues

■ Overview	232
■ What Is a Client Queue?	232
■ Viewing Client Queue Statistics	233
■ Locking a Client Queue	233
■ Unlocking a Client Queue	234
■ Browsing a Client Queue	235
■ Saving Client Queue Documents to a File	237
■ Inserting Documents into a Client Queue	238
■ Deleting Documents from a Client Queue	241
■ Clearing Client Queues	242
■ Proactively Deleting Documents from a Client Queue	243

Overview

This chapter describes how to view the list of documents in a queue and examine the content of the documents themselves. This chapter also explains how to delete documents from a queue, move documents from one queue to another, and insert documents into a queue.

What Is a Client Queue?

A *client queue* contains the published documents to which a client subscribes. Each client has one queue, which is part of its client-state object. A document remains in the queue until the client retrieves it (and acknowledges that it has retrieved the document successfully) or until the document expires (as determined by the [Time to Live](#) document type parameter). To reduce memory usage, volatile documents that have expired can be deleted at regular intervals, based on the size of the queue, from the client queues and forward queues before the client tries to retrieve them. For more information, see [“Proactively Deleting Documents from a Client Queue”](#) on page 243.

Each queue has a storage type that determines whether documents in the queue are saved in local memory (volatile storage) or are saved to disk (guaranteed storage). The storage type for a queue also dictates the lifecycle of the client. The lifecycle determines whether the Broker maintains state information for a client, including queue contents, after the client disconnects.

The client group to which a client belongs determines the queue storage type and consequently, the client lifecycle. For more information about setting queue storage type for client groups, see [“Client Queue Storage Type”](#) on page 174.

Note: The final determination of how a document is stored on Broker (volatile or guaranteed storage) is determined by the client group storage type and the document storage type. For more information, see [“Client Group Storage Types and Document Type Storage Types”](#) on page 161.

Client Queue Ownership

The client queue ownership is decided based on how the client queue is created.

<u>Client Queue Created By</u>	<u>Client Queue Owner</u>	<u>Client Queue Connection</u>
Broker Administrator (Using the JMS Admin API or with My webMethods)	User with credentials matching the credentials (SSL DN or basic authentication user)	Only the owner of the client queue can connect to the client queue.

<u>Client Queue Created By</u>	<u>Client Queue Owner</u>	<u>Client Queue Connection</u>
	name) specified while creating the client queue.	
Non Administrator client	Anonymous (even if the client queue was created by an SSL client or a basic authenticated client)	Anyone can connect to the client queue.

Viewing Client Queue Statistics

My webMethods maintains statistics for a queue, including:

- The total number of documents in the queue, including the number of documents available for delivery and the number of unacknowledged documents in the queue
- The last time a document was placed in the queue
- The last time a client retrieved a document from the queue
- The size of the queue

You might want to take corrective or preventive action based on the queue statistics. For example, if the statistics indicate that the queue contains a large number of documents and it is been a long time since a client program retrieved a document, you might want to verify that the client program is running properly.

For more information about viewing queue and other client statistics, see [“Viewing Client Statistics” on page 196](#).

Locking a Client Queue

Before you can modify the contents of a queue, you must lock the queue. (If you just want to browse the contents of a queue, you do not need to lock it.) Keep the following points in mind when working with locked client queues:

- You can have multiple queues locked at one time.
- You must have administrative privileges to lock a queue.
- Other users, including the queue owner, cannot clear the contents of a queue or delete the queue while it is locked.
- Broker continues to place documents to which the client subscribes in a locked queue.
- A client cannot retrieve documents from a locked queue.

- While the queue is locked, Broker processes document acknowledgements for documents that the client retrieved before you locked the queue.
- When you lock a queue, My webMethods creates a volatile client to own the queue lock. This client belongs to the admin client group and has an application name that matches the following format: Messaging Management Queue Browser on *ClientID (username)* Click the client ID in the Client List to view details about this client.
- The queue remains locked until you specifically unlock the queue, your My webMethods session ends, or the volatile client created to hold the queue lock is destroyed.

To lock a client queue

1. In My webMethods: **Messaging > Broker Servers > Clients**.
2. In the **Client List**, click the client ID of the client whose queue you want to lock. If the client does not appear in the list, use the **Search** tab to locate it.
3. On the Client Details page, click the **Configuration** tab.
4. Click **Lock Queue**.

Unlocking a Client Queue

When you finish working with a queue, unlock the queue so that queue activity, such as document retrieval, can resume. If you log out of My webMethods or your session expires before you unlock a queue, My webMethods unlocks the queue automatically. Only the user who locked the queue and a My webMethods administrator can unlock a queue.

To unlock a client queue

1. In My webMethods: **Messaging > Broker Servers > Clients**.
2. In the **Client List**, click the client ID of the client whose queue you want to unlock. If the client does not appear in the list, use the **Search** tab to locate it.
3. On the Client Details page, click the **Configuration** tab.
4. Click **Unlock Queue**.

Viewing Lock Information for a Queue

My webMethods maintains and displays information about the lock status of a queue. Specifically, on the **Configuration** tab for a client, you can view the client ID for the client that locked the queue, the session during which the queue was locked, and the time at which the queue was locked. For more information about viewing client configuration information, including queue lock status, see [“Viewing Client Configuration Information” on page 193](#).

Browsing a Client Queue

If you want to view the contents of a client's queue, you can browse the queue. When you browse a queue, you retrieve a list of the documents in the queue. For each document, My webMethods specifies the acknowledgement status of the document, the document size, and the document's sequence number.

For example, you might want to browse a queue to verify that a sales order was routed to the proper client. Or, if a client continually crashes when attempting to process a document from its queue, you might want to browse the queue to obtain more information about the document and possibly delete the document. Or, you might want to browse the dead letter queue to see which documents the Broker routed to this queue.

You can browse a locked or an unlocked queue.

To browse client queue contents

1. In My webMethods: **Messaging > Broker Servers > Clients**.
2. In the **Client List**, click the client ID of the client whose queue you want to browse. If the client does not appear in the list, use the **Search** tab to locate it.
3. On the Client Details page, click the **Browse Queue** tab.
4. Provide the following information:

In this field...	Specify...
Start Position	The location in the queue where you want to begin browsing. The default is 0 (the first document in the queue).
Number of Documents	The number of documents that you want to view at one time. The default is 10 documents.
Timeout	The length of time (in seconds) that My webMethods waits for the set of documents to be returned. When the specified time elapses, a time-out exception is thrown. The default is 30 seconds.

5. If you want to create (or edit filters) while viewing the queue content, click **Yes** next to **Document Type Filters**. **Yes** indicates that filters will be used to limit the number of documents for browsing. For more information about using filters when browsing a queue, see [“Filtering Queue Contents while Browsing” on page 236](#).

6. Click **Start**. My webMethods retrieves the first set of documents.

The **Browse Queue** tab displays the following information for each document.

Field	Description
Document Type	The document type for the document. Click the document type name to view document content.
Ack Status	Indicates whether or not the document has been retrieved by the client. <ul style="list-style-type: none"> ■ Available.The document is available for retrieval by the client program. ■ Unacked.The client program retrieved the document but has not yet acknowledged the document.
Sequence Number	Receipt sequence number for the document. The client assigns this number to the document when it is placed in the queue. The sequence number is unique within the client queue.
Size	Size of the document measured in bytes.

7. If necessary, do one or more of the following to view more documents in the queue or view information about a specific document in the queue:
- Click **Next Set** to retrieve the next group of documents in the queue.
 - Click the document type name to view document and document envelope content.

Note: Broker maintains a redelivery count for each document in a queue. A redelivery count of -1 indicates that the document has not been delivered to the client. A redelivery count of 0 indicates that the document has been delivered once. A redelivery count of 1 indicates that the document has been redelivered once.

For information about deleting documents in a queue, see [“Deleting Documents from a Client Queue” on page 241](#). For information about saving documents in a queue to file, see [“Saving Client Queue Documents to a File” on page 237](#).

Filtering Queue Contents while Browsing

You can limit the documents displayed when you browse a queue by creating document type filters. Each filter specifies the following:

- The document type
- An expression

My webMethods displays only those documents that match the specified document type and whose content satisfies the criteria specified by the expression.

You can apply multiple filters when browsing a queue. However, you can only create one filter for each document type.

Tip: You can sort the queue browsing results by document type by clicking the **Document Type** column name.

To filter client queue contents

1. In My webMethods: **Messaging > Broker Servers > Clients**.
2. In the **Client List**, click the client ID of the client whose queue you want to browse. If the client does not appear in the list, use the **Search** tab to locate it.
3. On the Client Details page, click the **Browse Queue** tab.
4. Click **Yes** next to **Document Type Filters**.
5. On the Document Type Filters page, on the **Filters** tab, click **Add Filter**.
6. In the **Document Type Name** field, specify the fully qualified name of the document type you want to filter while browsing the queue.
7. In the **Filter Expression** field, type the expression that you want to use to filter the contents of the queue while browsing. For information about the syntax used for filters, see the *webMethods Broker Client Java API Programmer's Guide*.
8. Click **OK**. My webMethods displays the Document Type Filters page.
9. Do one or more of the following:
 - Click **Close** to return to the **Browse Queue** tab on the Client Details page.
 - Click **Add Filter** to add a filter for a different document type.
 - Click **Reset Filters** to clear all filters.
 - Click the document type name to view details about the document type.

Saving Client Queue Documents to a File

You can save documents located on the **Browse Queue** tab to a file. You might want to save one or more of these documents for the following reasons:

- To maintain copies of one or more documents in the queue before you delete the document or clear the queue.

- To use the document for testing purposes later. For example, you might want to save the document and publish it with a test client.
- To modify the document and then reinsert it into the queue.

Note: When saving a document that contains a large amount of binary data to an XML file, My webMethods encodes the binary data using base 64 encoding. Consequently, it might take a long time to save a document to an XML file.

To save client queue documents to a file

1. Browse the queue to locate the documents you want to save. For information about browsing queue contents, see [“Browsing a Client Queue” on page 235](#).
2. On the **Browse Queue** tab, select the check box next to the documents that you want to save to a file.
3. Click **Save to File**. My webMethods displays the Export Documents page, which identifies the documents that will be saved to a file.
4. Click **Save to File**. The Internet browser prompts you to open the file or save the file to disk.
5. Select the option to save the file to disk. Enter a name and location for the exported documents and save the file.
6. On the Export Documents page, click **Back** to return to the Client Details page.

Inserting Documents into a Client Queue

You can add documents to a queue to make the documents available for processing by the client. For example, you might want to take a document from the dead letter queue and place it in the queue of the client that you want to process the document. Or, you might want to copy a document from a client queue experiencing a connection problem and place it in the queue of a client that you want to process the document. You might also want to reorder the documents in a queue by inserting documents at a new location and deleting documents at the old location.

To insert documents into a queue, you need to complete the following tasks:

- Build a list of documents to insert into the queue. You can either load documents from a file or copy documents from another client queue.
- Insert the documents at the beginning or end of the queue.

Keep the following points in mind when inserting documents to a queue:

- You can build a list of documents for locked or unlocked queue. However, you must lock a queue before you insert documents into the queue.

- You can only insert documents that the client group to which the client belongs has can-subscribe permissions.
- My webMethods maintains the list of documents to insert into the queue for the duration of your current session only. When you log out of My webMethods, your session is disconnected, or your session expires due to inactivity, My webMethods deletes the list of documents to insert into the queue.

The following sections provide more information about building a list of documents and inserting documents into a queue.

Loading Documents from a File

You can build a list of documents to insert into a queue by loading documents from a file. For example, you might want to insert a document that you edited after you saved it to file.

To load documents from a file

1. In My webMethods: **Messaging > Broker Servers > Clients**.
2. In the **Client List**, click the client ID of the client for which you want to insert documents. If the client does not appear in the list, use the **Search** tab to locate it.
3. On the Client Details page, click the **Browse Queue** tab.
4. Click **Insert Documents**.
5. Click **Add Document**. My webMethods displays the Add Documents page.
6. On the **Load Documents** tab, in the **File Name** field, enter the path to the file containing the documents you want to add to the queue. Click the **Browse** button to navigate to and select the file.
7. Next to **Where to Append**, select an option to indicate whether you want to add the documents at the beginning or end of the insert documents list.
8. Click **Load**. If the client group to which the client belongs does not have can-subscribe permissions for one or more of the documents in the file, My webMethods displays the following message:

```
Documents loaded. However, some documents were not loaded because
their document type didn't have can-subscribe permission.
```
9. Proceed to [“Inserting Documents” on page 240](#) for information about inserting documents into the queue.

Note: Use the buttons in the **Move Up** and **Move Down** columns to reorder the documents in the Insert Documents view.

Copying and Pasting Documents

You can add existing documents to a queue by copying and pasting the documents from another client queue. For example, you might want to copy a document from the dead letter queue into the queue of the client that you want to process the document.

To add documents to a queue by copying and pasting

1. In My webMethods: **Messaging > Broker Servers > Clients**.
2. Copy the documents you want to paste into the queue. You can copy documents from the following locations:
 - The **Browse Queue** tab of any client
 - The **Outgoing Documents** tab of a test client
 - The **Incoming Documents** tab of a test client
3. Click **Clients** at the top of the current My webMethods page.
4. In the **Client List**, click the client ID for the client in whose queue you want to paste the copied documents.
5. On the Client Details page, click the **Browse Queue** tab.
6. Click **Insert Documents**.
7. Click **Add Document**.
8. Click the **Paste Documents** tab.
9. Next to **Where to Paste**, select an option to indicate whether you want to insert the documents at the beginning or end of the insert documents list.
10. Click **Paste**. If the client group to which the client belongs does not have can-subscribe permissions for one or more of the documents in the file, My webMethods displays the following message:

Documents pasted. However, some documents were not pasted because their document type didn't have can-subscribe permission.
11. Proceed to [“Inserting Documents” on page 240](#) for information about inserting documents into the queue.

Inserting Documents

After you build the list of documents to add to queue, you can insert the documents into the queue. Keep the following points in mind before inserting documents into a queue:

- You must lock the queue before inserting documents.
- You can insert any documents listed on the Insert Documents view of the **Browse Queue** tab.

- You can insert documents at the beginning or end of the queue only.
- Broker inserts the documents into the queue in the same order in which the documents appear on the Insert Documents view of the **Browse Queue** tab. Use the buttons in the **Move Up** and **Move Down** columns to change the order of the documents.
- Make sure to insert documents to which the client subscribes only. Click the **Subscriptions** tab to view a list of document subscriptions.

To insert documents into a client queue

1. In My webMethods: **Messaging > Broker Servers > Clients**.
2. In the **Client List**, click the client ID of the client to which you want to add documents to the queue. If the client does not appear in the list, use the **Search** tab to locate it.
3. Lock the queue. For information about locking a queue, see [“Locking a Client Queue” on page 233](#).
4. On the Client Details page, click the **Browse Queue** tab.
5. Click **Insert Documents**.
6. Select the check box next to each document you want to insert into the queue.
7. Click **Insert to Queue**. My webMethods displays the Insert Documents to Queue Page.
8. On the **Insert Documents** tab, next to **Where to Insert**, select the option that indicates where in the queue you want to insert the documents.
9. Click **Insert**.

Deleting Documents from a Client Queue

You can delete any documents in a queue. You might want to delete documents from a queue if:

- A document causes the application that processes the document to behave unexpectedly.
- Documents published by a test client or invalid documents published by a misbehaving client.
- The client's subscriptions changed and you want to delete the already enqueued documents that the client no longer subscribes to.

You can delete documents from a client queue regardless of the documents acknowledgement status.

Tip: You can save a document before deleting it. For information about saving documents in a queue, see [“Saving Client Queue Documents to a File” on page 237](#).

To delete documents from a client queue

1. In My webMethods: **Messaging > Broker Servers > Clients**.
2. In the **Client List**, click the client ID of the client for which you want to delete documents in its queue. If the client does not appear in the list, use the **Search** tab to locate it.
3. Lock the queue. For information about locking a queue, see [“Locking a Client Queue” on page 233](#).
4. On the Client Details page, click the **Browse Queue** tab.
5. Browse the queue to locate the documents you want to delete. For information about browsing queue contents, see [“Browsing a Client Queue” on page 235](#).
6. Select the check box next to each document that you want to delete from the queue.
7. Click **Delete**. My webMethods removes the documents from the client queue.

Clearing Client Queues

You can remove all the documents from a queue by clearing it. You might want to clear a queue in the following situations:

- You are testing the client and want to resume testing with an empty client queue.
- A client published invalid documents and it is more efficient to clear the queue than locate and delete the individual invalid documents.
- A subscriber connection no longer exists. After moving the queue contents to another client's queue for processing, you can clear the documents from the original queue.

Before Clearing a Client Queue

Keep the following points in mind before clearing a queue:

- You can only clear an unlocked queue.
- Only the owner of a queue (that is, the client that created the queue) or a My webMethods user with administrative privileges can clear a queue.

Clearing a Client Queue

Use the following procedure to clear a client queue.

To clear a client queue

1. In My webMethods: **Messaging > Broker Servers > Clients**.

2. In the **Client List**, click the client ID of the client for which you want to clear the queue. If the client does not appear in the list, use the **Search** tab to locate it.
3. On the Client Details page, click the **Statistics** tab.
4. Under **Client Queue**, next to the **Length** field, click **Clear Queue**.

Proactively Deleting Documents from a Client Queue

Expired documents in the client queues and forward queues; that is, the documents that have completed their [Time to Live](#), are deleted only when the client tries to retrieve them. In cases where a client retrieves documents infrequently, this can lead to overloaded queues or memory overflow, which can further affect operations of other Brokers in the topology.

To reduce the memory usage considerably, you can proactively delete the expired volatile documents from the client queues and forward queues even before the client tries to retrieve them. Expired documents are deleted whenever the total memory used by the volatile documents exceeds the specified threshold limit.

You can proactively delete volatile documents from a queue by setting the `queue-cleanup-enable` and the `queue-cleanup-threshold` parameters in the `awbroker.cfg` file of each Broker Server instance.

To proactively delete volatile documents from a client queue

1. Stop Broker Server.
2. On the machine where Broker Server is installed, locate the Broker Server's configuration file (`awbroker.cfg`) and make a backup copy.
3. Open the configuration file with a text editor.
4. Locate the `queue-cleanup-enable` and the `queue-cleanup-threshold` parameters. If your configuration file does not include them, add them to the file, and set the values. For detailed descriptions of the configuration parameters, see “[webMethods Broker Server Configuration Parameters](#)” on page 621.
5. Save the configuration file.
6. Restart Broker Server.

11 Managing Forwarding Queues

■ Overview	246
■ What Is a Forwarding Queue?	246
■ Navigating to the Forwarding Queue Browser Page	246
■ Locking and Unlocking the Forwarding Queue	248
■ Browsing a Forwarding Queue	249
■ Saving the Forwarding Queue Documents to a File	252
■ Inserting Documents into a Forwarding Queue	253
■ Deleting Documents from a Forwarding Queue	255
■ Proactively Deleting Documents from a Forwarding Queue	256
■ Forwarding Messages to a Remote Cluster or Territory	256

Overview

This chapter describes how to view the list of documents in a forwarding queue for a remote Broker and examine the content of the documents.

What Is a Forwarding Queue?

A *forwarding queue* contains the published documents to which a remote Broker subscribes. For each remote Broker, a forwarding queue resides on the local Broker. A document remains in the queue until the queue forwards the document to the remote Broker and the remote Broker acknowledges that it has retrieved the document successfully.

Navigating to the Forwarding Queue Browser Page

In My webMethods, you must first navigate to the Forwarding Queue Browser page if you want to perform any administrative task on the forwarding queue for a remote Broker. You perform tasks such as locking the queue, unlocking the queue, browsing the queue, inserting documents into the queue, and deleting documents from the queue.

Navigating to the Forwarding Queue for a Remote Broker in Your Territory

Use the following procedure to navigate to the forwarding queue browser page for a remote Broker in your territory.

To navigate to the forwarding queue browser page for a remote Broker in your territory

1. In My webMethods: **Messaging > Broker Territories > Territory Brokers**.
2. In the **Territory Brokers List**, click on a Broker (local Broker) on which you want to browse the forwarding queue for the remote Broker. If your local Broker does not appear in the Territory Brokers list, use the **Search** tab to locate it.
3. On the **Territory Broker Details** page, click on the Forward Queue Length value corresponding to the remote Broker. This action opens the Forwarding Queue Browser Page on My webMethods.

Navigating to the Forwarding Queue for a Remote Broker in Your Cluster

Use the following procedure to navigate to the forwarding queue browser page for a remote Broker in your cluster.

To navigate to the forwarding queue browser page for a remote Broker in your cluster

1. In My webMethods: **Messaging > Broker Clusters > Cluster Brokers**.
2. In the **Cluster Brokers List**, click on a Broker (local Broker) on which you want to browse the forwarding queue for the remote Broker. If your local Broker does not appear in the Cluster Brokers list, use the **Search** tab to locate it.
3. On the **Cluster Broker Details** page, click on the Forward Queue Length value corresponding to the remote Broker in the cluster. This action opens the Forwarding Queue Browser Page on My webMethods.

Navigating to the Forwarding Queue for a Remote Broker in a Remote Territory

Use the following procedure to navigate to the forwarding queue browser page for a remote Broker in a remote territory across a gateway.

To navigate to the forwarding queue browser page for a remote Broker that resides in a remote territory across a gateway

1. In My webMethods: **Messaging > Broker Territories > Territory Gateways**.
2. In the **Territory Gateway List**, click the details icon corresponding to the gateway that connects your local territory (containing your local Broker on which you want to browse the remote Broker's forwarding queue) with the remote territory (containing the remote Broker whose forwarding queue you want to browse). If your gateway does not appear in the list, use the **Search** tab to locate it.
3. On the **Territory Gateway Information** page, click the **Configuration** tab.
4. Click on the **Queue Length** value. This action opens the Forwarding Queue Browser Page. This action opens the Forwarding Queue Browser Page on My webMethods.

Navigating to the Forwarding Queue for a Remote Broker in a Remote Cluster

Use the following procedure to navigate to the forwarding queue browser page for a remote Broker in a remote cluster across a gateway.

To navigate to the forwarding queue browser page for a remote Broker that resides in a remote cluster across a gateway

1. In My webMethods: **Messaging > Broker Clusters > Cluster Gateways**.
2. In the **Cluster Gateway List**, click the details icon corresponding to the gateway that connects your local cluster (containing your local Broker on which you want to browse the remote Broker's forwarding queue) with the remote cluster (containing the remote Broker whose forwarding queue you want to browse). If your gateway does not appear in the list, use the **Search** tab to locate it.
3. On the **Cluster Gateway Information** page, click the **Configuration** tab.
4. In the table corresponding to **Queue Length**, click on the Forward Queue Length value corresponding to the cluster Broker whose forwarding queue you want to browse. This action opens the Forwarding Queue Browser Page on My webMethods.

Locking and Unlocking the Forwarding Queue

Before you can modify the contents of a forwarding queue, you must lock the queue. There is no need to lock the queue if you only want to browse the contents. Keep the following points in mind when working with locked forwarding queues:

- You can have multiple queues locked at one time.
- You must have administrative privileges to lock a queue.
- Nonadministrative users on the local Broker or the remote Broker cannot clear the contents or delete documents from the forwarding queue in the locked state.
- The local Broker continues to publish to the locked forwarding queue, those documents that have been subscribed to by the remote Broker.
- A remote Broker cannot retrieve documents from a locked forwarding queue.
- While the forwarding queue is locked, remote Broker processes document acknowledgements even for those documents that it retrieved before you locked the queue.
- When you lock a queue, My webMethods internally creates a volatile client to own the queue lock. This client belongs to the admin client group.
- The queue remains locked until you specifically unlock the queue, your My webMethods session ends, or the volatile client created to hold the queue lock is destroyed.

When you finish working with a queue, unlock the queue so that queue activity, such as document retrieval, can resume. If you log out of My webMethods or if your session expires before you unlock a queue, My webMethods unlocks the queue automatically. Only the user who locked the queue and a My webMethods administrator can unlock a queue.

Follow these steps to lock and unlock the forwarding queue for a remote Broker:

To lock or unlock the forwarding queue for a remote Broker

1. Navigate to the Forwarding Queue Browser page. For more information, see [“Navigating to the Forwarding Queue Browser Page” on page 246](#).
2. To lock the queue, click the **Lock** button next to the lock icon. Or, click the **Unlock** button next to the lock icon to unlock the queue.

Browsing a Forwarding Queue

To view the contents of a forwarding queue for remote Broker, you browse the queue. When you browse a forwarding queue, you retrieve a list of documents from the queue. For each document, My webMethods displays the acknowledgement status of the document, the document size, and the document sequence number.

For example, you might want to browse a forwarding queue to verify whether a sales order was routed to the proper remote Broker. Or, if a remote Broker continually crashes when attempting to process a document from its forwarding queue, you might want to browse the queue to obtain more information about the document and possibly delete the document.

By using My webMethods you can browse both unlocked, and locked forwarding queues.

To browse the forwarding queue contents for a remote Broker

Important: While browsing the forwarding queue contents for a remote Broker that resides in a remote territory (or remote cluster) across a gateway, keep in mind that you must select the gateway in such a way that the forwarding queue resides on the local territory's (or cluster's) Broker and will belong to the remote territory's (or cluster's) Broker across the gateway.

1. Navigate to the Forwarding Queue Browser page. For more information, see: [“Navigating to the Forwarding Queue Browser Page” on page 246](#).
2. On the **Forwarding Queue Browser** page, specify values for the queue browser settings.

For configuring this field...	Specify...
Start Position	The location in the forwarding queue where you want to begin browsing. The default is 0 (the first document in the queue).
Number of Documents	The number of documents that you want to view at one time.

For configuring this field...	Specify...
-------------------------------	------------

The default is 10 documents.

Timeout	The time (in seconds) that My webMethods waits for the set of documents to be returned. When the specified time elapses, a time-out exception is thrown.
----------------	--

The default is 30 seconds.

- If you want to create (or edit) filters while viewing the forwarding queue content, click **Yes** next to **Document Type Filters**. **Yes** indicates that filters will be used to limit the number of documents for browsing. For more information about using filters when browsing a forwarding queue, see [“Filtering Forwarding Queue Contents while Browsing” on page 251](#).
- Click **Start** to browse the first set of documents.

The Document Types list displays the following information about each document in the queue.

Field	Description
Document Type	The document type for the document. Click the document type name to view document content.
Ack Status	Indicates whether or not the document has been retrieved by the remote Broker. <ul style="list-style-type: none"> ■ Available. The document is available for retrieval by the remote Broker. ■ Unacked. The remote Broker retrieved the document but has not yet acknowledged the document.
Sequence Number	Receipt sequence number for the document. The forwarding queue assigns this number to the document when it is placed in the queue. The sequence number is unique within the forwarding queue.
Size	Size of the document measured in bytes.

- If necessary, do one or more of the following to view more documents in the forwarding queue or view information about a specific document in the forwarding queue:
 - Click **Next Set** to retrieve the next group of documents in the forwarding queue.

- Click the document type name to view document and document envelope content.

For information about saving documents in a queue to file, see [“Saving the Forwarding Queue Documents to a File”](#) on page 252.

Filtering Forwarding Queue Contents while Browsing

You can limit the documents displayed when you browse a forwarding queue by creating document type filters. Each filter specifies the following:

- The document type
- An expression

My webMethods displays only those documents that match the specified document type and whose content satisfies the criteria specified by the expression.

You can apply multiple filters when browsing a queue. However, you can only create one filter for each document type.

Tip: You can sort the forwarding queue browser results by document type by clicking the **Document Type** column name.

Filtering Contents for a Remote Broker in Your Territory or Cluster

Use the following procedure to filter queue contents for a remote Broker in your territory or cluster.

To filter forwarding queue contents for a remote Broker in your territory or cluster

1. Navigate to the Forwarding Queue Browser page. For more information, see [“Navigating to the Forwarding Queue for a Remote Broker in Your Territory”](#) on page 246, or see [“Navigating to the Forwarding Queue for a Remote Broker in Your Cluster”](#) on page 247.
2. On the **Forwarding Queue Browser** page, click **Yes** next to **Document Type Filters**.
3. On the Document Type Filters page, on the **Filters** tab, click **Add Filter**.
4. In the **Document Type Name** field, specify the fully qualified name of the document type you want to filter while browsing the queue.
5. In the **Filter Expression** field, type the expression that you want to use to filter the contents of the queue while browsing. For information about the syntax used for filters, see the *webMethods Broker Client Java API Programmer's Guide* .
6. Click **OK**. My webMethods displays the Document Type Filters page.
7. Do one or more of the following:
 - Click **Add Filter** to add another filter for a different document type.

- Click **Reset Filters** to clear all filters.
 - Click the Edit icon to edit the filter expression.
8. Click **Close** to return to the forwarding queue browser page.

Filtering Contents for a Remote Broker in a Remote Territory or Cluster

Use the following procedure to filter queue contents for a remote Broker that resides in a remote territory or cluster.

To filter forwarding queue contents for a remote Broker that resides in a remote territory or a remote cluster across a gateway

1. Navigate to the Forwarding Queue Browser page. For more information, see [“Navigating to the Forwarding Queue for a Remote Broker in a Remote Territory” on page 247](#), or see [“Navigating to the Forwarding Queue for a Remote Broker in a Remote Cluster” on page 247](#).
2. On the **Forwarding Queue Browser** page, click "Yes" next to **Document Type Filters**.
3. On the Document Type Filters page, on the **Filters** tab, click **Add Filter**.
4. In the **Document Type Name** field, specify the fully qualified name of the document type you want to filter while browsing the queue.
5. In the **Filter Expression** field, type the expression that you want to use to filter the contents of the queue while browsing. For information about the syntax used for filters, see the *webMethods Broker Client Java API Programmer's Guide* .
6. Click **OK**. My webMethods displays the Document Type Filters page.
7. Do one or more of the following:
 - Click **Close** to return to the **Gateway Details** page.
 - Click **Add Filter** to add a filter for a different document type.
 - Click **Reset Filters** to clear all filters.
8. Click the document type name to view details about the document type.

Saving the Forwarding Queue Documents to a File

You can save documents contained in the forwarding queue for one or more of the following reasons:

- To maintain copies of one or more documents in the forwarding queue before you delete the documents or clear the queue.
- To use the documents for testing purposes later. For example, you might want to save the documents and publish it with a test client.
- To modify the documents and then reinsert them into the forwarding queue.

Note: When saving a document that contains a large amount of binary data to an XML file, My webMethods encodes the binary data using base 64 encoding. So, it might take a long time to save a document to an XML file.

To save documents from a queue to a file

1. Browse the queue to locate the documents you want to save. For information about browsing queue contents, see [“Browsing a Forwarding Queue” on page 249](#).
2. On the forwarding queue browser page, select the check box next to the document type list that you want to save to a file.
3. Click **Save to File**. My webMethods displays the Export Documents page, which identifies the documents types that are being saved to the file.
4. The Internet browser prompts you to open the file or save the file to disk. Select the option to save the file to disk. Enter a name and location for the exported documents and save the file.

Inserting Documents into a Forwarding Queue

You insert documents into a forwarding queue for a remote Broker for the following reasons:

- You want to copy a document from a client queue (or forwarding queue) that is experiencing a connection problem and place it in the forwarding queue for another remote Broker.
- You want to reorder the documents in a queue by inserting documents at a new location and deleting documents at the old location.

To insert documents into a queue, you need to complete the following tasks:

- Build a list of documents to insert into the queue. You can either load documents from a file or copy documents from another client queue or forwarding queue.
- Insert the documents into the queue.

Keep the following points in mind when inserting documents to a queue:

- You can build a list of documents for locked or unlocked queue. However, you must lock a queue before you insert documents into it.
- My webMethods maintains the list of documents to insert into the queue for the duration of your current session only. When you log out of My webMethods, or your session gets disconnected, or your session expires due to inactivity, My webMethods loses the list of documents that you want to insert into the queue.

The following sections provide more information about building a list of documents and inserting documents into a queue.

Loading Documents from a File

You can build a list of documents to insert into a queue by loading documents from a file. For example, you might want to insert a document that you edited after you saved it to a file.

To load documents from a file

1. Navigate to the Forwarding Queue Browser page. For more information, see [“Navigating to the Forwarding Queue Browser Page” on page 246](#).
2. Click **Insert Documents**.
3. Click **Add Document**. My webMethods displays the Add Documents page.
4. On the **Load Documents** tab, in the **File Name** field, enter the path to the file containing the documents you want to add to the queue. Click the **Browse** button to navigate to and select the file.
5. Next to **Where to Append**, select an option to indicate whether you want to add the documents at the beginning or end of the insert documents list.
6. Click **Load**.
7. Proceed to [“Inserting Documents” on page 255](#) for information about inserting documents into the queue.

Note: Use the buttons in the **Move Up** and **Move Down** columns to reorder the documents in the Insert Documents view.

Copying and Pasting Documents

You can insert documents into a forwarding queue by copying documents from another forwarding queue, or client queue and then pasting them into this queue.

To add documents to a forwarding queue by copying and pasting

1. Navigate to the Forwarding Queue Browser page. For more information, see [“Navigating to the Forwarding Queue Browser Page” on page 246](#).
2. In the Forwarding Queue Browser page, click **Insert Documents**.
3. Click **Add Document**.
4. Click the **Paste Documents** tab.
5. Next to **Where to Paste**, select an option to indicate whether you want to insert the documents at the beginning or end of the insert documents list.
6. Click **Paste**.

7. Proceed to [“Inserting Documents” on page 255](#) for information about inserting documents into the queue.

Inserting Documents

After you build the list of documents to add to queue, you can insert the documents into the queue. Keep the following points in mind before inserting documents into a queue:

- You must lock the queue before inserting documents. For more information, see [“Locking and Unlocking the Forwarding Queue” on page 248](#).
- You can insert any documents listed on the Insert Documents view of the **Browse Queue** tab.
- When you insert a document in the forwarding queue, you are adding it at the tail end of the queue.
- Broker inserts the documents into the queue in the same order in which the documents appear on the Insert Documents view of the **Browse Queue** tab. You use the buttons in the **Move Up** and **Move Down** columns to change the order of the documents.

To insert documents into a forwarding queue

1. Navigate to the Forwarding Queue Browser page. For more information, see [“Navigating to the Forwarding Queue Browser Page” on page 246](#).
2. In the Forwarding Queue Browser page, click **Insert Documents**.
3. Select the check box for each document you want to insert into the queue.
4. Click **Insert to Queue**.

Deleting Documents from a Forwarding Queue

You can delete any document from a forwarding queue. You might want to delete documents from a queue for the following reasons:

- A document causes the application that processes the document to behave unexpectedly.
- A local Broker forwards invalid documents to the forwarding queue of the remote Broker.
- The remote Broker's subscriptions changed and you want to delete the already enqueued documents that the remote Broker no longer subscribes to.

You can delete documents from a forwarding queue regardless of the document's acknowledgement status.

Tip: You can save a document before deleting it. For information about saving enqueued documents, see [“Saving the Forwarding Queue Documents to a File” on page 252](#).

To delete documents from a forwarding queue

1. Lock the forwarding queue. For more information, see [“Locking and Unlocking the Forwarding Queue” on page 248](#).
2. Browse the queue to locate the documents you want to delete. For more information, see [“Browsing a Forwarding Queue” on page 249](#).
3. Select the check box for each document you want to delete from the queue.
4. Click **Delete**.

Proactively Deleting Documents from a Forwarding Queue

Expired documents in the forwarding queues; that is, documents that have completed their "time to live" period, are deleted only when a remote Broker tries to retrieve them. In cases where a remote Broker does not retrieve documents frequently, this can lead to overloaded queues or memory overflow, which can further affect operations of other Brokers in the topology.

To significantly minimize the memory usage, you can proactively delete the expired volatile documents from the forwarding queues even before the remote Broker tries to retrieve them. Expired documents are deleted whenever the total memory used by the volatile documents exceeds the specified threshold limit.

You can proactively delete volatile documents from a queue by setting the `queue-cleanup-enable` and the `queue-cleanup-threshold` parameters in the `awbroker.cfg` file of each Broker Server instance.

You proactively delete documents from the forwarding queue in the same manner as you would delete them from a client queue. For more information, see [“Proactively Deleting Documents from a Client Queue” on page 243](#).

Forwarding Messages to a Remote Cluster or Territory

Provide the complete destination name if you want to forward the messages delivered to a queue on the local cluster or territory to a remote cluster or territory across the gateway.

For example, if you specify the destination as:

- Q1, Broker will try to deliver the message only to the Q1 queue present on the same Broker.

- `Broker2/Q1`, Broker will try to deliver the message only to the Q1 queue on Broker2 in the same territory or cluster.
- `/Cluster2/Broker2/Q1`, Broker will try to find a gateway that reaches Cluster2, forward the message through the gateway if the forward permissions are configured for the gateway, and deliver the message to the Q1 queue on Broker2.

While forwarding messages to a remote cluster or territory, if the message delivery path involves multiple Broker hops, enable the dead letter queue on all the Brokers so that if any message gets dropped, you can trace which Broker dropped the message.

12 Using Topology View and Document Trace

- Topology View of Brokers 260
- Table View of Trace Information 272
- Topology View of Territories 275

Topology View of Brokers

The Broker topology view provides a graphical representation of a Broker. The topology view shows:

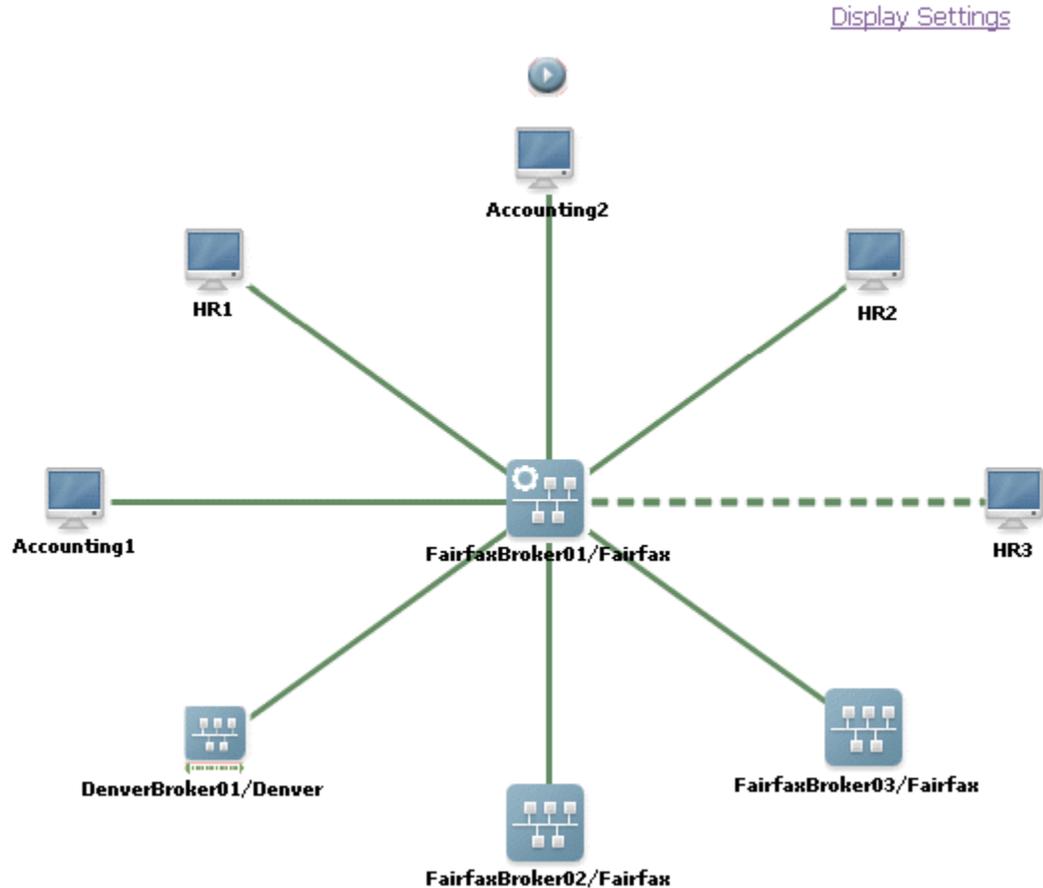
- Clients connected to the current Broker
- Remote Brokers connected to the current Broker
- Gateway Brokers in other territories where there is a gateway between the current Broker and the gateway Brokers shown in the topology view

Additionally, you can trace documents in the topology view to show the documents that are actively flowing from producers to Broker to consumers. For more information, see [“About Tracing Documents in Broker Topology View”](#) on page 263.

Note: The Broker topology view requires that you have the Scalable Vector Graphics (SVG) viewer, which is provided by Adobe, installed on the machine where your browser is running. You can download this viewer from [“http://www.adobe.com/svg/viewer/install/”](http://www.adobe.com/svg/viewer/install/). The SVG viewer with the Firefox Web browser is not supported. If you do not have the SVG viewer installed, when you attempt to use the Broker topology view, Broker prompts you to install it.

Data Displayed in the Broker Topology View

The following shows a sample of the Broker topology view. For more details about the meanings of the icons and lines displayed in the topology view, see the table after the diagram.



Icons and Lines	Description
	The current Broker. This is the Broker for which you are viewing details. You determine the labels that appear in the topology view for a Broker by defining display settings. For more information, see “Setting Your User Preferences for the Display Settings” on page 264.
	Remote Brokers. You determine the labels that appear in the topology view for a remote Broker by defining display settings. For more information, see “Setting Your User Preferences for the Display Settings” on page 264. You can switch to the details for a Broker in the display and view its topology view by clicking on the icon for a Broker.
	Gateway Broker, that is, a Broker in another territory where there is a gateway between the Broker and the current Broker. You determine the labels that appear in the topology view for a Broker by defining display settings. For more

Icons and Lines	Description
	<p>information, see “Setting Your User Preferences for the Display Settings” on page 264.</p> <p>You can switch to the details for a Broker in the display and view its topology view by clicking on the icon for a Broker.</p>
	<p>Client. You determine the labels that appear in the topology view for a client by defining display settings. For more information, see “Setting Your User Preferences for the Display Settings” on page 264.</p>
	<p>Active connection. A solid green line indicates that a connection has been made between the current Broker and the client or remote Broker.</p>
	<p>Disconnected connection. A dotted green line indicates that the connection could not be made between the current Broker and the client or remote Broker. You determine whether the topology view contains disconnected Brokers or clients by defining display settings. For more information, see “Setting Your User Preferences for the Display Settings” on page 264.</p>
	<p>Connection error. A solid red line indicates that a connection cannot be made due to a denied permission.</p>
	<p>The start trace button. Use this button to animate the topology view allowing you to trace documents actively flowing across client connections. For more information, see “About Tracing Documents in Broker Topology View” on page 263.</p>
	<p>The stop trace button. Use this button to stop the document trace in the topology view.</p>

Displaying the Broker Topology View

This section describes how to display the topology view for a Broker.

To define your display settings for the topology view or to narrow down the number of clients and Brokers that appear in the topology, see [“Setting Your User Preferences for the Display Settings”](#) on page 264.

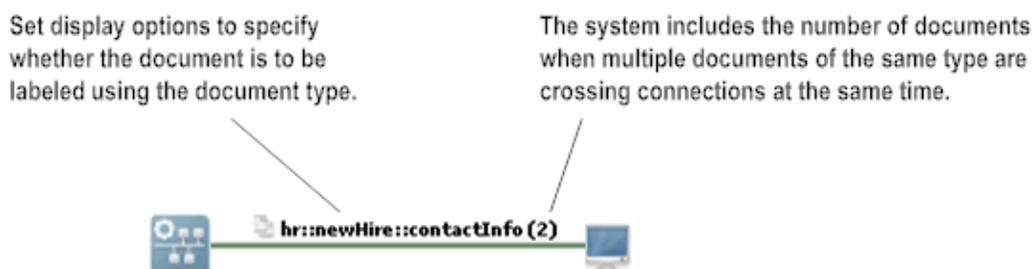
To display the topology view for a Broker

1. In My webMethods: **Messaging > Broker Servers > Brokers.**

2. In the **Brokers List**, click the Broker for which you want to display the topology view. If the Broker does not appear in the list, use the **Search** tab to locate it.
3. On the Broker Details page, click the **Topology** tab.

About Tracing Documents in Broker Topology View

While in the topology view, you can start a trace to view documents that are actively flowing across Broker connections, from producers to Broker to consumers. By default, Broker traces documents of all document types. The following shows a sample of a document being traced.



To define your display settings for the topology view, narrow down the number of clients and Brokers that appear in the topology view, and narrow down the documents that Broker traces, see [“Setting Your User Preferences for the Display Settings” on page 264](#).

Important: Tracing documents in the topology view can consume a lot of system resources, for example, the memory of the My webMethods Server. Use it as a diagnostic tool when needed and do not let it run for long periods of time. It is recommended that you do not trace documents in a production environment when there is a high load of activity.

After you start a document trace, the trace continues as long as you stay on the **Topology** tab of the Broker Detail page until you manually stop the trace. However, if you switch to another page within My webMethods, a timeout timer starts. The timeout value is 30 seconds. To conserve system resources, after 30 seconds elapse, the system automatically stops the document trace. If you switch back to the **Topology** tab of the Broker Detail page before the timeout expires, the system clears the timeout timer and the trace will continue. You cannot configure the timeout value.

Tracing Documents in the Topology View

Use the following procedure to trace documents in Broker's topology view.

To run a document trace in the topology view

1. Display the topology view as described in [“Displaying the Broker Topology View” on page 262](#).

2. If you want to customize the display in the topology view, narrow down the clients and Brokers in the display, and/or narrow down the documents that Broker traces, see [“Setting Your User Preferences for the Display Settings”](#) on page 264.
3. Click the start trace button .

While the trace is running, the start trace button is replaced with the stop trace button . Stop the trace by clicking the stop trace button .

Setting Your User Preferences for the Display Settings

The system saves the display settings for the Broker topology view as user preferences. As a result, the settings you make apply only to your My webMethods user account.

The following table lists the display settings, whether the setting affects the static topology view or both the static view and the topology view while tracing a document, and where you can find more information about how to set the display setting:

Display Setting	Static or Document Tracing	For more information, see...
Labels that appear next to the Brokers and clients	Static and Document Tracing	“Defining How to Display Data in the Broker Topology View” on page 265
Whether to display disconnected Brokers and/or clients	Static and Document Tracing	“Defining How to Display Data in the Broker Topology View” on page 265
Whether to display remote Brokers	Static and Document Tracing	“Defining How to Display Data in the Broker Topology View” on page 265
How often the system is to automatically refresh the data in the topology view	Static and Document Tracing	“Defining How to Display Data in the Broker Topology View” on page 265
Maximum number of nodes to display	Static and Document Tracing	“Defining How to Display Data in the Broker Topology View” on page 265

Display Setting	Static or Document Tracing	For more information, see...
Filters for narrowing down the Broker and client nodes in the display	Static and Document Tracing	“Defining the Brokers and Clients to Display in the Topology View” on page 267
Groups of Brokers or clients to display as a single node in the topology view	Static and Document Tracing	“Defining the Brokers and Clients to Display in the Topology View” on page 267
Documents to trace	Document Tracing Note This setting also affects tracing on the Trace tab and tracing documents in the territory topology view. For more information, see “Table View of Trace Information” on page 272 and “Topology View of Territories” on page 275 .	“Defining the Documents to Include in Traces” on page 271
Whether to label the traced documents	Document Tracing	“Defining How to Display Data in the Broker Topology View” on page 265

Defining How to Display Data in the Broker Topology View

You can define display settings that determine:

- Labels to use in the topology view
- Whether to include disconnected, remote Brokers and clients
- Whether to include remote Brokers
- How often to refresh the display for both a static topology view and during a document trace
- How many total nodes to display in the topology view

To define how to display data in topology view

1. Display the topology view as described in [“Displaying the Broker Topology View” on page 262.](#)
2. Click the **Display Settings** link.
3. On the **Display Options** tab, set the following fields:

Affects these display items	Field	Description
Remote Brokers	Show Broker Name	Select this check box to include the Broker name in the label for a Broker.
	Show Broker Host Name	Select this check box to include the Broker host name in the label for a Broker.
	Show Territory Name	Select this check box to include the territory name in the label for a Broker.
	Show Disconnected Brokers	Select this check box to include Brokers that are disconnected.
	Show Remote Brokers	Select this check box to include remote Brokers.
Clients	Show Client ID	Select this check box to include the client ID in the label for a client.
	Show Client Application Name	Select this check box to include the client application name in the label for a client.
	Show Client Group Name	Select this check box to include the client group name in the label for a client.
	Show Disconnected Clients	Select this check box to include clients that are disconnected.

Affects these display items	Field	Description
Documents	Show Document Type Name	Select this check box if you want traced documents to be labeled with the document type name. The system only displays documents when you start a document trace. For more information, see “About Tracing Documents in Broker Topology View” on page 263.
All items	Poll Interval	Specify how often (in seconds) you want the system to poll for data to update the information in the topology view. This affects both a static topology view and the view during a document trace. The default is 4 seconds.
	Maximum Client Nodes	Specify the maximum number of Broker and client nodes to include in the topology view. The default is 50.

4. Click **Save** to save your changes.
5. Click **OK**.

Defining the Brokers and Clients to Display in the Topology View

By default, the system displays the current Broker and all its connected clients in the topology view.

You can select to display remote Brokers; disconnected, remote Brokers; and disconnected clients using the settings on the **Display Options** tab. For more information, see [“Defining How to Display Data in the Broker Topology View”](#) on page 265.

You can define one or more node groupings to narrow down the number of nodes that the system displays in the topology view. A node grouping defines a group of Brokers or a group of clients. After identifying a node grouping, you can select whether you want to:

- View or hide the node grouping.
- Show each Broker or client using individual nodes for each, or combine all Brokers or clients in the node grouping and display them as a single node.

To define the Brokers and clients to display in the topology view

1. Display the topology view as described in [“Displaying the Broker Topology View” on page 262](#).
2. Click the **Display Settings** link.
3. Click the **Node Grouping** tab.
4. Click **New Group** to define a new node grouping.
5. On the Add Topology Display Group page, set the following fields:

Field	Description
Match Type and Match Name	<p>Use the Match Type and Match Name fields to identify a group of Brokers or a group of clients. The match is case sensitive.</p> <ul style="list-style-type: none"> ■ Match Type. Select the attribute of a Broker or client that you want to use to form the group. For example, use Broker Host Name to create a group of Brokers that have the same or similar host names. <p>To define a group of Brokers, use one of the following:</p> <ul style="list-style-type: none"> ■ BrokerName ■ BrokerHost Name ■ BrokerTerritory Name <p>To define a group of clients, use one of the following:</p> <ul style="list-style-type: none"> ■ Client ID ■ Client Application Name ■ Client Group Name <ul style="list-style-type: none"> ■ Match Name. Specify text to use to find matching items for the group. For example, if you want a group of Brokers that all have host names that begin with "east", specify BrokerHost Name in the Match Type field and <code>east*</code> in the Match Name field. <p>When specifying the Match Name, you can use Broker regular expressions such as the wild card character *. For more information, see the "Using Event Filters" chapter of the <i>webMethods Broker Client Java API Programmer's Guide</i>.</p>
Visible	Select the Visible check box to have the system include the group of Brokers or group of clients in the topology view.

Field	Description
	Clear the Visible check box if you do not want the system to display the Brokers or clients in the topology view.
Combined	Select the Combined check box to have the system combine all the matching Brokers or matching clients into a single node for the topology view. When the system displays the single node, it labels the node with the value you specify for Match Name . Clear the Combined check box if you want the system to display each matching Broker or matching client using individual nodes.

- Click **OK** to close the Add Topology Display Group page and return to the **Node Grouping** tab on the Broker Display Options page.
- If you want to create another grouping, repeat steps step 4 through step 6.
- After creating the node groupings, use the buttons in the **Move Up** and **Move Down** columns, which are on the Broker Display Options page, to put the groupings in the order you want. For more information, see [“About Ordering the Node Groupings” on page 270](#).
- Click **OK** on the Broker Display Options page.

Tip: You can export the list of node groupings to a .csv file by clicking **Export Table**.

Editing Existing Node Groupings

After you create a node grouping, you might want to update it. For example, you might originally have the **Combined** check box clear so that the members of the node group use individual nodes and want to change the node grouping to select the **Combined** check box so all members use a single node.

To edit the settings for an existing node grouping

- Display the topology view as described in [“Displaying the Broker Topology View” on page 262](#).
- Click the **Display Settings** link.
- Click the **Node Grouping** tab.
- Click  **Edit** for the node grouping you want to update.
- On the Edit Topology Display Group page, update the **Match Type**, **Match Name**, **Visible**, and/or **Combined** fields. For a description of these fields, see [“Defining the Brokers and Clients to Display in the Topology View” on page 267](#).

6. Click **OK** to close the Edit Topology Display Group page and return to the **Node Grouping** tab on the Broker Display Options page.
7. Click **OK** on the Broker Display Options page.

Deleting Node Groupings

If you no longer need a node grouping, you can delete it.

To delete node groupings

1. Display the topology view as described in [“Displaying the Broker Topology View” on page 262](#).
2. Click the **Display Settings** link.
3. Click the **Node Grouping** tab.
4. Select the check boxes in the rows of the node groupings you want to delete.
5. Click **Delete**.
6. Click **OK** on the Broker Display Options page.

About Ordering the Node Groupings

The system uses each node grouping in the order you list them on the **Node Grouping** tab of the Broker Display Options page. For each node grouping, the **Match Type** and **Match Name** fields determine the Brokers or clients that are in the grouping. Order is important because after a Broker or client is added to a node grouping, it is no longer available for consideration in any other node groupings.

For example, assume you have clients with the following client IDs: `north1`, `north2`, `north3`, `northwest1`, `northwest2`. You define the following node groupings in the following order:

Match Type	Match Name	Visible	These clients display in the topology view
Client ID	northwest*	no	north1 north2 north3
Client ID	north*	yes	

Because the node grouping to hide clients with client IDs that start with "northwest" is listed first, the clients `northwest1` and `northwest2` are included in the first, hidden node grouping and are therefore not available for the second, visible node grouping.

However, if you switch the order of the node groupings, as shown below, the clients with client IDs that start with "northwest" are displayed because they would be a part

of the first, visible node grouping, and therefore they are not available for the second, hidden node grouping.

Match Type	Match Name	Visible	These clients display in the topology view
Client ID	north*	yes	north1 north2 north3 northwest1 northwest2
Client ID	northwest*	no	

You should set your display settings as follows:

1. Use the **Display Options** tab to indicate whether you want the system to include disconnected Brokers, remote Brokers, or disconnected clients in the topology view. For more information, see [“Defining How to Display Data in the Broker Topology View” on page 265](#).
2. Order the node groupings to hide the Brokers and clients that you do not want included in the topology view. Create the node groupings and ensure the **Visible** check box is not selected.
3. Create groupings that you mark as visible and set the **Combined** check box based on whether you want the system to display members of the group using individual nodes or combined as a single node.

Ordering Node Groupings

Use the following procedure to order the node groupings.

To order the node groupings

1. Display the topology view as described in [“Displaying the Broker Topology View” on page 262](#).
2. Click the **Display Settings** link.
3. Click the **Node Grouping** tab.
4. Use the buttons in the **Move Up** and **Move Down** columns to put the groupings in the order you want.
5. Click **OK**.

Defining the Documents to Include in Traces

By default when you perform a document trace, the system traces documents of all document types. Use the **Documents** tab of the Broker Display Options page to limit the traced documents by selecting the document types that you do not want to trace.

Be aware that settings you make on the **Documents** tab also affect:

- The trace of documents on the **Trace** tab of the Broker Details page. For more information, see [“Table View of Trace Information” on page 272](#).
- The trace of documents in the territory topology view. For more information, see [“Topology View of Territories” on page 275](#).

To define the documents to include in traces

1. Display the topology view as described in [“Displaying the Broker Topology View” on page 262](#).
2. Click the **Display Settings** link.
3. Click the **Documents** tab.
4. Select the check box for each document type that you do not want to trace.
 - To easily select all document types, click **Hide All**.
 - If you want to clear all check boxes for all document types, click **Show All**.

Tip: You can export the list of document types to a .csv file by clicking **Export Table**.

5. Click **OK**.

Table View of Trace Information

You can have the system collect data about documents that are actively flowing across client connections. You can then use the **Trace** tab to view the trace data in a table format, which is referred to as the trace view.

After you start a document trace in trace view, the trace continues until you manually stop it. The trace does not timeout, even if you switch to another My webMethods page.

Important: The system records the trace data in the memory of the My webMethods Server. As a result, tracing documents will consume memory. Use it as a diagnostic tool when needed. Stop the trace after you have collected the data you need. For instructions, see [“Stopping the Trace in the Trace View” on page 274](#).

Data Displayed in Trace View

The following table describes the data the system displays in the table on the **Trace** tab.

Column	Description
Trace Number	An internally generated index number for a line in the trace.
Document Type	The document type of the traced document.
Broker/Client	A Broker name or client ID associated with the operation displayed in the Operation column.
Application Name	The client application name of the client that either published or received the document.
Operation	<p>The operation associated with the document that was traced. It will be one of the following:</p> <ul style="list-style-type: none"> ■ Brokeradded. A new Broker has been added. ■ Brokerreconnected. A connection to the Broker has been reestablished. ■ Brokerconnected. A connection to the Broker has been made. ■ Brokerdisconnected. The Broker is disconnected. Reasons might be that the Broker Server has been restarted or stopped. ■ Client created. A new client has been created. ■ Client connected. A connection to the client has been made. ■ Client destroyed. The client has been deleted. ■ Client disconnected. A client is disconnected from the Broker. ■ Drop. A client has no permission to receive the document, so the Broker is dropping the document. ■ Enqueue. A Broker received the document and placed it in a queue. ■ Publish. Either a client published a document or a remote Broker forwarded a document. ■ Receive. A Broker received an acknowledgment for the document from a client that subscribes to the document and has dequeued the document.
Time Stamp	The date and time the operation was logged for the traced document.

Setting Your User Preferences for the Trace View

For tracing documents, you define a user preference identifying the document types you want to trace. Because this setting is a user preference, it applies only to your My webMethods user account. Also, the document types you identify are the same document types that you identify for tracing documents in Broker topology view.

For instructions for how to define the document types you want to trace, see [“Defining the Documents to Include in Traces” on page 271](#).

Running Document Traces in Trace View

You start or resume a document trace to indicate that you want the system to start storing information about documents that are actively flowing across client connections.

To start or resume a document trace in trace view

1. In My webMethods: **Messaging > Broker Servers > Brokers**.
2. In the **Brokers List**, click the Broker for which you want to run a document trace. If the Broker does not appear in the list, use the **Search** tab to locate it.
3. On the Broker Details page, click the **Trace** tab.
4. Click **Resume Trace**.

When you start or resume a trace, the system begins storing trace data in the memory of the My webMethods Server. You must refresh the trace view to have the trace data displayed on the **Trace** tab.

Refreshing the Data in the Trace View

The system does not automatically display data in the trace view when you start or resume a trace. Similarly, the system does not automatically refresh the data on the **Trace** tab as a trace is running. You must manually request that the system display the data it has recorded in the memory of the My webMethods Server.

To refresh the data in trace view

- On the **Trace** tab, click **Refresh**.

Stopping the Trace in the Trace View

When you no longer need a document trace to run, you should stop the trace. The system does not automatically stop a trace. It will continue to run until you stop it or until you log out of My webMethods.

When you stop the trace, the system clears the trace data from memory. If you need to keep the trace data, you can save it to a file before stopping the trace. For instructions, see [“Exporting the Data in the Trace View to a .csv File” on page 275](#).

To stop a document trace

On the **Trace** tab, click **Pause Trace**.

If you want to start the trace again, you can resume it by clicking **Resume Trace**.

Exporting the Data in the Trace View to a .csv File

If you want to save the trace data to a file, you can export the displayed trace data to a Comma Separated Value (.csv) file.

To export the trace data to a .csv file

1. On the **Trace** tab, click **Export Table**.
2. In the EXPORT TABLE dialog, select the character coding you want to use.
3. Click **Export**.

Clearing the Trace Data

If you no longer need to view the trace data, you can clear it from the page. Clearing the trace data from the display does not remove the data from memory. You must stop the trace to remove trace data from the memory. For instructions, see [“Stopping the Trace in the Trace View” on page 274](#).

To clear trace data from your machine's memory

On the **Trace** tab, click **Clear**.

Topology View of Territories

The territory topology view provides a graphical representation of territories. The territories in the display are those associated with the Broker Servers in your Broker Server list. The display also shows the gateways between territories, if any.

Note: For the territory topology view to display correctly, you should ensure that your territory names are unique. If two or more territories have the same name, the topology might not display correctly.

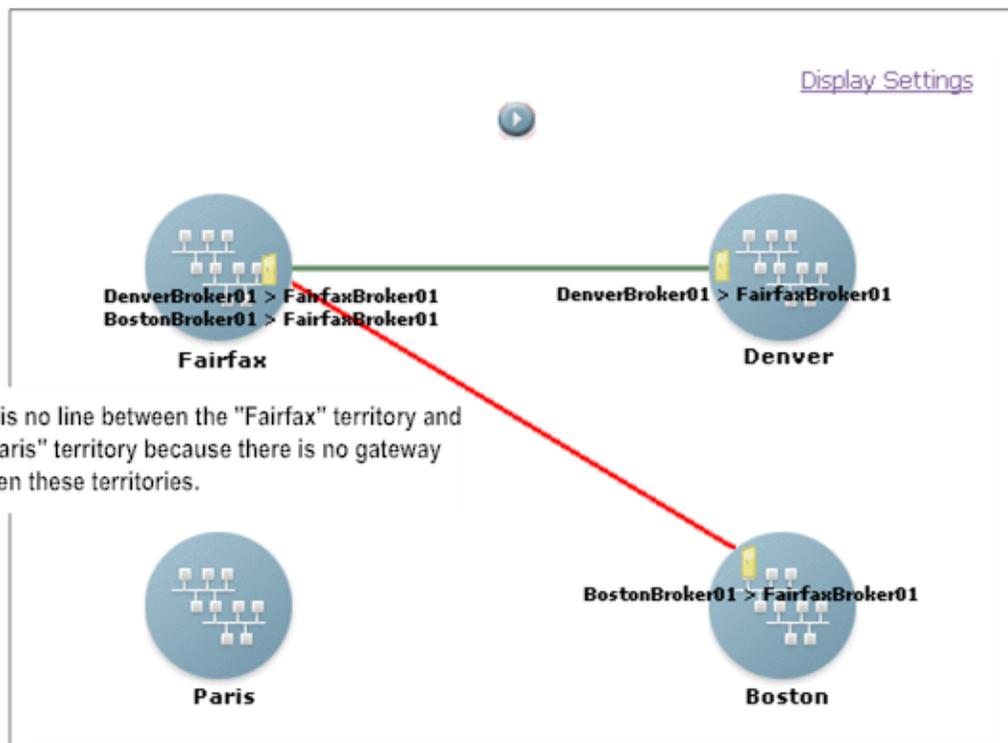
Additionally, you can trace documents in the territory topology view to display documents that are actively flowing across gateways. For more information, see [“About Tracing a Document in the Territory Topology View” on page 278](#).

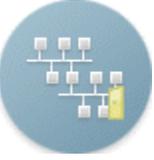
Note: The territory topology view requires that you have the Scalable Vector Graphics (SVG) viewer installed on the machine where your browser is running. You can download this viewer from [“http://www.adobe.com/svg/viewer/install/”](http://www.adobe.com/svg/viewer/install/). The SVG viewer with the Firefox Web browser is not

supported. If you do not have the SVG viewer installed, when you attempt to use the territory topology view, Broker prompts you to install it.

Data Displayed in the Territory Topology View

The following shows a sample of the territory topology view. For more details about the meanings of the icons and lines displayed in the topology view, see the table after the diagram.



Icons and Lines	Description
	A territory without a gateway Broker. An icon for a territory that does not have a yellow door on it represents a territory without a gateway Broker.
	A territory with a gateway Broker. An icon for a territory with a yellow door on it represents a territory with a gateway Broker. The yellow door represents the gateway Broker. You determine whether to include information about gateway Brokers in the topology view. The label indicates the names of the gateway Brokers and the direction of the

Icons and Lines	Description
	<p>gateway. For example, the label in the following sample indicates:</p> <ul style="list-style-type: none"> ■ The name of the gateway Broker in the Denver territory is DenverBroker01. ■ The name of the gateway Broker in the Fairfax territory is FairfaxBroker01. ■ The direction is from the Denver territory to the Fairfax territory. <div data-bbox="537 661 896 850" style="text-align: center;"> </div> <p>For more information about how to include information about gateway Brokers in the territory topology view, see “Setting Your User Preferences for the Territory Topology View” on page 279.</p>
	<p>Active gateway connection. A solid green line represents a gateway between two territories where the permissions are set up properly.</p>
	<p>Gateway connection error. A solid red line indicates that there is a problem with the permission setup for a gateway between the two territories.</p>
	<p>The start trace button. Use this button to animate the topology view allowing you to trace documents actively flowing across gateways. For more information, see “About Tracing a Document in the Territory Topology View” on page 278.</p>
	<p>The stop trace button. Use this button to stop the document trace in the topology view.</p>

Displaying the Territory Topology View

By default, the system displays information for all territories in the topology view.

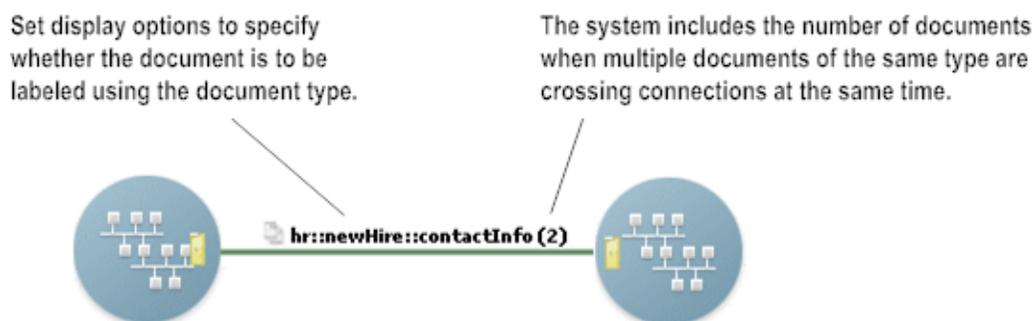
To define whether you want to include information about gateway Brokers in the topology view or to specify the territories you want included in the topology view, see [“Setting Your User Preferences for the Territory Topology View”](#) on page 279.

To display the territory topology

- In My webMethods: **Messaging > Broker Territories > Topology**.

About Tracing a Document in the Territory Topology View

While in the topology view, you can start a trace to view documents that are actively flowing across gateways. By default, the system traces documents of all document types. The following shows a sample of a document being traced.



To define your display settings for the topology view or specify the territories that appear in the topology view, see [“Setting Your User Preferences for the Territory Topology View” on page 279](#).

Important: Tracing documents in the topology view can consume a lot of system resources, for example, the memory of the My webMethods Server. Use it as a diagnostic tool when needed and do not let it run for long periods of time. It is recommended that you do not trace documents in a production environment when there is a high load of activity.

After you start a document trace, the trace continues as long as you stay on the Territories Topology page until you manually stop the trace. However, if you switch to another page within My webMethods, a timeout timer starts. The timeout value is 30 seconds. To conserve system resources, after 30 seconds elapse, the system automatically stops the document trace. If you switch back to the Topology page before the timer expires, the system clears the timeout timer and the trace will continue. You cannot configure the timeout value.

Tracing a Document in the Territory Topology View

Use the following procedure to trace a document in the territory topology view.

To run a document trace in the topology view

1. To limit the documents that the system traces, you must define the document types to trace per Broker. To do so, use the instructions in [“Defining the Documents to Include in Traces” on page 271](#).

To view the list of document types that the system will trace, you can use the **Documents** tab of the territory topology view display options. For instructions, see [“Viewing Documents that Are Included in Territory Topology Document Traces”](#) on page 285.

2. Display the topology view as described in [“Displaying the Territory Topology View”](#) on page 277.
3. To customize the display of the topology view or specify the territories to include, see [“Setting Your User Preferences for the Territory Topology View”](#) on page 279.
4. Click the start trace button .

While the trace is running, the start trace button is replaced with the stop trace button . Stop the trace by clicking the stop trace button .

Setting Your User Preferences for the Territory Topology View

The system maintains the display settings for the topology view as user preferences. As a result, the settings you make apply only to your My webMethods user account.

The following table lists the display settings, whether the setting affects the static topology view or both the static view and the topology view while tracing a document, and where you can find more information about how to set the display setting:

Display Setting	Static or Document Tracing	For more information, see...
Whether to display labels for the gateway Brokers	Static and Document Tracing	“Defining How to Display Data in the Territory Topology View” on page 280
Whether to label the traced documents	Document Tracing	“Defining How to Display Data in the Territory Topology View” on page 280
How often the system is to automatically refresh the data in the topology view	Static and Document Tracing	“Defining How to Display Data in the Territory Topology View” on page 280
Maximum number of nodes to display	Static and Document Tracing	“Defining How to Display Data in the Territory Topology View” on page 280

Display Setting	Static or Document Tracing	For more information, see...
Filters for identifying the territories to display	Static and Document Tracing	“Identifying the Territories to Show in the Topology View” on page 281
Groups territories to display as single node in the topology view	Static and Document Tracing	“Identifying the Territories to Show in the Topology View” on page 281
Documents to trace	Document Tracing	“Defining the Documents to Include in Traces” on page 271

Note: Permit the documents to include by setting the display options for Brokers.

Defining How to Display Data in the Territory Topology View

You can define display settings that determine:

- Labels to use in the topology view
- How often to refresh the display
- How many total nodes to display in the topology view

To define how to display data in the territory topology view

1. Display the topology view as described in [“Displaying the Territory Topology View” on page 277](#).
2. Click the **Display Settings** link.
3. On the **Display Options** tab, set the following fields:

Field	Description
Show Gateway Name	Select this check box to include information about the gateway Brokers.
Show Document Type Name	Select this check box if you want traced documents to be labeled with the document type name. The system only displays documents when you start a document trace. For more information, see “About Tracing a Document in the Territory Topology View” on page 278 .

Field	Description
Poll Interval	Specify how often (in seconds) you want the system to poll for data to update the information in the topology view. This affects both a static topology view and the view during a document trace. The default is 4 seconds.
Maximum Territory Node Number	Specify the maximum number of territory nodes to include in the topology view. The default is 10. Tip: Narrow down the number of territory nodes to display in the topology view so the number is less than the Maximum Territory Node Number . For more information, see “Identifying the Territories to Show in the Topology View” on page 281. If the number of territory nodes to display is greater than the Maximum Territory Node Number , the actual territory nodes displayed are random.

4. Click **Save** to save your changes.
5. Click **OK**.

Identifying the Territories to Show in the Topology View

By default, the system displays all territories in the topology view.

You can define one or more node groupings to identify a group of territories. After identifying a node grouping, you can select whether you want to:

- View or hide the territories in the node grouping.
- Show each territory using individual nodes for each or combine all territories in the node grouping and display them as a single node.

To identify the territories to display in the topology view

1. Display the topology view as described in [“Displaying the Territory Topology View”](#) on page 277.
2. Click the **Display Settings** link.
3. Click the **Node Grouping** tab.
4. Click **New Group** to define a new node grouping.
5. On the Add Topology Display Group page, set the following fields:

Field	Description
Match Type and Match Name	<p>Use the Match Type and Match Name fields to identify a group of territories. The match is case sensitive.</p> <ul style="list-style-type: none"> ■ Match Type. The Match Type identifies the attribute of a territory that you want to use to form the group. The only Match Type is Territory Name, which allows you to create a group of territories that have the same or similar territory names. ■ Match Name. Specify text to use to find matching territories for the group. For example, if you want a group of territories that all have names that begin with "p", specify p* in the Match Name field. <p>When specifying the Match Name, you can place an asterisk (*) at the end of the specified text to match multiple characters.</p>
Visible	<p>Select the Visible check box to have the system include the group of territories in the topology view.</p> <p>Clear the Visible check box if you do not want the system to display the territories in the topology view.</p>
Combined	<p>Select the Combined check box to have the system combine all the matching territories into a single node for the topology view. When the system displays the single node, it labels the node with the value you specify for Match Name.</p> <p>Clear the Combined check box if you want the system to display each matching territory using individual nodes.</p>

6. Click **OK** to close the Add Topology Display Group page and return to the **Node Grouping** tab on the Territory Display Options page.
7. If you want to create another grouping, repeat steps step 4 through step 6.
8. After creating the node groupings, use the buttons in the **Move Up** and **Move Down** columns, which are on the Territory Display Options page, to put the groupings in the order you want. For more information, see [“About Ordering the Node Groupings” on page 283](#).
9. Click **OK** on the Territory Display Options page.

Tip: You can export the list of node groupings to a .csv file by clicking **Export Table**.

Editing Existing Node Groupings

After you create a node grouping, you might want to update it. For example, you might originally have the **Combined** check box clear so the display shows the territories of the group using individual nodes and want to change the node grouping to select the **Combined** check box so the display shows all territories in the node grouping using a single node.

To edit the settings for an existing node grouping

1. Display the topology view as described in [“Displaying the Territory Topology View” on page 277](#).
2. Click the **Display Settings** link.
3. Click the **Node Grouping** tab.
4. Click  **Edit** for the node grouping you want update.
5. On the Edit Topology Display Group page, update the **Match Type**, **Match Name**, **Visible**, and/or **Combined** fields. For a description of these fields, see [“Identifying the Territories to Show in the Topology View” on page 281](#).
6. Click **OK** to close the Edit Topology Display Group page and return to the **Node Grouping** tab on the Topology Display Options page.
7. Click **OK** on the Territory Display Options page.

Deleting Node Groupings

If you no longer need a node grouping, you can delete it.

To delete node groupings

1. Display the topology view as described in [“Displaying the Territory Topology View” on page 277](#).
2. Click the **Display Settings** link.
3. Click the **Node Grouping** tab.
4. Select the check boxes in the rows of the node groupings you want to delete.
5. Click **Delete**.
6. Click **OK** on the Territory Display Options page.

About Ordering the Node Groupings

The system uses each node grouping in the order you list them on the **Node Grouping** tab of the Topology Display Options page. For each node grouping, the **Match Type** and **Match Name** fields determine the territories that are in the grouping. Order is important because after a territory is added to a node grouping, it is no longer available for consideration in any other node groupings.

For example, assume you have territories with the following names: `Fairbanks`, `Fairfax`, `Frankfurt`. You define the following node groupings in the following order:

Match Type	Match Name	Visible	These territories display in the topology view
Territory Name	Fa*	no	Frankfurt
Territory Name	F*	yes	

Because the node grouping to hide territories with names that start with "Fa" is listed first, the territories `Fairbanks` and `Fairfax` are included in the first, hidden node grouping and are therefore not available for the second, visible node grouping.

However, if you switch the order of the node groupings, as shown below, the territories with names that start with "Fa" are displayed because they are included in the first, visible node grouping, and therefore they are not available for the second, hidden node grouping.

Match Type	Match Name	Visible	These territories display in the topology view
Territory Name	F*	yes	Fairbanks Fairfax Frankfurt
Territory Name	Fa*	no	

You should set your display settings as follows:

1. Order node groupings to hide the territories that you do not want included in the topology view. Create the node groupings and ensure the **Visible** check box is not selected.
2. Create groupings that you mark as visible and set the **Combined** check box based on whether you want the system to display members of the group using individual nodes or combined as a single node.

Ordering Node Groupings

Use the following procedure to order node groupings.

To order the node groupings

1. Display the topology view as described in [“Displaying the Territory Topology View” on page 277](#).
2. Click the **Display Settings** link.

3. Click the **Node Grouping** tab.
4. Use the buttons in the **Move Up** and **Move Down** columns to put the groupings in the order you want.
5. Click **OK**.

Viewing Documents that Are Included in Territory Topology Document Traces

By default when you perform a document trace, Broker traces documents of all document types. To limit the documents to include in a trace, you set the documents for each Broker. For instructions, see [“Defining the Documents to Include in Traces” on page 271](#).

You can use the **Documents** tab of the Topology Display Options page to view the documents that the system will trace.

To view the documents that the system will trace

1. Display the topology view as described in [“Displaying the Territory Topology View” on page 277](#).
2. Click the **Display Settings** link.
3. Click the **Documents** tab.

Tip: You can export the list of document types to a .csv file by clicking **Export Table**.

4. Click **OK**.

13 Managing Broker Security

■ Overview	288
■ Data Protection	288
■ Broker Security Model	288
■ Securing Broker Server Using Basic Authentication	293
■ Securing Broker Server Using SSL	309
■ Securing Broker Server Using Basic Authentication Over SSL	324
■ Securing Broker Server Using FIPS	324
■ Access Control Lists	325
■ Broker Security and My webMethods Server	346
■ Converting Certificate Files	351

Overview

This chapter explains how webMethods Broker security works. It describes the Broker security model and explains how to configure *Basic Authentication*, *Secure Sockets Layer (SSL)*, *Federal Information Processing Standards (FIPS)*, and *Certificate Revocation List (CRL)* for Broker.

The setup and usage of basic authentication, SSL authentication, keystore files, truststore files, and Access Control Lists (ACLs), which are key elements of the Broker security model are covered in detail. Step-by-step instructions are provided to guide you through the basic authentication, SSL, FIPS, and CRL configuration and implementation process for all your Broker system components.

Note: This chapter assumes that the administrator implementing Broker security has a basic understanding of basic authentication, SSL, FIPS, and CRL including concepts such as certificate files, certificate authorities (CAs), certificate revocation, trusted roots, distinguished names (DNs), and public and private key pairs. For enabling basic authentication, knowledge of directories such as LDAP and ADSI is required.

Data Protection

Broker is in compliance with General Data Protection Regulation (GDPR) and does not store, collect, or process any personally identifiable information. By default, Broker allows full access to anonymous users for operations like publish/subscribe events and administrative operations like start, stop, and so on. This also includes the ability to browse the queues for messages which might contain personally identifiable information. The messages in the broker queues might contain personal data based on the business case. Hence, provide access only to authorized personnel.

Broker Security Model

The Broker security model is designed to safeguard all major components of your webMethods Broker installation as well as Broker data. The Broker security model uses:

- Basic authentication to authenticate Broker Servers and clients.

Authentication is the process of validating the identity of an entity attempting to establish a connection. Basic authentication provides a light-weight mechanism to authenticate Broker Servers and clients with users existing in the operating system, Lightweight Directory Access Protocol (LDAP), or Active Directory Service Interfaces (ADSI). For more information about configuring basic authentication, see [“Securing Broker Server Using Basic Authentication” on page 293](#).

- SSL to authenticate Broker Servers and clients.

The reference integrity through SSL guarantees the identity of a Broker Server to a requesting client and that of a client to a Broker Server. For more information about SSL configuration, see [“Securing Broker Server Using SSL” on page 309](#).

- FIPS to provide additional security.

You can ensure additional security on top of generic SSL on Broker Server by enabling FIPS. For more information, see [“Securing Broker Server Using FIPS” on page 324](#).

- CRL to provide increased security.

A certificate revocation check against a CRL provides a mechanism for protecting against using certificates that are compromised. For more information, see [“Securing Broker Server Using CRL” on page 323](#).

- Access Control List (ACL) to do following:

- Authorize administrative access to a Broker Server and client access to Broker objects and data.

Authorization is the process of granting (or denying) access permissions. The Broker security model authorizes access permissions by comparing a client's identity against a list of identities contained in an ACL. If the client's identity matches one of those listed, it is granted permission; if not, it is denied permission.

- Allow secure sharing of data between Brokers in territories, clusters, or gateways.

You can also set access permissions to control which Brokers can join a territory or a cluster. The Broker security model authorizes access permissions by comparing the identity of a Broker against a list of identities contained in an ACL. If the identity of a Broker matches one of those listed, it is granted permission; if not, it is denied permission.

For using ACL, Broker Server must be configured with one of the following:

- Basic Authentication
 - Basic Authentication with SSL
 - SSL (and optionally configure FIPS, or CRL, or both)
- Broker Server Port and Broker Monitor Port security.

You can specify individual IP addresses to which the Broker Server port and Broker Monitor port must bind to. You do this by specifying a value for the `broker-ipaddress` parameter in the `awbroker.cfg` file and the `monitor-ipaddress` parameter in the `awbrokermon.cfg` file.

For example, to bind Broker Monitor to the loopback address, you specify `monitor-ipaddress=127.0.0.1` in the `awbrokermon.cfg` file. For more information, see [“SSL Authentication and Broker Port Usage” on page 310](#).

It is not necessary that basic authentication, SSL, FIPS, or CRL be enabled when you are configuring the Broker Server Port and Broker Monitor Port security parameters.

Important: It is not mandatory to configure basic authentication or SSL to operate your webMethods Broker installation. However, if you do not configure basic authentication or SSL, any Broker user can establish the administrative credentials required to control your Broker Server, join a client group that has administrator permissions, reconfigure Broker, and access Broker data. Therefore, it is recommended that you read this chapter to understand how the Broker security model works, and then configure basic authentication or SSL to safeguard your Broker installation.

Authentication Identities

A key concept of the Broker security model is that of an *identity*. An identity represents the credentials of any entity requesting authentication or authorization within a Broker installation.

- For basic authentication, an identity is composed of user name authenticated by the *alias* (authentication system such as LDAP and ADSI) configured in the `basicauth.cfg` file.

For more information see [“Basic Authentication Configuration File” on page 294](#) and [“Configuring Basic Authentication Identity for a Broker Server ” on page 308](#).

- For SSL authentication, an identity is composed of SSL user DN and the SSL issuer DN.

For information about assigning SSL identity, see [“Assigning SSL Identities for a Broker Server ” on page 309](#).

An entity can be:

- A Broker Server
- The Broker user interface (or Broker admin components)
- Java, JMS, and C# client programs attempting to access the Broker

To fully implement the Broker security model, all of the above must be assigned identities.

Access Control Lists (ACLs)

There are two ACLs you can attach to a Broker component:

- A *user ACL*, which is a list of basic authentication user names and SSL user DNs. For example, `brokeruser` and `“CN=Broker Server, O=My Company”`.

- An *authenticator ACL*, which is a list of basic authentication system alias names and DNs for certification authorities or CAs (issuers of the user certificates). For example, "BrokerLDAP" and "CN=My Company SSL Issuer,O=My Company".

You configure ACLs to achieve one or more of the following:

- To restrict a client's administrative access to the Broker Server (for example, whether a client can stop or restart a Broker Server)
- To control whether a client can add or delete Brokers
- To protect the document types to which a client can publish or subscribe by enlisting authorized users in client groups
- To control whether clients have access to the system-defined admin client group
- To grant permissions for a Broker to access other Brokers in a territory
- To grant permissions for Brokers in one Broker territory to access remote Brokers in another territory through a Broker gateway
- To grant permissions for a Broker to access other Brokers in a cluster
- To grant permissions for Brokers in one Broker cluster to access remote Brokers in another cluster through a Broker cluster gateway

For more information, see ["Access Control Lists" on page 325](#).

Securing Broker System Components

The Broker security model uses ACLs with basic authentication or SSL to secure the following Broker system components:

- Broker Server
- Clients, including the Broker user interface and Broker command-line utilities
- Remote Brokers in a territory
- Remote Brokers in a territory gateway
- Remote Broker in a cluster
- Remote Brokers in a cluster gateway

Securing the Broker Server

The Broker Server is the first Broker component you secure using basic authentication or SSL.

For information about securing Broker Server using basic authentication, see ["Securing Broker Server Using Basic Authentication" on page 293](#).

For information about securing Broker Server using SSL, see ["Securing Broker Server Using SSL" on page 309](#).

Securing the Broker User Interface

To a Broker Server, the Broker user interface component is just another client (for example, a Java program) requesting a secure connection.

For information about securing the Broker user interface for basic authentication, see [“Configuring a Basic Authentication Identity for the Broker User Interface Component” on page 347](#).

For information about securing the Broker user interface for SSL authentication, see [“Configuring an SSL Identity for the Broker User Interface Component” on page 348](#).

Securing Broker Command-Line Utilities

When basic authentication or SSL authorization is used, most Broker command-line utilities must present a valid basic authentication identity or SSL identity before connecting (only `server_config`, `server_conf_restore`, and `broker_start` do not require an identity).

To simplify administration using SSL, it is recommended that you use the same keystore, truststore, and DN that you use for the Broker user interface.

For more information, see [“ webMethods Broker Command-Line Utilities” on page 525](#).

Securing Client Groups

A client application that participates in the authentication process must still be granted authorization to access specific document types. Assigning individual clients to client groups with particular sets of access permissions accomplishes that purpose. For example, you may want to retain a document type, such as "expenses," that is only accessible to clients belonging to the Broker client group "finance."

For information about securing the client groups, see [“Client Group ACLs” on page 326](#).

Important: For SSL authentication, you create SSL identities for clients that connect to the Broker Server in the same manner as you create identities for Broker Servers and the Broker user interface. However, you do not use the Broker user interface to manage the SSL identities of your client applications; these identities do not appear in the Broker user interface. You can only manage client identities through administrative tools of the client application, or through administrative tools that are compatible with the file format of the client's keystore and truststore files.

Securing Territories and Gateways

You can use a territory ACL to require permissions for a Broker to join and participate in a territory, and a gateway ACL to control whether Broker data is allowed to flow from

a connected territory across a gateway. For more information, see [“Territory ACLs” on page 329](#) and [“Territory Gateway ACLs” on page 331](#).

Securing Clusters and Cluster Gateways

You can use a cluster ACL to require permissions for a Broker to join and participate in a cluster, and a cluster gateway ACL to control whether Broker data is allowed to flow from a connected cluster across a cluster gateway. For more information, see [“Cluster ACLs” on page 330](#) and [“Cluster Gateway ACLs” on page 332](#).

Securing Broker Server Using Basic Authentication

Broker supports basic authentication to authenticate Broker clients by using the user name and password. You can include basic authentication users in an ACL to control access to the protected Broker objects.

Use basic authentication to secure Broker Servers, remote Brokers in a territory or a cluster, and remote Brokers in a territory gateway or a cluster gateway.

Important: Broker Server supports caching of authenticated users. When authentication information stored in a user directory becomes unavailable, the cached authentication information is used. The cached authentication information in the Broker Server is lost when the Broker Server is restarted.

Using the Basic Authentication Security Model

To use the basic authentication security model

1. Enable basic authentication on Broker Server by setting a user directory system and configuring the authentication mechanism in the basic authentication configuration file. For information about how to enable basic authentication on a Broker Server, see [“Enabling Basic Authentication” on page 294](#).
2. Configure basic authentication identity (user name and password) for a Broker Server. Broker Server uses this identity to identify itself to other Brokers in a territory or cluster, or to a Broker at the other end of a gateway. For information about configuring the basic authentication identity for a Broker Server, see [“Configuring Basic Authentication Identity for a Broker Server” on page 308](#).
3. Configure basic authentication identity for the Broker user interface component to administer an ACL protected Broker Server. For information, see [“Configuring a Basic Authentication Identity for the Broker User Interface Component” on page 347](#).
4. Assign ACLs to the Broker Server so that only authorized users can access its administrative functions, such as viewing Broker Server's identity, creating Brokers, enabling basic authentication, stopping the Broker Server, re-configuring Brokers,

and disabling basic authentication. You need to set the administrator identity for the client. For more information, see “[Broker Server ACLs](#)” on page 328.

5. Assign ACLs to client groups, territories, clusters, and gateways. For more information, see “[Client Group ACLs](#)” on page 326, “[Territory ACLs](#)” on page 329, “[Cluster ACLs](#)” on page 330, “[Territory Gateway ACLs](#)” on page 331, and “[Cluster Gateway ACLs](#)” on page 332.
6. Configure basic authentication identity for Java clients, JMS clients, C clients, C# clients, or other Broker Servers to connect to an ACL protected Broker Server. For information, see “[Managing Clients](#)” on page 187, *webMethods Broker Administration Java API Programmer’s Guide*, *webMethods Broker Client Java API Programmer’s Guide*, *webMethods Broker Messaging Programmer’s Guide*, and *webMethods Broker Client C API Programmer’s Guide*.

Enabling Basic Authentication

You must enable basic authentication on Broker, if you want Broker to authenticate Broker clients using the basic authentication credentials (user name and password).

To enable basic authentication on Broker

1. Set up a user authentication system. Broker authenticates the users against this user directory. Users can be defined in one or more of the following authentication system:
 - Operating system
 - LDAP
 - ADSI

Note: Broker does not handle administrative tasks such as changing passwords, creating, deleting, or modifying users.

2. Configure the authentication mechanism by specifying the configuration parameters for the authentication systems (for example, LDAP and ADSI) in the basic authentication configuration (*basicauth.cfg*) file.
3. Configure the basic authentication property in the *awbroker.cfg* file to point to the *basicauth.cfg* file, if it is not already set. For example, set `basic-auth-cfg-file=webMethods Broker_directory\data\awbrokersversion\default\basicauth.cfg`.

Basic Authentication Configuration File

You will find a sample basic authentication configuration file, in the following location:

%BROKER_HOME%\config folder

The basic authentication configuration file contains one or more directory aliases. An alias marks the beginning of a set of directory type configuration parameters. By default,

the basic authentication configuration file contains sample entries for each supported authentication system (OS, LDAP, and ADSI). All of these entries are commented by using a # (hash) symbol in the first column. You must alter this file and uncomment the authentication system you are using. If required, additional entries for authentication systems can be added by specifying a new alias, type, followed by the directory-type specific configuration parameters. The following is a sample basic authentication configuration file for your reference.

```
logfile=basicauth.log
loglevel=1

#Sample OS authentication entry
#alias=LocalOS
#authtype=OS
#os-win-auth-user-exist=true
#defaultdomain=eur

# Sample LDAP authentication entry
#scan-all-alias=0
#alias-min-disable-time=30
#alias-max-disable-time=120
#alias=LDAP1
#authtype=LDAP
#serverhost=ldap://linuxserver:389
#ldap-person-base-binddn=ou=users,o=webmtest
#ldap-group-base-binddn=ou=groups,o=webmtest
#ldap-person-objectclass=organizationalPerson
#ldap-group-objectclass=groupOfUniqueNames
#ldap-group-prs-attr=uniqueMember
#ldap-server-type=OpenLdap
#ldap-userid-field=uid
#ldap-groupid-field=cn
#ldap-allow-domain-as-base-bind-dn=true
#ldap-person-property-attr=cn,sn
#ldap-group-property-attr=cn
#resolve-groups=ru

# Sample ADSI authentication entry
#alias=ADSI1
#authtype=ADSI
#serverhost=server1
#adsi-forest-dn=dc=ad,dc=myCompany
#defaultdomain=eur
```

You need to manually maintain the basic authentication configuration file on the file system. After making changes to the file, you must restart Broker Server for the changes to take effect.

When a client connects to Broker Server and passes the credentials (user name, and password), Broker Server authenticates them against the configured aliases in sequence. If an alias is not accessible, or authentication fails against that alias, Broker Server tries to authenticate the user against the next alias in the list. If authentication fails for all aliases, the connection request is rejected.

Basic Authentication on UNIX

On UNIX, basic authentication through the operating system requires special file permissions set on the sagssxauthd2 daemon program. Make sure the sagssxauthd2

daemon program installed in the *Software AG_directory/common/security/ssx/auth* directory is set with these special file permissions:

- Root is the owner of the file.
- The `set-uid` flag is set for the owner.

If any of the above mentioned file permissions is not met, to enable operating system authentication from the root, execute the following command:

```
Software AG_directory/common/security/ssx/auth/set_daemon_privs.sh
```

When the first authentication request on the operating system occurs, Broker Server automatically creates a daemon process called `sagssxauthd2` with root as its owner. This daemon stops when you stop Broker Server.

Basic Authentication Configuration Parameters

The following table provides details of the parameters you configure in the basic authentication configuration file.

For this parameter...	Specify this value...	Applicable to...
<code>adsi-domain-dn</code>	<p>This value (together with <code>adsi-domain-short</code>) allows to create a mapping of short names (specified in the domain parameter) to long (distinguished name) entries.</p> <p>NoteThe elements are positional and must match in number with the next entry</p> <p>Example:</p> <pre>dc=euro,dc=myorg,dc=com;dc=usa,dc=myorg,dc=com;ou=developers,dc=euro,dc=myorg,dc=com</pre> <p>Default: None.</p>	ADSI
<code>adsi-domain-short</code>	<p>A semi-colon separated list of domain names, or specific nodes, or both.</p> <p>This value (together with <code>adsi-domain-dn</code>) allows to create a mapping of short names to long (distinguished name) entries.</p> <p>NoteThe elements are positional and must match in number with the <code>adsi-domain-dn</code> entry.</p>	ADSI

For this parameter...	Specify this value...	Applicable to...
	<p>Example: <code>euro;usa;euro-developers</code></p> <p>Default: None.</p>	
<code>adsi-forest-dn</code>	<p>The name of the ADSI forest here. This value is used numerous times when accessing the ActiveDirectory.</p> <p>Example: <code>dc=myorg,dc=com</code></p>	ADSI
<code>adsi-group-base-binddn</code>	<p>The BindDN that is used to access a group.</p> <p>NoteThis parameter is only useful when all groups that are accessed are found in the same node. Otherwise, specifying this value is optional.</p> <p>Default: None.</p>	ADSI
<code>adsi-person-base-binddn</code>	<p>The BindDN that is used to access a user.</p> <p>NoteThis parameter is only useful when all users that are accessed are found in the same node. Otherwise, specifying this value is optional.</p> <p>Default: None.</p>	ADSI
<code>alias</code>	<p>An alias marks the beginning of a set of directory type configuration parameters. In general, an <code>authtype</code> entry is specified below an alias entry followed by the directory-type specific configuration parameters.</p>	All configurations
<code>alias-max-disable-time</code>	<p>The maximum time you want to disable an unresponsive alias.</p> <p>Default: 120 seconds.</p>	All configurations
<code>alias-min-disable-time</code>	<p>The minimum time you want to disable an unresponsive alias.</p>	All configurations

For this parameter...	Specify this value...	Applicable to...
	Default: 30 seconds.	
authtype	<p>The authentication system you want to use for authentication:</p> <ul style="list-style-type: none"> ■ OS (native operating system) ■ LDAP (LDAP server) ■ ADSI (Active Directory Server Interface, Windows only) <p>Default: None.</p>	All configurations
defaultdomain	<p>A domain name that Broker must use when no domain name is specified while authenticating a user.</p> <p>Default: None.</p>	All configurations
defaultgroup	<p>A default group name that Broker must use when no group name is specified while authenticating a user.</p> <p>Default: None.</p>	OS, LDAP
ldap-allow-domain-as-base-binddn	<p>Boolean value.</p> <p>If this value is "true" or "1", the parameter "domainname" will be interpreted as a BaseBindDN. If defaultdomain is set, this value will be interpreted as BaseBindDN.</p> <p>Default: None.</p>	LDAP
ldap-allow-domain-as-base-bind-dn	<p>Boolean value.</p> <p>If the value is "true" or "1", the parameter "domainname" will be interpreted as a BaseBindDN.</p> <p>Example: ou=People, dc=myorg, dc=com</p> <p>Note: If you do not specify the domain name explicitly while the defaultdomain parameter is</p>	LDAP

For this parameter...	Specify this value...	Applicable to...
	<p>already set, this value is interpreted as the BaseBind domain name.</p> <p>Default: None.</p>	
<p>ldap-domain-long</p>	<p>Distinguished names of LDAP nodes.</p> <p>This value (together with ldap-domain-short) allows to create a mapping of short names (specified in the domain parameter) to long (distinguished name) entries. Note that the elements are positional and must match in number with the next entry.</p> <p>Example:</p> <pre>dc=euro,dc=myorg,dc=com;dc=usa,dc=myorg,dc=com;ou=developers,dc=euro,dc=myorg,dc=com</pre> <p>Default: None.</p>	<p>LDAP</p>
<p>ldap-domain-short</p>	<p>colon separated list.</p> <p>mapping of short names to long (distinguished name) entries.</p> <p>NoteThe elements are positional and must match in number with the next entry.</p> <p>Example: euro;usa;euro-developers</p> <p>Default: None.</p>	<p>LDAP</p>
<p>ldap-group-base-binddn</p>	<p>Distinguished Name for LDAP where the groups are to be found (see also ldap-person-base-binddn).</p> <p>Default: None.</p>	<p>LDAP</p>
<p>ldap-groupid-field</p>	<p>Property name that denotes a group entry.</p> <p>Default: None.</p>	<p>LDAP</p>

For this parameter...	Specify this value...	Applicable to...
<code>ldap-group-objectclass</code>	All object classes that are required to store a new group entry (comma separated list). Default: None.	LDAP
<code>ldap-group-property-attr</code>	Property names that can be accessed for a group entry. The value is a comma separated list, which contains the property name. Optionally, the property name might be followed by ":w" or ":r", depending on whether the property is writable (w) or only readable (r). Default: None.	LDAP
<code>ldap-group-prs-attr</code>	Property name of a group entry that points to the members of this group. Default: None.	LDAP
<code>ldap-password-field</code>	Property name that denotes the password field of a user entry. Default: None.	LDAP
<code>ldap-person-base-binddn</code>	Distinguished Name for LDAP where the authentication information is stored. While authenticating, this value will be prefixed with <code>ldap-userid-field</code> when issuing the LDAP authentication call. Default: None.	LDAP
<code>ldap-person-grp-attr</code>	Property name of a user entry that points to the group that the user is member of. Default: None.	LDAP
<code>ldap-person-objectclass</code>	All object classes that are required to store a new user entry (comma separated list).	LDAP

For this parameter...	Specify this value...	Applicable to...
	Default: None.	
<code>ldap-person-property-attr</code>	<p>Property names that can be accessed for a user entry.</p> <p>The value is a comma separated list, which contains the property name. Optionally, the property name might be followed by ":w" or ":r", depending on whether the property is writable (w) or only readable (r).</p> <p>Default: None.</p>	LDAP
<code>ldap-sasl-auth</code>	<p>Boolean value that forces the usage of the SASL (type DIGEST-MD5) authentication. This feature ensures that no passwords are transmitted over the wire. But be sure that this feature is supported by the LDAP server.</p> <p>Default: <code>false</code></p>	LDAP
<code>ldap-server-type</code>	<p>LDAP server type. Specifying this value will internally set the appropriate default values.</p> <p>The available server types are:</p> <ul style="list-style-type: none"> ■ ActiveDirectory <ul style="list-style-type: none"> See “Default Values for Active Directory Parameters” on page 305. ■ SunOneDirectory <ul style="list-style-type: none"> See “Default Values for SunOne Parameters” on page 306. ■ OpenLdap <ul style="list-style-type: none"> See “Default Values for OpenLdap Parameters” on page 306. <p>Default: <code>OpenLdap</code></p>	LDAP
<code>ldap-ssl-connection</code>	<p>Boolean value for enabling or disabling the SSL connection for LDAP.</p>	LDAP

For this parameter...	Specify this value...	Applicable to...
	Default: <code>false</code> OR 0(zero)	
<code>ldap-start-tls</code>	Boolean value that enforces the usage of a secure (TLS/SSL) line before the data traffic starts. Default: None.	LDAP
<code>ldap-userid-field</code>	Property name that denotes a user entry. Example: The DN: <code>uid=user01,ou=Test,dc=myorg,dc=com</code> requires the value of "uid". Default: None.	LDAP
<code>logfile</code>	An output file for logging. Default: <code>basicauth.log</code>	All configurations
<code>loglevel</code>	The log level. This value ranges from 0 (zero) for no logging to 6 (six) for maximum logging. Default: 1 (Minimum logging)	All configurations
<code>os-win-auth-no-domain-force-local</code>	Boolean value. When no domain is specified and this flag is on, then only the local machine will be checked for user authentication. If this flag is off, then the authentication will include automatically the domain that the machine is currently logged on to. Default: <code>false</code> OR 0(zero)	OS (Windows only)
<code>os-win-auth-user-exist</code>	When you authenticate a user on Windows 2000 operating system, you use the Security Support Provider Interface (SSPI) method by default. This method has a flaw that if the Guest account is enabled (and has no	OS (Windows 2000 only)

For this parameter...	Specify this value...	Applicable to...
	<p>password), the SSPI authentication returns successfully for any unknown user ID.</p> <p>Setting this value to "true" or "1" enforces an additional check to verify that the user really does exist and is not automatically mapped to the guest account.</p> <p>Default: <code>true</code></p>	
<code>os-win-check-local-groups</code>	<p>Boolean value.</p> <p>If this boolean flag is on ("true" or "1"), group membership is also validated against the local (PC) groups, rather than the domain groups.</p> <p>Default: <code>false</code> OR 0(zero)</p>	<p>OS (Windows only)</p>
<code>os-win-logonuser-on-2000</code>	<p>Boolean value.</p> <p>By default, SSPI is used to authenticate a user on Windows 2000 because the more common method "LogonUser()" requires additional rights by the caller. Still, LogonUser() is the preferred way and if the access rights are granted ("Act as part of the operating system"), it is less error prone to use the latter method.</p> <p>Default: <code>false</code></p>	<p>OS (Windows 2000 only)</p>
<code>os-win-no-impersonation</code>	<p>Boolean value that specifies whether any data access should be made under the impersonated user ID of the logged on user (<code>false</code>), or whether every access is made under the account of the running process (<code>true</code>).</p> <p>Default: <code>false</code></p>	<p>OS (Windows only)</p>
<code>resgroup-computed-property</code>	<p>Name of the computed property that will be read while resolving the group membership and the</p>	<p>LDAP</p>

For this parameter...	Specify this value...	Applicable to...
	<p>parameter <code>resolve-groups</code> is set to "ComputedProperty".</p> <p>Default: None.</p>	
<p><code>resolve-groups</code></p>	<p>The method you use to find all the groups that the user is a member of:</p> <ul style="list-style-type: none"> ■ RecurseUp (or RU). Use an attribute of the user to find the direct groups the user is a member of. Then continue up, using the same attribute until all groups are found. ■ RecurseDown (or RD). Perform one search for all groups that contain this user as a member. This search only finds the direct groups for a user. ■ ComputedProperty (or CP). Use a specific field that is a computed property and not a static field (evaluated at runtime by the LDAP server), to retrieve all group information in one call. <p>Default: <code>RecurseUp</code></p>	<p>LDAP</p>
<p><code>scan-all-alias</code></p>	<p>Boolean value.</p> <p>If <code>scan-all-alias=1</code>, Broker Server scans all the aliases configured in the <code>basicauth.cfg</code> file before authenticating a user. In this case, the basic authentication process take a long time.</p> <p>If <code>scan-all-alias=0</code>, Broker Server scans the aliases configured in the <code>basicauth.cfg</code> file and authenticates user with the first alias that match the user credentials. Broker Server does not scan all the aliases if the user credentials match with one of the aliases.</p> <p>Default: 0 (false).</p>	<p>All configurations</p>

For this parameter...	Specify this value...	Applicable to...
serverhost	Name of the server. Default: None.	LDAP, ADSI
serverport	Port number of serverhost. Default: 389	LDAP, ADSI
ssx-ldap-search-timeout	The maximum time, in seconds, SSX API will wait for the LDAP alias search status to be returned, before timing out the authentication request from Broker. Alternately, you can use the SSX_LDAP_TIMEOUT environment variable to configure the time-out value. Note When a value is specified for both <code>ssx-ldap-search-timeout</code> property and <code>SSX_LDAP_TIMEOUT</code> environment variable, the value of <code>ssx-ldap-search-timeout</code> takes the precedence. Default: 5 seconds For configuring the alias disable time, see “Disabling Basic Authentication Alias” .	LDAP

Default Values for Active Directory Parameters

The following table lists the default values for Active Directory parameters.

Parameter	Default Value
ldap-person-objectclass	top,person,organizationalPerson,user
ldap-group-objectclass	top,group
ldap-person-grp-attr	memberOf
ldap-group-prs-attr	member

Parameter	Default Value
ldap-passwd-field	unicodePwd
ldap-person-property-attr	cn:r,displayName:w,description:w, mail:w,telephoneNumber:w, physicalDeliveryOfficeName:w,givenName:w, sn:w,homeDirectory:r,ou:w
ldap-group-property-attr	cn:r,description:w

Default Values for SunOne Parameters

The following table lists the default values for SunOne parameters.

Parameter	Default Value
ldap-person-objectclass	top,person,organizationalperson,inetorgperson
ldap-group-objectclass	top,groupofuniquenames
ldap-person-grp-attr	./.
ldap-group-prs-attr	uniqueMember
ldap-passwd-field	userPassword
ldap-person-property-attr	uid:r,cn:r,sn:w,title:w, description:w,telephoneNumber:w, seeAlso:w,postalAddress:w, postalCode:w,postOfficeBox:w
ldap-group-property-attr	cn:r,ou:w,o:w,owner:w, description:w,businessCategory:w,seeAlso:w

Default Values for OpenLdap Parameters

The following table lists the default values for OpenLdap parameters.

Parameter	Default Value
<code>ldap-person-objectclass</code>	<code>top,person</code>
<code>ldap-group-objectclass</code>	<code>top, groupOfUniqueNames</code>
<code>ldap-person-grp-attr</code>	<code>./.</code>
<code>ldap-group-prs-attr</code>	<code>uniqueMember</code>
<code>ldap-passwd-field</code>	<code>userPassword</code>
<code>ldap-person-property-attr</code>	<code>cn:r,sn:w,description:w,telephoneNumber:w, seeAlso:w</code>
<code>ldap-group-property-attr</code>	<code>cn:r,ou:w,o:w,owner:w, description:w,businessCategory:w, seeAlso:w</code>

Basic Authentication Alias

The basic authentication aliases are the authentication systems configured in the `basicauth.cfg` file. If you specify any of these aliases in an authenticator ACL, all users listed under the specified alias will be authorized to use the ACL protected Broker objects.

Note: If you make any change in the authentication system in the `basicauth.cfg` file, make sure you update the authenticator ACL accordingly.

Disabling Basic Authentication Alias

When the authentication systems are slow or down, a client call might time-out if the user authentication takes more than 30 seconds.

Set the LDAP search time-out parameter using the `ssx-ldap-search-timeout` configuration parameter in the `basicauth.cfg` file or/and the `SSX_LDAP_TIMEOUT` environment variable to avoid client call time-out.

If webMethods Broker fails to connect to an alias even after three attempts, webMethods Broker disables an unresponsive alias as described in the following process to speed up the basic authentication process.

1. Disables an unresponsive alias for a period specified in the `alias-min-disable-time` parameter.
2. Enables the disabled alias and attempts re-authentication after the time specified in the `alias-min-disable-time` parameter elapses.
3. Doubles the alias disable time if re-authentication fails and attempts re-authentication in regular intervals at the end of the alias disable time until a value assigned to the `alias-max-disable-time` parameter is reached.
4. Attempts to re-authenticate the disabled alias in regular intervals specified by the `alias-max-disable-time` parameter after the alias disable time reaches the value specified in the `alias-max-disable-time` parameter.

For `alias-min-disable-time` and `alias-max-disable-time` parameter descriptions, see [“Basic Authentication Configuration Parameters”](#).

Configuration example for disabling the basic authentication alias

In the basic authentication configuration (`basicauth.cfg`) file, consider `alias-min-disable-time=10` and `alias-max-disable-time=180`.

If webMethods Broker fails to connect to an alias after three attempts, webMethods Broker does the following:

1. Suspends the unresponsive alias for 10 seconds (`alias-min-disable-time=10`).
2. Enables the disabled alias after 10 seconds and attempts re-authentication.
3. Doubles the alias disable time if re-authentication fails, and attempts re-authentication after the alias disable time elapses. This step repeats until the alias disable time reaches 180 seconds (`alias-max-disable-time=180`).
4. Continues to disable the alias for 180 if the alias is unresponsive and attempts re-authentication every 180 seconds (`alias-max-disable-time`) until the alias is responsive.

Configuring Basic Authentication Identity for a Broker Server

Use the following procedure to configure the basic authentication identity for a Broker Server. This is the outbound identity that the Broker Server uses to identify itself to other Brokers in a territory or a cluster, or to a Broker at the other end of the gateway. Configure the basic authentication identity for a Broker Server with ACL secured territory or cluster, and/or gateway.

To assign a basic authentication identity to a Broker Server

1. In My webMethods: **Messaging > Broker Servers > Servers**.
2. In the **Broker Servers** list, click the server on which to assign the basic authentication identity. If the server does not appear in the list, use the **Search** tab to locate it.
3. In the Broker Server Details page, click the **Server Identity** tab.

4. Click the **Basic Identity** tab.
5. Click **Change Configuration**.
6. Type the user name and password in the designated fields.
7. Click **Apply**.
8. Check that the **Status** field reads: `Enabled`

Disabling the Basic Authentication Identity of a Broker Server

For security reasons, you can disable the identity of a Broker Server. Once you disable the identity of a Broker Server, that Broker Server cannot identify itself to other Brokers in a territory or cluster or to a Broker at the other end of gateway.

To disable the basic authentication identity for a Broker Server

1. In My webMethods: **Messaging > Broker Servers > Servers**.
2. In the **Broker Servers List**, click the server with the basic authentication identity you want to disable. If the server does not appear in the list, use the **Search** tab to locate it.
3. Click the **Server Identity** tab.
4. Click the **Basic Identity** tab.
5. Click **Disable Basic Identity**.

Securing Broker Server Using SSL

Broker supports SSL to safeguard your Broker Servers and clients. SSL allows secure sharing of data between clients and Broker Servers, and between Brokers in territories, clusters, or gateways.

Broker Security Model and OpenSSL

SSL on the Broker is built using the OpenSSL library, an open-source implementation of the SSL/TLS protocol and general source cryptographic library. OpenSSL includes a command-line tool for SSL operations such as managing certificate files. You can use the OpenSSL command-line tool (see [“Managing Certificate Files with OpenSSL” on page 581](#)) along with My webMethods when configuring and managing SSL for Broker.

Assigning SSL Identities for a Broker Server

An SSL identity is composed of a user's SSL signed certificate and the trusted root of the certificate issuer (or authenticator). Information about an identity is stored in two certificate files: a *keystore* containing a private key/signed certificate pair (see [“Keystore](#)

File” on page 312), and the *truststore* containing the trusted root (see “Truststore File” on page 314). The SSL identity must be presented whenever:

- An attempt is made to connect to the Broker Server SSL port.
- A request is made to access Broker Server administrative functions or a Broker component where an ACL has been configured.

The procedures for creating and managing SSL identities are covered in “Creating Keystores and Truststores” on page 318.

One-Way and Two-Way SSL Authentication

The Broker security model allows you implement one-way or two-way SSL authentication between the Broker Server and a client.

- **One-way SSL authentication.** A client with truststore but without keystore. If you configure the client for one-way SSL authentication, you establish a much higher level of security than with a non-SSL connection. In one-way authentication, the identity of the Broker Server is authenticated by the client, and must be guaranteed through the Broker Server’s SSL certificate before a connection is made.

You can also configure basic authentication with one-way SSL authentication. For more information, see “Securing Broker Server Using Basic Authentication Over SSL” on page 324.

- **Two-way SSL authentication.** A client with both keystore and truststore. If you configure the client for two-way SSL authentication so that both the Broker Server and the client connecting to the Broker Server must be SSL authenticated, you can fully implement the Broker security model. You will be able to configure ACLs to protect data and access to Broker administrative functions. With two-way authentication, each client must be associated with a signed digital certificate in order to establish an SSL connection.

SSL Authentication and Broker Port Usage

Depending on the client's SSL configuration, clients connect to Broker Server through one of the three Broker Server ports available for client connections.

The base port of the Broker Server is used for non-SSL connections. The port that is numbered one less than the base port is used when the client is configured for one-way SSL authentication. The port numbered two less than the base port is used when the client is configured for two-way SSL authentication.

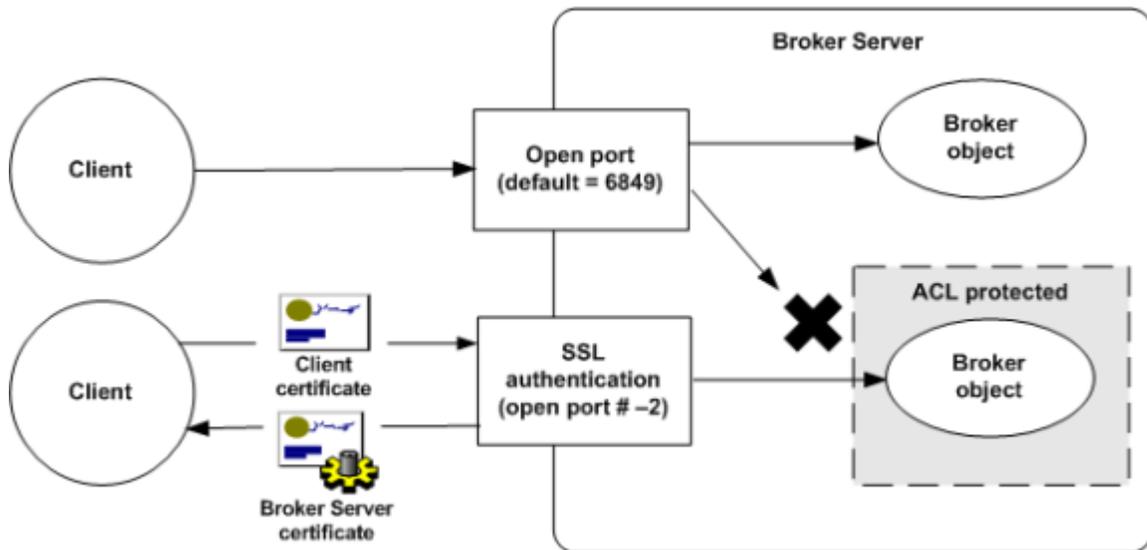
For example, if the default port 6849 is the base port of the Broker Server, port 6849 is used for non-SSL connections, port 6848 is used for one-way SSL connections, and the port 6847 is used for two-way SSL connections.

Broker Server Identity and SSL

The Broker will start enforcing ACLs on protected Broker objects when the Broker Server has established an identity for itself. At this point, anonymous clients are no longer able to access ACL-protected Broker objects.

The following figure illustrates how client-to-Broker Server connections work when the Broker Server has an identity.

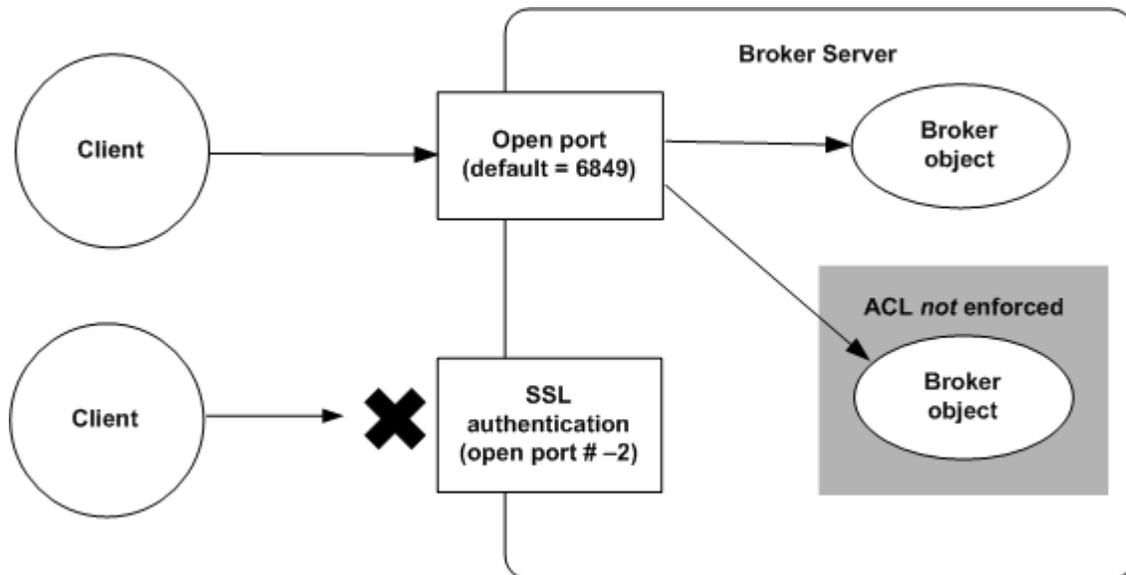
Client-to-Broker connections: Broker Server SSL identity configured and enabled



When a Broker Server and client authenticate each other's SSL identities, a connection is made through the (two-way) SSL port. That port's number is two below that of the Broker Server's assigned port. Under these circumstances, a client can access any Broker object protected by an ACL, so long as that client's identity is listed in the object's ACL.

Clients that do not present an SSL identity to the Broker Server for authentication (through the non-SSL port) cannot access Broker objects protected by an ACL (illustrated by the crosses).

The following figure illustrates how client-to-Broker Server connections work when the Broker Server does not have an SSL identity (or its SSL identity is disabled).

Client-to-Broker connections: no Broker Server SSL identity

If the Broker Server does not have an SSL identity (or the identity is disabled), ACLs are not enforced. Any client that can connect to the Broker Server can access any of its ACL-protected Broker objects.

For the SSL port to be used, the Broker Server must have an identity. Thus, any attempt to connect to the Broker Server by first authenticating its identity will fail (illustrated by the crosses). Only non-SSL connections will be successful.

Certificate Files

webMethods Broker stores its SSL certificates in keystore files and the trusted roots for the certificates in truststore files.

Note: If you are migrating from 6.5.x or previous version, you'll need to convert your existing certificate file into keystore and truststore. See the [“Converting Certificate Files” on page 351](#) for details.

Keystore File

webMethods Broker uses a special file called a *keystore* to store each Broker component's SSL certificate.

A keystore file contains one key pair, composed of a client's private key and signed certificate for its public key. The certificate contains the client's distinguished name (DN). This DN is usually associated with an alias in the Broker user interface and referred to there as the user name.

The keystore should be strongly protected with a password, and stored (either on the file system or elsewhere) so that it is accessible only to administrators.

Keystore File Formats

Before creating a keystore file to store a certificate, you need to know which file format to use.

- You can use the PKCS#12 certificate format for the keystore for any Broker component: the Broker Server, the Broker admin component, or a client.

PKCS#12 is a commonly used, standardized, certificate file format that provides a high degree of portability.

- You can use PEM (Privacy Enhanced Mail) format for the keystore for the Broker Server. You cannot use this format for the keystore for the Broker admin component or a client.

PEM is a base-64 encoded data format used for text-based communications; it provides the ability to encrypt the data before encoding it.

Note: Previous versions of webMethods Broker (version 6.5.2 and earlier) use the Spyrus certificate file format for keystores. The Spyrus format is not compatible with the keystore formats used in this version of Broker. For information about how to upgrade version 6.5.2 and earlier Broker keystores to keystores for the current version of Broker, see [“Converting Certificate Files” on page 351](#).

OpenSSL, the open-source implementation of SSL that Broker uses, provides its own set of certificate management tools. These tools are supplied with Broker. You can use the OpenSSL command-line tool to manage keystores in the PKCS#12 and PEM formats, and any other certificate management tools that work with these formats. For more information, see [“Managing Certificate Files with OpenSSL” on page 581](#).

Certificate Chains

It is possible for a single-entity certificate to have a list of signing certificates leading up to the original, self-signed root certificate. Such a certificate list, with each certificate signed by the next, is termed a certificate chain.

With a certificate chain, it is necessary to validate each subsequent signature in the list until a trusted CA certificate is reached. For Java and JMS clients that use SSL with Broker (Entrust certificate format), you must include the entire chain of certificates in a keystore and truststore. (For Broker clients using OpenSSL, that restriction is not necessary.) Also, any root CA certificates in use by clients must be in the Broker Server's truststore.

Installing a Certificate

To install a signed certificate, you generate a certificate request using a certificate editing tool and send the request to the CA (the same issuing entity as specified on the trusted root). After receiving the X.509-compliant signed certificate back from the CA, you again use the certificate editor and install the certificate into the keystore.

Setting up a keystore file is one of the first steps needed for configuring a Broker component for SSL. For step-by-step instructions on configuring a keystore, see [“Configuring SSL for Broker Server”](#) on page 317 and [“Creating Keystores and Truststores”](#) on page 318.

Truststore File

webMethods Broker uses a file called a *truststore* to store trusted roots. A *trusted root* is a certificate belonging to a Certification Authority (CA). The trusted root contains the CA's public key and the distinguished name of the issuing entity (referred to as the *authenticator name* in the Broker user interface). A truststore may contain the trusted roots for an entire certificate chain.

Truststores should be stored in a safe place on the file system to prevent unauthorized access and tampering.

Truststore File Formats

The allowable formats for truststores are different than those for keystores; PKCS#12 cannot be used as the format for truststores because it cannot hold multiple certificates. The following table lists the Broker components that must be assigned SSL identities and their allowable truststore formats.

Component	Truststore File Format
Broker Server	PEM
Broker admin component	JKS
Client (JMS)	JKS
Client (Java)	JKS

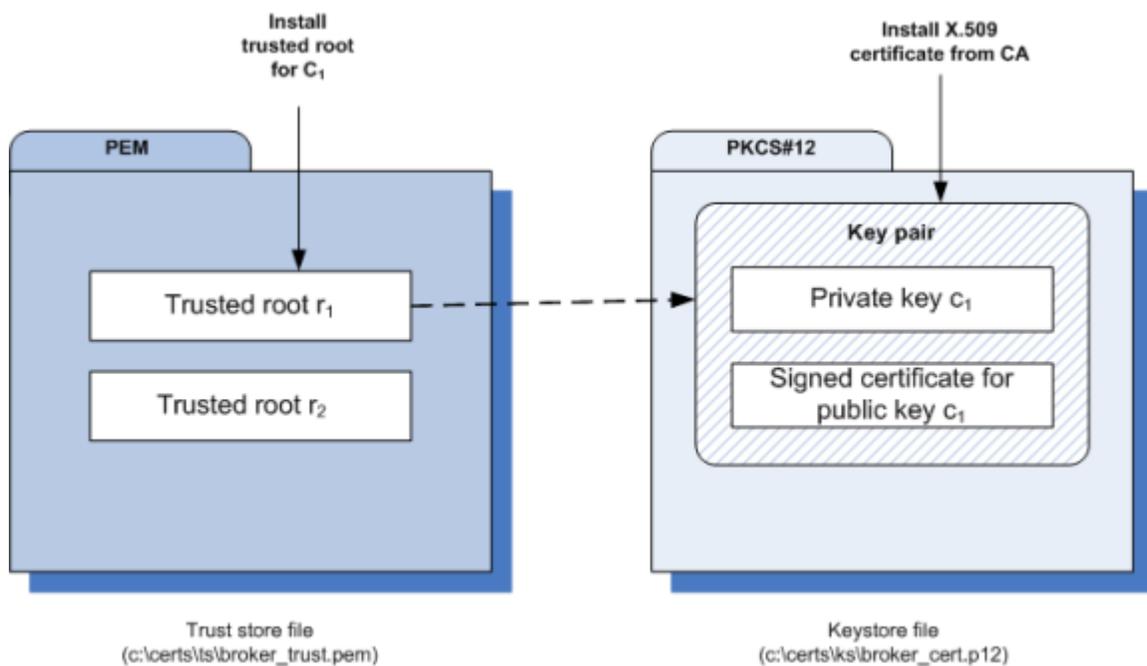
- The format for the Broker Server is PEM. Software AG recommends using the OpenSSL certificate editing tool to manage keystores in this format. You can also use other certificate management tools that work with the PEM format.
- The format of truststores for Java client programs, JMS client programs, and the Broker admin component is Java keystore (JKS). You create and manage JKS truststores at the command line using keytool, Oracle's key and certificate management tool.
- You manage the truststore for C# clients through the Microsoft Management Console (MMC). Instructions for managing keystores and truststores for C# client programs are provided in the *webMethods Broker Messaging Programmer's Guide*.

Setting up a truststore file is one of the first steps in configuring SSL for a Broker component. For information about configuring a truststore, see [“Managing Certificate Files with OpenSSL”](#) on page 581.

How Broker Uses a Keystore and Truststore

For a Broker component to be SSL authenticated, it must have a valid, authorized X.509 certificate installed in a keystore file and the trusted root for the CA that issued the certificate installed in a truststore file. The following figure illustrates these requirements and the relationship between the two files.

Example Truststore File and Keystore File Showing Relationship



As shown in the above figure, the same truststore file can contain multiple trusted roots. These trusted roots may be associated with numerous keystore files. A keystore file contains the key pair for a single Broker component, and can contain the entire certificate chain required for a Broker component's authentication.

Protecting Keystore and Truststore Files

Keystore and truststore files exist within the file system of your operating system, and since these are critically important files, you want to maintain them in a secure directory path. If either of the certificate files for a Broker component cannot be located, SSL authentication will be disabled and the connection may be switched to an open port. It is recommended that only users serving as My webMethods or Broker administrators have access to these certificate files.

Using SSL Encryption

You can encrypt SSL authenticated connections to the Broker Server as well as SSL authorized communications at the client group ACL level. By default, encryption is enabled.

Encryption is useful when the network traffic carries sensitive data, and in cases where the network may be susceptible to packet sniffing and other security breaches.

You can enable encryption for Broker clients at the client group level without setting up ACLs, which allows you to create system configurations where anonymous clients can make secure connections to a Broker.

Although encryption provides the highest level of security, there are associated performance costs. Therefore, you should understand the trade-offs between the security needs of your organization versus maintaining an acceptable level of system performance before employing encryption as part of your Broker security solution.

OpenSSL Cryptography

Cipher strings indicate what kind of cryptographic algorithm is used. Common cipher strings to use with webMethods Broker are as follows:

Cipher String	Description
HIGH:eNULL:@STRENGTH (default)	Uses algorithms with key lengths larger than 128 bits, some cipher suites with 128-bit keys, and plain text; algorithms are then sorted by strength and the strongest is used.
ALL	Uses all cipher suites, but does not allow unencrypted data.
ALL:eNULL:!EXP	Uses all cipher suites except export ciphers (that is, uses stronger encryption).
Note:	The complete list of OpenSSL cipher strings is available on "http://www.openssl.org/docs/apps/ciphers.html#CIPHER_STRINGS" .

Only advanced users should change the cipher suites for SSL authentication from the default settings to ensure authentication is not incorrectly configured.

If you select a cipher suite that is not supported, you will get error during the following operations:

- Setting a cipher suite that OpenSSL does not support.

- Performing SSL handshake when the cipher suite is not supported because of another run-time setting. For example, if the protocol is limited to TLSv1.1, some of the ciphers are restricted even though the OpenSSL library supports it.

If you change the cipher suite, Software AG recommends you to test the connectivity of the clients that are using SSL to connect to Broker.

Configuring SSL for Broker Server

Following is a high-level summary of the steps required to configure SSL for webMethods Broker. The detailed procedures for implementing the steps are covered in the sections after the high-level summary.

1. Create the keystores.

The Broker Server, Broker user interface, and each Broker Server client must have access to the digital certificates needed to authenticate their connections. The SSL certificates for each of these components resides in a keystore, a specially formatted files protected by a password. Keystores are located on the Broker Server Host, on the local machine hosting the browser that connects to the Broker user interface, and on machines where client applications reside.

The procedure for creating and configuring a keystore is covered in [“Creating Keystores and Truststores” on page 318](#) and [“Managing Certificate Files” on page 582](#).

2. Create the truststores.

The Broker Server, Broker user interface, and each Broker Server client must have access to the trusted roots corresponding to the digital certificates needed to authenticate their connections. Trusted roots reside in truststores, and are located on the Broker Server Host, on the local machine hosting the browser that connects to the Broker user interface, and on machines where client applications reside.

The procedures for creating and configuring truststores are covered in [“Managing Certificate Files with OpenSSL” on page 581](#).

You should evaluate how many truststores your Broker system needs. Keeping one truststore for all Broker components may suffice, but the increased security gained from having multiple truststore may better serve your needs.

3. Configure the Broker Server for SSL.

After you have configured a keystore and truststore entry for a Broker Server, you assign it an identity using My webMethods. This procedure is described in [“Configuring an SSL Identity for a Broker Server ” on page 320](#).

4. Configure the Broker user interface component for SSL.

For Broker SSL to work, the Broker Server must authenticate the SSL identity of the Broker user interface component. Thus, you need to assign the Broker user interface an identity.

This procedure is described in [“Configuring an SSL Identity for the Broker User Interface Component” on page 348](#).

5. Configure each client to enable SSL.

You use certificate editing tools to create and manage the keystores and truststores for clients, and use the client applications to assign the SSL identities and perform any additional SSL configuration. These tools must work with the appropriate certificate format required for each Broker component (see [“Keystore File Formats” on page 313](#) and [“Truststore File Formats” on page 314](#)).

You do not use My webMethods or a webMethods command-line utility to configure clients for SSL. For additional information, see [“Configuring SSL for Clients” on page 322](#).

Important: The use of SSL authentication is determined by whether a client-side keystore is passed to the Broker Server upon connection. You do not configure SSL authentication through the Broker user interface.

Creating Keystores and Truststores

Broker does not supply its own keystore or truststore files. You must create your own, and configure and manage them.

The instructions for creating and configuring keystores and truststores using the OpenSSL command line tool are described in [“Managing Certificate Files with OpenSSL” on page 581](#). Note, however, that you can create and configure Broker keystores and truststores using any certificate editing tool that works with PKCS#12 or PEM certificate formats.

Loading Keystores and Truststores into My webMethods

After you create the keystores and truststore(s) and move them to a secure location in your file system, you load the keystores and truststores to My webMethods. You can then manage SSL identities through the same user interface as your other My webMethods applications.

If you modify a keystore or truststore, you need to reload it to My webMethods (for details, see [“Modifying Keystores and Truststores” on page 320](#)).

Important: Make sure to secure your keystore file password, and that it is available before you create and configure that certificate file.

Loading Keystores from the Machine on Which Broker is Administered

Use the following procedure for loading keystores on a machine whose browser you use to administer Broker.

Load a keystore (from the file system of the same machine used to administer Broker)

1. In My webMethods: **Messaging > Settings > SSL Keystore**.

2. Click the **Upload Keystore** tab.
3. Type an alias for the keystore in the **Keystore Name** box.
4. Click **Browse** and locate the keystore file.
5. Click **Upload**.

Loading Keystores on the Machine Hosting My webMethods Server

Use the following procedure for loading keystores on the machine hosting My webMethods Server.

To load a keystore (from the file system of the machine hosting My webMethods Server)

1. In My webMethods: **Messaging > Settings > SSL Keystore**.
2. Click the **Add Keystore** tab.
3. Type an alias for the keystore in the **Keystore Name** box.
4. Type the full path name of the keystore in the **Keystore File** box.

Note: You cannot browse to the keystore if it is located on the My webMethods Server machine; you must know its full-path name.

5. Click **Add**.

Loading Truststores from the Machine on Which Broker is Administered

Use the following procedure for loading truststores on a machine whose browser you use to administer Broker.

To load a truststore (from the file system of the same machine used to administer Broker)

1. In My webMethods: **Messaging > Settings > SSL Truststore**.
2. Click the **Upload Truststore** tab.
3. Type an alias for the truststore in the **Truststore Name** box.
4. Click **Browse** and locate the truststore file.
5. Click **Upload**.

Loading Truststores on the Machine Hosting My webMethods Server

Use the following procedure for loading truststores on the machine hosting My webMethods Server.

To load a truststore (from the file system of the machine hosting My webMethods Server)

1. In My webMethods: **Messaging > Settings > SSL Truststore**.
2. Click the **Add Truststore** tab.
3. Type an alias for the truststore in the **Truststore Name** box.

4. Type the full path name of the truststore in the **Truststore File** box.

Note: You cannot browse to the truststore if it is located on the My webMethods Server machine; you must know its full-path name.

5. Click **Add**.

Modifying Keystores and Truststores

You may need to modify a keystore or truststore that has already been loaded into the Broker user interface. To do so, you do the following:

- Download the keystore or truststore to your local machine.
- Modify the file using OpenSSL or your certificate editor.
- Reload the modified certificate file back into the Broker user interface.

To download, modify, and reload a keystore (truststore)

1. In My webMethods: **Messaging > Settings > SSL Keystore (SSL Truststore)**.
2. Click **Download Keystore (Download Truststore)**.
3. On the Download Keystore (Download Truststore) panel, click **Keystore Name (Truststore Name)** and select the keystore (truststore) from the list.
4. Click **Download**.
5. Save the keystore (truststore) file.
6. Edit the keystore (truststore) using a certificate editor.

For more information, see [“Managing Certificate Files with OpenSSL” on page 581](#).

7. Reload the keystore (truststore) into the Broker user interface.

Configuring an SSL Identity for a Broker Server

You set either an SSL Identity for the Broker Server in My webMethods. To configure an SSL identity for a Broker Server, you open its keystore, select the DN to use as the Broker Server's identity, and assign that DN to the Broker Server.

Important: Make sure you have the file location and password of the Broker Server's keystore before starting this procedure.

To assign an SSL identity to the Broker Server

1. In My webMethods: **Messaging > Broker Servers > Servers**.
2. In the **Broker Servers List**, click the server on which to assign an identity. If the server does not appear in the list, use the **Search** tab to locate it.

3. In the Broker Server Details page, click the **Server Identity** tab.

4. Click the **SSL** tab. The **Status** should read:

Secure Sockets Layer needs to be configured

5. Click **Change Configuration**.

Note: When you configure SSL for the first time, the saved and active SSL configurations are the same, so you need not restart the Broker Server. When you change the SSL configuration, the **Continue Using Current Configuration** field displays a yellow icon with a status set to 'no'. You must restart the Broker Server to use the new SSL setting. When you set, change, or disable SSL for Broker Server, if the SSL settings are correct, the **Continue Using Current Configuration** field displays a green icon with a status set to 'yes'.

6. On the Change Broker SSL Settings page, do one of the following:

- If the keystore containing the identity is on a different machine than the one hosting Broker Server:
 - i. Click the **Remote Keystore** tab.
 - ii. Type the full path name of the **Remote SSL Keystore** in the box.
 - iii. Type the full path name of the **Remote SSL Truststore** in the box.
- If the keystore containing the identity is on the same machine as the one hosting Broker Server:
 - i. Click the **Local Keystore** tab.
 - ii. Click **Local SSL Keystore** and select a keystore name from the list.
 - iii. Click **Local SSL Truststore** and select a truststore name from the list.

7. Click **Keystore Type** and select the file format of the keystore.

8. Click **Truststore Type** and select the file format of the truststore.

9. The following options are available for advanced users:

- a. Select an **SSL Protocol** to use for authentication (the default is **All**).
- b. Enter the cryptographic **Cipher Suite** to use for authentication (the default is **HIGH:eNULL:@STRENGTH**).
- c. Reconfigure the maximum **Verification Depth** allowable for a certificate chain (the default is **9**).

10. Type the **Password** to the keystore file.

11. Click **Get User Name** to retrieve the user name (user DN) from the keystore.

12. Click **Save**.

13. Click the **Identity** tab.

- Click the **SSL** tab on the Broker Server Details page for the Broker Server that you just configured. Information about the SSL configuration is displayed, and the **Status** should read:

```
Secure Sockets Layer is configured and working
```

Disabling the SSL Identity for a Broker Server

You can disable the SSL identity for a Broker Server. After you disable the SSL configuration, all connections are made to that Broker Server through the base port (non-SSL port).

To disable the SSL identity for a Broker Server

- In My webMethods: **Messaging > Broker Servers > Servers**.
- In the **Broker Servers List**, click the server with the SSL connection you want to disable. If the server does not appear in the list, use the **Search** tab to locate it.
- Click the **Identity** tab.
- Click the **SSL** tab.
- Click **Disable SSL**.

Note: Once you disable the SSL identity of a Broker Server, all of the Broker Server's objects protected by ACLs become accessible to any clients; that is, the ACLs are not enforced.

Configuring SSL for Clients

You use OpenSSL or your certificate editing tool to create and manage the keystores and truststores for clients. However, you must use the client application to establish its SSL identity. You do not use My webMethods to configure SSL for Broker clients.

If you want to add a new client and enable SSL authentication for that client, and the trusted root of the client is different than that of the Broker Server, make sure to add the client's trusted root to the Broker Server truststore. If you do not, the client cannot be authenticated.

You can find information about configuring clients for SSL in the following locations:

For these clients...	Look in...
Adapters	The appropriate adapter documentation, in the section on configuring the adapter.
Client applications	The appropriate API manual for the client.

For these clients...	Look in...
Integration Servers	<i>webMethods Integration Server Administrator's Guide</i> .
Command-line utilities	" webMethods Broker Command-Line Utilities " on page 525 and the "JMSAdmin Command Reference" chapter in the <i>webMethods Broker Messaging Programmer's Guide</i> .
Note:	If an SSL connection between Broker and a client identified by an SSL DN is broken, then that client must reconnect with the same SSL identity used in its previous connection.

Securing Broker Server Using CRL

A *certificate revocation list (CRL)* is a list of certificates (or more specifically, a list of serial numbers for certificates) that are revoked by a certification authority (CA) and are no longer valid. A CRL provides increased security because it provides a mechanism for protecting against certificates that are compromised. The implementation of SSL with Broker Server supports the checking of certificates against the CRL.

Note: Broker supports certificate revocation checking only on the server side.

For more information, see "[Configuring Certificate Revocation Checking](#)" on [page 323](#).

Configuring Certificate Revocation Checking

You configure certificate revocation checking for Broker Server by using CRLs.

To configure certificate revocation checking by using CRLs

1. In My webMethods: **Messaging > Broker Servers > Servers**.
2. Configure SSL. For steps to configure SSL, see section "[Configuring SSL for Broker Server](#)" on [page 317](#).
3. In the **Broker Servers List**, click the Broker Server on which to configure revocation checking. If the Broker Server does not appear in the list, use the **Search** tab to locate it.
4. Click the **SSL** tab and then click the **Change Configuration** button.
5. Click **Local Keystore** tab.
6. In the **Certificate Revocation List File** field, type the full path name of the CRL file.

Note: You must place the CRL file in a location accessible to the Broker Server.

7. In the **Certificate Revocation List File Type** field, select either PEM or DER file format depending on the CRL type.
8. Click **Apply**.
9. Repeat steps 5 through 8 to configure certificate checking for remote keystore by clicking the **Remote Keystore** tab in step 5.

Configuring CRL Checking For the Entire Certificate Chain

To enable CRL checking for the entire certificate chain, set the `check-crl-all` configuration parameter in the `Software AG_directory\Broker\data\awbrokersversion\default\awbroker.cfg` file as shown below:

```
check-crl-all=1
```

By default, the CRL checking is not done for the entire certificate chain.

Securing Broker Server Using Basic Authentication Over SSL

Broker security works with basic authentication over one-way SSL authentication.

Whether you implement basic authentication with one-way SSL authentication or two-way SSL authentication, a client authenticates the identity of the Broker Server using the Broker Server's SSL certificate, and the client connection is made using basic authentication. You can optionally encrypt this connection through settings on the client for higher security.

In order to use SSL encryption with basic authentication, you only need to specify a truststore and not a keystore.

Clients can access any Broker object protected by an ACL, as long as that client's basic authentication identity is listed in the object's ACL.

Securing Broker Server Using FIPS

Federal Information Processing Standards (FIPS) are a set of standards that describe document processing, encryption algorithms and other information technology standards for use within non-military government agencies and by government contractors and vendors who work with the agencies.

webMethods Broker uses an embedded FIPS 140-2-validated cryptographic module (Certificate #1051) running on all platforms on which webMethods Broker is supported, per FIPS 140-2 Implementation Guidance section G.5.

You can start Broker Server either with FIPS enabled or disabled. When webMethods Broker starts, it checks the Broker configuration for FIPS mode configuration and records it in the Broker log file. For more information about enabling FIPS mode, see [“Starting Broker Server in FIPS mode” on page 325](#).

Note: A FIPS enabled Broker Server might not be able to communicate with a Broker Server on which FIPS is not supported. Compatibility issues can occur between two Brokers if you enable FIPS on only one Broker while SSL is configured on both.

Starting Broker Server in FIPS mode

Use the following procedure to start the Broker Server in FIPS mode.

To start Broker Server in FIPS mode

1. Stop Broker Server.
2. Locate the Broker Server's configuration file (`awbroker.cfg`) in the data directory and make a backup copy.
3. Open the configuration file in a text editor.
4. If the value of `enable-fips-mode` is set to 0 (default), change it to 1.
5. Start Broker Server.

Access Control Lists

You configure ACLs through the Broker user interface and enable them for the Broker object for which you want to specify permissions.

Before configuring ACLs, you must do at least one of the following:

- For basic authentication, configure the user names of the clients you want to grant authorization.
- For SSL authentication, configure the keystores containing trusted roots for the clients you want to grant authorization.

For a client to access a Broker object protected by an ACL, the client must have a basic authentication identity or an SSL identity listed in the corresponding ACL.

Anonymous clients (those clients connecting to the Broker Server through the non-SSL port, or the one-way SSL authentication port without the basic authentication credentials) cannot access Broker objects protected by ACLs.

The procedures for configuring and managing ACLs are covered in [“Configuring Access Control Lists” on page 333](#).

Authenticator vs. User ACLs

For authorization, you can attach user ACL and authenticator ACL to a Broker component. You can attach one or both of these ACLs to specify user rights; however, if

you use both ACLs, then the user seeking access must present an identity that satisfies the conditions of both ACLs in order to be granted authorization.

To populate an ACL, you use My webMethods to:

- Type the basic authentication user names and aliases manually.
- Select the SSL DNs associated with a specified keystore and truststore or type the DNs manually.

How User Rights Are Granted

Users are granted authorization by ACLs according to the following rules:

1. If you do not specify any ACL (user ACL or authenticator ACL), all users are granted full permissions.
2. If you specify only the authenticator ACL:
 - For basic authentication, a user will be granted access if one the alias configured in basicauth.cfg file authenticates the user, and this alias is listed in the authenticator ACL.
 - For SSL, a user will be granted access if the SSL certificate of the user is certified by one of the authenticators listed in the authenticator ACL.
3. If you specify only the user ACL:
 - For basic authentication, a user will be granted access only if the user's name is listed in the user ACL.
 - For SSL, a user will be granted access only if the user's DN is listed in the user ACL.
4. If you specify both user ACL and authenticator ACL:
 - For basic authentication, only users with both user name and authenticator alias listed in the respective ACLs will be granted access.
 - For SSL, only users with both the user's DN and its SSL certification authority listed in the respective ACLs will be granted access.

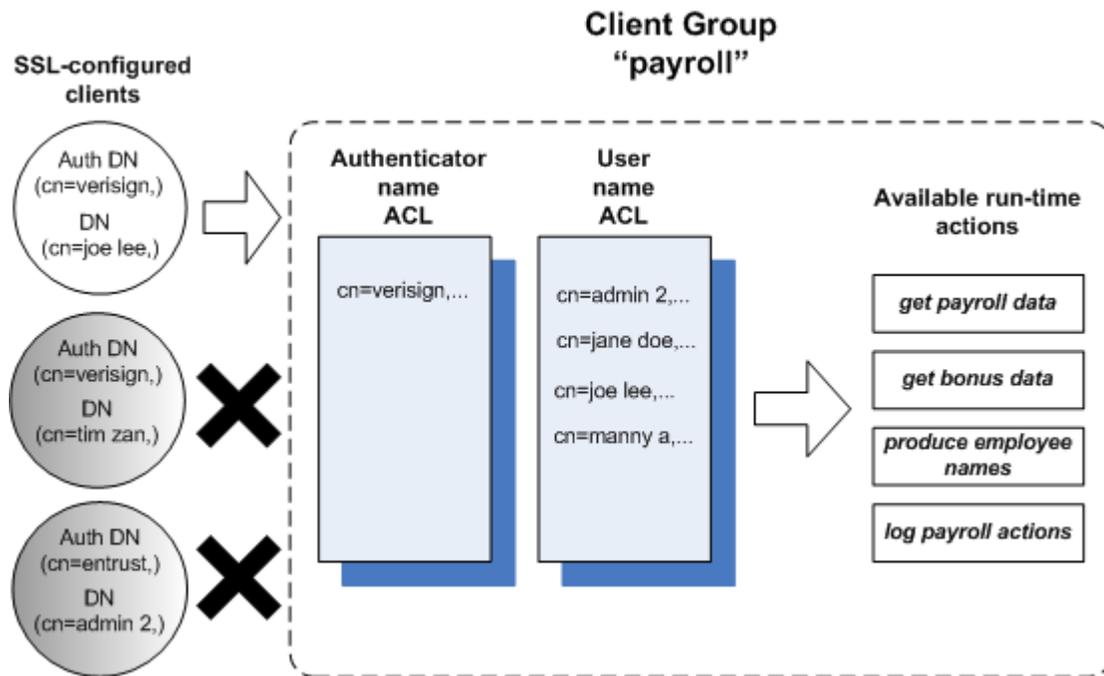
Client Group ACLs

You use client group ACLs to identify the clients that have permission to join a particular client group. Once a client becomes a member of the group, it can produce or consume the document types in the can-publish and can-subscribe lists, and various logs controlled by the client group (the run-time actions available to the client group).

Note: It is also possible to configure the granting of user rights on a per message basis. For more information, see [“Using Access Labels” on page 585](#).

The following figure illustrates how client group ACLs work.

How a Client Group Access Control List Works with SSL Clients



In the figure, three SSL-configured clients are shown attempting to join the client group "payroll," which has access to company payroll information. Each of these clients is identified both by a unique user DN and the DN of its certification authority.

Two ACLs are attached to this client group:

- An authenticator name ACL specifying that only clients with certificates signed by VeriSign are granted permission
- A user name ACL listing the four user DNs that are granted permission

Therefore, only clients with DNs satisfying both conditions (the top-most client) can join the client group "payroll." Other clients attempting to join this client group to access payroll information (such as the second and third clients from the top) are denied permission.

The "available run-time actions" shown in the figure correspond to the document types selected for the can-publish, can-subscribe, log-publish, and log-acknowledge lists. For example, a client joining "payroll" automatically becomes a subscriber to the document type that summarizes company bonus data; assuming that this document type was selected for the payroll client group's can-subscribe list.

For a step-by-step description of modifying ACLs for client groups, see [“About Configuring Client Group ACLs”](#) on page 337. For additional information about client groups, refer to [“Managing Client Groups”](#) on page 171.

admin Client Group ACLs

The admin client group is a system-defined client group with full access permissions for a given Broker. Each Broker has one admin client group, and you can enable ACLs for that client group, the same as with any other client group.

A client belonging to the admin client group has administrative access to a given Broker. Members of the admin client group can modify any Broker setting, and reconfigure the territories or local gateways to which the Broker belongs. As such, the admin client group is a control point for administrative access to a Broker. It is thus important that you use ACLs to secure the admin client group for each Broker as soon as possible.

For instructions on configuring ACLs for the admin client group, refer to [“About Configuring Client Group ACLs”](#) on page 337.

Broker Server ACLs

By attaching ACLs to a Broker Server, it is possible to limit administrative access to a Broker Server and the Brokers it controls.

Clients without ACL access to a Broker Server can only view certain types of information about Broker Server, such as the names of its Brokers, including the default Broker, statistical information, and whether basic authentication or SSL is configured. Clients with ACL access are granted user rights to do the following:

- View the Broker Server's identity
- Create Brokers
- Delete Brokers
- Get and set the following:
 - Basic authentication; clients with administrative access to the Broker Server can turn off basic authentication, which controls the security of the entire Broker system
 - SSL configuration; clients with administrative access to the Broker Server can turn off SSL, which controls the security of the entire Broker system
 - ACLs
 - License key (contained in the license file)
- Set the host description and logging configuration
- Purge log entries
- Stop the Broker Server

For information about configuring and attaching ACLs to a Broker Server, see [“About Configuring Broker Server ACLs”](#) on page 333.

Territory ACLs

You use territory ACLs to control which Brokers can join a territory. When a territory has an ACL attached, the basic authentication identity or the SSL identity of any Broker attempting to join must be listed on the territory ACL. If not, the Broker will be denied permission.

Brokers within a territory that belong to the same Broker Server share the same basic authentication identity or SSL identity as that of their Broker Server. This is because a Broker Server's identity is copied to all of its Brokers in the territory.

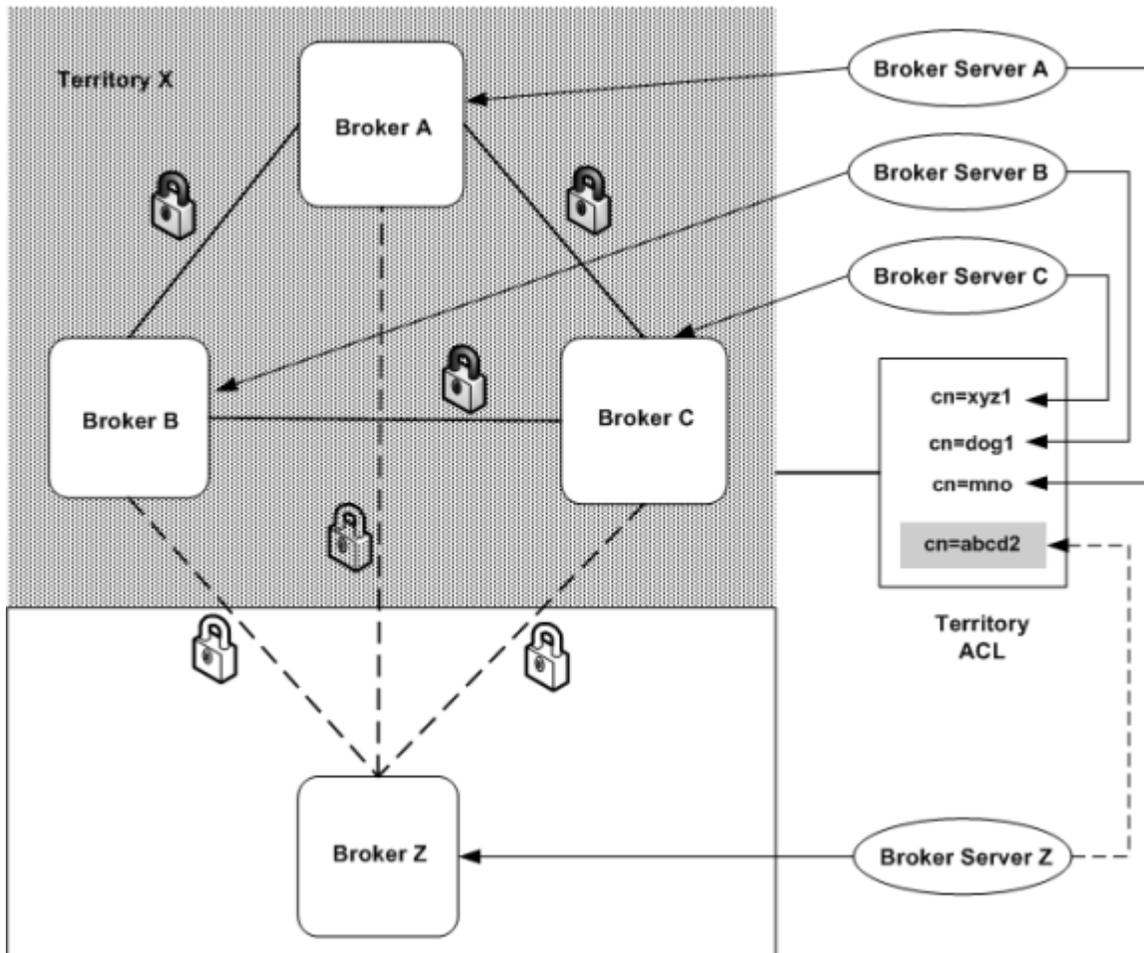
A territory ACL is shared among all the Brokers in the same territory. When you change the territory ACL on a particular Broker, that ACL is automatically propagated to the rest of the territory.

If you plan on opening the membership of the territory to additional Brokers, specifying their identities on a territory ACL is necessary if the enlisting Brokers have identities different than those specified by the ACL. If that is the case:

- Make sure that the Broker Server(s) of the enlisting Brokers have either basic authentication identities or SSL identities.
- Make sure that either basic authentication or SSL is enabled for those Broker Servers.
- Add the basic authentication identities or SSL identities of the Brokers you want to add to the territory ACL.

The following figure illustrates the process of adding a Broker to a territory protected by an ACL:

How a Territory Access Control List Works



In the figure, territory X consists of three Brokers, A, B, and C, that are controlled by their respective Broker Servers A, B, and C. The basic authentication identities or SSL identities for each of the Broker Server's controlling Brokers A, B, and C are listed in the ACL for territory X. For Broker Z (belonging to Broker Server Z) to join territory X, its identity (shaded DN in the ACL), which is the same as that of Broker Server Z, must be added to the territory ACL.

For information about configuring territory ACLs, see [“About Controlling Which Brokers Can Join a Territory”](#) on page 339.

Cluster ACLs

You use cluster ACLs to control which Brokers can join a cluster. When a cluster has an ACL attached, the SSL identity or the basic authentication identity of any Broker attempting to join must be listed on the cluster ACL. If not, the Broker will be denied permission.

Brokers within a cluster that belong to the same Broker Server share the same basic authentication identity or SSL identity as that of their Broker Server. This is because a Broker Server's identity is copied to all of its Brokers in the cluster.

A cluster ACL is shared among all the Brokers in the same cluster. When you change the cluster ACL on a particular Broker, that ACL is automatically propagated to the rest of the cluster.

If you plan on opening the membership of the cluster to additional Brokers, specifying their identities on a cluster ACL is necessary if the enlisting Brokers have identities different than those specified by the ACL. If that is the case:

- Make sure that the Broker Servers of the enlisting Brokers have basic authentication identities or SSL identities.
- Make sure that either basic authentication or SSL is enabled for those Broker Servers.
- Add the basic authentication identities or SSL identities of the Brokers you want to add to the cluster ACL.

For information about configuring cluster ACLs, see [“About Controlling Which Brokers Can Join a Cluster” on page 341](#).

Territory Gateway ACLs

A territory gateway allows a Broker in one territory to retrieve information from Brokers in another territory. You can set up ACLs on a territory gateway so that the Broker requesting information from a territory other than its own must be granted authorization to receive data.

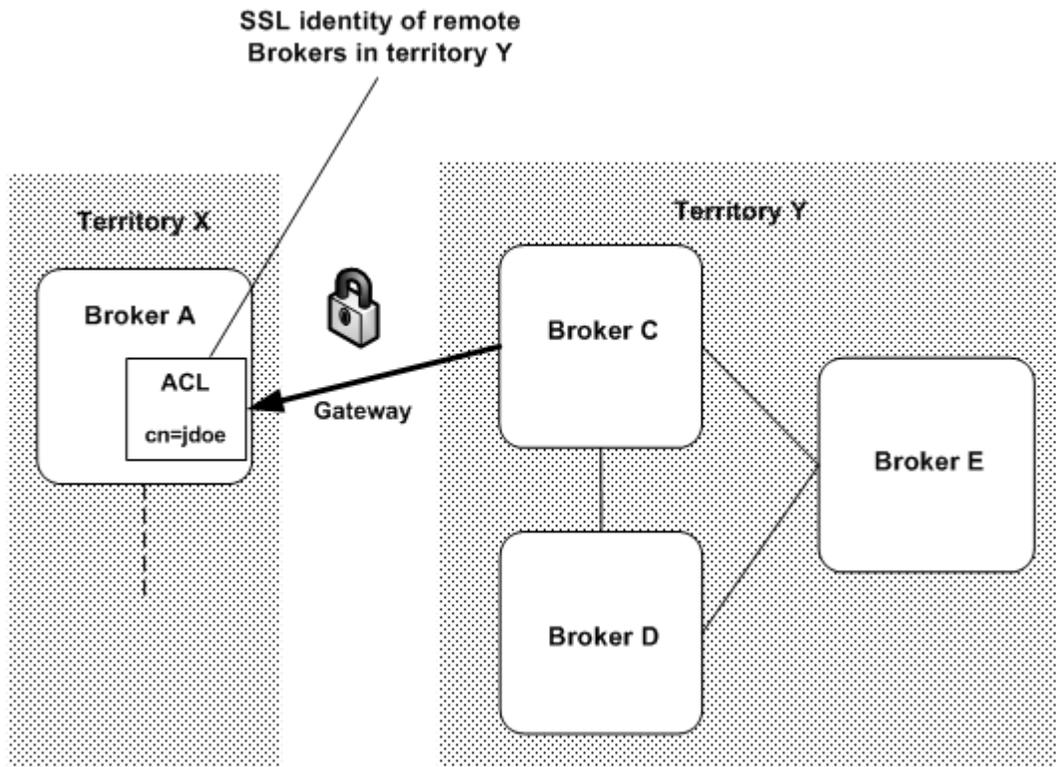
When you set up a territory gateway ACL, the user name ACL on the local gateway contains one entry: the basic authentication identity or SSL identity of the Broker Server on which the remote gateway resides. After permission is granted, information flows from the remote Brokers on the other side of the gateway.

Note: A territory gateway ACL is not shared between the two sides of a gateway connection. Each gateway requires its own ACL.

Normally, when you configure for SSL authentication, you do not need to attach an authenticator name ACL to a territory gateway. However, if the Broker identity from the other territory used a different certification authority, then you must import the trusted root of each territory into the truststore of the other.

The following figure shows the setup of a territory gateway ACL for a one-way flow of information. In the figure, an ACL containing the identity of the Brokers in territory Y is attached to a requesting Broker from territory X.

How a Territory Gateway ACL Works



For information about setting up an ACL for this type of configuration, see [“Setting Permissions to Connect to Remote Brokers across a Territory Gateway”](#) on page 344.

Cluster Gateway ACLs

A cluster gateway allows a Broker in one cluster to retrieve information from Brokers in another cluster. You can set up ACLs on a cluster gateway so that the Broker requesting information from a cluster other than its own must be granted authorization to receive data.

When you set up a cluster gateway ACL, the user name ACL on the local gateway contains one entry: the basic authentication identity or SSL identity of the Broker Server on which the remote gateway resides. After permission is granted, information flows from the remote Brokers on the other side of the gateway.

Note: A cluster gateway ACL is not shared between the two sides of a gateway connection. Each gateway requires its own ACL.

Normally, when you configure for SSL authentication, you do not need to attach an authenticator name ACL to a cluster gateway. However, if the Broker identity from the other cluster used a different certification authority, then you must import the trusted root of each cluster into the truststore of the other.

Configuring Access Control Lists

Once an ACL is enabled, only those users with identities listed in the ACL are granted access to the protected Broker object.

If a Broker object allows ACLs, you can configure a user ACL, an authenticator ACL, or both of these ACLs for the object.

You configure ACLs through My webMethods.

The following sections walk you through the steps for creating, configuring, enabling, and disabling ACLs for:

- The Broker Server
- A client group
- Remote Brokers in a territory
- Remote Brokers in a cluster
- Remote Brokers in a territory gateway
- Remote Brokers in a cluster gateway

Important: For ACLs to work, basic authentication or SSL must be configured and enabled for the Broker Server. If you attempt to create ACLs on a Broker object without first configuring and enabling basic authentication or SSL for its controlling Broker Server, the Broker user interface will direct you to first configure basic authentication or SSL.

About Configuring Broker Server ACLs

The point of using Broker Server ACLs is to control access to the administrative functions of the Broker Server. Once basic authentication or SSL is enabled for a Broker Server, you can use the Broker user interface to configure ACLs that limit administrative access to that server. For more information, see “[Broker Server ACLs](#)” on page 328.

Following are the procedures you need for managing Broker Server ACLs; separate procedures are provided for authenticator name and user name ACLs.

Adding Authenticator Names

Use the following procedure to add authenticator names to a Broker Server ACL.

To add authenticator names to a Broker Server ACL

1. In My webMethods: **Messaging > Broker Servers > Servers**.
2. In the **Broker Servers List**, click the server on which to attach the ACL. If the server does not appear in the list, use the **Search** tab to locate it.
3. On the Broker Server Details page, click the **ACL** tab, then click **Authenticator Names**.

4. Click **Add Authenticator Names**.
5. To add an authenticator name for basic authentication:
Click the **Enter Authenticator Name** tab, type the basic authenticator alias in the **Authenticator Name** box, and click **Add Authenticator Name**. Repeat for each authenticator alias you want to add. For more information about basic authenticator alias, see [“Basic Authentication Alias” on page 307](#).
6. To add an authenticator name for SSL authentication, do one of the following:
 - To add an authenticator name from the Broker Server keystore:
Click the **Select Authenticator Name** tab, check the **Authenticator Names** for issuers for whom to grant permission, and click **Add Authenticator Names**.
 - To add an authenticator name from a different keystore:
Click the **Enter Authenticator Name** tab, type the DN of the authenticator in the **Authenticator Name** box, and click **Add Authenticator Name**. Repeat for each authenticator DN you want to add.

An ACL is automatically enabled after you add entries; no additional actions are needed.

Note: If you do not specify which clients are granted permissions through a user name ACL, any user from basic authentication alias or a SSL DN from an issuer in the authenticator name list has administrative access to the Broker Server.

Adding User Names

Use the following procedure to add user names to a Broker Server ACL.

To add user names to a Broker Server ACL

1. In My webMethods: **Messaging > Broker Servers > Servers**.
2. In the **Broker Servers List**, click the server on which to attach the ACL. If the server does not appear in the list, use the **Search** tab to locate it.
3. On the Broker Server Details page, click the **ACL** tab, then click **User Names**.
4. Click **Add User Names**.
5. To add user names for basic authentication:
Click the **Enter User Name** tab, type the basic authentication user name in the **User Name** box, and click **Add User Name**. Repeat for each user you want to add.
6. To add user names for SSL authentication, do one of the following:
 - To add user names from the Broker Server keystore:
Click the **Select User Names** tab, check those **User Names** for whom to grant permission, and click **Add User Names**.

- To add user names from a different keystore:

Click the **Enter User Name** tab, type the user's DN in the **User Name** box, and click **Add User Name**. Repeat for each user you want to add.

An ACL is automatically enabled after you add entries; no additional actions are needed.

Disabling Broker Server ACLs

Use the following procedure to disable a Broker Server ACL.

To disable a Broker Server ACL

1. In My webMethods: **Messaging > Broker Servers > Servers**.
2. In the **Broker Servers List**, click the server with the ACL. If the server does not appear in the list, use the **Search** tab to locate it.
3. On the Broker Server Details page, click the **ACL** tab, then click either:
 - **Authenticator Names**, to display the list of authenticators belonging to the Broker Server authenticator name ACL.
 - **User Names**, to display the list of users belonging to the Broker Server user name ACL.
4. Click **Disable Authenticator Names** or **Disable User Names**.

Deleting Authenticator Names

Use the following procedure to delete authenticator names from a Broker Server ACL.

To delete authenticator and user names from Broker Server ACLs

1. In My webMethods: **Messaging > Broker Servers > Servers**.
2. In the **Broker Servers List**, click the server with the ACL(s). If the server does not appear in the list, use the **Search** tab to locate it.
3. On the Broker Server Details page, click the **ACL** tab.
4. Do one of the following:
 - To delete authenticator names:
 - i. Click **Authenticator Names**.
 - ii. Check the boxes next to the **Authenticator Names** you want to remove from the authenticator ACL.
 - To delete user names:
 - i. Click **User Names**.
 - ii. Check the boxes next to the **User Names** you want to remove from the user ACL.

5. Click **Delete**.

Granting Access Permissions to a New Client

Suppose you add a new client, and you want that client to be granted access permissions to a Broker Server. You must reconfigure that Broker Server's ACL. For the ACL to grant the new client access permissions, the following conditions must be met:

- Broker basic authentication security or SSL must be able to authenticate the client. For SSL, client's identity must exist in a client keystore and the Broker Server keystore.
- The new client's user name must appear in the user name ACL of the Broker Server.
- The new client's authenticator name must appear in the authenticator name ACL of the Broker Server.

To configure a new SSL client so that it satisfies these conditions, follow the steps in this procedure (it is assumed that the client's keystore has already been properly configured).

To set access permissions to the Broker Server for a new SSL client

1. Add the client's trusted root to the truststore file used by the Broker Server and its signed certificate to a new Broker Server keystore.

See [“Creating Keystores and Truststores” on page 318](#) and [“Managing Certificate Files with OpenSSL” on page 581](#).
2. In My webMethods: **Messaging > Broker Servers > Servers**.
3. In the **Broker Server List**, click the server with the ACL(s) you want to update. If the server does not appear in the list, use the **Search** tab to locate it.
4. To add the client's user DN to the ACL:
 - a. On the Broker Server Details page, click the **ACL** tab, then click **User Names**.
 - b. Click **Add User Names**.
 - c. Click the **Select User Names** tab, check the **User Name (DN)** for the new client, and click **Add**.
5. To add the client's issuer DN to the ACL:
 - a. On the Broker Server Details page, click the **ACL** tab, then click **Authenticator Names**.
 - b. Click **Add Authenticator Names**.
 - c. Click the **Select Authenticator Names** tab, check the **Authenticator Name (DN)** for the new client, and click **Add**.

An ACL is automatically enabled after you add entries; no additional actions are needed.

About Configuring Client Group ACLs

You can use client group ACLs to set access permissions to the client group's can-publish and can-subscribe lists, and the logs controlled by the client group. You also use these ACLs to restrict access to a Broker's admin client group, which has administrator privileges. For more information, see [“Client Group ACLs” on page 326](#).

- To limit administrative access to a specific Broker, configure ACLs for the admin client group. For more information, see [“admin Client Group ACLs” on page 328](#).
- If there are customized document logging utilities that require the eventLog client to be deployed in the Broker, secure the eventLog client group with an ACL. For more information, see [“eventLog Client Group” on page 178](#).

Registering User Names

Use the following procedure to register user names on a client group ACL.

To register user names on a client group ACL

1. In My webMethods: **Messaging > Broker Servers > Client Groups**.
2. Select the **Broker Server** and **Broker** of the client group whose ACL you want to configure.
3. Click **Go**. If the client group does not appear in the list, use the **Search** tab to locate it (see [“Searching for Client Groups” on page 178](#)).
4. Under **Client Groups**, click the client group whose ACL you want to configure.

View the **Configure** page for the client group. If the **Access Control** status reads **BrokerAdministrator identity required**, you need to configure the Broker user interface identity settings before continuing. See [“Configuring an SSL Identity for the Broker User Interface Component” on page 348](#) for instructions.

5. Click the **ACL** tab for the client group.
6. Click **Add User Names**.
7. To add user names for basic authentication:
Click the **Enter User Name** tab, type the basic authentication user name in the **User Name** box, and click **Add**. Repeat for each user you want to add.
8. To add user names for SSL authentication, do one of the following:
 - To add the user name in the Broker Server SSL keystore:
Click the **Select User Name** tab, click the **User Name** for whom to grant permission, and click **Add**.
 - To add user names from a different keystore:
Click the **Enter User Name** tab, type the user DN in the **User Name** box, and click **Add**. Repeat for each user you want to add.

The status message **User name(s) added to access control list** appears on the Client Group Details page.

The ACL is automatically enabled after you add entries; no additional actions are needed.

Registering Authenticator Names

Use the following procedure to register authenticator names on a client group ACL.

To register authenticator names on a client group ACL

1. In My webMethods: **Messaging > Broker Servers > Client Groups**.
2. Select the Broker Server and Broker of the client group whose ACL you want to configure.
3. Click **Go**. If the client group does not appear in the list, use the **Search** tab to locate it (see [“Searching for Client Groups” on page 178](#)).
4. Under **Client Groups**, click the client group whose ACL you want to configure.

View the **Configure** page for the client group. If the **Access Control** status reads **BrokerAdministrator identity required**, you need to configure the Broker user interface component identity settings before continuing. See [“Configuring a Basic Authentication Identity for the Broker User Interface Component” on page 347](#) or [“Configuring an SSL Identity for the Broker User Interface Component” on page 348](#) for instructions.

5. Click the **ACL** tab for the client group.
6. Click **Add Authenticator Names**.
7. To add an authenticator name for basic authentication:
Click the **Enter Authenticator Name** tab, type the alias of the authenticator in the **Authenticator Name** box, and click **Add**. Repeat for each authenticator alias you want to add.
8. To add an authenticator name for SSL authentication, do one of the following:
 - To add an authenticator name from the Broker Server keystore:
Click the **Select Authenticator Name** tab, check the **Authenticator Names** for issuers for whom to grant permission, and click **Add**.
 - To add an authenticator name from a different keystore:
Click the **Enter Authenticator Name** tab, type the DN of the authenticator in the **Authenticator Name** box, and click **Add**. Repeat for each authenticator DN you want to add.

The status message **Authenticator name(s) added to access control list** appears on the Client Group Details page.

The ACL is automatically enabled after you add entries; no additional actions are needed.

Note: If you do not specify which clients are granted permissions through a user name ACL, any user from basic authentication alias or a SSL DN from an issuer in the authenticator name list has administrative access to the Broker Server.

About Controlling Which Brokers Can Join a Territory

You use territory ACLs to control which Brokers can join a territory.

Following are procedures for:

- Configuring a territory ACL and specifying which Brokers are authorized to join the territory
- Removing Brokers from the list of those authorized to join a territory

Specifying Brokers Using User Name DNs

Use the following procedure to specify which Brokers can join a territory by using Broker user name DNs.

To specify which Brokers can join a territory (using Broker user name DNs)

1. In My webMethods: **Messaging > Broker Territories > Territory Brokers**.
2. In the **Territories List**, click the territory on which to create an ACL. If the territory is not in the list, use the **Search** tab to locate it.
3. On the **Territory BrokerDetails** page, click the **Territory** link.
4. Click the **ACL** tab.
5. Click **User Names** (if not already selected), then click **Add User Names**.
6. On the **Add User Names to Territory ACL** page, do the following:
 - To add user names for basic authentication:

Click the **Enter User Name** tab, and in the **User Name** box, type the basic authentication user name of the Broker Server whose Broker or Brokers are permitted to join and participate in the territory. Click **Add**. Repeat for any additional Broker Servers whose Broker or Brokers you also want to join.
 - To add user names for SSL authentication, do one of the following:
 - To add user names from the Broker Server SSL keystore:

Click the **Select User Names** tab, check the **User Names** belonging to the Broker Servers whose Brokers are permitted to join and participate in the territory, and click **Add**.
 - To add user names from a different keystore:

Click the **Enter User Name** tab, and in the **User Name** box, type the DN of the Broker Server whose Broker or Brokers are permitted to join and participate in the territory. Click **Add**. Repeat for any additional Broker Servers whose Broker or Brokers you also want to join.

The status message **User name(s) added to access control list** appears on the Territory Details page.

Specifying Brokers Using Authenticator Names

Use the following procedure to specify which Brokers can join a territory by using Broker authenticator names.

To specify which Brokers can join a territory (using Broker authenticator names)

1. In My webMethods: **Messaging > Broker Territories > Territory Brokers**.
2. In the **Territories List**, click the territory containing the ACL you want to configure. If the territory is not in the list, use the **Search** tab to locate it.
3. On the **Territory Details** page, click the ACL tab.
4. Click **Authenticator Names**, then click **Add Authenticator Names**.
5. On the **Add Authenticator Names to Territory ACL** page, do the following:
 - To add authenticator names for basic authentication:

Click the **Enter Authenticator Name** tab, type the authenticator's alias in the **Authenticator Name** box, and click **Add**. Repeat for each authenticator for whom to grant permission.
 - For SSL authentication, do one of the following:
 - To add authenticator names from the Broker Server SSL truststore:

Click the **Select Authenticator Names** tab, check the **Authenticator Names** for whom to grant permission, and click **Add**.
 - To add authenticator names from a different truststore:

Click the **Enter Authenticator Name** tab, type the authenticator's DN in the **Authenticator Name** box, and click **Add**. Repeat for each authenticator for whom to grant permission.

The status message **Authenticator name(s) added to access control list** appears on the Territory Details page.

The ACL is automatically enabled after you add entries; no additional actions are needed.

Deleting Brokers From a Territory ACL

Use the following procedure to delete Brokers from a territory ACL.

To delete Brokers from a territory ACL

1. In My webMethods: **Messaging > Broker Territories > Territory Brokers**.
2. In the **Territories List**, click the territory, click the territory containing the ACL from which you want to delete Brokers.
3. On the **Territory Details** page, click the **ACL** tab.
4. Do one of the following:
 - To delete authenticator names:
 - i. Click **Authenticator Names**.
 - ii. Check the boxes next to the **Authenticator Names** you want to remove from the authenticator ACL.
 - To delete user names:
 - i. Click **User Names**.
 - ii. Check the boxes next to the **User Names** (for the Broker Servers whose Brokers are participating in the territory) that you want to remove from the user ACL.
5. Click **Delete**.

About Controlling Which Brokers Can Join a Cluster

You use cluster ACLs to control which Brokers can join a cluster. Following are procedures for:

- Configuring a cluster ACL and specifying which Brokers are authorized to join the cluster
- Removing Brokers from the list of those authorized to join a cluster

Specifying Brokers Using User Names

Use the following procedure to specify which Brokers can join a cluster by using Broker user names.

To specify which Brokers can join a cluster (using Broker user names)

1. In My webMethods: **Messaging > Broker Clusters > Cluster Brokers**.
2. In the clusters list, click the cluster on which to create an ACL. If the cluster is not in the list, use the **Search** tab to locate it.
3. On the **Cluster Details** page, click the **ACL** tab.
4. Click **User Names** (if not already selected), then click **Add User Names**.
5. On the **Add User Names to Cluster ACL** page, do one of the following:
 - To add user names for basic authentication:

- i. Click the **Enter User Name** tab.
 - ii. In the **User Name** box, type the basic authentication user name of the Broker Server whose Broker or Brokers are permitted to join and participate in the cluster.
 - iii. Click **Add**.
 - iv. Repeat for any additional Broker Servers whose Broker or Brokers you also want to join.
- For SSL authentication, to add user names from the Broker Server SSL keystore:
 - i. Click the **Select User Names** tab.
 - ii. Check the **User Names** belonging to the Broker Servers whose Brokers are permitted to join and participate in the cluster
 - iii. Click **Add**.
 - For SSL authentication, to add user names from a different keystore:
 - i. Click the **Enter User Name** tab.
 - ii. In the **User Name** box, type the DN of the Broker Server whose Broker or Brokers are permitted to join and participate in the cluster.
 - iii. Click **Add**.
 - iv. Repeat for any additional Broker Servers whose Broker or Brokers you also want to join.

The status message **User name(s) added to access control list** appears on the Cluster Details page.

Specifying Brokers Using Authenticator Names

Use the following procedure to specify which Brokers can join a cluster by using Broker authenticator names.

To specify which Brokers can join a cluster (using Broker authenticator names)

1. In My webMethods: **Messaging > Broker Clusters > Cluster Brokers**.
2. In the clusters list, click the cluster containing the ACL you want to configure. If the cluster is not in the list, use the **Search** tab to locate it.
3. On the **Cluster Broker Details** page, click the **Cluster** link.
4. Click the **ACL** tab.
5. Click **Authenticator Names**, then click **Add Authenticator Names**.
6. On the **Add Authenticator Names to Cluster ACL** page, do one of the following:
 - To add authenticator names from a basic authentication configuration file:

Click the **Enter Authenticator Name** tab, type the authenticator alias in the **Authenticator Name** box, and click **Add**. Repeat for each authenticator for whom to grant permission.

- For SSL authentication, to add authenticator names from the Broker Server SSL truststore:

Click the **Select Authenticator Names** tab, check the **Authenticator Names** for whom to grant permission, and click **Add**.

- For SSL authentication, to add authenticator names from a different truststore:

Click the **Enter Authenticator Name** tab, type the authenticator's DN in the **Authenticator Name** box, and click **Add**. Repeat for each authenticator for whom to grant permission.

The status message **Authenticator name(s) added to access control list** appears on the Cluster Details page.

The ACL is automatically enabled after you add entries; no additional actions are required.

Deleting Brokers From a Cluster

Use the following procedure to delete Brokers from a cluster.

To delete Brokers from a cluster ACL

1. In My webMethods: **Messaging > Broker Clusters > Cluster Brokers**.
2. In the clusters list, click the cluster, and then click the cluster containing the ACL from which you want to delete Brokers.
3. On the **Cluster Details** page, click the **ACL** tab.
4. Do one of the following:
 - To delete authenticator names:
 - i. Click **Authenticator Names**.
 - ii. Check the boxes next to the **Authenticator Names** you want to remove from the authenticator ACL.
 - To delete user names:
 - i. Click **User Names**.
 - ii. Check the boxes next to the **User Names** (for the Broker Servers whose Brokers are participating in the cluster) that you want to remove from the user ACL.
 - iii. Click **Delete**.

Setting Permissions to Connect to Remote Brokers across a Territory Gateway

You configure ACLs on a Broker gateway, so that the Broker requesting information from the remote Brokers on the other side of the gateway must be granted permission to receive data.

For a Broker gateway ACL, the user name ACL on the local gateway contains either the SSL or the basic authentication identity of the Broker Server on which the remote gateway resides.

Configuring Broker Gateway ACLs Using User Names

Use the following procedure to configure a Broker gateway ACL using Broker user names.

To configure a Broker gateway ACL (using Broker user names)

1. In My webMethods: **Messaging > Broker Territories > Broker**.
2. Under **Gateway**, click the gateway whose ACL you want to configure.
3. On the **Territory Gateway Details** page, click the **ACL** tab.
4. Click **Add User Names** if not already selected.
5. Type the basic authentication user name or SSL user DN in the **Remote Broker's User Name** box.
6. Click the **Enable user name** box.
7. Click **Apply**.

Configuring Broker Gateway ACLs Using Authenticator Names

Use the following procedure to configure a Broker gateway ACL using Broker authenticator names.

To configure a Broker gateway ACL (using Broker authenticator names)

1. In My webMethods: **Messaging > Broker Territories > Broker**.
2. Under **Gateway**, click the gateway whose ACL you want to configure.
3. On the **Territory Gateway Details** page, click the **ACL** tab.
4. Click **Add Authenticator Names**, if not already selected.
5. Type the authenticator alias or DN in the **Remote Broker's User Name** box.

Note: For basic authentication, the authenticator name (alias) must exist in the remote Broker's basic authentication configuration file. For SSL, the authenticator name (DN) must exist in the remote Broker's truststore; if it does not, use your certificate editing tool to add it to the truststore.

6. Click **Apply**.

About Setting Permissions to Connect to Remote Brokers across a Cluster Gateway

You configure ACLs on a cluster gateway so that the Broker requesting information from the remote Brokers on the other side of the gateway must be granted permission to receive data.

For a cluster gateway ACL, the user name ACL on the local gateway contains one entry: the SSL identity of the Broker Server on which the remote gateway resides.

Configuring Cluster Gateway ACLs Using Broker User Names

Use the following procedure to configure a cluster gateway ACL using Broker user names.

To configure a cluster gateway ACL (using Broker user names)

1. In My webMethods: **Messaging > Broker Clusters > Cluster Brokers**.
2. Under **Gateway**, click the gateway whose ACL you want to configure.
3. On the **Cluster Gateway Details** page, click the **ACL** tab.
4. Click **Add User Names** if not already selected.
5. Type the basic authentication user name or SSL user DN in the **Remote Broker's User Name** box.
6. Click the **Enable user name** box.
7. Click **Apply**.

Configuring Cluster Gateway ACLs Using Broker Authenticator Names

Use the following procedure to configure a cluster gateway ACL using Broker authenticator names.

To configure a cluster gateway ACL (using Broker authenticator names)

1. In My webMethods: **Messaging > Broker Clusters > Cluster Brokers**.
2. Under **Gateway**, click the gateway whose ACL you want to configure.
3. On the **Cluster Gateway Details** page, click the **ACL** tab.
4. Click **Add Authenticator Names** if not already selected.
5. Type the authenticator alias or DN in the **Remote Broker's User Name** box.

Note: For basic authentication, the authenticator name (alias) must exist in the remote Broker's basic authentication configuration file. For SSL, the authenticator name (DN) must exist in the remote Broker's truststore; if it does not, use your certificate editing tool to add it to the truststore.

- Click **Apply**.

ACLs and Migration Issues

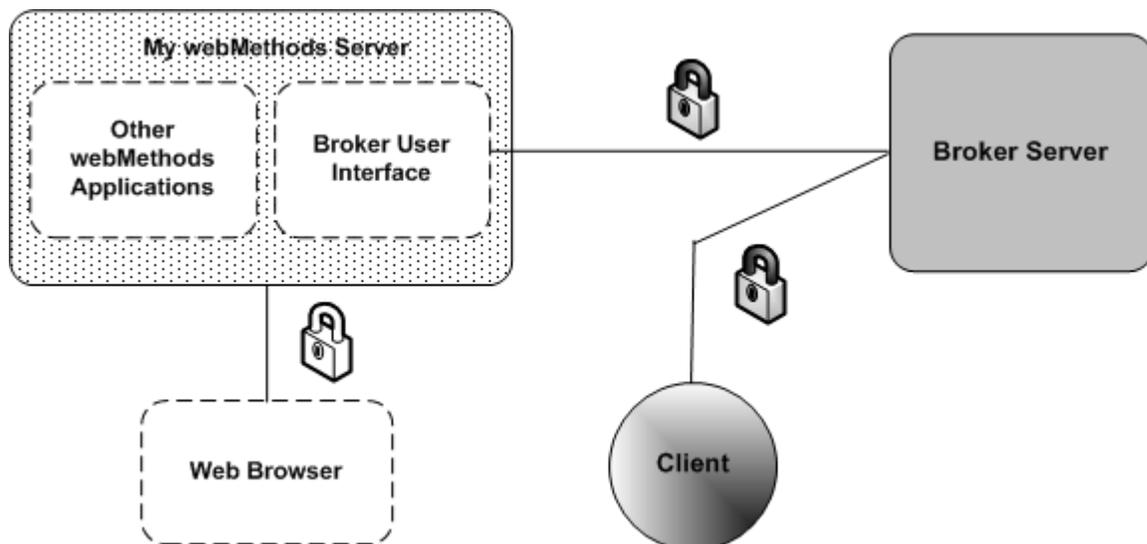
Because of the changes to certificate files and their usage, ACLs cannot be migrated from earlier versions of webMethods Broker to 7.1. You cannot import version 6.x ACL information into 7.1; you need to reconstruct the ACLs from scratch.

When previous version 6.x storage is used with the Broker 7.1 executable, and the storage contains Broker Server SSL identity and ACL information, the version 7.1 Broker will not be able to use the SSL information already in storage. You must establish a new version 7.1 keystore infrastructure for the Broker Server and reconfigure the Broker Server SSL identity using information from the new keystores. After establishing the SSL identity for the Broker Server, you need to manually remove the old DNs that contain "EM" as an attribute from the ACLs, and replace them with "Email=xxx" using the new keystore infrastructure.

Broker Security and My webMethods Server

The following figure shows the Broker security model within the context of My webMethods applications. In the figure, the lines represent connections, the shaded padlocks illustrate Broker connections that can be SSL-enabled, and the white padlock illustrates the connection between the local machine and the My webMethods Server machine, which can be SSL-enabled.

Broker SSL authentication and My webMethods Server authentication



The connection between the local machine hosting your browser and the My webMethods Server machine is established with user name and password authentication. This connection (which can be made with HTTPS, or HTTP over SSL)

is under the control of My webMethods security, and is not part of the Broker security model.

Basic authentication or SSL enabled connections under Broker control can exist between:

- A Broker client and the Broker Server.
- The Broker user interface and the Broker Server. (To the Broker Server, the Broker user interface is simply another client.)

You set either a basic authentication identity or an SSL Identity for the Broker Server in My webMethods.

Configuring a Basic Authentication Identity for the Broker User Interface Component

To a Broker Server, the Broker user interface component is a just another client (for example, a Java program) requesting an authenticated connection. Thus, you assign a basic authentication identity to the Broker user interface and connect to a basic authentication enabled server as you would with any client. That allows the Broker Server to authenticate the Broker user interface at connection time.

Establishing a basic authentication identity for the Broker user interface and connecting to the basic authentication enabled Broker Server is a requirement for enabling Broker security.

The same Broker user interface component can connect to different Broker Servers. To connect to each Broker Server, the Broker user interface component must have a unique basic authentication identity associated with each Broker Server. For example, if your Broker user interface component in My webMethods requires to connect to five different Broker Servers, you must configure five different basic authentication identities in My webMethods.

Once the identity of Broker user interface component in My webMethods to connect to a Broker Server is saved in the database, that identity is used to authenticate every user session. You can use the following procedure to set a basic authentication identity for Broker user interface component in My webMethods to connect to a Broker Server.

To assign a basic authentication identity to the Broker user interface component

1. In My webMethods: **Messaging > Settings > Broker Administrator Identity**
2. Select the server to which you want to connect.

Tip: If there are several servers, to search for the required server, enter the name of the server in **Server Name** text field and click **Search**.

3. Click **Change Identity**.
4. In the Client Authentication dialog, select **Username/Password** from **Client Authentication** list box.

5. In the **Username** text field, type your basic authentication user name.
6. In the **Password** text field, type your basic authentication password.
7. Optionally, you can configure encryption.
 - a. Select the **SSL Truststore**.

The truststore you use for the Broker user interface must be located on the machine hosting your My webMethods applications.
 - b. Select the SSL truststore's File Type (certificate file format) from the **SSL Truststore Type** list.
 - c. Click the **Yes** option for **Encryption**.
8. Click **Connect**.

The identity details along with the encrypted password are stored in the database which My webMethods uses. Once the user establishes an identity, the identity details will be stored for new user sessions.

Configuring an SSL Identity for the Broker User Interface Component

To Broker Server, the Broker user interface component is a just another client (for example, a Java program) requesting a secure connection. Thus, you assign an SSL identity to the Broker user interface and configure SSL as you would with any SSL client. That allows the Broker Server to authenticate the Broker user interface at connection time.

Establishing an SSL identity for the Broker user interface and connecting to the Broker Server (with its SSL identity configured) is a requirement for enabling Broker security.

Note: The password will be encrypted and stored in the database that My webMethods uses.

You can use the following procedure to set an SSL identity for the Broker user interface component in My webMethods.

To assign an SSL identity to the Broker user interface component

1. Make sure you have the file location and password of the SSL keystore for the Broker user interface before starting this procedure. For more information, see [“Creating Keystores and Truststores” on page 318](#).
2. In My webMethods: **Messaging > Settings > Broker Administrator Identity**
3. Select the server to which you want to connect.

Tip: If there are several servers, to search for the required server, enter the name of the server in **Server Name** text field and click **Search**.

4. Click **Change Identity**.

5. On the Client Authentication dialog, select **SSL** from the **Client Authentication** list box.
6. Select the SSL keystore file containing the identity you want to use from the **SSL Keystore** list.

The keystore you use for the Broker user interface must be located on the machine hosting your My webMethods applications.
7. Select the SSL keystore's file type (certificate file format) from the **SSL Keystore Type** list.
8. Type the **SSL Keystore Password**, then click **Get User Names** to retrieve the DN from the keystore.

The user DN from the keystore file displays next to **User Name**.
9. Select the **SSL Truststore** that contains the trusted root of the certificate in the selected keystore.

The truststore you use for the Broker user interface must be located on the machine hosting your My webMethods applications.
10. Select the SSL truststore's File Type (certificate file format) from the **SSL Truststore Type** list.
11. Click **Connect**.

Configuring for One-way SSL Authentication

When you configure the client for one-way SSL authentication, you only need to specify a truststore and not a keystore. The identity of the Broker Server is authenticated by the client, and must be guaranteed through the Broker Server's SSL certificate before a connection is made.

You can also configure basic authentication with one-way SSL authentication. For more information, see ["Securing Broker Server Using Basic Authentication Over SSL"](#).

To configure one-way SSL authentication for the Broker user interface component

1. In My webMethods: **Messaging > Settings > Broker Administrator Identity**
2. Select the server to which you want to connect.

Tip: If there are several servers, to search for the required server, enter the name of the server in **Server Name** text field and click **Search**.
3. Click **Change Identity**.
4. On the Client Authentication dialog, select **None** from the **Client Authentication** list box.
5. Select the **SSL Truststore** that contains the trusted root of the certificate in the selected keystore.

The truststore you use for the Broker user interface must be located on the machine hosting your My webMethods applications.

6. Select the SSL truststore's File Type (certificate file format) from the **SSL Truststore Type** list.
7. Click **Connect**.

Reconfiguring the Identity of the Broker User Interface Component

The identity of the Broker user interface component is persisted in the database that My webMethods uses.

When you delete the Broker Server from the My webMethods User Interface, the User Interface identity to connect to that Broker Server is lost. After adding the Broker Server again, you have to reconfigure the User Interface identity to connect to the added Broker Server.

Disabling the Identity of User Interface Component to the Broker Server

While being in an administrative session on the Broker user interface component, you can disable the identity that is currently set for connecting to a Broker Server. After you disable the identity, all the Broker Server operations such as adding a Broker, changing SSL settings, and changing basic authentication settings, that are protected by Broker Server ACL will no longer be available for anonymous logons.

To disable the Broker user interface component's SSL identity or basic identity

1. In My webMethods: **Messaging > Settings > Broker Administrator Identity**
2. Select the server.

Tip: If there are several servers, to search for the required server, enter the name of the server in **Server Name** text field and click **Search**.

3. Click **Disable Identity**.

Viewing the Identity Details of the Broker User Interface Component to the Broker Server

You can view the identity details of the Broker user interface component to any Broker Server in My webMethods.

To view the Identity Details of the Broker user interface component

1. In My webMethods: **Messaging > Settings > Broker Administrator Identity**
2. Select the Details icon () next to the required server.

The identity details of the Broker user interface component associated with the Broker Server are displayed.

Converting Certificate Files

Keystore files created for Broker version prior to 7.1 will not work with the current version of Broker Servers. For those keystore files to work, they must be modified to a format accessible by Broker 7.1.

The Broker Certificate Conversion utility, a command-line utility helps automate this process. It converts a pre-version 7.1 keystore file to the new format required by Broker.

This section briefly describes the Certificate Conversion utility and provides instructions for downloading and using the tool.

Note: Although keystore files can be converted using the Certificate Conversion utility, trusted roots from pre-version 7.1 keystores must be manually converted into truststore files. In addition, existing version 6.x ACLs must be reconstructed (see [“ACLs and Migration Issues” on page 346](#)).

Broker Certificate Conversion Utility

The Broker Certificate Conversion utility `ConvertCert.jar` is a command-line tool that allows you to upgrade pre-version 7.1 keystore files to the new keystore file format.

Single Certificate Format

Keystores contain only a single certificate or certificate chain, whereas keystores older than version 7.1 may contain multiple certificates. If there are multiple certificates (or certificate chains), the Conversion utility generates a directory that contains as many new keystore files as there are certificates in the pre-7.1 keystore.

Trusted Roots and Truststore Files

Unlike pre-version 7.1 keystores, keystores used by Broker 7.1 and later do not contain trusted roots. The Conversion utility removes the trusted roots from the pre-7.1 keystore files upon creating the new keystore file(s), but does not generate a truststore file. These truststore files must be created manually using a certificate editing utility (see [“Managing Certificate Files with OpenSSL” on page 581](#)).

Note: If the only information you have about the trusted roots for your certificates is in the old keystore file, create your truststore file before using the Certificate Conversion utility. That is because the utility deletes information about the trusted roots during the new keystore conversion process. If you need to retrieve the trusted root information, use `awcert` (Broker Certificate Manager Command-Line Utility available with previous versions of Broker) along with its command reference.

Downloading the Broker Certificate Conversion Utility

The utility can be downloaded from the Empower Product Support Web site. The file name of the utility is `ConvertCert.jar`.

Important: You must include the file `enttoolkit.jar` in the `CLASSPATH` for the Certificate Conversion Utility to work.

Command Usage

The following shows the syntax and usage for `ConvertCert.jar`:

```
java -jar ConvertCert.jar { export | list } old-keystore-file password
output-dir
```

Part	Description
<code>export or list</code>	Whether to export the certificates in the old keystore file to newly created keystore(s), or display a list of certificates in the old keystore file in the command window.
<code>old-keystore-file</code>	Name of the keystore file to be migrated.
<code>password</code>	Password for the old keystore file.
<code>output-dir</code>	Name of the directory in which the new keystore(s) will be created.

Remarks

- The format of the new keystore(s) is PKCS#12 (files contain a ".p12" suffix).
- The password or passwords for new keystores are the same as those for the old keystore. To ensure security, consider changing them as soon as possible with your certificate editing utility.
- New keystore files are named according to alias information extracted from a certificate's distinguished name, or DN. If multiple keystores are created, a unique numerical suffix is appended to each file name.

14 Managing Territories

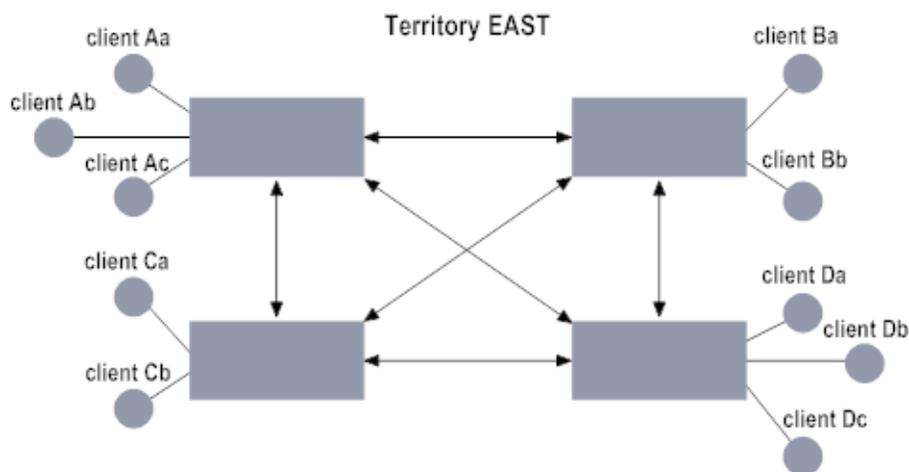
■ Introduction to Territories	354
■ Creating a Territory	359
■ Adding Brokers to a Territory	362
■ Removing a Broker from a Territory	363
■ Deleting a Territory	363
■ Viewing a List of Known Territories	363
■ Viewing Information about a Remote Broker Object	364
■ Deleting a Remote Broker Object	366

Introduction to Territories

A territory is a group of inter-connected Brokers that functions as one logical Broker. All Brokers in a territory maintain the same set of *document types* and *client groups*, however, they each support their own set of *clients*.

A client can connect to any Broker in the territory and publish documents to or receive documents from any other client in the territory. The Broker to which a client publishes documents takes care of forwarding those documents to subscribers that are connected to other Brokers in the territory. To an individual publisher or subscriber, a territory appears as one large Broker.

An Example Territory Made Up of Four Brokers



Brokers that operate in a territory are *peers*. There is no "central" or "controlling" Broker. Each Broker in the territory is responsible for notifying its peers of changes (e.g., additions, deletions, or modifications) that occur to its document types or client groups. By dynamically propagating changes to all Brokers in this manner, document types and client groups remain synchronized across the entire territory.

Why Implement a Territory?

If you were to keep adding clients to a single Broker, that Broker would, at some point, begin running out of resources such as memory, disk space, or socket connections. A territory enables you to scale a Broker to handle large numbers of clients by spreading the client population across multiple host machines.

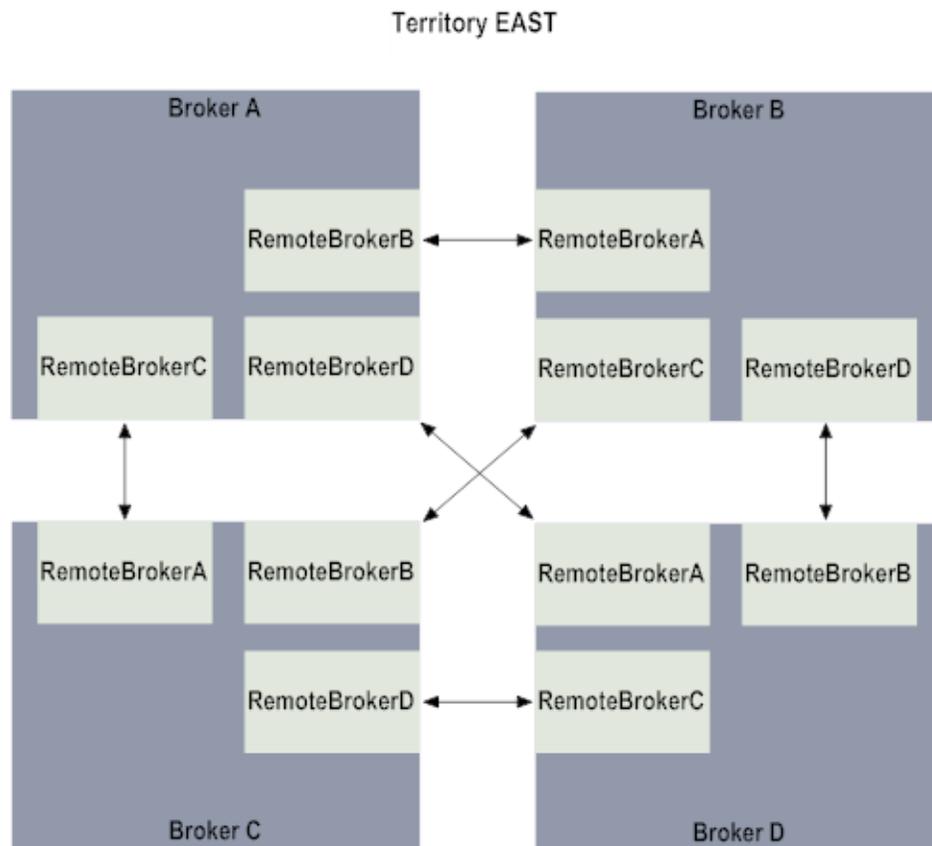
For additional flexibility, you can create multiple territories and link them together using *gateways*. A gateway is a connection that allows specified documents to flow between two territories. For more information about gateways, see [“Managing Territory Gateways” on page 367](#).

How Brokers in a Territory Communicate with Each Other

Each Broker in a territory is directly connected to each of its peers. For example, in a territory that consists of Brokers A, B, C and D, Broker A has a connection to Brokers B, C, and D; Broker B has connections to A, C, and D; Broker C has a connection to Brokers A, B, and D, and so forth.

Each Broker in a territory maintains a *Remote Broker object* for each of the Brokers to which it is connected. This object acts as the server-side state object for the Broker at the other end of the connection.

Each Broker in a territory maintains a Remote Broker object for each of its peers



In this capacity, the Remote Broker object serves the same function as does a client state object for a regular Broker client program. Like a client state object, the Remote Broker object maintains a list of *subscriptions* and a *queue* for the remote Broker.

However, unlike an ordinary client state object, a Remote Broker object also functions as a client of the remote Broker. In this capacity, the Remote Broker object continually retrieves documents for the local Broker from its queue on the remote Broker.

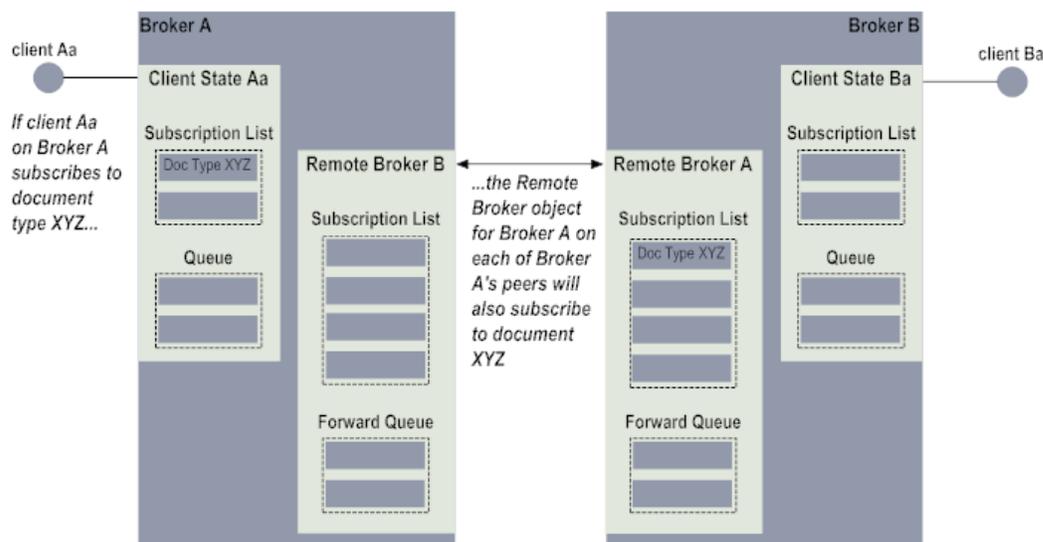
Subscriptions Maintained by a Remote Broker Object

A Remote Broker object subscribes to documents on behalf of clients on the remote Broker that it represents. The subscription list in a Remote Broker object will contain a subscription for each document type for which there is at least one subscriber on the remote Broker. A Remote Broker object automatically drops a subscription from its subscription list when the last remaining subscriber for the document type cancels its subscription.

By dynamically maintaining subscription lists in this way, the Remote Broker objects ensure that only documents for which there are actual subscribers flow between Brokers in a territory.

Note: When a Broker hosts a gateway to another territory, the gateway is also a "client" that subscribes to documents. Therefore, in addition to receiving documents that match regular client subscriptions, a Broker that hosts a gateway will also receive documents that match the gateway subscription list.

Subscriptions are registered in the Remote Broker objects



How Subscription Filters Are Handled in a Territory

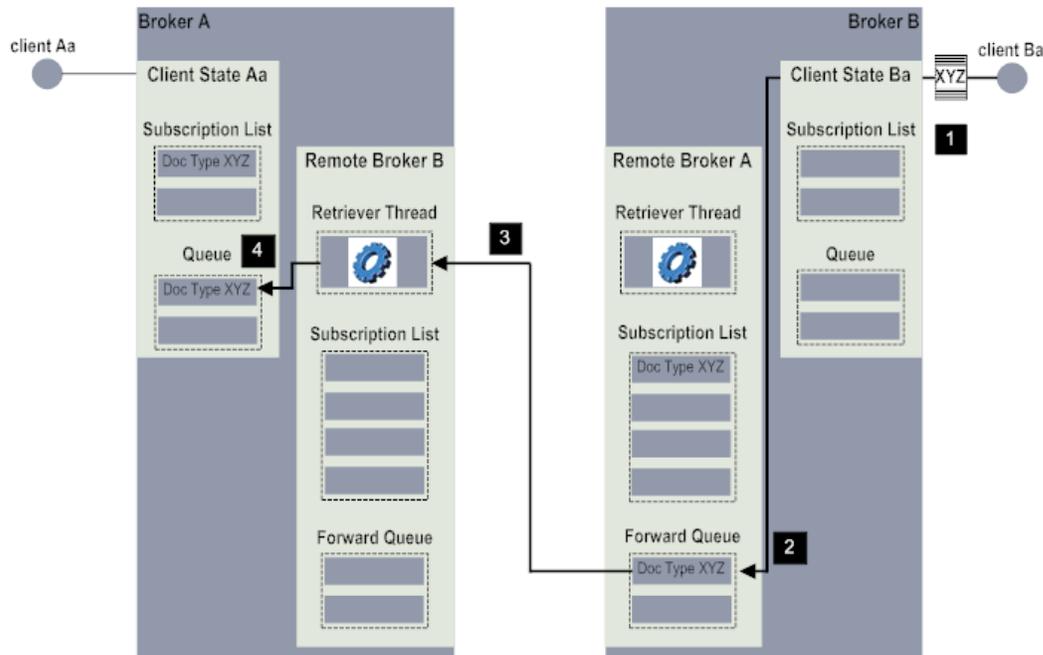
Subscription filters, even if present in a client subscription, are omitted from subscriptions in the Remote Broker object. The Remote Broker object subscribes only to unfiltered document types. Filtering is performed by the remote Broker after it retrieves documents from the Remote Broker object (that is, when the remote Broker inserts the documents into the subscriber's queues.)

The Queue Maintained by a Remote Broker

A Remote Broker object also maintains a queue, called the *forward queue*. The forward queue contains instances of documents that match subscriptions in the Remote Broker

object's subscription list. When the remote Broker is connected to its Remote Broker object, it continually retrieves documents from this forward queue and distributes the documents to its own clients.

A retriever thread gets documents from the forward queue and distributes the documents to local clients



A Remote Broker object is similar to an explicit-destroy client-state object in that it remains instantiated even if the remote Broker disconnects or goes offline. When the remote Broker comes back online, it automatically reconnects to its peers and retrieves documents from its forward queues.

Certain conditions can cause a Remote Broker object to become *orphaned*, meaning that the Remote Broker object exists for a Broker that has been permanently deleted or removed from the territory. An orphaned Remote Broker object will continue to collect documents in its forward queue and can potentially consume a great deal of storage if it is not administratively removed. To delete an orphaned Remote Broker object, see [“Deleting a Remote Broker Object”](#) on page 366.

How Delivered Documents Are Handled in a Territory

A Broker will not forward a delivered document to other Brokers in a territory unless the document is addressed to a client by its fully-qualified client ID (in the form `/territoryName/BrokerName/ClientID`). If a Broker receives a delivered document that is addressed to an unqualified clientID, it delivers the document locally. If the addressee does not exist on the local Broker, the document is sent to the dead-letter queue (if one is configured) or is discarded.

How Brokers Synchronize Metadata

To function as one logical Broker, all the Brokers in a territory must have identical document-type and client-group information.

When a Broker initially joins a territory, its document types and client groups (including ACL settings) are synchronized with the other Brokers in the territory. The subscription lists in the Remote Broker objects are also populated at this time.

After a Broker becomes a member of a territory, it is responsible for notifying its peers of any changes that occur to its document types or client groups. When a Broker in a territory receives such notification, it immediately applies the change to its own metadata. Because these metadata changes are self-propagating, you can make a change to a document type or client group on any Broker and that change will automatically be applied to all Brokers in the territory.

Note: Because it takes a certain amount of time for a change to propagate across the entire territory, if two administrators make a change to the same object on two different Brokers at nearly the same time, it is possible that one administrator's changes will be overwritten by the other's. To avoid this possibility, we suggest that you appoint a single administrator to manage updates to the document type and client group metadata for the territory.

Exporting and Importing Territory Information

You can export and import metadata about a Broker territory and its dependent objects in the format of an XML file. For information about this feature, see [“Exporting Territories” on page 487](#).

Securing a Territory

Within a territory, either all Brokers use SSL or no Brokers use SSL. You cannot mix SSL-enabled Broker Servers and non-SSL-enabled Broker Servers in the same territory. Similarly, encryption, when enabled, is enabled for all connections between Brokers in the territory. You cannot selectively enable encryption between certain Brokers.

When the Broker Servers in a territory are SSL-enabled, you can use an access control list (ACL) to identify which Brokers are authorized members of the territory. Brokers that are not listed in the ACL, are not allowed to join or participate in the territory.

For more information about securing a territory, see [“Territory ACLs” on page 329](#).

Leaving a Territory

To remove itself from a territory, a Broker must formally "leave" the territory. This process deletes the Remote Broker objects (and thus the forward queues and their

contents) from the Broker's peers in the territory. When the last Broker leaves a territory, that territory ceases to exist.

Unless it is the only remaining Broker in a territory, it is important that at least one of the Broker's peers be running when you remove a Broker from a territory. Otherwise, the Remote Broker objects that exist on the other Brokers in the territory will become orphaned, and you will have to manually remove them as described in [“Deleting a Remote Broker Object” on page 366](#).

Creating a Territory

To create a new territory, you use the procedure below to give the territory a name and add the first member Broker to it. Then you use the procedure in [“Adding Brokers to a Territory” on page 362](#) to add additional Brokers to the territory.

Note: If all the member Brokers reside on the same Broker Server (which might be the case in a development environment), you can create the entire territory using just the following procedure.

Requirements Checklist

Before you begin, make sure the following requirements are met:

- The domain name service (DNS) used by the host machines that participate in the territory must be capable of *bi-directional name resolution*. This means that your DNS must be able to resolve the name of the host machine to the correct IP address and also be able to resolve the machine's IP address back to the correct name. Verify the name and IP address of the host machine by using the commands described in [“Reverse Lookup” on page 360](#).

When a host machine uses the Dynamic Host Configuration Protocol (DHCP) to dynamically obtain its IP address, many DNS servers will not return the correct name for a given IP address. In this case, you must contact your IT department or your network administrator and ask that a static IP address be assigned to the host machine.

- Each Broker must have a name that is unique within the territory.
- Each territory must have a unique name.
- A Broker must not belong to another territory. To become part of the new territory, it must first leave its current territory. (For similar reasons, you cannot merge territories. To create a single territory where two existed before, the Brokers in one territory must leave it and then join the second territory.)
- All Broker Servers that participate in the territory must have the same SSL capabilities. That is, they must all be SSL-enabled or non-SSL enabled.

Reverse Lookup

Broker relies on reverse DNS lookup to determine the numeric IP address and the host name. Reverse lookup is also applicable when you create and use a connection factory with Broker host details.

Use the following commands at the command prompt to verify that the reverse lookup gets the fully qualified domain name (FQDN).

Command	Description
<code>nslookup <hostname></code>	Displays the fully qualified domain name and IP address of: <ul style="list-style-type: none"> ■ DNS ■ Host machine identified by <code>hostname</code>.
<code>nslookup <hostIP></code>	Displays the fully qualified domain name and IP address of: <ul style="list-style-type: none"> ■ DNS ■ Host machine identified by <code>hostIP</code>.
<code>nslookup -query=ptr</code> <code><reverse IP>.in-addr.arpa</code> <code><DNS></code>	Displays the fully qualified domain name of the host machine for which you specified the reverse IP address (<i>reverseIP</i>) of the host machine and the fully qualified domain name of the DNS (<i>DNS</i>).

Determine the fully qualified domain name and IP address of the DNS and the host machine by specifying the name of the host machine

```
C:\>nslookup host_machine
Server: dns.localhost.com
Address: 10.60.24.25
Name: host_machine.localhost.com
Address: 10.60.25.85
```

Determine the fully qualified domain name and IP address of the DNS and the host machine by specifying the IP address of the host machine

```
C:\>nslookup 10.60.25.85
Server: dns.localhost.com
Address: 10.60.24.25
Name: host_machine.localhost.com
Address: 10.60.25.85
```

Verify the fully qualified domain name and IP address of the DNS and host the machine by specifying the reverse IP address of the host machine and the fully qualified domain name of the DNS

```
C:\>nslookup -query=ptr 85.25.60.10.in-addr.arpa dns.localhost.com
Server: dns.localhost.com
Address: 10.60.24.2585.25.60.10.in-addr.arpa
name = host_machine.localhost.com
```

Creating Territories

Use the following procedure to create territories.

To create a territory

1. In My webMethods: **Messaging > Broker Territories > Brokers.**
2. Click **Add Territory.**
3. In **Territory Name**, type a name for the territory.
 - The name you choose must be unique among all territories that will be connected using gateways.

Tip: We recommend that you assign unique names to territories even if you do not plan to connect them using a gateway. Doing this will ensure that you can easily distinguish one territory from another in My webMethods.

- The name cannot contain the following characters:

@ / :

- The first character must not be #.
- The name cannot exceed 255 characters.

Example: OrderSys 01

4. Use the following steps to select one of the Brokers that will participate in this territory:
 - a. In the **Broker Server** field, select the Broker Server that hosts the Broker.
 - b. Use the **Selected Brokers** controls to specify the Brokers that you want to add.

Note: Only Brokers that are not already members of another territory appear in the **Available** list. If the Broker that you want to add to the territory does not appear in this list, check whether it is already a member of another territory.

5. Click **OK.**
6. If you want to assign an ACL to this territory so that only authorized Brokers can join it, perform the procedures in [“About Controlling Which Brokers Can Join a](#)

[Territory](#) on page 339. (All Broker Servers in the territory will need to be SSL-enabled to use this feature. For information about SSL-enabling a Broker Server, see [“Configuring SSL for Broker Server ”](#) on page 317.)

7. Use the procedure in [“Adding Brokers to a Territory”](#) on page 362 to add Brokers from other Broker Servers to the territory.

Adding Brokers to a Territory

Use the following procedure to add one or more Brokers to an existing territory. (This action is also known as "joining" a territory.)

Note: To use this procedure, the territory to which you want to add the Broker must already exist. If the territory does not already exist, see [“Creating a Territory”](#) on page 359.

When you add a Broker to a territory, its client groups and document types are automatically synchronized with the other Brokers in the territory. If the join process encounters conflicting definitions that it cannot reconcile, you will receive an error message. To add the Broker to the territory, you must manually reconcile the conflict or accept modifications that the join process proposes.

Before you add a Broker to a territory, review the [“Requirements Checklist”](#) on page 359 and be certain it meets the requirements outlined in the checklist.

To add a Broker to a territory

1. In My webMethods: **Messaging > Broker Territories > Brokers**.
2. In the **Territory Brokers List**, click **Add Broker**.
3. From the **Territory to Join** list on the Add Territory Broker page, select the territory to which you want to add a Broker. You can also select a specific Broker in a territory that you want to connect to.
4. Use the following steps to select the Brokers to add to the territory.
 - a. In the **Broker Server** field, select the Broker Server that hosts the Brokers that you want to add.
 - b. Use the **Selected Brokers** controls to specify the Brokers that you want to add to territory.

Note: Only Brokers that are not already members of another territory appear in the **Available** list. If the Broker that you want to add to the territory does not appear in this list, check whether it is already a member of another territory.

5. Click **Join**.

If the join process encounters conflicting client group or document type definitions during the synchronization process, you will receive an error message. To join the

territory, you can either manually edit the metadata or accept changes proposed by the join process.

Removing a Broker from a Territory

Use the following procedure to remove a Broker from an existing territory.

Important: Unless you are removing the last Broker from a territory, you should ensure that at least one other Broker in the territory is running when you perform this procedure. Otherwise, the other Brokers in the territory will not be notified to drop their Remote Broker object for this Broker and those objects will become orphaned. You will have to manually delete them as described in [“Deleting a Remote Broker Object” on page 366](#).

To remove a Broker from a territory

1. In My webMethods: **Messaging > BrokerTerritories > Brokers**.
2. In the **Territory Brokers List**, select the check box beside the Broker that you want to remove from a territory.
3. Click **Leave Territory**.

Deleting a Territory

You can dissolve a territory by simply removing every Broker from the territory. When the last Broker has been removed from the territory, the territory will cease to exist.

To remove a Broker from a territory, see [“Removing a Broker from a Territory” on page 363](#).

Viewing a List of Known Territories

You can use the following procedure to view a list of the territories that are known to the Broker Servers that you are managing (that is, the Broker Servers that appear in your Broker Server List in My webMethods).

You can also view a graphic representation of known territories on the Topology page. For procedures, see [“Topology View of Territories” on page 275](#).

To view the list of known territories

In My webMethods, select: **Messaging > BrokerTerritories > Brokers**.

The **Territory Brokers List** displays a list of known territories and their known members. (This page might take a few moments to load while My webMethods scans all known Broker Servers and Brokers for territories.)

Note: The Brokers listed under each territory do not necessarily represent the complete membership of the territory. If member Brokers reside on Broker Servers that are not in your Broker Server List, those Brokers will not appear in the list.

Column	Description
Broker@Server	The name of the Broker and the Broker Server on which it resides. Click the name to display details about the Remote Broker objects on this Broker.
Connected	The state of the Broker as described in “Viewing Basic Operating Statistics for the Broker” on page 148.
Connections	The number of clients currently defined on this Broker. This number includes explicit-destroy clients that are not currently connected.
Recent Deliveries or Total Deliveries	<ul style="list-style-type: none"> ■ Recent Deliveries displays the number of publish or deliver requests this Broker received during the last statistics collection interval. The value of the Time interval between statistical refresh field specifies the statistics collection interval. <ul style="list-style-type: none"> Note: Recent Deliveries data is available only if you have selected Enable Statistical Polling. ■ Total Deliveries displays the number of publish or deliver requests this Broker received. <ul style="list-style-type: none"> Note: Total Deliveries data is available only if you do not select Enable Statistical Polling.
Gateway	The list of gateways, if any, that this Broker hosts.

Viewing Information about a Remote Broker Object

Use the following procedure to view the **Territory Broker Details** tab, which displays the list of remote Broker objects that a Broker maintains. The information on this tab indicates:

- Whether the remote Broker is currently connected to the local Broker
- The number of documents that reside in the forward queue for the remote Broker and the time at which the Remote Broker last retrieved documents from this queue

- The number of documents that the local Broker has recently retrieved from its forward queue on the remote Broker and the time at which it last retrieved documents from that queue

To view information about a Remote Broker object

1. In My webMethods: **Messaging > Broker Territories > Brokers**.
2. In the **Territory Brokers List**, click the Broker on which the Remote Broker object resides.

The list at the bottom of the **Territory Broker Details** tab displays the following information for each Remote Broker object on the Broker.

Column	Description
Remote Broker	<p>The name of the Remote Broker object. This name identifies the remote Broker that the Remote Broker object represents.</p> <p>You can click this name to open the Territory Broker Information page for the remote Broker.</p>
Linked	<p>Indicates the state of the local Broker's connection to the remote Broker as follows:</p> <ul style="list-style-type: none"> ■  Yes. The remote Broker is currently connected to the local Broker. ■  No. The remote Broker is not currently connected to the local Broker.
Forward Queue Length	The number of documents currently queued for the remote Broker.
Forward Queue Size	The size of the queue, in bytes.
Recent Local Receipts	The number of documents that the local Broker has retrieved from the remote Broker during the last statistics collection interval.
Last Local Receipt	The time at which the local Broker last retrieved documents from the remote Broker.
Recent Remote Deliveries	The number of documents that the remote Broker retrieved from its forward queue during the last statistics collection interval.

Column	Description
Last Remote Delivery	The time at which the remote Broker last retrieved documents from its forward queue.
Keep Alive	<p>The frequency (in seconds) at which the local Broker issues a "keep-alive" message to this remote Broker. Keep-alive messages are generally used to maintain a connection between Brokers that are separated by a firewall. They prevent a firewall from disconnecting what it considers to be an idle connection.</p> <p>Click the keep-alive value if you want to change it.</p> <p>A keep-alive interval of 0 seconds disables the keep-alive feature. When disabled, the Keep Alive column for the Remote Broker object will display "Disabled."</p>

Deleting a Remote Broker Object

You can use the following procedure to delete a Remote Broker object from a Broker. If the remote Broker that the Remote Broker object represents is online, this procedure will remove that Broker from the territory. If the remote Broker is not online, this procedure removes the Remote Broker object from the local Broker and from every other Broker in the territory.

To delete a Remote Broker object

1. In My webMethods: **Messaging > Broker Territories > Brokers**.
2. In the **Territory Brokers List**, click the Broker on which the Remote Broker object resides.
3. In the **Territory Broker Details** tab, select the check box beside the Remote Broker object that you want to delete.
4. Click **Remove from Territory**.

15 Managing Territory Gateways

■ Overview	368
■ Creating a Territory Gateway Connection	370
■ Configuring and Controlling Territory Gateway Behavior	372
■ Creating a Territory Gateway Pair if You Control Both Brokers	373
■ Creating a Territory Gateway Pair if You Control Only One Broker	377
■ Using Territory Gateway Filters	379
■ Viewing Information about a Territory Gateway	383
■ Viewing Information about Documents Flowing between the Gateways	386
■ Deleting a Territory Gateway Connection	387
■ Removing an Allowed Document Type from a Territory Gateway	388
■ Pausing and Resuming a Territory Gateway	389
■ Preventing Connection Timeouts between Territories	390
■ Configuring the Forwarding Mode	391
■ Refusing Document-Type Updates	392
■ Using Broker Remote Publish	393

Overview

A territory gateway connection enables two territories to exchange documents. Unlike territories, whose constituent Brokers automatically share their document types and client groups, a territory gateway connection requires an administrator to explicitly specify which document types the territories are permitted to send and receive. Moreover, the territories never share client group information.

Typically, you use a territory gateway connection when you need to share documents between two different application or administrative domains. For example, in a configuration where one territory handles documents relating to employee records and another territory handles documents relating to IT resources, the IT territory might need to receive certain documents relating to employee start, exit, or transfer events. To enable these documents to travel to the IT territory, you can link the territories using a territory gateway connection.

You can also use a territory gateway connection to link identical territories that are geographically separated. For example, if you have a territory that handles documents relating to manufacturing operations in North America and you expand the application to operations in Asia, you might decide to create an identical territory for the Asian region and then link the territories using a territory gateway connection. In this case, you are using a gateway to conserve wide-area network bandwidth by explicitly restricting what document types can flow across the network link.

Territory Gateway Connections, Territory Gateways, and Territory Gateway Pairs

The term *territory gateway connection* refers to the link that enables two territories to exchange documents. To create this link, two Brokers, one from each territory, are designated to communicate with one another. To form a territory gateway connection, you must create and configure a *gateway* on each of these Brokers.

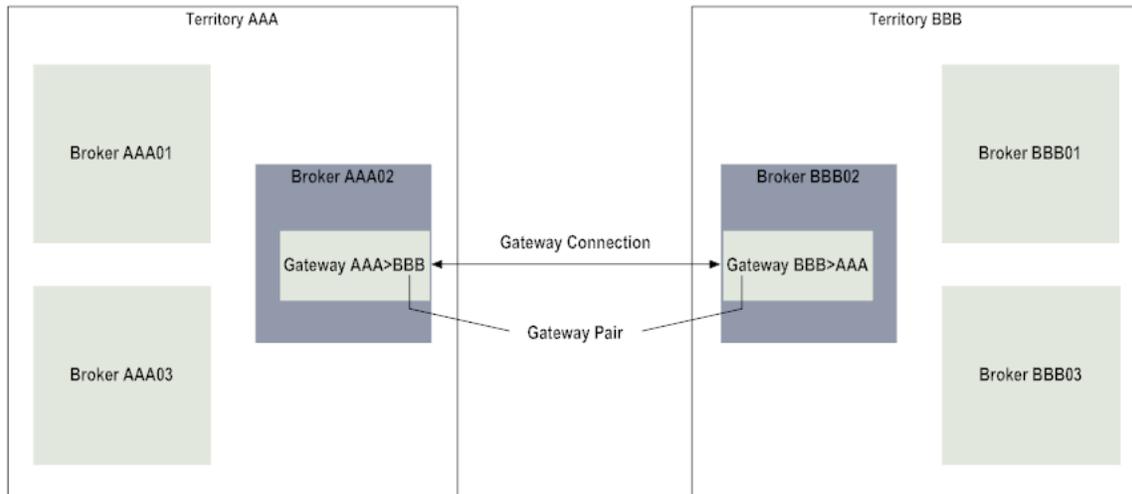
A territory gateway performs functions similar to those performed by a Remote Broker object in a territory. In this capacity, the territory gateway:

- Establishes and maintains a connection with its counterpart in the remote territory
- Maintains the subscription list and forward *queue* for the remote territory
- Retrieves documents for the local territory from the remote gateway

Unlike a Remote Broker object, however, a territory gateway requires an administrator to explicitly specify which document types it can "forward to" and "receive from" the remote territory. The territory gateway enforces these permissions on the documents that pass through it.

Note: The Broker that hosts a territory gateway is a regular member of its own territory and can handle regular clients in addition to the territory gateway.

Collectively, the two territory gateways that operate together to form a territory gateway connection are known as a *gateway pair*. A territory gateway connection will not function until both Brokers have been equipped with a territory gateway and both territory gateways in the pair are configured correctly.



Territory Gateway Names

In the Broker user interface, a territory gateway is identified by a name. As shown below, the name of the territory gateway is composed from the names of the two territories that the territory gateway connects:

localTerritory > *remoteTerritory*

Where *localTerritory* is the name of the territory where the territory gateway resides and *remoteTerritory* is the name of the territory to which the territory gateway connects.

The gateway pair shown in the diagram above consists of a gateway named "AAA \> BBB" and gateway named "BBB \> AAA."

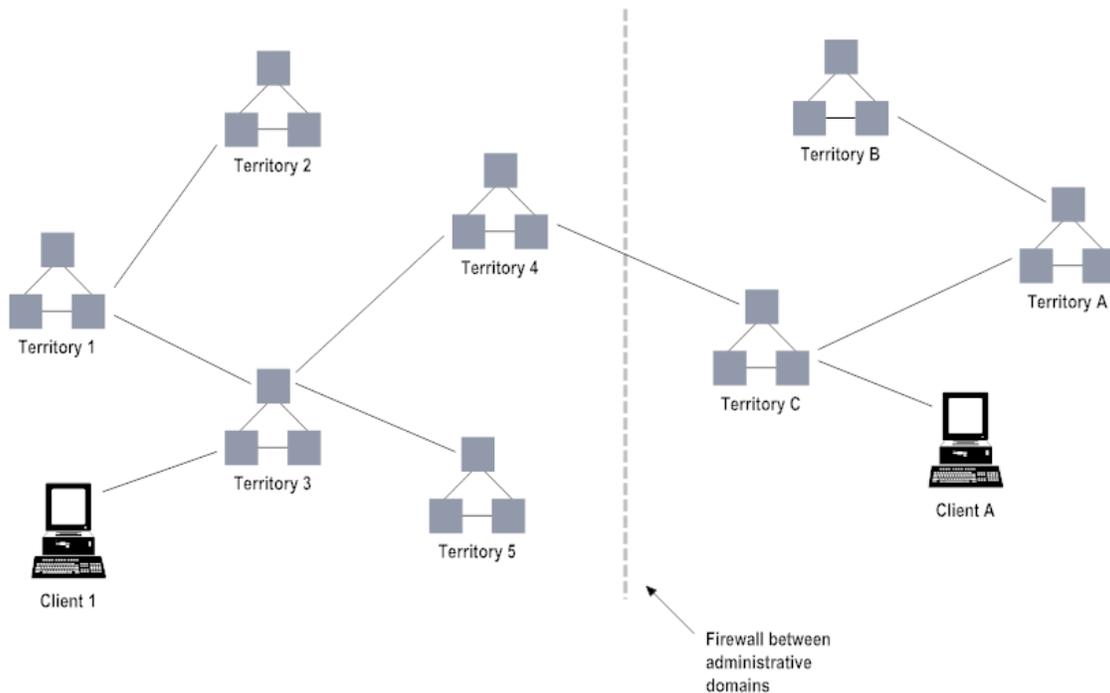
Connecting Multiple Territories with Territory Gateway Connections

Although only one gateway connection can exist between two territories, a territory can have territory gateway connections with multiple territories. For example, territory AAA can have only one gateway connection to territory BBB, but it can have additional territory gateway connections to other territories. If you wanted to connect AAA to territories BBB and CCC, for example, you would create and configure two territory gateways on territory AAA: one gateway, AAA>BBB, and another gateway, AAA>CCC. You could place these gateways on the same Broker or on different Brokers in territory AAA.

If you use territory gateway connections to link multiple territories, be aware that the set of connected territories must form a graph that has no cycles (that is, a path that

traverses the graph should not be able to return to its beginning). Visually, an acceptable graph looks like a tree.

The following figure shows a graph that crosses the boundary between two administrative domains. With the correct permissions set at each territory gateway, client 1 and client A can exchange documents with each other.



Creating a Territory Gateway Connection

To link territories with a territory gateways connection, you must to perform the following high-level steps

1. **Identify the list of document types that each territory needs to receive from the other** and *make sure that these document types exist in both territories*. If a required document type does not exist in one of the territories, the administrator of that territory must add it.
2. **Identify the two Brokers that will host the territory gateways**. The domain name service (DNS) used by the two host machines must be capable of *bi-directional name resolution*, meaning that the DNS must be able to resolve the name of the host machine to the correct IP address and also be able to resolve the machine's IP address back to the correct name.

When a host machine uses the Dynamic Host Configuration Protocol (DHCP) to dynamically obtain its IP address, many DNS servers will not return the correct name for a given IP address. In this case, you must contact your IT department or your network administrator and ask that a static IP address be assigned to the host machine.

3. **Determine what type of authentication and encryption is required between the territories** and make sure that the Broker Servers that host the two gateways support these requirements.

For example, if you want to secure each gateway with an access control list (ACL), both Broker Servers must be SSL-enabled (that is, have an SSL identity) and their keystores must be equipped with the trusted root and signed digital certificate for the Broker Server that hosts their counterpart in the gateway pair. For more information about SSL-enabling a Broker Server and assigning an ACL to a gateway, see [“Territory Gateway ACLs” on page 331](#).

4. **Create and configure each territory gateway in the gateway pair.** When you configure a territory gateway, you must do the following:
 - a. Specify the name and location of the Broker that hosts the territory gateway in the remote territory.
 - b. Assign "Allow Receive" permission to each document type that the local territory needs to receive.
 - c. Assign "Allow Forward" permission to each document type that the remote gateway's territory needs to receive. (If necessary, you can include a filter that permits only specified instances of the document type to pass to the remote territory.)

If you control both Broker Servers, you can create and configure both territory gateways at the same time as described in [“Creating a Territory Gateway Pair if You Control Both Brokers” on page 373](#). If you do not control both Broker Servers, you must create and configure your territory gateway as described in [“Creating a Territory Gateway Pair if You Control Only One Broker ” on page 377](#), and the administrator of the other Broker Server must create and configure its counterpart in the other territory. The territory gateway connection will not function until both gateways are configured.

Creating a "One-Way" Territory Gateway

Organizations often want to create what is known informally as a "one-way" territory gateway, meaning that they want documents to flow only in one direction. For example, a territory that handles employee-related documents might want to broadcast certain documents to, but never receive documents from, a territory that handles customer support documents.

To create this type of territory gateway, you must create a complete gateway connection (that is, you must create and configure *both* gateways in the gateway pair). Creating a one-way territory gateway *does not* mean that you create only one side of a territory gateway connection. To achieve the "one-way" nature of this type of connection, you must configure the permission settings on each territory gateway to restrict the flow of documents to one direction.

For example, if you want to limit the flow of documents from the employee territory (EMP) to the customer support territory (CUST), you would set permissions as follows:

On gateway **EMP>CUST**:

Assign this permission...	To...
Allow Forward	All document types that are permitted to travel to the CUST territory.
Allow Receive	None.

On gateway **CUST>EMP**:

Assign this permission...	To...
Allow Forward	None.
Allow Receive	All documents types that the EMP territory will forward to this territory (CUST), with restrictive filters, if necessary.

Configuring and Controlling Territory Gateway Behavior

You can use My webMethods to configure and control the following territory gateway behavior:

- **Pausing and Resuming a Territory Gateway.** The pause and resume controls enable you to halt the flow of outbound traffic from a territory gateway to the remote territory. For more information about these using controls, see [“Pausing and Resuming a Territory Gateway” on page 389](#)
- **Configuring the Keep-Alive Setting.** The keep-alive setting instructs the territory gateway to issue periodic "keep-alive" messages to the remote territory. You use this option if the territory gateway connects through a firewall that automatically drops connections that are idle for an extended period. For more information about using the keep-alive feature, see [“Preventing Connection Timeouts between Territories” on page 390](#).
- **Configuring the Forwarding Mode.** The forwarding mode determines how a territory gateway maintains the subscription list for the remote territory. For more information about specifying the forwarding mode, see [“Configuring the Forwarding Mode” on page 391](#).
- **Blocking Document Type Updates.** By default, any document type that territories are permitted to exchange is automatically synchronized if either territory makes a change to that document type. You can use the "refuse update" option to suppress

this behavior on one or both territory gateways. For information about blocking updates, see [“Refusing Document-Type Updates” on page 392](#).

Creating a Territory Gateway Pair if You Control Both Brokers

It is easiest to configure a territory gateway if you control both sides of the gateway. In this case, My webMethods can do much of the work for you. You only have to create one territory gateway. My webMethods automatically creates and configures its counterpart in the remote territory.

There are two major steps in creating a territory gateway pair using this procedure.

1. First, you must create the territory gateway pair as described in [“Defining the Territory Gateway Pair” on page 373](#).
2. Next, you must configure the list of document types that the territories are permitted to exchange as described in [“Specifying which Documents Types Territories Can Exchange” on page 374](#).

Before you begin this procedure, verify that you have administrative access to the two Broker Servers that will host the territory gateway pair. These Broker Servers must both appear in your Broker Server List and you must have authority to administer both of them if they are protected by an ACL.

If you have access to only one of the Broker Servers, you must configure the territory gateway in your territory using the procedure in [“Creating a Territory Gateway Pair if You Control Only One Broker ” on page 377](#). An administrator with access to the Broker Server that will host the other territory gateway must use the same procedure to create your territory gateway's counterpart in the remote territory.

Defining the Territory Gateway Pair

Use the following procedure to create a territory gateway pair on two Brokers that you control.

To define a territory gateway pair

1. In My webMethods: **Messaging > Broker Territories > Territory Gateways**
2. Click **Add**.
3. On the **Create Gateway Pair** tab, select the two Brokers that will form the territory gateway connection.
4. Click **Create**.
5. Proceed to the following section to specify which document types each territory gateway is permitted to send and receive. The territory gateway connection will not function until you do this.

Specifying which Documents Types Territories Can Exchange

After you define the territory gateway pair, you must specify which document types can flow from one territory to another. To enable a document to flow between territories, one territory gateway must allow that document type to be *forwarded* and the other territory gateway must allow that type to be *received*.

You can define which document types can travel through a territory gateway in either of the following ways:

- You can manually select the individual document types that the territory gateway can forward or receive as described in [“Specifying Permissions for Receiving Individual Document Types”](#) on page 374.
- You can select one or more client groups that identify the document types that the territory gateway Broker can forward or receive as described in [“Using Client Groups to Specify Permissions on Document Types”](#) on page 376.

With either procedure, you can optionally instruct My webMethods to automatically make the corresponding permission settings on its counterpart in the remote territory. If you do not use this option, you must configure the permissions on the remote territory gateway manually.

Specifying Permissions for Receiving Individual Document Types

Use the following procedure to set document types that a territory gateway can receive from another territory.

To select individual document types that a territory gateway can receive from the other territory

1. In My webMethods: **Messaging > Broker Territories > Territory Gateways**
2. In the **Territory Gateways List**, locate the territory gateway that you want to configure and click  **Details**.
3. Click the **Shared Document Types** tab.
4. To select the document types that this territory gateway Broker can receive from the other territory, do the following:
 - a. Click **Allow Receive** and then click **Add Receive Document Types**.
 - b. On the **Allow Receive Document Types** panel, select the document types that this territory gateway will receive from the other territory.

Note: If you do not see the document type that you need, it means the document type does not exist in one or both territories.

- c. Click **Add**.
5. Select one of the following options and click **Update**.

Enable this option...	If...
Yes. Set Permissions on Other Side of the Gateway	<p>You want My webMethods to automatically set the corresponding "forward" permissions for the selected document types on the remote territory gateway.</p> <p>Do not use this option unless you control both territory gateways.</p>
No	<p>You want to set these permissions only on the local territory gateway. If you select this option, you must manually set "forward" permissions for these document types on the remote territory gateway.</p>

6. If you want to filter the documents that this territory gateway will receive from the other territory, click  Edit Filter for the document type and define the filter as described in [“Filtering Documents” on page 379](#).
7. If this territory gateway Broker will forward documents to the other territory, use the following procedure to specify which document types it will send.

Specifying Permissions for Forwarding Individual Document Types

Use the following procedure to set permissions on individual document types that a territory gateway can forward to another territory.

To select individual document types that a territory gateway can forward to the other territory

1. In My webMethods: **Messaging > Broker Territories > Territory Gateways**
2. In the **Territory Gateways List**, locate the territory gateway that you want to configure and click  **Details**.
3. Click the **Shared Document Types** tab.
4. To select the document types that this territory gateway can forward to the other territory, do the following:
 - a. Click **Add Forward Document Types**.
 - b. On the **Allow Forward Document Types** panel, select the document types that this territory gateway will be allowed to forward to the other territory.
 - c. Click **Add**.
5. Select one of the following options and click **Update**.

Enable this option...	If....
Yes. Set Permissions on Other Side of the Gateway	<p>You want My webMethods to automatically set the corresponding "receive" permissions for the selected document types on the remote territory gateway.</p> <p>Do not use this option unless you control both territory gateways.</p>
No	<p>You want to set permissions only on the local territory gateway. If you select this option, you must manually set "receive" permissions for these document types on the remote territory gateway.</p>

Using Client Groups to Specify Permissions on Document Types

If you have one or more client groups that identify the set of documents that a territory gateway can exchange with the remote territory, you can use those groups to specify which documents are permitted to flow to and from this territory gateway.

When you use a client group to specify the document type permissions for a territory gateway, the can-publish and can-subscribe permissions in the client group are mapped to the territory gateway permissions as follows:

This permission for a document type in the client group...	Sets this permission for the document type on the territory gateway...
Can Publish	Allow Forward (Permits the territory gateway to forward instances of the document type to subscribers in the remote territory.)
Can Subscribe	Allow Receive (Permits the territory gateway to receive instances of the document type from publishers in the remote territory.)
Important: If you want to set permissions on both territory gateways, be sure that all of the document types in the selected client groups exist in both territories.	

To use client groups to specify which document types can flow to and from a territory gateway

1. In My webMethods: **Messaging > Broker Territories > Territory Gateways**
2. In the **Territory Gateways List**, locate the territory gateway that you want to configure and click  **Details**.

3. Click the **Shared Document Types** tab and then click **Hook Up Client Groups**.
4. Use the **Available Client Group** controls to select the client group(s) that contain the document type definitions and permissions that you want to apply to this territory gateway Broker.
5. Select one of the following options and click **Update**.

Enable this option...	If....
Yes. Enable Permissions on Other Side of the Gateway	You want My webMethods to automatically set the corresponding permissions for the specified document types on the remote territory gateway. (Do not use this option unless you control both territory gateways.)
No	You want to set permissions only on the local territory gateway. If you select this option, you must manually set the corresponding permissions for these document types on the remote territory gateway.

Creating a Territory Gateway Pair if You Control Only One Broker

When you configure a territory gateway that crosses administrative domains (such as between two companies), you might not control both Broker Servers that make up the territory gateway pair. In this case, you must create and configure the territory gateway in your territory and the administrator of the other Broker Server must create and configure the territory gateway in the remote territory.

To do this, you and the administrator of the other territory must each perform the following steps:

1. Create the territory gateway for your territory as described in [“Defining One Side of a Territory Gateway Pair” on page 378](#). You will need the following information from the other administrator.
 - The *exact* name of the remote territory
 - The name of the Broker that will host the territory gateway in the remote territory
 - The host name and port number of the Broker Server on which that Broker resides
2. Configure the list of document types that your territory gateway can exchange with the remote territory as described in [“Specifying which Documents the Territory Gateway Can Forward and Receive” on page 378](#).

Defining One Side of a Territory Gateway Pair

Use the following procedure to create one side of a territory gateway connection.

To define one side of the territory gateway pair

1. In My webMethods: **Messaging > Broker Territories > Territory Gateways**
2. Click **Add**.
3. Click the **Create Gateway** tab, and complete the following fields.

In this field...	Do the following...
Local Territory	Select the Broker on which you want the territory gateway created.
Remote Territory	Specify the name of the territory with which this territory gateway will exchange documents. Type the name exactly as it is defined in the other territory. This value is case sensitive. Example: OrdersTerritory
Remote Broker Server	Specify the host name and port number of the Broker Server that hosts (or will eventually host) the remote territory gateway. (The remote territory gateway does not have to exist at this point, but your territory gateway must know where the remote territory gateway will eventually reside.)
Remote Broker	Specify the name of the Broker that will host the territory gateway in the remote territory. Example: Orders

4. Click **Create**.

Specifying which Documents the Territory Gateway Can Forward and Receive

If you do not have administrative access to the remote territory, you and the administrator of the remote territory must use the procedures described in [“Specifying which Documents Types Territories Can Exchange” on page 374](#) to manually specify the permissions for the document types that the territories can exchange.

To ensure that documents travel across the territory gateway connection properly, permissions must be set correctly on each territory gateway. The document types that your territory will receive from the remote territory must be listed in your territory gateway's "Allow Receive" list. Additionally, these document types must appear in the remote territory gateway's "Allow Forward" list. Likewise, you must add to your territory gateway's "Allow Forward" list each document type that appears in the remote territory gateway's "Allow Receive" list. If a document type does not appear in the proper lists on both territory gateways, it will not be communicated across the territory gateway connection.

Using Territory Gateway Filters

A filter string specifies criteria for the contents of a document. The Broker uses the filter string to determine which documents match your criteria and allows only those documents that match the filter string to pass through the territory gateway.

You apply a filter string to document types that a territory gateway *can receive* from a remote territory.

To place a filter on a document type

1. In My webMethods: **Messaging > Broker Territories > Territory Gateways**
2. In the **Territory Gateways List**, locate the gateway that you want to configure and click  **Details**.
3. Click the **Shared Document Types** tab.
4. In the list of the document types that the territory gateway is permitted to receive, click  **Edit Filter** for the document type that you want to filter.
5. In the **Filter** text box, type a filter string that specifies the criteria that identifies which documents this territory gateway can forward to the remote territory. For information about specifying a filter string, see ["Filtering Documents" on page 379](#).
6. Click **OK**.

The filter appears in the **Filter** column in the **Allow Receive** table.

Filtering Documents

A filter string specifies criteria for the contents of a document. Filter strings can do any combination of the following:

- Compare document field contents against constants or computed values.
- Combine document field comparisons using the boolean operators `and`, `or`, and `not`.
- Perform arithmetic operations on document fields and constants.

- Contain regular expressions.
- Contain string and arithmetic constants.
- Contain a hint that specifies how documents should be processed.

For information about using regular expressions, hints, and filter functions from the `webMethods Broker` library, see the *webMethods Broker Client Java API Programmer's Guide*.

Filter Sample - Filtering JMS documents based on the value set on the document header

If you want to filter JMS documents that contain `JMSType` set to `name` in the header, set the following gateway filter.

```
_env.jms_type = "name"
```

Filter Sample - Filtering JMS documents based on a property value

If we want to set up a filter to match JMS documents that contain `jmsuser` property set to `admin`, set the following gateway filter

```
Properties.jmsuser = "admin"
```

Filter Sample - Filtering documents based on person's age and state of residence

Assume that a document contains a person's age and state of residence. The following filter string matches only those documents whose `age` field is greater than 65 and whose `state` field is equal to FL.

```
age > 65 and state = "FL"
```

In this example filter string, `age` and `state` represent document fields. This filter also contains an arithmetic constant 65 and a string constant "FL". The boolean operator `and` combines the field criterion for `age` and `state`.

Filter Rules

Filter strings must adhere to the following rules:

- Field names can be fully qualified, such as:

```
struct_field.seq_field[2]
```

- A character constant is a single character surrounded by single quotation marks. For example:

```
'A'
```

- A string constant is zero or more characters surrounded by double quotation marks. For example:

```
"account"
```

- If a character or string constant contains a single or double quotation mark, precede the quote with a backslash. For example:

```
'\"'
```

- You can use parentheses to control the order of operator precedence.

Filter Operators

The following tables contain the various operators that you can use to create filters. For a more complete list of, see the *webMethods Broker Client Java API Programmer's Guide*.

Note: The Integration Server and Software AG Designer use different filter syntax for subscribing to publishable documents. See the *Software AG Designer Online Help* for more information.

The following table shows the logical filter operators:

Note: Logical filter operator expressions are evaluated using a method similar to SQL expression evaluation, in that all operators are always evaluated. When a logical filter operator expression contains multiple operators, operator precedence determines the sequence in which the operations are performed. For example, when evaluating the expression "A OR B", both "A" and "B" are evaluated, even if "A" evaluates to "true".

Operator	Description
! not	Not
&& and	And
 or	Or

The following table shows the comparison filter operators:

Operator	Description
<	Less than
<=	Less than or equal to
>	Greater than
>=	Greater than or equal to
=	Equal to
==	Equal to

Operator	Description
!=	Not equal to

The following table shows the arithmetic filter operators:

Note: Implicit type conversion occurs when operands in an arithmetic operation have different types. The operands are converted to a larger value before the comparison occurs. Type `char` is considered numeric, but `boolean` is not.

Operator	Description
-	Unary minus
*	Multiplication
/	Division
%	Modulus Division
-	Subtraction
+	Addition

The following table shows the String operators:

Operator	Description
+	Concatenation
<	Less than
<=	Less than or equal to
>	Greater than
>=	Greater than or equal to
=	Equal to
==	Equal to

Operator	Description
!=	Not equal to

Viewing Information about a Territory Gateway

The Territory Gateways page displays a list of territory gateways that are known to the Broker Servers in your Broker Server List. This page provides general information about the territory gateways. You can view detailed information about the territory gateway on the Territory Gateway Details page.

To view information about a territory gateway

- In My webMethods: **Messaging > Broker Territories > Territory Gateways**

The **Territory Gateways List** displays the following information about each territory gateway that resides on a Broker Server that is part of your Broker Server List.

Column	Description
Local Territory > Remote Territory	The name of the territory gateway.
Local Broker	The Broker that hosts the territory gateway. Click this name to display the Territory Broker Information page for that Broker.
Remote Broker	The Broker that hosts this territory gateway's counterpart in the remote territory. Click this name to display the Territory Broker Information page for that Broker.
Status	Indicates the state of the local Broker's connection to the remote Broker as follows:

- **Active.** The territory gateway connection has been configured and is operational (that is, not paused).

Note: The **Active** status indicates that the territory gateway connection is logically available, but it does not necessarily mean that the two territory gateway Brokers are physically connected at that point in time. To determine whether the two Brokers are physically connected, go to step 2 and examine the **Linked** field on the Territory Gateway Details page.

Column	Description
	<ul style="list-style-type: none"> ■ Error. The territory gateway connection is not available for use. This state usually indicates a problem with the territory gateway configuration. <p>Note: You can hover over the icon for information about the error.</p> <ul style="list-style-type: none"> ■ Paused. This territory gateway is paused and is not currently forwarding any documents to the other territory. For information about pausing and unpausing a territory gateway, see “Pausing and Resuming a Territory Gateway” on page 389.

- Click  **Details** to display the Territory Gateway Details page for a territory gateway. The **Configuration** tab on this page displays the following information:

Field	Description
Brokers are Linked	<p>Indicates whether the Brokers that host the two territory gateways are physically connected as follows:</p> <ul style="list-style-type: none"> ■ Yes. Brokers are connected and able to exchange documents. ■ No. This territory gateway cannot connect to the other Broker.
Status	Indicates the state of the territory gateway connection. See Status description under preceding step.
Keep Alive Interval	<p>How often (in seconds) this territory gateway issues "keep-alive" events to the remote territory gateway. A territory gateway can use keep-alive events to prevent a firewall from disconnecting what it considers to be an idle connection. For more information about using keep-alive events, see “Preventing Connection Timeouts between Territories” on page 390.</p> <p>If a keep-alive interval of 0 is specified, this field displays "Disabled."</p>
Queue Length	Number of documents currently in the forward queue waiting to be retrieved by the remote territory gateway in the other territory.

Field	Description
Shared Document Types	Number of documents types that are shared and can, therefore, be exchanged through this territory gateway. Click this value to view the document types.
Authentication Type	<p>Indicates whether this gateway is SSL-enabled.</p> <ul style="list-style-type: none"> ■ None. The gateway is not SSL-enabled. The Broker Server on which it is hosted does not have an SSL identity and, therefore, the gateway cannot interact with the remote gateway using SSL. It will connect to the remote gateway using the non-SSL port. ■ SSL. The gateway is SSL-enabled. The Broker Server on which it is hosted has an SSL identity and it can interact with the remote gateway using SSL.
Access Control	<p>Indicates whether this gateway is secured with an access control list (ACL).</p> <ul style="list-style-type: none"> ■ An access control list is assigned to this gateway. For more information about configuring an ACL on a gateway, see “Territory Gateway ACLs” on page 331. ▲ An ACL has not been assigned to this gateway. To create one, you must have an SSL identity that belongs to the "admin" client group for the Broker that hosts this gateway.
Static Gateway Forwarding	Specifies whether this gateway maintains a static or dynamic subscription list for the remote territory. For more information about selecting a forwarding mode, see “Configuring the Forwarding Mode” on page 391 .
Encryption	Specifies whether this gateway will encrypt its interactions with the remote gateway.
Refuse All Shared Document Type Updates	Specifies whether this gateway accepts document-type updates from the remote territory. For more information about allowing or blocking updates to document type definitions, see “Refusing Document-Type Updates” on page 392 .

Viewing Information about Documents Flowing between the Gateways

Use the following procedure to view information about the activity between the two territories.

Note: You can also view a graphic representation of the territory gateway connection and observe documents actively flowing across it using the territory topology view. For more information, see [“Topology View of Territories” on page 275](#).

To view information about documents flowing through a territory gateway

1. In My webMethods: **Messaging > Broker Territories > Territory Gateways**
2. Click  **Details** to display the Territory Gateway Details page.
The **Queue Length** field on the **Configuration** tab reports the number of documents currently in the forward queue for the remote territory gateway.
3. If you want information about the number of documents that the remote territory has received from the local territory gateway, click the **Remote Broker** tab and examine the following statistics:

Field	Description
Recent Receipts	The number of documents that the remote territory has retrieved from its forward queue on this territory gateway during the last statistic collection interval. By default, the statistics collection interval represents one minute, however, you can configure this interval in My webMethods. For procedures, see the Time interval between statistical refresh parameter in “Configuring the Connection Parameters” on page 53 .
Total Receipts	The number of documents that the remote territory has retrieved from its forward queue on this territory gateway since the territory gateway was created.
Last Receipt	The time at which the remote territory last retrieved a document from its forward queue on this territory gateway.

4. If you want information about the number of documents that the local territory gateway has received from the remote territory, click the **LocalBroker** tab and examine the following statistics:

Field	Description
Recent Receipts	<p>The number of documents that this territory gateway retrieved from the remote territory during the last statistic collection interval.</p> <p>By default, the statistics collection interval represents one minute, however, you can configure this interval in My webMethods. For procedures, see the Time interval between statistical refresh parameter in “Configuring the Connection Parameters” on page 53.</p>
Total Receipts	The number of documents that this territory gateway has retrieved from the remote territory since the territory gateway was created.
Last Receipt	The time at which this territory gateway last retrieved a document from the remote territory.

Deleting a Territory Gateway Connection

To remove a territory gateway connection between two territories, you delete each member of the territory gateway pair. If you do not control the remote territory gateway, you should coordinate the removal of the territory gateway with the administrator of the remote Broker.

When you delete a territory gateway, you delete all documents that are currently queued for the remote territory gateway.

After one side of the territory gateway is deleted, the other side should be deleted promptly. Otherwise, it will continue to accumulate documents in its forward queue and can eventually consume a considerable amount of memory and/or queue storage.

Important: If you control only one side of the territory gateway pair and find it necessary to remove the territory gateway, there is no simple method to restore the connection. To restore the territory gateway, you must recreate the territory gateway and manually reconfigure the "allow forward" and "allow receive" permissions.

To delete a territory gateway connection

1. In My webMethods: **Messaging > Broker Territories > Territory Gateways**

2. In the **Territory Gateways List**, select the check box beside the territory gateway(s) that you want to delete.
3. Click **Delete**.

If the other member of a territory gateway pair is not under your control, the administrator of the remote territory gateway must perform this same series of steps.

Removing an Allowed Document Type from a Territory Gateway

Use the following procedure to remove a document type from the list of document types that a territory gateway is allowed to forward or receive.

If you control both territory gateways in the territory gateway pairs, you can remove the document type from both territory gateways at the same time. If you do not control the remote territory gateway, you must have an authorized administrator delete the document types from the remote territory gateway.

To remove a document type from a territory gateway

1. In My webMethods: **Messaging > Broker Territories > Territory Gateways**
2. Click  **Details** to display the Territory Gateway Details page and then click the **Shared Document Types** tab.
3. If you want to remove a document type from the list of types that this territory gateway can receive, click **Allow Receive**.
4. Select the check box beside the document type that you want to remove from the allowed list.
5. Click **Delete**.
6. Select one of the following options and click **Update**.

Enable this option...	If....
Yes. Remove document type(s) on other side of gateway	<p>You want My webMethods to automatically remove the document type from the opposite permission list on the remote territory gateway.</p> <p>Do not use this option unless you control both territory gateways.</p>
No	You want to remove the document type only from the local territory gateway. If you select this option, the

Enable this option...	If...
	document type must be manually removed from the opposite list on the remote territory gateway.

Pausing and Resuming a Territory Gateway

You can temporarily halt the flow of documents from a territory gateway to a remote territory by pausing the territory gateway. When you pause a territory gateway, the gateway locks the forward queue that it maintains for the remote territory. This action prevents the remote territory from retrieving documents from this territory gateway.

No documents are lost when you pause a territory gateway. The forward queue on the territory gateway continues to accumulate documents for the remote territory, but these documents will not flow to the other territory until the territory gateway is resumed.

Be aware that pausing a territory gateway halts the flow of *outbound* documents (documents that originated from the territory on which the territory gateway resides and are being forwarded to subscribers in the remote territory). It does not prevent that territory gateway from receiving documents from the remote territory. To halt the flow of inbound documents, you must pause the territory gateway in the remote territory, too.

Note: Administrators often pause a territory gateway when the territory that the territory gateway represents is undergoing service or significant configuration changes. For example, if you intend to make numerous changes to document types or permissions, you might pause the territory to ensure that all the changes are propagated to the other territory at one time.

Pausing a Territory Gateway

Use the following procedure to pause a territory gateway.

To pause a territory gateway

1. In My webMethods: **Messaging > Broker Territories > Territory Gateways**
2. Click  **Details** for the territory gateway whose outbound flow of documents you want to pause.
3. On the **Configuration** tab, click **Pause Forwarding**.

The **Status** field switches to  **Paused** and indicates the ID of the client that placed the pause on the territory gateway.

Important: Documents will accumulate in the territory gateway's forward queue for the remote territory. If the territory gateway is paused for a significant

amount of time, the number of documents in the queue could begin consuming a great deal of memory or disk storage resources.

Resume a Territory Gateway

Use the following procedure to resume a territory gateway.

To resume a territory gateway

1. In My webMethods: **Messaging > Broker Territories > Territory Gateways**
2. Click  **Details** for the territory gateway whose outbound flow of documents you want to restore.
3. On the **Configuration** tab, click **Resume Forwarding**.

The **Status** field switches to  **Active**, indicating the outbound documents are permitted to flow to the other territory.

Preventing Connection Timeouts between Territories

If your territory gateway connects to a remote territory through a firewall, you might need to activate the territory gateway's "keep-alive" feature to prevent the firewall from dropping the connection after a period of inactivity. When you enable the keep-alive feature, the territory gateway publishes "keep alive events" to the remote territory at a specified frequency. The keep-alive events prevent the connection from appearing idle to the firewall.

Important: Do not set a keep alive interval that is too short or you will create a lot of unnecessary traffic across the network. Strive for a value that is just long enough to prevent the connection from timing out.

Be aware that enabling this feature issues keep-alive messaging in one direction. That is, it instructs the territory gateway on which you enable the feature to issue keep-alive messages to the remote territory. If you need to activate keep-alive messages in both directions, you must activate the keep-alive feature on both territory gateways.

To implement the keep-alive feature correctly, you must understand the conditions that cause the firewall to drop a connection. In most cases, you will need to coordinate with the network administrators at each end of the territory gateway to obtain information about the behavior of the firewalls. Given this information, you can determine which territory gateway must issue keep-alive messages (it may be both) and the frequency at which the messages need to be issued to prevent the connection from dropping.

Viewing and Modifying the Keep-Alive Setting

Use the following procedure to view or modify the keep-alive setting.

To view or modify the keep-alive setting

1. In My webMethods, **Messaging > Broker Territories > Territory Gateways**
2. Click  **Details** for the territory gateway whose keep-alive setting you want to configure.
3. On the **Configuration** tab, examine the **Keep Alive Interval** field. It will display either the amount of time between keep-alive events, in seconds, or the word "Disabled" if the keep-alive feature is not activated.
4. If you want to modify or disable the keep-alive setting, do the following:
 - a. Click the current **Keep Alive Interval** value to display the Territory Gateway Change Keep Alive Interval page.
 - b. In **Keep Alive Interval (seconds)**, specify the frequency at which you want the territory gateway to publish keep-alive messages. Specify zero seconds if you want to disable the keep-alive feature.
 - c. Click **OK**.

Configuring the Forwarding Mode

A territory gateway maintains a subscription list that identifies the types of documents the remote territory wants to receive. This subscription list tells the territory gateway which documents it needs to forward to the remote territory. The *forwarding mode* determines how the territory gateway populates and maintains this subscription list.

- In *dynamic forwarding mode*, a territory gateway maintains the subscription list for a remote territory based on subscription events that the remote territory gateway issues when clients in its territory add and drop subscriptions to a document type that the territories share. Broker Servers earlier than version 6.5 use only dynamic forwarding mode.
- In *static forwarding mode*, a territory gateway automatically populates the subscription list based on the document types that the territory gateway is permitted to forward to the remote territory. Static forwarding mode is the default mode in Broker Servers 6.5 and later.

In static mode, it is important to configure the allow-forward permissions carefully. If the list includes document types for which there are no actual subscribers in the remote territory, those documents will be forwarded to the remote territory unnecessarily, wasting processing time, queue space, and network bandwidth. For information about configuring the allow-forward permissions, see [“Specifying which Documents Types Territories Can Exchange” on page 374](#).

Note: If you are running a Broker Server that supports both modes, we strongly recommend that you use static forwarding mode (the default). It is less prone to subscription propagation failures relating to network outages and timing factors.

To view or configure the forwarding mode

1. In My webMethods: **Messaging > Broker Territories > Territory Gateways**
2. Click  **Details** for the territory gateway whose forwarding option you want to view or configure.
3. On the **Configuration** tab, enable or disable the **Static Gateway Forwarding** option.

If the territory gateway is running on a Broker Server that does not support static territory gateway forwarding, you will not be able to enable this option.

Refusing Document-Type Updates

Territories that are connected by a territory gateway connection automatically exchange and apply updates to the document types that they are configured to exchange. For example, if territory A is configured to receive a document type "CancelOrder" from territory B, and an administrator in either territory modifies that document type, the change is automatically propagated to the other territory.

If you want your territory gateway to block document type changes from the remote territory, you can enable the **Refuse Document Type Updates** option. When this option is enabled and the remote territory publishes a change to a document types, the following occurs:

- The document type change is rejected by the territory gateway and the document type is marked "out of sync."
- An alert is written to the log file associated with the Broker Server on which the territory gateway is running.
- The territory gateway halts inbound traffic (that is, it stops retrieving all documents from the remote territory).

When a territory gateway enters this condition, it will not begin accepting documents again until one of the following occurs:

- The territory gateway's document type is updated to match the one on the remote territory.

-OR-

- The remote territory modifies its document type to match the one on this territory gateway.

You can block document-type updates globally or by individual document type.

Viewing or Configuring the Global Refuse Update Option

Use the following procedure to view or configure the global Refuse Update option.

To view or configure the global Refuse Update option

1. In My webMethods: **Messaging > Broker Territories > Territory Gateways**
2. Click  **Details** for the territory gateway whose update option you want to view or configure.
3. On the **Configuration** tab, enable or disable the **Refuse all shared document type updates**.

Note: Disabling this option does not change any update settings that are applied to specific document types. Those settings will remain in effect.

4. Click **Apply**.

Viewing or Configuring the Refuse Update Option for an Individual Document Type

Use the following procedure to view or configure the Refuse Update option for an individual document type.

To view or configure the Refuse Update option for individual document types

1. In My webMethods: **Messaging > Broker Territories > Territory Gateways**
2. Click  **Details** for the territory gateway whose document types you want to view or configure.
3. Click the **Shared Document Types** tab and use the **Allow Receive** and **Allow Forward** links to display the list in which the document type appears. If the document type appears in both lists, you can configure the update option on either list.
4. To determine whether the territory gateway currently accepts or rejects updates for the document type, examine the value in the **Refuse Updates** column.
5. If you want to change the document type's **Refuse Update** setting, do the following:
 - a. Select the check box beside the document type whose update option you want to change.
 - b. Click **Refuse Update** or **Allow Update**, depending on whether you want the territory gateway to reject or permit updates to the selected document type.

Using Broker Remote Publish

Remote publish allows a publisher to restrict the distribution of a document. Documents are published to the clients and territory gateways of only one remote Broker; therefore, distribution to other Brokers is limited, depending on the circumstance.

Remote publish is accessed by delivering documents to a particular client ID on a remote Broker. When the remote Broker receives the document, this client ID triggers the remote Broker to treat the document as if it had been published by the delivering Broker.

The `_env.destId` field is removed and the document is enqueued to clients and territory gateways with matching subscriptions. If the remote Broker is on the other side of a territory gateway, then the behavior varies slightly, as summarized in the table below.

Target Remote Broker	Document Destination
Broker in same territory	<ul style="list-style-type: none"> ■ All clients on remote Broker ■ All territories connected via territory gateways local to the remote Broker
Broker peer of local territory gateway	<ul style="list-style-type: none"> ■ All clients on remote Broker ■ All Brokers in the remote territory ■ All territories connected to the remote territory, except the territory of the delivery Broker
Broker in different territory	<ul style="list-style-type: none"> ■ All clients on remote Broker ■ All territories connected via territory gateways local to the remote Broker
Same as delivering Broker	<ul style="list-style-type: none"> ■ Same as if the document had been published

To perform a remote publish, you deliver documents to the `client-id:publish` on a remote Broker. For example:

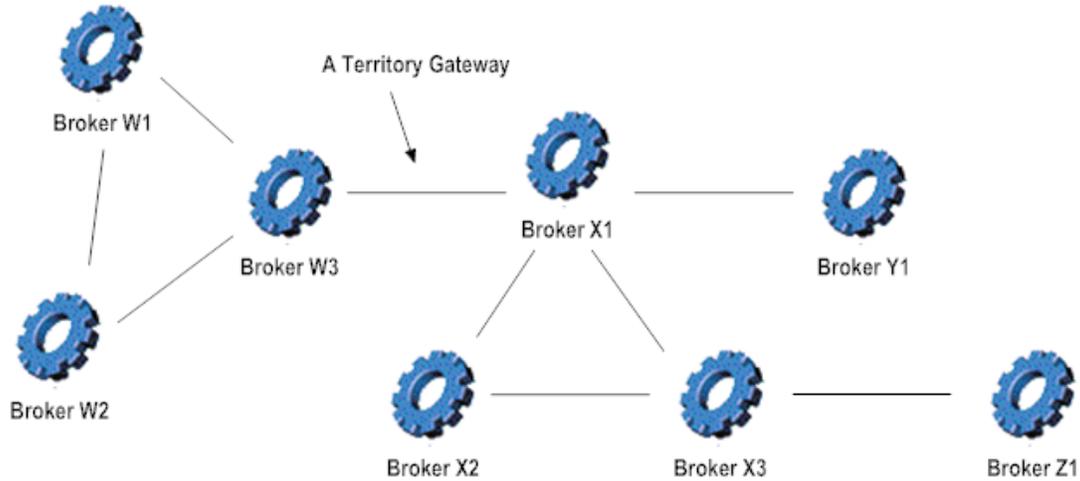
```
BrokerClient bc;
BrokerEvent event;
[...]
bc.deliver( "/T/OtherBroker/:publish", event);
```

If you know that the remote Broker is in the same territory as the delivering Broker, then you can omit the territory:

```
bc.deliver( "//OtherBroker/:publish", event);
```

The following examples are based on the Brokers and territories shown in the Broker Territories diagram. The first letter of the Broker's name indicates its territory.

Broker Territories



Broker in the Same Territory

Example 1: The delivering client is on Broker W1. The target Broker is `/w/w2/:publish`. A document is only published to clients of Broker W2.

Example 2: The delivering client is on Broker W1. The target Broker is `/w/w3/:publish`. A document is published to clients of Broker W3, which also sends the document to X1, where it will be distributed as a published document throughout territories X, Y, and Z.

Broker Peer of Local Territory Gateway

The delivering client is on Broker W3. The target Broker is `/x/x1/:publish`. A document is published throughout the X, Y, and Z territories. Territory Y has a local territory gateway on X1, and territory Z is reached when X3 receives the document.

Broker in a Different Territory

Example 1: The delivering client is on Broker W1. The target Broker is `/x/x2/:publish`. A document is only published to clients of Broker X2.

Example 2: The delivering client is on Broker W1. The target Broker is `/y/y1/:publish`. A document is only published to clients of Broker Y1. A document will never back track from where it came, so Broker Y1 will not send the document to X1.

Example 3: The delivering client is on Broker W1. Target Broker is `/x/x3/:publish`. A document is only published to clients of Broker X3 and to local territory gateways of X3 (Z1 in this case). If territory Z had more Brokers and territory gateways, they would also receive the document.

Same as Delivering Broker

The delivering client is on Broker W1. The target Broker is `/W/W1/:publish`. A document is published to clients of Broker W1, all Brokers of territory W, and all territories reachable, via territory gateways. The behavior is identical to publishing the document.

Note: A document using remote publish will look like a delivered document until it reaches the target Broker. Trace documents and activity traces will record the document as a delivery. The remote publish trace on the target Broker will also record the document as a delivery, but the enqueue traces will look like a publish occurred.

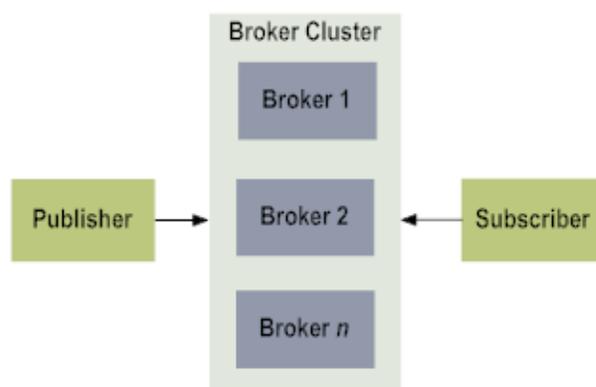
16 Managing Clusters

■ Introduction to Clusters	398
■ Creating Clusters	400
■ Adding Brokers to a Cluster	402
■ Removing a Broker from a Cluster	403
■ Deleting a Cluster	403
■ Viewing a List of Known Clusters	403

Introduction to Clusters

A cluster is a group of Brokers that functions as one logical Broker. All Brokers in a cluster maintain the same set of *document types* and *client groups*; however, they each support their own set of *clients*.

Clients can connect to a cluster and publish documents to or receive documents from a cluster. The cluster to which a client publishes documents takes care of forwarding those documents to subscribers that are connected to the cluster. To an individual publisher or subscriber, a cluster appears as one large Broker.



Brokers that operate in a cluster are *peers*. There is no "central" or "controlling" Broker. Each Broker in the cluster is responsible for notifying its peers of changes (for example, additions, deletions, or modifications) that occur to its document types or client groups. By dynamically propagating changes to all Brokers in this manner, document types and client groups remain synchronized across the entire cluster. Whenever a Broker joins or leaves a cluster, the peers are informed. When clients are deleted from a Broker in a cluster, other Brokers are not notified.

Why Implement a Cluster?

When the publishing load increases, you scale up client-side message distribution by creating a cluster of Brokers to handle the increasing load. Clusters enable scalability and availability of your messaging system. A cluster is similar to a territory, except that document forwarding within a cluster is disabled. The JMS subscriber is connected to all the Brokers in a cluster, and receives the documents published to any of the Brokers in the cluster.

Clusters facilitate client-side load balancing and failover when you use JMS cluster policies for client connections. For more information, see the *webMethods Broker Messaging Programmer's Guide*.

For additional flexibility, you can create multiple clusters and link them together using *cluster gateways*. A cluster gateway is a connection that allows specified documents to

flow between two clusters. For more information about cluster gateways, see [“Managing Cluster Gateways” on page 405](#).

How Brokers in a Cluster Communicate with Each Other

Each Broker in a cluster is directly connected to each of its peers. For example, in a cluster that consists of Brokers A, B, C and D, Broker A has a connection to Brokers B, C, and D; Broker B has connections to A, C, and D; Broker C has a connection to Brokers A, B, and D, and so forth.

How Delivered Documents Are Handled in a Cluster

The native clients receive documents published or delivered to the Broker. The documents published on other Brokers in the cluster are not received by native clients.

A Broker will not forward a delivered document to other Brokers in a cluster unless the document is addressed to a client by its fully-qualified client ID (in the form */clusterName/BrokerName/ClientID*). If a Broker receives a delivered document that is addressed to an unqualified clientID, it delivers the document locally. If the addressee does not exist on the local Broker, the document is sent to the dead-letter queue (if one is configured) or is discarded.

How Brokers Synchronize Metadata

To function as one logical Broker, all the Brokers in a cluster must have identical document-type and client-group information.

When a Broker initially joins a cluster, its document types and client groups (including ACL settings) are synchronized with the other Brokers in the cluster. The clients are not a part of the Broker metadata. The cluster does not synchronize the client state information across the Brokers in the cluster.

After a Broker becomes a member of a cluster, it is responsible for notifying its peers of any changes that occur to its document types or client groups. When a Broker in a cluster receives such notification, it immediately applies the change to its own metadata. Because these metadata changes are self-propagating, you can make a change to a document type or client group on any Broker and that change will automatically be applied to all Brokers in the cluster.

Note: Because it takes a certain amount of time for a change to propagate across the entire cluster, if two administrators make a change to the same object on two different Brokers at nearly the same time, it is possible that one administrator's changes will be overwritten by the other's. To avoid this possibility, we suggest that you appoint a single administrator to manage updates to the document type and client group metadata for the cluster.

Exporting and Importing Cluster Information

You can export and import metadata about a Broker cluster and its dependent objects in the format of an XML file. For information about this feature, see [“Exporting Clusters” on page 490](#).

Securing a Cluster

Within a cluster, it is recommended that all Brokers use SSL or no Brokers use SSL. Similarly, if encryption is enabled, it is enabled for all connections between Brokers in the cluster.

When the Broker Servers in a cluster are SSL-enabled, you can use an access control list (ACL) to identify which Brokers are authorized members of the cluster. Brokers that are not listed in the ACL, are not allowed to join or participate in the cluster.

For more information about securing a cluster, see [“Cluster ACLs” on page 330](#).

Leaving a Cluster

To remove itself from a cluster, a Broker must formally "leave" the cluster. When a Broker leaves a cluster, it informs its peers (at least one of them) about leaving. When the last Broker leaves a cluster, that cluster ceases to exist.

Unless it is the only remaining Broker in a cluster, it is important that at least one of the Broker's peers be running when you remove a Broker from a cluster. Otherwise, the state of the cluster will be inconsistent.

Creating Clusters

This section explains how to create clusters.

Requirements Checklist

Before you can create a cluster, make sure the following requirements are met:

- The domain name service (DNS) used by the host machines that participate in the cluster must be capable of *bi-directional name resolution*. This means that your DNS must be able to resolve the name of the host machine to the correct IP address and also be able to resolve the machine's IP address back to the correct name.

When a host machine uses the Dynamic Host Configuration Protocol (DHCP) to dynamically obtain its IP address, many DNS servers will not return the correct name for a given IP address. In this case, you must contact your IT department or your network administrator and ask that a static IP address be assigned to the host machine.

- Each Broker must have a name that is unique within the cluster.
- Each cluster must have a unique name.
- A Broker must not belong to another cluster. To become part of the new cluster, it must first leave its current cluster. (For similar reasons, you cannot merge clusters. To create a single cluster where two existed before, the Brokers in one cluster must leave it and then join the second cluster.)
- All Broker Servers that participate in the cluster must have the same SSL capabilities. That is, they must all be SSL-enabled or non-SSL enabled.

Creating a Cluster

To create a new cluster, you use the procedure below to give the cluster a name and add the first member Broker to it. Then you use the procedure in [“Adding Brokers to a Cluster” on page 402](#) to add additional Brokers to the cluster.

Note: If all the member Brokers reside on the same Broker Server (which might be the case in a development environment), you can create the entire cluster using just the following procedure.

To create a cluster

1. In My webMethods: **Messaging > Broker Clusters > Clusters**
2. Click **Add Cluster**.
3. In **Cluster Name**, type a name for the cluster.
 - The name you choose must be unique among all clusters that will be connected using cluster gateways.

Tip: We recommend that you assign unique names to clusters even if you do not plan to connect them using a cluster gateway. Doing this will ensure that you can easily distinguish one cluster from another in My webMethods.

- The name cannot contain the following characters:
@ / :
 - The first character must not be #.
 - The name cannot exceed 255 characters.
Example: AlertPublish05
4. Use the **Select Brokers** controls to specify the Brokers that you want to add.

Note: Only Brokers 8.0 and later version that are not already members of another cluster or territory, appear in the **Available** list. If the Broker that you want

to add to the cluster does not appear in this list, check whether it is already a member of another cluster or territory.

5. Click **OK**.
6. If you want to assign an ACL to this cluster so that only authorized Brokers can join it, perform the procedures in [“About Controlling Which Brokers Can Join a Cluster” on page 341](#). All Broker Servers in the cluster will need to be SSL-enabled to use this feature. For information about SSL-enabling a Broker Server, see [“Configuring SSL for Broker Server ” on page 317](#).
7. Use the procedure in [“Adding Brokers to a Cluster” on page 402](#) to add Brokers from other Broker Servers to the cluster.

Adding Brokers to a Cluster

Use the following procedure to add one or more Brokers to an existing cluster. (This action is also known as "joining" a cluster.)

Note: To use this procedure, the cluster to which you want to add the Broker must already exist. If the cluster does not already exist, see [“Creating a Cluster” on page 401](#).

When you add a Broker to a cluster, its client groups and document types are automatically synchronized with the other Brokers in the cluster. If the join process encounters conflicting definitions that it cannot reconcile, you will receive an error message. To add the Broker to the cluster, you must manually reconcile the conflict or accept modifications that the join process proposes.

Before you add a Broker to a cluster, review the [“Requirements Checklist” on page 400](#) and be certain it meets the requirements outlined in the checklist.

To add a Broker to a cluster

1. In My webMethods: **Messaging > Broker Clusters > ClusterBrokers**.
2. In the **Broker Clusters List** click **Add Broker**.
3. From the **Cluster to Join** list on the Add Cluster Broker page, select the cluster to which you want to add a Broker.
4. Use the following steps to select the Brokers to add to the cluster:
 - a. In the **Broker Server** field, select the Broker Server that hosts the Brokers that you want to add.
 - b. Use the **Selected Brokers** controls to specify the Brokers that you want to add to cluster.

Note: Only 8.0 and later version Brokers that are not already members of another cluster or territory appear in the **Available** list. If the Broker

that you want to add to the cluster does not appear in this list, check whether it is already a member of another cluster or territory.

5. Click **Join**.

If the join process encounters conflicting client group or document type definitions during the synchronization process, you will receive an error message. To join the cluster, you can either manually edit the metadata or accept changes proposed by the join process.

Removing a Broker from a Cluster

Use the following procedure to remove a Broker from an existing cluster.

Important: Unless you are removing the last Broker from a cluster, you should ensure that at least one other Broker in the cluster is running when you perform this procedure.

To remove a Broker from a cluster

1. In My webMethods: **Messaging > Broker Clusters > ClusterBrokers**.
2. In the **Cluster Brokers List** page, click the cluster whose Brokers you want to remove.
3. In the **Cluster Information** page, select the check box beside the Broker you want to remove from the cluster.
4. Click **Leave Cluster**.

Deleting a Cluster

You can dissolve a cluster by simply removing every Broker from the cluster. When the last Broker has been removed from the cluster, the cluster will cease to exist.

To remove a Broker from a cluster, see [“Removing a Broker from a Cluster” on page 403](#).

Viewing a List of Known Clusters

You can use the following procedure to view a list of the clusters that are known to the Broker Servers that you are managing (that is, the Broker Servers that appear in your Broker Server List in My webMethods).

To view the list of known clusters

- In My webMethods, select: **Messaging > Broker Clusters > ClusterBrokers**.

The **Broker Clusters List** displays a list of known clusters and their known members. (This page might take a few moments to load while My webMethods scans all known Broker Servers and Brokers for clusters.)

Note: The Brokers listed under each cluster do not necessarily represent the complete membership of the cluster. If member Brokers reside on Broker Servers that are not in your Broker Server List, those Brokers will not appear in the list.

Column	Description
Cluster>Broker@Server	The name of the Broker and the Broker Server on which it resides. Click the name to display details about the Remote Broker objects on this Broker.
Connected	The state of the Broker as described in “Viewing Basic Operating Statistics for the Broker” on page 148.
Connections	The number of clients currently defined on this Broker. This number includes explicit-destroy clients that are not currently connected.
Recent Deliveries or Total Deliveries	<ul style="list-style-type: none"> ■ Recent Deliveries displays the number of publish or deliver requests this Broker received during the last statistics collection interval. The value of the Time interval between statistical refresh field specifies the statistics collection interval. <p>Note: Recent Deliveries data is available only if you have selected Enable Statistical Polling.</p> ■ Total Deliveries displays the total number of publish or deliver requests this Broker received. <p>Note: Total Deliveries data is available only if you do not select Enable Statistical Polling.</p>
Gateway	The list of cluster gateways, if any, that are hosted by this Broker.

17

Managing Cluster Gateways

■ Overview	406
■ Creating a Cluster Gateway	409
■ Configuring and Controlling Cluster Gateway Behavior	411
■ Creating a Cluster Gateway when You Control Both Sides	412
■ Exchanging Document Types across a Cluster Gateway	413
■ Creating a Cluster Gateway when You Control Only One Side	417
■ Filtering Documents across a Cluster Gateway	418
■ Viewing Cluster Gateway Details	422
■ Editing Cluster Gateway Details	425
■ Viewing Document Details across a Cluster Gateway	425
■ Deleting a Cluster Gateway	427
■ Removing an Allowed Document Type from a Cluster Gateway	428
■ Pausing and Resuming a Cluster Gateway	429
■ Preventing Connection Timeouts between Clusters	430
■ Configuring the Forwarding Mode	431
■ Refusing Document Type Updates from a Remote Cluster Gateway	432

Overview

A *cluster gateway* enables two clusters to exchange documents. Unlike clusters, whose constituent Brokers automatically share their document types and client groups, a cluster gateway requires an administrator to explicitly specify which document types the clusters are permitted to send and receive. The client group information is not shared between two clusters. Only the Brokers within a cluster share the client group information.

Typically, you use a cluster gateway when you want to share documents between two different application domains or administrative domains. For example, in a configuration where one cluster handles documents related to employee records and another cluster handles documents related to IT resources, the IT cluster might need to receive certain documents related to employee start, exit, or transfer events. To enable these documents to travel to the IT cluster, you can link the clusters using a gateway.

Unlike a territory gateway, which allows only one connection between two territories, a cluster gateway allows multiple identical connections between two clusters. These multiple identical connections not only provide redundancy in the system to handle multi-publish scenarios, but they also facilitate support for failover mechanisms if a Broker or a cluster gateway connection in the gateway clusters fails.

To understand the concept of cluster gateways, suppose cluster C1 contains three Brokers: Broker A, Broker B, and Broker C, and cluster C2 has two Brokers: Broker X and Broker Y. The cluster gateway between cluster C1 and cluster C2 can have the following identical gateway connections:

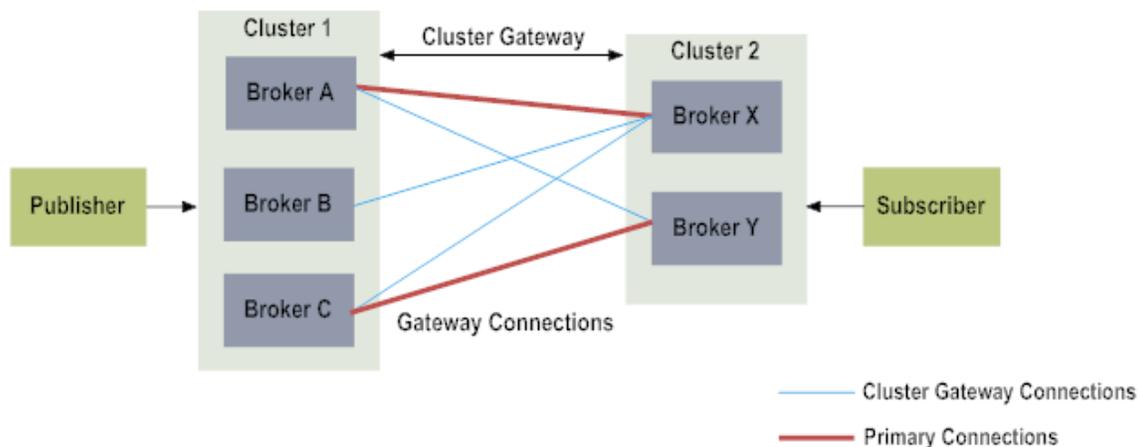
- Broker A to Broker X
- Broker A to Broker Y
- Broker B to Broker X
- Broker B to Broker Y
- Broker C to Broker X
- Broker C to Broker Y

You designate one of the gateway connections from a local Broker (in the local cluster) to a remote Broker (in the remote cluster) as the *primary connection*. The primary connection is the default gateway connection that the local Broker (in the local cluster) uses for forwarding the messages to a remote cluster. If the primary connection fails, the next gateway connection in that local Broker takes over the task of forwarding the messages to the remote cluster. For example, in the above scenario, you designate "Broker A to Broker X" gateway connection as the primary connection. If the "Broker A to Broker X" connection fails, the "Broker A to Broker Y" connection takes over the task of transporting messages from Broker A across the gateway between cluster C1 and cluster C2.

You can also use a cluster gateway connection to link identical clusters that are geographically separated. For example, if you have a cluster that handles documents related to manufacturing operations in North America and you expand the application to operations in Asia, you might decide to create an identical cluster for the Asian region and then link the clusters using a gateway connection. In this case, you are using a cluster gateway to conserve wide-area network bandwidth by explicitly restricting what document types can flow across the network link.

Important: When you use My webMethods to set or change permissions such as "Keep Alive Interval" and "Shared Document Types" on one gateway connection, the changes are applied to all the gateway connections. When you use Broker administration APIs to create and manage cluster gateways, make sure that all the cluster gateways have an identical set of permissions. Otherwise, the cluster gateway might behave inconsistently.

Cluster Gateway Connections, Cluster Gateways, Cluster Gateway Pairs, and Cluster Gateway Brokers



The term *cluster gateway connection* refers to the link from a local Broker (in the local cluster) to a remote Broker (in the remote cluster) that enables two clusters to exchange documents. A *cluster gateway* connection refers to an uni-directional link from one cluster to another. To create this link, two Brokers, one from each cluster, are designated to communicate with one another.

A *cluster gateway* consists of multiple cluster gateway connections that allow message flow from the local cluster to a remote cluster. To form a cluster gateway, you must create and configure a cluster gateway on each of these Brokers.

Note: The Broker that hosts a cluster gateway is a regular member of its own cluster and can handle regular clients in addition to the cluster gateway.

A *cluster gateway pair* refers to a pair of cluster gateways that allow bi-directional message exchange between two clusters. A cluster gateway will not function properly until all the Brokers have at least one outgoing cluster gateway connection.

A *cluster gateway* Broker performs functions similar to those performed by a Remote Broker object in a cluster. In this capacity, the local cluster gateway Broker:

- Establishes and maintains a connection with its counterpart in the remote cluster
- Maintains the subscription list and forward *queue* for the remote cluster
- Retrieves documents for the local cluster from the remote cluster gateway

Unlike a Remote Broker object, a cluster gateway Broker requires an administrator to explicitly specify which document types it can "forward to" and "receive from" the remote cluster. The cluster gateway Broker enforces these permissions on the documents that pass through it.

Note: You cannot create a gateway between a Broker cluster and a Broker territory.

Cluster Gateway Names

In the Broker user interface, a cluster gateway is identified by a name. The name of the cluster gateway is composed of the names of the two clusters that the cluster gateway connects:

localCluster > *remoteCluster*

where *localCluster* is the name of the cluster where the cluster gateway resides and *remoteCluster* is the name of the cluster to which the cluster gateway connects.

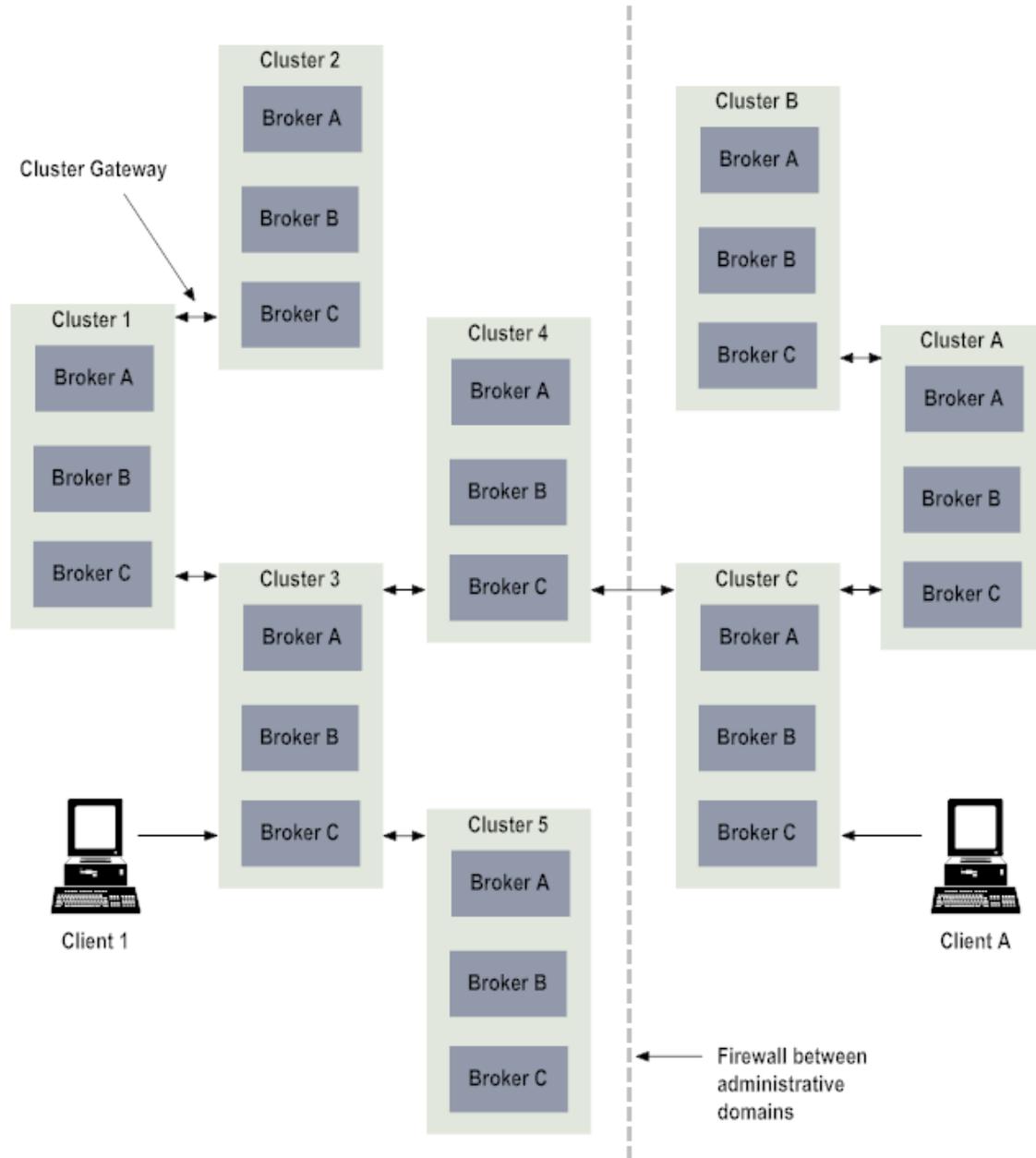
The cluster gateway pair shown in the diagram above consists of a cluster gateway named "Cluster1 \> Cluster2" and cluster gateway named "Cluster2 \> Cluster1."

Connecting Multiple Clusters with Cluster Gateways

You can connect a cluster to multiple clusters to build a network of multiple clusters. For example, you can have a cluster gateway between Cluster1 and Cluster2, as well as additional gateways from Cluster1 to other clusters. For example, if you want to connect Cluster1 to Cluster2 and Cluster3, you create and configure two cluster gateways on Cluster1: cluster gateway Cluster1>Cluster2 and cluster gateway Cluster1>Cluster3.

If you use cluster gateways to link multiple clusters, be aware that the set of connected clusters must form a graph that has no cycles (that is, a path that traverses the graph should not be able to return to its beginning). Visually, an acceptable graph looks like a tree.

The following figure shows a graph that crosses the boundary between two administrative domains. With the correct permissions set at each cluster gateway, client 1 and client A can exchange documents with each other.



Creating a Cluster Gateway

To link clusters with a cluster gateway, you must perform the following high-level steps:

1. **Identify the list of document types that each cluster needs to receive from the other cluster** and make sure that these document types exist in both clusters. If a required document type does not exist in one of the clusters, the administrator of that cluster must add it.

2. **Identify the Broker pairs that will create cluster gateway connections.** Make sure that each Broker in the local cluster is connected to at least one Broker in the remote cluster. The domain name service (DNS) used by the two host machines must be capable of bi-directional name resolution, meaning that the DNS must be able to resolve the name of the host machine to the correct IP address and also be able to resolve the machine's IP address back to the correct name.

When a host machine uses the Dynamic Host Configuration Protocol (DHCP) to dynamically obtain its IP address, many DNS servers will not return the correct name for a given IP address. In this case, you must contact your IT department or your network administrator and ask that a static IP address be assigned to the host machine.

3. **Determine what type of authentication and encryption is required between the clusters** and make sure that the Broker Servers that host the two cluster gateways support these requirements.

For example, if you want to secure each cluster gateway with an access control list (ACL), all Broker Servers must be SSL-enabled (that is, have an SSL identity) and their keystores must be equipped with the trusted root and signed digital certificate for the Broker Servers that hosts their counterpart in the cluster gateway pair. For more information about SSL-enabling a Broker Server and assigning an ACL to a cluster gateway, see ["Cluster Gateway ACLs" on page 332](#).

4. **Create and configure each gateway in the cluster gateway pair.** To configure a cluster gateway, you do the following:
 - a. Specify the local cluster, the remote cluster, and the remote Brokers that form the remote cluster.
 - b. Assign "Allow Receive" permission to each document type that the local cluster will receive.
 - c. Assign "Allow Forward" permission to each document type that the remote cluster will receive. (If necessary, you can include a filter that permits only specified instances of the document type to pass to the remote cluster.)

If you control Broker Servers on both clusters of the gateway pair, you can create and configure both sides of the gateway at the same time as described in ["Creating a Cluster Gateway when You Control Both Sides" on page 412](#). However, if you control Broker Servers on one of the clusters of the pair, you must create and configure your cluster gateway as described in ["Creating a Cluster Gateway when You Control Only One Side" on page 417](#), and the administrator of the Broker Servers on the other side of the pair must create and configure its counterpart in the other cluster.

Creating a "One-Way" Cluster Gateway

Organizations might often want to create what is known informally as a "one-way" cluster gateway, meaning that they might want documents to flow only in one direction. For example, a cluster that handles employee-related documents might want to

broadcast certain documents to, but never receive documents from, a cluster that handles customer support documents.

To create this type of cluster gateway, you must create a complete cluster gateway (that is, you must create and configure *both* cluster gateways in the cluster gateway pair). Creating a one-way cluster gateway *does not* mean that you create only one side of a cluster gateway connection. To achieve the "one-way" nature of this type of connection, you must configure the permission settings on each cluster gateway to restrict the flow of documents to one direction.

For example, if you want to limit the flow of documents from the employee cluster (EMP) to the customer support cluster (CUST), you would set permissions as follows:

On cluster gateway **EMP>CUST**:

Assign this permission...	To...
Allow Forward	All document types that are permitted to travel to the CUST.
Allow Receive	None.

On cluster gateway **CUST>EMP**:

Assign this permission...	To...
Allow Forward	None.
Allow Receive	All document types that the EMP cluster will forward to this cluster (CUST), with restrictive filters, if necessary.

Configuring and Controlling Cluster Gateway Behavior

You can use My webMethods to configure and control the following cluster gateway behavior:

- **Setting a Primary Connection for the Cluster Gateway Broker.** You set the "primary connection" to designate a default gateway connection that the local Broker and remote Broker must use to route messages between themselves across the gateway. For more information about setting the primary gateway connection, see [“Editing Cluster Gateway Details”](#) on page 425.
- **Pausing and Resuming a Cluster Gateway.** The pause and resume controls enable you to stop the flow of outbound traffic from a cluster gateway to the remote cluster. For more information about using these controls, see [“Pausing and Resuming a Cluster Gateway”](#) on page 429.

- **Configuring the Keep-Alive Setting.** The keep-alive setting instructs the local gateway Brokers to issue periodic "keep-alive" messages to the remote gateway Brokers. You use this option if the cluster gateway connects through a firewall that automatically drops connections that are idle for an extended period. For more information about using the keep-alive feature, see [“Preventing Connection Timeouts between Clusters” on page 430](#).
- **Configuring the Forwarding Mode.** The forwarding mode determines how a cluster gateway maintains the subscription list for the remote cluster. For more information about specifying the forwarding mode, see [“Configuring the Forwarding Mode” on page 431](#).
- **Blocking Document Type Updates.** By default, any change made to a document type permitted for exchange between two clusters will be updated synchronously in both the clusters. You can use the "refuse update" option to suppress this behavior on one or both cluster gateways. For information about blocking updates, see [“Refusing Document Type Updates from a Remote Cluster Gateway” on page 432](#).

Creating a Cluster Gateway when You Control Both Sides

It is easiest to configure a cluster gateway if you control both sides of the gateway. In this case, My webMethods can do much of the work for you. You only have to create one side of the cluster gateway. My webMethods automatically creates and configures its counterpart in the remote cluster.

There are two major steps in creating a cluster gateway pair using this procedure.

1. First, you must create the cluster gateway pair as described in [“Defining the Cluster Gateway Pair” on page 412](#).
2. Next, you must configure the list of document types that the clusters are permitted to exchange as described in [“Specifying which Documents the Cluster Gateway Can Forward and Receive” on page 418](#).

Before you begin this procedure, verify that you have administrative access to all Broker Servers in the pair of clusters that will host the cluster gateway. These Broker Servers must appear in your Broker Server List and you must have authority to administer all of them if they are protected by ACL.

If you have access to Broker Servers in only one of the pair of clusters, you must configure the cluster gateway by using the procedures in [“Creating a Cluster Gateway when You Control Only One Side” on page 417](#). An administrator with access to the Broker Server that will host the other cluster gateway must use the same procedure to create your cluster gateway's counterpart in the remote cluster.

Defining the Cluster Gateway Pair

Do the following to create a cluster gateway pair on all Brokers that you control:

To define a cluster gateway pair

1. In My webMethods: **Messaging > Broker Clusters > Cluster Gateways**.
2. Click **Add**.
3. On the **Create Gateway Pair** tab, select the local cluster and remote cluster that will form the cluster gateway connection.
4. Click **Edit Selection** button to add or remove Broker connections.
5. Click **Apply**.
6. Select the primary connection for each local Broker gateway.
7. Click **Create**.
8. Proceed to the following section to specify which document types each cluster gateway is permitted to send and receive. Otherwise, the cluster gateway connection will not function.

Exchanging Document Types across a Cluster Gateway

After you define the cluster gateway pair, you must specify which document types can flow from one cluster to another. To enable a document to flow between clusters, one cluster gateway must allow that document type to be *forwarded* and the other cluster gateway must allow that type to be *received*.

You can define which document types can travel through a cluster gateway in two ways:

- You can manually select the individual document types that the cluster gateway can forward or receive as described in [“Specifying which Documents the Cluster Gateway Can Forward and Receive” on page 418](#).
- You can select one or more client groups that identify the document types that the cluster gateway Broker can forward or receive as described in [“About Using Client Groups to Specify Permissions on Document Types” on page 415](#).

With either procedure, you can optionally instruct My webMethods to automatically make the corresponding permission settings on its counterpart in the remote cluster. If you do not use this option, you must configure the permissions on the remote cluster gateway manually.

Specifying Individual Document Types a Cluster Gateway Can Receive

Use the following procedure to set permissions on individual document types for a cluster gateway.

To select individual document types a cluster gateway can receive from other clusters

1. In My webMethods: **Messaging > Broker Clusters > Cluster Gateways**.
2. In the **Cluster Gateway List**, locate the cluster gateway that you want to configure and click  **Details**.
3. Click the **Shared Document Types** tab.
4. To select the document types that this cluster can receive from the other cluster, do the following:
 - a. Click **Allow Receive** and then click **Add Receive Document Types**.
 - b. On the **Allow Receive Document Types** panel, select the document types that this cluster gateway will receive from the other cluster.

Note: If you do not see the document type that you need, it means the document type does not exist in one or both clusters.

- c. Click **Apply**.
5. Select one of the following options and click **Update**.

Enable this option...	If....
Yes. Set Permissions on Other Side of the Cluster Gateway.	<p>You want My webMethods to automatically set the corresponding "forward" permissions for the selected document types on the remote cluster gateway.</p> <p>Do not use this option unless you control both cluster gateways.</p>
No	You want to set these permissions only on the local cluster gateway. If you select this option, you must manually set "forward" permissions for these document types on the remote cluster gateway.

6. If this cluster gateway Broker will forward documents to the other cluster, use the following procedure to specify which document types it will send.

Specifying Individual Document Types a Cluster Gateway Can Forward

Use the following procedure to set permissions on individual document types for a cluster gateway.

To select individual document types that a cluster gateway can forward to the other cluster

1. In My webMethods: **Messaging > Broker Clusters > Cluster Gateways**.

2. In the **Cluster Gateways List**, locate the cluster gateway that you want to configure and click  **Details**.
3. Click the **Shared Document Types** tab.
4. To select the document types that this cluster can forward to the other cluster across the gateway, do the following:
 - a. Click **Allow Forward** and then click **Add Forward Document Types**.
 - b. On the **Allow Forward Document Types** panel, select the document types that this cluster gateway will be allowed to forward to the other cluster.
 - c. Click **Add**.
5. Select one of the following options and click **Update**.

Enable this option...	If...
Yes. Set Permissions on Other Side of the Cluster Gateway	<p>You want My webMethods to automatically set the corresponding "receive" permissions for the selected document types on the remote cluster gateway.</p> <p>Do not use this option unless you control both cluster gateways.</p>
No	<p>You want to set permissions only on the local cluster gateway. If you select this option, you must manually set "receive" permissions for these document types on the remote cluster gateway.</p>

6. If you want to filter the documents that this cluster will forward to the other cluster across the gateway, click  **Edit Filter** for the document type and define the filter as described in [“Filtering Documents” on page 419](#).

About Using Client Groups to Specify Permissions on Document Types

If you have one or more client groups that identify the set of documents that a cluster gateway can exchange with the remote cluster, you can use those groups to specify which documents are permitted to flow to and from which cluster gateway.

When you use a client group to specify the document type permissions for a cluster gateway, the can-publish and can-subscribe permissions in the client group are mapped to the cluster gateway permissions as follows:

This permission for a document type in the client group...	Sets this permission for the document type on the cluster gateway...
Can Publish	Allow Forward (Permits the cluster gateway to forward instances of the document type to subscribers in the remote cluster.)
Can Subscribe	Allow Receive (Permits the cluster gateway to receive instances of the document type from publishers in the remote cluster.)
Important: If you want to set permissions on both cluster gateways, be sure that all of the document types in the selected client groups exist in both clusters.	

Using Client Groups to Specify Permissions on Document Types

Use the following procedure to use client groups to specify which document types can flow to and from a cluster gateway.

To use client groups to specify which document types can flow to and from a cluster gateway

1. In My webMethods: **Messaging > Broker Clusters > Cluster Gateways**.
2. In the **Cluster Gateways List**, locate the cluster gateway that you want to configure and click  **Details**.
3. Click the **Shared Document Types** tab and then click **Hook Up Client Groups**.
4. Use the **Available Client Group** controls to select the client groups that contain the document type definitions and permissions that you want to apply to this cluster gateway Broker.
5. Select one of the following options and click **Hook Up**.

Enable this option...	If....
Yes. Enable Permissions on Other Side of the Cluster Gateway	You want My webMethods to automatically set the corresponding permissions for the specified document types on the remote cluster gateway. (Do not use this option unless you control both cluster gateways.)
No	You want to set permissions only on the local cluster gateway. If you select this option, you must manually set the corresponding permissions for these document types on the remote cluster gateway.

Creating a Cluster Gateway when You Control Only One Side

When you configure a cluster gateway that crosses administrative domains (such as between two companies), you might not control Brokers in both the clusters that make up the gateway pair. In this case, you must create and configure the gateway in your cluster, and the administrator of the Brokers in the other cluster must create and configure the gateway in the remote cluster.

To do this, you and the administrator of the other cluster must each perform the following steps:

1. Create the cluster gateway for your cluster as described in [“Defining One Side of a Cluster Gateway Pair” on page 417](#). You will need the following information from the other administrator.
 - The *exact* name of the remote cluster
 - The names of Brokers in his/her cluster (remote cluster)
 - The names of the Broker Servers on which the Brokers reside
2. Configure the list of document types that your cluster gateway can exchange with the remote cluster as described in [“Specifying which Documents the Cluster Gateway Can Forward and Receive” on page 418](#).

Defining One Side of a Cluster Gateway Pair

Use the following procedure to create one side of a cluster gateway.

To define one side of the cluster gateway pair

1. In My webMethods: **Messaging > Broker Clusters > Cluster Gateways**.
2. Click **Add**.
3. Click the **Create Gateway** tab, and complete the following fields.

In this field...	Do the following...
Local Cluster	Select the Broker on which you want the cluster gateway created.
Remote Cluster	Specify the name of the other cluster of the gateway pair with which the local cluster will exchange documents through the cluster gateway. Type the name exactly as it is defined for that cluster. This value is case sensitive. Example: OrdersCluster

In this field...	Do the following...
RemoteBrokers	Click Add to specify the names of all Brokers in the remote cluster. Example: OrdersbROKER1, ORDERSBROKER2.

4. Click **Edit Selection** to select Brokers in the remote cluster that will create gateway connections through the cluster gateway.
5. Click **Apply**.
6. Select the primary connection.
7. Click **Create**.

Specifying which Documents the Cluster Gateway Can Forward and Receive

If you do not have administrative access to the remote cluster, you and the administrator of the remote cluster must use the procedures described in [“Exchanging Document Types across a Cluster Gateway” on page 413](#) to manually specify the permissions for the document types that the clusters can exchange.

To ensure that documents are sent across the cluster gateway connection, you must set the correct permissions on each cluster gateway connection. The document types that your cluster will receive from the remote cluster must be listed in your cluster gateway's "Allow Receive" list. Additionally, these document types must be listed in the remote cluster gateway's "Allow Forward" list. You must also add to your cluster gateway's "Allow Forward" list each document type that is listed in the remote cluster gateway's "Allow Receive" list. If a document type is not listed in the relevant lists on both cluster gateways, the document type cannot pass through the cluster gateway connection.

Filtering Documents across a Cluster Gateway

A filter string specifies criteria for the contents of a document. Broker uses the filter string to determine which documents match your requirements and allows only those documents to pass through the cluster gateway.

You apply filter strings to those document types that a cluster gateway *can receive* from a remote cluster.

To set a filter on a document type

1. In My webMethods: **Messaging > BrokerClusters > Cluster Gateways**.
2. In the **Cluster Gateway List**, locate the cluster gateway that you want to configure and click  **Details**.

3. Click the **Shared Document Types** tab.
4. In the list of the document types that the cluster gateway is permitted to forward, click  **Edit Filter** for the document type that you want to filter.
5. In the **Filter** text box, type a filter string that specifies the criteria that identifies which documents this gateway can forward to the remote cluster. For information about specifying a filter string, see [“Filtering Documents” on page 419](#).
6. Click **Apply**.

The filter appears in the **Filter** column in the **Allow Forward** table.

Filtering Documents

Suppose that a document contains a person's age and name of the state he lives in. The first document field has the label `age` and the second document field has the label `state`. The following filter string matches only those documents whose `age` field value is greater than 65 and whose `state` field value is equal to FL.

```
age > 65 and state = "FL"
```

In this example filter string, `age` and `state` represent document fields. This filter also contains an arithmetic constant, `65`, and a string constant, `"FL"`. The boolean operator `and` combines the field criterion for `age` and `state`.

Some additional examples of filter specifications are as follows:

```
debt > salary*0.50
>packaging = "portable" and price > 5000
>answer = 'Y' or answer = 'y'
>(answer = 'Y') or (answer = 'y')
```

Filter strings can:

- Compare document field contents against constants and computed values.
- Combine document field comparisons using the boolean operators `and`, `or` and `not`.
- Perform arithmetic operations on document fields and constants.
- Contain regular expressions.
- Contain string and arithmetic constants.
- Contain a hint that specifies how documents should be processed.

For information about using regular expressions, hints, and filter functions from the `webMethods Broker` library, see the *webMethods Broker Client Java API Programmer's Guide*.

Filter Rules

Filter strings must adhere to the following rules:

- Field names can be fully qualified, such as:

```
struct_field.seq_field[2]
```

- A character constant is a single character surrounded by single quotation marks. For example:

```
'A'
```

- A string constant is zero or more characters surrounded by double quotation marks. For example:

```
"account"
```

- If a character or string constant contains a single or double quotation mark, precede the quote with a backslash. For example:

```
'\''
```

- You can use parentheses to control the order of operator precedence.

Filter Operators

The following tables contain the various operators that you can use to create filters. For a more complete list of operators, see the *webMethods Broker Client Java API Programmer's Guide*.

Note: Integration Server and Software AG Designer use different filter syntax for subscribing to publishable documents. For more information, see the *Software AG Designer Online Help*.

The following table shows the logical filter operators:

Note: Logical filter operator expressions are evaluated using a method similar to SQL expression evaluation, in that all operators are always evaluated. When a logical filter operator expression contains multiple operators, operator precedence determines the sequence in which the operations are performed. For example, when evaluating the expression "A OR B", both "A" and "B" are evaluated, even if "A" evaluates to "true".

Operator	Description
! not	Not
&& and	And
or	Or

The following table shows the comparison filter operators:

Operator	Description
<	Less than
<=	Less than or equal to
>	Greater than
>=	Greater than or equal to
=	Equal to
==	Equal to
!=	Not equal to

The following table shows the arithmetic filter operators:

Note: Implicit type conversion occurs when operands in an arithmetic operation have different types. The operands are converted to a larger value before the comparison occurs. Type `char` is considered numeric, but `boolean` is not.

Operator	Description
-	Unary minus
*	Multiplication
/	Division
%	Modulus Division
-	Subtraction
+	Addition

The following table shows the String operators:

Operator	Description
+	Concatenation

Operator	Description
<	Less than
<=	Less than or equal to
>	Greater than
>=	Greater than or equal to
=	Equal to
==	Equal to
!=	Not equal to

Viewing Cluster Gateway Details

The Cluster Gateways page displays a list of cluster gateways known to the Broker Servers in your Broker Server List, and provides general information about them. View detailed information about the cluster gateway on the Cluster Gateway Details page.

To view information about a cluster gateway

1. In My webMethods: **Messaging > Broker Clusters > Cluster Gateways**.

The **Cluster Gateways List** displays the following information about each cluster gateway you create.

Column	Description
Local Cluster > Remote Cluster	Name of the cluster gateway.
Number of Connections	Displays the number of gateway connections you configured in your cluster gateway.
Status	<ul style="list-style-type: none"> ■ Active. The cluster gateway is configured, and is operational (that is, not paused). <p>Active status indicates that the cluster gateway is logically available, but does not necessarily mean that the two clusters in the gateway pair are physically connected at this point in time. To determine whether the two clusters are physically connected, go to step</p>

Column	Description
	<p>2 and examine the Brokers Linked field on the Cluster Gateway Details page.</p> <ul style="list-style-type: none"> ■ Error. The cluster gateway is not available for use. This state usually indicates a problem with the cluster gateway configuration. ■ Paused. This cluster gateway is paused and is not currently forwarding any documents between the clusters. For information about pausing and unpausing a cluster gateway, see “Pausing and Resuming a Cluster Gateway” on page 429.

2. Click  **Details** to display the **Cluster Gateway Information** page. The **Configuration** tab on this page displays the following information:

Field	Description
Brokers Linked	<p>Indicates whether each Broker in the local cluster is connected to at least one Broker in the remote cluster and vice-versa.</p> <ul style="list-style-type: none"> ■ Yes. Brokers are connected across the two gateway clusters, and able to exchange documents. ■ No. Each Broker in the local cluster is not connected to at least one Broker in the remote cluster and vice-versa. There is no exchange of documents through the cluster gateway.
Status	<p>Indicates the state of the cluster gateway connection. See Status description in the preceding step.</p>
Keep Alive Interval	<p>Specifies how often (in seconds) this cluster gateway must issue "keep-alive" events to the remote cluster gateway. A cluster gateway can use keep-alive events to prevent a firewall from disconnecting what it considers an idle connection. For more information about using keep-alive events, see “Preventing Connection Timeouts between Clusters” on page 430.</p> <p>If a keep-alive interval of 0 is specified, this field displays Disabled.</p>
Queue Length	<p>Indicates the number of documents currently in the forward queue of each gateway connection, waiting to</p>

Field	Description
	be retrieved by the remote cluster through the cluster gateway.
Shared Document Types	Indicates the number of document types that are shared and can, therefore, be exchanged through the cluster gateway. Click this value to view the shared document types.
Authentication Type	<p>Indicates whether this cluster gateway is SSL-enabled.</p> <ul style="list-style-type: none"> ■ None. The cluster gateway is not SSL-enabled. The cluster on which it is hosted does not have an SSL identity and, therefore, the cluster gateway cannot interact with the remote cluster by using SSL. It will connect to the remote cluster by using the non-SSL port. ■ SSL. The cluster gateway is SSL-enabled. The cluster on which it is hosted has an SSL identity, and can interact with the remote cluster using SSL.
Access Control	<p>Indicates whether this cluster gateway is secured with an access control list (ACL).</p> <ul style="list-style-type: none"> ■ ■ An access control list is assigned to this cluster gateway. For more information, see “Cluster Gateway ACLs” on page 332. ■ ▲ An ACL is not assigned to this cluster gateway. To create one, you must have a basic authentication identity or an SSL identity that belongs to the "admin" client group for the cluster that hosts this cluster gateway.
Static Gateway Forwarding	Specifies whether this cluster gateway maintains a static or dynamic subscription list for the remote cluster. For more information about selecting a forwarding mode, see “Configuring the Forwarding Mode” on page 431 .
Encryption	Specifies whether this cluster gateway will encrypt its interactions with the remote cluster gateway.
Refuse All Shared Document Type Updates	Specifies whether this cluster gateway accepts document-type updates from the remote cluster. For more information about allowing or blocking updates to document type definitions, see “Refusing Document

Field	Description
	Type Updates from a Remote Cluster Gateway” on page 432.

Editing Cluster Gateway Details

You can add cluster gateway connections, remove cluster gateway connections and set the primary gateways.

To edit cluster gateway details

1. In My webMethods: **Messaging > Broker Clusters > Cluster Gateways**.
2. Click  **Details** to display the Cluster Gateway Details page.
3. Click the **Connections** tab.
4. Click **Edit Selection** to add or remove a cluster gateway connection.
5. In the **Select Remote Broker Connections** dialog box, select Brokers in the remote cluster from the **Available** list to create gateway connections with this local Broker.
6. Click **Apply**.
7. In the Cluster Gateway Information page, under **Primary**, select the remote Broker with which you want this local Broker to create a new primary cluster gateway connection.
8. Click **Apply**.

Viewing Document Details across a Cluster Gateway

Use the following procedure to view information about the activity between the two clusters in the gateway pair.

To view information about documents flowing through a cluster gateway

1. In My webMethods: **Messaging > Broker Clusters > Cluster Gateways**.
2. Click  **Details** to display the Cluster Gateway Details page.
The **Queue Length** field on the **Configuration** tab reports the number of documents currently in the forward queue for all gateway connections in the cluster gateway.
3. If you want information about the number of documents that the remote cluster has received from the local cluster gateway, click the **Remote Broker** tab and examine the following statistics:

Field	Description
Recent Receipts	<p>The number of documents that the remote cluster has retrieved from the forward queue on this cluster gateway during the last statistic collection interval.</p> <p>By default, the statistics collection interval represents one minute, however, you can configure this interval in My webMethods. For procedures, see the Time interval between statistical refresh parameter in “Configuring the Connection Parameters” on page 53.</p> <p>Note: Recent Receipts data is available only if you have selected Enable Statistical Polling.</p>
Total Receipts	The number of documents that the remote cluster has retrieved from the forward queue on this cluster gateway since the cluster gateway was created.
Last Receipt	The time at which the remote cluster last retrieved a document from the forward queue on this cluster gateway.

4. In the **Remote Broker** tab, you also see the following information:

Field	Description
Name	Lists the names of the remote Brokers.
Description	Lists the descriptions you gave for the remote Brokers.
Connected	A green signal with the text "Yes" indicates that this remote Broker is connected to a local Broker through a gateway connection.
Cluster	Name of the cluster with which this Broker is associated.

5. If you want information about the number of documents that the local cluster gateway has received from the remote cluster, click the **Local Broker** tab and examine the following statistics:

Field	Description
Recent Receipts	<p>The number of documents that this cluster gateway retrieved from the remote cluster during the last statistic collection interval.</p> <p>By default, the statistics collection interval represents one minute; however, you can configure this interval in My webMethods. For procedures, see the Time interval between statistical refresh parameter in “Configuring the Connection Parameters” on page 53.</p>
Total Receipts	The number of documents that this cluster gateway has retrieved from the remote cluster since the cluster gateway was created.
Last Receipt	The time at which this cluster gateway last retrieved a document from the remote cluster.

6. In the **LocalBroker** tab, you also see the following information:

Field	Description
Name	Lists the names of the local Brokers.
Description	Lists the descriptions you gave for the local Brokers.
Connected	A green signal with the text "Yes" indicates that this local Broker is connected to a remote Broker through a gateway connection.
Cluster	Name of the cluster with which this Broker is associated.

Deleting a Cluster Gateway

To remove a cluster gateway established between two clusters, you delete each member of the cluster gateway pair. If you do not control the remote cluster, you should coordinate the removal of the remote cluster gateway with the administrator of the remote Broker.

If you delete a cluster gateway, all documents that are currently queued for the remote cluster gateway are deleted.

After one side of the cluster gateway is deleted, the other side should be deleted promptly. Otherwise, it will continue to accumulate documents in its forward queue and can eventually consume a considerable amount of memory and queue storage.

Important: It is not easy to restore a cluster gateway after you remove it. To restore the cluster gateway, you must recreate the cluster gateway, and manually reconfigure the "allow forward" and "allow receive" permissions.

To delete a cluster gateway connection

1. In My webMethods: **Messaging > Broker Clusters > Cluster Gateways**.
2. In the **Cluster Gateways List**, select the check box beside the cluster gateways you want to delete.
3. Click **Delete**.

If the other member of a cluster gateway pair is not under your control, the administrator of the remote cluster gateway must perform similar steps to remove the remote cluster gateway.

Removing an Allowed Document Type from a Cluster Gateway

Use the following procedure to remove a document type from the list of document types that a cluster gateway is allowed to forward or receive.

If you control both cluster gateways in the cluster gateway pair, you can remove the document type from both cluster gateways at the same time. If you do not control the remote cluster gateway, you must coordinate with the authorized administrator to delete the document types from the remote cluster gateway.

To remove a document type from a cluster gateway

1. In My webMethods: **Messaging > Broker Clusters > Cluster Gateways**.
2. Click  **Details** to display the Cluster Gateway Details page and then click the **Shared Document Types** tab.
3. If you want to remove a document type from the list of document types that this cluster gateway can receive, click **Allow Receive**.
4. Select the check box beside the document type that you want to remove from the allowed list.
5. Click **Delete**.
6. Select one of the following options and click **Update**.

Enable this option...	If...
Yes. Remove document type(s) on other side of cluster gateway	<p>You want My webMethods to automatically remove the document type from the opposite permission list on the remote cluster gateway.</p> <p>Do not use this option unless you control both cluster gateways.</p>
No	<p>You want to remove the document type only from the local cluster gateway. If you select this option, the document type must be manually removed from the opposite permission list on the remote cluster gateway.</p>

Pausing and Resuming a Cluster Gateway

You can temporarily halt the flow of documents from a cluster gateway to a remote cluster by pausing the cluster gateway. When you pause a cluster gateway, the gateway locks the forward queues that it maintains for the remote cluster. This pauses the flow of documents on all the gateway Brokers. This action prevents the remote cluster from retrieving documents from this cluster gateway.

No documents are lost when you pause a cluster gateway. The forward queue on the cluster gateway continues to accumulate documents for the remote cluster, but these documents will not flow to the other cluster until the cluster gateway is resumed.

Be aware that pausing a cluster gateway halts the flow of *outbound* documents (documents that originated from the cluster on which the cluster gateway resides and are being forwarded to subscribers in the remote cluster). It does not prevent that cluster gateway from receiving documents from the remote cluster. To halt the flow of inbound documents, you must pause the cluster gateway in the remote cluster, too.

Note: Administrators often pause a cluster gateway when the cluster hosting the cluster gateway is undergoing service or significant configuration changes. For example, if you intend to make numerous changes to document types or permissions, you might pause the cluster to ensure that all the changes are propagated to the other cluster at one time.

Pausing a Cluster Gateway

Use the following procedure to pause a cluster gateway.

To pause a cluster gateway

1. In My webMethods: **Messaging > Broker Clusters > Cluster Gateways.**

2. Click  **Details** for the cluster gateway whose outbound flow of documents you want to pause.
3. On the **Configuration** tab, click **Pause Forwarding**.

The **Status** field switches to  **Paused** and indicates the ID of the client that placed the pause on the cluster gateway.

Important: When you pause a cluster gateway on one side, documents will accumulate in the cluster gateway's forward queue on the other side (on the remote-cluster side). If the cluster gateway is paused for a significant amount of time, the number of documents in the queue can increase and begin to consume too much memory or disk space.

Resuming a Cluster Gateway

Use the following procedure to resume a cluster gateway.

To resume a cluster gateway

1. In My webMethods: **Messaging > Broker Territories > Cluster Gateways**.
2. Click  **Details** for the cluster gateway whose outbound flow of documents you want to restore.
3. On the **Configuration** tab, click **Resume Forwarding**.

The **Status** field switches to  **Active**, indicating the outbound documents are permitted to flow to the other cluster.

Preventing Connection Timeouts between Clusters

If your cluster gateway connects to a remote cluster through a firewall, you might need to activate the cluster gateway's "keep-alive" feature to prevent the firewall from dropping the connection after a period of inactivity. When you enable the keep-alive feature, the cluster gateway publishes "keep alive events" to the remote cluster at a specified frequency. The keep-alive events prevent the connection from appearing idle to the firewall.

Important: Do not set a keep-alive interval that is too short because doing so might create unnecessary traffic across the network. Set a value that is just adequate to prevent the connection from timing out.

If you set a value for the Keep Alive Interval on the local cluster, the gateway will send keep-alive messages to the remote cluster. To allow keep-alive messages to flow in both directions, you must set a value for the Keep Alive Interval on the remote cluster as well.

To implement the keep-alive feature, you must know what exactly causes the firewall to drop a connection. Ask the network administrators at each end of the cluster gateway

to provide behavioral information about the firewalls. With this information, you can determine which cluster gateway must send keep-alive messages and at what frequency so that the connection is intact.

To view or modify the keep-alive setting

1. In My webMethods: **Messaging > Broker Clusters > Cluster Gateways**.
2. Click  **Details** for the cluster gateway whose keep-alive setting you want to configure.
3. On the **Configuration** tab, examine the **Keep Alive Interval** field. It will display either the amount of time between keep-alive events, in seconds, or the word "Disabled" if the keep-alive feature is not activated.
4. If you want to modify or disable the keep-alive setting, do the following:
 - a. Click the current **Keep Alive Interval** value to display the Change Cluster Gateway Keep Alive Interval page.
 - b. In the **Keep Alive Interval (seconds)** field, specify the frequency at which you want the cluster gateway to publish keep-alive messages. Specify zero seconds if you want to disable the keep-alive feature.
 - c. Click **Apply**.

Configuring the Forwarding Mode

A cluster gateway maintains a subscription list that identifies the types of documents the remote cluster wants to receive. This subscription list tells the cluster gateway which documents it must forward to the remote cluster. The *forwarding mode* determines how the cluster gateway populates and maintains this subscription list.

- In *dynamic forwarding mode*, a cluster gateway maintains the subscription list for a remote cluster based on subscription events that the remote cluster gateway issues when clients in its cluster add and drop subscriptions to a document type that the clusters share. Broker Servers version 6.5 and earlier use only dynamic forwarding mode.
- In *static forwarding mode*, a cluster gateway automatically populates the subscription list based on the document types that the cluster gateway is permitted to forward to the remote cluster. Static forwarding mode is the default mode in Broker Servers version 6.5 and later.

In static mode, it is important to configure the allow-forward permissions carefully. If the list includes document types for which there are no actual subscribers in the remote cluster, those documents will be forwarded to the remote cluster unnecessarily, wasting processing time, queue space, and network bandwidth. For information about configuring the allow-forward permissions, see [“Exchanging Document Types across a Cluster Gateway” on page 413](#).

Note: If you are running Broker Server that supports both modes, we recommend that you use static forwarding mode (the default). This mode is less prone to subscription propagation failures related to network outages and timing factors.

To view or configure the forwarding mode

1. In My webMethods: **Messaging > Broker Clusters > Cluster Gateways**.
2. Click  **Details** for the cluster gateway whose forwarding option you want to view or configure.
3. On the **Configuration** tab, select or clear the **Static Gateway Forwarding** check box.

If the cluster gateway is running on a Broker Server that does not support static cluster gateway forwarding, you will not be able to enable this option.

Refusing Document Type Updates from a Remote Cluster Gateway

Clusters that are connected by a gateway automatically exchange and apply updates to the document types that they are configured to exchange. For example, if cluster A is configured to receive a document type "CancelOrder" from cluster B, and an administrator in either cluster modifies that document type, the change is automatically propagated to the other cluster.

If you want your cluster gateway to block document type changes from the remote cluster, you can enable the **Refuse All Shared Document Type Updates** option. When this option is enabled and the remote cluster publishes a change to a document type, the following occurs:

- The document type change is rejected by the cluster gateway and the document type is marked "out of sync."
- An alert is written to the log file associated with the Broker Servers in the clusters on which the cluster gateway is running.
- The cluster gateway halts inbound traffic (that is, it stops retrieving all documents from the remote cluster).

When a cluster gateway enters this condition, it will not resume accepting documents until one of the following occurs:

- The cluster gateway's document type is updated to match the document type on the remote cluster.
- The remote cluster modifies its document type to match the document type on this cluster gateway.

You can block document type updates globally or individually.

Blocking Document Types Globally

Use the following procedure to configure a cluster gateway to globally refuse document type update.

To view or configure a cluster gateway to globally refuse document type update

1. In My webMethods: **Messaging > Broker Clusters > Cluster Gateways**.
2. Click  **Details** for the cluster gateway whose update settings you want to view or configure.
3. On the **Configuration** tab, enable or disable the **Refuse all shared document type updates**.

Note: Disabling this option does not change any update settings that are applied to specific document types.

4. Click **Apply**.

Blocking Document Types Individually

Use the following procedure to configure a cluster gateway to refuse an individual document type update.

To view or configure a cluster gateway to refuse individual document type update

1. In My webMethods: **Messaging > Broker Clusters > Cluster Gateways**.
2. Click  **Details** for the cluster gateway whose document types you want to view or configure.
3. Click the **Shared Document Types** tab.
4. Click the **Allow Forward** tab.
5. Under **Document Types**, select the document types for which you want to configure the Refuse Update option, and then click the Refuse Update button.
6. To determine whether the cluster gateway currently accepts or rejects updates for the document type, examine the value (**Yes** or **No**) in the **Refuse Updates** column.
7. If you want to change the document type's **Refuse Update** setting, do the following:
 - a. Select the check box beside the document type whose update option you want to change.
 - b. Click **Refuse Update** or **Accept Update**, depending on whether you want the cluster gateway to reject or accept updates made to the selected document type.

18 Using My webMethods with JMS

■ Overview	436
■ JMS and Traditional Broker Applications	436
■ webMethods Broker as a JMS Provider	437
■ Binding Administered Objects in JNDI	440
■ Using Basic Authentication with JMS	441
■ Using SSL with JMS	441
■ Managing JMS Clients: A Road Map	442
■ Managing JNDI Naming Directories	443
■ Managing Connection Factories	449
■ Managing Cluster Connection Factories	455
■ Managing Composite Cluster Connection Factories	466
■ Managing Destinations	470
■ Assigning Publish and Subscribe Permissions	477
■ Creating a Durable Subscription	478

Overview

This chapter describes how to work with JMS using My webMethods. It explains the relationships between JMS and Broker objects and how to manage JMS objects administratively on the Broker. It also covers how to create durable subscribers and use SSL with JMS.

You can also manage JMS using the webMethods Broker JMSAdmin command-line utility or programmatically. For more information, see the *webMethods Broker Messaging Programmer's Guide*.

JMS and Traditional Broker Applications

webMethods Broker works with both "traditional" (that is, non-JMS) applications built with the Broker API and JMS applications.

- **Clients** built using the Broker API exchange data by publishing and subscribing to document types. Document types are created with Software AG Designer running under webMethods Integration Server.

The Broker API is a rich messaging API that allows a high degree of customizing. Compared to JMS messages, Broker documents have available a greater number of acknowledgment modes. Broker documents can be validated, and you can filter on any field (in a JMS message, you can only filter header and property fields). Generally, you want to use traditional Broker applications when your messaging environment has requirements such as the ones listed above that are not available under JMS.

See [“Managing Clients”](#) for information about working with traditional Broker applications.

- **JMS applications**, which can be standalone applications, webMethods Integration Server (IS) clients, or Java EE applications, exchange data with Broker using JMS message producers and consumers. All of these applications are built using the JMS API.

Working with JMS allows you to implement a standards-based mechanism for your messaging environment. In most cases, JMS programs store their administered objects in a JNDI namespace, a vendor-neutral naming and directory API. Generally, you want to use JMS for messaging when using a standards-based messaging application or when portability is a primary consideration.

webMethods Broker as a JMS Provider

webMethods Broker technology supplies the underlying routing and distribution facility for transporting messages. When webMethods Broker is used as a JMS provider, JMS clients connect to each other through one or more Brokers.

This section describes the webMethods support of the JMS protocol. The topics explain, at a high level, the principal JMS abstractions and facilities and the underlying Broker architecture components that support them.

JMS Messages

A JMS message is a Broker document; its document type is related to that of the topic or queue being published.

For information about the mapping of fields in a JMS message to a Broker document, see the *webMethods Broker Messaging Programmer's Guide*.

JMS Messaging Model

The webMethods Broker supports the JMS *publish-subscribe (pub-sub)* and *point-to-point (PTP)* messaging models.

Pub-Sub Messaging

In pub-sub messaging, message producers publish messages, and message subscribers consume those messages. If multiple message consumers subscribe to the same topic, each consumer will be given a copy of the message.

Broker's pub-sub messaging mechanism supports the JMS pub-sub messaging model. By default, a JMS topic is represented by a Broker document type of the same name, and the association between the two is through topic and document-type name matching. Message consumers subscribing to topics, known in JMS as durable subscribers, are implemented by Broker clients subscribing to document types.

Point-to-Point Messaging

In JMS PTP messaging, a message producer sends messages to a message queue, and a message consumer retrieves messages from that queue. A message queue is implemented as a guaranteed Broker client with the same name.

To accommodate shared-state clients, Broker creates Broker client sessions for message consumers connecting to a message queue.

The message queue may also involve multiple message producers that can send messages to a message queue, and multiple message consumers that can retrieve messages from the same queue.

Broker's document delivery mode supports the JMS PTP messaging model. With queue destinations, webMethods Broker, as the JMS provider, maps the message to a Broker document with a document type name matching that of the JMS message queue (`JMS::Queues::JMSQueueName`).

JMS Connections

The Broker implementation of a `ConnectionFactory` object captures the information required to create a connection or reconnect to a Broker on the target Broker Server. The JMS connection object is a collection of property settings defining the parameters of this connection. The implementation holds the network socket connection with the connection object created by the JMS client program.

For each JMS connection object, there is one network socket connection established with the target Broker Server. Message consumers and producers created using the same JMS connection factory share the same socket connection. A Broker client is created on the Broker Server to hold this socket connection. Message producers created from the same connection object use this same "base" Broker client to send or publish messages.

All clients created for subscribers on the same connection belong to the same client group, which is defined in the `ConnectionFactory` object for the connection. It is also the client group for the base Broker client. Note that you cannot override this client group setting.

Session

A JMS `session` object represents a separate execution thread within the JMS client application. The creation of a JMS session is a purely client-side operation that causes no actions to take place on the target Broker Server. A JMS session does not correspond to a Broker client session.

Message Consumer

Message consumers belonging to the same JMS connection share the same network socket connection to the Broker (held by the Broker client). Each message consumer has a corresponding Broker client on the Broker Server. Message consumers that connect to a shared-state Broker client are client sessions on that Broker client.

- If the message consumer's destination is a topic, a corresponding Broker client will be created on the Broker Server for that message consumer. That Broker client will subscribe to the Broker document type that has the same name as the JMS topic. If the message consumer defines a message selector, that message selector is translated to a Broker message filter and registered with the Broker client.

If the message consumer is a durable subscriber, the base Broker client must have a `clientID` (see [“JMS Connections” on page 438](#) and [“Durable Subscribers” on page 439](#)).

- If the message consumer's destination is a queue, the message consumer will be connected to a guaranteed Broker client (representing the JMS queue) on the Broker Server. If the queue defines a message selector, the message selector is translated to a Broker message filter and handed to the Broker client for evaluation when retrieving messages from the Broker client queue.

If the JMS message queue is defined as shared-state (represented by a shared-state Broker client), then multiple queue receivers can connect to it to retrieve messages. Each receiver will appear as a client session connecting to the shared-state Broker client, and each message is given only to one receiver.

Durable Subscribers

The JMS pub-sub model supports durable subscribers. A durable subscriber's subscription remains valid until explicitly removed by the client program; thus, a durable subscriber survives both client connection failures and server restarts.

A durable subscriber provides a means of guaranteeing document delivery. If the subscribing application is not running at the time a message is sent, or there is a disconnection between the JMS application and the JMS provider (or, in a Java EE application, between the MDB's container connection and the JMS provider), the messages that would have been received are held by the JMS provider in non-volatile storage. When the application reconnects, the messages are redelivered from the provider.

A durable subscriber is implemented as a guaranteed Broker client that subscribes to the corresponding document type. To create durable subscribers, you must specify a JMS connection.

JMS allows you to create durable subscribers on different connections with identical names. To accommodate this name scoping rule, the client IDs of the guaranteed Broker clients for the durable subscribers concatenate the connection client ID property and the durable subscriber names (supplied by the JMS client program). Thus, the durable subscriber names are identical in JMS but are associated with unique identifiers within Broker.

Message Delivery Mode

Each Broker message has a storage type of either *volatile* or *guaranteed*. The document storage type supports the JMS message delivery mode.

- A JMS message delivered with *persistent* delivery mode is represented by a Broker message with *guaranteed* storage type.
- A JMS message delivered with *non-persistent* delivery mode is represented by a Broker message with *volatile* storage type.

Whether a Broker message is written to durable storage depends on the combination of its storage type and the receiving Broker client's storage type. A Broker message is

written to durable storage only when its storage type and the receiving Broker client's storage type are both guaranteed.

Message Acknowledgement Mode

The JMS message acknowledgement modes are supported as follows:

- **DUPS_OK_ACKNOWLEDGE** is supported using the Broker's default message acknowledgement.
- **AUTO_ACKNOWLEDGE** is supported using `BrokerClient.acknowledge()` to acknowledge the current message received. The call is made by the underlying JMS session implementation.
- **CLIENT_ACKNOWLEDGE** is supported using `BrokerClient.acknowledgeThrough()` to acknowledge all the messages received. The method call is made by the JMS client application. In this mode, messages received by all message consumers belonging to the same session will be acknowledged.

Note: webMethods Broker allows for the acknowledgement of individual messages. In JMS, acknowledgement can only be set to all messages received or the last message received.

Temporary Topics and Queues

JMS temporary topics and queues are supported by Broker in a manner similar to regular topics and queues. Since there is no durability associated with a temporary topic or queue, the Broker clients that implement them are of the volatile storage type.

Temporary topics and queues are not associated with unique document types. All temporary topics use the document type `JMS::Temporary::Topic`. All temporary queues use the document type `JMS::Temporary::Queue`. Any JMS client application that is going to use temporary topics or temporary queues needs to have the appropriate can-publish or can-subscribe client group permissions set.

Binding Administered Objects in JNDI

The administered objects you use for JMS messaging (connection factories and destinations) and their properties are bound to locations in a standardized namespace built from a set of Java classes called JNDI (Java Naming and Directory Interface). The root location is called the *initial context*. JNDI serves as a template for the JMS administered objects, storing their property settings in specified locations relative to the initial context. These settings are applied when the JMS connection factories and destinations are created and used when the JMS application runs.

There may be cases where you do not want to use JNDI but want to configure the administered objects programmatically rather than administratively (that is, in your JMS application code). My webMethods supports the configuration of JMS applications

programmatically, using the `WmJMSAdminFactory` class. For more information, see the product Javadoc.

Note: Working with JMS administered objects programmatically has the advantage of reducing the amount of administration required; however, you cannot use JMS applications created programmatically with other JMS providers.

It is important to note that JNDI does not contain the JMS objects themselves, only the information necessary to configure them. For example, when working with a destination such as a topic, you must bind the topic in JNDI and, in a separate operation, create the topic on the Broker either programmatically using the JMS Admin API or with My webMethods.

Using Basic Authentication with JMS

If the connection between a JMS application and the Broker is secured through basic authentication, you first need to configure and enable basic authentication on the Broker Server. That process is detailed in [“Configuring SSL for Broker Server” on page 317](#).

When creating durable subscribers and queue clients for basic authentication secured JMS applications, you will need to supply the basic authentication user name and authorization identities for the JMS application.

For information about using the `JMSAdmin` command-line utility to configure basic authentication for JMS, see the *webMethods Broker Messaging Programmer’s Guide*.

Using SSL with JMS

If the connection between a JMS application and the Broker is secured through SSL, you first need to configure and enable SSL on the Broker Server. That process is detailed in [“Configuring SSL for Broker Server” on page 317](#).

You can then use My webMethods to configure SSL for a JMS application. This process involves configuring the connection factory that will be used to connect to the Broker Server, and consists of:

- Identifying the keystore file (repository of the SSL certificate) for the JMS application. This certificate file contains the JMS client application's SSL identity (also referred to as its SSL user identity). Since keystores are password protected, so you will need access to the JMS client keystore's password to configure it for SSL.
- Identifying the trust store file (repository of the trusted root of the SSL certificate) for the JMS application. This certificate file contains the SSL identity of the certification authority, or CA, of the JMS client's certificate (also referred to as its SSL authorization identity).

For configuration instructions, see [“Managing Connection Factories” on page 449](#).

When creating durable subscribers and queue clients for SSL-secured JMS applications, you will need to supply the SSL user and authorization identities for the JMS application (these are both SSL distinguished names). You need to have administrative access to the client's keystore file to obtain this information. For more information, see [“Keystore File” on page 312](#).

For information about using the JMSAdmin command-line utility to configure SSL for JMS, see the *webMethods Broker Messaging Programmer's Guide*.

Managing JMS Clients: A Road Map

You can administer your JMS applications using My webMethods. The table below describes the tasks required to administer a Broker/JMS client and where to go for detailed information or instructions about the step.

Step	Description	Look In...
Plan and configure client groups	<p>For the administered objects to be able to send and receive messages (or publish and subscribe to messages):</p> <ul style="list-style-type: none"> ■ Assign them to client groups. ■ Configure the client group so that it has permissions to publish and subscribe to the appropriate document types. ■ If necessary, configure basic authentication or SSL permissions for the client group (that is, its ACLs) so that the clients created by the JMS application can join the group. 	<p>“Managing Client Groups” on page 171, under:</p> <ul style="list-style-type: none"> ■ “Creating a Client Group” on page 179 ■ “Adding Document Type Permissions for Client Groups” on page 184 <p>For information about basic authentication or SSL permissions and client group ACLs, see “Client Group ACLs” on page 326.</p>
Establish the JNDI provider	Select and configure a supported JNDI provider to use as a naming directory.	“Managing JNDI Naming Directories” on page 443
Add the connection factories	Configure the connection factories for creating connections between the JMS programs and Broker Server.	“Adding Connection Factories” on page 452

Step	Description	Look In...
Bind the destinations	Configure the topics and queues in JNDI.	“Binding Destinations to JNDI” on page 471
Create the destinations	Create the topics and queues.	“About Creating Destinations” on page 472
Update client group permissions	Update the access permissions for client groups so they can publish and subscribe to the document types to use in durable subscriptions.	“Adding Document Type Permissions for Client Groups” on page 184
Create durable subscriptions	For subscriptions whose data is persisted if the subscriber is disconnected, link document type(s) to a topic.	“Creating a Durable Subscription” on page 478

Note: If you are configuring the JMS client's administered objects programmatically (that is, using the `WmJMSAdminFactory` class) rather than administratively, you do not establish a JNDI provider or bind the administered objects to JNDI.

Managing JNDI Naming Directories

This section describes how to manage and configure JNDI naming directories, which are used to store JMS administered objects.

For My webMethods to store administered objects on a server (for example, WebLogic, LDAP, or JBoss), the server must be active. WebLogic JNDI objects do not persist when you restart the WebLogic Server.

Note: If you are using WebLogic as the JNDI provider, make sure the `wm-jmsclient.jar` file is available under `samples\domains\wl_server` folder in your WebLogic installation directory. You will not be able to create connection factories, topics, and queues without the `wm-jmsclient.jar` file.

For LDAP servers, the server must either be running with schema validation turned off or it must have the Java object schema installed. The Java object schema is defined in RFC 2173.

If you are using the webMethods Naming Service for JMS (webMethods Naming Service) as the JNDI provider, make sure the Broker acting as the naming server is running. When you use webMethods Broker as a JNDI provider, make sure you do not use that Broker for publishing and subscribing messages.

Listing the JNDI Providers

You can list the JNDI providers that are installed and are being managed by Broker.

To list the JNDI providers

In My webMethods: **Messaging > Naming Directories > Providers**.

The **Providers List** displays the JNDI providers and their URLs.

Viewing a JNDI Provider's Properties

You can view the property settings for a JNDI provider.

To view a JNDI provider's property settings

1. In My webMethods: **Messaging > Naming Directories > Providers**.
2. Click a JNDI provider name from the JNDI Naming Service list. If the provider is not in the list, use the **Search** tab to locate it.

The JNDI provider property settings are displayed:

Field	Description
Provider Name	User-supplied name for the JNDI provider.
Initial Context Factory	The class name of the JNDI provider.
Provider URL	The URL of the session's initial context (that is, the JNDI directory in which to store JMS administered objects).
Security Principal	<p>The principal name, or user name supplied by My webMethods to the JNDI provider, if the provider requires one for accessing the JNDI directory.</p> <p>For information about whether a principal name is required, see the provider's product documentation.</p> <p>For information about the webMethods Naming Service security principal property, see the <i>webMethods Broker Messaging Programmer's Guide</i>.</p>

Field	Description
Security Credentials	<p>The credentials, or password, that My webMethods provides to the JNDI provider, if the provider requires security credentials to access the JNDI directory.</p> <p>For information about whether security credentials must be specified and their format, see the provider's product documentation.</p> <p>For information about the webMethods Naming Service security credentials property, see the <i>webMethods Broker Messaging Programmer's Guide</i>.</p>
Other Properties	<p>Any additional properties that the provider needs to configure JNDI.</p> <p>For information about whether a JNDI provider requires additional properties, see the product documentation for the provider.</p>
Provider JARs	<p>The CLASSPATH of any additional JAR files the JNDI provider needs to connect to the JNDI. These files must reside in a file system that is accessible to My webMethods.</p> <p>For more information about the JAR files and supporting classes a JNDI provider requires and the location of the files, see the product documentation for the provider.</p>
Number of Connection Factories	The number of connection factories that are bound to the namespace of this JNDI provider.
Number of Destinations	The number of topics and queues that are bound to the namespace of this JNDI provider.

- Optionally, click **Save Provider Properties** to save the setting for this JNDI provider in a properties file.

Editing a JNDI Provider's Properties

You can edit the property setting for a JNDI provider.

To edit a JNDI provider's property settings

- In My webMethods: **Messaging > Naming Directories > Providers**.

2. Click a JNDI provider name from the JNDI Naming Service list. If the provider is not in the list, use the **Search** tab to locate it.

3. Click the edit icon for the provider.

The JNDI provider property settings are displayed (see [“Viewing a JNDI Provider's Properties” on page 444](#)). If a field is editable, its setting is displayed in a text box.

4. Edit the property settings you want changed and click **OK**.

Adding a JNDI Provider

You can add and configure a new JNDI provider.

If you are adding WebLogic or JBoss as the JNDI provider, you must do *one* of the following:

- Include the additional JARs required by the JNDI provider in the CLASSPATH of My webMethods as explained in [“Including additional JAR files required by the JNDI provider” on page 448](#).
- Specify the CLASSPATH of the required JARs in the **Provider JARs** field as explained below.

To add and configure a new JNDI provider

1. In My webMethods: **Messaging > Naming Directories > Providers**.
2. Click **Add**.

The JNDI provider property settings are displayed (shown in the table below). If a field is editable, its setting is displayed in a text box.

Field	Description
Provider Name	User-supplied name for the JNDI provider.
Predefined Template	<p>The type of JNDI provider. Selections are:</p> <ul style="list-style-type: none"> ■ webMethodsNaming Service <p>For information about properties for the webMethods Naming Service, see the <i>webMethods Broker Messaging Programmer's Guide</i>.</p> <ul style="list-style-type: none"> ■ WebLogic ■ LDAP ■ JBoss

Field	Description
Initial Context Factory	The class name of the JNDI provider.
Provider URL	The URL of the session's initial context (that is, the JNDI directory in which to store JMS administered objects).
Security Principal	<p>The principal name, or user name supplied by My webMethods to the JNDI provider, if the provider requires one for accessing the JNDI directory.</p> <p>For information about whether a principal name is required, see the provider's product documentation.</p> <p>For information about the webMethods Naming Service security principal property, see the <i>webMethods Broker Messaging Programmer's Guide</i>.</p>
Security Credentials	<p>The credentials, or password, that My webMethods provides to the JNDI provider, if the provider requires security credentials to access the JNDI directory.</p> <p>For information about whether security credentials must be specified and their format, see the provider's product documentation.</p> <p>For information about the webMethods Naming Service security credentials property, see the <i>webMethods Broker Messaging Programmer's Guide</i>.</p>
Other Properties	<p>Any additional properties that the provider needs to configure the JNDI.</p> <p>The tooltip provides you the format for specifying the properties of a webMethods provider. For information about the JMS naming properties, see the <i>webMethods Broker Messaging Programmer's Guide</i>.</p> <p>For information about whether a JNDI provider requires additional properties, see the product documentation of that provider.</p>
Provider JARs	The CLASSPATH of any additional JAR files the JNDI provider needs to connect to the JNDI. These files must reside in a file system that is accessible to the Broker user interface.

Field	Description
	For more information about the JAR files and supporting classes a JNDI provider requires and the location of the files, see the product documentation for the provider.
Number of Connection Factories	The number of connection factories that are bound to the namespace of this JNDI provider.
Number of Destinations	The number of topics and queues that are bound to the namespace of this JNDI provider.

3. Perform one of the following steps:
 - a. Click **Predefined Template** and select a JNDI provider type from the list.
My webMethods fills in values for the provider's required fields. You can customize the property settings.
 - b. Upload a text file containing the JNDI provider properties by clicking on **Import Properties**, browsing for the JNDI properties file you want, and clicking **OK**.
4. Complete any additional property settings and click **OK**.

Including additional JAR files required by the JNDI provider

If you are adding WebLogic or JBoss as your JNDI provider, in the CLASSPATH of My webMethods, include the JARs required by the JNDI provider.

To add JAR files to the CLASSPATH of My webMethods

- If the JAR file is an OSGi bundle, do the following:
 1. Copy the JAR file to *Software AG_directory \<MWS instance name>\dropins* folder.
 2. Restart My webMethods.
- If the JAR file is not an OSGI bundle, do the following:
 1. Copy the JAR file into *Software AG_directory \MWS\lib* folder.
 2. Run the *Software AG_directory \MWS\bin\mws.bat* update command.
 3. Restart My webMethods.

For more information, see the "Adding Custom JAR Files" section in *Administering My webMethods Server*.

Managing Connection Factories

This section describes the management of connection factories. Connection factories can belong to several categories:

- Generic connection factories that allow their type (topic or queue) to be determined at application run time vs. connection factories whose type is predetermined.
- Connection factories for standalone JMS messaging applications vs. JMS applications running under application servers (XA connection factories). Only XA connection factories contain built-in support for transactions.

The steps in this section assume that you have already specified the JNDI provider and Broker with which to associate a connection factory and its configuration settings. If you have not done so, see [“Managing JNDI Naming Directories” on page 443](#).

Note: When naming JMS objects and subcontexts in an LDAP environment, you must include a prefix, such as `cn=` (common name), or `ou=` (organizational unit). My webMethods simplifies the use of LDAP service providers by allowing you to refer to object and context names without a prefix. If you do not supply a prefix, My webMethods automatically adds a default prefix.

Important: Before you create or modify a connection factory or destination, make sure the Broker that will store the JMS provider’s configuration settings is started. The Broker does not have to be started to view or remove a JMS administered object once that object is created.

Viewing the List of Connection Factories

Use the following procedure to view the list of connection factories.

To view the list of connection factories

1. In My webMethods: **Messaging > Naming Directories > Connection Factories**.
2. In the **Search** tab, select a JNDI provider from the **Provider Name** list.
3. Select a connection factory category by clicking one of the following **Type** selections on the **Search** tab (you can also select **All** to list connection factories of all types):
 - **ConnectionFactory**. A generic connection factory object, which can be used as either a `QueueConnectionFactory` or `TopicConnectionFactory` at run time.
 - **QueueConnectionFactory**. Used to make connections in a point-to-point application.
 - **TopicConnectionFactory**. Used to make connections in a publish-subscribe application.

- **XAConnectionFactory**. A generic connection factory object, where the object can be used in a transactional context as either an `XAQueueConnectionFactory` or `XATopicConnectionFactory` at run time.
- **XAQueueConnectionFactory**. Used to make connections in a point-to-point application, where the object will be used in a transactional context.
- **XATopicConnectionFactory**. Used to make connections in a publish-subscribe application, where the object will be used in a transactional context.

Note: You can only use XA connection factories with JMS applications running on application servers.

Viewing the Detailed Information About Connection Factories

Use the following procedure to view detailed information about a connection factory.

To view detailed information about a connection factory

1. Perform the steps listed in [“Viewing the List of Connection Factories” on page 449](#).

Note: The information for a cluster connection factory, used for the load balancing and client-side failover features, is different from that of non-cluster connection factories. For information about cluster connection factories, see [“Managing Cluster Connection Factories” on page 455](#).

2. Under **Lookup Name**, click the connection factory whose information you want to view.

The Connection Factory Detail page displays the following information:

Field	Description
Lookup Name	Name of the JNDI directory to which this connection factory is bound.
Provider	Name of the JNDI provider supporting this connection factory.
Connection Factory Type	For descriptions of each connection factory type, see “Viewing the Detailed Information About Connection Factories” on page 450 .
Connection Factory Client ID	The base client ID for connections created by this factory.

Field	Description
	<ul style="list-style-type: none"> ■ If a client ID is specified, the connection factory derives a client ID from this value each time a connection is established. ■ If a client ID is not specified, the connection factory assigns a unique client ID when a connection is established.
Broker Server	Name and port number of the controlling Broker Server.
Broker	<p>Select a Broker address from the list.</p> <p>This address is stored with the connection factory object so that JMS applications retrieving the object from JNDI connect to the correct Broker.</p>
Client Group	The name of the client group through which this connection factory connects.
Use SSL Encryption	<p>Whether traffic to connections made from this connection factory will be encrypted. The default is Yes.</p> <ul style="list-style-type: none"> ■ If set to Yes, all traffic to the Broker on this connection will be encrypted. ■ If set to No, the connection will be authenticated but no data will be encrypted.
Use Remote SSL Files	<p>Use the file paths for certificates as specified.</p> <p>If this setting is not selected, My webMethods uses the local, uploaded keystore and trust store names and converts them to file paths. You use the local option only if the file path is accessible to the client applications while establishing the connection to JNDI.</p>
SSL Keystore	<p>Absolute path and file name of the client keystore file that contains the certificate information for SSL connections.</p> <p>For more information about SSL keystores, see “Keystore File” on page 312.</p>
SSL Keystore Type	File type of the client keystore, which is PKCS12.

Field	Description
SSL Truststore	<p>Absolute path and file name of the truststore file containing the trusted root for the SSL certificate stored in the specified keystore.</p> <p>For more information about SSL truststores, see "Truststore File" on page 314.</p>
SSL Truststore Type	File type of the truststore certificate file, which is JKS.
Marshall In Class Name	<p>Fully qualified Java class name of the inbound marshaller if the JMS marshalling feature was used.</p> <p>Marshalling is an advanced feature that is used to connect to versions of Broker/JMS clients earlier than 6.5. For more information, see the "JMS Marshalling" appendix in the <i>webMethods Broker Messaging Programmer's Guide</i>.</p>
Marshall Out Class Name	<p>Fully qualified Java class name of the outbound marshaller if the JMS marshalling feature was used.</p> <p>Marshalling is an advanced feature that is used to connect to versions of Broker/JMS clients earlier than 6.5. For more information, see the "JMS Marshalling" appendix in the <i>webMethods Broker Messaging Programmer's Guide</i>.</p>

Adding Connection Factories

Use the following procedure to add a connection factory.

To add a connection factory

1. In My webMethods: **Messaging > Naming Directories > Connection Factories**.
2. On the Connection Factories List tab, click **Add Connection Factory**.
3. Complete the settings for the following on the Add Connection Factory page:

In this field...	Select or type...
Lookup Name	The name of the JNDI directory in which to bind this connection factory.

In this field...	Select or type...
Provider	The name of an existing JNDI naming directory from the list.
Connection Factory Type	See “Viewing the Detailed Information About Connection Factories” on page 450.
Connection Factory Client ID	<p>Optional. The base client ID for connections created by this connection factory.</p> <ul style="list-style-type: none"> ■ If you specify a client ID, the connection factory derives a client ID from this value each time a connection is established. ■ If you do <i>not</i> specify a client ID, the connection factory assigns a unique client ID when a connection is established.
Broker Server	The name and port number of the controlling Broker Server.
Broker	<p>Select a Broker address from the list.</p> <p>This address is stored with the connection factory object so that JMS applications retrieving the object from JNDI connect to the correct Broker.</p>
Client Group	The name of an existing client group or a new client group for this connection factory. If a new client group name is entered, that client group is created.
Use SSL Encryption	<p>Whether traffic to connections made from this connection factory will be encrypted. The default is Yes.</p> <ul style="list-style-type: none"> ■ If set to Yes, all traffic to the Broker on this connection will be encrypted. ■ If set to No, the connection will be authenticated but no data will be encrypted.
Use Remote SSL Files	<p>Use the file paths for certificates as specified.</p> <p>If this setting is not selected, My webMethods uses the local, uploaded keystore and trust store names and converts them to file paths.</p>

In this field...	Select or type...
SSL Keystore	<p>Absolute path and file name of the client keystore file that contains the certificate information for SSL connections.</p> <p>For more information about SSL keystores, see “Keystore File” on page 312.</p>
SSL Keystore Type	File type of the client keystore, which is PKCS12.
SSL Truststore	<p>Absolute path and file name of the truststore file containing the trusted root for the SSL certificate stored in the specified keystore.</p> <p>For more information about SSL truststores, see “Truststore File” on page 314.</p>
SSL Truststore Type	File type of the truststore certificate file, which is JKS.
Marshall In Class Name	<p>The fully qualified Java class name of the inbound marshaller if the JMS marshalling feature is used; otherwise, leave blank.</p> <p>Marshalling is an advanced feature that is used to connect to versions of Broker/JMS clients earlier than 6.5. For more information, see the "JMS Marshalling" appendix in the <i>webMethods Broker Messaging Programmer's Guide</i>.</p>
Marshall Out Class Name	<p>The fully qualified Java class name of the outbound marshaller if the JMS marshalling feature is used; otherwise, leave blank.</p> <p>Marshalling is an advanced feature that is used to connect to versions of Broker/JMS clients earlier than 6.5. For more information, see the "JMS Marshalling" appendix in the <i>webMethods Broker Messaging Programmer's Guide</i>.</p>

- Click **OK**.

Modifying Connection Factories

Use the following procedure to modify a connection factory.

To modify a connection factory

1. In My webMethods: **Messaging > Naming Directories > Connection Factories**.
2. In the **Search** tab, filter the list of connection factories by selecting the appropriate **Provider Name** and **Type**.
3. Locate the **Lookup Name** of the connection factory you want to modify, and then click its edit icon (last column of the table).
4. The Edit Connection Factory page is displayed (see [“Adding Connection Factories” on page 452](#) for a description of the properties).

You can modify the settings for **Connection Factory Client ID**, **Broker Server**, **Broker**, **Use SSL Encryption**, **Use Remote SSL Files**, **SSL Keystore**, and **SSL Truststore**.

Note: If you modify the **Connection Factory Client ID**, new clients created from this connection factory will use the new client ID; however, existing clients will retain the former client ID.

5. Click **OK**.

Deleting Connection Factories

Use the following procedure to delete a connection factory.

To delete one or more connection factories

1. In My webMethods: **Messaging > Naming Directories > Connection Factories**.
2. In the **Search** tab, filter the list of connection factories by selecting the appropriate **Provider Name** and **Type**.
3. Check the boxes next to the connection factories you want to delete.
4. Click **Delete**.

Managing Cluster Connection Factories

This section describes how to manage cluster connection factories.

Note: When naming JMS objects and subcontexts in an LDAP environment, you must include a prefix, such as `cn=` (common name) or `ou=` (organizational unit). My webMethods simplifies the use of LDAP service providers by allowing you to refer to object and context names without a prefix. If you do not supply a prefix, My webMethods automatically adds a default prefix.

Important: Before you create or modify a cluster connection factory, make sure the Brokers that will store the JMS provider's configuration settings are started.

The Brokers do not have to be started to view or remove a JMS administered object once that object is created.

Viewing a List of Cluster Connection Factories

Use the following procedure to view a list of cluster connection factories.

To view the list of cluster connection factories

1. In My webMethods: **Messaging > Naming Directories > Connection Factories**.
2. In the **Search** tab, select a JNDI provider from the **Provider Name** list.
3. Select a connection factory category by clicking one of the following **Type** selections on the **Search** tab (you can also select **All** to list connection factories of all types):
 - **ConnectionFactory**. A generic connection factory object, which can be used as either a `QueueConnectionFactory` or `TopicConnectionFactory` at run time.
 - **QueueConnectionFactory**. Used to make connections in a point-to-point application.
 - **TopicConnectionFactory**. Used to make connections in a publish-subscribe application.
 - **XAConnectionFactory**. A generic connection factory object, where the object can be used in a transactional context as either an `XAQueueConnectionFactory` or `XATopicConnectionFactory` at run time.
 - **XAQueueConnectionFactory**. Used to make connections in a point-to-point application, where the object will be used in a transactional context.
 - **XATopicConnectionFactory**. Used to make connections in a publish-subscribe application, where the object will be used in a transactional context.

Note: You can only use XA connection factories with JMS applications running on application servers.

Note: The connection factories list displays all connection factories, cluster connection factories, and composite cluster connection factories together. A **Yes** flag in the **Cluster Enabled** column indicates that the corresponding connection factory is a cluster connection factory.

Viewing Detailed Information About Cluster Connection Factories

Use the following procedure to view detailed information about a cluster connection factory.

To view detailed information about a cluster connection factory

1. Perform the steps listed in [“Viewing a List of Cluster Connection Factories” on page 456](#).

2. Under **Lookup Name**, click the cluster connection factory whose information you want to view.

The Connection Factory Detail page displays the following information:

Field	Description
Lookup Name	Name of the JNDI directory with which this cluster connection factory is bound.
Provider	Name of the JNDI provider supporting this cluster connection factory.
Connection Factory Type	For descriptions of each cluster connection factory type, see “Viewing the Detailed Information About Connection Factories” on page 450.
Connection Factory Client ID	<p>The base client ID for connections created by this factory.</p> <ul style="list-style-type: none"> ■ If a client ID is specified, the connection factory derives a client ID from this value each time a connection is established. ■ If a client ID is not specified, the connection factory assigns a unique client ID when a connection is established.
Cluster Name	The name of the cluster with which this cluster connection factory is associated.
Cluster Policy	<p>The load-balancing policy governing the cluster, namely:</p> <ul style="list-style-type: none"> ■ Round Robin ■ Sticky ■ Random ■ Weighted Round Robin ■ Multisend Guaranteed ■ Multisend Best Effort <p>For more information about the load-balancing policies, see “Load-Balancing Policies for Clusters” on page 462.</p>

Field	Description
Number OfBrokers	The maximum number of recipient Brokers in a cluster. This field is displayed only if you have selected Multisend Best Effort cluster policy. For more information, see “Editing Cluster Configuration” on page 463.
ClusterBrokers	List of Broker addresses bound with this cluster connection factory. JMS applications publishing to (or retrieving) documents from JNDI will use these Brokers for failover and load balancing.
Refresh Interval (minutes)	Default: 1440 minutes. After the specified minutes elapse, client applications refresh all connections to update changes made to the connection factory.
Client Group	Client group through which this cluster connection factory connects. The default is "JMSClient."
Use SSL Encryption	Whether traffic to connections made from this connection factory will be encrypted. The default is Yes . <ul style="list-style-type: none"> ■ If set to Yes, all traffic to the Broker on this connection will be encrypted. ■ If set to No, the connection will be authenticated but no data will be encrypted.
Use Remote SSL Files	Use the file paths for certificates as specified. If this setting is not selected, My webMethods uses the local, uploaded keystore and trust store names and converts them to file paths.
SSL Keystore	Absolute path and file name of the client keystore file that contains the certificate information for SSL connections. For more information about SSL keystores, see “Keystore File” on page 312.
SSL Keystore Type	File type of the client keystore, which is PKCS12.

Field	Description
SSL Truststore	<p>Absolute path and file name of the truststore file containing the trusted root for the SSL certificate stored in the specified keystore.</p> <p>For more information about SSL truststores, see "Truststore File" on page 314.</p>
SSL Truststore Type	File type of the truststore certificate file, which is JKS.
Marshall In Class Name	<p>Fully qualified Java class name of the inbound marshaller if the JMS marshalling feature was used.</p> <div style="border: 1px solid gray; padding: 5px; background-color: #f0f0f0;"> <p>Note: Marshalling is an advanced feature that is used to connect to versions of Broker/JMS clients earlier than 6.5. For more information, see the "JMS Marshalling" appendix in the <i>webMethods Broker Messaging Programmer's Guide</i>.</p> </div>
Marshall Out Class Name	<p>Fully qualified Java class name of the outbound marshaller if the JMS marshalling feature was used.</p> <p>Marshalling is an advanced feature that is used to connect to versions of Broker/JMS clients earlier than 6.5. For more information, see the "JMS Marshalling" appendix in the <i>webMethods Broker Messaging Programmer's Guide</i>.</p>

Adding Cluster Connection Factories

Use the following procedure to add a cluster connection factory.

To add a cluster connection factory

1. In My webMethods: **Messaging > Naming Directories > Connection Factories**.
2. On the Connection Factories List tab, click **Add Cluster Connection Factory**.
3. Click the **Cluster Connection Factory** tab.
4. Complete the following settings:

In this field...	Select or type...
Lookup Name	The name of the JNDI directory in which to bind this cluster connection factory.

In this field...	Select or type...
Provider	The name of an existing JNDI naming directory from the list.
Connection Factory Type	See “Viewing the Detailed Information About Connection Factories” on page 450.
Connection Factory Client ID	<p>Optional. The base client ID for connections created by this cluster connection factory.</p> <ul style="list-style-type: none"> ■ If you specify a client ID, the connection factory derives a client ID from this value each time a connection is established. ■ If you do <i>not</i> specify a client ID, the connection factory assigns a unique client ID when a connection is established.
Cluster Name	The name of the cluster with which this cluster connection factory is associated.
Cluster Configuration	You specify or edit the cluster configuration here. For more information, see “Editing Cluster Configuration” on page 463.
Client Group	The name of an existing client group or a new client group for this connection factory. If a new client group name is entered, that client group is created.
Use SSL Encryption	<p>Whether traffic to connections made from this connection factory will be encrypted. The default is Yes.</p> <ul style="list-style-type: none"> ■ If set to Yes, all traffic to the Broker on this connection will be encrypted. ■ If set to No, the connection will be authenticated but no data will be encrypted.
Use Remote SSL Files	<p>Use the file paths for certificates as specified.</p> <p>If this setting is not selected, My webMethods uses the local, uploaded keystore and trust store names and converts them to file paths.</p>

In this field...	Select or type...
SSL Keystore	<p>Absolute path and file name of the client keystore file that contains the certificate information for SSL connections.</p> <p>For more information about SSL keystores, see "Keystore File" on page 312.</p>
SSL Keystore Type	File type of the client keystore, which is PKCS12.
SSL Truststore	<p>Absolute path and file name of the truststore file containing the trusted root for the SSL certificate stored in the specified keystore.</p> <p>For more information about SSL truststores, see "Truststore File" on page 314.</p>
SSL Truststore Type	File type of the truststore certificate file, which is JKS.
Marshall In Class Name	<p>The fully qualified Java class name of the inbound marshaller if the JMS marshalling feature is used; otherwise, leave blank.</p> <p>Marshalling is an advanced feature that is used to connect to versions of Broker/JMS clients earlier than 6.5. For more information, see the "JMS Marshalling" appendix in the <i>webMethods Broker Messaging Programmer's Guide</i>.</p>
Marshall Out Class Name	<p>The fully qualified Java class name of the outbound marshaller if the JMS marshalling feature is used; otherwise, leave blank.</p> <p>Marshalling is an advanced feature that is used to connect to versions of Broker/JMS clients earlier than 6.5. For more information, see the "JMS Marshalling" appendix in the <i>webMethods Broker Messaging Programmer's Guide</i>.</p>

- You can edit the cluster configuration on the Add Cluster Connection factory page. For more information, see ["Editing Cluster Configuration" on page 463](#).
- Click **OK**.

Load-Balancing Policies for Clusters

A publisher sends JMS messages to a cluster based on the load-balancing policy you specify. You specify one of the following load-balancing policies for the cluster when you define the cluster connection factory:

- **Round Robin.** A policy where the first publish operation is routed to the first Broker in the cluster connection, the second publish operation is routed to the second Broker, and so on until all Brokers are used. Then the sequence repeats itself. Use this policy if you want to equally distribute the publish operations across all the Brokers in the cluster.
- **Sticky.** A policy where clients route all publish operations to the first Broker in the cluster connection. Only when the first Broker fails, the webMethods Broker Client API for JMS begins to route all publish operations to the next available Broker in the cluster. The publishing task will not be switched back from the second Broker to the first Broker even if the first Broker starts again unless the second Broker fails.
- **Random.** A policy where publish operations are routed randomly to any Broker in the cluster. It is possible that one of the Brokers might receive too many messages to process, while the other servers are idle.
- **Weighted Round Robin.** A policy where each Broker in the cluster handles a different message processing load. You can specify a "weight" to each Broker, which signifies how the Broker must share the message load relative to the other Brokers in the cluster. The weight determines how many more (or fewer) messages a Broker must receive, compared to the other Brokers in the cluster. You must carefully determine the relative weights to assign to each Broker.

The following examples explain the weighted round robin policy:

- If all Brokers in the cluster have the same weight, they will each share an equal proportion of the load.
- If one Broker has weight 1 and all other Brokers have weight 2, the Broker with weight 1 will bear half of the load compared to the other Brokers.
- If you have three Brokers in a cluster that must be governed by a weighted round robin policy, assign weight 4 for Broker 1, weight 12 for Broker 2, and weight 1 for Broker 3. In this example, for every 17 messages, Broker 1 gets 4 messages, Broker 2 gets 12 messages, and Broker 3 gets 1 message.
- **Multisend Guaranteed.** A multisend policy where a client publishes a message to x out of y Brokers in a cluster by using XA transactions. Each Broker in the cluster acts as an individual XA resource and the publish operation is considered to be successful if the message is received by x out of y Brokers in the cluster. For example, if there are 10 Brokers in cluster and you want to guarantee publishing to two Brokers, then you use two out of 10 Brokers. Multisend guaranteed policy works only with the XA connection type.

The client must use the correct JMS APIs to retrieve the XA resources, enlist the resources with the transaction manager, and perform the XA operations. The JMS APIs do not automatically perform these transaction management operations.

For more information about preventing duplicate message delivery by the JMS API to the subscriber, see the "Message Pre-Acknowledgement" section in the *webMethods Broker Messaging Programmer's Guide*.

- **Multisend Best Effort.** A multisend policy where a client publishes a message to all Brokers, or as many Brokers as possible, in the cluster. The publish operation is considered successful if even one of the Brokers receives the message. The message that the client publishes to the Brokers is non-transactional in nature. Multisend best effort policy does not work with the XA connection type.

For more information about preventing duplicate message delivery by the JMS API to the subscriber, see the "Message Pre-Acknowledgement" section in the *webMethods Broker Messaging Programmer's Guide*.

Editing Cluster Configuration

You can edit a cluster to change the load-balancing policy it uses. You can also select Brokers and set the refresh interval.

To edit a cluster configuration

1. In My webMethods: **Messaging > Naming Directories > Connection Factories**.
2. On the **Connection Factories List** page, click **Add Cluster Connection Factory**.
3. In the **Cluster Connection Factory** tab, click the **Edit** button.
4. Select one of the following load-balancing policies for the cluster:
 - **Round Robin**
 - **Sticky**
 - **Random**
 - **Weighted Round Robin**
 - **Multisend Guaranteed**
 - **Multisend Best Effort**

For more information about the load-balancing policies, see ["Load-Balancing Policies for Clusters"](#) on page 462.
5. Based on the load-balancing cluster policy you choose, specify the following details:

For this policy...	Do this...
Round Robin Sticky Random	<ol style="list-style-type: none"> a. Under Select Brokers for Cluster, select the Brokers that you want to associate with the cluster connection factory. The available options are: <ul style="list-style-type: none"> ■ All available Brokers. Select this option if you want to associate all available Brokers with the cluster. ■ Selected Brokers. Select this option if you want to choose Brokers from the list of available Brokers to associate with the cluster. b. If you chose the Selected Brokers option above, move Brokers from the Available list box to the Selected list box.
Weighted Round Robin	<ol style="list-style-type: none"> a. Under Select Brokers for Cluster, select Brokers that you want to associate with the cluster connection factory. b. Under Configuration, for each Broker listed in the table, specify the weight in the corresponding text box. The default is 1.
Multisend Guaranteed	<ol style="list-style-type: none"> a. Under Select Brokers for Cluster, select Brokers that you want to associate with the cluster connection factory. The available options are: <ul style="list-style-type: none"> ■ All available Brokers. Select this option if you want to associate all available Brokers with the cluster. ■ Selected Brokers. Select this option if you want to choose Brokers from the list of available Brokers to associate with the cluster. b. If you chose the Selected Brokers option above, move your Brokers from the Available list box to the Selected list box. c. In the Number of Brokers field, specify the number of guaranteed receiver Brokers in the cluster to which you want to send the messages. If the message is not delivered to the exact number of Brokers you specify here, the multi-send operation will fail. The value you specify for Number of Brokers must not be greater than the number of Selected Brokers.

For this policy...	Do this...
Multisend Best Effort	<ol style="list-style-type: none"> a. Under Select Brokers for Cluster, select Brokers that you want to associate with the cluster connection factory. The available options are: <ul style="list-style-type: none"> ■ All available Brokers. Select this option if you want to associate all available Brokers with the cluster. ■ Selected Brokers. Select this option if you want to choose Brokers from the list of available Brokers to associate with the cluster. b. If you chose the Selected Brokers option above, move Brokers from the Available list box to the Selected list box. c. Specify the maximum number of recipient Brokers in the cluster in the Number of Brokers field. If the number of available Brokers in the cluster is less than the value specified by Number of Brokers, the client sends the message to all the available Brokers. The default is All.

6. Specify a value for **Refresh Interval (minutes)** for the cluster. The default is 1440 minutes. After the specified minutes elapse, client applications refresh all connections to update changes made to the connection factory.
7. Select **Cluster Policy Override** to override a cluster policy. During runtime, you can specify the Brokers to which the messages must be sent in a cluster.
8. Click **Apply**.

Modifying Cluster Connection Factories

Use the following procedure to modify a cluster connection factory.

To modify a cluster connection factory

1. In My webMethods: **Messaging > Naming Directories > Connection Factories**.
2. In the **Search** tab, filter the list of cluster connection factories by selecting the appropriate **Provider Name** and **Type**.
3. Locate the **Lookup Name** of the cluster connection factory you want to modify, and then click its edit icon (last column of the table).

Note: You can identify a cluster connection factory by the **Yes** flag in the corresponding **Cluster Enabled** column.

The Edit Connection Factory page is displayed (see [“Adding Cluster Connection Factories” on page 459](#) for a description of the properties).

4. Modify the settings for connection factory client ID, remote SSL Files, SSL Keystore, SSL Truststore, cluster name, cluster policy, cluster Brokers, and refresh interval as desired.

Note: If you modify the **Connection Factory Client ID**, new clients created from this connection factory will use the new client ID; however, existing clients will retain the former client ID.

5. Click **OK**.

Deleting Cluster Connection Factories

Use the following procedure to delete a cluster connection factory.

To delete a cluster connection factory

1. In My webMethods: **Messaging > Naming Directories > Connection Factories**.
2. In the **Search** tab, filter the list of cluster connection factories by selecting the appropriate **Provider Name** and **Type**.

Note: You can identify a cluster connection factory by the **Yes** flag in the corresponding **Cluster Enabled** column.

3. Check the boxes next to the connection factories you want to delete.
4. Click **Delete**.

Managing Composite Cluster Connection Factories

This section describes how to manage composite cluster connection factories.

To view a list of composite cluster connection factories

1. In My webMethods: **Messaging > Naming Directories > Connection Factories**.
2. In the **Search** tab, select a JNDI provider from the **Provider Name** list.
3. Select a connection factory category by clicking one of the following **Type** selections on the **Search** tab (you can also select **All** to list connection factories of all types):
 - **ConnectionFactory**. A generic connection factory object, which can be used as either a `QueueConnectionFactory` or `TopicConnectionFactory` at run time.
 - **QueueConnectionFactory**. Used to make connections in a point-to-point application.
 - **TopicConnectionFactory**. Used to make connections in a publish-subscribe application.

- **XAConnectionFactory**. A generic connection factory object, where the object can be used in a transactional context as either an `XAQueueConnectionFactory` or `XATopicConnectionFactory` at run time.
- **XAQueueConnectionFactory**. Used to make connections in a point-to-point application, where the object will be used in a transactional context.
- **XATopicConnectionFactory**. Used to make connections in a publish-subscribe application, where the object will be used in a transactional context.

Note: You can only use XA connection factories with JMS applications running on application servers.

Note: The connection factories list displays all connection factories, cluster connection factories, and composite cluster connection factories together. A **Yes** flag in the **Composite Cluster** column indicates that the corresponding connection factory is a composite cluster connection factory.

Viewing Detailed Information About Composite Cluster Connection Factories

Use the following procedure to view detailed information about a composite cluster connection factory.

To view detailed information about a composite cluster connection factory

1. Perform the steps listed in [“Managing Composite Cluster Connection Factories” on page 466](#).
2. Under **Lookup Name**, click the cluster connection factory whose information you want to view.

The Composite Connection Factory Detail page displays the following information:

Field	Description
Lookup Name	The JNDI directory with which this composite cluster connection factory is bound.
Provider	The JNDI provider supporting this composite cluster connection factory.
Composite Cluster Policy	The load-balancing policy that selected for this composite cluster, namely: <ul style="list-style-type: none"> ■ Round Robin ■ Sticky

Field	Description
	<ul style="list-style-type: none"> ■ Random ■ Multisend Best Effort ■ Multisend Guaranteed <p>For more information about cluster policies, see “Editing Cluster Configuration” on page 463.</p>
Connection Factory Type	The connection factory type selected for this composite cluster connection factory. For more information about connection factory types, see “Managing Composite Cluster Connection Factories” on page 466.
Selected Connection Factories	The cluster connection factories that you selected for this composite cluster connection factory.
Refresh Interval	After the specified minutes elapse, client applications refresh all connections to update changes made to the connection factory. The default is 1440 minutes.
Cluster Policy Override	Whether to override the cluster policy. If you select Cluster Policy Override , you can specify the Brokers to which the messages must be sent in a cluster.

Adding Composite Cluster Connection Factories

Use the following procedure to add a composite cluster connection factory.

To add a composite cluster connection factory

1. In My webMethods: **Messaging > Naming Directories > Connection Factories**.
2. On the **Connection Factories List** tab, click **Add Cluster Connection Factory**.
3. Click the **Composite Cluster Connection Factory** tab.
4. Complete the following settings:

In this field...	Select or type...
Lookup Name	The name of the JNDI directory in which to bind this connection factory.

In this field...	Select or type...
Provider	The name of an existing JNDI naming directory from the list.
Cluster Policy	<p>One of the following load-balancing policies for the composite cluster:</p> <ul style="list-style-type: none"> ■ Round Robin ■ Sticky ■ Random ■ Multisend Best Effort ■ Multisend Guaranteed <p>For more information about cluster policies, see “Load-Balancing Policies for Clusters” on page 462.</p>
Connection Factory Type	A connection factory type for this composite cluster connection factory. For more information, see “Managing Composite Cluster Connection Factories” on page 466.
Select Connection Factories	<p>The cluster connection factories that you want to associate with this composite cluster connection factory.</p> <p>To select cluster connection factories, move them from the Available list box to the Selected list box.</p>
Refresh Interval (minutes)	The refresh interval in minutes. After the specified minutes elapse, client applications refresh all connections to update changes made to the connection factory. The default is 1440 minutes.
Cluster Policy Override	If you want to specify the Brokers to which the messages must be sent in a cluster.

5. Click **OK**.

Modifying Composite Cluster Connection Factories

Use the following procedure to modify a composite cluster connection factory.

To modify a composite cluster connection factory

1. In My webMethods: **Messaging > Naming Directories > Connection Factories**.

2. In the **Search** tab, filter the list of composite cluster connection factories by selecting the appropriate **Provider Name** and **Type**.
3. Locate the **Lookup Name** of the composite cluster connection factory you want to modify, then click its edit icon (last column of the table).

Note: You can identify a composite cluster connection factory by the **Yes** flag in the corresponding **Composite Cluster** column.

4. Modify the settings for **Cluster Policy**, **Connection Factory Type**, **Selected Connection Factories**, and **Refresh Interval (minutes)** as desired.
5. Click **OK**.

Deleting Composite Cluster Connection Factories

Use the following procedure to delete a composite cluster connection factory.

To delete one or more composite cluster connection factories

1. In My webMethods: **Messaging > Naming Directories > Connection Factories**.
2. In the **Search** tab, filter the list of composite cluster connection factories by selecting the appropriate **Provider Name** and **Type**.

Note: You can identify a composite cluster connection factory by the **Yes** flag in the corresponding **Composite Cluster** column.

3. Select the check boxes next to the connection factories you want to delete.
4. Click **Delete**.

Managing Destinations

This section describes how to manage JMS destinations, which are queues and topics. Managing JMS destinations consists of the following high-level tasks:

- Binding destinations to JNDI
- Creating destinations
- Viewing and editing destinations
- Deleting destinations

Note: The JMS standard defines *temporary queues* and *temporary topics*, which are objects that last only for the duration of a connection. These destinations can only be created programmatically; they cannot be created through My webMethods. However, you can view topics created programmatically through My webMethods.

Binding Destinations to JNDI

This procedure describes how to bind queues and topics in JNDI. Follow these instructions only if you are configuring destinations administratively.

To bind a destination to JNDI

1. In My webMethods: **Messaging > Naming Directories > Destinations**.
2. In the **Destinations List**, click **Add**.
3. Complete the following fields:

In this field...	Select or type...
Lookup Name	The name of the JNDI directory in which to bind this destination. You will use the lookup name later to create this destination object.
Provider	The name of the JNDI provider to use with this destination.

Destination Name

Queues:

A fully qualified name (for example, `/territory/broker/queuname`) or a partial name. Use the following guidelines:

- The name cannot exceed 255 bytes.
- The name must begin with an alphabet, followed by any combination of alphanumeric characters and underscores (`_`).
- If you use a naming prefix, separate it from the queue name with a double colon. (`: :`).

Note: Space character is allowed only when using the JMSAdmin command-line tool.

Topics:

A valid topic name or wild card (for example, `*`). Use the following guidelines:

- The name cannot exceed 255 bytes.
- The name must begin with an alphabet, followed by any combination of alphanumeric characters and underscores (`_`).

In this field...	Select or type...
	<ul style="list-style-type: none"> ■ If you use a naming prefix, separate it from the topic name with a double colon. (: :). <p>Note: Wild card character (*) is allowed only when using the JMSAdmin command-line tool.</p>
Destination Type	Whether the destination is a queue or topic .
Shared State	<p>Optional. Sharing a client state allows multiple clients, each using its own session, to process documents from a single client queue in parallel on a first-come, first-served basis.</p> <p>Select whether multiple connections can share the same connection client ID.</p>
Shared State Ordering	<p>Optional. The ordering of documents received on a shared state client:</p> <ul style="list-style-type: none"> ■ To receive documents in order from a publisher, set to publisher. ■ To receive documents in any order, set to none. <p>Note: You can set shared state ordering only after you select the check box next to Shared State. If you do not select the check box, shared state ordering defaults to publisher.</p>

4. Click **OK**.

About Creating Destinations

The procedures in this section detail how to create JMS queues and topics.

You create destinations either **From Naming Directories** (JNDI) or **At the Broker**.

- Use the **From Naming Directories** tab when you are binding the destination to JNDI.
- Use the **At the Broker** tab when you are configuring the destination programmatically (not using JNDI).

Creating a JMS Topic From a Naming Directory (JNDI)

Use the following procedure to create a JMS topic from a naming directory (JNDI).

To create a JMS topic from a naming directory (JNDI)

1. In My webMethods: **Messaging > Broker Servers > Document Types**.
2. In the **Document Type List**, click **Add Topic**.
3. Select the **From Naming Directory** tab.
4. Complete the following fields:

In this field...	Select or type...
Provider Name	The name of the JNDI provider in which the topic is bound from the list.
Topic	The topic lookup name. This is the Lookup Name that you used previously when binding the topic to JNDI.
Broker Server	The Broker Server for this topic.
Broker	The Broker for this topic.

5. Click **OK**.

A message displays indicating that the topic was created at the selected Broker.

Creating a JMS Topic Without Using a Naming Directory (JNDI)

Use the following procedure to create a JMS topic at the Broker, without using JNDI.

To create a JMS topic at the Broker (create a topic without using JNDI)

1. In My webMethods: **Messaging > Broker Servers > Document Types**.
2. In the **Document Type List**, click **Add Topic**.
3. Click **Add Topic**.
4. Select the **AtBroker** tab.
5. Complete the following fields:

In this field...	Select or type...
Broker Server	The Broker Server for this topic.
Broker	The Broker for this topic.
Document Type	The name of the document type that the topic contains.

- Click **OK**.

A message displays indicating that the topic was created at the selected Broker.

Creating a JMS Queue From a Naming Directory (JNDI)

Use the following procedure to create a JMS queue from a naming directory (JNDI).

To create a JMS queue from a naming directory (JNDI)

- In My webMethods: **Messaging > Broker Servers > Clients**.
- In the **Client List**, click **Add Queue**.
- Select the **From Naming Directory** tab.
- Complete the following fields:

In this field...	Select, type, or view...
Provider Name	The name of the JNDI provider in which the queue is bound.
Queue Lookup Name	The lookup name that you used previously when binding the queue to JNDI.
Connection Factory Lookup Name	The JNDI lookup name for the connection factory that you will use to create the queue.
Broker Server	The Broker Server for this queue.
Broker	The Broker for this queue.
User Name or SSL DN	The basic authentication user name or SSL user DN of the JMS client, if it is basic authentication/SSL-secured.
Alias Name or SSL Issuer DN	The basic authenticator alias or the SSL DN of the certification authority for the JMS client, if it is basic authentication/SSL-secured.

- Click **OK**.

A message displays indicating that the queue was created at the selected Broker.

Creating a JMS Queue Without Using a Naming Directory (JNDI)

Use the following procedure to create a JMS queue at the Broker, without using JNDI.

To create a JMS queue at the Broker (create a queue without using JNDI)

1. In My webMethods: **Messaging > Broker Servers > Clients**.
2. In the **Client List**, click **Add Queue**.
3. Select the **At Broker** tab.
4. Complete the following fields:

In this field...	Select or type...
Queue Name (Brokerclient ID)	A unique identifier for the queue.
Broker Server	The Broker Server for this queue.
Broker	The Broker for this queue.
Client Group	The name of the client group to which the queue is assigned.
Application Name	Optional. A name that is meaningful to your application.
Shared State	<p>Optional. Sharing a client state allows multiple clients, each using its own session, to process documents from a single client queue in parallel on a first-come, first-served basis.</p> <p>Select whether multiple connections can share the same connection client ID.</p>
Shared State Order	<p>Specify the ordering of documents received on a shared state client.</p> <ul style="list-style-type: none"> ■ To receive documents in order from a publisher, set to publisher. ■ To receive documents in any order, set to none. <div style="border: 1px solid gray; padding: 5px; margin-top: 10px;"> <p>Note: You can set shared state ordering only after you select the check box next to Shared State. If you do not select the check box, shared state ordering will default to publisher.</p> </div>
User Name or SSL DN	The basic authentication user name or the SSL user DN of the JMS client, if it is basic authentication/SSL-secured.

In this field...	Select or type...
Alias Name or SSL Issuer DN	The basic authenticator alias or the SSL DN of the certification authority for the JMS client, if it is basic authentication/SSL-secured.

- Click **OK**.

A message displays indicating that the queue was created at the selected Broker.

Viewing and Editing Destinations

This procedure describes how to view and edit a queue or topic. Note that you can only edit fields that specify a destination's shared state properties.

To view or edit a destination

- In My webMethods: **Messaging > Naming Directories > Destinations**.
- Select the destination's JNDI **Provider Name**.
- Filter the data displayed in the Destinations page by selecting a **Type (Topic, Queue, or All)**.
- Locate the destination whose information you want to view or edit, and then perform one of the following steps:
 - To view details about the destination, click the destination's **Lookup Name**.
 - To modify the destination's properties, click the destination's **Edit** icon.

The destination properties are as follows:

Field	Description
Lookup Name	Name of the JNDI directory to which the destination is bound (read-only).
Provider	Name of the JNDI provider for this destination (read-only).
Destination Name	Name of the topic or queue (read-only).
Destination Type	Identifies whether the destination object is a queue or topic (read-only).
Shared State	Sharing a client state allows multiple clients, each using its own session, to process documents from a

Field	Description
	<p>single client queue in parallel on a first-come, first-served basis.</p> <p>Select whether multiple connections can share the same connection client ID.</p>
Shared State Order	<p>Specify the ordering of documents received on a shared state client.</p> <ul style="list-style-type: none"> ■ To receive documents in order from a publisher, set to publisher. ■ To receive documents in any order, set to none. <p>Note: You can set shared state ordering only after you select the check box next to Shared State. If you do not select the check box, shared state ordering will default to publisher.</p>

Deleting a Destination

This section describes how to delete queues and topics.

To delete a destination

1. In My webMethods: **Messaging > Naming Directories > Destinations**.
2. In the Destinations List, under **Lookup Name**, select the check boxes next to the destinations that you want to delete.
3. Click **Delete**.

Assigning Publish and Subscribe Permissions

You assign publish and subscribe permissions for Broker/JMS clients at the client group level (the same as with non-JMS Broker clients). Each Broker/JMS client is assigned to a client group, and membership in a client group determines whether a client is allowed to publish or subscribe to document types associated with the group.

For more information, see [“Adding Document Type Permissions for Client Groups” on page 184](#).

Creating a Durable Subscription

Before creating a JMS durable subscription, make sure that the following prerequisites are met:

- If the durable subscriber's administered objects are stored in JNDI, then:
 - You must have already configured the JNDI provider as a naming directory in My webMethods.
 - You must have already configured the JMS administered objects (connection factory and topic) in My webMethods.

For more information, see [“Adding a JNDI Provider” on page 446](#) and [“Binding Administered Objects in JNDI” on page 440](#).

- If the client group to which a durable subscriber belongs is basic authentication protected, you will need the client's basic authentication user name and authenticator alias.
- If the client group to which a durable subscriber belongs is SSL-protected, you will need the client's SSL user and authorization DNs. That information is stored in the client's keystore. You will need administrator access (user name and password) to the client keystore file, and the awcert utility to obtain those DNs. For more information, see [“Keystore File” on page 312](#).

Note: You cannot create non-durable subscribers using My webMethods; however, you can view non-durable subscribers created programmatically (that is, created outside of the Broker user interface).

To create a durable subscription

1. In My webMethods: **Messaging > Broker Servers > Clients**.
2. On the **Search** tab, enter the **Server Name** and **BrokerName** to which the durable subscriber will belong and click **Go**.
3. Click **Add Durable Subscription**.
4. To create the durable subscriber from JMS objects bound to a JNDI namespace, click the **From Naming Directory** tab and complete the following fields:

In this field...	Select or type...
Durable Subscriber Name	A name for this durable subscriber. On the Clients panel, the Client ID is the durable subscriber name prefixed by its connection client ID and two pound sign (##) delimiters.

In this field...	Select or type...
Provider Name	The JNDI provider to which the administered objects for this durable subscriber are bound.
Topic Lookup Name	<p>A JNDI lookup name for the durable subscription topic. Use the following guidelines:</p> <ul style="list-style-type: none"> ■ The name cannot exceed 255 bytes. ■ The name must begin with an alphabetic character, followed by any combination of alphanumeric characters and underscores (_).
Connection Factory Lookup Name	The JNDI lookup name for the connection factory.
Broker Server	The name of the controlling Broker Server for the durable subscriber (read-only).
Broker	The name of the controlling Broker for the durable subscription (read-only).
Selector String (SQL92 Syntax)	<p>Optional. A message selector to filter subscriptions. For information about using JMS message selectors, see the <i>webMethods Broker Messaging Programmer's Guide</i>.</p>
No local	Optional. If you select this option, no subscriptions originating from the local machine will be received.
User Name or SSL DN	<p>The basic authentication user name or the SSL DN of the user(s) receiving the durable subscription.</p> <p>Provide this information only if the client group to which the durable subscribers belong is basic authentication/SSL-protected by a user ACL.</p>
Alias Name or SSL Issuer DN	<p>The basic authenticator alias or the SSL DN of the certification authority for the durable subscribers.</p> <p>Provide this information only if the client group to which the durable subscribers belong is basic authentication/SSL-protected by an authenticator or authorization (certificate issuer) ACL.</p>

5. If you are creating the JMS administered objects for the durable subscription programmatically (that is, you are not using JNDI), click the **AtBroker** tab and complete the following fields:

In this field...	Select or type...
Durable Subscriber Name	A name for the durable subscriber. On the Clients panel, the Client ID is the durable subscriber name prefixed by its connection client ID and two pound sign (##) delimiters.
Connection Client ID	A unique client identifier.
Broker Server	The controlling Broker Server for the durable subscription.
Broker	The controlling Broker for the durable subscription.
Client Group	The client group of the durable subscriber(s).
Document Type Name	The document type that the durable subscription contains.
Application Name	Optional. A name that is meaningful to your application. By default, the name "JMS" is used.
Shared State	Optional. Sharing a client state allows multiple clients, each using its own session, to process documents from a single client queue in parallel on a first-come, first-served basis. Select whether multiple connections can share the same connection client ID.
Shared State Order	Optional. The ordering of documents received on a shared state client: <ul style="list-style-type: none"> ■ To receive documents ordered by publisher, select publisher. ■ To receive documents in any order, select none. <div style="border: 1px solid gray; padding: 5px; margin-top: 10px;"> <p>Note: You can set shared state ordering only after you select the check box next to Shared State. If you</p> </div>

In this field...	Select or type...
	do not select the check box, shared state ordering will default to publisher .
Selector String (SQL92 Syntax)	Optional. The message selector that will be used to filter subscriptions. For information about using JMS message selectors, see the <i>webMethods Broker Messaging Programmer's Guide</i> .
No local	Optional. If you select this option, no subscriptions originating from the local machine will be received.
User Name or SSL DN	The basic authentication user name or the SSL DN of the user(s) receiving the durable subscription. Provide this information only if the client group to which the durable subscribers belong is basic authentication/SSL-protected by a user ACL.
Alias Name or SSL Issuer DN	The basic authenticator alias or the SSL DN of the certification authority for the durable subscribers. Provide this information only if the client group to which the durable subscribers belong is basic authentication/SSL-protected by an authenticator or authorization (certificate issuer) ACL.

19 Exporting and Importing Broker Metadata

■ Overview	484
■ Description of Functionality	484
■ Using Command-Line Utilities for Export and Import	484
■ Replicating Broker Metadata	485
■ Importing and Exporting Territories and Clusters	486

Overview

This chapter describes the webMethods Broker export and import functionality, which allows you to save metadata about Broker objects and reconstruct the objects with that data. The chapter explains how the metadata is saved, describes using both My webMethods and the command-line utilities for this functionality, and explains the steps needed to export metadata to a file and import the metadata back into Broker.

Description of Functionality

Broker includes capabilities that allow you to:

- Export a Broker object and its dependent objects to a file.

Exporting the metadata to a file allows you save the configuration settings for an entire hierarchy of related Broker objects (territory, territory gateway, cluster, cluster gateway, Broker Servers, Brokers, client groups, document types, clients), and re-create them at a later time. You can select which objects to include or exclude when exporting metadata.

- Import a saved file of Broker metadata.

You can import the saved Broker metadata, effectively re-creating the Broker objects and their relationships. You can import saved metadata to any valid location in your Broker network.

Using Command-Line Utilities for Export and Import

You can use the Broker command-line utilities to export and import Broker metadata. However, there are several differences between export and import with the command-line utilities and export and import with My webMethods:

- Each execution of a command-line utility allows the export and import of metadata for a single Broker; My webMethods allows you to export entire branches of Broker object data from the level of the territory downwards.
- With the command-line utilities, you cannot select which dependent objects to include in the saved Broker metadata; you must include or exclude all of them. Using My webMethods, you can select which dependent objects to export or import.

Exporting Broker Data

Use the `broker_save` utility from the command line to save Broker configuration information for a specified Broker to an ADL file. The utility includes options that allow you to save configuration information about the territory and Broker Server to which the Broker belongs, the basic authentication/SSL information needed to make a connection,

and the basic authentication/SSL information required by a client group Access Control List (ACL).

For more information, see [“broker_save” on page 558](#).

Importing Broker Data

Use the `broker_load` command-line utility to import Broker object configuration data from an ADL file generated by `broker_save`. Once the import has taken place, the Broker objects are re-created at the selected Broker location.

For more information, see [“broker_load” on page 552](#).

Automating Export and Import with Command Files

You can automate the saving (export) and loading (import) of Broker metadata by executing `broker_save` and `broker_load` from a command file.

Following is an example of a command file that exports metadata for four Brokers to four ADL files.

```
set BROKER_BIN=c:\webMethods Broker_directory\bin\
set BROKER_HOST=localhost:9000
set CG_ADL=ClientGroups_SSL.adl
set CERT_FILE=keystore01.cert
set DN="CN=ABC Manufacturing 1"

set BROKER1=Broker00010@%BROKER_HOST%
set BROKER2=Broker00020@%BROKER_HOST%
set BROKER3=Broker00030@%BROKER_HOST%
set BROKER4=Broker00040@%BROKER_HOST%

broker_save -broker brk01.adl %BROKER1% -certfile %CERT_FILE% -dn %DN%
broker_save -broker brk02.adl %BROKER2% -certfile %CERT_FILE% -dn %DN%
broker_save -broker brk03.adl %BROKER3% -certfile %CERT_FILE% -dn %DN%
broker_save -broker brk04.adl %BROKER4% -certfile %CERT_FILE% -dn %DN%
```

Replicating Broker Metadata

The export and import capability is useful for copying large numbers of related Broker objects to another location, as it significantly reduces the time required for object creation and configuration.

You can use the export and import functionality:

- To replicate hierarchies (or branches) of Broker objects from one location in your Broker system to another (for example, replicating a Broker Server and its dependent objects to a different territory).
- To replicate a hierarchy of objects to a different Broker installation.

Using export and import to replicate Broker metadata saves substantial administration work because this process does most of your Broker object creation and configuration.

You may only need to rename some Broker objects and reset their properties, rather than create and reconfigure hundreds of Broker objects one at a time.

Building a Production System Using Export and Import

You can use the export and import capability to generate a Broker production system from a development system. Saving a Broker development system's metadata to one or more ADL files, and importing the metadata to their desired locations in your network, accelerates the process of putting your Broker system into production. A high-level road map for this process involves the following steps:

1. Decide on a design for the Broker production system; then, decide the levels of the Broker development system to replicate.
2. Depending on the scale and design of your Broker development system, select whether to export metadata for a specific territory or for the objects belonging to a Broker Server. If necessary, select the subsets of objects and configuration data you want to save. Export the metadata from each individual Broker in the system.
3. Use My webMethods to export the Broker objects to an ADL file. Repeat the process as many times as necessary until all the Broker objects you want to retain are replicated.
4. Use My webMethods to import the Broker objects to the Broker environment that will contain the production system. Specify the import objects' target locations, and then begin importing the ADL files. Repeat the process as many times as necessary.
5. If necessary, perform any reconfiguration work on the Broker objects that were just replicated.

Note: You can rename territories, as well as Broker Servers and Brokers imported into a different territory. However, you cannot rename the lower-level objects such as client groups, clients, and document types.

Importing and Exporting Territories and Clusters

When exporting or importing from the territory level downwards, the hierarchy of objects displayed is as follows:

```
+ Territory
  + Broker1
    + Client groups
    + Clients
    + Document types
  + Broker2
  ...
  + BrokerN
  ...
  + Gateways
```

Note: Broker metadata is not batched during an import. If a Broker disconnection interrupts the data stream during an import, data that was imported prior to

the service interruption is saved by Broker; however, the remaining data must still be imported.

Following are step-by-step instructions for exporting and importing territories and clusters using My webMethods.

Exporting Territories

You can export metadata about a Broker territory and its dependent objects in the format of an XML file.

To export a territory

1. In My webMethods: **Messaging > Broker Territories > Territories**.
2. In the **Territories List**, click the territory that includes the metadata you want to export. If the territory is not in the list, use the **Search** tab to locate it.
3. On the **Territory Details** page, click the **Export** tab.
My webMethods displays the **Territory Content** in a tree view.
4. Select the check box next to the territory whose metadata you want to export.
5. Click the plus (+) tab to expand the nodes for the territory, and click to expand any dependent nodes. You can optionally include or exclude any dependent object in a territory by checking the box next to its name or leaving it blank.
6. Click **Export**.
7. On the Export Territory page, click **Include System Defined Data** if you want that data in the export file.
System-defined data includes information such as the system-supplied document types and client groups. You cannot import this data back into Broker; however, you may want to print out the export file to read the information.
8. Click **Export**.
9. The Export Confirmation Page displays information about the status of the export. Click **Proceed**.
10. Select whether to **Open** or **Save** the XML file in the dialog box and click **OK**.

Importing Territories

You can import metadata about a Broker territory and its dependent objects in the format of an XML file.

To import a territory

1. In My webMethods: **Messaging > Broker Territories > Territories**.
2. On the Territories page, click **Import**.

3. On the Territory Import page, check to see whether the export file content is listed. If the file is not listed, do the following:
 - a. Click **Upload File**.
 - b. On the Import Territory page, click **Browse** and select the XML file to import.
 - c. Click **Upload**.
4. In the **Territory Import** list, locate the territory whose metadata you want to import.
If you want to select dependent objects to include or exclude, expand the node for the territory and any dependent nodes, and then check the appropriate boxes.
5. The **Target Object** column provides links that allow you to set or change the location of the imported metadata for the data specified by the row.
 - If the text in the **Target** column for the territory reads *No target specified*, specify a location for the import operation by first clicking **Change Target**.
 - If a location is already specified, changing the location is optional.
6. If you selected **Change Target**, do one of the following on the Territory Import Broker Target page:
 - If the Broker Server and Broker to which you are importing the metadata already exist, select the **SelectBroker** tab and select a **Target Server Name** and **TargetBrokerName** from their respective lists. Click **OK**.
 - If the Broker to which you are importing the metadata is new, select the **NewBroker** tab, select a **Target Server Name**, and type the name of the new **TargetBrokerName**. Click **OK**.

Then, on the Territory Import page, reselect the objects to import.
7. Click **Import**.

Exporting Territory Gateway Information

You can only save Broker gateway information when exporting from a territory that contains gateway configuration information.

To export territory gateway information

1. In My webMethods: **Messaging > Broker Territories > Territories**.
2. In the **Territories List**, click the territory that includes the gateway configuration information that you want to export. If the territory is not in the list, use the **Search** tab to locate it.
3. On the **Territory Details** page, click the **Export** tab.

My webMethods displays the **Territory Content** in a tree view.

4. Click the plus (+) tab to expand the nodes for the territory. Expand the **Gateways** node to verify that the gateways whose metadata that you want to export is listed.
5. Click the box for the territory object, and for the gateways whose metadata you want to export.

Note: You must select the territory object for the selected gateway(s).

6. Click **Export**.
7. On the Export Territory page, you can optionally **Include System Defined Data** in the export file.
8. Click **Export**.
9. The Export Confirmation Page displays information about the status of the export. Click **Proceed**.
10. Select whether to **Open** or **Save** the XML file in the dialog box and click **OK**.

Importing Territory Gateway Information

Use the following procedure to import territory gateway information.

To import territory gateway information

1. In My webMethods: **Messaging > Broker Territories > Territories**.
2. On the Territories page, click **Import**.
3. On the Territory Import page, check to see whether the export file content is listed. If the file is not listed, do the following:
 - a. Click **Upload File**.
 - b. On the Import Territory page, click **Browse** and select the XML file to import.
4. In the **Territory Import** list, locate the territory that includes the gateway configuration information you want to import.

If you want to select dependent objects to include or exclude, expand the node for the territory and check the boxes for the objects to include.

5. Make sure the box for **Gateways**, or one of its dependent nodes, is selected, as well as the territory object to which the gateway belongs.

Note: You must select the territory object for the selected gateway(s).

6. The **Target** column provides links that allow you to set or change the location of the imported metadata.
 - If the text in the **Target** column for the territory reads `No target specified`, specify a location for the import operation by first clicking **Change Target**.

- If a location is already specified, changing the location is optional.
7. If you selected **Change Target**, do one of the following on the Territory Import Broker Target page:
 - If the Broker Server and Broker to which you are importing the metadata already exist, select the **SelectBroker** tab and select a **Target Server Name** and **TargetBrokerName** from their respective lists.
 - If the Broker to which you are importing the metadata is new, select the **NewBroker** tab, select a **Target Server Name**, and type the name of the new **TargetBrokerName**.
 8. Click **OK**.
 9. Click **Import**.

Exporting Clusters

You can export metadata about a Broker cluster and its dependent objects in the format of an XML file.

To export a cluster

1. In My webMethods: **Messaging > BrokerClusters > Clusters**.
2. In the **Clusters List**, click the cluster that includes the metadata you want to export. If the cluster is not in the list, use the **Search** tab to locate it.
3. On the **Cluster Details** page, click the **Export** tab.

My webMethods displays the cluster content in a tree view.
4. Click the box next to the cluster whose metadata you want to export.
5. Click the plus (+) tab to expand the nodes for the cluster, and click to expand any dependent nodes. You can optionally include or exclude any dependent object in a cluster by checking the box next to its name or leaving it blank.
6. Click **Export**.
7. The Export Confirmation Page displays information about the status of the export.
 - a. If you want to include system-defined data in the export file, click **Include System Defined Data**.

System-defined data includes information such as the system-supplied document types and client groups. You cannot import this data back into Broker; however, you may want to print out the export file to read the information.
 - b. Click **Export**.
8. Click **Save** in the File Download dialog and provide a name for the exported file. For example, FinanceClusterExport.zip.

Importing Clusters

You can import metadata about a Broker cluster and its dependent objects in the format of an XML file.

To import a cluster

1. In My webMethods: **Messaging > Broker Clusters > Clusters**.
2. On the Clusters page, click **Import**.
3. On the Cluster Import page, check to see whether the export file content is listed. If the file is not listed, do the following:
 - a. Click **Upload File**.
 - b. On the Import Cluster page, click **Browse** and select the .zip or the XML file to import.
 - c. Click **Upload**.
4. In the **Cluster Import** list, locate the cluster whose metadata you want to import.

If you want to select dependent objects to include or exclude, expand the node for the cluster and any dependent nodes, and then select the appropriate boxes.
5. Make sure the Target Objects are correct. To change the Target Object, click the corresponding Target Object link, and do one of the following on the Cluster Import Broker Target dialog:
 - If the Broker Server and Broker to which you are importing the metadata already exist, select the **Select Broker Name** tab and select a **Target Server Name** and **Target Broker Name** from their respective lists. Click **Apply**.
 - If the Broker to which you are importing the metadata is new, select the **New Broker Name** tab, select a **Target Server Name**, and type the name of the new **Target Broker Name**. Click **Apply**.

Then, on the Cluster Import page, reselect the objects to import.
6. Click **Import**.

Exporting Cluster Gateway Information

You can only save cluster gateway information when exporting from a cluster that contains cluster gateway configuration information.

To export cluster gateway information

1. In My webMethods: **Messaging > Broker Clusters > Clusters**.

2. In the **Clusters List**, click the cluster that includes the cluster gateway configuration information that you want to export. If the cluster is not in the list, use the **Search** tab to locate it.
3. On the **Clusters Details** page, click the **Export** tab.

My webMethods displays the cluster content in a tree view.

4. Click the plus (+) tab to expand the nodes for the cluster. Expand the **Gateways** node to verify that the cluster gateways whose metadata that you want to export is listed.
5. Select the check box for the cluster object and for the cluster gateways whose metadata you want to export.

Note: You must select the cluster object for the selected cluster gateway(s).

6. Click **Export**.
7. On the Export Cluster page, you can optionally **Include System Defined Data** in the export file.
8. Click **Export**.
9. Click **Save** in the File Download dialog and provide a name for the exported file.

Importing Cluster Gateway Information

Before you re-create a cluster gateway by importing the cluster gateway metadata, make sure you have completed the following:

1. Exported all the nodes of both the clusters connected by that cluster gateway from the source Broker Servers.
2. Imported all the nodes of both the clusters connected by that cluster gateway to the target Broker Servers.

Only after you have imported both the clusters of a cluster gateway, you can import and re-create a cluster gateway between the two clusters.

Use the following procedure to import cluster gateway information.

To import cluster gateway information

1. In My webMethods: **Messaging > Broker Clusters > Clusters**.
2. On the Clusters page, click **Import**.
3. On the Cluster Import page, check to see whether the export file content is listed. If the file is not listed, then do the following:
 - a. Click **Upload File**.
 - b. On the Import Cluster page, click **Browse** and select the .zip file or the XML file to import.

4. In the **Cluster Import** list, locate the cluster that includes the cluster gateway configuration information you want to import.

If you want to select dependent objects to include or exclude, expand the node for the cluster and select the check boxes for the objects to include.

5. Select the gateway and one of its dependent nodes, as well as the cluster object to which the gateway belongs.

Note: You must select the cluster object for the selected cluster gateway(s).

6. Make sure the Target Objects are correct. To change the Target Object, click the corresponding Target Object link, and do the following on the Cluster Import Gateway Target page:
 - a. Type the name of the **Remote Cluster Name** and the **Remote BrokerName**.
 - b. Select the **LocalBrokerName**.
 - c. Click **Apply**.
7. Click **Import**.

20 Exporting, Importing, and Deploying Assets

■ Overview	496
■ Exporting and Importing Broker Server and Broker Information	496
■ Asset Types and Dependencies	497
■ Exporting Broker Server and Broker Configuration to a File	498
■ Selecting JNDI Assets for Export	500
■ Importing Broker Server and Broker Configuration from a File	501
■ Exporting and Importing Territory Gateways	502
■ Deploying Broker Assets	503

Overview

This chapter describes:

- How to copy webMethods Broker configuration from one Broker to another Broker by using the **Export to File** and **Import from File** options available in the messaging user interface in My webMethods. See [“Exporting and Importing Broker Server and Broker Information”](#).
- How to deploy assets from one webMethods Broker to another webMethods Broker using webMethods Deployer. See [“Deploying Broker Assets”](#).

Exporting and Importing Broker Server and Broker Information

The export and import operations use ActiveWorks Definition Language (ADL) files for storage and retrieval of information. You can copy Broker Server and/or Broker information from a source Broker/Broker Server to a target Broker/Broker Server using the **Export to File** and **Import from File** options in My webMethods. In addition, you have the option of copying subsets of information. For example, you can copy only document types, or only client groups to an ADL file.

When you save the Broker Server and/or Broker information in an ADL file, you can store the ADL file in a source code control system, and move the file from a development environment to a production environment in a different network at a later date.

Use the Broker command line tools to process very large ADL files because the **Export to File** and **Import from File** functionality in My webMethods requires more memory. Using the Broker command line tools, you can import from the ADL file that was created using the **Export to File** option.

Important: webMethods Broker 9.0 and later export and import assets in ADL format, whereas webMethods Broker 8.2 SP2 or earlier versions exports assets in XML format. In webMethods Broker 9.0 and later, you cannot import the Broker assets that were exported in XML format from webMethods Broker 8.2 SP2 or earlier versions, but you can import the Broker assets that were exported in ADL format from webMethods Broker 8.2 SP3 or later versions.

For information about migrating the Broker assets stored in XML format to webMethods Broker, see the *Upgrading webMethods and Intelligent Business Operations Products* guide.

The **Export to File** and **Import from File** options enable you to export and import the following:

- Broker Server configuration.

Broker Server configuration information available for export/import includes:

- Broker Server description
- Logging configuration
- SSL configuration
- Access Control List

When you export Broker Server configuration data, you save the metadata about the Broker Server in an ADL file. For security reasons, export Broker Server configuration data with caution.

When you import a saved ADL file containing Broker Server information, make sure that the new Broker Server already exists. You cannot create a new Broker Server from an import file.

- Broker configuration

Broker configuration information available for export/import includes:

- Broker description
- Document type definitions
- Client group definitions
- Territory information
- Gateway information, including shared document types
- Access Control Lists for client groups, territories, and territory gateways

You can reconfigure the Brokers, client groups, clients, and document types from an export file. You can rename Brokers imported onto a different Broker Server or into a different territory; however, you cannot rename client groups, clients, and document types.

- JNDI assets such as JMS queues and JMS topics created by webMethods JNDI providers.

Asset Types and Dependencies

Some Broker assets have dependencies on other Broker components. When you export assets that have dependencies on other assets, if you select the **Include Dependencies** option, the dependent assets are also exported. For example, when you export a client group, the document types belonging to that client group are also exported.

The table lists the following:

- Asset types that you can export
- Dependencies of an asset type

Asset Type	Dependent On
Client	Client Group
Client Group	Document Type
Document Type	None
JMS Destination (JMS Queues and JMS Topics created by webMethods JNDI providers)	None

Exporting Broker Server and Broker Configuration to a File

Using the **Export to File** functionality in My webMethods, you can export:

- Only Broker Server information.
- Broker Server and Broker information.
- Only Broker information.

If you have configured SSL and ACLs on the Broker from which you want to export the configuration, make sure you have proper identity before you export.

To select the configuration data and export to a file

1. In My webMethods: **Administration > Messaging > Broker Servers > Servers**.
2. In the Broker Servers List, click the name of the Broker Server from which you want to export. If the Broker Server is not in the list, use the search functionality to locate it.
3. Click the **Brokers** tab.
4. In the Brokers List, click the Broker from which you want to export the assets. If the Broker does not appear in the list, use the search functionality to locate it.
5. On the Broker Details page, click the **Export to File** tab.

You see the name of the Broker Server and the Broker you selected for export.

6. Select the **Export Server Configuration** option if you want to export the metadata of the specified Broker Server. If you are using export and import only to replicate information about Brokers, client groups, clients, and document types, you do not need to save server configuration information. Use this option with caution for security reasons.

7. Select the **Export Broker Configuration** option if you want to export the configuration of the specified Broker.

8. Select the **Format** for saving the Broker Server and Broker configuration file.

- If you select **Platform Independent**, the export file includes unicode escape characters. You can use the configuration file to export Broker Server and Broker configuration to hosts that use different languages (and sometimes different types of host machine that use the same language). This is the default option.

Reading or editing configuration file might be difficult if the escape characters are present in the file. If your language uses an expanded character set, and you want to read or edit the Broker Server and Broker configuration file, you should save the configuration in native format.

- If you select **Native**, the export file does not include unicode escape characters. You can use the configuration file only on hosts of the same type and which use the same language and encoding.

In English, or other languages that do not use an extended character set, always use the platform-independent file format to save an export file.

9. Click **Proceed to Step 2**.

By default, all these Broker assets will be selected for export:

- **Export Document Types**
- **Export Client Groups**
- **Export Clients**
- **Include Dependencies**

Note that volatile clients and system defined document types are not exported, but if you have selected the **Include Dependencies** option, the dependent system defined document types are exported.

10. To select a subset of the document types, client groups, or clients for export, perform the following steps.

- a. Click the link mentioning the number of corresponding assets selected for export.
- b. If you have selected assets in multiple pages, click the **Selected** column title to sort by selected assets/not selected assets.
- c. Select or unselect the checkboxes corresponding to the assets for export.
- d. Use the **Select All/None** option to select or unselect all assets.
- e. Use the search functionality to search assets. Specify the asset name that you are looking for and click **Search**.
- f. Click **Submit Changes** to save the changes in the selection list.
- g. Click **Return to Step 2** to return to **What to Export** panel.

11. Use the **Include Dependencies** option to specify whether you want to add the dependencies of the assets selected for export to the export file.
12. Click **Export to File**. The assets selected for export and the corresponding dependencies (if you have specified **Include Dependencies**) will be stored in an ADL file.

The configuration is saved as an ADL file and compressed in a .zip file for downloading.

13. Download the configuration file by clicking **Click here to download ADL Zip-File**.

If you want to build the assets in the .zip source file and then use webMethods Deployer to deploy the Broker assets, see [“Building Assets for Deployment” on page 504](#).

Selecting JNDI Assets for Export

You select and export the JMS destinations (JMS queues and JMS topics) created by webMethods JNDI providers using My webMethods. For each JNDI context selected for export, Broker creates a separate XML file for storing the exported JNDI assets.

To select the JMS queues and JMS topics of a JNDI context for export

1. In My webMethods: **Administration > Messaging > Naming Directories > Providers**.
2. Click **JNDI Export**. The JNDI destinations belonging to the webMethods JNDI providers are displayed in the Providers > JNDI Export page.
3. If a JNDI destination does not appear in the list, use the search functionality to locate it. Export the JNDI destinations of one Provider at a time. The JNDI destinations list is refreshed each time you select a provider from the **Provider Name** list.
4. In the JNDI destinations list, select the JNDI destinations you want to export.
5. Click **Add To Export**. The selected JNDI destination is added to Export JNDI Assets list in the right panel.
6. Click **Add All To Export** if you want to select all the JNDI destinations listed for export.
7. If you do not want to export a JNDI destination listed for export, click **Remove** to remove that JNDI destination from the export list.
8. If you do not want to export any of the JNDI destinations listed for export, click **Remove All**. All the JNDI destinations selected for export will be cleared from the export summary. You must select the JNDI destinations again if you want to export later.
9. Click **Export JNDI Assets**.
 - a. Specify the filename for storing the exported asset.

- b. Specify the location to save the .xml file containing the exported asset and the corresponding dependencies. You have the following options:
- **Download to Browser** (streams the exported file to the browser). Use this option if My webMethods Server is part of a cluster. Otherwise, you might not know on which My webMethods Server instance the exported assets are stored.
 - **Export to My webMethods Server.** Use this option for large files. Downloading large files to a browser may take a long time.

The XML file of the exported JNDI context will be compressed in a .zip file and stored in the *My webMethods Server_directory\MWS\broker\acdl\user* directory. For information about how to build the assets using the XML source file, see [“Building Assets for Deployment” on page 504.](#)

Importing Broker Server and Broker Configuration from a File

Using the **Import to File** functionality in My webMethods, you can import Broker Server and Broker information from another Broker. You have the option of copying subsets of information while importing to a Broker.

To import the configuration data from a file to a target Broker

1. In My webMethods: **Administration > Messaging > Broker Servers > Servers.**
2. In the Broker Servers List, click the name of the Broker Server that is the target for import. If the Broker Server is not in the list, use the search functionality to locate it.
3. Click the **Brokers** tab.
4. In the Brokers List, click the Broker to which you want to import the configuration data. If the Broker does not appear in the list, use the search functionality to locate it.
5. On the Broker Details page, click the **Import from File** tab.
6. In the **Upload ADL File- Step 1** dialog, enter the path and name of the .adl file in the **Filename** field or click **Browse** to navigate to the .adl file or .zip file. Click **Upload**.

Use a zip file to compress the file if the import file size exceeds 20 MB.

7. Select one or more of the options in the **What to Import Step 2** dialog.

Select	To...
Overwrite Broker Server Configuration	Import and overwrite existing Broker Server SSL configuration and logging setup information.
Overwrite Broker Configuration	Import and overwrite the existing Broker configuration.
Make it the Default Broker	Make the selected Broker the default.

8. Click **Proceed to Step 3**.
9. Select one or more of the options in the **What to Import - Step 3** dialog.

Select	To...
Overwrite/create Document Types	Import (and overwrite) existing document types into the selected target Broker.
Overwrite/create Client Groups	Import client groups into the selected target Broker. This option merges the existing client groups or creates new ones.
Overwrite/create Clients	Import clients into the selected target Broker. This option overwrites the existing clients or creates new ones.

To select a subset of any of these options, click the corresponding selection number link. By default, all assets of each option are selected for import. Clear the check boxes of the assets you do not want to import, then click **Submit Changes**.

10. Click **Import from File**.

If the import file contains a new SSL configuration, you are prompted for the password for the certificate file. You need to stop and restart the Broker Server for the configuration to take effect.

When you import large files, edit the My webMethods Server JVM options in the *Software AG_directory\MWS\server\<instance>\bin\server.properties* file.

Exporting and Importing Territory Gateways

You can export territories and their gateways from one machine to another either by using My webMethods or by using the Broker command line utilities.

For information on how to export Brokers using My webMethods, see [“Exporting Broker Server and Broker Configuration to a File” on page 498](#). For information on how to export Brokers using command line utility, see [“broker_save” on page 558](#).

For information on how to import Brokers using My webMethods, see [“Importing Broker Server and Broker Configuration from a File” on page 501](#). For information on how to import Brokers using command line utility, see [“broker_load” on page 552](#).

To export two territories and their gateways from one machine to another

1. In the source machine, export Brokers one-by-one from both the territories.
Each Broker’s information will be stored in a separate .adl file.

2. In the target machine, do the following:
 - a. Create all the Brokers that were existing in the territories that you exported from the source machine. For information on how to create Brokers, see [“Creating a Broker” on page 140](#).
 - b. Import the Brokers of the first territory using the .adl files exported from the source machine.

You will get an exception because the Brokers of the second territory are not imported yet.
 - c. Import the Brokers of the second territory.

This step creates a one-way gateway between the second territory and the first territory.
 - d. Re-import the Brokers of the first territory.

This step creates the other gateway and completes the gateway-pair between the two territories.

Deploying Broker Assets

webMethods Broker enables you to export the assets created in a source Broker, build the assets for deployment, and then deploy these assets in another Broker or application using Deployer:

- Broker assets such as clients, client groups, and document types.
- JNDI assets such as JMS queues and JMS topics created by webMethods JNDI providers.

You cannot deploy territories, gateways, and ACLs on another Broker.

To deploy Broker assets and JNDI assets:

1. Understand the dependencies of assets. For more information, see [“Asset Types and Dependencies” on page 497](#).
2. Export the Broker assets for export. For more information, see [“Exporting Broker Server and Broker Configuration to a File” on page 498](#).
3. Select the JNDI assets for export. For more information, see [“Selecting JNDI Assets for Export” on page 500](#).
4. Build Broker assets and JNDI assets using the `build` ant target. For more information, see [“Building Assets for Deployment” on page 504](#).
5. Deploy Broker assets and JNDI assets to other webMethods Brokers using webMethods Deployer. For more information, see [“Deploying Assets Using webMethods Deployer” on page 504](#).

Building Assets for Deployment

The `build` ant target uses the source files specified in the `build.properties` file in the `Software AG_directory\common\AssetBuildEnvironment\master_build` directory, validates the specified assets and the corresponding dependencies, and then creates `.acdl` files of the exported assets.

If you want to create `.acdl` files from large source files, make sure you increase the Java heap size using the `jvmarg` variable in the `Software AG_directory\common\AssetBuildEnvironment\Broker\build.xml` file. For example, to build assets from a source file of size 600MB, set `jvmarg value="-Xmx640M"`.

To build assets

1. Specify the location of the source files of exported assets.
 - a. Extract the source files containing the exported assets (Broker assets are in `.adl` files and JNDI assets are in `.xml` files) from the `.zip` file in the `My webMethods Server_directory\MWS\broker\acdl\user` directory to a separate directory.
 - b. In the `Software AG_directory\common\AssetBuildEnvironment\master_build\build.properties` file, specify the location where you have saved the source files.

For information on how to select Broker assets for export, see [“Exporting Broker Server and Broker Configuration to a File”](#).

For information on how to select JNDI assets for export, see [“Selecting JNDI Assets for Export”](#).

2. Make sure you have specified the properties in the `Software AG_directory\common\AssetBuildEnvironment\master_build\build.properties` file. For more information about the properties, see the *webMethods Deployer User’s Guide*.
3. Execute the `ant build` command from the `Software AG_directory\common\AssetBuildEnvironment\master_build` directory.

This command will store the `.acdl` file of the assets in the directory specified by the `build.output.dir` property. The `.acdl` file contains the exported assets and the dependencies. This metadata schema is defined by Deployer and is common across the webMethods product suite. For more information about building assets, see the *webMethods Deployer User’s Guide*.

Deploying Assets Using webMethods Deployer

After you export the assets from a source webMethods Broker and build the assets for deployment, use webMethods Deployer to deploy the Broker assets and JNDI assets from the source webMethods servers or development environments to target webMethods servers.

webMethods Deployer enables you to perform:

- Full deployment
- Partial deployment
- Checkpoint and rollback

For information about deploying assets, see the *webMethods Deployer User's Guide*.

Partial Deployment

Broker supports partial deployment of Broker assets and JNDI assets. When you perform partial deployment in Deployer, only the assets selected from the list of exported assets (available in .acdl files) will be deployed.

Note: Export JMS destinations into one JNDI context at a time for partial deployment. Partial deployment of JMS destinations into multiple JNDI contexts is not supported.

For information about partial deployment of assets, see the *webMethods Deployer User's Guide*.

Checkpoint and Roll Back Assets

Deployer's checkpoint and roll back feature enables successful deployment. The checkpointed assets and the delivered assets are by default stored in the *Integration Server_directory/instances/instance_name/replicate/deployer/Broker* directory. When the size of the checkpointed or delivered assets increase, you might want to store the assets in another directory.

To configure the storage path for the checkpointed assets and the delivered assets

1. Open the *Integration Server_directory/instances/instance_name/packages/WmBrokerDeployer/config/wmBrokerDeployer.properties* file for edit.
2. Specify the asset storage path in the *persistenceManager.path* property.
 - Use slash "/" as the path separator.
 - If you specify a relative path, the specified directory is created under the *Integration Server_directory/instances/instance_name* directory.

If you specify *persistenceManager.path=Broker/Assets*, the assets are stored in the *Integration Server_directory/instances/instance_name/Broker/Assets* directory. If you specify *persistenceManager.path=C:/Broker/Assets*, the assets are stored in *C:/Broker/Assets*.

21 Using Command Central to Manage Broker

■ About webMethods Broker Administration	508
■ Configuring Broker Server License	508
■ Configuring SSL in Broker Server	509
■ Retrieving Configuration Details of Broker Server Base Port	510
■ Pausing and Resuming Message Publishing in Broker Servers	511
■ Using the Administration Link of Broker Server	512
■ Administering Broker Server Using My webMethods	513
■ webMethods Broker and the Command Line Interface	514
■ Monitoring webMethods Broker KPIs	519

About webMethods Broker Administration

You can administer Broker Servers through Command Central. Note that because Platform Manager uses Broker Monitor to obtain information about Broker Servers, Broker Monitor must be running if you want to administer Broker Servers through Command Central.

You can use Command Central to perform the following operations on webMethods Broker.

- View the number of Broker Servers running in each environment of your IT landscape
- View the versions of Broker Servers
- View the fixes applied to Broker Servers
- Configure Broker Server license
- Configure SSL in a Broker Server
- Retrieve Broker Server base port and SSL configuration details
- Start, stop, and restart Broker Server
- Pause and resume message publishing in Broker Server
- Monitor Broker Server installations
- Monitor run-time status, KPIs, and alerts of Broker Server instances
- Use the administration link of Broker Server

Note: webMethods Broker does not support **Debug** and **Safe mode** lifecycle operations.

Configuring Broker Server License

To change Broker Server license

1. In the Environments pane, select the environment in which Broker Server is installed.
2. Click the **Instances** tab.
3. Click the Broker Server instance for which you want to change the license.
4. Click the **Configuration** tab.
5. Select **Licenses** from the drop-down

The license type, status, and expiration date for the Broker Server license appear below the drop-down

- In the **License Type** column, click the Broker Server link.

Command Central displays the license key location. You can view the license file details when you expand **License Key Details**.

- Click **Edit**.
- Click **Browse** in the **License Upload Location** field, and then navigate to the new license file.

The new license file that you select is uploaded to the license location as shown in the **Server License Location** field.

- If you want to change the licence file location, edit the new path in the **Server License Location** field. The new location of the server license file is updated in the `awbroker.cfg` configuration file that resides in Broker Server's data directory.
- Click **Save** to save the new license.

Configuring SSL in Broker Server

To enable or disable SSL in Broker Server

- In the Environments pane, select the environment in which you want to configure the Broker Server.
- Click the **Instances** tab.
- Click the name of the Broker Server instance for which you want to configure SSL.
- Click the **Configuration** tab.
- Select **Ports** in the drop-down list to view the port settings configured.

Command Central displays the Broker Server port.

- Click the name of the port and click **Edit**.

Connection Basics displays the following non-editable fields.

Field	Description
Enabled	Whether the Broker Server base port is enabled.
Port Number	The Broker Server base port number. To change the base port, stop the Broker Server and change the port setting using the <code>server_config</code> command line utility in webMethods Broker.

Field	Description
	For information about the <code>server_config</code> command line utility in webMethods Broker, see “ webMethods Broker Command-Line Utilities ” on page 525.

- Expand **Security Configuration** and specify the following SSL settings.

Field	Specify
SSL Enabled	Whether SSL port is enabled. Click Yes to enable the SSL port settings. If you do not want to use SSL, click No .
Keystore Type	The keystore type. Select the keystore type. <ul style="list-style-type: none"> ■ PEM ■ PKCS12
Server Location of Keystore	The directory where the keystore file is located.
Password	The password to open the keystore file.
Truststore Type	The truststore file format. Select the truststore type. <ul style="list-style-type: none"> ■ PEM ■ DIR
Server Location of Truststore	The directory where the truststore file is located.

- Click **Test** to verify the port settings.
- Click **Save** to save the port changes.

Retrieving Configuration Details of Broker Server Base Port

Using Command Central, you can retrieve the configuration details of Broker Server’s base port.

Note: You cannot use Command Central to configure the Broker Server base port. If you want to configure the base port assigned to a Broker Server, stop the Broker Server and change the port setting using the `server_config` command line utility.

For information about `server_config` command line utility, see “[webMethods Broker Command-Line Utilities](#)” on page 525.

Retrieving a Broker Server’s base port configuration details

1. In the Environments pane, select the environment in which Broker Server is installed.
2. Click the **Instances** tab.
3. Click the Broker Server instance for which you want to view the port settings.
4. Click the **Configuration** tab.
5. Select **Ports** from the drop-down list to view the following read-only Broker Server port details:

Field	Description
Enabled	Indicates whether the Broker Server port is enabled or disabled.
Port	Indicates the base port of the Broker Server.
Protocol	Indicates the protocol used by the Broker Server.
Type	Indicates the type of Broker Server port.

Pausing and Resuming Message Publishing in Broker Servers

When the publishing load increases in a Broker Server, you can pause publishing, clear queues, and later resume the publishing.

To pause and resume message publishing in a Broker Server

1. In the Environments pane, select the environment in which Broker Server is installed.
2. Select the **Instances** tab.
3. Click the status icon corresponding to the Broker Server and select the required lifecycle operation:

- Click **Pause** to pause message publishing in all the Brokers belonging to the selected Broker Server. The status of the Broker Server changes to **Paused**. Use the  icon to refresh the status immediately. You can continue to perform administrative tasks on paused Brokers. The clients of a paused Broker can access and retrieve the messages from the Broker queue.
- Click **Resume** to resume message publishing in all the paused Brokers belonging to the Broker Server.

The status of the Broker Server changes to **Online**. Use the  icon to refresh the status immediately.

Using the Administration Link of Broker Server

When you have the administrative credentials to access the administration link of Broker Server in Command Central, you can use the Broker Server Details page in My webMethods.

By default, My webMethods Server running on `localhost:8585` is available for you when you click the **Broker Server Details** link. If you want to use My webMethods Server running on a different host machine, configure the host and port of the My webMethods Server you want to use.

Configuring the Host and Port of My webMethods Server

The default host and port of the My webMethods Server specified for Broker Server administration is `localhost:8585`.

Use Command Central command line interface to configure the host and port of My webMethods Server.

Pre-requisites for Viewing the Broker Server Details Page in My webMethods

You can access the Broker Server Details page in My webMethods only if the following conditions are true for the corresponding installation:

- My webMethods Server is installed.
- webMethods Broker user interface in My webMethods is installed.
- My webMethods Server is running.
- You have administrative credentials to access the Broker Server Details page in My webMethods.
- The Broker Server that you want to administer is added in My webMethods.

Viewing the Broker Server Details Page in My webMethods

To view the Broker Server Details page in My webMethods

1. In the Environments pane, select the environment in which the Broker Server you want to administer is installed.
2. Select the **Environments > Instances** tab.
3. Click the name of the Broker Server you want to administer.
4. In the **Overview** tab, click **Broker Server Details**.

Administering Broker Server Using My webMethods

About Administering Broker Server Using My webMethods

Through Command Central you can access and use the webMethods Broker user interface in My webMethods for administering a Broker Server.

By default, the webMethods Broker user interface of My webMethods running on `localhost:8585` is available for you when you click the **Broker Server Details** link. If you want to use the webMethods Broker user interface of a My webMethods Server running on a different host machine, configure the host and port of the My webMethods Server you want to use.

Configuring the Host and Port of My webMethods Server for webMethods Broker User Interface

The default host and port of the My webMethods Server specified for the webMethods Broker user interface is `localhost:8585`.

Use Command Central command line interface to configure the host and port of My webMethods Server. For more information, see *webMethods Command Central and webMethods Platform Manager Command Reference*.

Pre-requisites for Viewing the webMethods Broker User Interface in My webMethods

The webMethods Broker user interface in My webMethods can be accessed through Command Central only if the following conditions are true for the corresponding installation:

- My webMethods Server is installed.
- webMethods Broker user interface in My webMethods is installed.

- My webMethods Server is running.
- The Broker Server that you want to administer is added in My webMethods. For information about how to add a Broker Server in My webMethods, see *Administering webMethods Broker*.

Viewing the webMethods Broker User Interface in My webMethods

Before you click **Broker ServerDetails** to view the webMethods Broker user interface in My webMethods.

Ensure that the conditions listed in [“Pre-requisites for Viewing the webMethods Broker User Interface in My webMethods”](#) are satisfied.

To view the webMethods Broker user interface in My webMethods

1. In the Environments pane, select the environment in which the Broker Server you want to administer is installed.
2. Select the **Environments > Instances** tab.
The Instances table lists the name and the status of each instance that is part of the selected environment.
3. Click the name of the Broker Server you want to administer.
4. In the **Overview** tab, click **Broker ServerDetails**.
Command Central displays the Broker Server Details page in My webMethods in a separate window.

webMethods Broker and the Command Line Interface

Commands that webMethods Broker Supports

webMethods Broker supports the Platform Manager commands listed in the following table. The table lists where you can find general information about each command. Additionally, if there is webMethods Broker-specific information, the table lists where you can learn more about arguments and options that webMethods Broker supports or details about the actions Broker takes when you execute an `exec` command.

List of commands supported by Broker:

- `sagcc get configuration data`
- `sagcc update configuration data`
- `sagcc get configuration instances`
- `sagcc list configuration instances`
- `sagcc get configuration types`

- `sagcc list configuration types`
- `sagcc get inventory components`
- `sagcc list inventory components`
- `sagcc exec lifecycle`
- `sagcc get monitoring`

Configuration Types that webMethods Broker Supports

The following table lists the configuration types that webMethods Broker supports.

Configuration Type	Use to configure...
BROKER-MWSADMIN	<p>The protocol, host, and port of the My webMethods Server that hosts the administration user interface for the Broker for which the command was executed. The default value is <code>http://localhost:8585</code>.</p> <p>You can use the following commands to retrieve or update the value:</p> <ul style="list-style-type: none"> ■ <code>cc get configuration data</code> ■ <code>cc update configuration data</code>
COMMON-ADMINUI	<p>The full URL for the Broker Server Details page in My webMethods. You cannot change the value. You can use the <code>cc get configuration data</code> command to retrieve its value.</p>
COMMON-LICENSE	<p>The SagLic license file. You can use the following commands to add and update webMethods Broker-specific information to the SagLic file:</p> <ul style="list-style-type: none"> ■ <code>cc get configuration data</code> ■ <code>cc update configuration data</code> ■ <code>cc get configuration instances</code> ■ <code>cc list configuration instances</code> ■ <code>cc get configuration types</code> ■ <code>cc list configuration types</code>
COMMON-LICLOC	<p>The location where the license file resides in file system where Broker Server is installed.</p>
COMMON-PORTS	<p>The Broker Server listener port. You can use the following commands to retrieve or update port information:</p>

Configuration Type	Use to configure...
	<ul style="list-style-type: none"> ■ <code>cc get configuration data</code> ■ <code>cc update configuration data</code>
	<p>Note: You cannot use the command line interface to change the Broker Server port number. You can only change the SSL information.</p>

Example When Executing on Command Central

To change the URL of the My webMethods Server that hosts the Broker Server administration user interface to `http://localhost:8500`, do the following:

1. Read the current configuration of BROKER-MWSADMIN of the Broker-Server-8349 instance and store the configuration details in the BROKER-MWSADMIN.txt file:

```
sagcc get configuration data node_alias Broker-Server-8349
BROKER-MWSADMIN -p password -o BROKER-MWSADMIN.txt
```

2. Using a text editor, edit the BROKER-MWSADMIN.txt file to change the URL to `http://localhost:8500`.

3. Update BROKER-MWSADMIN using the new settings in the BROKER-MWSADMIN.txt file:

```
sagcc update configuration data node_alias Broker-Server-8349
BROKER-MWSADMIN -p password -i BROKER-MWSADMIN.txt
```

4. View the My webMethods Server URL change in the BROKER-MWSADMIN configuration:

```
sagcc get configuration data node_alias Broker-Server-8349
BROKER-MWSADMIN -p password -o BROKER-MWSADMIN.txt
```

5. Refresh the COMMON-ADMINUI configuration:

```
sagcc get configuration data node_alias Broker-Server-8349
COMMON-ADMINUI refresh=true -p password
```

Lifecycle Actions for Broker Server

The following table lists the actions that webMethods Broker supports with the `sagcc exec lifecycle` command and the operation taken against Broker Server when an action is executed.

Action	Description
<code>start</code>	Starts Broker Server. When successful, the Broker Server runtime status is set to ONLINE.

Action	Description
stop	Stops Broker Server. The Broker Server run-time status is STOPPED.
restart	Stops, then restarts Broker Server. The Broker Server run-time status is set to ONLINE.
pause	<p>Pauses operation on Broker Server. The Broker Server run-time status is set to PAUSED.</p> <p>When a Broker Server is paused, all Brokers that belong to the Broker Server stop publishing documents. However, Broker clients can still retrieve documents from the client queues, and you can still perform administrative tasks, such as creating clients and document types on the paused Brokers.</p>
resume	Resumes a previously paused Broker Server. As a result, Brokers that belong to the Broker Server start publishing documents again. The Broker Server run-time status is returns to ONLINE.

Monitoring Run-time Statuses for webMethods Broker

The following table lists the run-time statuses that webMethods Broker can return in response to the `sagcc get monitoring runtimestatus` and `sagcc get monitoring state` commands, along with the meaning of each run-time status.

Run-time Status	Meaning
ONLINE	Broker Server is running.
FAILED	Broker Server is not running due to some failure, and attempts to start it again have failed.
PAUSED	All Brokers belonging to the Broker Server have paused document publishing.
STARTING	Broker Server is starting.
STOPPED	Broker Server is not running because it was shut down normally.

Run-time Status	Meaning
STOPPING	Broker Server is stopping.
UNKNOWN	The status of Broker Server cannot be determined.
UNRESPONSIVE	Broker Server does not respond to a ping operation. However, other indicators, such as the existence of the PID or LOCK file indicate that Broker Server is running.

Monitoring Run-time States for webMethods Broker

In response to the `sagcc get monitoring rntimestate` and `sagcc get monitoring state` commands, webMethods Broker provides information about the following key performance indicators (KPIs):

KPI	Description
Data storage or configuration storage usage	<p>Use this KPI to monitor the Broker Server run-time memory storage or configuration storage so that you can take corrective actions if storage use approaches a critical value.</p> <p>The marginal, critical, and maximum values for this KPI depend on the maximum storage size configured for Broker Server.</p> <ul style="list-style-type: none"> ■ Marginal is 60% of the maximum storage size. ■ Critical is 80% of the maximum storage size. ■ Maximum is the configured storage size.
Memory usage	<p>Use this KPI to monitor use of Broker Server main memory so that you can take corrective actions if memory use approaches a critical value.</p> <p>The marginal, critical, and maximum values for this KPI depend the amount of main memory configured in the Broker Server configuration file (<code>awbroker.cfg</code>).</p> <ul style="list-style-type: none"> ■ Marginal is 80% of the maximum main memory. ■ Critical is 95% of the maximum main memory. ■ Maximum is the configured amount of main memory.

KPI	Description
Stalled queues	<p>Use this KPI to monitor stalled queues. A queue is considered stalled only if all the following conditions are true for that queue:</p> <ul style="list-style-type: none"> ■ A client is connected to the queue. ■ The queue contains at least one message. ■ More than five minutes has elapsed since the last time the client retrieved a message from the queue. <p>The KPI uses the following marginal, critical, and maximum values:</p> <ul style="list-style-type: none"> ■ Marginal is 1 stalled queue. ■ Critical is 50% of the maximum number of queues. ■ Maximum is determined by the greatest value among the following conditions: <ul style="list-style-type: none"> ■ 1 queue ■ 5% of the total number of client or forward queues in all Brokers. ■ Current number of stalled queues.

For more information about the webMethods Broker KPIs, see information about monitoring webMethods Broker in the [“Monitoring webMethods Broker KPIs”](#).

Monitoring webMethods Broker KPIs

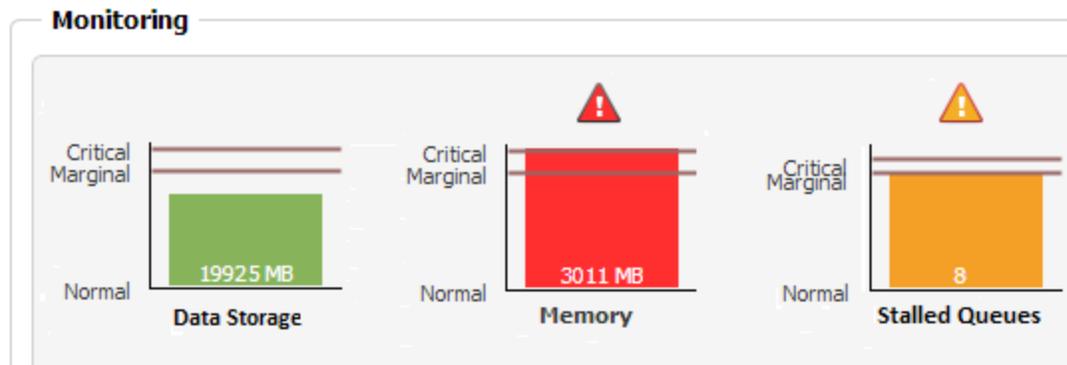
Overview

The visual key performance indicators (KPIs) and alerts enable you to monitor webMethods Broker’s health.

The following KPIs help you administer, troubleshoot, and resolve performance issues in webMethods Broker:

KPI	Description
Data Storage or Configuration Storage	Indicates the utilization of either the run-time data storage or the configuration data storage of Broker Server.

KPI	Description
Memory	Indicates the utilization of Broker Server memory.
Stalled queues	Indicates the performance of the message queues.



Storage Utilization KPI

Broker Server storage utilization indicator helps you to take corrective actions when either the run-time data storage or the configuration data storage of Broker Server reaches a critical value.

Marginal, Critical, and Maximum Values for Broker Server's Storage Utilization

The marginal, critical, and maximum values of run-time data storage and configuration data storage of Broker Server depend on the maximum storage size that you have configured for the Broker Server by using the `server_config` command.

Marginal Value	Critical Value	Maximum Value
60% of the Broker Server's maximum storage size.	80% of the Broker Server's maximum storage size.	Broker Server's maximum storage size.

Storage Utilization Display

The values of run-time data storage and configuration data storage define whether the storage indicator indicates the utilization of data storage or configuration data storage.

The storage indicator displays **Data Storage** if any of these conditions are true:

- The threshold category of both Broker Server data and configuration data storage are the same. That is, both the storage values are:

- Less than marginal (green ■)
- Greater than marginal but less than critical (yellow ■)
- More than critical and less than maximum (red ■)
- Broker Server data storage has reached a higher threshold compared to configuration data storage. For example, when Broker Server data storage is at the critical threshold (yellow ■) and configuration data storage is less than marginal (green ■), then storage indicator displays the data storage value.

The storage indicator displays **Configuration Storage** if the configuration data storage has reached a higher threshold compared to the Broker Server data storage. For example, when configuration data storage is at the maximum threshold (red ■) and Broker Server data storage is more than the marginal but less than critical threshold (yellow ■), then storage indicator displays the configuration data storage value.

<u>Storage Utilization Broker Server Data</u>	<u>Storage Utilization Configuration Data</u>	<u>Storage Value Displayed</u>
Less than marginal value	Less than marginal value	Broker Server Data
Less than marginal value	More than marginal value, but less than critical value	Configuration Data
Less than marginal value	More than critical value, but less than maximum value	Configuration Data
More than marginal value, but less than critical value	Less than marginal value	Broker Server Data
More than marginal value, but less than critical value	More than marginal value, but less than critical value	Broker Server Data
More than marginal value, but less than critical value	More than critical value, but less than maximum value	Configuration Data
More than critical value, but less than maximum value	Less than marginal value	Broker Server Data

<u>Storage Utilization Broker Server Data</u>	<u>Storage Utilization Configuration Data</u>	<u>Storage Value Displayed</u>
more than critical value, but less than maximum value	more than marginal value, but less than critical value	Broker Server Data
more than critical value, but less than maximum value	more than critical value, but less than maximum value	Broker Server Data

Memory Utilization KPI

The memory utilization indicator helps you monitor Broker Server's memory.

Marginal, Critical, and Maximum Values for Memory Utilization

The marginal, critical, and maximum values of memory utilization depend on the Broker Server's memory defined by the `max-memory-size` parameter in the Broker Server configuration file (`awbroker.cfg`).

<u>Marginal Value</u>	<u>Critical Value</u>	<u>Maximum Value</u>
80% of the <code>max-memory-size</code> parameter value.	95% of the <code>max-memory-size</code> parameter value.	Broker Server's memory limit defined in the <code>max-memory-size</code> parameter.

Stalled Queues KPI

The stalled queues indicator alerts you if messages are stuck for a long time or if messages are never retrieved from queues that are connected to clients.

A queue is considered to be stalled only if all these conditions are true:

- A client is connected to the queue
- The queue contains at least one message
- It has been more than five minutes since the client retrieved a message from the queue

<u>Marginal Value</u>	<u>Critical Value</u>	<u>Maximum Value</u>
1 queue.	50% of the maximum value.	Defined by whichever of these values is greater:

Marginal Value

Critical Value

Maximum Value

- 1 queue.
- 5% of the total number of client queues or forward queues in Brokers.
- Current number of stalled queues.

For example, if the number of stalled queues is zero, and 5% of the sum of client queues and forward queues is less than 1, then the maximum value is 1 queue.

A webMethods Broker Command-Line Utilities

■ Startup and Shutdown Utilities	526
■ Backup and Restore Utilities	526
■ Broker Server Configuration Utility	529
■ Broker Utilities	546
■ Directing Command-line Utilities to Use a Non-Default JRE or JDK	564
■ Specifying the JVM Options in Command-line Utilities	565

Startup and Shutdown Utilities

The `startup` and `shutdown` utilities help you start and stop all the Broker Servers monitored by Broker Monitor.

startup

Use the `startup` utility in the `webMethods Broker_directory/bin` directory to start Broker Monitor.

When you start a Broker Monitor, you will also start all the Broker Servers monitored by it. The `startup` script runs asynchronously, therefore the Broker Servers might take some time to start after executing the script.

Syntax

On Windows: `startup.bat`

On UNIX: `startup.sh`

shutdown

Use the `shutdown` utility in the `webMethods Broker_directory/bin` directory to stop Broker Monitor and all the Broker Servers monitored by it.

The `shutdown` script does not provide any warning message if there is no Broker Server running.

Syntax

On Windows: `shutdown.bat`

On UNIX: `shutdown.sh`

Backup and Restore Utilities

The following utilities are used to backup and restore the queue storage files associated with a Broker Server's configuration session. These utility programs reside in `webMethods Broker_directory/bin`.

Utility	Description
<code>server_conf_backup</code>	Creates a backup file containing the metadata for a Broker Server.

Utility	Description
<code>server_conf_restore</code>	Restores the metadata from a backup file created by <code>server_conf_backup</code> .

Important: The backup/restore feature only helps to recover corruption of the Broker Server's configuration session. It does *not* recover corrupted or uncorrupted Broker Server's run-time data storage. It is highly risky to restore the Broker Server's configuration data storage without removing its run-time data storage because of potential data inconsistencies between the two storage sessions. As a result, you must reinitialize the existing run-time data storage files during the recovery process.

For information about using the backup and restore utilities, see [“Backing Up and Restoring a Broker Server”](#) on page 121.

server_conf_backup

The `server_conf_backup` command creates a zip file that the `server_conf_restore` utility later uses to restore the metadata for a Broker Server. When the `server_conf_backup` command creates the backup zip file, it captures the absolute paths to the configuration data file(s).

This utility can be used while Broker Server is running.

Important: This utility can only be used with Broker Servers that use separate storage sessions. If you have a combined queue-storage session, you must use the file system's copy command as described on [“Backing Up a Combined Storage Configuration”](#) on page 129 or use the export and import feature to back up and restore the Broker Server's metadata. For more information about import and export, see [“Managing Client Groups”](#) on page 171.

For information about using this command, see [“Backing Up a Separate Storage Configuration”](#) on page 123.

Syntax

```
server_conf_backup backupFile [-h hostname:port] [-k keystore -p password
-t truststore] [-bu basic-auth-user -bp basic-auth-password] [-e] [-o] [-v]
[-b blockSize]
```

Argument	Description
<code>backupFile</code>	The fully qualified name of the file in which to store the backup data. <ul style="list-style-type: none"> ■ The utility automatically adds the <code>.zip</code> extension if you do not include it in the file name.

Argument	Description
	<ul style="list-style-type: none"> ■ The directory you specify must already exist. ■ If the file already exists, you must use the <code>-o</code> switch to overwrite it. Otherwise, the backup will be cancelled.
<code>-h hostname:port</code>	<p>The hostname and base port of the server you want to back up. The default is <code>localhost:6849</code>. Use this option to specify other Broker Servers on the local machine.</p> <p>Note: You cannot back up Broker Servers that reside on other hosts. This utility operates only on the local machine.</p>
<pre>-k keystore -p password -t truststore -bu basic-auth-user -bp basic-auth-password</pre>	<p>The authentication parameters needed to access the Broker Server if it is secured with an ACL.</p> <ul style="list-style-type: none"> ■ <code>keystore</code>. The name of the file containing the user SSL certificate. ■ <code>password</code>. The password for the keystore file. ■ <code>truststore</code>. The name of the file containing the trusted root of the user SSL certificate. ■ <code>basic-auth-user</code>. The user name for basic authentication. ■ <code>basic-auth-password</code>. The password for the basic authentication user.
<code>-e</code>	Turns on encryption to the Broker Server.
<code>-o</code>	If <code>backupFile</code> already exists, you must use this switch to overwrite it.
<code>-v</code>	Verbose mode. Prints out information about the progress and completion of the backup.

server_conf_restore

Use the `server_conf_restore` utility to restore the Broker Server metadata from a file that you created using `server_conf_backup`. To use this utility, you must first reinitialize the existing Broker Server's queue storage files. For procedures, see [“Restoring a Separate Storage Configuration” on page 125](#).

Syntax

```
server_conf_restore datadirbackupFile [-o] [-v] [-b blockSize]
```

Argument	Description
<i>datadir</i>	The location of the Broker Server's data directory.
<i>backupFile</i>	The name of the file where the backup is stored. Note: The backup files generated by the <code>server_conf_backup</code> command utility are zip files. Therefore, the <code>server_conf_restore</code> command will look for a zip file extension, even if you do not specify the ".zip" extension in your filename.
-o	When copying the contents of the backup zip files into the destination directory, the "-o" option instructs <code>server_conf_restore</code> to overwrite any existing configuration storage files with the backed up ones. If you do not specify "-o" then the <code>server_conf_restore</code> utility will prompt you for permission to overwrite any existing files.
-v	Verbose mode. Prints out information about the progress and completion of the backup.
-b <i>blockSize</i>	The size of the blocks that <code>server_conf_restore</code> will use when writing all of the configuration data storage files.

Example

```
server_conf_restore /var/opt/webMethods/awbrokersversion/default  
var/backups/server-6849.zip -o
```

Broker Server Configuration Utility

You use the `server_config` utility, which resides in `webMethods Broker_directory/bin`, to create and configure Broker Servers. This utility provides the following subcommands.

Syntax

```
server_config subcommand [options ...]
```

Subcommand	Description	See...
add	Add the executable associated with a Broker Server or create a new Broker Server that is a copy of an existing one.	“server_config add” on page 531
create	Create a new Broker Server.	“server_config create” on page 532
delete	Delete a Broker Server and all its data files.	“server_config delete” on page 538
help	Display help commands.	“server_config help” on page 538
list	List known Broker Servers on a given host machine.	“server_config list” on page 538
relocate	Configures the storage files in the new data directory.	“server_config relocate” on page 539
remove	Remove a Broker Server but retain all its data files.	“server_config remove” on page 540
start	Start a specified Broker Server.	“server_config start” on page 541
stop	Stop a specified Broker Server.	“server_config stop” on page 542
stopall	Stops Broker Monitor and all the Broker Servers monitored by it.	“server_config stopall” on page 542
storage	Configure storage sessions for a specified Broker Server.	“server_config storage” on page 543
update	Update an existing Broker Server.	“server_config update” on page 545

server_config add

The `add` subcommand has two uses:

- To control which Broker Server executable to use.
- To add a Broker Server by using or copying the configuration of an existing Broker Server. By specifying an existing configuration file, you can propagate Broker Server configurations among multiple platforms, add a previously configured Broker Server to an active configuration, or quickly upgrade an existing Broker Server deployment to a new release of webMethods Broker.

Syntax

```
add datadir [-e exec] [-k license-file-path] [-p port] [-S]
```

Or:

```
add datadir [-m awbroker-cfg] [-k license-file-path] [-p port] [-S]
```

Argument	Description
<i>datadir</i>	The path to the data directory for the Broker Server you are adding. Enclose the entire path in quotes if any portion of it contains a space.
<code>-e <i>exec</i></code>	The path to the <code>awbroker</code> executable file. This option allows you to run a Broker Server using an earlier release of webMethods Broker. The <code>-k</code> option (license file's absolute path) is required. Do not use in combination with the <code>-m</code> option. Enclose the entire path in quotes if any portion of it contains a space.
<code>-m <i>awbroker-cfg</i></code>	The path to the <code>awbroker.cfg</code> file to be used for the Broker Server to be added. A copy of the configuration file is placed in <i>datadir</i> . This option allows you to copy an existing Broker Server configuration. Do not use in combination with the <code>-e</code> option.
<code>-p <i>port</i></code>	The base port number to be used for the Broker Server to be added. This port number overrides any existing port number. This argument is needed if the default port 6849 is in use by another Broker Server.
<code>-k <i>license-file-path</i></code>	The Broker Server run-time license file. This license file overrides any existing license files.

Argument	Description
-S	Silent operation. No output is shown except for warnings and error messages.

The following command-argument format adds a server, possibly overriding executable, license file path, and port number for the new server. The server is created if it does not exist. This requires specifying a license file path.

```
add datadir [-e exec] [-k license-file-path] [-p port] [-S]
```

The following command-argument format creates a new server using an existing the configuration file as a template. If the configuration file (using -m awbroker-cfg option) is not specified, awbroker.cfg from the specified data director is used. The license file path and port number may be overridden.

```
add datadir [-m awbroker-cfg] [-k license-file-path] [-p port] [-S]
```

Example 1

The following example adds a new Broker Server (placing a configuration file in the new server directory) by copying the existing configuration file in the server2 directory and specifying a new port number.

```
server_config.exe add "C:\Program
Files\webmethodsversion\Broker\data\awbrokersversion\myserver"
-m "C:\Program
Files\webmethodsversion\Broker\data\awbrokersversion\server2\awbroker.cfg"
-p 6830
```

Example 2

The following example adds an existing Broker Server to the active configuration. The configuration file already exists in the old server directory.

```
server_config.exe add "C:\Program
Files\webmethods\Broker\data\awbrokersversion\oldserver"
```

server_config create

The `create` subcommand creates a new Broker Server in the specified data directory. When you create a new Broker Server, you can specify whether you want it to use separate or combined queue storage sessions. After the Broker Server is created, you cannot change its session type. The default is to create separate storage sessions.

For more information about creating Broker Servers, see [“Creating a Broker Server with the Default Log and Storage Files” on page 101](#).

Syntax for creating separate storage sessions

```
create datadir -k license-file-path [-e exec] [-p port] [-d desc] [-nostart] [-S]
[ -use_combined_storage ]
  [ -session_config sc-type
    [ -qs_log_file filename file-size
      { -qs_storage_file filename file-size [ reserved-size ] }* ]
```

```
[ -session_data sc-type
  [ -qs_log_file filename file-size
    { -qs_storage_file filename file-size [ reserved-size ]}* ]
```

Argument	Description
<i>datadir</i>	Fully qualified path to the data directory for the Broker Server being created. If the path includes spaces, enclose the spacing in double quotation marks. If the directory already exists, it must not contain a copy of the awbroker.cfg file.
<i>-e exec</i>	(optional) The path to the awbroker executable file. This option can be used to override the default location for the executable. By default, the executable is picked up from the <i>webMethods Broker_directory/bin</i> directory.
<i>-k license-file-path</i>	The absolute path to the Broker Server run-time license file. The location must have a valid license file in XML format as provided by Software AG.
<i>[-d desc]</i>	One-line description that describes the Broker Server you are creating. If the text string includes spaces, enclose the in double quotation marks. The description appears in various pages within the Broker user interface.
<i>[-p port]</i>	The Broker Server base port number. By default, the Broker Server uses port 6849.
<i>[-nostart]</i>	By default, the command starts the Broker Server automatically after it creates it. If you want to start the Broker Server manually later instead, use this argument to prevent the command from starting the Broker Server.
<i>[-S]</i>	By default, the command writes error messages to stderr and information messages to stdout. If you want to write error messages to stdout and suppress information messages instead, use this argument.
<i>[-use_combined_storage]</i>	Creates a combined storage session for both <i>session_config</i> and <i>session_data</i> .

Argument	Description
<i>sc-type</i>	Value of a session.
<pre data-bbox="256 409 535 472">[-qs_log_file filename fileSize]</pre>	<p data-bbox="657 409 1323 504"><i>filename</i> identifies the fully qualified path to the log file for the storage session. Enclose the entire path name in quotes if any portion of it contains a space.</p> <p data-bbox="657 525 1323 630"><i>fileSize</i> specifies the size of the log file. Follow the amount with k (kilobytes), m (megabytes), or g (gigabytes).</p> <p data-bbox="657 651 1323 714">Depending on which type of storage session you specify, the log file name changes. The default is:</p> <ul data-bbox="657 735 1323 903" style="list-style-type: none"> <li data-bbox="657 735 1323 808">■ For <code>-session_config</code>: <pre data-bbox="706 787 1356 808">-qs_log_file datadir/BrokerConfig.qs.log 32M</pre> <li data-bbox="657 829 1323 903">■ For <code>-session_data</code>: <pre data-bbox="706 882 1356 903">-qs_log_file datadir/BrokerData.qs.log 32M</pre> <p data-bbox="657 924 1323 1092">where the default log size is 32 MB. Anticipate a small delay in operation while the command initializes the files. For more information about log files for queue storage, see “Configuring Queue Storage” on page 84.</p>
<pre data-bbox="256 1144 535 1239">[-qs_storage_file filename fileSize [reservedSize]</pre>	<p data-bbox="657 1144 1323 1239"><i>filename</i> identifies the fully qualified path to the storage file. Each storage session can have multiple storage files.</p> <p data-bbox="657 1260 1323 1428"><i>fileSize</i> specifies the maximum space allowed for the storage file based on the storage session type specified, e.g., <code>session_config</code> or <code>session_data</code>. Change the value of <i>fileSize</i> to increase the storage size.</p> <p data-bbox="657 1449 1323 1585"><i>reservedSize</i> specifies the amount of storage that should be initially allocated. This value cannot be less than 16 MB. Once you set the <i>reservedSize</i>, you cannot decrease its size.</p> <p data-bbox="657 1606 1323 1669">Follow the size amounts with k (kilobytes), m (megabytes), or g (gigabytes).</p> <p data-bbox="657 1690 1323 1795">Depending on which type of storage session you specify, the storage file name changes. The default is:</p> <ul data-bbox="657 1816 1323 1913" style="list-style-type: none"> <li data-bbox="657 1816 1323 1913">■ For <code>-session_config</code>: <pre data-bbox="706 1869 1356 1913">-qs_storage_file datadir/BrokerConfig.qs.stor 512M 64M</pre>

Argument	Description
	<ul style="list-style-type: none"> ■ For <code>-session_data</code>: <pre>-qs_storage_file datadir/BrokerData.qs.stor 512M 64M</pre> ■ For <code>use_combined_storage</code>: <pre>-qs_storage_file datadir/Broker.qs.stor 512M 64M</pre> <p>where the default sizes are: a maximum storage size of 512 MB and a reserved storage size of 64 MB. Anticipate a small delay in operation while the command initializes the files.</p> <p>You cannot remove storage files or decrease their size.</p>

Example 1

The following example creates a new Broker Server files using port number 6840. This example uses the default storage parameters and creates a Broker Server with separate storage files.

```
server_config create "C:\SoftwareAG\Broker\data\awbrokersversion\server2"
-k C:\SoftwareAG\Broker\config\license.xml -p 6840
```

Example 2

The following example creates a new Broker Server that uses port number 7849 and separate storage sessions of specified sizes.

```
server_config create C:\Brokers\Server_7849 -p 7849
-k C:\SoftwareAG\Broker\config\license.xml
-desc "Broker for manufacturing"
-session_config qs
  -qs_log_file D:\BrokerLogs\Server_7849\BrokerConfig.qs.log 32M
  -qs_storage_file E:\BrokerStore\Server_7849\BrokerConfig.qs.stor 2G
-session_data qs
  -qs_log_file D:\BrokerLogs\Server_7849\BrokerData.qs.log 1G
  -qs_storage_file E:\BrokerStore\Server_7849\BrokerData.qs.stor 30G 5G
```

Syntax for creating combined storage sessions

```
server_config create datadir -k license-file-path
[-d desc] [-p port] [-nostart] [-S]
[-session_config qs [-qs_log_file filename fileSize]
[-qs_storage_file filename
fileSize [reservedSize]]
[-use_combined_storage]
```

Argument	Description
<code>datadir</code>	Fully qualified path to the data directory for the Broker Server being created. If the path includes

Argument	Description
	spaces, enclose the spacing in double quotation marks. If the directory already exists, it must not contain a copy of the awbroker.cfg file.
<code>-k license-file-path</code>	The absolute path to the Broker Server run-time license file. The location must have a valid license file in XML format as provided by Software AG.
<code>[-d desc]</code>	One-line description that describes the Broker Server you are creating. If the text string includes spaces, enclose the in double quotation marks. The description appears in various pages within the Broker user interface.
<code>[-p port]</code>	The Broker Server base port number. By default, the Broker Server uses port 6849.
<code>[-nostart]</code>	By default, the command starts the Broker Server automatically after it creates it. If you want to start the Broker Server manually later instead, use this argument to prevent the command from starting the Broker Server.
<code>[-S]</code>	By default, the command writes error messages to stderr and information messages to stdout. If you want to write error messages to stdout and suppress information messages instead, use this argument.
<code>[-qs_log_file filename fileSize]</code>	<ul style="list-style-type: none"> ■ <i>filename</i> identifies the fully qualified path to the log file for the storage session. Enclose the entire path name in quotes if any portion of it contains a space. ■ <i>fileSize</i> specifies the size of the log file. Follow the amount with k (kilobytes), m (megabytes), or g (gigabytes). <p>The default is:</p> <pre><code>-qs_log_file datadir/Broker.qs.log 32M</code></pre> <p>where the default log size is 32 MB. Anticipate a small delay in operation while the command initializes the files.</p>

Argument	Description
	You can remove or replace log files and you can increase or decrease their size.
<pre>[-qs_storage_file filename fileSize [reservedSize]</pre>	<ul style="list-style-type: none"> ■ <i>filename</i> identifies the fully qualified path to the storage file. The storage session can have multiple storage files. ■ <i>fileSize</i> specifies the maximum space allowed for the storage file based on the storage session type specified, for example, <i>session_config</i> or <i>session_data</i>. Change the value of <i>fileSize</i> to increase the storage size. ■ <i>reservedSize</i> specifies the amount of storage that should be initially allocated. This value cannot be less than 16 MB. Once you set the <i>reservedSize</i>, you cannot decrease its size. <p>Follow the size amounts with k (kilobytes), m (megabytes), or g (gigabytes).</p> <p>The default is:</p> <pre>-qs_storage_file datadir/Broker.qs.stor 512M 64M</pre> <p>where the default sizes are: a maximum storage size of 512 MB and a reserved storage size of 64 MB. Anticipate a small delay in operation while the command initializes the files.</p> <p>You cannot remove storage files or decrease their size.</p>
<pre>- use_combined_storage</pre>	Creates a combined storage session.

Example 1

The following example creates a new Broker Server that uses port number 6840 and a combined storage session of the default size.

```
server_config.exe create "C:\SoftwareAG\Broker\data\awbrokersversion\server2"
-p 6840 -k C:\SoftwareAG\Broker\config\license.xml -use_combined_storage
```

Example 2

The following example creates a new Broker Server that uses port number 7849 and a combined storage session of a specified size.

```
server_config create C:\Brokers\Server_7849 -p 7849
-k C:\SoftwareAG\Broker\config\license.xml -desc "Broker Server
for manufacturing"
-use_combined_storage
```

```
-session_config qs
  -qs_log_file C:\Brokers\Server_7849\Broker.qs.log 1G
  -qs_storage_file C:\Brokers\Server-7849\Broker.qs.stor 30G 5G
```

server_config delete

The `delete` subcommand removes the Broker Server configuration file, all of the data files associated with the Broker Server (and any other file(s) residing in the directory), and the data directory. Before you delete a Broker Server, make sure the Broker Server is not running.

Syntax

```
server_config delete datadir [-f] [-S]
```

Argument	Description
<i>datadir</i>	The data directory that belongs to the Broker Server you want to delete.
-f	Forced delete. The utility deletes the Broker Server without issuing a confirmation message.
-S	Silent operation. No output is shown except for warnings and error messages.

Example

```
server_config delete C:\SoftwareAG\Broker\data\awbrokersversion\server2 -f -S
```

server_config help

Lists all the subcommands for `server_config` and provides a brief explanation of each. If you need detailed information about a subcommand, use `server_config help` followed by the subcommand.

Syntax

```
server_config help subCommand
```

Example

```
server_config help add
```

server_config list

The `list` subcommand contacts the specified Broker Monitor on the specified host machine and provides a list of the known Broker Servers, their configurations, and current status. If the utility cannot contact the Broker Monitor, it provides a list of the

configurations of known Broker Servers from the Broker Server configuration file on the default Broker Monitor port. This is the only subcommand to `server_config` that you can use with a host other than the local host.

Syntax

```
server_config list [-h host] [-monitor_port port]
```

Argument	Description
<code>-h <i>host</i></code>	The name or IP address of the host machine on which the Broker Server is installed. If the <code>-h</code> argument is omitted, the utility lists the Broker Servers on <code>localhost</code> .
<code>-monitor_port <i>port</i></code>	The port number assigned to the Broker Monitor. If there are multiple webMethods Broker installations on the same host machine, use the <code>-monitor_port</code> argument to specify the installation from which you want to retrieve information. If the <code>-monitor_port</code> argument is omitted, the utility lists the Broker Servers on the default Broker Monitor port (6850).

Example

```
server_config list -h atlas -monitor_port 6851
```

server_config relocate

The `relocate` subcommand configures the Broker Server to use the new data directory paths for the storage files. If you want to use a storage file that is outside the data directory, use the `-qs_map_file` option to map the storage file.

The `relocate` subcommand does not support relocation of storage files belonging to a different operating system. If you want to migrate data from a different operating system, use the command-line utilities or My webMethods to export and import webMethods Broker metadata.

Syntax

```
server_config relocate datadir [-k license-file-path] [-b basic-auth-cfg] [-qs_map_file storage-file-old-path storage-file-new-path]*
```

Argument	Description
<code><i>datadir</i></code>	The absolute path to the new data directory.

Argument	Description
<code>-k license-file-path</code>	Optional. The absolute path to the Broker Server run-time license file. The location must have a valid license file in XML format as provided by Software AG. Use this option only if the license file is located outside the new data directory.
<code>-b basic-auth-cfg</code>	Optional. The absolute path to the new basic authentication configuration file (basicauth.cfg). Use this option only if the basic authentication configuration file (basicauth.cfg) is located outside the new data directory.
<code>-qs_map_file storage-file-old-path storage-file-new-path</code>	Optional. The absolute paths to the old storage file and the new storage file. Provide separate mapping for each storage file outside the new data directory. Use this option only if you want to use a storage file that is outside the new data directory.

Usage Notes

After you use the `relocate` subcommand, execute the `server_config add` command to add the new data directory to Broker Monitor's configuration file (awbrokermon.cfg).

Example

```
server_config relocate C:\SoftwareAG\Broker\data\awbrokers95\default
-k C:\SoftwareAG\Broker\config\license.xml
-b C:\SoftwareAG\Broker\config\basicauth.cfg-qs_map_file
D:\SoftwareAG\Broker90\Broker.qs.stor C:\SoftwareAG\Broker95\Broker.qs.stor
```

server_config remove

The `remove` subcommand removes the Broker Server from the configuration file but does not remove the data directory. Therefore you can add the Broker Server back to the configuration file at another time. When you execute `remove`, you are presented with configuration information for the Broker Server and prompted to continue. Before you remove the Broker Server, make sure it is not running.

Syntax

```
server_config remove datadir [-f] [-S]
```

Argument	Description
<i>datadir</i>	The data directory that belongs to the Broker Server that you want to remove.
<code>-f</code>	Forced delete. The utility removes the Broker Server without issuing a confirmation message.
<code>-S</code>	Silent operation. No output is shown except for warnings and error messages.

Example

```
server_config remove C:\SoftwareAG\Broker\data\awbrokersversion\server2 -f -S
```

server_config start

The `start` subcommand starts the specified Broker Server.

Syntax

```
server_config start -h host [-monitor_port port] [-broker_port port]
```

Argument	Description
<code>-h <i>host</i></code>	The host machine of the Broker Server that you want to start. If you do not specify a value, the IP address specified for the configured Broker Server in the Broker Monitor configuration file is assumed. If there is no IP address binding, the IP address of the localhost is used.
<code>-monitor_port <i>port</i></code>	The port number assigned to the Broker Monitor. If there are multiple webMethods Broker installations on the same host machine, use the <code>-monitor_port</code> argument to specify the installation that contains the Broker Server you want to start. If the <code>-monitor_port</code> argument is omitted, the utility lists the Broker Servers that are monitored on the default Broker Monitor port (6850).
<code>-broker_port <i>port</i></code>	The port number of the Broker Server that you want to start. If the <code>-broker_port</code> argument is omitted, the utility uses the default Broker Server port (6849).

Example

```
server_config start -h atlas -monitor_port 6851 -broker_port 6845
```

server_config stop

The `stop` subcommand stops the Broker Server.

Syntax

```
server_config stop -h host [-monitor_port port] [-broker_port port]
```

Argument	Description
<code>-h host</code>	The host machine of the Broker Server that you want to stop. If you do not specify a value, the IP address specified for the configured Broker Server in the Broker Monitor configuration file is assumed. If there is no IP address binding, the IP address of the localhost is used.
<code>-monitor_port port</code>	The port number assigned to the Broker Monitor. If there are multiple webMethods Broker installations on the same host machine, use the <code>-monitor_port</code> argument to specify the installation that contains the Broker Server you want to stop. If the <code>-monitor_port</code> argument is omitted, the utility uses the default Broker Monitor port (6850).
<code>-broker_port port</code>	The port number of the Broker Server that you want to stop. If the <code>-broker_port</code> argument is omitted, the utility uses the default Broker Server port (6849).

Example

```
server_config stop -h atlas -monitor_port 6851 -broker_port 6845
```

server_config stopall

The `stopall` subcommand stops the Broker Monitor and all the Broker Servers monitored by it.

Syntax

```
server_config stopall -h host [-monitor_port port]
```

Argument	Description
<code>-h host</code>	The host machine of the Broker Server that you want to stop. If you do not specify a value, the IP address specified for the configured Broker Server in the Broker Monitor configuration file is assumed. If there is no IP address binding, the IP address of the localhost is used.
<code>-monitor_port port</code>	The port number assigned to the Broker Monitor. If there are multiple webMethods Broker installations on the same host machine, use the <code>-monitor_port</code> argument to specify the installation that contains the Broker Server you want to stop. If the <code>-monitor_port</code> argument is omitted, the utility uses the default Broker Monitor port (6850).

Example

```
server_config stopall -h atlas -monitor_port 6851
```

server_config storage

The `storage` subcommand configures storage sessions for a specified Broker Server. You can use the `storage` subcommand to add storage files or modify the sizes of the log file or the storage files.

File Type	Max Number of Files	Default File Size	Maximum File Size
log file	1 per session	256 MB	8 GB
storage file	62 per session	512 MB	32 GB

Note: You must stop the Broker Server before configuring additional storage files.

Syntax

```
server_config storage datadir
[-session_config qs
[-qs_log_file filename fileSize]
[-qs_storage_file filename fileSize [reservedSize]]
[-qs_storage_file filename fileSize [reservedSize]]
...
[-session_data qs
[-qs_log_file filename fileSize]
[-qs_storage_file filename fileSize [reservedSize]]
[-qs_storage_file filename fileSize [reservedSize]]
...]
```

```
[-list]
```

Argument	Description
<i>datadir</i>	Fully qualified path to the data directory for the Broker Server whose queue storage you are configuring. If the path includes spaces, enclose the spacing in double quotation marks.
<pre>[-qs_log_file filename fileSize]</pre>	<p><i>filename</i> identifies the fully qualified path to the log file for the storage session. Enclose the entire path name in quotes if any portion of it contains a space.</p> <p><i>fileSize</i> specifies the size of the log file. Follow the amount with k (kilobytes), m (megabytes), or g (gigabytes). The default is.</p> <pre>-qs_log_file datadir/Broker.qs.log 32M</pre> <p>where the default log size is 32 MB. Anticipate a small delay in operation while the command initializes the files.</p> <p>You can remove or replace log files and you can increase or decrease their size.</p>
<pre>[- qs_storage_file filename fileSize [reservedSize]</pre>	<p><i>filename</i> identifies the fully qualified path to the storage file. The storage session can have multiple storage files.</p> <p><i>fileSize</i> specifies the maximum space allowed for the storage file based on the storage session type specified, e.g., <i>session_config</i> or <i>session_data</i>. Change the value of <i>fileSize</i> to increase the storage size.</p> <p><i>reservedSize</i> specifies the amount of storage that should be initially allocated. This value cannot be less than 16 MB. Once you set the <i>reservedSize</i>, you cannot decrease its size.</p> <p>Follow the size amounts with k (kilobytes), m (megabytes), or g (gigabytes). The default is:</p> <pre>-qs_storage_file datadir/Broker.qs.stor 512M 64M</pre> <p>where the default sizes are: a maximum storage size of 512 MB and a reserved storage size of 64 MB. Anticipate a small delay in operation while the command initializes the files.</p> <p>You cannot remove storage files or decrease their size.</p>

Example

The following example creates an additional run-time storage file for a Broker Server:

```
server_config storage C:\SoftwareAG\Broker\data\awbrokersversion\Server_7849
-session_data qs
  -qs_storage_file D:\MyBrokerStorage\Server_7849\BrokerData.qs.stor 30G 5G
```

server_config update

You use the `update` subcommand to modify the following operating parameters:

- Run-time license
- Broker Server description
- Base port number used by the Broker Server

Tip: On Windows, the port number is part of the service name. If you change the port number, the program attempts to change the service name, an action that may not succeed. Another strategy to update a port number on Windows is to use the `create` subcommand to create a new Broker Server, copy the data files from the old data directory (not including the `awbroker.cfg` file), and delete the old Broker Server using the `delete` subcommand.

Syntax

```
server_config update datadir [-k license-file-path] [-d description]
[-p port] [-S]
```

You must stop the Broker Server before running the `server_config` utility. For the changes to take effect, you must restart the Broker Server.

Argument	Description
<i>datadir</i>	The data directory for the Broker Server you are updating. Enclose the path name in quotes if any portion of it contains a space.
<code>-k <i>license-file-path</i></code>	The absolute path to the new license-key file.
<code>-d <i>description</i></code>	A new description of the Broker Server. This optional description appears on various pages in the Broker user interface. If the text includes spaces, enclose the entire description in quotation marks.

Argument	Description
<code>-p port</code>	The new base port to be used by the Broker. Stop the Broker before you attempt to change the port number.
<code>-s</code>	Silent operation. No output is shown except for warnings and error messages.

Example

The following example updates the configuration of a Broker to use a new run-time license and a new description.

```
server_config.exe update C:\SoftwareAG\Broker\data\awbrokersversion\server2
-k C:\SoftwareAG\Broker\config\license.xml
-desc "PRODUCTION--Broker Server for manufacturing"
```

Broker Utilities

Broker utilities are shown in the table below:

Command	Description	See...
<code>broker_buildall</code>	Compiles all intelligent integration components and scripted operations on a Broker.	“broker_buildall” on page 547
<code>broker_create</code>	Creates a new Broker.	“broker_create” on page 548
<code>broker_delete</code>	Deletes a Broker.	“broker_delete” on page 550
<code>broker_load</code>	Imports Broker data from a file to a Broker.	“broker_load” on page 552
<code>broker_ping</code>	Sends system ping documents through a Broker.	“broker_ping” on page 554
<code>broker_save</code>	Saves the configuration information of the specified Broker in a file.	“broker_save” on page 558

Command	Description	See...
<code>broker_start</code>	Starts the Broker Server.	“broker_start” on page 560
<code>broker_status</code>	Displays statistics from the command line for a specific Broker.	“broker_status” on page 561
<code>broker_stop</code>	Stops all Brokers running on the Broker Server, halts all document delivery, and disconnects all clients.	“broker_stop” on page 562

broker_buildall

Use the `broker_buildall` utility to compile all pre-webMethods 6.x intelligent integration components and scripted operations from a Broker.

When you run the `broker_buildall` utility, it compiles all components on the Broker that have the "Need to compile" status. If an error is encountered while compiling, `broker_buildall` writes a message to the event log and continues with the next component. You can recompile if necessary.

Syntax

```
broker_buildall [-force] [-output] [-h] [-?] [--] [broker@]server [:port]
[-idhelp] [id_options]
```

Argument	Description
<code>-h</code>	Displays a usage message.
<code>-?</code>	Displays usage help for Java command-line options.
<code>-force</code>	Causes the utility to bypass error checking. Forces a recompile for every Scripted Operation and Intelligent Integration Component regardless of their state.
<code>-output</code>	Outputs the standard output name of the component being compiled.
<code>--</code>	Allows the Broker name to start with the character <code>-</code> .

Argument	Description
<code>[broker@] server[:port]</code>	Identifies the name of the Broker Server (and optionally, the Broker and the port number) on which to load the Broker information. If you omit the Broker name, the default Broker is assumed. If you omit the Broker Server, only syntax checking is performed on the file.
<code>-idhelp</code>	Displays a usage message for the <code>[id_options]</code> .
<code>[id_options]</code>	<p>Provides identification needed for administrative access to the Broker if the Broker is protected by an Access Control List (ACL).</p> <p><code>-entrust</code> uses <code>entrust</code> as the SSL provider. Default is JSSE.</p> <p><code>-keystorefilename</code> specifies the name of the file containing the user SSL certificate.</p> <p><code>-passwordpassword</code> specifies the password for the keystore file.</p> <p><code>-truststorefilename</code> specifies the name of the file containing the trusted root of the user SSL certificate.</p> <p><code>-noencrypt</code> specifies not to use encryption for the connection. By default, every connection using a certificate is encrypted.</p> <p><code>-basicauthuserusername</code> specifies the user name for basic authentication. If you specify <code>-basicauthuser</code>, you must also specify <code>-basicauthpassword</code>.</p> <p><code>-basicauthpasswordpassword</code> specifies the password for the basic authentication user. If you specify <code>-basicauthpassword</code>, you must also specify <code>-basicauthuser</code>.</p>

broker_create

Use the `broker_create` utility to create a Broker on a Broker Server.

Syntax

```
broker_create -h [[--]broker[@server[:port]] [-default]
[-description text] [-createterr territory]
[-jointerr broker[@server[:port]]] [-createcluster cluster]
[-joincluster broker[@server[:port]]]
[-idhelp] [id_options]
```

```
[-timeout timeout]
```

Argument	Description
-h	Displays a usage message.
--	Allows the Broker name to start with the character -.
<i>broker</i> [@ <i>server</i> [: <i>port</i>]]	Specifies the name to be assigned to the Broker. Broker Server and port number are optional if the Broker Server is on the local host.
-default	Makes the Broker the default Broker.
-description <i>text</i>	Provides a short description of the Broker. This description is displayed on various pages in the Broker user interface.
-createterr <i>territory</i>	Creates a new territory and makes the new Broker the first member.
-jointerr <i>broker</i> [@ <i>server</i> [: <i>port</i>]]	Makes the new Broker a member of the territory that the specified Broker is a member of.
-createcluster <i>cluster</i>	Creates a new cluster and makes the new Broker the first member.
-joincluster <i>broker</i> [@ <i>server</i> [: <i>port</i>]]	Makes the new Broker a member of the cluster that the specified Broker is a member of.
-idhelp	Displays a usage message for the [<i>id_options</i>] listed below.
[<i>id_options</i>]	Provides identification needed for administrative access to the Broker if the Broker is protected by an Access Control List (ACL). -entrust uses entrust as the SSL provider. Default is JSSE.

Argument	Description
	<p>-keystore<i>filename</i> specifies the name of the file containing the user SSL certificate.</p> <p>-password<i>password</i> specifies the password for the keystore file.</p> <p>-truststore<i>filename</i> specifies the name of the file containing the trusted root of the user SSL certificate.</p> <p>-noencrypt specifies to not use encryption for the connection. By default, every connection using a certificate is encrypted.</p> <p>-basicauthuser<i>username</i> specifies the user name for basic authentication. If you specify -basicauthuser, you must also specify -basicauthpassword.</p> <p>-basicauthpassword<i>password</i> specifies the password for the basic authentication user. If you specify -basicauthpassword, you must also specify -basicauthuser.</p>
[-timeout <i>timeout</i>]	Specifies the timeout time in seconds for the API calls made by the <code>broker_create</code> utility. If any of these API calls to the Broker Server do not complete before the specified time, a time-out exception is thrown by the <code>broker_create</code> utility.

broker_delete

Use this utility to delete a Broker from a Broker Server and remove it from its territory, if it belongs to one. All client queues on the Broker are lost, all client queues are disconnected, and the Broker, all of its document types, and client groups are deleted permanently. By default, you are prompted to confirm this command.

Syntax

```
broker_delete [-h] [-y] [--] broker@server[:port] [-idhelp] [id_options]
[-timeout timeout]
```

Argument	Description
-h	Displays a usage message.
-y	Does not prompt for confirmation before deleting the Broker.
--	Allows the Broker name to start with the character -.
<i>broker@server[:port]</i>	Specifies the name of the Broker to be deleted and the Broker Server on which it resides. If you do not specify the port number, the default port is assumed.
-idhelp	Displays a usage message for the <i>[id_options]</i> listed below.
<i>[id_options]</i>	<p>Provides identification needed for administrative access to the Broker if the Broker is protected by an Access Control List (ACL).</p> <p>-entrust uses entrust as the SSL provider. Default is JSSE.</p> <p>-keystore<i>filename</i> specifies the name of the file containing the user SSL certificate.</p> <p>-password<i>password</i> specifies the password for the keystore file.</p> <p>-truststore<i>filename</i> specifies the name of the file containing the trusted root of the user SSL certificate.</p> <p>-noencrypt specifies to not use encryption for the connection. By default, every connection using a certificate is encrypted.</p> <p>-basicauthuser<i>username</i> specifies the user name for basic authentication. If you specify -basicauthuser, you must also specify -basicauthpassword.</p> <p>-basicauthpassword<i>password</i> specifies the password for the basic authentication user. If you specify -basicauthpassword, you must also specify -basicauthuser.</p>
<i>[-timeout timeout]</i>	Specifies the timeout time in seconds for the API calls made by the <code>broker_delete</code> utility. If any of these API calls to the Broker Server do not complete before the

Argument	Description
	specified time, a time-out exception is thrown by the <code>broker_delete</code> utility.

broker_load

Use the `broker_load` utility to import metadata from an ADL file.

If the import file contains a new SSL configuration, you may need to stop and restart the Broker Server for the configuration to take effect. In such cases, `broker_load` prompts for whether or not you want to stop and restart the Broker Server at that time. Also, if the import file does not contain the password for the keystore file, you are prompted for it.

The `broker_load` program divides large files into 2 MB parts. The parts are then imported sequentially to the Broker and reassembled. If an error occurs during this process, some document types may still be loaded. That is, the file may be partially loaded if an error occurs and the Broker is left in a partially updated state.

Syntax

```
broker_load [-h] input_file [-force] [-merge] [-write output_file]
[--] [broker@] server [:port] [-idhelp] [id_options] [-timeout timeout]
```

Argument	Description
<code>-h</code>	Displays a usage message.
<i>input_file</i>	Specifies the file you saved the Broker configuration information to using the <code>broker_save</code> command.
<code>-force</code>	Causes the utility to bypass error checking.
<code>-write <i>output_file</i></code>	Writes a copy of the definitions in the input file to the specified output file using the latest revision of the export file format. If no output file is specified, the only output is syntax errors.
<code>--</code>	Allows the Broker name to start with the character <code>-</code> .
[<i>broker@</i>] <i>server</i> [<i>:port</i>]	Specifies the name of the Broker Server (and, optionally, the Broker and the port number) on which to load the Broker information. If you omit the Broker name, the default Broker is assumed. If you omit the

Argument	Description
	Broker Server, only syntax checking is performed on the file.
-idhelp	Displays a usage message for the <code>[id_options]</code> listed below.
<code>[id_options]</code>	<p>Provides identification needed for administrative access to the Broker if the Broker is protected by an Access Control List (ACL).</p> <p>-entrust uses entrust as the SSL provider. Default is JSSE.</p> <p>-keystore<i>filename</i> specifies the name of the file containing the user SSL certificate.</p> <p>-password<i>password</i> specifies the password for the keystore file.</p> <p>-truststore<i>filename</i> specifies the name of the file containing the trusted root of the user SSL certificate.</p> <p>-noencrypt specifies to not use encryption for the connection. By default, every connection using a certificate is encrypted.</p> <p>-basicauthuser<i>username</i> specifies the user name for basic authentication. If you specify -basicauthuser, you must also specify -basicauthpassword.</p> <p>-basicauthpassword<i>password</i> specifies the password for the basic authentication user. If you specify -basicauthpassword, you must also specify -basicauthuser.</p>
-merge	<p>Merges non-system document types on import. To use -merge, the document types must have the same name.</p> <p>Note: <code>broker_load</code> returns an error message if the two documents have the same field names but different field types.</p>
<code>[-timeout timeout]</code>	Specifies the timeout time in seconds for the API calls made by the <code>broker_load</code> utility. If any of these API calls to the Broker Server do not complete before the specified time, a time-out exception is thrown by the <code>broker_load</code> utility.

broker_ping

Use `broker_ping` to send system ping documents through a Broker. If the document passes through the Broker Server and returns to `broker_ping`, a positive message is printed. By default, one document is sent. If no document returns, a negative message is printed.

Syntax

```
broker_ping [-h] [-s] [-c count] [-remote [/territory/]broker2]
[[-] [broker@]host[:port]] [-idhelp] [id_options] [-timeout timeout]
```

The `broker_ping` command uses the following arguments:

Argument	Description
-h	Displays a usage message.
-s	Sends a document through the Broker Server once every second.
-c count	Specifies the number of documents that are sent through the Broker.
-remote [/territory/]broker2	Pings a specified Broker in a specified territory.
--	Allows the Broker name to start with the character -.
[broker@]host[:port]	Specifies the name of the Broker Server (and, optionally, the Broker or port) you want to ping. If you omit the Broker name, the default Broker is assumed.
-idhelp	Displays a usage message for the <code>[id_options]</code> listed below.
[id_options]	Provides identification needed for administrative access to the Broker if the Broker is protected by an Access Control List (ACL). -entrust uses entrust as the SSL provider. Default is JSSE.

Argument	Description
	<p><code>-keystorefilename</code> specifies the name of the file containing the user SSL certificate.</p> <p><code>-passwordpassword</code> specifies the password for the keystore file.</p> <p><code>-truststorefilename</code> specifies the name of the file containing the trusted root of the user SSL certificate.</p> <p><code>-noencrypt</code> specifies to not use encryption for the connection. By default, every connection using a certificate is encrypted.</p> <p><code>-basicauthuserusername</code> specifies the user name for basic authentication. If you specify <code>-basicauthuser</code>, you must also specify <code>-basicauthpassword</code>.</p> <p><code>-basicauthpasswordpassword</code> specifies the password for the basic authentication user. If you specify <code>-basicauthpassword</code>, you must also specify <code>-basicauthuser</code>.</p>
<code>[-timeout timeout]</code>	Specifies the timeout time in seconds for the API calls made by the <code>broker_ping</code> utility. If any of these API calls to the Broker Server do not complete before the specified time, a time-out exception is thrown by the <code>broker_ping</code> utility.

Pinging a Remote Broker

You can use the `broker_ping` utility to ping a remote Broker in the same territory or in a territory that is connected by a territory gateway. The ping document passes through the local Broker to the remote Broker, allowing you to trace the connection between Brokers. For example, assume that you want to trace the connection between local Broker *Alpha* on the host *atlas* and remote Broker *Beta* in the same territory. The command is:

```
broker_ping -remote Beta Alpha@atlas
```

To ping the Broker *Gamma*, which is in the territory T-2, across a territory gateway, the command is:

```
broker_ping -remote /T-2/Gamma Alpha@atlas
```

To use `broker_ping` across a territory gateway, the document type `Broker::Ping` must be shared across the gateway.

broker_ping Return Codes

The return code of `broker_ping` specifies the status of the Broker pinged. `broker_ping` is successful if the return code is 0. If the return code is not 0, use the table below to understand why `broker_ping` was not successful.

Major Code	Main Reason	Possible Problem
0	<code>broker_ping</code> is successful	
2	Could not connect to the Broker	
100	Broker failure	Broker has memory or storage problem
101	Broker not running	Broker is not running on the specified port
102	Communication failure	<ul style="list-style-type: none"> ■ Transport initialization failed ■ Unable to find the specified host ■ Specified host not on the network ■ Unable to open the socket on the specified host ■ Invalid connection
103	Connection closed	
105	Specified host not found	
108	No memory	Memory allocation failed
109	No permission	
111	Protocol error	Arguments in the protocol missing or incorrect
112	Request timed out	
114	Security error	<ul style="list-style-type: none"> ■ Certificate expired

Major Code	Main Reason	Possible Problem
		<ul style="list-style-type: none"> ■ Distinguished name exists, but its certificate is not certified ■ Attempted connection to a non-secure port ■ Secure socket does not support the SSL version ■ Multiple certificates specified in the certificate file without specifying the default certificate
202	Client exists	
209	Invalid client	
210	Invalid client ID	
211	Invalid descriptor	
212	Invalid document	
213	Invalid document type name	
219	Invalid document type definition	
220	Null parameter exception	<ul style="list-style-type: none"> ■ Input parameter is null ■ Output parameter is null
221	Out of range exception	Input parameter value is out of range.
223	Unknown Broker name	
224	Unknown client group	
226	Unknown document type	
237	Invalid client group name	

Major Code	Main Reason	Possible Problem
240	Invalid Broker name	
272	Invalid transaction ID	
274	Invalid operation	<ul style="list-style-type: none"> ■ Queue is empty ■ 'from_index' value is out of bounds

broker_save

Use the `broker_save` utility to save Broker configuration information for a specified Broker to a file. Document types, client groups, and clients are always included in the saved file.

Syntax

```
broker_save [-h] [-broker] [-server] [-native] output_file
[ [--] [broker@]server[:port] ] [-idhelp] [id_options] [-timeout timeout ]
```

Argument	Description
-h	Displays a usage message.
-broker	Includes the Broker's configuration in the saved file. The configuration information includes: description, territory, and any gateways. The default is to exclude it from the file.
-server	Includes the Broker Server's SSL configuration and logging options in the save file. The default is to exclude them from the file.
-native	Writes Unicode characters using the native file format.
<i>output_file</i>	Absolute path of the file in which the information is saved.
--	Allows the Broker name to start with the character -.
[<i>broker@]server[:port]</i>	Specifies the name of the Broker Server (and, optionally, the Broker and port number) from which

Argument	Description
	to save the Broker information. If you omit the Broker name, the default Broker is assumed.
<code>-idhelp</code>	Displays a usage message for the <code>[id_options]</code> listed below.
<code>[id_options]</code>	<p>Provide identification needed for administrative access to the Broker if the Broker is protected by an Access Control List (ACL).</p> <p><code>-entrust</code> uses <code>entrust</code> as the SSL provider. Default is JSSE.</p> <p><code>-keystorefilename</code> specifies the name of the file containing the user SSL certificate.</p> <p><code>-passwordpassword</code> specifies the password for the keystore file.</p> <p><code>-truststorefilename</code> specifies the name of the file containing the trusted root of the user SSL certificate.</p> <p><code>-noencrypt</code> specifies to not use encryption for the connection. By default, every connection using a certificate is encrypted.</p> <p><code>-basicauthuserusername</code> specifies the user name for basic authentication. If you specify <code>-basicauthuser</code>, you must also specify <code>-basicauthpassword</code>.</p> <p><code>-basicauthpasswordpassword</code> specifies the password for the basic authentication user. If you specify <code>-basicauthpassword</code>, you must also specify <code>-basicauthuser</code>.</p>
<code>[-timeout timeout]</code>	Specifies the timeout time in seconds for the API calls made by the <code>broker_save</code> utility. If any of these API calls to the Broker Server do not complete before the specified time, a time-out exception is thrown by the <code>broker_save</code> utility.

Examples

In your configuration, there are Broker Servers Alpha (for Brokers named BrokerA and BrokerB) and Beta (for Brokers named BrokerC and BrokerD). To save a configuration file for each server and each Broker in the configuration, use:

For Broker Servers:

```
broker_save -server alpha.adl Alpha
```

```
broker_save -server beta.adl Beta
```

The above commands save the configuration of Alpha Broker Server in alpha.adl file, and the configuration of Beta Broker Server in beta.adl file.

For saving Brokers:

```
broker_save -broker A.adl BrokerA@Alpha
broker_save -broker B.adl BrokerB@Alpha
broker_save -broker C.adl BrokerC@Beta
broker_save -broker D.adl BrokerD@Beta
```

The above commands save the configuration of BrokerA, BrokerB, BrokerC, and BrokerD in A.adl, B.adl, C.adl, and D.adl files respectively.

broker_start

Use the broker_start utility to start a Broker Server.

To use this utility, the Broker Monitor must be running on the machine where the Broker Server resides. This utility waits up to 20 seconds for the Broker Server to start. When the Broker Server takes more than a second to start, you get a status update on the command line. For information about starting a Broker Server, see [“Starting Broker Server” on page 71](#).

Syntax

```
broker_start [-h] [server[:port]] [-monitor_port port] [-timeout timeout]
```

Argument	Description
-h	Displays a usage message.
[server[:port]]	Specifies the name of the Broker Server you want to start. If you omit the Broker Server name, the Broker Server on the local host is assumed. If you omit the port number, the default port 6849 is assumed.
[-monitor_port port]	Specifies the port number you want to use for the Broker Monitor port. You must specify the port number when you have Broker Monitor running on a non-default port (that is, a port other than 6850), or when multiple Broker Monitors are running on the same machine. If you do not specify a port number, the default Broker Monitor port, 6850, is assumed.
[-timeout timeout]	Specifies the maximum time in seconds the "Starting..." message is displayed at the command prompt. If the broker_start utility completes before the timeout time elapses, the "Started." message is displayed at the command prompt. If the timeout time elapses before the completion of the broker_start utility, the broker_start

Argument	Description
	operation continues in the background and returns the command prompt.

broker_status

The `broker_status` utility displays statistics from the command line for a specific Broker. The statistics displayed include Broker status, document delivery statistics, and client statistics.

Note: **Total documents queued for clients** is the total number of documents queued by Broker since Broker was created and not the number of documents that is currently in the queue.

Syntax

```
broker_status [-h] [-idhelp] [id_options] [-timeout timeout]
[broker@]server[:port] ...
```

Argument	Description
-h	Displays a usage message.
-idhelp	Displays a usage message for the [<i>id_options</i>] listed below.
[<i>id_options</i>]	Provides identification needed for administrative access to the Broker if the Broker is protected by an Access Control List (ACL). <ul style="list-style-type: none"> -entrust uses entrust as the SSL provider. Default is JSSE. -keystore<i>filename</i> specifies the name of the file containing the user SSL certificate. -password<i>password</i> specifies the password for the keystore file. -truststore<i>filename</i> specifies the name of the file containing the trusted root of the user SSL certificate. -noencrypt specifies to not use encryption for the connection. By default, every connection using a certificate is encrypted.

Argument	Description
	<p><code>-basicauthuserusername</code> specifies the user name for basic authentication. If you specify <code>-basicauthuser</code>, you must also specify <code>-basicauthpassword</code>.</p> <p><code>-basicauthpasswordpassword</code> specifies the password for the basic authentication user. If you specify <code>-basicauthpassword</code>, you must also specify <code>-basicauthuser</code>.</p>
<code>[-timeout timeout]</code>	Specifies the timeout time in seconds for the API calls made by the <code>broker_status</code> utility. If any of these API calls to the Broker Server do not complete before the specified time, a time-out exception is thrown by the <code>broker_status</code> utility.
<code>[broker@]server[:port]</code>	Specifies the name of the webMethods Broker (and optionally, the Broker and port number) from which to receive status. If you omit the Broker name, the Broker Server sends the status of all Brokers.

broker_stop

The `broker_stop` utility stops a specified Broker Server.

To use this utility, the Broker Monitor must be running on the machine where the Broker Server resides. This utility waits up to 20 seconds for the Broker Server to stop. When the Broker Server takes more than a second to stop, you get a status update on the command line. For information about stopping a Broker Server, see [“Stopping Broker Server ” on page 73](#).

Syntax

```
broker_stop [-h] [-idhelp] [-y] [server[:port]] [-monitor_port port]
[id_options] [-timeout timeout]
```

Argument	Description
<code>-h</code>	Displays a usage message.
<code>-idhelp</code>	Displays a usage message for the <code>[id_options]</code> listed below.
<code>-y</code>	Does not prompt for confirmation before stopping the Broker Server.

Argument	Description
<code>server[:port]</code>	Specifies the address of the Broker Server you want to stop. If you omit the Broker Server name, the Broker Server on the localhost is assumed. If you omit the port number, the default port 6849 is assumed.
<code>[-monitor_port port]</code>	Specifies the port number you want to use for the Broker Monitor port. You must specify the port number when you have Broker Monitor running on a non-default port (that is, a port other than 6850) or when multiple Broker Monitors are running on the same machine. If you do not specify a port number, the default Broker Monitor port, 6850, is assumed.
<code>[id_options]</code>	<p>Provides identification needed for administrative access to the Broker if the Broker is protected by an Access Control List (ACL).</p> <ul style="list-style-type: none"> -entrust uses entrust as the SSL provider. Default is JSSE. -keystore<i>filename</i> specifies the name of the file containing the user SSL certificate. -password<i>password</i> specifies the password for the keystore file. -truststore<i>filename</i> specifies the name of the file containing the trusted root of the user SSL certificate. -noencrypt specifies to not use encryption for the connection. By default, every connection using a certificate is encrypted. -basicauthuser<i>username</i> specifies the user name for basic authentication. If you specify -basicauthuser, you must also specify -basicauthpassword. -basicauthpassword<i>password</i> specifies the password for the basic authentication user. If you specify -basicauthpassword, you must also specify -basicauthuser.
<code>[-timeout timeout]</code>	Specifies the maximum time in seconds the "Stopping..." message is displayed at the command prompt. If the <code>broker_stop</code> utility completes before the timeout time elapses, the "Stopped." message is displayed at the command prompt. If the timeout time elapses before the completion of the <code>broker_stop</code> utility, the <code>broker_stop</code>

Argument	Description
	operation continues in the background and returns the command prompt.

Directing Command-line Utilities to Use a Non-Default JRE or JDK

webMethods and Intelligent Business Operations System Requirements document on the Empower Product Support website lists the JDKs supported by Broker. You can direct Broker command-line utilities to use a non-default JRE or JDK. If you redirect to a non-default JRE or JDK, apply maintenance updates from the appropriate vendor on a regular basis, as you would for JREs and JDKs you install yourself.

If you want to switch to the JDK installed by the installer, it is located in the *Software AG_directory\jvm* directory.

Important: On Windows, Broker command-line utilities can only be used with 32-bit JRE or 32-bit JDK.

Note: Software AG tests products only with the JDKs installed by the Software AG Installer. If you redirect products to use a different JDK or JRE and encounter issues, Software AG might require you to reproduce the issues with the JDK that is installed by the Software AG Installer.

To direct command-line utilities of webMethods Broker APIs for Java and JMS to a Non-Default JRE or JDK

1. Require JDK 1.6.
2. In the step below, *command* is any command other than `awbroker`, `awbrokermon`, `server_config`, or `server_qsck`.

```
webMethods Broker_directory\bin\command -DJAVA_HOME
=JDK_path host:port
```

On a Windows system, the command might look like this:

```
C:\SoftwareAG\Broker\bin\broker_status
-DJAVA_HOME=C:\jdk1.6 localhost
```

On a UNIX system, the command might look like this:

```
/opt/softwareag/Broker/bin/broker_status
-DJAVA_HOME=/opt/java1.6 localhost
```

Specifying the JVM Options in Command-line Utilities

The following command line utilities starts the JVM (Java Virtual Machine) internally to perform the tasks:

- `broker_buildall`
- `broker_create`
- `broker_delete`
- `broker_load`
- `broker_ping`
- `broker_save`
- `broker_start`
- `broker_status`
- `broker_stop`

You can specify any of the JVM options in these utilities. The command-line utility detects the JVM arguments and passes the arguments to the launching JVM.

For example, If you want to load large adl files on Broker, you can specify JVM memory of an initial size of 250MB and a maximum size of 512MB using the `-Xms` and `-Xmx` JVM options in the `broker_load` command as shown below:

```
broker_load -Xms250m -Xmx512m test.adl "Broker #1@localhost:6849"
```

B webMethods Broker Storage Utility

- Overview 568
- Processing Modes 568
- Processing Phases 568
- Files on which the Utility Operates 569
- Running the Broker Storage Utility in Check Mode 572
- Running the Broker Storage Utility in Fix Mode 573
- server_qsck Command Reference 574

Overview

Unexpected shutdowns of Broker Server caused by hardware failures or out-of-storage conditions can leave the Broker Server's queue storage system in an inconsistent state. To prevent further damage to the storage system, Broker Server will not start until these inconsistencies are reconciled.

The *Broker Storage Utility* is a command-line utility that you can use to scan queue storage files for inconsistencies. If the utility detects inconsistencies, you can use its commands to return the storage system to a consistent state.

Supported Broker Servers

You can use the Broker Storage Utility with Broker Server 6.0 and later.

The Broker Storage Utility supports both combined queue storage configurations and separate queue storage configurations.

Processing Modes

You can run the Broker Storage Utility in two modes.

- **In check mode**, the utility examines the queue storage system and reports inconsistencies that it detects. For information about running the utility in this mode, see [“Running the Broker Storage Utility in Check Mode” on page 572](#).
- **In fix mode**, the utility reconciles the inconsistencies that it detects in the queue storage system. For information about running the utility in this mode, see [“Running the Broker Storage Utility in Fix Mode” on page 573](#).

Processing Phases

In both processing modes, the Broker Storage Utility processes the storage files in three phases.

- **During phase one**, the utility checks the lowest level aspects of storage and ensures that the storage system is transactionally consistent (that is, that storage updates are applied as logical units of work). It also checks queue storage for missing or corrupt log and/or storage files.
- **During phase two**, the utility verifies that the mapping of the Broker's hierarchical data model to physical storage is intact and contains no inconsistencies. During this phase it verifies that the Broker Server contains logical Brokers, which in turn contain lists of clients, client groups, and document types. The utility does not examine queue storage for these individual objects or inspect the detailed contents of client queues.

- **During phase three**, the utility verifies that Broker's metadata is sufficiently consistent to allow the Broker to start properly. Items that the utility examines during this phase include, but are not limited to, metadata about clients, client groups, document types, and shared territory information. In fix mode, the utility removes items that are beyond repair.

The utility advances to the next phase only if it completes the current phase successfully. For example, if you run the utility in check mode and it encounters inconsistencies during phase one, the utility reports the inconsistencies and exits. You must resolve the inconsistencies in phase one before the utility will proceed to phase two.

Files on which the Utility Operates

In check mode, the Broker Storage Utility reads the set of files that make up the queue storage system. In fix mode, the utility reads and writes to these files. During initial problem diagnosis, you run the Broker Storage Utility in check mode on your existing queue storage files. If you decide that you want to run the utility in fix mode to reconcile inconsistencies in queue storage, you must do so on a *copy of queue storage*, not the original files. After the utility finishes, replace the original files with the repaired files.

To create a copy of queue storage that the utility can use, you must create a separate directory that contains the Broker Server's configuration file (`awbroker.cfg`) and *all* of the Broker Server's queue storage files. If your queue storage files are currently distributed across different directories or devices, you must copy the files from their existing locations into this single directory.

Tip: To locate all of the files associated with your queue storage system, run the `"server_config storage"` command with the `-list` option. For information, see ["server_config storage" on page 543](#).

Anytime you ask the Broker Storage Utility to operate on a directory other than the Broker Server's actual data directory, the utility expects to find all of the queue storage files in that directory. It does not look at the original queue storage files, only the ones in the directory upon which you have asked it to run. If you have not copied the complete set of queue storage files into this directory, the utility will fail.

The following tables identify the files that you must locate and place into the directory when you make a copy of queue storage. Many of these files exist in the Broker Server's data directory, so you might want to start by making a copy of that directory.

Filenames and Locations for Separate Storage Sessions

If you are using separate storage session, copy the following set of files into a single directory. Run the Broker Storage Utility on this directory.

Required Files	Description
awbroker.cfg	The Broker Server's configuration file. This file resides in the Broker Server's data directory.
BrokerConfig.qs and BrokerData.qs	The queue storage catalog files. These files reside in the Broker Server's data directory.
<i>BrokerConfig</i> .qs.log and <i>BrokerData</i> .qs.log	The queue storage log files. The default names for the log files are shown here. However, the names of these log files can be specified when a Broker Server is created. The names might be different on your system. Often the log files reside in the Broker Server's data directory, but sometimes they reside in different directories or on different devices.
<i>BrokerConfig</i> .qs.stor and all other *.qs.stor files for the configuration storage session <i>BrokerData</i> .qs.stor and all other *.qs.stor files for the run-time storage session	The storage files associated with the queue storage system. The default names of the storage files are shown here. However, storage file names can be specified when they are created. They might be different on your system. Sometimes storage files reside in the Broker Server's data directory, but often they exist in different directories or on different devices. Be certain to locate and copy all of the storage files associated with each storage session. If necessary, use the “server_config storage” on page 543 with the -list option to locate them all.

Example

If your existing queue storage files were distributed as follows:

```
C:\webMethods Broker_directory\data\awbrokersversion_number\MyBroker
  awbroker.cfg
  BrokerConfig.qs
  BrokerData.qs
C:\MyBrokerLogs
  MyBrokerConfig.qs.log
  MyBrokerData.qs.log
D:\MyBrokerStorage
  MyBrokerConfig01.qs.stor
  MyBrokerData01.qs.stor
  MyBrokerData02.qs.stor
```

You would gather all of these files into a single directory that looks like this:

```
E:\CopyofMyBrokerQueueStorage
  awbroker.cfg
  BrokerConfig.qs
  BrokerData.qs
```

```
MyBrokerConfig.qs.log
MyBrokerData.qs.log
MyBrokerConfig01.qs.stor
MyBrokerData01.qs.stor
MyBrokerData02.qs.stor
```

Filenames and Locations for a Combined Storage Session

If you are using a combined storage session, copy the following set of files into a single directory. Run the Broker Storage Utility on this directory.

Required Files	Description
awbroker.cfg	The Broker Server's configuration file. This file resides in the Broker Server's data directory.
Broker.qs	The queue storage catalog file. This file resides in the Broker Server's data directory.
<i>Broker</i> .qs.log	<p>The queue storage log file. The default name of the log file is shown here. However, the name of the log file can be specified when a Broker Server is created. It might be different on your system.</p> <p>Often the log file resides in the Broker Server's data directory, but sometimes it resides in a different directory or on a different device.</p>
<i>Broker</i> .qs.stor and all other *.qs.stor files for the combined storage session	<p>The storage files associated with the queue storage system. The default name of the initial storage file is shown here. However, the name of this file can be specified when the Broker Server is created. Its name might be different on your system. Additionally, multiple storage files can be defined, so your Broker Server might have more than one.</p> <p>Sometimes storage files reside in the Broker Server's data directory, but often they exist in different directories or on different devices. Be certain to locate and copy all of the storage files associated with the queue storage system. If necessary, use the “server_config storage” on page 543 with the -list option to locate them all.</p>

Example

If your existing queue storage files were distributed as follows:

```
C:\webMethods Broker_directory\data\awbrokersversion_number\MyBroker
  awbroker.cfg
  Broker.qs
  MyBroker.qs.log
```

```
D:\MyBrokerStorage
MyBroker01.qs.stor
MyBroker02.qs.stor
MyBroker03.qs.stor
```

You would gather all of these files into a single directory that looks like this:

```
E:\CopyofMyBrokerQueueStorage
awbroker.cfg
Broker.qs
Broker.qs.log
MyBrokerConfig.qs.log
MyBroker01.qs.stor
MyBroker02.qs.stor
MyBroker03.qs.stor
```

Running the Broker Storage Utility in Check Mode

Use the following procedure to run the Broker Storage Utility in check mode. In this mode, the utility will report inconsistencies in the queue storage system but will not attempt to reconcile them.

To run the Broker Storage Utility in check mode

1. Stop the Broker Server and navigate to the following directory:

```
webMethods Broker_directory/bin
```

2. Run the following command to start the utility:

```
server_qsck check dataDir [-p phaseNum] [-v]
```

where *dataDir* is the path to the Broker Server's data directory (if you want the utility to operate on the original queue storage files) or the directory that contains a complete copy of the queue storage system (if you want it to operate on a copy of the files). For more information about running the utility on a copy of queue storage, see [“Files on which the Utility Operates” on page 569](#).

Include...	If you want to...
-p	Specify the phase through which you want to perform the check. For example, if you want the utility to perform phases one and two, set this flag to 2. If omitted, all phases are performed.
-v	Enable verbose mode, which displays informational messages and identifies the object on which the utility is operating. If omitted, only error messages and inconsistencies are displayed.

For example:

```
server_qsck check /var/opt/webmethods/awbrokersversion/myBroker -p 1 -v
```

For additional information about the syntax for the "server_qsck" command, see ["server_qsck Command Reference" on page 574](#).

Running the Broker Storage Utility in Fix Mode

Use the following procedure to run the Broker Storage Utility in fix mode. In this mode, the utility examines queue storage for inconsistencies and makes the changes necessary to resolve the inconsistencies.

Important: This procedure will modify your queue storage file in an attempt to resolve inconsistencies. You should only run the utility in this mode on a copy of your queue storage system. For information about making a copy of queue storage, see ["Files on which the Utility Operates" on page 569](#).

By default, the utility automatically resolves inconsistencies that it detects in the storage system. However, you can optionally run the utility in "ask" mode, which instructs the utility to prompt you before it makes a change. For example, if the utility discovers that the storage system's log file is missing, it will ask whether you want to recreate the log file as shown here:

```
Phase 1: QS Log file for
"/var/opt/webmethods/awbrokersversion/default/Broker.qs"
  named "/var/opt/webmethods/awbrokersversion/default/Broker.qs.log" was not
found.
Attempt to recreate (y/yes)? _
```

If you reject the solution that the utility suggests, the utility will exit. For example, if the utility reports that the log file is missing, you might want to decline the utility's proposed solution (which is to recreate the file) and check whether the file has simply been moved to another directory or whether its location had been incorrectly specified in the Broker Server configuration file.

If the utility cannot reconcile inconsistencies in the storage system, make sure you have a copy of the original, unmodified storage files and contact Software AG Global Support for assistance.

To run the Broker Storage Utility in fix mode

1. Stop the Broker Server and navigate to the following directory:

```
webMethods Broker_directory/bin
```

Important: If you have not already made a copy of the queue storage files, do so before you continue. For more information, see ["Files on which the Utility Operates" on page 569](#).

2. Run the following command to start the utility:

```
server_qsck fix copyDir [-a] [-p phaseNum] [-v]
```

where *copyDir* is the path to the directory that contains a copy of the queue storage files.

Include...	If you want to...
-a	Instruct the utility to prompt you before it attempts to resolve inconsistencies. Omit the -a flag if you want the utility to automatically reconcile inconsistencies that it encounters.
-p	Specify the phase through which you want the utility to execute. For example, if you want the utility to reconcile inconsistencies uncovered during phases one and two, set this flag to 2. If omitted, all phases are performed.
-v	Enable verbose mode, which displays informational messages and identifies the object on which the utility is operating. If omitted, only error messages and inconsistencies are displayed.

For example:

```
server_qsck fix /var/opt/webMethods/awbrokersversion/myBrokerCopy -a -v
```

For additional information about the syntax for the `server_qsck` command, see [“server_qsck Command Reference” on page 574](#).

3. After you have corrected all the files, run the utility in check mode one last time to verify that there are no residual errors. Make corrections as necessary.
4. Copy the queue storage files back to their original locations.

server_qsck Command Reference

The `server_qsck` utility has the following four sub-commands:

```
server_qsck check
server_qsck fix
server_qsck help
server_qsck -version
```

server_qsck check

Performs a read-only check of the storage files. In this mode, the utility will apply the changes in the queue storage log file to the storage (.stor) files (that is, play the log file), but it will not fix any inconsistencies or make any other changes.

Checking stops as soon as an inconsistency occurs.

Note: Should the `server_qsck` utility encounter any errors reported by the system, the utility will print out an error message and exit with a non-0 exit code. You

can then report this message and exit code information back to Software AG Global Support for further assistance.

Syntax

```
server_qsck check dataDir [-p phaseNum]
[-v]
```

Argument	Description
<i>dataDir</i>	The fully qualified path to the Broker Server's data directory (if you want the utility to operate on the original queue storage files) or the directory that contains a complete copy of the queue storage system (if you want it to operate on a copy of the files).

Flag	Description
-p	<p>Specifies the phase through which you want to perform the check, where <i>phaseNum</i> specifies the phase as follows:</p> <ul style="list-style-type: none"> ■ 1-Checks whether the underlying data files are consistent. ■ 2-Checks whether the layer interfacing the Broker to the data files is consistent. ■ 3-Checks whether the basic metadata in the Broker is consistent. <p>If this flag is omitted, all three processing phases are performed.</p> <p>For more information about processing phases, see "Processing Phases" on page 568.</p>
-v	Enables verbose mode. This mode displays informational messages and identifies the object on which the utility is operating. If omitted, only error messages and inconsistencies are displayed.

Example

```
server_qsck check /var/opt/webMethods/awbrokersversion/myBroker -p 1 -v
```

server_qsck fix

Checks the storage files for inconsistencies and reconciles any inconsistencies that it finds.

Important: Make a copy of the storage system files before you use this command. Run the utility on the copies instead of the original files. If the utility is not able to resolve the issues with your storage files, you will need to have the original, unmodified files if you contact Software AG Global Support.

Syntax

```
server_qsck fix copyDir [-a] [-p phaseNum]
[-v]
```

Argument	Description
<i>copyDir</i>	The fully qualified path to the directory that contains a copy of the queue storage files. See “Files on which the Utility Operates” on page 569 for information about creating this directory.
Flag	Description
-a	Enables ask mode. In this mode, the utility prompts you before it reconciles an inconsistency. If omitted, the utility automatically reconciles inconsistencies.
-p	Specifies the phase through which you want to perform the check, where <i>phaseNum</i> specifies the phase as follows: <ul style="list-style-type: none"> ■ 1-Checks whether the underlying data files are consistent. ■ 2-Checks whether the layer interfacing the Broker to the data files is consistent. ■ 3-Checks whether the basic metadata in the Broker is consistent. <p>If this flag is omitted, all three processing phases are performed.</p> <p>For more information about processing phases, see “Processing Phases” on page 568.</p>
-v	Enables verbose mode. This mode displays informational messages and identifies the object on which the utility is operating. If omitted, only error messages and inconsistency messages are displayed.

Example

```
server_qsck fix /var/opt/webMethods/awbrokersversion/myBrokerCopy -a -v
```

server_qsck help

Displays usage information for the server_qsck command or one of its subcommands.

Syntax

```
server_qsck help [subCommand]
```

Argument	Description
<i>subCommand</i>	The specific subcommand (e.g., <code>check</code> or <code>fix</code>) for which you want usage information. If <i>subCommand</i> is omitted, general usage information is displayed.

Example

```
server_qsck help fix
```

server_qsck -version

Displays the Broker Storage Utility's version number.

Syntax

```
server_qsck -version
```


C Broker SSL Distinguished Name Syntax

- Overview 580
- Distinguished Name Syntax 580
- Using Distinguished Names with Broker 580

Overview

This appendix is a reference for the syntax conventions used when configuring SSL distinguished names (DNs) with Broker. A DN is that portion of a certificate that identifies either the owner or issuer of the certificate.

Distinguished Name Syntax

webMethods Broker uses the following format conventions for DNs:

- A field consists of a tag and a value, separated by an equal sign.
- Fields are separated by commas.
- Tags are case insensitive (CN or cn).
- Order of the fields is not taken into account.
- If a field is not set, it is omitted.
- Values that contain commas or equal signs must be contained within quotation marks.

The following is an example of a distinguished name:

```
CN=John Smith,OU=Engineering,O=ABCDCorp,  
L=Sunnyvale,ST=CA,C=USA,EMAIL=jsmith@ABCDCorp.com
```

Broker supports the use of multi-valued attributes separated by commas. For example, a distinguished name could have two common names, as follows:

```
CN=John Smith,CN=JSMITH
```

Using Distinguished Names with Broker

Follow these rules when entering DNs on the command line for utilities used with Broker.

- Enclose DNs in double quotation marks.
- If a value within the distinguished name contains one of the characters shown here in parentheses (, ; = + < > #), enclose the value with double quotation marks (as in O="Software AG USA, Inc.").
- On Windows, you must escape each quotation mark by preceding it with a backslash (\"). On UNIX (except for C shell), you do not need to escape interior double quotation marks if you enclose the DN in single quotation marks.

D Managing Certificate Files with OpenSSL

■ Overview	582
■ Managing Certificate Files	582

Overview

This appendix shows how to use the OpenSSL command editor for basic operations with Broker security, such as creating and configuring keystore and trust store files in different formats, converting certificate files from one format to another, and sending a request for a signed certificate to a CA.

Managing Certificate Files

Following is a list of typical operations needed to configure and manage SSL certificate (keystore and trust store) files:

- Generating a key and certificate request
- Viewing a certificate request
- Building a keystore
- Converting between certificate file formats (PEM and PKCS12)
- Viewing a PKCS12 file
- Viewing an SSL certificate in a PEM file
- Building a trust store

Generating a Key and Certificate Request

Use this command to generate a key pair and a signed certificate request to be sent to the certification authority (CA):

```
openssl req -newkey rsa:<keysize> -keyout <keyfile>.pem -keyform PEM  
-out <request>.pem -outform PEM
```

where the values for `-keyform` and `-outform` can be PEM or DER, and `keysize` is typically 1024 or 2048 (the higher the value, the larger the key size and the greater the security).

The command will prompt for the following:

- Passphrase for the keyfile
- Information about the certificate (DN components), depending on what is specified in `openssl.cnf`
- Extra attributes, which you can ignore

Viewing the Certificate Request

Use this command to view the certificate request that you will send to the CA:

```
openssl req -in <request>.pem -text -noout
```

After sending the request to the CA, you should get back a signed certificate.

Building a Keystore

Use this command to create a keystore file in PEM format:

```
cat <signed-certificate>.pem <keyfile>.pem <keystore.pem>
```

If you are using PKCS12 as the file type (preferred) instead of PEM, you must additionally convert the PEM file to PKCS12, as follows:

```
openssl pkcs12 -export -in <keystore>.pem -out <keystore.p12>
    -name "friendly name"
```

The "friendly name" (sometimes referred to as an alias) is optional. It can be any string.

Viewing a Certificate in a PKCS#12 File

This operation cannot be done directly using OpenSSL. You must first copy out the PKCS12 certificate to a PEM file, and then view the certificate in the PEM file.

```
openssl pkcs12 -in <keystore>.p12 -clcerts -out <afile>.pem
```

Viewing a Certificate in a PEM File

The following command allows you to dump the entire certificate in a readable format:

```
openssl x509 -in <afile>.pem -text -noout
```

You can also select which parts of the certificate to view, for example:

```
openssl x509 -in <afile>.pem -subject -noout
openssl x509 -in <afile>.pem -issuer -noout
openssl x509 -in <afile>.pem -dates -noout
```

By default, the file is assumed to be in PEM format. If the file is in DER format, you can specify the following:

```
-inform DER
```

Building a Trust Store

If the CA certificates are in PEM format, you only need to concatenate them to build the trust store; for example:

```
cat <ca-cert-1>.pem <ca-cert-2>.pem ... <ca-cert-n>.pem >
    <truststore>.pem
```

If the CA certificates are *not* in PEM format, convert them to PEM format first and then concatenate them.

If the CA certificates are in PKCS12 format, do the following:

```
openssl x509 -in <ca-cert-i>.p12 -out <ca-cert-i>.pem
```

If the CA certificates are in DER format, do the following:

```
openssl x509 -in <ca-cert-j>.der -inform PEM  
            -out <ca-cert-j>.der -outform PEM
```

E Using Access Labels

■ Overview	586
■ Types of Access Control	586
■ Per-Message Security	586
■ What Is an Access Label?	586
■ Using Access Labels	588
■ Acquiring an Access Label through Access Label Adapter	590
■ Behavior Across Broker Boundaries	595

Overview

Access labels allow the granting of user rights on a per-message basis. This appendix explains how access labels work in webMethods Broker and how to implement them.

Types of Access Control

A Broker client is allowed to publish and subscribe to documents based on:

- Document type
- Its client group can-publish and can-subscribe lists

webMethods Broker also supports access control at the individual document level. This control is *per-message based* rather than *type based*.

In certain situations where different access levels are required, type-based access control may be insufficient or may require too complex a solution.

For example, assume a document type `deleteRecord`. To provide different user rights based on Confidential, Secret, and Top Secret access, you might need to publish three document types: `deleteConfidentialRecord`, `deleteSecretRecord`, and `deleteTopSecretRecord`. This requires the creation of three separate client groups with different permissions for accessing the document types.

Access labels allow you to specify permissions for a document instance. In the above example, the Confidential, Secret, and Top Secret clients can use the single document type `deleteRecord`, with individual documents of that type having different access control levels.

Per-Message Security

Access labels allow you to prevent a Broker client from receiving a specific document regardless of whether its client group is authorized to receive the document type to which the message belongs. The publisher labels a document's access rights, and Broker enforces access at the per-message level.

Access labels also allow you to provide a receiving Broker client with information about the publisher's rights.

What Is an Access Label?

An access label is a bitmask composed of combinations of ones and zeroes. Examples of access label bitmasks are shown in the following table:

Bitmask	Description
110001	6-bit access label
100100	6-bit access label
110010010011	12-bit access label
100010001000	12-bit access label
110011001100110011001100	24-bit access label
100000000000100000000000	24-bit access label

You define access labels using a sequence of unsigned short values. Each value in the sequence represents a position in the bitmask. For the example bitmasks shown above, the corresponding access label definitions are shown in the following table:

Bitmask	Access Label Definition
110001	label[0]=5 label[1]=4 label[2]=0
100100	label[0]=5 label[1]=2
110010010011	label[0]=11 label[1]=10 label[2]=7 label[3]=4 label[4]=1 label[5]=0
100010001000	label[0]=11 label[1]=7 label[2]=3
110011001100110011001100	label[0]=23 label[1]=22 label[2]=19 label[3]=18 label[4]=15 label[5]=14

Bitmask	Access Label Definition
	label[6]=11 label[7]=10 label[8]=7 label[9]=6 label[10]=3 label[11]=2
100000000000100000000000	label[0]=23 label[1]=11

When defining access labels for the clients, duplicate values in the label are ignored. For example, the following labels:

```
label [0]=5
label [1]=4
label [2]=0
label [3]=5
label [4]=4
label [5]=0
```

are the same as:

```
label [0]=5
label [1]=4
label [2]=0
```

and both yield the following bitmask:

```
110001
```

However, duplicate values are not allowed when specifying an access label for a document during a publish operation.

The order in which you define the sequence of access labels is unimportant. For example, the following two groups of labels yield the same bitmask:

```
label [0]=5
label [1]=4
label [2]=0
```

and

```
label [0]=4
label [1]=0
label [2]=5
```

Note: Negative values are invalid and will generate errors.

Using Access Labels

Access labels work by performing a bitwise comparison between the access label bitmask assigned to the client and the access label bitmask assigned to the document.

A document is accessible by a client only if the document bitmask 1 bits have corresponding 1 bits in the client bitmask. For example, a client with the following access label bitmask:

```
110001
```

can access a document with access label bitmask

```
110001 or 100001
```

However, it cannot access a document with the following access label bitmasks:

```
100100 or 000100
```

The following truth table describes the bitmask comparison logic:

Document bit value	Client Bit Value: 0	Client Bit Value: 1
0	1	1
1	0	1

You can also describe the bitmask comparison logic with the following function:

```
if (document_access_label & (~client_access_label) != 0)
//FAIL
    else
//PASS
```

In general, setting more bits in the access level bitmask creates higher access levels. For example, setting a client access label bitmask to:

```
1111111
```

provides access to any document with an access label with any combination of the 6 lowest bits (0 through 5) set.

You can also use an access label to determine if a client has the right to label a document with a certain access level during a publishing operation. It works the same as determining whether a client can access a document with a certain access level.

For example, a client with the following access level bitmask:

```
110001
```

Can label a document with one of the following access labels:

```
110001 or 100001
```

But cannot label a document with these access labels:

```
100100 or 000100
```

The access label assigned to particular document is called a control label, and it is stored in the envelope field `_env.controlLabel`.

A publishing client is responsible for setting this field using an API. An option is provided to have the Broker set this field for the client by using the client's access label. This option (normally disabled) is enabled by using the following API calls:

- For Java API:

```
void BrokerClient.setAutomaticControlLabel (Boolean enabled);
```

- For JMS API:

```
void WmConnectionFactory.setAutomaticControlLabel (Boolean enabled);
```

- For C# API:

```
Boolean IConnectionFactory.AutomaticControlLabel;
```

Note: For more information about the API calls, see the appropriate webMethods Broker Programmer's Guide.

When publishing a document, the publishing client's access label is stored in the `_env.pubLabel` envelope field. This field is called the source label. The Broker does not make use of this information, but the receiving clients can. One possible usage for a receiving client is to determine the type of information it will return back to the publishing client in a request/reply situation.

The client's access label is sometimes called the receipt label. The receipt label is checked only when documents are pulled from the Broker client queue for delivery to the client. The receipt label is not checked when documents are placed in the queue.

Acquiring an Access Label through Access Label Adapter

If a Broker client has an identity (a distinguished name or DN) and an authenticator name (the issuer DN of the client's SSL certificate), the Broker automatically requests an access label for the client when establishing a client session, if an access label adapter (ALA) is available.

The ALA is a custom Java API program that uses a client on the system-defined `accessLabelAdapter` client group. It is not necessary for an ALA to create subscriptions.

The `accessLabelAdapter` client group has a life cycle of destroy on disconnect and storage type of volatile. You can place an Access Control List (ACL) on the `accessLabelAdapter` client group to control the access to this client group by ALAs.

Note: Software AG does not include an ALA as part of the product. You must create the adapter yourself.

The primary function of the ALA is to receive access label lookup documents (`Broker::ALA::lookup`) from the Broker, retrieve the access label from storage, and reply with access labels. A secondary function is to update or revoke access labels that the Broker already has.

The client access labels are usually stored in a file or a database as key-value pairs. The key fields for an access label are as follows:

- User DN (required)

- Authenticator DN (required)
- Certificate serial number (optional)
- Hint (optional)

The hint field allows a unique user DN, authenticator DN, and certificate serial number combination to have multiple access labels. If you use hint strings as a key field in the lookup table, then you must supply the appropriate hint strings (through `BrokerConnectionDescriptor`) when creating Broker clients.

Multiple ALAs can be connected to a single Broker and a single ALA can be connected to multiple Brokers. When multiple ALAs are connected to a single Broker, the Broker locates the ALA that contains the access label for a particular user. If multiple ALAs contain access labels for the same user, the Broker will use the value returned by the first ALA that it contacts.

Note: There is no advantage to using multiple ALAs unless you maintain multiple client access label databases.

ALA Communication

The Broker and ALA communicate using system-defined document types. All document types described are volatile with infinite time-to-live.

```
// Sent from Broker to ALA. ALA replies with
// Broker::ALA::label or Broker::ALA::error
Broker::ALA::lookup {
  unicode_string user;
  unicode_string authenticator;
  unicode_string serial;
  unicode_string hint;
};
// Sent from ALA to Broker, in response to
// Broker::ALA::lookup.
Broker::ALA::label {
  unicode_string user;
  unicode_string authenticator;
  short label[];
};
// Sent from ALA to Broker, in response to
// Broker::ALA::lookup.
Broker::ALA::error {
  unicode_string user;
  unicode_string authenticator;
  unicode_string error;
  unicode_string detail;
};
// Sent from ALA to Broker.
Broker::ALA::change {
  unicode_string user;
  unicode_string authenticator;
  unicode_string serial;
  short label[];
};
```

The ALA can use the `Broker::ALA::error` document to report problems during label lookup. The error field should be one of the following:

Error Field	Meaning
unknown	The user is not known to the ALA.
internal	A temporary failure has occurred. If the client has an existing label, it is not invalidated.

If the error string is not internal, the current access label is invalidated (e.g., cleared).

You can use the **detail** field to record an implementation-specific error message. If the **detail** field is present, the Broker logs an error for the failure. In error cases, any pending lookups for the user's label will fail. If there is a non-empty detail string, it is logged to the Broker Server's error log.

The ALA can use the `change` document to update an access label the Broker already knows about. You can effectively revoke a label by updating it with a zero-length label. The client is not notified when its access label changes.

The Broker delivers documents to the ALA client (that is, the `_env.destId` field is set). The ALA should confirm that `_env.pubId` is set to `//broker-name`.

The ALA should deliver documents to the Broker using `//broker-name` as the destination ID. The Broker ignores published ALA documents, as well as documents not delivered from a local member of the "accessLabelAdapter" client group.

Error Handling

The Broker takes action depending on the ALA failure mode:

Failure Mode	Broker Action
ALA is not running (not connected)	Requests for client access labels fail until the ALA is running again.
ALA does not respond	<p>Pending access label lookups fail after waiting 10 seconds for an ALA reply. Clients that already have access labels continue to function.</p> <p>In the future, the Broker may disconnect the ALA if the ALA does not respond to any requests for one minute. This might help get the ALA restarted.</p>
ALA disconnects	If an ALA disconnects, it will likely not respond within the 10 second time-out and requests to it will be treated as such.

Failure Mode	Broker Action
ALA supplies invalid replies	The ALA could reply to a lookup with an unknown DN, or invalid access label (for example, negative label values). Some invalid replies will be ignored, and some will result in the failure of an access label lookup. Regardless, the Broker will log each failure by the ALA.
Multiple ALAs running	The Broker uses the first ALA to connect; it ignores the other ALAs until the first one disconnects.

The Access Label Process

Use the following procedure to create an access label.

To create an access label

1. Configure the Broker.

The Broker uses clients connected to the system-defined client group `accessLabelAdapter` for access label lookups. You can use an ACL on the client group to restrict access to trusted adapters.

2. Start the ALA.

Multiple ALAs can be connected to a single Broker, and a single ALA can be connected to multiple Brokers. Client group replication helps support this usage by distributing the ACL. It is up to you to maintain consistency across multiple instances of the ALA.

The adapter does not need to make any subscriptions.

In a trusted system, the ALA should use SSL and have a certificate that matches one listed in the client group ACL.

3. Create the Broker client.

The creation of the client includes an application-defined hint string.

4. The Broker asks for an access label.

The Broker builds a `Broker::ALA::lookup` request document to get an access label for the client. The document includes the client user information (DN and issuer DN) and the application defined hint string.

5. The Broker delivers the document to one of the connected ALAs.

If there are no ALAs running on the Broker, the lookup fails and the client does not get an access label.

6. ALA returns an access label.

The ALA can make one of two responses to each label request: an access label or an error.

- A successful response (`Broker::ALA::label`) includes the access label.
- A failure response (`Broker::ALA::error`) includes an error type, DN, and optional error detail.

A response takes the form of a document delivered to the local Broker.

The ALA should be coded to ignore documents not from the local Broker (for example: check that `_env.pubId` is equal to `"//brokername"`).

The ALA should send its reply using `awDeliverReplyEvent` (or the equivalent Java API function) to make sure the envelope is properly set up. If the ALA does not use this function, the `_env.tag` field from the `Broker::ALA::lookup` document will not be copied over, and the result of the lookup will affect all outstanding lookup requests for that client's owning user.

The ALA should also copy the user and authenticator values from the request to the response. If the ALA does not copy these values, no clients will receive the labels.

7. The Broker gets a response from the ALA.

The Broker handles each kind of ALA response:

ALA Response	Broker Action
<code>Broker::ALA::label</code>	The Broker assigns the access label to the requesting client if this document includes a tag. If the tag is not set, any clients with pending requests for that user (DN and issuer DN) are updated.
<code>Broker::ALA::error</code>	Any clients with pending requests for that user (DN and issuer DN) get a failure code.

Expiration of Labels

Labels do not expire. They remain valid until the ALA reports a new label value for the client's owning user by means of a `Broker::ALA::change` document.

The Broker does attempt to refresh the client's access label each time the client reconnects. For shared-state clients, this happens each time the number of connected clients goes from zero to one.

If an ALA wants to expire labels, it may do so by issuing `Broker::ALA::change` documents, using its own expiration policy. Note that change documents apply to all clients with a matching DN and issuer DN pair.

Behavior Across Broker Boundaries

Document access labels are preserved unchanged between Brokers and across territory gateways. To maintain security, all interconnected Brokers and territories should share the same meaning for access label values.

F Configuration Audit Logging

■ Overview	598
■ Audit Log Description and Functionality	598
■ Configure Audit Logging and Identify the Information to Log	601
■ Viewing the Audit Log	602
■ Editing the Audit Log Parameters in the Broker Server Configuration File	602

Overview

This chapter describes the webMethods Broker configuration audit logging functionality, which allows you to log changes made to Broker Server, Broker objects, territories and gateways. The chapter explains how to configure the Broker Server configuration audit logging and describes how to view and maintain the audit log.

Audit Log Description and Functionality

The Broker Server configuration audit log receives messages from Broker Server. These messages represent any changes to the Broker Server configuration and authentication settings. For example, when a Broker is added or removed from the Broker Server, the Broker Server writes the event to the audit log. Audit log entries include the date and time the event occurred and other information depending on the connecting Broker client. For example, if the connecting client is a BrokerAdminClient, the log entries include the client ID.

You can use the information in the audit log to record and track any configuration changes made on the Broker Server. The information can be useful for recording events and troubleshooting.

Audit Log File Name and Location

The configuration audit log resides on the machine where Broker Server is installed. Each audit log file name is appended with a timestamp in seconds since 1970, for example, audit1183974252.log. The following table lists where the audit log files are located.

On this platform...	The default audit log directory is located here...
Windows	<i>C:\BrokerServerDataDir \auditlogs\audit<timestamp>.log</i>
UNIX	<i>BrokerServerDataDir /auditlogs/audit<timestamp>.log</i>

You can change the name and location of the audit log in the Broker Server configuration file. For instructions, see [“Editing the Audit Log Parameters in the Broker Server Configuration File” on page 602](#).

Audit Log File Size and Cache Size

The default audit log size is 20 megabytes; however, you can increase the maximum size up to 2 gigabytes. If the maximum size is exceeded, a new audit log is automatically created on the Broker Server. You can increase or decrease the maximum file size by

editing the Broker Server configuration file. For instructions, see [“Editing the Audit Log Parameters in the Broker Server Configuration File”](#) on page 602.

The `auditlog-writeInterval` parameter and the `auditlog-maxWriteCache` parameter control the number of audit records that are cached in memory before they are written to the log file. When either the `auditlog-writeInterval` parameter or the `auditlog-maxWriteCache` parameter achieves its assigned value, the audit records will be written to the file.

For example, if `auditlog-writeInterval=40` and `auditlog-maxWriteCache=200`, the audit records will be written to the file when either 200 records are collected in the cache or 40 seconds elapses after the last write (whichever value reaches first). The default value of `auditlog-maxWriteCache` is 500 and the default value of `auditlog-writeInterval` is 30 seconds.

Audit Log Categories

You can configure Broker Server to specify which types of activity you want to include in the log. The audit log can contain the following information:

Type	Description
Connection	<p>Records the following information when a connection is established to the Broker Server and/or disconnected from the Broker Server:</p> <ul style="list-style-type: none"> ■ Date and time the connection was established or disconnected ■ Hostname and port number ■ Socket file descriptor ■ Operation success or failure
Session	<p>Records the following information when a session is created or destroyed on the Broker Server.</p> <ul style="list-style-type: none"> ■ Date and time the session was established or destroyed ■ Session ID ■ Operation success or failure
Broker Server Modification	<p>Records the date, time and user information when the following changes are made to the Broker Server configuration:</p> <ul style="list-style-type: none"> ■ The license file is updated. ■ The server log settings are changed or purged.

Type	Description
	<ul style="list-style-type: none"> ■ The audit log settings are changed or purged. ■ A Broker is added and deleted.
Broker Server Security	Records the date, time and user information when changes are made to Broker Server ACL and SSL security settings.
Broker Server Start/Stop	Records the date, time and user information when the Broker Server is stopped or restarted.
Broker Modification	<p>Records the date, time, and user information when the following changes are made:</p> <ul style="list-style-type: none"> ■ A Broker description is changed. ■ The default Broker is changed. ■ Document Type Logging (Apply) is enabled or disabled.
Document Type Modification	<p>Records the date, time, and user information when the following changes are made to a Broker document type:</p> <ul style="list-style-type: none"> ■ A document type description is added or changed. ■ A Time to Live setting is changed. ■ A new topic is added (or pasted). ■ A document type is deleted.
Client Group Modification	<p>Records the date, time, and user information when the following changes are made to a client group:</p> <ul style="list-style-type: none"> ■ A new Client Group is added (or pasted). ■ An existing Client Group is deleted. ■ A Client Group description is changed. ■ A Log on Publish document type is added or removed from the client group. ■ A Log on Ack document type is added or removed from the client group.
Client Group Security	<p>Records the date, time, and user information when the following changes are made to client group security settings:</p> <ul style="list-style-type: none"> ■ The Access Label Required option becomes enabled for a Client Group.

Type	Description
	<ul style="list-style-type: none"> ■ ACL settings are changed. ■ A Can Publish document type is added or removed from the client group. ■ A Can Subscribe document type is added or removed from the client group.
Gateway Modification	<p>Records the date, time, and user information when the following changes are made to a gateway:</p> <ul style="list-style-type: none"> ■ A gateway Broker is added or deleted from the Broker Server. ■ Keep Alive, Static Gateway Forwarding, and Refuse Shared configuration settings are changed on a territory gateway.
Territory Modification	<p>Records the date, time, and user information when the following changes are made to a territory:</p> <ul style="list-style-type: none"> ■ A territory is added or deleted. ■ A Broker is added to a territory.
Client Queue	<p>Records the date, time, and user information when a client queue is cleared.</p>

Configure Audit Logging and Identify the Information to Log

Configure audit logging by using the user interface of the Broker Server in My webMethods.

To configure audit logging and to identify the information that you want to log

1. In My webMethods: **Messaging > Broker Server > Servers**
2. In the **Broker Server List**, click the server whose audit log you want to configure.
3. Click the **Audit Log** tab.
4. Select the **Enable Auditing** check box to enable audit logging.
5. Under **Audit Log Categories**, select the types of messages that you want Broker Server to include in the log. For a description of the log categories, see [“Audit Log Categories” on page 599](#).
6. Click **Apply**.

Viewing the Audit Log

You can view the audit log with any text editor. Each audit record contains the following information:

Field	Description
RecordId	A unique identifier for internal use.
Result	Indicator of success or failure for the operation. 0 indicates success; 1 indicates failure.
ClientId	The ID of the client performing the operation. If a client ID does not exist (for example, a server client does not have a client ID), this field will be NULL.
Operation	A description of the operation. The descriptions reflect the operations listed on the Audit Log tab.
Input	The input string given by the user for the operation being audited.
Date	The date and time at which the operation occurred and the record is logged.

Editing the Audit Log Parameters in the Broker Server Configuration File

The Broker Server configuration file (`awbroker.cfg`) contains the audit log configuration parameters. You can edit these parameters using any text editor.

To edit the audit log parameters in the Broker Server configuration file

1. Stop the Broker Server.
2. On the machine where Broker Server is installed, locate the Broker Server's configuration file (`awbroker.cfg`) and make a backup copy.
3. Open the configuration file in a text editor.
4. Edit the `auditlog-dir`, `auditlog-maxFileSize`, `auditlog-writeInterval`, and `auditlog-maxWriteCache` audit log parameters as necessary. For detailed

descriptions of these configuration parameters, see “ [webMethods Broker Server Configuration Parameters](#)” on page 621.

5. Save the configuration file.
6. Restart the Broker Server.

G webMethods Broker Document Logging

■ webMethods Broker Document Logging	606
■ Logging Utility Client Groups and Clients	607
■ Setting Up Document Logging	609
■ Logging Utility Management	613
■ Built-In Services	614

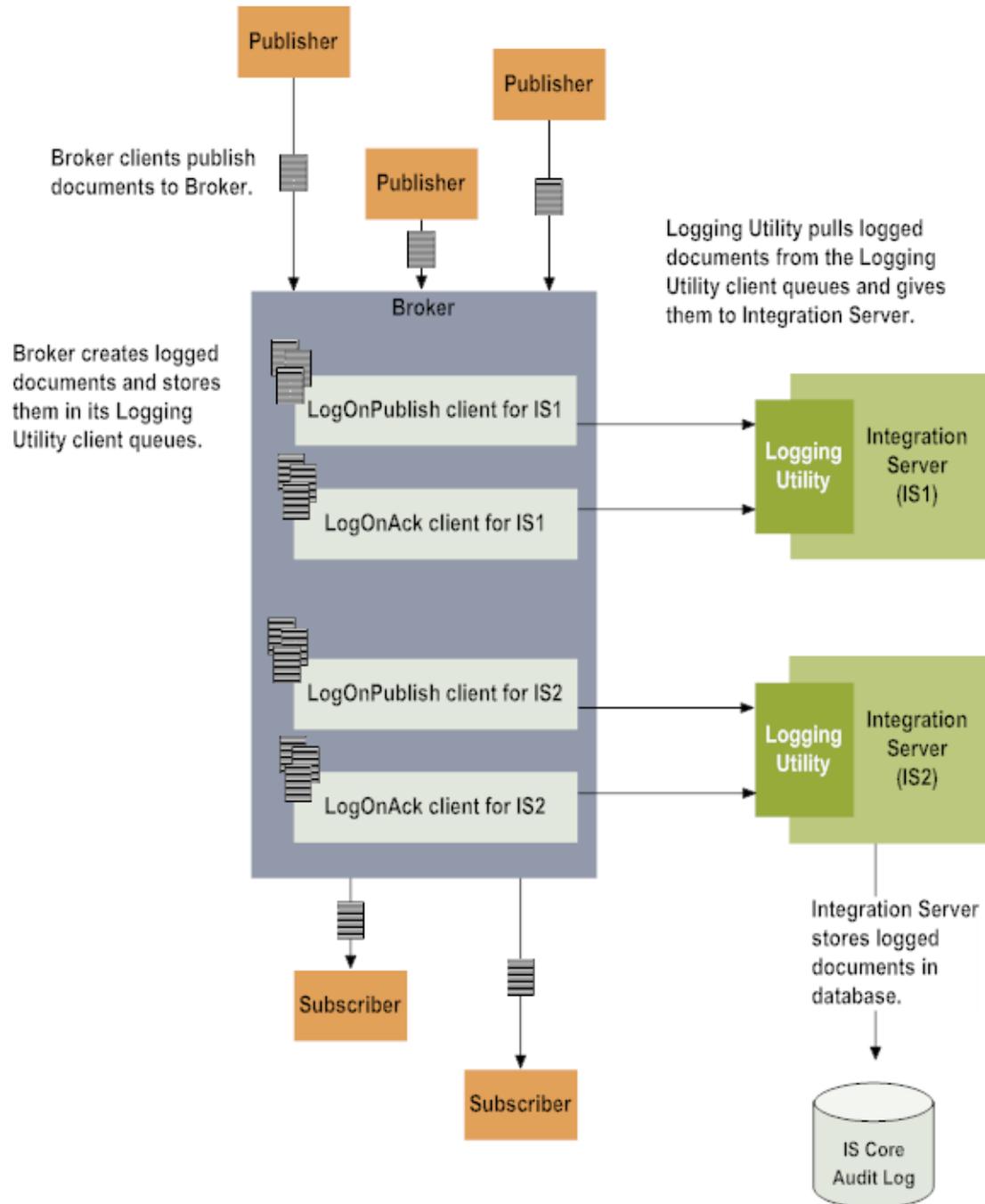
webMethods Broker Document Logging

You can configure your system to log documents that Broker clients publish to or subscribe to, on Brokers. Integration Server logs those documents to the IS Core Audit Log database component. Configuring your system to log documents that Broker clients publish to or subscribe to, on Brokers, involves these steps:

- During installation of webMethods Broker, you selected a preconfigured storage session for the Broker Server. Make sure the Broker Server storage files can accommodate the volume of document logging you expect. For instructions on adjusting the size of the storage file, see [“The Storage Files” on page 85](#).
- Install webMethods Logging Utility on Integration Server. The Logging Utility functions as a logging agent between Integration Server and Brokers. Also, install the IS Core Audit Log database component, if it is not already installed. For instructions, see *Installing webMethods and Intelligent Business Operations Products*.
- Configure Integration Server to connect to a Broker on which Broker clients will publish or subscribe to documents you want to log. For instructions, see *webMethods Integration Server Administrator’s Guide*.
- Set up document logging, as follows:
 - Enable document logging on the connected Broker.
 - Identify the documents you want the connected Broker to log, and specify whether to log the documents on publish or on subscriber acknowledgement.
 - Logging Utility (WmLogUtil package) is an Integration Server package. Therefore, it can communicate only with the Broker that is connected to Integration Server. If you want other Brokers in a territory or cluster also to perform document logging then you must configure Integration Server and Logging Utility for that Broker also.
 - Optionally, specify your own document IDs for logged documents. Doing so lets you provide a uniform business context for these documents that will be helpful when you view them later in My webMethods.
 - After document logging is set up, the connected Broker begins creating copies of the documents you identified. These copies are called *logged documents*, and are stored in Broker's *Logging Utility client queue*. Identify the number of logged documents Logging Utility must remove from Broker's Logging Utility client queue at one time and forward to Integration Server to store in the IS Core Audit Log database component.

For instructions, see [“Setting Up Document Logging” on page 609](#).

The following figure illustrates document logging with the Logging Utility.



Logging Utility Client Groups and Clients

When the Logging Utility connects to Broker for the first time, Broker creates two client groups and a client in each client group for Logging Utility:

- **LogOnPublish client group.** For grouping the Logging Utility client queues that store documents logged during publish to Broker.
- **LogOnAck client group.** For grouping the Logging Utility client queues that store documents logged during subscription acknowledgement to Broker.

Both client groups are created with the property "AllCanSubscribe," which means clients belonging to these client groups can subscribe to any document without the need to add that document to the client group's "cansubscribe" list.

Note: When you create a new Broker, the LogOnPublish and LogOnAck client groups are also created simultaneously with the default client groups.

Each client group contains the following clients:

- *<Integration Server client prefix>*_LogOnPublish. Broker creates this client in the LogOnPublish client group. The queue of this client stores the documents logged during publish to Broker.
- *<Integration Server client prefix>*_LogOnAck. Broker creates this client in the LogOnAck client group. The queue of this client stores the documents logged during subscriber acknowledgement to Broker.

Note: Logging Utility clients on Broker and Broker clients are different from one another. Do not consider them to be functionally similar.

Client ID

When you are connecting multiple Integration Servers to Broker, the Logging Utility on the Integration Server that first connects to the Broker creates the LogOnPublish and LogOnAck client groups. When the remaining Integration Servers connect to the Broker, only clients are created in the respective client groups.

Each Logging Utility client has an associated *Client ID*. The first part of the Client ID identifies the Integration Server with which the client is associated, and the second part identifies the associated Logging Utility client group.

Integration Server Client prefix is a string that identifies Integration Server to Broker. By default, Integration Server uses its license key for the prefix. For ease of use, you can replace it with a name of your choice.

Client ID has the following formats:

- *<Integration Server client prefix>*_LogOnPublish

For example, in the client ID, `Wjb5tM+bn8pBOTAWIc4XWbfdh5w_LogOnPublish`, `Wjb5tM+bn8pBOTAWIc4XWbfdh5w` is the name of the Integration Server, and `LogOnPublish` is the name of the Logging Utility client group.

- *<Integration Server client prefix>*_LogOnAck

For example, in the client ID, `Wjb5tM+bn8pBOTAWIc4XWbfdh5w_LogOnAck`, `Wjb5tM+bn8pBOTAWIc4XWbfdh5w` is the name of the Integration Server, and `LogOnAck` is the name of the Logging Utility client group.

Note: To be able to set up document logging effectively, the client prefixes of all Integration Servers in a cluster must be the same.

Identifying a Logging Utility Client by Client ID

Use the following procedure to identify a logging utility client by client ID.

To identify a Logging Utility client by client ID

1. Set the client prefix from Integration Server Administrator: **Settings > Messaging > webMethods Messaging Settings** For complete instructions, see *webMethods Integration Server Administrator's Guide*.
 - a. Click the Broker connection alias link.
 - b. Click **Edit Broker Connection Alias** and set the client prefix.
2. After setting the client prefix, restart Integration Server and log on to My webMethods Server.
3. In My webMethods: **Messaging > Clients**.
4. Specify a filter to select the Broker that is connected to Integration Server, and click **Go** to display the list of clients.
5. Search for the client ID with the client prefix you set in the Integration Server.

Setting Up Document Logging

You must set up document logging so that the Logging Utility clients in the LogOnPublish client group can log documents when publish activity occurs at Broker, and clients in the LogOnAck client group can log subscriber acknowledgement activity at Broker.

To set up document logging

1. Enable document logging. For more information, see [“Enable Document Logging” on page 610](#).
2. Identify the document types you want to log. For more information, see [“Identify Document Types to Log” on page 610](#).
3. Subscribe to document types from other Brokers. For more information, [“Subscribe to Document Types from Other Brokers in the Territory or Cluster” on page 611](#).
4. Specify document IDs. For more information, see [“Specify User-Defined Document IDs” on page 611](#).

5. Specify how many documents to pass to Integration Server. For more information, see [“Specify Number of Logged Documents to Give to Integration Server at One Time”](#) on page 612.
6. Configure any additional settings, as needed. For more information, see [“Other Configuration Settings”](#) on page 613.

Enable Document Logging

Use the following procedure to enable document logging on Broker.

To enable document logging on Broker

1. In My webMethods: **Administration > Messaging > Broker Servers > Servers**.
2. Click the name of the Broker Server that hosts the Broker to which Integration Server will connect. On the Broker Server Details page, do the following:
 - a. Click the **Brokers** tab and then click the name of the Broker to which Integration Server will connect.
 - b. Click the Configuration tab, select the **Document logging** check box, and then click **Apply**.

Identify Document Types to Log

Use the following procedure to identify document types you want the connected Broker to log. Specify whether to log the documents on publish or on subscriber acknowledgement.

To identify document types to log

1. In My webMethods: **Administration > Messaging > Broker Servers > Clients**.
2. In the Client List, click the client ID of the Broker client (*<Integration Server client prefix>_LogOnPublish* and *<Integration Server client prefix>_LogOnAck*) for which you want to register document type subscriptions. If the Broker client does not appear in the list, use the **Search** tab to locate it.
3. On the Client Details page, click the **Subscriptions** tab. You will see a list of all the document types to which the Broker client has "cansubscribe" permissions.
4. Select the check box next to each document type to which you want the Broker client to subscribe.
5. Click **Subscribe**. On the **Subscription** tab, My webMethods displays a **"Yes"** in the **Subscription** column for each document type to which the Broker client subscribes.

Subscribe to Document Types from Other Brokers in the Territory or Cluster

Use the following procedure to subscribe to document types from other Brokers.

To subscribe to document types from other Brokers

1. For each of the non-connected Broker in the territory or cluster for which to log documents, do the following:
 - a. In My webMethods: **Administration > Messaging > Broker Servers > Servers**.
 - b. Click the name of the Broker Server that hosts one of the Brokers. On the Broker Server Details page, do the following:
 - a. Click the **Brokers** tab and then click the name of the Broker to which Integration Server will connect.
 - b. Click the **Configuration** tab, clear the **Document logging** check box, and then click **Apply**.
 - c. Use the procedure in [“Identify Document Types to Log” on page 610](#) to identify the document types to log for the Broker.
2. On the Integration Server that hosts the Logging Utility, create triggers that subscribe to the documents to log. For instructions, see the *Publish-Subscribe Developer’s Guide* .

Specify User-Defined Document IDs

Use the following procedure to specify user-defined document IDs.

To specify user-defined document IDs

1. In Integration Server Administrator, go to the **Settings > Extended** page.
2. Click **Edit Extended Settings**.
3. In the **Extended Settings** box, set the `watt.server.auditDocIdField` property as follows:

```
watt.server.auditDocIdField=user_defined_field
```

user_defined_field must be of type Broker unicode string and can be up to 128 characters long. *user_defined_field* is case sensitive.

- For BrokerEvents published using the Java client API, define *user_defined_field* in one of the following ways:

- `watt.server.auditDocIdField=_env.field_name`, where *field_name* is an envelope field. For example, if `uuid` is set in the envelope, set `watt.server.auditDocIdField` as follows:

```
watt.server.auditDocIdField=_env.uuid
```

- `watt.server.auditDocIdField=field_name`, where *field_name* is a body field. For example, if SSN is set in the body, set `watt.server.auditDocIdField` as follows:


```
watt.server.auditDocIdField=SSN
```
 - For JMS messages published using the JMS API, define *user_defined_field* in one of the following ways:
 - `watt.server.auditDocIdField=_env.field_name`, where *field_name* is an envelope field. For example, if `Message.setJMSMessageID(String id)` is set in the envelope, set `watt.server.auditDocIdField` as follows:


```
watt.server.auditDocIdField=_env.jms_message_id
```
 - `watt.server.auditDocIdField=Properties.field_name`, where *field_name* is a property of the message. For example, if `Message.setStringProperty(customId, XYZ)` is set on the message before publishing, set `watt.server.auditDocIdField` as follows:


```
watt.server.auditDocIdField=Properties.customId
```
4. Click **Save Changes**, and then restart Integration Server.
 5. Add *user_defined_field* to your document types, as described in [“Identify Document Types to Log” on page 610](#).

Specify Number of Logged Documents to Give to Integration Server at One Time

Use the following procedure to specify the number of logged documents to pass to Integration Server at one time.

To specify the number of logged documents to pass to Integration Server at one time

1. In Integration Server Administrator, go to the **Packages > Management** page.
2. In the **Packages** list, click  in the row for the WmLogUtil package. Integration Server Administrator displays the Logging Utility home page.
3. Go to the **Settings > Configure** page and click **Change Settings**.
4. In the **Size** box, type the number of logged documents for the Logging Utility to remove from the Broker log queue and pass to Integration Server at one time. You can improve performance by retrieving a large number of documents from a Broker log queue at one time. However, too large a batch can degrade performance. The number of documents you retrieve at one time depends on environment-specific factors such as the amount of RAM on the Broker host machine, your database, and the size of the logged documents. To determine the optimum batch size, run an activation and tune the number.

Note: Although the Logging Utility allows you to specify any value for batch size, Broker can retrieve a maximum of 160 documents.

5. Click **Save Changes**.

Other Configuration Settings

You can configure the `SharedEventOrdering` and `BrokerTimeout` parameters in the `WmLogUtil\config\LoggingUtility.cnf` file.

Parameter	Description
<code>SharedEventOrdering</code>	<p>Specifies how the documents will be arranged. To handle high document volume in a clustered Integration Server environment, set the <code>SharedEventOrdering</code> parameter to <code>none</code>. This will allow multiple Integration Servers to process the logged documents in parallel. Default value is <code>Publisher</code>. Allowed values are <code>Publisher</code> and <code>none</code>.</p> <p>For example, if you want Integration Servers to process the documents ordered by publishers, set <code>SharedEventOrdering=Publisher</code>.</p>
<code>BrokerTimeout</code>	<p>Specifies how long (in milliseconds) Integration Server will wait before timing out the Broker. The default value is 30 seconds.</p> <p>For example, to configure the Broker timeout value to one minute, set <code>BrokerTimeout=600000</code></p>

Logging Utility Management

The Logging Utility starts automatically when you start Integration Server. The Logging Utility home page shows the utility's status.

Managing the Logging Utility

Use the following procedure to manage the Logging Utility.

To manage the Logging Utility

1. To open the Logging Utility home page, in Integration Server Administrator, go to the **Packages > Management** page.
2. In the **Packages** list, click  in the row for the WmLogUtil package. Integration Server Administrator displays the Logging Utility home page. The **Status** column shows the utility's current status. You can change that status as follows:

Task	Description
Start	Starts the Logging Utility. The utility begins pulling logged documents from the Broker log queue.
Stop	Stops the Logging Utility after completely processing any logged documents the utility has already pulled from the Broker log queue. The value in the Status column changes to Stop Pending . Note: Depending on your system, you might have to click Refresh to see when the status changes to Stop .
Suspend	Pauses the Logging Utility after completely processing any logged documents the utility has already pulled from the Broker log queue. The value in the Status column changes to Suspend Pending . Note: Depending on your system, you might have to click Refresh to see when the status changes to Suspended .
Resume	Resumes the suspended Logging Utility.
Refresh	Updates the information in the Status column.

3. If you want the Logging Utility to write debug entries in the Integration Server server log, click the word **Off** in the Debug Trace column. The word changes to **On**.

Built-In Services

You can use Logging Utility built-in services to work with the log queue of the Broker that is connected to the Logging Utility-equipped Integration Server. You can use these services to check the number of documents in the queue, view problematic documents in the queue, or delete a problematic document from the queue. The services below are available in the WmLogUtil package:

Service	Description
pub.loggingUtility.util:brokerLoggingLength	Retrieves the number of logged documents in the Broker's log queue.
pub.loggingUtility.util:brokerLoggingPeek	Displays logged documents in the Broker's log queue.
pub.loggingUtility.util:brokerLoggingSingleRemove	Removes a logged document from the Broker's log queue.

pub.loggingUtility.util:brokerLoggingLength

Retrieves the number of logged documents in the Broker's log queue.

Input Parameters

None.

Output Parameters

LoggingLength **String** Number of logged documents in the Broker's log queue.

Usage Notes

To use this service, run it directly from Software AG Designer and view the results on Designer's **Service Result** tab. Do not call the service from Integration Server services at run time.

pub.loggingUtility.util:brokerLoggingPeek

Displays logged documents in the Broker's log queue.

Input Parameters

NumberToPeek **String** Number of logged documents to view, starting with the oldest logged document in the log queue.

Output Parameters

BrokerEventStrings **String List** Collection of logged documents. Each element in BrokerEventStrings contains an entire logged document represented in String form.

Usage Notes

You might use this service to locate a problematic logged document.

To use this service, run it directly from Designer and view the results on Designer's **Service Result** tab. Do not call the service from Integration Server services at run time.

pub.loggingUtility.util:brokerLoggingSingleExtract

Removes a logged document from the Broker's log queue.

Input Parameters

DeleteEvent **String** Indicates whether to remove the oldest logged document from the log queue. Values are y, yes, true, or on and are case insensitive.

Output Parameters

BrokerEventString **String** Removed logged document in string form.

Usage Notes

To use this service, run it directly from Designer and view the results on Designer's **Service Result** tab. Do not call the service from Integration Server services at run time.

H Configuring My webMethods Server to Use the Embedded Database

■ Overview	618
■ Configuring My webMethods Server to Use Derby Database	619

Overview

This appendix explains how to configure My webMethods Server to use the embedded Apache Derby database.

Note: The facility to configure My webMethods Server using the embedded database Apache Derby is removed from My webMethods Server 10.3 and later. Choose an appropriate database when you install My webMethods Server. For information on configuring My webMethods Server, see *Administering My webMethods Server*.

You administer webMethods Broker by using My webMethods Server, which generally requires an RDBMS, such as Oracle or SQL Server. For complete information about configuring My webMethods Server to use an external RDBMS, see *Installing webMethods and Intelligent Business Operations Products*.

If you use My webMethods for the sole purpose of administering webMethods Broker, you have the option to run My webMethods by using the embedded database, Apache Derby. By using the embedded database you avoid the need for an external RDBMS.

When using the embedded Derby database, you have certain advantages. Derby is light-weight, open source, and is easy to install, deploy, and use.

Important: If you use the embedded database, and later want to use My webMethods Server to administer additional Software AG webMethods products, you will need to configure My webMethods Server to use an external RDBMS. Additionally, the webMethods Broker user, access rights, and directory configuration information that is stored in the embedded database cannot be migrated to the new external RDBMS.

For more information about configuring My webMethods Server to use Derby, see [“Configuring My webMethods Server to Use Derby Database” on page 619](#).

Running Multiple My webMethods Server Instances

You can run multiple instances of My webMethods Server on the same machine, each using the same database or each using a different, supported database. Therefore, you can run multiple My webMethods Server instances, all using the same embedded Derby database. For more information about the guidelines for multiple My webMethods Server instances, see *Administering My webMethods Server*.

You can control which My webMethods Server instance to use by running the `mws.bat` (Windows) or `mws.sh` (UNIX) command, specifying the name of the server instance, and a command keyword. For more information about the command syntax for My webMethods Server, see *Administering My webMethods Server*.

Configuring My webMethods Server to Use Derby Database

You can configure My webMethods Server to use the embedded Apache Derby database either when you install or after you install My webMethods Server.

- When you install My webMethods Server, see [“Configuring to use Derby when you install My webMethods Server”](#) on page 619 for the procedure to use Derby.
- After you install My webMethods Server, see [“Configuring to use Derby after you install My webMethods Server”](#) on page 619 for the procedure to create a new My webMethods Server instance that uses the Derby database.

Configuring to use Derby when you install My webMethods Server

To configure Derby when you install My webMethods Server

1. Use Software AG Installer to install My webMethods Server.
2. For the database connection, choose the **Embedded Database** option.
3. Complete the My webMethods Server installation. For more information about the installation process, see *Installing webMethods and Intelligent Business Operations Products*.

Configuring to use Derby after you install My webMethods Server

To use Derby database after you install My webMethods Server

1. Create a new My webMethods Server instance that uses the Derby database:
 - a. At the command line, type the following command to move to the My webMethods Server's bin directory:

```
cd Software AG_directory\MWS\bin
```

- b. Create a new server instance by running the following command:

```
mws new -Dserver.name=<MWS instance name \> -Dhttp.port=<Port number \>
```

where *MWS instance name* is any name for the new My webMethods Server instance and *Port number* is the port on which you want to run My webMethods Server.

In UNIX, after running the above mentioned command, if you are unable to create a new My webMethods Server instance that uses the embedded Derby database, run the command by explicitly specifying the database type and the database URL as shown:

```
mws new -Dserver.name=<MWS instance name \> -Dhttp.port=<Port number \>  
-Ddb.type=derby -Ddb.url=jdbc:derby:home:/data/db
```

For complete information about creating a new server instance, see *Administering My webMethods Server*.

2. Make sure that the *Software AG_directory \MWS\server* folder contains the *template-derby.zip* file. The *template-derby.zip* file contains all the necessary Derby files and the table structure for My webMethods Server.
3. Start My webMethods Server by following the steps provided in *Administering My webMethods Server* . When starting for the first time, My webMethods Server takes a considerable amount of time. If you can see the user interface screens, your installation of My webMethods Server was successful.
4. Make sure that My webMethods Server is using Derby. To do so, verify that the value of URL in the *My webMethods Server_directory \MWS\server\<MWS instance name \>\config\ mws.db.xml* file is pointing to:

```
jdbc:derby:home:/data/db
```

I webMethods Broker Server Configuration Parameters

■ Introduction	622
■ Broker Server Configuration Parameters	622

Introduction

The Broker Server configuration file (awbroker.cfg) in the Broker Server's data directory contains parameters that define a single Broker Server instance.

If your awbroker.cfg configuration file does not include the parameters required for your messaging system setup, add the parameters to the file, and set the values.

Broker Server Configuration Parameters

allow-unsafe-renegotiation

Specify 1 to force the Broker to allow legacy insecure renegotiation between the SSL enabled server and client. It is recommended you disable insecure renegotiation. Enabling insecure renegotiation may cause security vulnerability in the client-server applications. The default is 0 (disabled).

allow-pre-acked-on-browse

Specify 0 to disable allowing the tentatively pre-acknowledged documents . Default is 1 (enabled).

assert-config

Specify a comma separated list of options to set the assert options in the following format:

```
assert-config=option=value,option=value
```

The *value* for an *option* can be either 0 or 1.

List of assert options you can set:

- crash-checks
- crash-preconditions
- crash-expensivechecks
- crash-postconditions
- never-crash
- never-report
- report-checks
- report-expected
- report-expensivechecks
- report-preconditions
- report-postconditions

If you enable the report related assert options, Broker writes the assert messages to the Broker error log file (diagnostic log).

If you enable the crash related assert options, Broker performs additional checks. Broker crashes and creates core files if any of the run-time checks fail.

If you set `never-report` to 1, Broker will override all the report related assert options. If you set `never-crash` to 1, Broker will override all the crash related assert options. For example, if `never-crash=1`, `crash-preconditions=1`, Broker will not crash if a pre-condition run-time check fails and will not write the assert messages to the Broker error log file.

Types of checks that Broker can perform during run-time:

- Pre-condition check on the data incoming to a function
- Expected check on the expected value of a run-time data
- Post-condition check on a function just before returning
- Expensive check or process intensive checks on the correctness of the run-time data. Process intensive checks affects the performance.

The default is `never-report=0`, `never-crash=1`, `report-preconditions=1`, `crash-preconditions=1`, `report-checks=1`, `crash-checks=1`, `report-expected=1`, `report-expensivechecks=1`, `crash-expensivechecks = 1`, `report-postconditions=1`, `crash-postconditions=1`.

async

This argument specifies the Broker Server to write data to disk asynchronously. To enable the asynchronous write mode, specify the `async` argument in the storage session parameters, `session-config` and `session-data`. For example, `sessionconfig=qs:///var/opt/BrokerStorage/BrokerConfig.qs?async`

auditlog-dir

Specify the location and the name of the audit log directory. If you change the file name or location, include the full path. If you have enabled audit logging, but did not specify the audit log directory, Broker creates the audit logs in the `auditlogs` directory under the Broker Server's data directory.

auditlog-enabled

Specify 0 to disable audit logging. The default is 1 (enabled).

auditlog-errorAction

Specify 1 to enable the audit logging error action. This parameter specifies whether or not to perform an operation if there is an error during audit logging. The default is 0 (disabled). For example, consider that the server audit logging is enabled (`auditlog-enabled=1`), `auditlog-errorAction=1`, and the territory modification operation is defined for audit logging (`auditlog-operations=10`). If for some reason the territory modification operation cannot be logged due to audit log file failure, directory creation failure, or any other error, the territory modification operation will be discarded because audit logging is not possible. If you set `auditlog-errorAction=0`, irrespective of the audit logging success or failure, the territory modification operation will be performed.

auditlog-maxFileSize

Specify the maximum size of audit log that the Broker Server can create for audit logging before it creates a new audit log. You can specify the values in kilobytes (K), megabytes (M), and gigabytes (G). For example,

```
auditlog-maxFileSize=200K
auditlog-maxFileSize=20M
auditlog-maxFileSize=2G
```

The maximum log file size is 2 gigabytes. The default is 20M (20 megabytes).

auditlog-maxWriteCache

Specify the number of audit records that can be cached in memory before they are written to the audit log file. The audit records are flushed from the cache and written to the audit file at a regular time interval specified by the `auditlog-writeInterval` parameter or when the number of audit records cached reaches the value specified in the `auditlog-maxWriteCache` parameter. When the Broker Server shuts down, all the audit records collected are written to the log, regardless of the cache size. You can disable the audit log caching by setting this parameter to 1. If you disable the audit log caching, Broker writes the audit records to the audit log file as and when they are generated. The default is 500.

auditlog-operations

Specify the server operations you want to audit when audit logging is enabled.

```
CONNECTION = 0
SESSION = 1
SERVER MODIFY = 2
SERVER SECURITY = 3
SERVER START STOP = 4
BROKER MODIFY = 5
DOCTYPE MODIFY = 6
CLIENTGROUP MODIFY = 7
CLIENTGROUP SECURITY = 8
GATEWAY MODIFY = 9
TERRITORY MODIFY = 10
CLIENT QUEUE = 11
```

For example, if `auditlog-operations=1 4 6`, the audit logging for session creation/deletion, server start/stop, and event-type modifications will be enabled. Other options such as gateway modifications and document type modifications will not be audited by the server. The default values are 2 3 4 5 6 7 8 9 10 11. Only the Broker Server connections and sessions are not logged by default.

auditlog-writeInterval

Specify the time interval in seconds for flushing the audit records from the cache and writing these records to the audit log file. The default is 30 seconds.

auto-restart

Specify 1 to enable automatic Broker Server restart by Broker Monitor whenever Broker Server goes down. Specify 0 to disable automatic Broker Server restart. The option to disable automatic Broker Server restart is useful if you want to collect the diagnostic information before restarting the Broker Server that had abruptly stopped. The default is 1 (enabled).

batchmode

Specifies the batch mode for the asynchronous disk writes. The default value for `batchmode` depends on the `async` argument. If you specify asynchronous write, batching is disabled. If you do not specify asynchronous write, Broker's heuristic is used for batching. You can configure the storage sessions to write to disks in batches by specifying the storage session URL and the batch mode in the `session-config` or `session-data` parameters.

Valid values for `batchmode` are:

- 1 - Enable batching
- 2 - Disable batching
- 3 - Use Broker's heuristic

For example, if `session-data=qs:///var/opt/BrokerStorage/BrokerData.qs?async,max_cache_size=512`, asynchronous write is enabled and the `batchmode` is not set. In this case, Broker disables batching.

If `session-data=qs:///var/opt/BrokerStorage/BrokerData.qs?fast_restart=10,max_cache_size=512`, asynchronous write and `batchmode` are not set. In this case, Broker's heuristic is used.

If `sessiondata=qs:///var/opt/BrokerStorage/BrokerData.qs?batchmode=1`, batching is always on.

basic-auth-cfg-file

Specify the location of the basic authentication configuration file, `basicauth.cfg`.

For example, set `basic-auth-cfg-file=webMethods Broker_directory\data\awbrokersversion\default\basicauth.cfg`.

broker-ipaddress

Specify the IP address for Broker Server port binding. By default, the Broker Server ports bind to all the available IP interfaces on the system.

bypass-authentication

Specify 1 to temporarily bypass checking user credentials to connect to the Broker Server. Default is 0.

cbtrace-size

Specify the number of Broker trace records to be dumped. Broker traces are runtime traces generated by Broker. These traces are automatically written to `cbtrace.store` file whenever a Broker process stops abruptly. The default is 2000.

check-crl-all

Specify 1 to enable CRL (Certificate Revocation List) checking for the entire certificate chain in the SSL certificate. The default is 0 (disabled). When `check-crl-all=0`, only the first certificate is checked against CRL.

confirmed-pre-acknowledgement-timeout

Specify the maximum time in seconds a Broker in a cluster holds a confirmed pre-acknowledgement before discarding it. This parameter setting is required for cleaning

up the accumulated confirmed pre-acknowledgements in a cluster Broker, which never receives the published messages. The default is 600 (10 minutes).

connection-rcvbuf-size

Specify the TCP/IP receive buffer size in bytes. The maximum and minimum values are defined by the TCP/IP receive buffer size supported by the host machine of the Broker Server. Increasing this value may result in better performance on a high latency network. The default value is either the system default value or 524288 (512 KB), whichever value is higher.

connection-sndbuf-size

Specify the TCP/IP send buffer size in bytes. The maximum and minimum values are defined by the TCP/IP send buffer size supported by the host machine of the Broker Server. Increasing this value may result in better performance on high latency network. The default value is either the system default value or 524288 (512 KB), whichever value is higher.

corefiles-retained

Specify the limit for retaining the core files of the Broker Server in case of a crash. The recommended number is less than three because the core files consume a lot storage space. If the Broker Server is frequently dumping core, the production or test environment might cause shortage of storage space.

description

Specify a brief description for the Broker Server, if required. The default is an empty string.

diag

Specify 1 to enable diagnostic logging. It is recommended that you enable diagnostics logging only when you have to monitor the Broker Server behavior or troubleshoot an abnormal behavior of Broker Server. Use the `extra-args` and `redirect-output` arguments for diagnostic purpose. The default is 0 (disabled).

disable-activity-traces

Specify 1 to disable the activity traces of Broker. The default is 0 (the activity traces are enabled).

enable-fips-mode

Specify 1 to enable FIPS mode for Broker Server on the supported platforms. The default is 0 (disabled).

event-checksum

Specify 1 to force Broker to perform event checksum during publish. The default is 0 (disabled).

eventlog

Specify 0 to disable event logging on Windows. The default is 1 (enabled). View the Windows event log through the Windows Event Viewer.

executable

Specify the location of the Broker Server binary. It is recommended that you update the change in the Broker binary location using the `server_config` utility. If you edit

this parameter, make sure you update the awbrokermon.cfg file to avoid unpredictable behavior of Broker. To update the location of the Broker binary, remove the Broker Server (do not delete), then add (do not create) the Broker Server with the `-e <new full path of the binary>` option to update the new location of the Broker binary. For more information, see “[Broker Server Configuration Utility](#)”.

extra-args

This parameter is used to pass the command line arguments. You can use this parameter to pass a command line arguments through the awbroker.cfg file to enable diagnostic and debugging.

Use `extra-args` along with `redirect-output` to collect information about Broker operations for diagnostic purpose in the file specified by `redirect-output`. The table below lists the common arguments of `extra-args`.

Argument Name	Specifies...
<code>cqueue</code>	Client queue operations
<code>deliver</code>	Delivery operations
<code>event</code>	Events
<code>fqueue</code>	Forward queue operations
<code>mlop</code>	Multi-Broker protocols
<code>mbconnect</code>	Multi-Broker connect events
<code>mbjoin</code>	Territory or cluster join operations
<code>mbleave</code>	Territory or cluster leave operations
<code>mbreach</code>	Multi-Broker connectivity
<code>operation</code>	API calls
<code>operationv6</code>	JMS and C# API calls
<code>pio</code>	Parallel channel diagnostic information
<code>preack</code>	Pre-acknowledgements
<code>result</code>	API call result

Argument Name	Specifies...
resultv6	JMS and C# API call result
connect	TCP/IP socket connect operation
ssl	Monitoring SSL connect and disconnect operations
awssl	Monitoring SSL handshake operations
timer	Broker timer setting
tx	Transactions

For example, if you want to enable the JMS API diagnostic traces with SSL monitoring, set `extra-args=-blah +operationv6 +resultv6 +ssl +awssl`

fast_restart

Specifies the portion of the log file that must be played during run-time so that Broker restart is faster. Broker stores the guaranteed documents in the log files before storing these documents in the disk. Whenever Broker restarts, it must first play the log file. If the log file contains a lot of uncommitted data, storing this data will take a long time.

For example, if the log file size is 1GB, on an average, Broker will move 512MB of data from the log file to the disk every time Broker restarts. You can use the `fast_restart` argument to specify Broker to write the log file data to the store file more often so that during the next restart, there is fewer data to be moved from the log file to the store file.

Let us consider that the log file size is 1GB, and you have set `fast_restart=4`, then the entire log file will be played in four parts. That is Broker writes only 250 MB (1GB/4) of data to the disk at a time. In this case, at any given time, there will not be more than 250 MB of uncommitted data remaining in the log file during the next Broker restart. Broker would need to write 128MB of data on an average on each restart.

Set the `fast_restart` argument as shown below:

```
session-data=qs:///var/opt/BrokerStorage/BrokerData.qs?async,fast_restart=10
```

Be cautious while setting this argument. When the throughput of guaranteed documents is high, if you configure a small log file to play very often (for example, `fast_restart=100` for a 512MB log file), you do not optimize on the usage of the argument. By default, `fast_restart` is disabled.

file-encoding

Specify 1 to encode the configuration file. By default, there is no encoding.

filter-collation-locale

Specify the locale, which the Broker Server should use while comparing strings during subscription filter evaluation. If you specify null, no locale is used. By default, Broker Server uses the system locale.

forget-lost-transaction-timer-interval-seconds

Specify the interval in seconds between two checks. The default is 600 seconds (10 minutes).

forget-lost-transaction-time-days

Specify the age in days when transactions was heuristically completed. The default is 10 days.

forget-lost-transaction-batch-size

Specify the number of heuristically completed transactions to be deleted in one attempt. Default is 100.

gateway-broker-req-events

Specify the limit for the number of events that can be batched for a gateway exchange. This value should not be more than the value of `max-recv-events`. The default is 20. Increasing the batch size might help the throughput if small documents are transferred across the (WAN Wide Area Network). Only if you have to transfer very large documents between Brokers, it is recommended that you set the batch size to a small value.

hostname

Specify the name of the host machine on which the Broker Server is installed. The default is `localhost`. This option overrides the typical reverse IP lookup for the system. The configured Broker presents its `hostname` to the requesting Broker for multi-Broker operations and the reverse host name lookup is not performed.

internal

Specify 0 to disable the logging of system messages to the internal `logmsgs` file. The default is 1 (enabled).

license-key

Specify the location of the license file. If the license-key file is missing or if it is invalid, Broker Server writes an error to the journal log and exits. You cannot start Broker Server if the license-key file is missing or invalid. If the license has expired, Broker Server will start, but it will not accept any documents that clients publish (clients can continue to retrieve documents that have already been published). Broker Server writes a message to the system log to indicate that it is running under an expired license.

log-alert

Specify 0 to disable the logging of alert messages generated by the server. The default is 1 (enabled).

log-info

Specify 0 to disable the logging of information messages generated by the server. The default is 1 (enabled).

log-warning

Specify 0 to disable the logging of warning messages generated by the server. The default is 1 (enabled).

max_cache_size

Specify the limit for memory that Broker Server can use to cache queue storage data. You define the value for `max_cache_size` using the `session-config` or `session-data` parameters. Setting the `max_cache_size` argument ensures that some of the storage data is cached in memory for faster storage lookup. Increasing the value of `max_cache_size` might improve Broker performance because the storage lookup might occur on the cache rather than on the disk.

memory-alert-threshold

Specify the threshold value in percentage (%) of available memory, if the threshold is exceeded, all publish requests are denied. The default is 90.

memory-warn-threshold

Specify the threshold value in percentage (%) of available memory, if the threshold is exceeded, a warning is logged in the `server.log` file. The default is 80.

max-diag-log-file-size

Specify the maximum size of the diagnostic log file in bytes. The log file is renamed to `diag.log.old` once the maximum log file size is reached. Default value is 10485760 (10 MB).

max-log-file-size

Specify the maximum size of the basic authentication log file in bytes. The log file is renamed with the suffix `.old` once the maximum log file size is reached. The old file name is `basicauth.log.old` unless overridden with a different file name in `basicauth.cfg`. Default value is 10485760 (10 MB).

max-memory-size

Specify the main memory limit in megabytes for the Broker Server to store documents. For example, `max-memory-size=4096` will limit the Broker memory usage to 4GB. When this memory limit is reached, the Broker Server stops accepting documents. When the `max-memory-size` parameter is set, Broker Server monitors the main memory that it is using for document storage. If the size of the incoming document plus the total memory that is already in use exceeds the `max-memory-size` limit, Broker Server rejects the document and returns an "out of memory" error to the client. If `max-memory-size` is not specified (that is, this parameter is not including in the Broker Server's configuration file), Broker Server does not limit the memory usage, and will accept documents even though there might not be sufficient memory to process them.

max-open-transaction-per-client

Specify the number of open transactions that a client can have before Broker rejects creation of any new transaction for the client. Default is 1000.

max-open-transaction-per-broker

Specify the number of open transactions that a Broker can have before Broker rejects creation of any new transaction for any client in Broker. Default is 1000.

max-recv-events

Specify the maximum number of events that can be requested in a single request. The default is 160.

max-send-events-size

Specify the limit for the total size of events that can be sent by the Broker in a single request. The default is 131072 (128 KB).

max-server-log-file-size

Specify the maximum size of the server log file in bytes. Minimum value of this parameter is 1 MB. The default value is 10 MB.

max-worker-threads

Specify the limit the number of worker threads per server running on Windows. The default is 32. This parameter is not applicable on UNIX. On UNIX, Broker Server creates a new worker thread as and when it is required.

port

Specify the base port number of Broker Server. If you want to use SSL, make sure that the ports with numbers *baseport -1* and *baseport -2* are available. The default is 6849. It is recommended that you use *server_config* to change the port (by removing and adding the server). Directly editing the value of this parameter can result in unpredictable behavior on Windows as the port number might be a part of key in Windows Registry.

pre-acknowledgement-timer-interval

Specify the time interval in seconds for checking the pre-acknowledgement timeout status in a cluster Broker. The checks are done until the time specified by the *tentativepre-acknowledgement-timeout* parameter is reached. The timer works only for active sessions. So, if a client is disconnected from the durable queue, the timer does not work. The time-out of tentative pre-acknowledgement works in two phases to account for slow subscribers. Specify a value less than half the value of the *tentativepre-acknowledgement-timeout* parameter. If you specify a very small value, Broker checks the time-out status frequently. If you specify a very large value, Broker might not detect the time-out effectively. The default is 10 (10 seconds).

preallocate-memory

Specify 1 to enable the *preallocate-memory* parameter, if you want Broker Server to allocate the memory specified by *max-memory-size* when it starts. If the Broker Server cannot obtain the specified memory size during startup, it will immediately exit.

pub-net-address-in-deprecated-format

Specify 1 to retrieve the IP address from the *pubNetAddr* envelope field in the old six-byte format (deprecated format). The default is 0, where the IP address from the *pubNetAddr* envelope field is retrieved in the string format.

pub-net-address-in-envelope

Specify 1 to make the Broker insert the publisher's IPv4 or IPv6 address in the event/document envelope field, *pubNetAddr*. The default is 0 (disabled).

queue-cleanup-enable

Specify 1 to enable deletion of expired volatile documents from the client queues and forward queues if the total size of the expired volatile documents is equal to or greater than the memory size specified in the *queue-cleanup-threshold* parameter. If *queue-cleanup-enable=0*, when a client tries to retrieve a document from the queue, Broker

will delete the volatile documents for which the time to live has elapsed. The default is 0 (disabled).

queue-cleanup-threshold

Specify the memory limit (in MB) for storing the expired volatile documents. If the `queue-cleanup-enable` parameter is set to 1, Broker will delete the expired volatile documents when the `queue-cleanup-threshold` limit is reached. The value of the `queue-cleanup-threshold` parameter specifies the total size of the volatile documents in all the queues of the Broker Server instance. For example, to delete the expired volatile documents when their total size exceeds 2500 MB, specify `queue-cleanup-threshold=2500`. When determining the memory size, consider the queue usage. If you provide a very high threshold, the queues might get overloaded until the Broker deletes the expired documents. If you provide a lower threshold, Broker checks the queues frequently even if there are no expired documents in the queues. The minimum value allowed for the `queue-cleanup-threshold` parameter is 1 MB. The default is 1024 (1024 MB).

rbroker-max-send-events-size

Specify in bytes the message file size or the batch size limit that the remote Broker must use while sending messages to other Brokers in one transaction. If the network bandwidth is high, set this parameter to a higher value. The default is 16 MB.

rbroker-parallelchannel-delayed-close

Specify 1 to disable delay in closing parallel channels by two seconds. The default is 1 (enabled).

Specify 0 to disable delay in closing remote broker parallel channels by two seconds. This option is applicable only when remote broker parallel channel is enabled between territory, gateway, or cluster host. The default is 1 (enabled).

rbroker-parallelchannel-count

Specify the number of parallel channels the remote Broker can request while communicating with another Broker. Valid value is an integer greater than 1. The default is 5.

rbroker-parallelchannel-enabled

Specify 1 to enable parallel channels for message communication between a remote Broker in a territory, gateway, or a cluster with another Broker. The default is 0 (disabled).

rbroker-recovery-restart-forwarding-enabled

Specify 1 to enable the Broker Server to reconnect to the remote Broker in case of partial processing of received events. The default is 0 (disabled).

rbroker-select-different-join-broker

By default, the joining Broker selects the most compatible Broker from a territory's remote Broker list during territory join operation. Specify 0 to disable auto-selection; you can then select a Broker of your choice during the territory join operation.

rbroker-streaming-ack-timeout

Specify the maximum time in seconds a Broker must wait before sending an acknowledgement. Set this parameter only if you have enabled streaming between

remote Brokers. The default is 30 seconds. If you set `rbroker-streaming-ack-timeout=0`, the acknowledgement timeout is disabled, the forward queue of the sending Broker might hold a few unacknowledged documents unnecessarily for a longer time, and it would seem as though the documents are stuck. When streaming is enabled between remote Brokers, Broker sends an acknowledgement when either the `broker-streaming-ack-timeout` or `rbroker-streaming-window-size` parameter value is reached.

rbroker-streaming-enabled

Specify 1 to enable streaming in Broker Server. The default is 0 (Streaming is disabled). When streaming is enabled, the number of calls between two Brokers in territory or gateway is reduced. Streaming might help when you are transferring small documents using remote Brokers that are separated by a high latency Wide Area Network (WAN). Increasing the value of the `connection-sndbuf-size` and `connection-rcvbuf-size` parameters might improve the throughput on a high latency network.

rbroker-streaming-window-size

Specify the maximum number of documents a Broker must receive before sending an acknowledgement. Set this parameter only if you have enabled streaming between remote Brokers. Valid values are 1 to 1000. Increasing this value may result in better performance on a high latency network. The default is 1000. When streaming is enabled between remote Brokers, Broker sends an acknowledgement when either the `broker-streaming-ack-timeout` or `rbroker-streaming-window-size` parameter value is reached.

redirect-output

Specify the name (not the full path) of the log file to which the Broker Server's run-time console logs must be redirected.

remote-broker-req-events

Specify the limit for the number of requested events that can be batched during a remote Broker exchange. This value should not be more than the value of `max-recv-events`. The default is 20.

session-config

Specify the URL for the storage session configuration. You can specify additional arguments such as `async`, `batchmode`, `fast_restart`, and `max_cache_size`. For example, `session-config=qs:///var/opt/BrokerStorage/BrokerConfig.qs?async`

session-data

Specify the URL for the storage session data. You can specify additional arguments such as `async`, `batchmode`, `fast_restart`, and `max_cache_size`. For example, `session-data=qs:///var/opt/BrokerStorage/BrokerData.qs?max_cache_size=512,async`

snmp

Specify 1 to enable SNMP traps. The default is 0 (disabled).

ssl-accept-timeout

Specify the time-out time in milliseconds while accepting new SSL connections. The default is 5000 (5 seconds).

stop-queue-scan-on-pubid

Specify 0 if you want the Broker to continue scanning a queue even though a session encounters a document that cannot be retrieved because of the "order by publisher" condition. It is recommended that you specify `stop-queue-scan-on-pubid=1` to avoid performance issues while scanning large queues. Default is 1 (enabled). For example, consider a Broker queue contains Doc1(Publisher1), Doc2(Publisher2), Doc3(Publisher1), Doc4(Publisher2)... If client1 retrieves Doc1 and Doc2 and if the acknowledgement is pending for these documents, client2 will stop scanning the queue at Doc3 if `stop-queue-scan-on-pubid=1`. If `stop-queue-scan-on-pubid=0`, client2 would continue scanning the queue looking for a document that is not published by the publishers already being catered by client1.

storage-max-cache-size

This parameter has been deprecated. If `storage-max-cache-size` parameter is specified in the configuration file, it will be overridden by the value specified by `max_cache_size`.

storage-alert-threshold

Specify threshold value in percentage (%) of available storage, if the threshold is exceeded, all publish requests are denied. The default is 90.

storage-warn-threshold

Specify threshold value in percentage (%) of available storage, if the threshold is exceeded, a warning is logged in the `server.log` file. The default is 80.

syslog

Specify 0 to disable system logging on UNIX. The default is 1 (enabled). To understand the syslog daemon configuration and to determine the location of the log file, see the man page of "syslogd" for AIX, HP-UX, Solaris and Red Hat Enterprise Linux, or the man page of "syslog-ng" for SUSE Enterprise Server.

syslog-facility

Specify the system log facility on UNIX for the Broker Server. The valid values range from `local0` to `local7`.

tentative-pre-acknowledgement-timeout

Specify the minimum time in seconds the Brokers in the cluster will hold the published message before sending it to the subscriber. If a Broker does not receive the confirmed message pre-acknowledgement before the specified `tentative-preacknowledgement-timeout` time, the Broker forwards the published message to the subscriber. If a Broker receives the confirmed pre-acknowledgement before the specified `tentative-preacknowledgement-timeout` time, Broker deletes the published message from the client queue. The default is 20 (20 seconds).

use-localhost

Specify 1 to set the hostname of all the remote territory Brokers and remote gateway Brokers to the local server, and use the local server for troubleshooting territory/gateway issues. This parameter is used for running all the Broker Servers on localhost for diagnostic purpose only. The default is 0.

verify-ssl-certificate-purpose

Specify 1 to enable the SSL certificate purpose check on the Broker. The default is 0.

version

Specify the version of webMethods Broker. For example, if you have installed webMethods Broker 9.6, set `version=9.6`.

worker-timeout

Specify the time in milliseconds for the connection worker thread to timeout. The default is 100 milliseconds. This timeout indicates how long the worker thread will wait for new data on the connection before releasing the connection and going back to the worker thread pool. Specify a large value for this parameter for large document messaging or when the client response is slow.

J Broker Monitor Configuration Parameters

■ Introduction	638
■ Broker Monitor Configuration Parameters	638

Introduction

The Broker Monitor's configuration file (`awbrokermon.cfg`) residing in the `webMethods Broker_directory\bin` folder specifies information such as Broker Monitor's base port and IP address, the Broker Servers that reside on the host machine, and the logging parameters.

The `awbrokermon.cfg` file is updated automatically when you install Broker Server on the host machine, and when you define additional instances of Broker Server using the `server_config` utility.

Broker Monitor Configuration Parameters

monitor-alias

Specifies an automatically generated serial number for the Broker Monitor instance when multiple installations of Broker Server and Broker Monitor of the same versions are running. Windows does not display the instance number for the first instance of the Broker Monitor service. Subsequent instances of the service are numbered starting from (2). Do not edit this value.

monitor-allowed-client-ipaddress-list

Specifies a comma separated list of IPv4 and IPv6 addresses of the Broker Monitor clients. Provide the actual IP addresses of all the clients you wish to allow access to Broker Monitor. To configure Broker Monitor access for the localhost, provide the loopback IP address and link-local address (if any) along with the actual IP address of the localhost. For example, to specify the addresses of the localhost, set `monitor-allowed-client-ipaddress-list=10.20.47.71,127.0.0.1,2a00:2000:4061::e:134,::1,fe80::250:56ff:fe9d:45f8`. If this parameter contains IP addresses, only the clients specified in the IP address list can access the Broker Monitor. If you do not specify any IP address in this parameter, there is no restriction on any clients for administering the Broker Monitor and the corresponding Broker Servers. By default, no value is assigned to this parameter.

monitor-ipaddress

Specify the IP address for Broker Monitor port binding. By default, the Broker Monitor port binds to all the available IP interfaces on the system.

monitor-log-alert

Specify 0 to disable or 1 to enable logging of alert messages. The default is 1.

monitor-log-eventlog

Specify 0 to disable or 1 to enable logging of messages to the Windows Event Log Service. The default is 1 on Windows host machines.

monitor-log-info

Specify 0 to disable or 1 to enable logging of informational messages. The default is 1.

monitor-log-snmp

Specify 0 to disable or 1 to enable generation of SNMP traps. The default is 0.

monitor-log-syslog

Specify 0 to disable or 1 to enable logging of messages to the UNIX syslog. The default is 1 on UNIX host machines.

monitor-log-syslog-facility

Specify the UNIX syslog facility to which the messages are to be sent. The default is `local5`.

monitor-log-warning

Specify 0 to disable or 1 to enable logging of warning messages. The default is 1.

monitor-max-broker-start-delay

Specify the maximum time in seconds Broker Monitor must wait for a monitored Broker Server to start before reporting operation failure. The default is 30 (Seconds).

monitor-port

Specifies the Broker Monitor port.

monitor-process-creation

Specify `true` if you want Broker to start as a Windows application. The default is `false` (Broker starts as a Windows service). This parameter is specific to Windows.

monitor-start-servers

Specify `no` if you do not want Broker Monitor to restart any of the monitored Broker Servers when the Broker Monitor restarts or when any of the monitored Broker Servers go down. If `monitor-start-servers=no`, and if a monitored Broker Server with `auto-restart=1` stops abnormally, Broker Monitor will not attempt to re-start that Broker Server. The default is `yes`.

server-<broker_port>.datadir

Specifies the data directory of the Broker Server that is using the port specified by `<broker_port>`. This parameter is created automatically for each Broker Server monitored by the Broker Monitor. Edit this parameter value only if you re-configure the Broker Server.

server-<broker_port>.exec

Specifies the full path of the executable of the Broker Server that is using the port specified by `<broker_port>`. This parameter is created automatically for each Broker Server monitored by the Broker Monitor. Edit this parameter value only if you re-configure the Broker Server.

server-<broker_port>.port

Specifies the Broker Server port. This parameter is created automatically for each Broker Server monitored by the Broker Monitor. Edit this parameter value only if you re-configure the corresponding Broker Server.

server-<broker_port>.version

Specifies the version of the Broker Server that is using the port specified by `<broker_port>`. This parameter is created automatically for each Broker Server

monitored by the Broker Monitor. Edit this parameter value only if you re-configure the corresponding Broker Server.