IBM Tivoli Composite Application Manager for Applications
Version 7.3

WebSphere Message Broker Monitoring Agent User's Guide

SC14-7524-01

IBM Tivoli Composite Application Manager for Applications
Version 7.3

WebSphere Message Broker Monitoring Agent User's Guide

SC14-7524-01

Note
Before using this information and the product it supports, read the information in "Notices" on page 143.

This edition applies to version 7.3 of WebSphere Message Broker Monitoring agent (product number 5724-V09 on Windows, UNIX, and Linux systems; product number 5698-B23 on z/OS systems) and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

Figures

Tables

Chapter 1. Getting started

The WebSphere® Message Broker Monitoring agent is a monitoring and management tool that provides you with the means to verify, analyze, and tune message broker topologies associated with the following WebSphere products:

- IBM® WebSphere Message Broker V7
- IBM WebSphere Message Broker V8
- IBM Integration Bus V9

Important:

- Whenever you want to upgrade WebSphere MQ or WebSphere Message Broker in your environment, you must stop the WebSphere Message Broker Monitoring agent before the upgrade. Otherwise, the monitoring agent cannot connect to the queue manager after the upgrade.
- On Windows systems, after you upgrade WebSphere MQ to version 7.1 or later, you must add the WebSphere MQ installation path to the PATH system variable. Otherwise, the monitoring agent will become offline after WebSphere MQ upgrade.

New in this release

Version 7.3 of WebSphere Message Broker Monitoring agent has the following changes and enhancements:

- Support IBM Integration Bus V9.0 (also known as WebSphere Message Broker)

WebSphere Message Broker Monitoring agent

You can use the WebSphere Message Broker Monitoring agent to ensure the reliability and performance of your broker environment by detecting and correcting broker and message flow problems before they have an impact on service speed and availability. The WebSphere Message Broker Monitoring agent also reduces the amount of time it takes to deploy broker applications by helping you to debug message flows and providing statistics that you can use to tune your environment.

You can use the WebSphere Message Broker Monitoring agent to do the following tasks:

- Monitor the status of your IBM broker product and its components
- View information and performance statistics for broker topologies at broker, execution group, message flow, node, terminal, and thread level in both tabular and chart forms
- Issue IBM broker product commands using the WebSphere Message Broker Monitoring agent interface to directly manage your environment, or create automatic responses to WebSphere Message Broker events
- Receive alerts when performance thresholds are exceeded or when message flow events occur
- Retain data samples in history files and save them to a historical database for reporting and analysis purposes

The WebSphere Message Broker Monitoring agent collects data from WebSphere Message Brokers. The data is presented in charts and tables that you can examine to monitor the performance of your WebSphere Business Integration systems. The agents also evaluate the data to determine when specified values meet the criteria that you have defined, and trigger alerts or programmed actions in response.

In addition, the WebSphere Message Broker Monitoring agent provides a CandleMonitor node. When inserted into a message flow, the CandleMonitor node collects message flow and subflow performance statistics and provides a mechanism for generating user-defined events. For the following workspaces to contain data, the broker must have at least one deployed CandleMonitor node (For information about how to deploy a CandleMonitor node in a broker, see Chapter 3, "Monitoring with the CandleMonitor node," on page 37).

Workspaces and views

The *workspace* is the working area of the application window. The workspace provides you with status, definition, and statistical information in tabular and graphical form.

The workspaces predefined by the WebSphere Message Broker Monitoring agent provide the following information:

- Status information for brokers, components, and agents
- Information about brokers, execution groups, message flows, and message processing nodes
- Information about product, broker, and message flow events
- Information about the environment publish and subscribe configuration
- Performance statistics organized by broker, execution group, message flow, and subflow
- Accounting statistics organized by message flow, thread, node, and terminal

You can use the information that is provided by these workspaces to trace the causes of performance problems or the reasons why an alert is triggered.

You can also customize these workspaces and the views that they contain, or create your own workspaces and views to display information about a specific set of attributes.

For information about predefined workspaces included with the WebSphere Message Broker Monitoring agent, see Chapter 5, "Monitoring with workspaces," on page 61, and the WebSphere Message Broker Monitoring agent section of the Tivoli® Enterprise Portal online help. For information about how to create a workspace using a predefined workspace as a template, see "Creating a workspace using a predefined workspace as a template" on page 71. For detailed information about creating and customizing views and workspaces, see the Tivoli Enterprise Portal online help or *Tivoli Enterprise Portal User's Guide*.

Monitoring situations

Situations are descriptions of conditions to which you want to be alerted. When situations are used on monitored systems they can, for example, alert you to a broker that has not been started, or to a message flow event. Situations can also be used to automate responses to problems, such as restarting a component or stopping a message flow that is consuming too much CPU processing time.

The WebSphere Message Broker Monitoring agent provides a set of predefined situations, which are designed to help you monitor critical activity and to serve as templates for creating customized situations of your own.

For information about the predefined situations included with the WebSphere Message Broker Monitoring agent, see "Predefined situations" on page 55. For information about creating and editing situations, see the Tivoli Enterprise Portal online help.

Attributes

Attributes are characteristics or properties of the objects monitored by the WebSphere Message Broker Monitoring agent; for example, the status of brokers, execution groups, and message flows, or the average time taken for a message flow to process a message. You specify attributes in query definitions, which are used to collect information presented in workspace views and to specify the conditions, or situations that trigger alerts and automated actions.

Attributes are organized into groups of related items. The WebSphere Message Broker Monitoring agent monitors 28 groups of attributes.

Attributes are used to define the queries which collect the information presented in workspace views. They are also used to specify the conditions, or *situations*, that trigger alerts and actions.

For information about the attribute groups available with WebSphere Message Broker Monitoring agent, see *IBM Tivoli Composite Application Manager Agent for WebSphere Message Broker Reference* or Tivoli Enterprise Portal online help.

Attributes and queries:

Chart and table views use queries to specify which attribute values to request from a Tivoli Enterprise Monitoring agent for display. You can modify those queries or design custom views by creating your own queries that collect data for just those attributes you specify. For more information about how to design custom views, see Tivoli Enterprise Portal online help.

Attributes and situations:

You can also use the attributes available with WebSphere Message Broker Monitoring agent to create your own situations to monitor the performance of your WebSphere Business Integration applications and message flows. These situations can monitor your WebSphere Business Integration broker resources or analyze message flows to alert you to problems when attribute values exceed the thresholds you specify.

For information about the predefined situations available with WebSphere Message Broker Monitoring agent, see "Predefined situations" on page 55.

Take Action commands

Using Tivoli Enterprise Portal, you can interact directly with your brokers and message flows with the Take Action feature. You enter your own commands, or choose from a list of predefined commands.

You can add a Take Action command to a monitoring situation and configure it to be run every time the situation becomes true. If you have IBM Tivoli OMEGAMON® DE, you can also create automation policies using Take Action commands.

You can use the Take Action commands included with WebSphere Message Broker Monitoring agent to issue IBM broker product commands from Tivoli Enterprise Portal. For example, you can start or stop brokers and their queue managers, and start and stop message flows, or change their trace characteristics by issuing commands from Tivoli Enterprise Portal. You can even change the type of accounting statistics collected or the interval at which they are collected.

For information about the commands provided with WebSphere Message Broker Monitoring agent, see "Take Action commands" on page 56 or the WebSphere Message Broker Monitoring agent section of the Tivoli Enterprise Portal online help.

Broker data collection

The WebSphere Message Broker Monitoring agent collects data about deployed broker objects for use in drawing topology views and display in workspace tables. The collection of this information can be slow because of the large volume of data that must be retrieved from the broker. However, version 7.3 of WebSphere Message Broker Monitoring agent offers more efficient broker data collection to overcome these performance issues.

In versions of WebSphere Message Broker Monitoring agent prior to version 7.0, which did not include topology workspaces, broker data was requested from the broker every time the agent was started. This method has the advantage that data is always up-to-date, but data collection can be slow if complex message flows are involved. This method can cause long agent startup times and the large amounts of data that is requested from the broker can cause delays in the broker processing other messages.

The WebSphere Message Broker Monitoring agent version 7.3 includes persistent broker data and other new features to overcome these problems. Broker data is stored in a file even when the agent is stopped, so that there is no need for the agent to collect the data every time the agent is started. The process performed by the agent during startup is illustrated in Figure 1 on page 5.

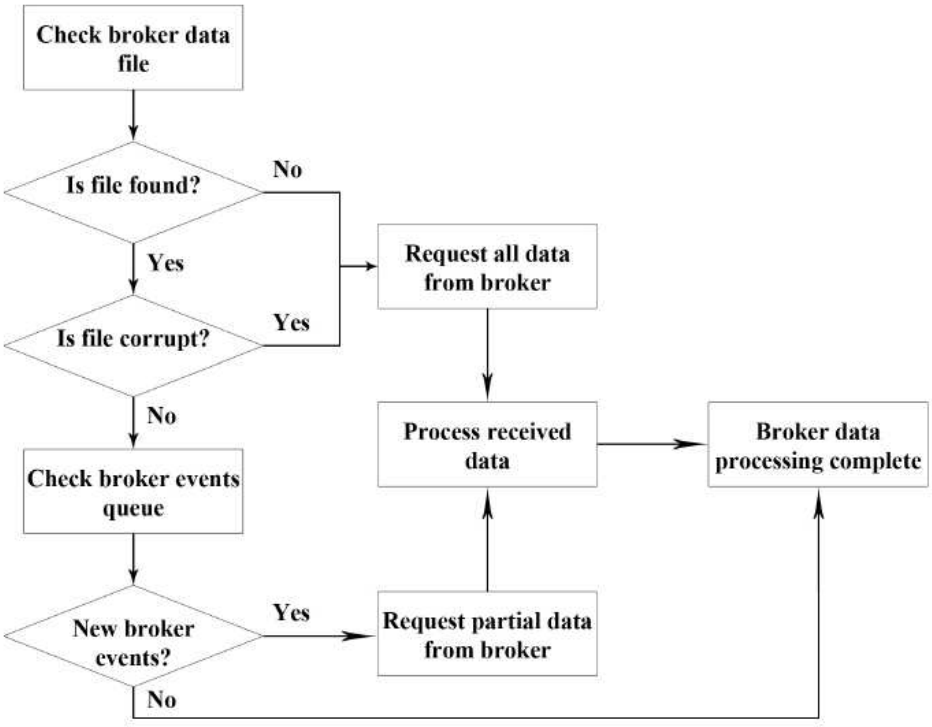

Figure 1. Broker data collection procedures performed at agent startup

Every time the agent is started, it loads the broker information from the broker data file. When it is the first time the agent has been started, the broker data file does not exist, so all broker data is requested from the broker, and one file is created with the received information. This information is also requested when the broker data file is corrupted.

If the broker data file is read successfully, the agent proceeds to check the reply queue for broker event messages. If the messages indicate that a broker object, such as a message flow, has changed, the agent requests information about only the changed object from the broker. As only information about changed objects is requested, agent startup times and broker overhead are greatly reduced.

Remember: The trace level and accounting/statistics settings are still collected every time the agent is started. However, the overhead placed on both the agent and broker by collecting this data is small.

Requesting all data from the broker, which is done when the agent is started for the first time, can place a significant overhead on the broker because of the large volume of data requested. From WebSphere Message Broker Monitoring agent version 7.0 onwards, the agent no longer requests all data simultaneously. Instead, requests for data are sent to the broker in stages. After sending a request for information (for example, about a single broker component) to the broker, the agent waits for a response before sending the next request.

By using this broker data collection method, the broker processes only one message from the agent at a time, instead of receiving a large number of requests simultaneously. This method greatly reduces the performance overhead on the

broker. However, the total time required for version 7.0 and later versions of the agent to collect all required information might be longer than with previous versions of the agent.

Historical data collection

You can use the historical data collection function of Tivoli Enterprise Portal to store data collected by the WebSphere Message Broker Monitoring agent. You can define the following properties of historical data collection:

- The interval at which data is collected
- The interval at which data is stored in a data warehouse (if you choose to do so)
- The location (either at the agent or at the Tivoli Enterprise Monitoring Server) at which the collected data is stored

For information about how to collection historical data for WebSphere Message Broker Monitoring agent, see Chapter 6, "Collecting historical data," on page 81.

IBM Tivoli Monitoring

IBM Tivoli Monitoring manages system and network applications on several operating systems and keeps track of the availability and performance of all parts of your enterprise. It provides IBM Tivoli OMEGAMON XE products with a common agent-server-client architecture:

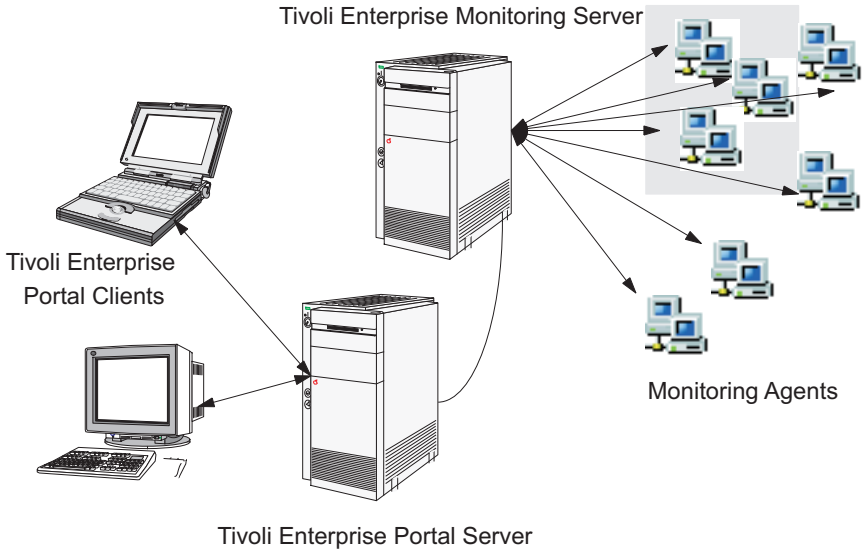

Figure 2. Agent–Server–Client architecture

Tivoli Enterprise Monitoring Server

Tivoli Enterprise Monitoring Server (monitoring server) gathers data from the Tivoli Enterprise Monitoring agent (monitoring agent), and acts as a collection and control point for alerts that are received from the agents. The Tivoli Enterprise Monitoring Server sends the data that it receives from the agents to Tivoli Enterprise Portal clients, where it is displayed in tabular or graphic views in a set of predefined or customized workspaces. The monitoring server also accepts requests for information or action from Tivoli Enterprise Portal clients and distributes them to the agents for processing.

Tivoli Enterprise Portal

Tivoli Enterprise Portal (portal) is the Java-based interface to the data monitoring and management resources of IBM Tivoli Monitoring. Depending on how it is installed, Tivoli Enterprise Portal can be used as either a desktop or browser-based client.

Tivoli Enterprise Portal has its own server, Tivoli Enterprise Portal Server (portal server). Tivoli Enterprise Portal Server performs common Tivoli Enterprise Portal functions, which reduces the processing performed by the Tivoli Enterprise Portal client.

Tivoli Enterprise Monitoring agents

Tivoli Enterprise Monitoring agents (monitoring agents) collect system or application data from *monitored*, or *managed* systems. For example, you can use the WebSphere MQ Monitoring agent to collect and analyze WebSphere MQ-specific data for all your remote and local queue managers from a single vantage point. The data is passed to Tivoli Enterprise Monitoring Server, and displayed in the Tivoli Enterprise Portal client.

The monitoring agents can also compare the current values of monitored properties against a set of defined conditions, and trigger alerts or actions when those conditions occur. They can accept and perform requested actions that are relayed to them from Tivoli Enterprise Portal clients by the monitoring server.

Configuration agents can create and configure objects. The WebSphere MQ Configuration agent can configure objects such as WebSphere MQ queue managers and all their components (queues, channels, processes, and other objects).

IBM Tivoli OMEGAMON DE product overview

The IBM Tivoli OMEGAMON DE feature package for Tivoli Enterprise Portal offers a process-driven view of your enterprise. You can use this software to bring together information from disparate sources in one workspace, including a range of operating systems, servers, databases, mainframes, and network and Internet components. Tivoli OMEGAMON DE software provides a single point of control from which you can manage all the resources that your business-critical applications rely on.

Tivoli OMEGAMON DE extends the capabilities of Tivoli OMEGAMON XE to include the following views:

- Enterprise-specific Navigator views

  The Navigator physical view presents the hierarchy of your managed enterprise by operating system and type of Tivoli Enterprise Monitoring agents. The Navigator business view offered by Tivoli OMEGAMON DE presents the hierarchy of managed objects. You can also define Navigator views for any logical groupings, such as business processes or by departmental hierarchy.
- Views of data from different types of monitoring agents in one workspace

  In a single workspace, you can build a table or chart with data from one type of monitoring agent, and another table or chart with data from a different agent. Within that workspace, you can display views from as many different agent types as are included on that branch of the Navigator.
- Linking application workspaces

  You can define links from a workspace associated with one type of monitoring agent to workspaces associated with other types of agents.

Policy management

The Tivoli Enterprise Portal policy management solution incorporates all the features of Tivoli OMEGAMON DE policy management and adds automation capabilities with the Workflow editor. You can use the Workflow editor to design sets of automated system processes, called policies, to resolve system problems. A policy performs actions, schedules work to be performed by users, or automates manual tasks.

Chapter 2. Customizing the monitoring agent

When you install the WebSphere Message Broker Monitoring agent, configuration parameters for the monitoring agents are set to their default values. This section describes the parameter file, and includes instructions for how to change parameter values if required.

Agent parameter file

The parameters that determine the operational and monitoring characteristics of an agent are stored in an XML file, which is created during the installation process of WebSphere Message Broker Monitoring agent. This XML file is referred to as the agent parameter file or the configuration file.

After installation, you can view and edit the agent parameter settings file at any time. You can also add optional configuration parameters to the file.

Names and locations of agent parameter files

This section describes the name and location of the agent parameter file on different operating systems.

Agent parameter files on z/OS systems

The agent parameter file member name is `KQIXML`. By default, it is installed in the following partitioned data set:

&rhilev.RKANDATV

Agent parameter files on UNIX and Linux systems

The agent parameter file name is `kqi.xml`.

If you specify broker and **agentId** parameters when the agent is started, the name of the agent parameter file is modified. The agent parameter file name for an agent started with optional broker and agentId parameters has the following form:

<*hostName*>_qi_<*brokerName*>_##_<*agentId*>.xml

If you specify only the **agentId** parameter and not broker parameters when the agent is started, then the agent parameter file name has the following form:

<*hostName*>_qi_<*agentId*>.xml

The agent parameter file is located in the <*install_dir*>/config directory, where <*install_dir*> is the directory where IBM Tivoli Monitoring is installed. The default directory is /opt/IBM/ITM.

Agent parameter files on Windows systems

The agent parameter file name is `kqi.xml`.

If you use the **Create Instance** option in the Manage Tivoli Monitoring Services window to replicate the agent, the name of the agent parameter file is modified. The agent parameter file name for a file associated with an agent has the following form:

kqi_<*instancename*>.xml

The agent parameter file is located in the <*install_dir*>\TMAITM6 directory, where <*install_dir*> is the directory where IBM Tivoli Monitoring is installed. The default directory is C:\IBM\ITM.

Default agent parameter file

The default agent parameter file contains a base set of parameters that are configured to their default values. However, you can modify the values of these parameters to suit the needs of your environment and you can add more parameters.

The default file on UNIX, Linux, and Windows systems

When you first install the WebSphere Message Broker Monitoring agent, the default kqi.xml file resembles the file that is displayed in Figure 3.

This default file contains the core parameters that control reporting and monitoring by the agent on UNIX, Linux, and Windows systems. Because no individual brokers or queue managers are specified in the file, using the default parameters displayed in this example results in all brokers being monitored by the agent.

```
<KqiAgent version="730"
          defaultRetainBrokerEvents="10"
          defaultRetainFlowEvents="10"
          retainProductEvents="10"
          discoveryInterval="300"
          defaultStatisticInterval="60"
          defaultFlowEventInterval="15"
          defaultHistoricalAccountingType="Archive"
          defaultRetainRecentSnapshotSamples="15"
          defaultRetainRecentArchiveSamples="5"
          defaultRetainRecentPubSubSamples="15"
          defaultRetainRecentResourceSamples="1"
          holdTimeForQuery="180"
          defaultReplyQueueName="KQI.AGENT.REPLY.QUEUE"
          defaultReplyQueueModel="SYSTEM.BROKER.MODEL.QUEUE"
          defaultCollectNodeData="NO"
          maximumMessageLength="10240"
          defaultPersistentBrokerData="NO"
          defaultRefreshInterval="300"
          defaultTakeActionAuthUsers="*"                    >

</KqiAgent>
```

Figure 3. Default kqi.xml file on Windows, UNIX, and Linux systems

The default file on z/OS systems

When you first install the WebSphere Message Broker Monitoring agent, the default KQIXML file resembles the file displayed in Figure 4 on page 11.

On z/OS® systems, you must specify all monitored brokers in the parameter file using the <MonitorBroker> tag. Using the parameters displayed in Figure 4 on page 11 results in the *M60ABRK* broker being monitored by the agent. To monitor additional brokers, add additional <MonitorBroker> tags as necessary.

```
<KqiAGENT version="730"
       agentid=""
       defaultRetainBrokerEvents="10"
       defaultRetainFlowEvents="10"
       retainProductEvents="10"
       discoveryInterval="300"
       defaultStatisticInterval="60"
       defaultFlowEventInterval="15"
       defaultHistoricalAccountingType="ARCHIVE"
       defaultRetainRecentSnapshotSamples="15"
       defaultRetainRecentArchiveSamples="5"
       defaultRetainRecentPubSubSamples="15"
       holdTimeForQuery="180"
       defaultReplyQueueName="KQI.AGENT.REPLY.QUEUE"
       defaultReplyQueueModel="SYSTEM.BROKER.MODEL.QUEUE"
       defaultCollectNodeData="NO"
       defaultTakeActionAuthUsers="*"
       maximumMessageLength="10240"
       defaultPersistentBrokerData="NO"
       defaultRefreshInterval="300"
       persistentDataPath="/var/cdl/OMEGAMON/BVT730/BVT1/CDHome/data"
       kqiUSSPath="/var/cdl/OMEGAMON/BVT730/BVT1/CDHome/kqi">
<MonitorBroker name="M60KBRK"
      alias="M60KBRK"
      envfileDirectory="/var/wmqi/M60KBRK">
</MonitorBroker>
</KqiAGENT>
```

Figure 4. Default KQIXML file on z/OS systems

Agent parameter file syntax

The agent parameter file contains a single pair of top-level <KqiAgent></KqiAgent> tags. It might also contain one or more instances of the <MonitorBroker></MonitorBroker> and <ConnectQueueManager></ConnectQueueManager>. Agent parameters are defined using attributes of these elements.

The parameters in the `kqi.xml` file (or `KQIXML` member on z/OS systems) are stored in XML format. You must adhere to XML syntax conventions when modifying the file. All attribute values must be enclosed in double quotation marks. However, the coding is not column-specific, so the spacing and line separation displayed in the sample files are not critical.

Setting agent parameters

You can change the agent configuration by modifying the default parameter values listed in the agent parameter file (see "Modifying an existing parameter"), or adding new parameters to it (see "Adding new parameters" on page 12). For detailed description about the configuration parameters, see "Agent parameter descriptions" on page 18.

Modifying an existing parameter

To modify a parameter in the agent parameter file, do the following steps:

1. Open the agent parameter file in a standard text editor.
2. Substitute the original value with your new value for the parameter that you want to modify.

Remember: The parameter value must be enclosed in a set of double quotation marks (" ").

3. Save and close the parameter file.

Example:

Original parameter:

```
defaultStatisticInterval="original_value"
```

Modified parameter:

```
defaultStatisticInterval="new_value"
```

Adding new parameters

To add new parameters to the agent parameter file, you must first determine whether an appropriate XML tag exists for the parameter.

- If the XML tag that the attribute belongs to already exists, insert the new parameter string into the tag block. The following examples illustrate how to add an **agentId** parameter to a shortened version of the default `kqi.xml` file:

  Original version:

```
<KqiAgent version="730"
   defaultRetainBrokerEvents="10"
   defaultRetainFlowEvents="10"
   retainProductEvents="10"
   discoveryInterval="300"
   defaultStatisticInterval="120"
   defaultFlowEventInterval="30"
   defaultHistoricalAccountingType="Archive"
   defaultRetainRecentSnapshotSamples="15"
   defaultRetainRecentArchiveSamples="5"
   defaultRetainRecentPubSubSamples="15"
   holdTimeForQuery="180"
   defaultReplyQueueName="SOME.NAME"
   defaultReplyQueueModel="SYSTEM.BROKER.MODEL.QUEUE"
   defaultCollectNodeData="NO"
   maximumMessageLength="10240"
   defaultPersistentBrokerData="NO"
   defaultRefreshInterval="300"
   defaultTakeActionAuthUsers="*">
 </KqiAgent>
```

  Modified version:

```
<KqiAgent version="730"
  agentId="new_value"
  defaultRetainBrokerEvents="10"
  defaultRetainFlowEvents="10"
  retainProductEvents="10"
  discoveryInterval="300"
  defaultStatisticInterval="120"
  defaultFlowEventInterval="30"
  defaultHistoricalAccountingType="Archive"
  defaultRetainRecentSnapshotSamples="15"
  defaultRetainRecentArchiveSamples="5"
  defaultRetainRecentPubSubSamples="15"
  holdTimeForQuery="180"
  defaultReplyQueueName="SOME.NAME"
  defaultReplyQueueModel="SYSTEM.BROKER.MODEL.QUEUE"
  defaultCollectNodeData="NO"
  maximumMessageLength="10240"
```

```
   defaultPersistentBrokerData="NO"
   defaultRefreshInterval="300"
   defaultTakeActionAuthUsers="*">
</KqiAgent>
```

- If the appropriate XML tag does not already exist, insert the correct opening and closing XML tags before adding the parameter string.

  For example, to set the **MonitorBroker** parameters, insert the <MonitorBroker></MonitorBroker> tags, and then add the parameters as follows:

  Original version:

```
<KqiAgent version="730"
   defaultRetainBrokerEvents="10"
   defaultRetainFlowEvents="10"
   retainProductEvents="10"
   discoveryInterval="300"
   defaultStatisticInterval="120"
   defaultFlowEventInterval="30"
   defaultHistoricalAccountingType="Archive"
   defaultRetainRecentSnapshotSamples="15"
   defaultRetainRecentArchiveSamples="5"
   defaultRetainRecentPubSubSamples="15"
   holdTimeForQuery="180"
   defaultReplyQueueName="SOME.NAME"
   defaultReplyQueueModel="SYSTEM.BROKER.MODEL.QUEUE"
   defaultCollectNodeData="NO"
   maximumMessageLength="10240"
   defaultPersistentBrokerData="NO"
   defaultRefreshInterval="300"
   defaultTakeActionAuthUsers="*">
</KqiAgent>
```

  Modified version:

```
<KqiAgent version="730"
   defaultRetainBrokerEvents="10"
   defaultRetainFlowEvents="10"
   retainProductEvents="10"
   discoveryInterval="300"
   defaultStatisticInterval="120"
   defaultFlowEventInterval="30"
   defaultHistoricalAccountingType="Archive"
   defaultRetainRecentSnapshotSamples="15"
   defaultRetainRecentArchiveSamples="5"
   defaultRetainRecentPubSubSamples="15"
   holdTimeForQuery="180"
   defaultReplyQueueName="SOME.NAME"
   defaultReplyQueueModel="SYSTEM.BROKER.MODEL.QUEUE"
   defaultCollectNodeData="NO"
   maximumMessageLength="10240"
   defaultPersistentBrokerData="NO"
   defaultRefreshInterval="300"
   defaultTakeActionAuthUsers="*">
<MonitorBroker name="MySpecialBrokerName"
    alias="BrokerAlias"
     statisticInterval="60"
    flowEventInterval="20"
    retainBrokerEvents="5"
    retainFlowEvents="5"
     collectNodeData="NO"
    takeActionAuthUsers="A?B,C*">
</MonitorBroker>
   <MonitorBroker name="AnotherBrokerName"
    collectNodeData="NO"   >
   </MonitorBroker>
   <ConnectQueueManager name="MY.BROKER.QMGR"
```

```
        replyQueueName="SOME.OTHER.NAME"
        replyQueueModel="MY.SPECIAL.MODEL.QUEUE">
    </ConnectQueueManager>
    </KqiAgent>
```

Tip: Each tag block encapsulates the parameters of a particular component. The `kqismpl.xml` sample file is included with UNIX, Linux, and Windows system versions of WebSphere Message Broker Monitoring agent; you can use this file as a template when you create your own `kqi.xml` file.

- On UNIX and Linux systems, the `kqismpl.xml` file is stored in the `<install_dir>/config` directory, where *<install_dir>* is the installation directory of IBM Tivoli Monitoring. The default installation directory is `/opt/IBM/ITM`.
- On Windows systems, the `kqismpl.xml` file is stored in the `<install_dir>\TMAITM6` directory, where *<install_dir>* is the installation directory of IBM Tivoli Monitoring. The default installation directory is `C:\IBM\ITM`.

Remotely configuring the WebSphere Message Broker Monitoring agent

On distributed systems such as Windows and UNIX operating systems, the agent parameter file is encoded using UTF–8 and can contain non-English language characters. On these systems, you can use the remote agent configuration feature instead of editing this file directly. On z/OS systems, however, the file is encoded using EBCDIC and must be edited manually.

You can remotely configure the WebSphere Message Broker Monitoring agent through Tivoli Enterprise Portal, or through the command line. See the following topics for detailed instructions:

- "Prerequisites"
- "Remotely configuring through Tivoli Enterprise Portal" on page 15
- "Remotely configuring through the command line" on page 16

Prerequisites

Before you can remotely deploy the WebSphere Message Broker Monitoring agent, identify the operating system where you want to remotely configure the agents and ensure that the following requirements are fulfilled in your monitoring environment:

- An OS agent is installed or deployed on the computer where you want to remotely configure the agents. For example, if you want to deploy the agents on the computer where the Windows operating system is running, you must have the monitoring agent for Windows OS installed on that computer. And the OS agent must be configured to connect to the same Tivoli Enterprise Monitoring Server as the WebSphere Message Broker Monitoring agent connects to. For information about how to install the OS agent, see *IBM Tivoli Monitoring Installation and Setup Guide*.
- The agent depot is populated on the Tivoli Enterprise Monitoring Server from which you configure agents across your environment. For example, if you want to remotely configure the WebSphere Message Broker Monitoring agent, you must have the WebSphere Message Broker Monitoring agent depot populated on the Tivoli Enterprise Monitoring Server. For detailed information about how to populate the agent depot, see the section about populating your agent depot in the chapter that explains how to deploy monitoring across your environment from a central location in *Installation and Setup Guide*.

Remotely configuring through Tivoli Enterprise Portal

To remotely configure the WebSphere Message Broker Monitoring agent through Tivoli Enterprise Portal, do the following steps:

1. Log on to the Tivoli Enterprise Portal.
2. In the Navigator view, navigate to the computer where you want to remotely configure the agent.
3. Right-click the Navigator item of the agent that you want to configure, and then click **Configure** from the menu. The Manage System Configuration window is opened as displayed in Figure 5.

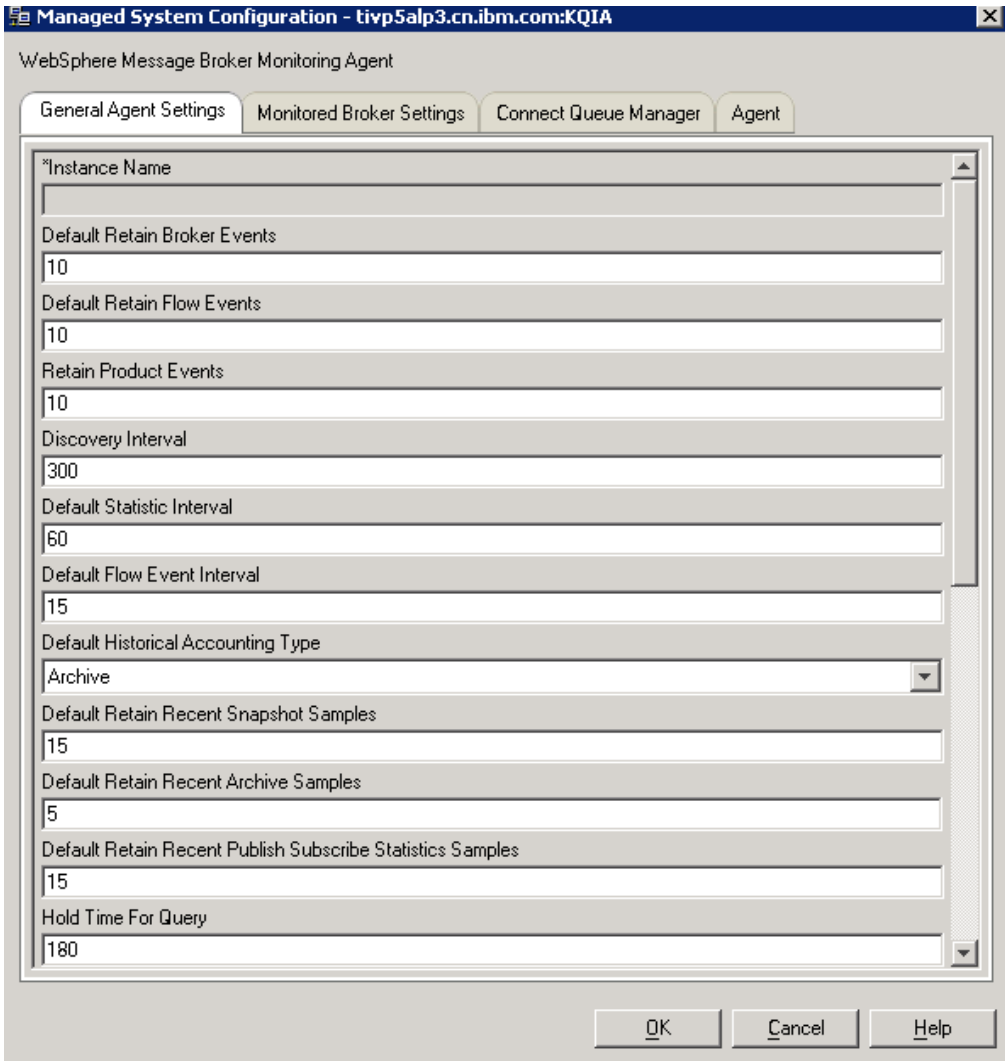

Figure 5. Managed System Configuration window

4. Provide the configuration information for agent in the required tabs.

   Tips:

   - The parameters for the **General Agent Settings** tab are the same as those parameters that are enclosed in the KqiAgent tag, except that there is an **`Instance Name`** parameter in the **General Agent Settings** tab, which indicates the name of the agent instance that you are configuring.

- The parameters for the **Monitored Broker Settings** tab are the same as those enclosed in the MonitorBroker tag.
- The parameters for the **Connect Queue Manager** tab are the same as those enclosed in the ConnectQueueManager tag.
- To get detailed information about the parameters for the **Agent** tab, at the lower right corner in the window, click **Help**.

5. When you finish the configuration, click **OK** to close the window.

Remotely configuring through the command line

Use the **tacmd configureSystem** command to remotely edit the configuration options of an agent. The **tacmd configureSystem** command has the following syntax:

```
tacmd configureSystem
                      {-m | --system} system
                      {-p | --property} section.name=value ...
```

To remotely configure the WebSphere Message Broker Monitoring agent from the command line, do the following procedure on the Tivoli Enterprise Monitoring Server to which the WebSphere Message Broker Monitoring agent is connected:

1. From a command prompt, change to the appropriate directory:
   - on Windows systems: *ITM_home*\bin
   - on UNIX or Linux systems: *ITM_home*/bin

   where *ITM_home* is the directory where IBM Tivoli Monitoring is installed.
2. Run the **tacmd login** command to log on to the Tivoli Enterprise Monitoring Server, to which the agent that you want to remotely configure reports.

   Provide the -m option with the host name of the system where the monitoring server is installed, and use the -u and -p options to specify the authenticated user.

   The following command logs on to the Tivoli Enterprise Monitoring Server on the *hostname* system with the *administrator* user ID, the *mypassword* password, and a login expiration time of *1440* minutes.

   ```
   tacmd login -s hostname -u administrator -p mypassword -t 1440
   ```
3. Run the **tacmd configureSystem** command to remotely configure the WebSphere Message Broker Monitoring agent.

   The following command configures a secondary agent instance whose **agentId** parameter is specified to *test* in the configuration file on the *AF66359F.cn.ibm.com* system.

   ```
   tacmd configureSystem -m test:AF66359F.cn.ibm.com:KQIA -p
   KqiAgent.defaultRetainBrokerEvents=5
   ```

   The following command configures the primary agent instance on the *AF66359F.cn.ibm.com* system by setting the value of the **defaultRetainBrokerEvents** parameter in the KqiAgent tag to *5*.

   ```
   tacmd configureSystem -m AF66359F.cn.ibm.com:KQIA -p
   KqiAgent.defaultRetainBrokerEvents=5
   ```

   Important: To ensure that the agent can be configured successfully, you must provide the -m option with the correct managed system name. The appropriate managed system name for each agent is displayed at the bottom of the workspace when you click the Navigator item of the agent instance in the Navigator view of Tivoli Enterprise Portal. You can also use the **tacmd listSystems** command to list the existing managed systems. The following

command lists all the systems in your enterprise with the product code *QI* (WebSphere Message Broker Monitoring agent). Use the managed system name that ends in **KQIA**.

```
tacmd listSystems -t QI
```

For detailed descriptions about the configuration parameters, see "Agent parameter descriptions" on page 18.

Creating multiple instances of the WebSphere Message Broker Monitoring agent

By default, the WebSphere Message Broker Monitoring agent that is created during installation monitors all brokers on your system. If you want to monitor one broker other than all brokers on the system, you can create a secondary agent instance.

Windows systems: Creating multiple instances of the monitoring agent

Perform the following steps to create a secondary instance of the WebSphere Message Broker Monitoring agent on Windows systems:

1. From the **Start** menu, click **Programs** > **IBM Tivoli Monitoring** > **Manage Tivoli Monitoring Services**. The Manage Tivoli Enterprise Monitoring Services window is displayed.
2. In the Manage Tivoli Enterprise Monitoring Services window, right-click **WebSphere Message Broker Monitoring Agent**, and then click **Create Instance** from the menu.
3. Enter a name for the instance when you are prompted and click **OK**. The new agent instance is created and is displayed in the Manage Tivoli Enterprise Monitoring Services window.
4. In the Manage Tivoli Enterprise Monitoring Services window, right-click the newly created agent instance, and then click **Configure Using Defaults**.
5. When you are asked whether to update the configuration file of the agent instance before configuring the agent, click **Yes**.
6. A message is displayed stating that configuration will wait for you to close your default text editor before continuing. Click **OK**. The configuration file is opened in your default text editor.
7. Edit the configuration file as follows:
   a. Ensure that the value of the **agentId** parameter is different to all other instances of the agent running on the same system.
   b. Use the <MonitorBroker> tag to specify which brokers the agent monitors. If this tag is not set, all brokers on the system are monitored. For further information about the <MonitorBroker> tag, see "MonitorBroker" on page 26.

   For more information about the related parameters, see "agentId" on page 18 and "MonitorBroker" on page 26.
8. Save and close the file.
9. A message is displayed stating that the configuration file edit session is complete. Click **Yes** to configure the agent.

The secondary agent instance is now configured and ready to start.

UNIX and Linux systems: Creating multiple instances of the monitoring agent

Do the following steps to create a secondary instance of the WebSphere Message Broker Monitoring agent on UNIX and Linux systems:

1. Navigate to the *install_dir*/bin directory, where *install_dir* is the installation directory of IBM Tivoli Monitoring.
2. To create a secondary instance of the WebSphere Message Broker Monitoring agent, run the **itmcmd agent start** command with the -o and -p options.

   ./itmcmd agent -o *agent_ID* -p *broker_name* start qi

   where *agent_ID* is a unique 4-digit alphanumeric agent ID, and *broker_name* is name of the monitored broker.

   - -o

     Specifies the monitoring agent ID (maximum of four characters).
   - -p

     Specifies the broker name (optional).

A secondary agent instance with the *agent_ID* name is created and it monitors the broker that you specified in the **itmcmd agent start** command.

Agent parameter descriptions

The parameters that you can set in the kqi.xml file (or in the KQIXML member on z/OS systems), and their associated XML tags, are described in the following sections. All the parameters are attributes of one of the three tags: KqiAgent, MonitorBroker, and ConnectQueueManager.

On distributed systems, the KqiAgent tag is required. The MonitorBroker and ConnectQueueManager tags are optional. If you only specify the KqiAgent tag in the agent parameter file, all brokers running on the same host system as the monitoring agent are monitored. However, if the monitored broker name contains non-alphanumeric ASCII characters, you must specify the **alias** attribute in the MonitorBroker tag. The alias is displayed as the managed system name of the broker on Tivoli Enterprise Portal. Broker names that contain non-alphanumeric ASCII characters might be displayed under wrong names on Tivoli Enterprise Portal.

KqiAgent

You can use the KqiAgent tag to specify parameters that are related to the agent itself. Any attributes that follow this tag apply to the agent as a whole and provide monitoring and connection defaults that can be overridden at the individual broker or queue manager level. All attributes are optional.

version

The **version** attribute specifies the version of WebSphere Message Broker Monitoring agent for which the parameters are set. This attribute is reserved for future product use. Do not alter this attribute from its default value unless instructed to do so by IBM Software Support.

agentId

The **agentId** attribute provides a short identifier (maximum length of 4 alphanumeric characters) for the WebSphere Message Broker Monitoring agent.

You must assign a unique agent ID to each agent in the following circumstances:

- You are running more than one WebSphere Message Broker Monitoring agent on the same system.
- You are running more than one monitored broker with the same name on different systems.

Unique agent IDs differentiate the broker-managed system names at the Tivoli Enterprise Monitoring Server, so that Tivoli Enterprise Portal presents them correctly.

This attribute is not set by default. To set it, supply an alphanumeric character string with no spaces. The value is used to create a unique managed system name.

defaultRetainBrokerEvents

The **defaultRetainBrokerEvents** attribute determines how many broker events to retain per broker. Report information is available only for retained broker events. When used with situations, these events are never reset automatically and if historical situations are active, you can view them historically.

The default number of events to retain is 10. This value must be greater than 0.

defaultRetainFlowEvents

The **defaultRetainFlowEvents** attribute determines how many message flow events to retain per broker. Report information is available only for retained message flow events. When used with situations, these events are never reset automatically and if historical situations are active, you can view them historically.

The default number of events to retain is 10. This value must be greater than 0.

retainProductEvents

The **retainProductEvents** attribute determines the number of product events to retain. Report information is available only for retained product events. When used with situations, these events are never reset automatically and if historical situations are active, you can view them historically.

The default number of events to retain is 10. This value must be greater than 0.

discoveryInterval

The **discoveryInterval** attribute determines the interval (in seconds) at which the agent attempts to discover new brokers that are created on the system. This attribute must be greater than 0; the default value is 300 seconds. This attribute can have a large value, for example, 86400 seconds (24 hours) or even longer when you are no longer creating new brokers. However, if you have a dynamic environment in which new brokers are created frequently, you must adjust this attribute accordingly.

defaultStatisticInterval

The **defaultStatisticInterval** parameter determines the minimum interval at which broker statistics are sampled, in seconds.

This parameter value controls the data collection interval of the following workspaces:

- Monitor Node Base Statistics
- Monitor Node Broker Statistics
- Monitor Node Execution Group Statistics
- Monitor Node Message Flow Statistics

- Monitor Node Sub-Flow Statistics

If you are collecting historical data for one of the statistics attribute groups, the historical data collection interval must be a multiple of the **defaultStatisticInterval** value; otherwise, historical data might be inaccurate. For example, if you configure historical data collection for one of the statistics attribute groups to run every 5 minutes (300 seconds), the **defaultStatisticInterval** value must be a submultiple of 300, such as 60 (1 minute), 75 (1.25 minutes), 100 (1.67 minutes), 150 (2.5 minutes), or 300 (5 minutes). Additionally, for accurate results, the historical data collection interval of all statistics attribute groups for which historical collection is being collected must be the same.

This value must be greater than 0; the default value is 60 seconds.

Tip: Do not specify a value that is less than 60 to the **defaultStatisticInterval** parameter. Otherwise, the data amount might be enormous.

defaultFlowEventInterval

The **defaultFlowEventInterval** attribute determines the interval at which message flow events are sampled, in seconds. This interval does not apply to broker events or product events.

This value must be greater than 0; the default value is 15 seconds.

defaultHistoricalAccountingType

The **defaultHistoricalAccountingType** attribute tells the agent whether to collect archive accounting data historically. The value of this attribute has no effect unless historical data collection for any of the four accounting attribute groups has been enabled in Tivoli Enterprise Portal. This attribute can have the following values:

- None

  No accounting data is logged historically, even if historical data collection is enabled in Tivoli Enterprise Portal.
- Archive

  Only Archive accounting data is logged historically. This is the default setting.

Tip: When you enable historical data collection on Tivoli Enterprise Portal globally, you can set this attribute to None to disable historical data collection on some specific broker. To do this, you must have configured one agent instance to monitor the broker only.

defaultRetainRecentSnapshotSamples

The **defaultRetainRecentSnapshotSamples** attribute specifies the default number of recent snapshot records that are stored by the agent for any given message flow. The default value of this attribute is 15, which is equivalent to approximately 5 minutes of snapshot data. However, the agent always ensures that it retains enough historical samples to log the data historically if historical data collection is enabled.

You can set this attribute to 0. When the **defaultRetainRecentSnapshotSamples** value is 0, the WebSphere Message Broker Monitoring agent does not store any recent snapshot records. The following workspaces will be empty:

- Snapshot Accounting Message Flow Statistics
- Snapshot Accounting Node Statistics

- Snapshot Accounting Terminal Statistics
- Snapshot Accounting Thread Statistics

defaultRetainRecentArchiveSamples

The **defaultRetainRecentArchiveSamples** attribute specifies the number of recent archive records that are stored by the agent for any given message flow. The greater this value, the more data is available in the Archive Accounting Message Flow Statistics workspace.

The default archive interval for a broker is 60 minutes. The default value of the **defaultRetainRecentArchiveSamples** attribute is 5, so 5 hours of recent archive data can be viewed in the Archive Accounting Message Flow Statistics workspace if the default archive interval has not been changed. The agent always ensures that it retains enough historical samples to log the data historically if historical data collection is enabled.

You can set this attribute to 0. When the **defaultRetainRecentArchiveSamples** value is 0, the WebSphere Message Broker Monitoring agent does not store any recent archive records. The following workspaces will be empty:

- Archive Accounting Message Flow Statistics
- Archive Accounting Node Statistics
- Archive Accounting Terminal Statistics
- Archive Accounting Thread Statistics

defaultRetainRecentResourceSamples

The **defaultRetainRecentResourceSamples** attribute specifies whether to store recent resource records for message flows. This attribute value can be either 1 or 0. The default resource interval for a broker is 20 seconds. So when this attribute is set to 1, 20 seconds of resource statistics data can be displayed in the Resource Statistics workspaces. To view more data about resource statistics, enable historical data collection on the concerned attribute group.

When this attribute is set to 0, the WebSphere Message Broker Monitoring agent does not subscribe the resource statistics events that are emitted by WebSphere Message Broker and the Resource Statistics workspaces are empty.

holdTimeForQuery

The **holdTimeForQuery** attribute specifies the length of time, in seconds, that the agent must retain detailed accounting data. This data must be retained for viewing detailed information in accounting workspaces. The greater this value, the more data is available in the accounting workspaces.

The default value of this attribute is 180, which means that, regardless of other attributes related to data retention, you can view detailed information about a selected data sample for at least 3 minutes after the data sample was originally displayed. If you view the retained data in a workspace, every time you refresh the workspace the length of time for which the data is retained is extended by the **holdTimeForQuery** value.

This attribute must have a value greater than 0. If you do not want the agent to retain data, specify a value of 1 (1 second).

For example, the Snapshot Message Flow Accounting workspace presents a list of monitored message flows and their statistics. By default, for the next 3 minutes you can select a row in the table and link to a more detailed view of the displayed

data (either Snapshot Thread Accounting or Snapshot Node Accounting). Each time that you view a data pertaining to a given message flow, you extend the time that you can view the sample by 3 minutes. If the sample is not retained for some other reason and if you do not access refresh the data for 3 minutes, the data is deleted. If the data is deleted, the data from the most recent sample is displayed instead.

defaultReplyQueueName

The **defaultReplyQueueName** attribute specifies the name of the queue that is used to receive publications and reply messages from the broker. A permanent dynamic queue is created using the model queue specified by "defaultReplyQueueModel" if the name given does not pre-exist as a predefined or a permanent dynamic queue on the queue manager of the broker. This value can be overridden for individual queue managers using the **replyQueueName** attribute.

If this attribute is changed after starting a WebSphere Message Broker Monitoring agent, remove the subscription associated with the previous queue name and the previous queue.

The default value is KQI.AGENT.REPLY.QUEUE.

defaultReplyQueueModel

The **defaultReplyQueueModel** attribute specifies the name of the queue that is used as a model for creating the agent reply queue if the name given in the **defaultReplyQueueName** attribute does not pre-exist as a predefined or a permanent dynamic queue on the queue manager of the broker. The agent creates a queue with the name specified by the **defaultReplyQueueName** attribute for the queue manager of the broker, using the properties of the queue specified by the **defaultReplyQueueModel** attribute as a model. This value can be overridden for individual queue managers using the **replyQueueModel** attribute.

The model queue specified by this attribute must be a permanent dynamic model queue. If the **persistentBrokerData** attribute is set to YES, the model queue must enable persistent messages, otherwise, broker event messages that are received when the agent is stopped might be lost. If these event messages are lost, the workspace information is not up-to-date, because the broker uses these messages to detect changes in objects deployed in the broker environment while the agent is stopped. If you cannot use a permanent dynamic model queue with persistent messages enabled, use one of the following options:

- Set the **persistentBrokerData** attribute to No. This setting forces the agent to automatically detect all deployed objects at startup, ensuring that workspace information is always up-to-date. And this setting increases processing overhead on the message broker and in complex environments this might cause long delays in starting the WebSphere Message Broker Monitoring agent.
- When you know that the broker environment has been updated, issue the **QI Refresh Broker Data** take action command to update workspace information. Issuing this take action command causes all broker information to be updated instead of only information about updated components, and so might take a long time.

The default value is SYSTEM.BROKER.MODEL.QUEUE.

defaultTakeActionAuthUsers

The **defaultTakeActionAuthUsers** attribute specifies which Tivoli Enterprise Portal users are authorized to issue the Take Action commands that are associated with a WebSphere Message Broker Monitoring agent. You can specify multiple values for

this attribute to authorize multiple users. You can also use the asterisk (*) and question mark (?) wildcard characters to authorize a group of users. Only Tivoli Enterprise Portal users who have an ID that matches the values of this attribute are authorized to issue Take Action commands handled by the WebSphere Message Broker Monitoring agent. The portal user IDs are defined at the Tivoli Enterprise Monitoring Server and do not necessarily exist on the system on which the agent is running. The value of this attribute applies only to the commands, which meet one of the following conditions:

- The command is not related to a particular broker.
- The command is related to a particular broker, and the broker does not have a specific list of users specified using the **takeActionAuthUsers** attribute.

The default value is an asterisk (*), which authorizes all Tivoli Enterprise Portal users to issue Take Action commands that are associated with this agent.

defaultCollectNodeData

The **defaultCollectNodeData** attribute determines whether node definition detail data collection and parsing are performed. Possible values are YES to enable this data or NO to disable it (case insensitive). This attribute can be overridden for individual message brokers using the "collectNodeData" on page 29 attribute.

If the **defaultCollectNodeData** attribute is set to NO, the following workspaces contains no data:

- Processing Node Attributes workspace
- Message Flow Node Topology workspace

In addition, the Queue Name and Queue Manager Name attributes in the following workspaces contains no data:

- Archive Accounting Node Statistics workspace
- Snapshot Accounting Node Statistics workspace

However, if message brokers in your environment have large message flows with a lot of message processing node data, enabling this option might significantly degrade system performance.

The default value is NO.

maximumMessageLength

The **maximumMessageLength** attribute specifies the maximum length of reply or event message from the broker that the agent processes. Its unit is in kilobytes. Any reply or event messages that exceed this length are processed, and so you cannot view the information that these messages contain in Tivoli Enterprise Portal. The default value is suitable for most environments. However, if your environment contains complex message flows, you might need to increase this value to ensure that all data is processed by the agent.

If an event message is truncated due to being longer than the **maximumMessageLength** value, the truncated message is still processed in order to report broker events. However, event details can be processed.

The default value is 10240 kilobytes. The maximum value is 102400.

defaultPersistentBrokerData

The **defaultPersistentBrokerData** attribute specifies whether data related to brokers is stored persistently by writing it to a file. Possible values are YES and

NO (case insensitive). When this option is enabled, the file used to store the data might be large, depending on the volume of broker information that needs to be stored.

The default value is NO.

Remember:

- The agent must be restarted to make the change of this attribute value take effect. Before you start the agent again, delete the following items (if any) from your environment:
  - Subscriptions that are not in use
  - The reply queue of a broker
- If you change this attribute from YES to NO, also delete the existing persistent data file before restarting the agent.

  Tip: The persistent data file is stored in the following directory:
  - Windows: *install_dir*\TMAITM6\logs\History\KQI
  - UNIX or Linux: *install_dir*/*arch*/qi/hist

  where *install_dir* is the directory where IBM Tivoli Monitoring is installed. The default is C:\IBM\ITM on Windows systems or /opt/IBM/ITM on UNIX or Linux systems. *arch* is the architecture code of the operating system of this computer. For a list of architecture codes, see Appendix D, "Architecture codes," on page 139.
- If you change this attribute from NO to YES, set the **defaultReplyQueueModel** attribute to the name of a permanent model queue. Make sure that the specified permanent model queue exists, such as SYSTEM.DURABLE.MODEL.QUEUE, which is automatically created by the broker.

defaultRefreshInterval

The **defaultRefreshInterval** attribute specifies the interval in seconds between requesting information not reported as broker events from the broker. This data is not related to the deployment of broker objects. This attribute includes trace and accounting/statistics settings for execution groups and message flows and broker subscription data. If you increase the length of the interval at which this data is collected, data in workspaces might be less up-to-date, but performance can be improved.

This data is only requested from the broker at this interval if a user is viewing the data in a workspace. If no user is viewing a workspace that contains this data, the data is not collected by the agent.

The default value is 300 seconds.

commandTimeoutInterval

The **commandTimeoutInterval** attribute specifies the amount of time (in seconds) that the agent waits for a response from a broker after sending a command before it issues a message indicating that the broker is not responding.

This value is not included in the configuration file by default, instead the value 300 is used. If specified, the value of this attribute must be greater than zero.

Do not modify this attribute unless directed to do so by IBM Software Support.

maximumCommandRetryCount

The **maximumCommandRetryCount** attribute specifies the number of times that the agent reissues commands after failing to receive a response from a broker before giving up.

This value is not included in the configuration file by default, instead the value 1 is used which means the agent does not issue commands after failing to receive the response from a broker. If specified, the value of this attribute must be greater than zero. If its value is set to be greater than 1, the agent issues commands for one time less than the specified number of times before giving up. For example, if you set the attribute to be six, the agent issues five times after it fails to receive a response from a broker before giving up.

Do not modify this attribute unless you are directed to do so by the IBM Software Support.

maximumAgentCollectionThreads

The **maximumAgentCollectionThreads** attribute specifies the number of agent collection threads that are used by the agent for monitoring brokers. You can modify this value to increase or decrease the number of brokers that the agent can monitor.

This value is not included in the configuration file by default, instead the value 64 is used, which creates enough threads to monitor 10 brokers. If specified, the value of this attribute must be greater than zero.

On AIX® systems, do not attempt to monitor more than 10 brokers with a single agent. If you have to monitor more than 10 brokers, split them among several agents. See “Monitoring a large number of brokers” on page 32 for further details.

To use a single agent to monitor more than 10 brokers on operating systems other than AIX systems, increase this value by 6 for each additional broker monitored.

Do not modify this attribute unless absolutely necessary because monitoring a large number of brokers with a single agent might adversely affect monitoring performance. To monitor large number of brokers, create additional agents to monitor some brokers. See “Monitoring a large number of brokers” on page 32 for more details.

persistentDataPath

The **persistentDataPath** specifies the path of persistent data files on z/OS systems only. This value is set automatically during the configuration to *candle home*\data, where *candle home* is the candle home directory specified by the user. However, this attribute has no default value and must be specified before starting the agent for the first time if not previously configured.

kqiUSSPath

The **kqiUSSPath** specifies the path of the executable files that run under USS. This parameter is for z/OS systems only. The value is set automatically during the configuration to <*candle_home*>\kqi, where <*candle_home*> is the candle home directory that you specified.

This parameter has no default value and must be specified before you start the agent for the first time if it is not previously configured.

defaultWMBInstallDirectory

The **defaultWMBInstallDirectory** attribute provides the installation directory of the brokers that you want to monitor to the WebSphere Message Broker Monitoring agent. The default value is blank. With the default configuration, the monitoring agent can automatically search for the installation directory and discover all brokers in that directory. Specify this attribute only when the monitoring agent cannot find the correct installation directory in a complex environment. You might have to configure this attribute in the following circumstances:

- The work path directory of WebSphere Message Broker (MQSI_WORKPATH) is not the default value.
- Both the 32-bit and 64-bit brokers are installed on a 64-bit Windows system at the same time.

The **defaultWMBInstallDirectory** attribute only applies to distributed systems. This attribute is not in the default `kqi.xml` file. You can add it in the `kqi.xml` file as needed.

Remember: You can specify only one directory for this attribute.

defaultWMQInstallDirectory

The **defaultWMQInstallDirectory** attribute provides the installation directory of WebSphere MQ to the WebSphere Message Broker Monitoring agent. Some JAR files of WebSphere MQ are required by the Configuration Manager Proxy (CMP) API to interact with the brokers. The default value is blank. With the default configuration, the monitoring agent can automatically search for the installation directory. Specify this attribute only when the monitoring agent cannot find the correct installation directory in a complex environment.

The **defaultWMQInstallDirectory** attribute only applies to distributed systems. This attribute is not in the default `kqi.xml` file. You can add it in the `kqi.xml` file as needed.

MonitorBroker

The MonitorBroker tag encapsulates parameters that apply to a single monitored broker. Specify one MonitorBroker tag for each broker.

On UNIX, Linux, and Windows systems, If no brokers are specified, all brokers running on the same host system as the agents are monitored, as determined by self-discovery. If any brokers are specified explicitly using this tag, any other brokers that you want to monitor must also be specified using MonitorBroker tags. All associated attributes are optional except for the **name** attribute.

On z/OS systems, WebSphere Message Broker Monitoring agent does not currently support self-discovery of brokers on z/OS systems. Each broker to be monitored must be explicitly specified using MonitorBroker tags and attributes. All associated attributes are optional except for the **name** and **componentDirectory** attributes.

Remember: Only alphanumeric ASCII characters can be included in a broker name on distributed systems, or EBCDIC characters on z/OS systems. Otherwise the navigator items might be displayed under wrong names on Tivoli Enterprise Portal. Alphanumeric ASCII character set consists of the numbers 0-9 , lowercase letters a-z, uppercase letters A-Z, and the underscore character (_). If the broker

name contains a space, a comma or any other special characters, you must specify the **alias** attribute for the broker. The alias is used to be displayed on Tivoli Enterprise Portal.

name

The **name** attribute specifies the name of the broker monitored by the agent.

The **name** attribute is required for all MonitorBroker tags.

alias

The **alias** attribute provides an alternative name for a broker in addition to that specified using the **name** attribute. Only alphanumeric ASCII characters can be included in a broker name on distributed systems, or EBCDIC characters on z/OS systems. Otherwise the navigator items might be displayed under wrong names on Tivoli Enterprise Portal. If the broker name contains a space, a comma or any other special characters, you must specify the **alias** attribute for the broker.

If specified, the alias is displayed as the managed system name (node name) of the broker on Tivoli Enterprise Portal. You can use this attribute if you want to specify a more user-friendly name for your broker to simplify managing your system. This is especially useful when the **name** attribute of a broker is exceptionally long or you have several brokers with similar names and you want to be able to distinguish them easily in Tivoli Enterprise Portal.

Remember: The alternative name must conform to the following rules:

- The maximum length is 22 characters.
- Only alphanumeric ASCII characters can be included on distributed systems, or EBCDIC characters on z/OS systems. Alphanumeric ASCII character set consists of the numbers 0-9, lowercase letters a-z, uppercase letters A-Z, and the underscore character (_).

For example, if you have a broker named BROKER1 on your system, you can specify an alias as follows:

```
<MonitorBroker name="BROKER1" alias="BK1"> </MonitorBroker>
```

envfileDirectory

This attribute is required only on z/OS systems and ignored if it is specified on other operating systems.

The **envfileDirectory** attribute specifies the z/OS UNIX system services (USS) directory that the broker `ENVFILE` file is located in.

statisticInterval

The **statisticInterval** attribute overrides the global **defaultStatisticInterval** attribute and determines the minimum interval at which broker statistics are sampled in seconds. If this attribute is not specified, the **defaultStatisticInterval** value is used instead. If this attribute is specified, the value must be greater than 0.

If you are collecting history data for one of the statistics attribute groups, the configured historical data collection interval must be a multiple of the **statisticInterval** value; otherwise, historical data collection might produce unpredictable results. For example, if you configure historical data collection for one of the statistics attribute groups to run every 5 minutes (300 seconds), the **statisticInterval** value must be a submultiple of 300, such as 60 (1 minute), 75 (1.25 minutes), 100 (1.67 minutes), 150 (2.5 minutes), or 300 (5 minutes). For

accurate results, the historical data collection interval must be the same value for all statistics attribute groups for which historical data collection is enabled.

Tip: Do not specify a value that is less than 60 to the **defaultStatisticInterval** attribute. Otherwise, the data amount might be enormous.

flowEventInterval

The **flowEventInterval** attribute overrides the global **defaultFlowEventInterval** attribute and determines the interval at which message flow events are sampled, in seconds. If this attribute is not specified, the **defaultFlowEventInterval** value is used instead. If this attribute is specified, the value must be greater than 0.

retainBrokerEvents

The **retainBrokerEvents** attribute overrides the global **defaultRetainBrokerEvents** attribute and determines how many broker events to retain per broker. Report information is available only for retained broker events. When used with situations, these events are never reset automatically and if historical situations are active, you can view them historically. If this attribute is not specified, the **defaultRetainBrokerEvents** value is used instead. If the attribute is specified, the value must be greater than 0.

retainFlowEvents

The **retainFlowEvents** attribute overrides the global **defaultRetainFlowEvents** attribute and determines how many message flow events to retain per broker. Report information is available only for retained message flow events. When used with situations, these events are never reset automatically and if historical situations are active, you can view them historically. If this attribute is not specified, the **defaultRetainFlowEvents** value is used instead. If this attribute is specified, the value must be greater than 0.

takeActionAuthUsers

The **takeActionAuthUsers** attribute specifies which Tivoli Enterprise Portal users are authorized to issue the Take Action commands to a particular agent running on a particular broker. You can specify multiple values for this attribute to authorize multiple users. Include the asterisk (*) and question mark (?) characters in the values to authorize a group of users. Only Tivoli Enterprise Portal users who have an ID that matches the values of this attribute are authorized to issue Take Action commands handled by the WebSphere Message Broker Monitoring agent. Tivoli Enterprise Portal user IDs are defined in the Tivoli Enterprise Monitoring Server and do not necessarily exist on the node on which the agent is running. The values of this attribute apply only to commands that are not related to a particular broker, or that are related to a broker that does not have a specific list of users specified using the **takeActionAuthUsers** attribute. This attribute overrides the global **defaultTakeActionAuthUsers** attribute.

If this attribute is not specified, the **defaultTakeActionAuthUsers** value is used instead.

historicalAccountingType

The **defaultHistoricalAccountingType** attribute tells the agent whether to collect historical archive accounting data. The value of this attribute has no effect unless historical data collection for any of the four accounting attribute groups has been enabled in Tivoli Enterprise Portal. This attribute overrides the global **defaultHistoricalAccountingType** attribute. For more information, see “defaultHistoricalAccountingType” on page 20.

retainRecentSnapshotSamples

The **retainRecentSnapshotSamples** attribute specifies the number of recent snapshot records to keep for any given message flow. It overrides the global **defaultRetainRecentSnapshotSamples** attribute. By default, this attribute is not specified in the configuration file; the value of **defaultRetainRecentSnapshotSamples** is used instead.

retainRecentArchiveSamples

The **retainRecentArchiveSamples** attribute determines the number of recent archive records to keep for any given message flow. It overrides the global **defaultRetainRecentArchiveSamples** attribute. By default this attribute is not specified in the configuration file, and the value of **defaultRetainRecentArchiveSamples** is used instead.

retainRecentPubSubSamples

The **retainRecentPubSubSamples** attribute specifies the minimum number of recent publish and subscribe data records stored by the agent. The higher this value, the more data is available in the Publish-Subscribe Statistics workspace. This attribute overrides the global **defaultRetainRecentPubSubSamples** attribute.

By default, this attribute is not specified in the configuration file. The global **defaultRetainRecentPubSubSamples** attribute is used instead.

retainRecentResourceSamples

The **retainRecentResourceSamples** attribute determines whether to keep recent resource records for any given message flow. This attribute value can be either 1 or 0. The default resource interval for a broker is 60 minutes. So when this attribute is set to 1, one hour of resource statistics data can be displayed in the Resource Statistics workspaces. When this attribute is set to 0, the WebSphere Message Broker Monitoring agent does not subscribe the resource statistics events that are emitted by WebSphere Message Broker and the Resource Statistics workspaces are empty. You can set this attribute to 0 only for WebSphere Message Broker 7.0 or later.

By default this attribute is not specified in the configuration file, and the value of **defaultRetainRecentResourceSamples** is used instead.

collectNodeData

The **collectNodeData** attribute determines whether node definition data collection and parsing are performed. Possible values are YES to enable these options or NO to disable them (case insensitive). If this attribute is not specified for a broker, the default value specified by the global "defaultCollectNodeData" on page 23 attribute is used instead.

If the **collectNodeData** attribute is set to NO, the following workspaces contains no data:

- Processing Node Attributes workspace
- Message Flow Node Topology workspace

In addition, the following workspaces only contains partial data:

- Archive Accounting Node Statistics workspace
- Snapshot Accounting Node Statistics workspace

However, if message brokers in your environment have large message flows with a lot of message processing node data, enabling this option might significantly degrade system performance.

persistentBrokerData

The **persistentBrokerData** attribute specifies whether data related to brokers is stored persistently by writing it to a file. Possible values are YES and NO (case insensitive). When this option is enabled, the file used to store might be large, depending on the volume of broker information that needs to be stored.

If this attribute is not specified, the value of the **defaultPersistentBrokerData** attribute is used instead.

Remember:

- The agent must be restarted to make the change of this attribute value take effect. Before you start the agent again, delete the following items (if any) from your environment:
  - Subscriptions that are not in use
  - The reply queue of a broker
- If you change this attribute from YES to NO, also delete the existing persistent data file before restarting the agent.

  Tip: The persistent data file is stored in the following directory:
  - Windows: *install_dir*\TMAITM6\logs\History\KQI
  - UNIX or Linux: *install_dir*/*arch*/qi/hist

  where *install_dir* is the directory where IBM Tivoli Monitoring is installed. The default is C:\IBM\ITM on Windows systems or /opt/IBM/ITM on UNIX or Linux systems. *arch* is the architecture code of the operating system of this computer. For a list of architecture codes, see Appendix D, "Architecture codes," on page 139.
- If you change this attribute from NO to YES, set the **replyQueueModel** attribute to the name of a permanent model queue. Make sure that the specified permanent model queue exists, such as SYSTEM.DURABLE.MODEL.QUEUE, which is automatically created by the broker.

refreshInterval

The **refreshInterval** attribute specifies the number of seconds interval between requesting information not reported as broker events from the broker. This data is not related to the deployment of broker objects. This includes trace and accounting/statistics settings for execution groups and message flows and broker subscription data. If you increase the length of the interval at which this data is collected, data in workspaces might be less up-to-date, but performance can be improved.

If this attribute is not specified, the value of **defaultRefreshInterval** is used instead.

WMBInstallDirectory

The **WMBInstallDirectory** attribute provides the installation directory of the broker that you want to monitor to the WebSphere Message Broker Monitoring agent. By default, the monitoring agent can automatically search for the installation directory and discover the broker. Specify this attribute only when the monitoring agent

cannot discover the broker in a complex environment. If this attribute is not specified, the monitoring agent will use the **defaultWMBInstallDirectory** value to search for all brokers.

The **WMBInstallDirectory** attribute only applies to distributed systems. This attribute is not in the default `kqi.xml` file. You can add it in the `kqi.xml` file as needed.

WMQInstallDirectory

The **WMQInstallDirectory** attribute provides the installation directory of WebSphere MQ to the WebSphere Message Broker Monitoring agent. Some JAR files of WebSphere MQ are required by the CMP API to interact with the broker. By default, the monitoring agent can automatically find this installation directory. Specify this attribute only when the monitoring agent cannot find it in a complex environment. If this attribute is not specified, the **defaultWMQInstallDirectory** value is used.

The **WMQInstallDirectory** attribute only applies to distributed systems. This attribute is not in the default `kqi.xml` file. You can add it in the `kqi.xml` file as needed.

ConnectQueueManager

The ConnectQueueManager tag encapsulates parameters that affect queue managers. You can specify multiple ConnectQueueManager tags for different queue managers.

This tag is optional, but if you specify it, the **name** attribute is required. If this tag is not specified, agent-level defaults are used for the remaining attributes, and the agent automatically connects to the associated queue manager of the monitored broker. However, if the default reply and model queue names are not sufficient for a broker queue manager, you must specify all attributes of this tag:

name

The **name** attribute specifies the name of the queue manager that the agent connects to. This attribute is required.

replyQueueName

The **replyQueueName** attribute specifies the name of the queue that is used by the agent to receive publications and reply messages from the broker using this queue manager. If the named queue does not exist, a permanent dynamic queue is created automatically.

If this attribute is changed after starting a WebSphere Message Broker Monitoring agent, remove the subscription associated with the previous queue name and the previous queue. If a single broker is monitored by more than one agent, specify a different reply queue for each agent.

If this attribute is not specified, the **defaultReplyQueueName** value is used instead.

replyQueueModel

The **replyQueueModel** attribute specifies the name of the queue that is used as a model for creating the agent reply queue on this queue manager. The agent can create a queue with the name specified by the **replyQueueName** attribute for the queue manager using the properties of the queue specified by the **replyQueueModel**

attribute as a model. Thus, by changing the properties of the queue named by this attribute, you can modify the properties of the agent reply queue used by the queue manager.

The model queue specified by this attribute must be a permanent dynamic model queue with persistent messages enabled when the **persistentBrokerData** attribute is YES, otherwise broker event messages received when the agent is stopped might be lost. In this case, as the broker uses these messages to detect changes in objects deployed in the broker environment while the agent is stopped and update workspace information accordingly. If you cannot use a permanent dynamic model queue with persistent messages enabled, use one of the following options:

- Set the **persistentBrokerData** attribute to NO. In this way, the agent can automatically detect all deployed objects at startup, ensuring that workspace information is always up-to-date. Setting the **persistentBrokerData** attribute to No increases processing overhead on the message broker, and in complex environments might cause long delays in starting the WebSphere Message Broker Monitoring agent.
- When you know that the broker environment has been updated, issue the **QI Refresh Broker Data** take action command to update workspace information. Issuing this take action command causes all broker information to be updated instead of only information related to updated components, and so might take a long time.

If this attribute is not specified, the **defaultReplyQueueModel** value is used instead.

Examples of agent configuration

This section presents several examples to explain how you can customize the configuration for better performance of the WebSphere Message Broker Monitoring agent.

Monitoring a large number of brokers

To minimize performance overhead, run only one WebSphere Message Broker Monitoring agent on each host system. However, if the performance of the agent becomes adversely affected by monitoring a large number of brokers simultaneously, you can split monitoring between several agents.

For most distributed systems, one agent per host system must suffice, because the agent is designed to monitor all brokers on such a system, or to monitor a subset of brokers, as specified in the agent parameter file.

For systems with a large number of brokers, such as z/OS systems, you might need to adjust the agent parameter that limits the number of brokers that are monitored (see “maximumAgentCollectionThreads” on page 25) and verify that monitoring the large number of brokers is not reducing performance. If performance is affected, you can divide monitoring among several agents.

If CPU usage is high or response times for displaying Tivoli Enterprise Portal workspace reports or situation monitoring actions are unacceptably long, performance might be adversely affected by monitoring too many brokers. You might need to experiment to determine the best number of agents for your enterprise.

Exception: On AIX systems, because of shared memory limitations, do not attempt to monitor more than 10 brokers with a single agent on an AIX host.

Dividing broker monitoring between different agents on Windows systems

To dividing broker monitoring between different agents, do the following steps:

1. Optional: Create the secondary instance of the WebSphere Message Broker Monitoring agent, if you have not done so. For more information about how to create the agent instance, see "Windows systems: Creating multiple instances of the monitoring agent" on page 17.
2. In the Manage Tivoli Enterprise Monitoring Services window, right-click the agent instance that you want to configure, and then click **Reconfigure**.
3. In the Agent Advanced Configuration window, click **OK**, until a message is displayed asking if you want to update the configuration file of the agent instance before configuring the WebSphere Message Broker Monitoring agent. Click **Yes**.
4. The following message is displayed. Click **OK**.

   ```
   Configuration will wait for you to close your default text editor before continuing.
   ```

   The configuration file is opened in your default text editor.
5. Edit the configuration file as follows:
   a. Specify the value of the **agentId** parameter to run more than one instance on a single Windows system. Ensure that the value of the **agentId** parameter is different from that of all the other agent instances running on the same system.
   b. Use the <MonitorBroker> tag to specify which brokers this agent monitors.

   Important: One broker can be monitored by only one agent. If there are two agents monitoring the same broker, the data displayed in workspaces is wrong. For more information about the related parameters, see "agentId" on page 18 and "MonitorBroker" on page 26.
6. Save and close the file.
7. A message is displayed stating that the configuration file edit session is complete. Click **Yes** to configure the agent.
8. Repeat step 2 to step 7, until you finish configuring all the agents and dividing monitoring for all the brokers in your environment.

Dividing broker monitoring between different agents on UNIX and Linux systems

Do the following steps to configure each agent that you want to use to monitor a set of brokers. For each agent, you must specify one broker to be the lead broker, which is used to send commands to the other brokers when the **itmcmd agent stop** and **itmcmd agent start** commands are issued.

1. Optional: Create the secondary instance of the WebSphere Message Broker Monitoring agent using the **itmcmd agent start** command, if you have not done so.

   Example:

   ```
   ./itmcmd agent -o AGT1 -p WBIBRK1 start qi
   ```

   The *hostname*_qi_WBIBRK1_##_AGT1.xml file is created automatically in this example, where *hostname* is the host name of the Tivoli Enterprise Monitoring Server. For more details about how to create a secondary instance, see "UNIX and Linux systems: Creating multiple instances of the monitoring agent" on page 18.
2. Edit the XML file for each agent as follows:

a. Specify the value of the **agentId** parameter to run more than one instance on a single system. Ensure that the value of the **agentId** parameter is different from that of all the other agent instances running on the same system.
b. Use the <MonitorBroker> tag to specify which brokers this agent monitors.

Important: One broker can be monitored by only one agent. If there are two agents monitoring the same broker, the data displayed in workspaces is wrong. For more information about the related parameters, see "agentId" on page 18 and "MonitorBroker" on page 26.

In this example (for the `unixhost01_qi_ WBIBRK1_##_ AGT1.xml` file), the *WBIBRK2* and *WBIBRK3* brokers are added:

```
<KqiAgent version="730"
          agentId="AGT1"
          defaultRetainBrokerEvents="10"
          defaultRetainFlowEvents="10"
          retainProductEvents="10"
          discoveryInterval="300"
          defaultStatisticInterval="60"
          defaultFlowEventInterval="15"
          defaultHistoricalAccountingType="Archive"
          defaultRetainRecentSnapshotSamples="15"
          defaultRetainRecentArchiveSamples="5"
       defaultRetainRecentPubSubSamples="15"
          holdTimeForQuery="180"
          defaultReplyQueueName="KQI.AGENT.REPLY.QUEUE"
          defaultReplyQueueModel="SYSTEM.BROKER.MODEL.QUEUE"
          defaultCollectNodeData="YES"
       maximumMessageLength="10240"
          defaultPersistentBrokerData="NO"
          defaultRefreshInterval="300"
          defaultTakeActionAuthUsers="*">
          <MonitorBroker name="WBIBRK1">
          </MonitorBroker>
          <MonitorBroker name="WBIBRK2">
          </MonitorBroker>
          <MonitorBroker name="WBIBRK3">
          </MonitorBroker>
</KqiAgent>
```

3. Save and close the file.
4. Stop and restart the agent, using the **itmcmd agent stop** and **itmcmd agent start** commands with the **-o** option.

   Example:

   ```
   itmcmd agent -o AGT1 stop qi
   itmcmd agent -o AGT1 start qi
   ```

5. Repeat step 2 to step 4 until you finish configuring all the agent instances and dividing monitoring for all the brokers in your environment.

Disabling broker data collection

In some extreme situations where a broker environment contains a large quantity of data, you might want to disable the collection of broker data.

You can do this by setting the **collectNodeData** parameter of the WebSphere Message Broker Monitoring agent (see "collectNodeData" on page 29 for further information).

When the broker data collection is disabled, there is no data in the following workspaces:

- Message Flow Node Topology workspace
- Processing Node Attributes workspace

In addition, there is only partial data in the following workspaces:

- Archive Accounting Node Statistics workspace
- Snapshot Accounting Node Statistics workspace

Remember: When snapshot accounting is enabled, the queue can become fast when the agent is not running.

Enabling persistent broker data collection

Before you enable persistent data collection, you must delete some files on the system by doing one of the following steps:

- Windows, UNIX, and Linux systems: Check the value of the **CTIRA_HIST_DIR** environment variable and delete all the .xml files within the specified directory.
- z/OS systems: Check the setting of the **persistentDataPath** parameter in the `kqi.xml` agent parameter file. Make sure that all the files within the specified directory are deleted.

To enable persistent broker data collection for all brokers in your environment, do the following steps:

1. Navigate to the installation directory of the WebSphere Message Broker Monitoring agent and open the `kqi.xml` agent parameter file in a standard text editor.
2. Set the **defaultPersistentBrokerData** parameter to `YES`.
3. Set the **defaultReplyQueueModel** parameter to the name of a permanent model queue. Make sure that the specified permanent model queue exists, such as SYSTEM.DURABLE.MODEL.QUEUE, which is automatically created by the broker.
4. Save and close the file.
5. Recycle the WebSphere Message Broker Monitoring agent.

This procedure enables persistent broker data for all brokers in your environment that do not have the **persistentBrokerData** parameter set in the `kqi.xml` file. To disable persistent broker data, set the **defaultPersistentBrokerData** parameter to `NO`.

To enable persistent broker data storage for a single message broker, do the following steps:

1. Navigate to the installation directory of the WebSphere Message Broker Monitoring agent and open the `kqi.xml` agent parameter file in a standard text editor.
2. Locate the MonitorBroker tag for the message broker that you want to enable the persistent broker data collection for.

   Tip: If there is no MonitorBroker tag in the agent parameter file, add it to the file. See "MonitorBroker" on page 26 for detailed information about this tag.
3. Set the **persistentBrokerData** parameter to `YES`.

4. Set the **replyQueueModel** parameter to the name of a permanent model queue. Make sure that the specified permanent model queue exists, such as SYSTEM.DURABLE.MODEL.QUEUE, which is automatically created by the broker.
5. Save and close the file.
6. Recycle the WebSphere Message Broker Monitoring agent.

To disable persistent broker data collection, set the **persistentBrokerData** parameter to NO.

If persistent broker data collection is enabled and the agent is not running, the queue in which event messages are stored gradually becomes full. If the queue becomes full, event messages might be lost, causing inaccuracies in the information displayed in Tivoli Enterprise Portal when the agent is started again.

Remember: If the WebSphere Message Broker Monitoring agent stops abnormally when persistent broker data collection is enabled, error messages might be included in the message broker log when the agent is started again. To avoid this problem, do the following tasks before starting the agent again:

- Using the Message Broker Toolkit, delete the $SYS/Broker/<*broker_label*>/# subscription.
- Delete the reply queue that is used by the WebSphere Message Broker Monitoring agent to store persistent data.

Chapter 3. Monitoring with the CandleMonitor node

The CandleMonitor node is an optional component of the WebSphere Message Broker Monitoring agent. This node collects message flow performance statistics for a broker and provides a mechanism for generating user-defined events within a message flow. These events can be used by situations to detect abnormalities in the message flow. For example, statistics that are generated by a CandleMonitor node can be used to create a situation that automatically stops a message flow that has messages flowing along an abnormal path.

This section provides instructions for installing, enabling, and positioning the CandleMonitor node to monitor message flow statistics and events, turning monitoring off and on, and customizing the CandleMonitor node.

The CandleMonitor node is a simple passthrough node with one input terminal and one output terminal, which is placed in a message flow to collect statistics (see Figure 6). Messages passing through the CandleMonitor node are propagated between its input and output terminals without change. The CandleMonitor node is implemented in the `kqipnode.lil` (or `kqipnode64.lil`) file.

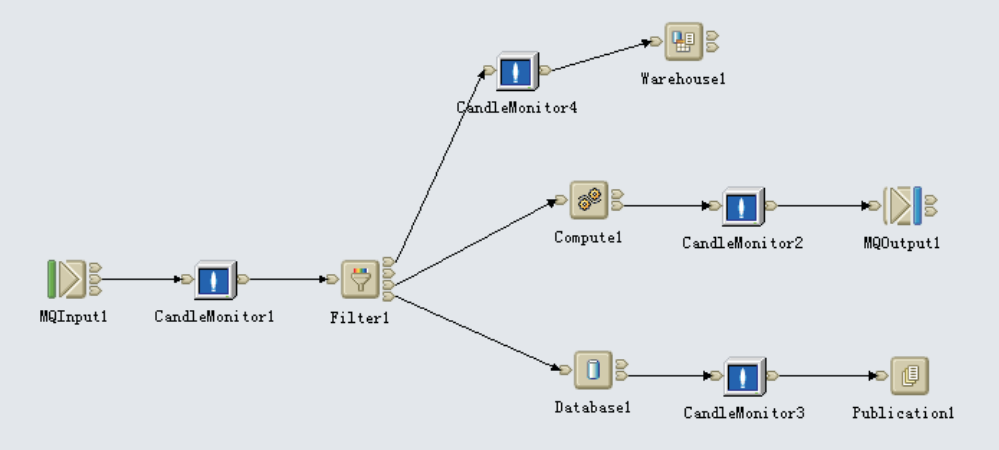

Figure 6. Example of a monitored flow

You can use the CandleMonitor node to take the following actions:

- View message flow reports at broker, execution group, message flow, and node levels
- Detect high input and output rates in message flows
- Detect high average time spent in message flows
- Detect high queue times for input to message flows
- Determine which part of a message flow is taking more time
- View statistics from the last sampling interval and average values taken over a particular period of time
- Be alerted to non-normal or low-use paths
- Detect events and trigger automatic responses

By default, statistics are sampled once per minute and events are sampled once every 15 seconds.

Remember: With broker V5 or later versions, accounting statistics can be made available in the Tivoli Enterprise Portal without using a CandleMonitor node. Some of the accounting statistics are similar to those provided by the CandleMonitor node. However, the CandleMonitor node can also be used to monitor subflows and define message flow events. Message flows without a CandleMonitor node are not represented in the Statistics workspaces.

Prerequisites

Before you can use the CandleMonitor node to monitor message flows, perform the following tasks:

1. Make the CandleMonitor node available in the broker environment.
   - For broker environment on distributed systems, see "Making the CandleMonitor node available in broker environments."
   - For broker environment on z/OS systems, see *IBM Tivoli OMEGAMON XE for Messaging for z/OS: Planning and Configuration Guide*, SC32-1830.
2. Make the CandleMonitor node available in the Message Brokers Toolkit, as described in "Making the CandleMonitor node available in Message Brokers Toolkit" on page 41.

Exception: If the broker to which the CandleMonitor node is deployed runs on AIX systems, enable the shared memory on AIX systems before the node is deployed to the broker. Because the WebSphere Message Broker is a 32-bit application that is memory intensive, the broker cannot attach to more than 11 shared memory segments. When it runs short of process address space, the broker cannot attach to the queue manager-related shared memory resources.

To enable the shared memory on AIX systems, do the following steps:

1. Stop the broker by running the following command, where *broker* is the name of your broker:

   ```
   mqsistop broker
   ```
2. Run the following command to ensure that the broker is running in an environment with the extended memory variable exported:

   ```
   export EXTSHM=ON
   ```
3. Restart the broker by running the following command, where *broker* is the name of your broker:

   ```
   mqsistart broker
   ```
4. On the DB2® server, ensure that the shared memory support is enabled.

Making the CandleMonitor node available in broker environments

This section describes how to make the CandleMonitor node available in your broker environment on distributed systems.

WebSphere Message Broker software includes message flow accounting and statistics that partially overlap with the statistics provided by the CandleMonitor node and can be monitored in Tivoli Enterprise Portal without including the CandleMonitor node in message flows. However, the CandleMonitor node provides additional statistics and the ability to monitor subflows and define message flow events. Message flows without a CandleMonitor node are not represented in Statistics workspaces.

Important: You must use a user ID with root authority when installing the CandleMonitor node.

Follow the corresponding instructions to install the CandleMonitor node in your broker environment, according to the version of broker that you use:

- 32-bit brokers on Windows systems: "Making the CandleMonitor node available in 32-bit broker environments on Windows systems"
- 64-bit brokers on Windows systems: "Making the CandleMonitor node available in 64-bit broker environments on Windows systems"
- brokers on UNIX or Linux systems: "Making the CandleMonitor node available in broker environments on UNIX and Linux systems" on page 40

Making the CandleMonitor node available in 32-bit broker environments on Windows systems

If the broker is 32-bit on a Windows system, do the following steps to install the CandleMonitor node in the broker environment:

1. Verify that the `kqipnode.lil` and `krwnt.dll` files were copied to the `bin` directory of the broker during the installation of WebSphere Message Broker Monitoring agent. If the file was not copied to the `bin` directory of the broker during installation, do it now by performing the following steps:
   a. Stop the broker.
   b. Copy the `kqipnode.lil` and `krwnt.dll` files from the *install_dir*`\TMAITM6\kqipnode32` directory (for 32-bit agent) or from the *install_dir*`\TMAITM6_x64\kqipnode32` directory (for 64-bit agent) to the `bin` directory of the broker.

   Tip: This is an example of a typical `bin` location: `C:\Program Files\IBM\MQSI\7.0\bin`.
2. Verify that the `kqipnode.lil` file in the `bin` directory of the broker *exactly* matches (in size, date, and the time last modified) the one that is in the `kqipnode32` directory.

   Important: A mismatch of the release-level of these files can cause a failure of the broker statistics and message flow events reporting.
3. Start the broker again.
4. Repeat step 1 to step 3 for each Windows system with a broker to be monitored.

The CandleMonitor node is now installed in your broker environment.

Remember: Before you can use the CandleMonitor node to monitor message flows, it must also be made available in the Message Brokers Toolkit. For instructions, see "Making the CandleMonitor node available in Message Brokers Toolkit" on page 41.

Making the CandleMonitor node available in 64-bit broker environments on Windows systems

If the broker is 64-bit on a Windows system, do the following steps to install the CandleMonitor node in the broker environment:

1. Verify that the `kqipnode64.lil`, `kbb.dll`, `klx.dll`, and `krwnt.dll` files were copied to the `bin` directory of the broker during the installation of WebSphere Message Broker Monitoring agent. If the files were not copied to the `bin` directory of the broker during installation, do it now by performing the following steps:

a. Stop the broker.
b. Copy the `kqipnode64.lil`, `kbb.dll`, `klx.dll`, and `krwnt.dll` files from the *install_dir*`\TMAITM6\kqipnode64` directory (for 32-bit agent) or from the *install_dir*`\TMAITM6_x64\kqipnode64` directory (for 64-bit agent) to the `bin` directory of the broker, where *install_dir* is the agent installation directory.

   Tip: This is an example of a typical `bin` location: `C:\Program Files\IBM\MQSI\7.0\bin`.

2. Verify that the `kqipnode64.lil` file in the `bin` directory of the broker *exactly* matches (in size, date, and the time last modified) the one that is in the `kqipnode64` directory.

   Important: A mismatch of the release-level of these files can cause a failure of the broker statistics and message flow events reporting.

3. Check if Microsoft .NET Framework 3.5 is installed on your system. If not, install it now.
   a. Log on the Windows system.
   b. Click **Start** > **Administrative Tools**.
   c. Click **Server Manager**.
   d. In the Server Manager window, click **Features** on the left navigation pane.
   e. On the right side of the window, click **Add Features**.
   f. In the Select Features window, select **.NET Framework 3.5 Features** and click **Install**.
4. Start the broker again.
5. Repeat step 1 to step 4 for each Windows system with a broker to be monitored.

The CandleMonitor node is now installed in your broker environment.

Remember: Before you can use the CandleMonitor node to monitor message flows, it must also be made available in the Message Brokers Toolkit. For instructions, see "Making the CandleMonitor node available in Message Brokers Toolkit" on page 41.

Making the CandleMonitor node available in broker environments on UNIX and Linux systems

To make the CandleMonitor node available in a broker environment running on a UNIX or Linux system, do the following steps:

1. Stop the broker if it is running.
2. Navigate to the `/usr/bin` directory.
3. Run the following commands to create a set of soft links:

```
(for 32-bit execution groups)ln -sf install_dir/arch/qi/bin/kqipnode.lil
        broker_dir/lil32_dir/kqipnode.lil
(for 64-bit execution groups)ln -sf install_dir/arch/qi/bin/kqipnode.lil
        broker_dir/lil64_dir/kqipnode.lil
ln -sf install_dir/arch/qi/bin/kqipnode.cfg
        broker_dir/lil/kqipnode.cfg
ln -sf install_dir/arch/qi/bin/langcode/CandleMonitorNode.cat
        broker_dir/messages/langcode/CandleMonitorNode.cat
```

   where:

   - *arch* is the architecture code of the operating system, see Appendix D, "Architecture codes," on page 139 for a list of architecture codes.
   - *broker_dir* is the full path of the broker installation directory.

- *install_dir* is the full path of the IBM Tivoli Monitoring installation directory.
- *langcode* is the code for the installed language pack, see Appendix C, "Language codes," on page 137 for a list of language codes.
- *lil32_dir* is the message broker directory for 32-bit plugin node.
- *lil64_dir* is the message broker directory for 64-bit plugin node.

4. Create another link by running one of the following commands depending on which operating system you are using:
   - On AIX systems:

     ```
     ln -sf install_dir/arch/qi/bin/langcode/CandleMonitorNode.cat
             /usr/lib/nls/msg/langcode/CandleMonitorNode.cat
     ```

   - On Solaris systems:

     ```
     ln -sf install_dir/arch/qi/bin/langcode/CandleMonitorNode.cat
             /usr/lib/locale/C/LC_MESSAGES/CandleMonitorNode.cat
     ```

   - On HP-11 systems:

     ```
     ln -sf install_dir/arch/qi/bin/langcode/CandleMonitorNode.cat
             /usr/lib/nls/msg/C/CandleMonitorNode.cat
     ```

   - On Intel-based Linux systems:

     ```
     ln -sf install_dir/arch/qi/bin/langcode/CandleMonitorNode.cat
             /usr/share/locale/C/LC_MESSAGES/CandleMonitorNode.cat
     ```

   - On Linux for zSeries systems:

     ```
     ln -sf install_dir/arch/qi/bin/langcode/CandleMonitorNode.cat
             /usr/share/locale/C/LC_MESSAGES/CandleMonitorNode.cat
     ```

5. Optional: To take full advantage of 64-bit architectures, if you are running the CandleMonitor node on a 64-bit AIX, Solaris, or HP-UX (PA-RISC, not Itanium) systems, you must also create links to an additional 64-bit version of the CandleMonitor node file. To do this, repeat step 2, replacing the `kqipnode.lil` file name with `kqipnode64.lil`.
6. Start the broker again.

Remember: After the CandleMonitor node is installed, it must be made available in the Message Brokers Toolkit. For instructions, see "Making the CandleMonitor node available in Message Brokers Toolkit."

Making the CandleMonitor node available in Message Brokers Toolkit

Before you can insert the CandleMonitor node into message flows, you must make the node available in the Message Brokers Toolkit.

To make the CandleMonitor node available in Message Brokers Toolkit V7.0 or later, do the following procedure:

1. Exit the Message Broker Toolkit if it is open.
2. Find the CandleMonitor node file. The file name and location vary depending on the operating system where the WebSphere Message Broker Monitoring agent is running:
   - Windows systems: *install_dir*`\TMAITM6\kqicm700.exe` for 32-bit agent or *install_dir*`\TMAITM6_x64\kqicm700.exe` for 64-bit agent
   - UNIX or Linux systems: *install_dir*`/arch/qi/bin/kqicm700.tar`

   where *install_dir* is the installation directory of the WebSphere Message Broker Monitoring agent; *arch* specifies the architecture code of the operating system of this computer. For a list of architecture codes, see Appendix D, "Architecture codes," on page 139.

3. Extract the CandleMonitor node file. Two subdirectories are included in this file:
   - `com.candle.monitor_7.0.0` (contains all the necessary plugin support files)
   - `com.candle.monitor.nl1_7.0.0` (contains all the native language support files)
4. Copy the two subdirectories (`com.candle.monitor` and `com.candle.monitor.nl1`) from the toolkit `workspace` folder to the following folder depending on the operating system where the toolkit is installed:
   - Windows systems: *WMBT_installdir*`\plugins`
   - UNIX or Linux systems: *WMBT_installdir*`/plugins`

   where *WMBT_installdir* is the installation directory of WebSphere Message Broker Toolkit.
5. Restart the Message Broker Toolkit.

Now you can see the CandleMonitor node in the message flow editor palette under the IBM category in the Broker Application Development perspective.

Placing the CandleMonitor node in message flows

The primary use of the CandleMonitor node is to produce the statistics that are displayed in the high-level statistics workspaces in Tivoli Enterprise Portal. To ensure that the correct statistics are gathered for the following workspaces, the CandleMonitor node must be correctly placed in message flows:

- Monitor Node Broker Statistics workspace
- Monitor Node Base Statistics workspace
- Monitor Node Execution Group Statistics workspace
- Monitor Node Events workspace
- Monitor Node Message Flow Statistics workspace

For information about how to place the CandleMonitor node for monitoring different aspects of the message flow, see the following instructions:

- "Monitoring the input or output of a message flow"
- "Monitoring subflows" on page 43
- "Monitoring other aspects of a message flow" on page 46

A secondary use of the CandleMonitor node is to produce user-defined message flow events that can be detected by Message Flow Event situations and viewed in the Monitor Node Events workspace. For information about how to use the CandleMonitor node for this purpose, see "Producing event messages" on page 46.

Important: You must provide unique label names to CandleMonitor nodes in a message flow. The names must be unique with respect to any other names in the entire message flow, including its subflows. If the CandleMonitor node label names are not unique, data for different nodes is combined in illogical ways. Do not use the default names that are assigned to your CandleMonitor nodes.

Monitoring the input or output of a message flow

When you are designing a message flow that is to be monitored by a broker, you must place a CandleMonitor node immediately after the MQInput node, so that the CandleMonitor node can gather the information that is required for input rate calculations. Another important location for placing the CandleMonitor node is

immediately in front of any MQOutput, MQReply, or Publication nodes, so that the CandleMonitor node can gather the information that is required to calculate output rate calculations.

Tip: If you want to place only a single CandleMonitor node in a message flow, the input position is the best choice, because most statistics are generated from this position.

To monitor the input of a message flow, perform the following procedure:

1. Place a CandleMonitor node immediately after the MQInput node.
2. Set the **type** attribute of the CandleMonitor node placed in this position to `input`.
3. Optional: If the input queue to a message flow contains messages that have put date and times that do not accurately reflect when the message was put into the input queue, and you are using a CandleMonitor node with the **type** attribute set to `input`, set the **collectQueueTime** attribute to `no`, so that queue times are not calculated for these messages. When queue time statistics are determined, queue times are calculated using the put date and time of the message in the queue. Put date and times are not accurate when origin context is preserved for a message during the put operation by the application performing this operation. Inaccuracies of put date and time commonly occur when an application is a message mover that transfers messages from one queue to another, or when an application passes or sets origin context for a message. If the input queue to *Message Flow B* is the output queue of *Message Flow A*, the broker passes the origin context so that the put date and time for the message in *Message Flow B* is not an accurate calculation of queue time. The **collectQueueTime** attribute for a CandleMonitor node placed in *Message Flow B* must be set to `no`.

To monitor the output of a message flow, do the following steps:

1. Place the CandleMonitor node immediately in front of any MQOutput, MQReply, or Publication nodes.
2. Set the **type** attribute of a CandleMonitor node placed in this position to `output`.

Monitoring subflows

A subflow is a section of a message flow that includes one or more message processing nodes. In general terms, a subflow can be any section of a message flow that can be separately identified. You can consider the message flow as being like a main routine and a subflow as being a subroutine. You can explicitly delineate the subroutine in the Message Brokers Toolkit by making a separate message flow, which is then embedded in the main flow (referred to here as *Type I*). Or, the message flow can have sections of nodes that you want to monitor as subflows, even though they are not explicitly delineated into a separate flow (referred to in this example as *Type II*).

When a message flow is deployed to a broker, the broker regards the entire message flow as a single entity. There is no obvious delineation in the broker for dividing the flow into separate subflows. The name assigned to a Type I subflow is not known to the broker (this entity is displayed only in the Message Brokers Toolkit and configuration manager). Any given message processing node is not aware of the other message processing nodes around it. Therefore, for the CandleMonitor node to be useful in monitoring both types of subflows, you must provide the required information by customizing the node.

To gather correct subflow statistics for either type of subflow, the subFlowOutput CandleMonitor node is required in the subflow; it is not optional, as is the case with the output node for a main message flow.

CandleMonitor node Statistics (the lowest level, most detailed report) combines data for all instances of the same node that are part of a subflow that has been embedded multiple times in the same message flow.

You can assign the **subFlowName** attribute to an input or output CandleMonitor node. When you use input and output node types in combination you do not need to insert two CandleMonitor nodes in a flow at the same position when a subflow comes either at a beginning or end of a message flow. The combination output node is probably used more, because a Type I subflow might have an output that is the destination for a message going through the whole message flow as well as through the subflow. (For a description of the effects of assigning a **subFlowName** attribute to a node type other than subFlowInput or subFlowOutput, see "subFlowName" on page 50.)

Monitoring a Type I subflow

To monitor a Type I subflow, do the following procedure:

1. Place a CandleMonitor node immediately after the Input Terminal in the flow, and place another CandleMonitor node immediately in front of the Output Terminal or other output of the flow (see Figure 7 on page 45).
2. Assign the same value (`subFlow1` in Figure 7 on page 45) to the **subFlowName** attribute of all nodes in the subflow.

   Tip: For Type I message flows, you can set the **subFlowName** attribute to the name that was specified in the Message Brokers Toolkit for the message flow that represents the subflow.
3. Set the **type** attribute of the CandleMonitor node placed in the input position to `subFlowInput`.
4. Set the **type** attribute of the CandleMonitor node placed in the output position to `subFlowOutput`.

   Exception: If the output is a node such as MQOutput and represents the end of the message flow for a message going down a particular path, set the **type** attribute of the CandleMonitor node placed in the output position to `output`.

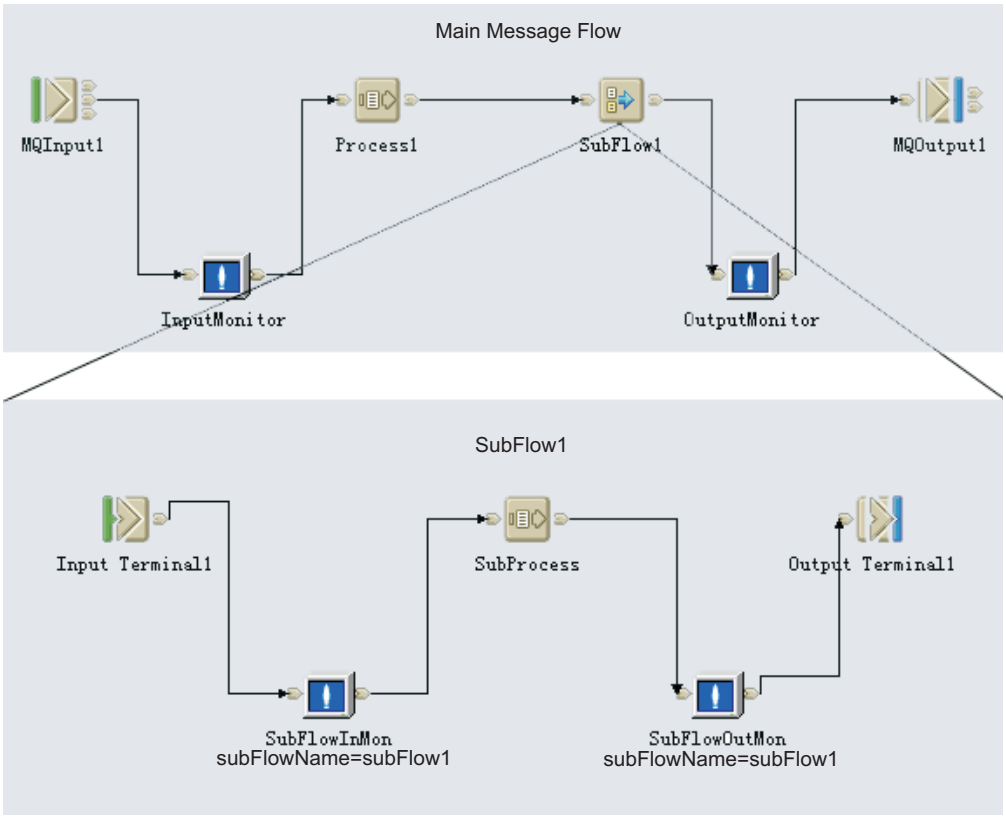

Figure 7. Type I subflow

Remember: Subflows embedded in subflows, or nested subflows, are supported. However, the **subFlowName** attribute for any nested subflows must be different for different subflows. You must be aware of the names assigned to subflow monitoring nodes and not use them again.

Monitoring a Type II subflow

To monitor a Type II subflow, do the following steps:

1. Place a CandleMonitor node at any designated starting place in the message flow.
2. Set the **type** attribute of the CandleMonitor node to `subFlowInput`, and specify the **subFlowName** attribute.

   Remember: For Type II message flows, you need to use a different name that uniquely describes what is being monitored from the name that was specified in the Message Brokers Toolkit for the message flow to represent the subflow.
3. Place another CandleMonitor node at a corresponding end point in the message flow.
4. Set the **type** attribute of the CandleMonitor node to `subFlowOutput`, and specify the **subFlowName** attribute. (see Figure 8 on page 46)

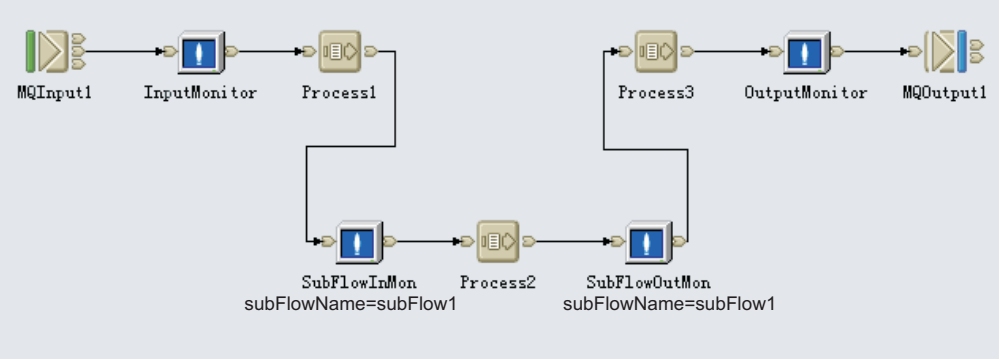

Figure 8. Type II subflow

Remember:

- If the section of message flow that is to be monitored has multiple input connectors or multiple output connectors, you must add multiple CandleMonitor nodes in the same way with the same **subFlowName**.
- Subflows embedded in subflows, or nested subflows, are supported. However, the **subFlowName** attribute for any nested subflows must be different for different subflows. You must be aware of the names assigned to subflow monitoring nodes and not use them again.

Monitoring other aspects of a message flow

You can place the CandleMonitor node anywhere in a message flow and gather statistics for that particular portion of the message flow.

To monitor aspects of a message flow other than the input or output, perform the following procedure:

1. Place a CandleMonitor node in a place other than the beginning or end of a flow.
2. Set the **type** attribute of the CandleMonitor node to `other`. The statistics can be viewed in the low-level statistics and Monitor Node Base Statistics workspace in Tivoli Enterprise Portal.

Producing event messages

You can use the CandleMonitor node to produce user-defined message flow events that can be detected by Message Flow Event situations and viewed in the Monitor Node Events workspace. The WebSphere Message Broker Monitoring agent includes other information about events to help identify which portion of the message flow is affected, as well as to isolate which message is being processed at the time the event occurs.

To use the CandleMonitor node to produce user-defined message flow events, do the following steps:

1. Place a CandleMonitor node at any designated place.

   Tip: Do not place the CandleMonitor node in portions of the message flow that are used regularly during typical processing, because this might produce a large number of events. Place the CandleMonitor node in paths of a message flow that represent processing failures or other irregular conditions that

warrant an alert notification. In this capacity, the CandleMonitor node serves only as an alert mechanism; it cannot perform error recovery because it is a pass-through node only.

2. Set the **type** attribute of the CandleMonitor node to `other`.
3. Set the **eventMessage** attribute to the message text that you want to display in the workspace when a message enters the CandleMonitor node.

   Tip: Do not set the **eventMessage** attribute for any input or output type node, because this setting might produce an unnecessarily large number of events.

Guidelines for monitoring with the CandleMonitor node

See the following guidelines for monitoring with the CandleMonitor node:

- For production flow statistics, place the CandleMonitor node immediately after MQInput (or other input) nodes and set the **type** attribute to `input`.
- For output rates and counts, place the CandleMonitor node immediately in front of MQOutput, MQReply, and Publication nodes and set the **type** attribute to `output`.
- For any other placement in the flow, do not set the **type** attribute to `input` or `output`, because inaccurate data might be generated.
- Generally, an **eventMessage** attribute must be specified only for nodes with a type attribute of `subFlowInput`, `subFlowOutput`, and `other`. Do not set this attribute for the typical processing part of a flow or for a node with a type attribute of input or output (unless the node is not used frequently and there is a specific reason for doing so) because it might have an adverse effect on system performance.
- For message flow development, nodes that have the type of other can be used to determine areas of a flow that have poor performance.
- If you are going to shut down the broker or agent, the broker or agent must complete the entire shutdown process whenever possible.

Customizing a CandleMonitor node

Some aspects of the CandleMonitor node behaviors are controlled by configuration variables. You can modify the values of these variables to suit your requirements. After changing the value of any of these variables, you must restart the broker before the changes can take effect.

Important: All of the `kqipnode` variable names and values are case sensitive. You must enter the variable names exactly as shown. For example, if you are specifying the no override value for the **KQIActivateNode** attribute, you must specify `NoOverride`.

Windows systems: Changing the values of configuration variables

On Windows operating systems, configuration variables are available in the registry that is created when you configure the monitoring agent.

To change the values of configuration variables on Windows systems, do the following steps:

1. Stop the WebSphere Message Broker Monitoring agent that has variables you want to edit.

2. From the **Start** menu, click **Programs** > **IBM Tivoli Monitoring** > **Manage Tivoli Monitoring Services**.
3. Right-click the WebSphere Message Broker Monitoring agent and click **Advanced** > **Edit Variables**.
4. Do one of the following procedures, depending on whether the variable that you want to edit is in the list:
   - If the variable is in the list, do the following steps:
     a. Select the variable and click **Edit**.
     b. Enter the new value in the **Value** field and click **OK**.
   - If the variable is not in the list, do the following steps:
     a. Click **Add**.
     b. In the **Variable** list, click the arrow to display the list.
     c. Select the variable that you want to add from the variable list.
     d. Enter the new value in the **Value** field and click **OK**.

   See "Configuration variables" on page 51 for descriptions of the variables and their valid values.
5. Click **OK**.
6. You are prompted if you want to update the `kqi.xml` file before you configure the WebSphere Message Broker Monitoring agent. Click **No**.

Important: All of the `kqipnode` variable names and values are case sensitive. You must enter the variable names exactly as shown. For example, if you are specifying the no override value for the **KQIActivateNode** attribute, you must specify `NoOverride`.

UNIX or Linux systems: Changing the values of configuration variables

On UNIX and Linux operating systems, the configuration variables are stored in the `kqipnode.cfg` file in the *<install_dir>*`/`*<arch_code>*`/qi/bin` directory, where *<install_dir>* is IBM Tivoli Monitoring installation directory and *<arch_code>* is architecture code for the operating system on which the agent is running. For a list of architecture codes, see Appendix D, "Architecture codes," on page 139.

To change the values of configuration variables on UNIX or Linux systems, edit the `kqipnode.cfg` file directly. The format of the configuration file is as follows:

```
variable value
variable value
```

See "Configuration variables" on page 51 for descriptions of the variables and their valid values.

Important: All of the `kqipnode` variable names and values are case sensitive. You must enter the variable names exactly as shown. For example, if you are specifying the no override value for the **KQIActivateNode** attribute, you must specify `NoOverride`.

Deleting the CandleMonitor node from Message Brokers Toolkit

To delete the CandleMonitor node, do the following steps:

1. Close the toolkit if it is active.

2. Perform one of the following steps, depending on which type of system the Message Brokers Toolkit is installed on:
   - If the Message Brokers Toolkit is installed on a Windows system, delete any subdirectories with names beginning with `com.candle.monitor` from the following directory (or an alternative location if the Message Brokers Toolkit is not installed in the default location), where *WMBT_installdir* is the Message Brokers Toolkit installation directory.

     WMBT_installdir`\evtoolkit\eclipse\plugins`
   - If the Message Brokers Toolkit is installed on a Linux or UNIX system, delete any subdirectories with names beginning with `com.candle.monitor` from the following directory (or an alternative location if the Message Brokers Toolkit is not installed in the default location), where *WMBT_installdir* is the Message Brokers Toolkit installation directory.

     WMBT_installdir`/evtoolkit/eclipse/plugins`

CandleMonitor node attributes and configuration variables

This section introduces the attributes and configuration variables related to the CandleMonitor node.

Attributes

You can set the attributes of the CandleMonitor node when you place it in a message flow. The attribute values that you set determine how the node operates.

type

The **type** attribute specifies the type of statistics that are gathered. Valid values are:

- input

  Identifies the node as a main message flow input node that records the entry of messages into the message flow. For example, specify the **type** attribute to input when placing the CandleMonitor node immediately after the MQInput node.
- output

  Identifies the node as a main message flow output node that records the exit of messages from the message flow. For example, specify the **type** attribute to output when placing the CandleMonitor node immediately in front of MQOutput, MQReply, or Publication node.
- subFlowInput

  Identifies the node as a subflow input node
- subFlowOutput

  Identifies the node as a subflow output node
- other

  Identifies the node as being used for other purposes, such as temporary collection of statistics or message flow event generation. For example, specify the **type** attribute to other when placing the CandleMonitor node wherever necessary to debug your message flows.

collectQueueTime

The **collectQueueTime** attribute specifies if the CandleMonitor node collects queue times. Valid values are:

- no

  Queue times are not collected for these messages.
- yes

Queue times are collected for these messages.

Queue times are calculated by using the *put date and time* (the time and date at which the message was put into the queue) of the messages in the queue. If the input queue to a message flow has messages with put date and times that do not reflect accurately when the message was put into the input queue, set the **collectQueueTime** attribute to `no` so that queue times are not collected for these messages. (See also "Monitoring the input or output of a message flow" on page 42.)

eventMessage

The **eventMessage** attribute is used to produce user-defined message flow events that can trigger Message Flow Events situations and reports. If this attribute is set to anything other than cleared (""), an event is produced when a message enters the CandleMonitor node.

When a CandleMonitor node is placed for generating events, set the **type** attribute to `other` and the **eventMessage** attribute to the message text that appears in Tivoli Enterprise Portal workspaces.

By default this value is cleared ("") and no events are produced.

subFlowName

The **subFlowName** attribute specifies an identifying name for the subflow. This attribute is required for CandleMonitor nodes which are of type subFlowInput and subFlowOutput; otherwise, these nodes cannot produce correct subflow statistics without the **subFlowName** attribute specified.

You can specify the **subFlowName** attribute for other types of nodes. The following table summarizes the effect of specifying the **subFlowName** attribute for each node type.

Table 1. The effect of providing a subFlowName attribute for each node type

Value of the type attribute	subFlowName attribute requirement	Effect
input	Optional	The node is a combination node; it marks not only the beginning of a message flow, but also the beginning of a subflow in the message flow.
output	Optional	The node is a combination node; it marks not only the end of a message flow, but also the end of a subflow within the message flow.
other	Optional	The node is a part of the subflow. This combination of settings has little effect. However, it causes the CandleMonitor node to be displayed in the list of CandleMonitor nodes for the named subflow. Statistics for this node are displayed in the Monitor Node Base Statistics workspace.
subFlowInput	Required	The subFlowInput type node marks the beginning of a monitored subflow and requires a name so that statistical data is correlated with the subflow. There can be multiple inputs to a subflow; for multiple inputs, use the same subFlowName for each node.
subFlowOutput	Required	The subFlowOutput type node marks the end of a monitored subflow and requires a name so that statistical data is correlated with the subflow. There can be multiple outputs for a subflow; for multiple outputs, use the same subFlowName for each node.

For more information about placing CandleMonitor nodes and required attribute values, see "Monitoring subflows" on page 43.

activateNode

The **activateNode** attribute controls the activation of instances of a CandleMonitor node. Valid values are:

- yes

  The CandleMonitor node is active. This setting is appropriate for a test environment.
- no

  The CandleMonitor node is not active.
- eventOnly

  The CandleMonitor node is activated only if the eventMessage attribute is specified. The node does not produce statistics; it produces only message flow events. This setting is useful when you are interested in message flow events, but you do not want to gather statistics.
- inputOutputOnly

  The CandleMonitor node is activated if the node type is input or output. This setting produces statistics (and events, if the eventMessage attribute is set).
- inputOutputAndEventOnly

  The CandleMonitor node is activated if the node type is input or output, or if the eventMessage attribute is set. Statistics are produced; events are produced also if the eventMessage attribute is set. This setting is useful in a production environment when you want message flow statistics and event messages, but you do not want any of the additional details provided by other node types.

If you are monitoring WebSphere Business Integration broker 5.0 or later, you can configure the activateNode attribute. Set this attribute when you are configuring bar files, so the attribute can have a different setting for different bar files in test and production environments. In addition, with version 5.0 you can promote the attribute setting so that this attribute has the same setting for each CandleMonitor node in a message flow.

The set of possible values for the attribute supports specification of some general rules for when the node must be active; for example, it can be active for nodes of type input and output only. In general, when you are customizing a CandleMonitor node in a message flow, use the default value (`yes`) to facilitate node usage during testing. And then, when you are deploying the node to a production broker, configure the bar file setting for the **activateNode** attribute to the level that you want and promote the attribute to automatically change the settings of all nodes in the flow to that activation level.

You can specify a runtime override setting for the **activateNode** attribute, which allows different activation levels for different brokers (see "KQIActivateNode" on page 52).

Configuration variables

The following sections provide a detailed description of some configuration variables.

KQIMemorySize

The **KQIMemorySize** variable specifies the size (in bytes) of a shared memory segment created by the plug-in. The CandleMonitor node creates three of these shared memory segments for holding different types of data. The default value is 32768 bytes. Do not modify this variable unless instructed to do so by the IBM Software Support.

KQITempDirectory

The **KQITempDirectory** variable is not applicable on Windows systems.

The **KQITempDirectory** variable specifies the directory to be used for shared memory and mutexes. The broker and the agent must have the required authority to read, write and create files in this directory. Its default value is `/tmp`.

By default, temporary files used by the CandleMonitor node and related to mutexes and shared memory are stored in the `/tmp` directory. The user IDs used by the broker and the WebSphere Message Broker Monitoring agent must both have write access to this directory. If the user IDs do not have this access, you can use the **KQITempDirectory** variable in the `kqi.xml` file to specify a different directory for storing this information.

You must also use this variable to specify a different location if there is a possibility that the `/tmp` directory becomes full or when the `/tmp` directory is used by a large number of applications, because this might result in files being accidentally overwritten by other programs.

The only safe time to delete files with file names containing **kqi** from the directory specified in the **KQITempDirectory** variable is after you shut down all brokers and WebSphere Message Broker Monitoring agents on the system. The **KQITempDirectory** variable is used by all brokers and WebSphere Message Broker Monitoring agents on the system, so they must all be stopped before you can delete any files that have **kqi** in the name. Under typical circumstances, you do not need to delete the files because when all the brokers and agents are stopped, they are not present. However, in exceptional circumstances where the files still exist after all the components have stopped, you can delete the files manually. These files implement shared memory and mutex, and WebSphere Message Broker Monitoring agent might not behave normally if you delete them while the processes that use the files are active.

KQINodeTrace

The **KQINodeTrace** variable determines whether `kqipnode.lil` tracing is turned on. Valid values are Off and On. The default is Off. For tracing to occur, the broker normal tracing must be turned on for one or more of the applicable broker, execution group, message flow. Set this value to On only when instructed to do so by the IBM Software Support.

KQIActivateNode

Use this variable to override the CandleMonitor node **activateNode** attribute of any broker on the system. If a value other than the default `NoOverride` is specified, that value overrides the value set for the attribute for allCandleMonitor nodes deployed on a broker.

The possible values are:

- NoOverride

  The value of the **activateNode** attribute that is deployed for each instance of the CandleMonitor node is honored; no override takes place. This is the default setting.
- Yes

  Every instance of the CandleMonitor node is active.
- No

All instances of the CandleMonitor node are inactive. No statistics or message flow events are generated.

- EventOnly

  An instance of the CandleMonitor node is active only if the **eventMessage** attribute is assigned. No statistics are produced; only message flow events are produced.

- InputOutputOnly

  An instance of the CandleMonitor node is active only if the type is input or output. Statistics are generated (and events, if the **eventMessage** attribute is assigned).

- InputOutputAndEventOnly

  Instances of the CandleMonitor node are active only if they are of type input or output, or if the **eventMessage** attribute is assigned. Statistics are produced, and message flow events are produced, if the **eventMessage** attribute is assigned.

KQIActivateNodeForBROKERNAME

Use this variable to override the CandleMonitor node **activateNode** attribute of any one broker on the system. If this variable is specified, the CandleMonitor node that is running in the *BROKERNAME* broker uses this setting instead of the setting specified in the **KQIActivateNode** attribute.

Important: This variable name does not exist by default; you must add it to the `kqipnode.cfg` file (UNIX, Linux, or z/OS systems) or type it in the variable field instead of selecting it from the list (Windows systems).

WebSphere Message Broker V8 or later support: Known limitations and problems

The following limitations or problems are identified when you use the CandleMonitor node with WebSphere Message Broker V8 or later.

CandleMonitor node statistics is not available

Limitation: When a CandleMonitor node is placed in a message flow and the message flow with the same name appears multiple times within an execution group, the agent cannot get statistical data to display in the following workspaces:

- Monitor Node Base Statistics
- Monitor Node Broker Statistics
- Monitor Node Event Statistics
- Monitor Node Execution Group Statistics
- Monitor Node Message Flow Statistics
- Monitor Node Sub-Flow Statistics

For example, a CandleMonitor node is placed in a message flow name MF1. This MF1 message flow is deployed directly in an execution group named EX1 and also in an application within the EX1 execution group. In this case, the MF1 message flow appears twice within the EX1 execution group, which will cause the problem.

To avoid this problem, if a message flow is deployed with a CandleMonitor node, deploy it only once in an execution group. For example, if you deploy this message flow directly within an execution group, make sure that other containers (application or library) in this execution group do not contain this message flow.

CandleMonitor node cannot be placed in deployable subflows

Problem: CandleMonitor node cannot be placed in deployable subflows using WebSphere Message Brokers Toolkit 8.0.

Solution: This problem happens when you use WebSphere Message Brokers Toolkit 8.0.0.0. To use CandleMonitor node in deployable subflows, apply the fix of PMR 63949,000,672 or upgrade to WebSphere Message Brokers Toolkit 8.0.0.1 or later.

Chapter 4. Using situations and Take Action commands

This section describes the predefined situations and Take Action commands that are included with WebSphere Message Broker Monitoring agent.

Predefined situations

The WebSphere Message Broker Monitoring agent provides two types of predefined situations, agent-level situations and broker-level situations. Most predefined situations are designed to help you to monitor critical activities and serve as templates for creating customized situations for your own use. There are two agent-level situations, QI_Automation_Start_Component and QI_Product_Events. The broker-level predefined situations for this agent have different names according to the version of brokers. The situation names begin with the letters **WMB** for brokers of version 7.0 or later.

- WMB_Average_Flow_Time_High
- WMB_Broker_Not_Started
- WMB_Broker_QMgr_Not_Connected
- WMB_Exception_Terminal_Invoked
- WMB_Monitor_Node_Events
- WMB_MsgFlow_Elapsed_Time_High

Predefined situations are activated once they are distributed to the node that you wish to monitor. Once they are configured correctly, the situation alerts provided with the WebSphere Message Broker Monitoring agent will trigger event notification. You can edit these situations to better reflect your site-specific standards or requirements.

Remember: When you distribute a situation to the managed system, in the available managed system group list, `MQSI_BROKER_V7` is the group of all managed systems for WebSphere Message Broker V7.0 or later.

Tip: If you choose to modify a predefined situation, make a copy first to ensure fallback if necessary.

Alerts

The following situations issue alerts based on critical status criteria. Except where noted, these situations apply to broker-managed systems. Alerts are raised if the following conditions are met:

- A Product Events row occurs. This situation applies to agent-managed systems. (QI_Product_Events)
- The Component.Component Status is found to be not started. If the situation becomes true, the proper command is issued to start the component. This situation applies to agent-managed systems. (QI_Automation_Start_Component)
- The Message Flow Statistics.Average Flow Seconds value is found to exceed a threshold. (WMB_Average_Flow_Time_High)
- The Broker Information.Broker Status value is found to be not started. (WMB_Broker_Not_Started)

- The Broker Information.Queue Manager Status value is found to be not connected. (WMB_Broker_QMgr_Not_Connected)
- Messages have gone down a failure path in a message flow. You can create a new situation based on this situation using parameters that are appropriate for your environment. In this case the links included in the predefined situation are invalid. You can create a new link using the Link Wizard. (WMB_Exception_Terminal_Invoked)
- A Message Flow Events row occurs. (WMB_Monitor_Node_Events)
- The maximum time that any message takes to go through a given message flow during an archive statistics interval exceeds 100 milliseconds. You can create a new situation based on this situation using a threshold value that is appropriate for your environment. In this case the links included in the predefined situation are not valid. You can create a new link using the Link Wizard. (WMB_MsgFlow_Elapsed_Time_High)

Take Action commands

You can use Take Action commands to send commands from the Tivoli Enterprise Portal to systems in your managed enterprise. You can use it to, for example, start or stop a component. These commands otherwise must be entered from a command prompt, or from the Message Brokers Toolkit.

You can issue the commands from the Take Action view, from the situation window when a situation becomes true, from the Navigator, or from a row in a table view. For more information about how to run a Take Action command, see Tivoli Enterprise Portal online help.

The Take Action commands for the WebSphere Message Broker Monitoring agent have names beginning with the characters **QI**.

For more details about the Take Action commands included in the WebSphere Message Broker Monitoring agent, see *IBM Tivoli Composite Application Manager Agent for WebSphere Message Broker Reference*.

Take Action user authorization

You can configure the WebSphere Message Broker Monitoring agent so that only a designated set of users can issue Take Action commands.

Use the following parameters to specify authorized Take Action users:

- At the agent level, use the **defaultTakeActionAuthUsers** parameter.
- At the monitored broker level, use the **takeActionAuthUsers** parameter.

See “Agent parameter descriptions” on page 18 for details.

Sending a Take Action command

You can use the take action feature to issue a command on a managed system in your monitored enterprise. For example, you can restart a broker that is not functioning properly.

Your user ID must have the authorization to send the Take Action commands that are associated with the WebSphere Message Broker Monitoring agent. The user ID must be identifiable and authorized on the system where the action will take place.

Use Tivoli Enterprise Portal to send a Take Action command to the systems in your managed enterprise:

1. Select the Navigator item that is associated with the component or application on which you want to run the command.
2. Right-click the Navigator item. You can also right-click a row in a table view or a bar in a bar chart.
3. Click **Take Action** > **Select**. The Take Action window is displayed.
4. In the **Name** field, click the arrow to display the list.
5. Select the command that you want to run from the **Name** list. For example, **QI Start Broker**.
6. In the Edit Argument Values window, enter the name of the component on which you want to run the command in the **Value** column. For example, if you chose **QI Start Broker** in the previous step, enter the name of the broker that you want to start. Click **OK**.
7. Click the host system in **Destination Systems** where the component is located and click **OK**.

A message is displayed indicating the status of the action after the Take Action command is sent.

Take Action commands in situations

You can use Take Action commands in situations that you create. The *reflex automation* term refers to a situation that issues a command.

To issue a command to an agent from a situation, enter the `QI:`*`syntax`* form of the command on the **Action** page of the Situations Editor. Figure 9 illustrates an example of using the **`QI Start Component`** command in a situation.

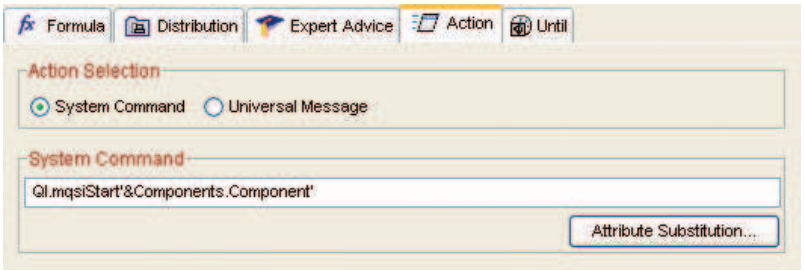

Figure 9. Example of using the QI Start Component command in a situation

This example displays the predefined situation, QI_Automation_Start_Component, which substitutes an attribute from the Components workspace of the product. The **Attribute Substitution** button is used to choose the attribute (this prevents misspellings and inserts the ampersand character which is used for attribute substitution). The single quotation marks enclosing the command parameter are required. When you are using a Take Action command in a situation, create a working, manual version of the Take Action command before you attempt to automate its use by adding it to a situation.

Scenarios of using Situations and Take Action commands

This section provides several examples of monitoring situations that are common to many broker product environments, and illustrates how situations and Take Action commands can be used in these circumstances.

Each scenario presents a case in which you want to monitor some aspect or component of your broker product environment, followed by a description of how to set up the WebSphere Message Broker Monitoring agent to gather important information.

Preventing inadvertent use of trace active

This scenario describes how you can use the WebSphere Message Broker Monitoring agent to prevent the trace active feature from being used inadvertently and adversely affecting broker performance.

After you have configured the WebSphere Message Broker Monitoring agent, start the agent so that the necessary statistics can be gathered, and then perform the following steps:

1. Define a situation for the following workspaces that verifies that the Trace Level and User Trace Level attribute values are not `none`. An alert is triggered whenever tracing is active so that you can determine whether the tracing activity is valid.
   - Broker Status
   - Execution Group Status
   - Message Flow Status
2. Deploy the situation to the broker managed systems where you want to detect active trace.

Determining when a message flow has failed

This scenario describes how you can use the WebSphere Message Broker Monitoring agent to determine when a message flow has failed and to notify you of a failure in a message flow.

To determine when a message flow has failed, do the following steps so that the WebSphere Message Broker Monitoring agent can gather the appropriate statistics:

1. Place CandleMonitor nodes after failure terminals for processing nodes in the message flow.
2. Set the **`type`** attribute of these CandleMonitor nodes to `other`.
3. Describe the failures in a meaningful way in the **`eventMessage`** attribute of the CandleMonitor nodes.
4. Start the WebSphere Message Broker Monitoring agent again.

After you have positioned the CandleMonitor nodes and the WebSphere Message Broker Monitoring agent is running, do the following steps:

1. Define situations for message flow events so that an alert is raised each time a message flows down a path that is failing.
2. Deploy the situations that you define in the previous step to the system on which the monitored message flow is running.
3. Open the Monitor Node Events workspace to see data about the message being processed when the events that you define occur.

Collecting requested system trace data for a broker on a remote system

This scenario describes how you can use the WebSphere Message Broker Monitoring agent to collect system trace data requested by IBM Software Support for a broker that is located on a remote system.

After you have configured the WebSphere Message Broker Monitoring agent, start the agent so that the necessary statistics can be gathered, and then do the following steps:

1. Log on to the Tivoli Enterprise Portal.
2. In the Navigator view, right-click the Navigator item of a broker and click **Take Action** > **Select**. The Take Action window is displayed.
3. In the **Name** field, click the arrow to display the list.
4. Select **QI Change Trace Broker** from the **Name** list. The Edit Argument Values window is displayed.
5. In the Edit Argument Values window, specify the required argument values for the command, and then click **OK**.
6. In the Take Action window, select the broker managed system for the broker that needs to be traced from the **Destination Systems** list and click **OK**.

Stopping a message flow that has a full output queue

This scenario describes how you can use the WebSphere Message Broker Monitoring agent to quickly stop a message flow that is failing because it has a full output queue. This scenario is intended for users who do not always start the Configuration Manager and Message Brokers Toolkit so that they can conserve system resources.

After you have configured the WebSphere Message Broker Monitoring agent, start the agent so that the necessary statistics can be gathered, and then do the following steps:

1. Log on to the Tivoli Enterprise Portal.
2. In the Navigator view, right-click the Navigator item of a broker and click **Take Action** > **Select**. The Take Action window is displayed.
3. In the **Name** field, click the arrow to display the list.
4. Select **QI Stop Message Flow(s)** from the **Name** list. The Edit Argument Values window is displayed.
5. In the Edit Argument Values window, set the arguments to the appropriate execution group and message flow and click **OK**.
6. In the Take Action window, select the broker managed system for the broker with the failing message flow from the **Destination Systems** list and click **OK**.

Automatically starting a broker that is stopped

This scenario describes how you can use the WebSphere Message Broker Monitoring agent to automatically restart a broker whenever it is stopped.

After you have configured the WebSphere Message Broker Monitoring agent, start the agent so that necessary statistics can be gathered, and then perform the following steps:

1. Define a situation for the Broker Status workspace that checks whether the broker status is stopped.
2. Set the action for the situation to use the **`WMB Start Broker`** command to restart the broker.

   Tip: There is a predefined situation for the Components workspace that you can use for the same purpose, but that situation also starts the configuration manager and user name server, if either one stops.

Starting and stopping message flows at periodic intervals

This scenario describes how you can use the WebSphere Message Broker Monitoring agent to automatically start and stop message flows at certain times of the day.

You can use situations and take action commands to start and stop individual message flows or to start and stop all the message flows within an execution group.

For individual message flows

To start or stop individual message flows at intervals, after you have configured the WebSphere Message Broker Monitoring agent, start the agent so that you can gather the necessary statistics, and then perform the following procedure:

1. Define a situation for the **Status** attribute in the Message Flow Status workspace. Use the **Local Time** attribute group to specify when a message flow must be started or stopped.
2. Set the action for the situation to use the **WMB Start Message Flow(s)** to start the message flow or use the **WMB Stop Message Flow(s)** command to stop the message flow; specify both the execution group name and the message flow name.

For all message flows within an execution group

To start or stop all message flows within an execution group at intervals, after you have configured the WebSphere Message Broker Monitoring agent, start the agent so that you can gather the necessary statistics, and then do the following steps:

1. Define a situation for the **Started Message Flows** attribute in the Execution Group Status workspace. Use the **Local Time** attribute group to specify when all message flows in the execution group must be started or stopped.
2. Set the action for the situation to use the **WMB Start Message Flow(s)** to start all message flows or use the **WMB Stop Message Flow(s)** command to stop all message flows. Specifying only the execution group name argument and do not specify a message flow name argument.

Chapter 5. Monitoring with workspaces

The WebSphere Message Broker Monitoring agent is installed with default views that are displayed in workspaces. Where applicable, links are provided in the workspace to link from a parent view to a more detailed view of a selected row, or to a related workspace (for example, a workspace containing historical information).

You can customize the format and appearance of the views in a workspace.

Tip: Customize a copy of the workspace. Use the **Save as** command first to copy the workspace and give it a new name, and then make changes to the copy. You can set the workspace that you create in this manner to be the default workspace for a given Navigator item. If you do not replace the product-provided workspaces, you can see new features that might become available in later versions of WebSphere Message Broker Monitoring agent without having to revert to the original workspace. For information on customizing workspaces and views, see the Tivoli Enterprise Portal online help.

The following sections provide information about workspace groups.

Before you begin

Before you begin, there are several things to understand about how the workspace works:

- Some prior conditions must be met, so that the WebSphere Message Broker Monitoring agent can collect data for certain workspaces. For information about the related workspaces, see "Data availability."
- Two types of statistical data are provided, broker accounting statistics data and CandleMonitor node statistics data. Both types of data can provide information about message flows in a broker. To decide which type best suits your environment, see "Comparison between broker accounting statistics data and CandleMonitor node statistics data" on page 62.

Data availability

The availability of some data and statistics depends on prior conditions. If these conditions are not met, no information is available in the workspace.

- For archive accounting statistics to be available in the following workspaces, configure the brokers to collect it and set the destination of the statistics output (`-o`) to `xml`. You can configure the broker to collect archive accounting statistics by issuing the **WMB Change Flow Stats** command.
  - Archive Accounting Message Flow Statistics workspace
  - Archive Accounting Thread Statistics workspace
  - Archive Accounting Node Statistics workspace
  - Archive Accounting Terminal Statistics workspace
- For snapshot accounting statistics to be available in the following workspaces, configure the brokers to collect it and set the destination of the statistics output (`-o`) to `xml`. You can configure the broker to collect snapshot accounting statistics by issuing the **WMB Change Flow Stats** command.
  - Snapshot Accounting Message Flow Statistics workspace

  - Snapshot Accounting Thread Statistics workspace
  - Snapshot Accounting Node Statistics workspace
  - Snapshot Accounting Terminal Statistics workspace
- For user statistics to be available, issue the **WMB Create User Statistics** command for the Tivoli Enterprise Portal user ID.
- For message flow and subflow statistics and events to be available in the following workspaces, CandleMonitor nodes must be included in the message flow (see Chapter 3, "Monitoring with the CandleMonitor node," on page 37):
  - Monitor Node Broker Statistics workspace
  - Monitor Node Base Statistics workspace
  - Monitor Node Execution Group Statistics workspace
  - Monitor Node Events workspace
  - Monitor Node Message Flow Statistics workspace
  - Monitor Node Sub-Flow Statistics workspace
- For accounting origin information to be available in accounting workspaces, participating message flows must be configured to provide the appropriate origin identifier. If you do not specify a value, the Accounting Origin column in the Accounting workspaces contains the anonymous value.

Comparison between broker accounting statistics data and CandleMonitor node statistics data

The WebSphere Message Broker Monitoring agent provides two types of statistical data to monitor message flows in a broker:

- Broker accounting and statistics data
- CandleMonitor node statistics data

You can use the information in this topic to make a choice that best suits your environment between the two data types.

Broker accounting and statistics data

When statistics are collected for the broker and XML format is specified as the output destination, the WebSphere Message Broker Monitoring agent reports on accounting and statistics data that is produced by the broker. The agent automatically subscribes to the broker to receive this type of data.

For the accounting and statistics data, you only need to use the **mqsichangeflowstats** command to enable statistics collection at the broker, and after that you can see the data in the Tivoli Enterprise Portal workspaces. With the **mqsichangeflowstats** command, you can indicate whether you want to do the following things:

- Collect statistics for a specific message flow
- Collect statistics for all message flows
- Collect thread related statistics
- Collect node related statistics (including terminals for the nodes)

All these data are supported by the WebSphere Message Broker Monitoring agent.

Tip: You can issue the **mqsichangeflowstats** command to the broker-managed systems at any time by using the predefined **WMB Change Flow Stats** command.

There are two types of data collection: snapshot and archive. Although they are used for different purposes, data of both types support the same levels of details about message flows, threads, nodes, and terminals.

Archive data is intended for use in more long-term accounting and statistics data. This data is the type that you want to collect continuously for general monitoring of message flows. Archive data is collected at an interval that you can configure, with the minimum interval being 1 minute. The default interval for this data is 60 minutes. To change this interval, use the **mqsichangebroker** command with the **-v** parameter or the **WMB Change Broker** command from Tivoli Enterprise Portal. The broker must be stopped when the command is issued.

Snapshot data is the type that you want to collect for a short period of time when you are troubleshooting a problem in one or more message flows. The snapshot data is collected every 20 seconds, and you cannot change this interval. Performance can be affected by collecting snapshot data, and the agent can be impacted by the amount of data published. So you must be careful to enable snapshot data collection only for particular message flows being debugged when you need it and to disable it when you finish debugging. You can enable or disable the snapshot data collection with the **WMB Change Flow Stats** command from Tivoli Enterprise Portal.

There are four levels of data: message flow, thread, node, and terminal. The attributes for the accounting and statistics data vary for different levels. The data of message flow level include elapsed and CPU timings, input and output message counts and sizes, and various special or error counts. The data of thread level report on threads that process message flows and includes CPU and Elapsed timings, and message size and rate. The node level data report on elapsed and CPU timings for each node in a message flow. The terminal level data present counts of invocations of the various terminals for nodes in the message flow.

A set of workspaces is available to display the following accounting and statistics data. And situations can be targeted at the most current interval data to automatically detect problems.

- Data collected during the most current interval
- Data collected during several most recent intervals for trending
- Historical data (when historical data collection is enabled)

The historical data collection function can track the archive statistics for accounting purposes without purchasing or developing a second application.

If you intend to use the accounting origin support to organize your data, you must configure participating message flows to provide the appropriate origin identifier. As described in the WebSphere Message Broker documentation, this involves coding an ESQL statement in either a compute, database, or filter node that sets the value that you want. In addition, you must specify the **-b basic** parameter on the **mqsichangeflowstats** command that you use to start data collection.

CandleMonitor node statistics data

CandleMonitor node statistics data are produced by the CandleMonitor node that are placed in message flows. The CandleMonitor node can be placed multiple times within message flows, depending on the amount of monitoring that you want. To use the CandleMonitor node for monitoring, you must do the following things:

1. Make the CandleMonitor node available in the Message Broker Toolkit.
2. Modify the message flow.
3. Redeploy the message flow to the broker.

Tip: For more information about the usage of CandleMonitor node, see Chapter 3, "Monitoring with the CandleMonitor node," on page 37.

For typical monitoring, it is sufficient to place one CandleMonitor node at the beginning of the flow (after input node) and one CandleMonitor node at the end of the flow (before the output node). Only the message flows with at least one CandleMonitor node are represented in the reported data.

You can also configure the CandleMonitor node to produce user-defined message flow events for situation detection and Tivoli Enterprise Portal display of problems. For example, a message flows down a failure path, and the CandleMonitor node can automatically report exceptions that are propagated from any node as message flow events. Configuration parameters for activating certain nodes can also be used to disable different types of nodes from reporting data to scale back data collection. So you can place many nodes during message flow development (for example, to monitor sub-flows in a more granular pattern) and leave them in place later when moving to production, although they are inactive.

The CandleMonitor node implementation is provided at the broker by the `kqipnode.lil` file, which must be made available to the broker before deploying a message flow with the node. When the broker initializes the `kqipnode.lil` file, it will set up a shared memory area for recording the statistics. When a message flows through the CandleMonitor node, a little processing of data is required here. Instead, the WebSphere Message Broker Monitoring agent reads the shared memory and does all the calculations and summarization of data. So the CandleMonitor node has little impact on the message flow. By default, the agent does this work at a 1-minute interval. If there is any message flow event that is posted by a CandleMonitor node, the WebSphere Message Broker Monitoring agent also picks up the event from shared memory. And this occurs every 15 seconds by default. You can use the agent configuration parameters to modify these intervals.

Basically, the same set of statistics giving elapsed timings, input and output message counts, and queue timings are available at the following summarization levels:

- CandleMonitor node
- Sub-flow
- Message flow
- Execution group
- Broker

The base data that are collected by the agent are reported in Monitor Node Base Statistics workspace. Data in Sub-Flow Statistics workspace are summarized for each subflow that is delineated by the CandleMonitor nodes of subflow type. Data in Monitor Node Message Flow Statistics workspace are summarized for each message flow with at least one CandleMonitor node of input type. The statistics are summarized also at the execution group level in Monitor Node Execution Group Statistics workspace and at the broker level in Monitor Node Broker Statistics workspace.

You can also use the **WMB Create User Statistics** command to create what are called collectively as user statistics. These are the same statistics and levels, except that you collect the data when you want by issuing a Take Action command named **WMB Sample User Statistics**. Issue one **WMB Sample User Statistics** command when you want the interval to begin; issue another **WMB Sample User Statistics** command when you want the interval to end. This type of collection is useful, for example, to gather statistics for a certain set of messages flowing.

Cumulative statistics with **Overall** attribute names are maintained internally at the CandleMonitor node level. These statistics are simultaneously reset to 0 for all CandleMonitor nodes when a deploy operation to a broker involves any message flow containing a CandleMonitor node, when any CandleMonitor node detects an overflow condition for statistics, or when the Reset Statistics Take Action command has been issued.

A reset of statistics includes all statistics that are maintained for the broker to preserve the integrity of summarized statistics. The CandleMonitor node produces an Event Log message when a reset occurs because of a numeric overflow condition, and the monitoring agent logs a message when a reset is detected.

At the time of a reset of statistics, workspaces displays that the **Overall** statistics have started over from zero.

Historical workspaces displays data before the reset combined with data following the reset for the interval in which the reset occurred. This ensures that no historical data is lost. In subsequent intervals, the historical workspaces displays the **Overall** values as having started over from zero.

Table 2. Comparison between broker accounting statistics and CandleMonitor node statistics

		Differences	
Category	**Similarities**	**Broker accounting and statistics**	**CandleMonitor node statistics**
Data attributes	Both types have the following attributes: • Elapsed Timings • Message Counts • Message Sizes • Byte Rates	• Special error counts at the message flow level • CPU Timings • Invocation counts for nodes and terminals	• Queue Time • Message Flow Events

Table 2. Comparison between broker accounting statistics and CandleMonitor node statistics (continued)

Category	Similarities	Differences	
		Broker accounting and statistics	**CandleMonitor node statistics**
Data levels	Both types have the message flow level.	• Thread • Node • Terminal Data of all levels are available with archive and snapshot accounting.	• Broker • Execution group • Sub-flow • CandleMonitor node Data of all levels are available with regular and user-defined statistics. **Remember:** The levels of statistics are all only summarizations of the same base statistics collected, and they are only applicable with respect to where the CandleMonitor node has been deployed. The applications and libraries feature of the broker is not reflected in CandleMonitor node data, which might cause some confusion if the same message flow name occurs in multiple applications or libraries.
Detection of message flowing on a failure path		Accounting and statistics data at the terminal level can be used in a situation to detect that a failure terminal has an invocation count that is greater than zero, which means that a message flowed to a failure path. However no information about the message is available.	CandleMonitor node statistics can be positioned along a failure path in a message flow with the **event** attribute set to an event message. This message flow event can be displayed at the Tivoli Enterprise Portal and can be detected by situations. The message ID and correlation ID are among the message data available.

Table 2. Comparison between broker accounting statistics and CandleMonitor node statistics (continued)

Category	Similarities	Differences	
		Broker accounting and statistics	**CandleMonitor node statistics**
Collection interval	The interval can be changed for some or all of the data collection.	Archive interval has a minimum of 1 minutes and a default of 60 minutes. Snapshot interval cannot be changed, and the default is 20 seconds.	The default interval is 1 minutes for statistics (which can be set to less than 1 minute) and 15 seconds for message flow events. The interval must be set in the `kqi.xml` agent configuration file with the **`defaultStatisticsInterval`** and **`defaultFlowEventInterval`** parameters. For more information about these parameters, see "Agent parameter descriptions" on page 18.
Performance		Use Archive accounting for regular monitoring. Snapshot accounting should be used only for problem determination and not for regular monitoring.	CandleMonitor node has little impact on a message flow. The interval does not impact the broker performance.
Agent installation		Accounting and statistics data have no extra installation steps.	CandleMonitor node requires root authority for installation of the `kqipnode.lil` file at the broker on UNIX systems. On z/OS systems, additional steps are required to integrate the node into the broker environment. You must make the CandleMonitor node available in the Message Broker Toolkit separately.
Configuration	Configuration parameters are provided in the `kqi.xml` configuration file. The WebSphere Message Broker Monitoring agent must be restarted to make the parameter change take effect.	Accounting and statistics data are configured by the **`mqsichangeflowstats`** command, which is also available as a Take Action command from the Tivoli Enterprise Portal interface. The configuration can be done dynamically.	Primary configuration of the CandleMonitor node is within the Message Broker Toolkit to implement message flows with the node. The `kqipnode.cfg` configuration file is available at the broker. The broker must be restarted to make the configuration change take effect.

Table 2. Comparison between broker accounting statistics and CandleMonitor node statistics (continued)

Category	Similarities	Differences	
		Broker accounting and statistics	**CandleMonitor node statistics**
Reason most often chosen	Both types provide data that are required to determine the situation of a message flow.	Accounting and statistics data are chosen because they are easy to configure with dynamic configuration changes possible. Accounting and statistics data includes CPU timings at the message flow, thread and node levels, various message flow error counts, and node and terminal invocation counts. Also, they support the application and libraries feature of the broker. When using these statistics, you are recommended to use the companion WebSphere MQ monitoring agent to determine queue times.	CandleMonitor node statistics data are chosen because of the desire to use the message flow events feature. Other reasons for choosing in the past have diminished in recent broker releases.

Workspace summary

Use the references in this section to find information about the following workspaces:

- "Agent and application status workspaces" on page 70
- "Broker and message flow information workspaces" on page 70
- "Event workspaces" on page 70
- "Statistics workspaces" on page 70
- "Accounting workspaces"

Accounting workspaces

Accounting workspaces provide statistics at message flow, node, thread, and terminal level. The data in these workspaces is sampled at snapshot (short term) or archive (long term) intervals. Top-level snapshot and archive workspaces present current accounting data for that interval. Each workspace links to workspaces that display the data for the most recent collection intervals and for the historical collection intervals, if those intervals are configured.

- The Snapshot Accounting Message Flow Statistics and Archive Accounting Message Flow Statistics workspaces provide statistics at the message flow level.
- The Snapshot Accounting Node Statistics and Archive Accounting Node Statistics workspaces provide statistics at the processing node level.

- The Snapshot Accounting Terminal Statistics and Archive Accounting Terminal Statistics workspaces provide statistics at the terminal level.
- The Snapshot Accounting Thread Statistics and Archive Accounting Thread Statistics workspaces provide statistics at the thread level.

Exception: Historical data cannot be collected for the following workspaces:

- Snapshot Accounting Message Flow Statistics
- Snapshot Accounting Node Statistics
- Snapshot Accounting Terminal Statistics
- Snapshot Accounting Thread Statistics

Retained data in accounting workspaces

How many table rows are retained and displayed in accounting workspaces are controlled by three factors:

- Agent configuration

  In the agent configuration file, the **defaultRetainRecentSnapshotSamples** parameter determines the size of recent snapshot accounting tables. The **defaultRetainRecentArchiveSamples** parameter determines the size of recent archive accounting tables. Sometimes the number of rows displayed in recent snapshot accounting tables might be larger than the **defaultRetainRecentSnapshotSamples** value. It is normal if the extra rows are less than four, because a broker publishes snapshot accounting data every 20 seconds and the WebSphere Message Broker Monitoring agent checks and cleans the retained table every 60 seconds.

- The refresh frequency on Tivoli Enterprise Portal

  If you stay in one of the accounting workspaces and keep refreshing the workspace frequently, the retained data will not be cleaned up and the agent will return more and more data in the workspace. It has the same effect when you configure the workspace to automatically refresh data after a certain time period by using the menu option on Tivoli Enterprise Portal. Again, if the specified time period is very short, more and more data is returned in the workspace after each refresh.

  The **holdTimeForQuery** parameter also determines the maximum interval (in seconds) of refresh frequency. It specifies the length of time that the agent must retain accounting data. If you refresh the workspace more frequently than the interval specified by the **holdTimeForQuery** parameter, all the previously retained data will remain and will not be cleaned up.

  Do not enable automatic refresh of accounting workspaces, unless the refresh interval is longer than the **holdTimeForQuery** value. Otherwise, the monitoring agent might require a lot of memory to retain data.

- Historical data collection configuration

  The **defaultHistoricalAccountingType** parameter determines which type of accounting data to be collected historically. Do not set this parameter to All or Snapshot unless necessary. This parameter is useful only when the historical data collection has been enabled. If historical data collection is enabled for any of the accounting tables, the accounting data is kept in the agent memory till the data is collected by historical collection. Always set the same collection interval for all the accounting data.

Agent and application status workspaces

- The Agent Status workspace summarizes event and broker product component information at the agent level.
- The Components workspace presents a list of broker product components and their state at the agent level.
- The Broker Summary workspace summarizes broker status and definition data when multiple brokers are monitored by the same agent.
- The Broker Status workspace summarizes event and definition information at the broker level.

Broker and message flow information workspaces

- The Broker Status workspace contains information about a broker-managed system, such as the status of the broker, its process ID, information about the operating system where the broker runs, its job on z/OS systems, its associated queue manager.
- The Message Flow Node Topology workspace provides a graphical view of a message flow (including its nodes and connection between nodes) and a table view of detailed information about the message processing nodes.
- The Processing Node Attributes workspace lists all attribute values for the selected message processing node within a message flow.

Event workspaces

- The Product Events workspace presents events that are generated by the agent when a problem occurs that affects the agent ability to collect data.
- The Broker Status Events workspace lists events that are generated by a broker as they occur.
- The Monitor Node Events workspace lists events that are generated by a CandleMonitor node as they occur.

Statistics workspaces

- The Monitor Node Broker Statistics workspace provides summarized message flow statistics at the broker level.
- The Monitor Node Base Statistics workspace provides message flow statistics at the individual node level at which they are collected.
- The Monitor Node Execution Group Statistics workspace provides summarized message flow statistics at the execution group level.
- The Monitor Node Message Flow Statistics workspace provides summarized message flow statistics at the message flow level.
- The Monitor Node Sub-Flow Statistics workspace provides summarized message flow statistics at the subflow level.

Remember:

- After a message flow is restarted, its statistics are reset in the preceding workspaces.
- Message flow accounting and statistics are provided that partially overlap with the statistics provided by the CandleMonitor node and can be monitored in Tivoli Enterprise Portal without including the CandleMonitor node in message flows. However, the CandleMonitor node provides additional statistics, and the ability to monitor sub-flows and define message flow events. Message flows without a CandleMonitor node are not represented in statistics workspaces.

Resource Statistics workspaces (WebSphere Message Broker V7.0 or later only)

Resource Statistics workspaces provide the statistics of system resources that are collected by a broker. The statistical information includes performance and operating details of resources that are used by execution groups. You can use the resource statistics to ensure that your systems are using the available resources in the most efficient manner.

Prerequisite: For the workspaces to contain data, you must enable resource statistics collection at the broker in the first place. To do it, issue the **mqsichangeresourcestats** with `-c active` specified for the whole broker or optionally for an execution group within the broker.

Statistics are collected for the following types of resources:

- File

  The File Resource Statistics workspace provides the statistical information for the local file system of any file actions that are done by any file node.
- JDBC connection pools

  The JDBC Connection Pools Resource Statistics workspace provides statistical information about each JDBC Provider configurable service.
- Java virtual machine (JVM)

  The JVM Resource Statistics workspace provides the statistical information about the JVM resources that are used by execution groups.
- ODBC

  The ODBC Resource Statistics workspace provides the statistical information about each ODBC DSN that has been accessed since the execution group starts.
- Parsers

  The Parsers Resource Statistics workspace provides the statistical information about the parser resources within each execution group.
- SOAP

  The SOAP Input Resource Statistics workspace provides the statistical information about SOAP nodes on a per-operation basis.

Creating a workspace using a predefined workspace as a template

You can create your own workspaces from any predefined workspace to display information about a specific set of attributes. The new workspace is associated with the same Navigator item as the original workspace.

Your user ID must have Workspace Author Mode permission to save the new workspace.

Use Tivoli Enterprise Portal to complete this task.

To create a new workspace using a predefined one as a template, perform the following steps:

1. Open the predefined workspace that you want to use as a template.
2. To create a copy of the predefined workspace, click **File** > **Save Workspace As**.
3. Enter a workspace name and, optionally, a description. The workspace name is displayed on the title bar.
4. Optional: Select one or more of the following workspace options:

- **Assign as default for this Navigator Item**: Select this option if you want this workspace to be displayed when this Navigator Item is clicked.
- **Do not allow modifications**: Select this option to prevent this workspace from being modified in the future.
- **Only selectable as the target of a workspace link**: Select this option if you do not want this workspace to be displayed unless it is linked to from another workspace.

5. Click **OK**. A copy of the predefined workspace is created with the name that you entered.
6. Open the new workspace and click **Edit** > **Properties** to customize it to meet your requirements.

Creating the user statistics workspace

In a development environment, sampling statistics at a set interval might not provide the flexibility that you need. You can use the user statistics feature, implemented by three Take Action commands, to collect statistics that are associated with a particular Tivoli Enterprise Portal logon ID. These user statistics are displayed in the following statistics workspaces that they are derived from and can be linked to from related statistics workspaces, as listed in one of the following tables, depending on the version of WebSphere Message Broker.

Table 3. Workspaces with user statistics links

Workspace	User statistics links
Monitor Node Broker Statistics	• User Monitor Node Broker Statistics • User Monitor Node Execution Group Statistics
Monitor Node Execution Group Statistics	• User Monitor Node Execution Group Statistics • User Monitor Node Message Flow Statistics
Monitor Node Message Flow Statistics	• User Monitor Node Message Flow Statistics • User Monitor Node Sub-Flow Statistics • User Monitor Node Base Statistics
Monitor Node Sub-Flow Statistics	• User Monitor Node Sub-Flow Statistics • User Monitor Node Base Statistics
Monitor Node Base Statistics	User Monitor Node Base Statistics

To create the user statistics workspace, issue a **WMB Create User Statistics** command for the user's logon ID before the user statistics workspaces can be accessed.

Remember:

- If you try to follow a link to a workspace before the command has been issued for your logon ID, no data is available in the workspace that you link to.
- The user statistics are not collected historically.

Scenarios of monitoring with workspaces

This section provides examples of monitoring situations that are common to many broker product environments, and explains how the WebSphere Message Broker Monitoring agent can be used in these circumstances.

Each scenario presents a case in which you need to monitor some aspect or component of your broker product environment, followed by a description of how to set up WebSphere Message Broker Monitoring agent to gather important information.

Monitoring application message flow performance

This scenario describes how you can use the WebSphere Message Broker Monitoring agent in a performance monitoring situation in which an application message flow is taking excessively long time to process messages.

Do the following steps to enable the WebSphere Message Broker Monitoring agent to gather the appropriate statistics:

1. Place one CandleMonitor node immediately after the MQInput node so that the entry of the messages into the message flow is recorded.
2. Set the **type** attribute of this CandleMonitor node to `input`.
3. Place another CandleMonitor node immediately in front of the MQOutput node so that the exit of the messages from the message flow is recorded.
4. Set the **type** attribute of this CandleMonitor node to `output`.
5. Start the WebSphere Message Broker Monitoring agent.

After you have positioned the CandleMonitor nodes and the WebSphere Message Broker Monitoring agent is running, perform the following procedure:

1. Open the Monitor Node Message Flow Statistics and examine the values of the **Current Average Queue Time** and **Current Average Flow Time** attributes.
2. If you think that the times reported by the **Current Average Queue Time** attribute are excessively long, increase the Additional Instances value for the monitored message flow. The Additional Instances parameter specifies the number of additional instances of the message flow that the execution group must run so that more messages can be processed concurrently.
3. If you think that the times reported by the **Current Average Flow Time** attribute are excessively long, then you need to debug the monitored message flow. To perform this task, do the following steps:
   a. Insert CandleMonitor nodes before each processing node in the message flow. This can help you determine which node is causing the problem.
   b. Set the **type** attribute of the new CandleMonitor nodes to `other`.
   c. Start the WebSphere Message Broker Monitoring agent again.
   d. Open the Monitor Node Sub-Flow Statistics workspace and examine the flow times of any subflows in the message flow.
   e. Compare statistics for the subflows to determine which section of the message flow is causing problems.

Determining application delivery failure of messages

This scenario describes how you can use the WebSphere Message Broker Monitoring agent to determine the cause of the problem when messages that are destined for an application are not being received by that application.

Perform the following steps to enable the WebSphere Message Broker Monitoring agent to gather the appropriate statistics:

1. Place one CandleMonitor node immediately after the MQInput node so that the entry of the messages into the message flow is recorded.
2. Set the **type** attribute of this CandleMonitor node to `input`.

3. Place another CandleMonitor node immediately in front of the MQOutput node so that the exit of the messages from the message flow is recorded.
4. Set the **type** attribute of this CandleMonitor node to `output`.
5. Start the WebSphere Message Broker Monitoring agent.

After you have placed and set the CandleMonitor nodes, and the WebSphere Message Broker Monitoring agent is running, perform the following procedure:

1. Open the Message Flow Status workspace and examine the **Status** attribute. The Status column for the monitored message flow must have the value **Started**. If it does not, start the message flow manually.
2. Open the Monitor Node Message Flow Statistics workspace and compare the **Current Message Input Count** and **Current Message Output Count** attributes. A discrepancy between these values indicates that messages are being lost in the monitored message flow.
3. Open the Monitor Node Events workspace and determine whether any exceptions have been triggered by a node in the monitored message flow.

Debugging a message flow

This scenario describes how you can use the WebSphere Message Broker Monitoring agent to debug the operation of a message flow.

To determine which node in the message flow is causing a problem, do the following steps so that the WebSphere Message Broker Monitoring agent can gather appropriate statistics for you to analyze:

1. Place CandleMonitor nodes on each side of nodes or subflows that you think are functioning incorrectly in the message flow.
2. Set the **type** attribute of a CandleMonitor node placed in front of a node or subflow to `subFlowInput`.
3. Set the **type** attribute of a CandleMonitor node placed following a node or subflow to `subFlowOutput`.
4. Set the **subFlowName** attribute of each pair of nodes to the same value, which must be unique within the message flow.
5. Start the WebSphere Message Broker Monitoring agent.

After you have positioned the CandleMonitor nodes and the WebSphere Message Broker Monitoring agent is running, do the following steps:

1. Open the Monitor Node Sub-Flow Statistics workspace, and examine the flow times of all subflows in the message flow.
2. Compare statistics for different subflows to determine which section of the message flow is problematic.

Verifying the broker configuration

This scenario describes how you can use the WebSphere Message Broker Monitoring agent to verify that the broker configuration matches your configuration as depicted in the Message Brokers Toolkit.

After you have configured the WebSphere Message Broker Monitoring agent, start the agent so that the necessary statistics can be gathered, and then perform the following steps:

1. Examine the various informational workspaces, such as Execution Group Status, Message Flow Status, Message Processing Nodes. These workspaces contain data from the broker perspective as opposed to the Message Brokers Toolkit perspective.
2. Compare the data in the workspaces with what you expect to determine whether the correct configuration is deployed to the broker.

Planning broker capacity

This scenario describes how you can use the WebSphere Message Broker Monitoring agent to plan the capacity for your brokers.

To determine the most recent message that was sent between two applications, do the following steps so that the WebSphere Message Broker Monitoring agent can gather the necessary statistics:

1. Place one CandleMonitor node immediately following the MQInput node so that the entry of the message into the message flow is recorded.
2. Set the **type** attribute of the CandleMonitor node to `input`.
3. Place one CandleMonitor node immediately in front of the MQOutput node so that the exit of the message from the message flow is recorded.
4. Set the **type** attribute of the CandleMonitor node to `output`.
5. Start the WebSphere Message Broker Monitoring agent again.

After you have positioned the CandleMonitor nodes and the WebSphere Message Broker Monitoring agent is running, do the following steps:

1. Open statistics workspaces, such as Monitor Node Message Flow Statistics for individual message flows, Monitor Node Execution Group Statistics for summarization over an entire execution group, or Monitor Node Broker Statistics for statistics summarizing the operation of an entire broker.
2. Examine the statistics workspaces and their historical workspaces (which you can link to from the statistics workspaces).
3. If you find that for a particular message flow the **Current Message Input Rate** value is higher than expected, or that the **Current Average Queue Time** value is unacceptably long, add more instances to the flow in the Message Brokers Toolkit.
4. If you find that for a particular execution group the **Current Message Input Rate** value is higher than expected, or that the **Current Average Queue Time** value is unacceptably long, add a new execution group and assign instances as appropriate.
5. If you find that for a particular broker the **Current Message Input Rate** value is higher than expected, or that the **Current Average Queue Time** value is unacceptably long, consider adding a new broker with a configuration that can offload some processing from the current broker.

Ensuring reasonable message flow response times

This scenario describes how you can use the WebSphere Message Broker Monitoring agent to ensure that the response times of your message flows are reasonable.

To monitor message flow response times, perform the following steps so that the WebSphere Message Broker Monitoring agent can gather the necessary statistics:

1. Place one CandleMonitor node immediately following the MQInput node so that the entry of the message into the message flow is recorded.

2. Set the **type** attribute of the CandleMonitor node to `input`.
3. Place one CandleMonitor node immediately in front of the MQOutput node so that the exit of the message from the message flow is recorded.
4. Set the **type** attribute of the CandleMonitor node to `output`.
5. Start the WebSphere Message Broker Monitoring agent again.

After you have positioned the CandleMonitor nodes, and the WebSphere Message Broker Monitoring agent is running, do the following steps:

1. Open the Monitor Node Message Flow Statistics workspace for individual message flows.
2. Examine the workspaces and their historical equivalents. You can link to the historical workspaces from the current statistics workspaces.
3. If the **Current Average Flow Time** for a particular message flow is higher than expected, debug the message flow as described in "Debugging a message flow" on page 74. Compare the Average Flow Times for each section of the message flow to identify where problems are occurring.

Displaying a graphical view of your environment

You can use the topology workspaces to get a graphical view of your broker environment. For different versions of brokers, the ITCAM agent for WebSphere Message Broker provides different sets of workspaces to present the topology overview with various tiers.

- Broker Status (broker topology, execution group topology, and message flow topology)
- Message Flow Node Topology (message flow topology)

Viewing the topology of your broker environment

The Broker Status workspace provides a graphical overview of your broker environment, including execution groups and messages flows.

Figure 10 on page 77 illustrates an example of the broker environment topology displayed in the Broker Status workspace. In the broker topology view of the Broker Status workspace, you can click a message flow node to link to the Message Flow Node Topology workspace for more information about the message flow.

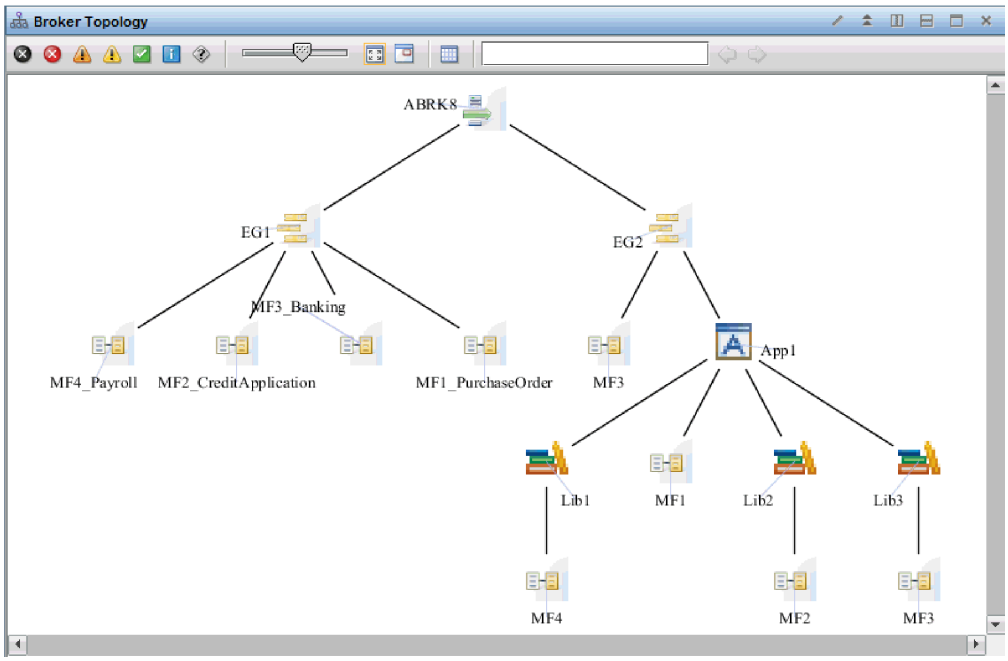

Figure 10. The broker topology view

The broker topology view has three to five tiers. Each tier contains icons representing different objects in the broker environment, represented by standard WebSphere Message Broker icons. The different tiers are as follows:

- **Tier 1**: The message broker. This tier contains a single icon representing the monitored message broker.
- **Tier 2**: Execution groups. This tier presents all the execution groups that exist within the broker.
- **Tier 3**: Applications/Libraries/Message flows. This tier presents all the applications/libraries/message flows that are contained within each execution group.
- **Tier 4**: Libraries/Message flows. This tier presents all the libraries or message flows that are contained within each application or library.
- **Tier 5**: Message flows. This tier presents all the message flows that are contained within each library.

If either the broker or a message flow is stopped, a red X is displayed in the upper-left corner of its icon. A red X is displayed in the upper-left corner of the icon representing an execution group if all of its user-defined message flows are stopped. If some but not all of the user-defined message flows in an execution group are started, a yellow exclamation mark (!) is displayed in the upper left corner of the icon representing the execution group. Figure 11 on page 78 presents the message broker icons for a started and a stopped message broker.

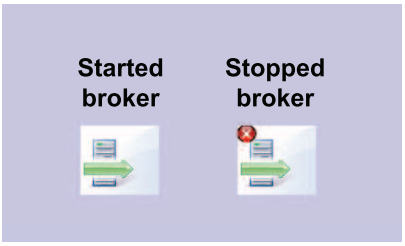

Figure 11. Started and stopped broker icons in the broker topology view

Exception: No default system message flows is displayed in the broker topology view.

Viewing the topology of a message flow

You can get a graphical overview of a single message flow, including its nodes and the links between them, in the following ways:

- In the Message Flow Status table of the Broker Status workspace, right-click a row and click **Link To** > **Message Flow in Broker Status** or **Link To** > **Message Flow Node Topology**.
- In the Broker Topology view of the Broker Status workspace, right-click a message flow node and click **Link To** > **Message Flow Node Topology**.

Figure 12 illustrates an example of the message flow topology view displayed in the Message Flow Node Topology workspace.

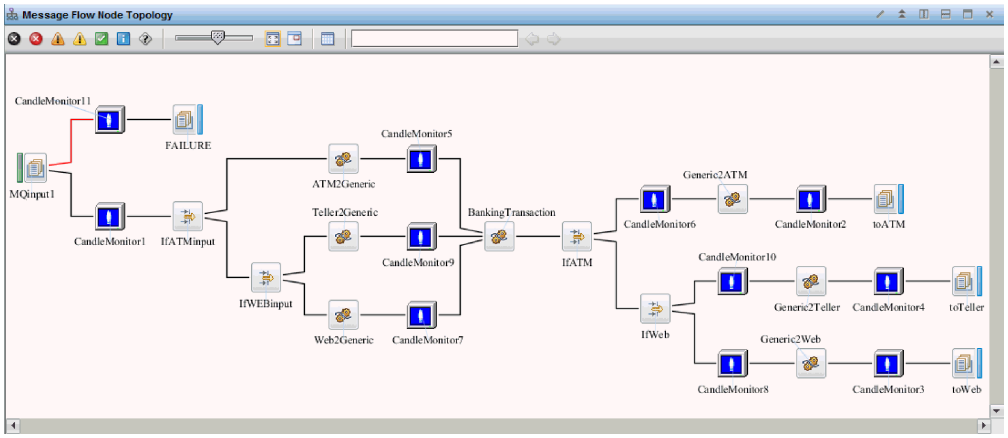

Figure 12. The message flow topology view

The message flow topology view presents each node in the message flow and the links between them. No matter how many terminals on two nodes are connected, there is only one link between the two nodes in the message flow topology view. The color of each link indicates its function. The icons and links used to represent the topology are the same in the WebSphere Message Brokers Toolkit. For more information, see WebSphere Message Brokers Toolkit documentation

Remember: Beginning from WebSphere Message Brokers Toolkit V8.0, subflows can be defined in one of two resource types, either a `.subflow` file or a `.msgflow` file. The message flow topology is not applicable to the subflows that are defined in `.subflow` files.

The only icons that are not the same as WebSphere Message Brokers Toolkit icons are the icons that represent the CandleMonitor node and user-defined nodes. These are shown in Table 4.

Table 4. A selection of message flow topology nodes

Icon	Node	Notes
	CandleMonitor node	
	User-defined node	All user-defined nodes in the message flow are denoted by this icon, regardless of function.

Because the information included in the message flow topology view is collected from the WebSphere Message Broker Monitoring agent, it reflects the objects that are deployed in the Message Broker environment. Objects that exist as part of a message flow design in the WebSphere Message Brokers Toolkit, but are not deployed to the environment, cannot be displayed. For this reason, subflows are displayed in the topology view as part of the main message flow, and are not displayed separately as they are in the Message Brokers Toolkit.

Viewing the topology of an execution group

You can get a graphical overview of an execution group, including its message flows by right-clicking a row in the Execution Group Status table of the Broker Status workspace and clicking **Link To** > **Execution Group in Broker Status**.

Figure 13 illustrates an example of the execution group topology view displayed in the Broker Status workspace.

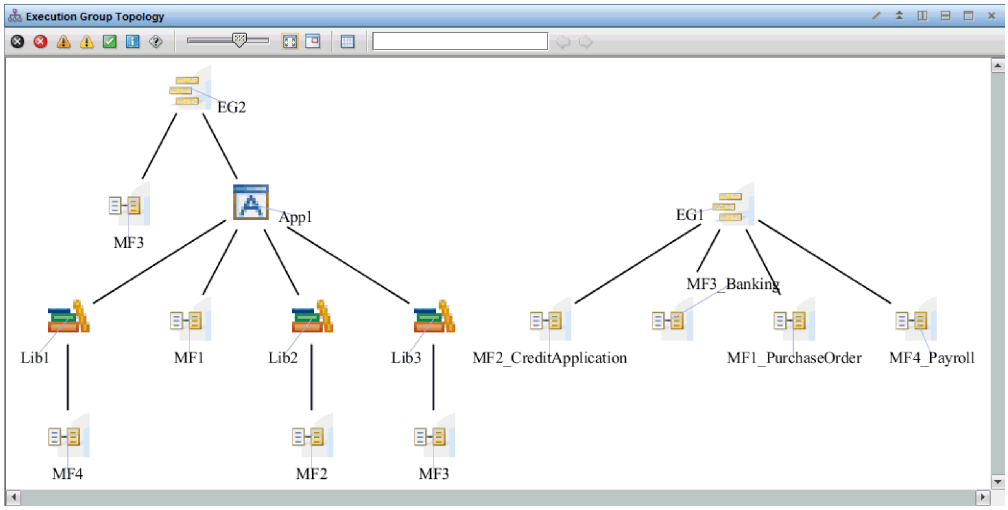

Figure 13. The execution group topology view

If a message flow is stopped, a red X is displayed in the upper left corner of its icon. A red X is displayed in the upper left corner of the icon representing the execution group if all of its user-defined message flows are stopped. If some but

not all of the user-defined message flows in an execution group are started, a yellow exclamation mark (!) is displayed in the upper left corner of the icon representing the execution group.

Exception: No default system message flows is displayed in the Execution Group Topology view.

Chapter 6. Collecting historical data

Tivoli Enterprise Portal provides options for configuring the collection and storage of historical data from the WebSphere Message Broker Monitoring agent. You can use these options to specify settings that are related to historical data collection, including the attributes for which historical data is collected, the collection interval, warehousing interval, the length of time that historical data must be stored and what reports are generated from the data.

Important: To view certain historical workspaces within Tivoli Enterprise Portal, you must configure historical data collection for the attribute groups that contain attributes that are displayed in those workspaces.

Initial settings for historical collection

The following historical tables are disabled by default after installation. To enable them, you must first enable historical data collection for the indicated attribute groups. Each attribute group corresponds to a product workspace.

- If a Broker Status Events row occurs, an entry is added to the historical log.
- If a Monitor Node Events row occurs, an entry is added to the historical log.
- If a Product Events row occurs, an entry is added to the historical log. This situation applies to agent-managed systems.
- A Broker Status log entry is written every 15 minutes (one row per broker).
- A Components log entry is written every 15 minutes (one row per broker product component that was created on the system). This situation applies to agent-managed systems.
- An Execution Group Status log entry is written every 15 minutes (one row per execution group).
- A Message Flow Status log entry is written every 15 minutes (one row per message flow).
- A File Resource Statistics log entry is written every 20 seconds (one row per execution group).
- A JDBC Connection Pools Resource Statistics log entry is written every 20 seconds (one row per execution group).
- A JVM Resource Statistics log entry is written every 20 seconds (four rows per execution group).
- A ODBC Resource Statistics log entry is written every 20 seconds (one row per execution group).
- A Parsers Resource Statistics log entry is written every 20 seconds (three rows or more per execution group).
- A SOAPInput Resource Statistics log entry is written every 20 seconds (one row per execution group).

Remember: The following historical tables are pure event tables. Historical data collection of the pure event tables is not affected by the collection interval that you set in the Historical Collection Configuration window on Tivoli Enterprise Portal.

- Broker Status Events
- Monitor Node Events
- Product Events

- File Resource Statistics
- JDBC Connection Pools Resource Statistics
- JVM Resource Statistics
- ODBC Resource Statistics
- Parsers Resource Statistics
- SOAPInput Resource Statistics

The following tables listed must always be configured with the same historical data collection interval for accurate results. The configured historical data collection interval must also be a multiple of the value of the **defaultStatisticInterval** (or **statisticInterval**) agent parameter; otherwise, historical data collection might produce unpredictable results. For example, if you configure historical data collection for these statistics attribute groups to run every 5 minutes (300 seconds), the **defaultStatisticInterval** (or **statisticInterval**) parameter value must be a submultiple of 300, such as 60 (1 minute), 75 (1.25 minutes), 100 (1.67 minutes), 150 (2.5 minutes), or 300 (5 minutes).

- A Monitor Node Broker Statistics log entry is written every 15 minutes (one row per broker).
- A Monitor Node Execution Group Statistics log entry is written every 15 minutes (one row per execution group).
- A Monitor Node Message Flow Statistics log entry is written every 15 minutes (one row per message flow per execution group).
- A Monitor Node Sub-Flow Statistics log entry is written every 15 minutes (one row per monitored subflow per message flow containing the subflow).
- A Monitor Node Base Statistics log entry is written every 15 minutes (one row per CandleMonitor Node).

The following tables gather data only if data is produced. By default, the agent collects historical data only for use as archive data, not snapshot. See "defaultHistoricalAccountingType" on page 20 and "historicalAccountingType" on page 28 for more information.

- An Archive Accounting Message Flow Statistics log entry is written for each message flow archive statistics report that is produced by the broker.
- An Archive Accounting Thread Statistics log entry is written for each thread archive statistics report that is produced by the broker.
- An Archive Accounting Node Statistics log entry is written for each node archive statistics report that is produced by the broker.
- An Archive Accounting Terminal Statistics log entry is written for each terminal archive statistics report that is produced by the broker.

Remember:

- Historical data can be collected only for archive accounting data, not snapshot accounting data, although the snapshot data can be displayed correctly in the historical workspaces. Do not collect historical snapshot data, otherwise, the data amount might be enormous.
- The collection interval of archive accounting data is determined by the broker with the **mqsichangebroker -v** command. If the collection interval of the archive accounting data is longer than that of the historical data, the historical versions of these archive accounting workspaces might be empty during one historical collection interval. This is because during this historical collection interval, no data is produced by the broker.

Starting historical data collection

For historical data to be available for an attribute group on one or more managed systems, you must start historical data collection for the attribute group.

Your user ID must have Configure History permission to open the History Collection Configuration window. If you do not have the permission, you will not see the menu item or tool for historical configuration.

Use the History Collection Configuration window to complete this task.

1. To open the History Collection Configuration window, click **Edit** > **History Configuration** from the Tivoli Enterprise Portal.
2. Click **WebSphere Message Broker** in the left side of the window.
3. Click **Create new collection setting**. The **Create New Collection Settings** window is opened.
4. In the **Name** field, enter a name of up to 256 characters.
5. Optional: In the **Description** field, enter a description of up to 64 characters for the collection.
6. Select an attribute group from the **Attribute Group** list. Only attribute groups that are appropriate for historical collection and reporting are displayed in the list.
7. Click **OK**. The configuration tabs for the collection are displayed.
8. Complete the fields in the **Basic** tab:
   - **Collection Interval** is the frequency of data transmission to the short-term history file on the computer where the data is saved (Tivoli Enterprise Monitoring Agent or Tivoli Enterprise Monitoring Server). The options are every one, five, 15, or 30 minutes, every hour, or once per day. The default interval is 15 minutes. The shorter the interval is, the faster and larger the history file grows. A short interval should be used only for an attribute group that is critical in your work.
   - **Collection Location** is where the short-term history data file resides: at the TEMA (Tivoli Enterprise Monitoring Agent) or the TEMS (Tivoli Enterprise Monitoring Server). The default location is TEMA, which minimizes the performance impact on the Tivoli Enterprise Monitoring Server from historical data management.
   - **Warehouse Interval** determines whether the collected data is warehoused and how often. The options are 15 minutes, 30 minutes, 1 hour, 12 hours, 1 day or Off.
9. In the **Distribution** tab page, select managed systems for which you want to start historical data collection from the **Available Systems** list and click the left arrow to move them to the **Start collection on** list.

   Remember: In the available managed system group list, `MQSI_BROKER_V7` is the group of all managed systems.
10. Click **OK** to start data collection on the managed systems.

Stopping historical data collection

Stop data collection on one or more managed systems for a historical collection definition when you are no longer interested in gathering data on the managed system or would like to temporarily disable data collection.

Your user ID must have Configure History permission to open the History Collection Configuration window. If you do not have this permission, you will not see the tool for historical configuration.

Use the History Collection Configuration window to complete this task.

1. To open the History Collection Configuration window, click **Edit** > **History Configuration** in the Tivoli Enterprise Portal.
2. Click the plus sign (+) to expand the **WebSphere Message Broker** branch on the left side of the window.
3. Click the collection that the managed systems for which you want to stop historical data collection are assigned to.
4. Click the **Distribution** tab.
5. In the **Start collection on** list, select the managed systems that you want to stop historical data collection and click the right arrow to move it to the **Available Systems** or **Available Managed System Group** list.

   Remember: `MQSI_BROKER_V7` is the group of all managed systems.
6. To save your changes, click **Apply** to keep the window open or **OK** to close it.

Historical data collection no longer occurs on the managed systems that you move out of the **Start collection on** list for the historical collection definition.

Viewing historical data for a selected time frame

In historical workspaces, you can choose to display only historical data that is collected over a particular period of time in which you are interested.

Historical data collection must be configured and distributed to the managed systems that you are querying data from.

Use Tivoli Enterprise Portal to complete the task.

1. Navigate to the historical workspace for which you want to view data from a particular period of time.
2. Click the **Specify Time Span for Query** button, located in the top left corner of each view in the historical workspace. The Select the Time Span window is displayed.
3. Select what data you want to be displayed in the table. There are the following available options:
   - **Real time**

     If you select this option, only the data collected during the most recent sampling period is displayed in the table.
   - **Last**

     If you select this option, you can choose to display all historical data going back to a certain date and time. For example, all data collected over the past 24 hours.
   - **Custom**

     If you select this option you can specify the exact period for which you want historical data to be displayed.
4. Do one of the following steps, depending on which option you have selected in the previous step.

- If you selected the **Last** option, enter the time period for which you want data to be displayed in the field provided, and select the units in which it is specified (for example, hours or days). You can also specify the following parameters:
  - **Use Detailed data**

    If you select this option, the data from the detailed data tables is displayed in the table without summarization. You can also select the column that you want to be used in determining whether data falls within the selected period from the Time column list of columns containing timestamps.
  - **Use summarized data**

    If you select this option, the data from the summarized data tables is displayed in the table. This data is aggregated by the time frame configured in the Historical Collection Configuration window. If you configured shift times when you installed IBM Tivoli Monitoring, you can also select for which shifts and days data is displayed. See your IBM Tivoli Monitoring documentation for further information.
- If you selected the **Custom** option, you can specify the following parameters:
  - **Use Detailed data**

    If you select this option, the data from the detailed data tables is displayed in the table without summarization. You can also select the column that you want to be used in determining whether data falls within the selected period from the Time column list of columns containing timestamps.
  - **Use summarized data**

    If you select this option, the data from the summarized data tables is displayed in the table. From the Interval list, select the time period over which you want the data to be aggregated. If you configured shift times when you installed IBM Tivoli Monitoring, you can also select for which shifts and days data is displayed. See your IBM Tivoli Monitoring documentation for further information.
  - In the **Start time and End time** fields, select the time period for which you want data to be displayed.

Exception: Summarization is not performed on the attributes of string, timestamp, and enumeration types. It is meaningless to view summarized data in some workspaces such as Broker Status Event.

5. Optional: To apply the time span to all other views in this workspace that use the same query, select the **Apply to all views associated with this view's query** check box.
6. To exit the Select the Time Span window, click **OK**.

The workspace is refreshed to reflect the time span you selected.

Offline collection of historical data

Historical data is collected for some workspaces even when the agent is stopped, provided that the **persistentBrokerData** option is YES. Because the reply queue used by the agent is persistent, message flow accounting, publish-subscribe statistics, and broker event data is still delivered to the queue when the agent is stopped. After the agent is started again, stored data is processed and can be viewed in historical information workspaces. Historical information for the following workspaces is collected when the agent is stopped:

- Archive Accounting Message Flow Statistics
- Archive Accounting Thread Statistics
- Archive Accounting Node Statistics
- Archive Accounting Terminal Statistics
- Broker Status Events
- Resource Statistics

Chapter 7. Running reports with Tivoli Common Reporting

Tivoli Common Reporting is a reporting feature available to users of Tivoli products, and provides a consistent approach to viewing and administering reports. Cognos® data model and sample reports for WebSphere Message Broker Monitoring agent are provided in a report package for use with Tivoli Common Reporting 2.1.1. The report package is a set of historical reports for both raw and summarized data that is collected in Tivoli Data Warehouse. You can use the sample reports to create you own reports on the data model.

Prerequisites

Before you can run reports with Tivoli Common Reports, make sure that the following requirements are met in your environment.

- The WebSphere Message Broker Monitoring agent is installed and running.
- The Warehouse Proxy Agent is started.
- The Summarization and Pruning Agent is started.
- Historical data collection is enabled for the related attribute group.
- Summarized tables and views are created and populated in the Tivoli Data Warehouse.

Tips:

- To check whether the required summarized tables and views have been created, run the following queries against Tivoli Data Warehouse. If the tables and views have been correctly created, you can see the result sets that contain HV, DV, WV, QV, and YV in each base table, for example, Broker_Statistics, Broker_Statistics_HV, Broker_Statistics_DV, Broker_Statistics_WV, Broker_Statistics_MV, Broker_Statistics_QV, and Broker_Statistics_YV.
  - DB2

    ```
    select distinct "TABNAME" from SYSCAT.TABLES where
    "TABSCHEMA" = 'ITMUSER'
    ```

  - Oracle

    ```
    select distinct "TABLE_NAME" from USER_TABLES
    ```

  - SQL Server

    ```
    select TABLE_NAME "VIEWNAME" from INFORMATION_SCHEMA.TABLES
    ```

- Cognos reports can be run against yearly, quarterly, monthly, weekly, daily, and hourly summarization intervals. You can decide which summarization interval is important to you to run reports against, and enable summarization for the related attribute group.
  - Accounting Message Flow Statistics
  - Accounting Node Statistics
  - Accounting Terminal Statistics
  - Accounting Thread Statistics
  - Broker Status
  - Broker Status Events
  - Components
  - Execution Group Status
  - File Resource Statistics

- JDBC Connection Pools Resource Statistics
- JVM Resource Statistics
- Message Flow Status
- Monitor Node Base Statistics
- Monitor Node Broker Statistics
- Monitor Node Events
- Monitor Node Execution Group Statistics
- Monitor Node Message Flow Statistics
- Monitor Node Sub-Flow Statistics
- ODBC Resource Statistics
- Parsers Resource Statistics
- Product Events
- SOAPInput Resource Statistics

Installing Cognos reports for WebSphere Message Broker Monitoring agent

Installing Cognos reports for WebSphere Message Broker Monitoring agent includes installing agent-specific reports, configuring Cognos data source, and creating Tivoli Reporting and Analytics Model (TRAM) dimensions. The Cognos reports for WebSphere Message Broker Monitoring agent must be installed on Tivoli Common Reporting server.

A report package for the WebSphere Message Broker Monitoring agent is provided in a folder named `REPORTS` on the installation disk. Use one of the following folders within the `REPORTS` folder to install Cognos reports depending on the database type of Tivoli Data Warehouse:

- DB2 and SQL Server: `ITCAM_Agents_for_WebSphere_Messaging_v71`
- Oracle: `ITCAM_Agents_for_WebSphere_Messaging_v71_for_Oracle`

1. If the report package folder is on a remote system, do one of the steps depending on the operating system where the Tivoli Common Reporting server is installed:
   - Windows: Map the remote folder to a local drive.
   - Systems other than Windows: Mount the remote system.
2. In the report package folder, do one of the following steps to start the installation program
   - Windows: Double-click the `setup_windows.exe` file.
   - Systems other than Windows: Run the `setup_<platform>.bin` file.
3. Select the language of the installation program and click **OK**.
4. In the Welcome window, click **Next**.
5. In the Choose the Installation Folder window, specify the path to the Tivoli Common Reporting component directory and click **Next**.

   Important: After this step, you cannot step back to this window to change the installation folder again. If you want to change the installation folder afterwards, cancel the installation program and start it again from Step 2.
6. In the Choose the reports for the installation window, select **WebSphere Message Broker Monitoring Agent Reports** and click **Next**.

7. In the Cognos Engine Configuration window, type the Tivoli Common Reporting user name and password, and then click **Next**.
8. In the Cognos Data Source TDW Configuration window, provide all required database information to define the Cognos data source, and click **Next**.

   Remember: If you choose to skip this step, you must configure the connection to a database to access your data after the installation is complete. For instructions about how to configure a data connection, see Tivoli Common Reporting documentation.
9. In the Data Script runDbScript Configuration window, provide all required database information to create TRAM dimensions and click **Next**.

   Important: The user ID that is used to run the database script must have administrator access to the IBM_TRAM schema. The script requires the administrator access to delete the IBM_TRAM schema related objects and re-create them.

   Remember: Do not skip this step, otherwise, you must manually add TRAM dimensions, which are required for running Cognos reports and using the data model.
10. Review the installation information and click **Install**.
11. After the installation is complete, click **Done** to exit.

Working with reports

Topics in this section provide instructions about how to run Cognos reports. If you did not use the installation wizard to install Cognos reports (see "Installing Cognos reports for WebSphere Message Broker Monitoring agent" on page 88), you must do the following steps before you can run Cognos reports:

1. Save the report package for WebSphere Message Broker Monitoring agent to the *TCR_component_dir*\cognos\deployment directory.
2. Import the WebSphere Message Broker Monitoring agent report package to Tivoli Common Reporting.
3. Create the Cognos data source for your report.

Tips:

- The report package for WebSphere Message Broker Monitoring agent is provided as a compressed file named WebSphereBrokerReports.zip on the installation disk in the following folders:
  - For DB2 and SQL Server: REPORTS\ITCAM_Agents_for_WebSphere_Messaging_v71\reports\cognos_reports\Messaging\packages\WebSphereBrokerReports.zip
  - For Oracle: REPORTS\ITCAM_Agents_for_WebSphere_Messaging_v71_for_Oracle\reports\cognos_reports\Messaging\packages\WebSphereBrokerReports_Oracle.zip
- For more information about how to work with reports using IBM Cognos 8 Business Intelligence Reporting, see the IBM Tivoli Common Reporting Information Center.

Creating or editing Web-based reports

You can create and customize your own reports using the Cognos Report Studio, which is a web-based report editor.

1. Log on to the Tivoli Common Reporting interface, and go to **Common Reporting**.
2. In the Work with reports window on the right, select **Report Studio** from the **Launch** list.
3. Use the menu controls to create a report or edit existing reports by formatting the layout and manipulating the data that appears in the report.
4. Save your report, and run it anytime you want to present on its underlying data.

Creating ad hoc reports

After you import the report package provided for WebSphere Message Broker Monitoring agent, you can create ad hoc reports by using simple queries and formatting.

To create ad hoc reports, perform the following steps:

1. Log on to the Tivoli Common Reporting interface, and go to **Common Reporting**.
2. In the Work with reports window on the right, select **Query Studio** from the **Launch** list.
3. Select the WebSphere Message Broker Monitoring Agent package. A new window is open and you can create a report.
4. From the navigation on the left, drag the data items that you want to include in your report.

   Remember: Always drag the attributes that belong to the same query to your report. If you drag two attributes from different queries, for example, one from Broker Statistics Daily and the other from Node Accounting Daily, the resulting data might be wrong. However, you can drag any time-related attributes from the Time identifier to your report.
5. Once you finish editing the report data and appearance, save the report by specifying a name, and optionally a description, and a screen tip.

   Tip: When you do an ad hoc query, you can either see the live data as you drag the items, or switch the mode to show placeholders for the data and then run the report.

Some other things that you can do in a report

- Combining metrics in one table

  You can combine different metrics into one table. For example, you can drag Trace Level from the Message Flow Status table, and Avg Elapsed Microseconds from the Accounting Message Flow Statistics table to the same table.
- Grouping data

  You can group the data by clicking the Type column and then the Group icon  .
- Creating a chart

  To create a chart, click the Chart icon  . A chart is created for each section with appropriate groups.

For more information about using Query Studio, see Query Studio User Guide at the IBM Cognos 8 v4 Business Intelligence Information Center.

Data model of WebSphere Message Broker Monitoring agent

The data model for WebSphere Message Broker Monitoring agent is a star schema with dimensions or identifiers that are separated from facts or metrics. Metrics are measurable (numeric) attributes, which can be aggregated by identifiers. The relationship among the metric tables is defined using two common identifiers, resources and time.

The WebSphere Message Broker Monitoring agent collects various metrics. All the metrics are modeled for the agent. The metrics are classified into two categories, key metrics and extended metrics.

Key metrics

The key metrics are the most important or the most frequently used metrics. The key metrics are divided into three groups, resource usage, performance, and availability. You can either see the raw metrics or summarized (daily and hourly) metrics.

Extended metrics

In addition to the key metrics, the extended metrics include all the other metrics of the WebSphere Message Broker Monitoring agent.

Identifiers

Identifiers are used to link metrics data across different agents. The two primary identifiers that are used by the WebSphere Message Broker Monitoring agent are resource and time.

Resource identifiers include execution group and host name. Time identifiers include various attributes of time, by which the metrics can be grouped, such as standard timestamp, date, minute, hour, day, week of year, month, quarter, or year.

Attributes

Attributes describe the identifiers. For example, the detailed information about a channel can be described by the attributes, such as channel type, query type, or command level.

Sample reports

Eight sample reports are provided in the report package for WebSphere Message Broker Monitoring agent. For the agent to get data to display in the sample reports, historical data collection must be enabled for the related attribute groups.

Table 5. Historical attribute groups for sample reports

Sample report	Attribute group
"Broker Daily Availability report" on page 92	Broker Status (detailed data)
"Broker Weekly Availability report" on page 96	

Table 5. Historical attribute groups for sample reports (continued)

Sample report	Attribute group
"Broker Execution Group Daily Availability report"	Execution Group Status (detailed data)
"Broker Execution Group Weekly Availability report" on page 93	
"Broker Message Flow Daily Availability report" on page 93	Message Flow Status (detailed data)
"Broker Message Flow Weekly Availability report" on page 95	
"Broker Top n Elapsed Microseconds report" on page 95	Accounting Message Flow Statistics (summarized data)
"Broker Message Flow Detail report" on page 94	

Broker Daily Availability report

This report shows the daily availability of the broker that you specified, including the availability of the whole day and the availability of 24 hours. Use this report to check the broker status in one day.

Table 6. Parameters of Broker Daily Availability report

Parameter group	Parameter name	Usage
Date Range	Report Period	Select the day you want to check the broker status.
Sampling Interval	Sampling Interval	Specify the collection interval in the TEP History Collection Configuration panel.
Resource Selection	Host Name	Specify the host name of the broker that you want to check.
	Broker Name	Specify the name of the broker that you want to monitor.
Record Type	Record Type	Select 1 for snapshot report and 2 for archived report. **Note:** WebSphere Message Broker V7.0 or later can only use archived report.

Broker Execution Group Daily Availability report

This report shows the daily availability of the execution group that you specified, including the availability of the whole day and the availability of 24 hours. Use this report to check the execution group status in one day.

Table 7. Parameters of Broker Execution Group Daily Availability report

Parameter group	Parameter name	Usage
Date Range	Report Period	Select the day you want to check the execution group status.
Sampling Interval	Sampling Interval	Specify the collection interval in the TEP History Collection Configuration panel.

Table 7. Parameters of Broker Execution Group Daily Availability report (continued)

Parameter group	Parameter name	Usage
Resource Selection	Host Name	Specify the host name of the broker that you want to check.
	Broker Name	Specify the name of the broker that you want to monitor.
	Execution Group Name	Select the execution group that runs within the broker.

Broker Execution Group Weekly Availability report

This report shows the weekly availability of the execution group that you specified, including the availability of a whole week and the availability of each day in the week. Use this report to check the execution group status in one week.

Table 8. Parameters of Broker Execution Group Weekly Availability report

Parameter group	Parameter name	Usage
Date Range	Report Period	Select one day of a week that you want to check the execution group status.
Sampling Interval	Sampling Interval	Specify the collection interval in the TEP History Collection Configuration panel.
Resource Selection	Host Name	Specify the host name of the broker that you want to check.
	Broker Name	Specify the name of the broker that you want to monitor.
	Execution Group Name	Select the execution group that runs within the broker.

Broker Message Flow Daily Availability report

This report shows the daily availability of the message flow that you specified, including the availability of the whole day and the availability of 24 hours. Use this report to check the message flow status in one day.

Table 9. Parameters of Broker Message Flow Daily Availability report

Parameter group	Parameter name	Usage
Date Range	Report Period	Select the day you want to check the message flow status.
Sampling Interval	Sampling Interval	Specify the collection interval in the TEP History Collection Configuration panel.
Resource Selection	Host Name	Specify the host name of the message flow that you want to check.
	Broker Name	Specify the name of the broker.
	Execution Group Name	Specify the name of the execution group.
	Message Flow Name	Specify the name of the message flow that you want to check.

Table 9. Parameters of Broker Message Flow Daily Availability report (continued)

Parameter group	Parameter name	Usage
Status	Warning	Specify an integer as the warning threshold. If the message flow availability percentage is lower than this threshold, the value is marked as warning status.
	Critical	Specify an integer as the critical threshold. If the message flow availability percentage is lower than this threshold, the value is marked as critical status.

Broker Message Flow Detail report

The report shows the usage of the selected message flow during the specified time period. The line chart shows maximum elapsed microseconds, average elapsed microseconds, and minimum elapsed microseconds that the message flow takes to process in different colors. The table shows detailed information about elapsed time.

You can use this report to identify the elapsed time trend over a specified period. The report can be run hourly, daily, weekly, monthly, quarterly, or yearly.

Table 10. Parameters of Broker Message Flow Detail report

Parameter group	Parameter group	Usage
Date Range	Report Period	Select the time span for the report from the predefined time range, such as Last Week, Current Month, Last 30 Days.
	Start Date	Select a start date from the calendar and start time from the time widget. You must specify both date and time.
	End Date	Select an end date from the calendar and start time from the time widget. You must specify both date and time.
Summarization Selection	Summarization Type	Select the summarization types, such as Hourly, Daily, Weekly, Monthly, Quarterly, Yearly from the list.
	Shift Period	If shifts are enabled, the hourly table displays the shift period as 1 or 2, depending on the peak and off-peak hours that are configured in the data warehouse. The daily table consists of 1 and 2 corresponding to the peak and off-peak hours, and -1 corresponding to the summarized value for that day. If shifts are not enabled, the default value is -1.
	Vacation Period	If the vacation period is not enabled, the default value is -1. You can enter the value 1 or 2, if the vacation period is enabled.
Resource Selection	Broker Name	Specify the name of the broker that you want to monitor.
	Execution Group Name	Specify the name of the execution group that runs within the broker.
	Message Flow Name	Specify the name of the message flow that is associated with the execution group.

Table 10. Parameters of Broker Message Flow Detail report (continued)

Parameter group	Parameter group	Usage
Record Type	Record Type	Select 1 for snapshot report and 2 for archived report. **Note:** WebSphere Message Broker V7.0 or later can only use archived report.

Broker Message Flow Weekly Availability report

This report shows the weekly availability of the message flow that you specified, including the availability of a whole week and the availability of each day in the week. Use this report to check the message flow status in one week.

Table 11. Parameters of Broker Message Flow Weekly Availability report

Parameter group	Parameter name	Usage
Date Range	Report Period	Select one day of a week that you want to check the message flow status.
Sampling Interval	Sampling Interval	Specify the collection interval in the TEP History Collection Configuration panel.
Resource Selection	Host Name	Specify the host name of the broker that you want to check.
	Broker Name	Specify the name of the broker that you want to monitor.
	Execution Group Name	Specify the name of the execution group.
	Message Flow Name	Specify the name of the message flow that you want to check.
Status	Warning	Specify an integer as the warning threshold. If the message flow availability percentage is lower than this threshold, the value is marked as warning status.
	Critical	Specify an integer as the critical threshold. If the message flow availability percentage is lower than this threshold, the value is marked as critical status.

Broker Top *n* Elapsed Microseconds report

This report shows the top *n* message flows that take the most average time in microseconds to process. By default, this report shows the top 20 message flows for the selected broker. The chart view of the report shows the top *n* message flows and highlights the average elapsed time in different colors according to the threshold that you specified. The table shows more details about each message flow.

You can use this report to identify the overused message flows of a collection of systems. The report can be run hourly, daily, weekly, monthly, quarterly, or yearly. You can also click the concerned message flow in the chart view and navigate to the Message Flow Detail report to see more details of the selected message flow.

Table 12. Parameters of Broker Top n Elapsed Microseconds report

Parameter group	Parameter name	Usage
Date Range	Report Period	Select the time span for the report from the predefined time range, such as Last Week, Current Month, Last 30 Days.
	Start Date	Select a start date from the calendar and start time from the time widget. You must specify both date and time.
	End Date	Select an end date from the calendar and start time from the time widget. You must specify both date and time.
Summarization Selection	Summarization Type	Select the summarization types, such as Hourly, Daily, Weekly, Monthly, Quarterly, Yearly from the list.
	Shift Period	If shifts are enabled, the hourly table displays the shift period as 1 or 2, depending on the peak and off-peak hours that are configured in the data warehouse. The daily table consists of 1 and 2 corresponding to the peak and off-peak hours, and -1 corresponding to the summarized value for that day. If shifts are not enabled, the default value is -1.
	Vacation Period	If the vacation period is not enabled, the default value is -1. You can enter the value 1 or 2, if the vacation period is enabled.
Number of Systems	Number of Systems to Display	Specify an integer as the number of top resources that you want to see.
Record Type	Record Type	Select 1 for snapshot report and 2 for archived report. **Note:** WebSphere Message Broker V7.0 or later can only use archived report.
Elapsed Microseconds Prompt Status	Warning	Specify an integer as the warning threshold. If the average elapsed time in microseconds of a message flow exceeds this threshold, the value is marked as warning status.
	Critical	Specify an integer as the critical threshold. If the average elapsed time in microseconds of a message flow exceeds this threshold, the value is marked as critical status.

Broker Weekly Availability report

This report shows the weekly availability of the broker that you specified, including the availability of a whole week and the availability of each day in the week. Use this report to check the broker status in one week.

Table 13. Parameters of Broker Weekly Availability report

Parameter group	Parameter name	Usage
Date Range	Report Period	Select one day of a week you want to check the broker status.
Sampling Interval	Sampling Interval	Specify the collection interval in the TEP History Collection Configuration panel.

Table 13. Parameters of Broker Weekly Availability report (continued)

Parameter group	Parameter name	Usage
Resource Selection	Host Name	Specify the host name of the broker that you want to check.
	Broker Name	Specify the name of the broker that you want to monitor.
Record Type	Record Type	Select 1 for snapshot report and 2 for archived report. **Note:** WebSphere Message Broker V7.0 or later can only use archived report.

Known problems and workarounds

This section contains problems that might occur when you work with Cognos reports.

Arithmetic overflow errors in ad hoc querying

If you drag certain columns in an ad hoc query and it returns an arithmetic overflow error, switch to preview with limited data or to preview with no data and add Standard Timestamp to the query.

Certain columns might average or sum up to a number that is larger than the size supported by the database. So the SQL error of arithmetic overflow is returned. If you see the data by hourly timestamp or daily timestamp, or set a query to limit the data, the aggregated value is forced to be within the supported size.

`No data available` in ad hoc querying on two tables

The `No data available` message is displayed in an ad hoc querying on two tables, but the resulting data indicates that the two tables are queried.

This error occurs because there is no relationship defined between the two tables. Make sure all your ad hoc queries have at least one identifier.

Errors of missing table or attribute

Make sure that all the prerequisites are met and the warehouse is collecting historical data. If you enable historical data collection appropriately, you are able to use the data model for WebSphere Message Broker Monitoring agent.

If you have all the required tables but still get this error, it might be because the WebSphere Message Broker Monitoring agent is not compatible with the version that is used in the generic model.

To check the query that runs, open the report in the Report Studio. Click **Tools** > **Show Generated SQL/MDX**. The queries in the report are displayed. You can view the native SQL.

The table schema is not ITMUser

If the table schema that you use is not ITMUser, use the framework manager to update the schema to what you used.

1. Extract the report package to your local system that the framework manager is installed on.

2. In the `model\WebSphere Message Broker Monitoring Agent v7.1 Data Model` folder, open the `WebSphere Message Broker Monitoring Agent v7.1 Data Model.cpf` file with the framework manager.
3. In the Project Viewer view, expand **Data Sources** and click **TDW**.
4. In the Property view, change the schema property from `ITMUser` to the schema that you used and save the changes.
5. In the Project Viewer view, expand **Packages**, right-click the WebSphere Message Broker Monitoring Agent package and click **Publish Packages** to publish the package to the Cognos server.

Duplicated broker names for the Broker Name selection

If the managed system node name that is related to a broker is changed, duplicate broker names might exist in the **Broker Name** field on the **Parameter Selection** page. This is because the node name change affect the data correlation. The data for the same broker is not correlated for together until after the old node name has rolled out of the warehouse over time.

To distinguish the duplicated broker names, edit the report in the report studio to change the Display Value property of the **`Broker Name`** parameter to Origin Node on the **Parameter Selection** page.

Remember: A change to any of the following values can cause the change of the managed system node name that is associated with the agent data for brokers:

- WebSphere Message Broker Monitoring agent ID
- Name of the broker
- Alias name of the broker

Chapter 8. Configuring in a cluster environment on Windows systems

This section contains instructions for configuring the WebSphere Message Broker Monitoring agent to run in a Microsoft Cluster Service (MSCS) cluster environment on Windows systems.

MSCS clusters are different from WebSphere MQ clusters, as follows:

WebSphere MQ clusters
: WebSphere MQ clusters are groups of two or more queue managers running on one or more computers, providing automatic interconnection, and allowing queues to be shared for load balancing and redundancy.

MSCS clusters
: MSCS clusters are groups of two or more computers, connected together and configured in such a way that, if one fails, MSCS performs a failover, transferring the state data of applications from the failing computer to another computer in the cluster and re-initiating their operation there.

You can use MSCS to connect servers into a cluster, giving higher availability of data and applications, and making it easier to manage the system. MSCS can automatically detect and recover from server or application failures. For more information about MSCS clusters, see "MSCS clusters."

The WebSphere Message Broker Monitoring agent supports both active/active and active/passive clustering. If you are configuring the agent in an active/active cluster environment, see "Active/active clustering" on page 101 for instructions. If you are configuring the agent in an active/passive cluster environment, see "Active/passive clustering" on page 106 for instructions.

MSCS clusters

By using the Microsoft Cluster Service (MSCS), you can connect servers into a cluster, which provides higher availability of data and applications, and makes it easier to manage the system. MSCS can automatically detect and recover from server or application failures.

MSCS supports failover of virtual servers, which correspond to applications, Web sites, print queues, or file shares (including their disk spindles, files, IP addresses, and other aspects).

A failover is an automatic operation that switches to a redundant or standby system in an event of a software, hardware, or network interruption. By using the failover operation, MSCS detects a failure in an application on one computer in the cluster, and shuts down the disrupted application in an orderly manner, transfers its state data to the other computer, and re-initiates the application there.

To understand how a failover works, start by looking at a two-computer cluster. A *two-computer cluster* comprises two computers (for example, A and B) that are jointly connected to a network for client access by using a virtual IP address. They might also be connected to each other by one or more private networks. A and B share at least one disk for the server applications on each to use. Another shared

disk must be set up as a redundant array of independent disks (RAID) Level 1, for the exclusive use of MSCS; this shared disk is known as the *quorum disk*. You use MSCS to monitor both computers to check that the hardware and software are running correctly.

In a simple setup of a two-computer cluster, both computers have all the applications installed on them, but only computer A runs with live applications; computer B is just running and waiting. If computer A encounters any one of a range of problems, MSCS shuts down the disrupted application in an orderly manner, transfers its state data to the other computer, and re-initiates the application on the second computer. Applications can be made cluster-aware so that they interact fully with MSCS and fail over gracefully.

A typical setup for a two-computer cluster is shown in Figure 14.

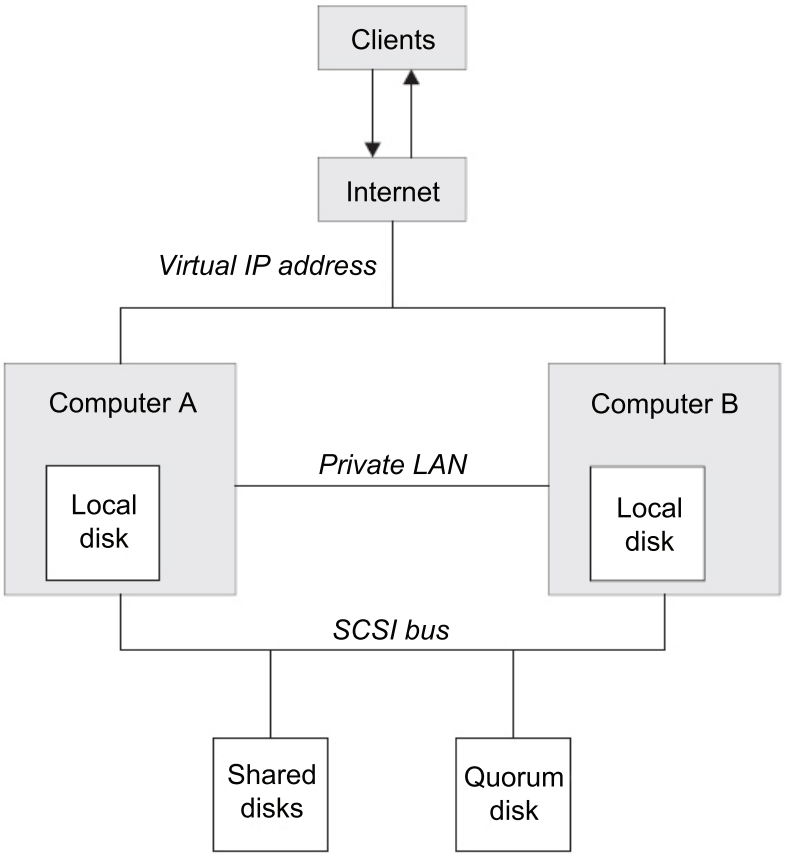

Figure 14. A two-computer MSCS cluster

Each computer can access the shared disk, but only one at a time, under the control of MSCS. In the event of a failover, MSCS switches the access to the other computer. The shared disk is typically a RAID, but need not be.

Each computer is connected to the external network for client access, and each has an IP address. However, an external client, communicating with this cluster, sees only one virtual IP address, and MSCS routes the IP traffic within the cluster appropriately.

MSCS also communicates between the two computers, either over one or more private connections or over the public network, to monitor their states by using the heartbeat signal, synchronize their databases, and perform other related tasks.

Active/active clustering

This section describes how you can configure the WebSphere Message Broker Monitoring agent to run in an active/active cluster environment.

Prerequisites

Before you begin configuring the WebSphere Message Broker Monitoring agent to run in an active/active cluster environment, ensure that the two systems that host the WebSphere Message Broker Monitoring agent are correctly configured. Ensure that both systems fulfill the following requirements:

- Microsoft Windows 2003 Server is installed and includes Microsoft Cluster Server (MSCS), which is used to manage your cluster environment.
- You have used MSCS to configure both systems as cluster nodes.
- WebSphere Message Broker is installed and configured to run in a cluster environment. See your WebSphere Message Broker documentation for information about how to install WebSphere Message Broker in a cluster environment.
- WebSphere MQ is installed and configured to run in a cluster environment. See your WebSphere MQ documentation for information about how to install WebSphere MQ in a cluster environment.
- Message brokers and configuration managers that you want to monitor are created.
- The DB2 database is installed to run in a domain environment. See the DB2 documentation for information about how to install the DB2 software in a domain environment.
- The IBM Tivoli Monitoring framework is installed. This monitoring software must be installed separately on both cluster nodes. For instructions about how to install IBM Tivoli Monitoring in a cluster environment, see your IBM Tivoli Monitoring documentation.
- The WebSphere Message Broker Monitoring agent version 7.0 or later version is installed. This agent must be installed separately on both cluster nodes. See *Installation and Setup Guide* for installation instructions.

Also ensure that you have two separate logical drives in the cluster environment available for storing log and historical data collected from the agents. These drives are referenced as drives R and S in the following procedure.

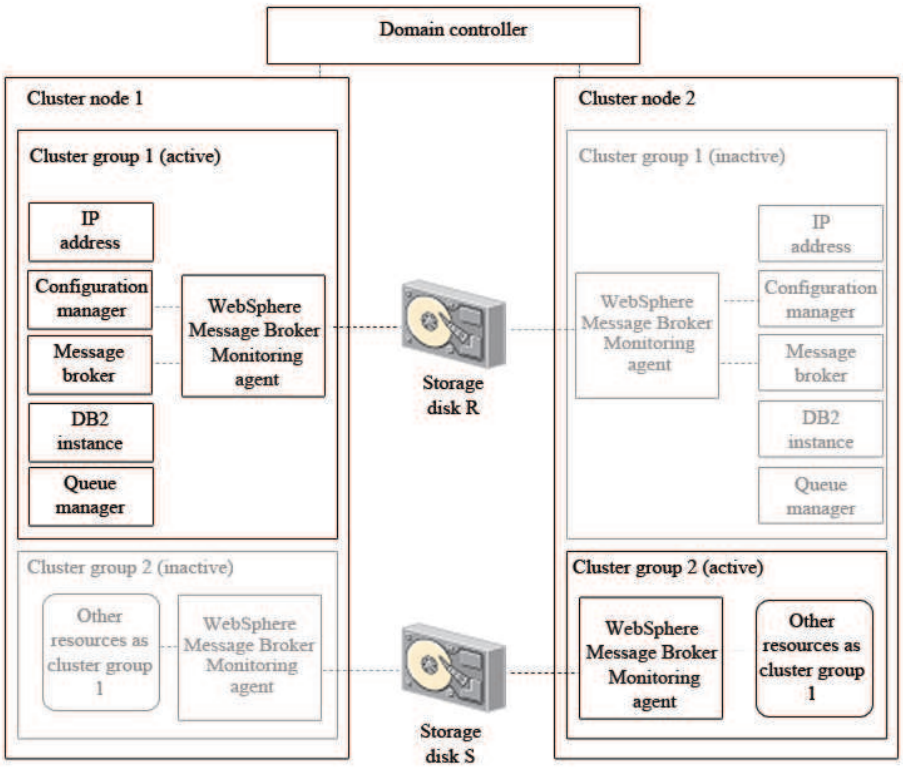

Figure 15. An example active/active cluster environment architecture with one cluster group active on each cluster node

An example of an active/active cluster environment is displayed in Figure 15. The environment consists of two cluster nodes on separate physical systems. Each cluster node hosts two cluster groups. The cluster groups hosted by each system are the same, so between them there are two identical copies of cluster group 1 and two identical copies of cluster group 2. Each cluster group contains the following resources:

- A message broker
- A configuration manager
- A DB2 instance
- An IP address
- A queue manager
- A WebSphere Message Broker Monitoring agent

Only one copy of each cluster group can be active simultaneously. For example, when cluster group 1 is active on cluster node 1 (as in Figure 15), the copy of cluster group 1 on cluster node 2 is inactive. In most environments with two cluster nodes and two cluster groups where both cluster nodes are running correctly, one cluster group is running on each cluster node, balancing the load between the two systems. If one of the nodes fails, the second cluster group on the node that is still active is started to continue the work of the cluster group that was active on the node that failed.

Information shared between different copies of the same agent, such as logs, is stored on a separate disk that can be accessed by all copies of the agent running on different cluster nodes. If the node that hosts the active agent fails and a copy of

the agent on the other node is started, shared information such as log files can still be read and written to the disk as if the same copy of the agent was still running. The agent is installed separately on each cluster node.

Configuring the WebSphere Message Broker Monitoring agent

To configure the WebSphere Message Broker Monitoring agent to run in a cluster environment, complete the following steps:

Important: To complete the following procedure, you must have two cluster groups on each cluster node because it is the most common scenario. If you have more than two cluster groups, create additional instances of the WebSphere Message Broker Monitoring agent to monitor the message brokers in each additional cluster group.

1. Create new instances of the WebSphere Message Broker Monitoring agent on both cluster nodes.
2. Set local variables on each cluster node.
3. Configure the Tivoli Enterprise Portal Server to list agents running in cluster groups by cluster name instead of host system name in Tivoli Enterprise Portal.
4. Use the Cluster Administrator ID to add a resource of the **Generic Service** type named KQI1 to cluster group 1.
5. Use the Cluster Administrator ID to add a resource of the **Physical Disk** type named R to cluster group 1.
6. Use the Cluster Administrator ID to add a resource of the **Generic Service** type named KQI2 to cluster group 2.
7. Use the Cluster Administrator ID to add a resource of the **Physical Disk** type named S to cluster group 2.
8. Use the Cluster Administrator ID to set the group owner of cluster group 1 to cluster node 1 and the group owner of cluster group 2 to cluster node 2.
9. Use the Cluster Administrator ID to start the broker, configuration manager, WebSphere Message Broker Monitoring agent, and other resources in each cluster group.

You have now completed the configuration of the WebSphere Message Broker Monitoring agent to monitor brokers in an active/active cluster environment.

Creating new instances of the WebSphere Message Broker Monitoring agent

Create new instances of the WebSphere Message Broker Monitoring agent on both cluster nodes by performing the following procedure:

1. Start Manage Tivoli Enterprise Monitoring Services.
2. Right-click the WebSphere Message Broker Monitoring agent, and then click **Create Instance** to create a new instance of the WebSphere Message Broker Monitoring agent to monitor the broker and configuration manager in cluster group 1.
3. Enter a name for the instance when prompted. In this procedure, use `KQI1` for the instance name. Click **OK**.
4. Right-click the WebSphere Message Broker Monitoring agent, and then click **Create Instance** again to create a second new instance of the WebSphere Message Broker Monitoring agent to monitor the broker and configuration manager in cluster group 2.
5. Enter a name for the instance when prompted. In this procedure, use `KQI2` for the instance name. Click **OK**.

6. Edit the KQI1 and KQI2 configuration files as follows:
   a. Add the following attribute directly after the version attribute inside the KqiAgent tag, where *instance_name* is the name of the WebSphere Message Broker Monitoring agent instance. Enter a string of up to four characters. The instance name must be different from all other agent instance names that run on the same system node. However, this value must be identical to the value of the *instance_name* used for the agent instance monitoring the failover broker by the same name on the failover node.

      ```
      agentId="instance_name"
      ```

   b. Add the following tags directly after the opening KqiAgent tag, where *broker_name* is the name of the message broker you want to monitor.

      ```
      <MonitorBroker name="broker_name"> </MonitorBroker>
      ```

   c. Optional: If the agent is only monitoring a single message broker, set the **discoveryInterval** parameter to a high value. This setting can reduce the number of unnecessary broker discover operations performed. For example, you can set the value to 2592000 seconds (30 days).

      The resulting file looks similar to the following example:

      ```
      <KqiAgent version="710"
        agentId="KQI"
        defaultRetainBrokerEvents="10"
        defaultRetainFlowEvents="10"
        retainProductEvents="10"
        discoveryInterval="2592000"
        defaultStatisticInterval="60"
        defaultFlowEventInterval="15"
        defaultHistoricalAccountingType="Archive"
        defaultRetainRecentSnapshotSamples="15"
        defaultRetainRecentArchiveSamples="5"
        defaultRetainRecentPubSubSamples="15"
        holdTimeForQuery="180"
        defaultReplyQueueName="KQI.AGENT.REPLY.QUEUE"
        defaultReplyQueueModel="SYSTEM.BROKER.MODEL.QUEUE"
        defaultTakeActionAuthUsers="*">
        <MonitorBroker name="BK1"> </MonitorBroker>
      </KqiAgent>
      ```

   d. Stop the primary WebSphere Message Broker Monitoring agent instance.

      Remember: Do not use the primary WebSphere Message Broker Monitoring agent to monitor brokers in a cluster environment.

Setting local variables on each cluster

Set local variables on each cluster node by performing the following procedure:

1. Start Manage Tivoli Enterprise Monitoring Services.
2. Right-click the KQI1 agent instance that you created in the previous procedure, and then click **Advanced** > **Edit Variables**.
3. In the Override Local Variable Settings window, add the following variables and specify their values:
   - CTIRA_SIT_PATH
   - CTIRA_LOG_PATH
   - CTIRA_HIST_DIR

   Remember: Each agent must have its own logical drive on which to store data. More than one agent cannot share a single drive.
   If you specify the values as in the following example, a directory named `R:\WMB\kqi\BK\logs` is created to store the logs of the WebSphere Message Broker Monitoring agent, the `R:\WMB\kqi\BK\logs\History` directory is created

to store historical data of the agent, and the R:\WMB\kqi\BK\sitpath directory is created to store the situation data, where R is the letter assigned to the drive where log files and historical data collected by KQI1 agent are stored:

```
CTIRA_SIT_PATH=R:\WMB\kqi\BK\sitpath
CTIRA_LOG_PATH=R:\WMB\kqi\BK\logs
CTIRA_HIST_DIR=R:\WMB\kqi\BK\logs\History
```

4. To close the window, click **OK**.
5. Right-click the KQI2 agent instance that you created in the previous procedure, and then click **Advanced** > **Edit Variables**.
6. In the Override Local Variable Settings window, add the following variables and specify their values:
   - CTIRA_SIT_PATH
   - CTIRA_LOG_PATH
   - CTIRA_HIST_DIR

   If you specify their values as in the following example, a directory named S:\WMB\kqi\BK\logs is created to store the logs of the WebSphere Message Broker Monitoring agent, S:\WMB\kqi\BK\logs\History is created to store its historical data, and S:\WMB\kqi\BK\sitpath is created to store its situation data, where S is the letter assigned to the drive where log files and historical data collected by KQI2 agent are stored. Make sure that the path names do not contain any spaces.

   ```
   CTIRA_SIT_PATH=S:\WMB\kqi\BK\sitpath
   CTIRA_LOG_PATH=S:\WMB\kqi\BK\logs
   CTIRA_HIST_DIR=S:\WMB\kqi\BK\logs\History
   ```
7. To close the window, click **OK**.
8. Change the start mode of both KQI1 and KQI2 agents to manual startup.

Configuring the Tivoli Enterprise Portal to list agents running in the cluster groups

Use the following procedure to configure the Tivoli Enterprise Portal to list agents running in cluster groups by cluster name instead of host system name in the portal:

1. In the Manage Tivoli Enterprise Monitoring Services window, right-click the agent instance icon, and then click **Advanced** > **Edit Variables** from the displayed menu.
2. In the Override Local Variable Settings window, click **Add**.
3. Select CTIRA_HOSTNAME for the **Variable** field, set the value to the name of the MSCS cluster that you want to display on Tivoli Enterprise Portal, and click **OK**.

   Important: Leave .TYPE=REG_EXPAND_SZ in the CTIRA_HOSTNAME variable when you change the value so that the registry is updated correctly.
4. Perform one of the following procedures, depending on which operating system the Tivoli Enterprise Portal Server is running.
   - If the Tivoli Enterprise Portal Server is running on a Windows system, perform the following procedure:
     a. Stop the portal server if it is running.
     b. In the Manage Tivoli Enterprise Monitoring Services window, right-click the Tivoli Enterprise Portal Server, and then click **Advanced** > **Edit Variables** from the displayed menu. The Tivoli Enterprise Portal Server Override Local Variable Setting window is displayed.
     c. Click **Add**. The Add Environment Setting Override window is displayed.

d. In the **Variable** menu, look for the **KFW_TOPOLOGY_CLUSTER_LIST** parameter. If the parameter exists, append AFF_MQSI_AGENT AFF_MQSI_BROKER to any existing values, separated by a space. If it does not already exist, create it and set the value to AFF_MQSI_AGENT AFF_MQSI_BROKER.

e. Reconfigure and recycle the portal server.

- If the Tivoli Enterprise Portal Server is running on a UNIX or Linux system, perform the following procedure:

a. Stop the portal server if it is running.

b. Navigate to the *ITM_HOME*/config directory, where *ITM_HOME* is the IBM Tivoli Monitoring installation directory.

c. Open the cq.ini file in a text editor.

d. Look for the **KFW_TOPOLOGY_CLUSTER_LIST** parameter. If it already exists, add AFF_MQSI_AGENT AFF_MQSI_BROKER to any existing values, separated by a space. If it does not already exist, add the following line to the file:

```
KFW_TOPOLOGY_CLUSTER_LIST=AFF_MQSI_AGENT AFF_MQSI_BROKER
```

e. Save and close the file.

f. Reconfigure and recycle the portal server.

Active/passive clustering

This section describes how you can configure the WebSphere Message Broker Monitoring agent to run in an active/passive cluster environment.

Prerequisites

Before you begin configuring the WebSphere Message Broker Monitoring agent to run in an active/passive cluster environment, ensure that the two systems that host the WebSphere Message Broker Monitoring agent are correctly configured. Ensure that both systems fulfill the following requirements:

- Microsoft Windows 2003 Server is installed that includes Microsoft Cluster Server (MSCS), which is used to manage your cluster environment.
- You have used MSCS to configure both systems as cluster nodes.
- WebSphere Message Broker is installed and configured to run in a cluster environment. See your WebSphere Message Broker documentation for information about how to install it in a cluster environment.
- WebSphere MQ is installed and configured to run in a cluster environment. See your WebSphere MQ documentation for information about how to install it in a cluster environment.
- The DB2 database is installed to run in a cluster environment. See the DB2 documentation for information about how to install the DB2 software in a cluster environment.
- Message brokers and configuration managers that are to be monitored are created.
- The IBM Tivoli Monitoring framework is installed. This monitoring software must be installed separately on both cluster nodes. For instructions about how to install IBM Tivoli Monitoring in a cluster environment, see the IBM Tivoli Monitoring documentation.
- The WebSphere Message Broker Monitoring agent version 7.0 or later version is installed. This agent must be installed separately on both cluster nodes. See *Installation and Setup Guide* for installation instructions.

Also ensure that you have a separate logical drive in the cluster environment available for storing log and historical data collected from the agents. The drive is referenced as drive R in the following procedure.

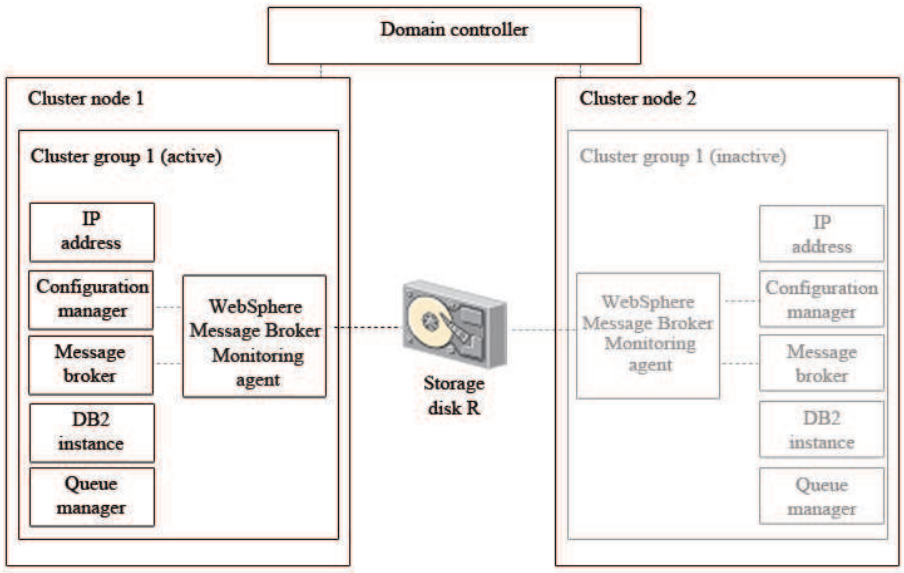

Figure 16. An example cluster environment architecture with a cluster group active on one cluster node and inactive on the other node

An example of an active/passive cluster environment is displayed in Figure 16. The environment consists of two cluster nodes on separate physical systems. Each cluster node hosts one cluster group (There is no limit to the number of cluster groups that can be hosted by a cluster node). The cluster groups hosted by each system are the same, so between them there are two identical copies of cluster group 1. Each cluster group contains the following components:

- A message broker
- A configuration manager
- A DB2 instance
- An IP address
- A queue manager
- A WebSphere Message Broker Monitoring agent

Only cluster groups on one cluster node are active at one time. For example, if cluster group 1 is active on cluster node 1 (as in Figure 16), the copy of cluster group 1 on cluster node 2 is inactive. In an active/passive cluster environment with two cluster nodes, only cluster groups on the active cluster node are running. If the active node fails, the cluster groups on the other node are started to continue the work of the cluster groups that were active on the node that failed.

Information shared between different copies of the same agent, such as logs, is stored on a separate disk that can be accessed by all copies of the agent running on different cluster nodes. If the node that hosts the active agent fails and a copy of the agent on the other node is started, shared information such as log files can still be read and written to the disk as if the same copy of the agent was still running. The agent is installed separately on each cluster node. Shared disks store logs and historical information that must be accessed by different copies of the same agent.

Configuring the WebSphere Message Broker Monitoring agent

To configure the WebSphere Message Broker Monitoring agent to run in an active/passive cluster environment, perform the following procedure:

Important: To complete the following procedure, you must one cluster group on each cluster node. If you have more than one cluster group, create additional instances of the WebSphere Message Broker Monitoring agent to monitor the resources in each additional cluster group.

1. Create a new instance of the WebSphere Message Broker Monitoring agent on both cluster nodes.
2. Set local variables on each cluster node.
3. Configure the Tivoli Enterprise Portal Server to list agents running in cluster groups by cluster name instead of host system name in Tivoli Enterprise Portal.
4. Use the Cluster Administrator ID to add a resource of type **Generic Service** named KQI to cluster group 1.
5. Use the Cluster Administrator ID to add a resource of type **Physical Disk** named R to cluster group 1.
6. Use the Cluster Administrator ID to set the group owner of cluster group 1 to cluster node 1.
7. Use the Cluster Administrator ID to start the broker, configuration manager, the WebSphere Message Broker Monitoring agent, and other resources in cluster group 1 on cluster node 1 or cluster node 2.

Now the configuration of the WebSphere Message Broker Monitoring agent to monitor brokers in an active/passive cluster environment is completed.

Creating new instances of the WebSphere Message Broker Monitoring

Remember: Do not use the primary WebSphere Message Broker Monitoring agent to monitor brokers in a cluster environment.

Create a new instance of the WebSphere Message Broker Monitoring agent by performing the following procedure on both cluster nodes:

1. Start Manage Tivoli Enterprise Monitoring Services.
2. Right-click the WebSphere Message Broker Monitoring agent, and then click **Create Instance** to create a new instance of the WebSphere Message Broker Monitoring agent to monitor the broker and the configuration manager in cluster group 1.
3. Enter a name for the instance when prompted. In this procedure, it is presumed you entered the name `KQI`. Click **OK**.
4. Edit the KQI agent configuration file as follows:
   a. Add the following attribute directly after the version attribute inside the KqiAgent tag:

      ```
      agentId="instance_name"
      ```

      where *instance_name* is the name of the WebSphere Message Broker Monitoring agent instance. Enter a string of up to four characters. The instance name must be different from all other agent instance names that run on the same system node. However, this value must be identical to the value of the *instance_name* used for the agent instance monitoring the failover broker by the same name on the failover node.

b. Add the following tags directly after the opening KqiAgent tag:

```
<MonitorBroker name="broker_name"> </MonitorBroker>
```

where *broker_name* is the name of the message broker that you want to monitor.

c. Optional: If the agent is only monitoring a single message broker, set the **discoveryInterval** parameter to a high value. This setting can reduce the number of unnecessary operations that are performed to discover brokers. For example, you can set the value to 2592000 seconds (30 days).

The resulting file looks similar to the following example:

```
<KqiAgent version="710"
  agentId="KQI"
  defaultRetainBrokerEvents="10"
  defaultRetainFlowEvents="10"
  retainProductEvents="10"
  discoveryInterval="2592000"
  defaultStatisticInterval="60"
  defaultFlowEventInterval="15"
  defaultHistoricalAccountingType="Archive"
  defaultRetainRecentSnapshotSamples="15"
  defaultRetainRecentArchiveSamples="5"
  defaultRetainRecentPubSubSamples="15"
  holdTimeForQuery="180"
  defaultReplyQueueName="KQI.AGENT.REPLY.QUEUE"
  defaultReplyQueueModel="SYSTEM.BROKER.MODEL.QUEUE"
  defaultTakeActionAuthUsers="*">
  <MonitorBroker name="BK1"> </MonitorBroker>
</KqiAgent>
```

5. Stop the primary WebSphere Message Broker Monitoring agent instance.

Setting local variables on each cluster

Set local variables by performing the following procedure on each cluster node:

1. Start Manage Tivoli Enterprise Monitoring Services.
2. Right-click the KQI agent instance that you create in the previous procedure, and then click **Advanced** > **Edit Variables**.
3. In the Override Local Variable Settings window, add the following variables and specify their values:
   - CTIRA_SIT_PATH
   - CTIRA_LOG_PATH
   - CTIRA_HIST_DIR

   If you specify their values as in the following example, a directory named `R:\WMB\kqi\BK\logs` is created to store the logs of the WebSphere Message Broker Monitoring agent and `R:\WMB\kqi\BK\logs\History` is created to store its historical data. However, you need to manually create the directory specified for CTIRA_SIT_PATH on the shared disk, which is `R:\WMB\kqi\BK\sitpath` in this example.

   ```
   CTIRA_SIT_PATH=R:\WMB\kqi\BK\sitpath
   CTIRA_LOG_PATH=R:\WMB\kqi\BK\logs
   CTIRA_HIST_DIR=R:\WMB\kqi\BK\logs\History
   ```

   where R is the letter assigned to the drive where log files and historical data collected by WebSphere Message Broker Monitoring agent are stored.

   Remember:

   - Make sure that the path names do not contain any spaces.

- Each agent must have its own logical drive on which to store data. More than one agent cannot share a single drive.

4. To close the window, click **OK**.
5. Change the start mode of both KQI agents to manual startup.

Configuring the Tivoli Enterprise Portal to list agents running in the cluster groups

Use the following procedure to configure the Tivoli Enterprise Portal to list agents running in cluster groups by cluster name instead of host system name in the portal:

1. In the Manage Tivoli Enterprise Monitoring Services window, right-click the agent instance icon, and then click **Advanced** > **Edit Variables** from the displayed menu.
2. In the Override Local Variable Settings window, click **Add**.
3. Select CTIRA_HOSTNAME for the **Variable** field, set the value to the name of the MSCS cluster that you want to display on Tivoli Enterprise Portal, and click **OK**.

   Important: Leave `.TYPE=REG_EXPAND_SZ` in the CTIRA_HOSTNAME variable when you change the value so that the registry is updated correctly.
4. Perform one of the following procedures, depending on which operating system the Tivoli Enterprise Portal Server is running.
   - If the Tivoli Enterprise Portal Server is running on a Windows system, perform the following procedure:
     a. Stop the portal server if it is running.
     b. In the Manage Tivoli Enterprise Monitoring Services window, right-click the Tivoli Enterprise Portal Server, and then click **Advanced** > **Edit Variables** from the displayed menu. The Tivoli Enterprise Portal Server Override Local Variable Setting window is displayed.
     c. Click **Add**. The Add Environment Setting Override window is displayed.
     d. In the **Variable** menu, look for the **KFW_TOPOLOGY_CLUSTER_LIST** parameter. If the parameter exists, append `AFF_MQSI_AGENT AFF_MQSI_BROKER` to any existing values, separated by a space. If it does not already exist, create it and set the value to `AFF_MQSI_AGENT AFF_MQSI_BROKER`.
     e. Reconfigure and recycle the portal server.
   - If the Tivoli Enterprise Portal Server is running on a UNIX or Linux system, perform the following procedure:
     a. Stop the portal server if it is running.
     b. Navigate to the *ITM_HOME*`/config` directory, where *ITM_HOME* is the IBM Tivoli Monitoring installation directory.
     c. Open the `cq.ini` file in a text editor.
     d. Look for the **KFW_TOPOLOGY_CLUSTER_LIST** parameter. If it already exists, add `AFF_MQSI_AGENT AFF_MQSI_BROKER` to any existing values, separated by a space. If it does not already exist, add the following line to the file:
        ```
        KFW_TOPOLOGY_CLUSTER_LIST=AFF_MQSI_AGENT AFF_MQSI_BROKER
        ```
     e. Save and close the file.
     f. Reconfigure and recycle the portal server.

Known limitations

Occasionally, the status of a broker and the broker configuration manager associated with it might be different in the Broker Information workspace in Tivoli Enterprise Portal. The information displayed in the portal might also be different from that displayed in the Cluster Administrator. However, the information displayed in Tivoli Enterprise Portal is correct.

This situation occurs because of the MSCS cluster failing to completely stop the broker. If this happens, manually stop the component that is not running (either the broker or configuration manager) and then restart it again if necessary. Run the following command to stop the component:

```
mqsistop <component_name>
```

Chapter 9. Configuring in a cluster environment on AIX systems

This section contains instructions for configuring the WebSphere Message Broker Monitoring agent to run in a cluster environment on AIX systems using High Availability Cluster Multi Processing (HACMP™). For information about how to configure hardware such as redundant power supplies, redundant disk controllers, disk mirroring, or multiple network or adapter configurations, see the HACMP documentation.

The WebSphere Message Broker Monitoring agent can be configured to run in either an active/active or active/passive environment. Before you begin configuring the WebSphere Message Broker Monitoring agent, ensure that your environment fulfills the requirements stated in "Prerequisites."

For instructions about how to install and configure theWebSphere Message Broker Monitoring agent for use in an HACMP cluster environment, see "Configuring the WebSphere Message Broker Monitoring agent" on page 116

Prerequisites

The WebSphere Message Broker Monitoring agent can be configured to run in either an active/active or active/passive environment. Before you begin configuring the WebSphere Message Broker Monitoring agent, be sure to review the following requirements for either environment that the two systems forming the cluster nodes must fulfill:

- "Active/active clustering"
- "Active/passive clustering" on page 115

Active/active clustering

Before you begin configuring the WebSphere Message Broker Monitoring agent to run in an HACMP active/active cluster environment, ensure that the two systems that form the cluster nodes in the environment are correctly configured. Both systems must fulfill the following requirements:

- The HACMP software is installed and your HACMP cluster environment is correctly configured.
- Both cluster nodes have access to a minimum of two shared disks, on which historical information shared between copies of the WebSphere Message Broker Monitoring agent running on different cluster nodes is stored. You must have a separate shared disk available in your cluster environment for each instance of the agent. If you want to have more than two agents running on each cluster node, increase the number of shared disks accordingly.
- WebSphere MQ is installed and configured to run in an HACMP cluster environment. See your WebSphere MQ documentation for information about how to install WebSphere MQ in a cluster environment.
- The message brokers that you want to monitor have been created on both cluster nodes within the HACMP cluster environment. Ensure that failover occurs correctly. See your message broker product documentation for more information.

- The DB2 database is configured to run in an HACMP environment. See the DB2 documentation for information about how to install the DB2 software in an HACMP environment.

An example of a cluster environment is displayed in Figure 17. The environment consists of two cluster nodes running on separate physical systems. Each cluster node hosts two cluster groups. The cluster groups hosted by each system are the same, so between them there are two identical copies of cluster group 1 and two identical copies of cluster group 2. Each cluster group contains one or more message brokers and an instance of the WebSphere Message Broker Monitoring agent to monitor each message broker.

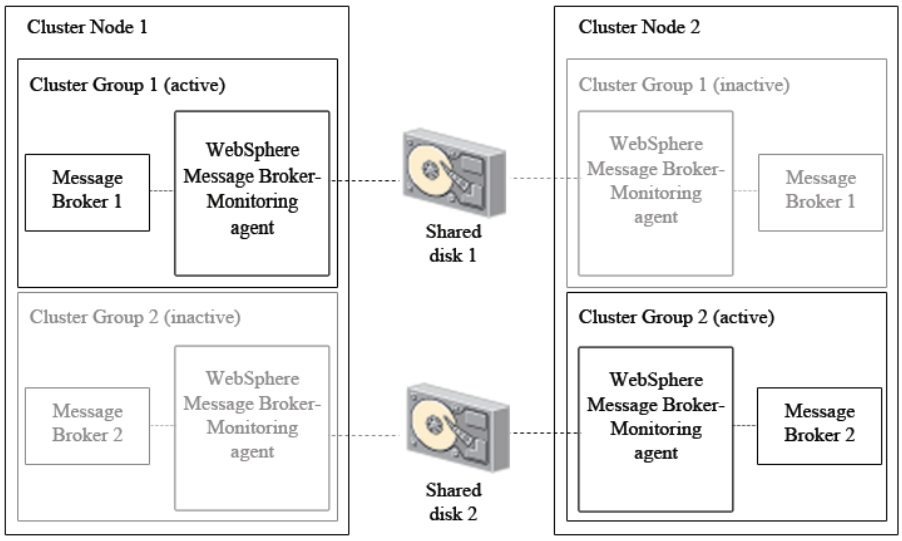

Figure 17. An example active-active cluster environment architecture with one cluster group active on each cluster node

Only one copy of each cluster group can be active simultaneously. For example, when cluster group 1 is active on cluster node 1 (as in Figure 17), the copy of cluster group 1 on cluster node 2 is inactive. In most environments with two cluster nodes and two cluster groups where both cluster nodes are running correctly, only one cluster group runs on each cluster node, balancing the load between the two systems. When one of the nodes fails, the second cluster group on the node that is still active is started to continue the work of the cluster group that was active on the node that failed.

Information shared between different copies of the same agent, such as historical data files, is stored on a separate disk that can be accessed by all copies of that agent running on different cluster nodes. In active/active clustering, at least two instances of the agent run on each cluster node, each requiring a separate shared disk to store shared information. If the node that hosts the active agent fails and a copy of the agent on the other node is started, shared information such as historical data files can still be read and written to the disk as if the same copy of the agent was still running.

To install and configure the WebSphere Message Broker Monitoring agent, repeat the procedure in "Configuring the WebSphere Message Broker Monitoring agent" on page 116 for each instance of the WebSphere Message Broker Monitoring agent in your environment.

Important: You must repeat this procedure for different copies of the same agent instance running on different cluster nodes.

Active/passive clustering

Before you begin configuring the WebSphere Message Broker Monitoring agent to run in an HACMP active/passive cluster environment, ensure that the two systems that form the cluster nodes in the environment are correctly configured. Both systems must fulfill the following requirements:

- The HACMP software is installed and your HACMP cluster environment is correctly configured.
- Both cluster nodes have access to a shared disk, on which historical information shared between copies of the WebSphere Message Broker Monitoring agent running on different cluster nodes is stored.
- WebSphere MQ is installed and configured to run in an HACMP cluster environment. See your WebSphere MQ documentation for information about how to install WebSphere MQ in a cluster environment.
- The message brokers that you want to monitor have been created on both cluster nodes within the HACMP cluster environment. Ensure that failover occurs correctly. See your message broker product documentation for more information.
- The DB2 database is configured to run in an HACMP environment. See your DB2 documentation for information about how to install the DB2 software in an HACMP environment.

An example of an active/passive cluster environment is displayed in Figure 18. The environment consists of two cluster nodes running on separate physical systems. The cluster groups hosted by each system are the same, so between them there are two identical copies of cluster group 1. Each cluster group contains one or more queue managers and an instance of the WebSphere Message Broker Monitoring agent to monitor each message broker.

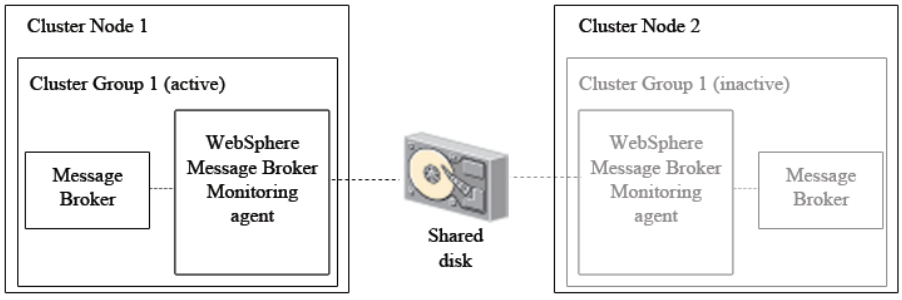

Figure 18. An example active-passive cluster environment architecture

Only cluster groups on one cluster node are active at one time. For example, when cluster group 1 is active on cluster node 1 (as in Figure 18), the copy of cluster group 1 on cluster node 2 is inactive. In an active/passive cluster environment

with two cluster nodes, only cluster groups on the active cluster node are running. If the active node fails, the cluster groups on the other node are started to continue the work of the cluster groups that were active on the node that failed.

Information shared between different copies of the same agent, such as historical data files, is stored on a separate disk that can be accessed by all copies of the agent running on different cluster nodes. If the node that hosts the active agent fails and a copy of the agent on the other node is started, shared information such as historical data files can still be read and written to the disk as if the same copy of the agent was still running. The agent is installed separately on each cluster node.

Remember: Using a shared disk is preferred over using a Network File System (NFS) mounted file system because results are unpredictable when the NFS mounted file system is not available.

To install and configure the WebSphere Message Broker Monitoring agent, repeat the procedure in "Configuring the WebSphere Message Broker Monitoring agent" for each instance of the WebSphere Message Broker Monitoring agent in your environment.

Important: You must repeat this procedure for different copies of the same agent instance running on different cluster nodes.

Configuring the WebSphere Message Broker Monitoring agent

Remember: WebSphere Message Broker supportpac IC91 must have been installed when installing WebSphere Message Broker product in the HACMP cluster environment. See your WebSphere Message Broker product documentation for further information.

To install and configure the WebSphere Message Broker Monitoring agent for use in an HACMP cluster environment, perform the following steps:

1. Install the WebSphere Message Broker Monitoring agent on the cluster node on which you want the agent to run. For installation instructions, see *Installation and Setup Guide*.
2. Create new instances of the WebSphere Message Broker Monitoring agent for each message broker.
3. Set local variables in the `qi.ini` configuration file.
4. Create the directories where historical and situation data files are stored.
5. Configure the Tivoli Enterprise Portal Server to list agents running in cluster groups by the cluster name instead of the host name in the Tivoli Enterprise Portal.
6. Create a file containing the startup script used to start the agent.
7. Create a file containing the shutdown script used to stop the agent.
8. Set the scripts that are used to start and stop the agent in HACMP.

Important: You must repeat this procedure for different copies of the same agent instance that are running on different cluster nodes.

The WebSphere Message Broker Monitoring agent is now configured to operate in an HACMP cluster environment.

Creating new instances of the WebSphere Message Broker Monitoring agent for each message broker

Create new instances of the WebSphere Message Broker Monitoring agent for each message broker that you want to monitor by performing the following procedure:

1. Navigate to the *ITM_HOME*/config directory, where *ITM_HOME* is the directory where the IBM Tivoli Monitoring is installed. The default directory is /opt/IBM/ITM.
2. Create a new configuration file for each instance of the WebSphere Message Broker Monitoring agent by copying the default kqi.xml configuration file to the *hostname*_qi_*agent_instance_ID*.xml file, where *hostname* is the host name of cluster node, *agent_instance_ID* is the WebSphere Message Broker Monitoring agent ID.

   Remember: The *agent_instance_ID* is string of up to four characters. It must be different for each agent instance that runs on the same cluster node. However, it must be identical to the value for the agent instance running on the failover cluster node that corresponds to this instance.
3. Edit each of the newly created configuration files as follows:
   a. Add the following attribute directly after the version attribute inside the KqiAgent tag:

      ```
      agentId="agent_instance_ID"
      ```

      where *agent_instance_ID* is the ID of the WebSphere Message Broker Monitoring agent instance. This must be the same as that specified in the filename in the previous step.
   b. Add the following tags directly after the opening KqiAgent tag:

      ```
      <MonitorBroker name="broker_name">
      </MonitorBroker>
      ```

      where *broker_name* is the name of the message broker that you want to monitor.
   c. Set the **discoveryInterval** parameter to a high value, because the agent is only monitoring a single message broker. This setting can reduce the number of unnecessary operations hat are performed to discover brokers. For example, you can set the value to 2592000 seconds (30 days). The resulting file looks similar to the following example:

      ```
      <KqiAgent version="710"
                agentId="MQHA"
                defaultRetainBrokerEvents="10"
                defaultRetainFlowEvents="10"
                retainProductEvents="10"
                discoveryInterval="2592000"
                defaultStatisticInterval="60"
                defaultFlowEventInterval="15"
                defaultHistoricalAccountingType="Archive"
                defaultRetainRecentSnapshotSamples="15"
                defaultRetainRecentArchiveSamples="5"
                defaultRetainRecentPubSubSamples="15"
                holdTimeForQuery="180"
                defaultReplyQueueName="KQI.AGENT.REPLY.QUEUE"
                defaultReplyQueueModel="SYSTEM.BROKER.MODEL.QUEUE"
                defaultCollectNodeData="YES"
                defaultTakeActionAuthUsers="*"                    >
                <MonitorBroker name="HABK">
                </MonitorBroker>
      </KqiAgent>
      ```

Setting local variables in the agent configuration file

Set local variables by performing the following procedure:

1. Navigate to the *ITM_HOME*/config directory, where *ITM_HOME* is the directory where IBM Tivoli Monitoring is installed.
2. Open the qi.ini file.
3. Locate the $CTIRA_SIT_PATH variable and change it to the following code:

   ```
   CTIRA_SIT_PATH=$CANDLEHOME$/$BINARCH$/$PRODUCTCODE$/sit$INSTANCE_HISTORY$
   ```

   Remember: Using the $INSTANCE_HISTORY$ variable in the qi.ini file distinguishes the instances of the WebSphere Message Broker Monitoring agent on the cluster node.
4. Save and close the qi.ini file.

Creating the directories for historical and situation data files

Create the directories where historical and situation data files are stored by performing the following procedure on each cluster node:

1. Create the *ITM_HOME*/aix513/qi/sit directory for storing situation data, where *ITM_HOME* is the directory in which IBM Tivoli Monitoring is installed.

   Remember: The user ID that is used to run the agent must have write access to this directory.
2. Create separate directories for storing historical and situation data from each agent instance on the shared disk as follows, where *disk_name* is the name of the shared disk on which historical data is stored, and *instance_name* is the name of the WebSphere Message Broker Monitoring agent instance.
   - For storing historical data: *disk_name*/kqi/hist/*instance_name*
   - For storing situation data: *disk_name*/kqi/sit/*instance_name*

   Remember: The user ID that is used to run the agent must have write access to these directories.
3. Create a link to the shared disk on which historical data is stored by running the following command, where *disk_name* is the name of the shared disk, *ITM_HOME* is the directory in which IBM Tivoli Monitoring is installed, and *instance_name* is the name of the WebSphere Message Broker Monitoring agent instance.

   ```
   ln –sf disk_name/kqi/hist/instance_name ITM_HOME/aix513/qi/hist
   /instance_name
   ```

   Remember: The symbolic link definition is used to equate the qi.ini specification to a directory on the shared disk on which historical and situation data is stored.
4. Create a link to the shared disk on which situation data is stored by running the following command, where *disk_name* is the name of the shared disk, *ITM_HOME* is the directory in which IBM Tivoli Monitoring is installed, and *instance_name* is the name of the WebSphere Message Broker Monitoring agent instance:

   ```
   ln –sf disk_name/kqi/sit/instance_name ITM_HOME/aix513/qi/sit
   /instance_name
   ```

   Remember: The symbolic link definition is used to equate the qi.ini specification to a directory on the shared disk on which historical and situation data is stored.

Configuring the Tivoli Enterprise Portal to list agents running in the cluster groups

Configure the Tivoli Enterprise Portal Server to list agents that are running in the cluster groups by the cluster name instead of the host name in the Tivoli Enterprise Portal. Perform the following steps:

1. Navigate to the *ITM_HOME*/config directory.
2. Open the qi.ini agent configuration file.
3. Add the following line, CTIRA_HOSTNAME=*cluster_name*, where *cluster_name* is the name of the HACMP cluster that is displayed in Tivoli Enterprise Portal. This value must be identical to the value of the CTIRA_HOSTNAME attribute of the agent running on the failover node.
4. Perform one of the following procedures, depending on which operating system the Tivoli Enterprise Portal Server is running on.
   - If the Tivoli Enterprise Portal Server is running on a Windows system, perform the following procedure:
     a. Stop the Tivoli Enterprise Portal Server if it is running.
     b. In the Manage Tivoli Enterprise Monitoring Services, right-click the Tivoli Enterprise Portal Server, and then click **Advanced** > **Edit Variables** from the displayed menu. The Tivoli Enterprise Portal Server Override Local Variable Setting window is displayed.
     c. Click **Add**. The Add Environment Setting Override window is displayed.
     d. In the **Variable** menu look for the **KFW_TOPOLOGY_CLUSTER_LIST** parameter. If it exists, append AFF_MQSI_AGENT AFF_MQSI_BROKER to any existing values, separated by a space. If it does not already exist, create it and set its value to AFF_MQSI_AGENT AFF_MQSI_BROKER.
     e. Reconfigure and recycle the portal server.
   - If the Tivoli Enterprise Portal Server is running on a UNIX or Linux system, perform the following procedure:
     a. Stop the Tivoli Enterprise Portal Server if it is running.
     b. Navigate to the *ITM_HOME*/config/ directory, where *ITM_HOME* is the IBM Tivoli Monitoring installation directory.
     c. Open the cq.ini file in a text editor.
     d. Look for the **KFW_TOPOLOGY_CLUSTER_LIST** parameter. If it already exists, add AFF_MQSI_AGENT AFF_MQSI_BROKER to any existing values, separated by a space. If it does not already exist, add the following line to the file:
        ```
        KFW_TOPOLOGY_CLUSTER_LIST=AFF_MQSI_AGENT AFF_MQSI_BROKER
        ```
     e. Save and close the file.
     f. Reconfigure and recycle the portal server.

Creating a file that is used to start the agent

Important: When writing a startup script, ensure that the DB2 database is started first, followed by the monitored message broker, and finally the WebSphere Message Broker Monitoring agent.

Create a file containing the startup script that is used to start the agent by performing the following steps:

1. Create a new text file.
2. Enter the following lines to start the DB2 instance, where *db2_instance* is the name of the DB2 database instance.

```
su - db2_instance -c ". db2_instance/sqllib/db2profile; db2start"
```

3. Enter the following lines to start the queue manager that is used by the message broker, where *MC91_install* is the directory in which WebSphere MQ supportpac MC91 is installed, and *QM_name* is the name of the queue manager used by the message broker.

   ```
   MC91_install/bin/hamqm_start QM_name
   ```

4. Enter the following lines to start the monitored message broker, where *IC91_install* is the directory in which WebSphere Message Broker supportpac IC91 is installed, *broker_name* is the name of the monitored message broker

   ```
   IC91_install/hamqsi_start_broker broker_name
   ```

5. Enter the following lines to start the WebSphere Message Broker Monitoring agent, where *ITM_HOME* is the directory in which IBM Tivoli Monitoring is installed, and *instance_name* is the name of the WebSphere Message Broker Monitoring agent instance.

   ```
   ITM_HOME/bin/itmcmd agent -o instance_name start qi
   ```

6. Save the file as `kqi_start.sh`.

Creating a file that is used to stop the agent

Important: When writing a shutdown script, ensure that the WebSphere Message Broker Monitoring agent is stopped first, followed by the monitored message broker, and finally the DB2 database.

Create a file containing the shutdown script used to stop the agent by performing the following steps:

1. Create a new text file.
2. Enter the following lines to stop the WebSphere Message Broker Monitoring agent, where *ITM_HOME* is the directory in which IBM Tivoli Monitoring is installed, and *instance_name* is the name of the WebSphere Message Broker Monitoring agent instance.

   ```
   ITM_HOME/bin/itmcmd agent -o instance_name stop qi
   ```

3. Enter the following lines to stop the monitored message broker, where *IC91_install* is the directory in which WebSphere Message Broker supportpac IC91 is installed, *broker_name* is the name of the monitored message broker

   ```
   IC91_install/bin/hamqsi_stop_broker broker_name 5
   ```

4. Enter the following lines to stop the queue manager that is used by the message broker, where *MC91_install* is the directory in which WebSphere MQ supportpac MC91 is installed, and *QM_name* is the name of the queue manager used by the message broker.

   ```
   MC91_install/bin/hamqm_start QM_name
   ```

5. Enter the following lines to stop the DB2 instance, where *db2_instance* is the name of the DB2 database instance.

   ```
   su - db2_instance -c ". db2_instance/sqllib/db2profile; db2stop"
   ```

6. Save the file as `kqi_stop.sh`.

Setting the scripts that are used to start and stop the agent in an HACMP environment

Perform the following procedure to set the scripts used to start and stop the agent in an HACMP environment:

1. Open the cluster group in the HACMP cluster software.
2. Under Application Server set the start script as `kqi_start.sh`.

3. Under Application Server set the stop script as `kqi_stop.sh`.

Appendix A. Accessibility

Accessibility features help users with physical disabilities, such as restricted mobility or limited vision, to use software products successfully. With the major accessibility features in this product, users can do the following things:

- Use assistive technologies, such as screen-reader software and digital speech synthesizer, to hear what is displayed on the screen. Consult the product documentation of the assistive technology for details on using those technologies with this product.
- Operate specific or equivalent features using only the keyboard.
- Magnify what is displayed on the screen.

In addition, the product documentation was modified to include the following features to aid accessibility:

- All documentation is available in both HTML and convertible PDF formats to give the maximum opportunity for users to apply screen-reader software.
- All images in the documentation are provided with alternative text so that users with vision impairments can understand the contents of the images.

Magnifying what is displayed on the screen

You can enlarge information on the product windows using facilities provided by the operating systems on which the product is run. For example, in a Microsoft Windows system environment, you can lower the resolution of the screen to enlarge the font sizes of the text on the screen. Refer to the documentation provided by your operating system for more information.

Navigating the interface using the keyboard

Standard shortcut and accelerator keys are used by the product and are documented by the operating system. Refer to the documentation provided by your operating system for more information.

Appendix B. Disk space requirements for historical data tables

This appendix provides information about disk space requirements for historical data tables.

Historical data tables

The amount of default space required for a 24-hour period on a monitored system varies greatly depending on customer configuration. The following estimates are taken from an example Windows system with the following WebSphere Message Broker components installed: configuration manager, user name server, Message Brokers Toolkit, and a single message broker. The broker has two monitored execution groups, four message flows being monitored configured with two subflows, 12 CandleMonitor nodes, a total of 50 nodes with 150 terminals, and two threads per message flow. Accounting statistics has been turned on to collect all possible archive data for the four message flows with the default interval in place at 60 minutes, and the default agent parameter setting is in place to only collect archive data for history, not snapshot data.

Important: The historical collection interval must be set to the same value for each of the five statistics tables (broker statistics, execution group statistics, message flow statistics, subflow statistics, and CandleMonitor node statistics). Also, the historical collection interval must be set to the same value for each of the four accounting tables (message flow accounting, thread accounting, node accounting, and terminal accounting). The historical collection interval can be set to be a different value for the two groups of tables (statistics and accounting). The default for all collection is 15 minutes. Do not set up multiple collection intervals for either the statistics group of tables or the accounting group of tables because different settings of historical collection interval lengths can cause the data recorded historically to be inaccurate for these attribute groups.

Table 14. Historical data tables of the WebSphere Messaging Broker Monitoring agent

Attribute history table	File name for historical data	Estimated space required per managed system per 24-hour period
Accounting Message Flow Statistics	kqitacmf	168 kilobytes
Accounting Node Statistics	kqitacnd	2371 kilobytes
Accounting Terminal Statistics	kqitactr	7340 kilobytes
Accounting Thread Statistics	kqitacth	310 kilobytes
Broker Status	kqitbrks	111 kilobytes
Broker Status Events	kqitbsev	28 kilobytes
Components	kqitcomp	330 kilobytes
Execution Group Status	kqitegrs	144 kilobytes
File Resource Statistics	kqitrsfl	147 kilobytes
JDBC Connection Pools Resource Statistics	kqitrsjd	10968 kilobytes

Table 14. Historical data tables of the WebSphere Messaging Broker Monitoring agent (continued)

Attribute history table	File name for historical data	Estimated space required per managed system per 24-hour period
JVM Resource Statistics	kqitrsjv	34956 kilobytes
Message Flow Status	kqitmfls	1935 kilobytes
Message Processing Nodes	kqitmpns	2857 kilobytes
Monitor Node Base Statistics	kqitmnfn	1791 kilobytes
Monitor Node Broker Statistics	kqitmnbr	66 kilobytes
Monitor Node Events	kqitmnev	23 kilobytes
Monitor Node Execution Group Statistics	kqitmneg	179 kilobytes
Monitor Node Message Flow Statistics	kqitmnmf	834 kilobytes
Monitor Node Sub-Flow Statistics	kqitmnsf	255 kilobytes
ODBC Resource Statistics	kqitrsod	8775 kilobytes
Parsers Resource Statistics	kqitrsps	26730 kilobytes
Product Events	kqitprev	29 kilobytes
SOAPInput Resource Statistics	kqitrssp	9956 kilobytes
Total Default Space		107446 kilobytes

Historical table record sizes

The following tables present the record size and frequency for each historical data table.

Table 15. Historical table record sizes of the WebSphere Messaging Broker Monitoring agent

History table	Record size	Frequency
Accounting Message Flow Statistics	1796 bytes	1 row per message flow with accounting feature turned on per accounting interval for archive data and, if selected for history, per 20 seconds for snapshot data
Accounting Node Statistics	2024 bytes	1 row per node per message flow with the accounting feature being turned on per accounting interval for archive data and, if selected for history, per 20 seconds for snapshot data

Table 15. Historical table record sizes of the WebSphere Messaging Broker Monitoring agent (continued)

History table	Record size	Frequency
Accounting Terminal Statistics	2088 bytes	1 row per terminal per node per message flow with the accounting feature being turned on per accounting interval for archive data and, if selected for history, per 20 seconds for snapshot data
Accounting Thread Statistics	1656 bytes	1 row per thread per message flow with the accounting feature being turned on per accounting interval for archive data and, if selected for history, per 20 seconds for snapshot data
Broker Status	1184 bytes	1 row per interval
Broker Status Events	972 bytes	1 row per broker event publication (pure event table, so not affected by interval)
Components	880 bytes	1 row per WebSphere broker component installed on system monitored by agent per interval
Execution Group Status	772 bytes	1 row per execution group per interval
File Resource Statistics	1052 bytes	1 row per execution group per interval (pure event table, so not affected by interval)
JDBC Connection Pools Resource Statistics	1300 bytes	1 row per execution group every 20 seconds (pure event table, so not affected by interval)
JVM Resource Statistics	1036 bytes	4 rows per execution group every 20 seconds (pure event table, so not affected by interval)
Message Flow Status	1720 bytes	1 row per message flow per interval
Message Processing Nodes	2540 bytes	1 row per message processing node per interval
Monitor Node Base Statistics	1592 bytes	1 row per CandleMonitor node per interval
Monitor Node Broker Statistics	704 bytes	1 row per interval
Monitor Node Events	1604 bytes	1 row per message flow event detected (pure event table, so not affected by interval)
Monitor Node Execution Group Statistics	956 bytes	1 row per monitored execution group per interval

Table 15. Historical table record sizes of the WebSphere Messaging Broker Monitoring agent (continued)

History table	Record size	Frequency
Monitor Node Message Flow Statistics	1112 bytes	1 row per monitored message flow per interval
Monitor Node Sub-Flow Statistics	1360 bytes	1 row per monitored subflow per interval
ODBC Resource Statistics	1040 bytes	1 row per execution group every 20 seconds (pure event table, so not affected by interval)
Parsers Resource Statistics	1056 bytes	3 rows or more per execution group every 20 seconds (pure event table, so not affected by interval)
Product Events	1204 bytes	1 row per product monitoring event noted by agent (pure event table, so not affected by interval)
SOAPInput Resource Statistics	1180 bytes	1 row per execution group every 20 seconds (pure event table, so not affected by interval)

Historical space requirement worksheets

Use the following worksheets to estimate expected file sizes and additional disk space requirements for your site. A sample calculation is provided for each historical data collection table.

Table 16. Components (kqitcomp) worksheet

Interval	Record size	Formula	Expected file size per 24-hour period
15 min.	880 bytes	(60/15 x 24 x 880 x 4) / 1024 for 4 installed components	330 kilobytes

Table 17. Product Events (kqitprev) worksheet

Interval	Record size	Formula	Expected file size per 24-hour period
N/A	1204 bytes	(1204 x 25) / 1024 for 25 product monitoring events occurring	29 kilobytes

Table 18. Accounting Message Flow Statistics (kqitacmf) worksheet

Interval	Record size	Formula	Expected file size per 24-hour period
1 hour *	1796 bytes	(60/60 x 24 x 1796 x 4)/1024 for 4 monitored message flows	168 kilobytes

Remember: * This is the default interval; even if you set the history interval to less, the data can only be produced as often as the default interval occurs.

Table 19. Accounting Node Statistics (kqitacnd) worksheet

Interval	Record size	Formula	Expected file size per 24-hour period
1 hour *	2024 bytes	(60/60 x 24 x 2024 x 50)/1024 for a total of 50 nodes in monitored message flows	2371 kilobytes

Remember: * This is the default interval; even if you set the history interval to less, the data can only be produced as often as the default interval occurs.

Table 20. Accounting Terminal Statistics (kqitactr) worksheet

Interval	Record size	Formula	Expected file size per 24-hour period
1 hour *	2088 bytes	(60/60 x 24 x 2088 x 150)/1024 for a total of 150 terminals in monitored message flows	7340 kilobytes

Remember: * This is the default interval; even if you set the history interval to less, the data can only be produced as often as the default interval occurs.

Table 21. Accounting Thread Statistics (kqitacth) worksheet

Interval	Record size	Formula	Expected file size per 24-hour period
1 hour *	1656 bytes	(60/60 x 24 x 1656 x 8)/1024 for 4 monitored message flows with 2 threads each	310 kilobytes

Table 21. Accounting Thread Statistics (kqitacth) worksheet (continued)

Interval	Record size	Formula	Expected file size per 24-hour period

Remember: * This is the default interval; even if you set the history interval to less, the data can only be produced as often as the default interval occurs.

Table 22. Broker Status (kqitbrks) worksheet

Interval	Record size	Formula	Expected file size per 24-hour period
5 min.	1184 bytes	(60/15 x 24 x 1184 x 1)/1024 for 1 broker	111 kilobytes

Table 23. Broker Status Events (kqitbsev) worksheet

Interval	Record size	Formula	Expected file size per 24-hour period
N/A	972 bytes	(972 x 30)/1024 for 30 broker events occurring	28 kilobytes

Table 24. Execution Group Status (kqitegrs) worksheet

Interval	Record size	Formula	Expected file size per 24-hour period
15 min.	772 bytes	(60/15 x 24 x 772 x 2)/1024 for 2 execution groups	144 kilobytes

Table 25. File Resource Statistics (kqitrsfl) worksheet

Interval	Record size	Formula	Expected file size per 24-hour period
20 sec.	1052 bytes	(60/20 x 60 x 24 x 1052 x 1 x 2)/1024 for 2 execution groups	147 kilobytes

Table 26. JDBC Connection Pools Resource Statistics (kqitrsjd) worksheet

Interval	Record size	Formula	Expected file size per 24-hour period
20 sec.	1300 bytes	(60/20 x 60 x 24 x 1300 x 1 x 2)/1024 for 2 execution groups	10968 kilobytes

Table 27. JVM Resource Statistics (kqitrsjv) worksheet

Interval	Record size	Formula	Expected file size per 24-hour period
20 sec.	1036 bytes	(60/20 x 60 x 24 x 1036 x 4 x 2)/1024 for 2 execution groups	34956 kilobytes

Table 28. Message Flow Status (kqitmfls) worksheet

Interval	Record size	Formula	Expected file size per 24-hour period
15 min.	1720 bytes	(60/15 x 24 x 1720 x 12)/1024 for 12 message flows	1935 kilobytes

Table 29. Message Processing Nodes (kqitmpns) worksheet

Interval	Record size	Formula	Expected file size per 24-hour period
15 min.	2540 bytes	(60/15 x 24 x 2540 x 12)/1024 for 12 message flows	2857 kilobytes

Table 30. Monitor Node Base Statistics (kqitmnfn) worksheet

Interval	Record size	Formula	Expected file size per 24-hour period
15 min.	1592 bytes	(60/15 x 24 x 1592 x 12)/1024 for 12 CandleMonitor nodes in flows	1791 kilobytes

Table 31. Monitor Node Broker Statistics (kqitmnbr) worksheet

Interval	Record size	Formula	Expected file size per 24-hour period
15 min.	704 bytes	(60/15 x 24 x 704 x 1)/1024 for 1 broker	66 kilobytes

Table 32. Monitor Node Events (kqitmnev) worksheet

Interval	Record size	Formula	Expected file size per 24-hour period
N/A	1604 bytes	(1604 x 15)/1024 for 15 message flow events occurring	23 kilobytes

Table 33. Monitor Node Execution Group Statistics (kqitmneg) worksheet

Interval	Record size	Formula	Expected file size per 24-hour period
15 min.	956 bytes	(60/15 x 24 x 956 x 2)/1024 for 2 monitored execution groups	179 kilobytes

Table 34. Monitor Node Message Flow Statistics (kqitmnmf) worksheet

Interval	Record size	Formula	Expected file size per 24-hour period
15 min.	1112 bytes	(60/15 x 24 x 1112 x 8)/1024 for 4 monitored message flows	834 kilobytes

Table 35. Monitor Node Sub-Flow Statistics (kqitmnsf) worksheet

Interval	Record size	Formula	Expected file size per 24-hour period
15 min.	1360 bytes	(60/15 x 24 x 1360 x 2)/1024 for 2 monitored sub-flows	255 kilobytes

Table 36. ODBC Resource Statistics (kqitrsod) worksheet

Interval	Record size	Formula	Expected file size per 24-hour period
20 sec.	1040 bytes	(60/20 x 60 x 24 x 1040 x 1 x 2)/1024 for 2 execution groups	8775 kilobytes

Table 37. Parsers Resource Statistics (kqitrsps) worksheet

Interval	Record size	Formula	Expected file size per 24-hour period
20 sec.	1056 bytes	(60/20 x 60 x 24 x 1056 x 3 x 2)/1024 for 2 execution groups	26730 kilobytes

Table 38. SOAPInput Resource Statistics (kqitrssp) worksheet

Interval	Record size	Formula	Expected file size per 24-hour period
20 sec.	1180 bytes	(60/20 x 60 x 24 x 1180 x 1 x 2)/1024 for 2 execution groups	9956 kilobytes

Remember: * This is the default interval; even if you set the history interval to less, the data can only be produced as often as the default interval occurs.

In the worksheet examples, the minimum collection interval unit of 15 minutes is used. You can create a summary table that provides a representative disk storage space figure for all of the history files and archived files for a one-week time period, if all collection is done at the remote agent managed system. To do so, multiply the expected file size per 24 hours total times seven. Note that historical collection cannot be turned on for those tables not collected by default. If historical data is desired for those tables, a much longer collection interval than the default 15 minutes is required because the data is not expected to change often. You must spread the disk space requirements among the systems where data collection is performed.

Historical disk space summary worksheet

The following tables are disk space summary worksheets for the WebSphere Message Broker Monitoring agent.

Table 39. Disk space summary worksheet for historical tables

History table	Historical data table size (kilobytes) (24 hours)	Number of archives	Subtotal space required (kilobytes)
Accounting Message Flow Statistics			
Accounting Node Statistics			
Accounting Terminal Statistics			
Accounting Thread Statistics			
Broker Status			
Broker Status Events			
Components			
Execution Group Status			
File Resource Statistics			
JDBC Connection Pools Resource Statistics			
JVM Resource Statistics			
Message Flow Status			
Message Processing Nodes			
Monitor Node Base Statistics			
Monitor Node Broker Statistics			
Monitor Node Events			
Monitor Node Execution Group Statistics			
Monitor Node Message Flow Statistics			
Monitor Node Sub-Flow Statistics			
ODBC Resource Statistics			
Parsers Resource Statistics			
Product Events			

Table 39. Disk space summary worksheet for historical tables (continued)

History table	Historical data table size (kilobytes) (24 hours)	Number of archives	Subtotal space required (kilobytes)
SOAPInput Resource Statistics			
Total disk space required			

Appendix C. Language codes

Table 40 lists the languages supported by the WebSphere Message Broker Monitoring agent, and their corresponding language codes.

Table 40. Language codes for the supported languages

Language	**code**
English	en_US
German	de_DE
Spanish	es_ES
French	fr_FR
Italian	it_IT
Japanese	ja_JP
Korean	ko_KR
Portuguese (Brazilian)	pt_BR
Simplified Chinese	zh_CN
Traditional Chinese	zh_TW

Appendix D. Architecture codes

Abbreviations are used in IBM Tivoli software to represent the various operating system architectures. The following table shows the most current listing of these abbreviations.

This information can also be found in the following file on UNIX systems: *install_dir*/registry/archdsc.tbl.

Table 41. Operating system architecture abbreviations

Abbreviation	Operating system architecture
aix513	AIX v5.1 (32 bit)
aix516	AIX v5.1 (64 bit)
aix523	AIX v5.2 (32 bit)
aix526	AIX v5.2 (64 bit)
aix533	AIX v5.3 (32 bit)
aix536	AIX v5.3 (64 bit)
citrix	Citrix Metaframe
hp10	HP-UX v10.01/10.10
hp102	HP-UX v10.20
hp11	HP-UX v11
hp116	HP-UX v11 (64 bit)
li622	Linux Intel v2.2
li6223	Linux Intel v2.2 (32 bit)
li624	Linux Intel v2.4
li6242	Linux Intel v2.4 GCC 2.9.5 (32 bit)
li6243	Linux Intel v2.4 (32 bit)
li6245	Linux Intel v2.4 GCC 2.9.5 (64 bit)
li6246	Linux Intel v2.4 (64 bit)
li6262	Linux Intel v2.6 GCC 2.9.5 (32 bit)
li6263	Linux Intel v2.6 (32 bit)
li6265	Linux Intel v2.6 GCC 2.9.5 (64 bit)
li6266	Linux Intel v2.6 (64 bit)
ls322	Linux zSeries, 2.2 kernel
ls3223	Linux zSeries, v2.2 (32 bit)
ls3226	Linux zSeries, v2.2 (64 bit)
ls324	Linux zSeries, v2.4
ls3243	Linux zSeries, v2.4 (32 bit)
ls3246	Linux zSeries, v2.4 (64 bit)
ls3262	Linux S390 v2.6 GCC 2.9.5 (32 bit)
ls3263	Linux S390 v2.6 (32 bit)
ls3265	Linux S390 v2.6 GCC 2.9.5 (64 bit)

Table 41. Operating system architecture abbreviations (continued)

Abbreviation	Operating system architecture
ls3266	Linux S390 v2.6 (64 bit)
osf1	Digital UNIX (before V5.0)
os390	OS/390® or z/OS systems
os400	OS/400®
sol24	Solaris v2.4
sol25	Solaris v2.5
sol26	Solaris v2.6
sol273	Solaris v7 (32 bit)
sol276	Solaris v7 (64 bit)
sol283	Solaris v8 (32 bit)
sol286	Solaris v8 (64 bit)
sol293	Solaris v9 (32 bit)
sol296	Solaris v9 (64 bit)
sol503	Solaris v10 (32 bit)
sol506	Solaris v10 (64 bit)
sol603	Solaris v10 Opteron (32 bit)
sol606	Solaris v10 Opteron (64 bit)
tsf50	Tru64 v5.0
unix	UNIX system
winnt	Windows 2000 and Windows 2003 Server

Appendix E. Library for WebSphere Message Broker Monitoring agent

The following documents are available in the library for WebSphere Message Broker Monitoring agent:

- *IBM Tivoli Composite Application Manager Agents for WebSphere Messaging: Installation and Setup Guide*

  Describes how to install WebSphere MQ Monitoring agent, WebSphere MQ Configuration agent, and WebSphere Message Broker Monitoring agent on Windows, UNIX, Linux, and i5/OS™ systems.
- *IBM Tivoli Composite Application Manager Agents for WebSphere Messaging: Upgrade and Migration Guide*

  Provides information about how to upgrade or migrate from previous versions of WebSphere MQ Monitoring agent, WebSphere MQ Configuration agent, and WebSphere Message Broker Monitoring agent to version 7.3.
- *IBM Tivoli Composite Application Manager Agent for WebSphere Message Broker User's Guide*

  Provides instructions for using the features of the WebSphere Message Broker Monitoring agent.
- *IBM Tivoli Composite Application Manager Agents for WebSphere Messaging: Troubleshooting Guide*

  Provides problem determination and resolution information for the issues most commonly encountered when using WebSphere MQ Monitoring agent, WebSphere MQ Configuration agent, and WebSphere Message Broker Monitoring agent.

Notices

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

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

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

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

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

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

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement might not apply to you.

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

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

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

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

IBM Corporation
2Z4A/101
11400 Burnet Road
Austin, TX 78758 U.S.A.

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

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

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

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

Privacy policy considerations

IBM Software products, including software as a service solutions, ("Software Offerings") may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering's use of cookies is set forth below.

Depending upon the configurations deployed, this Software Offering may use session cookies that collect each user's user name for purposes of session management, authentication, and single sign-on configuration. These cookies cannot be disabled.

If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, See IBM's Privacy Policy at http://www.ibm.com/privacy and IBM's Online Privacy Statement at http://www.ibm.com/privacy/details the section entitled "Cookies, Web Beacons and Other Technologies" and the "IBM Software Products and Software-as-a-Service Privacy Statement" at http://www.ibm.com/software/info/product-privacy.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml.

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

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

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

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

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

Glossary

This glossary includes terms and definitions for ITCAM Agents for WebSphere Messaging.

The following cross-references are used in this glossary:

- See refers you from a term to a preferred synonym, or from an acronym or abbreviation to the defined full form.
- See also refers you to a related or contrasting term.

To view glossaries for other IBM products, go to www.ibm.com/software/globalization/terminology (opens in new window).

A

access The ability to read, update, or otherwise use a resource. Access to protected resources is usually controlled by system software.

access management
The process of controlling access to IT services, data, or other assets.

address space
The range of addresses available to a computer program or process. Address space can refer to physical storage, virtual storage, or both. See also buffer pool.

agent Software that is installed to monitor systems. An agent collects data about an operating system, a subsystem, or an application.

aggregation
The process of collecting, interpreting, and sorting data from various locations into a single file.

alert A message or other indication that signals an event or an impending event. See also event.

attribute

1. The application properties that are measured and reported on, such as the amount of memory that is used or a message ID. See also attribute group.
2. Data that is associated with a component. For example, a host name, IP address, or the number of hard drives can be attributes associated with a server component.

attribute group
A set of related attributes that can be combined in a view or a situation. See also attribute, situation, view.

audit A process that logs modifications to the database and plan.

B

batch

1. Pertaining to a group of jobs to be run on a computer sequentially with the same program with little or no operator action.
2. A group of records or data processing jobs brought together for processing or transmission.

batch job
A predefined group of processing actions submitted to the system to be performed with little or no interaction between the user and the system.

batch mode
The condition established so that batch processing can be performed.

BPM See business performance management.

broker
A set of execution processes that host one or more message flows. See also execution group, message flow.

buffer pool
An area of memory into which data pages are read and in which they are modified and held during processing. See also address space.

bundle
A packaged collection of software products that is purchased as one item and that has its own product identifier (PID).

business performance management (BPM)
The monitoring, management, and tuning

of business performance in real time through the analysis of business relevant information.

C

channel
A WebSphere MQ object that defines a communication link between two queue managers (message channel) or between a client and a queue manager (MQI channel). See also queue manager.

client A software program or computer that requests services from a server. See also host, server.

cluster
1. In WebSphere MQ, a group of two or more queue managers on one or more computers, providing automatic interconnection, and allowing queues to be advertised among them for load balancing and redundancy.
2. In Microsoft Cluster Server, a group of computers, connected together and configured in such a way that, if one fails, MSCS performs a failover, transferring the state data of applications from the failing computer to another computer in the cluster and reinitiating their operation there.

cluster queue manager
A queue manager that is a member of a cluster. A queue manager can be a member of more than one cluster.

component
A software item that is part of a software product, and might be separately identified, but is not individually licensed.

condition
1. An expression that consists of an agent attribute, an operator such as great than or equal to, and a value. It can be read as, "If - system condition - compared to - value - is true. See also situation.
2. A test of a situation or state that must be in place for a specific action to occur.

configuration
The manner in which the hardware and software of a system, subsystem, or network are organized and interconnected.

D

data set
The major unit of data storage and retrieval, consisting of a collection of data in one of several prescribed arrangements and described by control information to which the system has access.

dead-letter queue (DLQ)
A queue to which a queue manager or application sends messages that cannot be delivered to their correct destination.

deployment
The process of installing and configuring a software application and all its components.

DLQ See dead-letter queue.

dynamic queue
A local queue created when a program opens a model queue object.

E

enterprise
The composite of all operational entities, functions, and resources that form the total business concern and that require an information system.

event An occurrence of significance to a task or system. Events can include completion or failure of an operation, a user action, or the change in state of a process. See also alert, situation.

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

F

full repository
A complete set of information about every queue manager in a cluster. This set of information is called the repository or sometimes the full repository and is usually held by two of the queue managers in the cluster. See also partial repository.

function
Any instruction or set of related instructions that performs a specific operation.

H

host A computer that is connected to a network and that provides an access point to that network. The host can be a client, a server, or both a client and server simultaneously. See also client, server.

hot standby
A redundant server that, if the primary server or hub server fails, assumes the responsibilities of the failed server.

I

integration
The software development activity in which separate software components are combined into an executable whole.

L

launch-in-context
An operation in which a user starts a secondary application from a primary application to perform a specific task. Using the parameters, navigation instructions, and user credentials that are supplied by the primary application, the secondary application opens to the specific place in which to complete the task.

M

managed object
A resource that is subject to management as viewed from a systems management perspective. Examples of such resources are a connection, a scalable system, or a line.

managed system
A system that is being controlled by a given system management application.

manager
An entity that monitors or controls one or more managed objects by (a) receiving notifications regarding the objects and (b) requesting management operations to modify or query the objects.

message flow
A sequence of processing steps that execute in the broker when an input message is received. Message flows are defined in the workbench by including a number of message flow nodes, each of which represents a set of actions that define a processing step. The connections in the flow determine which processing steps are carried out, in which order, and under which conditions. See also broker, execution group, subflow.

middleware
Software that acts as an intermediate layer between applications or between client and server. It is used most often to support complex, distributed applications in heterogeneous environments.

module
A program unit that is discrete and identifiable with respect to compiling, combining with other units, and loading.

monitoring agent
See agent.

multi-instance queue manager
A queue manager that is configured to share the use of queue manager data with other queue manager instances. One instance of a running multi-instance queue manager is active, other instances are on standby ready to take over from the active instance. See also queue manager.

O

offering

1. A logical unit of software packaging and sharing that has a managed development and maintenance life cycle and customer visible attributes (offering features, product IDs, licenses, maintenance contracts, and so forth). An offering is a serviceable software asset that is orderable by an IBM customer. It can be a collection of common components, assemblies, and other offerings.
2. The element or integrated set of elements (hardware, software, services) designed to satisfy the wants and needs of current and/or prospective customers. A solution is the application of the offering in a specific customer environment. See also solution.

P

partial repository
A partial set of information about queue managers in a cluster. A partial repository is maintained by all cluster queue managers that do not host a full repository. See also full repository.

performance management

1. The discipline that encompasses capacity planning, collecting performance data, and tuning resources.
2. The management processes and systems needed to effectively deliver business services.

PID See product identifier.

platform
The combination of an operating system and hardware that makes up the operating environment in which a program runs.

policy A set of considerations that influence the behavior of a managed resource or a user.

product ID
See product identifier.

product identifier (PID, product ID)
A unique value that identifies an IBM software product. Every mainframe and distributed IBM software product has a PID.

Q

query In a Tivoli environment, a combination of statements that are used to search the configuration repository for systems that meet certain criteria. The query object is created within a query library.

queue An object that holds messages for message-queueing applications. A queue is owned and maintained by a queue manager.

queue manager
A component of a message queuing system that provides queuing services to applications. See also channel, multi-instance queue manager.

queue-sharing group
In WebSphere MQ for z/OS, a group of queue managers in the same sysplex that can access a single set of object definitions stored in the shared repository, and a single set of shared queues stored in the coupling facility.

R

registry
A repository that contains access and configuration information for users, systems, and software.

S

sampled event
An event that happens when a situation becomes true. Situations sample data at regular intervals. When the situation is true, it opens an event, which is closed automatically when the situation returns to false.

segment
A set of customers/buyers within a market who have common wants, needs, characteristics and buying behavior. These wants and needs are sufficiently homogeneous that a consistent set of strategies, marketing campaigns and sales tactics can be directed toward them.

server A software program or a computer that

provides services to other software programs or other computers. See also client, host.

service request
A request from a user for help, information, advice, or access to an IT service.

severity level
A classification for an event that indicates its degree of severity. The predefined severity levels, in order of descending severity, are: fatal, critical, warning, minor, harmless, and unknown.

situation
A set of conditions that, when met, creates an event. See also attribute group, condition, event.

snapshot
A capture of data at a point time for performance analysis.

solution
A combination of products that addresses a particular customer problem or project.

started task
In MVS, a process that begins at system start and runs unattended. Started tasks are generally used for critical applications. The UNIX equivalent of a started task is a daemon.

state An indication associated with an icon, color, and severity level assigned to a situation at a point in time. A situation can reflect one of the following states: critical, warning, or informational.

status The true or false condition of a situation.

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

subnet
See subnetwork.

subnetwork (subnet)
A network that is divided into smaller independent subgroups, which still are interconnected.

subscription
In a Tivoli environment, the process of identifying the subscribers that the profiles are distributed to.

summarization
The process of aggregating events and then submitting the set of events with a much smaller number of summary events.

system
A computer and its associated devices and programs.

T

TCP/IP
See Transmission Control Protocol/Internet Protocol.

threshold
A customizable value for defining the acceptable tolerance limits (maximum, minimum, or reference limit) for an application resource or system resource. When the measured value of the resource is greater than the maximum value, less than the minimum value, or equal to the reference value, an exception or event is raised.

transaction
A unit of processing consisting of one or more application programs, affecting one or more objects, that is initiated by a single request.

Transmission Control Protocol/Internet Protocol (TCP/IP)
An industry-standard, nonproprietary set of communication protocols that provides reliable end-to-end connections between applications over interconnected networks of different types.

transmission queue
A local queue on which prepared messages destined for a remote queue manager are temporarily stored.

U

upgrade
To install a new version or release of a product to replace an earlier version or release of the same product.

user profile
A description of a user that includes such information as user ID, user name, password, access authority, and other attributes that are obtained when the user logs on.

V

view A window pane, or frame, in a workspace. It may contain data from an agent in a chart or table, or it may contain a terminal session or notepad, for example. A view can be split into two separate, autonomous views. See also attribute group.

W

workspace

1. A window comprised of one or more views.
2. In Tivoli management applications, the working area of the user interface, excluding the Navigator pane, that displays one or more views pertaining to a particular activity. Predefined workspaces are provided with each Tivoli application, and systems administrators can create customized workspaces.

Index

D

E

F

G

H

I

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Printed in USA

SC14-7524-01