SeeBeyond ICAN Suite

e*Gate Integrator Intelligent Queue Services Reference Guide

Release 5.0.5 for Schema Run-time Environment (SRE)

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This chapter introduces you to this guide, its general purpose and scope, and its organization. It also provides sources of related documentation and information.

This guide explains how to use SeeBeyond Technology Corporation™ (SeeBeyond™) Intelligent Queues (IQs) and IQ Services. It presents the following information:

- Overview of each available IQ
- How to configure each specific IQ
- How to implement and maintain IQs in a production environment with the e*Gate Integrator system

Important: *Any operation explanations given here are generic, for reference purposes only, and do not necessarily address the specifics of setting up and/or operating individual e*Gate systems.*

The reader of this guide is presumed to be a developer or system administrator with responsibility for maintaining the e*Gate system; to have expert knowledge of Windows and UNIX operations and administration; and to be thoroughly familiar with Windows-style GUI operations. If implementing one of the database IQs, you also need to be an experienced system administrator for the specific DBMS for that IQ.

Note: *Readers of this guide also need to be familiar with the **e*Gate Integrator System Administration and Operations Guide**.*

1.1 Contents of This Guide

This guide provides the following information:

- **Chapter 2, "IQ Overview"** describes the general features of IQs.
- **Chapter 3, "Installing Database IQs"** describes how to install IQs.
- **Chapter 4, "Configuring the SeeBeyond Standard IQ"** describes how to configure SeeBeyond Standard IQs.
- **Chapter 5, "Configuring the SeeBeyond Memory Loopback IQ"** describes how to configure SeeBeyond Memory Loopback IQs.
- **Chapter 6, "Configuring the ODBC IQ" d**escribes how to configure ODBC IQs.

- **Chapter 7, "Configuring the Oracle IQ"** describes how to configure Oracle IQs.
- **Chapter 8, "Configuring the Sybase IQ"** describes how to configure the Sybase IQs.
- **Chapter 9, "Configuring the IBM MQSeries IQ"** describes how to configure IBM MQSeries IQs.
- **Chapter 10, "Administering IQs"** describes how to use the IQ Administrator application to monitor IQs and view/edit the Events they contain.
- **Appendix A, "Oracle Database Schema"** describes the Oracle schema for Oracle IQs.

1.2 SeeBeyond Web Site

The SeeBeyond Web site is your best source for up-to-the-minute product news and technical support information at **www.seebeyond.com**.

Chapter 2

IQ Overview

This chapter provides a summary of how *Intelligent Queues (IQs)* operate in the e*Gate system, explains the different IQ types, and describes basic IQ properties.

2.1 Intelligent Queues

SeeBeyond IQ services and associated IQs provide a high-performance, non-volatile means for storing data inside the e*Gate Integrator environment. SeeBeyond IQ Services provide the mechanism for moving Events between IQs and handling the low-level implementation of data exchange (such as system calls to initialize or reorganize a database).

SeeBeyond provides a file-based IQ service, a Java™ Message Service (JMS™) IQ service, a memory-loopback IQ service, and specialized IQ Services that work with database management systems (DBMSs) to provide additional queuing and data storage functionality.

Note: *These specialized IQ Services are referred to throughout this guide as the* ***database IQs****.*

The following types of IQ Services are currently available:

SeeBeyond IQ Services

- SeeBeyond Standard IQ
- SeeBeyond Memory Loopback IQ
- SeeBeyond JMS IQ

SeeBeyond Database IQ Services

- ODBC IQ (for Microsoft SQL Server 7.0 and 2000)
- Oracle IQ (for Oracle 8.1.6 and 8.1.7)
- Sybase IQ (for Sybase OpenClient 11.1.1 and 12.0 and Sybase Server 11.9 and 12.0)
- IBM MQSeries IQ (for IBM MQSeries 5.1 and 5.2, server and client modes).

2.1.1 SeeBeyond IQs

The SeeBeyond Standard IQ, SeeBeyond Memory Loopback IQ, and SeeBeyond JMS IQ are provided when you purchase the e*Gate system. The SeeBeyond Standard IQ uses a file-based queuing mechanism with memory mapping of files to ensure high performance. The SeeBeyond Memory Loopback IQ is an in-process, memory-based mechanism to support an individual e*Way or Business Object Broker (BOB) module in a standalone manner. It does not span multiple modules and does not provide Event persistence: If the IQ is shut down during processing, data may be lost. The SeeBeyond JMS IQ is an implementation of Java Message Service (JMS). For details, see the *SeeBeyond JMS Intelligent Queue User's Guide*.

2.1.2 Database IQs

The database IQs—ODBC IQ, Oracle IQ, Sybase IQ, and IBM MQSeries IQ—are purchased separately. Each database IQ provides Event persistence by using a DBMS to store IQ Events. In addition, the ODBC, Oracle, and Sybase IQs allow SQL querying of the queue data, and also provide scheduled Event journaling.

The IBM MQSeries IQ is available in both server and client modes. In server mode, e*Gate components communicate directly with the IBM MQSeries server. In client mode, e*Gate components communicate with the server via the IBM MQSeries client.

The database overhead of the database IQs could result in slower system performance as compared to that of the standard file-based IQ service. Using a database for persistent message queuing, however, provides the robustness and fault tolerance of the DBMS, including transactional commits of database changes. In case of a system crash, IQ recovery via automatic rollback of IQ operations occurs.

2.2 System Requirements

The amount of disk space required to queue the data depends on the type and size of the data being processed, as well as any external applications performing the processing.

2.2.1 Supported Operating Systems

The SeeBeyond IQs run under all operating systems supported by the e*Gate system. For information on these operating systems, see the *e*Gate Integrator Installation Guide*.

2.2.2 Disk Space Quota Limitations

In UNIX systems, do not set any disk-quota limitations on the user that you use to install or run the e*Gate system. e*Gate calculates available disk space in terms of total disk space available on the system, and does no quota checking. If you impose a disk-quota restriction on the user, you risk losing data when IQ-storage demands exceed the user's quota.

2.2.3 External System Requirements for the Database IQs

Following are the external system requirements for each database IQ:

ODBC IQ

- Microsoft SQL Server 7.0 or 2000 server and client
- Merant ODBC 4.0 driver

Windows systems:

- The SQL Server client must be installed on all Participating Hosts that use the ODBC IQ.
- You must set up an ODBC connection using the SQL Server driver on all Participating Hosts that use the ODBC IQ. See **"Creating the ODBC Connection (Windows)" on page 42**.

UNIX systems:

- You must create the **.odbc.ini** file and set the necessary environment variables for the ODBC driver. See **"Creating the ODBC Connection (UNIX)" on page 44**.

Oracle IQ

- Oracle 8.1.6 or Oracle 8.1.7, server and client

  The Oracle client must be installed on all Participating Hosts that use the Oracle IQ. All environment variables and client interfaces should be configured to ensure a connection with the Oracle server.

Sybase IQ

- Sybase OpenClient 11.1.1 or 12.0
- Sybase Server 11.9 or 12.0

In addition, the following patches are required for the Sybase OpenClient Client Library:

- Solaris: Patch 8376
- AIX: Patch 8745
- HP-UX 11: Patch 8747

IBM MQSeries IQ

- IBM MQSeries 5.1 or 5.2 server

  In server mode, the IBM MQSeries 5.1 server package must be installed on all Participating Hosts that use the MQSeries IQ.

- IBM MQSeries 5.1 or 5.2 client (client mode only)

  In client mode, the IBM MQSeries 5.1 or 5.2 client package must be installed on all Participating Hosts that use the IBM MQSeries IQ. The server package must be installed on the machine where the IBM MQSeries queue manager will run (it does not have to be the same machine as the e*Gate Participating Host).

- IBM MQSeries Publish/Subscribe SupportPac

The IBM MQSeries Publish/Subscribe SupportPac must be installed with the server package. You can find instruction on where to go to download the SupportPac from IBM's Web site at **www.ibm.com**.

Chapter 3

Installing Database IQs

This chapter describes how to install the database IQs, and includes a list of installed files for each IQ. The SeeBeyond Standard IQ, SeeBeyond Memory Loopback IQ, and SeeBeyond JMS IQ are installed automatically when you install the e*Gate Integrator Registry Host.

Note: *For information on how to install the Registry Host, see the* ***e*Gate Integrator Installation Guide****.*

3.1 Installing on Windows

To install an IQ on Windows

1. Exit all Windows programs before running the setup program, including any anti-virus applications.
2. Log in on the workstation on which you want to install the IQ. You must have Administrator privileges to install an IQ.
3. Insert the installation CD-ROM into the CD-ROM drive.
4. If the CD-ROM drive's "Autorun" feature is enabled, the setup application should launch automatically. Otherwise, use Windows Explorer or the Control Panel's **Add/Remove Applications** feature to launch the file **setup.exe** on the CD-ROM drive.
5. The InstallShield setup application will launch. Follow the on-screen instructions until you come to the Select Components screen.
6. Check the box labelled **Add-ons**. Then, follow the on-screen instructions to install the IQ.

Note: *Be sure to install the IQ files in the suggested "client" installation directory. The installation utility detects and suggests the appropriate installation directory.* ***Unless you are directed to do so by SeeBeyond support personnel, do not change the suggested "installation directory" setting.***

3.2 Installing an IQ on UNIX

The procedure below describes how you install an IQ on a Unix system.

To install an IQ on UNIX

1. Log in on the workstation containing the CD-ROM drive. Use the user name for the user who owns the IQ files. This user must has privileges to create files in the e*Gate directory tree; root privileges are not necessary.
2. Insert the installation CD-ROM into the drive.
3. If necessary, mount the CD-ROM drive.
4. At the shell prompt, type:

   cd /cdrom/setup
5. Start the installation script by typing:

   setup.sh

   A menu of options appears.
6. Select **e*Gate Addon Applications**. Then, follow the on-screen instructions to install the IQ.

Note: *Install the IQ files in the suggested **client** installation directory. The installation utility detects and suggests the appropriate installation directory.*

Important: *Unless you are directed to do so by SeeBeyond support personnel, do not change the suggested **installation directory** setting.*

3.3 Installation Files/Directories

The IQ installation process installs the following files within the e*Gate directory tree. All files are committed to the **default** schema on the Registry Host. The database IQ files are also installed within the *<eGate>***\client** tree on the Participating Host.

Table 1 Installation Files and Directories

IQ	e*Gate Directory	File(s)
SeeBeyond JMS	\iqservices\	stc_iqms.dll
SeeBeyond Standard IQ	\iqservices\	stc_iqstandard.dll
SeeBeyond Memory Loopback IQ	\iqservices\	stc_iqloopback.dll
ODBC IQ	\iqservices	stc_iqodbc.dll
Oracle IQ	\iqservices\	stc_iqoracle7.dll stc_iqoracle8.dll stc_iqoracle8i.dll

Table 1 Installation Files and Directories (Continued)

IQ	e*Gate Directory	File(s)
Sybase IQ	\iqservices\	stc_iqsybase11.dll
	\bin\	stciqsbadmin.exe
IBM MQSeries IQ	\iqservices\	stc_iqmqm.dll stc_iqmqc.dll

3.4 Configuring IQs

After you install an IQ, you must create at least one IQ component in the e*Gate Schema Designer, configure it, and incorporate it into a schema. You configure an IQ by modifying the IQ component's properties in the **IQ Properties** dialog box. You must configure each IQ component in your schema separately in order for the IQs to function correctly.

For configuration information related to a specific IQ, see the appropriate chapter in this guide. For general information on creating IQ components, see the *e*Gate Integrator User's Guide*.

Chapter 4

Configuring the SeeBeyond Standard IQ

This chapter describes how to plan and configure the SeeBeyond Standard IQ.

4.1 Defining the IQ

The first step after you create an IQ component is to define the IQ Service the IQ will use. This identifies the type of IQ.

Note: *In order to optimize performance you should design your schema with no more than three SeeBeyond Standard IQ components per IQ Manager.*

To define the IQ Service for the SeeBeyond Standard IQ

1. In Schema Designer, in the Navigator's **Components** tab, select the Participating Host and Control Broker of the IQ Manager whose IQs you will be configuring.
2. In the Component Editor, select the desired IQ and then edit its properties.

   The **IQ Properties** dialog box appears.
3. On the **General** tab, select **STC_Standard** from the **Service** drop-down list.
4. Click **OK**.

4.2 IQ Manager and Disk Space Thresholds

The IQ Manager checks the amount of free disk space on the system where IQs are located. If you have not set a disk space threshold for the Participating Host, the IQ Manager shuts down if the amount of free disk space is less than 5 MB.

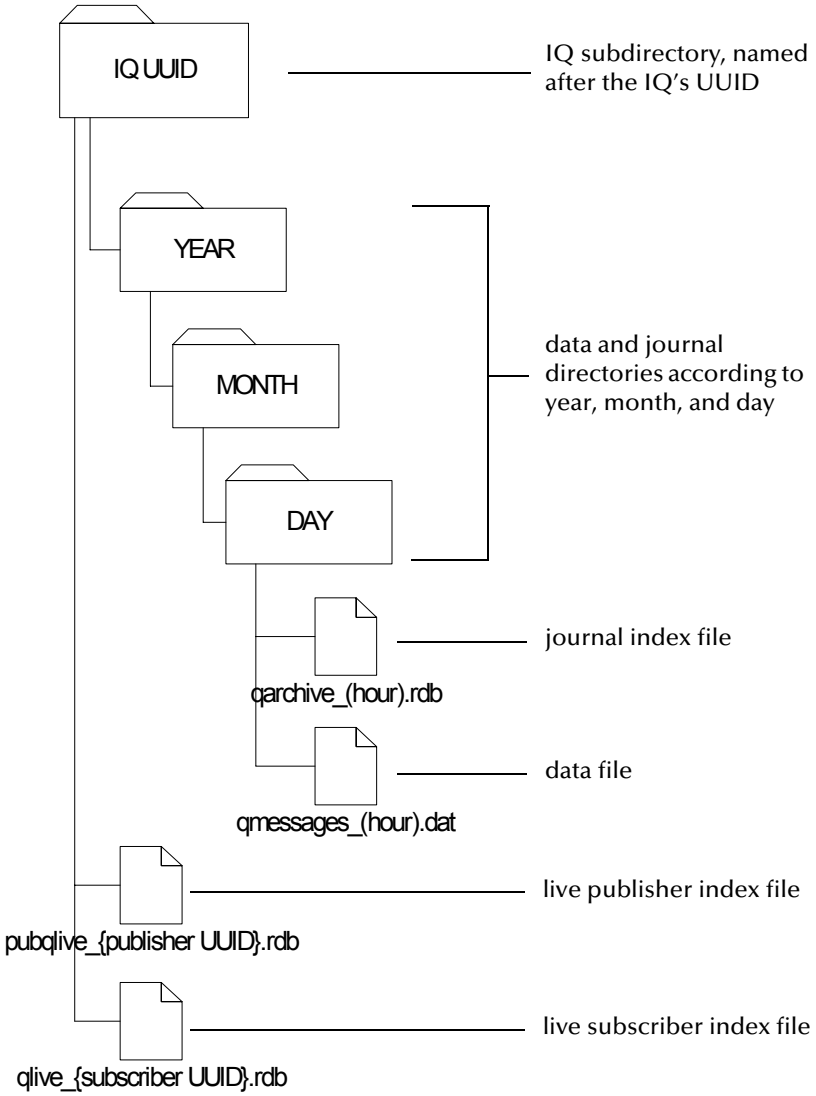
If there is a disk space threshold set for the Participating Host, the IQ Manager compares the threshold setting to the default 5 MB, and uses the highest setting. For example, if you set the threshold to 100 MB, the IQ Manager does not shut down until the disk space reaches below 100 MB.

For information about setting threshold properties for the Participating Host, refer to the *e*Gate Integrator User's Guide*.

4.3 IQ Operation

The SeeBeyond Standard IQ stores all data in disk files, including the content of the Event, expiration times, and priority. The files are then named and organized into directories based on a combination of the Universal Unique Identification (UUID) for the IQ, publishing, and subscribing components, and the date and time the data is processed. The following diagram depicts this organizational and naming structure:

Figure 1 SeeBeyond Standard IQ File Structure



The SeeBeyond Standard IQ uses two types of files in this structure: *index* and *data*.

- **Index files** contain information related to the Events and pointers to the data files. They have an **.rdb** file extension.
- **Data files** contain the actual contents of the Events. They have a **.dat** extension.

All index and data files associated with an IQ (represented by a single IQ component in the e*Gate Schema Designer) are stored in a subdirectory on the Participating Host system. By default, this subdirectory is created in the **\eGate\client\iq** folder, but you can modify this location for each IQ. For information on modifying the location of the IQ files, see **"Storage Locations (dirs)" on page 22**. The name of an IQ subdirectory corresponds to the UUID for a particular IQ. To obtain an IQ's UUID, use the **stcregutil** utility to export the schema that contains the IQ and include the **-bu** flag on the command line. For more information on exporting schema and **stcregutil**, see the *e*Gate Integrator System Administration and Operations Guide*.

Index files are divided into live and journal files. Each live file corresponds to a publisher or subscriber in the schema, and is named according to the UUID of the IQ. For example, a live index file corresponding to a publisher with a UUID of {1114BF24-63FE-11D4-A475-D8C232826E3F} would be **pubqlive_{1114BF24-63FE-11D4-A475-D8C232826E3F}.rdb**, and a live index file corresponding to a subscriber with a UUID of {18D48050-63FE-11D4-A475-B93ED49FAD6A} would be **qlive_ {18D48050-63FE-11D4-A475-B93ED49FAD6A}.rdb**.

The IQ subdirectory contains the live index files and additional data and journal subdirectories named according to the year/month/day of GMT they were created. Month and day subdirectories are hierarchically nested in the year and month subdirectories, respectively. For example, if e*Gate processed data on July 25, 2001, the data and journal subdirectories would be **2001**, **07**, and **25**. The day directories contain the journal index files and the data files for that day. The journal index and data files are named according to the hour of GMT when they were created. For example, if data is processed at 10:00 AM, the journal index file is **qarchive_10.rdb** and the associated data file is **qmessages_10.dat**.

When a Collaboration publishes an Event to an IQ, the contents of the Event are written to the data file and information related to the Event (such as expiration times and priority) is written to the live publisher and subscriber index files. After a subscribing Collaboration retrieves the Event from the IQ and completes any required processing, the entry for the Event in the live subscriber index file is removed and a new line is written in the journal index file. When you run an IQ cleanup operation, expired and processed entries in the live publisher and subscriber and journal index files and the data files themselves are moved or deleted based on Event expiration and journal expiration times. For more information on IQ cleanup operations, see **"Configuring IQ Cleanup" on page 32**.

Note: *If you reimport the current schema (creating new IQs), you must use the* ***-bu*** *command flag to maintain the current IQ UUIDs. Otherwise, even though the IQs may have the same logical names as the existing IQs, the system generates new unique UUIDs.*

4.4 Optional Parameters

You can control the behavior of the disk files by modifying the following parameters in the IQ's initialization string:

- **Disk synchronization (sync)**—Determines when the data is written to the disk file from memory. When you enable auto recovery, the disk synchronization setting is ignored.
- **Growth rate (gb)**—Specifies the number of bytes the data file will grow when additional space is required
- **Storage locations (dirs)**—Specifies the paths for the location of the live index and journal subdirectories (including the data and journal index files), overriding the information in the **.egate.store** file
- **Archive storage locations (archdir)**—Specifies the absolute path for the location of the archive.

Each of these parameters is described below. For information on how to modify the parameters in the initialization string, see **"Adding the Initialization String" on page 23**.

4.4.1 Disk Synchronization (sync)

The SeeBeyond Standard IQ synchronizes the cache to data stored by the operating system on disk at scheduled times. The **sync** parameter allows you to determine whether it will be the IQ Manager, or the operating system, that maintains cache/disk integrity and controls the synchronization schedule. The installation default, **sync=TRUE**, is to allow the IQ Manager to control the operation.

If you choose to override the default and have the operating system write the data to disk according to its schedule, the data stored in the cache at any given time may be different from data that has been written to the disk. Thus, if the IQ Manager stops abnormally while the cache and disk file are not synchronized, data is lost if **sync** was set to **FALSE**.

Note: *If the **Auto Recovery** check box is selected, disk synchronization is automatically in effect regardless of the setting. In other words, Auto Recovery overrides a displayed **sync=FALSE** setting and forces a result equivalent to **sync=TRUE**.*

4.4.2 Growth Rate (gb)

The **gb** parameter allows you to specify the number of bytes by which to increase the IQ data file when additional space is required. The installation default is 1,000,000 bytes (approximately 1 MB). The number of bytes to increase the data file must be greater than the largest Event that will be processed.

The maximum size of the data file in 32-bit operating systems is 4 GB. If you are processing data that exceeds this capacity, data will be lost.

4.4.3 Storage Locations (dirs)

The **dirs** parameter allows you to configure the IQs to store live index and data files in separate directories, or on separate drives. However, both the live index and data file directories must be on the same Participating Host as the IQ Manager.

If you configure this parameter to store the files in different directories, the IQ subdirectory occurs in both places. In the location specified for the live index files, only the live index files appear in the IQ subdirectory. In the location specified for the data files, the journal subdirectories appear, including the journal index files and the data files.

The default drive and root directory for these files is specified in the .egate.store file. For information on this file, see the *e*Gate Integrator System Administration and Operations Guide*. The installation default is to store both index and data files in the **\egate\client\iq** directory.

4.4.4 Archive Storage Location (archdir)

You use the **archdir** parameter to specify the absolute path to the location where you want to store archive files. For information about archiving events, refer to **"Setting Up Event Archiving in Standard IQs" on page 35**.

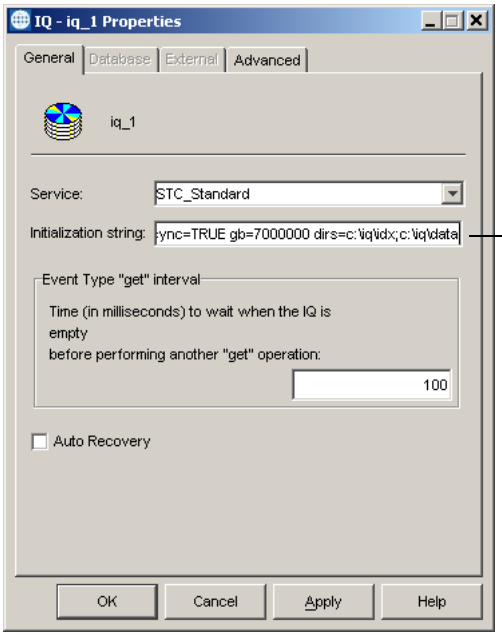
4.5 Adding the Initialization String

The initialization string is an optional command line you can add to the SeeBeyond Standard IQ to modify the parameters described above.

To enter an initialization string

1. In the **Components** tab of the e*Gate Schema Designer, expand the **Participating Host** folder and then click the IQ manager (typically ***hostname*_iqmgr**).
2. In the IQ Contents pane, double-click the IQ for which you want to specify the initialization string.
3. In the **Initialization string** box, type the parameters and values. Use spaces between each *parameter=value* pair.

The screen below shows a sample initialization string.



The following table shows the parameters for the initialization string.

Table 2 Initialization String Parameters

Name	Value	Description
sync	**TRUE** or **FALSE**	**TRUE** (the default) specifies that the IQ Manager controls the cache synchronization to disk. **FALSE** specifies that the operating system controls the synchronization schedule; however, this setting is ignored if Auto Recovery is in effect. **Note:** These values are case-sensitive.
gb	Integer, from 1 through 4,284,867,295 (4 GB)	Specifies the number of bytes to expand data file by when additional space is required. The default is 1,000,000 (approximately 1 MB).
dirs	Semicolon-delimited list of paths. For example: **c:\iq\idx;c:\iq\data**	The first path specifies the location for the index files; the second path specifies the location for the data files. **Note:** This parameter overrides the **IQueueIndex** and **IQueueData** values in the **.egate.store** file.
archdir	Absolute path for archive file without spaces	Specifies the location of the archive file. The IQ Manager creates the directory if it does not exist.

Table 2 Initialization String Parameters (Continued)

Name	Value	Description
fast_cleanup	**TRUE** or **FALSE**	**FALSE** (the default) has no effect. **TRUE** has the following effects when auto recovery is enabled: ▪ The IQ Manager does not update the log files during cleanup; however, Events deleted during cleanup are still removed permanently. ▪ Events journaled by the IQ manager during cleanup are not shown as journaled after a recovery is performed by the IQ Manager to restore the IQ. However, even though the Events appear as unjournaled, they are not sent by the IQ Manager to subscribers, because they have already met the "to be journaled" criterion. To journal these Events after a recovery, you need to schedule the IQ Manager to perform a cleanup on the IQ.
free_buffer	**TRUE** or **FALSE**	Adding **free_buffer=TRUE** to the **Initialization string** field forces the IQ Manager to free buffers used by **iq-put**, **iq-get**, and **iq-peek** operations after the operations are completed. Adding **free_buffer=FALSE** (or keeping the field blank) causes the IQ Manager to keep the buffers for subsequent **iq-put**, **iq-get**, and **iq-peek** operations.

4.6 Enabling Auto Recovery

You can enable auto recovery for SeeBeyond IQs (not database IQs). When auto recovery is enabled, the IQ Manager saves all IQ operational information to log files. The IQ Manager uses these log files to perform recovery in the event of queue file corruption due to an abnormal shutdown. A new log file is generated each hour.

Enabling auto recovery may result in diminished performance and increased disk space use and space due to logging operations. You can enhance performance otherwise by setting the **fast_cleanup=TRUE** option in the initialization string as described in the previous section.

Auto recovery is disabled by default. With auto recovery enabled, the IQ Manager automatically synchronizes cache to the hard disk even if your initialization string has **sync=FALSE**. For information, refer to **"Adding the Initialization String" on page 23**.

The SeeBeyond IQ Manager uses a status file to determine whether a shutdown requires recovery. This status file has the states "good" and "bad." You can use a text editor to view the status file or to edit the file to a "bad" state to force the IQ Manager to perform a recovery. You may want to do this in the event that you are unsure whether a queue is corrupted.

4.7 Enabling Standard IQ First In, First Out Order

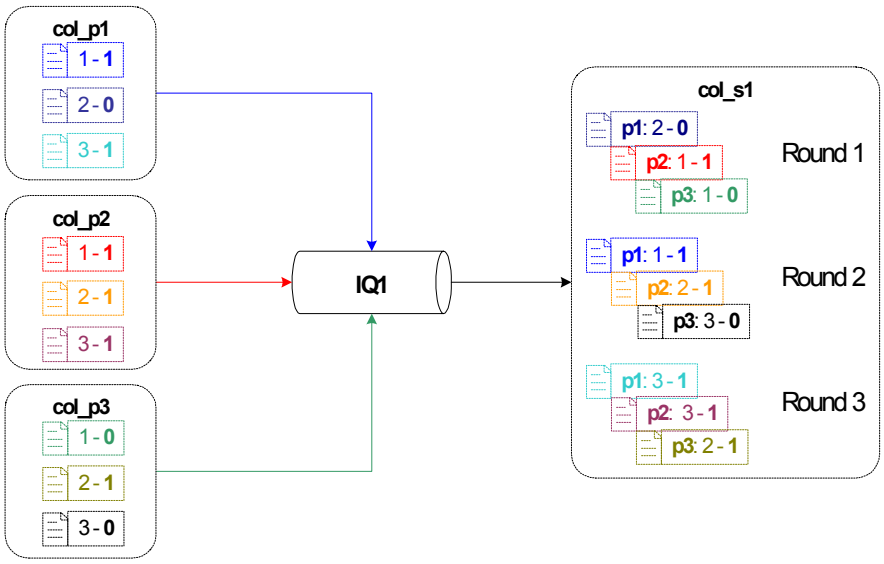
If it is applicable to your environment, you can opt to change IQ processing for multi-mode e*Ways to first in, first out (FIFO) order. With FIFO disabled (which is the e*Gate default), subscribing Collaborations retrieve Events in round robin fashion from each *triggering* Event Type/Publisher combination to which they subscribe.

For example, imagine a schema where Collaboration **col_s1** subscribes to an Event Type published by Collaborations **col_p1**, **col_p2**, and **col_p3** to **IQ1**.

In round robin fashion, the subscribing Collaboration **col_s1** retrieves from **IQ1** a single Event from one of the publishing Collaborations, for example, **col_p3**. This Event has the highest priority (with 0 being the highest) and the oldest (lowest) sequence number compared to all the Events published by **col_p3**.

col_s1 processes this Event, sends a "done" acknowledgement to **IQ1**, and retrieves the highest priority/oldest sequence number Event for the next publishing Collaboration, for example, **col_p2**, and so on. The figure below shows the Event processing when FIFO is disabled.

Figure 2 Non-FIFO Processing

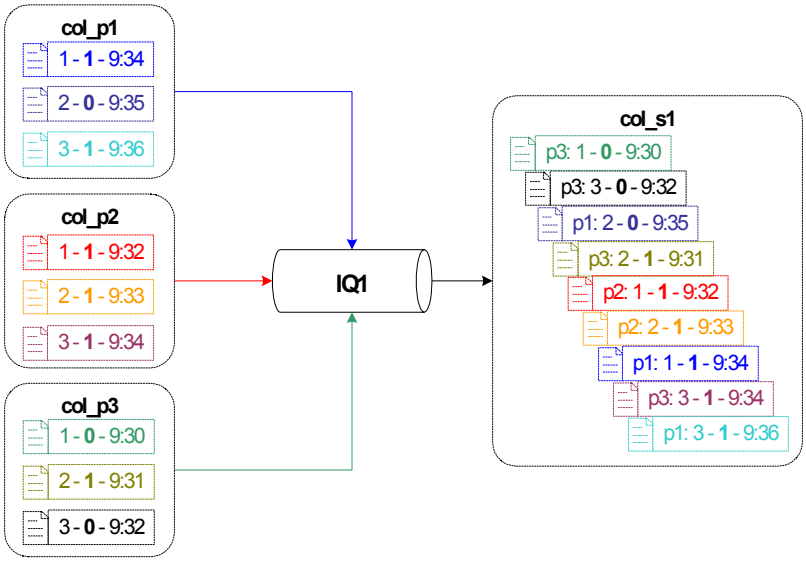


In the figure above, each Collaboration publishes three Events, for example, Event 1 - **1**, where 1 is the Event sequence number, and **1** is the Event priority. With non-FIFO processing, the IQ Manager retrieves the Events with the oldest sequence number and the highest priority and processes them in that order before moving on onto the next round. For example, in the first round, **IQ1** retrieves and processes the following Events:

- From **col_p1**, Event 2 - **0**
- From **col_p2**, Event 1 - **1**
- From **col_p3**, Event 1 - **0**

The figure below shows the IQ Manager processing with FIFO enabled.

Figure 3 FIFO Processing



With FIFO enabled, the subscribing Collaboration no longer cycles in round robin retrieving one Event at a time. Instead, it retrieves the highest priority/lowest sequence number Event for each publishing Collaboration per Event type. In the schema discussed above, **col_s1** retrieves and caches all Events from **col_p1**, **col_p2**, and **col_p3**. It compares the priorities and enqueue time of all four Events. It processes the Event with the highest priority and oldest enqueue time. The Collaboration retrieves Events for each Event Type/Publisher combination for which there is no Event cached. It then goes on to process the next Event by determining the processing order.

The figure above shows the processing order for all Events cached. This is a simplified scenario. In a real schema, **col_s1** retrieves other Events as they come in and compares them each time with the cached Events before processing; so if col_s1 retrieves a new Event with priority 0 when it is about to process p3: 2 - 1 - 9:31, the 0 priority Event is processed first.

Notes about FIFO

- If a rollback occurs, all cached Events roll back.
- To ensure FIFO Event processing for a multiple IQ setup, use standard IQs only. FIFO processing works differently for JMS IQs. For information about FIFO in JMS IQs, refer to the *SeeBeyond JMS Intelligent Queue User's Guide*.
- The enqueue time is set by a publishing Collaboration when it sends the Events to the IQ Manager. Because FIFO relies on the enqueue time, the system clocks on Participating Hosts must be synchronized with accurate GMT for FIFO to occur as expected. For example, if system A in New York is set to 9:46 PM, system B in Los Angeles must be set to 6:46 PM.
- Events may not be processed in enqueue time order if a publishing Collaboration or an IQ is slower handling acknowledgements. In that case, the Events are received in correct enqueue time order, but may appear to be out of order when you check the creation time.

- Using the instance **receive()** method in the business rule of a Java Collaboration without specifying an Event Type as an argument will return a message of one of the Event Types that the instance subscribes to. Internally, the **receive()** method cycles through every Event Type to which the instance subscribes—in a round-robin fashion—and attempts to receive a message. When it receives a message, it stops and remembers the Event Type so that the next time **receive()** is called, it can start with the next Event Type in line. The returned message will be the earliest published message of the Event Type available at that point in time—assuming multiple publishers are publishing messages of that Event Type. The round-robin cycle stops once it has exhausted all Event Types that the instance subscribes to and once there are no more Event Types available.

  If the instance **receive** method is called with an Event Type as an argument--for example, `receive(event-type1)`--then a message of the specified Event Type is returned. Again, the message is the earliest published message of the specified Event Type at that point in time.

- If you use the IQ Manager to delete a cached Event, the Event is not published.

FIFO Impact on Republishing

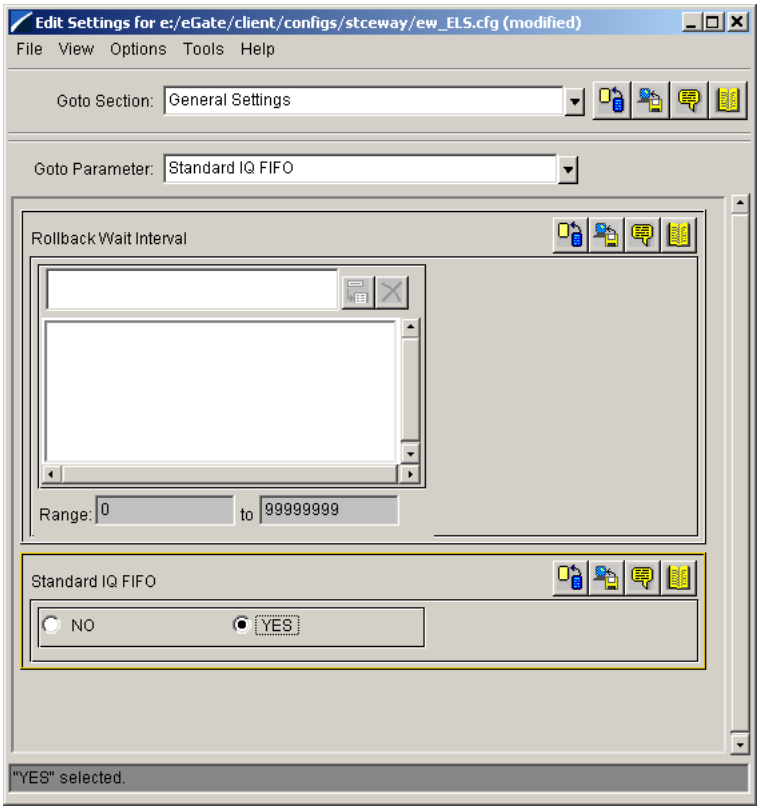
If a published Event fails, the subscribing Collaboration continues to wait for the Event to be processed. You must republish the Event to continue processing. FIFO applies to republished events as follows. When you republish the Event, its sequence number remains the same, but its enqueue time is updated to the time of the republish. Also, its priority number is decreased by 1, thereby increasing the priority level (unless the original priority level was already 0).

This is so that even though the Event has a later enqueue time, the higher priority level ensure earlier selection for processing. For example, if the Event's priority was 3 originally, the priority of the republished Event is 2. If the original priority was 0, republishing does not change the Event's priority.

To enable Standard IQ FIFO order

1. In the **Components** tab of the e*Gate Schema Designer, expand the Participating Host folder, and click the Control Broker (***hostname***_**cb**).
2. In the Contents pane, double-click the multi-mode e*Way for which you want to enable FIFO.
3. In the **GoTo Section** list, click **General Settings**.

4 Under **Standard IQ FIFO**, click **YES** as show below.



5 On the **File** menu, click **Save**.

4.8 Event Status

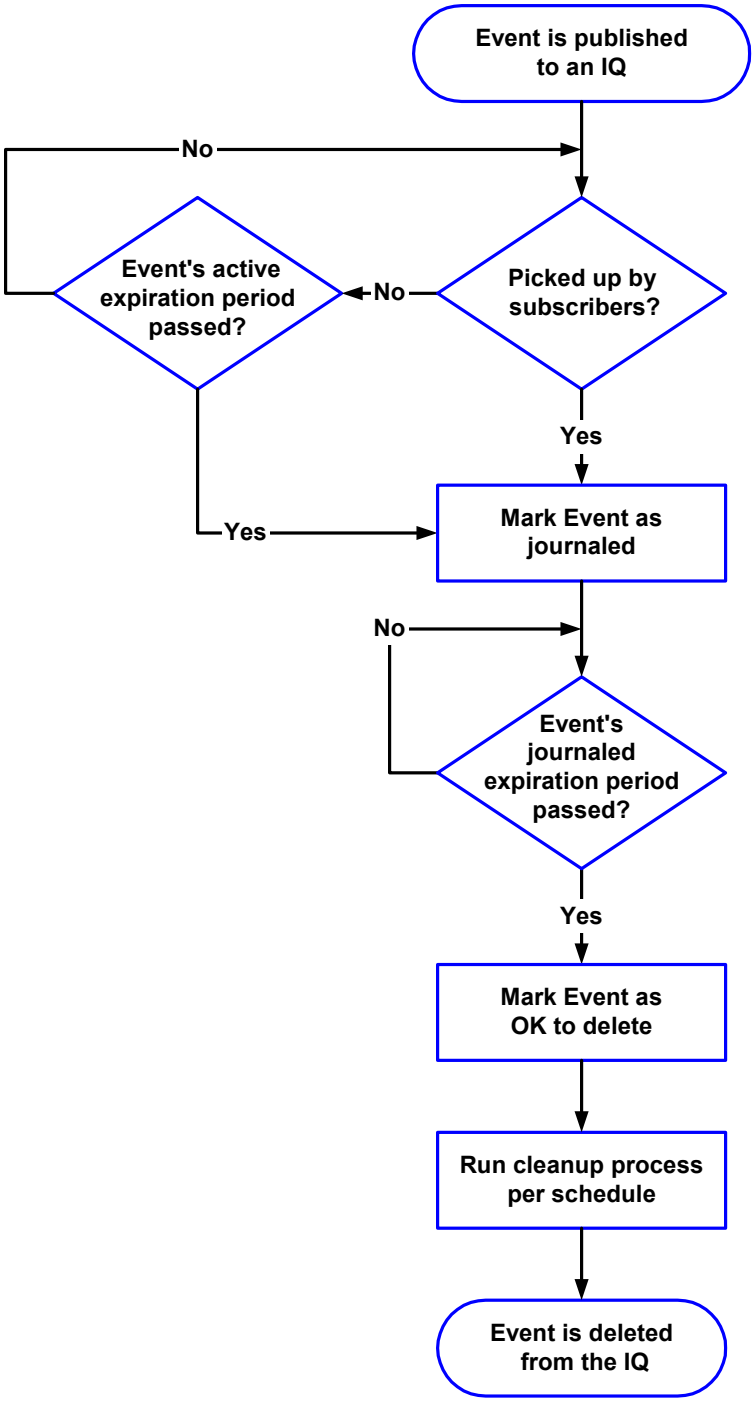
An Event within an IQ can be in one of two states:

- *Active* Events are published but not yet written to the journal, and are waiting for one or more subscribers to retrieve the Event and mark it **DONE**. An active Event is written to the journal after all subscribers mark it **DONE** or after its expiration time elapses, whichever comes first.
- *Journaled* Events are Events that are no longer active, either because they were picked up and marked "done" by all their subscribers or because they expired. If a journaled Event remains journaled past its expiration time, it is deleted when you run an IQ cleanup operation (see **"Configuring IQ Cleanup" on page 32**).

Note: *The IBM MQSeries IQ does not journal Events.*

The following flowchart illustrates the steps e*Gate takes in the journaling and IQ cleanup processes.

Figure 4 Journaling and Deleting Events

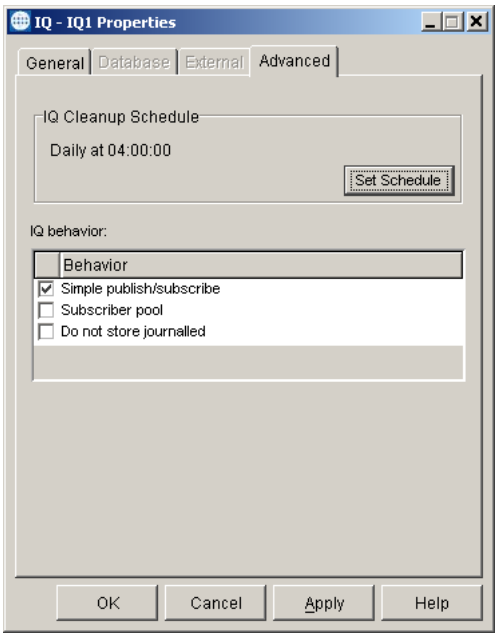


4.8.1 Subscriber Pooling

If multiple Collaborations are subscribing to an Event Type published to a single IQ, you can configure the IQ to change the status of an Event when any subscriber accesses it, or to wait until all subscribers take action on it. Changing an Event's status based on the activities of any one of a number of available subscribers is called *subscriber pooling*.

To configure subscriber pooling for an IQ

1. In the **Components** tab of the e*Gate Schema Designer, expand the Participating Host folder and the Control Broker folder.
2. Double-click the IQ Manager for which you want to configure IQs.
3. In the Component Editor, double-click the IQ. to display the **IQ Properties** dialog box.
4. Click the **Advanced** tab.
5. Under **IQ behavior**, select **Subscriber pool**.



6. Click **OK** to close the **IQ Properties** dialog box.

4.8.2 Setting Expiration Times

e*Gate sets an Event's expiration times when the Event is published. To set an Event's expiration times, you must configure the publication properties of the Collaboration publishing that Event.

To set the expiration times for a Collaboration's published Events

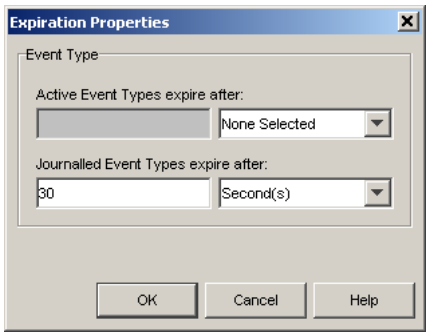
1. Follow step 1 through 4 in **"Subscriber Pooling" on page 31**.
2. Under **Behavior**, do one of the following:

- Select **Do not store journaled** to delete journaled Events (Events that have been picked up or have expired) even though the journal Event expiration time has not been reached. Go to step 5 below.
- Clear **Do not store journaled** to delete journaled Events only when the journal Event expiration time is reached. Continue with step 3. Note that the files are only deleted when the cleanup schedule runs.

3 When the new properties sheet displays, click **Expiration**.

The **Expiration Properties** dialog box appears. See Figure 5.

Figure 5 Expiration Properties



4 Set the **Active** and **Journaled** expiration times. You can select no expiration, years, months, weeks, or days.

5 Click **OK** on each open dialog box until you return to the e*Gate Schema Designer's main screen.

4.9 Configuring IQ Cleanup

The IQ cleanup process deletes expired Events from an IQ's journal. This process provides the only means to automatically delete journaled Events from an IQ.

Note: *To delete individual Events from an IQ, you can use the* ***stciqutil*** *utility (see* **"Deleting Events from an IQ" on page 73***). For Standard IQs, you can also use the IQ Administrator application (see* **"To delete an Event or a range of Events" on page 66***).*

4.9.1 Setting the IQ Cleanup Schedule

The IQ cleanup runs on a schedule that you configure, and each IQ has its own schedule. Since the IQ Manager starts the cleanup process, the IQ Manager must start automatically for the IQ cleanup schedule to work.

To set the IQ cleanup schedule

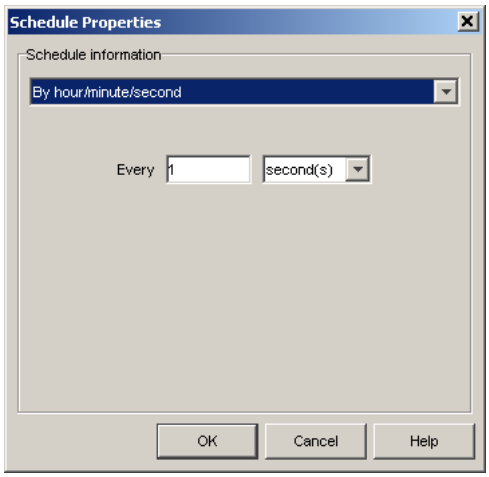
1 In Schema Designer, in the Navigator's **Components** tab, select the Participating Host and Control Broker of the IQ Manager whose IQs you will be configuring.

2 In the Component Editor, select the desired IQ and then edit its properties.

The **IQ Properties** dialog box appears.

3 Select the **Advanced** tab, and then click **Set Schedule**.

The **Schedule Properties** dialog box appears.



4 Under **Schedule Information**, click one of the following:

- **No schedule**—No IQ cleanup is run.
- **One time schedule**—The IQ cleanup is run one time only, on the date and time you select.

  Note: *The* ***Time*** *box shows the time entered for the last schedule, or the time when the dialog box was displayed last.*
- **By hour/minute/second**—The IQ cleanup is run based on the number of hours, minutes, or seconds that you select.
- **Daily**—The IQ cleanup is run at the time you select everyday.
- **Weekly**—The IQ cleanup is run on the day and time you select every week.
- **Monthly on date**—The IQ cleanup is run on the date and time you select every month.
- **Monthly on day**—The IQ cleanup is run on the day and time you select every month (for example, the first Monday, the second Tuesday, or the third Saturday).

The default is to run the cleanup daily at 4:00 AM.

Note: *The schedule uses the local time zone of the IQ Manager supervising the IQ.*

4.9.2 Cleanup for the SeeBeyond Standard IQ

Each hour, e*Gate creates a new journal index file and a new data file to hold that hour's IQ data. The cleanup process deletes the files only after **all** the entries in the journal index file have been marked **OK to delete**. See **Figure 4 on page 30**.

If the journal expiration time is set to zero, the IQ journal index and data files are never deleted unless you selected the **Do not store journaled** option in the IQ properties.

Note: *If a queue cleanup is already running, a second queue cleanup will not start.*

4.9.3 Deleting Sybase IQs or IQ Data

To help you delete Sybase IQs, e*Gate includes a special command-line utility, **stciqsbadmin**. This utility is installed in the **\client\bin** directory and enables you to do the following:

- Remove all the Events in an IQ
- Remove an IQ
- Remove all the IQs in an e*Gate schema
- Remove all the IQs and shared resources for a database user

This utility takes no command-line arguments. To use it, simply start the utility and respond to the prompts.

Note: *You need a valid Sybase user name and password to use this utility.*

4.9.4 Deleting Oracle IQs or IQ Data

To help you delete Oracle IQs, e*Gate includes a special command-line utility to perform this function. Like the Sybase IQ utility, the Oracle IQ utility enables you to do the following:

- Remove all the Events in an IQ
- Remove an IQ
- Remove all the IQs in an e*Gate schema
- Remove all the IQs and shared resources for a database user

The utility is located on the installation CD-ROM in the **\utils\oracle** directory. You must run the utility from a machine that contains the Oracle client. To run the utility, either:

- Copy the files from the CD-ROM directory to your local drive, then navigate to that directory from a command-line prompt, or
- From a command line, navigate to the **\utils\oracle** directory on the CD-ROM.

On the command line, type the following:

```
sqlplus username/password@servicename @stciqrm.sql
```

where

username is the user name for the Oracle database
password is the password associated with the username
servicename is the service name for the Oracle database, as defined by your DBA

4.10 Coordinating Expiration Times and Cleanup Schedules

The expiration times and cleanup schedules you select will be unique to your installation; but whatever settings you choose, make sure that the values you select both maintain a relatively consistent journal size and balance the load on the cleanup process.

If you run the cleanup process too infrequently, disk space usage will continue to climb up until the point at which the cleanup runs. At this point, the cleanup process may require significant system resources to clear the journal, costing system performance and time. Running the cleanup process too frequently means that the cleanup process has very little to do whenever it runs, impacting system performance for little gain.

We recommend that you determine Event expiration times first; then, adjust your cleanup schedule accordingly to support that decision. For example, if you determine that Events should expire after one week, running the cleanup daily will ensure that the system always has the prior seven days' worth of Events on any given day. Other considerations (disk or system loading) may also arise. You may need to try several different combinations of settings before you find the optimal balance.

Note: *If available disk space is a concern, you can set a disk-usage threshold at which e*Gate automatically sends you a warning alert. For information on setting disk usage thresholds, see the* ***e*Gate Integrator User's Guide****.*

4.11 Setting Up Event Archiving in Standard IQs

This section describes how you set up Event archiving. To enable archiving, you do the following:

- Specify the location of the archive
- Set up a cleanup schedule
- Specify the journal expiration time for all Event Types that publish to the IQ or configure the IQ to not store journaled Events.

At cleanup time, all events of the selected Event Types that are active expired or journal expired are archived. For information about IQ cleanup, refer to **"Configuring IQ Cleanup" on page 32**.

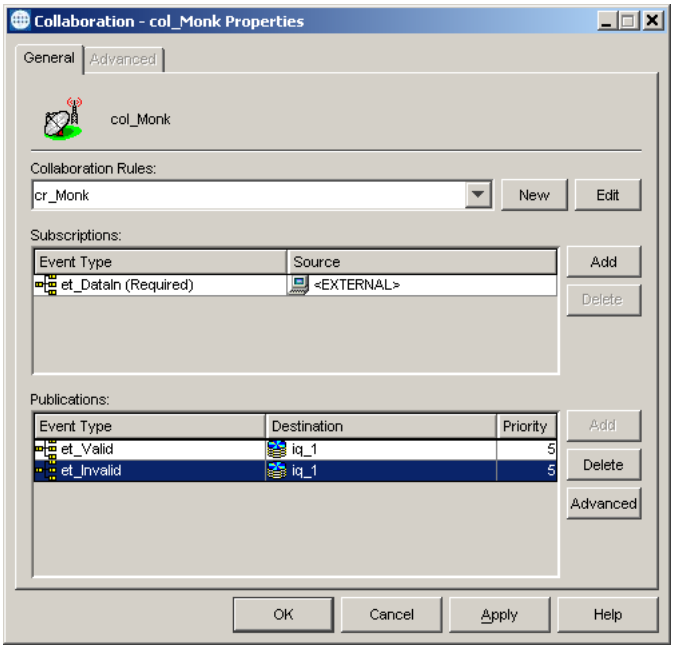
When you enable archiving, the IQ manager creates an archive file named ***IQname-YYYY-MM-DD-HH*.archive**. The time is local time. For example, Events that were enqueued to IQ1 during the 11th hour on June 26, 2003 are in an archive file named **IQ1-2003-06-26-11.archive**. The archive includes all Events that were journaled or expired during the hour; for example, all Events enqueued from 10:00 through 10:59:59:999 can be included in the **IQ1-2003-06-26-10.archive**. Only journal expired Events are archived, so if the expiration time is more than an hour, the Events would not be included in the archive for the next hour. The earliest time that the IQ manager can create an archive for the 10th hour enqueued and expired Events is at 11. When this archive is actually created depends on your settings for the cleanup schedule.

The archive files are portable; you can view archive files with the IQ Administrator regardless of the platform on which the archives were created. You can store archives on CD-ROM if necessary. You can view archives and republish Events to the IQ as described in **"Viewing Archives and Republishing Events" on page 81**.

To set up Event archiving

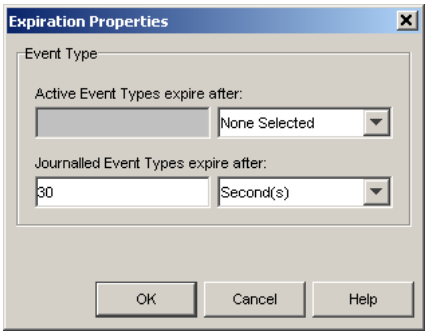
1. In the **Components** tab of the e*Gate Schema Designer, expand the **Participating Host** folder, and then click the IQ manager (***hostname*_iqmgr**).
2. In the IQ Contents pane, double-click the IQ for which you want to archive Events.
3. In the **Initialization string** box, type **archdir=*path*,** where *path* is the absolute path to the archive directory. For Windows, the path must include a mapped drive. For UNIX the path must be a mounted drive. If the directory does not exist, the IQ Manager creates the directory. For example: **e:\eGate\Archives** (Windows) or **/egate/archive** (UNIX).
4. Click the **Advanced** tab.
5. Click **Set Schedule**.
6. Under **Schedule Information**, select the interval for the IQ cleanup.
7. Click **OK** to close the **Schedule Properties** dialog box.
8. Click **OK** to close the **IQ Properties** dialog box.
9. In the **Components** tab, double-click the publishing e*Way.
10. Double-click the Collaboration for the e*Way.

    The **Collaboration Properties** dialog box appears as shown below.



11. Under **Publications**, click the Event Type you want to archive.
12. Click **Advanced**.
13. Click **Expiration**.

The **Expiration Properties** dialog box appears as shown below.



14 Under **Journalled Event Types expire after**, enter the amount of days, hours, minutes, or seconds that it takes for journaled Events to expire for this Event Type.

15 Repeat steps 11 through 14 for additional Event Types. You must specify the journal expiration time for ***all Event Types that publish to the same IQ***. If you do not set this for all Event Types, the IQ manager does not create the archive file unless you selected the **Do not store journalled** option in the IQ properties.

Chapter 5

Configuring the SeeBeyond Memory Loopback IQ

This chapter describes how to configure the SeeBeyond Memory Loopback IQ.

5.1 Defining the IQ

The first step after you create an IQ component is to define the IQ Service the IQ will use. This identifies the type of IQ.

To define the IQ Service for the SeeBeyond Memory Loopback IQ

1. Launch the e*Gate Schema Designer.
2. Open the schema that contains the IQ component you want to define.
3. In the Component editor, select the IQ and display its properties.

   The **IQ Properties** dialog box appears.
4. On the **General** tab, select **STC_Memory_Loopback** from the **Service** drop-down list.
5. Click **OK**.

5.2 Implementation

In order to use the Memory Loopback IQ, you must create two Collaborations within a single component (e*Way or BOB): One to subscribe to the Event Type (inbound) and one to publish the Event Type to the IQ (outbound). If you are publishing multiple Event Types, each one must be published to a separate IQ. For information on creating and configuring modules, Collaborations, and IQs, see the *e*Gate Integrator User's Guide*.

The Memory Loopback IQ only functions when passing Events within the same component; it does not span between processes. Within an e*Way or BOB, you should configure one Collaboration to publish Events to the Memory Loopback IQ, and another Collaboration within the same e*Way or BOB to subscribe to the Events

published by the first Collaboration. The Memory Loopback IQ cannot be used to pass Events from one component (e*Way or BOB) to another.

Note: *Queuing Events in this manner is highly volatile. Events can be lost if the system goes down while processing within this IQ.*

Chapter 6

Configuring the ODBC IQ

This chapter describes how to configure the ODBC IQ.

6.1 Defining the IQ

The first step after you create an IQ component is to define the IQ Service the IQ will use. This identifies the type of IQ.

To define the IQ Service for the ODBC IQ

1. Launch the e*Gate Schema Designer.
2. Open the schema that contains the IQ component you want to define.
3. In the Component editor, select the IQ and display its properties.

   The **IQ Properties** dialog box appears.
4. On the **General** tab, select **ODBC** from the **Service** drop-down list.
5. Select the **Database** tab. Complete the following fields:

Field	Description
Database name	This is the name of the data source as per the ODBC connection. This name must match the name specified during the ODBC connection setup (see the **"Creating the ODBC Connection (Windows)" on page 42**).
Login name	The username to log in to the database.
Password	The password associated with the login username for the database.
Confirm password	The password associated with the login username for the database. The password must match what was typed in the **Password** field.
Participating Host	The Participating Host on which the IQ runs.

   The **Schema name** field can be left blank.
6. Click **OK**.

6.2 Database Functions

The ODBC IQ uses a number of functions to communicate with the database. The defaults for these functions are listed below.

Note: *Unless you are directed to do so by SeeBeyond support personnel, do not change the default settings.*

1. In the Schema Designer's Component editor, select the **Services** folder.
2. Select **Odbc** and display its properties.

Table 3 ODBC IQ Properties

	Function	Default
General	Initialize function	IQ_Odbc_Initialize
	Terminate function	IQ_Odbc_Terminate
	Reorganize function	IQ_Odbc_Reorganize
	Shared handle function	IQ_Odbc_GetSharedHandle
	Control function	IQ_Odbc_Control
Access functions	Get function	IQ_Odbc_MsgGet
	Peek function	IQ_Odbc_MsgPeek
	Put function	IQ_Odbc_MsgPut
	Mark function	IQ_Odbc_MsgMark
	Expire function	IQ_Odbc_MsgExpire
	Free function	IQ_Odbc_MsgFree
Transaction functions	Begin transaction function	IQ_Odbc_BeginTran
	Commit function	IQ_Odbc_Commit
	Rollback function	IQ_Odbc_Rollback
	Post Transaction	

6.3 External Configuration Requirements

Before you can use the ODBC IQ you must do the following:

- Install the Merant 4.0 ODBC driver.
- Create the necessary tables and stored procedures on the database. See **"Creating the Database"**, below.
- Create the ODBC connection on the Participating Host that uses the IQ.

- To create the ODBC connection on Windows systems, see **"Creating the ODBC Connection (Windows)"**, below.
- To create the ODBC connection on UNIX systems, see **"Creating the ODBC Connection (UNIX)" on page 44**.

6.3.1 Installing the Merant Driver

Install the Merant ODBC 4.0 driver on the Participating Host that uses the ODBC IQ. You can obtain the driver from the Merant installation CD. Follow the manufacturer's instructions to install the driver.

6.3.2 Creating the Database

For New Installations

If you have not previously created a database for an e*Gate IQ on a Microsoft SQL server, you no longer need to run the SQL script **IQcreate.sql** as you did in previous e*Gate releases. All tables, indexes, and stored procedures are automatically created when the e*Gate components connect to the Microsoft SQL server; all you need to do is choose this as your default database (in step 10 of the **procedure on page 44**).

For Upgrading Installations

To reuse a database created for an e*Gate IQ in a previous e*Gate installation (prior to release 4.5.2), you need to drop all e*Gate created stored procedures (stored procedures created by **IQcreate.sql**) in the database; you no longer need to run the SQL script **IQcreate.sql** as you did in previous e*Gate releases—at release 4.5.2 and later, all tables, indexes, and stored procedures are automatically created when the e*Gate components connect to the Microsoft SQL server.

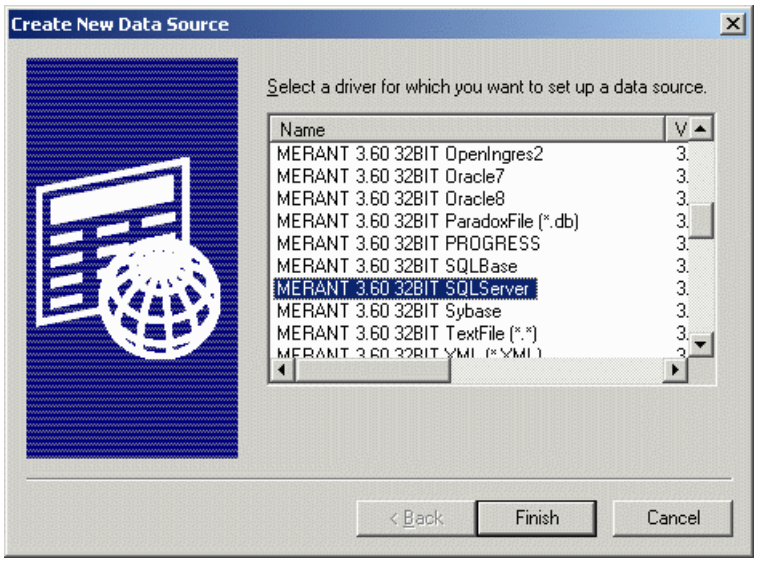
6.3.3 Creating the ODBC Connection (Windows)

Important: *Install the SQL Server client on the Participating Host* ***before*** *creating the ODBC connection.*

To create the ODBC connection on the Participating Host

1. On the taskbar, click the **Start** button, point to **Settings**, then click **Control Panel**.
2. Do the following:
   - On Windows 2000 or Windows XP systems, double-click **Administrative Tools**, then double-click **Data Sources (ODBC)** to open the **ODBC Data Source Administrator** dialog box.
3. Select the **System DSN** tab.
4. Click **Add**.

Figure 6 Create New Data Source

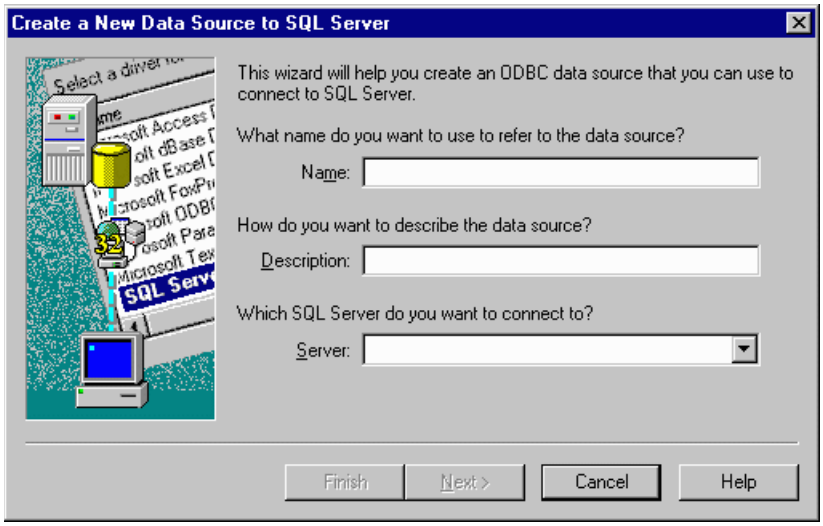


5 In the **Create New Data Source** dialog box, select **MERANT 4.0 32BIT SQLServer**. You may need to scroll down to see the driver.

6 Click **Finish**.

This launches the wizard to create a new data source to the SQL Server database.

Figure 7 Create a New Data Source to SQL Server - Data Source



7 Type a name and description for your data source.

Important: *The name of the data source must exactly match the* ***Database name*** *on the* ***Database*** *tab in the* ***IQ Properties*** *dialog box. (See* **"Defining the IQ" on page 40***.)*

8 In the **Server** field, select the server the IQ will connect to.

9 Click **Next**.

10 When prompted to choose the default database for the ODBC connection, choose the database you just created.

The rest of the wizard allows you to set up configuration parameters related to the SQL Server connection. Follow the on-screen instructions to complete the wizard, or contact your DBA for assistance.

6.3.4 Creating the ODBC Connection (UNIX)

Creating the ODBC connection on UNIX systems requires the following tasks on the Participating Host that uses the IQ:

- Create the ODBC data source definition file.
- Set the shared library search path.
- Set the ODBC_HOME environment variable for your system.

The tasks can be performed in any order. Each task is described below.

Creating the Data Source Definition File

In the UNIX environment, there is no ODBC Administrator. To configure a data source, you must create or edit an existing **.odbc.ini** file, a plain text file that is normally located in the user's $HOME directory. You maintain this file using any text editor, and define data source entries as described in the "Connecting to a Data Source Using a Connection String" section of each driver's chapter.

Following is an example **.odbc.ini** file with the required sections for the ODBC IQ. Create or modify an existing file in a text editor, replacing the system-specific information in the example with the correct values for your implementation. For a description of each element, see the explanation following the example.

After you have completed the file, save it in your $HOME directory as **.odbc.ini**.

Important: *Do not include the line numbers in your file. They are present in the example for explanation purposes only.*

```
1   [ODBC Data Sources]
2   SeeBeyondServer=MERANT 4.0 SQLServer driver
3
4   [SeeBeyondServer]
5   Driver=/opt/odbc/lib/dgmsss15.so
6   Description=SQL Server
7   Database=SeeBeyondDB
8   ServerIPAddress=10.2.28.2,1300
9   TDS=7.0
10  QuoteID=No
11  AnsiNPW=No
12
13  [ODBC]
14  Trace=0
15  TraceFile=odbc.log
16  TraceDll=/opt/odbc/lib/odbctrac.so
17  InstallDir=/opt/odbc
```

Explanation

The following table describes the system-specific information in the **.odbc.ini** file. Items not listed in the table should appear in your file exactly as they appear in the example.

Table 4 Explanation of **.odbc.ini**

Line Number	Description
2	The logical name of the SQL Server machine on which the database runs. Replace **SeeBeyondServer** with the actual name of your server.
4	The title of the section containing configuration parameters for the server. Replace **SeeBeyondServer** with the name you entered in line 2.
5	The location and filename of the ODBC driver that connects to SQL Server. If necessary, replace **/opt/odbc/lib/** with the correct path to your driver, and **dgmsss15.so** with the filename of your ODBC driver. (***Note***: If you are using an HP-UX system, the file extension for the driver will be **.sl**.)
7	The name of the database you are using with the ODBC IQ. Replace **SeeBeyondDB** with the name of your database.
8	The server's IP address and port number. Replace **10.2.28.2** with your server's IP address and **1300** with the port number.
9	If you are using SQL Server 2000, replace **7.0** with **2000**.
15	The name of the trace log file. If you want log information written to a different file, replace **odbc.log** with the desired filename.
16	The location and filename of the trace library for the ODBC driver. If necessary, replace **/opt/odbc/lib/** with the correct path to the library file, and **odbctrac.so** with the correct library filename.
17	The installation directory for the ODBC driver. If necessary, replace **/opt/odbc** with the correct path for your implementation.

Setting up the Shared Library Search Path

You must set up the shared library search path used by the ODBC IQ. The library search path environment variable allows the ODBC core components and drivers to be located at the time of execution.

You can define the environment variable in *.profile* in the Korn/Bash shell or in *.cshrc* in the C shell. Use the following scripts to set the variable:

Caution: *The following scripts assume the ODBC driver is installed in the* ***/opt/odbc*** *directory. If your driver is installed in a different directory, make the appropriate substitutions.*

Korn/Bash Shell

```
if [ "$LD_LIBRARY_PATH" = "" ]; then
  LD_LIBRARY_PATH=/opt/odbc/lib
else
  LD_LIBRARY_PATH=/opt/odbc/lib:$LD_LIBRARY_PATH
fi
export LD_LIBRARY_PATH
```

C Shell

```
if ($?LD_LIBRARY_PATH == 1) then
  setenv LD_LIBRARY_PATH  /opt/odbc/lib:${LD_LIBRARY_PATH}
else
  setenv LD_LIBRARY_PATH  /opt/odbc/lib
endif
```

Setting ODBC_HOME

The ODBC_HOME environment variable identifies the location of the ODBC directory. To set this variable run the following command, where **/opt/odbc** is the location of the ODBC driver:

Korn/Bash Shell

```
export ODBC_HOME=/opt/odbc
```

C Shell

```
setenv ODBC_HOME /opt/odbc
```

Chapter 7

Configuring the Oracle IQ

This chapter describes how to configure the Oracle IQ.

7.1 Defining the IQ

The first step after you create an IQ component is to define the IQ Service the IQ will use. This identifies the type of IQ.

To define the IQ Service for the Oracle IQ

1. Launch e*Gate Schema Designer.
2. Open the schema that contains the IQ component you want to define.
3. In the Component editor, select the IQ and display its properties.

   The **IQ Properties** dialog box appears.
4. On the **General** tab, select **Oracle8i** from the **Service** drop-down list.
5. Select the **Database** tab and fill in the fields on this tab. For help with the fields, click **Help**.
6. Click **OK**.

7.2 Database Functions

The Oracle IQ uses a number of functions to communicate with the Oracle database. The defaults for these functions are listed below.

Note: *Unless you are directed to do so by SeeBeyond support personnel, do not change the default settings.*

1. In the Schema Designer's Component editor, select the **Services** folder.
2. Select **Oracle8i** and display its properties.

Table 5 Oracle IQ Properties

Tab	Function	Default
General	Initialize function	IQ_Oracle_Initialize
	Terminate function	IQ_Oracle_Terminate
	Reorganize function	IQ_Oracle_Reorganize
	Shared handle function	IQ_Oracle_GetSharedHandle
	Control function	IQ_Oracle_Control
Access functions	Get function	IQ_Oracle_MsgGet
	Peek function	IQ_Oracle_MsgPeek
	Put function	IQ_Oracle_MsgPut
	Mark function	IQ_Oracle_MsgMark
	Expire function	IQ_Oracle_MsgExpire
	Free function	IQ_Oracle_MsgFree
Transaction functions	Begin transaction function	IQ_Oracle_BeginTran
	Commit function	IQ_Oracle_Commit
	Rollback function	IQ_Oracle_Rollback
	Post Transaction	

7.3 External Configuration Requirements

Oracle-based IQs rely on external databases. Because of this, these IQs have special requirements that your DBA must configure before the IQs can operate properly.

The user must be able to access the tablespaces **egatetable** and **egateindex**.

The user under which the e*Gate process is running must have the following privileges:

- resource
- connect
- create/drop public synonym

Your DBA can grant the above privileges using the following SQL statement:

```
GRANT RESOURCE, CONNECT, CREATE PUBLIC SYNONYM, DROP PUBLIC SYNONYM
TO user_name;
```

where *user_name* is the user that wants to access an Oracle IQ.

Note: *For detailed information on the Oracle IQ database schema, see* **Appendix A**.

Chapter 8

Configuring the Sybase IQ

This chapter describes how to configure the Sybase IQ.

8.1 Defining the IQ

The first step after you create an IQ component is to define the IQ Service the IQ will use. This identifies the type of IQ.

To define the IQ Service for the Sybase IQ

1. Launch the e*Gate Schema Designer.
2. Open the schema that contains the IQ component you want to define.
3. In the Component editor, select the IQ and display its properties.

   The **IQ Properties** dialog box appears.
4. On the **General** tab, select **Sybase11** from the **Service** drop-down list.
5. Select the **Database** tab and fill in the fields on this tab. For help with the fields, click **Help**.
6. Click **OK**.

8.2 Database Functions

The Sybase IQ uses a number of functions to communicate with the Sybase database. The defaults for these functions are listed below.

Note: *Unless you are directed to do so by SeeBeyond support personnel, do not change the default settings.*

1. In the Schema Designer's Component editor, select the **Services** folder.
2. Select **Sybase11** and display its properties.

Table 6 Sybase IQ Properties

	Function	Default
General	Initialize function	IQ_Sybase_Initialize
	Terminate function	IQ_Sybase_Terminate
	Reorganize function	IQ_Sybase_Reorganize
	Shared handle function	IQ_Sybase_GetSharedHandle
	Control function	IQ_Sybase_Control
Access functions	Get function	IQ_Sybase_MsgGet
	Peek function	IQ_Sybase_MsgPeek
	Put function	IQ_Sybase_MsgPut
	Mark function	IQ_Sybase_MsgMark
	Expire function	IQ_Sybase_MsgExpire
	Free function	IQ_Sybase_MsgFree
Transaction functions	Begin transaction function	IQ_Sybase_BeginTran
	Commit function	IQ_Sybase_Commit
	Rollback function	IQ_Sybase_Rollback
	Post Transaction	

8.3 External Configuration Requirements

To set up Sybase for Sybase IQs, do the following within Sybase:

1. Create a database for e*Gate IQs.
2. Create a Sybase login ID for e*Gate.
3. Set the e*Gate IQ database to be the default database for the e*Gate login.
4. Make the e*Gate login ID the owner of the e*Gate IQ database.
5. Change the properties of the IQ database such that the option **DDL in transaction** is checked (enabled).
6. Change the size of the e*Gate IQ database. The actual size required depends on the number of IQs, the size and number of Events, and the IQ cleanup schedule. We recommend you begin with an initial size of at least 150 MB.

After you have created and configured the Sybase database, you need to configure the Sybase server. Follow these steps:

1. Adjust the procedure cache. Sybase IQs make extensive use of stored procedures (approximately five per IQ); therefore, the amount of procedure cache depends on the number of IQs. We recommend you assign at least 15 MB of procedure cache.

2 Adjust the number of user connections. Because an e*Gate component that subscribes or publishes to an IQ requires its own user connection, you must assign one user connection for every subscriber and publisher in the e*Gate system that accesses a Sybase IQ.

Chapter 9

Configuring the IBM MQSeries IQ

This chapter describes how to configure the IBM MQSeries IQ.

9.1 Defining the IQ

The first step after you create an IQ component is to define the IQ Service the IQ will use. This identifies the type of IQ.

To define the IQ Service for the IBM MQSeries IQ

1. In Schema Designer, in the Navigator's **Components** tab, select the Participating Host and Control Broker of the IQ Manager whose IQs you will be configuring.
2. In the Component Editor, select the desired IQ and then edit its properties.

   The **IQ Properties** dialog box appears.
3. On the **General** tab, select **IBM_MQSeries_IQ** or **IBM_MQSeries_Client_IQ** (depending on your implementation) from the **Service** drop-down list.
4. Select the **Database** tab. Complete the following fields:

Field	Description
Database name	Type the name of the IBM MQSeries queue manager — for example, **MyQueueManager**. To enable a connection to *any* available queue manager, enter * (asterisk) in this field.
Schema name	Type the name of the IBM MQSeries stream queue — for example, **EGATE.STREAM.QUEUE**.
Login name, Password, Confirm password, Participating Host	To keep a subscription persistent in publish/subscribe mode, leave these fields blank. To cause a subscriber to unregister its subscription with the MQSeries broker during shutdown, type a **Login name** of **FALSE** (five characters, all uppercase).

Important: *When naming the queue manager and stream queue, be sure to follow the rules stated in the IBM MQSeries document MQSeries Application Programming Guide.*

5. Click **OK**.

9.2 Database Functions

The IBM MQSeries IQ uses a number of functions to communicate with the IBM MQSeries system. The defaults for these functions are listed below.

Important: *Unless you are directed to do so by SeeBeyond support personnel,* ***do not*** *change the default settings.*

1. In the e*Gate Schema Designer's Component editor, select the **Services** folder.
2. Select **IBM_MQSeries_IQ** or **IBM_MQSeries_Client_IQ** and display its properties.

Table 7 IBM MQSeries IQ Properties

Tab	Function	Default
General	Initialize function	IQ_MQM_Initialize
	Terminate function	IQ_MQM_Terminate
	Reorganize function	IQ_MQM_Reorganize
	Shared handle function	IQ_MQM_GetSharedHandle
	Control function	IQ_MQM_Control
Access functions	Get function	IQ_MQM_MsgGet
	Peek function	IQ_MQM_MsgPeek
	Put function	IQ_MQM_MsgPut
	Mark function	IQ_MQM_MsgMark
	Expire function	IQ_MQM_MsgExpire
	Free function	IQ_MQM_MsgFree
Transaction functions	Begin transaction function	IQ_MQM_BeginTran
	Commit function	IQ_MQM_Commit
	Rollback function	IQ_MQM_Rollback
	Post Transaction	

9.3 External Configuration Requirements

To enable the IBM MQSeries IQ to communicate properly with the IBM MQSeries system, the following IBM MQSeries objects must be created prior to running the IQ in e*Gate:

- **Queue manager**

- **Stream queue** — used by an e*Gate publisher to send Events to the IBM MQSeries Publish/Subscribe broker.
- **e*Gate subscriber model queue** — used as a template to create e*Gate MQSeries IQ queues
- **e*Gate dead-letter queue** — holds undeliverable Events sent to it by the IBM MQSeries Publish/Subscribe broker or the IBM MQSeries queue manager
- **IBM MQSeries server connection channel** (client mode only)

The stream queue, e*Gate subscriber model queue, and e*Gate dead-letter queue are IBM MQSeries queues which are required by the e*Gate system.

9.3.1 Naming the Queues

The IBM MQSeries queue manager name and stream queue name should correspond to the names used during your IQ configuration; see the **procedure on page 52**. The e*Gate subscriber model queue name must be EGATE.SUBSCRIBER.MODEL.QUEUE. The name of the dead-letter queue can be any name of your choice.

Note: *You must identify the name of the dead-letter queue in the queue manager.*

Naming Restrictions

IBM MQSeries imposes several rules on object names. The names you choose for the queue manager and the queues need to conform to the following character set. For details on IBM MQSeries naming restrictions see Chapter 4 of the IBM MQSeries document *MQSeries Application Programming Guide*.

The character set that can be used for naming all IBM MQSeries objects is:

- uppercase A-Z
- lowercase a-z
- numerics 0-9
- underscore (_)

The maximum length of an IBM MQSeries queue name is 48 characters. However, since e*Gate will add the e*Gate schema name to a queue name (see **"IBM MQSeries Queues Dynamically Created by e*Gate" on page 58**), the maximum length of your e*Gate queue name must be less than 48 minus the schema name length plus one, as in the following equation:

*maximum length of e*Gate queue name < 48 - (length of schema name + 1)*

For example, if the name of your schema is **Production** (10 characters), the maximum number of characters available for the e*Gate queue name is 37 (= 48 - (10+1)).

9.3.2 Creating e*Gate-Required Objects on the Server

SeeBeyond provides a script to facilitate the creation of the three e*Gate-required queues and the server connection channel on the IBM MQSeries server. However, you must manually create the IBM MQSeries queue manager first. The name of the queue manager must correspond to the database name you entered in the IQ Configuration dialog box in the Schema Designer. (See the **procedure on page 52**.)

To create the IBM MQSeries objects on the server

1. Create and start the queue manager.
2. Start the MQSeries Publish/Subscribe broker.
3. Run the MQSC script **stcmqiq.mqsc**.

   The script is on the installation CD-ROM in the **\utils\mqseries** directory. To run the script, at a command prompt type:

   ```
   runmqsc queue_manager < stcmqiq.mqsc
   ```

   where *queue_manager* is the name of the queue manager you created in step 1.

If the script is executed successfully, the following objects will be created:

- EGATE.STREAM.QUEUE, a stream queue
- EGATE.SUBSCRIBER.MODEL.QUEUE, a subscriber model queue
- EGATE.DEAD.LETTER.QUEUE, a dead-letter queue
- EGATE_CHANNEL, a server connection channel

You should modify the queue properties to suit your specific requirements.

9.3.3 Defining the Client Connection Channel

Important: *The IBM MQSeries IQ client mode implementation uses MQCONN to connect to the MQSeries server. All limitations that apply to MQCONN apply to the IQ in client mode.*

If you are using the IBM MQSeries IQ in client mode, before running data through the IQ you must complete two tasks:

1. Start the IBM MQSeries listening program on the server and specify the queue manager to which the listening program will connect.
2. Define a client connection channel.

There are two ways to define the client connection channel:

- Define the MQSERVER environment variable on the client machine.
- Create a client connection channel on the server machine and copy the channel definition table to the client machine.

Defining the Client Connection Channel with the MQSERVER Environment Variable

If you use the MQSERVER environment variable to define the client connection channel, only one client connection channel is available to the e*Gate modules. In this case, all IQs in an e*Gate schema must define the same IBM MQSeries queue manager (the value in the **Database name** field in the **IQ Properties** dialog box), and the queue manager must be the one specified by the IBM MQSeries listening program.

To define the MQSERVER environment variable in Windows XP or Windows 2000:

- At a command prompt, type

  ```
  SET MQSERVER=EGATE_CHANNEL/TCP/server_name
  ```

  where *server_name* is the IP address of the server machine.

To define the MQSERVER environment variable in UNIX:

- At a command prompt, type

  ```
  setenv MQSERVER EGATE_CHANNEL/TCP/server_name
  ```

  where *server_name* is the IP address of the server machine.

Client channels defined by the MQSERVER environment variable can handle messages with a maximum size of 4 MB. For messages larger than 4MB, you must define the client connection channel by creating the channel on the server and copying the channel definition table to the client machine.

Defining the Client Connection Channel on the Server

If you define the client connection channel on the server you can utilize multiple IBM MQSeries queue managers in an e*Gate system. In this case, you should define the client connection channels at one of the queue managers and then copy the channel definition table to all client machines. The queue manager field for each of the client connection channels must correspond to the IBM MQSeries queue manager specified for each IQ component in your e*Gate schema (the value in the **Database name** field in the **IQ Properties** dialog box).

To define the client connection channel on the server:

1. Use the DEFINE CHANNEL command to create a client channel on the IBM MQSeries server. The name of the channel must match the name of the server connection channel defined in the **stcmqiq.mqsc** script.

Important: *The default name for the server connection channel is EGATE_CHANNEL. If you are utilizing multiple queue managers you should modify this name appropriately; see* **"Example Configuration for Multiple IBM MQSeries Queue Managers" on page 57**.

2. Set the MAXMSGL parameter of both the client and server connection channels to increase the maximum message size, if necessary.

3 Copy the channel definition file AMQCLCHL.TAB (a binary file) from the server to the client machine or make the file on the server available to the client machine. By default the file is in

\mqm\qmgrs*queue_manager*\@ipcc (Windows XP or Windows 2000)

/var/mqm/qmgrs/*queue_manager*/@ipcc (UNIX)

where *queue_manager* is the name of the queue manager.

4 On the client machine, set the environment variable MQCHLLIB to specify the path to the channel definition table.

For more information on how to create IBM MQSeries channels, see the *IBM MQSeries Client System Administration Guide*.

Example Configuration for Multiple IBM MQSeries Queue Managers

The following example illustrates the use of multiple IBM MQSeries queue managers in an e*Gate system.

1 Log into the e*Gate Integrator Schema Designer and open a schema. Create two IQ components, **IQ1** and **IQ2**. Define the IQ properties as follows:

Table 8 Properties for IQ1

Service	IBM_MQSeries_Client_IQ
Database name	Queue_Mgr_1

Table 9 Properties for IQ2

Service	IBM_MQSeries_Client_IQ
Database name	Queue_Mgr_2

2 On the IBM MQSeries server machine, create two IBM MQSeries queue managers. Name them **Queue_Mgr_1** and **Queue_Mgr_2**.

3 Start the queue managers on the IBM MQSeries server.

4 Open the script file **stcmqiq.mqsc** in a text editor. (This file is located on the installation CD-ROM in the **\utils\mqseries** directory.) Under "* Step 10 - Define a server connection channel," edit the following line:

Old	New
DEFINE CHANNEL('EGATE_CHANNEL') +	DEFINE CHANNEL('EGATE_CHANNEL_1') +

5 Save and close the file.

6 Run the script to create the e*Gate-required queues and a server channel for the queue manager **Queue_Mgr_1**. (See **"Creating e*Gate-Required Objects on the Server" on page 55**.)

7 Repeat steps 4 and 5 and change the name of the server channel to: **EGATE_CHANNEL_2**

8 Run the script again to create a second server channel for the queue manager **Queue_Mgr_2**.

9 Use the IBM MQSeries command **runmqsc** to create the client channels on **Queue_Mgr_1**. Issue the following two commands:

```
DEFINE CHANNEL(EGATE_CHANNEL_1) CHLTYPE(CLNTCONN) TRTYPE(TCP) +
CONNAME(IP_address(port)) QMNANE("Queue_Mgr_1")

DEFINE CHANNEL(EGATE_CHANNEL_2) CHLTYPE(CLNTCONN) TRTYPE(TCP) +
CONNAME(IP_address(port)) QMNANE("Queue_Mgr_2")
```

where *IP_address* is the IP address of the IBM MQSeries server machine and *port* is the port number for the listening program for each queue manager.

Note: *If the queue managers are running on the same machine, the port numbers must be different for each queue manager.*

10 Start the listening program for each queue manager by issuing the following commands:

```
runmqlsr -t tcp -m Queue_Mgr_1 -p port

runmqlsr -t tcp -m Queue_Mgr_2 -p port
```

where *port* is the port number used by the listening program for that queue manager.

11 Copy the channel definition file AMQCLCHL.TAB (a binary file) from the server to the client machines or make the file on the server available to the client machines.

12 Set the MQCHLLIB environment variable on all client machines.

13 When you incorporate the IQs into the schema and start the e*Gate modules, you can verify that they are accessing the two IQs managed by the two IBM MQSeries queue managers.

Caution: *When you are using the IQ in the simple publish/subscribe mode, you must start the MQSeries broker and all e*Gate modules that will access the MQSeries IQ* ***before*** *sending any data through the system. Make sure to start the MQSeries broker first, then the modules, to ensure that the modules get registered with the broker.*

9.4 IBM MQSeries Queues Dynamically Created by e*Gate

The following IBM MQSeries queues are dynamically created when you start all e*Gate modules defined in your schema:

- e*Gate queues. The names are of the form *SchemaName.QueueName*.
- Publisher reply-to queues. The names of these queues are of the form of the e*Gate publisher UUID with the brackets and hyphens replaced with periods. For example, if the UUID of a publisher inside a Collaboration is **{BB7624A1-88A3-11D3-9AB2-AC5E95C53DAF}**, the corresponding reply queue name would be **.BB7624A1.88A3.11D3.9AB2.AC5E95C53DAF.**

9.5 IBM MQSeries Commands

Following are some frequently-used IBM MQSeries commands to assist you in creating the necessary IBM MQSeries objects. For a complete list of all available commands, see the *IBM MQSeries System Administration Guide*.

IBM MQSeries Server Commands

crtmqm *queue_manager*
Creates a queue manager where *queue_manager* is the name of the queue manager.

strmqm *queue_manager*
Starts a queue manager where *queue_manager* is the name of the queue manager.

strmqbrk -m *queue_manager*
Starts a broker for a queue manager where *queue_manager* is the name of the queue manager.

runmqlsr -t tcp -m *queue_manager* [-p *port_number*]
Starts the listening program on the server, where *queue_manager* is the name of the queue manager and *port_number* is the port number the listening program will use. The port number is optional. If you supply a port number, it will overwrite the default port number of 1414.

runmqsc -m *queue_manager*
Starts IBM MQSeries Commands where *queue_manager* is the name of the queue manager.

DEFINE CHANNEL(*channel_name*) CHLTYPE(SVRCONN) TRPTYPE(TCP)
In IBM MQSeries Commands, defines a server connection channel where *channel_name* is the name of the channel. Note that the parentheses are part of the command.

DEFINE CHANNEL(*channel_name*) CHLTYPE(CLNTCONN) TRPTYPE(TCP) + CONNAME(*IP_address*) QMNAME(*queue_manager*)
In IBM MQSeries Commands, defines a client connection channel where *channel_name* is the name of the channel, *IP_address* is the IP address of the IBM MQSeries server, and *queue_manager* is the name of the queue manager. Note that the parentheses are part of the command.

IBM MQSeries Client Commands (UNIX)

setenv MQSERVER EGATE_CHANNEL/TCP/*IP_address* (*port number*)
Defines a simple client channel called EGATE_CHANNEL where *IP_address* is the IP address of the IBM MQSeries server and *port_number* is the port number the listening program will use. Note that the parentheses are part of the command. If the port number is not specified, the default port number 1414 is used.

IBM MQSeries Client Commands (Windows XP or Windows 2000)

set MQSERVER=EGATE_CHANNEL/TCP/*IP_address*(*port_number*)

Defines a simple client channel called EGATE_CHANNEL where *IP_address* is the IP address of the IBM MQSeries server and *port_number* is the port number the listening program will use. Note that the parentheses are part of the command. If the port number is not specified, the default port number 1414 is used.

Chapter 10

Administering IQs

This chapter describes how to use the IQ Administrator application to monitor and manipulate the status of IQs and the Events they contain.

10.1 Overview

IQ Administrator (formerly called "Queue Viewer") is a special-purpose application accessed from Schema Manager. This application provides real-time information on IQs and their Events.

Supported IQs

The IQ Administrator application monitors SeeBeyond Standard IQs—that is, IQs using the STC_Standard IQ Service. JMS Administrator is a similar tool for monitoring SeeBeyond JMS IQs; for information on JMS Administrator, see the *SeeBeyond JMS Intelligent Queue User's Guide*.

Note: *For sites that use ACL security, the minimum privileges to allow a User to run Schema Manager and start IQ Administrator are* ***IQ View*** *and* ***Module View*** *(for example, a* ***Monitor*** *role). However, to allow a User to manipulate IQs and Events requires* ***Module Edit*** *privileges (for example, a* ***Module*** *or* ***Operations*** *role).*

10.1.1 Purpose and Features

IQ Administrator allows you to do any of the following:

- Within the current schema, browse an expandable/collapsible hierarchy view of IQ Managers, IQs, publishers, Event Types, subscribers, and individual Events.
- View the status and contents of each IQ, with color-coding of each Event's state: white Events are Unread by subscriber; blue Events are Read by subscriber; yellow Events are Journaled.
- See a summary of Events for a specified Event Type or subscriber.
- Query the details of any individual Event, such as its size, timestamp, and priority.

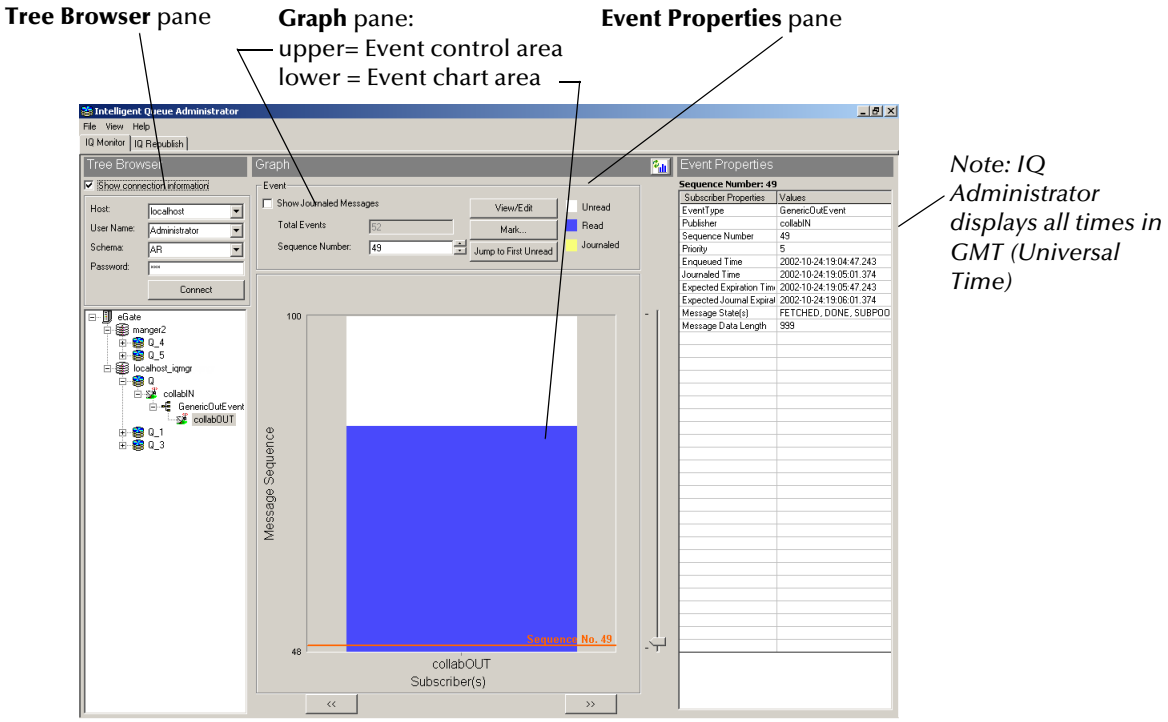
Note: *Because IQ Administrator can be used simultaneously in many time zones, its timestamps are in GMT (Universal Time), unlike log file timestamps, which are in local time.*

- Edit or query the status of an individual Event or range of Events, so that you can locate an Event that is causing a problem, diagnose and possibly solve the problem, and cause it to be deleted or re-sent.

10.1.2 Graphical User Interface (GUI)

The IQ Administrator presents information in a four-pane layout illustrated in Figure 8.

Figure 8 GUI Map for IQ Administrator



GUI Panes, Areas, and Controls

Tree Browser pane

- **Show connection information** check box—When selected (as shown in Figure 8), this provides a box showing continuous display of the current Registry Host, user, and schema, and it allows you to re-connect at any time. When cleared (the default), it hides the box and causes the **Refresh Tree Browser** button to display instead.
- **Refresh Tree Browser** button (not shown in Figure 8)—Clicking this button updates the Tree Browser display and collapses the tree view.
- **Tree view**—This provides an Explorer-like view of the current schema that allows you to expand and collapse the hierarchy of IQ Managers, IQs, publishers, Event Types, and subscribers.

When you select an Event Type or subscriber, the **Graph** pane and **Event Properties** pane are populated with information on Events contained in the current selection.

Graph pane

- **Refresh Graph** button—Clicking this updates the display of the **Graph** pane for the current Event Type or subscriber. The button is available only when the current selection in the Tree Browser pane is an Event Type or subscriber.
- **Event control area**—This area contains controls that allow you to specify whether to display journaled Events, to view/edit individual Events, and to mark or delete individual Events or ranges of Events.
- **Event chart area**—This area displays one or more bar charts showing the range of sequence numbers and state (Read/Unread/Journaled) of Events contained in the current Event Type or subscriber. It also contains controls for going forward or back through several subscribers and setting the focus up or down within a sequence.

For detailed information on the **Graph** pane, see **"Using Event Charts" on page 65**.

Event Properties pane

The top of the **Event Properties** pane shows you which Event currently has focus.

The remainder of the **Event Properties** pane is a table showing you the current Event's properties and their values; see **"Event Properties" on page 68**.

Resize IQ Administrator controls

The four arrows at the lower right of IQ Administrator allow you to shrink or stretch the display horizontally or vertically.

10.2 Using IQ Administrator

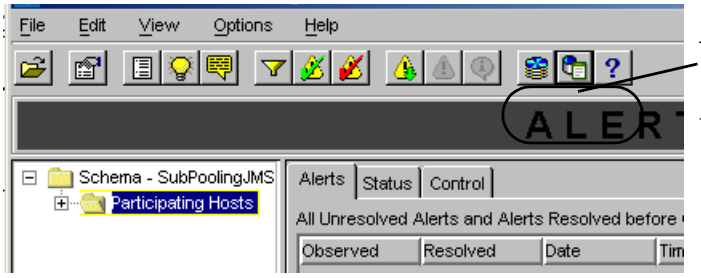
You access IQ Administrator via a toolbar button of Schema Manager.

To start IQ Administrator

1 Start Schema Manager and log in to the schema you want to monitor.

The toolbar buttons for the IQ Administrators are on the far right. See Figure 9.

Figure 9 The Schema Manager Toolbar



2 On the toolbar, click IQ Administrator .

The IQ Administrator application starts. See **Figure 8 on page 62**.

10.2.1 Connecting or Reconnecting to a Host or Schema

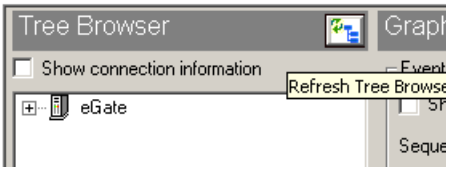
When the **Show connection information** check box is selected, you can easily refocus the current session of IQ Administrator onto a new Registry Host or a new schema, and/or re-establish a connection to the current schema. When the check box is cleared, the **Refresh Tree Browser** button allows you to update the current connection.

To update the current connection

- In the Tree Browser pane with the **Show connection information** check box cleared, click the **Refresh Tree Browser** button.

  The connection is updated and the tree is collapsed; see Figure 10.

Figure 10 Show Connection Information (check box cleared)

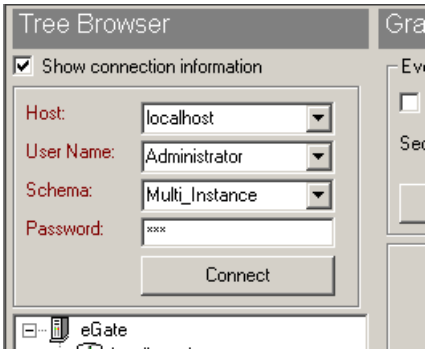


To establish a new connection

1. In the Tree Browser pane, select the **Show connection information** check box.

   Current connection information is shown; see Figure 11.

Figure 11 Show Connection Information (check box selected)



2. As needed, enter or select new values for Registry Host, user name, schema name, and password.
3. Click **Connect**.

   If you made no alterations in step 2, the connection is re-established; otherwise, a new connection is attempted using the parameters you specified.

10.2.2 Listing IQ Managers and Their Contents

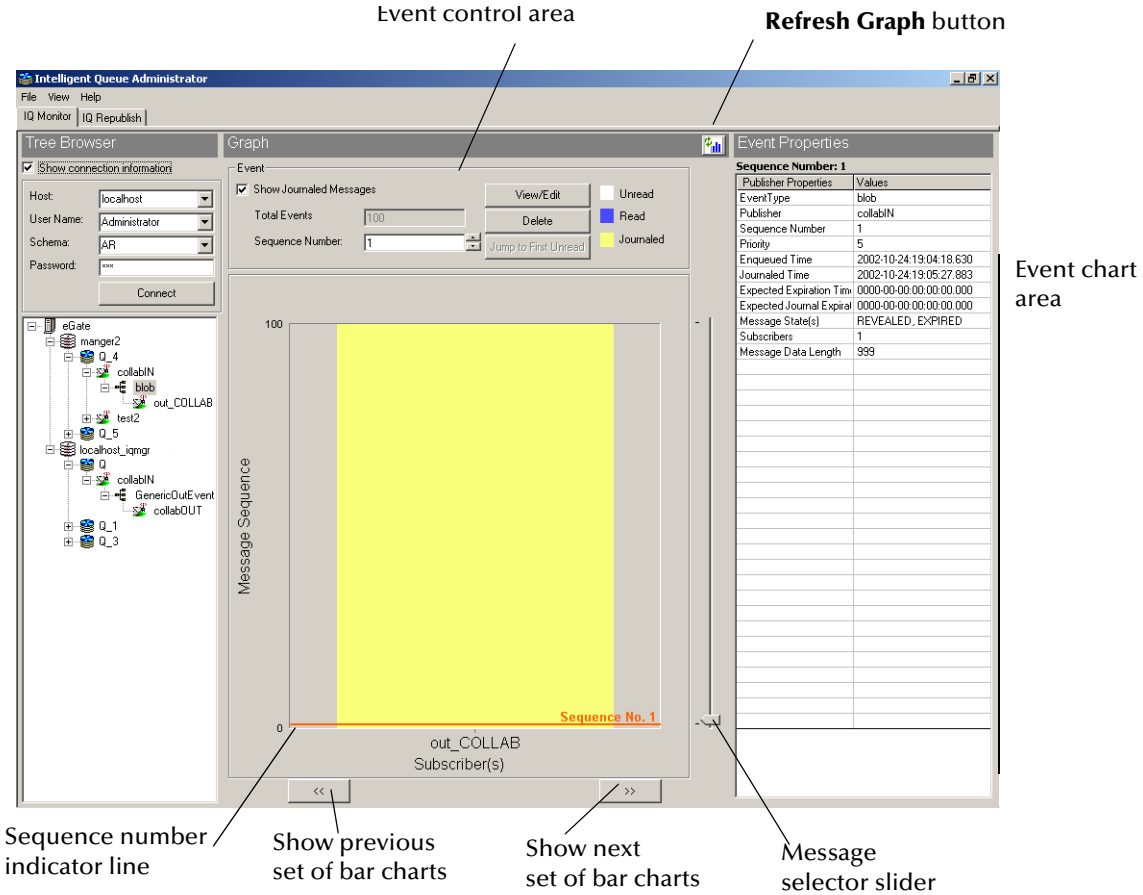
Opening the **eGate** folder displays a list of all IQ Managers for the current schema; opening an IQ Manager displays a list of all the IQs it manages; opening an IQ displays a list of publishers (in others words, the Collaborations that publish to the IQ); opening a publisher displays a list of the Event Types it publishes; and opening an Event Type displays a list of the subscribers (in other words, the Collaborations that subscribe that the Event Type).

10.2.3 Using Event Charts

The **Graph** pane provides a graphical display of the aggregate Events contained in the selected Event Type or subscriber. Within the **Graph** pane, the Event control area specifies the type and range of Events and provides access to the viewing/editing and deleting/marking controls, while the Event chart area displays Event status.

- When an Event Type is selected, the buttons in the Event control area allow you to view or edit an individual Event or to delete an individual Event or range of Events. A bar chart is provided for each subscriber of that Event Type.

Figure 12 Event Charts for an Event Type with Two Subscribers



- When a subscriber is selected, the buttons in the Event control area allow you to view or edit an Event, to mark an individual Event or a range, or to set the sequence number indicator line on the first unread Event for this subscriber.

Note: *If you have selected the* ***Do not store journalled*** *checkbox in the IQ Manager properties, you cannot view journaled Events under the subscribing Collaboration. You can view these Events by clicking the Event Type icon.*

Special Considerations for Subscriber Pooling

For information on concepts and procedures, see **"Subscriber Pooling" on page 31**.

When an IQ has been configured to use subscriber pooling, each Event is *processed* by one and only one subscriber, but the Event's existence and sequence number can be *seen* by **all** subscribers. When the Event is processed by the single destination subscriber, its appearance on the Event chart indicates that it has been processed by all subscribers. When viewing at the subscriber level, you can view the Event state and payload if and only if you are viewing the subscriber processing the Event (the **SUBPOOL** subscriber). For non-receiving subscribers, the message **No message available** is displayed.

To specify the type and range of Events to chart

1. In the **Tree Browser** pane, click the Event Type or subscriber whose Events you want to chart.

   In the **Graph** pane, the Event chart area displays a bar chart of Events for the selected Event Type or subscriber.
2. If you want to chart journaled messages as well as read and unread messages, in the Event control area, select the **Show Journalled Messages** check box.
3. To specify a range of Events, in the **Sequence Number** boxes, select or type lower and upper bounds for the range.
4. To select a particular Event, do one of the following:
   - In the **Sequence Number** boxes, select or type the Event's sequence number for both the lower and upper bound.
   - Drag the Message Selector slider up or down to the correct sequence number.
   - Click **Jump to First Unread** to place the indicator on the lowest boundary between a blue area (representing Events that have been read by the selected subscriber) and a white area (representing one or more Events that have not yet been read by this subscriber).

To delete an Event or a range of Events

1. Click an Event Type and specify an Event or range using the **procedure on page 66**.
2. Click **Delete**.
3. The system asks you to confirm the selection.
4. Double-check that the Event number or range is correct and then click **Yes**.

Note: *When subscriber pooling is in effect, all subscribers display all Events in the IQ, but you are only able to delete Events for the subscriber that is receiving them. See* **"Special Considerations for Subscriber Pooling" on page 66***.*

To view the content of an Event

1. Use the **procedure on page 66** to specify the Event you want to view.
2. Click **View/Edit**.

   A text editor window displays the content of the Event. If you specified a range, the editor displays the content of the highest-numbered Event only.
3. When you have finished browsing the Event's text, close the text editor window.

To edit the content of an Event and re-send it

1. Use the **procedure on page 66** to specify the Event you want to edit.
2. Click **View/Edit** and use the text editor to make additions, deletions, or modifications as needed.
3. When you have finished editing the Event's text, close the text editor window.
4. In response to questions from the text editor and the IQ Administrator asking you to confirm your changes, click **Yes**.

   After you edit and re-send an Event, it is reclassified as Unread for this subscriber, and its Message State property for this subscriber is set to CLEAN.

Note: *When you edit and re-send an Event, its sequence number is preserved in all cases. In Standard IQs, edited Events are retrieved by publishers in the same order as if they had not been edited.*

To change the status of an Event or range of Events

1. Click a subscriber and specify an Event or range using the **procedure on page 66**.

Note: *When subscriber pooling is in effect, all subscribers display all Events in the IQ, but you can change the status of Events only for the subscriber that is receiving them. See* **"Special Considerations for Subscriber Pooling" on page 66***.*

2. Click **Mark**.
3. In the **Mark a message** dialog box, set the status of this Event or range to either Unread or Read (for the selected subscriber) and then click **Mark It**.
4. In response to the system prompt, click **Yes** to confirm the changed status.

10.3 Event Properties

10.3.1 Overview

From the standpoint of an IQ or publisher, an Event can be in only one of two states:

- *Active* Events (REVEALED status) are published but not yet marked as journaled, and are waiting for one or more subscribers to retrieve the Event and mark it DONE. An active Event is marked as journaled by cleanup after all subscribers mark it DONE or after its expiration time elapses, whichever comes first.
- *Journaled* Events are Events that are no longer active, either because they were picked up and marked DONE by all their subscribers or because they expired. If a journaled Event remains journaled past its expiration time, it is deleted when you run an IQ cleanup operation.(See **"Configuring IQ Cleanup" on page 32**.)

From the standpoint of a subscriber, an Event can be either Unread (CLEAN status), retrieved and being processed (FETCHED status), or retrieved and processed (FETCHED,DONE status).

10.3.2 Property Names and Values

Table 10 Event Properties

Property Name	Type	Description
Event Type	text string	The Event Type containing the publisher(s) of the Event, from the e*Gate schema.
Publisher	text string	The name of the publisher of the Event, from the e*Gate schema.
Sequence Number	integer	Where in the queue this Event falls. Higher numbers represent Events that are later in the queue.
Priority	integer	Relative ranking of the importance of this Event, from 0(most urgent, to be processed before any priority-2 Events) to 999,999,999 (least urgent, to be processed only after all other events have been processed). Default: **5**
Enqueued Time	date/timestamp	When the Event was placed on the queue (in GMT).
Journaled Time	date/timestamp	When the Event was journaled (in GMT).
Expected Expiration Time	date/timestamp	When the Event was (or will be) marked as being expired (in GMT).
Expected Journal Expiration Time	date/timestamp	When the journal for the Event is next scheduled for cleanup (in GMT).
Message State(s)	text string	See Table 11 **"Message States" on page 69**.

Table 10 Event Properties (Continued)

Property Name	Type	Description
Subscribers	integer	How many Collaborations are subscribing to Events of the selected Event Type.
Message Data Length	integer	How many bytes the Event contains.

Table 11 Message States

Message State	Explanation
CLEAN	The selected subscriber has not retrieved the Event.
DONE	The selected subscriber has read and processed the Event.
EXPIRED	Either or both of the following: ▪ For the publisher of this Event Type, the elapsed time since the Event was enqueued has exceeded the set expiration time. ▪ The Event has been fully processed by all subscribers.
FETCHED	The selected subscriber has picked up and read the Event but has not finished processing it.
FETCHED,DONE	The selected subscriber has picked up, read, and processed the Event.
REVEALED	For the publisher of this Event Type, the Event has been made available for subscribers to retrieve.

10.4 Using stciqutil

e*Gate includes an IQ utility, **stciqutil.exe**, that enables you to view and manipulate an IQ's contents outside of the e*Gate environment. This section explains how to use **stciqutil** to perform basic tasks, but it does not include a comprehensive list of all qualifiers and available flags. For a complete reference on this utility's command-line options, see the *e*Gate Integrator System Administration and Operations Guide*.

Important: *Because of the complexity of IQ storage, we strongly recommend that you only use the SeeBeyond IQ utilities for interfacing with the IQs.*

Caution: *Be extremely careful if you use this command to manipulate and reload IQ data. Errors can cause e*Gate to process the queue data incorrectly, and may cause other consequences in the systems that receive the processed e*Gate data.*

Usage

stciqutil *command-flags @command-file*

Where *command-flags* (separated by spaces) are one of those shown in Table 12 in this section, and *command-file* is an optional ASCII text file containing command flags and their arguments. Table 12 shows the **stciqutil** command arguments.

Table 12 Command Arguments for stciqutil

Flag	Purpose	Required
-h	Displays a help message.	
-v	Verbose mode; shows additional information as commands are processed.	
--ver	Displays version information. Note that this flag requires two dashes.	
-d	Debug log on.	
-rh *host-name*	Registry host.	Yes
-rs *schema-name*	Registry schema name (If not specified, "default" schema is used).	Yes
-rp *Registry-port*	Specifies the Registry port that this command affects.	
-un *user-name*	User name as defined within the specified schema.	Yes
-up *password* **or** !*directory-name*	Password for the specified user name.	Yes
-iq *IQ-name*	Name of the IQ as defined within the specified schema.	Yes
-cnt	Count the messages in the IQ. When used with **-qd**, the count is output to a **.cnt** file.	
-ar	Dump only journaled Events (requires **-qd** or **-cnt**).	Only use this flag for SeeBeyond Standard IQs
-live	Allows users to work with live (active) Events only, that is, Events that have not been fetched and marked "done."	
-fnd	Dump only Events marked "fetched" but not marked "done" (requires **-qd** or **-cnt**).	
-qd	Dump IQ.	
-od *directory*	Directory into which to dump output information.	
-dt *date-range*	Dump date range, in the format YYYY-MM-DD-HH-mm-SS-sss-YYYY-MM-DD-HH-mm-SS-sss	
-ma *range*	Dump Major sequence number range (for example, 100-900).	
-mi *range*	Dump Minor sequence number range (for example, 001-999).	

Table 12 Command Arguments for stciqutil (Continued)

Flag	Purpose	Required
-pub *publisher-name*	Dump Events only from this publisher.	
-event *event-name*	Dump Events of only this Event Type.	
-nosubs	Do not filter messages based upon the subscriber.	
-ld	Reload the IQ.	
-id *directory-name*	Directory from which to reload the IQ.	
-sub *subscriber-name*	Subscriber name(s); a list of subscribers separated by commas (no spaces), ALL (for subscribers extant in the schema at reload time) or ALLORIG (for subscribers specified in the IQ dump files).	
-cfg *path*	File path of the configuration file used to substitute for the Registry.	
-mm	Sequence numbers first in the dump/reload file names.	
-keys	Shows all keys (Major and Minor) associated with the current IQ, to **stdout** (monitor display); requires the command's necessary flags plus -iq, -pub, and -event.	
-mark DELETED	Allows you to mark a single Event as deleted. The DELETED argument is required. The following flags are also required: ♦ **-dt**: Must be an exact enqueue time. ♦ **-ma**: Must be an exact Major sequence number. ♦ **-mi**: Must be an exact Minor sequence number. In this context, you cannot use a range with the flags in the previous list.	

10.4.1 Dumping the Contents of an IQ

When you use **stciqutil** to dump the contents of an IQ, it creates four files per Event, using the following naming convention:

IQname-YYYY-MM-DD-HH-mm-SS-sss-NNNNN-nnnnn.ext

where

IQname is the name of the IQ (as defined within the e*Gate Schema Designer)
YYYY-MM-DD-HH-mm-SS-sss is the enqueue time
NNNNN is the major sequence number
nnnnn is the minor sequence number
ext is one of the following: **att** (attributes), **hdr** (header), **pth** (path blob), or **evt** (Event data).

Generally, only the .**att** file is human-readable, unless the queued Event was in plain ASCII text (in which case the .**evt** file will also be readable).

e*Gate imposes no limit on the size of the IQ that can be dumped; however, you must have sufficient disk space to contain both the IQ data and the IQ dump.

To dump an IQ

- At a command prompt, type the following (as a single command line):

```
stciqutil -rh hostname -rs Schemaname -un user -up passwd
     -event Ename -iq Qname -pub Pubname -od dirname -qd
```

where

hostname is the name of a Registry Host
schemaname is the name of a schema
user is the name of an e*Gate user
passwd is the password for that user name
Ename is the name of the Event Type whose Events you want to dump
Qname is the name of the IQ to dump
Pubname is the name of the Event Type's publishing Collaboration
dirname is the name of the directory into which to dump the IQ

To dump only Events that have been journaled, add the **-ar** switch to the IQ-dump command line.

Note: *Dumping an IQ only copies the contents of an IQ; it does not "empty" the IQ of its contents.*

10.4.2 Displaying IQ Event Keys

You can display all of the major and minor Event keys associated with an IQ. The output is printed to your screen and is not saved in a file. You can use the output from this operation as an argument to other command flags, such as the **-dt** option.

To display the Event keys associated with an IQ

- At the command prompt, type the following (as a single command line):

```
stciqutil -rh hostname -rs Schemaname -un user -up passwd
     -event Ename -iq Qname -pub Pubname -keys
```

where

hostname is the name of a Registry Host
schemaname is the name of a schema
user is the name of an e*Gate user
passwd is the password for that user name
Ename is the name of the Event Type for whose Events you want to display keys
Qname is the name of the IQ for whose Events you want to display keys
Pubname is the name of the Event Type's publishing Collaboration

10.4.3 Reloading an IQ

When you reload an IQ, you put new Events or Events you had dumped into a directory back into the IQ. The dumped Events will be inserted into the queue as unique, new Events with different enqueue times than the originals.

Caution: *Reloading Events does not overwrite the original Events in the IQ. For information on deleting the original Events, see* ***"Deleting Events from an IQ"****, below, or see*

*the **Monk Developer's Reference** for information regarding the Monk command **iq-mark-unusable**.*

To reload an IQ

- At the command prompt, type the following (as a single command line):

```
stciqutil -rh hostname -rs Schemaname -un user -up passwd
     -event Ename -iq Qname -sub Subname -id dirname -ld
```

where

hostname is the name of a Registry Host
schemaname is the name of a schema
user is the name of an e*Gate user
passwd is the password for that user name
Ename is the name of the Event Type whose Events you want to reload
Qname is the name of the IQ you want to reload
Subname is the name of the Event Type's subscribing Collaboration(s)
dirname is the name of the directory from which to reload the IQ

10.4.4 Deleting Events from an IQ

If you are reloading an IQ and you want e*Gate to use the new Events in place of the original Events that were dumped, you should delete the existing Events ***before*** you perform the reload operation. If you perform the deletion procedure after reloading an IQ, both the original and the reloaded Events will be marked "deleted."

The **stciqutil** command flag **-mark DELETED** marks only a single Event as deleted at a time. To delete multiple Events, you can run a shell script that runs the **stciqutil** command multiple times.

Note: *The command flag **-mark DELETED** marks an Event as deleted in the IQ and make it unavailable for processing. The Event is not physically deleted from the IQ until the next IQ cleanup is run.*

To delete a single Event from an IQ

- At the command prompt, type the following (as a single command line):

```
stciqutil -rh hostname -rs Schemaname -un user -up passwd
-iq Qname -pub Pubname -event Ename -mark DELETED -ma MAnumber
-mi MInumber -dt Etime
```

where

hostname is the name of a Registry Host
schemaname is the name of a schema
user is the name of an e*Gate user
passwd is the password for that user name
Qname is the name of the IQ that contains the Event
Pubname is the name of the Event's publishing Collaboration
Ename is the name of the Event Type of the Event
MAnumber is the major sequence number of the Event

MInumber is the minor sequence number of the Event
Etime is the enqueue time for the Event

You can obtain the major sequence number, minor sequence number and enqueue time for an individual Event in a queue by obtaining the Event keys for the IQ. (See **"Displaying IQ Event Keys" on page 72**.)

Example

The following example demonstrates how to delete an Event from an IQ. The example uses the following information:

- Registry name: **Host**
- Schema name: **Tester**
- Username: **Administrator**
- Password: **STC**
- Publishing Collaboration name: **tester_Pub**
- Event Type name: **blob**
- IQ name: **tester_IQ**

Note: *To obtain the IQ and Event Type names, select the publishing Collaboration and view its properties. Then, select the* ***Publications*** *tab and identify the information there.*

1 To obtain the Event keys for the IQ, type the following at a command prompt:

```
stciqutil -rh Host -rs Tester -un Administrator -up STC
-iq tester_IQ -event blob -pub tester_Pub -keys
```

This will produce the following output (simulated in this example):

```
-----------------------------------------------------------------
 IQ Keys
 Publisher  : tester_Pub
 Event Type : blob
 IQ         : tester_IQ
-----------------------------------------------------------------
 Major          Minor       Enqueue Time            Priority
-----------------------------------------------------------------
  0              1           09282000:18:47:34:984   0
  0              2           09282000:18:47:35:046   0
  0              4           09282000:18:47:35:171   0
  0              5           09282000:18:54:44:015   0
  0              6           09282000:18:54:44:062   0
  0              7           09282000:18:54:44:156   0
  0              8           09282000:18:54:44:265   0
-----------------------------------------------------------------
```

2 To mark the seventh Event as deleted, type the following at a command prompt:

```
stciqutil -rh Host -rs Tester -un Administrator -up STC
-iq tester_IQ -pub tester_Pub -event blob -mark DELETED
-ma 0 -mi 7 -dt 09282000:18:54:44:156
```

3 To verify that the Event has been marked deleted, display the Event keys for the IQ again.

```
stciqutil -rh Host -rs Tester -un Administrator -up STC
-iq tester_IQ -event blob -pub tester_Pub -keys

 ---------------------------------------------------------------
  IQ Keys
  Publisher  : tester_feeder_Pub
  Event Type : blob
  IQ         : tester_IQ
 ---------------------------------------------------------------
  Major          Minor       Enqueue Time            Priority
 ---------------------------------------------------------------
    0              1          09282000:18:47:34:984   0
    0              2          09282000:18:47:35:046   0
    0              4          09282000:18:47:35:171   0
    0              5          09282000:18:54:44:015   0
    0              6          09282000:18:54:44:062   0
    0              8          09282000:18:54:44:265   0
 ---------------------------------------------------------------
```

This verifies that minor sequence Event number 7 has been deleted.

10.4.5 Counting Events in an IQ

To count the number of Events of a particular Event Type in an IQ

- At the command prompt, type the following (on a single command line):

```
stciqutil -rh hostname -rs Schemaname -un user -up passwd
     -event Ename -iq Qname -pub Pubname -cnt
```

where

hostname is the name of a Registry Host
schemaname is the name of a schema
user is the name of an e*Gate user
passwd is the password for that user name
Ename is the name of the Event Type whose Events you want to count
Qname is the name of the IQ from which you want to obtain the count
Pubname is the name of the Event Type's publishing Collaboration

10.5 Verifying and Recovering IQs

If you suspect a standard IQ has become corrupted, you have several tools for verifying and recovering data, using the command-line utilities **stciqstdutil.exe** and **stciqutil.exe**.

Important: *Because of the complexity of IQ storage, we strongly recommend that you only use the SeeBeyond IQ utilities for interfacing with the IQs. For complete information on* ***stciqstdutil.exe****, see the procedure below and* **"Running stciqstdutil.exe" on page 77***.*

To verify that a queue is intact

1 Run **stciqstdutil.exe** against the index file (**.rdb** file) you suspect. For example, after learning the name of the **.rdb** file, you would enter the following (on all one line):

```
stciqstdutil.exe -ip egate/client/iq/{3E2D28EE-010B-11D4-
  BE68-B62F6823CE53}
  -n qlive_{71A66202-7D35-11D4-9271-E2D1B5816FCD}.rdb
```

If the utility can dump an entire index file, it displays the following message:

```
Wrote <n> record(s) to file: <dump_file_name>
```

2 Repeat against any other index files you suspect.

If the utility is able to dump all the index files, then the index files are intact and you can ***ignore*** error messages like the following:

```
stciqstdutil (Error): Bad Column Counts:
  Table IQISubscriber In 13 Existing 11
```

However, if the utility fails to dump any one index file, then the queue is corrupted. Continue with the **procedure on page 76** to dump the queue message index and queue message content.

To dump the queue message index and queue message content

1 Run **stciqstdutil.exe** against the index file (**.rdb** file) you suspect, but use an additional option, **-dp**. For example, you would enter the following (on all one line):

```
stciqstdutil.exe -ip egate/client/iq/{3E2D28EE-010B-11D4-
  BE68-B62F6823CE53}
  -n qlive_{71A66202-7D35-11D4-9271-E2D1B5816FCD}.rdb
  -dp egate/client/iq/{3E2D28EE-010B-11D4-BE68-B62F6823CE53}
```

This command dumps both the queue message index and queue message content. Be prepared for increased disk space usage.

If you see messages saying that the **qmessages_*.dat** file is corrupted, then the queue is corrupted. Continue with the step below to recover the Events in the IQ.

2 Use **stciqstdutil.exe** to dump messages in **qmessages_*.dat** files, using the following command:

```
stciqstdutil -extract
```

A new file, **qmessages_**<*hh*>**.dat**, is created every hour by the IQ Manager. You can choose which hour's messages to dump by selecting the correct hour for the **qmessages_*.dat** file. If the **qmessages_*.dat** file is corrupted, then **stciqstdutil.exe** will stop at the last good message in the file.

3 Delete or reload messages.

If you reached step 2, the queue index files are already corrupted, and information on which message has been processed by which subscriber is lost. You need to decide which messages to reload after you dump the messages in **qmessages_**<*hh*>**.dat** files.

In the iq directory (described below) generated by **stciqstdutil.exe**, you can delete any messages you do not want to reload. There are four files associated with each message.

4 Do one of the following:

- Delete all messages you if you do not want to reload any messages
- Reload all messages back and then use IQ Administrator to mark or delete messages you do not want to be processed. Continue with the **procedure on page 77** to reload the messages.

To reload messages using stciqutil.exe

1 Stop the IQ Manager.

2 Back up the iq directory.

3 Remove the iq directory.

4 Run the following command, all on one line, making the appropriate substitutions:

```
stciqutil -rh [1] -rs [2] -un [3] -up [4]
  -ld -id iq1 -iq iq1 -sub ALL
```

where:

- **-rh** ***[1]*** specifies the Registry Host (such as **localhost**)
- **-rs** ***[2]*** specifies the schema to back up (such as **MySchema**)
- **-un** ***[3]*** specifies the user name (such as **Administrator**)
- **-up** ***[4]*** specifies the password for this user (such as **STC**)

Running stciqstdutil.exe

The **stciqstdutil.exe** can be used:

- To dump records stored in an iq index file (**.rdb** file). See **"Dumping Records" on page 78**.
- To dump a **qmessages**_<*hh*>**.dat** file. See **"Dumping Records" on page 78**.
- To delete records from an iq index file.

  Proceed with caution when deleting a record from an index file. It is usually safer to delete a record from a subscriber index file if you do not want the record to be picked up by a subscriber. Deleting a publisher record is more dangerous, since you need to ensure corresponding subscriber records are also deleted from the corresponding subscriber index files.

The utility does not rely on the e*Gate IQ Manager, and it does not rely on eGate Registry Service, except when logical names in the dump files are desired.

Command-line options for stciqstdutil

To summon command-line help for **stciqstdutil.exe**, enter the following command:

```
stciqstdutil.exe -h
```

See Table 13.

Table 13 Command-Line Options for **stciqstdutil.exe**

Option	Result
-h	Displays command-line help.
--ver	Shows version.
-n <*param*>	Required except for **-extract**: Index file name.
-ip <*param*>	Required except for **-extract**: Path to the queue index file.
-dp <*param*>	Optional: Path to the queue data file ***year*** subdirectory.
-iq <*param*>	Optional: The iq logical name to which the index file belongs.
-rh <*param*>	Optional: Registry host name.
-rs <*param*>	Optional: Registry schema name.
-un <*param*>	Optional: User name.
-up <*param*>	Optional: User password.
-rp <*param*>	Optional: Registry port.
-cnt	Optional: Count number of records only; do not create dump files
-delete <*param*>	Delete the specified record(s) from index file. Used with **-ip** and **-n** options. Other options ignored.
-extract <*param*>	Extract Event content from the specified **qmessages**_<*hh*>**.dat** file. Can be used with optional **-iq**, **-rh**, **-rs**, **-un**, **-up** options. Other options ignored.
-gln <*param*>	Optional: Get the logical name of publisher, subscriber, IQ, or Event Type by supplying a string with a UUID. Must be used with -rh, -rs, -un, and -up.

Dumping Records

The **stciqstdutil.exe** utility can dump records stored in any of the following types of iq index file (**.rdb** file):

- A *publisher index file* contains records published by a publisher.
- A *subscriber index file* contains records to be fetched by a subscriber. (A subscriber journals a subscriber record after it has fetched it.)
- A *journal index file* contains both journaled publisher and subscriber records.

The output is written to a dump file (**.dump** file). The dump file contains Event Type, publisher, subscriber, enqueued time, state of the Event, and so forth, for each record.

For an index file, if the -dp option is supplied, Event content is also dumped. The Event content is simultaneously written to the **.dump** file and a **.data** file. The **.data** file

contains Event content only. For a **qmessages_**<*hh*>**.dat** file, both the **.dump** and **.data** file are generated.

If the **-iq** option is used, the records are dumped to a directory in a format recognized by **stciqstdutil**. The **stciqstdutil.exe** utility can later be used to reload the Events to the queue. The directory name has to be the real logical name of the iq to which the index file or **qmessages_**<*hh*>**.dat** file belongs; if it is not, **stciqstdutil.exe** is unable to reload the Events.

Another way to reload the Events is to use a file-based e*Way to read the **.data** file.

Usually the Event Type, publisher, subscriber are displayed as UUIDs in the dump file. To display logical names, you need to supply the **-rh**, **-rs**, **-un**, and **-up** options. In this case, the utility also tries to get the iq logical name if the directory path supplied with the **-ip** option or file name supplied with **-extract** option contains the path to the queue directory.

Examples of stciqstdutil.exe usage

- Example 1:

```
stciqstdutil -ip . -n qlive_{71A66202-7D35-11D4-
  9271-E2D1B5816FCD}.rdb -dp .
```

Example 1 dumps records in **qlive_{71A66202-7D35-11D4-9271-E2D1B5816FCD}.rdb** and Event content associated with each record. In this case the index file and the data file ***year*** subdirectory are located in the current directory.

- Example 2:

```
stciqstdutil -ip egate/client/iq/{3E2D28EE-010B-11D4-
  BE68-B62F6823CE53}
  -n qlive_{71A66202-7D35-11D4-9271-E2D1B5816FCD}.rdb
  -dp egate/client/iq/{3E2D28EE-010B-11D4-BE68-B62F6823CE53}
  -iq iq1
```

Example 2 dumps index file in the queue directory **egate/client/iq/{3E2D28EE-010B-11D4-BE68-B62F6823CE53}**. The output is also written to subdirectory **iq1** for reloading by **stciqutil**.

- Example 3:

```
stciqstdutil -ip . -n qarchive_01.rdb -dp ../../..
  -rh localhost -rs schema -un user -up password
```

Example 3 dumps records stored in **qarchive_01.rdb** and associated Event content. Since the **-iq**, **-rh**, **-rs**, **-un**, and **-up** options are used, the dump file will contain logical names.

- Example 4:

```
stciqstdutil -extract qmessages_01.dat
```

Example 4 extracts Event content from **qmessages_01.dat**.

- Example 5:

```
stciqstdutil -extract qmessages_23.dat -iq iq1
```

Example 5 extracts Event content from **qmessages_23.dat** and writes it to the **iq1** subdirectory for reloading by **stciqutil.exe**.

- Example 6:

```
stciqstdutil -ip . -n qlive_{71A66202-7D35-11D4-
  9271-E2D1B5816FCD}.rdb -delete 10:20
```

Example 6 deletes records 10 to 20 from subscriber index file **qlive_{71A66202-7D35-11D4-9271-E2D1B5816FCD}.rdb**.

- Example 7:

```
stciqstdutil -ip . -n qarchive_01.rdb -delete 10
```

Example 7 deletes record 10 from the **qarchive_01.rdb** index file.

10.6 Maintaining the Notification Queue

The Notification Queue is an internal IQ that the Control Broker uses to store all notifications. This IQ has its own cleanup schedule, controlled and managed separately from Event IQs. You can adjust this IQ's cleanup schedule the same way you configure an IQ's cleanup schedule (the procedure is described below).

The default schedule cleans up resolved notifications everyday at 4:00 AM; in most cases you will not need to change this.

To configure the Notification IQ cleanup schedule

1. In Schema Designer, in the Navigator's **Components** tab, select the Participating Host and Control Broker whose Notification Queue you will be configuring.
2. In the Component Editor, select the desired IQ and then edit its properties.

   The **Control Broker Properties** dialog box appears.
3. Select the **Notification Setup** tab. Then, under **Notification Queue cleanup schedule**, click **Set Schedule**.
4. Under **Schedule information**, use the controls to determine the cleanup schedule. (For a description of each control, see step 4 of the **procedure on page 33**.) The default is to run the cleanup daily at 4:00 AM.
5. Click **OK** until you return to the e*Gate Schema Designer's main window.

Important: *If the Notification Queue cleanup runs while you are using the Schema Manager, you must reconnect to the Control Broker or re-open the current schema to view any changes to the displayed notifications.*

10.7 Viewing Archives and Republishing Events

The section below describes how you view archives and republish Events. You can republish Events from one schema to another. When you republish Events, the sequence number remains the same, but the enqueue time and journal expiration time are current. Note that if a republished Event has the same sequence number as an already enqueued Event, you cannot view the republished Event unless you delete the enqueued message with the same number.

When you republish Events, you select the archived Event Type and specify the Event Type you want to republish to. It is possible to republish archived Event Types to different Event Types than in the original schema. When you view an archive created by another schema, all component names are changed to UUIDs.

If you republish a large archive, the IQ Administrator remains busy and unavailable for other actions until the republish is complete. You can start another IQ Administrator to monitor the progress. For details on using the IQ Administrator, refer to **"Using IQ Administrator" on page 63**.

To view archives and republish Events

1. In Schema Manager, click **Launch IQ Administrator**  .
2. Click the **IQ Republish** tab.
3. To include deleted or uncommitted Events, select **Include Deleted/Uncommitted Events**.
4. In the Browse Archives pane, click **Browse**.

   The **Select IQ Archive File** dialog box appears.
5. Click **Browse** and locate the **.archive** file you want to view. This file can be either in the archive folder you specified in the initialization string or in a location where the file was moved.

   There must be disk space available in the **.archive** file location, and you have write permissions to the directory because opening the **.archive** file creates an **.rdb** index file.
6. Click **Open**.

   The Browse Archives pane displays the Event Types and the number of Events per Event Type included in the archive file.

   Opening the archive creates an .rdb index file in the same directory as the .archive file. If the directory has an existing index file for this archive, no new index file is created.
7. To view the properties of an Event, select the Event in the **Message No** box.

   The Event properties pane displays the property information for the event.
8. To view the contents of an Event, select the Event in the **Message No** box and click **View**.

9 To republish Events, select the Event Type you want to republish to in the Select Republish Event Type pane.

10 In the Browse Archives pane, select the Event Type you want to republish, and click **Republish**.

The **Intelligent Queue Administrator - Republish messages** dialog box appears.

11 In the **Message No** and **To** boxes, select the starting and ending message number of the range of Events you want to republish.

12 Click **Republish**.

The republished Events retain their original sequence numbers; the enqueue time changes to the current time, and the journaled time changes accordingly. The priority number is lowered by one, unless it is zero.

Appendix A

Oracle Database Schema

This appendix describes the database schema for the Oracle IQ, including the possible Event states for published and subscribed Events in the IQ.

A.1 Tables

All tables are created in the tablespace EGATETABLE. There is a corresponding publisher and subscriber table pair for each Oracle IQ in the e*Gate schema; therefore, the exact number of tables in the database depends on how many Oracle IQs have been defined.

The database has the following table types:

STC_IQ*XXX*

This is the publisher table that stores the Events. *XXX* is a unique number generated using information in the STC_IQ_NAME_INDEX table.

Columns

The STC_IQ*XXX* table has the following columns:

Column Name	Data Type	Constraint
Publisher	char(38)	not null
MajorMsgSeq	int	not null
MinorMsgSeq	int	not null
MsgType	char(38)	not null
Priority	int	not null
tmEnqueue	date	not null

Column Name	Data Type	Constraint
tmReveal	date	null
tmDelete	date	null
tmJournal	date	null
tmExpire	date	null
PubState	int	not null
Encryption	int	null
Compression	int	null
JournalExpirationJul	int	null
JournalExpirationdwTime	int	null
JournalExpirationTime	date	null
Behavior	int	not null
MsgExpirationTime	date	null
MsgPathLen	int	null
MsgHdrLen	int	null
MsgDataLen	int	null
MsgSeq	number	not null
Msg	long raw	null

STC_IQ*XXX*_SUB

This is the subscriber table that stores all of the subscribers for the subscribed Events in the STC_IQ*XXX* table.

The relationship between STC_IQ*XXX* and STC_IQ*XXX*_SUB is one-to-many.

Columns

The STC_IQ*XXX*_SUB table has the following columns

:

Column Name	Data Type	Constraint
Subscriber	char(38)	not null
SubState	int	not null
SubKey	raw(64)	null
tmGet	date	null
tmInProcess	date	null
tmDone	date	null
tmUnusable	date	null
MsgSeq	number	not null

STC_IQ_NAME_INDEX

This table stores the latest unique number which is used for IQ name generation.

Columns

The STC_IQ_NAME_INDEX table has the following columns:

Column Name	Data Type	Constraint
NameIndex	number	null

STC_IQ_NAME_MAP

This table is used to map a logical queue name defined in the e*Gate Schema Designer to the physical table in the Oracle database.

Columns

The STC_IQ_NAME_MAP table has the following columns:

:

Column Name	Data Type	Constraint
LogicalName	varchar2(255)	primary key
InternalName	varchar2(30)	not null
ControlLock	number	default 0
Version	number	null

STC_IQ_VERSION

This table stores the version number of the Oracle IQ. This information allows the Oracle IQ to handle queues that are generated by an older version.

Columns

The STC_IQ_VERSION table has the following columns:

Column Name	Data Type	Constraint
Version	number	null

A.2 Indexes

All indexes are created in the tablespace EGATEINDEX. They are used to speed up Event retrieval.

There are four indexes created for each Oracle IQ:

STC_IQ*XXX*_PKEY_IDX

This index uses the MsgType, Publisher, Priority, tmEnqueue, MajorMsgSeq, and MinorMsgSeq columns in table STC_IQ*XXX*.

STC_IQ*XXX*_PSEQ_IDX

This index uses the MsgSeq column in table STC_IQ*XXX*.

STC_IQ*XXX*_SGET_IDX

This index uses the Subscriber and Substate columns in table STC_IQ*XXX*_SUB.

STC_IQ*XXX*_SSEQ_IDX

This index uses the MsgSeq and Substate columns in table STC_IQ*XXX*_SUB.

A.3 Stored Functions

There are three stored functions used in the Oracle IQ:

STC_IQ_JRNL_EXP_TIME_FN

This function calculates journal expiration time.

STC_IQ_JUL2CAL_FN

This function converts a julian date to a calender date.

STC_IQ_MAP_NAME_FN

This function maps a logical IQ name to an internal name. This is done to eliminate the restriction of the database naming convention. The internal name will have the following format:

```
stc_iqname_index
```

where *name_index* is a number taken from the STC_IQ_NAME_INDEX table.

A.4 Stored Procedures

Each Oracle IQ in an e*Gate schema uses four stored procedures to support its operations:

STC_IQ*XXX*_JRNL_PRC

STC_IQ*XXX*_JRNL_PRC journals Events in the IQ. It performs the following operations:

1. It marks all expired Events as journaled. (Expired Events are defined as those that are either clean or revealed and have an expiration time which is less than the journal time.) Note that an Event can be expired if ***any*** of the Event's subscribers are in a clean state at the expiration time.
2. It marks all Events that are done as journaled.
3. It marks all Events that are marked expired as journaled.
4. It marks all deleted Events as journaled.
5. It deletes all entries in the subscriber table that are journaled and have journal expiration times less than the current journal time.
6. It deletes all entries in the publisher table that are journaled and have journal expiration times less than the current journal time.

The multiple update statements are required because there are multiple journal states: DGIQ_PUBSTATE_J, DGIQ_PUBSTATE_EJ, DGIQ_PUBSTATE_REJ, and DGIQ_PUBSTATE_DJ. (For information on these Event states see **"Event States" on page 88**.)

Only the specified publisher's Events are reorganized during the procedure.

STC_IQ*XXX*_MRK_CLN_PRC

STC_IQ*XXX*_MRK_CLN_PRC marks an Event as clean. It is used only by subscribers and performs the following operations:

- It obtains the rowid and msgseq using the key.
- It uses the msgseq value to mark the subscriber clean.
- It uses the rowid to mark the publisher in DGIQ_SUB_POOL_READY mode.

STC_IQ*XXX*_MRK_JRNL_PRC

STC_IQXXX_MRK_JRNL_PRC marks Events as "journal" in the IQ. It performs the following operations:

1. It marks all expired Events as journaled. (Expired Events are defined as those that are either clean or revealed and have an expiration time which is less than the journal time.) Note that an Event can be expired if ***any*** of the Event's subscribers are in a clean state at that time.

2 It marks all Events that are done as journaled.

3 It marks all Events that are marked expired as journaled.

4 It marks all deleted Events as journaled.

The multiple update statements are required because there are multiple journal states: DGIQ_PUBSTATE_J, DGIQ_PUBSTATE_EJ, DGIQ_PUBSTATE_REJ, and DGIQ_PUBSTATE_DJ. (For information on these Event states see **"Event States" on page 88**.)

It is possible that more than one publisher's Events will be marked by this procedure.

STC_IQ*XXX*_POOL_PRC

STC_IQ*XXX*_POOL_PRC marks an Event as fetched when the IQ is in subscriber pooling mode. It performs the following operations:

- It tries to lock the Event.
  - If the lock attempt fails, the procedure returns immediately without performing any operation.
  - If the lock attempt succeeds, it changes the Behavior column value in the STC_IQ*XXX* table to DGIQ_SUB_POOL_COMPLETED and marks the Substate column value in the STC_IQ*XXX*_SUB table as fetched. The procedure also deletes other subscriber entries that point to the same Event.

A.5 Event States

The following sections describe the possible states for Events published and subscribed to in the IQ. The number in parentheses next to each state indicates the value in the Oracle database for that state. Note that some of the states listed are not currently implemented in the Oracle IQ, and are reserved for future use.

A.5.1 Publisher-assigned States

The following Event states are assigned by the publisher:

DGIQ_PUBSTATE_ERROR (-2) — Error state.

DGIQ_PUBSTATE_A (-1) — Event state is not relevant at this time.

DGIQ_PUBSTATE_C (0) — The Event is clean.

DGIQ_PUBSTATE_R (1) — The Event is revealed.

DGIQ_PUBSTATE_CLEANABLE (2) — The Event can be cleaned.

DGIQ_PUBSTATE_D (3) — The Event is deleted.

DGIQ_PUBSTATE_E (4) — The Event is expired.

DGIQ_PUBSTATE_RE (5) — The Event is revealed but expired.

DGIQ_PUBSTATE_READYTOBEDEL (6) — The Event is ready to be deleted.

DGIQ_PUBSTATE_J (7) — The Event is journaled.

DGIQ_PUBSTATE_EJ (8) — The Event is expired and journaled.

DGIQ_PUBSTATE_REJ (9) — The Event is revealed, expired and journaled.

DGIQ_PUBSTATE_DJ (10) — The Event is deleted and journaled.

A.5.2 Subscriber-assigned States

The following Event states are assigned by the subscriber:

DGIQ_SUBSTATE_ERROR (-2) — Error state.

DGIQ_SUBSTATE_A (-1) — Event state is not relevant at this time.

DGIQ_SUBSTATE_C (0) — The Event is clean.

DGIQ_SUBSTATE_NE (1) — The Event cannot be expired.

DGIQ_SUBSTATE_FD (2) — The Event is fetched and done.

DGIQ_SUBSTATE_FID (3) — The Event has been fetched, processed, and is done.

DGIQ_SUBSTATE_FU (4) — The Event is fetched and unusable.

DGIQ_SUBSTATE_FND (5) — The Event is fetched but not done.

DGIQ_SUBSTATE_F (6) — The Event is fetched.

DGIQ_SUBSTATE_FI (7) — The Event is fetched and in process.

Index

Symbols

A

B

C

D

E

F

G

I

R

S

T

U

W