FioranoMQ® 9

FAQ's

Fiorano

Content

FioranoMQ - Concepts and Architecture

FioranoMQ

Question 1: What is FioranoMQ?

FioranoMQ incorporates a 100% pure Java implementation of JMS (the Java Message Service API (version 1.1) released by Sun Microsystems, Inc.), which isolates developers from the details of low-level network programming and provides a standards-based method to access distributed-system services.

FioranoMQ is an event-driven communication platform that dramatically reduces the development time for network applications, providing for location independent application development through its Publish/Subscribe and PTP (Point to Point) communication APIs. It also includes support for critical network services such as transactions and guaranteed message delivery, considerably easing the process of developing and deploying Internet, Intranet and Extranet applications. JMS APIs and services allow developers to add value to an application by focusing more on its core business logic, rather than its low-level infrastructure. FioranoMQ implements all of the JMS Pub/Sub and PTP APIs and includes support for HA, Durable connection, XA support, HT, LMS, Clustering through Dispatcher-Bridge& Repeater, Application server integration, DMQ, Message compression and Encryption, security, remote administration, guaranteed message delivery, scalability and language interoperability.

The security implementation includes integrated SSL support (with digital certificates) for encrypted data transfers together with ACL-based security, built on Java 2 standards, allowing an external administrator to control access to system resources. Comprehensive Administration API allows remote monitoring of servers through JMS; while an integrated and completely native C runtime library, C#runtime library and C++ runtime library (in addition to the standard Java library) allows C, C# runtime library and C++ Java programs to seamlessly communicate with each other. Automatic store-and-forward across multiple servers ensures scalability, performance and guaranteed message delivery across faulty networks. The FioranoMQ platform brings all of the benefits of standards-based JMS API’s to developers without forgoing performance or scalability.

Question 2: What are the system and software requirements for FioranoMQ?

The following table lists the minimum requirements for FioranoMQ:

Components	Minimum Requirement
Platform	FioranoMQ server and clients can run on any platform that supports os, JRE 1.2 or above. Special/Limited editions of FioranoMQ clients can run on JRE 1.1 as well. Supported platforms include, Microsoft Windows, Sun Solaris, Red Hat Linux, IBM AIX and HP-UX etc
Memory	Pentium class PC with a minimum of 256 MB of RAM
Hard Disk	For Windows installer, 250 MB of free disk space is required. For Linux and Unix flavor installer, 210 MB of free disk space is required.

The Windows version of the FioranoMQ installation comes bundled with Sun's 1.4 JRE. The Unix version of the installer does not come bundled with a JRE. Users of FioranoMQ on Solaris are recommended to use JavaSoft's JRE 1.4 (hotspot enabled).

Question 3: What does FioranoMQ provide other than providing complete JMS implementation?

FioranoMQ provides for a 100% Java implementation of the JMS specification. It also provides features like, HA (High Availability) , Durable connection, XA support, Hierarchy Topic, Large Message Support, Clustering through Dispatcher, Bridge, Repeater, Application server integration, DMQ, Messagecompression and Encryption.

For more details, refer to *FioranoMQ Concept Guide*.

FioranoMQ Product Line

Question 4: What are the types of licenses available for FioranoMQ?

FioranoMQ has separate releases for Windows and Unix (and related OS). FioranoMQ Server is available in 3 versions. Please refer to the license agreement for more information.

Demo version

The Demo version of FioranoMQ supports a large number of concurrent clients, unlimited number of topics and queues; and is not node locked. This version of FioranoMQ is valid only for a 45-day limited license. It can be used only for evaluation purposes.

Developer-subscription version

The Developer-subscription version of FioranoMQ is node locked and is valid for 365 days at a time. The only restriction it holds is that it cannot be used for deployment. This version is only meant for development purposes.

Deployment version

The Deployment version of FioranoMQ is the full deployment version of FioranoMQ. This version is typically a perpetual license; node locked to a single IP, and contains no restrictions on the number of clients, topics, or queues.

Demo Version and Developer Demo Versions of FioranoMQ can be downloaded from http://www.fiorano.com.

Other key products encompassing the FioranoMQ product family include:

FioranoMQ Bridges

FioranoMQ provides for:

- **MQSeries:** Bridges FioranoMQ Bridge can be configured between an instance of FioranoMQ and IBM MQSeries. Bridges to IBM MQSeries support all JMS Message types.
- **MSMQ Bridge:** The FioranoMQ Bridge for MSMQ uses Microsoft's COM architecture for communicating with the MSMQ Server. The bridge communication with the MSMQ server is done through java wrapper files, generated over a type library file provided by Microsoft for this purpose.
- **Tibrv Bridges:** The FioranoMQ Bridge for Tibrv uses Tibrv's JAVA APIs for communicating with the Tibrv daemon. The bridge communicates with the Tibrv daemon using the NATIVE (JNI) implementation provided by Tibrv.

Fiorano InfoBus

Fiorano’s Multicast implementation of JMS is a server less implementation. It provides for a distributed messaging system using a server less,zero-administration version of a JMS implementation. FioranoMQ Multicast incorporates a 100% pure Java multicast-backbone which works on all platforms.

Interoperability with other Middleware Platforms

Question 5: How does FioranoMQ integrate with other legacy Middleware messaging Servers such as MQSeries/MSMQ?

FioranoMQ provides Bridges to MSMQ and any other JMS Messaging provider.

Question 6: Does the XML Toolkit work for Stream and Object Messages?

Fiorano's XML Interoperability Toolkit allows for Interoperability across JMS Vendors. The Toolkit provides APIs to convert a JMS Message to an XML document and also to recreate a JMS Message from an XML document. The toolkit currently supports conversion for TextMessage and MapMessage. The toolkit is generic and supports XML Parser plug-ins at runtime. This toolkit can be used to communicate across enterprises that use different JMS Vendors for their messaging needs. The DTD (Document Type Definition) of the Messages is public and can be used by any non-Fiorano JMS vendor to communicate with FioranoMQ.

The XML toolkit defines DTD only for Text and Map Messages. Hence only Text and Map Messages have an XML converter.

FioranoMQ SSL

Question 7: Does FioranoMQ support SSL (Secure Socket Layer) communications?

FioranoMQ allows for secure connections through Secure Socket Layer between the message clients and the message brokers. The secure version of FioranoMQ server includes integrated SSL support for encrypted data transfers (40-bit (export version) and 128-bit (U.S. version) encryption of all data transfers. Authentication is provided through digital certificates using server and client-side callbacks. FioranoMQ SSL version is 100% Java and is available on both Windows and Unix platforms. For more information, refer to **Chapter 6 FioranoMQ Security and Web Support** in the *FioranoMQ 2007 Handbook*.

Question 8: How does FioranoMQ provide SSL support?

FioranoMQ provides support for SSL using Phaos SSLava Toolkit (http:// www.phaos.com) and through SSL implementation from Sun (JSSE) http:// java.sun.com/products/jsse/.

For more information refer to **Chapter 6 FioranoMQ Security and Web Support** in the *FioranoMQ 2007 Handbook.*

Question 9: Does FioranoMQ SSL version support all the features that are offered by FioranoMQ?

Yes, FioranoMQ SSL version provides all the features that the regular FMQ offers.

Question 10: Are there any limitations using the SSL version of FioranoMQ?

The SSL version of the software is not available for download on our website, due to US import laws. In order to evaluate the SSL version of the software, please send a mail to support@fiorano.com.

The SSL version of FioranoMQ does not work on Windows 98.

FioranoMQ Server Resources

Question 11: What is the memory footprint of the MQ Server and Client?

FioranoMQ Server has a 52 MB memory footprint on startup. The minimum Client runtime libraries (which would include fmq-rtl.jar and fmq-common.jar) are nearly 17.4 MB.

Question 12: How much secondary disk space is needed to run the MQ Server?

FioranoMQ installation occupies 210 MB hard disk space for Linux and UNIX, 250 MB hard-disk space for Windows. As persistent messages are logged on to secondary storage, the amount of free hard disk memory for MQ Server would depend largely on the JMS Applications that access the persistent storage engine. If the volume of persistent storage is high,a larger amount of secondary hard-disk space is required.

FioranoMQ Resource Constraints

Question 13: How many concurrent client connections/Sessions/Consumer/Topics can the FioranoMQ Server support?

There are no specified limits on the number of connections that can be created with the FioranoMQ Server.The number of connections is limited only by the hardware, the JVM used and the client Application. The FioranoMQ Server implements internal memory management algorithms to keep track of the approximate amount of memory being consumed by the system. As the memory limit of the JVM is reached, the FioranoMQ Server automatically starts blocking and (where allowed by JMS semantics) overflowing its internal buffers. The memory manager thus automatically governs the speed of running applications. Hence, the number of concurrent connections depends on the Client Applications. It is strongly recommend to simulate JMS clients and verify the concurrent connection limit of the FioranoMQ Server on your deployment environment. For more information, read the FioranoMQ Scalability, Reliability and Performance section.

FioranoMQ solves Java's Connection Management limitations using Fiorano Scalable Connection Manager

FioranoMQ does not have specified limits on the resources that can be allocated on the MQServer by the Client Applications. As explained earlier,FioranoMQ implements a highly sophisticated algorithm to protect the MQServer from running out of memory. The precise number of destinations/sessions/consumers that can be created on the MQServer depends on the JVM, hardware used and the client applications.

Question 14: What is the maximum message size that can be sent using FioranoMQ?

FioranoMQ does not have a specified limit on the resources that can be allocated on the MQServer by the Client Applications. For optimal performance, it is recommended to limit the message size to 100K. For messages of sizes greater than 100K, the developer can employ a straightforward algorithm to split messages as chunks while sending and regroup them on receipt. Again, the optimal message size largely depends on the JVM and platform being used.

Transactions

Question 15: Does the Client transaction have a transaction boundary?

FioranoMQ provides transaction support at the Client end and is limited by the JVMs memory allocation. To prevent "OutOfMemory" situations in the client, FioranoMQ limits "maximum transaction boundary" to 2048 KB or 2 MB.

Question 16: Does FioranoMQ support distributed transactions?

FioranoMQ has added support for distributed transactions after the 6.0 release and a beta version of the FioranoMQ server with this support is available on request.

Contact support@fiorano.com for an evaluation version of the above mentioned beta version.

Publishing Messages to FioranoMQ

Question 17: When does a call to publish on a Queue or a Topic return?

The publish call made by the Publisher in the context of a non-transacted session does not return till the message is delivered to the MQServer. If the published message is persistent, FioranoMQ assures that the call to publish does not return till the messages have been logged on to the persistent store.

Question 18: What is the functionality of publish call in a transacted Session?

In the case of a transacted session, the publish call logs the message on client-side to be sent to the MQServer only on commit call. The call to commit() does not return until the messages have been logged on to the persistent store.

Question 19: What is the difference between publishing persistent/non-persistent messages to a Queue?

Persistent messages sent to a queue are stored in the FioranoMQ persistent store on the server side. These messages can be retrieved from this store by connecting to the FioranoMQ server, even after it has been re-started.

Non-persistent messages are held in-memory on the FioranoMQ server side and are delivered once and only once to the receivers. In case of the FioranoMQ server going down, these messages are no-longer available as they are not logged anywhere.

For more information on Non-persistent messages on queues, refer to **FioranoMQ Non-Persistent Messages**.

FioranoMQ Persistent Store

Question 20: What database does FioranoMQ use for storing Messages?

FioranoMQ uses an optimized file based data store to store messages. By using an optimized file based store, FioranoMQ performs at least 10 times faster than other competing JMS vendors when persistent messages are retrieved. FioranoMQ also ensures efficient storage (a single copy of the message is stored in the persistent store, even when there are more than 1 receivers of the message), management, retrieval and deletion of messages from this persistent store.

In addition, FioranoMQ has introduced support for RDBMS to be used as the data-store for storing persistent messages and durable subscriptions. A beta version of this support is available for evaluation on request. Contact support@fiorano.com for the same.

This version currently supports Oracle, DB2, MSSQL, MySQL, and Cloudscape databases.

Client Connection

Question 21: How does FioranoMQ handle client disconnections?

In FioranoMQ Server, client connections closed either using the close() API or abruptly using CTRL+C. If the Socket connection between the Client and the MQ Server times out,the MQServer cannot detect these half-open connections until the underlying TCP layer times out the client socket connection. To ensure that FioranoMQ does not have "half-open" connections, please ensure to turn on the PING options.

Question 22: Is there a reconnection (with the server) handling mechanism?

The installed Exception Listener gets invoked when the connection to the messaging server is lost. Within your onException() you can write your own code to reestablish connection with MQServer.

FMQ also provides for a revalidate() call that revalidates connections. Revalidate re-establishes connection with MQServer and restores the validity of created sessions, senders and receivers on this connection. FMQ provides revalidation for both topic and queue connections.

After the 6.0 release, FioranoMQ has introduced DurableConnections where reconnection to the FioranoMQ server has become transparent to the clients. Now clients can automatically reconnect to the primary FioranoMQ server or backup FioranoMQ server (if primary is un-available) without any extra logic in the client application.

Installing and Starting FioranoMQ Server

Installation

Question 1: How do I install FioranoMQ Server on Windows platform?

In this procedure, you have to run the self-installing executable to unpack and install FioranoMQ. FioranoMQ Installation wizard installs FioranoMQ and adds FioranoMQ in the startup directory. FioranoMQ installer comes bundled with JavaSoft 1.5 version of the JRE. FioranoMQ setup program installs the FioranoMQ daemon into the start-up group on all Windows machines. Subsequent versions of the software install the daemon as a service under Windows NT.

Troubleshooting the Windows Installation

Following is a list of potential problems that can occur during installation and solutions:

corrupt cabinet file

The FioranoMQ executable that you have downloaded is corrupted (cabinet file contains compressed application, data, resource and DLL files). Check the file size against the expected file size listed in the download instructions. If they do not match, try downloading again.

System Error during Decompression

You might not have sufficient space on the disk that contains your TEMP directory. This program cannot be run in DOS mode. Perform the following steps:

1. Open the MS-DOS shell (**Windows** > **Start** > **Programs** >**MS-DOS Prompt**)
2. Right-click the title bar, select **Properties**
3. Choose the Program tab, click **Advanced**
4. **Note:** Make sure the item "Prevent MS-DOS-based programs from detecting Windows" is unchecked.
5. Click **OK**, click **OK** again.
6. Exit the MS-DOS shell. Restart your computer.

Error Message

If you see the following error message on Windows:

config.nt. the system file is not suitable for running MS-DOS and Microsoft Windows Applications.

It indicates a problem with the %SystemRoot%\System32\COMMAND.COM file that has been seen on some installations of Windows 2000. If you encounter this error message when you try to launch the installer, refer to the Microsoft web site for information about resolving the problem.

Any other installation problems

For any other installation issues, send an e-mail to the FioranoMQ Support team

Question 2: How do I install FioranoMQ Server on Solaris and other UNIX platforms?

In this procedure, you have to unpack the archive containing complete installation of FioranoMQ.To install FioranoMQ using this "tar.gz", perform the following steps:

1. Extract all the files in "tar.gz" (untar and unzip) to a directory (say /export/ home)
2. To run the FioranoMQ scripts without changes, the PATH and CLASSPATH system variables needs to be modified
3. To ensure that the changes are valid through out the system, please make appropriate changes to the System PATH and CLASSPATH variables
4. Ensure that $FMQ_DIR points to Fiorano's installation directory. If you have installed FioranoMQ in /export/home/Fiorano, FMQ_DIR must point to / export/home/Fiorano/fmq.

Troubleshooting the Solaris Installation

Following is the list of potential problems that can occur during installation from the archive and their solutions.

corrupt archive file

FioranoMQ archive that you have downloaded is corrupted. Check the file size against the expected file size listed in the download instructions. If they do not match, try downloading FioranoMQ again.

Any other installation problems

For any other installation problems, please do not hesitate to e-mail FioranoMQ Support team.

Starting up FioranoMQ Server

Question 3: How do I launch FioranoMQ server on Solaris and Windows?

On Windows and related platforms

There are multiple ways of launching FioranoMQServer on a Windows based platform.

Starting FioranoMQ Server from the Windows Start Menu

FioranoMQ can be launched from the Start group of the Windows machine. FioranoMQ can be started by clicking **Start** > **Programs > Fiorano** > **FioranoMQ** > **FioranoMQ Server**.

Starting FioranoMQ Server from the Windows console window

The FioranoMQ Server can additionally be launched using the script runCon-tainer.bat located in the fmq\bin directory of FioranoMQ installation. FioranoMQ installation for Windows comes bundled with JavaSoft 1.5 JRE and the environment variable %JAVA_HOME% is set for this JRE. Another variable named %FIORANO_HOME% points to the Fiorano directory of FMQ installation. You can edit the batch files in the fmq\bin directory of the installation to point to the appropriate Java run-time environment. Both %FMQ_DIR% and %JAVA_HOME% are set in the batch file, fmq-vars.bat. This batch file is generated during installation to point to the appropriate place and is called automatically from all the startup scripts, including fmq.bat.

FioranoMQ works with any version of JRE that is compliant with JavaSoft JRE 1.2 and higher. To point to a JRE other than the one bundled with FioranoMQ, edit the fmq-vars.bat file in the fmq\bin directory and set %JAVA_HOME% to point to the root directory of the JRE installation (not the bin directory). For example: JAVA_HOME=c:\jre1.2.2.

Please read the comments in the fmq.bat script and ensure that all the environment variables are set correctly.

On Unix and related platforms

The Unix version of the installer does not come bundled with a JRE. It is assumed that the machine, on which FioranoMQ is installed, has a version 1.3 and higher of JRE installed. It is recommended using JRE 1.3 (Hotspot enabled).

"fiorano_vars.sh"

Prior to executing any FioranoMQ script, you need to set environment options FMQ_DIR and JAVA_HOME in the "fiorano_vars.sh" script. FMQ_DIR should point to the installation directory of FioranoMQ and JAVA_HOME should point to a version of jre/jdk 1.2 and above. This script ("fiorano_vars.sh") gets invoked from all FioranoMQ batch scripts.

fmq.sh

The FioranoMQ Server can be started using the fmq.sh script file located in the fmq/bin directory of the FioranoMQ installation directory. The fmq.sh script uses the environment variables ($FMQ_DIR, $JAVA_HOME) that have been set in fiorano_vars.sh to launch an instance of FioranoMQServer. On Unix and related platforms, it is important that the numbers of file descriptors are set to a high value, typically the highest value allowed. In addition, the fiorano_vars.sh script checks whether the MQServer is started by root user and ensures the number of file descriptors available for this process is set to value greater than 10,000. A directed warning message is displayed if this is not fulfilled. It is recommend either to start the MQServer logged in as super user or make sure the Java process that spawns MQServer can open 10,000 file-descriptors.

The scripts run on the Bourne shell. You must port them to other shells, if required.

Question 4: How do I start the FioranoMQ Server using script files?

FioranoMQ comes bundled with the script files to start the MQ Server, as follows:

On Windows and related platforms

FioranoMQ Server can be started by clicking **Start** > **Programs > Fiorano** > **FioranoMQ** > **FioranoMQ Server**.

Alternatively, FioranoMQ Server can be started using the script, fmq.bat in the fmq\bin directory of Fiorano's installation. FioranoMQ installation for Windows comes bundled with JavaSoft 1.4 jre and the environment variables are set for this jre. You can edit the fmq-vars.bat batch files in fmq\bin directory of the installation to point to the appropriate java run-time environment. You can do this by modifying JAVA_HOME to point to any other JDK/JRE installation.

```
SET JAVA_HOME=d:\FIORAN~
```

FioranoMQ works well with any version of jre that is compliant with JavaSoft jre 1.2.2 and higher.

On Unix and related platforms

The Unix version of the installer does not come bundled with a jre. It is assumed that the machine on which FioranoMQ is installed, has a version of java (1.2.2 and higher) installed. FioranoMQ Server can be started using the fmq.sh script file located in the fmq/bin directory of Fiorano's installation. Details of the libraries to include and other environment settings can be found in the batch files. The environment variable $FMQ_DIR should point to the installation directory of FioranoMQ and environment variable $JAVA_HOME should point to java installation directory. On Unix environments, before running this script, it is important to execute the following command on the command-line:

unlimit descriptors

This command is essential since it sets the total number of open file descriptors per process to the highest possible limit. The default value is set to 64, which only allows a maximum of 64 file handles to be opened. This creates a problem at times, since the Fiorano MQ server maintains open file handles for offline-cache entries (for performance reasons).

Question 5: How do I modify the port number on which a server listens for requests?

The port number, corresponding to the port on which a server listens for requests, can be modified by modifying the Port value in the Connection Manager module in Fiorano Studio.

For more information refer to **Chapter 3 Connection Management** in the *FioranoMQ 2007 Handbook.*

Question 6: How do I modify the transport (HTTP, TCP) that is used by the server?

Transport that is used by a server can be changed using the Fiorano Studio. The underlying protocol in the case of TCP, PHAOS_SSL and SUN_SSL is TCP, where PHAOS_SSL and SUN_SSL provide SSL support with Phaos and JSSE being the security toolkits, respectively. For the case of HTTP, communication is over the HTTP protocol and in case of HTTPS_PHAOS and HTTPS_SUN, communication is over HTTPS. For more information refer to **Chapter 3 Connection Management** in the *FioranoMQ 2007 Handbook.*

Question 7: For which versions of JDK does the FioranoMQ's support for SOCKS work?

The support for SOCKS works with JDK versions 1.4 and above. This support might not work for versions below 1.4 due to some bug in Sun's Socket implementation.

Question 8: What are the configurations files read by the server?

FioranoMQ provides a rich set of functionalities that allow enterprise administrators to further fine-tune FioranoMQ, based on application needs. FioranoMQ provides for both static and run-time configuration and debugging utilities.

FioranoMQ provides for extensive set of options that can be set to the server, as a part of server configuration file (configs.xml). All configuration changes made to configs.xml are dynamic.

configs.xml

FioranoMQ's server configuration file, configs.xml is located in the fmq\profiles\FioranoMQ\conf" folder of installation directory, %FMQ_DIR%. FioranoMQ server reads the values of various parameters from this xml file. Once the server is configured, information related to the following components can be read from this file.

Question 9: Are there any changes in the VM parameters to start the MQ Server?

There are no configuration changes required to the JVM. A single Fiorano JMS Server can support 4000 persistent concurrent connections (observed on a 2 processor, 450 MHz each) with Solaris 8 and JDK1.2.2 Solaris VM (build Solaris_JDK_1.2.2_05a, native threads, sunwjit). The number of supported concurrent connection depends on the machine (hardware) on which the server is hosted, the JRE being used on that machine, and the client application.

To increase the number of concurrent connections allowed by the JRE, you should decrease the stack reserve per thread to 256KB (or less). Sun and Microsoft JVMs ship with a default stack reserve of 1 MB/thread. To decrease the stack reserve, start the JVM using the command

```
java -Xss256k ..... <rest of command line>
```

By using a smaller stack reserve and running on a reasonably fast machine, you can easily support over 1000 concurrent client connections to the Fiorano JMS Server.

If you wish to scale this to thousands of concurrent client connections, use FioranoMQ's Scalable ConnectionManager. To configure FioranoMQ to use SCM, refer to **Chapter 3 Connection Management** in the *FioranoMQ 2007 Handbook*.

Question 10: How do I know that FioranoMQ is running?

FioranoMQ provides monitoring capability for its main port using the telnet facility. With this, you can check if FioranoMQ server is running or not. FioranoMQ can be configured in monitorable mode, wherein external applications such as Telnet can connect to the server and retrieve versioning information. In order to do this, the first key pressed after the telnet window has connected to the FioranoMQ server should have it's ASCII value equal to the MonitoringRequest set in the FioranoMQ server.

Run the following command:

```
telnet <server IP> <server port>
```

If telnet client is able to connect, it indicates server is running.

In addition, as a check, try running a sample Publisher and Subscriber(s) Application and check if data being sent by the publisher is being received by the subscriber(s).

Question 11: How many ports does FioranoMQ use?

FioranoMQ, by default, creates an instance of Server Socket on port 1856 to receive client connections and another on 1858 for RMI connector. For details on starting MQServer on different ports, refer to **Chapter 3 Connection Management** in the *FioranoMQ 2007 Handbook*.

Question 12: How do I clear the database of a server?

To clear the FioranoMQ database, run the script ClearDB.bat/ClearDB.sh, located in %FMQ_DIR%\fmq\bin folder. This script accepts profile name & DB Path in the following format:

```
ClearDB.bat/ClearDB.sh [profile name] [DB Path]
```

(%FMQ_DIR%\fmq\bin). You, need to modify the batch file in case, the database is created at a separate location.

This script clears all persistent information relating to security, subscribers and messages from the FioranoMQ server. It should be used only when the FioranoMQ server is not running.

Question 13: How do I modify the location of the database that gets created by FioranoMQ?

By default, the database is created in FioranoMQ installation directory. For example, \Program Files\Fiorano\FioranoMQ200xxx\fmq\profiles\FioranoMQ\run.

You can change the path of database directory, by assigning complete path of desired location, while running the server with the parameter "-dbPath <dir>" thorugh script fmq.bat/fmq.sh.

For example,

```
fmq.bat/fmq.sh -dbPath C:\db
```

To change the database directory path, do the following steps:

1. Invoke the Fiorano Admin Studio.
2. Select **Tools > Configure FioranoMQ** from the menu bar, select the deployments.lst file in the resulting **Browse Deployment.lst** dialog box and click the **Open** button. This shifts the FMQ environment to the offline mode.
3. Now, navigate **to FioranoMQ > Core > FileDBs > PUBSUB** in the Server Explorer.

   The properties of the PUBSUB are displayed in the **Properties** pane.
4. Change the path of database directory to the desired location.
5. Right-click the **FioranoMQ domain** in the Server Explorer and select the **Save** option from the resulting shortcut menu.
6. Save changes to the Configs.xml file and click the **Save** button.

The FioranoMQ server needs to be re-started, for this change, to take effect.

Question 14: Do I need to change the VM options to run my client Applications?

Normally there would not be a need to change JVM options for a JMS client application with FioranoMQ. For applications where many transacted sessions are being used per application and the maximum size of the transaction buffer is large (default is 1 MB), the JVM heap size might have to be changed.

Changing the JVM parameters would also depend on any non-JMS work being performed by a client application.

Applications creating large number of connections, sessions and other JMS objects in a single JVM might also need an increase in the JVM heap size, though loading a single client application to that extent is not recommended.

Question 15: Does FioranoMQ support jdk1.4?

FioranoMQ server and clients can be run on jdk1.4 without any problems.

Question 16: Can the FioranoMQ server be started as a child process that is a part of another process?

On Windows platforms there is no problem in running the FioranoMQ server as a child process within another process but on Unix based systems it is recommended, that the FioranoMQ server be run as a stand alone process.

This is because the FioranoMQ server requires a high value of FDs (file descriptors), for its proper functioning and in Unix based systems; there is a restriction on the number of FDs that can be used by a child process.

Stopping FioranoMQ Server

Question 17: What happens if the FioranoMQ Server is abruptly shut down?

FioranoMQ Server has been designed to take care of abrupt shutdowns like Ctrl+C.

In the eventuality of shutting down the server abruptly, on restart, MQServer automatically recovers and starts off in the most recent stable mode.

It is recommended using shutdown script to stop the MQServer. "Shutdown"script can be located in the fmq\bin directory of Fiorano's installation on Windows based machines and in fmq\bin directory of Fiorano's installation on Unix based machines.

Licensing

Question 18: How do I upgrade from the developer version of FioranoMQ to Deployment version?

After the purchase of FioranoMQ, you need to receive your FioranoMQ deployment license keys through e-mail. Download the full version of the software from www.fiorano.com. The instructions for installing the developer as well deployment version of the software is sent to you as a part of the e-mail.

TroubleShooting

Question 19: FioranoMQ Server does not start?

Please perform the following checks:

- **Check classpath** -Check CLASSPATH and the environment settings in the batch file that is used to start the MQ Server. In addition, ensure that the server is using the same version of run-time libraries.
- **Check Server port-** Check whether another instance of the MQServer has been started on the same port, or if there is any other application that has been started on MQServer port.
- **Check Log Files-** Check the details in the Log files. Log files (mqerr.log. mqout.log) can be located in the bin directory of Fiorano's installation.

If problem persists, please contact: support@fiorano.com.

Question 20: I have just installed FioranoMQ across PC anywhere and the Admin Studio and Admin Tool does not start. When I run the Admin Tool a DOS prompt opens with the following messages, with no further response:

```
Using FMQ_DIR "C:\PROGRA~1\Fiorano\FIORAN~1.5"

Using JAVA_HOME C:\PROGRA~1\Fiorano\FIORAN~1.5\jre1.4

Dec 9, 2004 3:58:29 PM java.util.jar.Attributes read

WARNING: Duplicate name in Manifest: Depends-On Dec 9, 2004 3:58:29 PM
java.util.jar.Attributes read WARNING: Duplicate name in Manifest: Depends-On Dec 9,
2004 3:58:29 PM java.util.jar.Attributes read WARNING: Duplicate name in Manifest:
Depends-On Dec 9, 2004 3:58:29 PM java.util.jar.Attributes read WARNING: Duplicate
name in Manifest: Depends-On Dec 9, 2004 3:58:29 PM java.util.jar.Attributes read
WARNING: Duplicate name in Manifest: Depends-On Dec 9, 2004 3:58:29 PM
java.util.jar.Attributes read WARNING: Duplicate name in Manifest: Depends-On
```

Some java applications, using a swing user interface, do not work when connected using PC anywhere. To run FMQ over PC Anywhere, you need to add

```
-J-Dsun.java2d.noddraw=true
```

To the classpath in the batch file, Studio.bat, that starts Admin Studio. For more details on running Java applications on PC anywhere, please refer to the following link:

http://www.thauvin.net/wiki/display/Tips/Java+Screen+Refresh+and+pcAnywhere

FioranoMQ - Administration Issues

Admin Objects

Question 1: How do I add/remove user defined Topics, Queues and Connection Factories to an existing JMS Server?

FioranoMQ provides a comprehensive set of Administrative APIs, which allow an authorized administrator (a user with admin privileges) to create/destroy/enumerate resources, such as topics, users and connection factories. For more information, refer to **Administering FioranoMQ Server Using APIs** in the *FioranoMQ 2007 Handbook*.

Question 2: Is it possible for all users of the system to create Topics and Queues?

FioranoMQ allows only the Administrator to create Topics and Queues. By setting the following option to true, the Enterprise Administrator grants rights to all users of the system to create Topics and Queues.

ALLOW_ONTHEFLY_CREATION_OF_DESTINATIONS=true

Question 3: Are there any restrictions to the usage of special characters while naming a Destination, Connection Factory, SubscribtionID, ClientID? If so, provide a list of such characters.

Yes, FioranoMQ restricts the usage of most of the special characters in "named objects" (destination, connection factory) and in client IDs as well as subscriber Ids. The only characters that are supported are:

- [A-Z]
- [a-z]
- [0-9]
- "\\"
- "/"
- " "
- "?"
- """
- "<"
- ">"
- "|"
- "["
- "]"
- "+"
- ";"
- ":"

Users

Question 4: How do I control the users logging into my system?

When the API, Topic Connection Factory tcf = tcf.create topic connection (String userName, String passwd) is used by the JMS Client Application, the JMS Server authenticates the user login using the password assigned by the administrator. If the passwords do not match, the connection request is rejected.

In addition, the FioranoMQ Server allows "anonymous" connections. This implies that it is possible to connect to the server using the tcf.createTopicConnection(); without supplying any user name or password.

For more information refer to **Chapter 6 FioranoMQ Security** in the *FioranoMQ 2007 Handbook*.

Other APIs

Question 5: What other Administrative facilities does the FioranoMQ Server provide?

FioranoMQ provides a comprehensive Administrative API, which allows an authorized Administrator (a user with admin privileges) to query the current state of the system. The following types of queries are supported:

- Determining the set of users, those are currently connected to the server.
- Determining all client IDs, on which durable subscriptions have been created on the server.
- For each client ID, determining the subscriber IDs, for each created subscription.

In addition to that, MQ Server can be managed through JMX. All sub-systems of FMQ are exposed through JMX and can be managed online.

For more information refer to **Chapter 1 Configuring FioranoMQ through JMX Tools** in the *FioranoMQ 2007 Handbook*.

Question 6: Can messages be purged from Queues?

FioranoMQ provides the following Administration API that allows the Administrator of FioranoMQ to purge all messages from the Queues:

Administrator: purgeQueueMessages (String queueName)

For more information refer to **Chapter 22 Administering FioranoMQ Server using APIs** in the *FioranoMQ 2007 Handbook*.

Question 7: How do I determine the Queue size?

You can determine the size of a queue using the Admin API, MQAdminService::get Number Of DeliverableMessages (String queueName).

Question 8: Is there any API available to determine the number of messages pending for any given durable subscriber on a topic?

The API to determine the number of deliverable messages for the given durable subscriber is:

Fiorano.jms.runtime.admin.MQAdminService.get Number Of Deliverable Messages(clientID, subscriberID);

Question 9: How do I perform tracing and logging in FioranoMQ?

FioranoMQ allows administrators to dynamically set tracing levels for various server components. These trace levels, define the “verbosity” of logs generated by the MQServer. Logs can be directed to a console or to a file. MQAdministrator can control the number of log files generated and the number of lines of log in each log file. This can be done by using the Fiorano Studio, a GUI to administer FioranoMQ, to set log levels accordingly for different components. For more information, refer to **Chapter 22 Administering FioranoMQ Server Using APIs** in the *FioranoMQ 2007 Handbook*.

Admin Studio

Question 10: How do I launch the Fiorano Admin Studio?

Fiorano is bundled with a GUI utility to administer the FioranoMQ server. The GUI provides a number of facilities in terms of configuring managing Users, groups, topics, queues, connection factories, durable subscriber, dispatcher, repeater, snooper, bridges, logging and tracing. As in the case of FioranoMQ server, the AdminGUI can be launched in two possible ways on Windows platforms:

On Windows and related platforms

Starting Fiorano Admin Studio from the Windows Start Menu

FioranoMQ Admin Studio can be launched from the Start group of the Windows machine. FioranoMQ can be started by clicking **Start>Programs>Fiorano>FioranoMQ >Fiorano Studio.**

Prior to running the above script, start FMQ server by running the script fmq.bat located in fmq\bin directory of FioranoMQ installation (%FMQ_DIR%\fmq\bin).

FioranoMQ installation for Windows comes bundled with JavaSoft 1.4 JRE and the environment variable %JAVA_HOME% is set for this JRE. You can edit the batch files in the scripts directory of the installation to point to the appropriate Java runtime environment. Both %FMQ_DIR% and %JAVA_HOME% are set in the batch file, fmqvars.bat. This batch file is generated during installation to point to the appropriate place and is called automatically from all the startup scripts, including Studio.bat.

Please read the comments in the Studio.bat script and ensure that all the environment variables are set correctly.

On Unix and related platforms

The Unix version of the installer does not come bundled with a JRE. It is assumed that the machine on which FioranoMQ is installed has a version 1.3 and higher of JRE installed. It is recommended to use JRE 1.4 (Hotspot enabled).

fiorano_vars.sh

Prior to executing any FioranoMQ script, you need to set, environment options FMQ_DIR and JAVA_HOME in the fiorano_vars.sh script. FMQ_DIR should point to the installation directory of FioranoMQ and JAVA_HOME should point to a version of jre/jdk 1.4 and above. This script (fiorano_vars.sh) gets invoked from all FioranoMQ batch scripts.

Studio.sh

Fiorano Admin Studio can be started using the Studio.sh script file, located in the $FMQ_DIR/Studio/bin directory of the FioranoMQ installation directory. The Studio.sh script uses the environment variables ($FMQ_DIR, $JAVA_HOME) that have been set in fiorano_vars.sh to launch an instance of the Fiorano Admin Studio. The scripts run on the Bourne shell. If required, you must port them to other shells.

For more information refer to **Chapter 22 Administering FioranoMQ Server Using API's** in the *FioranoMQ 2007 Handbook*.

Question 11: How do I launch the Admin Tool with server running on a protocol other than TCP?

The FioranoMQ server can run on six different protocols, namely TCP, HT-TP,PHAOS_SSL,SUN_SSL, HTTPS_PHAOS and HTTPS_SUN. By default, the FioranoMQ AdminTool connects to the FioranoMQ server over TCP. If the FioranoMQ server is running on a protocol different than TCP, then you can change the protocol on which Fiorano Admin Studio needs to be run by changing the Transport Protocol property of the FMQ server connection.

For more information refer to **Chapter 3 Connection Manager** in the *FioranoMQ 2007 Handbook.*

JNDI Store

Question 1: Does FioranoMQ integrate with JNDI/LDAP servers like Netscape Directory Server?

FioranoMQ 7.0 onwards provides for integration with JNDI compliant directory servers like Netscape Directory Server and Open LDAP. For more information refer to **Chapter 24 Application Server Integration** in the *FioranoMQ 2007 Handbook*.

Inter- Network Connectivity

Question 2: I want to run FioranoMQ on a machine with two network interfaces. One of the interface (say with IP Address 192.168.*.*) is visible to internal clients and can be used by them to connect to the server, while the other interface (with IP Address 10.48.*.*) is visible to external clients and can be used by the them to connect to the server. Is it possible to achieve this in FMQ? If yes, what are the expected configuration changes that should be made?

It is possible to run FMQ server on machine with multiple network interfaces where each (or some) interface is visible to different network.

Here are the two possible FMQ configurations, either of which can be used to run FMQ in above mentioned scenario.

Solution 1: Create Separate Connection Factories

You should create separate Connection factories for internal and external clients. The external clients should use the connection factory which has the connect url `http:// 10.48.*.*` and internal clients use the connection factory with the connect url `http://192.168.*.*` .

You can create the connection factories by adding the following tags in the ad-min.xml.

```
<TopicConnectionFactory> connFactoryForExternalClients</Name> .. http://
10.48.*.*:1856</ConnectURL> .. false</UpdateConnectURL> </TopicConnectionFactory>
```

```
<TopicConnectionFactory> connectionFactoryForInternalClients</Name> .. http://
192.168.*.*:1856</ConnectURL> .. false</UpdateConnectURL> </TopicConnectionFactory>
```

Solution 2: Create a Connection Factory with a backup URL

You can create a connection factory which has a connect url and a backup url. The connect url may be equal to the IP address valid for external clients (`http://10.48.*.*`) and the backup url may be equal to the IP address valid for internal clients (`http:// 192.168.*.*`) or vice versa.

You can specify the connect, as well as backup url in the connection factory as described below:

```
<TopicConnectionFactory> myTCF</Name> .. http:// 10.48.*.*:1856</ConnectURL> http://
192.168.*.*:1856</BackupConnectURLs> ..
```

Question 3: I am using two RMI JMX adapters to monitor two separate components of FioranoMQ(server/repeater/bridge etc.).How ever I am getting exception at the startup, that RMI connector cannot start up, because port 1858 is busy/already in use. What should I do now?

It is possible to run multiple instances of FioranoMQ server, their different components that is, standalone repeater, Standalone Bridge and so on. It is also possible to monitor them all, by using multiple RMI JMX connector. How ever, FioranoMQ RMI connector by default runs on port 1858, so, to start multiple instances of RMI adaptors, user must, assign them a unique and free port. To assign a port, to an RMI connector, following steps need to be followed.

1. Launch Fiorano Studio using %Fiorano_Home%\Studio\bin\Studio.bat or by selecting **Start > Programs > Fiorano > FioranoMQ8.x > Fiorano Studio.**
2. Select **Tools > Configure FioranoMQ** from the menu bar, select the %Fiorano_Home%\fmq\profiles\%selectedProfile% directory, from the resulting **Select Profile Directory** dialog box and click the **Open** button.
3. Now, navigate to %**selectedProfile**% > **Connector >RmiConnector** in the Server Explorer.
4. Change the property Naming Port from default 1858 to the desired unique and free port number.
5. After making the above change, right-click the FioranoMQ domain in the Server Explorer and select the **Save** option from the resulting shortcut menu. Your change is saved in the configs.xml file.
6. Restarting the application with changed configuration runs the RMI connector on the specified port.

Question 4: I am using FioranoMQ message snooper to snoop messages on topics/ queues. How can I see, all snooped messages of registered destinations, through Fiorano Studio.

Please follow these steps to see messages in Fiorano Message Snooper.

1. Launch Fiorano Studio using %Fiorano_Home%\Studio\bin\Studio.bat or by selecting **Start > Programs > Fiorano > FioranoMQ8.x > Fiorano Studio.**
2. Connect to FioranoMQ using Fiorano Studio. Navigate to **FMQ Server Connection > Snooper**. Right click on snooper and choose add destinations to snooper in order to register a new destination for message snooping.
3. After having added all destinations in Message Snooper, select queue node appearing under snooper node. Right click on the queue node and choose browse messages. This shows your messages from all queues that are registered on the snooper. Similarly to view messages of topics registered on snooper, right-click topic node appearing under snooper node and choose browse message again.

FioranoMQ Security and Web Support

SSL Encryption

Question 1: What is SS- based 40-128 bit security? How does it use Digital Certificates? Do I need SSL?

FioranoMQ supports 40 to128-bit encryption of data. FioranoMQ is configured to provide more than 40-bit encryption in the United States only.

Question 2: Does Fiorano provide support for content-based encryption?

FioranoMQ does not provide toolkits for encryption, neither does it interpret the contents of the message. As a result, the developer can use any encryption technology to encrypt the message.

Ciphersuites

Question 3: What is a ciphersuite?

A ciphersuite is a collection of cryptographic algorithms and is used in conjunction with a SSL connection. Algorithms in a ciphersuite consist of:

- **A key exchange algorithm**: Diffie-Helmman or RSA
- **A certificate format**: Digital Signature Standard (DSS) or RSA
- **A symmetric encryption algorithm**: DES, RC4, RC2, IDEA, and NULL
- **The mode that the encryption algorithm is used in**: Cipher Block Chaining (CBC), Encrypt-Decrypt-Encrypt (EDE)
- **A message digest or hash algorithm**: MD5 or SHA

Question 4: How are ciphersuites selected for use in an SSL connection?

The FioranoMQ server has a set of acceptable ciphersuites. They can be controlled using the set ServerCipherSuites() method in the SSLParams context object.

Each SSL client has a set of acceptable ciphersuites. They are controlled by using the setClientCipherSuites() method in the SSLParams context object. Please refer to the MySecurityManager.java in any samples directory for more information.

The SSL protocol selects a common ciphersuite acceptable to both client and server. The actual ciphersuite for the connection has to be selected as the first of those, requested by the client that is supported by the server. If there is no common ciphersuite, you would be receiving a Fatal Handshake exception.

Question 5: If I don't set a client or server ciphersuite at runtime by instantiating an SSLParams object (e.g. using the setClientCipherSuites() and setServerCipher-Suites() methods before creating the SSLSocket connection) what ciphersuites are used?

The SSLParams object contains a list of default ciphersuites. These ciphersuites are coded in the SSLParams.java class in the arrays SSLParams.serverCipher-Suites and SSLParams.clientCipherSuites. These are the ciphersuites used if they are not set at runtime.

Digital Certificates

Question 6: What is a digital certificate?

A certificate is used to authenticate an entity. It consists of a public key which is bound to the information about the entity, such as the e-mail address of the entity. Certificates in SSL are in a format called X.509. The corresponding SSLava class that encapsulates X.509 certificates is crysec.X509. The certificate consists of the public key, entity information, and a signature by the entity that issued the certificate.

Certificates used in SSL are of two types:

- RSA
- DSS

They correspond to the algorithm used to create the signature.

Question 7: Who issues or generates certificates?

Certificates are generally issued by Certification Authorities (CAs). CAs verifies the identity of an entity and signs the certificate.

Certificates can be issued by Phaos products such as the J/CA Certification Toolkit (pure Java CA library), as well as the Centuris Public Key Infrastructure Platform (products for building enterprise CAs).

Certificates can also be issued by services such as Verisign or Thawte. SSLava is compatible with the certificate formats issued by such services.

Question 8: What is "server certification" and "client certification"?

Certificates are used to authenticate the computer that is connecting to an SSL connection. For example, an SSL client checks the server's certificate, to verify it. This is called server authentication or server certification.

Similarly, an SSL server can check a client's certificate to verify it. This is called client authentication or client certification.

Except for anonymous ciphersuites, server certificates are required for SSL connections. That is, you need to set a server certificate object in the SSLParams object before constructing a server socket.

However, client certificates are optional. To request client certificates, use the flag setRequestClientCert in the SSLParams class, before constructing an SSLServer-Socket.

Question 9: How do I obtain the certificate of the peer? In other words, if I am a client, how do I see the server's certificate, and vice versa?

All the samples, in the PubSub and the Ptp directory, contain details, on how the client can retrieve and authenticate the server's digital certificates.

Similarly, the server can also retrieve and authenticate the client's certificate.

Question 10: How are certificates checked? What is a root CA fingerprint?

Certificates are checked by obtaining the public key of the signer, and verifying the certificate cryptographically.

In practice, the signer's public key is obtained from a certificate. This means that a certificate chain consisting of the certificate of the entity and the certificate of the signer is used by SSL to authenticate the entity.

A certificate chain consists of an arbitrary number of certificates, each of which has signed a predecessor in the chain. The root certificate, also known as the root CA certificate, is the certificate with no predecessor.

FioranoMQ does not perform client-or server-side authentication. The developer, using FioranoMQ SSL needs to code the authentication logic.

Question 11: How do I create a certificate chain for use with an SSL server?

To create a certificate chain for use with an SSL server, you need to have your Security Manager implement the Server Security Manager

Your Security Manager has to implement the following method:

```
public SSLParams getSSLParams () { return m_params; }
```

m_params represents SSLParams containing the SSLCertificate. To create an SSLCertificate object, and initialize the field certificateList:

```
SSLCertificate cert = new SSLCertificate(); cert.certificateList = new Vector();
```

The Vector certificate list corresponds to the certificate chain. Now, add the first element to the certificate chain, corresponding to the server certificate:

```
cert.certificateList.addElement(new X509(new File("server-cert.der")));
```

This is the server certificate. The following certificate in the chain is the certificate of the entity that signed the server certificate, usually, the CA certificate:

```
cert.certificateList.addElement(new X509(new File("ca-cert.der")));
```

Finally, you need to set the private key of the server certificate. An example is:

```
cert.privateKey = new PrivateKeyPKCS8(new File("server-key.der"));
```

Then, you have created a certificate chain object that you can now, set in the SS-LParams context object before the SSL connection, using:

```
m_params.setServerCert(cert);
```

After the SSLParams are created, you need to set these params to the Server-SecurityManager and pass it on to FioranoInitialContext.

```
MyServerSecurityManager serverCertificateHandler = new MyServerSecurityManager (); ic
= new FioranoInitialContext (serverCertificateHandler)
```

All connections issued from the client now uses the properties set in these certificates.

A complete working example of My Server Security Manager can be found in all the sample directories.

Question 12: How is it possible to trust a root CA certificate, if there is no way of verifying that it has been signed by another certificate?

The answer is in the certificate fingerprint mechanism, supported by SSLava. It is possible to create a unique digital fingerprint corresponding to any digital data by using a hash algorithm such as, MD5 or SHA. SSLava uses a 16-byte MD5 certificate fingerprints to store fingerprints of root CA certificates that it trusts.

During the SSL handshake, SSLava verifies each certificate in the certificate chain. When it reaches the root CA certificate, it computes the hash of the root CA certificate and compares it against the trusted fingerprint list. If each certificate is verified correctly, and the root CA fingerprint is on the trusted list, then the certificate chain is considered to be trusted.

The trusted root CA fingerprints are stored in the array, SSLParams.rootCAFinger-prints, and can be set at runtime using the SSLParams method setRootCAFinger-prints(byte).

To compute a root CA fingerprint, use the MD5test program found in the test/ directory, and pass as the argument the DER encoded root CA certificate. This is a 16 byte value.

To customize the certificate verification procedure, you should use the SSLCertificateVerifier class. Please refer to the on-line documentation, in $FMP_DIR/docs/ PhaosHTMLDocs directory.

ACL/ACE based Security

Question 13: What is Access Control List (ACL) based security? What granularity of security does it provide?

ACL/ACE based security, allows the security policy for an application to be controlled externally, through an administration tool. FioranoMQ implements a comprehensive security model with complete support for ACL/ACE based security. ACL"s are attached to destinations (Topics and Queues), allowing the administrator to control access to topics. Access permissions can be set to:

- Publish to a Topic/Queue.
- Create a normal subscription on a Topic/Queue.

- Create a durable subscription on a Topic/Queue.

Fiorano MQ incorporates a security system that is highly granular. Apart from the ACL/ACE based security restrictions on users to access created Destinations, the FioranoMQ server also supports selective sharing of data across servers. Every server can be administered to selectively import and export messages from other servers. Multi-level enterprise security is a necessity for integrating business processes that extend beyond corporate boundaries. FioranoMQ servers provide for built-in firewalls for incoming as well as outgoing messages. The built-in firewall support is provided using Access Control Lists and Access-Control Entries that restrict incoming as well as outgoing messages from the FioranoMQ server. Additional security concerns are addressed by providing publish To () APIs. These APIs allow application developers to selectively publish messages to one or more server(s) in the enterprise network. The server(s) can be identified by either a unique fully-qualified server name or by a unique (InetAddress, port) tuple. Additional proprietary APIs have been added to the FioranoPublisher class to support this feature.

Question 14: Are there any limitations on the use of ACL with Applets?

Netscape communciator, does not come bundled with java.security package, and so ACL/ACE features of FioranoMQ cannot be used by an Applet executing within Netscape. Reference to the following FioranoMQ runtime classes, using Netscape's VM throws a java.lang.VerifierError to occur.

```
fiorano.jms.rtl.User fiorano.jms.rtl.Administrator fiorano.jms.rtl.UserGroup
fiorano.jms.rtl.Neighbour
```

If your JMS Applet is running on Netscape Communicator, it needs to use Fiorano's ACL/ACE based Security APIs, place ACL/ACE based Java 2 Security APIs in the local machine's CLASSPATH.

Question 15: How to use ACL based security with FMQ?

For more information refer to **Chapter 6 FioranoMQ Security** in *the FioranoMQ 2007 Handbook*.

FioranoMQ Scalability, Reliability and Performance

Load Balancing using FioranoMQ Dispatcher Services

Question 1: What is the objective of clustering? What are the software/hardware requirements for clustering?

Server clustering, allows, clients connected on different FioranoMQ servers to exchange information, without being connected to each other's server. Fiorano's load balancing architecture involves the use of Dispatcher-enabled MQServer, to route, incoming client connections, to the least-loaded server in a cluster. (MQ Administrator can set up an instance of MQServer to act as Dispatcher, by using the Fiorano Admin Studio). The dispatcher component of FioranoMQ Server is connected to multiple FioranoMQ servers, which can run on multiple machines, or, the same machine. All of these servers, become part of the cluster that is serviced by the dispatcher.

Clustering requires no additional software/hardware as such other than what is required for FioranoMQ server.

Question 2: What is the best recommended approach for configuring FioranoMQ so as to minimize the single point of failure problem?

This can be achieved through clustering (The FMQ administrator can set up multiple FMQ servers in the cluster to provide dispatcher facilities). This prevents single point of failure.

Question 3: Are client connections automatically distributed to the least used JMS Server in a multiple server cluster?

FioranoMQ's load balancing architecture involves the use of a special Dispatcher application, to route incoming client connections to the least-loaded server in a cluster. Any MQServer can be launched as a dispatcher. All of these servers become part of the cluster that is serviced by the dispatcher. Connecting the servers as neighbors allow topics to be shared transparently, between the clients and servers, to exchange information seamlessly across the servers.

Question 4: Can multiple instances of Dispatcher be started to prevent the dispatcher from being the single point of failure?

You can run multiple instances of the Dispatcher Tool, to have N-MQ Servers (server farms) associated with each Dispatcher. In addition, you can have the same MQServer as a part of multiple clusters.

Question 5: Can I launch multiple instances of dispatcher on the same machine?

Multiple instances of the dispatcher server can be started on the same machine on different ports.

Question 6: Is there any option available for setting backup/failover URLs for FioranoMQ clients?

The failover support of FioranoMQ is enhanced by the use of N-Failover URLs, with which clients can transparently move to a backup server if the primary server goes down.

For more information refer to section **19.7.1 N Failover URLs Support** in the *FioranoMQ 2007 Handbook.*

Question 7: Intermittently, the following exception trace can be viewed, which is causing loss of data in the production system: LEAST_LOADED_SERVER_CALCULATION_ERROR :: JMSException occurred while executing lookupLeastLoaded method of ConnectionFactoryProxy. at fiorano.jms.runtime.pubsub.FioranoTopicConnectionFactoryProxy.lookupLeast-Loaded (Unknown Source) at fiorano.jms.runtime.pubsub.FioranoTopicConnectionFactoryProxy.createConnec-tion (Unknown Source) at fiorano.jms.runtime.pubsub.FioranoTopicConnectionFactory.createConnection-Proxy (Unknown Source) at fiorano.jms.runtime.pubsub.FioranoTopicConnectionFactory.createTopicConnec-tion (Unknown Source) What is the reason for this??

This exception occurs due to the fact that the dispatcher server was unable to route or load balance the new connection request to a member server of its cluster.

This can occur due to the following reasons:

1. The dispatcher was unable to retrieve the least loaded server from the cluster due to some error.
2. The dispatcher was unable to connect to the least loaded server or the preferred server.
3. The dispatcher was able to connect to the least loaded server of the cluster but that server does not have the Connection Factory object being used to create the connection from the client side. If all the servers are running on the same LDAP Admin store then this should not occur. To ensure that the connection factory object exists on all the member servers, run the FioranoMQ AdminTool on all the servers and check the list of Topic Connection Factories under the TCF tab.

For case 2 the reason could be that the concerned member server of the cluster is not active and running.

In order to know the exact cause of the exception, use the following catch block ,where this exception is being thrown ,at client side:

```
catch(FioranoException e) { e.printCompleteStackTrace() }
```

If you are using the ExceptionListener, you can additionally use the e.printCompleteStackTrace() call in the onException() method.

In addition, check the mqout and mqerr logs, for any exception that might have occurred on the FioranoMQ server side.

To gain information about these exceptions at the server side, increase the Log levels of the following parameters on the dispatcher server:

```
FMQDispatcherServices=10 Dispatcher=10
```

Increase these Trace levels using the Log Manager tab of the Fiorano Studio.

Question 8: How does FioranoMQ provide for High Availability of its data-store?

FioranoMQ runs over a proprietary file-based data store where all persistent information is stored in files. To add support for High Availability to FioranoMQ servers running in a cluster, use RAID/SAN hard disks for storage of persistent data.

In case of RAID, the information stored on one disk is replicated on one or more disks. Thus, if a primary server writing on say, disk1 crashes, the secondary server would access, all information from the replicated disk2. Setting this up for a cluster provides High Availability support within that cluster of FioranoMQ servers.

Message Replication

Question 9: How does FioranoMQ support Message Replication across different instances of FioranoMQ server?

FioranoMQ supports Message Replication, over topics, using the FioranoMQ repeater and server-to-server communication over queues, using the FioranoMQ bridges. These modules, coupled with the FioranoMQ dispatcher can be used, to set up a fully loaded, balanced cluster of FioranoMQ servers with support for failover.

Server-to-Server Communication - FioranoMQ Neighboring

Question 10: What does server- to- server communication mean? How can I make use of it?

Server-to-server communication, allows two or more JMS servers, to share information in a controlled manner. This allows clients on one server to exchange information with clients connected to another server in the cluster, without each client having to connect explicitly to each server, allowing large numbers of concurrent client connections in the cluster.

Server- to- server communication, can be used, to logically distribute the system domain into closed systems with controlled boundaries for inter-system communication.

Question 11: What is a repeater?

Repeater is a JMS Client Application, which is responsible for server -to -server communication by repeating messages from one topic to the other. The repeater is started as a stand-alone Java Application and can run in secure and non-secure mode. For more information, refer to **Chapter 10 Repeater** in the *FioranoMQ 2007 Handbook*.

Question 12: How different is the repeater from the dispatcher?

Repeater and Dispatcher are completely different components of the FioranoMQ Enterprise suite. The Dispatcher manages a cluster of servers, and is responsible for routing incoming client requests to the least-loaded server in the cluster. The Repeater on the other hand, is a JMS Client Application that is responsible for server- to- server communication by repeating messages from one topic to the other.

.

Question 13: Can I run a single instance of Repeater to connect multiple machines in a cluster?

You can run a single instance of Repeater, to connect multiple machines in the cluster.

Question 14: What information does a Repeater propagate?

Repeater propagates only Topic related information. In addition, FioranoMQ supports the concept of a distributed queue using Bridges. For more information on configuring bridges, refer to **Chapter 11 Bridge** in the *FioranoMQ 2007 Handbook*.

Question 15: How do I have an application on one host send point-to-point messages to an application on another host?

For more information, refer to **Chapter 11 Bridge** in the *FioranoMQ 2007 Handbook*.

Question 16: How do I send Object Messages from Publisher on one MQ Server to a Subscriber on another MQ Server?

To publish and subscribe to Object Messages in a server cluster, ensure that all the intermediate routers have a reference, to the published Object in its CLASSPATH.

Question 17: Do network failures in a server cluster stall publishing of messages to the local server?

No. An application can continue to publish to the local server even in case of network failures between remote servers. The local JMS server stores messages and forwards them to remote nodes as soon as the network comes up again.

Question 18: How do I connect two servers?

FioranoMQ architecture allows multiple FioranoMQ Servers to be connected together, allowing clients connected on one server to exchange information with clients connected on any other MQ server. Servers can be connected (both over LAN and WAN), using FioranoMQ Repeater. All server-to-server communication is handled transparently by FioranoMQ software and the client application does not need to be modified it, in any way. MQ Servers can be part of the same LAN or can be spread across multiple WANs. Repeater is enabled for HTTP/HTTPS as well. For more information, refer to **Chapter 10 Repeater** in the *FioranoMQ 2007 Handbook*.

Question 19: How do I perform load balancing in a cluster?

Fiorano MQ's load balancing architecture involves the use of Dispatcher-enabled MQServer to route incoming client connections to the least-loaded server in a cluster. FioranoMQ Sever can also act, as the dispatcher, to route incoming client connections to least loaded server in a cluster. MQ Administrator can set up an instance of MQServer in, from the Admin Tool.

Question 20: How do I integrate FioranoMQ with MSMQ, MQSeries, TibRv or other JMS servers?

FioranoMQ Bridge,can be used to forward requests and responses between two different message queuing systems. Communication to any other JMS vendor is done using the standard JMS API. Communication to MSMQ is done, using MSMQ Java APIs and communication to TibRv is done using TibRv Java APIs. In addition, the FioranoMQ Bridge sends messages as JMS Messages to the target JMS vendor. The FioranoMQ Bridge is configured using XML based configuration files that allow a single instance of the Bridge to "bridge" any number of messaging servers. For more information, refer to **Chapter 11 Bridge** in the *FioranoMQ 2007 Handbook*.

Concurrent Connections

Question 21: How does the FioranoMQ Server achieve scalability?

For details on Fiorano's Scalable Connection Manager, refer to the Whitepaper or **Chapter 3 Connection Management** in the *FioranoMQ 2007 Handbook*.

Question 22: The following exceptions were thrown while running FioranoMQ. What do these exceptions mean and why are they thrown?

Exception 1

```
java.net.SocketException: Too many open files at
java.net.PlainSocketImpl.accept(Compiled Code) at
java.net.ServerSocket.implAccept(Compiled Code) at
java.net.ServerSocket.accept(Compiled Code) at
fioranomq.t3.srvr.ListenThread.run(Compiled Code)
```

Exception 2

```
java.io.IOException: Too many open files

at java.lang.UNIXProcess.forkAndExec(Native Method) at
java.lang.UNIXProcess.(UNIXProcess.java:54) at
java.lang.UNIXProcess.forkAndExec(Native Method) at
java.lang.UNIXProcess.(UNIXProcess.java:54) at java.lang.Runtime.execInternal(Native
Method) at java.lang.Runtime.exec(Runtime.java:551) at
java.lang.Runtime.exec(Runtime.java:477) at java.lang.Runtime.exec(Runtime.java:443)

...
```

Cause of the problem

Both the exceptions thrown above,indicate a problem, related to resources of operating systems being used. This problem usually occurs, when several users are connected to the server at the same time. Java opens many files, in order to read in the classes required to run your application. High volume applications can use a lot of file descriptors. This could lead to a lack of new file descriptors. The number of file differs from one operating system to another.

Windows

By default, open file handles (file descriptors) limit is set to 16,384.

Linux

The Admin user can set the file descriptors limit in the etc/security/limits.conf configuration file, as shown in following example.

```
soft nofile 1024 hard nofile 4096
```

A system-wide file descriptor limit can also be set, by adding the following three lines to the /etc/rc.d/rc.local startup script:

```
# Increase system-wide file descriptor limit. echo 4096 > /proc/sys/fs/file-max echo
16384 > /proc/sys/fs/inode-max
```

Solaris

You can modify maximum value defined in rlim_fd_max. By default, the value is set to 65,536.

HP-UX

Nfile, defines the maximum number of open files. This value is usually determined by the formula: ((NPROC*2)+1000) where NPROC is usually: ((MAXUS-ERS*5)+64). For a MAXUSERS of 400, this works out to 5128. You can usually set it higher, maxfiles is the Soft file limit per process and maxfiles_lim is the hard file limit per process.

AIX

The file descriptors limit is set in the /etc/security/limits file and its default value is 2000. This limit can be changed by the ulimit command or the setrlimit subroutine. The maximum size is defined by the constant OPEN_MAX.

You need to first monitor file descriptors and get a status of open files and other potential issues. This, then needs to be followed by increasing, the number of file descriptors according to the operating system.

Known FioranoMQ Server Issues

Increasing the number of File Descriptors usually fixes this kind of problem, but you also need to make sure, that the FioranoMQ Server as an application does not use too many files and that open files are getting closed properly so that file descriptors are released.

All issues reported to Fiorano Customer Support have involved a lack of file descriptors or an overflow of the descriptor table. This problem always occurs when the OS notifies the java process that no new file descriptor can be allocated. In this case, you need to increase the number of file descriptors.

Crash/Disaster Recovery

Question 23: What recovery mechanism does FioranoMQ have in case of disaster/crash of the server?

FioranoMQ provides for recovery and extraction of its data store in case of any crash or disaster using the Analyzer/Extractor Toolkit. With this toolkit, all persistent information can be analyzed and extracted to a new location for further use.

For more information, refer to **Chapter 23 Fiorano DB Recovery Tool** in the *FioranoMQ 2007 Handbook*.

Other Issues

Question 24: What throughput degradation would occur if all messages were replicated to queue copies on several servers in a cluster?

In most cases, clustering for messages is achieved using store and forward mechanism, where the messages are moved around in the network. If a copy is created for each queue, then receivers needs to identify the message to the point, where they have already consumed messages, to avoid receiving duplicate messages on fail-over. Queue, forwarding works as a background daemon, which may increase end-to-end time delay (between send and receive) by up to 40%. However, the throughput of the cluster (defined as the number of messages handled by the server cluster) remains largely unaffected by clustering.

JMS - Core Programming Issues

This section answers frequently asked questions regarding the usage of JMS APIs for writing simple/complex applications.

Compiling/Running JMS Application using FioranoMQ

Question 1: How do I compile JMS client applications using FioranoMQ?

FioranoMQ installation on Windows, Unix and Solaris comes bundled with batch scripts to compile JMS Applications. On Windows you can find compile-client.bat in the fmq\bin directory of Fiorano's installation. On Unix environments, you can find compile-client.sh in the fmq\bin directory of the installation. Please follow the instructions in the batch script to set appropriate environment variables to compile the Client JMS Applications.

Question 2: How do I run JMS client applications using FioranoMQ?

FioranoMQ installation on Windows, Unix and Solaris comes bundled with batch scripts to run JMS Applications. On Windows you can find run-client.bat in the fmq\bin directory of Fiorano's installation. On Unix environments, you can find run-Client.sh in the fmq\bin directory of the installation. Please follow the instructions in the batch script to set appropriate environment variables to run the Client JMS Applications. FioranoMQ does not require any changes in the VM parameters to run the JMS Clients.

Question 3: How do I run a sample FioranoMQ application under Eclipse?

Following is a step-by-step procedure for running a sample Fiorano application under Eclipse. It has been tested under version 2.0. Before executing these steps, please make sure that your FioranoMQ server is running.

1. Create a new project from **File > New > Project**.
2. Select **java** in the left pane and **java Project** in the right pane. Click **Next**.
3. Give a project name (say FioranoMQ).
4. In the Resource perspective select **FioranoMQ**, right-click and select **properties.**
5. In the left pane select **Java Build Path.**
6. In the right pane, select **Libraries** and click **Add external Jars**.
7. Select the following zips and jars from the lib directory of your Fiorano Installer.
    - fmq-common.jar
    - fmq-rtl.jar
    - jndi.jar
8. Click **OK** to close Properties of FioranoMQ dialog box.
9. Again select FioranoMQ in the **Resources** perspective and from Menu bar select **File >Import.**
    i. The Import dialog box is displayed. From the (Import) Select dialog box, select **File System** and click **Next.**

   ii. In the (Import) File System Dialog box, click **Browse** and select the directory of the FMQ samples (which you want to run say C:\MyTemp\Fiorano\FIORAN~1\fmq\samples\PubSub\P ubSub). The PubSub folder is added in the left pane, and various samples is added in the right pane.
   iii. From the right pane, select **Subscriber.java.**
   iv. Click **Finish.**

10. Right-click **FioranoMQ** in Resource perspective and select **rebuild** project. This compiles all the java files automatically in this **project.**
11. Double-click **Subscriber.java**. This opens the source code for Subscriber in the IDE.
12. Click **java perspective** on the left panel. Click **Run** from the drop-down list on the toolbar and select **Java Application** from the **Run As** menu. This launches the class in the active editor, or the selected class in the Navigator, as a local Java application.
13. After executing the above step, you can see the following output in the console:

```
Created InitialContext :: javax.naming.InitialContext@7077e Creating topic connection
Creating topic session: not transacted, auto ack Creating topic, subscriber Ready to
subscribe for messages :
```

14. Start a publisher in the dos prompt, and publish some messages. These messages are received by the subscriber.

Similar steps can be followed for running the publisher and other FMQ related sample applications in Eclipse.

Question 4: How do I run a sample FioranoMQ application under Visual Age 3.0?

The sample applications, bundled with FioranoMQ, can run successfully, on Visual Age 3.0.

For example, consider a project called Ayrton to which you want to add the samples (in the Workbench Window). In the project Ayrton, there exists a package called Ayrton. Following are the instructions to compile the samples:

1. Right-click the project Ayrton and select **Import** from the menu. The **Import dialog box** is displayed. Select the Jar file and click **Next.**
2. The **Import from a jar/zip file** dialog box is displayed. Choose jndi.jar (located in Fiorano Installation Directory\fmq\lib directory). Select **Project: Ayrton** and click **Finish.**
3. Repeat the above step for fmq-rtl.jar and fmq-common.jar (located in Fiorano Installation directory\fmq\lib directory).
4. Navigate to the package Ayrton, right-click it and select **Add > class**. A **create class Dialog Box** is displayed. Set the following parameters for this dialog box:
   - Project: Ayrton
   - Package: Ayrton
   - Class Name: Subscriber
   - Superclass: java.lang.Object
   - check: Browse the class when finished

- check: Compose the class visually

5. Click **Next**. The **Attributes** dialog box is displayed. In the Add import statements, add the following packages:
    - java.io.*;
    - java.net.*;
    - java.utility.*;
    - javax.naming.*;
    - javax.jms.*;
    - fiorano.jms.rtl.*;

In **Which interface should class implement**, add the following:

- javax.jms.MessageListener
- javax.jms.ExceptionListener

6. Click **Finish.**
7. You have a Subscriber class in the Ayrton -- Ayrton-- Subscriber Hierarchy. Further, under Subscriber the following would be present:
    - Subscriber()
    - main(String [ ])
    - onException(JMSException)
    - onMessage(Message)
8. . Enter the following code for main(String [ ])

```
Subscriber subscriber = new Subscriber (); try { // 1. Create the InitialContext
Object used for looking up // JMS administered objects on the Fiorano/EMS // located
on the default host. // Hashtable env = new Hashtable (); env.put
(Context.SECURITY_PRINCIPAL, "anonymous"); env.put (Context.SECURITY_CREDENTIALS,
"anonymous"); env.put (Context.PROVIDER_URL, "http://localhost:1856"); env.put
(Context.INITIAL_CONTEXT_FACTORY,
"fiorano.jms.runtime.naming.FioranoInitialContextFactory"); InitialContext ic = new
InitialContext (env); System.out.println ("Created InitialContext :: " + ic); // 1.1
Lookup Connection Factory and Topic names // TopicConnectionFactory tcf =
(TopicConnectionFactory) ic.lookup ("primaryTCF"); Topic topic =
(Topic)ic.lookup("primaryTopic");
```

```
// 2. create and start a topic connection System.out.println("Creating topic
connection"); TopicConnection topicConnection = tcf.createTopicConnection(); //
Register an Exception Listner topicConnection.setExceptionListener (subscriber);
topicConnection.start (); // 3. create topic session on the connection just created
System.out.println("Creating topic session: not transacted, auto ack"); TopicSession
topicSession = topicConnection.createTopicSession(false,1); // 4. create subscriber
System.out.println("Creating topic, subscriber"); TopicSubscriber topicSubscriber =
topicSession.createSubscriber(topic); // 5. install an asynchronous listener/callback
on the subscriber object // just created System.out.println ("Ready to subscribe for
messages :"); topicSubscriber.setMessageListener (new Subscriber ()); } catch
(Exception e) { e.printStackTrace (); }
```

9. Enter the following code for onException(JMSException):

String error = e.getErrorCode (); System.out.println (error);

((fiorano.jms.common.FioranoException)e).printCompleteStackTrace();

10. Enter the following code for **onMessage(Message):**

```
try { // When a message is received, print it out to // standard output // TextMessage
textmsg2 = (TextMessage)msg; System.out.println("Received : " + textmsg2.getText() );
} catch (JMSException e) { e.printStackTrace (); }
```

The above code can also be found in Fiorano's Installation directory/fmq/samples/
PubSub/PubSub/Subscriber.java

Initial Context

Question 5: What is an Initial Context object?

FioranoMQ is a JNDI-compliant messaging server. Initial context is a bound handle to a directory server, which is used to look up administered resources. In FioranoMQ Server, the directory server is embedded into the base MQ Server, hence there is no special lookup object. In addition, FioranoMQ provides for integration with external JNDI compliant directory servers.

Question 6: Does looking up administered object hold up any references in remote VM?

Lookup is a Directory service call, which looks up administered resources and obtains information about the same. Lookups do not result in any resource allocation on the MQ Server, or the corresponding Directory Server.

Question7: In case of FioranoMQ, what is the name of the InitialContextFactory to be used for JNDI lookup. Also, what does a typical jndi.properties file to be used by FioranoMQ clients comprise of?

Incase of FioranoMQ, the Initial Context Factory class is fiorano.jms.runtime.naming.FioranoInitialContextFactory, which can located in fmq-rtl.jar (found in \fmq\lib folder of Fiorano's installation directory) .

Thus, default jndi.properties file, incase of FioranoMQ, is as follows:

```
java.naming.factory.initial=fiorano.jms.runtime.naming.FioranoInitialContextFactory
java.naming.provider.url=http://localhost:1856
java.naming.security.principal=anonymous java.naming.security.credentials=anonymous
```

This can be used by JMS clients to lookup ConnectionFactory and destination objects from an instance of FioranoMQ server, running on the same machine over TCP (default protocol).

Connection Factory

Question 8: What is the role of a Connection Factory? Do I need multiple Connection factories?

A connection factory is a looked up resource, which contains information (inclusive of the IP address) about a named FioranoMQ Server. A connection factory reference can be used to create multiple connections to a FioranoMQ Server. The following code snippet shows how a single instance of the connection factory can be used to create multiple connections to the MQ Server. So, there is no explicit need to create multiple instance of the Connection Factory.

```
TopicConnectionFactory tcf = (TopicConnectionFactory) ic.lookup ("primaryTCF"); //
Create first connection TopicConnection tc1 = tcf.createTopicConnection(); // Create
second connection TopicConnection tc2 = tcf.createTopicConnection();
```

JMS Connection

Question 9: Do I need different connection instance for every session?

A connection instance can be used to create any number of sessions. Typically a JMS Client creates a connection and all publishers/Subscribers are multiplexed on this connection using one or more sessions. Connection is a heavy weight object and creation of a session should be avoided unless necessary. In some cases, sophisticated client applications may use multiple connections for connecting to multiple JMS Servers (Connection Factories) or for achieving improved performance for speed critical applications.

Question 10: Does JMS allow for creation of multiple connections? When is it required?

The JMS API does allow multiple connections to be created by a single application to either the same or to different JMS servers. Multiple connections might be required, for instance, in cases where different topics of interest are distributed across different servers. If a single application uses a very large number of topics, then multiple connections to a single FioranoMQ server might be of value by helping to distribute the load across multiple threads (since each connection runs in a separate thread on the server).

JMS Sessions

Question 11: What role does a session play in JMS? How heavy weight is a session? What does the statement "executes in a single Threaded Context" mean?

A JMS Session plays two critical roles:

- Manages transactions for publishers and subscribers.
- Serializes the execution of callbacks for all registered consumers of the Session.

Sessions are lightweight objects. Each session is operated upon, by at most, one thread at a time and all callbacks registered on a single session are executed serially. Sessions control the "level of desired concurrency" in the system. To illustrate this point, consider an application that wishes to create subscriptions on topics T1 and T2 on a session S1.

If messages received on both topics T1 and T2 are to be handled by one single process (one message at a time), then the procedure is as follows:

```
TopicConnection con = TopicConnectionFactory.createConnection (); TopicSession ts =
con.createTopicSession (false, 1); ts.createSubscriber (T1); ts.createSubscriber (T2);
If messages received on T1 and T2 can be processed concurrently, then the approach to
follow will be: TopicConnection con = TopicConnectionFactory.createConnection ();
TopicSession ts1 = con.createTopicSession (false, 1); ts1.createSubscriber (T1);
TopicSession ts2 = con.createTopicSession (false, 1); ts2.createSubscriber (T2);
```

In the second case a separate Session objects is used for each of the consumer, allowing both of them to receive messages concurrently from the MQ Server.

Question 12: Can FioranoMQ handle transacted sessions?

FioranoMQ supports client-transacted sessions as defined in the JMS specifications.

Question 13: Can MultipleSessions be created on a single connection?

Yes, you can create large number of sessions on the same connection.

Transactions

Question 14: What are basic transactions?

Transaction is a means of grouping a set of operations as an atomic operation, which guarantees consistency. Publisher and Subscriber transactions can be defined separately as follows:

A Publisher transaction is used to group a set of published messages into an atomic unit. FioranoMQ Server guarantees that either all the messages in this unit are published or none of them are published from the client to the server.

A Subscriber transaction serves to group a set of consumed messages and acknowledges or rollbacks this unit.

Distributed Transactions

Question 15: What are distributed transactions and two phase commit protocols?

Distributed transactions are the implementation of all JMS interfaces starting with "XA" prefix. Distributed transactions span over multiple client and server processes and uses two-phase commit protocols for maintaining consistency of transactions.

Temporary Destinations

Question 16: What is a temporary Topic/Queue? How is it used?

A temporary destination is created on the fly by an application and is destroyed as soon as the connection creating the temporary destination is closed. It is typically used for creating temporary inboxes for receiving directed replies (Request-reply interactions). Name of each temporary destination is guaranteed to be unique.

Publishing Messages

Question 17: What are the modes of Publishing messages? Which one should I use?

Messages can either be published as Non-Persistent messages or Persistent messages. All persistent messages are logged into the FioranoMQ Server. Thus if a message is being published which cannot be missed by any of the Subscribers (active as well as passive) then the message should be marked Persistent.

Question 18: What is the overhead involved in publishing Persistent messages?

Publishing persistent messages comes with an overhead of message logging. When a message is marked Persistent then the publish/commit call does not return until the data is persisted in the off-line database by the MQ Server.

Subscribers

Question 19: What are the types of Subscribers? What are the scenarios in which they are used?

Subscribers can either be Durable Subscribers or Non-Durable Subscribers. If a subscriber wishes to receive all published messages, even those that were published while the subscription was inactive, then the subscriber must register as a durable subscription with the MQServer.

Guaranteed Messaging

Question 20: In FioranoMQ, can messages be persisted/stored on the client side in case the message is not able to reach the FioranoMQ server for some reason?

Client side caching of messages is an important feature that is very beneficial in avoiding any message loss in case of server crash or network failures in a cluster. This feature has been introduced since FioranoMQ 7.1.

Question 21: Which database does Fiorano JMS Server use for storing persistent messages?

FioranoMQ Server uses a proprietary file-based database written in 100% Java. In addition, it uses an index based flat file schema for added performance. The performance of this database is multiple orders better than most generic databases and keeps the server lightweight.

Additionally, FioranoMQ server can be configured to store messages in some RDBMS also. For details of configuration and supported databases please refer to **the Chapter 5 Configuring Message Store** in the *FioranoMQ 2007 Handbook*.

Question 22: Does Fiorano JMS Server guarantee message delivery?

Yes, all persistent messages are logged in a persistent database and redelivered to the durable subscribers when they log in next time. FioranoMQ server maintains state of each of the durable subscription registered with it.

FioranoMQ and Non-Persistent Messages

Question 1: JMS specifications do not guarantee delivery of Non-Persistent messages. As FioranoMQ server is 100% compliant with the JMS specifications, can it also drop NP messages in certain circumstances? In case it can, can you please describe when it drops the NP messages? Also let us know how can the configurable parameters be tweaked so that dropping of NP messages can be reduced significantly?

Yes, FioranoMQ server is 100% compliant with the JMS specs. It can drop NP messages on topics in one possible scenario. Although very rare, this scenario may surface in some of the business applications.

FioranoMQ server may drop NP messages only in one of the publish-subscribe model. However, it never drops them in point to point messaging model. Let us discuss the pubsub scenario in which it may drop the NP messages.

Consider that there are n subscribers (n>=1) listening for messages that are published on a topic. If all these subscribers are processing the messages at a rate which is comparable to the publish rate, then the NP messages never drops.

But if one or more subscribers are very slow, compared to the publish rate, then FioranoMQ server may drop the NP messages. Before actually dropping the NP messages, FMQ server attempts to slow down the rate at which the messages are published (if configured). But even after this slows down, the publisher may keep on publishing the NP messages at a considerable rate (if slow down parameters are configured to values which do not slowdown the publisher much), then some NP messages may be dropped by the server.

An FMQ administrator can modify the server configurations to minimize no of messages from being dropped, or can configure the topic so that the no of messages published on it are never dropped.

Configure TopicMetaData

While creating a topic, the FMQ Administrator can specify whether the messages can be dropped or not. He can use the following API in the topic metadata object to disable message dropping.

```
public void disableMessageDropping()
```

By default, topics are configured such that no messages can be dropped. Therefore, the administrator needs to explicitly and invoke the above API to disable the message dropping.

This disabling of message dropping, may lead to drop in, throughput the particular topic.

The above API is present only in FMQ 7.5 onwards. In case you are using an earlier version of FMQ, please try configuring the server so that it never drops the no messages.

Configure Server

An FMQ administrator can configure the server in such a way, that the server never drops the NP messages in any practical scenario. Modifying the default configurations may lead to blocking of the publishers, which means that the publish rate may drop down, when the subscriber rate is very slow. However, the publisher publishs at its normal speed, as soon as the slow subscriber closes, or starts processing messages at a fast pace.

Before discussing the exact parameters that needs to be modified, let us discuss the internals of the server.

FMQ server stores the non-persistent messages in its in-memory buffers. The non-persistent messages are dropped when no more messages can be added to the in-memory buffers i.e. when the in-memory buffers start overflowing. The administrator can configure the size of these in-memory buffers, as per their publish-sub-scribe rate, such that they never overflow. Administrators can also control the rate at which these buffers are filled by specifying the time interval for which publishers should wait before attempting to add a new message.

Configuring ExTopicSubscriberHandle.BasicAllowedSize parameter

This variable depicts the size of in-memory buffers of each subscriber instance. The default size of the subscriber buffer is 1024KB. Increase this value to 2048KB or 4096KB or even higher depending upon the message traffic using Fiorano Admin Studio in the offline mode. Perform the following steps to set the value of this parameter:

1. Invoke the Fiorano Admin Studio.
2. **Select Tools > Configure FioranoMQ** from the menu bar, select FioranoMQ folder in the resulting Select Profile Directory dialog box and click the **Open** button. This shifts the FMQ environment to the offline mode.
3. Now, navigate to **FioranoMQ > Jms > PubSub > PubSubManager** in the Server Explorer pane. The properties of the PubSubManager are displayed in the Properties pane.
4. Select the ellipsis against the parameter BasicAllowedSize. Type in the new value in the resulting BasicAllowedSize dialog box and click at the **OK** button.
5. Right-click the **FioranoMQ domain** in the Server Explorer and select the **Save** option from the resulting shortcut menu.

In case, if the subscribers listening on a topics are processing the messages fast enough that these buffers do not fill up to the brim. In such a case, the non-persistent messages never drops.

FioranoMQ and Application Servers

Question 1: How do I integrate FioranoMQ with an Application Server?

FioranoMQ integrates seamlessly with many Application Servers such as JBoss and Oracle9i. There are three ways in which Application Server users can leverage the functionalities of JMS:

- Implementing advanced JMS APIs to support Application Server Integration
- Using Message Driven beans (MDBs)
- Integrating FioranoMQ with an EJB Application Server

For more information, refer to **Chapter 24 Application Server Integration** in the *FioranoMQ 2007 Handbook*.

Question 2: Is it possible to develop Message Driven Beans (MDBs) through the integration of FioranoMQ and JBOSS?

Yes, you can integrate FioranoMQ with JBoss to implement the Message Driven Bean functionality. For more information, refer to the section **24.5 FioranoMQ 2007 - JBOSS 3.2.5 Application Server** in the *FioranoMQ 2007 Handbook*.

Question 3: How do I integrate FioranoMQ with Oracle Application Server?

The FioranoMQ Server can be successfully integrated with Oracle Application Server. For more information, refer to the section **24.12 FioranoMQ 2007 - Oracle 10g Application Server** in the *FioranoMQ 2007 Handbook*.

Question 4: How can ATG Dynamo Message Service be configured to work with FioranoMQ Server?

To configure Dynamo Message Service to work with FioranoMQ, declare the provider in the Patch Bay configuration, and then configure the appropriate message sources and sinks to use the Destinations in that provider. For more information refer to the section **24.6 FioranoMQ 2007- ATG Dynamo Message Service** in the *FioranoMQ 2007 Handbook*.

Question 5: If I have multiple queues, do I need to add all of them in the application.xml file of OC4J ? If so, how can that be done?
value="http://172.31.5.5:1856"/>
name="resource.names"value="primaryQueue"/>

Multiple resources can be specified by separating the values with comma, for example,

```
<property name="resource.names" value="primaryQueue,primaryQCF,secondaryQueue" />
```

Question 6: With HA, do I have to add the backupurl in this configuration file too? If so how would the syntax look like?

To make the MDBs work with HA and Backup URL, AllowDurableConnection needs to be passed as property, for example,

```
<property name="java.naming.factory.initial" value="fiorano.jms.runtime.naming.Fiora-
noInitialContextFactory" /><property name="java.naming.provider.url"
value="http://localhost:1856" /><property name="BackupConnectURLs"
value="http://localhost:2856" /><property name="AllowDurableConnections" value="true"
/><property name="resource.names" value="primaryQueue,primaryQCF" />
```

FioranoMQ and Applets

Developing and deploying Applets using FioranoMQ

Question 1: How do we develop Applets using FioranoMQ?

Answer: FioranoMQ applets are written in exactly the same way as normal applications are written except for a minor change during the initialization of a connection with the server.

Browser security restrictions allow uncertified Applets to establish connections only with the machine from which the applet is downloaded, i.e. the web server machine. This requires that unlike normal applications (which can connect to a FioranoMQ server running on any machine), uncertified applets are allowed to connect only to the machine from which the applet is downloaded.

(**Note:** Certified applets are allowed to connect to any host to which the web client has authorized access by the certificate provider. Different levels of permissions can be set for certificates from different certificate venders by the web client. An Applet carrying a certificate automatically invokes a window on web clients which do not allow it requisite permissions, asking if the user is interested in granting the permissions. Security parameter for all certificate holders can be explicitly set by using the advanced security options of popular browsers such as Netscape and Internet Explorer.)

To use FioranoMQ services within an applet, the following steps need to be taken:

1. The FioranoMQ Server must be installed on the same machine as the web-server.
2. The IP address of the web-server machine needs to be provided by the Applet to create an instance of the class FioranoInitialContext. This IP address can be statically coded within the applet, or obtained dynamically using the generic method *Applet.getCodeBase()* from the *java.awt.applet* package. The rest of the code remains exactly the same as that for normal FioranoMQ applications.

Question 2: How do we deploy Applets using FioranoMQ?

Deployment of FioranoMQ applets is same as deployment of any other applet on a web server. The applet is developed as specified in the section above.

Once the applet has been developed, it is compiled and class files are created for the same. Once this is done, an HTML file using the applet tag is created. For example, consider an applet "`MyApplet.java`", which is compiled into the class file "`MyApplet.class`".

Presented below is a sample HTML file (MyApplet.html) that runs the Applet - "MyApplet.class".

```
//////////// Source listing -- "MyApplet.html" ////////////////////////
<html>
<head><title>Title of Applet Page</title></head>
<body>
<hr>
<applet code="MyApplet.class" width=400 height=310
archive="..\..\..\..\extlib\jms\jms.jar", "..\..\..\lib\client\fmq-client.jar">
</applet>
<hr>
</body>
</html>
```

The important section is the one with the 'Applet' tag. This has the following sections:

1. code=<name of the class which extends the Applet class>
2. width=<width of applet window, in pixels>
3. height=<height of applet window>
4. archive="jms.jar and fmq-client.jar <-- Relative path of the FioranoMQ JMS jar and runtime library jar files"

Out of these the mandatory attributes are code, width and height.

A complete multi-user chat applet developed using JMS APIs can be found in the samples\applets directory of the FioranoMQ distribution. This directory contains the files MultiChat.java and Login.java.

Applet Support

Question 3: Can I use JMS from within Web based applets?

The Fiorano runtime library can be used within web-based applets on popular browsers, namely, Netscape Communicator (4.0 and above) and Internet Explorer

(4.0 and above). This allows developers to take advantage of the location independent communication APIs implemented in JMS.

Question 4: I am building an applet to communicate with FioranoMQ's messaging server. Does the applet need to make fmq-rtl.jar available? Does the applet need all the classes?

You would need to bundle fmq-rtl.jar and fmq-common.jar with your applets.

Question 5: When I resize a page in Netscape, the entire page gets reloaded and if there is an applet on the page the Applet's stop and start methods are called, respectively. Does this action break the JMS connection?

Resizing the Applet does not break the connection. While coding your Applet, ensure that the connection is not created every time start() is called and the connection does not get deleted when stop() is called.

Question 6: I ran the sample applet using AppletViewer and it worked properly, but when I run it in Internet Explorer I get an error. Could it occur due to inability to find the localhost?

Internet Explorer’s SecurityManager does not allow socket connections to the local machine. You can try connecting the Applet to MQServer running on different machine.

Question 7: When I run the applet using AppletViewer, I get a SecurityException. Do I need to change any environment settings?

It is possible that Fiorano library and its dependencies have not been assigned permission to make a socket connection with FioranoMQ. Check the security policy of your webserver.

The following policy, grants permission,for any application to use the applet:

```
grant { permission java.security.AllPermission "", ""; };
```

Other Applet Issues

Question 8: What causes the exception :IllegalAccesserror:could not create TopicConnectionFactory?

The illegal Access Error is a java.lang error, thrown when an application attempts to access or modify a field, or to call a method that it does not have access to. Typically, this error is captured by the compiler. This error can occur only at run time if the definition of a class has incompatibly changed. This is not a FioranoMQ Error. For example, this error can occur while trying to create a TopicConnection-Factory, which is an Administered Object (such Administered objects can only be created/destroyed by the Administrator). For this, you need to use the Administration APIs (for details, refer to **Chapter 22 Administering FioranoMQ Server Using APIs** in the *FioranoMQ 2007 Handbook*) after connecting to FioranoMQ as an administrator.

Question 9: What causes the exception :"could not calculate least loaded server”

This issue is due to mismatch of client side libraries with FioranoMQ. Hence you need to use the latest libraries.

Question 10: Why does the MQ applet not work in Internet Explorer 5.0?

Internet Explorer 5.0 has issues while using JNDI (specifically the call - "new InitialContext (env)" which fails with a security Exception due to a bug in JNDI). The solution is to directly use Fiorano’s JNDI context factory i.e. the code in Applet which was originally as follows:

```
Hashtable env =...InitialContext ic = new InitialContext (env);
```

```
needs to be changed to :
```

```
Hashtable env =...Context ic = (new
fiorano.jms.runtime.naming.FioranoInitialContextFac-tory ()). getInitialContext(env);
```

An alternative approach is to use the latest JRE 1.3 plug in technology that results in IE using the Sun's VM, rather than the embedded one.

Question 11: Does downloading the applet from Netscape Browser require SSL certification?

SSL is not necessary for downloading from Netscape Communicator.

Question 12: Is the only way to package the zip file with our applet jar, is to unzip them into the same folder that we use to build our applet jar file? What about applet signing?

You do not have to unzip the archives. You can use multiple archives in the archive tag of html file. If you are using jre 1.3 plug in, you can use an HTML converter toolkit that generates an html that works well with all the browsers.

To sign jar files and modify security permission, please refer to the following links:

http://java.sun.com/j2se/1.3/docs/tooldocs/win32/jarsigner.html
http://java.sun.com/j2se/1.3/docs/guide/security/permissions.html

Question 13: I tried to run a sample applet code on JRun web server on a particular machine, using IE5.0 as browser, publishing some text from the publisher.java sample (in %FMQ_DIR%/fmq/samples/pubsub/pubsub/Publisher.java) window. But, the applet does not receive any message, although they have same Topic names?

Although, both the samples are using same Topic name to send and receive information, the message formats are different. You may need to modify the publisher application to publish MapMessages with appropriate message properties for the MultiChat application to receive them.

FioranoMQ C/C++ Support

Native Support

Question 1: Can FioranoMQ be used through language APIs other than Java?

FioranoMQ provides C/C++ native runtime client libraries that can be used by the C/C++ applications to connect to the FMQ server. FioranoMQ also provides C# assemblies which can be used by .Net applications to exchange information with other JMS/native clients. In addition to this, FioranoMQ provides C/C++ wrappers and ActiveX components to publish and receive messages.

Question 2: How does the FioranoMQ C++ Runtime Library support JMS/Java?

C++ Runtime Library provides complete JMS support (PubSub and PTP). Support for C++ APIs is provided using both the native runtime library and the JNI (Java Native Interface) library. This implies that the clients have an option to use either the native library or the JNI library as per their requirements

Question 3: What are the requirements for native C++ implementation?

Native C++ library is a simple C++ binary (library/DLL) with no external dependencies.

Environment Settings and Initializations

Question 4: Are there any platform dependencies and/or requirements for the C++ FioranoMQ Runtime Library?

C++ Runtime Library has been tested and certified to work with Microsoft Visual C++ 6.0 and above on Windows NT platform. On Solaris, the C++ Runtime library has been tested and certified to work with Apogee's APCC 4.0 compiler and Sun Workshop 5.0 and above. C++ Runtime library requires an ANSI compliant compiler. Refer to the C++ Runtime Library Guide for complete details on the usage of the C++ Runtime Library.

Question 5: Does the C++ Runtime Library require a JVM or a java client to be distributed with a client application?

FioranoMQ provides both a native runtime library as well as a JNI library. In case you are using the native C++ library, you do not need a JVM, but in case the JNI library is used, you have to include the C++ runtime with a jre (1.1.8 or higher) on all client installations.

Question 6: What is the purpose of JAVA_HOME?

The JNI C++ runtime library needs a dll from the JRE installation. It loads the JVM.dll from %JAVA_HOME%/bin/server directory.

Question 7: How do I link FioranoMQ JNI C++ DLL with JDK1.2.2?

To link FioranoMQ JNI C++ library with jdk1.2.2, you have to ensure that jvm.dll is located in the system PATH. System PATH has a reference to locate Message-Adaptor.dll.

Question 8: Does FioranoMQ C++ runtime support jdk1.3?

FioranoMQ does provide support for jdk/jre1.3.

Question 9: What is the utility of MessageAdaptor.dll?

MessageAdaptor represents an instance of the installed C++ callback. Fiorano's Java runtime libraries asynchronously invoke the installed C++ callbacks using the MessageAdaptor.dll.

FioranoMQ and ActiveX Controls

Fiorano MQ and ActiveX Controls

Question 1: How does FioranoMQ provide for ActiveX controls?

FioranoMQ's ActiveX controls allow Win32 applications to leverage the power of JMS. FioranoMQ provides JMS ActiveX controls that can be embedded in any application developed in VisualBasic, VJ++ or VC++. These downloadable ActiveX controls enable the Win32 Applications to use both PTP and Publisher-Subscriber JMS architectures. These ActiveX controls provide for seamless exchange of JMS messages among Win32 Applications and between Win32 and Java Applications. FioranoMQ supports publishing and synchronous and asynchronous receipt of messages on topics and queues. FioranoMQ uses the bridge provided by Microsoft to register JavaBeans as ActiveX controls. The bridge used is bundled along with Microsoft SDK for Java 4.0. This bridge enables the Java Class or Bean to be registered as an ActiveX control or component to be used as any other ActiveX control from within Win32 applications.

Question 2: How do I use ActiveX controls in my Win32 Applications?

Download the ActiveX controls for Pub/Sub and PTP. These samples illustrate the use of JMS ActiveX controls within a Visual Basic application. Register the Java classes as ActiveX controls. You need to use javareg to register Java classes as ActiveX Controls. After the classes are registered as ActiveX controls, an application developed in Visual Basic can use the controls as follows:

On successful initialization, the controls return a positive value and if there is an error while initializing, a negative value is returned. Depending on the return parameter, the developer can determine the flow of action as follows:

```
Private Sub Form_Load()-+++++ Dim x As Integer Dim y As Integer x =
Me.FMQPublisher1.Initialize y = Me.FMQSubscriber1.Initialize If x = -1 Or y = -1 Then
```

`MsgBox "MQServer not up yet.Start the server and restart the app", vbCritical, Error Unload Me End If< End Sub The VB application can start publishing and subscribing like a normal Java application. Private Sub Command1_Click() FMQPublisher1.publish "Sending >>" & Text1.Text End Sub Private Sub Command2_Click() Text2.Text = "Received this >>" & FMQSubscriber1.receiveMessage End Sub The QueueReceiver and QueueSender controls can be used in a similar fashion. To bind onto the local MQ server you can run the sample without any further set up. In order to use a non local MQ server, say on a host called "asterix" you need to set the host and the port as follows: FMQPublisher1.setHost "asterix" FMQPublisher1.setPort 2001`

To publish on various queues and topics other than the default (primaryQueue/pri-maryTopic), you need to change the connectionfactory names and the topic/queue names as follows:

```
FMQPublisher1.setTopicConnectionFactoryName "primaryTCF" FMQPublisher1.setTopicName
"primaryTopic"
```

All the above mentioned scripts are with respect to a Visual Basic development environment.

Question 3: How do I set up a sample VB Application with ActiveX controls that communicate with each other?

You first need to register the Activex controls on your system.

To import the controls into the Visual Basic controls toolbox, perform the following steps:

1. Navigate to Projects in the menubar and select Components. This can also be accessed through a windows shortcut Ctrl+T.
2. This opens a dialog box that displays the various ActiveX controls and components registered with the windows system.
3. Select the check boxes corresponding to the following: fio-rano.fmq.beans.FMQPublisherBean and fiorano.fmq.beans.FMQSubscriber-Bean.
4. Now, these two controls is displayed on the controls tool box. To use them, drag and drop them on the sample application you have created.

If you have not yet created a sample application, follow the steps given below to develop it, shown in Figure 10-1, with the following assumptions:

1. The Publisher-subscriber model is to be used.
2. The default topic is used. This implies that the messages are published/sub-scribed on the "primaryTopic".

For example, the sample application has the following functionalities:

- Text in the text field is published when Publish is clicked
- Messages are added to the list of received messages when Subscribe is clicked.

A message is to be published first before subscribing to it, else Subscriber times-out after 5 seconds, and return a blank string.

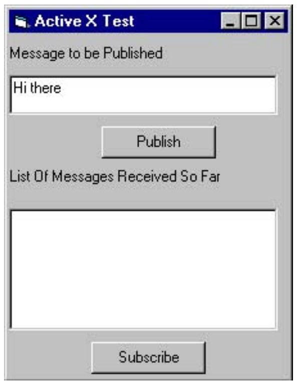


FIGURE 10-1 Sample VB Application using ActiveX Controls

The following code shows the basic setup needed to create a sample application:

```
Private Sub Form_Load() Dim x As Integer Dim y As Integer< x =
Me.FMQPublisher1.Initialize y = Me.FMQSubscriber1.Initialize If x = -1 Or y = -1 Then
MsgBox "MQServer not up yet.Start the server and restart the app", vbCritical, Error
Unload Me End If End Sub Private Sub publishBtn_Click() FMQPublisher1.publish "Sending
>>" & message.Text End Sub Private Sub subscribeBtn_Click() Dim str As String< str =
"Received this >>" & FMQSubscriber1.receiveMessage mesgList.AddItem str End Sub
```

Question 4: What are the limitations of ActiveX controls?

The controls that are currently provided ,do not allow for asynchronous data receipts. Each control works in the context of its sessions and connections. Future version of these controls provides for resource sharing among the controls. Currently, the ActiveX controls support only TextMessages.

JMS Beans/ActiveX Controls

Question 5: Can JMS be used inside a JavaBeans or C++ COM object?

The JMS design and FioranoMQ Server architecture allows JMS APIs to be used as Java Beans or as COM objects. ActiveX components for FioranoMQ are expected to be released soon.

Question 6: Does Fiorano provide ActiveX Controls to communicate with the Messaging Server?

FioranoMQ does, provide support for ActiveX controls.

Other ActiveX Issues

Question 7: What are the Platform requirements and software that needs to be installed?

Following are the platform requirements:

- Microsoft SDK for Java 4.0
- Microsoft's JRE (jview version 5.0 and higher)

- Any Win32 development environment, such as VB, VC or VJ++.

Question 8: How do I register the JavaBeans as ActiveX controls?

Batch files are provided for both installing and uninstalling the beans as controls. These scripts use the Microsoft SDK for Java 4.0 to install/uninstall the components (found in the output directory of the samples).

In case, you developed the control for use with FioranoMQ, then you need to edit these scripts to include your control name. Care should be taken here to specify a unique class for your class.

Question 9: What are the necessary setup issues to use JavaBeans ActiveX controls?

Prior to using these ActiveX components, make sure that javareg executable is available in the System PATH. javareg (of the MS SDK) is a binary executable usually present in the "bin" directory of MS SDK installation. The CLASSPATH should include all of the following:

FMQ Classes: jsse.jar, httpclient.zip, jetty.zip, sslava.jar, jnet.jar, jcert.jar, dxml.jar, xml4j.jar, jnet.jar, jms.zip, fmq-rtl.jar and fmq-common.jar.

All these files are located in the /fmq/lib directory of FioranoMQ installation.

In case the CLASSPATH settings are incorrect, javareg only register the ActiveX components in the Windows Registry but fails to make the tlb files, which are required for using the ActiveX components from any Win32 application.

Question 10: While trying to run the VB examples that use FioranoMQ ActiveX controls, why does it throw a "ClassNotFound" exception?

This exception is encountered when JNDI classes are not available in the CLASSPATH settings. Please set jndi.jar in your classpath and then try registering the ActiveX components. You could either use this jndi.jar or the rt.jar file that is bundled with JDK installation for the same purpose.

Active JMS

Question 11: How can FioranoMQ be made to work with ActiveJMS ?

Active JMS is a freely available open source project that provides non-proprietary ActiveX JMS client APIs. You can use these APIs to access JMS functionality of FioranoMQ.

Follow the steps given below to use Active JMS with FioranoMQ.

Download Active JMS from the following link:

http://active-jms.sourceforge.net/

Install Active JMS.

Copy the fmq-rtl.jar, fmq-common.jar and csp.zip in the fmq/lib/ext directory of the JRE that was configured with the activeJMS in its installation.

Update the active-jms.properties file (present in the "samples" directory of Active JMS installation) and make the following modifications.

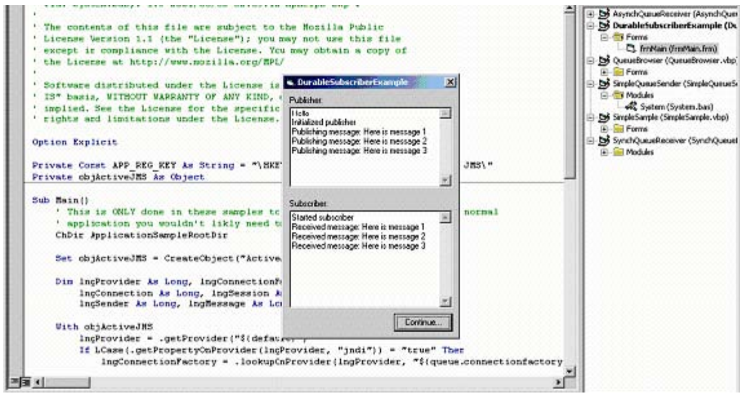
Set the resource provider as FioranoMQ. activejms.providers.default=FioranoMQ

Update the username-password or the url of the server in the following properties as per your configurations.

```
activejms.providers.FioranoMQ.username=anonymous
activejms.providers.FioranoMQ.password=anonymous
activejms.providers.FioranoMQ.properties.java.naming.pro-
vider.url=http://localhost:1856
```

Register the active-jms.tlb present in the bin directory in your visual basic project reference. (Select Projects > References > Active JMS Bean Control)

Finally, run the default visual basic samples that are shipped with the active-JMS installation. You should be able to send and receive JMS messages with FMQ server. The following figure displays the output of one such test.



```
'
' The contents of this file are subject to the Mozilla Public
' License Version 1.1 (the "License"); you may not use this file
' except in compliance with the License. You may obtain a copy of
' the License at http://www.mozilla.org/MPL/
'
' Software distributed under the License is
' IS" basis, WITHOUT WARRANTY OF ANY KIND,
' implied. See the License for the specific
' rights and limitations under the License.

Option Explicit

Private Const APP_REG_KEY As String = "\HKE                     JMS\"
Private objActiveJMS As Object

Sub Main()
    ' This is ONLY done in these samples to                      normal
    ' application you wouldn't likly need t
    ChDir ApplicationSampleRootDir

    Set objActiveJMS = CreateObject("Active

    Dim lngProvider As Long, lngConnectionF
        lngConnection As Long, lngSession A
        lngSender As Long, lngMessage As Lo

    With objActiveJMS
        lngProvider = .getProvider("$(defau
        If LCase(.getPropertyOnProvider(lngProvider, "jndi")) = "true" Then
            lngConnectionFactory = .lookupOnProvider(lngProvider, "$(queue.connectionfactory
```

DurableSubscriberExample

Publisher:

```
Hello
Initialized publisher
Publishing message: Here is message 1
Publishing message: Here is message 2
Publishing message: Here is message 3
```

Subscriber:

```
Started subscriber
Received message: Here is message 1
Received message: Here is message 2
Received message: Here is message 3
```

Continue...
AsynchQueueReceiver (AsynchQue
DurableSubscriberExample (Du
Forms
frmMain (frmMain.frm)
QueueBrowser (QueueBrowser.vbp)
Forms
SimpleQueueSender (SimpleQueueS
Modules
System (System.bas)
SimpleSample (SimpleSample.vbp)
Forms
SynchQueueReceiver (SynchQueue
Modules

FIGURE 10-2 JMS messages sent and received with FMQ server

For more details on Active JMS, please refer to the document "Getting Started.doc" shipped with the Active JMS installation.

Miscellaneous FAQ’s

Question 1: Do Queues provide for Client-side buffering?

PubSub subsystem can implement Client-side message buffering and thereby increase rates at which data is received by the client. Whereas, a Point-to-Point system essentially involves an intelligent polling mechanism to retrieve message from the server. There can be no buffering in the client for Queues, as a message can be delivered to a single queue receiver. It is recommended to use topics instead of queues for your messaging needs, since message transfer on topics is faster than that on Queues.

PubSub APIs can be used to simulate Queues. Please refer to $FMQ_DIR/fmq/ samples/PubSub/Queues.

Question 2: How to decide between "single topic complex selector Vs multiple topics simple selector" approach?

If the message types are previously segregated, it is recommended to send them on different topics. Message selection hits the performance drastically. The performance hit can be anywhere between 10 to 60 % depending upon message size. Also in case it is mandatory to use selection, use of simpler selection criteria is suggested (by way of segregating categories of messages to different topics.)

Question 3: What is the advantage of using server-side message filtering?

Server side message filtering reduces a lot of socket calls, since messages for a client are dropped by the server if they do not match the selector set by the client.

Question 4: Is it possible to change a consumer's selector dynamically or I have to close the connection, reset the selector and reconnect?

Consumer's message selectors cannot be modified dynamically. However it is not necessary to close the connection or reconnect for this purpose. It is sufficient if the consumer is closed (subscriber.close() API) and a new consumer is created with the new selector.

Question 5: Does FioranoMQ support multiple subscribers to the same Queue?

FioranoMQ allows multiple concurrent subscribers to receive data from the same Queue. FioranoMQ additionally ensures that in scenarios where multiple subscribers receive message from the Queue, none of the subscriptions “starve”. FioranoMQ ensures that there is no "starvation" if there are N-listeners on the same Queue. This is ensured by having a Job Queue within the MQServer, with utmost N-jobs (where N is the number of Subscriptions on the Queue).

Question 6: Does FioranoMQ support installation of the same callback on multiple Queues and Topics?

A Client Application can install the same callback for any number of subscriptions (on Topics or on Queues). JMS design prevents the callback from having a synchronization block.

Question 7: What happens if Client ID is not set for the connection on which the Durable Subscription is created?

JMS Provider sets a default clientID to every connection, if the clientID is not set by the JMS Application. Every durable subscription is identified by a unique combination of clientID and SubscriptionID. Hence, if the Durable Subscription does not set a clientID, every time the subscription is created, it would result in creation of a new Durable Subscription with a different clientID. Hence, you need to explicitly set the clientID for DurableSubscription.

Question 8: Does FioranoMQ provide for concurrent execution of MessageListeners on a Session?

A JMS Session is a single threaded context and does not provide for concurrent execution of MessageListeners on a session. Hence, FioranoMQ does not support concurrent execution of MessageListeners on a session.

Question 9: How do I set the configuration options in FioranoMQ?

FioranoMQ provides a comprehensive set of configuration options. The options can be set in Fiorano Admin Studio, a user- friendly GUI. For more information refer to **Chapter 3 Connection Management** in the *FioranoMQ 2007 Handbook*.

Question 10: Does FioranoMQ support logging?

FioranoMQ supports logging and dynamic tracing facilities. Logs can be redirected to files (as txt or html) or to the console. FioranoMQ allows users to plug in their own log mechanism.

For more information refer to **section 8.7 Logging and Tracing** in the *FioranoMQ 2007 Handbook.*

Question 11: How do I perform Administrative activities using the administration tool?

FioranoMQ provides for a graphical Administration Tool using to administer the FioranoMQ Server. This GUI tool allows the Server administrator to perform a large number of administrative tasks such as the following:

- Creation/deletion of administered objects (such as destinations, users, and connectionfactories)
- Managing ACLs on different destinations
- Managing clusters of MQ servers
- Monitoring the status of repeaters. The Admin Tool can be launched in one of the following manners:
- On Windows platforms, click Start > Programs > Fiorano > FioranoMQ 8.x > Fiorano Studio**.**

  Or

  Run the script, Studio.bat available in the Studio\bin folder of Fiorano installation. You can run the tool by typing the line Studio.bat at the command prompt.

• On Unix-based platforms, $FMQ_DIR/Studio/bin contains the script, Stu-dio.bat, that can be used to run the FioranoMQ Administration Tool. You can run the tool by executing the bourne shell script Studio.sh.

FioranoMQ Admin Studio uses the APIs exposed by the FioranoMQ Server. You can write your own tools that use these APIs. For more information refer to **Chapter 22 Administering FioranoMQ Server Using APIs** in the *FioranoMQ 2007 Handbook*.

Question 12: I received a system error during decompression, while trying to install FioranoMQ. What is the reason for this error?

One of the possible reasons could be lack of available free space in your "c:\" or the system directory in which you are trying to install FioranoMQ. (FioranoMQ installable needs around 120 MB of free space).

The steps required to resolve this issue is as follows:

1. Identify a drive on your machine with more than 120 MB of free space.
2. Create a directory d:\temp (if it does not already exist).
3. Click **Start > Settings** > **Control Panel** and double-click **System icon**.
4. Select **Advanced tab.**
5. Click **Environment Variables.**
6. In User Variables list, scroll down to select **TEMP** variable and set it's value to d:\temp. Scroll down to select **TMP** variable and set it's value to d:\temp.
7. Similarly, in System Variables list, scroll down to select **TEMP** variable and set it's value to d:\temp. Scroll down to select **TMP** variable and set it's value to d:\temp
8. Click **OK.**
9. Now try installing FioranoMQ on this machine.

Question 13: Is it possible to submit and consume messages (Simple Text) into/from a Fiorano queue, using a GCC (C++) application?

It is possible to compile C++RTL application on Solaris using GCC compiler. Following are the step-by-step instructions, along with the required files:

Files Required:

1. **libjms.so file** Library file containing the sources for C++ RTL, compiled with gcc on Solaris
2. **MessageAdapto**r Library file present in the lib directory of FMQ installation on Solaris
3. **cppc** Shell script for compiling your applications

Instructions:

Copy the libjms.so (and MessageAdaptor, if required) files to the %FMQ_DIR%/ fmq/lib directory and the cppc (replacement for jmscc.sh) to the %FMQ_DIR%\fmq\bin directory. Before you compile your applications using cppc, set the following shell variables:

1. FMQ_DIR (For example, FMQ_DIR=/export/home/ashish/FioranoMQ5.22)
2. JAVA_HOME (For Example, JAVA_HOME=/usr/java)

The cppc script requires that the libjava.so and libjvm.so files be present in the $JAVA_HOME/jre/lib/sparc directory. If they are not present in the specified directory, change the cppc script to match your path.

The MessageAdaptor, fmq-rtl.jar and fmq-common.jar are loaded from $FMQ_DIR/fmq/lib directory. The rt.jar is loaded from $JAVA_HOME/jre/lib. Make sure these files exist in their respective paths.

Question 14: Are persistent messages actually flushed to the disk before sending back an acknowledgement, or do you only acknowledge once in memory? A few other messaging vendors claim high performance numbers using this cache memory type system for persistent messages.

All persistent messages are written to the disk before a call to publish returns. FioranoMQ offers high-performance due to a file-based datastore that is optimized for messaging.

Question 15: For the API call public Enumeration elements (byte type) method of the MQAdminService class, what are the valid values for type?

The valid arguments (Constants along with their byte values) for elements() method of MQAdminService class are as follows:

```
NAMED_TOPIC = 0 NAMED_QUEUE = 1; NAMED_TOPIC_CONNECTION_FACTORY = 2;
NAMED_QUEUE_CONNECTION_FACTORY = 3; NAMED_ADMIN_CONNECTION_FACTORY = 4; NAMED_USER =
5; NAMED_GROUP = 6; NAMED_NEIGHBOUR = 7;
```

Question 16: Does FioranoMQ set RFH2 header on messages while connecting to IBM MQSeries, through FMQ bridge?

FioranoMQ does not set RFH2 header on the messages sent to MQSeries queue (through FMQBridge). FMQBridge uses the JMS wrapper APIs provided by IBM MQSeries. FMQBridge sends the messages to these wrapper APIs, which further set RFH2 header on the messages before sending them to MQSeries queue.

Question 17: In a clustered environment, where a dispatcher dispatches requests to one of the n servers, does each of the n servers maintain their own repository, or is there a central repository that all servers utilize?

In a clustered environment, each of the n servers maintains their own separate repository. The dispatcher provides load balancing at the connection level. Once the connection is made to one particular server, all send/receive and message caching is performed at that server. The only requirement in this case is that Connection-Factories (topic/queue) should be replicated on all the member servers as the lookup operation is performed on the dispatcher server.

A better solution for this replication is to use an LDAP server to store named objects such as TCFs, QCFs, Topics and Queues. Similarly, in this case, the LDAP server is only used for lookup while the MessageRepository for each member server is unique.

Question 18: I am making a call to getDurableSubscribersForTopic, and the API documentation says that it only returns an Enumeration of ClientIDs. I have tried String and TopicSubscriber. But with both I receive a run time class cast exception. What is the underlying class of the Enumeration? As I loop through, I have to know what Objects enumeration I have in order to process that correctly.

The call adminService.getDurableSubscribersForTopic(topicName) returns an enumeration of String array (String[]) and not String. This array consists of two strings, the ClientID of the connection on which the subscriber was made and the subscriberID of the concerned subscriber.

The code snippet for retrieving the SubscriberIDs registered on the concerned Topic is as follows:

```
// Get all the subscribers for this topic Enumeration jmsSubscribers =
adminService.getDurableSubscribersForTopic("primary- topic"); while
(jmsSubscribers.hasMoreElements()) { String[] sName = (String[])
jmsSubscribers.nextElement(); for(int i=0; i < sName.length;i++)

{ System.out.println("Client ID " + sName[i]); System.out.println("Subscriber ID " +
sName[i++]);

}
```

Question19: Can I use the character “:” while naming Queues and Topics?

FioranoMQ creates a directory with the specified Topic/Queue name. If the directory name (containing " : " character) is allowed by the Operating System (on which FMQ server is running), then FioranoMQ server allows the creation of such Topic names. For example, Windows does not allow creation of directory names with characters such as “\ / : * ? " < > ! ”, whereas Linux does. Thus, FioranoMQ running on Linux allows creation of Topic names with " : " character in it.

Question 20: Does FioranoMQ supports, silent install through install shield with WAIT?

FioranoMQ supports silent install through install shield with WAIT. Following are the instructions for achieving the same:

1. Extract the '.exe' FioranoMQ installer to a folder of your choice.
2. The folder disk1 in the extracted folders has the file setup.exe
3. Generate the file setup.iss using the following command:

   ```
   setup.exe -r -f1d:\setup.iss
   ```

4. Open the setup.iss file and make the following change to a parameter value:

   ```
   [SetupType-0] Result=301 szDir=
   ```

   The value of the szDir variable under [SetupType-0] would be empty (as shown in the code above) in the generated setup.iss file. Copy the value of the szDir variable under [SdAskDestPath-0] (shown in the following code) to this szDir variable:

   ```
   [SdAskDestPath-0] szDir=F:\Program Files\Fiorano\FioranoMQ Result=1
   ```

5. After making the above mentioned changes, execute the following command on the DOS-prompt for Silent install of FioranoMQ with WAIT:

```
start /w setup.exe /SMS -s -a -s d:\setup.iss
```

This command waits on the DOS-prompt until the Silent installation of FioranoMQ is completed, and exits as soon as the same is done.

Question 21: Is C++ RTL of FioranoMQ generic to all Operating Systems or proprietary to a specific OS?

FioranoMQ comes bundled with C++ runtime libraries for the Win32 and Solaris platforms. The libraries do not have any platform specific code and can be compiled on any OS.

Question 22: Who's SOAP API does FioranoMQ support?

FioranoMQ supports SOAP messages and does not impose any restrictions on any vendor specific SOAP APIs. To demonstrate the SOAP support, we have implemented the samples using Apache SOAP APIs. But SOAP APIs of other vendors also works with FioranoMQ.

Question 23: Can I setup the FioranoMQ server on a machine that is using dual IPs, where one IP address is for the FioranoMQ server inside a corporate firewall and a separate IP for machines coming through the same firewall?

The FioranoMQ server can be setup on a dual IP machine by setting up the server socket using the hostname of machine and using an LDAP server as the JNDI store for storing the ConnectionFactories. In this case, the use of LDAP is recommended as it provides a common URL for looking up the connection factories.

Each of the ConnectionFactory can have one IP address as the primaryURL and the other IP address as the backup or failover URL. Hence, if a client is trying to connect to a wrong IP address, the attempt fails, and it is automatically transferred to the correct IP address.

For the example mentioned in the above question, the use of dual IPs can be avoided by using the HTTP proxy supports provide by FioranoMQ. In that case the FioranoMQ server should be running on the HTTP protocol. The clients inside the firewall get connected directly to the internal IP address while the clients outside the firewall gets connected to the same internal IP address through a HTTP proxy. The proxy takes care of forwarding all requests to the internal IP address.

Question 24: Is there any functionality in FMQ that would allow a message not to be delivered before a specific time?

No, there is no functionality to allow a message not to be delivered before a specific time in the released versions of FMQ.

Question 25: I tested the timed Request/Reply sample when FioranoMQ servers were running on HA mode. All I changed in code was adding the "BackupConnectURLs" when creating InitialContext object. The two applications (requestor and replier) ran well until I shutdown the primary active server. The backup server becomes active but clients, that are requestor and/or replier didn't detect that. The replier didn't get requests after the failure of the active primary server. While this test works for topic pub/sub and queue sender/receiver, I was wondering why request/reply doesn't work. Is this a known bug? Is there something related to the temporary topic concept that is involved in this mechanism? Does HA support request/reply?

Request/reply mechanism over temporary destinations does not work in HA. On startup, FioranoMQ server deletes all the temporary destinations existing on the server. However, you can create the request/reply functionality over normal (non temporary) destinations in your applications, which works effectively in FMQ HA solution.

Question 26: In the following sample code, it appears that I do start the Topic Connection and QueueConnection before using them... could looking up the Topics be an issue?

```
env.put(Context.SECURITY_PRINCIPAL, config.getJMSuser());

env.put(Context.SECURITY_CREDENTIALS, config.getJMSpassword());
env.put(Context.PROVIDER_URL, config.getJMSURL());
env.put(Context.INITIAL_CONTEXT_FACTORY, "fiorano.jms.runtime.naming.FioranoIni-
tialContextFactory"); InitialContext ic = new InitialContext(env);
TopicConnectionFactory tcf = (TopicConnectionFactory) ic.lookup("primaryTCF");
QueueConnectionFactory qcf = (QueueConnectionFactory) ic.lookup("primaryQCF");
orderTopic = (Topic) ic.lookup("BORIS_ORDERS"); orderChangeRequestQueue = (Queue)
ic.lookup("BORIS_ORDER_REQUEST"); tc = tcf.createTopicConnection(); tc.start(); ts =
tc.createTopicSession(false, 1); qc = qcf.createQueueConnection(); qc.start(); qs =
qc.createQueueSession(false, Session.AUTO_ACKNOWLEDGE); TopicSubscriber tsub =
ts.createSubscriber(orderTopic); tsub.setMessageListener(new OrderReceiver());
```

Looking up the Topics won't be an issue until you want to receive the published messages on this connection.

Question 27: Does Fiorano create the second TCP connection when the Connection is started or stopped?

FioranoMQ creates the second TCP connection when connection is started e.g. tc.start() in your case. Need of starting a connection basically depends on the usage of connection. If you are using the connection for a subscriber you have to start the connection before receiving the messages. But in case of Publishers, one can avoid starting connection. So you can defer tc.start() call until you have to receive messages on this connection.

Question 28: So for a subscriber connection you never need to start it?

For a subscriber it's necessary to start the connection, in order to receive the messages. However, for a publisher, you need not to start the connection.

Question29: When running XA samples, I see the following exception stracktrace? What is the reason for this error?

```
Initializing .... Connection established......... Created the database table Inserted
the messages into the database Got the oracle XA Connection
Objectoracle.jdbc.xa.client.OracleXAConnection@1f5d386 Got the oracle XA Resource
Objectoracle.jdbc.xa.client.OracleXAResource@18088c0 Got the oracle XID
Objectoracle.jdbc.xa.OracleXid@fec107 Exception occured. Trying to close the resource
javax.transaction.xa.XAException at
oracle.jdbc.xa.OracleXAResource.recover(OracleXAResource.java:626) at
FioranoOracleXASample.createRDBMSObjects(FioranoOracleXASample.java:285) at
FioranoOracleXASample.createObjects(FioranoOracleXASample.java:308) at
FioranoOracleXASample.main(FioranoOracleXASample.java:153)
```

This error occurred, because the FioranoOracleXASample was unable to perform an XA recover operation. Therefore, the sample application did not seem to close its connection with Oracle properly, probably because either or both application or db server was not shut down properly and they crashed. Now that FioranoOracleXASample is attempting to re-establish the connection, Oracle needs to rollback any transactions that were in progress, but the Oracle user that FioranoOracleX-ASample is using to log into the database (scott, in this example) cannot perform the recovery. The solution is to give the Oracle user permission to perform the recovery, specifically to access the internal Oracle tables used to manage recovery. In SQL Plus as SYSOPER or SYSDBA, run this command:

```
grant select on DBA_PENDING_TRANSACTIONS to PUBLIC
```

If you don't want to grant this permission to all users, specify only the user listed in the error (in this example, scott). Then restart the database, and this time you should be able to successfully re-run FioranoOracleXASample.

Question 30: How can I change the user limit for the maximum number of open file descriptors for a user other than root?

The steps required, to change the user limit, for the maximum number of open file descriptors for a user other than root are as follows:

1. Login as root.
2. Edit the file /etc/security/limits.conf as root and make the following changes or add the following lines, respectively:

   ```
   devel soft nofile 10000

   devel hard nofile 20000
   ```

The "soft limit" in the first line, defines the number of file handles, or open files that the developer user have after login. If the developer user gets error messages about running out of file handles, then the developer user can increase the number of file handles like in this example up to 20000 (hard limit) by running the following command:

```
ulimit -n 20000
```

1. Setting "hard" and "soft" limits in the given example might not work properly when you login as developer from an SSH session. It might work if you log in as root and so to developer. If you have this problem, you should try to set UsePrivilegeSeparation in /etc/ssh/sshd_config to "no" and restart the SSH daemon by executing service sshd restart.
2. To ensure that the changes have taken place, login as developer from an SSH session and run the following command:

```
ulimit -n
```

You should see the output as 10000 which is the soft limit defined above.