# Oracle® Message Broker

Release Notes

Release 2.0.1.0 for UNIX and Windows NT

September 2000

**Part No. A85436-01**

## Contents



## Documentation

The Oracle Message Broker documentation is available in the directory $ORACLE_HOME/omb/2.0/doc (or %ORACLE_HOME%\omb\2.0\doc on Windows NT). Refer to this directory for the following Oracle Message Broker guides:

- *Oracle Message Broker Administration Guide Release 2.0.1.0*
- *Oracle Message Broker Installation Guide Release 2.0.1.0 for SPARC Solaris*
- *Oracle Message Broker Installation Guide Release 2.0.1.0 for Windows NT*

# JMS Features

## JMS Specification Support

Oracle Message Broker Release 2.0.1.0 supports JMS as described in the JMS 1.0.1 Specification. See "Unsupported JMS Functionality" for a list of features not supported in this release.

## Unsupported JMS Functionality

In addition to the limitations listed in this section, there are JMS limitations and restrictions based on the type of driver selected. Refer to the driver specific notes for a list of limitations by driver. For all drivers, the following JMS functionality is not supported in this release of Oracle Message Broker.

- `Message.acknowledge()` is not supported *[JMS 3.6]*
- The `TemporaryQueue` interface is not supported *[JMS 5.4]*
- The `TopicRequestor` interface is not supported *[JMS 6.14]*
- The `TemporaryTopic` interface is not supported *[JMS 6.6]*
- The `ServerSession` interface is not supported *[JMS 7.2.2]*
- The `ServerSessionPool` interface is not supported *[JMS 7.2.3]*
- The `ConnectionConsumer` interface is not supported *[JMS 7.2.4]*
- Transacted sessions cannot have a producer and a consumer with a message listener for the same destination.
- The name of the message broker entry stored in the LDAP administrative directory is fixed, as `cn=msg_broker`. This restriction allows for only one broker entry per OMB Instance in the directory.
- Oracle Message Broker does not support AUTO_ACKNOWLEDGE delivery mode for non-transacted sessions. For non-transacted sessions, the Oracle Message Broker only supports the following delivery mode:

  ```
  oracle.oas.mercury.MercurySession.IMMEDIATE_ACKNOWLEDGE
  ```

  Oracle Message Broker client-side code from previous releases of Oracle Message Broker that used non-transacted sessions with AUTO_ACKNOWLEDGE delivery mode must be updated to use IMMEDIATE_ACKNOWLEDGE delivery mode.

  See the *Oracle Message Broker Administration Guide* for more details on the supported acknowledgment mode for non-transacted sessions, and the handling of runtime exceptions.

## Limitations and Known Errors

Oracle Message Broker Release 2.0.1.0 supports JMS as described in the JMS 1.0.1 Specification. See "Unsupported JMS Functionality" for a list of features not supported in this release. In addition, this section covers the following limitations and known errors:



### Oracle Message Broker Release 2.0.1.0 Versions

There are differences between Oracle Message Broker Release 2.0.1.0, as supplied on the Oracle 8*i* Release 8.1.7 CD and the version supplied on the Oracle Message Broker Release 2.0.1.0 CD. In particular, the Oracle Adapter Framework and the Dynamic Monitoring Service (DMS) GUI are available on the Oracle Message Broker Release 2.0.1.0 CD.

### Deinstallation Restriction

When you need to deinstall the Oracle Message Broker, always use the Oracle Universal Installer. The Oracle Universal Installer places an inventory file on your system to keep track of installed Oracle products. If you remove files manually, the inventory file is not updated and attempts to reinstall Oracle Message Broker will not succeed.

### Administration Restrictions and Notes

1. For administration, there are several Release 1.0 attributes that either do not exist in Release 2.0 or that are handled differently. Older administration scripts that use any of these attributes may report errors.

   The following list contains further details on this change.

   a. `aq_username`: The attribute `aq_username` for entries of type `aq_server` is now mandatory. Scripts that create an AQ Server now need to include a value for the `aq_username` attribute.

   b. `remote_dir_dn`: The attribute `remote_dir_dn` of the entry type `propagation_job` has been renamed to `remote_dn`. This attribute can now be either the DN of a remote directory entry or the DN of a RemoteHTTP entry. Existing directory entries need not be changed since the administration code will translate `remote_dir_dn` values to appropriate values for `remote_dn`.

      `AdminUtil` scripts that use `remote_dir_dn` will fail with an unknown attribute error message.

   c. `is_queriable`: The attribute `is_queriable` associated with a queue entry or a topic entry has been deprecated. Existing directory entries need not be changed, since the administration code translates values from `is_queriable` to appropriate values for the new attribute, `aq_adt`. The `is_queriable` attribute value `true` translates to the value `queriable` for the `aq_adt` attribute. The `is_queriable` attribute value `false` translates to the value `raw` for the `aq_adt` attribute. Any update to an existing directory entry results in the replacement of `is_queriable` by `aq_adt`.

      `AdminUtil` scripts that use `is_queriable` will fail with an unknown attribute error message.

2. Do not use samples / admin / security / ldap / oid / rootaci.ldif with OID. If this script is run, then after the Directory Server is shutdown, it cannot be restarted.

3. To modify a user or a group entry you need to delete the entry and then recreate it, when working with `AdminUtil` or `ombadmin`. User entries are created in the Users container below the OMB entry. Group entries are created in the Groups container below the OMB entry. Attributes for Users and Groups can only be changed when the entry is created.

4. Using the propagation wizard with ombadmin, users can select a driver to associate with a propagation log queue. The wizard allows selections for the AQLite Driver, although propagation is not supported for AQLite. Users should only select either AQ logging queues or MQSeries logging queues when working with the ombadmin propagation wizard.

### System Specific Limitations (Windows NT)

1. On Windows NT systems, when you run the Oracle Message Broker utilities, use double quotes (" ") to enclose all arguments that contain a DN or an RDN.
2. When running Oracle Message Broker on Windows NT systems, for connections to Oracle8*i*, use dedicated server connections. Do not use multi-threaded server connections (MTS). The Oracle8*i* connections should only use dedicated servers. For more information on MTS, see the *Oracle8i Administrator's Guide*.
3. The CLASSPATH variable on Windows NT systems is limited to 512 characters. If this limit is exceeded, and the CLASSPATH is too long, class not found exceptions may thrown by any of the Oracle Message Broker components.
4. For Windows NT systems, the Oracle Message Broker supports commands that allow Oracle Message Broker to run as an NT service. The following commands are only supported for Release 2.0.1.0 when a patch is applied: `Jlaunch`, `Register`, `Unregister`. A software patch adds support for these commands. Contact your Oracle Support Services representative or support.us.oracle.com to obtain further information (associated bug 1319440).

### Resource Exceptions

Resource exceptions can be thrown on any method that increases the state of the Oracle Message Broker. The Oracle Message Broker state is increased by creating producers or consumers. The state is also increased by sending messages using the Volatile Driver. An exception is also thrown when the Oracle Message Broker is low on memory. The Oracle Message Broker attempts to reclaim memory by calling `System.gc()` prior to throwing resource exceptions.

The resource exceptions that are thrown are of the type:

```
javax.jms.ResourceAllocationException
```

To avoid resource exceptions, start the Oracle Message Broker with a larger JVM heap size.

### AQ Driver Restrictions

1. Intermittent failures are possible when the JDBC based AQ driver is used with AQ queues that have been created using the `ombaq_map_msg` object type. The failure generates an ORA-600 exception. The failure does not occur for JMS topics (AQ multi-consumer queues). The exception occurs on receive when the queue is empty. The bug number for this bug is: 1288749.

2. When the JDBC-based AQ Driver is used on a multi-CPU NT system, the Oracle Message Broker may block while closing a JMS session that uses the AQ Driver. This occurs infrequently (about once per day). When this occurs, the Oracle Message Broker process should be stopped and restarted. This is less likely to occur when Oracle Message Broker is deployed in local mode.The bug number for this bug is: 1318694.

3. AQ Driver Administration Limitation – Removing an AQ subscriber, topic, or queue can take a long time. The request, using the administration utilities, is performed without any timeout set. Attempting one of these requests when there are uncommitted sends may cause the operation to block for an arbitrarily long time.

4. AQ Driver Name Restrictions – When Oracle Message Broker administrators create any of the following objects for an AQ Driver: a topic, a queue, a propagation job, or a durable subscriber, the object's name must follow the object name guidelines as specified in the *Oracle8i SQL Reference Guide*, with regard to reserved characters. In general, this means the name must only contain alphanumeric characters, and must not start with a numeric character.

5. AQ Driver Queue Message Store Limitations

| Limitation | Value |
|---|---|
| Maximum length of a queue name | 30 bytes |
| Maximum length of a subscriber name | 30 bytes |
| Maximum length of a correlation ID | 128 characters |

6. AQ Driver JMS Limitations (Queriable Messages Only)

| Limitation | Value |
|---|---|
| Maximum length of a destination name (queue or topic) | 64 bytes |
| Maximum length of a reply_to field name | 64 bytes |
| Maximum length of JMS Header type field | 32 bytes |

### AQ Lite Driver Restrictions

1. The AQ Lite Driver is only supported for the Windows NT platform.

2. The AQ Lite Driver is only supported with JDK 1.2.x.

3. The AQ Lite Driver only supports PERSISTENT JMS delivery mode *[JMS 4.7]*.

4. The AQ Lite Driver supports only the JMS point-to-point (PTP) model. The AQ Lite Driver does not support the JMS publish / subscribe messaging model.
5. The AQ Lite Driver does not support JMS Queue Browsers.
6. The AQ Lite Driver does not support message expiration.
7. The AQ Lite Driver does not support deferred delivery of messages.
8. The AQ Lite Driver does not support Queriable queues or queriable topics.
9. The AQ Lite Driver creates AQ Lite queues in the 'stand alone' mode since it *does not* make use of the propagation functionality provided by AQ Lite. This simplifies AQ Lite queue and topic administration, at the expense of disallowing the propagation of messages from the Oracle Lite Database using the AQ Lite propagation functionality independent of Oracle Message Broker. Future releases will support AQ Lite queues that are created in the 'master-slave' mode to utilize propagation features built into AQ Lite.
10. The following list of known limitations in the AQ Lite component of Oracle 8*i* Lite Release 4.0.0.2 affect the functioning of the AQ Lite Driver.
    a. Messages may arrive out of order when more than one enqueuer or dequeuer operate against a given AQ Lite queue even if the enqueues and dequeues are performed serially. Messages may also arrive out of order when a single enqueuer enqueues messages and another dequeuer dequeues messages concurrently on the same AQ Lite queue.
    b. Messages may not appear in priority order when dequeued from the same queue even when the enqueues and dequeues are performed by the same client.
    c. Concurrent enqueues and dequeues to the same queue using two or more clients can cause run time exceptions within AQ Lite which results in failed enqueues or dequeues. In rare cases the queues may become unusable and may need to be dropped in order to stop further occurrence of runtime exceptions in AQ Lite.
    d. AQL-1043 (Message Payload Error) errors may occur under stress (when using large message sizes and for large volumes of enqueues and dequeues).

### MQSeries Driver Restrictions

MQSeries Limitations

| Limitation | Value |
|---|---|
| Maximum length of a Correlation ID | 24 bytes |
| Maximum length of a Message ID | 24 bytes |
| Maximum header size | In JMS mode, the entire message is placed in the body. The restriction is based on the size of the serialized message. |
| Maximum header size | In native mode, JMS header information is used for some of the MQSeries header. |
| Maximum length of queue manager name | 48 bytes |

### TIBCO Driver Restrictions

TIBCO Driver Topic Name Limitations – When an Oracle Message Broker administrator creates a topic for the TIBCO Driver, the common name (cn) for the topic must comply with the following rules for TIBCO subjects (see the *Rendezvous Administration Guide* for details). Alternatively, if the topic attribute provider_queue_name is supplied for the topic name, the value provided for the provider_queue_name must comply with these rules:

1. Each subject name is a string of characters is divided into elements by the dot (.) character.
2. It is illegal to incorporate the dot character into an element by using an escape sequence.
3. Subject names are limited to a total length of 500 characters (including delimiters), divided into at most 100 elements.
4. The maximum element length is 255 characters. Typical subject names are shorter and use fewer elements.
5. For maximum speed and throughput rates, use short subject names.
6. Subject names beginning with underscore are reserved.
7. It is illegal for application programs to send to subjects with underscore as the first character of the first element, except _INBOX and _LOCAL.
8. Dot, ".", separates elements within a subject name.
9. Greater-than, ">" is a wildcard character. This matches any number of trailing elements.

10. Asterisk, "*" is a wildcard character. This matches one element.
11. Do not use tabs, spaces, or any unprintable character in a subject names.

### Propagation Restrictions

1. There is a restriction on IIOP propagation. If the LDAP Directory associated with the receiving broker for a propagation job is SSL enabled, it can only use the SSL level 1 (this level supports data encryption only). SSL level 0 is also supported. Using SSL level 0, SSL is disabled.
2. Configuring the Oracle Message Broker to run with multiple built-in HTTP listeners using different SSL levels causes propagation job failures.
3. When a built-in HTTP listener fails to start due to improper configuration, the propagation manager writes error messages repeatedly in the Oracle Message Broker omblog file. This problem can cause the omblog file to grow very fast. There is no workaround for this problem. Users should check the log file when the Oracle Message Broker starts up to make sure the built-in HTTP listener starts correctly. For example, this problem occurs when using an invalid HTTP port number, or when using an invalid wallet location or password when SSL security is specified.
4. When the Oracle Message Broker starts with a built-in HTTP listener, the listener may fail to start with the following error message printed to the omblog file:

   ```
   HTTP Listener networking error
   ```

   This message indicates that an invalid hostname, port number, or an invalid wallet location or password was specified in the HTTP listener entry (prop_http).
5. When a sending broker attempts to send a message to a receiving broker, and the connection fails for any reason, including: the receiving broker is not running, the communication line is down, invalid specification of hostname or port number, or when an invalid wallet location or password is specified, the propagation manager writes the following error to the omblog file:

   ```
   Failed to create a socket to host hostname:port#, error code = 20
   ```

6. Propagation is not supported for destinations that use the Oracle Message Broker AQLite Driver. That is, using a propagation source, or a propagation target that is an AQLite queue is not supported.

### Java String Limitation

The JMS specification requires strings to be encoded in UTF-8 format when:

1. A String is encoded into a BytesMessage (BytesMessage.writeUTF)
2. ObjectMessage.setObject(*arg*) is called and *arg* has the type java.lang.String.
3. When ObjectMessage.setObject is called and the object contains an element that is a string.

The only other way to put a string into a BytesMessage is to call BytesMessage.writeChar for each character in the string. UTF-8 limits Strings to 65535 characters. The implications for working with Strings longer than 65535 characters follow.

TextMessage.setText accepts strings that have more than 65535 characters. These messages will not be handled correctly when they are converted to UTF-8 in either of the following cases:

1. You cannot call BytesMessage.writeUTF if the string has more than 65535 characters.
2. You cannot call ObjectMessage.setObject with a string that has more than 65535 characters.
3. You cannot call ObjectMessage.setObject with an object that contains a string with more than 65535 characters.

### Javadoc Error

In the Oracle Message Broker Javadoc, the setDeliveryMode() description states that the producer's default delivery mode is set to PERSISTENT. This is only true for the MQSeries Driver, the AQ Driver, and the AQ Lite Driver. When using Oracle Message Broker with the Volatile Driver, the Multicast Driver, or the TIBCO Driver, the default mode is NON_PERSISTENT.

### Message Size Restrictions

The maximum size for messages sent using the Oracle Message Broker is limited by several factors.

For clients in local mode, the maximum message size is limited by the memory and resources available in the JVM running the client and the Oracle Message Broker.

For clients in Remote Mode, the maximum message size is limited by the memory and resources available in the JVM running the client and the Oracle Message Broker.

### C++ API Limitations

- BytesMessage has the following limitation: Only the reset(), readByte(), readUnsignedByte(), writeByte(), readBytes() and writeBytes() methods are implemented.
- The C++ API for the Oracle Message Broker has been tested with Sun CC 5.0 on Solaris 2.6 and with MS Visual C++ 5.0 on NT 4.0.

### Client Identifier Restriction

If a client does not explicitly set a connection's client identifier, the value for the client identifier is null, and the method call `Connection.getClientID()` returns null.

Due to upcoming changes required for JMS 1.0.2, it is recommended that if a client explicitly sets the client identifier using `setClientID()`, that the set be performed immediately after creating the connection and before any other action on the connection is taken.

### Security Restrictions

1. IIOP over SSL is not supported.
2. If an administrator specifies a mismatch between the security SSL settings, where the client side is SSL enabled and the server side is not SSL enabled, the mismatch may cause SSL to hang (where client and server are used in terms of a distributed system). This can occur in connections from a JMS client or the Oracle Message Broker to the LDAPS Directory, or using Oracle Message Broker propagation. The hang occurs when the SSL level is set to 1, 2, or 3, on the client side, and the SSL level for the server side is set to 0 (non-SSL).

## JDK Limitations

The following JDK limitations may affect the Oracle Message Broker or Oracle Message Broker clients.

### Version Limitations

The Oracle Message Broker and the JMS client-side runtime throws an exception if either are started with an unsupported JVM version. When the system property oracle.oas.mercury.anyjvm is set, the Oracle Message Broker and the client-side runtime do not check the JVM version.

An unsupported JVM version is a JVM that does not set the system property "java.version", or one that sets the system property to a string that starts with: '1.0', '1.1.0', '1.1.1', '1.1.2', '1.1.3', '1.1.4' or '1.1.5'.

### Solaris Limitations

1. The Oracle Message Broker should not be used with reference implementations of the JDK.
2. The Oracle Message Broker must be used with a JDK running with native threads. The production implementations of JDK use native threads by default.

### Windows NT Limitations

1. Access exceptions may occur in javai.dll when using `Thread.start` when multiple threads are started in a loop. The workaround is to call `Thread.sleep` in between calls to `Thread.start`.

   Bugs 213274 and 4213270 describe this problem in Javasoft bug parade.

   For example, the following code can cause exceptions on dual-processor Windows NT systems.

   ```
   class MyThread extends Thread { ... }
   for (int i = 0; i < 10; i++)
      {   MyThread mt = new MyThread();   mt.start(); }
   ```

   The workaround is:

   ```
   class MyThread extends Thread { ... }
   for (int i = 0; i < 10; i++)
      {   MyThread mt = new MyThread();   mt.start();
   Thread.currentThread().sleep(500); }
   ```

2. Intermittent access exceptions may occur in JMS clients that cause the client to terminate. Failures occur in the function jio_snprintf(), part of javai.dll. There are no known workarounds. This exception only occurs when the client calls shutdown(). For information on this problem, refer to the following web site,

   `http://developer.java.sun.com/developer/bugParade/bugs/4138925.html`

3. To use Oracle Message Broker with JDK 1.1.8 on Windows NT systems, the following modification must be performed after running the installer:

   a. Use the jar command to unjar %OMB_HOME%\classes\ldap.jar:

      ```
      jar xf ldap.jar
      ```

   b. Edit the file com/sun/jndi/ldap/jndiprovider.properties to remove all occurrences of com.sun.jndi.ldap.obj.RemoteToAttrs and com.sun.jndi.ldap.obj.MarshalledToObject. Change the lines that specify the java.naming.factory.object and the java.naming.factory.state properties to comments by adding a "#" at the beginning of the line.

   c. Use the jar command to repackage ldap.jar:

      ```
      jar cf ldap.jar com
      ```

   The reason for this modification is the aurora_client.jar file is placed in the classpath. The aurora_client.jar contains the classes in step 2 above, that use JDK 1.2 specific features.

### Generic Bugs

The Oracle Message Broker may encounter the following problems that may prevent the runtime from increasing the size of the heap.

The garbage collector may prevent the JVM runtime from increasing the size of its heap. The workaround for this problem is to start the JVM with a maximum and initial heap size that are the same. Using these options, the garbage collector is unable to resize the heap.

For a given heap size, the JDK allows a fixed number of objects to be concurrently allocated. If the allocated objects are small, the JDK throws out of memory exceptions when there is a significant (> 50%) amount of heap space available. The garbage collector also has a problem when most of the heap is allocated. This problem may prevent the runtime from increasing the size of the heap.

The workaround is to be aware of the maximum number of objects for a specific heap size. The maximum number of objects per MB of heap space is approximately 25,000.

Javasoft bug ids for these bugs include: 4209215, 4055198.

### JDK 1.1.x and JDK 1.2 Memory Requirements

JDK 1.2 uses much more memory than JDK 1.1.x. For example, an Oracle Message Broker started with a 64MB heap, using the volatile driver, and

JDK 1.1.8 uses about 85MB. The Oracle Message Broker with the same configuration using JDK 1.2 uses about 120MB at startup.

## Release 2.0.1.0 Changes

1. This release adds support for the Oracle Database Server Release 8.1.7, and Oracle Internet Directory Release 2.1.1.