Sun Java System Message Queue 4.1 发行说明

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95054
U.S.A.

文件号码 820–3193
2007 年 9 月

版权所有 2007 Sun Microsystems, Inc.　4150 Network Circle, Santa Clara, CA 95054 U.S.A.　保留所有权利。

对于本文档中介绍的产品，Sun Microsystems, Inc. 对其所涉及的技术拥有相关的知识产权。需特别指出的是（但不局限于此），这些知识产权可能包含在美国和其他国家/地区申请的一项或多项其他专利或待批专利。

美国政府权利 – 商业用途。政府用户应遵循 Sun Microsystems, Inc. 的标准许可协议，以及 FAR（Federal Acquisition Regulations，即“联邦政府采购法规”）的适用条款及其补充条款。

本发行版可能包含由第三方开发的内容。

本产品的某些部分可能是从 Berkeley BSD 系统衍生出来的，并获得了加利福尼亚大学的许可。UNIX 是 X/Open Company, Ltd. 在美国和其他国家/地区独家许可的注册商标。

Sun、Sun Microsystems、Sun 徽标、Solaris 徽标、Java 咖啡杯徽标、docs.sun.com、 Java 和 Solaris 是 Sun Microsystems, Inc. 在美国和其他国家/地区的商标或注册商标。所有的 SPARC 商标的使用均已获得许可，它们是 SPARC International, Inc. 在美国和其他国家/地区的商标或注册商标。标有 SPARC 商标的产品均基于由 Sun Microsystems, Inc. 开发的体系结构。

OPEN LOOK 和 Sun™ 图形用户界面是 Sun Microsystems, Inc. 为其用户和许可证持有者开发的。Sun 感谢 Xerox 在研究和开发可视或图形用户界面的概念方面为计算机行业所做的开拓性贡献。Sun 已从 Xerox 获得了对 Xerox 图形用户界面的非独占性许可证，该许可证还适用于实现 OPEN LOOK GUI 和在其他方面遵守 Sun 书面许可协议的 Sun 许可证持有者。

本服务手册所介绍的产品以及所包含的信息受美国出口控制法制约，并应遵守其他国家/地区的进出口法律。严禁将本产品直接或间接地用于核设施、导弹、生化武器或海上核设施，也不能直接或间接地出口给核设施、导弹、生化武器或海上核设施的最终用户。严禁出口或转口到美国禁运的国家/地区以及美国禁止出口清单中所包含的实体，包括但不限于被禁止的个人以及特别指定的国家/地区的公民。

本文档按“原样”提供，对于所有明示或默示的条件、陈述和担保，包括对适销性、适用性或非侵权性的默示保证，均不承担任何责任，除非此免责声明的适用范围在法律上无效。

目录

◆ ◆ ◆ 第 1 章

Sun Java System Message Queue 4.1 发行说明

版本 4.1

文件号码 820-3193

本发行说明包含了发行 Sun Java™ System Message Queue 4.1 时可用的重要信息。本说明主要介绍新增功能和增强功能、已知问题和限制以及其他信息。在使用 Message Queue 之前，请先阅读本文档。本发行说明还包含有关 Message Queue 4.0 发行版的信息，请参见第 15 页中的 “关于 Message Queue 4.0”以了解该发行版中引入的功能的相关信息。

本发行说明的最新版本可以在 Sun Java System Message Queue 文档 Web 站点找到。请在安装和设置软件前仔细查阅此 Web 站点，完成安装和设置后也要定期查看最新的发行说明和产品文档。

本发行说明包含以下部分：

本文档引用第三方 URL，并提供其他相关信息。

Sun 对本文档中提到的第三方 Web 站点的可用性不承担任何责任。对于此类站点或资源中的（或通过它们获得的）任何内容、广告、产品或其他材料，Sun 并不表示认可，

也不承担任何责任。对于因使用或依靠此类站点或资源中的（或通过它们获得的）任何内容、产品或服务而造成的或连带产生的实际或名义损坏或损失，Sun 概不负责，也不承担任何责任。

发行说明修订历史记录

下表列出了 Message Queue 产品的所有 4.x 发行版的发行日期，并描述了与每个发行版关联的主要更改。

表 1–1 修订历史记录

日期	更改描述
2006 年 5 月	针对 Message Queue 4.0 版的本文档的初始发行版。
2007 年 1 月	针对 Message Queue 4.1 Beta 版的本文档的初始发行版。增加了 JAAS 支持说明。
2007 年 4 月	针对 Message Queue 4.1 Beta 版的本文档的第二个发行版。增加了高可用性功能。
2007 年 9 月	向客户分发的本文档的第三个发行版。增加了 Java Enterprise System Monitoring Framework、固定 C 端口、错误修复以及其他功能的支持说明。

关于 Message Queue 4.1

Sun Java System Message Queue 是一种功能全面的消息服务，提供符合 Java Messaging Specification (JMS) 1.1 的可靠、异步的消息传送功能。此外，Message Queue 还提供 JMS 规范之外的许多功能，以满足大型企业部署的需要。

Message Queue 4.1 版增加了高可用性、Java 验证和授权服务 (Java Authentication and Authorization Service, JAAS)、使用固定 C 端口以及 Java Enterprise System Monitoring Framework 支持。它还增加了少量的增强功能和错误修复功能。本部分包含以下信息。

- 第 7 页中的 “4.1 发行版中的新增功能”
- 第 15 页中的 “硬件和软件要求”

有关 Message Queue 4.0 中引入的功能的信息，请参见第 15 页中的 “关于 Message Queue 4.0” 。

4.1 发行版中的新增功能

Message Queue 4.1 引入了高可用性（数据和服务可用性）代理群集、JAAS 支持以及各种其他次要功能。本部分介绍了这些功能，并提供了进一步的使用参考信息。

- 第 7 页中的 “高可用性”
- 第 8 页中的 “JAAS 支持”
- 第 13 页中的 “持久性存储库格式更改”
- 第 13 页中的 “代理配置”
- 第 13 页中的 “JES Monitoring Framework 支持”
- 第 14 页中的 “事务管理”
- 第 14 页中的 “用于 C 客户端连接的固定端口”

高可用性

Message Queue 4.1 引入了高可用性群集，这些群集提供了数据可用性以及服务可用性。如果客户端与高可用性代理之间的连接断开，则会自动将其重新连接到群集中的另一个代理。提供新连接的代理将接管故障代理的持久性数据和状态，并继续为故障代理的客户端提供不间断的服务。您可以通过安全连接来运行高可用性代理。

高可用性代理要求使用高可用数据库 (Highly Available Database, HADB)。如果没有此类数据库，或者数据可用性对您并不重要，您可以继续使用传统群集，这些群集提供自动重新连接和服务可用性。

配置高可用性代理的过程非常简单：您可以为群集中的每个代理指定以下类型的代理属性。

- **群集成员属性**，它们指定代理是高可用性群集的一部分，并指定群集 ID 以及代理 ID。
- **高可用数据库** *(HADB)* **属性**，它们为数据库指定了持久性消息模型 (JDBC)、HADB 供应商名称以及特定于供应商的配置属性。
- **故障检测和接管属性**，它们指定了应如何检测和处理代理故障。

要使用此功能，您必须执行以下操作：

1. 安装高可用数据库。
2. 安装 JDBC 驱动程序的 .jar 文件。
3. 为高可用的持久性存储库创建数据库模式。
4. 为群集中的每个代理设置与高可用性相关的属性。
5. 启动群集中的每个代理。

有关高可用性以及将其与传统群集进行对比的概念性讨论，请参见《Sun Java System Message Queue 4.1 Technical Overview》中的第 4 章 “Broker Clusters”。有关高可用性的过程和参考信息，请参见《Sun Java System Message Queue 4.1 Administration Guide》中的第 8 章 “Broker Clusters”和《Sun Java System Message Queue 4.1 Administration Guide》中的“Cluster Configuration Properties”。

如果将 HADB 数据库与 Message Queue 4.0 版一起使用，并且要使用高可用性群集，则可以使用 `dbmgr` 实用程序升级到 HADB 共享存储库。有关详细信息，请参见第 39 页中的 “代理群集”。

JAAS 支持

除了基于文件的内置验证机制以及基于 LDAP 的内置验证机制外，Message Queue 还支持 Java 验证和授权服务 (Java Authentication and Authorization Service, JAAS)，它允许将各种服务插入到代理中以验证 Message Queue 客户端。本部分介绍了代理为符合 JAAS 的验证服务提供的信息，并且说明了如何配置代理以使用此类服务。

而介绍 JAAS API 则超出了本文档的范围。如果需要了解更多信息，请参考以下资源。

- 有关 JAAS API 的完整信息，请参见《Java Authentication and Authorization Service (JAAS) Reference Guide》。

  `http://java.sun.com/j2se/1.5.0/docs/guide/security/jaas/JAASRefGuide.html`

- 有关编写 LoginModule 的信息，请参见《Java Authentication and Authorization Service (JAAS) LoginModule Developer's Guide》。

  `http://java.sun.com/j2se/1.5.0/docs/guide/security/jaas/JAASLMDevGuide.html`

JAAS API 是 J2SE 中的核心 API，因此，它是 Message Queue 运行环境中的一个不可或缺的组成部分。JAAS 在应用程序和验证机制之间定义了一个抽象层，它无需更改应用程序代码即可插入所需的机制。对于 Message Queue 服务，抽象层位于代理（应用程序）和验证提供者之间。通过设置几个代理属性，您可以插入任何符合 JAAS 的验证服务并升级此服务，而不会产生中断或需要更改代理代码。

如果使用的是基于 JAAS 的验证，则可以使用 JMX 客户端来管理代理，但必须先手动设置 JAAS 支持（通过设置与 JAAS 相关的代理属性），然后才能启动代理。无法使用 JMX API 更改这些属性。

JAAS 元素

图 1–1 显示了 JAAS 的基本元素：JAAS 客户端、符合 JAAS 的验证服务以及 JAAS 配置文件。

- JAAS 客户端是一个应用程序，它需要使用符合 JAAS 的验证服务来执行验证。它使用 LoginModule 与此服务进行通信并负责提供回调处理程序，LoginModule 可以调用此处理程序以获取用户名、密码以及其他相关信息。
- 符合 JAAS 的验证服务包含一个或多个 LoginModule 以及执行所需验证的逻辑。LoginModule 可以包含验证逻辑，也可以使用专用协议或 API 与提供该逻辑的模块进行通信。
- JAAS 配置文件是一个文本文件，JAAS 客户端使用该文件来查找与 JAAS 服务进行通信所需的LoginModule。

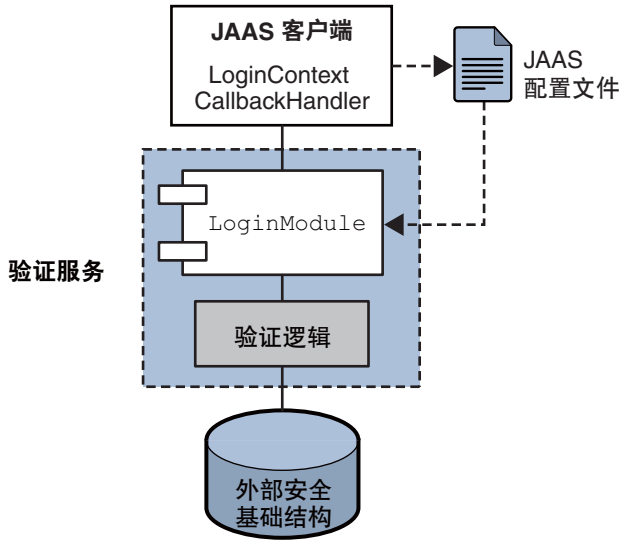

图 1–1 JAAS 元素

下一部分说明了 Message Queue 服务如何使用这些元素来提供符合 JAAS 的验证。

JAAS 和 Message Queue

下图说明了 Message Queue 代理如何使用 JAAS。它显示了上图所示的 JAAS 模型的更复杂实现。

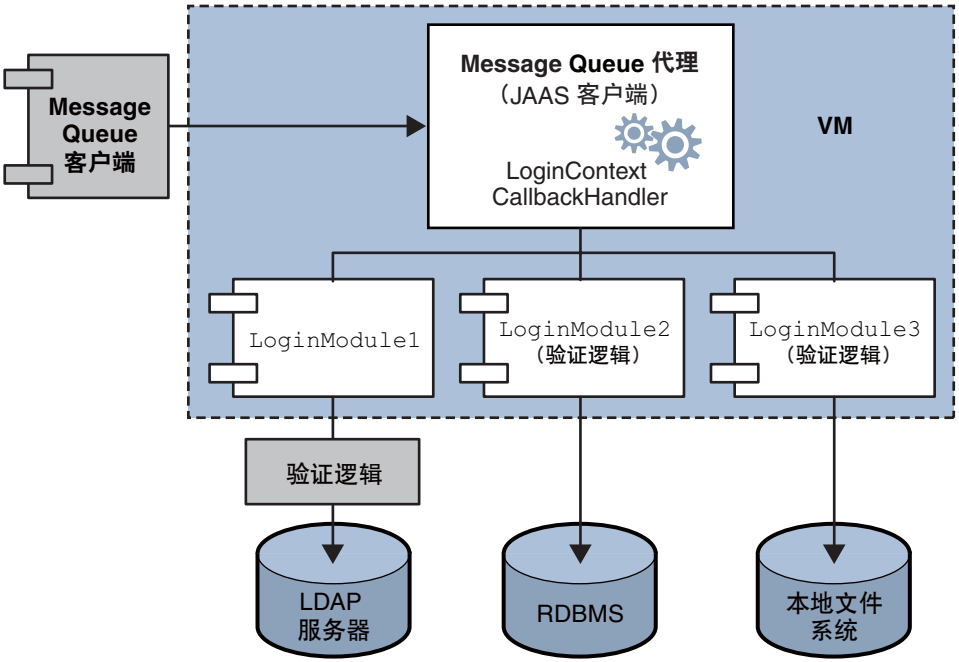

图 1–2 Message Queue 如何使用 JAAS

与更简单的示例所显示的一样，验证服务层与代理是分开的。验证服务包含一个或多个登录模块 (LoginModule) 以及附加验证模块（如果需要）。登录模块在与代理相同的 Java 虚拟机中运行。对于登录模块来说，Message Queue 代理表示为 `LogInContext`，并通过 `CallBackHandler`（作为代理运行时代码的一部分）与登录模块进行通信。

验证服务还提供一个 JAAS 配置文件，其中包含登录模块的入口。该配置文件指定了这些模块的使用顺序以及一些使用条件。在代理启动时，JAAS 通过 Java 系统属性 `java.security.auth.login.config` 或 Java 安全属性文件来查找该配置文件。然后，它根据代理属性 `imq.user_repository.jaas.name` 的值在 JAAS 配置文件中选择一个入口。该入口指定将用于验证的登录模块。正如图中所示，代理可以使用多个登录模块。（图 1–3 显示了配置文件、登录模块和代理之间的关系。）

代理使用 JAAS 插件验证服务这一情况对 Message Queue 客户端来说仍是完全透明的。客户端继续像以前一样连接到代理，以便传递用户名和密码。反过来，代理使用回调处理程序将此信息传递给验证服务，该服务使用此信息验证用户并返回结果。如果验证成功，代理将允许建立连接；如果失败，客户端运行时环境将返回 JMS 安全异常，客户端必须对其进行处理。

验证 Message Queue 客户端后，如果要进行进一步授权，代理将按正常方式继续操作；它将查看访问控制文件，以确定是否授权经过验证的客户端执行它要完成的操作：访问目的地、使用消息和浏览队列等。

设置符合 JAAS 的验证

设置符合 JAAS 的验证的过程包括设置一些代理和系统属性，以便选择这种验证类型、指定配置文件位置以及指定要使用的登录模块入口。

本部分说明了 JAAS 客户端、登录模块和 JAAS 配置文件是如何相互关联的，然后介绍了设置符合 JAAS 的验证所需的过程。下图显示了配置文件、登录模块和代理之间的关系。

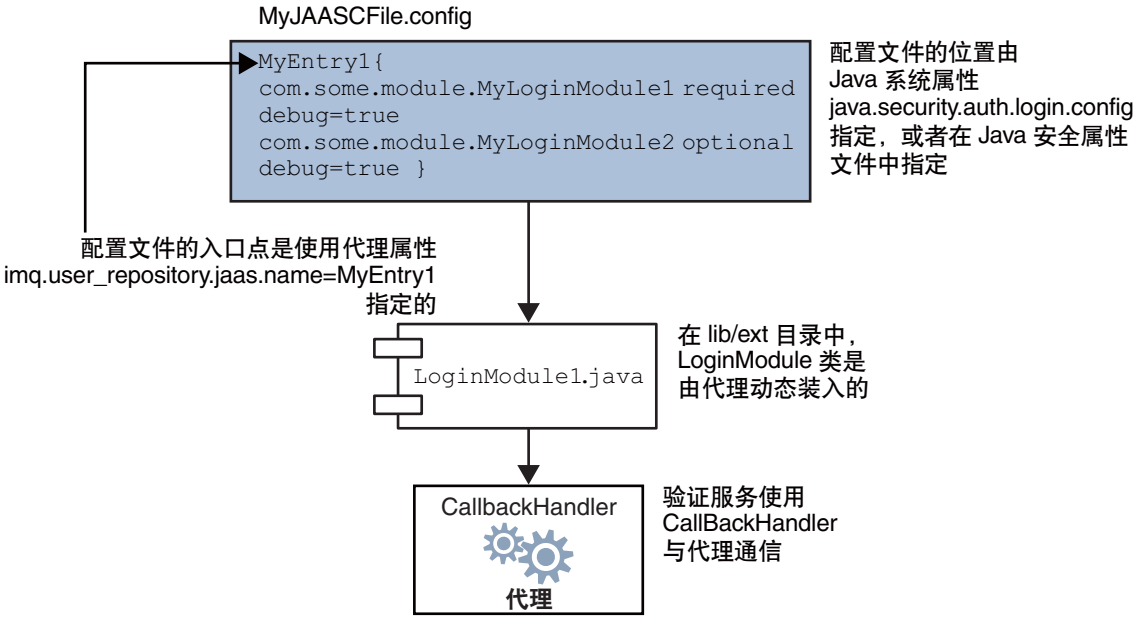

图 1–3 设置 JAAS 支持

正如图中所示，JAAS 配置文件 `MyJAASCFile.config` 包含对几个登录模块（组合在一个入口点中）的引用。代理通过查看 Java 系统属性 `java.security.auth.login.config` 或 Java 安全属性文件来查找配置文件。要使用的登录模块是通过查看代理属性 `imq.user_repository.jaas.name` 确定的，该属性指定配置文件中的所需入口。可以在 `lib/ext` 目录中找到这些模块的类。

要为 Message Queue 设置 JAAS 支持，您必须完成以下步骤。（在开发环境中，可以由开发者完成所有这些步骤。在生产环境中，将由管理员完成其中的某些任务。）

1. 创建一个或多个用于实现验证服务的登录模块类。下面列出了代理支持的 JAAS 回调类型。

   `javax.security.auth.callback.LanguageCallback`
   代理使用此回调为验证服务传递代理运行的语言环境。可以使用该值来进行本地化。

`javax.security.auth.callback.NameCallback`
代理使用此回调为验证服务传递 Message Queue 客户端在请求连接时指定的用户名。

`javax.security.auth.callback.TextInputCallback`
当 `TextInputCallback.getPrompt()` 为 `imq.authentication.type` 时，代理使用此回调为验证服务指定 `imq.authentication.type` 的值。目前，此字段的值只能为 `basic`。这表示使用 Base-64 密码编码。

`javax.security.auth.callback.PasswordCallback`
代理使用此回调为验证服务传递 Message Queue 客户端在请求连接时指定的密码。

`javax.security.auth.callback.TextOutputCallback`
通过将文本输出记录到代理日志文件中，代理使用此回调为验证服务提供日志记录服务。回调的消息类型 `ERROR`、`INFORMATION`、`WARNING` 分别映射到代理日志级别 `ERROR`、`INFO` 和 `WARNING`。

2. 使用引用登录模块类的入口创建 JAAS 配置文件，并为 Message Queue 管理员指定此文件的位置。（可以远程查找该文件，并且可以使用 url 指定其位置。）
3. 记下 JAAS 配置文件中的入口（它引用登录实现类）的名称。
4. 将用于实现登录模块的类归档为 jar 文件，然后将 jar 文件放在 Message Queue 的 `lib/ext` 目录中。
5. 配置与 JAAS 支持相关的代理属性。表 1–2 对此进行了描述。
6. 设置以下系统属性以指定 JAAS 配置文件的位置。

   `java.security.auth.login.config=`*JAAS_Config_File_Location*

   例如，可以在启动代理时指定配置文件。

   `imqbrokerd -Djava.security.auth.login.config=`*JAAS_Config_File_Location*

   可以使用其他方法来指定 JAAS 配置文件的位置。有关其他信息，请参见

   `http://java.sun.com/j2se/1.5.0/docs/guide/security/jaas/tutorials/LoginConfigFile.html`

下表列出了设置 JAAS 支持所需的代理属性。

表 1–2 用于 JAAS 支持的代理属性

属性	描述
`imq.authentication.type`	设置为 `basic` 以表示使用 Base-64 密码编码。这是 JAAS 验证唯一允许的值。
`imq.authentication.basic.user_repository`	设置为 `jaas` 以指定 JAAS 验证。
`imq.accesscontrol.type`	设置为 `file`。

表 1–2 用于 JAAS 支持的代理属性 （续）

属性	描述
imq.user_repository.jaas.name	设置为 JAAS 配置文件中的所需入口的名称，该入口引用要用作验证机制的登录模块。这是在步骤 3 中记下的名称。
imq.user_repository.jaas.userPrincipalClass	此属性（由 Message Queue 访问控制使用）指定登录模块中的 java.security.Principal 实现类，代理使用该类来提取主体名称，以表示 Message Queue 访问控制文件中的用户实体。如果未指定此属性，则改用在请求连接时从 Message Queue 客户端传递的用户名。
imq.user_repository.jaas.groupPrincipalClass	此属性（由 Message Queue 访问控制使用）指定登录模块中的 java.security.Principal 实现类，代理使用该类来提取主体名称，以表示 Message Queue 访问控制文件中的组实体。如果未指定此属性，则忽略 Message Queue 访问控制文件中的组规则（如果有）。

持久性存储库格式更改

Message Queue 4.1 版更改了 JDBC 存储库以支持高可用性。因此，JDBC 存储库版本已提高到 410。JDBC 存储库版本 350、370 和 400 将自动迁移到 410 版本模式。

请注意，基于文件的持久性存储库版本仍保持为 370，因为未对此版本进行任何更改。

代理配置

已将 IMQ_DEFAULT_EXT_JARS 属性添加到 imqenv.conf 文件中。可以设置此属性，以指定在代理启动时包含在 CLASSPATH 中的外部 .jar 文件的路径名。如果使用此属性指定外部 .jar 文件的位置，则不再需要将这些文件复制到 lib/ext 目录中。外部 jar 可以指 JDBC 驱动程序，也可以指 JAAS 登录模块。下面的样例命令指定了 jdbc 驱动程序的位置。

```
IMQ_DEFAULT_EXT_JARS=/opt/SUNWhadb4/lib/hadbjdbc4.jar:/opt/SUNWjavadb/derby.jar
```

JES Monitoring Framework 支持

Message Queue 支持 Sun Java Enterprise System (JES) Monitoring Framework，它允许使用通用图形界面来监视 Java Enterprise System 组件。此界面是由一个基于 Web 的控制台（名为 Sun Java System Monitoring Console）实现的。如果将 Message Queue 与其他 JES 组件一起运行，您可能会发现使用单个界面来管理所有这些组件要更方便一些。

JES 监视框架定义了所有 JES 组件产品使用的通用数据模型 (Common Data Model, CMM)。此模型为所有 JES 组件实现了集中且统一的视图。Message Queue 向 JES 监视框架公开以下对象：

- 安装的产品

- 代理实例名称
- 代理端口映射器
- 每个连接服务
- 每个物理目的地
- 持久性存储库
- 用户系统信息库

其中的每个对象将映射到 CMM 对象，可以使用 JES 监视控制台来监视 CMM 对象的属性。在运行时，管理员可以使用控制台来查看性能统计信息、创建规则（以自动进行监视）以及确认警报。有关将 Message Queue 对象映射到 CMM 对象的详细信息，请参见 Sun Java Enterprise System 监视指南。

要启用 JES 监视，您必须执行以下操作：

1. 按照 Sun Java Enterprise System 安装指南中提供的说明，安装并配置部署中的所有组件（Message Queue 和其他组件）。
2. 按照 Sun Java Enterprise System 监视指南中所述，为所有监视的组件启用并配置 Monitoring Framework。
3. 按照 Sun Java Enterprise System 监视指南中所述，在单独的主机上安装监视控制台，启动主代理，然后启动 Web 服务器。

使用 JES Monitoring Framework 并不会影响代理性能，因为收集度量的全部工作都是由监视框架完成的，它从代理的现有监视数据基础结构中提取数据。

事务管理

以前，仅允许通过管理方式回滚处于 PREPARED 状态的事务。也就是说，如果作为分布式事务一部分的会话没有正常终止，代理管理员将无法清除处于某种状态的事务。在 Message Queue 4.1 中，可使用 imqcmd 实用程序清除（回滚）处于以下状态的事务：STARTED、FAILED、INCOMPLETE、COMPLETE、PREPARED。

为帮助您确定能否回滚特定事务（特别是未处于 PREPARED 状态时），imqcmd 实用程序提供了额外的数据作为 imqcmd query txn 输出的一部分：它为启动事务的连接提供连接 ID，并指定事务的创建时间。通过使用此信息，管理员可以确定是否需要回滚事务。通常，管理员应避免提前回滚事务。

用于 C 客户端连接的固定端口

C 客户端可以使用 MQ_SERVICE_PORT_PROPERTY 连接属性来指定要连接到的固定端口。如果尝试通过防火墙或者需要绕过代理的端口映射器服务（用于动态分配端口），这可能是非常有用的。

请记住，还需要在代理端配置 JMS 服务端口。例如，如果要通过 ssljms 将客户端连接到端口 1756，您应该执行以下操作。

- 在客户端：将 MQ_SERVICE_PORT_PROPERTY 设置为 1756，然后将 MQ_CONNECTION_TYPE_PROPERTY 设置为 SSL。

- 在代理端：将 `imq.`*serviceNameType.protocol*`.port` 属性设置为 1756，如下所示。

```
imq.ssljms.ssl.port=1756
```

注 – `MQ_SERVICE_PORT_PROPERTY` 连接属性是在 Message Queue 3.7 版 Update 2 中引入的。

硬件和软件要求

有关 4.1 版的硬件和软件要求，请参阅《Sun Java System Message Queue 4.1 Installation Guide》。

关于 Message Queue 4.0

Message Queue 4.0 发行版仅限于支持 Application Server 9 PE。它是一个次要发行版，其中包括一些新功能，少量的增强功能和一些错误修复。本部分包含以下信息。

- 第 15 页中的 “4.0 发行版中的新增功能”
- 第 29 页中的 “硬件和软件要求”

4.0 发行版中的新增功能

Message Queue 4.0 包括以下新功能：

- 第 16 页中的 “对 C API 和 C 客户端运行时环境的接口更改”
- 第 16 页中的 “对 Java API 和 Java 客户端运行时环境的接口更改”
- 第 16 页中的 “显示有关持久性存储库的信息”
- 第 16 页中的 “持久性存储库格式更改”
- 第 17 页中的 “代理管理”
- 第 18 页中的 “JDBC 持久性支持”
- 第 18 页中的 “SSL 支持”
- 第 18 页中的 “JMX 支持”
- 第 23 页中的 “客户端运行时环境日志记录”
- 第 27 页中的 “连接事件通知”

这些功能将在以下各部分中进行描述。

注意 – 在版本 4.0 中进行了一些比较细微但可能会引起问题的更改，其中之一就是不再使用命令行选项来指定密码。此后，必须将所有密码都存储在文件中，如第 37 页中的 “过时的密码选项”中所述。

对 C API 和 C 客户端运行时环境的接口更改

Message Queue 4.0 版中增加了两个新属性，可在已放到停用消息队列中的所有消息上设置这些属性。

- JMS_SUN_DMQ_PRODUCING_BROKER 指示生成消息时所在的代理。
- JMS_SUN_DMQ_DEAD_BROKER 指示将消息标记为停用的代理。

对 Java API 和 Java 客户端运行时环境的接口更改

Message Queue 4.0 版中增加了两个新属性，可在已放到停用消息队列中的所有消息上设置这些属性。

- JMS_SUN_DMQ_PRODUCING_BROKER 指示生成消息时所在的代理。
- JMS_SUN_DMQ_DEAD_BROKER 指示将消息标记为停用的代理。

显示有关持久性存储库的信息

已将 query 子命令添加到 imqdbmgr 命令。使用此子命令可显示有关持久性存储库的信息，包括存储库版本、数据库用户以及是否已创建数据库表。

以下是此命令所显示的信息的示例。

```
imqdbmgr query

[04/Oct/2005:15:30:20 PDT] Using plugged-in persistent store:
        version=400
        brokerid=Mozart1756
        database connection url=jdbc:oracle:thin:@Xhome:1521:mqdb
        database user=scott
Running in standalone mode.
Database tables have already been created.
```

持久性存储库格式更改

为了提高性能，Message Queue 3.7 UR1 对持久性存储库格式进行了两处更改。一处是对文件存储库的更改，另一处是对 JDBC 存储库的更改。

- 文件存储库中保留的事务数据的格式

  已经对事务状态信息（存储在 Message Queue 的基于文件的持久性存储库中）的格式进行了更改，以减少磁盘 I/O 和提高 JMS 事务性能。

- Oracle JDBC 存储库

  在 Message Queue 的以前版本中，适用于 Oracle 的存储库结构使用 LONG RAW 数据类型来存储消息数据。在 Oracle 8 中，Oracle 引入了 BLOB 数据类型，而不再使用 LONG RAW 类型。Message Queue 3.7 UR1 已转为使用 BLOB 数据类型，以提高性能和提供更多支持。

由于这些更改会影响存储库兼容性，因此已将 Message Queue 3.7 UR1 中文件存储库和 JDBC 存储库的版本从 350 更改为 370。

Message Queue 4.0 对 JDBC 存储库进行了一些更改，以便实现优化和支持未来的增强功能。因此，JDBC 存储库版本已提高到 400。请注意，在 4.0 版中，基于文件的持久性存储库版本仍保持为 370，因为未对此版本进行任何更改。

Message Queue 4.0 支持将持久性存储库自动转换为基于文件的持久性存储库和 JDBC 持久性存储库的最新版本。`imqbrokerd` 首次启动时，如果实用程序检测到早期版本的存储库，则会将存储库迁移到新格式，并留下早期的存储库。

- 基于文件的存储库版本 200 和 350 将迁移到 370 版本格式。
- JDBC 存储库的 350 和 370 版本将被迁移到 400 版本格式。（如果需要升级 200 版本的存储库，需要先逐步升级到 3.5 或 3. 6 中间版本。）

如果需要回滚此升级，可以先卸载 Message Queue 4.0，然后重新安装以前运行的版本。由于未对存储库的早期副本进行任何更改，因此代理可以运行存储库的早期副本。

代理管理

命令实用程序 (`imqcmd`) 添加了一个子命令和几个选项，管理员可通过它们停止代理、在指定时间间隔后关闭代理、销毁连接或设置 java 系统属性（例如，与连接相关的属性）

- 停止代理可使其进入静默状态，以便在关闭或重新启动代理之前处理完所有消息。无法对处于停止状态的代理创建新连接。要停止代理，请输入如下命令。

  ```
  imqcmd quiesce bkr -b Wolfgang:1756
  ```

- 要在指定的时间间隔后关闭代理，请输入如下命令。（时间间隔指定了在代理关闭之前等待的秒数。）

  ```
  imqcmd shutdown bkr -b Hastings:1066 -time 90
  ```

  如果指定了时间间隔，代理将记录一条消息，指出何时关闭代理。例如，

  ```
  Shutting down the broker in 29 seconds (29996 milliseconds)
  ```

  当代理等待关闭时，可通过以下方式影响其行为。

  - 继续接受管理 jms 连接。
  - 不接受任何新 jms 连接。
  - 现有 jms 连接将继续工作。
  - 代理无法接管高可用性群集中的任何其他代理。
  - imqcmd 实用程序不会阻止，它将向代理发送关闭请求并立即返回。

- 要销毁连接，请输入如下命令。

  ```
  imqcmd destroy cxn -n 2691475382197166336
  ```

  使用 `imqcmd list cxn` 或 `imqcmd query cxn` 命令可获取连接 ID。

- 要使用 `imqcmd` 设置系统属性，请使用新的 –D 选项。它适用于设置或覆盖 JMS 连接工厂属性或与连接相关的 java 系统属性。例如：

```
imqcmd list svc -secure -DimqSSLIsHostTrusted=true
imqcmd list svc -secure -Djavax.net.ssl.trustStore=/tmp/mytruststore
           -Djavax.net.ssl.trustStorePassword=mytrustword
```

有关 `imqcmd` 命令语法的完整信息，请参见《Sun Java System Message Queue 4.1 Administration Guide》中的第 13 章 “Command Line Reference”。

JDBC 持久性支持

目前可以将 Apache Derby 版本 10.1.1 作为符合 JDBC 的持久性存储库提供程序。

SSL 支持

从 4.0 发行版开始，客户端连接工厂属性 `imqSSLIsHostTrusted` 的默认值为 `false`。如果应用程序依赖于以前的默认值 `true`，则需要对此属性进行重新配置，并将其明确设置为 `true`。

将代理配置为使用自签名证书时，您可以选择信任主机。在这种情况下，除了指定连接应使用基于 SSL 的连接服务（通过 imqConnectionType 属性）外，还应该将 `imqSSLIsHostTrusted` 属性设置为 true。

例如，要在代理使用自签名证书时安全地运行客户端应用程序，请使用如下命令。

```
java -DimqConnectionType=TLS
      -DimqSSLIsHostTrusted=true <ClientAppName>
```

要在代理使用自签名证书时安全地运行管理工具 `imqcmd`，请使用如下命令。

```
imqcmd list svc -secure -DimqSSLIsHostTrusted=true
```

JMX 支持

已添加符合 Java Management Extension (JMX) 规范的新 API，用于配置和监视 Message Queue 代理。使用此 API，可以在 Message Queue 客户端应用程序中以编程方式来配置和监视代理函数。在 Message Queue 的早期版本中，只能从命令行或管理控制台访问这些函数。

此 API 由一组 JMX **受管理** *Bean (MBean)* 组成，用于管理以下与 Message Queue– 相关的资源：

- 消息代理
- 连接服务
- 连接
- 目的地
- 消息生成方
- 消息使用方
- 事务

- 代理群集
- 日志记录
- Java 虚拟机 (Java Virtual Machine, JVM)

这些 MBean 提供了一些**属性**和**操作**，用于同步轮询和处理基础资源的状态，此外还提供了**通知**，以使客户端应用程序能够在状态发生更改时异步侦听和响应这些更改。使用 JMX API，客户端应用程序可以执行如下配置和监视任务：

- 设置代理的端口号
- 设置代理的最大消息大小
- 暂停连接服务
- 设置连接服务的最大线程数
- 获取服务上的当前连接数
- 销毁连接
- 创建目的地
- 销毁目的地
- 启用或禁用自动创建目的地
- 清除来自目的地的所有消息
- 获取自代理启动以来目的地收到的累积消息数
- 获取队列的当前状态（正在运行或已暂停）
- 获取某主题当前消息生成方的数量
- 清除来自长期订阅者的所有消息
- 获取当前 JVM 堆大小

有关 JMX API 的简介以及完整的参考信息，请参见《Sun Java System Message Queue 4.1 Developer's Guide for JMX Clients》。

代理支持：与 JMX 有关的属性

为了支持 JMX API，添加了几个新的代理属性（请参见表 1–3）。无法使用 Message Queue 命令实用程序 (`imqcmd`) 从命令行设置这些属性。可以使用代理实用程序 (`imqbrokerd`) 的 `-D` 选项设置这些属性，也可以在代理的实例配置文件 (`config.properties`) 中手动编辑这些属性。此外，某些属性（`imq.jmx.rmiregistry.start`、`imq.jmx.rmiregistry.use` 和 `imq.jmx.rmiregistry.port`）还可以使用表 1–4 中所述的新代理实用程序选项进行设置。此表列出了每个选项，指定了选项类型，并描述了选项用途。

表 1–3 用于支持 JMX 的新代理属性

属性	类型	描述
imq.jmx.rmiregistry.start	Boolean	指定在代理启动时是否启动 RMI 注册表。 如果为 true，代理将在 imq.jmx.rmiregistry.port 指定的端口上启动 RMI 注册表，并使用此注册表存储 JMX 连接器的 RMI 桩模块。请注意，在这种情况下将忽略 imq.jmx.rmiregistry.use 的值。 **默认值**：false
imq.jmx.rmiregistry.use	Boolean	指定是否使用外部 RMI 注册表。 仅当 imq.jmx.rmiregistry.start 为 false 时适用。 如果为 true，代理将在 imq.jmx.rmiregistry.port 指定的端口上使用外部 RMI 注册表，以便存储 JMX 连接器的 RMI 桩模块。代理启动时外部 RMI 注册表必须已处于运行状态。 **默认值**：false
imq.jmx.rmiregistry.port	Integer	RMI 注册表的端口号 仅当 imq.jmx.rmiregistry.start 或 imq.jmx.rmiregistry.use 为 true 时适用。然后即可将 JMX 连接器配置为使用 RMI 注册表，方法是将此端口号添加到这些连接器的 JMX 服务 URL 的 URL 路径中。 **默认值**：1099
imq.jmx.connector.list	String	预配置的 JMX 连接器的名称，以逗号分隔。 **默认值**：jmxrmi,ssljmxrmi
imq.jmx.connector.activelist	String	代理启动时要激活的 JMX 连接器的名称，以逗号分隔。 **默认值**：jmxrmi

表 1–3 用于支持 JMX 的新代理属性 （续）

属性	类型	描述
`imq.jmx.connector.`*connectorName* `.urlpath`	`String`	连接器 *connectorName* 的 JMX 服务 URL 的 *urlPath* 部分 当必须明确设置 JMX 服务 URL 路径时（例如使用共享的外部 RMI 注册表时），此选项会非常有用。 **默认值**：如果使用 RMI 注册表存储 JMX 连接器的 RMI 桩模块（即，如果 `imq.jmx.registry.start` 或 `imq.jmx.registry.use` 为 `true`）： `/jndi/rmi://`*brokerHost*`:`*rmiPort* */brokerHost/brokerPort/connectorName* 如果不使用 RMI 注册表（这是默认情况，`imq.jmx.registry.start` 和 `imq.jmx.registry.use` 均为 `false`）： `/stub/`*rmiStub* 其中 *rmiStub* 是 RMI 桩模块自身的编码和序列化表示
`imq.jmx.connector.`*connectorName* `.useSSL`	`Boolean`	指定连接器 *connectorName* 是否使用安全套接字层 (Secure Socket Layer, SSL)。 **默认值**：`false`
`imq.jmx.connector.`*connectorName* `.brokerHostTrusted`	`Boolean`	指定是否信任代理为连接器 *connectorName* 提供的任何证书。 仅当 `imq.jmx.connector.` *connectorName*`.useSSL` 为 `true` 时适用。 如果为 `false`，Message Queue 客户端运行时环境将验证提供给它的所有证书。如果证书的签名者不在客户端的信任存储库中，验证将失败。 如果为 `true`，则会跳过证书验证。此选项可能非常有用，例如，在使用自签名证书进行软件测试时。 **默认值**：`false`

`imq.jmx.connector.list` 属性用于定义一组要在代理启动时创建的命名 JMX 连接器；`imq.jmx.connector.activelist` 用于指定要激活其中哪些连接器。每个命名的连接器都有其自身的一组属性：

`imq.jmx.connector.`*connectorName* `.urlpath`
`imq.jmx.connector.`*connectorName* `.useSSL`
`imq.jmx.connector.`*connectorName* `.brokerHostTrusted`

默认情况下，将创建两个 JMX 连接器，名称分别为 `jmxrmi` 和 `ssljmxrmi`；前者被配置为不使用 SSL 加密 (`imq.jmx.connector.jmxrmi.useSSL = false`)，后者则使用 SSL 加密 (`imq.jmx.connector.ssljmxrmi.useSSL = true`)。默认情况下，代理启动时只激活 `jmxrmi` 连接器；有关如何激活 `ssljmxrmi` 连接器以进行安全通信的信息，请参见第 22 页中的 “针对 JMX 客户端的 SSL 支持”。

为了方便起见，还向命令行代理实用程序 (`imqbrokerd`) 中添加了新的选项（表 1–4），用于控制 RMI 注册表的使用、启动和端口。这些选项的用途和影响与表 1–3 中所述的等效代理属性相同。此表列出了每个选项，指定了选项的等效代理属性，并描述了选项用途。

表 1–4 用于支持 JMX 的新代理实用程序选项

选项	等效代理属性	描述
`-startRmiRegistry`	`imq.jmx.rmiregistry.start`	指定在代理启动时是否启动 RMI 注册表。
`-useRmiRegistry`	`imq.jmx.rmiregistry.use`	指定是否使用外部 RMI 注册表。
`-rmiRegistryPort`	`imq.jmx.rmiregistry.port`	RMI 注册表的端口号

命令行命令实用程序 (`imqcmd`) 中添加了一个新的子命令（表 1–5），用于列出代理启动时创建并启动的 JMX 连接器的 JMX 服务 URL。不使用 Message Queue 简便类 `AdminConnectionFactory` 的 JMX 客户端需要使用此信息来获取 JMX 连接器；另外，通过通用的 JMX 浏览器（例如 Java 监视和管理控制台 `jconsole`），还可以使用此信息来管理或监视 Message Queue。

表 1–5 新的命令实用程序子命令

子命令	描述
`list jmx`	列出 JMX 连接器的 JMX 服务 URL

针对 JMX 客户端的 SSL 支持

如上所述，默认情况下将使用预配置的 JMX 连接器 `jmxrmi` 来配置 Message Queue 消息代理，从而进行不安全的通信。需要使用安全套接字层 (Secure Socket Layer, SSL) 进行安全通信的应用程序，必须激活备用的安全 JMX 连接器 `ssljmxrmi`。这需要执行以下步骤：

1. 获取和安装签名证书，方法与为 `ssljms`、`ssladmin` 或 `cluster` 连接服务获取和安装签名证书（如 Message Queue 管理指南 中所述）相同。
2. 在信任存储库中安装根证书颁发机构颁发的证书（如有必要）。
3. 将 `ssljmxrmi` 连接器添加到代理启动时要激活的 JMX 连接器列表：

   ```
   imq.jmx.connector.activelist=jmxrmi,ssljmxrmi
   ```

4. 使用 Message Queue 代理实用程序 (`imqbrokerd`) 启动代理，可以将密码文件中的密钥库密码传递给此实用程序，也可以在提示时从命令行键入密码。
5. 默认情况下，`ssljmxrmi` 连接器（或基于 SSL 的任何其他连接器）被配置为验证提供给它的所有代理 SSL 证书。要避免进行此验证（例如，在使用自签名证书进行软件测试时），请将代理属性 `imq.jmx.connector.ssljmxrmi.brokerHostTrusted` 设置为 `true`。

在客户端，必须使用将 `ssljmxrmi` 指定为首选连接器的 URL 配置管理员连接工厂 (`AdminConnectionFactory`)：

```
AdminConnectionFactory  acf = new AdminConnectionFactory();
acf.setProperty(AdminConnectionConfiguration.imqAddress, "mq://myhost:7676/ssljmxrmi");
```

如果需要，可使用系统属性 `javax.net.ssl.trustStore` 和 `javax.net.ssl.trustStorePassword` 将 JMX 客户端指向信任存储库。

客户端运行时环境日志记录

本部分介绍了 Message Queue 4.0 对客户端运行时环境日志记录的支持，记录内容包括与连接和会话有关的事件。

JDK 1.4（及更高版本）包含 `java.util.logging` 库。此库实现了一个标准记录程序接口，可用于特定于应用程序的日志记录。

Message Queue 客户端运行时环境使用 Java 日志记录 API 来实现日志记录功能。可以使用所有 J2SE 1.4 日志记录工具来配置日志记录活动。 例如，应用程序可以使用以下 Java 日志记录工具来配置 Message Queue 客户端运行时环境输出日志记录信息的方式：

- 日志记录处理程序
- 日志记录过滤器
- 日志记录格式化程序
- 日志记录级别

有关 Java 日志记录 API 的详细信息，请参见位于 `http://java.sun.com/j2se/1.4.2/docs/guide/util/logging/overview.html` 的 Java 日志记录概述

日志记录的名称空间、级别和活动

Message Queue 提供程序定义了一组与日志记录级别和日志记录活动相关联的日志记录名称空间，以使 Message Queue 客户端能够在正确设置日志记录配置后记录连接和会话事件。

Message Queue 客户端运行时环境的根日志记录名称空间被定义为 `javax.jms`。Message Queue 客户端运行时环境中的所有记录程序均将此名称用作父名称空间。

用于 Message Queue 客户端运行时环境的日志记录级别与 java.util.logging.Level 类中定义的级别相同。此类定义了七个标准日志级别，另外还有两个用于打开和关闭日志记录的设置。

OFF　关闭日志记录。

SEVERE　最高优先级，最大值。由应用程序定义。

WARNING　由应用程序定义。

INFO　由应用程序定义。

CONFIG　由应用程序定义。

FINE　由应用程序定义。

FINER　由应用程序定义。

FINEST　最低优先级，最小值。由应用程序定义。

ALL　启用所有消息的日志记录。

通常，Message Queue 客户端运行时环境中发生的异常和错误由具有 javax.jms 名称空间的记录程序进行记录。

- 从 JVM 抛出并由客户端运行时环境捕获的异常（例如 IOException）由具有 javax.jms 日志记录名称空间的记录程序记录为 WARNING 级别。
- 从客户端运行时环境抛出的 JMS 异常（例如 IllegalStateException）由具有 javax.jms 日志记录名称空间的记录程序记录为 FINER 级别。
- 从 JVM 抛出并由客户端运行时环境捕获的错误（例如 OutOfMemoryError）由具有 javax.jms 日志记录名称空间的记录程序记录为 SEVERE 级别。

下表列出了可记录的事件和日志级别，要记录 JMS 连接事件和会话事件，必须设置此级别。

下表描述了连接的日志级别和事件。

表 1–6 javax.jms.connection 名称空间的日志级别和事件

日志级别	事件
FINE	连接已创建
FINE	连接已启动
FINE	连接已关闭
FINE	连接已断开
FINE	已重新连接

表 1–6 javax.jms.connection 名称空间的日志级别和事件 （续）

日志级别	事件
FINER	其他连接活动，例如 setClientID
FINEST	消息、确认、Message Queue 操作和控制消息（例如提交事务）

对于会话，将在日志记录中记录以下信息。

- 传送给使用方的消息的每个日志记录包括 ConnectionID、SessionID 和 ConsumerID。
- 由生成方发送的消息的每个日志记录包括 ConnectionID、SessionID、ProducerID 和目的地名称。

下表描述了会话的日志级别和事件。

表 1–7 javax.jms.session 名称空间的日志级别和事件

日志级别	事件
FINE	会话已创建
FINE	会话已关闭
FINE	生成方已创建
FINE	使用方已创建
FINE	目的地已创建
FINER	其他会话活动，例如提交会话。
FINEST	已生成并使用消息。（未在日志记录中记录消息属性和主体。）

默认情况下，输出日志级别是从运行应用程序所在的 JRE 继承而来的。可查看 JRE_DIRECTORY/lib/logging.properties 文件以确定该级别。

可通过编程方式或使用配置文件来配置日志记录，还可以控制进行日志记录的范围。以下部分介绍了这些可行的操作。

使用 JRE 日志记录配置文件

以下示例说明了如何在 JRE_DIRECTORY/lib/logging.properties 文件（用于设置 Java 运行环境的日志级别）中设置日志记录的名称空间和级别。使用此 JRE 的所有应用程序都将具有相同的日志记录配置。下面的样例配置将 javax.jms.connection 名称空间的日志记录级别设置为 INFO，并指定将输出写入 java.util.logging.ConsoleHandler。

```
#logging.properties file.
# "handlers" specifies a comma separated list of log Handler
# classes. These handlers will be installed during VM startup.
```

```
# Note that these classes must be on the system classpath.
# By default we only configure a ConsoleHandler, which will only
# show messages at the INFO and above levels.

    handlers= java.util.logging.ConsoleHandler

# Default global logging level.
# This specifies which kinds of events are logged across
# all loggers. For any given facility this global level
# can be overriden by a facility-specific level.
# Note that the ConsoleHandler also has a separate level
# setting to limit messages printed to the console.

    .level= INFO

# Limit the messages that are printed on the console to INFO and above.

    java.util.logging.ConsoleHandler.level = INFO
    java.util.logging.ConsoleHandler.formatter =
                                    java.util.logging.SimpleFormatter

# The logger with javax.jms.connection name space will write
# Level.INFO messages to its output handler(s). In this configuration
# the ouput handler is set to java.util.logging.ConsoleHandler.

    javax.jms.connection.level = INFO
```

为特定的应用程序使用日志记录配置文件

还可以从用于运行应用程序的 Java 命令行定义日志记录配置文件。该应用程序将使用在指定的日志记录文件中定义的配置。在以下示例中，`configFile` 使用的格式与 `JRE_DIRECTORY/lib/logging.properties` 文件中定义的格式相同。

```
java -Djava.util.logging.config.file=configFile MQApplication
```

以编程方式设置日志记录配置

以下代码使用 `java.util.logging` API 来记录连接事件，方法是将 `javax.jms.connection` 名称空间的日志级别更改为 `FINE`。您可以在应用程序中包含此类代码，以便通过编程方式设置日志记录配置。

```
import java.util.logging.*;
//construct a file handler and output to the mq.log file
//in the system's temp directory.

    Handler fh = new FileHandler("%t/mq.log");
    fh.setLevel (Level.FINE);
```

```
//Get Logger for "javax.jms.connection" domain.

    Logger logger = Logger.getLogger("javax.jms.connection");
    logger.addHandler (fh);

//javax.jms.connection logger would log activities
//with level FINE and above.

    logger.setLevel (Level.FINE);
```

连接事件通知

连接事件通知允许 Message Queue 客户端侦听关闭和重新连接事件，并根据通知类型和连接状态采取适当的操作。例如，如果发生故障转移，并且客户端重新连接到了其他代理，则应用程序可能希望清除事务状态，并继续执行新事务。

如果 Message Queue 提供程序检测到某个连接存在严重问题，它将调用该连接对象的已注册的异常侦听器。它调用侦听器的 `onException` 方法，并向其传递一个用于描述该问题的 `JMSException` 参数。Message Queue 提供程序还提供一个事件通知 API，该 API 允许客户端运行时环境向应用程序通知有关连接状态更改的信息。通知 API 由以下元素定义：

- `com.sun.messaging.jms.notification` 软件包，用于定义事件侦听器和通知事件对象。
- `com.sun.messaging.jms.Connection` 接口，用于定义 `javax.jms.Connection` 接口的扩展。

以下部分介绍了可以触发通知的事件，并说明了如何创建事件侦听器。

连接事件

下表列出并介绍了事件侦听器可以返回的事件。

请注意，当发生连接事件时，不会调用 JMS 异常侦听器。仅当客户端运行时环境达到最大重新连接尝试次数时，才会调用异常侦听器。客户端运行时环境在调用异常侦听器之前，始终会调用事件侦听器。

表 1–8 通知事件

事件类型	含义
`ConnectionClosingEvent`	如果 Message Queue 客户端运行时环境从代理收到通知，指明由于管理员请求关闭而即将关闭连接，则 Message Queue 客户端运行时环境将生成此事件。

表 1–8 通知事件 （续）

事件类型	含义
ConnectionClosedEvent	关闭连接时（由于代理错误或管理员请求关闭或重新启动连接），Message Queue 客户端运行时环境将生成此事件。 如果事件侦听器收到 ConnectionClosedEvent，则应用程序可以使用所收到的事件的 getEventCode() 方法，来获取用于指明关闭原因的事件代码。
ConnectionReconnectedEvent	Message Queue 客户端运行时环境已重新连接到代理。此代理可以是客户端先前连接的代理，也可以是其他代理。 应用程序可以使用所收到的事件的 getBrokerAddress 方法，来获取它所重新连接到的代理的地址。
ConnectionReconnectFailedEvent	Message Queue 客户端运行时环境无法重新连接到代理。每次重新连接尝试失败时，该运行时环境都会生成新的事件，并将该事件传送到事件侦听器。 发生连接事件时不会调用 JMS 异常侦听器。仅当客户端运行时环境达到最大重新连接尝试次数时，才会调用 JMS 异常侦听器。客户端运行时环境在调用异常侦听器之前，始终会调用事件侦听器。

创建事件侦听器

以下代码示例说明了如何设置连接事件侦听器。只要发生连接事件，客户端运行时环境就会调用事件侦听器的 onEvent 方法。

```
//create an MQ connection factory.

com.sun.messaging.ConnectionFactory factory =
        new com.sun.messaging.ConnectionFactory();

//create an MQ connection.

com.sun.messaging.jms.Connection connection =
       (com.sun.messaging.jms.Connection )factory.createConnection();

//construct an MQ event listener.  The listener implements
//com.sun.messaging.jms.notification.EventListener interface.

com.sun.messaging.jms.notification.EventListener eListener =
       new ApplicationEventListener();
```

```
//set event listener to the MQ connection.

connection.setEventListener ( eListener );
```

事件侦听器示例

在此示例中，应用程序选择让事件侦听器将连接事件记录到应用程序的日志记录系统中：

```
public class ApplicationEventListener implements
                com.sun.messaging.jms.notification.EventListener {

public void onEvent ( com.sun.messaging.jms.notification.Event connEvent ) {
          log (connEvent);
}
private void log ( com.sun.messaging.jms.notification.Event connEvent ) {
          String eventCode = connEvent.getEventCode();
          String eventMessage = connEvent.getEventMessage();
          //write event information to the output stream.
         }
}
```

硬件和软件要求

有关 4.0 版硬件和软件要求，请参考《Sun Java System Application Server Platform Edition 9 发行说明》。

此发行版中修复的错误

下表显示了 Message Queue 4.1 版中修复的错误。

表 1–9 Message Queue 4.1 中修复的错误

错误	描述
6381703	如果传送消息的代理重新启动，则可能会两次提交处理的远程消息。
6388049	无法清除未完成的分布式事务。
6401169	imqcmd 的提交和回滚选项不提示进行确认。
6473052	自动创建的队列的默认设置应该为循环传送。(`MaxNumberConsumers` = -1)。
6474990	代理日志显示 `imqcmd list dst` 命令发生了 `ConcurrentModificationException`。

表 1–9 Message Queue 4.1 中修复的错误 （续）

错误	描述
6487413	限制行为是 REMOVE_OLDEST 或 REMOVE_LOWER_PRIORITY 时发生内存泄漏。
6488340	代理自旋，并且客户端等待确认回复。
6502744	代理不遵循停用消息队列具有 1000 个消息的默认限制。
6517341	当客户端连接到高可用性群集时，客户端运行时环境需要改进重新连接逻辑，以便允许客户端重新连接，而无论 *imqReconnectEnabled* 属性值是什么。
6528736	Windows 自动启动服务 (*imqbrokersvc*) 在启动过程中崩溃。
6561494	当两个使用方共享一个会话时，消息将传送到错误的使用方。
6567439	如果在代理重新启动后提交 PREPARED 事务中生成的消息，这些消息的传送顺序就会不正确。

下表介绍了 Message Queue 4.0 中修复的错误。

表 1–10 Message Queue 4.0 中修复的错误

错误号	描述
4986481	在 Message Queue 3.5 中调用 Session.recover 时，可能会在自动重新连接模式下挂起。
4987325	在调用 Session.recover 后，将重新传送的消息的 Redelivered 标志设置为 false。
6157073	将新连接消息更改为既包含总连接数，也包含服务上的连接数。
6193884	在使用非 ASCII 字符显示消息的语言环境中，Message Queue 向系统日志输出垃圾消息。
6196233	无法使用 JMSMessageID 选择消息。
6251450	在群集关闭过程中，connectList 出现 ConcurrentModificationException。
6252763	java.nio.HeapByteBuffer.putLong/Int 出现 java.nio.BufferOverflowException。
6260076	使用 Oracle 存储库，在代理启动后的第一条消息发布完后，消息发布速度变得很慢。
6260814	对 JMSXUserID 进行处理的选择器始终得到 false 值。
6264003	队列浏览器显示属于尚未提交的事务的消息。
6271876	关闭具有未使用消息的使用方时，连接流控制无法正常工作。
6279833	Message Queue 不允许两个代理使用相同的 JDBC 表。
6293053	除非清除存储库中的内容（使用 –reset store），否则当系统的 IP 地址更改时，主代理将无法正确启动）

表 1–10 Message Queue 4.0 中修复的错误 （续）

错误号	描述
6294767	Message Queue 代理需要在其打开的网络套接字上设置 SO_REUSEADDR。
6304949	无法设置 TopicConnectionFactory 的 ClientID 属性。
6307056	txn 日志成为性能瓶颈。
6320138	Message Queue C API 无法确定回复标头中的队列名称。
6320325	即使在 Solaris 上同时安装了 JDK 1.4 和 JDK 1.5，代理有时也会选取前者。
6321117	多代理群集初始化抛出 java.lang.NullPointerException。
6330053	从订户提交事务时，JMS 客户端抛出 java.lang.NoClassDefFoundError。
6340250	支持 C-API 中的 MESSAGE 类型。
6351293	添加对 Apache Derby 数据库的支持。

重要信息

本部分介绍核心产品文档中未包括的最新信息。本部分包含以下主题：

- 第 31 页中的 “安装说明”
- 第 31 页中的 “兼容性问题”
- 第 32 页中的 “Message Queue 4.1 的文档更新”

安装说明

有关预安装说明和升级过程的信息以及在 Solaris、Linux 和 Windows 平台上安装 Message Queue Platform Edition 的所有其他相关信息，请参阅《Sun Java System Message Queue 4.1 Installation Guide》。

有关预安装说明的信息以及在 Solaris、Linux 和 HPUX 平台上安装 Message Queue Enterprise Edition 的所有其他相关信息，请参阅 Sun Java Enterprise System 安装指南。

有关在 Solaris、Linux、HPUX 和 Windows 平台上升级到 Message Queue Enterprise Edition 的升级和迁移说明的信息，请参阅 Sun Java Enterprise System 升级和迁移指南。

兼容性问题

本部分包含 Message Queue 4.1 中的兼容性问题。

接口稳定性

Sun Java System Message Queue 使用的许多接口可能会随着时间的变化而发生更改。《Sun Java System Message Queue 4.1 Administration Guide》中的附录 B "Stability of Message Queue Interfaces"根据接口的稳定性对其进行了分类。接口越稳定，在产品的后续版本中对其进行更改的可能性就越小。

与 Message Queue 的下一个主要发行版相关的问题

在 Message Queue 下一个主要发行版中进行的某些更改可能会导致您的客户端与该发行版不兼容。现在提供此信息是为了让您针对这些更改做好准备。

- 作为 Sun Java System Message Queue 的一部分进行安装的个别文件的位置可能会发生更改。如果现有的应用程序依赖于某些 Message Queue 文件的当前位置，则这些应用程序可能会中断。
- 3.5 及更早版本的代理可能无法再在具有较高版本代理的群集中运行。
- 在以后的版本中，Message Queue 客户端可能无法使用早于 1.5 的 JDK 版本。

Message Queue 4.1 的文档更新

与本发行说明文档不同，Message Queue 4.1 只包含一个新文档：《Sun Java System Message Queue 4.1 Developer's Guide for JMX Clients》。Message Queue 4.0 发行版中引入了此文档。在 4.1 版中，已添加了介绍 JMX 模型的概念性信息。

针对 Message Queue 3.6 SP3, 2005Q4 发布的 Message Queue 文档在有关 Application Server 9 PE 客户端需求方面包含了最新内容。可在以下位置找到此文档集。

`http://docs.sun.com/app/docs/coll/1307.1`

安装和升级信息

更新了《Sun Java System Message Queue 4.1 Installation Guide》，以反映特定于平台的信息。此文档现在包含与 Message Queue 4.1 相关的安装和升级信息。

管理指南

更新了管理指南，以提供有关高可用性群集、JAAS 支持以及 JMX 支持的信息。

适用于 Java 客户端的开发者指南

更新了适用于 Java 客户端的开发者指南，以反映添加了客户端运行时环境日志记录支持以及连接事件通知。

适用于 C 客户端的开发者指南

更新了适用于 C 客户端的开发者指南，以反映添加了 `MQGetDestinationName` 函数、`MQ_Message` 消息类型以及固定端口。

已知问题和限制

本部分包含了 Message Queue 4.1 中已知问题的列表。涵盖以下产品领域：

有关当前错误、错误的状态和解决方法的列表，Java Developer Connection™ 成员应参见 Java Developer Connection Web 站点上的 Bug Parade 页。在报告新的错误之前请先查看该页。虽然未列出所有的 Message Queue 错误，但如果您想了解是否已报告了某个问题，可以将该页作为一个很好的起点。

```
http://bugs.sun.com/bugdatabase/index.jsp
```

注 – 可以免费获得 Java Developer Connection 成员资格，但需要进行注册。有关如何成为 Java Developer Connection 成员的详细信息，请访问 Sun 的 "For Developers" Web 页。

要报告新错误或提交功能请求，请向 `imq-feedback@sun.com` 发送电子邮件。

安装问题

本部分介绍了与 Message Queue 4.1 版安装相关的问题。

产品注册表和 JES

Message Queue 4.1 版是由新安装程序安装的，它还安装并升级了 Message Queue 所需的共享组件；例如，JDK、NSS 库和 JavaHelp 等。此安装程序和 Java Enterprise System (JES) 安装程序并不共享相同的产品注册表。如果 Message Queue 安装程序删除随 JES 安装的 Message Queue 版本并升级到 Message Queue 4.1，JES 产品注册表可能会处于不一致的状态。因此，在运行 JES 卸载程序时，它可能会误删 Message Queue 4.1 及其所依赖的共享组件，这些内容并不是它安装的。

升级 JES 安装程序所安装的软件的最佳方法如下所示。

1. 使用 JES 卸载程序删除 Message Queue 及其共享组件。
2. 使用 Message Queue 安装程序安装 Message Queue 4.1。

选择合适的 JRE

在 Message Queue 4. 1 安装程序 JDK 选择屏幕中，您可以选择系统上的现有 JDK/JRE 以供 Message Queue 使用。遗憾的是，显示的列表还包含用于运行安装程序的 JRE。此 JRE 是安装程序包的一部分，并未实际安装在系统上。（**错误** *6585911*）

安装程序使用的 JRE 可以通过其路径进行识别，该路径应该位于解压缩的安装程序目录中，并且应包括子目录 `mq4_1–installer`。例如：

some_directory`/mq4_1–installer/usr/jdk/instances/jdk1.5.0/jre`

不要选择此 JRE 以供 Message Queue 使用。而应选择系统上的其他 JDK。如果不存在其他 JDK，则针对您的平台执行相应的操作。

- Solaris 或 Linux：选择“安装并使用默认 JDK”。
- Windows：在运行 Message Queue 4.1 安装程序之前，下载并安装 JDK。

在 Windows 上进行安装

在 Windows 上安装 Message Queue 时，请注意以下限制。

- 安装程序没有在“开始”>“程序”菜单中添加 Message Queue 条目（**错误** *6567258*）。要启动管理控制台，请使用《Sun Java System Message Queue 4.1 Administration Guide》中的“Starting the Administration Console”所示的命令行。
- 安装程序没有将 `IMQ_HOME\mq\bin` 目录添加到 `PATH` 环境变量中。（**错误** *6567197*）。在调用 Message Queue 实用程序 (`IMQ_HOME\mq\bin\`*command*) 时，用户需要将此条目添加到其 `PATH` 环境变量中，或者提供完整的路径名。
- 安装程序没有将条目添加到 Windows 注册表中，以表明安装了 Message Queue。
- 在无提示模式下运行时，安装程序将立即返回。确实执行了安装；但用户无法知道无提示安装实际上是何时完成的。（**错误** *6586560*）
- Windows 上不支持文本模式 (`installer –t`)。如果在 Windows 上以文本模式运行安装程序，则会显示错误消息。即使在非英语语言环境中运行安装程序，也会以英语显示此消息。（**错误** *6594142*）
- 即使在非英语语言环境中运行安装程序，也会在安装程序安装主目录屏幕上以英语显示字符串 "Install Home"。（**错误** *6592491*）

在 Solaris 上进行安装

错误消息和“不完整”摘要状态会误导尝试使用`installer-n` 命令进行安装的用户。实际上，已成功执行了该命令。（**错误** *6594351*）

在 Linux 上进行安装

以下问题会影响 Linux 平台上的安装

- 在“JDK 选择”面板上，滚动列表仅显示一项。这使得在列表中选择其他 JDK 变得非常困难。（**错误** *6584735*）

- 如果 JDK 是最新的，并且用户在 JDK 选择屏幕上选择了“安装默认 JDK”，则安装程序仍会尝试安装 JDK，并报告无法安装软件包。尽管出现此问题，但安装已成功完成。（**错误** *6581310*）
- 在模拟运行模式 (`installer –n`) 下运行安装程序时，摘要屏幕将显示一些错误消息，并且还会显示“不完整”安装状态。此状态是错误的并且会误导用户；模拟运行并未在系统上安装任何内容；它仅创建一个可随后用于安装的应答文件。（**错误** *6594351*）
- 如果系统上存在旧版本的 Message Queue 本地化 RPM，Message Queue 4.1 本地化 RPM 安装（在多语种软件包屏幕上选中“安装 Message Queue 多语种软件包”复选框时，会执行此类安装）将会失败。安装失败的原因是与以前 3.7 UR1 安装中的 Il8 软件包发生冲突。（**错误** *6594381*）

  解决方法 先使用 `rpm –e` 命令手动删除本地化 RPM，然后再运行 4.1 安装程序。要确定此处相关的 RPM，请参见《Sun Java System Message Queue 4.1 Installation Guide》中的“Message Queue Packages (RPMs)”。

在所有平台上进行安装

这些问题会影响所有平台上的安装。

- 当安装程序正在安装 Message Queue 4.1 并显示进度屏幕时，“取消”按钮处于活动状态。如果此时选择“取消”按钮，则会导致安装不完整或中断。（**错误** *6595578*）
- 安装程序摘要屏幕包含一些链接，单击这些链接时将启动日志或摘要页面查看器。如果使用窗口关闭按钮 "X" 关闭此查看器窗口，而不是使用标有“关闭”的按钮，则无法重新打开此查看器窗口。（**错误** *6587138*）

  解决方法 使用标有“关闭”的按钮关闭此窗口。
- 当系统装有旧版本的 Message Queue 和 NSS/NSPR 时，安装程序升级仅列出需要升级的 Message Queue；它不会提到需要升级 NSS/NSPR。仅更新屏幕存在此问题，因为所有相关软件将作为安装过程的一部分进行升级（如安装就绪屏幕所示，其中显示了正确的信息）。（**错误** *6580696*）

  解决方法 不需要任何解决方法，因为如果 NSS/NSPR 文件不是最新的，则会安装这些文件并卸载旧版本。
- 在文本模式 (`installer –t`) 下运行安装程序或卸载程序时，摘要屏幕将显示包含日志/摘要文件的目录，但不会列出这些文件的名称。（**错误** *6581592*）
- 如果具有指定名称的文件不存在，则会生成不一致且含糊不清的错误消息。（**错误** *6587127*）

版本信息

安装程序未明确显示 Message Queue 版本信息。（**错误** *6586507*）

在 Solaris 平台上，请参阅下表以确定所安装的版本。

表 1–11 版本格式

安装程序所显示的版本	Message Queue 发行版
4.1.0.0	4.1
3.7.0.1	3.7 UR1
3.7.0.2	3.7 UR2
3.7.0.3	3.7 UR3
3.6.0.0	3.6
3.6.0.1	3.6 SP1
3.6.0.2	3.6 SP2
3.6.0.3	3.6 SP3
3.6.0.4	3.6 SP4

注 – 对于 3.6 SP4 修补程序发行版（如 3.6 SP4 Patch 1），安装程序显示的发行版字符串保持不变。您需要运行 `imqbrokerd –version` 命令以确定确切的版本。

在 Linux 平台上，无法提供简单的格式转换。Linux 上的安装程序显示的版本号采用以下格式。

```
<majorReleaseNumber>.<minorReleaseNumber>-<someNumber>
```

例如，3.7–22。它告诉我们这是 3.7 发行版之一，但没有指出具体的版本。要确定该版本，请运行 `imqbrokerd –version` 命令。

本地化问题

以下问题与本地化问题有关。

- 在非英语语言环境中以文本模式 (`installer –t`) 运行安装程序时，多字节字符将显示为乱码。（**错误** *6586923*）
- 用户可以在安装程序摘要屏幕中查看摘要报告。遗憾的是，在多字节语言环境中运行安装程序时，此报告（HTML 页）显示乱码。（**错误** *6587112*）

  解决方法 编辑 HTML 文件以纠正其中指定的字符集。HTML 文件应包含如下内容。

  `meta http-equiv=”Content-Type” content=”text/html; charset=UTF-8`

  将 "UTF-8" 替换为 *locale_name*`.UTF-8`。例如，Solaris 上的 `ja_JA.UTF-8` 或 `ko.UTF-8`；linux 上的 `ja_JA.utf8` 或 `ko_KO.utf8`。
- 在安装程序进度屏幕上，进度栏显示奇怪的字符。在非英语语言环境中，工具提示是固定编码的。（**错误** *6591632*）

- Windows 上不支持文本模式 (`installer –t`)。如果在 Windows 上以文本模式运行安装程序，则会显示错误消息。在非英语语言环境中运行安装程序时，不会本地化此消息。（**错误** *6594142*）
- 无论在哪种语言环境中运行安装程序，安装程序的许可证屏幕都会显示英语许可证文本。（**错误** *6592399*）

  解决方法 要访问本地化的许可证文件，请查看 `LICENSE_MULTILANGUAGE.pdf` 文件。
- 未本地化安装程序使用帮助文本。（**错误** *6592493*）
- 安装程序摘要 HTML 页上显示的字符串 "None" 是用英语固定编码的。（**错误** *6593089*）
- 除了法语以外的语言环境均未本地化版权页面。（**错误** *6590992*）
- 在德语语言环境中运行安装程序时，欢迎屏幕未显示在其他语言环境中看到的完整文本。（**错误** *6592666*）
- 安装程序安装主目录屏幕上显示的字符串 "Install Home" 没有进行本地化。即使在非英语语言环境中运行安装程序，也会以英语显示该字符串。（**错误** *6592491*）
- 在文本模式 (`installer –t`) 下运行安装程序时，无论在哪种语言环境中运行安装程序，都会使用英语响应选项 "Yes" 和 "No"。（**错误** *6593230*）
- 安装程序 JDK 选择屏幕上的浏览器按钮工具提示是使用英语固定编码的。（**错误** *6593085*）

过时的密码选项

在以前版本的 Message Queue 中，可以对以下命令使用 `–p` 或 `–password` 选项以交互方式来指定密码：`imqcmd`、`imqbrokerd` 和 `imdbmgr`。从 4.0 版开始，将不再使用这些选项。必须按照以下方式提供密码。

1. 在仅用于存储密码的文件中将密码属性设置为所需的值。

   使用以下语法在密码文件中指定密码。

   PasswordPropertyName= MyPassword
2. 使用 `–passfile` 选项传递密码文件的名称。

密码文件可以包含以下列出的一个或多个密码。

- 用于打开 SSL 密钥库的密钥库密码。可使用 `imq.keystore.password` 属性指定此密码。
- LDAP 系统信息库密码，用于在非匿名连接的情况下与 LDAP 目录进行安全连接。可使用 `imq.user_repository.ldap.password` 属性指定此密码。
- 用于连接到符合 JDBC 的数据库的 JDBC 数据库密码。可使用 `imq.persist.jdbc.vendorName.password` 属性指定此密码。属性名称的 *vendorName* 组件是用于指定数据库供应商的变量。选项包括 `hadb`、`derby`、`pointbase`、`oracle` 或 `mysql`。

- imqcmd 命令（用于执行代理管理任务）的密码。可使用 imq.imqcmd.password 属性指定此密码。

在以下示例中，将 JDBC 数据库的密码设置为 abracadabra。

```
imq.persist.jdbc.mysql.password=abracadabra
```

可以将代理配置为使用密码文件，该文件可通过以下任一方法创建。

- 在代理的 config.properties 文件中设置以下属性。

  ```
  imq.passfile.enabled=true
  imq.passfile.dirpath=MyFileDirectory
  imq.passfile.name=MyPassfileName
  ```

- 使用 imqbrokerd 命令的 –passfile 选项。

  ```
  imqbrokerd –passfile MyPassfileName
  ```

一般问题

本部分包含 Message Queue 4.1 中的一般问题。其中某些问题是以前的 Message Queue 版本引入的。

- 使用 HTTP 传输的 JMS 客户端突然终止时（例如，使用 Ctrl-C），代理要花费大约一分钟的时间才能释放客户端连接和所有关联的资源。

  如果在这一分钟内客户端的另一个实例启动，并且该实例尝试使用同一个客户端 ID、长期订阅或队列，则可能会收到“客户端 ID 已经在使用”的异常。这实际上不是什么问题，只是上述终止过程的副作用。如果客户端在延迟约一分钟后启动，则应当一切正常。

- SOAP 客户端。以前，用于引用 mail.jar 和 mail.jar 的 SAAJ 1.2 实现 jar 无需位于 CLASSPATH 中。在 SAAJ 1.3 中删除了此引用，因此 Message Queue 客户端必须将 mail.jar 明确放入 CLASSPATH 中。

管理/配置问题

以下是有关管理和配置 Message Queue 的问题

- 在 Windows 计算机上，当 CLASSPATH 包含双引号时，imqadmin 和 imqobjmgr 实用程序将抛出错误（**错误号** *5060769*）。

  解决方法 可以忽略此错误消息；代理会正确处理，并将任何错误通知使用方。此错误不影响系统的可靠性。

- 如果所提供的值中包含空格，则所有 Solaris 和 Windows 脚本中的 -javahome 选项都不起作用（**错误号** *4683029*）。

Message Queue 命令和实用程序使用 `javahome` 选项来指定要使用的备用 Java 2 兼容运行时。但是，备用 Java 运行时的路径名不能包含空格。以下是包含空格的路径示例。

Windows：`C:/jdk 1.4`

Solaris： `/work/java 1.4`

解决方法 请在不包含空格的位置或路径中安装 Java 运行时。

- `imqQueueBrowserMaxMessagesPerRetrieve` 属性指定客户端运行时环境在浏览队列的内容时一次检索到的最大消息数。请注意，客户端应用程序始终可以获取队列上的所有消息。因此，`imqQueueBrowserMaxMessagesPerRetrieve` 属性会影响如何对排队的消息进行分块，以便传送到客户端运行时环境（消息较少时用较大的块，或消息较多时用较小的块），但不会影响所浏览的全部消息。更改此属性的值可能会对性能产生影响，但不会导致客户端应用程序获取的数据增加或减少（**错误号** *6387631*）。

代理问题

以下问题将影响 Message Queue 代理。

- 在如何为循环传送配置代理方面一直存在一些误区。解决方案非常简单，并且可以对其进行配置。

  1. 将目的地属性 `maxNumActiveConsumers` 设置为 -1。这会打开循环传送功能。
  2. 将目的地属性 `consumerFlowLimit` 设置为 1。这会指定在向下一个使用方传送消息之前传送到某个使用方的消息数。对于不同的消息分块方式，请将此属性设置为所需的值。默认情况下，为每个使用方传送 100 个消息。

- 如果持久性存储库打开过多目的地，将无法访问代理。（**错误号** *4953354*）。

  解决方法 这种情况是由于代理达到了系统打开文件描述符限制所致。在 Solaris 和 Linux 上可使用 `ulimit` 命令来增加文件描述符限制。

- 目的地被销毁后，使用方将被孤立（**错误号** *5060787*）。

  目的地被销毁后，活动的使用方将被孤立。使用方孤立后，他们将不再接收消息（即使重新创建了目的地）。

  解决方法 此问题没有解决方法。

代理群集

以下问题会影响群集代理。

- 本发行版中只支持完全连接的代理群集。这意味着群集中的每个代理均必须与群集中的其他所有代理直接通信。如果使用 `imqbrokerd -cluster` 命令行变量连接代理，请务必小心以确保包含了群集中的所有代理。
- 使用 HADB 的代理无法处理大于 10 MB 的消息。（**错误** *6531734*）

- 如果客户端连接到高可用性代理，客户端运行时环境将尝试重新连接，直到成功时为止（无论将 `imqAddressListIterations` 设置为什么值）
- 与作为群集一部分的代理连接的客户端目前不能使用 `QueueBrowser` 来浏览位于该集群中远程代理上的队列。此客户端只能浏览位于直接连接的代理上的队列的内容。此客户端仍可以向任何队列发送消息，或使用来自集群中任何代理上的任何队列的消息，此限制只影响浏览。
- 在传统群集中，如果要将 4.1 代理和 3.x 代理放在一个群集中，必须为 4.1 代理设置属性 `imq.autocreate.queue.maxNumActiveConsumers=1`。否则，这些代理将无法建立群集连接。
- 在转换为高可用性群集时，可以使用 Message Queue 管理器实用程序 (imqdbmgr) 将现有的独立 HADB 持久性数据存储库转换为共享 HADB 存储库。这些命令如下所示。

  ```
  imqdbmgr upgrade hastore
  ```

  在下列情况下，您可以使用此实用程序。

  - 从 4.0 独立 HADB 存储库移到 4.1 共享 HADB 存储库。在这种情况下，代理将自动升级存储库。可随后运行 `imqdbmgr` 命令，转换升级的数据存储库以进行共享使用。
  - 从 4.1 独立 HADB 存储库移到共享 HADB 存储库。在这种情况下，您只需运行上面所示的 `imqdbmgr` 命令，转换数据存储库以进行共享使用。

  由于此命令仅支持 HADB 存储库转换，因此，无法使用它将基于文件的存储库或其他 JDBC 存储库转换为共享 HADB 存储库。如果以前运行 3.x 版本的 Message Queue，则必须创建一个 HADB 存储库，然后手动将数据迁移到该存储库中以便使用高可用性功能。
- 如果 HADB 存储库存储的消息超过 10,000 个，则使用 `imqdbmgr upgrade hastore` 命令转换为 HADB 存储库的操作可能会失败，并显示“设置的锁定太多”消息。（**错误号** *6588856*）。

  （**解决方法**）使用以下命令增加锁定数。

  ```
  hadbm set NumberOfLocks=<desiredNumber>
  ```

  有关其他信息，请参见 Sun Java System Application .Server 9.1 Enterprise Edition Troubleshooting Guide 中的 "HADB Problems"。
- 如果在一个事务中提交的远程消息超过 500 个，代理可能会返回错误“HADB-E-12815：表内存空间已用尽。”（**错误号** *6550483*）

  有关其他信息，请参见 Sun Java System Application .Server 9.1 Enterprise Edition Troubleshooting Guide 中的 "HADB Problems"。
- 在代理群集中，代理会对要传送到尚未启动的远程连接的消息进行排队（**错误号** *4951010*）。

  解决方法 一旦启动远程连接，这些消息将会由使用方接收。如果该使用方的连接关闭，这些消息将重新传送给另一个使用方。

- 如果在一个事务中使用多个来自远程代理的消息，则可能会在该代理中记录以下错误消息。该消息是无害的，可以将其忽略：

```
[26/Jul/2007:13:18:27 PDT] WARNING [B2117]:
Message acknowledgement failed from
mq://129.145.130.95:7677/?instName=a&brokerSessionUID=3209681167602264320:
    ackStatus = NOT_FOUND(404)\
    Reason = Update remote transaction state to COMMITED(6):
transaction 3534784765719091968 not found, the transaction
may have already been committed.
    AckType = MSG_CONSUMED
    MessageBrokerSession = 3209681167602264320
    TransactionID = 3534784765719091968
        SysMessageID = 8-129.145.130.95(95:fd:93:91:ec:a0)-33220-1185481094690
        ConsumerUID = 3534784765719133952\par

[26/Jul/2007:13:18:27 PDT] WARNING Notify commit transaction
[8-129.145.130.95(95:fd:93:91:ec:a0)-33220-1185481094690,
[consumer:3534784765719133952, type=NONE]]
TUID=3534784765719091968 got response:
com.sun.messaging.jmq.jmsserver.util.BrokerException:
 Update remote transaction state to COMMITED(6):
  transaction 3534784765719091968 not found, the transaction may have already
  been committed.:
com.sun.messaging.jmq.jmsserver.util.BrokerException: Update remote transaction
state to COMMITED(6): transaction 3534784765719091968 not found, the transaction
 may have already been committed.r
```

  如果 `imq.txn.reapLimit` 属性与一个事务中的远程消息数相比较小，当通知将事务中的后续消息提交到消息主代理时，将会记录此消息。（**错误** *6585449*）

  解决方法 要避免出现此消息，请增加 `imq.txn.reapLimit` 属性的值。

JMX 问题

在 Windows 平台上，事务管理器监视 MBean 的 `getTransactionInfo` 方法将返回具有错误事务创建时间的事务信息（**错误号** *6393359*）。

解决方法 请改用事务管理器监视 MBean 的 `getTransactionInfoByID` 方法。

SOAP 支持

您需要注意与 SOAP 支持有关的两个问题

- 从 Message Queue4.0 发行版开始，将不再为 SOAP 管理的对象提供支持。

- SOAP 开发依赖于以下几个文件：SUNWjaf、SUNWjmail、SUNWxsrt 和 SUNWjaxp。在 Message Queue 4.1 版中，仅当运行带有 JDK 1.6.0 或更高版本的 Message Queue 时，才能使用这些文件。

可再分发的文件

Sun Java System Message Queue 4.1 中包含以下一组文件，您可以使用这些文件，并以二进制格式自由分发它们：

- fscontext.jar
- imq.jar
- imqjmx.jar
- imqxm.jar
- jaas.jar
- jms.jar
- libmqcrt.so (HPUX)
- libmqcrt.so (UNIX)
- mqcrt1.dll (Windows)

此外，还可以再分发 LICENSE 和 COPYRIGHT 文件。

为残疾人士提供的辅助功能

欲获得自本介质发行以来所发布的辅助功能，请联系 Sun 索取有关 "Section 508" 法规符合性的产品评估文档，以便确定哪些版本最适合部署辅助功能解决方案。可通过以下网址获取应用程序的更新版本：

http://sun.com/software/javaenterprisesystem/get.html

有关 Sun 在辅助功能方面所做出的努力，请访问 http://sun.com/access。

如何报告问题和提供反馈

如果您在使用 Sun Java System Message Queue 期间遇到问题，请通过以下方式与 Sun 客户支持部门联系：

- 位于 http://www.sun.com/service/sunone/software 的 Sun 软件支持联机服务。

  此站点上有一些链接，通过这些链接可以访问知识库、联机支持中心和 Product Tracker，还可了解维护方案以及用于联系支持部门的电话号码。
- 随维护合同一起分发的电话号码。

为了更好地帮助您解决问题，请在联系支持部门时提供以下信息：

- 问题描述，包括问题出现时的情况及其对您的操作的影响。
- 计算机类型、操作系统版本和产品版本，包括可能影响问题的所有修补程序和其他软件。

- 用来再现该问题的详细步骤。
- 所有错误日志或核心转储。

Sun Java System 软件论坛

以下位置提供了一个 Sun Java System Message Queue 论坛：

http://swforum.sun.com/jive/forum.jspa?forumID=24

我们欢迎您的参与。

Java 技术论坛

您可能会对 Java 技术论坛中的 JMS 论坛感兴趣。

http://forum.java.sun.com

Sun 欢迎您提出意见

Sun 致力于提高其文档的质量，并十分乐意收到您的意见和建议。

要共享您的意见，请访问 http://docs.sun.com，然后单击“发送意见”(Send Comments)。在联机表单中提供文档标题和文件号码。文件号码包含七位或九位数字，可在书的标题页或在文档顶部找到该号码。例如，本书的标题为《Sun Java System Message Queue 4.1 发行说明》，文件号码为 820-3193。

提出意见时您还需要在表格中输入文档的英文文件号码和标题。本文档的英文文件号码是 819-7753，文档标题为《Sun Java System Message Queue 4.1 Release Notes》。

其他 Sun 资源

从以下 Internet 位置可以找到有用的 Sun Java System 信息：

- 文档

  http://docs.sun.com/prod/java.sys
- 专业服务

  http://www.sun.com/service/sunps/sunone
- 软件产品和服务

  http://www.sun.com/software
- 软件支持服务

http://www.sun.com/service/sunone/software

- 支持和知识库

  http://www.sun.com/service/support/software

- Sun 支持和培训服务

  http://training.sun.com

- 咨询和专业服务

  http://www.sun.com/service/sunps/sunone

- 开发者信息

  http://developers.sun.com

- Sun 开发者支持服务

  http://www.sun.com/developers/support

- 软件培训

  http://www.sun.com/software/training