



Sun Java™ System

Message Queue 3.5

管理指南

Service Pack 1

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

文件号码: 817-7211

版权所有 © 2004 Sun Microsystems, Inc., 4150 Network Circle, Santa Clara, California 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 <http://www.sun.com/patents> 中列出的一项或多项美国专利，以及在美国和其他国家 / 地区申请的一项或多项其他专利或待批专利。必须依据许可证条款使用。本分发软件可能包括由第三方开发的产品。

Sun、Sun Microsystems、Sun 徽标、Java、Solaris、Sun[tm] ONE、JDK、Java Naming and Directory Interface、Javadoc、JavaMail、JavaHelp、Java 咖啡杯徽标和 Sun[tm] ONE 徽标是 Sun Microsystems, Inc. 在美国和其他国家 / 地区的商标或注册商标。

所有 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 Sun Microsystems, Inc. 开发的体系结构。

UNIX 是由 X/Open Company, Ltd. 在美国和其他国家 / 地区独家许可的注册商标。

本产品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。严禁出口或转口到受美国禁运的国家 / 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。

目录

- 图目录 15**
- 表目录 17**
- 过程目录 21**
- 前言 23**
 - 本指南的读者 23
 - 本指南的结构 24
 - 约定 25
 - 文本约定 25
 - 目录变量约定 26
 - 其他文档资源 28
 - Message Queue 文档集 28
 - 联机帮助 29
 - JavaDoc 29
 - 示例客户机应用程序 29
 - Java 消息服务 (JMS) 规范 29
 - 相关的第三方 Web 站点参考 30
- 第 1 章 概述 31**
 - 什么是 Sun Java System Message Queue? 31
 - 产品版本 33
 - 平台版 33
 - 企业版 33
 - 企业消息传送系统 34
 - 企业消息传送系统的要求 34
 - 集中式消息传送与点对点消息传送 34
 - 消息传送系统概念 35
 - 消息 35
 - 消息服务体系结构 36

消息传送模型	36
JMS 规范	37
JMS 消息结构	37
JMS 编程模型	37
JMS 受管理对象	39
JMS/J2EE 编程：消息驱动 Bean	39
消息驱动 Bean	40
J2EE 应用服务器支持	41
JMS 消息传送相关问题	41
JMS 提供者无关	41
编程域	42
客户机标识符	43
可靠消息传送	43
确认 / 事务	44
持久性存储器	45
性能折衷	45
消息选择	46
消息顺序和优先级	46
第 2 章 Message Queue 消息传送系统	47
Message Queue 消息服务器	48
代理	48
连接服务	50
消息路由器	53
持久性管理器	57
安全性管理器	60
监视服务	63
物理目标	68
队列目标	68
主题目标	70
自动创建的（与管理员创建的相对）目标	70
临时目标	72
多代理群集（企业版）	73
多代理体系结构	73
在开发环境中使用群集	76
群集配置属性	76
Message Queue 客户机运行时	76
消息生成	77
消息使用	77
Message Queue 受管理对象	78
连接工厂受管理对象	79
目标受管理对象	80
在客户机启动时覆盖属性值	81

第 3 章 Message Queue 管理任务和工具	83
Message Queue 管理任务	83
开发环境	83
生产环境	84
设置操作	84
设置生产环境	84
维护操作	85
设置生产环境	85
Message Queue 管理工具	86
管理控制台	86
命令行实用程序概述	87
命令行语法	88
通用命令行选项	89
 第 4 章 管理控制台教程	 91
准备工作	92
启动管理控制台	92
启动管理控制台	92
获得帮助	93
显示管理控制台帮助信息	93
使用代理	95
启动代理	95
启动代理	95
添加代理	96
将代理添加到管理控制台	96
更改管理员密码	98
更改管理员密码	98
连接到代理	98
连接到代理	98
查看连接服务	99
查看可用连接服务	99
将物理目标添加到代理	100
将队列目标添加到代理	101
使用物理目标	101
查看物理目标的属性	102
清除目标中的消息	103
删除目标	103
获取有关主题目标的信息	103
使用对象存储	104
添加对象存储	104
添加文件系统对象存储	104
检查对象存储属性	107
显示对象存储属性	107

连接到对象存储	107
连接到对象存储	107
添加连接工厂被管理对象	107
将连接工厂添加到对象存储	108
添加目标被管理对象	109
将目标添加到对象存储	109
被管理对象的属性	111
查看或更新目标对象的属性	111
更新控制台信息	111
运行样例应用程序	112
运行 HelloWorldMessageJNDI 应用程序	112
第 5 章 启动与配置代理	115
配置文件	115
实例配置文件	116
合并属性值	116
属性命名语法	117
编辑实例配置文件	118
启动代理	123
imqbrokerd 命令的语法	123
启动示例	124
启动使用默认代理名称和配置的代理实例	124
启动具有企业版试用许可证的代理实例	124
启动具有插入持久性的已命名代理实例	124
imqbrokerd 选项概述	125
使用群集（企业版）	128
群集配置属性	128
连接代理	130
连接方法	130
将代理连接到群集	130
安全的交叉代理连接	130
配置群集内安全连接	130
管理群集中的代理	131
将代理添加到群集	131
将新代理添加到现有群集	131
重新启动群集中的代理	131
重新启动现有群集的现有代理	131
从群集中删除代理	132
从现有群集中删除代理	132
管理主管代理的配置更改记录	132
备份配置更改记录	132
备份配置更改记录	132
恢复配置更改记录	133

万一发生故障时恢复主管代理	133
日志记录	133
默认日志记录配置	133
日志消息格式	134
更改日志记录器配置	134
更改代理的日志记录器配置	134
更改输出通道	135
更改日志文件转移标准	136
第 6 章 代理和应用程序管理	137
命令行实用程序	138
imqcmd 命令语法	138
imqcmd 子命令	138
imqcmd 选项概述	140
使用 imqcmd 命令	141
imqcmd 用法示例	142
管理代理	142
显示代理信息	144
更新代理属性	144
控制代理状态	145
暂停和恢复代理	145
关闭并重新启动代理	146
显示代理度量依据	146
管理连接服务	147
列出连接服务	148
显示连接服务信息	149
更新连接服务属性	149
显示连接服务度量依据	150
暂停和恢复连接服务	150
获得连接信息	151
管理目标	152
创建目标	154
列出目标	156
显示目标信息	156
更新目标属性	157
显示目标度量依据	158
暂停和恢复目标	158
清除目标	159
销毁目标	159
压缩目标	159
监视目标的磁盘利用	160
回收未使用的目标磁盘空间	160
回收未使用的目标磁盘空间	161

管理长期订阅	161
管理事务	162

第 7 章 管理受管理对象 165

关于对象存储库	166
LDAP 服务器对象存储库	166
文件系统对象存储库	167
受管理对象	168
连接工厂受管理对象属性	168
目标受管理对象的属性	170
对象管理器实用程序 (imqobjmgr)	171
imqobjmgr 命令的语法	171
imqobjmgr 子命令	171
imqobjmgr 命令选项概述	171
所需的信息	173
使用命令文件	174
添加和删除受管理对象	176
添加连接工厂	176
添加主题或队列	177
删除受管理对象	179
获得信息	179
受管理对象列表	179
单个对象的相关信息	180
更新受管理对象	180

第 8 章 管理安全性 183

验证用户	184
使用文本文件用户信息库	184
创建用户信息库	184
用户管理器实用程序 (imqusermgr)	185
组	187
状态	187
用户名和密码的格式	188
填充和管理用户信息库	188
更改默认的管理员密码	189
使用 LDAP 服务器管理用户信息库	190
编辑配置文件以使用 LDAP 服务器	190
授权用户:	
访问控制属性文件	192
创建访问控制属性文件	193
访问规则语法	193
权限计算	194

连接访问控制	195
目标访问控制	196
目标自动创建访问控制	197
加密：使用基于 SSL 的服务（企业版）	198
设置通过 TCP/IP 的基于 SSL 的服务	198
设置基于 SSL 的连接服务	198
步骤 1：生成自签名证书	199
重新生成密钥对	200
步骤 2：在代理上启用基于 SSL 的服务	200
在代理中启用基于 SSL 的服务	200
步骤 3. 启动代理	201
步骤 4：配置并运行基于 SSL 的客户机	202
设置通过 HTTP 的 SSL 服务	203
使用密码文件	204
 第 9 章 分析和调整消息服务	205
关于性能	205
性能调整过程	205
性能方面	206
基准检验	206
基线使用模式	207
影响性能的因素	208
影响性能的应用程序设计因素	209
传送模式（持久性 / 非持久性消息）	210
使用事务	211
确认模式	212
长期与非长期订阅	212
使用选择器（消息过滤）	213
消息大小	214
消息主体类型	214
影响性能的消息服务因素	215
硬件	215
操作系统	216
Java 虚拟机 (JVM)	216
连接	216
消息服务器体系结构	218
代理限制和行为	219
数据存储性能	219
客户机运行时配置	219
监视消息服务器	220
监视工具	220
Message Queue 命令行实用程序 (mqcmd)	220
要使用 metrics 子命令，请执行下列操作：	222

Message Queue 代理日志文件	225
要使用日志文件来报告度量依据信息，请执行下列操作：	225
基于消息的监视 API	226
要设置基于消息的监视，请执行下列操作：	227
选择适当的监视工具	229
度量依据数据说明	230
JVM 度量依据	231
代理范围度量依据	231
连接服务度量依据	233
目标度量依据	234
性能问题疑难解答	237
问题：客户机无法建立连接	237
症状：	237
可能原因：	237
问题：连接的吞吐量太慢	241
症状：	241
可能原因：	241
问题：客户机无法创建消息生成方	242
症状：	242
可能原因：	242
问题：消息的生成过程发生延迟或变慢	243
症状：	243
可能原因：	243
问题：消息在消息服务器中堆积	246
症状：	246
可能原因：	246
问题：消息服务器吞吐量呈间歇性	249
症状：	249
可能原因：	250
问题：消息无法到达使用方	251
症状：	251
可能原因：	251
调整配置以提高性能	252
系统调整	252
Solaris 调整：CPU 使用、分页 / 交换 / 磁盘 I/O	252
Java 虚拟机调整	252
调整传输协议	253
调整基于文件的持久性存储库	256
代理调整	256
内存管理：增大代理在负荷下的稳定性	256
多使用方队列性能	257
客户机运行时消息流调整	258
消息流测量	258

消息流限制	258
附录 A Message Queue 数据的位置	261
Solaris	261
Linux	262
Windows	263
附录 B 设置插入的持久性	265
简介	265
插入支持 JDBC 的数据存储	266
插入支持 JDBC 的数据存储	266
JDBC 相关的代理配置属性	267
数据库管理器实用程序 (imqdbmgr)	270
imqdbmgr 命令的语法	271
imqdbmgr 子命令	271
imqdbmgr 命令选项概述	272
附录 C HTTP/HTTPS 支持（企业版）	273
HTTP/HTTPS 支持体系结构	273
实现 HTTP 支持	275
实现 HTTP 支持	275
步骤 1: 在 Web 服务器上部署 HTTP 隧道 Servlet	275
部署为 Jar 文件	275
部署为 Web 归档文件	276
步骤 2: 配置 httpjms 连接服务	276
激活 httpjms 连接服务	276
步骤 3: 配置 HTTP 连接	277
配置连接工厂	277
使用一个 Servlet 访问多个代理	278
使用 HTTP 代理	278
示例 1: 在 Sun Java System Web Server 上部署 HTTP 隧道 Servlet	279
部署为 Jar 文件	279
添加隧道 Servlet	279
配置隧道 Servlet 的虚拟路径 (Servlet URL)	280
在 Web 服务器启动时装入隧道 Servlet	280
禁用服务器访问日志	281
部署为 WAR 文件	281
将 HTTP 隧道 Servlet 部署为 WAR 文件	281
示例 2: 在 Sun Java System Application Server 7.0 上部署 HTTP 隧道 Servlet	282
使用部署工具	282
在 Application Server 7.0 环境下部署 HTTP 隧道 Servlet	282
修改 server.policy 文件	283

修改 Application Server 的 server.policy 文件	283
实现 HTTPS 支持	284
实现 HTTPS 支持	284
步骤 1: 为 HTTPS 隧道 Servlet 生成自签名证书	284
步骤 2: 在 Web 服务器上部署 HTTPS 隧道 Servlet	285
部署为 Jar 文件	285
部署为 Web 归档文件	286
步骤 3: 配置 httpsjms 连接服务	286
激活 httpsjms 连接服务	286
步骤 4: 配置 HTTPS 连接	287
配置 JSSE	288
配置 JSSE	288
输入根证书	288
配置连接工厂	288
使用一个 Servlet 访问多个代理	289
使用 HTTP 代理	289
示例 3: 在 Sun Java System Web Server 上部署 HTTPS 隧道 Servlet	290
部署为 Jar 文件	290
添加隧道 Servlet	290
配置隧道 Servlet 的虚拟路径 (Servlet URL)	291
在 Web 服务器启动时装入隧道 Servlet	292
禁用服务器访问日志	292
部署为 WAR 文件	292
修改 HTTPS 隧道 Servlet WAR 文件	292
将 HTTPS 隧道 Servlet 部署为 WAR 文件	293
示例 4: 在 Sun Java System Application Server 7.0 上部署 HTTPS 隧道 Servlet	294
使用部署工具	294
在 Application Server 7.0 环境下部署 HTTPS 隧道 Servlet	294
修改 server.policy 文件	295
修改 Application Server 的 server.policy 文件	295
附录 D 使用代理作为 Windows 服务	297
将代理作为 Windows 服务运行	297
服务管理器实用程序 (imqsvcadmin)	298
imqsvcadmin 命令语法	298
imqsvcadmin 子命令	298
imqsvcadmin 选项概述	298
删除代理服务	299
重新配置代理服务	299
使用替代 Java 运行时	300
查询代理服务	300
疑难解答	300
查看记录的服务错误事件	300

附录 E 技术说明	301
系统时钟设置	301
建议同步	301
避免将系统时钟设置为早于当前时间	302
OS 定义的文件描述符限制	302
确保持久性数据的安全	303
内置持久性存储库	303
插入的持久性存储库	303
 附录 F Message Queue 资源适配器	 305
 附录 G Message Queue 实现方案可选的 JMS 功能	 307
 附录 H Message Queue 接口的稳定性	 309
 词汇表	 313
 索引	 317

图目录

图 1-1	集中式消息传送与点对点消息传送	35
图 1-2	消息服务体系结构	36
图 1-3	JMS 编程对象	38
图 1-4	与 MDB 进行消息传送	40
图 2-1	Message Queue 系统的体系结构	47
图 2-2	代理服务组件	49
图 2-3	连接服务支持	51
图 2-4	持久性管理器支持	58
图 2-5	安全性管理器支持	61
图 2-6	监视服务支持	64
图 2-7	多代理（群集）体系结构	74
图 2-8	消息传送操作	77
图 2-9	将消息传送到 Message Queue 客户机运行时	78
图 3-1	本地和远程管理实用程序	87
图 5-1	代理配置文件	117
图 9-1	通过 Message Queue 服务传送消息	208
图 9-2	传送模式的性能影响	211
图 9-3	订阅类型的性能影响	213
图 9-4	消息大小的性能影响	214
图 9-5	传输协议速度	217
图 9-6	传输协议的性能影响	218
图 9-7	更改大小为 1K（1024 字节）的数据包的 inbufsz 的结果	255
图 9-8	更改大小为 1K（1024 字节）的数据包的 outbufsz 的结果	255
图 C-1	HTTP/HTTPS 支持体系结构	274

表目录

表 1	本书内容	24
表 2	文档约定	25
表 3	Message Queue 目录变量	26
表 4	Message Queue 文档集	28
表 1-1	JMS 编程对象	42
表 2-1	主要代理服务组件及功能	49
表 2-2	代理支持的连接服务	50
表 2-3	连接服务属性	52
表 2-4	消息路由器属性	56
表 2-5	持久性管理器属性	59
表 2-6	安全性管理器属性	62
表 2-7	日志种类	64
表 2-8	度量依据主题目标	65
表 2-9	监视服务属性	66
表 2-10	自动创建配置属性	70
表 2-11	目标属性	80
表 3-1	通用 Message Queue 命令行选项	89
表 5-1	代理实例配置属性	118
表 5-2	imqbrokerd 选项	125
表 5-3	群集配置属性	128
表 5-4	imqbrokerd 日志记录器选项及相应的属性	135
表 6-1	imqcmd 子命令	138
表 6-2	imqcmd 选项	140
表 6-3	用于管理代理的 imqcmd 子命令	143
表 6-4	由 imqcmd 更新的代理属性	144
表 6-5	用于管理连接服务的 imqcmd 子命令	147
表 6-6	代理支持的连接服务	148

表 6-7	由 <code>imgcmd</code> 更新的连接服务属性	149
表 6-8	用于管理连接服务的 <code>imgcmd</code> 子命令	151
表 6-9	用于管理目标的 <code>imgcmd</code> 子命令	152
表 6-10	目标属性	154
表 6-11	目标磁盘利用度量依据	160
表 6-12	用于管理长期订阅的 <code>imgcmd</code> 子命令	161
表 6-13	用于管理事务的 <code>imgcmd</code> 子命令	162
表 7-1	LDAP 对象存储库属性	166
表 7-2	文件系统对象存储库属性	167
表 7-3	连接工厂受管理对象属性	168
表 7-4	目标受管理对象的属性	170
表 7-5	<code>imgobjmgr</code> 子命令	171
表 7-6	<code>imgobjmgr</code> 选项	172
表 7-7	命名惯例示例	177
表 8-1	用户信息库中的初始条目	185
表 8-2	<code>imgusermgr</code> 子命令	186
表 8-3	<code>imgusermgr</code> 选项	186
表 8-4	用户名和密码不能包含的字符	188
表 8-5	LDAP 相关属性	190
表 8-6	访问规则的语法元素	194
表 8-7	目标访问控制规则的元素	196
表 8-8	密钥存储属性	199
表 8-9	密码文件中的密码	204
表 9-1	高可靠性和高性能方案比较	210
表 9-2	<code>imgcmd metrics</code> 子命令语法	221
表 9-3	<code>imgcmd metrics</code> 子命令选项	222
表 9-4	<code>imgcmd query</code> 子命令语法	224
表 9-5	度量依据主题目标	226
表 9-6	度量依据监视工具的正面和反面因素	229
表 9-7	JVM 度量依据	231
表 9-8	代理范围度量依据	231
表 9-9	连接服务度量依据	233
表 9-10	目标度量依据	235
表 A-1	Message Queue 数据在 Solaris 平台上的位置	261
表 A-2	Message Queue 数据在 Linux 平台上的位置	262
表 A-3	Message Queue 数据在 Windows 平台上的位置	263
表 B-1	JDBC 相关属性	267

表 B-2	imqdbmgr 子命令	271
表 B-3	imqdbmgr 选项	272
表 C-1	httpjms 连接服务属性	276
表 C-2	用于部署 HTTP 隧道 Servlet Jar 文件的 Servlet 参数	280
表 C-3	httpsjms 连接服务属性	287
表 C-4	用于部署 HTTPS 隧道 Servlet Jar 文件的 Servlet 参数	291
表 D-1	imqsvcadm 子命令	298
表 D-2	imqsvcadm 选项	299
表 G-1	可选的 JMS 功能	307
表 H-1	Message Queue 接口的稳定性	309
表 H-2	接口稳定性分类方案	311

过程目录

设置生产环境	84
设置生产环境	85
启动管理控制台	92
显示管理控制台帮助信息	93
启动代理	95
将代理添加到管理控制台	96
更改管理员密码	98
连接到代理	98
查看可用连接服务	99
将队列目标添加到代理	101
查看物理目标的属性	102
清除目标中的消息	103
删除目标	103
添加文件系统对象存储	104
显示对象存储属性	107
连接到对象存储	107
将连接工厂添加到对象存储	108
将目标添加到对象存储	109
查看或更新目标对象的属性	111
运行 HelloWorldMessageJNDI 应用程序	112
启动使用默认代理名称和配置的代理实例	124
启动具有企业版试用许可证的代理实例	124
启动具有插入持久性的已命名代理实例	124
将代理连接到群集	130
配置群集内安全连接	130
将新代理添加到现有群集	131
重新启动现有群集的现有代理	131

从现有群集中删除代理	132
备份配置更改记录	132
万一发生故障时恢复主管代理	133
更改代理的日志记录器配置	134
回收未使用的目标磁盘空间	161
编辑配置文件以使用 LDAP 服务器	190
设置基于 SSL 的连接服务	198
重新生成密钥对	200
在代理中启用基于 SSL 的服务	200
要使用 metrics 子命令，请执行下列操作：	222
要使用日志文件来报告度量依据信息，请执行下列操作：	225
要设置基于消息的监视，请执行下列操作：	227
插入支持 JDBC 的数据存储	266
实现 HTTP 支持	275
激活 httpjms 连接服务	276
添加隧道 Servlet	279
配置隧道 Servlet 的虚拟路径 (Servlet URL)	280
在 Web 服务器启动时装入隧道 Servlet	280
禁用服务器访问日志	281
将 HTTP 隧道 Servlet 部署为 WAR 文件	281
在 Application Server 7.0 环境下部署 HTTP 隧道 Servlet	282
修改 Application Server 的 server.policy 文件	283
实现 HTTPS 支持	284
激活 httpsjms 连接服务	286
配置 JSSE	288
添加隧道 Servlet	290
配置隧道 Servlet 的虚拟路径 (Servlet URL)	291
在 Web 服务器启动时装入隧道 Servlet	292
禁用服务器访问日志	292
修改 HTTPS 隧道 Servlet WAR 文件	292
将 HTTPS 隧道 Servlet 部署为 WAR 文件	293
在 Application Server 7.0 环境下部署 HTTPS 隧道 Servlet	294
修改 Application Server 的 server.policy 文件	295
查看记录的服务错误事件	300

前言

《Sun Java™ System Message Queue 3.5 SP1 管理指南》一书提供了对 Message Queue 消息传送系统执行管理任务所需的背景和信息。

本前言包含以下章节：

- [第 23 页的“本指南的读者”](#)
- [第 24 页的“本指南的结构”](#)
- [第 25 页的“约定”](#)
- [第 28 页的“其他文档资源”](#)

本指南的读者

本指南供需要执行 Message Queue 管理任务的系统管理员以及应用程序开发者使用。

Message Queue 管理员负责设置和管理 Message Queue 消息传送系统，特别是该系统的核心 Message Queue 消息服务器。本书并不要求读者掌握所有消息传送系统的知识或对其有一定了解。

本指南也供应用程序开发者使用，使其更好地了解如何优化应用程序，从而充分利用 Message Queue 消息传送系统的功能和灵活性。

本指南的结构

请从头到尾按部就班地阅读本指南。下表简要介绍了各章的内容：

表 1 本书内容	
章	说明
第 1 章 “概述”	对 Message Queue 消息传送系统及其术语进行了深入地概念综述。
第 2 章 “Message Queue 消息传送系统”	论述 Message Queue 消息传送系统，特别是 Message Queue 代理和 Message Queue 客户机运行时，它们协同提供消息传送服务。
第 3 章 “Message Queue 管理任务和工具”	论述 Message Queue 管理任务和管理工具，并介绍用于管理的命令行实用程序及其通用功能。
第 4 章 “管理控制台教程”	提供一个实用教程，借助此教程，您将了解管理控制台（Message Queue 消息服务器的图形界面）的操作方法。
第 5 章 “启动与配置代理”	介绍如何启动和配置 Message Queue 代理和代理群集。
第 6 章 “代理和应用程序管理”	介绍如何执行与管理 Message Queue 代理相关的（应用程序无关）任务，以及如何执行用于管理消息传送应用程序的任务。
第 7 章 “管理受管理对象”	介绍如何执行与创建和管理 Message Queue 受管理对象相关的任务。
第 8 章 “管理安全性”	介绍如何执行与应用程序相关的安全任务，例如管理验证、授权和加密。
第 9 章 “分析和调整消息服务”	说明用于监视和分析消息服务器性能的有关技术，并介绍如何调整消息服务器以优化其性能。
附录 A “Message Queue 数据的位置”	说明不同类别的 Message Queue 数据的位置。
附录 B “设置插入的持久性”	介绍如何设置 Message Queue 以使用 JDBC 兼容数据库执行持久性功能。
附录 C “HTTP/HTTPS 支持（企业版）”	介绍如何在消息传送客户机和 Message Queue 消息服务器之间建立 HTTP 连接服务。
附录 D “使用代理作为 Windows 服务”	介绍如何使用 Message Queue 服务管理实用程序 (imgsvcadmin) 安装、查询和删除代理（作为 Windows 服务运行）。
附录 E “技术说明”	提供与本书中涉及的主题相关的大量专业技术说明，但与针对 Message Queue 的管理工作无关。
附录 F “Message Queue 资源适配器”	说明 Message Queue 资源适配器的定义、如何部署它，以及如何配置和使用它。
附录 G “Message Queue 实现方案可选的 JMS 功能”	说明 Message Queue 产品如何处理 JMS 规范中列出的各项内容（JMS 提供者可以选择实现这些项）。

表 1 本书内容 (续)

章	说明
附录 H “Message Queue 接口的稳定性”	说明不同 Message Queue 接口的稳定性。
“词汇表”	定义 Message Queue 文档中使用的术语。

约定

本节介绍本文档中使用的有关约定。

文本约定

表 2 文档约定

格式	说明
斜体文本	斜体文本代表占位符。斜体文本可用相应地子句或值替换。斜体文本还用来表示文档标题、强调、引入的的单词或短语。
等宽文本	等宽文本表示实例代码、在命令行输入的命令、目录、文件、路径名、错误消息文本、类名称、方法名称（包括签名中的所有元素）、软件包名称、保留字和 URL。
[]	方括号用来表示命令行语法语句中的可选值。
全部大写文本	全部大写的文本表示文件系统类型（GIF、TXT、HTML 等）、环境变量 (IMQ_HOME) 或首字母缩略词（Message Queue、JSP）。
键名 + 键名	用加号连接的一组同时按下的键：Ctrl+A 表示同时按下这两个键。
键名 - 键名	用连字符连接的一组依次按下的键：Esc-S 表示先按 Esc 键，然后释放它，再按 S 键。

目录变量约定

Message Queue 使用三种目录变量；其设置因平台而异。表 3 介绍了这些变量并概述了如何在 Solaris™、Windows 和 Linux 平台上使用这些变量。

表 3 Message Queue 目录变量

变量	说明
IMQ_HOME	<p>Message Queue 文档中通常使用此变量，表示 Message Queue 根目录（根安装目录）：</p> <ul style="list-style-type: none">• Solaris 平台上，不存在 Message Queue 根安装目录。因此，Message Queue 文档中不使用 IMQ_HOME 表示 Solaris 平台上的文件位置。• Solaris 平台上，对于 Sun Java System Application Server，Message Queue 根安装目录为 Application Server 根目录下的 /imq。• Windows 平台上，Message Queue 根安装目录由 Message Queue 安装程序设置（默认情况下为 C:\Program Files\Sun\MessageQueue3）。• Windows 平台上，对于 Sun Java System Application Server，Message Queue 根安装目录为 Application Server 根目录下的 /imq。• Linux 平台上，不存在 Message Queue 根安装目录。因此，Message Queue 文档中不使用 IMQ_HOME 表示 Linux 平台上的文件位置。
IMQ_VARHOME	<p>这是 /var 目录，其中存储了 Message Queue 临时或动态创建的配置和数据文件。可设置为指向任何目录的环境变量。</p> <ul style="list-style-type: none">• Solaris 平台上，IMQ_VARHOME 默认为 /var/imq 目录。• Solaris 平台上，对于 Sun Java System Application Server 测试版，IMQ_VARHOME 默认为 IMQ_HOME/var 目录。• Windows 平台上，IMQ_VARHOME 默认为 IMQ_HOME\var 目录。• Windows 平台上，对于 Sun Java System Application Server，IMQ_VARHOME 默认为 IMQ_HOME\var 目录。• Linux 平台上，IMQ_VARHOME 默认为 /var/opt/imq 目录。

表 3 Message Queue 目录变量 (续)

变量	说明
IMQ_JAVAHOME	<p>这是一个环境变量，指向 Message Queue 可执行文件所需的 Java™ 运行时 (JRE) 的位置：</p> <ul style="list-style-type: none">• Solaris 平台上，IMQ_JAVAHOME 默认为 /usr/j2se/jre 目录，但是用户可以选择性地将其值设置为所需的 JRE 驻留的位置。• Windows 平台上，IMQ_JAVAHOME 默认为 IMQ_HOME\jre，但是用户可以选择性地将其值设置为所需的 JRE 驻留的位置。• Linux 平台上，Message Queue 首先在 /usr/java/j2sdkVersion 目录中查找 Java 运行时，然后再从 /usr/java/j2reVersion 目录查找，但是用户可以选择性地将 IMQ_JAVAHOME 的值设置为所需的 JRE 驻留的位置。

在本指南中，显示 IMQ_HOME、IMQ_VARHOME 和 IMQ_JAVAHOME 时，*不使用*特定平台的环境变量表示法或语法（例如，在 UNIX® 平台上为 \$IMQ_HOME）。路径名通常采用 UNIX 目录分隔符表示法 (/)。

其他文档资源

除本指南外， Message Queue 还提供了其他文档资源。

Message Queue 文档集

表 4 按照通常的使用顺序列出了 Message Queue 文档集中的文档。 -

表 4 Message Queue 文档集		
文档	读者	说明
《Message Queue 安装指南》	开发者和管理员	介绍如何在 Solaris、Linux 和 Windows 平台上安装 Message Queue 软件。
《Message Queue 发行说明》	开发者和管理员	包含对新功能、限制、已知错误以及技术说明的介绍。
《Message Queue 管理指南》	管理员，也推荐开发者阅读	提供使用 Message Queue 管理工具执行管理任务时所需的背景和信息。
《Message Queue Java Client Developer's Guide》	开发者	为使用 JMS 和 SOAP/JAXM 规范 Message Queue 实现方案开发 Java 客户机程序的人员提供快速入门教程和编程信息。
《Message Queue C Client Developer's Guide》	开发者	为使用 Message Queue 消息服务的 C 接口 (C-API) 开发 C 客户机程序的人员提供编程和参考文档。

联机帮助

Message Queue 包含用于执行 Message Queue 消息服务的管理任务的命令行实用程序。要访问这些实用程序的联机帮助，请参见第 89 页的“通用命令行选项”。

Message Queue 还包含图形用户界面 (GUI) 管理工具，即管理控制台 (imqadmin)。管理控制台包含上下文有关联机帮助。

JavaDoc

JavaDoc 格式的 Message Queue Java 客户机 API（包含 JMS API）文档所在的目录取决于操作系统（请参见附录 A “Message Queue 数据的位置”）。

此文档可以在任何 HTML 浏览器中浏览，如 Netscape 或 Internet Explorer。它包括标准 JMS API 文档以及 Message Queue 受管理对象的 Message Queue 特定 API（请参见《Message Queue Java Client Developer's Guide》的第 3 章），这些对消息传送应用程序的开发者很有帮助。

示例客户机应用程序

许多示例应用程序提供了样例客户机应用程序代码，这些示例应用程序所在的目录取决于操作系统（请参见附录 A “Message Queue 数据的位置”）。

请参见位于该目录及其每个子目录中的 README 文件。

Java 消息服务 (JMS) 规范

可以在以下位置查找 JMS 规范：

<http://java.sun.com/products/jms/docs.html>

规范包含样例客户机代码。

相关的第三方 Web 站点参考

本文档中所引用的第三方 URL 提供了附加的相关信息。

注意

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。

本章提供了 Sun Java™ System Message Queue 的入门知识，适合于管理员和程序员阅读。

什么是 Sun Java System Message Queue?

Message Queue 产品为分布式应用程序进行可靠的异步消息传送提供了基于标准的解决方案。Message Queue 也是实现了 Java™ Message Service (JMS) 开放标准的企业消息传送系统：事实上它是 JMS 参考实现方案。然而，Message Queue 也是具有企业强度特点的全功能 JMS 提供者。

JMS 规范用于说明一套消息传送语义和行为，以及应用编程接口 (API)，它们为 Java 语言应用程序提供了在分布式环境中创建、发送、接收和读取消息的通用方法（请参见第 37 页的“JMS 编程模型”）。除了支持 Java 消息传送应用程序外，Message Queue 还提供了 Message Queue 服务的 C 语言接口 (Message Queue C-API)。

使用 Sun Java System Message Queue 软件，在不同平台和操作系统上运行的进程可以连接到一个通用的 Message Queue 消息服务（请参见第 36 页的“消息服务体系结构”），以发送和接收信息。这样，应用程序开发者就可以将精力集中在应用程序的业务逻辑上，而不是集中在“应用程序如何在网络中进行可靠的通信”这样的细枝末节上。

Message Queue 具有超出 JMS 规范最低要求的功能，包括：

集中式管理。 提供命令行和 GUI 两种工具来管理 Message Queue 服务并管理应用程序相关实体，如目标、事务、长期订阅和安全性。Message Queue 还支持对 Message Queue 服务进行远程监视。

可伸缩的消息服务。 您可以在以串连模式（多代理群集）工作的大量 Message Queue 消息服务器组件（代理）之间平衡负荷，从而为不断增加的 Message Queue 客户机（组件或应用程序）提供服务。

客户机连接故障转移。 自动恢复已经与 Message Queue 消息服务器断开的客户机连接。

可调整的性能。 在可以降低传送可靠性的情况下，提高 Message Queue 服务的性能。

多种传输协议。 支持 Message Queue 客户机使用多种不同传输协议（包括 TCP 和 HTTP）与 Message Queue 消息服务器进行通信的功能，并支持使用安全 (SSL) 连接。

JNDI 支持。 支持将 Java 命名和目录接口 (JNDI) 的基于文件的实现方案和 LDAP 实现方案用作对象存储和用户信息库。

SOAP 消息传送支持。 支持通过 JMS 消息传送来创建和传送 SOAP 消息（即符合简单对象访问协议 (SOAP) 规范的消息）。SOAP 允许在分布式环境中的点之间交换结构化 XML 数据。有关详细信息，请参见 *《Message Queue Java Client Developer's Guide》*。

有关相关 JMS 兼容问题的文档，请参见[附录 G “Message Queue 实现方案可选的 JMS 功能”](#)。

产品版本

Sun Java System 可在两个版本中找到 Message Queue: 平台版和企业版, 每个版本对应于不同的功能集和许可功能, 如下所述。(有关对 Message Queue 进行版本升级的说明, 请参见《Message Queue 安装指南》。)

平台版

此版本可以从 Sun 的 Web 站点免费下载, 并且还附带有 Sun Java System Application Server 平台。平台版对 Message Queue 消息服务器所支持的客户机连接的数量没有限制。它附带两个许可证, 如下所述:

- **基本许可证。**此许可证提供基本的 JMS 支持 (它是 JMS 的全权提供者), 但是不具有某些企业功能, 如负荷平衡 (多代理消息服务)、HTTP/HTTPS 连接、安全连接服务、可伸缩连接功能、客户机连接故障转移、多使用方队列传送、基于消息远程监视以及 C-API 支持。此许可证没有期限限制, 因此可用于要求不太严格的生产环境。
- **90 天试用企业许可证。**此许可证具有基本许可证所没有的全部企业功能, 如支持多代理消息服务、HTTP/HTTPS 连接、安全连接服务、可伸缩连接功能、客户机连接故障转移、多使用方队列传送、基于消息远程监视以及 C-API 支持。但是, 软件许可证的期限限制为 90 天, 因此, 此许可证适用于评估本产品企业版所提供的企业功能 (请参见“[企业版](#)”)。

注意

可通过启动 Message Queue 消息服务器 (一个 Message Queue 代理实例) 来启用 90 天试用许可证, 如第 124 页的“[启动具有企业版试用许可证的代理实例](#)”中所述。

企业版

此版本用于在生产环境中部署和运行消息传送应用程序。它支持多代理消息服务、HTTP/HTTPS 连接、安全连接服务、可伸缩连接功能、客户机连接故障转移、多使用方队列传送、基于消息远程监视和 C-API 支持。您还可以使用企业版进行开发、调试, 并可以装入测试消息传送应用程序和组件。企业版对许可证没有期限限制, 对多代理消息服务中的代理数量也没有限制, 但对使用的 CPU 数量有限制。

企业消息传送系统

企业消息传送系统使独立的分布式应用程序或应用程序组件可以通过消息进行交互。这些组件（无论是在相同的主机、相同的网络上运行，还是通过 Internet 松散地连接在一起）均使用消息传送来传递数据以协调各自的功能。

企业消息传送系统的要求

企业应用程序系统一般都包括大量的分布式组件，这些组件在全天候的关键任务操作中交换数以万计的消息。要支持这样的系统，一个企业消息传送系统一般必须满足以下要求：

可靠的传送。 从一个组件向另一个组件传送的消息不能由于网络或系统故障而丢失。这就意味着系统必须能够保证成功地传送消息。

异步传送。 为了使大量组件能够同时交换消息，并支持高密度吞吐量，消息的发送就不能取决于使用方是否可以立即接收消息。如果某个使用方正忙或处于脱机状态，系统必须允许进行这样的操作：先将消息发送出去，然后当该使用方准备就绪时再接收消息。这就是所说的“异步消息传送”，也常称为“存储并转发”消息传送。

安全性。 消息传送系统必须支持基本的安全功能：用户验证、消息及资源的访问授权和摘要式加密。

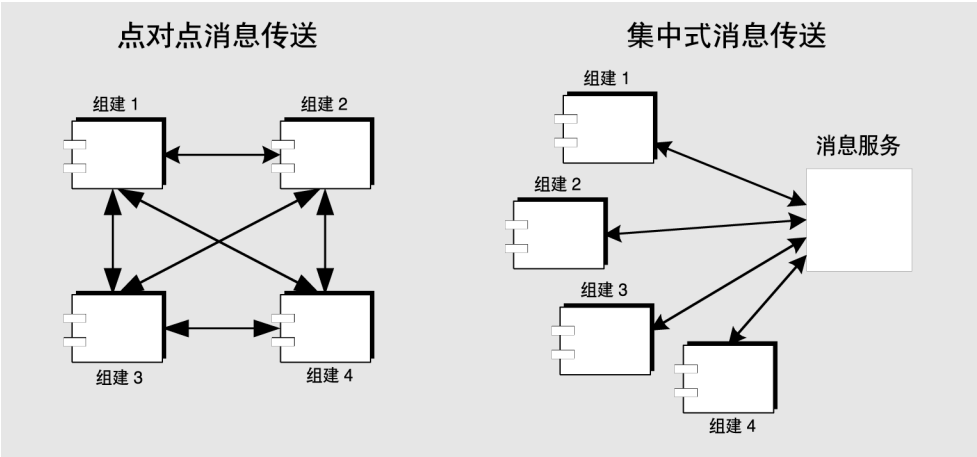
可伸缩性。 消息传送系统必须能够在不降低系统性能或消息吞吐量的前提下容纳不断增长的负荷，即用户数量和消息数量的增加。当业务和应用程序扩展时，这将成为一个十分重要的要求。

易管理性。 消息传送系统必须提供用于监视和管理消息传送以及优化系统资源的工具。这些工具有助于衡量和维护系统的可靠性、安全性和性能。

集中式消息传送与点对点消息传送

传统的点对点消息传送系统很难满足企业消息传送系统的要求，如图 1-1 所示。

图 1-1 集中式消息传送与点对点消息传送



在点对点系统中，每一个消息传送组件与所有其他组件都保持连接。这些连接可以实现快速、安全和可靠的传送，但是支持可靠性和安全性的代码必须驻留在每一个组件中。随着组件不断添加到系统中，连接的数量将成指数级增加。这使得异步消息传送和可伸缩性很难实现。集中式管理也变得问题重重。

企业消息传送首选的解决方案就是采用集中式消息传送系统，如图 1-1 中所示。在该解决方案中，每个消息传送组件只需与一个中央消息服务保持连接。消息服务提供组件之间的消息路由和传送，并负责可靠传送及安全。组件通过定义完善的程序接口与消息服务进行交互。当系统中添加组件时，连接的数量只会线性增加，这样就可以轻松地通过调整消息服务来调整系统。另外，中央消息服务为系统提供集中式管理。

消息传送系统概念

以下几个基本概念构成了企业消息传送系统的基础，包括：消息、消息服务体系结构和消息传送模型。

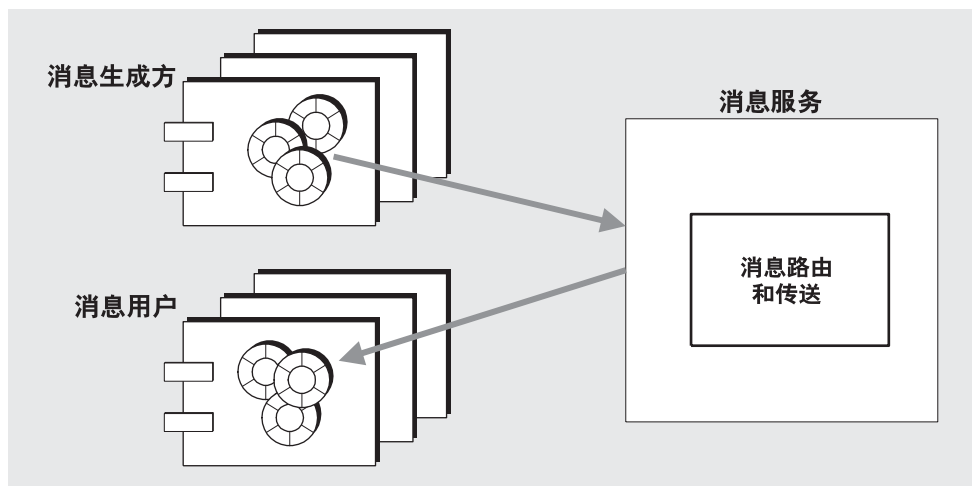
消息

消息由采用某种格式的数据（消息主体）和说明消息属性或特征（例如目标、有效期或其他由消息传送系统确定的属性）的元数据（消息标题）构成。

消息服务体系结构

图 1-2 所示为消息传送系统的基本体系结构。它包括消息生成方和消息使用方，它们利用通用消息服务来交换消息。同一消息传送组件（或应用程序）中可以驻留任意数量的消息生成方和消息使用方。消息生成方将消息发送至消息服务。接下来，消息服务使用消息路由和传送组件将消息传送至一个或多个已注册对此消息的请求的消息使用方。消息路由和传送组件负责保证将消息传送至所有相应的使用方。

图 1-2 消息服务体系结构



消息传送模型

生成方和使用方之间存在多种关系：一对一、一对多和多对多。例如，您可能会有以下类型的消息：

- 一个生成方至一个使用方
- 一个生成方至多个使用方
- 多个生成方至一个使用方
- 多个生成方至多个使用方

这些关系一般可简化为两个消息传送模型：**点对点消息传送**和**发布/订阅消息传送**。点对点传送模型主要处理由一个特定生成方创建并由一个特定使用方接收的消息。发布/订阅传送模型主要处理由任意数量的生成方创建并由任意数量的使用方接收的消息。这两种消息传送模型可以组合起来使用。

由于历史原因，消息传送系统支持这两种消息传送模型的各种组合方式。Java 消息服务 (JMS) 规范为 Java 编程创建了利用 API 进行消息传送的标准语义。它既支持点对点消息传送模型，也支持发布 / 订阅消息传送模型（请参见第 42 页的“编程域”）。

JMS 规范

JMS 规范规定了一套管理消息传送的规则和语义，包括编程模型、消息结构和 API。因为 Message Queue 提供 JMS 的实现方案，所以 JMS 概念是理解 Message Queue 消息传送系统如何工作的基础。本简介说明了理解本书其余各章所需的概念和术语。

JMS 消息结构

JMS 消息由三部分组成：标题、属性和主体。

标题 标题指定了消息的 JMS 属性：目标、是否持久、有效期及优先级。这些属性控制了消息传送系统传送消息的方式。

属性 属性（可以看作标题的扩充）为可选项，应用程序可以使用属性提供的值根据各种选择标准来过滤消息。属性为可选项。

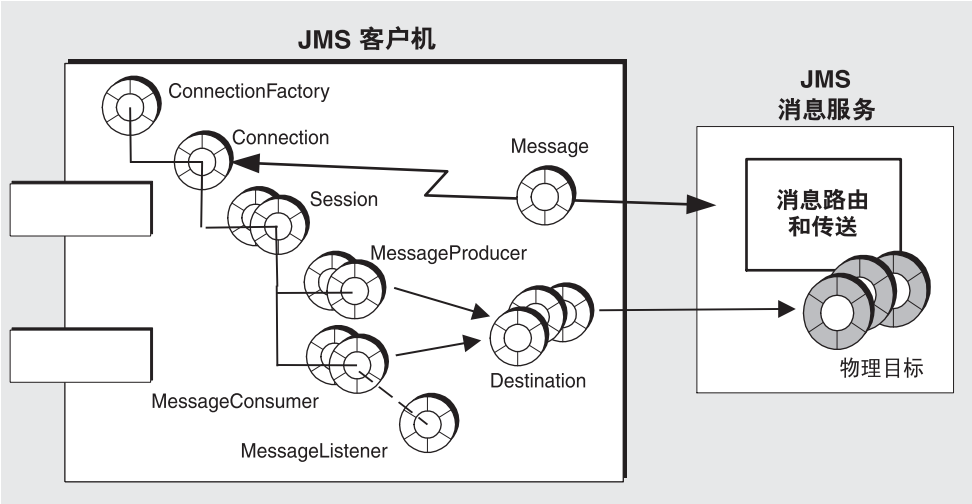
消息主体 消息主体包括要交换的实际数据。JMS 支持六种主体类型。

JMS 编程模型

在 JMS 编程模型中，JMS 客户机（组件或应用程序）通过 JMS 消息服务交换消息。消息生成方将消息发送至消息服务，消息使用方则从消息服务接收这些消息。这些消息传送操作是使用一组实现 JMS 应用编程接口 (API) 的对象（由 JMS 提供者提供）来执行的。

本节将介绍实现 JMS API 的对象和用于设置进行消息传送的 JMS 客户机的对象。有关详细信息，请参见《Message Queue Java Client Developer's Guide》。图 1-3 所示为用于编写消息传送的 JMS 对象。

图 1-3 JMS 编程对象



在 JMS 编程模型中，JMS 客户机使用 `ConnectionFactory` 对象创建一个连接，向消息服务发送消息以及从消息服务接收消息均是通过此连接来进行。`Connection` 是客户机与消息服务的活动连接。创建连接时，将分配通信资源以及验证客户机。这是一个相当重要的对象，大多数客户机均使用一个连接来进行所有的消息传送。

连接用于创建会话。`Session` 是一个用于生成和使用消息的单线程上下文。它用于创建发送和接收消息的生成方和使用方，并为所发送的消息定义发送顺序。会话通过大量确认选项或通过事务来支持可靠传送。

客户机使用 `MessageProducer` 向指定的物理目标（在 API 中表示为目标身份对象）发送消息。消息生成方可指定一个默认传送模式（持久性消息与非持久性消息）、优先级和有效期值，以控制生成方向物理目标发送的所有消息。

同样，客户机使用 `MessageConsumer` 对象从指定的物理目标（在 API 中表示为目标对象）接收消息。消息使用方可使用消息选择器，借助它，消息服务可以只向消息使用方发送与选择标准匹配的那些消息。

消息使用方可以支持同步或异步消息接收。异步使用可通过向使用方注册 `MessageListener` 来实现。当会话线程调用 `MessageListener` 对象的 `onMessage()` 方法时，客户机将使用消息。

JMS 受管理对象

JMS 规范通过指定封装有提供者特有配置信息的 *受管理对象*，使得独立于提供者的客户机更易于访问这些对象。

在第 37 页的“JMS 编程模型”介绍的对象中，有两个对象取决于 JMS 提供者如何实现 JMS 消息服务。连接工厂对象取决于提供者传送消息时使用的基本协议和机制，而目标对象则取决于提供者使用的特有命名惯例和物理目标容量。

通常，这些“提供者特有”属性使得 JMS 客户机代码取决于特定的 JMS 实现方案。不过，JMS 规范要求将提供者特有的实现方案和配置信息封装在连接工厂和目标对象中，可以通过非提供者特有的标准方式对这些对象进行访问。

受管理对象由管理员创建和配置，并存储在命名服务中，由客户机通过标准“Java 命名和目录服务”（JNDI）查找代码来进行访问。以这种方式使用受管理对象可使客户机代码与提供者无关。

两种类型的受管理对象（即连接工厂和目标对象）都可以封装提供者特有的信息，但它们在客户机中的使用情况明显不同。连接工厂用于创建与消息服务器的连接，而目标对象用于标识物理目标。

JMS/J2EE 编程：消息驱动 Bean

除在第 37 页的“JMS 编程模型”中介绍的通用 JMS 客户机编程模型外，还有专门在 Java 2 Platform Enterprise Edition（J2EE 平台）应用程序上下文中使用的 JMS。这些专用的 JMS 客户机称为 *消息驱动 Bean*，是 EJB 2.0 规范 (<http://java.sun.com/products/ejb/docs.html>) 中指定的企业 JavaBean (EJB) 系列组件之一。

之所以需要消息驱动 bean，是因为其他 EJB 组件（会话 bean 和实体 bean）只能被同步调用。这些 EJB 组件只能通过标准 EJB 接口来访问，因此不具备异步接收消息的机制。

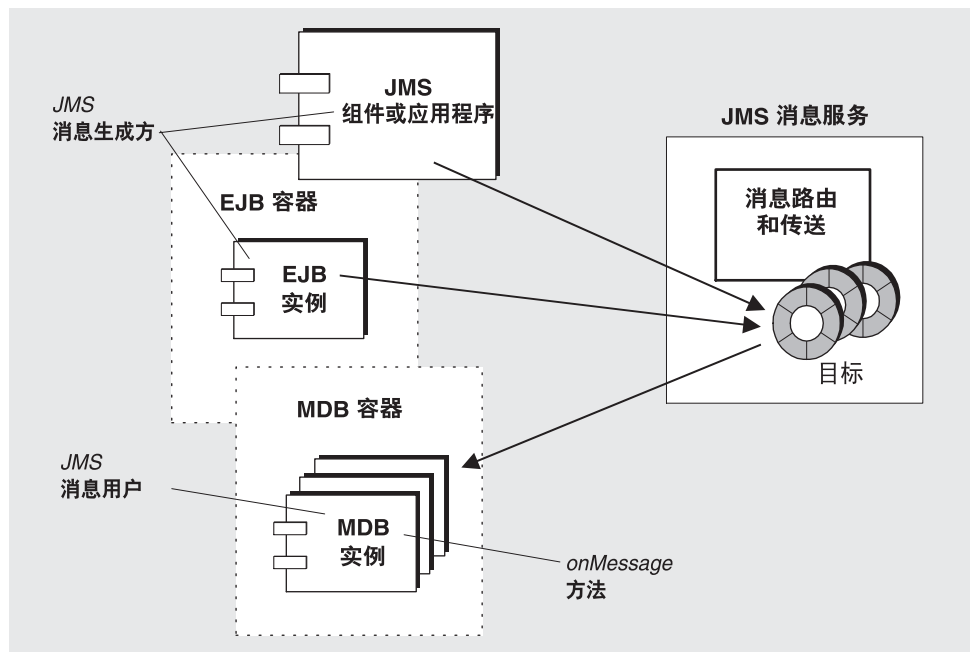
但是，异步消息传送是许多企业应用程序的一项必不可少的要求。这些应用程序大多需要服务器端组件能够互相通信和响应，而不占用服务器资源。因此就需要有这样一种 EJB 组件：不需要紧密耦合到消息生成方即可接收和使用消息。对于服务器端组件必须响应应用程序事件的任何应用程序，均必须具有上述功能。在企业应用程序中，此功能还必须在负荷增加时具有可伸缩性。

消息驱动 Bean

消息驱动 bean (MDB) 是由专用的 EJB 容器（为所支持的组件提供分布式服务的软件环境）支持的专用 EJB 组件。

消息驱动 Bean MDB 是实现 JMS `MessageListener` 接口的 JMS 消息使用方。当 MDB 容器接收消息时，`onMessage` 方法（由 MDB 开发者编写）将被调用。`onMessage()` 方法使用消息的方式与标准 `MessageListener` 对象的 `onMessage()` 方法使用消息的方式一样。不能以调用其他 EJB 组件的方法来远程调用 MDB 的方法，因此不存在与之关联的主接口或远程接口。MDB 可使用来自单一目标的消息。如图 1-4 所示，独立 JMS 应用程序、JMS 组件、EJB 组件或 Web 组件均可生成消息。

图 1-4 与 MDB 进行消息传送



MDB 容器 MDB 由专用 EJB 容器支持，负责创建并设置 MDB 实例，以进行消息的异步使用。这包括建立与消息服务的连接（包括验证）、创建与给定目标关联的会话池、管理在会话池和关联 MDB 实例之间接收到的消息的分发。由于容器控制了 MDB 实例的有效期，因此它通过管理 MDB 实例池来容纳外来的消息负荷。

与 MDB 关联的是一个部署描述符，它指定了容器设置消息使用时使用的受管理对象的 JNDI 查找名称：连接工厂和目标。部署描述符还可能包括可由部署工具用于配置容器的其他信息。每个这样的容器只支持一个 MDB 的实例。

J2EE 应用服务器支持

在 J2EE 体系结构中（请参见位于 <http://java.sun.com/j2ee/download.html#platformspec> 的 J2EE 平台规范），EJB 容器由 J2EE 应用服务器托管。应用程序服务器为各种容器提供所需的资源：事务管理器、持久性管理器、命名服务以及 JMS 提供者（对于消息传送和 MDB）。

在 Sun Java System Application Server 中，JMS 消息传送资源由 Sun Java System Message Queue 提供：

- 对于 Sun Java System Application Server 7.0，Message Queue 消息传送系统被作为它的本地 JMS 提供者集成到应用程序服务器中。
- 对于 Sun J2EE 1.4 Application Server，Message Queue 被作为嵌入式 JMS 资源适配器插入到应用程序服务器中（请参见附录 F “Message Queue 资源适配器”）。
- 对于应用程序服务器的后续发行版本，将利用标准资源适配器部署和配置方法将 Message Queue 插入到应用程序服务器中。

JMS 消息传送相关问题

本章介绍了许多影响 Message Queue 消息服务管理的 JMS 编程相关问题，着重介绍 Message Queue 管理员需要的概念和术语。

JMS 提供者无关

JMS 指定受管理对象的用途（请参见第 39 页的“JMS 受管理对象”）以支持可移植到其他 JMS 提供者的客户机应用程序的开发。受管理对象允许 JMS 客户机使用逻辑名称查找和引用提供者特定的对象。这样，客户机代码无需知道提供者使用的特有命名语法、寻址语法或可配置属性。从而使代码与提供者无关。

受管理对象是 Message Queue 系统对象，由 Message Queue 管理员创建并配置。这些对象存放在 JNDI 目录服务中，JMS 客户机使用 JNDI 查找来访问这些对象。

Message Queue 受管理对象也可以由客户机对其进行实例化，而不是在 JNDI 目录服务中查找。但这样做的缺点是要求应用程序开发者使用提供者特定的 API，并降低了 Message Queue 管理员成功控制和管理 Message Queue 消息服务器的能力。

有关受管理对象的详细信息，请参见第 78 页的“Message Queue 受管理对象”。

编程域

JMS 支持两种截然不同的消息传送模型：点对点模型和发布 / 订阅模型。

点对点（队列目标） 消息从一个生成方传送至一个使用方。在此传送模型中，目标是一个**队列**。消息首先被传送至队列目标，然后根据队列传送策略，从该队列（请参见第 68 页的“队列目标”）将消息传送至向此队列进行注册的某一个使用方，一次只传送一条消息。可以向队列目标发送消息的生成方的数量没有限制，但每条消息只能发送至、并由一个使用方成功使用。如果没有已经向队列目标注册的使用方，队列将保留它收到的消息，并在某个使用方向该队列进行注册时将消息传送给该使用方。

发布 / 订阅（主题目标） 消息从一个生成方传送至任意数量的使用方。在此传送模型中，目标是一个**主题**。消息首先被传送至主题目标，然后传送至**所有已订阅**此主题的活动使用方。可以向主题目标发送消息的生成方的数量没有限制，并且每个消息可以发送至任意数量的订阅使用方。主题目标也支持**长期订阅**的概念。长期订阅表示使用方已向主题目标进行注册，但在消息传送时此使用方可以处于非活动状态。当此使用方再次处于活动状态时，它将接收此信息。如果没有已经向主题目标注册的使用方，主题不保留其接收到的消息，除非有非活动使用方注册了长期订阅。

这两种消息传送模型使用表示不同编程域的 API 对象（其语义稍有不同）进行处理，如表 1-1 所示。

表 1-1 JMS 编程对象

基本类型 (统一域)	点对点域	发布 / 订阅域
Destination (Queue 或 Topic) ¹	Queue	Topic
ConnectionFactory	QueueConnectionFactory	TopicConnectionFactory
Connection	QueueConnection	TopicConnection
Session	QueueSession	TopicSession
MessageProducer	QueueSender	TopicPublisher
MessageConsumer	QueueReceiver	TopicSubscriber

1. 取决于编程方法，您可以指定特定的目标类型。

可以使用表 1-1 第一列中列出的统一域对象编写点对点 and 发布 / 订阅消息传送。这是首选方法。然而，为了符合早期的 JMS 1.02b 规范，可以使用点对点域对象编写点对点消息传送，使用发布 / 订阅域对象编制发布 / 订阅消息传送。

客户机标识符

JMS 提供者必须支持 *客户机标识符* 的概念，客户机标识符将 JMS 客户机至消息服务的连接与消息服务代表客户机维护的状态信息关联起来。按照定义，客户标识符是唯一的，并且每次只应用到一个用户。客户机标识符与长期订阅名称（请参见第 42 页的“发布 / 订阅（主题目标）”）结合使用，确保每个长期订阅只对应一个用户。

JMS 规范允许客户机通过 API 方法调用来设置客户机标识符，但最好使用连接工厂受管理对象（请参见第 39 页的“JMS 受管理对象”）从管理级别来设置它。但如果硬链接至连接工厂，那么每个用户均需要单独的连接工厂以拥有一个唯一的标识。

使用可在 `ConnectionFactory` 对象中配置的特殊变量替代语法，`Message Queue` 提供了一个途径，使客户机标识符特定于 `ConnectionFactory` 和用户。使用此方式时，创建长期订阅的多个用户可以使用一个 `ConnectionFactory` 对象，而不必担心命名冲突或安全性得不到保障。用户的长期订阅将因此得到保护，不会因另一用户设置了错误的客户机标识符而导致意外删除或不可用。

有关如何使用此 `Message Queue` 功能的详细信息，请参见《*Message Queue Java Client Developer's Guide*》中连接工厂属性的介绍。

无论何时，如果要创建长期订阅，客户机标识符必须由客户机使用 JMS API 在编写程序时设置，或者在客户机使用的 `ConnectionFactory` 对象中进行管理级别的配置。

可靠消息传送

JMS 定义了两种 *传送模式*：

持久性消息 保证这些消息只被传送一次和成功使用一次。对于这些消息，可靠性是优先考虑的因素。

非持久性消息 保证这些消息最多被传送一次。对于这些消息，可靠性并非主要的考虑因素。

对于*持久性*消息，确保可靠性有两个方面。一个是确保至目标和出自目标的持久性消息传送成功。另一个就是确保在将持久性消息传送至使用方之前，消息服务没有丢失持久性消息。

确认 / 事务

可靠消息传送的关键在于确保至目标和出自目标的持久性消息传送成功。这可以通过 Message Queue 会话支持的两个通用机制实现：确认或事务。事务（可以是本地事务或分布式事务）是由分布式事务管理器控制。

确认

可以将会话配置为使用确认来确保可靠传送。

对于生成方，这意味着在生成方的 `send()` 方法返回前，消息服务确认已向其目标发送持久性消息。对于使用方，这意味着在消息服务从该目标删除持久性消息前，客户机确认从该目标发出的持久性消息已传送并已使用。

本地事务

也可以将会话配置为*已处理*，这样，可以将一个或多个消息的生成和 / 或使用组成原子单元，也就是*事务*。JMS API 提供了启动、提交或回滚事务的方法。

在事务中生成或使用消息时，代理跟踪各个发送和接收过程，并在客户机发出提交事务的调用时完成这些操作。如果事务中特定的发送或接收操作失败，则出现异常。客户机代码通过忽略异常、重试操作或回滚整个事务来处理异常。在事务提交时，将完成所有成功的操作。在事务进行回滚时，将取消所有成功的操作。

本地事务的范围始终为一个会话。也就是说，可以将单个会话的上下文中执行的一个或多个生成方或使用方操作组成一个本地事务。

由于事务的范围只能为单个的会话，因此不存在既包括消息生成又包括消息使用的端对端事务。（换句话说，至目标的消息传送和随后进行的至客户机的消息传送不能放在同一个事务中。）

分布式事务

Message Queue 也支持*分布式事务*。也就是说，消息的生成和使用可作为较大的分布式事务的一部分，该分布式事务中包括涉及其他资源管理器（如数据库系统）的操作。在分布式事务中，分布式事务管理器使用在 Java 事务 API (JTA)、XA 资源 API 规范中定义的两阶段提交协议跟踪和管理由多个资源管理器（如消息服务和数据库管理器）执行的操作。在 Java 中，JTA 规范说明了资源管理器和分布式事务管理器之间的交互。

支持分布式事务是指消息传送客户机可通过 JTA 定义的 XAResource 接口参与分布式事务。此接口定义了实现两阶段提交的许多方法。当客户机端进行 API 调用时，Message Queue 代理只与分布式事务管理器（由 Java 事务服务 (JTS) 提供）协作来跟踪分布式事务中的各种发送和接收操作、事务状态并完成消息传送操作。

处理本地事务时，客户机通过忽略异常、重试操作或回滚整个分布式事务来处理异常。

Message Queue 通过 XA 连接工厂支持分布式事务。XA 连接工厂使您可以创建 XA 连接，XA 连接又可以使您创建 XA 会话（请参见第 37 页的“JMS 编程模型”）。另外，要支持分布式事务还需第三方 JTS 或提供 JTS 的 J2EE 兼容应用服务器。

持久性存储器

可靠性的另一个重要方面是确保持久性消息传送至目标后，消息服务在向使用方传送它们之前不会丢失这些消息。这意味着在持久性消息传送至目标时，消息服务必须将其放入持久性数据存储（请参见第 57 页的“持久性管理器”）。如果消息服务由于某种原因导致失败，它可以恢复此消息并将此消息传送至相应的使用方。虽然这样增加了消息传送的开销，但却增加了可靠性。

消息服务还必须存储长期订阅。这是因为要确保主题目标模式下的可靠传送，仅仅恢复持久性消息是不够的。消息服务还必须恢复有关主题的长期订阅的信息，否则它就不能将消息传送至在消息到达时处于非活动状态、随后又恢复活动状态的订阅者。

需要可靠消息传送的消息传送应用程序必须将消息指定为持久性消息，并使用队列目标或主题目标的长期订阅。

性能折衷

消息传送的可靠性越高，需要的开销和带宽就越多。性能和可靠性之间的折衷是设计时要重点考虑的一个方面。您可以选择生成和使用非持久性消息来获得最佳性能。另一方面，您可以通过生成和使用持久性消息并使用事务会话来获得最佳可靠性。在这两种极端之间有许多选择，这取决于应用程序的要求，包括使用 Message Queue 特有的连接和确认属性，请参见《Message Queue Java Client Developer's Guide》。在第 209 页的“影响性能的应用程序设计因素”中对这些折衷进行了更为详细的讨论。

消息选择

JMS 提供了一种机制，使用它，消息服务可根据消息选择器中的标准来执行消息过滤和路由。生成方客户机可在消息中放入应用程序特有的属性，而使用方客户机可使用基于这些属性的选择标准来表明对消息是否感兴趣。这就简化了客户机的工作，并避免了向不需要这些消息的使用方传送消息的开销。然而，它也使得处理选择标准的消息服务增加了一些额外开销。在 JMS 规范中对消息选择器语法和语义进行了简要说明。

消息顺序和优先级

通常，可以确保将单个会话向目标发送的所有消息按其发送顺序传送至使用方。然而，如果为这些消息分配了不同的优先级，消息传送系统将首先尝试传送优先级较高的消息。

除此之外，客户机应用程序使用消息的顺序只与消息的生成顺序存在一个粗略的关系。这是因为向目标传送消息和从目标传送消息可以取决于多种影响时间的因素，如消息的发送顺序、发送消息的会话（连接）、是否是持久性消息、消息的有效期、消息的优先级、队列目标的消息传送规则（请参见第 68 页的“队列目标”）和消息服务可用性。

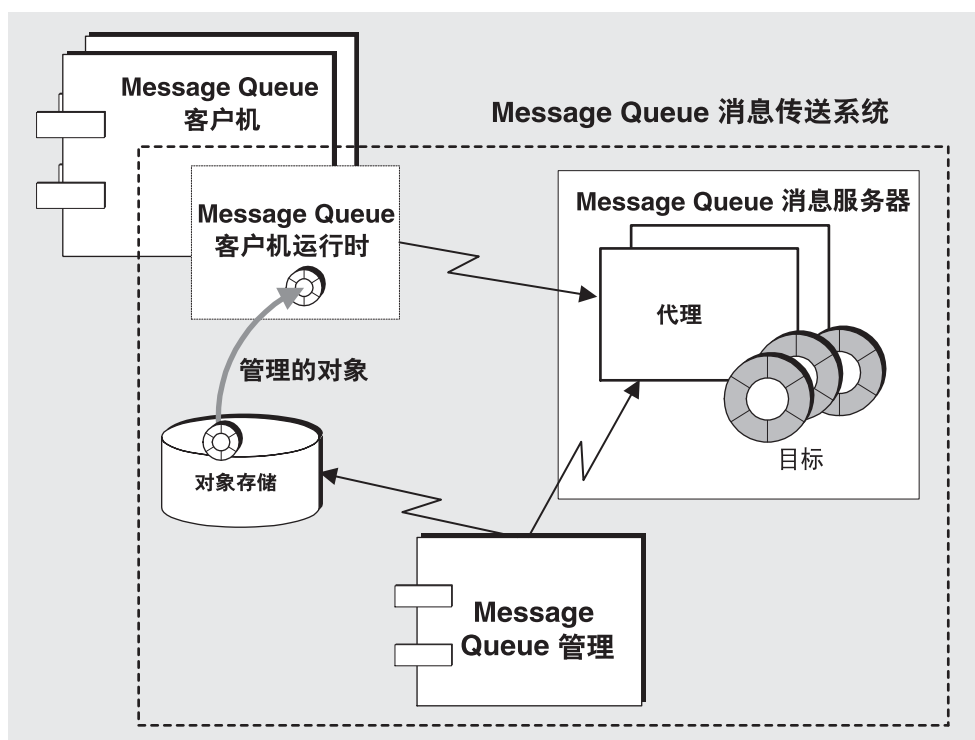
对于使用多个互连代理（请参见第 73 页的“多代理群集（企业版）”）的 Message Queue 消息服务来说，客户机使用消息的顺序更为复杂，这是因为从不同代理的目标传送消息的顺序是不确定的。因此，一个代理可能会先于另一个代理传送消息，即使是后者先接收到消息。

无论何时，对于一个给定的使用方，优先级较高的消息总是先于优先级较低的消息。

Message Queue 消息传送系统

本章介绍 Sun Java™ System Message Queue 消息传送系统，其中重点说明的是系统的主要组成部分（如图 2-1 所示），并阐述了这些组成部分如何配合工作来提供可靠的消息传送。

图 2-1 Message Queue 系统的体系结构



Message Queue 消息传送系统的主要组成部分（如图 2-1 所示）包括：

- Message Queue 消息服务器
- Message Queue 客户机运行时
- Message Queue 受管理对象
- Message Queue 管理

前三个组成部分将在下面的小节中进行详细说明。最后一个部分将在第 3 章“Message Queue 管理任务和工具”中介绍。

Message Queue 消息服务器

本节介绍 Message Queue 消息服务器（如第 47 页的图 2-1 所示）的各个组成部分，包括：

代理 Message Queue 代理为 Message Queue 消息传送系统提供传送服务。消息传送依赖于大量支持组件，这些组件负责处理连接服务、消息的路由和传送、持久性、安全性以及日志记录（有关详细信息，请参见“代理”）。消息服务器可以使用一个或多个代理实例（请参见第 73 页的“多代理群集（企业版）”）。

物理目标 消息的传送过程分为两个阶段：首先从生成消息的客户机（生成方客户机）将消息发送到由代理维护的物理目标，然后再从目标发送到一个或多个使用消息的客户机（使用方客户机）。物理目标是指代理的物理内存和 / 或持久性存储器中的位置（有关详细信息，请参见第 68 页的“物理目标”）。

代理

在 Message Queue 消息传送系统中，消息传送（从生成方客户机到目标，然后再从目标到一个或多个使用方客户机）由代理（或以串连模式工作的代理实例群集）来执行。要执行消息传送，代理必须建立与客户机之间的通信通道、执行验证和授权、适当的路由消息、确保可靠传送并提供用于监视系统性能的数据。

为了实现这么多复杂的功能，代理使用了大量的各种内部组件，每个组件都在传送过程中担当着各自不同的角色。图 2-2 所示为代理组件，表 2-1 则对这些组件进行了简要说明。消息路由器组件执行关键的消息路由和传送服务，而其他组件则提供消息路由器所依赖的重要的支持服务。

图 2-2 代理服务组件

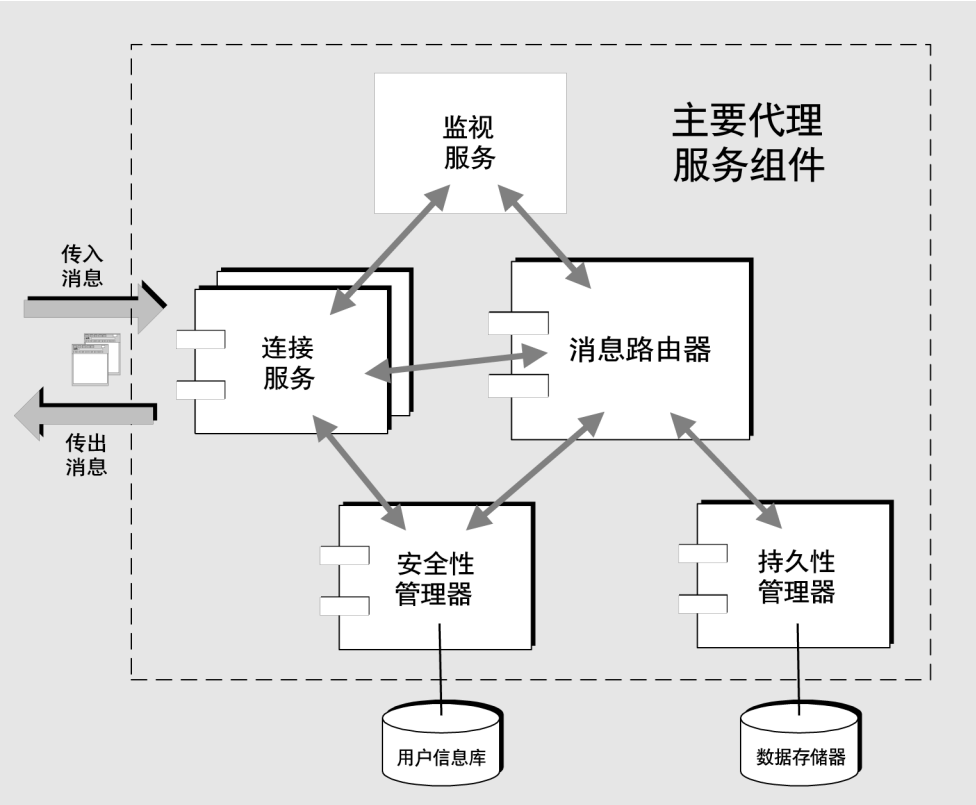


表 2-1 主要代理服务组件及功能

组件	说明 / 功能
消息路由器	管理消息的路由和传送，这里的消息包括：JMS 消息以及在 Message Queue 消息传送系统中用于支持 JMS 消息传送的控制消息。
连接服务	管理代理和客户机之间的物理连接，用于传输外来和外出消息。
持久性管理器	管理向持久性存储器写入数据，这样即使系统发生故障也不会导致无法传送 JMS 消息。
安全性管理器	为请求连接到代理的用户提供验证服务，并为通过验证的用户提供授权服务（访问控制）。

表 2-1 主要代理服务组件及功能（续）

组件	说明 / 功能
监视服务	生成度量依据和诊断信息，这些内容可以写入到大量的输出通道中。管理员利用这些内容进行监视和管理。

可以根据负荷条件、应用程序的复杂性等因素配置这些内部组件，以优化代理的性能。下面的几个小节将详细介绍不同组件执行的功能及其属性（可以配置这些属性来影响代理的行为）。

连接服务

Message Queue 代理既支持与 Message Queue 应用程序客户机的通信，也支持与 Message Queue 管理客户机的通信（请参见第 86 页的“[Message Queue 管理工具](#)”）。各个服务取决于其服务类型和协议类型。

服务类型 指定服务提供的是 JMS 消息传送 (NORMAL) 服务，还是 Message Queue 管理 (ADMIN) 服务

协议类型 指定支持服务的基本传输协议层。

表 2-2 显示了 Message Queue 代理当前支持的连接服务：

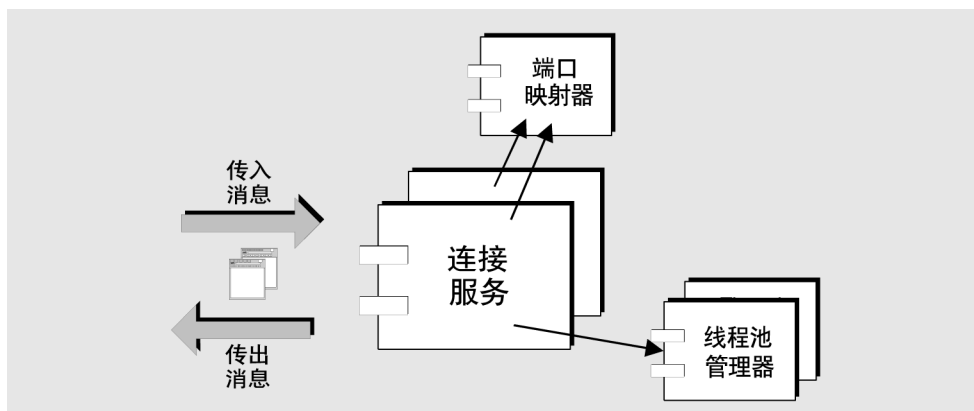
表 2-2 代理支持的连接服务

服务名称	服务类型	协议类型
jms	NORMAL	tcp
ssljms（企业版）	NORMAL	tls（其安全性基于 SSL）
httpjms（企业版）	NORMAL	http
httpsjms（企业版）	NORMAL	https（其安全性基于 SSL）
admin	ADMIN	tcp
ssladmin（企业版）	ADMIN	tls（其安全性基于 SSL）

可以将代理配置为运行以上任一或全部连接服务。每个连接服务仅在由代理的主机名和端口号指定的特定端口上可用。端口可动态分配，或者可指定连接服务可用的端口。

每项服务都具有一个线程池管理器，并向通用端口映射器服务注册它本身，如[图 2-3](#)所示。

图 2-3 连接服务支持



端口映射器

Message Queue 提供了一个 **端口映射器**，用于将端口映射到不同的连接服务。端口映射器本身位于标准端口号 7676。当客户机建立与代理的连接时，首先与端口映射器进行通信，请求其所需的连接服务端口号。

在配置 jms、ssljms、admin 和 ssladmin 等连接服务时，也可以为它们分配 **静态** 端口号，但这是在特殊情况下使用的手段（例如，通过防火墙连接），一般不建议使用。分别使用第 276 页的表 C-1 和第 287 页的表 C-3（这两个表格位于附录 C “HTTP/HTTPS 支持（企业版）”。）中说明的属性来配置 httpjms 和 httpsjms 服务。

线程池管理器

每个连接服务都是多线程的，从而支持多个连接。这些连接所需的线程在线程池中进行维护，而线程池又由 **线程池管理器** 组件来管理。通过配置线程池管理器，可以设置线程池中维护的线程的最大和最小数量。当连接需要线程时，这些线程将被添加到线程池中。当线程数量超过最小数量时，系统将关闭变为空闲状态的线程，直到达到最小阈值，以此来节省内存资源。您希望此值足够大，这样就不必不断创建新线程。如果连接负荷较重，线程数量可能会增加，直到达到线程池的最大数量。此后，连接不得不一直等待，直到某个线程可用。

线程池中的线程可专用于单个连接（**专用模型**），也可根据需要分配到多个连接（**共享模型**）。

专用模型 每个到代理的连接需要两个专用线程：一个用于处理连接的外来消息，另一个用于处理连接的外出消息。这样，连接的数量就被限制为线程池中最大线程数量的一半，但却提供了较高的性能。

共享模型（企业版） 只要发送或接收消息，连接就会由共享线程处理。由于每个连接不需要专用线程，此模型增加了连接服务可支持的连接数量，因而，也增加了代理可支持的连接数量。但这同时带来了共享线程相关的某些性能开销。线程池管理器使用一组分配器线程来监视连接活动并根据需要将连接分配到线程。可通过限制每个此类分配器线程所监视的连接的数量使此活动带来的性能开销最小化。

安全性

每个连接服务均支持特定的验证和授权（访问控制）功能（请参见第 60 页的“安全性管理器”）。

连接服务属性

表 2-3 所示为与连接服务相关的可配置属性。（有关配置这些属性的说明，请参见第 5 章“启动与配置代理”）

表 2-3 连接服务属性

属性名称	说明
imq.service.activelist	按名称列出启动代理时将变为活动状态的连接服务，服务之间用逗号分隔。支持的服务包括：jms、ssljms、httpjms、httpsjms、admin 和 ssladmin。 默认值：jms 和 admin
imq.ping.interval	代理通过连接连续尝试 ping Message Queue 客户机运行的时间周期。 默认值：120 秒
imq.hostname	指定当有多个主机可用（例如，一台计算机中安装了多个网络接口卡）时所有连接服务将绑定到的主机（主机名或 IP 地址）。 默认值：所有可用的 IP 地址
imq.portmapper.port	指定代理的主端口，即端口映射器所在的端口。如果主机上运行多个代理实例，则必须为每个实例分配唯一的端口映射器端口。 默认值：7676
imq.portmapper.hostname	指定当有多个主机可用（例如，一台计算机中安装了多个网络接口卡）时端口映射器将绑定到的主机（主机名或 IP 地址）。 默认值：继承 imq.hostname 的值
imq.portmapper.backlog	指定端口映射器在拒绝请求之前可处理的并发请求的最大数量。该属性设置了可存储在操作系统待办事项中等待端口映射器处理的请求的数量。 默认值：50

表 2-3 连接服务属性 (续)

属性名称	说明
<code>imq.service_name.protocol_type¹.port</code>	仅适用于 <code>jms</code> 、 <code>ssljms</code> 、 <code>admin</code> 和 <code>ssladmin</code> 服务，指定命名的连接服务的端口号。 默认值：0（端口由端口映射器动态分配） 要配置 <code>httpjms</code> 和 <code>httpsjms</code> 连接服务，请参见附录 C “HTTP/HTTPS 支持（企业版）”。
<code>imq.service_name.protocol_type¹.hostname</code>	仅适用于 <code>jms</code> 、 <code>ssljms</code> 、 <code>admin</code> 和 <code>ssladmin</code> 服务，指定当有多个主机可用（例如，一台计算机中安装了多个网络接口卡）时命名的连接服务将绑定到的主机（主机名或 IP 地址）。 默认值：继承 <code>imq.hostname</code> 的值
<code>imq.service_name.min_threads</code>	指定线程数量，达到该数量后，线程池中将维护此数量的线程供命名的连接服务使用。 默认值：取决于连接服务（请参见第 118 页的表 5-1）。
<code>imq.service_name.max_threads</code>	指定线程数量，达到该数量后，新的线程将不会被添加到线程池中供命名的连接服务使用。该数量必须大于零，并且必须大于 <code>min_threads</code> 的值。 默认值：取决于连接服务（请参见第 118 页的表 5-1）。
<code>imq.service_name.threadpool_model</code>	为命名的连接服务指定线程是专用于特定的连接 (<code>dedicated</code>)，还是根据需要由多个连接共享 (<code>shared</code>)。共享模型（线程池管理）将增加代理支持的连接数量，但只能用于 <code>jms</code> 和 <code>admin</code> 连接服务。 默认值：取决于连接服务（请参见第 118 页的表 5-1）。
<code>imq.shared.connectionMonitor_limit</code>	仅适用于共享线程池模型，用于指定分配器线程可监视的连接的最大数量。（系统会分配足够多的分配器线程来监视所有连接。）此值越小，系统向线程分配活动连接的速度就越快。值 -1 表示没有限制。 默认值：取决于操作系统（请参见第 118 页的表 5-1）。

1. `protocol_type` 为表 2-2 中指定的协议类型。

消息路由器

使用支持的连接服务在客户机与代理之间建立起连接后，即可进行消息的路由和传送。

基本传送机制

广义来讲，代理处理的消息可分为两类：由生成方客户机发往使用方客户机的 JMS 消息（即有效负荷消息）以及从客户机发出或发往客户机的大量用于支持 JMS 消息传送的控制消息。

如果外来消息为 JMS 消息，代理会根据其目标的类型（队列或主题）将其发送到使用方客户机：

- 如果目标为主题，JMS 消息将立即被路由到此主题的所有活动订阅者。对于未处于活动状态的长期订阅者，消息路由器将保留消息，直到订阅者变为活动状态后再将消息传送给该订阅者。
- 如果目标为队列，JMS 消息将被放置在相应的队列中，并在到达队列前端时被传送到相应的使用方。消息到达队列前端的顺序取决于它们进入队列的顺序以及它们的优先级。

消息路由器将消息传送到其所有目标使用方后，将从内存中删除此消息。如果该消息为持久性消息（请参见第 43 页的“可靠消息传送”），还会将其从代理的持久性数据存储中删除。

可靠传送：确认和事务

如果增加可靠传送（请参见第 43 页的“可靠消息传送”）的要求，则刚才讨论的传送机制就会复杂得多。可靠传送涉及两个方面：一是确保消息成功传送到代理以及从代理成功传送出去，二是确保在实际传送消息前，代理没有丢失消息或传送信息。。

为了确保至代理和出自代理的消息传送能够成功，Message Queue 使用了大量称为“确认”的控制消息。

例如，当生成方向目标发送 JMS 消息（有效负荷消息而非控制消息）时，代理会发回一条说明它已接收到 JMS 消息的控制消息，这条控制消息就是代理确认。（默认情况下，只有当生成方将 JMS 消息指定为持久性消息时，Message Queue 才会进行上述操作。）生成方客户机使用代理确认来确保至目标的传送成功（请参见第 77 页的“消息生成”）。

同样，当代理将 JMS 消息传送到使用方时，使用方客户机会发回一个确认，说明它已接收并处理了此消息。客户机可以在创建会话对象时指定发送这些确认的自动程度和频率，但原则是消息路由器在收到每个消息使用方（即接收它所传送的消息的客户机，例如某个主题的每个订阅者）的确认之前，不会从内存中删除 JMS 消息。

对于某个主题的长期订阅者，消息路由器会将每条 JMS 消息保留在该目标中，以便在每个长期订阅者变为活动使用方时将消息传送出去。消息路由器会在接收客户机确认时进行记录，并且只有在接收到所有确认后才会删除 JMS 消息（除非在这之前 JMS 消息已过期）。

另外，消息路由器会将代理确认发送回客户机，确认已接收到客户机确认。使用方客户机使用代理确认来确保代理不会多次传送 JMS 消息（请参见第 77 页的“消息使用”）。如果由于某些原因使代理未能接收到客户机确认，则可能发生多次传送的情况。

如果代理没有接收到客户机确认并传送了 JMS 消息，此消息将被标注“重新传送”标志。通常，如果在代理接收到客户机确认之前客户机连接被关闭，并且随后又打开了新的连接，那么代理会重新传送 JMS 消息。例如，如果队列的消息使用方在确认消息之前脱机，随后另一位使用方向队列进行了注册，则代理会向该新使用方重新传送未确认的消息。

上述客户机和代理的确认过程同样适用于编组为事务的 JMS 消息传送。对于后者，客户机和代理确认在事务级别以及各个 JMS 消息发送或接收级别运行。当提交事务时，将自动发送代理确认。

代理会跟踪事务，使它们可以在出现故障时进行提交或回滚。该事务管理也支持本地事务（较大的分布式事务的一部分，请参见第 44 页的“分布式事务”）。代理会一直跟踪这些事务的状态，直到它们被提交。当代理启动时，它会检查所有未提交的事务，在默认情况下，回滚未处于 PREPARED 状态的所有事务。

可靠传送：持久性

可靠传送的另一个方面是确保在实际传送消息前，代理没有丢失消息或要传送的信息。通常，消息将始终保留在内存中，直到被传送或过期。但如果代理出现故障，这些消息将会丢失。

生成方客户机可以将消息指定为持久性消息，在这种情况下，消息路由器会将此消息传送到持久性管理器，该管理器将把消息存储在数据库或文件系统中（请参见第 57 页的“持久性管理器”），这样即使代理出现故障，消息也可以得到恢复。

管理内存资源和消息流

代理的性能和稳定性取决于可用的系统资源以及资源（如内存）的使用效率。特别是，当消息的生成大大快于使用时，消息路由器可能无法承受大量的消息，可能会用光其所有内存资源。为防止这种情况，消息路由器使用了三级内存保护以在资源不足时保持操作系统运转：

单个目标的消息限制 可设置物理目标的属性，指定消息数量的限制和消息使用的总内存限制（请参见第 154 页的表 6-10），也可设置在达到这些限制时，消息路由器将采取四种响应中的哪一种。四种限制行为为：

- 减慢消息生成方
- 丢弃内存中最旧的消息
- 根据消息的生存期丢弃内存中优先级最低的消息
- 拒绝最新的消息

系统范围消息限制 系统范围消息限制构成了第二道防线。可指定应用于系统中所有目标的系统范围限制：消息总数和所有消息占用的内存（请参见第 56 页的表 2-4）。如果达到了任何系统范围消息限制，消息路由器将拒绝新消息。

系统内存阈值 系统内存阈值是第三道防线。可指定可用系统内存的阈值，如果超过该阈值，代理可采取更为严格的操作，以此来防止内存过载。采取的操作取决于内存资源的状态：green（可用内存充足）、yellow（代理内存不足）、orange（代理内存严重不足）、red（代理无可用内存）。随着代理的内存状态由 green 变为 yellow，再变为 orange，最后变为 red，代理所采取的措施也会越来越严格，操作类型如下：

- 将消息从活动内存交换至持久性存储器（请参见第 57 页的“持久性管理器”）；通常可以将不存储的非持久性消息换出内存，以便系统回收内存
- 限制非持久性消息的生成方，最后甚至会停止进入代理的消息流。（持久性消息流自动由代理确认的每条消息的要求限制）

这些措施都会降低性能。

如果达到系统内存的阈值，则说明没有适当地设置针对目标的消息限制以及系统范围消息限制。在某些情况下，阈值不可能及时地捕捉到潜在的内存过载。因此，您不应依赖此功能来控制内存资源，而应该分别配置目标，共同优化内存资源。

消息路由器属性

有关管理内存资源的系统范围限制和系统内存阈值的详细信息，请参见表 2-4。（有关设置这些属性的说明，请参见第 5 章“启动与配置代理”）

表 2-4 消息路由器属性

属性名称	说明
imq.message.expiration.interval	指定过期消息的回收频率（以秒为单位）。默认值：60
imq.system.max_count	指定代理保留的消息的最大数量。附加消息将被拒绝。值 -1 表示没有限制。默认值：-1
imq.system.max_size	指定代理保留的消息的最大总大小（以字节、千字节或兆字节为单位）。附加消息将被拒绝。值 -1 表示没有限制。默认值：-1
imq.message.max_size	指定消息主体允许的最大大小（以字节、千字节或兆字节为单位）。任何大于此大小的消息将被拒绝。值 -1 表示没有限制。默认值：70m（兆字节）
imq.resource_state.threshold	指定将触发各个内存资源状态的内存占用百分比。资源状态的值可以是 green、yellow、orange 和 red。默认值：分别为 0、80、90 和 98

表 2-4 消息路由器属性 (续)

属性名称	说明
<code>imq.resource_state.count</code>	指定当触发各个内存资源状态时允许同时处理的一批外来消息的最大数量。当系统内存愈加不足时，此限制可限制消息的生成方。 默认值：分别为 5000、500、50 和 0
<code>imq.transaction.autorollback</code>	指定 (true/false) 当启动代理时，处于 PREPARED 状态的分布式事务是否自动回滚。如果为 false，则必须使用 <code>imqcmd</code> 手动提交或回滚事务（请参见第 162 页的“管理事务”）。 默认值：false

持久性管理器

对于发生故障后待恢复的代理，需要重新创建其消息传送操作的状态。这需要它将所有持久性消息以及基本的路由和传送信息保存到数据存储中。*持久性管理器*组件用于管理此信息的写入和检索。

为了恢复出现故障的代理，不仅需要恢复未传送的消息，代理还必须完成以下操作：

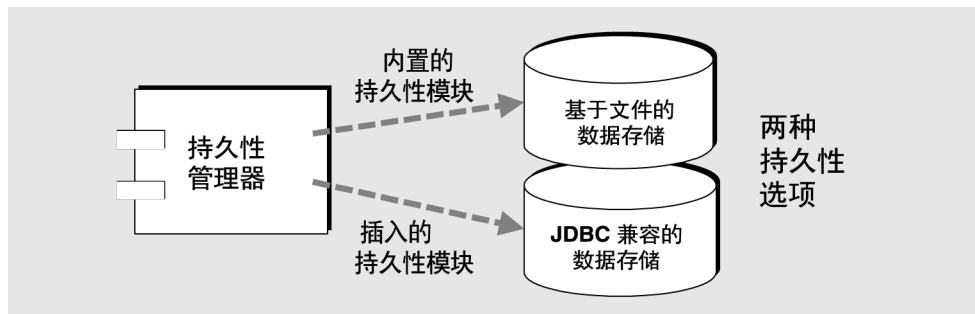
- 重新创建目标
- 恢复每个主题的长期订阅列表
- 恢复每条消息的确认列表
- 重新生成所有已提交事务的状态

持久性管理器用于管理所有这些状态信息的存储和检索。

当代理重新启动时，它将重新创建目标和长期订阅，恢复持久性消息和所有事务的状态，并重新创建未传送消息的路由表。然后代理才可以恢复消息传送。

Message Queue 既支持内置的持久性模块，也支持插入的持久性模块（请参见图 2-4）。内置的持久性是基于文件的数据存储。插入的持久性使用 Java 数据库连接 (JDBC™) 接口，并需要 JDBC 兼容的数据存储。通常，内置的持久性比插入的持久性速度更快，但某些用户更希望获得使用 JDBC 兼容的数据库系统的冗余和管理功能。

图 2-4 持久性管理器支持



内置的持久性

Message Queue 持久性存储器的默认解决方案是基于文件的数据存储。该解决方案使用单个文件来存储持久性数据，如消息、目标、长期订阅和事务等。

基于文件的数据存储位于由与数据存储相关联的代理实例的名称 (*instanceName*) 标识的目录内（请参见附录 A “[Message Queue 数据的位置](#)”）：

```
.../instances/instanceName/fs350/
```

可构建基于文件的数据存储，以便持久性消息所在的目标将其保存在目录中。大多数消息存储在由大小可变的记录组成的单个文件中。

为减少添加和删除消息带来的碎片，可压缩大小可变的记录文件（请参见第 159 页的“[压缩目标](#)”）。另外，内置的持久性管理器可在其各自的文件中存储大小超过配置阈值 (`imq.persist.file.message.max_record_size`) 的消息，而不是存储在大小可变的记录文件中。对于这些单个文件，可维护文件池以便重新使用这些文件。当消息文件不再需要时，无需删除，可将其添加至目标目录的空闲文件池中，以便存储新的消息。

可配置目标文件池中文件的最大数量

(`imq.persist.file.destination.message.filepool.limit`) 并指定文件池中空闲文件所占的百分比 (`imq.persist.file.message.filepool.cleanratio`)，达到此百分比后将清除文件池（清空），而不是仅标记这些空闲文件供重复使用（不清空）。此清除文件的百分比越高，文件池所需的磁盘空间越小，但维护文件池所需的开销就越大。另外，也可以指定关闭时是否清除标记的文件 (`imq.persist.file.message.cleanup`)。如果清除这些文件，它们所占用的磁盘空间就会减少，但代理将需要更长的时间来进行关闭操作。

所有其他持久性数据（目标、长期订阅和事务）都存储在其自己的单独文件中：所有目标在一个文件中，所有长期订阅在另一个文件中，依此类推。

要使可靠性最大化，可指定 (imq.persist.file.sync.enabled) 持久性操作使用物理存储设备与内存中状态同步。这可帮助避免由于系统崩溃而导致的数据丢失，但会消耗一部分性能。

由于数据存储中可能包含敏感的或私有的消息，因此建议保护 `...instances/instanceName/fs350/` 目录，防止未授权就进行访问。有关说明，请参见第 303 页的“确保持久性数据的安全”。

插入的持久性

您可以设置一个代理，以访问任何可通过 JDBC 驱动程序进行访问的数据存储。该过程涉及：设置多个与 JDBC 相关的代理配置属性、使用数据库管理器实用程序 (imqdbmgr) 来创建具有相应架构的数据存储。附录 B “设置插入的持久性”。中详细说明了上述过程以及相关的配置属性。

持久性管理器属性

第 59 页的表 2-5 详细说明了与持久性相关的配置属性。（有关设置这些属性的说明，请参见第 5 章“启动与配置代理。”）

除第一个属性之外，表 2-5 中的所有属性都仅能用于内置持久性。与插入的持久性相关的属性在第 267 页的表 B-1 中。

表 2-5 持久性管理器属性

属性名称	说明
imq.persist.store	指定代理是使用内置的、基于文件的 (file) 持久性，还是使用插入的、JDBC 兼容的 (jdbc) 持久性。 默认值: file
imq.persist.file.sync.enabled	指定持久性操作是否使内存中状态与物理存储设备同步。如果为 true，则可以避免由于系统崩溃导致的数据丢失，但持久性操作的性能会有所降低。 默认值: false
imq.persist.file.message.max_record_size	适用于内置的、基于文件的持久性，指定将添加到消息存储文件中的消息的最大大小（相对于存储在单独文件中的消息）。 默认值: 1m（兆字节）
imq.persist.file.destination.message.filepool.limit	适用于内置的、基于文件的持久性，指定目标文件池中可供重复使用的空闲文件的最大数量。该数量越大，代理处理持久性数据的速度越快。超过该值的空闲文件将被删除。代理将根据需要创建文件或删除超过此限制的文件。 默认值: 100

表 2-5 持久性管理器属性 (续)

属性名称	说明
imq.persist.file.message. filepool.cleanratio	适用于内置的、基于文件的持久性，指定保持清除状态（清空）的目标文件池中空闲文件的百分比。该值越大，在操作中清除文件所需的开销越大，但文件池所需的磁盘空间越小。 默认值：0
imq.persist.file.message. cleanup	适用于内置的、基于文件的持久性，指定代理在关闭时是否清除目标文件池中的空闲文件。如果值为 false，则关闭代理的速度会加快，但文件存储需要的磁盘空间会更多。 默认值：false

安全性管理器

Message Queue 提供验证和授权（访问控制）功能，同时也支持加密功能。

验证和授权功能取决于用户信息库（请参见第 61 页的图 2-5）：包含消息传送系统的用户信息（用户名、密码和组成员资格）的文件、目录或数据库。用户名和密码用于在请求连接至代理时对用户进行验证。用户名和组成员资格（与访问控制文件配套使用）用于授权操作，例如为目标生成消息或使用目标的消息。

Message Queue 管理员可填充 Message Queue 提供的用户信息库（请参见第 184 页的“使用文本文件用户信息库”），也可以将原有的 LDAP 用户信息库插入到安全性管理器组件中（请参见第 190 页的“使用 LDAP 服务器管理用户信息库”）。文本文件用户信息库使用起来较简单，但在安全性上也易遭攻击，因此应只将其用于测试和开发目的，而 LDAP 用户信息库相对较安全，因此更适合用于生产目的。

验证

Message Queue 安全性支持基于密码的验证。当客户机请求连接到代理时，该客户机必须提交用户名和密码。安全性管理器将把客户机提交的用户名和密码与用户信息库中存储的用户名和密码进行比较。密码在从客户机传送到代理的过程中，将使用 Base 64 编码或消息摘要 (MD5) 进行编码。要想获得更安全的传送，请参见第 62 页的“加密（企业版）”。可以分别配置每个连接服务使用的编码的类型，也可以基于代理范围设置编码方式。

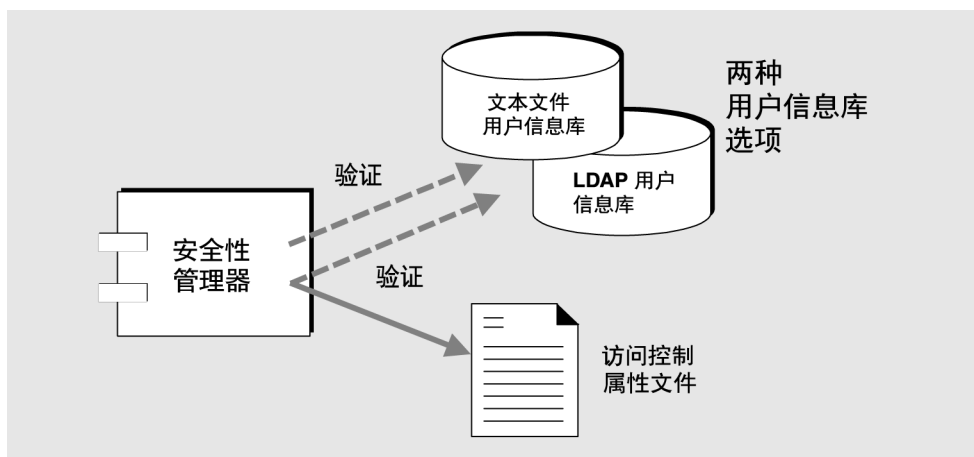
授权

客户机应用程序的用户通过验证后，即可以获得授权来执行各种 Message Queue 相关的活动。安全性管理器既支持基于用户的访问控制，也支持基于组的访问控制：根据用户在用户信息库中分配到的用户名或组的不同，该用户所拥有的执行特定 Message Queue 操作的权限也不同。这些访问控制在访问控制属性文件中指定（请参见图 2-5）。

当用户尝试执行某个操作时，安全性管理器将检查该用户的用户名和组成员资格（通过用户信息库）是否属于被指定允许访问该操作的用户名或组成员资格（在访问控制属性文件中）。访问控制属性文件指定了执行以下操作的权限：

- 建立与代理的连接
- 访问目标：创建任意给定目标或所有目标的使用方、生成方或队列浏览器
- 自动创建目标

图 2-5 安全性管理器支持



默认的访问控制属性文件只明确引用一个组：*管理员*（请参见第 187 页的“组”）。*管理员*组中的用户拥有管理服务连接权限。管理服务使用户可以执行管理功能，如创建目标以及监视和控制代理。默认情况下，您定义的其他任何组中的用户都无法获得管理服务连接。

作为 Message Queue 管理员，可以在用户信息库中定义组并使用户与这些组关联（虽然在文本文件用户信息库中不完全支持组）。然后，通过编辑访问控制属性文件，可以按照用户和组的不同目的（生成和使用消息，或浏览队列目标中的消息）指定对目标的权限。可以分别指定各个目标或指定所有目标仅能够被特定用户或组访问。

另外，如果将代理配置为允许自动创建目标（请参见第 70 页的“自动创建的（与管理员创建的相对）目标”），则可以通过编辑访问控制属性文件来控制代理可为谁自动创建目标。

加密（企业版）

要对客户机和代理之间发送的消息进行加密，需要使用基于安全套接字层 (SSL) 标准的连接服务。SSL 通过在启用 SSL 的代理与启用 SSL 的客户机之间建立加密连接来提供连接级别的安全性。

为了使用 Message Queue 基于 SSL 的连接服务，需要使用密钥工具实用程序 (imqkeytool) 生成专用密钥 / 公用密钥对。此实用程序将公用密钥嵌入自签名的证书中，并将其放入 Message Queue 密钥存储中。Message Queue 密钥存储本身设有密码保护，要解除锁定，需要在启动时提供密钥存储的密码。请参见第 198 页的“加密：使用基于 SSL 的服务（企业版）”。

解除密钥存储的锁定后，代理即可将证书传送给所有请求连接的客户机。然后，客户机使用此证书建立与代理的加密连接。

表 2-6 所示为验证、授权、加密以及其他安全性通信的可配置属性。（有关配置这些属性的说明，请参见第 5 章“启动与配置代理”）

表 2-6 安全性管理器属性

属性名称	说明
imq.authentication.type	指定传送密码时应使用 Base 64 编码 (basic) 还是 MD5 摘要 (digest)。设置代理支持的所有连接服务的编码。 默认值: digest
imq.service_name.authentication.type	指定传送密码时应使用 Base 64 编码 (basic) 还是 MD5 摘要 (digest)。设置命名的连接服务的编码，此设置将覆盖任意代理范围的设置。 默认值: 继承 imq.authentication.type 的值
imq.authentication.basic.user_repository	对于 Base 64 编码，指定用于验证的用户信息库的类型：基于文件 (file) 或 LDAP (ldap)。有关其他 LDAP 属性，请参见第 190 页的表 8-5。 默认值: file
imq.authentication.client.response.timeout	指定系统等待客户机响应来自代理的验证请求的时间（以秒为单位）。 默认值: 180（秒）
imq.accesscontrol.enabled	为代理支持的所有连接服务设置访问控制 (true/false)。表明系统是否要检查已验证的用户是否如访问控制属性文件所指定，拥有使用连接服务或执行与特定目标有关的特定 Message Queue 操作的权限。 默认值: true
imq.service_name.accesscontrol.enabled	为命名的连接服务设置访问控制 (true/false)，此设置将覆盖代理范围的设置。表明系统是否要检查已验证的用户是否如访问控制属性文件所指定，拥有使用连接服务或执行与特定目标有关的特定 Message Queue 操作的权限。 默认值: 继承 imq.accesscontrol.enabled 的值

表 2-6 安全性管理器属性 (续)

属性名称	说明
imq.accesscontrol.file.filename	为代理实例支持的所有连接服务指定访问控制属性文件的名称。文件名称指定访问控制目录的相对文件路径（请参见附录 A “Message Queue 数据的位置”）。 默认值: accesscontrol.properties
imq.service_name.accesscontrol.file.filename	为代理实例的命名连接服务指定访问控制属性文件的名称。文件名称指定访问控制目录的相对文件路径（请参见附录 A “Message Queue 数据的位置”）。 默认值: 继承 imq.accesscontrol.file.filename 的值。
imq.passfile.enabled	指定 (true/false) 是否在密码文件中指定用于安全性通信的用户密码（适用于 SSL、LDAP 和 JDBC™）。 默认值: false
imq.passfile.dirpath	指定包含密码文件的目录的路径（根据操作系统）。 默认值: 请参见附录 A “Message Queue 数据的位置”
imq.passfile.name	指定密码文件的名称。 默认值: 密码文件
imq.keystore.property_name	适用于基于 SSL 的服务: 指定与 SSL 密钥存储相关的安全性属性。请参见第 199 页的表 8-8。

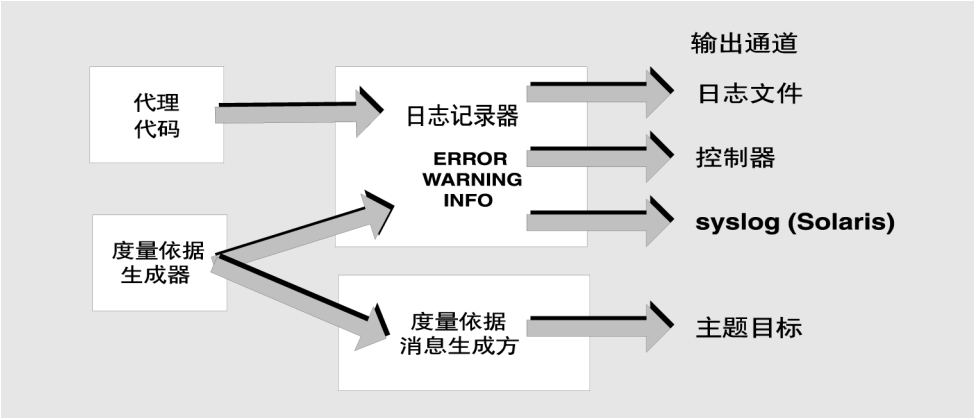
监视服务

代理包含大量用于监视和诊断其操作的组件。包括:

- 生成数据的组件（记录事件的代理代码和度量依据生成器）
- 通过多种输出通道记录输出信息的日志记录器组件（请参见 “日志记录器”）
- 将包含度量依据信息的 JMS 消息发送到主题目标供 JMS 监视客户机使用的消息生成方。

图 2-6 中是通用方案的图解。

图 2-6 监视服务支持



度量依据生成器

度量依据生成器提供有关代理活动的信息，如流入、流出代理的消息、代理内存中的消息数量及其使用的内存量、打开的连接数量和正在使用的线程数量。

可以打开或关闭度量依据数据的生成，并指定生成度量依据报告的频率。

日志记录器

Message Queue 日志记录器记录代理代码和度量依据生成器生成的信息，并将该信息写入多个输出通道中：标准输出（控制台）、日志文件以及 Solaris™ 平台上的 syslog 守护程序进程。

可指定日志记录器收集的信息类型，以及写入每个输出通道的类型。

例如，可以指定日志记录器级别（日志记录器收集的信息类型）：从最严重和最重要的信息（错误）到次要信息（度量依据数据）。表 2-7 所示为信息的种类（按重要性的降序排列）：

表 2-7 日志种类

种类	说明
ERROR	指出可能导致系统故障的问题的消息。
WARNING	需要注意但不会导致系统故障的警报。
INFO	度量依据及其他信息性消息的报告。

要设置日志记录器级别，需要指定其中的某个种类。日志记录器将记录指定种类及所有更高级别种类的数据。例如，如果指定记录 `WARNING` 级别的信息，则日志记录器将记录警告信息和错误信息。

可以分别指定写入到每个输出通道的日志记录器的种类集。例如，如果将日志记录器级别设置为 `INFO`，则可以指定只将错误和警告写入控制台，只将信息（度量依据数据）写入日志文件。（有关配置和使用 `Solaris syslog` 的信息，请参见 `syslog(1M)`、`syslog.conf(4)` 和 `syslog(3C)` 手册页。）

对于日志文件，还可以指定何时关闭日志文件并将输入转移到新文件。在日志文件达到指定的大小或生存期后，将保存该文件并创建一个新的日志文件。日志文件写入一个目录中，该目录用与日志文件相关联的代理实例的名称 (`instanceName`) 标识（请参见附录 A “`Message Queue` 数据的位置”）：

```
.../instances/instanceName/log/
```

创建新的转移日志文件时，将保留最新的 9 个日志文件的归档文件。

有关配置日志记录器的信息，请参见第 66 页的表 2-9 和第 134 页的“更改日志记录器配置”。

度量依据消息生成方（企业版）

消息生成方组件按一定的时间间隔接收度量依据生成器的信息，并将信息写入消息中，然后根据消息中包含的度量依据信息类型，将消息发送至多个度量依据主题目标之一。

共有五种度量依据主题目标，其名称显示在表 2-8 中，其中还有传送到每个目标的度量依据消息类型。

表 2-8 度量依据主题目标

主题目标名称	度量依据消息类型
<code>mq.metrics.broker</code>	代理度量依据
<code>mq.metrics.jvm</code>	Java 虚拟机度量依据
<code>mq.metrics.destination_list</code>	目标及其类型的列表
<code>mq.metrics.destination.queue. monitoredDestinationName</code>	指定名称队列的目标度量依据
<code>mq.metrics.destination.topic. monitoredDestinationName</code>	指定名称主题的目标度量依据

这些度量依据主题目标订阅的 Message Queue 客户机使用目标中的消息，并处理消息中包含的度量依据信息。例如，客户机可订阅 `mq.metrics.broker` 目标以接收和处理诸如代理中消息总数的信息。

度量依据消息生成方是一个内部 Message Queue 客户机，可创建包含与度量依据数据相对应的名称值对的消息（类型为 `MapMessage`）。仅当相应的度量依据主题目标有一个或多个订阅者时，才生成这些消息。

度量依据消息生成方生成的消息类型为 `MapMessage`；取决于其包含的度量依据类型，它会由很多名称 / 值对组成。每个名称 / 值对与度量依据数量及其值相对应。例如，代理度量依据消息包含多个度量依据数量的值，包括流入和流出代理的消息数量，当前在内存中的消息数量和大小，等等。有关每种类型的度量依据消息报告的度量依据数量的详细信息，请参见《*Message Queue Java Client Developer's Guide*》，其中介绍了如何写入使用度量依据消息的 Message Queue 客户机。

除了包含在度量依据消息主体中的度量依据信息之外，每条消息的标题有两个属性：一个指定度量依据消息类型，一个记录时间戳。这些标题属性可由 Message Queue 客户机应用程序用于从度量依据消息中提取数据，并记录其相应的时间戳。

监视服务属性

表 2-9 所示为用于通过代理设置信息的生成、记录和度量依据消息生成的可配置属性。（有关配置这些属性的说明，请参见第 5 章“启动与配置代理”）

表 2-9 监视服务属性

属性名称	说明
<code>imq.metrics.enabled</code>	指定 (true/false) 是否要将度量依据信息写入日志记录器。不影响度量依据消息的生成（请参见 <code>imq.metrics.topic.enabled</code> ）。 默认值: true
<code>imq.metrics.interval</code>	如果启用了度量依据记录 (<code>imq.metrics.enabled=true</code>)，指定将度量依据信息写入日志记录器的时间间隔（以秒为单位）。值 -1 表示从不。不影响度量依据消息的生成时间间隔（请参见 <code>imq.metrics.topic.interval</code> ）。 默认值: -1
<code>imq.log.level</code>	指定日志记录器级别：可以写入到输出通道的输出的种类。包括指定的种类以及所有更高级别的种类。分别为（由高至低）：ERROR、WARNING 和 INFO。 默认值: INFO
<code>imq.log.file.output</code>	指定写入到日志文件的日志记录信息的种类。允许的值为：由竖线 () 分隔的任意日志记录种类集，或 ALL 和 NONE。 默认值: ALL

表 2-9 监视服务属性 (续)

属性名称	说明
imq.log.file.dirpath	指定包含日志文件的目录的路径（根据操作系统）。 默认值：请参见附录 A “Message Queue 数据的位置”
imq.log.file.filename	指定日志文件的名称。 默认值：log.txt
imq.log.file.rolloverbytes	指定日志文件的大小（以字节为单位），达到该大小后输出将转移到新的日志文件。值 -1 表示不进行基于文件大小的转移。 默认值：-1
imq.log.file.rolloversecs	指定日志文件的生存期（以秒为单位），达到该值后输出将转移到新的日志文件。值 -1 表示不进行基于文件生存期的转移。 默认值：604800（一周）
imq.log.console.output	指定将哪些种类的日志记录信息写入到控制台。允许的值：由竖线 () 分隔的任意日志记录种类集，或 ALL 和 NONE。 默认值：ERROR WARNING
imq.log.console.stream	指定是否将控制台输出写入到标准输出 (OUT) 或标准错误 (ERR)。 默认值：ERR
imq.log.syslog.facility	（仅对于 Solaris）指定 Message Queue 代理应将 syslog 记录为哪种系统日志工具。属性值镜像 syslog(3C) 手册页中列出的值。适用于 Message Queue 的值包括：LOG_USER、LOG_DAEMON 和 LOG_LOCAL0 到 LOG_LOCAL7。 默认值：LOG_DAEMON
imq.log.syslog.logpid	（仅对于 Solaris）指定 (true/false) 是否将代理进程 ID 与消息一起记录。 默认值：true
imq.log.syslog.logconsole	（仅对于 Solaris）指定 (true/false) 如果无法将消息发送到 syslog，是否将其写入到系统控制台。 默认值：false
imq.log.syslog.identity	（仅对于 Solaris）此属性指定的标识字符串应添加到每个记录到 syslog 的消息的前面。 默认值：imqbrokerd_ 代理实例名称。
imq.log.syslog.output	（仅对于 Solaris）指定将哪些种类的日志记录信息写入到 syslogd(1M)。允许的值：由竖线 () 分隔的任意日志记录种类，或 ALL 和 NONE。 默认值：ERROR

表 2-9 监视服务属性 (续)

属性名称	说明
imq.log.timezone	指定日志时间戳的时区。标识符与 java.util.TimeZone.getTimeZone() 使用的一致。例如: GMT、America/LosAngeles、Europe/Rome 和 Asia/Tokyo。 默认值: 本地时区
imq.metrics.topic.enabled	指定 (true/false) 是否启用度量依据消息生成。如果为 false, 则试图订阅度量依据主题目标时会抛出客户端异常。 默认值: true
imq.metrics.topic.interval	指定生成度量依据消息 (发送到度量依据主题目标) 的时间间隔 (以秒为单位)。 默认值: 60
imq.metrics.topic.persist	指定 (true/false) 度量依据消息是否为持久性。 默认值: false
imq.metrics.topic.timetolive	指定生成度量依据消息 (发送到度量依据主题目标) 的生命周期 (以秒为单位)。默认值: 300

物理目标

Message Queue 消息传送基于两个传送阶段: 首先, 消息从生成方客户机传送到代理上的目标, 然后再从代理上的目标传送到一个或多个使用方客户机。目标分两种类型 (请参见第 42 页的“编程域”): 队列 (点对点传送模型) 和主题 (发布 / 订阅传送模型)。这些目标表示代理的物理内存中的位置 (外来消息在被路由到使用方客户机前将保留在其中)。

可以使用 Message Queue 管理工具创建物理目标 (请参见第 151 页的“获得连接信息”)。也可以按照第 70 页的“自动创建的 (与管理员创建的相对) 目标”中的介绍来自动创建目标。

本节介绍这两种类型的物理目标 (队列和主题) 的属性和行为。

队列目标

队列目标用于点对点消息传送。在这种消息传送中, 会有多个使用方在目标中注册请求, 但消息最终仅传送到其中的一个使用方。当来自消息生成方的消息到达时, 它们将排成队列的形式, 然后发送到单个消息使用方。

多个使用方的队列传送

当队列目标中的任何消息仅传送到单个使用方时，Message Queue 允许多个使用方向队列注册。代理会将外来消息路由至其他使用方，在使用方中平衡负荷。

对多个使用方的队列传送的实现是根据以下队列目标属性使用可配置的负荷平衡的方法：

- `maxNumActiveConsumers`：指定在负荷平衡队列传送中活动的使用方的数量（一个或多个）
- `maxNumBackupConsumers`：指定当任何活动使用方失败时可代替活动使用方的备份使用方的数量（无或多个）。

如果使用方数量超过这两个属性的和，将拒绝新的使用方。（Message Queue 平台版支持每个队列多达 3 个使用方（2 个活动和 1 个备份），而 Message Queue 企业版支持无限个使用方。）

负荷平衡机制考虑了不同使用方的消息使用率。其工作方式如下：

队列目标中的消息按可配置大小的批次（队列目标的 `consumerFlowLimit` 属性）路由至新的可用活动使用方（以便向队列注册）。这些消息传送之后，其他到达此队列的消息将在使用方可用时（即，当使用方先前已使用了传送给他们的可配置百分比的消息）按批次路由至使用方。每个使用方的发送率取决于使用方的当前容量和消息处理速率。

当消息生成速率较低时，代理可在活动使用方之间不平衡地发送消息。如果活动使用方多于所需个数，其中的一些可能永远收不到消息。

如果活动使用方失败，则第一个备份使用方变为活动，并取代失败使用方的任务。当队列目标中有多于一个活动使用方时，消息使用的顺序是不一定的。

在代理群集环境中，对多个使用方的传送可设置为优先考虑本地使用方。队列目标属性 `localDeliveryPreferred` 使您可指定仅当生成方的主代理（即，生成方将消息发送到的代理，本地代理）中无使用方时，才将消息传送到远程使用方。这可在路由到远程使用方（通过其主代理）可能导致流量降低时提升性能。（此属性需要目标范围未限制为仅本地传送。请参见第 154 页的表 6-10。）

内存的注意事项

由于消息可以在队列中保留相当长的一段时间，因此内存资源可能会成为一个问题。为队列分配内存时，您一定希望分配得恰到好处，向一个队列分配过多内存会导致内存不足，而太少的话消息就会被拒绝。出于灵活性的考虑，可以基于每个队列的负荷要求在创建队列时设置物理属性：队列消息的最大数量、分配给队列消息的最大内存和队列消息的最大大小（请参见第 154 页的表 6-10）。

主题目标

主题目标用于发布 / 订阅消息传送，在该消息传送中，消息最终将被传送到所有在目标中注册请求的使用方。当来自生成方的消息到达时，它们将被路由到所有订阅该主题的使用方。如果使用方已注册为长期订阅该主题，则当消息被传送到主题时并不需要这些使用方必须处于活动状态。代理将存储此消息，直到使用方再次处于活动状态，然后将消息传送给它。

通常，消息不会在主题目标中保留过长的时间，因此，内存资源通常也不是大问题。不过，您可以为目标收到的任意消息配置允许的最大大小（请参见第 154 页的表 6-10）。

自动创建的（与管理员创建的相对）目标

Message Queue 消息服务器是消息传送系统的核心，因此其性能和可靠性对企业应用程序的正常运行至关重要。因为目标可能使用大量的资源（取决于它们处理的消息的数量和大小以及注册的消息使用方的数量和长期性），所以需要它们严格地进行管理，以确保消息服务器的性能和可靠性。因此为应用程序创建目标、监视目标以及根据需要重新配置它们的资源要求已成为管理员的标准做法。

不过，有些时候可能需要动态创建目标。例如，在开发和测试阶段，可能希望代理根据需要自动创建目标，而不需要管理员进行干预。

Message Queue 支持这种 *自动创建* 功能。启用自动创建后，每当 MessageConsumer 或 MessageProducer 尝试访问不存在的目标时，代理会自动创建一个目标。（客户机应用程序的用户必须拥有自动创建特权，请参见第 197 页的“目标自动创建访问控制”。）

不过，当自动（而非明确）创建目标时，可能会导致不同客户机应用程序（使用相同的目标名称）之间发生冲突，或者导致系统性能降低（由于支持目标需要使用资源）。因此，当不再使用自动创建的目标时，也就是当目标不再与消息使用方客户机保持连接且不再包含任何消息时，代理会自动将其销毁。重新启动代理后，只有当自动创建的目标包含持久性消息时，才会重新创建这些目标。

可以使用表 2-10 中所示的属性来配置 Message Queue 消息服务器，以便启用或禁用自动创建功能。（有关配置这些属性的说明，请参见第 5 章“启动与配置代理”）

表 2-10 自动创建配置属性

属性名称	说明
imq.autocreate.topic	指定 (true/false) 是否允许代理自动创建主题目标。 默认值: true
imq.autocreate.queue	指定 (true/false) 是否允许代理自动创建队列目标。 默认值: true

表 2-10 自动创建配置属性 (续)

属性名称	说明
imq.autocreate.destination. maxNumMsgs	指定自动创建的目标中允许的未使用消息的最大数量。 默认值: 100,000
imq.autocreate.destination. maxTotalMsgBytes	指定目标中允许的内存的最大总量 (以字节为单位)。 默认值: 10m (兆字节)
imq.autocreate.destination. limitBehavior	指定当达到内存限制的阈值时代理响应的方式。允许的值为: FLOW_CONTROL — 减慢生成方 REMOVE_OLDEST — 丢弃最旧的消息 REMOVE_LOW_PRIORITY — 根据消息的生存期丢弃优先级最低的消息 REJECT_NEWEST — 拒绝最新的消息 默认值: REJECT_NEWEST
imq.autocreate.destination. maxBytesPerMsg	指定自动创建的目标中允许的任何单个消息的最大大小 (以字节为单位)。 默认值: 10k (10,240)
imq.autocreate.destination. maxNumProducers	指定目标允许的生成方的最大数量。当达到此限制时, 将无法创建新的生成方。 默认值: 100
imq.autocreate.destination. isLocalOnly	仅适用于代理群集。指定目标不能在其他代理上复制, 因而将消息传送限制为本地使用方 (连接到创建目标的代理的使用方)。在创建目标之后此属性无法更新。 默认值: false
imq.autocreate.queue. maxNumActiveConsumers	指定在从自动创建的队列目标的负荷平衡传送中可以处于活动状态的最大使用方数。值为 -1 表示不限制数量。 默认值: 1
imq.autocreate.queue. maxNumBackupConsumers	指定在从自动创建的队列目标的负荷平衡传送期间出现任何错误时, 可以代替这些活动使用方的最大备份使用方的数量。值为 -1 表示不限制数量。 默认值: 0

表 2-10 自动创建配置属性 (续)

属性名称	说明
imq.autocreate.queue. consumerFlowLimit	指定将要在一批中传送给使用方的消息的最大数量。在负荷平衡队列传送中，为负荷平衡开始之前路由至活动使用方的队列消息的初始数量（请参见第 69 页的“多个使用方的队列传送”）。此限制可被为目标使用方在其各自的连接中设置的较低的值覆盖（请参见《Message Queue Java Client Developer's Guide》中的“连接工厂”属性中的信息）。值为 -1 表示不限制数量。 默认值：1000
imq.autocreate.topic. consumerFlowLimit	指定将要在一批中传送给使用方的消息的最大数量。值为 -1 表示不限制数量。 默认值：1,000
imq.autocreate.queue. localDeliveryPreferred	仅应用于代理群集中的负荷平衡队列传送。指定仅当在本地代理中没有使用方时才将消息传送到远程的使用方。要求未将自动创建的目标限制为仅本地传送 (isLocalOnly = false)。 默认值：false

临时目标

某些客户机需要一个目标来接收对发送至其他客户机的消息的回复。这些客户机使用 JMS API 明确创建和销毁的目标称为临时目标。这些目标是为连接而创建的，并由代理维护，而且仅在连接期间存在。临时目标无法由管理员销毁，而且只要此目标在使用中（即连接有活动的消息使用方），那么它也无法由客户机应用程序销毁。临时目标与管理员创建或自动创建的目标不同（后两者包含持久性消息），它们不会被持久保存，并且在重新启动代理时也不会重新创建临时目标，但是，可通过 Message Queue 管理工具看到这些内容（请参见第 152 页的表 6-9）。

多代理群集（企业版）

Message Queue 企业版使用多个互连的代理实例（代理群集）来支持消息服务器的实现方案。群集支持为消息服务器提供了可伸缩性。

随着连接到代理的客户机数量以及所传送的消息数量的不断增加，代理最终将超出资源限制（例如文件描述符和内存限制）。适应不断增加的负荷的方法之一是将更多的代理（即更多代理实例）添加到 Message Queue 消息服务器，从而将客户机连接和消息传送分布到多个代理。

也可以使用多代理优化网络带宽。例如，您可能希望在一组远程代理之间使用低速的长途网络链接，而在客户机与其各自的代理实例之间使用高速链接进行连接。

虽然使用代理群集还有其他原因（例如，适应具有不同用户信息库的工作组或处理防火墙限制），但故障转移并不是其中的一个原因。当 Message Queue 允许失败的连接使用群集中的其他代理重新建立连接时，将丢失状态信息。因此，群集中的某个代理不能用作其他发生故障的代理的自动备份。

换言之，Message Queue 目前不支持高可用性消息服务器。但是，您可以使用 Sun Cluster 软件和高可用性数据库提供代理故障转移。也可以将消息传送应用程序设计为使用多个代理来实现自定义的故障转移解决方案。

有关配置和管理代理群集的信息，请参见第 128 页的“使用群集（企业版）”。

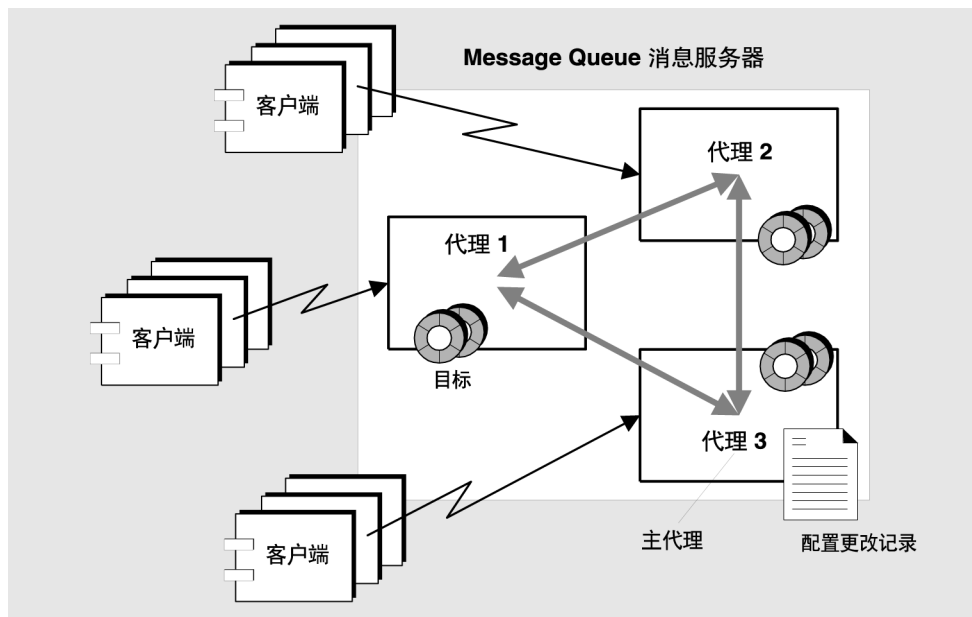
以下几个小节介绍了 Message Queue 代理群集的体系结构和内部功能。

多代理体系结构

多代理消息服务器允许将客户机连接分布在多个代理实例上，如图 2-7 中所示。从客户机的角度来看，每个客户机都连接到各自的代理（其主代理），发送和接收消息时就好像主代理是群集中唯一的代理。但从消息服务器的角度来看，主代理与群集中的其他代理串联起来进行工作，为主代理直接连接的消息生成方和使用方提供传送服务。

理论上，群集中的代理可以以任意拓扑结构连接。但 Message Queue 仅支持完全连接的群集，即在该拓扑结构中，每个代理都与群集中的其他所有代理直接连接，如图 2-7 所示。

图 2-7 多代理（群集）体系结构



消息传送

在多代理配置中，每个目标复制于群集中的所有代理上。（在大多数情况下，群集环境中的目标属性通常 *共同* 应用于目标的所有实例，即，应用于整个群集，而不是应用于单个目标实例。同时，`isLocalOnly` 属性设置为 `true` 的目标不会在群集中复制。有关详细信息，请参见第 154 页的表 6-10 中有关目标属性的说明。）

每个代理都了解向所有其他代理的目标进行注册的消息使用方。因此，每个代理既可以将消息从其直接连接的消息生成方路由到远程消息使用方，也可以从远程生成方将消息发送到其直接连接的使用方。

在群集配置中，与每个消息生成方直接连接的代理将对该直接相连的生成方发送给它消息执行路由。因此，持久性消息是通过消息的主代理进行存储和路由的。

要使群集中的代理之间的流量最小化，可通过使用方连接上的流控制机制来管理消息的传送（从目标到客户机运行时）。按这种方式，仅在代理要把消息传送给连接到目标代理上的使用方时，才把消息从一个代理发送到另一个代理，从而避免了代理之间不必要的消息传递。同时，在某些情况下（例如，到多个使用方的队列传送），可以指定到本地使用方的传送的优先级大于传送到远程使用方的优先级，从而使代理之间的流量最小化（请参见 `localDeliveryPreferred` 队列目标属性，第 154 页的表 6-10）。

当需要客户机和消息服务器之间的安全加密消息传送时，可将群集配置为同时保护群集中的代理之间的消息传送（请参见第 130 页的“安全的交叉代理连接”）。

群集同步

当管理员创建或销毁代理上的目标时，此信息会自动传播到群集中其他所有代理。同样，当消息使用方向其主代理进行注册，或当使用方与其主代理断开连接（无论是明确断开，还是因为客户机、网络故障或其主代理故障而断开），该使用方的相关信息都会被传播到整个群集。与此类似，长期订阅的相关信息也会被传播到群集中的所有代理。

注意

网络繁忙和 / 或消息较大可能会堵塞内部群集连接。该不断增加的延迟有时还会引起锁定协议超时错误。这样，客户机在尝试创建长期订阅者或将消息使用方排队时可能会引发异常。通常，使用高速连接可以避免这些问题。

将目标和消息使用方的相关信息传播至特定代理时，通常要求在共享资源中进行更改时该代理处于联机状态。如果进行这样的更改时代理处于脱机状态，那么将会发生怎样的情况呢（例如，代理崩溃、随后又重新启动，或者新代理被动态添加到群集）？

为了解决脱机的代理（或添加的新代理）带来的问题，Message Queue 会提供一份记录，其中包括对群集中所有持久性实体的更改，即所有已创建或已销毁的目标和长期订阅的记录。代理被动态添加到群集后，首先从此配置更改记录中读取目标和长期订阅者信息，当它进入联机状态后，将与其他代理交换当前活动使用方的相关信息。通过这些信息，新的代理将完全集成到群集中。

配置更改记录由群集中的一个代理（即指定为主管代理的代理）进行管理。因为主管代理是向群集动态添加代理的关键，所以应始终先启动此代理。如果主管代理未联机，则群集中的其他代理将无法完成初始化。

如果主管代理脱机，而其他代理又无法访问配置更改记录，则 Message Queue 将不允许在群集中传播目标和长期订阅。此时，如果尝试创建或销毁目标和长期订阅（或尝试大量相关操作，如重新激活长期订阅），将引发异常。

在关键任务的应用程序环境中，最好定期备份配置更改记录，以避免因记录意外损坏或主管代理故障带来的损失。可以使用 `imqbrokerd` 命令的 `-backup` 选项（此选项用于创建包含配置更改记录的备份文件，请参见第 125 页的表 5-2）来完成上述备份。然后可以使用 `-restore` 选项来恢复配置更改记录。

如果需要，可以更改用作主管代理的代理，更改方法为：备份配置更改记录，修改相应的群集配置属性（请参见第 128 页的表 5-3）以指定新的主管代理，然后使用 `-restore` 选项重新启动新的主管代理。

在开发环境中使用群集

在开发环境中，群集用于测试，而且可伸缩性和代理恢复 不是很重要，因此基本不需要主管代理。在配置为没有主代理的环境中，Message Queue 不再要求必须运行主管代理才能启动其他代理，并允许对目标和长期订阅进行更改，并且可以在群集的所有运行代理中传播更改。如果代理脱机、随后又恢复，它将不会与脱机期间所作的更改同步。

在测试环境下，通常自动创建目标（请参见第 70 页的“自动创建的（与管理员创建的相对）目标”），并且由正在测试的应用程序创建和销毁这些目标的长期订阅。目标和长期订阅中的这些更改将在整个群集中传播。但是，如果将环境重新配置为使用主管代理，则 Message Queue 将重新强制要求必须运行主管代理才能更改目标和长期订阅并在整个群集中传播这些更改。

群集配置属性

启动时，群集中的每个代理都必须接收到有关群集中其他代理的信息（主机名和端口号）。此信息用于在群集中的代理之间建立连接。另外，每个代理也必须知道主管代理（如果使用）的主机名和端口号。

群集中的所有代理应使用共同的群集配置属性集。为此，可以将它们放入一个中心群集配置文件，启动时每个代理均会引用该文件。

也可以复制群集配置属性，然后分别为每个代理提供这些属性。但最好不要采用这种做法，因为这会导致群集配置中出现不一致。只保留一份群集配置属性可以确保所有代理获得相同的信息。

有关群集配置属性的详细信息，请参见第 128 页的“使用群集（企业版）”。

群集配置文件可以用于存储一组代理通用的所有代理配置属性。虽然，它原本是用来配置群集的，但也可以用来存储群集中所有代理通用的其他代理属性。

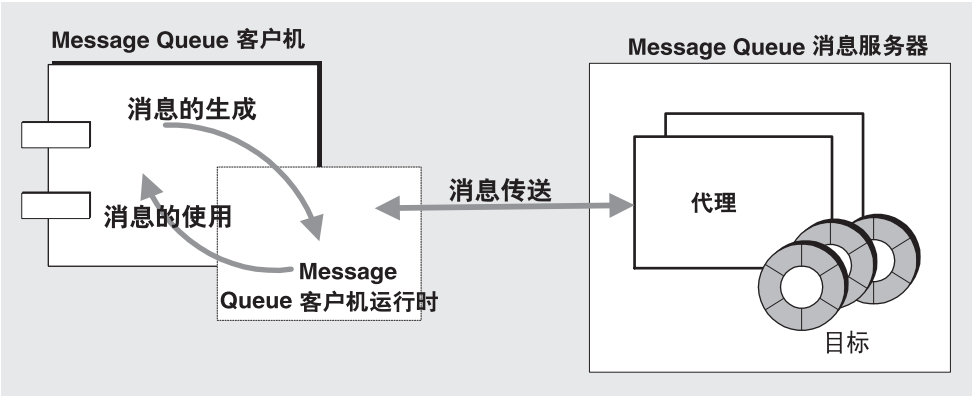
Message Queue 客户机运行时

Message Queue 客户机运行时向客户机应用程序提供至 Message Queue 消息服务的接口，即向 Java 客户机应用程序提供第 37 页的“JMS 编程模型”和具有相应的 C 接口的 C 客户机应用程序中介绍的所有 JMS 编程对象。Message Queue 客户机运行时支持客户机向目标发送消息以及从目标接收消息所需的所有操作。

本节提供 Message Queue 客户机运行时的工作方式的高级说明。影响客户机应用程序设计、Java 客户机运行时和 C 客户机运行时的性能的因素在《Message Queue Java Client Developer's Guide》和《Message Queue C Client Developer's Guide》中分别进行探讨。

图 2-8 说明了在消息的生成和使用中客户机应用程序如何与 Message Queue 客户机运行时之间进行交互以及在消息传送中 Message Queue 客户机运行时如何与 Message Queue 消息服务器之间进行交互。

图 2-8 消息传送操作



消息生成

在消息生成中，由客户机创建消息，然后通过连接将消息发送至代理中的目标。如果将 MessageProducer 对象的消息传送模式设置为持久性（有保证的传送，传送一次且仅传送一次），客户机线程将阻塞，直到代理确认消息已传送到其目标并存储在代理的持久性数据存储中为止。如果消息不是持久性的，则代理不会返回代理确认消息（在属性名称中称为 "Ack"），并且客户机线程不会阻塞。

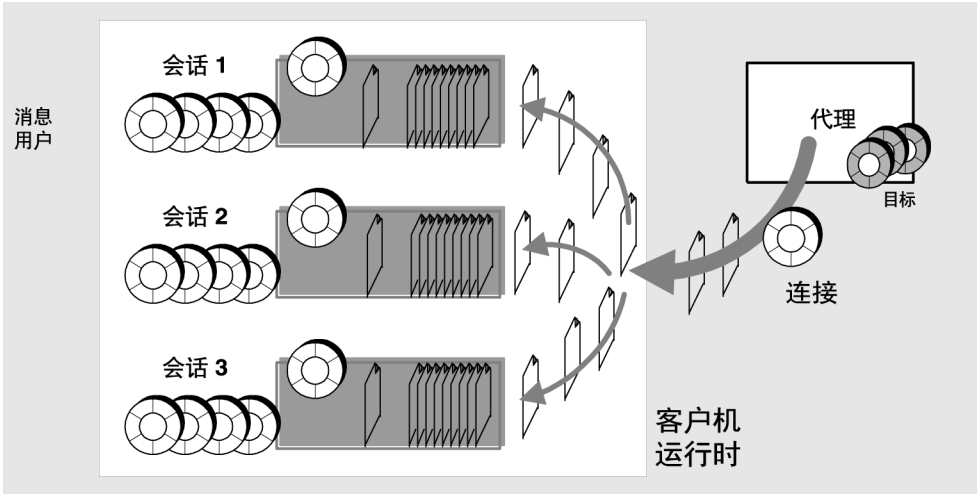
消息使用

消息的使用要比生成复杂得多。在以下条件下，将把到达代理中的目标的消息通过连接传送到 Message Queue 客户机运行时：

- 客户机已经设置了给定目标的使用方
- 使用方的选择条件（如果存在）与到达给定目标的消息的选择条件匹配
- 已告知连接开始传送消息。

通过连接传送的消息将分发到相应的 Message Queue 会话，在这些会话中，它们排队等待供相应的 MessageConsumer 对象使用，如图 2-9 所示。每次从每个会话队列中取出一条消息（会话是单线程的），然后同步使用（通过调用 receive 方法的客户机线程）或异步使用（通过调用 MessageListener 对象的 onMessage 方法的客户机线程）消息。

图 2-9 将消息传送到 Message Queue 客户机运行时



代理在向客户机运行时传送消息时，会对消息进行相应地标记，但无法实际得知这些消息是否已被接收或使用。因此，只有客户机确认接收到消息后，代理才会将消息从其目标中删除。

Message Queue 受管理对象

受管理对象可以使客户机应用程序代码与提供者无关。为此，它们需要将提供者特有的实现方案和配置信息封装在对象中，而客户机应用程序可以通过与提供者无关的方式使用这些对象。受管理对象由管理员创建和配置，并存储在命名服务中，由客户机应用程序通过标准 JNDI 查找代码来访问。

Message Queue 提供两种类型的受管理对象：ConnectionFactory 和 Destination。虽然两者都用于封存提供者特有的信息，但在客户机应用程序中，它们的用途却有很大的差异。ConnectionFactory 对象用于创建至消息服务器的连接，而 Destination 对象用于标识物理目标。

通过受管理对象，可以非常容易地控制和管理 Message Queue 消息服务器：

- 您可以通过要求客户机应用程序访问预配置的 `ConnectionFactory` 对象来控制连接的行为（请参见第 168 页的“[连接工厂受管理对象属性](#)”）。
- 您可以通过要求客户机应用程序访问与现有物理目标对应的预配置 `Destination` 对象来控制物理目标的创建。（您还需要禁用代理的自动创建功能，请参见第 70 页的“[自动创建的（与管理员创建的相对）目标](#)”。）
- 您可以通过覆盖由客户机应用程序设置的消息标题值来控制 Message Queue 消息服务器资源（请参见第 168 页的“[连接工厂受管理对象属性](#)”）。

因此，受管理对象使您（Message Queue 管理员）可以控制消息服务器配置的详细信息，同时又可以使客户机应用程序与提供者无关：它们不需要了解提供者特有的语法和对象命名惯例（请参见第 41 页的“[JMS 提供者无关](#)”）或提供者特有的配置属性。

您可以使用 Message Queue 管理工具创建受管理对象，如第 7 章“[管理受管理对象](#)”中所述。创建受管理对象时，可以将对象指定为只读，也就是说，不允许客户机应用程序更改 Message Queue 特有的配置值（即在创建对象时设置的值）。换句话说，客户机代码无法设置只读受管理对象的属性值，而且也无法使用客户机应用程序的启动选项覆盖这些值，如第 81 页的“[在客户机启动时覆盖属性值](#)”中所述。

虽然客户机应用程序可以独立地实例化 `ConnectionFactory` 和 `Destination` 这两种受管理对象，但这样会削弱受管理对象的基本用途（即使 Message Queue 管理员可以控制应用程序所需的代理资源以及调整其性能的用途）。另外，直接实例化受管理对象将使客户机应用程序成为提供者特有的应用程序，而无法实现与提供者无关。

连接工厂受管理对象

`ConnectionFactory` 对象用于建立客户机应用程序与 Message Queue 消息服务器之间的物理连接，还可用于指定连接的行为以及使用此连接访问代理的客户机运行时的行为。

如果要支持分布式事务（请参见第 44 页的“[本地事务](#)”），需要使用支持分布式事务的特殊的 `XAConnectionFactory` 对象。

要创建 `ConnectionFactory` 受管理对象，请参见第 176 页的“[添加连接工厂](#)”。

通过配置 `ConnectionFactory` 受管理对象，可以指定该对象生成的所有连接通用的属性值（属性）。`ConnectionFactory` 和 `XAConnectionFactory` 对象共享同一属性集。按照其影响的行为，这些属性可分为多个种类：

- 连接规范
- 自动重新连接行为
- 客户机标识
- 消息标题覆盖
- 可靠性和流控制
- 队列浏览器行为
- 应用程序服务器支持
- JMS 定义的属性支持

每个种类及其相应属性将在 《*Message Queue Java Client Developer's Guide*》 中进一步讨论。虽然作为 Message Queue 管理员，可能会要求您调整这些属性的值，但通常是由应用程序开发者来决定需要调整哪些属性以调节客户机应用程序的性能。[第 168 页的表 7-3](#) 按字母顺序概述了这些属性。

目标受管理对象

Destination 受管理对象表示代理中与公开命名的 Destination 对象对应的物理目标（队列或主题）。[表 2-11](#) 中列出了它的两个属性。通过创建 Destination 对象，可以使客户机应用程序的 MessageConsumer 和 / 或 MessageProducer 对象能够访问相应的物理目标。

要创建 Destination 受管理对象，请参见[第 177 页的“添加主题或队列”](#)。

表 2-11 目标属性

属性 / 属性名	说明
imqDestinationName	指定物理目标的供应商特有的名称。在创建物理目标时指定此名称。目标名称必须只包含字母数字字符（不包括空格），可以以字母字符或 "_" 和 "\$" 字符开头。它们不能以字符串 "mq." 开头。 默认值: Untitled_Destination_Object
imqDestinationDescription	指定用于管理对象的信息。 默认值: 目标对象的说明

在客户机启动时覆盖属性值

像使用任何 Java 应用程序一样，您可以使用命令行来启动消息传送应用程序以指定系统属性。此机制也可以用来覆盖客户机应用程序代码中使用的受管理对象的属性值。例如，可以覆盖在客户机应用程序代码中通过 JNDI 查找来访问的受管理对象的配置。

要在客户机应用程序启动时覆盖受管理对象的设置，请使用以下命令行语法：

```
java [-Dattribute=value ]... clientAppName
```

其中，*attribute* 与第 168 页的“[连接工厂受管理对象属性](#)”中记录的任意 `ConnectionFactory` 受管理对象的属性相对应。

例如，如果希望客户机应用程序连接到其他代理，而不是连接到在客户机代码中访问的 `ConnectionFactory` 受管理对象中指定的代理，可以使用命令行覆盖来启动客户机应用程序以设置另一个代理的 `imqBrokerHostName` 和 `imqBrokerHostPort`。

如果已将受管理对象设置为只读，则无法使用命令行覆盖来更改其属性的值。所有此类覆盖将被忽略。

Message Queue 管理任务和工具

Sun Java™ System Message Queue 管理由若干任务以及用于执行这些任务的若干工具组成。

本章首先概述了管理任务，然后介绍了管理工具，其中着重介绍命令行管理实用程序的通用特性。

Message Queue 管理任务

您需要执行哪些特定任务取决于您处于开发环境还是处于生产环境。

开发环境

在开发环境中，工作的重点是 Message Queue 客户机应用程序编程。Message Queue 消息服务器主要用于测试。在开发环境中，强调的是灵活性，您通常采用以下惯例：

- 您想做最少的管理工作，主要是为开发者启动代理以便用于测试。
- 您使用数据存储（基于内置文件持久性）、用户信息库（基于文件的用户信息库）、访问控制属性文件和对象存储（文件系统存储）的默认实现方案。利用这些默认的实现方案进行开发测试通常已经足够了。
- 如果是执行多代理测试，很可能不会使用主管代理。
- 您通常使用自动创建的目标，而不使用明确创建目标
- 您可能想在客户机代码中实例化受管理对象，而不使用集中管理的受管理对象。

生产环境

在生产环境中，由于必须在其中可靠地部署和运行应用程序，所以管理就重要得多了。需要执行的管理任务取决于消息系统的复杂性和它必须支持的应用程序的复杂性。不过，通常这些任务可分为设置操作和维护操作。

设置操作

► 设置生产环境

通常需要执行以下全部设置操作或至少执行某些设置操作：

- **管理员安全性**（保护管理工具的使用）：
 - 确保 **admin** 连接服务处于激活状态（请参见第 52 页的表 2-3）
 - 授权：允许针对特定的个人或 **admin** 组访问 **admin** 连接服务（请参见第 195 页的“连接访问控制”）
 - 如果针对某个组授权，请确保管理员属于该 **admin** 组。
 - 基于文件的用户信息库：有默认的 **admin** 组。确保管理员属于 **admin** 组，或者如果使用默认的 **admin** 用户，请更改 **admin** 密码（请参见第 189 页的“更改默认的管理员密码”）。
 - **LDAP** 用户信息库：确保管理员属于 **admin** 组
- **一般安全性**（请参见第 8 章“管理安全性”）：
 - 验证：创建进入基于文件的用户信息库的入口或将代理配置为使用现有 **LDAP** 用户信息库
(至少需要设置保护管理功能的密码。)
 - 授权：修改访问控制属性文件中的访问设置
 - 加密：设置基于 **SSL** 的连接服务
- **受管理对象**（请参见第 7 章“管理受管理对象”）：
 - 配置或设置 **LDAP** 对象存储
 - 创建连接工厂和目标受管理对象
- **代理群集**（请参见第 128 页的“使用群集（企业版）”）：
 - 创建中心配置文件
 - 使用主管代理

- **持久性：**将代理配置为使用插入的持久性，而不是内置的持久性（请参见附录 B “设置插入的持久性”）
- **内存管理：**设置目标属性，以便消息数和为消息分配的内存容量与可用的代理内存资源相适合（请参见第 154 页的表 6-10）

维护操作

► 设置生产环境

另外，在生产环境中，需要对 Message Queue 消息服务器资源进行严格的监视和控制。应用程序性能、可靠性和安全性是优先考虑的因素，并且需要使用 Message Queue 管理工具执行如下所述的若干任务：

- **应用程序管理：**
 - 禁用代理的自动创建功能（请参见第 70 页的表 2-10）
 - 代表应用程序创建物理目标（请参见第 154 页的“创建目标”）
 - 设置对目标的用户访问权限（请参见第 192 页的“授权用户：访问控制属性文件”）
 - 监视和管理目标（请参见第 152 页的“管理目标”）
 - 监视和管理长期订阅（请参见第 161 页的“管理长期订阅”）
 - 监视和管理事务（请参见第 162 页的“管理事务”）
- **代理管理和调整：**
 - 使用代理度量依据来调整和重新配置代理（请参见第 205 页上的第 9 章“分析和调整消息服务”）
 - 管理代理内存资源（请参见第 205 页上的第 9 章“分析和调整消息服务”）
 - 将代理添加到群集中以平衡负荷（请参见第 128 页的“使用群集（企业版）”）
 - 恢复出现故障的代理（请参见第 123 页的“启动代理”）
- **管理受管理对象：**
 - 按照需要创建其他连接工厂和目标受管理对象（请参见第 176 页的“添加和删除受管理对象”）
 - 调整连接工厂的属性值以提高性能和吞吐量（请参见第 168 页的“连接工厂受管理对象属性”和第 180 页的“更新受管理对象”）

Message Queue 管理工具

Message Queue 管理工具分为两类：命令行实用程序和图形用户界面 (GUI) 管理控制台 (imqadmin)。控制台结合了两个命令行实用程序的功能：命令行实用程序 (imqcmd) 和对象管理器实用程序 (imqobjmgr)。可以使用控制台（和这两个命令行实用程序）来远程管理代理以及管理 Message Queue 受管理对象。其他命令行实用程序 (imqbrokerd、imqusermgr、imqdbmgr 和 imqkeytool) 必须在其关联的代理的同一主机上运行，如第 87 页的图 3-1 所示。

可以从联机帮助中获得有关管理控制台的信息。“[命令行实用程序概述](#)”。介绍了通常用于执行特殊任务的命令行实用程序。

管理控制台

可以使用管理控制台执行以下任务：

- 连接到代理并对其进行管理。
- 在代理上创建和管理物理目标
- 连接到对象存储
- 向对象存储添加受管理对象并对其进行管理。

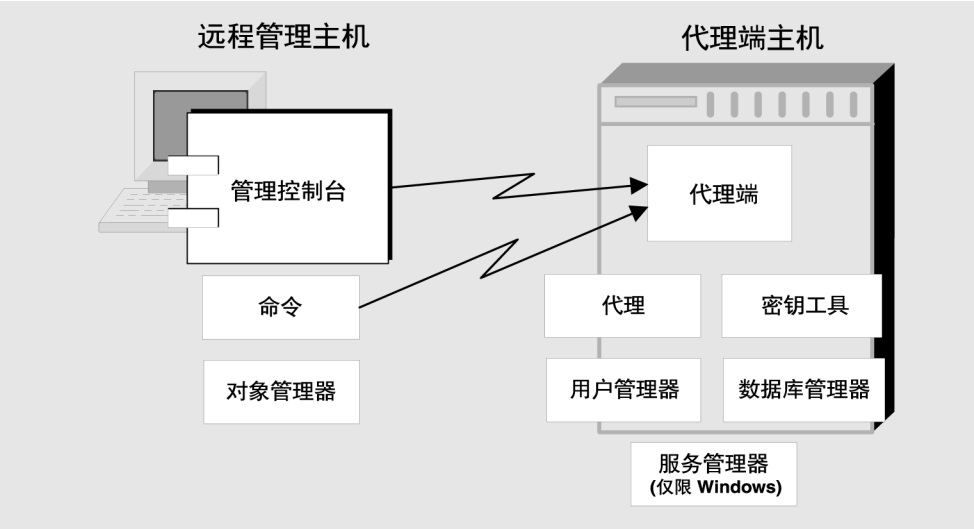
某些任务不能使用管理控制台来执行，主要有：启动代理、创建代理群集、配置代理更加具体的属性和物理目标以及管理用户数据库。

第 4 章 “[管理控制台教程](#)” 提供了一个简明扼要的实用教程，帮助您熟悉控制台，并说明了如何使用它完成基本任务。

命令行实用程序概述

本节介绍用于执行 Message Queue 管理任务的命令行实用程序。使用 Message Queue 实用程序启动和管理代理并执行其他更加具体的管理任务。

图 3-1 本地和远程管理实用程序



所有 Message Queue 实用程序均可通过命令行接口 (CLI) 来访问。实用程序命令共用通用格式、语法惯例和选项，本章下一小节将进行介绍。可以在后面的章节中找到有关使用命令行实用程序更为详细的信息。

代理 (imqbrokerd) 使用代理实用程序启动代理。使用 imqbrokerd 命令的选项来指定是否应将代理连接到群集中，并指定其他配置信息。第 5 章 “启动与配置代理” 中介绍了 imqbrokerd 命令。

命令 (imqcmd) 启动了代理后，您可以使用命令行实用程序创建、更新和删除物理目标；控制代理及其连接服务；管理代理的资源。第 6 章 “代理和应用程序管理” 中介绍了 imqcmd 命令。

对象管理器 (imqobjmgr) 使用对象管理器实用程序在可通过 JNDI 访问的对象存储中添加、列出、更新和删除受管理对象。通过将 JMS 客户机与 JMS 提供者特有的命名和配置格式分离，受管理对象使 JMS 客户机可以与提供者无关。第 7 章 “管理受管理对象” 中介绍了 imqobjmgr 命令。

用户管理器 (imqusermgr) 使用用户管理器实用程序来填充用于验证和授权用户的基于文件的用户信息库。第 8 章 “管理安全性” 中介绍了 imqusermgr 命令。

密钥工具 (imqkeytool) 使用密钥工具实用程序来生成用于 SSL 验证的自签名证书。第 8 章 “管理安全性” 和附录 C “HTTP/HTTPS 支持 (企业版)” 中介绍了 imqkeytool 命令

数据库管理器 (imqdbmgr) 使用数据库管理器实用程序来创建和管理用于持久性存储器的 JDBC 兼容数据库。附录 B “设置插入的持久性” 中介绍了 imqdbmgr 命令。

服务管理器 (imqsvcadm) 使用服务管理器实用程序可以将代理作为 Windows 服务来进行安装、查询和删除。附录 D “使用代理作为 Windows 服务” 中介绍了 imqsvcadm 命令。

命令行语法

Message Queue 命令行接口实用程序是简单的 shell 命令。也就是说，从 Windows、Linux 或 Solaris 命令 shell 的角度来看，命令行接口实用程序的名称本身是一个命令，而它的子命令或选项只是要传送到该命令的变量而已。因此，没有、也不需要用于启动或退出实用程序的命令。

所有的命令行实用程序均共享以下命令语法：

```
Utility_Name [subcommand] [argument] [[-option_name [-option_argument]]...]
```

Utility_Name 指定 Message Queue 实用程序的名称，例如，imqcmd、imqobjmgr 和 imqusermgr 等。

以下四点需要注意：

- 在子命令之后指定选项（如果此实用程序还接受变量类型的操作数，则还包括变量）。
- 如果变量包含空格，请将整个变量置于引号中。将属性 - 值对置于引号中通常是最安全的做法。
- 如果在命令行中指定了 -v（版本）或 -h/-H（帮助）选项，所有此命令行中的其他选项均不执行。有关通用选项的说明，请参见第 89 页的表 3-1。
- 用空格分隔子命令、变量、选项和选项变量。

以下是不包含子命令子句的一个命令行示例。此命令启动默认代理。

```
imqbrokerd
```


以下命令要稍微复杂一些：它将为用户名为 admin 且相应密码为 admin 的管理员（用户）销毁一个名为 myQueue 的 queue 类型的目标，此销毁不需要确认，并且控制台上不显示输出。

```
imqcmd destroy dst -t q -n myQueue -u admin -p admin -f -s
```

通用命令行选项

表 3-1 介绍了通用于所有 Message Queue 管理实用程序的选项。除了要在命令行中指定的子命令之后指定下述选项的要求以外，对这些选项（或任何传送至实用程序的其他选项）的输入顺序无任何特殊要求。

表 3-1 通用 Message Queue 命令行选项

选项	说明
-h	显示指定实用程序的使用帮助
-H	显示详细的使用帮助，包括属性列表和示例（只支持 imqcmd 和 imqobjmgr）。
-s	打开无提示模式：不显示输出。对于 imqbrokerd，指定为 -silent。
-v	显示版本信息。
-f	执行给定操作，而不提示用户进行确认。
-pre	（仅用于 imqobjmgr）打开预览模式，允许用户查看命令行中其他部分的执行效果而无需实际执行此命令。此项功能有助于检查默认属性值。
-javahome <i>path</i>	指定要使用的替代 Java 2 兼容运行时（默认使用系统上的运行时或 Message Queue 附带的运行时）。

管理控制台教程

本教程重点介绍如何使用管理控制台—管理 Message Queue 消息服务器的图形界面。跟随本教程，您将学习如何进行以下操作：

- 启动代理，使用控制台连接到代理并对其进行管理。
- 在代理上创建物理目标。
- 创建对象存储，并使用控制台连接到对象存储
- 将被管理对象添加到对象存储

本教程旨在设置运行简单 JMS 兼容应用程序 HelloWorldMessageJNDI 所需的目标和被管理的对象， HelloWorldMessageJNDI 应用程序可以在示例应用程序 /demo 目录的 helloworld 子目录中找到（请参见[附录 A “Message Queue 数据的位置”](#)）。在本教程的最后一部分，您将运行此应用程序。

本教程主要用来指导您通过使用管理控制台来执行基本管理任务。即使了解了本教程，您还是需要仔细阅读《Message Queue Java Client Developer's Guide》或本《*管理指南*》的其他章节。

某些 Message Queue 管理任务无法使用图形工具来完成，您将需要使用命令行实用程序执行如下所示的这些任务：

- 配置某些物理目标属性
某些物理目标属性无法使用管理控制台来配置。可以按照标题为[第 152 页的“管理目标”](#)的一节中的说明配置这些属性。
- 创建代理群集
有关详细信息，请参见[第 128 页的“使用群集（企业版）”](#)。
- 管理用户数据库
有关详细信息，请参见[第 184 页的“验证用户”](#)。

准备工作

必须安装 Message Queue 产品才能启动本教程。有关详细信息，请参见《*Message Queue 安装指南*》。请注意，本教程以 Windows 为中心，并为 UNIX® 用户添加了说明。

在本教程中，选择 "Item1" > "Item2" > "Item3" 表示您应该下拉名为 "Item1" 的菜单，从该菜单选择 "Item2"，然后从 "Item2" 提供的选择中选择 "Item3"。

启动管理控制台

管理控制台是您用于完成以下操作的图形工具：

- 创建对代理的引用并连接到代理
- 管理代理
- 在代理上创建物理目标，代理将使用它来进行消息传送
- 连接到放置 Message Queue 被管理对象的对象存储

被管理对象使您可以管理 JMS 兼容应用程序的消息传送需要。有关详细信息，请参见第 78 页的“[Message Queue 受管理对象](#)”。

► 启动管理控制台

1. 选择 “开始” > “程序” > "Sun Java System Message Queue 3.5 SP1" > “管理”。

显示控制台窗口之前，您可能需要等待几秒钟。

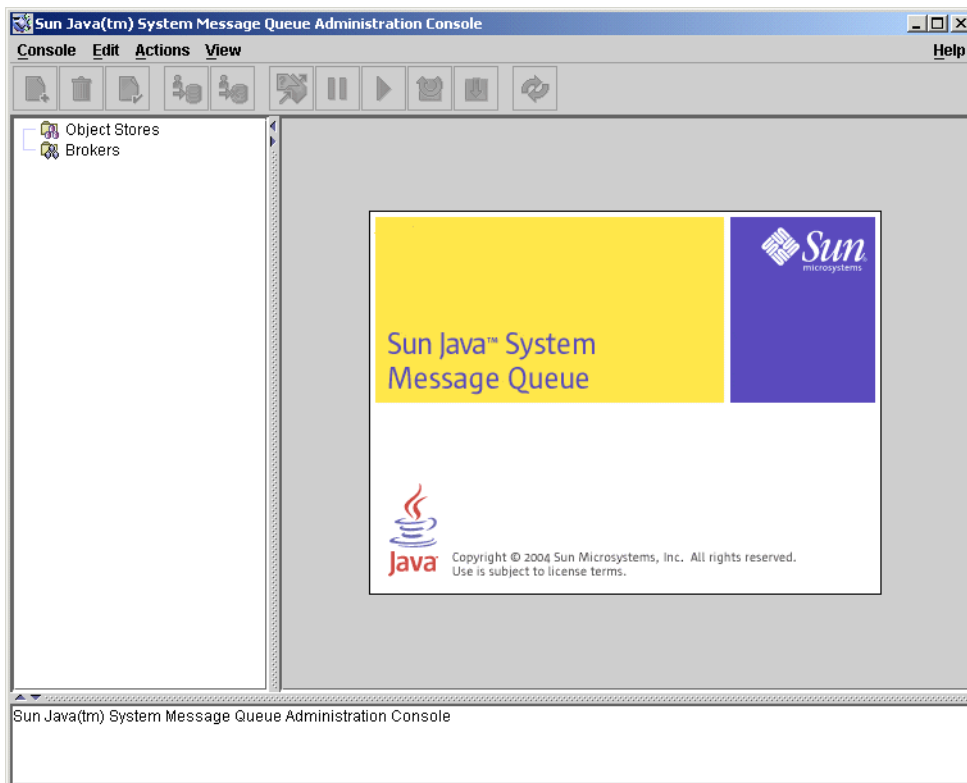
非 Windows 用户：在命令提示符下输入以下命令：

`/usr/bin/imqadmin` （在 Solaris 操作系统上）

`/opt/imq/bin/imqadmin` （在 Linux 操作系统上）

2. 请花几秒钟仔细观察一下控制台窗口。

控制台包括位于顶部的菜单栏、菜单栏下面的工具栏、左侧的浏览窗格、右侧的结果窗格（现在显示标识 Sun Java System Message Queue 产品的图形）和底部的状态窗格。



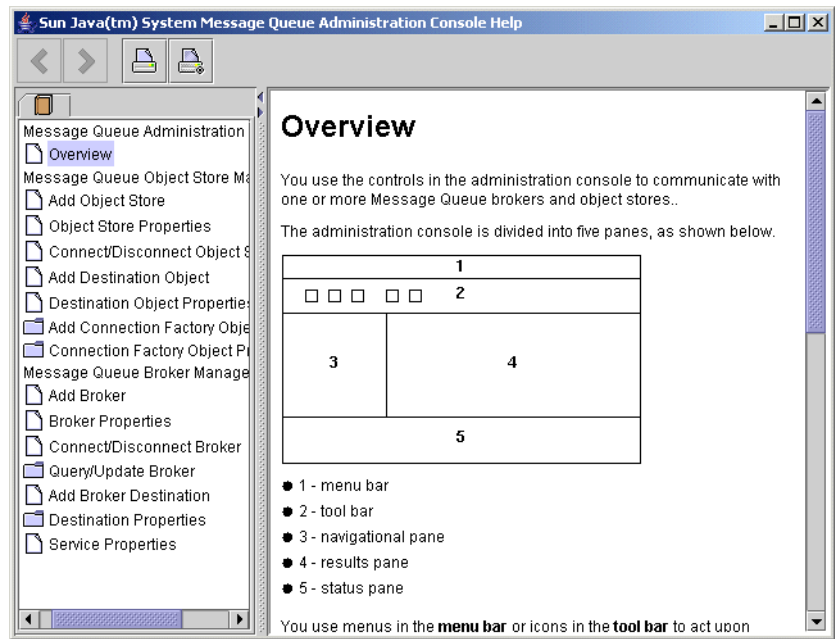
任何教程都不可能提供全部的信息，因此让我们首先找出如何获得管理控制台的帮助信息。

获得帮助

在菜单栏的最右侧找到“Help”菜单。

► 显示管理控制台帮助信息

1. 下拉“Help”菜单并选择“Overview”。将显示一个帮助窗口。



请注意帮助信息是如何组织的。左侧的浏览窗格显示目录，右侧的结果窗格显示在浏览窗格中选择的任何项的内容。

观察 “Help” 窗口的结果窗格。它显示管理控制台的概貌，并标识了每个控制台窗格的用途。

- 2. 观察 “Help” 窗口的浏览窗格。它使用以下三个区域来组织主题：概述、对象存储管理和代理管理。其中每个区域都包含文件和文件夹。每个文件夹提供的是包含多个选项卡的对话框的帮助，每个文件则提供简单对话框或选项卡的帮助。

您的第一个控制台管理任务（请参见第 96 页的“添加代理”）将是通过控制台创建对所管理的代理的引用。但是，启动之前请查阅联机帮助以获得有关信息。

- 3. 单击 “Help” 窗口的浏览窗格中的 “Add Broker” 项。

您会发现结果窗格发生了变化。它现在包含一些文本，说明添加代理的含义并介绍 “Add Broker” 对话框中每个字段的使用。字段名称以粗体文本显示。

- 4. 仔细阅读帮助文本。
- 5. 关闭 “Help” 窗口。

使用代理

代理为 Message Queue 消息传送系统提供传送服务。消息传送的过程可分为两个阶段：首先将消息传送到代理上的物理目标，然后将其传送给一个或多个使用方客户机。

使用代理涉及以下任务：

- 启动和配置代理

可以从 Windows 上的“开始”>“程序”菜单或使用 `imqbrokerd` 命令启动代理。如果使用 `imqbrokerd` 命令，可以使用命令行选项指定代理配置信息。如果使用“程序”菜单，可以使用控制台和第 5 章“启动与配置代理”。中说明的其他方法来指定配置信息。

注意

使用管理控制台无法启动代理实例。

- 使用管理控制台或使用 Command 命令行实用程序 (`imqcmd`) 管理代理及其服务。
- 创建客户机应用程序所需的物理目标
- 监视资源使用以提高吞吐量和可靠性

代理支持与应用程序客户机和管理客户机之间的通信。它利用不同的连接服务实现此功能，并且您可以将代理配置为运行这些服务之一或全部运行。有关连接服务的详细信息，请参见第 50 页的“连接服务”。

启动代理

使用管理控制台无法启动代理。按照以下过程中的说明启动代理（另请参见第 5 章“启动与配置代理”）。

► 启动代理

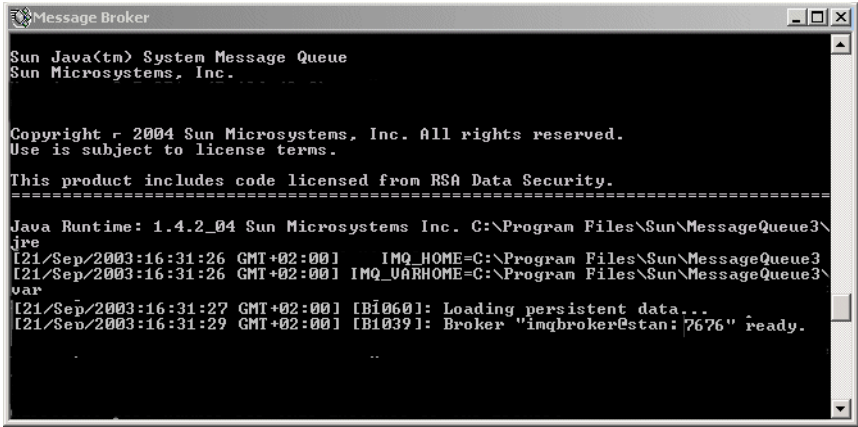
1. 选择“开始”>“程序”>“Sun Java System Message Queue 3.5 SP1”>“Message Broker”。

非 Windows 用户：输入以下命令启动代理。

`/usr/bin/imqbrokerd` （在 Solaris 操作系统上）

`/opt/imq/bin/imqbrokerd` （在 Linux 操作系统上）

显示出命令提示符窗口，表明代理就绪。



2. 现在让我们把注意力转回到 “Administration Console” 窗口。您现在可以将代理添加到控制台并连接到代理。

在管理控制台中添加对代理的引用之前不必启动代理，但是必须启动代理才能连接到它。

添加代理

添加代理在管理控制台中创建了一个对该代理的引用。添加代理之后，您就可以连接到它。

► 将代理添加到管理控制台

1. 在浏览窗格的 “Brokers” 上单击右键并选择 “Add Broker”。
2. 在 “Broker Label” 字段中输入 MyBroker。

这样便提供了一个在管理控制台中标识代理的标签。

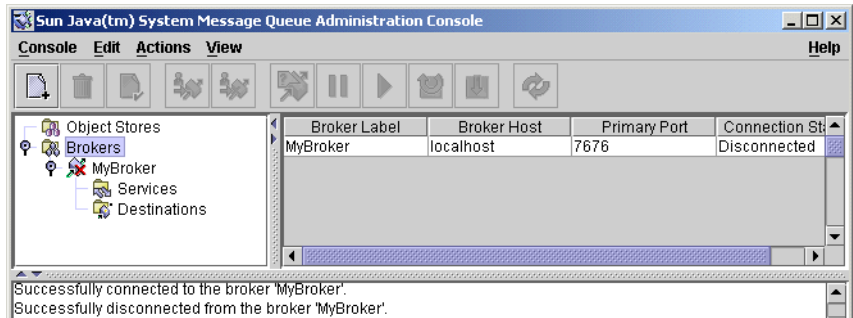


注意，在对话框中已指定默认主机名 (localhost) 和默认主端口 (7676)。这些值是以以后需要指定的值，即在您配置客户机用来建立与此代理的连接的连接工厂时需要指定这些值。

将“Password”字段保留为空。如果在连接时指定密码，则密码将会更安全。

3. 单击“OK”以添加代理。

观察浏览窗格。“Brokers”下方应已列出刚才添加的代理。代理图标上的红色 X 表示该代理当前未连接到控制台。



4. 在“MyBroker”上单击右键，并从弹出式菜单中选择“Properties”。

将显示代理属性对话框。可以使用此对话框更新添加代理时指定的任意属性。

5. 单击“Cancel”关闭对话框。

更改管理员密码

当连接到代理时，如果添加代理时未指定密码，系统将提示您指定密码。默认情况下，管理控制台可以作为用户 `admin` 并使用密码 `admin` 连接到代理。为了提高安全性，建议您在连接之前更改默认管理员密码 (`admin`)。

► 更改管理员密码

1. 打开命令提示符窗口。如果窗口已打开，则请继续下一步操作。
2. 输入如下命令，并用自己的密码代替 `abracadabra`。然后，您指定的密码将代替 `admin` 的默认密码。

```
imqusermgr update -u admin -p abracadabra
```

更改将立即生效。每当使用某个 **Message Queue** 命令行实用程序或管理控制台时，必须指定新密码。

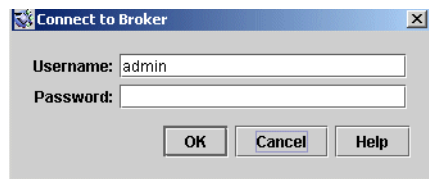
尽管客户机使用不同于管理员的连接服务，但是也为您分配了默认用户名和密码，这样不必进行大量的管理设置便可以测试 **Message Queue**。默认情况下，客户机可以作为用户 `guest` 并使用密码 `guest` 连接到代理。但是，应尽快建立客户机的安全用户名和密码。有关详细信息，请参见第 184 页的“验证用户”。

连接到代理

► 连接到代理

1. 在 "MyBroker" 上单击右键并选择 “Connect to Broker”。

将显示一个对话框，您可以在其中指定用户名和密码。



2. 在 “Password” 字段中输入 `admin` 或在第 98 页的“更改管理员密码”中指定的任何密码值。

指定用户名 `admin` 并提供正确密码将使您以管理员权限连接到代理。

3. 单击 “OK” 连接到代理。

连接到代理后，可以从 “Actions” 菜单中选择各个菜单项，以获得有关代理的信息、暂停和恢复代理、关闭和重新启动代理以及从代理断开连接。

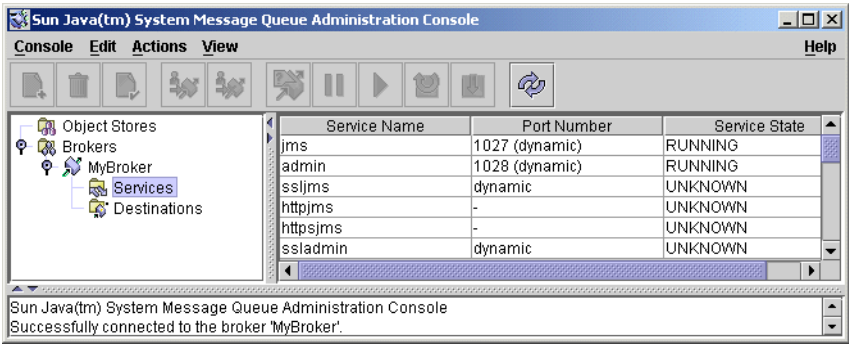
查看连接服务

从代理提供的连接服务及其支持的物理目标可以区分代理。

► 查看可用连接服务

1. 在浏览窗格中选择 “Services”。

在结果窗格中列出了可用服务。对于每项服务，均提供了其名称、端口号和状态。



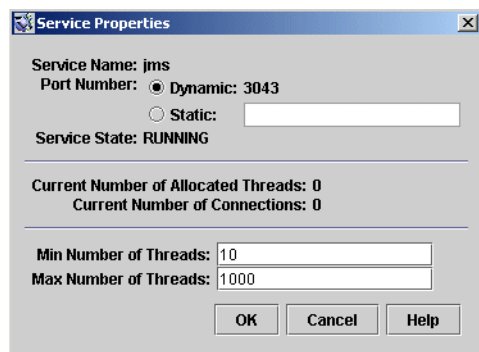
2. 单击结果窗格中的 "jms" 服务，以选中它。

3. 下拉 “Actions” 菜单并注意高亮显示的项。

您可以选择暂停 jms 服务或查看和更新其属性。

4. 从 “Actions” 菜单中选择 “Properties”。

请注意，在 “Service Properties” 对话框中，您可以分配服务的静态端口号并可以更改分配给此服务的最小和最大线程数。



5. 单击 “OK” 或 “Cancel” 关闭 “Properties” 对话框。
6. 在结果窗格中选择 “admin” 服务。
7. 下拉 “Actions” 菜单。

您会注意到无法暂停此服务，暂停项已被禁用。admin 服务是指向代理的管理员链接。如果您暂停该服务，将再也无法访问代理。

8. 选择 “Actions” > “Properties” 查看 admin 服务的属性。
9. 完成时请单击 “OK” 或 “Cancel”。

将物理目标添加到代理

默认情况下，为代理启用目标自动创建，这样就允许它动态地创建物理目标。因此，在开发环境中，您不必为了测试客户机代码而明确地创建目标。

不过，在生产设置中，建议您明确地创建物理目标。这使您（管理员）可以完全掌握代理上使用的目标。

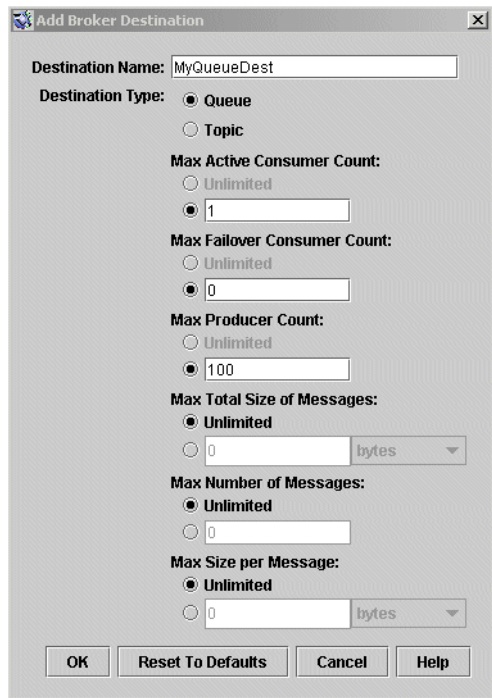
您可以控制代理是否可以通过设置 `imq.autocreate.topic` 或 `imq.autocreate.queue` 属性来添加自动创建的目标。有关详细信息，请参见第 70 页的“自动创建的（与管理员创建的相对）目标”。

在本教程的这节中，您将把一个物理目标添加到代理。应注意分配给目标的名称，您以后会在创建对应此物理目标的被管理对象时需要此名称。

► 将队列目标添加到代理

1. 在 "MyBroker" 的 “Destinations” 节点上单击右键并选择 “Add Broker Destination”。

将显示以下对话框：



The image shows a dialog box titled "Add Broker Destination". It contains several configuration fields:

- Destination Name:** A text field containing "MyQueueDest".
- Destination Type:** Two radio buttons: "Queue" (selected) and "Topic".
- Max Active Consumer Count:** Two radio buttons: "Unlimited" and "1" (selected).
- Max Failover Consumer Count:** Two radio buttons: "Unlimited" and "0" (selected).
- Max Producer Count:** Two radio buttons: "Unlimited" and "100" (selected).
- Max Total Size of Messages:** Two radio buttons: "Unlimited" (selected) and "0" (with a "bytes" dropdown).
- Max Number of Messages:** Two radio buttons: "Unlimited" (selected) and "0".
- Max Size per Message:** Two radio buttons: "Unlimited" (selected) and "0" (with a "bytes" dropdown).

At the bottom, there are four buttons: "OK", "Reset To Defaults", "Cancel", and "Help".

2. 在 “Destination Name” 字段中输入 MyQueueDest。
3. 选择 “Queue” 单选按钮（如果尚未选中此按钮）。
4. 单击 “OK” 添加物理目标。

现在，目标显示在结果窗格中。

使用物理目标

在代理上添加一个物理目标后，您可以按照稍后的步骤说明来执行以下任意一项任务：

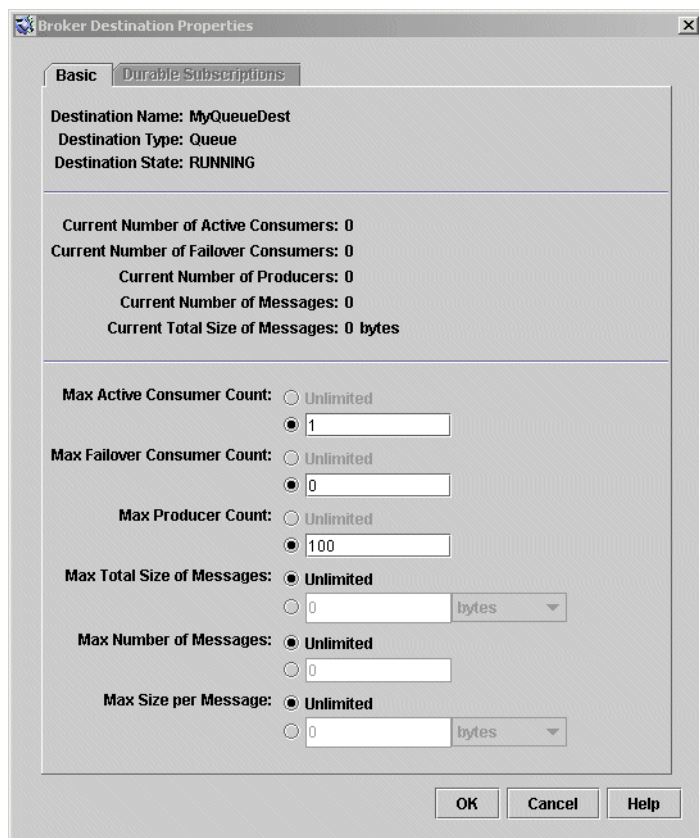
- 查看和更新物理目标的属性

- 清除目标上的消息
- 删除目标

► 查看物理目标的属性

1. 选择 "MyBroker" 的 "Destinations" 节点。
2. 在结果窗格中选择 "MyQueueDest"。
3. 选择 “Actions” > “Properties”。

将显示以下对话框：



注意，对话框显示有关队列以及您可更改的某些属性的当前状态信息。

4. 单击 “Cancel” 关闭对话框。

➤ 清除目标中的消息

1. 在结果窗格中选择物理目标。
2. 选择 “Actions” > “Purge Messages”。

将显示一个确认对话框。

清除消息将删除消息，留下一个空目标。

➤ 删除目标

1. 在结果窗格中选择物理目标。
2. 选择 “Edit” > “Delete”。

将显示一个确认对话框。

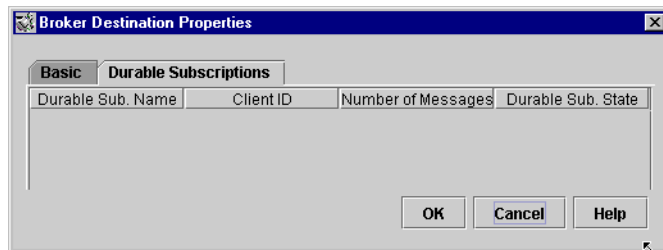
注意

请不要删除 MyQueueDest 队列目标。

删除目标将清除该目标上的消息并删除目标。

获取有关主题目标的信息

代理主题目标属性对话框包含一个列出长期订阅的相关信息的附加选项卡。对于队列目标，该选项卡处于禁用状态。



使用此对话框可以执行以下操作:

- 清除长期订阅，从而删除与长期订阅关联的所有消息
- 删除长期订阅，从而清除与长期订阅关联的所有消息并删除该长期订阅

使用对象存储

对象存储可以是 LDAP 目录服务器或文件系统存储（文件系统上的目录），用于存储 Message Queue 被管理对象。这些被管理对象将封装有关客户机应用程序使用的对象的 Message Queue 特有实现方案和配置信息。

尽管可以在客户机代码中实例化和配置被管理对象，但建议您（管理员）最好创建和配置这些对象，并将其存储在对象存储中，客户机应用程序将通过 JNDI 访问该对象存储。这将使客户机代码与提供者无关。

有关被管理对象的详细信息，请参见第 78 页的“[Message Queue 受管理对象](#)”。

您不能使用管理控制台来*创建*对象存储。必须按照以下章节中的说明提前创建。

添加对象存储

添加对象存储将在管理控制台中创建一个对现有对象存储的引用。即使退出并重新启动控制台，此引用也将保留。

► 添加文件系统对象存储

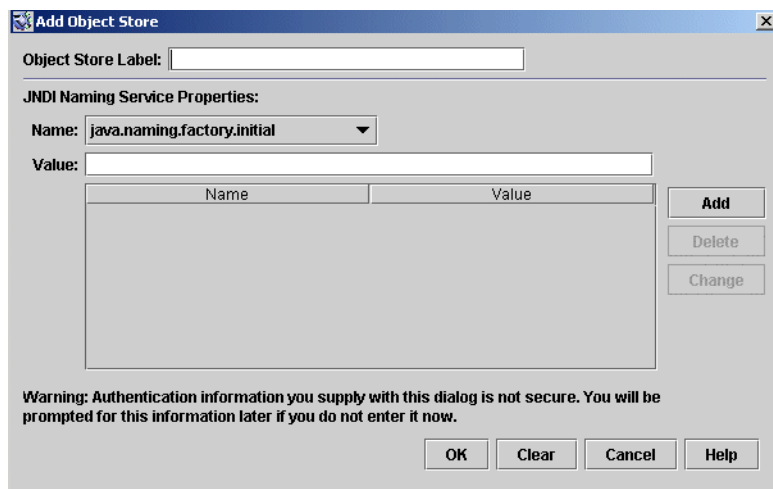
1. 如果在您的 C 驱动器上没有名为 Temp 的文件夹，那么现在就请创建这个文件夹。

本教程中使用的样例应用程序假定对象存储是 C 驱动器上一个名为 Temp 的文件夹。文件系统对象存储通常可以是任何驱动器上的任何目录。

非 Windows 用户：可以使用现有的 /tmp 目录（应该已经存在）。

2. 在“Object Stores”上单击右键并选择“Add Object Store”。

将显示以下对话框：



3. 在名为 "ObjectStoreLabel" 的字段中输入 MyObjectStore。

这仅是在管理控制台中提供一个对象存储显示的标签。

在以下步骤中，您将需要输入 JNDI 名称 / 值对。JMS 兼容应用程序使用这些名称 / 值对查找被管理对象。

4. 从 “Name” 下拉列表中，选择 java.naming.factory.initial。

此属性使您可以指定希望使用哪个 JNDI 服务提供者。例如，文件系统服务提供者或 LDAP 服务提供者。

5. 在 “Value” 字段中，输入以下内容

```
com.sun.jndi.fscontext.RefFSContextFactory
```

这表示您将使用文件系统存储。（对于 LDAP 存储，将指定 com.sun.jndi.ldap.LdapCtxFactory。）

在生产环境中，您可能希望将 LDAP 目录服务器用作对象存储。有关设置服务器和进行 JNDI 查找的信息，请参见第 166 页的“LDAP 服务器对象存储库”。

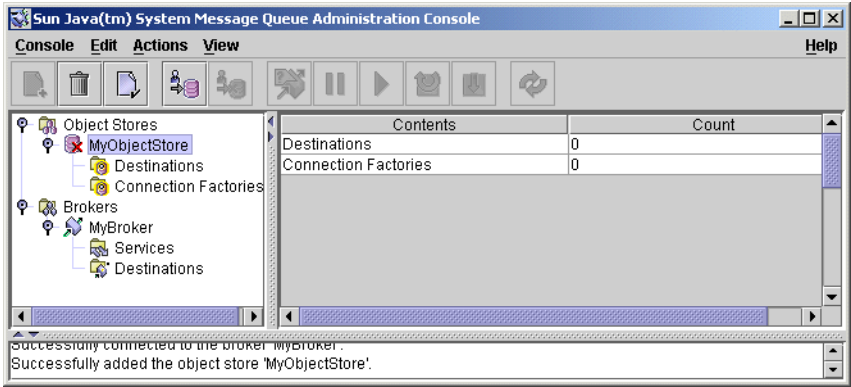
6. 单击 “Add” 按钮。

您会注意到现在属性概要窗格中列出了属性及其值。

7. 从 “Name” 下拉列表中，选择 java.naming.provider.url。

此属性使您可以指定对象存储的确切位置。对于文件系统类型的对象存储，这将是现有目录的名称。

- 8. 在 “Value” 字段中，输入以下内容
file:///C:/Temp
(在 Solaris 和 Linux 操作系统上为 file:///tmp)
- 9. 单击 “Add” 按钮。
您会注意到现在属性概要窗格中列出了属性及其值。如果正在使用 LDAP 服务器，还可能必须要指定验证信息。这对文件系统存储不是必需的。
- 10. 单击 “OK” 添加对象存储。
- 11. 如果未在浏览窗格中选中节点 "MyObjectStore"，那么现在请选中它。
现在，管理控制台的显示如下所示：



浏览窗格中列出了该对象存储，结果窗格中则列出了对象存储的内容 “Destinations” 和 “Connection Factories”。我们尚未将任何被管理对象添加到对象存储，如结果窗格的 “Count” 列中所示。

浏览窗格中的对象存储图标上绘制了一个红色的 X。这表示它处于断开连接的状态。需要先连接到对象存储，然后才能使用它。

检查对象存储属性

当管理控制台从对象存储断开连接时，可以查看和更改对象存储的某些属性。

► 显示对象存储属性

1. 在浏览窗格的 "MyObjectStore" 上单击右键。
2. 从弹出式菜单中选择 “Properties”。

将显示一个对话框，其中显示所有在添加对象存储时指定的属性。您可以更改这些属性中的任意属性，并单击 “OK” 更新旧信息。

3. 单击 “OK” 或 “Cancel” 关闭对话框。

连接到对象存储

必须先连接到对象存储，然后才能将对象添加其中。

► 连接到对象存储

1. 在浏览窗格的 "MyObjectStore" 上单击右键。
2. 从弹出式菜单中选择 “Connect to Object Store”。

您会注意到对象存储图标上的红色 X 不见了。现在可以将对象、连接工厂和目标添加到对象存储。

添加连接工厂被管理对象

可以使用管理控制台来创建和配置连接工厂。客户机代码使用连接工厂连接到代理。通过配置连接工厂，可以控制使用连接工厂创建的连接的行为。

有关配置连接工厂的信息，请参见联机帮助和 《*Message Queue Java Client Developer's Guide*》。

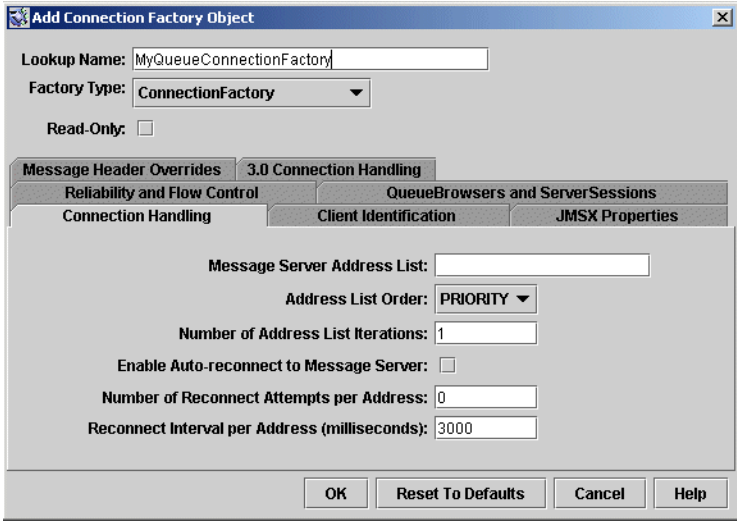
注意

管理控制台仅列出和显示了 Message Queue 被管理对象。如果对象存储中包含与希望添加的被管理对象具有相同查找名称的非 Message Queue 对象，则您将在尝试执行添加操作时收到一条错误消息。

► 将连接工厂添加到对象存储

- 1. 如果仍未连接，请连接到 MyObjectStore （请参见第 107 页的 “连接到对象存储”）
- 2. 在 “Connection Factories” 节点上单击右键，并选择 “Add Connection Factory Object”。

将显示 “Add Connection Factory Object” 对话框。



- 3. 在 “LookupName” 字段中输入名称 "MyQueueConnectionFactory"。
这是客户机代码查找连接工厂时使用的名称，如 HelloWorldMessageJNDI.java 中的以下代码行所示：

```
qcf= (javax.jms.QueueConnectionFactory)
      ctx.lookup("MyQueueConnectionFactory")
```

- 4. 从下拉菜单中选择 "QueueConnectionFactory" 以指定连接工厂的类型。
- 5. 单击 “Connection Handling” 选项卡。
- 6. 您通常会在 “Message Server Address List” 字段中输入客户机将连接的代理的地址。此字段的一个示例如下所示：

```
mq://localhost:7676/jms
```

您不需要输入值，因为默认情况下已经将连接工厂配置为连接在端口 7676 的本地主机上运行的代理，这是教程示例所期望的配置。

7. 逐个单击此对话框的各个选项卡，查看您可以配置的连接工厂的信息类型。使用“Add Connection Factory Object”对话框右下角的“Help”按钮来获得有关各个选项卡的信息。目前请勿更改任何默认值。
8. 单击“OK”创建队列连接工厂。
9. 观察结果窗格。列出了新创建的连接工厂的查找名称和类型。

添加目标被管理对象

目标被管理对象与代理上的物理目标相关联。或者说，它们指向这些目标，使客户机可以查找并找到物理目标，而与命名和配置这些目标的提供者特有方法无关。

当客户机发送消息时，它将查找（或实例化）目标被管理对象，并在 JMS API 的 `send()` 方法中引用它。然后代理负责将消息传送给与该被管理对象相关联的物理目标。

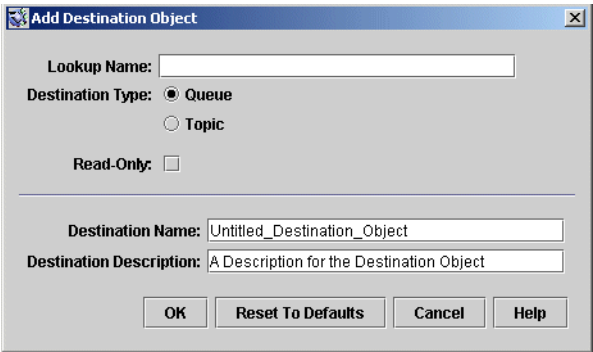
- 如果已创建与该被管理对象相关联的物理目标，则代理会将消息传送给该物理目标。
- 如果尚未创建物理目标，并且物理目标的自动创建功能已启用，代理将自动创建物理目标并将消息传送给该目标。
- 如果尚未创建物理目标，并且物理目标的自动创建功能已被禁用，代理将无法创建物理目标，也就无法传送消息。

在本教程的下一部分中，您将添加一个与前面添加的物理目标相对应的被管理对象。

► 将目标添加到对象存储

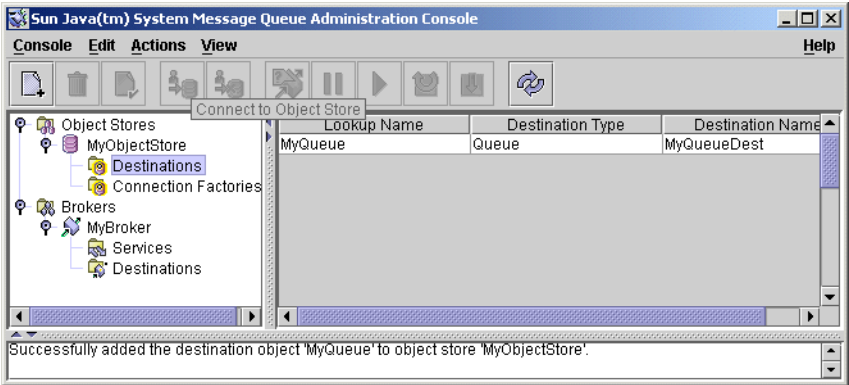
1. 在浏览窗格中的“Destinations”节点（在“MyObjectStore”节点下）上单击右键。
2. 选择“Add Destination Object”。

管理控制台将显示一个“Add Destination Object”对话框，用于指定有关对象的信息。



- 3. 在 “Lookup Name” 字段中输入 "MyQueue"。
JNDI 查找调用使用该查找名称来查找对象。在样例应用程序中，此调用如下所示：

```
queue=(javax.jms.Queue)ctx.lookup("MyQueue");
```
- 4. 对于 “Destination Type”，请选择 “Queue” 单选按钮。
- 5. 在 “Destination Name” 字段中输入 MyQueueDest。
这是您在代理上添加物理目标时指定的名称（请参见第 100 页的 “将物理目标添加到代理”）。
- 6. 单击 “OK”。
- 7. 在浏览窗格中选择 “Destinations”，并注意在结果窗格中如何显示有关刚才添加的队列目标被管理对象的信息。



被管理对象的属性

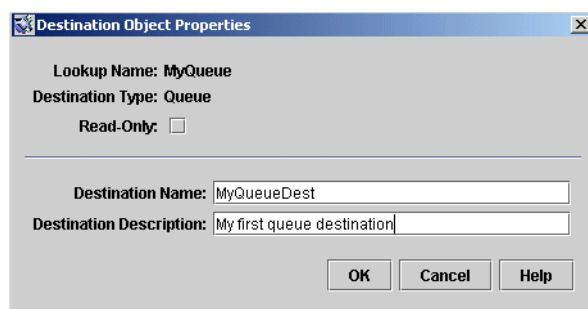
要查看或更新被管理对象的属性，需要在浏览窗格中选择“Destinations”或“Connection Factories”，然后在结果窗格中选择特定对象，并选择“Actions”>“Properties”。

► 查看或更新目标对象的属性

1. 在浏览窗格中选择“MyObjectStore”的“Destinations”节点。
2. 在结果窗格中选择“MyQueue”。
3. 选择“Actions”>“Properties”以查看“Destination Object Properties”对话框。

请注意，您仅能更改目标名称和说明的值。要更改查找名称，就必须先删除对象，然后添加一个命名为所需查找名称的新队列被管理对象。

4. 单击“Cancel”关闭对话框。



更新控制台信息

不管使用对象存储还是使用代理，都可以通过选择“View”>“Refresh”来更新任何元素或元素组的显示。

运行样例应用程序

提供样例应用程序 HelloWorldMessageJNDI 是为了在本教程中使用（其位置的相关信息，请参见下面的步骤 1）。它使用在前面的教程中创建的物理目标和被管理对象：名为 MyQueueDest 的队列物理目标、JNDI 查找名称分别为 MyQueueConnectionFactory 和 MyQueue 的队列连接工厂被管理对象和队列被管理对象。

代码创建简单的队列发件人和收件人，并发送和接收一条 "Hello World" 消息。

► 运行 HelloWorldMessageJNDI 应用程序

1. 使包含 HelloWorldMessageJNDI 应用程序的目录成为当前目录，例如：

```
cd IMQ_HOME\demo\helloworld\helloworldmessagejndi （在 Windows 操作系统上）
```

```
cd /usr/demo/imq/helloworld/helloworldmessagejndi （在 Solaris 操作系统上）
```

```
cd /opt/imq/demo/helloworld/helloworldmessagejndi （在 Linux 操作系统上）
```

您应查看 HelloWorldMessageJNDI.class 文件是否存在。（如果要更改该应用程序，需要使用《Message Queue C Client Developer's Guide》的快速入门教程中用于编译客户机应用程序的指令对其重新编译。

2. 设置 CLASSPATH 变量以便包括含有 HelloWorldMessageJNDI.class 文件及 Message Queue 产品附带的以下 jar 文件的当前目录：jms.jar、imq.jar 和 fscontext.jar。有关设置 CLASSPATH 的说明，请参见《Message Queue Java Client Developer's Guide》。

JNDI jar 文件 (jndi.jar) 是随 JDK 1.4 一起提供的。如果您使用的是该 JDK，则不必将 jndi.jar 添加到 CLASSPATH 设置。如果您使用的是 JDK 的早期版本，则必须将 jndi.jar 包含在 CLASSPATH 中。有关其他信息，请参见《Message Queue Java Client Developer's Guide》。

3. 运行应用程序之前，请打开源文件 HelloWorldMessageJNDI.java 并仔细阅读该源文件。虽然这个源文件很简短，但是其注释非常详细，您可以很清楚地了解该应用程序是如何使用您利用教程所创建的被管理对象和目标。
4. 通过执行以下命令之一来运行 HelloWorldMessageJNDI 应用程序：

```
java HelloWorldMessageJNDI （在 Windows 操作系统上）
```

```
% java HelloWorldMessageJNDI file:///tmp （在 Solaris 和 Linux 操作系统上）
```


如果应用程序成功运行，则会看到以下输出：

```
java HelloWorldMessageJNDI
Using file:///C:/Temp for Context.PROVIDER_URL

Looking up Queue Connection Factory object with lookup name:
MyQueueConnectionFactory
Queue Connection Factory object found.
Looking up Queue object with lookup name: MyQueue
Queue object found.

Creating connection to broker.
Connection to broker created.

Publishing a message to Queue: MyQueueDest
Received the following message: Hello World
```


启动与配置代理

安装 Sun Java™ System Message Queue 以后，可以使用 `imqbrokerd` 命令来启动代理。代理实例的配置由一组配置文件和使用 `imqbrokerd` 命令传送的选项共同控制，这些选项会覆盖配置文件中相应的属性。

本章说明了 `imqbrokerd` 命令的语法，以及如何使用命令行选项和配置文件来配置代理实例。此外，本章还介绍了如何执行以下操作：

- 编辑代理实例配置文件
- 使用代理群集
- 控制代理的日志记录

关于如何启动代理并将其用作 Windows 服务的说明，请参见第 297 页的“使用代理作为 Windows 服务”。

配置文件

已安装代理的配置文件模板用于配置代理，这些模板所在目录随操作系统的不同而有所不同，如附录 A “Message Queue 数据的位置”。所示。

该目录存储以下文件：

- **启动时装入的默认配置文件。**文件名是 `default.properties`。该文件不可编辑。您可能需要读取该文件来确定默认设置以及查找要更改的属性的确切名称。
- **安装配置文件，**其中包含安装 Message Queue 时指定的所有属性。文件名是 `install.properties`。安装完成后该文件不可编辑。

实例配置文件

首次运行一个代理时，系统将创建一个实例配置文件，用于指定该代理的实例的配置属性。实例配置文件存储在一个目录中，该目录用与此配置文件相关联的代理实例的名称 (*instanceName*) 标识（请参见附录 A “[Message Queue 数据的位置](#)”）：

```
.../instances/instanceName/props/config.properties
```

注意

.../instances/*instanceName* 目录（和实例配置文件）由相应代理实例的创建人拥有。代理实例的所有后续启动操作必须由同一用户执行。

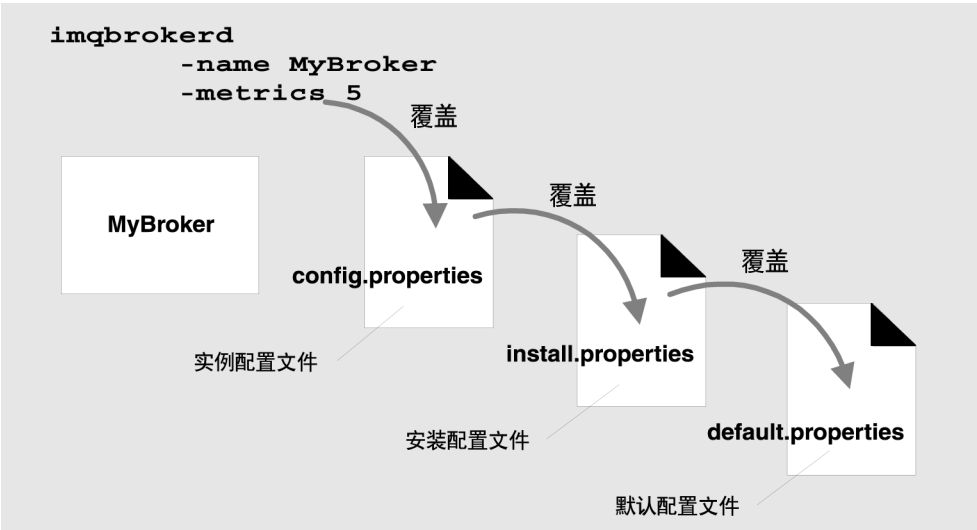
实例配置文件由代理实例维护。使用管理工具更改配置时，此文件也将被修改。您还可以手动编辑实例配置文件来更改配置（请参见第 118 页的“[编辑实例配置文件](#)”）。要完成此操作，您必须是 .../instances/*instanceName* 目录的拥有者，或者以超级用户身份登录以便更改目录权限。

如果连接群集中的代理实例（请参见第 73 页的“[多代理群集（企业版）](#)”），那么可能还需要使用[群集配置文件](#)来指定群集配置信息。有关详细信息，请参见第 128 页的“[群集配置属性](#)”。

合并属性值

启动时，系统会合并不同配置文件中的属性值。它将使用在安装配置文件和实例配置文件中设置的值来覆盖默认配置文件中指定的值。您可以用 `imqbrokerd` 命令选项来覆盖得到的值。图 5-1 所示为以上说明的图解。

图 5-1 代理配置文件



属性命名语法

配置文件中的所有 Message Queue 属性定义均使用以下命名语法：

```
propertyName=value [[, value1] ...]
```

例如，下面的条目指定：在拒绝其他消息前，代理可以在内存和持久性存储器中最多保留 50,000 条消息：

```
imq.system.max_count=50000
```

下面的条目指定：将每天（86400 秒）创建一个新日志文件：

```
imq.log.file.rolloversecs=86400
```

第 118 页的表 5-1 按字母顺序列出了代理配置属性及其默认值。

编辑实例配置文件

首次运行代理实例时，系统将自动创建一个 config.properties 文件。您可以编辑该实例配置文件，以自定义相应代理实例的行为和资源使用。

代理实例仅在启动时读取 config.properties 文件。要永久更改 config.properties 文件，您可以：

- 使用管理工具。关于可以使用 imqcmd 设置的属性的信息，请参见第 144 页的表 6-4。
- 在代理实例关闭时编辑 config.properties 文件，然后重新启动实例。（在 Solaris 和 Linux 平台中，只有第一个启动代理实例的用户具有编辑 config.properties 文件的权限。）

表 5-1 按字母顺序列出了代理配置属性及其默认值。关于各个属性的含义和用法的更多信息，请参考指定的交叉参考的小节。

表 5-1 代理实例配置属性

属性名称	类型	默认值	参考
imq.accesscontrol.enabled	布尔值	true	第 62 页的表 2-6
imq.accesscontrol.file.filename	字符串	accesscontrol.properties	第 62 页的表 2-6
imq.authentication.basic.user_repository	字符串	file	第 62 页的表 2-6
imq.authentication.client.response.timeout	整数 (秒)	180	第 62 页的表 2-6
imq.authentication.type	字符串	digest	第 62 页的表 2-6
imq.autocreate.destination.isLocalOnly	布尔值	false	第 70 页的表 2-10
imq.autocreate.destination.limitBehavior	字符串	REJECT_NEWEST	第 70 页的表 2-10
imq.autocreate.destination.maxBytesPerMsg	字节字符串 ¹	10k	第 70 页的表 2-10
imq.autocreate.destination.maxNumMsgs	整数	100,000	第 70 页的表 2-10
imq.autocreate.destination.maxNumProducers	整数	100	第 70 页的表 2-10

¹ 键入为 *字节字符串* 的值，可以使用字节、千字节和兆字节的形式来表示；例如：1000 表示 1000 字节；7500b 表示 7500 字节；77k 表示 77 千字节（77 x 1024 = 78848 字节）；17m 表示 17 兆字节（17 x 1024 x 1024 = 17825792 字节）

表 5-1 代理实例配置属性 (续)

属性名称	类型	默认值	参考
imq.autocreate.destination. maxTotalMsgBytes	字节字符串 ¹	10m	第 70 页的表 2-10
imq.autocreate.queue	布尔值	true	第 70 页的表 2-10
imq.autocreate.queue. consumerFlowLimit	整数	1000	第 70 页的表 2-10
imq.autocreate.queue. localDeliveryPreferred	布尔值	false	第 70 页的表 2-10
imq.autocreate.queue. maxNumActiveConsumers	整数	1	第 70 页的表 2-10
imq.autocreate.queue. maxNumBackupConsumers	整数	0	第 70 页的表 2-10
imq.autocreate.topic	布尔值	true	第 70 页的表 2-10
imq.autocreate.topic. consumerFlowLimit	整数	1,000	第 70 页的表 2-10
imq.cluster. <i>property_name</i>			第 128 页的表 5-3
imq.hostname	字符串	所有可用的 IP 地址	第 52 页的表 2-3
imq.httpjms.http. <i>property_name</i>			第 276 页的表 C-1
imq.httpsjms.https. <i>property_name</i>			第 287 页的表 C-3
imq.keystore. <i>property_name</i>			第 199 页的表 8-8
imq.log.console.output	字符串	ERROR WARNING	第 66 页的表 2-9
imq.log.console.stream	字符串	ERR	第 66 页的表 2-9
imq.log.file.dirpath	字符串	请参见附录 A “Message Queue 数据的位置”	第 66 页的表 2-9
imq.log.file.filename	字符串	log.txt	第 66 页的表 2-9
imq.log.file.output	字符串	ALL	第 66 页的表 2-9
imq.log.file.rolloverbytes	整数 (字节)	-1 (不转移)	第 66 页的表 2-9
imq.log.file.rolloversecs	整数 (秒)	604800	第 66 页的表 2-9
imq.log.level	字符串	INFO	第 66 页的表 2-9

¹ 键入为 *字节字符串* 的值，可以使用字节、千字节和兆字节的形式来表示：例如：1000 表示 1000 字节；7500b 表示 7500 字节；77k 表示 77 千字节（77 x 1024 = 78848 字节）；17m 表示 17 兆字节（17 x 1024 x 1024 = 17825792 字节）

表 5-1 代理实例配置属性 (续)

属性名称	类型	默认值	参考
imq.log.syslog.facility	字符串	LOG_DAEMON	第 66 页的表 2-9
imq.log.syslog.identity	字符串	imqbrokerd_\${imq.instanceName}	第 66 页的表 2-9
imq.log.syslog.logconsole	布尔值	false	第 66 页的表 2-9
imq.log.syslog.logpid	布尔值	true	第 66 页的表 2-9
imq.log.syslog.output	字符串	ERROR	第 66 页的表 2-9
imq.log.timezone	字符串	当地时区	第 66 页的表 2-9
imq.message.expiration.interval	整数 (秒)	60	第 56 页的表 2-4
imq.message.max_size	字节字符串 ¹	70m	第 56 页的表 2-4
imq.metrics.enabled	布尔值	true	第 66 页的表 2-9
imq.metrics.interval	整数 (秒)	-1 (从不)	第 66 页的表 2-9
imq.metrics.topic.enabled	布尔值	true	第 66 页的表 2-9
imq.metrics.topic.interval	整数 (秒)	60	第 66 页的表 2-9
imq.metrics.topic.persist	布尔值	false	第 66 页的表 2-9
imq.metrics.topic.timetolive	整数 (秒)	300	第 66 页的表 2-9
imq.passfile.dirpath	字符串	请参见附录 A “Message Queue 数据的位置”	第 62 页的表 2-6
imq.passfile.enabled	布尔值	false	第 62 页的表 2-6
imq.passfile.name	字符串	密码文件	第 62 页的表 2-6
imq.persist.file.destination.message.filepool.limit	整数	100	第 59 页的表 2-5
imq.persist.file.message.cleanup	布尔值	false	第 59 页的表 2-5
imq.persist.file.message.filepool.cleanratio	整数	0	第 59 页的表 2-5

¹ 键入为 *字节字符串* 的值，可以使用字节、千字节和兆字节的形式来表示；例如：1000 表示 1000 字节； 7500b 表示 7500 字节； 77k 表示 77 千字节（77 x 1024 = 78848 字节）； 17m 表示 17 兆字节（17 x 1024 x 1024 = 17825792 字节）

表 5-1 代理实例配置属性 (续)

属性名称	类型	默认值	参考
imq.persist.file.message.max_record_size	字节字符串 ¹	1m	第 59 页的表 2-5
imq.persist.file.sync.enabled	布尔值	false	第 59 页的表 2-5
imq.persist.jdbc.property_name			第 267 页的表 B-1
imq.persist.store	字符串	file	第 59 页的表 2-5
imq.ping.interval	整数	120	第 52 页的表 2-3
imq.portmapper.backlog	整数	50	第 52 页的表 2-3
imq.portmapper.hostname	字符串	从 imq.hostname 继承	第 52 页的表 2-3
imq.portmapper.port	整数	7676	第 52 页的表 2-3
imq.resource_state.count	整数 (百分比)	5000 (green) 500 (yellow) 50 (orange) 0 (red)	第 56 页的表 2-4
imq.resource_state.threshold	整数 (百分比)	0 (green) 80 (yellow) 90 (orange) 98 (red)	第 56 页的表 2-4
imq.service.activelist	列表	jms、admin	第 52 页的表 2-3
imq.service_name.accesscontrol.enabled	布尔值	从系统范围属性继承值	第 62 页的表 2-6
imq.service_name.accesscontrol.file.filename	字符串	从系统范围属性继承值	第 62 页的表 2-6
imq.service_name.authentication.type	字符串	从系统范围属性继承值	第 62 页的表 2-6
imq.service_name.max_threads	整数	1000 (jms) 500 (ssljms) 500 (httpjms) 500 (httpsjms) 10 (admin) 10 (ssladmin)	第 52 页的表 2-3

¹ 键入为字节字符串的值，可以使用字节、千字节和兆字节的形式来表示：例如：1000 表示 1000 字节；7500b 表示 7500 字节；77k 表示 77 千字节（77 x 1024 = 78848 字节）；17m 表示 17 兆字节（17 x 1024 x 1024 = 17825792 字节）

表 5-1 代理实例配置属性 (续)

属性名称	类型	默认值	参考
imq.service_name.min_threads	整数	10 (jms) 10 (ssljms) 10 (httpjms) 10 (httpsjms) 4 (admin) 4 (ssladmin)	第 52 页的表 2-3
imq.service_name.protocol_type. hostname	字符串	从 imq.hostname 继承	第 52 页的表 2-3
imq.service_name.protocol_type. port	整数	0 (动态分配)	第 52 页的表 2-3
imq.service_name. threadpool_model	字符串	dedicated (jms) dedicated (ssljms) dedicated (httpjms) dedicated (httpsjms) dedicated (admin) dedicated (ssladmin)	第 52 页的表 2-3
imq.shared. connectionMonitor_limit	整数	512 (Solaris 和 Linux) 64 (Windows)	第 52 页的表 2-3
imq.system.max_count	整数、 0 (没有限制)	-1	第 56 页的表 2-4
imq.system.max_size	字节字符串 1、 0 (没有限制)	-1	第 56 页的表 2-4
imq.transaction.autorollback	布尔值	false	第 56 页的表 2-4
imq.user_repository.ldap. property_name			第 190 页的表 8-5

1 键入为 *字节字符串* 的值，可以使用字节、千字节和兆字节的形式来表示：例如：1000 表示 1000 字节； 7500b 表示 7500 字节； 77k 表示 77 千字节（77 x 1024 = 78848 字节）； 17m 表示 17 兆字节（17 x 1024 x 1024 = 17825792 字节）

启动代理

要启动代理实例，请使用 `imqbrokerd` 命令。

注意 使用管理控制台 (`imqadmin`) 或命令行实用程序 (`imqcmd`) 无法启动代理实例。这两个 Message Queue 管理工具只有代理实例已在运行时才能使用。

要覆盖一个或多个属性值，请指定有效的 `imqbrokerd` 命令行选项。命令行选项覆盖代理配置文件中的值，但仅对当前代理会话有效：命令行选项不会被写入到实例配置文件中。

imqbrokerd 命令的语法

`imqbrokerd` 命令的语法如下所示（选项与变量之间以一个空格分隔）：

```
imqbrokerd [[ -Dproperty=value] ...]
[ -backup fileName]
[ -cluster "[broker1] [[,broker2]...]"
[ -dbuser userName] [ -dbpassword password]
[ -force]
[ -h|-help]
[ -javahome path]
[ -ldappassword password]
[ -license licenseName]
[ -loglevel level]
[ -metrics interval]
[ -name instanceName]
[ -password keypassword] [ -passfile fileName]
[ -port number]
[ -remove instance]
[ -reset data]
[ -restore fileName]
[ -shared]
[ -silent|-s] [ -tty]
[ -upgrade-store-nobackup]
[ -version]
[ -vmargs arg1 [[arg2]...]
```

注意 在 Solaris 中，可以将代理配置为异常退出后自动重新启动，方法是把 `/etc/imq/imqbrokerd.conf` 配置文件中的 `RESTART` 属性设置为 `YES`。

注意 在 Solaris 和 Linux 平台中，对包含配置信息和持久性数据的目录的权限取决于第一次启动代理实例的用户的 `umask`。因此，为了让代理实例正常运行，就只能由原用户启动该实例。

启动示例

下面的示例说明如何使用 `imqbrokerd` 命令。有关 `imqbrokerd` 命令行选项的更多详细信息，请参见第 125 页的表 5-2。

► 启动使用默认代理名称和配置的代理实例

使用以下命令：

```
imqbrokerd
```

该命令启动一个位于本地计算机上的代理（名为 `imqbroker`）的默认实例，且其端口映射器的端口号是 7676。

► 启动具有企业版试用许可证的代理实例

如果您具有平台版许可证，但希望试用企业版功能 90 天，那么您可以使用 `-license` 命令行选项并将 `"try"` 作为要使用的许可证进行传送，即可启用企业版试用许可证：

```
imqbrokerd -license try
```

每次启动代理实例时都必须使用此选项，否则默认情况下将回退到基本的平台版许可证。

► 启动具有插入持久性的已命名代理实例

要启动使用插入数据存储（请参见第 265 页上的附录 B “设置插入的持久性”）的名为 `myBroker` 的代理，并且此代理要求用户名和密码，请使用以下命令：

```
imqbrokerd -name myBroker -dbuser myName -dbpassword myPassword
```

imqbrokerd 选项概述

表 5-2 说明了 imqbrokerd 命令的选项以及受到各个选项影响的配置属性（如果存在）。

表 5-2 imqbrokerd 选项

选项	所影响的属性	说明
-backup <i>fileName</i>	不影响任何属性。	仅适用于代理群集。将主管代理的配置更改记录备份到指定的文件。请参见第 132 页的“备份配置更改记录”。
-cluster "[<i>broker1</i>] [<i>,broker2</i>]..."	将 imq.cluster.brokerlist 设置为要连接的代理的列表。	仅适用于代理群集。连接至指定的主机和端口上的所有代理。该列表将与 imq.cluster.brokerlist 属性中的列表合并。如果没有指定 <i>host</i> 的值，将使用 localhost。如果没有指定 <i>port</i> 的值，将使用值 7676。关于如何使用该选项连接多个代理的详细信息，请参见第 128 页的“使用群集（企业版）”。
其中 <i>broker</i> 可以是		
• <i>host[:port]</i>		
• [<i>host</i>]: <i>port</i>		
-dbpassword <i>password</i>	将 imq.persist.jdbc.password 设置为指定密码	指定插入的 JDBC 兼容数据存储的密码。请参见附录 B “设置插入的持久性”。
-dbuser <i>userName</i>	将 imq.persist.jdbc.user 设置为指定的用户名	指定插入的 JDBC 兼容数据存储的用户名。请参见附录 B “设置插入的持久性”。
-Dproperty= <i>value</i>	设置系统属性。覆盖实例配置文件中的相关属性值。	将指定属性设置为指定值。关于代理配置属性，请参见第 118 页的表 5-1。 注意： 请仔细检查使用 D 选项设置的属性的拼写和格式。如果传送了不正确的值，系统不会向您发出警告，Message Queue 将无法对这些属性进行设置。
-force	不影响任何属性。	执行操作，而无需用户确认。此选项仅用于 -remove instance 和 -upgrade-store-nobackup 选项，通常要求确认。
-h -help	不影响任何属性。	显示帮助。不执行命令行上的其他选项。
-javahome <i>path</i>	不影响任何属性。	指定替代 Java 2 兼容 JDK 的路径。默认使用捆绑的运行时。
-ldappassword <i>password</i>	将 imq.user_repository.ldap.password 设置为指定的密码	指定访问 LDAP 用户信息库的密码。请参见第 190 页的“使用 LDAP 服务器管理用户信息库”。

表 5-2 imqbrokerd 选项 (续)

选项	所影响的属性	说明
-license <i>[licenseName]</i>	不影响任何属性。	指定装入一个与 Message Queue 产品版本的默认许可证不同的许可证。如果不指定许可证名称，该选项将列出系统中安装的所有许可证。根据所安装的 Message Queue 版本的不同， <i>licenseName</i> 的取值也不同：pe（平台版，只具有基本功能）、try（平台版的 90 天试用企业功能）和 unl（企业版）。请参见第 33 页的“产品版本”。
-loglevel <i>level</i>	将 imq.broker.log.level 设置为指定级别。	将日志记录级别指定为下列四项之一：NONE、ERROR、WARNING 或 INFO。默认值是 INFO。有关详细信息，请参见第 64 页的“日志记录器”。
-metrics <i>interval</i>	将 imq.metrics.interval 设置为指定秒数。	指定代理度量依据写入日志记录器的时间间隔（以秒为单位）。
-name <i>instanceName</i>	将 imq.instancename 设置为指定名称。	指定该代理的实例名称，并使用相应的实例配置文件。如果不指定代理名称，实例名将设置为 imqbroker。 注： 如果在同一台主机上运行了多个代理实例，每个实例必须具有唯一的名称。
-passfile <i>fileName</i>	将 imq.passfile.enabled 设置为 true。将 jmq.passfile.dirpath 设置为包含文件的路径。 将 imq.passfile.name 设置为该文件名。	指定文件名，从该文件中可以读取 SSL 密钥存储、LDAP 用户信息库或 JDBC 兼容数据库的密码。有关详细信息，请参见第 204 页的“使用密码文件”。
-password <i>keypassword</i>	将 imq.keystore.password 设置为指定密码。	指定 SSL 证书密钥存储的密码。有关详细信息，请参见第 60 页的“安全性管理器”。
-port <i>number</i>	将 imq.portmapper.port 设置为指定端口号。	指定代理的端口映射器的端口号。默认情况下，该端口号设置为 7676。要在同一台服务器上运行两个代理实例，每个代理的端口映射器必须拥有不同的端口号。 Message Queue 客户机使用此端口号连接到代理实例。
-remove instance	不影响任何属性。	导致代理实例被删除：删除与实例相关联的实例配置文件、日志文件、持久性存储库与其他文件和目录。除非指定 -force 选项，否则需要用户确认。

表 5-2 imqbrokerd 选项 (续)

选项	所影响的属性	说明
-reset store messages durables props	不影响任何属性。	<p>根据给定变量，重置代理实例的数据存储（或数据存储的子集）或配置属性。</p> <p>重置数据存储将清除所有的持久性数据，包括持久性消息、长期订阅和事务信息。这使您可以启动一个完全清空的代理实例。也可以仅清除所有的持久性消息或仅清除所有的长期订阅。（如果不希望下次重新启动代理实例时重置持久性存储库，请勿使用 -reset 选项重新启动代理实例。）有关详细信息，请参见第 57 页的“持久性管理器”。</p> <p>重置代理的属性，用一个空文件代替现有的实例配置文件 (config.properties)：所有的属性将使用默认值。</p>
-restore fileName	不影响任何属性。	<p>仅适用于代理群集。用指定的备份文件替换主管代理的配置更改记录。该文件必须是原来用 -backup 选项创建的那个备份文件。请参见第 133 页的“恢复配置更改记录”。</p>
-shared	将 imq.jms.threadpool_model 设置为 shared。	<p>指定使用共享线程池模型来实现 jms 连接服务。在该模型中，连接之间共享线程，以增加代理实例支持的连接数量。有关详细信息，请参见第 50 页的“连接服务”。</p>
-silent -s	将 imq.log.console.output 设置为 NONE。	<p>停止向控制台记录日志信息。</p>
-tty	将 imq.log.console.output 设置为 ALL	<p>指定在控制台上显示所有消息。默认情况下，只显示 WARNING 和 ERROR 级别的消息。</p>
-upgrade-store-nobackup	不影响任何属性。	<p>指定从不兼容版本升级到 Message Queue 3.5 或 Message Queue 3.5 SPx 将自动删除原有的数据存储。有关其他详细信息，请参见《Message Queue 安装指南》。</p>
-version	不影响任何属性。	<p>显示所安装产品的版本号。</p>
-vmargs arg1 [[arg2]...]	不影响任何属性。	<p>指定传送到 Java VM 的变量。变量之间用空格分隔。如要传送多个变量，或是变量内包含空格，请使用闭合的引号。例如： imqbrokerd -tty -vmargs "-Xmx128m -Xincgc"</p>

使用群集（企业版）

本节介绍用于配置多代理群集的属性，并介绍了连接代理的两种方法，并说明如何管理群集。关于群集的介绍，请参见第 73 页的“多代理群集（企业版）”。

使用群集时，请确保同一个群集中所有代理的主机时钟处于同步状态（请参见第 301 页的“系统时钟设置”）。

群集配置属性

将代理连接到群集时，所有连接的代理必须指定为群集配置属性集。这些属性说明了群集中代理的合作。表 5-3 概述了与群集相关的配置属性。对于群集中的所有代理，标有星号 (*) 的属性必须具有相同值。

表 5-3 群集配置属性

属性名称	说明
imq.cluster.brokerlist*	指定群集中的所有代理。由 <i>host:port</i> 条目的列表（用逗号分隔各条目）组成，其中 <i>host</i> 是每个代理的主机名， <i>port</i> 是其端口映射器的端口号。例如： host1:3000, host2:8000, ctrhost
imq.cluster.masterbroker*	指定群集中哪个代理为记录状态更改的主管代理（如果存在）。该属性由 <i>host:port</i> 组成，其中 <i>host</i> 是主管代理的主机名， <i>port</i> 是其端口映射器的端口号。对生产环境设置该属性。例如，ctrhost:7676
imq.cluster.url*	指定代理配置文件的位置。此属性的适用情况：多个代理引用一个中心配置文件，而不是分别配置每个代理。由 URL 字符串组成：如果是在 Web 服务器上，可以使用标准的 http:URL 来访问。如果是在共享的驱动器上，可以使用 file:URL 来访问。 例如：http://webserver/imq/cluster.properties file:/net/mfsserver/imq/cluster.properties
imq.cluster.port	用于为群集中的 <i>每个代理</i> 指定 cluster 连接服务的端口号。 cluster 连接服务用于群集中代理之间的内部通信。 默认值：0（动态分配端口）
imq.cluster.hostname	对于群集中的 <i>每个代理</i> ，如果有多个主机可用（例如，一台计算机中安装了多个网络接口卡）， cluster 连接服务将绑定到该属性指定的主机（主机名或 IP 地址）。 cluster 连接服务用于群集中代理之间的内部通信。 默认值：继承 imq.hostname 的值（请参见第 52 页的表 2-3）

表 5-3 群集配置属性（续）

属性名称	说明
imq.cluster.transport*	指定 cluster 连接服务要在群集中的代理之间进行内部通信所使用的网络传输。为了在代理之间传送安全、加密的消息，请将群集中所有代理的此属性设置为 ssl。默认值：tcp

可以使用以下两种方法之一来设置群集属性：

- 可以在每个代理的实例配置文件中（或启动各个代理的命令行中）设置与群集相关的配置属性。例如，要连接代理 A（位于 host1、7676 端口）、代理 B（位于 host2、5000 端口）和代理 C（位于 ctrlhost、7676 端口），则代理 A、B 和 C 的实例配置文件需要设置如下属性。

```
imq.cluster.brokerlist=host1, host2:5000, ctrlhost
```

如果决定更改群集配置，此方法要求更新所有代理中与群集相关的属性。

- 在一个中心群集配置文件中设置群集配置属性。这些属性可能包括要连接代理的列表 (imq.cluster.brokerlist)、群集连接服务使用的网络传输 (imq.cluster.transport) 以及主管代理 (imq.cluster.masterbroker) 的地址（可选）。

如果使用该方法，还必须将群集中的每个代理的 imq.cluster.url 属性设置为指向群集配置文件的位置。从易于维护的角度出发，建议使用这种方法来配置群集。

以下代码样例显示了群集配置文件的内容。host1 和 ctrlhost 都是在默认端口上运行。这些属性指定 host1、host2 和 ctrlhost 连接到一个群集中，并指定 ctrlhost 作为主管代理。

```
imq.cluster.brokerlist=host1,host2:5000,ctrlhost
imq.cluster.masterbroker=ctrlhost
```

连接到此群集中的每个代理的实例配置文件都必须包含群集配置文件的 URL，例如：

```
imq.cluster.url=file:/home/cluster.properties
```

连接代理

本节介绍如何将代理连接至群集以及如何配置群集，以在群集中的代理之间传送安全、加密的消息。

连接方法

将代理连接到群集一般有两种方法：使用群集配置文件连接，或不使用群集配置文件连接。

无论使用哪种方法，启动的每个代理将每隔五秒钟就尝试一次连接到其他代理。主管代理启动后，该尝试就会成功。如果群集中的某个代理在主管代理之前启动，该代理将保持暂停状态，拒绝客户机连接。主管代理启动后，暂停的代理将自动进入正常运行状态。

方法 1：不使用群集配置文件连接

► 将代理连接到群集

1. 使用带有 `-cluster` 选项的 `imqbrokerd` 命令来启动代理，并将（要连接的）代理的完整列表指定为 `-cluster` 选项的变量。
2. 请在启动要连接到群集的每个代理时执行该操作。

例如，以下命令启动了一个新代理，并将其连接到运行于 `host1` 的默认端口上的代理、运行于 `host2` 的 `7677` 端口上的代理和运行于 `localhost` 的 `7678` 端口上的代理。

```
imqbrokerd -cluster host1,host2:7677,:7678
```

方法 2：使用群集配置文件连接

也可以创建一个群集配置文件，来指定要连接的代理列表（可选：指定主管代理的地址）。这种定义群集的方法更适合生产系统。如果使用此方法，必须将群集中的每个代理的 `imq.cluster.url` 属性值设置为指向群集配置文件。

安全的交叉代理连接

如果希望在群集的代理之间传送安全、加密的消息，您需要按照下面的步骤配置 `cluster` 连接服务，以使用基于 SSL 的传输协议。

► 配置群集内安全连接

1. 对于群集中的各个代理，请设置基于 SSL 的连接服务。

请参见第 198 页的“设置通过 TCP/IP 的基于 SSL 的服务”中的说明。

2. 将 `imq.cluster.transport` 群集配置属性设置为 `ssl`。

如果未使用群集配置文件，您需要为群集中的各个代理设置此属性。

管理群集中的代理

设置代理群集后，可能需要添加新代理、重新启动群集中原有的代理，或者从群集中删除代理。

将代理添加到群集

► 将新代理添加到现有群集

- 如果使用群集配置文件，则
 - a. 将新代理添加到群集配置文件中的 `imq.cluster.brokerlist` 属性。
 - b. 对群集中的各个代理执行以下命令。


```
imqcmd reload cls
```

该命令强制所有的代理重新装入 `imq.cluster.brokerlist` 属性，以保证群集中代理的所有持久性信息都是最新的。
 - c. 启动新代理时，使用命令行中的 `-D` 选项指定 `imq.cluster.url` 属性。

这会将代理指向群集配置文件。
- 如果没有使用群集配置文件，那么启动新代理时，在命令行中使用 `-D` 选项指定 `imq.cluster.brokerlist`、`imq.cluster.transport`（如果使用安全群集连接服务）和 `imq.cluster.masterbroker` 属性（如果需要）。

重新启动群集中的代理

如果群集中的代理因某种原因崩溃或关闭，您需要将其作为群集的成员重新启动。

► 重新启动现有群集的现有代理

- 如果不是使用群集配置文件定义群集，那么重新启动代理时，请在命令行中使用 `-D` 选项指定 `imq.cluster.brokerlist` 属性（如有必要，请一并指定 `imq.cluster.masterbroker` 属性）。如果群集中没有主管代理，可以在重新启动代理仅使用 `-cluster` 选项来指定群集中的代理列表。
- 如果使用群集配置文件定义群集，请在用于启动代理的命令行中使用 `-D` 选项来指定 `imq.cluster.url` 属性。

从群集中删除代理

► 从现有群集中删除代理

- 如果代理 A、B 和 C 均是使用以下命令行启动，那么仅仅重新启动 A 将无法将其从群集中删除。

```
imqbrokerd -cluster A,B,C
```

而是需要使用以下命令行重新启动所有的其他代理：

```
imqbrokerd -cluster B,C
```

然后，需要启动代理 A，且不指定 `-cluster` 选项。

- 如果代理列表是使用群集配置文件来指定，那么需要执行以下操作：
 - 从配置文件中删除该代理的相关内容。
 - 更改或删除要删除的代理的 `imq.cluster.url` 属性，这样代理就不会再使用该通用属性。
 - 使用 `imqcmd reload cls` 命令强制所有的代理重新装入它们的群集配置，从而重新配置群集。

管理主管代理的配置更改记录

每个群集都可以拥有一个主管代理，它将记录群集的持久性状态的所有更改。此状态包括长期订阅和管理员创建的物理目标等有关信息。所有的代理在启动时都会咨询主管代理（代理主管依次查询其配置更改记录），以同步关于这些持久性对象的信息。因此，主管代理的故障将使得类似同步变得不可能。其结果是，如果主管代理失败，那么您就无法创建或删除物理目标或长期订阅。

因为主管代理的配置更改记录包含着重要的信息，所以定期对其进行备份，并在出现故障时予以恢复，这一点非常重要。

以下几节介绍如何备份和恢复配置更改记录。

备份配置更改记录

► 备份配置更改记录

使用 `imqbrokerd` 命令的 `-backup` 选项。例如，

```
imqbrokerd -backup mybackuplog
```

请及时进行如上的备份操作，这很重要。恢复一个很旧的备份可能会导致丢失信息：最后一次完成备份后对物理目标或长期订阅的所有更改都将丢失。

恢复配置更改记录

► 万一发生故障时恢复主管代理

1. 关闭群集中的所有代理。
2. 用以下命令恢复主管代理的配置更改记录：

```
imqbrokerd -restore mybackuplog
```
3. 如果要给主管代理分配新名称或端口号，必须更新群集配置文件，指定主管代理是群集中的代理之一，并指定它的新名称（使用 `imq.cluster.masterbroker` 属性）。
4. 重新启动所有代理。

恢复代理将不可避免地导致某些过时的数据重新装入到代理的配置更改记录中。不过，只要经常进行前一小节所介绍的定期备份，可以将这个问题的影响减到最小。

因为主管代理记录对持久性对象的所有更改的历史记录，所以每过一段时间其数据库就会有显著的扩充。备份和恢复操作有利于压缩和优化该数据库。

日志记录

本节介绍了代理的默认日志记录配置，并说明了如何更改该配置，从而将日志信息重定向到替代输出通道以及如何更改日志文件转移标准。关于日志记录的介绍，请参见第 64 页的“日志记录器”。有关使用日志记录报告代理度量依据的信息，请参见第 220 页的“监视工具”。

默认日志记录配置

启动代理时，系统自动将其配置为将日志输出保存到位于特定目录中的一组滚动生成的日志文件中，该目录的名称由与此日志文件相关联的代理实例的名称 (`instanceName`) 标识（请参见附录 A “Message Queue 数据的位置”）：

```
.../instances/instanceName/log/
```

日志文件是纯文本文件。它们的名称如下（从最早生成到最近生成）：

```
log.txt
log_1.txt
log_2.txt
...
log_9.txt
```

默认情况下，日志文件每星期转移一次；系统维护九个备份文件。

- 要更改保存日志文件的目录，请将 `img.log.file.dirpath` 属性设置为所需的路径。
- 要将日志文件的根名称由 `log` 改为其他名称，请设置 `img.log.file.filename` 属性。

代理支持三种日志种类：ERROR、WARNING 和 INFO（请参见第 64 页的表 2-7）。设置日志记录级别后，将收集所有高于和等于该级别的消息。默认日志级别是 INFO。这表示默认情况下将记录所有 ERROR、WARNING 和 INFO 消息。

日志消息格式

已记录的消息由时间戳（请参见第 66 页的表 2-9 了解如何更改时间戳时区）、消息代码和消息本身组成。信息的大小取决于所设置的日志级别。以下是一个 INFO 消息的示例。

```
[13/Sep/2000:16:13:36 PDT] B1004 Starting the broker service using tcp [
25374,100] with min threads 50 and max threads of 500
```

更改日志记录器配置

第 66 页的表 2-9 说明了所有的日志记录器属性。

► 更改代理的日志记录器配置

1. 设置日志级别。
2. 设置一个和多个日志记录种类的输出通道（文件、控制台或两者）。
3. 如果将输出记录到文件中，请为该文件配置转移标准。

可以通过设置日志记录器属性完成上述步骤。可以使用以下两种方法之一来进行设置：

- 启动代理之前，在该代理的 `config.properties` 文件中更改或添加日志记录器属性。
- 在启动代理的 `imgbrokerd` 命令中指定日志记录器命令行选项。也可以使用代理选项 `-D` 来更改日志记录器属性（或任何代理属性）。

在命令行上传送的选项将覆盖代理实例配置文件中指定的属性。表 5-4 列出了影响日志记录的 imqbrokerd 选项。

表 5-4 imqbrokerd 日志记录器选项及相应的属性

imqbrokerd 选项	说明
-metrics <i>interval</i>	指定度量依据信息写入日志记录器的时间间隔（以秒为单位）。
-loglevel <i>level</i>	设置日志级别为以下三项之一：ERROR、WARNING 或 INFO。
-silent	停止向控制台记录日志信息。
-tty	将所有消息发送到控制台。默认情况下，只显示 WARNING 和 ERROR 级别的消息。

以下小节介绍了如何更改默认配置，以执行以下操作：

- 更改输出通道（日志消息的目标）
- 更改转移标准

更改输出通道

默认情况下，错误和警告消息除了记录到日志文件中以外，还会显示在终端上。（在 Solaris 中，错误消息还会写入到系统的 syslog 守护程序中。）

可以用以下方式更改日志消息的输出通道：

- 要在屏幕上显示所有的日志种类（对于给定的级别）输出，请在 imqbrokerd 命令中使用 -tty 选项。
- 要阻止在屏幕上显示日志输出，请在 imqbrokerd 命令中使用 -silent 选项。
- 使用 imq.log.file.output 属性指定将哪些种类的日志记录信息写入到日志文件。例如，
imq.log.file.output=ERROR
- 使用 imq.log.console.output 属性指定将哪些种类的日志记录信息写入到控制台。例如，
imq.log.console.output=INFO
- 在 Solaris 中，使用 imq.log.syslog.output 属性指定将哪些种类的日志记录信息写入到 Solaris syslog 中。例如，
imq.log.syslog.output=NONE

注意

在更改日志记录器输出通道前，必须确保日志记录级别已设置为支持映射到输出通道的信息。例如，如果将日志级别设置为 `ERROR`，然后将 `imq.log.console.output` 属性设置为 `WARNING`，那么将不会记录任何消息，因为没有启用对 `WARNING` 消息的记录。

更改日志文件转移标准

转移日志文件有两个标准：时间和大小。默认是使用时间标准，每七天转移一次文件。

- 要更改时间间隔，需要更改 `imq.log.file.rolloversecs` 属性。例如，以下属性定义将时间间隔更改为十天：

```
imq.log.file.rolloversecs=864000
```

- 要将转移标准更改为取决于文件大小，需要设置 `imq.log.file.rolloverbytes` 属性。例如，以下定义将在文件达到 500,000 字节的限制时将代理定向到转移文件。

```
imq.log.file.rolloverbytes=500000
```

如果同时设置与时间相关和与大小相关的转移属性，首先达到的限制将触发转移。如前所述，代理最多维护九个转移文件。

代理和应用程序管理

本章介绍如何执行与管理代理及其提供的服务相关的任务。某些任务并不针对特定的客户机应用程序。这些任务包括：

- 控制代理的状态：可以暂停、恢复、关闭和重新启动代理。
- 查询和更新代理属性
- 管理连接服务

其他代理任务针对特定的应用程序执行，包括管理物理目标、长期订阅和事务等任务：

- **Message Queue** 消息通过代理目标路由到接收者或订阅者。您负责在代理上创建这些目标。
- **Message Queue** 负责为长期订阅者分配并维护资源，即使进行长期订阅的客户机处于非活动状态。使用 **Message Queue** 命令工具可以获得长期订阅的相关信息，也可以销毁长期订阅或删除其中的消息以节约 **Message Queue** 资源。
- **Message Queue** 事务和分布式事务由代理来跟踪。如果出现错误，可能需要手动提交或回滚事务。

本章介绍如何使用命令行实用程序 (`imqcmd`) 来执行这些任务。许多相同的任务还可以使用管理控制台（**Message Queue** 消息服务器的图形界面）来完成。有关详细信息，请参见第 4 章 [“管理控制台教程”](#)。。

命令行实用程序

命令行实用程序可用于管理代理及其提供的服务。本节介绍了基本的 `imqcmd` 命令语法，提供了一个子命令列表，并概述了 `imqcmd` 的选项。后续各节将介绍如何使用这些命令完成特定的任务。

imqcmd 命令语法

`imqcmd` 命令的一般语法如下：

```
imqcmd subcommand argument [options]
imqcmd -h|H
imqcmd -v
```

请注意，如果指定 `-v`、`-h` 或 `-H` 选项，将不会执行命令行中指定的子命令。例如，输入以下命令将显示版本信息，而不执行 `restart` 子命令。

```
imqcmd restart bkr -v
```

imqcmd 子命令

表 6-1 列出了命令行实用程序 (`imqcmd`) 包含的子命令：本章涉及到的面向任务的几节中对子命令进行了详细说明。

表 6-1 `imqcmd` 子命令

子命令和参数	说明
<code>commit txn</code>	提交事务。
<code>compact dst</code>	压缩一个或多个目标的内置的、基于文件的数据存储。
<code>create dst</code>	创建目标。
<code>destroy dst</code>	销毁目标。
<code>destroy dur</code>	销毁长期订阅。
<code>list cxn</code>	列出代理的连接。
<code>list dst</code>	列出代理中的目标。
<code>list dur</code>	列出对主题的长期订阅。
<code>list svc</code>	列出代理中的服务。
<code>list txn</code>	列出代理中的事务。

表 6-1 imqcmd 子命令 (续)

子命令和参数	说明
metrics bkr	显示代理度量依据。
metrics dst	显示目标度量依据。
metrics svc	显示服务度量依据。
pause bkr	暂停代理中的所有服务。
pause dst	暂停代理中的一个或多个目标。
pause svc	暂停代理中的单项服务。
purge dst	清除目标中的所有消息，但不销毁该目标。
purge dur	清除长期订阅中的所有消息，但不销毁该长期订阅。
query bkr	查询并显示代理中的信息。
query cxn	查询并显示连接中的信息。
query dst	查询并显示目标中的信息。
query svc	查询并显示服务中的信息。
query txn	查询并显示事务中的信息。
reload cls	重新装入代理群集配置。
restart bkr	重新启动当前运行的代理实例。不能用于启动新的代理实例。
resume bkr	恢复代理中的所有服务。
resume dst	恢复代理中的一个或多个暂停的目标。
resume svc	恢复一项服务。
rollback txn	回滚某项事务。
shutdown bkr	关闭代理实例。关闭代理实例后可以使用 imqbrokerd 命令重新启动，但不能使用 imqcmd 的 restart bkr 子命令启动。
update bkr	更新代理属性。
update dst	更新目标属性。
update svc	更新服务属性。

imqcmd 选项概述

表 6-2 列出了 imqcmd 命令的选项。有关其用法的论述，请参见以下基于任务的各节。

表 6-2 imqcmd 选项

选项	说明
-b <i>hostName:port</i>	指定代理的主机名及其端口号。默认值为 localhost:7676。 要仅指定端口，请使用：-b :7878 要仅指定主机名，请使用：-b somehost
-c <i>clientID</i>	指定订阅某个主题的长期订阅者的 ID。请参见第 161 页的“管理长期订阅”。
-d <i>destinationName</i>	指定主题名。与 list dur 和 destroy dur 子命令一起使用。请参见第 161 页的“管理长期订阅”。
-f	执行操作，而无需用户确认。
-h	显示使用帮助。不在命令行上执行其他命令。
-H	显示使用帮助、属性列表和示例。不在命令行上执行其他命令。
-int <i>interval</i>	指定 metrics bkr、metrics dst 和 metrics svc 子命令显示度量依据输出的时间间隔，以秒为单位。
-j <i>javahome path</i>	指定要使用的替代 Java 2 兼容运行时（默认情况下使用系统上的运行时或 Message Queue 附带的运行时）。
-m <i>metricType</i>	指定要显示的度量依据信息的类型。将此选项与 metrics dst、metrics svc 或 metrics bkr 子命令一起使用。metricType 的值取决于为目标、服务还是代理生成度量依据。
-msp <i>numSamples</i>	指定 metrics bkr、metrics dst 和 metrics svc 子命令在其度量依据输出中显示的度量依据样例的数量。
-n <i>argumentName</i>	指定子命令参数的名称。根据子命令的不同，它可能是服务名称、物理目标、长期订阅、连接 ID 或事务 ID。
-o <i>attribute=value</i>	指定属性值。根据子命令参数的不同，它可能是代理属性（请参见第 142 页的“管理代理”）、服务属性（请参见第 147 页的“管理连接服务”）或目标属性（请参见第 152 页的“管理目标”）。
-p <i>password</i>	指定管理员的密码。如果省略此值，系统会提示您输入密码。
-pst <i>pauseType</i>	指定在暂停目标时是暂停生成方、使用方，还是二者都暂停。请参见第 152 页的“管理目标”。
-rtm <i>timeout</i>	指定 imqcmd 子命令的初始（重试）超时周期（以秒为单位）。超时是 imqcmd 子命令在向代理发出请求之后等待的时间长度。该子命令随后的每次重试将使用初始超时周期的倍数作为超时值。默认值：10

表 6-2 imqcmd 选项 (续)

选项	说明
-rtr numRetries	指定 imqcmd 子命令首次超时之后尝试重试的次数。默认值：5
-s	无提示模式。不显示任何输出。
-secure	使用 ssladmin 连接服务指定代理的安全管理连接（请参见第 202 页的“步骤 4: 配置并运行基于 SSL 的客户机”）。
-svn serviceName	指定列出其连接的服务。请参见第 151 页的“获得连接信息”。
-t destType	指定目标类型：t（主题）或 q（队列）。请参见第 152 页的“管理目标”。
-tmp	显示临时目标。请参见第 152 页的表 6-9。
-u userName	指定管理员名称。如果省略此值，系统会提示您输入用户名。
-v	显示版本信息。不在命令行上执行其他命令。

每次发出一个 imqcmd 子命令时，都必须指定以下选项：主机名和端口号 (-b)、用户名 (-u)、密码 (-p) 和安全连接 (-secure)。如果不指定主机名和端口号，系统将使用默认值。如果不指定用户名和密码，系统会提示您输入这些信息。如果不指定 -secure，连接将是不安全的。

注意 要使用 -secure 选项，必须首先设置并启用目标代理实例中的 ssladmin 服务，如第 198 页的“设置通过 TCP/IP 的基于 SSL 的服务”中所述。

使用 imqcmd 命令

要使用 imqcmd 命令管理代理，必须执行以下操作：

- 使用 imqbrokerd 命令启动代理。
请参见第 123 页的“启动代理”。命令行实用程序只能用于管理正在运行的代理，不能用于启动代理。
- 如果代理没有在本地主机的 7676 端口上运行，必须使用 -b 选项指定目标代理。
- 指定适当的管理员用户名和密码。如果不指定，系统会提示您输入这些信息。总之请记住，使用 imqcmd 执行的每个操作都要在用户信息库中进行验证。有关详细信息，请参见第 184 页的“验证用户”。

安装 Message Queue 时，系统会安装默认的用户信息库，这是一个纯文本文件。信息库出厂时包括两个条目：一个供管理员用户使用，另一个供临时用户使用。使用这两个条目即可建立代理实例连接，无需执行额外的操作。例如，如果只是想测试 Message Queue，可以使用默认的用户名和密码 (admin/admin) 来运行 imqcmd 实用程序。

如果要设置生产系统，则需要完成一些额外的工作，对管理用户进行验证和授权（请参见第 8 章“管理安全性”）。特别是，需要在 Message Queue 用户信息库中添加条目（请参见第 184 页的“使用文本文件用户信息库”）。也可以将 LDAP 目录服务器作为用户信息库（请参见第 190 页的“使用 LDAP 服务器管理用户信息库”）。

imqcmd 用法示例

以下示例说明了 imqcmd 命令的用法：

- 列出运行在 localhost 7676 端口上的代理的属性：
`imqcmd query bkr -u admin -p admin`
- 列出运行在 myserver 1564 端口上的代理的属性；用户名为 alladin，用户密码为 abracadabra。
`imqcmd query bkr -b myserver:1564 -u alladin -p abracadabra`

如果指定用户名 alladin 属于 admin 组，将以管理员客户机的身份连接到指定代理。

- 列出运行在 localhost 7676 端口上的代理的属性，且将命令的初始超时设置为 20 秒，重试（超时后）次数为 7。
`imqcmd query bkr -u admin -p admin -rtm 20 -rtr 7`

管理代理

命令实用程序包含的子命令可用于执行以下代理管理任务：

- [显示代理信息](#)
- [更新代理属性](#)
- [显示代理度量依据](#)
- [控制代理状态](#)

要管理代理的连接服务，请参见第 147 页的“管理连接服务”。要管理代理目标，请参见第 152 页的“管理目标”。

表 6-3 列出了用于管理代理的 `imqcmd` 子命令。如果没有指定主机名或端口，则假定默认值为 `(localhost:7676)`。

表 6-3 用于管理代理的 `imqcmd` 子命令

子命令语法	说明
<code>metrics bkr [-b <i>hostName:port</i>]</code> <code>[-m <i>metricType</i>]</code> <code>[-int <i>interval</i>]</code> <code>[-msp <i>numSamples</i>]</code>	显示默认代理或指定主机和端口上代理的代理度量依据。 请使用 <code>-m</code> 选项指定要显示的度量依据类型： ttl 显示消息和数据包流入和流出代理的度量依据。（默认度量依据类型） rts 显示消息和数据包流入和流出代理的速率（每秒）度量依据。 cxn 显示连接、虚拟内存堆和线程。 使用 <code>-int</code> 选项指定显示度量依据的时间间隔（以秒为单位）。默认值为 5 秒。 使用 <code>-msp</code> 选项指定在输出中显示的样例数量。默认值为 unlimited 数量（无限）。
<code>pause bkr [-b <i>hostName:port</i>]</code>	暂停默认代理或指定主机和端口上的代理。请参见第 145 页的“暂停和恢复代理”。
<code>query bkr -b <i>hostName:port</i></code>	列出默认代理或指定主机和端口上的代理的当前属性设置。还将列出与指定代理连接且正在运行的代理（在多个代理群集中）。
<code>reload cls</code>	仅适用于代理群集。强制群集中的所有代理重新装入 <code>imq.cluster.brokerlist</code> 属性并更新群集信息。有关详细信息，请参见第 131 页的“将代理添加到群集”。
<code>restart bkr [-b <i>hostName:port</i>]</code>	关闭并重新启动默认代理或指定主机和端口上的代理。 请注意，此命令将使用第一次启动该代理时指定的选项重新启动该代理。如果要执行不同的操作，必须先关闭代理，然后再使用所需的选项重新启动它。
<code>resume bkr [-b <i>hostName:port</i>]</code>	恢复默认代理或指定主机和端口上的代理。
<code>shutdown bkr [-b <i>hostName:port</i>]</code>	关闭默认代理或指定主机和端口上的代理。
<code>update bkr [-b <i>hostName:port</i>]</code> <code>-o <i>attribute=value</i></code> <code>[-o <i>attribute=value1</i>] ...</code>	更改默认代理或指定主机和端口上的代理的指定属性。

请记住，使用表 6-3 中列出的子命令时，除非您希望代理在 localhost 7676 端口上运行，否则必须指定代理主机名和端口号。

显示代理信息

要查询并显示某个代理的信息，请使用 query bkr 子命令。例如：

```
imqcmd query bkr -u admin -p admin
```

此命令语句的输出如下：

Version	3.5 SP1
Instance Name	imqbroker
Primary Port	7676
Current Number of Messages in System	0
Current Total Message Bytes in System	0
Max Number of Messages in System	unlimited (-1)
Max Total Message Bytes in System	unlimited (-1)
Max Message Size	70m
Auto Create Queues	true
Auto Create Topics	true
Auto Created Queue Max Number of Active Consumers	1
Auto Created Queue Max Number of Backup Consumers	0
Cluster Broker List (active)	
Cluster Broker List (configured)	
Cluster Master Broker	
Cluster URL	
Log Level	INFO
Log Rollover Interval (seconds)	604800
Log Rollover Size (bytes)	unlimited (-1)

更新代理属性

可以使用 update bkr 子命令更新表 6-4 中列出的任何代理属性。请注意，对代理的更新会自动写入代理的实例配置文件。

表 6-4 由 imqcmd 更新的代理属性

属性	参考
imq.autocreate.queue	第 70 页的表 2-10

表 6-4 由 imqcmd 更新的代理属性 (续)

属性	参考
imq.autocreate.topic	第 70 页的表 2-10
imq.autocreate.queue.maxNumActiveConsumers	第 70 页的表 2-10
imq.autocreate.queue.maxNumBackupConsumers	第 70 页的表 2-10
imq.cluster.url	第 128 页的表 5-3.
imq.log.level	第 66 页的表 2-9
imq.log.file.rolloversecs	第 66 页的表 2-9
imq.log.file.rolloverbytes	第 66 页的表 2-9
imq.system.max_count	第 56 页的表 2-4
imq.system.max_size	第 56 页的表 2-4
imq.message.max_size	第 56 页的表 2-4
imq.portmapper.port	第 52 页的表 2-3

例如，以下命令可关闭队列目标的自动创建：

```
imqcmd update bkr -o "imq.autocreate.queue=false"
-u admin -p admin
```

控制代理状态

启动代理后，可以使用下列 imqcmd 子命令控制其状态。

暂停和恢复代理

- **暂停代理。**暂停代理将暂停代理的连接服务线程，从而使代理停止监听连接端口。其结果是代理将无法接受新连接、接收消息、发送消息。

但是，暂停代理不会暂停 **admin** 连接服务，允许执行控制代理消息流所需的管理任务。例如，当某个目标受到消息攻击时，您可以暂停该代理，然后执行下列操作之一，以期解决该问题：跟踪消息来源、限制目标的大小或销毁目标。

暂停代理也不会暂停群集连接服务。但是，群集中的消息传送依赖于群集中不同代理所执行的传送功能。

以下命令将暂停运行在 myhost 1588 端口上的代理。

```
imqcmd pause bkr -b myhost:1588 -u admin -p admin
```

（也可暂停单个连接服务，请参见第 150 页的“暂停和恢复连接服务”，或暂停单个目标，请参见第 158 页的“暂停和恢复目标”）

- **恢复代理。**恢复代理将重新激活代理服务线程，使代理恢复监听端口。以下命令将恢复运行在 localhost 7676 端口上的代理。

```
imqcmd resume bkr -u admin -p admin
```

关闭并重新启动代理

- **关闭代理。**关闭代理将终止代理进程。这属于正常终止：代理停止接收新的连接和消息，完成现有消息的传送后将终止代理进程。以下命令将关闭运行在 ctrlsrv 1572 端口上的代理。

```
imqcmd shutdown bkr -b ctrlsrv:1572 -u admin -p admin
```

- **重新启动代理。**关闭并重新启动代理。以下命令将重新启动运行在 localhost 7676 端口上的代理。

```
imqcmd restart bkr -u admin -p admin
```

显示代理度量依据

要显示有关代理的度量依据信息，请使用 metrics bkr 子命令。例如，要获得消息在 10 秒间隔内流入和流出代理的速率，请使用以下命令：

```
imqcmd metrics bkr -m rts -int 10 -u admin -p admin
```

此命令语句的输出如下：

Msgs/sec		Msg Bytes/sec		Pkts/sec		Pkt Bytes/sec	
In	Out	In	Out	In	Out	In	Out
0	0	27	56	0	0	38	66
10	0	7365	56	10	10	7457	1132
0	0	27	56	0	0	38	73
0	10	27	7402	10	20	1400	8459
0	0	27	56	0	0	38	73

有关使用 imqcmd 报告代理度量依据的详细说明，请参见第 220 页的“监视工具”。

管理连接服务

命令行实用程序包含的子命令可用于执行以下连接服务管理任务：

- 列出连接服务
- 显示连接服务信息
- 更新连接服务属性
- 显示连接服务度量依据
- 暂停和恢复连接服务

有关 Message Queue 连接服务的概述信息，请参见第 50 页的“连接服务”。

表 6-5 列出了用于管理连接服务的 `imqcmd` 子命令。如果未指定主机名和端口，则会假定默认值 (`localhost:7676`)。

表 6-5 用于管理连接服务的 `imqcmd` 子命令

子命令语法	说明
<code>list svc [-b <i>hostName:port</i>]</code>	列出默认代理或指定主机和端口上的代理中的所有连接服务。
<code>metrics svc -n <i>serviceName</i></code> <code>[-b <i>hostName:port</i>]</code> <code>[-m <i>metricType</i>]</code> <code>[-int <i>interval</i>]</code> <code>[-msp <i>numSamples</i>]</code>	显示默认代理或指定主机和端口上的代理中指定服务的度量依据。 使用 <code>-m</code> 选项指定要显示的度量依据类型： ttl 显示消息和数据包按指定服务方式流入和流出代理的度量依据。（默认度量依据类型） rts 显示消息和数据包按指定服务方式流入和流出代理的速率（每秒）度量依据。 cxn 显示连接、虚拟内存堆和线程。 使用 <code>-int</code> 选项指定显示度量依据的时间间隔（以秒为单位）。默认值为 5 秒。 使用 <code>-msp</code> 选项指定在输出中显示的样例数量。默认值为非限制数量（无限）。
<code>pause svc -n <i>serviceName</i></code> <code>[-b <i>hostName:port</i>]</code>	暂停运行在默认代理或指定主机和端口上的代理中的指定服务。不能暂停管理服务。
<code>query svc -n <i>serviceName</i></code> <code>[-b <i>hostName:port</i>]</code>	显示运行在默认代理或指定主机和端口上的代理中的指定服务的信息。
<code>resume svc -n <i>serviceName</i></code> <code>[-b <i>hostName:port</i>]</code>	恢复运行在默认代理或指定主机和端口上的代理中的指定服务。

表 6-5 用于管理连接服务的 imqcmd 子命令（续）

子命令语法	说明
update svc -n serviceName [-b hostName:port] -o attribute=value [-o attribute=value1]...	更新运行在默认代理或指定主机和端口上的代理中的指定服务的指定属性。有关服务属性的说明，请参见第 149 页的表 6-7。

代理支持与应用程序客户机和管理客户机的连接。表 6-6 显示了 Message Queue 代理当前支持的连接服务。“服务名称”一栏中的值用于为 -n 选项指定服务名称。如该表所示，每项服务都是通过它使用的服务类型 - NORMAL（应用程序客户机）或 ADMIN（管理客户机）和底层传输层来指定的。

表 6-6 代理支持的连接服务

服务名称	服务类型	协议类型
jms	NORMAL	tcp
ssljms（企业版）	NORMAL	tls（其安全性基于 SSL）
httpjms（企业版）	NORMAL	http
httpsjms（企业版）	NORMAL	https（其安全性基于 SSL）
admin	ADMIN	tcp
ssladmin（企业版）	ADMIN	tls（其安全性基于 SSL）

列出连接服务

要列出某个代理中可用的连接服务，请使用如下命令：

```
imqcmd list svc [-b hostName:portNumber] -u admin -p admin
```

例如，以下命令将列出运行在主机 myServer（位于 6565 端口）上的代理所支持的服务。

```
imqcmd list svc -b MyServer:6565 -u admin -p admin
```

以下命令将列出运行在 localhost 7676 端口上的代理支持的所有服务：

```
imqcmd list svc -u admin -p admin
```

该命令的输出信息如下：

Service Name	Port Number	Service State
admin	41844 (dynamic)	RUNNING
httpjms	-	UNKNOWN
httpsjms	-	UNKNOWN
jms	41843 (dynamic)	RUNNING
ssladmin	dynamic	UNKNOWN
ssljms	dynamic	UNKNOWN

显示连接服务信息

要查询并显示某项服务的信息，请使用 `query` 子命令。例如，

```
imqcmd query svc -n jms -u admin -p admin
```

此命令语句的输出如下：

Service Name	jms
Service State	RUNNING
Port Number	60920 (dynamic)
Current Number of Allocated Threads	0
Current Number of Connections	0
Min Number of Threads	10
Max Number of Threads	1000

更新连接服务属性

可以使用 `update` 子命令更改表 6-7 中列出的一个或多个服务属性的值。

表 6-7 由 `imqcmd` 更新的连接服务属性

属性	说明
port	为要更新的服务指定的端口（不适用于 <code>httpjms</code> 和 <code>httpsjms</code> ）。0 值表示端口由端口映射器动态分配。
minThreads	为服务指定的最小线程数。
maxThreads	为服务指定的最大线程数。

以下命令将指定给 jms 服务的最小线程数更改为 20。

```
imqcmd update svc -n jms -o "minThreads=20"
```

显示连接服务度量依据

要显示有关某个服务的度量依据信息，请使用 metrics 子命令。例如，要获得 jms 连接服务处理的消息和数据包的累计总数，请使用以下命令：

```
imqcmd metrics svc -n jms -m ttl -u admin -p admin
```

此命令语句的输出如下：

Msgs		Msg Bytes		Pkts		Pkt Bytes	
In	Out	In	Out	In	Out	In	Out
164	100	120704	73600	282	383	135967	102127
657	100	483552	73600	775	876	498815	149948

有关使用 imqcmd 报告连接服务度量依据的详细说明，请参见第 220 页的“监视工具”。

暂停和恢复连接服务

要暂停除管理服务之外的其他服务（不能暂停管理服务），请使用如下命令：

```
imqcmd pause svc -n serviceName -u admin -p admin
```

暂停服务影响如下：

- 代理将停止接受暂停服务的新客户机连接。如果 Message Queue 客户机尝试打开新的连接，将出现异常。
- 暂停服务的所有现有连接将保持活动状态，但是代理将暂停这些连接上的所有消息处理，直到服务恢复。（例如，如果客户机尝试发送消息，将禁止使用 send() 方法，直到服务恢复。）
- 会保留代理已经接收的任何消息的消息传送状态。（例如，服务恢复之后，事务不会中断，消息传送将恢复。）

要恢复某项服务，请使用如下命令：

```
imqcmd resume svc -n serviceName -u admin -p admin
```

获得连接信息

命令实用程序包含的子命令可用于列出并获得有关连接的信息。

表 6-8 列出了应用于连接的 `imqcmd` 子命令。如果没有指定主机名和端口，则会假定为 `localhost` 和 `7676`。

表 6-8 用于管理连接服务的 `imqcmd` 子命令

子命令语法	说明
<code>list cxn [-svn serviceName]</code> <code> [-b hostName:port]</code>	列出默认代理或指定主机和端口上的代理中指定服务名称的所有连接。如果未指定服务名称，将列出所有连接。
<code>query cxn -n connectionID</code> <code> [-b hostName:port]</code>	显示默认代理或指定主机和端口上的代理中的指定连接的信息。

要查询并显示某项连接服务的信息，请使用 `query` 子命令。例如，

```
imqcmd query cxn -n 421085509902214374 -u admin -p admin
```

此命令语句的输出如下：

Connection ID	421085509902214374
User	guest
Service	jms
Producers	0
Consumers	1
Host	111.22.333.444
Port	60953
Client ID	
Client Platform	

管理目标

所有 Message Queue 消息都是通过在特定代理上创建的队列和主题目标路由到其使用方客户机上。

命令行实用程序包含的子命令可用于执行以下目标管理任务：

- [创建目标](#)
- [列出目标](#)
- [显示目标信息](#)
- [更新目标属性](#)
- [显示目标度量依据](#)
- [暂停和恢复目标](#)
- [清除目标](#)
- [销毁目标](#)
- [压缩目标](#)

有关目标的介绍，请参见第 68 页的“物理目标”。

表 6-9 概述了 imqcmd 目标子命令。如果不使用默认代理 (localhost:7676)，则必须指定主机名和端口。

表 6-9 用于管理目标的 imqcmd 子命令

子命令语法	说明
<code>compact dst [-t destType -n destName]</code>	为指定类型和名称的目标压缩内置的、基于文件的数据存储。如果未指定目标类型和名称，则会压缩所有目标。在压缩之前必须暂停目标。
<code>create dst -t destType -n destName [-o attribute=value] [-o attribute=value1]...</code>	创建指定类型、名称和属性的目标。目标名称必须只包含字母数字字符（不包括空格），可以以字母字符或 "_" 和 "\$" 字符开头。但不能以字符串 "mq." 开头。
<code>destroy dst -t destType -n destName</code>	销毁指定类型和名称的目标。

表 6-9 用于管理目标的 imqcmd 子命令（续）

子命令语法	说明
<code>list dst [-t <i>destType</i>] [-tmp]</code>	列出指定类型的所有目标，同时包含临时目标的选项（请参见第 72 页的“临时目标”）。 类型参数可有两个值： <i>destType</i> =q（队列） <i>destType</i> =t（主题） 如果未指定类型，将列出所有类型的所有目标。
<code>metrics dst -t <i>destType</i> -n <i>destName</i> [-m <i>metricType</i>] [-int <i>interval</i>] [-msp <i>numSamples</i>]</code>	显示指定类型和名称的目标的度量依据信息。 使用 -m 选项指定要显示的度量依据类型： ttl 显示消息和数据包流入和流出目标以及驻留内存的度量依据。（默认度量依据类型） rts 显示消息和数据包（每秒）流入和流出目标的速率度量依据及其他比率信息。 con 显示使用方相关度量依据。 dsk 显示磁盘使用情况度量依据。 使用 -int 选项指定显示度量依据的时间间隔（以秒为单位）。默认值为 5 秒。 使用 -msp 选项指定在输出中显示的样例数量。默认值为无限数量（无限）。
<code>pause dst [-t <i>destType</i> -n <i>destName</i>] [-pst <i>pauseType</i>]</code>	对于指定类型和名称的目标，暂停将消息传送给使用方（-pst CONSUMERS），或暂停从生成方传送消息（-pst PRODUCERS），或二者同时暂停（-pst ALL）。如果未指定目标类型和名称，则暂停所有目标。默认值为 ALL。
<code>purge dst -t <i>destType</i> -n <i>destName</i></code>	清除指定类型和名称的目标中的消息。
<code>query dst -t <i>destType</i> -n <i>destName</i></code>	列出指定类型和名称的目标的相关信息。
<code>resume dst [-t <i>destType</i> -n <i>destName</i>]</code>	恢复指定类型和名称的暂停目标中的消息传送。如果未指定目标类型和名称，则恢复所有目标。
<code>update dst -t <i>destType</i> -n <i>destName</i> -o <i>attribute=value</i> [-o <i>attribute=value1</i>]...</code>	更新指定目标上指定属性的值。 属性名称可以是表 6-10 中列出的任意属性。

创建目标

创建目标时，必须指定以下内容：

- 目标类型：主题或队列
- 目标名称：必须只包含字母数字字符（不包括空格），可以以字母字符或 "_" 和 "\$" 字符开头。但不能以字符串 "mq." 开头。
- 目标属性的任意非默认值

很多目标属性用于管理代理内存资源和消息流。例如，可指定目标允许的最多生成方数量或目标允许的最多消息数（或最大消息量）。这些限制与使用代理配置属性基于代理范围进行设置的限制类似（请参见第 55 页的“管理内存资源和消息流”）。还可指定达到这些限制范围时代理的响应方式。

还有一些目标属性仅可应用于队列目标。这些属性用于指定在向多个使用方传送负荷平衡消息中使用的活动和备份使用方的数量（请参见第 68 页的“队列目标”）。

表 6-10 说明了应用于每种目标的属性。可在创建或更新目标时设置属性值。对于自动创建的目标，可在代理的实例配置文件中设置默认属性值（请参见第 115 页的“配置文件”）。

表 6-10 目标属性

目标类型	属性	默认值	说明
队列和主题	maxNumMsgs ¹	-1 (无限制)	指定目标中允许的未使用消息的最大数量。
队列和主题	maxTotalMsgBytes ¹	-1 (无限制)	指定目标中未使用消息允许的内存的最大容量（以字节为单位）。
队列和主题	limitBehavior	REJECT_NEWEST	指定在达到内存限制阈值时代理响应的方式。允许的值： FLOW_CONTROL — 减慢生成方 REMOVE_OLDEST — 丢弃最旧的消息 REMOVE_LOW_PRIORITY — 根据消息存在的时间丢弃优先级最低的消息（生成消息的客户机不会收到消息删除的通知） REJECT_NEWEST — 拒绝最新的消息（生成消息的客户机不会获得拒绝持久性消息的异常，但是不会收到拒绝非持久性消息的通知）

1. 在群集环境中，此属性将应用于群集中每个目标实例中，而不是应用到群集中的所有实例。

表 6-10 目标属性 (续)

目标类型	属性	默认值	说明
队列和主题	maxBytesPerMsg	-1 (无限制)	指定目标中允许的任何单个消息的最大大小 (以字节为单位) (生成消息的客户机会获得拒绝持久性消息的异常, 但是不会收到拒绝非持久性消息的通知)。
队列和主题	maxNumProducers ¹	-1 (无限制)	指定目标允许的生成方的最大数量。达到此限制范围时, 将无法创建新的生成方。
仅队列	maxNumActiveConsumers	1	指定从队列目标的负荷平衡传送中可以处于活动状态的最大使用方数。值为 -1 表示不限制数量。(平台版将此值限定为 2。)
仅队列	maxNumBackupConsumers	0	在出现从队列目标的负荷平衡传送错误时, 指定代替这些活动使用方的最大备份使用方数量。值为 -1 表示不限制数量。
队列和主题	consumerFlowLimit	主题: 1000 队列: 1000	指定在一批中将要传送给使用方的消息的最大数量。在负荷平衡队列传送中, 负荷平衡开始之前路由至活动使用方的队列消息的初始数量 (请参见第 69 页的“多个使用方的队列传送”)。此限制可被目标使用方在其各自的连接中指定的较低值覆盖 (请参见《 <i>Message Queue Java Client Developer's Guide</i> 》中的“连接工厂”属性的有关信息)。值为 -1 表示不限制数量。
仅队列	localDeliveryPreferred	false	仅应用于代理群集中的负荷平衡队列传送。指定仅当本地代理中没有使用方时才将消息传送到远程使用方。要求目标不限于仅本地传送 (isLocalOnly = false)。
队列和主题	isLocalOnly	false	仅适用于代理群集。指定目标不能在其他代理上复制, 因而将消息限制为仅向本地使用方传送 (连接到创建目标的代理的使用方)。创建目标后, 此属性无法更新。

1. 在群集环境中, 此属性将应用于群集中每个目标实例中, 而不是应用到群集中的所有实例。

- 要创建队列目标, 请输入如下命令:

```
imqcmd create dst -n myQueue -t q -o "maxNumActiveConsumers=5"
```

请注意, 目标名称必须只包含字母数字字符 (不包括空格), 可以以字母字符或 "_" 和 "\$" 字符开头。但不能以字符串 "mq." 开头, 该字符串为度量依据主题目标保留 (请参见第 65 页的表 2-8)。
- 要创建主题目标, 请输入如下命令:

```
imqcmd create dst -n myTopic -t t -o "maxBytesPerMsg=5000"
```

列出目标

可获得有关目标当前属性值、目标相关的生成方和使用方的数量、以及消息传送度量依据（如目标中消息的数量和大小）的信息。

要查找需要获得信息的目标，可先使用 `list dst` 子命令列出特定代理的所有目标。例如，要获得在 `myHost 4545` 端口上运行的代理中的所有目标列表，请输入以下命令：

```
imqcmd list dst -b myHost:4545
```

`list dst` 子命令可选择性地指定要列出的目标类型或选择性地包含临时目标（使用 `-tmp` 选项）。临时目标由客户机创建，通常用于接收发送到其他客户机的信息回复（请参见第 72 页的“临时目标”）。

显示目标信息

要获得有关目标当前属性值的信息，请使用 `query dst` 子命令，如以下命令：

```
imqcmd query dst -t q -n XQueue -u admin -p admin
```

此命令语句的输出如下：

```
-----
Destination Name      Destination Type
-----
XQueue                Queue

On the broker specified by:

-----
Host                  Primary Port
-----
localhost             7676

Destination Name      XQueue
Destination Type      Queue
Destination State     RUNNING
Created Administratively true

Current Number of Messages      0
Current Total Message Bytes     0
Current Number of Producers     0
Current Number of Active Consumers 0
Current Number of Backup Consumers 0

Max Number of Messages      unlimited (-1)
Max Total Message Bytes     unlimited (-1)
```

Max Bytes per Message	unlimited (-1)
Max Number of Producers	100
Max Number of Active Consumers	1
Max Number of Backup Consumers	0
Limit Behavior	REJECT_NEWEST
Consumer Flow Limit	100
Is Local Destination	false
Local Delivery is Preferred	false

输出同时还显示与目标相关联的生成方和使用方的数量。对于队列目标，将同时包括活动使用方和备份使用方。

可以使用 `update dst` 子命令更改一个或多个属性的值（请参见第 157 页的“更新目标属性”）。

更新目标属性

可以使用 `update dst` 子命令更改目标的属性，而使用 `-o` 选项指定要更新的属性。如果要更新多个属性，可以多次使用 `-o` 选项。例如，以下命令将 `maxBytesPerMsg` 属性更改为 1000，同时将 `MaxNumMsgs` 属性更改为 2000：

```
imqcmd update dst -t q -n myQueue -o "maxBytesPerMsg=1000"
-o "maxNumMsgs=2000" -u admin -p admin
```

要获得可以更新的属性的列表，请参见第 154 页的表 6-10。

不能使用 `update dst` 子命令更新目标的 *type* 属性或更新 `isLocalOnly` 属性。

显示目标度量依据

要获得有关目标的消息度量依据信息，请使用 `metrics dst` 子命令，如以下命令：

```
imqcmd metrics dst -t q -n XQueue -m ttl -u admin -p admin
```

此命令语句的输出如下：

Msgs		Msg Bytes		Msg Count			Total Msg Bytes (k)			Largest
In	Out	In	Out	Current	Peak	Avg	Current	Peak	Avg	Msg (k)
200	200	147200	147200	0	200	0	0	143	71	0
300	200	220800	147200	100	200	10	71	143	64	0
300	300	220800	220800	0	200	0	0	143	59	0

有关使用 `imqcmd` 报告目标度量依据的详细说明，请参见第 220 页的“监视工具”。

暂停和恢复目标

可暂停目标以控制从生成方到目标的消息传送，或从目标到使用方的消息传送，或者二者同时控制。特别是，可暂停到目标的消息流，有助于防止在消息生成明显快于使用时，目标所具有的过多消息。

要暂停流进流出目标的消息传送，请使用 `pause dst` 子命令，如以下命令所示：

```
imqcmd pause dst -n myQueue -t q -pst PRODUCERS -u admin -p admin
imqcmd pause dst -n myTopic -t t -pst CONSUMERS -u admin -p admin
```

如果已经暂停目标并希望恢复传送，请输入以下命令：

```
imqcmd resume dst -n myQueue -t q
```

在多代理群集中，目标实例驻留在群集的每个代理中。必须分别暂停每个目标。

清除目标

可以清除某个目标上当前排队的所有消息。清除目标意味着物理目标上排队的所有消息都将被删除。当目标中堆积的消息占用了大量系统资源时，可能需要清除这些消息。这可能发生在某个队列没有注册使用方客户机，但仍在接收大量消息的情况下。也可能发生在某个主题的长期订阅者始终处于非活动状态的情况下。在上述两种情况下，都没有必要保留消息。

要清除目标中的消息，请使用 `purge dst` 子命令，如以下命令所示：

```
imqcmd purge dst -n myQueue -t q -u admin -p admin
imqcmd purge dst -n myTopic -t t -u admin -p admin
```

如果关闭代理后不希望重新在启动代理时传送旧消息，请使用 `-reset messages` 选项清除旧消息，例如：

```
imqbrokerd -reset messages -u admin -p admin
```

这样可以避免重新启动代理后清除目标的麻烦。

在多代理群集中，目标实例驻留在群集的每个代理中。必须分别清除每个目标。

销毁目标

要销毁目标，请使用 `destroy dst` 子命令，如以下命令所示：

```
imqcmd destroy dst -t q -n myQueue -u admin -p admin
```

销毁目标将清除该目标中的所有消息并将该目标从代理上删除，此操作是不可恢复的。

压缩目标

如果使用内置的、基于文件的数据存储（相对于插入的 **JDBC 兼容数据存储**）作为消息的持久性存储库，可以监视磁盘的利用率并在需要时压缩磁盘。

可构建基于文件的消息存储，以便根据消息所在的目标将其保存在目录中。在每个目标的目录中，大多数消息存储在一个文件中，该文件由大小可变的记录组成，即大小可变的记录文件。（为减少文件碎片，大小超过可配置的阈值的消息将存储在其自己的单独文件中。）由于各种大小的消息可保持持久，并随后从大小可变的记录文件中删除，因而可能会在文件中出现漏洞，即文件中的空闲记录无法重新使用。

要管理未使用的空闲记录，命令行实用程序可包含用于监视每个目标的磁盘利用的子命令，以及在磁盘利用率降低时回收空闲磁盘空间的子命令。

监视目标的磁盘利用

要监视目标的磁盘利用，请使用以下 `imqcmd` 子命令：

```
imqcmd metrics dst -t q -n myQueue -m dsk -u admin -p admin
```

此命令语句的输出如下：

Reserved	Used	Utilization Ratio
806400	804096	99
1793024	1793024	100
2544640	2518272	98

子命令输出中的各列具有以下含义：

表 6-11 目标磁盘利用度量依据	
度量依据	说明
保留的	所有记录使用的磁盘空间（以字节为单位），其中包括保存活动消息的记录以及等待重新使用的空闲记录
已用的	保存活动消息的记录使用的磁盘空间（以字节为单位）
利用率	保留的磁盘空间除已用的磁盘空间所得的系数。比率越高，用于保存活动消息的磁盘空间就越多。

回收未使用的目标磁盘空间

磁盘利用模式取决于使用特定目标的消息发送应用程序的特征。根据流入和流出目标的消息总数的相对值，以及流入和流出目标的消息的相对大小，保留的磁盘空间可能会随时间而增加。

如果消息生成率大于消息使用率，通常应该重新使用空闲记录，且利用率应偏高。但是，如果消息生成率等于或小于消息使用率，则利用率将较低。

总之，请尽量使保留的磁盘空间稳定并使磁盘利用率保持较高水平。凭经验而论，如果系统达到稳定状态，其中保留的磁盘空间量较为稳定，磁盘利用率较高（大于 75%），则不必回收未使用的磁盘空间。如果系统达到稳定状态，而利用率较低（低于 50%），可压缩磁盘以回收空闲记录占用的磁盘空间。

如果保留的磁盘空间随时间持续增加，则应通过设置目标内存限制属性和限制行为来重新配置目标的内存管理（请参见第 154 页的表 6-10）。

► 回收未使用的目标磁盘空间

1. 暂停目标。

```
imqcmd pause dst -t q -n myQueue -u admin -p admin
```

2. 压缩磁盘。

```
imqcmd compact dst -t q -n myQueue -u admin -p admin
```

3. 恢复目标。

```
imqcmd resume dst -t q -n myQueue -u admin -p admin
```

如果未指定目标类型和名称，则会为所有目标执行这些操作。

管理长期订阅

要管理代理的长期订阅，需要使用 imqcmd 子命令。长期订阅是指某台客户机长期注册订阅某个主题；长期订阅有唯一标识，要求代理保留该订阅的消息，即使消息使用方处于非活动状态。通常，代理只能在消息已过期的情况下删除为长期订阅者保留的消息。

表 6-12 概述了 imqcmd 长期订阅子命令。如果不使用默认代理 (localhost:7676)，则必须指定代理的主机名和端口。

表 6-12 用于管理长期订阅的 imqcmd 子命令

子命令	说明
list dur -d destName	列出指定目标的所有长期订阅。
destroy dur -n subscrName -c client_id	销毁指定客户机 ID 的指定长期订阅（请参见第 43 页的“客户机标识符”）。
purge dur -n subscrName -c client_id	清除指定客户机 ID 的指定长期订阅的所有消息（请参见第 43 页的“客户机标识符”）。

例如，以下命令将列出 SPQuotes 主题下的所有长期订阅

```
imqcmd list dur -d SPQuotes
```

对于某个主题下的每个长期订阅，list dur 子命令将返回长期订阅的名称、用户的客户机 ID、该主题下排队消息的数量以及长期订阅的状态（活动 / 非活动）。例如：

Name	Client ID	Number of Messages	Durable Sub State
-----	-----	-----	-----
myDurable	myClientID	1	INACTIVE

可以使用 list dur 子命令返回的信息标识希望销毁或清除其消息的长期订阅。使用订阅名称和客户机 ID 可以标识订阅。例如：

```
imqcmd destroy dur -n myDurable -c myClientID
```

管理事务

客户机应用程序启动的所有事务都由代理进行跟踪。这些事务可以是 XA 资源管理器管理的简单 Message Queue 事务或分布式事务（请参见第 44 页的“本地事务”）。每个事务都有一个 Message Queue 事务 ID，这是一个 64 位数字，唯一标识代理上的事务。分布式事务也有一个分布式事务 ID (XID)，长度为 128 字节，由分布式事务管理器指定。Message Queue 负责维护 Message Queue 事务 ID 与 XID 之间的关联。

对于分布式事务来说，失败的事务可能会保持 PREPARED 状态，而不会提交。因此，作为管理员需要监视并回滚或提交那些处于 PREPARED 状态的事务。

表 6-13 概述了 imqcmd 事务子命令。如果不使用默认代理 (localhost:7676)，则必须指定代理的主机名和端口。

表 6-13 用于管理事务的 imqcmd 子命令

子命令	说明
list txn	列出代理跟踪的所有事务。
query txn -n <i>transaction_id</i>	列出指定事务的相关信息。
commit txn -n <i>transaction_id</i>	提交指定的事务。

表 6-13 用于管理事务的 imqcmd 子命令（续）

子命令	说明
rollback txn -n <i>transaction_id</i>	回滚指定的事务。

例如，以下命令将列出某个代理中的所有事务。

```
imqcmd list txn
```

对于每个事务，list 子命令将返回事务 ID、状态、用户名、消息或确认数量以及创建时间。例如：

Transaction ID	State	User name	# Msgs/ # Acks	Creation time
64248349708800	PREPARED	guest	4/0	1/30/02 10:08:31 AM
64248371287808	PREPARED	guest	0/4	1/30/02 10:09:55 AM

该命令显示了代理中的所有事务，包括本地事务和分布式事务。只能提交或回滚处于 PREPARED 状态的事务。只有当您知道该事务由于失败而处于 PREPARED 状态，而且分布式事务管理器当前没有提交该事务时才可以这样做。

例如，如果代理的 auto-rollback 属性被设置为 false（请参见第 56 页的表 2-4），那么必须在启动代理时手动提交或回滚处于 PREPARED 状态的事务。

list 子命令还显示事务中生成和确认的消息数量（#Msgs/#Acks）。提交事务之前不会传送消息，也不会处理确认。

query 子命令可以显示相同的信息以及许多其他值：客户机 ID、连接标识和分布式事务 ID (XID)。例如，

```
imqcmd query txn -n 64248349708800
```

语句的输出如下：

Client ID	
Connection	guest@192.18.116.219:62209->jms:62195
Creation time	1/30/02 10:08:31 AM
Number of acknowledgements	0
Number of messages	4

State	PREPARED
Transaction ID	64248349708800
User name	guest
XID	6469706F6C7369646577696E6465723130313234313431313030373230

commit 和 rollback 子命令可用于提交和回滚分布式事务。正如前文所述，只能提交或回滚处于 PREPARED 状态的事务。例如：

```
imqcmd commit txn -n 64248349708800
```

也可以配置代理，使它在启动时自动回滚处于 PREPARED 状态的事务。有关详细信息，请参见[第 56 页的表 2-4](#)中的 imq.transaction.autorollback 属性。

管理受管理对象

使用受管理对象可以开发供其他 JMS 提供者使用的客户机应用程序。受管理对象是指包含提供者特定的配置信息和命名信息的对象。这些对象通常由 Message Queue 管理员创建，由客户机应用程序用来与代理建立连接，然后用于向物理目标发送消息并从物理目标接收消息。

有关受管理对象的概述信息，请参见第 78 页的“[Message Queue 受管理对象](#)”。

Message Queue 为创建和管理受管理对象提供了两个管理工具：命令行对象管理器实用程序 (imqobjmgr) 和 GUI 管理控制台。您可以使用这两个工具执行以下操作：

- 在对象存储库中添加或删除受管理对象。
- 列出现有的受管理对象。
- 查询和显示受管理对象的相关信息。
- 修改对象存储库中现有的受管理对象。

本章说明如何使用对象管理器实用程序 (imqobjmgr) 执行上述任务。由于这些任务涉及到了解您正在使用的对象存储库以及您正在创建的受管理对象的属性，本章在说明如何使用 imqobjmgr 管理受管理对象之前介绍了这两个主题的背景。

有关使用管理控制台的信息，请参见第 4 章“[管理控制台教程](#)”。。

关于对象存储库

受管理对象放置在便于访问的对象存储库中，客户机应用程序可以通过 JNDI 查找访问这些对象。可以使用两种类型的对象存储库：标准 LDAP 目录服务器和文件系统对象存储库。

LDAP 服务器对象存储库

对于生产型消息传送系统，建议使用 LDAP 服务器对象存储库。很多供应商都提供 LDAP 实现，而且 LDAP 实现可以在分布式系统中使用。LDAP 服务器还提供对生产环境特别有用的安全功能。

Message Queue 管理工具可以管理 LDAP 服务器上的对象存储库。不过，您可能需要按照 LDAP 服务器文档中的规定先配置 LDAP 服务器以存储 Java 对象并执行 JNDI 查找。

将 LDAP 服务器用作对象存储库时，需要指定表 7-1 中显示的属性。这些属性分为以下几类：

- **初始上下文：**对于 LDAP 服务器对象存储库，该属性是固定的。
- **位置：**按照 LDAP 服务器中的设置指定用于存储受管理对象的 URL 和目录路径。特别是，必须检查指定的路径是否存在。
- **安全信息：**取决于 LDAP 提供者。应参阅 LDAP 实现附带的文档，以决定是所有的操作都需要安全信息，还是只有更改存储数据的操作需要安全信息。

表 7-1 LDAP 对象存储库属性

属性	说明
java.naming.factory.initial	LDAP 服务器上 JNDI 查找的初始上下文 com.sun.jndi.ldap.LdapCtxFactory
java.naming.provider.url	LDAP 服务器 URL 和目录路径信息。例如： ldap://mydomain.com:389/ou=mqobjs, o=myapp 其中受管理对象存储在 /myapp/mqobjs 目录中
java.naming.security.principal	负责验证 LDAP 服务器呼叫者的人员的身份。此条目的格式取决于验证方案。例如： uid=fooUser, ou=People, o=mq 如果不指定此属性，性能将由 LDAP 服务提供者决定。

表 7-1 LDAP 对象存储库属性 (续)

属性	说明
java.naming.security.credentials	负责验证 LDAP 服务器呼叫者的人员的凭证。此属性的值取决于验证方案：它可能是散列密码、纯文本密码、密钥、证书等。例如： fooPasswd 如果不指定此属性，性能将由 LDAP 服务提供者决定。
java.naming.security.authentication	要使用的安全级别。其值是以下关键字之一：none、simple 或 strong。 例如，如果指定 simple，缺少任何 principal 或 credential 值时，系统都会提示您。这使您可以一种更安全的方式提供身份信息。 如果不指定此属性，性能将由 LDAP 服务提供者决定。

文件系统对象存储库

Message Queue 还支持文件系统对象存储库实现。文件系统对象存储库尚未进行完整测试，因此不建议用于生产系统。它的优势就是在开发环境中非常易于使用。与其设置一台 LDAP 服务器，还不如在本地文件系统中创建一个目录。

不过，文件系统存储不能用作在多个计算机节点上配置的客户机的集中式对象存储库，除非这些客户机可以访问该对象存储库所在的目录。此外，任何可以访问这个目录的用户都可以使用 Message Queue 管理工具创建和管理受管理对象。

使用文件系统对象存储库时，需要指定表 7-2 中显示的属性。这些属性分为以下几类：

- **初始上下文：**对于文件系统对象存储库，该属性的值是固定的。
- **位置：**该属性的值指定用于存储受管理对象的目录路径。目录必须存在且必须拥有 Message Queue 管理工具用户以及将访问该存储的客户机应用程序用户的相应访问权限。

表 7-2 文件系统对象存储库属性

属性	说明
java.naming.factory.initial	文件系统对象存储库上 JNDI 查找的初始上下文： com.sun.jndi.fscontext.RefFSContextFactory
java.naming.provider.url	目录路径信息。例如： file:///C:/myapp/mqobjs

受管理对象

有关受管理对象的概述信息，请参见第 78 页的“[Message Queue 受管理对象](#)”。

Message Queue 受管理对象有两种基本类型：连接工厂和目标。*连接工厂*受管理对象由客户机应用程序用来建立与代理之间的连接。*目标*受管理对象由客户机应用程序用来标识生成方向其发送消息或使用方从其检索消息的目标。（SOAP 消息传送使用特殊的 SOAP 端点受管理对象，有关详细信息，请参阅《*Message Queue Java Client Developer's Guide*》。）

可以根据消息传送模型（点对点或者发布 / 订阅）使用特定类型的连接工厂和目标。例如，在点对点编程中，可以使用队列连接工厂和队列目标。同样，在发布和订阅编程中，可以使用主题连接工厂和主题目标。也可以使用非特定的连接工厂和目标受管理对象类型，只要它们属于支持分布式事务的连接工厂类型（要查看所有支持的类型，请参见第 42 页的表 1-1）。

受管理对象的属性是使用属性 - 值对指定的。下面将介绍这些属性。

连接工厂受管理对象属性

表 7-3 列出了连接工厂（和 XA 连接工厂）受管理对象的属性。最重要的属性是 `imqAddressList`，它用于指定客户机将与之建立连接的代理。第 176 页的“[添加连接工厂](#)”一节说明了如何在向对象存储库中添加连接工厂受管理对象时指定属性。

有关连接工厂属性的说明以及有关如何使用它们的信息，请参见《*Message Queue Java Client Developer's Guide*》和适用于下列 Message Queue 类的 JavaDoc API 文档：`com.sun.messaging.ConnectionConfiguration`。

表 7-3 连接工厂受管理对象属性

属性 / 属性名	类型	默认值
<code>imqAckOnAcknowledge</code>	字符串	没有值
<code>imqAckOnProduce</code>	字符串	没有值
<code>imqAckTimeout</code>	字符串	0 毫秒
<code>imqAddressList</code>	字符串	没有值
<code>imqAddressListIterations</code>	整数	1
<code>imqAddressListBehavior</code>	字符串	PRIORITY
<code>imqBrokerHostName (Message Queue 3.0)</code>	字符串	localhost
<code>imqBrokerHostPort (Message Queue 3.0)</code>	整数	7676

表 7-3 连接工厂受管理对象属性 (续)

属性 / 属性名	类型	默认值
imqBrokerServicePort (Message Queue 3.0)	整数	0
imqConfiguredClientID	字符串	没有值
imqConnectionFlowCount	整数	100
imqConnectionFlowLimit	整数	1000
imqConnectionFlowLimitEnabled	布尔值	false
imqConnectionType (Message Queue 3.0)	字符串	TCP
imqConnectionURL (Message Queue 3.0)	字符串	http://localhost/imq/ tunnel
imqConsumerFlowLimit	整数	1000
imqConsumerFlowThreshold	整数	50
imqDefaultPassword	字符串	guest
imqDefaultUsername	字符串	guest
imqDisableSetClientID	布尔值	false
imqJMSDeliveryMode	整数	2 (持久性)
imqJMSExpiration	Long	0 (不过期)
imqJMSPriority	整数	4 (一般)
imqLoadMaxToServerSession	布尔值	true
imqOverrideJMSDeliveryMode	布尔值	false
imqOverrideJMSExpiration	布尔值	false
imqOverrideJMSHeadersTo TemporaryDestinations	布尔值	false
imqOverrideJMSPriority	布尔值	false
imqQueueBrowserMaxMessages PerRetrieve	整数	1000
imqQueueBrowserRetrieveTimeout	Long	60,000 (毫秒)
imqReconnectAttempts	整数	0
imqReconnectEnabled	布尔值	false
imqReconnectInterval	Long	3000 (毫秒)
imqSetJMSXAppID	布尔值	false

表 7-3 连接工厂受管理对象属性 (续)

属性 / 属性名	类型	默认值
imqSetJMSXConsumerTXID	布尔值	false
imqSetJMSXProducerTXID	布尔值	false
imqSetJMSXRcvTimestamp	布尔值	false
imqSetJMSXUserID	布尔值	false
imqSSLIsHostTrusted (Message Queue 3.0)	布尔值	true

目标受管理对象的属性

表 7-4 列出了标识物理主题或队列目标的目标受管理对象的属性。第 177 页的“添加主题或队列”一节说明了如何在向对象存储库中添加目标受管理对象时指定这些属性。

最重要的属性是 `imqDestinationName`。这是您为与主题或队列受管理对象相对应的物理目标指定的名称。您也可以为目标提供说明，帮助您将它与您创建以支持更多应用程序的其他目标区分开。

有关详细信息，请参见适用于 `Message Queue` 类 `com.sun.messaging.DestinationConfiguration` 的 JavaDoc API 文档。

表 7-4 目标受管理对象的属性

属性 / 属性名	类型	默认值
imqDestinationDescription	字符串	目标对象的说明
imqDestinationName	字符串 ¹	Untitled_Destination_Object

1. 目标名称只能包含字母数字字符（不能包含空格），且必须以字母字符或者字符“_”和 / 或 “\$” 开头。

对象管理器实用程序 (imqobjmgr)

对象管理器实用程序允许您创建和管理 Message Queue 受管理对象。本节介绍了基本的 imqobjmgr 命令语法，提供了一个子命令列表，并概述了 imqobjmgr 命令选项。下文说明如何使用 imqobjmgr 子命令完成特定任务。

imqobjmgr 命令的语法

imqobjmgr 命令的一般语法如下：

```
imqobjmgr subcommand [options]
imqobjmgr -h|H
imqobjmgr -v
```

请注意，如果指定 -v、-h 或 -H 选项，将不会执行命令行中指定的子命令。例如，输入以下命令将显示版本信息，而不执行 list 子命令。

```
imqobjmgr list -v
```

imqobjmgr 子命令

表 7-5 列出了对象管理器实用程序 (imqobjmgr) 包含的子命令：

表 7-5 imqobjmgr 子命令	
子命令	说明
add	在对象存储库中添加受管理对象。
delete	从对象存储库中删除受管理对象。
list	列出对象存储库中的受管理对象。
query	显示指定的受管理对象的相关信息。
update	修改对象存储库中的现有受管理对象。

imqobjmgr 命令选项概述

表 7-6 列出了 imqobjmgr 命令的选项。有关如何使用这些选项的论述，请参见基于任务的后续各节。

表 7-6 imqobjmgr 选项

选项	说明
-f	执行操作，而无需用户确认。
-h	显示使用帮助。不在命令行上执行其他命令。
-H	显示使用帮助、属性列表和示例。不在命令行上执行其他命令。
-i <i>fileName</i>	指定命令文件的名称，该文件包含所有或部分子命令子句，指定了对象类型、查找名称、对象属性、对象存储库属性或其他选项。通常用于重复性信息，例如对象存储库属性。
-j <i>attribute=value</i>	指定标识和访问 JNDI 对象存储库所需的属性。请参见第 166 页的“LDAP 服务器对象存储库”和第 167 页的“文件系统对象存储库”。
-j <i>javahome path</i>	指定要使用的替代 Java 2 兼容运行时（默认使用系统上的运行时或 Message Queue 附带的运行时）。
-l <i>lookupName</i>	指定受管理对象的 JNDI 查找名称。此名称在对象存储库的上下文中必须唯一。
-o <i>attribute=value</i>	指定受管理对象的属性。请参见第 168 页的“连接工厂受管理对象属性”和第 170 页的“目标受管理对象的属性”。
-pre	预览模式。指出在不执行命令的情况下将执行的操作。
-r <i>read-only_state</i>	指定某个受管理对象是否为只读对象。true 值表示受管理对象为只读对象。客户不能修改只读受管理对象的属性。默认情况下，read-only state 设置为 false。
-s	无提示模式。不显示任何输出。
-t <i>objectType</i>	指定 Message Queue 受管理对象的类型： q = 队列 t = 主题 cf = 连接工厂 qf = 队列连接工厂 tf = 主题连接工厂 xcf = XA 连接工厂（分布式事务） xqf = XA 队列连接工厂（分布式事务） xtf = XA 主题连接工厂（分布式事务） e = SOAP 端点 ¹
-v	显示版本信息。不在命令行上执行其他命令。

1. 此受管理对象类型用于支持 SOAP 消息（请参见《Message Queue Java Client Developer's Guide》）。

下一节说明使用 `imgobjmgr` 子命令时需要提供的信息。

所需的信息

执行大多数与受管理对象相关的任务时，必须指定以下信息作为 `imgobjmgr` 子命令的选项：

- **受管理对象的类型：**

[表 7-6](#) 显示了允许的类型。

- **受管理对象的 JNDI 查找名称：**

这是在客户机代码中表示对象存储库中的受管理对象（使用 JNDI）时使用的逻辑名。

- **受管理对象的属性**（`add` 和 `update` 子命令特别需要）：

- 对于目标：代理上的物理目标名称。这是使用 `imgcmd create dst` 子命令的 `-n` 选项指定的名称。如果不指定名称，将使用默认名称 `Untitled_Destination_Object`。
- 对于连接工厂：最常用的属性是地址列表 (`imgAddressList`)，用于指定客户机将试图连接的消息服务器地址（一个或多个）。如果不指定此信息，则使用本地主机和默认端口号 (7676)，这意味着客户机将试图与本地主机端口 7676 上的某个代理建立连接。[第 176 页的“添加连接工厂”](#)一节说明了如何指定对象属性。

有关其他属性的详细信息，请参见 [第 168 页的“连接工厂受管理对象属性”](#)。

- **对象存储库属性：**

此信息取决于您使用的是文件系统存储还是 LDAP 服务器，但必须包括以下属性：

- JNDI 实现的类型（初始上下文属性）。例如，文件系统或 LDAP。
- 受管理对象在对象存储库中的位置（提供者 URL 属性），即它所在的“文件夹”。
- 访问对象存储库所需的用户名、密码和授权类型（如果有）。

有关对象存储库属性的详细信息，请参见 [第 166 页的“LDAP 服务器对象存储库”](#)和 [第 167 页的“文件系统对象存储库”](#)。

使用命令文件

imgobjmgr 命令允许您指定命令文件的名称，该文件使用 Java 属性文件语法表示全部或部分 imgobjmgr 子命令子句。

在对象管理器实用程序 (imgobjmgr) 中使用命令文件对于指定对象存储库属性特别有用，因为对象存储库属性可能在 imgobjmgr 的多次调用中都相同，并且通常需要多次键入。使用命令文件还可以避免一种情况，即避免超出命令行所允许的最大字符数。

imgobjmgr 命令文件的一般语法如下（版本属性反映命令文件的版本，而非 Message Queue 产品的版本，它不是一个命令行选项，它的值必须设置为 2.0）：

```
version=2.0
cmdtype=[ add | delete | list | query | update ]
obj.type=[ q | t | qf | tf | cf | xqf | xtf | xcf | e ]
obj.lookupName=lookup name
obj.attrs.objAttrName1=value1
obj.attrs.objAttrName2=value2
obj.attrs.objAttrNameN=valueN
...
objstore.attrs.objStoreAttrName1=value1
objstore.attrs.objStoreAttrName2=value2
objstore.attrs.objStoreAttrNameN=valueN
...
```

作为如何使用命令文件的一个示例，请考虑使用以下 imgobjmgr 命令：

```
imgobjmgr add
-t qf
-l "cn=myQCF"
-o "imgAddressList=mq://foo:777/jms"
-j "java.naming.factory.initial=
    com.sun.jndi.ldap.LdapCtxFactory"
-j "java.naming.provider.url=
    ldap://mydomain.com:389/o=img"
-j "java.naming.security.principal=
    uid=fooUser, ou=People, o=img"
-j "java.naming.security.credentials=fooPasswd"
-j "java.naming.security.authentication=simple"
```

此命令可以包含在一个文件中（例如 MyCmdFile 文件），其内容如下：

```

version=2.0
cmdtype=add
obj.type=qf
obj.lookupName=cn=myQCF
obj.attrs.imqAddressList=mq://foo:777/jms
objstore.attrs.java.naming.factory.initial=\
    com.sun.jndi.ldap.LdapCtxFactory
objstore.attrs.java.naming.provider.url=\
    ldap://mydomain.com:389/o=img
objstore.attrs.java.naming.security.principal=\
    uid=fooUser, ou=People, o=img
objstore.attrs.java.naming.security.credentials=fooPasswd
objstore.attrs.java.naming.security.authentication=simple

```

然后可以使用 `-i` 选项将该文件传递给对象管理器实用程序 (imqobjmgr):

```
imqobjmgr -i MyCmdFile
```

也可以使用命令文件指定一些选项，而使用命令行指定其他选项。这样即可使用命令文件指定在实用程序的多次调用中都相同的那部分子命令子句。例如，以下命令可以指定添加连接工厂受管理对象所需的所有选项，但不包括指定受管理对象存储位置的选项。

```

imqobjmgr add
-t qf
-l "cn=myQCF"
-o "imqAddressList=mq://foo:777/jms"
-i MyCmdFile

```

在本例中，文件 `MyCmdFile` 将包含以下定义：

```

version=2.0
objstore.attrs.java.naming.factory.initial=\
    com.sun.jndi.ldap.LdapCtxFactory
objstore.attrs.java.naming.provider.url=\
    ldap://mydomain.com:389/o=img
objstore.attrs.java.naming.security.principal=\
    uid=fooUser, ou=People, o=img
objstore.attrs.java.naming.security.credentials=fooPasswd
objstore.attrs.java.naming.security.authentication=simple

```

可以从以下位置找到命令文件的其他示例：

IMQ_HOME/demo/imgobjmgr

添加和删除受管理对象

本节说明如何在对象存储库中添加连接工厂和主题或队列目标的受管理对象。

注意 对象管理器实用程序 (imgobjmgr) 只列出和显示了 Message Queue 受管理对象。如果对象存储库中包含与希望添加的受管理对象具有相同查找名称的非 Message Queue 对象，则您在尝试执行添加操作时将收到一条错误消息。

添加连接工厂

要使客户机应用程序获得与代理的连接，需要添加一个用来表示客户机应用程序所需连接类型的受管理对象：主题连接工厂或队列连接工厂

要添加队列连接工厂，请使用如下所示的命令：

```
imgobjmgr add
-t qf
-l "cn=myQCF"
-o "imgAddressList=mq://myHost:7272/jms"
-j "java.naming.factory.initial=
    com.sun.jndi.ldap.LdapCtxFactory"
-j "java.naming.provider.url=ldap://mydomain.com:389/o=imq"
-j "java.naming.security.principal=
    uid=fooUser, ou=People, o=imq"
-j "java.naming.security.credentials=fooPasswd"
-j "java.naming.security.authentication=simple"
```

上述命令创建一个查找名称为 cn=myQCF 的受管理对象，它连接到 myHost 上运行的代理并在端口 7272 上监听。受管理对象存储在 LDAP 服务器上。通过指定一个命令文件作为 imgobjmgr 命令的参数，也可以实现此目的。有关详细信息，请参见第 174 页的“使用命令文件”。

注意

命名惯例：如果使用 LDAP 服务器存储受管理对象，必须像上例那样指定一个前缀为 "cn=" 的查找名称 (cn=myQCF)。可以通过 -l 选项指定查找名称。如果正在使用文件系统对象存储库，则不必使用 cn 前缀，不过，请不要使用带有 "/" 的查找名称。请参见表 7-7。

表 7-7 命名惯例示例

对象存储库类型	好名称	禁用名称
LDAP 服务器	cn=myQCF	myQCF
文件系统	myTopic	myObjects/myTopic

添加主题或队列

要使客户机应用程序能够访问代理上的物理目标，可以将标识这些目标的受管理对象添加到对象存储库中。

最好先创建物理目标，然后再将对应的受管理对象添加到对象存储库中。使用命令行实用程序 (imgcmd) 在代理上创建物理目标，这些目标由对象存储库中的目标受管理对象标识。有关创建物理目标的信息，请参见第 151 页的“获得连接信息”。

以下命令添加了一个标识主题目标的受管理对象，这个主题目标的查找名称为 myTopic，物理目标名称为 TestTopic。受管理对象存储在 LDAP 服务器上。

```
imgobjmgr add
-t t
-l "cn=myTopic"
-o "imgDestinationName=TestTopic"
-j "java.naming.factory.initial=
    com.sun.jndi.ldap.LdapCtxFactory"
-j "java.naming.provider.url=
    ldap://mydomain.com:389/o=img"
-j "java.naming.security.principal=
    uid=fooUser, ou=People, o=img"
-j "java.naming.security.credentials=fooPasswd"
-j "java.naming.security.authentication=simple"
```

这是同一个命令，只不过受管理对象存储在 Solaris 文件系统中：

```
imgobjmgr add
-t t
-l "cn=myTopic"
-o "imgDestinationName=TestTopic"
-j "java.naming.factory.initial=
    com.sun.jndi.fscontext.RefFSContextFactory"
-j "java.naming.provider.url=
    file:///home/foo/img_admin_objects"
```

以 LDAP 服务器为例，可以使用命令文件 MyCmdFile 指定子命令子句。该文件将包含以下内容：

```
version=2.0
cmdtype=add
obj.type=t
obj.lookupName=cn=myTopic
obj.attrs.imgDestinationName=TestTopic
objstore.attrs.java.naming.factory.initial=
    com.sun.jndi.fscontext.RefFSContextFactory
objstore.attrs.java.naming.provider.url=
    file:///home/foo/img_admin_objects
objstore.attrs.java.naming.security.principal=
    uid=fooUser, ou=People, o=img
objstore.attrs.java.naming.security.credentials=fooPasswd
objstore.attrs.java.naming.security.authentication=simple
```

使用 -i 选项将该文件传递给 imgobjmgr 命令：

```
imgobjmgr -i MyCmdFile
```

注意	如果使用 LDAP 服务器存储受管理对象，必须像上例那样指定一个前缀为 "cn=" 的查找名称。可以通过 -l 选项指定查找名称。如果使用文件系统对象存储库，则无需使用该前缀。
-----------	--

添加队列对象与此基本相同，不同之处在于要为 -t 选项指定 q。

删除受管理对象

可以使用 `delete` 子命令删除受管理对象。必须指定对象的查找名称、类型和地址。

以下命令删除了查找名称为 `cn=myTopic` 且存储在 LDAP 服务器上的主题的受管理对象。

```
imgobjmgr delete
-t t
-l "cn=myTopic"
-j "java.naming.factory.initial=
    com.sun.jndi.ldap.LdapCtxFactory"
-j "java.naming.provider.url=
    ldap://mydomain.com:389/o=img"
-j "java.naming.security.principal=
    uid=fooUser, ou=People, o=img"
-j "java.naming.security.credentials=fooPasswd"
-j "java.naming.security.authentication=simple"
```

获得信息

可以使用 `list` 和 `query` 子命令列出对象存储库中的受管理对象，显示单个对象的相关信息。

受管理对象列表

使用 `list` 子命令可以获得所有受管理对象的列表，或者某个特定类型的所有受管理对象的列表。以下样例代码假定受管理对象存储在 LDAP 服务器上。

以下命令将列出所有对象。

```
imgobjmgr list
-j "java.naming.factory.initial=
    com.sun.jndi.ldap.LdapCtxFactory"
-j "java.naming.provider.url=
    ldap://mydomain.com:389/o=img"
-j "java.naming.security.principal=
    uid=fooUser, ou=People, o=img"
-j "java.naming.security.credentials=fooPasswd"
-j "java.naming.security.authentication=simple"
```

以下命令将列出类型为 `queue` 的所有对象。

```
imgobjmgr list
-t q
-j "java.naming.factory.initial=
    com.sun.jndi.ldap.LdapCtxFactory"
-j "java.naming.provider.url=
    ldap://mydomain.com:389/o=img"
-j "java.naming.security.principal=
    uid=fooUser, ou=People, o=img"
-j "java.naming.security.credentials=fooPasswd"
-j "java.naming.security.authentication=simple"
```

单个对象的相关信息

使用 `query` 子命令可以获得单个受管理对象的相关信息。必须指定对象的查找名称，以及包含受管理对象的对象存储库的属性（如初始上下文和地址）。

在以下示例中，使用 `query` 子命令显示查找名称为 `cn=myTopic` 的对象的相关信息。

```
imgobjmgr query
-l "cn=myTopic"
-j "java.naming.factory.initial=
    com.sun.jndi.ldap.LdapCtxFactory"
-j "java.naming.provider.url=
    ldap://mydomain.com:389/o=img"
-j "java.naming.security.principal=
    uid=fooUser, ou=People, o=img"
-j "java.naming.security.credentials=fooPasswd"
-j "java.naming.security.authentication=simple"
```

更新受管理对象

可以使用 `update` 命令修改受管理对象的属性。必须指定对象的查找名称和地址。可以使用 `-o` 选项修改属性值。

此命令将改变表示主题连接工厂的受管理对象的属性：

```
imgobjmgr update
-t tf
-l "cn=MyTCF"
-o imgReconnectAttempts=3
-j "java.naming.factory.initial=
    com.sun.jndi.ldap.LdapCtxFactory"
-j "java.naming.provider.url=
    ldap://mydomain.com:389/o=img"
-j "java.naming.security.principal=
    uid=fooUser, ou=People, o=img"
-j "java.naming.security.credentials=fooPasswd"
-j "java.naming.security.authentication=simple"
```

更新受管理对象

管理安全性

本章介绍如何执行与安全性相关的任务。这些任务包括验证、授权和加密。

验证用户 您负责维护用户列表、用户组以及用户信息库中的密码。各个代理实例可以使用不同的用户信息库。本章第一部分介绍如何创建、填充和管理信息库。有关 Message Queue 安全性的介绍，请参见第 60 页的“安全性管理器”。

授权用户：访问控制属性文件 您负责编辑访问控制属性文件，此属性文件用于将每个用户对代理操作的访问权限映射到用户名或用户组成员资格。各个代理实例可以使用不同的访问控制属性文件。本章第二部分介绍如何自定义此文件。

加密：使用基于 SSL 的服务（企业版） 使用基于安全套接字层 (SSL) 标准的连接服务可以对客户机与代理之间传送的消息进行加密。有关 Message Queue 如何处理加密的介绍，请参见第 62 页的“加密（企业版）”。本章最后一部分介绍如何设置基于 SSL 的连接服务，并提供了与 SSL 有关的其他信息。

在代理需要提供密码才能安全访问 SSL 密钥存储、LDAP 用户信息库或 JDBC 兼容持久性存储库的情况下，提供密码的方式有三种：

- 让系统在代理启动时提示您输入密码
- 启动代理时通过命令行选项提供密码（请参见第 123 页的“启动代理”和第 125 页的表 5-2）
- 将密码存储在代理启动时可以访问的密码文件中（请参见第 204 页的“使用密码文件”）

验证用户

当某个用户试图与代理建立连接时，代理将通过检查该用户提供的用户名和密码对其进行验证。如果用户提供的用户名和密码与配置每个代理时指定要参照的代理特定的用户信息库中的用户名和密码相匹配，则允许该用户连接到该代理。信息库有两种类型：

- **Message Queue** 出厂时附带的文本文件信息库

这种用户信息库易于使用，但容易受到安全性攻击，因此仅用于测试和开发目的。您可以使用用户管理器实用程序 (`imqusermgr`) 填充和管理信息库。要启用验证，您需要用每个用户名、密码和用户组名称填充用户信息库。

有关设置和管理用户信息库的详细信息，请参见 [“使用文本文件用户信息库”](#)。

- **LDAP 服务器**

这可以是现有的或新的 **LDAP** 目录服务器，这种服务器使用 **LDAP v2** 或 **v3** 协议。它并不象文本文件信息库那样易于使用，但它更安全并可伸缩，因此更适用于生产环境。

如果您目前使用的是 **LDAP** 用户信息库，您需要使用 **LDAP** 供应商提供的工具来填充和管理该用户信息库。有关详细信息，请参见第 190 页的 [“使用 LDAP 服务器管理用户信息库”](#)。

使用文本文件用户信息库

Message Queue 提供了一个文本文件用户信息库，还提供了一个命令行工具 **Message Queue** 用户管理器 (`imqusermgr`)，您可以使用它填充和管理文本文件用户信息库。以下各节介绍文本文件用户信息库以及如何使用 **Message Queue** 用户管理器实用程序 (`imqusermgr`) 来填充和管理该信息库。

创建用户信息库

文本文件用户信息库是特定于实例的。默认的用户信息库（名为 `passwd`）是为启动的每个代理实例创建的。此用户信息库所在的目录由与该信息库相关联的代理实例的名称 (`instanceName`) 标识（请参见附录 A [“Message Queue 数据的位置”](#)）：

```
.../instances/instanceName/etc/passwd
```

信息库创建时附带了两个条目（行），如表 8-1 所示。

表 8-1 用户信息库中的初始条目

用户名	密码	组	状态
admin	admin	admin	处于活动状态
guest	guest	anonymous	处于活动状态

这些初始条目使 Message Queue 代理安装后即可使用，无需任何管理员的介入。换句话说，使用 Message Queue 代理之前无需设置初始用户 / 密码。

初始 guest 用户条目使客户机可以使用默认的 guest 用户名和密码连接到代理实例（例如，用于测试目的）。

初始 admin 用户条目使您可以通过 `imqcmd` 命令，使用默认的 admin 用户名和密码管理代理实例。建议您更新此初始条目以更改密码（请参见第 189 页的“更改默认的管理员密码”）。

以下各节说明如何填充和管理文本文件用户信息库。

用户管理器实用程序 (imqusermgr)

此用户管理器实用程序 (imqusermgr) 允许您编辑或填充文本文件用户信息库。

本节介绍了基本的 imqusermgr 命令语法，提供了一个子命令列表，并概述了 imqusermgr 命令选项。下文说明如何使用 imqusermgr 子命令完成特定任务。

使用 imqusermgr 之前，请谨记以下内容：

- 如果代理特定的用户信息库不存在，您需要启动相应的代理实例来创建此信息库。
- imqusermgr 命令需要运行在已经安装了代理的主机上
- 您需要具有写入信息库的相应权限：即在 Solaris 和 Linux 平台上，您的身份必须是超级用户或首次创建代理实例的用户

imqusermgr 命令的语法

imqusermgr 命令的一般语法如下：

```
imqusermgr subcommand [options]
imqusermgr -h
imqusermgr -v
```

请注意，如果指定 `-v` 或 `-h` 选项，将不会执行命令行中指定的子命令。例如，输入以下命令将显示版本信息，而不是执行 `list` 子命令。

```
imqusermgr list -v
```

imqusermgr 子命令

表 8-2 列出了 imqusermgr 子命令。

表 8-2 imqusermgr 子命令

子命令	说明
add [-i <i>instanceName</i>] -u <i>userName</i> -p <i>passwd</i> [-g <i>group</i>] [-s]	将用户和关联的密码添加到指定的 （或默认的）代理实例信息库中，并有选择地指定用户所属的组。
delete [-i <i>instanceName</i>] -u <i>userName</i> [-s] [-f]	从指定的 （或默认的）代理实例信息库中删除指定用户。
list [-i <i>instanceName</i>] [-u <i>userName</i>]	显示指定的 （或默认的）代理实例信息库中有关指定用户或所有用户的信息。
update [-i <i>instanceName</i>] -u <i>userName</i> -p <i>passwd</i> [-a <i>state</i>] [-s] [-f]	更新指定的 （或默认的）代理实例信息库中指定用户的密码和 / 或状态。
update [-i <i>instanceName</i>] -u <i>userName</i> -a <i>state</i> [-p <i>passwd</i>] [-s] [-f]	

注意 以下各节以默认代理实例作为示例。

imqusermgr 命令选项概述

表 8-3 列出了 imqusermgr 命令的选项。

表 8-3 imqusermgr 选项

选项	说明
-a <i>active_state</i>	指定用户是否处于活动状态 (true/false)。true 表示处于活动状态。这是默认值。
-f	执行操作，无需用户确认。
-h	显示使用帮助。不执行命令行上的其他选项。
-i <i>instanceName</i>	指定此命令要应用到的代理实例用户信息库。如果未指定，则将使用默认的 <i>instanceName</i> imqbroker。
-p <i>passwd</i>	指定用户密码。
-g <i>group</i>	指定用户组。有效值包括 admin、user 和 anonymous。
-s	设置无提示模式。

表 8-3 imqusermgr 选项 (续)

选项	说明
-u <i>userName</i>	指定用户名。
-v	显示版本信息。不执行命令行上的其他选项。

组

向代理实例的用户信息库中添加用户条目时，您可以为该用户指定三个预定义组中的一个组，这三个组为：admin、user 或 anonymous。如果不指定任何组，则默认属于 user 组。

- **admin 组。** 用于代理管理员。默认情况下，指定到该组中的用户可以配置和管理代理。您可以为 admin 组指定多个用户。
- **user 组。** 用于普通（非管理）Message Queue 客户机用户。多数客户机用户将访问 user 组中已经过验证的代理。默认情况下，该组中的应用程序客户机用户可以生成和使用所有主题和队列的消息，也可以浏览任意队列中的消息。
- **anonymous 组。** 用于那些不想使用代理已知用户名（也许客户机应用程序不知道可以使用的实际用户名）的 Message Queue 客户机。这类似于多数 FTP 服务器中的匿名帐户。您一次只能为 anonymous 组指定一个用户。与 user 组相比，您需要通过访问控制限制此组的访问权限，或者在部署时从该组中删除用户。

要更改某个用户所属的组，您必须删除该用户的条目，然后再添加一个用户条目并为其指定一个新组。

您可以指定访问规则，定义组成员可以执行哪些操作。有关详细信息，请参见第 192 页的“授权用户：访问控制属性文件”。

状态

向信息库中添加用户时，用户的默认状态是活动的。要使用户处于非活动状态，您必须使用 update 命令。例如，以下命令将使用户 JoeD 处于非活动状态：

```
imqusermgr update -u JoeD -a false
```

处于非活动状态的用户条目将保留在信息库中，但不能打开新连接。当某个用户处于非活动状态时，如果您试图添加具有相同名称的另一个用户，操作将失败。必须删除处于非活动状态的用户条目，或者更改新用户的名称，或者为新用户指定不同的名称。这样可以防止添加重复的用户名。

用户名和密码的格式

用户名和密码必须遵循以下原则：

- 用户名不能包含表 8-4 中列出的字符。

表 8-4 用户名和密码不能包含的字符

字符	说明
*	星号
,	逗号
:	冒号

- 用户名和密码不能包含新行符或回车符。
- 如果用户名或密码包含空格，必须将整个用户名或密码放在引号中。
- 用户名或密码必须至少包含一个字符。
- 除了命令 `shell` 规定的命令行中最多可输入的字符数之外，密码或用户名的长度没有限制。

填充和管理用户信息库

要在信息库中添加用户，可以使用 `add` 子命令。例如，以下命令将向默认代理实例用户信息库添加用户名为 `Katharine`，密码为 `sesame` 的用户。

```
imqusermgr add -u Katharine -p sesame -g user
```

要从信息库中删除用户，可以使用 `delete` 子命令。例如，以下命令将删除用户 `Bob`：

```
imqusermgr delete -u Bob
```

要更改用户的密码或状态，可以使用 `update` 子命令。例如，以下命令将 `Katharine` 的密码更改为 `alladin`：

```
imqusermgr update -u Katharine -p alladin
```

要列出一个或所有用户的相关信息，可以使用 `list` 命令。以下命令将显示用户名为 `isa` 的相关信息：

```
imqusermgr list -u isa
```

```
% imqusermgr list -u isa

User repository for broker instance:imqbroker
-----
User Name      Group      Active State
-----
isa            admin      true
```

以下命令将列出所有用户的相关信息：

```
imqusermgr list
```

```
% imqusermgr list

User repository for broker instance:imqbroker
-----
User Name      Group      Active State
-----
admin          admin      true
guest          anonymous  true
isa            admin      true
testuser1      user       true
testuser2      user       true
testuser3      user       true
testuser4      user       false
testuser5      user       false
```

更改默认的管理员密码

为安全起见，应该将 admin 的默认密码更改为只有您自己知道的密码。为此，需要使用 imqusermgr 工具。

以下命令将 mybroker 代理实例的默认密码更改为 grandpoobah。

```
imqusermgr update -i mybroker -u admin -p grandpoobah
```

要快速确认此更改是否已生效，可以在代理实例运行时运行任何命令行工具。例如，可以使用以下命令：

```
imqcmd list svc -i mybroker -u admin -p grandpoobah
```

使用旧密码将失败。

更改密码后，使用任何 Message Queue 管理工具（包含管理控制台）时都应该提供新密码。

使用 LDAP 服务器管理用户信息库

如果要使用 LDAP 服务器管理用户信息库，必须在实例配置文件中设置某些代理属性。设置这些属性后，不论何时某个用户试图连接到代理实例或执行某些信息传送操作，代理实例将在 LDAP 服务器中查询用户和组的相关信息。此实例配置文件 (config.properties) 位于一个目录中，该目录用与此配置文件相关联的代理实例的名称 (instanceName) 标识（请参见附录 A “Message Queue 数据的位置”）：

```
.../instances/instanceName/props/config.properties
```

► 编辑配置文件以使用 LDAP 服务器

- 1. 通过设置以下属性，指定您正在使用 LDAP 用户信息库：

```
imq.authentication.basic.user_repository=ldap
```

- 2. 设置 imq.authentication.type 属性，确定是以 Base64 编码的形式 (basic) 还是以 MD5 摘要的形式 (digest) 将密码从客户机传送给代理。使用 LDAP 目录服务器管理用户信息库时，必须将验证类型设置为 basic。例如，

```
imq.authentication.type=basic
```

- 3. 还必须设置控制 LDAP 访问的代理属性。表 8-5 列出了存储在代理实例配置文件中的这些属性。Message Queue 使用 JNDI API 与 LDAP 目录服务器进行通信。有关这些属性的语法及其所引用术语的详细信息，请参见 JNDI 文档。Message Queue 使用 Sun JNDI LDAP 提供者并使用简单验证。

Message Queue 支持 LDAP 验证故障转移：可以为要试图进行的验证指定 LDAP 目录服务器列表（请参见表 8-5 中的 imq.user.repos.ldap.server 属性）。

表 8-5 LDAP 相关属性

属性	说明
imq.user_repository. ldap.server	LDAP 服务器的 <i>host:port</i> ，其中 <i>host</i> 指定运行目录服务器的主机的全限定 DNS 名称， <i>port</i> 指定目录服务器用于通信的端口号。要指定故障转移服务器的列表，请使用以下语法： <i>host1:port1 ldap://host2:port2 ldap://host3:port3...</i> 列表中的条目以空格分隔。请注意：第一个地址后的每个故障转移服务器的地址均以 ldap 开头。

表 8-5 LDAP 相关属性 (续)

属性	说明
imq.user_repository. ldap.principal	代理用来绑定到目录服务器以进行搜索时使用的独特名称。 如果目录服务器允许匿名搜索，则无需为此属性指定值。
imq.user_repository. ldap.password	与代理使用的独特名称关联的密码。只能在密码文件中指定 此属性（请参见第 204 页的“使用密码文件”）。 有多种提供密码的方式。最安全的方式是让代理提示您输入 密码。较安全的方式是使用密码文件并对密码文件进行读保 护。最不安全的方式是使用下列命令行选项指定密码： imqbrokerd -ldappassword。 如果目录服务器允许匿名搜索，则无需输入密码。
imq.user_repository. ldap.base	用户条目的目录库。
imq.user_repository. ldap.uidattr	提供者特定的属性标识符，其值用于唯一标识一个用户。例 如：uid、cn 等。
imq.user_repository. ldap.usrfilter	JNDI 搜索过滤器（以逻辑表达式的形式表示搜索查询）。 通过为用户指定搜索过滤器，代理可以缩小搜索范围，从而 使搜索更有效。有关详细信息，请参见位于以下位置的 JNDI 教程： http://java.sun.com/products/jndi/tutorial 。 此属性并不是必须设置的。
imq.user_repository. ldap.grpsearch	指定是否启用组搜索的布尔值。请参见 LDAP 提供者提供 的文档，决定您是否可以将用户与组相关联。 请注意，Message Queue 不支持嵌套组。 默认值：false
imq.user_repository. ldap.grpbase	组条目的目录库。
imq.user_repository. ldap.gidattr	提供者特定的属性标识符，其值为组名。
imq.user_repository. ldap.memattr	组条目中的属性标识符，其值是组成员的独特名称。
imq.user_repository. ldap.grpfiltler	JNDI 搜索过滤器（以逻辑表达式的形式表示搜索查询）。 通过为组指定搜索过滤器，代理可以缩小搜索范围，从而使 搜索更有效。有关详细信息，请参见位于以下位置的 JNDI 教程： http://java.sun.com/products/jndi/tutorial 此属性并不是必须设置的。
imq.user_repository. ldap.timeout	指定搜索的时间限制（以秒为单位）的一个整数。默认值 为 180 秒。

表 8-5 LDAP 相关属性 (续)

属性	说明
imq.user_repository. ldap.ssl.enabled	指定代理在与 LDAP 服务器通信时是否应使用 SSL 协议的布尔值。默认值为 false。

请参见代理的 `default.properties` 文件，查看样例（默认）LDAP 用户信息库相关属性的设置。

- 4. 必要时还需要编辑访问控制属性文件中的用户 / 组和规则。有关使用访问控制属性文件的详细信息，请参见第 192 页的“授权用户：访问控制属性文件”。
- 5. 如果您希望代理在连接验证和组搜索时通过 SSL 与 LDAP 目录服务器进行通信，您需要在 LDAP 服务器中激活 SSL，然后在代理配置文件中设置以下属性：
 - 指定 LDAP 服务器进行 SSL 通信时所使用的端口。例如：

```
imq.user_repository.ldap.server=myhost:7878
```
 - 将代理属性 `imq.user_repository.ldap.ssl.enabled` 设置为 `true`。

授权用户： 访问控制属性文件

连接到代理实例后，用户可能希望生成消息、在目标上使用消息或在队列目标上浏览消息。当用户试图进行此类操作时，代理将检查代理特定的 *访问控制属性文件*（ACL 文件），确认该用户是否有权执行此操作。ACL 文件包含指定某个用户（或一组用户）可以执行哪些操作的规则。默认情况下，所有授权用户都可以在任何目标上生成和使用消息。您可以编辑 ACL 文件，将这些操作限制到某些用户和组。

不管用户信息是放置在文本文件用户信息库（请参见第 184 页的“使用文本文件用户信息库”）中还是放置在 LDAP 用户信息库（请参见第 190 页的“使用 LDAP 服务器管理用户信息库”）中，都使用 ACL 文件。

创建访问控制属性文件

ACL 文件是特定于实例的。默认的文件（名为 `accesscontrol.properties`）是为启动的每个代理实例创建的。此 ACL 属性文件所在目录用与该 ACL 文件相关联的代理实例的名称 (*instanceName*) 标识（请参见附录 A “[Message Queue 数据的位置](#)”）：

```
.../instances/instanceName/etc/accesscontrol.properties
```

ACL 文件的格式与 Java 属性文件类似。首先定义文件的版本，然后指定访问控制规则，规则分为三部分：

- 连接访问控制
- 目标访问控制
- 目标自动创建访问控制

`version` 属性定义 ACL 属性文件的版本，不能更改此条目。

```
version=JMQFileAccessControlModel/100
```

下面介绍指定访问控制的 ACL 文件的三个组成部分，然后说明访问规则的基本语法并介绍如何计算权限。

访问规则语法

在 ACL 属性文件中，访问控制用于定义特定用户或组对受保护的资源（如目标和连接服务）具有哪些访问权限。访问控制由一个规则或一组规则组成，每个规则都由一个 Java 属性表示：

这些规则的基本语法如下：

```
resourceType.resourceVariant.operation.access.principalType = principals
```

表 8-6 介绍了语法规则的元素。

元素	说明
<i>resourceType</i>	下列选项之一：connection、queue 或 topic。
<i>resourceVariant</i>	<i>resourceType</i> 指定的类型的一个实例。例如，myQueue。通配符 (*) 可用于表示所有连接服务类型或所有目标。
<i>operation</i>	其值取决于所设置访问规则的类型。
<i>access</i>	下列选项之一：allow 或 deny。
<i>principalType</i>	下列选项之一：user 或 group。有关详细信息，请参见第 187 页的“组”。
<i>principals</i>	可能具有规则左侧指定了访问权限的人。如果 principalType 是 user，则可能是单个用户或以逗号分隔的用户列表；如果 principalType 是 group，则可能是单个组或以逗号分隔的组列表。通配符 (*) 可用于表示所有用户或所有组。

下面是一些访问规则实例：

- 以下规则表示所有用户都可以向名为 q1 的队列发送消息。
queue.q1.produce.allow.user=*
- 以下规则表示任何用户都可以向任何队列发送消息。
queue.*.produce.allow.user=*

注意	要指定非 ASCII 用户、组或目标名称，必须使用 Unicode 换码符 (\uxxxx) 表示法。如果在您编辑并保的 ACL 文件中，这些名称采用了非 ASCII 编码，则可以通过 Java native2ascii 工具将此文件转换为 ASCII。有关详细信息，请访问 http://java.sun.com/j2se/1.4/docs/guide/intl/faq.html
-----------	---

权限计算

计算一系列规则所代表的权限时需应用以下原则：

- 指定的访问规则将覆盖一般访问规则。应用以下两条规则之后，所有用户都可以向所有队列发送消息，但是 Bob 不能向 tq1 发送消息。
queue.*.produce.allow.user=*

```
queue.tq1.produce.deny.user=Bob
```

- 指定给明确 *principal* 的访问权限将覆盖指定给 ** principal* 的访问权限。以下规则将拒绝 Bob 向 tq1 发送消息，但允许其他人发送。

```
queue.tq1.produce.allow.user=*
```

```
queue.tq1.produce.deny.user=Bob
```

- 用户的 ** principal* 规则将覆盖组的对应 ** principal* 规则。例如，以下两条规则允许所有授权用户向 tq1 发送消息。

```
queue.tq1.produce.allow.user=*
```

```
queue.tq1.produce.deny.group=*
```

- 授予用户的访问权限将覆盖授予用户所属组的访问权限。在以下示例中，如果 Bob 是 "User" 组的成员，他将不能向 tq1 发送消息，但是 "User" 组中的其他成员则可以发送。

```
queue.tq1.produce.allow.group=User
```

```
queue.tq1.produce.deny.user=Bob
```

- 任何未通过访问规则明确授予的访问权限均默认为被拒绝。例如，如果 ACL 文件不包含任何访问规则，则所有用户的所有操作都将被拒绝。
- 如果同时允许和拒绝同一个用户或组的访问权限，则访问权限将相互抵消。例如，以下两条规则将使 Bob 无法浏览 q1：

```
queue.q1.browse.allow.user=Bob
```

```
queue.q1.browse.deny.user=Bob
```

以下两条规则将使 "User" 组无法使用 q5 中的消息。

```
queue.q5.consume.allow.group=User
```

```
queue.q5.consume.deny.group=User
```

- 如果多条规则等号左侧的内容都相同，那么只有最后一条规则起作用。

连接访问控制

ACL 属性文件中的连接访问控制部分包含代理连接服务的访问控制规则。连接访问控制规则的语法如下：

```
connection.resourceVariant.access.principalType = principals
```

为 *resourceVariant* 定义了两个值：NORMAL 和 ADMIN。默认情况下，所有用户都可以访问 NORMAL 类型的连接服务，但只有 admin 组中的用户可以访问 ADMIN 类型的连接服务。

您可以编辑连接访问控制规则，限制某个用户的连接访问权限。例如，以下规则将拒绝 Bob 访问 NORMAL，但允许其他人访问：

```
connection.NORMAL.deny.user=Bob
connection.NORMAL.allow.user=*
```

可以使用星号 (*) 指定所有授权用户或组。

不能创建自己的服务类型，只能使用常量 NORMAL 和 ADMIN 指定的预定义类型。

目标访问控制

访问控制属性文件的目标访问控制部分包含基于目标的访问控制规则。这些规则决定谁（用户 / 组）可以在哪里（目标）执行什么（操作）。这些规则控制的访问类型包括向队列发送消息、向主题发布消息、从队列接收消息、订阅主题以及浏览队列中的消息。

默认情况下，任何用户或组都能够对任何目标进行任意类型的访问。您可以添加更多特定的目标访问规则或编辑默认的规则。本节下面介绍目标访问规则的语法，必须理解该语法才能编写自己的规则。

目标规则的语法如下：

```
resourceType.resourceVariant.operation.access.principalType = principals
```

表 8-7 列出了这些元素：

表 8-7 目标访问控制规则的元素	
组件	说明
<i>resourceType</i>	必须是 queue 或 topic 其中之一。
<i>resourceVariant</i>	某个目标名或所有目标 (*), 星号表示所有队列或所有主题。
<i>operation</i>	必须是 produce、consume 或 browse 其中之一。
<i>access</i>	必须是 allow 或 deny 其中之一。
<i>principalType</i>	必须是 user 或 group 其中之一。

可以将访问权限授予一个或多个用户和 / 或一个或多个组。

以下示例说明了不同类型的目标访问控制规则：

- 允许所有用户向任意 `queue` 目标发送消息。
`queue.*.produce.allow.user=*`
- 拒绝 `user` 组的任何成员订阅 `Admissions` 主题。
`topic.Admissions.consume.deny.group=user`

目标自动创建访问控制

ACL 属性文件的最后一部分访问规则指定代理将为哪些用户和组自动创建目标。

当用户在尚不存在的目标上创建生成方或使用方时，如果代理的 `auto-create` 属性已启用且物理目标尚不存在，代理将会创建该目标。

默认情况下，任何用户或组都有权让代理为其自动创建一个目标。此特权由以下规则指定：

```
queue.create.allow.user=*
topic.create.allow.user=*
```

您可以编辑 ACL 文件以限制此类访问权限。

目标自动创建访问规则的一般语法如下：

```
resourceType.create.access.principalType = principals
```

其中 *resourceType* 为 `queue` 或 `topic`。

例如，以下规则将允许代理为除 `Snoopy` 之外的每个用户自动创建 `topic` 目标。

```
topic.create.allow.user=*
topic.create.deny.user=Snoopy
```

请注意，目标自动创建规则的效果必须与目标访问规则的效果一致。例如，如果您 1) 更改目标访问规则，禁止任何用户向目标发送消息；但是 2) 启用了目标的自动创建，那么如果目标不存在，代理将创建一个目标，但不会向该目标发送任何消息。

加密：使用基于 SSL 的服务（企业版）

Message Queue 企业版支持基于安全套接字层 (SSL) 标准的连接服务：通过 TCP/IP (`ssljms` 和 `ssladmin`) 和通过 HTTP (`httpsjms`)。这些基于 SSL 的连接服务允许对在客户机和代理之间传递的消息进行加密。当前的 Message Queue 发行版支持基于自签名服务器证书的 SSL 加密。

要使用基于 SSL 的连接服务，需要使用密钥工具实用程序 (`imqkeytool`) 生成专用密钥 / 公用密钥对。此实用程序将公用密钥嵌入自签名证书，此证书将传递给请求连接代理的客户机，客户机需要使用该证书建立加密连接。

尽管 Message Queue 基于 SSL 的连接服务在概念上很相似，但它们的设置方法却不尽相同。因此以下各节将分别讨论通过 TCP/IP 和通过 HTTP 的安全连接。

设置通过 TCP/IP 的基于 SSL 的服务

有三种基于 SSL 的连接服务通过 TCP/IP 提供直接、安全的连接。

ssljms 此连接服务用于在客户机和代理之间通过安全、加密的连接传送消息。

ssladmin 此连接服务用于在 Message Queue 命令行实用程序 (`imqcmd`)（命令行管理工具）和代理之间创建安全、加密的连接。不支持管理控制台 (`imqadmin`) 安全连接。

群集 此连接服务用于在群集的代理之间通过安全、加密的连接传送消息（请参见第 130 页的“安全的交叉代理连接”）。

► 设置基于 SSL 的连接服务

1. 生成自签名证书。
2. 在代理中启用 `ssljms` 连接服务、`ssladmin` 连接服务或群集连接服务。
3. 启动代理。
4. 配置并运行客户机（仅适用于 `ssljms` 连接服务）。

设置 `ssljms` 和 `ssladmin` 连接服务的过程基本相同，不同之处在于步骤 4，配置并运行客户机。

下文详细介绍了每个步骤。

步骤 1：生成自签名证书

Message Queue 的 SSL 支持用于保护所传输数据的安全，假定客户机正在与已知且可信任的服务器进行通信。因此，仅使用自签名证书实现 SSL。

运行 `imqkeytool` 命令，为代理生成自签名证书。可以对 `ssljms` 连接服务、`ssladmin` 连接服务或群集连接服务使用相同的证书。在命令提示符下输入以下内容：

```
imqkeytool -broker
```

命令行实用程序会提示您提供所需的信息。（在 UNIX® 操作系统上，可能需要以超级用户 (root) 身份运行 `imqkeytool` 命令，以获得创建密钥存储所需的权限。）

首先，`imqkeytool` 提示您输入密钥存储密码，然后提示您输入组织信息，最后提示您进行确认。收到您的确认后，它将在生成密钥时暂停。然后它要求您输入密码以锁定特定密钥对（关键字密码），此时请按回车键响应此提示：这将使关键字密码与密钥存储密码相同。

注意

请记住您提供的密码，稍后（启动代理时）您需要向代理提供此密码，使代理可以打开密钥存储。您还可以将密钥存储密码存储在密码文件中（请参见第 204 页的“使用密码文件”）。

运行 `imqkeytool` 命令以运行 JDK `keytool` 实用程序，生成自签名证书并将其放置 Message Queue 的密钥存储中。密钥存储所在的目录因操作系统而异，如附录 A “Message Queue 数据的位置” 中所示

密钥存储格式与 JDK1.2 `keytool` 实用程序支持的格式相同。

表 8-8 列出了 Message Queue 密钥存储的可配置属性。（有关配置这些属性的说明，请参见 Chapter 5 “启动与配置代理”）。

表 8-8 密钥存储属性

属性名称	说明
<code>imq.keystore.file.dirpath</code>	适用于基于 SSL 的服务：指定包含密钥存储文件的目录的路径。默认值：请参见附录 A “Message Queue 数据的位置”
<code>imq.keystore.file.name</code>	适用于基于 SSL 的服务：指定密钥存储文件的名称。默认值：Keystore

表 8-8 密钥存储属性（续）

属性名称	说明
imq.keystore.password	适用于基于 SSL 的服务：指定密钥存储密码。只能在密码文件中指定此属性（请参见第 204 页的“使用密码文件”）。 有多种提供密码的方式。最安全的方式是让代理提示您输入密码。较安全的方式是使用密码文件并对密码文件进行读保护。最不安全的方式是使用下列命令行选项指定密码： imqbrokerd -ldappassword。

您可能需要重新生成密钥对以解决某些问题，例如：

- 您忘记了密钥存储密码。
- 启动代理时，基于 SSL 的服务初始化失败并出现异常：
java.security.UnrecoverableKeyException:Cannot recover key。

出现异常的原因可能是您提供的密码与您在第 199 页的“步骤 1：生成自签名证书”中生成自签名证书时的密钥存储密码不同。

➤ 重新生成密钥对

1. 删除代理的密钥存储，它的位置如附录 A “Message Queue 数据的位置”中所示
2. 再次运行 imqkeytool，生成第 199 页的“步骤 1：生成自签名证书”中描述的密钥对。

步骤 2：在代理上启用基于 SSL 的服务

要在代理上启用基于 SSL 的服务，您需要将 ssljms（或 ssladmin）添加到 imq.service.activelist 属性中。

注意	基于 SSL 的 cluster 连接服务是使用 imq.cluster.transport 属性启用的，而不是使用 imq.service.activelist 属性启用的。请参见第 130 页的“安全的交叉代理连接”。
----	---

➤ 在代理中启用基于 SSL 的服务

1. 打开代理的实例配置文件。

实例配置文件位于一个目录中，该目录用与此配置文件相关联的代理实例的名称 (instanceName) 标识（请参见附录 A “Message Queue 数据的位置”）：


```
.../instances/instanceName/props/config.properties
```

2. 为 `imq.service.activelist` 属性添加一个条目（如果此条目尚不存在），并将基于 SSL 的服务包括在列表中。

默认情况下，此属性包括 `jms` 和 `admin` 连接服务。您需要添加 `ssljms` 或 `ssladmin` 连接服务或两个服务都添加（取决于您要激活的服务）：

```
imq.service.activelist=jms,admin,ssljms,ssladmin
```

步骤 3. 启动代理

启动代理并提供密钥存储密码。可以通过以下方法之一提供密码：

- 让代理启动时提示您输入密码：

```
imqbrokerd
Please enter Keystore password:mypassword
```

- 使用 `imqbrokerd` 命令的 `-password` 选项：

```
imqbrokerd -password mypassword
```

- 将密码放在代理启动时访问的密码文件中（请参见第 204 页的“使用密码文件”）。您需要首先设置以下代理配置属性（请参见第 118 页的“编辑实例配置文件”）：

```
imq.passfile.enabled=true
```

设置此属性后，您可以通过以下两种方式之一访问密码文件：

- 将密码文件的位置传递给 `imqbrokerd` 命令：

```
imqbrokerd -passfile /tmp/mypassfile
```

- 启动代理时不选择 `-passfile` 选项，但使用以下两个代理配置属性指定密码文件的位置：

```
imq.passfile.dirpath=/tmp
```

```
imq.passfile.name=mypassfile
```

要查看与密码文件相关的代理属性列表，请参见第 62 页的表 2-6。

启用带有 SSL 的代理或客户机时，您可能会注意到它在几秒钟内使用了大量的 CPU 资源。这是因为 Message Queue 使用 JSSE（Java 安全套接扩展）来实现 SSL。

JSSE 使用 `java.security.SecureRandom()` 生成随机数。此方法需要大量时间生成初始随机数初始化向量，这就是您看到 CPU 使用增加的原因。生成初始化向量后，CPU 的使用将降到正常水平。

步骤 4：配置并运行基于 SSL 的客户机

最后，您需要配置客户机以使用安全连接服务。根据您使用的连接服务，有两种类型的客户机：使用 `ssljms` 的应用程序客户机和使用 `ssladmin` 的 Message Queue 管理客户机（如 `imqcmd`）。下面分别介绍这两种客户机。

应用程序客户机

您需要确保客户机的类路径中有必需的安全套接扩展 (JSSE) Jar 文件，还需要告诉该文件使用 `ssljms` 连接服务。

1. 如果您的客户机不是使用 J2SDK1.4（它具有内置的 JSSE 和 JNDI 支持），请确保客户机的类路径中有以下 Jar 文件：

`jsse.jar`、`jnet.jar`、`jcrt.jar` 和 `jndi.jar`

2. 请确保客户机的类路径中有以下 Message Queue Jar 文件：

`imq.jar` 和 `jms.jar`

3. 启动客户机并连接到代理的 `ssljms` 服务。执行上述操作的方法之一是输入如下命令：

```
java -DimqConnectionType=TLS clientAppName
```

设置 `imqConnectionType`，通知连接使用 SSL。

有关在客户机应用程序中使用 `ssljms` 连接服务的详细信息，请参见《Message Queue Java Client Developer's Guide》中有关使用受管理对象的章节。

管理客户机 (`imqcmd`)

您可以通过在使用 `imqcmd` 时包含 `-secure` 选项来建立一个安全管理连接（请参见第 140 页的表 6-2），例如：

```
imqcmd list svc -b hostName:port -u adminName -p adminPassword -secure
```

其中 `adminName` 和 `adminPassword` 是 Message Queue 用户信息库中的有效条目。（如果使用的是文本文件信息库，请参见第 189 页的“更改默认的管理员密码”）。

列出连接服务是查看 `ssladmin` 服务是否正在运行的一种方法，您可以成功建立安全管理连接，如以下输出所示：

列出指定的代理上的所有服务：

Host	Primary Port	
localhost	7676	
Service Name	Port Number	Service State
admin	33984 (dynamic)	RUNNING
httpjms	-	UNKNOWN
httpsjms	-	UNKNOWN
jms	33983 (dynamic)	RUNNING
ssladmin	35988 (dynamic)	RUNNING
ssljms	dynamic	UNKNOWN

成功列出服务。

设置通过 HTTP 的 SSL 服务

在这种基于 SSL 的连接服务 (httpsjms) 中，客户机和代理通过 HTTPS 隧道 Servlet 建立安全连接。有关 HTTPS 支持所需的体系结构以及实现方案，请参见第 273 页上的附录 C “HTTP/HTTPS 支持（企业版）”。

使用密码文件

如果您不希望代理启动时提示您输入密码，或者要求您作为 `imqbrokerd` 命令选项提供密码，您可以将所需的密码放在 *密码文件* 中。然后可以在启动代理时使用 `-passfile` 选项指定密码文件：

```
imqbrokerd -passfile myPassfile
```

密码文件是包含密码的纯文本文件。该文件没有加密，因此没有手动提供密码安全。但您可以为该文件设置权限，限制哪些用户可以查看它。在密码文件中设置的权限需要为启动代理的用户提供读访问权限。

表 8-9 列出了密码文件可能包含的密码：

表 8-9 密码文件中的密码	
密码	说明
<code>imq.keystore.password</code>	为基于 SSL 的服务指定密钥存储密码（请参见第 199 页的表 8-8）。
<code>imq.user_repository.ldap.password</code>	指定与代理的独特名称（用于绑定到已配置的 LDAP 用户信息库）相关联的密码（请参见第 190 页的表 8-5）。
<code>imq.persist.jdbc.password</code>	指定必要时用于打开数据库连接的密码（请参见第 267 页的表 B-1）。

样例密码文件所在的位置因操作系统而异，如附录 A “Message Queue 数据的位置” 中所示。

分析和调整消息服务

本章包括大量有关如何分析和调整 Message Queue 服务以优化消息传送应用程序性能的主题。包括下列主题：

- [关于性能](#)
- [影响性能的因素](#)
- [监视消息服务器](#)
- [性能问题疑难解答](#)
- [调整配置以提高性能](#)

关于性能

性能调整过程

您的消息传送应用程序的性能取决于该应用程序与 Message Queue 服务之间的交互。因此，若要获得最佳性能，则需要应用程序开发者和管理员的共同努力。

优化性能的过程从应用程序设计开始，一直持续到部署应用程序之后对消息服务的调整阶段。性能调整过程包括下列阶段：

- 定义应用程序的性能要求
- 设计的应用程序要考虑到影响性能的因素（特别是可靠性和性能之间的折衷）
- 建立性能衡量基线
- 调整或重新配置消息服务以优化性能。

上述过程常常需要反复进行。在应用程序的部署过程中，**Message Queue** 管理员应该评估消息服务器是否适用于应用程序的总体性能要求。如果基准检验测试符合这些要求，管理员就可以按本章中所述调整系统。但是，如果基准检验测试不符合性能要求，则可能需要重新设计应用程序或者修改部署体系结构。

性能方面

通常，性能是对消息服务将消息从生成方传送到使用方时的速度和效率的一种衡量。但是，取决于您的需要，可能会有几个不同的性能方面对您特别重要。

连接负荷 系统所能支持的消息生成方、消息使用方或并行连接的数量。

消息吞吐量 每秒钟能通过消息传送系统抽取的消息数或消息字节数。

延迟 特定消息从消息生成方传送到消息使用方所需的时间。

稳定性 消息服务的总体可用性，或者当负荷较重或发生故障时性能的下降程度。

效率 消息的传送效率，这是一种与使用的计算资源相关的消息吞吐量的衡量。

这些不同的性能方面总而言之是相辅相成的。如果消息吞吐量很高，则意味着消息在消息服务器上作为待办事项累积的可能性就会变小，因此延迟时间也应该较低（单条消息可以很快地传送）。但是，延迟可能取决于许多因素：通信链接的速度、消息服务器的处理速度和客户机的处理速度，以及其他众多方面。

在任何情况下，性能都有几个不同方面。哪些方面对您最重要？这通常要取决于特定应用程序的要求。

基准检验

基准检验是为消息传送应用程序创建测试套件，并针对此测试套件衡量消息吞吐量或其他性能方面的一种过程。

例如，您可以创建这样一个测试套件：一定数量的生成方客户机使用一定数量的连接、会话和消息生成器，将标准大小的持久性或非持久性消息按照某个特定速率发送至队列或主题（这完全取决于消息传送应用程序的设计）。同样，测试套件中还包括一定数量的使用方客户机，它们使用一定数量的连接、会话和特定类型的消息使用器（通过特定的确认模式来使用测试套件的目标中的消息）。

使用标准的测试套件可以衡量消息的生成和使用之间所花的时间或者消息的平均吞吐速率，您还可以监视系统以观察连接线程使用情况、消息存储数据、消息流数据以及其他相关度量依据。这样就可以提高消息的生成速率、消息生成方的数量或者其他变量，直到性能受到负面影响为止。可能达到的最大吞吐量就是消息服务配置的基准。

使用此基准可以修改测试套件的某些特征。控制所有可能影响性能的因素时请小心（请参见第 209 页的“影响性能的应用程序设计因素”），您可以记录这些因素导致基准发生了哪些变化。例如，您可以将连接数或消息大小增加五倍或十倍，并记下对性能产生的影响。

相反，您也可以将基于应用程序的因素保持不变，而以某种控制方式更改代理配置（例如，更改连接属性、线程池属性、JVM 内存限制、限制行为、内置与插入的持久性等），并记下这些更改如何影响性能。

当您需要通过调整消息服务来增加所部署的应用程序的性能时，这种应用程序基准检验可以提供非常有价值的信息。基准检验可用于更准确地预测所作的更改或一系列更改产生的影响。

通常，基准检验应该在受控制的测试环境下运行，并且运行足够长的时间，以使消息服务能够稳定。（性能在启动时会受到实时编译的负面影响，因为要将 Java 代码转换为机器代码。）

基线使用模式

部署并运行消息传送应用程序后，建立基线使用模式是很重要的。您需要知道何时发生峰值需求，并且需要能够量化该需求。例如，需求通常会随最终用户的数量、活动级别、一天当中的时间，或者以上所有这些因素而发生波动。

要建立基线使用模式，您需要在一段较长的时间内监视消息服务器，并查看下列数据：连接数、存储在代理（或特定目标）中的消息数、流入和流出代理（或特定目标）的消息，以及活动的用户数目等等。还可以使用度量依据数据中提供的平均值和峰值。

将这些基线度量依据数据与设计时的期望值进行比较是非常重要的。通过执行此操作，可以检查客户机代码是否运转正常：例如，检查连接是否未保持打开或者已使用的消息是否仍然保留在未确认状态。这些编码错误会消耗消息服务器资源，并可能极大地影响性能。

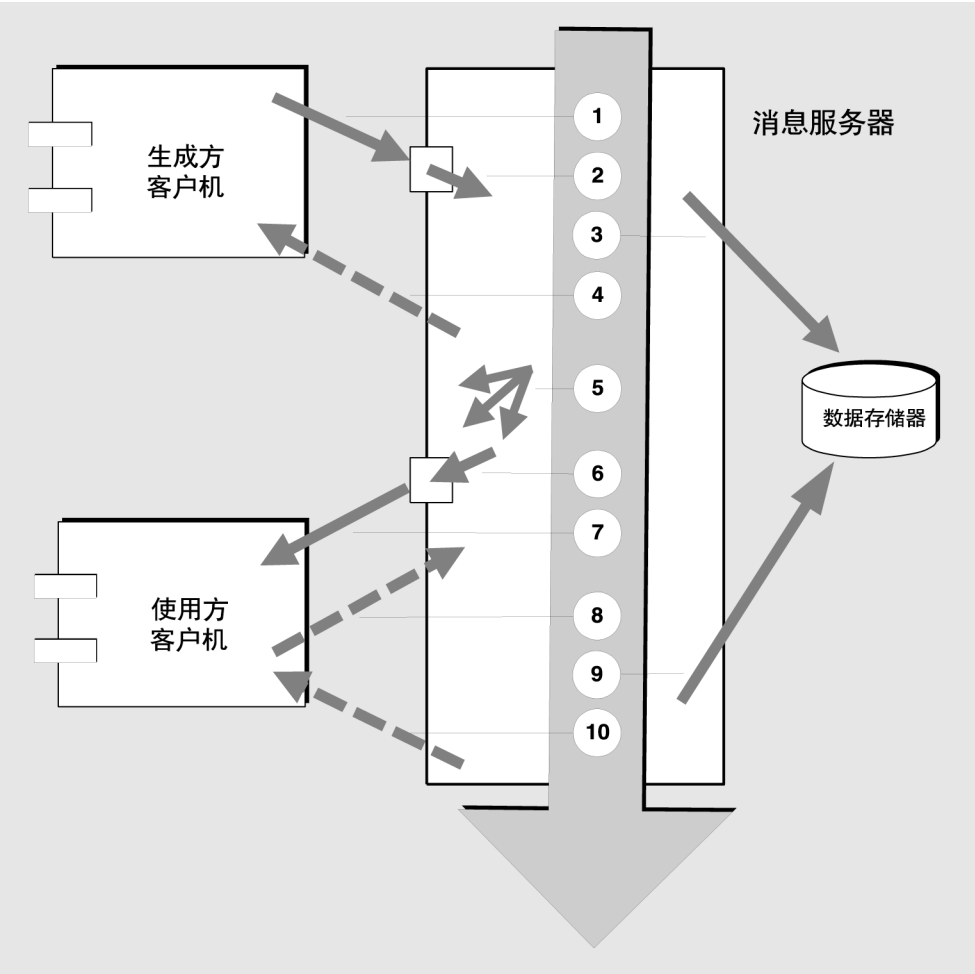
基线使用模式有助于确定如何调整系统以优化性能。例如，如果某个目标的使用频率明显高于其他目标，那么您可能需要对该目标设置比其他目标更高的内存限制，或者相应地调整限制行为。如果需要的连接数明显大于允许的最大线程池大小，那么可能需要增大线程池的大小，或者采用共享线程模型。如果峰值消息流远远大于平均消息流，这可能影响您在内存不足时采用的限制行为。

通常，您对使用模式了解得越多，就能越好地将系统调整为这些模式。并计划将来的需要。

影响性能的因素

消息延迟和消息吞吐量是两个主要的性能指示器，它们通常取决于典型消息在完成消息传送过程中的各个步骤时所需的时间。下面步骤显示了如何以持久、可靠的方式传送消息。这些步骤将在插图之后讲述。

图 9-1 通过 Message Queue 服务传送消息



1. 消息从生成方客户机传送到消息服务器
2. 消息服务器读取消息
3. 消息被放置到持久性存储器当中（出于可靠性的考虑）
4. 消息服务器确认收到消息（出于可靠性的考虑）
5. 消息服务器确定消息的路由
6. 消息服务器写出消息
7. 消息从消息服务器传送到使用方客户机
8. 使用方客户机确认收到消息（出于可靠性的考虑）
9. 消息服务器处理客户机的确认（出于可靠性的考虑）
10. 消息服务器确定已经处理客户机确认

由于这些步骤是连续的，所以它们中的任何一个都可能成为消息从生成方客户机到使用方客户机的传送过程中的瓶颈。这些步骤中的大多数都取决于消息传送系统的物理特征：网络带宽、计算机处理速度和消息服务器体系结构等等。但是，有一些步骤还取决于消息传送应用程序的特征和该应用程序要求的可靠性级别。

以下各小节讨论应用程序设计因素和消息传送系统因素这二者对性能的影响。尽管应用程序设计和消息传送系统因素在消息的传送过程中交互紧密，但它们彼此是独立的。

影响性能的应用程序设计因素

应用程序的设计决策对消息传送的总体性能有很大影响。

对性能影响最大的主要是那些影响消息传送的可靠性因素。包括下列因素：

- 传送模式（持久性 / 非持久性消息）
- 使用事务
- 确认模式
- 长期与非长期订阅

其他影响性能的应用程序设计因素有：

- 使用选择器（消息过滤）
- 消息大小
- 消息主体类型

接下来的几节讲述这些因素当中的每一个对消息传送性能所产生的影响。通常，性能和可靠性应该折衷。增加可靠性的因素可能会导致性能降低。

下表显示各个应用程序设计因素大体上如何影响消息传送性能。该表显示了两个方案（一个高可靠性、低性能方案和一个高性能、低可靠性方案）和分别相应于这两个方案的应用程序设计因素选项。在这两种极端情况之间，有很多可以同时影响可靠性和性能的选项和折衷。

表 9-1 高可靠性和高性能方案比较

应用程序设计因素	高可靠性 低性能方案	高性能 低可靠性方案
传送模式	持久性消息	非持久性消息
使用事务	事务会话	非事务
确认模式	AUTO_ACKNOWLEDGE 或 CLIENT_ACKNOWLEDGE	DUPS_OK_ACKNOWLEDGE
长期 / 非长期订阅	长期订阅	非长期订阅
使用选择器	消息过滤	无消息过滤
消息大小	小消息	大消息
消息主体类型	复杂主体类型	简单主体类型

注意 在接下来的图中，性能数据是在一个使用基于文件的持久性双 CPU、1002 Mhz Solaris 8 系统上生成的。性能测试首先预热 Message Queue 代理，使实时编译器优化系统，并启动持久性数据库。

代理预热后，会创建单个生成方和单个使用方，并在 30 秒钟内生成消息。系统将记录使用方接收所有生成的消息所需的时间，并计算吞吐率（每秒收到的消息数）。对于表 9-1 中所示的不同应用程序设计因素组合，此情形会重复出现。

传送模式（持久性 / 非持久性消息）

如第 43 页的“可靠消息传送”中所述，持久性消息能够在消息服务器发生故障的情况下保证消息的传送。代理会把消息存储在持久性存储库中，直到所有预期的使用方都确认已使用消息为止。

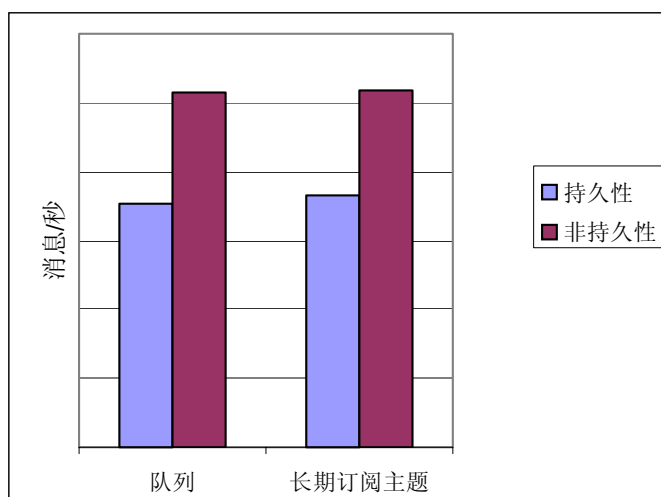
通过代理处理持久性消息比处理非持久性消息要慢，理由如下：

- 代理必须可靠地存储持久性消息，以使其即使在代理出现故障时也不会丢失。

- 代理必须确认它所收到的每一条持久性消息。如果生成消息的方法返回时没有异常，则对代理的传送即已确保成功。
- 根据客户机确认模式的不同，代理可能需要确认使用方客户机对持久性消息的确认。

持久性和非持久性模式的性能之间可以有很大的差别。图 9-2 比较了在两种可靠的传送情况下持久性和非持久性消息的吞吐量：同时将 10k 大小的消息传送到队列和长期订阅的主题。两种情况下都使用 `AUTO_ACKNOWLEDGE` 确认模式。

图 9-2 传送模式的性能影响



使用事务

事务是一种保证，保证在事务会话中生成和使用的消息将作为一个单元进行处理或不进行处理（回滚）。

Message Queue 同时支持本地和分布式事务（有关详细信息，请分别参见“本地事务”和第 44 页的“分布式事务”）。

消息在事务会话中生成或确认时比在非事务会话中要慢，理由如下：

- 其他信息必须随每个生成的消息存储。
- 在某些情况下，事务中通常不该存储的消息也存储了（例如，传送到没有订阅的主题目标的持久性消息通常应该删除，但在事务开始时，却没有关于订阅方面的信息）。

- 提交事务时，该事务中有关消息的使用和确认的信息必须存储并处理。

确认模式

确保可靠传送 JMS 消息的一种机制是，客户机确认使用了由 Message Queue 消息服务器传送给它的消息（请参见第 54 页的“可靠传送：确认和事务”）

如果在客户机尚未确认消息之前就关闭了会话，或者如果消息服务器在处理确认之前发生了故障，代理将重新传送该消息，并设置一个 JMSRedelivered 标志。

对于非事务会话，客户机可以选择三种确认模式之一，这三种模式有其各自的性能特征：

- `AUTO_ACKNOWLEDGE`。使用方处理消息后，系统会自动确认该消息。此模式可确保在提供者发生故障后至少重新传送一次消息。
- `CLIENT_ACKNOWLEDGE`。应用程序控制确认消息的时间点。该会话中自上次确认以来处理的所有消息都将被确认。如果消息服务器在处理一组确认时发生故障，则该组中的一个或多个消息将被重新传送。
- `DUPS_OK_ACKNOWLEDGE`。此模式指示系统以一种惰性方式确认消息。在提供者发生故障后，可以重新传送多条消息。

（`CLIENT_ACKNOWLEDGE` 模式的用法与事务的用法类似，不同之处在于：它不能保证当提供者在处理过程中发生故障时，所有确认都将一起处理）。

确认模式可以影响性能的理由如下：

- `AUTO_ACKNOWLEDGE` 和 `CLIENT_ACKNOWLEDGE` 模式需要在代理和客户机之间有额外的控制消息。附加的控制消息会添加额外的处理开销，并干扰 JMS 有效负荷消息，从而导致处理延迟。
- 在 `AUTO_ACKNOWLEDGE` 和 `CLIENT_ACKNOWLEDGE` 模式中，客户机必须等到代理确认它已处理客户机的确认后，才能使用其他消息。（这种代理确认可确保代理不会无意地重新传送这些消息。）
- Message Queue 持久性存储库必须采用使用方收到的所有持久性消息的确认信息进行更新，因而降低了性能。

长期与非长期订阅

主题目标的订阅者可以归为两类，即长期订阅和非长期订阅的订阅者，如第 42 页的“发布 / 订阅（主题目标）”中所述：

长期订阅的可靠性较高，但代价是吞吐较慢，原因如下：

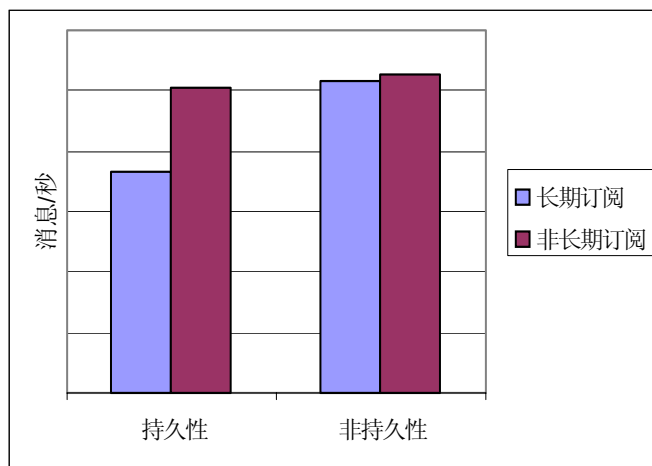
- Message Queue 消息服务器必须永久地存储指定给每个长期订阅的消息的列表，因此如果消息服务器发生故障，该列表只有在恢复之后才可用。

- 长期订阅的持久性消息是永久存储的，因此如果消息服务器发生故障，消息仍可以在恢复后传送（只要相应的使用方处于活动状态）。与此相反，非长期订阅的持久性消息不是永久存储的（如果消息服务器发生故障，相应的使用方的连接就会断开，消息不再被传送）。

图 9-3 比较了两种情况下的长期订阅和非长期订阅主题目标的吞吐量：10K 大小的持久性和非持久性消息。两种情况下都使用 `AUTO_ACKNOWLEDGE` 确认模式。

从图 9-3 中可以看到只有在持久性消息情况下，使用长期订阅的性能影响才会很明显，出现这种影响是因为持久性消息只有在长期订阅时才会永久存储，如上文所述。

图 9-3 订阅类型的性能影响



使用选择器（消息过滤）

应用程序的开发者常常需要将消息组的目标定为特定使用方。实现此愿望的方法有两种：将每组消息分别指向唯一的目标；或者是仅使用单个目标，并为每个使用方注册一个或多个选择器。

选择器是一种只请求特定消息的字符串，这些消息的属性值（请参见第 37 页的“JMS 消息结构”）应该与传送到特定使用方的字符串匹配。例如，选择器 `NumberOfOrders > 1` 仅传送 `NumberOfOrders` 属性值为 2 或更大值的消息。

将使用方注册到选择器会降低性能（与使用多个目标相比），因为会需要进行额外的处理，以处理每个消息。如果使用选择器，则必须对它进行解析，以使它与将来的消息匹配。另外，路由每个消息时，都必须检索该消息的属性，并与选择器相比较。但是，使用选择器为消息传送应用程序提供了更大的灵活性。

消息大小

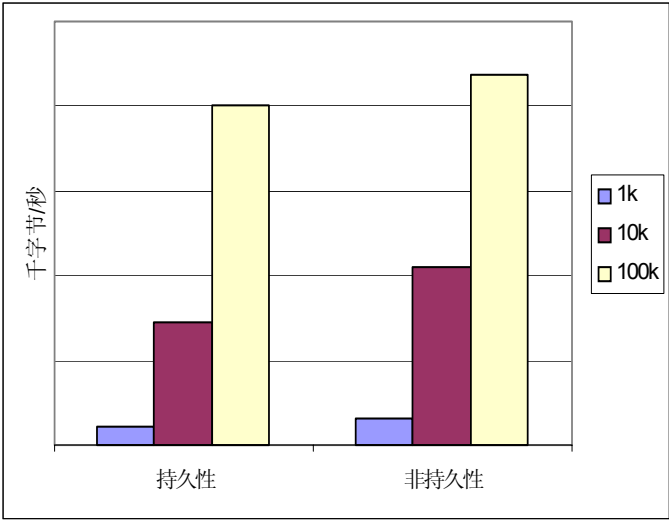
消息大小会影响性能是因为从生成方客户机到代理以及从代理到使用方客户机之间必须传递更多的数据，并且对于持久性消息，必须存储更大的消息。

但是，通过将较小的消息成批地变为一个消息，对各个消息的路由和处理就可以变得尽可能地简单，从而获得总体性能的提高。此时，就会丢失有关各个消息的状态的信息。

图 9-4 比较了在两种情况下，1k、10k 和 100k 大小的消息的吞吐量（KB/ 秒）：持久性和非持久性消息。每种情况都将消息发送至队列目标，并使用 AUTO_ACKNOWLEDGE 确认模式。

图 9-4 显示在这两种情况下，传送较大消息的开销都比传送较小消息时要少。您还可以看到在 1k 和 10k 大小的消息上出现的情况：非持久性消息较持久性消息接近 50% 的性能提高在 100k 大小的消息上并未得到体现，可能是因为在这种情况下，网络带宽已经成为消息吞吐量的瓶颈。

图 9-4 消息大小的性能影响



消息主体类型

JMS 支持五种消息主体类型，按照复杂性顺序大致显示如下：

- `BytesMessage`: 按照应用程序所确定的格式包含一组字节
- `TextMessage`: 一种简单的 `java.lang.String`

- `StreamMessage`: 包含 Java 基元值流
- `MapMessage`: 包含一组名 - 值对
- `ObjectMessage`: 包含 Java 序列化对象

尽管通常情况下消息类型是由应用程序的需要所决定的，但较复杂的类型（`MapMessage` 和 `ObjectMessage`）会增加性能成本，对数据进行序列化和反序列化的成本。性能成本取决于数据的简单或复杂程度。

影响性能的消息服务因素

消息传送应用程序的性能不但受应用程序设计的影响，而且还受执行消息路由和传送的消息服务的影响。

以下各节讨论影响性能的各个消息服务因素。了解这些因素的影响对于评估消息服务以及诊断并解决在部署的应用程序中可能发生的性能瓶颈来说至关重要。

Message Queue 服务中影响性能的最重要的因素有：

- 硬件
- 操作系统
- Java 虚拟机 (JVM)
- 连接
- 代理限制和行为
- 消息服务器体系结构
- 数据存储性能
- 客户机运行时配置

以下各节讲述这些因素中的每一个对消息传送性能所产生的影响。

硬件

对于 Message Queue 消息服务器和客户机应用程序而言，CPU 处理速度和可用内存都是消息服务性能的主要决定因素。大多数软件限制都可以通过增加处理能力来消除，而添加内存则可以同时提高处理速度和容量。但是，仅通过升级硬件来克服瓶颈通常过于昂贵。

操作系统

由于不同操作系统的效率不同，所以即使硬件平台相同，性能也会各不相同。例如，操作系统使用的线程模型会对消息服务器可以支持的并行连接数产生重要影响。在所有硬件都相同的情况下，Solaris 通常比 Linux 快，而后者通常又比 Windows 快。

Java 虚拟机 (JVM)

消息服务器是受主机 JVM 支持并运行在其中的一种 Java 进程。因此，JVM 处理是决定消息服务器路由和传送消息的速度和效率的重要因素。

特别是 JVM 的内存资源管理至关重要。必须为 JVM 分配足够的内存以满足不断增大的内存负荷。另外，JVM 将定期地释放未使用的内存，而这种内存释放会延迟消息的处理。JVM 内存堆越大，内存释放过程中可能遇到的延迟就越长。

连接

客户机和代理之间的连接数目和速度可能影响消息服务器可以处理的消息数以及消息的传送速度。

消息服务器连接限制

对消息服务器的所有访问都是通过连接的方式进行的。对并行连接数的任何限制都会影响可以同时使用消息服务器的生成方或使用方客户机的数目。

消息服务器的连接数通常由可用的线程数限制。Message Queue 使用了线程池管理器，您可以将其配置为支持专用线程模型或共享线程模型（请参见第 51 页的“[线程池管理器](#)”）。专用线程模型的速度非常快，因为每个连接都有专用的线程，但是连接数目受可用线程数的限制（每个连接都有一个输入线程和一个输出线程）。共享线程模型对连接数不加任何限制，但是在大量连接之间共享线程时会有明显的开销和吞吐量延迟，特别是当这些连接都很繁忙时。

传输协议

Message Queue 软件允许客户机使用各种低级别的传输协议与消息服务器进行通信。Message Queue 支持第 51 页的“[连接服务支持](#)”中所示的连接服务（及相应协议）。协议的选择基于应用程序的要求（加密、通过防火墙访问等），但是所作的选择会影响总体性能。

图 9-5 传输协议速度



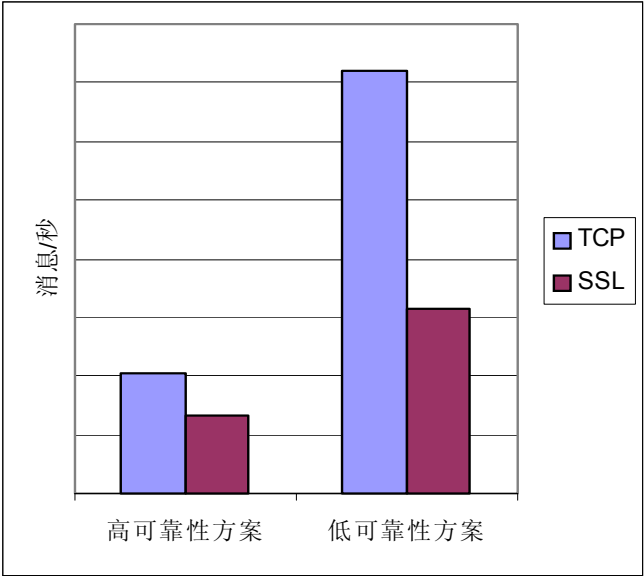
图 9-5 反映不同协议技术的性能特征：

- TCP 提供与代理通信的最快方法。
- SSL 在收发消息时比 TCP 要慢 50% 到 70%（对于持久性消息是 50%，对于非持久性消息则接近 70%）。另外，SSL 在建立初始连接时比较慢（可能需要数秒钟），因为客户机和代理（在使用 HTTPS 时为 Web 服务器）需要建立专用密钥，以供加密数据以进行传输时使用。性能的下降源自于加密和解密每个低级别 TCP 数据包时所需的额外处理。

图 9-6 比较了两种情况下的 TCP 和 SSL 吞吐量：一个高可靠性方案（将 1k 大小的持久性消息发送至长期订阅主题目标，并使用 AUTO_ACKNOWLEDGE 确认模式）和一个高性能方案（将 1k 大小的非持久性消息发送至非长期订阅主题目标，并使用 DUPS_OK_ACKNOWLEDGE 确认模式）。

图 9-6 显示了在高可靠性情况下产生较小影响的协议。这可能是因为在高可靠性情况下所需的持久性开销在限制吞吐量方面是比协议速度更重要的因素。

图 9-6 传输协议的性能影响



- HTTP 比 TCP 或 SSL 都要慢。它使用 `servlet`，该 `servlet` 在 Web 服务器上作为客户机和代理之间的代理服务器运行。性能开销涉及封装 HTTP 请求中的数据包，并涉及消息要经过两个跃点（客户机到 `servlet`，`servlet` 到代理）后才到达代理这一要求。
- HTTPS 比 HTTP 要慢是因为需要额外的开销以加密客户机和 `servlet` 之间以及 `servlet` 和代理之间的数据包。

消息服务器体系结构

Message Queue 消息服务器可以作为单个代理实现，也可以作为多个互相连接的代理实例（即代理群集）实现。

随着连接到代理的客户机数量以及所传送的消息数量的不断增加，代理最终将超出资源限制（例如文件描述符、线程和内存限制）。适应不断增加的负荷的方法之一是将更多的代理实例添加到 Message Queue 消息服务器，从而将客户机连接以及消息路由和传送分布到多个代理。

通常，如果客户机（特别是消息生成方客户机）均匀地分布在群集中，则这种调整最为有效。由于在群集的代理之间传送消息会涉及到开销，因此连接数有限或消息传送速率有限的群集，其性能可能会比单个代理要低。

您也可以使用代理群集来优化网络带宽。例如，您可能希望在群集内的一组远程代理之间使用低速的长途网络链路，而在客户机与其各自的代理实例之间使用高速链接进行连接。

有关群集的详细信息，请参见第 73 页的“多代理群集（企业版）”和第 128 页的“使用群集（企业版）”。

代理限制和行为

消息服务器可能需要处理的消息吞吐量是消息服务器所支持的消息传送应用程序的使用模式的一项功能。但是，消息服务器在资源方面有限：内存、CPU 周期等。因此，在消息服务器变得无响应或不稳定的位置，它可能会崩溃。

Message Queue 消息服务器具有管理内存资源并防止代理用尽内存的内置机制。这些机制包括可配置的消息数限制，或代理可以拥有的消息字节数或其各自的目标，以及当达到目标限制时可以实施的一组行为（请参见第 55 页的“管理内存资源和消息流”）。

通过仔细地监视和调整，这些可配置机制可以用于平衡消息的内流和外流，以使系统过载不会发生。尽管这些机制会造成开销并限制消息的吞吐量，但它们可以维护操作的完成性。

数据存储性能

Message Queue 既支持内置的持久性，也支持插入的持久性（请参见第 57 页的“持久性管理器”）。内置的持久性是基于文件的数据存储。插入的持久性使用 Java 数据库连接 (JDBC™) 接口，并需要 JDBC 兼容的数据存储。

内置的持久性明显比插入的持久性要快，但是 JDBC 兼容的数据库系统可以提供应用程序需要的冗余、安全性和管理功能。

如果是内置持久性，您可以通过指定让持久性操作将内存中状态与数据存储同步，从而将可靠性增至最大。这有助于消除因系统崩溃而导致的数据丢失，但代价是性能的下降。

客户机运行时配置

Message Queue 客户机运行时可提供客户机应用程序及其与 Message Queue 消息服务的接口。它支持客户机向目标发送消息以及接收来自这些目标的消息所需的所有操作。客户机运行时是可配置的（通过设置连接工厂属性值），使您可以设置通常能够提高性能和消息吞吐量的属性和行为。

例如，Message Queue 客户机运行时支持下列可配置行为：

- 连接流测量 (imgConnectionFlowCount)，此行为有助于防止由于 JMS 消息和 Message Queue 控制消息通过同一连接时所形成的流引起的拥塞。

- 连接流限制 (imgConnectionFlowLimit)，此行为有助于通过限制等待使用的、可以通过连接传送到客户机运行时的消息数，来避免客户机的资源限制。
- 用户流限制 (imgConsumerFlowLimit)，此行为有助于改进在多用户队列传送情况下用户之间的负荷平衡（因此向任何一个用户发送的消息数都是均衡的），并有助于阻止连接上的任何一个用户影响连接上的其他用户。此属性可以限制每个用户等待使用的、可以通过连接传送到客户机运行时的消息数。还可以将该属性配置为队列目标属性 (consumerFlowLimit)。

有关这些行为和用来配置这些行为的属性的详细信息，请参见第 258 页的“客户机运行时消息流调整”。

监视消息服务器

Message Queue 服务器经配置后可以提供能够用于监视其性能的度量依据信息。本节描述可以用于监视消息服务器的各种工具以及能够使用这些工具获得的度量依据数据。

有关如何使用度量依据数据来解决性能问题或者分析和调整消息服务器性能的信息，请参见第 237 页的“性能问题疑难解答”。

监视工具

可以使用下列工具获得度量依据信息：

- [Message Queue 命令行实用程序 \(imgcmd\)](#)
- [Message Queue 代理日志文件](#)
- [基于消息的监视 API](#)

以下各节描述如何使用这些工具中的每一种以获得度量依据信息。有关不同工具之间的比较，请参见第 229 页的“选择适当的监视工具”。

Message Queue 命令行实用程序 (imgcmd)

命令行实用程序 (imgcmd) 是 Message Queue 的基本命令行管理工具。通过它可以管理代理及其连接服务，以及应用程序特有的资源，例如物理目标、长期订阅和事务。imgcmd 命令在第 6 章“代理和应用程序管理”中有所说明

imqcmd 命令的功能之一是它为作为一个整体的代理、为单个的连接服务以及为单个的目标获得度量依据信息的能力。要获得度量依据数据，通常应该使用 imqcmd 的 metrics 子命令。度量依据数据将按照您指定的时间间隔或者次数写到控制台屏幕上。

您也可以使用 query 子命令（请参见第 224 页的“imqcmd query”）获得度量依据数据的更具体的子集。

imqcmd metrics

imqcmd metrics 的语法和选项分别如表 9-2 和表 9-3 所示。

表 9-2 imqcmd metrics 子命令语法

子命令语法	提供的度量依据数据
metrics bkr [-b <i>hostName:port</i>] [-m <i>metricType</i>] [-int <i>interval</i>] [-msp <i>numSamples</i>] [-u <i>userName</i>] [-p <i>password</i>	显示默认代理的代理度量依据，或显示指定主机和端口上的代理的代理度量依据。
或	
metrics svc -n <i>serviceName</i> [-b <i>hostName:port</i>] [-m <i>metricType</i>] [-int <i>interval</i>] [-msp <i>numSamples</i>] [-u <i>userName</i>] [-p <i>password</i>	显示默认代理或指定主机和端口上的代理中指定服务的度量依据。
或	
metrics dst -t <i>destType</i> -n <i>destName</i> [-b <i>hostName:port</i>] [-m <i>metricType</i>] [-int <i>interval</i>] [-msp <i>numSamples</i>] [-u <i>userName</i>] [-p <i>password</i>	显示指定类型和名称的目标的度量依据信息。

表 9-3 imqcmd metrics 子命令选项

子命令选项	说明
-b <i>hostName:port</i>	指定度量依据数据所报告的代理的主机名和端口。默认为 localhost:7676
-int <i>interval</i>	指定显示度量依据的时间间隔（以秒为单位）。默认值为 5 秒。
-m <i>metricType</i>	指定要显示的度量依据类型。 ttl 显示流入和流出代理的消息和数据包的度量依据（默认度量依据类型） rts 显示消息和数据包流入和流出代理的速率（每秒）的度量依据 cxn 显示连接、虚拟内存堆和线程（仅适用于代理和连接服务） con 显示与用户相关的度量依据（仅适用于目标） dsk 显示磁盘使用情况度量依据（仅适用于目标）
-msp <i>numSamples</i>	指定输出中显示的样例数。默认值为无限数（无穷大）。
-n <i>destName</i>	指定度量依据数据要报告的目标（如果有）的名称。没有默认值。
-n <i>serviceName</i>	指定度量依据数据要报告的连接服务（如果有）。没有默认值。
-t <i>destTyp</i>	指定度量依据数据要报告的目标（如果有）的类型（队列或主题）。没有默认值。
-u <i>userName</i>	指定您（管理员）的名称。如果省略此值，系统会提示您输入。
-p <i>password</i>	指定您（管理员）的密码。如果省略此值，系统会提示您输入。

过程：使用 metrics 子命令显示度量依据数据

本节讲述使用 metrics 子命令报告度量依据信息的过程。

► 要使用 metrics 子命令，请执行下列操作：

1. 启动度量依据信息需要的代理。
请参见第 123 页的“启动代理”。
2. 发出相应的 imqcmd metrics 子命令和选项，如表 9-2 和表 9-3 中所示。

度量依据输出: *imqcmd metrics*

本节显示代理范围、连接服务和目标度量依据的 *metrics* 子命令输出示例。

代理范围度量依据。 要获取在 10 秒的时间间隔内流入和流出代理的消息和数据包的速率，请使用 *metrics bkr* 子命令：

```
imqcmd metrics bkr -m rts -int 10 -u admin -p admin
```

此命令产生的输出结果类似如下内容（请参见第 231 页上的表 9-8 中的数据说明）：

Msgs/sec		Msg Bytes/sec		Pkts/sec		Pkt Bytes/sec	
In	Out	In	Out	In	Out	In	Out
0	0	27	56	0	0	38	66
10	0	7365	56	10	10	7457	1132
0	0	27	56	0	0	38	73
0	10	27	7402	10	20	1400	8459
0	0	27	56	0	0	38	73

连接服务度量依据。 要获取 *jms* 连接服务所处理的消息和数据包的累计总数，请使用 *metrics svc* 子命令：

```
imqcmd metrics svc -n jms -m ttl -u admin -p admin
```

此命令产生的输出结果类似如下内容（请参见第 233 页上的表 9-9 中的数据说明）：

Msgs		Msg Bytes		Pkts		Pkt Bytes	
In	Out	In	Out	In	Out	In	Out
164	100	120704	73600	282	383	135967	102127
657	100	483552	73600	775	876	498815	149948

目标度量依据。 要获取有关目标的度量依据信息，请使用 *metrics dst* 子命令：

```
imqcmd metrics dst -t q -n XQueue -m ttl -u admin -p admin
```

此命令产生的输出结果类似如下内容（请参见第 235 页上的表 9-10 中的数据说明）：

Msgs		Msg Bytes		Msg Count			Total Msg Bytes (k)			Largest
In	Out	In	Out	Current	Peak	Avg	Current	Peak	Avg	Msg (k)
200	200	147200	147200	0	200	0	0	143	71	0
300	200	220800	147200	100	200	10	71	143	64	0
300	300	220800	220800	0	200	0	0	143	59	0

要获取有关目标的用户的信息，请使用下面的 `metrics dst` 子命令：

```
imqcmd metrics dst -t q -n SimpleQueue -m con -u admin -p admin
```

此命令产生的输出结果类似如下内容（请参见第 235 页上的表 9-10 中的数据说明）：

Active Consumers			Backup Consumers			Msg Count		
Current	Peak	Avg	Current	Peak	Avg	Current	Peak	Avg
1	1	0	0	0	0	944	1000	525

imqcmd query

`imqcmd query` 命令的语法和选项如表 9-4 中所示，并带有该命令所提供的度量依据数据的说明。

表 9-4 `imqcmd query` 子命令语法

子命令语法	提供的度量依据数据
<code>query bkr</code> <code>[-b hostName:port]</code> <code>[-int interval]</code> <code>[-msp numSamples]</code>	有关当前存储在代理内存和持久性存储库中的消息个数和消息字节数目的信息（请参见第 144 页的“显示代理信息”）
或 <code>metrics svc -n serviceName</code> <code>[-b hostName:port]</code> <code>[-int interval]</code> <code>[-msp numSamples]</code>	有关指定的连接服务的当前已分配线程数和连接数的信息（请参见第 149 页的“显示连接服务信息”）
或	

表 9-4 imqcmd query 子命令语法 (续)

子命令语法	提供的度量依据数据
<code>metrics dst -t destType</code> <code>-n destName</code> <code>[-b hostName:port]</code> <code>[-int interval]</code> <code>[-msp numSamples]</code>	有关指定目标的当前生成方的数目、活动和备份使用方的数目以及存储在内存和持久性存储库中的消息数和消息字节数的信息（请参见第 156 页的“显示目标信息”）

注意 由于 imqcmd query 提供的度量依据数据有限，第 230 页的“度量依据数据说明”，一节的表中未提供此工具。

Message Queue 代理日志文件

Message Queue 日志记录器获取代理代码、调试器和度量依据生成器等生成的信息，并将该信息写到多个输出通道：标准输出（控制台）、日志文件以及 Solaris™ 平台上的 syslog 守护程序进程。日志记录器在第 64 页的“日志记录器”中有所说明。

您可以指定日志记录器所收集的信息类型以及写到每个输出通道的类型。特别地，您可以指定希望将度量依据度量依据信息写入到日志文件。

过程：使用代理日志文件报告度量依据数据

本节讲述使用代理日志文件报告度量依据信息的过程。有关配置日志记录器的一般信息，请参见第 133 页的“日志记录”。

► 要使用日志文件来报告度量依据信息，请执行下列操作：

- 1. 配置代理的度量依据生成功能：
 - a. 确认 imq.metrics.enabled=true
默认情况下，用于日志记录的度量依据生成功能会打开。
 - b. 将度量依据生成时间间隔设置为比较合适的秒数。
`imq.metrics.interval=interval`
此值可以在 config.properties 文件中设置，或者在启动代理时使用 `-metrics interval` 命令行选项进行设置。
- 2. 确认日志记录器是否收集度量依据信息：

```
imq.log.level=INFO
```

这是默认值。此值可以在 config.properties 文件中设置，或者在启动代理时使用 -loglevel *level* 命令行选项进行设置。

- 3. 确认日志记录器是否已设置为将度量依据信息写入日志文件：

```
imq.log.file.output=INFO
```

这是默认值。它可以在 config.properties 文件中设置。

- 4. 启动代理。

度量依据输出：日志文件

下面显示了到日志文件的代理度量依据输出样例（请参见表 9-7 和第 231 页上的表 9-8 中的度量依据数据说明）：

```
[21/Jul/2003:11:21:18 PDT]
Connections: 0      JVM Heap: 8323072 bytes (7226576 free) Threads: 0 (14-1010)
  In: 0 msgs (0bytes) 0 pkts (0 bytes)
  Out: 0 msgs (0bytes) 0 pkts (0 bytes)
Rate In: 0 msgs/sec (0 bytes/sec) 0 pkts/sec (0 bytes/sec)
Rate Out: 0 msgs/sec (0 bytes/sec) 0 pkts/sec (0 bytes/sec)
```

基于消息的监视 API

Message Queue 提供度量依据的监视功能，通过该功能，代理可以将度量依据数据写入到 JMS 消息，然后根据消息中包含的度量依据信息类型将它们发送到众多度量依据主题目标中的某一个目标。

您可以通过编写具有下列功能的客户机应用程序来访问此度量依据信息：订阅度量依据主题目标、使用这些目标中的消息以及处理消息中包含的度量依据信息。度量依据中介绍了通用方案。

共有五个度量依据主题目标，它们的名称以及传送到各个目标的度量依据消息的类型显示在表 9-5 中。

表 9-5 度量依据主题目标

主题名称	度量依据消息类型
mq.metrics.broker	代理度量依据
mq.metrics.jvm	Java 虚拟机度量依据
mq.metrics.destination_list	目标及其类型列表

表 9-5 度量依据主题目标 (续)

主题名称	度量依据消息类型
mq.metrics.destination.queue. monitoredDestinationName	具有指定名称的队列的目标度量依据
mq.metrics.destination.topic. monitoredDestinationName	具有指定名称的主题的目标度量依据

过程：设置基于消息的监视

本节讲述使用基于消息的监视功能来收集度量依据信息的过程。该过程包括客户机开发和管理任务这两方面。

► **要设置基于消息的监视，请执行下列操作：**

1. 编写度量依据监视客户机。

有关以编程方式编写订阅度量依据主题目标、使用度量依据消息并从这些消息中提取度量依据数据的客户机的说明，请参见 《*Message Queue Java Client Developer's Guide*》。
2. 通过在 config.properties 文件中设置代理属性来配置代理的度量依据消息生成方：
 - a. 启用度量依据消息生成。

设置 imq.metrics.topic.enabled=true

默认值为 true。
 - b. 设置生成度量依据消息的时间间隔（以秒为单位）。

设置 imq.metrics.topic.interval=interval

默认值为 60 秒。
 - c. 指定是否需要让度量依据消息成为持久性的消息（即，当代理发生故障时它们是否还能保留）。

设置 imq.metrics.topic.persist

默认值为 false。
 - d. 指定在删除度量依据消息之前它们能在各自的目标中保留的时间。

设置 imq.metrics.topic.timetolive

默认值为 300 秒。

3. 设置需要对度量依据主题目标设置的任何访问控制。

请参见下面的“[安全性和访问注意事项](#)”，中的讨论。

4. 启动度量依据监视客户机。

当用户订阅度量依据主题时，系统就会自动创建度量依据主题目标。创建度量依据主题后，代理的度量依据消息生成方就会开始向度量依据主题发送度量依据消息。

安全性和访问注意事项

出于以下两个原因，需要限制对主题目标的访问：

- 度量依据数据可能包含有关代理及其资源的敏感信息
- 对度量依据主题目标的过多订阅可能会增大代理开销，从而给性能带来负面影响

出于这些方面的考虑，建议限制对度量依据主题目标的访问。

监视客户机的验证和授权控制与其他任何客户机都相同。只有保持在 Message Queue 用户信息库中的用户才允许连接代理。

您可以通过某个访问控制属性文件，限制对特定度量依据主题目标的访问，从而提供其他保护，如第 192 页的“[授权用户：访问控制属性文件](#)”中所述。

例如，`accesscontrol.properties` 文件中的下列条目将拒绝除用户 1 和用户 2 之外的其他所有用户对 `mq.metrics.broker` 度量依据主题的访问。

```
topic.mq.metrics.broker.consume.deny.user=*
topic.mq.metrics.broker.consume.allow.user=user1,user2
```

下列条目将仅允许用户 `user3` 监视主题 `t1`。

```
topic.mq.metrics.destination.topic.t1.consume.deny.user=*
topic.mq.metrics.destination.topic.t1.consume.allow.user=user3
```

根据度量依据数据的敏感度不同，您也可以使用加密连接将度量依据监视客户机连接至代理。有关使用加密连接的信息，请参见第 198 页的“[加密：使用基于 SSL 的服务（企业版）](#)”。

度量依据输出：度量依据消息

使用基于消息的监视 API 获得的度量依据数据输出是您编写的度量依据监视客户机的一个函数。您只会受到代理中度量依据生成器提供的数据的限制。有关此数据的完整列表，请参见第 230 页的“度量依据数据说明”。

选择适当的监视工具

前面各节中讨论的每种监视工具都各有其优缺点。

例如，使用 `imqcmd metrics` 命令可以在您需要时迅速地抽取适合您需要的信息，但这会造成查看历史信息或者以编程方式操纵数据变得有些困难。

另一方面，日志文件可以提供度量依据数据的长期记录，但是日志文件中的信息很难解析为有意义的信息。

基于消息的监视 API 可用于轻松地提取您需要的信息、对其进行处理、以编程方式操纵或格式化数据以及提供图形或发送警报；但是您必须编写自定义应用程序以捕获并分析数据。

另外，这些工具中的每一种所收集到的由代理生成的度量依据信息的子集会稍有不同。有关哪些度量依据数据是由何种监视工具收集的信息，请参见第 230 页的“度量依据数据说明”。

表 9-6 从每种工具的正反两面对这些工具进行了比较。

表 9-6 度量依据监视工具的正面和反面因素

度量依据 监视工具	正面因素	反面因素
imqcmd metrics	远程监视	单条命令无法获取所有数据
	便于抽样检查	难于以编程方式分析数据
	在命令选项中设置报告的时间 间隔，可以随时更改	不能创建历史记录
	易于选择感兴趣的特定数据	难于看到历史趋势
	数据以简易的表格格式提供	

表 9-6 度量依据监视工具的正面和反面因素 (续)

度量依据 监视工具	正面因素	反面因素
日志文件	定期采样 创建历史记录	需要配置代理属性；必须关闭并重新启动代理才能生效 仅限本地监视 数据格式非常难于读取或解析；没有解析工具 报告时间间隔不能随时更改；所有度量依据数据也是如此 在数据的选择上不能提供灵活性 仅限代理度量依据；目标和连接服务度量依据不包括在内 如果时间间隔过短，性能可能会受影响
基于消息的监视 API	远程监视 易于选择感兴趣的特定数据 数据可以按照电子格式分析，并能够以任何格式提供	需要配置代理属性；必须关闭并重新启动代理才能生效 需要编写您自己的度量依据监视客户机 报告时间间隔不能随时更改；所有度量依据数据也是如此

度量依据数据说明

代理所报告的度量依据信息可以归为下列几类：

- **Java 虚拟机 (JVM) 度量依据。** 有关 JVM 堆大小的信息。
- **代理范围度量依据。** 有关存储在代理中的消息的信息以及有关流入和流出代理的消息的信息，这两种信息都以消息数和字节数表示（包括绝对数量和速率）。此类别还包括有关内存使用情况的信息。
- **连接服务度量依据。** 有关连接和连接线程资源的信息，以及有关特定连接服务的消息流的信息。
- **目标度量依据。** 有关流入和流出特定目标的消息的信息、有关目标的使用方的消息以及有关内存和磁盘空间使用情况的信息。

以下各节讲述每个类别中的可用度量依据数据。有关下列各表中提到的监视工具的信息，请参见第 220 页的“监视工具”。

JVM 度量依据

表 9-7 列出并说明了代理为代理进程 JVM 堆生成的度量依据数据，并显示了使用不同的度量依据监视工具可以获得哪些数据。

表 9-7 JVM 度量依据

度量依据数量	说明	imqcmd metrics bkr (metricType)	日志文 件	度量依据消息 (metrics topic) ²
JVM 堆： 可用内存	可用于 JVM 堆的内存量	是 (cxn)	是	是 (...jvm)
JVM 堆： 总内存	当前的 JVM 堆大小	是 (cxn)	是	是 (...jvm)
JVM 堆： 最大内存	JVM 堆可以达到的最大大小。	否	是 ¹	是 (...jvm)

- 1. 仅在代理启动时显示。
- 2. 有关度量依据主题目标的名称，请参见第 226 页上的表 9-5。

代理范围度量依据

表 9-8 列出并说明了代理所报告的有关代理范围内的度量依据信息的数据。它还显示了哪些数据可以使用不同的度量依据监视工具获得。

表 9-8 代理范围度量依据

度量依据数量	说明	imqcmd metrics bkr (metricType)	日志文 件	度量依据消息 (metrics topic) ¹
连接数据				
Num connections	当前对代理打开的连接数	是 (cxn)	是	是 (...broker)
Num threads	当前使用的线程数	是 (cxn)	是	否
Min threads	特定的线程数，达到该数量后，线程池中 将维护此数量的线程供连接服务使用	是 (cxn)	是	否
Max threads	特定的线程数，达到该数量后，线程池中 将不会添加新的线程以供连接服务使用	是 (cxn)	是	否
存储消息数据				

表 9-8 代理范围度量依据 (续)

度量依据数量	说明	imqcmd metrics bkr (metricType)	日志文 件	度量依据消息 (metrics topic) ¹
Num messages	当前存储在代理内存和持久性存储库中的 JMS 消息数	否 使用 query bkr	否	是 (...broker)
Total message bytes	当前存储在代理内存和持久性存储库中的 JMS 消息字节数	否 使用 query bkr	否	是 (...broker)
消息流数据				
Num messages in	自从代理上次启动以来流入该代理的 JMS 消息数	是 (ttl)	是	是 (...broker)
Message bytes in	自从代理上次启动以来流入该代理的 JMS 消息字节数	是 (ttl)	是	是 (...broker)
Num packets in	自从代理上次启动以来流入该代理的数据包 (包括 JMS 消息和控制消息) 数	是 (ttl)	是	是 (...broker)
Packet bytes in	自从代理上次启动以来流入该代理的数据包 (包括 JMS 消息和控制消息) 字节数	是 (ttl)	是	是 (...broker)
Num messages out	自从代理上次启动以来流出该代理的 JMS 消息数	是 (ttl)	是	是 (...broker)
Message bytes out	自从代理上次启动以来流出该代理的 JMS 消息字节数	是 (ttl)	是	是 (...broker)
Num packets out	自从代理上次启动以来流出该代理的数据包 (包括 JMS 消息和控制消息) 数	是 (ttl)	是	是 (...broker)
Packet bytes out	自从代理上次启动以来流出该代理的数据包 (包括 JMS 消息和控制消息) 字节数	是 (ttl)	是	是 (...broker)
Rate messages in	JMS 消息流入代理的当前速率	是 (rts)	是	否
Rate message bytes in	JMS 消息字节流入代理的当前速率	是 (rts)	是	否
Rate packets in	数据包 (包括 JMS 消息和控制消息) 流入代理的当前速率	是 (rts)	是	否
Rate packet bytes in	数据包 (包括 JMS 消息和控制消息) 字节流入代理的当前速率	是 (rts)	是	否
Rate messages out	JMS 消息流出代理的当前速率	是 (rts)	是	否
Rate message bytes out	JMS 消息字节流出代理的当前速率	是 (rts)	是	否

表 9-8 代理范围度量依据 (续)

度量依据数量	说明	imqcmd metrics bkr (metricType)	日志文件	度量依据消息 (metrics topic) ¹
Rate packets out	数据包 (包括 JMS 消息和控制消息) 流出代理的当前速率	是 (rts)	是	否
Rate packet bytes out	数据包 (包括 JMS 消息和控制消息) 字节流出代理的当前速率	是 (rts)	是	否
目标数据				
Num destinations	代理中的物理目标数	否	否	是 (...broker)

1. 有关度量依据主题目标的名称, 请参见第 226 页上的表 9-5。

连接服务度量依据

表 9-9 列出并说明了代理为各个连接服务所报告的度量依据数据。它还显示了哪些数据可以使用不同的度量依据监视工具获得。

表 9-9 连接服务度量依据

度量依据数量	说明	imqcmd metrics svc (metricType)	日志文件	度量依据消息 (metrics topic)
连接数据				
Num connections	当前打开的连接数	是 (cxn) 还有 query svc	否	否
Num threads	所有连接服务当前总计使用的线程数	是 (cxn) 还有 query svc	否	否
Min threads	特定的供所有连接服务使用的总计线程数, 达到该数量后, 线程池中将维护此数量的线程供连接服务使用	是 (cxn)	否	否
Max threads	特定的供所有连接服务使用的总计线程数, 达到该数量后, 线程池中将不会添加新的线程以供连接服务使用	是 (cxn)	否	否
消息流数据				
Num messages in	自从代理上次启动以来流入连接服务的 JMS 消息数	是 (ttl)	否	否

表 9-9 连接服务度量依据 (续)

度量依据数量	说明	imqcmd metrics svc (metricType)	日志 文件	度量依据消息 (metrics topic)
Message bytes in	自从代理上次启动以来流入连接服务的 JMS 消息字节数	是 (ttl)	否	否
Num packets in	自从代理上次启动以来流入连接服务的数据包 (包括 JMS 消息和控制消息) 数	是 (ttl)	否	否
Packet bytes in	自从代理上次启动以来流入连接服务的数据包 (包括 JMS 消息和控制消息) 字节数	是 (ttl)	否	否
Num messages out	自从代理上次启动以来流出连接服务的 JMS 消息数	是 (ttl)	否	否
Message bytes out	自从代理上次启动以来流出连接服务的 JMS 消息字节数	是 (ttl)	否	否
Num packets out	自从代理上次启动以来流出连接服务的数据包 (包括 JMS 消息和控制消息) 数	是 (ttl)	否	否
Packet bytes out	自从代理上次启动以来流出连接服务的数据包 (包括 JMS 消息和控制消息) 字节数	是 (ttl)	否	否
Rate messages in	JMS 消息通过连接服务流入代理的当前速率。	是 (rts)	否	否
Rate message bytes in	JMS 消息字节流入连接服务的当前速率	是 (rts)	否	否
Rate packets in	数据包 (包括 JMS 消息和控制消息) 流入连接服务的当前速率	是 (rts)	否	否
Rate packet bytes in	数据包 (包括 JMS 消息和控制消息) 字节流入连接服务的当前速率	是 (rts)	否	否
Rate messages out	JMS 消息流出连接服务的当前速率	是 (rts)	否	否
Rate message bytes out	JMS 消息字节流出连接服务的当前速率	是 (rts)	否	否
Rate packets out	数据包 (包括 JMS 消息和控制消息) 流出连接服务的当前速率	是 (rts)	否	否
Rate packet bytes out	数据包 (包括 JMS 消息和控制消息) 字节流出连接服务的当前速率	是 (rts)	否	否

目标度量依据

表 9-9 列出并说明了代理为各个目标所报告的度量依据数据。它还显示了哪些数据可以使用不同的度量依据监视工具获得。

表 9-10 目标度量依据

度量依据数量	说明	imqcmd metrics dst (metricType)	日志 文件	度量依据消息 (metrics topic) ¹
使用方数据				
Num active consumers	当前活动使用方数	是 (con)	否	是 (...destName)
Avg num active consumers	自从代理上次启动以来的平均活动使用方数	是 (con)	否	是 (...destName)
Peak num active consumers	自从代理上次启动以来的最大活动使用方数	是 (con)	否	是 (...destName)
Num backup consumers	当前备份使用方数 （仅适用于队列）	是 (con)	否	是 (...destName)
Avg num backup consumers	自从代理上次启动以来的平均备份使用方数 （仅适用于队列）	是 (con)	否	是 (...destName)
Peak num backup consumers	自从代理上次启动以来的最大备份使用方数 （仅适用于队列）	是 (con)	否	是 (...destName)
存储消息数据				
Num messages	当前存储在目标内存和持久性存储库中的 JMS 消息数	是 (con) (ttl) (rts) 还有 query dst	否	是 (...destName)
Avg num messages	自从代理上次启动以来存储在目标内存和持久性存储库中的平均 JMS 消息数	是 (con) (ttl) (rts)	否	是 (...destName)
Peak num messages	自从代理上次启动以来存储在目标内存和持久性存储库中的最大 JMS 消息数	是 (con) (ttl) (rts)	否	是 (...destName)
Total message bytes	当前存储在目标内存和持久性存储库中的 JMS 消息字节数	是 (ttl) (rts) 还有 query dst	否	是 (...destName)
Avg total message bytes	自从代理上次启动以来存储在目标内存和持久性存储库中的平均 JMS 消息字节数	是 (ttl) (rts)	否	是 (...destName)
Peak total message bytes	自从代理上次启动以来存储在目标内存和持久性存储库中的最大 JMS 消息字节数	是 (ttl) (rts)	否	是 (...destName)

表 9-10 目标度量依据 (续)

度量依据数量	说明	imqcmd metrics dst (metricType)	日志 文件	度量依据消息 (metrics topic) ¹
Peak message bytes	自从代理上次启动以来目标接收到的单个消息中 JMS 消息的最大字节数	是 (ttl) (rts)	否	是 (...destName)
消息流数据				
Num messages in	自从代理上次启动以来流入此目标的 JMS 消息数	是 (ttl)	否	是 (...destName)
Msg bytes in	自从代理上次启动以来流入此目标的 JMS 消息字节数	是 (ttl)	否	是 (...destName)
Num messages out	自从代理上次启动以来流出此目标的 JMS 消息数	是 (ttl)	否	是 (...destName)
Msg bytes out	自从代理上次启动以来流出此目标的 JMS 消息字节数	是 (ttl)	否	是 (...destName)
Rate num messages in	JMS 消息流入目标的当前速率	是 (rts)	否	否
Rate num messages out	JMS 消息流出目标的当前速率	是 (rts)	否	否
Rate msg bytes in	JMS 消息字节流入目标的当前速率	是 (rts)	否	否
Rate Msg bytes out	JMS 消息字节流出目标的当前速率	是 (rts)	否	否
磁盘使用情况数据				
Disk reserved	基于目标文件的存储库中所有消息记录（活动的和空闲的）使用的磁盘空间（以字节为单位）	是 (dsk)	否	是 (...destName)
Disk used	基于目标文件的存储库中活动消息记录使用的磁盘空间（以字节为单位）	是 (dsk)	否	是 (...destName)
Disk utilization ratio	已用磁盘空间除以保留磁盘空间所得的商。比率越高，用于容纳活动消息时所占用的磁盘空间就越多	是 (dsk)	否	是 (...destName)

1. 有关度量依据主题目标的名称，请参见第 226 页上的表 9-5。

性能问题疑难解答

使用 Message Queue 服务支持应用程序时，可能会发生很多性能问题。这些问题包括：

- 问题：客户机无法建立连接
- 问题：连接的吞吐量太慢
- 问题：客户机无法创建消息生成方
- 问题：消息的生成过程发生延迟或变慢
- 问题：消息在消息服务器中堆积
- 问题：消息服务器吞吐量呈间歇性
- 问题：消息无法到达使用方

下面会逐个讨论这些问题，并说明这些问题的可能起因及解决方案。

问题：客户机无法建立连接

症状：

- 客户机无法建立新连接。
- 客户机无法对失败的连接进行自动重新连接。

可能原因：

- 客户机应用程序不是闭合连接，导致连接数超出资源限制。

要确认这是否就是问题的起因，请执行下列操作：

列出代理的所有连接：

```
imqcmd list cxn
```

输出结果将列出所有的连接以及发起每个连接的主机，从而展示特定客户机的特有的打开连接数。

要解决此问题，请执行下列操作：

重写有问题的客户机，以关闭未使用的连接。

- 代理未运行或者网络连接有问题。

要确认这是否就是问题的起因，请执行下列操作：

- Telnet 到代理的主端口（例如，默认端口为 7676），并验证代理是否以端口映射器输出结果作为响应。
- 验证代理进程是否正在主机上运行。

要解决此问题，请执行下列操作：

- 启动代理。
- 修复网络连接问题。
- 连接服务处于非活动状态或者已暂停。

要确认这是否就是问题的起因，请执行下列操作：

检查所有连接服务的状态：

```
imqcmd list svc
```

如果某个连接服务的状态显示为 `unknown` 或 `paused`，则客户机将无法使用该服务建立连接。

要解决此问题，请执行下列操作：

- 如果连接服务的状态显示为 `unknown`，则它将从活动服务列表中消失 (`imq.service.active`)。如果是基于 SSL 的服务，那么还可能是因为服务未被正确配置，导致代理在代理日志中生成下面的条目：`ERROR [B3009]: Unable to start service ssljms: [B4001]:Unable to open protocol tls for ssljms service...`，后面带有关于引起此异常的解释。

要正确配置 SSL 服务，请参见第 198 页的“[设置通过 TCP/IP 的基于 SSL 的服务](#)”。
- 如果连接服务的状态显示为 `paused`，则可以恢复该服务（请参见第 150 页的“[暂停和恢复连接服务](#)”）。
- 连接所要求的线程数不够。

要确认这是否就是问题的起因，请执行下列操作：

在代理日志中检查下面的条目：`WARNING [B3004]: No threads are available to process a new connection on service ...Closing the new connection.`

另请检查连接服务上的连接数以及当前使用的线程数：

```
imqcmd query svc -n serviceName  
或  
imqcmd metrics svc -n serviceName -m cxn
```

每个连接都需要两个线程：一个用于外来消息，另一个用于外出消息（请参见第 51 页的“线程池管理器”）。

要解决此问题，请执行下列操作：

- 如果使用专用线程池模型 (`imq.service_name.threadpool_model=dedicated`)，则最大连接数是该线程池中的最大线程数的一半。因此，要增加连接数，请增加线程池的大小 (`imq.service_name.max_threads`) 或切换到共享线程池模型。
- 如果使用共享线程池模型 (`imq.service_name.threadpool_model=shared`)，则最大连接数是下面两个属性的乘积的一半。连接监视限制 (`imq.service_name.connectionMonitor_limit`) 和最大线程数 (`imq.service_name.max_threads`)。因此，要增加连接数，可以增加线程池的大小或增大连接监视限制。
- 最终，可支持的连接数（或连接上的吞吐量）将达到输出 / 输出限制。此时，可以使用多代理群集（请参见第 128 页的“使用群集（企业版）”）在群集内的代理实例之间分布连接。
- Solaris 或 Linux 平台上连接数所需的文件描述符不足（请参见第 302 页的“OS 定义的文件描述符限制”）。

要确认这是否就是问题的起因，请执行下列操作：

在代理日志中检查类似下面的条目：Too many open files。

要解决此问题，请执行下列操作：

增大文件描述符限制，如 ulimit 手册页中所述。

- TCP 待办事项限制了可以同时建立的新连接请求数目。

TCP 待办事项设置可以存储在系统待办事项 (`imq.portmapper.backlog`) 中的同时连接请求数限制后，端口映射器才会拒绝额外的请求。（在 Windows 平台上，有一种硬编码的待办事项限制：Windows 台式机限制为 5，而 Windows 服务器限制为 200。）

出于待办事项限制而拒绝请求通常是一种由于同时连接请求数过多而导致的瞬态现象。

要确认这是否就是问题的起因，请执行下列操作：

检查代理日志，以查看是否在某些连接请求被接受的同时还有另外一些连接请求被拒绝。被拒绝的连接请求会返回 "java.net.ConnectException: Connection refused" 消息。

要解决此问题，请执行下列操作：

下列方法可用于解决 TCP 待办事项限制：

- 对客户机进行编程，使其在较短的时间间隔后重试所尝试的连接（此方法之所以生效通常是由于此问题的瞬态性）。
- 增大 `imq.portmapper.backlog` 的值。
- 检查客户机是否过于频繁地反复关闭和打开连接。
- 操作系统限制了并行连接数。

Windows 操作系统许可证对支持的并行远程连接数进行了限制。

要确认这是否就是问题的起因，请执行下列操作：

检查是否有可用于连接的足够线程（使用 `imqcmd query svc`），并检查您的 Windows 许可协议的条款。如果您可以从本地客户机建立连接，但不能从远程客户机建立连接，则操作系统的限制可能就是问题的起因。

要解决此问题，请执行下列操作：

- 升级 Windows 许可证，以允许更多的连接。
- 通过设置多代理群集将连接分布在大量代理实例之间。
- 对用户的验证或授权失败。

验证可能因为密码错误而失败，原因是在用户信息库中没有该用户的条目，或者该用户没有对连接服务的访问权限。

要确认这是否就是问题的起因，请执行下列操作：

检查代理日志中的条目，判断是否有 "Forbidden" 错误消息。此消息指明存在验证错误，但不会指明该错误的原因。

- 如果使用的是基于文件的用户信息库，请输入下面的命令：

```
imqusermgr list -i instanceName -u userName
```

如果输出结果显示的是用户，则说明可能提交了错误的密码。如果输出结果显示错误 [B3048]: User does not exist in the password file, 则说明用户信息库中没有该用户的条目。

- 如果使用的是 LDAP 服务器用户信息库，请使用相应的工具检查是否存在该用户的条目。
- 检查访问控制属性文件以查看是否对连接服务有访问限制。

要解决此问题，请执行下列操作：

- 如果在用户信息库中没有该用户的条目，那么将该用户添加到用户信息库中（请参见第 188 页的“填充和管理用户信息库”）。
- 如果使用了错误的密码，请提供正确的密码。

- 如果访问控制属性设置不当，则编辑访问控制属性文件以授予连接服务权限（请参见第 195 页的“连接访问控制”）。

问题：连接的吞吐量太慢

症状：

- 消息的吞吐量与预期不符。
- 支持的代理连接数受到限制不是如第 237 页的“问题：客户机无法建立连接”中所述的原因，而是出于消息的输入 / 输出速率。

可能原因：

- 网络连接或 WAN 太慢。

要确认这是否就是问题的起因，请执行下列操作：

Ping 网络，查看返回 ping 需要多长时间，然后咨询网络管理员。另外还可以使用本地客户机发送并接收消息，并将传送时间与远程客户机（使用网络链路）的传送时间相比。

要解决此问题，请执行下列操作：

如果连接太慢，则升级网络链路。

- 与 TCP 相比，连接服务协议本身就慢。例如，基于 SSL 或基于 HTTP 的协议比 TCP 要慢（请参见第 217 页上的图 9-5）。

要确认这是否就是问题的起因，请执行下列操作：

如果您使用的是基于 SSL 或基于 HTTP 的协议，请尝试使用 TCP，然后比较传送时间。

要解决此问题，请执行下列操作：

应用程序的要求通常会限定要使用的协议，因此您可以做的事情就很少了，无非是按第 253 页的“调整传输协议”中所述尝试调整协议而已。

- 连接服务协议未优化调整。

要确认这是否就是问题的起因，请执行下列操作：

尝试调整协议，并查看是否发生了变化。

要解决此问题，请执行下列操作：

尝试按第 253 页的“调整传输协议”中所述调整协议。

- 消息太大，以致于占用了太多的带宽。

要确认这是否就是问题的起因，请执行下列操作：

尝试使用较小的消息运行基准检验。

要解决此问题，请执行下列操作：

- 使用 `java.util.zip` 压缩消息主体。
- 将消息作为要发送的数据的通知来使用，而使用其他协议移动数据。
- 使连接吞吐量变慢的可能原因也就是消息传送过程中某个步骤的瓶颈。

要确认这是否就是问题的起因，请执行下列操作：

如果上述的每一条看来都不是造成连接吞吐量变慢的原因，请参见[第 208 页上的图 9-1](#) 以了解其他可能的瓶颈，并检查与下列问题相关的症状：

- [第 243 页的“问题：消息的生成过程发生延迟或变慢”](#)
- [第 246 页的“问题：消息在消息服务器中堆积”](#)
- [第 249 页的“问题：消息服务器吞吐量呈间歇性”](#)

要解决此问题，请执行下列操作：

遵循前面有关问题疑难解答的各节中提供的问题解决方案指导。

问题： 客户机无法创建消息生成方

症状：

- 无法为目标创建消息生成方，客户机收到异常。

可能原因：

- 目标被配置为仅允许有限数量的生成方。

限制某个目标所能支持的生成方 (`maxNumProducers`) 数目是避免消息在该目标上堆积的方法之一。

要确认这是否就是问题的起因，请执行下列操作：

检查目标（请参见[第 156 页的“显示目标信息”](#)）：

```
imgcmd query dst
```

输出结果将显示当前的生成方数目以及 `maxNumProducers` 的值。如果这两个值相同，则说明生成方的数量已达到所配置的限制。如果新的生成方被代理拒绝，代理将返回 "ResourceAllocationException [C4088]: A JMS destination limit was reached" 消息，且在代理日志中生成如下条目: [B4183]: Producer can not be added to destination。

要解决此问题，请执行下列操作：

增大 `maxNumProducers` 属性的值（请参见第 157 页的“更新目标属性”）。

- 由于访问控制属性文件中的设置，用户未获得创建消息生成方的授权。

要确认这是否就是问题的起因，请执行下列操作：

如果新的生成方被代理拒绝，代理将返回 "JMSSecurityException [C4076]: Client does not have permission to create producer on destination" 消息，并在代理日志中生成如下条目: [B2041]: Producer on destination denied 和 [B4051]: Forbidden guest。

要解决此问题，请执行下列操作：

更改访问控制属性，允许用户生成消息（请参见第 196 页的“目标访问控制”）。

问题：消息的生成过程发生延迟或变慢

症状：

- 发送持久性消息时，`send()` 方法并没有返回，且客户机发生阻塞。
- 发送持久性消息时，客户机收到异常。
- 生成方客户机速度变慢。

可能原因：

- 消息服务器上堆满了待办事项（消息堆积在代理的内存中），从而作出使消息生成方减慢的响应。

当目标内存中的消息数或消息字节数达到配置的限制时，代理会尝试根据指定的限制行为节约内存资源。下列限制行为会使消息生成方变慢：

- `FLOW_CONTROL`：代理不会立即确认收到持久性消息（这样就会阻塞生成方客户机）
- `REJECT_NEWEST`：代理拒绝新消息（并为每个被拒绝的持久性消息发出异常）。

同样，如果代理范围的内存（对于所有目标）中消息数或消息字节数达到配置的限制，代理将尝试通过拒绝最新的消息来节约内存资源。

另外，如果达到了系统内存限制（由于目标或代理范围限制设置不正确），代理将采取越来越严格的操作来防止内存过载，包括限制消息生成方。

有关这些机制的讨论，请参见第 55 页的“管理内存资源和消息流”。

要确认这是否就是问题的起因，请执行下列操作：

如果某个消息因为达到了配置的消息限制而被代理拒绝，代理将返回 "JMSEException [C4036]: A server error occurred"，并在代理日志中生成下列条目：WARNING [B2011]: Storing of JMS message from IMQconn failed，后面跟有一条表明已达到限制的消息：

- 如果消息限制位于目标上，则代理将生成类似下面的条目：[B4120]: Can not store message on destination *destName* because capacity of *maxNumMsgs* would be exceeded。
- 如果消息限制是代理范围的，则代理将生成类似下面的条目：[B4024]: The Maximum Number of messages currently in the system has been exceeded, rejecting message。

大多数情况下，您可以通过两种方式在发生拒绝之前检查消息限制情况：分别使用相应的 `imqcmd` 命令（请分别参见第 235 页上的表 9-10 和第 231 页上的表 9-8）查询目标和代理并检查它们所配置的消息限制设置，以及监视当前在目标上（或者作为一个整体在代理中）的消息数或消息字节数。

要解决此问题，请执行下列操作：

有很多方法可以解决由于消息变得堆积而导致生成方变慢的问题。

- 修改目标（或代理范围）上的消息限制，请小心不要超出内存资源。通常，需要根据每个目标来管理内存，这样才不会达到代理范围的消息限制。有关详细信息，请参见第 256 页的“代理调整”。
- 将目标上的限制行为更改为当达到消息限制时并不减慢消息的生成，而是在内存中废弃消息。例如，您可以指定 `REMOVE_OLDEST` 和 `REMOVE_LOW_PRIORITY` 限制行为，这些行为可以删除在内存中堆积的消息（请参见第 154 页上的表 6-10）。
- 代理无法将持久性消息保存到数据存储中。

如果代理无法访问数据存储或者将持久性消息写入数据存储，则生成方客户机将阻塞。如果达到了如前所述的目标或代理范围消息限制，也将发生此情况。

要确认这是否就是问题的起因，请执行下列操作：

如果代理无法写入数据存储库，它将在代理日志中生成下列条目之一：
[B2011]: Storing of JMS message from connectionID failed... 或
[B4004]:Failed to persist message messageID...

要解决此问题，请执行下列操作：

- 如果是内置持久性，则尝试增大基于文件的数据存储的磁盘空间。
- 如果是 JDBC 兼容的数据存储，则检查插入的持久性是否正确配置（请参见附录 B “设置插入的持久性”）。如果是这样，请向数据库管理员咨询，以解决其他数据库问题。
- 代理的确认超时太短。

如果连接太慢或消息服务器反应迟缓（由于 CPU 使用率太高或者内存资源不足导致），代理用于确认收到持久性消息的时间比连接工厂的 `imqAckTimeout` 属性的值允许的时间要长。

要确认这是否就是问题的起因，请执行下列操作：

如果超出了 `imqAckTimeout` 值，代理将返回 "JMSEException [C4000]: Packet acknowledge failed"。

要解决此问题，请执行下列操作：

更改 `imqAckTimeout` 连接工厂属性的值（请参见第 168 页的“连接工厂受管理对象属性”）。

- 生成方客户机遇到了 JVM 限制。

要确认这是否就是问题的起因，请执行下列操作：

- 检查客户机应用程序是否收到了“内存不足”错误。
- 使用诸如 `freeMemory()`、`MaxMemory()` 和 `totalMemory()` 等运行时方法检查 JVM 堆中的可用内存。

要解决此问题，请执行下列操作：

调整 JVM（请参见第 252 页的“Java 虚拟机调整”）。

问题：消息在消息服务器中堆积

症状：

- 代理（或特定目标）中的消息数目或消息字节数随着时间稳定增加。

要查看消息是否在堆积，请检查代理中的消息数目或消息字节数如何随时间而改变，并比较已配置的限制。首先检查配置的限制：

```
imqcmd query bkr
```

（注意：imqcmd metrics bkr 子命令不会显示此信息）。

然后检查每个目标中的消息堆积情况。

```
imqcmd query dst -t destType -n destName
或
imqcmd metrics dst -t destType -n destName -m ttl
```

要检查消息是否已超出配置的目标或代理范围限制，请在代理日志中检查如下条目：WARNING [B2011]: Storing of JMS message from...failed。此条目后面接有另一个阐明限制已超出的条目。

- 消息的生成发生了延迟，或者生成的消息被代理拒绝。
- 消息到达使用方的时间过长。

可能原因：

- 客户机代码缺陷：使用方不确认消息。

消息会保留在目标中，直到消息所发送到的所有使用方都进行了确认为止。因此，如果客户机没有确认已使用的消息，该消息会在目标中堆积，而不会删除。

例如，客户机代码可能会存在下列缺陷：

- 使用 CLIENT_ACKNOWLEDGEMENT 或事务会话的使用方可能没有定期调用 `Session.acknowledge()` 或 `Session.commit()`。
- 使用 AUTO_ACKNOWLEDGE 会话的使用方可能因为某种原因而挂起。

要确认这是否就是问题的起因，请执行下列操作：

如果消息服务器不繁忙，即消息流入和流出目标的速率很低，则说明消息可能由于未被确认而堆积。

检查消息流入和流出代理的速率。

```
imqcmd metrics bkr -m rts
```

然后检查每个单独目标的流速：

```
imqcmd metrics bkr -t destType -n destName -m rts
```

再检查客户机代码以查看消息是否被正确确认。

- 主题目标上有非活动的长期订阅。

如果长期订阅是非活动的，则消息会存储在目标中，直到相应的使用方变为活动状态且能够使用这些消息为止。

要确认这是否就是问题的起因，请执行下列操作：

检查每个主题目标上的长期订阅的状态：

```
imqcmd list dur -d destName
```

要解决此问题，请执行下列操作：

可以采用下列任意操作：

- 清除所有存在问题的长期订阅的消息（请参见第 161 页的“管理长期订阅”）。
- 指定主题的消息限制以及限制行为属性（请参见第 154 页上的表 6-10）。例如，您可以指定 REMOVE_OLDEST 和 REMOVE_LOW_PRIORITY 限制行为，这些行为可以删除在内存中堆积的消息。
- 清除来自相应目标的所有消息（请参见第 159 页的“清除目标”）。
- 限制可以在内存中保留消息的时间：您可以重写成方客户机以便对每个消息都设置一个有效期值。您可以通过设置 `imqOverrideJMSEExpiration` 和 `imqJMSEExpiration` 连接工厂属性（请参见第 168 页上的表 7-3）来覆盖共享一个连接的所有生成方的这些设置。
- 队列中可以使用消息的使用方太少。

如果消息可以传送到活动使用方太少，队列目标可能会随着消息的堆积而变得堆满了待办事项。只要有下列任何原因，都会发生此情况：

- 目标的活动使用方太少。
- 使用方客户机建立连接失败。
- 活动使用方没有使用与队列中的消息匹配的选择器。

要确认这是否就是问题的起因，请执行下列操作：

要确定使用方不可用的原因，请检查目标上活动使用方的数量：

```
imqcmd metrics dst -n destName -t q -m con
```

要解决此问题，请执行下列操作：

您可以根据使用方不可用的原因采取下列任意操作：

- 通过启动其他使用方客户机来为队列创建更多的活动使用方。
- 调整 `imq.consumerFlowLimit` 代理属性以优化对多个使用方的队列传送（请参见第 257 页的“多使用方队列性能”）。
- 指定队列的消息限制以及限制行为属性（请参见第 154 页上的表 6-10）。例如，您可以指定 `REMOVE_OLDEST` 和 `REMOVE_LOW_PRIORITY` 限制行为，这些行为可以删除在内存中堆积的消息。
- 清除来自相应目标的所有消息（请参见第 159 页的“清除目标”）。
- 限制消息可以在内存中保留的时间：您可以重写生成方客户机以对每个消息都设置一个有效期值，也可以通过设置 `imqOverrideJMSEExpiration` 和 `imqJMSEExpiration` 连接工厂属性（请参见第 168 页上的表 7-3）来覆盖共享一个连接的所有生成方的这些设置。
- 消息使用方的处理速度太慢，跟不上消息生成方的速度。

在这种情况下，主题的订阅者或队列的接收者使用消息的速度要比生成方发送消息的速度慢。会有一个或多个目标因为这种不平衡而堆满了消息。

要确认这是否就是问题的起因，请执行下列操作：

检查消息流入和流出代理的速率：

```
imqcmd metrics bkr -m rts
```

然后检查每个单独目标的流速：

```
imqcmd metrics bkr -t destType -n destName -m rts
```

要解决此问题，请执行下列操作：

- 优化使用方客户机代码。
- 对于队列目标，增大活动使用方的数目（请参见第 257 页的“多使用方队列性能”）。
- 客户机的确认处理过程减慢了消息的使用。

影响对客户机确认过程的处理有两个因素：

- 在处理客户机确认的过程中会消耗大量的代理资源。作为结果，如果使用方客户机一直阻塞到代理对客户机确认进行确认时为止，则对于这样的确认模式，消息的使用会变慢。

- JMS 有效负荷消息和 Message Queue 控制消息（例如客户机确认）共享同一连接。结果，控制消息可能会被 JMS 有效负荷消息阻挡，从而使消息的使用变慢。

要确认这是否就是问题的起因，请执行下列操作：

检查与数据包流相关的消息流。如果每秒数据包数与消息数的不成比例，则客户机确认可能有问题。

另外请检查客户机是否收到 "JMSException [C4000]: Packet acknowledge failed" 消息。

要解决此问题，请执行下列操作：

- 修改客户机使用的确认模式，例如切换到 DUPS_OK_ACKNOWLEDGEMENT 或 CLIENT_ACKNOWLEDGEMENT。
- 如果使用 CLIENT_ACKNOWLEDGEMENT 或事务会话，则将更多数量的消息组合到一个确认中。
- 调整使用方和连接流控制参数（请参见第 258 页的“客户机运行时消息流调整”）。
- 代理无法适应生成的消息。

在这种情况下，消息流入代理的速度比代理可以将它们路由并发送到使用方的速度要快。代理的迟缓可能由下列任一或全部内容的限制所导致：CPU、网络套接字读 / 写操作、磁盘读 / 写操作、内存分页、持久性存储库或 JVM 内存限制。

要确认这是否就是问题的起因，请执行下列操作：

检查有无其他原因导致此问题。

要解决此问题，请执行下列操作：

- 升级计算机或数据存储的速度。
- 使用代理群集将负荷分布到多个代理实例之间。

问题：消息服务器吞吐量呈间歇性

症状：

- 消息的吞吐量不定期地下降，然后又恢复正常性能。

可能原因：

- 代理的内存资源非常低。

由于目标和代理的限制设置得不正确，代理采取了越来越严格的措施以防止内存过载，这样就导致代理变得非常迟缓，直到消息的堆积得到清除为止。

要确认这是否就是问题的起因，请执行下列操作：

在代理日志中检查内存小的情况 ([B1089]: In low memory condition, broker is attempting to free up resources)，该情况后面会接有一个描述新内存状态和已用内存总量的条目。

另外请检查 JVM 堆中的可用内存：

```
imqcmd metrics bkr -m cxn
```

当总 JVM 内存接近 JVM 内存最大值时，可用内存就会很小。

要解决此问题，请执行下列操作：

- 调整 JVM（请参见第 252 页的“Java 虚拟机调整”）。
- 增大系统交换空间。
- 正在发生 JVM 内存回收（垃圾收集）。

内存回收会定期清扫整个系统，以释放内存。发生此操作时，所有的线程都会阻塞。要释放的内存量以及 JVM 堆的大小越大，因内存回收而导致的延迟就越长。

要确认这是否就是问题的起因，请执行下列操作：

监视计算机上的 CPU 使用。发生内存回收时，它会产生显著下降。

另外，使用下列命令行选项启动代理：

```
-vmargs -verbose:gc
```

其标准输出指明发生内存回收的时间。

要解决此问题，请执行下列操作：

在多个 CPU 的计算机中，将内存回收设置为并行发生：

```
-XX:+UseParallelGC=true
```

- JVM 正在使用实时编译器以提高性能。

要确认这是否就是问题的起因，请执行下列操作：

检查有无其他原因导致此问题。

要解决此问题，请执行下列操作：

让系统运行一段时间，性能应该会有所改善。

问题：消息无法到达使用方

症状：

- 使用方未收到生成方发送的消息。

可能原因：

- 限制行为导致在代理上删除了消息。

如果目标内存中的消息数目或消息字节数达到了配置限制，代理将尝试节约内存资源。当达到限制时，代理将采取下列三个可配置的行为，从而导致消息丢失：

- REMOVE_OLDEST: 删除最旧的消息
- REMOVE_LOW_PRIORITY: 删除根据消息时间排列时优先级最低的消息
- REJECT_NEWEST: 拒绝新消息（发出一个有关被拒绝的持久性消息的异常）。

如果代理内存中的消息数量或消息字节数达到配置的限制，代理将尝试通过拒绝最新的消息来节约内存资源。

要确认这是否就是问题的起因，请执行下列操作：

在代理日志中检查下面的条目：WARNING [B2011]: Storing of JMS message from...failed。此条目后面接有另一个阐明限制已超出的条目。但是不会有显示对消息的删除操作的条目。

要解决此问题，请执行下列操作：

更改限制或更改行为。

- 消息超时值过期

代理将删除超时值已过期的消息。如果目标上完全堆满了消息，有效期值过短的消息将被删除。

要确认这是否就是问题的起因，请执行下列操作：

在代理日志文件中检查如下条目：Expiring Messages: Expired *n* messages。

要解决此问题，请执行下列操作：

使用覆盖

- 不同计算机的系统时钟不同步。

如果时钟之间不同步，则代理对消息有效期的计算可能会错误，从而导致消息超过它们的截止时间而被删除。

要确认这是否就是问题的起因，请执行下列操作：

检查所有计算机上的时钟。

要解决此问题，请执行下列操作：

同步时钟（请参见第 301 页的“系统时钟设置”）。

- 使用方客户机未能启动在某个连接上的消息传送。

除非客户机代码建立了连接，并在该连接上启动了消息传送，否则消息将无法传送。

要确认这是否就是问题的起因，请执行下列操作：

检查客户机代码是否能建立连接并启动消息传送。

要解决此问题，请执行下列操作：

重写客户机代码，以建立连接并启动消息传送。

调整配置以提高性能

系统调整

以下各节讲述您可以对操作系统、JVM 和通信协议所作的调整。

Solaris 调整：CPU 使用、分页 / 交换 / 磁盘 I/O

有关对操作系统的调整，请参见系统文档。

Java 虚拟机调整

默认情况下，代理使用大小为 192MB 的 JVM 堆。通常，这对于较大的消息负荷来说太小，应该增大。

当代理快要耗尽 Java 对象使用的 JVM 堆空间时，它将使用各种技术（如流控制和消息交换）来释放内存。在极端情况下，代理甚至关闭客户机连接以释放内存并减少消息内流。所以最好将最大 JVM 堆空间设置得足够大，以避免这种情况。

但是，与系统的物理内存相比，如果最大 Java 堆空间设置过大，代理将继续增大 Java 堆空间，直至整个系统耗尽内存。这会导致性能的降低、不可预计的代理崩溃和 / 或影响系统中运行的其他应用程序和服务的行为。通常，需要有足够的物理内存以便操作系统和其他应用程序在计算机上运行。

总的说来，好的方法是：估算正常和峰值系统内存容量，并配置 Java 堆大小，使其足以提供良好性能，但同时不应过大，以免引起系统内存问题。

要更改代理的最小和最大堆大小，请在启动代理时使用 `-vmargs` 命令行选项。例如：

```
/usr/bin/imqbrokerd -vmargs "-Xms256m -Xmx1024m"
```

此命令会将启动 Java 堆大小设置为 256MB，将最大 Java 堆大小设置为 1GB。

- 在 Solaris 上，如果通过 `/etc/rc`（即 `/etc/init.d/imq`）启动代理，请在 `/etc/imq/imqbrokerd.conf` 文件中指定代理的命令行参数。有关详细信息，请参见该文件中的注释。
- 在 Windows 上，如果将代理作为 Window 服务启动，请使用 `imqsvcadmin install` 命令的 `-vmargs` 选项指定 JVM 参数。请参见第 298 页的“[服务管理器实用程序 \(imqsvcadmin\)](#)”。

在任何情况下，请通过检查代理的日志文件或通过使用 `imqcmd metrics bkr -m cxn` 命令验证设置。

调整传输协议

选择了符合应用程序需要的协议后，基于该协议进行其他调整可以提高性能。

协议的性能可以使用下面三个代理属性进行修改：

- `imq.protocol protocol_type nodelay`
- `imq.protocol protocol_type inbufsz`
- `imq.protocol protocol_type outbufsz`

对于 TCP 和 SSL 协议，这些属性会影响客户机和代理之间的消息传送速度。对于 HTTP 和 HTTPS 协议，这些属性会影响 Message Queue 隧道 servlet（在 Web 服务器上运行）和代理之间的消息传送速度。对于 HTTP/HTTPS 协议，还有其他可以影响性能的属性（请参见第 255 页的“[HTTP/HTTPS 调整](#)”）。

协议调整属性将在以下各节中讲述。

nodelay

`nodelay` 属性影响给定协议的 Nagle 算法（TCP/IP 上的 TCP_NODELAY 套接字级选项的值）。Nagle 算法用于提高使用慢速连接（例如广域网 (WAN)）的系统上的 TCP 性能。

如果使用了此算法，TCP 将通过把多个数据捆绑为较大的数据包来尝试防止将多个小块数据发送到远程系统。如果写入套接字中的数据没有填满需要的缓冲区大小，协议将延迟发送数据包，直到缓冲区被填满，或者已经过了特定的延迟时间为止。填满了缓冲区或者发生了超时后，数据包将发送。

对于大多数消息传送应用程序，如果数据包发送过程中没有延迟（Nagle 算法未启用），则性能是最佳的。这是因为客户机和代理之间的大多数交互都是请求 / 响应交互：客户机向代理发送数据包，并等待响应。例如，典型的交互包括：

- 创建连接
- 创建生成方或使用方
- 发送持久性消息（代理确认收到消息）
- 在 `AUTO_ACKNOWLEDGE` 或 `CLIENT_ACKNOWLEDGE` 会话中发送客户机确认（代理确认对客户机确认的处理）

对于这些交互，大多数数据包都比缓冲区大小要小。这意味着如果使用 Nagle 算法，代理会在向使用方发送响应之前延迟几毫秒。

但是，在连接较慢以及不需要代理响应的情况下，Nagle 算法可以提高性能。例如，当客户机发送非持久性消息或者当客户机确认未被代理确认（`DUPS_OK_ACKNOWLEDGE` 会话）时，就属于这样的情况。

inbufsz/outbufsz

`inbufsz` 属性用于在读取来自套接字的数据的输入流上设置缓冲区大小。同样，`outbufsz` 用于设置代理用来将数据写入套接字的输出流的缓冲区大小。

通常，这两个参数都应该设置为比收发的平均数据包要稍大的值。一个很好的经验是将这些属性值设为平均数据包的大小再加上 1k（舍入为最接近的 k 值）。

例如，如果代理正在接收主体大小为 1k 的数据包，则该数据包的总体大小（消息主体 + 标题 + 属性）约为 1200 字节。大小为 2k（2048 字节）的 `inbufsz` 可以提供合理的性能。

增大 `inbufsz` 或 `outbufsz`（使其大于该值）可以稍微提高性能，但这样会增加每个连接所需的内存。

图 9-6 显示对大小为 1K 的数据包更改 `inbufsz` 的结果。

图 9-7 更改大小为 1K（1024 字节）的数据包的 `inbufsz` 的结果

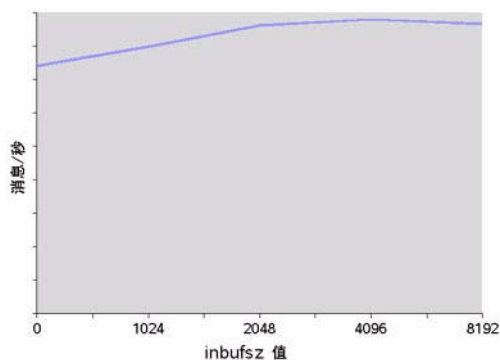
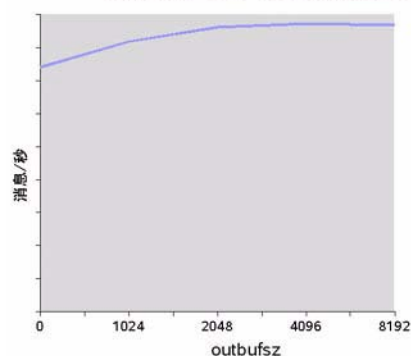


图 9-8 显示对大小为 1K 的数据包更改 `outbufsz` 的结果。

图 9-8 更改大小为 1K（1024 字节）的数据包的 `outbufsz` 的结果



HTTP/HTTPS 调整

除了前面两节讨论的一般属性之外，HTTP/HTTPS 的性能还受到客户机向作为 Message Queue 隧道 servlet 宿主的 Web 服务器发出 HTTP 请求的速度的限制。

Web 服务器可能需要优化，以处理单个套接字上的多个请求。在 JDK 1.4 版及更高版本中，对 Web 服务器的 HTTP 连接会一直保持（对 Web 服务器的套接字保持打开），以使 Web 服务器在处理多个 HTTP 请求时使用的资源最小化。如果使用 JDK 1.4 版的客户机应用程序的性能比运行早期 JDK 版本的同一应用程序要低，则可能需要调整 Web 服务器的保持配置参数以提高性能。

除了这样的 Web 服务器调整之外，您还可以调整客户机轮询 Web 服务器的频率。HTTP 是一种基于请求的协议。这意味着使用基于 HTTP 的协议的客户机需要定期地检查 Web 服务器，以查看是否有消息在等待。imq.httpjms.http.pullPeriod 代理属性（以及相应的 imq.httpsjms.https.pullPeriod 属性）可指定 Message Queue 客户机运行时轮询 Web 服务器的频率。

如果 pullPeriod 值为 -1（默认值），客户机运行时将在前一个请求返回后立即轮询服务器，从而将各个客户机的性能最大化。结果，每个客户机连接都会在 Web 服务器中独占一个请求线程，这样可能会耗费 Web 服务器资源。

如果 pullPeriod 值为正数，客户机运行时将定期向 Web 服务器发送请求，以查看是否有挂起的数据。在这种情况下，客户机不会独占 Web 服务器中的请求线程。因此，如果有大量的客户机在使用 Web 服务器，您可以通过将 pullPeriod 设为正值来节约 Web 服务器资源。

调整基于文件的持久性存储库

有关调整基于文件的持久性存储库的信息，请参见第 58 页的“内置的持久性”。

代理调整

以下各节讲述为了提高性能而可以对代理属性作出的调整。

内存管理：增大代理在负荷下的稳定性

内存管理可以在逐个目标的级别上配置，也可以在系统范围级别（集中地对于所有目标）上配置。

使用目标限制

有关目标限制的信息，请参见第 152 页的“管理目标”。

使用系统范围限制

如果消息生成方的数目超过消息使用方的数目，则消息可能在代理中堆积。尽管代理确实包含限制生成方和在内存过小的情况下将消息交换出活动内存的机制（请参见第 55 页的“管理内存资源和消息流”），最好还是对代理可以保持的消息数和消息字节总数进行严格限制。

可以通过设置 imq.system.max_count 和 imq.system.max_size 代理属性来控制这些限制。有关设置代理属性的信息，请参见第 118 页的“编辑实例配置文件”或第 125 页的“imqbrokerd 选项概述”。

例如


```
img.system.max_count=5000
```

上面定义的值表示代理最多仅能保留 5000 条未传送 / 未确认的消息。如果发送了其他消息，它们将被代理拒绝。如果消息是持久性的，当生成方尝试发送该消息时，会收到一个异常。如果消息是非持久性的，代理将自行废弃该消息。

若要让非持久性消息像持久性消息那样返回异常，请对客户机使用的连接工厂对象设置下列属性：

```
imgAckOnProduce = true
```

上面的设置会降低向代理发送非持久性消息的性能（客户机会等待回复后才发送下一条消息），但通常这是可以接受的，因为内流到代理的消息通常不是系统瓶颈。

如果在发送消息的过程中返回了异常，客户机应该暂停一会，然后再次重试发送。

多使用方队列性能

多队列使用方处理队列目标中消息的效率取决于可配置的队列目标属性，即活动使用方数目 (`maxNumActiveConsumers`) 和一次性可以传送到使用方的最大消息数 (`consumerFlowLimit`)。第 154 页上的表 6-10 介绍了这些属性。

要达到优化的消息吞吐量，必须有足够数量的活动使用方以适应队列的消息生成方的速率，并且队列中的消息必须以最大化使用速率的方式路由和传送给活动使用方。第 69 页的“多个使用方的队列传送”中介绍了在多个使用方之间平衡消息传送的一般机制。

如果消息在队列中堆积，这可能是因为没有足够的活动使用方来处理消息负荷。也可能是每批传送给使用方的消息太多，导致消息在使用方堆积。例如，如果每批的大小 (`consumerFlowLimit`) 太大，某个使用方就可能会收到一个队列中的所有消息，而其他活动使用方则一个也没收到。如果使用方速度特别快，这也不会成为问题。

但是如果使用方相对较慢，而您又希望将消息均匀地分布给它们，则需要将每批的大小减小。每批的大小越小，将消息传送到使用方所需的开销就越多。但是对于较慢的使用方，使用较小的批大小通常却能获得网络性能的提升。

客户机运行时消息流调整

本节讨论影响性能的流控制行为（请参见第 219 页的“客户机运行时配置”）。这些行为可配置为连接工厂被管理对象的属性。有关设置连接工厂属性的信息，请参见第 7 章“管理受管理对象”。

消息流测量

客户机收发的消息（JMS 消息）以及 Message Queue 控制消息通过同一客户机 - 代理连接传递。如果控制消息（例如代理确认）的传送被 JMS 消息阻挡，则会发生延迟。为防止此类拥塞，Message Queue 会测量通过连接的 JMS 消息流。

JMS 消息是分批的（由 `imqConnectionFactoryCount` 属性指定），因此只会传送集合编号；传送完一批后，JMS 消息的传送会挂起，而挂起的控制消息会传送。当另一批 JMS 消息传送后，接着又传送排队等候的控制消息，这样循环往复。

如果客户机正在执行需要代理作出大量响应的操作，`imqConnectionFactoryCount` 的值应该保持较低；例如，当客户机正在使用 `CLIENT_ACKNOWLEDGE` 或 `AUTO_ACKNOWLEDGE` 模式、持久性消息、事务、队列浏览器时，或者当客户机正在添加或删除使用方时。从另一方面说，如果客户机仅在使用 `DUPS_OK_ACKNOWLEDGE` 模式的连接上有简单的使用方，则可以增大 `imqConnectionFactoryCount` 而不会降低性能。

消息流限制

在遇到本地资源（例如内存）限制之前，存在 Message Queue 客户机运行时可以处理的 JMS 消息数限制。如果达到了此限制，性能将受影响。因此，Message Queue 可以让您限制每个使用方（或每个连接）能够通过连接传送和能够在客户机运行时中缓冲以等待使用的消息数。

基于使用方的限制

当传送到客户机运行时的 JMS 消息数超过任意使用方的 `imqConsumerFlowLimit` 值时，将停止传送该使用方的消息。仅当该使用方的未使用消息数下降到低于 `imqConsumerFlowThreshold` 设置的值时才会恢复。

下例说明了如何使用这些限制：以主题使用方的默认设置为例

```
imqConsumerFlowLimit=1000  
imqConsumerFlowThreshold=50
```

创建使用方后，代理将向此使用方发送第一批 1000 条消息（前提是有这么多），中间不会暂停。发送 1000 条消息后，代理将停止传送，除非客户机运行时要求更多消息。客户机运行将保留这些消息，直到应用程序处理它们为止。在要求代理发送下一批消息之前，客户机运行时会允许应用程序使用至少 50%

(`imqConsumerFlowThreshold`) 的消息缓冲区容量（即 500 条消息）。

在同等情况下，如果阈值为 10%，则客户机运行时会等待应用程序使用至少 900 条消息，然后才会要求发送下一批消息。

下一批消息的大小按如下计算：

$$\text{imqConsumerFlowLimit} - (\text{current number of pending msgs in buffer})$$

因此，如果 `imqConsumerFlowThreshold` 为 50%，则下一批的大小会在 500 和 1000 之间波动，这取决于应用程序处理消息的速度。

如果 `imqConsumerFlowThreshold` 设置得过高（接近 100%），代理就会发送较小的分批消息，这样会降低消息的吞吐量。如果该值过低（接近 0%），则客户机可以在代理传送下一组消息之前处理完剩余的缓冲消息，从而导致消息吞吐量的降低。通常，除非您有特别的性能或可靠性考虑，否则不需要更改

`imqConsumerFlowThreshold` 属性的默认值。

基于使用方的流控制（特别是 `imqConsumerFlowLimit`）是管理客户机运行时中内存的最好方法。通常，根据客户机应用程序的不同，您应该知道在任意连接上需要支持的使用方数、消息的大小以及可用于客户机运行时的内存总量。

基于连接的限制

但是在某些客户机应用程序中，使用方数量不是确定的，这取决于最终用户所作的选择。在这些情况下，您仍可以使用连接级流限制来管理内存。

连接级流控制可以限制针对一个连接上的所有使用方而缓冲的总消息数。如果此数字超过了 `imqConnectionFlowLimit`，则通过该连接进行的消息传送将停止，直到该总数降到连接限制以下为止（只有在将 `imqConnectionFlowLimitEnabled` 属性设置为 `true` 时，`imqConnectionFlowLimit` 才会启用）。

在会话中排队等候的消息数是使用该会话的消息使用方数量以及每个使用方的消息负荷的函数。如果客户机在生成或使用消息时表现出延迟，您通常可以通过下列操作提高性能：重新设计应用程序，以便在更大数量的会话之间分布消息生成方和使用方，或者在更大数量的连接之间分布会话。

调整配置以提高性能

Message Queue 数据的位置

Sun Java System Message Queue 使用了多种数据，每种数据所存储的具体位置因操作系统而异，如以下各节中所示。在下面的各表中，*instanceName* 标识了数据关联的代理实例的名称。

Solaris

表 A-1 中显示了 Message Queue 数据在 Solaris 平台上的位置。

注意	Sun Java System 应用程序服务器附带的 Message Queue 在 Solaris 平台上的数据位置显示在第 263 页的表 A-3 中。
----	--

表 A-1 Message Queue 数据在 Solaris 平台上的位置

数据类型	在 Solaris 平台上的位置
代理实例配置属性	/var/imq/instances/ <i>instanceName</i> /props/ config.properties
代理配置文件模板	/usr/share/lib/imq/props/broker/
持久性存储库（消息、 目标、长期订阅、事务）	/var/imq/instances/ <i>instanceName</i> /fs350/ 或支持 JDBC 的数据存储
代理实例日志文件目录（默认位 置）	/var/imq/instances/ <i>instanceName</i> /log/
受管理对象 （对象存储）	您选择的本地目录 或 LDAP 服务器

表 A-1 Message Queue 数据在 Solaris 平台上的位置 (续)

数据类型	在 Solaris 平台上的位置
安全性: 用户信息库	<code>/var/imq/instances/instanceName/etc/passwd</code> 或 LDAP 服务器
安全性: 访问控制文件 (默认位置)	<code>/var/imq/instances/instanceName/etc/accesscontrol.properties</code>
安全性: 密码文件目录 (默认位置)	<code>/var/imq/instances/instanceName/etc/</code>
安全性: 密码文件示例	<code>/etc/imq/passfile.sample</code>
安全性: 代理的密钥存储文件位置	<code>/etc/imq/</code>
JavaDoc API 文档	<code>/usr/share/javadoc/imq/index.html</code>
示例应用程序和配置	<code>/usr/demo/imq/</code>
Java 归档 (.jar) 文件、Web 归档 (.war) 文件和资源适配器归档 (.rar) 文件	<code>/usr/share/lib/</code>

Linux

表 A-2 显示了 Message Queue 数据在 Linux 平台上的位置。

表 A-2 Message Queue 数据在 Linux 平台上的位置

数据类型	在 Windows 平台上的位置
代理实例配置属性	<code>/var/opt/imq/instances/instanceName/props/config.properties</code>
代理配置文件模板	<code>/opt/imq/lib/props/broker/</code>
持久性存储库 (消息、目标、长期订阅、事务)	<code>/var/opt/imq/instances/instanceName/fs350/</code> 或支持 JDBC 的数据存储
代理实例日志文件目录 (默认位置)	<code>/var/opt/imq/instances/instanceName/log/</code>
受管理对象 (对象存储)	您选择的本地目录 或 LDAP 服务器
安全性: 用户信息库	<code>/var/opt/imq/instances/instanceName/etc/passwd</code> 或 LDAP 服务器

表 A-2 Message Queue 数据在 Linux 平台上的位置 (续)

数据类型	在 Windows 平台上的位置
安全性: 访问控制文件 (默认位置)	/var/opt/imq/instances/ <i>instanceName</i> /etc/accesscontrol.properties
安全性: 密码文件目录 (默认位置)	/var/opt/imq/instances/ <i>instanceName</i> /etc/
安全性: 密码文件示例	/etc/opt/imq/passfile.sample
安全性: 代理的密钥存储文件位置	/etc/opt/imq/
JavaDoc API 文档	/opt/imq/javadoc/index.html
示例应用程序和配置	/opt/imq/demo/
Java 归档 (.jar) 文件、Web 归档 (.war) 文件和资源适配器归档 (.rar) 文件	/opt/imq/lib/

Windows

表 A-3 显示了 Message Queue 数据在 Windows 平台上的位置，以及在 Sun Java System 应用程序服务器附带的某些 Message Queue 安装上的位置。有关详细信息，请参见 第 26 页的表 3 以及 IMQ_HOME 和 IMQ_VARHOME 的定义。

表 A-3 Message Queue 数据在 Windows 平台上的位置

数据类型	在 Windows 平台上的位置
代理实例配置属性	IMQ_VARHOME\instances\ <i>instanceName</i> \props\config.properties
代理配置文件模板	IMQ_HOME\lib\props\broker\
持久性存储库 (消息、目标、长期订阅、事务)	IMQ_VARHOME\instances\ <i>instanceName</i> \fs350\ 或支持 JDBC 的数据存储
代理实例日志文件目录 (默认位置)	IMQ_VARHOME\instances\ <i>instanceName</i> \log\
受管理对象 (对象存储)	您选择的本地目录 或 LDAP 服务器
安全性: 用户信息库	IMQ_VARHOME\instances\ <i>instanceName</i> \etc\passwd 或 LDAP 服务器

表 A-3 Message Queue 数据在 Windows 平台上的位置 (续)

数据类型	在 Windows 平台上的位置
安全性: 访问控制文件 (默认)	IMQ_VARHOME\instances\instanceName\ etc\accesscontrol.properties
安全性: 密码文件目录 (默认位置)	IMQ_HOME\etc\
安全性: 密码文件示例	IMQ_HOME\etc\passfile.sample
安全性: 代理的密钥存储文件位置	IMQ_HOME\etc\
JavaDoc API 文档	IMQ_HOME\javadoc\index.html
示例应用程序和配置	IMQ_HOME\demo\
Java 归档 (.jar) 文件、Web 归档 (.war) 文件和资源适配器归档 (.rar) 文件	IMQ_HOME\lib\

设置插入的持久性

本附录说明如何设置代理，使之使用插入的持久性访问支持 JDBC 的数据存储。

简介

Message Queue 代理包含一个持久性管理器组件，该组件用于管理持久性信息的写入和检索（请参见第 57 页的“持久性管理器”）。默认情况下，持久性管理器配置为访问基于文件的内置数据存储，但是可以重新对其进行配置，以插入任何可通过 JDBC 兼容的驱动程序进行访问的数据存储。

要配置代理以使用插入的持久性，需要在代理实例配置文件中设置一些 JDBC 相关的属性。还需要创建相应的数据库架构以执行 Message Queue 持久性操作。

Message Queue 提供的数据库管理器实用程序 (mqdbmgr) 可以使用 JDBC 驱动程序和代理配置属性创建和管理插入的数据库。

本附录中列出的步骤以 Java 2 Platform, Enterprise Edition (J2EE) SDK 附带的 PointBase DBMS 为例进行说明。1.4 版本可以从 java.sun.com 下载。该示例使用 PointBase 的嵌入式版本（而不是客户机 / 服务器版本）。这些步骤以 PointBase 示例中的路径名和属性名为例进行说明。它们标有“示例：”字样。

可以在附录 A “Message Queue 数据的位置”中显示的示例位置找到 Oracle 和 PointBase 的示例配置。此外，PointBase 嵌入式版本、PointBase 服务器版本、Oracle 和 Cloudscape 的示例作为实例配置文件中的注释值提供。

插入支持 JDBC 的数据存储

插入支持 JDBC 的数据存储只需几个步骤。

► 插入支持 JDBC 的数据存储

1. 在代理的配置文件中设置 JDBC 相关属性。

请参见第 267 页的表 B-1 中列出的属性文档。

2. 在以下路径中放置一个 JDBC 驱动程序 jar 文件的副本或符号链接：

/usr/share/lib/imq/ext/ （在 Solaris 平台上）

/opt/imq/lib/ext/ （在 Linux 平台上）

IMQ_VARHOME\lib\ext （在 Windows 平台上）

副本示例(Solaris):

```
% cp j2eeSDK_install_directory/pointbase/lib/pointbase.jar  
/usr/share/lib/imq/ext
```

符号链接示例(Solaris):

```
% ln -s j2eeSDK_install_directory/lib/pointbase/pointbase.jar  
/usr/share/lib/imq/ext
```

3. 创建 Message Queue 持久性所需的数据库架构。

使用 imqdbmgr create all 命令（对于嵌入式数据库）或者 imqdbmgr create tbl 命令（对于外部数据库）。请参见第 270 页的“数据库管理器实用程序(imqdbmgr)”。

示例:

- a. 转到 imqdbmgr 所在的目录。

cd /usr/bin （在 Solaris 平台上）

cd /opt/imq/bin （在 Linux 平台上）

cd IMQ_HOME/bin （在 Windows 平台上）

- b. 输入 imqdbmgr 命令。

```
imqdbmgr create all
```

注意 如果使用嵌入式数据库，建议在以下目录中创建数据库架构：

```
.../instances/instanceName/dbstore/databatbaseName。
```

如果嵌入式数据库没有设置用户名和密码保护，则可能设置了文件系统权限保护。要确保代理能够对数据库进行读写操作，运行该代理的用户应该是使用 `imqdbmgr` 命令创建该嵌入式数据库的同一个用户（请参见第 270 页的“数据库管理器实用程序 (`imqdbmgr`)”）。

JDBC 相关的代理配置属性

代理的实例配置文件位于一个目录中，该目录用与此配置文件相关联的代理实例的名称 (`instanceName`) 标识（请参见附录 A “Message Queue 数据的位置”）：

```
.../instances/instanceName/props/config.properties
```

如果该文件尚不存在，必须使用 `-name instanceName` 选项启动代理，以便 Message Queue 创建该文件。

表 B-1 列出了插入支持 JDBC 的数据存储时需要设置的配置属性。在使用插入持久性的每个代理实例的实例配置文件 (`config.properties`) 中设置这些属性。

利用实例配置属性可以自定义用于创建 Message Queue 数据库架构的 SQL 代码：有一个可配置的属性可指定用于创建每个数据库表的 SQL 代码。正确指定插入数据库使用的数据类型时需要这些属性。

由于数据库供应商之间存在准确 SQL 语法不兼容问题，请务必检查数据库供应商提供的相应文档并对表 B-1 中的属性进行相应调整。例如，对于 PointBase 数据库，可能需要调整 `IMQMSG35` 表中 `MSG` 列所允许的最大长度（请参见 `imq.persist.jdbc.table.IMQMSG35` 属性）。

表 B-1 包含了将为 PointBase DBMS 示例指定的值。

表 B-1 JDBC 相关属性

属性名称	说明
<code>imq.persist.store</code>	指定基于文件或基于 JDBC 的数据存储。 示例: <code>jdbc</code>

表 B-1 JDBC 相关属性 (续)

属性名称	说明
<code>imq.persist.jdbc.brokerid</code> (可选)	<p>指定附加到数据库表名称中的代理实例标识符。当多个代理实例使用同一个数据库作为持久性数据存储时，标识符可使代理实例保持唯一。（嵌入式数据库只存储一个代理实例的数据，因此通常不需要代理实例标识符。）标识符必须是字母数字字符串，其长度不能超过数据库所允许的最大表名长度减去 12 后的长度。</p> <p>示例: <code>PointBase</code> 嵌入式版本不需要。</p>
<code>imq.persist.jdbc.driver</code>	<p>指定连接到数据库的 JDBC 驱动程序的 Java 类名。</p> <p>示例:</p> <pre>com.pointbase.jdbc.jdbcUniversalDriver</pre>
<code>imq.persist.jdbc.opendburl</code>	<p>指定打开现有数据库连接的数据库 URL。</p> <p>示例:</p> <pre>jdbc:pointbase:embedded:dbName; database.home= ../instances/instanceName/dbstore</pre>
<code>imq.persist.jdbc.createdburl</code> (可选)	<p>指定打开创建数据库连接的数据库 URL。（仅在要使用 <code>imqdbmgr</code> 创建数据库时才指定。）</p> <p>示例:</p> <pre>jdbc:pointbase:embedded:dbName;new, database.home= ../instances/instanceName/dbstore</pre>
<code>imq.persist.jdbc.closedburl</code> (可选)	<p>指定当代理关闭时关闭当前数据库连接的数据库 URL。</p> <p>示例: <code>PointBase</code> 不需要</p>
<code>imq.persist.jdbc.user</code> (可选)	<p>指定打开数据库连接的用户名（如果需要）。出于安全性考虑，可以指定值，而不使用命令行选项：</p> <pre>imqbrokerd -dbuser 和 imqdbmgr -u</pre>
<code>imq.persist.jdbc.needpassword</code> (可选)	<p>指定数据库是否要求输入为访问代理的密码。如果值为 <code>true</code>，则表示需要提供密码。可以使用以下命令行选项指定密码：</p> <pre>imqbrokerd -dbpassword imqdbmgr -p</pre> <p>如果不使用命令行选项或密码文件提供密码（请参见第 204 页的“使用密码文件”），代理将提示输入密码。</p>
<code>imq.persist.jdbc.password</code> (可选)	<p>指定用于打开数据库连接的密码（如果需要）。只能在密码文件中指定该属性（请参见第 204 页的“使用密码文件”）。</p> <p>有多种提供密码的方式。最安全的方式是让代理提示您输入密码。较安全的方式是使用密码文件并对密码文件进行读保护。最不安全的方式是使用以下命令行选项指定密码：</p> <pre>imqbrokerd -dbpassword imqdbmgr -p</pre>

表 B-1 JDBC 相关属性 (续)

属性名称	说明
imq.persist.jdbc.table.IMQSV35	用于创建版本表的 SQL 命令。 示例: <pre>CREATE TABLE \${name} (STOREVERSION INTEGER NOT NULL, BROKERID VARCHAR(100))</pre>
imq.persist.jdbc.table.IMQCCREC35	用于创建配置更改记录表的 SQL 命令。 示例: <pre>CREATE TABLE \${name} (RECORDTIME BIGINT NOT NULL, RECORD BLOB(10k))</pre>
imq.persist.jdbc.table.IMQDEST35	用于创建目标表的 SQL 命令。 示例: <pre>CREATE TABLE \${name} (DID VARCHAR(100) NOT NULL, DEST BLOB(10k), primary key(DID))</pre>
imq.persist.jdbc.table.IMQINT35	用于创建 Interest 表的 SQL 命令。 示例: <pre>CREATE TABLE \${name} (CUID BIGINT NOT NULL, INTEREST BLOB(10k), primary key(CUID))</pre>
imq.persist.jdbc.table.IMQMSG35	用于创建消息表的 SQL 命令。 示例: <pre>CREATE TABLE \${name} (MID VARCHAR(100) NOT NULL, DID VARCHAR(100), MSGSIZE BIGINT, MSG BLOB(1m), primary key(MID))</pre> <p>MSG 列的默认最大长度为 1 Megabyte (1m)。如果希望消息超过该长度，请对长度进行相应的设置。如果已经创建了这些表，则需要重新创建它们以进行更改。</p>
imq.persist.jdbc.table.IMQPROPS35	用于创建属性表的 SQL 命令。 示例: <pre>CREATE TABLE \${name} (PROPNAME VARCHAR(100) NOT NULL, PROPVALUE BLOB(10k), primary key(PROPNAME))</pre>
imq.persist.jdbc.table.IMQILIST35	用于创建 Interest 状态表的 SQL 命令。 示例: <pre>CREATE TABLE \${name} (MID VARCHAR(100) NOT NULL, CUID BIGINT, DID VARCHAR(100), STATE INTEGER, primary key(MID, CUID))</pre>

表 B-1 JDBC 相关属性 (续)

属性名称	说明
imq.persist.jdbc.table.IMQTXN35	用于创建事务表的 SQL 命令。 示例: CREATE TABLE \${name} (TUID BIGINT NOT NULL, STATE INTEGER, TSTATEOBJ BLOB(10K), primary key(TUID))
imq.persist.jdbc.table.IMQTACK35	用于创建事务确认表的 SQL 命令。 示例: CREATE TABLE \${name} (TUID BIGINT NOT NULL, TXNACK BLOB(10k))

所有代理配置属性的值都可以使用 -D 命令行选项进行设置。如果某个数据库要求设置某些特定的数据库属性，也可以在启动代理 (imqbrokerd) 时使用 -D 命令行选项进行设置，或者使用数据库管理器实用程序 (imqdbmgr) 进行设置。

示例:

以 PointBase 嵌入式数据库为例，可以使用 -D 命令行选项定义 PointBase 系统目录，而不是在数据库连接 URL 中指定数据库的绝对路径（如表 B-1 示例中所示）：

```
-Ddatabase.home=IMQ_VARHOME/instances/instanceName/dbstore
```

在这种情况下，可以分别将用于创建和打开数据库的 URL 简单指定为：

```
imq.persist.jdbc.createdburl=jdbc:pointbase:embedded:dbName;new
```

和

```
imq.persist.jdbc.opendburl=jdbc:pointbase:embedded:dbName
```

数据库管理器实用程序 (imqdbmgr)

Message Queue 为设置持久性所需的架构提供了数据库管理器实用程序 (imqdbmgr)。当 Message Queue 数据库表被损坏或者您希望使用不同的数据库作为数据存储时，也可以使用该实用程序删除数据库表。

本节介绍了基本的 imqdbmgr 命令语法，提供了一个子命令列表，并概述了 imqdbmgr 命令选项。

imqdbmgr 命令的语法

imqdbmgr 命令的一般语法如下：

```
imqdbmgr subcommand argument [options]
imqdbmgr -h|-help
imqdbmgr -v|-version
```

请注意，如果指定 `-v` 或 `-h` 选项，将不会执行命令行中指定的子命令。例如，输入以下命令将显示版本信息，而不执行 `create` 子命令。

```
imqdbmgr create all -v
```

imqdbmgr 子命令

表 B-2 列出了数据库管理器实用程序 (imqdbmgr) 包含的子命令：

表 B-2 imqdbmgr 子命令

子命令 和参数	说明
create all	创建一个新的数据库和 Message Queue 持久性存储库架构。此命令用于嵌入式数据库系统，使用时需指定 <code>imq.persist.jdbc.createdburl</code> 属性。
create tbl	在现有数据库系统中创建 Message Queue 持久性存储库架构。此命令用于外部数据库系统。
delete tbl	删除当前持久性存储库数据库中的现有 Message Queue 数据库表。
delete oldtbl	删除早期版本的持久性存储库数据库中的所有 Message Queue 数据库表。将持久性存储库自动迁移至 Message Queue 当前版本后使用。
recreate tbl	删除当前持久性存储库数据库中的现有 Message Queue 数据库表，然后重新创建 Message Queue 持久性存储库架构。
reset lck	对锁进行重置，以便其他进程可以使用该持久性存储库数据库。

imqdbmgr 命令选项概述

表 B-3 列出了 imqdbmgr 命令的选项。

表 B-3 imqdbmgr 选项	
选项	说明
-Dproperty=value	将指定的属性设置为指定值。
-b instanceName	指定代理实例名称并使用对应的实例配置文件。
-h	显示使用帮助。不在命令行执行其他命令。
-p password	指定数据库密码。
-u name	指定数据库用户名。
-v	显示版本信息。不在命令行执行其他命令。

HTTP/HTTPS 支持（企业版）

Message Queue Enterprise Edition（请参见第 33 页的“产品版本”）包含对 HTTP 和 HTTPS 连接的支持。（HTTPS 传输连接使用安全套接字层 (SSL)，比 HTTP 连接更安全。）此支持使客户机应用程序可以使用 HTTP 协议（而不是直接 TCP 连接）与代理进行通信。本附录介绍了实现此支持所使用的体系结构，并说明了如何设置客户机，使之能够使用基于 HTTP 的连接进行 Message Queue 消息传送。

注意	HTTP/HTTPS 支持可用于 Java 客户机，但不能用于 C 客户机。
-----------	--

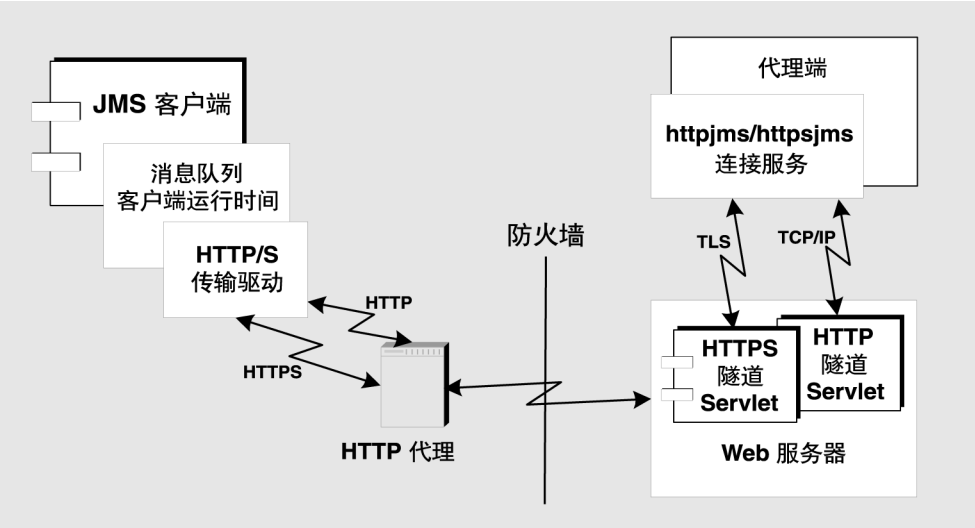
HTTP/HTTPS 支持体系结构

可以通过 HTTP/HTTPS 连接进行 Message Queue 消息传送。因为 HTTP/HTTPS 连接通常可以穿过防火墙，因此可以通过防火墙将客户机应用程序与代理隔开。

第 274 页的图 C-1 列出了提供 HTTP/HTTPS 支持所需的主要组件。

- 在客户端，HTTP 或 HTTPS 传输驱动程序将 Message Queue 消息封装到 HTTP 请求中，并确保将这些请求以正确的顺序发送给 Web 服务器。
- 必要时，客户机可以使用 HTTP 代理服务器与代理进行通信。可以在启动客户机时使用命令行选项指定代理地址。有关详细信息，请参见第 278 页的“使用 HTTP 代理”。
- 将 JMS 消息转发给代理之前，需要将 HTTP 或 HTTPS 隧道 Servlet（均随 Message Queue 一起提供）装入 Web 服务器，并使用它们从客户机 HTTP 请求中提取 JMS 消息。HTTP/HTTPS 隧道 Servlet 还将代理消息发送回客户机，以响应客户机发出的 HTTP 请求。可以使用一个 HTTP/HTTPS 隧道 Servlet 访问多个代理。

图 C-1 HTTP/HTTPS 支持体系结构



- 在代理端，httpjms 或 httpsjms 连接服务对来自相应隧道 Servlet 的消息进行展开和分离。
- 如果 Web 服务器失败并重新启动，所有连接都将恢复且不会对客户机产生影响。如果代理失败并重新启动，将抛出异常，而客户机必须重新建立连接。如果 Web 服务器和代理同时失败（这种情况不太可能发生）而代理没有重新启动，Web 服务器将恢复客户机连接并等待代理恢复连接而不会通知客户机。要避免这种情况，请务必重新启动代理。

从图 C-1 可以看出，HTTP 和 HTTPS 支持所需的体系结构非常相似。主要区别在于，HTTPS（httpsjms 连接服务）中的隧道 Servlet 与客户机应用程序和代理之间的连接都是安全的。

与代理之间的安全连接是通过支持 SSL 的隧道 Servlet（Message Queue 的 HTTPS 隧道 Servlet）提供的，该 Servlet 会向请求连接的任何代理发送自签名证书。代理使用该证书建立与 HTTPS 隧道 Servlet 的加密连接。建立此连接后，客户机应用程序和 Web 服务器可以协商建立客户机应用程序和隧道 Servlet 之间的安全连接。

实现 HTTP 支持

以下各节介绍了实现 HTTP 支持所需执行的步骤。

► 实现 HTTP 支持

1. 在 Web 服务器上部署 HTTP 隧道 Servlet。
2. 配置代理的 httpjms 连接服务并启动代理。
3. 配置 HTTP 连接。

步骤 1：在 Web 服务器上部署 HTTP 隧道 Servlet

在 Web 服务器上部署 HTTP 隧道 Servlet 的常见方法有两种：

- 将其部署为支持 Servlet 2.1 或更早版本的 Web 服务器 Jar 文件
- 将其部署为支持 Servlet 2.2 或更高版本的 Web 服务器 Web 归档 (WAR) 文件

部署为 Jar 文件

部署 Message Queue 隧道 Servlet 分为三个步骤，首先要使主机 Web 服务器可以访问相应的 Jar 文件，然后要对该 Web 服务器进行配置，使之在启动时装入该 Servlet，最后需要指定该 Servlet URL 的上下文根分区。

隧道 Servlet Jar 文件 (imqervlet.jar) 包含 HTTP 隧道 Servlet 所需的所有类，该文件所在的目录因操作系统而异（请参见[附录 A “Message Queue 数据的位置”](#)）。

可以使用任何支持 Servlet 2.x 的 Web 服务器装入此 Servlet。Servlet 类名为：

```
com.sun.messaging.jmq.transport.  
httpunnel.servlet.HttpTunnelServlet
```

Web 服务器必须能够访问 imqervlet.jar 文件。如果计划在不同的主机上运行 Web 服务器和代理，应该在 Web 服务器能够访问的位置放置 imqervlet.jar 文件的一个副本。

还需要配置 Web 服务器以在启动时装入此 Servlet，可能需要指定该 Servlet URL 的上下文根分区（请参见[第 279 页的“示例 1：在 Sun Java System Web Server 上部署 HTTP 隧道 Servlet”](#)）。

同时建议您禁用 Web 服务器的访问日志功能以提高性能。

部署为 Web 归档文件

要将 HTTP 隧道 Servlet 部署为 WAR 文件，需要使用 Web 服务器提供的部署机制。HTTP 隧道 Servlet WAR 文件 (imqhttp.war) 所在的目录中包含 .jar、.war 和 .rar 文件，具体情况取决于您的操作系统（请参见[附录 A “Message Queue 数据的位置”](#)）。

WAR 文件包含一个部署描述符，该描述符包含 Web 服务器装入和运行 Servlet 所需的基本配置信息。根据 Web 服务器的不同，可能还需要指定该 Servlet URL 的上下文根分区（请参见[第 282 页的“示例 2: 在 Sun Java System Application Server 7.0 上部署 HTTP 隧道 Servlet”](#)）。

步骤 2：配置 httpjms 连接服务

默认情况下，未为代理激活 HTTP 支持，因此您需要重新配置代理以激活 httpjms 连接服务。重新配置后，可以按照[第 123 页的“启动代理”](#)中介绍的步骤启动代理。

► 激活 httpjms 连接服务

- 1. 打开代理的实例配置文件。

实例配置文件存储在一个目录中，该目录用与此配置文件相关联的代理实例的名称 (instanceName) 标识（请参见[附录 A “Message Queue 数据的位置”](#)）：

```
.../instances/instanceName/props/config.properties
```

- 2. 将 httpjms 值添加到 imq.service.activelist 属性中：

```
imq.service.activelist=jms,admin,httpjms
```

启动时，代理将在其主机上查找运行的 Web 服务器和 HTTP 隧道 Servlet。但是要访问远程隧道 Servlet，需要重新配置 servletHost 和 servletPort 连接服务属性。

还可以重新配置 pullPeriod 属性以提高性能。[第 276 页的表 C-1](#) 详细介绍了 httpjms 连接服务配置属性。

表 C-1 httpjms 连接服务属性

属性名称	说明
imq.httpjms.http.servletHost	必要时可以更改此值，以指定运行 HTTP 隧道 Servlet 的主机的名称（主机名或 IP 地址）。（可以是远程主机或本地主机上的特定主机名。）默认值：localhost

表 C-1 httpjms 连接服务属性 （续）

属性名称	说明
imq.httpjms.http.servletPort	需要更改此值，以指定代理用于访问 HTTP 隧道 Servlet 的端口号。（如果更改了 Web 服务器上的默认端口，也必须对此属性做相应的更改。）默认值：7675
imq.httpjms.http.pullPeriod	指定客户机运行时从代理提取消息的 HTTP 请求的时间间隔（以秒为单位）。（注意，该属性在代理上设置并传播到客户机运行时。）如果值为零或为负数，客户机将始终使一个 HTTP 请求处于待处理状态，这样可以随时尽快地提取消息。如果客户机的数量过多，会大量消耗 Web 服务器资源，可能导致服务器停止响应。在这种情况下，应将 pullPeriod 属性设置为正秒数。此属性设置客户机 HTTP 传输驱动程序在发出下一个提取请求之前等待的时间。将此属性设置为正值将以牺牲客户机响应时间为代价来保持 Web 服务器资源的可用性。默认值：-1
imq.httpjms.http.connectionTimeout	指定客户机运行时在抛出异常前等待 HTTP 隧道 Servlet 响应的时间（以秒为单位）。（注意，该属性在代理上设置并传播到客户机运行时。）该属性还指定代理在与 HTTP 隧道 Servlet 进行了通信后等待断开的时间。在这种情况下可能会超时，因为代理和隧道 Servlet 不知道客户机正在访问的 HTTP Servlet 是否已经异常终止。默认值：60

步骤 3：配置 HTTP 连接

客户机应用程序必须使用正确配置的连接工厂管理对象，建立与代理之间的 HTTP 连接。本节介绍 HTTP 连接配置问题。

配置连接工厂

要实现 HTTP 支持，需要将连接工厂的 imqAddressList 属性设置为 HTTP 隧道 Servlet URL。HTTP 隧道 Servlet URL 的一般语法如下：

`http://hostName:port/contextRoot/tunnel`

其中，`hostName:port` 是作为 HTTP 隧道 Servlet 宿主的 Web 服务器的名称和端口，`contextRoot` 是在该 Web 服务器上部署隧道 Servlet 时设置的路径。

通常有关连接工厂属性，特别是有关 imqAddressList 属性的详细信息，请参见《*Message Queue Java Client Developer's Guide*》。

可以按照以下方法之一设置连接工厂属性：

- 在创建连接工厂管理对象（请参见第 176 页的“添加连接工厂”）的 imqobjmgr 命令中使用 -o 选项，或在使用管理控制台 (imqadmin) 创建连接工厂管理对象时设置属性。

- 在启动客户机的命令中使用 `-D` 选项（请参见《*Message Queue Java Client Developer's Guide*》）。
- 通过编程方式在客户机代码中创建连接工厂（请参见《*Message Queue Java Client Developer's Guide*》）之后，使用 API 调用设置其属性。

使用一个 Servlet 访问多个代理

即使正在运行多个代理，也无需配置多个 Web 服务器和多个 Servlet 实例。可以在并行运行的多个代理之间共享一个 Web 服务器和一个 HTTP 隧道 Servlet 实例。如果多个代理实例在共享一个隧道 Servlet，必须配置 `imqAddressList` 连接工厂属性，如下所示：

```
http://hostName:port/contextRoot/tunnel?ServerName=bkrHostName:instanceName
```

其中，`bkrHostName` 是代理实例主机名，`instanceName` 是您希望客户机访问的特定代理实例的名称。

要查看是否输入了正确的 `bkrHostName` 和 `bkrHostName` 字符串，可以通过浏览器访问 Servlet URL，生成 HTTP 隧道 Servlet 的状态报告。状态报告将列出 Servlet 正在访问的所有代理：

```
HTTP tunnel servlet ready.
Servlet Start Time :Thu May 30 01:08:18 PDT 2002
Accepting TCP connections from brokers on port : 7675
Total available brokers = 2
Broker List :
  jpgserv:broker2
  cochin:broker1
```

使用 HTTP 代理

如果使用 HTTP 代理访问 HTTP 隧道 Servlet：

- 将 `http.proxyHost` 系统属性设置为代理服务器主机名。
- 将 `http.proxyPort` 系统属性设置为代理服务器端口号。

可以通过在启动客户机应用程序的命令中使用 `-D` 选项来设置这些属性。

示例 1：在 Sun Java System Web Server 上部署 HTTP 隧道 Servlet

本节说明如何在 Sun Java System Web Server 上将 HTTP 隧道 Servlet 部署为 Jar 文件和 WAR 文件。使用的方法取决于 Sun Java System Web Server 的版本：如果 Sun ONE Web Server 不支持 Servlet 2.2 或更高版本，则不能处理 WAR 文件部署。

部署为 Jar 文件

下面的说明适用于使用基于浏览器的管理 GUI 在 Sun Java System Web Server 6.1 上部署 HTTP 隧道 Servlet 的情况。此过程包含以下通用步骤：

1. 添加 Servlet
2. 配置 Servlet 虚拟路径
3. 装入 Servlet
4. 禁用 Servlet 访问日志

以下小节对这些步骤进行了说明。通过使用 Web 浏览器访问 Servlet URL 可以验证是否成功部署了 HTTP 隧道 Servlet。它应该显示状态信息。

添加 Servlet

► 添加隧道 Servlet

1. 选择 "Servlets" 选项卡。
2. 选择 "Configure Servlet Attribute"。
3. 在 "Servlet Name" 字段中指定隧道 Servlet 的名称。
4. 将 "Servlet Code"（类名）字段设置为以下值：

```
com.sun.messaging.jmq.transport.  
httpunnel.servlet.HttpTunnelServlet
```

5. 在 "Servlet Classpath" 字段中输入 imqservlet.jar 的完整路径。例如：

/usr/share/lib/imq/imqservlet.jar（在 Solaris 操作系统上）

/opt/imq/lib/imqservlet.jar（在 Linux 操作系统上）

IMQ_HOME/lib/imqservlet.jar（在 Windows 操作系统上）

6. 在 "Servlet args" 字段中输入可选参数，如表 C-2 所示：

表 C-2 用于部署 HTTP 隧道 Servlet Jar 文件的 Servlet 参数

参数	默认值	参考
servletHost	所有主机	请参见第 276 页的表 C-1
servletPort	7675	请参见第 276 页的表 C-1

如果使用两个参数，请使用逗号分隔：

`servletPort=portNumber, servletHost=...`

servletHost 和 servletPort 参数仅应用于 Web 服务器和代理之间的通信，仅当默认值存在问题时才需要进行设置。但是，在这种情况下，还必须相应地设置代理配置属性（请参见第 276 页的表 C-1），例如：

`img.httpjms.http.servletPort`

配置 Servlet 虚拟路径 (Servlet URL)

► **配置隧道 Servlet 的虚拟路径 (Servlet URL)**

- 1. 选择 "Servlets" 选项卡。
- 2. 选择 “Configure Servlet Virtual Path Translation”。
- 3. 设置 “Virtual Path” 字段。

“Virtual Path” 是隧道 Servlet URL 的 `/contextRoot/tunnel` 分区：

`http://hostName:port/contextRoot/tunnel`

例如，如果将 `contextRoot` 设置为 `img`，则 “Virtual Path” 字段将为：

`/img/tunnel`

- 4. 设置 “Servlet Name” 字段的值，使之与第 279 页的 “添加 Servlet” 中的步骤 3 中设置的值相同。

装入 Servlet

► **在 Web 服务器启动时装入隧道 Servlet**

- 1. 选择 "Servlets" 选项卡。
- 2. 选择 “Configure Global Attributes”。
- 3. 在 “Startup Servlets” 字段中，输入与第 279 页的 “添加 Servlet” 中的步骤 3 中的值相同的 Servlet 名称值。

禁用服务器访问日志

不是必须禁用服务器访问日志，但禁用服务器访问日志可以获得更佳的性能。

► 禁用服务器访问日志

1. 选择 “Status” 选项卡。
2. 选择 “Log Preferences Page”。
3. 使用 “Log” 客户机访问控制禁用日志。

部署为 WAR 文件

下面的说明适用于在 Sun Java System Web Server 6.0 Service 上的部署。

通过使用 Web 浏览器访问 Servlet URL 可以验证是否成功部署了 HTTP 隧道 Servlet。它应该显示状态信息。

► 将 HTTP 隧道 Servlet 部署为 WAR 文件

1. 在基于浏览器的管理 GUI 中，选择 “Virtual Server Class” 选项卡并选择 “Manage Classes”。
2. 选择相应的虚拟服务器类名（例如，defaultClass）并单击 “Manage” 按钮。
3. 选择 “Manage Virtual Servers”。
4. 选择相应的虚拟服务器名称并单击 “Manage” 按钮。
5. 选择 “Web Applications” 选项卡。
6. 单击 “Deploy Web Application”。
7. 为 “WAR File On” 和 “WAR File Path” 字段选择相应的值，以指向 imqhttp.war 文件，该文件所在的目录因操作系统而异（请参见附录 A “Message Queue 数据的位置”）。
8. 在 “Application URI” 字段中输入路径。

“Application URI” 字段值为隧道 Servlet URL 的 `/contextRoot` 分区：

```
http://hostName:port/contextRoot/tunnel
```

例如，如果将 `contextRoot` 设置为 `imq`，则 “Application URI” 字段将为：

```
/imq
```

9. 输入要在其中部署 Servlet 的安装目录路径（通常位于 Sun Java System Web Server 的安装根目录下）。
10. 单击 “OK”。

11. 重新启动 Web 服务器实例。

Servlet 现在即被部署到以下位置：

```
http://hostName:port/contextRoot/tunnel
```

客户机现在即可使用此 URL 连接至使用 HTTP 连接的消息服务。

示例 2：在 Sun Java System Application Server 7.0 上部署 HTTP 隧道 Servlet

本节说明如何在 Sun Java System Application Server 7.0 上将 HTTP 隧道 Servlet 部署为 WAR 文件。

需要执行两个步骤：

- 使用 Application Server 7.0 部署工具部署 HTTP 隧道 Servlet
- 修改应用程序服务器实例的 `server.policy` 文件

使用部署工具

► 在 Application Server 7.0 环境下部署 HTTP 隧道 Servlet

1. 在基于 Web 的管理 GUI 中，选择

“App Server” > “Instances” > “server1” > “Applications” > “Web Applications”。

2. 单击 “Deploy” 按钮。

3. 在 “File Path:” 文本字段中，输入 HTTP 隧道 Servlet WAR 文件 (`imqhttp.war`) 的位置。

`imqhttp.war` 文件的位置因操作系统而异（请参见[附录 A “Message Queue 数据的位置”](#)）

4. 单击 “OK”。

5. 在下一个屏幕上，为 “Context Root” 文本字段设置值。

“Context Root” 字段值为隧道 Servlet URL 的 `/contextRoot` 分区：

```
http://hostName:port/contextRoot/tunnel
```

例如，可以将 “Context Root” 字段设置为：

```
/imq
```

6. 单击 “OK”。

下一个屏幕显示隧道 Servlet 已成功部署，默认情况下处于启用状态，在这种情况下，它位于以下位置：

```
/var/opt/SUNWappserver7/domains/domain1/server1/applications/  
j2ee-modules/imqhttp_1
```

Servlet 现在即被部署到以下位置：

```
http://hostName:port/contextRoot/tunnel
```

客户机现在即可使用此 URL 连接至使用 HTTP 连接的消息服务。

修改 server.policy 文件

Application Server 7.0 实施了一套默认的安全策略，除非经过修改，否则它们将阻止 HTTP 隧道 Servlet 与 Message Queue 代理进行连接。

每个应用程序服务器实例都有一个包含其安全策略或规则的文件。例如，Solaris 上 server1 实例的此文件的位置为：

```
/var/opt/SUNWappserver7/domains/domain1/server1/config/  
server.policy
```

要使隧道 Servlet 与 Message Queue 代理进行连接，此文件中还必须有另外一个条目。

► 修改 Application Server 的 server.policy 文件

1. 打开 server.policy 文件。
2. 添加以下条目：

```
grant codeBase  
"file:/var/opt/SUNWappserver7/domains/domain1/server1/  
applications/j2ee-modules/imqhttp_1/-"  
{  
    permission java.net.SocketPermission "*",  
        "connect,accept,resolve";  
};
```

实现 HTTPS 支持

以下各节介绍了实现 HTTPS 支持所需执行的步骤。所需操作与第 275 页的“实现 HTTP 支持”中的操作类似，只是增加了生成和访问 SSL 证书的步骤。

► 实现 HTTPS 支持

1. 为 HTTPS 隧道 Servlet 生成自签名证书。
2. 在 Web 服务器上部署 HTTPS 隧道 Servlet。
3. 配置代理的 `httpsjms` 连接服务并启动代理。
4. 配置 HTTPS 连接。

下文详细介绍了每个步骤。

步骤 1：为 HTTPS 隧道 Servlet 生成自签名证书

Message Queue 的 SSL 支持用于保护所传输数据的安全，假定客户机正在与已知且可信任的服务器进行通信。因此，仅使用自签名的服务器证书实现 SSL。在 `httpsjms` 连接服务体系结构中，HTTPS 隧道 Servlet 充当代理和应用程序客户机的服务器。

运行 `imqkeytool` 实用程序，为隧道 Servlet 生成自签名证书。在命令提示符下输入以下内容：

```
imqkeytool -servlet keystore_location
```

命令行实用程序会提示您提供所需的信息。（在 Unix 操作系统上，可能需要以超级用户（root 用户）身份运行 `imqkeytool` 命令，以获得创建键值存储所需的权限。）

首先，`imqkeytool` 提示您输入键值存储密码，然后提示您输入组织信息，最后提示您进行确认。收到确认后，它将在生成密钥对时暂停。然后它要求您输入密码以锁定特定密钥对（密钥密码），此时请按回车键响应此提示：这将使密钥密码与键值存储密码相同。

注意	请记住您输入的密码，稍后需要将此密码提供给隧道 Servlet 使其能够打开键值存储。
-----------	---

运行 `imqkeytool` 命令以运行 JDK `keytool` 实用程序，生成自签名证书并将其放置在 Message Queue 的键值存储文件中，该文件的位置由 `keystore_location` 参数指定。（键值存储格式与 JDK1.2 `keytool` 支持的键值存储格式相同。）

注意 HTTPS 隧道 Servlet 必须可以访问该键值存储。确保将 *keystore_location* 中已生成的键值存储移动 / 复制到 HTTPS 隧道 Servlet 可以访问的位置（请参见第 285 页的“[步骤 2：在 Web 服务器上部署 HTTPS 隧道 Servlet](#)”）。

步骤 2：在 Web 服务器上部署 HTTPS 隧道 Servlet

在 Web 服务器上部署 HTTPS 隧道 Servlet 的常见方法有两种：

- 将其部署为支持 Servlet 2.1 或更早版本的 Web 服务器 Jar 文件
- 将其部署为支持 Servlet 2.2 或更高版本的 Web 服务器 Web 归档 (WAR) 文件

在任一种情况下，都应确保激活 Web 服务器的加密功能，确保客户机与代理之间的端到端通信是安全的。

部署为 Jar 文件

部署 Message Queue 隧道 Servlet 分为三个步骤，首先要使主机 Web 服务器可以访问相应的 Jar 文件，然后要对该 Web 服务器进行配置，使之在启动时装入该 Servlet，最后需要指定该 Servlet URL 的上下文根分区。

隧道 Servlet Jar 文件 (imqservlet.jar) 包含 HTTPS 隧道 Servlet 所需的所有类，该文件所在的目录因操作系统而异（请参见[附录 A “Message Queue 数据的位置”](#)）。

可以使用任何支持 Servlet 2.x 的 Web 服务器装入此 Servlet。Servlet 类名为：

```
com.sun.messaging.jmq.transport.
httpunnel.servlet.HttpsTunnelServlet
```

Web 服务器必须能够访问 imqservlet.jar 文件。如果计划在不同的主机上运行 Web 服务器和代理，应该在 Web 服务器能够访问的位置放置 imqservlet.jar 文件的一个副本。

还需要配置 Web 服务器以在启动时装入此 Servlet，可能需要指定该 Servlet URL 的上下文根分区（请参见第 290 页的“[示例 3：在 Sun Java System Web Server 上部署 HTTPS 隧道 Servlet](#)”）。

确保 JSSE Jar 文件位于 Web 服务器上运行 Servlets 的类路径中。请参阅 Web 服务器文档，了解如何执行此操作。

配置 Web 服务器有一个重要的方面，即指定 HTTPS 隧道 Servlet 用来与代理建立安全连接所需的自签名证书的位置和密码。必须将您在第 284 页的“步骤 1：为 HTTPS 隧道 Servlet 生成自签名证书”中创建的键值存储放置在 HTTPS 隧道 Servlet 可以访问的位置。

同时建议您禁用 Web 服务器的访问日志功能以提高性能。

部署为 Web 归档文件

要将 HTTPS 隧道 Servlet 部署为 WAR 文件，需要使用 Web 服务器提供的部署机制。HTTPS 隧道 Servlet WAR 文件 (imghttps.war) 所在的目录因操作系统而异（请参见附录 A “Message Queue 数据的位置”）。

WAR 文件包含一个部署描述符，该描述符包含 Web 服务器装入和运行 Servlet 所需的基本配置信息。根据 Web 服务器的不同，可能还需要指定该 Servlet URL 的上下文根分区（请参见第 294 页的“示例 4：在 Sun Java System Application Server 7.0 上部署 HTTPS 隧道 Servlet”）。

但是，imghttps.war 文件的部署描述符无法知道您放置隧道 Servlet 所需的键值存储的位置（请参见第 284 页的“步骤 1：为 HTTPS 隧道 Servlet 生成自签名证书”）。因此您需要在部署 imghttps.war 文件之前编辑隧道 Servlet 部署描述符（一个 XML 文件），以指定键值存储的位置。

步骤 3：配置 httpsjms 连接服务

默认情况下，未为代理激活 HTTPS 支持，因此您需要重新配置代理以激活 httpsjms 连接服务。重新配置后，可以按照第 123 页的“启动代理”中介绍的步骤启动代理。

► 激活 httpsjms 连接服务

1. 打开代理的实例配置文件。

实例配置文件存储在一个目录中，该目录用与此配置文件相关联的代理实例的名称 (*instanceName*) 标识（请参见附录 A “Message Queue 数据的位置”）：

```
.../instances/instanceName/props/config.properties
```

2. 将 httpsjms 值添加到 img.service.activelist 属性中：

```
img.service.activelist=jms,admin,httpsjms
```

启动时，代理将在其主机上查找运行的 Web 服务器和 HTTPS 隧道 Servlet。但是要访问远程隧道 Servlet，需要重新配置 servletHost 和 servletPort 连接服务属性。

还可以重新配置 `pullPeriod` 属性以提高性能。表 C-3 详细介绍了 `httpsjms` 连接服务配置属性。

表 C-3 `httpsjms` 连接服务属性

属性名称	说明
<code>imq.httpsjms.https.servletHost</code>	必要时可以更改此值，以指定运行 HTTPS 隧道 Servlet 的主机的名称（主机名或 IP 地址）。（可以是远程主机或本地主机上的特定主机名。）默认值： <code>localhost</code>
<code>imq.httpsjms.https.servletPort</code>	需要更改此值，以指定代理用于访问 HTTPS 隧道 Servlet 的端口号。（如果更改了 Web 服务器上的默认端口，也必须对此属性做相应的更改。）默认值： <code>7674</code>
<code>imq.httpsjms.https.pullPeriod</code>	指定每一台客户机创建从代理提取消息的 HTTP 请求的时间间隔（以秒为单位）。（注意，该属性在代理上设置并传播到客户机运行时。）如果值为零或为负数，客户机将始终使一个 HTTP 请求处于待处理状态，这样可以随时尽快地提取消息。如果客户机的数量过多，会大量消耗 Web 服务器资源，可能导致服务器停止响应。在这种情况下，应将 <code>pullPeriod</code> 属性设置为正秒数。此属性设置客户机 HTTP 传输驱动程序在发出下一个提取请求之前等待的时间。将此属性设置为正值将以牺牲客户机响应时间为代价来保持 Web 服务器资源的可用性。默认值： <code>-1</code>
<code>imq.httpsjms.https.connectionTimeout</code>	指定客户机运行时在抛出异常前等待 HTTPS 隧道 Servlet 响应的 时间（以秒为单位）（注意，该属性在代理上设置并传播到客户机运行时。）该属性还指定代理在与 HTTPS 隧道 Servlet 进行了通信后等待断开的时间。在这种情况下可能会超时，因为代理和隧道 Servlet 不知道客户机正在访问的 HTTPS Servlet 是否已经异常终止。默认值： <code>60</code>

步骤 4：配置 HTTPS 连接

客户机应用程序必须使用正确配置的连接工厂管理对象，建立与代理之间的 HTTPS 连接。

但是，客户机还必须能够访问 Java 安全套接扩展 (JSSE) 提供的 SSL 库，并且必须具有一个根证书。SSL 库是随 JDK 1.4 一起提供的。如果您使用的是早期版本的 JDK，请参见“配置 JSSE”，；否则，请转至“输入根证书”。

解决上述问题后，可以继续配置 HTTPS 连接。

配置 JSSE

► 配置 JSSE

1. 将 JSSE Jar 文件复制到 JRE_HOME/lib/ext 目录中。

包括 jsse.jar、jnet.jar 和 jcert.jar

2. 通过添加以下内容来静态添加 JSSE 安全服务供应商：

```
security.provider.n=com.sun.net.ssl.internal.ssl.Provider
```

将这些内容添加到 JRE_HOME/lib/security/java.security 文件中，（其中 *n* 是安全服务供应商软件包的下一个可用优先编号）。

3. 如果使用的不是 JDK1.4，则需要在启动客户机应用程序的命令中使用 -D 选项，以设置下列 JSSE 属性：

```
java.protocol.handler.pkgs=com.sun.net.ssl.internal.www.protocol
```

输入根证书

如果默认情况下签署您的 Web 服务器证书的 CA 的根用户证书不在信任数据库中，或者使用的是专用 Web 服务器证书，则必须将该证书添加到信任数据库中。如果出现上述情况，请执行以下操作；否则，请转至“[配置连接工厂](#)”。

如果证书保存在 *cert_file* 中，而您的键值存储为 *trust_store_file*，请运行以下命令：

```
JRE_HOME/bin/keytool -import -trustcacerts
-alias alias_for_certificate -file cert_file
-keystore trust_store_file
```

显示以下问题时请选择 YES: Trust this certificate?

还需要在启动客户机应用程序的命令中使用 -D 选项，以指定下列 JSSE 属性：

```
javax.net.ssl.trustStore=trust_store_file
javax.net.ssl.trustStorePassword=trust_store_passwd
```

配置连接工厂

要实现 HTTPS 支持，需要将连接工厂的 *imqAddressList* 属性设置为 HTTPS 隧道 Servlet URL。HTTPS 隧道 Servlet URL 的一般语法如下：

```
https://hostName:port/contextRoot/tunnel
```

其中，*hostName:port* 作为 HTTPS 隧道 Servlet 宿主的 Web 服务器的名称和端口，*contextRoot* 是在该 Web 服务器上部署隧道 Servlet 时设置的路径。

通常有关连接工厂属性，特别是有关 `imqAddressList` 属性的详细信息，请参见《*Message Queue Java Client Developer's Guide*》。

可以按照以下方法之一设置连接工厂属性：

- 在创建连接工厂管理对象（请参见第 176 页的“添加连接工厂”）的 `imqobjmgr` 命令中使用 `-o` 选项，或在使用管理控制台 (`imqadmin`) 创建连接工厂管理对象时设置属性。
- 在启动客户机应用程序的命令中使用 `-D` 选项（请参见《*Message Queue Java Client Developer's Guide*》）。
- 通过编程方式在客户机应用程序代码中创建连接工厂（请参见《*Message Queue Java Client Developer's Guide*》）之后，使用 API 调用设置其属性。

使用一个 Servlet 访问多个代理

即使正在运行多个代理，也无需配置多个 Web 服务器和多个 Servlet 实例。可以在并行运行的多个代理之间共享一个 Web 服务器和一个 HTTPS 隧道 Servlet 实例。如果多个代理实例共享一个隧道 Servlet，则必须配置 `imqAddressList` 连接工厂属性，如下所示：

```
https://hostName:port/contextRoot/tunnel?ServerName=bkrHostName:instanceName
```

其中，`bkrHostName` 是代理实例主机名，`instanceName` 是您希望客户机访问的特定代理实例的名称。

要查看是否输入了正确的 `bkrHostName` 和 `instanceName` 字符串，可以通过浏览器访问 Servlet URL，生成 HTTPS 隧道 Servlet 的状态报告。状态报告将列出 Servlet 正在访问的所有代理：

```
HTTPS tunnel servlet ready.
Servlet Start Time : Thu May 30 01:08:18 PDT 2002
Accepting secured connections from brokers on port : 7674
Total available brokers = 2
Broker List :
  jpgserv:broker2
  cochín:broker1
```

使用 HTTP 代理

如果使用 HTTP 代理访问 HTTPS 隧道 Servlet：

- 将 `http.proxyHost` 系统属性设置为代理服务器主机名。

- 将 `http.proxyPort` 系统属性设置为代理服务器端口号。

可以通过在启动客户机应用程序的命令中使用 `-D` 选项来设置这些属性。

示例 3：在 Sun Java System Web Server 上部署 HTTPS 隧道 Servlet

本节说明如何在 Sun Java System Web Server 上将 HTTPS 隧道 Servlet 部署为 Jar 文件和 WAR 文件。使用的方法取决于 Sun Java System Web Server 的版本：如果 Sun ONE Web Server 不支持 Servlet 2.2 或更高版本，则不能处理 WAR 文件部署。

部署为 Jar 文件

以下说明适用于使用基于浏览器的管理 GUI 在 Sun Java System Web Server 6.1 上部署 HTTP 隧道 Servlet 的情况。此过程包含以下通用步骤：

1. 添加 Servlet
2. 配置 Servlet 虚拟路径
3. 装入 Servlet
4. 禁用 Servlet 访问日志

在以下小节中，对这些步骤进行了说明。通过使用 Web 浏览器访问 Servlet URL 可以验证是否成功部署了 HTTPS 隧道 Servlet。它应该显示状态信息。

添加 Servlet

► 添加隧道 Servlet

1. 选择 “Servlets” 选项卡。
2. 选择 “Configure Servlet Attribute”。
3. 在 “Servlet Name” 字段中指定隧道 Servlet 的名称。
4. 将 “Servlet Code”（类名）字段设置为以下值：

```
com.sun.messaging.jmq.transport.  
httptunnel.servlet.HttpsTunnelServlet
```

5. 在 “Servlet Classpath” 字段中输入 `imqservlet.jar` 的完整路径。例如：
- `/usr/share/lib/imq/imqservlet.jar` （在 Solaris 操作系统上）
 - `/opt/imq/lib/imqservlet.jar` （在 Linux 操作系统上）
 - `IMQ_HOME/lib/imqservlet.jar` （在 Windows 操作系统上）
6. 在 “Servlet args” 字段中输入必需参数和可选参数，如表 C-4 所示。

表 C-4 用于部署 HTTPS 隧道 Servlet Jar 文件的 Servlet 参数

参数	默认值	是否必需?	另请参见
keystoreLocation	无	是	第 199 页的表 8-8
keystorePassword	无	是	第 199 页的表 8-8
servletHost	所有主机	无	第 287 页的表 C-3
servletPort	7674	无	第 287 页的表 C-3

参数之间用逗号分隔，例如：

```
keystoreLocation=keystore_location,keystorePassword=keystore_password,
servletPort=portnumber
```

`servletHost` 和 `servletPort` 参数仅应用于 Web 服务器和代理之间的通信，仅当默认值存在问题时才需要进行设置。但是，在这种情况下，还必须相应地设置代理配置属性（请参见第 287 页的表 C-3），例如：

```
imq.httpsjms.https.servletPort
```

配置 Servlet 虚拟路径 (Servlet URL)

► **配置隧道 Servlet 的虚拟路径 (Servlet URL)**

1. 选择 “Servlets” 选项卡。
2. 选择 “Configure Servlet Virtual Path Translation”。
3. 设置 “Virtual Path” 字段。

“Virtual Path” 是隧道 Servlet URL 的 `/contextRoot/tunnel` 分区：

```
https://hostName:port/contextRoot/tunnel
```

例如，如果将 `contextRoot` 设置为 `imq`，则 “Virtual Path” 字段将为：

```
/imq/tunnel
```

4. 设置 “Servlet Name” 字段的值，使之与第 290 页的 “添加 Servlet” 中的步骤 3 中设置的值相同。

装入 Servlet

► 在 Web 服务器启动时装入隧道 Servlet

1. 选择 "Servlets" 选项卡。
2. 选择 “Configure Global Attributes”。
3. 在 “Startup Servlets” 字段中，输入与第 290 页的 “添加 Servlet” 中的步骤 3 中的值相同的 Servlet 名称值。

禁用服务器访问日志

不是必须禁用服务器访问日志，但禁用服务器访问日志可以获得更佳的性能。

► 禁用服务器访问日志

1. 选择 “Status” 选项卡。
2. 选择 “Log Preferences Page”。
3. 使用 “Log” 客户机访问控制禁用日志。

部署为 WAR 文件

以下说明是指对 Sun Java System Web Server 6.0 Service Pack 2 部署的说明。通过使用 Web 浏览器访问 Servlet URL 可以验证是否成功部署了 HTTPS 隧道 Servlet。它应该显示状态信息。

部署 HTTPS 隧道 Servlet 之前，请确保 JSSE Jar 文件位于 Web 服务器的类路径中。要达到此目的，最简单的方法是将 jsse.jar、jnet.jar 和 jcert.jar 文件复制到 IWS60_TOPDIR/bin/https/jre/lib/ext 中。

另外，部署 HTTPS 隧道 Servlet 之前，必须指定键值存储密码并修改部署描述符，使其指向放置键值存储文件的位置。

► 修改 HTTPS 隧道 Servlet WAR 文件

1. 将 WAR 文件复制到临时目录中。

```
cp /usr/share/lib/imq/imqhttps.war /tmp (在 Solaris 操作系统上)
```

```
cp /opt/imq/lib/imqhttps.war /tmp (在 Linux 操作系统上)
```

```
cp IMQ_HOME/lib/imqhttps.war /tmp (在 Windows 操作系统上)
```

2. 使临时目录成为当前目录。

```
$ cd /tmp
```

3. 提取 WAR 文件的内容。

```
$ jar xvf imghttps.war
```

4. 列出 WAR 文件的部署描述符。

```
$ ls -l WEB-INF/web.xml
```

5. 编辑 web.xml 文件，以提供正确的 keystoreLocation 和 keystorePassword 参数值（如有必要，还包括 servletPort 和 servletHost 参数）。

6. 重新装入 WAR 文件的内容。

```
$ jar uvf imghttps.war WEB-INF/web.xml
```

现在即可使用这个修改后的 imghttps.war 文件部署 HTTPS 隧道 Servlet 了。（如果担心会泄漏键值存储密码，可以使用文件系统权限限制对 imghttps.war 文件的访问。）

► 将 HTTPS 隧道 Servlet 部署为 WAR 文件

1. 在基于浏览器的管理 GUI 中，选择 “Virtual Server Class” 选项卡。单击 “Manage Classes”。
2. 选择相应的虚拟服务器类名（例如，defaultClass）并单击 “Manage” 按钮。
3. 选择 “Manage Virtual Servers”。
4. 选择相应的虚拟服务器名称并单击 “Manage” 按钮。
5. 选择 “Web Applications” 选项卡。
6. 单击 “Deploy Web Application”。
7. 为 “WAR File On” 和 “WAR File Path” 字段选择相应的值，以指向修改后的 imghttps.war 文件（请参见第 292 页的 [“修改 HTTPS 隧道 Servlet WAR 文件”](#)）。
8. 在 “Application URI” 字段中输入路径。

“Application URI” 字段值为隧道 Servlet URL 的 /contextRoot 分区：

```
https://hostName:port/contextRoot/tunnel
```

例如，如果将 contextRoot 设置为 img，则 “Application URI” 字段将为：

```
/img
```

9. 输入要在其中部署 Servlet 的安装目录路径（通常位于 Sun Java System Web Server 的安装根目录下）。
10. 单击“OK”。
11. 重新启动 Web 服务器实例。

Servlet 现在即被部署到以下位置：

```
https://hostName:port/img/tunnel
```

客户机现在即可使用此 URL 连接至使用安全 HTTPS 连接的消息服务。

示例 4：在 Sun Java System Application Server 7.0 上部署 HTTPS 隧道 Servlet

本节说明如何在 Sun Java System Application Server 7.0 上将 HTTPS 隧道 Servlet 部署为 WAR 文件。

需要执行两个步骤：

- 使用 Application Server 7.0 部署工具部署 HTTPS 隧道 Servlet
- 修改应用程序服务器实例的 server.policy 文件

使用部署工具

► 在 Application Server 7.0 环境下部署 HTTPS 隧道 Servlet

1. 在基于 Web 的管理 GUI 中，选择
“App Server” > “Instances” > “server1” > “Applications” > “Web Applications”。
2. 单击“Deploy”按钮。
3. 在“File Path:”文本字段中，输入 HTTPS 隧道 Servlet WAR 文件 (imghttps.war) 的位置。

imghttps.war 文件的位置因操作系统而异（请参见[附录 A “Message Queue 数据的位置”](#)）

4. 单击“OK”。

5. 在下一个屏幕上，为 “Context Root” 文本字段设置值。

“Context Root” 字段值为隧道 Servlet URL 的 `/contextRoot` 分区：

```
https://hostName:port/contextRoot/tunnel
```

例如，可以将 “Context Root” 字段设置为：

```
/img
```

6. 单击 “OK”。

下一个屏幕显示隧道 Servlet 已成功部署，默认情况下处于启用状态，在这种情况下，它位于以下位置：

```
/var/opt/SUNWappserver7/domains/domain1/server1/applications/  
j2ee-modules/imghttps_1
```

Servlet 现在即被部署到以下位置：

```
https://hostName:port/contextRoot/tunnel
```

客户机现在即可使用此 URL 连接至使用 HTTPS 连接的消息服务。

修改 server.policy 文件

Application Server 7.0 实施了一套默认的安全策略，除非经过修改，否则它们将阻止 HTTPS 隧道 Servlet 与 Message Queue 代理进行连接。

每个应用程序服务器实例都有一个包含其安全策略或规则的文件。例如，Solaris 上 server1 实例的此文件的位置为：

```
/var/opt/SUNWappserver7/domains/domain1/server1/config/  
server.policy
```

要使隧道 Servlet 与 Message Queue 代理进行连接，此文件中还必须有另外一个条目。

► 修改 Application Server 的 server.policy 文件

1. 打开 server.policy 文件。
2. 添加以下条目：

```
grant codeBase  
"file:/var/opt/SUNWappserver7/domains/domain1/server1/  
applications/j2ee-modules/imghttps_1/-"
```

```
{  
    permission java.net.SocketPermission "*",  
        檢 onnect,accept,resolve";  
};
```


使用代理作为 Windows 服务

本附录说明如何使用服务管理器 (imgsvcadmin) 实用程序安装、查询和删除作为 Windows 服务运行的代理。

将代理作为 Windows 服务运行

安装 Message Queue 时，可以选择将代理安装为 Windows 服务。安装了 Message Queue 后，还可以使用 imgsvcadmin 将代理安装为 Windows 服务。

将代理安装为 Windows 服务意味着它将在系统启动时启动，并在后台运行直到系统关闭。因此，不使用 imqbrokerd 命令启动代理，除非希望启动其他实例。要将任何启动选项传递给代理，可以使用 imgsvcadmin 命令的 -args 参数（请参见第 299 页的表 D-2）并准确指定您希望 imqbrokerd 命令使用的选项（请参见第 123 页的“启动代理”）。照常使用 imqcmd 命令来控制代理操作。

当作为 Windows 服务运行时，任务管理器将代理列为两个可执行的进程。第一个是本地 Windows 服务包装程序 imqbrokersvc.exe。第二个是实际运行代理的 Java 运行时。

一次只能安装一个代理并作为 Windows 服务运行。

服务管理器实用程序 (imqsvcadmin)

可以使用服务管理器实用程序 (imqsvcadmin) 安装、查询和删除作为 Windows 服务运行的代理。本节介绍了 imqsvcadmin 命令的基本语法，提供了一个子命令列表，概述了 imqsvcadmin 命令选项，并说明了如何使用这些命令执行特定的任务。

imqsvcadmin 命令语法

imqsvcadmin 命令的一般语法如下：

```

imqsvcadmin subcommand [options]

imqsvcadmin -h
```

请注意，如果指定 -v、-h 或 -H 选项，将不会执行命令行中指定的其他子命令。例如，输入以下命令将显示帮助信息，而不是执行 query 子命令。

```

imqsvcadmin query -h
```

imqsvcadmin 子命令

表 D-1 列出了 Message Queue 服务管理器实用程序 (imqsvcadmin) 包含的子命令：

表 D-1 imqsvcadmin 子命令	
子命令	说明
install	安装服务并特定启动选项。
query	显示 imqsvcadmin 命令的启动选项。启动选项包括服务的启动方式（手动或自动）、服务的位置、 Java 运行时的位置以及启动时传递给代理的参数值。
remove	删除服务。

imqsvcadmin 选项概述

表 D-2 列出了 imqsvcadmin 命令的选项。有关如何使用这些选项的论述，请参见基于任务的以下各节。

表 D-2 imqsvcadmin 选项

选项	说明
-h	显示使用帮助。不在命令行执行其他命令。
-javahome <i>path</i>	指定要使用的替代 Java 2 兼容运行时（默认使用系统上的运行时或 Message Queue 附带的运行时）的路径。 示例: <code>imqsvcadmin -install -javahome d:\jdk1.4</code>
-jrehome <i>path</i>	指定替代 Java 2 兼容 JRE 的路径。 示例: <code>imqsvcadmin -install -jrehome d:\jre\1.4</code>
-vmargs <i>arg</i> [[<i>arg</i>]...]	指定传递给正在运行代理服务的 Java VM 的其他参数。（也可以在 Windows 服务控制面板的“启动参数”字段中指定这些参数。） 示例: <code>-vmargs "-Xms16m -Xmx128m"</code>
-args <i>arg</i> [[<i>arg</i>]...]	指定传递给代理服务的其他命令行参数。有关 <code>imqbrokerd</code> 选项的说明，请参见第 123 页的“启动代理”。 （也可以在 Windows 服务控制面板的“启动参数”字段中指定这些参数。）例如， <code>imqsvcadmin -install -args "-passfile d:\imqpassfile"</code>

使用 `-javahome`、`-vmargs` 和 `-args` 选项指定的信息存储在 **Window** 注册表项 `JavaHome`、`JVMArgs` 和 `ServiceArgs` 下，其路径位于

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet
\Services\iMQ_Broker\Parameters
```

删除代理服务

删除代理服务之前，应使用 `imqcmd shutdown bkr` 命令关闭代理。然后使用 `imqsvcadmin remove` 命令删除服务并重新启动计算机。

重新配置代理服务

要重新配置服务，首先要删除服务，然后重新安装并使用 `-args` 参数指定不同的启动选项。

使用替代 Java 运行时

可以使用 `-javahome` 或 `-jrehome` 选项指定替代 Java 运行时的位置。（也可以在 Windows 服务控制面板的“启动参数”字段中指定这些选项。）请注意，“启动参数”字段将反斜杠 (\) 作为转义符进行处理，因此如果要使用它作为路径分隔符，必须键入两次，例如 `-javahome d:\\jdk1.3`。

查询代理服务

要确定代理服务的启动选项，请使用 `imqsvcadmin` 命令的 `-q` 选项。

```
imqsvcadmin -query

Service iMQ_Broker is installed.
Display Name: iMQ_Broker
Start Type: Manual
Binary location: c:\Program Files\Sun Microsystems\
                  Message Queue 3.5\bin\imqbrokersvc
JavaHome: c:\j2sdk1.4.0
Broker Args: -passfile d:\imqpassfile
```

疑难解答

如果试图启动服务时收到错误消息，可以通过执行以下操作查看记录的错误事件。

► 查看记录的服务错误事件

1. 启动事件查看器。
2. 在“日志” > “应用程序”下查找。
3. 选择“查看” > “刷新”，以查看所有的错误事件。

技术说明

本附录包含有关以下主题的简短说明：

- [系统时钟设置](#)
- [OS 定义的文件描述符限制](#)
- [确保持久性数据的安全](#)

系统时钟设置

使用 Message Queue 系统时，在同步系统时钟时应小心，避免将它们设置为早于当前时间。

建议同步

建议您同步所有与 Message Queue 系统交互的主机上的时钟。这在使用消息失效期 (TimeToLive) 时尤为重要。同步主机时钟失败可能导致 TimeToLive 无法按预期方式工作（消息可能无法传送）。您应在启动任何代理之前同步时钟。

Solaris 可以在本地主机上运行 `rdate` 命令以与远程主机达到同步。（必须为超级用户（即 `root` 用户）才能运行此命令。）例如，以下命令将使本地主机（称为 Host 2）与远程主机 Host1 同步：

```
# rdate Host1
```

Linux 命令类似于 Solaris，但必须提供 `-s` 选项：

```
# rdate -s Host1
```

Windows 可以运行带有时间子命令的网络命令将本地主机与远程主机同步。例如，以下命令将本地主机（称为 Host 2）与远程主机 Host1 同步：

```
net time \\Host1 /set
```

避免将系统时钟设置为早于当前时间

应避免在运行 Message Queue 代理的系统上将系统时钟设置为早于当前时间。

Message Queue 使用时间戳来帮助标识诸如事务和长期订阅这样的内部对象。如果系统时钟设置为早于当前时间，理论上有可能生成重复的内部标识符。代理将尝试通过为标识符引入一些随机性以及通过在运行时检测时钟偏差来补偿这种情况，但是如果代理未运行时将系统时钟调整为远远早于当前时间，则会有少量出现标识符重复的可能性。

如果需要在运行代理的系统上将系统时钟设置调整为早于当前时间若干秒钟，建议您在没有事务和长期订阅，或在代理没有运行时执行此操作，然后在代理进行备份之前等候一段时间（时钟偏差的时间）。

不过最理想的方法是在启动任何代理前先同步时钟，然后使用适当的技术确保部署后时钟不会明显出现偏差。

OS 定义的文件描述符限制

在 Solaris 和 Linux 平台上，客户机或代理在其中运行的 shell 对客户机可以使用的文件描述符的数量进行了不严格的限制。在 Message Queue 系统中，客户机创建的每个连接，或代理接受的每个连接都使用其中一种文件描述符。每个具有持久性消息的目的地也使用文件描述符。

因此，连接数由这些因素限制。如果不更改文件描述符限制，Solaris 上代理或客户机运行的连接不能超过 256 个，Linux 上不能超过 1024 个。（连接限制实际上要少一些，因为使用文件描述符可保持持久。）

要更改文件描述符限制，请参见 ulimit 手册页。此限制需在要执行客户机或代理的每个 shell 中更改。

确保持久性数据的安全

代理使用持久性存储库，它包含与其他信息一起临时存储的消息文件。由于这些消息可能包含专用信息，所以建议您保护数据存储以防止未授权的访问。

代理可以使用内置的持久性，也可以使用插入的持久性。

内置持久性存储库

使用内置持久性的代理将持久性数据写入平面文件数据存储，它所在的目录因平台而异（请参见附录 A “[Message Queue 数据的位置](#)”）：

```
.../instances/instanceName/fs350/
```

其中，*instanceName* 为标识代理实例的名称。

第一次启动代理实例时，将创建 *instanceName/filestore/* 目录。保护此目录的过程取决于运行代理的操作系统。

Solaris 和 Linux IMQ_VARHOME/instances/*instanceName*/filestore/ 目录的访问权限取决于启动代理实例的用户的 **umask**。因此，可以通过适当设置 **umask** 来限制获取启动代理实例并读取其持久性文件的权限。或者，管理员（超级用户）可以通过将 IMQ_VARHOME/instances 目录上的权限设为 700 来保护持久性数据。

Windows 可以通过您使用的 Windows 操作系统提供的机制来设置 IMQ_VARHOME/instances/*instanceName*/filestore/ 目录的访问权限。这通常涉及打开目录的属性对话框。

插入的持久性存储库

使用插入的持久性的代理向 JDBC 兼容数据库写入持久性数据。

对于由数据库服务器管理的数据库（例如，Oracle 数据库），建议创建一个用户名和密码来访问 Message Queue 数据库表（名称以 "IMQ" 开头的表）。如果数据库不允许保护单个表，请创建一个仅由 Message Queue 代理使用的专用数据库。请参见数据库供应商提供的文档，以了解如何创建用户名 / 密码访问。

代理打开数据库连接所需的用户名和密码可以作为代理配置属性提供。但是，启动代理时将它们作为命令行选项更为安全（请参见《Message Queue 管理指南》的附录 A “设置插入的持久性”）。

对于代理通过数据库的 JDBC™ 驱动程序直接访问的嵌入式数据库（例如，Cloudscape 数据库），通常通过在要存储持久性数据的目录上设置文件权限（如以上 [“内置持久性存储库”](#)，中所述）来提供安全性。为确保代理和 imqdbmgr 实用程序都可以读写数据库，无论如何，两者均应由同一用户运行。

Message Queue 资源适配器

Message Queue 包括一个 JMS 资源适配器。

资源适配器是在符合 J2EE 连接器体系结构 (J2EECA) 1.5 规范的 J2EE 1.4 兼容应用程序服务器上插入附加功能的标准方式。此体系结构允许任何 J2EE 1.4 兼容应用程序服务器与外部系统以标准方式进行交互。这些外部系统包括各种企业信息系统 (EIS) 和各种消息传送系统，例如 JMS 提供者。

J2EECA 1.5 提供的标准化交互包括连接池、线程池、事务与安全上下文传播，以及各种消息驱动 bean 容器的支持。该规范还包括创建连接工厂和其他受管理对象的标准化方式。

通过将 JMS 资源适配器插入应用程序服务器，即可在应用程序服务器环境中部署和运行的 J2EE 组件之间交换 JMS 消息。这些组件所需的 JMS 连接工厂和目标受管理对象可使用 J2EE 应用程序服务器管理工具进行创建和配置。

但是其他管理操作（如管理消息服务器和物理目标）则不包括在 J2EECA 规范之中，且仅能通过提供者特定工具执行。

Message Queue 资源适配器嵌入在 Sun J2EE 1.4 Application Server 之中。但是，Message Queue 资源适配器尚未经过任何其他 J2EE 1.4 application server 认证。

Message Queue 资源适配器是一个单独的文件 (mqjmsra.rar)，其所在的目录因操作系统而异，如附录 A “Message Queue 数据的位置”。中所示。mqjmsra.rar 文件中包含资源适配器部署描述信息 (ra.xml) 以及必须由应用程序服务器用于使用适配器的 Jar 文件。

您可按照应用程序服务器附带的资源适配器部署和配置说明在任何 J2EE 1.4 兼容应用程序服务器上使用 Message Queue 资源适配器。随着商业 J2EE 1.4 应用程序服务器的上市以及 Message Queue 资源适配器经过了那些应用程序服务器的认证，本附录将在相关部署和配置步骤中提供特定信息。

Message Queue 实现方案可选的 JMS 功能

JMS 规范指出了某些项是可选的 - 每个 JMS 提供者（供应商）可以选择是否实现这些项。Message Queue 产品对这些可选项的处理如下所示：

表 G-1 可选的 JMS 功能

JMS 规范中的章节	说明和 Message Queue 处理
3.4.3 JMSTimestampID	<p>“由于创建消息 ID 比较麻烦，并会使消息容量增大，所以如果提示某些 JMS 提供者应用程序不使用消息 ID，则它们能够优化消息开销。JMS 消息生成方提供了禁用消息 ID 的提示。”</p> <p>Message Queue 实现： 产品不会禁用消息 ID 生成（在 MessageProducer 中调用的所有 setDisableTimestampID() 均将忽略）。所有消息都将包含有效的 TimestampID 值。</p>
3.4.12 覆盖消息标题字段	<p>“JMS 未明确定义管理员如何覆盖这些标题字段值。不要求 JMS 提供者支持此管理选项。”</p> <p>Message Queue 实现： Message Queue 产品支持通过配置连接工厂管理对象以管理方式覆盖消息标题字段的值（请参见第 168 页的表 7-3）。</p>
3.5.9 JMS 定义的属性	<p>“JMS 为 JMS 定义的属性保留 ‘JMSX’ 属性名称前缀。”</p> <p>“除有另行规定，否则对这些属性的支持是可选的。”</p> <p>Message Queue 实现 Message Queue 产品支持由 JMS 1.1 规范定义的 JMSX 属性（请参见第 168 页的表 7-3）。</p>
3.5.10 针对提供者的属性	<p>“JMS 为针对提供者的属性保留 ‘JMS_<vendor_name>’ 属性名称前缀。”</p> <p>Message Queue 实现： 针对提供者的属性的目的在于为支持提供者本地客户机使用 JMS 提供所需的特殊功能。它们不应该用于 JMS 至 JMS 的信息传送。Message Queue 3.5 SP1 不使用针对提供者的属性。</p>

表 G-1 可选的 JMS 功能 *（续）*

JMS 规范中的章节	说明和 Message Queue 处理
4.4.8 分布式事务	<p>“JMS 不要求提供者支持分布式事务。”</p> <p>Message Queue 实现: 此版本的 Message Queue 产品支持分布式事务（请参见第 44 页的“分布式事务”）。</p>
4.4.9 多会话	<p>“对于 PTP < 点对点分布模型 >，JMS 不为相同的队列指定并行 QueueReceiver 的语义；但 JMS 并不禁止提供者支持此功能”。有关详细信息，请参见 JMS 规范的 5.8 节。</p> <p>Message Queue 实现: Message Queue 实现支持对多个用户的队列传送。有关详细信息，请参见第 69 页的“多个使用方的队列传送”。</p>

Message Queue 接口的稳定性

Sun Java System Message Queue 使用的许多接口可能对管理员自动化管理任务很有帮助。表 H-1 按照稳定性（即本产品的后续版本中对其进行更改的可能性）对这些接口进行了分类。第 311 页的表 H-2 介绍了分类方案。

表 H-1 Message Queue 接口的稳定性

接口	分类
imqbrokerd 命令行接口	开发中
imqadmin 命令行接口	不稳定
imqcmd 命令行接口	开发中
imqdbmgr 命令行接口	不稳定
imqkeytool 命令行接口	开发中
imqobjmgr 命令行接口	开发中
imqusermgr 命令行接口	不稳定
imqobjmgr 命令文件	开发中
imqbrokerd 命令	稳定
imqadmin 命令	不稳定
imqcmd 命令	稳定
imqdbmgr 命令	不稳定
imqkeytool 命令	稳定
imqobjmgr 命令	稳定
imqusermgr 命令	不稳定
JMS API (javax.jms)	标准
JAXM API (javax.xml)	标准
C-API	开发中

表 H-1 Message Queue 接口的稳定性 (续)

接口	分类
基于消息监视 API	开发中
受管理对象 API (com.sun.messaging)	开发中
imq.jar 位置和名称	稳定
jms.jar 位置和名称	开发中
imqbroker.jar 位置和名称	专用
imqutil.jar 位置和名称	专用
imqadmin.jar 位置和名称	专用
imqservlet.jar 位置和名称	开发中
imqhttp.war 位置和名称	开发中
imqhttps.war 位置和名称	开发中
imqjmsra.rar 位置和名称	开发中
imqxm.jar 位置和名称	开发中
jaxm-api.jar 位置和名称	开发中
saa-j-api.jar 位置和名称	开发中
saa-j-impl.jar 位置和名称	开发中
activation.jar 位置和名称	开发中
mail.jar 位置和名称	开发中
dom4j.jar 位置和名称	专用
fscontext.jar 位置和名称	不稳定
imqbrokerd、imqadmin、imqcmd、imqdbmgr、imqkeytool、imqobjmgr 和 imqusermgr 的输出	不稳定
代理日志文件的位置和内容格式	不稳定
密码文件	不稳定
accesscontrol.properties	不稳定

表 H-2 接口稳定性分类方案

分类	说明
专用	用户无法直接使用。在任何发行版中都可能对其进行改进或将其删除。
开发中	可供用户使用。可能会在主版本（例如 3.0 和 4.0）或次要版本（例如 3.1 和 3.2）中做不兼容更改。所有更改都将慎重逐步地进行。虽然我们会尽量保证所有更改都是兼容的，但不能保证做到这一点。
稳定	可供用户使用。只在主版本（例如 3.0 和 4.0）中做不兼容更改。
标准	可供用户使用。这些接口根据官方认可的标准进行定义，由标准组织控制。很少对这些接口做不兼容更改。
不稳定	可供用户使用。可能会在主版本（例如 3.0 和 4.0）或次要版本（例如 3.1 和 3.2）中做不兼容更改。用户需注意，在以后的发行版中可能会删除这些接口，也可能以一种不兼容的方式更改这些接口。建议用户不要在不稳定的接口上创建显式相关性。

词汇表

本词汇表提供了使用 Sun Java System Message Queue 时可能遇到的术语和概念的相关信息。

受管理对象 由管理员创建并由一个或多个 JMS 客户机使用的预配置 Message Queue 对象，例如连接工厂或目标。

受管理对象的使用使得 JMS 客户机能够独立于提供者，即它使 JMS 客户机不再专属于提供者。管理员将这些对象置于 JNDI 名称空间，JMS 客户机可以使用 JNDI 查找对它们进行访问。

异步通信 一种通信模式，在这种模式下，消息的发送方不需要等到发送方法返回即可继续其他工作。

授权 消息服务确定用户是否可以访问消息服务资源（如连接服务或目标）的过程。

代理 管理消息路由、传送、持久性、安全性和日志的 Message Queue 实体，它为管理员监视和调节性能和资源使用提供了一个接口。

客户机 与其他使用消息服务的客户机进行交互的应用程序（或软件组件），以交换消息。

客户机标识符 将连接及其对象与代表客户机的 Message Queue 消息服务器所维护的状态关联的标识符。

客户机运行时 请参见 Message Queue 客户机运行时。

群集 两个或多个互连的代理，以串联模式提供消息传送服务。

配置文件 一个或多个文本文件，包含用于配置代理的 Message Queue 设置。配置文件中的属性是特定于某个实例或与某个群集相关的。

连接 1) 到 Message Queue 消息服务器的活动连接。可以是队列连接或主题连接。
2) 使用连接底层 Message Queue 消息服务器生成和使用消息的会话工厂。

连接工厂 客户机创建 Message Queue 消息服务器连接时使用的受管理对象。可以是 QueueConnectionFactory 对象或 TopicConnectionFactory 对象。

使用 消息用户从目标接收消息的过程。

用户 由用于从目标接收消息的会话创建的对象 (MessageConsumer)。在点对点传送模型下，用户为收件人或浏览器 (QueueReceiver 或 QueueBrowser)；在发布 / 订阅传送模型下，用户即为订阅者 (TopicSubscriber)。

数据存储 用于永久存储代理所需的信息（长期订阅、目标数据、持久性消息、审计数据）的数据库。

传送模式 消息传送可靠性的指示符：持久性传送模式，确保消息传送并成功使用一次（且仅一次）；非持久传送模式，确保消息至少传送一次。

传送模型 消息传送的方式：点对点或发布 / 订阅。JMS 中有两个独立的编程域与这两种模型相对应（使用特定的客户机运行时对象、特定的目标类型（队列或主题）），除此之外，还有一个统一的编程域。

传送策略 队列如何在注册了多个消息用户时路由消息的规范。策略包括：单向、故障转移和循环。

目标 Message Queue 消息服务器上的物理目标，生成的消息先被传送到此处，然后再路由并传送给用户。此物理目标由受管理对象标识和封装，客户机使用受管理对象指定生成和 / 或使用消息的目标。

域 JMS 客户机用于对 JMS 消息传送操作进行编程的一组对象。有两个编程域：一个用于点对点传送模型，另一个用于发布 / 订阅传送模型。

JMS（Java 消息服务） 定义 Java 客户机如何访问消息服务设备的接口和语义的标准集。这些接口为 Java 程序创建、发送、接收和阅读消息提供了一个标准方法。

JMS 提供者 实现消息传送系统的 JMS 接口并添加完整产品所需的管理和控制功能的产品。

Message Queue 客户机运行时 在 JMS 客户机与 Message Queue 消息服务器之间提供通信接口的软件。客户机运行时支持客户机向目标发送消息以及从目标接收消息所需的所有操作。

Message Queue 消息服务器 为 Message Queue 消息传送系统提供传送服务的软件，传送服务包括 JMS 客户机的连接、消息路由和传送、持久性、安全性以及日志。消息服务器负责维护从 JMS 客户机接收消息以及向使用消息的客户机发送消息的物理目标。

消息选择器 用户基于 JMS 消息标头中的属性值（选择器）选择消息的方法。消息服务基于消息选择器中的标准对消息进行过滤和路由。

消息服务 请参见 Message Queue 消息服务器。

消息 JMS 客户机使用的异步请求、报告或事件。消息包括标头（可以在其中添加其他字段）和主体两部分。消息标头指定标准字段和可选属性。消息主体包含要传输的数据。

消息传送 企业应用程序使用的异步请求、报告或事件系统，使松散耦合的应用程序可以安全可靠地传送信息。

点对点传送模型 生成方将消息发送给特定队列，用户从指定用来接收消息的队列中提取消息。一条消息只能传送给一个消息用户。

生成 将消息传送给客户机运行时，然后传送给目标。

生成方 由用于向某个目标发送消息的会话创建的对象 (MessageProducer)。在点对点传送模型中，生成方为发件人 (QueueSender)；在发布 / 订阅传送模型中，生成方为发布人 (TopicPublisher)。

发布 / 订阅传送模型 发布人和订阅者通常是匿名的，可以动态发布或订阅某个主题。系统将某个主题的多个发布人发布的消息分发给多个订阅者。

队列 管理员为实现点对点传送模型而创建的对象。队列可以一直接收消息，即使使用其消息的客户机处于非活动状态。队列可用作生成方和用户之间的中转站。

会话 发送和接收消息的单线程环境。可以是队列会话或主题会话。

主题 管理员为实现发布 / 订阅传送模型而创建的对象。可以将主题看作目录结构中的一个节点，负责收集和分发送给它的消息。使用主题作为消息传送的中转站，可以使消息发布人与消息订阅者相分离。

事务 不可细分的工作单位，要么整个完成，要么整个回滚。

用户组 Message Queue 客户机用户所属的组，用于授权对 Message Queue 消息服务器资源（如连接或目标）的访问。

索引

A

admin 连接服务 [50, 148](#)

API 文档 [29, 262, 263, 264](#)

安全

对象存储库 [166](#)

安全套接字层标准, [请参见 SSL](#)

安全性

管理器, [请参见安全性管理器](#)

加密, [请参见加密](#)

授权, [请参见授权](#)

验证, [请参见验证](#)

安全性管理器

关于 [60](#)

属性 [62](#)

作为代理组件 [49](#)

B

版本, 产品

关于 [33](#)

平台 [33](#)

企业 [33](#)

编程域 [42](#)

标准

关于 [64](#)

监视工具, [请参见标准监视工具](#)

数据, [请参见标准数据](#)

消息, [请参见标准消息](#)

主题目标 [65, 226](#)

标准监视工具

比较 [229](#)

基于消息的监视 API [226](#)

Message Queue 命令行实用程序 (imqcmd) [220](#)

Message Queue 日志文件 [225](#)

标准数据

代理, [请参见代理标准](#)

连接服务, [请参见连接服务标准](#)

目标, [请参见目标标准](#)

使用 imqcmd metrics [222](#)

使用代理日志文件 [225](#)

使用基于消息的监视 API [227](#)

说明性列表 [230](#)

标准消息

关于 [65, 226](#)

类型 [65, 226](#)

内容 [66](#)

C

Cloudscape [265](#)

操作系统

调整 Solaris 性能 [252](#)

性能影响 [216](#)

插入的持久性

关于 [59](#)

设置 [265](#)

性能调整 [256](#)

D 部分

查询

代理 144

连接服务 147, 149, 151

长期订阅

ClientID, 和 43

管理 161

关于 42

id 140

列出 161

清除消息 161

销毁 161, 162

性能影响 212

长期订阅者, 请参见[长期订阅](#)

持久性

插入的, 请参见[插入的持久性](#)

持久性管理器, 请参见[持久性管理器](#)

内置的 58

数据存储 请参见[数据存储](#)

持久性管理器

插入的持久性 265

关于 57

JDBC 数据存储 267

数据存储, 请参见[数据存储](#)

属性 59

作为代理组件 49

持久性消息 43

D

代理

安全性管理器, 请参见[安全性管理器](#)

标准, 请参见[代理标准](#)

查询 144

持久性管理器, 请参见[持久性管理器](#)

从故障中恢复 57

多代理群集, 请参见[代理群集](#)

访问控制, 请参见[授权](#)

更新属性 144

关闭 146

管理 142

关于 48

HTTP 支持 275

httpjms 连接服务属性 276

HTTPS, 支持 284

httpsjms 连接服务属性 287

互连, 请参见[代理群集](#)

恢复 143, 145

JDBC 支持, 请参见[JDBC 支持](#)

监视, 请参见[代理监视服务](#)

控制状态 145

连接到 141

连接到一起 130

连接服务, 请参见[连接服务](#)

列出连接服务 148

内存管理 55, 154, 219

配置文件, 请参见[配置文件](#)

启动 123

启动基于 SSL 的服务 201

确认 (Ack) 54, 168

群集, 请参见[代理群集](#)

日志记录, 请参见[日志记录器](#)

实例名称 126

实例配置属性 118

属性 144

Windows 服务, 运行为 297

显示属性 144

限制行为 55, 219

消息流控制, 请参见[消息流控制](#)

消息路由, 请参见[消息路由器](#)

消息容量 56, 122, 145

暂停 143, 145

重新启动 57, 143, 146

主管代理 75

自动创建目标属性 70

组件和功能 48

代理标准

报告间隔, 日志记录器 126

标准数量 231

标准消息 66

日志记录器属性 66, 225

使用 imqcmd 146, 223, 224

使用代理日志文件 226

使用基于消息的监视 226

代理监视服务

关于 63

属性 66

代理群集

安全的交叉代理连接 130

连接代理 130

配置更改记录 75

配置属性 76, 128

群集配置文件 76, 128

设置属性 129

使用原因 73, 218

体系结构 73, 218

添加代理到 131

同步时钟 128

信息传播 75

性能影响 218

选项以指定 125

在仅用于开发的环境中 76

重新启动代理 131

主管代理 75, 76

代理实例, 请参见代理

点对点传送 42

订阅

长期, 请参见长期订阅

关于 42

端口, 动态分配 51

端口映射器

端口分配 52, 126

关于 51

队列

负载均衡传送, 请参见负载均衡队列传送

属性 154

添加受管理对象 177

自动创建 70, 119

对象存储

位置 261, 262, 263

对象存储库

关于 166

LDAP 服务器 166

LDAP 服务器属性 166

文件系统存储 167

文件系统存储属性 167

F

发布 / 订阅传送 42

防火墙 51, 273

访问规则 194

访问控制文件

版本 193

访问规则 194

格式 193

使用 192

位置 262, 263, 264

分布式事务

关于 44

XA 资源管理器 44, 162

请参见 XA 连接工厂

负载均衡的队列传送

性能调整 257

负载均衡队列传送

关于 69

属性 71, 155

服务类型

ADMIN 50

NORMAL 50

G

更新

代理 144

连接服务 148, 149, 151

工具, 管理, 请参见管理工具

故障切换 73

关闭代理 143, 146

管理

代理 142

目标 152

管理工具

管理控制台 86

关于 86

命令行实用程序 87

管理任务

开发环境 83

H 部分

生产环境 [84](#)

H

HTTP

代理 [273](#)

连接服务, 请参见 [httpsjms 连接服务](#)

支持体系结构 [273](#)

传输驱动程序 [273](#)

HTTP 连接

多个代理 [278](#)

请求时间间隔 [277](#)

隧道 Servlet, 请参见 [HTTPS 隧道 Servlet](#)

支持 [273](#)

HTTP 隧道 Servlet

部署 [275](#)

关于 [273](#)

httpjms 连接服务

关于 [50, 148](#)

配置 [276](#)

设置 [275](#)

HTTPS

连接服务, 请参见 [httpjms 连接服务](#)

支持体系结构 [273](#)

HTTPS 连接

多个代理 [289](#)

请求时间间隔 [287](#)

隧道 Servlet, 请参见 [HTTPS 隧道 Servlet](#)

支持 [273](#)

HTTPS 隧道 Servlet

部署 [285](#)

关于 [273](#)

httpsjms 连接服务

关于 [50, 148](#)

配置 [286](#)

设置 [284](#)

环境变量, 请参见 [目录变量](#)

恢复

代理 [143, 145](#)

连接服务 [147, 150](#)

目标 [158](#)

会话

介绍 [38](#)

确认选项 [44](#)

已处理 [44](#)

I

imq.accesscontrol.enabled 属性 [62, 118](#)

imq.accesscontrol.file.filename 属性 [63, 118](#)

imq.authentication.basic.user_repository 属性 [62, 118](#)

imq.authentication.client.response.timeout 属性 [62, 118](#)

imq.authentication.type 属性 [62, 118](#)

imq.autocreate.destination.isLocalOnly 属性 [71, 118](#)

imq.autocreate.destination.limitBehavior 属性 [71, 118](#)

imq.autocreate.destination.maxBytesPerMsg 属性 [71, 118](#)

imq.autocreate.destination.maxCount 属性 [71, 118](#)

imq.autocreate.destination.maxNumMsgs 属性 [71](#)

imq.autocreate.destination.maxNumProducers 属性 [71, 118](#)

imq.autocreate.destination.maxTotalMsgBytes 属性 [71, 119](#)

imq.autocreate.queue 属性 [70, 119, 144](#)

imq.autocreate.queue.consumerFlowLimit 属性 [72, 119](#)

imq.autocreate.queue.localDeliveryPreferred 属性 [72, 119](#)

imq.autocreate.queue.maxNumActiveConsumers 属性 [71, 119, 145](#)

imq.autocreate.queue.maxNumBackupConsumers 属性 [71, 119, 145](#)

imq.autocreate.topic 属性 [70, 119, 145](#)

imq.cluster.brokerlist 属性 [128](#)

imq.cluster.masterbroker 属性 [128](#)

- imq.cluster.port 属性 128
- imq.cluster.transport 属性 129
- imq.cluster.url 属性 128, 145
- imq.hostname 属性 52, 119
- imq.httpjms.http.connectionTimeout 属性 277
- imq.httpjms.http.pullPeriod 属性 277
- imq.httpjms.http.servletHost 属性 276
- imq.httpjms.http.servletPort 属性 277
- imq.httpsjms.https.connectionTimeout 属性 287
- imq.httpsjms.https.pullPeriod 属性 287
- imq.httpsjms.https.servletHost 属性 287
- imq.httpsjms.https.servletPort 属性 287
- imq.keystore.file.dirpath 属性 199
- imq.keystore.file.name 属性 199
- imq.keystore.password 属性 200, 204
- imq.log.console.output 属性 67, 119
- imq.log.console.stream 属性 67, 119
- imq.log.file.dirpath 属性 67, 119
- imq.log.file.filename 属性 67
- imq.log.file.name 属性 119
- imq.log.file.output 属性 66, 119
- imq.log.file.rolloverbytes 属性 67, 119, 145
- imq.log.file.rolloversecs 属性 67, 119, 145
- imq.log.level 属性 66, 119, 145
- imq.log.syslog.facility 属性 67, 120
- imq.log.syslog.identity 属性 67, 120
- imq.log.syslog.logconsole 属性 67, 120
- imq.log.syslog.logpid 属性 67, 120
- imq.log.syslog.output 属性 67, 120
- imq.log.timezone 属性 68, 120
- imq.message.expiration.interval 属性 56, 120
- imq.message.max_size 属性 56, 120, 145
- imq.metrics.enabled 属性 66, 120
- imq.metrics.interval 属性 66, 120
- imq.metrics.topic.enabled 属性 68, 120
- imq.metrics.topic.interval 属性 68, 120
- imq.metrics.topic.persist 属性 68, 120
- imq.metrics.topic.timetolive 属性 68, 120
- imq.passfile.dirpath 属性 63, 120
- imq.passfile.enabled 属性 63, 120
- imq.passfile.name 属性 63, 120
- imq.persist.file.destination.message.filepool.limit 属性 59, 120
- imq.persist.file.message.cleanup 属性 60, 120
- imq.persist.file.message.filepool.cleanratio 属性 60, 120
- imq.persist.file.message.max_record_size 属性 59, 121
- imq.persist.file.message.vrfile.max_record_size 属性 58
- imq.persist.file.sync.enabled 属性 59, 121
- imq.persist.jdbc.brokerid 属性 268
- imq.persist.jdbc.closedburl 属性 268
- imq.persist.jdbc.createdburl 属性 268
- imq.persist.jdbc.driver 属性 268
- imq.persist.jdbc.needpassword 属性 268
- imq.persist.jdbc.opendburl 属性 268
- imq.persist.jdbc.password 属性 204, 268
- imq.persist.jdbc.table.IMQCCREC35 属性 269
- imq.persist.jdbc.table.IMQDEST35 属性 269
- imq.persist.jdbc.table.IMQINT35 属性 269
- imq.persist.jdbc.table.IMQLIST35 属性 269
- imq.persist.jdbc.table.IMQMSG35 属性 269
- imq.persist.jdbc.table.IMQPROPS35 属性 269
- imq.persist.jdbc.table.IMQSV35 属性 269
- imq.persist.jdbc.table.IMQTACK35 属性 270
- imq.persist.jdbc.table.IMQTXN35 属性 270
- imq.persist.jdbc.user 属性 268
- imq.persist.store 属性 59, 121, 267
- imq.ping.interval 属性 52, 121
- imq.portmapper.backlog 属性 52, 121
- imq.portmapper.hostname 属性 52, 121
- imq.portmapper.port 属性 52, 121, 145
- imq.protocol protocol_type inbufsz 253
- imq.protocol protocol_type nodelay 253
- imq.protocol protocol_type outbufsz 253
- imq.resource_state.count 属性 57, 121

imq.resource_state.threshold 属性 56, 121
 imq.service.activelist 属性 52, 121
 imq.service_name.accesscontrol.enabled 属性 62, 121
 imq.service_name.accesscontrol.file.filename 属性 63, 121
 imq.service_name.authentication.type 属性 62, 121
 imq.service_name.max_threads 属性 53, 121
 imq.service_name.min_threads 属性 53, 122
 imq.service_name.protocol_type.hostname 属性 53, 122, 128
 imq.service_name.protocol_type.port 属性 53, 122
 imq.service_name.threadpool_model 属性 53, 122
 imq.shared.connectionMonitor_limit 属性 53, 122
 imq.system.max_count 属性 56, 122, 145
 imq.system.max_size 属性 56, 122, 145
 imq.transaction.autorollback 属性 57, 122, 164
 imq.user_repository.ldap.base 属性 191
 imq.user_repository.ldap.gidattr 属性 191
 imq.user_repository.ldap.grpbase 属性 191
 imq.user_repository.ldap.grpfilter 属性 191
 imq.user_repository.ldap.grpsearch 属性 191
 imq.user_repository.ldap.memattr 属性 191
 imq.user_repository.ldap.password 属性 191, 204
 imq.user_repository.ldap.principal 属性 191
 imq.user_repository.ldap.server 属性 190
 imq.user_repository.ldap.ssl.enabled 属性 192
 imq.user_repository.ldap.timeout 属性 191
 imq.user_repository.ldap.uidattr 属性 191
 imq.user_repository.ldap.usrfilter 属性 191
 IMQ_HOME 目录变量 26
 IMQ_JAVAHOME 目录变量 27
 IMQ_VARHOME 目录变量 26
 imqAckOnAcknowledge 属性 168
 imqAckOnProduce 属性 168
 imqAckTimeout 属性 168
 imqAddressList 属性 168
 imqAddressListBehavior 属性 168
 imqAddressListIterations 属性 168

imqbrokerd 命令
 关于 87
 命令语法 123
 使用 123
 选项 125
 imqBrokerHostName 属性 (Message Queue 3.0) 168
 imqBrokerHostPort 属性 (Message Queue 3.0) 168
 imqBrokerServicePort 属性 (Message Queue 3.0) 169
 imqcmd 命令
 安全连接到代理 141, 202
 标准监视 220
 关于 87
 连接到代理 141
 命令语法 138
 目标管理 152
 事务管理 162
 选项 140
 用于 138
 子命令 138
 imqConfiguredClientID 属性 169
 imqConnectionFlowCount 属性 169
 imqConnectionFlowLimit 属性 169
 imqConnectionFlowLimitEnabled 属性 169
 imqConnectionType 属性 (Message Queue 3.0) 169
 imqConnectionURL 属性 (Message Queue 3.0) 169
 imqConsumerFlowLimit 属性 169
 imqConsumerFlowThreshold 属性 169
 imqdbmgr 命令
 关于 88
 命令语法 271
 选项 272
 子命令 271
 imqDefaultPassword 属性 169
 imqDefaultUsername 属性 169
 imqDestinationDescription 属性 80, 170
 imqDestinationName 属性 80, 170
 imqDisableSetClientID 属性 169
 imqFlowControlLimit 属性 169
 imqJMSDeliveryMode 属性 169
 imqJMSExpiration 属性 169

- imqJMSPriority 属性 169
- imqkeytool 命令
 - 关于 88
 - 命令语法 199, 284
 - 使用 199, 284
- imqLoadMaxToServerSession 属性 169
- imqobjmgr 命令
 - 关于 87
 - 命令语法 171
 - 使用 171
 - 选项 171
 - 子命令 171
- imqOverrideJMSDeliveryMode 属性 169
- imqOverrideJMSExpiration 属性 169
- imqOverrideJMSHeadersToTemporaryDestinations 属性 169
- imqOverrideJMSPriority 属性 169
- imqQueueBrowserMax MessagesPerRetrieve 属性 169
- imqQueueBrowserRetrieveTimeout 属性 169
- imqReconnectAttempts 属性 169
- imqReconnectEnabled 属性 169
- imqReconnectInterval 属性 169
- imqSetJMSXAppID 属性 169
- imqSetJMSXConsumerTXID 属性 170
- imqSetJMSXProducerTXID 属性 170
- imqSetJMSXRcvTimestamp 属性 170
- imqSetJMSXUserID 属性 170
- imqSSLIsHostTrusted 属性 (Message Queue 3.0) 170
- imqsvcadm 命令
 - 关于 88
 - 命令语法 298
 - 使用 298
 - 选项 298
 - 子命令 298
- imqusermgr 命令
 - 关于 88
 - 密码 188
 - 命令语法 185
 - 使用 185

- 选项 186
- 用户名 188
- 子命令 186

J

- J2EE 应用程序
 - EJB 规范 39
 - JMS, 和 39
 - 消息驱动 bean, 请参见消息驱动 bean
- Java 虚拟机, 请参见 JVM
- JDBC 支持
 - 关于 59
 - 驱动程序 265, 268
 - 设置 265
- JDK
 - 指定路径 125
 - 指定路径的选项 140, 172, 299
- JMS
 - 编程模型 37
 - 规范 29, 31, 37
 - 消息结构 37
- jms 连接服务 50, 148
- JNDI
 - 查找 78, 81, 104, 173
 - 查找名称 173, 177
 - 初始上下文 166, 167
 - 地址 (提供商 URL) 166, 167
 - 对象存储 87
 - 对象存储库 166
 - 对象存储库属性 166, 173
 - Message Queue 支持 32
 - 受管理对象, 和 39, 41
 - 消息驱动 bean, 和 41
- JVM
 - 标准, 请参见 JVM 标准
 - 性能调整 252
 - 性能影响 216
- JVM 标准
 - 标准数量 231

K 部分

- 使用 `imqcmd metrics` [222](#)
- 使用代理日志文件 [226](#)
- 使用基于消息的监视 [226](#)
- 基于 SSL 的连接服务
 - 启动 [201](#)
 - 设置 [183, 198](#)
- 基准检验, 性能 [206](#)
- 加密
 - 关于 [62](#)
 - 基于 SSL 的服务 [198](#)
 - 密钥工具, 和 [62](#)
- 监视, 请参见[性能监视](#)
- 监听器 [38, 40](#)
- 键值存储
 - 文件 [284](#)

K

- 客户机
 - 编程模型 [37](#)
 - 标识符 (ClientID) [43](#)
 - 应用程序, 请参见[客户机应用程序](#)
 - 运行时, 请参见[客户机运行时](#)
- 客户机应用程序
 - 示例 [29, 262, 263, 264](#)
 - 提供商无关 [41](#)
 - 系统属性, 和 [81](#)
 - 影响性能的因素 [209](#)
- 客户机运行时
 - 关于 [76](#)
 - 配置 [219](#)
 - 消息流调整 [258](#)
- 可靠传送 [43](#)
 - 性能折衷 [45, 210](#)
- 可移植性, 请参见[提供商无关](#)
- 控制消息 [54](#)

L

- LDAP 服务器
 - 对象存储库属性 [166](#)
 - 验证故障切换 [190](#)
 - 用户系统信息库访问 [190](#)
- 连接
 - 查询 [151](#)
 - 介绍 [38](#)
 - 列出 [151](#)
 - 相关命令 [151](#)
 - 性能影响 [216](#)
- 连接, 到代理 [141](#)
- 连接服务
 - `admin` [50, 148](#)
 - 标准数据, 请参见[连接服务标准](#)
 - 查询 [147, 151](#)
 - 端口映射器, 请参见[端口映射器](#)
 - 访问控制 [62](#)
 - 服务类型 [50](#)
 - 更新 [148, 149, 151](#)
 - 关于 [49](#)
 - HTTP, 请参见[HTTP 连接](#)
 - `httpjms` [50, 148](#)
 - HTTPS, 请参见[HTTPS 连接](#)
 - `httpsjms` [50, 148](#)
 - 恢复 [147, 150](#)
 - `jms` [50, 148](#)
 - 基于 SSL [200](#)
 - 静态端口 [52](#)
 - 连接类型 [50](#)
 - 启动时激活 [52](#)
 - 群集 [130, 198](#)
 - `ssladmin`, 请参见[ssladmin 连接服务](#)
 - `ssljms`, 请参见[ssljms 连接服务](#)
 - 属性 [52, 149](#)
 - 线程池管理器 [51](#)
 - 线程分配 [149](#)
 - 显示属性 [149](#)
 - 相关命令 [147](#)
 - 暂停 [147, 150](#)
- 连接服务标准
 - 标准数量 [233](#)

- 使用 imqcmd 标准 150
- 使用 imqcmd metrics 223
- 使用 imqcmd query 224
- 连接工厂受管理对象
 - ClientID 43
 - 覆盖 81
 - 关于 79
 - JNDI 查找 39
 - 介绍 38
 - 属性 79, 168
 - 添加 176
- 临时目标 72, 156
- 流控制, 请参见消息流控制
- 路由, 请参见消息路由器

M

- MDB, 请参见消息驱动 bean
- 密码
 - 编码 62
 - JDBC 204
 - LDAP 204
 - 密码文件, 请参见密码文件
 - 命名惯例 188
 - SSL 密钥存储 126, 200, 204
- 密码, 默认 169
- 密码文件
 - 代理配置属性 63
 - 命令行选项 126
 - 使用 204
 - 位置 204, 262, 263, 264
- 密码文件, 请参见密码文件
- 密钥存储
 - 属性 199
 - 文件 199
- 密钥对
 - 生成 199
 - 重新生成 200
- 密钥工具 62
- 命令文件 174

- 命令行实用程序
 - 关于 87
 - 基本语法 88
 - imqbrokerd, 请参见, imqbrokerd 命令
 - imqcmd, 请参见, imqcmd 命令
 - imqdbmgr 请参见, imqdbmgr 命令
 - imqkeytool, 请参见, imqkeytool 命令
 - imqobjmgr, 请参见, imqobjmgr 命令
 - imqsvcadm, 请参见, imqsvcadm 命令
 - imqusermgr, 请参见, imqusermgr 命令
 - 通用选项 89
- 命令行语法 88
- 命令选项 89
- 目标
 - 标准, 请参见目标标准
 - 创建 154
 - 访问控制 196
 - 更新属性 153
 - 管理 152
 - 恢复 153, 158
 - 获得有关信息 153
 - 介绍 48
 - 类型 68, 153
 - 列出 153, 156
 - 临时 72, 156
 - 清除消息来自 153, 159
 - 属性 154
 - 属性值 156
 - 物理 68
 - 显示属性值 156
 - 限制范围于群集中 71, 155
 - 限制行为 55, 154
 - 销毁 152, 154, 159
 - 信息 156
 - 压缩基于文件的数据存储 152
 - 暂停 153, 158
 - 主题, 请参见主题
 - 传送批量消息 72, 155
 - 自动创建 70, 197
- 目标标准
 - 标准数量 234
 - 使用 imqcmd 标准 153

N 部分

- 使用 imqcmd metrics [221, 223](#)
- 使用 imqcmd query [225](#)
- 使用基于消息的监视 [226](#)
- 目标受管理对象
 - 关于 [80](#)
 - 介绍 [38](#)
 - 属性 [170](#)
- 目录变量
 - IMQ_HOME [26](#)
 - IMQ_JAVAHOME [27](#)
 - IMQ_VARHOME [26](#)

N

- 内存管理
 - 对于代理 [55](#)
 - 使用目标属性 [154](#)
 - 性能调整 [256](#)
- 内置的持久性 [58](#)

O

- Oracle [265](#)

P

- PointBase [265](#)
- 配置更改记录 [75](#)
- 配置文件
 - 安装 [115](#)
 - 编辑 [118](#)
 - 模板 [261, 262, 263](#)
 - 模板位置 [261, 262, 263](#)
 - 默认 [115](#)
 - 实例 [116, 129, 144, 261, 262, 263](#)
 - 位置 [261, 262, 263](#)
- 瓶颈, 性能 [209](#)

- 平台版 [33](#)

Q

- 启动
 - 代理 [123](#)
 - 基于 SSL 的连接服务 [201](#)
 - 群集中的代理 [131](#)
- 企业版 [33](#)
- 清除, 来自目标的消息 [159](#)
- 权限
 - 访问控制属性文件 [61, 193](#)
 - 管理服务 [61](#)
 - 计算 [194](#)
 - 键值存储 [284](#)
 - Message Queue 操作 [60](#)
 - 密码文件 [204](#)
 - 嵌入式数据库 [267](#)
 - 数据存储 [59](#)
 - 用户系统信息库 [185](#)

- 确认
 - 代理 [54](#)
 - 等待周期 [168](#)
 - 关于 [44, 54](#)
 - 客户机 [54, 78](#)
 - 事务, 和 [55](#)
 - 传送, 对象 [54](#)

- 群集, *请参见*[代理群集](#)
- 群集连接服务
 - 端口号 [128](#)
 - 设置, 安全 [130, 198](#)
- 群集配置文件 [76](#)

R

- 日志记录, *请参见*[日志记录器](#)
- 日志记录器
 - 标准信息 [66](#)

- 种类 64
- 更改配置 134
- 关于 64
- 级别 64, 66, 126
- 输出通道 64, 135, 225
- 消息格式 134
- 写入到控制台 67, 127
- 重定向日志消息 136
- 转移标准 136
- 作为代理组件 50

- 日志文件
 - 默认位置 65, 261, 262, 263
 - 转移标准 67

- 容器
 - EJB 40
 - MDB 40

S

Servlet, 隧道 请参见 [HTTP 隧道 Servlet/HTTPS 隧道 Servlet](#)

SOAP 32

SSL

- 关于 62
- 加密, 和 198
- 连接服务, 请参见 [基于 SSL 的连接服务](#)
- 通过 HTTP 203
- 通过 TCP/IP 198

ssladmin 连接服务

- 关于 50, 148
- 设置 198

ssljms 连接服务

- 关于 50, 148
- 设置 198

syslog 65, 135

生成方

- 关于 38
- 目标限制 155
- 目标限制于 71

实例配置文件, 请参见 [配置文件](#)

示例应用程序 29, 262, 263, 264

事务

- 分布式, 请参见 [分布式事务](#)
- 管理 162
- 关于 44
- 回滚 162
- 确认, 和 55
- 提交 162, 163
- 信息 162
- 性能影响 211

使用方 38

受管理对象

- 查询 180
- 查找名称 172
- 队列, 参见 [队列](#)
- 对象存储库, 参见 [对象存储库](#)
- 更新 180
- 关于 39, 78
- 类型 39, 78, 168
- 连接工厂, 请参见 [连接工厂受管理对象](#)
- 列表 179
- 目标, 请参见 [目标受管理对象](#)
- 删除 179
- 属性 168
- 所需的信息 173
- 提供商无关 79
- XA 连接工厂, 请参见 [连接工厂受管理对象](#)
- 主题, 参见 [主题](#)

授权

- 管理 192
- 关于 60
- 用户组 61
- 请参见 [访问控制文件](#)

数据, Message Queue, 位置 261

数据存储

- 关于 57
- JDBC 可访问 59
- 位置 261, 262, 263
- 文本文件 58
- 性能影响 219
- 重置 127

属性

T 部分

- 安全性 62
- 持久性 59
- 代理, 更新 144
- 代理监视服务 66
- 代理实例配置 118
- httpjms 连接服务 276
- httpsjms 连接服务 287
- 连接服务 52
- 密钥存储 199
- 目标 154
- 目标, 请参见目标, 属性
- 内存管理 56, 154
- 群集配置 128
- 日志记录器 66
- 受管理对象 168
- 受管理对象, 请参见受管理对象, 属性
- 消息路由器 56
- 与 JDBC 相关 267
- 自动创建 70
- 隧道, Servlet 请参见 HTTP 隧道 Servlet/HTTPS 隧道 Servlet

T

- TCP 50, 148
- TLS 50, 148
- 提供商无关
 - 关于 41
 - 受管理对象 79

W

- Windows 服务, 代理运行行为 297
- 文件描述符限制 302

X

- XA 连接工厂
 - 请参见连接工厂受管理对象
- XA 连接工厂, 关于 45
- XA 资源管理器, 请参见分布式事务系统属性, 设置 81
- 线程池管理器
 - 共享线程 52
 - 关于 51
 - 专用线程 51
- 限制行为
 - 代理 55
 - 目标 55, 154
- 消息
 - 标准 65
 - 标准消息, 请参见标准消息
 - 持久性 43, 55, 57
 - 大小, 和性能 214
 - 代理限制 56, 122, 145
 - 点对点传送 42
 - 发布 / 订阅传送 42
 - 负荷平衡队列传送 69
 - 过滤, 请参见选择器
 - 过期回收 56
 - 监听器 38, 78
 - 结构 37
 - 介绍 37
 - 可靠传送 44
 - 控制 54
 - 流控制, 请参见消息流控制
 - 路由和传送 53
 - 目标限制 154
 - 排列优先级 46
 - 排序 46
 - 确认 54
 - SOAP 32
 - 生成 77
 - 使用 77
 - 吞吐量性能 206
 - 延迟 206
 - 重新传送 55
 - 主体类型, 和性能 214

- 传送模式, 请参见[传送模式](#)
- 传送模型 [36, 42](#)
- 自目标清除 [153](#)
- 消息服务
 - 关于 [36](#)
 - 影响性能的因素 [215](#)
- 消息服务器
 - 多代理, 请参见[代理群集](#) [73](#)
 - 关于 [48](#)
 - 体系结构 [218](#)
- 消息流控制
 - 测量 [258](#)
 - 代理 [55, 154](#)
 - 限制 [258](#)
 - 性能调整 [258](#)
 - 性能影响 [219](#)
- 消息路由器
 - 关于 [53](#)
 - 属性 [56](#)
 - 作为代理组件 [49](#)
- 消息驱动 bean
 - 部署描述符 [41](#)
 - 关于 [40](#)
 - MDB 容器 [40](#)
 - 应用服务器支持 [41](#)
- 消息生成方, 请参见[生成方](#)
- 消息传送模型 [36, 42](#)
- 消息传送系统
 - Message Queue 体系结构 [48](#)
 - 体系结构 [36](#)
 - 消息服务 [36](#)
- 协议, 请参见[传输协议](#)
- 协议类型
 - HTTP [50, 148](#)
 - TCP [50, 148](#)
 - TLS [50, 148](#)
- 性能
 - 调整, 请参见[性能调整](#)
 - 关于 [205](#)
 - 衡量 [206](#)
 - 基线模式 [207](#)
 - 基准检验 [206](#)
- 监视, 请参见[性能监视](#)
- 可靠性折衷 [45, 210](#)
- 瓶颈 [209](#)
- 疑难解答 [237](#)
- 影响因素, 请参见[性能影响因素](#)
- 优化, 请参见[性能调整](#)
- 指示器 [206](#)
- 性能调整
 - 代理调整 [256](#)
 - 过程概述 [205](#)
 - 客户机运行时调整 [258](#)
 - 系统调整 [252](#)
- 性能监视
 - 标准数据, 请参见[标准数据](#)
 - 工具, 请参见[标准监视工具](#) [220](#)
- 性能影响因素
 - 操作系统 [216](#)
 - 长期订阅 [212](#)
 - 代理限制行为 [219](#)
 - JVM [216](#)
 - 连接 [216](#)
 - 确认模式 [212](#)
 - 事务 [211](#)
 - 数据存储 [219](#)
 - 消息大小 [214](#)
 - 消息服务器体系结构 [218](#)
 - 消息流控制 [219](#)
 - 消息主体类型 [214](#)
 - 选择器 [213](#)
 - 硬件 [215](#)
 - 传输协议 [216](#)
 - 传送模式 [210](#)
- 许可证
 - 对于 Message Queue 版本 [33](#)
 - 启动具有企业版许可证的代理实例 [124](#)
 - 启动选项 [126](#)
- 选择器
 - 关于 [46](#)
 - 性能影响 [213](#)
 - 作为消息属性 [37](#)

Y 部分

Y

验证

- 管理 [184](#)
- 关于 [60](#)

疑难解答 [237](#)

硬件,性能影响 [215](#)

应用程序, *请参见*[客户机应用程序](#)

应用程序服务器,和 Message Queue [41](#)

用户名

- 格式 [188](#)
- 默认 [185](#)
- 属性 [169](#)

用户系统信息库

- 管理 [188](#)
- 关于 [60](#)
- LDAP 服务器 [190](#)
- 平台相关性 [185](#)
- 属性 [62](#)
- 填充 [188](#)
- 位置 [262](#),[263](#)
- 文本文件 [184](#)
- 用户状态 [187](#)
- 用户组 [187](#)

用户组

- 关于 [60](#)
- 默认 [61](#)
- 删除指定 [187](#)
- 预定义 [187](#)

域 [42](#)

Z

暂停

- 代理 [143](#),[145](#)
- 连接服务 [147](#),[150](#)
- 目标 [153](#),[158](#)

证书 [199](#),[284](#)

重新启动代理 [143](#),[146](#)

重新传送标志 [55](#)

主管代理 [75](#),[76](#)

主题

- 关于 [42](#)
- 属性 [154](#)
- 添加受管理对象 [177](#)
- 自动创建 [70](#),[119](#)
- 作为物理目标 [70](#)

主题目标, *请参见*[主题](#)

传输协议

- 相对速度 [217](#)
- 协议类型, *请参见*[协议类型](#)
- 性能调整 [253](#)
- 性能影响 [216](#)

传送,可靠 [43](#)

传送模式

- 持久性 [43](#)
- 非持久性 [43](#)
- 性能影响 [210](#)

自动创建目标

- 关于 [70](#)
- 属性 [70](#)

自签名证书 [199](#),[284](#)

资源适配器 [41](#)

组件

- EJB [39](#)
- MDB [40](#)