Sun Java™ System

Message Queue 3.6 SP3

技術摘要

2005Q4

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

文件號碼：819-3566

目錄

圖

表

前言

本書 (「Sun Java™ System Message Queue 3.6 SP3 2005Q4 技術摘要」) 會介紹 Message Queue 訊息傳送服務的技術、概念、架構、功能以及特色。

因此，「Message Queue 技術摘要」爲 Message Queue 文件集中的其他書籍提供了基礎知識。閱讀 Message Queue 文件集中的其他書籍之前，應先閱讀本書。

本前言涵蓋下列各節：

本書適用對象

本指南適用於管理員、應用程式開發者與其他計劃使用 Message Queue 產品的使用者，或是希望瞭解該產品技術、概念、架構、功能以及特色的使用者。

管理員必須負責設定及管理 Message Queue 訊息傳送服務。本書假設您不瞭解訊息傳送系統知識。

應用程式開發者需負責撰寫使用 Message Queue 服務的 Message Queue 用戶端應用程式，以與其他用戶端應用程式交換訊息。本書假設您不瞭解 Java Message Service (JMS) 規格 (由 Message Queue 服務實作)。

閱讀本書之前

閱讀本書沒有任何先決條件。閱讀 Message Queue 開發者與管理指南之前，應閱讀本書以瞭解 Message Queue 的基本概念。

本書架構

本書是按從頭到尾順序閱讀的方式設計的；每一章節是以之前章節中包含的資訊爲基礎進行說明。下表簡要描述了每章的內容：

表 1　本書內容與架構

章節	說明
第 1 章 「訊息傳送系統：簡介」	介紹訊息傳送中介軟體技術、討論 JMS 標準，以及說明 Message Queue 服務如何應用該項標準。
第 2 章 「用戶端程式設計模型」	說明 JMS 程式設計模型，以及應如何使用 Message Queue 用戶端執行階段來建立 JMS 用戶端。說明對 C++ 用戶端與 SOAP 訊息傳輸的執行階段支援。
第 3 章「Message Queue 服務」	討論管理工作與工具，並說明用來配置連線、路由、永久性、安全性與監視的代理程式服務。
第 4 章 「代理程式叢集」	討論 Message Queue 代理程式叢集的架構及使用。
第 5 章 「Message Queue 和 J2EE」	探索在 J2EE 平台環境中實作 JMS 支援的結果。
附錄 A 「選擇性 JMS 功能的 Message Queue 實作」	說明 Message Queue 產品如何處理 JMS 可選項目。

表 1　本書內容與架構 (續)

章節	說明
附錄 B「Message Queue 功能」	列出 Message Queue 功能、總結這些功能的執行步驟，並提供參照以便獲得進一步資訊。
字彙表	提供使用 Message Queue 時，可能遇到的專有名詞和概念的資訊。

本書使用慣例

本節提供了有關本文件中所用慣例的資訊。

文字慣例

表 2　文件慣例

格式	說明
斜體	斜體文字表示定位字元。請在看到斜體文字時將其取代爲適當的字串或值。斜體文字也用於指定需要強調的文件標題，或用於引入的單字或片語。
固定間距字體	固定間距文字表示範例程式碼、在指令行上所輸入的指令、目錄名稱、檔案名稱或路徑名稱、錯誤訊息文字、類別名稱、方法名稱 (包括簽名中的所有元素)、套裝軟體名稱、保留字以及 URL。
[]	指示指令行語法敘述中可選值的方括號。方括號也用於表示對話方塊中的所有 UI 視窗、按鈕、選項、訊息和對話方塊標題等等。
全部大寫	全大寫的文字表示檔案系統類型 (GIF、TXT、HTML 等等)、環境變數 (IMQ_HOME) 或縮寫字 (JMS、JSP)。
「書名」	書名。
鍵+鍵	同時按下的按鍵透過加號連接在一起：Ctrl+A 表示同時按下這兩個鍵。
鍵-鍵	連續按下的按鍵透過連字號連接在一起：Esc-S 表示按下 Esc 鍵，然後釋放，接著按下 S 鍵。

目錄變數慣例

Message Queue 使用三個目錄變數，其設定方式根據平台的不同而不同。表 3 描述了這些變數，並概述其在 Solaris™、Windows 以及 Linux 平台上的使用方式。

表 3 Message Queue 目錄變數

變數	說明
IMQ_HOME	此變數通常用於 Message Queue 元件中，指的是 Message Queue 基底目錄 (根安裝目錄)： • 在 Solaris 上，沒有 Message Queue 根安裝目錄。因此，未在 Message Queue 元件中使用 IMQ_HOME 來指代 Solaris 上的檔案位置。 • 在 Solaris 上，Sun Java System Application Server 的 Message Queue 根安裝目錄位於 Application Server 基底目錄下的 /imq。 • 在 Windows 上，Message Queue 根安裝目錄由 Message Queue 安裝程式設定 (依預設，該目錄爲 C:\Program Files\Sun\MessageQueue3)。 • 在 Windows 上，Sun Java System Application Server 的 Message Queue 根安裝目錄位於 Application Server 基底目錄下的 /imq。 • 在 Linux 上，沒有 Message Queue 根安裝目錄。因此，未在 Message Queue 元件中使用 IMQ_HOME 來指定 Linux 上的檔案位置。
IMQ_VARHOME	這是儲存 Message Queue 暫存檔或動態建立的配置檔與資料檔的 /var 目錄。可以將其設定爲指向任何目錄的環境變數。 • 在 Solaris 上，IMQ_VARHOME 預設爲 /var/imq 目錄。 • 在 Solaris 上，針對 Sun Java System Application Server, Evaluation Edition，IMQ_VARHOME 預設爲 IMQ_HOME/var 目錄。 • 在 Windows 上，IMQ_VARHOME 預設爲 IMQ_HOME/var 目錄。 • 在 Windows 上，針對 Sun Java System Application Server，IMQ_VARHOME 預設爲 IMQ_HOME/var 目錄。 • 在 Linux 上，IMQ_VARHOME 預設爲 /var/opt/imq 目錄。

表 3　Message Queue 目錄變數 (續)

變數	說明
IMQ_JAVAHOME	這是指向 Message Queue 可執行檔所需的 Java™ 執行環境 (JRE) 位置的環境變數： • 在 Solaris 上，IMQ_JAVAHOME 依下列順序搜尋 Java 執行階段，但是使用者可以選擇性地將值設定爲所需 JRE 駐留的任意位置。 Solaris 8 或 9： /usr/jdk/entsys-j2se /usr/jdk/jdk1.5.* /usr/jdk/j2sdk1.5.* /usr/j2se Solaris 10： /usr/jdk/entsys-j2se /usr/java /usr/j2se • 在 Linux 上，Message Queue 先依下列順序搜尋 Java 執行階段，但是使用者可以選擇性地將 IMQ_JAVAHOME 值設定爲所需 JRE 駐留的任意位置。 /usr/jdk/entsys-j2se /usr/java/jre1.5.* /usr/java/jdk1.5.* /usr/java/jre1.4.2* /usr/java/j2sdk1.4.2* • 在 Windows 上，IMQ_JAVAHOME 預設爲 IMQ_HOME/jre，但是，使用者可以選擇性地將值設定爲所需 JRE 駐留的任意位置。

在本指南中，IMQ_HOME、IMQ_VARHOME 以及 IMQ_JAVAHOME 的顯示*不附帶*平台特定的環境變數表示法或語法 (例如 UNIX® 上的 $IMQ_HOME)。路徑名稱通常使用 UNIX 目錄分隔符號表示法 (/)。

相關文件

除了本指南以外，Message Queue 還提供了其他文件資源。

Message Queue 文件集

構成 Message Queue 文件集的文件依照一般的使用次序列示在表 4 中。

表 4 Message Queue 文件集

文件	使用者	說明
「Message Queue Installation Guide」	開發者與管理員	解釋如何在 Solaris、Linux 以及 Windows 平台上安裝 Message Queue 軟體。
「Message Queue 版本說明」	開發者與管理員	包括對新功能、限定、已知錯誤以及技術性注意事項的描述。
「Message Queue 技術概要」	開發者、管理員、設計人員	介紹 Message Queue 產品的技術、概念、架構、功能以及特色。
「Message Queue 管理指南」	管理員，作時建議開發者使用	提供使用 Message Queue 管理工具執行管理工作時所需的背景與資訊。
「Message Queue Developer's Guide for Java Clients」	開發者	為使用 Message Queue 執行 JMS 和 SOAP/JAXM 標準的 Java 用戶端程式開發者提供快速入門教學和程式設計資訊。
「Message Queue Developer's Guide for C Clients」	開發者	為使用 C 介面 (C-API) 到 Message Queue 訊息服務的 C 用戶端程式開發者提供程式設計和參照文件。

線上說明

Message Queue 包括用於執行 Message Queue 訊息服務管理工作的指令行公用程式。若要存取這些公用程式的線上說明，請參閱「Message Queue 管理指南」。

Message Queue 還包括圖形化使用者介面 (GUI) 管理工具，即管理主控台 (`imqadmin`)。上下文關聯的線上說明包括在管理主控台中。

JavaDoc

下列位置提供 JavaDoc 格式的 Message Queue Java 用戶端 API (包括 JMS API) 文件：

平台	位置
Solaris	`/usr/share/javadoc/imq/index.html`
Linux	`/opt/sun/mq/javadoc/index.html/`
Windows	`IMQ_HOME/javadoc/index.html`

本文件可在任何 HTML 瀏覽器 (如 Netscape 或 Internet Explorer) 中檢視。它包括標準的 JMS API 文件，以及用於 Message Queue 管理物件的 Message Queue 特定的 API (請參閱「Message Queue Developer's Guide for Java Clients」的第 3 章)，對訊息傳送應用程式的開發者很有幫助。

範例用戶端應用程式

許多提供範例用戶端應用程式碼的範例應用程式所在目錄位置因作業系統而異 (請參閱「Message Queue 管理指南」)。

請參閱該目錄及其各個子目錄中的 README 檔案。

Java Message Service (JMS) 規格

JMS 規格可在以下位置找到：

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

規格包括範例用戶端程式碼。

相關協力廠商文件參考

本文件提供了協力廠商的 URL 及其他相關資訊作爲參考。

備註 Sun 對於本文件中所提及之協力廠商網站的使用不承擔任何責任。Sun 對於此類網站或資源中的 (或透過它們所取得的) 任何內容、廣告、產品或其他材料不做背書，也不承擔任何責任。對於因使用或依靠此類網站或資源中的 (或透過它們所取得的) 任何內容、產品或服務而造成的或連帶產生的實際或名義上之損壞或損失，Sun 概不負責，也不承擔任何責任。

Sun 歡迎您提出寶貴意見

Sun 致力於提高文件品質，因此誠心歡迎您提出意見與建議。

若要提出您的意見，請至 http://docs.sun.com 並按一下 [傳送您的回饋意見] (Send Comments)。在線上表格中，請提供文件標題及文件號碼。文件號碼位於書本的標題頁或文件的頂部，通常是一組七位或九位數的數字。

提出意見時您還需要在表格中輸入此文件的英文標題和文件號碼。例如，本文件的英文文件號碼爲 819-2574，完整標題爲「Sun Java™ System Message Queue 3.6 SP3 Technical Overview」。

如需查看 Message Queue 特有的問題，也可參閱
http://swforum.sun.com/jive/forum.jspa?forumID=24

第 1 章

訊息傳送系統：簡介

Sun Java™ System Message Queue 是一種中介軟體產品，用於執行與擴充 Java Message Service (JMS) 標準的訊息傳送。如果您已經相當瞭解這點，應該從第 27 頁的「Message Queue：元素和功能」一節開始閱讀。否則，請從頭開始閱讀。

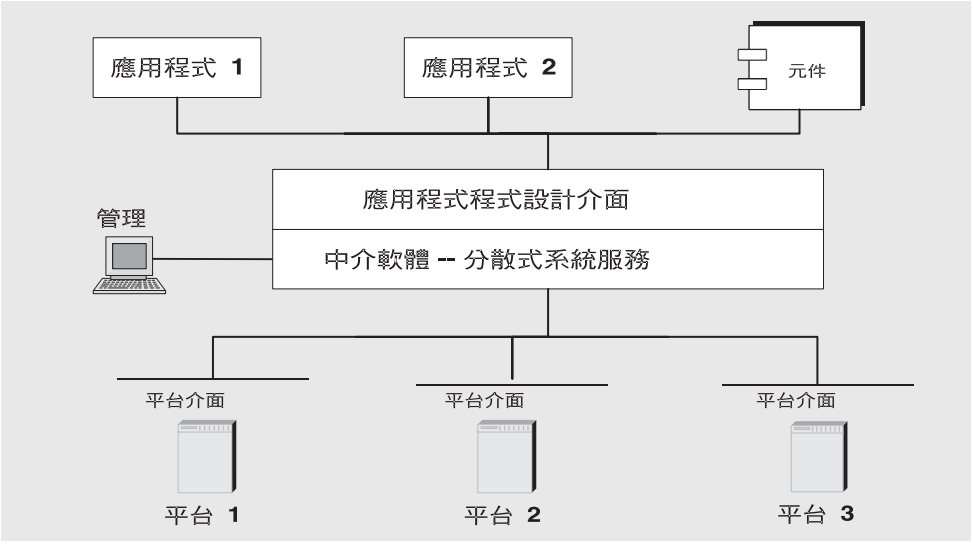
本章描述訊息傳送技術，將說明 Message Queue 等產品，解釋 Message Queue 的執行方式，並擴充標準化此技術的 JMS 規格。本節涵蓋下列主題：

訊息導向中介軟體 (MOM)

由於企業、機構和技術日新月異，提供服務的軟體系統必須能夠適應這些變更。企業可能因爲合併、新增服務或擴充可用的服務，而急需重建企業資訊系統。在此迫切的情況下，企業需要整合新的元件或儘可能有效地調整現有的元件。整合異質元件最簡單的方法並非將其重建爲同質元素，而是提供一個層級，使其可以無視差異進行通訊。此層級稱爲*中介軟體*，可讓獨立開發及在不同網路平台上執行的軟體元件 (應用程式、Enterprise Java Bean、Servlet 和其他元件) 彼此互動。當此互動成爲可能時，網路就能變成電腦。

如圖 1-1 所示，中介軟體介於應用程式層與平台層 (作業系統與基礎網路服務) 之間。

圖 1-1 中介軟體



分散於不同網路節點的應用程式會使用應用程式介面進行通訊，不用擔心主控其他應用程式的作業環境之詳細資料，也不用擔心連線到這些應用程式的服務。此外，藉由提供管理介面，這種新型虛擬的互連應用程式系統非常可靠安全。其效能可以測量和調整，並且可以在不遺失功能的情況下進行縮放。

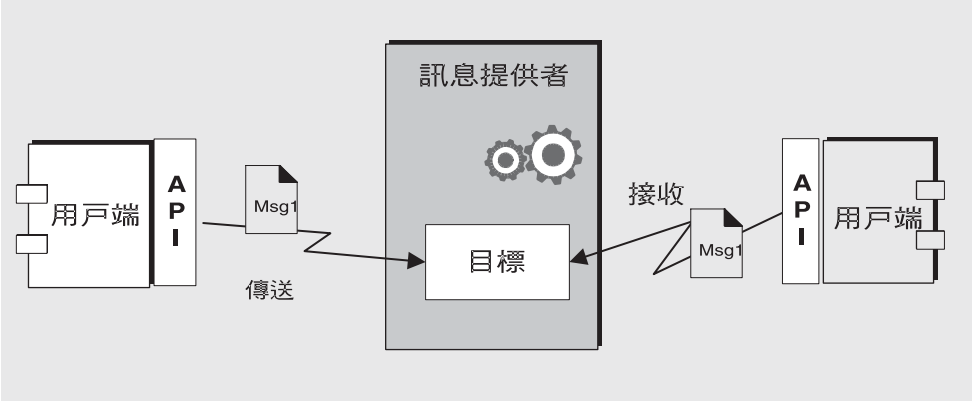
中介軟體可以分爲下列種類：

- 遠端程序呼叫或 RPC 型中介軟體：允許應用程式中的程序像呼叫本機程序那樣呼叫遠端應用程式中的程序。該中介軟體執行連結機制，尋找遠端程序並使其可供呼叫者使用。在過去，這類中介軟體會處理程序型程式，現在它還可包含物件型元件。
- 物件請求代理程式或 ORB 型中介軟體：使應用程式的物件可以在異質網路上進行分發與共用。
- 訊息導向中介軟體或 MOM 型中介軟體：透過傳送與接收訊息，允許分散的應用程式進行通訊與交換資料。

所有這些模型均可以透過網路，使一個軟體元件影響另一個元件的運作方式。不同之處在於 RPC 型和 ORB 型中介軟體會建立元件緊密耦合的系統，而 MOM 型系統則允許耦合較鬆散的元件。在 RPC 型或 ORB 型系統中，當程序呼叫另一個程序時，必須等待被呼叫的程序傳回，才能繼續其他工作。如前所述，在這些模型中，中介軟體的部份功能是超級連結程式，在網路上尋找被呼叫程序，並使用網路服務傳送函數或方法參數至此程序，然後再傳回結果。

MOM 型系統可透過非同步交換訊息來進行通訊，如圖 1-2 所示。

圖 1-2 MOM 型系統



訊息導向中介軟體利用訊息傳送提供者協調訊息傳送作業。MOM 系統的基本元素爲用戶端、訊息和 MOM 提供者 (包含 API 和管理工具)。MOM 提供者使用不同的架構來路由和傳送訊息：它可以使用集中化訊息伺服器，或者將路由和傳送功能分發給每部用戶端電腦。部份 MOM 產品則結合這兩種方法。

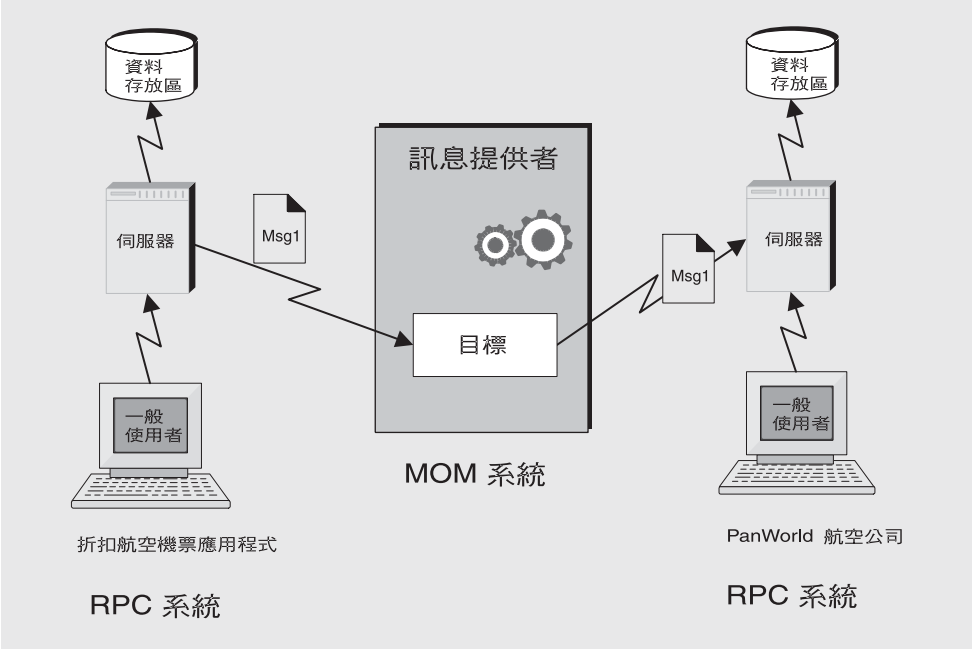
用戶端可以使用 MOM 系統，呼叫 API 傳送訊息給提供者所管理的目標。此呼叫會呼喚提供者服務來路由和傳送訊息。用戶端可在傳送訊息後繼續其他作業，確信提供者會保留訊息直到接收的用戶端接收到訊息爲止。此訊息型模型會耦合提供者的協調，以便有可能建立元件鬆散耦合的系統。這類系統即使在個別元件或連線失敗時，還能夠可靠地繼續運作，而不會當機。

讓訊息傳送提供者協調用戶端之間的訊息傳送還有一個優點，就是透過新增管理介面，得以監視並調整效能。因此，除了傳送、接收及處理訊息問題之外，用戶端應用程式能有效解決每個問題。若要解決互通性、可靠性、安全性、可延伸性和效能等問題，需視執行 MOM 系統的程式碼以及管理員而定。

到目前爲止，我們已描述了使用訊息導向中介軟體連接分散式元件的優點。但是也有缺點：其中一個缺點源於鬆散耦合本身。使用 RPC 系統，要等到被呼叫的函數完成其工作後，才會傳回呼叫函數。在非同步系統中，呼叫的用戶端可以繼續接收載入工作，直到需要處理此工作的資源耗盡且被呼叫的元件失敗爲止。當然，雖然透過監視效能和調整訊息流量可以將這些情況降到最低或避免出現，但是 RPC 系統不會這麼做。重要的是瞭解每種系統的優點和職責。各個系統適用於不同的工作。有時，需要結合兩種系統才能取得需要的實際運作方式。

圖 1-3 顯示 MOM 系統如何啓用兩種 RPC 型系統之間的通訊。圖的左側顯示將用戶端、伺服器和資料存放區元件分散到不同網路節點，以改善效能的應用程式。這是折扣航空訂位系統：一般使用者付費使用此服務，指定目的地和時間以尋找最便宜的機票。資料存放區會保留參與此程式的註冊使用者和航空公司之相關資訊。根據使用者的請求，伺服器上的邏輯會詢問參與航空公司價格、排序資訊，並向使用者顯示最便宜的三家航空公司。圖的右側顯示 RPC 型系統，表示任何一家參與航空公司之訂票 / 訂位系統。圖的右側將會視折扣程式連線到航空公司的數量重複。對於此類航空公司，資料存放區將保留其可用班機的相關資訊 (座位、班機時間和票價)。伺服器元件會回應一般使用者輸入的資料來更新該資訊。航空公司的伺服器也可訂閱 MOM 服務，接受來自折扣訂位系統的資訊請求，並傳回座位和票價資訊。如果用戶決定購買 PanWorld 航空公司的折扣機票，該系統的伺服器元件會更新資料存放區的資訊，然後產生請求者的機票，或傳送訊息到折扣服務以產生機票。

圖 1-3 結合 RPC 和 MOM 系統



此範例描述 RPC 系統和 MOM 系統之間的一些差異。之前已經提過耦合分散式元件的方式之差異。還有一項差異是 RPC 系統通常用於分散與連接用戶端元件和伺服器元件，其中用戶端通常是一般使用者；而 MOM 系統，用戶端通常是異質軟體元件，僅能透過訊息傳送的方式彼此相互作業。

MOM 系統有一個較爲嚴重的問題，即 MOM 是作爲專用產品來執行的。當依賴 SuperMOM-X 的公司併購使用 SuperMOM-Y 的公司時，會發生什麼事？ 要有標準的訊息傳送介面才能解決此問題。如果 SuperMOM-X 和 SuperMOM-Y 均執行此介面，則開發以在其中一個系統上執行的應用程式也可在另一個系統上執行。這類介面應該簡單易懂，卻又能提供足夠的功能，以支援複雜的訊息傳送應用程式。1998 年引入的 Java Message Service (JMS) 規格即爲此應運而生。下一節將描述 JMS 的基本功能，並說明如何開發該標準以涵蓋現有的專用 MOM 產品之常用元素，以及允許差異性和進一步的發展。

作爲 MOM 標準的 JMS

Java Messaging Service 規格起初是爲了讓 Java 應用程式存取現有的 MOM 系統而開發。自其問市後，已經爲許多現有的 MOM 供應商所採納，並作爲其本身的非同步訊息傳送系統。

JMS 設計人員在建立 JMS 規格時，想要擷取現有訊息傳送系統的重要元素。這些元素包括

- 訊息傳送提供者路由及傳送訊息的概念。
- 不同的訊息傳送模式或網域，例如點對點訊息傳送以及出版 / 訂閱訊息傳送
- 同步與非同步訊息接收的設備
- 可靠訊息傳送的支援
- 常用訊息格式，例如串流、文字和位元組

供應商執行 JMS 規格的方式是提供 *JMS 提供者*，其中包括執行 JMS 介面的程式庫；路由與傳送訊息的功能；以及管理、監視和調整訊息傳送服務的管理工具。路由和傳送功能可透過集中化訊息伺服器或代理程式來執行，或可透過每個用戶端執行階段的部份功能來執行。

JMS 提供者可同時扮演多種角色：它可以建立作爲獨立產品，或作爲更大型分散式執行階段系統的內嵌元件。作爲獨立產品時，可用於定義企業應用程式整合系統的主網路；而內嵌於應用程式伺服器時，則可支援元件間的訊息傳送。例如，J2EE 使用 JMS 提供者來執行訊息驅動 Bean，讓 EJB 元件得以傳送和接收訊息。

若要建立一套包含所有現有系統功能的標準，可能會造成系統難以瞭解與執行。相反，JMS 會定義訊息傳送概念和功能相同之處最少的共同點。這使得標準易於瞭解，且可最大化 JMS 應用程式在 JMS 提供者之間的可攜性。請注意，JMS 是 API 標準，不是協定標準。在供應商之間移動 JMS 用戶端很容易。但是不同的 JMS 供應商一般無法直接彼此互相通訊。

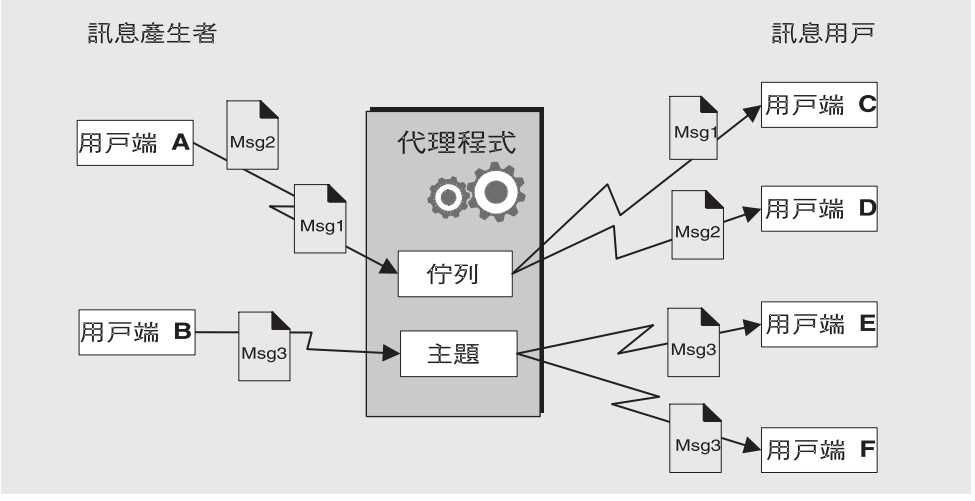
下一節將說明 JMS 規格所定義的基本物件和訊息傳送模式。

JMS 訊息傳送物件和模式

若要傳送或接收訊息，JMS 用戶端必須先連線到通常作爲訊息代理程式執行的 JMS 提供者：開啓用戶端與代理程式之間通訊通道的連線。然後，用戶端必須設定建立、產生和使用訊息的階段作業。您可以將階段作業想成是定義用戶端與代理程式之間特定對話的訊息串流。用戶端本身是*訊息產生者*及 / 或*訊息用戶*。訊息產生者會傳送*訊息*到代理程式管理的*目標*。訊息用戶會存取目標以使用訊息。訊息包括標頭、可選特性和內文。內文會保留資料；標頭則包含代理程式需要路由及傳送訊息的資訊；而特性可由用戶端應用程式或提供者定義，以滿足其處理訊息的需求。連線、階段作業、目標、訊息、產生者和用戶是組成 JMS 應用程式的基本物件。

用戶端應用程式藉由這些基本物件，可以使用兩種訊息傳送模式 (或網域) 來傳送與接收訊息。如圖 1-4 所示。

圖 1-4 JMS 訊息傳送模式



用戶端 A 和 B 是訊息產生者，經由兩種不同的目標傳送訊息到用戶端 C、D 和 E。

- 用戶端 A、C 和 D 之間的訊息傳送描述了點對點的模式。用戶端可以使用此模式，傳送訊息到佇列目標，僅讓一個接收者從該目標取得訊息。其他存取該目標的接收者無法取得此訊息。
- 用戶端 B、E 和 F 之間的訊息傳送描述出版 / 訂閱模式。用戶端可以使用此廣播模式，傳送訊息到主題目標，讓任意多的使用訂閱者從中接收訊息。每個用戶皆可取得各自的訊息副本。

不管是哪個網域，訊息用戶均能選擇同步或非同步取得訊息。同步用戶經由明確的呼叫可擷取訊息，而非同步用戶則可指定回呼方法，呼叫該方法可傳送擱置訊息。用戶也可以指定內送訊息的選取條件，以過濾訊息。

管理物件

JMS 規格建立了一套標準，結合現有 MOM 系統的許多元素，但不會企圖用盡所有可能性。相反地，它試圖建立可延伸的方案，以容納差異並得以進一步發展。JMS 留給個別的提供者許多訊息傳送元素，讓他們自行決定要如何定義與執行。這些元素包括負載平衡、標準錯誤訊息、管理 API、安全性、基礎線路協定和訊息存放區。下一節「Message Queue：元素和功能」將說明 Message Queue 如何執行這些元素，以及如何擴充 JMS 規格。

JMS 未完全定義的訊息傳送元素有兩個：連線工廠和目標。雖然這兩個都是 JMS 程式設計模型中的基本元素，但是提供者定義與管理這些物件的方式存在許多現有和預期的差異，所以不可能也不需要建立共同的定義。因此，這兩個物件不會以程式設計的方式建立，一般會使用管理工具來建立與配置。這兩個物件之後會儲存在物件存放區，並由 JMS 用戶端透過標準 JNDI 查找來存取。

- *連線工廠管理物件*可用於產生用戶端至代理程式的連線。這些物件封裝提供者特定的資訊，以管理訊息傳送運作方式的特定方面：連線處理、用戶端識別、訊息標頭置換、可靠性和流量控制等。源自指定連線工廠的每個連線會顯示該工廠的配置運作方式。
- *目標管理物件*可用於參照代理程式上的實體目標。這些物件封裝提供者特定的命名 (位址語法) 慣例，並指定使用目標內的訊息傳送網域：佇列或主題。

JMS 用戶端不需要查找管理物件，他們可以透過程式設計建立這些物件 (之後這些物件會儲存在代理程式的記憶體中)。若要有快速參考原型，透過程式設計建立這些物件可能是最簡單的方式。但是若要在生產環境中部署，在中央儲存庫內查找管理物件，會比較容易控制與管理訊息傳送的運作方式：

- 對連線工廠物件使用管理物件，管理員可以經由重新配置這些物件，來調整訊息傳送的效能。不需要重新撰寫程式碼就能改善效能。
- 對實體目標使用管理物件，管理員可以經由請求用戶端存取這些預先配置的物件，來控制這些目標在代理程式上的激增。
- 管理物件會讓開發者避免使用提供者特定的執行詳細資訊，允許他們爲一個提供者開發的程式碼在不變更或變更很少的情況下就可移植至其他提供者。

管理物件爲基本的 JMS 應用程式圖片增添了最終的創意，如圖 1-5 所示。

圖 1-5 JMS 應用程式的基本元素

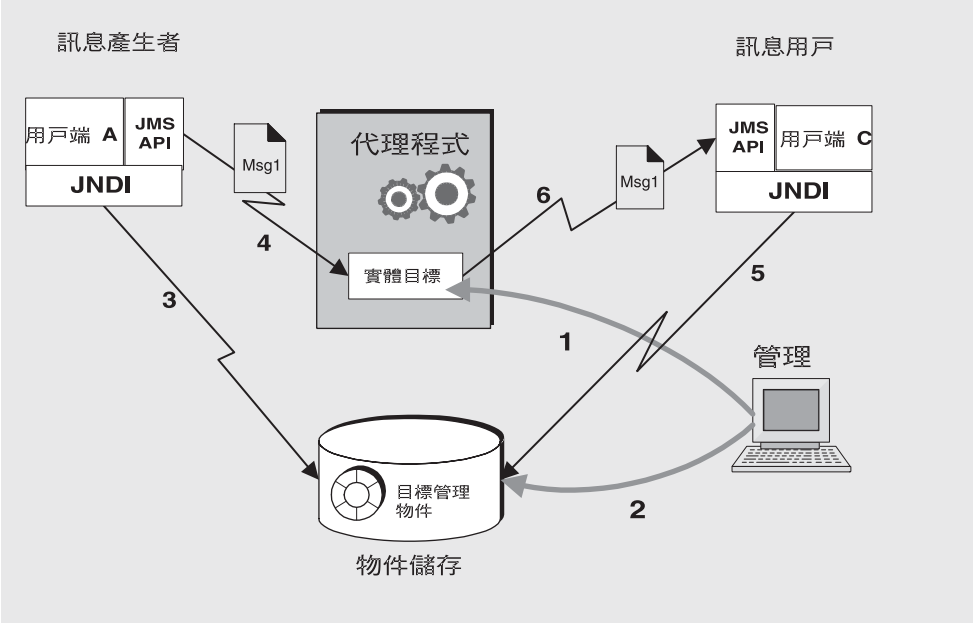


圖 1-5 顯示訊息產生者和訊息用戶如何使用目標管理物件存取對應的實體目標。標示的步驟表示管理員和用戶端應用程式使用此機制傳送和接收訊息時，需要採取的動作。

1. 管理員在代理程式上建立實體目標。
2. 管理員建立目標管理物件，並透過指定實體目標的對應名稱和類型進行配置：佇列或主題。
3. 訊息產生者使用 JNDI 查找呼叫，查找目標管理物件。
4. 訊息產生者傳送訊息到目標。

5. 訊息用戶查找目標管理物件，期望從中獲取訊息。
6. 訊息用戶從目標取得訊息。

使用連線工廠管理物件的程序很類似。管理員使用管理工具建立並配置連線工廠管理物件。用戶端查找連線工廠物件，並以此建立連線。

雖然使用管理物件增加了訊息傳送程序的步驟，但同時也增加了訊息傳送應用程式的強固性和可移植性。

Message Queue：元素和功能

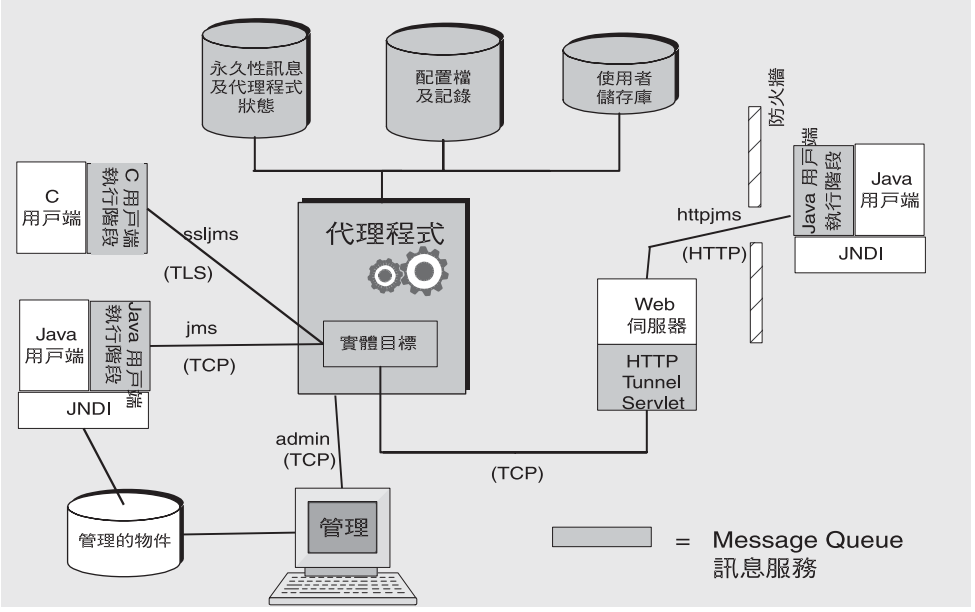
到目前爲止，我們已說明了訊息導向中介軟體的元素，以及使用 JMS 增加 MOM 應用程式的可移植性。接下來要說明 Message Queue 如何執行 JMS 規格，並介紹其用於提供可靠、安全和可延伸的訊息傳送服務之功能與工具。

首先，Message Queue 和許多 JMS 提供者一樣，可以用作獨立產品，也可以用作內嵌在 J2EE 應用程式伺服器的啓用技術，以提供非同步的訊息傳送。第 5 章「Message Queue 和 J2EE」對 Message Queue 在 J2EE 中所扮演的角色會有更詳盡的說明。其他 JMS 提供者不同的是，Message Queue 已指定爲 JMS 參照執行。此指定證明了 Message Queue 是正確完整的 JMS 執行。亦保證未來的 JMS 修訂和延伸可以繼續使用 Message Queue 產品。

Message Queue 服務

作爲 JMS 提供者，Message Queue 提供*訊息傳送服務*，該服務執行 JMS 介面並提供管理服務與控制。到目前爲止，在對 JMS 提供者的描述中，我們側重說明代理程式在轉送訊息中的作用。但事實上，JMS 提供者必須包括除代理程式以外的許多元素，以提供可靠、安全、可延伸的訊息傳送。圖 1-6 顯示組成 Message Queue 訊息服務的元素。這些元素包含多種連線服務 (支援不同的協定)、管理工具和資料存放區，以傳送訊息、監視和處理使用者資訊。Message Queue 服務包括圖中灰階標示的所有元素。

圖 1-6 Message Queue 服務



根據所見基本 JMS 模型的複雜程度，功能完整的 JMS 提供者其複雜程度是可想而知的。下列各節將說明上述的 Message Queue 服務元素。這些元素可分爲三類：代理程式、用戶端執行階段支援和管理。

連線到代理程式

如圖 1-6 所示，應用程式用戶端和管理用戶端均可連線到代理程式。JMS 規格並不表示提供者會執行任何特定線路的協定。應用程式用戶端和管理用戶端用於連線到代理程式的 Message Queue 服務，目前位於 TCP、TLS、HTTP 或 HTTPS 協定的最上層。(位於 HTTP 最上層的服務可讓訊息通過防火牆。)

- 提供 JMS 支援並可讓用戶端連線到代理程式 (`jms`、`ssljms`、`http` 或 `https`) 的服務類型爲 `NORMAL`，並位於 `TCP`、`TLS`、`HTTP` 或 `HTTPS` 協定的最上層。
- 可讓管理員連線到代理程式 (`admin`、`ssladmin`) 的服務類型爲 `ADMIN`，並位於 `TCP` 或 `TLS` 協定的最上層。

依預設，啓動代理程式時，jms 和 admin 服務會啓動並執行。此外，您可以將代理程式配置爲執行這些連接服務的任何一種或全部。每個服務支援特定的認證與授權 (存取控制) 功能，且每個服務爲多重執行緒，支援多重連線。

如果連線失敗，Message Queue 服務會自動重新嘗試將用戶端連線至同一個代理程式，或連線到其他代理程式 (如果已啓用此功能)。如需詳細資訊，請參閱附錄 B「Message Queue 功能」中的自動重新連線功能。

用戶端可以在建立取得連線的來源連線工廠時，配置連線執行階段支援。選項可讓您指定代理程式連線的目標、如何處理重新連線、訊息流量控制等。如需有關連線配置方式的其他資訊，請參閱第 43 頁的「連線工廠與連線」。

代理程式

代理程式是訊息服務的核心，可以可靠地路由和傳送訊息、認證使用者，並收集監視效能的資料。

- 爲了路由及傳送訊息，代理程式會將內送訊息置於其各自的目標，並管理進出這些目標的訊息流量。
- 爲了提供可靠的傳送，代理程式會使用永久存放區來儲存狀態資訊和永久性訊息 (直到訊息被接收爲止)。如果代理程式或連線失敗，儲存的資訊會讓代理程式復原代理程式的狀態，再重試作業。
- 爲了提供正在交換的資料安全性，代理程式會使用經過認證的連線。或者可以經由 SSL 之類的安全協定執行，來加密資料。代理程式還會使用並管理儲存庫，以保留使用者資訊以及使用者可以存取的資料或作業。代理程式會查找此儲存庫內的資訊，以認證請求服務的使用者，並授權使用者想要執行的作業。
- 爲了監視系統，代理程式會產生度量和診斷資訊，供管理員存取以測量效能並調整代理程式。度量資訊亦可透過程式設計的方式取得，以允許應用程式調整訊息流量和模式來改善效能。

Message Queue 服務提供多種管理工具，管理員可用於配置代理程式支援。如需更多資訊，請參閱第 31 頁的「管理」。

用戶端執行階段支援

建立 Message Queue 用戶端時，所連結的程式庫便會提供用戶端執行階段支援。您可以將用戶端執行階段想像爲 Message Queue 服務變成爲用戶端一部份的一個階段。例如，當用戶端程式碼呼叫 API 傳送訊息時，會呼叫這些程式庫中的程式碼，程式碼會爲協定封裝適當的訊息位元，而此協定是用於將訊息轉發至代理程式上的實體目標。

Java 和 C 用戶端支援

只有支援 Java 用戶端時才需要 JMS 提供者；但是，如圖 1-6 所示，Message Queue 用戶端可以使用 Java 或提供者特定的 C API，來傳送或接收訊息。這些介面會在 Java 或 C 執行階段程式庫中執行，它會執行建立代理程式連線與依據連線服務請求封裝位元的實際工作。

- Java 用戶端執行階段提供 Java 用戶端與代理程式互動所需的物件。這些物件包括連線、階段作業、訊息、訊息產生者與訊息用戶。
- C 用戶端執行階段提供 C 用戶端與代理程式互動所需的功能和結構。它支援 JMS 程式設計模型的程序版本。C 用戶端無法使用 JNDI 來存取管理物件，但是可以透過程式設計建立連線工廠和目標。

Message Queue 服務提供 C API，使舊版 C 和 C++ 應用程式能參與 JMS 型訊息傳送。這兩種 API 所提供的功能中有許多不同之處，會在第 55 頁的「Java 用戶端與 C 用戶端」中進行說明。

請謹記，JMS 規格是僅限於 Java 用戶端使用的標準。C 支援是 Message Queue 提供者特有的支援，不能用於計劃移植到其他提供者的用戶端應用程式。

Java 用戶端的 SOAP 支援

Message Queue Java 用戶端也可以傳送和接收包裝成 JMS 訊息的 SOAP 訊息。SOAP (簡單物件存取協定) 允許在分散式環境中的各點之間交換結構化資料。XML 方案會指定交換的資料。

Sun SOAP 處理目前限制爲使用點對點模型，且不保證可靠性。您可以將 SOAP 訊息包裝在 JMS 訊息中，並使用代理程式路由該訊息，利用完整功能的 Message Queue 訊息傳送以保證傳送可靠，並讓您使用主題及點對點網域。Message Queue 提供公用程式常式，讓訊息產生者將 SOAP 訊息包裝到 JMS 訊息中，訊息用戶可以用於從 JMS 訊息擷取 SOAP 訊息。

第 54 頁的「使用 SOAP 訊息」爲您提供有關 SOAP 訊息傳送處理更加詳細的說明。

管理

Message Queue 服務提供指令行工具，可用於執行下列作業：

- 啓動並配置代理程式。
- 建立並管理目標、管理代理程式連線，及管理代理程式資源。
- 在 JNDI 物件存放區中新增、列示、更新和刪除管理物件。
- 寫入和管理檔案型使用者儲存庫。
- 建立和管理用於永久儲存的 JDBC 相容資料庫。

您也可以建立 GUI 型管理主控台，以執行下列指令行功能：

- 連接至代理程式並對其進行管理。
- 建立和管理實體目標。
- 連線到物件存放區、新增物件到存放區，及管理物件。

調整 Message Queue 服務

隨著用戶端數目或連線數量的增加，您可能需要調整訊息服務以清除瓶頸或改善效能。Message Queue 訊息服務會視您的需求提供多種調整選項。爲方便起見，這些選項可分爲下列種類：

- *垂直調整*的方式是新增更多處理能力及延伸可用資源。方法是新增更多處理器或記憶體、切換至共用執行緒模型，或以 64 位元模式執行 Java VM。

  如果您使用點對點網域，可以藉由允許多個用戶存取佇列，來調整用戶端。使用這種方式，可以指定使用中和備份用戶的最大數目。負載平衡機制也會考慮用戶的目前處理能力與訊息處理率。這屬於 Message Queue 功能。(JMS 規格定義僅有一名用戶存取佇列的訊息傳送運作方式；允許多名用戶的佇列是提供者特定的運作方式。Message Queue 開發者指南提供此調整選項的詳細資訊。)

- *無狀態水平調整*的方式是使用其他代理程式，並重新發行現有的用戶端到這些代理程式。此方式易於執行，但是僅在訊息傳送作業可以劃分爲獨立的工作群組時才適用。

- *狀態水平調整*方式是將代理程式連線到*叢集*中。在代理程式叢集中，每個代理程式都會連線到叢集中的其他代理程式，亦會連線到其本機應用程式用戶端。代理程式可位於相同的主機或分散在網路中不同的主機。有關目標和用戶的資訊會複製到叢集中的所有代理程式上。目標或訂閱者的更新也會進行傳播，因此每個代理程式可以從產生者路由訊息到其直接連線到用戶的代理程式，而這些用戶是連線到叢集的其他代理程式。在使用備份用戶的情況下，如果代理程式或連線失敗，則傳送到無法存取的用戶之訊息，會轉寄至其他代理程式上的備份用戶。

  如果代理程式或連線失敗，有關永久性實體的狀態資訊 (目標和長期訂閱) 可能會不同步。例如，如果叢集代理程式失敗，且在叢集的其他代理程式上建立目標，則當第一個代理程式重新啓動時，將不知道這個新目標。若要避免此問題，可以將叢集中的某個代理程式指定爲*主代理程式*。此代理程式負責追蹤*主配置檔案*中對目標和長期訂閱的所有變更，以及負責更新叢集中暫時離線的代理程式。如需其他資訊，請參閱第 4 章「代理程式叢集」。

  若是代理程式或連線失敗，即使使用主代理程式，Message Queue 僅提供服務可用性，而不會提供資料可用性。例如，若是叢集代理程式變成無法使用，代理程式所保留的任何永久性訊息都會變成無法使用，直到代理程式回復爲止。目前確保資料可用性的唯一方法，是透過使用 SunCluster Message Queue 代理程式。在這種情況下，永久存放區會位於共用檔案系統上。如果代理程式失敗，則第二個節點上的 Message Queue 代理程式會啓動某個代理程式，來接管共用存放區。用戶端會重新連線到該代理程式，並由此繼續取得服務及存取永久性資料。

Message Queue 作爲啓用技術

Java 2 Platform, Enterprise Edition (J2EE 平台) 是一種 Java 程式設計環境中分散式元件模型的規格。J2EE 平台的一個要求是分散式元件能透過可靠的非同步訊息交換而彼此互動。此功能由 JMS 提供者提供，它有兩個作用：提供服務和支援訊息驅動 Bean (MDB)，這是一種可使用 JMS 訊息之特殊類型的 Enterprise Java Bean (EJB) 元件。

J2EE 相容應用程式伺服器必須使用指定 JMS 提供者提供的資源介面，才能使用該提供者的功能。Message Queue 提供這類資源介面。藉由使用外掛 JMS 提供者的支援，在應用程式伺服器環境中部署並執行的 J2EE 元件 (包括 MDB)，可以在這些元件間與外部 JMS 元件間交換 JMS 訊息。這提供分散式元件的強大整合功能。

如需 Message Queue 資源介面的詳細資訊，請參閱第 5 章「Message Queue 和 J2EE」。

產品版本

Message Queue 有兩個可用版本：企業版與平台版。兩個版本都提供完整的 JMS 規格執行，但每個版本都對應到不同的功能集與授權容量。

企業版在平台版新增了下列功能：

- HTTP 與 HTTPS 支援
- C 用戶端支援
- 可延伸的連線功能
- 代理程式叢集
- 每個佇列可以向不限數量的訊息用戶發送佇列。(平台版限制每個佇列傳送到三個用戶。)
- 用戶端連線容錯移轉到叢集中的其他代理程式。
- 訊息型監視 API

Message Queue 企業版依據所使用的 CPU 數量，具有不受期限的授權。授權未對多重代理程式訊息服務中的代理程式數目做出限制。

Message Queue 平台版未對代理程式所支援的用戶端連線做出數目限制。它提供基本授權或 90 天的試用授權：

- **基本授權**沒有期限限制，但不包括企業版功能。
- **90 天的試用企業授權**可讓您使用並評估所有企業版功能，不過軟體強制規定該授權只有 90 天的有效期。如需使用此授權的說明，請參閱「Message Queue 管理指南」中所述的啓動選項。

如需有關授權功能、重新發行權限，以及將平台版升級到企業版的更多資訊，請參閱「Message Queue Installation Guide」。

Message Queue 功能摘要

Message Queue 的功能遠超出了 JMS 規格的需求。這些功能可讓 Message Queue 與由大量分散式元件組成的系統整合，可在全天候的關鍵任務作業中交換數以萬計的訊息。如需這些功能的摘要，請參閱附錄 B「Message Queue 功能」。

第 2 章

用戶端程式設計模型

本章將說明 Message Queue 用戶端程式設計的基礎。涵蓋下列主題：

本章著重描述 Java 用戶端的設計及執行。總體而言，C 用戶端設計與 Java 用戶端設計基本相同。本章的最後一節將會摘要說明 Java 用戶端與 C 用戶端之間的不同。如需程式設計 Message Queue 用戶端的詳細討論，請參閱「Message Queue Developer's Guide for Java Clients」與「Message Queue Developer's Guide for C Clients」。

第 3 章「Message Queue 服務」將說明如何使用 Message Queue 服務來支援、管理及調整訊息傳送的效能。

設計與效能

Message Queue 應用程式的行爲取決於許多因素：用戶端設計、連線配置、代理程式配置、代理程式調整與資源管理等。有部份因素屬於開發者的責任，其他因素則須由管理員考量。但實際上，開發者應瞭解 Message Queue 服務的支援方式，進而調整應用程式設計；而管理員也應利用調整應用程式的機會，瞭解設計的目標。訊息傳送行爲不只可以經由重新設計而達到最佳化，同時也可透過謹慎的監視與調整達到此目的。因此，要製作優良 Message Queue 應用程式的關鍵，在於開發者與管理員必須瞭解應用程式在生命週期各階段中所能夠執行的動作，並共享期望行爲與觀測行爲之相關資訊。

訊息傳送網域

訊息傳送中介軟體可讓元件與應用程式藉由產生及使用訊息的方式進行通訊。JMS API 會定義兩種樣式或*訊息傳送網域*來管理這項通訊：*點對點訊息傳送*與*出版 / 訂閱訊息傳送*。JMS API 即在支援這些樣式。基本的 JMS 物件：連線、階段作業、產生者、用戶、目標及訊息等，皆可用來指定這兩個網域中的訊息傳送行爲。

點對點訊息傳送

在點對點網域中，訊息產生者稱爲*傳送者*，而用戶稱爲*接收者*。他們會藉由稱爲*佇列*的目標來交換訊息：傳送者會*產生*訊息到佇列，而接收者則會*使用*佇列上的訊息。

圖 2-1 所示是點對點網域中最簡易的訊息傳送作業。`MyQueueSender` 傳送 `Msg1` 到佇列目標 `MyQueue1`。然後，`MyQueueReceiver` 取得 `MyQueue1` 上的訊息。

圖 2-1 簡易的點對點訊息傳送

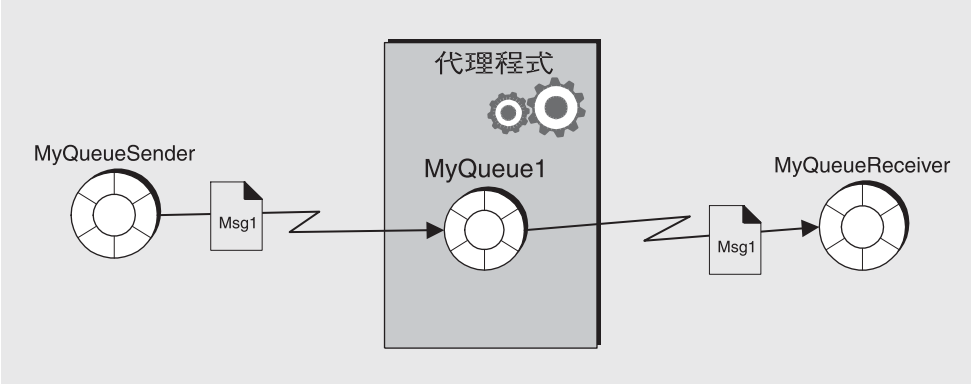
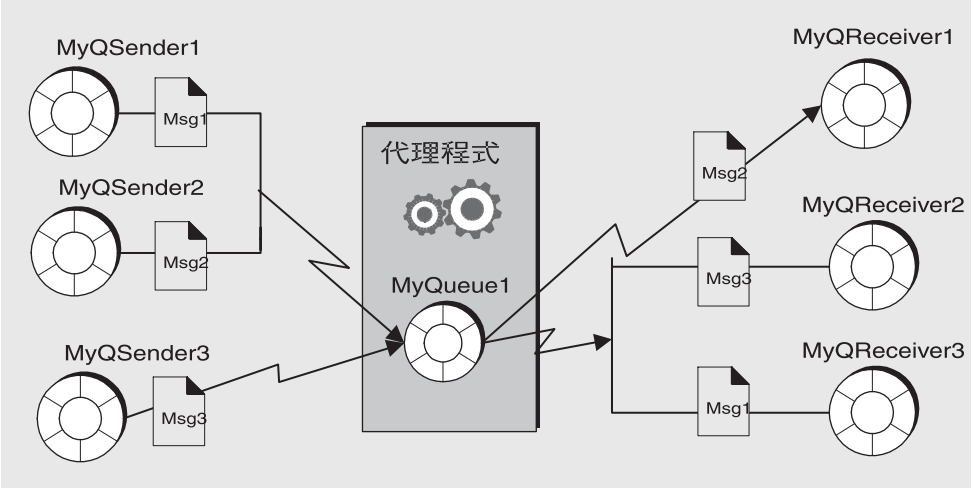


圖 2-2 所示圖片，是較爲複雜的點對點訊息傳送，說明此網域中可能發生的情況。MyQSender1 與 MyQSender2 兩位傳送者皆使用相同的連線將訊息傳送到 MyQueue1。MyQSender3 則使用另一條連線將訊息傳送到 MyQueue1。在接收端上，MyQReceiver1 會使用 MyQueue1 上的訊息，而 MyQReceiver2 與 MyQReceiver3 則會共用同一條連線，以使用 MyQueue1 上的訊息。

圖 2-2 複雜的點對點訊息傳送



這個較複雜的圖片解釋了許多有關點對點訊息傳送的其他資訊。

- 多位產生者可以傳送訊息到相同的佇列上。產生者可選擇共用同一條連線或不同的連線，但都皆可存取相同的佇列。
- 如有多位接收者，則可使用相同佇列中的訊息，但每一則訊息只可供一位接收者使用。因此，`Msg1`、`Msg2` 與 `Msg3` 會由不同的接收者所使用。(這屬於 Message Queue 的延伸。)
- 接收者可選擇共用同一條連線或不同的連線，但都皆可存取相同的佇列。(這屬於 Message Queue 的延伸。)
- 傳送者與接收者不會受到時間的影響：當用戶端將訊息傳出時，不論其是否正在執行中，接收者皆能擷取訊息。
- 執行階段可動態新增或刪除傳送者與接收者，以便於訊息傳送系統視需要進行擴大或縮小。
- 訊息會依照傳送的順序放置於佇列上，但其使用順序則視訊息到期日、訊息優先權，以及使用訊息時，是否使用了選擇器而定。

點對點模型具有許多優點：

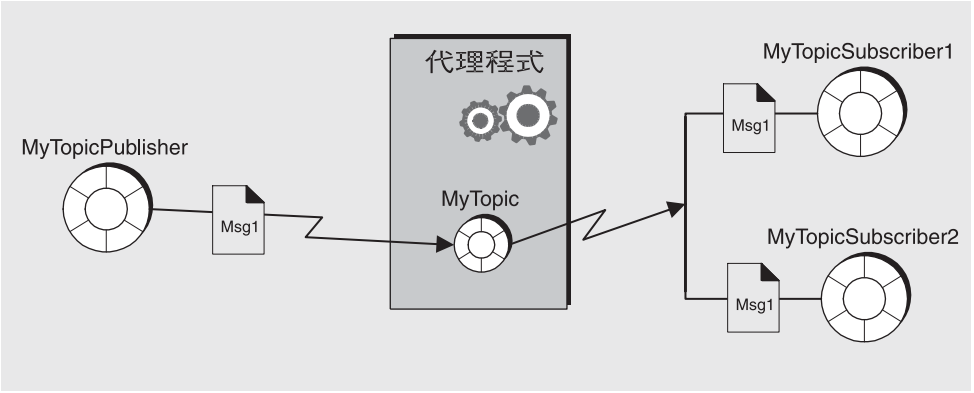
- 由於同一個佇列上的訊息可供多名接收者使用，因此若接收訊息的順序不是很重要，可以在訊息使用負載上達到平衡。(這屬於 Message Queue 的延伸。)
- 以佇列爲目標的訊息一律會予以保留，而不論其是否有接收者。
- Java 用戶端可使用佇列瀏覽器物件來檢視佇列的內容。其可根據這項檢視所取得的資訊來使用訊息。亦即，使用模型雖然多是 FIFO (先進先出)，但用戶若是透過訊息選擇器而得知所需的訊息，便可以使用不在佇列開頭的訊息。管理用戶端也可以使用佇列瀏覽器來監視佇列內容。

出版 / 訂閱訊息傳送

在出版 / 訂閱網域中，訊息產生者稱爲*出版者*，而訊息用戶則稱爲*訂閱者*。他們會藉由稱爲*主題*的目標來交換訊息：出版者會產生主題相關的訊息，而訂閱者則會*訂閱*主題，並*使用*主題所提供的訊息。

圖 2-3 所示是出版 / 訂閱網域中的簡易訊息傳送作業。MyTopicPublisher 將 Msg1 出版到目標 MyTopic 上。然後，MyTopicSubscriber1 與 MyTopicSubscriber2 各自從 MyTopic 接收一份 Msg1。

圖 2-3 簡易的出版 / 訂閱訊息傳送

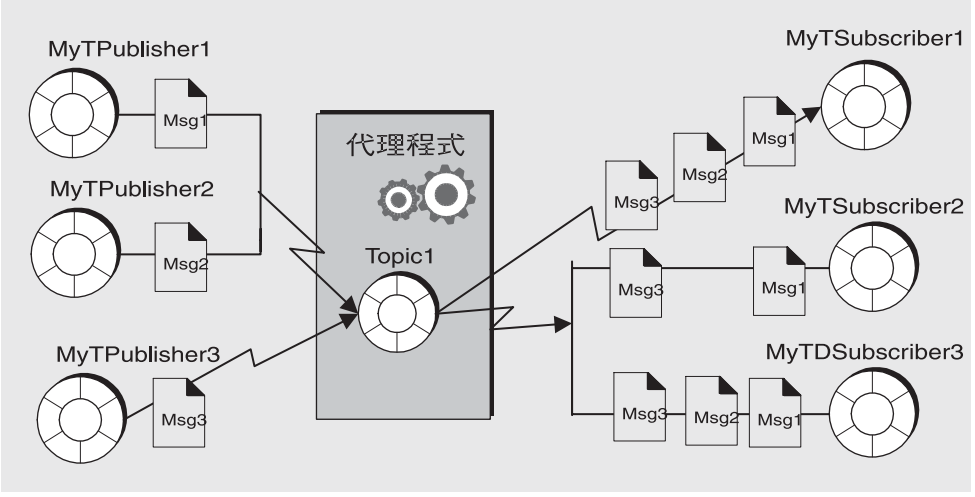


出版 / 訂閱模型中不一定要有多位訂閱者；此圖中之所以顯示兩名訂閱者，表明此網域可讓您廣播訊息。某項主題的所有訂閱者皆可收到任何出版到該主題的訊息。

訂閱者可以是非長期或長期的訂閱者。代理程式會保留所有使用中訂閱者的訊息，但若是長期訂閱者，便只會保留非使用中訂閱者的訊息。

圖 2-4 所示圖片，是較爲複雜的出版 / 訂閱訊息傳送，說明此樣式中可能發生的情況。數名產生者出版訊息到 Topic1 目標。數名訂閱者使用 Topic1 目標上的訊息。除非訂閱者利用選擇器來篩選訊息，否則每位訂閱者都會取得所有出版到所選主題的訊息。圖 2-4 MyTSubscriber2 中已將 Msg2 篩選在外。

圖 2-4 複雜的出版 / 訂閱訊息傳送



這個較複雜的圖片解釋了許多有關出版 / 訂閱訊息傳送的其他資訊。

- 多位產生者可以出版訊息到相同的主題上。產生者可選擇共用同一條連線或不同的連線，但都皆可存取相同的主題。
- 多名訂閱者可以使用相同主題所提供的訊息。除非訂閱者使用選擇器將某些訊息篩選在外，或訊息在被使用之前即已過期，否則訂閱者會擷取所有出版到主題上的訊息。
- 訂閱者可選擇共用同一條連線或不同的連線，但都皆可存取相同的主題。
- 長期訂閱者可以是使用中或非使用中的訂閱者。代理程式會保留非使用中訂閱者的訊息。
- 執行階段可動態新增或刪除出版者與訂閱者，以便於訊息傳送系統視需要進行擴大或縮小。
- 訊息會依照傳送的順序出版到主題上，但其使用順序則視訊息到期日、訊息優先權，以及使用訊息時，是否使用了選擇器而定。
- 出版者與訂閱者會受到時間的影響：主題訂閱者只可使用訂閱建立之後所出版的訊息。

出版 / 訂閱模型主要的優點在可以將訊息廣播給多名訂閱者。

網域專用及統一的 API

JMS API 可定義讓您用來執行點對點或出版 / 訂閱網域的介面與類別。這些就是表 2-1 中的第 2 欄與第 3 欄所顯示的*網域專用* API。JMS API 另外還可定義*統一*網域，讓您進行一般訊息傳送用戶端的程式設計。這類用戶端的行為，取決於它產生訊息與使用訊息的目標類型。若目標為佇列，則訊息傳送的行為會依據點對點樣式；若目標為主題，則訊息傳送的行為會依據出版 / 訂閱樣式。

表 2-1 JMS 程式設計網域及物件

基本類型 (統一網域)	點對點網域	出版 / 訂閱網域
Destination (Queue 或 Topic)*	Queue	Topic
ConnectionFactory	QueueConnectionFactory	TopicConnectionFactory
Connection	QueueConnection	TopicConnection
Session	QueueSession	TopicSession
MessageProducer	QueueSender	TopicPublisher
MessageConsumer	QueueReceiver	TopicSubscriber

* 取決於程式設計的方法，您必須指定特定的目標類型。

統一網域已在 JMS 版本 1.1 中說明過。如果您需要符合較早的 JMS 1.02b 規格，可以使用網域專用的 API。使用網域專用的 API 亦可提供清晰的程式設計介面，避免出現特定類型的程式設計錯誤：例如，為佇列目標建立長期訂閱。然而，網域專用 API 的缺點是您不能在同一作業事件或同一作業階段中結合點對點和出版 / 訂閱作業。如果您需要執行這些動作，應該選擇統一網域的 API。如需結合兩個網域的範例，請參閱第 48 頁的「請求回覆樣式」。

程式設計物件

用於執行 JMS 訊息傳送的物件，在各程式設計網域間仍維持相同的本質：連線工廠、階段作業、產生者、用戶、訊息與目標等。這些物件會顯示在圖 2-5 中。此圖將從連線工廠物件開始，完整說明物件導出的方式。

連線工廠物件與目標物件會顯示在物件存放區中。強調這些物件通常會被視為管理物件而進行建立、配置及管理。假設本章中的連線工廠與目標皆是利用管理方式建立 (而不是利用程式設計所建立)。

圖 2-5　JMS 程式設計物件

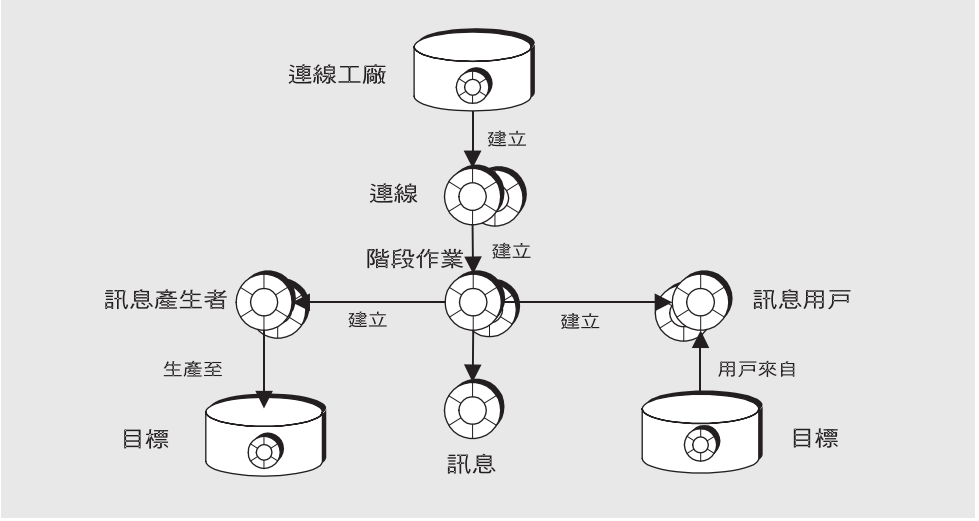


表 2-2 總結了傳送與接收訊息時的必要步驟。請注意，傳送者與接收者的步驟 1 與步驟 3 至 6 皆相同。

表 2-2　產生與使用訊息

產生訊息	使用訊息
1. 管理員建立受連線工廠所管理的物件。	
2. 管理員建立實體目標與參照此目標的管理物件。	
3. 用戶端透過 JNDI 查詢取得連線工廠物件。	
4. 用戶端透過 JNDI 查詢取得目標物件。	
5. 用戶端建立連線，並設定此連線專有的特性。	
6. 用戶端建立階段作業，並設定用以控制訊息傳送可靠性的特性。	
7. 用戶端建立訊息產生者。	用戶端建立訊息用戶。
8. 用戶端建立訊息。	用戶端啓動連線。
9. 用戶端傳送訊息。	用戶端接收訊息。

下列幾節將說明產生者與用戶所使用的物件：連線、階段作業、訊息與目標。我們將藉由描述訊息的產生與使用，來結束對 JMS 物件的討論。

連線工廠與連線

用戶端會使用連線工廠物件 (ConnectionFactory) 來建立連線。連線物件 (連線) 代表用戶端與代理程式之間使用中的連線。它會使用依預設啓動的基本連線服務，或使用管理員爲該用戶端專門啓動的基本連線服務。

通訊資源的分配以及用戶端的認證均在建立連接時進行。相對而言，這是一個十分重要的物件，大多數用戶端均使用單一連線來進行所有的訊息傳送。連線可支援同時使用：多名產生者與用戶共用相同的連線，而無數量上的限制。

建立連線工廠時，藉由設定其特性，可以對所有源自此連線工廠的連線配置其行爲。對於 Message Queue，這些特性會指定下列資訊：

- 代理程式所在主機的名稱、所需的連線服務，以及用戶端存取該服務時使用的連接埠。
- 連線失敗時應以何種方式處理代理程式的自動重新連線。(這項功能會在連線中斷時，將用戶端重新連線到相同 (或不同) 的代理程式上。但不一定會進行資料容錯移轉：重新連線到不同的代理程式時，可能會遺失永久性訊息與其他狀態資訊。)
- 需要代理程式追蹤其長期訂閱之用戶端的 ID。
- 嘗試進行連線之用戶的預設名稱與密碼。連線時若未指定密碼，可以利用這項資訊認證用戶並進行作業授權。
- 對於不需考量可靠性的用戶端，是否需停用代理程式確認。
- 如何管理代理程式與用戶端執行階段之間的控制流程及有效負載訊息。
- 如何處理佇列瀏覽。(僅限 Java 用戶端。)
- 是否應覆寫特定的訊息標頭欄位。

您可以從指令行覆寫連線工廠特性，以啓動用戶端應用程式。您也可以透過設定連線特性，而覆寫任何指定連線的特性。

您可以使用連線物件來建立階段作業物件、設定異常偵聽程式，或取得 JMS 的版本與提供者資訊。

階段作業

若連線代表用戶端與代理程式之間的通訊通道，則階段作業便代表用戶端與代理程式之間的單一對話。您主要會使用階段作業物件建立訊息、訊息產生者與訊息用戶。建立階段作業時，必須透過多個確認選項或透過作業事件來配置可靠的傳送作業。如需更多資訊，請參閱第 50 頁的「可靠的訊息傳送」。

根據 JMS 規格，階段作業是指用來生產及使用訊息的單執行緒環境。您可以爲單一階段作業建立多名訊息產生者與用戶；但若要連續加以使用，則會受到限制。Java 用戶端與 C 用戶端的執行緒在執行上略有不同。如需執行緒執行與限制的相關資訊，請參閱適當的開發者指南。

您也可以使用階段作業物件來執行下列作業：

- 爲並未使用管理物件來定義目標的用戶端建立及配置目標。
- 建立及配置暫時性主題與佇列，這些將會用爲請求 - 回覆樣式的一部份。請參閱第 48 頁的「請求回覆樣式」。
- 支援作業事件處理。
- 定義產生或使用訊息的順序。
- 爲非同步用戶序列化訊息偵聽程式的執行工作。
- 建立佇列瀏覽器。(僅限 Java 用戶端。)

訊息

訊息組成包括下列三個部份：標頭、特性和內文。您必須瞭解此結構，才能編寫正確的訊息，以及配置特定的訊息傳送行爲。

訊息標頭

每個 JMS 訊息都需要標頭。標頭中含有十個預先定義的欄位，會列示在表 2-3 中，並加以說明。

表 2-3 JMS 定義的訊息標頭

欄頭屬性	說明
JMSDestination	指定訊息傳送所至之目標物件的名稱。(由提供者設定。)
JMSDeliveryMode	指定訊息是否具有永久性。(預設是由提供者所設定，或由產生者或個別訊息的用戶端明確設定。)
JMSExpiration	指定訊息的逾期時間。(預設是由提供者所設定，或由產生者或個別訊息的用戶端設定。)
JMSPriority	指定介於 0 (低) 與 9 (高) 之間的訊息優先權。(預設是由提供者所設定，或由產生者或個別訊息的用戶端明確設定。)
JMSMessageID	指定提供者爲環境的唯一訊息 ID。(由提供者設定。)
JMSTimestamp	指定提供者接收訊息的時間。(由提供者設定。)

表 2-3　JMS 定義的訊息標頭 (續)

標頭欄位	設定者
JMSCorrelationID	供用戶端定義和訊息間一致性的值。(必要時由用戶端設定。)
JMSReplyTo	指定用戶傳送回覆所至的目標。(必要時由用戶端設定。)
JMSType	可由訊息選擇器評估的值。(必要時由用戶端設定。)
JMSRedelivered	指定訊息是否已傳送但尚未確認。(由提供者設定。)

如您在閱讀此表格時所見，訊息標頭欄位具有多種用途：識別訊息、配置訊息路由、提供訊息處理的相關資訊等。

JMSDeliveryMode 是最重要的欄位之一，可決定訊息傳送的可靠性。此欄位可指出訊息是否具有永久性。

- *永久性訊息*保證能夠傳送出去，並順利使用一次。永久性訊息不會因訊息服務失敗而遺失。
- *非永久性訊息*最多能傳送一次。非永久性訊息則可能因訊息服務失敗而遺失。

有些訊息標頭欄位是由提供者設定 (代理程式或用戶端執行階段)，有些則是由用戶端設定。訊息產生者必須配置標頭值，才能取得特定的訊息傳送行為；而訊息用戶則必須讀取欄位值，才能夠瞭解傳閱訊息的方式，以及訊息可能需要的進一步處理。

標頭欄位 (JMSDeliveryMode、JMSExpiration 與 JMSPriority) 可設定為三種不同的層級：

- 適用於源自連線工廠之每個連線發出的訊息。
- 適用於產生的各項訊息。
- 適用於特定訊息產生者發出的各項訊息。

這些欄位若不只設在一個層級上，則連線工廠的設定值便會覆寫個別訊息的設定值；特定訊息的設定值則會覆寫訊息產生者的設定值。

訊息標頭欄位的常數名稱則會隨所用語言而有所不同。如需相關資訊，請參閱「Message Queue Developer's Guide for Java Clients」或「Message Queue Developer's Guide for C Clients」。

訊息特性

訊息中也可能含有可選的標頭欄位 (稱爲*特性*)，會以特性名稱與特性值成對指定。「特性」可讓用戶端與提供者延伸訊息標頭，並可包含任何有助於用戶端或提供者識別及處理訊息的資訊。訊息特性可讓接收用戶端要求僅傳送符合指定條件的訊息。例如，使用用戶端可指定只需要紐澤西州之兼差員工的薪資訊息。提供者將不會傳送任何不符指定條件的訊息。

JMS 規格可定義九種標準特性。這些特性部份是由用戶端設定，部份則是由提供者設定。其名稱會以保留字元「JMSX」爲開頭。用戶端或提供者可使用這些特性來決定訊息的傳送者、訊息狀態，以及傳送訊息的頻率與時機。這些特性對提供者在發送訊息及提供診斷資訊時非常有用。

Message Queue 也可定義訊息特性，以識別壓縮訊息及訊息無法傳送時的處理方式。如需相關資訊，請參閱「Message Queue Developer's Guide for Java Clients」。

訊息內文

訊息內文中含有用戶端所要交換的資料。

JMS 訊息的類型可決定內文所要涵括的項目，以及用戶應以何種方式進行處理，如表 2-4 所指定。階段作業物件中包含各種訊息內文類型的建立方法。

表 2-4 訊息內文類型

類型	說明
StreamMessage	訊息內文包含 Java 原始值的串流。它被依序寫入及讀取。
MapMessage	訊息內文包含名稱 - 值對的組合。項目的順序未加以定義。
TextMessage	訊息內文包含 Java 字串，如XML 訊息。
ObjectMessage	訊息內文包含序列化的 Java 物件。
BytesMessage	訊息內文包含未解譯位元組的串流。

Java 用戶端可設定特性讓用戶端執行階段壓縮所要傳送的訊息內文。位於用戶端 Message Queue 上的執行階段可在傳送郵件前加以壓縮。

產生訊息

訊息產生者可在連線與階段作業的環境中傳送或出版訊息。產生訊息的方式並不複雜：用戶端使用訊息產生者物件 (MessageProducer) 將訊息傳送至實體目標 (在 API 中即爲目標物件)。

建立產生者時，可以指定所有產生者訊息的預設傳送目標。您也可以指定管理永久性、優先權與存在時間之訊息標頭欄位的預設值。這些預設值在指定後即可供該產生者所發出的各項訊息使用；除非您在傳送時指定了替代目標，或爲指定訊息的標頭欄位設定了替代值，而覆寫了預設值。

訊息產生者也可以設定 JMSReplyTo 訊息標頭欄位來執行請求 - 回覆樣式。如需相關資訊，請參閱第 48 頁的「請求回覆樣式」。

使用訊息

訊息用戶可在連線與階段作業的環境中接收訊息。用戶端會使用訊息用戶物件 (MessageConsumer) 接收來自特定實體目標 (在 API 中即爲目標物件) 的訊息。

下列三項因素會影響代理程式傳送訊息給用戶的方式：

- 使用屬於同步或非同步
- 是否使用選擇器篩選內送的訊息
- 若是從主題目標使用訊息，訂閱者是否爲長期訂閱者。

用戶所需的可靠性程度，是另一個會影響訊息傳送與用戶端設計的主要因素。請參閱第 50 頁的「可靠的訊息傳送」。

同步與非同步用戶

訊息用戶可以支援同步或非同步的訊息使用。

- 同步使用表示用戶明確請求傳送訊息並隨後使用訊息。

  同步用戶會依據請求訊息的方式不同，選擇在訊息傳入前一直等待訊息 (無限期)；或在指定的時間內等待訊息；或在沒有訊息可使用時立即傳回。(「已使用」表示物件可立即供用戶端使用。已順利傳送但代理程式尚未完成處理的訊息則無法使用。)

- 非同步使用表示訊息自動傳送到已爲用戶註冊的訊息偵聽程式物件 (MessageListener)。當階段作業執行緒呼叫訊息偵聽程式物件的 onMessage() 方法時，用戶端即可使用該訊息。

使用選擇器篩選訊息

訊息用戶可以使用訊息選擇器，令訊息服務僅將符合選取條件的訊息傳送至訊息用戶。您可以在建立用戶時指定此條件。

選擇器會使用類似 SQL 的語法來比對訊息特性。例如，

```
color = 'red'
```

```
size > 10
```

Java 用戶端也可以在瀏覽佇列時指定選擇器，以讓您檢視可供使用的訊息。

使用長期訂閱者

您可以使用階段作業物件建立對主題的長期訂閱者。即使訂閱者處於非使用中的狀態，代理程式仍會保留這些訂閱者類型的訊息。

由於代理程式必須維護訂閱者的狀態，並在訂閱者重新啓動時繼續執行訊息傳送，因此代理程式在指定的訂閱者傳入與傳出時，都必須能夠加以識別。訂閱者的身份是由建立訂閱者的連線 ClientID 特性，以及您在建立訂閱者時所指定的訂閱者名稱建構而成。

請求回覆樣式

您可以將產生者與用戶結合在同一條連線中 (使用統一 API 時，甚至可結合在同一個階段作業中)。此外，JMS API 可讓您使用暫時的目標，對您的訊息傳送作業執行請求 - 回覆樣式。

訊息產生者必須執行下列動作，才能設定請求 - 回覆樣式：

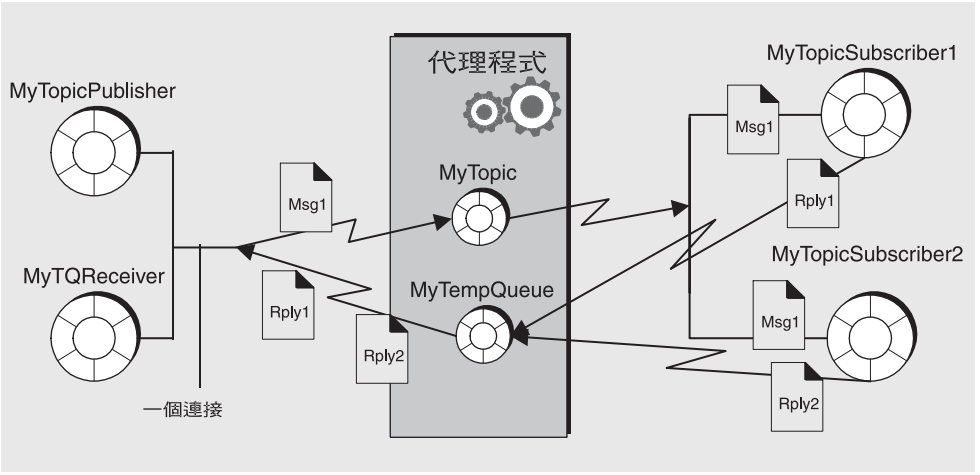
1. 建立可供用戶傳送回覆的暫時目標。
2. 在要傳送的訊息中，將訊息標頭的 JMSReplyTo 欄位設爲該暫時目標。

當訊息用戶處理訊息時，會檢查訊息的 JMSReplyTo 欄位，以決定是否需要回覆，並將回覆傳送到指定的目標。

請求 - 回覆機制除了免去產生者設定回覆目標之管理物件的麻煩之外，還簡化了用戶回應請求的程序。當產生者在處理之前必須確定請求是否已處理時，此樣式非常有用。

圖 2-6 說明請求 / 回覆樣式將訊息傳送到主題，並從暫時佇列中接收回覆。

圖 2-6 請求 / 回覆樣式



如圖所示，MyTopicPublisher 會產生 Msg1，並傳送到目標 MyTopic 上。MyTopicSubsriber1 與 MyTopicSubscriber2 會在收到訊息後，傳送回覆到 MyTempQueue，而 MyTQReceiver 則會由此處進行擷取。應用程式如須出版報價資訊給大量用戶端，且將其 (回覆) 訂單排入佇列依序處理，此樣式可能非常有用。

暫時目標的存在情況取決於建立它們的連線。任何產生者都可以對暫時目標進行傳送，但只有透過建立目標之相同連線建立的用戶，才可以存取暫時目標。

由於請求 / 回覆樣式是在建立暫時目標時決定，因此不應在下列情況下使用此樣式：

- 若預期建立暫時目標的連線可能會在您傳送回覆之前終止。
- 若必須傳送永久性訊息到暫時目標。

可靠的訊息傳送

訊息傳送會在兩個躍點上執行：第一個躍點會將訊息從產生者傳送到代理程式上的實體目標；第二個躍點則會將訊息從該目標傳送給用戶。因此，訊息可能會在這三個階段之一中遺失：當訊息位於傳送至代理程式的躍點上時；當訊息位於代理程式記憶體中，而代理程式碰巧失敗時；當訊息位於代理程式傳送給用戶的躍點上時。可靠的傳送可確保傳送不會在上述任何一個階段中失敗。由於非永久性訊息一律會隨著代理程式失敗而遺失，因此可靠的傳送僅適用於永久性訊息。

有兩種機制可確保傳送的可靠性：

- 用戶端可使用確認或作業事件確保訊息的順利產生與使用。
- 代理程式可將訊息儲存在永久存放區中；如此一來，即使代理程式在訊息使用之前即失敗，仍可擷取已儲存的訊息副本，並重試該作業。

以下各節描述確保可靠性的兩個方面。

確認

*確認*是在用戶端與訊息服務之間傳送的訊息，可確保郵件傳送的可靠性。產生者與用戶的確認用法不盡相同。

在訊息產生的情況下，代理程式會確認它已收到訊息，並將其放置在它的目標上而永久存放。產生者的 `send()` 方法會暫停運作，直到收到此確認爲止。在傳送永久性訊息時，會對用戶端自動進行這些確認作業。

在訊息使用的情況下，必須在用戶端確認已收到目標傳送的訊息，並加以使用之後，代理程式才能夠刪除該目標上的該項訊息。JMS 指定不同的確認模式來代表不同程度的可靠性。

- 在 `AUTO_ACKNOWLEDGE` 模式中，階段作業自動確認用戶端使用的每個訊息。階段作業執行緒會暫停運作，以等待代理程式確定已處理每個使用訊息用戶端的確認。
- 在 `CLIENT_ACKNOWLEDGE` 模式中，用戶端會在一或多項訊息已被使用之後，呼叫訊息物件的 `acknowledge()` 方法，以進行明確的確認。這會使階段作業確認自前次方法呼叫之後，由階段作業所使用的所有訊息。階段作業執行緒會暫停運作，以等待代理程式確定已處理用戶端的確認。

  Message Queue 提供一種方法讓用戶端只針對某項訊息接收與否進行確認，藉以延伸此模式。
- 在 `DUPS_OK_ACKNOWLEDGE` 模式中，階段作業會在使用 10 個訊息後確認。此模式中因爲不需要使用代理程式確認，因此階段作業執行緒不會暫停來等待代理程式的確認。雖然此模式可確保不會遺失任何訊息，但無法確保不會收到重複的訊息，這是其名稱：`DUPS_OK` 所致。

對於效能高於可靠性的用戶端而言，Message Queue 服務提供 NO_ACKNOWLEDGE 方法來延伸 JMS API。在此模式中，代理程式不會追蹤用戶端確認，因此無法確保使用用戶端能夠順利處理訊息。選擇此模式可讓您在將非永久性訊息傳送給非長期訂閱者時，都保有較佳的效能。

作業事件

*作業事件*是將一或多項訊息之產生及 (或) 使用，歸類爲不可分割單位的方法。先前所討論的用戶端與代理程式確認，亦屬於作業事件。在此情況下，用戶端執行階段與代理程式確認會隱含地在作業事件層級上執行。確定作業事件時，代理程式回應會被自動傳送。

階段作業可配置成*作業事件*，而 JMS API 則提供用於啓動、確認或回復作業事件的方法。

在作業事件中生產或使用訊息時，訊息服務會追蹤各種傳送和接收過程，並在 JMS 用戶端發出確定作業事件的呼叫時完成這些作業。如果作業事件中特定的傳送或接收作業失敗，則會出現異常。用戶端程式碼可以透過忽略異常、重試作業或回轉整個作業事件來處理異常。在作業事件確定後，將完成所有的作業。在作業事件回轉後，將取消所有成功的作業。

作業事件的範圍一律爲單一階段作業。亦即，可以將單一階段作業的環境中執行的一個或多個產生者或用戶作業歸類爲一個作業事件。由於作業事件僅能是單一階段作業，因此不能有同時包含訊息生產和訊息使用的端對端作業事件。

JMS 規格也支援*分散式*作業事件。亦即，訊息的生產和使用可作爲較大的分散式作業事件的一部分，其中包括涉及其他資源管理員 (如資料庫系統) 的作業。作業事件管理員 (如 Java Systems Application Server 所提供) 必須能夠支援分散式作業事件。

在分散式作業事件中，分散式作業事件管理員使用 Java Transaction API (JTA)、XA 資源 API 規格中定義的兩階段確定協定，追蹤和管理多個資源管理員 (如訊息服務和資料庫管理員) 執行的作業。在 Java 中，JTA 規格描述了資源管理員和分散式作業事件管理員之間的互動。

支援分散式作業事件表示，訊息傳送用戶端可以透過 JTA 定義的 XAResource 介面參與分散式作業事件。此介面定義了實施兩階段確定的許多方法。當用戶端進行 API 呼叫時，JMS 訊息服務僅與 Java Transaction Service (JTS) 提供的分散式作業事件管理員協作，來追蹤分散式作業事件中的各種傳送和接收作業以及作業事件狀態，並完成訊息傳送作業。

處理本機作業事件時，用戶端可以藉由忽略異常、重試作業或回轉整個分散式作業事件來處理異常。

永久性儲存

在另一方面，可靠性也必須確保代理程式在將永久性訊息傳送至用戶之前，訊息不會遺失。這表示當訊息送達實體目標時，代理程式必須將其存放在永久的資料存放區。代理程式若因故失敗，可於稍後恢復該訊息，並將其傳送給適當的用戶。

代理程式也必須永久儲存長期訂閱。否則代理程式若是失敗，便無法將訊息傳送給訊息抵達主題目標之後，成爲使用中狀態的長期訂閱者。

需要保證訊息傳送的訊息應用程式必須將訊息指定爲永久性訊息，並傳送至主題目標的長期訂閱或佇列目標。

第 3 章「Message Queue 服務」將說明由 Message Queue 服務提供的預設訊息存放區，以及管理員應如何設定及配置替代庫。

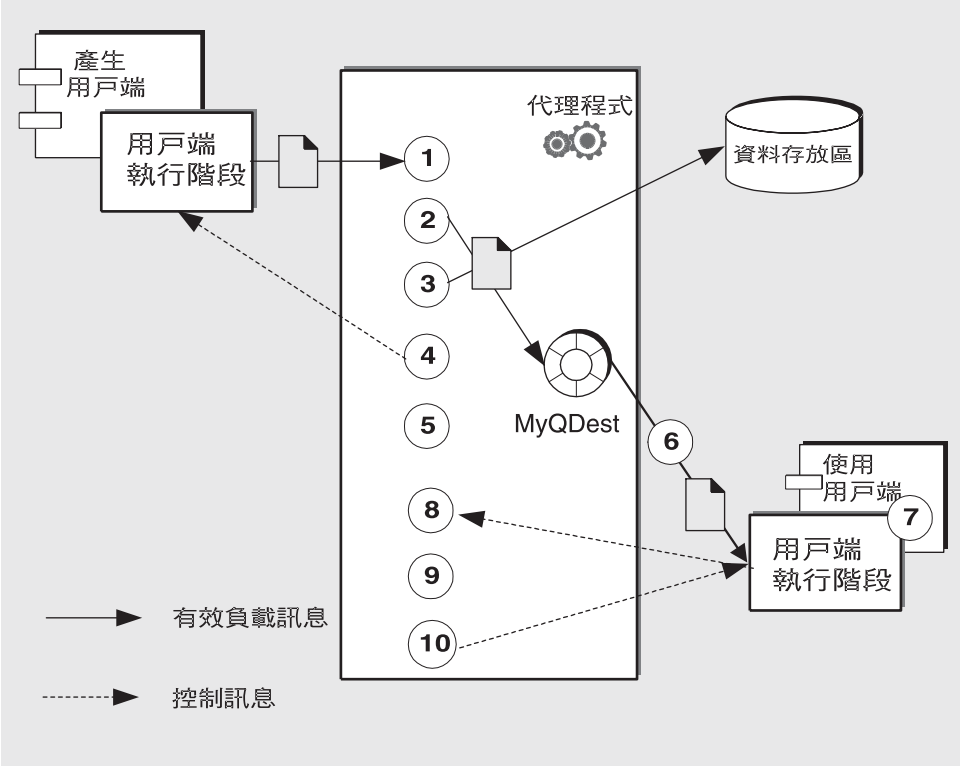
訊息在系統中的行程

本節將藉由總結目前已介紹的內容，說明如何使用 Message Queue 服務，將訊息從產生者傳送給用戶。爲了讓您有完整的概念，我們將加入更多的詳細資訊：系統在傳遞程序中所處理的訊息分成兩類：

- **有效負載訊息**：產生者傳送給用戶的訊息。
- **控制訊息**：在代理程式與用戶端執行階段之間傳遞的私人訊息，可確保有效負載訊息順利傳送，並可控制連線間的訊息流量。

如需訊息傳送的說明，請參閱圖 2-7。

圖 2-7　訊息傳送步驟



安全傳送永久性訊息時的訊息傳送步驟如下所示：

訊息生產

1. 用戶端執行階段透過連線，將訊息從訊息生產者傳送至代理程式。

訊息處理和訊息發送

2. 代理程式從連線讀取訊息，並將其放入適當的目標中。
3. 代理程式將 (永久性) 訊息放入資料存放區中。
4. 代理程式向訊息生產者的用戶端執行階段確認收到訊息。
5. 代理程式判斷訊息的發送情況。
6. 代理程式將訊息從目標寫入適當的連線，並爲用戶在上面標記唯一的識別碼。

訊息使用

7. 訊息用戶的用戶端執行階段從連線傳送訊息至訊息用戶。
8. 訊息用戶的用戶端執行階段向代理程式確認訊息的使用。

訊息結束

9. 代理程式處理用戶端確認，並在收到所有確認後刪除 (永久性) 訊息。
10. 代理程式向用戶的用戶端執行階段確認示已處理用戶端確認。

若管理員從目標中刪除訊息，或管理員移除或重新定義長期訂閱，而導致主題目標中的訊息尚未傳送即被移除，則該訊息便可能會在使用之前，已由代理程式移除。在其他情況下，您可以讓代理程式將訊息儲存在名爲*停用訊息佇列*的特殊目標中，而不要將其移除。當訊息因爲過期、記憶體限制而被移除，或在傳送時因用戶端發生異常而失敗時，會置於停用訊息佇列中。將訊息儲存在停用訊息佇列中可讓您進行系統的疑難排解，並在特定情況下恢復訊息。

使用 SOAP 訊息

SOAP (請參閱第 30 頁的「Java 用戶端的 SOAP 支援」) 可讓您在分散式環境中的兩個點之間交換結構化資料 (由 XML 機制所指定)。Sun 的 SOAP 目前不支援可靠的 SOAP 訊息傳送，亦不支援 SOAP 訊息的出版。必要時，您可以使用 Message Queue 服務取得可靠的 SOAP 訊息傳送，以出版 SOAP 訊息。Message Queue 服務無法直接傳送 SOAP 訊息，但可讓您將 SOAP 訊息納入 JMS 訊息中，像一般的 JMS 訊息一樣產生及使用這些訊息，並從 JMS 訊息中取出 SOAP 訊息。

Message Queue 透過下列兩種套裝軟體提供 SOAP 支援：javax.xml.messaging 與 com.sun.messaging.xml。您可以使用在這些程式庫中所提供的類別來接收 SOAP 訊息，以將 SOAP 訊息納入 JMS 訊息中，以及取出 JMS 訊息中的 SOAP 訊息。J2EE 平台上提供的套裝軟體 java.xml.soap，可讓您用來組合及分解 SOAP 訊息。

若要獲取可靠的 SOAP 訊息傳送，必須執行下列動作

1. 使用 java.xml.soap 套裝軟體中所定義的物件來建置 SOAP 訊息，或使用 javax.xml.messaging 套裝軟體中所定義的 Servlet 來擷取 SOAP 訊息，或使用 JAX-RPC 等 Web 服務來擷取 SOAP 訊息。
2. 使用 MessageTransformer 公用程式將 SOAP 訊息轉換爲 JMS 訊息。
3. 將 JMS 訊息傳送到所需的目標。
4. 以非同步或同步的方式使用 JMS 訊息。
5. 使用 JMS 訊息後，利用 MessageTransformer 公用程式將其轉換成 SOAP 訊息。
6. 使用 SAAJ API (定義於 java.xml.soap 套裝軟體) 分解 SOAP 訊息。

如需 SOAP 訊息及其處理的詳細資訊，請參閱「Message Queue Developer's Guide for Java Clients」。

Java 用戶端與C 用戶端

Message Queue 爲其訊息傳送服務提供了 C API，可讓傳統的 C 應用程式與 C++ 應用程式加入 JMS 型的訊息傳送。

JMS 程式設計模型是 Message Queue C 用戶端設計的基礎。「Message Queue Developer's Guide for C Clients」說明了 C 資料類型與函數執行此模型的方式。

C 介面與 Java 介面皆支援下列功能：

- 出版 / 訂閱點對點連線
- 同步與非同步接收
- CLIENT、AUTO 與 DUPS_OK 確認模式
- 本機作業事件
- 階段作業恢復
- 暫時主題與佇列

- 訊息選擇器

但請務必瞭解，Java Message Service 規格僅爲 *Java* 用戶端的標準；因此 C Message Queue API 僅適用於 Message Queue 提供者，而無法用於其他 JMS 提供者。其他 JMS 提供者無法處理包含 C 用戶端的訊息傳送應用程式。

C 介面不支援下列功能：

- 使用管理物件
- 對映、串流或物件訊息類型
- 用戶型流量控制
- 佇列瀏覽器
- JMS 應用程式伺服器設備 (ConnectionConsumer、分散式作業事件)
- 接收或傳送 SOAP 訊息
- 接收或傳送壓縮的 JMS 訊息
- 自動重新連線或故障移轉，這些功能可在連線失敗時，讓用戶端執行階段自動重新連線到代理程式
- `NO_ACKNOWLEDGE` 模式

第 3 章

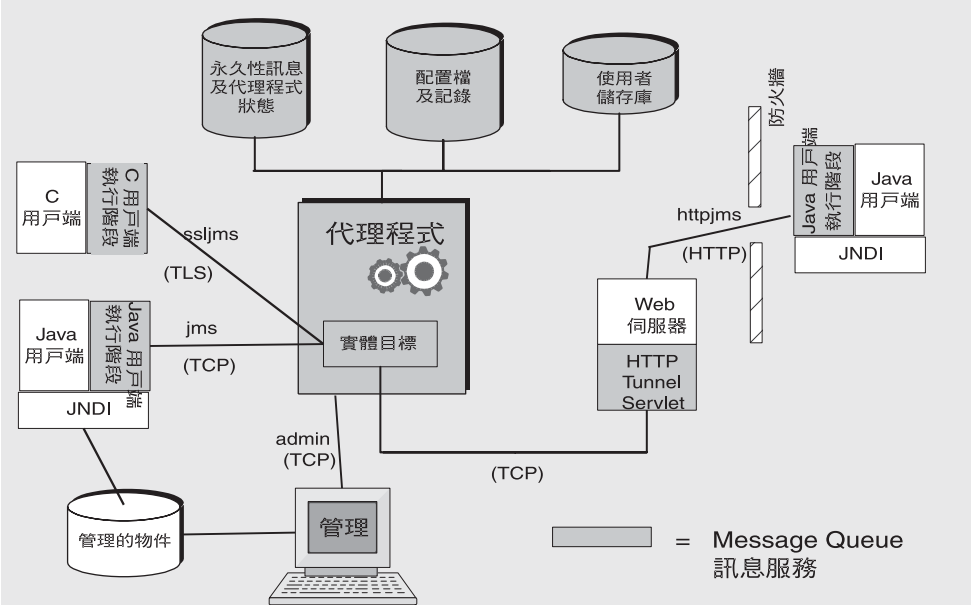
Message Queue 服務

Message Queue 用戶端效能取決於用戶端的設計以及您配置和管理 Message Queue 服務的方式。本章將更詳細地說明第一章中介紹的 Message Queue 服務，檢查服務的元件，介紹用以配置這些元件的工具，並總結在不同環境中管理訊息服務所需的工作。本章涵蓋以下小節：

元件服務

圖 3-1 顯示了 Message Queue 服務。第 2 章會描述程式設計模型，以及用戶端如何使用 Java 和 C API 與用戶端執行階段 (存取用戶端應用程式的一部份訊息服務) 互動。本章側重介紹存取管理員的訊息服務之元件及服務。

圖 3-1 Message Queue 服務



您可以透過設定代理程式特性來控制 Message Queue 服務。這些特性根據某個特性所影響的服務或代理程式元件，劃分爲多個種類。代理程式服務包括：

- **連線服務**：管理代理程式與其用戶端之間的實體連線，傳輸內送和外寄訊息。
- **路由服務**：路由和傳送 JMS 訊息，並控制由訊息服務所使用的訊息，以支援可靠的傳送。
- **永久性服務**：管理資料寫入永久儲存，以及從永久儲存擷取資料。
- **安全性服務**：認證使用者連線到代理程式並授權使用者的動作。
- **監視服務**：產生度量和診斷資訊，並將此資訊寫入指定的輸出通道。

以下各節說明每項服務，並總結針對特殊需求用以自訂服務的特性。

代理程式特性可在不同的配置檔案中定義，也可在用於啓動代理程式的指令行上定義。「Message Queue 管理指南」會說明這些配置檔案以及優先順序，根據此順序，一個檔案的特性值可用於置換其他檔案設定的值。使用啓動指令設定的特性可置換所有其他設定。

連線服務

您可以使用連線相關的特性，配置和管理代理程式及其用戶端之間的實體連線。Message Queue 用戶端可用的連線服務在第 28 頁的「連線至代理程式」中有介紹，其中描述的可用連線服務如下：服務的名稱、類型和基礎協定。連線服務爲多重執行緒，並透過專屬連接埠使用，此專屬連接埠可由代理程式的連接埠對映器動態指定，或由管理員靜態指定。依預設，啓動代理程式時，jms 和 admin 服務會啓動並執行。

由於每個連線均涉及兩方，兩端都會發生連線配置，因此需要協調：

- 用戶端必須配置連線工廠物件的部份屬性，來要求非預設的連線服務、主機和連接埠；指定要連線的代理程式清單，以免需要重新連線到其他代理程式；以及配置重新連線的運作方式。用戶端也可以指定偵測間隔，以測試失敗的連線。
- 然後，管理員就能使用代理程式特性，啓動非預設的連線服務，指派所需的靜態連接埠，配置執行緒，並指定在使用多張網路卡的情況下所要連線的主機。管理員也可以指定偵測間隔，以測試是否可以存取用戶端，這對管理資源很有用。

用戶端可以透過防火牆連線到 Message Queue 服務。若要這麼做，可以請防火牆管理員開啓特定的連接埠，然後連線到該 (靜態) 連接埠；或者使用第 90 頁的「HTTP 連線」中總結的 HTTP 或 HTTPS 服務。

每個連線服務也支援特定認證與授權功能。請參閱第 63 頁的「安全性服務」，以獲得更多資訊。

連接埠對映器

常駐在代理程式主連接埠 7676 的共用*連接埠對映器*會爲連線服務動態指定連接埠。當 Message Queue 用戶端執行階段設定與代理程式的連線時，它會首先連絡連接埠對映器，爲選擇的連線服務請求連接埠號碼。

配置連線服務時，可以透過爲 jms、ssljms、admin 與 ssladmin 連線服務指定*靜態*連接埠號碼，覆寫連接埠對映器。但是，靜態連接埠通常僅在特殊情況下使用，例如透過防火牆連線，一般不建議使用。

執行緒池管理

每種連線服務均爲多重執行緒，支援多重連線。這些連線所需的執行緒由代理程式在執行緒池中維護。其配置方式則取決於您爲最小執行緒與最大執行緒所指定的值，以及您所選擇的執行緒模型。

您可以設定代理程式特性，指定執行緒的最小數值和最大數值。當連線需要執行緒時，就會爲支援該連線的服務新增執行緒到執行緒池數。最小值會指定可配置的執行緒數。當可用執行緒超過此最小臨界值時，系統將會關閉執行緒，這樣這些執行緒在再次達到最小值之前會變爲可用的，從而可以節省記憶體資源。在負載量較大的情況下，執行緒的數目會增加，直至達到執行緒池的最大數目爲止；此時會拒絕新的連線，直至有執行緒變爲可用爲止。

您選擇的執行緒模型會指定執行緒是專屬於單一連線，還是由多重連線共用：

- 在專屬模型中，與代理程式的每條連線均需要兩個執行緒：一個用於內送訊息，另一個用於外寄訊息。這會限制可能的連線數量，但會提高效能。
- 在共用模型中傳送或接收訊息時，共用執行緒會處理連線。因爲每個連線不需要專屬的執行緒，所以此模型會增加可能連線的數量，但是執行緒管理的耗用時間也會增加，進而會影響效能。

目標與路由服務

用戶端連線到代理程式後，便能進行路由與傳送訊息。在此階段中，代理程式會負責建立與管理不同類型的實體目標，確保訊息流量順暢，以及有效使用資源。代理程式使用與路由和目標相關的代理程式特性，以符合您應用程式所需的方式來管理這些工作。

之前已介紹過代理程式上*實體目標*的概念，這是一個訊息傳送到訊息用戶之前儲存訊息的記憶體位置。實體目標有四種：

- **管理員建立的目標**由管理員使用 GUI (`imqadmin`) 或 `imqcmd` 公用程式所建立。這些目標對應到以程式設計方式建立的邏輯目標，或對應到由管理員建立並供用戶端查找的目標管理物件。您可以使用 `imqcmd` 公用程式爲每個管理員建立的目標設定或更新特性。
- **自動建立的目標**會在每次訊息用戶或產生者嘗試存取不存在的目標時，自動由代理程式建立。這些目標一般會在開發期間使用。您可以設定代理程式特性，以防止建立此類目標。您可以設定代理程式特性，以在特定代理程式上配置所有自動建立的目標。

  當不再使用自動建立的目標時，此目標將會由代理程式自動銷毀：亦即，當此目標沒有使用者用戶端且不再包含任何訊息時。如果代理程式重新啓動，僅會在目標包含永久性訊息時才會重新建立這類目標。
- **暫存目標**由需要目標 (用於接收訊息回覆) 的用戶端透過程式設計方式明確建立和銷毀。如其名稱所示，這些目標是暫時的，僅在建立的連線期間由代理程式維護。

  暫存目標不會永久地儲存，也不會在代理程式重新啓動時再次建立，但是它們對於管理工具而言是可見的。

- **停用的訊息佇列**是代理程式啓動時自動建立的專用目標，用於儲存停用訊息以備診斷之用。您可以使用 imqcmd 公用程式設定停用訊息佇列的特性。

管理目標

您可以使用 imqcmd 公用程式管理目標。管理目標包括下列一項或多項工作：

- 建立、暫停、繼續或銷毀目標。
- 列出代理程式上的所有目標。
- 顯示目標狀態與特性的相關資訊
- 顯示目標的度量資訊。
- 壓縮目標用於存留訊息的磁碟空間。
- 更新實體目標的特性。

管理工作會隨正在管理的目標類型而有所變更：管理員建立的目標、自動建立的目標、暫存目標或停用的訊息佇列。例如，暫存目標不需要明確銷毀，自動建立的特性會經由代理程式配置特性配置，並套用到該代理程式上所有自動建立的目標。

配置實體目標

爲了最佳化效能，您可以在建立或更新實體目標時設定特性。可以設定的特性包括下列項目：

- 目標的類型和名稱。
- 目標的個別限制和總計限制 (訊息的最大數目、最大總位元組數、每則訊息的最大位元組數、產生者的最大數目)。
- 超過個別限制或總計限制時，代理程式應執行的操作。
- 單一批次可以傳送的最大訊息數目。
- 目標的停用訊息是否應傳送到停用訊息佇列。
- 如果是叢集代理程式，目標是否應複製到叢集的其他代理程式。

您也可以針對佇列目標配置備份用戶的最大數目，並且可以指定 (針對叢集代理程式) 是否要傳送到本機佇列。

您也可以配置停用訊息佇列的限制和運作方式。不過請注意，此佇列的預設特性和標準佇列的預設特性不同。

管理記憶體

目標可使用大量資源 (取決於它們處理的訊息數目和大小，以及註冊的用戶數目和期限)，因此必須密切管理這些資源，以確保良好的訊息傳送服務效能與可靠性。

您可以設定特性以避免代理程式有過多的內送訊息，進而避免代理程式記憶體不足。代理程式使用三層記憶體保護，以在資源不足時維持訊息傳送服務作業：目標限制、系統範圍限制和系統記憶體臨界值。理想情況下，如果目標限制和系統範圍限制設定適當，則應絕對不會到達嚴重的系統記憶體臨界值。

目標訊息限制

您可以設定目標屬性，以管理每個目標的記憶體和訊息流量。例如，您可以指定允許用於目標的產生者最大數目、目標中允許的最大訊息數量 (或大小)，或每個單一訊息的最大大小。

您也可以指定到達這類限制時，代理程式應採取的反應措施：減緩產生者、捨棄最舊的訊息、捨棄最不重要的訊息，或拒絕最新的訊息。

系統範圍訊息限制

您也可以使用特性設定套用到代理程式上所有目標的限制：可以指定訊息總數和所有訊息使用的記憶體。如果到達任何系統範圍訊息限制，代理程式會拒絕新的訊息。

系統記憶體臨界值

最後，可以使用特性來設定臨界值，代理程式在到達此臨界值時，會不斷採取重要動作以防止記憶體超過負載。此動作取決於記憶體資源的狀態：`green` (大量記憶體可用)、`yellow` (代理程式記憶體正在減少)、`orange` (代理程式記憶體不足) 以及 `red` (代理程式無記憶體)。如果代理程式記憶體的狀態從 `green` 變爲 `red`，代理程式會不斷採取重要動作：

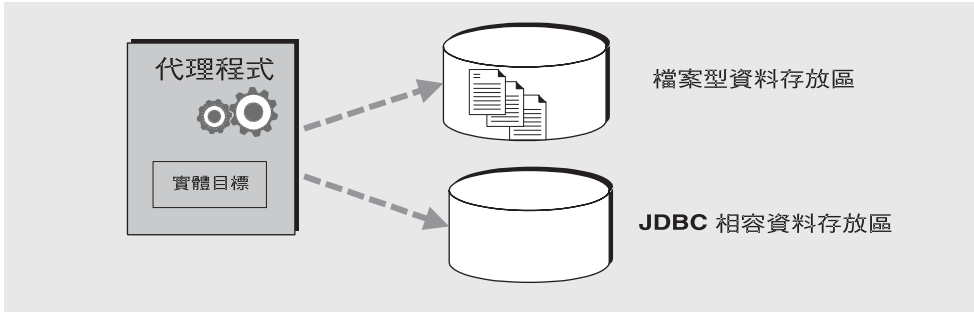
- 它會捨棄資料存放區中永久性訊息的內部記憶體副本。
- 它會減少非永久性訊息的產生者，最終使訊息停止流入代理程式。永久性訊息流量會自動受到代理程式對每個訊息要求的限制。

永久性服務

如果要在代理程式失敗時回復，則此代理程式需要重新建立其訊息傳送作業的狀態。若要執行此操作，就必須將狀態資訊儲存到資料存放區。重新啓動代理程式時，它會使用儲存的資料來重新建立目標和長期訂閱，回復永久性訊息，回復開啓的作業事件，並爲未傳送的訊息重新建立路由表格。然後，它可以恢復訊息傳送。

Message Queue 服務支援檔案型和 JDBC 相容的永久性模組 (請參閱圖 3-2)，且預設使用檔案型永久性模組。

圖 3-2　永久性支援



檔案型永久性

檔案型永久性是使用個別檔案儲存永久性資料的機制。如果您使用檔案型永久性，您可以設定代理程式特性執行下列作業：

- 壓縮資料存放區以減少因新增和移除訊息而產生的分段程序。
- 在*每次*寫入時利用實體儲存裝置同步化內部記憶體狀態。這有助於消除因系統當機而產生的資料遺失。
- 管理配置訊息到資料存放檔案，以及管理檔案管理和儲存所需的資源。

檔案型永久性通常快於 JDBC 型永久性；但是某些使用者更喜歡使用 JDBC 相容存放區所提供的備援和管理控制功能。

JDBC 型永久性

JDBC 型永久性使用 Java 資料庫連接 (JDBC™) 介面，將代理程式連接到 JDBC 相容的資料存放區。若要讓代理程式透過 JDBC 驅動程式存取資料存放區，您必須執行下列作業：

- 設定與 JDBC 相關的代理程式配置特性。使用這些特性指定要使用的 JDBC 驅動程式，認證代理程式爲 JDBC 使用者，建立所需的表格等等。
- 使用 `imqdbmgr` 公用程式以使用適當的模式來建立資料存放區。

「Message Queue 管理指南」中詳細介紹了完成這些工作和相關配置特性的完整程序。

安全性服務

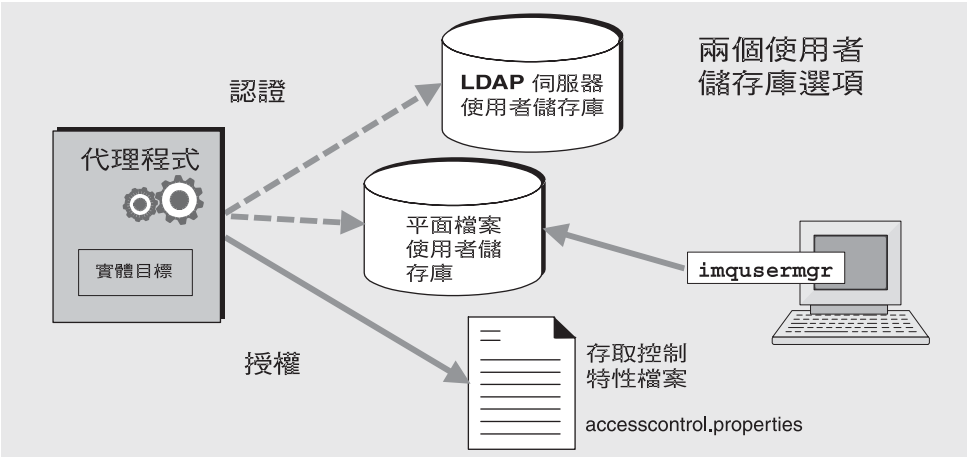
Message Queue 服務支援每個代理程式實例的認證和授權 (存取控制)，也支援加密：

- *認證*可確保只有經過驗證的使用者可以建立與代理程式的連線。
- *授權*會指定哪個使用者或群組有存取資源及執行特定作業的權限。
- *加密*可保護訊息，以防透過連線傳送時遭到篡改。

認證和授權取決於包含有關訊息傳送系統使用者的資訊 (如使用者名稱、密碼和群組成員身份) 的儲存庫。此外，若要授權特定作業給使用者或群組，則代理程式必須檢查指定使用者或群組可執行作業的*存取控制特性檔案*。您負責設定代理程式需要認證使用者和授權使用者動作的資訊。

圖 3-3 顯示代理程式所需的元件，以提供認證和授權。

圖 3-3 安全性管理員支援



如圖 3-3 所示，您可以在 Message Queue 服務隨附的平面檔案使用者儲存庫中儲存使用者資料，或者您可以外掛於預先存在的 LDAP 儲存庫中。您可以設定代理程式特性來表示您的選擇。

- 如果選擇平面檔案儲存庫，必須使用 `imqusermgr` 公用程式來管理儲存庫。使用與內建此選項非常容易。
- 如果要使用現有的 LDAP 伺服器，可以使用 LDAP 供應商提供的工具來寫入和管理此使用者儲存庫。此外，必須在代理程式實例配置檔案中設定特性，讓代理程式能查詢 LDAP 伺服器，以取得使用者和群組的相關資訊。

  如果比例調整性很重要，或者如果需要不同代理程式共用儲存庫，則選擇使用 LDAP 比較好。如果使用代理程式叢集，可能也屬於這種情況。

認證和授權

當用戶端請求連線時，用戶端必須提供使用者名稱和密碼。代理程式會對指定名稱和密碼與儲存在使用者儲存庫中的名稱和密碼進行比較。將此密碼從用戶端傳送至代理程式時，系統會使用基本 64 編碼或訊息摘要 (MD5) 雜湊法對密碼進行編碼。MD5 用於平面檔案儲存庫，LDAP 儲存庫則需要基本 64。如果使用 LDAP，可能會想使用安全 TLS 協定。您可以設定代理程式特性以分別配置每種連線服務所使用的編碼類型，或在代理程式範圍基礎上設定編碼。

當使用者嘗試執行作業時，代理程式會檢查使用者名稱和群組成員身份 (從使用者儲存庫)，是否與爲存取此作業所指定的那些名稱和成員身份 (在存取控制特性檔案中) 相符。存取控制特性檔案可指定以下作業的許可權給使用者或群組：

- 連線至代理程式
- 存取目標：爲任何給定目標或所有目標建立用戶、產生者或佇列瀏覽器
- 自動建立目標

設定代理程式特性以指定下列資訊：

- 是否已啓用存取控制
- 存取控制檔案名稱
- 密碼應該如何編碼
- 系統應等候用戶端回應來自代理程式的認證請求的時間
- 安全連線所需的資訊

加密

若要加密在用戶端與代理程式之間傳送的訊息，您需要使用基於安全套接層 (SSL) 標準的連線服務。透過在已啓用 SSL 的代理程式與已啓用 SSL 的用戶端之間建立已加密連接，SSL 可提供連接級別的安全性。

您可以設定代理程式特性，以指定要使用的 SSL 密鑰存放區之安全性特性，以及密碼檔案的名稱和位置。

監視服務

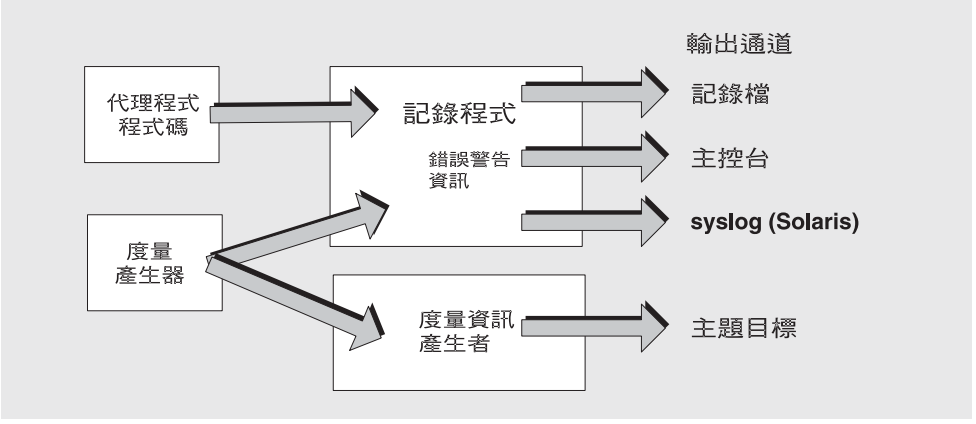
代理程式包含可監視和診斷應用程式和代理程式效能的元件。這些元件包括：

- 產生資料的元件，包括記錄事件的度量產生器與代理程式碼。
- 記錄程式元件，可將輸出資訊寫入到多個輸出通道。

- 訊息產生器，可將包含度量資訊的 JMS 訊息傳送到 JMS 監視用戶端使用的主題目標。

圖 3-4 中說明了一般方案。

圖 3-4 監視服務支援



度量產生器

度量產生器提供代理程式活動的相關資訊，例如流入和流出代理程式的訊息流量、代理程式記憶體中的訊息數目和使用的記憶體、開啓連線的數目，以及使用的執行緒數目。

您可以設定代理程式特性，以開啓或關閉度量資料的產生，並指定產生度量報告的頻率。

記錄程式

Message Queue 記錄程式會記錄代理程式碼和度量產生器產生的資訊，再將該資訊寫入標準輸出 (主控台)、日誌檔，在 Solaris™ 平台上如果發生錯誤時，會寫入 `syslog` 常駐程式程序。

您可以設定代理程式特性以指定記錄程式收集的資訊類型，以及寫入每個輸出通道的類型。在日誌檔中，您還可以指定關閉日誌檔的位置以及將輸出自動重建至新檔案的位置。一旦日誌檔達到指定容量或存在時間，系統將儲存此日誌檔並建立新的日誌檔。

如需配置記錄程式以及如何使用它來取得效能資訊的詳細資訊，請參閱「Message Queue 管理指南」。

度量訊息產生者（企業版）

圖 3-4 中顯示的度量訊息產生者會在固定時間間隔，從度量產生器收到資訊，並將資訊寫入訊息，接著根據訊息中包含的度量資訊類型，將訊息傳送到一些度量主題目標之一。

訂閱至這些度量主題目標的 Message Queue 用戶端，可使用目標中的訊息，並處理訊息中所包含的度量資料。這允許開發者建立自訂監視工具以支援訊息傳送應用程式。如需每個度量訊息類型中所報告之度量數目的詳細資訊，請參閱「Message Queue Developer's Guide for Java Clients」。如需配置度量訊息產生的詳細資訊，請參閱「Message Queue 管理指南」。

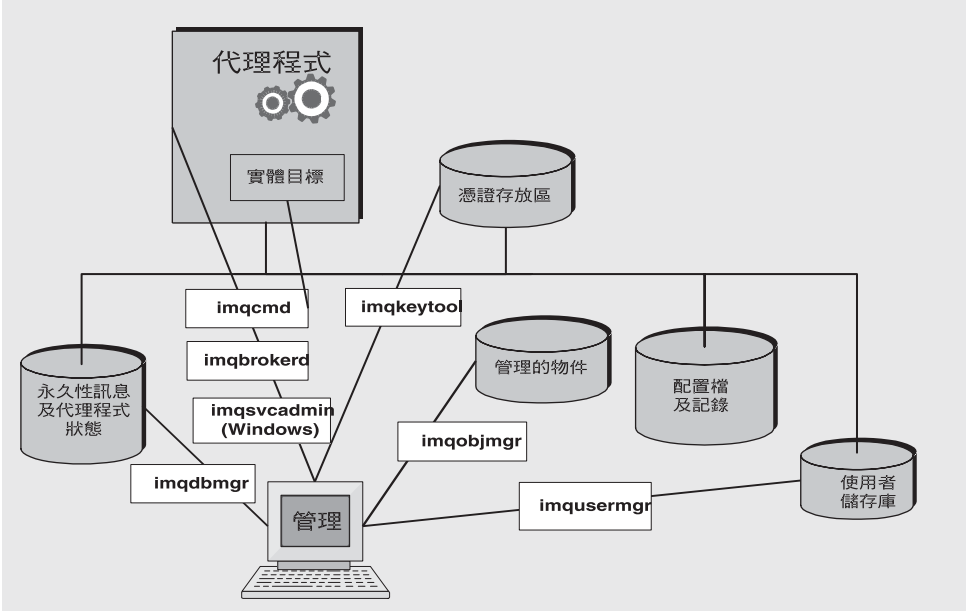
管理工具和工作

本節描述用於配置 Message Queue 服務的工具，以及完成支援開發或生產環境所需的工作。

管理工具

圖 3-5 顯示除用戶端連線以外的訊息服務檢視，並著重在代理程式元件和用以管理這些元件的工具。

圖 3-5 管理工具



您可以使用下列指令行工具，配置及管理 Message Queue 服務。

- 使用 imqbrokerd 公用程式啓動代理程式。您可以使用 imqbrokerd 指令的選項，指定是否應在叢集中連線代理程式，並指定其他啓動配置資訊。
- 啓動代理程式後，請使用 imqcmd 公用程式建立、更新和刪除實體目標，控制代理程式及其連線服務，以及管理代理程式的資源。
- 使用 imqobjmgr 公用程式，新增、列示、更新及刪除 JNDI 物件存放區中管理的物件。
- 使用 imqusermgr 公用程式寫入檔案型使用者儲存庫，以進行使用者認證和授權。
- 使用 imqdbmgr 公用程式建立和管理用於永久性儲存的 JDBC 相容資料庫。(內建檔案存放區不需要外部管理。)
- 使用 imqkeytool 公用程式，產生用於 SSL 認證的自身簽名憑證。
- 使用 imqsvcadmin 公用程式安裝、查詢和移除作爲 Windows 服務的代理程式。

GUI 型管理主控台結合了 imqcmd 和 imqobjmgr 公用程式的部份功能。您可以用於執行下列作業：

- 連接至代理程式並進行管理。

- 建立和管理實體目標。
- 連線到物件存放區，新增物件到存放區，及管理物件。

支援開發環境

開發用戶端元件時，最好保持管理工作量爲最低。Message Queue 產品是專爲協助您執行此作業設計的，拿出包裝盒即可使用。只需啓動代理程式即可。下列作業會讓您專注在開發工作上：

- 使用資料存放區的預設實作 (內建檔案永久性)、使用者儲存庫 (檔案型) 與存取控制特性檔案。這些適用於開發測試。預設使用者儲存庫會使用預設實體建立，可讓您在安裝後立即使用代理程式。您可以使用預設使用者名稱 (guest) 與密碼 (guest) 以認證用戶端。
- 透過建立用於此目標的目錄，可使用簡單的檔案系統物件存放區，並儲存管理物件。如果您根本不想建立存放區，也可以直接在程式碼中創設管理物件。
- 使用自動建立的實體目標，而不是在代理程式上明確建立這些目標。請參閱適當的開發者指南以取得資訊。

支援生產環境

在生產環境中，訊息服務管理在應用程式效能和滿足企業可延伸性、可用性和安全性需求上，發揮關鍵作用。在此環境中，管理員還要執行多項工作。這些工作大致上可分爲設定與維護作業。

設定作業

一般必須執行下列設定作業：

- 安全管理存取

  不論您使用檔案型儲存庫或 LDAP 使用者儲存庫，請確定管理員在 admin 群組中且有安全密碼。視需要爲管理員建立進入代理程式的安全連線。

- 安全用戶端存取

  不論您使用型檔案的儲存庫或 LDAP 使用者儲存庫，請以可存取訊息服務的使用者名稱寫入使用者儲存庫，並編輯存取控制特性檔案以爲使用者提供適當的授權。視需要設定 SSL 型連線服務。若要避免未經認證的連線，請確定要變更「guest」的使用者密碼。

- 建立和配置實體目標

設定目標屬性，以使代理程式資源支援訊息數目和爲訊息配置的記憶體容量。

- 建立和配置管理物件。

  如果您想使用 LDAP 物件存放區，請配置並設定該存放區。建立並配置連線工廠和目標管理物件。

- 若需要狀態水平調整，請建立代理程式叢集。

  建立中央配置檔案並指定主代理程式。

維護作業

若要監視和控制代理程式資源，以及調校應用程式效能，必須在部署應用程式之後，執行下列作業：

- 支援和管理應用程式用戶端
  - 監視和管理目標、長期訂閱和作業事件。
  - 停用自動建立的功能
  - 監視和管理停用的訊息佇列
- 監視與調校代理程式
  - 回復失敗的代理程式
  - 監視、調校和重新配置代理程式。
  - 管理代理程式記憶體資源。
  - 視需要延伸叢集。
- 管理受管理物件

  視需要建立其他管理物件，並調整連線工廠屬性以改善效能與流量。

調整 Message Queue 服務

Message Queue 服務可以經由連線多個代理程式並允許其共用資訊，來進行水平調整。這可讓任一代理程式存取遠端目標，並爲大量用戶端提供服務。請參閱第 4 章「代理程式叢集」，以取得其他資訊。

第 4 章

代理程式叢集

Message Queue 企業版支援*代理程式叢集的使用：*代理程式群組協作工作，提供向用戶端傳送訊息服務。叢集可使管理員透過分散數個代理程式間的用戶端連線，根據訊息流量來調整訊息傳送作業。

本章說明代理程式叢集的架構和內部功能。涵蓋下列主題：

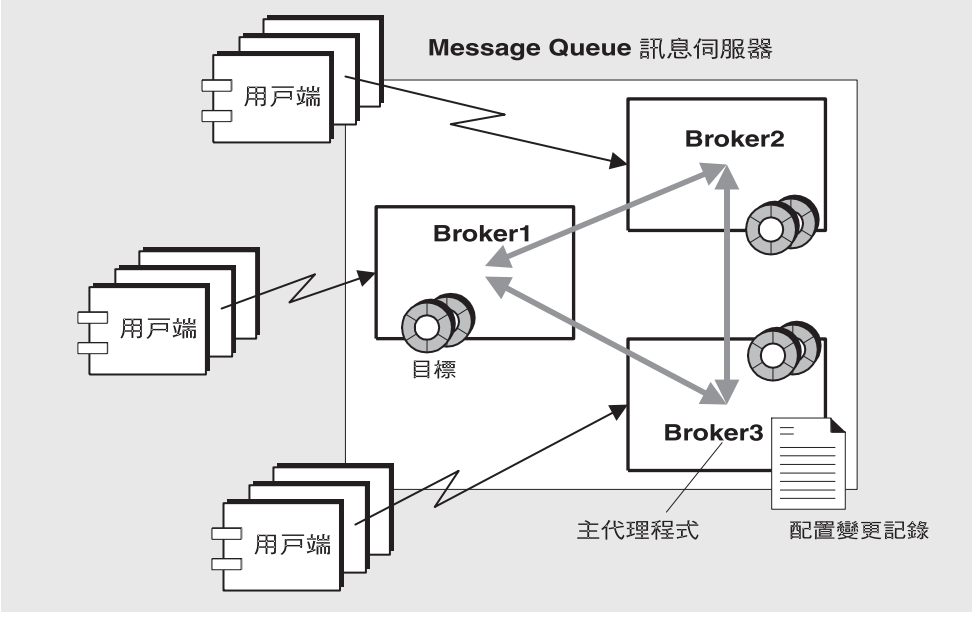
請注意，代理程式叢集提供服務可用性，但不提供資料可用性。如果叢集中的代理程式失敗，連線至該代理程式的用戶端可以重新連線到叢集中的其他代理程式，但是在重新連線到替代代理程式時，部分資料可能會遺失。

叢集架構

圖 4-1 顯示代理程式叢集的 Message Queue 架構。叢集中的每個代理程式會直接連線到其他所有代理程式。每個用戶端 (訊息產生者或用戶) 都擁有單獨的*主代理程式*，藉以直接通訊，進行發送和接收訊息，就好像此主代理程式爲叢集中唯一的代理程式一樣。事實上，主代理程式與其他代理程式一起合作，爲所有連線的用戶端提供傳送服務。

在叢集中，服務可用性取決於代理程式是否能夠共用目標與長期用戶的資訊。如果叢集代理程式失敗，此狀態資訊有可能會不同步。爲防止出現這種可能情況，可以將叢集內的某個代理程式指定爲*主代理程式*。主代理程式會維護*配置變更記錄*，以追蹤對叢集的永久性實體 (目標與長期訂閱) 所作的變更。此記錄可用以傳遞此類變更資訊至變更發生時離線的代理程式。

圖 4-1 叢集架構



下列各節說明即使一或多個代理程式處於離線的狀態，如何在叢集中執行訊息傳送以及如何配置和同步代理程式。

訊息傳送

在叢集配置中，代理程式會共用目標與訊息用戶的資訊：每個代理程式均瞭解下列資訊：

- 叢集中所有實體目標的名稱、類型和屬性
- 每個訊息用戶的名稱、位置和偏好
- 上述的更新 (刪除、新增或重新配置)

這可讓每個代理程式從其本身直接連線的訊息產生者，路由訊息到遠端訊息用戶。產生者的主代理程式與用戶的主代理程式有不同的職責：

- 產生者的主代理程式負責保留及路由來自產生者的訊息，記錄、管理作業事件，以及處理來自使用用戶端的確認。
- 用戶的主代理程式負責保留用戶相關的資訊、轉寄訊息至用戶，以及讓產生者的代理程式瞭解用戶是否仍然可用，以及訊息是否成功使用。

叢集代理程式會一起運作，以將叢集內的訊息流量降到最低；例如，如果遠端代理程式針對相同的主題目標有兩個相同的訂閱，則訊息只會經由線路傳送一次。您可以設定目標特性，指定傳送至本機用戶優先於傳送至遠端用戶，以進一步減少流量。

如果用戶端與代理程式之間需要安全且加密的訊息傳送，可以配置叢集以提供代理程式之間安全的訊息傳送。

目標屬性

針對叢集代理程式上實體目標所設定的屬性，可以套用到叢集中該目標的所有實例；但是這些屬性指定的部分限制會整個套用到叢集，而其他限制則會套用到個別目標實例。表 4-1 列出您可以爲實體目標設定的屬性，並指定屬性的範圍。

表 4-1 叢集代理程式上實體目標的特性

特性名稱	範圍
maxNumMsgs	每個代理程式。藉由在叢集中分散產生者，可讓您提高未使用訊息的限制總量。
maxTotalMsgBytes	每個代理程式。藉由在叢集中分散產生者，可讓您提高未使用訊息的記憶體限制總量。
lmitBehavior	全域。
maxBytesPerMsg	每個代理程式。
maxNumProducers	每個代理程式。

表 4-1 叢集代理程式上實體目標的特性 (續)

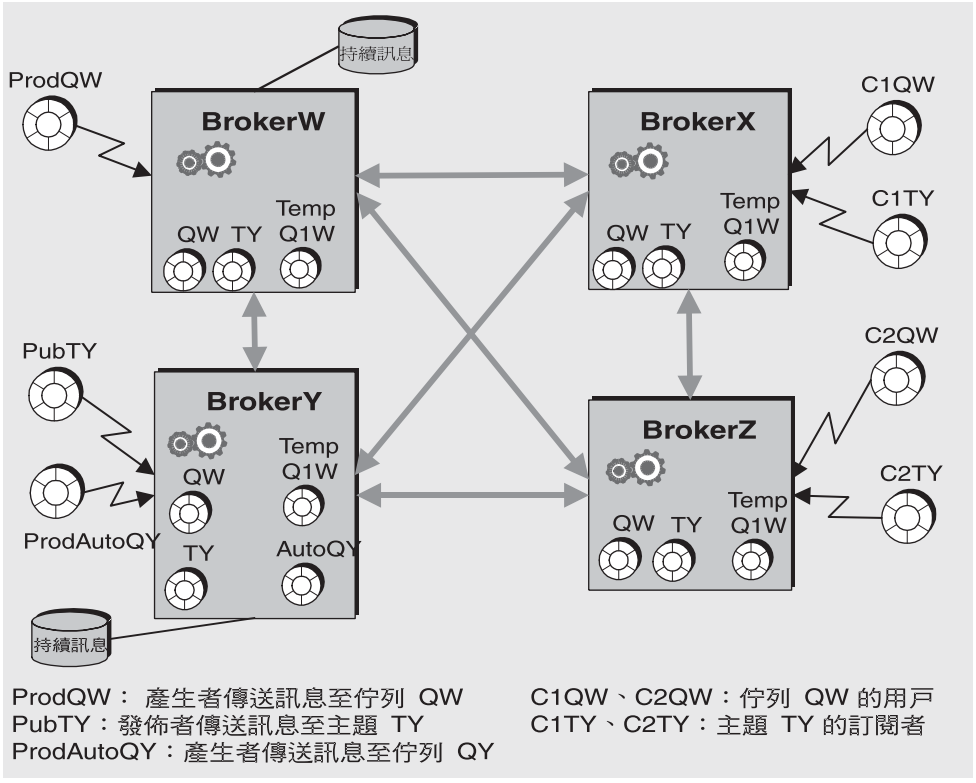
特性名稱	範圍
maxNumActiveConsumers	全域。
maxNumBackupConsumers	全域。
consumerFlowLimit	全域。
localDeliveryPreferred	全域。
isLocalOnly	全域。
useDMQ	每個代理程式。

叢集和目標

不管目標是由管理員建立的、自動建立的還是暫時的，都會影響目標在叢集中傳遞的方式，以及在連線時或代理程式失敗時處理目標的方式。

圖 4-2 顯示四種叢集代理程式。顯示了代理程式之間的直接 (私人) 連線，以及用戶端與其所連代理程式之間的連線。圖中描述多種可能性，並在隨後小節中進行說明。

圖 4-2　叢集範例



使用回覆發送模型產生至佇列

如前圖所示：

1. 管理員會建立實體目標 QW。在建立叢集期間，佇列會複製到整個叢集。
2. 產生者 ProdQW 傳送訊息到佇列 QW，並使用回覆發送模型，引導回覆到暫存佇列 TempQ1W。(暫存佇列會在應用程式建立暫存目標並新增用戶時建立並複製。)
3. 主代理程式 BrokerW 保留傳送到 QW 的訊息，並將該訊息路由到第一個符合此訊息選取條件的使用中用戶。訊息會傳送到用戶 C1QW (在 BrokerX 上) 或用戶 C2QW (在 BrokerY 上)，需視哪個用戶已備妥接收訊息而定。接收訊息的用戶會傳送回覆到目標 TempQ1W。

產生者自動建立的目標

如前圖所示：

1. 產生者 ProdAutoQY 會傳送訊息到目標 AutoQY，但是代理程式上不存在該目標。
2. 代理程式會保留訊息並建立目標 AutoQY。

   自動建立的目標不會自動複製到整個叢集。只有當用戶選擇接收佇列 AutoQY 的訊息時，用戶的主代理程式才會建立目標 AutoQY，並傳送訊息到用戶。自動建立的目標會在用戶建立它的位置複製到整個叢集。

發佈者主題目標

如前圖所示：

1. 管理員會建立實體主題目標 TY。管理員建立的目標 TY，會複製到整個代理程式叢集 (在使用目標之前)。
2. 發佈者 PubTY 傳送訊息到 TY。
3. 主代理程式 BrokerY 保留發佈到 TY 的任何訊息，並將訊息路由到符合此訊息選取條件的所有主題訂閱者。

在連線或代理程式失敗時處理目標

表 4-2 說明不同類型的目標如何在叢集中進行複製與刪除。

表 4-2 處理叢集中的目標

目標	傳遞和刪除
管理員建立的	建立目標時，目標會在叢集中傳遞，並且每個代理程式會一直儲存與目標相關的資訊。 當管理員明確刪除目標時，目標會被銷毀。 如果有主代理程式，會在主代理程式中儲存建立與刪除記錄，以便叢集中的代理程式能同步化狀態資訊。
暫存	建立目標時，目標會在叢集中傳遞。 如果允許與暫存目標關聯的用戶重新連線，目標會一直儲存在用戶的主代理程式中。否則一律不會儲存目標。 如果用戶失去了連線，則會刪除所有代理程式上的目標。 如果用戶的主代理程式當機，且允許用戶重新連線，則與此用戶關聯的暫存目標會受到暫停。如果使用者用戶端未在指定期間內重新連線，便會認為用戶端失敗並刪除目標。

表 4-2　處理叢集中的目標 (續)

目標	傳遞和刪除
自動建立的	當建立產生者但目標不存在時，會在產生者的主代理程式上建立目標。 爲不存在的目標建立用戶時，與該用戶和目標相關的資訊會傳遞到整個叢集。 自動建立的目標可以經由管理員明確刪除，或者在下列情況下自動刪除： • 當給定期間內沒有用戶或訊息時，由每個代理程式自動刪除。 • 當代理程式重新啓動且沒有該目標的訊息時，由每個代理程式自動刪除。

叢集配置

若要在啓動時在叢集中的代理程式之間建立連線，每個代理程式都必須傳送所有其他代理程式的主機名稱與連接埠號碼 (包括主代理程式，如果有)。此資訊由一組*叢集配置特性* 指定，而叢集中所有代理程式的這些設定都應該相同。雖然可以爲每個代理程式個別指定配置特性，但這種方法易產生錯誤並容易導致叢集配置的不一致性。相反，建議您將叢集配置特性置於一個在啓動時每個代理程式均參照的中央*叢集配置檔案*。這樣可確保所有代理程式共用相同的配置資訊。

請參閱「Message Queue 管理指南」以獲得叢集配置特性的詳細資訊。

備註　雖然叢集配置檔案原本是用於配置叢集，但亦方便儲存叢集中所有代理程式共用的其他特性。

叢集更新

無論叢集的配置何時變更，關於變更的資訊即會自動傳遞至叢集中的所有代理程式。發生下列情況之一時，叢集配置會有所變更：

- 建立或銷毀叢集中某個代理程式上的目標。
- 目標的特性已變更。
- 訊息用戶以其主代理程式註冊。
- 訊息用戶與其本機代理程式中斷連線 (不論是明確中斷連線，還是因爲用戶端、代理程式或網路故障而中斷連線)。
- 訊息用戶會建立主題的長期訂閱。

這類變更資訊將立即傳遞至發生變更時叢集內所有線上的代理程式。但是，離線的代理程式 (例如，已損毀的代理程式) 在變更發生時將不會收到變更通知。考量到離線的代理程式，Message Queue 爲叢集維護一個*配置變更記錄*，記錄所有已建立或銷毀的永久性實體 (目標與長期訂閱)。當離線的代理程式重新連線時 (或當新的代理程式加入叢集中時)，代理程式會參考此記錄以取得關於目標與長期訂閱者的資訊，然後與其他代理程式交換目前使用中的訊息用戶之相關資訊。

叢集中的一個代理程式會指定爲*主代理程式*，負責維護配置變更記錄。因爲其他代理程式無法在沒有主代理程式的情況下完成初始化，所以主代理程式務必是叢集中首先啓動的代理程式。如果主代理程式離線，則配置資訊將無法在叢集中傳遞，因爲其他代理程式無法存取配置變更記錄。在這些情況下，如果您嘗試建立、重新配置或銷毀目標或長期訂閱，或嘗試執行如重新啓動長期訂閱等相關作業，則會出現異常。(但非管理訊息的傳送則會繼續正常工作。) 使用主代理程式和配置變更記錄是選擇性的。如果您擔心叢集配置變更或代理程式失敗之後叢集無法同步，才會需要主代理程式和配置變更記錄。

第 5 章

Message Queue 和 J2EE

Java 2 Platform, Enterprise Edition (J2EE 平台) 是標準伺服器平台裝載多層式和精簡型用戶端企業應用程式的規格。J2EE 平台的其中一個需求是分散式元件能透過可靠的非同步訊息傳送而彼此互動。這樣的互動能透過 JMS 提供者而達成。事實上，Message Queue 即是 J2EE 平台的參照 JMS 實作。

本章說明在 J2EE 平台環境中實作 JMS 支援的結果。本章涵蓋下列主題：

- 第 80 頁的「JMS/J2EE 程式設計：訊息驅動 Bean」
- 第 81 頁的「J2EE 應用程式伺服器支援」

如需有關將 Message Queue 用作 J2EE 相容應用程式伺服器的 JMS 提供者之其他資訊，請參閱「Message Queue 管理指南」。

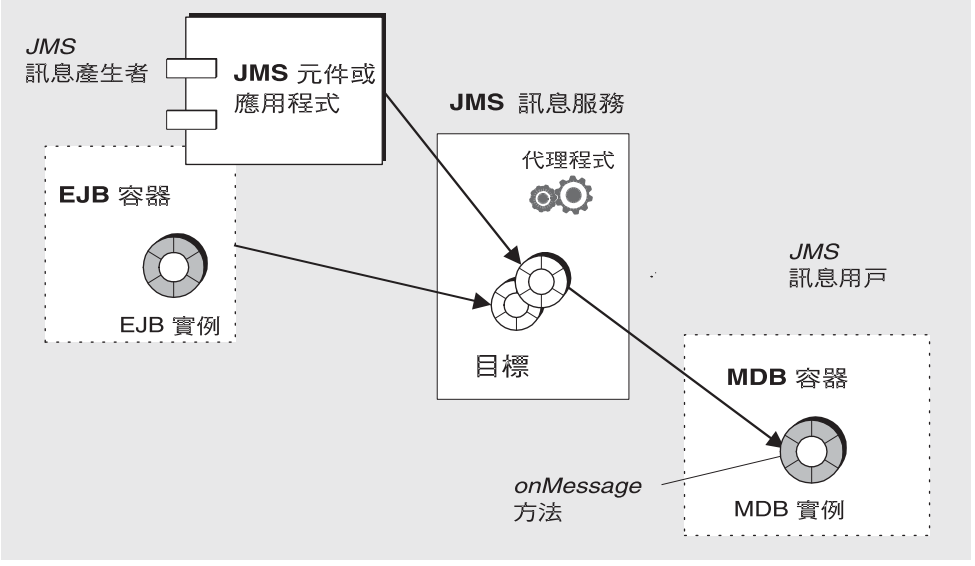
JMS/J2EE 程式設計：訊息驅動 Bean

除第 2 章中介紹的一般 JMS 用戶端程式設計模型之外，還有專用於 J2EE 平台應用程式環境的 JMS 用戶端。此專用的用戶端稱爲*訊息驅動 Bean*，是 EJB 2.0 (以及之後版本) 規格 (http://java.sun.com/products/ejb/docs.html) 中所描述的 Enterprise JavaBeans (EJB) 元件系列之一。

因爲其他 EJB 元件 (階段作業 Bean 和實體 Bean) 僅可透過標準 EJB 介面同步呼叫，所以產生了對訊息驅動 Bean 的需求。但是，許多企業應用程式需要非同步訊息傳送。這些應用程式大都需要伺服器端元件在不佔用伺服器資源的情況下，能夠互相通訊。因此，就出現了對 EJB 元件的需求，可以不需要緊密耦合到訊息產生者即可接收和使用訊息。對於伺服器端元件必須回應應用程式事件的任何應用程式，均必須具備此功能。在企業應用程式中，此功能還必須能夠在負載不斷增加的情況下進行延伸。

訊息驅動 Bean (MDB) 是專用 EJB 容器 (爲所支援的元件提供分散式服務) 支援的 EJB 元件。

圖 5-1 與 MDB 進行訊息傳送



- JMS 訊息驅動 Bean 是實作 JMS `MessageListener` 介面的 EJB。在 MDB 容器接收訊息時，會呼叫 `onMessage` 方法 (由 MDB 開發者編寫)。`onMessage()` 方法使用該訊息的方式與標準 `MessageListener` 物件之 `onMessage()` 方法使用訊息的方式一樣。您不能以呼叫其他 EJB 元件上方法的方式遠端呼叫 MDB 上的方法：因此不存在與其關聯的本機介面或遠端介面。MDB 可以使用來自單一目標的訊息。獨立式 JMS 應用程式、JMS 元件、EJB 元件或 Web 元件均可產生訊息，如圖 5-1 所示。
- 專用 EJB 容器支援 MDB。它會建立並設定 MDB 實例，以用於非同步訊息。該容器會設定與訊息服務 (包括認證) 的連線，建立與給定目標相關聯的階段作業池，並管理階段作業池之間的分配訊息作業。由於容器控制 MDB 實例的生命週期，因此容器透過管理 MDB 實例池來容納內送訊息負載。

  與 MDB 關聯的是一個部署描述元，此描述元會於設定訊息使用時，爲容器使用的連線工廠和目標設定屬性。部署描述元還可以包括部署工具用於配置容器所需的其他資訊。每個這類容器均支援單一 MDB 的實例。

J2EE 應用程式伺服器支援

在 J2EE 架構中，EJB 容器由 J2EE 應用程式伺服器主控。應用程式伺服器提供各種容器所需的資源：作業事件管理員、持續性管理員、名稱服務以及用於訊息傳送和 MDB 的 JMS 提供者。

在 Sun Java System Application Server 中，JMS 訊息傳送資源由 Sun Java System Message Queue 提供：

- 對於 Sun Java System Application Server 7.0，Message Queue 訊息傳送系統已整合至應用程式伺服器整合，作爲其原生 JMS 提供者。
- 對於 Sun J2EE 1.4 Application Server，Message Queue 外掛到應用程式伺服器，作爲內嵌的 JMS 資源介面。

如需新版的 Application Server，Message Queue 會使用標準的資源介面部署和配置方法，外掛到應用程式伺服器。

如需有關 J2EE 架構的資訊，請參閱 `http://java.sun.com/j2ee/download.html#platformspec` 上的 J2EE 平台規格。

JMS 資源介面

*資源介面*是一種可將附加功能插入到符合 J2EE 1.4 規格的應用程式伺服器中的標準方式。該標準是由 J2EE Connector Architecture (J2EECA) 1.5 規格所定義，允許應用程式伺服器以標準方式與外部系統進行互動。外部系統可包括企業資訊系統 (EIS)，以及訊息傳送系統：例如，JMS 提供者。Message Queue 包括一個 JMS 資源介面，允許應用程式伺服器將 Message Queue 用作 JMS 提供者。

將 JMS 資源介面插入至應用程式伺服器，可讓在應用程式伺服器中已部署並執行的 J2EE 元件交換 JMS 訊息。使用 J2EE 應用程式伺服器管理工具，可以建立並配置這些元件所需的 JMS 連線工廠與目標管理物件。

但是，諸如管理代理程式與實體目標的其他管理作業，不包括在 J2EECA 規格中，且這些作業只能透過提供者的特定工具來執行。

Message Queue 資源介面整合在 Sun J2EE 1.4 應用程式伺服器中。但是，還未被任何其他 J2EE 1.4 應用程式伺服器認證過。

Message Queue 資源介面爲單一檔案 (`imqjmsra.rar`)，位於作業系統的某個目錄中，此目錄位置因作業系統而異 (請參閱「Message Queue 管理指南」)。`imqjmsra.rar` 檔案包含資源介面部署描述元 (`ra.xml`) 以及應用程式伺服器使用該介面所需的 JAR 檔案。

遵循應用程式伺服器隨附的資源介面部署與配置說明，您即可在任何 J2EE-1.4 相容應用程式伺服器中使用 Message Queue 資源介面。當 J2EE 1.4 應用程式伺服器面市銷售，並且已爲這些應用程式伺服器認證 Message Queue 資源介面時，Message Queue 文件將會提供有關部署與配置程序的具體資訊。

附錄 A

選擇性 JMS 功能的 Message Queue 實作

JMS 規格會指出特定的選擇性項目：每個 JMS 提供者 (供應商) 選擇性擇是否實作這些項目。本附錄描述 Message Queue 產品如何處理 JMS 選擇性項目。

表 A-1 說明 Message Queue 服務如何處理 JMS 選擇性項目。

表 A-1 選擇性 JMS 功能

JMS 規格中的章節	對應的 Message Queue 實作
3.4.3 JMSMessageID	「由於訊息 ID 建立和增加訊息容量載為複雜，如果某些 JMS 提供者收到應用程式未使用訊息 ID 的提示，則可能會最佳化訊息耗用時間。JMS 訊息產生者會提供停用訊息 ID 的提示。」 **Message Queue 實作：** 產品不停用訊息 ID 的產生 (MessageProducer 中的所有 setDisableMessageID() 呼叫均會被忽略)。所有訊息均將包含有效的 MessageID 值。
3.4.12 覆寫訊息標頭欄位	「JMS 未明確定義管理員如何覆寫這些標頭欄位值。JMS 提供者不需要支援此管理選項。」 **Message Queue 實作：** Message Queue 產品透過在用戶端執行階段，支援訊息標頭欄位值的管理覆寫 (請參閱第 44 頁的「訊息標頭」)。
3.5.9 JMS 定義的特性	「JMS 可為 JMS 定義的特性保留 「JMSX」特性名稱字首。」「對這些特性的支援是選擇性的，除非另有說明。」 **Message Queue 實作：** Message Queue 產品支援 JMS 1.1 規格定義的 JMSX 特性 (請參閱 「Message Queue 管理指南」)。
3.5.10 提供者特定的特性	「JMS 可為提供者特定的特性保留 「JMS_<*vendor_name*>」特性名稱字首。」 **Message Queue 實作：** 提供者特定的特性是用於提供需要的特殊功能，以支援 JMS 與提供者原生用戶端一起使用。但不能用於 JMS 執行 JMS 訊息傳送。

表 A-1 選擇性 JMS 功能 (續)

JMS 規格的章節	關於 Message Queue 實作
4.4.8 分散式作業事件	「JMS 不需要提供者支援分散式作業事件。」 **Message Queue 實作：** 此版本的 Message Queue 產品支援分散式作業事件 (請參閱第 51 頁的「作業事件」)。
4.4.9 多重階段作業	「對於 PTP < 點對點的傳遞模型 >，JMS 沒有為相同的佇列指定並行運作的 QueueReceivers 之語義；但是，JMS 並不禁止提供者支援此功能。」請參閱 JMS 規格的第 5.8 節，以取得更多資訊。 **Message Queue 實作：** Message Queue 實作支援到多個用戶的佇列傳送。如需更多資訊，請參閱第 36 頁的「點對點訊息傳送」。

Message Queue 功能

Message Queue 服務完整實作 JMS 1.1 規格，提供可靠的非同步彈性訊息傳送 如需有關 JMS 相容性問題的資訊，請參閱附錄 A「選擇性 JMS 功能的 Message Queue 實作」。但是，Message Queue 的功能超出了 JMS 需求。您可以使用這些功能整合並監視由許多分散式元件組成的系統，在全天候的關鍵任務作業中交換數以千計的訊息。

本書已在描述 Message Queue 服務的過程中，介紹過這些功能。爲了方便起見，此附錄提供了 Message Queue 功能之摘要：簡短描述每項功能，摘要使用功能所需的作業，並爲本書中介紹這些功能之章節提供參照，以及爲 Message Queue 文件集中詳細描述這些功能之具體文件提供參照。

表 B-1 中按照字母順序列出 Message Queue 的功能，大致上可分爲以下幾種類別。

- 整合支援
  - HTTP 連線
  - 安全連線
  - C 用戶端支援
  - SOAP 支援
  - J2EE 資源介面

- 安全性
  - 認證
  - 授權
  - 加密 (請參閱安全連線)
- 可延伸性
  - 執行緒管理
  - 執行緒管理
  - 執行緒管理
- 可用性
  - 記憶體資源管理
  - 對用戶端的訊息流量控制
  - 自動重新連線
  - 可靠的資料永久性
  - 連線偵測
- 可管理性
  - 管理工具
  - 訊息型監視 API
  - 可調校的效能
  - 可配置的實體目標
  - 代理程式配置
  - 停用的訊息佇列
- 彈性伺服器配置
  - 可配置的永久性
  - LDAP 伺服器支援
  - JNDI 服務提供者支援

- 效能
  - 訊息壓縮
  - 可調校的效能
  - 可配置的實體目標

表 B-1 訊息佇列功能

功能	說明與參照
管理工具	Message Queue 服務包括 GUI 和指令行工具，可管理目標、作業事件、長期訂閱、管理物件存放區、使用者儲存庫、JDBC 相容的資料存放區和伺服器憑證。 **參照** 第 67 頁的「管理工具」。 「Message Queue 管理指南」的「管理工作和工具」。
認證	認證使用者對找與代理程式的連線。 Message Queue 服務藉由對照儲存在使用者儲存庫中的值驗證使用者名稱及密碼，讓使用者得以連線到代理程式。儲存庫可以是 Message Queue 隨附的平面檔案儲存庫或是 LDAP 儲存庫 (LDAP v2 或 v3 協定)。 **使用方式** 1. 建立使用者儲存庫或使用預設的實例。 2. 使用 `imqusermgr` 工具管入儲存庫 **參照** 第 65 頁的「認證和授權」。 「Message Queue 管理指南」的「管理安全性」。
授權	授權使用者執行特定作業。 Message Queue 服務可讓您建立存取控制特性檔案，指定使用者和使用者群組可以執行的作業。代理程式會在用戶端嘗試建立連線、建立產生者、建立使用者或瀏覽佇列時，檢查此檔案。 **使用方式** 編輯碼代理程式實例自動建立的存取控制特性檔案。 **參照** 第 65 頁的「認證和授權」。 「Message Queue 管理指南」的「管理安全性」。

表 B-1 訊息佇列功能 (續)

功能	描述與參照
自動重新連線	管理員設定連線工廠管理物件的連線屬性，以在連線失敗時自動重新連線。可以重新連線到相同的代理程式，或者連線到叢集中的其他代理程式 (如果在使用叢集)。您可以指定嘗試重新連線的次數，以及每次嘗試的間隔時間。若是叢集代理程式，亦可以指定在代理程式清單中重複連線的次數，以及是否要在清單中以特定的順序重複連線。 **參照** 第 59 頁的「連線服務」。 「Message Queue 管理指南」中的「管理受管理物件」與「管理物件屬性參照」。
代理程式叢集	管理員可以藉由將這些代理程式實例群組爲代理程式叢集，平衡眾多實例間用戶端連線及訊息傳送。 **使用步驟** 1. 指定叢集中每個代理程式的叢集配置特性。若要這麼做，可以使用配置檔案，或設定每個代理程式的特性。 2. 若是主代理程式，請啓動主代理程式 3. 啓動叢集中的其他代理程式。 **參照** 第 4 章「代理程式叢集」。 「Message Queue 管理指南」中的「使用叢集」。
代理程式配置	管理員可以設定代理程式特性，來調校 Message Queue 服務效能。這包括路由服務、永久性服務、安全性、監視和管理物件的管理作業。 **參照** 第 3 章「Message Queue 服務」。 「Message Queue 管理指南」中的「配置代理程式」與「代理程式特性參照」。
C 用戶端支援	C 用戶端可以使用 Message Queue 訊息傳送服務傳送與接收訊息。C API 可讓傳統 C 應用程式與 C++ 應用程式參與 JMS 型訊息傳送。 支援大部分標準 JMS 功能的 C 用戶端執行階段都支援 Message Queue C API，除了下列情況之外：使用管理物件；對映、串流或物件訊息以外類型；分散式作業事件；以及佇列瀏覽器。C 用戶端執行階段也不支援 Message Queue 的大部分企業功能。 **參照** 第 55 頁的「Java 用戶端與 C 用戶端」。 「Message Queue Developer's Guide for C Clients」。

表 B-1　訊息佇列功能 (續)

功能	說明與參照
壓縮訊息	Java 用戶端可以設定訊息特性，讓用戶端執行階段壓縮正在傳送的訊息。用戶端的執行階段會解壓縮訊息，再將訊息傳送給用戶。您還可使用提供的其他特性，判定壓縮訊息是否真能改善效能。 **參照** 第 46 頁的「訊息服務」。 「Message Queue Developer's Guide for Java Clients」中的「Message Queue Clients: Design and Features」。
可配置的永久性	管理員可以配置代理程式使用 Message Queue隨附的檔案型永久存放區或 JDBC 相容資料庫，例如Oracle 8i。 **使用方式** 設定與檔案系統永久存放區或 JDBC 相容儲存相關的代理程式特性。 **參照** 第 62 頁的「永久性服務」。 「Message Queue 管理指南」中的「配置代理程式」。
可配置的實體目標	管理員可以在建立目標時設定實體目標特性，以定義部分訊息傳送的運作方式。您可以針對任何目標配置下列運作方式：未使用訊息的最大數目或這類訊息允許的最大記憶體數量，當到達記憶體限制時，代理程式應該拒絕這些訊息；產生者和用戶的最大數目；訊息的最大大小；單一批次中所傳送的訊息之最大數目；目標是否只傳送給本機用戶；以及是否可將目標的停用訊息移至停用訊息佇列。 **參照** 第 60 頁的「目標與路由服務」。 「Message Queue 管理指南」中的「管理實體目標」與「實體目標特性參照」。
連線偵測	管理員可以設定連線工廠屬性，以指定從用戶端執行階段到代理程式的偵測作業頻率。這可讓用戶端事先偵測到失敗的連線。 **參照** 第 59 頁的「連線服務」。 「Message Queue 管理指南」中的「連線工廠屬性」。
停用的訊息佇列	Message Queue 訊息服務會建立停用訊息佇列，以保留逾期的訊息或代理程式無法處理的訊息。您可以檢查佇列的內容，以監視、審核系統效能，或對系統效能進行疑難排解。 **參照** 第 60 頁的「目標與路由服務」。 「Message Queue 管理指南」中的「管理實體目標」。

表 B-1 訊息佇列功能 (續)

功能	說明與參照
HTTP 連線	Java 用戶端可以建立至代理程式的 HTTP 連線。 HTTP 傳輸允許透過防火牆傳送訊息。Message Queue 使用在 Web 伺服器環境中執行的 HTTP 通道 Servlet 來實作HTTP 支援。用戶端產生的訊息會作爲 HTTP 請求負裝在用戶端執行階段中，並透過防火牆使用 HTTP 傳送到通道 Servlet。通道 Servlet 會從HTTP 請求擷取 JMS 訊息，並透過 TCP/IP 將訊息傳送到代理程式。 **使用方式** 1. 在 Web 伺服器上部署 HTTP 通道 Servlet。 2. 配置代理程式的 `httpjms` 連線服務，並啓動代理程式。 3. 配置 HTTP 連線 4. 取得至代理程式的 HTTP 連線。(僅限 Java 用戶端。) **參照** 第 28 頁的「連線至代理程式」。 「Message Queue 管理指南 」的附錄 C「啓用 HTTP 支援」。
互動式監視	管理員可以使用 `imqcmd metrics` 指令，從遠端監視代理程式。監視的資料包括 JVM 度量、代理程式訊息流量、連線、連線資源、訊息、目標訊息流量、目標用戶、目標資源使用。 **參照** 第 65 頁的「監視服務」。 「Message Queue 管理指南 」中的「監視訊息伺服器」。
J2EE 資源介面	Message Queue 提供可以外掛到 J2EE 相容應用程式伺服器的資源介面。應用程式伺服器將 Message Queue 用作JMS 提供者，以滿足 J2EE 的需求，讓在應用程式伺服器中執行的分散式元件能使用可靠、非同步的訊息彼此互動。 **使用方式** 設定介面屬性以配置介面。 **參照** 第 81 頁的「J2EE 應用程式伺服器支援」。 「Message Queue 管理指南 」中的「JMS 資源介面屬性參照」。
JNDI 服務提供者支援	用戶端可以使用 JNDI API 查找管理物件。 管理員可以使用 `imqobjmgr` 公用程式，在使用 JNDI 存取的物件存放區中新增、列示、更新和刪除管理物件。 **參照** 第 67 頁的「管理工具」。 「Message Queue 管理指南 」中的「指令參照」。

表 B-1　訊息佇列功能 (續)

功能	說明與參照
LDAP 伺服器支援	管理員可以使用 LDAP 伺服器來於管理物件存放區，以及儲存認證與授權所需的使用者資訊。依預設，Message Queue 會為此資料提供檔案型儲存。 **管理物件的使用方式** 1. 使用供應商提供的工具，來匯入與管理使用者儲存庫。 2. 設定 LDAP 相關代理程式特性。 3. 設定管理使用者的存取控制。 **參照** 「Message Queue 管理指南」中的「管理安全性」。 **使用者儲存庫的使用方式** 1. 使用供應商提供的工具，來設定 LDAP 伺服器。 2. 設定 LDAP 相關代理程式特性，以定義存放區的初始環境和位置。 3. 設定與保護 LDAP 伺服器作業相關的 LDAP 相關代理程式特性。 **參照** 第 63 頁的「安全性服務」。 「Message Queue 管理指南」中的「管理受管理物件」。
記憶體資源管理	管理員可以配置下列運作方式： 1. 設定目標特性以指定產生者的最大數目、所有訊息量的最大值，以及任一訊息的最大大小。 2. 設定目標特性以控制訊息流量 3. 設定目標特性以管理每個目標的訊息流量 4. 設定代理程式特性為該代理程式指定所有目標上的訊息限制。 5. 設定代理程式特性以指定可用的系統記憶體臨界值，當到達臨界值時，代理程式會採取越來越重要的動作，以防止記憶體超載。此動作取決於記憶體資源的狀態。 **參照** 第 60 頁的「目標與路由服務」。 「Message Queue 管理指南」中的「配置代理程式」、「代理程式特性參照」與「實體目標特性參照」。
訊息壓縮	開發者可以設定訊息標頭特性，讓用戶端執行階段壓縮訊息再進行傳送。用戶端執行階段會解壓縮訊息，再將訊息傳送給用戶。 **參照** 第 46 頁的「訊息特性」。 「Message Queue Developer's Guide for Java Clients」中的「Message Compression」。

表 B-1 訊息佇列功能 (續)

功能	說明與參照
對用戶端的訊息流量控制	管理員或開發者可以配置連線，指定各項流量限制與計算方法，以將有效負載與控制訊息衝突降至最低，且可將訊息流量放至最大。 **使用方式** 設定連線工廠管理物件 (管理員) 的流量控制屬性，或者設定連線工廠 (開發者) 的流量控制特性。 **參照** 第 43 頁的「連線工廠與連線」。 「Message Queue 管理指南」中的「管理受管理物件」與「管理物件屬性參照」。
訊息型監視 API	Java 用戶端可以使用監視 API 建立自訂的監視應用程式。監視應用程式是從特殊佇量主題目標擷取佇量訊息的用戶。 **使用方式** 1. 登入佇量監視用戶端。 2. 設定代理程式特性以配置代理程式的佇量訊息產生者。 3. 設定佇量主題目標的存取控制。 4. 啓動監視用戶端。 **參照** 第 65 頁的「監視服務」。 「Message Queue Developer's Guide for Java Clients」中的「Using the Metrics Monitoring API」。 「Message Queue 管理指南」中的「監視訊息使用者」。
佇列傳送至多個用戶	用戶端可以爲給定的佇列註冊多個用戶。 管理員可以爲佇列指定使用中用戶的最大數目，以及備份用戶的最大數目。代理程式可以分散訊息到註冊用戶，平衡用戶間負載以允許系統比例調整。 **使用方式** 設定實體目標特性 `maxNumActiveConsumers` 和 `maxNumBackupConsumers`。 **參照** 第 36 頁的「點對點訊息傳送」。 「Message Queue 管理指南」中的「實體目標特性」與「多重用戶佇列效能」。
可靠的資料永久性	若要取得絕對的可靠性，藉由將 `imq.persist.file.sync.enabled` 特性設爲 True，可要求作業系統能同步寫入資料到永久存放區。這樣可以消除因系統當機而產生的資料遺失，但會影響效能。請注意，雖然資料並未遺失，但是叢集中的其他代理程式將無法使用該資料，因爲目前沒有叢集代理程式共用該資料。當系統備份時，代理程式能可靠地繼續作業。 **參照** 第 62 頁的「永久性服務」。 「Message Queue 管理指南」中的「代理程式特性參照」。

表 B-1　訊息佇列功能 (續)

功能	說明與參照
安全連線	用戶端可以使用安全套接層 (SSL) 標準 (透過 TCP/IP 與 HTTP 傳輸)，安全地傳輸訊息。這些 SSL 型連線服務允許加密訊息在用戶端與代理程式之間傳送。 SSL 支援是以自我簽名的伺服器憑證為基礎。Message Queue 提供可產生私密金鑰/公開密鑰對並將公開密鑰封裝於自我簽名憑證中的公用程式。此憑證會傳送給任何請求連線到代理程式的用戶端，且該用戶端會使用此憑證來設定加密連線。 **使用方式** 1. 產生自我簽名或簽名憑證。 2. 啟用安全服務。 3. 啟動代理程式 4. 配置用戶端安全性連線屬性並執行用戶端。 **參照** 第 28 頁的「連線至代理程式」。 第 63 頁的「安全性服務」。 「Message Queue 管理指南」中的「管理安全性」。 「Message Queue Developer's Guide for Java Clients」。 「Message Queue Developer's Guide for C Clients」。
SOAP 支援	用戶端可以接收 SOAP (XML) 訊息，將訊息包裝成 JMS 訊息，並使用 Message Queue 將其當作JMS 訊息執行交換訊息。 用戶端可以使用特殊 Servlet 來接收 SOAP 訊息；使用公用程式類別將 SOAP 訊息包裝成 JMS 訊息，以及使用其他公用程式類別從JMS 訊息擷取 SOAP 訊息。用戶端可以使用標準 SAAJ 程式庫來組合與分解 SOAP 訊息。 **參照** 第 54 頁的「使用 SOAP 訊息」。 「Message Queue Developer's Guide for Java Clients」中的「Working With SOAP Message」。
執行緒管理	管理員可以指定或指派給任何特定連線服務的最大執行緒與最小執行緒數目。管理員也可以判定連線服務是否使用共用的執行緒模型增加流量，這可讓專屬配置連線的執行緒得以供其他連線使用。 **使用方式** 設定連線服務執行緒相關特性。 **參照** 第 59 頁的「執行緒池管理」。 「Message Queue 管理指南」中的「配置代理程式」。

表 B-1　訊息佇列功能 (續)

功能	說明與參照
可調校的效能	管理員可以設定代理程式特性以調整記憶體使用情況、執行緒資源、訊息流量、連線服務、可靠性參數，以及其他會影響訊息流量與系統效能的元素。 **參照** 第 65 頁的「監視服務」。 「Message Queue 管理指南」中的「監視訊息傳服器」與「分析與調校訊息服務」。

字彙表

本字彙表提供有關使用 Message Queue 時，可能遇到的專有名詞和概念的資訊。如需包含 Sun Java System 中使用的所有專有名詞之字彙表，請參閱 http://docs.sun.com/doc/819-4632

確認　控制用戶端和代理程式間的訊息交換，以確保傳送可靠。有兩種通用類型的確認：用戶端確認和代理程式確認。

管理物件　預先配置的物件 (連線工廠或目標)，封裝了提供者特定的實作細節，由管理員建立以供一個或多個 JMS 用戶端使用。使用管理物件可讓 JMS 用戶端獨立於提供者之外。管理物件由使用 JNDI 查找的 JMS 用戶端放置在 JNDI 名稱空間中，並由其存取。

非同步訊息傳送　一種訊息的交換，其訊息傳送不取決於用戶是否可以接收訊息。換言之，訊息傳送者在繼續其他工作之前無需等待傳回傳送方法。如果訊息用戶忙碌或離線，訊息會先被傳送，然後在用戶準備就緒時接收訊息。

認證　用來確認只有經驗證的使用者才允許設定連線至代理程式的過程。

授權　是訊息服務決定使用者是否可存取訊息服務資源 (如連線服務或目標) 來執行訊息服務支援的特定作業之過程。

代理程式　Message Queue 實體，可管理訊息路由、傳送、永久性、安全性以及記錄，並提供監視和調校效能與資源使用的介面。

用戶端　與使用訊息服務交換訊息的其他用戶端進行互動式操作的應用程式 (或軟體元件)。用戶端可以是產生型用戶端、使用型用戶端，或兩者皆是。

用戶端識別碼　是一種識別碼，此識別碼將連線及其物件與由表示用戶端的 Message Queue 代理程式所維護的狀態相關聯。

用戶端執行階段　爲訊息用戶端提供 Message Queue 訊息服務介面的 Message Queue 軟體。用戶端執行階段支援用戶端將訊息傳送至目標和從目標接收訊息所需的所有作業。

叢集　是透過協作提供可延伸訊息傳送服務的兩個或多個互連代理程式。

連線　是用戶端與代理程式之間的通訊通道，用以傳送有效負載訊息和控制訊息。

連線工廠　用戶端用於建立與代理程式連線的管理物件。這可能是一個 `ConnectionFactory` 物件、`QueueConnectionFactory` 物件或 `TopicConnectionFactory` 物件。

用戶　由階段作業建立的物件 (`MessageConsumer`)，用於接收從目標傳送之訊息。在點對點傳送模型中，用戶爲接收者或瀏覽者 (`QueueReceiver` 或 `QueueBrowser`)；在出版 / 訂閱傳送模型中，用戶即爲訂閱者 (`TopicSubscriber`)。

資料存放區　是永久性儲存代理程式所需資訊 (長期訂閱、有關目標的資料、永久性訊息以及稽核資料) 的資料庫。

停用的訊息　因正常處理或明確的管理員動作之外的原因而從系統上被移除的訊息。過期的訊息、因記憶體超限或因傳送失敗而從目標移除的訊息都可稱爲停用訊息。您可選擇將停用的訊息儲存在停用訊息佇列上。

停用的訊息佇列　是代理程式啓動後自動建立的專用目標，用於儲存停用訊息以備診斷之用。

傳送模式　訊息傳送可靠性的指標：保證訊息可以傳送並成功使用一次且僅爲一次 (永久性傳送模式)，或保證訊息至多傳送一次 (非永久性傳送模式)。

傳送模型　訊息傳送的模型：點對點或出版 / 訂閱。在 JMS 中，每種傳送模型都有單獨的程式設計網域，使用特定用戶端執行階段物件和特定目標類型 (佇列或主題)，以及統一的程式設計網域。

目標　Message Queue 代理程式中的實體目標，產生的訊息會傳送至此目標，以便路由並隨後傳送至用戶。此實體目標由管理物件識別和封裝，用戶端使用此管理物件指定它要產生訊息和 / 或使用訊息的目標。

網域　JMS 用戶端用於程式化 JMS 訊息傳送作業的一組物件。有兩種程式設計網域：一種用於點對點傳送模型，另一種用於出版 / 訂閱傳送模型。

加密　一種保護訊息免於在透過連線傳送時被篡改的機制。

群組　Message Queue 用戶端使用者所屬的群組，旨在授權對連線、目標和特定作業的存取權。

JMS 提供者　爲訊息傳送系統實作 JMS 介面，並新增配置和管理系統所需的管理和控制功能的產品。

訊息服務　一種提供在分散式元件或應用程式間非同步可靠訊息交換的中介軟體服務。它包含代理程式、用戶端執行階段、數個代理程式執行其功能所需的資料存放區，以及配置與監視代理程式和調校效能所需的管理工具。

訊息　訊息用戶端使用的非同步請求、報告或事件。訊息包含標頭 (其他欄位可新增至此) 和內文。訊息標頭指定標準欄位和可選特性。訊息內文包含要傳送的資料。

訊息傳送　是企業應用程式使用的非同步請求、報告或事件的系統，可讓無密切關聯的應用程式可靠安全地傳送資訊。

產生者　由階段作業建立的物件 (MessageProducer)，用於將訊息傳送至目標。在點對點傳送模型中，產生者爲傳送者 (`QueueSender`)；在出版 / 訂閱傳送模型中，產生者爲出版者 (`TopicPublisher`)。

佇列　由管理員建立的可實施點對點傳送模型的物件。佇列始終用於保留訊息，即便使用其訊息的用戶端處於非使用中，亦是如此。佇列被用作產生者與用戶之間的媒介保留位置。

選擇器　用於排序和路由訊息的訊息標頭特性。訊息服務基於訊息選擇器中存在的條件執行訊息過濾與路由。

階段作業　執行傳送和接收訊息的單一執行緒環境。可以爲佇列階段作業或主題階段作業。

主題　由管理員建立的可實作出版 / 訂閱傳送模型的物件。主題可作爲內容階層 (負責收集和分布接收到的訊息) 中的節點來檢視。透過將主題用作媒介，訊息出版者可與訊息用戶保持獨立。

作業事件　必須完整或整體回復的不可分割的工作。

索引

L

M

O

S

T

X

中文

索引

六畫

七畫

八畫

九畫

十畫

十一畫

十二畫

十三畫

十四畫

十七畫

十六畫

十七畫

十八畫

十八章