# Sun Java Enterprise System 5 監視指南

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

文件號碼：820–1593
2007 年 3 月

# 目錄

**前言** ..... 7

**1 Java ES Monitoring 簡介** ..... 13
Monitoring Framework 和 Monitoring Console 元件 ..... 13
Java ES Monitoring 運作方式 ..... 14
Common Monitoring Model (CMM) ..... 14
CMM 設備 ..... 15
節點代理程式 ..... 15
主代理程式 ..... 16
安裝順序建議 ..... 17

**2 啓用和配置 Monitoring Framework** ..... 19
已安裝目錄的配置 ..... 20
在 Access Manager 中使用 Monitoring Framework ..... 21
▼ 在 Access Manager 中啓用監視 ..... 21
在 Application Server 中使用 Monitoring Framework ..... 22
▼ 在 Application Server 中啓用監視 ..... 22
在 Calendar Server 中使用 Monitoring Framework ..... 22
▼ 在 Calendar Server 中啓用監視 ..... 23
在 Directory Server 中使用 Monitoring Framework ..... 23
▼ 在 Directory Server 中啓用監視 ..... 23
在 Instant Messaging 中使用 Monitoring Framework ..... 24
▼ 在 Instant Messaging 中啓用監視 ..... 24
在 Messaging Server 中使用 Monitoring Framework ..... 24
▼ 在 Messaging Server 中啓用監視 ..... 24
在 Portal Server 中使用 Monitoring Framework ..... 25
▼ 在 Portal Server 中啓用監視 ..... 25

在 Web Server 中使用 Monitoring Framework ..... 25
▼ 在 Web Server 中啓用監視 ..... 25
設定 Common Agent Container ..... 26
▼ 啓用 Common Agent Container 的監視功能 ..... 26
Monitoring Framework 疑難排解 ..... 27
在 HP-UX 平台上使用 Monitoring Framework ..... 27
在 Microsoft Windows 上使用 Monitoring Framework ..... 27
▼ 重新啓動節點代理程式 ..... 27
`mfwkadm` 指令 ..... 28
提要 ..... 28
描述 ..... 29
選項 ..... 30
子指令 ..... 30
範例 ..... 39
結束狀態 ..... 44
屬性 ..... 44
另請參閱 ..... 44

**3 安裝和使用 Monitoring Console** ..... 45
安裝 Monitoring Console ..... 45
▼ 使用 Java ES 安裝程式安裝 Monitoring Console ..... 46
▼ 在 Solaris Zone 中安裝 Monitoring Console ..... 46
▼ 配置 Monitoring Console ..... 47
▼ 取消配置 Monitoring Console ..... 47
已安裝目錄的配置 ..... 48
啓動 Monitoring Console ..... 48
▼ 啓動 Monitoring Console ..... 48
▼ 連線至節點代理程式 ..... 49
使用 Monitoring Console ..... 51
▼ 選擇性停用和重新啓用監視 ..... 51
▼ 建立新的監視規則 ..... 52
Monitoring Console 疑難排解 ..... 59

**A CMM 物件引用** ..... 61
CMM 物件簡介 ..... 61

**B 各項元件顯示的受監視物件** ..... 63
Common Agent Container 的設備 ..... 63
Access Manager 的設備 ..... 63
Application Server 的設備 ..... 63
Calendar Server 的設備 ..... 63
Directory Server 的設備 ..... 64
Instant Messaging 的設備 ..... 64
Messaging Server 的設備 ..... 64
Portal Server 的設備 ..... 64
Web Server 的設備 ..... 64

**索引** ..... 65

# 前言

本書說明 Sun Java™ Enterprise System 5 (Java ES) 的新監視功能。監視由 Sun Java System Monitoring Framework 2.0 和 Sun Java System Monitoring Console 1.0 實作。

本指南的程序說明如何針對各個已安裝元件配置和啓用 Monitoring Framework，並說明如何在 Monitoring Console 中檢視所有的受監視資料。本指南未說明此架構以外個別元件的記錄檔和其他監視機制。

## 本書適用對象

本書適用於下列對象：

- 需要針對 Java ES 部署設計維護規劃的軟體架構設計師。
- 執行 Java ES 安裝和配置的系統管理員。
- 監視和維護 Java ES 部署的系統管理員和技術人員。

## 閱讀本書之前

您應該熟知下節所述 Java ES 文件集中的文件。您還應瞭解需要監視的 Java ES 元件設計和功能。

此外，如果您想要安裝和配置監視元件，請先完成所有其他元件的安裝。在執行任何安裝或配置之前，您應該先參閱「適用於 UNIX 的 Sun Java Enterprise System 5 版本說明」。

## Java ES 文件集

Java ES 文件集介紹部署規劃及系統安裝。系統文件的 URL 是 `http://docs.sun.com/coll/1286.2` 和 `http://docs.sun.com/coll/1412.2`。如需 Java ES 的簡介，請參閱下表依序列出的書籍。

表 P–1 Java Enterprise System 文件

| 文件標題 | 目錄 |
| --- | --- |
| 「適用於 UNIX 的 Sun Java Enterprise System 5 版本說明」<br>「Sun Java Enterprise System 5 Release Notes for Microsoft Windows」 | 包含有關 Java ES 的最新資訊，包括已知問題。此外，元件有各自的版本說明，如版本說明集合 (http://docs.sun.com/coll/1315.2 和 http://docs.sun.com/coll/1426.2) 所列。 |
| 「Sun Java Enterprise System 5 技術摘要」 | 介紹 Java ES 的技術及概念基礎。還會介紹元件、架構、流程和功能。 |
| 「Sun Java Enterprise System Deployment Planning Guide」 | 介紹規劃及設計以 Java ES 爲基礎的企業部署解決方案。說明部署計劃及設計的一些基本概念及原則、介紹解決方案生命週期，以及提供在部署以 Java ES 爲基礎的解決方案時所使用的高層級範例和策略。 |
| 「Sun Java Enterprise System 5 安裝規劃指南」 | 協助您發展 Java ES 部署之硬體、作業系統及網站方面的實作規格。介紹諸如要在安裝及部署計劃中處理的元件相依性等問題。 |
| 「適用於 UNIX 的 Sun Java Enterprise System 5 安裝指南」<br>「Sun Java Enterprise System 5 Installation Guide for Microsoft Windows」 | 引導您進行安裝 Java ES 的程序。還介紹如何在安裝後配置元件，以及如何確認它們是否運作正常。 |
| 「適用於 UNIX 的 Sun Java Enterprise System 5 安裝參照」 | 提供更多關於配置參數的資訊，提供工作表做爲配置規劃之用，並列出參考資料，例如 Solaris 作業系統和 Linux 作業環境的預設目錄和連接埠號。 |
| 適用於 UNIX 的 Sun Java Enterprise System 5 升級指南<br>「Sun Java Enterprise System 5 Upgrade Guide for Microsoft Windows」 | 提供從先前安裝版本升級至 Java ES 5 的指示。 |
| 「Sun Java Enterprise System 5 監視指南」 | 提供爲各個產品元件設定監視架構的說明，以及使用 Monitoring Console 檢視即時資料和建立監視規則的說明。 |
| 「Sun Java Enterprise System Glossary」 | 定義 Java ES 文件中使用的術語。 |

# 預設路徑和檔案名稱

下表說明實作監視的 Java ES 元件預設路徑和檔案名稱。

**表 P–2** 預設路徑和檔案名稱

| 定位字元 | 描述 | 預設值 |
|---|---|---|
| *mfwk-base* | 表示自動安裝 Monitoring Framework 共用元件的目錄。此路徑也會做爲配置目錄的一部分。 | Solaris 系統：`/opt/SUNWmfwk`<br>Linux 系統：`/opt/sun/mfwk` |
| *MConsole-base* | 表示爲 Monitoring Console 選擇的安裝目錄。 | Solaris 系統：`/opt/SUNWjesmc`<br>Linux 系統：`/opt/sun/jesmc` |
| *WebConsole-base* | 表示自動安裝 Web 主控台共用元件的目錄。 | Solaris 系統：`/etc/webconsole/console`<br>Linux 系統：`/etc/opt/webconsole/console` |
| *AccessMgr-base* | 表示爲 Sun Java System Access Manager 選擇的安裝目錄。 | Solaris 系統：`/opt/SUNWam`<br>Linux 系統：`/opt/sun/identity` |
| *AppServer-base* | 表示爲 Sun Java System Application Server 選擇的安裝目錄。 | Solaris 系統：`/opt/SUNWappserver/appserver`<br>Linux 系統：`/opt/sun/appserver` |
| *CalServ-base* | 表示爲 Sun Java System Calendar Server 選擇的安裝目錄。 | Solaris 系統：`/opt/SUNWics5`<br>Linux 系統：`/opt/sun/calendar` |
| *DirServ-base* | 表示爲 Sun Java System Directory Server 選擇的安裝目錄。 | Solaris 系統：`/opt/SUNWdsee/ds6`<br>Linux 系統：`/opt/sun/ds6` |
| *IM-base* | 表示爲 Sun Java System Instant Messaging 選擇的安裝目錄。 | Solaris 系統：`/opt/SUNWiim`<br>Linux 系統：`/opt/sun/im` |
| *MsgServ-base* | 表示爲 Sun Java System Messaging Server 選擇的安裝目錄。 | Solaris 系統：`/opt/SUNWmsgsr`<br>Linux 系統：`/opt/sun/messaging` |
| *Portal-base* | 表示爲 Sun Java System Portal Server 選擇的安裝目錄。 | Solaris 系統：`/opt/SUNWportal`<br>Linux 系統：`/opt/sun/portal` |
| *WebServer-base* | 表示爲 Sun Java SystemWeb Server 選擇的安裝目錄。 | Solaris 系統：`/opt/SUNWwbsvr7`<br>Linux 系統：`/opt/sun/webserver` |

# 印刷排版慣例

下表描述本書中所使用的印刷排版慣例。

**表 P–3** 印刷排版慣例

| 字體 | 意義 | 範例 |
| --- | --- | --- |
| `AaBbCc123` | 指令、檔案及目錄的名稱；螢幕書面輸出。 | 請編輯您的 `.login` 檔案。<br>請使用 `ls -a` 列出所有檔案。<br>**電腦名稱**`% you have mail.` |
| **`AaBbCc123`** | 您所鍵入的內容 (與螢幕書面輸出相區別)。 | **電腦名稱**`% su`<br>`Password:` |
| **術語強調變數** | 新的字彙或術語、要強調的詞。將用實際的名稱或數值取代的指令行變數。 | 要刪除檔案，請鍵入 `rm` **檔案名稱**。 |
| *AaBbCc123* | 保留未譯的新的字彙或術語、要強調的詞。 | 請參閱使用者指南中的第 6 章。<br>**快取**是儲存在本機上的副本。<br>**不**儲存檔案。 |

# 指令範例中的 Shell 提示符號

下表顯示預設系統提示及超級使用者提示。

**表 P–4** Shell 提示

| Shell | 提示 |
| --- | --- |
| 在 UNIX 及 Linux 系統上的 C Shell | **電腦名稱**`%` |
| 在 UNIX 及 Linux 系統上的 C Shell 超級使用者 | **電腦名稱**`#` |
| 在 UNIX 及 Linux 系統上的 Bourne Shell 及 Korn Shell | `$` |
| 在 UNIX 及 Linux 系統上的 Bourne Shell 及 Korn Shell | `#` |
| Microsoft Windows 指令行 | `C:\` |

# 符號慣例

下表說明本書可能使用的符號。

表 P–5 符號慣例

| 符號 | 描述 | 範例 | 意義 |
|---|---|---|---|
| `[ ]` | 包含可選引數和指令選項。 | `ls [-l]` | 不需要 `-l` 選項。 |
| `{ \| }` | 包含所需指令選項的一組選擇。 | `-d {y\|n}` | `-d` 選項需要您使用 `y` 引數或是 `n` 引數。 |
| `${ }` | 指出變數參照。 | `${com.sun.javaRoot}` | 參照 `com.sun.javaRoot` 變數的值。 |
| - | 結合多個同步按鍵。 | Control-A | 按下 Control 鍵同時按住 A 鍵。 |
| + | 結合多個連續按鍵。 | Ctrl+A+N | 按下 Control 鍵、放掉然後再按下後續的鍵。 |
| → | 指出圖形使用者介面中的功能表項目選項。 | [檔案] → [新增] → [範本] | 從 [檔案] 功能表選擇 [新增]。從 [新增] 子功能表選擇 [範本]。 |

# 文件、支援與訓練

Sun 網站提供了下列附加資源的相關資訊：

- 文件 (`http://www.sun.com/documentation/`)
- 支援 (`http://www.sun.com/support/`)
- 培訓 (`http://www.sun.com/training/`)

# 搜尋 Sun 產品文件

除了從 docs.sun.com℠ 網站搜尋 Sun 產品文件外，還可藉由在搜尋欄位中鍵入下列語法來使用搜尋引擎：

*search-term* `site:docs.sun.com`

例如，若要搜尋「broker」，請鍵入下列語法：

```
broker site:docs.sun.com
```

若要將其他 Sun 網站納入您的搜尋中 (例如，java.sun.com、www.sun.com 和 developers.sun.com)，請在搜尋欄位中用 `sun.com` 來替代 `docs.sun.com`。

# 協力廠商網站參考

本文件中提供了協力廠商 URL 以供參考，另亦提供其他相關的資訊。

**備註 –** Sun 對本文件中提到的協力廠商網站的可用性不承擔任何責任。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，Sun 並不表示認可，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。若要分享您的意見，請至 `http://docs.sun.com`，然後按一下 [Send Comments (傳送意見)]。請在線上表單中提供完整的文件標題與文件號碼。文件號碼是一個七位或九位的數字，可以在書的標題頁面或文件的 URL 中找到。例如，本書的文件號碼爲 820-1593。

◆ ◆ ◆ 第 1 章

# 1 Java ES Monitoring 簡介

本指南說明 Sun Java™ Enterprise System (Java ES) 的 Monitoring Framework 2.0 和 Monitoring Console 1.0 元件。這些元件會共同實作發行版本 5 中引入的新監視功能。

本指南的程序說明如何針對各個已安裝元件啓用 Monitoring Framework，並說明如何在 Monitoring Console 中檢視所有受監視的資料。本指南未說明本架構以外個別元件可能實作之記錄檔、錯誤訊息以及其他監視機制。Monitoring Framework 或 Monitoring Console 都不會對受監視元件提供管理或控制功能。如需有關元件管理的資訊，請參閱產品隨附的文件。

本章介紹監視概念，並說明 Monitoring Framework 的架構。

本章包含以下各節：


- 第 13 頁的 「Monitoring Framework 和 Monitoring Console 元件」
- 第 14 頁的 「Java ES Monitoring 運作方式」
- 第 17 頁的 「安裝順序建議」


## Monitoring Framework 和 Monitoring Console 元件

Sun Java System Monitoring Framework 提供設備元件所需的基礎架構，並顯示其中的屬性以供觀察。其中會根據業界標準的共用資訊模型 (CIM) 規格，定義稱爲 *Common Monitoring Model (CMM)* 的受監視物件階層。各個產品元件會顯示表示可觀察屬性的物件，而且**節點代理程式**會將一部主機上的多個元件集合爲單一檢視。Monitoring Framework 也提供機制以收集作業統計資料，並根據使用者定義的臨界值定義警示。

Sun Java System Monitoring Console 是監視 Java ES 元件的圖形介面。其中包括**主代理程式**，可連線至 Java ES 部署中的所有節點代理程式。Monitoring Console 是網路型應用程式，需要藉助 Sun Java SystemWeb Console 才能夠經由 HTTP 從各處連線。主畫面提供所有已啓用元件的摘要狀態，包括已觸發的所有警示。然後您可以存取各個元件的受監視物件階層，檢視所有受監視屬性的詳細狀態和即時值。Monitoring Console 介面可供您顯示任何警示的詳細資訊、確認警示，或根據任何屬性建立新的監視規則。

# Java ES Monitoring 運作方式

整個監視程序包含收集執行階段資料、顯示資料及運算服務品質標準，以便系統管理員可評估效能，並收到警示通知。在執行階段作業期間，管理員只需要與 Monitoring Console 互動，即可檢視效能統計資料、建立規則以進行自動監視和確認警示。但是對於配置、疑難排解和進階監視而言，這可協助瞭解 Monitoring Framework 的架構，以及與 Monitoring Console 連線的方式。

Java ES 中的監視基於下列概念：

- Common Monitoring Model (CMM) 可確保所有 Java ES 元件都顯示可比對屬性的統一物件和值。
- CMM 介面定義的 Java 物件會爲產品元件提供標準化的設備。
- 節點代理程式會顯示系統上所有已安裝元件的受監視物件，並管理這些物件的統計資料、規則和警示。
- 獨立主機上的主代理程式會收集所有節點代理程式的所有受監視物件，並且將資料提供給 Monitoring Console。

以下各節將詳細說明上述各項關於監視架構的概念。

## Common Monitoring Model (CMM)

標準化監視機制的基礎，是定義哪些物件需要受到監視，並且在所有受監視元件上採用這些物件。爲達成此目標，監視架構將 Common Monitoring Model (CMM) 定義爲分散式管理專案小組 (DMTF) 所維護之共用資訊模型 (CIM) 的延伸。CMM 既是可指定受監視物件 (例如電腦、應用程式等等) 的資訊模型，也是可指定統一值 (例如作業狀態值) 的資料模型。就資訊模型而言，CMM 也會定義物件的屬性，例如服務所處理的請求數量，以及物件之間的關係，例如服務由某部電腦代管。

由於有 CMM，所以即使基礎實作不同，所有產品元件的概念 (例如應用程式、服務、存取點等等) 仍然會一樣。舉例來說，Web Server 可能會顯示處理 HTTP 請求的服務，而 Directory Server 可能會顯示處理 LDAP 請求的服務。但標準物件會擷取這兩項功能的共通點，例如測量已處理請求數量的功能、在指定時間內回應請求的平均時間等。

以外，某些資料值已經過標準化，所以其意義在整個系統中是一致的。例如，不管受監視的是哪個產品元件，作業狀態 DEGRADED 表示服務仍然可用，但效能都已大幅度降低。

CMM 規格納入於設備使用的 Java 介面和類別中，如附錄 A 中所述。

## CMM 設備

在 Monitoring Framework 中，設備是實作 CMM 定義的一組 Java 介面和類別。對於 Java ES 的新監視功能，產品元件已提供程式碼實例化 CMM 物件，並透過受監視物件的屬性顯示即時值。各個元件實作的 CMM 物件會決定可監視的項目，因此，某些元件會比其他元件顯示較少的屬性。附錄 B 中提供了一份清單，列出各個產品元件所顯示的受監視物件和屬性。

## 節點代理程式

在監視術語中，節點表示由唯一完全合格之網域名稱或 IP 位址所識別的單一邏輯主機。節點可以是整個系統，也可以是配置爲虛擬系統的 Solaris Zone。節點代理程式會與該主機上配備的所有元件相互通訊，並顯示其中所有的受監視物件。節點代理程式也會管理所有邏輯，以收集效能統計資料、監視規則中定義的臨界值，並針對其中所包含的受監視物件產生警示。

下圖表示單一主機上的節點代理程式內容，其中包含三個 Java ES 產品元件的實例。還顯示如何在節點代理程式中實例化設備，以顯示產品元件所提供的值。

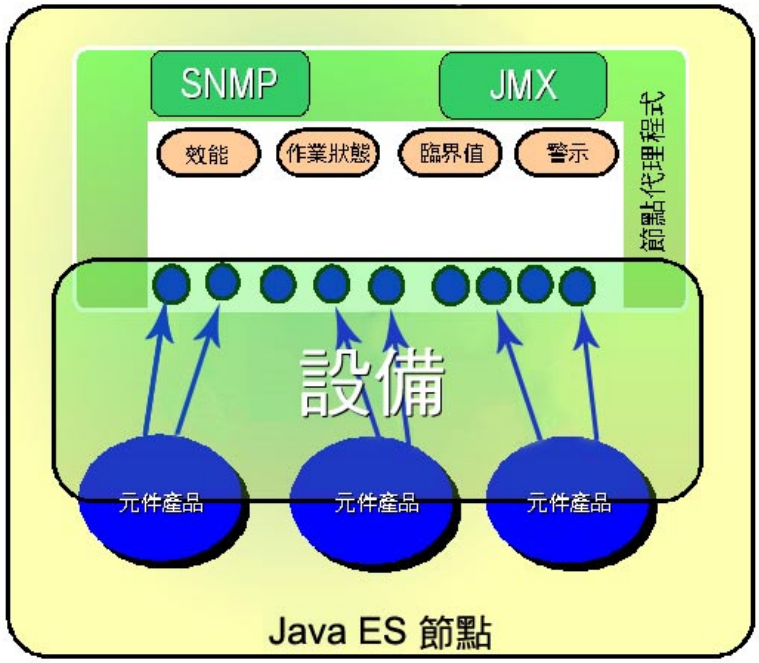



圖 1–1　節點代理程式圖

節點代理程式已實作爲載入 Common Agent Container (其本身爲 Java 虛擬機器) 中的模組。節點代理程式根據 Java Management Extensions (JMX) 進行實作，JMX 爲監視和遠端管理的標準 Java 延伸。任何啓用 JMX 的監視應用程式只要支援 CMM，都可以存取節點代理程式中的受監視物件。節點代理程式也可以使用 JMX 功能，透過簡易網路監視協定 (SNMP) 顯示某些受監視物件。

# 主代理程式

主代理程式會在 Monitoring Console 安裝時部署在獨立機器上。主代理程式是以所有節點的名稱和位址加以配置，因此可集合所有節點代理程式的受監視物件。主代理程式也以 JMX 爲基礎，與節點代理程式相互通訊，並且會載入於本機 Common Agent Container。

下圖表示連線至兩個節點的主代理程式。Monitoring Console 會連線至主代理程式，監視各節點上的三個元件。如果需要使用 SNMP 進行監視，由於主代理程式不會集合 SNMP 屬性，所以您必須分別連線至各個節點。主代理程式僅能搭配 Monitoring Console，其他監視應用程式無法存取。

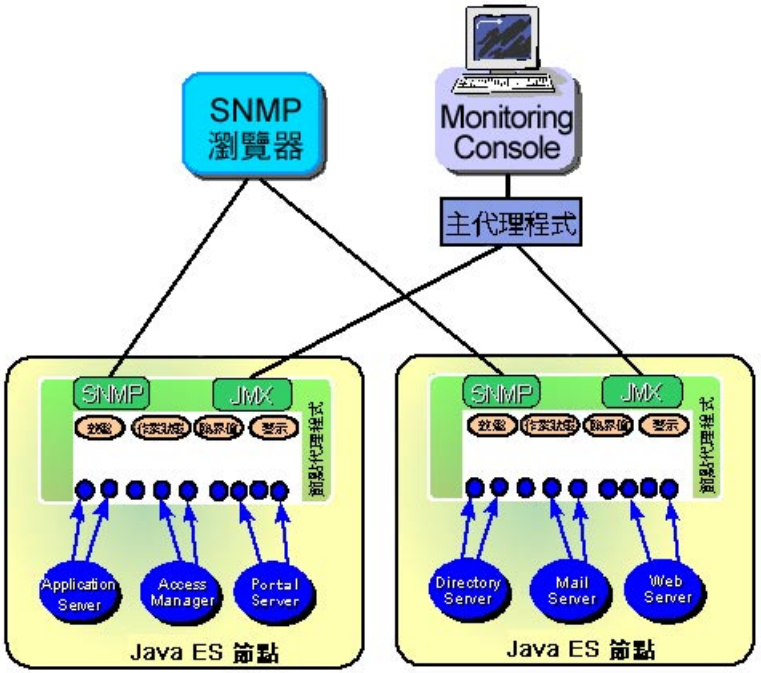



**圖 1–2** 整體監視架構圖

# 安裝順序建議

如果您選擇評估或部署 Java ES 的監視功能，按照下列順序進行安裝是最容易的一種選擇：

1. 根據「適用於 UNIX 的 Sun Java Enterprise System 5 安裝指南」中的建議和指示，在部署中安裝和配置所有元件。
2. 依第 2 章所述，爲您所有的受監視元件啓用和配置 Monitoring Framework。
3. 依第 3 章所述，在獨立主機上安裝 Monitoring Console 並啓動主代理程式，然後啓動 Web 伺服器。所有受監視元件此時應該會出現在 Monitoring Console 中，並受到監視。

---

**備註 –** 由於此發行版本中的節點代理程式和主代理程式不相容，因此 Monitoring Console 必須安裝在不包含任何其他 Java ES 元件的主機上。如需詳細資訊，請參閱第 59 頁的 「Monitoring Console 疑難排解」。

---

啓用監視之後，只要修改部署的元件，就必須爲主代理程式重新啓動容器，並爲 Monitoring Console 重新啓動 Web 伺服器，如第 27 頁的 「Monitoring Framework 疑難排解」所述。

◆ ◆ ◆ 第 2 章

# 2 啓用和配置 Monitoring Framework

如第 14 頁的 「Java ES Monitoring 運作方式」 所述，Monitoring Framework 提供各個受監視元件所需的設備和節點代理程式。因此，Monitoring Framework 是共用元件，只要使用 Java Enterprise System 安裝程式安裝受監視元件，它就會自動安裝。

然而，許多受監視元件不會預設啓用監視，而且某些需要進一步的配置，才會出現在節點代理程式中。請針對您所安裝的各項產品元件，依本章的程序進行操作。

---

**備註 –** 在進行本章的程序之前，建議先安裝和配置要在指定主機上執行的所有產品元件。在執行任何安裝或配置之前，您應該先參閱「適用於 UNIX 的 Sun Java Enterprise System 5 版本說明」。

這些程序會使用 `mfwksetup` 指令 (通常不需要這個指令，因此未加以說明)。

---

本章包含以下各節：


- 第 20 頁的 「已安裝目錄的配置」
- 第 21 頁的 「在 Access Manager 中使用 Monitoring Framework」
- 第 22 頁的 「在 Application Server 中使用 Monitoring Framework」
- 第 22 頁的 「在 Calendar Server 中使用 Monitoring Framework」
- 第 23 頁的 「在 Directory Server 中使用 Monitoring Framework」
- 第 24 頁的 「在 Instant Messaging 中使用 Monitoring Framework」
- 第 24 頁的 「在 Messaging Server 中使用 Monitoring Framework」
- 第 25 頁的 「在 Portal Server 中使用 Monitoring Framework」
- 第 25 頁的 「在 Web Server 中使用 Monitoring Framework」
- 第 26 頁的 「設定 Common Agent Container」
- 第 27 頁的 「Monitoring Framework 疑難排解」
- 第 28 頁的 「`mfwkadm` 指令」

# 已安裝目錄的配置

Monitoring Framework 是共用元件，會在必要時自動安裝。如需作業系統上所安裝的套裝軟體名稱，請參閱「適用於 UNIX 的 Sun Java Enterprise System 5 安裝參照」中的第 5 章「可安裝的套裝軟體清單」。下表說明 Monitoring Framework 套裝軟體中的目錄。預設安裝目錄 *mfwk-base* 具有下列意義，如第 9 頁的 「預設路徑和檔案名稱」所述：

- Solaris 系統：`/opt/SUNWmfwk`
- Linux 系統：`/opt/sun/mfwk`

**表 2–1** Monitoring Framework 使用的目錄

| 路徑 | 內容說明 |
|---|---|
| *mfwk-base*`/config` | 配置檔案的範本 |
| Solaris 系統：*mfwk-base*`/lib`<br>Linux 系統：*mfwk-base*`/share/lib` | Java 歸檔 (`.jar`) 檔案 |
| Solaris 系統：*mfwk-base*`/lib`<br>Linux 系統：*mfwk-base*`/share/lib` | 32 位元執行階段程式庫檔案 (`.so`) |
| Solaris SPARC® 系統：*mfwk-base*`/lib/sparcv9`<br>Solaris x86 系統：*mfwk-base*`/amd64`<br>Linux 系統：*mfwk-base*`/lib64` | 64 位元執行階段程式庫檔案 (`.so`) |
| *mfwk-base*`/bin` | 公用程序檔和私用二進位檔案 |
| *mfwk-base*`/mib` | Monitoring Framework 所支援的 SNMP MIB 文字版本 |
| *mfwk-base*`/xml` | 適用於代理程式和主代理程式的 Common Agent Container 描述元範本 (由 `mfwksetup` 指令部署) |
| *mfwk-base*`/dtd` | OSS/J 功能的 DTD 檔案。 |
| `/etc/`*mfwk-base*`/config` | 配置檔案，包含安全性相關的配置檔案 |
| `/etc/`*mfwk-base*`/xml` | 適用於代理程式和範例的 Common Agent Container 描述元 |
| `/var/`*mfwk-base*`/logs` | Monitoring Framework 的記錄檔 |
| `/var/`*mfwk-base*`/reports` | 監視規則報告的基底目錄 |
| `/var/`*mfwk-base*`/alarms` | 警示檔案的儲存庫 |

# 在 Access Manager 中使用 Monitoring Framework

依預設，Access Manager 中會啓用監視，但因爲有限制，所以受監視元件無法出現在 Monitoring Console 中。

請參閱第 63 頁的 「Access Manager 的設備」，以取得可監視的物件和屬性清單。

## ▼ 在 Access Manager 中啓用監視

1 **下列指令可暫時在 Access Manager 中停用監視：**

```
cacaoadm unregister-module com.sun.cmm.am.xml
cacaoadm restart
```

2 **開啓 Access Manager XML 描述元檔案進行編輯：**

```
vi /etc/AccessMgr-base/config/com.sun.cmm.am.xml
```

3 **找到包含下列項目的指令行：**

```
<param-name>Product Name</param-name>
<param-value>Access Manager</param-value>
```

並且將第二行修改爲：

```
<param-value>Java ES Access Manager</param-value>
```

儲存檔案，並結束編輯器。

4 **註冊經過修改的 XML 模組：**

```
mfwk-base/bin/mfwksetup -u /etc/AccessMgr-base/config/com.sun.cmm.am.xml
mfwk-base/bin/mfwksetup -r /etc/AccessMgr-base/config/com.sun.cmm.am.xml
```

5 **重新啓動 Common Agent Container：**

```
cacaoadm restart
```

**疑難排解** 由於協力廠商 Web 容器的運作方式未經過測試，因此，在 Websphere 或 Weblogic 中部署 Access Manager 時，會預設停用監視。雖然此配置不受支援，但是您可以如第 51 頁的 「選擇性停用和重新啓用監視」所述啓用監視。

# 在 Application Server 中使用 Monitoring Framework

請參閱第 63 頁的 「Application Server 的設備」，以取得可監視的物件和屬性清單。

## ▼ 在 Application Server 中啓用監視

1. **編輯 `/var/`*AppServer-base*`/domains/domain1/config/domain.xml` 檔案，並且將所有的 `module-monitoring-level` 設定從 `OFF` 變更為 `HIGH`。或者：**

   a. **登入位於 `https://`*hostname*`:4849` 的 Application Server 管理主控台**

   b. **選取 [配置]，然後選取 [伺服器配置 (管理配置)]**

   c. **將監視值設定為 `HIGH`**

   d. **將所有其他值設定為 `HIGH`**

2. **使用下列指令重新啓動 Application Server：**

   ```
   cd AppServer-base/appserv/bin
   asadmin stop-domain domain1
   asadmin start-domain user myUser domain1
   ```

   在出現提示時，輸入 *myUser* 的密碼。

3. **如果您已經以 Application Server 部署 Portal Server 的實例，並進行監視，則重新啓動 Application Server 會干擾 Portal Server 監視。若要讓 Portal Server 實例出現在 Monitoring Console 中，您必須使用瀏覽器連線至入口頁面。例如，載入 `http://portalserv.example.com:8080/portal` 頁面可監視 `portalserv.example.com`。**

**疑難排解** 由於受到限制，當 Application Server 當機或關閉時，將從 Monitoring Framework 中移除 Application Server 的受監視物件。發生此狀況時，Application Server 不會出現在 Monitoring Console 中，而且無法再受監視。

# 在 Calendar Server 中使用 Monitoring Framework

請參閱第 63 頁的 「Calendar Server 的設備」，以取得可監視的物件和屬性清單。

### ▼ 在 Calendar Server 中啓用監視

1. **編輯 `ics.conf` 檔案：**

   ```
   vi CalServ-base/cal/config/ics.conf
   ```

2. **增加下列行：**

   ```
   local.mfagent.enable="yes"
   ```

3. **註冊 Calendar Server XML 模組：**

   ```
   mfwk-base/bin/mfwksetup -r /opt/SUNWics5/cal/lib/com.sun.cmm.cs.xml
   ```

4. **設定 `LD_LIBRARY_PATH` 環境變數如下：**

   ```
   LD_LIBRARY_PATH=mfwk-base/lib:$LD_LIBRARY_PATH
   export LD_LIBRARY_PATH
   ```

5. **重新啓動 Calendar Server：**

   ```
   cd CalServ-base/cal/sbin/
   ./stop-cal
   ./start-cal
   ```

6. **重新啓動 Common Agent Container：**

   ```
   cacaoadm restart
   ```

## 在 Directory Server 中使用 Monitoring Framework

請參閱第 64 頁的 「Directory Server 的設備」，以取得可監視的物件和屬性清單。

### ▼ 在 Directory Server 中啓用監視

1. **建立暫存密碼檔案：**

   ```
   echo –n password > /tmp/pwd
   ```

2. **使用下列指令啓用監視外掛程式：**

   ```
   DirServ-base/ds6/bin/dscfg enable-plugin -e -p 389 -w /tmp/pwd  "Monitoring Plugin"
   ```

3. **重新啓動 Directory Server：**

   ```
   cd DirServ-base/ds6/bin
   ./dsadm restart /var/DirServ-base/DSinstance/
   ```

# 在 Instant Messaging 中使用 Monitoring Framework

請參閱第 64 頁的 「Instant Messaging 的設備」，以取得可監視的物件和屬性清單。

## ▼ 在 Instant Messaging 中啓用監視

1. **開啓 Instant Messaging XML 描述元檔案進行編輯：**

   ```
   vi /etc/IM-base/default/com.sun.cmm.im.xml
   ```

2. **將安裝位置從 *IM-base* 變更為 `/etc/IM-base/default`。**

3. **註冊經過修改的 Instant Messaging XML 描述元：**

   ```
   mfwk-base/bin/mfwksetup -r /etc/IM-base/default/com.sun.cmm.im.xml
   ```

4. **將下列行增加至 *IM-base*`/config/iim.conf` 檔案中，以啓用設備：**

   ```
   iim_server.monitor.enable = true
   ```

5. **使用下列指令重新啓動 Instant Messaging：**

   ```
   cd IM-base/sbin
   ./imadmin stop
   ./imadmin start
   ```

6. **重新啓動 Common Agent Container：**

   ```
   cacaoadm restart
   ```

# 在 Messaging Server 中使用 Monitoring Framework

請參閱第 64 頁的 「Messaging Server 的設備」，以取得可監視的物件和屬性清單。

## ▼ 在 Messaging Server 中啓用監視

1. **使用下列指令啓用設備：**

   ```
   MsgServ-base/sbin/configutil -o local.mfagent.enable -v 1
   ```

2. **註冊 Messaging Server XML 模組：**

   ```
   mfwk-base/bin/mfwksetup -r MsgServ-base/lib/com.sun.cmm.ms.xml
   ```

3. **重新啓動 Messaging Server：**

```
cd MsgServ-base/sbin
./stop-msg
./start-msg
```

4. **重新啓動 Common Agent Container：**

```
cacaoadm restart
```

# 在 Portal Server 中使用 Monitoring Framework

請參閱第 64 頁的 「Portal Server 的設備」，以取得可監視的物件和屬性清單。

## ▼ 在 Portal Server 中啓用監視

- **若要啓用 Portal Server，使用者必須登入至**

```
http://FullHostname:8080/portal/dt
```

此舉會編譯入口網站 JSP，建立可供監視的入口網站實例。

**疑難排解** 只要重新啓動 Portal Server 所在的 Application Server，您都必須使用此程序手動重新啓用監視。

# 在 Web Server 中使用 Monitoring Framework

請參閱第 64 頁的 「Web Server 的設備」，以取得可監視的物件和屬性清單。

## ▼ 在 Web Server 中啓用監視

1. **使用下列指令啓動 Web Server：**

```
cd /var/WebServer-base/https-FullHostname/bin
./startserv
```

2. **啓動管理伺服器：**

```
cd /var/WebServer-base/admin-server/bin
./startserv
```

# 設定 Common Agent Container

Common Agent Container 是另一個共用元件，Monitoring Framework 必須使用這個元件，才能夠執行節點代理程式。根據安裝順序的不同，Common Agent Container 可能會停止，需要重新啓動。此外，Common Agent Container 已經過配置，也可以受到監視。如需有關受監視物件的說明，請參閱第 63 頁的「Common Agent Container 的設備」。

若要檢查 Common Agent Container 和節點代理程式是否已啓動，請執行下列指令：

```
cacaoadm status
```

如果出現類似下列的訊息，則表示節點代理程式在執行中：

```
default instance is DISABLED at system startup.
Smf monitoring process:
26996
Uptime: 0 day(s), 0:57
```

如果出現類似下列的訊息，則表示節點代理程式未執行：

```
default instance is DISABLED at system startup.
default instance is not running.
```

## ▼ 啓用 Common Agent Container 的監視功能

Common Agent Container 是共用元件，其中有設備可進行監視。如第 15 頁的「節點代理程式」所述，主機或區域中的所有 Java ES 元件都共用 Common Agent Container 和節點代理程式。在需要監視 Common Agent Container 的部署中，以超級使用者身份在各個邏輯主機上執行此作業。

1. **如果 Common Agent Container 已執行，請使用下列指令停止：**

   ```
   cacaoadm stop
   ```

2. **啓用容器自身的設備：**

   ```
   cacaoadm set-param enable-instrumentation=true
   ```

3. **檢查剛才設定的參數值，並重新啓動 Common Agent Container：**

   ```
   cacaoadm get-param enable-instrumentation
   cacaoadm start
   ```

4. **建立金鑰密碼：**

   ```
   echo –n password > /etc/mfwk-base/config/security/password.cacao
   ```

5 **產生您的金鑰：**

```
mfwk-base/bin/cpgenkey -n cacao -p /etc/mfwk-base/config/security/password.cacao
```

6 **註冊 Common Agent Container 本身的監視模組：**

```
cacaoadm register-module /usr/lib/cacao/ext/instrum/config/com.sun.cacao.instrum.xml
cacaoadm register-module /usr/lib/cacao/ext/instrum_jesmf/config/com.sun.cacao.instrum.jesmf.xml
cacaoadm register-module /usr/lib/cacao/ext/instrum_jesmf/config/com.sun.cacao.cmm.xml
```

# Monitoring Framework 疑難排解

另請參閱「適用於 UNIX 的 Sun Java Enterprise System 5 版本說明」中所列的已知問題。

## 在 HP-UX 平台上使用 Monitoring Framework

依預設，HP-UX 上的 Java 虛擬機器 (JVM) 未針對 Monitoring Framework 所需的密集作業處理預設進行調校，因此可能會導致 `OutOfMemory` 異常狀況。若要配置 JVM，請從下列位置下載 `HPjconfig` 工具，並且執行該工具：`http://h21007.ww2.hp.com/dspp/tech/tech_TechDocumentDetailPage_IDX/1,1701,1620,00.html`。

## 在 Microsoft Windows 上使用 Monitoring Framework

透過 Monitoring Framework 在 Windows 平台上監視 Java ES 元件的方式完全受到支援，不過其中略有差異。例如，您必須升級至 Java 1.5 或更新版，以避免某些已知問題。對於其他已知問題，請參閱「Sun Java Enterprise System 5 Release Notes for Microsoft Windows」。

## ▼ 重新啓動節點代理程式

如果需要重新啓動節點代理程式所在的 Common Agent Container，透過該節點代理程式監視的元件將不會顯示在 Monitoring Console 中，直到您執行以下程序爲止：

1 **重新啓動節點代理程式所在的 Common Agent Container：**

```
cacaoadm restart
```

2 **重新啓動主代理程式所在的 Common Agent Container。主代理程式會在主機的 Monitoring Framework 中執行，或在已安裝 Monitoring Console 的區域中執行。**

```
cacaoadm restart
```

主代理程式會自動重新連線至先前監視的所有節點代理程式。

3 **重新啓動 Monitoring Console 所在的 Web 伺服器：**

```
/usr/sbin/smcwebserver restart
```

# mfwkadm 指令

本小節複製了 mfwkadm 指令 (線上手冊 1M 小節中的維護指令) 的線上手冊。使用此指令可管理節點代理程式的內容，包括所有受監視元件的模組，以及您在此節點上定義的所有監視規則 (又稱爲工作)。線上手冊中的某些術語和說明已經過修改，以符合本文件中的術語和說明。

## 提要

**mfwkadm** --help

**mfwkadm start**

**mfwkadm stop**

**mfwkadm restart**

**mfwkadm list-params**

**mfwkadm list-modules**

**mfwkadm info** *runningInstance*

### 效能監視

**mfwkadm pm-job observable-classes**

**mfwkadm pm-job observable-objects** [class=*objectClass*] [domain= *objectDomain*]

**mfwkadm pm-job observable-attributes** class=*objectClass*

**mfwkadm pm-job list**

**mfwkadm pm-job info** *jobName*

**mfwkadm pm-job create** *jobName* granularity=*integerValue* object=*objectName* [object=*objectName* ...]

**mfwkadm pm-job delete** *jobName*

**mfwkadm pm-job suspend** *jobName*

**mfwkadm pm-job resume** *jobName*

### 作業狀態監視

**mfwkadm opstat-job observable-classes**

**mfwkadm opstat-job observable-objects** [class=*objectClass*] [domain= *objectDomain*]

**mfwkadm opstat-job observable-attributes** class= *objectClass*

**mfwkadm opstat-job list**

**mfwkadm opstat-job info** *jobName*

**mfwkadm opstat-job create** *jobName* granularity= *integerValue* object=*objectName* [object=*objectName* ...]

**mfwkadm opstat-job delete** *jobName*

**mfwkadm opstat-job suspend** *jobName*

**mfwkadm opstat-job resume** *jobName*

### 臨界值監視

**mfwkadm thrsh-job observable-classes**

**mfwkadm thrsh-job observable-objects** [class=*objectClass*] [domain= *objectDomain*]

**mfwkadm thrsh-job observable-attributes** class=*objectClass*

**mfwkadm thrsh-job list**

**mfwkadm thrsh-job info** *jobName*

**mfwkadm thrsh-job create** *jobName* granularity=*integerValue* attributeName=*attributeName* attributeType=*attributeType* thresholdValue=*thresholdValue* thresholdOffset=*offsetValue* thresholdDirection=[ RISING | FALLING ] object=*objectName*

**mfwkadm thrsh-job delete** *jobName*

**mfwkadm thrsh-job suspend** *jobName*

**mfwkadm thrsh-job resume** *jobName*

## 描述

mfwkadm 公用程式是一種管理 Monitoring Framework 代理程式 (又稱爲節點代理程式) 的指令行介面。節點代理程式會在 Common Agent Container 中執行。mfwkadm 公用程式可用來停止和重新啓動節點代理程式，並管理節點代理程式執行的監視工作。應該從執行節點代理程式的相同主機上執行這個指令。請務必遵照此處所述關於此指令的引數順序。

若要變更輸出訊息的語言，請將 LC_MESSAGE 環境變數設定爲您的**語言環境**。mfwkadm 指令將使用 lib/resources 目錄中 JesmfMessages_ *locale*.pm 檔案所包含的訊息。如果語言環境沒有對應的訊息檔案，或者未指定任何語言環境，mfwkadm 指令將使用 JesmfMessages.pm 檔案中的預設訊息組。

mfwkadm 公用程式具有下列子指令。以星號 (*) 標記者需要執行 Common Agent Container，並載入節點代理程式。

- start
- stop
- restart
- list-params (*)
- list-modules (*)
- info (*)
- pm-job (*)
- opstat-job (*)
- thrsh-job (*)

根據載入的 Common Agent Container 模組數量，在啓動節點代理程式和 mfwkadm 公用程式可供使用之間，會有幾秒鐘到幾分鐘的延遲。在這段時間中，指令會失敗，並出現明確的訊息。

# 選項

支援下列選項。

--help　　顯示用法摘要。

# 子指令

start
　啓動 Monitoring Framework 節點代理程式和關聯的元件產品模組，而不停止 Common Agent Container。

　此動作會先部署節點代理程式，然後在 Common Agent Container 中部署關聯的元件產品模組。此設備是一種包裝程式，優先於 cacaoadm 公用程式的 lock 和 undeploy 子指令。

　start 子指令只會啓動節點代理程式，以及與 Monitoring Framework 相關聯的 Java ES 元件模組。元件模組的前綴是 com.sun.cmm。

　**安全性：**start 子指令只能由已啓動 Common Agent Container 的使用者執行。否則會出現類似下列的錯誤訊息：

```
Error occured in mfwkadm
Problem running /usr/sbin/cacaoadm unlock com.sun.mfwk 2>&1.
Stdout/Stderr: This command must be run by user: [root].
```

`stop`
停止 Common Agent Container 中的 Monitoring Framework 節點代理程式及其關聯的 Java ES 元件模組。

此動作會先停止在 Common Agent Container 中部署的所有 Java ES 元件模組，然後停止節點代理程式。此設備是一種包裝程式，優先於 `cacaoadm` 公用程式的 `lock` 和 `unlock` 子指令。

`stop` 子指令只會先停止與 Monitoring Framework 相關聯的 Java ES 元件模組，再停止節點代理程式自身。元件模組的前綴是 `com.sun.cmm`。

**安全性：**`stop` 子指令只能由已啓動 Common Agent Container 的使用者執行。否則會出現類似下列的錯誤訊息：

```
在 mfwkadm 中發生錯誤
執行 /usr/sbin/cacaoadm unlock com.sun.mfwk 2>&1 時發生問題。
Stdout/Stderr：這個指令必須由使用者 [root] 執行。
```

`restart`
重新啓動 Common Agent Container 中的 Monitoring Framework 節點代理程式及其相關聯的 Java ES 元件模組。

此動作會嘗試停止然後重新啓動 Common Agent Container 中的節點代理程式及其相關聯的模組，方式與 `stop` 和 `start` 子指令相同。

**安全性：**`restart` 子指令只能由已啓動 Common Agent Container 的使用者執行。否則會出現類似下列的錯誤訊息：

```
Error occured in mfwkadm
Problem running //usr/sbin/cacaoadm unlock com.sun.mfwk 2>&1.
Stdout/Stderr: This command must be run by user: [root].
```

`list-params`
列出與 Monitoring Framework 節點代理程式相關聯的所有配置參數。

**安全性：**此指令沒有任何使用者限制。

`list-modules`
顯示實作 Common Monitoring Model (CMM) 且載入至 Common Agent Container 的元件產品模組清單。此子指令也會列出各個已安裝 Java ES 元件的所有執行中實例。各個元件可以有 0、1 或多個執行中實例。

**安全性：**對於不是啓動 Common Agent Container 的使用者，已安裝 Java ES 元件的清單不會包含元件實例。

`info` *runningInstance*
顯示關於已命名 *runningInstance* 的資訊。*runningInstance* 必須符合 `list-modules` 子指令輸出所列的執行中實例。 顯示的資訊包括：

- 對於各種類型的監視工作，會按照類別名稱順序顯示所有與執行中實例相關聯的可觀察物件。可觀察物件是可分別使用 `pm-job`、`opstat-job` 或 `thrsh-job` 子指令建立效能監視工作、作業狀態工作或臨界值監視工作的物件。
- 對於各個類別的可觀察物件，會顯示所有可觀察屬性，包括各個屬性的名稱和類型。

**安全性**：對於不是啓動 Common Agent Container 的使用者，則不會顯示任何資訊。

## 效能監視

`pm-job observable-classes`
顯示所有目前可觀察的物件類別清單，以便建立效能監視工作。

`pm-job observable-objects` [class= *objectC lass*] [domain=*objectDomain*]
顯示所有目前可觀察的物件清單，以便建立效能監視工作。預設會列出所有可觀察類別的所有物件和各個網域中的所有物件。物件清單會按照類別名稱排序。

class=*objectClass*
指定選擇性 *objectClass* 會將輸出限制於該指定類別的可觀察物件。*objectClass* 必須是 `pm-job observable-classes` 子指令所列出的其中一個類別。

domain=*objectDomain*
指定選擇性 *objectDomain* 會將輸出限制於該網域的可觀察物件。物件的網域是物件名稱中冒號 (「:」) 字元之前的字串。

`pm-job observable-attributes class=`*objectClass*
顯示指定 *objectClass* 中所有可觀察屬性的清單。屬性會與名稱和類型一併顯示。*objectClass* 必須是可支援效能監視工作類別的其中一個，如 `pm-job observable-classes` 子指令所列。

`pm-job list`
顯示目前所有已定義效能監視工作的清單。會針對各個具有已定義效能工作的物件列出工作，並且物件會按照類別名稱排序。針對各個工作顯示的資訊與 `pm-job info` 子指令顯示的資訊相同。

**安全性**：對於不是啓動 Common Agent Container 的使用者，則不會顯示任何工作。

`pm-job info` *jobName*
顯示名稱爲 *jobName* 的效能監視工作詳細資訊。*jobName* 必須是 `pm-job list` 子指令顯示的項目。此子指令會顯示下列資訊：

- 效能監視工作的名稱。
- 「按照物件」或「按照類別」的效能監視工作類型。按照物件進行的工作會監視一個或多個已命名物件實例，而按照類別進行的工作會監視物件類別的各個實例。請注意，無法使用 `mfwkadm` 公用程式建立按照類別進行的工作。

- 效能監視工作的狀態：使用中已啓動、使用中未啓動或已暫停。使用中已啓動的工作目前已排程執行，並且正在收集資料。使用中未啓動的工作正在執行，但是由於目前時間不在工作排程之內，因此未收集資料。已暫停的工作未執行，並且未收集任何資料。使用 `pm-job suspend` 和 `pm-job resume` 子指令可變更效能監視工作的執行狀態。
- 效能監視工作的顆粒性 (以秒爲單位)。這是此工作的資料收集時間間隔。
- 監視工作的報告週期。報告週期與顆粒性相乘等於通知頻率。例如，如果顆粒性週期是 10 秒，而報告週期爲 6，按照事件進行的工作報告會每 10 秒收集一次資料，並且每 60 秒 (10*6) 傳送一次包含 6 份報告的通知。如果工作也按照檔案進行報告，則會每 60 秒傳送一次包含 6 個已產生檔案所在位置的事件。
- 效能監視工作是否在按照事件進行報告。這表示，效能監視工作的結果會傳送至已註冊的用戶端做爲通知。
- 效能監視工作是否在按照檔案進行報告。這表示，效能監視工作的報告會寫入至本機檔案，而包含檔案名稱的通知會傳送至已註冊的用戶端。
- 效能監視工作的報告格式皆爲 `XML`。
- 效能監視工作的排程。排程會指定工作在什麼日期和時間，處於使用中已啓動或使用中未啓動狀態 (分別表示是否收集資料)。 對於按照物件進行的工作：
- 所觀察物件的清單，按照名稱排序。
- 如果只指定部分可觀察的屬性，則受觀察物件的受觀察屬性會按照名稱和類型列出。 對於按照類別進行的工作：
- 所觀察類別的清單，按照名稱排序。
- 如果只指定部分可觀察的屬性，則受觀察類別的受觀察屬性會按照名稱和類型列出。這些屬性是所有類別共用的屬性。

**安全性**：對於不是啓動 Common Agent Container 的使用者，不會顯示任何資訊。

`pm-job create` *jobName* granularity= *integerValue* object=*objectName* [object= *objectName* ...]
在一或多個物件上建立新的效能監視工作。`mfwkadm` 指令無法建立按照類別進行的工作。建立效能監視工作時，可設定下列參數：

*jobName*
唯一識別效能監視工作的字串。任何其他效能監視工作都不可使用 *jobName*。

granularity=*integerValue*
當工作處於使用中已啓動狀態時，連續兩次收集測量資料啓動之間的指定時間 (以秒爲單位)。例如，顆粒性週期可以是 300 秒 (5 分鐘)、900 秒 (15 分鐘)、1800 秒 (每半小時)、3600 秒 (每小時)。在大多數狀況下，300 秒的顆粒性週期已足夠。對於某些測量而言，以較大顆粒性週期收集資料可能更有意義。

object=*objectName* [object= *objectName* ...]
指一個或多個可觀察物件，讓效能監視工作向此物件收集資料並提出報告。*objectName* 必須是 `pm-job list` 或 `pm-job observable-objects` 子指令顯示的項目。指定多個 object= *objectName* 參數會建立一個監視多個物件的效能監視工作。

**安全性：**此子指令只能由啓動 Common Agent Container 的使用者執行。

`pm-job delete` *jobName*
刪除名稱爲 *jobName* 的效能監視工作。*jobName* 必須是 `pm-job list` 子指令顯示的項目。

**安全性：**此子指令只能由啓動 Common Agent Container 的使用者執行。

`pm-job suspend` *jobName*
暫停名稱爲 *jobName* 的效能監視工作。已暫停的工作不會處於使用中狀態，並且不論排程爲何，都不再收集資料。然而，工作將維持已定義狀態，並且能夠以 `pm-job resume` 子指令再次設定爲使用中。*jobName* 必須是由 `pm-job list` 子指令顯示的項目。

**安全性：**此子指令只能由啓動 Common Agent Container 的使用者執行。

`pm-job resume` *jobName*
繼續名稱爲 *jobName* 的效能監視工作。已繼續的工作會根據排程開始收集資料，並傳送報告。*jobName* 必須是其中一個由 `pm-job list` 子指令顯示的項目。這與 `pm-job suspend` 子指令相對應。

**安全性：**此子指令只能由啓動 Common Agent Container 的使用者執行。

## 作業狀態監視

`opstat-job observable-classes`
顯示所有目前可觀察的物件類別清單，以便建立作業狀態監視工作。

`opstat-job observable-objects` [class= *objectClass*] [domain=*objectDomain*]
顯示所有目前可觀察的物件清單，以便建立作業狀態監視工作。預設會列出所有可觀察類別的所有物件和各個網域中的所有物件。物件清單會按照類別名稱排序。

class=*objectClass*
指定選擇性 *objectClass* 會將輸出限制於該指定類別的可觀察物件。*objectClass* 必須是 `opstat-job observable-classes` 子指令所列出的其中一個類別。

domain=*objectDomain*
指定選擇性 *objectDomain* 會將輸出限制於該網域的可觀察物件。物件的網域是物件名稱中冒號 (「:」) 字元之前的字串。

`opstat-job observable-attributes class=` *objectClass*
顯示指定 *objectClass* 中所有可觀察屬性的清單。屬性會與名稱和類型一併顯示。*objectClass* 必須是 `opstat-job observable-classes` 子指令所列出的其中一個類別。

`opstat-job list`
顯示目前所有已定義作業狀態監視工作的清單。會針對各個具有已定義作業狀態工作的物件列出工作，並且物件會按照類別名稱排序。針對各個工作顯示的資訊與 `opstat-job info` 子指令顯示的資訊相同。

**安全性：**對於不是啓動 Common Agent Container 的使用者，不會顯示任何工作。

`opstat-job info` *jobName*
顯示名稱爲 *jobName* 的作業狀態監視工作詳細資訊。*jobName* 必須是 `opstat-job list` 子指令顯示的項目。此子指令會顯示下列資訊：

- 作業狀態監視工作的名稱。
- 「按照物件」或「按照類別」的作業狀態監視工作類型。按照物件進行的工作會監視已命名物件實例，而按照類別進行的工作會監視物件類別的各個實例。請注意，無法使用 `mfwkadm` 公用程式建立按照類別進行的工作。
- 作業狀態監視工作的狀態。使用中已啓動、使用中未啓動或已暫停。使用中已啓動的工作目前已排程執行，並且正在收集資料。使用中未啓動的工作正在執行，但是由於目前時間不在工作排程之內，因此未收集資料。已暫停的工作未執行，並且未收集任何資料。使用 `opstat-job suspend` 和 `opstat-job resume` 子指令可變更作業狀態監視工作的執行狀態。
- 作業狀態監視工作的顆粒性 (以秒爲單位)。這是此工作的資料收集時間間隔。
- 作業狀態監視工作是否在按照事件進行報告。這表示，作業狀態監視工作的結果會傳送至已註冊的用戶端做爲通知。
- 作業狀態監視工作是否在按照檔案進行報告。這表示，作業狀態監視工作的報告會寫入至本機檔案，而包含檔案名稱的通知會傳送至已註冊的用戶端。
- 作業狀態監視工作的報告格式皆爲 `XML`。
- 作業狀態監視工作的排程。排程會指定工作在什麼日期和時間，處於使用中已啓動或使用中未啓動狀態 (分別表示是否收集資料)。
- 對於按照物件進行的工作，則是按照名稱排序的受觀察物件清單。
- 對於按照類別進行的工作，則是按照名稱排序的受觀察物件清單。

**安全性**：對於不是啓動 Common Agent Container 的使用者，不會顯示任何資訊。

`opstat-job create` *jobName* granularity= *integerValue* object=*objectName* [object= *objectName* ...]
在一個或多個物件上建立新的作業狀態監視工作。`mfwkadm` 指令無法建立按照類別的工作。建立效能監視工作時，會設定下列參數：

*jobName*
唯一識別作業狀態監視工作的字串。任何其他作業狀態監視工作都不可使用 *jobName*。

granularity=*integerValue*
當工作處於使用中已啓動狀態時，兩次連續收集測量資料啓動之間的指定時間 (以秒爲單位)。

object=*objectName* [object= *objectName* ...]
指一個或多個可觀察物件，讓作業狀態監視工作向此物件收集資料並提出報告。*objectName* 必須是 `opstat-job list` 或 `opstat-job observable-objects` 子指令顯示的項目。指定多個 object= *objectName* 參數會建立一個監視多個物件的作業狀態工作。

**安全性：**此子指令只能由啓動 Common Agent Container 的使用者執行。

`opstat-job delete` *jobName*
刪除名稱爲 *jobName* 的作業狀態監視工作。*jobName* 必須是 `opstat-job list` 子指令顯示的項目。

**安全性：**此子指令只能由啓動 Common Agent Container 的使用者執行。

`opstat-job suspend` *jobName*
暫停名稱爲 *jobName* 的作業狀態監視工作。已暫停的工作不會處於使用中狀態，並且不論排程爲何，都不再收集資料。然而，工作將維持已定義狀態，並且能夠以 `opstat-job resume` 子指令再次設定爲使用中。*jobName* 必須是 `opstat-job list` 子指令顯示的項目。

**安全性：**此子指令只能由啓動 Common Agent Container 的使用者執行。

`opstat-job resume` *jobName*
繼續名稱爲 *jobName* 的作業狀態監視工作。已繼續的工作會根據排程收集資料，並傳送報告。*jobName* 必須是 `opstat-job list` 子指令顯示的項目。這與 `opstat-job suspend` 子指令相對應。

**安全性：**此子指令只能由啓動 Common Agent Container 的使用者執行。

## 臨界值監視

`thrsh-job observable-classes`
顯示所有目前可觀察的物件類別清單，以便建立臨界值監視工作。

`thrsh-job observable-objects` [class= *objectClass*] [domain=*objectDomain*]
顯示所有目前可觀察的物件清單，以便建立臨界值監視工作。預設會列出所有可觀察類別的所有物件和各個網域中的所有物件。物件清單會按照類別名稱排序。

class=*objectClass*
指定選擇性 *objectClass* 會將輸出限制於該指定類別的可觀察物件。*objectClass* 必須是 `thrsh-job observable-classes` 子指令所列出的其中一個類別。

domain=*objectDomain*
指定選擇性 *objectDomain* 會將輸出限制於該網域的可觀察物件。物件的網域是物件名稱中冒號 (「:」) 字元之前的字串。

`thrsh-job observable-attributes class=`*objectClass*
顯示指定 *objectClass* 中所有可觀察屬性的清單。屬性會與名稱和類型一併顯示。*objectClass* 必須是 `thrsh-job observable-classes` 子指令所列出的其中一個類別。

`thrsh-job list`
顯示目前所有已定義臨界值監視工作的清單。會針對各個具有已定義臨界值工作的物件列出工作，並且物件會按照類別名稱排序。針對各個工作顯示的資訊與 `thrsh-job info` 子指令顯示的資訊相同。

**安全性：**對於不是啓動 Common Agent Container 的使用者，不會顯示任何工作。

`thrsh-job info` *jobName*

顯示名稱爲 *jobName* 的臨界值監視工作詳細資訊。*jobName* 必須是 `thrsh-job list` 子指令顯示的項目。此子指令會顯示下列資訊：

- 臨界值監視工作的名稱。
- 臨界值監視工作的多重性。在此發行版本中，只可使用在一個物件上監視一個屬性的簡單臨界值工作。
- 臨界值監視工作的狀態：使用中已啓動、使用中未啓動或已暫停。使用中已啓動的工作目前已排程執行，並且正在收集資料。使用中未啓動的工作正在執行，但是由於目前時間不在工作排程之內，因此未收集資料。已暫停的工作未執行，並且未收集任何資料。使用 `thrsh-job suspend` 和 `thrsh-job resume` 子指令可變更臨界值監視工作的執行狀態。
- 臨界值監視工作的顆粒性 (以秒爲單位)。這是此工作的資料收集時間間隔。
- 臨界值監視工作的排程。排程會指定工作在什麼日期和時間，處於使用中已啓動或使用中未啓動狀態 (分別表示是否收集資料)。
- 臨界值監視工作的警示配置。當受監視屬性的受觀察值超過定義的臨界值時，會觸發此警示。其中會顯示警視的類型和嚴重性。
- 臨界值監視工作的受觀察物件。
- 套用臨界值的屬性名稱。
- 會觸發警示的臨界值。
- 會在臨界值觸發警示的值進度方向，`RISING` 或 `FALLING`。
- 臨界值的偏差偏移。方向是 `RISING` 時，在受觀察屬性小於 *thresholdValue-offsetValue* 之前，都不會再次觸發警示。方向是 `FALLING` 時，在受觀察屬性大於 *thresholdValue+offsetValue* 之前，都不會再次觸發警示。即使偏移爲零，運作方式仍會是如此。

**安全性**：對於不是啓動 Common Agent Container 的使用者，不會顯示任何資訊。

`thrsh-job create` *jobName* object= *objectName* granularity=*integerValue* attributeName= *attributeName* attributeType=*attributeType* thresholdValue= *thresholdValue* thresholdOffset=*offsetValue* thresholdDirection= `[RISING|FALLING]`

建立新的臨界值監視工作，監視單一物件的一個屬性。建立臨界值工作時，可設定下列參數：

*jobName*

唯一識別臨界值監視工作的字串。任何其他臨界值監視工作都不可使用 *jobName*。

object=*objectName*

臨界值監視工作會收集屬性值以比對臨界值的可觀察物件。*objectName* 必須是 `thrsh-job list` 或 `thrsh-job observable-objects` 子指令顯示的項目。

granularity=*integerValue*
當工作處於使用中已啓動狀態時，連續兩次觀察屬性值啓動之間的指定時間 (以秒爲單位)。

attributeName=*attributeName*
指屬性名稱，讓臨界值監視工作向此屬性收集值並且與臨界值比對。*attributeName* 必須由 `thrsh-job info` 或 `thrsh-job observable-attributes` 子指令列出。

attributeType=*attributeType*
所需監視的可觀察屬性類型。*attributeType* 必須由 `thrsh-job info` 或 `thrsh-job observable-attributes` 子指令列出。

thresholdValue=*thresholdValue*
以 `thresholdDirection` 中指定的方向越過時，會讓此臨界值工作觸發警示的受監視屬性值。

thresholdOffset=*offsetValue*
*offsetValue* 會判斷觸發連續警示的臨界值工作偏差。*offsetValue* 必須爲零或正值。在觸發警示事件之後，將不會觸發新的警示事件，直到受監視屬性的值超過 *offsetValue* 和 `thresholdDirection` 定義的範圍。

thresholdDirection=`[RISING|FALLING]`
方向爲 `RISING` 時，在受觀察屬性值小於 *thresholdValue-offsetValue* 之前，都不會再次觸發警示事件。方向爲 `FALLING` 時，在受觀察屬性值大於 *thresholdValue+offsetValue* 之前，都不會再次觸發警示事件。即使 *offsetValue* 爲零，運作方式仍會是如此。

**安全性：**此子指令只能由啓動 Common Agent Container 的使用者執行。

`thrsh-job delete` *jobName*
刪除名稱爲 *jobName* 的臨界值監視工作。*jobName* 必須是 `thrsh-job list` 子指令顯示的項目。

**安全性：**此子指令只能由啓動 Common Agent Container 的使用者執行。

`thrsh-job suspend` *jobName*
暫停名稱爲 *jobName* 的臨界值監視工作。暫停的工作不會處於使用中狀態，並且不論排程爲何，都不再收集資料。然而，工作將維持已定義狀態，並且能夠使用 `thrsh-job resume` 子指令再次設定爲使用中。*jobName* 必須是 `thrsh-job list` 子指令顯示的項目。

**安全性：**此子指令只能由啓動 Common Agent Container 的使用者執行。

`thrsh-job resume` *jobName*
繼續名稱爲 *jobName* 的臨界值監視工作。已繼續的工作會根據排程收集資料，並傳送報告。*jobName* 必須是 `thrsh-job list` 子指令顯示的項目。這與 `thrsh-job suspend` 子指令相對應。

**安全性：**此子指令只能由啓動 Common Agent Container 的使用者執行。

# 範例

下列虛構案例說明如何使用 mfwkadm 公用程式，以及其中的選項和子指令。

list-modules 子指令顯示目前主機上的 Java ES 元件實例，以及 Common Agent Container 中的對應模組。下列範例列出兩個已安裝元件，分別是沒有任何執行中實例的 Directory Server，以及有一個執行中實例的 Web 伺服器。

```
$ mfwkadm list-modules

Installed products and their running instances:
===============================================

Instances for installed product: com.sun.cmm.ds:collectionID=/opt/SUNWdsee/ds6,
name=Sun Java(TM) System Directory Server,type=CMM_InstalledProduct
------------------------------

No instance.

Instances for installed product: com.sun.cmm.ws:collectionID=/var/opt/SUNWwbsvr7,
name=WebServer,type=CMM_InstalledProduct
------------------------------

/wsPrefix/com.sun.cmm.ws:name=https-hostname.example.com,type=CMM_ApplicationSystem
```

下列 info 子指令顯示 Web 伺服器實例中的可觀察物件，以及各個工作類型的類別和屬性。

```
$ mfwkadm info /wsPrefix/com.sun.cmm.ws:name=https-hostname.example.com,\\
type=CMM_ApplicationSystem

Information about running instance: /wsPrefix/com.sun.cmm.ws:
name=https-hostname.example.com,type=CMM_ApplicationSystem
==================================

Observable objects for performance jobs:
----------------------------------------

+ Objects of class: com.sun.cmm.settings.CMM_ApplicationSystemSetting

        /wsPrefix/com.sun.cmm.ws:name=https-hostname.example.com-setting,
type=CMM_ApplicationSystemSetting

        Observable attributes:

        Caption [STRING]
        ConfigurationDirectory [STRING]
        CreationClassName [STRING]
```

```
        Description [STRING]
        DirectoryName [STRING]
        ElementName [STRING]
        InstanceID [STRING]
        Name [STRING]
        URL [STRING]

+ Objects of class: com.sun.cmm.settings.CMM_KeepAliveSetting

        /wsPrefix/com.sun.cmm.ws:name=process-1-keepalive-setting,
type=CMM_KeepAliveSetting

        Observable attributes:

        AllocationUnit [STRING]
        Caption [STRING]
        ConnectionsUpperBound [LONG]
        CreationClassName [STRING]
        Description [STRING]
        ElementName [STRING]
        InputUnit [STRING]
        InstanceID [STRING]
        LowerAllocationLimit [LONG]
        LowerInputLimit [LONG]
        LowerOutputLimit [LONG]
        Name [STRING]
        OtherAllocationUnit [STRING]
        OtherInputUnit [STRING]
        OtherLowerAllocationLimit [LONG]
        OtherLowerInputLimit [LONG]
        OtherLowerOutputLimit [LONG]
        OtherOutputUnit [STRING]
        OtherUpperAllocationLimit [LONG]
        OtherUpperInputLimit [LONG]
        OtherUpperOutputLimit [LONG]
        OutputUnit [STRING]
        QueuedUpperBound [LONG]
        SecondsTimeout [LONG]
        TimeoutUpperBound [LONG]
        UpperAllocationLimit [LONG]
        UpperInputLimit [LONG]
        UpperOutputLimit [LONG]
        ...
```

下列指令顯示已定義效能監視工作的清單。在此範例中，有一個名稱爲 `myPerfJob` 的用於監視一個物件的效能工作：

```
$ mfwkadm pm-job list
```

```
BY_OBJECTS performance jobs:
============================

Performance job information for: myPerfJob
------------------------------

Type:                BY_OBJECTS
State:               ACTIVE_ON_DUTY
Granularity period:  30
Reporting period:    1
By event:            EVENT_SINGLE
By file:             EVENT_SINGLE
Report format:       XML
Schedule:
        Global start time: Immediately
        Global stop time: Forever
        Weekly schedule: Everyday
        Daily schedule: All day
Observed objects:
                /wsPrefix/com.sun.cmm.ws:name=virtualServer-hostname.example.com-
webApp-/-stats,type=CMM_VirtualServerWebModuleStats
Observed attributes:
                All available

BY_CLASSES performance jobs:
============================

No jobs found.
```

下列指令會建立作業狀態監視工作，與兩個從 `opstat-job info` 或 `opstat-job observable-objects` 子指令取得的可觀察物件相關聯：

```
$ mfwkadm opstat-job create myOpStatJob granularity=60 \\
object=/wsPrefix/com.sun.cmm.ws:name=process-1,type=CMM_UnixProcess \\
object=/wsPrefix/com.sun.cmm.ws:name=process-1-DNSCache1,type=CMM_DnsCache
```

下列指令會暫停上面建立的工作：

```
$ mfwkadm opstat-job suspend myOpStatJob
```

下列指令會顯示潛在臨界值監視工作的可觀察類別：

```
$ mfwkadm thrsh-job observable-classes

Threshold jobs observable classes:
==================================

com.sun.cmm.cim.CIM_ScopedSettingData
```

```
com.sun.cmm.cim.CIM_SettingData
com.sun.cmm.cim.CIM_StatisticalData
com.sun.cmm.cim.statistics.CIM_EthernetPortStatistics
com.sun.cmm.cim.statistics.CIM_NetworkPortStatistics
com.sun.cmm.cim.statistics.j2ee.CIM_J2eeJVMStats
com.sun.cmm.cim.statistics.j2ee.CIM_J2eeStatistic
com.sun.cmm.settings.CMM_ApplicationSystemSetting
com.sun.cmm.settings.CMM_KeepAliveSetting
com.sun.cmm.settings.CMM_QueueTimeoutSetting
com.sun.cmm.settings.CMM_RFC2788ApplicationSystemSetting
com.sun.cmm.settings.CMM_ScopedSettingData
com.sun.cmm.settings.CMM_SoftwareResourceSetting
com.sun.cmm.settings.CMM_SWRBufferSetting
com.sun.cmm.settings.CMM_SWRLimitSetting
com.sun.cmm.settings.CMM_SWRQueueSetting
com.sun.cmm.settings.CMM_VirtualServerSetting
com.sun.cmm.statistics.CMM_ApplicationSystemStats
com.sun.cmm.statistics.CMM_ApplicationSystemWatchdogStats
com.sun.cmm.statistics.CMM_ConnectionQueueStats
com.sun.cmm.statistics.CMM_DnsCacheStats
com.sun.cmm.statistics.CMM_EthernetPortStats
com.sun.cmm.statistics.CMM_FileCacheStats
com.sun.cmm.statistics.CMM_HTTPResponsesStats
com.sun.cmm.statistics.CMM_JVMJSR174ExtStats
com.sun.cmm.statistics.CMM_JVMJSR174Stats
com.sun.cmm.statistics.CMM_JVMStats
com.sun.cmm.statistics.CMM_NetworkPortStats
com.sun.cmm.statistics.CMM_OperatingSystemStats
com.sun.cmm.statistics.CMM_ProcessorStats
com.sun.cmm.statistics.CMM_ProcessStats
com.sun.cmm.statistics.CMM_QueueTimeoutStats
com.sun.cmm.statistics.CMM_RFC2788ApplicationTableStats
com.sun.cmm.statistics.CMM_ServiceStats
com.sun.cmm.statistics.CMM_SoftwareResourceStats
com.sun.cmm.statistics.CMM_SolarisEthernetPortStats
com.sun.cmm.statistics.CMM_SolarisNetworkPortStats
com.sun.cmm.statistics.CMM_SolarisOperatingSystemStats
com.sun.cmm.statistics.CMM_SolarisProcessorStats
com.sun.cmm.statistics.CMM_SolarisProcessorSysinfoStats
com.sun.cmm.statistics.CMM_SolarisProcessorVmStats
com.sun.cmm.statistics.CMM_Statistic
com.sun.cmm.statistics.CMM_SWRBufferStats
com.sun.cmm.statistics.CMM_SWRCacheStats
com.sun.cmm.statistics.CMM_SWRLimitStats
com.sun.cmm.statistics.CMM_SWRQueueStats
com.sun.cmm.statistics.CMM_UnixOperatingSystemStats
com.sun.cmm.statistics.CMM_UnixProcessStats
com.sun.cmm.statistics.CMM_VirtualServerWebModuleStats
com.sun.cmm.statistics.CMM_WebModuleStats
```

下列指令會顯示監視上一範例中所找到 com.sun.cmm.statistics.CMM_SWRQueueStats 類別物件的臨界值工作可觀察屬性：

```
$ mfwkadm thrsh-job observable-attributes \\
class=com.sun.cmm.statistics.CMM_SWRQueueStats

Threshold jobs observable attributes:
=====================================

Class: com.sun.cmm.statistics.CMM_SWRQueueStats

Attributes:

BufferSize [LONG]
EntriesCount [LONG]
EntriesHighWaterMark [LONG]
EntriesLowWaterMark [LONG]
EntriesTotal [LONG]
ErrorCount [INTEGER]
FailedOperations [LONG]
LowerLimit [LONG]
OperationsCount [LONG]
OtherLowerLimit [LONG]
OtherUpperLimit [LONG]
OverflowsCount [LONG]
QueuedCount [LONG]
QueuedHighWater [LONG]
SampleInterval [LONG]
TotalQueuedCount [LONG]
UpperLimit [LONG]
```

下列指令是另一個工作建立範例，其中包含臨界值工作：

```
$ mfwkadm thrsh-job create myThreshJob granularity=30 \\
object=/wsPrefix/com.sun.cmm.ws:name=process-1-threadPool-NativePool-stats,\\
type=CMM_SWRQueueStats attributeName=EntriesCount attributeType=LONG \\
thresholdValue=1000 thresholdOffset=10 thresholdDirection=RISING
```

下列範例會說明上一範例中所建立臨界值監視工作之 thrsh-job info 子指令的輸出：

```
$ mfwkadm thrsh-job info myThreshJob

Threshold job information for: myThreshJob
------------------------------

Type:                SIMPLE
State:               ACTIVE_ON_DUTY
Granularity period:  30
```

```
Schedule:
        Global start time: Immediately
        Global stop time: Forever
        Weekly schedule: Everyday
        Daily schedule: All day
Alarm configuration:
               Type: QualityOfServiceAlarm
               Severity: INDETERMINATE
Threshold definition(s):
               Object: /wsPrefix/com.sun.cmm.ws:name=process-1-threadPool-
NativePool-stats,type=CMM_SWRQueueStats
                        Attribute: EntriesCount [LONG]
                        Value: 1000
                        Direction: RISING
                        Offset: 10
```

# 結束狀態

會傳回下列結束值：

0　成功完成

1　發生錯誤

# 屬性

| 屬性類型 | 屬性值 |
| --- | --- |
| 可用性 | SUNWmfwk |
| 介面穩定性 | Contract Private |

# 另請參閱

cacao.5、cacaoadm.1m

◆ ◆ ◆ 第 3 章

# 3 安裝和使用 Monitoring Console

Monitoring Console 是網路型應用程式，可顯示設備收集的所有監視資料，其依賴主代理程式集合來自各節點代理程式的所有值和警示通知。

安裝 Monitoring Console 後，您可以從任何主機上的簡單瀏覽器視窗安全地進行存取，即使在網際網路上，只要配置防火牆爲允許，就可以進行存取。使用圖形介面，您便可以即時看見監視值、檢視和確認警示，並且建立觸發自訂警示的規則。

**備註 –** 在執行任何安裝或配置之前，您應該先參閱「適用於 UNIX 的 Sun Java Enterprise System 5 版本說明」。

本章包含以下各節：


- 第 45 頁的 「安裝 Monitoring Console」
- 第 48 頁的 「已安裝目錄的配置」
- 第 48 頁的 「啓動 Monitoring Console」
- 第 51 頁的 「使用 Monitoring Console」
- 第 59 頁的 「Monitoring Console 疑難排解」


## 安裝 Monitoring Console

由於此發行版本主代理程式的限制，您無法在節點代理程式所在的相同主機上安裝主代理程式。因此，Monitoring Console 無法安裝在其他受監視的 Java ES 元件所在主機上。除非您已經配置 Solaris Zones，否則將其必須安裝在專屬主機上。如需更多資訊，請參閱下面第 46 頁的 「在 Solaris Zone 中安裝 Monitoring Console」

由於共用元件相依性，安裝 Monitoring Console 也會同時安裝 Monitoring Framework。主控台需要架構和 Common Agent Container 才能載入主代理程式，但與節點代理程式不同的是，主代理程式無法由使用者配置。尤其在安裝 Monitoring Console 的主機或區域中，您不應使用 `mfwkadm` 指令。

## ▼ 使用 Java ES 安裝程式安裝 Monitoring Console

由於此後期測試版的限制，您必須將 Monitoring Console 安裝在未安裝任何 Java ES 元件的主機或 Solaris Zone 中。因此在本程序中，您只會安裝 Monitoring Console 元件。

本程序使用安裝程式的圖形介面。如需有關如何以其他模式執行安裝程式的資訊，請參閱「適用於 UNIX 的 Sun Java Enterprise System 5 安裝指南」中的第 4 章「使用基於文字的介面安裝」和「適用於 UNIX 的 Sun Java Enterprise System 5 安裝指南」中的第 5 章「以無訊息模式安裝」。

1. **從 Java ES 發行版本所在平台對應的目錄中，啓動 `installer` 應用程式。如需更多資訊，請參閱「適用於 UNIX 的 Sun Java Enterprise System 5 安裝指南」中的「開始安裝」。**
2. **在歡迎畫面之後繼續，接受授權合約、選擇 [升級] 或 [安裝]、選取 [安裝新軟體]，然後按 [下一步]。**
3. **在 [選取元件] 畫面中，選取僅安裝 Sun Java System Monitoring Console。按 [下一步]。**
4. **安裝程式會檢查共用元件所需的升級項目。完成之後，按 [下一步]。**
5. **安裝程式此時會檢查系統需求。如果作業系統需要修補程式，請取消安裝，然後在系統中增加必要的修補程式，並重新啓動此程序。如果不需要修補程式，請按 [下一步]。**
6. **在 [選取配置類型] 畫面中，選擇 [立即配置]，並且在接下來的 [自訂配置] 畫面中，按 [下一步]。**
7. **安裝程式已經完成安裝 Monitoring Console 的準備工作，按 [下一步] 便可開始進行安裝。如果尚未註冊 Java ES 的部署，您可在安裝期間開啓產品註冊視窗。**
8. **安裝完成之後，可查閱安裝摘要和記錄，然後按一下 [安裝完成] 結束安裝程式。**

**接下來的步驟** 您現在應繼續進行第 47 頁的 「配置 Monitoring Console」。

## ▼ 在 Solaris Zone 中安裝 Monitoring Console

您可以使用 Solaris Zones，在 Java ES 其他元件所在的相同實體主機上安裝 Monitoring Console。這些元件將出現在全域區域中，而您將建立稀疏根本機區域做爲 Monitoring Console 的邏輯主機。請依下列順序進行。

1. **安裝和配置所有的 Java ES 元件，但不包含全域區域中的 Monitoring Console。在全域區域中完成所有選取元件的安裝後配置，以便所有的伺服器實例都在執行。**
2. **在全域區域的安裝程序中，Monitoring Framework 會安裝成為全域區域中的共用元件。執行第 2 章中適用於已安裝元件的所有程序。**

3. **在相同主機上，建立稀疏根本機區域做為 Monitoring Console 的邏輯主機。由於是稀疏根區域，因此可看見安裝在 *mfwk-base* 中的 Monitoring Framework (請參閱第 9 頁的 「預設路徑和檔案名稱」)。**

4. **按照第 46 頁的 「使用 Java ES 安裝程式安裝 Monitoring Console」的程序，在稀疏根本機區域中安裝 Monitoring Console。**

5. **使用下列指令，在稀疏根區域中配置 Monitoring Framework：**

   ```
   cd mfwk-base/bin
   ./mfwksetup -i
   ```

   使用全域區域中的檔案時，此指令會在本機區域建立必要的 Monitoring Framework 配置檔案。

**接下來的步驟** 您現在應繼續進行第 47 頁的 「配置 Monitoring Console」。

## ▼ 配置 Monitoring Console

此程序說明如何在個別的實體主機中配置 Monitoring Console。如果您在 Solaris Zone 建立的邏輯主機上安裝 Monitoring Console，則指令相同，但是必須在該區域的檔案系統中執行。

1. **在 Monitoring Framework 中使用下列指令初始化主代理程式：**

   ```
   cd mfwk-base/bin
   ./masetup -i
   ```

2. **使用下列指令重新啓動 Common Agent Container (`cacao`)：**

   ```
   cacaoadm restart
   ```

## ▼ 取消配置 Monitoring Console

如果您在要安裝其他元件的主機上安裝和配置 Monitoring Console，由於 Monitoring Framework 中會發生衝突，您將無法監視這些元件。若要以節點代理程式監視新元件，您必須取消配置 Monitoring Console 的主代理程式。

- **以超級使用者身份執行下列指令以取消配置 Monitoring Console：**

  ```
  cacaoadm stop
  cacaoadm unregister-module com.sun.mfwk.masteragent.xml
  cacaoadm register-module /etc/mfwk-base/xml/com.sun.mfwk.xml
  cacaoadm restart
  ```

# 已安裝目錄的配置

如需作業系統上所安裝套裝軟體的名稱，請參閱「適用於 UNIX 的 Sun Java Enterprise System 5 安裝參照」中的第 5 章「可安裝的套裝軟體清單」。下表說明 Monitoring Console 套裝軟體中的目錄。預設安裝目錄 *MConsole-base* 具有下列意義，如第 9 頁的「預設路徑和檔案名稱」所述：

- Solaris 系統：`/opt/SUNWjesmc`
- Linux 系統：`/opt/sun/jesmc`

**表 3–1** Monitoring Console 使用的目錄和檔案

| 路徑 | 內容說明 |
|---|---|
| *MConsole-base*`/WEB-INF/classes` | Web 應用程式 Servlet 類別 |
| *MConsole-base*`/WEB-INF/lib` | Web 應用程式 JAR 相依性 |
| *MConsole-base*`/WEB-INF/*.xml` | Web 應用程式描述元 |
| *MConsole-base*`/css` | 樣式表檔案 |
| *MConsole-base*`/html` | HTML 檔案 |
| *MConsole-base*`/images` | 使用者介面中使用的 GIF 影像檔 |
| *MConsole-base*`/js` | JavaSript™ 檔案 |
| *MConsole-base*`/*.jsp` | JavaServer Pages™ 檔案 |
| *WebConsole-base*`/prereg/jesmc/*.reg` | Monitoring Console 的 Web 主控台檔案 |

# 啓動 Monitoring Console

Monitoring Console 是 Web 應用程式，可透過任何可連線至所安裝主機的瀏覽器使用。透過自動安裝在相同主機上的 Web 主控台，即可存取 Monitoring Console。下列程序說明如何存取 Monitoring Console 並檢視受監視的元件。

## ▼ 啓動 Monitoring Console

1. **您必須先重新啓動 Web 主控台的 Web 伺服器。在已安裝 Monitoring Console 的主機或區域中，執行此指令：**

   ```
   /usr/sbin/smcwebserver restart
   ```

2. **等候 Web 主控台啓動。使用下列指令檢查主控台是否已就緒：**

   ```
   /usr/sbin/smcwebserver status
   ```

您可能需要多次執行此指令，直到看見下列訊息爲止：

```
Sun Java(TM) Web Console is running.
```

3 **從任何可連線至 Monitoring Console 主機的瀏覽器，使用下列 URL 開啓 Web 主控台。如果安裝在 Solaris Zone 中，則 *MC-host* 為您指定給該區域的邏輯主機名稱：**

`https://`*MC-host*`.`*domain*`:6789`

4 **根據配置瀏覽器的方式，您會看見關於不信任憑證的訊息。您必須信任此憑證，才能夠存取 Web 主控台。**

5 **在出現提示時，使用 Monitoring Console 主機的 Root 密碼，以 `root` 身份登入至 Web 主控台。**

登入之後，Web 主控台會列出其提供的所有服務。

6 **若要開啓 Monitoring Console 的主視窗，請按一下「其他」標題下的 Sun Java System Monitoring Console，如以下螢幕擷取所示。**


## ▼ 連線至節點代理程式

第一次啓動 Monitoring Console 時，您必須指出受監視元件的所在位置。指出 Java ES 部署中各個節點代理程式的位置之後，主控台會自動顯示各個節點代理程式中的所有元件。如果稍後透過將 Java ES 元件安裝到新主機中來將其增加到部署中，您也需要重複此程序。

增加節點代理程式後，每次您存取主控台時 Monitoring Console 都會重新連線，直到您將其明確移除爲止。如果先前增加的節點代理程式無回應，請按照第 27 頁的 「重新啓動節點代理程式」的程序進行。

1 **針對已安裝 Monitoring Console 的邏輯主機，以及包含欲監視之節點代理程式和 Java ES 元件的主機，同步化兩者的日期和時間。不論是自動或手動同步化，各個主機的時間差距必須在大約 10 分鐘之內。**

2 **必要時，按一下 Monitoring Console 左側階層根位置的 [部署] 連結，瀏覽至部署層級的顯示內容。此時選取右側窗格的 [主機] 標籤，然後按一下 [增加]。**

3 **在顯示的 [增加主機] 對話方塊中輸入必要資訊，如以下螢幕擷取所示：**


- 主機名稱：爲已配置受監視元件的節點代理程式，輸入完全合格的主機名稱。
- 連接埠：11164；除非您已在節點代理程式常駐的主機上另行配置了 Monitoring Framework。

- 遠端主機 Root 密碼：輸入節點代理程式所在系統的 Root 密碼。

4 **按一下 [測試連結]。**

如果連線資訊正確無誤，而且主機代理程式已經過配置並執行，則對話方塊會顯示目前已連線的訊息。

5 **按一下 [確定] 結束 [增加主機] 對話方塊，新增的名稱便會出現在主機清單中。主機節點代理程式中的所有受監視元件此時也會出現在左欄中。**

6 **針對 Java ES 部署中已安裝受監視元件的各個主機，重複此程序。**

**接下來的步驟** 此時您便可以瀏覽左欄所列的元件，以檢視作業狀態、顯示的受監視屬性及觸發的所有警示。

# 使用 Monitoring Console

本節中的程序會說明如何與 Monitoring Console 互動。

## ▼ 選擇性停用和重新啓用監視

Java ES 監視機制採用簡易設計，以避免影響生產系統的效能。然而在某些狀況下最好停止收集監視值，這樣設備對效能的影響才能趨近於零。Monitoring Console 針對不同主機提供停止收集的方法，如以下程序所述。

1 **必要時，按一下 Monitoring Console 左側階層根位置的 [部署] 連結，瀏覽至部署層級的顯示內容。然後按一下右側窗格中的 [主機] 標籤。**


[主機] 標籤上的表格會列出包含 Monitoring Console 所監視 Java ES 元件的所有主機清單。

2 **使用該表格左欄的核取方塊，選取需要停止監視的所有主機。按一下主機表格上端的 [停用]。**

更多資訊

### 結果

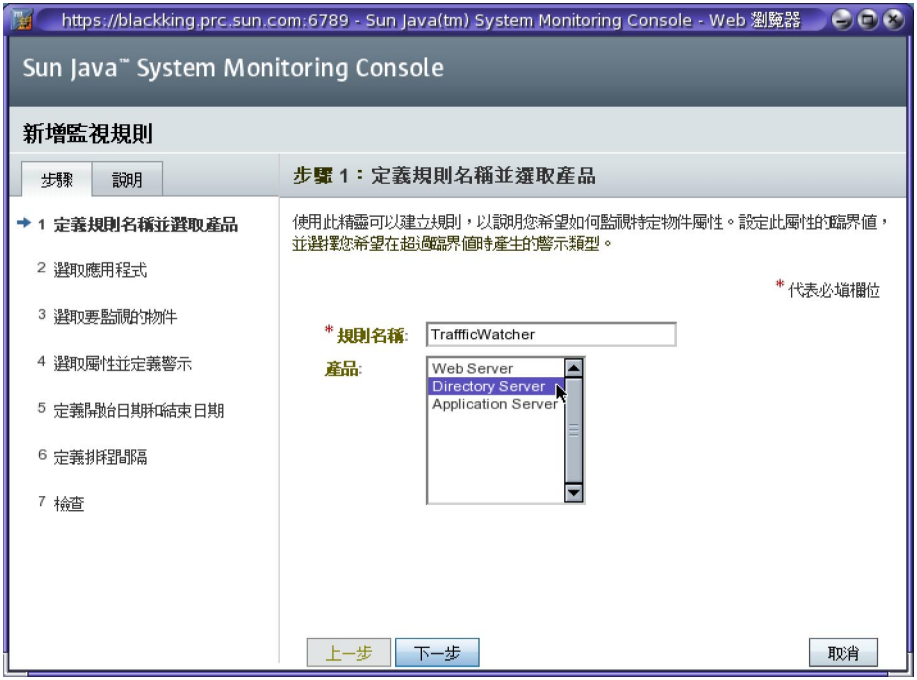
停用主機上的監視時，該主機階層中的所有監視物件都會停用。在停用狀態中，受監視物件雖然可能包含上次的值，但還是不會更新。附屬於已停用物件的監視規則會暫停。若要啓用已停用的主機，請使用主機表格上端的 [啓用] 按鈕遵守此程序。

## ▼ 建立新的監視規則

監視規則也稱爲監視工作，是使用者定義的一組受監視值條件，用來觸發警示。Monitoring Console 的監視規則精靈可協助您定義需要監視的條件。

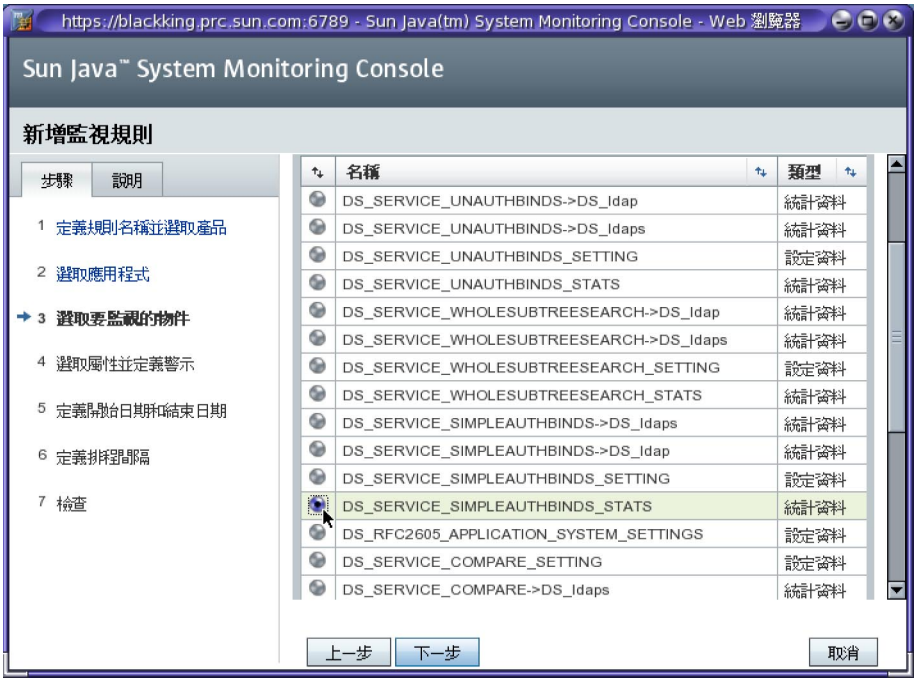
1 **必要時，按一下 Monitoring Console 左側階層根位置的 [部署] 連結，瀏覽至部署層級的顯示內容。此時選取右側窗格的 [監視規則] 標籤，如以下螢幕擷取所示，然後按一下監視規則表格中的 [新增]：**


2 命名新的監視規則，並選擇需要監視的伺服器類型。



3 選擇需要監視的元件產品實例。如果有相同產品在兩個主機上分別安裝實例，某些實例會在此表格中顯示相同的名稱。在這種狀況下，這些實例的順序可能與它們在左側窗格階層中的順序相同，但是目前無法確定。您可能需要針對這兩個實例建立相同的監視規則，以確保您的規則已確實定義。

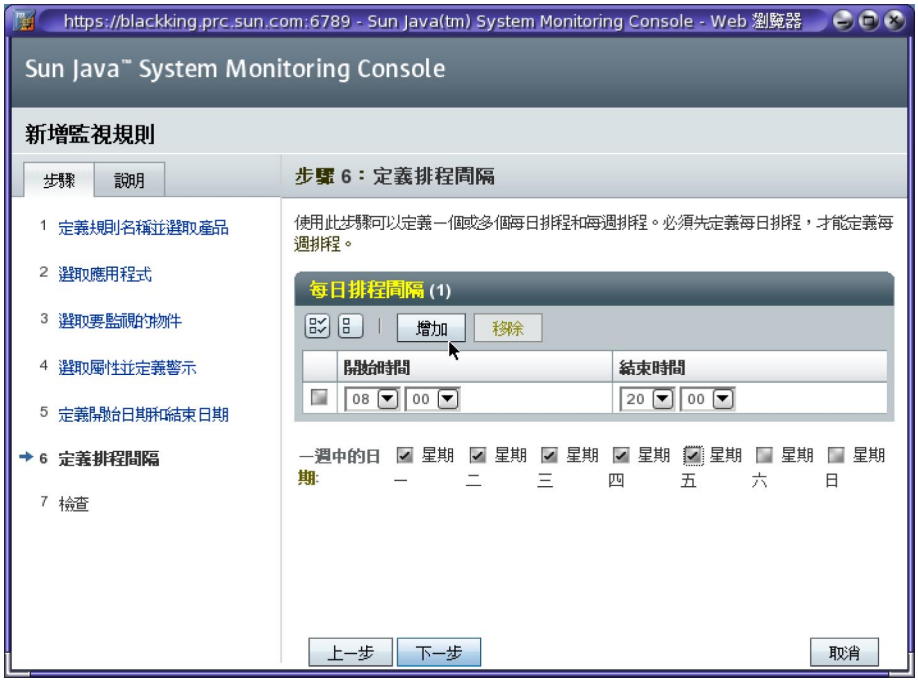

4 選取需要監視其屬性的物件：




**5** 現在，您終於可以指定受監視的屬性，以及會產生警示的值。


6 輸入規則的開始日期和結束日期。這和決定規則執不執行的排程不同，開始日期和結束日期會決定規則的定義存在時間。如果開始時間是過去的時間 (預設值一向如此)，將立即開始執行與此規則相關的監視。


**7** 您可以使用控制來建立規則受到持續監視的一或多段時間範圍。您也可以選取一週內的幾天來建立每週排程。




**8** 在規則精靈的這個最後步驟中，檢查您輸入的內容，並且按一下 [結束] 以建立新規則。


規則精靈結束時，[監視規則] 標籤應該會再次出現，而您定義的新規則將出現在規則表格中。

# Monitoring Console 疑難排解

另請參閱「適用於 UNIX 的 Sun Java Enterprise System 5 版本說明」中所列的已知問題。

如果主代理程式與節點代理程式發生衝突，請檢查下列條件：

- 如果您使用 Solaris Zones，請確定您已在稀疏根本機區域中安裝 Monitoring Console。
- 確定主機或區域中未遺留任何先前安裝的受監視元件。
- 不論是哪種情形，您都需要解除安裝 Monitoring Console 和所有元件、更正問題並重新安裝 Monitoring Console。

如果您解除安裝 Monitoring Console 並重新安裝在相同的主機上，其將無法初始化，且不會出現在 Web 主控台中。發生這種狀況時，請在 Monitoring Console 主機上執行 `masetup –i` 指令以初始化主代理程式。然後按照第 48 頁的 「啓動 Monitoring Console」的程序進行。

監視規則只有在處於使用中狀態時，才能加以停用。如果您需要停用目前已排程爲停用的規則，您必須變更排程以暫時啓用此規則，或將整個規則移除。

由於 Windows 平台的限制，主機統計資料中的 handleCount 和 threadCount 值會保持爲 0 (零)。

◆ ◆ ◆ 附錄 A

# CMM 物件引用

Common Monitoring Model (CMM) 是以 Java 程式語言中實作的共用資訊模型 (CIM) 延伸。CIM 顯示於 com.sun.cmm.cim.* 套裝軟體的 Java 介面中。CMM 顯示於延伸 CIM 介面的 `com.sun.cmm.*` 套裝軟體介面中。在節點代理程式中，監視的物件將由實作 CMM 介面的類別表示。下表顯示各個物件類別可監視的屬性。

## CMM 物件簡介

CMM 以少數核心介面爲基礎，這些介面定義所監視物件類型可顯示的屬性。下面列出的類別代表 CMM 所定義的多種監視物件類型，並提供幾項重要屬性：

| | |
|---|---|
| `CMM_InstalledProduct` | 整個 Java ES 元件產品。例如 Java ES Directory Server。 |
| `CMM_ApplicationSystem` | 已安裝和配置的 Java ES 元件產品實例。可表示執行中或未執行的實例。此物件的一般屬性爲管理員的連絡資訊、系統的作業狀態，以及應用程式的啓動或停止時間。 |
| `CMM_Service` | 元件產品的特定功能，例如 Java ES Directory Server 認證服務。一般屬性爲服務的作業狀態。 |
| `CMM_SoftwareResource` | 表示環境中的軟體實體，例如快取、執行緒池等。一般屬性爲快取大小。 |
| `CMM_LogicalComponent` | 由服務處理的實體，一般使用者可以看見，但不表示實際的實體資源或軟體功能。例如，軟體實例的一組配置參數，而非實例本身。 |
| `CMM_ServiceAccessURI` | 服務可供使用的點。一般屬性爲連接埠號或統一資源識別碼 (URI)。 |
| `CMM_RemoteServiceAccessPoint` | 遠端連線的存取與定址資訊。一般屬性爲 URI 或連線的作業狀態 (開啓或關閉)。 |

| | |
|---|---|
| CMM_Process | 正在執行之程式的單一實例。一般屬性爲記憶體或 CPU 使用率。 |
| CMM_UnitaryComputerSystem | Java ES 部署所使用的單一主機，例如桌上型機器或伺服器。一般屬性爲可用處理器的數量，或實體記憶體的數量。 |
| CMM_OperatingSystem | 使主機電腦硬體可供使用的軟體或韌體。一般屬性爲系統中可用虛擬記憶體的數量。 |
| CMM_JVM | Java ES 伺服器所使用的 Java 虛擬機器。範例屬性爲 Java 虛擬機器的版本編號。 |
| CMM_DatabaseService | 代表資料庫執行的作業，例如提供使用者存取。一般屬性爲資料庫的允許連線數量上限。 |
| CMM_CommonDatabase | 指定類型資料庫上共用的特性。一般屬性爲最近備份的日期。 |

◆ ◆ ◆ 附錄 B

# 各項元件顯示的受監視物件

本附錄各小節列出各個支援監視的產品元件所提供的 CMM 物件。即使只提供物件屬性的子集，也會列出這些屬性。

## Common Agent Container 的設備

尚未說明。

## Access Manager 的設備

尚未說明。

## Application Server 的設備

尚未說明。

## Calendar Server 的設備

尚未說明。

# Directory Server 的設備

尚未說明。

# Instant Messaging 的設備

尚未說明。

# Messaging Server 的設備

尚未說明。

# Portal Server 的設備

尚未說明。

# Web Server 的設備

尚未說明。

# 索引

字彙表，連結到，8