J2EE™ 的 Rational® XDE 模型建構準則

Rational Software 白皮書
TP 154, 05/03

目錄

1. 簡介

此文件提供如何在 Rational XDE™ Java Platform Edition 上代表及建構 RUP 模型構件的建議。當然，您是否決定要在 XDE 中爲這些 RUP® 構件建模，是專案特定決策。在本文件中，我們加註 XDE 提供自動化支援的那些模型，以及不提供支援的那些模型，因爲這會影響您的決定。

由於所有 XDE 模型存在於 XDE 專案內，因此 XDE 專案結構這一節提供有關應建立哪些 XDE 專案以及應在那些專案中建立哪些 XDE 模型檔的建議。

RUP 和 XDE 都使用「模型」這個詞彙，而 RUP 模型和 XDE 模型之間的對映不一定是一對一。在 RUP 模型對 XDE 模型的對映這一節，有描述 RUP 模型到 XDE 模型的對映。

XDE 模型檔中每一個 RUP 模型構件的結構，會在它自己的區段中加以描述。

2. 範圍

本文件重點在描述建議的 XDE 模型檔結構，而不是在開發相關聯的 RUP 構件內容的流程上。本文件也不描述詳細啓發，它定義包含所描述之 XDE 模型的 XDE 專案。如需如何定義、開發及建模 RUP 構件內容的相關資訊，請參閱 RUP。如需專案的詳細資訊，請參閱 IDE 文件。

本文件不說明完整範例，而是使用已選取的範例來強調涵蓋的要點；然而，所有範例均彼此一致，而且是取自實際的 XDE 模型。

這一版的文件不討論標示庫開發。

本文件所描述的專案和模型結構只是建議，您可以用任何同等有效的結構加以取代。

3. XDE 專案結構

本文件的焦點在於如何建構 XDE 模型。然而，由於所有 XDE 模型都存在於 XDE 專案內，因此，我們一定要提供專案結構的簡介，其中含有我們建議的模型結構。

對於許多人正在開發的 J2EE 企業應用程式，我們建議您建立下列 XDE 專案和模型。

附註：如果有使用 XDE 建立專案 精靈，在建立專案時，就會自動建立許多模型。事實上，如果您使用的是 WSS AD XDE，如果您建立*企業應用程式建模專案*，則會自動建立這個多重專案結構的大部分，包括許多模型在內。 XDE 也提供模型範本來快速提升模型內容。

XDE 專案	說明	XDE 模型 <建議的模型名稱> (<XDE 檔案類型：模型範本>]
應用程式專案（XDE 基本建模專案）	應用程式專案代表整個應用程式。包含說明全部應用程式的 XDE 模型檔	- 使用案例模型（Rational XDE：使用案例模型） - 分析模型（Rational XDE：分析模型） - 整體設計模型（Rational XDE：設計模型） - 整體實作模型（Rational XDE：空白模型） - EAR 部署模型（Java：EAR 部署模型）
資料建模專案（XDE 資料建模專案）	資料建模專案包含應用程式資料建模所需的資源，以及資料模型與資料庫之間的來回轉換工程。	- 邏輯資料模型（資料：邏輯資料模型） - 實體資料模型（資料：*供應商特定實體資料模型檔* ）[1] - 網域模型（資料：*供應商特定網域模型檔*）
EJB 專案（XDE EJB 建模專案）	EJB 專案包含實作 EJB 所需的資源。內含的元素已套裝及部署爲 EJB 模組（.EJB - JAR 檔）。 個別的 EJB 專案可定義給個別的 EJB 或 EJB 集合（每一個 EJB 專案可包含至多一個 Java 程式碼模型）。建議爲每一個要產生的 EJB-JAR 建立一個 EJB 專案。如果有定義個別專案，則專案的名稱應反映其內容。[2]	- EJB 程式碼模型（Java：EJB 程式碼模型） - EJB 部署模型（Java：EJB 部署模型）
Web 專案（XDE Web 建模專案）	Web 專案代表應用程式的 Web 資源。內含的元素已套裝及部署到 Web 保存檔（WAR 檔）。 個別的 Web 專案可定義給呈現邏輯的特定領域。建議爲每一個需要產生的 WAR 建立一個 Web 專案。如果有定義個別專案，則專案的名稱應反映其內容。[3]	- Java 程式碼模型（Java：Java 1.3/1.4 程式碼模型） - JSP 標示庫模型（Web：JSP 標示庫模型）[4] - 虛擬目錄模型（Web：虛擬目錄模型）[5] - Web 部署模型（Web：Web 部署模型)

[1] Rational XDE 提供實體資料庫支援給多個資料庫供應商。 XDE 支援的每一個資料庫供應商有供應商特定的範本。

[2]如果您使用 XDE for WSAD，當您建立其他的 EJB（建模）專案時，此精靈將要求應用程式專案管理 EAR。您應該重複使用上述相同的應用程式（建模）專案。

[3]如果您使用 XDE for WSAD，當您建立其他的 Web（建模）專案時，此精靈將要求應用程式專案管理 EAR。您應該重複使用上述相同的應用程式（建模）專案。

[4]每一個專案可以有多個標示庫模型。事實上，每一個 .tld 檔需要個別模型。無論如何，這一版的文件不討論標示庫開發。

[5]每一個 XDE Web 專案可以有多個虛擬目錄模型。

這種專案和模型組織的一個範例顯示在圖 1（注意唯一的模型名稱）。

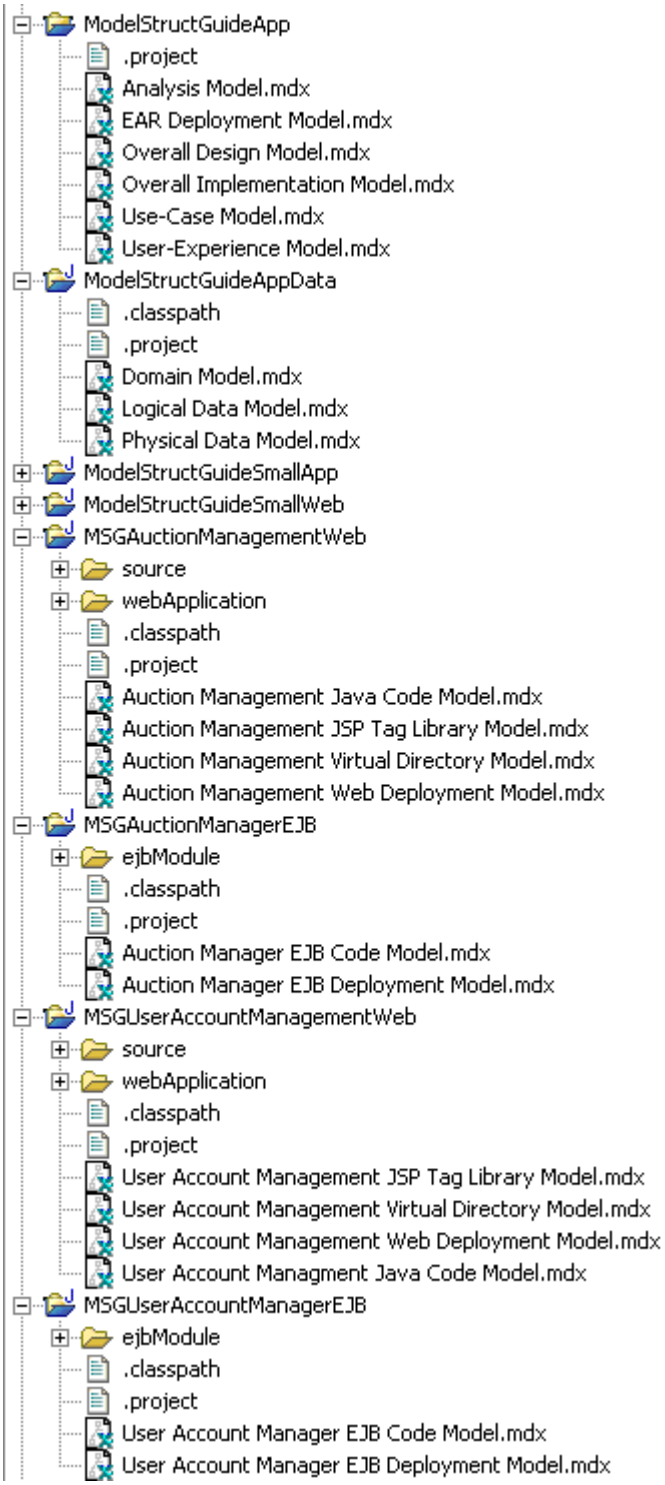

圖 1：XDE 專案與模型組織範例

另外，如果應用程式真的很小，而且只由一人開發，則上述專案結構可簡化成兩個專案，一個包含全應用程式和非 Web 元素，另一個包含 Web 元素。除了減少專案數之外，也可以減少模型數。例如，對於小型單人專案，可簡化如下：

- 不維護個別分析模型。分析與設計都是在 XDE 來回轉換模型中執行。
- 不維護「整體設計模型」和「整體實作模型」。專案夠小，故可直接查看 XDE 來回轉換模型而得到概觀。同時，使用案例實現化是在 EJB 程式碼模型中維護，並包含對虛擬目錄模型中的元素的參照。
- 不維護個別邏輯資料模型。實體資料綱目直接在「實體資料模型」中開發。

下表是這種「小型專案結構」的總結。

XDE 專案	**說明**	**XDE 模型** **<建議的模型名稱> (<XDE 檔案類型：模型範本>]**
應用程式專案（XDE EJB 建模專案）	應用程式專案代表應用程式的非 Web 層面。它包含描述應用程式整體的模型、資料模型和 EJB 特定模型。	- 使用案例模型（Rational XDE：使用案例模型） - 實體資料模型（資料：*供應商特定的實體資料模型檔*） - EJB 程式碼模型（Java：EJB 程式碼模型） - EJB 部署模型（Java：EJB 部署模型） - JAR 部署模型（Java：EAR 部署模型）
Web 專案（XDE Web 建模專案）	Web 專案代表應用程式的 Web 資源。內含的元素已套裝及部署到 Web 保存檔（WAR 檔）。	- Java 程式碼模型（Java: Java 1.3/1.4 程式碼模型） - JSP 標示庫模型（Web：JSP 標示庫模型）[6] - 虛擬目錄模型（Web：虛擬目錄模型）[7] - Web 部署模型（Web：Web 部署模型)

小型專案和模型組織的一個範例顯示在圖 2

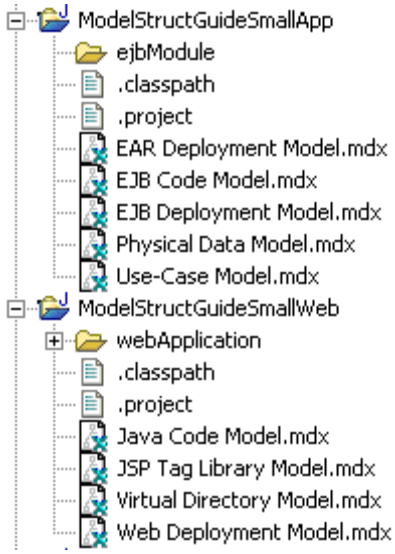

圖 2：小型 XDE 專案和模型組織範例

專案和個別模型檔數量的實際選取是架構選項，可因不同專案而異。然而，不論定義了多少個專案，每一個專案只能有一個 XDE Java 程式碼模型檔。如需專案和它們可包含的 XDE 模型檔的相關資訊，請參閱 XDE 文件。

同時強烈建議，在所有 XDE 專案中，*XDE 模型名稱必須是唯一的*。在嘗試解決 XDE 模型之間的參照時，這點非常重要。如需交互模型參照及解決問題的相關資訊，請參閱 XDE 文件。

[6]每一個專案可以有多個標示庫模型。事實上，每一個 .tld 檔需要個別模型。無論如何，這一版的文件不討論標示庫開發。

[7]每一個 XDE Web 專案可以有多個虛擬目錄模型。

本文件的範例所使用的專案和模型結構是顯示在圖 1 。請注意，已定義多個 EJB 和 Web 專案。有關多個 EJB 和 Web 專案的基本理由，請參閱實作子系統這一節。

4. RUP 模型到 XDE 模型的對映

在說明如何在 XDE 表示 RUP 模型構件之前，務必先解決「RUP 模型」與「XDE 模型」之間的混淆問題，因爲它們是不同的，而且從 RUP 模型對映到相關聯的 XDE 模型不一定都是一對一（接近，但不是一對一）。由於「模型」同時使用於 RUP 和 XDE 中，因此初步假設它們應該相同。然而，RUP 中的模型分隔流程重點（分析 vs. 設計 vs. 實作等等），而 XDE 中的模型則分隔開發重點（分隔程式碼模型以說明程式語言套裝結構與虛擬目錄結構，爲不同的程式語言和開發環境分隔程式碼模型等等）。爲了減輕混淆情形，在本白皮書的內容中，「模型」這個詞彙明確地限定爲 RUP 或 XDE。

下表彙總 RUP 模型到 XDE 模型的對映。XDE 模型就是在 XDE 專案結構這一節所介紹的那些模型。每一個 XDE 模型的結構將在本白皮書的後面章節加以描述。

RUP 模型	**<XDE 專案>：< XDE 模型名稱>**
使用案例模型	應用程式專案：使用案例模型
分析模型	應用程式專案：分析模型
設計模型	應用程式專案：整體設計模型 [每一個 XDE 來回轉換模型檔中的設計類別（如下所示）]
資料模型	XDE 資料模型： - 資料建模專案：邏輯資料模型 - 資料建模專案：*供應商特定*實體資料模型 - 資料建模專案：*供應商特定*網域模型
實作模型	應用程式專案：整體實作模型 XDE 來回轉換模型[8] - EJB 專案：EJB 程式碼模型 - Web 專案：Java 程式碼模型 - Web 專案：JSP 標示庫模型 - Web 專案：虛擬目錄模型
部署模型	XDE 部署模型[9] - 應用程式專案：EAR 部署模型"[10] - EJB 專案：EJB 部署模型 - Web 專案：Web 部署模型

[8]爲了簡單起見，我在本白皮書中將使用「XDE 來回轉換模型」來代表這些 XDE 模型。

[9]爲了簡單起見，我在本白皮書中將使用「XDE 部署模型」來代表這些 XDE 模型。

[10]EAR 部署模型與個別 XDE 部署模型「交錯」。它包含圖表來說明部署節點及其連接。它也包含將個別部署模型中已定義的個別保存檔對映到部署節點的圖表。

5. 使用案例模型

「使用案例模型」的建議結構顯示在圖 3

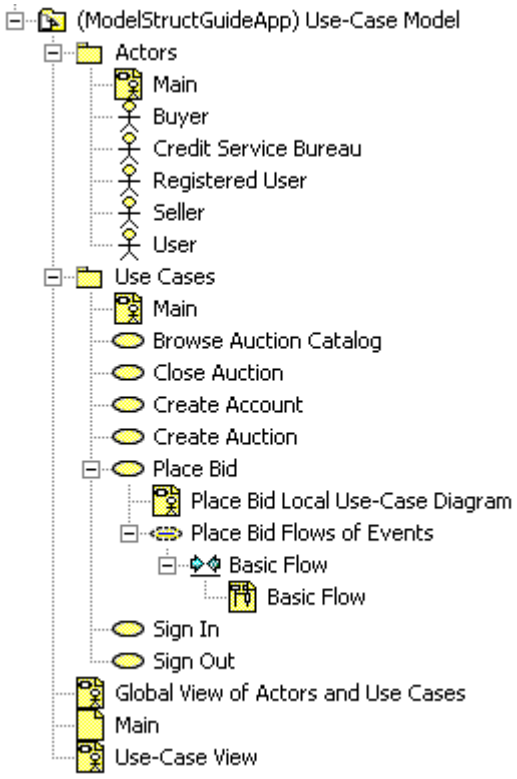

圖 3：使用案例模型結構

「使用案例模型」分割爲兩個套件：「參與者」和「使用案例」。

除了包含**參與者**和**使用案例**的**使用案例模型**圖表之外，還可以使用其他圖表來闡明**使用案例**的不同層面。下列補充模型元素可併入到**使用案例模型**中的**使用案例**模型元素之下，如下所示圖 3：

- 「競標區域使用案例圖表」圖表包含「競標」**使用案例**和參與該**使用案例**的**參與者**。
- 「事件競標流程」合作實例包含互動實例，以圖形方式說明使用案例說明中所描述的事件流程（亦即，**參與者**和**使用案例**之間的互動）。使用案例合作實例不得與**使用案例實現化**混淆不清（請參閱分析模型這一節和設計使用案例實現化這一節的說明），因爲「使用案例模型」中的合作實例完全是「黑箱作業」，並不說明應用程式內元素的互動情形。
- 「事件競標流程」活動圖形包含活動圖，以圖形方式說明使用案例說明中所描述的事件流程。

在圖 3 所顯示的範例中，圖 3 中的「參與者和使用案例的廣域視圖」圖表包含所有**使用案例**和**參與者**及其關係，這和「主要程式」圖表不同，「主要程式」圖表只包含有「主要程式」圖表的套件中的元素。如果有許多**參與者**和**使用案例**，可使用多個圖表來表達「參與者和使用案例的廣域視圖」圖表的資訊。

「使用案例視圖」圖表代表軟體架構的使用案例視圖。如需架構視圖的相關資訊，請參閱 RUP。

您可以在「參與者」和「使用案例」套件中建立其他套件，以進一步組織內含的模型元素，如圖 4 所示

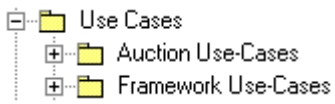

圖 4：其他的使用案例套件分割

6. 分析模型

分析模型是**分析類別**和分析**使用案例實現化**所在之處。

附註：是否應該維護個別的**分析模型**和**設計模型**，屬於專案特定決策。如果有建立但未維護個別的**分析模型**，則**分析類別**將移到適當的**設計模型**分割[11] 且已修正。另一個選項是在**設計模型**中建立**分析類別**和**分析使用案例實現化**，然後從該處將它們發展到設計表單中。請參閱設計模型這一節，以瞭解如何在 XDE 中表示**設計模型**。

分析模型的建議結構顯示在圖 5

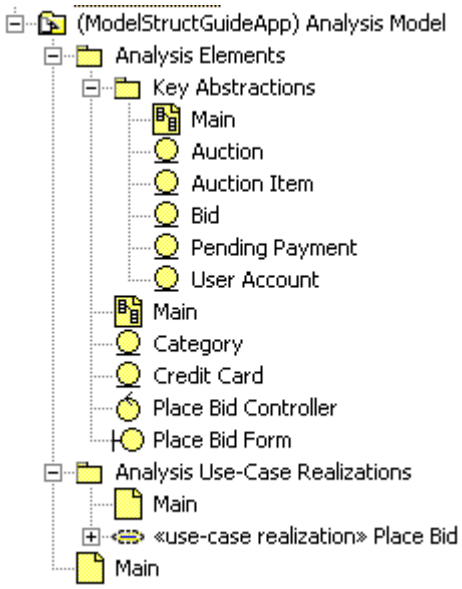

圖 5：分析模型結構

「分析元素」套件包含**分析類別**。**分析類別**的實例出現在「分析使用案例實現化」套件的圖表中。

除了**分析類別**之外，也可以在「分析元素」套件中定義套件，以進一步分割內含的**分析類別**（請參閱圖 5 中的「關鍵摘要」套件）。此額外分割是選用性的，尤其不維護個別 **分析模型**時，可選用它。在這些情況中，**分析類別**可視爲「暫時性」（亦即，它們會存在直到發展成設計元素爲止），因此其組織被認爲不重要。一個可能的異常狀況是關鍵摘要**分析類別**。

如圖 5 所示，「關鍵摘要」套件包含的**分析類別**，被認爲代表系統的關鍵摘要。如前所述，這個套件是選用性的。另一個替代方案是在「分析元素」套件中的類別圖上代表關鍵摘要。然而，建立個別套件能夠更明確地將**分析類別**分類爲關鍵摘要。事實上，即使未全面維護個別**分析模型**，有些專案仍選擇維護關鍵摘要 **分析類別**。在這些情況中，定義個別套件來包含所維護的**分析類別**會有幫助。
附註：關鍵摘要也會出現在「整體設計模型」的「邏輯視圖：關鍵摘要」圖表中。如需相關資訊，請參閱設計模型這一節。

「分析使用案例實現化」套件包含分析層次**使用案例實現化**，它們就「 分析元素」套件中的**分析類別**而論，說明如何執行**使用案例**。每一個分析**使用案例實現化**會實現 **使用案例模型**中的一個**使用案例**，其名稱與該**使用案例**相同，而且應具有圖 6 所顯示的結構：

[11]稍後您會看到，正確的「設計模型分割」就是其中一個 XDE 來回轉換模型中的一個套件，因爲技術特定元素的設計是在來回轉換模型中執行。

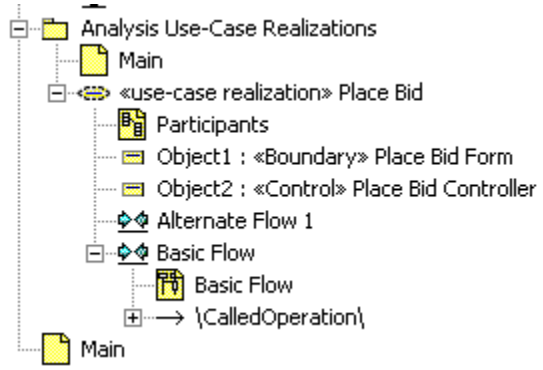

圖 6：「分析使用案例實現化」套件結構

「參與者」圖表顯示參與**使用案例實現化**的**分析類別**（來自「分析元素」套件），亦即，其實例出現在互動圖中的那些**分析類別**，以及支援互動圖所描述之協同作業的關係。

「流程」互動實例（「基本流程」和「替代流程 1」）包含序列圖，說明事件的**使用案例**流程。每一個事件重要使用案例流程應該有一個互動實例。在相關聯的**使用案例**的執行期間，互動實例中的序列圖說明參與的**分析類別**之間的流程。

7. 設計模型

RUP **設計模型**由多個 XDE 模型表示 –「整體設計模型」和位於個別 XDE 來回轉換模型中的來回轉換設計元素（來回轉換設計元素是參與來回轉換工程的詳細設計元素）。如此一來，即可運用個別來回轉換模型中可用的自動化。例如，XDE EJB 型樣可用來建立指定 EJB 的類別。

「整體設計模型」說明整體應用程式的設計，並包含跨越多個 XDE 來回轉換模型的元素。它包含邏輯分割區來啓發個別來回轉換模型的組織，以及**使用案例實現化**來連結一切（**使用案例實現化**說明不同來回轉換模型的設計元素之間的協同作業）。「整體設計模型」包含了參照來回轉換設計元素的圖表。如需個別 XDE 來回轉換模型的相關資訊，請參閱實作模型這一節。

另一個可能性是在相同 XDE 程式碼模型中代表**設計模型**和**實作模型**。唯有當您只有一個目標實作語言，而且團隊規模很小時，這才行得通。如需小型專案結構的範例，請參閱 XDE 專案結構這一節。

維護「整體設計模型」是選用性的，但卻是組織圖表、提升摘要層次及提供設計元素場所的好辦法，而且還能得知要應用的實作機制。

「整體設計模型」的建議結構顯示在圖 7

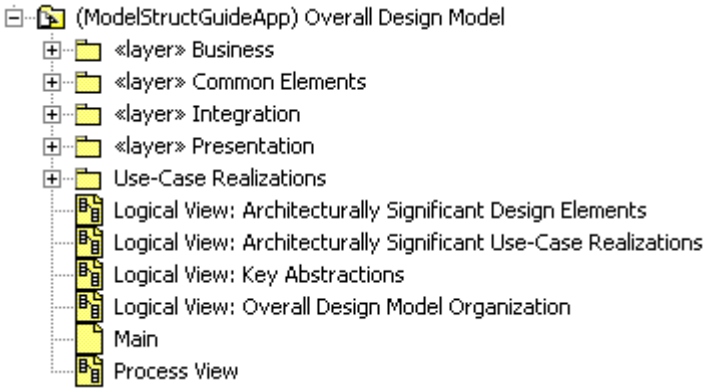

圖 7：整體設計模型結構

「整體設計模型」包含下列套件：

- <<layer>> 套件包含（或包含參照的圖表）系統的設計元素（**設計類別**、**介面**和**設計子系統**）。 此結構代表特定分割策略，這在設計層這一節已有說明。
- 「使用案例實現化」套件包含設計層次 使用案例實現化。關於使用案例實現化的內部結構，在設計使用案例實現化這一節有更詳細的討論。

代表架構視圖的圖表，其圖表名稱中有包含 “View”。如需架構視圖的相關資訊，請參閱 RUP。

「邏輯視圖：關鍵摘要」圖表包含系統的關鍵摘要。有數個選項可維護這些關鍵摘要：

- 有維護完整的**分析模型**。在該情況下，「邏輯視圖：關鍵摘要」圖表包含來自**分析模型**的**分析類別**，它們代表系統的關鍵摘要。
- 只維護部分**分析模型**，亦即，只有關鍵摘要。在該情況下，「邏輯視圖：關鍵摘要」圖表包含來自**分析模型**的**分析類別**，它們代表系統的關鍵摘要。
- 不維護**分析模型**的任一部分。在該情況下，代表關鍵摘要的**分析類別**可維護在**設計模型**的一個套件中，它叫作「關鍵摘要」

如需**分析模型**的相關資訊，請參閱分析模型這一節。

7.1 設計層

<<layer>> 套件包含系統的設計元素（例如 **設計類別**、**介面** 和 **設計子系統**），它們是從 **分析類別**發展而成的。 <<layer>> 套件可包含任何數量的子套件，它們進一步分割內含的設計元素。設計**使用案例實現化**（包含在「設計模型」的「使用案例實現化」套件中，並於 設計使用案例實現化這一節的標題之下討論）是從這些套件所包含的設計元素方面來撰寫的。

設計模型可遵循任何數量的分割策略。這一節所描述的分割策略顯示在圖 8

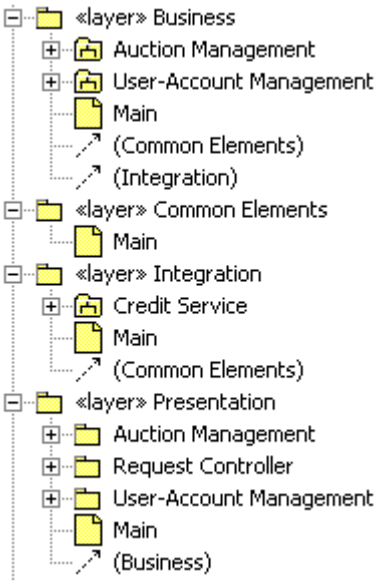

圖 8：設計套件分割範例

在這個範例中，第一層套件被視爲層，其中每一層有特定責任。第二層套件依商業功能進一步分割層套件元件。

「呈現」層套件負責處理與一般使用者的互動。在 J2EE 應用程式中，可能位於「呈現」層 套件的設計元素包括 HTML 頁面、JavaServer Pages (JSP) 和 Servlet。您可以進一步將「呈現」層套件分割成子套件，將屬於同一組相關**使用案例**的元素加以分組；例如圖 8 的「拍賣管理」套件：

「商業」層套件負責執行任何商業流程。在本文件所呈現的「設計模型」結構中，「商業」層套件包含一組設計子系統套件，每一個主要商業功能各一個（例如圖 8 的「拍賣管理」和「使用者帳號管理」子系統套件）。**設計子系統**套件在設計子系統這一節的標題下有更詳細的說明。

「整合」層套件負責提供後端系統資源的存取，包括資料庫和外部系統。在本文件所呈現的**在設計模型**結構中，「整合」層套件也包含設計子系統套件，每一個外部系統各一個（例如圖 8 的「信貸服務」子系統套件）。**設計子系統**套件在設計子系統這一節的標題下有更詳細的說明。

「共用元素」層套件包含各層共用的元素。

同樣地，這一節所描述的結構可以用不同的結構來取代，以反映不同的分割策略。

7.2 設計子系統

設計子系統由「整體設計模型」中的子系統套件來表示。每一個設計子系統套件應該具有相同的結構。結構的細節因所擷取的**設計子系統** 的詳細程度而異。

更正式而嚴格的**設計子系統**結構的範例顯示在圖 9

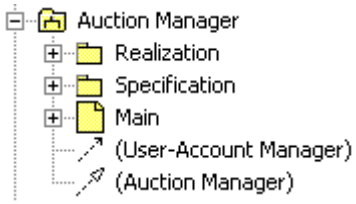

圖 9：設計子系統結構

此設計子系統套件結構支援在設計子系統套件內定義個別的「規格」和「實現化」套件。此結構受到 *UML Components: A Simple Process for Specifying Component-Based Software*（作者：J. Cheesman 和 J. Daniels）這本書的影響。您可以使用不包含這些分割的簡化設計子系統套件結構，這並不影響本文件所定義的其他模型檔案結構。下列章節將討論每一個「規格」和「實現化」套件。

7.2.1 子系統規格

「規格」套件包含**設計子系統**介面的說明。[12] 子系統規格的一個範例顯示在圖 10

圖 10：設計子系統規格範例

[12]在這個簡單的範例中，您可能想知道針對該介面的個別套件需求。然而，在真實專案中，該套件是值得維護的，因爲它可包含對於說明該子系統的參照，尤其，它包含介面限制，例如作業的先決條件和後續條件。

7.2.2 子系統實現化

「實現化」套件包含**設計子系統**規格如何實現的說明。設計子系統套件的「實現化」套件的範例顯示在圖 11

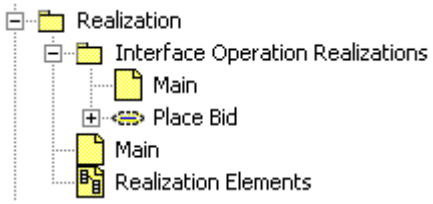

圖 11：設計子系統實現化範例

「實現化元素」圖表包含對於實現子系統的設計元素的參照。設計元素本身可位於「實現化」套件中，或位於它們參與來回轉換工程的個別 XDE 程式碼模型中。如果需要詳細資訊，請參閱 XDE 來回轉換模型這一節。

「介面作業實現化」套件包含合作實例，說明子系統元素如何實現**設計子系統**介面的重要作業（在「規格」套件中）。每一個重要子系統介面作業有一個合作實例。[13]「介面作業實現化」套件的一個範例顯示在圖 12

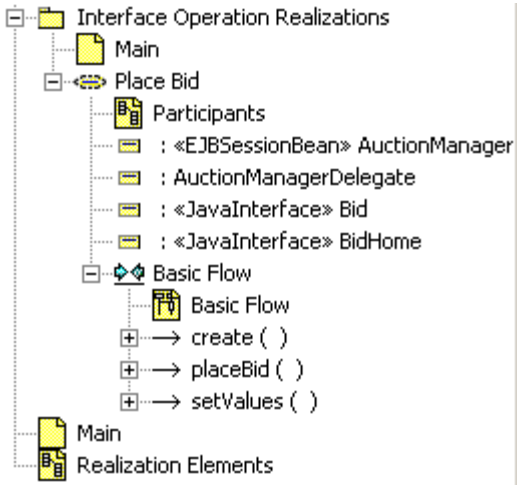

圖 12：介面作業實現化套件範例

就像分析層次**使用案例實現化**（稍早在分析模型這一節已討論過）和設計層次**使用案例實現化**（稍後將在設計使用案例實現化這一節加以討論）一樣，每一個介面作業實現化包含一個類別圖，內含參與實現化的子系統元素（ 圖 12 中的「參與者」圖表），以及一個互動圖，它說明那些參與者如何分工合作來執行子系統介面作業（ 圖 12 中的「基本流程」圖）。

[13]並非所有作業都需要定義在這個層次。有些較簡單的作業不需要個別的合作實例。

7.3 設計使用案例實現化

「使用案例實現化」套件包含設計層次**使用案例實現化**。每一個**使用案例實現化**與 **使用案例模型**中的一個**使用案例**相關聯，其名稱與該**使用案例**相同，而且應具有「圖 16」所顯示的結構。

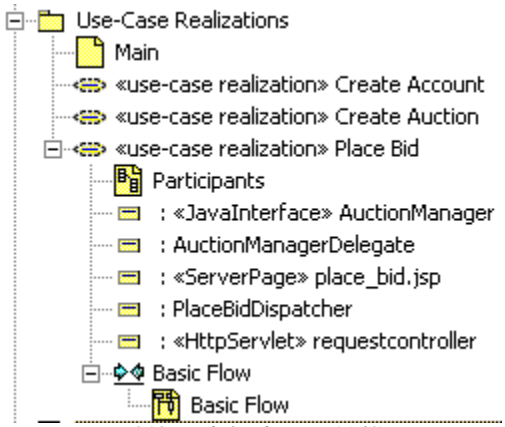

圖 13：設計使用案例實現化結構

使用案例實現化「參與者」圖表顯示參與**使用案例實現化**的設計元素（亦即，其實例出現在**使用案例實現化**互動圖中的那些設計元素），以及支援互動圖所描述之協同作業的關係。

「基本流程」圖是互動圖的一個範例，它說明在相關聯的**使用案例**的執行期間，參與的設計元素之間的流程。**使用案例**中的每一個事件流程應該有一個互動實例。

請注意，**使用案例實現化**圖表可能（而且經常會）包含對於設計元素的參照，這些設計元素實際上是位於個別 XDE 來回轉換模型中。**使用案例實現化**示範個別的來回轉換模型中的元素之間的合作關係。

8. 資料模型

RUP **資料模型**是由多個 XDE 模型檔來表示：

- **邏輯資料模型**（選用性）。代表邏輯資料模型，是資料庫的邏輯設計之應用程式獨立視圖。
- **實體資料模型**。代表資料庫供應商特定的實體資料模型。它包含詳細模型元素來定義資料庫表格的特定性質。「實體資料模型」XDE 模型檔也包含資料庫特定實作構件，來於供應商特定資料庫中實作這些表格。
- **網域模型**（選用性）。代表可用來定義「實體資料模型」之一致資料類型的資料庫供應商特定資料類型。

XDE 模型檔的分隔提供了最佳彈性來支援**設計模型**、**資料模型**和實體資料庫之間的自動化。

以下對每一個 XDE 模型檔有更詳細的說明。

8.1 邏輯資料模型（選用性）

當專案需要建立對資料庫設計很重要的主要實體和關係的獨立式邏輯資料表示法時，「邏輯資料模型」可使用於這類情況。建立 XDE 邏輯資料模型是選用性的，因爲資料庫設計團隊可以將 **設計模型**中的持續性 **設計類別**轉換成**資料模型**中的表格，直接在 XDE 實體資料模型中建立起始實體資料庫設計結構（請參閱下一節實體資料模型）。

XDE 邏輯資料模型可依需要分割成主題區套件。主題區套件定義實體類別的邏輯群組。 XDE 邏輯資料模型也可包含「共用元素」套件，內含跨越主題區的模型元素。

名稱中有 “View” 的圖表是用來記載架構的資料視圖。「資料視圖：整體邏輯資料模型組織」圖表是用來記載邏輯資料模型的高階資料組織，如 XDE 邏輯資料模型的主要分割（亦即套件）中所表示。「資料視圖：主要邏輯資料元素」是用來記載**資料模型**的主要邏輯元素。如需架構視圖的相關資訊，請參閱 RUP。

「邏輯資料模型」的建議結構的一個範例顯示在圖 14

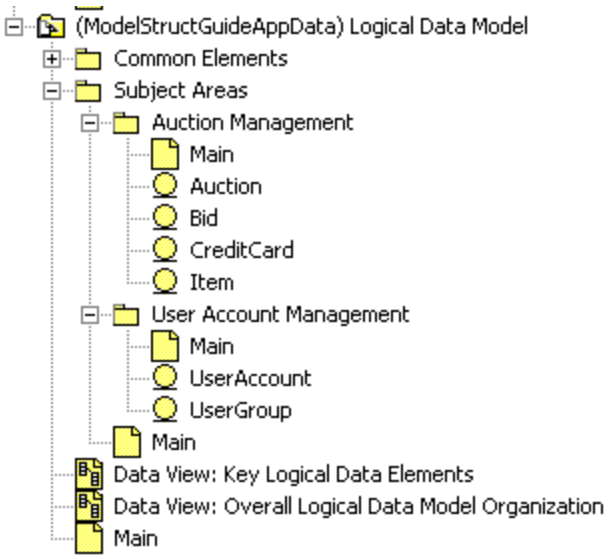

圖 14：邏輯資料模型結構

在這個範例中，有兩個主題區套件：「拍賣管理」和「使用者帳號管理」。每一個主題區套件包含構成邏輯資料模型的實體類別。但並沒有直接對映到 **設計模型**中的套件結構，只是有些類似而已。

8.2 實體資料模型

實體資料模型包含詳細的資料庫表格和預存程序設計，它們透過 XDE Data Modeler 正向工程機能來實作資料庫。實體資料模型也包含用來定義資料庫實體儲存體配置的模型元素。一般而言，模型元素包含資料庫和表格空間，它們再構成目標儲存媒體上的資料庫表格的實際佈置。

在建立實體資料模型時，資料庫設計師必須選取適當的目標資料庫。支援的資料庫包括：DB2 MVS、DB2 UDB、Oracle、Sybase 和 SQL Server。 XDE 將 XDE 模型檔名稱預設爲已選取的資料庫。在本文件的「實體資料模型」範例中，XDE 模型檔名稱已更新爲「實體資料模型」。在建立「實體資料模型」時，資料庫設計師可選擇接受預設名稱。

實體資料模型的建議結構的一個範例顯示在圖 15

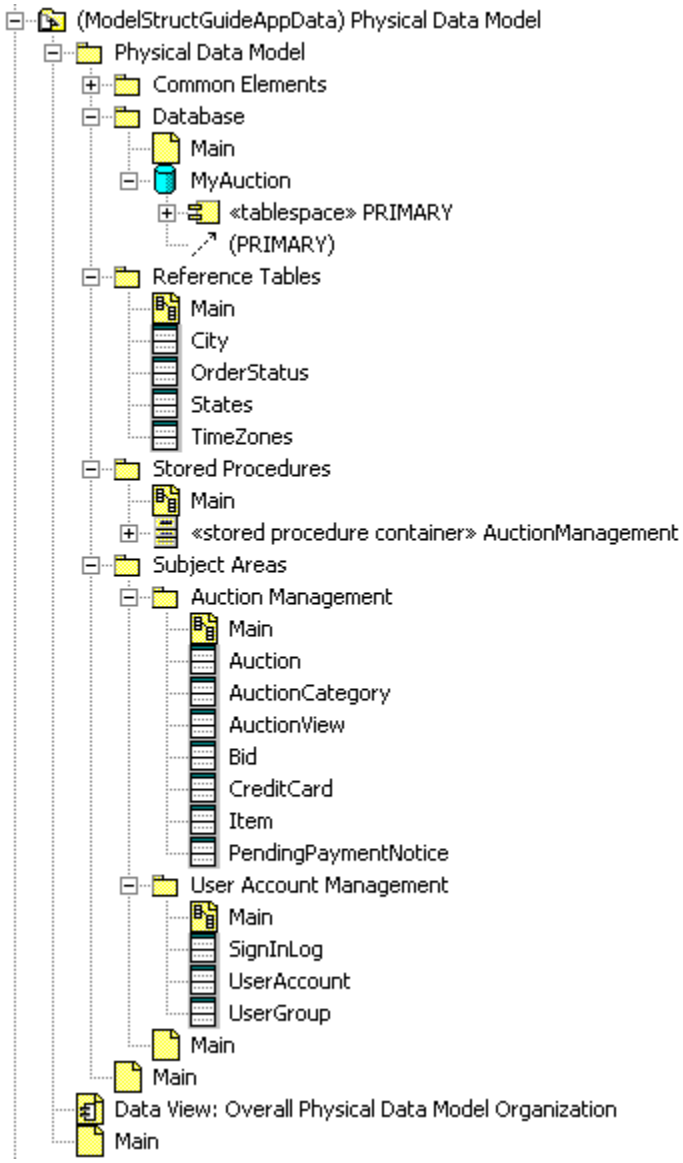

圖 15：「實體資料模型」結構

「共用元素」套件包含資料庫表格和跨越主題區的視圖。

「資料庫」套件包含定義資料庫實體儲存體配置的模型元素。它包含資料庫和表格空間，它們構成目標儲存媒體上的資料庫表格的實際佈置。表格空間是用來邏輯地群組資料庫內的表格。如需定義表格空間的準則，請參閱 RUP。「資料庫」套件可依需要分割爲較低階的套件，視應用程式的複雜度而定。

在圖 15 所顯示的範例中，「資料庫」套件包含單一資料庫，MyAuction，其相關聯的表格空間 PRIMARY 和表格實現化關係。表格空間可命名爲任何適合資料庫專案的名稱。對於 MyAuction 資料庫，只定義一個叫作 PRIMARY 的表格空間。執行正向工程時，會建立透過與資料庫表格空間的實現化關係而鏈結到資料庫的表格（在資料庫或 DDL 中）。

「參照表格」套件包含靜態資料表格，它們保留應用程式所需的「固定」資料資訊。

「預存程序」套件包含代表資料庫預存程序的所有類別（<<stored procedure container>> 類別和相關聯的 <<stored procedure>> 作業）。與單一表格相關的預存程序可連同預存程序所參照的表格，一起套裝在「預

存程序」套件或「主題區」套件中，視您要呈現一個「以預存程序爲中心」或「以表格爲中心」的視圖而定[14]

「主題區」套件包含邏輯地分組相關表格和視圖集合的套件[15]，建議在主題區套件中建立視圖及表格。此建議純粹是爲了組織而已。在有使用視圖的主題區中保存視圖很有幫助，這樣可以把它們放在與表格一樣的主題區內。在圖 15 所顯示的範例中，有兩個主題區套件：「拍賣管理」和「使用者帳號管理」。主題區套件的數量視應用程式的複雜度而定。然而，一般而言，「邏輯資料模型」中的主題區套件會「啓發」實體資料模型中的主題區套件。邏輯資料模型中的主題區是實體資料模型中的主題區的摘要。

主題區套件中的表格包含對表格所定義的直欄和觸發程式。表格是透過下列其中一項而建立

- XDE 類別到表格轉換功能。
- XDE 對現有的資料庫功能進行反向工程[16]
- 由資料庫設計師手動建立。

對現有的資料庫進行反向工程時，會在 XDE 實體資料模型中建立綱目套件。這些套件的名稱是以資料庫擁有者爲基礎[17] 進行反向工程的資料庫。建議反向工程的表格移到 主題區　套件內的主題區套件中，並刪除反向工程的綱目套件。將表格移到主題區套件的作用是組織表格，使資料庫設計師可以依需要來更新表格。”

名稱中有 “View” 的圖表是用來記載架構的資料視圖。「資料視圖：整體實體資料模型組織」圖表是用來記載實體資料模型的高階資料組織，如 XDE 實體資料模型的主要分割（亦即套件）中所表示。如需架構視圖的相關資訊，請參閱 RUP。

8.3 網域模型（選用性）

網域模型是選用性的 XDE 模型，用來儲存資料庫的使用者定義資料類型。網域可讓資料庫設計師在資料庫設計上重複使用元素內容。資料庫設計師使用網域，在資料庫各處一貫地記載直欄內容。直欄的名稱是定義在表格中；網域是用來定義直欄的 *TypeExpression*。

[14]以表格爲中心的視圖可讓您只靠一個視圖就可以更瞭解資料庫設計/作業。以預存程序爲中心的視圖可簡化尋找和變更/維護預存程序的工作。

[15]有些人不瞭解如何在實體資料模型中使用主題區套件，因爲它需要額外的維護工作，來維護邏輯和實體資料庫主題區套件。實體資料模型中的主題區是爲了與邏輯資料模型（如果有使用它的話）一致，這對於「大型」實體資料模型以及沒有邏輯資料模型的情況更是如此。在這種情況下，主題區套件可用來管理從「類別到表格」轉換而產生的表格。

[16]通常資料庫會進行反向工程一次，然後使用 XDE 的 Compare 和 Sync 功能，使所有未來的更新同步。

[17]在 XDE 內，會擷取資料庫擁有者作爲 <<database>> 元件的一項內容。在作爲連線字串的「位置」內容內，有一個綱目屬性。對資料庫進行反向工程時，這通常是指資料庫擁有者。

網域模型的建議結構的一個範例[18] 顯示在圖 14

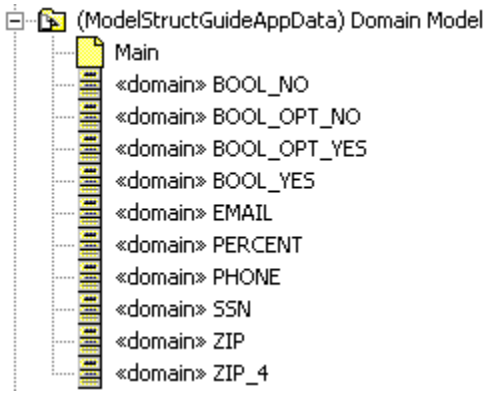

圖 16：網域模型　　結構

這個範例示範在「SQL Server 網域」套件內組織的 SQL Server 網域值。當資料庫設計師定義大量網域時，資料庫設計師可能需要使用「SQL Server 網域」套件下的套件來組織網域。

9. 實作模型

RUP **實作模型**由多個 XDE 模型來代表 –「整體實作模型」和多個 XDE 來回轉換模型 [19]，使用多個來回轉換模型可以用最有效的方式代表不同的實作考量，例如描述 Java 套裝結構 vs. 虛擬目錄結構。同時，可發揮個別來回轉換模型中可用的自動化（亦即，XDE 允許自動化建立實作技術特定模型元素，並利用來回轉換工程，讓這些模型元素與程式碼同步化）。

「整體實作模型」說明整體應用程式的實作，並包含跨越多個來回轉換模型的元素。它包含的圖表參照來回轉換設計元素中的元素，且通常本身不「擁有」任何模型元素。「整體實作模型」不參與任何 XDE 來回轉換工程。

維護「整體實作模型」是選用性的；然而，它對於說明實作的整體結構很有幫助，包括整合單元（在 RUP 中是定義爲**實作子系統**）在內，且對於記載架構的實作視圖也很有幫助。 **實作子系統**在 實作子系統這一節有更詳細的討論

XDE 來回轉換模型的數目視已定義的 XDE 專案和模型組織而定（如需詳細資訊，請參閱XDE 專案結構這一節）。然而，有一個選項是要爲每一個**實作子系統**定義個別專案（和來回轉換模型）。如需如何在 XDE 中代表**實作子系統**的相關資訊，請參閱實作子系統這一節。另外，如前所述，如果您有單一目標實作語言，且團隊規模很小，您可以選擇使用單一 XDE 來回轉換模型來同時代表 RUP **設計模型**和**實作模型**。

「整體實作模型」的一個範例顯示在圖 17

圖 17：整體實作模型結構

如圖 17所顯示，「整體實作模型」只包含圖表。代表架構視圖的圖表，其圖表名稱中有包含　iew。如需

[18]在 XDE 內，支援數個供應商資料庫，包括 DB2、Oracle、Sybase 和 SQL Server。建立網域 XDE 資料模型時，資料庫設計師將選取適當的供應商資料庫來建立網域 XDE 資料模型。 XDE 將建立已選取的資料庫供應商的預設網域清單。

[19] XDE 來回轉換模型是混合式模型。它們是用來代表 RUP **設計模型**和 RUP **實作模型**。 XDE 來回轉換模型中的模型元素代表 RUP **設計類別**（直接對映到實體類別的類別被視爲 **設計類別**）和實體實作檔案。 XDE 來回轉換模型的結構代表實體目錄結構。

架構視圖的詳細資訊，請參閱表 RUP。

「實作視圖：可部署的實作元素」圖表參照將部署到 XDE 部署模型之節點的保存檔。保存檔本身實際上已內含在部署模型中。如果需要詳細資訊，請參閱部署模型這一節。

「實作視圖：實作子系統」圖表參照應用程式的實作子系統。可在圖表的**實作子系統**之間描繪其相依關係。這些相依關係代表**實作子系統**的匯入項目，並決定應該整合**實作子系統**的順序。如需如何在 XDE 中代表實作子系統的相關資訊，請參閱實作子系統這一節。

「實作視圖：實作模型結構」圖表包含對所有 XDE 模型的參照，這些 XDE 模型是用來代表**實作模型**及其關係。
附註：當個別專案針對每一個**實作子系統**而定義之後，「實作視圖：實作模型結構」圖表的內容對於「實作視圖：實作子系統」圖表而言是冗餘的，可以省略。如需實作子系統的相關資訊，請參閱實作子系統這一節。

9.1 實作子系統

RUP **實作子系統**是整合的單元[20] 就其本身而論，它們與 J2EE 模型保持一致（**實作子系統** 可套裝及部署在 J2EE 模組中）。由於設定 XDE 專案結構的一個可能方式是定義每一個 J2EE 保存檔的 XDE 專案，因此，所識別的 **實作子系統**可用來驅動哪些 XDE 專案應定義爲支援詳細的設計和實作。尤其，RUP **實作子系統**可在 XDE 中代表爲 XDE 專案。這是用於這些準則的方法，其中**實作子系統**是使用 XDE 專案來代表。

這些準則中的範例的**實作子系統**如下：

- 在「整體設計模型」中的每一個設計子系統的實作子系統。例如：使用者帳戶管理程式和拍賣管理程式。
- 拍賣管理呈現元素的實作子系統
- 使用者帳號管理呈現元素的實作子系統

[20] 識別**實作子系統**的準則並不在本文件的範圍之內。如果需要詳細資訊，請參閱 RUP。

相關聯的 XDE 專案和模型顯示在圖 18

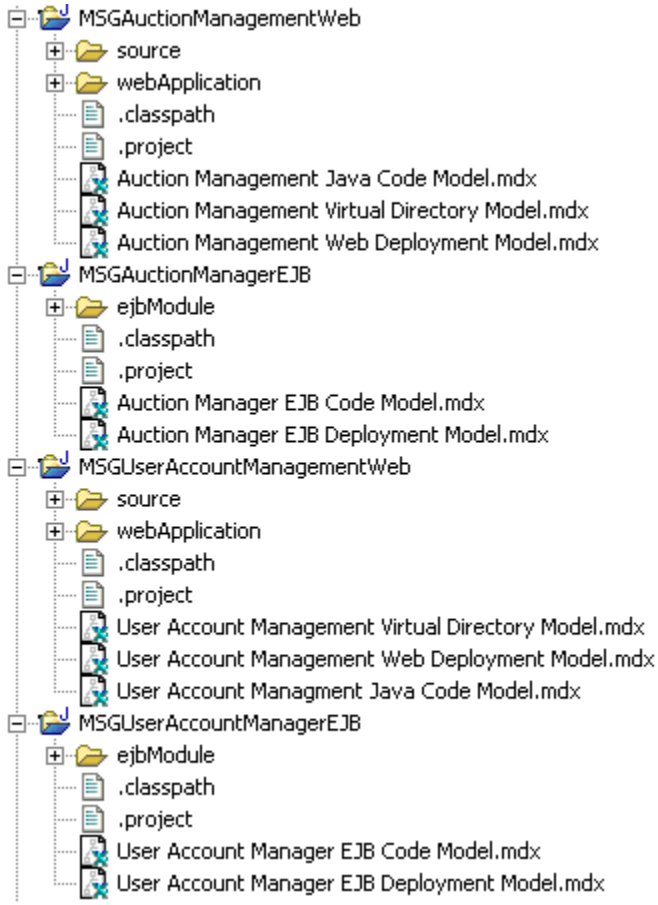

圖 18：作爲 XDE 專案的實作子系統

定義多個 Web 和 EJB 專案時，強烈建議專案名稱和內含模型檔必須是唯一的。這樣可以更容易解譯模型中的圖表，以及解析交互模型參照。在圖 18 所顯示的範例中，**實作子系統**的名稱使用於專案的名稱中，以及每一個內含模型的名稱中。

另外，如果專案規模很小，RUP **實作子系統**可在 XDE 中代表爲 XDE 模型中的套件。圖 19 提供此案例的範例，其中每一個實作子系統是使用 XDE 套件來代表（auctionmanager 和useraccountmanager 套件）。

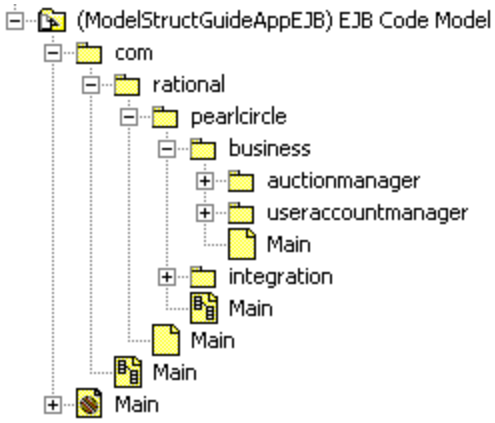

圖 19：作爲套件的實作子系統

9.2 XDE 來回轉換模型

本節提供下列 XDE 來回轉換模型的範例。

- （EJB 專案）EJB 程式碼模型（請參閱 EJB 專案：EJB 程式碼模型這一節）
- （Web 專案）Java 程式碼模型（請參閱 Web 專案：Java 程式碼模型這一節）
- （Web 專案）虛擬目錄模型（請參閱 Web 專案：虛擬目錄模型這一節）

一般而言，在建構 XDE 來回轉換模型時，建議該結構與邏輯結構儘量保持一致，如「整體設計模型」所述（如設計模型這一節所述），當然，實作限制要列入考慮。儘量使**設計模型**和 **實作模型**的結構保持一致，使兩者之間的可追蹤性變成隱含的，而且在維護上更加直接而明確。這點很重要，因爲**設計模型**和**實作模型**之間的對映需要加以維護和管理（這是維護和管理架構的一部分）。

設計類別是 XDE 來回轉換模型的一部分，可透過下列其中一項來建立

- XDE 自動化，用來建立實作技術相依元素，如 EJB、Servlet 等等，作法是從頭建立元素或從技術獨立元素（如**分析類別**）轉換元素。
- 現有的實作的 XDE 反向工程
- 手動建立。

9.2.1 EJB 專案：EJB 程式碼模型

EJB 程式碼模型包含需要實作 EJB 的 Java 資源（例如，實作 Bean 類別、Home 介面類別、遠端介面類別等等）。

EJB 程式碼模型的 Source Root 內容應該設定爲要放置程式碼的目錄。例如，EJB 程式碼模型的 Source Root 內容可設定爲 EJB 專案的 ejbModule 子目錄[21]

[21]如果您使用 XDE 建立專案精靈來建立專案，則會自動建立專案子目錄，並自動設定程式碼模型的 Source Root 內容。

EJB 程式碼模型的一個範例顯示在圖 20

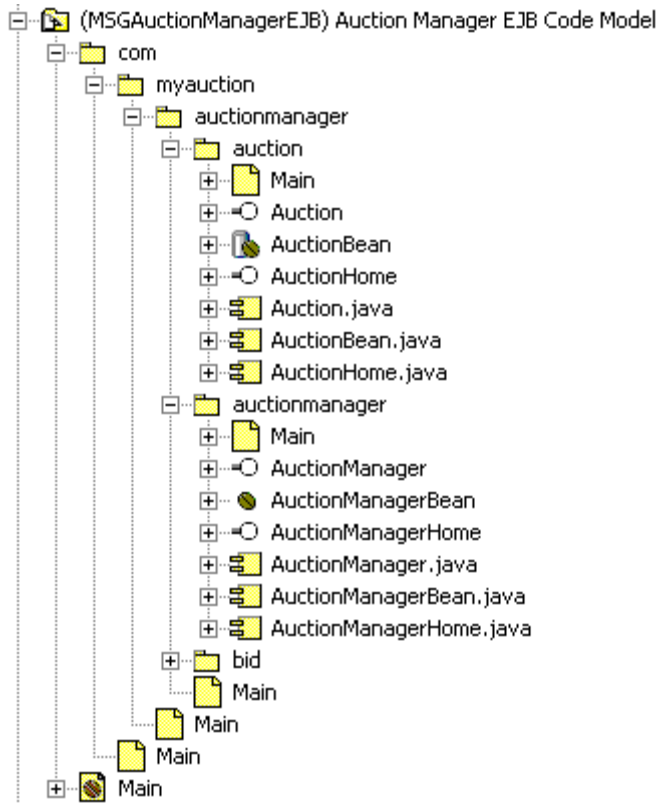

圖 20：EJB 程式碼模型範例

顯示在圖 20 中的 EJB 程式碼模型包含實作拍賣管理程式**設計子系統**的 Java 資源。它是 拍賣管理程式**設計子系統**套件的實作對應項目，該套件屬於「整體設計模型」中的「商業層」套件，在設計層這一節會加以討論。

如圖 20 所示，EJB 程式碼模型結構遵循的慣例是使用網域名稱作爲起始 Java 套件名稱。範例應用程式的網域名稱爲 www.myauction.com。因此，包含實作元素的套件放置在 com 套件內的 myauction 套件中。所以，myauction 套件內的所有 Java 元素會有一個完整名稱，其字首爲 com.myauction。例如，auction 套件的完整名稱爲com.myauction.business.auctionmanager.auction。使用網域名稱作爲起始Java 套件名稱的慣例可保證 Java 類別名稱爲唯一的，即使納入第三方 Java 類別程式庫亦然。

在 myauction 套件內，此結構反映「整體設計模型」的結構（在設計模型這一節會加以討論）。包含拍賣管理程式**設計子系統**（business 套件）的設計層有一個套件，還有一個代表拍賣管理程式 **設計子系統**（auctionmanager 套件）的套件。其他的套件也可以定義在 EJB 程式碼模型中，來收集相關的模型元素（例如，auctionmanager、auction 和 bid 套件）。
附註：由於 Java 程式語言不允許套件名稱中有空白，所以 Java 套件名稱與相關聯的「整體設計模型」套件的名稱可能不同。

如圖 20 所顯示，EJB 程式碼模型不只包含程式碼檔案（.java 元素）的視覺化表示法，也包含指定那些實作元素的類別。這些類別代表已逐步形成並發展到可以實作的 RUP **設計類別**，以 XDE 而言，還可以進行來回轉換工程。附註：XDE 提供型樣，來自動建立用來指定 EJB 的所有類別。

9.2.2 Web 專案：Java 程式碼模型

Web 專案 Java 程式碼模型包含 Java Web 資源（例如，JavaBeans、Servlet、Helper 類別等等）。

Web 專案的 Java 程式碼模型的 Source Root 內容應該設定爲要放置程式碼的目錄。例如，Web 專案的

Java 程式碼模型的 Source Root 內容可設定爲 Web 專案的 Java Source 子目錄[22]

Web 專案 Java 程式碼模型的一個範例顯示在圖 21。

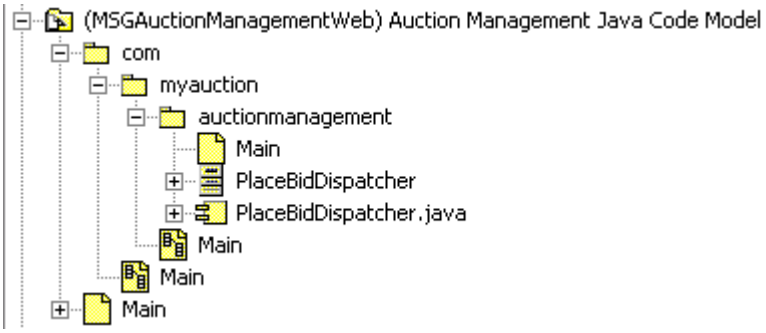

圖 21：Web 專案 Java 程式碼範例

顯示在圖 21 中的 Java 程式碼模型包含實作拍賣管理呈現設計元素的 Java 資源。它是「拍賣管理」套件的 Java 實作對應項目，該套件屬於「整體設計模型」中的「呈現層」套件，在設計層這一節會加以討論。

如圖 21 所示，Web 專案 Java 程式碼模型結構遵循的慣例是使用網域名稱作爲起始 Java 套件名稱。範例應用程式的網域名稱爲 www.myauction.com。因此，包含實作元素的套件放置在 com 套件內的 myauction 套件中。所以，myauction 套件內的所有 Java 元素會有一個完整名稱，其字首爲com.myauction。例如，auctionmanagement 套件的完整名稱爲 com.myauct\on.presentation.auctionmanagement。使用網域名稱作爲起始 Java 套件名稱的慣例可保證 Java 類別名稱爲唯一的，即使納入第三方 Java 類別程式庫亦然。

在 myauction 套件內，此結構反映「整體設計模型」的結構（在設計模型這一節會加以討論）。包含拍賣管理呈現元素（presentation 套件）的設計層有一個套件。還有一個代表拍賣管理呈現元素（auctionmanagement 套件）的套件。
附註：由於 Java 程式語言不允許套件名稱中有空白，所以 Java 套件名稱與相關聯的「整體設計模型」套件的名稱可能不同。

如圖 21 所顯示，Web 專案 Java 程式碼模型不只包含程式碼檔案（.java 元素）的視覺化表示法，也包含指定那些實作元素的類別。這些類別代表已逐步形成並發展到可以實作的 RUP **設計類別**，以 XDE 而言，還可以進行來回轉換工程。

9.2.3 Web 專案：虛擬目錄模型

虛擬目錄模型包含非 Java 程式碼 Web 資源（例如 JSP 和 HTML 網頁）。每一個 XDE Web 專案可以有多個虛擬目錄模型。多個虛擬目錄模型支援有多個虛擬目錄的 J2EE 應用程式的開發。在這種應用程式中，產生的網站實際上是分割成沒有共同的根目錄。例如，在線上零售商店中，www.mystore.com 可用於型錄購物，而 order.mystore.com 可用於監督訂購狀態。

虛擬目錄模型的 Source Root 內容應該設定爲要部署到 Web 儲存器的 Web 資源的存放目錄。例如，虛擬目錄模型的 Source Root 內容可設定爲 Web 專案的 Web Content 子目錄[23]

[22]如果您使用 XDE 建立專案精靈來建立專案，則會自動建立專案子目錄，並自動設定程式碼模型的 Source Root 內容。

[23]如果您使用 XDE 建立專案精靈來建立專案，則會自動建立專案子目錄，並自動設定程式碼模型的 Source Root 內容。

虛擬目錄模型的一個範例顯示在圖 22

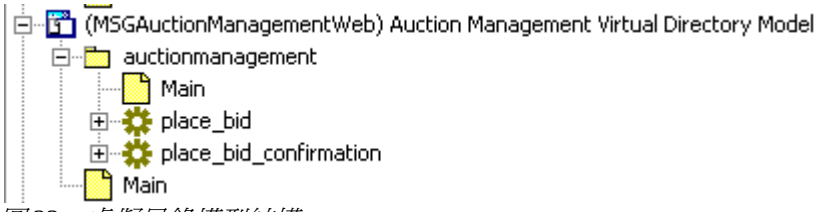

圖 22：虛擬目錄模型結構

顯示在圖 22 中的虛擬目錄模型包含實作拍賣管理呈現設計元素的非 Java 程式碼資源。它是「拍賣管理」套件的非 Java 實作對應項目，該套件屬於「整體設計模型」中的「呈現層」套件，在設計層這一節會加以討論。

在這個範例中，虛擬目錄模型的結構反映「整體設計模型」的結構（在設計層這一節會加以討論）。「整體設計模型」中的每一個套件有一個套件，其內容將由非 Java 元素所實作，這些元素將部署到 Web 儲存器，例如 JSP 和 HTML 網頁。由於 Web 伺服器存取的目錄名稱有所限制，因此，在虛擬目錄下的目錄名稱不一定與「整體設計模型」中的那些目錄名稱相同。

如圖 22 所顯示，虛擬目錄模型包含指定實作元素的類別的視覺化表示法。這些類別代表已逐步形成並發展到可以實作的 RUP **設計類別**，以 XDE 而言，還可以進行來回轉換工程。和 EJB 程式碼模型與 Web 專案 Java 程式碼模型不同，XDE 不會在虛擬目錄模型中產生相關聯的程式碼檔案的視覺化表示法。相關聯的程式碼檔案的名稱是以類別的內容來維護。

10. 部署模型

RUP **部署模型**是由多個 XDE 模型來代表–「EAR 部署模型」和一組個別 XDE 部署模型，包含要部署的元素的每一個 XDE 專案各一個。使用多個部署模型可發揮 XDE 的部署自動化（XDE 爲 J2EE 保存檔和部署描述子的建立和修正，以及爲這些模型元素與其程式碼套裝在保存檔中的模型元素的同步化，提供自動化功能）。

本節提供下列部署模型的範例：

- （應用程式專案）EAR 部署模型（請參閱 EAR 部署模型這一節）
- （EJB 專案）EJB 部署模型（請參閱 EJB 部署模型這一節）
- （Web 專案）Web 部署模型（請參閱 Web 部署模型這一節）

以下是一些關於 XDE 部署值得注意的一般項目：

- 部署描述子在 XDE 部署模型中不明確地以檔案代表（UML 構件）。而是以 XDE 部署模型的內容來代表。XDE 使用部署模型所包含的元素，以及指派給那些元素的內容值，來決定寫入到模型的部署描述子檔案中的內含資訊。
- XDE 對所有部署使用 EAR 部署模型，即使您只部署單一 EJB-JAR 或 WAR 也一樣。這反映大部分應用程式伺服器的限制，它們需要所有部署都有 EAR。因此，即使您只爲 EJB-JAR 或 WAR 的部署建模，仍然要使用 EAR 部署模型來部署到 XDE 所支援的應用程式伺服器。然而，XDE 可以*匯出*任何保存檔到檔案系統。您可以利用此功能來部署到 XDE 不支援的應用程式伺服器。在匯出保存檔之後，您可以呼叫伺服器特定工具來完成部署。

- 不同 J2EE 保存檔可定義給不同部署環境（測試、正式作業等等）。這些保存檔可定義在相同的 XDE 部署模型或個別的 XDE 部署模型中。XDE 自動化支援兩者，但它有定義專案的預設部署模型和每一個部署模型的預設保存檔，不過您可以調整這些定義，因爲它們已在 XDE 部署模型中建模。不論您選擇哪一種方法，都應該在 EJB 專案中定義 EJB 部署模型，以及在 Web 專案中定義 Web 部署模型 [24]

10.1 EAR 部署模型

「EAR 部署模型」包含 J2EE 應用程式保存檔（.EAR 檔）的視覺化表示法、EAR 部署描述子資訊和要部署 EAR 的節點。「EAR 部署模型」也包含圖表來顯示 EAR 內含哪些 J2EE 模組（來自其他部署模型）。

「EAR 部署模型」也可用來說明整體應用程式的部署配置，以及架構的「部署視圖」。「EAR 部署模型」可包含圖表來顯示所有部署節點及其連接，以及哪些保存檔要部署到哪些節點。這類圖表可參照來自所有個別部署模型的元素（節點和保存檔）。

「EAR 部署模型」的一個範例顯示在圖 23

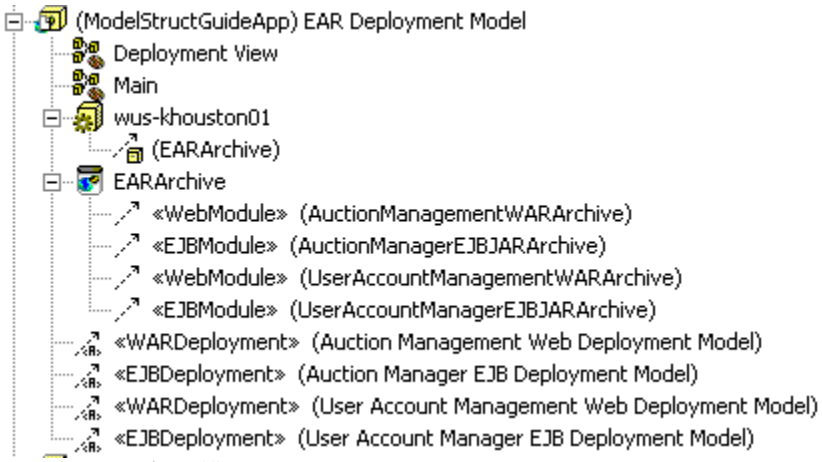

圖 23：EAR 部署模型

「部署視圖」圖表代表架構的部署視圖。它包括部署節點、其交互連接以及哪些 J2EE 保存檔要部署到這些節點。在某些情況下，此圖表只包含 J2EE 應用程式保存檔和它要部署到的應用程式伺服器節點。然而，在獨立式 J2EE 模組保存檔部署到特定節點的情況下，此圖表應顯示哪些保存檔部署到哪些節點。如需架構視圖的相關資訊，請參閱 RUP。

10.2 EJB 部署模型

EJB 部署模型包含 EJB 元件、EJB-JAR 構件和 EJB 部署描述子資訊。「EAR 部署模型」也包含圖表來顯示 EJB-JAR 內含哪些實作元素（來自 EJB 程式碼模型）。確切地說，在 EJB 部署模型中，EJB 元件是用來代表實作元素，而且它是對映到 EJB-JAR 的 EJB 元件。

[24] EJB 部署模型*必須*在 EJB 專案中，但它與相對應的 EJB 程式碼模型不一定是*相同*專案。事實上，您可以將不同程式碼模型的 EJB 加以「混合及比對」，成爲一個部署模型。相同的意見也適用於 Web 模型。

EJB 部署模型的一個範例顯示在圖 24

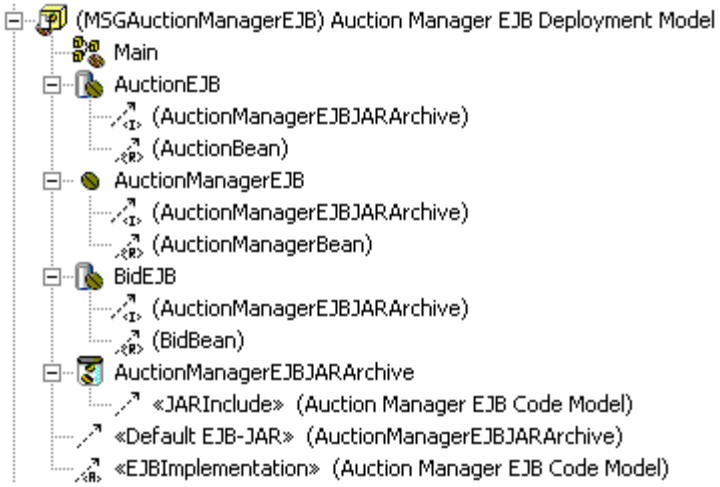

圖 24：EJB 部署模型範例

10.3 Web 部署模型

Web 部署模型包含 Web 元件、WAR 保存檔和 Web 部署描述子資訊。「Web 部署模型」也包含圖表來顯示 WAR 內含哪些實作元素（來自 Web 專案來回轉換模型）。確切地說，在 Web 部署模型中，Web 元件是用來代表實作元素，而且它是對映到 WAR 的 Web 元件。

Web 部署模型的一個範例顯示在圖 25

圖 25：Web 部署模型

Rational
the software development company

兩個總公司：
Rational Software
18880 Homestead Road
Cupertino, CA 95014
電話：(408) 863-9900

Rational Software
20 Maguire Road
Lexington, MA 02421
電話：(781) 676-2400

免付費專線：(800) 728-1212
電子郵件：info@rational.com
網址：www.rational.com
國際辦事處：www.rational.com/worldwide