

## 企業資料整合新秀 ODI – Oracle Data Integrator

### Oracle Fusion Middleware 的最後一塊拼圖

眼尖的讀者這一兩年可能會在 Oracle 規劃的 Fusion Middleware 架構中發現到一個不尋常的地方，就是原本用來擔任 ETL 工作的 Data Warehouse Builder 被另一個新產品：Data Integrator 給取代了。

眾所周知的，Oracle 這幾年爲了擴張並鞏固資料庫及其相關領域的版圖，不斷的收購其他知名的品牌，例如 Sun、BEA 等等。其中有一家叫 Sunopsis 的，就是今天 Oracle Data Integrator 的前身。這個軟體以其創新的架構與作法，顛覆了人們對傳統 ETL 工具的認知，發揮出了傳統工具所無法達到的效能與彈性，因此當大家對 Fusion Middleware 中原本的 Data Warehouse Builder 效能有所疑慮的時候，此產品適時的補足了整個 Fusion Middleware 的藍圖，成爲最後一塊重要的拼圖。

### 傳統 ETL 工具的缺點

從最早的 Hand Coding，到後來陸續開發出來的所謂 ETL 專屬引擎，這些傳統的 ETL 工具有著一些共同的弱點，現分述如下：

#### 相容性差

這裡指的是用甲工具做出來的 ETL 程式，很難、或甚至不能與乙工具做出來的 ETL 程式結合執行，導致每引進一次新的 ETL Engine，就有不少程式要改寫。

#### 效能黑箱

通常透過 ETL 專屬引擎自己會針對資料拋轉的過程進行一些最佳化的調校，但是由於它本身是一個已經定型的工具，對使用者來說是一個黑箱作業，萬一日後又遇到其它的效能瓶頸時，根本不知道該如何進行 Tuning 的工作。而通常得到的建議則是硬體升級，如此又將導致額外的投資。

### ODI 的革新

#### 一個 ETL 程式的整合管理平台

Oracle Data Integrator 提供的不只是一個 ETL 引擎。考慮到資料拋轉可能遇到的種種狀況，例如各種異質的資料源、舊系統遺留下來的 Hand Coding ETL 程式、或雨會有追除錯的需求等，ODI 提供了一個前所未見的 ETL 程式專用的整合管理平台。在此平台內，ODI 提供了：

#### 1.環境設定功能：

讓使用者定義出何者為開發測試環境、何者為正式生產環境，滿足一般『開發』、『測試』、與『上線』流程的需求。

#### 2.異質資料源設定功能：

使用者可隨需求自行定義各種異質資料源，原則上只要能取得該資料源所提供的 driver(jdbc/odbc)，ODI 都可以順利的連接到該資料源。

#### 3.元資料(Metadata)擷取功能：

使用者可因此直接套用 Metadata，而無須再對資料表定義其相關欄位與資料型別。

#### 4.知識模組(Knowledge Module/KM)：

ODI 隨安裝內建了一系列常用的知識模組，例如用以連結資料源的模組(LKM)、用以進行資料整合的模組(IKM)、用以檢核資料一致性的模組(CKM)等等，幾乎網羅了所有主流 RDBMS 及 Application，使用者因此不必再自行撰寫所謂的轉檔程式。

#### 5.整合其他 Procedure 的功能：

若存在有舊系統留下來的 ETL 程式，或因某些特殊需求而必須於現有 KM 外自行撰寫一些 OS Script、Java Script、或 SQL Script 等程式，都可以透過 ODI 平台進行整合。

#### 6.流程控制與追蹤除錯的功能：

就像 Visual Studio 一樣，ODI 是個所謂的 IDE 整合環境，除了可以在裡面設計 ETL 程式外，還可進行流程控制，並結合追蹤除錯。這是其他單純的 ETL Engine 所無法做到的。

7.負載平衡的功能：當單一一個 ODI Agent(或 Server)已無法應付大量的 ETL 工作時，可隨時增加 Agent，一來可分擔 Loading，二來還可以和原來的 Agent 串聯，形成所謂的 Load Balance。

### 一個開放式的架構

ODI 的開放性表現在下面兩個地方：

1.ETL 程式碼公開：透過 ODI 的 Log，使用者可以很清楚的看到每一個步驟所實際執行的指令，不論是 SQL Statement 還是 OS Command，都可以被完整的追蹤到。如此不但除錯容易，當效能出現問題時，也方便管理員透過對原始指令的檢視來進行必要的語法 Tuning。

2.ODI 程式本身原始碼公開：除了核心 Java Class 較難瀏覽其內容外，其他包含在 ODI 套裝軟體內的 Batch 或 Script 檔都可以被直接閱讀，使用者因而得以了解這些程式會進行哪些動作。不但避免了傳統 ETL 工具黑箱作業的弊病，也在除錯及效能調校上非常有幫助。

## 結語

長久以來，企業一直就有相當高的 ETL 需求。自行 Coding 雖然可以節省不少軟硬體的成本，但卻會相對耗費大量的人力與時間；若透過 Third Party 廠商開發的專屬 ETL 引擎，雖可節省時間與人力，但卻必須投資相關的軟體甚至硬體設備。若這些投資能在未來證實有效益自然沒問題，但現實狀況卻多半是隨著資料量或要拋轉的資料表增加，原有的引擎逐漸會出現效能瓶頸，而廠商絕大多數的回應是設備升級，造成重複投資。

ODI 有鑑於此，採取了開放式的架構，使得 ETL 工作不再是黑箱，因此進一步的效能調整成為可能；再加上為數眾多的現成 KM、搭配可自行撰寫 Procedure 的功能，使得它也保有如同自行 Coding 一樣的彈性。最後再搭配上一個完整的操作管理平台，成為所謂的整合是 IDE 介面，更是使得這個產品達到更高的完整度，任何與 ETL 相關的動作都可以透過這個平台實施。難怪會被 Oracle 當作補足 Middleware Solution 最重要的一塊拼圖。