Gestern OWB, heute ODI

Christian Piasecki
Enpit consulting OHG
Stadtlanfert 7, 33106 Paderborn

Schlüsselworte:

Oracle Warehouse Builder, Oracle Data Integrator, Erfahrungsbericht

Einleitung

Nach dem letzten Statement of Direction von Oracle zum Thema Oracle Data Integrator und Oracle Warehouse Builder und der damit verbunden Abkündigung des OWB ist es nun wirklich Zeit sich mit dem ODI auseinander zu setzen.
Anhand eines Projekts bei dem die Zeiterfassungsdaten von verschiedenen Projektmitarbeitern mit Hilfe des ODI in einer Oracle Datenbank konsolidiert werden soll, wird aufgezeigt, wo die Unterschiede und Gemeinsamkeiten für OWB-Entwickler liegen.

Aufgabenstellung

Folgende Aufgabenstellung gilt es umzusetzen:

- Projektmitarbeiter erfassen ihre Tätigkeiten mit verschiedenen Systemen z.B. Mobile-Endgeräten
- Datenlieferung erfolgen immer Monatsweise mindestens einmal am Anfang des Folgemonats
- Aufwände sollen zentral in einer Datenbank konsolidiert werden
- Es wird nicht ein System zur Erfassung vorgeschrieben, sondern Datenanlieferungen werden integriert

Aus dieser Aufgabenstellung lässt sich folgendes Architekturbild ableiten:

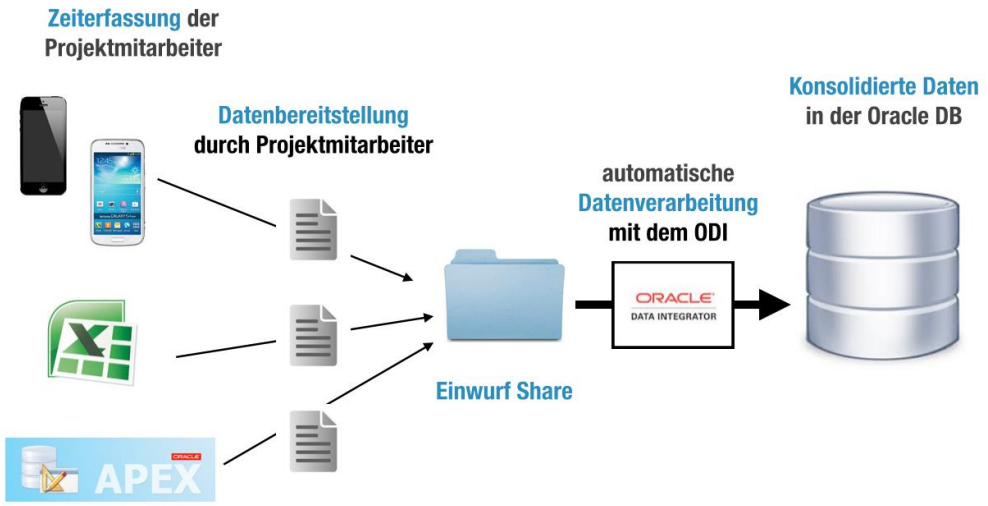

Abb. 1: Schaubild / Architektur

Erste Schritte / Einlesen einer CSV-Datei

Um zu eruieren, ob die Aufgabenstellung mit dem ODI unter vertretbarem Aufwand umsetzbar ist, wird zu Anfang absichtlich ein einfacher UseCase gewählt, bei dem eine CSV-Datei in eine Oracle Tabelle eingelesen werden soll.

Bei der Umsetzung mit dem OWB wäre folgendes zu tun:

- Anlegen eines Moduls mit Speicherort, das auf ein Datenbank-Verzeichnis zeigt
- Anlegen eines Moduls mit Speicherort, das als Target-Schema in der Datenbank dient
- Anlegen eines File-Objekts für die Text-Datei
- Erstellen eines Mappings mit dem File-Objekt als Quelle und einer Tabelle als Ziel-Objekt
- optional erstellen und verwenden einer externen Tabelle als Quelle
- Mapping deployen
- Mapping ausführen

In unserer Umsetzung mit dem ODI müssen folgende Arbeitsschritte erledigt werden:

- Quelle Definieren
- logisches Schema erstellen
- Model erzeugen
- Datenspeicher und Format definieren
- Reverse Engineering der Datenstruktur
- Ziel-Schema Definieren
- Import der Zieldatenstruktur
- Mapping erstellen
- Knowledge Module zuordnen
- Mapping ausführen

Betrachtet man die einzelnen Schritte zusammenhängend ergibt sich am Ende folgendes Bild und man erkennt dass der Ablauf in beiden Tools fast identisch ist:

Abb. 2: Gegenüberstellung OWB/ODI – Einlesen einer CSV-Datei

Will man eine Bewertung des ODI vornehmen, könnte man folgendes sagen:

- gleicher Aufwand beim erstmaligen Erstellen von Speicherorten in beiden Tools
- Beide Tools unterstützen einen beim Einlesen der Datei-Metadaten
- Mapping Designer in beiden Tools sehr ähnlich
- Wann wird welches Integration/Loading Knowledge Modul genommen und warum?
- Anlegen von Zieltabellen im ODI nicht sehr komfortabel
    - Lassen sich nur aus dem Mapping mit Create Table erstellen (gibt beim 2ten Durchlauf eine Warnmeldung, falls nichts aus)
    - Partionierung, Indexe in unterschiedlichen Tablespaces
    - ➔ Objekte lieber selber direkt in der DB anlegen

→ OWB auf Oracle Datenbank zugeschnitten, ODI kommt aus der Middleware und ist heterogener aufgestellt

Workflow Umsetzung mit dem ODI

Nachdem der einfache UseCase mit dem ODI erfolgreich umgesetzt wurde, kann der ganze Projektworkflow mit dem ODI in Angriff genommen werden. Dabei sind folgende Punkte zu beachten:

- Dateien werden in einem zentralen Ordner als txt-/csv-Datei (gleicher Aufbau) abgelegt
- Ein Prozess überwacht diesen Ordner und verarbeitet die abgelegten Dateien automatisch
- Löschen von Daten die für die Ressource und Monat schon da sind
- Daten laden
- Datei verschieben

Setzt man diese Aufgabenstellung mit dem ODI um, sieht die Package Implementierung so aus:

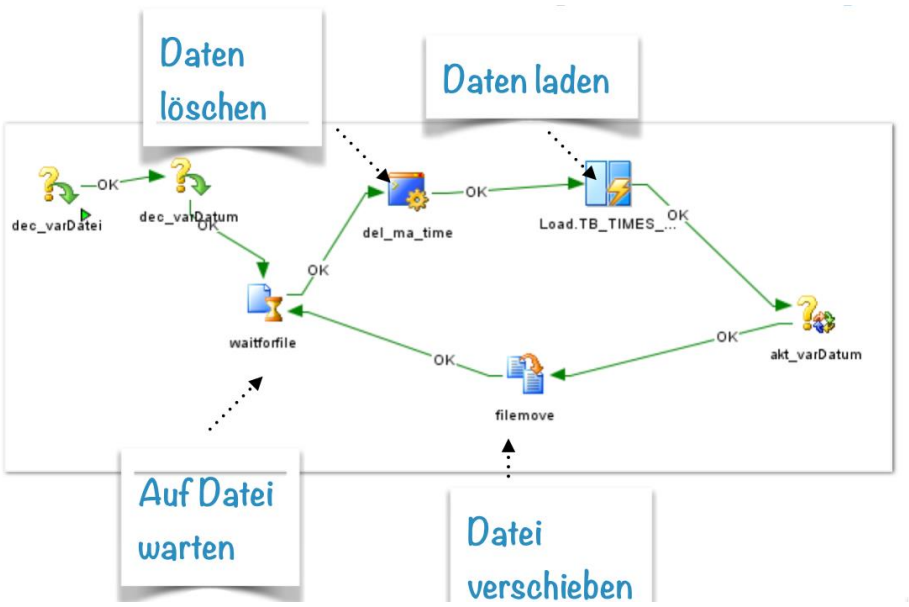

Abb. 3: ODI Package

Unterschiede

Wo liegen nun die Unterschiede, die einem OWB-Entwickler auffallen, die im ODI anders gelöst werden müssen:

DB-Funktionen/Prozeduren

- keine Möglichkeit diese zu importieren und direkt einzubinden
- 2 Möglichkeiten:
    1. In einer Expression aufrufen
    2. ODI-Prozedur als Wrapper erstellen

➔ Nachteil: Kein Import der Objekte und Unterstützung in der IDE

Pre-/Post-Mapping

- Im ODI gibt es kein Pre-/Post-Mapping Operatoren
    - Im ODI muss diese Funktion mit anderen Mitteln umgesetzt werden
    - Die einfachste Möglichkeit ist, die Prozeduren in einer Package vorher aufzurufen

➔ Testen der Verarbeitung muss über eine Package durchgeführt werden

Workflow - Packages / Load Plans

- Im OWB wird die Ausführung mehrerer Mappings durch Workflows gesteuert (seriell, parallel)
- Im ODI gibt es Packages und Load Plans
- Unterscheiden sich z.B. in der Transaktionssteuerung und bieten nicht die gleichen Funktion (z.B. Load Plans bei paralleler Ausführung verwenden, Packages bei Schleifen)

➔ Einarbeitung nötig (es gibt auch keine Migration von Workflows)

Mehrere Tabellen Befüllen

- Es ist wie im OWB möglich mehrere Tabellen in einem zu befüllen
    - Ein Splitt-Operator steht zudem zur Verfügung
    - Allerdings gibt es keine Target-Load-Order um die Tabellen in einer bestimmten Reihenfolge zu befüllen

➔ Aufspaltung in mehrere Mappings nötig

ODI – Szenarien

- ODI-Objekte werden als Szenarien bereitgestellt
    - Szenarien können von unterschiedlichen Objekten erstellt werden (Mappings, Prozeduren, Packages)
    - Der Code des Objekts wird „eingefroren“ und bereitgestellt (Entwicklung -> Produktion)
    - Bei Änderungen in den Objekten, muss ein neues Szenario erstellt und wiederum bereitgestellt werden

Fazit / Bewertung

Will man die Produktivität der Umsetzung mit dem ODI bewerten und ein Fazit ziehen, würde das im Moment folgendermaßen aussehen:

- Durch Flowbased Mappings und Packages ist die Konzeption mit dem OWB vergleichbar
- Produktivität leidet im Moment, da
    - DB-Objekte (Packages, Prozeduren, Funktionen) extra gewrappt werden müssen
    - Testen über Packages und nicht Mappings erfolgen muss
- Nützliche Komponenten in den ODI-Packages (filewait, Variablen deklarieren, erhöhen)
- ODI ist kein Hexenwerk, allerdings muss man sich wie mit jeder neuen Technologie intensiv damit auseinander setzen
- In heterogenen Projekten hat der ODI seine Vorteile
- In reinen Oracle-Umgebungen ist der OWB besser geeignet

➔ Selber loslegen und eigene Erfahrungen sammeln!

Kontaktadresse:

Christian Piasecki
enpit consulting OHG
Stadtlanfert 7
D-33106 Paderborn

Telefon: +49 5251 20202 84
Fax: +49 (0) 12-345 6788
E-Mail christian.piasecki@enpit.de
Internet: www.enpit.de