Vergleich von OWB Repositories mittels Scripting

Oliver Gehlert
Metafinanz-Informationssysteme GmbH
München

Schlüsselworte:

Oracle Warehouse Builder, OMB, Datawarehouse, Migration

Einleitung

Der Oracle Warehousebuilder ist ein mächtiges Werkzeug zur Modellierung von ETL-Prozessen, bei dem alle Metainformationen in einem Repository abgelegt sind. Auf diese Metadaten kann man nicht nur über die OWB-Oberfläche zugreifen, sondern auch mit OMB*Plus bzw. der Java-API.
In diesem Vortrag werden wir uns auf den Zugriff mittels OMB*Plus konzentrieren. Informationeen zum Zugriff über die Java-API finden Sie im Vortrag „Einführung in die Java API“ meines Kollegen Carsten Herbe.
Die im Projekt anstehende Migration des Warehousebuilders von Version 10.2. nach 11.2 gab den Anstoß, sich um qualitätssichernde Maßnahmen zu kümmern. Bei der Migration von Release 9.2 nach Release 10.2 wurde das Repository nicht fehlerfrei übernommen. Die beschriebenen Prozeduren sind aber nicht nur auf Migrationen beschränkt, es können hiermit auch unterschiedliche Stages miteinander verglichen werden. Eine weitere Möglichkeit ist ein Vorher/Nachher-Vergleich, um zu validieren, dass nur die geplanten Änderungen implementiert wurden.
Alle Skripte werden nach der DOAG Konferenz zum Download bereitgestellt.

Versionsvergleich innerhalb eines Repositorys

Innerhalb eines Repository kann man Entwicklungsstände über sogenannte Snapshots speichern. Hierbei muss man zwischen kompletten (full) Snapshots bzw. deren Signatur unterscheiden. Diese Snapshots können mehrere OWB Repositoryobjekte, sowie deren Abhängigkeiten enthalten.
Erstellt man vor jeder Mappingänderung einen Snapshot, so kann man anschließend den aktuellen Stand mit dem Snapshot vergleichen. Über die grafische Oberfläche erhält man hier einen sehr detaillierten Report, der die Differenzen im Detail ausgibt. Beim Speichern des Ergebnisses als XML-Dokument kommt es allerdings zu einem erheblichen Informationsverlust. Es wird nur gespeichert, welches Objekt bzw. Attribut sich geändert hat, aber nicht welche Werte dieses Attribut aktuell bzw. vorher hatte.

Der Vergleich eines Snapshots mit dem aktuellen Mapping ist auch per Skript möglich:

```
OMBCOMPARE SNAPSHOT 'MAP_STA_COBA_ANTRAG_100621' WITH CURRENT  FOR MAPPING
'/FIV4/STAGING_DESIGNER/MAP_STA_COBA_ANTRAG'               OUTPUT       TO
'C:/temp/coba_ALLA.xml' WRITE ALL;
```

Das Ergebnis dieses Befehls ergibt eine XML-Datei, die dem gespeichertem Ergebnis des grafischen Vergleichs entspricht. Daher gelten hier die selben Einschränkungen wie oben.

```
<CompareTree                TargetSnapshot="CURRENTCOMPONENTDEFINITION"
SourceSnapshot="DOAG_SNAP_1"       DiffOf="[TargetSnapshot       minus
SourceSnapshot](CURRENTCOMPONENTDEFINITION minus DOAG_SNAP_1)" >
    <MetaModelTree         TargetSnapshot="CURRENTCOMPONENTDEFINITION"
SourceSnapshot="DOAG_SNAP_1"  MetaModelDiffOf="[TargetSnapshot  minus
SourceSnapshot](CURRENTCOMPONENTDEFINITION minus DOAG_SNAP_1)">
    </MetaModelTree>
    <LogicalComponentModelTree
TargetSnapshot="CURRENTCOMPONENTDEFINITION"
SourceSnapshot="DOAG_SNAP_1"   LogicalDiffOf="[TargetSnapshot   minus
SourceSnapshot](CURRENTCOMPONENTDEFINITION minus DOAG_SNAP_1)">
      <OWBObject            InstanceOf="oracle.owb.mapping.BatchMap"
PhysicalName="MAP_DWH_DIM_VGB"          LogicalName="MAP_DWH_DIM_VGB"
DiffState="UNCHANGED"        AggregateState="AGGREGATE        UPDATED"
UOID="8FA919292A7733A2E040A8C04E342420"     TypeDescription="Mapping"
MyRole="Maps" ParentRole="InstalledModule">
        <BoundTo             InstanceOf="oracle.owb.iconset.IconSet"
PhysicalName="" LogicalName="" AssocDiffState="ASSOCIATION UNCHANGED"
AssocUOID="null"           AssocTypeDescription="Icon           Set"
AssocRole="IconObject" ObjectRole="Element" />
        <OWBObject…
```

Bei der OWB Version 9.0.4 wurde noch das passende Stylesheet für die XML-Datei mitgeliefert, so dass man eine lesbar HTML-Ausgabe erstellen konnte. Hat man noch Zugriff auf die Installationsmedien einer OWB Version bis 9.2, so kann man die notwendigen Dateien extrahieren:

- mcmview.bat
- wb_mcmpp.xsl
- xschema.jar
- xmlparserv2.jar

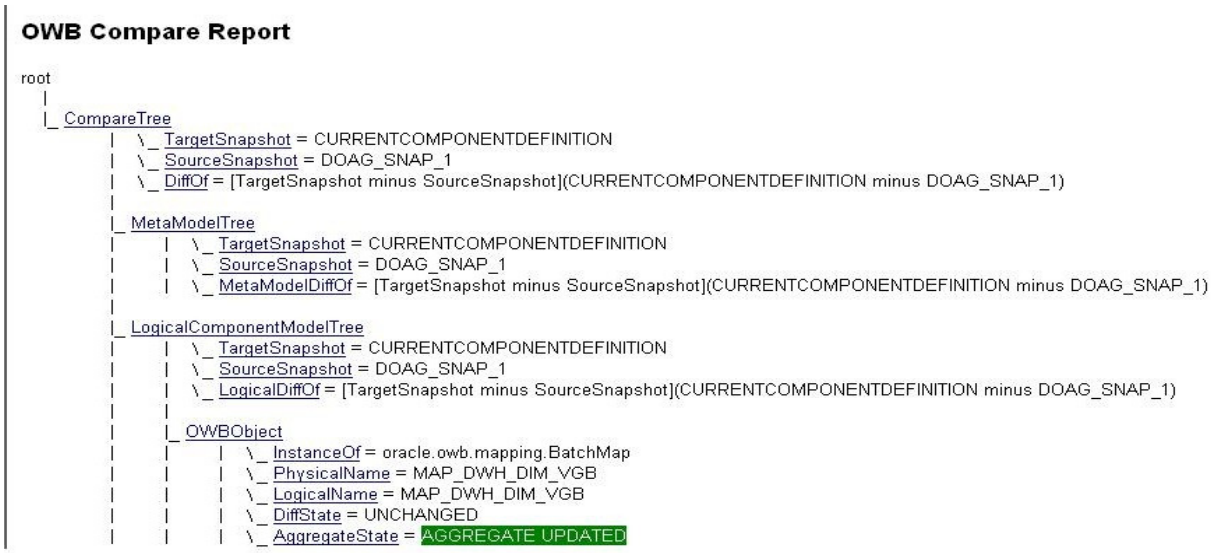

Abbildung 1 Formatierte Ausgabe der XML-Datei

Möchte man Vorher-/Nachher-Werte von Attributen oder Expressions auf einem Block sehen, so ist es effizienter, wenn man die Mappings in „Textform“ im Dateisystem speichert und dann die Textdateien miteinander vergleicht.
Um ein aussagekräftiges Ergebnis zu erzielen, müssen die Textdateien folgende Bestandteile eines Mappings enthalten:

- Operatoren
- Attribute
- Verbindungen
- Expressions
- Hints
- Generation Mode

Wäre es nicht einfacher, den generierten PL/SQL-Code der Mappings zu vergleichen?
Vergleicht man den Code des gesamten Mappings, so ist der Vergleich deutlich aufwändiger, da das Package eine große Anzahl an Debug-Code und Monitoringinformationen enthält, der abhängig vom Client und dessen Ländereinstellungen ist. Dadurch kann der Quellcode, trotz identischer Metadaten im Repository, zahlreiche Unterschiede aufweisen.

Vergleich über Repositorygrenzen hinweg

In unserem Projekt war der Mappingvergleich innerhalb eines Repositorys nur zweitrangig. Wichtiger war für uns der Vergleich zweier Repositorys über Versions- und Plattformunterschiede hinweg.
Alle Informationen aus dem OWB Repository in der Version 10.2 unter Solaris mussten in das neue Repository in der Version 11.2 unter Linux importiert werden. Aufgrund der Erfahrungen aus der vorherigen Migration lag unser Augenmerk insbesondere auf Expressions und Joinbedingungen.
Da bereits der Textdump eines gesamten Repositorys relativ viel Speicherplatz belegt, entschieden wir uns für ein mehrstufiges Vorgehen. Einem Grobvergleich, der nur die Typ und Namen der einzelnen Repository-Objekte vergleicht. Im zweiten Schritt wurden dann Transformationsspezifikationen und Mappingoperatoren aufgenommen.
Die nächsten Detaillierungsstufen umfassten dann die Attribute aller Mappingobjekte und im finalen Schritt auch alle Verbindungen zwischen den Attributen.

Grobvergleich

Für den Grobvergleich müssen folgende Repositoryobjekte in eine Textdatei extrahiert werden:

- Tabellen
- Views
- Sequences
- Transformationen
- Materialized Views
- Mappings

Locations und Benutzer wurden im neuen Repository neu angelegt und daher nicht verglichen.
Bei Mappings, Views und Materialized Views wurden auch die Spaltendefinitionen ausgelesen, bei den Transformationen und Mappings nur die Namen.

Bei der ersten Testmigration kam es bereits nach diesem Schritt zu ersten Abweichungen. Die Transformationen waren nicht komplett importiert worden, obwohl die Endmeldung des Imports

lautete: „Import erfolgreich beendet“. Eine Analyse der Logfiles ergab, dass die Transformationen übersprungen wurden. Die Meldung lautete: „Object not selected for import“. Für einen erfolgreichen Import mussten die Transformationen mit der Option „Import all objects“ importiert werden.

Grobvergleich Teil 2

Nach erfolgreichem Import der Transformationen konnte man einen feineren Vergleich durchführen. Hierbei wurden jetzt die Spezifikationen der Transformationen mit aufgenommen, die Mappingeigenschaften wie Generierungsmodus, Hints und Codeoptionen.
Bemerkenswert bei diesem Vergleich war die Abfrage der Definition von Funktionen und Prozeduren innerhalb von Packages. Hier musste mittels „OMBCC ‚Packagename’“ in das Package gewechselt werden.

Detailvergleich

Der Detailvergleich umfasst neben den obigen Informationen auch alle Attributdefinitionen, Expressions und Joinbedingungen. Dadurch erhöht sich die Datenmenge je Mapping deutlich, aber diese Informationen sind unabdingbar, da es hier bei der letzten Migration zu Fehlern kam.
Das deutlich umfangreichere Skript wird im Rahmen des Vortrags genauer besprochen und steht im Anschluss an die DOAG zum Download zur Verfügung.

Komplettvergleich

Um den Komplettvergleich abzuschließen fehlen jetzt „nur“ noch die Verbindungen zwischen den einzelnen Mappingattributen. Für die Bestimmung der Verknüpfungen gibt es zwei unterschiedliche Varianten, eine elegante, die berücksichtigt, ob sich ein Objekt in einer In-, Out- oder In-/Outgroup befindet und nur ausgehende Verbindungen berücksichtigt. Oder eine einfache Lösung, die für jedes Attribut ein bzw. ausgehende Verbindungen bestimmt. Die einfache Lösung führt dann zur doppelten Ausgabe aller Verbindungen!

Best Practice beim Import eines kompletten Repositories

Bei der Migration von 10.2 nach 11.2 haben sich einige Vorgehensweisen bewährt. Ein ganz wichtiger Punkt hierbei ist die Bereinigung des zu migrierenden Repositories. Sicherungskopien von Mappings oder nicht mehr verwendete Mappings sollten vor dem Export gelöscht werden. Beim Import von Mappings, die auf nicht mehr existierende Spalten bzw. Objekte verweisen kann es zum Komplettabbruch des Imports kommen.
Transformationen sollten mit einem Extraexport transportiert werden. Dieser muss mit den Einstellungen:

- Import all objects
- Merge Metadata
- Match by name

Importiert werden. Bei der Einstellung „Import selected objects“ werden die Transformationen beim Import übersprungen. Im Logfile steht dann folgende Meldung

```
Object xxx skipped, because it was not selected for import!
```

Der Gesamtstatus des Imports ist damit erfolgreich und ein folgender Import von Mappings führt zu fehlenden Referenzen.
Beim Import von mehr als 20 Mappings kam es bei uns reproduzierbar zu Fehlermeldungen. Importierte man die Mappings in Blöcken von je 15 Stück, so waren diese Imports fehlerfrei.

Ausblick

Ab dem Release 11.2 können Spaltenreihenfolgen von Operatoren verändert werden. Je nach verwendetem Tool für den Vergleich der Dumpdateien kommt es hier zur Meldung von Unterschieden, obwohl keine echte Änderung vorliegt. Eine Umsortierung von Spalten sollte daher vermieden werden, wenn diese nicht notwendig ist, z. B. bei Union Operatoren.

Bei Migrationen reicht aber der reine Metadatenvergleich nicht aus, einige Mappings ließen sich unter 11.2 nicht erfolgreich generieren, obwohl die Metainformationen übereinstimmten. Hier mussten noch die Mappingeigenschaften „Generate optimized Code“ bzw. „ANSI Syntax“ umgestellt werden. Diese Änderung muss dann natürlich bei Folgevergleichen berücksichtigt werden!

Fazit

Der OWB-Repositoryvergleich per Skripting ermöglicht auf einfache Weise Qualitätssicherung und Dokumentation. Im Rahmen von Migration kann man auf diese Weise sicherstellen, dass die Metadaten vollständig und fehlerfrei in das neue Repository übernommen worden sind. Eine Äquivalenz der daraus generierten Mappings kann allerdings nicht abgeleitet werden!
Beim Vergleich von unterschiedlichen Stages kann man validieren, ob die gewünschten Objekte erfolgreich deployed wurden.
Bei Fragen zu den einzelnen Skripten bzw. weiteren Fragen zu Skripting können Sie sich gerne an mich wenden.

Kontaktadresse:

Oliver Gehlert
Metafinanz-Informationssysteme GmbH
Leopoldstraße 146
D-80804 München

Telefon: +49 (0) 89 360531-0
Fax: +49 (0) 89 360531-5015
E-Mail oliver.gehlert@metafinanz.de
Internet: www.metafinanz.de