# Sinn und Zweck der Code Templates im Oracle Warehouse Builder 11.2

**Negib Marhoul**
**ORACLE Deutschland B.V. & Co. KG**
**Potsdam, Deutschland**

**Schlüsselworte**

Oracle Data Warehouse, Oracle Warehouse Builder, OWB, Oracle Data Integrator, ODI

## Einleitung

Mit der Einführung vom Oracle Warehouse Builder 11gR2 im Jahr 2009 wurde die Funktion der sog. Code Templates eingeführt. Diese Funktion bietet eine Alternative zu der üblichen Anbindung von Non-Oracle Systemen mit Gateways oder generischen ODBC. Die Funktionalität der Code Templates hat ihren Ursprung in den „knowledge modules“ des Oracle Data Integrators. Laut dem Statement of Direction für Oracle Data Integrator und Oracle Warehouse Builder (Stand Erstausgabe Januar 2010, aktualisiert im März 2013) ist eine Migration bestehender Oracle Warehouse Builder Projekte in ein ODI Projekt nicht erforderlich. Wenn jedoch eine Migration in die ODI Umgebung bereits angedacht ist, dann sollten folgende Empfehlungen berücksichtigt werden. Um eine einfache technische Migration in die zukünftige Integrationsumgebung vorzubereiten, wird empfohlen, dass OWB Anwender ausgewählte bestehende Mappings in Code Template Mappings migrieren und neue Mappings mit den Code Templates implementieren. Vor diesem Hintergrund wird eine genauere Betrachtung der Code Templates erforderlich.

Im Folgenden werden die Code Templates vorgestellt und der ETL Vorgang am Beispiel eines auf Code Templates basierten Mappings erläutert. Dabei werden die Vorteile und Nachteile dieser Funktion dargestellt und gleichzeitig einige praktische Tipps skizziert, um den Empfehlungen aus dem Statement of Direction nachzukommen.

## Was sind Code Templates?

Data Warehouse Projekte zeichnen sich dadurch aus, dass viele Daten aus heterogenen Datenbanksystemen zentral in eine Datenbank integriert werden. Eine Aufgabe besteht darin, die Quellsysteme der erforderlichen Daten zu ermitteln und diese mit den nativen Verbindungsmöglichkeiten anzubinden. Nicht selten sind unterschiedliche Datentransportverfahren für die gleiche Technologie möglich. Um das effizienteste Datentransportverfahren auszuwählen, werden in erster Linie die Performanz, Flexibilität und geringste Systembelastung zur Bewertung herangezogen. Im homogenen Oracle DB Umfeld werden im Allgemeinen die Database Links für die Datenintegration verwendet. Darüberhinaus werden für die Anbindung von Non-Oracle Datenbanken die heterogenen Services der Oracle DB genutzt. Die heterogenen Services (HS) der Oracle DB stellen ein Framework bereit, um mit einem Oracle Gateway oder einer generischen ODBC Verbindung die Non-Oracle Datenbanken anzubinden.
Ab dem OWB 11.2 bilden die Code Templates die Basis, um heterogene Datenbanksysteme über native JDBC Schnittstellen zu verbinden. Die Code Templates kommen in den Code Template Mappings zum Einsatz. Die Code Template Mappings (CTMs) bilden das Framework und ermöglichen die Programmierung von ETL Strecken für die Integration von Daten aus Non-Oracle Datenbanksystemen.

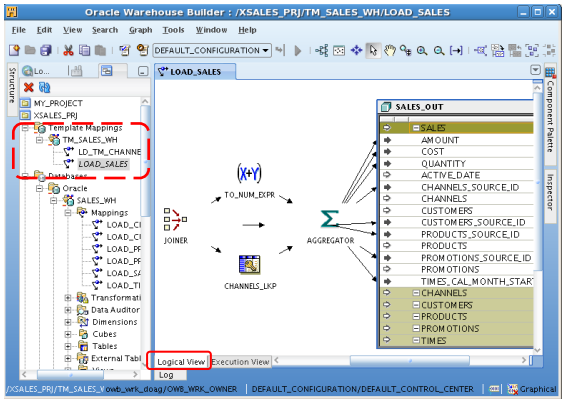



*Abb. 1: Code Template Mappings Logical View*

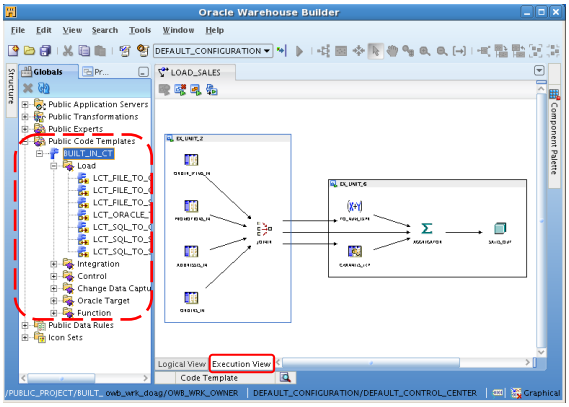



*Abb. 2: Code Template Mappings Execution View*

Das CTM umfasst den Datentransport sowie die Datenverarbeitung in Form der bekannten OWB Components. Abbildung 1 zeigt die Logical Views eines CTMs. Man sieht, dass Joiner, Lookups, Aggregatoren und die speziellen Oracle Datenbank Objekte, wie Dimensions und Cubes, als Operatoren in den ETL Vorgang eingebettet werden können. In der Execution View der CTMs (siehe Abbildung 2) wird der gesamte ETL Vorgang in einzelne Verarbeitungseinheiten (Execution Units) eingeteilt. Jede dieser Einheiten hat vom Entwickler eine Code Template Definition zugewiesen bekommen. Die Code Templates werden mit dem OWB im Standard ausgeliefert und können aus der Globalen Projekt Konfiguration direkt genutzt werden. Ein CT kapselt einen Satz an Operationen, die die Logik für den Datenfluss beinhalten. Diese Operationen sind generisch implementiert. Dasselbe Code Template kann für unterschiedliche Systeme verwendet werden. Der OWB nutzt die generische Implementierung der Code Templates, um durch die JDBC-Verbindungsfunktion flexibel auf Non-Oracle Systemen zuzugreifen, Daten zu extrahieren und in eine Zieldatenbank zu integrieren. Schon in der Vergangenheit standen mit dem OWB einige Funktionen bereit, um heterogene Systeme in den ETL Prozess einzubinden. Allerdings waren diese nicht selten etwas komplizierter aufzusetzen und häufig plattformspezifisch und limitiert (z.B. generische ODBC Treiber per „Heterogene Services“). Auch mussten spezielle Funktionen wie die „transportable tablespaces“ oder „data pump“ außerhalb des ETL Prozesses ausgeführt werden. Dagegen stehen für die meisten Datenbanksysteme native und plattformunabhängige JDBC Treiber zur Verfügung, die einfach in die OWB Umgebung eingebunden werden können. Die Treiber-Datei wird in das vorgesehene OWB Verzeichnis kopiert. Es wird keine weitere Konfiguration benötigt.

Der ETL Code wird vom OWB generiert. Das gilt auch im Fall der Code Templates, wobei der generische Quellcode für das Quellsystem und Zielsystem als Jython API im CT hinterlegt ist.

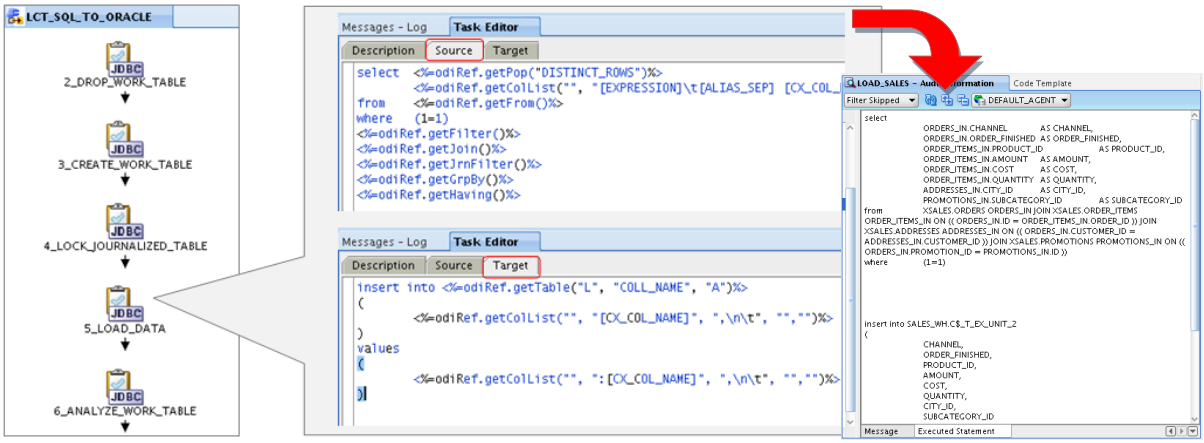



*Abb. 3: „Lade“ Code Template mit Quell und Ziel Operationen und generiertem Quellcode*

Neben den Standard Code Templates können auch eigene Code Templates implementiert werden, z.B. um klassische Ladeoperationen zu optimieren.
Ein einfacher Ladevorgang für ein MS SQL Server könnte in einem Bulk-Load Vorgang umgewandelt werden, in dem systemspezifische Funktionen wie das BCP von Microsoft (vergleichbar mit expdp/impdp von Oracle) genutzt werden. Durch diese „open connectivity" ist es seit dem OWB 11.2 möglich auch Non-Oracle Systeme über die gleichen Methoden mit Daten zu beladen. Neben dem bidirektionalen Datenaustausch können auch strukturierte Dateien wie XML oder CSV generiert werden. Ursprünglich stammen die Code Templates aus dem Oracle Data Integrator und können dort in der Form der sog. Knowledge Module wieder gefunden werden. Es ist davon auszugehen, dass die Code Template Mappings des OWBs sich wesentlich einfacher in die zukünftige strategische Plattform (ODI) migrieren lassen.

**Einsatz der Code Templates**
Wie eingangs erläutert, werden die CodeTemplate Mappings analog zu den klassischen OWB Mappings entwickelt. Dem Entwickler stehen derselbe Design-Editor und dieselbe Components Palette zur Verfügung, um den ETL Vorgang zu implementieren. Im zweiten Schritt der Entwicklung wird die Einteilung des ETL Vorgangs in die sog. Execution Units vorgenommen. Dabei muss jeder Execution Unit ein Code Template zugewiesen werden. Aus der Perspektive des Datenflusses werden für die einzelnen Etappen unterschiedliche Code Templates benötigt. Um beispielsweise die Daten zwischen zwei getrennten Systemen (Execution Units) zu transportieren, werden Load-Code Templates (LCT) verwendet. Um Daten innerhalb desselben Systems zu verarbeiten, werden Integration-Code Templates eingesetzt (ICT). Diese unterschiedlichen CTs werden benötigt, da z.B. das LCT Daten zwischen zwei getrennten Systemen transportiert und dazu dedizierte Funktionen der beteiligten Technologien verwendet. Die Code Templates beeinflussen, wie die Daten transportiert und ggf. auch spezifisch vorverarbeitet werden. Die hauptsächliche Transformation der Daten findet im Mapping statt.

Folgendes Beispiel soll den Einsatz der Code Templates und CT Mappings verdeutlichen. Das Zielsystem ist eine Oracle Datenbank und als Quellsysteme kommen Oracle und MS SQL zum Einsatz. Die Daten der MS SQL DB werden über eine JDBC Verbindung und das Code Template LCT_SQL_TO_Oracle in der Ex_Unit_2 mit den Daten der Oracle DB vereint. Die Ergebnismenge wird dann über einen Database Link in das Zielsystem geschrieben und dort weiter verarbeitet.

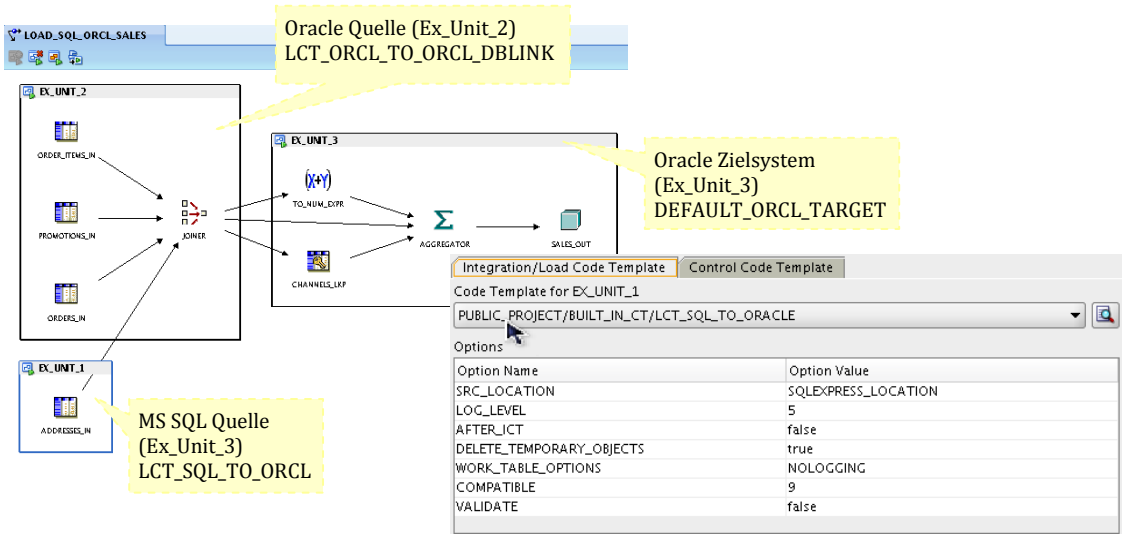



*Abb. 4: Beispiel für ein Code Template Mapping mit MS SQL als eine Quelle und Oracle als Ziel*

Bei der Implementierung des Datenflusses besteht Flexibilität. Man könnte das obere Beispiel so verändern, dass die Join-Operation auf dem MS SQL Server abläuft und im letzten Schritt das Ergebnis aus dem MS SQL Server in die Oracle DB geschrieben wird.

Im Gegensatz zu den klassischen OWB Mappings kann der Entwickler mit den CTMs den Datenfluss des ETL Vorgangs optimal an die gegebene Systeminfrastruktur anpassen. Es ist nicht zu empfehlen, unnötig Daten zwischen getrennten Systemen zu kopieren. Es ist effizienter, die Daten in einem Schritt zusammenzufassen und anschließend die weitere Verarbeitung durchzuführen. In Abbildung 5 wird ein Join im Quellsystem durchgeführt, um die Datenmenge vor dem Transport zu reduzieren. Ein weiterer Join wird im Zielsystem durchgeführt, um unnötiges Datenkopieren zu vermeiden.

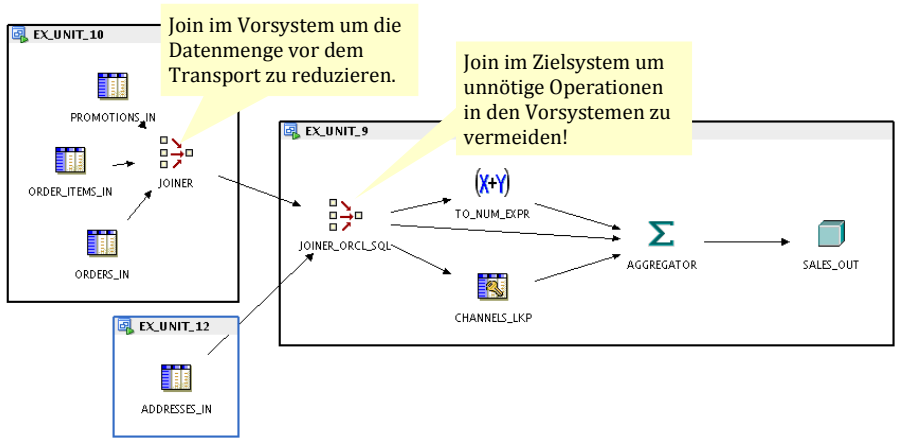



*Abb. 5: Angepasster Datenfluss bei leicht verändertem Mapping*

Im klassischen OWB Mapping wird der Join immer im Zielsystem ausgeführt.
Die systemspezifischen Code Templates haben die Einschränkung, dass nicht alle Objekte der OWB Component Palette unterstützt werden. Bestimmte Operatoren oder Objekte lassen sich nicht in Non-Oracle Systemen ausführen. Der Look-Up Operator und die Oracle Datenbankobjekte „Dimension“ bzw. „Cube“ werden nicht unterstützt. Diese Operatoren können nur genutzt werden, wenn das System, auf dem die Verarbeitung durchgeführt wird, eine Oracle DB ist und als Code Template DEFAULT_ORCL_TARGET ausgewählt wurde. Andernfalls wird das Deployment mit einer Fehlermeldung beendet. Ein Code Template Mapping kann aber neben dem Default Template auch systemspezifische Templates enthalten. Damit können komplexe ETL Vorgänge mit heterogenen Datenbanken sehr einfach und flexibel entwickelt werden.

Die Code Template Mappings werden in zwei automatisierten Schritten ausgeführt.

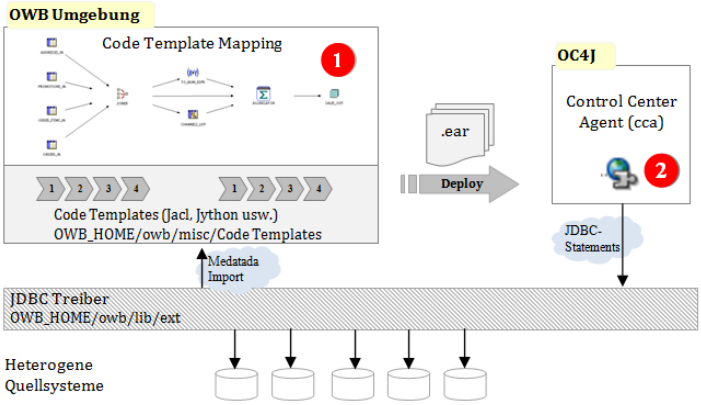



*Abb. 6: Das Deployment und Aufbau der Software Architektur*

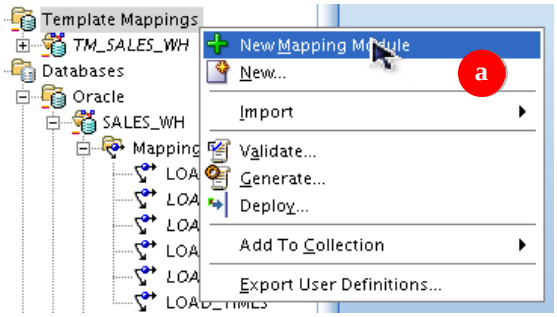
Im ersten Schritt werden die Skripte generiert und in eine .ear-Datei verpackt. Im zweiten Schritt werden diese Anweisungen durch den Control Center Agent auf dem definierten System ausgeführt. Das Deployment erfolgt mit dem OWB Control Center. Im Gegensatz zum klassischen Deployment von OWB Mappings werden im Fall von Code Templates Mappings keine PL/SQL Packages in die Datenbank geschrieben. Eine Ausnahme bilden die Mappings, die das DEFAULT_ORCL_TARGET Code Template beinhalten. Aus diesen Mappings werden PL/SQL Packages generiert, und diese werden in einem weiteren Schritt in die Datenbank geschrieben.

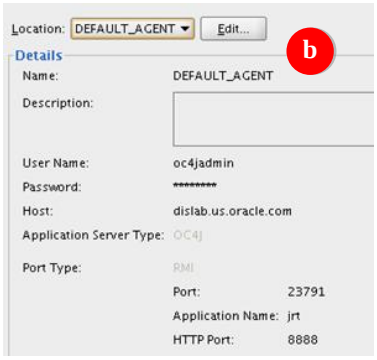
Im Allgemeinen findet die Datenverarbeitung auf den DB Systemen mengenbasiert und performant statt. Bei generischen SQL Code Templates werden die Daten per JDBC vom Java Agent gelesen und in das Zielsystem geschrieben. Dieser Ansatz bietet ein hohes Maß an Flexibilität, ist jedoch bei sehr großen Datenmengen unter Umständen nicht optimal für die Performanz. Daher sollten in solchen Fällen eher systemspezifische Code Templates genutzt oder entwickelt werden.

**Migration eines OWB Mappings in ein Code Template Mapping**
Die beschriebenen Funktionen der Code Template Mappings und der Code Templates des OWBs finden sich im Oracle Data Integrator wieder. Sowohl die Agent-Architektur als auch das Prinzip der flexiblen Anbindung heterogener Systeme sind analog aufgebaut. Es lässt sich daraus schließen, dass ein weicher Übergang in die strategische Produktlinie (ODI) von Oracle bereits vorbereitet worden ist. Oracle empfiehlt im Statement of Direction, dass einfache Mappings in Code Template Mappings migriert werden und neue Mappings als Code Template Mappings entwickelt werden.

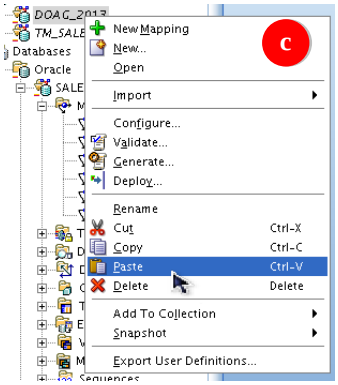
Im Folgenden wird beschrieben, wie in drei Schritten eine Migration eines bestehenden Mappings durchgeführt wird.

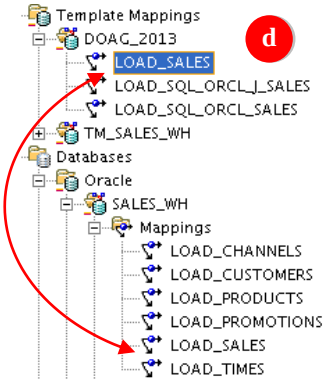
1. Erstellen eines neuen Template Mapping Moduls und zuweisen des Control Center Agents



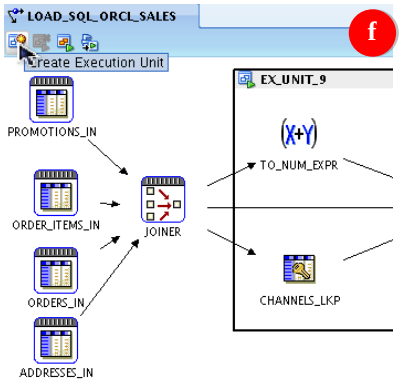



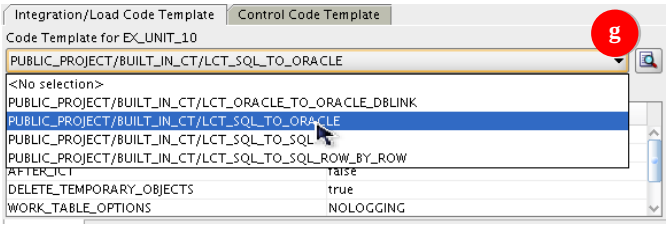

2. Copy und Paste eines bestehenden Mappings in das neue Mapping Modul



3. Erstellen der Execution Units und Zuweisen der Code Templates




**Fazit**

Code Template Mappings können leicht in bestehenden Projekten genutzt werden, da sich ihr Aufbau nicht wesentlich von dem der OWB Mappings unterscheidet. Sie bieten vielmehr die Möglichkeit, tiefer in den ETL Vorgang einzugreifen, und erlauben eine individuelle Anpassung an spezielle Technologien oder Systeminfrastrukturen. Mit der JDBC Connectivity können sehr flexibel heterogene Systeme angebunden werden. Eine Migration von bestehenden Mappings kann im Einzelfall sehr schnell umgesetzt werden. Es bleibt jedoch offen, ob in Zukunft alle OWB Operatoren genauso reibungslos in die ODI Plattform migriert werden können.

**Kontaktadresse:**
Negib Marhoul
ORACLE Deutschland B.V. & Co. KG
Schiffbauergasse 14
14467 Potsdam

Telefon: +49 (0) 1743098754
Fax: +49 (0) 3312007561
E-Mail: negib.marhoul@oracle.com
Internet: www.oracle.com