Erfahrungsbericht - Oracle ADF 11g im produktiven Einsatz

Martin Kunze
Robotron Datenbank-Software GmbH
01189 Dresden

Schlüsselworte:

Erfahrungsbericht, Oracle ADF 11g, Tipps und Tricks

Einleitung

Das Application Development Frameworks 11g (ADF) ist fester Bestandteil der Oracle Fusion Middleware. ADF ist das zentrale Java EE Entwicklungs-Framework von Oracle. Mit ADF lässt sich die Entwicklung von modernen Geschäftsanwendungen erleichtern.

Bereits im Jahr 2009 wurde die "Production"-Version des Frameworks und der dazugehörigen Entwicklungsumgebung JDeveloper 11g veröffentlicht. 2009 erfolgte bei der Robotron Datenbank Software GmbH eine systematische Evaluierung dieses Frameworks. Als Ergebnis dieser Untersuchung wurde die Praxistauglichkeit von ADF als Entwicklungsplattform verifiziert. Mittlerweile wurden mehrere größere Anwendungen auf der Basis dieses Frameworks realisiert. In diesem Artikel wird über die in diesen Projekten eingesetzten Technologien und die Projekterfahrungen berichtet.

Ein besonderer Fokus wird auf die Darstellung der Herausforderungen beim Umstieg einer existierenden Oracle Forms-Anwendung zu einer ADF-Anwendung gelegt. Insbesondere Themen wie Datenbank-Connection-Pooling, Transaktionskontrolle und Wiederverwendung von PL/SQL-Logik werden detailliert betrachtet.

Motivation

In der Vergangenheit wurde bei der Robotron Datenbank-Software GmbH primär Oracle Forms als Hauptentwicklungsplattform eingesetzt. Diese Plattform ist etabliert und aufgrund der wenigen Änderungen/Neuerungen als sehr stabil anzusehen. Der Hauptkritikpunkt in der Anwendungsentwicklung stellt jedoch häufig die nicht mehr zeitgemäße Benutzungsoberfläche dar. Die zunehmende Anforderung einer modernen, web-fähigen Benutzungsoberfläche für Anwendungen ist einer der Beweggründe, den Einsatz von Oracle ADF zu forcieren.

Technologien

Die technologische Basis aller ADF-Projekte ist der sogenannte "Fusion Stack" (Abbildung 1). Dieser besteht aus den ADF-Business-Components (BC) und einzelnen Web-Services in der Business-Service-Schicht, den ADF-Taskflows als Controller sowie den ADF-Faces als Präsentationsschicht.

Die Technologien ADF-Desktop-Integration (Integration in Office) und auch ADF-Mobile (Entwicklung von mobilen Anwendungen) werden aktuell nicht von Robotron in der Softwareentwicklung genutzt. ADF-Swing als eine mögliche Alternative für Desktopanwendungen wird von Robotron aufgrund der sehr geringen Unterstützung durch Oracle in Hinblick auf die Bedienbarkeit der Entwicklungsumgebung und die unzureichende Dokumentation nicht weiter verwendet. Die Ankündigung von Oracle, ADF-Swing aus der nächsten Release-Version von ADF zu entfernen, bestätigt diese Entscheidung:

"With the new release of Oracle JDeveloper 11.1.2 we have announced the deprecation of the ADF Swing technology" (Quelle: JDeveloper/ADF 11g Release 2 – Release Notes)

Als Datenbank kommt bei allen Projekten die Oracle eigene Datenbank (10g und verstärkt 11g R2) zum Einsatz.

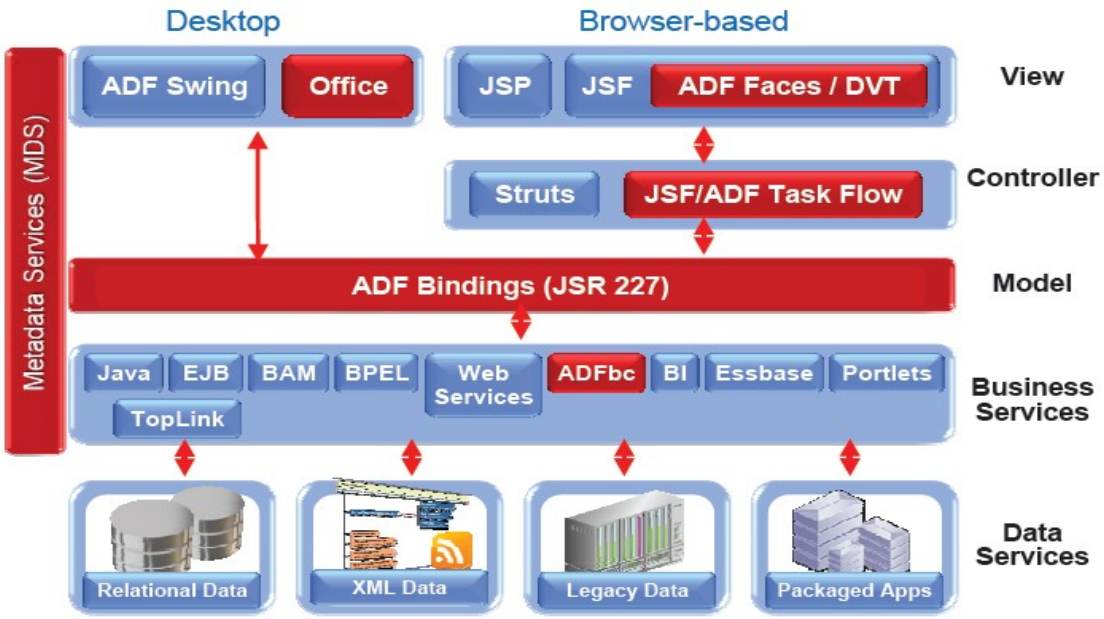

Abbildung 1: verwendete ADF-Technologie

Herausforderung

Hauptentwicklungsplattform der letzten Jahre war bei der Robotron Datenbank-Software GmbH Oracle Forms. Die größte Herausforderung beim Umstieg auf Oracle ADF stellt das Verständnis der Architektur einer ADF-Anwendung dar, da sich diese grundsätzlich von der Architektur einer Forms-Anwendung unterscheidet. Obwohl Forms-Anwendungen wie normale Web-Anwendungen deployed werden, basiert Forms letztendlich auf einer Client-Server-Architektur. ADF-Anwendungen (ADF-Swing ausgenommen) basieren hingegen auf einer Webarchitektur. Diese unterschiedlichen Ansätze müssen auch in der Programmierung berücksichtigt werden.

Folgende Themen sollten beim Umstieg von Forms zu ADF berücksichtig werden:

1. Connection-Pooling
    a. Während in Forms-Anwendung (oft) eine 1:1-Beziehung zwischen Nutzer und DB-Verbindung besteht, wird in Webanwendungen der Gebrauch des Connection-Poolings genutzt. D.h., eine DB-Verbindung kann nicht mehr eindeutig einem Nutzer zugeordnet werden.
2. Mittelschichttransaktionen vs. Datenbanktransaktionen
    a. Transaktionen in der Mittelschicht sind nicht exklusiv an eine Datenbanktransaktion gebunden. Änderungen auf der Datenbankebene (beispielsweise über PL/SQL) müssen innerhalb eines Request abgearbeitet werden.
    b. Die Transaktionssteuerung in der Mittelschicht erfordert, dass einzelne Prozeduren und Abläufe in der Datenbank weder ein COMMIT noch ein ROLLBACK absetzen.
    c. Datenbankprozeduren haben keine Kenntnis über Änderungen in der Mittelschicht, bis diese Änderungen mit einem Commit abgeschlossen werden. Daher muss sichergestellt werden, dass nach einer Änderung in der Datenbank die entsprechenden Elemente in der Mittelschicht aktualisiert werden.
3. Sperrverhalten
    a. Pessimistischen Locking, welches häufig in Forms-Anwendungen verwendet wird, ist für Webanwendungen nicht empfehlenswert. So kann das „unsachgemäße“ Beenden einer Anwendung, indem der Browsers geschlossen wird, zu ungewünschten Locks in der Datenbank führen.
    b. Teilweise existieren Anforderungen, einzelne Datensätze bereits beim Betrachten und nicht erst beim Editieren zu sperren (in ADF-BC wird der Datensatz beim pessimistischen Locking erst dann gesperrt, wenn eine setAttribute-Methode aufgerufen wird). Mithilfe der ADF-Business-Components ist jedoch auch diese Anforderung problemlos zu realisieren.
4. PL/SQL
    a. In Forms verwendete PL/SQL-Logik lässt sich über verschiedene Wege in ADF-Anwendungen wiederverwerten. Einerseits kann PL/SQL direkt im Code mittels Java (java.sql.Statement und deren konkrete Implementierungen) ausgeführt werden. Andererseits ist es auch möglich, ADF-BC-View-Objects in Verbindung mit DB-Object- und DB-Table-Type direkt auf PL/SQL-Prozeduren zu erstellen (Abbildung 2).

```
create or replace type EMPLOYEE_TYPE as object (
 EMPLOYEE_ID Number,
 FIRST_NAME Varchar2(20),
 LAST_NAME Varchar2(25),
 EMAIL Varchar2(25),
 DEPARTMENT_ID Number
)

create or replace type EMPLOYEE_TABLE as table of EMPLOYEE_TYPE;

create or replace
function getEmployeesPipelined(departmentId Number) return EMPLOYEE_TABLE
PIPELINED as
begin
   for rec in (select … from employees where DEPARTMENT_ID = departmentId)
loop
```

```
        pipe    row(new    EMPLOYEE_TYPE(rec.EMPLOYEE_ID,    rec.FIRST_NAME,
rec.LAST_NAME, rec.EMAIL, rec.DEPARTMENT_ID));
  end loop;
  return;
end;
```

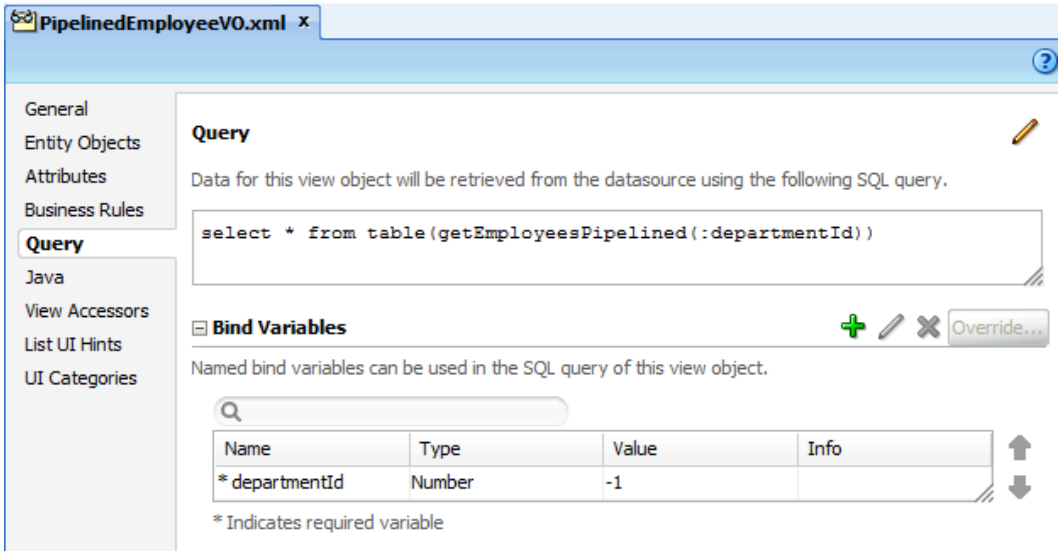

Abbildung 2 PL/SQL-Verwendung in ADF BC

5. Globale PL/SQL-Variablen
    a. Globale PL/SQL-Variablen sollten, aufgrund des Connection-Poolings in ADF-Anwendung grundsätzlich vermieden werden.
6. Security
    a. Bestehende Forms-Anwendungen verwenden häufig Security-Mechanismen, die direkt mit der Datenbank verwurzelt sind. ADF-Anwendungen hingegen implementieren die Security in der Mittelschicht. Eine Wiederverwendung der alten Logik ist damit schwierig.

Erfahrungen

In den bisherigen Projekten bei der Robotron Datenbank-Software GmbH wurden folgende Erfahrungen gesammelt:

1. Beim Umstieg auf Oracle ADF sollte der Einarbeitungsaufwand für die Entwickler nicht unterschätzt werden. Bis ein Entwickler mit dem Framework und der Entwicklungsumgebung richtig umgehen kann, vergehen in der Regel mehrere Monate. Nach dieser Einarbeitungszeit steht einer schnellen Anwendungsentwicklung jedoch (fast) nichts mehr im Wege.
2. Das Framework bietet den Entwicklern zahlreiche Möglichkeiten, einen „Use-Case“ umzusetzen. Es ist daher ratsam, Richtlinien zur Umsetzung vorzugeben, um eine einheitliche Anwendungen zu erhalten. Die Richtlinien sollten möglichst „Best-Practise“-Lösungen aufzeigen.
3. Die sogenannte „Browserhölle“ existiert in abgeschwächter Form auch für ADF-Anwendungen weiterhin. Vor dem Projektbeginn sollte daher mit dem Auftraggeber geklärt werden, welche Webbrowser verwendet werden sollen. Mit älteren Webbrowsern wie dem IE7 sind komplexe und performante Anwendungen nur schwierig zu realisieren. Mit dem zunehmenden Umstieg auf das Betriebssystem Win 7 und den Webbrowser IE9 gehören diese Probleme bald der Vergangenheit an.
4. Eine Migration ist nicht für jede Anwendung sinnvoll. Besonders für größere Anwendungen, die über Jahre gewachsen sind und deren Architektur nicht der einer Webanwendung entspricht, ist ein Umstieg nur mit enormen Aufwand und Kosten möglich und daher nicht unbedingt empfehlenswert. Statt von einer Migration sollte in diesem Zusammenhang aufgrund der unterschiedlichen Architektur von Forms- und ADF-Anwendungen besser von einer "Transformation" bzw. einer "Modernisierung" gesprochen werden.

Fazit

Oracle ADF 11g und der JDeveloper 11g vereinfachen die Entwicklung von Webanwendungen erheblich. Oracle stellt kontinuierlich neue Features zur Verfügung. Die Stabilität und Geschwindigkeit des JDevelopers ist mit Release 2 deutlich besser geworden. Die Performance des integrierten Weblogic-Servers ist jedoch noch als deutlich verbesserungswürdig anzusehen. Auch Bugs beziehungsweise Veränderungen am Framework (z.B. AM-Pooling-Mechanismen) nach einem Release-Wechsel trüben ein wenig den insgesamt positiven Eindruck. Wir gehen davon aus, dass diese Defizite in den nächsten Versionen abgestellt werden.

Die Erfahrungen in den Projekten zeigen, dass es nicht immer sinnvoll ist, bestehende (Forms)-Anwendungen nach ADF zu transformieren. Aufgrund der unterschiedlichen Architektur und abhängig von der Komplexität und dem Umfang der bestehenden Anwendungen kann ein Umstieg sehr kostenintensiv sein. Eine gründliche Evaluierung aller damit verbundenen Vor- und Nachteile ist daher essentiell.

Kontaktadresse:

Martin Kunze
Robotron Datenbank-Software GmbH
Stuttgarter Straße 29
01189 Dresden

Telefon: +49 (0) 351/2 58 59 28 75
E-Mail martin.kunze@robotron.de
Internet: www.robotron.de