Forms Legacy
Ein ADF Panorama

Enno Schulte
OPITZ CONSULTING GmbH
Gummersbach

Schlüsselworte:

Application Development Framework, ADF, Contextual Events, UI Shell, Rich Internet Applications, Task Flows, Regions, ADF Faces, Forms

Einleitung

Obwohl es noch viele Entwickler leugnen, ist Forms vielleicht nicht tot aber zweifelsohne das Cobol der Neuzeit. Die Nachfolge ist seitens Oracle geklärt. Wer State-Of-The-Art entwickeln möchte soll zum Application Development Framework (ADF) greifen. Trotz der alt anmutenden Versionsnummer 11 ist ADF in seiner aktuellen Form noch relativ neu. Dementsprechend fehlt es der Wirtschaft an Projektreferenzen und Best-Practices.
Dieses Manuskript soll die aus der Forms-Ära gewachsenen Ansprüche an eine Entwicklungsumgebung aufzeigen und mit dem heutigen Stand von ADF vergleichen. Weiter sollen die reichhaltigen Möglichkeiten zur Front-End Gestaltung präsentiert werden.

Forms Legacy

Oracle Forms erlaubte als eine der ersten WYSIWYG Entwicklungsumgebungen interaktive Dialogmasken zu erstellen, ohne dafür viel Code schreiben zu müssen. Dabei wurde die Logik, in Form von PL/SQL, in die Datenbank verlagert und war von dort aus in verschiedenen Masken wiederverwendbar, wodurch die Notwendigkeit zur Programmierung weiter reduziert wurde.
Ein weiteres Merkmal der Entwicklung mit Forms war die unkomplizierte Integration von Masken zum Bearbeiten von Massendaten. Unterstützt wurde der Entwickler dabei von diversen Dialogen und Wizards die einen Großteil der Arbeit bereits erledigten.
Die auf diese Weise gewachsene Form der Anwendungsentwicklung steht bis heute zur Verfügung und obwohl die Technologie im Grunde über 30 Jahre alt ist, starten immer noch neue Projekte basierend auf Oracle Forms. Eine derartig lange Lebensdauer und der fortwährende Support durch Oracle hat definitiv Vertrauen geschaffen, das es nun gilt in die nächste Entwicklungsumgebung mitzunehmen.
Was ist also Forms Erbe?
Zum einen ist es eine Vielzahl an Entwicklern die schwer dafür zu begeistern sind, für eine Maske zur tabellarischen Verarbeitung von Daten, mehr als ein Select-Statement schreiben zu müssen. Dabei sind sie es gewohnt durch Wizards und Dialoge in ihrer täglichen Arbeit unterstützt zu werden. Auf der anderen Seite stehen ebenso viele Legacy Systeme, deren Logik mit der Datenbank verschmolzen ist und deren Neuentwicklung eine enorme Kostenstelle ist.
Schlussendlich bleiben noch die Besitzer und Verwalter dieser Legacy Systeme, die in kommenden Projekten ihre Unternehmenssoftware auf den neuesten Stand der Technik heben wollen, um möglichst großen Nutzen aus den aufkommenden Service orientierten Architekturen zu ziehen. Die zum Betrieb von ADF Applikationen notwendigen Infrastrukturen sind dabei in den meisten Fällen bereits vorhanden.

Sofern das Vertrauen der Forms-Ära noch nachwirkt, kann man davon ausgehen das ADF viele dieser Systeme ersetzen und es in Zukunft eine Menge Migrationsprojekte geben wird.

Ein ADF Panorama

Mit dem ADF schickt Oracle nun einen Java EE basierten Nachfolger für Forms ins Rennen. Aber hat das Framework das Potenzial die Nachfolge von Forms anzutreten?
Damit es zu Migrationsprojekten kommt, müsste zuerst die Entscheidung für ADF als zukunftsweisende Technologie getroffen werden. Um die entsprechende Klientel von den Vorteilen des Frameworks zu überzeugen wartet ADF mit seiner vollen Integration in die Middleware auf. Es bietet diverse Schnittstellen und bedient sich der weit verbreiteten Java EE Architektur. Um sich von den anderen Java EE basierten Frameworks zu unterscheiden liefert Oracle mit dem JDeveloper ein Tool, das den gesamten Entwicklungsprozess unterstützt und darüber hinaus vereinfachen und beschleunigen soll.
Eine Bewertung von ADF ohne den JDeveloper ist schlichtweg nicht möglich. Er ermöglicht erst viele der prozessbeschleunigenden Features. Allen voran den deklarativen Entwicklungsansatz, den man schon aus Forms kennt. Zudem ist er gespickt mit einer Vielzahl an Wizards die besonders für Anfänger einen erheblichen Mehrwert darstellen.
Der JDeveloper ist in gewisser Weise das Herz der Anwendungsentwicklung mit ADF und somit das Tool, was letztendlich für Entwickler den Ausschlag gibt, ob sie sich mit dem Framework wohl fühlen oder nicht. Dabei besteht das Problem darin, Eclipse-verliebte Java Entwickler und programmierscheue Forms Entwickler unter einen Hut zu bringen. Es wurde bereits häufig erlebt, dass Java Entwickler in ADF Schulungen unglücklich über den hohen Anteil an Wizards und deklarativer Entwicklung waren. Während die Forms-Entwickler sich nicht für die Programmierung Java basierter Validierungsregeln begeistern konnten.
An dieser Stelle gilt es einen Paradigmenwechsel herbeizuführen und die noch räsonierenden Stimmen über die instabilen Vorgängerversionen des JDevelopers zu überzeugen.
Ein sich dafür eignendes Feature ist die mühelose Anbindung an ein Metadata Service Repository (MDS), das es ermöglicht, die Änderungen des Benutzers am Frontend, Session übergreifend zu persistieren.
Neben netten Features wie dem MDS und dem Aufruf der Anwendung ohne vorherige Installation irgendeines Plug-Ins, sind wohl die reichhaltigen Komponenten aus dem ADF Faces Bibliotheken eines der Hauptargumente aus Anwendersicht. Abbildung 1 zeigt in einem Beispiel wie eine typische UI mit ADF Faces Komponenten gestaltet werden kann. Dabei kommt das UI Shell Pattern zum Einsatz. Dieses ermöglicht es, einzelne Task Flows innerhalb eines Reiters partiell nachzuladen. Auf der Abbildung sind zwei typische Tabellen und ein Netzdiagramm zu sehen. Für die Implementierung des UI Shell Patterns sind zwar ein paar Zeilen Code notwendig, nicht aber für die Gestaltung der Oberfläche. Die Tabellen als auch die Diagramme lassen sich mit wenigen Klicks in dieser Form generieren.

Abb. 1: Ein Taskflow im UI Shell Pattern

Ein weiteres Argument für die Entwicklung mit ADF ist der starke Fokus auf die Wiederverwendbarkeit in nahezu allen Bereichen. Dazu lassen sich Task Flows beispielsweise innerhalb von Regionen dynamisch laden. Die selbe Technik kommt im Rahmen des UI Shell Pattern zum Tragen. Auf Abbildung 2 (unten) ist in einem neuen Reiter ein anderer Task Flow zu sehen, der sich jedoch des in Abbildung 1 zu sehenden Task Flows bedient und diesen in einer eigenen Region mit einbettet.
Zudem bietet das Framework die Möglichkeit die so gleichzeitig sichtbaren Task Flows untereinander kommunizieren zu lassen. Dabei folgen die Regionen dem üblichen Observer-/Listener-Pattern und reagieren auf diverse Events, welche bei ADF als Contextual Events bezeichnet werden.

Abb. 2: Die Integration verschiedener Task Flows in einem Reiter

Zur Interaktion zwischen den Regionen stehen zudem weitere Techniken zu Verfügung. Als weiteres Beispiel ist eine Drag and Drop (DnD) Operation in Abbildung 3 dargestellt. Der Prozess der gezeigt wird, veranschaulicht das Hinzufügen eines Mitarbeiters zu einem Team.
Mit Techniken wie DnD und Contextual Events lassen sich Task Flows auf die verschiedensten Arten miteinander verknüpfen. Zudem sind Task Flows parametrisierbar und lassen sich auch in eigenständige Bibliotheken auslagern um in unterschiedlichen Applikationen wiederverwendet werden zu können.

Abb. 3: Drag and Drop Beispiel

Oft ist es von großem Vorteil ausgelagerte Task Flows in vielen Applikationen wiederverwenden zu können. Ein Beispiel dafür wäre ein Task Flow zum Bearbeiten von Mitarbeitern. Dieser könnte in vielen verschiedenen Anwendungen zum Einsatz kommen. Sei es ein System für die Abwicklung von Projekten oder einfach nur ein System zur Stammdatenpflege. Datensätze, wie die von Mitarbeitern werden oft an mehreren Stellen bearbeitet. Ein Taskflow, der die ID eines Mitarbeiters als Parameter aufnimmt und daraufhin eine Maske zur Bearbeitung anzeigt, ist eine einfache Anforderung für ADF basierte Anwendungen. In Abbildung 4 ist ein solcher Task Flow zu sehen der zudem in einem Inline Window im Browser ausgeführt wird.

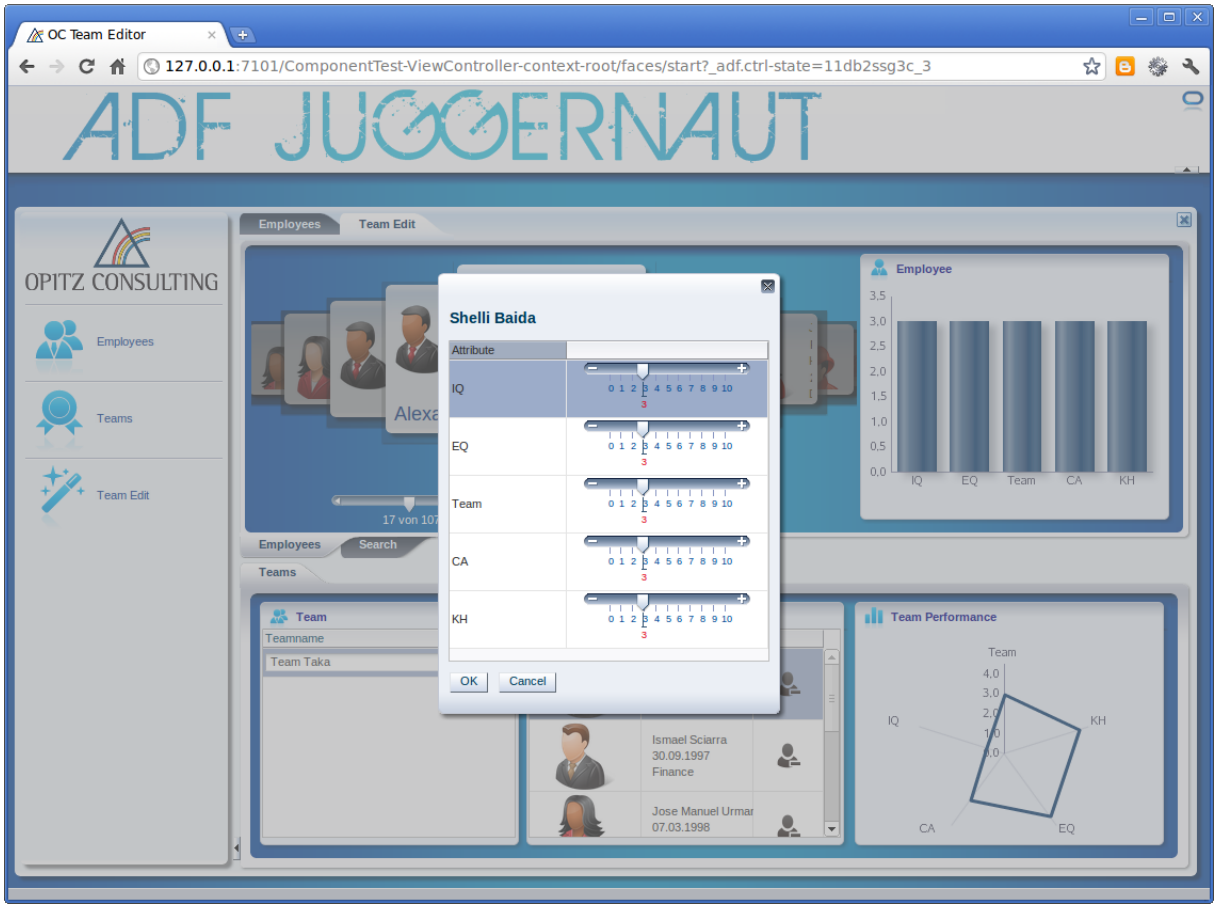

Abb.4: Inline Window Dialog zum Bearbeiten von Mitarbeiterdetails

Die Kombination aus Task Flow gesteuerten Prozessen und den vielen reichhaltigen ADF Faces Komponenten mindern die Grenzen zu Desktopapplikationen mehr und mehr und lassen ADF im Zeitalter der Rich Internet Applications die Potenziale browserbasierter Anwendungen voll ausschöpfen. Es wirkt, als sei der Weg in eine Zukunft erfolgreicher Migrationsprojekte mit ADF geebnet. Doch damit es dazu kommen kann, fehlt es dem Markt noch an der kritischen Masse kompetenter Entwickler, die das Leistungsvermögen des Frameworks an seine Grenzen treibt.

Enno Schulte
OPITZ CONSULTING GmbH
Kirchstraße 6
D-51647 Gummersbach

Telefon: +49 (0) 2261-6001 0
Fax: +49 (0) 2261-6001 4200
E-Mail enno.schulte@opitz-consulting.com
Internet: www.opitz-consulting.com