Seit Mitte 2010 gibt es Oracle Application Express (Apex) in der Version 4.0. Diese hat die Produktivität des Entwicklers nochmals drastisch erhöht und dabei dem Endanwender viele Standard-Funktionalitäten zusätzlich zur Verfügung gestellt. Dabei wird häufiger die Frage gestellt, ob Apex auch als Ersatz für Anwendungen geeignet ist, die auf Oracle Forms basieren. Leider lässt sich diese Frage nicht pauschal mit „ja" oder „nein" beantworten. Es kommt nämlich sehr auf den Anwendungsfall sowie die Komplexität an.

Kommt Apex nach Forms?

Niels de Bruijn, MT AG

Zunächst muss die Frage beantwortet werden, warum man überhaupt von Oracle Forms weg möchte. Oracle Forms ist ein sehr gutes Produkt für datenzentrierte Anwendungen und wird es auch weiterhin bleiben. Hauptauslöser sind oftmals die horrenden Lizenzkosten bei einer Migration. Oder man hat strategisch beschlossen, keine Neuentwicklung unter Oracle Forms mehr vorzunehmen.

Ist die Ablösung der Forms-Anwendung bereits beschlossene Sache, dann gibt es viele Argumente, die für den Einsatz von Apex sprechen:

- Die vielen Standardfeatures begeistern jeden Endanwender
- Apex ist im Standardumfang der Datenbank enthalten und verursacht keine zusätzlichen Lizenzkosten
- Apex ist für Oracle-Forms-Entwickler schnell erlernbar
- Sowohl die Ausführung als auch die Entwicklung von Anwendungen findet im Browser statt
- Die Produktivität ist sehr hoch, Apex ist deklarativ bedienbar und erfordert, wenn überhaupt, nur minimale Programmierarbeiten in PL/SQL

Rapid Application Development mit Apex 4.0

Wie produktiv Apex für Entwickler tatsächlich ist, hat zuletzt ein direkter Vergleich zwischen dem Google Web Toolkit (GWT) und Apex gezeigt: Bei gleicher Funktionalität (Masken zur Stammdatenverwaltung und eine Suchfunktion) wurden durch Experten Prototypen sowohl auf Basis von Apex als auch mit dem GWT entwickelt. Für die Apex-Lösung wurden neun Personentage benötigt, wohingegen für den GWT-Ansatz 27 Personentage anfielen.

Funktionale manuelle Migration

Wenn feststeht, dass Apex die Zieltechnologie darstellen soll, stellt sich die Frage, wie man die Funktionalität der Forms-Anwendung in die Apex-Umgebung überführt. Obwohl Apex eine Möglichkeit dazu vorsieht, gibt es keine wirklich praktikable Lösung für eine automatisierte Migration von Oracle Forms nach Apex: Das Ergebnis des Apex-Migrationsassistenten ist in 99 Prozent aller Fälle inakzeptabel und erfordert so viele Nacharbeiten, dass eine Neuentwicklung sinnvoller wäre. Besser ist es also, die Anwendung neu zu entwickeln und dabei einige Richtlinien zu beachten:

- Wo immer möglich, die Assistenten (Wizards) einsetzen
- Immer auf Views aufsetzen, niemals direkt auf Tabellen
- Apex als reine View-Schicht betrachten und die komplette Geschäftslogik in PL/SQL-Packages unterbringen
- Bei komplexen Views „instead of"-Trigger verwenden, damit weiterhin die Assistenten verwendet werden können
- Richtlinien für die Entwicklung vorab aufstellen (und einhalten!)

Für kleine und einfache Forms-Anwendungen reicht eine „One Man Show" meist aus. Für umfangreiche und anspruchsvolle Forms-Lösungen wird ein Team mit unterschiedlichen Kompetenzen vorausgesetzt. Neben Apex-Experten sind dann auch Datenbank-Profis und Experten mit Know-how über HTML, CSS und JavaScript/jQuery gefragt. Ein Qualitäts- und ein Projektmanager runden das Team ab.

Client/Server- vs. Web-Architektur

Egal, ob die Zieltechnologie Apex, Java, .Net oder PHP ist: Sobald ein Wechsel von einer Client/Server- Architektur in eine Web-Architektur erfolgt, wird das Ergebnis für den Endanwender niemals identisch mit der bisherigen Lösung sein. Eine Benutzeroberfläche im Web verhält sich grundsätzlich anders als bei Forms – und sieht auch anders aus. Außerdem gibt es viele Standard-Funktionalitäten in Apex, die man zusätzlich einsetzen kann und sollte. So wird man auf den Einsatz von „Interactive Reports" nicht verzichten wollen. Damit lassen sich die Stammdaten einer Anwendung bequem durch den Endanwender gruppieren, sortieren, ein- und ausblenden, kalkulieren, summieren und aufbereiten.

Das Locking-Verhalten ist bei Web-Anwendungen optimistisch und es gibt keine dedizierte Datenbank-Session pro Benutzer mehr. Änderungen an den Daten sind im Normallfall explizit zu speichern. Vieles wurde bereits im Web-Umfeld getan, um die Mimik von Forms-Anwendungen auch in Web-Anwendungen nachzuahmen, Fakt bleibt aber, dass die Endanwender sich umgewöhnen müssen. Deswegen ist es umso wichtiger, die Benutzer frühzeitig am Migrationsprojekt zu beteiligen. Auch hier kann Apex eine wichtige Rolle spielen: Damit lässt sich in wenigen Tagen ein Prototyp bereitstellen. Somit kann man das Design und die

Benutzerführung schnell klären und damit den Umfang der funktionalen Spezifikation drastisch reduzieren. Innerhalb von wenigen Iterationen entstehen zusammen mit dem Fachbereich konkrete Vorstellungen, wie die Anwendung aussehen muss und wie man sie bedient.

Erfolgreich am Ziel

Der Erfolg einer Migration hängt von verschiedenen Faktoren ab:

- Wie gut ist das Datenmodell?
- Wie gut sind die Entwickler?
- Wie kompliziert sind die Masken und inwiefern muss jede Maske 1:1 nachgebaut werden?

Die hohe Geschwindigkeit der Entwicklung mit Apex ist im Wesentlichen nur dann gegeben, wenn auch das Datenmodell gut strukturiert ist. Leider ist das nicht immer der Fall und es geht viel Zeit damit verloren, das Datenmodell zu verstehen und die richtigen SQL-Abfragen zu erstellen. Eine gute Anwendung braucht ein gutes Datenmodell, Apex hin oder her.

Steht das Datenmodell, dann kommt es entscheidend auf die Apex-Entwickler an. Auch Apex-Projekte können in eine Schieflage geraten, wenn nicht gewisse Richtlinien für die Entwicklung eingehalten werden. Außerdem ist viel Erfahrung notwendig, um zu beurteilen, ob etwas mit Apex-Bordmitteln geht und wie. Hierbei bieten die Quellen im Internet eine gute Unterstützung. Besonders auf den deutschsprachigen Apex-Seiten von Carsten Czarski sind viele praktische Tutorials zu finden (http://apex.oracle.com/url/apexcommunity).

Gerade bei der Ablösung von Forms-Anwendungen ist eine gründliche Analyse pro Maske notwendig, bevor mit der Umsetzung angefangen werden kann. In dieser Phase sollte einerseits das Verhalten der Maske erörtert und andererseits abgestimmt werden, wie die Maske im Browser aussehen wird. Erfahrungsgemäß muss man einen Kompromiss zwischen Bedienbarkeit und Aufwand finden. Die 80/20-Regel gilt nämlich auch für Apex-Projekte:

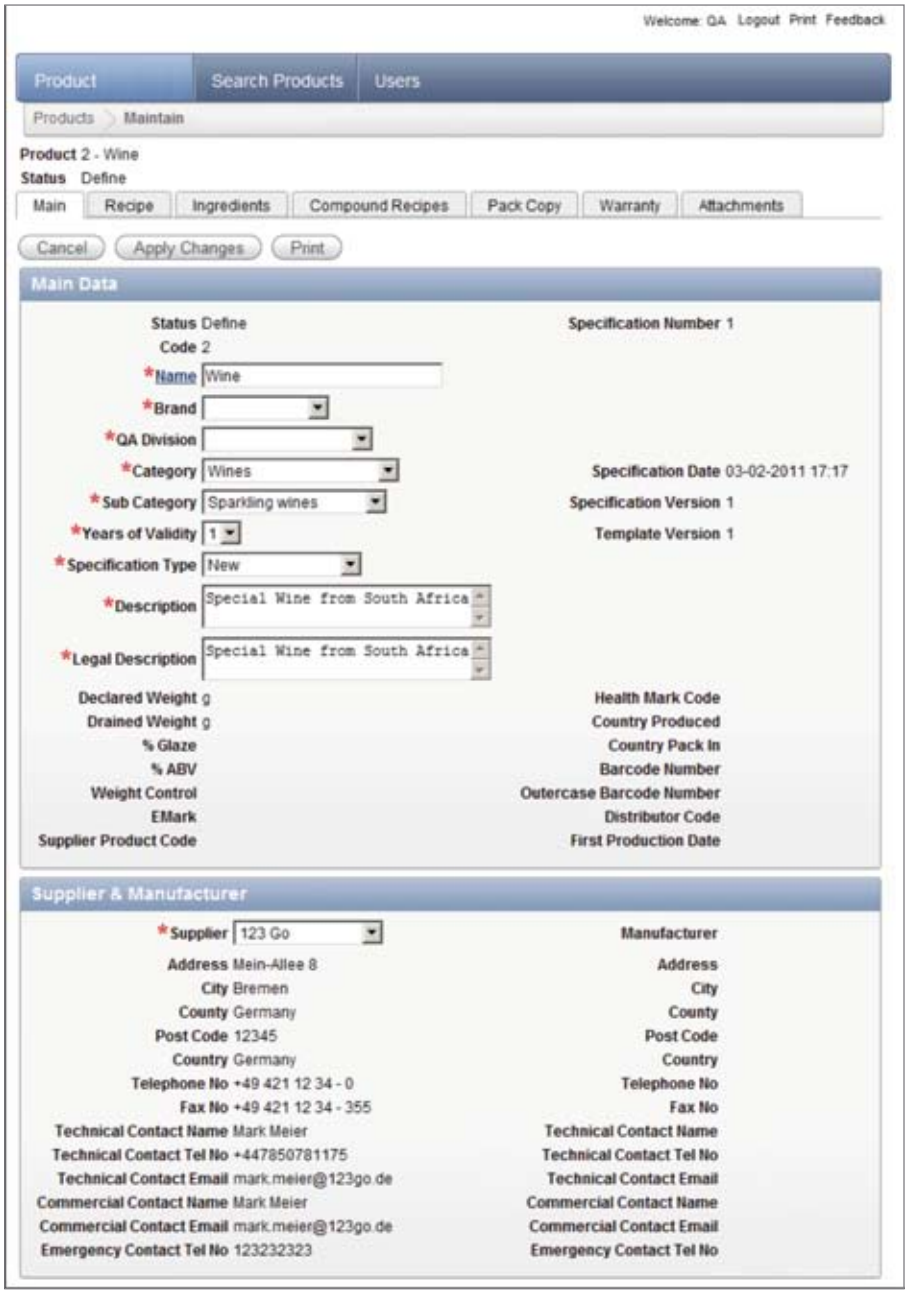

Abbildung 1: Screenshot des Apex-Prototyps

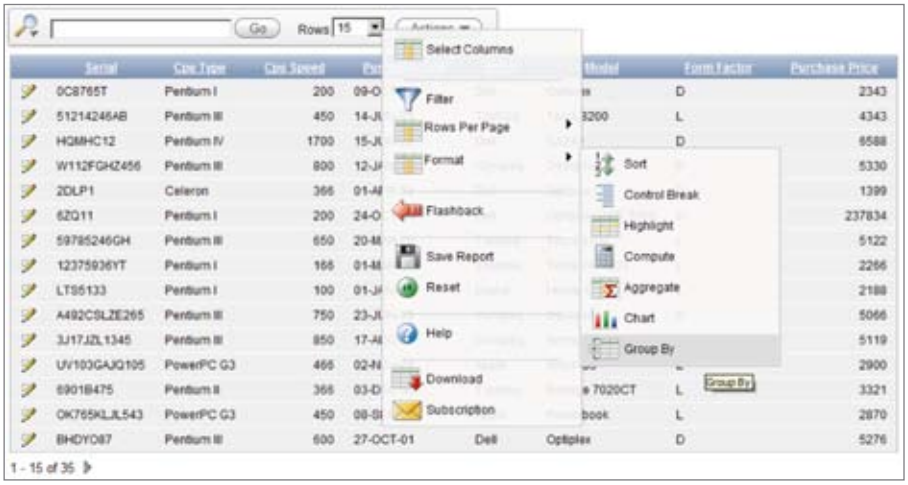

Abbildung 2: Interactive Report in Apex 4.0

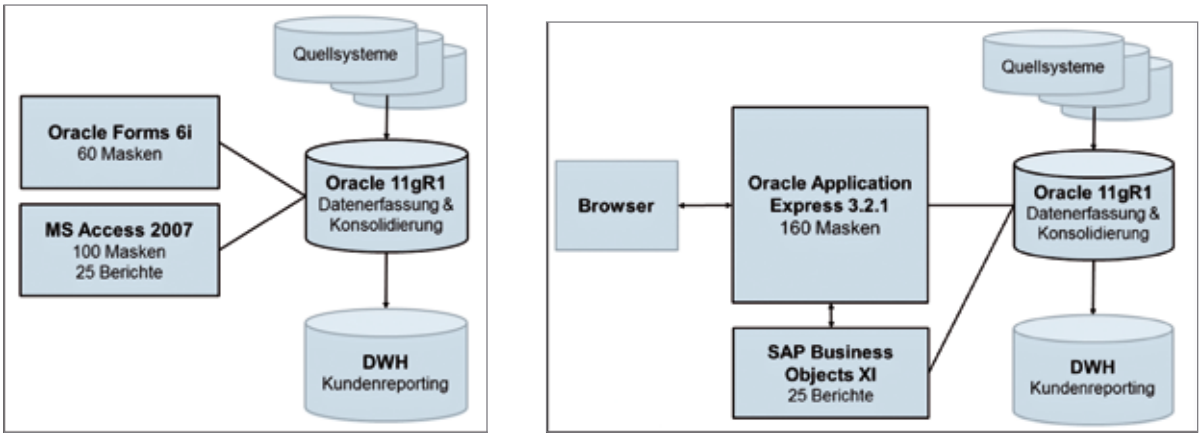

Abbildung 3: Architektur, vorher und nachher

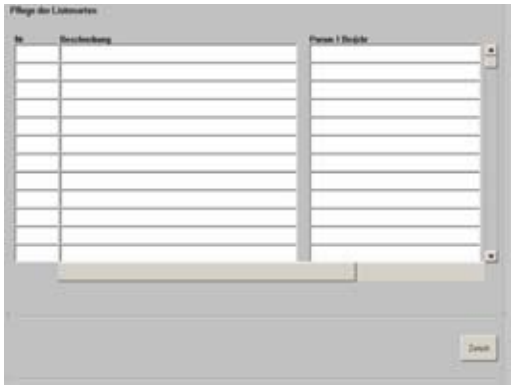

Abbildung 4: Von Oracle Forms 6i …

80 Prozent der Funktionalität lassen sich in 20 Prozent der Entwicklungszeit realisieren und umgekehrt.

Problemfall Forms-Masken

Rein wirtschaftlich betrachtet ist die Ablösung komplexer Oracle-Forms-Anwendungen meist nicht besonders sinnvoll, egal ob Apex, Java oder .Net als Zieltechnologie favorisiert wird. Im Falle von Apex gibt es bei Apex 4.0 noch einige Limitationen, die zwar umgangen werden können, die jedoch bei komplexen Forms-Masken ein manuelles Nacharbeiten erfordern und dadurch die Entwicklungszeit schnell in die Höhe treiben. Beispielsweise gibt es in 4.0 noch keine Möglichkeit, auf der gleichen Seite mehrere tabellarische Formulare oder Berichte vom Typ „Interactive Report" deklarativ hinzuzufügen. Gleiches gilt für Forms-Masken mit einer dreistufigen Master-Detail-Detail-Beziehung. Erfreulicherweise ist das Apex-Team bereits dabei, diese für komplexe Forms-Masken oft benötigten Funktionalitäten mit dem nächsten Release, Apex 4.1, bereitzustellen. Weitere Informationen zu Apex 4.1 finden Sie im sogenannten „Statement of Direction" unter http://bit.ly/be6N4K.

Ein Beispiel aus der Praxis

Die Union Investment Gruppe, einer der größten deutschen Asset-Manager für private und institutionelle Anleger, nutzt das Oracle-basierte System „OraDIS", das primär der verteilten Pflege von Fonds-Stammdaten dient.

Abbildung 5: … nach Apex 3.2

Vor der Migration kamen als Frontend MS Access 2007 sowie Oracle Forms 6*i* zum Einsatz. Eine Beendigung des Standard- und Extended Supports für Oracle Forms 6*i* seitens des Herstellers und das unzureichende Antwortverhalten von MS Access 2007 waren die maßgeblichen Auslöser für das Migrationsprojekt.

In einem „Proof of Concept" fand daraufhin Ende 2009 eine Realisierung mittels Apex statt (siehe Abbildung 3). Anschließend wurden im Zeitraum April bis August 2010 etwa 160 funktionale Dialoge, zu denen auch 60 Oracle-Forms-Masken gehörten, nach Apex 3.2 migriert. Seit Mitte August 2010 wird die Applikation durch 100 Endanwender produktiv verwendet.

Die Oracle-Forms-Masken waren von der Komplexität her noch recht überschaubar und ließen sich ohne große Schwierigkeiten in Apex umsetzen. Zwei Herausforderungen gab es dennoch: In Forms-Masken können die gleichen Felder auch für die Suche verwendet werden. Da es in Apex im Standard keinen integrierten „Suchmodus" gibt, wurden zusätzliche Suchfelder über den Bericht dargestellt, um eine ähnliche Funktionalität abzubilden. Außerdem gab es einige Masken mit einer „fixierten" Berichtsspalte. Wenn man horizontal scrollt, sollte die erste Spalte sichtbar bleiben. Dank einer Erweiterung auf Basis von jQuery konnte diese Funktionalität umgesetzt werden.

Eine Demo dazu steht unter http://portal.mt-ag.com/pls/apex/f?p=8000000013. Die Abbildungen 4 bis 7 zeigen das Ergebnis der Migration.

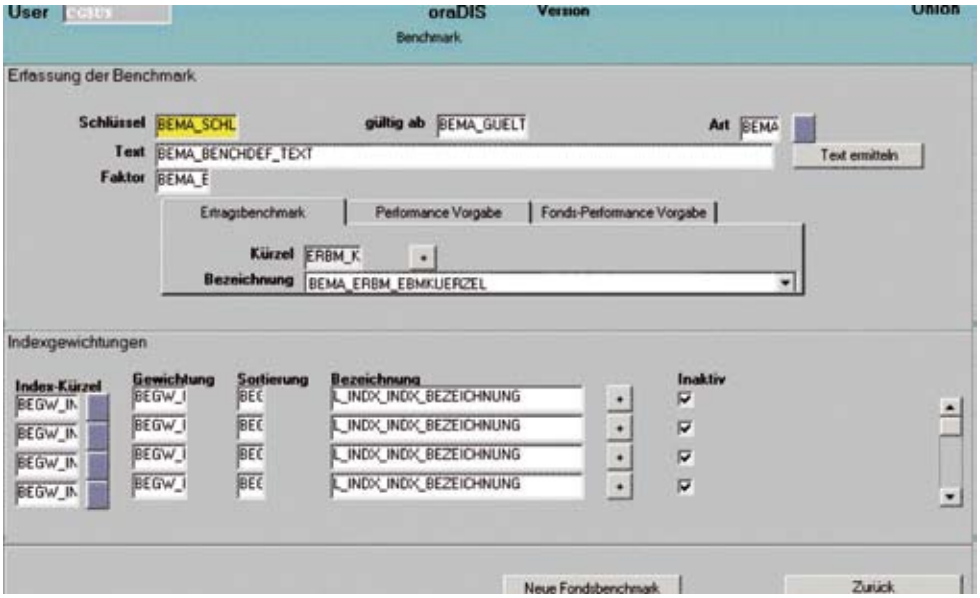

Abbildung 6: Von Oracle Forms 6i …

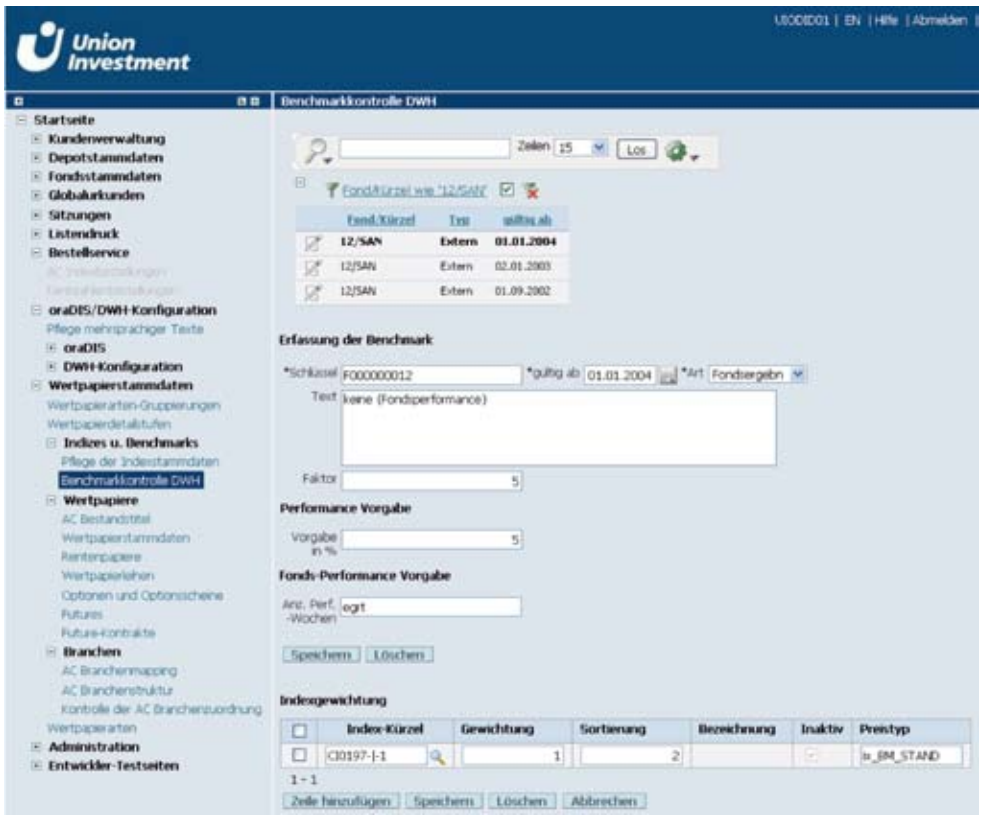

Abbildung 7: … nach Apex 3.2

Fazit

Sehr komplexe Forms-Masken sollte man mit Application Express in der Version 4.0 vorerst noch nicht migrieren. Die Version 4.1, die übrigens noch für das Jahr 2011 angekündigt ist, verspricht wesentliche Verbesserungen, die notwendig sind, um auch komplexe Masken deklarativ anzugehen. Einfache bis mittelkomplexe Forms-Masken lassen sich dagegen durchaus mit Apex 4.0 schnell und effizient umsetzen. Wenn man dabei hauptsächlich mithilfe der Apex-Assistenten vorgeht und sich an einige Richtlinien hält, so braucht man im Vergleich zu anderen Entwicklungsumgebungen bis zu 60 Prozent weniger Entwicklungszeit – Rapid Application Development eben.

Kontakt:
Niels de Bruijn
niels.de.bruijn@mt-ag.com