1000+ FMBs
- Forms Modernisierung zu Java –
Ein Praxisbericht

Roland Hörmann
SIB Visions GmbH
Wehlistraße 29 / Stiege 1 / 2.Stock, 1200 Wien

Schlüsselworte
Oracle Forms, Oracle Forms Migration, Oracle Forms Modernisierung, Java, Open Source, Rapid Application Development

Einleitung
In diesem Vortrag werden die Erfahrungen eines erfolgreichen Forms Modernisierungs Projektes, des deutschen Konzerns REMONDIS, in die Java Open Source Welt berichtet.
Wir zeigen mit Beispielen, von unterschiedlich komplexen Masken aus dem Projekt, in welchen Schritten und mit welchen technischen Mitteln die Masken in Java umgesetzt wurden. Wir behandeln die Umsetzung der mächtigen Forms Funktionen (LOVs, Master/Detail, Suche, Enter Query Mode, Repeating Frame, Forms Trigger) in der Java Welt.
Die automatische Überführung von GUI und Datenbindung, sowie die GUI Designer gestützte und teilautomatisierte Erstellung von hunderten von Masken wird aus der Praxis beleuchtet.
Bei diesem Praxisbericht gehen wir auch auf die Erfahrungen und Herausforderungen ein, die vom Forms Entwicklerteam - aufgrund des Kontaktes mit Java - gesammelt wurden.

Ausgangslage
Die REMONDIS AG mit 6,4 Mrd. Euro Umsatz und mehr als 30.000 Mitarbeitern in 35 Ländern in über 500 Standorten gehört zu den größten Unternehmen der Abfallwirtschaft. Im konkreten Projekt wird die zentrale Verwaltungsapplikation RUMS von Oracle Forms 6 Schrittweise nach Java migriert. Der Grund für eine Migration sind sowohl technische Probleme (Forms 6 läuft nicht mehr in allen Systemumgebungen problemlos) sowie insbesondere die Anforderung, dass zukünftig auch Kunden, Lieferanten und Partner die REMONDIS Systeme als Service nutzen können. Diese externen Services fordern Veränderungen an den Applikationen und man will damit auch gleich für moderne Web-Technologien vorbereitet sein. RUMS umfasst mehr als 1.500 Masken und läuft in 200 dezentralen Standorten in Europa. Je Standort ist eine Oracle Datenbank Instanz im Einsatz, welche von den Client/Server Forms angesprochen wird. Eine Migration nach ADF wäre mit sehr hohen Applikationsserver Lizenzkosten verbunden gewesen, daher wurde nach Alternativen gesucht.
Neben den Kostenaspekten sind die Benutzerakzeptanz und geringe Schulungsaufwände eine zentrale Anforderung. Dabei muss eine nahtlose Integration von Forms und Java Masken für die Benutzer gewährleistet sein. Im Zuge der Migration möchte man eine über 20 Jahre gewachsene Applikation auch technisch überarbeiten und für moderne Unternehmensservices vorbereiten bzw. umsetzen.

Analyse der bestehenden Forms
Am Projektbeginn Stand die Analyse der bestehenden Forms im Mittelpunkt. Dies wurde durch das Forms Kernteam gemeinsam mit Schlüsselpersonen des Java Entwicklerteams durchgeführt. Als Vorbereitung hatten die Forms Entwickler Java Schulungen besucht.

Im ersten Schritt wurde eine Liste mit wiederkehrenden bzw. speziellen Funktionen erstellt.

Auszug aus der Funktionsliste:

- Benutzerlogin über Datenbanktabellen mit spezifischer PL/SQL Funktion, welche die Rechte und Rollen des Benutzers ermittelt
- Hierarchisches Menü/Programmauswahl, gruppiert nach Modul und Bereich inkl. Suche
- Toolbar mit Neu/Löschen/Navigation zum aktuellen Forms Block
- Einheitliche Masken mit Header und Footer zur Anzeige von: Firmengruppe, Firma, Niederlassung, Betriebsstätte, …
- Tastaturbedingung für Schnellerfassungsmasken
- Suche & Bearbeiten / Enter Query Mode - F7/F8
- Mehrzeilige Tabellen / Repeating Frame
- Parameterübergabe zwischen Masken
- DB-Locking / Daten Manipulationsstrategien (CRUD)
- Mehrsprachigkeit
- Globales Datumsformat, Feldformatierungen
- Fehlende Foreign Keys / Primary Keys
- Zentrale Stammdatentabelle – LOVs
- LOVs bzw. Auswahl über Suchdialog
- Diverse Forms Trigger

Gemeinsam mit den Java Entwicklern und technischen Beratern wurde geprüft ob es für alle Funktionen eine praktikable Lösung gibt bzw. wie eine erstellt werden könnte. Dabei war im ersten Schritt relevant ob es überhaupt möglich ist, und im zweiten Schritt, wie kann dies mit wenig oder keinem Codier Aufwand umgesetzt werden. Dabei war wichtig die Funktionen zu vereinheitlichen und an einer zentralen Stelle zu lösen um den konkreten Aufwand in den Masken möglichst gering zu halten. Eine Auswahl an Lösungen wird später an Hand von RUMS konkret erläutert.

Im nächsten Schritt wurden die Masken kategorisiert um einen möglichst schnellen und effizienten Weg für die Migration festzulegen.
Kategorien:

1. Einfache Stammdatenmasken
   Masken mit beliebigen Master/Detailbeziehungen, LOVs, keine bzw. keine notwendigen GUI Forms Trigger
2. Mittelkomplexe Verwaltungsmasken
   Masken mit typischen Forms Funktionen, inkl. Forms Trigger mit Logik, teilweise in der Datenbank bzw. direkt in den Forms, viele Tabellen, Views, Tabs und Master/Details.
3. Komplexe Verwaltungsmasken
   komplexe Logiken bzw. Ausreizen von Forms Funktionen; mit JavaBeans/ActiveX usw.

Im konkreten Projekt fallen 94% der Masken in die Kategorie 1 + 2. Die Kategorien 1 + 2 lassen sich sehr rasch, automatisiert und mit wenig manuellen Aufwänden in Java umsetzen. Die Kategorie 3 Masken können zwar bezüglich GUI und Datenbindung schnell erstellt werden, aber die komplexen Logiken und Spezifikas müssen, mit Forms üblichen Aufwänden, manuell codiert werden.

Die Analyse ist ein Iterativer Prozess, welcher im ersten Schritt alle wichtigen Punkte beinhalten sollte (eigentlich muss) um dann Schritt für Schritt zu verfeinern. Die Verfeinerung erfolgt soweit wie möglich vor dem Entwicklungsstart, ist aber damit nicht abgeschlossen. In der Entwicklungsphase der vereinheitlichten zentralen bzw. spezifischen Funktionen sind zyklisch Anpassungen zu machen. In dieser Phase wurde immer mehr der Weg von einer 1:1 Migration zu einer praktischen Weiterentwicklung der RUMS Applikation gewählt. Hier ist nicht nur eine technische Entwicklung

vorangeschritten, sondern auch eine Änderung der Ansichten des Forms Entwicklerteams, hinsichtlich des Aussehens und der Funktionalität der RUMS Applikation.

Ein Migrationsprojekt ist nicht nur hinsichtlich der Technologie, sondern auch hinsichtlich der Benutzer und Entwickler ein schrittweiser und menschlicher Entwicklungsprozess.

Umsetzung in Java
Nach der Analyse Phase Stand vor allem die Entwickelung der zentralen und wiederkehrenden Funktionen im Fokus. Das verwendete Java Open Source Framework JVx bietet hier schon sehr viel Standard Funktionalität hinsichtlich der Entwicklung von komplexen Datenbank getriebenen ERP Lösungen an. Viele der sehr Forms spezifischen Funktionen inkl. der nahtlosen Integration von Java Masken in Forms Masken ist mit den JVx Forms Erweiterungen möglich. Damit müssen nur die RUMS spezifischen Funktionen möglichst zentral gelöst werden.

Der RUMS Applikationsrahmen mit Toolbar, Menü/Programmauswahl, RUMS Screen Basisklasse und weiteren Funktionen wurden vom JVx Framework Entwicklungsteam umgesetzt. Im Projekt wurde der Code an das RUMS Entwicklerteam übergeben. Dies war auch gleichzeitig der Startschuss für das Java & Forms Kernteam mit den zentralen und allgemeinen Funktionen von RUMS zu beginnen.

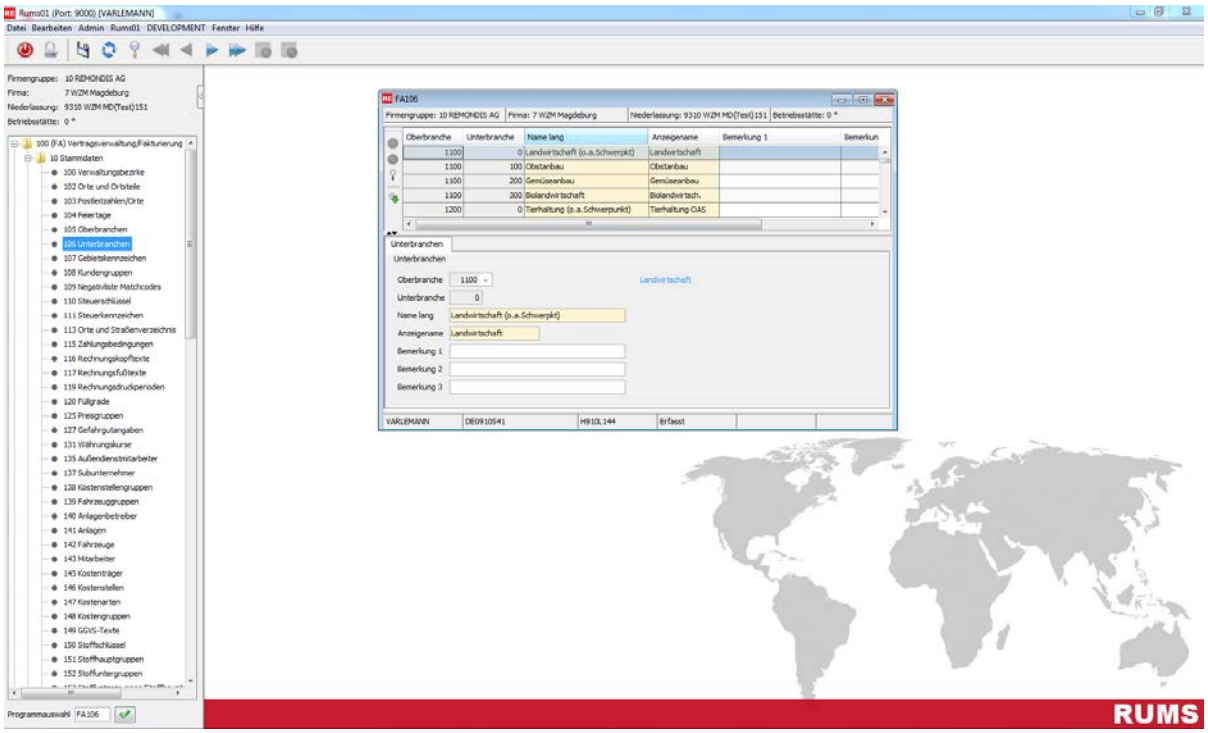

Abb. 1: RUMS Applikationsrahmen und Stammdaten Maske in Java

An Hand des Applikationsrahmens kam die Stärke von Java und JVx, mit der flexiblen Erweiterbarkeit gegenüber Forms, in der Praxis zur Geltung.

In RUMS ist das Menü zur Programmauswahl ein Baum, fixiert im linken Bereich der Applikation. Welche Programme dem Benutzer zur Verfügung stehen, ist sehr spezifisch und kann über eine bestehende PL/SQL Logik ermittelt werden. Für Adaptierung des Menüs wurde der Applikationsrahmen erweitert und eine einzige Funktion überschrieben. Die Ermittlung der Liste der Programme wurde in der Java Middleware durch den direkten Aufruf der PL/SQL Prozedur durchgeführt.

Um in RUMS ein einheitliches Screen Layout zu ermöglichen wurde die Screen Basisklasse von JVx erweitert und um die notwendigen Informationen wie Firmengruppe, Firma, Niederlassung, Betriebsstätte und andere Benutzer bzw. Client spezifische Parameter anzuzeigen. Die Informationen kommen direkt aus der Datenbank und von Java System Parametern (z.B. aktuelle angemeldeter Benutzer). Die Datenbankabfragen/SELECTs können mit JVx direkt in der Middleware durchgeführt werden, ähnlich zu Forms.

Die typische Block Navigation von Forms, in der Toolbar, ist bereits in den JVx Forms Erweiterungen enthalten und somit eine Standardfunktion. Das gleiche gilt auch für den so oft verwendeten Enter/Execute Query Mode (F7/F8) um eine Liste von Datensätzen zu filtern, oder den Einsatz von Repeating Frames um mehrzeilige Tabellen mit Editoren und Labels umzusetzen. Letzteres lässt sich über ein Java GUI Control umsetzten, in welches man einfach die gewünschten Elemente hinzufügt – wie aus Forms gewohnt, nur eben mit Java.

Eine weitere Anforderung ist, dass ausgewählte Masken auch für die Bedienung und Schnellerfassung per Tastatur umgesetzt werden können. Dies wurde durch einen Tab Index für die editierbaren GUI Controls umgesetzt, um die Navigation in einer bestimmten Reihenfolge zu ermöglichen.

Bei der konkreten Umsetzung der RUMS Oberfläche ist besonders wichtig, dass mit möglichst wenig Code, leicht verständliche, überschaubare Masken erstellt werden können. Manche GUI Funktionen müssen 1:1 migriert werden (Thema 30.000 Benutzer schulen), und die Masken müssen ein bestimmtes Verhalten erfüllen. Dafür wird eine praktikable Lösung benötigt ohne Wenn und Aber. In JVx können alle Funktionen flexibel und zentral an das spezifische Projekt angepasst werden. Dies ist sehr wichtig, weil RUMS damit einheitlicher und einfacher wird. Damit entsteht eine Java Lösung mit wenig Code, welche auch noch in 10 Jahren wart bar ist. Damit können zukünftige Änderungen zentral gelöst werden, anstatt in tausenden Forms die Funktionalität anzupassen.

Die Umsetzung von komplexen Masken, mit wenig Source Code, ist in der Java Welt absolut nicht der Standard. Diese Effizienz haben wir nur bei JVx gefunden. Bei ADF haben uns die XML Dateien und die Zeitaufwändigen großen Code Stücke um Abweichungen vom Standardverhalten umzusetzen, abgeschreckt.

Apropo - tausende Forms. Wir haben in Forms die Mehrsprachigkeit dadurch gelöst, dass wir unsere 1.500 Forms pro Sprache kopiert und abgeändert haben. Dies lösen wir in Java über eine XML Datei pro Maske bzw. einer zentralen Datei für globale Übersetzungen. Diese XML Dateien erstellen wir automatisiert aus unseren Übersetzungstabellen der Datenbank. Die Verwaltung erfolgt über Java Masken.

Datenbindung und Persistenz in Java

Eine der Kernanforderung für Forms Entwickler ist die Anbindung der Masken an die Datenbank. Die Möglichkeiten von Forms sind hier zum Unterschied in der Java Welt, viel effizienter, mächtiger und sehr Datenbank nahe. In Java wird bei der Entwicklung immer von einem Objektmodell ausgegangen und Datenbanken werden eher nebensächlicher betrachtet. JVx bietet in diesem Bereich eine moderne Java Persistenz mit der Effizienz und Mächtigkeit von Forms an.

Damit lassen sich Java übliche Multi-Tier Applikationen erstellen, welche mit der einfachen Angabe des Tabellen- oder Viewnamen an GUI Elemente gebunden werden können. Dabei entsteht extrem wenig Code und für Forms Entwickler ist es schnell verständlich. Das Lesen und Ändern der Daten über beliebige Master/Detail Ebenen wird automatisch in JVx gelöst. Dabei kann zwischen verschiedenen Lese- /Schreibmechanismen gewählt bzw. diese adaptiert werden. Für RUMS ist die Rückschreibestrategie „Speichern bei Zeilenwechsel“ oder „Schließen der Maske“ ausreichend. In ausgewählten Masken werden alle Datenänderung beim Verlassen der Maske geschrieben oder verworfen (über mehrere Tabellen hinweg). Für diese Masken wird der Master Datensatz gelockt, sobald die erste Änderung durch den Benutzer in einem Eingabefeld durchgeführt wird. Damit erreichen wir die Transaktionssicherheit und das falls ein weiterer Benutzer dieselben Daten bearbeiten möchte, wird er über die Datensatzsperre informiert.

Das sehr praktische Forms Trigger Konzept lässt sich mit JVx sowohl auf der Client Seite (Präsentationsschicht), als auch auf Applikationsserver Seite (Business Logik Schicht), als auch mit Oracle Datenbanktrigger über allen Schichten hinweg umsetzen z.B.: Pre, Post Insert/Update/Delete (= before/after CRUD).

In RUMS werden für die Auswahl von einem Datensatz, anstatt von LOVs oft Forms typische Auswahldialoge verwendet. Im Zuge der Analyse wurde entschieden die weitaus flexiblere LOV Standard Funktionalität von JVx einzusetzen. LOVs werden in JVx automatisch auf Grund des Datenmodells ermittelt. Wenn es einen ForeignKey zur Stammdatentabelle gibt, wird diese automatisch als LOV hinzugefügt. Damit brauchen, keine spezifischen SELECTs auf die Mastertabelle mit join auf die aussagekräftigen Stammdatenspalten durchgeführt werden, um diese Spalten zu ermitteln. In Summe fällt weitaus weniger wiederkehrender Code in den Masken an.

Für die typische zentrale Stammdatentabelle, welche alle Forms LOVs beinhaltet, kann Datentechnisch kein ForeignKey Constraint zu den nutzenden Tabellen angelegt werden. Dies ist aus historischen oder Performancegründen auch noch an anderen Stellen im Datenmodell zu finden. Für diese kann mit einer Zeile Java Code, die Verknüpfung für LOVs oder Master/Details festgelegt werden.

Forms und Java integriert

Eine Kernanforderung für dieses Projekt ist die Schrittweise Migration und der nahtlosen Integration von Forms und bereits umgestellten Java Masken.
Das JVx Forms Erweiterungspaket einhält die Integrationsfunktionen von bestehenden JVx Masken in Oracle Forms 6, 10g und 11g. Ab 10g erfolgt die Integration über JavaBeans direkt eingebettet. In unseren Fall (Forms 6) wird beim Start der Forms automatisch die Java Client Applikation im Hintergrund gestartet. Sobald eine Maske aus dem Menü der RUMS Forms Applikation gestartet wird, wird der entsprechende Screen in der Java Applikation geöffnet und der Fokus wird auf die Java Applikation gewechselt. Analog funktioniert dies von der Java Applikation zu einer Forms Maske.
Um dies zu ermöglichen kommuniziert die Forms mit Java über ein ActiveX Control in der Form. Beide Technologien können Parameter beim Screen Öffnen übergeben (bzw. aneinander kommunizieren), kennen die Applikationsübergreifenden Informationen und können Ereignisorientiert interagieren.

GUI Designer, Teilautomatisierte Migration
Um die Migration von 1.500 Forms effizient über die Bühne zu bringen, muss ein automatisierter Weg eingeschlagen werden.

Wer sich mit dem Thema schon etwas befasst hat, weiß das eine vollautomatische Migration unmöglich ist. Selbst teilautomatisierte Migrationen führen oft in eine sehr große Menge an unübersichtlich, maschinell erzeugten Code, der nicht mehr wart bar ist.

Wie kamen wir zu einer effizienten und praktikablen Lösung für RUMS?

Die RUMS Masken teilautomatisiert zu migrieren, haben wir im ersten Schritt verworfen und nach einem Alternativen weg gesucht. Die JVx Entwickler bieten neben Ihrem Framework auch einen visuelles Drag & Drop Entwicklungswerkzeug Namens VisionX an. Neben vielen anderen Funktionen können sehr effizient Masken auf ein bestehendes Datenmodell erstellt werden. Vor allem bei Datenmodellen mit Primary/Foreign Keys können Wizard gesteuert sehr schnell Masken erstellt werden. Leider passte das Layout der Screens nicht mit unserer Erwartungshaltung zusammen. Ein paar Gespräche später, wurden uns Erweiterungsmöglichkeiten zugänglich gemacht und dokumentiert. Damit konnten wir die Erstellung von Screens auf unsere Bedürfnisse anpassen um die bestehenden Screens mit wenigen Klicks zu erstellen. Damit kamen wir der teilautomatisierten Migration schon sehr nahe, aber der noch viel wertvollere Effekt ist, das alle neuen Funktionen nach der gleichen einheitlichen Methodik einfach erstellt werden können. Der Schlüssel bei umfangreichen und komplexen Applikationen wie RUMS ist die vereinheitlichte und zentrale Lösung der wiederkehrenden Funktionen. Forms verleitet leider sehr zu konkreten Lösungen in jedem Screen. Durch die Größe und der langen Historie der meisten Forms Lösungen, ist viel solcher beinahe gleicher Code entstanden, der bereinigt werden sollte. Solche Aspekte sprechen auch wieder gegen Teilautomatisierte Code Generierung.

Eine teilautomatische Migration ist in Bezug auf das GUI Layout und die Datenbindung inkl. Persistenz eine Interessante Möglichkeit, welche wir im Rahmen des RUMS Migrationsprojektes noch unter die Lupe nehmen werden. Dies ist vor allem für einfache Masken bzw. Masken mit vielen Eingabefeldern und Tabellen, welche wir 1:1 übernehmen möchten von Vorteil. Die Frage die sich uns gestellt hat war, wie viel müssen wir nachbearbeiten bzw. können wir die Generierung an unsere Bedürfnisse anpassen. Die Übersichtlichkeit des Codes und die wenigen Code Zeilen haben uns in den ersten Tests überzeugt.

Forms Entwicklerteam – Der Weg in die Java Welt
Ich kann es gleich vorwegnehmen, die Java Syntax war sicherlich nicht die Herausforderung in diesem Projekt. Forms ist sicherlich eines der effizientesten 4GL Werkzeuge für die Erstellung von Datenbankapplikationen. Eine Charakteristik einer Forms Anwendung ist eine lange Liste an Items und Properties umgeben von Datenbank nahen Prozeduralen Code. Es können zwar die Funktionen auslagern und gruppiert werden, aber zur Objektorientierten Programmierung ist es trotzdem noch ein recht ordentlicher Schritt. Dies kann man zwar in Java Kursen lernen, aber die praktische Anwendung ist dann wieder ein anderes paar Schuhe. Damit sind vor allem die in Java üblichen Methoden und Vorgangsweisen (Patterns) Neuland, das einfach etwas Zeit braucht. Gemeinsam mit den Java Entwicklern im Haus, als auch mit den Entwicklern des JVx Teams ist dies eine machbare Aufgabe.

Ein Migrationsprojekt ist nicht nur hinsichtlich der Technologie, sondern auch hinsichtlich der Entwickler ein schrittweiser und menschlicher Entwicklungsprozess. Das Schöne daran ist, wenn sich die ersten Erfolge einmal eingestellt haben, glaubt man kaum welche Motivation und Elan von den Forms Entwicklern in einem solchen Projekt entstehen. Das bringt RUMS nach 10 Jahren Stillstand in die Zukunft.

Kontaktadressen:

Dirk Varlemann
REMONDIS IT Services GmbH & Co. KG
Brunnenstr. 138
44536 Lünen
Deutschland

Telefon: +49 2306 106-147
Fax: +49 2306 106-165
E-Mail dirk.varlemann@remondis.de
Internet: www. remondis.de

Roland Hörmann
SIB Visions GmbH
Wehlistr. 29 / Stiege 1 / 2.Stock
A-1200 Wien
Österreich

Telefon: +43 1 934 6009 616
Fax: +43 1 934 6009 999
E-Mail roland.hoermann@sibvisions.com
Internet: www.sibvisions.com