

Von Oracle Forms zu Oracle ADF und J2EE

Modernisierung von Oracle Forms
Applikationen auf das Oracle Application
Development Framework und die J2EE
Architektur

PITSS.CON 8.0.0

White Paper, Juni 2009

Einführung	3
Warum J2EE? Warum ADF? Was spricht für einen Wechsel?	3
Die Herausforderung einer Neuerstellung.....	4
Kompetenzen	4
Anwenderoberfläche	5
Business Logik	5
Die PITSS.CON Lösung	6
Prozess	8
Wirtschaftliche Vorteile	10
Zusammenfassung.....	12

Einführung

Oracle bietet heutzutage ein umfassendes Sortiment an Entwicklungslösungen und Technologien. Die technischen Möglichkeiten zur Erstellung neuer Anwendungen sind nahezu unbegrenzt. Doch was geschieht mit bestehenden Legacy Applikationen basierend auf Oracle Forms? Lassen sie sich in die neue J2EE Architektur integrieren? In welchem Umfang lässt sich Nutzen aus den neuen Technologien ziehen und sich gleichzeitig das kostspielige und riskante Unterfangen vermeiden, Applikationen ganz neu zu schreiben? Mit diesem Dokument soll näher auf diese Fragen eingegangen werden, und es soll ein Überblick sowohl über die Chancen als auch über die Herausforderungen geboten werden, die mit einem solchen Wechsel der Software Architektur einhergehen. Ferner sollen die Perspektiven und Lösungsansätze diskutiert werden, die die Produktpalette von PITSS.CON wie Forms2ADF und Forms2SOA mit dem Business Logic Assistant sowie dem Web Service Assistant bieten.

Doch zunächst einmal:

Warum J2EE? Warum ADF? Was spricht für einen Wechsel?

Das Internet ändert unsere Geschäftsabläufe wie nie zuvor, und damit auch die Anforderungen, die die Anwender an die Software haben. Die meisten Applikationen, die wir im Laufe der vergangenen Jahre mit Forms entwickelt haben, lassen sich durch geeignete Werkzeuge meist kostengünstig auf die neuste Version Forms anheben und damit web-fähig machen. Es gibt jedoch Szenarien im Geschäftsabläufen, die durch eine webbasierte Architekturen noch besser erfüllt werden können, wie: reichhaltiger gestaltete Anwenderoberflächen, Datenhaltung in der Logikschicht usw. Für solche Szenarien empfiehlt Oracle, die Applikation gemäß der J2EE Architektur neu zu schreiben und mit den restlichen Modulen auf dem Anwendungsserver zu integrieren.

„[...]Oracle empfiehlt Kunden, ihre bestehenden Forms and Reports Applikationen als Internet-Anwendung zu betreiben und dabei die Möglichkeiten einer Neuentwicklung in J2EE mit JDeveloper und ADF in Erwägung zu ziehen. Diese Applikationen sollen auf dem Anwendungsserver zusammengeführt werden, um so effizient gemeinsame Dienste und Business Logik zu nutzen.“¹

J2EE verspricht auf lange Sicht eine leichtere Pflege der Applikationen aufgrund von offenen Standards und vereinfachter Integration mit anderen Produkten. Allerdings erscheint J2EE den Datenbankentwicklern trotz der Vorteile komplex und schwierig. Oracle Forms Entwickler (oder allgemeiner 4GL Entwickler) sind eine äußerst hohe Produktivität gewohnt, die sich mit J2EE so nicht erreichen lässt. Hier ist es schwierig, Kunden die gleich kurzen Entwicklungszeiten zu bieten, die sie gewohnt sind. Aus diesem Grund empfiehlt Oracle das Oracle Application Development Framework (ADF) um die J2EE Design Pattern (Model View Controller, MVC) einfacher zu implementieren. Mit seiner visuellen, selbsterklärenden Umgebung erlaubt ADF schneller Code zu schreiben und die Entwicklungszeit zu reduzieren. Aber ist das alles wirklich so einfach, wie es scheint?

¹ Oracle Forms – Oracle Reports – Oracle Designer: Statement of Direction, Oracle, October 2008

Die Herausforderung: Neuentwicklung

Das Rezept für ein erfolgreiches Re-Engineering Projekt von Forms nach ADF lautet:

- 1) **Stellen Sie sicher, dass Sie über ein sehr gutes Verständnis der bestehenden und neu zu erstellenden Forms Applikation in ihrer gesamten Komplexität verfügen** – Legacy Applikationen wurden vor langer Zeit entwickelt und haben oftmals eine erhebliche Größe und Komplexität erreicht. In den vielen Fällen sind die damaligen Forms Entwickler nicht mehr im Unternehmen und notwendige Dokumentationen unzureichend oder überhaupt nicht vorhanden. Die Java Entwickler, die für die Neuerstellung der Anwendung zuständig sind, tun sich oft schwer die Komponenten, deren Verknüpfungen vollständig zu verstehen, und benötigen u. U. Schulungen zum Verhalten der laufenden Forms-Anwendung.
- 2) **Eignen Sie sich umfassende Kenntnisse der neuen Technologien an und überführen Sie den bestehenden Funktionsumfang in die neue Architektur bei gleichzeitiger Verbesserung** – dies ist eine anspruchsvolle Aufgabe, weil Forms und J2EE von Grund auf unterschiedlich sind. Darüber hinaus ist die Migration der Applikation nur dann erfolgreich, wenn die neue Version anwendungsfreundlicher ist und über bessere technische Fähigkeiten verfügt, wie z.B. eine intuitivere Oberfläche. Es sollte schließlich nicht das Ziel sein, dass die hohe Investition in das Modernisierungsprojekt, lediglich eine exakte 1 zu 1 Kopie der ursprünglichen Anwendung hervorbringt.
- 3) **Etablieren Sie ein Projektmanagement, um den Fortschritt der Migration zu planen und zu überwachen** – Software Projekte, auf Grundlage neuer Technologie, scheitern oft an unvorhergesehenen Problemen, an Erfahrungsmangel und an fehlenden Implementierungsstandards. Es empfiehlt sich das Projekt kontinuierlich und gewissenhaft auf bekannte Beschränkungen in Hinblick auf Zeit, Budget und Qualität zu überwachen, um den Erfolg von Beginn zu gewährleisten.

Hinter diesen drei einfachen Zutaten werden die wirklichen Herausforderungen sichtbar:

Kompetenzen

Gemischtes Kompetenzprofil: Kenntnisse sowohl von PL/SQL und Forms als auch von Java, JavaScript, XML und J2EE sind für ein erfolgreiches Projekt unverzichtbar. Java Kenntnisse dürften auf dem Markt einfach zu

finden sein, aber der ausschlaggebende Faktor ist das Verständnis sowohl der Forms Komplexität als auch der neuen Technologie samt dessen Zielarchitektur.

Anwenderoberfläche

Anpassung an die Stärken und Schwächen des Internet Modells: im

Gegenzug zur höheren Benutzerfreundlichkeit weisen browserbasierte Applikationen auch gewisse Einschränkungen auf. Oracle Forms bietet eine äußerst dateneffiziente Architektur, die es möglich macht, in einem Applet Tausende von Datensätzen abzufragen, Hunderte von Feldern auf einer Seite anzeigen, auf Client Rechner zuzugreifen sowie Dateien via Host Anweisungen zu bearbeiten. Daher muss eine typische Forms Applikation umgestaltet werden, um in einem einfachen Browser zu laufen – etwa, das gruppenweise Anzeigen von je 20 Datensätze, die hundert Felder auf verschiedene Seiten zu verteilen oder die Zugriffe auf den Client Rechner zu verringern.

Applikationen, die sich nicht auf diese Weise anpassen lassen, sollten in durch einen Upgrade auf die neueste Forms Version angehoben werden, bis die neue Technologie dies in geeigneter Form unterstützt.

Business Logik

Forms Anwendungen besitzen oft eine eng verzahnte Logik, die innerhalb eines Code Segments beispielsweise Business Logik mit Aktionen der Anwendungsoberfläche, Datenvalidierung, Datenbearbeitung usw. vereint. Bei der Übertragung eines solchen Code Segments nach ADF muss **die Logik in ihre Komponenten aufgetrennt** und nach dem MVC Design Patter neu definiert werden, damit eine echte ADF Architektur entstehen kann.

```
if p_dept_id < 0 or p_dept_id > 99 then
  message('Invalid department number');
else
  if pac_util.check_dept(p_dept_id, p_dept_name, p_location_id) then
    insert into dept (id, name, location_id)
      values (p_dept_id, p_dept_name, p_location_id);
  end if;
  go_block('dept');
end if;
```

Abbildung 1: gemischte Business Logik am Beispiel eines Triggers in einer Forms Applikation

Lösungsansätze für die Logik sind:

- 1) **Umsetzung des gesamten Codes in Java** – ein solcher Ansatz wäre nur dann erfolgreich, wenn der Code zu 100% von Hand umgeschrieben würde. In einem solchen Fall ginge die gesamte Investition in die Forms Business Logik verloren. Es gibt zwar Tools auf dem Markt, die eine automatische Konvertierung des gesamten Oracle Forms Codes nach Java versprechen, aber der so erzeugte Code ist sequenziell und

„Heutzutage hat man ein Vielzahl von Möglichkeiten Software zu schreiben, die mit einer Oracle Datenbank arbeitet. Sie können Java und JDBC verwenden oder Visual Basic und ODBC, oder Sie können sich für Delphi, C++ usw. entscheiden. Sie werden jedoch

prozedural wie PL/SQL und nicht objektorientiert, wie Java Code eigentlich sein sollte. So generierter Code enthält das gleiche Gemisch wie der ursprüngliche Code, bestehend aus Datenverarbeitung, Business Logik, Aktionen der Anwenderoberfläche oder gar Runtime-Befehle, die zu eigentümlichen, komplexen Java-Konstrukten führen. Dieser Code ist schwer verständlich und erfüllt in den meisten Fällen nicht die Erwartungen an Java konformer Code. Eine Pflege des Java-Codes ist selbst nach Bereitstellung der proprietär eingebrachten Elemente kaum denkbar.

Der Hauptgrund, warum dies nicht als der geeignete Lösungsansatz erscheint, liegt jedoch darin, dass wir hier von Datenbankanwendungen ausgehen müssen, deren Business Logik direkt auf große Datenmenge angewendet wird somit ist ein Großteil der Logik weit besser in der Nähe der Daten in der Oracle Datenbank aufgehoben als in der Präsentationsschicht.

- 2) **Beibehaltung der Business Logik in PL/SQL und Ablage in der Datenbank** – der Code für Datenzugriffe oder Datenbearbeitung ist in der Regel in der Datenbank schneller und reduziert dort den Datenverkehr im Netzwerk auf ein Minimum. PL/SQL hat sowohl in Oracle Forms Applikationen als auch in der Datenbank alle im Laufe der Jahre angefallenen geschäftlichen Anforderungen gemeistert und wird das auch in Zukunft in gewohnt zuverlässiger, performanter Weise tun. Ein solcher Ansatz stellt eine enorme Risiko-Minimierung dar. Migriert man nach J2EE, dann sind natürlich nicht alle Code Segmente zur Verwendung in der Datenbank geeignet, wie Code für die Anwenderoberfläche, Navigationscode und Feld-Validierungen. Diese sind näher an der Präsentationsschicht.
- 3) **Die kombinierte Strategie:** Code für die Anwenderoberfläche in Java, Datenbearbeitung in der Datenbank. Dies ist die Lösung, die wir klar empfehlen und im folgenden Abschnitt näher erläutern werden.

Die PITSS.CON Lösung

Unser Lösungsansatz vereint die umfassenden Analysefähigkeiten der PITSS.CON Enterprise Edition EE zum Erlangen eines detaillierte Verständnisses der Architektur der Oracle Forms und Reports Applikation mit leistungsfähigen Re-Engineering Funktionen wie „Application Analysis“ und „Application Engineering“, den Forms2ADF Assistant, den Business Logic Assistant bis hin zu dem Web Services Assistant.

PITSS.CON lädt die Forms und Reports Applikationen samt Datenbank-Schemen ins Repository, parst die Programmstrukturen und den gesamten

bemerkten, dass das Schreiben von hocheffizientem Code für den Zugriff auf die Oracle Datenbank in keiner anderen Sprache so einfach ist wie in PL/SQL.“²

„Ein PL/SQL Programm mit effektivem Cursor Handling kann um ein Vielfaches schneller als ein Java Programm ausgeführt werden, das für die gleiche Aufgabe geschrieben wurde und auf einem Applications Server läuft.“³

² Oracle PL/SQL Programming, Fourth Edition, Steven Feuerstein with Bill Pribyl, 2005
³ Oracle PL/SQL for Dummies, Michael Rosenblum, Paul Dorsey, 2006

Code und erstellt daraus feingranulare Metadaten mit allen applikationsrelevanten Objekt und deren Abhängigkeiten.

Dank dieser tiefen Kenntnis der Applikation kann PITSS.CON die Anzahl der Forms Komponenten enorm erhöhen, die in natürlicher Form in die ADF Welt überführt werden.

Architektur

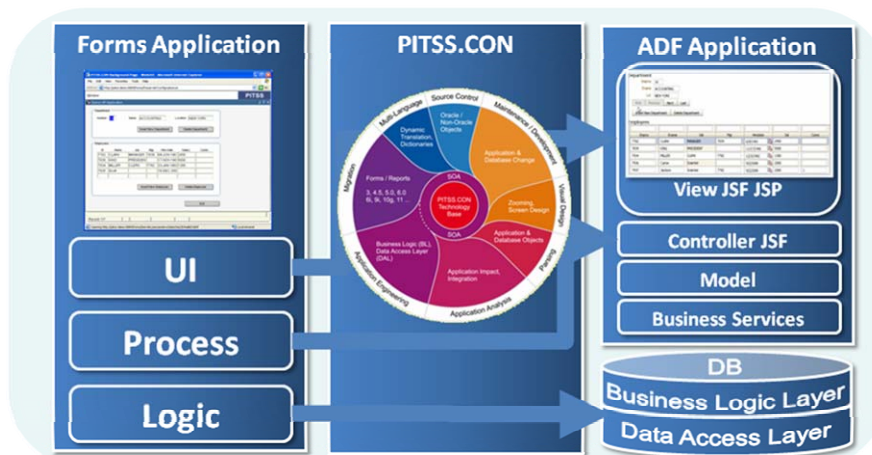


Abbildung 2: Die Architektur des Pitss Forms2ADF Assistant

Der größte Teil der Business Logik in Forms gilt der Bearbeitung von Daten. Diesbezügliche Logik gehört selbstverständlich in die Oracle Datenbank. PITSS.CON unterstützt die Transformation der datenintensiven Business Logik in die Datenbank und erzeugt dort einen Business Logik Layer (BL Layer). Nach der Transformation des Codes von Forms and Reports in die Datenbank ist diese BL Schicht nicht mehr nur den Oracle Forms and Reports Programmen zugänglich, sondern auch von Oracle ADF und Oracle APEX aus ausführbar, oder kann als Web Service WS angeboten werden. Damit ist die neugeschaffene BL von anderen Technologie verwendbar und kann hinter unterschiedlichsten Oberflächen, BEPL Prozessen und innerhalb des Enterprise Service Bus ESB seine Aufgaben wahrnehmen. Dies stellt nicht nur einen enormen Schutz Ihrer Investitionen dar, sondern steigert nachhaltig den Wert, durch Wiederverwendbarkeit und Rückführung redundanten Codes zu zentralen organisierten Funktionen. Reduzierung der Wartungsaufwände, effiziente Weiterentwicklung, Oberflächenunabhängigkeit sind weitere positiv begleitende Merkmale dieses Ansatzes. Zukünftigen Anpassungen oder Modernisierungsvorhaben lassen sich leichter realisieren.

Das Ablegen der Business Logik in der Datenbank, unterstützt den gesamten langläufigen Migrationsprozess in optimaler Weise, da der neu erzeugte, neu strukturierte und effizientere Code sowohl in der Forms wie auch in der entstehende ADF Applikation zum Einsatz kommt. Das Schrittweise migrieren führt zu besser getestetem Code, minimiert die Migrationsrisiken und beschleunigt so den Gesamtprozess. Ein weiterer Aspekt dieses Ansatzes ist

die Implementierung einer serviceorientierten Architektur (SOA), denn im gesamten SOA Konzept geht es gerade um die Wiederverwendung und Integration von Komponenten.

Die noch verbleibenden Forms Komponenten werden durch den ADF-Assistent in das ADF MVC Pattern basierend auf ADF Faces, JSF und ADF Business Components überführt.

Prozess

- 1) **Applikationsanalyse** – der erste Schritt besteht darin, sämtliche Applikationen (FMB, PLL usw.) sowie deren Datenbank-Schemen ins PITSS.CON Repository zu laden, um eine genaue Analyse durchzuführen. In dieser Phase wird die Migrationsstrategie festgelegt, die sich nach der Beschaffenheit und der Komplexität der jeweiligen Applikation sowie den Anforderungen an die Oberfläche und dem Ausmaß der Anpassung an die WEB Technologie richtet.
- 2) **Dead Code Analysis** – der nächste Schritt ist die Bereinigung der Applikation von nicht genutztem Code, so dass nur noch die Kernfunktionen erhalten bleiben. Dies ist der ideale Ausgangspunkt, um wieder die Kontrolle über die bestehende Applikation zurückzugewinnen, die Applikation auf das notwendige Minimum zurückzufahren und um eine tragende Dokumentation für die Reise in die Modernisierung zu erstellen. Unserer Erfahrung nach stellt sich in diesem Schritt heraus, dass 20–30% der Applikationsobjekte redundant sind, ungenutzte Tabellen, Bibliotheken, mehrfach vorhandene oder ungenutzte Code Abschnitte oder sogar komplett obsolete Funktionen, wie beispielsweise Kalenderfunktionen für ein Datumsfeld, die in der neuen J2EE Umgebung nicht mehr benötigt werden.
- 3) **Migration von Business Logik in die Datenbank** – dieser Schritt ist die Grundlage für die erfolgreiche Modernisierung hin zu einer serviceorientierten Architektur. Hierbei lässt sich mit den folgenden Hilfsmitteln viel Zeit und Geld sparen:
 - **BL Assistant** – extrahiert Dienste aus dem Forms and Reports Code ausgehend von einer Forms-spezifischen Aktion, wie dem Anklicken einer Schaltfläche (z.B. WHEN-BUTTON-PRESSED). PITSS.CON konstruiert die Code Kette, aus der sich ein Dienst, eine Funktion oder ein Prozess zusammensetzt, und zeigt visuell auf, inwieweit der Dienst in die Datenbank migrierbar werden kann. Der BL Assistant führt dann die Transformation durch und setzt den neu erzeugten Code wieder der ursprünglichen Applikation ein. Die so entstandenen Datenbankdienste stehen nicht nur jeder Forms Applikation zur Verfügung, sondern auch ADF Applikationen, und lassen sich sogar als Web Service noch breiter verfügbar machen.



- **Web Service Wizard** – stellt jede gespeicherte Funktion als Web Dienst zur Verfügung und öffnet die Applikation damit für der Außenwelt.
 - **DAL Assistant** – modernisiert eine „table based“ Applikation und setzt sie auf „procedure based“ Zugriffe um. Dabei erzeugt der DAL Assistant eine Schicht in der Datenbank, die wie die BL Schicht einfachst in anderen Technologien integriert werden kann. Der Datenzugriff kann anschließend über die erzeugten Dienste gesteuert werden. Diese Schicht lässt sich später erweitern, um auch komplexe Geschäftsregeln und Sicherheitsvorkehrungen aufzunehmen.
 - **Application Analysis und Application Impact** – ermöglichen Ihnen, schnell und präzise auszuwerten, welche Auswirkungen eine geplante Änderungen hat, nicht verwendeten oder problematischen Code zu erkennen und den Übergang in eine serviceorientierte Architektur reibungsloser zu gestalten.
- 4) **Toolgestütztes Re-Engineering von Applikationen** im JDeveloper ADF – nachdem die Applikation um den Großteil der darin enthaltenen Business Logik bereinigt wurde, kann die Nachbildung der Anwenderoberfläche nahezu auf Knopfdruck bewerkstelligt werden. Der Forms2ADF Assistant erzeugt automatisch die dazugehörigen ADF Objekte:
- Geschäftsfunktionen mit ADF Business Components;
 - Datenobjekte, auf die von mehreren ADF Applikationen aus zugegriffen werden kann, als Unterstützung für einfacheres Aufrufen der gespeicherten Business Logik von der ADF Applikation aus;
 - Abschluss der Model-, View- und Controller-Schichten MVC, Analyse aller Komponenten der Forms Applikation – Fenster, Canvas-Elemente, Blöcke, Objekte usw. und Erstellung der dazugehörigen ADF-Objekte unter Berücksichtigung jeder einzelnen Objekteigenschaft und dessen Größe, in dem dieses Objekt im ADF angelegt wird;
 - Generierung von Java Methoden für die Benutzeroberfläche durch die Übersetzung der PL-/SQL-Business Logik in Java Code, wobei automatisch die Zuordnung von Variablen und das Aufrufen von gespeicherter Business Logik über die Datenzugriffsobjekte implementiert wird.

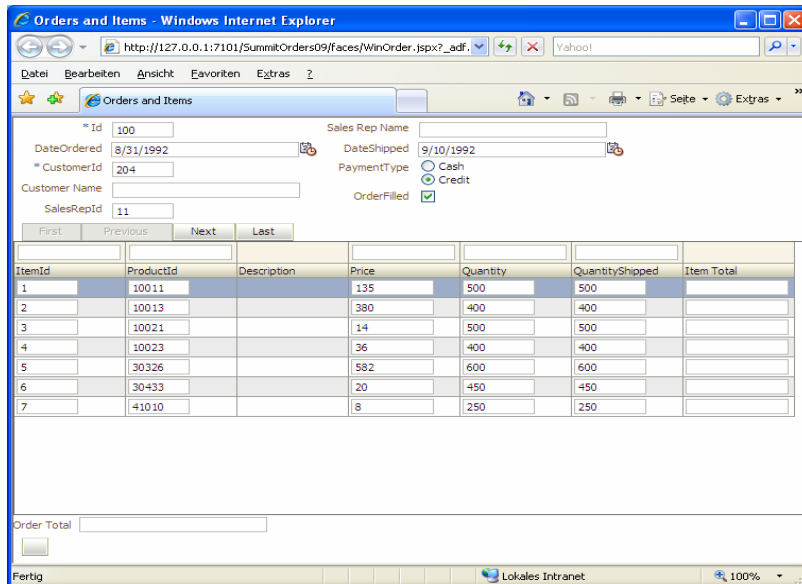


Abbildung 3: Beispiel einer vom Pitss Forms2ADF Assistant automatisch generierten Seite

5) **Nachbearbeitungs-Prozess** – Das Ergebnis des zuvor geschilderten Prozesses wird ausführlich und umfassend dokumentiert. Diese Dokumentation ist eine hervorragende Basis für Entwickler, die an der konvertierten Applikation arbeiten, um Schätzung hinsichtlich des noch benötigten Arbeitsaufwands für Feinabstimmungen zu gewinnen. In der Regel sind manuelle Anpassungen nach der Erstellung für die folgenden Aufgaben möglich:

- Anpassung des Weblayouts – falls gewünscht, können die mit dem Tool erstellten Seiten mit JDeveloper weiter verfeinert werden
- Neuordnung der Seitennavigation mithilfe vorgefertigter Komponenten
- Verfeinerung und Aktivierung der vom Tool übersetzten Java Methoden
- vollständige Definition der Komponenten, die sich nicht 100% toolgestützt generieren ließen, oder das Anbieten von Lösungen, mit denen sich die Beschränkungen des Web-Technologie umgehen lassen (beispielsweise Client Dateisystem oder Zugriff auf die Registry)

Wirtschaftliche Vorteile

Durch unsere jahrzehntelange Erfahrung als Anbieter von Modernisierungsleistungen und -werkzeugen für Oracle Forms und Reports Applikationen können wir sehr gut die Komplexität einer Umstellung von Oracle Forms auf J2EE einschätzen. Wir bekommen bestätigt, dass unser Ansatz die optimale und gleichzeitig die einzig praktikable Lösung darstellt, die echte Zeit- und Kosteneinsparungen gepaart mit Risikominimierung mit sich bringt.

- **Produktivität** – viel effizienter im Vergleich zur manuellen Neuerstellung durch automatisierte, regelbasierende Durchführung ähnlicher Vorgänge und weitestmögliche natürliche Integration und Wiederverwendung von Forms Komponenten in J2EE.
- **Effektivität** – sehr großer Funktionsumfang, der in ADF automatisch nachgebildet wird. Wir bieten nicht nur eine Lösung für die Benutzeroberfläche, sondern PITSS.CON stellt Ihnen auch herausragende Werkzeuge zur Analyse von Business Logik und Unterstützung bei der Transformation in die Datenbank zur Verfügung.
- **Konsistenz** – die Anwendung leistungsfähigen Entwicklungsverfahren führt zu einem einheitlichen Look-and-Feel, konsistenter Namensvergabe, Codier-Methoden sowie besserer Wartbarkeit.
- **Natürliche Architektur** – wir streben danach, automatische Lösungen anzubieten, mit denen Anwendungen erstellt werden können, die J2EE konform, leicht verständlich und einfach weiterzuentwickeln sind, sich durch ein natürliches Design auszeichnen und nicht von Anwendungen zu unterscheiden sind, die von Hand neu geschrieben wurden.
- **Keine proprietären Komponenten** – die erzeugte ADF Anwendung ist reiner Quellcode, frei von jeglicher proprietärer Software, DLLs usw. und kann weitergegeben und verbreitet werden.
- **Unterstützung für Entwickler**
 - **für Java Entwickler:** Hilfe beim Erlangen einer tiefgreifenden Kenntnis der Forms Applikation und deren Komplexität durch detaillierte Berichte und umfassende Analysen.
 - **für Forms Entwickler:** einfacherer Übergang auf J2EE durch Schritt-für-Schritt-Ansatz, Leitfäden und Coaching durch unsere Berater. Formsentwickler bringen ihr Knowhow direkt in den Migrationsprozess ein.

Basis für ein erfolgreiches Projekt, durch die Mitnahme und Motivation sämtlicher Applikationsentwickler.

- **Risikominimierung** – durch die Wahl bewährter Herangehensweisen und Öffnung der Applikation für ein weites Spektrum an Technologien und Zukunftschancen. Durch die Extraktion der datenspezifischen Business Logik und die Ablage als Datenbankschicht wird die Applikation weniger abhängig von technischen Änderungen an der Anwenderoberfläche, flexibler und robuster.

Zusammenfassung

Wir sind uns bewusst, dass die Entscheidung zum Umstieg auf J2EE und JDeveloper ADF einen wichtigen Schritt und eine große Herausforderung darstellt. PITSS führt Sie durch die strategischen Entscheidungen und bietet eine Lösung, mit der die typischen Risiken eines solchen Umstiegs erheblich minimiert werden:

- Konsolidierung der Business Logik in die Datenbank – für welche Lösungsansatz auch immer Sie sich entscheiden (Beibehaltung von Forms oder Umstellung auf ein anderes Framework), der erste Schritt der Risikominimierung besteht darin, die Business Logik zu konsolidieren, also von nicht genutzten Code zu bereinigen und redundante Code zusammenzuführen sowie diesen in die Datenbank zu verschieben. Allein dieser Schritt kann mit einer Code-Reduzierung und –Konsolidierung von 20 – 30% zur Amortisation des Projektes beitragen
- Vorbereitung auf die neue Technologie durch das Beschaffen und Aneignen der erforderlichen Kenntnisse
- Start von Pilotprojekten – Migration nach ADF von Programmen, die externe Verwendung finden, auf die mit einem Browser zugegriffen werden soll, bei denen es nur geringe Anwenderinteraktion gibt und der Client Zugriff unkritisch ist; gleichzeitig Festlegung, welche Anwendungen in Forms bleiben müssen – meist Applikationen für den internen Gebrauch, insbesondere, wenn sie datenintensiv sind oder sehr häufig die Interaktion des Anwenders erfordern
- Schätzung des Arbeitsaufwands – die PITSS.CON „Moderization Edition“ ME wird Ihnen erhebliche Zeitersparnis bringen und den Ausschlag zum Erfolg geben. Zahlreiche Kunden sind bereits auf uns zugekommen, nachdem sie mitunter mehrmals vergeblich versucht hatten, Legacy Anwendungen von Hand zu modernisieren und dafür ganze Mannjahre an Arbeitsaufwand und gewaltige Budgets investiert hatten – mit enttäuschendem Ergebnis.

Wir würden uns freuen, wenn wir Ihr Interesse für unsere Lösungen wecken konnten und stehen Ihnen mit Rat und Tat zur Seite. Fordern Sie uns auf Ihrem Modernisierungsweg für Oracle Forms und Reports oder bauen Sie erst einmal ein tiefes Verständnis für Ihre Applikation durch eine umfassende Analyse auf. Lernen Sie PITSS.CON und dessen Fähigkeiten kennen.

Über PITSS

Die PITSS GmbH ist der führende Anbieter von integrierten Komplettlösungen für das effektive Management von Oracle Forms Applikationen. Die innovative Software PITSS.CON unterstützt Unternehmen ganzheitlich bei der erfolgreichen Analyse, Migration, Weiterentwicklung und Wartung von Oracle Forms Applikationen. Damit bietet PITSS Oracle Forms Kunden einen fließenden Übergang in die SOA Welt. PITSS.CON überzeugt durch seine hochgradige Automatisierung und Leistungsfähigkeit. Migrations- und Entwicklungsprojekte werden so äußerst schnell, wirtschaftlich und zuverlässig umgesetzt. PITSS.CON erzielt für Unternehmen über alle Entwicklungsprozesse eine Kostenersparnis von durchschnittlich 30% in Migrationsprojekten sogar von 90%. PITSS ist Oracle Certified Advantage Partner und hat seine Kunden in Europa, USA und Asien.



Von Oracle Forms zu Oracle ADF und J2EE

Juni 2009

Autoren: Magdalena Serban, Bahar Us
Co-Autoren: Andreas Gaede, Martin
Disterheft

Korrekturleser: Chris Baker

PITSS in Europa

Deutschland +49-711-728.752.00
info@pitss.com www.pitss.com

PITSS in Amerika

USA +1.248.740.0935
info@pitssamerica.com
www.pitssamerica.com

Copyright 2009, PITSS GmbH