Oracle Application Development Framework Overview

An Oracle White Paper
August 2005

ORACLE FUSION MIDDLEWARE

Oracle Application Development Framework Overview

Introduction 3
The J2EE Challenge 3
Oracle ADF – Making J2EE simpler 3
The Oracle ADF Architecture 5
The Oracle ADF Layers 5
The View Layer 6
The Controller Layer 6
The Business Services Layer 6
The Model Layer 6
Productivity with Choice 7
Visual and Declarative J2EE Development 8
The Business Services layer 8
The Controller Layer 9
The Model Layer 10
The View layer 10
Conclusion 12

Oracle Application Development Framework Overview

INTRODUCTION

The J2EE Challenge

J2EE is an standard, robust, scalable, and secure platform that forms the basis for many of today's enterprise applications. J2EE, however, is just a set of specifications for building multi tier applications using the Java language; it doesn't provide tools for building the applications. This creates a challenge for developers when first starting J2EE development. Many developers are confused by the complexity of the J2EE APIs - a direct result of J2EE's comprehensive nature.

Over the years, best practices and design patterns have evolved for the J2EE platform. Implementing these best practices usually involves writing scores of infrastructure code, adding another obstacle for developers building their first J2EE application.

The way to solve these obstacles is to use a framework that abstracts the complexity of the J2EE platform but still leverages it's power.

Oracle ADF – Making J2EE simpler

Oracle JDeveloper 10g introduces the Oracle Application Development Framework – an innovative J2EE development framework.

Oracle Application Development Framework (Oracle ADF) simplifies J2EE development by minimizing the need to write code that implements design patterns and application's infrastructure. Oracle ADF provides these implementations as part of the framework. Recognizing that having a set of runtime services is not enough, Oracle ADF is also focused on the development experience providing a visual and declarative approach to J2EE development.

Oracle ADF is an evolution, an improvement, and an extension of frameworks that were included with previous versions of JDeveloper.

The key characteristics of Oracle ADF that make it unique among other J2EE frameworks are:

Development Environment – Many of the other J2EE frameworks are missing a development tool that will make it easier to develop with them. Oracle JDeveloper provides a perfect tool for building Oracle ADF-based applications using visual aids and declarative approach to minimize coding.

Platform Independence – Other frameworks lock the developer into a specific software vendor. The Oracle ADF runtime, however, can be installed on any J2EE compliant application server.

Technology Choice – Realizing the developers have preferences for the way they implement different layers of the application, Oracle ADF supports multiple technologies for each of the layers of the application. Oracle ADF doesn't enforce a specific technology on the developer.

End-to-End Solution – Oracle ADF doesn't focus on just one layer of the J2EE architecture. Oracle ADF provides a complete solution for every J2EE layer and every development life-cycle phase.

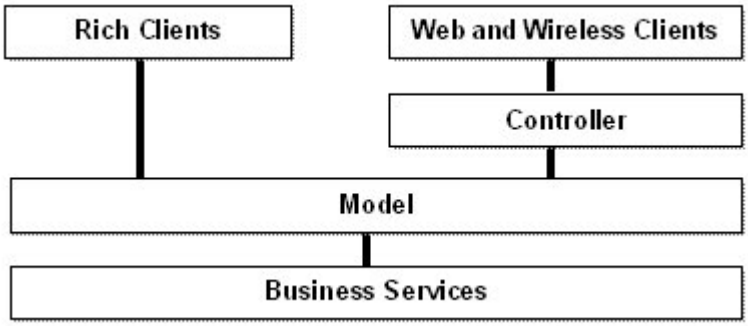
THE ORACLE ADF ARCHITECTURE

Oracle ADF is based on the Model-View-Controller (MVC) design pattern. MVC separates the application architecture into three layers:

- Model - handles interaction with data-sources and runs the business logic
- View - handles the application user interface
- Controller - handles the application flow and acts as the interface between the Model and the View layers

Separating applications into these three layers simplifies maintaining and reusing components across applications. The independence of each layer from the others results in a loosely coupled architecture. Oracle ADF provides an easy way to implement the MVC architecture.

The Oracle ADF Layers



Oracle ADF a high level architecture

Oracle ADF is based on four layers:

- The Business Services layer - provides access to data from various sources and handles business logic.
- The Model layer - provides an abstraction layer on top of the Business Services layer, enabling the View and Controller layers to work with different implementations of Business Services in a consistent way.
- The Controller layer - provides a mechanism to control the flow of the Web application.
- The View layer - provides the user interface to the application.

Oracle ADF lets developers choose the technology they prefer to use when implementing each of the layers. Oracle ADF provides the same visual and declarative development experience regardless of the technology stack used.

The View Layer

The View layer provides the user interface to the application. The view layer uses HTML, rich Java components, or XML and its variations to render the user interface. The View layer can be Web-based, client server-based, or even a wireless implementation.

The Controller Layer

The Controller layer controls the application's flow. Web-based applications are composed of multiple Web pages with dynamic content. The controller layer manages the flow between these pages. Different models can be used when building this layer. The most prominent architecture for Java-based Web applications relies on a servlet that acts as the controller. The Apache Jakarta Struts controller, an open source framework controller, is the de facto standard controller for Java-based Web systems. Oracle ADF uses the Struts controller to manage the flow of Web applications.

The Business Services Layer

The Business Services layer manages interaction with a data persistence layer. It provides such services as data persistence, object/relational mapping, transaction management, and business logic execution.

The Business Services layer in Oracle ADF can be implemented as simple Java classes, EJB, Web services, TopLink objects, or Oracle ADF Business Components.

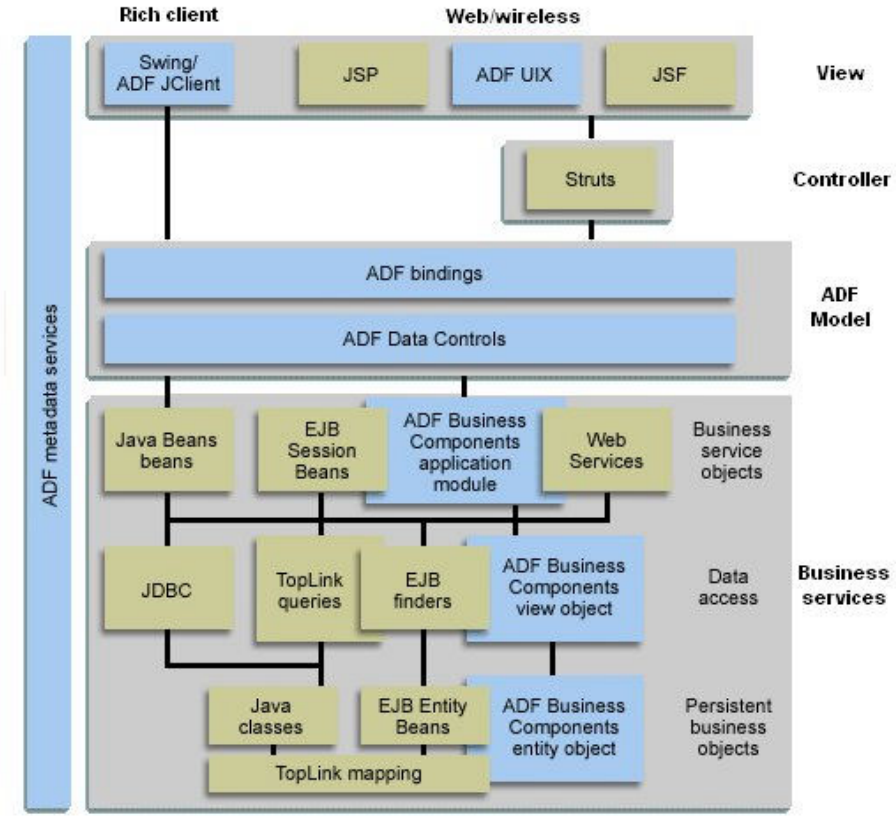
The Model Layer

The Model layer connects the Business Services to the objects that use them in the other layers. Oracle ADF provides a Model layer implementation that sits on top of Business Services, providing a single interface that can be used to access any type of Business Service. Developers get the same development experience when binding any type of Business Service layer implementation to the View and Controller layers. The Model layer in Oracle ADF served as the basis for JSR-227 "A Standard Data Binding & Data Access Facility for J2EE".

Productivity with Choice

Developers can choose different technologies to implement each of Oracle ADF 's layers, and still get the same productive development experience.

The following diagram shows the various options available for developers when building Oracle ADF applications.



Oracle ADF technologies

The diagram displays components implemented by Oracle ADF in blue. Oracle provides an easy way to use the other components with the Oracle ADF framework. The glue that integrates the various components of J2EE applications and makes development so flexible is the Oracle ADF model layer. EJB, Web Services, JavaBeans, and TopLink objects can all be used as Business Services for the Oracle ADF Model.

View layers can include Swing applications as well as HTML interface using JSP, ADF UIX, or JSF.

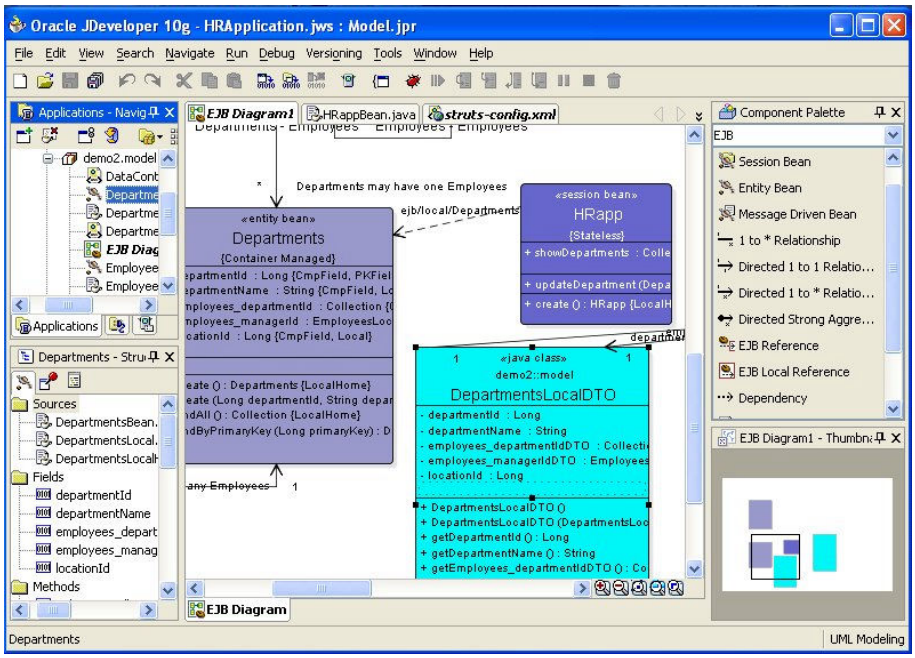
VISUAL AND DECLARATIVE J2EE DEVELOPMENT

A Critical aspect of making a development framework useful is having a development tool that simplify the creation of applications using this framework.

Oracle offers visual and declarative tools for each layer of Oracle ADF. These tools benefit Java developers even if they don't use the runtime features of Oracle ADF.

The Business Services layer

Oracle JDeveloper includes a UML class diagram that models and generates EJB, Web Services, TopLink objects, Simple Java classes, and Oracle ADF Business Components. Developers can drag-and-drop tables from the database browser onto the diagram to generate Business Services that provide Java interfaces to these tables.



EJB Modeling using UML

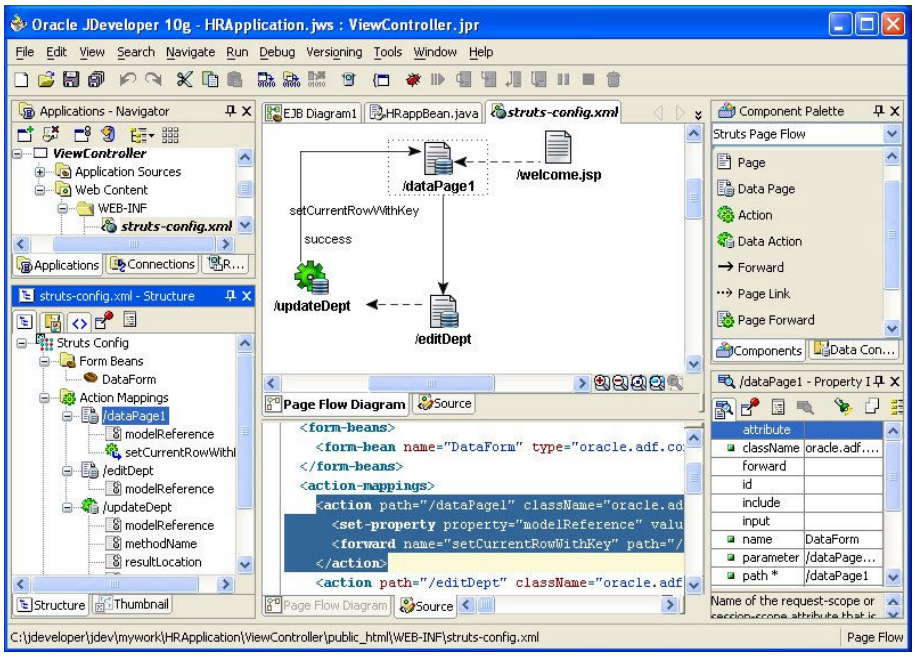
Oracle JDeveloper provides declarative interfaces that simplify the creation of EJB minimizing the code needed to implement EJB interfaces. EJB design pattern implementations can be generated with a click of a button.

Oracle JDeveloper integrates the TopLink Mapper to help developers visually map Java objects to database tables using the powerful TopLink persistence layer.

Oracle ADF Business Components is a framework focused on creating objects, which implement the Business Services layer on top of a database, in a more declarative way. It provides out-of-the-box services such as transaction management, resource pooling, locking, declarative validation rules, translation, and object-relational mapping. Oracle ADF Business Components let developers leverage the full power of SQL when building Java objects that access their databases. With built-in implementation of common J2EE design patterns in the framework, the performance and scalability of the application is assured.

The Controller Layer

Oracle JDeveloper provides a page flow modeler for the open-source Apache Jakarta Struts framework. This modeler provides a visual interface that simplifies the development of the application flow. Developers model their page flows using simple drag and drop of Struts components onto the diagram. The diagram is automatically synchronized with the source in the struts-config.xml file. Using the Data Control Palette, developers can associate Business Services methods with Struts actions. The Struts page flow diagram provides drill-down access to Web page editing and Struts action coding.



Struts Page Flow Modeler

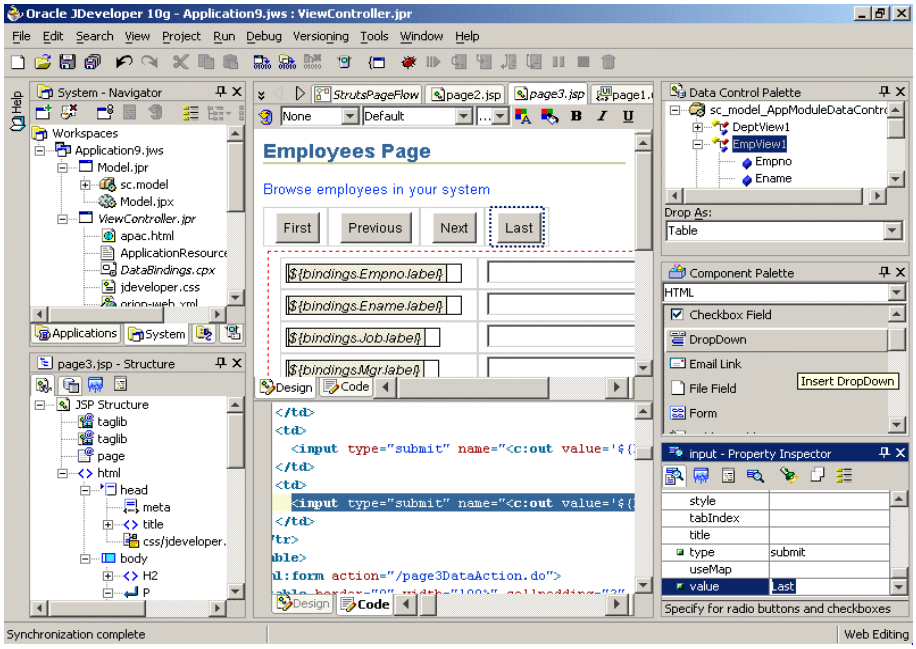
The Model Layer

Oracle JDeveloper provides a very easy way to bind components from the Business Services layer to your Controller and View layers using an innovative implementation of the Model layer. Developers can simply drag-and-drop data objects and bind them to their user interface implementation. The same mechanism enables an easy binding of controller actions to methods defined in the Business Services layer.

The View layer

Oracle JDeveloper provides a visual layout editor for both HTML and Swing based user interfaces. Developers use the component palette to add visual components to the application's user interface. The component palette can be extended with any standard JavaBean or JSP Tag Library. Developers can use the Property Inspector to declaratively define the attributes of visual components. The visual layout editor is synchronized with the source code at all times, so developers can choose their preferred editing mode.

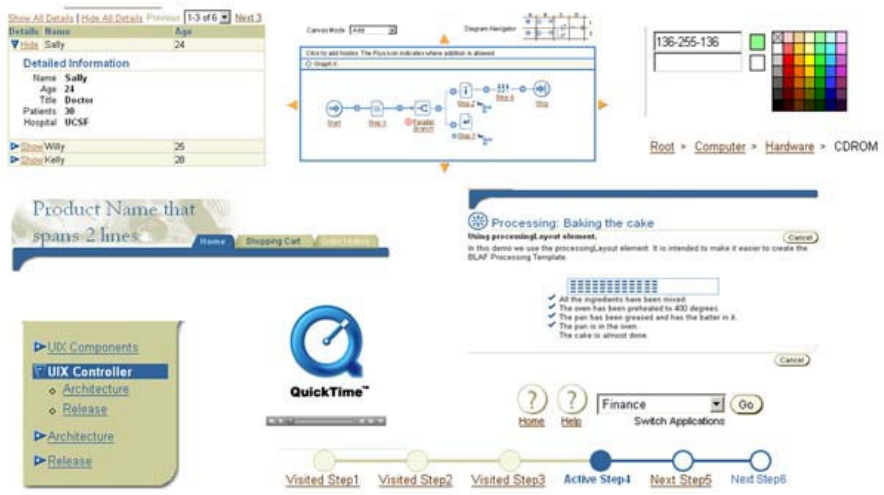
The Data Control Palette window provides a view into the Business Services layer. Developer can bind user interface components to Business Service with a simple drag-and-drop from this palette.



Visual JSP editing

For Web applications Oracle ADF also includes ADF UIX – a large set of UI components that developers can use to declaratively define advanced HTML user interfaces with rich functionality. ADF UIX features include:

- Consistent look and feel based on templates
- Powerful user interface customization
- Easy multi-language translation implementation
- Flexible implementation as Web or mobile interface
- Innovative partial page rendering at runtime
- Rich set of built-in components and functionality such as table with column sorting, calendar, color picker, list of values, and more
- Declarative page definition



Oracle ADF – a rich set of UI components

CONCLUSION

Oracle ADF makes J2EE development simpler by providing out of the box implementation of design patterns and infrastructure code. Oracle ADF provides a choice of development approach, technologies used, and deployment platform. Combining the advance architecture of Oracle ADF with the visual development environment of Oracle JDeveloper 10g, provides a perfect solution for both novice and experienced developers looking to be more productive when developing Java applications.

More resources about Oracle ADF and Oracle JDeveloper can be found on the Oracle Technology Network (OTN), at http://otn.oracle.com/product/jdev.

Oracle Application Development Framework Overview
April 2004
Author: Shay Shmeltzer
Contributing Authors: Regis Louis

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:
Phone: +1.650.506.7000
Fax: +1.650.506.7200
www.oracle.com

Copyright © 2004, Oracle. All rights reserved.
This document is provided for information purposes only
and the contents hereof are subject to change without notice.
This document is not warranted to be error-free, nor subject to
any other warranties or conditions, whether expressed orally
or implied in law, including implied warranties and conditions of
merchantability or fitness for a particular purpose. We specifically
disclaim any liability with respect to this document and no
contractual obligations are formed either directly or indirectly
by this document. This document may not be reproduced or
transmitted in any form or by any means, electronic or mechanical,
for any purpose, without our prior written permission.
Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective owners.