Advanced Analytic Applications with Oracle JDeveloper and Oracle Business Intelligence Beans

An Oracle White Paper
November 2004

ORACLE

Advanced Analytic Applications with Oracle JDeveloper and Oracle Business Intelligence Beans

Advanced Analytic Applications with Oracle JDeveloper and Oracle Business Intelligence Beans

EXECUTIVE OVERVIEW

Many organizations have created large data warehouses to provide their user community with a self-service environment. Most business intelligence tools provide complex query engines and interfaces to help users create and manage sophisticated OLAP type queries. However, many senior executives often require access to powerful OLAP queries within a simplified environment.

Oracle Business Intelligence Beans (BI Beans) with Oracle JDeveloper 10*g* are designed specifically to provide an efficient platform for quick and easy development of powerful business intelligence applications. This paper will describe how the BI Beans function within Oracle's integrated technology stack and how using the BI Beans accelerates the development of feature-rich business intelligence applications, such as the Executive Insight application shown below.

Figure 1: Executive Insight Business Intelligence Application

INTRODUCTION

JDeveloper with BI Beans is a J2EE™ development environment with end-to-end support for modeling, developing, debugging, and deploying e-business applications. BI Beans is a J2EE extension to JDeveloper, allowing developers to quickly and easily create BI applications. BI Beans in JDeveloper 10g accelerates development with code generation, visual editors and J2EE frameworks while offering sophisticated team support, testing utilities and performance tuning tools. Deep integration of industry best practices helps developers to deliver high performance, architecturally sound applications faster.

To maximize developer productivity and freedom to choose implementation at the various layers of the architecture, JDeveloper provides a comprehensive set of integrated tools to support the complete development lifecycle. JDeveloper/BI Beans simplifies BI compliant J2EE development by providing wizards, editors, visual design tools, drag and drop data binding to user interfaces, and deployment tools to create high-quality business intelligence solutions. By using visual, declarative, and guided-coding techniques, the framework allows application developers who are not necessarily J2EE experts to quickly become productive.

The aim of this white paper is to show how to exploit JDeveloper 10g's unique features to create an executive cockpit application. The schema and project files used to create this application can be downloaded from the BI Beans area on Oracle Technology Network. It is based on an Oracle OLAP Analytic Workspace and showcases many of the advanced OLAP features such as forecasting and what-if data modeling.

ORACLE BUSINESS INTELLIGENCE BEANS

Oracle Business Intelligence Beans leverages the Oracle OLAP option to provide Java components for the development of analytical applications.

The key objective of BI Beans is to provide the reusable application components and services that will enable the rapid development of business intelligence applications. This development process is enhanced through integration with JDeveloper. Together, JDeveloper and BI Beans provide a highly productive development platform. The following sections of this paper outline the functions of each core component of BI Beans and the benefits of its integration with JDeveloper.

Oracle itself uses the BI Beans in many of its Business Intelligence products to ensure customers see a consistent user interface across the various reporting products and tools. For example, Business Intelligence Beans technology is currently embedded within OracleBI Discoverer and OracleBI Spreadsheet Add-In. Users of these products view the same OLAP Query and Calculation wizards.

Core Components

The components that comprise BI Beans fall into three general categories: 1) presentation beans, 2) OLAP beans, and 3) catalog services. The following diagram describes the relationships among the components:

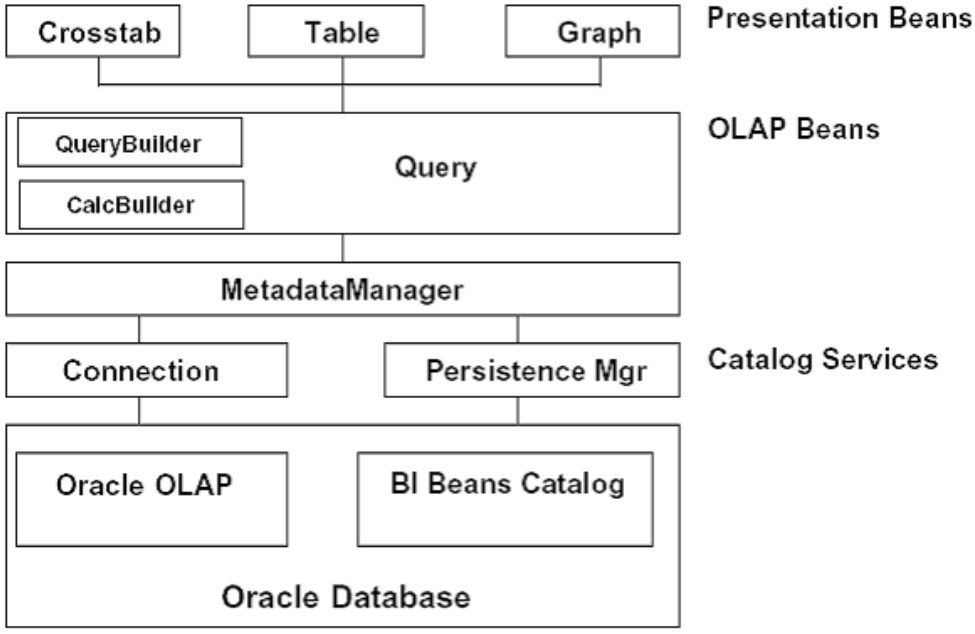

Figure 2: BI Beans Overview

Presentation Beans

The Presentation beans provide the display of complex query results in user-friendly, attractive graphs, tables, and crosstabs. The beans achieve this function by providing data-aware rendering components that are highly customizable both at design time and at runtime through a comprehensive API. The default user interfaces for these beans have undergone extensive usability testing, both formal and through their use in production-level Oracle applications such as Oracle Discoverer and Oracle Reports.

Graph

Using the graph bean an end user can create "boardroom quality" presentations. The graph bean supports 70+ graph types, with special customizers for formatting various graph components (layout, legend, series, axes). With the intuitive custom formatting tools, a novice end user can easily create compelling presentations for senior management. Another benefit is the support for printing, with zoom capabilities that allow users to specify the range to print (for example, all page items or any subset of page items). Graphs are also interactive, allowing users to explore the data by drilling or pivoting.

Crosstab and Table

Like graphs, crosstabs and tables provide formatting and printing support. They also allow users to navigate through the data by drilling, paging, pivoting, and scrolling. In addition, users can define cell-level formatting that is dependent on the data and/or dimension values for a cell. Such rich formatting alerts users to important aspects of their data and makes it easy to spot "problems." Developers do not have to worry about the sizing of crosstabs and tables because they are auto-sized based on the frame or HTML window. This means that users will always have the best view of the data.

OLAP Beans

While the Presentation beans are responsible for rendering the data in different formats, the OLAP beans provide the business logic. This separation of the business logic from the presentation is extremely important, because it enables multiple clients (such as Java, HTML and WAP) to access the same application code.

The following components comprise the OLAP beans.

Connection Bean

The Connection bean provides the connection to Oracle database. An application may access multiple databases; there is no specific limit to the number of servers that are accessible to an application. This capability enables, for example, one report to be derived from a Sales History database while another is derived from the General Ledger database.

Query

Behind any presentation is a query. The query specifies exactly what the user is viewing in the presentation: the measures (such as Sales and Costs), the dimensions (such as Product, Geography, and Time), the selections for each dimension (such as the last 6 months), and the layout of the dimensions (such as Product in the rows, Time in the Columns, and paging on Geography).

Much of the business logic for a business intelligence application is contained in the query. The query provides the OLAP navigation capabilities on the presentation objects. A user might think that drilling down on the Northeast Region in a crosstab is an operation that takes place on the crosstab bean. In reality, the operation is affecting the query that the underlying crosstab is displaying. As mentioned earlier, the crosstab is simply a rendering engine that displays query results. User interactions such as paging and pivoting are methods on the query.

The query uses the Oracle OLAP API to retrieve and manipulate data to provide the advanced analytic capabilities that are offered in Oracle10g. The Oracle OLAP API achieves this by generating highly tuned SQL to resolve the request from the query.

QueryBuilder

The QueryBuilder provides a simple user interface to define sophisticated queries. The user interface is patented and extensively tested for ease of use. The QueryBuilder is a very powerful tool that enables users to specify the following query properties without needing to know the underlying query language:

- Measures and dimensions – Chosen using a list tool which displays a hierarchical list.
- Dimension selections – Specified by defining exceptions, rankings, top/bottom selections, hierarchical selections (children and ancestors), text matching, and so on. For example, "Top 5 Products sold in each City".
- The layout of the dimensions within the presentation.

End-users across an organization have different skill sets and analytical requirements. The QueryBuilder is designed to be highly customizable and componentized to meet these various needs. Developers can turn off capabilities to simplify the user interface or enable it to be used in different contexts. For example, the application might simply provide the list tool or only show the user's favorite selections. Developers can also use components of the QueryBuilder (such as condition templates) in other parts of an application or add and remove condition types for different purposes.

The QueryBuilder also makes it easy to reuse selections by allowing users to save and retrieve defined selections as favorites. Users often want to use a popular selection or query (such as "Top 5 Products") in multiple reports or presentations.

A key strength of the QueryBuilder is that the end-user does not need to understand a query language to define the query. Powerful queries are made simple by presenting the query definition in business terms, which end users can modify to meet their needs. For example, a default query definition can be "Show the Top 5 Products based on Sales". An end-user can then modify the word "Top" to "Bottom" in the definition to show the bottom 5 Products based on Sales.

CalculationBuilder

The CalculationBuilder bean provides a user interface to create derived calculations. Like the QueryBuilder, end-users define new metrics using templates, which eliminates the need to understand the underlying SQL. For example, an end-user might be interested in seeing Sales Growth compared to last year for a particular product line, even though Sales Growth has not been defined by the database administrator. The CalculationBuilder enables the user to define this calculation. The calculation templates are organized by type. In our example, the end-user selects the appropriate Time Series template to define the new measure.

Catalog Services

BI Beans Catalog

The BI Beans Catalog provides a secure, searchable, and extensible object storage mechanism that enables end users to save personal analyses and share analyses with other members of the user community. The catalog stores the definitions of tables, crosstabs, graphs, calculations, and queries in XML format.

These objects are organized into folders, where access privileges are granted at the user and user group level. When saving a report, for instance, an end-user could save the document into a Finance Group folder, which would then be accessible to his or her peers in the finance organization. The catalog also supports pluggable authentication schemes and storage mechanisms. For example, developers may want to authenticate users against a company LDAP server or save objects to a data store other than the two supported by default – the Oracle database and the local file system.

The catalog is searchable based on keyword, name, description, author, or date and is filterable by object type (for example, show graphs only). This allows users to concentrate on the objects of interest in a large pool of defined objects.

The catalog is also extensible. Applications can define their own persistable objects by implementing the persistence interfaces. For example, a developer might create a compound object that contains a graph and a crosstab. This new object may contain layout information, page titles, and so on. By implementing the persistence interface, this new object can behave just like any native object; for example, the object can be saved and searched.

MetadataManager

The MetadataManager bean provides an integrated view of all the metadata that is used in the definition of analytical applications. It merges at runtime the metadata that has been defined in the OLAP Catalog with the metadata that is stored in the BI Beans Catalog.

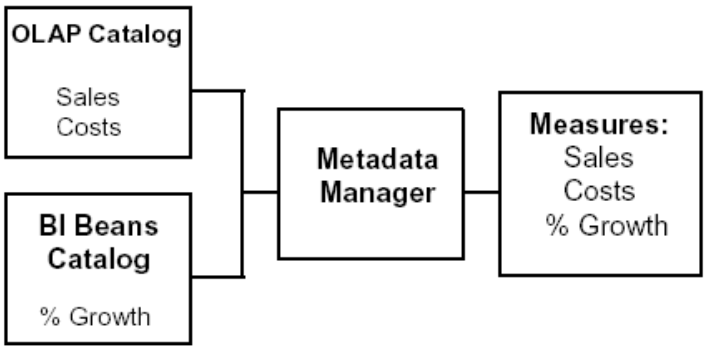

Figure 3: MetadataManager Overview

End users need to access common pieces of information (such as Sales and Costs); this information is stored in the OLAP Catalog within Oracle10g OLAP.

Users also need access to personal calculations (such as % Growth) whose definitions are stored in the BI Beans Catalog. These personal calculations might not be meant for general use, in which case they will not have been defined in the common schema.

When creating a new query or calculation, users need an integrated view of both personal and common metadata. The MetadataManager provides that integrated view of information.

APPLICATION DEVELOPMENT USING ORACLE JDEVELOPER AND BI BEANS

Using the BI Beans within JDeveloper provides the most productive development environment for analytical applications. BI Beans are standard JavaBeansTM that can be used in any Java development environment that supports JavaBeans components. However, JDeveloper is the development environment of choice because of its integrated support for the BI Beans.

The development of BI applications using Java is a two-stage process. First, the developer defines the application objects, which can be graphs, tables, crosstabs, queries, or calculations. Then, the developer defines the application logic or code, which manipulates the application objects to create useful BI applications.

JDeveloper provides several tools to facilitate the development process during both stages. The following sections will illustrate the development process by stepping through creating a page of the Executive Insight Dashboard application.

Application Overview

The Executive Insight application consist of the following pages:

- **Home Page** – The home page is the starting point for the briefing. The home page is designed to provide a high level overview of the key business drivers, allowing a user to drill down to more detailed analysis on subsequent pages. The page is divided into two areas: 1) high level analysis reports and 2) briefing page shortcuts. Clicking on the briefing page shortcut buttons launches the selected page.
- **Product Insight** – The Product Insight page is designed to have a dashboard look and feel. This page is divided into thee main regions: 1) buttons that provide quick access to each product division, 2) six detailed reports providing analysis of revenue, costs and margin over time, and 3) briefing navigation buttons. The aim of this page is to provide a detailed snapshot of business performance and to highlight potential problems relating to margin, revenue and costs growth.
- **Customer Insight** – Similarly to the Product Insight page, the Customer Insight Page is also divided into three main regions. The aim of this page is to provide a detailed snapshot of business performance to highlight potential problems relating to margin, revenue and or costs growth from a

geographical perspective. The graphs and screen layout is identical to the Product Insight page.

- **Sales Analysis** - The aim of this page is to provide a detailed analysis of current and future sales revenue for individual products. While previous pages have concentrated on high-level category and region analysis, with this page it is possible to drill down into the detailed information and perform forecasting analysis on sales of individual products.
- **What-If Analysis** – In this example we provide an analysis of future promotional activity. The user can run a particular type of marketing campaign during selected periods and view projection of sales revenue, costs and margin. This projection is based on modeling the previous campaign results and projecting the data forward to estimate the future sales
- **Ad-Hoc Analysis** – On this page the user can analyze any of the presentations displayed on the previous pages using the ad-hoc analysis tools provided by the BI Beans.

Figure 4: Page-to-Page Navigation in the Executive Insight Application

This document will focus on developing the Product Insight page, highlighting the use of the new BI Beans advanced JSP tags. The following image shows the key features that will be explored in the next parts of the document.

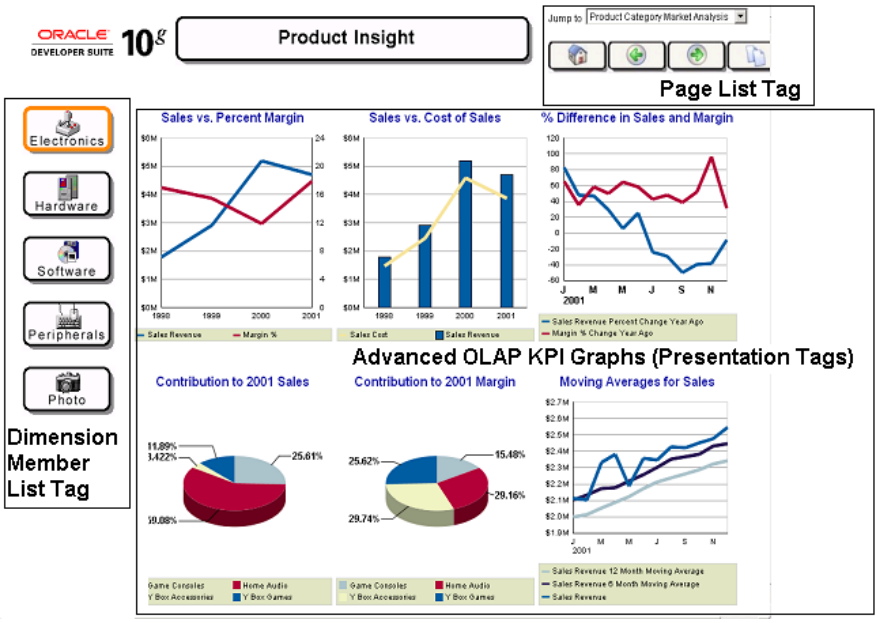

Figure 5: Usage of the BI Beans JSP Tags on the Product Insight Page

Creating BI Presentations

JDeveloper provides wizards that guide developers through the process of creating new BI objects. The BI Designer wizard establishes a live connection to the data in an Oracle database. Developers can visually create a data presentation, a query, or a custom calculation using wizards and without writing a single line of code. This simplifies and speeds up the definition process significantly.

JDeveloper ensures increased productivity by providing live access to data while the presentation objects are created and modified. The presentation editor enables users to modify presentation properties with a live data source. This means, for example, that the developer does not need to compile and run the application to view the results of a query in a presentation. This is very important because the format of the presentation is typically driven by the data itself. For example, when creating a Graph, the application developer needs to see the live data to be able to select the correct Graph type, color format, and other visual properties.

Another key benefit of using BI Beans with JDeveloper is the integration of the BI Beans Catalog with the development environment. A given application may contain hundreds of BI application objects. To make such an application scalable, the definitions of application objects need to be stored in the catalog – not in application code. This integration also ensures the reusability of objects, because objects are stored and shared through public folders.

To create the Graphs used by the Executive Insight BI Beans application, the application developer simply steps through the Presentation Builder wizard that is based on the QueryBuilder bean and is built into JDeveloper.

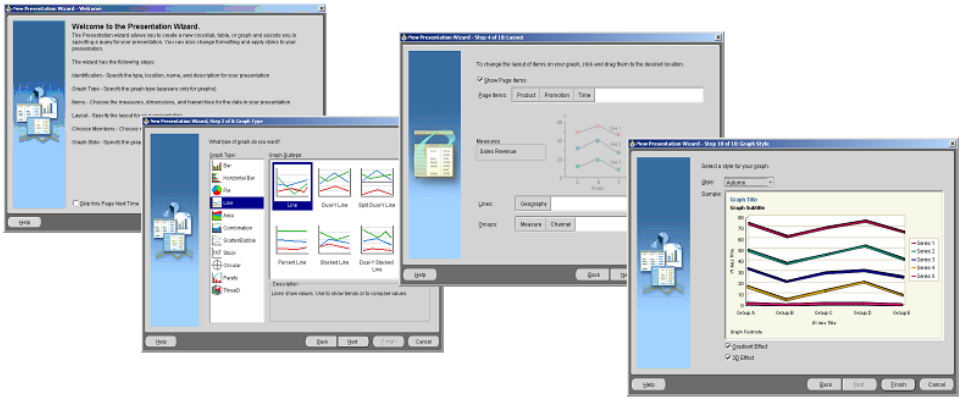

Figure 6: Creating a Key Performance Indicators Graph in JDeveloper and BI Beans

Developing the Application Code

Developers who are familiar with JDeveloper concepts will quickly recognize and understand how to develop applications using the BI Beans. The development of BI applications uses generic JDeveloper concepts whenever possible, and extends those concepts when required. This is abundantly clear when you view the JDeveloper tools palette, project navigator, and wizards; the BI Beans fit seamlessly into the environment.

JDeveloper provides wizards to make the development of applications simpler, including those that guide developers, in a step-by-step process, through the creation of Java applications or applets for Java clients and JavaServer Pages (JSPs) for thin-clients. Such application wizards greatly reduce the amount of time required to develop an application and the wizards provide the flexibility in deployment. Developers can easily create both Java-client and thin-client applications that are based on the same application objects.

Developing the Product Insight JSP Page

To create a new BI Beans JSP Page, the application developer simply steps through the JDeveloper's New Object Gallery wizard.

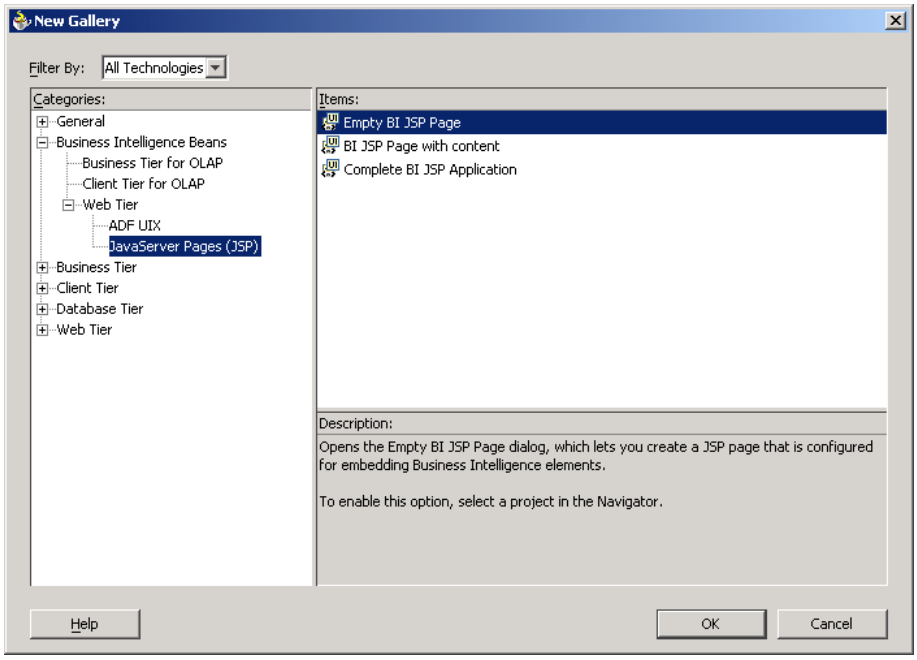

Figure 7: Generating a new BI JSP Page in JDeveloper

BI Beans integration with JDeveloper 10g includes a custom JSP tag library, which contains JSP tags that encapsulate BI logic. For example, instead of writing many lines of custom code to display a graph, a developer can include a simple presentation JSP tag in their application. The custom tag library allows the rapid development of BI applications, and provides access to all of the BI Beans components.

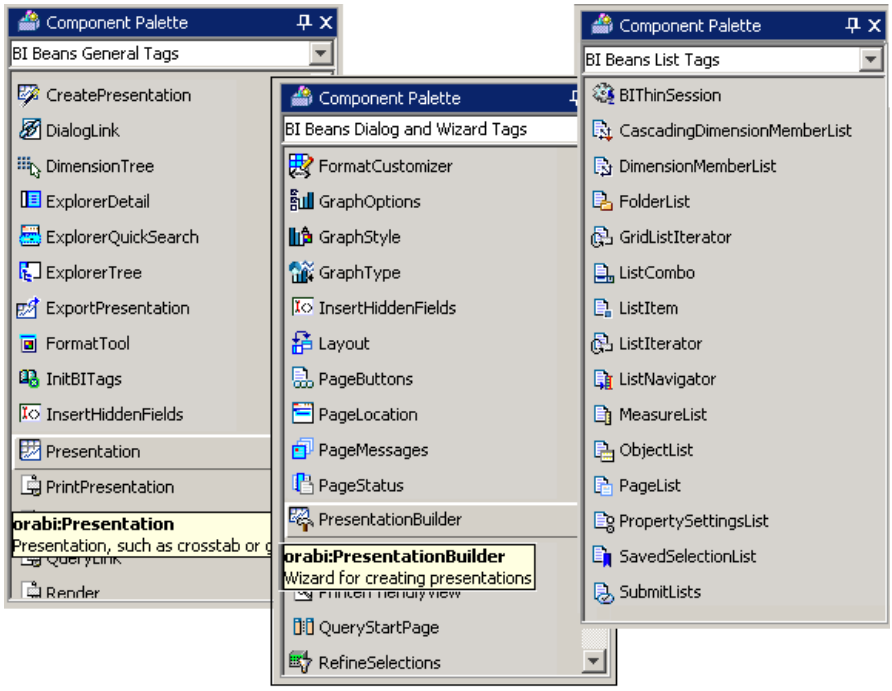

Figure 8: BI Beans JSP tag library in JDeveloper

BI Beans JSP tags also integrate with the Visual JSP Editor, a new feature in JDeveloper 10g. The JSP/HTML Visual Editor is a canvas in which the application developers directly manipulate visual user interface components in a JSP page using actions such as drag and drop. When the developer opens a file in the Visual Editor, the page is rendered in HTML and associated web formats, just as it will appear in a web browser when deployed. When the developer edits the visual characteristics of the page in the editor, they immediately see the results of the edits, just as they will appear in a browser.

The developers can also use their favorite HTML editor to edit the HTML part of the JSP page.

To create the Product Insight page in JDeveloper, the application developer simply goes through the following steps:

1. Create the Graphs displayed in the page using the New Graph wizard
2. Create a new Empty BI Beans JSP page using the wizard
3. Using the JDeveloper visual editor (or another HTML editor), create the HTML table layout with 2 rows and 3 columns
4. Insert the Graphs in the cells by dragging and dropping the Presentation tags
5. Create another vertical row layout to the left and insert the Product buttons using the DimensionMemberList BI Beans tag
6. Insert the page navigation controls in the top right using the PageList BI Beans tag

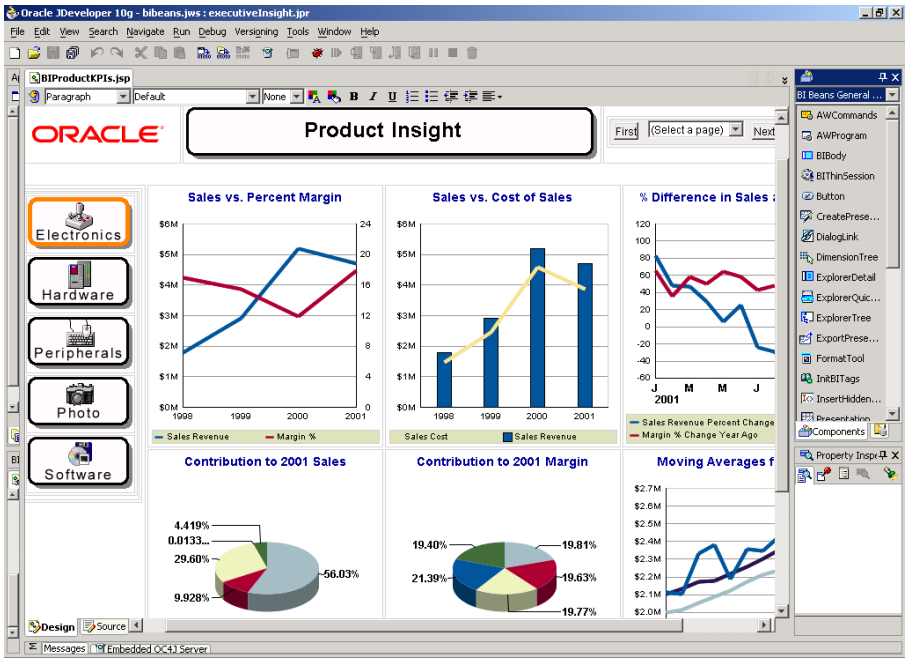

Figure 9: Developing BI Beans Dashboard Page in the JSP Visual Editor

Using BI Beans List Tags for Rapid Application Development

The BI Beans in JDeveloper 10g introduces a new feature, which is a set of List tags. The List tags allow application developers to quickly generate lists of objects on the page:

- Dimension Member List
- Saved Selection List
- Page List
- Measure List

The application developer can choose to render the list controls as buttons, links, radio buttons, images, etc.

To illustrate this concept, let us use as an example the Product buttons on the left hand side of the Product Insight page. To create these buttons, the application developer can use the new DimensionMemberList tag to create the buttons for all the Products from the desired level of Product dimension. The developer simply steps through the wizard to choose the dimension, hierarchy, dimension members, and then specify that the resulting controls should be rendered as images.

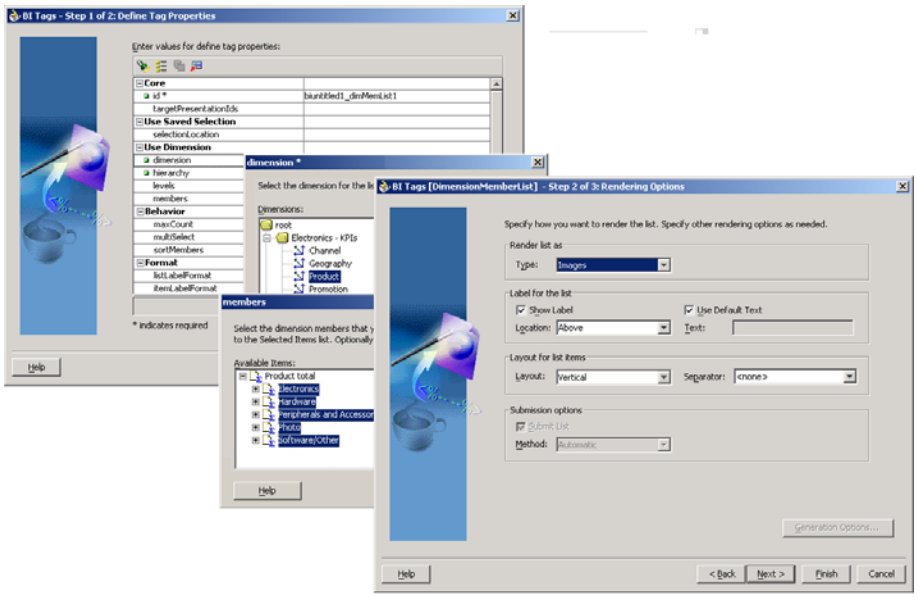

Figure 10: Creating the Product Category Buttons

Similarly, the application developer can use the PageList tag to generate the briefing book navigation controls on the top right without writing any code. The PageList tag automatically generates the navigation controls for all the JSP pages present in the JDeveloper project.

CONCLUSION

This paper has shown how the BI Beans inside JDeveloper framework allow application developers who are not necessarily J2EE experts to quickly become productive. Developers can quickly and easily create customized BI solutions using the new wizards, editors, and visual design tools.

With faster development cycles, more innovative features and strict standards compliance, JDeveloper 10g with BI Beans changes developers expectation of what a development environment should be. The business value is clear: faster time-to-market with lower up-front investment and reduced total cost of ownership. Oracle JDeveloper with BI Beans is the industry's first complete and integrated Java development environment for business intelligence applications.

Building Advanced Analytic Applications with Oracle JDeveloper and Oracle Business Intelligence Beans
November 2004
Author: Katarina Obradovic-Sarkic, Senior Product Manager, Oracle Business Intelligence Tools
Contributing Authors: Keith Laker, Principal Product Manager, Oracle Business Intelligence Tools

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

Filename: Advanced Analytic Applications with Oracle JDeveloper and Oracle BI Beans
Directory: C:\OW2004\Master-FINAL
Template: C:\Documents and Settings\cschua\Application Data\Microsoft\Templates\Normal.dot
Title: Advanced Analytic Applications with Oracle JDeveloper and Oracle BI Beans
Subject: Advanced Analytic Applications with Oracle JDeveloper and Oracle BI Beans
Author: Katia Obradovic-Sarkic, Senior Product Manager, Oracle Business Intelligence
Keywords: BI Beans, Business Intelligence, JDeveloper, JAVA, JSP
Comments: Oracle Business Intelligence Beans (BI Beans) with Oracle JDeveloper 10g are designed specifically to provide an efficient platform for quick and easy development of powerful business intelligence applications. This paper will describe how the BI Beans fu
Creation Date: 11/15/2004 7:24 PM
Change Number: 7
Last Saved On: 11/17/2004 6:23 PM
Last Saved By: user
Total Editing Time: 14 Minutes
Last Printed On: 12/1/2004 2:43 PM
As of Last Complete Printing
Number of Pages: 17
Number of Words: 3,777 (approx.)
Number of Characters: 21,531 (approx.)