

DEVELOPING APPLICATIONS WITH BUSINESS INTELLIGENCE BEANS AND ORACLE9i JDEVELOPER: OUR EXPERIENCE

*Chris Claterbos, Vlamis Software Solutions, Inc.
claterbos@vlamis.com*

INTRODUCTION

With the introduction of Oracle 9i OLAP and BI Beans the landscape for development of BI applications is changing. This leaves clients with existing Express based applications developed in Web Agent and Express Objects asking “How do we move our applications?”. The direct migration of such applications is not fully supported but there is hope. This presentation will address the issues involved in converting and migrating these type applications over to the new technology. Example applications will be presented and used in the presentation.

OVERVIEW OF THE ORACLE 9i AND BI BEANS ARCHITECTURE

OLAP AND ORACLE 9i – WHAT IS IT?

Businesses need to analyze their businesses in ways that decision makers at all levels can quickly respond to changes in the business climate. While a standard transactional query might ask, “How many bolts were sold last month?” An analytical query might ask, “How do sales in the Midwest for the last 3 months compare with the forecast? Now how does that compare to a year ago?”

Analytical queries require an online analytical processing (OLAP) solution. Oracle 9i provides comprehensive support for OLAP:

- The Oracle relational database management system (RDBMS) remains the most efficient and secure way to store your data. By developing a data warehouse, you can provide data in a form suitable for business analysis.
- OLAP Services provides a Java OLAP API and an analytical engine. OLAP Services provides the means to build analytical applications that support complex statistical, mathematical, and financial calculations along with predictive analytical functions such as forecasting, modeling, consolidations, allocations, and scenario management. Because the OLAP API is all Java, OLAP Services supports deployment of analytical applications to large, geographically distributed user communities on the Internet.
- The Oracle BI Beans complements OLAP Services by providing pre-built Oracle JDeveloper or other Java development environments to build analytical applications, which can be deployed as either Java or HTML (“thin”) clients.

OLAP SERVICES

Oracle9i OLAP Services provides the query performance and calculation capability of a multidimensional database. In addition, it provides a Java OLAP API that is appropriate for the development of internet-ready analytical applications.

Unlike other marriages of OLAP and RDBMS technology, Oracle9i OLAP Services is not a thinly disguised multidimensional database using bridges to move data from the relational data store to a multidimensional data store. Instead, it is truly an OLAP enabled relational database.

As a result, Oracle9i provides the performance and calculations of a multidimensional database along with the scalability, accessibility, security, manageability, and high availability of the Oracle9i database.

In its broadest definition, OLAP Services consists of the following components: the Oracle Java OLAP API, one or more OLAP services that run as child processes of a database instance, a metadata repository in each database instance, and tools within Oracle Enterprise Manager for creating OLAP metadata and managing OLAP services.

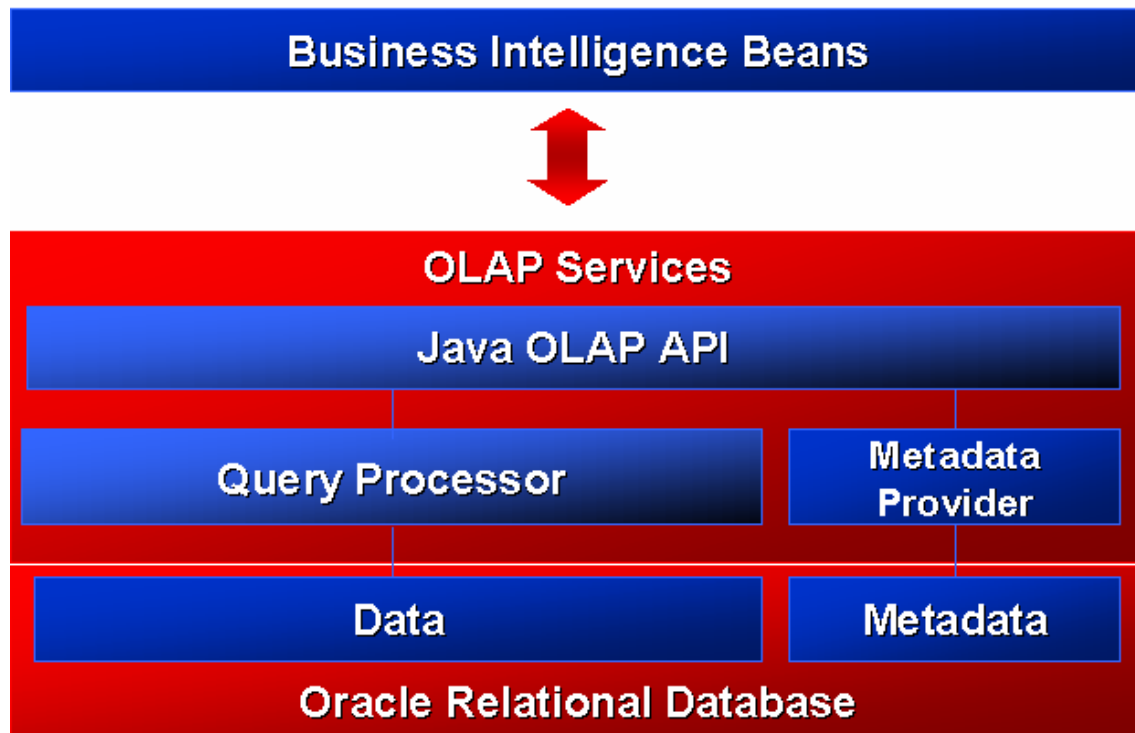


Figure 1. Oracle 9i and OLAP Services Architecture

ANALYSIS FUNCTIONS

The enhancements to SQL to support analysis functions have greatly improved in the recent years. The goal of these enhancements is to make it easier to write highly performant business intelligence queries against the database. A list of the family of queries that have been added is presented below:

- Ranking family
- Window Aggregate
- Reporting Aggregate family LAG/LEAD
- Linear Regression family
- Inverse Percentile family
- Hypothetical Rank and Distribution family
- FIRST/LAST Aggregates family

MATERIALIZED VIEWS

One of the techniques employed in data warehouses to improve performance is the creation of summaries, or aggregates. Materialized Views are a special kind of aggregate view which improves query execution times by pre-calculating expensive joins and aggregation operations prior to execution, and storing the results in a table in the database. For example, a table may be created which would contain the sum of sales by region and by product.

COMMON WAREHOUSE METADATA

One of the most important advances is the establishment of a metadata standard. Metadata is used to represent a wide range of information, including information about the sources that are used to create the data warehouse, data transformations, security and business rules, etc. Metadata is what gives meaning to the data in the data warehousing environment. In order for tools to interoperate, they must understand each other's metadata, and they must be able to cooperate and use the metadata generated by each other.

Common Warehouse Metadata (CwM) defines a standard for metadata sharing. The objective of Common warehouse Metadata is to provide an open standard metamodel and application program interfaces (APIs) for data warehousing and decision support tools. Common Warehouse Metadata enables end user customers to dramatically lower the initial and ongoing costs of building, operating and using data warehouses for decision support by providing an open object model and an open, network-oriented API.

JAVA OLAP API

The Oracle OLAP API is a Java programming interface to OLAP Services. The analytical BI Beans are built using this API; you can extend (or even replace) the functionality provided by the BI Beans by using Java classes.

The Oracle Java OLAP API is designed from the 'ground up' for OLAP. It is a modern Java, object-oriented API. It provides a simple means of expressing complex multidimensional queries containing with high analytic content.

The Java OLAP API complements the new SQL analytical functions the relational database. Relatively simple analytical applications might choose to use SQL, while more sophisticated analytical applications will use the Java OLAP API. OLAP Services will make significant use of new SQL analytical functions in the relational database

Key features of the Java OLAP API include:

- A multidimensional object model. Other Java OLAP API APIs do not provide an object model - it's up to each application to implement one. This is not a trivial task.
- A complete set of OLAP calculation functions (for example, statistical, mathematical, financial, and time series functions).
- Support for asymmetric queries. For example, nested rankings.
- Calculated (virtual) dimension members that can be used just as stored dimension members.
- Incremental query construction. For example, select all products at the Item level, keep products where sales grew 50% or more as compared with last year, remove products with a margin of less than 25%.
- Multidimensional cursors allow an application to request a subset of data within the users query (for example, only those cells of data which can actually be displayed on the users monitor). This offers significant performance tuning opportunities to the application developer.

ORACLE BUSINESS INTELLIGENCE BEANS

To meet the needs of the marketplace, Oracle has developed BI Beans. These components enable the rapid development of OLAP applications. The combination of the Java OLAP API and Common Warehouse Metadata provide APIs that make the BI capabilities of those servers accessible to Java programmers. The role of BI Beans is to complete the engine-API-tools offering by providing the following:

- Graphical views of OLAP data including a pivot table and rich business charts
- User interfaces for easily formulating OLAP queries, calculations, sorting criteria and other analytic functions, both at application design-time and at run-time
- Support for visual programming using Java integrated development environments (IDE)
- Support for building Internet computing-compliant n-tier OLAP applications
- Interoperability with third-party components through support for open standards, notably Beans, CORBA/IIOP and InfoBus

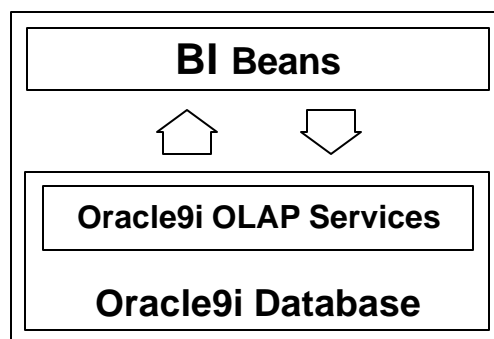


Figure 2. BI Beans and Oracle 9i Interaction

The Oracle Java Tools will provide a tools infrastructure (IDE, repository, n-tier development, modeling, etc.) that will enable developers to build applications that take advantage of Internet computing. The BI Beans leverage the services provided by Java and fit naturally into the Java environment. People who become proficient in developing applications using Java will be able to construct robust BI applications without having to learn separate tools or infrastructure-related concepts.

Key Features of BI Beans:

- Support visual development
- Transform any Java IDE into a powerful OLAP development tool
- Standard pages without programming
- Optimized for Oracle environments
- Open, standards-based
- Beans: properties, methods, and events
- Property Editors and Customizers
- BI functionality can be built into any Java based application including Relational based applications

Oracle's JDeveloper has been enhanced with design-time running of objects, wizards for creation of "BI Pages" and a rich set of customizers and property editors to allow for very rapid BI application development.

ROBUST SET OF ANALYTICAL COMPONENTS

The first release of the BI Beans focuses on the most common BI components used in analytical applications today. In particular, the following JavaBean components that comprise the application fall into three general categories: 1) presentation beans, 2) analysis beans and 3) persistence services. Below is a diagram describing the relationships between the components.

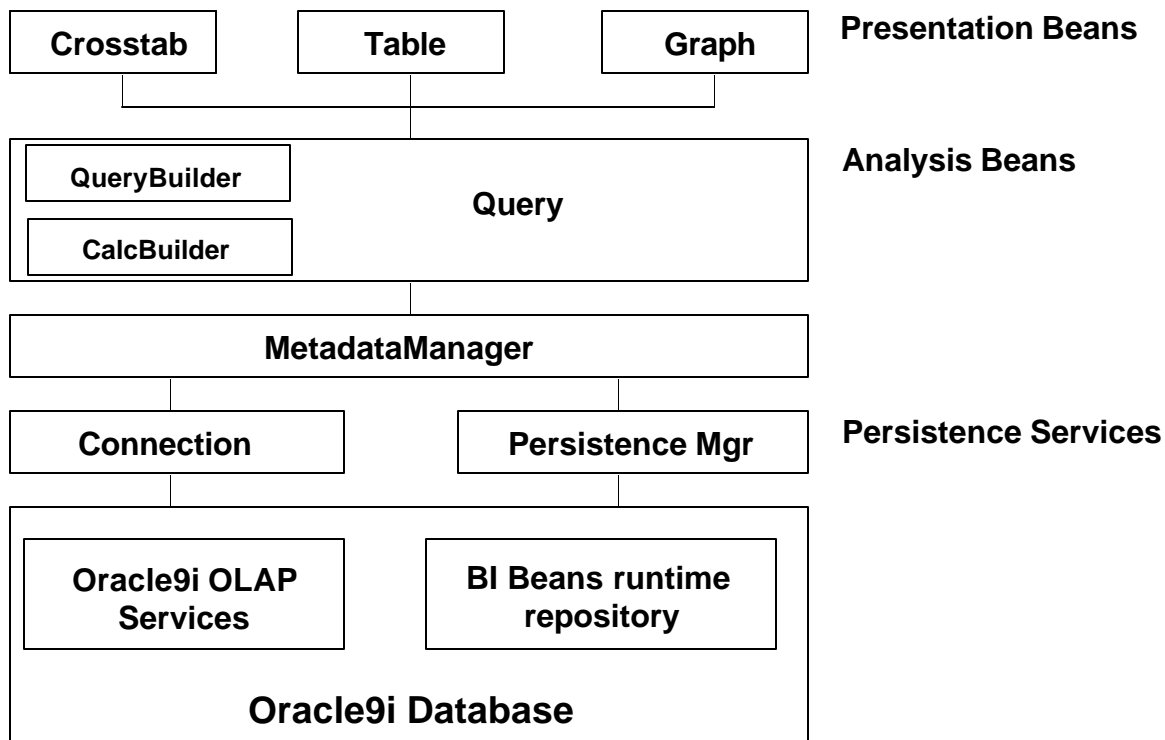


Figure 4. BI Beans Architecture

PRESENTATION BEANS

GRAPH, TABLE AND CROSTAB

The Presentation Beans display the results of queries using graphs, tables and crosstabs. Depending on the nature and requirements of the application, these data presentations may be Java or HTML based. For example, high-end analytical users who spend a large percentage of their work week analyzing past business performance or developing forecasts of future performance need a highly interactive environment. The Java versions of the Presentation Beans enable a rich, interactive experience. For example:

- rotating the dimensions in a presentation is accomplished using drag and drop
- formatting is achieved through direct manipulation. A formatting toolbar is offered to enhance the interaction.
- data entry with clipboard support is provided.
- frozen row and column headers aid the navigation through large reports.

The HTML versions of the presentation beans can display the custom formats that have been defined and saved by the Java client. They also have the added benefit of not requiring Java to be downloaded to the client. However, the interactive definition of formatted documents is more difficult and end users are subject to the limitations of HTML.

ANALYSIS BEANS

The Presentation Beans are solely responsible for rendering the data in different formats. The business logic is provided by the Analysis Beans. This separation of the business logic from the presentation is extremely important because it enables multiple clients (e.g. Java, HTML, WAP, etc.) to access the same application code.

The following components comprise the Analysis Beans:

QUERY

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

Much of the business logic for a business intelligence application is contained in the Query. One typically thinks that drilling down on the Northeast Region in a crosstab as an operation that takes place on the Crosstab bean. In reality, this is not the case; the operation is affecting the query that the underlying Crosstab is displaying. As mentioned earlier, the Crosstab is simply a rendering engine that is displaying query results. User interactions such as paging and pivoting are also methods on the query.

The Query bean uses the Java OLAP API to query and manipulate data. The Java OLAP API, in turn, generates highly tuned SQL to resolve the request.

QUERYBUILDER

The QueryBuilder provides a simple user interface to define the query. It is a very powerful tool that enables users to specify the following query properties:

- the measures and dimensions
- the dimension selections (e.g. Top 5 Products sold in each City)
- the layout of the dimensions on the report

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 modify to meet their needs.

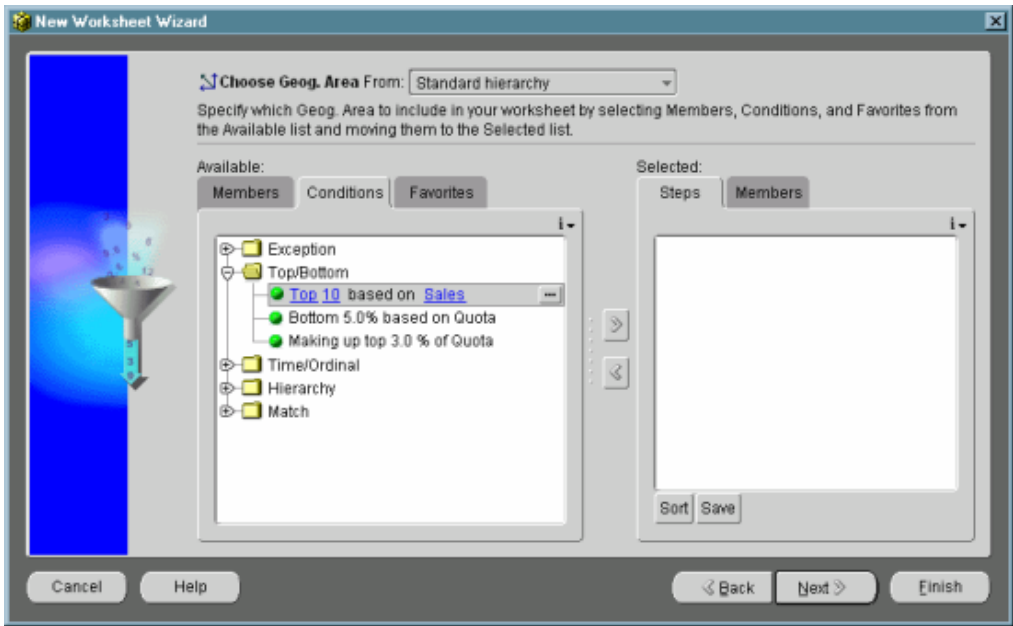


Figure 5 - QueryBuilder

CALCBUILDER

The Calculation Builder allows users to create derived calculations (custom measures) through templates. Calculation Builder has full support for the advanced analytic functions of Oracle9i OLAP, including functions for time-series analysis, and many other calculations.

BI BEANS CATALOG

The BI Beans Catalog is used to save, retrieve, and manage all developer-defined and user-defined analytical objects, such as reports, graphs, favorite queries, and custom measures. Object definitions are stored in the Catalog as XML. The BI Beans Catalog is designed to support large distributed user communities who share analytical objects in collaborative environments. Developers take advantage of the Catalog at design time and hook application logic to the Catalog, so users can access Catalog functionality from the application at runtime.

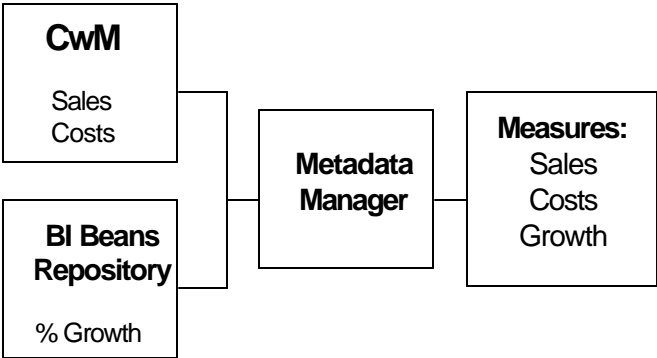


Figure 6 BI Catalog Architecture

End users need to access common pieces of information (e.g. Sales and Costs); this information is stored in the CwM or ECM. Users also need access to personal calculations (e.g. % Growth) whose definitions are stored in the beans repository. These personal calculations may not be meant for general use, so they have not been defined into the common schema.

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

USER INTERFACE COMPONENTS

BI Beans also provides Java-based and HTML versions of user-interface components to support the opening and saving of objects to the Catalog. The BI Explorer (pictured right) provides Catalog browsing functionality similar to Windows Explorer. You can also create folders and set security privileges.

MIGRATION OF DATA ISSUES

Oracle Express application data is currently stored in either an express database (MOLAP) or relationally via Oracle Express RAM. The Oracle OLAP API and Oracle OLAP services do not fully access data in either of these formats. With 9i Release 2 the Analytic Workspace will be stored in an Oracle database but the data will not be converted to a relational storage model. With respect to RAM databases the data is stored in relational format but it is not fully dimensionally aware.

Conversion of this data to allow Oracle OLAP and OLAP API applications to see the data can be a large issue. The data that is already stored in the relational database may be accessed by creating the necessary dimensions and building the Data Cubes and folders from the relational fact tables using the OLAP Cube Builder in Oracle Enterprise Manager (OEM). The existing MOLAP data located in the Analytic Workspace is more problematic. Oracle presently does not provide a conversion utility to perform this task. In future releases of Oracle OLAP and Oracle BI Beans applications may be able to access the data contained in the Analytic Workspace without conversion. If conversion is desired the data can be converted manually by writing building relational tables and exporting the data out of Express and importing the data into the relational tables. Vlamis Software will be releasing a toolset that will perform this process if a turnkey solution is desired.

OVERVIEW OF WHAT CAN AND CANNOT BE MIGRATED/ CONVERTED

When it comes to applications the picture is much less favorable. At present, there are generally four types of applications that use Oracle Express data. The most prevalent applications are Oracle Applications (Oracle Sales Analyzer, Financial Analyzer and Demand Planner), Oracle Express Analyzer applications (Client Server Mode), Oracle Express Objects (OEO) Applications, and Oracle Express Web Agent Applications. With regards to Oracle Applications Oracle is planning on providing migration paths for all of these applications and I will not address this any further. For those that have Analyzer, OEO and Web Agent applications the picture is becoming clear. Client Server applications using Analyzer and Express Object will not have a migration path. Since the new development paradigm is JAVA there is no direct way to convert these applications. With respect to Express Web Agent there may be hope. Oracle is testing and will soon release a version of Oracle Web Agent for 9i which will support existing web agent applications running under the 9i environment.

CONVERSION VS. NEW DEVELOPMENT

Since Analyzer, OEO and Web Agent applications cannot be converted at the present time then the big question is "What Now?". Since Analyzer and Web Agent applications are generally a collection of data views of data that is contained in the OLAP database it is a straightforward task to redefine these views using the presentation wizard contained in JDeveloper or VSS's BI Analyzer. Once these presentations are created and saved in the BI Catalog the presentations can be viewed using a Java or Web front-end.

OEO applications are much more problematic. The solution here is to re-write the application using JDeveloper and Java using the existing OEO application as a specification. By using the Wizards and samples provided in the new development environment re-building the application will take much less time than the original development did, depending on the sophistication of the original OEO application. While a lot of people will shy away from this effort, it should be pointed out that a large percentage of any development effort is design and data. With the OEO data converted, if necessary, the original application can be used as the design. So the coding of the front-end in Java is the remaining step. With the new functionality provided in JDeveloper and the BI Wizards this phase of the project is much easier and the resulting applications are more stable and deployment options are much more flexible.

ISSUES WITH DESIGN AND IMPLEMENTATION

There are several issues that need to be considered when migrating/converting Express applications. Since BI Beans is now based upon Java there are enhanced functionality and flexible deployment options along with some limitations. There are some things that cannot be done in Java that could be done in OEO. Most of these limitations pertain to windows and operating system functions that cannot be performed. In reality, maybe these functions should not be used anyway.

Deployment options are now more flexible and have a much bigger impact on how the application is designed. The choice of fat or thin java/html client needs to be made and with this choice comes limitations on how the application looks and operates. If the desire is to have a simple high speed reporting system then thin client (html) deployment is the best choice. If download speed and workstation foot-print is not an issue and a more sophisticated front-end is necessary then a java client based application may be the best. A java client application will more closely resemble the windows client look and feel provided by OEO and Express Analyzer.

Figures 6 and 7 show examples of applications that were developed using JDeveloper and BI Beans and deployed either as a Java Client or a Thin (HTML) client. During the presentation the development process and applications possible with JDeveloper and BI beans will be demonstrated.

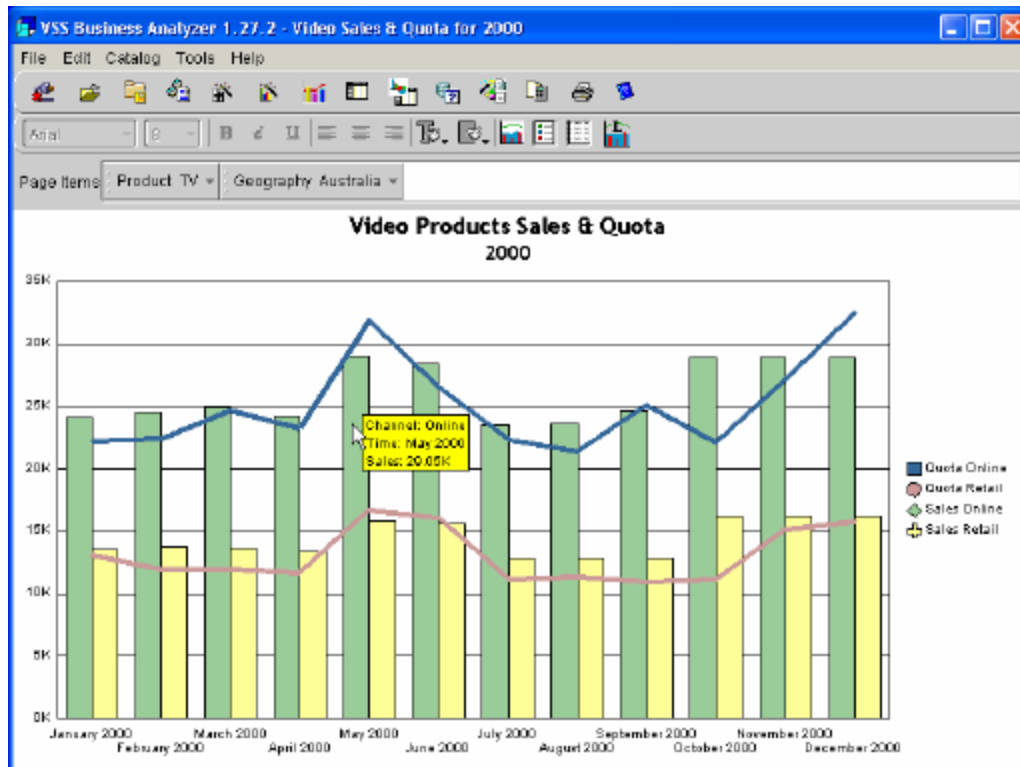


Figure 6. Thick BI Application (VSSBA)

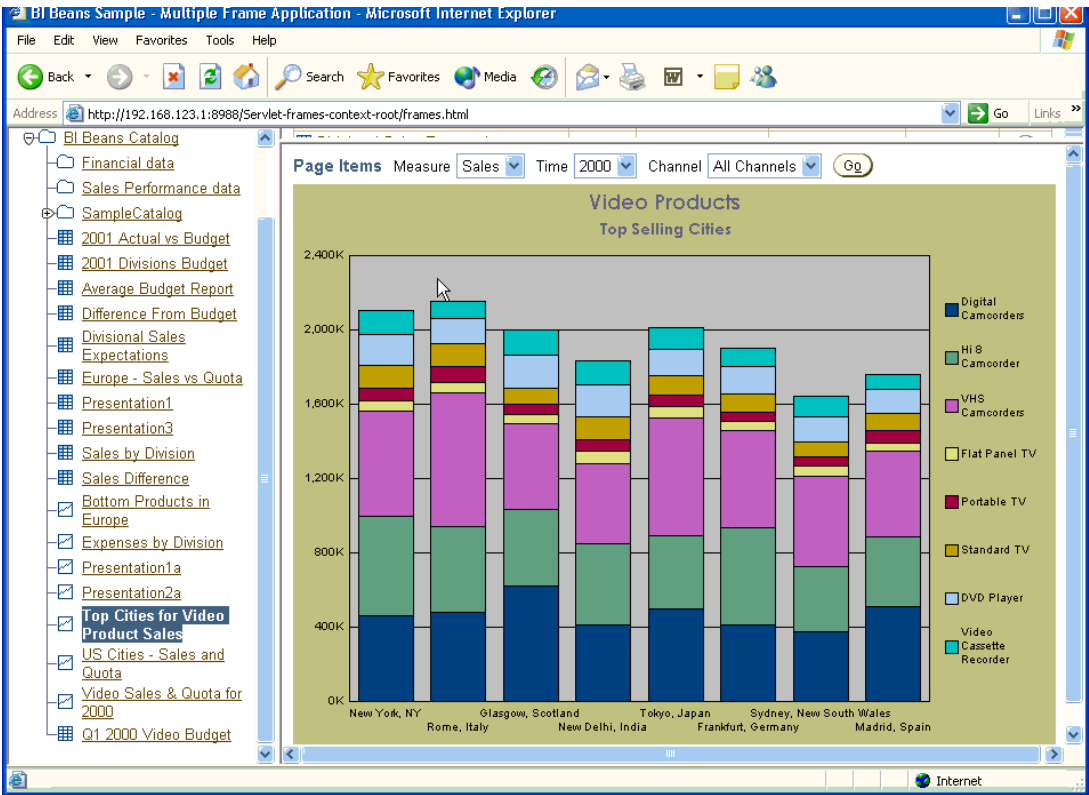


Figure 7. Thin BI Application