ORACLE
FUSION MIDDLEWARE

An Oracle White Paper
September 2014

Oracle Fusion Middleware MapViewer

ORACLE

Introduction

A picture, as they say, is worth a thousand words. This is particularly true when trying to capture the complexity of interactions among people, resources, products, and business processes distributed over geographic space. For many centuries people have relied on maps to capture and simplify these complex relationships, turning them into readily consumable, powerful packages of unambiguous information. With Oracle Database and Fusion Middleware products, the basic Oracle platform delivers powerful, universally understood geospatial capabilities to every developer.

Every Oracle Database includes extensive spatial capabilities in the Oracle Locator feature. Oracle Spatial and Graph option for Oracle Database Enterprise Edition expands Oracle Locator functionality with support for geocoding, routing, and advanced spatial data models to manage topologies, networks, linear referencing, gridded data and georaster imagery. Users can load all types of geometric data into Oracle Database, create spatial indexes, and issue spatial queries through SQL. Because of this, Oracle Database is an industry standard for managing geospatial data.

Oracle Fusion Middleware MapViewer complements the geographic data management capacity of Oracle Database by providing a generic web-based means of delivering and viewing any geospatial data in the database. This creates enormous potential for analyzing and understanding the geographic component(s) of any business, by unlocking the enterprise information in many corporate warehouses and making it available in easy to use mapping applications. For instance, business applications such as Field Service, Transportation and Logistics, Product Lifecycle Management, Human Resources, and Real Estate commonly render and visualize the massive amount of data they control because there is a geographic component such as an address tied to the data. Developers of location-based services, data publishers in state and local government, and architects of web services and more traditional applications can all easily integrate MapViewer into their web-based solutions.

Usage Scenarios

Common use cases include reporting (e.g. sales by region), proximity search (stores or service providers nearby), and analysis (asset locations and condition, and top N insured properties within projected hurricane track). Consequently MapViewer is integrated into enterprise applications such as work and asset management, field service, human capital management, inventory management, and business intelligence dashboards. Similarly it is used in web sites for store locators, coverage and jurisdiction information, or public works project information.

Reporting

A map is often used as a specialized chart or graph type to present information such as state-wise sales performance (Figure 1),

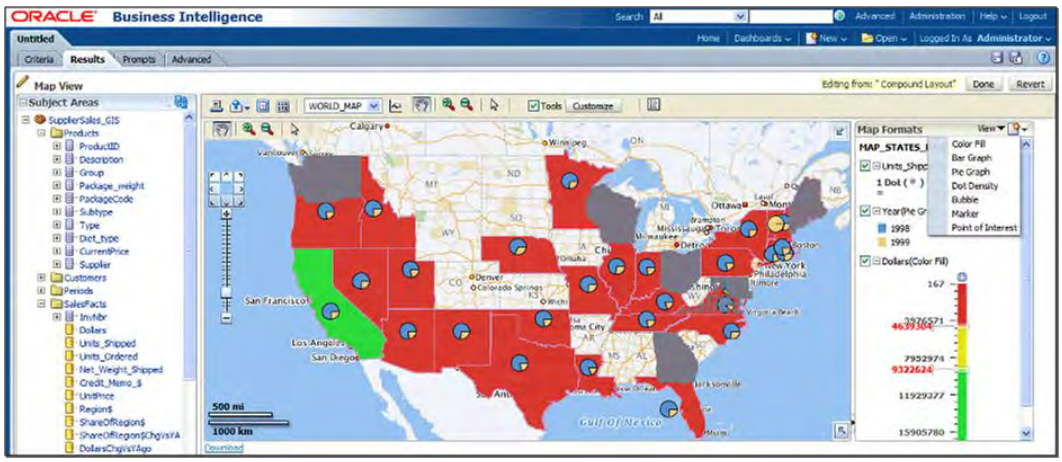

Figure 1: Sales Performance

or store locations and their current inventory levels (Figure 2 below).

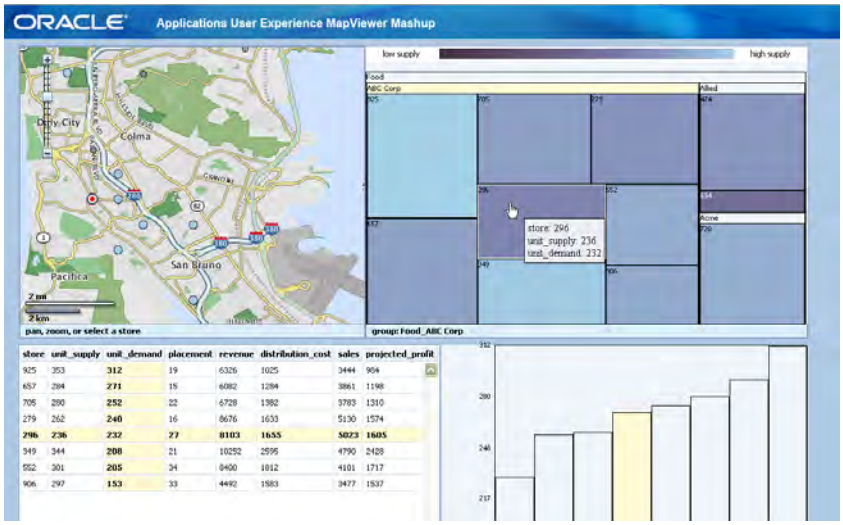

Figure 2: Store locations with inventory levels

Proximity Search

The following screenshots (Figure 3 and Figure 4) show the use of maps as a search interface. In both cases the map is used as an interactive UI component for first specifying search criteria and subsequently displaying the search results. In the first example (Figure 3) the user is searching for incidents that occurred within a specific distance from a location (a park) identified by clicking on the map.

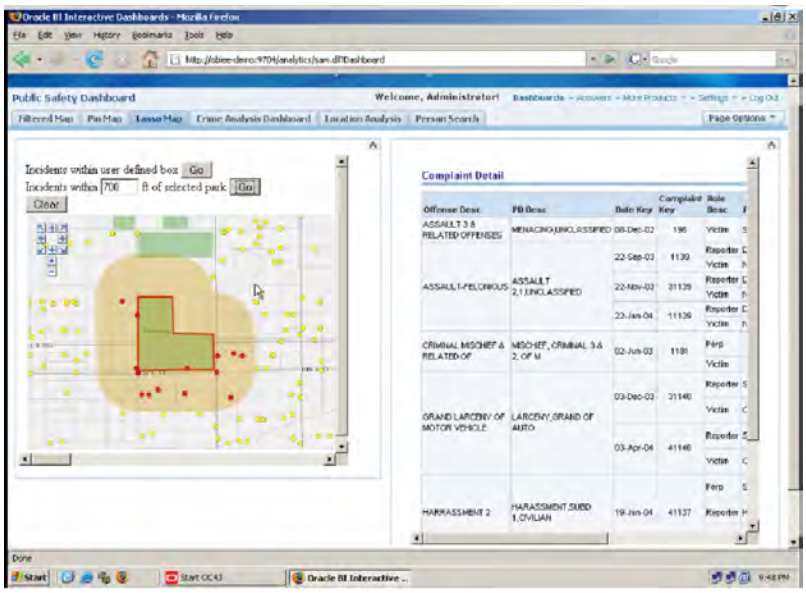

Figure 3: Incidents within a specific distance from a location

The second example (Figure 4) shows a summary (aggregate) report of demographic information for census areas within specified drive times from a point clicked on the map.

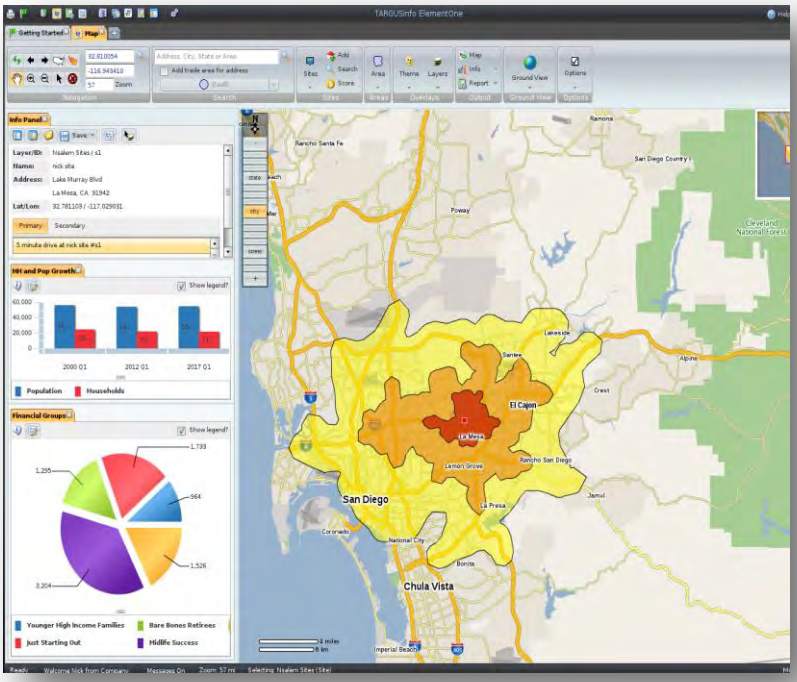

Figure 4: Summary report of demographics within specified drive times

Analysis

The second example above (Figure 4) is also an instance of displaying the results of a spatial analysis query (compute drive time polygons and aggregate census demographics) on a map. This section includes additional similar examples.

Figure 5 displays the result of a network analysis (traveling salesperson tour) that computes a near optimal tour between specified locations.

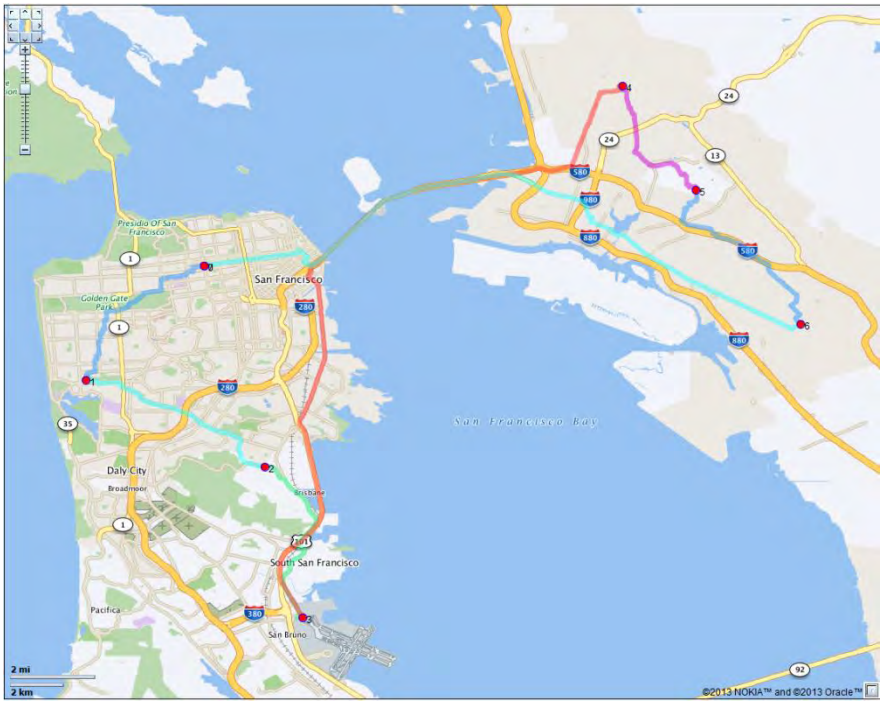

Figure 5: Traveling salesperson tour

The screenshot below (Figure 6) shows the result of an Asset Condition Score analysis for a given area of interest. The area of interest in this case is the current map extent or area displayed in the map window. The asset condition score is computed only for those assets (pipelines in this instance) that are within the current map extent.

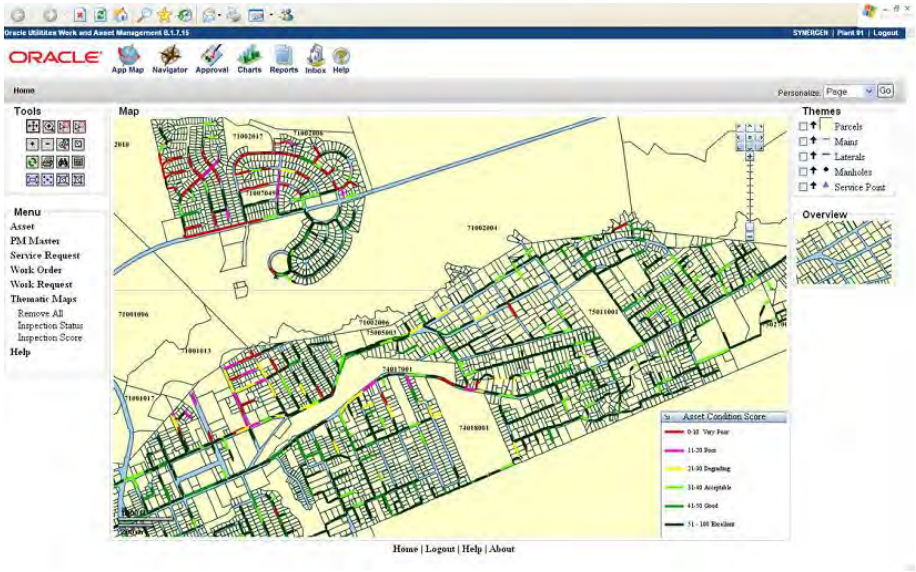

Figure 6: Asset Condition Score analysis in an area of interest

The next example (Figure 7) shows how multiple metrics may be displayed on a map image. The user selects a section of the gaming establishment's floor from the layout map on the left and is presented with a heatmap and tabular report on the right. These show a summary report of the cash in, win/loss, and numbers of players at individual machines within the selected area of interest.

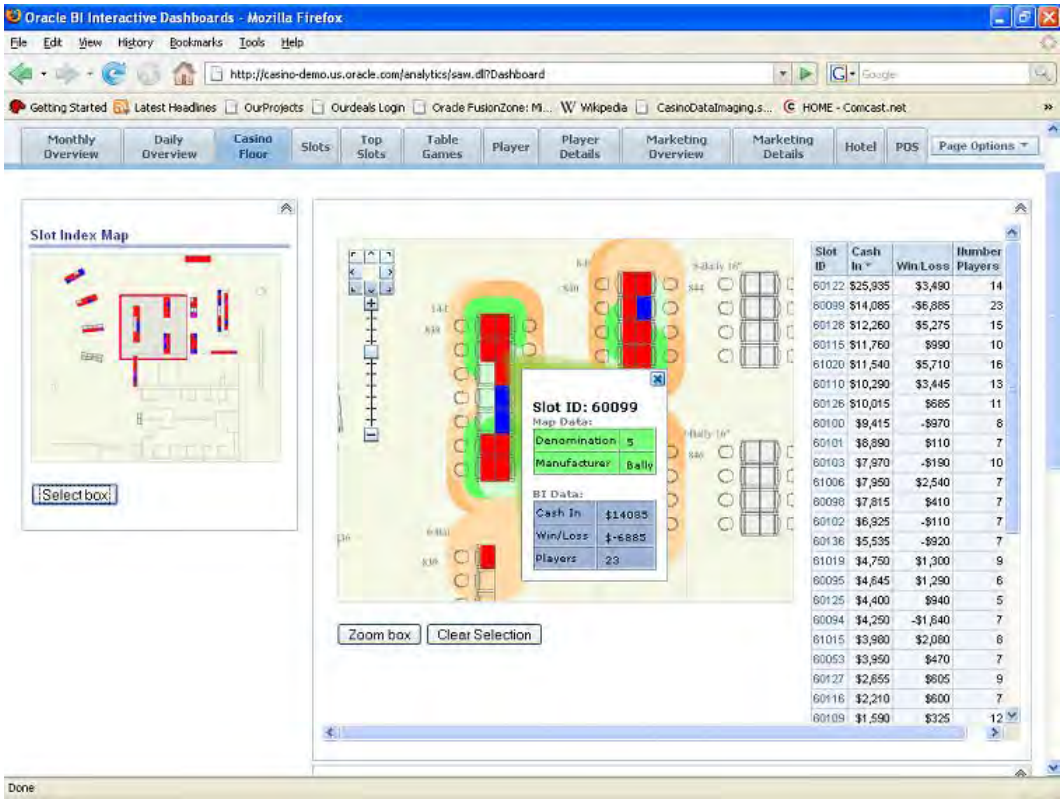

Figure 7: Floor plan layout map with heatmap and tabular report

Feature Overview

The MapViewer component in Fusion Middleware is written in Java and runs inside the Oracle WebLogic Server (WLS) or Glassfish Server. When it is up and running, MapViewer *listens* for client requests, which can range from map requests to administrative requests such as defining a data source or listing predefined maps in a data source. All requests will be sent using the HTTP POST method, with the content of the requests encoded in an XML format.

The companion white paper "Oracle Fusion Middleware MapViewer Primer" (OTN MapViewer Training page) and the MapViewer user guide provide further details on the technology and functionality. This section is a very brief overview of MapViewer functionality.

MapViewer includes a suite of technologies called Oracle Maps that consists of a map tile server, a dynamic geospatial query service known as the Feature of Interest (FOI) server, and a Javascript based web mapping client library.

The map tile server automatically fetches and serves cached map image tiles, rendered by MapViewer or other web-enabled map providers, to web application clients. The clients can then automatically stitch multiple map image tiles into a seamless large base map. Because the map

image tiles are pre-generated and cached, the application users will experience fast map viewing performance.

The feature of interest server renders spatial feature layers managed by Oracle Spatial and Graph, as well as individual geospatial features (points, lines, or polygons) created by an application. Unlike the cached image tiles, which typically represent static content, FOIs are dynamic and represent real-time database or application contents. The dynamic FOIs and the static cached base map enable you to build Web mapping applications.

The JavaScript mapping client is a browser side map display engine that fetches map content from the servers and presents it to client applications. It also provides customizable map-related user interaction control, such as map dragging and clicking, for the application. A sample Oracle Maps application is shown below.

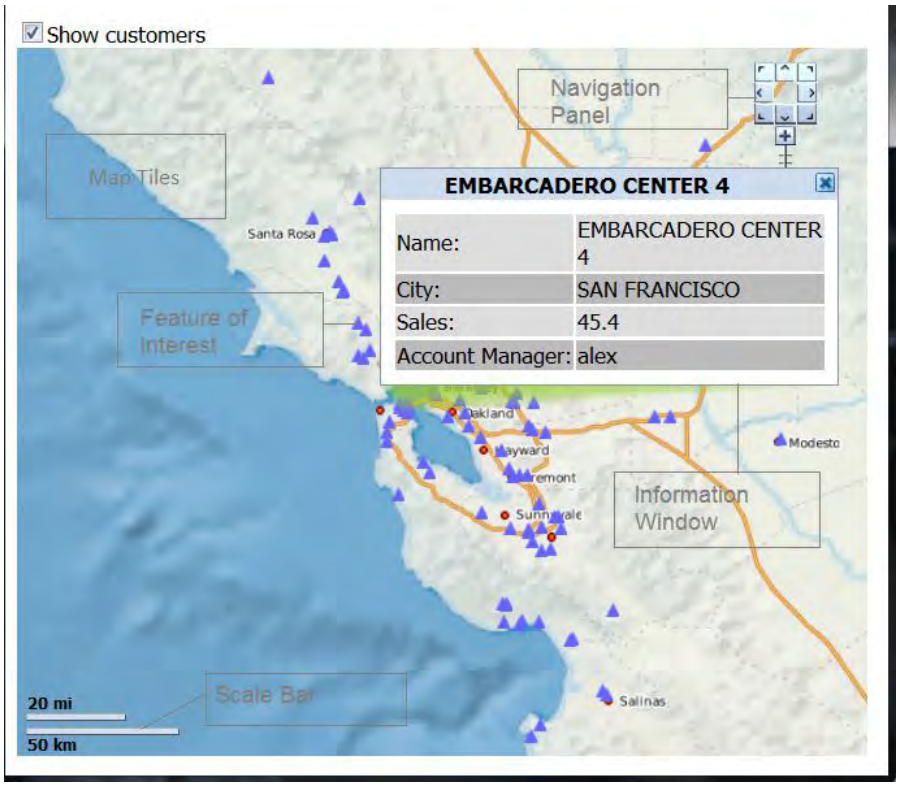

Figure 8: A Sample Oracle Maps Application

The sample application displays points of interest (customers) on a base map. The map consists of two layers:

- The base map layer displays the ocean, county boundaries, cities, and highways.
- The FOI layer displays customer locations as markers on top of the base map.

In addition to these two layers, a scale bar is displayed in the lower-left corner of the map, and a navigation panel is displayed in the upper-right corner.

The application user can use the mouse to drag the map. When this happens, new image tiles and FOIs are automatically fetched for the spatial region that the map currently covers.

Oracle Maps client applications running inside web browsers are pure HTML and JavaScript pages that do not require any plug-ins. Therefore, you can build the application using any web technology that delivers content as pure HTML. Such technologies include JavaServer Pages, Java Servlets, ASP, PHP, and .NET C#. MapViewer comes with over 50 tutorials illustrating various aspects of the Oracle Maps functionality.

Besides the basic functionality of high performance interactive rendering of geospatial content stored in an Oracle Database there are interfaces that help developers extend and enhance MapViewer. Some of them are:

- Custom Map Tile Provider interface
  The Oracle Maps API provides a generic get map tile interface. This lets developers use map tiles from a 3rd party provider (such as Nokia) in their MapViewer application. So the base map or imagery could come from an external online service while the user or application specific content (e.g. Features of Interest) is retrieved from an Oracle database.
- Custom (external) Attribute Data Provider Support
  MapViewer now supports middle-tier joins of external attribute data (such as sales) with geometries stored in Oracle Spatial and Graph, providing even more thematic mapping possibilities.
- Custom Spatial Data Provider Support
  MapViewer provides a generic spatial data provider interface. This interface lets developers extend MapViewer to work with file-based data sources such as ESRI shapefiles or other database software that manages geospatial information.

The following section briefly describes MapViewer application development with Oracle tools such as JDeveloper or Application Express (APEX). The subsequent section presents examples of using some of the features outlined above in this section.

Developing MapViewer Applications

MapViewer has XML, Java, and Javascript APIs for application development. These, combined with Oracle development tools, make it simple to develop web-mapping applications. There are two technical white papers (one each for the Java and Javascript APIs) titled "Developing Spatial Applications using Oracle Spatial and MapViewer". Similarly, the how-to note "Oracle Application Express Plug-in for AJAX Maps: Map Integration the Easy Way" covers the basics of including web-mapping functionality in an APEX application.

Oracle JDeveloper contains a rich feature set of rapid application development components known as the Application Development Framework (ADF) and associated set of Data Visualization Tools (DVT), or components, such as dials, gauges and charts. ADF includes Java Server Faces compliant components known as ADF Faces that include data visualization components. The DVT GeoMap component is one of them. It exposes the MapViewer Javascript API functionality and handles all JSF related event, partial page refresh, component communication, synchronization, and other infrastructure services. The following JDeveloper screenshot shows the DVT GeoMap drag-and-drop component listed in the Component Palette menu and the Create Geographic Map dialog that appears at design time.

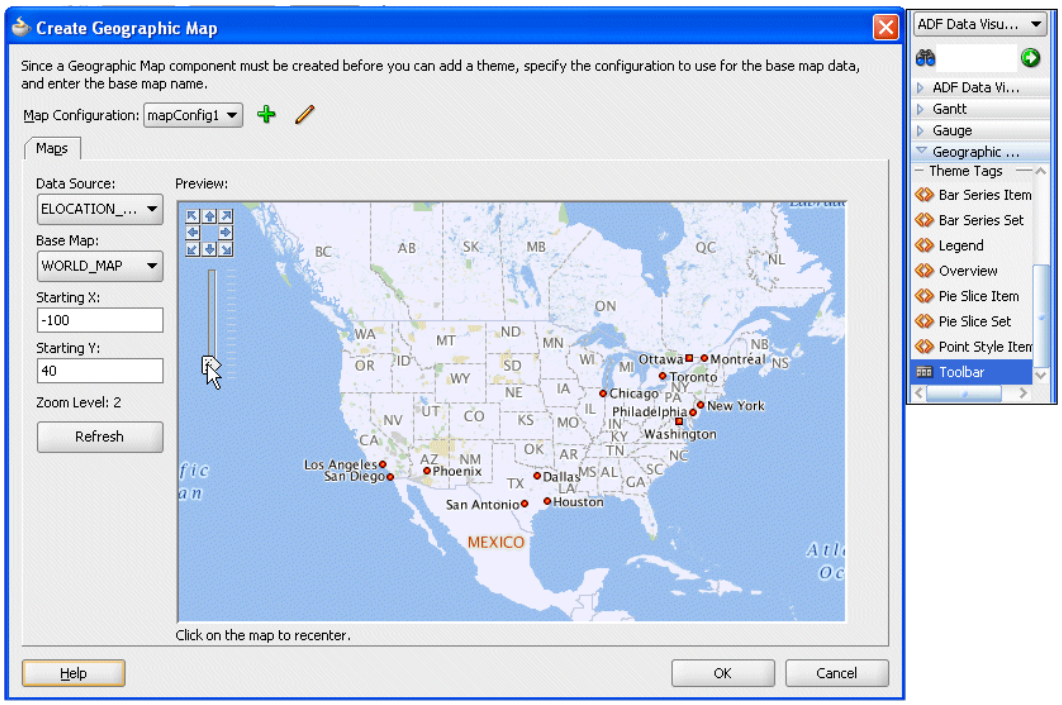

Figure 9: JDeveloper "Create Geographic Map" dialog and component menu

A step-by-step tutorial, "Charting with ADF Data Visualization Components - Graphs, Gauge, Maps, Pivot Table and Gantt", is available on OTN at http://docs.oracle.com/cd/E18941_01/tutorials/toc.htm.

Examples of Using Specific Features

This section briefly describes how specific features such as the JavaScript API and methods to support external data sources may be used in applications. A companion document ("Using MapViewer Interfaces for External Data Sources") provides technical details (i.e. the specific methods and sample code) and is also available from the MapViewer page on Oracle Technology Network (http://www.oracle.com/technetwork/middleware/mapviewer/overview/index.html).

JavaScript API

The Oracle Maps tutorial is part of the MapViewer quickstart kit on OTN (the mvdemo samples application) and contains over 50 examples demonstrating the API. The API enables location services application such as the one shown below. It is a screenshot of a demo application accessible at http://slc02okf.oracle.com/mvdemo/demo/mapviewer_jsp_demo.jsp, on deployment of the mvdemo samples application (i.e. mvdemo.ear file). The screenshot shows a thematic map of states based on the number of injuries due to hail in 1996. A customizable info-window pops up if the user clicks on a state.

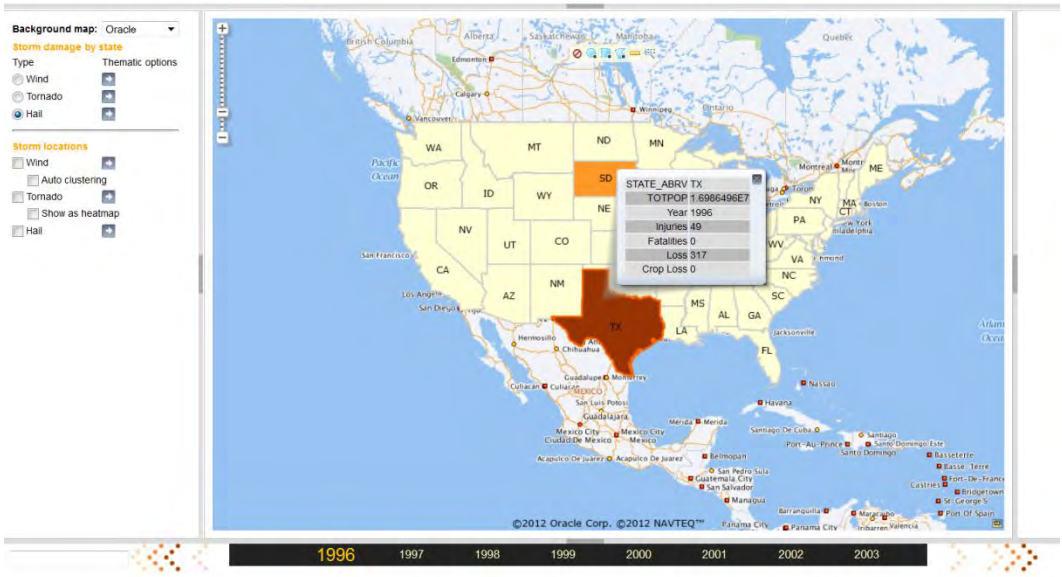

Figure 10: Demo application using the Javascript API

Custom Map Tile Provider

MapViewer renders geospatial content, such as street networks or satellite imagery, stored in an Oracle Database. Sometimes users may prefer to render only application-specific content, such as customer or business partner locations, on background maps or imagery obtained from an external source. An example of this is high-resolution imagery from DigitalGlobe's online service as shown in Figure 11.

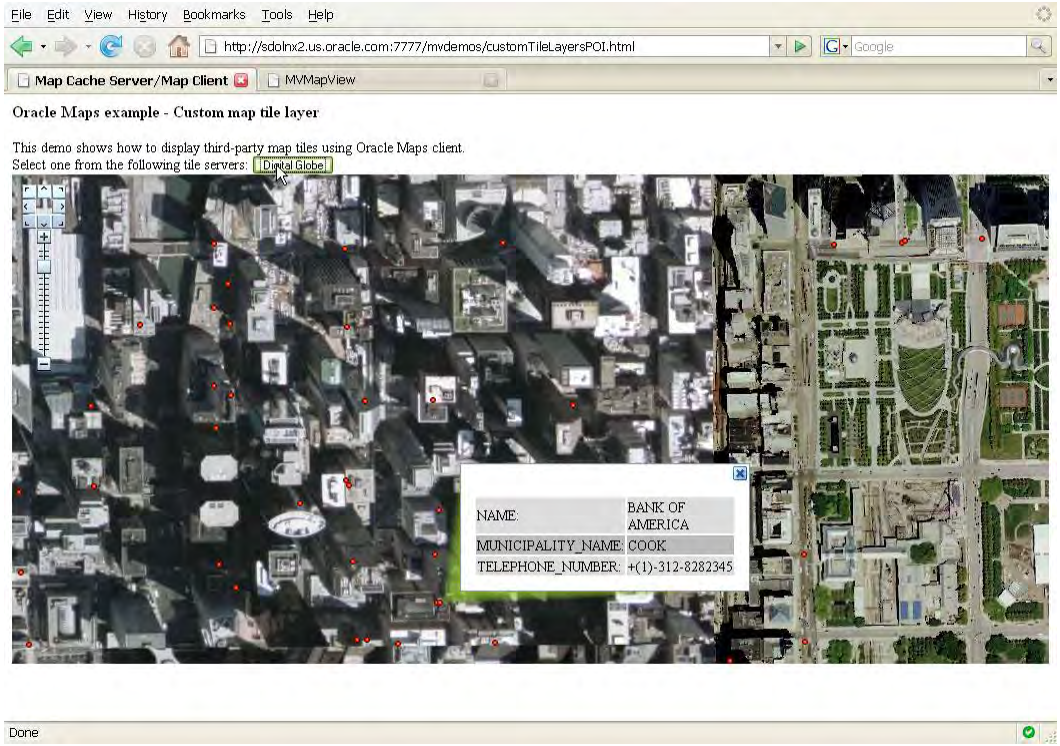

Figure 11: DigitalGlobe high-resolution imagery as background map

GeoRSS (and web services) Support

MapViewer supports GeoRSS (www.georss.org) and the OpenGeospatial Consortitium's (OGC) Web Feature Server (WFS) and Web MapServer (WMS) standards. This allows users to subscribe to a GeoRSS feed and display the locations and associated content on a map. In the example below, the map includes geotagged content such as photos from Flickr, restaurant reviews from Yelp, or Wikipedia articles related to an area around a postal code.

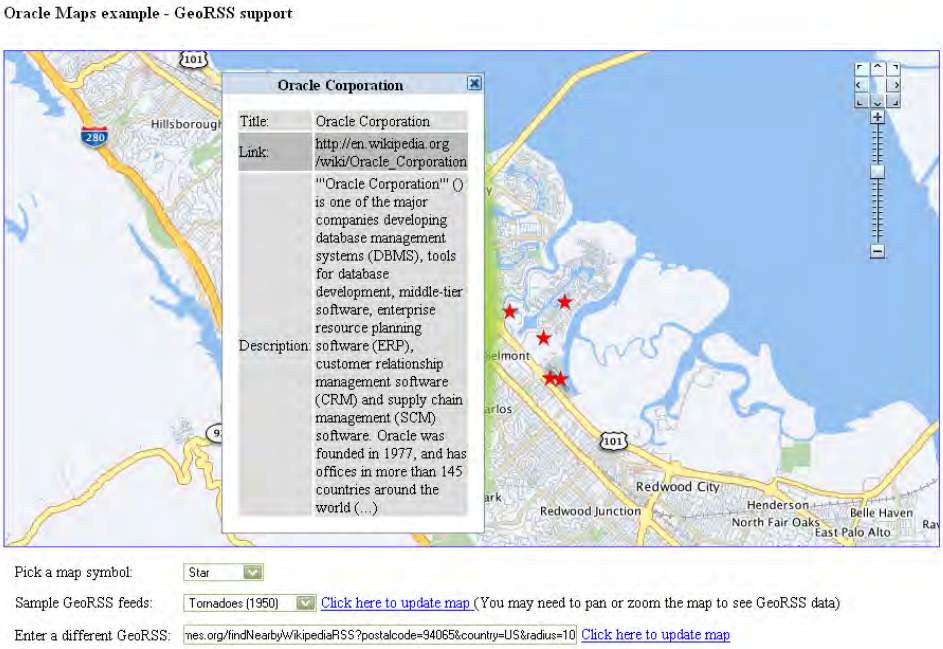

Figure 12: Geotagged content including GeoRSS Feeds

External Attribute (non-spatial) Data Provider

Often the data to be displayed on a map does not contain geographic coordinates and may not even be stored in an Oracle Database. This data can still be displayed on a map, however, if there is some means of linking or joining it to geospatial data via a common attribute such as a place name, identifier, or street address. MapViewer provides an interface for this purpose. It is commonly used with applications based on the Oracle Business Intelligence Enterprise Edition platform or JDeveloper where the data source is not necessarily an Oracle Database. It is particularly useful for BI applications when used in conjunction with the JavaScript APIs of both (OBIEE, MapViewer) products.

The presentation, "Map Views in Oracle Business Intelligence Enterprise Edition 11g", and accompanying workbook, "Building Map-Based Dashboards" (available on OTN) discuss the functionality and technical details. The workbook contains a step-by-step guide for common

usage scenarios: (i) displaying thematic maps in reports, (ii) adding charts to a thematic map, (iii) invoking a detail report for a location or region selected on the map (known as master-detail linking in OBIEE), and (iv) using Action Links in thematic maps (i.e. performing an action such navigating to a different dashboard page or a detail report).

The following screenshot shows a map view. The OBIEE Answers platform is mashing up project information, stored in spreadsheets, with a sales territory map based on the sales region name.

Figure 13: Oracle Business Intelligence Enterprise Edition with MapViewer

The same external attribute, or non-spatial, data provider interface has been used to mashup content from non-Oracle data sources such as Access Database, or reports published as XML documents, with Oracle Spatial and Graph data and rendered with MapViewer.

Conclusion

MapViewer provides web application developers a versatile means to integrate and visualize business data with maps. It uses the basic capability included with Oracle Database (either Oracle Spatial and Graph or Locator) to manage geographic mapping data. It hides the complexity of spatial data queries and the cartographic rendering process from application developers. They can easily integrate MapViewer into their applications. This creates enormous potential for understanding and capturing the geographic component(s) of any business, by unlocking the enterprise information in many corporate warehouses and making it available to basic mapping applications.

Oracle Fusion Middleware MapViewer
September 2014
Author: Jayant Sharma
Contributing Authors: Carol Palmer, LJ Qian, Charles Murray

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

oracle.com

Oracle is committed to developing practices and products that help protect the environment

Hardware and Software, Engineered to Work Together