Tushar Gadhia
Senior Product Manager
Server Technologies Division
Oracle Corporation
Web Enabling PL/SQL Based Database Applications
Introduction

PL/SQL is an easy-to-use, high-performance procedural language extension of SQL. Many Fortune 500 companies have successfully used the PL/SQL platform to build mission-critical applications. In one of the customer surveys done in 1998, it was voted as the most important and the most satisfying feature from Oracle. It was also voted as being very important for businesses in the next several years.

The evolution of the World Wide Web has led to an explosion of information in this ever changing world. This presentation focuses on how to web enable PL/SQL based database applications.
Traditional PL/SQL-based Database Applications
Traditional PL/SQL-based Database Applications

In a typical client/server setup, the server components are on a single server. There are multiple clients machines which run the GUI components and they access the server which contains the PL/SQL business logic, and the data. The figure above depicts a typical client/server architecture application.

Several database applications are client/server applications developed with Oracle Developer. In general, your data and associated business rules written in PL/SQL are kept on the server side. The graphical user interface (GUI), a set of Oracle Developer components, is deployed on the client side. The client machine accesses the server via SQL*Net or Net8. The GUI presents business data to users and provides ways for users to manipulate it. The client side may also contain logic, written in PL/SQL, to validate or normalize user input before it is sent to the server.

Rather than Oracle Developer, many customers use either PRO*C, OCI, or Java on the client side, with or without a custom GUI. These applications still tend to follow the basic client/server structure, keeping the data and business logic on the server and the user interface on the client.
Web-enabling PL/SQL-based Database Applications
Web-enabling PL/SQL-based Database Applications

Web-enabling your PL/SQL-based applications typically involves creating a new browser-based front-end for the application. The new front-end interface communicates with the back-end server via HTTP and Net8/SQ*Net. Generally, a browser based interface can coexist with your current UI, since the basic server side infrastructure is the same for each configuration. The above figure illustrates the coexistence of the old and the new interfaces.

If the architecture of your application is well-designed, your PL/SQL business logic in the server will remain unchanged as you Web-enable your applications. Furthermore, your current business process should remain unaffected as you Web-enable your business applications. If the data, user interface, and business logic are interspersed, the task of web-enabling the application can be significantly more difficult.
Web-enabling PL/SQL-based Database Applications

There are a number of ways to Web-enable your existing PL/SQL-based database applications. Each of them has its own system requirements and benefits, and is suitable to certain application categories or deployment environments. You should consider your application needs and pick the one that is the most suitable to you. The figure above summarizes the matching of different application categories and the Web-enabling solutions.

This presentation focuses on three different application categories:

- Oracle Developer applications
- Custom-GUI applications
- Applications without GUI

Additionally, this presentation covers the following Web-enabling solutions for the above application categories:

- Oracle Developer Web-deployment
- Oracle WebDB
- PL/SQL Web Toolkit
- PL/SQL Server Pages (PSP)
- Java Applets
- CGI
Web-enabling PL/SQL-based Database Applications

To Web-enable any kind of PL/SQL application, your environment should consist of some or all of the following:

- A browser, like Netscape, and so on
- A Web Server, such as the Oracle Application Server (OAS), or the WebDB Listener. The WebDB Listener is a very lightweight Web server, with a built-in PL/SQL Web gateway. The OAS also comes with a PL/SQL Web gateway.
- An Oracle database. The database typically contains the PL/SQL procedures to do the UI, the PL/SQL business logic, and the PL/SQL Web Toolkit packages that facilitate the development of Web applications.

Oracle WebDB provides a simple, lightweight PL/SQL gateway to execute your PL/SQL applications from the Web. It is suitable for a deployment environment with a small to medium number of users. The PL/SQL Web gateway of Oracle Application Server, on the other hand, offers load-balancing, fault-tolerance and high-availability features. It is suitable for the deployment of mission-critical applications with a large number of users.
Web-enabling PL/SQL-based Database Applications

Web-enabling Oracle Developer Applications
Web-enabling Oracle Developer Applications

When Web-enabling Oracle Developer applications, you can choose between a Java UI or an HTML UI.

Java UI

If you want to deploy your Oracle Developer applications on the Web immediately without putting in additional development effort, you should consider the Web-deployment feature available in Oracle Developer Release 6.0. In the Web-deployment mode, Oracle Developer executes the UI components of your applications as Java applets (i.e. downloadable Java applications) in your Web browser. Developer generates the Java applets automatically. Accordingly, a certified Java environment is needed on your client machines. Two possible Java environments are:

- Oracle JInitiator for running your applications from within a browser
- Oracle AppletViewer for running your applications from your desktop

**Note:** For more information on the Web-deployment of Oracle Developer applications, see the documentation for Oracle Developer, Release 6.0.

HTML UI

HTML is simple, easy, and light weight. If you decide that an HTML UI is more appropriate than a Java UI for your applications, you should consider using one of the following:

- Oracle WebDB
- PL/SQL Web Toolkit
- PL/SQL Server Pages (PSP)

Of the three options, WebDB currently provides the simplest and easiest Web application development environment.
Web-enabling Oracle Developer Applications with a Java UI

If you decide to Web-enable Oracle Developer applications using a Java UI, you can take advantage of the Oracle Developer Server and its three-tiered architecture.

In Oracle Developer Server’s three-tiered architecture the application logic and processing are performed on the middle-tier application server.

**Form Server**

The Form Server uses a three-tiered architecture to display applications to the user in a Java applet. The application logic and the Forms Server reside on a machine in the middle-tier. The application display is rendered by the Java client.

**Reports Server**

The Reports Server uses a three-tiered architecture to publish internet standard display formats (HTML, HTMLCSS and PDF). The report is run on the machine in the middle-tier, and the output can be viewed using a Web browser.
Forms Server Architecture

In this three tiered architecture the Forms Server executes the application on the middle tier while the application display is rendered by the Java client.

Role of the Forms Server

The middle tier Forms Server is an optimized and specialized application server for deploying applications on the internet. It also acts as a client to the database.
WebDB

- **HTML-based development tool** for building web pages using a browser-based interface
- Automatically generates scaleable HTML, optimized for the browser being used
- Contains build **wizards** that guide you through the process of creating:
  - HTML objects (forms, reports, charts, ...)
  - Web page design
- Executes **PL/SQL procedures** over HTTP with:
  - Web gateway in WebDB Listener
  - Web gateway in the OAS

Web-enabling Oracle Developer Applications with an HTML UI

If you decide to Web-enable Oracle Developer applications using an HTML UI, you have several options - WebDB, PL/SQL Web Toolkit, and PL/SQL Server Pages (PSP). This section discusses all of these options.

**WebDB**

WebDB is an HTML-based development tool for building HTML Web pages with content based on data stored in the Oracle database. Web pages can be created using a browser-based interface.

It includes a set of UI components, such as HTML forms, charts, reports, and menus. These are the basic building blocks for web-enabled database application development. WebDB also provides wizards to guide developers through the various steps in creating those components and linking them to the data sources. Its template mechanism enables developers to ensure a consistent look-and-feel for all of their applications. WebDB is especially suitable for existing Oracle Developer database applications which primarily use forms, charts and reports. These components can easily be re-created in HTML from the UI component library of Oracle WebDB.

There is no need to know PL/SQL to create a component. Selecting options in the build wizard guides WebDB to automatically write the PL/SQL code for building the component. The language used to implement WebDB applications is PL/SQL. PL/SQL can be executed with the PL/SQL gateway that is part of the Oracle Application Server (OAS), or the WebDB Listener.

The use of PL/SQL is totally transparent to WebDB users; however, the use of PL/SQL means that an experienced user could expand on the functionality implemented in WebDB applications and merge logic already in PL/SQL procedures into their WebDB applications.
WebDB

- Reduces the amount of Web and database knowledge required to build web-enabled applications
- Comes with its own lightweight runtime environment for executing PL/SQL based Web applications
- Customizability of the UI is restricted
  - For more control over UI, consider PL/SQL Web Toolkit, PSP

WebDB (continued)

A major benefit of Oracle WebDB is that it reduces the amount of Web and database knowledge required to produce a professional, functional web-enabled database application. It handles most of the technical details of HTML and HTTP for the developers, allowing them to focus on the application logic and presentation. A disadvantage of Oracle WebDB is that the customizability of the UI is somewhat restricted. Developers who want complete control of the UI should consider developing their UI programmatically with PL/SQL Web Toolkit or PSP.

By building on the solid foundation of PL/SQL, WebDB applications can easily be expanded with PL/SQL and still leverage any existing investments in application logic.

WebDB comes with its own lightweight runtime environment for the execution of PL/SQL-based Web applications. It is especially suitable for existing Oracle Developer applications which primarily use forms, reports, and charts.

Note: For more information on the WebDB cartridge or Oracle WebDB in general, please refer to the product documentation.
WebDB - How Does it Work?

WebDB can be used to create UI components quickly and easily. Use a browser and access WebDB. For example, you may want to create a form component using WebDB. Simply follow the wizard steps to quickly create the form of your choice. When done, store the form in the database. The appropriate PL/SQL code also gets generated at this stage.

Using a browser, a user can access the form created above by specifying the URL to execute the stored procedure to generate the associated HTML form.
PL/SQL Web Toolkit

If you are comfortable programming in PL/SQL and want to have complete control of the Web UI of your applications, you should consider developing your UI directly in PL/SQL using PL/SQL Web Toolkit.

PL/SQL Web Toolkit is a set of PL/SQL packages designed to facilitate the development of Web applications. Introduced originally in Oracle Web Agent (OWS 1.0), the toolkit has evolved into a complete package of useful functionality for Web application development. It includes the following PL/SQL packages:

• HTP/HTF – PL/SQL procedures and functions which generate HTML tags, allowing the development of HTML pages with very little knowledge of HTML syntax.

• OWA_COOKIE, OWA_IMAGE, and OWA_UTIL – routines for interaction with Web browsers via HTTP, such as the use of HTTP cookies, image map input, and CGI environment variables.

• OWA_TEXT – utilities for managing large amount of text.

• OWA_PATTERN – useful for locating patterns and substituting strings in text.

• OWA_OPT_LOCK – implements an optimistic locking scheme that enables PL/SQL users to check if a row has been modified before attempting to update it.
Developing a Web UI for your database applications programmatically with PL/SQL Web Toolkit involves writing PL/SQL stored procedures which generate the UI. The procedures are saved in files with a "sql" extension. These files are then written to the Oracle database.

The stored procedures, specified in the URLs of the applications, are invoked by a Web browser via HTTP. User input is passed as parameters to the procedures being invoked. The procedures make calls to your existing PL/SQL business logic, retrieve or update business data, and finally call the HTP/HTF packages to generate the Web UI in HTML.

**Note:** PL/SQL Web Toolkit is part of both Oracle WebDB and Oracle Application Server. In an upcoming release, it will also be shipped as a standard part of the database, making use of a web gateway embedded directly in the database.
PL/SQL Web Toolkit - Example
PL/SQL Web Toolkit - Example (continued)

The screen above shows portions of the code to generate the form on the previous page using PL/SQL Web Toolkit. Note that the dotted lines in the code above indicate that there is more code involved, but for the sake of showing relevant pieces of code, these lines of code were replaced with the dots.
PL/SQL Server Pages (PSP)

- Oracle’s PL/SQL scripting solution for server-side Web application development
- A PSP page is a server-side Web page with embedded PL/SQL scripts in HTML pages, marked up with special tags
- Allows you to include dynamic content, including the results of SQL queries, inside Web pages
- You can author the web pages in a script-friendly HTML authoring tool, and drop the pieces of PL/SQL code into place

PL/SQL Server Pages (PSP)

To some experienced PL/SQL developers, writing PL/SQL stored procedures to generate Web UI may prove to be interesting and challenging. Others, who may be more familiar with HTML than PL/SQL might want to produce the Web UI without a lot of programming effort, perhaps using a visual HTML editor of some sort.

In Oracle Rel. 8.1.6, there is a new PL/SQL scripting functionality called PL/SQL Server Pages (PSP). This functionality enables the development of dynamic Web pages for HTTP and HTML based applications. Scripting with PL/SQL Server Pages (PSP) is the appropriate tool for the HTML programmers.

A PSP page is an HTML page with embedded PL/SQL scripts marked up with special script tags. These little scripts (referred to as scriptlets) contain logic which is executed on the server side to produce the dynamic content of the page as it is requested by a Web browser.

Scripting in HTML pages fulfills the demand for a rapid development environment of dynamic Web pages. It allows scriptlets to be embedded in HTML pages while the HTML pages still look like HTML pages. Usually, these scriptlets will perform database queries or invoke other PL/SQL logic to retrieve data and display it in the Web page.
PL/SQL Server Pages (PSP)

- Embedding the PL/SQL code in the HTML page that you create lets you write content quickly and follow a rapid, iterative development process.
- You maintain central control of the software, with only a web browser required on the client machine.
- Because the processing is done on the server -- in this case, the database server rather than the web server -- the browser receives a plain HTML page with no special script tags, and you can support all browsers and browser levels equally.

PL/SQL Server Pages (PSP)

Server-side scripts are executed on the server when pages are requested by the browsers. They are used to process user input, generate dynamic page content that is customized according to the user input. Server-side scripts are "consumed" on the server-side and will not appear on the client-side.

The browser receives a plain HTML page with no special script tags. Scripting with PSP offers many advantages. It is suitable for quick-prototyping of Web applications, allowing developers to first create the skeleton of the applications with plain HTML pages, adding scriptlets for dynamic content later. Because a PSP page is an HTML page, you may use your favorite HTML authoring tool to create and edit the basic layout of the page, adding the dynamic content generated by the embedded scriptlets after the page has been created.

Finally, the scripting approach to dynamic Web page development encourages the separation of application logic (embedded scriptlets) from the presentation logic (static HTML content). Such a separation makes PSP pages easier to develop, debug and maintain.
PL/SQL Server Pages (PSP)

To develop and deploy PL/SQL Server Pages, you need:

- Oracle server ver. 8.1.6 or later
- A PL/SQL web gateway:
  - WebDB PL/SQL Gateway
  - OAS PL/SQL Cartridge

PL/SQL Server Pages (PSP)

As mentioned earlier, the PSP functionality will be available with Oracle server Rel. 8.1.6 and higher. To develop and deploy PSP, you need a PL/Sql Web gateway. Currently, you can choose between the following gateways:

- WebDB PL/SQL Gateway
- OAS PL/SQL Cartridge
PSP - How Does it Work?

Writing a PL/SQL Server Page

A PSP page is a server-side Web page (in HTML or XML) with embedded PL/SQL scripts marked up with special tags.

You can start with an existing web page, or with an existing stored procedure, using an editor. With a few additions and changes you can create dynamic web pages that perform database operations and display the results.

It can contain whatever content you like, with text and tags interspersed with PSP directives, declarations, and scriptlets.

The PSP file you create this way must have a “.psp” extension.

Loading the PSP as a Stored Procedure

From the command line, use the `loadpsp` command to load one or more PSP files into the database as stored procedures. Each .psp file corresponds to one stored procedure.

The pages are compiled and loaded in one step, to speed up the development cycle. They are compiled into PL/SQL stored procedures.

Calling a PSP

Once a PSP has been turned into a stored procedure, you can call it by retrieving an HTTP URL through a Web browser.
PSP - Example
PSP - Example

The screen above shows portions of the code to generate the form on the previous page using PSP. Note that the dotted lines in the code above indicate that there is more code involved, but for the sake of showing relevant pieces of code, these lines of code were replaced with the dots.
Choosing Between PSP and the PL/SQL Web Toolkit

- Use PL/SQL Web Toolkit if:
  - You have a large body of PL/SQL code that produces formatted output, producing HTML tags may be more convenient by changing print stmts. to call the HTP package of the toolkit
  - You use minimal tagging around query results

- Use PSP if:
  - You have a large body of HTML and want to include dynamic content
  - You want to get precise control over HTML tagging
  - Most work is done using HTML authoring tools
Web-enabling PL/SQL-based Database Applications

Web-enabling Applications with a Custom GUI
Web-enabling Applications with a Custom GUI

If your applications use a custom GUI instead of the Oracle Developer GUI, you still have a choice between an HTML or a Java UI when you Web-enable them. If your applications require complex graphic capability or user interaction, you may develop your Web UI in Java.

If you decide to use an HTML UI for your applications, you can use:

- Oracle WebDB
- PL/SQL Web Toolkit
- PL/SQL Server Pages (PSP)

Note: The discussion earlier in this presentation on how to Web-enable your Oracle Developer applications with Oracle WebDB, PL/SQL Web Toolkit and PSP above still applies here to any application with proper separation of UI from data and business logic.
Web-enabling Applications with a Custom GUI

When Web-enabling an application with a Custom GUI, the environment is somewhat different if you use the Java UI option.

In such cases, you need to develop your Web UI as Java applets that will be downloaded to and executed in your users’ Java-capable Web browsers. Your Java applets make calls to your PL/SQL business logic on the server using JDBC, the Java database API, and SQL/J. For more details on Java, see the documentation for the Java programming language and Oracle JDBC drivers.

A user needs to specify the appropriate URL in a browser to bring up the application.
Web-enabling PL/SQL-based Database Applications

Web-enabling Applications with No GUI
Web-enabling Applications with No GUI

If your PL/SQL-based database applications do not have a GUI, you may still want to make your applications available on the Web so that your users may invoke the application from their Web browser. For instance, you may have an application that involves command line programming. How to Web-enable these applications depends on their nature. In general, you may consider deploying your applications as CGI executables in your Web server.

If your applications allow users to extract business data from the database and write it to a file in a certain format (e.g. Microsoft Excel spreadsheet), you may simply deploy your applications as CGI executables which return their output via HTTP. Your users will be able to retrieve and view the output inside their Web browsers. It may also be possible for you to convert your applications so that the output of your applications is formatted in HTML and the output can be presented nicely in a Web browser.

If your applications allow users to upload business information to the database, you may design HTML pages with forms as the front-end interface to your applications, which will be converted to CGI executables. Your users will enter the input and hit the submit button to submit the information. It is possible to upload files from a Web browser using an HTTP file-upload mechanism. For more information on the topic, see books on Web servers, HTTP and CGI.
Web-related Oracle Supplied Packages
Web-related Oracle Supplied Packages

Some useful PL/SQL packages which allow database applications to interact with users or external programs on the Web:

- **UTL_TCP** – allows PL/SQL applications to communicate with external programs or network services using the TCP/IP network protocol. Many Internet standard application protocols, such as HTTP, SMTP, POP3, IMAP and NNTP, utilize the TCP/IP protocol.
- **UTL_HTTP** – allows PL/SQL applications to retrieve Web pages over HTTP.
- **UTL_SMTP** – allows PL/SQL applications to send electronic mail using SMTP.
UTL_TCP

- UTL_TCP is a PL/SQL utility that allows simple TCP/IP-based communication between the server and the outside world.
- UTL_TCP package consists of:
  - OPEN_CONNECTION - open conn. to service
  - CLOSE_CONNECTION - close open TCP/IP conn.
  - AVAILABLE - determine no. of bytes available for reading
  - GET_RAW, GET_TEXT, GET_LINE
  - READ_RAW, READ_TEXT, READ_LINE
  - WRITE_RAW, WRITE_TEXT, WRITE_LINE
  - ........

UTL_TCP

The UTL_TCP package enhances PL/SQL’s support for Internet computing. It allows simple TCP/IP-based communication between the server and the outside world.

By adding packages that allow PL/SQL applications to communicate over standard Internet protocols, Oracle 8i becomes an even more compelling platform for developing applications that closely couple database access with Internet connectivity. The addition of this package also allows developers to enhance their existing PL/SQL applications to take advantage of the Internet through the use of these standard protocols.

UTL_TCP consists of several functions. Some of the main ones are listed in the slide above.
UTL_TCP: Example

The above code example illustrates how the UTL_TCP package might be used to retrieve a Web page over HTTP. It connects to a Web server listening at port 80 (standard port for HTTP) and requests the root document.
UTL_HTTP

- UTL_HTTP is an Oracle-supplied package that makes Hypertext Transfer Protocol (HTTP) callouts from PL/SQL and SQL
  - Read URL --> Contact site --> Return data (HTML)
- Use it to access data on the Internet or to call Oracle Web Server cartridges
- UTL_HTTP consists of two entry points:
  - REQUEST
  - REQUEST_PIECES

Note: UTL_HTTP can also use environment variables to specify its proxy behavior. For example, on UNIX, setting the environment variable `http_proxy` to a URL specifies to use that service as the proxy server for HTTP requests. Setting the environment variable `no_proxy` to a domain name specifies to not use the HTTP proxy server for URLs in that domain.

In Oracle 8.1.6, UTL_HTTP supports HTTPS for secured HTTP communication.
UTL_HTTP.REQUEST - Example

The REQUEST function returns up to the first 2000 bytes of data retrieved from a given URL.

```sql
SVRMR> SELECT utl_http.request('http://www.oracle.com/')
FROM dual;

UTL_HTTP.REQUEST('HTTP://WWW.ORACLE.COM/')
<html>
<head><title>Oracle Corporation Home Page</title>
<!--changed Jan. 16, 19
1 row selected.
```

UTL_HTTP.REQUEST - Example

The REQUEST function returns up to the first 2000 bytes of data retrieved from the given URL. Its syntax is as follows:

```sql
UTL_HTTP.REQUEST (url IN VARCHAR2,
proxy IN VARCHAR2 DEFAULT NULL,
wallet_path IN VARCHAR2 DEFAULT NULL,
wallet_password IN VARCHAR2 DEFAULT NULL)
RETURN VARCHAR2;
```

If you are behind a firewall, include the proxy parameter.
UTL_HTTP.REQUEST_PIECES - Example

The REQUEST_PIECES function returns a PL/SQL table of 2,000-byte pieces of data retrieved from the given URL.

```plsql
DECLARE
    pieces utl_http.html_pieces;
BEGIN
    pieces := UTL_HTTP.REQUEST_PIECES('http://www.oracle.com/');
    FOR i in 1 .. pieces.count loop
        .... -- process each piece
    END LOOP;
END;
```

UTL_HTTP.REQUEST_PIECES - Example

The REQUEST_PIECES function returns a PL/SQL table of 2,000-byte pieces of the data retrieved from the given URL. It returns a PL/SQL table of type UTL_HTTP.HTML_PIECES. Each element of that PL/SQL table is a string of length 2,000. The final element may be shorter than 2,000 characters.

The elements of the PL/SQL table returned by REQUEST_PIECES are successive pieces of the data obtained from the HTTP request to that URL.

Its syntax is as follows:

```plsql
TYPE html_pieces IS TABLE OF varchar2(2000)
index by binary_integer;

UTL_HTTP.REQUEST_PIECES (url
    url IN VARCHAR2,
    max_pieces NATURAL DEFAULT 32767,
    proxy IN VARCHAR2 DEFAULT NULL,
    wallet_path IN VARCHAR2 DEFAULT NULL,
    wallet_password IN VARCHAR2 DEFAULT NULL)
RETURN HTML_PIECES;
```
UTL_SMTP

- UTL_SMTP allows application programs written in PL/SQL to communicate over the Internet using the Simple Mail Transfer Protocol (SMTP)
- UTL_SMTP package consists of:
  - OPEN_CONNECTION - open conn. to SMTP server
  - COMMAND - perform generic SMTP cmd.
  - HELO - perform initial handshaking w/SMTP server
  - EHLO - perform initial handshaking w/SMTP server with extended info. returned
  - MAIL - initiate mail transaction with server
  - RCPT - specify recipient of email msg.
  - ........

**UTL_SMTP**

This package allows application programs written in PL/SQL to communicate over the Internet using the Simple Mail Transfer Protocol (SMTP).

The purpose of this package is to support “Internet-aware” PL/SQL applications. This package will allow application developers to add significant value to their products by easily adding tight, standards-based integration to Internet email services.

UTL_SMTP consists of several functions. Some of the main ones are listed in the slide above.
UTL_SMTP - Example

The above code example illustrates how the UTL_SMTP package might be used by an application to send email. The application connects to an SMTP server at port 25 and sends a simple text message.
Summary

- Web-enabling existing PL/SQL-based applications preserves your investment in your information infrastructure while offering tremendous cost savings
- Oracle provides a wide variety of products and solutions to facilitate Web-deployment:
  - Oracle Developer
  - Oracle WebDB
  - PL/SQL Web Toolkit
  - PL/SQL Server Pages (PSP)
  - Oracle Application Server (OAS)