ORNL/CP-102577

USING THE PL/SQL CARTRIDGE OF THE ORACLE APPLICATION SERVER TO DEPLOY WEB APPLICATIONS

Connie L. Begovich, Oak Ridge National Laboratory

Introduction

Deploying business applications on the internal Web is a priority at Oak Ridge National Laboratory (Lockheed Martin Energy Research) and Lockheed Martin Energy Systems, Inc. as with most corporations. Three separate applications chose the Oracle Application Server (OAS), using the PL/SQL cartridge as a Web deployment method. This method was chosen primarily because the data was already stored in Oracle tables and developers knew PL/SQL or at least SQL. The Database Support group had the responsibility of installing, testing, and determining standard methods for interfacing with the PL/SQL cartridge of the OAS. Note that the term Web Application Server was used for version 3, but in this discussion, OAS will be used for both version 3 and version 4.

Applications

The applications currently deployed are:

- Travel Information System (TIS) - for entering information for travel requests, submitting the information to the travel agency personnel that make the necessary arrangements, entering expense accounts, and approving both the travel requests and expense reports.
- Delivery Tracking System (DTS) – for tracking material deliveries from the local receiving points to the local delivery points.
- Cylinder Information Database (CID) – for tracking cylinders of different types of wastes.

All three applications were originally deployed on version 3.0.1.8 of OAS on a Sun with Solaris 2.5.

Version 3 used a combination of listeners, PL/SQL cartridges, and Data Access Description (DAD) to deploy the applications. The listener receives requests from a certain port. If a virtual address that refers to a PL/SQL cartridge is specified in the request, then the appropriate PL/SQL package is executed using the specified DAD.

OAS 3.0 Installation and Setup

Installation of version 3.0 OAS was straightforward. The only problems we encountered was when trying to set up Netscape as an external listener. We initially set up a OAS listener to be on port 80, which is the default. When trying to install the Netscape listener, we were unable to complete the registration in OAS. We finally found that the OAS listener on port 80 had to be stopped and any process open on port 80 had to be killed before we could complete the external listener registration. In addition, because the Netscape listener was installed as root, we had to change protections on the Netscape root directories to allow oracle write access.

The different types of listeners that were installed in this first deployment are shown in the following table:

Listener	Port	Type of Listener	Used for
admin	8888	OAS	Administration
cossas	80	Netscape Enterprise	TIS
sslsas	443	Netscape Enterprise	TIS, DTS, CID
web1	8889	OAS	Oracle Forms Web Deployment (testing)

DISCLAIMER

Listener	Port	Type of Listener	Used for
admin	8888	OAS	Administration
webp	8890	OAS	Oracle Forms Web Deployment (production)

The DADs and their related PL/SQL cartridges were installed for the following environments and systems:

DAD Name	Environment	DAD user ID	OS and Oracle version
hostdevl	Development	none	AIX Oracle 7.3.2.3
hostqa	Testing	None	AIX Oracle 7.3.2.3
Traveldevl	Development	TIS's user ID	AIX Oracle 7.3.2.3
Travelqa	Testing	TIS's user ID	AIX Oracle 7.3.2.3
Travelprod	Production	TIS's user ID	AIX Oracle 7.3.2.3
Deltrkdevl	Development	DTS's user ID	NT Oracle 7.3.3.6
Deltrkprod	Production	DTS's user ID	NT Oracle 7.3.3.6
Ciddevl	Development	CID's user ID	Digital Unix Oracle 7.3.4
Cid	Production	CID's user ID	Digital Unix Oracle 7.3.4

Requirements for Database Pipes

Before deployment, we needed to meet the security requirements of Web application at Lockheed Martin Energy Systems/Research. A Web application must contain a timestamp. When a user submits a Web page, the application must determine if the same user that wrote the page is submitting it and if a given amount of time has passed since the page was written. If the timestamp of the Web page is beyond the limit, the user will be asked to log in again. Of course, if a timestamp is written clearly to the Web page, a user with just a little knowledge of the Web could edit the page source and submit it – therefore, bypassing the time check (or the user check). Therefore, we wrote a PL/SQL procedure to encrypt a string. The input parameters to the procedure include a user-defined string, an application name and key, and a check sum value. The procedure returns an encrypted string containing the user-defined string along with the application name, application key, and a timestamp. This string is written to the Web page. A submit from this Web page sends this encrypted string and a related decryption routine decrypts the strings if the same application name, application key, and checksum are passed to the decryption routine. Then, the decryption routine verifies that the time is within the desired range of the timestamp. Note that the application name, application key, and checksum are all used to determine that the same application is trying to decrypt the key that sent it, so that just having the encrypted string is not enough information for decryption. Note that the encryption method is certainly not DES-encryption, but is complicated enough to avoid easily breaking the code. However, because of some Internet browsers inability to store non-ASCII characters, the encrypted string had to contain only ASCII characters.

Also associated with security was which schema was going to be used to run the application PL/SQL routines. This is tied to how the Data Access Descriptors (DADs) of the OAS are set up. During development and initial testing, the DAD was set up so that each user (using what we call a user ID) had to logon to the database. Therefore, access was controlled by the database and each user had to have an account and execute access to the PL/SQL procedures/functions (remember that roles do not work for execute access to PL/SQL procedures/functions in Oracle 7.3). However, once the applications were deployed, the access was to be through a single user ID and password, which then could be stored in the DAD. Access to the PL/SQL procedures/functions can then be granted only to this user ID. To validate the user before allowing access to the application, a user ID validation server was available that could be accessed through C routines. Since Oracle 7.3 does not allow PL/SQL routines to call C routines directly, we had to use database pipes (dbms_pipe package). The PL/SQL side of the dbms_pipe accepts the user ID and related password, sends it to the dbms_pipe daemon running on database server, which then uses C and TCP/IP routines to call the user ID validation server. Note that encryption was again involved here to avoid user ID/passwords being sent in clear text across the network.

The combination of using the encrypted timestamp and validating the user independent of the database allows the application to re-prompt the user for user ID and password once the user has timed out of the Web application (has exceeded the time frame allowed between the current time and the encrypted timestamp on the string). In addition, the user ID can be encrypted in the string along with the timestamp, so that the application can be assured that the user is the same one that generated the screen. These routines are shared among the applications and the dbms_pipe daemon runs on both a AIX system and a Digital Unix system. The NT system , which was used to deploy the Delivery Tracking System, uses a database link to the AIX system to use the dbms-pipe daemon.

The dbms pipe daemon is being used for more than the user ID and password validation. One or more of the applications required other operating-system type commands.

- Validation of a charge number using a remote function call
- Retrieving data from SAP by logging on remotely
- Copying files from an oracle directory

Each of these are described below.

The remote function call was deployed similar to the remote function call to the user ID validation server. The Web application sends the request to a PL/SQL routine owned by oracle. The PL/SQL routine passes the charge number to the daemon through dbms_pipe utilities. The daemon validates the charge number through a remote function call and returns the status to the PL/SQL routine, which passes it back to the application.

In the second case, instead of using a remote function call, the data required was specific to the application. The application needed data from SAP, which is deployed on a completely different set of servers. In this case, a remote logon was set up to allow the dbms_pipe daemon to retrieve information from SAP. Unfortunately, the linkage of all these different libraries caused an incompatibility; therefore, an independent operating system command was set up to call the SAP routines. The dbms_pipe daemon calls a separate command, passing it parameters on the command line. Because a range of values is returned (e.g., the request is for all line managers of a certain individual), a record number is returned. The PL/SQL routine either selects the data from a temporary table or returns the record number to the application.

The applications also needed a method for reading/writing files from the database into specific UNIX directories. Oracle's utl_file package is the method for reading and writing to files, but once again there are some security holes. First of all, any file written or read the directories specified in the initialization file can be read or written by any valid user of Oracle. To restrict that access, execute access to the utl_file package was revoked from everyone except oracle. Then an interface was built to restrict a user ID to specific directories. Then to copy or delete the files, the developers gave oracle access to their directories through group access. The dbms_pipe daemon was used to copy or delete the files, also checking access to the Oracle directories. Note that the dbms_pipe daemon had to be linked when the developer group was active for oracle.

OAS 3.0 Problems and Solutions

Some of the problems encountered in OAS 3.0 and the solutions we provided were:

Problem	Solution
Client screen problems on Macintosh using a specific version of Internet Explorer	Use the Netscape external listener instead of an OAS listener
Getting the DB Login screen when the DAD contained a stored user ID and password	Restart the OAS; this occurred when someone using a DAD which did not store a user ID and password and entered an invalid user ID/password ; it caused the DB Login on all cartridges
WRB processes hanging and not going away	Use a shell script to find those processes and remove them plus automatically refresh the OAS processes every night; the hung processes occurred when someone would stop the processing from their desktop
When the number of WRB processes exceeded the	When it occurred, the OAS had to be restarted;

maximum allowed (50 in our case) for the PL/SQL cartridge by more than 5 or so, the OAS would start looping	removing the WRB hung processes helped prevent the condition

Migrating to OAS 4.0

The TIS and DTS have been migrated to OAS 4.0. There were several differences in this version and again we learned several lessons. These are described below.

The main difference in administration is that the combination of cartridges and DAD's are now applications in OAS 4.0. Another difference is that you no longer specify which listener can be used for the different applications, which was available in OAS 3. Therefore, if you want an application to only accept incoming requests from the SSL listener, you would have to build the logic in the application.

The Web admin interface to OAS 4.0 is an improvement over version 3.0, but required some learning and testing to determine and set the different options. The admin interface can be used for most operations; however, because we again installed the Netscape servers as root instead of oracle, the Netscape listeners have to be stopped and started from the root command line.

The owsctl command also has quite a few new options in version 4.0. In addition, log files can be set up in separate directories for each application. The setting of the severity level for error messages is also now defined at the application level.

The applications that were running in production in OAS 3.0 could be used without change in OAS 4.0. However, the 3.0 PL/SQL packages, such as owa_util, could not be used with OAS 4.0 and vice versa. The OAS 4.0 installs these packages under the schema oas_public, which is an improvement over the installer choosing and/or setting up the schema.

Lessons Learned

Only one DAD should be added for each application. Otherwise, errors were generated in the wrb.log file. Originally, we set up the development application to have DADs both with and without the stored logon user ID/password; however, that was abandoned fairly quickly.

Although the external Netscape listener was used very successfully with version 3.0, in OAS 4.0 the listener, using Enterprise 3.51, would not stay up. With any load at all, the listener went down with no apparent reasons. The wrb.log, the xlf.log, or the Netscape error logs did not contain any errors or warnings. Therefore, we abandoned using the external Netscape server except for SSL connections. Note that we also found that if any changes were made to the Netscape configuration files after they had been registered in OAS as external listeners, the external listener had to be unregistered and then reregistered. Otherwise, the listeners would not restart after the configuration changes.

In addition, the configuration and tuning of the OAS was difficult. The number of instances, threads, and hosts were varied until we found a working solution. Using threads seemed to introduce problems so the number of threads was set to 1 for all environments. The table below lists the current configuration of the TIS application in all environments (development, testing, and production) and of the DTS production environment. TIS production can be expected to have up to 50 concurrent users with hundreds of transactions submitted per day. DTS should have 2 or 3 concurrent users who consistently add and verify data.

Application	Environment	DAD Hosts		Cartridge Server Instances	
		Minimum	Maximum	Minimum	Maximum
TIS	Production	5	40	40	50
TIS	Testing	2	20	20	30

TIS	Development	2	20	20	30
DTS	Production	1	3	1	5

Another important parameter was the max session idle time (Site->Application->Configuration->Web parameter). The current configurations of all applications have it set to 0. Otherwise, if the user presses the Netscape stop button, the OAS process continues working. In testing, a new incoming request also tries to use that application process, even if there are other application processes running. When max session idle time is set, these processes will continue spinning until the max session idle time is reached.

Another problem is again due to the timeout/security issue. Timeout on each of these installations of Oracle is set to a specific number of minutes for all accounts. By default, the OAS stays logged onto the database host systems for at least the minimum number of hosts. If timeout passes without any CPU or I/O activity, the process is marked sniped, and Oracle 7.3 is not very good at cleaning up the sniped processes. If too many of these processes are sniped, the database system could reach its maximum number of processes that are licensed. Therefore, this factor needs to be considered when setting the maximum number of hosts or the Oracle timeout parameters of the DAD user.

Stopping and restarting the OAS every night is still being done even in version 4.0. This cleans up any OAS processes. However, be sure to schedule this when the databases referred to by the DAD's are available. If the database is down and the OAS is started, at least the first few tries for that application show the database as still down or the application just spins. This batch job is also used to copy the log files and remove some of the older log files.

Other tuning parameters, which did seem to have at least some affect on the speed of the applications, were setting the directory rescan and the process model. Setting the time limit for directory rescans (Site->HTTP Listeners->listener->Server) to a higher value speeds up the loading of static files, especially since graphic files were the primary static HTML stored on the OAS server in our applications. Using the non-collapsed version (Utilities->Process Model) seemed to help increase the application efficiency because the system had plenty of memory and swap space.

Future of OAS

The Database Support group is now a part of SAIC. If you have questions about the current configuration or setup, contact dbsupport@ornl.gov. The CID application is being moved to OAS 4.0. It requires a SSL listener, and the OAS listener is being set up that way instead of using the external Netscape server. In addition, there are other application being developed that uses the OAS. Even with the problems encountered, the OAS PL/SQL cartridge option is a viable option.