

Upgrading to 11.0.3.Something.Something (Or “Not For the Faint of Heart”)

Barbara Matthews *OnCallDBA.com*

Paul Greenwood *Ultradent*

Introduction

This paper will describe how Ultradent upgraded to Version 11.0.3 of the Oracle Applications from Version 11.0.2. Included is a description of how to clone the database and code from production to a test environment, as well as how to perform the upgrade. This paper will describe issues and problems that we encountered as we did this upgrade. This paper also describes how to deal with certain types of patch failures and issues that you might encounter when applying any Oracle patch. We’ve also described how we’ve set up our environment.

We’ll describe how to clone the TEST instance from ORAUPI, our production instance, and will also describe how to clone a third instance, DEVL. We’re doing this because maintaining two instances on one machine turned out to be a bigger hurdle than cloning the first. If you find the details excruciatingly detailed, take pity on your comrades in arms, because we’ve literally tried to make it so that a junior Unix SA/DBA/Applications SA could do a clone without too much head scratching.

We’ll confess right up front that we expect to continue to refine our refresh and upgrade process additionally after publishing this paper. We recommend, therefore, that you retrieve the latest version of this paper at www.oncalldb.com. Also, this paper describes the process that *we* use and can serve as a guideline for your own environment, but, as noted, we are still learning and refining this process. You should look at other relevant papers and Oracle Metalink documents before attempting to do refreshes and upgrades. Additional literature that we’ve found helpful includes: Metalink Note:91985.1 *Step by Step on Cloning the Oracle_Home (including db) and APPL_TOP* and *Successful Migration of Oracle Applications Data*, by Jim Basler, available at www.orapub.com.

Contents

Assumptions About Our Environment

Cloning the Production Database to Another Instance on Another Machine

Creating a Second Instance on Another Machine

Upgrading the New Instance

Assumptions About Our Environment and Rules of Thumb That We Follow

Our Environment:

- Our oracle Unix account is called oraupi, and our applmgr Unix account is called applupi on both our test and production servers. We have a legacy Oracle database that uses the oracle account.
- Our database instances are ORAUPI (production), TEST and DEVL.
- We have four servers, upi_db and upi_apps for ORAUPI (production), and upi_tdb and upi_tapps for TEST and DEVL.
- We plan to install an Applications 11i instance on our test environment next.
- Our TEST and DEVL instances share the same \$ORACLE_HOME. Our TEST and DEVL instances have two different \$APPL_TOPs, since TEST is running Oracle Applications Version 11.0.2 and DEVL is running Oracle Applications Version 11.0.3.

It is useful to keep a list available of key port information for each database:

TEST

WEB ADMIN PORT: http://upi_tapp:8878

FORMS_PORT: 8990

To log onto test apps: http://upi_tapp:7990/TEST_j.htm

\$ORACLE_HOME= /upid01/app/oraupi/product/805

\$APPL_TOP= /upid01/app/applupi/1102

DEVL

WEB ADMIN PORT: http://upi_tapp:8888

FORMS_PORT: 8991

To log onto devl apps: http://upi_tapp:8000/DEVL_j.htm

\$ORACLE_HOME= /upid01/app/oraupi/product/805

\$APPL_TOP= /ora_dev/ora_apps/d01/app/applupi/1102

PROD

WEB ADMIN PORT: http://upi_app:8878

FORMS_PORT: 8990

To log onto ORAUPI apps: http://upi_app:7990/ORAUPI_j.htm

\$ORACLE_HOME= /upid01/app/oraupi/product/805

\$APPL_TOP= /upid01/app/applupi/1102

Patches:

- NEVER apply patches to production during business hours. Apply patches when there are no users on the system or they will get error messages as files change underneath them.
- NEVER apply patches to production that haven't been thoroughly tested on a different database instance first.
- We hesitate to apply patches when our users are in a hurry to get back to work. Generally, we only apply patches to production on the weekends, and we only apply patches to our TEST and DEVL instances if users understand the risk that the patch may not apply correctly or quickly. This is called "living with Murphy's Law".

Hardware:

- Buy a test machine that has enough disk to support at least two instances – one instance will be a mirror of production (we call ours TEST), and the other will be used to test major upgrades and enhancements (we call ours DEVL). If you cannot support two instances, then you risk running into timing issues where you've done a major upgrade to your TEST

environment, haven't completed the testing yet, but need to test a small change or patch on a mirror of production before applying the major upgrade.

- Do not skimp on disk for your test machine. Our goal was to be able to clone the production database as quickly as possible. When we were short on disk we found ourselves making all kinds of tradeoffs and changes that slowed the cloning process and made it more complicated. Your TEST servers should look as much like your production servers as possible. As we do not do heavy-duty performance benchmark testing, we concluded that having sufficient disk was more important than matching up on CPU or memory for the TEST servers.
- Be sure that your TEST database server has a tape drive. Interestingly enough, it's pretty easy to buy a server without a tape drive. If you wait until after the initial purchase, the price of the tape drive stands out to management and may delay the purchase process. We found ourselves coming up with all kinds of convoluted procedures for doing refreshes because we initially didn't have a tape drive on the TEST servers.

Choices you get to make if you don't have a tape drive:

- A. Shutdown the production database and clone over the network
- B. Shutdown the production server, attach the tape drive to the test server, copy the data, stop the test server, and move the tape drive back to the production server. Uggh.

- Ever ask yourself if your TEST environment really needs to be backed up? We did too until the day we had a massive system failure that took out our boot disk and lost our recently upgraded 11.0.3 instance. Think carefully about whether you can really afford to lose work in your TEST environment, particularly if you're the one who'll have to redo all the upgrade work.

Cloning the Production Database to Another Instance

TEST

- The TEST instance was actually the first instance that we created. The database and concurrent manager are located on a server called `upi_tdb` and the web and forms code is located on `upi_tapp`. We used the Oracle Applications Version 11.0.2 One Hour Install to create this first instance, and have cloned it several times since from copies of production. TEST's layout looks exactly like ORAUPI's, with a mount point called `/ora_apps`.

DEVL

- If you have enough disk in place, the easiest way to create your DEVL instance is to use the One Hour Install and then clone the database and applications code from ORAUPI (production). As our disk came in over a period of time, we ended up following a more complicated process. OK, we'll admit it... we upgraded our TEST environment from Version 11.0.2 to Version 11.0.3, and then had an emergency that required that we put the newly upgraded environment onto tape so we could refresh TEST from the 11.0.2 version of ORAUPI again.

At this point, we lacked enough disk on our test servers to support two database instances. By this time, we had done the 11.0.2 to 11.0.3+ upgrade twice already, so there was no way we were going to use the 11.0.3 One Hour Install to create the DEVL instance and then apply all of the 11.0.3+ megapatches for a third time. Instead we created the DEVL instance by recovering from tape after our new disk was installed without using the One Hour Install.

Ultimately, we maintain two sets of application code, since we are maintaining a Version 11.0.2 and a Version 11.0.3 applications codeset on the same machine. This allows us to apply applications patches to the appropriate codeset. Currently we are maintaining two sets of RDBMS code.

- With more than one instance, you'll need to decide if you want to have two separate oracle and applmgr accounts, or if you want to use a menu to control access to the \$APPL_TOPs and \$ORACLE_HOME. We decided to use one oracle and applmgr Unix account (oraupi and applupi) with a menu called from each account's .profile. This allowed us to bring over code and database clones without changing the ownership on files. We created two separate mount points, /ora_apps for the TEST instance, and /ora_dev for the DEVL instance. We are still debating whether it is easier to use one account or two, and would welcome others' opinions about this concept.

Files We Maintain on upi_db (the Production ORAUPI Database Server):

Before beginning the refresh, review the TEST and ORAUPI application and database server configurations. We've done some pre-work to simplify our refresh process. After our first refresh, we made sure that certain files that we knew we had to customize each time we completed a refresh were already on ORAUPI. This made it so that every refresh carried those files over and eliminated the extra modification. Files we changed were:

1. Under ORAUPI's applupi home directory, we have:

ORAUPI_concmgr_deactivate.sh, ORAUPI_concmgr_start.sh, ORAUPI_concmgr_stop.sh, TEST_concmgr_deactivate.sh, TEST_concmgr_start.sh, TEST_concmgr_stop.sh, DEVL_concmgr_deactivate.sh, DEVL_concmgr_start.sh and DEVL_concmgr_stop.sh.

These programs stop, start and abort the concurrent managers for each of the three instances.

2. Under ORAUPI's \$APPL_TOP, we have DEVL.env, ORAUPI.env and TEST.env.

These are the environmental files for the three database instances we support.

3. Under ORAUPI's \$UPI_TOP, we have resp_TEST.sql and resp_DEVL.sql. resp_TEST.sql changes the responsibility names that include PRODUCTION!!! to TEST. resp_DEVL.sql changes the responsibility names that include TEST to DEVL.

The following example shows what resp_TEST.sql looks like when it runs successfully:

```
old:Oracle Training PRODUCTION!!!,new:Oracle Training TEST
old:UPI Sales Team Leader PRODUCTION!!!,new:UPI Sales Team Leader TEST
old:UPI Sales Manager PRODUCTION!!!,new:UPI Sales Manager TEST
old:UPI Trade Show Manager PRODUCTION!!!,new:UPI Trade Show Manager TEST
old:UPI Receivables User PRODUCTION!!!,new:UPI Receivables User TEST
old:UPI General Ledger User PRODUCTION!!!,new:UPI General Ledger User TEST
old:UPI Payables User PRODUCTION!!!,new:UPI Payables User TEST
old:UPI Help Desk PRODUCTION!!!,new:UPI Help Desk TEST
```

Note that these programs are optional – we like it to be *very* clear to our users which database instance they are using when they log into the applications, so we've changed the names of

Oracle's seeded responsibility names and our own enhancement responsibilities to include the instance name.

4. Under oraupi's home directory, we keep copies of DEVL_db_start.sh, DEVL_db_stop.sh, DEVL_rra_tm_start.sh, DEVL_rra_tm_stop.sh, ORAUPI_db_start.sh, ORAUPI_db_stop.sh, ORAUPI_rra_tm_start.sh, ORAUPI_rra_tm_stop.sh, TEST_db_start.sh, TEST_db_stop.sh, TEST_rra_tm_start.sh, DEVL_rra_tm_stop.sh.

These programs start and stop the database, and start and stop the Report Review Agent & Transaction Manager for each database.

5. Under oraupi's \$ORACLE_HOME, we have ORAUPI.env, TEST.env and DEVL.env.

These are another set of environment files necessary to access the database

6. Although root's home directory will not be copied as part of the refresh process, we keep copies of :
oa_DEVL_start.sh, oa_DEVL_stop.sh,
oa_ORAUPI_start.sh, oa_ORAUPI_stop.sh,
oa_TEST_start.sh and oa_TEST_stop.sh
on ORAUPI so that they are available if needed.

These programs start the database, the web, and the concurrent manager for each of the databases.

7. Under oraupi's \$ORACLE_HOME/dbs, we also maintain init.ora files for each of the databases: initORAUPI.ora, initDEVL.ora, and initTEST.ora

These programs are the initialization programs for each database instance, and have been configured to lower certain parameters like shared_pool_size and db_block_buffers for the TEST and DEVL instances, since our test database server has less memory and cpu available.

8. For sqlnet, we also maintain copies of the following files:
DEVL.lrc, DEVL.lsr, DEVL.trc, DEVL.tsr, ORAUPI.lrc, ORAUPI.lsr, ORAUPI.trc,
ORAUPI.tsr, TEST.lrc, TEST.lsr, TEST.trc, and TEST.tsr. We also have listener.DEVL.ora, listener.TEST.ora, tnsnames.DEVL.ora and tnsnames.TEST.ora.

Files We Maintain on upi_tdb (the Production ORAUPI Applications Server):

1. We are testing the idea of creating the web listener and forms server for DEVL on production but leaving it disabled. One issue we ran into when refreshing DEVL was that we had to go through the tedious steps of re-creating the DEVL web listener after every refresh of TEST or DEVL, so if the listener carries over from production all we should need to do is start it up once the refresh is complete. With any refresh from machine to machine, some changes must be made to the web setup because the two machine names are different. So if we can keep a trusty spare DEVL web listener and forms server set up and ready to run, then we may also add a TEST web listener setup to ORAUPI that has all of its parameters set to the correct machine.

2. Under applupi's home directory, we have DEVL_formlis_start.sh, DEVL_formlis_stop.sh, TEST_formlis_start.sh, TEST_formlis_stop.sh, ORAUPI_formlis_start.sh and ORAUPI_formlis_stop.sh.

These programs start and stop the forms servers for each database.

3. Under ORAUPI's \$APPL_TOP, we have DEVL.env, ORAUPI.env and TEST.env.

These are the environmental files for the three database instances we support.

4. Under oraupi's home directory, we have DEVL_httplis_start.sh, DEVL_httplis_stop.sh, TEST_httplis_start.sh, DEVL_httplis_stop.sh, ORAUPI_httplis_start.sh and ORAUPI_httplis_stop.sh.

These programs start and stop the web listener for each database.

5. Under oraupi's \$ORACLE_HOME, we have ORAUPI.env, TEST.env and DEVL.env.

These are environment files necessary to access the database

6. Under root's home directory, we have oa_DEVL_start.sh, oa_DEVL_stop.sh, oa_TEST_start.sh, oa_TEST_stop.sh, oa_ORAUPI_start.sh and oa_ORAUPI_stop.sh.

These programs start and stop the web listener for each database, and start and stop the forms servers for each database.

7. For sqlnet, we also maintain copies of the following files:
DEV.L.trc, DEVL.tsr, ORAUPI.trc, ORAUPI.tsr, TEST.trc, and TEST.tsr. We also have
tnsnames.DEVL.ora and tnsnames.TEST.ora.

Clearly what we're after here is the fastest way to do a refresh with the shortest amount of time reconfiguring scripts on the destination machine, so that we can minimize human error.

Make Sure You Have Enough Space on TEST:

You must ensure that there is enough space to hold all the files that will be carried over from ORAUPI. Our plan is to copy: \$APPL_TOP, all data files, and all redo logs. For our environment, we can use the du -k Unix command to find out how much space is currently being used for the data and code:

du -k /ora_apps/d01/app/applupi	4889288
du -k /ora_apps/d01/oradata/ORAUPI	2223432
du -k /ora_apps/d02/oradata/ORAUPI	28930008
du -k /ora_apps/d03/oradata/ORAUPI	11828168
du -k /ora_apps/d04/oradata/ORAUPI	5146200
du -k /usr/oracle/dbs/oradata/ORAUPI	4908321
du -k /usr/oracle/dbs2/oradata/ORAUPI	7499739

All told, this means that ORAUPI is using about 65.4 gigabytes of disk space, so we'll need that much on upi_tdb. We don't expect much change on disk requirements from refresh to refresh

for the application servers, but you should also make sure there is enough room on upi_tapp to clone upi_app.

WARNING: Before running any script mentioned in this paper it would be prudent to check the script's contents and be sure that you understand what the scripts are doing. We do not claim to be terrific script writers, but the scripts we use do what we need them to do. These scripts should be used only as a guideline for your own work.

Note that we've complicated our refresh process somewhat by renaming the database from ORAUPI to TEST. If you left everything as ORAUPI on your test environment, then there would be very few changes that needed to be made to get things running. We're not comfortable with maintaining two instances that are named the same because we don't want developers (or ourselves) to get confused.

Things You Can Do That Will Mess Up the Database Refresh:

- Take a cold backup of the ORAUPI database, but bring the ORAUPI database up again before the backup is complete. Everything will look fine until you try to bring up your test database instance. You'll have the right number of files, but some of them will be dated from when you took down the database and some will be dated from when you started the database back up. The new database will not restart and you'll get an error message telling you that the files are inconsistent. You'll need to redo your refresh.
- Use an old control file that doesn't reflect the names of all the data files that are currently in use. In this case the database will come up, but when you try to use some part of the application, the missing file will cause you to see error messages. Before bringing up your new database, you should always compare the number of database files that you have copied to the number of data files on the originating system. If you bring up the database with missing files, you'll have to shut it down, delete all the files and start over again. Make sure you make a timely backup of your control file to avoid having this problem occur twice.
- Copy data files from upi_db to upi_tdb while the database is still up. The database is in an inconsistent state and you'll have to start again, this time being sure to shutdown the source database, ORAUPI.

EXAMPLE 1: Refreshing TEST from ORAUPI by shutting down ORAUPI and copying files over the network:

1. Edit your crontab on upi_db, upi_tdb, upi_app and upi_tapp and comment out backups.
2. If you've done any enhancements, you'll likely have a subdirectory where developers store their work. For us, it's \$UPI_TOP. Before beginning the refresh, we always copy \$UPI_TOP's contents on the test servers (both the applications server upi_tdb and the database server upi_tapp) to a separate disk on their respective machines. This ensures that when we override \$UPI_TOP with a copy from ORAUPI, we do not lose in-process work that developers were doing on TEST.
3. Put scheduled concurrent requests on hold on ORAUPI. Deactivate the concurrent manager on ORAUPI.
4. Create an ORAUPI controlfile
5. Rename the controlfile.
6. ftp (transfer) the controlfile from the ORAUPI database server to the TEST database server

7. Edit the controlfile on the TEST database server.
8. Shutdown the Forms and HTTP listeners for ORAUPI and TEST on the applications servers.
9. Shutdown the ORAUPI and TEST database server's Transaction Manager, Concurrent Manager, Report Review Agent and Database.
10. Remove the TEST database servers' datafiles, redo logs, controlfiles, and \$APPL_TOP. Remove the TEST application server's \$APPL_TOP.
11. If they don't already exist, create nfs links from upi_db to upi_tdb and from upi_app to upi_tapp.
12. If they don't already exist, create symbolic links on upi_tdb like those you have on upi_db and on upi_tapp like you have on upi_app
13. Copy the ORAUPI database server's datafiles, redo logs, \$APPL_TOP and \$ORACLE_HOME to the TEST database server.
14. Copy the ORAUPI application server's \$APPL_TOP and \$ORACLE_HOME to the TEST application server.
15. Bring up the TEST database using the controlfile that you created in Step 6. Make sure it has the correct number of files and is not sending errors to \$ORACLE_BASE/admin/TEST/bdump/alert_TEST.ora.
16. Restart ORAUPI's database, webserver, sqlnet listeners and concurrent manager. Make sure you can log in to the applications.
17. Take concurrent requests that are On Hold on ORAUPI Off Hold.
18. Change the TEST webserver configuration so it points to TEST.
19. Bring up the TEST webserver.
20. Start the sqlnet listeners for TEST.
21. Log into the TEST application as a user with the System Administrator responsibility. Make changes to the configuration of the TEST concurrent manager.
22. Run resp_TEST.sql to change the database name for the apps responsibilities (optional).
23. Put scheduled records on hold. Don't start up the concurrent manager until you've put scheduled requests on hold, or programs may start running and printing that you didn't expect to.
24. Start up the concurrent manager on TEST.
25. Edit your crontab on upi_db, upi_tdb, upi_app and upi_tapp and turn backups back on.
26. Turn archiving on on TEST.

EXAMPLE 2: Refreshing TEST from ORAUPI using a backup tape:

1. Edit your crontab on upi_app and upi_tapp and comment out backups.
2. If you've done any enhancements, you'll likely have a subdirectory where developers store their work. For us, it's \$UPI_TOP. Before beginning the refresh, we always copy \$UPI_TOP's contents on the test servers (both the applications server upi_tdb and the database server upi_tapp) to a separate disk (we use a disk called /depot/ora_apps) on their respective machines. This ensures that when we override \$UPI_TOP with a copy from ORAUPI, we do not lose in-process work that developers were doing on TEST.
3. Backup the ORAUPI database and application servers, including code and data files, to tape. Your backup routine should include a step that backs up the ORAUPI controlfile to trace.
4. Shutdown the Forms and HTTP listeners for TEST on the applications server upi_tapp.
5. Shutdown the TEST database server's Transaction Manager, Concurrent Manager, Report Review Agent and Database on upi_tdb.
6. Remove the TEST database server's datafiles, redo logs, controlfiles, and \$APPL_TOP. If you leave the file structures in place, you shouldn't have to recreate any symbolic links on the TEST database server when restoring.

7. Restore from tape the ORAUPI database server's datafiles, redo logs, and \$APPL_TOP to the TEST database server upi_tdb.
8. Locate ORAUPI's backup controlfile on the TEST database server (it should be under \$ORACLE_BASE/admin/ORAUPI/udump) and modify it. Change the application and database server configurations to be TEST instead of ORAUPI.
9. Bring up the TEST database using the controlfile from Step 6. Make sure it has the correct number of files and is not sending errors to \$ORACLE_BASE/admin/TEST/bdump/alert_TEST.ora.
10. Change the TEST webserver configuration so it points to TEST.
11. Bring up the TEST webserver.
12. Start the sqlnet listeners for TEST.
13. Log into the TEST application as a user with the System Administrator responsibility. Make changes to the configuration of the TEST concurrent manager.
14. Run resp_TEST.sql to change the database name for the apps responsibilities (optional).
15. Put scheduled requests on hold. Don't start up the concurrent manager until you've put scheduled requests on hold, or programs will start running and printing that you didn't want to.
16. Start up the concurrent manager on TEST.
17. Edit your crontab on upi_db, upi_tdb, upi_app and upi_tapp and turn backups back on.
18. Turn archiving on on TEST.

Detailed Steps (NWR is for a network refresh, TR is for tape refreshes):

1. (NWR) Deactivate the Concurrent Manager on ORAUPI's database server as soon as users give you permission to start shutting down (for us, this needs to be prior to 10:30 p.m. due to some large request sets that run after that). This will reduce the possibility of a long-running concurrent request kicking off when you're ready to start working, preventing you from getting started. As Unix user applupi, cd /home/applupi. Run ORAUPI_concmgr_deactivate.

This program should do the following:

```
# Stop the Oracle ORAUPI Instance Concurrent Managers - Step 3 of complete shutdown
oraupi
# ORAUPI_concmgr_deactivate.sh
# Script to deactivate concurrent manager.
# This script can also be run as root.
echo ""
echo "Deactivating ORAUPI Concurrent Managers"
echo ""
CONCSUB apps/password SYSADMIN 'System Administrator PRODUCTION!!!'
SYSADMIN CONCURRENT FND DEACTIVATE
```

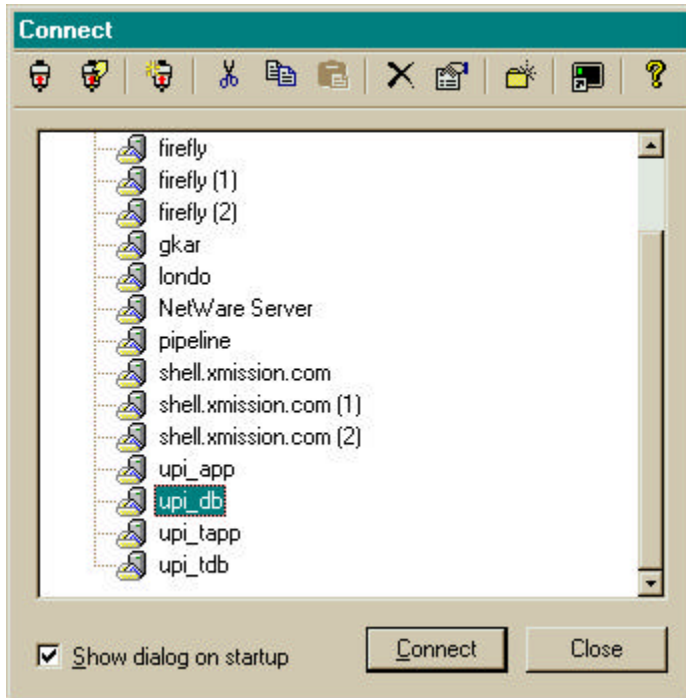
Make sure all the processes are gone by typing:

ps -ef | grep LIB

at a command line prompt. If a concurrent request is still running when you want to get started, log into the applications and terminate the request, or wait for it to finish. Use the **kill -9 pid** command to kill any concurrent manager processes that refuse to go away.

2. (NWR) Create a copy of the ORAUPI database's controlfile from ORAUPI's database server:

- Double click on the CRT icon (or whatever tool you use for telnet access). Highlight `upi_db`. Click on the Connect box at the bottom:



- Log into `upi_db` as Unix user `oraupi`.
- Run `svrmgrl`
`svrmgrl` allows you to issue certain database commands required by Oracle, including starting and stopping the database and creating backup controlfiles.
- Use “alter database backup controlfile to trace” to get a controlfile that can be edited.
The *alter database backup controlfile* command is a special one provided by Oracle that takes the controlfile, which is in an executable but not readable format, and creates a file that you can read and edit.

```

ORAUPI -- /home/oraupi $ svrmgrl
Oracle Server Manager Release 3.0.5.0.0 - Production
(c) Copyright 1997, Oracle Corporation. All Rights Reserved.
Oracle8 Enterprise Edition Release 8.0.5.1.0 - Production
PL/SQL Release 8.0.5.1.0 - Production
SVRMGR> connect internal
Connected.
SVRMGR> alter database backup controlfile to trace;
Statement processed.
SVRMGR> exit

```

- ftp the trace file to the `$ORACLE_HOME/dbs` directory on `upi_tdb`. Name it `control.ctl`:
 - a. On the ORAUI database server `upi_db`, `cd` to the `udump` directory, which is where a ‘backup controlfile to trace’ command automatically saves its output:
 - b. On ORAUI, use the `ls -lt | more` command to find the most recent trace file that exists. That file *should* hold the results of the backup controlfile to trace command that you issued. Occasionally if there is other activity on the database, there might be other trace files created around the same time that you created your controlfile. You

- should look at the different files, starting at the top of the list, for the one that looks like a controlfile (see Step 4 for an example).
- c. vi that file to see if it is indeed the right file.

```

ORAUPI -- /upid01/app/oraupi/product/805/dbs $ cd $ORACLE_BASE/admin/ORAUPI/udump
ORAUPI -- /upid01/app/oraupi/admin/ORAUPI/udump $ ls -lt | more
total 32256
-rw-r--r-- 1 oraupi dba          6453      Dec 10    22:40 ora_16943.trc
-rw-r--r-- 1 oraupi dba          6452      Dec 10    21:55 ora_15107.trc
-rw-r--r-- 1 oraupi dba          6375      Dec 10    04:45 ora_17895.trc
-rw-r--r-- 1 oraupi dba          6376      Dec 9      20:27 ora_26877.trc
-rw-r--r-- 1 oraupi dba          6376      Dec 9      04:35 ora_26377.trc
-rw-r--r-- 1 oraupi dba          6373      Dec 8      04:33 ora_6328.trc
-rw-r--r-- 1 oraupi dba          6375      Dec 7      22:02 ora_18270.trc
-rw-r--r-- 1 oraupi dba          6375      Dec 7      05:15 ora_14113.trc
-rw-r--r-- 1 oraupi dba        49069      Dec 3      11:22 oewb_sp.out
-rw-r--r-- 1 oraupi dba        57909      Dec 3      11:19 oewb_cb.out
-rw-r--r-- 1 oraupi dba       167534      Dec 3      11:14 oewb_createby.trc
-rw-r--r-- 1 oraupi dba        22029      Dec 3      11:14 oewb_salesp.trc
-rw-r--r-- 1 oraupi dba          6372      Dec 3      02:05 ora_9044.trc
-rw-r--r-- 1 oraupi dba       209199      Dec 2      14:36 ora_7383.trc
-rw-r--r-- 1 oraupi dba          3734      Dec 2      14:25 ora_11654.trc
-rw-r--r-- 1 oraupi dba        67908      Dec 2      09:21 history1.prf
-rw-r--r-- 1 oraupi dba          6294      Dec 2      04:45 ora_29389.trc
-rw-r--r-- 1 oraupi dba       313783      Dec 2      01:38 barb.trc
-rw-r--r-- 1 oraupi dba       221337      Dec 2      01:31 ora_27395.trc
ORAUPI -- /upid01/app/oraupi/admin/ORAUPI/udump $ vi ora_16943.trc

```

The file should look something like the following (look for the STARTUP NOMOUNT at the bottom):

```

"ora_16943.trc" 168 lines, 6453 characters
Dump file /upid01/app/oraupi/admin/ORAUPI/udump/ora_16943.trc
Oracle8 Enterprise Edition Release 8.0.5.1.0 - Production
PL/SQL Release 8.0.5.1.0 - Production
ORACLE_HOME = /upid01/app/oraupi/product/805
System name:  HP-UX
Node name:    upi_db
Release:      B.11.00
Version:      A
Machine:      9000/800
Instance name: ORAUPI
Redo thread mounted by this instance: 1
Oracle process number: 63
Unix process pid: 16943, image: oracleORAUPI

*** SESSION ID:(235.1039) 1999.12.10.22.40.52.612
*** 1999.12.10.22.40.52.611
# The following commands will create a new control file and use it to open the database.
# Data used by the recovery manager will be lost. Additional logs may be required for

```

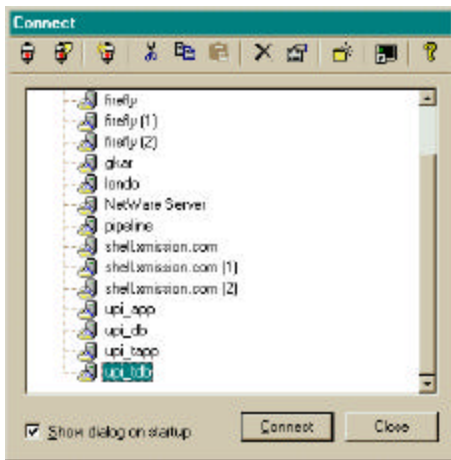
```
# media recovery of offline data files. Use this only if the current version of all online
# logs are available.
STARTUP NOMOUNT
CREATE CONTROLFILE REUSE DATABASE "ORAUPI" NORESETLOGS
ARCHIVELOG
MAXLOGFILES 16
MAXLOGMEMBERS 2
```

- d. Type **:q!** to exit the file once you've made sure you have the right one. Write down the name of the file. In this example, the filename is ora_16943.trc, but yours will be different.
- e. Rename the file to control.ctl:
Type **mv ora_16943.trc control.ctl** (using the filename that you wrote down in place of ora_16943.trc). You've just renamed the file to be called control.ctl.
- f. Type **ftp upi_tdb**. When it asks for a password, enter **oraupi**, then press return and then enter the Unix oraupi password. Basically, you could see this as making a telephone call – you are on one machine, the production ORAUPI database machine, and you are connecting to another machine, the TEST database machine. You'll be transferring a file (ftp means file transfer protocol) from one machine to another.
- g. Type **cd /upid01/app/oraupi/product/805/dbs**
- h. Type **put control.ctl** The put command puts the file that you have on ORAUPI onto UPI_TDB in the \$ORACLE_HOME/dbs directory.
- i. Type **quit** to log out of ftp

```
ORAUPI -- /upid01/app/oraupi/admin/ORAUPI/udump $ ftp upi_tdb
Connected to upi_tdb.
220 upi_dev FTP server (Version 1.1.214.2 Mon May 11 12:21:14 GMT 1998) ready.
Name (upi_tdb:oraupi): oraupi
331 Password required for oracle.
Password:
230 User oraupi logged in.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> cd /upid01/app/oraupi/product/805/dbs
ftp> put control.ctl
200 PORT command successful.
150 Opening BINARY mode data connection for control.ctl.
226 Transfer complete.
6453 bytes sent in 0.01 seconds (1079.07 Kbytes/s)
ftp> quit
221 Goodbye.
ORAUPI -- /upid01/app/oraupi/admin/ORAUPI/udump $ exit
```

- j. Log off of the production database server upi_db by typing **exit** and closing the window.
- k. Log onto the test database server upi_tdb:

Double click on the CRT icon. Highlight upi_tdb. Click on the Connect box at the bottom:



- l. Log into upi_tdb as Unix user oraupi.
- m. Change directories to the \$ORACLE_HOME/dbs directory:
cd \$ORACLE_HOME/dbs
- n. Edit control.ctl on the TEST database server upi_tdb.
vi control.ctl

Some basic **vi** commands for editing files:

The **j** key moves you down

The **k** key moves you up

The **l** key moves you to the right

The **h** key moves you to the left

x deletes a letter

dd deletes a line

5dd deletes 5 lines

i puts you in insert mode

]] takes you to the bottom of a document

[[takes you to the top of a document

:1 takes you to the first line, or top, of a document

:10 takes you to the 10th line of a document

The **Esc** key takes you out of insert mode

To search for something, press the **/** key and type what you are looking for, then press return

To save your work, press the **esc** key, then the **:**, then **wq** and press return

To exit without saving your work, press the **esc** key, then the **:**, then **q!**

- o. Remove the following first lines:

```
Dump file /upid01/app/oraupi/admin/ORAUPI/udump/ora_26877.trc
Oracle8 Enterprise Edition Release 8.0.5.1.0 - Production
PL/SQL Release 8.0.5.1.0 - Production
ORACLE_HOME = /upid01/app/oraupi/product/805
System name:      HP-UX
Node name:        upi_db
Release:          B.11.00
Version:          A
Machine:          9000/800
Instance name:    ORAUPI
Redo thread mounted by this instance: 1
Oracle process number: 63
Unix process pid: 26877, image: oracleORAUPI
```

```

*** SESSION ID:(257.1704) 1999.12.09.20.27.30.033
*** 1999.12.09.20.27.30.033
# The following commands will create a new control file and use it
# to open the database.
# Data used by the recovery manager will be lost. Additional logs may

```

p. Change the “Create Controlfile” statement to the following:

```

CREATE CONTROLFILE SET DATABASE "TEST" RESETLOGS
NOARCHIVELOG

```

NOTE: It is VERY important that you doublecheck to make sure that your controlfile looks exactly as specified.

q. Remove the following lines from the end of the file:

```

# Recovery is required if any of the datafiles are restored backups,
# or if the last shutdown was not normal or immediate.
RECOVER DATABASE
# All logs need archiving and a log switch is needed.
ALTER SYSTEM ARCHIVE LOG ALL;
# Database can now be opened normally.
ALTER DATABASE OPEN;

```

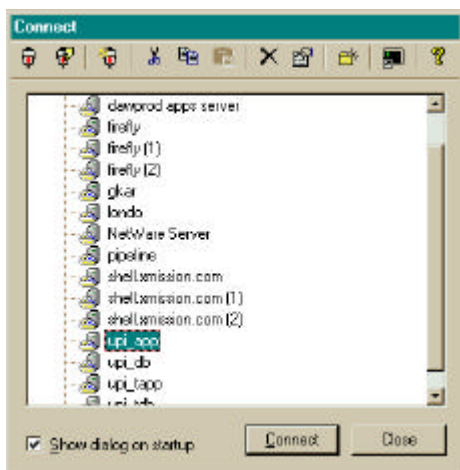
r. Use the **j** key or **]]** to move down to the lines at the bottom.

Type **dd** to remove a line.

Save the newly edited control.ctl

Type **:wq** to save the changes that you’ve made.

3. {NWR) Shutdown the Forms and HTTP Listeners for the ORAUPI and TEST instances.
Log on to upi_app as root



In the root directory on the production application server upi_app run oa_ORAUPI_stop.sh

This program should look like:

```

# Stop the Oracle ORAUPI Instance Forms & HTTP Listeners - Steps 1 and 2 of complete
# shutdown (before doing shutdown on DB/Concurrent Mgr Server UPI_DEV)

```

```
su - applupi ORAUPI_formlis_stop.sh
su - oraupi ORAUPI_httplis_stop.sh
su - oraupi COM_httplis_stop.sh
```

Exit from the production applications server and close the window: **exit**
(NWR, TR) In the root directory on the TEST application server run oa_TEST_stop.sh
Exit from the applications server UPI_TAPP and close the window: **exit**

4. (NWR,TR) Bring down the TEST Transaction Manager, Concurrent Manager, Report Review Agent and Database:
Log on to upi_tdb as root
In the root directory run oa_TEST_stop.sh
Make sure that no “LIB” processes or ORAUPI instance background processes remain.
ps -ef | grep LIB
5. (NWR) Bring down the ORAUPI Transaction Manager, Concurrent Manager, Report Review Agent and Database.
Log on to upi_db as root
In the root directory run oa_ORAUPI_stop.sh, which should look like the following:
more oa_ORAUPI_stop.sh
Stop the Oracle ORAUPI Instance DBs, Report Review Agents & Trans Mgrs, and
Concurrent Managers - Steps 3, 4, and 5 of complete shutdown (after doing shutdown
on Application Server UPI_APP)
su - applupi ORAUPI_concmgr_stop.sh
su - oraupi ORAUPI_rra_tm_stop.sh
su - oraupi ORAUPI_db_stop.sh
Make sure that no “LIBR” processes or UPI instance background processes remain.
ps -ef | grep LIB
6. (NWR,TR) Remove the TEST datafiles, redo logs and controlfiles and \$APPL_TOP:
Log on to UPI_tdb as Unix user root.
cd /ora_apps
remove all data files, \$ORACLE_HOME and \$APPL_TOP
rm -R /ora_apps/*
This leaves the file structure in place (including symbolic links), but gets rid of all the files.

Note that if you’ve deleted everything and removed the underlying subdirectories, you’ll want to recreate the following directories as user root, but with owner applupi and group dba on both the applications and database servers:

```
mkdir /tempdata/ORAUPI/log
mkdir /tempdata/ORAUPI/out
mkdir /tempdata/ORAUPI/tmp
```

These directories hold the log and output files for the concurrent manager, and the tmp directory allows you to edit files. If you try to edit a file and get an error that says /tempdata/tmp doesn’t exist, then you need to create these directories. If you try to run a concurrent request after you’ve brought everything up and the request fails, check to see if the request output or log files exist under the log and out directories. If they don’t, then you need to change permissions on the directories. Note that you generally don’t want to bring

the contents of the log and out directories over from production, because it will take so long to copy and you shouldn't need the files anyway. If you needed to bring those files over, you would modify cpREFRESH1.sh in step 8 to copy them.

7. (NWR) If you are copying from disk, shutdown the ORAUPI database before copying files (VERY important). As Unix user oraupi on the production database server:

```
svrmgrl
connect internal
shutdown immediate;
exit;
```

8. (NWR) Copy the ORAUPI datafiles, redo logs \$ORACLE_HOME and \$APPL_TOP to UPI_tdb over an NFS mount. This will take a while (for us, about 4 hours).

Log on to ORAUPI_db as root

cd to /ora_apps and run the cpREFRESH1.sh script.

You must have an NFS mount defined on upi_db to move files to upi_tdb. This script should run the following:

```
# This gets the datafiles, redo logs, $ORACLE_HOME and $APPL_TOP (note that this
assumes there is an nfs mount called upi_tdb between oraupi_db and upi_tdb):
```

```
nohup cp -rp /ora_apps/d01/* /upi_tdb/ora_apps/d01 &
nohup cp -rp /ora_apps/d02/* /upi_tdb/ora_apps/d02 &
nohup cp -rp /ora_apps/d03/* /upi_tdb/ora_apps/d03 &
nohup cp -rp /ora_apps/d04/* /upi_tdb/ora_apps/d04 &
```

```
#This gets the ORAUPI data files and leaves behind the JIT data files:
```

```
nohup cp -rp /usr/oracle/dbs/ORAUPI/* /upi_tdb/ora_apps/usr/oracle/dbs/ORAUPI &
nohup cp -rp /usr/oracle/dbs2/ORAUPI/* /upi_tdb/ora_apps/usr/oracle/dbs2/ORAUPI &
```

```
# This gets the $UPI_TOP code and patch directories
```

```
nohup cp -rp /ora_apps/UPI_TOP/* /upi_tdb/ora_apps/UPI_TOP &
nohup cp -rp /ora_apps/patch/* /upi_tdb/ora_apps/patch &
nohup cp -rp /ora_apps/rllapps/* /upi_tdb/ora_apps/rllapps &
nohup cp -rp /ora_apps/sqlcom/* /upi_tdb/ora_apps/sqlcom &
```

```
# This is taxware
```

```
nohup cp -rp /ora_apps/tools/* /upi_tdb/ora_apps/tools &
```

9. (TR) On upi_tdb, restore the ORAUPI datafiles, redo logs, \$ORACLE_HOME and \$APPL_TOP from tape.

10. (NWR) On upi_tdb, rename or remove the controlfiles that were just copied from production:

Find the files using the ls command or look for their locations in initORAUPI.ora:

```
ls /ora_apps/*/oradata/ORAUPI/cnt*
```

Delete each of the control files:

```
rm /ora_apps/d01/oradata/ORAUPI/cntrl01.ctl
rm /ora_apps/d02/oradata/ORAUPI/cntrl02.ctl
```


rm /ora_apps/d03/oradata/ORAUPI/ctrl03.ctl

11. (NWR,TR) Count the number of datafiles on ORAUPI, and then count the number of files that you placed on TEST. You should have the same number of datafiles!

On ORAUPI,:

select count(*) from sys.dba_data_files;

On TEST as Unix user root:

ls -lt /ora_apps/d*/oradata/ORAUPI/*

ls -lt /usr/oracle/dbs*/oradata/ORAUPI/*

12. Test that your database will likely come up by making a copy of your controlfile to a Unix .sh file, and changing all the lines that list files to have ls -lt in front, get rid of the double quotes around the file name, get rid of the commas, and get rid of anything that isn't naming a file. Then run your file. If it comes back with any file not found errors, then you'll likely need to change the path of the data file in your controlfile so it reflects what the file is called on TEST.

13. (NWR,TR) Build the new controlfile and open the database.

Log on to UPI_tdb as oraupi.

Run svrmgrl and connect internal:

svrmgrl

connect internal

Perform a startup nomount:

Startup nomount pfile=\$ORACLE_HOME/dbs/initTEST.ora

Build the new control file by running the edited file control.ctl (NWR) or by running the edited backup controlfile that was restored from tape (TR):

@/tmp/control.ctl

Open the database and reset the logs:

Alter database open resetlogs;

exit

Note that it takes about 10 minutes to resetlogs on our system.

13. (NWR, TR) Check alert_test.log to ensure that the database opened and isn't reporting any missing files:

tail -f \$ORACLE_BASE/admin/ORAUPI/bdump/alert_TEST.log

14. (NWR,TR) Make modifications on the applications server, UPI_Tapp, to change the OAS server name (this is the web stuff):

- Change upi_app to upi_tapp in:

/ora_apps/d01/app/oraupi/product/805/ows/admin/ows/website30/wrb/config/wrb.app

/ora_apps/d01/app/oraupi/product/805/ows/admin/ows/website30/wrb/config/omnaddr

/ora_apps/d01/app/oraupi/product/805/ows/admin/ows/website30/httpd_upi_app/owl.cfg

/ora_apps/d01/app/oraupi/product/805/ows/admin/ows/website30/httpd_upi_app/admin/config/svadmin.cfg

/ora_apps/d01/app/oraupi/product/805/ows/admin/ows/website30/httpd_upi_app/lprd/config/svlprd.cfg

/ora_apps/d01/app/oraupi/product/805/ows/admin/ows/website30/httpd_upi_app/www/config/

```
svwww.cfg
/ora_apps/d01/app/oraupi/product/805/ows/3.0/admin/servapp.dfl
/home/oraupi/.profile (this shouldn't need to be changed, since we didn't copy /home/oraupi)
/home/applupi/.profile (this shouldn't need to be changed, since we didn't copy /home/applupi)
/ora_apps/d01/app/applupi/1102/TEST.env (this shouldn't need to be changed, as I placed a copy
on ORAUPI)
```

- The directory /ora_apps/d01/app/oraupi/product/805/ows/admin/ows/website30/httpd_upi_app should be moved to httpd_upi_tapp. This is necessary as substitutions are used when finding files located here:
mv /ora_apps/d01/app/oraupi/product/805/ows/admin/ows/website30/httpd_upi_app
/ora_apps/d01/app/oraupi/product/805/ows/admin/ows/website30/httpd_upi_tapp

- Recreate the symbolic link for:
/ora_apps/d01/app/oraupi/product/805/ows/3.0/admin/svadmin.cfg
To do this:
as root rm svadmin.cfg
ln -s
/upid01/app/oraupi/product/805/ows/admin/ows/website30/httpd_upi_tapp/admin/config/
svadmin.cfg svadmin.cfg

15. (NWR,TR) As Unix user oraupi on upi_tapp AND upi_tdb,
cp /upid01/app/oraupi/product/805/network/admin/listener.TEST.ora
/upid01/app/oraupi/product/805/network/admin/listener.ora

16. (NWR,TR)As Unix user oraupi on upi_tapp AND upi_tdb:

```
cp /upid01/app/oraupi/product/805/network/admin/tnsnames.TEST.ora
/upid01/app/oraupi/product/805/network/admin/tnsnames.ora
```

17. Start the web server for both the ORAUPI and TEST database servers:
(NWR,TR) Log on to upi_tapp as root. From the root directory run the oa_TEST_start.sh script, which should look like:

```
# Start the Oracle TEST Instance Forms & HTTP Listeners - Steps 4 and 5 of complete
# startup (after doing startup on DB/Concurrent Mgr Server UPI_DEV)
su - oraupi TEST_httplis_start.sh
su - applupi TEST_formlis_start.sh
```

(NWR)Log on to upi_app as root
From the root directory run the oa_ORAUPI_start.sh script

18. (NWR,TR) Start the sqlnet listeners on UPI_tdb. Log onto UPI_tdb as Unix user oraupi.
Run TEST_rra_tm_start.sh, which should look like:
Start the Oracle ORAUPI Instance Report Review Agent & Transaction Manager - Step 2
of complete startup
TEST
cd \$ORACLE_BASE/admin/\$ORACLE_SID/scripts
aplsnctl.sh start \$ORACLE_SID
aplsnctl.sh start APPS_\$ORACLE_SID
exit

19. (NWR,TR) On UPI_tdb, log in as applupi.

Run \$UPI_TOP/resp_TEST.sql as oracle user apps to change the responsibility names within the applications so they say Test instead of Production!!! This program should look like:

-- Replace PRODUCTION!!! with TEST in responsibility names

-- 14 Dec 99 William Faulk

SET SERVEROUTPUT ON SIZE 40000

DECLARE

v_new_name fnd_responsibility_tl.responsibility_name%TYPE;

v_pos NUMBER;

CURSOR c_resp IS

SELECT application_id,

responsibility_id,

responsibility_name

FROM fnd_responsibility_tl

WHERE UPPER(responsibility_name) like '%PRODUCTION!!!%';

BEGIN

FOR v_resp IN c_resp LOOP

v_pos := instr(v_resp.responsibility_name,'PRODUCTION!!!');

IF v_pos > 0 THEN

v_new_name := SUBSTR(v_resp.responsibility_name,1,v_pos - 1)||

'TEST'||SUBSTR(v_resp.responsibility_name,v_pos + 13,100);

dbms_output.put_line('old:'||v_resp.responsibility_name||',new:'||

v_new_name);

UPDATE fnd_responsibility_tl

SET responsibility_name = v_new_name

WHERE responsibility_id = v_resp.responsibility_id;

END IF;

END LOOP;

END;

/

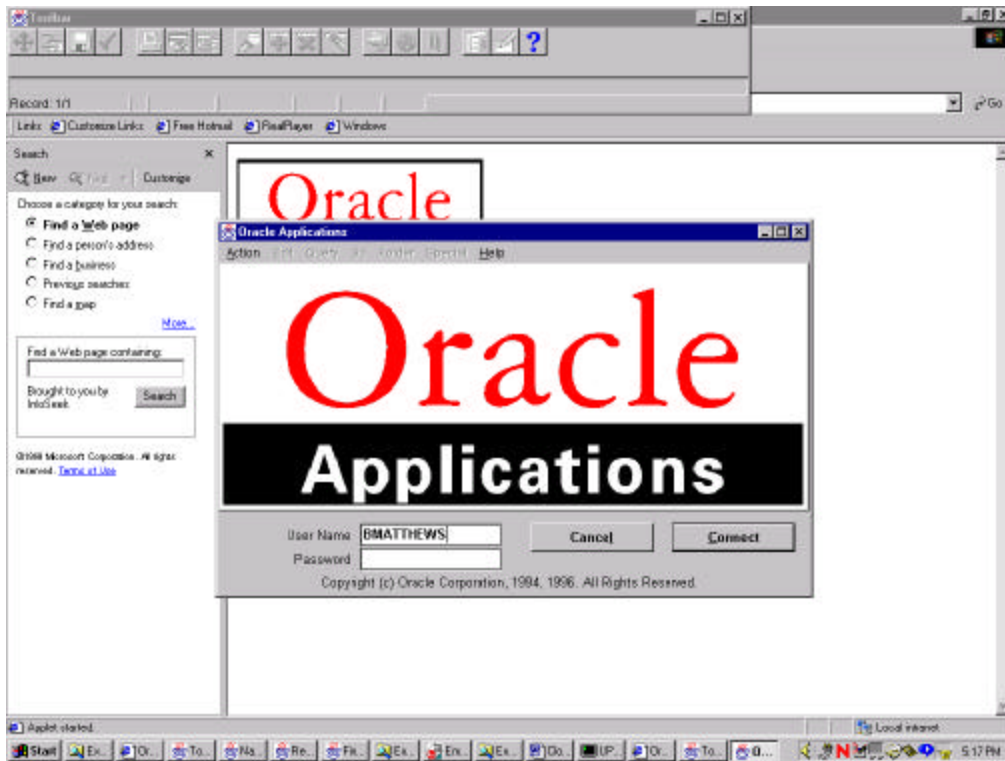
20. (NWR,TR) Log onto the Financials Application that connects to upi_tdb:

a. Double-click on the **Internet Explorer** icon on your desktop.

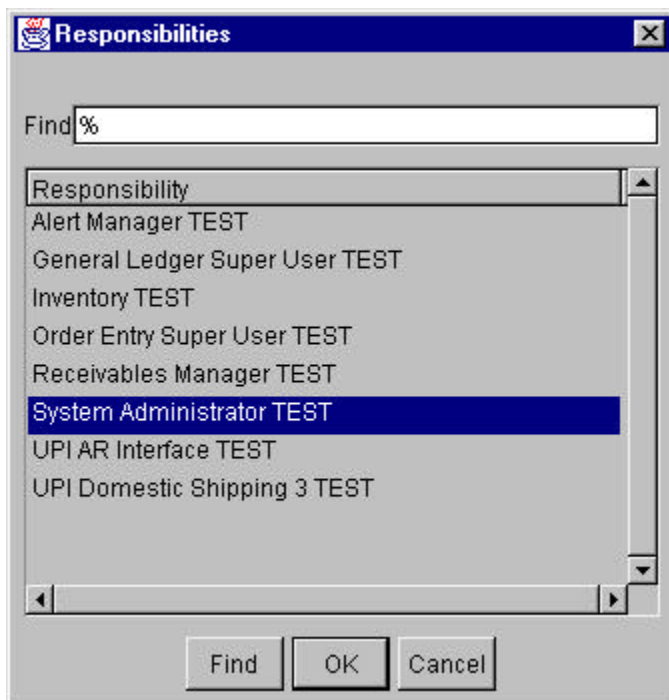
b. Left click on the **Favorites** box at the top of the screen

c. Land your icon on **Ultradent test**

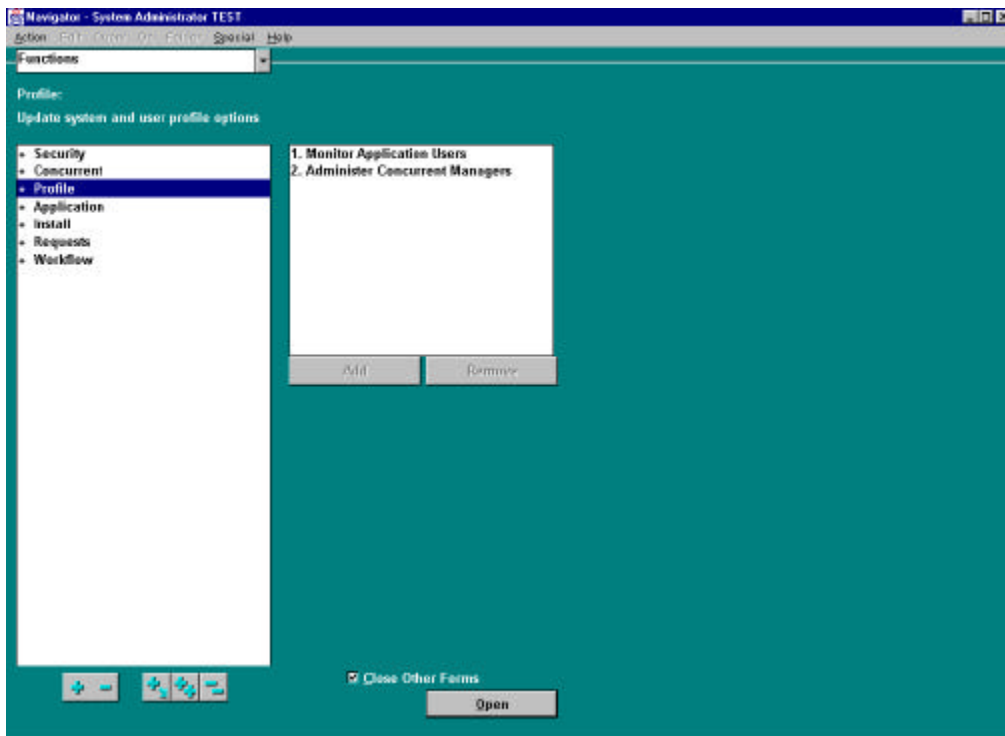
For username, use an account that has the System Administrator responsibility:



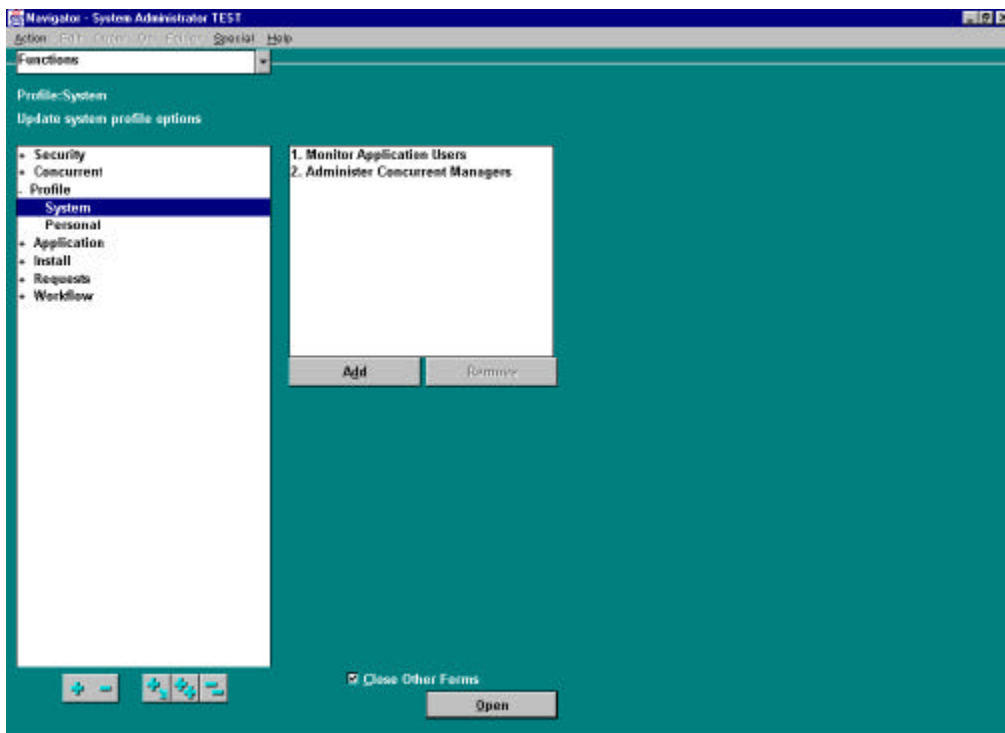
- d. Select the System Administrator responsibility then click on OK. You'll see System Administrator TEST instead of System Administrator PRODUCTION!!! if resp_TEST.sql has been run correctly.



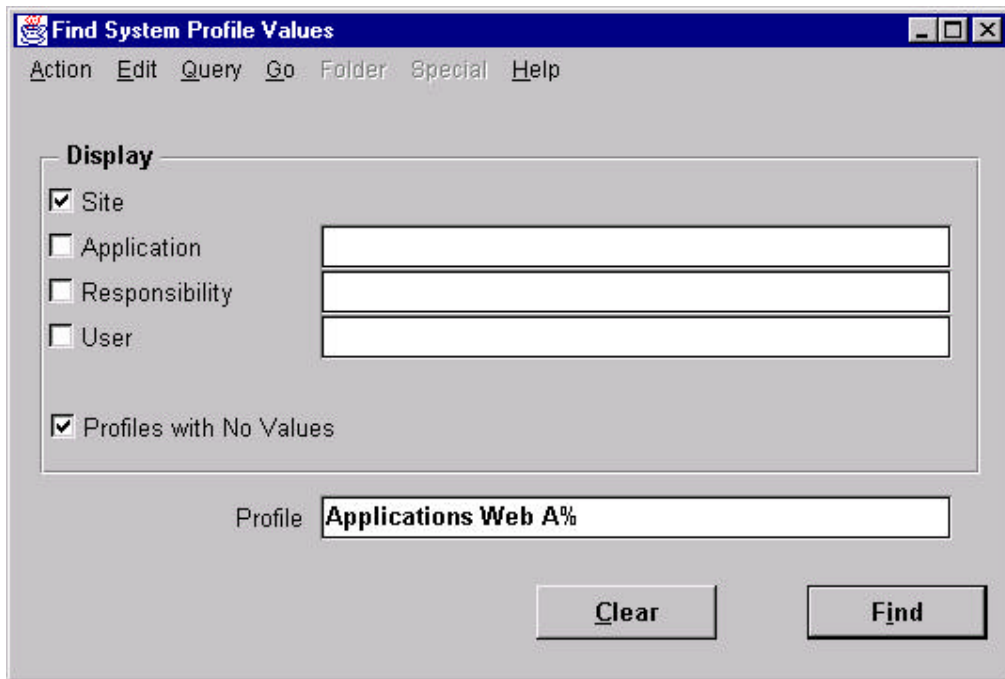
- e. Click on **Profile**



f. Double-click on **System**:



g. Query up each of four profile options and change them so they match the values below. Enter the profile option you are searching for in the Profile field at the bottom of the screen. Then press the Find button at the bottom of the screen.



Find System Profile Values

Action Edit Query Go Folder Special Help

Display

☒ Site

☐ Application

☐ Responsibility

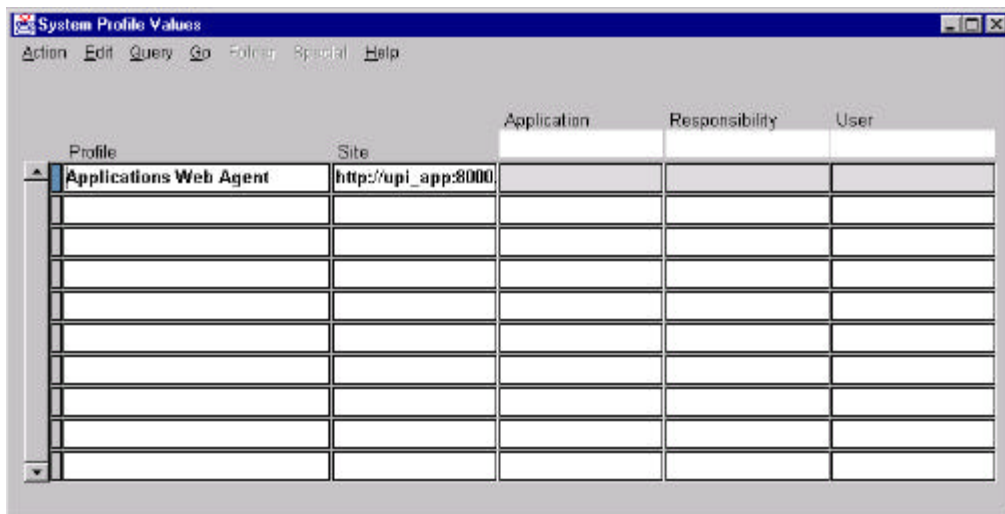
☐ User

☒ Profiles with No Values

Profile

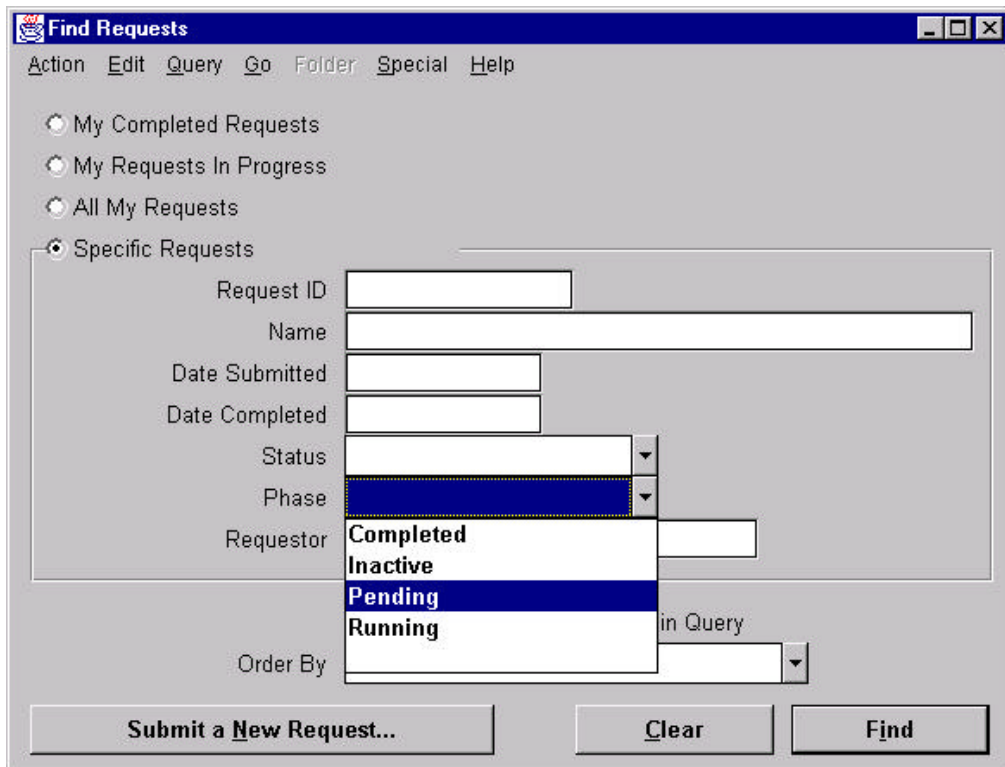
Clear Find

- h. Make changes to the following application profile values in UPI forms
- Applications Web Agent `http://upi_tapp:7990/TEST/plsql`
 - Help System Base Url `http://upi_tapp:7990/OA_DOC/`
 - OE: Transaction Manager `OEORPC_TEST`
 - Site name `Ultradent TEST`
- i. You change the profile option values by changing the values in the Site column:



Profile	Site	Application	Responsibility	User
Applications Web Agent	http://upi_app:8000			

- j. Once you've changed the profile values, go to Concurrent/Requests and query up all requests that are pending. Click on the Specific Requests circle on the left. Choose Pending from the pull down list for Phase. Then press Find:



Find Requests

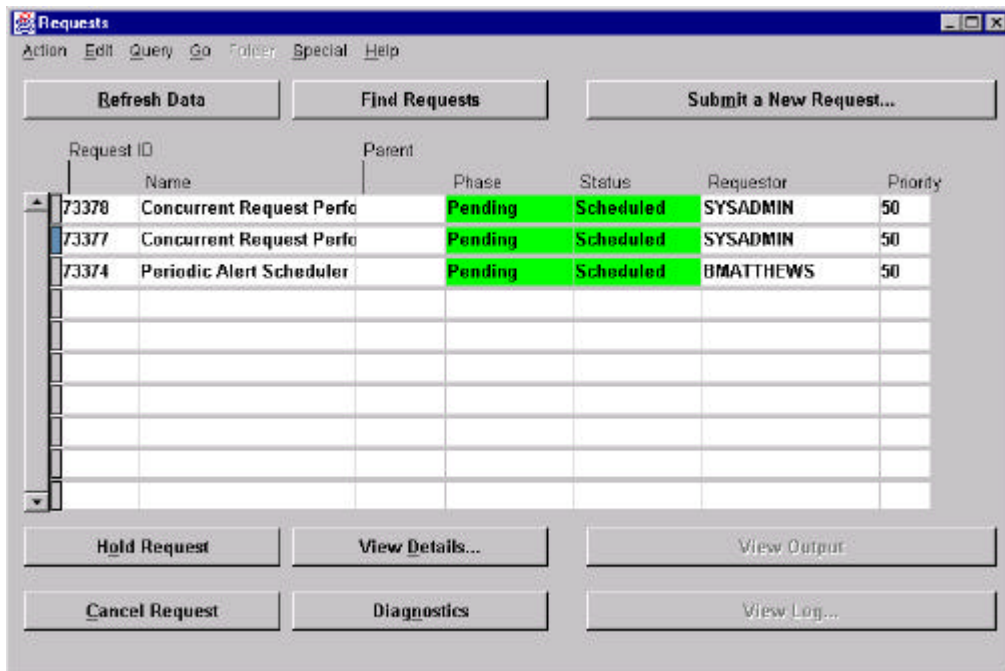
Action Edit Query Go Folder Special Help

☐ My Completed Requests
☐ My Requests In Progress
☐ All My Requests
☒ Specific Requests

Request ID
 Name
 Date Submitted
 Date Completed
 Status
 Phase
 Requestor

Order By

- k. If you see any requests click on the Request ID of the request, and then click on Hold Request button. This will put the request on hold.

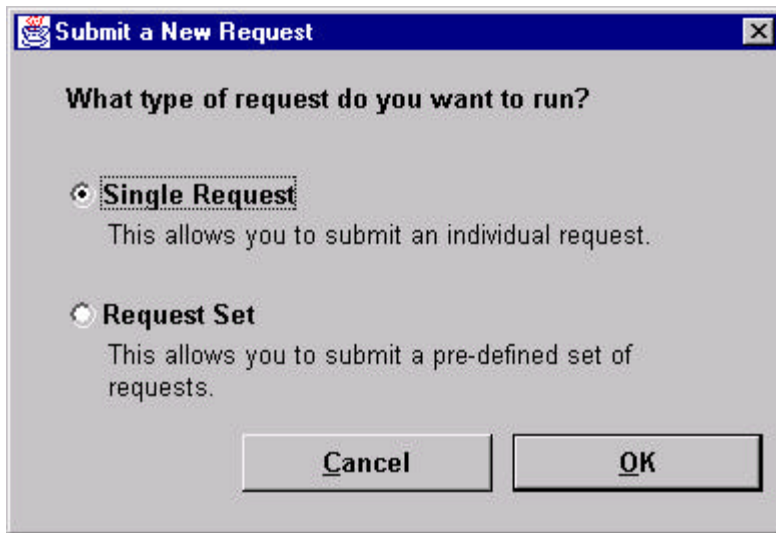


Requests

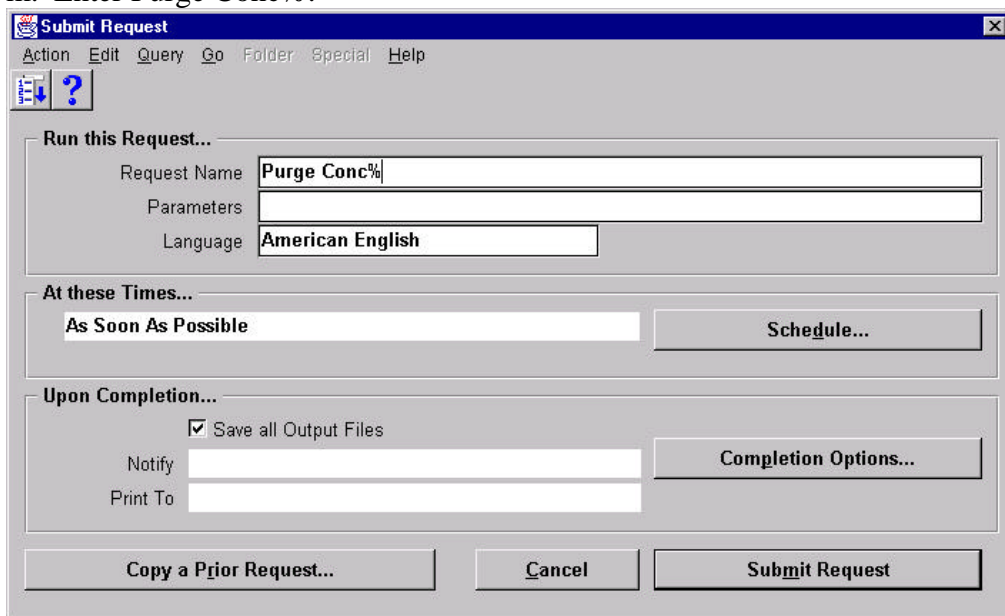
Action Edit Query Go Folder Special Help

Request ID	Name	Parent	Phase	Status	Requestor	Priority
73378	Concurrent Request Perfo		Pending	Scheduled	SYSADMIN	50
73377	Concurrent Request Perfo		Pending	Scheduled	SYSADMIN	50
73374	Periodic Alert Scheduler		Pending	Scheduled	BMATTHEWS	50

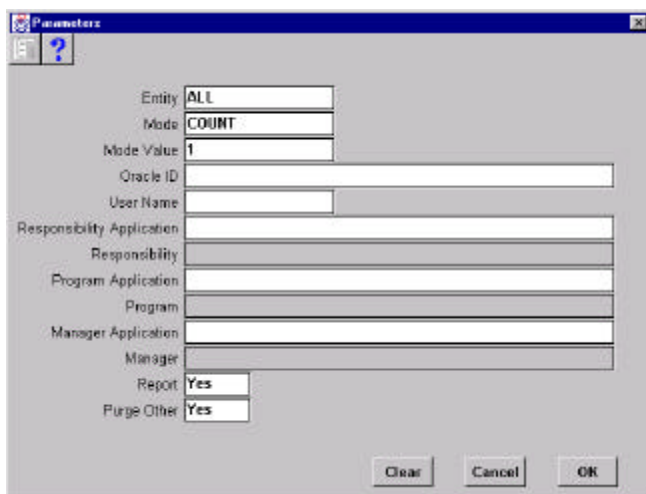
- l. Now click on the Submit a New Request... button at the top right of the screen. Click OK for a Single Request:



m. Enter Purge Conc%:



n. A screen will pop up. Type ALL in the Entity field, Count in the Mode field, and 1 in the Mode Value field. This tells the program to delete all records other than this one:



- o. Click on OK.
- p. Log onto the upi_tdb server as Unix user applupi. Run TEST_concmgr_start.sh. This will start the concurrent manager.
- q. Go back to the application screens and select **Concurrent/Manager/Administer** and make sure that the concurrent manager has restarted. If the concurrent manager is running, it looks like the following. In particular, the Actual column should have numbers in it.

The screenshot shows a window titled 'Administer Concurrent Managers' with a menu bar (Action, Edit, Query, Go, Folder, Refresh, Help). Below the menu bar is a table with columns: Name, Node, Actual, Target, Running, Pending, and Status. The table lists several managers, with the 'Internal Manager' on the 'upi_tdb' node showing 1 Actual and 1 Target. Other managers like 'Conflict Resolution Manager', 'Receivables Tax Manager', 'CRP Inquiry Manager', 'Fast Requests Manager (UP)', 'Inventory Manager', 'INV Remote Procedure Ma', 'MRP Manager', 'Shipping Transaction Mana', 'PA Streamline Manager', 'PO Document Approval Ma', and 'Receiving Transaction Man' also show Actual and Target values. At the bottom of the window are buttons for 'Terminate', 'Deactivate', 'Verify', 'Processes', and 'Requests'.

Name	Node	Processes		Requests		Status
		Actual	Target	Running	Pending	
Internal Manager	upi_tdb	1	1		0	
Conflict Resolution Manager		1	1		0	
Receivables Tax Manager		1	1			
CRP Inquiry Manager		2	2			
Fast Requests Manager (UP)		2	2	0	0	
Inventory Manager		3	3	0	0	
INV Remote Procedure Ma		3	3			
MRP Manager		1	1	0	0	
Shipping Transaction Mana		8	8			
PA Streamline Manager		1	1	0	0	
PO Document Approval Ma		3	3			
Receiving Transaction Man		3	3			

- r. Now as a final check, go to **Concurrent/Requests** and query up where Status is Error (use the pulldown screen). You shouldn't find any requests with a status of error, since this database is starting fresh. You've deleted all the requests that ran on Production prior to today, and you put any pending requests on hold. The only request that should have run since you started things back up is your request to Purge Concurrent Requests. In all likelihood, that is still running. The test database should be set. If you find any requests that have errored out, look at the top one and click on View Details. Look at the Date Completed – it should be a date prior to the refresh. If the date is prior to the refresh, then the Purge Concurrent Requests just hasn't gotten to it yet. You only need to worry if a request has errored since you restarted the database.
- s. (NWR) Now check Production:
 - Log onto the production upi_db as Unix user oraupi.
 - Type **ORAUPI_db_start.sh**
 - Type **ORAUPI_rra_tm_start.sh**
 - Log onto the production application by choosing **Ultradent ORAUPI** from the Favorites list after double-clicking on the **Internet Explorer** icon.
 - Log on as Unix user applupi and run ORAUPI_concmgr_start.sh
 - Check that the concurrent manager is up.

- Check that nothing has errored since we made the copy (you'll likely find lots of errored requests, but you only need to look at the one at the top of the list and check the time when it failed to know if you have a problem).
- Make sure you can use SQLPLUS to log into the production database. Click on the **Start** button at the bottom of your PC screen, then choose **Programs**, and follow the path to **SQLPLUS**, and when you get this screen, log in:

The screenshot shows a 'Log On' dialog box from SQL*Plus. It has a title bar 'Log On' and three input fields: 'User Name:' containing 'system', 'Password:' containing '*****', and 'Host String:' containing 'ORAUPI'. Below the fields are 'OK' and 'Cancel' buttons.

- If you get in, then production is OK.
- (NWR,TR) Now try the same thing for TEST, except use TEST for the Host String instead of ORAUI. If you get in, then the TEST instance is OK.
 - Ultradent uses a set of tools that provide status information about the databases. You can download these tools from www.oncalldb.com. If you are using the tools, make the morning reports work on TEST by doing the following:
 - `cd $UPI_TOP/sql/trutek_tools`
 - `cp toolkit.ini.TEST toolkit.ini`
 - We use archiving on our TEST environment so we can run nightly hot backups. To turn archiving back on for TEST:
 - `$ORACLE_HOME/dbs/initTEST.ora` should point to the archive directories that you've set up:


```
log_archive_dest    = /upid01/app/oraupi/admin/ORAUI/arch/arch
log_archive_duplex_dest = /upid01/app/oraupi/admin/ORAUI/arch1/arch
log_archive_format  = %t_%s.log
log_archive_min_succeed_dest = 2
log_archive_start = true    # if you want automatic archiving
```
 - Now, you'd think you'd be done and could just start up the database and archiving would be working, but that's not the case.
 - In `svrmgrl`:


```
startup mount pfile=$ORACLE_HOME/dbs/initTEST.ora
alter database archivelog;
alter database open;
archive log all;
```

archive log list

You should see the following:

Database log mode:	Archive Mode
Automatic archival	Enabled

If you don't see this, then archiving isn't working.

- alter system switch logfile;
- Now go to your archive directories and make sure you have archive log files in them. If you don't, then you didn't do this right.
- A special note on starting the database: If you've gotten into the habit of logging into svrmgrl and starting the database by typing:
startup
without telling it which pfile to use, you'll have endless problems once we have the second database instance going.
You should enter:
startup pfile=\$ORACLE_HOME/dbs/initTEST.ora
or
startup pfile=\$ORACLE_HOME/dbs/initDEVL.ora
to be sure you are pointing to the right database.

Cloning the DEVL Instance

The steps for setting up a second instance are similar to those for the first instance. For our second instance, DEVL, we created a second mount point on upi_tdb and upi_tapp and restored all of the \$APPL_TOP code and data files there from a backup of our 11.0.3+ work. The second mount point is called ora_dev, and looks just like TEST's /ora_app directory structure except that you'd follow the path /ora_dev/ora_apps.... We use the same RDBMS code for both TEST and DEVL, and use scripts to point to certain files that point to SID-specific information. We found that the second web listener, which we called DEVL, had to be set up through the same web interface as the lprd listener that carries over from production. This caused us considerable confusion, because every time we refresh TEST we have to redefine the DEVL listener. Since the mount points are different for the two instances, we also had to modify all of our scripts to point to the /ora_dev mount point instead of the /ora_apps mount point.

Before we go further in this description, we want to make it very clear that creating and maintaining this second instance has given us the most trouble. We warn you, therefore, to truly consider this paper as a guideline for how to clone databases. In particular, we understand now that the biggest mistake we've made when it comes to creating a second instance is believing that we could do so without following Oracle's number one recommendation. According to Oracle, the only supported mechanism for creating additional databases is to get out your One Hour Install CD and install a starter database from there, and then once you're sure it is working, refresh the code and data from another database. What this supported mechanism does is structure your database and code the way it needs to be and the way Oracle Support expects it to be. If you do like we have, where you try to create a second instance from a backup, you'll likely spend extra hours trying to mesh what you've done with what the One Hour Install would have done. Nonetheless, we're describing our process because it actually helped us understand

better what was happening behind the scenes, particularly with the web configuration, and it also highlighted some issues that we think you really ought to be thinking about.

Since we started from scratch on the Web Listener configuration, this section will show how to set up the web listener configuration through the browser. If you look back at the details on our TEST web listener setup, we actually modified a number of programs, primarily to change the name of the database from ORAUPI to DEVL, and to change the name of the applications server from upi_app to upi_tapp. Making the changes through the browser seems more straightforward.

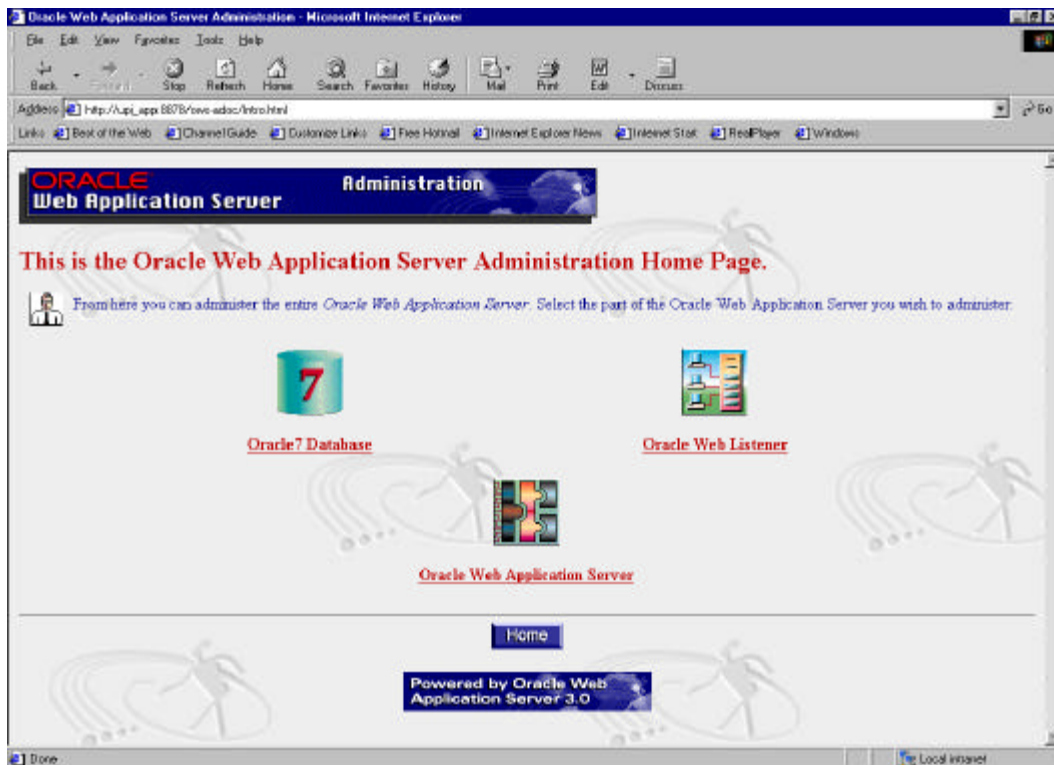
1. Point your browser to http://upi_tapp:7990 and log on as admin/password



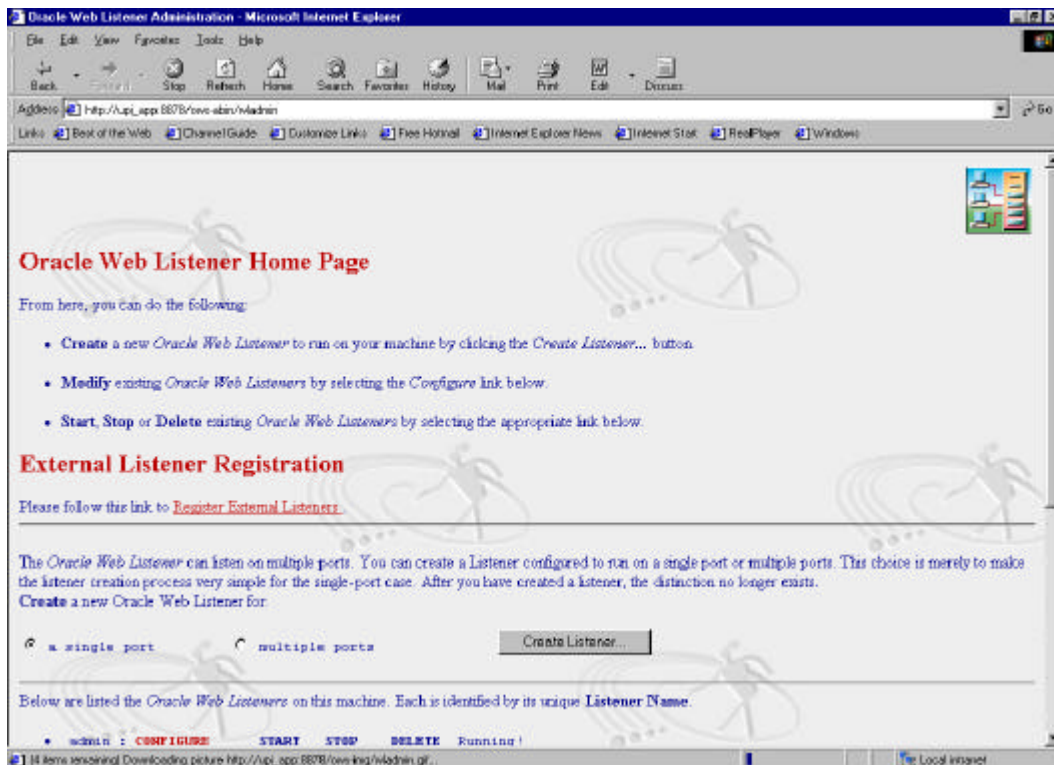
2. Choose “Web Application Server Manager”

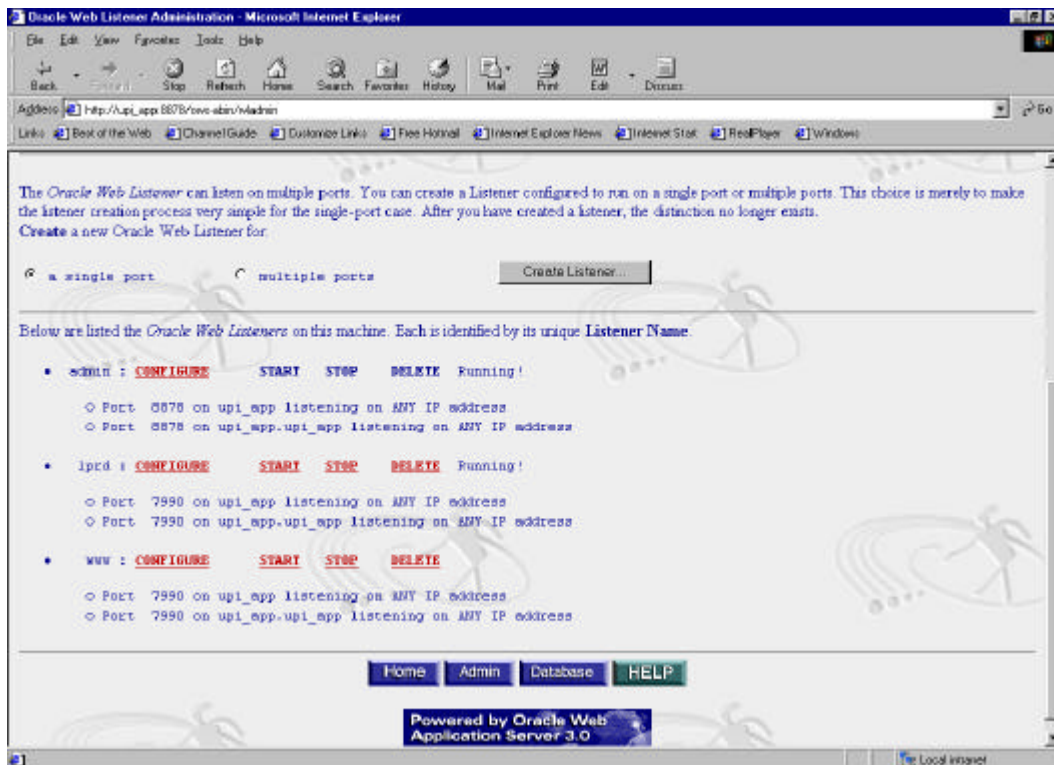


3. Choose “Oracle Web Listener”

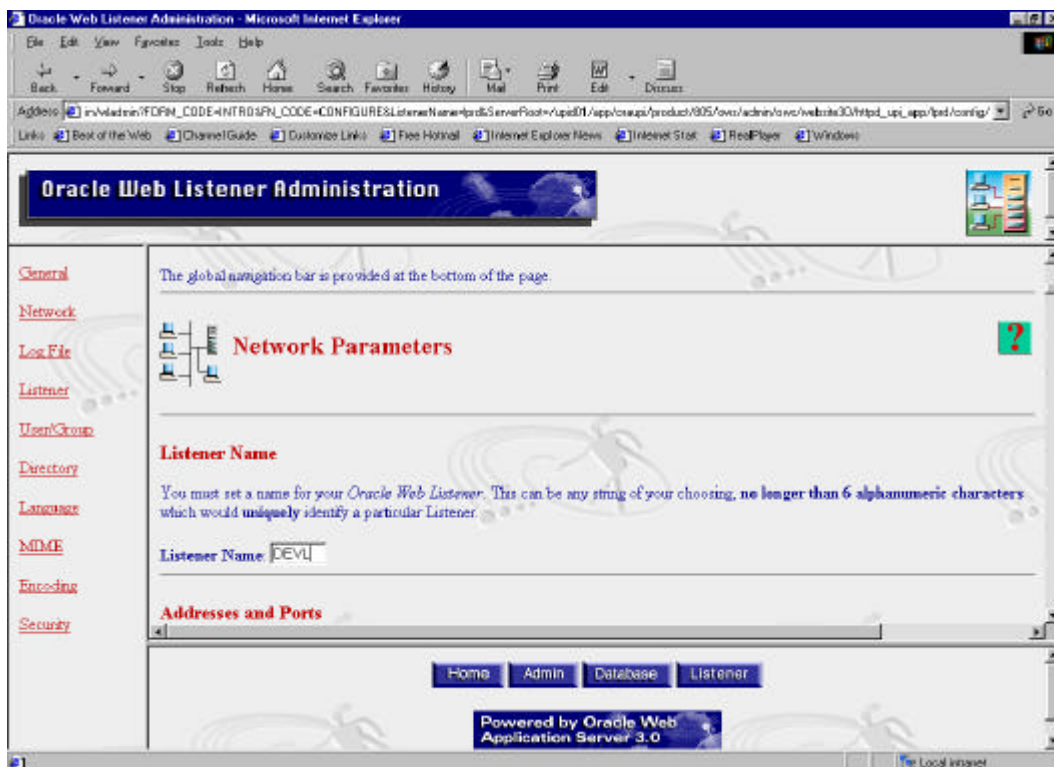


4. Scroll down and click on the Configure option for the Web Listener that serves the source environment.

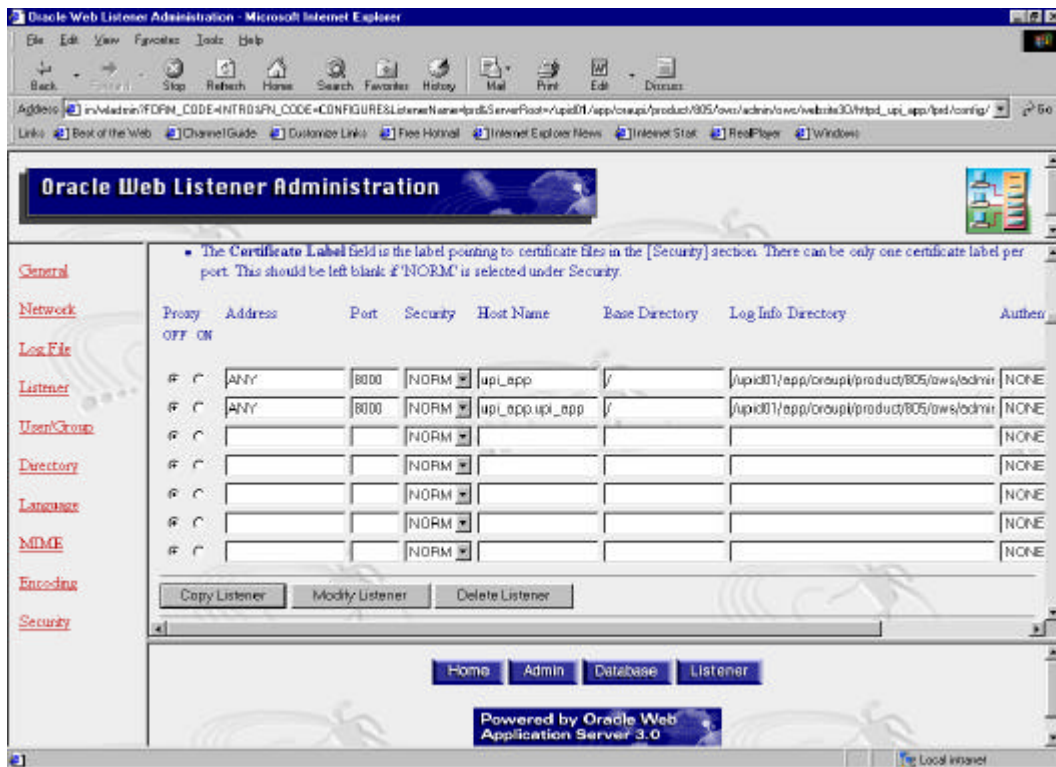




5. Change the listener name (ours is lprd) to the Oracle_SID of your new DB (DEVL)



6. In "Addresses and Ports" change the port number to a new unused port number. For DEVL we changed the port from 7990 to 8000:

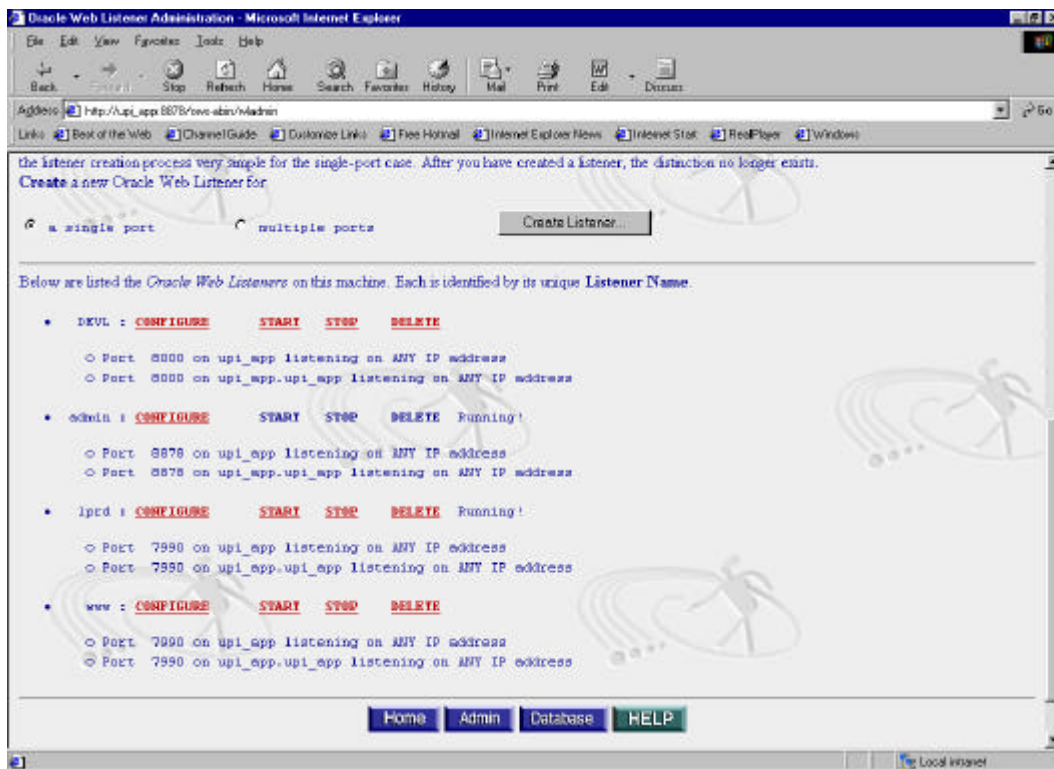


Note that at this point, the Host Name and Log Info Directory are still pointing to ORAUP's directory structure, rather than DEVL's mount point.

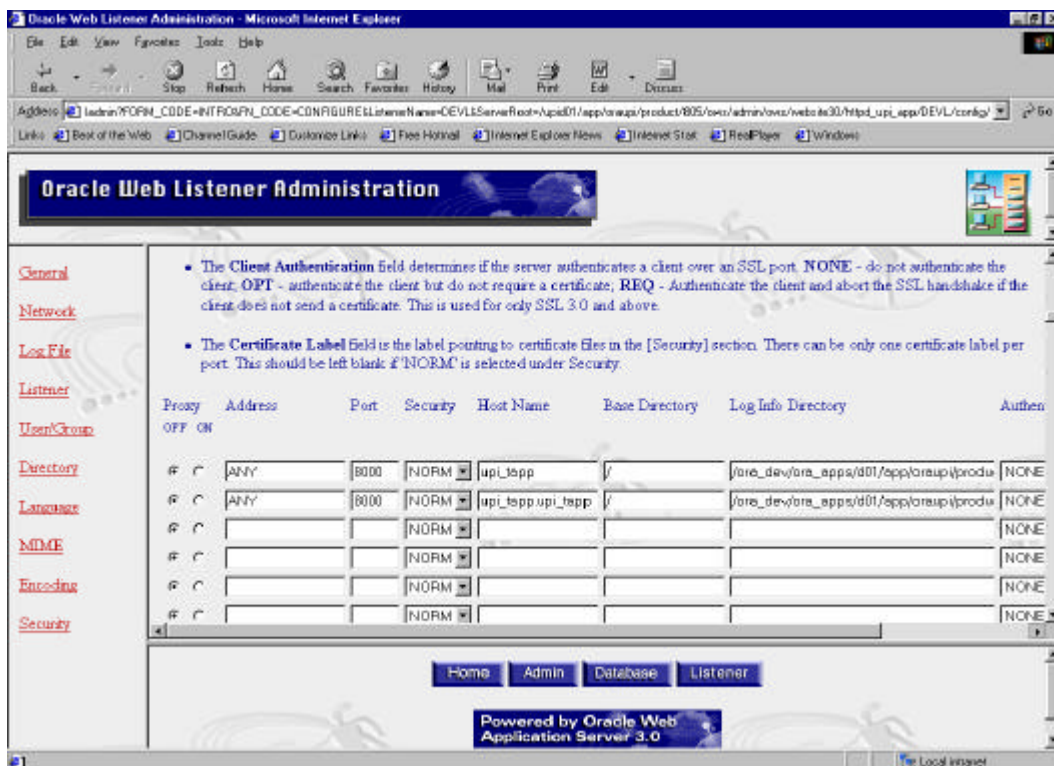
7. Click on Copy Listener



If you scroll down you'll see the second Web Listener, called DEVL. Note that the DEVL web listener isn't running yet:



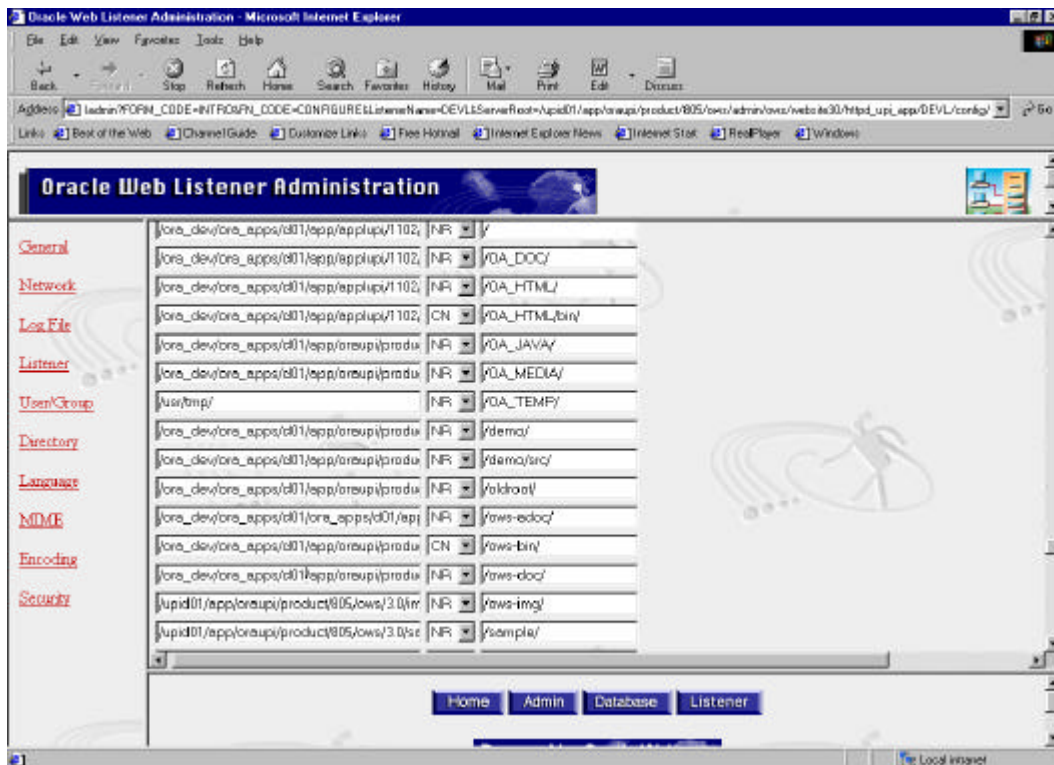
- Click on Configure for your new listener and modify the directory section to reflect the structure you have created. Note that we are changing the directory structure from /ora_apps, which points to TEST's \$APPL_TOP, to /ora_dev/ora_apps, which is DEVL's \$APPL_TOP. Also make sure the log files are going to the correct spot. Note that there are a number of places where you will need to make changes to references to directories, so be sure to scroll all the way down through this screen.



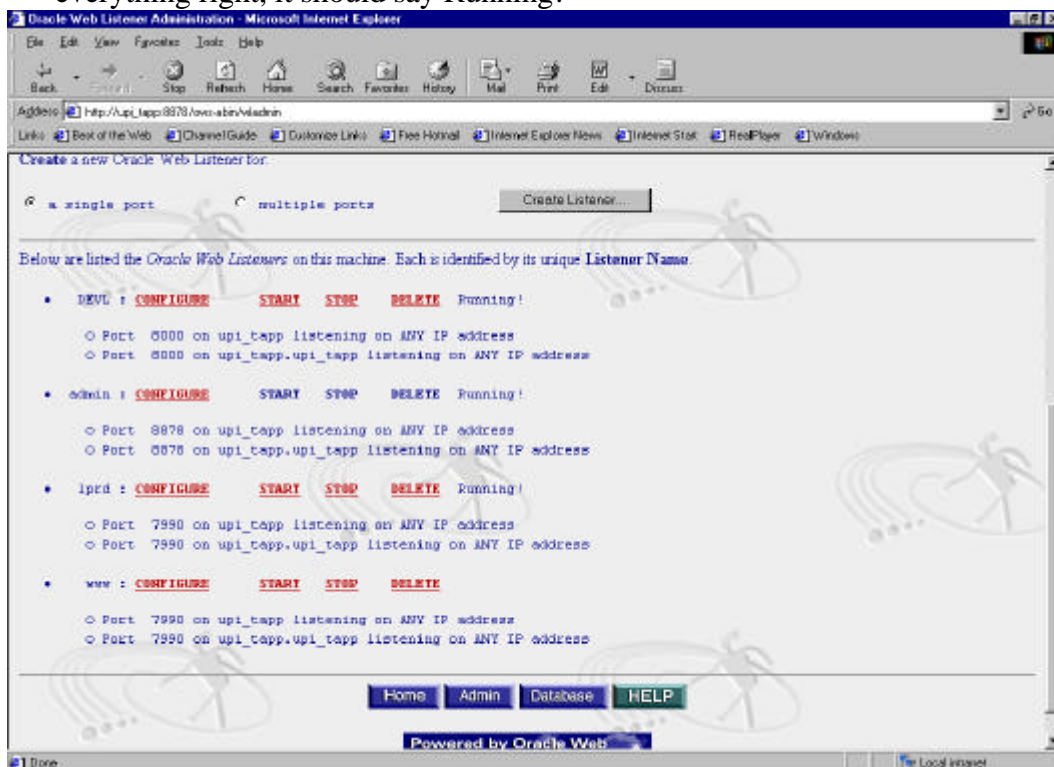


Note that in our first attempt to create a DEVL listener on ORAUPI so that it would copy over for refreshes, this is where we hit a failure point. This configuration file tries to create a file called /ora_dev/ora_apps/d01/app/oraupi/product/805/ows/admin/svDEVL.cfg. Unfortunately, there is no directory structure called /ora_dev/ora_apps on our production database server, upi_tdb. So we're still thinking about how to simplify this part of our refresh process.





9. Click Modify. Start the Web listener for DEVL by clicking on START. If you've done everything right, it should say Running!



We've shown you two ways to configure a web listener on a test system. One directly modifies scripts, and the other works through the browser to the web listener. Both are prone to typographical error, but we like working through the browser because you make all of the changes in one place, rather than across several scripts. We suspect also that modifying through the browser is the Oracle Support approved methodology. Note that until we find a

way to create a DEVL port that carries over from production, every time we refresh DEVL, we will lose DEVL's web listener port and have to create it again. Also, on TEST we use the already defined lprd port that carried over from production, and so still must modify the port to point to the correct machine name, upi_tapp, rather than upi_app.

Configuring a Second Forms Server for DEVL

Note that if you skip this next step, you might never realize that you are running your second instance against the correct database, but the wrong forms code. You must create a second admin listener for DEVL to ensure that you are running the correct SQL*FORMS code. We can assure you that it's pretty easy to not realize that you are doing this... when you apply patches, for example, you are logged directly onto your Unix processor, so the code will land in the right place. Only the forms code is affected by this listener, and only your jinitiator users will use the forms code.

We assume you are using a static html configuration for this next part:

1. Log on as Unix user applupi, being sure you are pointing to the correct \$APPL_TOP, and copy the file \$OA_HTML/US/TEST_j.htm or \$OA_HTML/US/ORAUPI_j.htm to \$OA_HTML/US/DEVL_j.htm. This is the file that we call when users log onto the applications through jinitiator. Yours is likely called something else, and was created as a copy of afsamples.htm with some parameter changes. Change the port number from 8990, which is the one we are using for TEST, to 8991, which is the one we use for DEVL. These lines appear twice in DEVL_j.htm. Change the database name to DEVL, and change the references to the \$APPL_TOP location.
7. Start the forms server. You must be sure that you are pointing to the correct \$APPL_TOP when you run this command:
DEVL -- /home/applupi \$ f45ctl start port=8991
port=8991
Choosing default pool 1 for this process...
1
Forms environment:
ORACLE_HOME = /upid01/app/oraupi/product/805
PATH =
/upid01/app/oraupi/product/805/bin:/upid01/app/oraupi/product/805/bin:/usr/ccs/bin:/usr/sbin:/upid01/app/oraupi/local/java/jre/bin:/usr/bin:/opt/ansic/bin:/usr/ccs/bin:/usr/contrib/bin:/opt/nettladm/bin:/opt/fc/bin:/opt/fcms/bin:/opt/upgrade/bin:/opt/pd/bin:/usr/bin/X11:/usr/contrib/bin/X11:/opt/perf/bin:/opt/java/bin:/opt/hparray/bin:/opt/resmon/bin:/opt/pred/bin:/opt/hpnp/bin:/opt/ignite/bin:/opt/langtools/bin:/opt/imake/bin:/opt/omni/bin:/usr/local/bin:/home/oracle/bin:/upid01/app/oraupi/product/805/ows/3.0/bin:/hrdev/cobol.solaris/cobol/bin:/hrdev/cobol.solaris/cobol:/ora_dev/ora_apps/d01/app/applupi/1102/fnd/11.0.28/bin:/ora_dev/ora_apps/d01/app/applupi/1102/ad/11.0.28/bin
LD_LIBRARY_PATH =
/upid01/app/oraupi/product/805/lib:/ora_dev/ora_apps/d01/app/oraupi/local/java/jre/lib:/upid01/app/oraupi/product/805/lib:/upid01/app/oraupi/product/805/lib:/usr/lib:/upid01/app/oraupi/product/805/ows/3.0/lib:/upid01/app/oraupi/product/805/ows/3.0/omx/lib:/upid01/app/oraupi/product/805/lib:/hrdev/cobol.solaris/cobol/coblib

FORMS45_PATH =

/upid01/app/oraupi/product/805/forms45/html:/ora_dev/ora_apps/d01/app/applupi/1102/au/11.0.28/resource

Forms listener started on port 8991.

8. You can double-check that your forms server is active by typing the command `ps -ef | grep 8991`:

```
applupi 2988    1  0 08:55:19 pts/tb    0:00 f45srvm port=8991 pool=1
```

You should now have two forms servers, one for TEST and one for DEVL.

To test that you are pointing to the correct forms server, shutdown the TEST forms server and log into the applications. If you can get in to DEVL without an error, you should be set.

Upgrading the DEVL Instance from 11.0.2 to 11.0.3 and Beyond

Remember that expression your Mom used to say? “Practice Makes Perfect”. The first time we did this upgrade, it took us a solid week. We did this upgrade on our test environment 3 times, for a variety of reasons and mishaps. We recommend that you run through the 11.0.2 to 11.0.3 upgrade at least twice on your test environment before running it on production. We were stumped time and again by patches behaving differently between attempts. To smooth the refresh process, we tracked how long it took to apply each patch, and whenever we ran into a problem, we corrected it on production, if possible, to eliminate that particular problem from occurring the next time we ran the upgrade. We also found cases where the patch hung for performance reasons. We’ve included in this document alternative statements you can use for those situations. These occurred, as one might expect, on the megapatches and one-off patches that we applied after upgrading to 11.0.3 rather than on the 11.0.3 upgrade, which has been, as far as we can tell, more comprehensively tested by Oracle’s customers. We’ve also documented our workarounds for other issues that we hit that we couldn’t “pre-fix” on production so that we could eliminate the need to call Oracle Support and work through the same problems a second time.

Assumptions and Rules of Thumb:

- This process assumes we will leave the TEST instance running at Version 11.0.2 and apply occasional one-off patches to it to deal with specific functionality issues on production. We will upgrade DEVL from Version 11.0.2 to 11.0.3.1+. Once testing is complete, we will either upgrade the 11.0.2 TEST instance and then upgrade PRODUCTION, or we’ll upgrade straight to PRODUCTION.
- Make sure the concurrent manager is down on the DEVL instance and that there are no applmgr processes. As Unix user applupi, run `DEVL_deactivate_concmgr.sh`, then `ps -ef | grep LIB`. Kill any applupi-owned LIB processes using the `kill -9 pid` command, or wait for them to go away on their own.
- Remember to cancel any backups of both the apps and database server while performing the upgrade.
- Oracle CD Mounting Instructions for HP-UX:
`nohup /usr/sbin/pfs_mountd`
`nohup /usr/sbin/pfsd`
`/usr/sbin/pfs_mount -t rrip -x unix /dev/dsk/c5t2d0 /SD_CDROM`

- As apps/apps in SQL:
update fnd_printer_styles set length=66 where printer_style_name='LANDWIDE';
Although we kept good notes, we cannot remember which of the 11.0.3 patches failed with this problem.
- We had to fix ownership on \$_TOP/lib/* and \$_TOP/bin/* to 775 using the command:
chmod 775 /upid01/app/applupi/1102/*/*/*/*/. You shouldn't have to do this. We had problems with patches failing on our first attempts because of ownership issues. Normally you wouldn't expect to see these kind of issues, since the One Hour Install ought to put things into place with correct permissions.
- In applying patches, we concluded that we could run two c drivers at the same time, or a c driver along with a d or g driver, but we couldn't run d or g drivers at the same time because sometimes the g drivers use the fnd_install_processes table, and the d drivers definitely use this table... only one process can access this table at a time. We concluded that the best order to apply things was: run the c drivers on the apps and database servers at the same time, then run the database server d driver, then the database server g driver, then the apps server g driver. You do not need to apply the d driver on the apps server.
- Apply all the drivers for the patch except the d drivers on the application server. Then move on to the next patch and do the same thing. Usually there is a c,d and g driver, but sometimes there isn't. You apply any drivers that exist under the patch.
- If you start adpatch, and it says it has already started, say No and then Yes to get it to throw away what it thought it was working on and start fresh on your upgrade. You might see this message if someone had partially applied a patch at some other time.
- Keep track of how long it takes to apply each part of the patch. This will help you later on when you are trying to estimate how long it will take to upgrade production.
- Log files for us are located at \$APPL_TOP/admin/DEVL/log
- We had patches fail trying to find files that ended with *pker.sql that should have been located on the database server, but were located on the applications server instead. We were mystified at how this could happen, since the original One Hour Install should have placed all files in the proper locations, and these were the only files that landed on the wrong server. We located all files on upi_tapp that end with pker.sql and copied them to the upi_tdb. This eliminated all the problems we had the first time we ran the upgrade where patches failed because Oracle couldn't find certain files. Our theory is that the One Hour Install incorrectly placed these files on the application server rather than on the database server where they belong. We've given up trying to understand how this could have happened but believe that this problem alone supports our belief that you should actually perform major upgrades not twice, but three times: twice on a test system, and then finally on the production system. This allows you to "pre-fix" problems to ensure that the production upgrade is as speedy as possible.
- If a patch stops and asks you to fix a failed worker, you have to go to \$APPL_TOP/admin/log, find the log file, look at it and figure out why the patch failed. Then you have to fix the problem. Then, to get the patch to keep going where it left off, you log onto another window and run adctrl. adctrl gives you a menu that allows you to tell the patch manager to restart a failed worker.
- Our patch directory for all patches is located at: /ora_apps/patch
- When we download patches from Oracle Support, we download them to our production database and applications servers as well as to our test servers. This ensures that if we refresh the test servers and write over the patch directory, we won't have to go back to Oracle Support to request that the patch be put out to their website again.
- We created subdirectories for each of the major versions of upgrades:
11.0.2 The database version we started with

- 11.0.3 The upgrade from 11.0.2 that we initially performed, which came on a CD
- 11.0.3+ The additional Megapatches that we applied. We encountered many problems after upgrading to 11.0.3, and were repeatedly asked by Oracle Support to apply additional megapatches. Until 11.0.4 becomes available, the only way you can migrate from 11.0.2 to these megapatches is to upgrade to 11.0.3 first, and then apply megapatches one by one from there. One of the biggest problems that we had when we began applying megapatches was that a patch would fail because we were missing some pre-requisite. We wasted many hours on the phone with assorted Oracle Support analysts who gave us conflicting stories about which megapatches we needed, and in which order the megapatches needed to be applied. Based on our experiences, we will hesitate to apply megapatches in the future and will wait instead for the next point release when it becomes available. We believe that a point release is more stable and requires less time and problems to upgrade than megapatches.
- postpost11.0.3.1 These were additional patches that trickled in for assorted problems after we finished upgrading our DEVL environment to 11.0.3+ and before we could implement those changes into production. We store all new patches in /ora_apps/patch/postpost11.0.3.1.
- When we finish applying a patch, we change the name of the patch, using the mv command, to patchnameDONE:
patch 165344 would become 165344DONE by typing:
mv 165344 165344DONE
 - You might also consider naming the patch so you can remember what the patch was supposed to fix:
mv 165344 165344AcctPerfDONE

SPECIAL NOTE: The times listed in our chart of patches show how long it took us to apply patches *after* we battled through problems with them on the first two attempts. These times do not reflect the hours spent with Oracle Support diagnosing problems with the patches. This is why we recommend that you run through the upgrade at least twice before attempting to upgrade your production environment.

We confess to having a few one-off patches where we do not know what they were supposed to fix and cannot tell what they were supposed to fix from the readme.txt.

Notice in the timings how much longer it takes to dredge through the Megapatches than to run a point release upgrade.

APPLICATION SERVER upi_tapp

cd to the right directory, then ls to make sure you are in the right directory. Then type adpatch and answer the questions

On the apps server, answer the questions as follows:

Is this the correct APPL_TOP [Yes] ? **Y**

Filename [adpatch.log] : **PRESS THE ENTER KEY**

Do you wish to activate this feature [Yes] ?**N**

Please enter the batchsize [1000] : **PRESS THE ENTER KEY**

Do you currently have files used for installing or upgrading the database installed in this \$APPL_TOP [Yes] ?**N**

Do you currently have Java and HTML files for Self-Service Applications installed in this APPL_TOP [Yes] ?**Y**

Do you currently have Oracle Applications forms files installed in this \$APPL_TOP [Yes] ?**Y**

Do you currently have concurrent program files installed in this \$APPL_TOP [Yes] ?**N**

You are about to apply a patch to the installation of Oracle Applications in your ORACLE database 'DEVL'

using ORACLE executables in '/ora_dev/ora_apps/d01/app/oraupi/product/805'.

Is this the correct database [Yes] ?**Y**

Enter the password for your 'SYSTEM' ORACLE schema:**ENTER PASSWORD**

Enter the ORACLE password of Application Object Library [APPS] :**ENTER PASSWORD**

The default directory is [/ora_dev/ora_apps/patch/11.0.3/pre1103/793451] :**PRESS THE ENTER KEY**

Please enter the name of your AutoPatch driver file [patch.drv] : **c793451.drv** {This is the c driver with the patch number and then .drv after it)

Do you want to continue with AutoPatch [Yes] ?**Y**

When you are all done, you should cd .. to move back to the main directory, then cd to the directory for the next patch that you want to apply.

DATABASE SERVER upi_tdb

cd to the right directory, then ls to make sure you are in the right directory.
Then type adpatch and answer the questions

On the database server, you answer the questions as follows:

Is this the correct APPL_TOP [Yes] ? **Y**

Filename [adpatch.log] : **PRESS THE ENTER KEY**

Do you wish to activate this feature [Yes] ?**N**

Please enter the batchsize [1000] : **PRESS THE ENTER KEY**

Do you currently have files used for installing or upgrading the database installed in this \$APPL_TOP [Yes] ?**Y**

Do you currently have Java and HTML files for Self-Service Applications installed in this APPL_TOP [Yes] ?**N**

Do you currently have Oracle Applications forms files installed in this \$APPL_TOP [Yes] ?**N**

Do you currently have concurrent program files installed in this \$APPL_TOP [Yes] ?**Y**

You are about to apply a patch to the installation of Oracle Applications in your ORACLE database 'DEVL'

using ORACLE executables in '/ora_dev/ora_apps/d01/app/oraapi/product/805'.

Is this the correct database [Yes] ?**Y**

Enter the password for your 'SYSTEM' ORACLE schema:**ENTER PASSWORD**

Enter the ORACLE password of Application Object Library [APPS] :**ENTER PASSWORD**

The default directory is [/ora_dev/ora_apps/patch/11.0.3/pre1103/793451] :**PRESS THE ENTER KEY**

Please enter the name of your AutoPatch driver file [patch.drv] : **c793451.drv** {This is the c driver with the patch number and then .drv after it)

Do you want to continue with AutoPatch [Yes] ?**Y**

If the driver completes successfully, you'll see the following message:

AutoPatch is complete.

AutoPatch may have written informational messages to the file
/ora_dev/ora_apps/d01/app/applupi/1102/admin/TEST/log/adpatch.lgi

You should check the file
/ora_dev/ora_apps/d01/upid01/app/applupi/1102/admin/TEST/log/adpatch.log
for errors.

If it doesn't complete successfully you should come and find me.

When you are done applying all the drivers for a given patch, cd .. to move back to the main directory and then cd to the next patch directory.

PATCH	DESCRIPTION	TOT TIME	APPS TIME	DB TIME
Pre-work to fix things so patches would bomb less	Changes were to production, primarily moving files from the wrong server and changing ownership on files.	2 hr		
11.0.3 Pre-upgrade 793451	/oracle/patch/11.0.3/pre1103 pre-install mode (pre1103 directory)	9min	upi_tapp C 7min	upi_tdb C 2min
693219	AOL prereq if not JL (Latin Am localizations)	14min	C 7min G 5min	C 1min G 1min
770675	BOM prereq	14min	C 4min G 3min	C 3min D 4min G 3min
696537	HR prereq Don't apply ecpatch as we don't use ec stuff	6min	C 3min	C 3min
\$APPL_TOP/admin/adsysapp.sql	sqlplus apps/password @\$APPL_TOP/admin.adsysapp.sql	5min		5min
11.0.3 Upgrade merg1103	/oracle/patch/11.0.3/merg1103/apps1103 don't apply hr1103 as we don't use it	5hr39 min	cr1103.drv 32min	cr1103.drv 1hr37min dr1103.drv 3hr30min
11.0.3 Post-upgrade c857097.drv	/oracle/patch/11.0.3/post1103 don't apply mrc1103.zip as we don't use that	6 min	C 5min	C 1 min
TOTAL 11.0.2 -11.0.3 UPGRADE		8 hr 33 min		
11.0.3.1 (MEGAPATCHES) 1155774 – 11.0.AOL.F	-794429 was already applied by cr1103.drv in merg1103 -replaced 1003972 with 1155774 -You must follow the instructions in the README.txt BEFORE you start applying the patch if you want to avoid having the patch fail	3hr45 min	C 11min G 4 min	C 50min D 60min G 1hr30min
961231 - 11.0.AR.D	Modify patch 961231 to set the initial extent to 1M and next extent to 4M in files b644217.sql and b932818a.sql before applying patches to upi_tdb.	4hr50 min	C 10min G 1hr30min	C 1hr15min D 1hr30min G 15min
953727 - 11.0.INV.D		42min	C 5min G 1min	C 20min D 13min G 3min
1039837 - prereq to OE D		3min	C 1min	C 1min D 1min
953825 - 11.0.OE.D	- This patch hung forever, so we changed the contents of this patch to use these statements instead: delete from mtl_so_rma_interface where rma_line_id in (select rma_line_id from mtl_so_rma_interface minus select line_id from so_lines_all); delete from mtl_so_rma_receipts where rma_interface_id in (select rma_interface_id from mtl_so_rma_receipts minus select rma_interface_id from mtl_so_rma_interface);	3hr	C 8min G 1hr	C 60min D 45min G 5min

PATCH	DESCRIPTION	TOT TIME	APPS TIME	DB TIME
	<p>OR:</p> <p>delete from mtl_so_rma_interface where not exists (select 1 from so_lines_all where line_id = mtl_so_rma_interface.rma_line_id);</p> <p>delete from mtl_so_rma_receipts where not exists (select 1 from mtl_so_rma_interface where rma_interface_id = mtl_so_rma_receipts.rma_interface_id);</p> <p>Which is best varies on the size of the tables. NOT IN is the absolute worst when the inner table is large.</p>			
1064696 - misc OE patch		7min	C 3min	C 4min
986073 - 11.0.AP.D	- We had to copy apsetdef.sql from apps server to db server before applying patch	1hr30 min	C 18min G 10min	C 13min D 36min G 12min
991532 - 11.0.GL.D		1hr	C 7min G 14min	C 12min D 19min G 9min
954257 - 11.0.PO.D		1hr10 min	C 4min G 25 min	C 13min D 20min G 9min
731727 then	731727 is a pre-req for 968037	9min	C 1min	C 2min D 6min
968037 - 11.0.MRP.D		3hr51 min	C 10min G 14min	C 3hrs D 22min G 5min
964665 - 11.0.BOM.D		51min	C 4min G 12min	C 15min D 15min G 5min
1179399 TAX cumulative patch	/ora_apps/patch/post11.0.3.1 Fixed a pile of problems that accounting was looking at. Includes AutoInvoice performance (Not in 11.0.AR.D).	40min	C 2min G 2min	C 10min D 19min G 3min
1067485	Fixes PO Megapatch by adding missing programs	18min	C 4 min G 3 min	C 5 min D 4 min G 2 min
915022	Fixes AP Megapatch by adding missing programs	10min	C 2 min G 2 min	C 2 min D 3 min G 1 min
TOTAL 11.0.3 MEGAPATCH		About 22 hours		
ONE-OFF PATCHES	Following are patches that we highly recommend you consider for your environment.			
897214 PERFORMANCE PATCH – HIGHLY RECOMMENDED	WFPurgePerf – 897214 and 990511 create a new concurrent request, called Purge Obsolete Workflow Data, that cleans out your workflow tables. You should run this concurrent request nightly. Requested/Tested by Barb. Applied to Production 8/20/00. Submitted Purge Obsolete	5min	C 1min	C 1min D 3min

PATCH	DESCRIPTION	TOT TIME	APPS TIME	DB TIME
	Workflow Data concurrent request to run nightly.			
990511 PERFORMANCE PATCH – HIGHLY RECOMMENDED	WFPurgePerf – Oracle wrote this patch because by the time we noticed that we needed to Purge Obsolete Workflow Data, the tables were so enormous that Purge Obsolete Workflow Data couldn't run to completion the first time without running out of rollback segments, no matter how large we made them. Requested/Tested by Barb. Applied to Production 8/20/00. Submitted Purge Obsolete Workflow Data concurrent request to run nightly.	4 min	C 1min	C 1min D 2min
928739 PERFORMANCE PATCH – HIGHLY RECOMMENDED	CollReqStatPerf Profile option Concurrent: Collect Request Statistics is seeded by Oracle without being set. If you set it to N, it still puts records in the applsys.fnd_conc_req_stat table. This patch makes the N setting work. I set it to N, then truncated the table, then ran some concurrent requests to make sure the table was no longer being populated. Works like a charm. Oracle apparently intends to use this table to someday collect concurrent manager statistics, but this functionality doesn't currently work. Requested/Tested by Barb. Applied to Production 8/20/00. Table had 3300372 records. Set Concurrent: Collect Request Statistics to N. Truncated it. Ran some concurrent requests. Table now has 0 rows.	5min	C 2min	C 1min D 2min
754671 PERFORMANCE PATCH – HIGHLY RECOMMENDED	ConcMgrPerf This is a performance patch for the Standard Request Submission form. It creates a number of indexes that make the Purge Concurrent Request program run much faster. Requested/Tested by Barb. Applied to Production 8/20/00.	11min	C 2min G 3min	C 1min D 3min D 1min G 1min
	One of these patches (we don't know which) added a new profile option called Concurrent: Enable Request Submission in View Mode. It is defaulted to N, which causes certain responsibilities like SYSADMIN to not be able to click on the Submit Request button following certain menu paths. To fix this, change this profile option to Y and then log back in again.			
TOTAL Highly Recommended 1-Off Patches		25min		
Other Patches	These are 1-Off Patches that we've applied to deal with odd problems. We've included them more for our information than yours, so feel free to ignore this section.			
1035788	PO	35min	C 5min G 4min	C 13min D 12min G 1min
1196184	PO	7min	C 1min G 4min	C 1min G 1min

PATCH	DESCRIPTION	TOT TIME	APPS TIME	DB TIME
998411	PO	8min	C 1min G 3min	C 2min D 1min G 1min
856365	AOL – no idea how to test this, as it is changing .o files and the readme doesn't explain what it fixes. Concurrent Manager not processing subrequests.	3min	C 1min	C 2min
973520	AR – only works on 11.0.3, fixes a P1 problem with sales tax.	16min	C 3min	C 1min D 12min
1195777	Must run against Taxware 3.1. Supposed to fix a P1 problem with Sales Tax. Didn't help.			
1305553	Supposed to fix Order Entry P1 problem with Sales Tax. Applied to 11.0.3 DEVL. Didn't help.	30min	C 2min G 4min	C 12min D 11min G 1min
742104	Provides version 110.5 of raaira.lpc. Fixes problem where credit memos are not writing to AR_PAYMENT_SCHEDULES and AR_RECEIVABLE_APPLICATIONS correctly. Applied TEST and PRODUCTION (11.0.2).		C 1min	C 1min
Problem telnetting to apps servers after HP Patch	Paul has applied a workaround (is it included in the restart script for the test servers?).			
UPI_INVP_CMERGE_SPDM.sql	Located under \$UPI_TOP/sql. Copy this on top of \$INV_TOP/sql/INVP_CMERGE_SPDM.sql to improve performance until Oracle makes this into a performance patch. We have submitted this to Oracle to improve the performance of the Customer Merge. When it becomes available as a patch, we'll update this paper with the patch number. Only works with 11.0.3.	5 min		5 min
995035	Makes Receivables Interface provide more detailed information. For Dan's problem with autoinvoicing. Applied 995035 to the 11.0.2 TEST database instance.		C G	C D G
1043465	Helps with Sean's problems.		C D	C D G
894094	This is the patch that makes a sales tax problem go away by eliminating the ability to view sales tax. We decided not to reapply, as it didn't resolve the problem.	NA	NA	NA
707772	Error messages on the validation report for autoinvoice (raxtrx) are preventing you from completing the invoice process for customers. Autoinvoice is modified to ignore statuses. Applied TEST (11.0.2) and ORAUPI (11.0.2). To make this work, had to replace ARXRWMAI.fmb with an original copy (because we had customized it), and fixed WSHFSCDL.fmx by correcting a spelling error in package body sc_del_lines_update_pvt. The error is X_LastWUpdated_By should be X_Last_Updated_By.	2hr45 min	C 25min G 10min	C 1hr45 min D 15min G 10min

PATCH	DESCRIPTION	TOT TIME	APPS TIME	DB TIME
707722	<p>Autoinvoice was setting the bill_to_site_use_id of RA_CUSTOMER_TRX table to NULL if the bill_to site is inactive. This bill_to_site_use_id is an important one irrespective of the status. Also, once the order is shipped, it should be billed irrespective of the bill_to site's status. This bug has been fixed.</p> <p>Applied 10/23/00 to TEST after a refresh from production. C driver on apps server kept failing, saying that forms458.a had an early end of file. Renamed forms458.a to forms458.a.barb and ftp'd forms458.a from the database server (the files were dated the same and were the same size, but perhaps a bit came across wrong in the refresh). Ran: as Oracle, cd \$ORACLE_HOME/forms45/lib, make -f ins_forms45wv8.mk, which worked this time. Then re-started the patch and it worked fine. There is a newer version of Developer 2000 available, which we would have upgraded to if this hadn't worked. New version is available on Oracle's ftp site at apps/patchsets/aol/d2kpatch12, HP9800, OS11, Requires that you apply an interoperability patch 1317243, available under Metalink/patches. Apply the interoperability patch after the D2K patch. We won't upgrade for now.</p> <p>G driver on apps server failed with: The following Oracle Forms objects did not generate successfully: ar forms/US ARXCWMAI.fmx ar forms/US ARXRWMAI.fmx This was due to custom code for these forms. After fixing this problem, received FRM-40800 User exit FND does not exist when trying to log onto the apps. Had to cd \$ORACLE_HOME/bin and rename f45webm to f45webm.notapps, then cd \$FND_TOP/bin and run adrelink force=y ranlib=y "fnd f45webm" to get this to go away.</p>		C 5 min G 30 min	C 1 hr D 10 min G 2 min
1168375	Dunning Letter year is truncated to 2 digits, such that year 2000 is printed as 20. Increased size of c_dun_date, f_dun_date and f_dun_date1. Applied TEST and PRODUCTION. Fixes problem.	Not sure	Not sure	Not sure
1182741	<p>Applied to our 11.0.2 Test & Production environments.</p> <p>1. Problem Description Receivables Interface errors out with 'ORA-01458: invalid length inside variable character string' when freight currency is different than order currency/delivery currency This is happening only on some platforms</p> <p>2. Solution Description Resolved the issue of memory corruption causing it to happen. 3. Fixed File Names and versions oeifcc.lpc Ver 110.11</p>			
805759	When printing dunning letters, balance due amount	c 3min		C 4min

PATCH	DESCRIPTION	TOT TIME	APPS TIME	DB TIME
	was less than the invoice amount. by an amount equal to the finance charge amount. This occurred only when the system option "Accrue Interest" was set to "yes". Otherwise the finance charge was OK. Fix was made to arfups and armpps functions in arfccf.lpc and armpps.lpc files before calling the update payment schedule routine armups. Applied TEST and PRODUCTION 11.0.2. Fixed the problem.			D 5min
Oracle Recommended HP Patch Sets	Applied to upi_tdb and upi_tapp: XSWGRI100 June 2000 quarterly patch set PHCO_16383 System Patch Tool PHCO_16438 PFS cumulative patch PHCO_19090 libc cumulative patch PHCO_19666 libpthreads cumulative patch PHCO_20882 fsck_vsfs(1m) cumulative patch PHCO_21187 Cumulative SAM.ObAM patch PHKL_18543 PM/VM/UFS/async/scsi/io /DMAPI/JFS/perf patch PHKL_20016 2nd CPU not recognized in G70/H70/I70 PHKL_20202 Fix pthread error return, nfs/tcp panic PHKL_20674 fix VxFS unmount hang & MMF sync panics PHKL_21392 VxFS performance, hang, icache, DPF's PHKL_21532 boot, JFS, IOperf, PA8600, 3Gdata,NFS,IDS,PM,VM PHNE_15995 ARPA Transport cumulative patch PHSS_16404 id(1) and linker tools cumulative patch PHSS_21493 X/Motif2.1 Runtime APR2000 Periodic Patch	Not sure	Not sure	Not sure
Total 1-Off Patches				
ADADMIN	On DB run Option 2, then Option 2,5,6 On App Server run Option 1, then Option 2,5,6	5hr	#2 7 min #5 6 min #6 2 hrs 30 min	#2:1 hr 10 min #5 5 min #6 50 min
RMINC_FIX_OBJECTS	@\$UPI_TOP/sql/RMINC_fix_objects.sql (this just recompiles any invalid objects... generally after applying patches there are a lot of them).	30min		30min
Customizations	Once all the changes are made, we had to reapply our customizations.	30min		

One Final Note:

This paper and presentation are available at www.oncalldb.com. We'll likely update it as we continue our cloning and upgrading and patching process, so you may want to check in from time to time to see if there are any changes.

Conclusion

We called this paper "Not For the Faint of Heart" because cloning and upgrading applications databases is so complex to do, and certainly all the options can be difficult to get your head around. If you have written procedures or programs that help with this task, or have found workarounds for patch problems, or have even discovered some patches that are 'must haves', we'd love to hear from you. And if you can think of easier ways to do what we've described, let us know, because we'd like this to be as simple as possible.

About the Authors

Barbara Matthews
Principal Consultant
OnCallDBA.com
barb@oncalldb.com

Paul Greenwood
Unix Guy/DBA/Applications System Administrator
Ultradent
gpaul@ultradent.com

Last revised: August 31, 2000