Oracle9*i* release 2 database migration from PA-RISC to HP-UX 11i v2 on the Intel® Itanium® architecture

Table of contents

Abstract

Migration of an Oracle9*i* release 2 database from a PA-RISC-based HP-UX server to an Itanium®-based server running HP-UX is a very simple process, thanks to the compatibility between HP-UX for PA-RISC and HP-UX for the Itanium architecture. This paper, which is targeted at IT professionals with HP-UX and Oracle® administration expertise, documents the migration process and includes several sample migration scenarios with step-by-step instructions. Also included are results of performance testing that demonstrate there is no discernible performance difference between a database migrated from PA-RISC and a database created "natively" on an Itanium-based system.

Introduction

More than a decade ago, HP began working on a new computer architecture to serve as a follow-up to its highly successful PA-RISC architecture. HP called this architecture EPIC (Explicitly Parallel Instruction Computing) and, in 1994, partnered with Intel® to create Intel's new 64-bit Itanium® architecture. The Intel Itanium Processor Family is now widely recognized as the next-generation standard computing platform.

With the introduction of HP-UX 11i v2, HP now offers an enterprise-ready operating system capable of supporting a full line of HP-UX UNIX® servers based on the Itanium Processor Family. Since Itanium-based servers running HP-UX will offer a substantial and cost-effective performance improvement over the previous generation of PA-RISC-based servers running HP-UX, we expect a great deal of interest in migrating to the Itanium Processor Family.

As members of HP's Oracle lab, we have extensively tested the process of migrating an Oracle database from a PA-RISC-based HP-UX environment to an Itanium-based server environment running HP-UX. This paper, the result of our extensive testing, is intended to be used as a guide for anyone who wishes to migrate an Oracle database from HP-UX on PA-RISC to HP-UX on an Itanium-based system.

In this paper we will discuss the direct result of testing that we've done in the lab for Oracle database migration scenarios on HP-UX. We will not discuss the migration of applications other than the Oracle database, nor will we discuss migration in non-HP-UX environments (such as Tru64 UNIX or Linux).

Migration made easy

One of the major goals HP had in co-designing the Intel Itanium architecture was to maintain a high degree of compatibility with the PA-RISC-based HP-UX environment. One result of this compatibility is that a database created on a PA-RISC system will not need any sort of data conversion in order to run properly on an Itanium-based system. Migrating an Oracle database in an HP-UX environment from PA-RISC to the Itanium architecture will thus be a very simple process. With the right level of preparation, as shown below, this will be as simple as shutting down the database on the PA-RISC HP-UX server and bringing it back up on the Itanium-based HP-UX server.

Intended audience

Though we do discuss some of the steps involved in performing such activities as installing Oracle and building HP-UX Logical Volume Manager (LVM) volume groups, we assume that anyone using this paper will have sufficient expertise to understand the Oracle and HP-UX concepts that we discuss:

- **HP-UX administration:** creating Logical Volume Manager volume groups and logical volumes, account/user administration, HP-UX software/patch installation, and remote-login administration (/etc/rhosts and .rhosts)
- **Oracle database:** installing, laying out, configuring, and backing up a database

Migration process summary

To migrate an Oracle database from HP-UX on PA-RISC to HP-UX on the Intel Itanium architecture, the database on the PA-RISC system must be running one of the following Oracle versions:

- **Oracle9*i* release 2 (9.2.0.2)**
- **Oracle8*i* (8.1.7):** an 8.1.7 database can be migrated to the Intel Itanium architecture, but it **must** be upgraded to Oracle9*i* release 2 before it can be opened

An Oracle8*i* database can thus either be upgraded to Oracle9*i* release 2 and then migrated to the Intel Itanium architecture or migrated first and then upgraded to Oracle9*i* release 2 on the Itanium-based system.

The process of migrating an Oracle database from a PA-RISC system to an Itanium-based system can be summarized as follows:

- **Upgrade the Oracle database** on the PA-RISC system to either Oracle8*i* (8.1.7) or Oracle9*i* release 2
- **Prepare the Itanium-based system**, including HP-UX installation, Serviceguard installation (for a Real Application Cluster [RAC] database), and Oracle installation. Also, create Logical Volume Manager structure and file systems (if necessary).
- **Copy or move database files** from the PA-RISC system to the Itanium-based system; moving the database files (i.e., moving the database's disks over to the Itanium-based system or reconfiguring your SAN to permit the Itanium-based system to access them) is generally quicker and easier than copying, but copying allows you to keep the database in place on the PA-RISC system as a backup
- **Modify the database configuration files** (e.g., init.ora, listener.ora) to reflect the Itanium-based system's name and IP address
- **Upgrade the database to Oracle9*i* release 2**, if not already done

These steps may differ slightly depending on the nature of your Oracle installation—the differences will be highlighted in the step-by-step instructions below. We've included instructions which will accommodate the following three Oracle scenarios:

- **Single-instance Oracle** (non-clustered) implemented using **file systems** built on top of LVM logical volumes
- **Single-instance Oracle** (non-clustered) implemented using **raw volumes** (LVM logical volumes without file systems)
- **Clustered Oracle** (Oracle9*i* Real Application Cluster) implemented using **raw volumes** (LVM logical volumes without file systems)

We will not discuss clustered Oracle with file systems because this combination is not supported with HP-UX. We will not consider the case of raw volumes implemented with so-called "hard partitions" (non-LVM raw disks) as this approach is very rarely used.

Minimize downtime

As you'll see in the following sections, there is no need to do any data conversion to migrate the database. This is of great advantage not only for ease of migration but also for minimal downtime. The exact same database can be used when moving to the new Itanium-based systems. A simple database shutdown from the PA-RISC server and then a startup from the Itanium-based server is all that is needed. Another option could be to do a business copy of your database and then move that copy to be connected to the Itanium-based systems. You know your environment best and which process makes the most sense. We guarantee the data compatibility, and that leaves you with the flexibility to use whichever method suits your environment best.

"Migrated database" vs. "native database" performance

Before getting into the details of the migration, you may be wondering how your existing database will perform on the new server. It's been well documented that application programs will perform better when they are recompiled into native object code, but does this "native advantage" apply to data as well? Is there a performance advantage in rebuilding a PA-RISC database so it is "native" on an Itanium-based system? Great news: the answer is no. There is no need to rebuild your database.

There is no special conversion being made when moving an Oracle database between PA-RISC and the Intel Itanium architecture. The database structures and the database block layout are identical to PA-RISC, so when you move your PA-RISC database to the Itanium architecture, the database engine has no knowledge of the fact that it is operating on data that was created on another platform. There would be no noticeable performance benefit from going through the process of re-creating your database natively on the Itanium-based system.

Laboratory performance comparisons

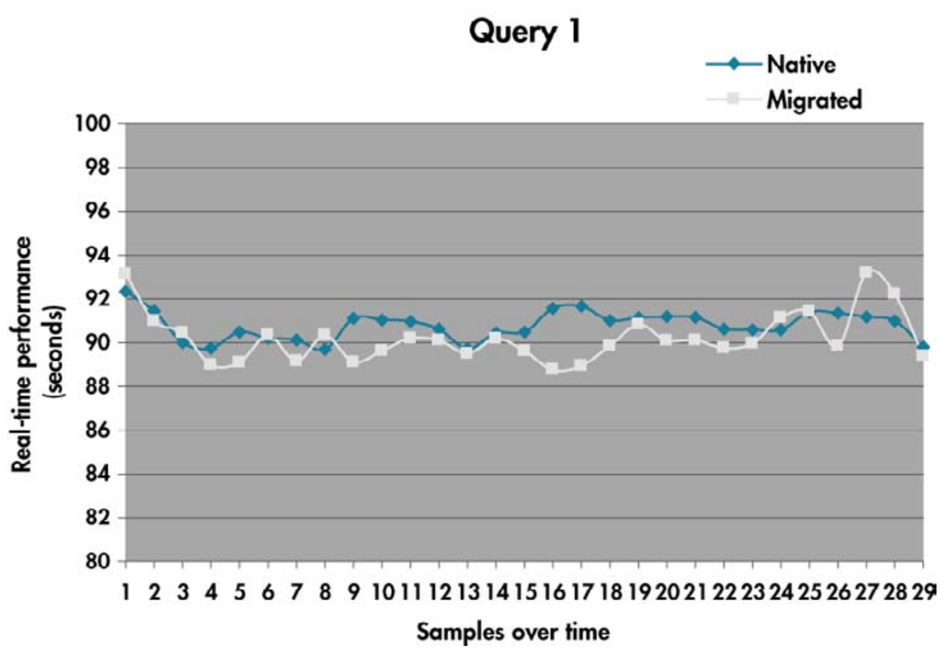
To test the performance, some decision support system (DSS) workloads from a major standard DSS benchmark suite were used, running several tests on the same system. These DSS workloads were chosen because they included more than 20 queries that stress the system in different ways (i.e., I/O intensive, CPU intensive, etc.).

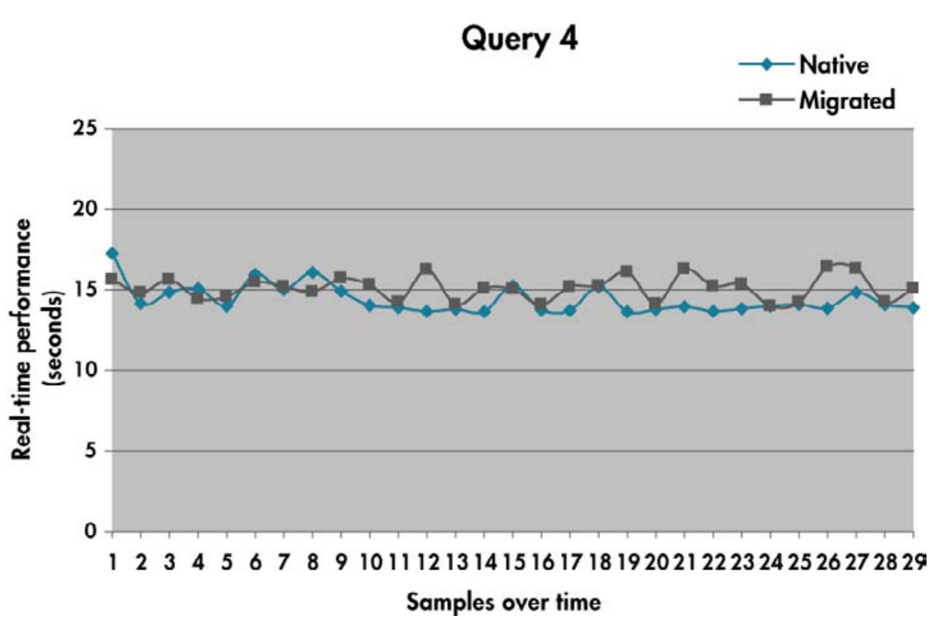
The following methodology was used:

1) Run the whole package natively on the Itanium-based system. That includes creating the database, populating it, and running the queries.
2) Create and populate the same database on a PA-RISC environment and then migrate that database to the Itanium-based system, where the same query workload was run and measured.
3) Compare the performance results achieved during steps 1 and 2.

In both cases, the queries were run and measured on the Itanium-based system. The only difference is that the database was created in different Oracle9.2 environments (PA-RISC vs. Intel Itanium architecture).

The following charts show the performance of two of the queries over time. In the interest of space, we didn't include all 22 queries since they all showed similar behavior.





While there was some fluctuation in the individual data points due to the random nature of these tests, the graphs show that overall there is very little, if any, performance difference between the "native" and "migrated" databases.

Incidentally, we used a migrated database on a four-way Itanium-based HP Server rx5670 when we ran the Oracle Applications Standard Benchmark, the results of which we published in early 2003. In the highly competitive world of industry-standard benchmarks, every effort is made to eliminate potential performance issues so we get the best possible result—and we certainly wouldn't have used a migrated database if there was a performance penalty associated with it.

Migration process step-by-step

Note: The following steps are presented as a guide for users experienced with HP-UX administration concepts (such as managing Logical Volume Manager) and Oracle administration concepts (such as installing and configuring Oracle). We've assumed an appropriate level of familiarity with HP-UX commands in the list that follows. If you need more detail than what's included here, check the man page for the command in question. **Unless otherwise specified, all the following HP-UX commands should be performed when logged in to the system as the root user.**

1) Ensure that the database on the PA-RISC system is Oracle9*i* release 2 (version 9.2.0.x.x) or Oracle8*i* (version 8.1.7). Upgrade to either Oracle9*i* release 2 or Oracle8*i* version 8.1.7 if necessary.
2) Prepare the Itanium-based system(s) for Oracle migration:
   a. Install HP-UX 11i version 2 and any recommended patches.
   b. For a clustered (RAC) installation, install Serviceguard Extension for Real Application Clusters (eRAC) version 11.14.01. Configure the multi-node cluster using standard Serviceguard procedures.
   c. Modify the HP-UX user/account database (/etc/passwd and /etc/group) to ensure the Oracle owner and dba group are set up correctly.
   d. Verify that Oracle9*i* release 2 has been installed.

If Oracle9*i* release 2 is installed, continue to the next step. If you are installing a RAC database and the Oracle software was installed as a single instance, you'll need to enable the RAC option by relinking the executables as follows:

```
cd $OH/rdbms/lib
make –f ins_rdbms.mk rac_on ioracle
```

If Oracle9*i* release 2 has not been installed, install it now. If you are installing a RAC (clustered) database, make sure to install the Real Application Clusters option. Remember, your Serviceguard cluster must be running when installing the Oracle RAC option.

   e. Create the LVM structure, which will be used to hold the database.
      i. Create the volume group directories (e.g., *mkdir* /dev/vgoracle) and the LVM "group" files (*mknod* /dev/vgoracle/group . . .) using standard LVM procedures.
      ii. If you'll be copying your database to the Itanium-based system, create the volume groups and logical volumes using standard LVM procedures (*vgcreate* and *lvcreate*). If your Oracle database is implemented using file systems, create the file systems (*newfs*).
   f. If your Oracle database is implemented using file systems, create the file-system mount points (e.g., *mkdir* /ora01).
3) Copy or move your database to the Itanium-based system(s). **Select step (a) if you're copying or step (b) if you're moving.**
   a. If you're copying your database, shut down the database on the PA-RISC system, and then copy the data over to the Itanium-based system using whatever method is most appropriate for your environment and that you're most comfortable with. For example, for a raw-volume implementation, you might prefer to copy the raw volumes to tape and then restore the tape on the Itanium-based system. If both the PA-RISC and Itanium-based systems are on the same network, you could use *dd* through a network pipe to the Itanium-based system, as in this example:

```
cd /dev/vgoracle
dd if=roravol1 | remsh IPF1 "dd of=/dev/vgoracle/roravol1"
```

   (Repeat the *dd* command for each additional logical volume to be copied.)

   You could also use a network pipe if your database is file system based. For example, to copy all the files from the "/dir" directory on the PA-RISC system to the "/dir" directory on the Itanium-based system called

"IPF1," issue the following commands while logged in to the PA-RISC system (be sure to type these correctly—especially the blank spaces which delimit the parentheses):

```
cd /dir
tar cf - . | ( remsh IPF1 cd /dir  \; tar xf - )
```

Regardless of whether your database is raw or file system based, be sure to copy the critical incidental Oracle files such as the control files, init.ora files, listener.ora and tnsnames.ora, etc.

b. If you're moving your database:

   i. Shut down the database. Making a backup copy is highly recommended.

   ii. For a file-system-based database, unmount the file system(s) using the *umount* command.

   iii. Deactivate the volume group(s) using the *vgchange* command.

   iv. Create a map file, which will facilitate moving the volume group(s) to the Itanium-based system by using the *vgexport* command. Use the *–p* (preview) option to avoid expunging the volume group from the PA-RISC system (to allow for backtracking if necessary). We also recommend using the *–s* (shared) option, which was designed to simplify the process of moving volume groups between the nodes of a clustered system, but which is also available for non-clustered systems. Here's an example of a *vgexport* command, which will create a map file named 'mapfile' for the volume group 'vgORA':

   ```
   vgexport -p -m mapfile -s /dev/vgORA
   ```

   Copy the map file to the Itanium-based system (where it will be used by a *vgimport* command to re-create the volume group).

   v. Provide the Itanium-based system access to the disks containing the database. This could require physically recabling the disks from the PA-RISC system to the Itanium-based system. It could also be accomplished by reconfiguring your SAN, or (in the case of a disk array) reconfiguring the LUNs to be visible to the Itanium-based system. On the Itanium-based server, perform an *ioscan* and then an *insf* to initiate configuration of the newly added disks or LUNs.

   vi. Import the volume group(s) using the map files and the volume group directories and group files you created earlier. For example, use:

   ```
   vgimport  -m mapfile -s /dev/vgORA
   ```

   vii. Change the volume group special files to be owned by the same owner and group as on the PA-RISC system, and ensure correct permissions:

   ```
   chown -R oracle:dba /dev/vgrac
   chmod 644 /dev/vgrac/*
   ```

   viii. Activate the volume group(s) using *vgchange –a y*. (For a clustered system, you'll need to take the extra steps to change the volume groups to permit sharing, using *vgchange –c y –S y*, and then activate the volume group in shared mode using *vgchange –a s*.)

   ix. If your database is implemented using file systems, mount the logical volumes contained in the volume groups you've just activated (e.g., *mount /dev/vgORA/lv01 /ora01*).

4) On the Itanium-based system, edit the Oracle configuration files (e.g., init.ora, listener.ora, etc.) to reflect the new IP address, hostname, pathnames, and other characteristics of the Itanium-based system.

5) Log in as the Oracle user and start up the database. (If this is a Real Application Clusters installation, verify that the RAC option is installed; *sqlplus* will show the phrase "With the Real Application Clusters option" in its banner as long as the database instance is running.)

6) For databases implemented on raw volumes, verify that the asynchronous I/O driver is properly configured and Oracle is using it. While the database is up, the *fuser /dev/async* command should indicate that the Oracle db-writer process (ora_dbw0_SID) has /dev/async open. If it doesn't, you'll need to take the following steps to enable asynchronous I/O (for further details on setting up HP asynchronous I/O, see Oracle Metalink document ID 139272.1):

   a. Using System Administration Manager (SAM), ensure that the 'asyncdsk' driver is in the kernel.

   b. Create the async device file and change the ownership and permissions:

   ```
   mknod /dev/async c 101 0x0
   chown oracle:dba /dev/async
   chmod 660 /dev/async
   ```

   c. Enable asynchronous I/O within Oracle by changing the disk_async_IO parameter to "true" in the initSID.ora file.

d. Make sure that the group which owns the database—usually 'dba'—is given the "MLOCK" privilege. This is accomplished with the following command:

```
setprivgrp dba MLOCK
```

You can verify that the privilege is properly set using the command *getprivgrp dba*, which should respond back with the string "dba: MLOCK".

7) If the database is Oracle8*i* version 8.1.7, then upgrade it to Oracle9*i* release 2 (9.2.0.2) using Oracle's recommended upgrade procedures.

Sample migrations

Single-instance Oracle on file systems

This is an example of migrating a file-system-based single-instance database from PA-RISC to an Itanium-based system.

Source (PA-RISC) environment:

- Single-node PA-RISC server running HP-UX 11.0 operating system
- Oracle8*i* database (version 8.1.7)
- Database implemented using file systems built on LVM logical volumes

Upgrade to Oracle9*i* release 2

Before migrating the Oracle8*i* database to the Itanium-based system, we upgraded it to Oracle9*i* release 2. This was accomplished in place on the PA-RISC system using the standard upgrade procedure. Oracle's *dbua* utility, the Database Upgrade Assistant, was employed in order to simplify the migration. (For more information on upgrading from Oracle8*i* to Oracle9*i*, please see Oracle Technology Network (OTN) or Oracle Metalink online.)

In keeping with the recommended upgrade procedure, the database was backed up prior to the upgrade to Oracle9*i*. We also tested the upgraded database on the PA-RISC system to make sure the upgrade worked successfully.

Instead of upgrading to Oracle9*i* release 2 before migrating, another option would have been to leave the Oracle8*i* database at version 8.1.7 during the migration to an Itanium-based system, and then upgrade it to Oracle9*i* release 2 once it's on the new system.

Migrating the Oracle9*i* release 2 database from PA-RISC to the Intel Itanium architecture

The Oracle software (version 9.2.0.2) was installed on the Itanium-based system, which was running HP-UX 11i v2 (11.23).

In the Oracle-home directory on the Itanium-based system, we set up the shell's environment file (.profile) to create the ORACLE_SID environment variable using the same SID as on the PA-RISC system. (In the text that follows, we'll refer to the Oracle Home on the PA-RISC system as "$oh-pa" and on the Itanium-based system as "$oh-ipf".)

Using *rcp*, we copied all the database configuration and parameter files (including the initSID.ora file) from the PA-RISC system to the Itanium-based system:

```
rcp $oh-pa/dbs/*.ora ipf-node:$oh-ipf/dbs
rcp $oh-pa/dbs/orapwsid ipf-node:$oh-ipf/dbs
rcp -r $oh-pa/admin/sid* ipf-node:$oh-ipf/admin
```

The last command shown above copied the admin directory, which contains bdump, udump, etc.

We ensured that the access permissions and file owners for all these files and directories were the same on the Itanium-based system as on the PA-RISC system. (Improperly set file permissions would cause an ORA-01102 when opening the database.)

Ordinarily, at this point one would have to choose between copying or moving the database to the Itanium-based system. We performed this exercise once using each method and documented them both here.

Alternative 1: copying the database to the Itanium-based system
By copying the database, we were able to leave the database intact on the PA-RISC system, allowing for a fallback in case of problems.

The copy operation was done using *rcp* across the network:

```
rcp -r $oh-pa/oradata/sid ipf-node:$oh-ipf/oradata/sid
```

We then logged back in as the Oracle user on the Itanium-based system (to properly set the environment variables) and successfully started up the database.

Alternative 2: moving the database to the Itanium-based system
By moving the database, we didn't have to wait for the files to copy, but we didn't have a database left on the PA-RISC system in case of problems.

First, we shut down the database on the PA-RISC system. Then we **logged in to the PA-RISC system as the root user**, unmounted the file system, and deactivated the volume group:

```
umount /opt/oracle/db
vgchange -a n /dev/vg_sample
```

Next, we used the *vgexport* command to create the map file:

```
vgexport -p -s -m vg.map /dev/vg_sample
```

Then we copied the map file to the Itanium-based system.

Now it was time to move the disks to the Itanium-based system. In this particular example, we were able to physically remove the hot-swappable FC10 disk module containing the volume group and plug it into an empty slot on an FC10 connected to the Itanium-based system. (We also could have recabled the FC10 from the PA-RISC system to the Itanium-based system.) Once we logged in to the Itanium-based system as the root user, a simple *ioscan* command was all that was needed to recognize the new disk.

Next, we imported the volume group to the Itanium-based system using *vgimport*. (Even though the device path for the newly added disk was different from the path on the PA-RISC system, *vgimport* successfully imported the volume group because of the information recorded in the map file by the *vgexport –s* option. Without the *–s* option, we would have had to manually edit the map file to specify the device path for the disk we moved to the Itanium-based system.)

```
vgimport -s -m vg.map /dev/vg_sample
```

Then we activated the newly imported volume group and mounted the file system:

```
vgchange -a y /dev/vg_sample
mount /dev/vg_sample/lvol_db /opt/oracle/db
```

Logging back in as the Oracle user, we were able to start up the database without difficulties.

Single-instance Oracle on raw volumes

This is an example of migrating a single-instance raw-volume database from PA-RISC to an Itanium-based system.

Source (PA-RISC) environment:

- Single-node PA-RISC server running HP-UX 11.0 operating system
- Oracle9*i* release 2 database (version 9.2.0.2)
- Database implemented using "raw" LVM logical volumes (no file systems)

Migrating the database from PA-RISC to the Intel Itanium architecture

The Oracle software (version 9.2.0.2) was installed on the Itanium-based system, which was running HP-UX 11i v2 (11.23).

In the Oracle-home directory on the Itanium-based system, we set up the shell's environment file (.profile) to create the ORACLE_SID environment variable using the same SID as on the PA-RISC system. Note that even though the database is implemented without file systems, the database configuration files (e.g., init.ora) still reside in a directory in a file system (known as the Oracle Home). In the text that follows, we'll refer to the Oracle Home on the PA-RISC system as "$oh-pa" and on the Itanium-based system as "$oh-ipf".

Using *rcp*, we copied all the database configuration and parameter files (including the initSID.ora file) from the PA-RISC system to the Itanium-based system:

```
rcp $oh-pa/dbs/*.ora ipf-node:$oh-ipf/dbs
rcp $oh-pa/dbs/orapwsid ipf-node:$oh-ipf/dbs
rcp -r $oh-pa/admin/sid* ipf-node:$oh-ipf/admin
```

The last command shown above copied the admin directory, which contains bdump, udump, etc.

We ensured that the access permissions and file owners for all these files and directories were the same on the Itanium-based system as on the PA-RISC system. (Improperly set file permissions would cause an ORA-01102 when opening the database.)

At this point we were ready to either copy or move the database to the Itanium-based system; we decided to move the database.

Moving the database to the Itanium-based system

First, we shut down and backed up the database on the PA-RISC system. Then we **logged in to the PA-RISC system as the root user** and deactivated the volume group:

```
vgchange –a n /dev/vg_disk1
```

Next, we used the *vgexport* command to create the map file:

```
vgexport –p –s –m vg.map /dev/vg_disk1
```

Then we copied the map file to the Itanium-based system.

Now it was time to move the disks to the Itanium-based system. In this particular example, we were able to physically remove the hot-swappable FC10 disk module that contained the volume group and plug it into an FC10 connected to the Itanium-based system. (We also could have recabled the FC10 from the PA-RISC system to the Itanium-based system). Once we logged in to the Itanium-based system as the root user, a simple *ioscan* command was all that was needed to recognize the new disk.

Next, we imported the volume group to the Itanium-based system using *vgimport*. (Even though the device path for the newly added disk was different from the path on the PA-RISC system, *vgimport* successfully imported the volume group because of the information recorded in the map file by the *vgexport –s* option. Without the *–s* option, we would have had to manually edit the map file to specify the device path for the disk we moved to the Itanium-based system.)

```
vgimport –s –m vg.map /dev/vg_disk1
```

Then we activated the newly imported volume group:

```
vgchange –a y /dev/vg_disk1
```

Finally, we configured asynchronous I/O using the steps outlined above in section 6 of the “Migration process step-by-step” chapter.

Logging back in as the Oracle user, we were able to start up the database without difficulty.

Multiple-instance Real Application Clusters

Source (PA-RISC) environment:

- Two-node PA-RISC cluster running HP-UX 11i (11.11) operating system; node names NodePA1 and NodePA2
- Oracle9*i* release 2 database (version 9.2.0.2) with the Real Application Clusters option
- Database implemented using raw LVM logical volumes (no file systems)
- FC10 storage, directly connected to each node

Preparing the Itanium-based system cluster

On both nodes in the Itanium-based system cluster we installed HP-UX 11i version 2 (11.23) and Serviceguard Extension for Real Application Clusters (eRAC) version 11.14.01. We configured the two-node cluster per the installation instructions for eRAC and then started up the cluster (using *cmruncl*).

Oracle was installed on both nodes of the Itanium-based system cluster, with the RAC option enabled.

Moving the database from PA-RISC to the Itanium-based system cluster

To move the database to the Itanium-based system cluster, we physically disconnected the disk cables from the PA-RISC cluster and reconnected them to the Itanium-based system cluster. Once the physical connection was made, we imported the volume group containing the database and then modified the Oracle configuration files before starting the database up.

Following are the steps we took.

On the PA-RISC nodes, we:

- Shut down the database **on both nodes**
- Deactivated the volume group **on both nodes**

```
vgchange –a n /dev/vgrac
```

- Saved the volume group configuration (on one of the nodes) as a map file, which we copied to the Itanium-based server nodes (note the use of the *–s* option to the *vgexport* command; this writes the VGID [Volume Group ID] to the map file, which will facilitate the import process)

```
vgexport –s –p –m /tmp/vgrac.map /dev/vgrac
rcp /tmp/vgrac.map NodeIPF1:/tmp
rcp /tmp/vgrac.map NodeIPF2:/tmp
```

- Physically disconnected the disk cables from both nodes

Then, on **each** of the two Itanium-based system nodes, we:

- Physically connected the disk cables and then ran *ioscan* to prompt recognition of the newly added disks
- Created the volume group directory and group file, then imported the volume group using the map file we created on the PA-RISC node; when we created the 'group' file, we used a minor number of 0x020000 after checking to make sure that no other volume group on the system was using this same minor number

```
mkdir /dev/vgrac
mknod /dev/vgrac/group c 64 0x020000
vgimport –s –m /tmp/vgrac.map /dev/vgrac
```

- Checked the /etc/lvmtab file to verify that the volume group was imported correctly (the output of *strings /etc/lvmtab* was checked for a listing of the /dev/vgrac volume group along with the pathname of the newly added disk)
- Changed the volume group's special files to be owned by user "oracle" and group "dba" and then set permissions correctly using

```
chown –R oracle:dba /dev/vgrac
chmod 644 /dev/vgrac/*
```

- Activated the volume group in shared mode using

```
vgchange –a s /dev/vgrac
```

- For ease of administration, on the PA-RISC system we had created symbolic links pointing to the device files identifying the raw logical volumes (these links were located in a directory under $ORACLE_HOME); to keep a similar environment on the new Itanium-based system, we copied those links from one of the PA-RISC nodes after logging in as user "oracle":

```
cd $OH/oradata/racdb
rcp NodePA1:/o9ir2/oradata/racdb/* .
```

- Copied and then edited init.ora and other configuration files to reflect the characteristics of the Itanium-based system environment (IP addresses, etc.)

```
cd $OH/admin/racdb/pfile
rcp NodePA1:/o9ir2/admin/racdb/pfile/*.ora .
```

  Note: we kept the SID and service names the same as on the PA-RISC nodes so these didn't need to be changed.

Once the Oracle configuration was properly edited, we were able to start up the database on each of the Itanium-based system nodes.

Conclusion

HP co-developed the Intel Itanium architecture to be compatible with PA-RISC, and this has led to migrations that are extremely simple, require no data conversions, and introduce no performance penalties for a migrated database.

HP customers can be confident in migrating their Oracle databases to the Intel Itanium architecture—confident that their migration will go smoothly and confident that their database will perform optimally despite having been migrated from another architecture.

For more information

For more information about Itanium-based solutions from HP, visit **www.hp.com/go/itanium**.

HP-UX 11i release names and release identifiers

With HP-UX 11i, HP delivers a highly available, secure, and manageable operating system that meets the demands of end-to-end Internet-critical computing. HP-UX 11i supports enterprise, mission-critical, and technical computing environments. HP-UX 11i is available on both PA-RISC systems and Itanium-based systems.

Each HP-UX 11i release has an associated release name and release identifier. The uname (1) command with the -r option returns the release identifier. The following table shows the releases available for HP-UX 11i.

Table 1. HP-UX 11i releases

Release name	Release identifier	Supported processor architecture
HP-UX 11i v1	B.11.11	PA-RISC
HP-UX 11i v1.5	B.11.20	Intel Itanium
HP-UX 11i v1.6	B.11.22	Intel Itanium
HP-UX 11i v2	B.11.23	Intel Itanium

5981-6427EN Rev. 1, 05/2003

invent