A Novel Approach of Using Data Guard for Disaster Recovery & Rolling Upgrades

Auf Akhtar, Syed S. Rizvi, and Khaled M. Elleithy
{sakhtar ,srizvi, elleithy}@bridgeport.edu
Computer Science and Engineering Department
University of Bridgeport
Bridgeport, CT 06601

Abstract

Though we already had a standardized way (from Oracle) for Disaster Recovery policy, a major crash of the PLM database made our DBAs to come up with a proposal to execute a procedure using Data Guard together with Oracle Transportable Tablespace which resulted in the successful and timely migration of the PLM Production from Oracle 9.2.0.3 on HP N-Class to Oracle 9.2.0.5 on HP-Superdome in just 8 minutes. This same procedure also gives a fast, reliable way to upgrade from Oracle9i to Oracle Database10g with minimal downtime. And future database upgrades get even easier using Data Guard 10g Rolling Upgrades.

Index Terms— Oracle Data Guard, database, disaster recovery.

1. Introduction

Adidas is the leading sports apparel manufacturer selling its products worldwide. Adidas has recently acquired Reebok to compete with Nike. Adidas has deployed Oracle Data Guard [1] for disaster recovery (DR) protection for numerous Oracle9*i* databases. Data Guard, a built-in feature of Oracle Database Enterprise Edition, guarantees a disaster recovery solution that is database aware and fully integrated with other Oracle High Availability (HA) features.

Adidas also has a number of new projects in development on Oracle Database 10g. Adidas plans to use Data Guard10*g* Rolling Upgrades, a new feature that dramatically reduces the downtime required to upgrade from one database release to the next (Oracle 10.1.0.3 is the minimum database release required for rolling upgrades). Like many Oracle users, Adidas will soon upgrade existing production systems from Oracle9*i* to Oracle Database 10*g*. Adidas's testing showed that in their environment, the upgrade from Oracle9*i* to Oracle 10*g* requires a minimum of 25 minutes to complete. This would break the Adidas IT service level agreement that stipulates planned database downtime cannot exceed 15 minutes per quarter. A solution needed to be found. Demonstrating the ingenuity that Adidas IT has become known for, they developed a procedure using the combination of Oracle9*i* Data Guard and Oracle Transportable Tablespace to cut the downtime required for an Oracle Database 10*g* upgrade in half, to just 8 minutes.

2. Adidas High Availability and Disaster Recovery Strategy

Adidas protects against CPU failures by using Oracle Real Application Clusters (RAC) for all database services, and implementing full application service redundancy in multiple nodes for the application tier. In addition, the Adidas Data Center is compartmented into 6 fully independent cells, each completely autonomous from the other with regards to power and cabling. All hardware components are distributed across these cells in a fully redundant mode: two fully independent networks, at least two production machines, redundant disk enclosures, etc. The application components are also distributed in a redundant mode over this same infrastructure. For example, each of the two RAC instances is implemented in different LPARS residing in different cells. This design addresses all the possible single failures that could impact availability [3]. Data Guard Redo Apply is used to maintain a transactionally consistent, physical copy of each Oracle database at a remote DR site located 30km from the primary data center. Data Guard SQL Apply is also used to maintain copies of production databases for reporting purposes. The reporting databases take advantage of SQL Apply's ability to provide read access at the same time as it is applying updates received from the production database. Traditional disk replication is used for application and other components, and for any data that resides outside of the Oracle Database [4, 5]. Against this backdrop, Adidas is preparing to upgrade from Oracle9*i* to Oracle Database 10*g*. The upgrade procedure must address the following requirements:

3. Adidas upgrade requirements

• Total database downtime for planned maintenance must be less than 15minutes/quarter. Upgrading from Oracle9i to Oracle Database 10g can not exceed this 15 minute window

• One of the first databases to be upgraded supports the Adidas Electronic Ticketing Server. It has 42 table spaces and is approximately 174 GB in size.

• In the event that anything goes wrong with a database upgrade, Adidas must be able to fall back to their previous environment without exceeding the maximum allowed 15 minutes of downtime.

4. Adidas upgrade procedure

Adidas utilizes the following steps to upgrade to Oracle Database 10*g*:

Step 1: A Redo Apply standby database (physical standby) is created on a separate system running Oracle9*i.* Standard procedures are used to instantiate the standby database from a hot backup taken of the primary production database.

The primary database is configured to ship redo data to the standby using Data Guard LGWR ASYNC transport services. LGWR ASYNC ships redo data to the standby server as it is committed using an asynchronous process. The standby database receives the redo data and writes it to standby redo logs (SRLs). Then following a log switch on the primary, Data Guard archives the SRL and completes the process of applying the redo data to the standby database.

Note that the process used by Data Guard to maintain the standby copy of the production database is significantly different from that used by remote mirroring solutions such as SRDF. SRDF must replicate

every I/O to all database files, online logs, archive logs and the control file. This means that remote mirroring sends each database change at least three times to the remote site (compared to once for Data Guard), increasing network I/O significantly.

Step 2: The "shell" of an Oracle 10g database is created on the standby system using the same metadata structure as the primary database but with no data. (Fig 1.).

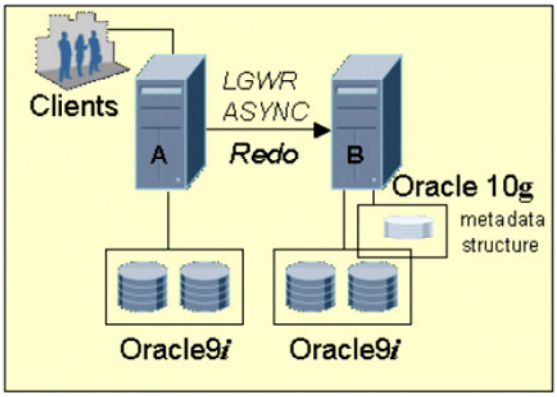

Fig 1. Instantiation of the standby database and metadata structure

Step 3: When ready to upgrade, a logs switch is done on the primary insuring that all redo in Data Guard's asynchronous buffer has been shipped to the standby database. The application is then shutdown and the primary database is closed. A standard Data Guard failover operation is performed. This completes the process of applying all of the redo received by the standby server to the standby database. The standby database is then brought up in the primary role and opened for read/write access. This enables Transportable Tablespace to write to the database. Simple commands are used to execute the failover process:

```
SQL> alter database recover managed standby database finish;
SQL> alter database commit to switchover to primary;
SQL> shutdown immediate;
SQL> startup;
```

In parallel with the above steps, the application is restarted on the "empty" Oracle 10g database.

Step 4: Export and Transportable Tablespace are used to move the tablespaces from the Oracle9i standby database to the new Oracle Database 10g shell. This is a two-step process. First set the tablespaces that are involved to be read only. Confirm that the tablespaces involved are self-contained as follows:

```
SQL> EXECUTE SYS.DBMS_TTS.TRANSPORT_SET_CHECK ('users,tools',TRUE,TRUE);
SQL> SELECT * FROM SYS.TRANSPORT_SET_VIOLATIONS;
```

This query should return no rows, identifying that there are no dependencies beyond these tablespaces. Secondly, create the transportable table set using export. Note this doesn't export the data itself just the

metadata. Also do a full export of the database but specify ROWS=n so that only the metadata is exported. This will ensure that all the objects such as packages, procedures etc that will exist in the new Oracle 10g Database.

```
% exp SYS/oracle TRANSPORT_TABLESPACE=y
TABLESPACES=(users,tools) \
TTS_FULL_CHECK=y
% exp SYSTEM/oracle FULL=y FILE=full.dmp ROWS=n
```

Step 5: Import the tablespaces into the new Oracle 10g database. Since the tablespaces are on the same system no data needs to be copied from one location to another.

The tablespaces are "plugged" into the new Oracle 10g database. Import the full import that you took earlier.

```
% imp SYS/oracle TRANSPORT_TABLESPACE=y FILE=expat.dmp
DATAFILES=('/oracle/oradata/users01.dbf','/oracle/oradata/tools01.dbf')
% imp SYS/oracle FILE=full.dmp ignore=y
```

Fig. 2 illustrates steps 3-5, requiring total downtime of 13 minutes

Step 6: The upgrade is complete. All that remains is to point users at application that is already running with the new Oracle 10g database.

Step 7: The final step in the process is to take a hot backup of the new Oracle 10g version of the production database, and instantiate a physical standby at the remote DR site.

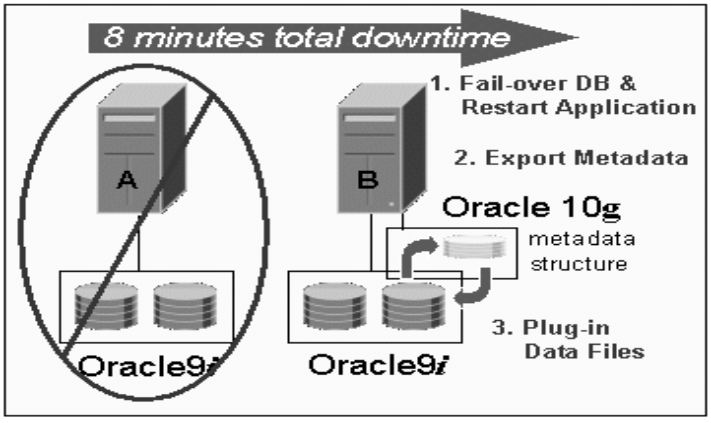

Fig 2. Export metadata to Oracle 10g, plug in data files.

4.1. Fallback scenario

Should any problems occur while the upgrade is being done which would result in a more downtime than allowed, the original Oracle9i database is started, the application is pointed at it and restarted – and the upgrade is deferred to a later time.

4.2. Transportable tablespace

The Oracle Transportable Tablespace [2] feature allows users to move a user tablespace quickly across Oracle databases. It is the most efficient way to move bulk data between databases. In the Adidas example above, the data does not move, it is simply "un-plugged" from the Oracle9i database, and "plugged" into Oracle Database 10g.

Moving data using Transportable Tablespace is much faster than performing either an export/import or unload/load of the same data. This is because transporting a tablespace only requires the copying of datafiles and integrating the tablespace structural information. Transportable Tablespace can also be used to move both table and index data, thereby avoiding the index rebuilds required when importing or loading table data as shown in Fig. 3.

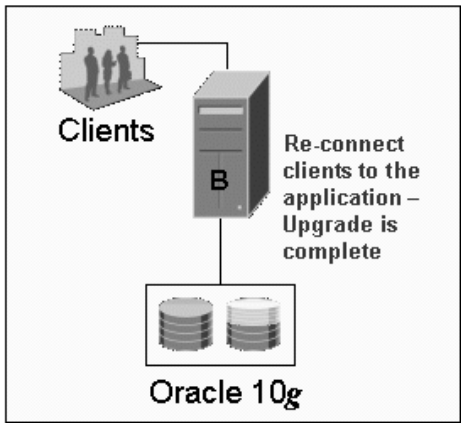

Fig 3. Upgrade complete- Production running on Oracle 10g

Adidas has used Transportable Tablespace in the example above to assist in database migration. Additional uses include:

- Exporting and importing partitions in data warehousing tables
- Publishing structured data on CDs
- Copying multiple read-only versions of a tablespace on multiple databases
- Archiving historical data
- Performing tablespace point-in-time-recovery (TSPITR)

Oracle9i Tranportable Tablespace also provided Adidas with a significant degree of flexibility. In one example, Adidas was able to transport tablespaces from Oracle single instance (non-RAC) running on

HP N-Class with HP Logical Volume Manager, to Oracle RAC running in HP Superdome with Veritas Cluster Volume Manager. However, Oracle9i still required both source and target databases to be on the same OS platform. Oracle Database 10g adds cross platform support to Transportable Tablespace, enabling users to transport tablespaces across OS platforms. This functionality can be used to:

- Provide an easier and more efficient means for content providers to publish structured data and distribute it to customers running Oracle on a different platform
- Simplify the distribution of data from a data warehouse environment to data marts which are often running on smaller systems on different platforms
- Enable the sharing of read only tablespaces across a heterogeneous cluster (the nodes must have same endianess)
- Allow a database to be migrated from one platform to another (use with Data Pump or Import/Export)

If the source platform and the target platform are of different endianness, then an additional conversion step can be automated using RMAN on either the source or target platform to convert the tablespace being transported to the target format. If they are of the same endianness, then no conversion is necessary and tablespaces can be transported as if they were on the same platform.

4.3. Data guard 10g & rolling upgrades

Data Guard10g (from release 10.1.0.3 and forward), is an out-of-the-box solution for rolling database upgrades that greatly simplifies the above process and dramatically reduce total database downtime.

The Data Guard 10g Rolling Upgrade process begins in a similar fashion using standard Data Guard procedures to instantiate a standby database, but then differs by using Data Guard SQL Apply to create a logical copy of the primary database. Once the standby has been created and Data Guard is shipping redo data from primary to the standby database, the rolling upgrade process is executed as follows:

1. Stop SQL Apply on the standby database and upgrade the Oracle database software on the standby to version “n+1”. The primary database continues to process transactions. Redo accumulates on the primary server while the upgrade process proceeds to the next step.

2. SQL Apply is restarted on the standby database, and the Data Guard configuration operates in mixed mode (Oracle version “n” on the primary, and “n+1” on the standby). Data Guard automatically resynchronizes the two databases, making the standby database current with the latest transactions from the primary database. Production then continues for a period of time in this mixed mode until the DBA staff is confident that the new Oracle release is operating as expected. Through this phase of the upgrade, there has been zero application downtime. The upgrade process then proceeds to the next step.

3. Perform a standard Data Guard switchover, reversing the role of the standby server to that of the “new” primary. This is the first and only time during the upgrade process where downtime is required. Total database downtime is no greater than the time it takes to complete the switchover process, a task that is completed in seconds when following Oracle best practices. Activate user applications and services on the new primary database. If everything behaves as expected – complete the upgrade process by proceeding to step 4. If there are problems that cannot be quickly resolved, simply open the previous primary database, switch users back and start over.

4. Everything has checked out, the new primary is running production as expected. Now it is time to complete the process and upgrade the original primary (now the new standby database). Halt SQL Apply on the new standby. Upgrade the Oracle database software to version n+1. Restart SQL Apply and allow Data Guard to automatically bring the new standby completely up to date with the new primary. As with steps 1 and 2, step 4 requires zero downtime for the production application.

5. Conclusion

Adidas has expanded on its success with Data Guard as a DR solution, and is using it together with Transportable Tablespace to reduce database downtime and achieve service level agreements as they upgrade Oracle9i databases to Oracle Database 10g. Data Guard 10g utilizes similar principles to implement rolling database upgrades that simplify the previous, "hand crafted" process with a standard, out-of-the-box solution. Most importantly, Data Guard 10g Rolling Upgrades reduce database downtime to near zero.

References

[1] Sergio Marti, T.J. Giuli, Kevin Lai, and Mary Baker. Mitigating routing misbehavior in mobile ad hoc networks. In Proceedings of MOBICOM 2000, pages 255–265, 2000

[2] Y. Hu D. Johnson, D. Maltz. The dynamic source routing protocol For mobile ad hoc networks (dsr). http://www.ietf.org/internetdrafts/ draft-ietf-manet-dsr-09.txt, April 2003.

[3] Qi He, Dapeng Wu, and Pradeep Khosla. SORI: A secure and objective reputation -based incentive scheme for ad hoc networks. In IEEE Wireless Communications and Networking Conference (WCNC 2004), Atlanta, GA, USA, March 2004.

[4] Sergio Marti, T.J. Giuli, Kevin Lai, and Mary Baker. Mitigating routing misbehavior in mobile ad hoc networks. In Proceedings of MOBICOM 2000, pages 255–265, 2000.

[5] Pietro Michiardi and Refik Molva. CORE: A collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks. Sixth IFIP conference on security communications, and multimedia (CMS 2002), Portoroz, Slovenia., 2002.

Author biographies

AUF AKHTAR has recently finished his M.S. in Computer Sciences and Engineering from University of Bridgeport. His research interest includes database design.

SYED S. RIZVI is a Ph.D. student of Computer Engineering at the University of Bridgeport. He received a B.S. in Computer Engineering from Sir Syed University of Engineering and Technology and an M.S. in Computer Engineering from Old Dominion University in 2001 and 2005 respectively. In the past, he has done research on bioinformatics projects where he investigated the use of Linux based cluster search engines for finding the desired proteins in input and outputs sequences from multiple databases. For last one year, his research focused primarily on the modeling and simulation of wide range parallel/distributed systems and the web based training applications. Syed Rizvi is the author of 15 scholarly publications in various areas. His current research focuses on the design, implementation and comparisons of algorithms in the areas of multiuser communications, multipath signals detection, multi-access interference estimation, computational complexity and combinatorial optimization of multiuser receivers, peer-to-peer networking, and reconfigurable coprocessor and FPGA based architectures.

KHALED ELLEITHY received the B.Sc. degree in computer science and automatic control from Alexandria University in 1983, the MS degree in computer networks from the same university in 1986, and the MS and Ph.D. degrees in computer science from the Center for Advanced Computer Studies at the University of Louisiana at Lafayette in 1988 and 1990, respectively. From 1983 to 1986, he was with the Computer Science Department, Alexandria University, Egypt, as a lecturer. From September 1990 to May 1995 he worked as an assistant professor at the Department of Computer Engineering, King Fahd University of Petroleum and Minerals, Dhahran, Saudi Arabia. From May 1995 to December 2000, he has worked as an Associate Professor in the same department. In January 2000, Dr. Elleithy has joined the Department of Computer Science and Engineering in University of Bridgeport as an associate professor. In May 2003 Dr. Elleithy was promoted to full professor. In March 2006, Professor Elleithy was appointed Associate Dean for Graduate Programs in the School of Engineering at the University of Bridgeport.
Dr. Elleithy published more than seventy research papers in international journals and conferences. He has research interests are in the areas of computer networks, network security, mobile communications, and formal approaches for design and verification.