FIDELITY NATIONAL FINANCIAL

FIS
FIDELITY NATIONAL INFORMATION SERVICES

ORACLE® MAXIMUM AVAILABILITY

ARCHITECTURE CASE STUDY

FIDELITY NATIONAL
INFORMATION SERVICES

Oracle Database 10g Release 2

Deploying the Oracle Maximum Availability Architecture

Automatic Storage Management
Oracle Real Application Cluster
Oracle Data Guard
Oracle Recovery Manager (RMAN)
Flashback Database
Enterprise Manager Grid Control

By Charles Kim

Maximum Availability Architecture (MAA) Implementation for Fidelity National Financial

Table of Contents

INTRODUCTION **6**

FNF OVERVIEW 6

EXECUTIVE SUMMARY **7**

BUSINESS PROBLEM AND DRIVERS FOR MIGRATION 8
RETURN ON INVESTMENT WITH NEW 10GRAC ARCHITECTURE 10

IMPLEMENTATION OVERVIEW **11**

SOFTWARE ARCHITECTURE 12
Software Stack *12*
Database & Server Specifications *12*
HARDWARE 13
IMPLEMENTATION PLAN 17
Servers *17*
Storage *18*
ASM and File System Layout *18*
Naming Conventions *19*
ALIAS for RAW DEVICE *20*
Tablespace Strategy *22*
Data Partitioning Strategy *22*
MATERIALIZED VIEW IMPLEMENTATION 25
MV and PARALLELISM STATISTICAL NUMBERS *25*
MVs (used parallelism) *26*
Indexes on MVs *26*
BACKUP/RECOVERY PLAN 26
RMAN BACKUP STATISTICS 29
Level 0 FULL DATABASE BACKUP *30*
Level 1 INCREMENTAL DATABASE BACKUP *31*
ARCHIVE LOG MAINTENANCE STRATEGY WITH DATA GUARD 32

DATA GUARD IMPLEMENTATION **34**

DATA GUARD SETUP REQUIREMENTS **36**

PASSWORD FILE 36
ENABLE FLASHBACK RECOVERY 36
PRIMARY DATABASE CONFIGURATION AT THE CHICAGO TECHNOLOGY DATA CENTER 37
PHYSICAL STANDBY CONFIGURATION AT LITTLE ROCK TECHNOLOGY DATA CENTER 38

CREATE STANDBY REDO LOGS: 39
ARCHIVE LOGGING REQUIREMENT 39
Start / Stop Redo Apply Process 40
Protection Mode Upgrade / Downgrade 40
GAP RESOLUTION 41
FAL_SERVER / CLIENT 42
OPERATIONS THAT AFFECT STANDBY DATABASE 42

MISSION CRITICAL DATA GUARD TEST CASE SCENARIOS 44

REVERT THE DATABASE BACK TO A PHYSICAL STANDBY: 47
PRIMARY RAC AND DG ARCHIVE LOG GAP DETECTION NOTIFICATION 51
EDOC SERVICE MANAGEMENT OBJECTIVES 52
EDOC PRODUCTION RECOVERY OBJECTIVES 54
GRID CONTROL IMPLEMENTATION 54
ASM MAINTENANCE 57
ASM REBALANCE ACTIVITIES 59
REBALANCE SUMMARY: 61
FUTURE PLANS OF ASM 62
AWR AND ADDM 63

LESSONS LEARNED 64

RECOMMENDATIONS FROM LESSONS LEARNED 68
CONCLUSION 70
CREATE/CONFIGURE ASM INSTANCES(WITHOUT DBCA): 73
Create Disk Group for FlashBack Area: 75
CREATE DISKGROUPS 75
ADD asm_diskgroups to init+ASM1.ora initialization file: 75
Register FIRST instance to cluster: 75
Register Other instance(s) to the cluster 76
CREATE/CONFIGURE RAC DATABASE AND INSTANCES 76
Steps to convert standalone non-RAC database to RAC database: 77
Create RAC Data Dictionary Views 78
STANDARDS introduced: 79
Assumptions: 79
CREATE TNSNAMES Entry using VIP 80

APPENDIX C – DATA GUARD STEP BY STEP COOKBOOK 84

PURPOSE OF THIS COOKBOOK 84
PREPARE THE PRIMARY DATABASE FOR DATA GUARD 84
Create a password on the Standby $ORACLE_HOME/dbs directory 84
Create a password on the PRIMARY $ORACLE_HOME/dbs directory (if not already) 84
Enable Force Logging – 84
Backup the primary database using RMAN – 84
Configure INIT.ORA Parameters on the Primary Site 86
Create a standby backup controlfile to build the Data Guard database 87
Create a copy of the init.ora file and push it out to the DR site: 88
Make necessary modifications to the spfile on the DR site: 89
Restore the standby control file using rman 90
Mount the database 90
Restore the database from the backup file system 91
Restore ARCHIVELOGS at the DR Site 92
Create TNSNAMES entries for FAL_CLIENT and FAL_SERVER 94
Enable Flashback Recovery On Primary and Standby Database 94

RAC SPECIFICs 96
Register The RAC Data Guard Database Environment *96*
Add Standby Redo Logs For Redo Log Synchronization From Primary *97*
Verify that the standby redo logs were created: *97*
START AND MONITOR DATA GUARD 99
Check on Data Guard Processes *99*
APPENDIX D- RAC SERVICE MANAGEMENT 102
APPENDIX E - INTERCONNECT INFORMATION 103
APPENDIX F – SPECIAL THANKS AND CONTRIBUTING MEMBERS 104
INDEX 106

INTRODUCTION

FNF OVERVIEW

Fidelity National Financial, Inc. (FNF), number 248 on the Fortune 500, is a leading provider of outsourced products and services, including technology solutions, title and escrow services, personal insurance, claims management services and more. FNF has standardized on Oracle RAC, Automatic Storage Management, Data Guard, Recovery Manager (RMAN), and Flashback technologies for all their Mission-Critical applications and committed a considerable investment in time, money and resources in this architecture.

The application described in this case study was migrated to Oracle Database 10g Release 2, and is the eDoc System used to house the mortgage title metadata and over 50 million PDF/XML Title Repository documents within the Oracle Database.

EXECUTIVE SUMMARY

Fidelity National Financial operations utilize a wide array of computer systems to produce title and escrow documents. Increasingly, the external marketplace and internal FNF operations require electronic availability of data and documents created and stored in the production systems. This need is driven by consumer demand for quicker, cheaper, and more streamlined real estate transactions, and by a desire to realize the internal cost savings associated with electronic processes (e.g., the elimination of manual keying efforts, the reduction of paper shipping costs, etc.).

The eDocument (eDoc) Fulfillment, designed to eliminate these duplicative efforts, functions as a single centralized clearinghouse for electronic title and escrow documents from these various production systems (i.e., TEAM, SIMON, NGS, SoftPro and Sierra). Data (both document-specific and general transaction-related data), along with document images, will be extracted from FNF's production systems and stored in a standardized format. Several methods, including both manual user interfaces (i.e., Ad Hoc queries) and automated system-to-system transactions (i.e., Push and Request/Response fulfillment), will be available for retrieving the documents and data from this central repository.

By integrating with, and retrieving content from the eDoc Fulfillment system, internal FNF systems and external customer systems will have access to data/documents from all of the major FNF title production systems in a consistent format and without additional development efforts accomplished by receiving nightly (and daily) feeds from various internal and vendor sources. This system is used across business lines for title searches, and the database maintains data used by several Fulfillment systems, most notably, Title Point, Sierra, SoftPro, Starter Insight, eDoc, NGS and Starter Search. The bulk of the data within this database consist of PDF and XML data.

The EDoc application was originally created on a DB2 database, DB2 UDB Enterprise Server Edition (ESE), running on an 8X16 AIX p590 system:

```
$ db2 connect to edocdb

   Database Connection Information

 Database server        = DB2/AIX64 8.1.4
 SQL authorization ID   = DB2EDOC0
 Local database alias   = EDOCDB
```

Over time, the EDoc system was continually running at server capacity. Instead of increasing the capacity of the database server for DB2, the decision was made to migrate to Oracle RAC. The goal of this RAC migration project was to provide an environment that

would easily scale and adapt to the growing business. FNF realized that Oracle RAC could be leveraged to provide an increase in overall CPU horsepower, higher utilization of its resources, load balancing and scalability.

FNF successfully implemented 10g RAC stack and Grid infrastructures. Additionally, Oracle Recovery Manager was deployed to implement a Backup and Recovery capability and Data Guard was utilized to provide remote disaster recovery and High Availability across a Wide Area Network. Collectively, the utilization of Oracle High Availability Features and implementation utilizing Oracle Maximum Availability Architecture best practices as enabled FNF to meet service level agreements at the lowest cost.

The project was initiated in March 2006, and the system was completed and moved into production in June 2006. The new system architecture has proven to be more reliable by eliminating single points of failure and more scalable by providing the capability to add capacity on demand.

BUSINESS PROBLEM AND DRIVERS FOR MIGRATION

The decision to deploy Oracle RAC was simple. FNF needed an infrastructure that would expand and scale as more and more legacy applications migrated to eDoc. Initially, eDoc started as a port from DB2 to Oracle RAC. FNF executives realized the database potential to provide higher scalability for the dynamically growing business and to consolidate myriads of systems that house title documents in a phased approach.

Migrating to Oracle was a natural choice considering:

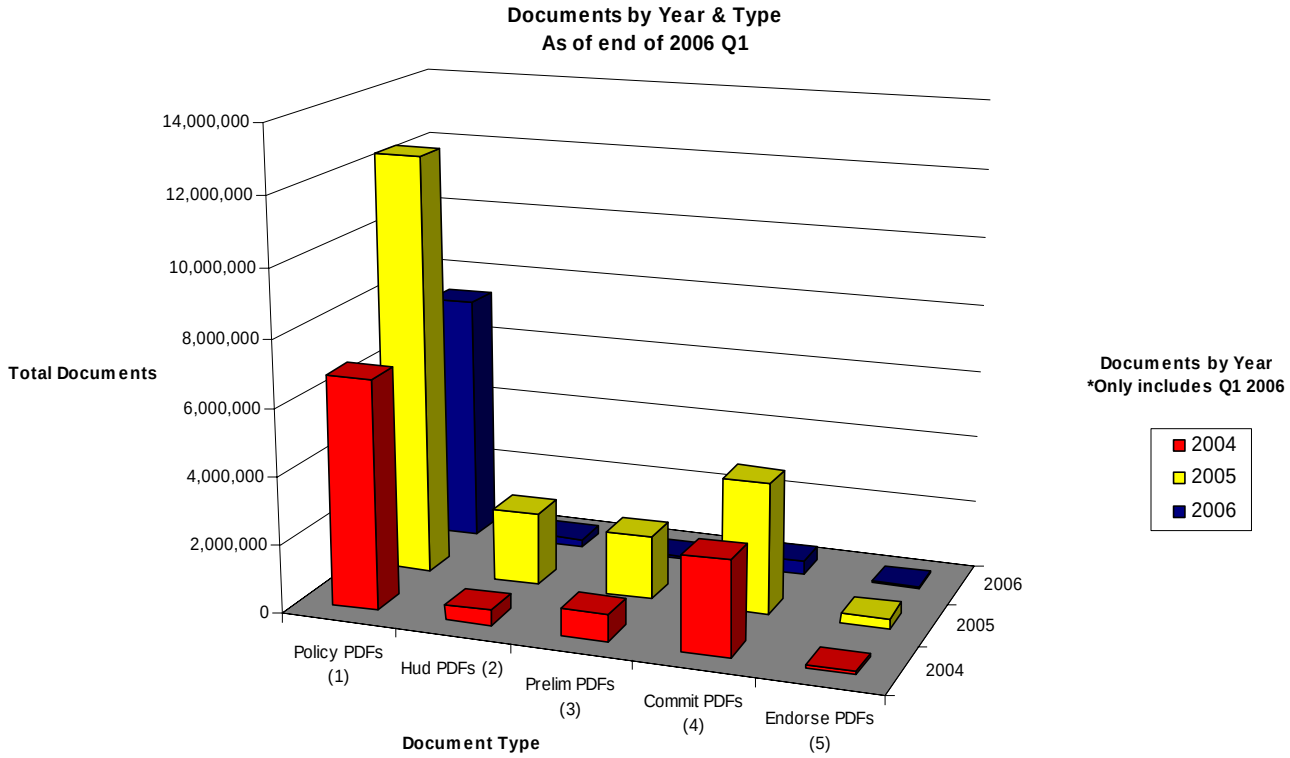
- DB2's optimistic locking model. Reads in DB2 had to be set with isolation levels that allowed uncommitted reads
- There was no partitioning strategy implemented in DB2
- Only cold/offline backup occurred every weekend in DB2
- Development community has higher technical Oracle Expertise than DB2 since FNF standardized on Oracle as the database RDBMS of choice in Q1 2005
- Backup of 50 million PDF files with more than 6 TB of data was a nightmare or almost impossible.
- EM Grid Control provides the complete monitoring and maintenance solution. By utilizing event level alerts with custom alert notification method, FNF is able to integrate Grid Control with Tivoli
- ASM provides I/O load balancing and even distribution of database extent blocks within the diskgroup
- RAC replaced HACMP as the cluster failover solution. RAC will greatly enhance server usability since idle machines will not be lying around
- Data Guard provided a disaster recovery solution over a WAN that is optimized to protect Oracle data, is integrated with RAC and ASM, and allows productive uses of standby

database resources for testing, backups, and other activities. Data Guard is included with the Oracle Database License and reduced our network, storage, and support costs relative to other solutions.

- Oracle Recovery Manager (RMAN) provided a very functional backup and recovery capability for VLDB and is included with the Oracle database license.

There was a big concern regarding the current file system backup strategy as more and more documents were migrated into eDoc. The average size of each PDF was originally estimated to be about 60k. As business requirements grew, the average size grew to 140k. In another mission critical application called Net Global Solutions, the average document increased in size to approximately 1.1.MB.

The chart below summarizes the breakdown of number of PDFs by document type as of Q1 2006.



RETURN ON INVESTMENT WITH NEW 10GRAC ARCHITECTURE

Implementation of Oracle Database10g features, including Automatic Storage Management (ASM), Real Application Clusters (RAC), and Automatic Database Diagnostic Monitor (ADDM), helped improve the mean time to deploy the application.

One key feature that Oracle allows is the ability to load and import data during business hours. In DB2, this option was not considered because of DB2's optimistic locking model. Using database services and resource manager, FNF DBAs were able to achieve production data loads during business hours without impacting production title searches.

From a business perspective several key issues were resolved:

- **Performance Improvements were achieved during the migration effort**
  In DB2, FNF QA group was never able to run eDoc wildcard searches with more than 4 users and the maximum number of regular load runner users were 40 with 90% CPU usage. In Oracle, FNF successfully ran 40 wildcard searches concurrently, and the maximum number of LoadRunner sessions was 220. This maximum load of 220 was only hit because the number of LoadRunner licenses exceeded its capacity. CPU usage across 3 nodes was about 60%.
- **Simplified Backup Architecture**
  While the database was running in DB2, all the PDF documents were stored in File systems on EMC storage. Backing up 50 million separate PDF files proved to be a backup nightmare. During the conversion to Oracle, the decision was made to store all PDFs inside the database in BLOB data types. DBAs utilized SQL*Loader to load millions of PDFs inside the database.
- **eDoc was the last database on DB2 ported to Oracle**.

  Earlier in Q1 of 2005, FNF started the process to migrate off of DB2 to Oracle as the enterprise database solution

IMPLEMENTATION OVERVIEW

FNF's current standards include IBM AIX servers and Hitachi for SAN storage, Oracle Database 10g Release 2 RAC infrastructure, Oracle Data Guard disaster recovery infrastructure, and IBM Websphere for Application Servers middleware . The original DB2 database server was an IBM AIX p590 LPAR with 16 CPUs on EMC. During the migration, eDoc hardware was re-architected to IBM p670 Servers on Hitachi TagmaStorage USP1100 SAN Storage.

FNF decided to implement Oracle RAC cluster in order to create a system that has the ability to expand by adding nodes to meet future workloads. FNF recognized the provisioning ability to pay as you grow.

FNF's Project Implementation plan included the following major phases:

1. Preparation of Oracle Database 10gR2 on RAC and ASM. AIX NIM Images were used to build consistency in the OS patch level and software components. System Administrators were able to reproduce NIM images approximately in 4 hour time frames.

   FNF built series of "cookbooks" to deliver a consistency in their RAC Builds. These cookbooks included the initial Clusterware setup to ASM installation and configuration to RAC installation.

   Contrary to Oracle's recommendation to utilize Database Configuration Assistant (DBCA), FNF's primary goal was to be able to create RAC environments that were even more easily repeatable and automated as much as possible. By scripting the majority of the RAC build process using Korn shell scripting, FNF was able to achieve RAC builds in a timely manner and were able to achieve installation consistency that conformed to DBA standards. By creating RAC environments using scripts, FNF realized the additional benefit of DBAs knowing that RAC was not a "black box" architecture. DBAs who were new to RAC easily understood RAC architecture by reviewing implementation scripts.

   Database engineers strive to achieve 100% automation. The motto of the team is to automate as much as possible and reduce redundant effort. As new standards are introduced, they are immediately incorporated into the automation process. Automation efforts include database installation to creating RAC instances to RAC service management. The future DBA technology roadmap is to fully implement silent installations using response files to achieve complete builds in command line mode.

   Instead of creating one master driver script to create the RAC environment, FNF RAC-build processes are broken down into multiple steps. Please refer to ***Appendix B: RAC Command Line Implementation*** for complete details.

2. Migration of the EDoc system from DB2 to Oracle Database 10gR2. This migration process had several components. One of such components included Informatica porting the majority of the transactional data.

3. Loading unstructured and structured files vital to EDoc from file system into the database. The DBAs took on the responsibility of loading PDFs into the database. Informatica was used to load XML data into the database.

Oracle RAC Services were deployed to allow the appropriate workload on suitable servers, thus distributing the load evenly. Initial plans were to implement separate SERVICE_NAMES with preferred instances to tune the RAC environment, but the default client and server load balancing proved to be effective. The internal service name of EDOCPRD_RAC was utilized at the time of implementation.

SOFTWARE ARCHITECTURE

There are multiple components to eDoc. The web presentation layer, which provides Title Starter Search capabilities is developed utilizing J2EE framework for IBM Websphere clustered JVM to communicate with the database via Oracle JDBC Thin 10.2 drivers. eDoc also has another component called Importer, which is a farm of PCs running java and loading XML/PDF and metadata into the database.

Software Stack

Middle-tier Server	• AIX 5.3 ML2 • Oracle JDBC 10.2 Thin Driver • 2 WebSphere 6.0.2.9 Application Servers (2 X 8) Clustered • 2 Web Servers With Load Balancing to App Servers
Importer Servers	• 8 FTP Servers running Windows 2000 to receive incoming files from other applications • 4 PC Servers running Windows 2000 to load PDF/XML into the Oracle Database. Java via JDBC Thin Drivers are utilized

Database & Server Specifications

Attribute	Description
Database Name	EDOCPRD
Primary Server Name	Ictcdb001, ictcdb002, ictcdb003
Primary domain name	.ngs.fnf.com

Attribute	Description
Database Type	Oracle RAC
Operating System	AIX 5.3 ML2
Storage Type	SAN – Hitachi managed by Oracle ASM
Server Models	IBM Power4
RAM per server	16G
CPU	4 at 1500 MHz
Failover Server Name	Iltcdb001, iltcdb002, iltcdb003
Database Version	Oracle Database 10g Enterprise Edition Release 10.2.0.2.0 - 64bit Production
Database Options	Oracle 64-bit version server
Oracle Home	/apps/oracle/product/10.2.0/RACDB
ASM Home	/apps/oracle/product/10.2.0/ASM
Clusterware Home	/apps/oracle/product/10.2.0/CRS
Default file destination	Diskgroup +DG_EDOC_PD101
Database Owner	Oracle (UNIX ID)
NLS Character Set	WE8MSWIN1252
DB Block Size	16K & 32k
Total SGA	8G per instance
Database Listener Name	EDOC_<hostname>
Database Listener Port	60010

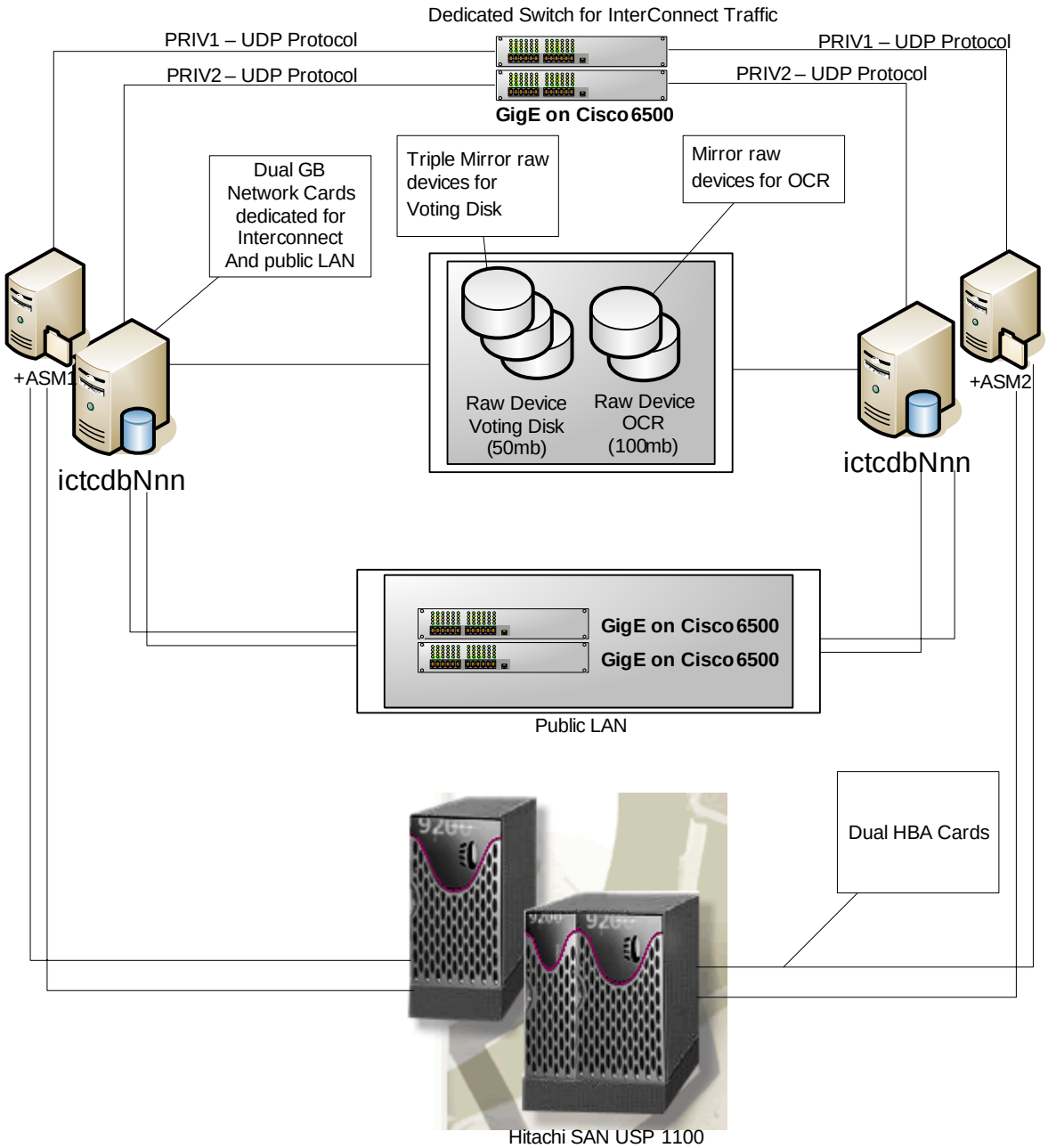
Multiple Oracle database block sizes were implemented. The intent of the 32k block size was to accommodate Large Objects Types.

HARDWARE

Database Server – IBM AIX p670	• (3) - 4 CPUs (1500 MHz), 16Gb memory • 2 NICs for Interconnect - EtherChannel (active/passive) • 2 NICs for Public Interface - EtherChannel (active/passive) • 2 HBAs
Data Guard-Standby – IBM AIX p670	• See Data Guard Section
Middle-tier Server – IBM AIX p570	• 2 nodes • AIX 5.3 ML2

	Websphere 6.0.2.9
Brocade SAN Switches	(2)
HDS Universal Storage Processor	• MPIO– multipathing • 1 RAID5 groups with 128Gb LUNs and 2 RAID 1+0 groups with 32Gb LUNs presented to OS (for ASM disks) • DG_EDOC_PD101 diskgroup contains 48 ASM disks • DG_EDOC_PF101 diskgroup contains 16 ASM disks • DG_EDOC_PD501 diskgroup contains 80 ASM disks

FNF RAC Architecture setup is illustrated below in a conceptual form:



It is critical that OCR disks are mirrored using Oracle technology. Voting disks need to be at a minimum setup with triple mirrors to ensure 51% cluster survivability. As voting disks are configured, they need to be setup with 50% + availability in mind (1, 3, 5,9,11, etc). In the event that Oracle RAC detects less than 51% availability of the voting disks, the entire cluster will fail.

Cluster of 3 nodes were interconnected with two Gigabit Ethernet NICs for cluster interconnection and redundant public network connection to service clients. TNSNAMES exclusively used Virtual IP Addresses (VIPs), and remote_listeners were implemented to ensure connectivity to surviving nodes:

```
EDOCPRD_RAC =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = ictcdb001-vip.ngs.fnf.com)(PORT = 60010))
      (ADDRESS = (PROTOCOL = TCP)(HOST = ictcdb002-vip.ngs.fnf.com)(PORT = 60010))
      (ADDRESS = (PROTOCOL = TCP)(HOST = ictcdb003-vip.ngs.fnf.com)(PORT = 60010))
      (LOAD_BALANCE = yes)

    )
    (CONNECT_DATA =
      (SERVICE_NAME = EDOCPRD)
      (FAILOVER_MODE =
        (TYPE = SELECT)
        (METHOD = BASIC)
        (RETRIES = 180)
        (DELAY = 5)
      )
    )
  )
```

FNF standardized on the following for the UDP and TCP packets:

- 256k TCP receive space / sendspace
- 256k UDP receive space / sendspace

These parameters are monitored on a regular basis in the AWR reports and adjusted accordingly.

Raw devices were used for ASM Disks. Raw device aliases were created to provide device naming persistency across all the RAC nodes.

Fast Application Notification (FAN) and Fast Connection Failover (FCF) will be phased into the project. At the time of implementation, IBM Websphere did not certify Oracle JBDC thin drivers with their clustered JVM. FNF is working on this certification with IBM. FAN and FCF will be extremely beneficial to FNF, as there are two to three different workload types with varying service level objectives.

IMPLEMENTATION PLAN

Servers

Initially all efforts were focused on preparing a QA (test) cluster. Dedicated to eDoc, FNF has three RAC environments for testing and implementation:

- QA
- Prod
- Disaster Recovery

Since FNF has myriad of other RAC systems dedicated for other applications, DBAs were able to perform additional RAC setup and troubleshooting steps on other environments.

FNF benefited greatly from having a QA and DR cluster, as this provided a proof point for all configuration changes as well as Oracle recommended patches. Once the QA cluster was setup, configured, and patched, accordingly by RacPack recommended patch-list, the fault injection (validation) phase began. This phase included various tests to validate the resiliency of the cluster as well as remainder of the technology stack. See ***Appendix A*** for a test plan.

The production cluster was configured as a three-node cluster that mimics the QA cluster, including all patches, Oracle parameters settings and AIX system settings.

Concurrently, there was also a task to setup and configure the Data Guard server and Grid Control (EM) server. Grid Control will be used to manage all aspects of the database environments including DEV, QA, Production and Data Guard systems. The initial management of eDoc environment was conducted utilizing DBConsole.

As stated above, the technology stack consisted of CRS for cluster and node management, ASM for database storage and RAC for the clustered database. This technology stack afforded FNF with a low cost scalable solution, since products such as IBM HACMP products were not required. Since no third party clustering tools or utilities were employed, raw devices were used to house the OCR, Voting disk and ASM disks. All other database related files were housed in ASM. FNF deployed IBM MPIO product for I/O multipathing. FNF used the following Metalink *Note: 293819.1 Placement of voting and OCR disk files in 10gRAC* to setup and configure their placement of OCR/Voting disk files.

Storage

The half populated HDS USP was configured with RAID5 and RAID 10 group sets depending on the type of data stored on the RAID group. On the Hitachi USP 1100, 128GB of cache was purchased with 30TB of useable space. Originally, eDoc had 8GB, 32GB, 128GB and 256GB disks in the diskgroup. These disproportionate sized disks caused an inability for FNF to utilize the full capacity of the disk groups. But, it provided a great learning experience and additional confidence with ASM as painful steps had to be taken to remedy this situation. Please refer to the "***Lessons Learned Section***" for additional details. Eventually, FNF SAN Architects standardized on consistent LUN sizes. Unfortunately, FNF adopted the LUN size standard after eDoc's go-live date: 32GB and 128GB

Each LUN is being evenly managed and mapped across all ACP pairs. This provided an even I/O distribution across disk controllers. A total of 144 LUNs were defined to the servers for Oracle 10gR2 Database RAC. IBM MPIO was used as the multipathing tool over the two HBAs configured per server.

Most ASM Best Practices documented by Oracle were applied. However, FNF created three disk groups to support their backup/recovery strategy (discussed in next section). DG_EDOC_PD101 diskgroup was used to house all the data files, set of redo logs and control files. The DG_EDOC_PF101 diskgroup was used as the Flash Recovery Area (FRA); however, FNF strategically chose to maintain only the Flashback logs and archive logs in the FRA. Finally the DG_EDOC_PF501 diskgroup held all the large objects (PDF and XML). The ASM Best Practices document can be found on the following site:

http://www.oracle.com/technology/products/database/asm/pdf/asm_bestpractices_9_7.pdf

ASM and File System Layout

The EDOCPROD database is made up of the following Disk groups within ASM:

Mount Point/ASM Diskgroup	Description
/apps/oracle	Contains the Oracle binaries, admin and OFA directories. Intentionally sized at 32GB to address three ORACLE HOMES: 1. Clusterware 2. ASM Home 3. RAC Binary Home This includes the ADMIN directories for each of the HOMES.
+DG_EDOC_PD101	ASM diskgroup for tablespaces associated with all non LOB data and index segments. RAID level is 0 + 1
+DG_EDOC_PD501	ASM diskgroup for tablespaces associated with all PDF, XML and CLOB related segments. RAID level is 5
+DG_EDOC_PF101	ASM diskgroup location for archive logs, redo logs and Flashback area. RAID level is 0 + 1.

Naming Conventions

Naming Conventions used were:

PF = P stands for Production and F stands for Flashback area

PD = Again, P stands for Production and D stands for Data

There were total of 144 disks allocated to the production ASM instance:

Name	Number of Disks	Size
DG_EDOC_PD101	48 ASM disks	1.5 TB
DG_EDOC_PF101	16 ASM disks	512 GB
DG_EDOC_PD501	80 ASM disks	10 TB

The following query provides detailed information pertaining to the disk and diskgroups associated with the eDoc production RAC database. Two disk sizes were standardized for the database depending on the tier level of storage. For all RAID 1+0, the SAN architects provided 32 GB disks. For RAID 5 storage, SAN architects provided 128 GB disks.

```
  1  select a.name DiskGroup, b.disk_number Disk#, b.name DiskName, b.total_mb, b.path
  2  from v$asm_disk b, v$asm_diskgroup a
  3  where a.group_number (+) =b.group_number
  4* order by b.group_number, b.disk_number, b.name
SQL> /

DISKGROUP       DISK# DISKNAME                 TOTAL_MB PATH
--------------- ----- -------------------- ------------ --------------------------------
                    0                           8,193 /dev/edoc_pd101_disk01
                    1                           8,193 /dev/edoc_pd101_disk02
                    2                           8,193 /dev/edoc_pd101_disk03
                    3                           8,193 /dev/edoc_pd101_disk04
***************                              ------------
sum                                             32,772

DG_EDOC_PD101       4 DG_EDOC_PD101_0004        32,788 /dev/edoc_pd101_disk05
                    5 DG_EDOC_PD101_0005        32,788 /dev/edoc_pd101_disk06
                   .. .. [ DISK 6 to 49 ]       32,788
                   50 DG_EDOC_PD101_0050        32,788 /dev/edoc_pd101_disk51
                   51 DG_EDOC_PD101_0051        32,788 /dev/edoc_pd101_disk52
***************                              ------------
sum                                          1,573,824

DG_EDOC_PD501       0 DG_EDOC_PD501_0000       131,075 /dev/edoc_pd501_disk75
                    1 DG_EDOC_PD501_0001       131,075 /dev/edoc_pd501_disk76
                    2 DG_EDOC_PD501_0002       131,075 /dev/edoc_pd501_disk77
                   .. .. [ DISK 3 to 77 ]      131,075
                   78 DG_EDOC_PD501_0078       131,075 /dev/edoc_pd501_disk93
                   79 DG_EDOC_PD501_0079       131,075 /dev/edoc_pd501_disk94
***************                              ------------
sum                                         10,486,000

DG_EDOC_PF101       0 DG_EDOC_PF101_0000        32,788 /dev/edoc_pf101_disk01
                    1 DG_EDOC_PF101_0001        32,788 /dev/edoc_pf101_disk02
                    2 DG_EDOC_PF101_0002        32,788 /dev/edoc_pf101_disk03
                   .. .. [ DISK 3 to 13 ]       32,788
                   14 DG_EDOC_PF101_0014        32,788 /dev/edoc_pf101_disk15
                   15 DG_EDOC_PF101_0015        32,788 /dev/edoc_pf101_disk16
***************                              ------------
sum                                             524,608


148 rows selected.
```

ALIAS for RAW DEVICE

FNF chose to implement aliases for raw devices to achieve device persistency. Instead of mapping ASM disks to raw device names (/dev/rhdisk50), DBAs asked System Administrators to create aliases for the raw devices using mknod (//dev/edoc_pd101_disk50).

DBAs wanted to prevent disks from having different names from one node to another. The primary objective was to protect the disk name persistency so that ASM would consistently recognize the disk names especially after a disk add or delete activity. For example, hdisk2 on node1 might show up to be hdisk3 on node2, etc.

To ensure that disk names are always consistent at the major and minor number level, SAs created device aliases via mknod on each node. For example, /dev/edocp_ocr_disk and /dev/edocp_vote_disk referencing appropriate major and minor number.

```
EDOCPRD1 > ls -ltr /dev/* |grep -i "20,  8"
crw-rw----   1 oracle   oinstall     20,  8 Mar  2 17:07 /dev/rhdisk20
brw-r-----   1 oracle   oinstall     20,  8 Mar  2 17:07 /dev/hdisk20
crw-rw----   1 oracle   oinstall     20,  8 Jul 27 07:21 /dev/edocp_vote_disk
```

SAs maintained major and minor numbers within Object Data Manager (ODM) so that it would not be reused.

AIX tools like topas and iostat reported I/O activity at the rhdisk level. DBAs, on the other hand, utilized the alias names. To effectively marry what the DBAs viewed versus what the SA and SAN Architects viewed in their tools, the DBAs created the following Alias to Raw Device Mapping script to identify the relationship based on major and minor numbers:

```
export PREFIX=$1
export MONTHS="jan|feb|mar|apr|may|jun|jul|aug|sep|oct|nov|dec"

ls -l /dev/${PREFIX}* |while read DESC NUM OWNER GROUP MAJOR MINOR MONTH DAY TIME FILE
do
  print "$MAJOR $MINOR" | egrep -i $MONTHS 1>/dev/null 2>&1
  export RC=$?

  if [ "$RC" -eq 0 ]; then
    ls -l /dev/* |grep "$MAJOR" |grep rhdisk |awk -F" " {'print $9'} |read RAW_DEVICE_FILE
    export FILE=$TIME
  else
    ls -l /dev/* |grep "$MAJOR $MINOR" |grep rhdisk |grep -v $FILE  |awk -F" " {'print $10'}
|read RAW_DEVICE_FILE
  fi
  [ "$RAW_DEVICE_FILE" = "" ] && ls -l /dev/* |grep "$MAJOR  $MINOR" |grep rhdisk |egrep -v
"$FILE|grep"  |awk -F" " {'print $10'} |
read RAW_DEVICE_FILE

  print "$FILE: $MAJOR $MINOR - $RAW_DEVICE_FILE"
done
```

Sample Output

```
/dev/edoc_pd101_disk01: 20, 2 - /dev/rhdisk22
/dev/edoc_pd101_disk02: 20, 5 - /dev/rhdisk45
/dev/edoc_pd101_disk03: 20, 1 - /dev/rhdisk10
/dev/edoc_pd101_disk04: 20, 0 - /dev/rhdisk5
/dev/edoc_pd101_disk05: 20, 3 - /dev/rhdisk30
/dev/edoc_pd101_disk06: 20, 6 - /dev/rhdisk62
/dev/edoc_pd101_disk07: 20, 11 - /dev/rhdisk8
/dev/edoc_pd101_disk08: 20, 7 - /dev/rhdisk72
/dev/edoc_pd101_disk09: 20, 12 - /dev/rhdisk10
```

Tablespace Strategy

The EDOCPROD database is not an OLTP system. All data coming into the system is received through nightly batch load processes. The PDF and XML data within the EDOCPROD database accounts for 80% of its bulk. In order to sustain optimal performance the tablespaces must be sized appropriately. The tablespace standard that EDOCPROD utilizes is as follows:

- All tablespaces that contain PDF or XML data will be sized with 32k blocks
- All tablespaces that contain data other than PDF and XML will be sized with 16k blocks
- All tablespaces that do not contain PDF or XML data will be named based on their tables' functional areas(Starter Search, Direct Delivery, Document Library, Logon Access, Importer and other processes) unless the table is large enough to be contained in its own tablespace. Example (DL_EDOC_DATA, DL_EDOC_IDX, DL_ORDER_DATA, etc).
- Tablespace naming for PDF and XML partitions will be based on the type of document, document structure and a time component. (Example: POLICY_IMG_2006_Q1)
- The maximum size a data file for a PDF or XML type table can be up to 10GB. Once this threshold is met a new data file must be created. The maximum size of a data file for any non PDF or XML type table can be large as 4GB. Once this threshold is met a new data file must be created.
- Since EDOCPRD standardized on Oracle Managed Files (OMF) normal data file naming standards do not apply. To add data files you simply issue the command :

  'alter tablespace <tablespace_name> add data file '+DG_EDOC_PD101' size 4000m;'

  - Normally, FNF's database standards do NOT allow autoextensible files. After the eDoc database grew in capacity over 125GB in a single weekend, changes were made to the database standards to allow data files to be autoextensible.
  - Data files that are allowed to auto extend MUST have a maximum size associated at either 4GB or 10GB depending on the data type that the data file is supporting.

Data Partitioning Strategy

FNF took advantage of Oracle partitioning in order to manage, maintain, archive and improve query performance for this very large database environment. The EDOCPROD database utilized the LIST and RANGE partitioning strategies. List partitioning allowed for partitions to reflect real-world groupings while LIST partitioning provided implementation strategy to improve query performance. The two tables that followed this partitioning strategy were DL_PROPERTY_ADDRESS and DL_EDOC based on the FIPS_CD as the partitioning key.

The Federal Information Processing Standards codes (FIPS codes) are a standardized set of numeric or alphabetic codes (based on county and state) issued by the National Institute of Standards and Technology (NIST) to ensure uniform identification of geographic entities through all federal government agencies.

In the Starter Search application each user's query and result set were associated with one FIPS code. The tables that utilized the range partition strategy were all the XML and IMAGE tables. Range partitioning was implemented to improve manageability, specifically from a backup perspective. These tables were partitioned based on the AUDIT_REC_CREATE_DT column which was auto populated using a before insert trigger. Details of each partitioning strategy can be found below:

- Tables that contain PDF or XML were partitioned
  - AUDIT_REC_CREATE_DT served as the partitioning key and the partitioning strategy was RANGE PARTITIONING
  - Partition naming was based on table name, type and year(PART_COM_IMG_2004)
  - The concept of 'rolling' partition was utilized for each quarter of data.
- Tables that contain a FIPS code column were partitioned
  - FIPS code column became the partitioning key and the partitioning strategy was LIST PARTITIONING
  - Partitioning naming was based on county name and state (example LOS_ANGELES_CA)
  - Partitions that have rows greater less than 300,000 were grouped together and shared a common partition(DEFLT)
  - All partitions in which the number of rows exceeds 300,000 were placed in their own partition
  - All documents that do not have a valid FIPS code were associated with a default FIPS code of UNCLN

SQL LOADER PROCESS

Since eDoc houses the majority of binary and character type large objects, this paper will describe processes used to load data from the DBA perspective. Two different technologies were leveraged to load XML and PDFs inside the database. Informatica was used to load XML metadata into the CLOB data type. The DBAs leveraged Oracle's SQL*Loader utility to load approximately 50 million PDFs into the database.

Loading 3.85 million commitment documents into the 2005 Commitment Image Partition took approximately 47 hours. Direct load option could not be leveraged because the SQL*Loader control file that was generated out of DB2 was not broken down to the partition level with millions of rows in unstructured sort order.

UNLOADING DATA out of the database

Occasionally, other systems needed mechanisms to extract data out of the eDoc. FNF database security policies forbid local oracle directory access on the database server. As a

workaround, remote directories were created on NON-Production databases, and PDFs were extracted from production database using database links to remote directories.

```
CREATE OR REPLACE procedure edoc_xtract_wk6 (
          p_CNTY_OR_OFFC_CD in varchar2,
          p_ORDER_NBR in varchar2,
          p_DOC_TYPE in varchar2,
          p_EDOC_ID in number
) is

  blob_length    integer := 0;
  out_file       UTL_FILE.FILE_TYPE;
  chunk_size     BINARY_INTEGER := 32767;
  blob_position  INTEGER := 1;

  v_blob         blob;
  v_buffer       RAW(32767);
  v_edoc_id      number := 0;
  v_name         varchar2(100) := ' ';

BEGIN

  v_name := p_CNTY_OR_OFFC_CD || '_' || p_ORDER_NBR || '_' || p_DOC_TYPE || '_' || p_EDOC_ID ||
'.pdf';

  case substr(p_DOC_TYPE,1,3)
   when 'OWN' then SELECT BLOB_CONTENT INTO v_blob FROM edoc_xtract_wk2b WHERE EDOC_ID =
p_edoc_id;
   when 'LEN' then SELECT BLOB_CONTENT INTO v_blob FROM edoc_xtract_wk2b WHERE EDOC_ID =
p_edoc_id;
   when 'PRL' then SELECT BLOB_CONTENT INTO v_blob FROM edoc_xtract_wk3b WHERE EDOC_ID =
p_edoc_id;
   when 'CMT' then SELECT BLOB_CONTENT INTO v_blob FROM edoc_xtract_wk4b WHERE EDOC_ID =
p_edoc_id;
   when 'LCP' then SELECT BLOB_CONTENT INTO v_blob FROM edoc_xtract_wk5b WHERE EDOC_ID =
p_edoc_id;
   else       null;
  end case;

  blob_length:=DBMS_LOB.GETLENGTH(v_blob);
  -- dbms_output.put_line(blob_length);

  out_file := UTL_FILE.FOPEN ('/u03/informatica/Edoc_Xtract', v_name, 'wb', chunk_size);

  if blob_length > 0 then
    WHILE blob_position <= blob_length LOOP
      IF blob_position + chunk_size - 1 > blob_length THEN
         chunk_size := blob_length - blob_position + 1;
      END IF;
      DBMS_LOB.READ(v_blob, chunk_size, blob_position, v_buffer);
      UTL_FILE.PUT_RAW(out_file, v_buffer, TRUE);
      blob_position := blob_position + chunk_size;
    END LOOP;
  end if;

  UTL_FILE.FCLOSE (out_file);

END;
```

MATERIALIZED VIEW IMPLEMENTATION

For enhanced performance, materialized views (MV) were introduced with de-normalized and aggregated data. These "Flattened" table structures were designed to become the single source for the eDoc search engine. MVs also had to be partitioned based on STATE_CD LIST PARTITION.

The MVs were refreshed with the complete option on a nightly basis after the nightly batch loads. In order to improve the performance, the MV data and index tablespaces were converted to nologging tablespaces. DBAs wanted to reduce the amount of redo traffic generated during the MV refresh processes. The major factor that went into this decision was that ALL the data was still available in the underlying normalized tables, and in the event of a catastrophe, the data could be easily rebuilt.

The DBAs architected two materialized views, one that was current image of the data and another one that was being refreshed. The DBAs did not want to incur an outage in the event that the materialized view refresh failed or was delayed for some reason. Every night a new materialized view was created. Only when a successful rebuild of the new materialized wiew is created, the local synonym associated with the view for the application user is dropped and recreated to point to the new materialized view. The swapping of the local synonym provided instantaneous access to the refreshed data.

Although this design required double storage for the materialized views, the assurance of always having the materialized view was worth the additional storage capacity. The development community realized that in the event of a failover to DR, immediate rebuild of the MVs became a requirement since they reside on nologging tablespaces. As an immediate DR failover solution, regular views can be deployed to provide access to the data realizing that the performance will be degraded for several hours until the MVs are rebuilt.

MV and PARALLELISM STATISTICAL NUMBERS

The Materialized View (DL_STARTERS_MV) is based on 4 tables:

1. Dl_edoc(50 million rows and 3.5G)
2. dl_order(47 million rows and 3.7G)
3. dl_property_address(55 million rows and 12.2G)
4. dl_loan(21 million rows and 2.2G).

The dl_property_address table also has 2 CLOBs which have been separated out of the table segment and put into their own lob segment. These CLOBs are a little over 1.2TB. The DL_STARTERS_MV itself is 7.7G and has 44 million rows in it. FNF was able to create the MV after 1 hour 45 minutes with the no parallel and no logging option. Using the default parallel setting, which were 3 slave processes on each node (9 slaves total), FNF DBAs were able to create the materialized view with partition on STATE_CD in approximately 28 minutes. Total size of the MV was 7756M. Please see below for additional statistics for MV and associated indexes.

MVs (used parallelism)

Below are some statistics indicating how the three node RAC was utilized with parallelism to create the MVs in a timely manner:

- Dl_STARTERS_MV – 10,246M, 44202658 rows, took approximately 28 minutes to build. This MV was created with LIST partitioning strategy.
- DL_PARTY_MV – 1,290M – 42582889 rows, took 34 seconds to build.
- DL_PROPERTY_TAX_APN_MV – 502M , 18442448 rows, took 1 minute 33 seconds to build.

Indexes on MVs

Each of the indexes took around 2 minutes to create. DBAs used parallel and no logging options to achieve maximum performance across all three nodes. All of the dl_starters_mv indexes were local partition indexes except for dl_starters_mv_u1 which was GLOBAL.

```
DL_PARTY_MV_N1 – 202M
DL_STARTERS_MV_N9 – 1000M
DL_STARTERS_MV_N8 – 1247M
DL_STARTERS_MV_N7 – 990M
DL_STARTERS_MV_N6 – 2068M
DL_STARTERS_MV_N5 – 1019M
DL_STARTERS_MV_N4 – 1369M
DL_STARTERS_MV_N3 – 2585M
DL_STARTERS_MV_N2 – 1192M
DL_STARTERS_MV_N1 – 843M
DL_STARTERS_MV_U1 – 952M
```

BACKUP/RECOVERY PLAN

Initially, FNF wanted to adopt a backup standard for 10gR2 databases to utilize RMAN database copy images to replace their shadow images on Hitachi. The first backup of the database would be a level 0 backup. All subsequent backups will be a level 1 backup. Before a level 1 backup each night, the level 1 backup from the previous night would be merged with the level 0 backup to instantiate a new baseline level 0 backup.

Because of the overwhelming size of the eDoc database, an independent backup architecture had to be designed to accommodate the needs of this mammoth document

repository. Compressed level 0 backup sets and incremental compressed backups became the backup solution for the eDoc system. The original design was to perform level 0 backups during the weekends and perform incremental level 1 backups during the weekdays.

> As a general internal guideline for FNF/FIS, databases less than 1 TB will utilize the database copy method for backups. Databases that are larger than 1TB will adopt the compressed backup set strategy.

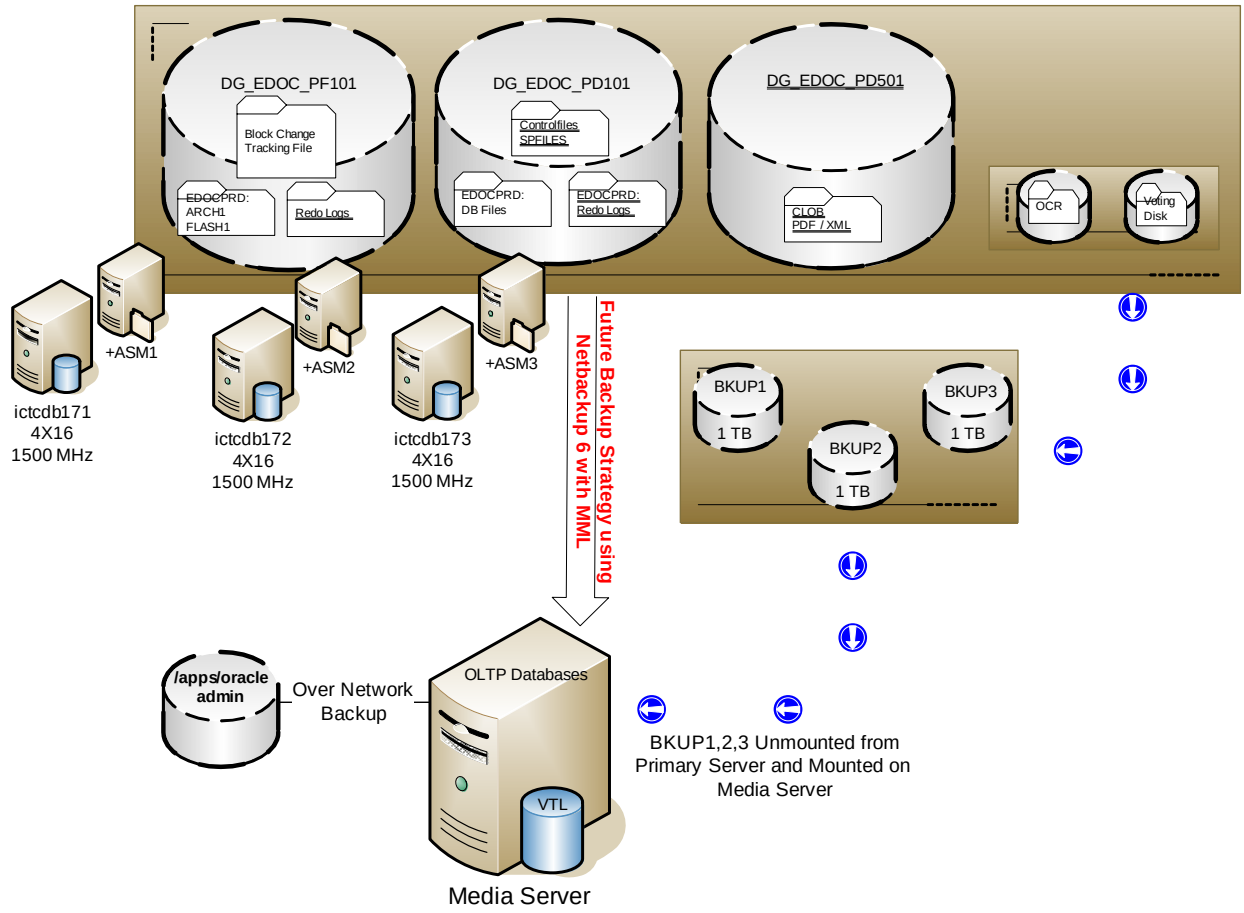
Block change tracking file (BCT) was implemented to speed up incremental backups. By enabling Oracle's block change tracking file, only changed blocked would be scanned and backed up. With eDoc's 12 TB database, the BCT file has grown to over 500MB:

```
  1* SELECT * FROM V$BLOCK_CHANGE_TRACKING
SQL> /

STATUS     FILENAME                                                                BYTES
---------- ---------------------------------------------------------------------- ----------
ENABLED    +DG_EDOC_PD101/edocprd/changetracking/rman_change_tracking.dbf     546375680
```

Strategically, all database backups were performed on RAC Node #1 to disk. Three separate 1TB file systems were mounted for online RMAN disk backups. These same file systems were also NFS mounted to the other nodes.

```
EDOCPRD1 > df -gt
Filesystem    GB blocks      Used      Free %Used Mounted on
/dev/appslv       31.88     23.71      8.16   75% /apps/oracle
/dev/bkup1lv    1023.75    453.20    570.55   45% /data/oracle/EDOCPRD/bkup1
/dev/bkup2lv    1021.75    467.44    554.31   46% /data/oracle/EDOCPRD/bkup2
/dev/bkup3lv    1023.62    438.63    585.00   43% /data/oracle/EDOCPRD/bkup3
```

These backup file systems are un-mounted and then mounted on the media server for local backups to eliminate network traffic and latency. Once mounted on the media server, the file systems were backed up to the Vault Tape Library (VTL), which in turn, was swept to tape.



The backup strategy now is to perform FULL Level 0 backups during the weekend and incremental level 1 backups during the weekdays. At the time of production deployment, the only option was to perform backups to disk since Veritas Netbackup Advanced Client 5.x was not available on AIX.

There are plans to upgrade the OS to 5.3 TL4. Immediately following the OS upgrade, the plans are to upgrade Netbackup Advanced Client to 6.x. Once the OS upgrade is complete, the three Terabytes of allocated file system for backups need to be converted as ASM disks in the flash recovery area (FRA).

RMAN backups will backup the database into the flash recovery area inside ASM. The RMAN command backup recovery area allows you to back up all files required to restore the database via RMAN from a recovery area to an sbt (tape) device. DBAs now need to adjust their scripts accordingly to perform backup the FRA to tape using commands similar to:

```
run{
allocate channel c1 device type sbt;
backup recover area;
}
```

The new version of the netbackup will allow RMAN backup of ASM flash recovery area directly to "tape" using the media management layer. The primary goal FNF's ASM backups to "tape" are to eliminate "over-the-network" backups. There are two possible initiatives to perform localized backups:

1. Add the media server as the 4th RAC node using the addnode.sh script. This will allow the media server to mount the ASM instance and perform local backups
2. Convert the 3rd instance as the media server for eDoc

RMAN BACKUP STATISTICS

The amount of time RMAN backup takes was directly correlated to the size of the current image partition. As we got closer to the end of the quarter, the backup window increased in time and so did the size of the backup-sets on disk.

FNF was able to achieve up to 80% compression of backup-sets before they implemented table level compression for CLOB data types that store XML data. Now, FNF sees on the average of 50% compression for OLTP type data.

At the end of the quarter, new partitions are created to take care of future needs. Once a quarter ends, the new partition is used. Before the old partition can be set to READ only, the following must occur:

- Delete the rows that were marked for delete near the end of the quarter to reclaim back space
- Shrink the table's space -
  - ALTER TABLE edoc_owner.dl_edoc_endorse_xml SHRINK SPACE ;
  - alter table edoc_owner.dl_edoc_policy_xml modify lob(policy_txt)(shrink space);
- Decrease the tablespace so that used space is 100%
- Mark the tablespace as READ ONLY
  - Alter tablespace pol_xml_2004 read only;
- Update statistics of newly created read only partition
- Lock statistics
- Backup new read only Tablespace
  - backup as compressed backupset incremental level 0 diskratio=0 tag EDOC_backup_Tz_17_JUL_2006 filesperset 5 tablespace PRELIM_XML_2006_Q2 ;

At the end of every quarter, the most recent partition for the PDF and XML tablespaces were made read-only. After the conversion to read-only tablespaces, these tablespaces were

backed up once with a "*Forever Retention*" period and did not participate in the Level 0 or Level 1 backup again.

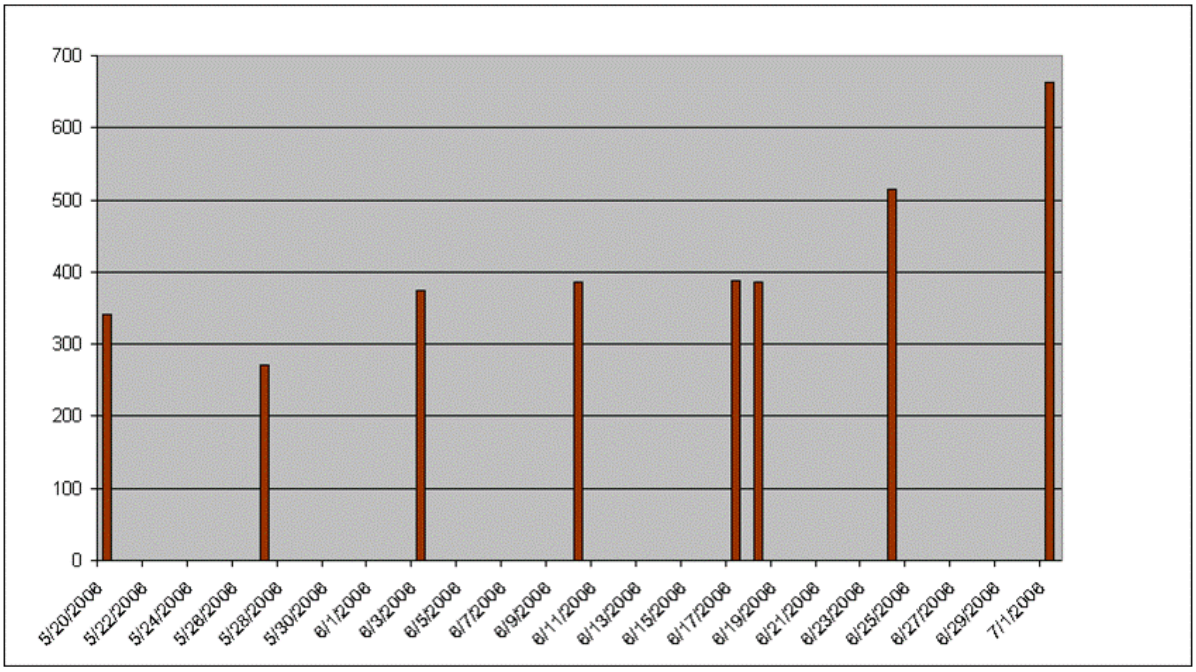
Level 0 FULL DATABASE BACKUP

```
run
{
allocate channel d1 type disk format '/apps/oracle/admin/EDOCPRD1/bkup1/%d.%s.%p.%t.DB';
set limit channel d1 kbytes = 4000000;
allocate channel d2 type disk format '/apps/oracle/admin/EDOCPRD1/bkup2/%d.%s.%p.%t.DB';
set limit channel d2 kbytes = 4000000;
allocate channel d3 type disk format '/apps/oracle/admin/EDOCPRD1/bkup3/%d.%s.%p.%t.DB';
set limit channel d3 kbytes = 4000000;

backup as compressed backupset incremental level 0 skip readonly diskratio=0 tag
EDOCPRD1_backup_0z_full_070106 filesperset 5 (database) ;
sql "alter system archive log current";
sql "alter system switch logfile";
sql "alter system switch logfile";
resync catalog;
change archivelog all validate;
sql "alter database backup controlfile to trace";
sql "alter database backup controlfile to
''/apps/oracle/admin/EDOCPRD1/log/control01_EDOCPRD1_070106.bkup'' reuse";
backup as compressed backupset diskratio=0 filesperset 5 format
'/apps/oracle/admin/EDOCPRD1/bkups/%d.%s.%p.%t.ARC' skip inaccessible (archivelog all delete
input) ;

sql "alter database backup controlfile to trace";
backup diskratio=0 tag EDOCPRD1_control_070106 format
'/apps/oracle/admin/EDOCPRD1/bkups/%d.%s.%p.%t.CTL' (current controlfile) ;
release channel d1;
release channel d2;
release channel d3;
}
```

The backup window slowly increased as the days progressed near the end of the quarter. Below are some numbers from RMAN Level 0 and Level 1 backup statistics during the months of May and June.

	Duration (Beginning of Quarter)	Duration (End of Quarter)	Size (Beginning of Quarter)	Size (End of Quarter)
Level 0	5 hrs	23 hrs	60 GB	660 GB
Level 1	15 mins	3.5 hrs	100MB	20 GB

The following diagram depicts the size of the level 0 backup-sets during the months of May, June and first day of July of 2006 reported in GB's:



Since read-only tablespaces were skipped during RMAN backups, it takes about 5 hours to perform a FULL level 0 with compression enabled at the beginning of the quarter.

Level 1 INCREMENTAL DATABASE BACKUP

The key success factor to FNF's backup solution is Oracle RMAN's ability to compress backups, skip readonly tablespaces and achieve significant performance improvements using the Block Change Tracking feature.

```
run
{
allocate channel d1 type disk format '/apps/oracle/admin/EDOCPRD1/bkup1/%d.%s.%p.%t.DB';
set limit channel d1 kbytes = 4000000;
allocate channel d2 type disk format '/apps/oracle/admin/EDOCPRD1/bkup2/%d.%s.%p.%t.DB';
set limit channel d2 kbytes = 4000000;
allocate channel d3 type disk format '/apps/oracle/admin/EDOCPRD1/bkup3/%d.%s.%p.%t.DB';
set limit channel d3 kbytes = 4000000;
```

```
backup as compressed backupset incremental level 1 skip readonly diskratio=0 tag
EDOCPRD1_backup_1z_full_071906 filesperset 5 (database) ;
sql "alter system archive log current";
sql "alter system switch logfile";
sql "alter system switch logfile";
resync catalog;
change archivelog all validate;
sql "alter database backup controlfile to trace";
sql "alter database backup controlfile to
''/apps/oracle/admin/EDOCPRD1/log/control01_EDOCPRD1_071906.bkup'' reuse";
backup as compressed backupset diskratio=0 filesperset 5 format
'/apps/oracle/admin/EDOCPRD1/bkups/%d.%s.%p.%t.ARC' skip inaccessible (archivelog all delete
input) ;

sql "alter database backup controlfile to trace";
backup diskratio=0 tag EDOCPRD1_control_071906 format
'/apps/oracle/admin/EDOCPRD1/bkups/%d.%s.%p.%t.CTL' (current controlfile) ;
release channel d1;
release channel d2;
release channel d3;
}
```

Future database roadmap may consider backups at our disaster recovery site from our physical Data Guard implementation.

ARCHIVE LOG MAINTENANCE STRATEGY WITH DATA GUARD

Archive logs are backed up routinely to disk. In the event of delayed archive shipping, the "delete input option" cannot be used. Instead, archive logs in ASM must be preserved for a minimum of 24 hours to ensure propagation of archivelogs to the DR site. Since the DR site is often utilized for performance load testing, discussed later in this paper, additional archivelogs must be preserved at the primary site.

Archivelog management became another technical challenge for FNF DBAs. To delete archivelogs except for the past two days, FNF used the delete option with the specific parameters:

```
RMAN> backup archivelog all not backed up 2 times;
Starting backup at 24-JUL-06
current log archived
allocated channel: ORA_DISK_1
channel ORA_DISK_1: sid=1076 devtype=DISK
channel ORA_DISK_1: starting archive log backupset
channel ORA_DISK_1: specifying archive log(s) in backup set
input archive log thread=1 sequence=1453 recid=1592 stamp=596629267
channel ORA_DISK_1: starting piece 1 at 24-JUL-06
channel ORA_DISK_1: finished piece 1 at 24-JUL-06
piece
handle=/u01/app/oracle/admin/DBATOOLS/flash1/DBATOOLS/backupset/2006_07_24/o1_mf_annnn_TAG2006072
4T102120_2d9scmpn_.bkp tag=TAG20060724T102120 comment=NONE
channel ORA_DISK_1: backup set complete, elapsed time: 00:00:04
Finished backup at 24-JUL-06


RMAN> delete noprompt archivelog until time 'sysdate -2' backed up 2 times to device type disk;
released channel: ORA_DISK_1
allocated channel: ORA_DISK_1
channel ORA_DISK_1: sid=1076 devtype=DISK
```

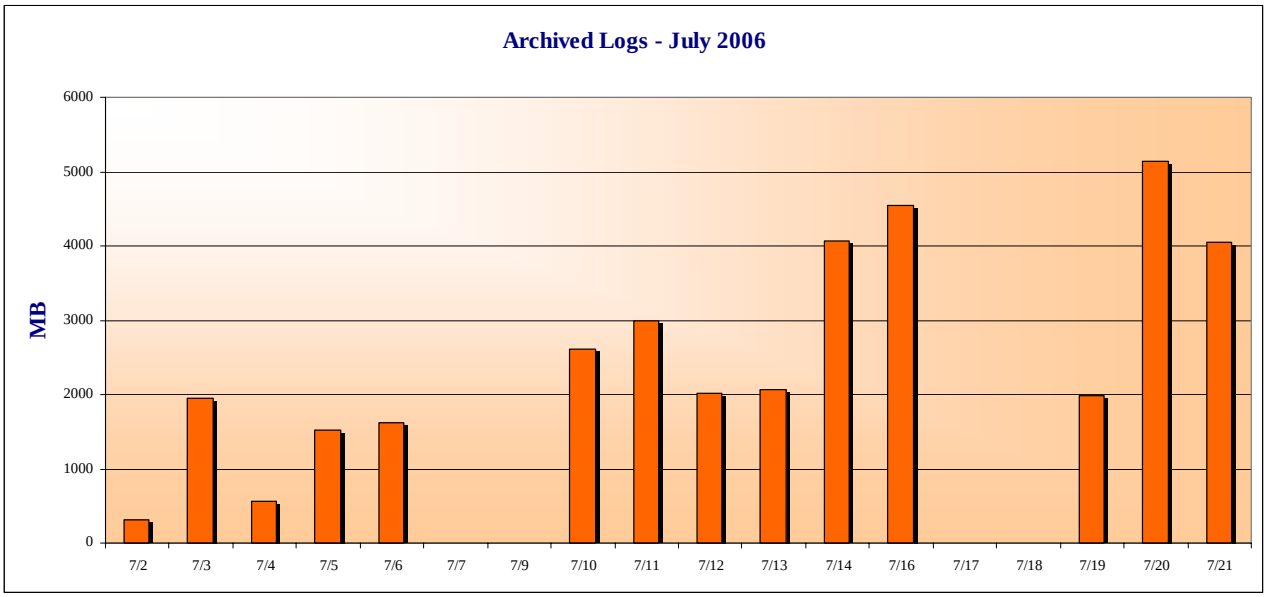
```
RMAN>
RMAN> **end-of-file**
```

Often individual archive logs had to be deleted. ASM posed another layer of complexity as archive logs are no longer manageable from OS command line utilities.

```
RMAN> delete noprompt archivelog like
'+DBA_DD501/edocpoc/archivelog/2006_06_15/thread_1_seq_14.1228.593196865%';

released channel: ORA_DISK_1
allocated channel: ORA_DISK_1
channel ORA_DISK_1: sid=1059 instance=EDOCPOC2 devtype=DISK
```

Using ASM command interactive mode (asmcmd), DBAs were also able to delete archive log files using the rm command.

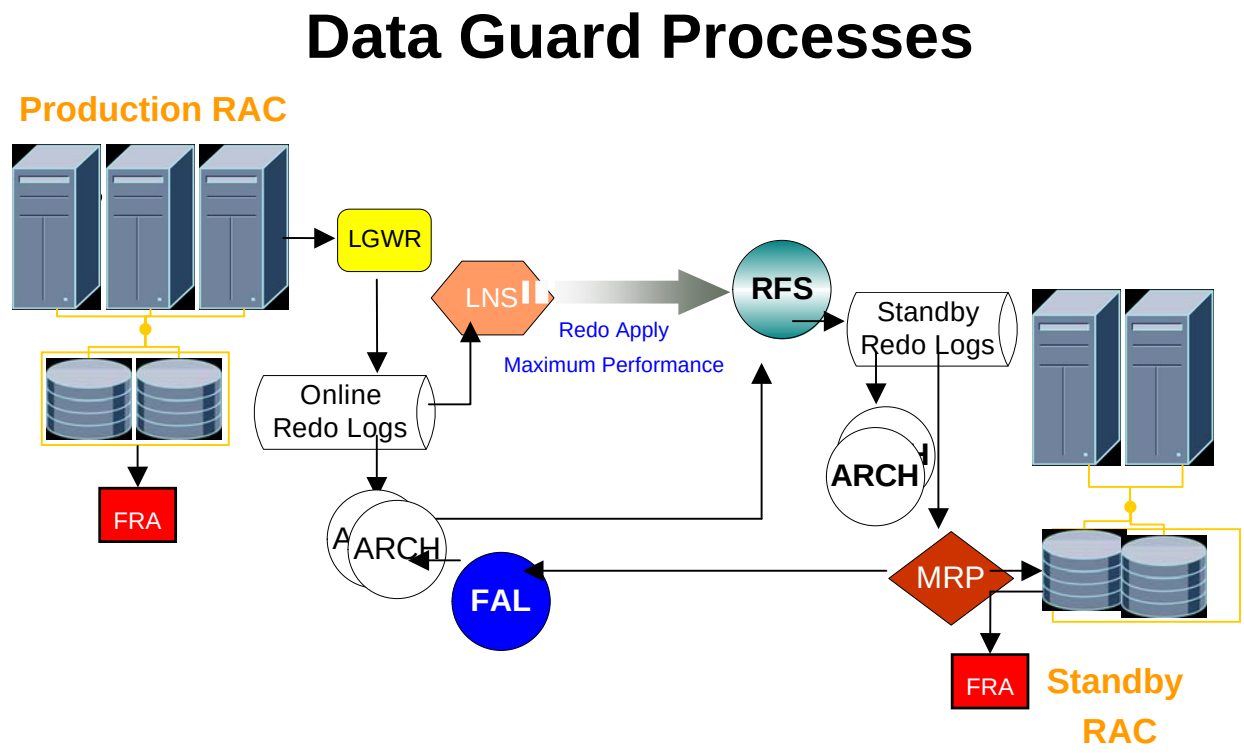
Month end processing always generated an enormous amount of archive logs. Please refer to the following chart for archivelog generation statistics:



DATA GUARD IMPLEMENTATION

FNF strategically chose Data Guard as the disaster recovery solution. FNF DBAs did not want to sacrifice performance at the primary database server while implementing Data Guard. The obvious choice was the maximum performance model (the default) to provide the greatest data protection without affecting performance of the primary database. FNF's primary redo data stream is asynchronously written to the standby redo logs at the remote DR site. Although standby redo logs are not a requirement in the maximum performance architecture, it is recommended.

The illustration below highlights the redo stream / archive log process flow from the primary data center to the DR site.



There were four primary objectives with using Data Guard to fulfill disaster recovery requirements:

1. Four hour Recovery Time Objective (RTO) and two hour Recovery Point Objective (RPO)
2. Provide an environment for stress testing and hotfix testing. This is achieved by creating a guaranteed savepoint, disable recovery, perform stress / HotFix test in the DR site, Flashback the database to the guaranteed savepoint, re-enable Data Guard
3. Perform failover and reinstate using Flashback Database Feature
4. Perform switchover and switchback to support planned outages

The server architecture differed between the primary and DR site due to budgetary considerations:

Data Guard-Primary– IBM AIX p670 (1.5GHz)	(3) – 4 CPUs, 16Gb memory 2 NICs for Interconnect - EtherChannel (active/passive) 2 NICs for Public Interface - EtherChannel (active/passive) 2 HBAs – AIX MPIO
Data Guard-Standby – IBM AIX p570 (1.6GHz)	(2) - 3CPUs, 12Gb memory 2 NICs for Interconnect - No EtherChannel 2 NICs for Public Interface – No EtherChannel 2 HBAs – AIX MPIO

At the same time, the business was aware that they would be running on a reduced capacity.

Data Guard is configured as a physical standby in the remote data center in Little Rock. Data Guard was instantiated using RMAN and resides on ASM on a two node RAC (3x12). Data Guard is configured with the "Maximum Performance" using LGWR/async transport protocol. Standby database is frequently opened as a READ ONLY database to verify the Redo Apply process.

Flashback in conjunction with Data Guard was implemented to fully utilize all the features of Oracle 10gR2 including Flashback database capabilities. Instead of dedicating separate hardware for load/stress testing, the DR Data Guard database was opened in READ WRITE mode to allow QA to perform their tasks. Once the performance testing against production data and capacity was complete, the database was flashed-back to the point prior to the QA testing start time, and Archive Logs were applied to bring the database back in sync with the primary database.

Responding to a hardware/software failure at either site, the primary and Data Guard databases can be gracefully switched and failed over. In the event of complete primary data center inaccessibility, the standby database is activated to take the primary role. (Please refer to OpenWorld Session ***S281211*** – ***Data Guard Customer Experiences: Practical Lessons, Tips and Techniques*** – for more details pertaining to FNF)

Similarly for Data Guard, FNF DBAs opted for command line interface instead of using GUI tools. DBAs regularly peeked into the database alert logs to see what commands were being executed behind the scenes in Oracle's Grid Control to "cheat" at setting up the physical standby and failover/switchover situations.

DATA GUARD SETUP REQUIREMENTS

Appendix C provides step-by-step cookbook approach to building a Data Guard environment on ASM. Similarly to the RAC setup processes, scripts are used with a configuration file to automate the build process as much as possible. If you have any questions with the details on Data Guard setup processes, please send an email inquiry to Charles.Kim@fnf.com.

There are several key Data Guard setup considerations. First, the DBAs should be cognizant of the timezone differences between the primary and DR database servers. As a recommendation, the DBAs should push to have both environments at the same timezones. Applications that are sensitive to time will be impacted by this decision.

Questions arose when FNF first started the Data Guard implementation about can the remote physical standby be non-ASM? Since ASM forces OMF naming standards, databases in Data Guard should be consistent in their configuration. Since FNF's primary data center standards implemented ASM, the target physical data guard environment also was setup using ASM.

DB_UNIQUE_NAME initialization parameter plays a key role in the data guard. The primary and Data Guard DB_UNIQUE_NAME must be unique in the data guard configuration. At FNF, the standard name for DB_UNIQUE_NAME on the DR site was ${ORACLE_SID}DR. Since DB_UNIQUE_NAME is set, DB_FILE_NAME_CONVERT and LOG_FILE_NAME_CONVERT initialization parameters are set using the ${ORACLE_SID}DR standard naming convention.

Finally, the DB_UNIQUE_NAME is also used to register the database to Oracle's cluster registry with srvctl.

PASSWORD FILE

The password file is a requirement for Data Guard. The password file should be created on the primary and physical standby database (if it doesn't already exist). Command syntax:

```
> orapwd file=[password file name] password=[sys password] entries=25
```

As a general rule, when the orapwd file is updated on the primary database, it is also pushed to the DR site using sftp or rsync.

ENABLE FLASHBACK RECOVERY

Below is a simple script to enable Flashback recovery. It provides an interactive session with the DBAs as to what needs to be done. The database must be in mounted state to enable Flash Recovery Area.

fb_setup.sql

```
prompt -- Be sure to set DB_FILE_RECOVERY_DEST_SIZE first ...
```

```
prompt ALTER SYSTEM SET db_file_recovery_dest_size = '300G' SCOPE=BOTH SID='*';
prompt -- ... and then set DB_FILE_RECOVERY_DEST and DB_FLASHBACK_RETENTION_TARGET
prompt ALTER SYSTEM SET db_file_recovery_dest = '+ DG_EDOC_PF501' SCOPE=BOTH SID=
'*';
prompt ALTER SYSTEM SET db_flashback_retention_target = 2880;
```

```
SQL> show parameter recovery_file

NAME                                 TYPE VALUE
------------------------------------ ---- ------------------------------
db_recovery_file_dest                stri +DG_EDOC_PF501
db_recovery_file_dest_size           big  300G
```

Because the sheer size and volume of transactions in the eDoc database, the flash recovery area had to be sized at 300GB. Currently, the only thing that is stored in the FRA is the Flashback logs.

Flashback should be enabled for both primary and DR databases. FNF DBAs chose to implement the Flashback database option as an alternative to setting a time delay to the application of archived redo log files. In the event of data corruption, the DR site can utilize the Flashback database option to a point in time to recover from the application of corrupted or erroneous data to the standby database.

PRIMARY DATABASE CONFIGURATION AT THE CHICAGO TECHNOLOGY DATA CENTER

Below are the key Oracle initialization parameters on the primary database server to enable Data Guard:

```
DB_NAME                      EDOCPOC
DB_UNIQUE_NAME               EDOCPOC
SERVICE_NAMES:               EDOCPOC

LOG_ARCHIVE_CONFIG                  'DG_CONFIG=(EDOCPOC, EDOCTLR)'
LOG_ARCHIVE_DEST_STATE_1             enable
LOG_ARCHIVE_DEST_1                  'LOCATION=+DBA_DD501
                                    VALID_FOR(ALL_LOGFILES, ALLROLES)
                                    DB_UNIQUE_NAME=EDOCPOC'
LOG_ARCHIVE_DEST_STATE_2             enable
LOG_ARCHIVE_DEST_2                  'SERVICE=EDOCPOC_LTC lgwr async
                                    DELAY=minutes (default 30mins)
                                    VALID_FOR(ONLINE_LOGFILES, PRIMARY_ROLE)
                                    DB_UNIQUE_NAME=EDOCTLR'

FAL_CLIENT                          EDOCPOC_CTC
FAL_SERVER                          EDOCPOC_LTC
```

```
DB_CREATE_FILE_DEST                +DBA_DD501
DB_FILE_NAME_CONVERT               '+DG_EDOC_PD501','+DBA_DD501'
LOG_FILE_NAME_CONVERT              '+DG_EDOC_PD501','+DBA_DD501'
STANDBY_FILE_MANAGEMENT             AUTO
```

The lgwr_async parameter defines that Data Guard will use asynchronous redo transport services to stream redo data to the standby location.

PHYSICAL STANDBY CONFIGURATION AT LITTLE ROCK TECHNOLOGY DATA CENTER

Below are the key Oracle initialization parameters on the failover database server that impact Data Guard:

```
DB_NAME                            EDOCPOC
DB_UNIQUE_NAME                     EDOCTLR
SERVICE_NAMES:                     EDOCTLR

LOG_ARCHIVE_CONFIG                 'DG_CONFIG=(EDOCTLR, EDOCPOC)'
LOG_ARCHIVE_DEST_STATE_1            enable
LOG_ARCHIVE_DEST_1                 'LOCATION=+DG_EDOC_PD501
                                   VALID_FOR(ALL_LOGFILES, ALLROLES)
                                   DB_UNIQUE_NAME=EDOCTLR'
LOG_ARCHIVE_DEST_STATE_2            enable
LOG_ARCHIVE_DEST_2                 'SERVICE=EDOCPOC_CTC lgwr async
                                   DELAY=minutes (default 30mins)
                                   VALID_FOR(ONLINE_LOGFILES, PRIMARY_ROLE)
                                   DB_UNIQUE_NAME=EDOCPOC'

FAL_CLIENT                         EDOCPOC_LTC
#  primary  use this net8 name to push logs to

FAL_SERVER (net8 name)                    EDOCPOC_CTC
#  fetch archived logs from primary server.

DB_CREATE_FILE_DEST                +DG_EDOC_PD501
DB_FILE_NAME_CONVERT               '+DBA_DD501', '+DG_EDOC_PD501'
LOG_FILE_NAME_CONVERT              '+DBA_DD501', '+DG_EDOC_PD501'
STANDBY_FILE_MANAGEMENT             AUTO
```

CREATE STANDBY REDO LOGS:

According to Oracle, Data Guard can recover and apply more from redo data from a standby redo log than from archived redo log files alone. With this in mind and for increased availability, FNF DBAs decided to implement standby redo logs. In addition, standby redo logs were multiplexed similarly to the way online redo logs are multiplexed.

As far as implementing standby redo logs, there were some guidelines implemented:

- When the DBA adds a redo group on primary database, the DBA must add an equivalent standby redo group. Using group numbers will simplify maintainability of standby redo log file groups.

  Do not skip log file group numbers for standby redo log file groups. This will incur additional space in the standby database control file.
- Recommended to add a redo group to a thread; thread is optional for standby database. If a thread is not assigned at creation time, Oracle will implicitly assign thread to standby.
- Ensure that log files sizes are identical on the primary and standby databases
- Determine the appropriate number of standby redo log file groups. The following equation is recommended:
  (maximum number of logfiles for each thread + 1) * maximum number of threads
- Create equivalent standby redo log file groups on the primary database that mirror the Data Guard Standby database. This will reduce the time required to switchover to the standby role and reduce the amount of time needed for manual DBA tasks.

```
SQL> alter database add standby logfile group 4 ('+DG_EDOC_PD501','+DG_EDOC_PD501')
size 150M;
Database altered.

SQL> alter database add standby logfile group 24 ('+DG_EDOC_PD501','+DG_EDOC_PD501')
size 150M;
Database altered..
```

Furthermore, DBAs need to verify that current database parameters such as MAXLOGFILES and MAXLOGMEMBERS are adequately set for creating additional standby log groups and logfiles. One of the ways to do this is by looking at the trace file created by the “alter database backup controlfile to trace” command.

ARCHIVE LOGGING REQUIREMENT

With any Data Guard implementation, forced database logging is a requirement.

```
Alter database force logging
SQL> /
```

```
Database altered.
```

If the database is already in "force logging" mode, you will see the ORA-12920 message displayed.

Although FNF DBAs implemented forced logging at the database level, nologging option was overridden for key tablespaces that performed complete refresh of data on a daily basis. Summarized materialized view data did not need to be involved in the redo transport process since all the relevant data existed in the underlying baseline tables. The rebuild of the key materialized views would have to be re-created at the DR site as a manual failover process.

Although both primary and DR Data Guard environments are multi-node RAC databases, only one node will be active to receive and apply Redo/Archive logs. Instead of having all the instances in the RAC participate in the Data Guard redo apply process, the first instance was targeted for this task. The second instance was left shutdown until a failover / switchover occurred.

Start / Stop Redo Apply Process

To start the redo apply process, enter the following command:

```
> alter database recover managed standby database disconnect from session;
```

The "DISCONNECT FROM SESSION" option causes the redo apply to run in a background session. The command to stop redo apply is:

```
> alter database recover managed standby database cancel;
```

To start real-time apply, include the USING CURRENT LOGFILE clause on the SQL statement:

```
alter database recover managed standby database using current logfile
disconnect from session;
```

If the standby database is already set for delayed application of archive logs, this can be disabled by:

```
alter database recover managed standby database nodelay;
```

The above two options will enable REAL TIME Apply with no delays to the physical standby implementation.

Transport: LWGR, ASYNC

Redo Apply: Real time Apply overwrites the delay option

Delay: No Delay

Protection Mode Upgrade / Downgrade

Situations will arise when protection modes may need to be upgraded or downgraded. Below are the requirements specified at each of the protection modes offered by Data Guard:

Max Protection:	LWGR, SYNC, AFFIRM, require standby Redo
Max Availability:	LWGR, SYNC, AFFIRM, require standby Redo
Max Performance:	LWGR, ASYNC/SYNC, AFFIRM/NOAFFIRM, not require standby Redo

ARCH, SYNC, AFFIRM/NOAFFIRM does not require standby Redo logs on the Data Guard site, though the default recommendation from Oracle is to use them. FNF does use standby Redo logs.

On primary system …shutdown and mount before

a). alter database set standby database to maximize [protection|availability|performance] ;

b). alter system set log_archive_dest_2 = 'SERVICE=EDOCPOC_LTC optional **lgwr sync affirm** VALID_FOR=(ONLINE_LOGFILES, PRIMARY_ROLE) DB_UNIQUE_NAME=EDOCPOC'

DELAY=minutes (default is 30mins);

Repeat step a). on the standby system.

Repeat step b). on the standby after a switchover.

GAP RESOLUTION

Typically archive log gap resolution occurs automatically. In rare situations, automatic gap recovery may not take place (such as when the required archive log is no longer on-disk at the primary database) and DBAs may need to perform gap recovery manually.

To manually resolve gaps:

a). Determine gap on standby db

```
select thread#, low_sequence#, high_sequence# from v$archive_gap;
```

b). Based on the output above, locate archived logs on primary:

```
select name from v$archived_log
where thread# = ##
and dest_id=1 (dest_1)
and sequence# between <low_sequence# > and <high_sequence#>;
```

c). Copy these files to standby server (rman restore if ASM)

```
RMAN> copy archivelog
'+DBA_DD501/EDOCPOC/ARCHIVELOG/2006_07_07/thread_2_seq_60.1227.595183927'
to '/tmp/dba/t2s60.dbf';

sftp /tmp/dba/t2s60.dbf oracle@iltcdb001:/tmp/dba/EDOCPOC_t2s60.arc
```

d). Register these files on standby database:

```
SQL> alter database register physical logfile '/tmp/dba/EDOCPOC_t2s60.arc';

Database altered.
```

e). Restart redo apply:

```
alter database recover managed standby database disconnect from session;
```

f). Repeat from a). until there are no more gaps:

The v$archive_gap fixed view only returns the next gap that is currently blocking Redo Apply from continuing. After resolving the gap and starting redo apply process, query v$archive_gap again on the physical standby to determine the next gap.

FAL_SERVER / CLIENT

The FAL mechanism handles the following types of archive gaps and problems:

- When creating a physical or logical standby database, the FAL mechanism can automatically retrieve any archived redo log files generated during a hot backup of the primary database.
- When there are problems with archived redo log files that have already been received on the standby database, the FAL mechanism can automatically retrieve archived redo log files to resolve any of the following situations:
    1. When the archived redo log file is deleted from disk before it is applied to the standby database.
    2. When the archived redo log file cannot be applied because of a disk corruption.
    3. When the archived redo log file is accidentally replaced by another file (for example, a text file) that is not an archived redo log file before the redo data has been applied to the standby database.
- When you have multiple physical standby databases, the FAL mechanism can automatically retrieve missing archived redo log files from another physical standby database.

The FAL client and server are configured using the FAL_CLIENT and FAL_SERVER initialization parameters that are set on the standby database.

OPERATIONS THAT AFFECT STANDBY DATABASE

Changes made to the Init.ora parameter on the primary site must manually be changed on the standby database(s)

If the DBA renames a datafile in the primary database, the DBA must manually rename the same datafiles in standby even though standby_file_management parameter is set to AUTO. Below are the steps required to rename OMF datafile in the physical standby:

1. Check for gaps in logs

```
select thread#, low_sequence#, high_sequence# from v$archive_gap;

select max(r.sequence#) last_seq_recd, max(l.sequence#) last_seq_sent
from v$archived_log r, v$log l
where r.dest_id = 2 and l.archived= 'YES';
```

2. Stop RedoApply – Stop DG

```
alter database recover managed standby database cancel;
```

3. Shutdown Immediate
4. Rman copy datafile ‘old_file’ to ‘+DG’
5. Startup mount
6. Alter database rename file ‘old_file’ to ‘new_file’;

7. Start DG

```
alter database recover managed standby database using current logfile disconnect;
```

When Adding/Dropping/Resizing Online Redo Logs on the primary database, the same corresponding action for the online and standby redo logs needs to be performed on the physical standby database.

Stop RedoApply – Stop DG

```
Alter database recover managed standby database cancel;
Alter system set standby_file_management = ‘MANUAL’;
Alter database add logfile thread # group # size 100m;
Alter database drop logfile group # ;
Alter system set standby_file_management = ‘AUTO’;
```

MISSION CRITICAL DATA GUARD TEST CASE SCENARIOS

FNF DBAs setup Data Guard test cases that were relevant to their business requirements and carefully tested each of the scenarios below

1. Open standby READ-ONLY occasionally to verify Redo Apply
2. The database can be turned over to QA for FULL performance / load testing. At this point, the database can be open for FULL Read Write mode. After QA turns the database back to the DBAs, the database can be flashed-back to the guaranteed savepoint and re-enabled for physical standby mode.
3. Switchover and switchback
4. Fail-over and use Flashback to build the failed system as standby database.
5. Flashback primary and standby databases to revert a logical error

In the examples below, there are referenced set of monitoring and maintenance scripts related to Data Guard. These scripts can be made available upon request.

1. Open standby READ-ONLY occasionally to verify Redo Apply

Need to set audit_trail = 'NONE'

While the physical database is open, redo data is still received; however, Redo Apply Services is stopped and the physical standby database is not kept synchronized with the primary database.

```
--  dg_stdby_proc.sql
set time on
set lines 132
set pagesize 9999
col client_pid format a12
select pid, process, status, client_process, client_pid,
         thread#, sequence#, block#, blocks, delay_mins
from v$managed_standby
;


       PID PROCESS   STATUS       CLIENT_P CLIENT_PID      THREAD#  SEQUENCE#     BLOCK#     BLOCKS
DELAY_MINS
---------- --------- ------------ -------- ------------ ---------- ---------- ---------- ---------- --
--------
    573880 ARCH      CLOSING      ARCH     573880                1         26       4097
5          0
    225728 ARCH      CLOSING      ARCH     225728                2         35          1
1737          0
    713110 RFS       IDLE         UNKNOWN  1274270               0          0          0
0          0
```

```
    176572 RFS       IDLE          LGWR     1081558              1         27      11283
1           0
    491816 RFS       IDLE          UNKNOWN  442398               0          0          0
0           0
    532652 RFS       IDLE          LGWR     651320               2         37         35
1           0
    213370 MRP0      WAIT_FOR_LOG N/A       N/A                  2         37          0
0           0

7 rows selected.

SQL> alter database recover managed standby database cancel;

Database altered.

SQL> alter database open;

Database altered.

SQL> @dg_db_role

DATABASE_ROLE    DB_UNIQUE_ OPEN_MODE  PROTECTION_MODE      PROTECTION_LEVEL     SWITCHOVER_STATUS
---------------- ---------- ---------- -------------------- -------------------- --------------------
PHYSICAL STANDBY EDOCTLR    READ ONLY  MAXIMUM PERFORMANCE  MAXIMUM PERFORMANCE  SESSIONS ACTIVE
```

By default, the “alter database open” command, will open the standby database in read-only mode. Alternatively, the command, “alter database open read only;” can be issued. At any time, the database can be shutdown and mounted and re-enabled for managed recovery.

2. The database can be turned over to QA for FULL performance / load testing. At this point, the database can be open for FULL Read Write mode for a specified period of time. After QA turns the database back to the DBAs, the database can be flashed back to the guaranteed savepoint and re-enabled for physical standby mode.

This is by far the biggest benefit for FNF. They are able to truly leverage the full hardware allocated for DR for multiple purposes. FNF is able to take advantage of this key feature because of Oracle 10g introduced the ability to create savepoints and also utilize the flashback recovery area.

On standby:

```
SQL> alter database recover managed standby database cancel;
```

```
Database altered.

SQL> create restore point before_lt guarantee flashback database;

Restore point created.
```

For more information about "**before_lt",** query v$restore_point

On primary database(all nodes):

```
SQL> alter system archive log current;

System altered.

SQL> alter system set log_archive_dest_state_2=defer;

System altered.
```

On standby:

```
SQL>@dg_db_role
col db_unique_name format a10
select database_role, db_unique_name, open_mode, protection_mode,
       protection_level, switchover_status
from v$database
/


DATABASE_ROLE    DB_UNIQUE_ OPEN_MODE  PROTECTION_MODE      PROTECTION_LEVEL
SWITCHOVER_STATUS
---------------- ---------- ---------- -------------------- -------------------- --------
------------
PHYSICAL STANDBY EDOCTLR    MOUNTED    UNPROTECTED          UNPROTECTED          SESSIONS
ACTIVE

SQL> alter database activate standby database;

Database altered.

SQL> @dg_db_role

DATABASE_ROLE    DB_UNIQUE_ OPEN_MODE  PROTECTION_MODE      PROTECTION_LEVEL
SWITCHOVER_STATUS
---------------- ---------- ---------- -------------------- -------------------- --------
------------
PRIMARY          EDOCTLR    MOUNTED    UNPROTECTED          UNPROTECTED          SESSIONS
ACTIVE
```

```
SQL> startup mount force;
ORACLE instance started.

Total System Global Area 2097152000 bytes
Fixed Size                  2072576 bytes
Variable Size            1040187392 bytes
Database Buffers         1040187392 bytes
Redo Buffers               14704640 bytes
Database mounted.

SQL> @dg_db_role

DATABASE_ROLE    DB_UNIQUE_ OPEN_MODE  PROTECTION_MODE      PROTECTION_LEVEL
SWITCHOVER_STATUS
---------------- ---------- ---------- -------------------- -------------------- --------
------------
PRIMARY          EDOCTLR    MOUNTED    UNPROTECTED          UNPROTECTED          TO
STANDBY

SQL> alter database set standby database to maximize performance;

Database altered.

SQL> alter system set log_archive_dest_state_2=defer;

System altered.


SQL> alter database open;

Database altered.
```

During this time when the database is in OPEN mode, archivelogs are NOT shipped to the physical standby. If space allows, DBAs should consider setting up the archive log repository to maintain protection while the standby database is in open read write mode.

REVERT THE DATABASE BACK TO A PHYSICAL STANDBY:

```
SQL> startup mount force;

SQL> flashback database to restore point before_lt;

Flashback complete.

SQL> alter database convert to physical standby;

Database altered.
```

```
SQL> startup mount force;
```

```
ORACLE instance started.

Total System Global Area 2097152000 bytes
Fixed Size                  2072576 bytes
Variable Size            1040187392 bytes
Database Buffers         1040187392 bytes
Redo Buffers               14704640 bytes
Database mounted.

SQL> alter database recover managed standby database disconnect from session;

Database altered.

SQL> @dg_db_role

DATABASE_ROLE    DB_UNIQUE_ OPEN_MODE  PROTECTION_MODE      PROTECTION_LEVEL
SWITCHOVER_STATUS
---------------- ---------- ---------- -------------------- -------------------- --------
------------
PHYSICAL STANDBY EDOCTLR    MOUNTED    MAXIMUM PERFORMANCE  MAXIMUM PERFORMANCE  SESSIONS
ACTIVE
```

On primary site (all nodes):

```
SQL> alter system set log_archive_dest_state_2=enable;

System altered.
```

3. Switch-over

The switchover process has two phases. In the first phase, the existing primary database undergoes a transition to a standby database. In the second phase, the standby database undergoes a transition to a primary database.

On primary: (shutdown all other instances except one in open mode)

```
Alter system switch logfile;
```

`Select * from v$log` – make sure that only one CURRENT log (associate the current open thread)

```
SQL> alter database backup controlfile to trace;

Database altered.
```

Converting primary to standby mode:

```
SQL> @dg_db_role

DATABASE_ROLE    DB_UNIQUE_ OPEN_MODE  PROTECTION_MODE      PROTECTION_LEVEL
---------------- ---------- ---------- -------------------- --------------------
SWITCHOVER_STATUS
--------------------
PRIMARY          EDOCPOC    READ WRITE MAXIMUM PERFORMANCE  MAXIMUM PERFORMANCE
SESSIONS ACTIVE


SQL> Alter database commit to switchover to physical standby with session shutdown;

Database altered.

SQL> shutdown immediate;
SQL> startup mount
```

Converting standby to primary mode:

On standby: (shutdown all other instances except one)

Remove any delay (Gaps)

In mount and MRPO is running

```
SQL> @dg_stdby_proc

       PID PROCESS   STATUS       CLIENT_P CLIENT_PID      THREAD#  SEQUENCE#     BLOCK#
BLOCKS DELAY_MINS
---------- --------- ------------ -------- ------------ ---------- ---------- ----------
---------- ----------
    176538 ARCH      CLOSING      ARCH     176538                1         30          1
10          0
    348358 ARCH      CLOSING      ARCH     348358                1         29          1
439          0
    389258 MRP0      WAIT_FOR_LOG N/A      N/A                   2         60          0
0          0

SQL> @dg_stop

Database altered.

SQL> @dg_db_role

DATABASE_ROLE    DB_UNIQUE_ OPEN_MODE  PROTECTION_MODE      PROTECTION_LEVEL
SWITCHOVER_STATUS
```

```
--------------- ---------- ---------- ------------------- ------------------- --------
------------
PHYSICAL STANDBY EDOCTLR    MOUNTED    MAXIMUM PERFORMANCE  MAXIMUM PERFORMANCE  SESSIONS
ACTIVE

SQL> alter database commit to switchover to primary;
SQL> alter database open;
SQL> shutdown immediate;
SQL> startup;
SQL> alter system set log log_archive_dest_state_2=enable;
```

4. Fail-over and use Flashback to build the failed system as standby database.

On standby Identify and Resolve any gaps and restart the database

Data Guard should resolve the Gap Automatically. If a situation arises to resolve the gap manually, refer to the GAP RESOLUTION SECTION above

```
@dg_gap
@dg_missing_arc
Copy archived logs with sequences between low_sequence# and high_sequence# for each
thread.
Alter database register physical logfile 'filename.arc';
Repeat until @dg_gap returns no rows
.
alter database recover managed standby database finish force;
alter database commit to switchover to primary;

shutdown immediate;
Backup the new primary database
startup;
@dg_dest2 (defer)
```

Reinstate the failed primary database as a standby:

On the new primary select to_char(standby_became_primary_scn) from v$database;

On the failed primary:

```
Shutdown immediate;
Startup mount;
Flashback database to scn <standby_became_primary_scn>;
Alter database convert to physical standby;
Shutdown immediate;
Startup mount;
@dg_start
```

5. Flashback primary and standby databases to revert a logical error

On primary:

Making sure standby caught up (dg_gap) prior Flashback primary database

Flashback doesn't handle media recovery thus DBAs may have to perform recovery manually during Flashback (this is not a Data Guard function)

```
Flashback database to scn or
Flashback database to timestamp to_timestamp('07-JUL-06 13:30:00', 'DD-MON-RR
HH24:MI:SS');
Open resetlogs;
Select to_char(resetlogs_change# - 2) from v$database;
```

On standby:

```
Select to_char(current_scn) from v$database;
If current_scn > <resetlogs-2_scn> then flashback standby database to scn <resetlogs-
2_scn>;
@dg_start;
```

Both standby and primary databases were configured to use the same RMAN catalog for backup/recovery needs so that the same backup policy was applied when the database role switched. The RMANPROD catalog database was also Data Guarded to the DR site to protect an up-to-date image of the RMAN Catalog.

FNF used both in-house scripts to monitor Redo transport and Redo Apply services. During the duration when the DR database was opened for QA testing activities, archive logs are manually shipped to the DR site using custom scripts to ensure up to date availability of production data.

PRIMARY RAC AND DG ARCHIVE LOG GAP DETECTION NOTIFICATION

Archive log gap detection has become a point of contention during peak system loads and document import activities. DBAs wrote custom monitoring jobs to compare the primary and target databases archive log statuses. This shell script is dynamic and the gap sequence can be modified according to a designated threshold.

```
0,10,20,30,40,50 * * * * /apps/oracle/general/sh/dg_archivelog_monitor.ksh EDOCPRD_RAC
EDOCPRD_JTC 10 > /tmp/dg_archivelog_monitor.EDOCPRD.log 2>&1
```

In this particular example, the DBAs are notified when the archive sequence from the primary database and data guard database exceeds 10 archive logs. Below is a snippet of code to perform the gap threshold detection:

```
print "
set head on pages 66 feed on ver off lines 2000 trims on
set pagesize 9999
set lines 200
set num 12
col name format a78
col t# format 99
col s# format 999999
col applied format a4
select name, creator, thread# t#, sequence# s#, applied, first_change#, next_change#
from v\$archived_log
where standby_dest = 'NO'
and creator <> 'SRMN'
and completion_time > sysdate - 1
order by 3,4;
" |sqlplus -s sys/$(cat $SH/.sys.${DG_SID}.pw)@${DG_SID} as sysdba > $REPORT_DR_STATUS

print "
set head off pages 0 feed off ver off lines 2000 trims on
select 'thread'||thread#, rtrim(ltrim(max(sequence#)))
from v\$archived_log
where applied = 'YES'
group by thread#
order by 1;
" |sqlplus -s sys/$(cat $SH/.sys.${DG_SID}.pw)@${DG_SID} as sysdba > $DG_ARCHIVELOG
THREADS_DG=$(wc -l $DG_ARCHIVELOG |awk {'print $1'})
```

The log file generated by this monitoring script has detailed information about what the last archive that was applied on the primary server compared to the last applied archive log on the standby database server. Because the script is connecting to a mounted database on the standby, the SYS password is required for the script to work since the DBAs are deploying this monitoring script from a centralized monitoring server.

eDOC Service Management Objectives

Originally scoped out to be a nightly batch process, the data load importer process slowly started to run into the business hours.

FNF DBAs had to react immediately to the performance implications of the importer process running into the business hours. DBAs quickly setup multiple service names using the srvctl utility:

```
srvctl add service -d EDOCPRD -s EDOCPRD_IMPORTER -r EDOCPRD3 -a
EDOCPRD1,EDOCPRD2
```

Notes:

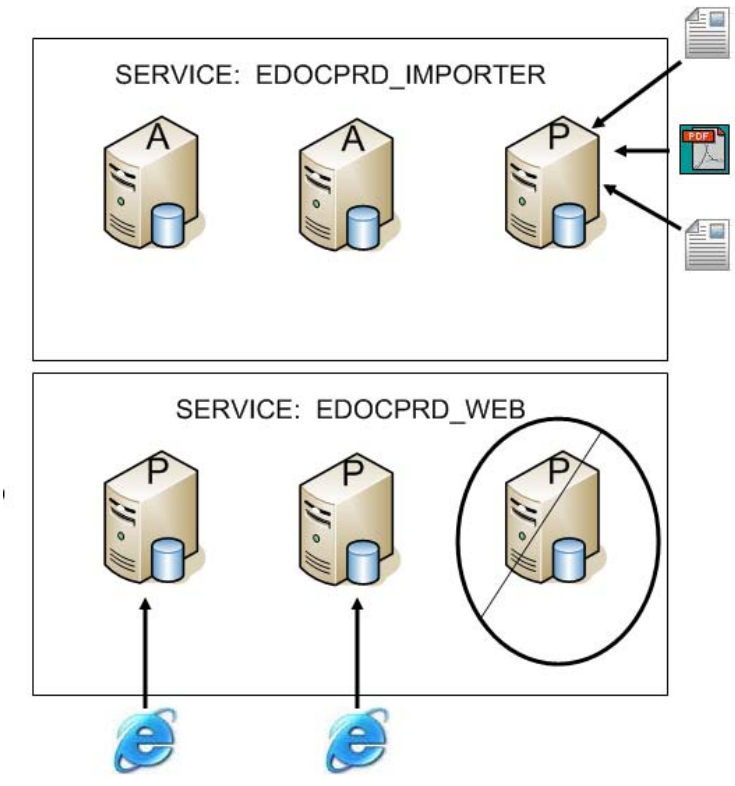
The –r option denotes preferred instance.

The –a option denotes available instance(s).

Two immediate service names were considered:

1. EDOCPRD_WEB for web search engines
2. EDOCPRD_IMPORTER for the daily data load processes

By separating the two services names to different RAC instances, FNF was able to achieve - additional workload management. If the data load process had to be run in the business hours, the third instance was dedicated for the EDOC_IMPORTER process. The first and second instances were available for the search processes without impacting performance. The service associated with EDOCPRD_WEB would be shutdown against the third instance.



After the load is complete, eDoc DBAs restart the service on instance #3 to re-establish a balanced load across all nodes. Future plans are to use Resource Manager to throttle CPU Utilization for Service Names and be able to utilize all three RAC instances accordingly.

In terms of service management, DBCA will perform the following tasks automatically:

- Add the service
- Start the service
- Add service to init.ora service_names parameter
- Create associated TNSNAMES entry

From the command line, everything can be done manually. This is one of the few exceptions where the GUI tool is recommended over command line because srvctl will allow incorrect configuration while the DBCA GUI will not. For example, srvctl will allow an instance to be listed as the preferred instance and an available instance. This caused DBAs some

headache while going through the learning curve. When the DBAs attempted to shutdown the service name on a specified instance, srvctl utility shutdown the instance instead.

Please refer to Appendix C for complete details on service management with srvctl.

eDOC PRODUCTION RECOVERY OBJECTIVES

- To minimize the four-hour outage window, a database copy can be created locally. This is where RMAN is physically creating an image copy of eDOC database and nightly merging incremental changes to the copy. In the event of failure, switch and start eDOC Database from the copy. Oracle is using incremental backups and archived logs to roll database forward. Enough archived logs need to be retained to meet the two-hour data loss requirement. FNF is not utilizing this technology yet; it is on their radar for future implementation. This will allow them to have localized business continuity similar to what an EMC BCV or Hitachi SI would provide.

- To respond to a natural disaster, a remote physical standby database is utilized. Oracle Data Guard replicates changes from Chicago data center to Little Rock data center. Changes can be applied at a configurable interval. In the event Chicago becomes inaccessible, FNF can switch to the remote standby and start the eDOC database in Little Rock. Oracle utilizes archive logs to roll database forward. Again, enough archived logs need to be retained to meet two-hour data loss requirement. In most circumstances (except for when the standby database is open read-write for testing, or following extended network outages), standby apply is current with redo received from the primary database by using Data Guard Real-Time Apply. This means there is no backlog of archive logs to recover, and switchover or failover operations can be executed immediately.

- To reverse most unwanted database changes, Oracle Flashback database is used. Oracle will use Flashback logs to rewind the database back to a point-in-time. How far back eDOC can be re-winded (Flashback window) depends on database workload and Flashback logs availability. The Flashback window is currently set to one day for eDOC. It's anticipated that 256GB is adequate to retain enough logs.

GRID CONTROL IMPLEMENTATION

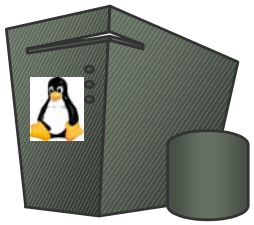
The Enterprise Manager Grid Control system was configured to run on a Linux Redhat® 4 Update 4 server. Linux Redhat® server contains the Grid Control software, while the repository database resided on the AIX database server.

At the time of production rollout, Grid Control servers were not available and thus localized DBConsole had to be implemented to provide monitoring and administration. In addition, FNF standardized to implement tools such as AWR, ADDM, Quest's Spotlight and Quest Central for database performance monitoring, but Enterprise Manager Grid Control will slowly replace these toolsets. With RAC, DBConsole deployment posed some challenges

during the initial setup. Below is the emca script utilized to create eDoc's DBConsole and repository:

```
emca -config dbcontrol db -repos create -cluster -silent \
     -HOST ictcdb003.ngs.fnf.com \
     -PORT 60010 \
     -ORACLE_HOME /apps/oracle/product/10.2.0/RACDB \
     -DB_UNIQUE_NAME EDOCPRD \
     -SERVICE_NAME EDOCPRD_RAC \
     -DBSNMP_PWD dbsnmp_password \
     -SYSMAN_PWD sysman_password \
     -SYS_PWD sys_password  \
     -ASM_USER_PWD asm_password  \
     -ASM_OH /apps/oracle/product/10.2.0/ASM \
     -EM_NODE ictcdb003 \
     -EM_SID_LIST EDOCPRD3,EDOCPRD2,EDOCPRD1 \
     -CLUSTER_NAME crs \
     -LOG_FILE $SH/EMConfig_EDOCPRD.log;
```

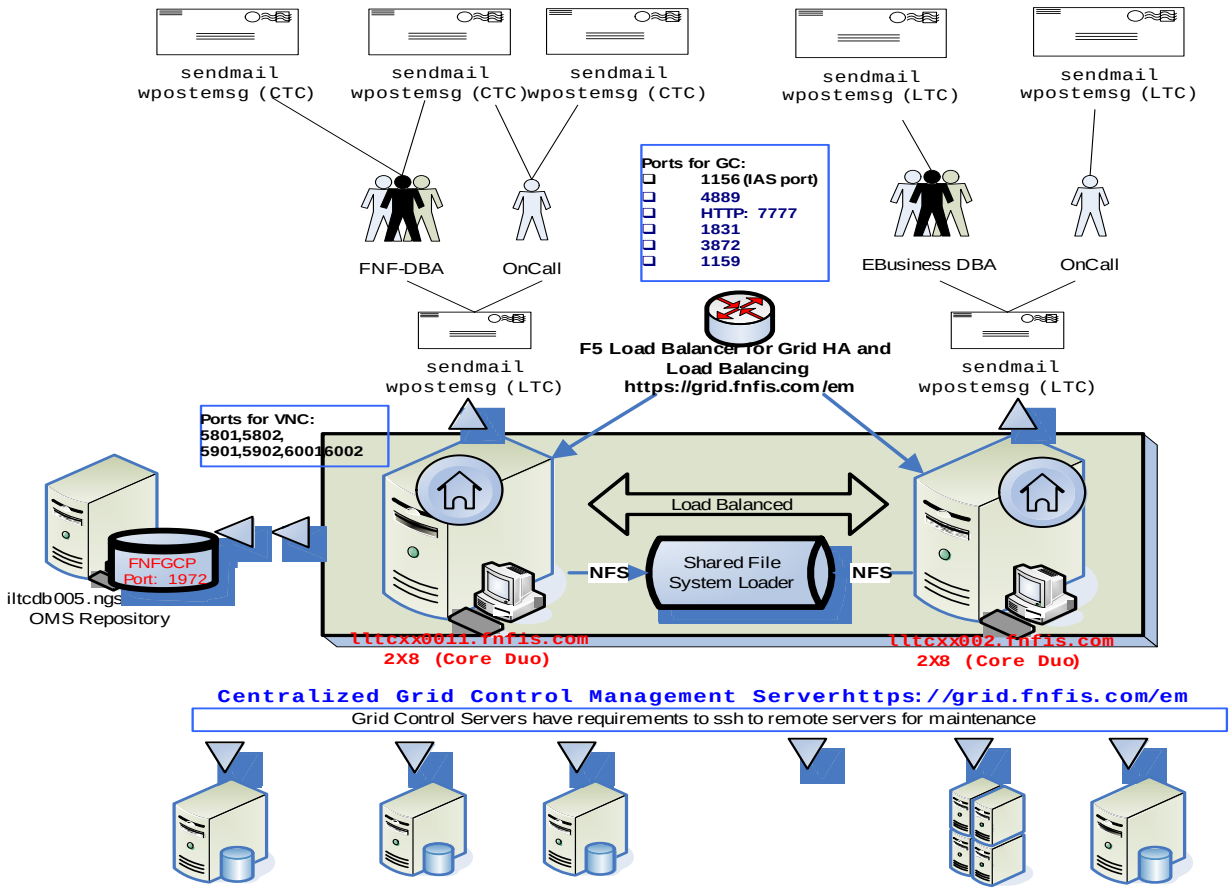
FNF's deployment of DBConsole was replaced by 10gR2 Grid Control. FNF tried to implement 10gR1 Grid Control on AIX but ran into several obstacles. Strategically, FNF DBAs decided to abandon the AIX implementation of Grid Control and focus on a Linux implementation on Redhat.



Initially, the architecture was to separate the Grid Control management of QA and development environments from Production RAC environments. Two HP Proliant servers were purchased in the primary data center for this purpose. After initial deployment in the DEV/QA environment, the DEV and QA Grid Control Servers were merged with the production Grid Control Server for High Availability considerations.

In addition, a Grid Control server was configured in Jacksonville to manage Disaster Recovery environments. Each of the Grid Control servers is 2X8 HP Proliant® DL385 servers with dual core functionality.

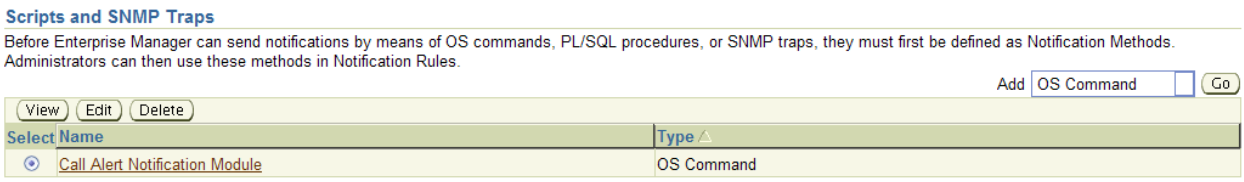
The diagram below depicts FNF's Grid Control architecture integrated with Tivoli at the primary data center for high availability considerations:



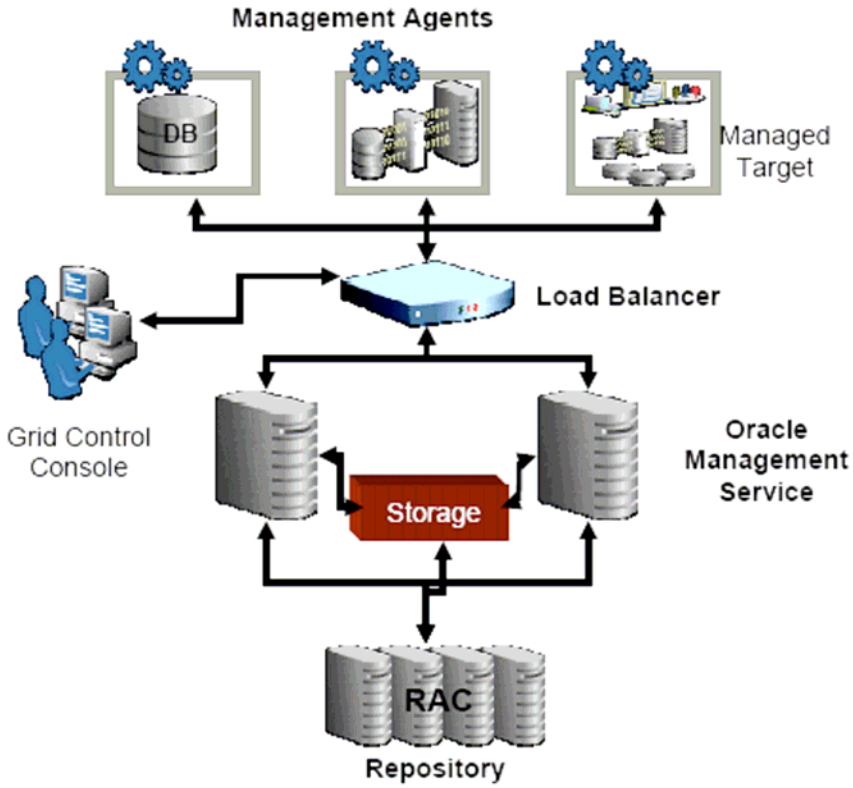
The Grid Control implementation realized the need for Network Appliance (NetApps) hardware as the shared file system for the agent uploaded XML files. F5 Big IP load balancers were also leveraged for load balance connections and to provide high availability in the event one of the Grid Control OMS servers crashed. F5 load balancer also fulfilled a security concern of accessing the web pages in a non-secure protocol since the SSL encryption occurred at the hardware level via: https://grid.fnfis.com/em.

When DBAs use the above mentioned URL, they will be redirected to the load balanced initial login page.

To meet the data center alert notification requirements, events within Grid Control had to be enhanced to make shell script callouts that send corresponding Tivoli calls to the surveillance teams in the “Setup - Notification Methods” page.



The goal of the DBA team will be to implement the EM Maximum Availability Architecture (EM MAA) Best Practices similar to this diagram:



This single environment will monitor, manage and provide diagnostic for all the database servers in the Little Rock Technology Center from DEV all the way to the production RAC environments. This will also include monitoring both 9i and 10g database versions.

The only part that FNF has not planned to incorporate into the overall EM MAA architecture is the Data Guard Component. The DBAs have future plans to re-visit this architecture.

The grid server in the Jacksonville Technology Center (JTC) will serve as our DR monitoring and DEV/QA activity that will eventually reside at in JTC.

The production Grid Control Production database, FNFGCP, will also be migrated to a two node RAC environment to address the HA requirement dictated for a centralized monitoring, monitoring and diagnostic environment. This server will also be moved from AIX 5.2 ML9 to Redhat Linux 4 Update 4 operating system.

ASM MAINTENANCE

FNF DBAs were not able to access approximately 1.1 TB of disk space within their ASM instance. DBAs quickly realized that the imbalance of disk sizes presented to the ASM diskgroup contributed to this issue. FNF incurred a significant ASM maintenance window to rectify this situation.

The amount of time to rebalance the ASM diskgroup was directly proportionate to the power limit and amount of disk space involved with the activity. With smaller 8GB disks, the

amount of time to drop and rebalance the disks seemed trivial. In one diskgroup, where only the 32GB and 8GB disks were mixed, the loss of unusable space was not as apparent because the total size of 8GB disks accumulated to only 32GB. In the following examples, these 8GB disks are dropped and rebalanced. The following section details the activity FNF DBAs incurred to remedy the situation.

```
1  select disk_number, name, total_mb, free_mb, path, header_status
2  from v$asm_disk
3  where group_number = 1
4* order by 1,2
```

```
DISK_NUMBER NAME       TOTAL_MB    FREE_MB PATH                           HEADER_STATU
----------- ------------------------------ ---------- ---------- --------------------------------
          0 DG_EDOC_PD101_0000                  32788      17008 /dev/edoc_pd101_disk01
MEMBER
          1 DG_EDOC_PD101_0001                  32788      17024 /dev/edoc_pd101_disk02
MEMBER
          2 DG_EDOC_PD101_0002                  32788      17025 /dev/edoc_pd101_disk03
MEMBER
          3 DG_EDOC_PD101_0003                  32788      17028 /dev/edoc_pd101_disk04
MEMBER
         45 DG_EDOC_PD101_0045                  32788      17026 /dev/edoc_pd101_disk46
MEMBER
         46 DG_EDOC_PD101_0046                  32788      17029 /dev/edoc_pd101_disk47
MEMBER
         47 DG_EDOC_PD101_0047                  32788      17029 /dev/edoc_pd101_disk48
MEMBER
         48 DG_EDOC_PD101_0048                   8193       4148 /dev/edoc_pd101_disk49
MEMBER
         49 DG_EDOC_PD101_0049                   8193       4148 /dev/edoc_pd101_disk50
MEMBER
         50 DG_EDOC_PD101_0050                   8193       4148 /dev/edoc_pd101_disk51
MEMBER
         51 DG_EDOC_PD101_0051                   8193       4148 /dev/edoc_pd101_disk52
MEMBER
52 rows selected.
```

It took about five minutes with five slave processes to drop an 8GB disk.

```
SQL> alter diskgroup DG_EDOC_PD101 drop disk 'DG_EDOC_PD101_0051' rebalance power 5 wait;


Diskgroup altered.
```

The same process took about three minutes with eleven slave processes.

```
SQL> alter diskgroup DG_EDOC_PD101 drop disk 'DG_EDOC_PD101_0050' rebalance power 11 wait;


Diskgroup altered.
```

It also took about three minutes to drop both disks with eleven slave processes

```
SQL> alter diskgroup DG_EDOC_PD101 drop disk 'DG_EDOC_PD101_0049', 'DG_EDOC_PD101_0048' rebalance
power 11 wait;
```

```
Diskgroup altered.
```

Four 8GB disks were removed from DG_EDOC_PD101 and free_mb went down almost evenly across remaining disks.

```
  1  select disk_number, name, total_mb, free_mb, path, header_status
  2  from v$asm_disk
  3  where group_number = 1
  4* order by 1,2

DISK_NUMBER NAME                           TOTAL_MB    FREE_MB PATH
HEADER_STATU
----------- ------------------------------ ---------- ---------- --------------------------------
------------
          0 DG_EDOC_PD101_0000                32788      16688 /dev/edoc_pd101_disk01
MEMBER
          1 DG_EDOC_PD101_0001                32788      16688 /dev/edoc_pd101_disk02
MEMBER
          2 DG_EDOC_PD101_0002                32788      16689 /dev/edoc_pd101_disk03
MEMBER
          3 DG_EDOC_PD101_0003                32788      16689 /dev/edoc_pd101_disk04
MEMBER
         41 DG_EDOC_PD101_0041                32788      16689 /dev/edoc_pd101_disk42
MEMBER
         42 DG_EDOC_PD101_0042                32788      16687 /dev/edoc_pd101_disk43
MEMBER
         43 DG_EDOC_PD101_0043                32788      16688 /dev/edoc_pd101_disk44
MEMBER
         44 DG_EDOC_PD101_0044                32788      16689 /dev/edoc_pd101_disk45
MEMBER
         45 DG_EDOC_PD101_0045                32788      16689 /dev/edoc_pd101_disk46
MEMBER
         46 DG_EDOC_PD101_0046                32788      16689 /dev/edoc_pd101_disk47
MEMBER
         47 DG_EDOC_PD101_0047                32788      16688 /dev/edoc_pd101_disk48
MEMBER

48 rows selected.
```

ASM REBALANCE ACTIVITIES

The different mix of LUN sizes on the PD501 diskgroup (128GB and 256GB sizes) contributed to approximately 1.1 TB of unusable disk space in the ASM diskgroup. A major outage had to be incurred to remove the 256GB disks and add corresponding 128GB disks.

During the process, only the first ASM instance was open to perform the “drop and add” disk activities (the other ASM instances were shutdown). First, FNF DBAs had to add sufficient amount of 128GB LUN disks into the disk group to be able to perform the drop disk function. The addition of 3.5 TB took approximately 10 hours to complete on node #1.

```
SQL> alter diskgroup DG_EDOC_PD501 add disk
  2  '/dev/edoc_pd501_disk47',
  3  '/dev/edoc_pd501_disk48',
```

```
  4  '/dev/edoc_pd501_disk49',
  5  '/dev/edoc_pd501_disk50',
  6  '/dev/edoc_pd501_disk51',
  7  '/dev/edoc_pd501_disk52',
  8  '/dev/edoc_pd501_disk53',
  9  '/dev/edoc_pd501_disk54',
 10  '/dev/edoc_pd501_disk55',
 11  '/dev/edoc_pd501_disk56',
 12  '/dev/edoc_pd501_disk57',
 13  '/dev/edoc_pd501_disk58',
 14  '/dev/edoc_pd501_disk59',
 15  '/dev/edoc_pd501_disk60',
 16  '/dev/edoc_pd501_disk61',
 17  '/dev/edoc_pd501_disk62',
 18  '/dev/edoc_pd501_disk63',
 19  '/dev/edoc_pd501_disk64',
 20  '/dev/edoc_pd501_disk65',
 21  '/dev/edoc_pd501_disk66',
 22  '/dev/edoc_pd501_disk67',
 23  '/dev/edoc_pd501_disk68',
 24  '/dev/edoc_pd501_disk69',
 25  '/dev/edoc_pd501_disk70',
 26  '/dev/edoc_pd501_disk71',
 27  '/dev/edoc_pd501_disk72',
 28  '/dev/edoc_pd501_disk73',
 29  '/dev/edoc_pd501_disk74'
 30  rebalance power 11 wait;

Diskgroup altered.

Elapsed: 10:02:45.86
```

Again on node #1, using the power limit of 11, the drop and rebalance activity of approximately 3.3 TB took eight hours and forty-one minutes.

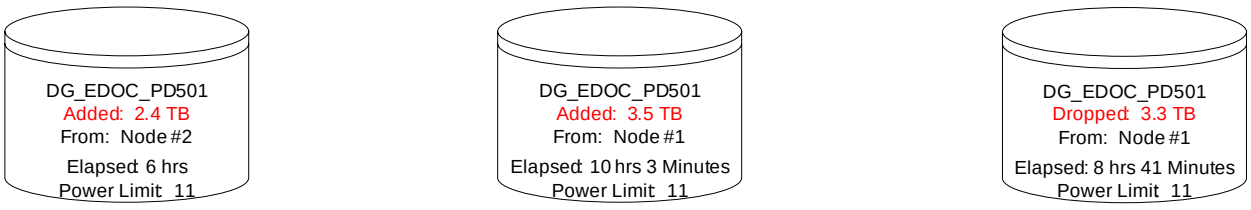
```
SQL> alter diskgroup DG_EDOC_PD501 drop disk
  2   'DG_EDOC_PD501_0000',
  3   'DG_EDOC_PD501_0001',
  4   'DG_EDOC_PD501_0002',
  5   'DG_EDOC_PD501_0003',
  6   'DG_EDOC_PD501_0004',
  7   'DG_EDOC_PD501_0005',
  8   'DG_EDOC_PD501_0006',
  9   'DG_EDOC_PD501_0007',
 10   'DG_EDOC_PD501_0008',
```

```
 11   'DG_EDOC_PD501_0009',
 12   'DG_EDOC_PD501_0010',
 13   'DG_EDOC_PD501_0011',
 14   'DG_EDOC_PD501_0012',
 15   'DG_EDOC_PD501_0013' rebalance power 11 wait;

Diskgroup altered.

Elapsed: 08:41:59.51
```

Additional disks were added anticipating future capacity growth requirements. During this exercise, approximately 2.4 TB of 128GB LUN Disks were added in 6 hours on node #2.

```
13:40:33 SQL> alter diskgroup DG_EDOC_PD501 add disk
13:40:33   2  '/dev/edoc_pd501_disk75',
13:40:33   3  '/dev/edoc_pd501_disk76',
13:40:33   4  '/dev/edoc_pd501_disk77',
13:40:33   5  '/dev/edoc_pd501_disk78',
13:40:33   6  '/dev/edoc_pd501_disk79',
13:40:33   7  '/dev/edoc_pd501_disk80',
13:40:33   8  '/dev/edoc_pd501_disk81',
13:40:33   9  '/dev/edoc_pd501_disk82',
13:40:33  10  '/dev/edoc_pd501_disk83',
13:40:33  11  '/dev/edoc_pd501_disk84',
13:40:33  12  '/dev/edoc_pd501_disk85',
13:40:33  13  '/dev/edoc_pd501_disk86',
13:40:33  14  '/dev/edoc_pd501_disk87',
13:40:33  15  '/dev/edoc_pd501_disk88',
13:40:33  16  '/dev/edoc_pd501_disk89',
13:40:33  17  '/dev/edoc_pd501_disk90',
13:40:33  18  '/dev/edoc_pd501_disk91',
13:40:33  19  '/dev/edoc_pd501_disk92',
13:40:33  20  '/dev/edoc_pd501_disk93',
13:40:33  21  '/dev/edoc_pd501_disk94'
13:40:33  22  rebalance power 11 wait;

Diskgroup altered.

Elapsed: 05:57:25.28
```

REBALANCE SUMMARY:



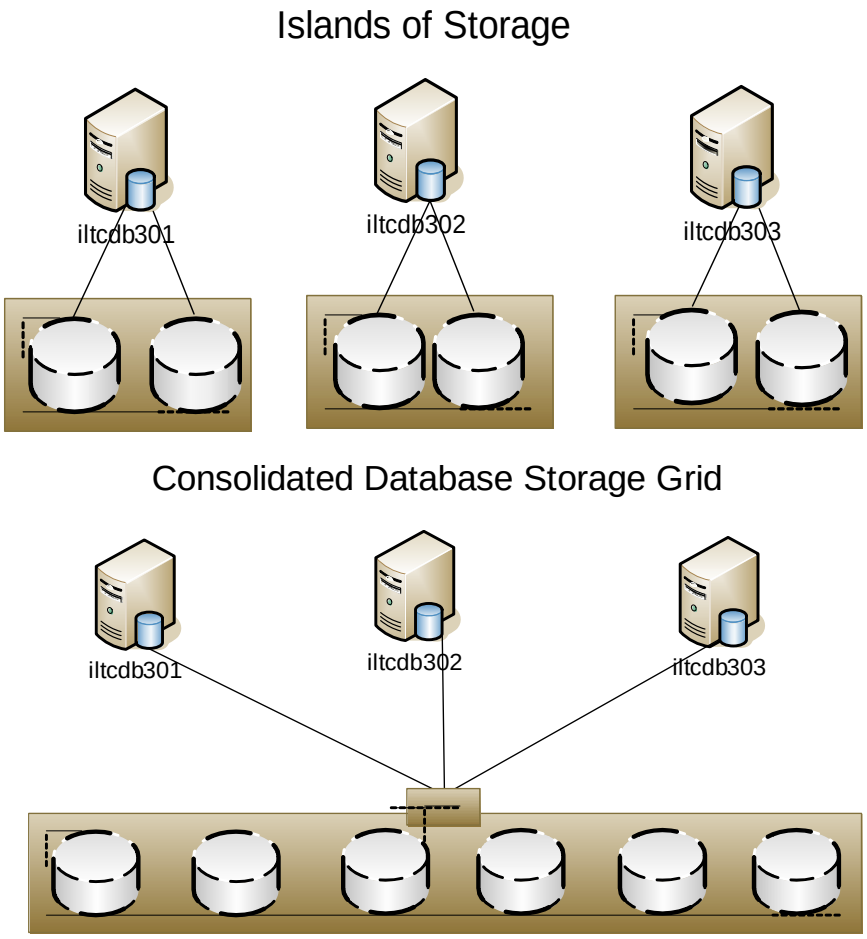
The addition and drop of ASM disks can occur at the same time. FNF DBAs were cautious in their efforts since they were dealing with multi-terabytes of disk allocation in a single weekend.

FUTURE PLANS OF ASM

FNF Database Administration has 100% reliance with ASM technology. FNF made a collective decision to standardize on ASM for all 10g database environments. ASM is now the standard for both RAC and non-RAC databases. Recently, databases were migrated from one data center to another. During the database relocation efforts, all the 10g databases were migrated to ASM.

Future Plans of ASM include leveraging a consolidated database storage grid. Instead of having free space available in pockets of islands of storage, free space can be utilized by other projects and applications. This means all databases will utilize disk farms for shared storage.

Oracle Clusterware is a required component to setting up a consolidated database storage grid. The only caveat is that DBAs will create a non-RAC database. Please review the following diagram for additional insight:



When the FNF/FIS went live with Oracle 11.5.10.2 EBusiness Suite on RAC with ASM, this "consolidated database storage grid" architecture was implemented. In the disaster recovery site, the pool of storage is shared between the DR Data Guard environment with DEV and QA environments.

AWR AND ADDM

AWR and ADDM reports were automated to run every 30 minutes during QA load test times. At the end of each collection period, the AWR in HTML format and ADDM reports in Text format were e-mailed to all the DBAs for review as attachments. In addition, the AWR repository retention was modified to retain 30 days of history.

LESSONS LEARNED

The following sections briefly describe some of the lessons learned by FNF during RAC implementations in numerous environments.

1. Production EDOCPRD Database server was mistakenly put on the pre-production subnet. As a result, the subnet had to be changed and the OCR re-registered.

```
1. Export ocr
sudo -u root /apps/oracle/product/10.2.0/CRS/bin/ocrconfig -export ocr_042006.dmp

2. Verify interface info before
EDOCPOC1 > ifconfig –a
EDOCPOC1 > oifcfg getif
en8  10.249.199.0  global  public
en9  172.16.32.0  global  cluster_interconnect

3. Stop All Resources
4. Modify listener.ora file to use hostname if applicable
5. Disable and stop CRS on each node
6. Remove and reconfigure public interface (SA)
7. Configure Public Interface (You must start CRS otherwise will get PRIF-10 error )
8. Modify VIPs -- Check en8 interface for new vip
9. Startup All Resources
EDOCPOC2 > sudo -u root srvctl modify nodeapps -n ictcdb622 -A
10.249.199.253/255.255.255.0/en8
EDOCPOC2 > sudo -u root srvctl modify nodeapps -n ictcdb621 -A
10.249.199.251/255.255.255.0/en8
```

2. ASM LUN size differences caused FNF to not be able to utilize 1.1 TB of ASM disk-capacity. Even though, everything with ASM indicated that there was 1.1 TB available FNF was not able to utilize it.

```
SQL> select name, total_mb, free_mb from v$asm_diskgroup;

NAME                           TOTAL_MB    FREE_MB
------------------------------ ---------- ----------
DG_EDOC_PD101                   1606596     729402
DG_EDOC_PD501                   7864010    1175795
DG_EDOC_PF101                    524608         50


SQL> alter tablespace POL_IMG_2006_Q2 add data file '+DG_EDOC_PD501' size
SQL> 10000m
  2  /
alter tablespace POL_IMG_2006_Q2 add data file '+DG_EDOC_PD501' size 10000m
*
ERROR at line 1:
ORA-01119: error in creating database file '+DG_EDOC_PD501'
ORA-17502: ksfdcre:4 Failed to create file +DG_EDOC_PD501
ORA-15041: diskgroup space exhausted
```

3. The ability to mirror OCR and voting disk at Oracle level is a 10gR2 feature and was not implemented with the 1st eDoc go-live date. Since then, FNF has instituted a new

standard to implement a minimal triple mirror of these devices to achieve (n+1) node survivability.

4. ASM provided some learning opportunities for all of the DBAs. ASM has its own paradigm shift to adjust since there are no copy capabilities within ASM. With Data Guard, FNF DBAs considered ASM to ASM file copies using dbms_file_transfer. RMAN was significantly faster and the dbms_file_transfer strategy was abandoned. An example of both RMAN Copy and DBMS_FILE_TRANSFER are provided to provided below.

Example of RMAN COPY

```
RMAN> copy datafile '+DG_DBA_DD501/clidba/datafile/users.345.616944627' to
'+DG_DBA_DF501/CLIDBA/DATAFILE/users2.dbf';


        Starting backup at 02-APR-07
        using target database control file instead of recovery catalog
        allocated channel: ORA_DISK_1
        channel ORA_DISK_1: sid=1080 devtype=DISK
        channel ORA_DISK_1: starting datafile copy
        input datafile fno=00005 name=+DG_DBA_DD501/clidba/datafile/users.345.616944627
        output filename=+DG_DBA_DF501/clidba/datafile/users2.dbf tag=TAG20070402T104536 recid=2
stamp=618835540
        channel ORA_DISK_1: datafile copy complete, elapsed time: 00:00:08
        Finished backup at 02-APR-07
```

Example of the DBMS_FILE_TRANSFER

```
1* create directory source_dir as '+DG_DBA_DD501/clidba/datafile'
SQL> /
Directory created.


SQL> create directory target_dir as '+DG_DBA_DF501/clidba/datafile';
Directory created.


SQL> exec DBMS_FILE_TRANSFER.COPY_FILE('SOURCE_DIR','users.345.616944627', 'TARGET_DIR', 'users.dbf');
PL/SQL procedure successfully completed.


Within ASMCMD utility, you can see the newly created OMF file and the associated alias:
ASMCMD [+DG_DBA_DF501/CLIDBA/DATAFILE] > ls -ltr
Type      Redund  Striped  Time             Sys  Name
DATAFILE  UNPROT  COARSE   APR 02 10:00:00  Y    COPY_FILE.738.618835347  N    users.dbf =>
+DG_DBA_DF501/CLIDBA/DATAFILE/COPY_FILE.738.618835347
```

Note: Please note the name of the COPIED file is called COPY_FILE.[OMF.EXTENSION]

- OMF became another area of contention for the DBAs. DBAs had a hard time giving up some of their OFA standards.

5. FNF has several "old time veteran" seasoned DBAs who have not fully embraced RMAN. They were forced to not only learn RMAN but to know it intimately since RMAN was really their only interface between ASM and AIX OS File system. ASM does not have a copy utility nor does it have FTP capabilities unless XML DB is installed on the database.

6. Flashback Area Management posed some issues during month end massive loads where 125 GB of raw data were often loaded. FNF interim solution was to turn off Flashback prior to month end HUGE Load. Because FRA was disabled and then re-enabled, FRA did not clean up properly several times. DBAs had to manually invoke asmcmd and remote the files manually.

```
$ asmcmd "ls +DG_EDOC_PF101/EDOCPRD/FLASHBACK" |awk {'print "asmcmd rm
+DG_EDOC_PF101/EDOCPRD/FLASHBACK/" $1'} > delete_flashback.asmcmd


$ksh delete_flashback.asmcmd
```

Flash enablement required the database to be in mounted state. When DBAs had to re-enable Flashback, a database outage had to be incurred.

7. During the upgrade to 10gR2 Patch set 1, DBAs ran into a minor upgrade issue where the installer complained about the Unix UID being greater than 64k. This minor bug forced the DBAs to upgrade the RAC environment one node at a time.

   To perform a single node update, DBAs had to invoke runInstaller with the updateNodeList option to CHANGE the CLUSTER_NODES to be SINGLE NODE configuration:

```
cd /apps/oracle/product/10.2.0/CRS/oui/bin
./runInstaller -updateNodeList "CLUSTER_NODES=ictcdb622" -local
ORACLE_HOME=/apps/oracle/product/10.2.0/CRS
```

In hind sight, this was a blessing in disguise. The DBAs learned from the beginning on how to perform rolling upgrades of the clusterware.

8. Metalink has a great article, **Doc ID: 259301.1** to convert crs_stat output to tabular format. This article also provides detailed command line options to srvctl. In addition to providing the output in a cleaner format, the HA Resource column is not truncated.

```
EDOCPRD1 > dba_crs
HA Resource                                   Target     State
-----------                                   ------     -----
ora.EDOCPRD.EDOCPRD1.inst                     ONLINE     ONLINE on ictcdb001
ora.EDOCPRD.EDOCPRD2.inst                     ONLINE     ONLINE on ictcdb002
ora.EDOCPRD.EDOCPRD3.inst                     ONLINE     ONLINE on ictcdb003
ora.EDOCPRD.db                                ONLINE     ONLINE on ictcdb002
ora.ictcdb001.ASM1.asm                        ONLINE     ONLINE on ictcdb001
ora.ictcdb001.EDOC_ICTCDB001.lsnr             ONLINE     ONLINE on ictcdb001
```

```
ora.ictcdb001.gsd                          ONLINE     ONLINE on ictcdb001
ora.ictcdb001.ons                          ONLINE     ONLINE on ictcdb001
ora.ictcdb001.vip                          ONLINE     ONLINE on ictcdb001
ora.ictcdb002.ASM2.asm                     ONLINE     ONLINE on ictcdb002
ora.ictcdb002.EDOC_ICTCDB002.lsnr          ONLINE     ONLINE on ictcdb002
ora.ictcdb002.gsd                          ONLINE     ONLINE on ictcdb002
ora.ictcdb002.ons                          ONLINE     ONLINE on ictcdb002
ora.ictcdb002.vip                          ONLINE     ONLINE on ictcdb002
ora.ictcdb003.ASM3.asm                     ONLINE     ONLINE on ictcdb003
ora.ictcdb003.EDOC_ICTCDB003.lsnr          ONLINE     ONLINE on ictcdb003
ora.ictcdb003.gsd                          ONLINE     ONLINE on ictcdb003
ora.ictcdb003.ons                          ONLINE     ONLINE on ictcdb003
ora.ictcdb003.vip                          ONLINE     ONLINE on ictcdb003
```

9. Refreshing Materialized Views with NOLOGGING Options disabled Parallel Execution across RAC nodes. The NOLOGGING Option caused the materialized view to run with one process on the originating node.

   As more and more issues were discovered with the materialized view refreshes, DBAs had to think creatively to resolve the performance issues. Instead of forcing logging at the database level, DBAs adopted to force logging at the tablespace level. The tablespaces associated with the materialized view were set to nologging to allow complete refreshes and index creation to be performed with increased performance.

10. Cluster Verify (cluvfy) on AIX has a bug when checking for patches and software components. It expects all components to be there (HACMP, GPFS, and ASM) and reports of missing components.

11. Of course, there were external circumstances DBAs had to be involved in
    a. DB2 to Oracle NLS_Character Set Conversion Issue " ¿ " – FNF faced issues of invalid characters after migrating the data over to Oracle. To resolve this invalid character, the Informatica team changed the code page to Unicode UTF from Informatica server.
    b. 4K truncation of CLOB data – Another Informatica Setting

12. DBAs realized that indexes needed to be dropped if not used. DBAs quickly implemented index monitoring to capture index usage information.

13. There were a couple of incidents where the database was literally in a hung state due to a known bug with Oracle with SYS.AUDSES$ sequence. After the DBAs changed the cache size for this sequence, they then modified application sequences accordingly. The audses$ sequence nextval cache size is supposed to be fixed in 10.2.0.3.

    At the time of the hang situation, DBAs had to perform system state dumps and provide them to Oracle Support to analyze. System statedumps need to be done twice after waiting one minute. There are also multiple levels of system statedumps that can be performed. Below is the system state dump script implemented:

```
alter session set tracefile_identifier='sysstatedump';
oradebug setmypid
oradebug unlimit

prompt Issuing command:  oradebug -g all dump systemstate 10
oradebug -g all dump systemstate 10
prompt Sleeping for 1 minute
!sleep 60
oradebug -g all dump systemstate 10

prompt We may not be able to do dump of locks
prompt Issuing command:  oradebug -g all dump locks 8
prompt If the sqlplus session hangs here for more than several minutes
prompt    kill it by hitting control-c
oradebug -g all dump locks 8
prompt Sleeping for 1 minute
!sleep 60
oradebug -g all dump locks 8
```

RECOMMENDATIONS FROM LESSONS LEARNED

It is crucial to realize that the success factor for RAC deployment depends heavily on the UNIX System Administration team who understands clustering technology. FNF was fortunate to have system administrators who were subject matter experts in HACMP, MPIO, Hitachi HDLM, SAN Architecture, AIX system administration, raw device technologies and much more. SAs were able to leverage their expertise in previous clustering technologies towards Oracle RAC.

FNF recommends as a general best practice thoroughly reviewing MetaLink and the patch database to find patches that may be applicable to your environment. In addition, DBAs need to keep up with the latest Oracle release and review it to see if it is a candidate for the RAC environment.

In the RAC world, the log files “tell a lot of the story” about the RAC environment. All the log files should be reviewed on a routine basis for CRS, ASM and Database related errors.

Oracle provides a free shell script utility called OS Watcher to report CPU, RAM, I/O and Network stress. This tool can be leveraged as an additional arsenal for the DBA for monitoring Oracle servers. This should be part of every RAC deployment - Note 301137.1.

At the same time, DBAs need to learn how to run racdiag: Doc_ID: **Note:135714.1** to collect troubleshooting information for the RAC environment.

DBAs participated in the 10gR2 beta software evaluation with Oracle Corporation. One of the DBAs was selected and participated on the on-site beta test event at Oracle Headquarters. The combination of both activities allowed FNF DBAs to become subject matter experts in the 10g R2 feature sets in the areas of RAC and server technologies before the rest of the Oracle Community.

For financial companies with mission critical applications, the database organization believe that it is imperative to have an early adopter as a pilot upgrade project for the rest of the databases. At FIS, the Oracle infrastructure databases such as RMAN repository, DBA Tools database and Grid Control repository serve as such pilot projects and are upgraded to the latest Oracle Release on RAC before the rest of the business in a production environment. Previous experiences have clearly proven that by "eating their own dog food" by upgrading their databases first to the latest release of Oracle, DBAs were able to establish a significant higher level of credibility with business stakeholders. At the same time, the DBAs gained a higher level of support experience in the new version of the database. If opportunity is provided to participate in Oracle's new release beta program, DBAs are encouraged to seize it.

CONCLUSION

Because of our successful implementation of eDoc on 12 TB on a 3 node RAC, we have proven to executives of RAC high availability and scalability. With such proven architecture, senior management looks to the DBAs to take future environments to RAC.

The future of RAC and Data Guard look promising at Fidelity National Information Services. With budgetary constraints, the company is investigating the feasibility of migrating from AIX servers to commodity hardware running on HP Proliant servers. With the proven success of Enterprise Linux, FIS can architect a future of running mission critical database servers on Redhat 64-bit Linux servers.

APPENDIX A—TEST PLAN

High Availibility Test
– SQL*Plus Failover from DB tier Instance/Session Failover
– Binaries Reinstall –With/Without Backup
DB Binaries lost, Datafiles & CRS binaries intact
All Binaries lost
– Instance Crash – Hard/Soft
Effect on remaining nodes in cluster
Effect on Application
Restoring instance to the cluster
– Hard O/S Crash
– Listener Loss
– CRS
CRS binaries lost
Voting Disk lost
OCR lost
Addition/Removal of resources from CRS
Recovering without impact on DB
– Cluster Inter-Connect Failure
– Switch Failure
– SAN Failure
– Hardware impacts node addition/removal
Impact on the cluster
– Grid Control failure
– Failover from Middle Tier

Backup Test
- Strategy
RMAN full backups
Incremental backups

Security Test
– Strategy
Access within/outside the group

Disaster Recover Test
– DR strategy

- Graceful shutdown
- Hard Failure

– Recoveries

- Point of Time
- Full
- Instance Self Recovery

Grid Control

- – ADDM
- Use ADDM to isolate any applications
- Performance Monitoring and Tuning
- Setup Alerts
- – AWR
- Create Baselines and thresholds for SLA management

Use Tuning & Monitoring for GCS,GES resources/latency

APPENDIX B: RAC COMMAND LINE IMPLEMENTATION

CREATE/CONFIGURE ASM INSTANCES(WITHOUT DBCA):

FNF Database Administration intentionally chose not to use DBCA. DBAs new to RAC needed to know the "behind the scenes" to RAC implementation. There were series of "step by step cookbooks" followed to create FNF's RAC environment. This following section will describe in detail (with accompanying scripts for automation purposes) how to setup a RAC environment.

All of the scripts mentioned in this section are custom scripts developed and maintained by FNF DBAs.

The steps below assume that the following has been completed successfully:

- CRS installation with OCR/Voting Disk was installed and setup
- ASM software was installed
- Oracle RAC binaries were installed

CREATE +ASM1 entry in /etc/oratab

```
+ASM1:/apps/oracle/product/10.2.0/ASM:N
```

Create init+ASM1.ora on each node:

+ASM1 > gen_rac_asm_init

```
###########################################################################
# ASM Init.ora parameter
###########################################################################

########################################
# Cluster Database
########################################
cluster_database=true

########################################
# Diagnostics and Statistics
########################################
background_dump_dest=/apps/oracle/admin/+ASM/bdump
core_dump_dest=/apps/oracle/admin/+ASM/cdump
user_dump_dest=/apps/oracle/admin/+ASM/udump

########################################
# Miscellaneous
########################################
instance_type=asm

########################################
# Pools
########################################
large_pool_size=12M

########################################
# Security and Auditing
########################################
```

```
remote_login_passwordfile=exclusive

#asm_diskgroups=+DG_DATA1_T1
asm_diskstring='/dev/ngsv2*'
#asm_diskgroups=''

+ASM3.instance_number=3
+ASM2.instance_number=2
+ASM1.instance_number=1
+ASM3.local_listener="(ADDRESS=(PROTOCOL=TCP)(NODE=ictcdb453-vip)(PORT=1970))"
+ASM2.local_listener="(ADDRESS=(PROTOCOL=TCP)(NODE=ictcdb452-vip)(PORT=1970))"
+ASM1.local_listener="(ADDRESS=(PROTOCOL=TCP)(NODE=ictcdb451-vip)(PORT=1970))"
```

View the contents of gen_rac_asm_init script:
http://dba.ctt.com/sh/gen_rac_asm_init

This script reads two configuration file:
http://dba.ctt.com/sh/rac.conf
and
http://dba.ctt.com/sh/gen.conf

Run Orapwd file=orapw+ASM1 on each node

```
orapwd file=orapw+ASM1 password=[SYS_PASSWORD] entries=25
```

Create OFA Directory for ASM: /apps/oracle/admin/ASM

```
cd $HOME/admin
mkdir ASM
ln –s ASM +ASM1
cd ASM
mkdir bdump  cdump  hdump  pfile  udump
gen_rac_asm_init > $HOME/admin/ASM/pfile/init+ASM.ora
cd $ORACLE_HOME/dbs
ln –s $HOME/admin/ASM/pfile/init+ASM.ora init+ASM1.ora
```

STARTUP NOMOUNT THE ASM INSTANCE

```
sqlplus / as sysdba
Startup nomount
```

Create Disk Group for Data:

```
+ASM1 > ./gen_dg DG_NGS_DD501 /dev/ngsv2_dd501
CREATE DISKGROUP DG_NGS_DD501
EXTERNAL REDUNDANCY
DISK
'/dev/ngsv2_dd501_disk01'
,'/dev/ngsv2_dd501_disk02'
,'/dev/ngsv2_dd501_disk03'
,'/dev/ngsv2_dd501_disk04'
,'/dev/ngsv2_dd501_disk05'
,'/dev/ngsv2_dd501_disk06'
,'/dev/ngsv2_dd501_disk07'
,'/dev/ngsv2_dd501_disk08'
/
```

To see the script content of gen_dg:
http://dba.ctt.com/sh/gen_dg

Create Disk Group for FlashBack Area:

```
+ASM1 > ./gen_dg DG_NGS_DF501 /dev/ngsv2_df501
CREATE DISKGROUP DG_NGS_DF501
EXTERNAL REDUNDANCY
DISK
'/dev/ngsv2_df501_disk01'
,'/dev/ngsv2_df501_disk02'
,'/dev/ngsv2_df501_disk03'
,'/dev/ngsv2_df501_disk04'
/
```

CREATE DISKGROUPS

```
@DG_DATA.sql  -- This is the first script above.  You have to save it as DG_DATA.sql
@DG_FLASH.sql –- This is the second script above.  You have to save the output ass DG_FLASH.sql
Shutdown immediate
```

ADD asm_diskgroups to init+ASM1.ora initialization file:

```
asm_diskgroups=DG_NGS_DD501,DG_NGS_DF501
```

```
Startup (will auto mount the diskgroups)
@$SQL/df
Shutdown immediate
```

Register FIRST instance to cluster:

```
srvctl add asm –n ictcdb621 –i +ASM1 –o /apps/oracle/product/10.2.0/ASM
```

Push $ORACLE_HOME/dbs/init+ASM1.ora to second node as $ORACLE_HOME/dbs/init+ASM2.ora

```
cd $ORACLE_HOME/dbs
$SH/spush ictcdb452 init+ASM1.ora
$SH/remote_mv ictcdb452 init+ASM1.ora init+ASM2.ora
```

Note:

View remote_mv script: http://dba.ctt.com/sh/remote_mv

Push ASM OFA directory to the other node(s)

```
cd $HOME
cd admin
$SH/spush ictcdb452 +ASM
$SH/spush ictcdb452 +ASM1
$SH/remote_mv ictcdb452 +ASM1 +ASM2
```

Register Other instance(s) to the cluster

```
srvctl add asm –n ictcdb622 –i +ASM2 –o /apps/oracle/product/10.2.0/ASM
```

Startup each +ASM instance:

```
Srvctl start asm –n ictcdb451
Srvctl start asm –n ictcdb452
Srvctl start asm –n ictcdb453
```

CREATE/CONFIGURE RAC DATABASE AND INSTANCES

FIRST CREATE STANDALONE NON-RAC Database using the gen script:

http://dba.ctt.com/sh/gen

This file references a configuration file which specifies parameters like asm vs. file system and names for diskgroups:

http://dba.ctt.com/sh/gen.conf

Create database using generated script:

ictcdb451:/apps/oracle/general/sh

```
NGSNDV1 > ./gen -o database
PREFIX=/data
ORACLE_BASE_DIR=/apps/oracle
ORACLE_FILE_SYSTEM=asm
DEFAULT_DISK_GROUP=+DG_DATA1_T1
REDO_DISK_GROUP1=+DG_DATA1_T1
REDO_DISK_GROUP2=+DG_DATA1_T1

#   Please review parameters from gen.conf Please enter Y to continue: [Y]

Please Enter Database Name:
NGSNDV
Please enter version:
Acceptable versions are [8i/9i/10g]
10g
Please enter the password for sys:
xxxxxxx
Please enter the password for system:
xxxxxxx
```

This script will create the following output using the DG specified in the gen.conf file:

```
create database  NGSNDV
user sys identified by xxxxxx
user system identified by xxxxxx
  maxdatafiles  1021
    maxinstances  4
    maxlogfiles   16
    character set WE8MSWIN1252
    national character set AL16UTF16
    data file
        '+DG_DATA1_T1'      size 1000M
```

```
        autoextend off extent management local
    sysaux data file '+DG_DATA1_T1' size 1000m
    default temporary tablespace temp
    tempfile '+DG_DATA1_T1' size 4000m
    uniform size 1m
    undo tablespace undo_rbs
    data file '+DG_DATA1_T1' size 4000m
    logfile
        ('+DG_DATA1_T1'
         ,'+DG_DATA1_T1')         size 300M,
        ('+DG_DATA1_T1'
         ,'+DG_DATA1_T1')         size 300M,
        ('+DG_DATA1_T1'
         ,'+DG_DATA1_T1')         size 300M,
        ('+DG_DATA1_T1'
         ,'+DG_DATA1_T1')         size 300M,
        ('+DG_DATA1_T1'
         ,'+DG_DATA1_T1')         size 300M,
        ('+DG_DATA1_T1'
         ,'+DG_DATA1_T1')         size 300M;
REM - Creates data dictionary views.
@?/rdbms/admin/catalog.sql

REM - Scripts for procedural option
@?/rdbms/admin/catproc.sql

REM - Scripts for export/import -
@?/rdbms/admin/catexp.sql

create tablespace TOOLS;
create tablespace USERS;

alter user sys temporary tablespace temp;
alter user system default tablespace tools temporary tablespace temp;
alter user system quota unlimited on tools;
alter user system quota 0 on system;

connect system/xxxxxxxx
@?/rdbms/admin/catdbsyn.sql
@?/sqlplus/admin/pupbld.sql
create user oracle identified externally
default tablespace tools
temporary tablespace temp;
grant dba to oracle;
spool off
```

Please run the log file in the /tmp directory: **/tmp/NGSNDV.log**

Steps to convert standalone non-RAC database to RAC database:

The cleanest way to do this conversion is to name the standalone non-RAC database as a RAC database name. For example, set the db_name and db_unique_name to EDOCPRD and the instance_name to EDOCPRD1. To change the name of an existing database, you can do one of the following: a) Re-create the control file and set the database name. If you're using ASM, this will create another directory within the ASM db directory structure, so you'll have to use RMAN Copy to move it to the new directory.

b) Export the data and re-create the entire database from scratch with the new name (provided you have the create scripts), and import the data into the new database.

By changing the name to this standard, you are basically making it a single-node RAC database and adding nodes/instances to it later.

Also, set up the Oracle directory structure to use this database name, for example:
/apps/oracle/admin/EDOCPRD/… and create a soft link as EDOCPRD1 → EDOCPRD
(do this on each node)

Do the same thing for the pfile init.ora in $ORACLE_HOME/dbs directory for example:
initEDOCPRD1.ora -> /apps/oracle/admin/EDOCPRD/pfile/initEDOCPRD.ora
(do this on each node)

Also set the oratab entry as the instance name on each node (not the database name), for example:
EDOCPRD1:/apps/oracle/product/10.2.0/RACDB:Y

Remember to carry out the following steps on the first node/instance only. On the second and subsequent nodes/instances, you will just set up the init.ora file and mount the datatabase.

Create RAC Data Dictionary Views

```
SQL> @?/rdbms/admin/catclust.sql
```

FOR EACH INSTANCE: You will need to modify init.ora file to include:

```
*.cluster_database_instances=3
*.cluster_database=true
EDOCPRD2.thread=2
EDOCPRD2.instance_number=2
EDOCPRD2.undo_tablespace='UNDO_RBS2'
*.remote_login_passwordfile='exclusive'
```

Create spfile from pfile; -- put spfile on shared DG

```
create spfile=' +DG_EDOC_PD101/EDOCPRD/PARAMETERFILE/spfile.275.584472553' from pfile;
```

To view the pfile back:

```
create pfile='/tmp/initEDOCPRD2.ora' from spfile;
```

Modify pfile to reference shared SPFILE

```
spfile='+DG_EDOC_PD101/EDOCPRD/PARAMETERFILE/spfile.275.584472553'
```

ADD DATABASE TO RAC CLUSTER

```
Srvctl add database –d EDOCPRD –o /apps/oracle/product/10.2.0/RACDB
```

Use Srvctl to add each instance

```
srvctl add instance -d EDOCPRD -i EDOCPRD1 -n [HOSTNAME]
```

For your first RAC instance, shutdown the database manually and start the database using srvctl utility:

```
srvctl start instance -d EDOCPRD -i EDOCPRD2
Or
Srvctl start database –d EDOCPRD
```

We do not want to use DBCA to create RAC cluster environments. We do not know what is happening behind the scene.

Please use $SH/gen_rac_init to add the instance. **This script accepts two parameters:**

1. INSTANCE_NAME
2. INSTANCE_NUMBER

STANDARDS introduced:

1. UNDO tablespace will be called undo_rbs1 for instance #1, undo_rbs2 for instance #2, etc.
2. Redo log group numbers will start with 21 for instance #2, 31 for instance #3, etc.
3. Thread numbers will be thread#2 for instance#2, thread#3 for instance#3, etc.

Assumptions:

At this point, we have a single node rac UNDO tablespace will be created in the db_create_file_dest directory as specified in init.ora parameter

Redo logs will be created in the db_create_file_dest directory as specified in init.ora parameter

To view the gen_rac_init script:

```
http://dba.ctt.com/sh/gen_rac_init
```

This script will generate the following output for the second node/instance. Execute all lines from SQL*Plus logged in as the SYS user (still on the first node/instance):

```
EDOCPRD1 > gen_rac_init EDOCPRD2 2
-- Adding groups to thread# 2
ALTER DATABASE ADD LOGFILE THREAD 2 GROUP 21 size 300m;
```

```
ALTER DATABASE ADD LOGFILE THREAD 2 GROUP 22 size 300m;
ALTER DATABASE ADD LOGFILE THREAD 2 GROUP 23 size 300m;
ALTER DATABASE ADD LOGFILE THREAD 2 GROUP 24 size 300m;
ALTER DATABASE ADD LOGFILE THREAD 2 GROUP 25 size 300m;
ALTER DATABASE ADD LOGFILE THREAD 2 GROUP 26 size 300m;
ALTER DATABASE ENABLE THREAD 2;

-- Creating undo_rbs2 tablespace for 2
create undo tablespace undo_rbs2 data file size 2000m;

-- Setting instance number
alter system set instance_number=2 scope=spfile sid='EDOCPRD2';
alter system set thread=2 scope=spfile sid='EDOCPRD2';
alter system set undo_tablespace=undo_rbs2 scope=spfile sid='EDOCPRD2';

-- Increasing total number of cluster databases
alter system reset cluster_database_instances scope=spfile sid='*';
alter system set cluster_database_instances=2 scope=spfile sid='*';

-- Adding instance EDOCPRD2 to RAC
host srvctl add instance -d EDOCPRD -i EDOCPRD2 -n [HOSTNAME]
host srvctl start instance -d EDOCPRD -i EDOCPRD2
```

We are using spfile. The only entry in our init.ora file should be the location of the spfile , so modify pfile to reference shared SPFILE (and delete everything else):

```
spfile='+DG_EDOC_PD101/EDOCPRD/PARAMETERFILE/spfile.275.584472553'
```

Notice that even the spfile is an OMF file.

For you second and subsequent instance, make sure to create:

- Orapw$ORACLE_SID file in $ORACLE_HOME/dbs
- Init$ORACLE_SID.ora in $ORACLE_HOME/dbs with one line spfile='location of spfile';

CREATE TNSNAMES Entry using VIP

+ASM1 > gen_rac_tnsnames

```
NGSDVA_RAC =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = ictcdb451-vip.ngs.fnf.com)(PORT = 1970))
      (ADDRESS = (PROTOCOL = TCP)(HOST = ictcdb452-vip.ngs.fnf.com)(PORT = 1970))
```

```
      (ADDRESS = (PROTOCOL = TCP)(HOST = ictcdb453-vip.ngs.fnf.com)(PORT = 1970))
      (LOAD_BALANCE = yes)
    )
    (CONNECT_DATA =
      (SERVICE_NAME = EDOCPRD)
      (FAILOVER_MODE =
        (TYPE = SELECT)
        (METHOD = BASIC)
        (RETRIES = 180)
        (DELAY = 5)
      )
    )
  )
```

gen_rac_tnsnsames script reads rac.conf to generate the tnsnames.ora output for review:
http://dba.ctt.com/sh/gen_rac_tnsnames

RUN Enhanced crs_stat to see availability of all background daemons:
To view this script:
http://dba.ctt.com/sh/dba_crs

EDOCPRD > $SH/dba_crs

```
HA Resource                                   Target     State
-----------                                   ------     -----
ora.EDOCPRD.EDOCPRD1.inst                     ONLINE     ONLINE on ictcdb001
ora.EDOCPRD.EDOCPRD2.inst                     ONLINE     ONLINE on ictcdb002
ora.EDOCPRD.EDOCPRD3.inst                     ONLINE     ONLINE on ictcdb003
ora.EDOCPRD.db                                ONLINE     ONLINE on ictcdb002
ora.ictcdb001.ASM1.asm                        ONLINE     ONLINE on ictcdb001
ora.ictcdb001.EDOC_ICTCDB001.lsnr             ONLINE     ONLINE on ictcdb001
ora.ictcdb001.gsd                             ONLINE     ONLINE on ictcdb001
ora.ictcdb001.ons                             ONLINE     ONLINE on ictcdb001
ora.ictcdb001.vip                             ONLINE     ONLINE on ictcdb001
ora.ictcdb002.ASM2.asm                        ONLINE     ONLINE on ictcdb002
ora.ictcdb002.EDOC_ICTCDB002.lsnr             ONLINE     ONLINE on ictcdb002
ora.ictcdb002.gsd                             ONLINE     ONLINE on ictcdb002
ora.ictcdb002.ons                             ONLINE     ONLINE on ictcdb002
ora.ictcdb002.vip                             ONLINE     ONLINE on ictcdb002
ora.ictcdb003.ASM3.asm                        ONLINE     ONLINE on ictcdb003
ora.ictcdb003.EDOC_ICTCDB003.lsnr             ONLINE     ONLINE on ictcdb003
ora.ictcdb003.gsd                             ONLINE     ONLINE on ictcdb003
ora.ictcdb003.ons                             ONLINE     ONLINE on ictcdb003
ora.ictcdb003.vip                             ONLINE     ONLINE on ictcdb003
```

Check on running instances on all the RAC nodes

```
+ASM1 > rac_status
  oracle 377074      1   0   Mar 20      -  0:15 asm_pmon_+ASM1
  oracle 479334      1   0 12:31:17      -  0:07 ora_pmon_NGSDVA1
  oracle 606250      1   0 16:04:41      -  0:04 ora_pmon_NGSQAA1
  oracle 438448      1   0   Mar 20      -  0:13 asm_pmon_+ASM2
  oracle 602250      1   0 12:31:18      -  0:07 ora_pmon_NGSDVA2
  oracle 839714      1   0 16:04:41      -  0:04 ora_pmon_NGSQAA2
  oracle 188596      1   0 12:31:17      -  0:10 ora_pmon_NGSDVA3
  oracle 315398      1   0 16:04:43      -  0:04 ora_pmon_NGSQAA3
  oracle 225542      1   0   Mar 20      -  0:11 asm_pmon_+ASM3
```

rac_status script accepts a parameter so you can view grep on any process on all RAC nodes. This scripts reads rac.conf configuration file so it will scan only the nodes specified in the rac.conf file.

Check on CRS background processes

```
+ASM1 > rac_status d.bin
  oracle 217208 303332   0   Mar 20      -  2:42 /apps/oracle/product/10.2.0/CRS/bin/ocssd.bin
    root 229494      1   0   Mar 20      -  3:38 /apps/oracle/product/10.2.0/CRS/bin/crsd.bin
reboot
  oracle 270352      1   0   Mar 20      -  3:13 /apps/oracle/product/10.2.0/CRS/bin/evmd.bin
    root  86150      1   0   Mar 20      -  3:39 /apps/oracle/product/10.2.0/CRS/bin/crsd.bin
reboot
  oracle 299246      1   0   Mar 20      -  3:07 /apps/oracle/product/10.2.0/CRS/bin/evmd.bin
  oracle 319670 340126   0   Mar 20      -  2:41 /apps/oracle/product/10.2.0/CRS/bin/ocssd.bin
    root 225348      1   0   Mar 20      -  3:14 /apps/oracle/product/10.2.0/CRS/bin/crsd.bin
reboot
  oracle 274520 176360   0   Mar 20      -  2:41 /apps/oracle/product/10.2.0/CRS/bin/ocssd.bin
  oracle 200962      1   0   Mar 20      -  0:17 /apps/oracle/product/10.2.0/CRS/bin/evmd.bin
```

APPENDIX C – DATA GUARD STEP BY STEP COOKBOOK

PURPOSE OF THIS COOKBOOK

The primary purpose of this cookbook is to automate what is conceived to be the more tedious part of setting up Data Guard when using the SQL*Plus command line. Since the focus is automation, myriads of scripts are provided with URLs to the source of the script. There are also many references to the FIS DBA website: http://dba.fnfis.com/docs/dg/ with a virtual docs/dg directory. The complete set of scripts will be provided in the script depot and will be provided on request.

The name of the database is intentionally called CLIDBA (Command Line Interface DBA). The primary focus will be to create a Data Guarded environment from 100% command line interface. Also, all of the database environments are ASM to ASM data guard environments.

PREPARE THE PRIMARY DATABASE FOR DATA GUARD

Create a password on the Standby $ORACLE_HOME/dbs directory

```
cd $ORACLE_HOME/dbs
CLIDBA > orapwd file=orapwCLIDBA password=oracle123 entries=25
```

Create a password on the PRIMARY $ORACLE_HOME/dbs directory (if not already)

```
cd $ORACLE_HOME/dbs
CLIDBA > orapwd file=orapwCLIDBA password=oracle123 entries=25
```

Enable Force Logging –

Place the primary database in FORCE LOGGING mode after database creation:

```
SQL> ALTER DATABASE FORCE LOGGING;
```

Backup the primary database using RMAN –

http://dba.fnfis.com/docs/dg/dg_backup_database.sql

```
CLIDBA > cat dg_backup_database.sql
run
{
allocate channel d1 type disk;
set limit channel d1 kbytes = 4000000;
allocate channel d2 type disk;
set limit channel d2 kbytes = 4000000;
```

```
allocate channel d3 type disk;
set limit channel d3 kbytes = 4000000;
allocate channel d4 type disk;
set limit channel d4 kbytes = 4000000;


backup as compressed backupset incremental level 0 diskratio=0 tag CLIDBA_backup_0z_full_03112007 filesperset 5
format '/apps/oracle/admin/CLIDBA/bkups/%d.%s.%p.%t.DB' (database) ;
sql "alter system archive log current";
sql "alter system switch logfile";
sql "alter system switch logfile";
change archivelog all validate;
sql "alter database backup controlfile to trace";
sql "alter database backup controlfile to ''/apps/oracle/admin/CLIDBA/log/control01_CLIDBA_03112007.bkup''
reuse";
backup as compressed backupset diskratio=0 filesperset 5 format
'/apps/oracle/admin/CLIDBA/bkups/%d.%s.%p.%t.ARC' skip inaccessible (archivelog all ) ;
sql "alter database backup controlfile to trace";
backup diskratio=0 tag CLIDBA_control_03112007 format '/apps/oracle/admin/CLIDBA/bkups/%d.%s.%p.%t.CTL'
(current controlfile);


release channel d1;
release channel d2;
release channel d3;
release channel d4;
}
```

There are two options to registering a database backup. The first option is use the rman catalog and the controlfile. The second option is to use JUST the controlfile without the catalog.

In this example, we will clone the database using JUST the control file. First connect to the source database via the command: "rman target /" command.

```
CLIDBA > rman target /

Recovery Manager: Release 10.2.0.2.0 - Production on Sun Mar 11 14:21:16 2007

Copyright (c) 1982, 2005, Oracle.  All rights reserved.

connected to target database: CLIDBA (DBID=1936074589)

RMAN> @dg_backup_database.sql
```

For complete output of the RMAN backup, please visit the DBA website:
http://dba.fnfis.com/docs/dg/CLIDBA_rman_backup.log

Configure INIT.ORA Parameters on the Primary Site

First, modify the master config file: dg_master.conf

http://dba.fnfis.com/docs/dg/dg_master.conf

The following parameters will need to be modified:

- DATA_DG
- DATA_DG2 (if applicable)
- FLASH_DG
- DR_VIP
- PORT

The values for PRIMARY_HOSTNAME, PRIMARY_DOMAIN and PRIMARY_VIP do not need to be modified since it is derived from the current environment where the script is being executed.

The primary driver script to generate the alter system commands to setup data guard on the primary and standby site is dg_alter_system.ksh which utilizes sed and parses out the values from dg_alter_system_p.txt and dg_alter_system_dg.txt template files based on the master config file (dg_master.conf).

Second, on the primary database, execute the alter_system.ksh script and pass the parameter "P" and redirect the output to dg_alter_system_p.sql:

```
dg_alter_system.ksh P > dg_alter_system_p.sql
```

Please note that all the scripts are intentionally standardized to start with dg_ prefix.

This script will generate a series of alter system commands that will prepare the primary site for DR:

```
CLIDBA > dg_alter_system.ksh P
spool alter_system_CLIDBA.log
-- alter system set audit_trail='NONE' scope=spfile;     # added for DG need for open read only
alter system set db_recovery_file_dest_size=32g comment='# added for DG' scope=both sid='*';
alter system set db_unique_name='CLIDBA' comment ='# added for DG' scope=spfile sid='*';


alter system set db_file_name_convert='+DG_DBA_DD101/CLIDBADR','+DG_DBA_DD101/CLIDBA'   comment='# added for
DG' scope=spfile sid='*';


alter system set db_recovery_file_dest_size=32g comment='# added for DG' scope=both sid='*';
alter system set db_recovery_file_dest='+DG_DBA_DF501' comment='# added for DG' scope=both sid='*';
alter system set event='10298 trace name context forever, level 32'  comment='# added for RMAN backup on NFS
file system' scope=spfile sid='*';


alter system set fal_client='CLIDBA_LTC' comment='# added for DG Net service name in prim tnsnames.ora points
to Stdby' scope=both sid='*';
```

```
alter system set fal_server='CLIDBA_JTC' comment='# added for DG Net service name in stdby tnsnames.ora points
to Primary' scope=both sid='*';


alter system set log_archive_config='DG_CONFIG=(CLIDBA,CLIDBADR)' comment='# added for DG primary followed by
standby db unique name' scope=both sid='*';


alter system set log_file_name_convert='+DG_DBA_DF501/CLIDBADR','+DG_DBA_DF501/CLIDBA' comment='# added for DG'
scope=spfile sid='*';


alter system set standby_file_management='AUTO' comment='# added for DG' scope=both sid='*';


alter system set log_archive_config='DG_CONFIG=(CLIDBA,CLIDBADR)' comment='# added for DG primary followed by
standby db unique name' scope=both sid='*';


alter system set log_archive_dest_1='LOCATION=+DG_DBA_DF501
VALID_FOR=(ALL_LOGFILES,ALL_ROLES)
DB_UNIQUE_NAME=CLIDBA' comment='# added for DG' scope=both sid='*';


alter system set log_archive_dest_2='SERVICE=CLIDBA_JTC lgwr async reopen=30
VALID_FOR=(ONLINE_LOGFILES,PRIMARY_ROLE)
DB_UNIQUE_NAME=CLIDBADR' comment='# added for DG' scope=both sid='*';


alter system set log_archive_dest_state_2='defer' comment='# added for DG' scope=both sid='*';


spool off
```

Third, logon to the primary database and run the script that was generated:

```
sqlplus / as sysdba
@dg_alter_system_p.sql
```

There should be NO ERRORS to the execution of the script above. The initialization file on the primary site should look like this after the changes introduced above:

http://dba.fnfis.com/docs/dg/initCLIDBA.primary.ora

Create a standby backup controlfile to build the Data Guard database

This script is captured in the backup_standby_controlfile.sql script:
http://dba.fnfis.com/docs/dg/dg_backup_standby_controlfile.sql

```
CLIDBA > rman target /


Recovery Manager: Release 10.2.0.2.0 - Production on Sun Mar 11 17:35:47 2007


Copyright (c) 1982, 2005, Oracle.  All rights reserved.
```

```
connected to target database: CLIDBA (DBID=1936074589)


RMAN> @backup_standby_controlfile.sql


RMAN> backup current controlfile for standby diskratio=0 tag=STANDBY_INIT
format='/apps/oracle/admin/CLIDBA/bkups/%d.%s.%p.%t.CTLSTDBY';
Starting backup at 11-MAR-07
using target database control file instead of recovery catalog
allocated channel: ORA_DISK_1
channel ORA_DISK_1: sid=1071 devtype=DISK
channel ORA_DISK_1: starting full datafile backupset
channel ORA_DISK_1: specifying datafile(s) in backupset
including standby control file in backupset
channel ORA_DISK_1: starting piece 1 at 11-MAR-07
channel ORA_DISK_1: finished piece 1 at 11-MAR-07
piece handle=/apps/oracle/admin/CLIDBA/bkups/CLIDBA.11.1.616959358.CTLSTDBY tag=STANDBY_INIT
comment=NONE
channel ORA_DISK_1: backup set complete, elapsed time: 00:00:01
Finished backup at 11-MAR-07


RMAN> **end-of-file**
```

Copy the Standby Controlfile to the DR Server using SFTP

```
CLIDBA > echo "put CLIDBA.11.1.616959358.CTLSTDBY /apps/oracle/admin/CLIDBA/bkups" |sftp
iltcdb408
Connecting to iltcdb408...
sftp> put CLIDBA.11.1.616959358.CTLSTDBY /apps/oracle/admin/CLIDBA/bkups
Uploading CLIDBA.11.1.616959358.CTLSTDBY to
/apps/oracle/admin/CLIDBA/bkups/CLIDBA.11.1.616959358.CTLSTDBY
CLIDBA.11.1.616959358.CTLSTDBY
100%   10MB  10.4MB/s   00:01                                                    100%
13MB 817.0KB/s 00:16
```

Create a copy of the init.ora file and push it out to the DR site:

```
SQL> create pfile='/tmp/initCLIDBA.ora' from spfile;


File created.


CLIDBA > scp initCLIDBA.ora oracle@iltcdb408:/apps/oracle/admin/CLIDBA/pfile
initCLIDBA.ora
100% 2482     2.4KB/s   00:00
```

STANDBY SITE - Query the ASM instance and view the available disk groups:

http://dba.fnfis.com/docs/dg/lsdg.ksh

```
State    Type    Rebal  Unbal  Sector  Block       AU  Total_MB  Free_MB  Req_mir_free_MB
Usable_file_MB  Offline_disks  Name
MOUNTED  EXTERN  N      N         512   4096  1048576    295092   144720                0
144720              0  DG_DD501/
```

You can see that the diskgroup on the primary site is different than the DR site. On the DR site, the diskgroup is DG_DD501 where as the diskgroup on the primary site in this particular case is DG_DBA_DD501 and DG_DBA_DF501.

Make necessary modifications to the spfile on the DR site:

Before we can make modifications to the spfile, we must first convert the init.ora file to spfile. First, we must manually create the spfile parameter directory:

Change SID to +ASM1 and launch asmcmd in interactive mode:

```
export ORACLE_SID=+ASM1
export ORAENV_ASK=NO
. oraenv
asmcmd –p
mkdir +DG_DD501/clidbadr/parameterfile
```

Change SID back to CLIDBA and Create spfile from pfile:

```
export ORACLE_SID=CLIDBA
export ORAENV_ASK=NO
. oraenv

sqlplus / as sysdba
startup nomount pfile=/apps/oracle/admin/CLIDBA/pfile/initCLIDBA.ora;
```

Finally convert pfile to spfile:

```
  1* create spfile='+DG_DD501/clidbadr/parameterfile/spfileCLIDBA' from pfile
SQL> /

File created.
```

Shutdown immediate the database and change the initCLIDBA.ora file to reflect the spfile:

```
spfile='+DG_DD501/clidbadr/parameterfile/spfileCLIDBA'
```

Now we are ready to make necessary modifications to init.ora on the standby database:

```
./dg_alter_system.ksh DG > dg_alter_system_dg.sql
```

The dg_alter_system.ksh script accepts the DG parameter to create the syntax to alter the Data Guard environment.

http://dba.fnfis.com/docs/dg/dg_alter_system.ksh

As mentioned earlier, this script works with pre-created template files and uses sed to parse and replace values based on the DG master configuration file. Please be aware that the generated script needs to be reviewed and make sure that it is in compliance to the DG environment. For example, if your diskgroup from the source node and DR node is different, please make appropriate corrections manually in the dg_alter_system_dg.sql file. If the diskgroups are the same, you should not have to make any changes.

Run dg_alter_system_dg.sql script. This script intentionally only modifies the spfile so the database has to be bounced for the new parameters to take effect.

```
sqlplus / as sysdba
@dg_alter_system_dg.sql
```

Shutdown and restart the database in nomount mode:

```
SQL> shutdown immediate;
SQL> startup nomount;
```

Restore the standby control file using rman

```
CLIDBA > rman target /

Recovery Manager: Release 10.2.0.2.0 - Production on Sun Mar 11 22:10:21 2007

Copyright (c) 1982, 2005, Oracle.  All rights reserved.

connected to target database: CLIDBA (not mounted)

RMAN> restore controlfile from '/apps/oracle/admin/CLIDBA/bkups/CLIDBA.11.1.616959358.CTLSTDBY';

Starting restore at 11-MAR-07
using target database control file instead of recovery catalog
allocated channel: ORA_DISK_1
channel ORA_DISK_1: sid=1090 devtype=DISK

channel ORA_DISK_1: restoring control file
channel ORA_DISK_1: restore complete, elapsed time: 00:00:19
output filename=+DG_DD501/clidbadr/controlfile/control01.ctl
Finished restore at 11-MAR-07
```

Mount the database

```
RMAN> sql "alter database mount";

sql statement: alter database mount
released channel: ORA_DISK_1
```

Restore the database from the backup file system

```
RMAN> run
2> {
3> allocate channel d1 type disk;
4> allocate channel d2 type disk;
5> allocate channel d3 type disk;
6> allocate channel d4 type disk;
7> restore database;
8> switch datafile all;
9> release channel d1;
10> release channel d2;
11> release channel d3;
12> release channel d4;
13> }
using target database control file instead of recovery catalog
allocated channel: d1
channel d1: sid=1090 devtype=DISK

allocated channel: d2
channel d2: sid=1089 devtype=DISK

allocated channel: d3
channel d3: sid=1082 devtype=DISK

allocated channel: d4
channel d4: sid=1081 devtype=DISK

Starting restore at 11-MAR-07
Starting implicit crosscheck backup at 11-MAR-07
Crosschecked 10 objects
Finished implicit crosscheck backup at 11-MAR-07

Starting implicit crosscheck copy at 11-MAR-07
Crosschecked 1 objects
Finished implicit crosscheck copy at 11-MAR-07

searching for all files in the recovery area
cataloging files...
no files cataloged


channel d1: starting datafile backupset restore
channel d1: specifying datafile(s) to restore from backup set
restoring datafile 00003 to +DG_DD501/clidbadr/datafile/sysaux.342.616944065
channel d1: reading from backup piece /apps/oracle/admin/CLIDBA/bkups/CLIDBA.3.1.616947691.DB
channel d2: starting datafile backupset restore
channel d2: specifying datafile(s) to restore from backup set
restoring datafile 00001 to +DG_DD501/clidbadr/datafile/system.340.616944047
```

```
restoring datafile 00004 to +DG_DD501/clidbadr/datafile/tools.344.616944625
channel d2: reading from backup piece /apps/oracle/admin/CLIDBA/bkups/CLIDBA.1.1.616947691.DB
channel d3: starting datafile backupset restore
channel d3: specifying datafile(s) to restore from backup set
restoring datafile 00002 to +DG_DD501/clidbadr/datafile/undo_rbs.341.616944059
restoring datafile 00005 to +DG_DD501/clidbadr/datafile/users.345.616944627
channel d3: reading from backup piece /apps/oracle/admin/CLIDBA/bkups/CLIDBA.2.1.616947691.DB
channel d1: restored backup piece 1
piece handle=/apps/oracle/admin/CLIDBA/bkups/CLIDBA.3.1.616947691.DB tag=CLIDBA_BACKUP_0Z_FULL_03112007
channel d1: restore complete, elapsed time: 00:00:26
channel d2: restored backup piece 1
piece handle=/apps/oracle/admin/CLIDBA/bkups/CLIDBA.1.1.616947691.DB tag=CLIDBA_BACKUP_0Z_FULL_03112007
channel d2: restore complete, elapsed time: 00:00:33
channel d3: restored backup piece 1
piece handle=/apps/oracle/admin/CLIDBA/bkups/CLIDBA.2.1.616947691.DB tag=CLIDBA_BACKUP_0Z_FULL_03112007
channel d3: restore complete, elapsed time: 00:00:34
Finished restore at 11-MAR-07


released channel: d1

released channel: d2

released channel: d3

released channel: d4

RMAN> **end-of-file**
```

Restore ARCHIVELOGS at the DR Site

First, review the RMAN backup logfile to determine the log sequences that need to be restored from each of the threads.

OPTIONAL STEP TO CHANGE ARCHIVE DESTINATION

If needed, change your archive destination at the DR site before initiating the restore process of archivelogs:

```
RMAN> run {set archivelog destination to '/apps/oracle/admin/CLIDBA/bkups';}


executing command: SET ARCHIVELOG DESTINATION
```

By restoring the archivelogs in advance, we can speed up the DR REDO APPLY process on the DR site by so that the FAL_CLIENT does not have to fetch these across the WAN.

```
CLIDBA > rman target /

Recovery Manager: Release 10.2.0.2.0 - Production on Sun Mar 11 23:25:24 2007
```

```
Copyright (c) 1982, 2005, Oracle.  All rights reserved.

connected to target database: CLIDBA (DBID=1936074589, not open)

RMAN> @dg_restore_archivelogs.sql

RMAN> restore archivelog from logseq=6 until logseq=9 thread=1 all;
Starting restore at 11-MAR-07
using target database control file instead of recovery catalog
allocated channel: ORA_DISK_1
channel ORA_DISK_1: sid=1082 devtype=DISK

channel ORA_DISK_1: starting archive log restore to default destination
channel ORA_DISK_1: restoring archive log
archive log thread=1 sequence=8
channel ORA_DISK_1: reading from backup piece
/apps/oracle/admin/CLIDBA/bkups/CLIDBA.7.1.616947715.ARC
channel ORA_DISK_1: restored backup piece 1
piece handle=/apps/oracle/admin/CLIDBA/bkups/CLIDBA.7.1.616947715.ARC
tag=TAG20070311T142154
channel ORA_DISK_1: restore complete, elapsed time: 00:00:02
channel ORA_DISK_1: starting archive log restore to default destination
channel ORA_DISK_1: restoring archive log
archive log thread=1 sequence=6
channel ORA_DISK_1: reading from backup piece
/apps/oracle/admin/CLIDBA/bkups/CLIDBA.8.1.616947715.ARC
channel ORA_DISK_1: restored backup piece 1
piece handle=/apps/oracle/admin/CLIDBA/bkups/CLIDBA.8.1.616947715.ARC
tag=TAG20070311T142154
channel ORA_DISK_1: restore complete, elapsed time: 00:00:02
channel ORA_DISK_1: starting archive log restore to default destination
channel ORA_DISK_1: restoring archive log
archive log thread=1 sequence=7
channel ORA_DISK_1: reading from backup piece
/apps/oracle/admin/CLIDBA/bkups/CLIDBA.9.1.616947715.ARC
channel ORA_DISK_1: restored backup piece 1
piece handle=/apps/oracle/admin/CLIDBA/bkups/CLIDBA.9.1.616947715.ARC
tag=TAG20070311T142154
channel ORA_DISK_1: restore complete, elapsed time: 00:00:02
Finished restore at 11-MAR-07

RMAN> **end-of-file**
```

Create TNSNAMES entries for FAL_CLIENT and FAL_SERVER

The following script will generate the tnsnames entries necessary to setup the fal_client and fal_server parameters. This is driven by by dg_master.conf configuration file and parses out the template file dg_tnsnames.txt:

http:///dba.fnfis.com/docs/dg/dg_tnsnames.txt

The script dg_tnsnames.ksh script will produce the following output:

./dg_tnsnames.ksh

```
CLIDBA_LTC =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = iltcdb708-vip.ngs.fnf.com)(PORT = 1525))
    )
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = CLIDBA)
    )
  )

CLIDBA_JTC =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = iltcdb408.ngs.fnf.com)(PORT = 1525))
    )
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = CLIDBA)
    )
  )
```

Copy this to the respective tnsnames.ora file on the primary server and the standby server.

Enable Flashback Recovery On Primary and Standby Database

```
CLIDBA > sqlp
SQL*Plus: Release 10.2.0.2.0 - Production on Sun Mar 11 17:36:40 2007
Copyright (c) 1982, 2005, Oracle.  All Rights Reserved.
Connected to:
Oracle Database 10g Enterprise Edition Release 10.2.0.2.0 - 64bit Production
With the Partitioning, Real Application Clusters, OLAP and Data Mining options

SQL> shutdown immediate;
Database closed.
Database dismounted.
ORACLE instance shut down.
```

```
SQL> SQL> start
SQL>
SQL>
SQL> startup mount;
ORACLE instance started.

Total System Global Area 1996488704 bytes
Fixed Size                  2072352 bytes
Variable Size             922747104 bytes
Database Buffers         1056964608 bytes
Redo Buffers               14704640 bytes
Database mounted.

SQL> alter database flashback on;

Database altered.

SQL> alter database open;

Database altered.

Exit and start the RAC database using srvctl
srvctl start database -d CLIDBA
```

RAC SPECIFICs

Register The RAC Data Guard Database Environment

As our standard, the database name will have the DR appended to the primary site $ORACLE_SID. Also, when we register the database, we use the PHYSICAL_STANDBY attribute as the database role with the default startup option of mount with the –n option of the primary site's $ORACLE_SID since the database name is different from the unique name given by the -d option.

```
srvctl add database -d CLIDBADR -o $ORACLE_HOME -r PHYSICAL_STANDBY -s mount -n CLIDBA
```

Register the instance with OCR

When we add the instance, we have to use the database name that has DR appended to the primary database's $ORACLE_SID.

```
FNFEBSP1 > srvctl add instance -d CLIDBADR -i CLIDBA1 -n iltcdb408
srvctl add instance -d CLIDBADR -i CLIDBA1 -n iltcdb408
```

Add Standby Redo Logs For Redo Log Synchronization From Primary

Standby redo logs are required for the maximum protection and maximum availability. The standby log file sizes need to be identical to the sizes of the on-line redo logs. Minimally, the configuration should have one more standby log file than the number of on-line redo log files.

Although standby redo logs are required for the maximum protection and maximum availability modes, it is also recommended for the maximum performance mode. The LGWR ASYNC transport mode is recommended for all databases. Data Guard can recover and apply more redo data from a standby redo log than from archived redo log files alone.

The following korn shell script will generate the SQL scripts to create the standby redo logs in the DG environment. By default, it will generate N+2 number of standby redo logs than what is on the primary database.

http://dba.fnfis.com/docs/dg/dg_create_standby_redo.ksh

The output of this script will look similar to the following:

```
alter database add standby logfile group 5 ('+DG_DD501/clidbadr/onlinelog/stdby_redo_05a') size
100m;
alter database add standby logfile group 6 ('+DG_DD501/clidbadr/onlinelog/stdby_redo_06a') size
100m;
alter database add standby logfile group 7 ('+DG_DD501/clidbadr/onlinelog/stdby_redo_07a') size
100m;
alter database add standby logfile group 8 ('+DG_DD501/clidbadr/onlinelog/stdby_redo_08a') size
100m;
alter database add standby logfile group 9 ('+DG_DD501/clidbadr/onlinelog/stdby_redo_09a') size
100m;
alter database add standby logfile group 10 ('+DG_DD501/clidbadr/onlinelog/stdby_redo_10a') size
100m;
```

The script examines the current size of the online redo logs and also counts the number of groups for thread 1 and creates the script accordingly. This script also sequentially increments from the highest group number from thread one also.

Couple of things to note about standby redo logs:

- The recommendation for the group # should sequentially follow the group numbers from the primary site
- The recommendation is to make n+1 standby redo groups on the DG database

  (maximum number of logfiles for each thread + 1) * maximum number of threads

Verify that the standby redo logs were created:

```
SQL> SELECT GROUP#,THREAD#,SEQUENCE#,ARCHIVED,STATUS FROM V$STANDBY_LOG;

    GROUP#    THREAD#  SEQUENCE# ARC STATUS
---------- ---------- ---------- --- ----------
```

```
         5          1          0 NO  UNASSIGNED
         6          1         12 YES ACTIVE
         7          0          0 YES UNASSIGNED
         8          0          0 YES UNASSIGNED
         9          0          0 YES UNASSIGNED
        10          0          0 YES UNASSIGNED

6 rows selected.
```

START AND MONITOR DATA GUARD

Finally, it is time to start Data Guard in managed recovery mode.

```
@dg_start
```

```
alter database recover managed standby database disconnect;
```

Check on Data Guard Processes

Once the DG is in managed standby, we need to monitor the RFS process to make sure archive logs are being shipped and applied in an appropriate timeline:

http://dba.fnfis.com/docs/dg/dg_stdby_proc.sql

```
SQL> @dg_stdby_proc
select pid, process, status, client_process, client_pid, thread#,
       sequence#, block#, blocks, delay_mins
from v$managed_standby
/

       PID PROCESS   STATUS       CLIENT_P CLIENT_PID      THREAD#  SEQUENCE#     BLOCK#     BLOCKS DELAY_MINS
---------- --------- ------------ -------- ------------ ---------- ---------- ---------- ---------- ----------
    561736 ARCH      CLOSING      ARCH     561736                1        138          1        303          0
   1146924 ARCH      CLOSING      ARCH     1146924               2         73          1        323          0
    782736 MRP0      WAIT_FOR_LOG N/A      N/A                   1        139          0          0          0
    738270 RFS       IDLE         UNKNOWN  288404                0          0          0          0          0
   1163362 RFS       IDLE         UNKNOWN  321426                0          0          0          0          0
    541262 RFS       IDLE         LGWR     567080                1        139      13652          1          0
   1048682 RFS       IDLE         LGWR     475416                2         74      11710          1          0

7 rows selected.
```

When this query is issued on the primary database, it provides information about ARCH and LNS processes. Respectively, when the query is issued on the DR database, it provides information about ARCH, MRP and RFS.

Couple of things to look out for:

1. Look out for big gaps in sequence# column for RFS and MRP0. This is an indication of degraded log apply services
2. Make sure that RFS and MRP0 is running

On the client side, we will need to enable the second archive destination:

```
alter system set log_archive_dest_state_2='enable' sid='*';
```

At the same time, the alert logs from both of the databases should be closely monitored for errors. Errors such as below on the primary site:

```
ORA-12514: TNS:listener does not currently know of service requested in connect descriptor
PING[ARC0]: Heartbeat failed to connect to standby 'CLIDBA_JTC'. Error is 12514.
```

should be reviewed and corrective actions to the TNSNAMES.ORA file should be made.

Occasionally, we may have to manually resolve archive gaps from the primary database and the DG database. We can detect for Archive Log Sequence Gaps:

```
@ck_gap.sql

select max(r.sequence#) last_seq_recd, max(l.sequence#) last_seq_sent from v$archived_log r, v$log l
where r.dest_id = 2 and l.archived= 'YES'
07:13:44 SQL> /

LAST_SEQ_RECD LAST_SEQ_SENT
------------- -------------
137           139
```

We can also query the v$dataguard_status view to obtain pertinent information related to media recovery on the DR site:

http://dba.fnfis.com/docs/dg/dg_status.sql

```
SQL> @dg_status.sql
SQL> col message for a65
SQL> set lines 255 pages 66
SQL>
SQL> select message, to_char(timestamp, 'DD-MON-RR HH24:MI:SS') timestamp, severity, facility
  2  from v$dataguard_status
  3  order by timestamp
  4  /

MESSAGE                                                          TIMESTAMP          SEVERITY      FACILITY
---------------------------------------------------------------- ------------------ ------------- --------------------
ARC0: Archival started                                           12-MAR-07 01:00:46 Informational Log Transport Services
ARC1: Archival started                                           12-MAR-07 01:00:46 Informational Log Transport Services
ARC0: Becoming the 'no FAL' ARCH                                 12-MAR-07 01:00:46 Informational Log Transport Services
ARC0: Becoming the 'no SRL' ARCH                                 12-MAR-07 01:00:46 Informational Log Transport Services
ARC1: Becoming the heartbeat ARCH                                12-MAR-07 01:00:46 Informational Log Transport Services
Attempt to start background Managed Standby Recovery process     12-MAR-07 01:00:53 Control       Log Apply Services
MRP0: Background Managed Standby Recovery process started        12-MAR-07 01:00:53 Control       Log Apply Services
Managed Standby Recovery starting Real Time Apply                12-MAR-07 01:00:58 Informational Log Apply Services
Redo Shipping Client Connected as PUBLIC                         12-MAR-07 01:01:18 Informational Log Apply Services
-- Connected User is Valid                                       12-MAR-07 01:01:18 Informational Log Apply Services
RFS[1]: Assigned to RFS process 725538                           12-MAR-07 01:01:18 Informational Remote File Server
RFS[1]: Identified database type as 'physical standby'           12-MAR-07 01:01:18 Informational Remote File Server

12 rows selected.
```

The v$archived_log view can provide detailed information about the status of the redo log apply service:

http://dba.fnfis.com/docs/dg/dg_exam_arc_logs.sql

```
SQL> @dg_exam_arc_logs.sql
SQL> set time on
16:14:33 SQL> set pagesize 9999
16:14:33 SQL> set lines 200
16:14:33 SQL> set num 9
16:14:33 SQL> col name format a78
16:14:33 SQL> col t# format 99
16:14:33 SQL> col s# format 9999
16:14:33 SQL> col applied format a7
16:14:33 SQL> select name, creator, thread# t#, sequence# s#, applied, first_change#, next_change#
16:14:33   2  from v$archived_log
16:14:33   3  where standby_dest = 'NO'
16:14:33   4  order by 3,4
16:14:33   5  ;

NAME                                                                            CREATOR  T#    S# APPLIED FIRST_CHANGE#
NEXT_CHANGE#
------------------------------------------------------------------------------- ------- --- ----- ------- ------------- -----------
--
+DG_DD501/clidbadr/archivelog/2007_03_12/thread_1_seq_5.638.616983797          ARCH      1     5 YES            159014
179177
+DG_DD501/clidbadr/archivelog/2007_03_11/thread_1_seq_6.621.616980337          SRMN      1     6 YES            179177
179182
+DG_DD501/clidbadr/archivelog/2007_03_11/thread_1_seq_7.622.616980339          SRMN      1     7 YES            179182
179187
+DG_DD501/clidbadr/archivelog/2007_03_11/thread_1_seq_8.620.616980335          SRMN      1     8 YES            179187
179212
+DG_DD501/clidbadr/archivelog/2007_03_12/thread_1_seq_9.639.616983827          ARCH      1     9 YES            179212
195755
+DG_DD501/clidbadr/archivelog/2007_03_12/thread_1_seq_10.637.616983735        ARCH      1    10 YES            195755
196078


6 rows selected.
```

APPENDIX D- RAC SERVICE MANAGEMENT

```
To add a service:
srvctl add service -d SNAPOC -s IMPORTER -r SNAPOC2 -a SNAPOC1

To start a service:
srvctl start service -s IMPORTER -d SNAPOC

Check on the existing service:
srvctl config service -s IMPORTER -d SNAPOC
IMPORTER PREF: SNAPOC2 AVAIL: SNAPOC1

SQL> show parameter service_name

NAME                                 TYPE        VALUE
------------------------------------ ----------- ------------------------------
service_names                        string      SNAPOC, QUERY
SQL> alter system set service_names='SNAPOC, QUERY, DBA, IMPORTER' scope=both;

System altered.

SQL> show parameter service_name

NAME                                 TYPE        VALUE
------------------------------------ ----------- ------------------------------
service_names                        string      SNAPOC, QUERY, DBA, IMPORTER
SQL> alter system register;

System altered.

Now it works:
srvctl stop service -s IMPORTER -d SNAPOC -i SNAPOC2 (you can only stop a
preferred instance)
srvctl stop service -s QUERY -d SNAPOC -i SNAPOC2   (if both of them are
preferred, you can stop either one)
```

The trick is to make both of them preferred:

```
srvctl add service -s DBA -r snapoc1,snapoc2 -d SNAPOC -P basic
```

-P option is failover mode (none,basic, preconnect)

```
srvctl start service -s DBA -d SNAPOC
```

The internal service EDOCPRD_RAC cannot be managed by srvctl.

```
SNAPOC1 > srvctl config service -s EDOCPRD_RAC -d EDOCPRD
PRKO-2120 : The internal database service EDOCPRD_RAC cannot be managed with srvctl.
iltcdb708.ngs.fnf.com:/apps/oracle
SNAPOC1 > srvctl stop service -s EDOCPRD_RAC -d EDOCPRD -i EDOCPRD1
PRKO-2120 : The internal database service EDOCPRD_RAC cannot be managed with srvctl.
```

APPENDIX E - INTERCONNECT INFORMATION

AWR Report Latency Name	Low	Typical	High
Average time to process cr block request	0.1	1	10
Average time to process cr block receive time (ms)	0.3	4	12
Average time to process current block request	0.1	3	23
Avg global cache current block receive time (ms)	0.3	8	30

In a RAC AWR report, there is a table in the RAC Statistics section containing average time (latencies) for some Global Cache Services and Global Enqueue Services operations.

The GES and GCS together maintain a global resource directory (GRD) to record information about resources and enqueues.

The GRD remains in the memory and is stored on all the instances. Each instance manages a portion of the directory. The distributed nature of the GRD is a key point for the fault tolerance of RAC. The GRD is an internal database that records and stores the current status of the data blocks. Whenever a block is transferred out of a local cache to another instance's cache, the GRD is updated. The following resource information is available in GRD:

1. Data Block Addresses (DBA). This is the address of the block being modified. Location of most current version of the data block. This exists only if multiple nodes share the data block.
2. Modes of the data blocks ((N)Null, (S)Shared, (X)Exclusive).
   The Roles of the data blocks (local or global). This indicates the role in which the data block is being held by the instance.
   SCN - System Change Number.
3. Image of the Data Block - it could be past image or current image. Current image represents the copy of the block held by the current instance. Past image represents the global dirty data block image maintained in the cache.

Latencies listed in the above table should be monitored over time, and significant increases in their value should be investigated. Factors that may cause variations to those latencies include:

1. Utilization of the IPC protocol. User-mode IPC protocols are faster
2. Scheduling delays, when the system is under high CPU utilization
3. Log flushes for current blocks served

Other RAC latencies in the AWR reports are mostly derived from V$GES_STATISTICS

APPENDIX F – SPECIAL THANKS AND CONTRIBUTING MEMBERS

White Paper prepared by:

Charles Kim

Oracle Certified DBA, Redhat Certified Technician, Microsoft Certified Professional

Chief Oracle Database Engineering Counsel

Charles has over 16 years of IT experience and has worked with Oracle since 1991. Charles is an author for Apress and a technical editor for Oracle Press. Charles has presented advanced topics for IOUG and Oracle Openworld on such topics as RAC/ASM and 7x24 High Availability Considerations using Oracle Advanced Replication, Hot Standby and Quest Shareplex.

Currently, Charles serves as the *Chief Oracle Database Engineering Counsel at Fidelity Information Services and* provides technical guidance and leads the Oracle SWAT team. Charles helps appraise/identify Oracle database operational deficiencies and infrastructure improvement needs and works with the Resource Manager of each Oracle database area to define and execute the remedy/mitigation plans. In addition, Charles provides the technical leadership in strengthening and standardizing the database infrastructure foundation across all FIS/FTS (Fidelity Technology Solutions) Oracle database areas.

Without the hard work and commitment of the instrumental personnel below, FNF would not have had the opportunity to leverage MAA as THE proven solution for the enterprise.

Executive IT Sponsor:

Mike Amble, Senior VP of Operations is responsible for operational service delivery. This includes the management of the Chicago, Jacksonville and Little Rock Technology Centers, and network engineering and implementation. Before joining Fidelity, Mr. Amble served as the CIO of GMAC Residential and was the Senior Vice-president of Corporate Technology for First American Real Estate Solutions.

Contributing DBA Members:

Allyson Wu, Oracle Certified DBA
Director of Enterprise Oracle Database Administration

Michael Powell, Oracle Certified DBA
Senior Database Engineer

Kim Khuu, Oracle Certified DBA
Senior Database Engineer

Ashley Nguyen, Oracle Certified DBA
Senior Database Administrator

Sang Rhee, Oracle Certified Associate
Senior Database Administrator

Additional thanks to our graphics designer:

Daniel Pitner, Web Architect

Edited by:

Leisa Pitner

INDEX

ADDM, 11, 57, 65, 74
AIX, 8, 12, 13, 14, 18, 22, 29, 56, 57, 68, 69, 70, 72
alert logs, 36, 102
ALIAS, 21
Archive logs, 33, 42
ARCHIVELOGS, 94
ASM, 9, 11, 12, 14, 15, 17, 18, 19, 20, 21, 29, 30, 33, 34, 36, 43, 60, 61, 62, 64, 65, 66, 67, 68, 69, 70, 75, 76, 77, 78, 79
ASM disks, 18
autoextensible files, 23
Automatic Storage Management. *See* : ASM
AWR, 17, 57, 65, 74, 105
baseline tables, 42
BCT. See : Block change tracking, See : Block change tracking
BLOB, 11, 25
Block change tracking file, 28
block size, 14
Brocade SAN Switches, 15
cache, 19, 69, 105
CLIDBA, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 102
CLOB, 19, 24, 30, 69
clone, 87
Clusterware, 12, 14, 19, 64
data block, 105
Data Block Addresses, 105
data blocks, 105
Data Guard, 6, 7, 14, 18, 33, 35, 36, 38, 39, 40, 41, 42, 43, 46, 52, 53, 56, 65, 67, 72
data loads, 11
DB Block Size, 14
DB2, 8, 9, 11, 12, 13, 69
Disaster Recovery, 18, 57
DR, 18, 33, 35, 36, 39, 42, 47, 53, 59, 65, 88, 90, 91, 92, 94, 98, 101, 102, See : Disaster Recovery
eDoc, 8, 9, 10, 11, 12, 13, 18, 19, 20, 23, 24, 25, 26, 28, 30, 39, 55, 57, 66, 72
EDoc, 8, 13
eDocument, 8
Enterprise Linux, 72
FAL client, 44
FAL_SERVER, 40, 44
Fast Application Notification, 17
Fast Connection Failover, 17
fault injection, 18
Federal Information Processing Standards codes, 24
File System Layout, 19
FIPS, 24, See : Federal Information Processing Standards, See : Federal Information Processing Standards, See : Federal Information Processing Standards, See : Federal Information Processing Standards
Flash Recovery Area, 19, 38
Flashback, 19, 20, 35, 36, 38, 39, 46, 49, 52, 53, 56, 68
Flashback Recovery, 96
force logging, 41, 42, 69
FRA, 19, 29, 39, 68
GCS, 74, 105
GES, 74, 105
GRD, 105
Grid Control, 6, 9, 18, 36, 56, 57, 58, 71, 73, 74
HACMP, 9, 18, 69, 70
HDS Universal Storage Processor, 15
Hitachi, 12, 14, 19, 27, 70
Hitachi TagmaStorage, 12
HP Proliant, 57, 72
IBM AIX, 12, 14, 36
IBM MPIO, 18, 19
IBM Websphere, 12, 13, 17
import, 11, 53, 79, 80
indexes, 27, 69
Informatica, 13, 24, 69
init.ora, 55, 80, 81, 82, 90, 91
J2EE, 13
JVM, 13, 17
latencies, 105
LGWR ASYNC, 99
Lock statistics, 30
LUN, 19, 61, 62, 63, 66
materialized views, 26, 42
MAXLOGFILES, 41
MAXLOGMEMBERS, 41
Naming Conventions, 20
National Institute of Standards and Technology, 24
NLS Character Set, 14
nologging, 42, 69
OCR, 17, 18, 66, 73, 75
OCR disks, 17
OMF. See : Oracle Managed Files, See : Oracle Managed Files, See : Oracle Managed Files, See : Oracle Managed Files
optimistic locking model, 9
Oracle Database 10g, 2, 12, 14

Oracle JDBC, 13
Oracle Managed Files, 23
partitioning, 9, 23, 24, 27
partitions, 23, 24, 30
persistency, 17, 21
pfile, 76, 80, 81, 82, 90, 91
QA, 11, 18, 36, 46, 47, 53, 57, 65
Quest Central, 57
Quest's Spotlight, 57
RAC, 7, 9, 11, 12, 13, 14, 16, 17, 18, 19, 20, 27, 28, 30, 36, 42, 53, 54, 55, 57, 59, 64, 65, 68, 69, 70, 71, 72, 75, 78, 79, 80, 81, 82, 83, 84, 97, 98, 104, 105
RAC Statistics, 105
RacPack, 18
RAID 1+0, 15, 20
RAID 5, 20
Raw devices, 17
Redhat 64-bit Linux, 72
REDO APPLY, 94
redo logs, 19, 35, 41, 45
registering a database backup, 87
remote_listeners, 17
response files, 12
restart the database, 52, 92
Restore, 48, 92, 93
RMAN, 27, 28, 29, 30, 31, 32, 33, 34, 36, 43, 53, 56, 67, 68, 71, 73, 79, 86, 87, 88, 90, 92, 93, 94, 95
SAN storage, 12
Sequence Gaps, 102
SID, 39, 54, 82, 91, 98
silent installations, 12
slave processes, 26, 60, 61
spfile, 80, 81, 82, 88, 89, 90, 91, 92
Standby Redo Logs, 99
Storage, 11, 12, 14, 15, 19
tablespace, 23, 30, 66, 69, 79, 80, 81, 82
Tablespace Strategy, 23
TCP, 17, 76, 83
threshold, 23, 53, 54
TNSNAMES, 17, 55, 82, 96, 102
Total SGA, 14
UDP, 17
Vault Tape Library, 28
Veritas Netbackup Advanced Client, 29
Virtual IP Addresses, 17
Voting disk, 18
Voting disks, 17
wildcard, 11
XML, 8, 13, 19, 23, 24, 30, 68