Design Specification

DocRef: TS/TR-07-4.0.6

Oracle Data Guard replication in Odyssey

Tim Read

21. February 2008

FBC: 2007/1612

Revision History

10

Version	***Comments***	***Date***	***Author***
0.1	Initial version	13. March 2007	Tim Read
0.2	Accumulated additions.	30. April 2007	
	Further changes in response to comments, plus removing some sections to appendices.		
	Finialising design ideas for review.		
	Incorporating feedback		Tim Read
0.3	Incorporating more feedback	1. May 2007	Tim Read
0.4	Update to incorporate design to align with current implementation and to reflect that it won't be released with Helen. Also changed Sun Cluster to Solaris Cluster through.	4. January 2008	Tim Read
0.5	Updates to incorporate comment on previous revision.	14. January 2008	Tim Read
0.6	Updates to incorporate comments. Mainly the need for a stop method to handle pmfadm properly. Added info on auto shadow RG and why it won't work. Incorporated previously missing feedback.	18. January 2008	Tim Read
0.7	Tweaks to resource status sections	15. February 2008	Tim Read
0.71	Removed 'Confidential' classification. Removed names	21. February 2008	Tim Read

Always update the revision history after making changes to this document.

Table of Contents

1. Project Description

The current Odyssey release can control two forms of data replication:

- Host-based, with Sun StorageTek AVS
- Storage-based, with Hitachi TrueCopy and EMC SRDF

This project will introduce support for a third model:

- Application-based, with Oracle Data Guard (ODG). It uses the Oracle Data Guard Broker interface (dgmgrl) rather than sqlplus command to control the configuration.

The design for the implementation to support this replication method is described in this document. Although the module is planned to be released with the fourth major Odyssey release, codenamed *Kirke*, the hooks required for it have been put back into the third release, codenamed *Helen*.

2. Replication with an Oracle Database

It is technically possible to support an Oracle database within an Odyssey partnership using the existing block-based replication technologies, subject to certain limitations imposed by the architecture, however there are disadvantages to doing this, as described below.

2.1. Disadvantages of block-based replication

Host-based replication (e.g. AVS) does not work with Oracle RAC, since the basis of RAC is that multiple hosts can directly access the same external storage simultaneously, with synchronization managed by Oracle. Since no single host sees all disk writes, there is no easy way to ensure that all writes get copied to the secondary site, with write ordering maintained. Veritas Volume Replicator (VVR, an equivalent to AVS) can do this when used with Veritas Cluster Server (VCS) by defining one host as a master, which apparently centralizes the management of the writes. The performance impact of this remains to be seen. For Solaris Cluster only Oracle HA can be used with host-based replication.

Storage-based replication overcomes this difficulty, and can work with RAC, as shown by the list of supported products at http://www.oracle.com/technology/deploy/availability/htdocs/vendors_remote_mirror.html however there are also disadvantages to this approach. Even a straightforward SQL query can result in changes to many tables within a database, affecting tens or even hundreds of disk blocks. Copying all of these blocks to the remote site will take time, and will require a high-capacity network link.

In addition to the performance issue, a block-copy operation cannot be atomic, since the replication software has no concept of a transaction that can be mapped onto the changes from a single SQL request. As a consequence, a system or network failure while the copy is in progress could leave the secondary database in an internally-inconsistent state. Careful management of replication, separating logs and data, is required to minimize this risk.

2.2. Application-based replication

With a transaction-based application like a DBMS, better performance and data integrity can be achieved by having the application itself perform the replication, which is the

basis behind ODG, http://www.oracle.com/technology/deploy/availability/htdocs/DataGuardOverview.html.

When Data Guard is active, each time the primary database receives an SQL request that modifies the database, this transaction request is stored in a Redo log, and copied to the secondary site, where it is applied to the secondary database. In this way the information passed between the sites is minimized, and since it is the transaction itself which is transferred, the atomicity of the operation is maintained.

2.3. Oracle Data Guard

Data Guard is the name given by Oracle to a software framework which allows one or more standby databases to be maintained as transactionally-consistent copies of a primary database. It is a standard part of the Enterprise Edition of Oracle Database.

Databases in an ODG configuration can be in one of two *database roles*: primary or standby. The standby can be created in one of two ways described below. The way in which the data is replicated is covered in the sub-section on “Replication mode“ below.

2.3.1. Physical Standby

In this mode, the standby database is dedicated to receiving updates from the primary, using what Oracle call “Redo Apply”. Updates are supplied as Redo log entries, and are applied directly to the physical database. The resulting database is physically identical in terms of on-disk structures to that on the primary.

A physical standby cannot be accessed while it is receiving updates. If access is required, for example to take reports, the database must be shut-down and restarted in read-only mode. When this is done the Redo log entries will accumulate, and will be applied later when the database is returned to standby mode.

During switch-over (i.e. role-swap between primary and standby) the use of a physical standby database requires that the primary be shut-down and restarted in standby mode. The secondary can be switched directly to primary if it has never been open read-only, otherwise it too must be shut-down and restarted as a primary database.

2.3.2. Logical Standby

In the logical standby mode, the updates from the primary are transferred as SQL (known, not unreasonably, as “SQL Apply”), essentially re-executing the request on the

secondary system. Although the information in the standby database will match that in the primary, the on-disk structure is not guaranteed to be the same. For all practical purposes this is irrelevant, and one advantage of this approach is that the standby database can be accessed for read-only queries and report generation while replication is ongoing.

A further advantage of the logical standby mode is that the operating mode of the database can be switched from primary to logical standby, and vice-versa, without requiring a full shut-down of either instance.

There have apparently been some reports of bugs in the logical standby mode of operation, and a number of Oracle customers still prefer to use physical standby.

2.3.3. Replication mode

Oracle can replicate data in one of three modes known as: maximum performance, maximum availability and maximum protection. These trade off recovery point objectives for database performance. Maximum protection offers zero data loss but with a performance cost while maximum performance offers higher performance but may loss data in a switch-over/fail-over.

2.4. Control of Data Guard

For RAC or single node configurations, ODG can be controlled and monitored via a distributed management framework known as Data Guard Broker. This has a graphical interface (through Oracle Enterprise Manager Grid Control) and a command-line interface (dgmgrl). The Data Guard broker interface (dgmgrl) allows a much simpler method for controlling such configurations. Instead of requiring multiple sqlplus commands to switch over site, a command of the form: “switchover to new_db_name” can be used. This also avoids having to have direct communication with remote databases as all this is managed by Data Guard broker.

Dgmgrl could be the main method of interaction between a Database Administrator (DBA) and the Oracle software. It also offers a logical point of control for Odyssey for the RAC and single node (or HA container/zone) configurations. Clearly the Broker offers a careless or malicious DBA the opportunity to override Odyssey control, and possibly disrupt Odyssey operation. Nevertheless, it would not be reasonable to prevent its use for general maintenance, etc.

For configurations using traditional Solaris Cluster HA-Oracle, dgmgrl cannot be used (see Oracle Metalink document 413696.1). Descriptions of the design that could be used for such a configuration are given in “Appendix 1 – unsupported options“.

When creating an Odyssey ODG configuration the user will be required to set up the ODG configuration in advance. The Odyssey module does not perform that set up for you. It does, however, check that when an ODG configuration is added to Odyssey that the configuration properties match those in the ODG configuration itself.

3. Odyssey / Oracle Data Guard Architectural Model

Odyssey extends the Solaris Cluster Resource model. Solaris Cluster combines resources into a Resource Group (RG) and manages the RGs within a Cluster by means of the Resource Group Manager (RGM). Odyssey combines one or more RGs with an instance of a Data Replication mechanism into a Protection Group (PG), and manages the PGs within a Partnership of two clusters, using the Inter-Cluster Resource Manager (ICRM). [1]

3.1. Protection Groups

A protection group will normally be composed of one or more Resource Groups (RGs) controlling the application, and a module to control the data replication subsystem. This model presumes that the application is independent of the replication system, which is clearly not the case when application-level replication is used.

3.1.1. Protection group states

Within an Odyssey PG, the RGs on one partner cluster (the Primary) will be `on-line`, and on the other (the Secondary) they will be `unmanaged`. Use of the `unmanaged` state is intended to ensure that if the Secondary cluster is rebooted the resources will not be brought on-line automatically by the cluster framework.

This creates a potential problem with application-level replication, since on the secondary cluster the application must be on-line, although in a (database) standby mode.

3.2. Oracle Data Guard replication model

The Odyssey model is a fail-over one, where a service is activated on the secondary site when the primary site is unavailable. For Data Guard it is important that Oracle is always running on both sites, and so it is not possible to have the actual database start/stop operation controlled by Odyssey. Instead, Odyssey will control which site is primary through dgmgrl, but will not manipulate the databases or the resource groups that contain them directly. Recall that the Oracle RAC server proxy resource groups themselves are controlled by the Solaris Cluster framework.

3.2.1. Oracle RAC

With Oracle 10g RAC, Oracle have introduced the Oracle Cluster Ready Services (CRS). There are advantages in using CRS to manage Oracle, and integrating CRS with Sun Cluster, and this has been addressed by the Oracle RAC Manageability project, FBC1413 [3].

Odyssey does not plan to alter the management approach described in this project. By adopting the approach defined in FBC1413, we retain compatibility with non-Odyssey RAC deployments, and simplify the management of Oracle for both the DBA and the Odyssey developers.

In this model the database will be started at boot time under the control of CRS, and Odyssey will use the ODG Broker, via the dgmgrl CLI, to control which database is the primary and which the secondary and also control the replication mode.

The module will support RAC in Solaris 10 containers [9] in a future release.

3.2.2. Data Guard replication module

In the Odyssey implementation, one component of a PG is the replication control module. This module is active within the ICRM on both the primary and secondary clusters. These two module instances cooperate to ensure that the replication mode is correct and consistent within the PG. This component will manifest itself as a Solaris Cluster (replication) resource group and one or more resources.

For Data Guard replication, a module will be developed that controls the database role (Primary, or Logical/Physical Standby) and the replication mode.

Normal Odyssey operations such as switch-over and take-over will be translated into Data Guard operations, with appropriate changes made to the database role.

3.2.3. Shadow Oracle Resource Groups

The second component of an Odyssey PG is one or more resource groups that represent the application resources using the replicated data. As highlighted previously, this cannot include an RG that directly references a RAC database because it must be on-line on both the primary and secondary clusters to allow replication to work.

To ensure that there is always at least one Oracle-specific RG in the PG, a Solaris Cluster GDS agent will be developed which will be deployed in a shadow Oracle

resource group. Because the initial release only supports Oracle RAC, the agent will only have start and probe methods.

The state of the shadow Oracle resource group will reflect whether a site is the primary or standby for a particular Oracle database. At the primary site, the shadow RG will be on-line, while at the second site it will be off-line and unmanaged. This corresponds to the behaviour observed in other application resource groups under AVS, SRDF or TrueCopy replication. Furthermore, the shadow Oracle RAC server proxy resource will reflect the status of the real Oracle RAC server proxy resource by simply using an “scha_resource_get -O status” call on each probe cycle. Failure to retrieve this will not be deemed critical as the real resource will still have the status information.

4. Design Details Overview

The section describes the design for supporting Oracle 10g RAC in more detail. Because HA-Oracle are cannot be supported for one reason or another, discussion of these has been moved to “Appendix 1 – unsupported options“.

The design has several key constraints and requirements:

- Multiple Oracle 10g RAC database must be supported in a single Odyssey configuration. These databases may be contained in one or more protection groups.
- The pairs of Oracle 10g RAC databases will be running on both Primary and Secondary clusters at all times, so have a degree of independence from Odyssey.
- The Odyssey replication control module will be used to modify the *database roles* (Primary or Standby) and the *replication mode* (Maximum performance, availability or protection).
- The design allows for only one standby Oracle database per Oracle primary database, whereas Oracle 10g itself supports up to nine Oracle standby databases.
- A (shadow) Solaris Cluster resource group and resource is created by the Odyssey Data Guard module for inclusion in the PG. This must manually be added to the PG. Failure to do so is not fatal, but will result in a less complete configuration and less obvious indications of which cluster is primary.
- The shadow RG will be created automatically by the ODG module with the identical node list, maximum and desired primaries to the RG it shadows.
- No two Oracle Data Guard configurations can have names that differ only by mapping '.' to '_', e.g. acme_west_com, acme_west.com and acme.west.com are all considered identical. This is a consequence of the Solaris Cluster CCR not allowing names with '.' in. Instead, these names must map '.' to '_' in order to use them as part of a CCR key field.
- The replication fail-over RG has a weak (+) affinity on the real (scalable) Oracle RAC server proxy RG. This is the reverse of AVS, TrueCopy or

SRDF configurations where the database would have a strong affinity on the underlying replication RG. This follows from the fact that these architectures are driven by the separate replication mechanism, rather than the integrated approach that ODG has. The weak affinity is simple there to provide some indication of which node the replication might be coming from.

- At some point, changes will be needed to *Kirke* to add support both to the CLI and BUI (browser interface) to recognise the fact that not all services will added entities of type “device group” to protection groups. Both the CLI and BUI will need to be changed to use a more generic name, e.g. replication object.

The Odyssey framework will be responsible, via the Data Guard mbean, for the creation and deletion of the additional resource groups and resources required for the integration.

4.1. Oracle 10g R2 RAC

Oracle 10g RAC software installation is complex, but once installed, the Solaris Cluster data services wizards make it relatively straightforward to configure the storage dependencies and the RAC proxy server resources. The additional Data Guard configuration also poses some challenges, but once set up, it can be controlled and managed through Data Guard broker (dgmgrl) or Oracle Enterprise Manager Grid Control.

The key resources and resource groups that comprise one end of such a Geographic Edition cluster configuration are shown in Illustration 1. Here the shadow Oracle resource group needs to be scalable, rather than fail-over, to reflect the nature of the Oracle proxy resource group used under Solaris Cluster 3.2. The shadow Oracle resource group must have the Auto_restart_on_new_cluster property set to false. This approach separates out the real Oracle RAC server proxy resource group from the protection group. The shadow resource group is unmanaged on the standby site without affecting the on-going replication being performed by the Oracle RAC database represented by the real Oracle RAC server proxy resources.

When the user adds an Oracle Data Guard configuration, e.g. acme.com, to a protection group (acme-pg), using 'geopg add-device ...' the odg_control script called from the ODG mbean will create:

- A replication resource group called acme-pg-odg-rep-rg, i.e. XX-odg-rep-rg (XX = pg name), if the RG does not exist already.
- A replication resource in acme-pg-odg-rep-rg called acme_com-odg-rep-rs, i.e. YY-odg-rep-rs (YY = odg_config_name)
- A shadow Oracle RAC server proxy resource group called acme_com-rac-proxy-svr-shadow-rg, i.e. YY-rac-proxy-svr-shadow-rg (YY = odg_config_name)
- A shadow Oracle RAC server proxy resource in the shadow Oracle RAC server proxy RG called acme_com-rac-proxy-svr-shadow-rs, i.e. YY-rac-proxy-svr-shadow-rs (YY = odg_config_name)

Conversely, when a user removes an Oracle Data Guard configuration the odg_control, via the ODG mbean, will remove the above entities with the exception of the <pg_name>-odg-rep-rg, if other <odg_config_name>-odg-rep-rs resources remain.

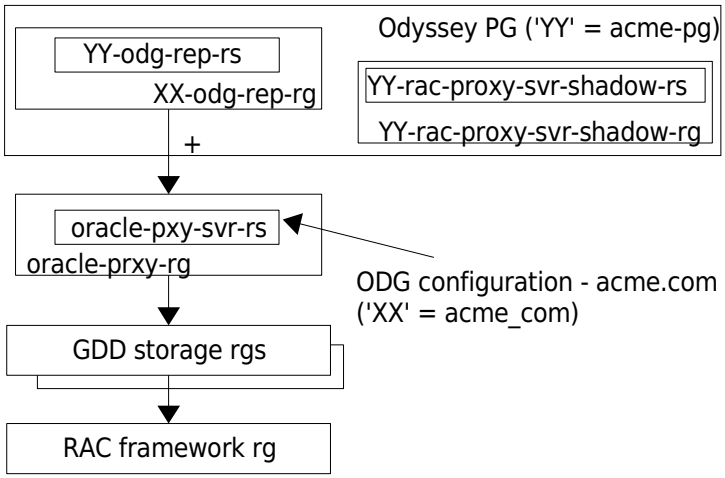

Illustration 1: Oracle RAC configurations with Data Guard

The acme_com-rac-proxy-svr-shadow-rg must be added to the protection group using a separate, manual configuration step that would be the analogous to adding a resource group (to be protected) to an Odyssey AVS configuration. This cannot currently be automated because of the internal event sequence of Odyssey. (An investigation into its feasibility resulted in event loops causing the configuration to be created and torn down in an endless cycle)

4.2. Data Guard broker (dgmgrl)

Before the operation of Data Guard Broker is covered it is important to understand how Oracle authenticates its users. This is covered below as background to the discussion on dgmgrl and how the password requirement it has is handled.

4.2.1. Oracle user and DBA authentication

An Oracle database can authenticate normal users using a number of mechanisms: the operating system, an external (or network) mechanism or through the database itself. Database administrators can only be authenticated either through the operating system using a Solaris group that Oracle recognises as being privileged, e.g. 'dba' or via an Oracle password file (this is not /etc/password).

When the using the operating system method, a DBA can simply log in with their Solaris user id and then connect to the database with:

```
% sqlplus '/ as sysdba'
```

The Solaris user id, e.g. oracle, would effectively be their point of authentication so long as they are in Solaris group dba. If the choose to log in using an Oracle database user name, e.g.

```
% sqlplus sys/oracle as sysdba
```

Then the password, in this example, 'oracle' is encrypted by Oracle using a modified AES algorithm and compares with the value held in the database for that user, i.e. sys. Only if they match are the users permitted to log in.

If the password is not specified on the command line, then it is prompted for by the program. The same is true of dgmgrl.

4.2.2. Accessing Data Guard Broker

ODG Broker simplifies the set up of Oracle primary/standby databases. Table 1-1 [4], for example, shows a comparison between dgmgrl and manual set up. However, it does have an number of requirements which are listed in section 2.2 [4].

Access to dgmgrl on the local node can be achieve by the Oracle user by issuing the command:

```
$ dgmgrl /
```

This is effectively using operating system authentication, the ORACLE_SID environment variable specifies which database instance is to be accessed and the program then connects **locally**. However this does not allow switch-over or take-overs to be performed because the operating system authentication is not propagated to the remote machine and thus it has insufficient privileges to perform the operation. The same problem can be demonstrated using sqlplus thus:

```
$ sqlplus /@<connect_string> as sysdba
```

Instead, the dgmgrl command **must** be issued with an Oracle database user name granted sysdba privileges together with a password and connect string (which resolves to an Oracle SID), thus:

```
$ dgmgrl sysdba_user/<passwd>@<connect_string>
```

This allows dgmgrl to authenticate any remote operations using the password file mechanisms. This is why Oracle has a requirement to keep these external password files synchronized (by the DBA copying them over when DBA password changes are made) between two such sites in a ODG configuration.

The difference between an ordinary Oracle database user and one with sysdba privileges is analogous to an ordinary Solaris user and the root user, the latter having privileges to do things like shutdown, mount, etc.

An Oracle SID is a name that uniquely identifies an Oracle database or instance. If an Oracle RAC database is called acme and it is installed on two nodes, the instances of that database will have SIDs: acme1 and acme2.

The output shown in Text 1: Sample dgmgrl commands and output below demonstrates running dgmgrl without any command line parameters. When prompted with the “DGMGRL>” a connection is made interactive through the “bar” connect string to the bar.sfbay database in the configuration (here the database has a domain qualifier on, but it doesn't have to have one).

A connection to dgmgrl restricts the scope of the Data Guard broker output to the configuration pertaining to the specific ORACLE_SID implied by the connection. As a result, the output to be parsed only contains primary and standby databases for this configuration.

Data Guard broker can co-ordinate the entire switch-over process including the shut-

```
DGMGRL> connect sys/oracle@bar
Connected.
DGMGRL> show configuration
Configuration
  Name:                bar.sfbay
  Enabled:             YES
  Protection Mode:     MaxPerformance
  Fast-Start Failover: DISABLED
  Databases:
    bar  - Physical standby database
    barb - Primary database

Current status for "bar.sfbay":
SUCCESS
DGMGRL> show database verbose bar ;
Database
  Name:            bar
  Role:            PHYSICAL STANDBY
  Enabled:         YES
  Intended State:  ONLINE
  Instance(s):
    bar

  Properties:
    InitialConnectIdentifier       = '(DESCRIPTION=(ADDRESS_LIST=(ADDRESS=(PROTOCOL=tcp)(HOST=oracle-
dg-zone)(PORT=1521)))(CONNECT_DATA=(SERVICE_NAME=bar.sfbay)(SERVER=DEDICATED)))'
    LogXptMode                     = 'ASYNC'
    Dependency                     = ''
   ... (deleted) ...
    LatestLog                      = '(monitor)'
    TopWaitEvents                  = '(monitor)'

Current status for "bar":
SUCCESS

DGMGRL> connect sys/oracle@bardr
Connected.
DGMGRL> show database verbose bar ;
Database
  Name:            bar
  Role:            PHYSICAL STANDBY
  Enabled:         YES
  Intended State:  ONLINE
  Instance(s):
    bar

  Properties:
    InitialConnectIdentifier       = '(DESCRIPTION=(ADDRESS_LIST=(ADDRESS=(PROTOCOL=tcp)(HOST=oracle-
dg-zone)(PORT=1521)))(CONNECT_DATA=(SERVICE_NAME=bar.sfbay)(SERVER=DEDICATED)))'
    LogXptMode                     = 'ASYNC'
    Dependency                     = ''
   ... (deleted) ...
    LatestLog                      = '(monitor)'
    TopWaitEvents                  = '(monitor)'

Current status for "bar":
SUCCESS
```

Text 1: Sample dgmgrl commands and output

down of existing primary site and start-up of the new primary. Consequently, the start method need only call the following:

```
% dgmgrl -silent << EOF
connect scgeo_dba/<dba_password>@<service_name>
switchover to <db_name>
EOF
```

The command should run without any unexpected errors within a preset Odyssey switch-over timeout. Because the return code of dgmgrl does not necessarily represent success or failure, a certain set of ORA-xxxxx errors are ignored as being normal output during switch-over and take-over. Other ORA-xxxxx errors are taken as exceptions and reported back.

Because dgmgrl requires a sysdba privileged Oracle database user name and password, Odyssey will require that these be supplied when adding an Oracle ODG configuration to a protection group. This necessitates extensions to the Odyssey framework to securely handle such passwords and minimise their exposure, at any time, during set-up or operation.

As long as the DBA password is never exposed on the command line by any routine called by the ODG mbean and it is not accessible to anyone who does not have root or Oracle privileges, then the process is no less secure than anyone operating this from the command line. For example, if a user has root access, they can simply 'su' to the Solaris Oracle user id and access the database and do as they wish – create new users, change passwords, etc.

4.2.3. Changes to Odyssey to support password property handling

The following section describes changes that have been put back into the *Helen* release of Odyssey to support password property handling, i.e. where the entity added to a protection group (e.g. an ODG configuration) requires a password property.

First, the passwords must be read in from the command line during the execution of the geopg command. Password properties for this and other, future ODG modules must conform to the pattern '*_password'. When the geopgi (a back-end program called by geopg) parse the protection group properties list, it looks for such parameters. If the password has been supplied in cleartext as, say, “...-p sysdba_password=foobar”, then it will warn the user that this is insecure, but continue with normal processing described below. For any password properties that have been specified like “... -p sysdba_password= -p other_password= ...”, then the geopgi program goes into non-echo mode and prompts for these passwords.

Once all the parameters have been processed, these pairs are written into an internal password file on the local node which is root readable only. A separate internalPasswordFile property is inserted onto the properties list with the value <host name>:<file name>.

Once in the core Odyssey Java code, this argument is unpacked and the file read remotely via an internal CACAO to CACAO call. The passwords are then obfuscated by converting the characters into the hexadecimal representation of their character codes before being written to the Solaris Cluster CCR, if the rest of the properties are correct and complete and the validation succeeds.

This means that the passwords are only available from the CCR with users possessing root access and are secure from casual users who may see the contents of the CCR displayed on the screen.

When required, the passwords can be queried and 'unofuscated' from the CCR and supplied to the appropriate programs to achieve the relevant switchovers, takeovers or status queries.

4.3. SUNW.gds ODG Replication Resource Type

The SUNW.gds replication resource type will monitor the state of the ODG replication using the Oracle dgmgrl command. This section considers the Oracle (ORA-xxxxx) error codes that dgmgrl might report and how the probe script should respond to them.

4.3.1. Data Guard broker replication status and failure modes

The status of an ODG configuration can be determined using the 'show configuration' and 'show database <db_name>' commands. The table below shows the range of Oracle error messages that may be encountered when a failure has occurred in the ODG system.

SUNW.gds probe scripts are run under hatimerun by the RGM to ensure that they either return a status or are timed out after a certain period. This means that if the dgmgrl command hangs for a long period of time the probe will be stopped. If, however, the dgmgrl “show configuration” returns a busy message, then the probe should retry the command to attempt and get an indication of the status.

Given the huge range of potential Oracle failure scenarios, the Odyssey module will make no attempt to recover them. Indeed, it is likely that any attempt to rectify the

problem may conflict with work being done by the Oracle DBA, either through dgmgrl or via sqlplus. So, instead the module will simply reflect the current messages presented by dgmgrl through the Solaris Cluster Geographic Edition GUI and CLI interfaces. The table below gives an indication of the range of messages that can be received.

Failure event	***Dgmgrl output***
Invalid Oracle sysdba account password	ORA-01017: Invalid username/password; Logon denied.
Oracle listener shut-down	Unable to connect to database ORA-12541: TNS: no listener
Oracle RAC start-up	During start-up dgmgrl may report one or more of the errors below as it resolves which database should be primary and which standby: ORA-16525: the Data Guard broker is not available yet ORA-01089: immediate shutdown in progress – no operations are permitted ORA-03135: connection lost contact ORA-03114: Not connected to ORACLE ORA-16607: one or more databases have failed
Switch-over	During switch-over dgmgrl may variously report: ORA-03135: connection lost contact ORA-16607: one or more databases have failed
After fail-over	Primary site may report: ORA-16795: database resource guard detects that database re-creation is required ORA-16661: the standby database needs to be reinstated ORA-16653: failed to reinstate database

When a Data Guard configuration exists, it can be in one of two states: disabled or
410 enabled. Databases within the configuration can also be enabled or disabled. The state diagrams below show how configurations are created and destroyed.

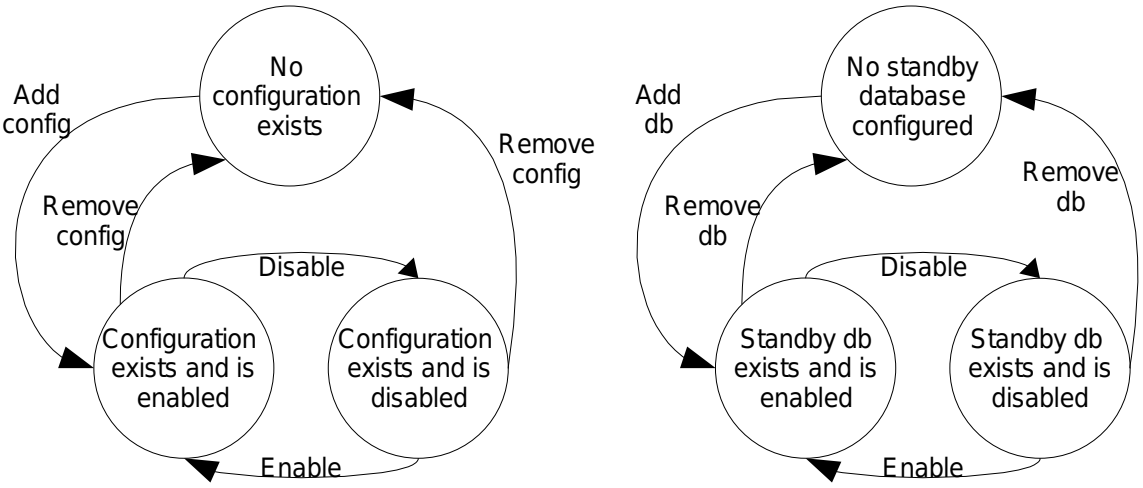

Illustration 2: Possibly states for Data Guard broker configuration

4.3.2. Resource status

The status of the SUNW.gds replication resource will reflect the aggregate status of the Data Guard broker configuration. For configurations where the roles of the configured databases match those expected by Odyssey, the status messages for each set of states is given in the table below. Because the resource only monitors the database it will only have the status of degraded or online/OK.

In the examples in the table below, foo is the primary database, foodr the standby, sys is the sysdba privileged username to use and each database has a connect string identical to their individual names.

The resource status only changes when the probe executes, unlike the Oracle RAC server proxy resource that gets updates triggered by the Oracle FAN mechanism. This approach has to be used because there is no FAN interface to ODG changes. The resource status is generated from the return code (see 5.3.2 Error conditions below) of the probe routine.

Code	***Resource Status***
0 (zero)	OK
E_ODG_PROGRAM_FAILED_TO_READ_CCR	Degraded
E_ODG_FAILED_TO_GET_SYSDBA_PASSWD	Degraded
E_ODG_LOCAL_SITE_NOT_PRIMARY	Degraded

Code	*Resource Status*
E_ODG_DATABASE_ROLE_DOES_NOT_MATCH_ROLE_IN_ODG	Degraded
E_ODG_CONFIG_NAME_MISMATCH	Degraded
E_ODG_DATABASE_IS_DISABLED	Degraded
E_ODG_SITE_DISABLED	Degraded
E_ODG_ODG_CONFIGURATION_DISABLED	Degraded
E_ODG_NO_SUCH_FILE	Unknown
E_ODG_UNEXPECTED_ERROR	Unknown
E_ODG_INTERRUPTED_OR_TIMED_OUT	Unknown
E_ODG_WRONG_PASSWORD_OR_CONNECT_NAME	Unknown
E_ODG_ORACLE_SID_NOT_FOUND	Faulted
E_ODG_PROGRAM_EXITED_NON_ZERO	Faulted
E_ODG_PROTECTION_MODE_MISMATCH	Faulted
E_ODG_DATABASE_NOT_FOUND_IN_ODG_CONFIG	Faulted
E_ODG_OERR_FROM_DGMGRL	Faulted

The following table below shows the status message that will be returned under other erroneous conditions (generally, E_ODG_OERR_FROM_DGMGRL). In most cases these result in a 'Faulted' status. The list is not exhaustive and, in general, the status message will reflect the Oracle errors found in the dgmgrl output.

Condition	*Resource status message*
Odyssey primary/secondary state doesn't match ODG dgmgrl status	Role "primary database" given for database foo does not match role "physical standby database" configured Oracle Data Guard.
During switch-over	Oracle errors "ORA-16534: no more requests accepted" were found in the Oracle Data Guard broker (dgmgrl) output when connecting using "sys/<password>@foo" or Oracle errors "ORA-16625: cannot reach the database" were found in the Oracle Data Guard broker (dgmgrl) output when connecting using "sys/<password>@foo" or Oracle errors "ORA-12537: TNS:connection closed" were found in the Oracle Data Guard broker (dgmgrl) output when connecting using "sys/<password>@foo" or

	Oracle errors "ORA-12514: TNS:listener does not currently know of service requested in connect descriptor" were found in the Oracle Data Guard broker (dgmgrl) output when connecting using "sys/<password>@foo" or The command or probe was interrupted or timed out. (This is due to dgmgrl being unresponsive during switchover)
Incorrect password configured in Odyssey for sysdba_password	Oracle errors "ORA-01017: invalid username/password; logon denied" were found in the Oracle Data Guard broker (dgmgrl) output when connecting using "sys/<password>@foo" or Oracle errors "ORA-16625: cannot reach the database" were found in the Oracle Data Guard broker (dgmgrl) output when connecting using "sys/<password>@foodr" (where the password are not synchronized between sites)
dgmgrl was performing health checks and these did not complete before the probe was stopped by hatimerun.	Oracle errors "ORA-16610: command 'Broker automatic health check' in progress" were found in the Oracle Data Guard broker (dgmgrl) output when connecting using "sys/<password>@foo"

4.3.3. Recovery scenarios

Although the validation mechanism will prevent an invalid configuration being configured, it is possible for the Oracle DBA to make changes outside Odyssey that will invalidate the configuration. The following sections describe how to recover from these configuration issues.

4.3.3.1. Configuration has been removed

The identical configuration must be re-created via dgmgrl by the Oracle DBA. Once the configuration is complete, the probe will detect it has been re-instated. The SUNW.gds replication resource will return to the OK state.

4.3.3.2. Primary, standby or configuration has been disabled

The item that has been disabled must be re-enabled through dgmgrl. Once complete and the status according to dgmgrl is 'SUCCESS', the SUNW.gds replication resource will return to the OK state.

4.3.3.3. Standby database has been removed from the configuration

The standby database must be reconfigured through dgmgrl. Once complete and the status according to dgmgrl is 'SUCCESS', the SUNW.gds replication resource will return to the OK state.

4.3.3.4. Replication stops due to networking problems

If network problems prevent replication and/or dgmgrl from acquiring the status of the configuration, dgmgrl may give a status other than 'SUCCESS'. To rectify the situation, any networking issue must resolved together with any steps needed to re-enable the replication from within dgmgrl. Once complete and the status according to dgmgrl is 'SUCCESS', the SUNW.gds replication resource will return to the OK state.

4.3.3.5. Database roles do not match the Odyssey configuration

The administrator can either perform a switch-over within dgmgrl to bring the configuration back into alignment, or more likely, perform a switch-over (or take-over) from within Odyssey to bring the configurations into alignment. This operation will result in a NO-OP at the Data Guard level.

4.3.3.6. Primary or secondary site has failed

The ODG configuration should indicate a fault from the remaining site. When the failed site is back on-line again, the appropriate component should be re-enabled via dgmgrl if required. Once complete and the status according to dgmgrl is 'SUCCESS', the SUNW.gds replication resource will return to the OK state.

4.3.3.7. Invalid Data Guard user name or password

Either the Odyssey ODG configuration must be brought into line with the Oracle ODG configuration or vice versa. If the Odyssey configuration is to be updated, then the “geopg modify-device-group -p sysdba_password= -p sysdba_username=XXX <odg_config_name> <pg_name>” command should be issued. Alternatively, the password or account can be modified within Oracle via sqlplus.

4.3.4. Resource callback methods

The SUNW.gds replication resource type defines the start, stop and probe methods.

4.3.4.1. Start

The start method does nothing apart from stopping pmf taking any action when the last process in the process tree, referenced by the tag (${resource_group},${resource},0.svc), disappears. This resource will leave no processes to monitor after it exits. This will be done using:

```
/bin/sleep ${start_timeout:-300} &
/usr/cluster/bin/pmfadm -s \
     ${resource_group},${resource},0.svc
```

The probe command below must return 0 for the service to start correctly. If the probe does not return 0, it will be retried after 2 seconds until it either returns 0 or start_timeout is exceeded. If the start does fail, the service will be restarted again.

4.3.4.2. Stop

The stop method simply stops the process monitoring to avoid any error messages that might arise from someone disabling the resource while the sleep is still running.

```
/usr/cluster/bin/pmfadm -s \
     ${resource_group},${resource},0.svc KILL
```

4.3.4.3. Probe

The probe will check the current Oracle Data Guard configuration effectively using:

```
% ( /tmp/.GeoInput.$$ ; /bin/rm /tmp/.GeoInput.$$ ) \
     | dgmgrl -silent  > /tmp/.GeoOutput.$$
Where /tmp/.GeoInput.$$ contains
connect <sysdba_username>/<password>@<connect_string>
"show configuration"
```

The .GeoInput file will only be readable by the Oracle user as it will contain the sysdba_password in clear text. The input file is deleted immediately to avoid the password being stored anywhere in clear text for longer than is needed. The probe script will have extracted the sysdba_username and sysdba_password from the Solaris Cluster CCR by way of an API script to return the values of these fields. This binary will be root executable only. Because it uses an API script to query the CCR directly, it does not require Odyssey to be running for the resource probe to be able to query dgmgrl. Recall that the databases run independently of Odyssey's direct control of their start and

stop methods.

To ensure that the output is always in the expected language and format, all calls to dgmgrl will be made with the Oracle NLS_LANG environment variable set to AMERICAN_AMERICA.US7ASCII (the format being language_territory.charset).

The probe will make the following checks of the ODG configuration from the output gathered from “show configuration”

- If the overall status is “SUCCESS” or if the site is disabled
- If the local database role and type is the one expected by Odyssey
- If the remote database is enabled

The probe will log messages to syslog.

4.4. SUNW.gds shadow Oracle RAC proxy resource type

Resources of this type are responsible for interfacing with the 'real' Oracle server resources that are active in the configuration. In the initial release, this will be SUNW.oracle_rac_server_proxy resources, but in later releases may also include SUNW.oracle_server resources if the configuration described in the section “HA-Oracle in an HA Solaris 10 container“ below are supported by Oracle. The SUNW.gds shadow RAC proxy design should be flexible enough to accommodate this change in later releases.

Because this resource is only a 'shadow' of the 'real' resource, i.e. Something that can be transitioned from on-line to unmanaged as the direction of replication is changed, the resource has almost no functionality. It exists simply to have its state changed and reflect the status of the 'real' resource and add some additional, useful status information.

4.4.1. Resource callback methods

Again, since this is a SUNW.gds resource, only the start, stop and probe methods have been supplied. These methods will have the following functionality.

4.4.1.1. Start

This uses the same method as the ODG replication resource type in section 4.3.4.1 above:

```
/usr/cluster/bin/pmfadm -s \
     ${resource_group},${resource},0.svc
```

4.4.1.2. Stop

This uses the same method as the ODG replication resource type in section 4.3.4.2 above:

```
/usr/cluster/bin/pmfadm -s \
     ${resource_group},${resource},0.svc KILL
```

4.4.1.3. Probe

The probe will always exit 0 and will set the resource status to that of the RAC proxy resource that it shadows. It will augment this with information about the site being data guard primary.

5. Oracle Data Guard (ODG) Mbean design

The ODG data replication mbean implements the generic data replication mbean interface (GeoDataReplicationMBean.java). Its functionality will include:

- Creating the shadow Oracle resource resource group and resource when the Oracle Data Guard configuration is added to the protection group. This resource group will then have to be added to the protection group in additional, manual step.
- Creating the replication resource group, if required, and resource when the ODG configuration is added to the protection group.
- Supporting protection group activation/de-activation/migration through the ODG broker command, dgmgrl. The local invocation of dgmgrl handles all the necessary communication between the Oracle databases.
- Retrieving the status of the ODG data replication mbean to reflect the aggregated data replication state of the ODG configuration under the control of the protection group. The ODG data replication mbean posts event notification about data replication status change.

Because Oracle controls both the database and the replication mechanism there is no need for the ODG Mbean to perform any additional coordination between stopping/starting the database and stopping/starting the replication.

5.1. ODG Mbean properties

The ODG Mbean will have the following properties:

Property name	***Description***	***Type***
local_database_name	Local database name	Local, required
remote_database_name	Remote database name	Local, required
local_db_service_name	Local database connect string or service name	Local, required
remote_db_service_name	Remote database connect string or service name	Local, required
local_rac_proxy_svr_rg_name	Local Oracle RAC server proxy resource group name for the RG containing the 'real' Oracle RAC server proxy resource	Local, required
remote_rac_proxy_svr_rg_name	Remote Oracle RAC server proxy resource group name for the RG containing the 'real' Oracle RAC	Local, required

	server proxy resource	
replication_mode	Replication mode, either maxperformance, maxavailability or maxprotection	Global, required
standby_type	Database standby type, either logical or physical	Global, required
sysdba_username	An Oracle sysdba privileged username that can be used with dgmgrl to probe the status of the ODG configuration	Global, required
sysdba_password	The password for the sysdba username specified in the sysdba_username property (Note, this is never displayed in any output from geopg, etc)	Global, required

Internally, the ODG Mbean will have the following property:

Property name	***Description***	***Type***
real_ODG_name	This is the real ODG configuration name without any '.' mapped to '_'. This is used for display purposes through the program.	Global, required

These objects will be created, modified and destroyed as ODG configurations are manipulated.

5.2. ODG Replication Resource Group and Resource Configuration

ODG mbean will monitor the ODG replication through the replication and shadow Oracle resource group and resources. State change events are propagated from CMAS and handled in the mbean to reflect that status. The mbean will also control the direction of
570 replication by calling out to shell script that will interact with the Oracle dgmgrl program allowing it to issue the relevant switchover and failover (takeover) commands.

The mbean will create the replication resource group and resource together with the shadow Oracle RAC proxy server resource group and resource when the ODG configuration is added to the protection group.

To an attempt to satisfy RFE CR 6206766, the naming of the resource groups and resources should conform to the following convention, given a protection group named <PGName> and an ODG configuration called <ODG_name>:

- The replication resource group will be called <PGName>-odg-rep-rg. This is

consistent with the approach used by the AVS module.

- The replication resource will be called <ODG_name>-odg-rep-rs.
- The Oracle RAC proxy server shadow resource group will be called <ODG_name>-rac-proxy-svr-shadow-rg
- The Oracle RAC proxy server shadow resource will be called <ODG_name>-rac-proxy-svr-shadow-rs

If the ODG name contains any '.' characters, e.g. sales.acme.com, these will be mapped to '_' to give sales_acme_com. This makes the names acceptable to the CCR.

5.3. GeoDataReplicationMbean Methods Overridden

The ODG data replication mbean will override the following methods, defined by the generic data replication mbean interface (GeoDataReplicationMbean.java):

- initParameter – initialization work, registering with event listener
- cleanup – deinitialization work, e.g. deleting listener when PG is remove.
- getReplicationType – returns the replication type, i.e. ODG
- applicationResGroupAdded – will call checkApplicationResGroup.
- checkApplicationResGroupToAdd - will call checkApplicationResGroup.
- checkApplicationResGroup – check that the resource group added does not contain a RAC server proxy resource. Calls out to the odg_control shell script.
- remapDeviceGroupName – converts Device Group names to value ones, e.g. sales.acme.com to sales_acme_com
- createAdditionalProperties – create additional (internal) properties, such as real_ODG_name which might be 'sales.acme.com'.
- remapToDataReplicationSpecificMessage – take generic messages and remap them so that the say '... Oracle Data Guard ...' rather than '... Device Group ...'. This function maintains the internationalization of the messages.
- getDeviceGroupDisplayName – returns the internal real_ODG_name property.

- getDisplayableProperties – returns the displayable properties, hiding the real_ODG_name property and giving '******' for the sysdba_password property value.
- getDataReplicationTextDomain – returns Globals.I18_ODG
- modifyDeviceGroup – checks and modifies an ODG configuration. Calls out to the odg_control shell script.
- createDeviceGroup – checks and creates an ODG configuration held. Calls out to the odg_control shell script.
- isPublicProperty – returns true.
- isMappedLocalProperty – is the mapped property local or not
- getInternalRGs – returns the combined list of the replication RG and shadow Oracle RAC proxy server resource groups for this PG.
- getStatus – returns the overall status of the mbean
- deleteDeviceGroup – checks and removes an ODG configuration. Calls out to the odg_control shell script.
- startReplication – tries to bring PG online with the current role. Calls out to the odg_control shell script.
- stopReplication – tries to bring PG offline. Calls out to the odg_control shell script.
- createConfiguration – No OP.
- initConfiguration – creates an ODG configuration held. Calls out to the odg_control shell script.
- removeConfiguration - removes an ODG configuration. Calls out to the odg_control shell script.
- checkAndApplyConfiguration - creates an ODG configuration held. Calls out to the odg_control shell script.
- updateProperties – No OP for now as there are no updatable properties specific to an ODG protection group.
- checkBeforeRoleChange – checks before a role change. Calls out to the

odg_control shell script.

- changeRole – changes role. Calls out to the odg_control shell script.
- getDefaultPropertyValue – supplies sensible defaults for some of the properties, i.e. replication_mode = “maxperformance”, standby_type = “physical”

The following method will not be overridden as there is no additional work to do when removing the shadow Oracle RAC proxy server resource group.

- applicationResGroupRemoved

5.3.1. Operation characteristics

All operations must be idempotent. The same operation, when performed multiple times, should always return the same state, unless some other operation is performed which has changed the state:

state-A -> <operation X with parameters x> -> state-B

state-B -> <operation X with parameters x> -> state-B

where state-A is a starting state and state-B is the state where the operation completes successfully.

If the operation fails, leaving the object in state-B1, rather than state-B:

state-A -> <operation X with parameters x> -> state-B1

One of the following could happen:

- If no other operation is performed that has changed the error condition, the same operation should not change its state:

state-B1 -> <operation X with parameters x> -> state-B1

- If some other operation is performed that has changed the error condition:

state-B1 -> <operation Y > ->state-B2

The operation may be completed now by either the same operation which starts the job:

state-B2 -> <operation X with parameters x > ->state-B

or by a different operation:

state-B2 -> <operation Z > ->state-B

For example, the changeRole operation has failed, leaving the PG in a state with an inconsistent role configuration. Repeat the same operation will fail with the same error until the configuration is revalidated with checkAndApplyConfiguration operation.

5.3.2. Error conditions

The ODG Mbean can raise any of the following error conditions:

N°	*Code*	*Message*
101	E_ODG_PROGRAM_FAILED_TO_READ_CCR	Program {0} failed to read the cluster configuration repository (CCR)
102	E_ODG_NO_RESOURCE_OF_THIS_TYPE_FOUND	No {0} resource exists in resource group {1}.
103	E_ODG_CAN_NOT_CREATE_SHADOW_RG	Unable to create shadow Oracle RAC server proxy resource group {0}.
104	E_ODG_RESOURCE_DOES_NOT_MATCH_EXPECTED_TYPE	esource {0} is not of expected type - \"{1}\".
105	E_ODG_UNABLE_TO_DETERMINE_RS_PROPERTY	Unable to determine resource property {0} for resource {1}.
106	E_ODG_UNABLE_TO_REGISTER_RES_TYPE	Unable to register resource type (0}.
107	E_ODG_UNABLE_TO_CREATE_RESOURCE	Unable to create resource {0} of type {1} in resource group {2}.
108	E_ODG_NO_SUCH_FILE	File {0} does not exist.
109	E_ODG_ORACLE_SID_NOT_FOUND	Unable to determine the ORACLE_SID for database {0} in ORACLE_HOME {1} on node {2}.
110	E_ODG_PROGRAM_EXITED_NON_ZERO	Program {0} returned a non-zero exit code.
111	E_ODG_PROTECTION_MODE_MISMATCH	Protection mode "{0}" given for local database {1} does not match configured value "{2}".
112	E_ODG_UNEXPECTED_ERROR	nexpected error - {0}.
113	E_ODG_DATABASE_NOT_FOUND_IN_ODG_CONFIG	Database {0} does not exist in the configured Oracle Data Guard database list "{1}".

N°	Code	Message
114	E_ODG_DATABASE_ROLE_DOES_NOT_MATCH_ROLE_IN_ODG	Role "{0}" given for database {1} does not match role "{2}" configured Oracle Data Guard.
115	E_ODG_FAILED_TO_GET_SYSDBA_PASSWD	Failed to get password for sysdba username for Oracle Data Guard configuration {0} in protection group {1}.
116	E_ODG_CAN_NOT_REMOVE_SHADOW_RG	Unable to remove shadow Oracle RAC server proxy resource group {0}.
117	E_ODG_UNEXPECTED_RESOURCES_FOUND_IN_SHADOW_RG	Unexpected resources "{0}" found in shadow RAC proxy server resource group {1}.\nDo not place any additional resources in this resource group
118	E_ODG_CAN_NOT_GET_RG_NODELIST	Unable to retrieve nodelist for resource group {0}.
119	E_ODG_LOCAL_SITE_NOT_PRIMARY	Local cluster {0} is not primary for Oracle Data Guard configuration {1}.
120	E_ODG_RS_IN_REP_RG_NOT_DELETED	Resource {0} in replication resource group {1} is not deleted.
122	E_ODG_UNABLE_TO_CREATE_OR_UPDATE_REP_RG	Unable to create or update resource group {0} where RG_affinities={1}.
123	E_ODG_FAILED_TO_DISABLE_RESOURCE	Failed to disable resource {0} in resource group {1}.
124	E_ODG_FAILED_TO_DELETE_RESOURCE	Failed to delete resource {0} in resource group {1}.
125	E_ODG_FAILED_TO_ONLINE_RG	Failed to bring resource group {0} online.
126	E_ODG_CONFIG_NAME_MISMATCH	Oracle Data Guard configuration name {0} found does not match {1}.
127	E_ODG_UNABLE_TO_DELETE_RG	Failed to delete resource {0} in resource group {1}.
129	E_ODG_OERR_FROM_DGMGRL	Oracle errors "{0}" were found in the Oracle Data Guard broker (dgmgrl) output when connecting using "{1}".
130	E_ODG_INCONSISTENT_ROLE	Role not configured consistently for protection group {0}.
131	E_ODG_RG_DOES_NOT_EXIST	Resource group {0} does not exist.

N°	Code	Message
132	E_ODG_DATABASE_IS_DISABLED	Database {0} is in the disabled state.
133	E_ODG_FAILED_TO_ENABLE_RESOURCE	Failed to enable resource {0} in resource group {1}.
134	E_ODG_FAILED_TO_MANAGE_RESOURCE_GROUP	Failed to manage resource group {0}.
135	E_ODG_SWITCHOVER_FAILED	Switchover to database {0} in protection group {1} failed.
136	E_ODG_INTERRUPTED_OR_TIMED_OUT	The command or probe was interrupted or timed out.
137	_ODG_NOT_A_CLUSTER_MEMBER	{0} is not a member of cluster {1}.
138	E_ODG_DUPLICATE_NODE_NAME	{0} is specified more than once.
139	E_ODG_RAC_PROXY_SVR_RG_NOT_ALLOWED	Resource group {0} contains a SUNW.scalable_rac_server_proxy resource ({1}) and is not allowed to be added to the protection group.
140	E_ODG_SITE_DISABLED	Oracle Data Guard configuration {0} is disabled on cluster {1}.
141	E_ODG_UNABLE_TO_CREATE_ODG_CONFIG	Unable to create Oracle Data Guard configuration {0}.
142	E_ODG_UNABLE_TO_MODIFY_ODG_CONFIG	Unable to modify Oracle Data Guard configuration {0}.
143	E_ODG_UNABLE_TO_DELETE_DG	Unable to delete Oracle Data Guard configuration {0}.
144	E_ODG_UNABLE_TO_CREATE_PROPERTY	Unable to create property {0}.
146	E_ODG_UNABLE_TO_UPDATE_GLOBAL_PROPERTY	Unable to update property {0}.
148	E_ODG_UNABLE_TO_UPDATE_LOCAL_PROPERTY	Unable to update property {0}.
150	E_ODG_UNABLE_TO_GET_PROPERTY	Unable to retrieve property.
151	E_ODG_UNABLE_TO_GET_CLUSTER_NODELIST	Unable to get cluster nodelist.
152	E_ODG_WRONG_PASSWORD_OR_CONNECT_NAME	Password or connect name ({0}) for remote site is incorrect.
154	E_ODG_ODG_CONFIGURATION_DISABLED	Oracle Data Guard configuration {0} is disabled.
158	E_ODG_APP_RG_DEPEND_ON_RG_OUTSIDE_PG	Resource groups in protection group {0} must not have an inter-resource group dependency on resource group {1}, which is not in the same protection group.

N°	Code	Message
160	E_ODG_SERVER_REQUEST_FAILED_DUE_TO_TIMEOUT	Error in running control script on host {0}. \nOperation timed out after {1} seconds.
162	E_ODG_SERVER_REQUEST_FAILED_WITH_REASON	Error in running control script on host {0} due to system error - \n\t\"{1}\"
200	E_ODG_CONFIG_ERROR	Configuration error detected for propection group {0}.
210	E_ODG_INVALID_PROPERTY_FILE	Invalid property file {0}.
221	E_ODG_MISSING_PROPERTY	Property {0} is not set.
222	E_ODG_DUPLICATE_PROPERTY	Duplicate property {0}.
223	E_ODG_INVALID_PROPERTY	Invalid property {0}.
224	E_ODG_INVALID_PROPERTY_VALUE	Invalid value for property {0}.
225	E_ODG_ODG_CONFIG_ALREADY_IN_PG	Oracle Data Guard configuration {0} already in protection group {1}.
226	E_ODG_ODG_CONFIG_NOT_FOUND_IN_PG	Oracle Data Guard configuration {0} is not found in protection group {1}.
231	E_ODG_UNABLE_TO_NOTIFY_STATUS_CHANGE	Unabled to send change notification for data replication status.
234	E_ODG_SAME_PROPERTY_VALUE	Property value already set. No modification is needed.
235	E_ODG_UNEXPECTED_EXCEPTION	Unexpected exception.

5.3.3. Error recovery

When a protection group is in a configuration error state, after the error is fixed at the Oracle end, the protection group may be revalidated to clear the configuration error.

5.4. ODG control script (odg_control)

The data replication layer is controlled by the Odyssey framework through the data replication mbean. ODG data replication mbean operates on the data replication layer by invoking the /opt/SUNWscgrepodg/lib/odg_control script, which in turn calls function in the /opt/SUNWscgrepodg/lib/odg_control_lib shell script. (This makes disaster recovery operations more modular.) Odg_control and other scripts are contained in the SUNWscgrepodg package.

The odg_control scripts returns 0 if the operation is completed successfully. Otherwise

an error code is returned and the corresponding error message is printed. The error code and message is described in the table in section “Error conditions“ above.

The major difference between ODG and, say, AVS replication is that device group are added to an AVS replicated PG, while with ODG PGs they are ODG configurations. In the *Helen* release, neither the CLI nor the GUI have been modified to use more generic descriptions, e.g. replication object. This will need to be changed in the *Kirke* release.

The following sub-sections list the main functions in odg_control, the two key routines also document where they are in turn called from.

5.4.1. Function get_nodelist_for_rg

This function will be needed because individual RAC server proxy resource groups may not necessarily span the same set of nodes. This may even be true when these RGs are in the same PG. Therefore, it will be necessary to retrieve the node list for each RG so that subsequent calls to dgmgrl, etc will not target a node without Oracle binaries installed. The returned node list will have the requested separator, e.g. ' ' or ','

5.4.2. Function apply_odg_configuration

This will check that the parameters for the ODG configuration are valid, i.e. match the configuration held by ODG itself (using output from dgmgrl as a check) and then applies or re-applies the configuration. This includes creating the replication RG and resource as well as the Oracle RAC proxy server shadow RG and resource. The latter can then be added to the PG if it hasn't been already.

This function will be called from the applyOdgConfiguration method in the ODG Java code which will in turn be called from createDeviceGroup, modifyDeviceGroup and checkAndApplyConfiguration.

5.4.3. Function remove_odg_configuration

This function will remove the ODG configuration. This will include removing the RAC proxy server shadow RG and resource as well as the replication resource corresponding to this ODG configuration. Only if there are no other resources in the replication RG itself will it be removed as well.

This function will be called from the removeOdgConfiguration method in the ODG Java code which will in turn be called from createDeviceGroup, deleteDeviceGroup and

checkAndApplyConfiguration.

5.4.4. Function start_or_stop_replication

This function will start or stop the ODG replication. Before it does that, it will check to see if the ODG configuration matches the state expected by Odyssey. If it doesn't, or if it can't get a 'SUCCESS' response to “show configuration” from dgmgrl, it will return an error.

5.4.5. Function check_application_rgs

This function will check that the application RG list supplied does not contain any RGs with a RAC server proxy resource in, i.e. an RG managing a 'real' Oracle RAC server. Controlling this would be undesirable from an ODG replication perspective and if these were being replicated in a different manner, e.g. TrueCopy, they should not be present in this group anyway.

5.4.6. Function check_switchover

This function will check that the ODG configuration is in a suitable state to execute a switchover. The requirements are that a 'SUCCESS' response is retuened from “show configuration” when run from dgmgrl, that the primary database (site) is the expected one and that the remote database (secondary) is enabled.

5.4.7. Function check_takeover

This function will check that what is left of the ODG configuration is in a suitable state to execute a takeover (or failover in Oracle parlance). The target secondary database must be enabled.

5.4.8. Function switchover

This function will switch the primary site to the current secondary site. It will execute a “switchover to <remote_db_name>” through dgmgrl and check that it receives “*succeeded*new*<remote_db_name>” pattern as the last line of the output. Various normal Oracle errors that are returned during this process will be ignored, e.g. ORA-01109 (database not open).

5.4.9. Function takeover

This function will again check that the target secondary hasn't been disabled in the interim period since the check_takeover has been run. If the role of the supposed secondary already turns out to be primary, then the function will return success immediately. If there is work to be done, then the command “"failover to <local_db_name> immediate" will be run via dgmgrl from the secondary (as it cannot be run from the primary in any circumstance). From this point on, the function will return success regardless of whether the command succeeds or not. Any failure will be picked up by subsequent probe cycles and will have to be rectified by the DBA.

5.5. Status of ODG data replication mbean

The status of the mbean reflects the aggregated status of all SUNW.gds replication resources in the replication resource group.

The following table illustrates the mapping from the state/status of each SUNW.gds replication resource to the mbean status. An 'X' represent any possible state for the resource and simply demonstrates that the most restrictive state governs the overall state of the mbean.

Unknown	***Faulted***	***Degraded***	***Online***	***Mbean status***
True	X	X	X	UNKNOWN
False	True	X	X	FAULTED
False	False	True	X	DEGRADED
False	False	False	True	ONLINE

6. Risks/Issues

The major issue for this project is the need to have the Oracle DBA password to perform an ODG switch-over or fail-over. The method developed to handle passwords minimises exposure of a password in clear text. It is only ever present in the input files used to drive the dgmgrl commands. These input files are deleted after use and furthermore, they are only readable by the Oracle user (and root).

7. Release Information

7.1. Packaging

Two new package will be created for ODG: SUNWscgrepodg and SUNWscgrepodgu. SUNWscgrepodg will contain the shell scripts, while SUNWscgrepodgu will contain the class files and associated property files.

With the *Kirke* release there should be cosmetic changes made in the Odyssey framework packages to allow the CLI and BUI (web interface) to reflect a more generic approach to object replication than 'device group' when constructing protection groups. The ODG module will not necessitate such changes, the module being capable of working using the device-group constructs, but will benefit from their implementation through more intuitive interfaces.

7.2. Installation

The ODG packages will be installed as a part of the Odyssey product.

7.3. Administration

The Odyssey framework manages the SUNW.gds replication and shadow Oracle RAC proxy server resource via the RGM framework.

7.4. Upgrade

The SUNW.gds resource type follows the same upgrade procedures for existing Solaris Cluster resource types.

8. Acknowledgements

The format and content of this document draws heavily on the AVS and AVS Mbean design documents [6] [7]. I am grateful to the authors for being able to re-use so much of their content and layout.

9. References

1. Odyssey Architecture Document: http://hadoc.france/cgi-bin/getdoc?ref=TS-TR-03-29
2. FBC 2003/1303 HA Agent Support for Oracle Data Guard Standby Instances: http://galileo.sfbay.sun.com/SC/dev-process/features/2003/1303/
3. FBC 2005/1413 Oracle RAC Manageability on Sun Cluster: http://galileo.sfbay.sun.com/SC/dev-process/features/2005/1413/
4. Data Guard Broker documentation http://download-uk.oracle.com/docs/cd/B19306_01/server.102/b14230/toc.htm
5. Oracle Metalink http://metalink.oracle.com
6. Design Document: GeoCtlAVS Resource Type for Control of Data Replication with Remote Mirror in Odyssey GeoCluster http://hadoc.france/cgi-bin/getdoc/design.pdf?ref=TS-TR-03-106.1.3
7. Data Replication Mbean Design Documenet - AVS section http://hadoc.france/cgi-bin/getdoc/?ref=TS-TR-04-14.0.2
8. FBC 2005/1495 Log Viewer http://galileo.sfbay.sun.com/SC/dev-process/features/2005/1495/
9. FBC 2004/1405 Clusterized zones http://galileo.sfbay.sun.com/SC/dev-process/features/2004/1405/

10. Appendix 1 – unsupported options

This section contains descriptions of Oracle configuration options: HA-Oracle and HA-Oracle agent running within an HA Solaris 10 container, that will not be supported by the initial release of the product because they would be unsupported by Oracle or have other technical issues.

10.1. Integration with Solaris Cluster HA Oracle agent

Configurations using a standard Solaris Cluster HA-Oracle cannot use dgmgrl because it is unsupported by Oracle. (see Oracle Metalink document 413696.1). The main technical issue is that the change of host name breaks their configuration file. While not insurmountable, lack of support from Oracle is the main obstacle.

If it were possible, it would be achieved through the existing Solaris Cluster HA-Oracle agent (SUNW.oracle resource type), which was modified to support the concept of a Data Guard standby instance. See FBC1303 [2] for the technical details of the changes. In this model, the database would be started under the control of the Solaris Cluster HA Oracle agent, and Odyssey would manipulate the Solaris Cluster HA-Oracle Dataguard_role resource property of the database being controlled. On the primary site, it would be changed to 'standby', while on the secondary site it would be changed to 'primary'. During the change over though, both sites would have the property set to 'in_transition' to avoid any failure detection kicking in.

10.2. HA-Oracle in an HA Solaris 10 container

This configuration, where Oracle has been made HA by virtue of being in an HA container and having a probe running to restart it on failure, has a mixture of benefits and drawbacks for customers wishing to use it:

The benefits are that the Oracle installation is greatly simplified, the container making it look like a standard single system installation. This avoids having to create symbolic links for init<SID>.ora files, or manage multiple copies of listener.ora and tnsnames.ora, etc. Similarly, no tweaks are required to make the standard, standalone Oracle Enterprise Manager configuration work. The biggest benefit, from this project's perspective, is that the Data Guard broker (dgmgrl) continues to work after a fail-over because the hostname, as seen by Oracle, does not change. The system can also be managed through Oracle Enterprise Manager Grid Control too.

The drawback of this configuration is that failures of an Oracle database cannot currently (until RFE 6443496 has been implemented and putback) force a 'give-over' of the HA-zone to another Solaris node. This is because, from the RGM's point of view, the fail-over zone is not present on the alternative node, at the point of failure, and therefore isn't a viable candidate for Oracle to fail-over to. Consequently, Oracle is constrained only to restart in the same zone and only an entire zone failure would cause the zone to be migrated. A further consideration is that a zone fail-over will add it's outage time to that of Oracle compared with a non fail-over (static) zone configuration, where the zone is already booted.

The key resources and resource groups that comprise one end of such a Geographic Edition cluster configuration are shown in Illustration 3 below. As will be seen in the subsequent alternative set-ups, the driving force is the Oracle server resource. Unless this resource is up and running, no replication can take place. Here we ignore the possibility that the system administrator has taken the resource group offline and started Oracle outside Solaris Cluster control.

To fulfil Odyssey's protection group architecture, i.e. Have a resource group that can be moved into the online or unmanaged state, to reflect the notion of primary and secondary sites, a shadow fail-over Oracle resource group is created. This resource group must have the Auto_restart_on_new_cluster property set to false. There is no real requirement for this resource group to be online on the same node as the real Oracle RG so no affinities are set.

Within that resource group, is a shadow Oracle resource that can modify the properties of the real (i.e. The resource of type SUNW.oracle_server) Oracle resource when it is moved between the online and unmanaged states.

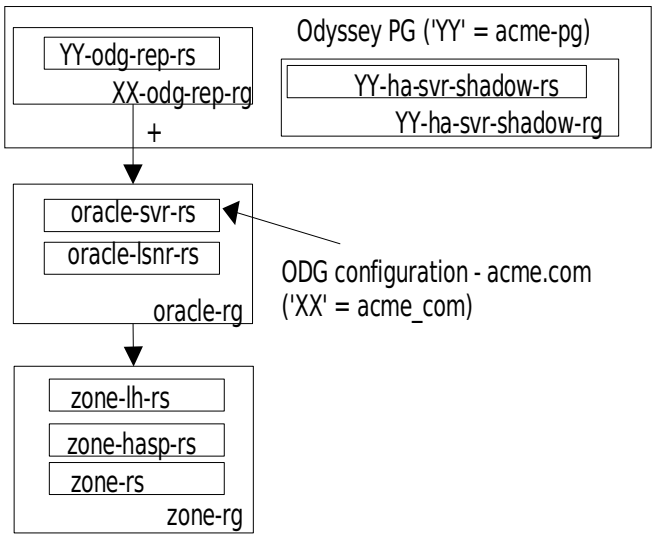

Illustration 3: Potential Oracle in an HA Solaris container configurations with Data Guard

The replication will be monitored via an ODG replication resource located in the ODG replication RG.

Note: Oracle have stated that they may choose not to support this configuration. Consequently, this option will not be supported in the initial release.

10.3. Standard HA Oracle

This is typically the most common Oracle configuration but correct installation is more complex than one performed on a non cluster node. The level of complexity depends on whether a global or fail-over file system is used and whether the Oracle home is centralized or on local storage. Furthermore, extra work is needed to make the Oracle Enterprise Manager work in this environment. However, the biggest draw-back with this configuration is that it does not work with Data Guard broker (see Oracle Metalink [5] document number 413696.1) and requires on-going configuration and management through SQL*Plus. It may be possible to overcome the hostname change that underlies this problem by using zone nodes which use the same name, but again there is the remaining issue of lack of Oracle support for the configuration.

The two major plus points for this approach is that it has a faster fail-over time compared with the Oracle in a zone architecture and can also force a 'give-over' if it has failed repeatedly to start on a particular node.

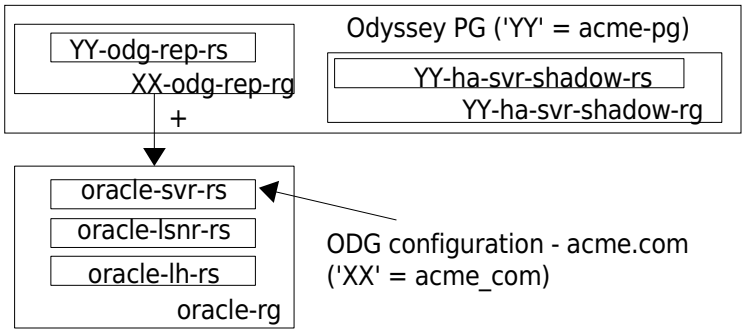

Illustration 4: Potential HA Oracle configuration with Data Guard

The key resources and resource groups that comprise one end of such a Geographic Edition cluster configuration are shown in Illustration 4 above. Here the configuration of the resource groups and resource is almost identical to that used for HA Oracle in an HA Solaris 10 container. However, because the dgmgrl interface is not supported by Oracle for this configuration it **will not be supported** by the initial release of the module. This should not be a major issue since it is expected that the majority of large enterprise users who want to use this type of protection will be using Oracle RAC anyway.

880

```
ALTER SYSTEM SET log_archive_dest_2='' SCOPE=BOTH;

ALTER SYSTEM SET log_archive_dest_state_2='ENABLE' SCOPE=BOTH;

ALTER DATABASE   MOUNT

ALTER DATABASE OPEN

ALTER SYSTEM SET
log_archive_dest_2='location="dgsby_bar"','valid_for=(STANDBY_LOGFILE,STANDBY
_ROLE)' SCOPE=BOTH SID='bar';

ALTER SYSTEM SET log_archive_dest_state_2='ENABLE' SCOPE=BOTH SID='bar';

ALTER SYSTEM SET standby_archive_dest='dgsby_bar' SCOPE=BOTH SID='bar';

ALTER SYSTEM SET log_archive_trace=0 SCOPE=BOTH SID='bar';

ALTER SYSTEM SET log_archive_format='%t_%s_%r.dbf' SCOPE=SPFILE SID='bar';

ALTER SYSTEM SET standby_file_management='MANUAL' SCOPE=BOTH SID='*';

ALTER SYSTEM SET archive_lag_target=0 SCOPE=BOTH SID='*';

ALTER SYSTEM SET log_archive_max_processes=2 SCOPE=BOTH SID='*';

ALTER SYSTEM SET log_archive_min_succeed_dest=1 SCOPE=BOTH SID='*';

ALTER SYSTEM SET
fal_server='(DESCRIPTION=(ADDRESS_LIST=(ADDRESS=(PROTOCOL=TCP)(HOST=oracle-
lh)(PORT=1521)))(CONNECT_DATA=(SERVICE_NAME=barb_XPT)(SERVER=dedicated)))'
SCOPE=BOTH;

ALTER SYSTEM SET
fal_client='(DESCRIPTION=(ADDRESS_LIST=(ADDRESS=(PROTOCOL=TCP)(HOST=oracle-
dg-
zone)(PORT=1521)))(CONNECT_DATA=(SERVICE_NAME=bar_XPT)(INSTANCE_NAME=bar)(SER
VER=dedicated)))' SCOPE=BOTH;

ALTER DATABASE RECOVER MANAGED STANDBY DATABASE  THROUGH ALL SWITCHOVER
DISCONNECT  USING CURRENT LOGFILE
```

Text 2: Example of the Oracle initialization parameters manipulated by dgmgrl

For HA ODG configurations, the equivalent SQL*Plus commands to replicate dgmgrl control are show in Text 2: Example of the Oracle initialization parameters manipulated by dgmgrl above. Attempting to replicate this sequence of commands together with the many configuration options would add significant complexity to the project. Consequently, no HA-ODG support will be available in the initial release.

11. Appendix 2 – sample Oracle networking configuration files

This appendix provides some sample init.ora, listener.ora and tnsnames.ora files for use with an HA Solaris 10 container and HA-Oracle configured with Data Guard. They could also be used for HA-Oracle without the HA Solaris 10 container, bearing in mind that dgmgrl does not work for this configuration.

In the configuration ORACLE_HOME is set to /oracle/oracle/product/10.2.0/db_1 and ORACLE_SID is either hazprd (production) or hazsby (standby). All the data is stored under /oradata, mainly in directories /oradata/arch, /oradata/flash_recovery_area and /oradata/<sid>.

- inithazprd.ora

```
hazprd.__db_cache_size=159383552
hazprd.__java_pool_size=4194304
hazprd.__large_pool_size=4194304
hazprd.__shared_pool_size=117440512
hazprd.__streams_pool_size=0
*.archive_lag_target=0
*.audit_file_dest='/oracle/oracle/product/10.2.0/db_1/admin/hazprd/adump'
*.background_dump_dest='/oracle/oracle/product/10.2.0/db_1/admin/hazprd/bdump'
*.compatible='10.2.0.3.0'
*.control_files='/oradata/hazprd/control01.ctl','/oradata/hazprd/control02.ctl','/oradata/hazprd/contr
ol03.ctl'
*.core_dump_dest='/oracle/oracle/product/10.2.0/db_1/admin/hazprd/cdump'
*.db_block_size=8192
*.db_file_multiblock_read_count=16
*.db_file_name_convert='/oradata/hazsby/','/oradata/hazprd'
*.db_name='hazprd'
*.db_recovery_file_dest='/oradata/flash_recovery_area'
*.db_recovery_file_dest_size=2147483648
*.db_unique_name='hazprd'
*.dg_broker_start=TRUE
*.dispatchers='(PROTOCOL=TCP) (SERVICE=hazprdXDB)'
*.fal_client='(DESCRIPTION=(ADDRESS_LIST=(ADDRESS=(PROTOCOL=tcp)(HOST=ora-prim-
zone)(PORT=1521)))(CONNECT_DATA=(SERVICE_NAME=hazprd_XPT)(INSTANCE_NAME=hazprd)(SERVER=dedicated)))'
*.fal_server='(DESCRIPTION=(ADDRESS_LIST=(ADDRESS=(PROTOCOL=tcp)(HOST=ora-stby-
zone)(PORT=1521)))(CONNECT_DATA=(SERVICE_NAME=hazsby_XPT)(SERVER=dedicated)))'
*.job_queue_processes=10
*.local_listener=''
*.log_archive_config='dg_config=(hazprd','hazsby)'
*.log_archive_dest_1='location="/oradata/arch" valid_for=(ONLINE_LOGFILE,ALL_ROLES)'
hazprd.log_archive_dest_1='location="/oradata/arch"','valid_for=(ONLINE_LOGFILE,ALL_ROLES)'
*.log_archive_dest_2='service="(DESCRIPTION=(ADDRESS_LIST=(ADDRESS=(PROTOCOL=tcp)(HOST=ora-stby-
zone)(PORT=1521)))(CONNECT_DATA=(SERVICE_NAME=hazsby_XPT)(INSTANCE_NAME=hazsby)(SERVER=dedicated)))"',
'   LGWR SYNC AFFIRM delay=0 OPTIONAL max_failure=0 max_connections=1   reopen=300
db_unique_name="hazsby" register net_timeout=180  valid_for=(online_logfile,primary_role)'
hazprd.log_archive_dest_state_1='ENABLE'
*.log_archive_dest_state_2='ENABLE'
*.log_archive_format='%t_%s_%r.dbf'
hazprd.log_archive_format='%t_%s_%r.dbf'
*.log_archive_max_processes=2
*.log_archive_min_succeed_dest=1
hazprd.log_archive_trace=0
```

```
*.open_cursors=300
*.pga_aggregate_target=96468992
*.processes=150
*.remote_login_passwordfile='EXCLUSIVE'
*.sga_target=290455552
*.standby_archive_dest='/oradata/arch'
hazprd.standby_archive_dest=''
*.standby_file_management='MANUAL'
*.undo_management='AUTO'
*.undo_tablespace='UNDOTBS1'
*.user_dump_dest='/oracle/oracle/product/10.2.0/db_1/admin/hazprd/udump'
```

- inithazsby.ora

```
hazsby.__db_cache_size=130023424
hazsby.__java_pool_size=4194304
hazsby.__large_pool_size=4194304
hazsby.__shared_pool_size=146800640
hazsby.__streams_pool_size=0
*.archive_lag_target=0
*.audit_file_dest='/oracle/oracle/product/10.2.0/db_1/admin/hazsby/adump'
*.background_dump_dest='/oracle/oracle/product/10.2.0/db_1/admin/hazsby/bdump'
*.compatible='10.2.0.3.0'
*.control_files='/oradata/hazsby/control01.ctl','/oradata/hazsby/control02.ctl','/oradata/hazsby/contr
ol03.ctl'#Restore Controlfile
*.core_dump_dest='/oracle/oracle/product/10.2.0/db_1/admin/hazsby/cdump'
*.db_block_size=8192
*.db_domain='sfbay'
*.db_file_multiblock_read_count=16
*.db_file_name_convert='/oradata/hazprd/','/oradata/hazsby/'
*.db_name='hazprd'
*.db_recovery_file_dest='/oradata/flash_recovery_area'
*.db_recovery_file_dest_size=2147483648
*.db_unique_name='hazsby'
*.dg_broker_start=TRUE
*.dispatchers='(PROTOCOL=TCP) (SERVICE=hazsbyXDB)'
*.fal_client='(DESCRIPTION=(ADDRESS_LIST=(ADDRESS=(PROTOCOL=tcp)(HOST=ora-stby-
zone)(PORT=1521)))(CONNECT_DATA=(SERVICE_NAME=hazsby_XPT)(INSTANCE_NAME=hazsby)(SERVER=dedicated)))'
*.fal_server='(DESCRIPTION=(ADDRESS_LIST=(ADDRESS=(PROTOCOL=tcp)(HOST=ora-prim-
zone)(PORT=1521)))(CONNECT_DATA=(SERVICE_NAME=hazprd_XPT)(SERVER=dedicated)))'
*.job_queue_processes=10
*.log_archive_config='dg_config=(hazprd,hazsby)'
*.log_archive_dest_1='location="/oradata/arch"'
hazsby.log_archive_dest_1='location="/oradata/arch"','valid_for=(ALL_LOGFILES,ALL_ROLES)'
*.log_archive_dest_2='service=hazsby valid_for=(ONLINE_LOGFILE,PRIMARY_ROLE) db_unique_name=hazsby'
*.log_archive_dest_3=''
hazsby.log_archive_dest_state_1='ENABLE'
*.log_archive_dest_state_3='ENABLE'
*.log_archive_format='%t_%s_%r.dbf'
hazsby.log_archive_format='%t_%s_%r.dbf'
*.log_archive_max_processes=2
*.log_archive_min_succeed_dest=1
hazsby.log_archive_trace=0
*.open_cursors=300
*.pga_aggregate_target=96468992
*.processes=150
*.remote_login_passwordfile='EXCLUSIVE'
*.sga_target=290455552
*.standby_archive_dest='/oradata/arch'
hazsby.standby_archive_dest='/oradata/arch'
*.standby_file_management='MANUAL'
*.undo_management='AUTO'
*.undo_tablespace='UNDOTBS1'
*.user_dump_dest='/oracle/oracle/product/10.2.0/db_1/admin/hazsby/udump'
```

Sample listener.ora files

Key entries to making the configuration work are those where GLOBAL_DBNAME is set to <sid>_DGMGRL.sfbay. These are need to allow dgmgrl to start and stop the instances. Note, it the listener is on a non-standard port, i.e. Not 1521, entries for local_listener are needed in the init<sid>.ora files.

- hazprd listener.ora

```
SID_LIST_LISTENER =
  (SID_LIST =
    (SID_DESC =
      (SID_NAME = hazprd)
      (ORACLE_HOME = /oracle/oracle/product/10.2.0/db_1)
    )
    (SID_DESC =
      (SID_NAME = hazprd)
      (GLOBAL_DBNAME = hazprd_DGMGRL)
      (ORACLE_HOME = /oracle/oracle/product/10.2.0/db_1)
    )
    (SID_DESC =
      (SID_NAME = PLSExtProc)
      (ORACLE_HOME = /oracle/oracle/product/10.2.0/db_1)
      (PROGRAM = extproc)
    )
  )

LISTENER =
  (DESCRIPTION_LIST =
    (DESCRIPTION =
      (ADDRESS = (PROTOCOL = TCP)(HOST = ora-prim-zone)(PORT = 1521))
      (ADDRESS = (PROTOCOL = IPC)(KEY = EXTPROC0))
    )
  )
```

- hazsby listener.ora

```
SID_LIST_LISTENER =
  (SID_LIST =
    (SID_DESC =
      (SID_NAME = hazsby)
      (ORACLE_HOME = /oracle/oracle/product/10.2.0/db_1)
    )
    (SID_DESC =
      (SID_NAME = hazsby)
      (GLOBAL_DBNAME = hazsby_DGMGRL)
      (ORACLE_HOME = /oracle/oracle/product/10.2.0/db_1)
    )
    (SID_DESC =
      (SID_NAME = PLSExtProc)
      (ORACLE_HOME = /oracle/oracle/product/10.2.0/db_1)
      (PROGRAM = extproc)
    )
  )

LISTENER =
  (DESCRIPTION_LIST =
    (DESCRIPTION =
      (ADDRESS = (PROTOCOL = TCP)(HOST = ora-stby-zone)(PORT = 1521))
```

```
      (ADDRESS = (PROTOCOL = IPC)(KEY = EXTPROC0))
    )
  )
```

- tnsnames.ora

```
HAZPRD_DGMGRL =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = ora-prim-zone)(PORT = 1521))
    )
    (CONNECT_DATA =
      (SERVICE_NAME = hazprd)
    )
  )

HAZPRD =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = ora-prim-zone)(PORT = 1521))
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = hazprd)
    )
  )

HAZSBY =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = ora-stby-zone)(PORT = 1521))
    )
    (CONNECT_DATA =
      (SERVICE_NAME = hazsby)
    )
  )

HAZSBY_DGMGRL =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = ora-stby-zone)(PORT = 1521))
    )
    (CONNECT_DATA =
      (SERVICE_NAME = hazsby)
    )
  )

EXTPROC_CONNECTION_DATA =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = IPC)(KEY = EXTPROC0))
    )
    (CONNECT_DATA =
      (SID = PLSExtProc)
      (PRESENTATION = RO)
    )
  )
```

12. Appendix 3 – dgmgrl equivalent sqlplus commands

This section describes some of the sqlplus command sequences that equate to the actions performed by dgmgrl.

12.1. SQL*Plus switch-over command sequence

The switch-over method using sqlplus is considerably more complex than the simple 'switchover to <SID>' used by dgmgrl. The commands used by dgmgrl are logged to the alert_<sid>.log file and are shown below for a specific configuration.

On the existing primary the sequence is:

```
ALTER DATABASE RECOVER MANAGED STANDBY DATABASE CANCEL
ALTER DATABASE COMMIT TO SWITCHOVER TO PRIMARY WAIT WITH SESSION SHUTDOWN
shutdown
startup
ALTER SYSTEM SET standby_archive_dest='' SCOPE=BOTH SID='hazsby';
ALTER SYSTEM SET log_archive_dest_3='' SCOPE=BOTH SID='hazsby';
ALTER SYSTEM SET
log_archive_dest_2='service="(DESCRIPTION=(ADDRESS_LIST=(ADDRESS=(PROTOCOL=tcp)(HOST=ora-prim-
zone)(PORT=1521)))(CONNECT_DATA=(SERVICE_NAME=hazprd_XPT)(INSTANCE_NAME=hazprd)(SERVER=dedicated)))"',
'   LGWR SYNC AFFIRM delay=0 OPTIONAL max_failure=0 max_connections=1   reopen=300
db_unique_name="hazprd" register net_timeout=180  valid_for=(online_logfile,primary_role)' SCOPE=BOTH;
ALTER SYSTEM SET log_archive_dest_state_2='ENABLE' SCOPE=BOTH;
ALTER DATABASE OPEN
ALTER SYSTEM SET log_archive_trace=0 SCOPE=BOTH SID='hazsby';
ALTER SYSTEM SET log_archive_format='%t_%s_%r.dbf' SCOPE=SPFILE SID='hazsby';
ALTER SYSTEM SET standby_archive_dest='' SCOPE=BOTH SID='hazsby';
ALTER SYSTEM SET standby_file_management='MANUAL' SCOPE=BOTH SID='*';
ALTER SYSTEM SET archive_lag_target=0 SCOPE=BOTH SID='*';
ALTER SYSTEM SET log_archive_max_processes=2 SCOPE=BOTH SID='*';
ALTER SYSTEM SET log_archive_min_succeed_dest=1 SCOPE=BOTH SID='*';
ALTER SYSTEM SET db_file_name_convert='/oradata/hazprd/','/oradata/hazsby/' SCOPE=SPFILE;
ALTER SYSTEM SET
log_archive_dest_2='service="(DESCRIPTION=(ADDRESS_LIST=(ADDRESS=(PROTOCOL=tcp)(HOST=ora-prim-
zone)(PORT=1521)))(CONNECT_DATA=(SERVICE_NAME=hazprd_XPT)(INSTANCE_NAME=hazprd)(SERVER=dedicated)))"',
'   LGWR SYNC AFFIRM delay=0 OPTIONAL max_failure=0 max_connections=1   reopen=300
db_unique_name="hazprd" register net_timeout=180  valid_for=(online_logfile,primary_role)' SCOPE=BOTH;
ALTER SYSTEM SET log_archive_dest_state_2='ENABLE' SCOPE=BOTH;
```

While on the standby they are:

```
ALTER DATABASE COMMIT TO SWITCHOVER TO PHYSICAL STANDBY WITH SESSION SHUTDOWN
ALTER SYSTEM SET log_archive_dest_2='' SCOPE=BOTH;
ALTER SYSTEM SET log_archive_dest_state_2='ENABLE' SCOPE=BOTH;
shutdown
(alter database CLOSE NORMAL
alter database DISMOUNT)
startup
ALTER SYSTEM SET log_archive_dest_1='location="/oradata/arch"','valid_for=(ALL_LOGFILES,ALL_ROLES)'
SCOPE=BOTH SID='hazprd';
ALTER SYSTEM SET log_archive_dest_state_1='ENABLE' SCOPE=BOTH SID='hazprd';
ALTER SYSTEM SET standby_archive_dest='/oradata/arch' SCOPE=BOTH SID='hazprd';
ALTER SYSTEM SET log_archive_trace=0 SCOPE=BOTH SID='hazprd';
ALTER SYSTEM SET log_archive_format='%t_%s_%r.dbf' SCOPE=SPFILE SID='hazprd';
ALTER SYSTEM SET standby_file_management='MANUAL' SCOPE=BOTH SID='*';
```

```
ALTER SYSTEM SET archive_lag_target=0 SCOPE=BOTH SID='*';
ALTER SYSTEM SET log_archive_max_processes=2 SCOPE=BOTH SID='*';
ALTER SYSTEM SET log_archive_min_succeed_dest=1 SCOPE=BOTH SID='*';
ALTER SYSTEM SET db_file_name_convert='/oradata/hazsby/','/oradata/hazprd' SCOPE=SPFILE;
ALTER SYSTEM SET fal_server='(DESCRIPTION=(ADDRESS_LIST=(ADDRESS=(PROTOCOL=tcp)(HOST=ora-stby-
zone)(PORT=1521)))(CONNECT_DATA=(SERVICE_NAME=hazsby_XPT)(SERVER=dedicated)))' SCOPE=BOTH;
ALTER SYSTEM SET fal_client='(DESCRIPTION=(ADDRESS_LIST=(ADDRESS=(PROTOCOL=tcp)(HOST=ora-prim-
zone)(PORT=1521)))(CONNECT_DATA=(SERVICE_NAME=hazprd_XPT)(INSTANCE_NAME=hazprd)(SERVER=dedicated)))'
SCOPE=BOTH;
ALTER DATABASE RECOVER MANAGED STANDBY DATABASE  THROUGH ALL SWITCHOVER DISCONNECT  USING CURRENT
LOGFILE
```

The critical SQL commands statements centre on the cancelling of the existing recovery, the transition to the new primary via the manipulation of the fal_client and fal_server properties and the restart of the databases. The exact commands also depend on whether the standby database is a logical or physical standby.