

ORACLE®

Zero Downtime: Hiding Planned Maintenance and Unplanned Outages from Applications

Carol Colrain
Consulting Member of Technical Staff,
Technical Lead for Client-Failover, RAC Development

ORACLE

ORACLE
DATABASE

Safe Harbor Statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

Program Agenda

- 1 Problems to Solve
- 2 Fast Application Notification
- 3 Continuous Connections
- 4 Hiding Planned Maintenance
- 5 Hiding Unplanned Outages
- 6 Success Stories

1 ➤ What problems confront applications at database outages?



In-Flight Work

Pre-12c Situation

Database outages cause in-flight work to be lost, leaving users and applications in-doubt

- Restart applications and mid-tiers
- User frustration
- Cancelled work
- Duplicate submissions
- Errors even when planned
- Developer pains

Sorry. Internal Server Error - 500 Error
We are currently experiencing an issue with our servers on coolcar.com. Please come back later.

6. Estimated Trip Cost

| | |
|---------------------------------|----------------|
| Flight Total | 1,536.69 AUD |
| San Francisco, CA - Hotel Total | 1,800.00 USD ‡ |
| | 1,950.65 AUD |

Trip Total: 3,487.35 AUD ‡
3,218.00 USD

‡ Please note that this total is based on available information. The estimated cost may not include taxes and fees.

- Remember to obtain an original invoice for all your expenses where required under the Global Travel Policy. The invoice should always include the name and address of your Oracle company. Failure to obtain a proper invoice may increase Oracle's costs by up to 25%.

Please Note:
If you do not receive a confirmation after clicking Purchase Trip (EX. receive a **blank error message** or an error message that states **"You already have an active session in the online booking site"**), please call CWT before clicking purchase trip again.

[Purchase Trip](#) [Start Over](#)

How do we reach all applications?

- Move work to different instance/database with no errors reported to applications at planned maintenance
- Hide unplanned database outages from the applications
- Take adoption out of the developers hands to configuration/operation only
- Work with current drivers and older database, whenever possible


2 ➤ Outage Detection

The dead thing cannot tell you that it's dead



Applications Waste Time

- Hanging on TCP/IP timeouts
- Connecting when services are down
- Not connecting when services resume
- Receiving errors during planned maintenance
- Processing partial results when server is down
- Attempting work at slow, hung, or dead nodes



**Performance
issues not
reported in your
favorite tools.**

Fast Application Notification

Proven since 10g



- **Down** – received in low ms to invoke failover
- **Planned Down** – drains sessions for planned maintenance with no user interruption whatsoever
- **Up** – Re-allocates sessions when services resume
- **Load %** - Advice to balance sessions for RAC locally and GDS globally
- **Affinity** - Advice when to keep conversation locality

12c: Auto-Configuration
+ Global Data Services

12c FAN: Standardized, Auto-Configured

| Client | 10g | 11g | 12c |
|----------------------------------|-----|-----|------------|
| JDBC Implicit Connection Cache | ONS | ONS | desupport |
| JDBC Universal Connection Pool | | ONS | ONS |
| OCI/OCCL driver | AQ | AQ | ONS |
| ODP.NET Unmanaged Provider (OCI) | AQ | AQ | ONS |
| ODP.NET Managed Provider (C#) | | ONS | ONS |
| OCI Session Pool | AQ | AQ | ONS |
| WebLogic Active GridLink | | ONS | ONS |
| Tuxedo | | ONS | ONS |
| Listener | ONS | ONS | ONS |

12c JDBC FAN Auto-Configures

- **12c JDBC clients and 12c Oracle database**
 - Check ons.jar is included in the class path
 - To enable FAN set the pool property
 - **fastConnectionFailoverEnabled=true**
- **Before 12c - JDBC clients or database**
 - also set the pool property for remote ons
 - **ONSConfiguration=nodes=mysun05:6200,mysun06:6200, mysun07:6200,mysun08:6200**

12c OCI FAN Auto-Configures

- **12c OCI clients and 12c Oracle database**

Use srvctl to configure the service for AQ HA Notification:

```
srvctl modify service -db EM -service GOLD -notification TRUE
```

For the client, enable in oraaccess.xml

```
<oraaccess>  
  <default_parameters>  
    <events>true</events>  
  </default_parameters>  
</oraaccess>
```

- **Before 12c OCI clients or database**

- Enable OCI_EVENTS at environment creation OCIEnvCreate(..)
- Link the app with the client thread o/s library.

12c ODP.Net FAN Auto-Configures

- **12c ODP.Net clients and 12c Oracle database**

Use srvctl to configure the service for AQ HA Notification:

```
srvctl modify service -db EM -service GOLD -notification TRUE
```

To enable FAN, in the connection string -

- "user id=oracle; password=oracle; data source=HA; **pooling=true; HA events=true;**"

To enable Runtime Load Balancing, also in the connection string -

- "user id=oracle; password=oracle; data source=HA; **pooling=true; HA events=true; load balancing=true;**"

FAN with other Java Application Servers

Use UCP – a simple DataSource replacement

General Properties

Additional Properties

Scope
cells:expe-was:nodes:ee001a:servers:ST6AppServerEE001A

Name
Oracle JDBC Driver UCP ST6_QC02P01

Description
Oracle JDBC Driver UCP ST6_QC02P01

Class path
\${WAS_INSTALL_ROOT}/jdbc/ojdbc7.jar
\${WAS_INSTALL_ROOT}/jdbc/ucp.jar
\${WAS_INSTALL_ROOT}/jdbc/ons.jar

Native library path

Isolate this resource provider

Implementation class name
oracle.ucp.jdbc.PoolDataSourceImpl

Apply OK Reset Cancel

IBM WebSphere
Apache Tomcat

Class path to be set for UCP JDBC Provider
\${WAS_INSTALL_ROOT}/jdbc/ojdbc7.jar
\${WAS_INSTALL_ROOT}/jdbc/ucp.jar
\${WAS_INSTALL_ROOT}/jdbc/ons.jar

Pool Data Source

Monitor FAN

- Create a FAN callout in `..$GRID_HOME/racg/userco`
- Download ONS subscriber (ONCCTL) from OTN RAC page

```
oncctl
```

```
..
```

```
VERSION=1.0 event_type=SERVICEMEMBER service=orcl_swing_pdb2 instance=orcl1 database=orcl  
db_domain= host=sun01 status=down reason=USER timestamp=2014-07-30 12:02:51 timezone=-07:00
```

```
VERSION=1.0 event_type=SERVICEMEMBER service=orcl_swing_pdb10 instance=orcl1 database=orcl  
db_domain= host=sun01 status=down reason=USER timestamp=2014-07-30 12:02:52 timezone=-07:00
```

```
VERSION=1.0 event_type=SERVICE service=orcl_swing_pdb10 database=orcl db_domain= host=sun01  
status=down reason=USER
```


Lessons Learned - FAN/FCF

- Always return connections to the pool for stable connection usage, runtime balancing, and planned draining
- Use current client (12102) against current or earlier RDBMS
- Use `LOAD_BALANCE=off` (default) across DG sites for FAN/ONS sets
- For Java:
 - Do not mix up jar versions : use UCP, ONS, and ojdb7 12102 jars
- For OCI and ODP.Net Unmanaged Provider
 - When using databases before 12c, FAN is posted over AQ

Continuous Connections

Applications should see no errors while services relocate.



Connections Appear Continuous

while a service is temporarily unavailable

Safe for logon storms

Retry while service is unavailable

New

alias =(DESCRIPTION =

(CONNECT_TIMEOUT=90) **(RETRY_COUNT=30)(RETRY_DELAY=3)**

(TRANSPORT_CONNECT_TIMEOUT=3)

(ADDRESS_LIST =

(LOAD_BALANCE=on)

(ADDRESS = (PROTOCOL = TCP)(HOST=primary-scan)(PORT=1521)))

(ADDRESS_LIST =

(LOAD_BALANCE=on)

(ADDRESS = (PROTOCOL = TCP)(HOST=secondary-scan)(PORT=1521)))

(CONNECT_DATA=(**SERVICE_NAME** = gold-cloud)))

OCI Only

Expand scan

Lessons Learned – New Connections

- ALWAYS use services to connect to the database.
 - Do not use the database service or PDB service – these are for administration only, not HA
- Use current client driver (12102) with current or older RDBMS
- Use one DESCRIPTION – more cause long delays connecting
- Set CONNECT_TIMEOUT=90 or higher to prevent logon storms
 - Do not also set JDBC property oracle.net.ns.SQLnetDef.TCP_CONNTIMEOUT_STR as it overrides
- Set LOAD_BALANCE=on per address to expand SCANs
- Do not use Easy*Connect – it has no HA capabilities.

Transparent Planned Maintenance

Applications should see no errors during maintenance.



Transparent Planned Maintenance

- Drains work away from instances targeted for maintenance initiated by FAN
 - Supports well behaved applications using Oracle pools
 - WebLogic Active GridLink, UCP, ODP.NET unmanaged and managed, OCI Session Pool, PHP
 - 3rd party application servers using UCP DataSource: IBM Websphere, Apache Tomcat,..
- Failover at transactional disconnect
 - applications adapted for TAF SELECT with OCI or ODP.Net unmanaged provider
 - applications with own/custom failover

DBA steps - Drain Work at Safe Places

Repeat for each service allowing time to drain

- **Stop service (no –force)**

```
srvctl stop service -db .. -instance .. [-service] .. (omitting -service stops all)
```

- **or Relocate service (no –force)**

```
srvctl relocate service -db .. -service .. -oldinst .. -newinst
```

```
srvctl relocate service -db .. -service .. -currentnode.. -targetnode
```

- **Wait to allow sessions and XA branches to drain.** (see notes)
- **For remaining sessions, stop transactional per service**

```
exec dbms_service.disconnect_session('[service]', DBMS_SERVICE.POST_TRANSACTION);
```

- **Now stop the instances using your preferred method including opatch**
- **For major maintenance operations, disable to prevent restarts**

```
srvctl disable instance -db .. -instance
```

How it works

Applications using ...

Oracle pools or drivers – WebLogic Active GridLink, UCP, ODP.NET managed/unmanaged, OCI, Tuxedo

3rd party App Servers using UCP: IBM WebSphere, Apache Tomcat, RedHat JBoss

DBA Step

srvctl [relocate | stop] service (no -force)

Immediately

New work is redirected by listeners

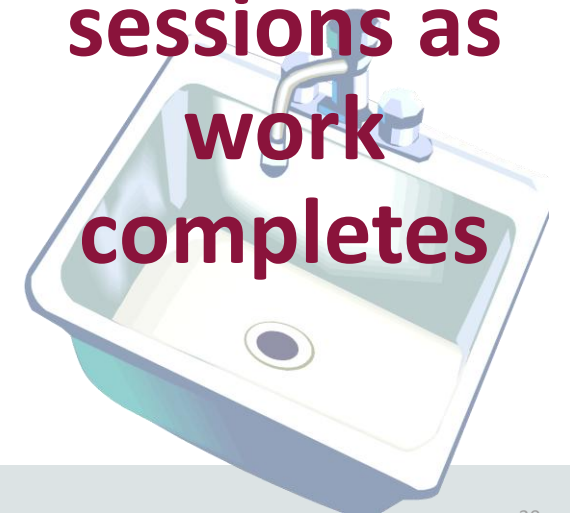
Sessions Drain

Idle sessions are released

Active sessions are released when returned to pools

FAN Planned

**Pools drain
sessions as
work
completes**



Planned Maintenance at NEC

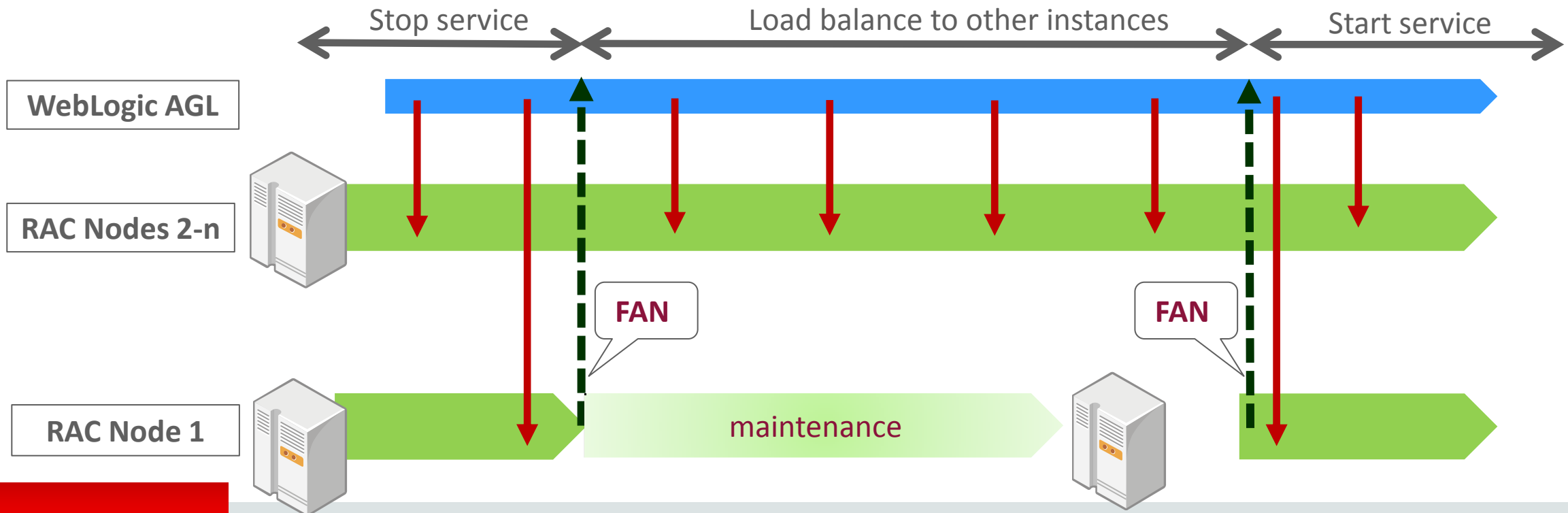
Empowered by Innovation

NEC

WebLogic Active GridLink and Real Application Clusters

1. `srvctl stop services` at one instance & drain (e.g. 5-7s)
2. Instance shutdown
3. Apply patch or change parameter or other maintenance
4. Restart instance & service

No errors, application continues



Planned Maintenance at NEC

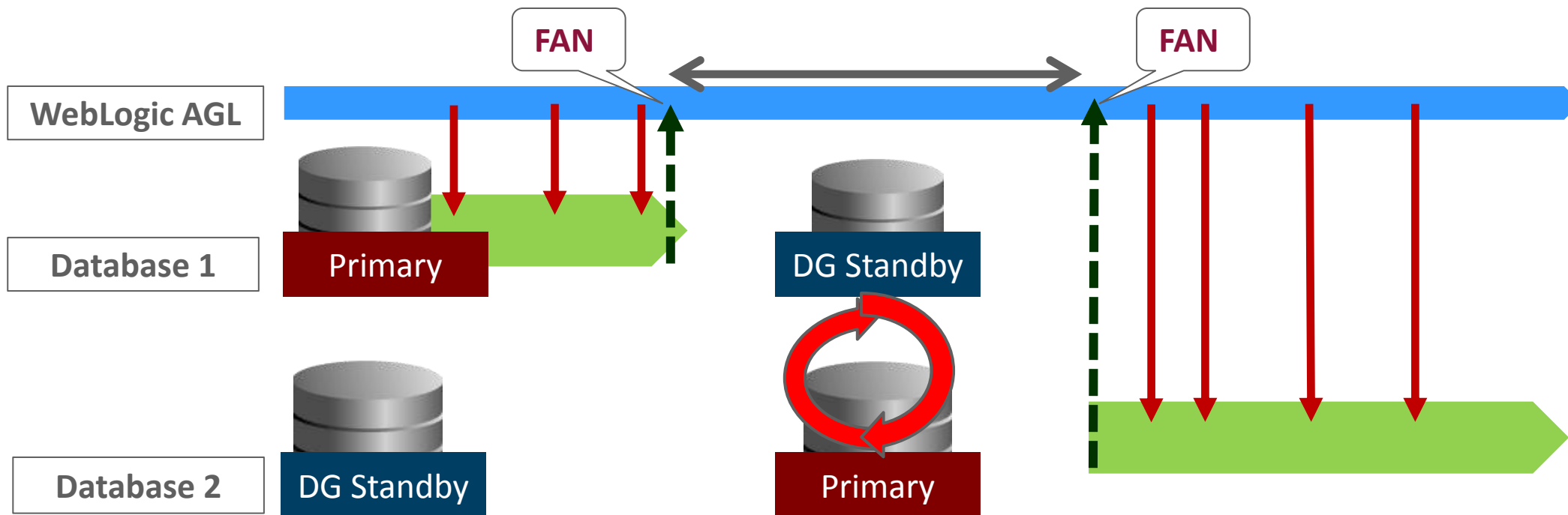
Empowered by Innovation

NEC

WebLogic Active GridLink and Data Guard

1. srvctl stop services on primary site & drain (e.g. 25s – 30s)
2. Data Guard switchover
3. New primary database open, start service, rebalance

No errors, application continues



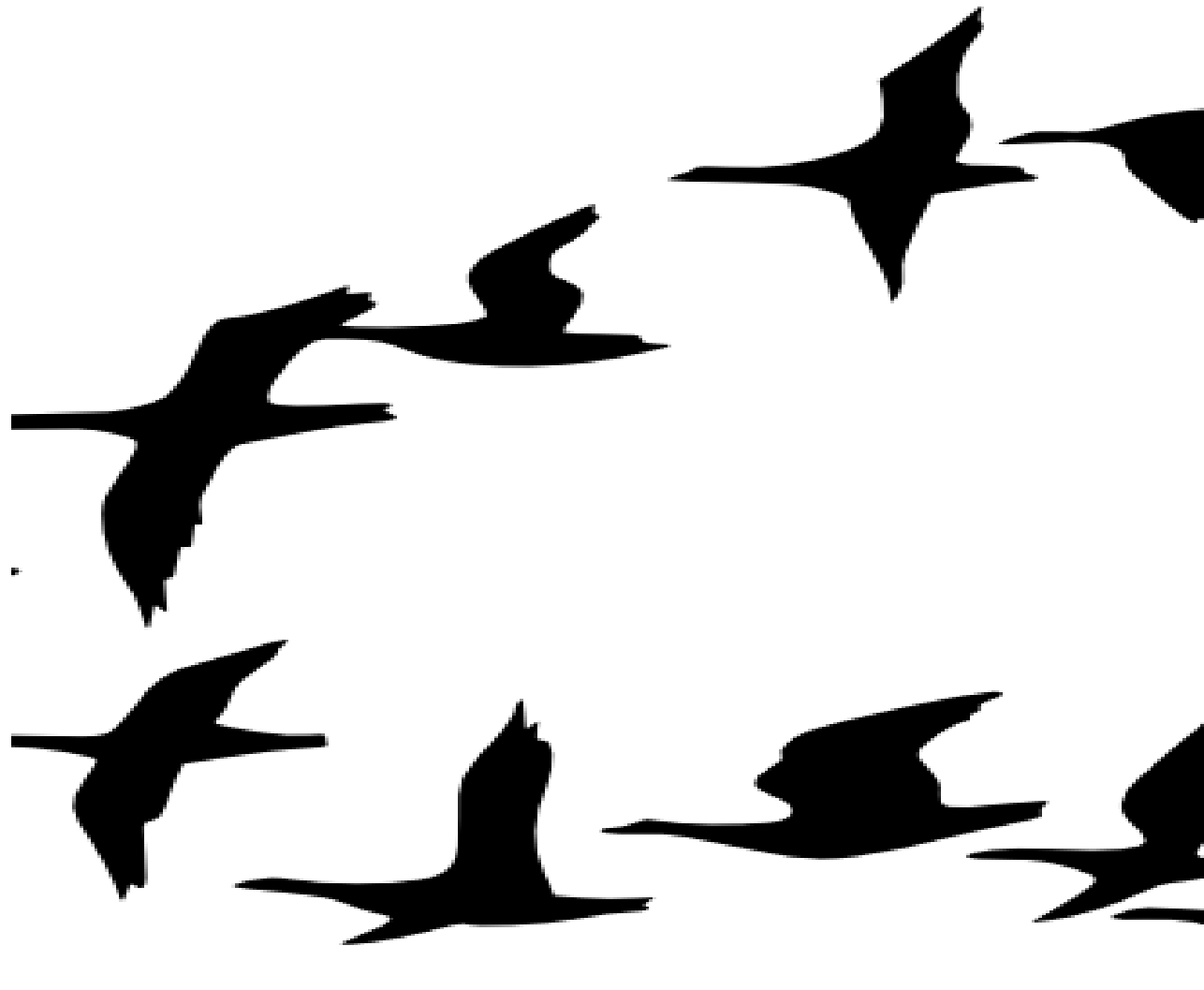
High Availability by Patch Type

| | One- Off | PSU/CPU | Bundle Patch | Patch Set |
|---------------|----------|---------|-----------------|-----------|
| RAC Rolling | 96% | All | Most | No |
| Standby First | 98% | All | All | No |
| Out of Place | All | All | Exadata bundles | No |
| Online - Hot | 82%* | No | No | No |

** Available from 11.2.0.2 onward*

Enterprise Applications

| Application | DBA operation at planned maintenance | Configuration Setting |
|-------------|--------------------------------------|-----------------------|
| Siebel | disconnect sessions transactional | NET |
| PeopleSoft | | NET and TAF SELECT |
| JD Edwards | | NET |
| Informatica | | NET |



Planned Draining Demonstration

Lessons Learned – Planned Maintenance

- Focus on the user experience
 - Allow application work to complete before progressing to stop the instance/database targeted for planned maintenance
 - Drain work invoked by FAN for modern apps that return to pool
 - Use Transactional Disconnect for older-style apps (Disconnect at service level)
- Use TNS with `RETRY_COUNT` & `RETRY_DELAY` to eliminate Net errors
- Check for bug 19783627 if you see unexpected disconnects

New Concepts

Application Failover with Oracle 12c



New Concepts

Database Request: Unit of work submitted by the application

Logical Transaction ID: ID for current, or if no current, next transaction on a session

Reliable Commit Outcome: reliable test that last call to DB was committed and completed ?

Recoverable Error: remove lists of error codes (e.g., ora-3113, ora-1034, ora-1089..)

is_recoverable attribute for JDBC thin

OracleException.IsRecoverable property for ODP.NET

OCI_ATTR_ERROR_IS_RECOVERABLE error handle attribute for OCI

Database Request – UCP example

```
PoolDataSource pds = GetPoolDataSource();
```

```
Connection conn = pds.getConnection();
```

```
PreparedStatement pstmt = ...
```

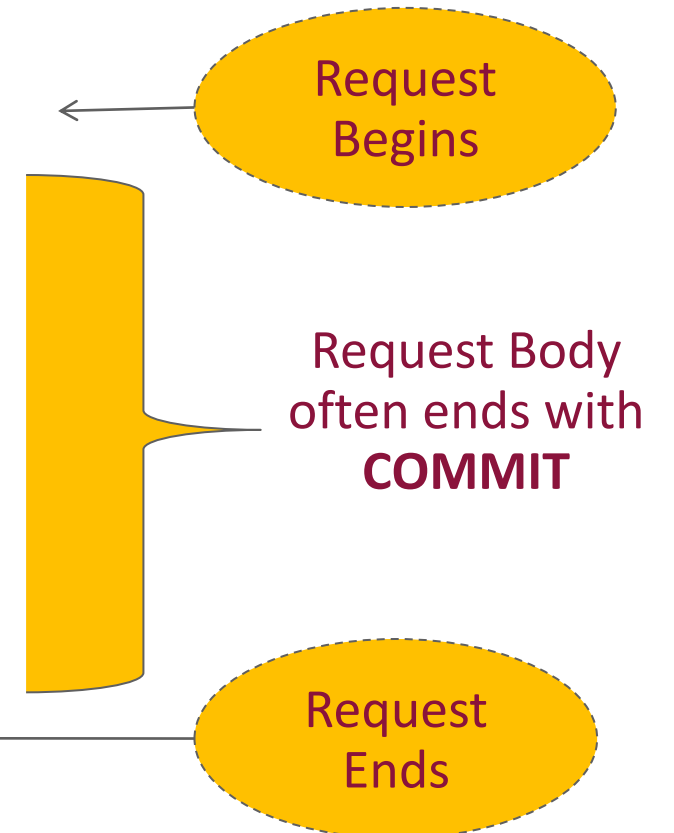
```
...
```

```
SQL, PL/SQL, local calls, RPC
```

```
...
```

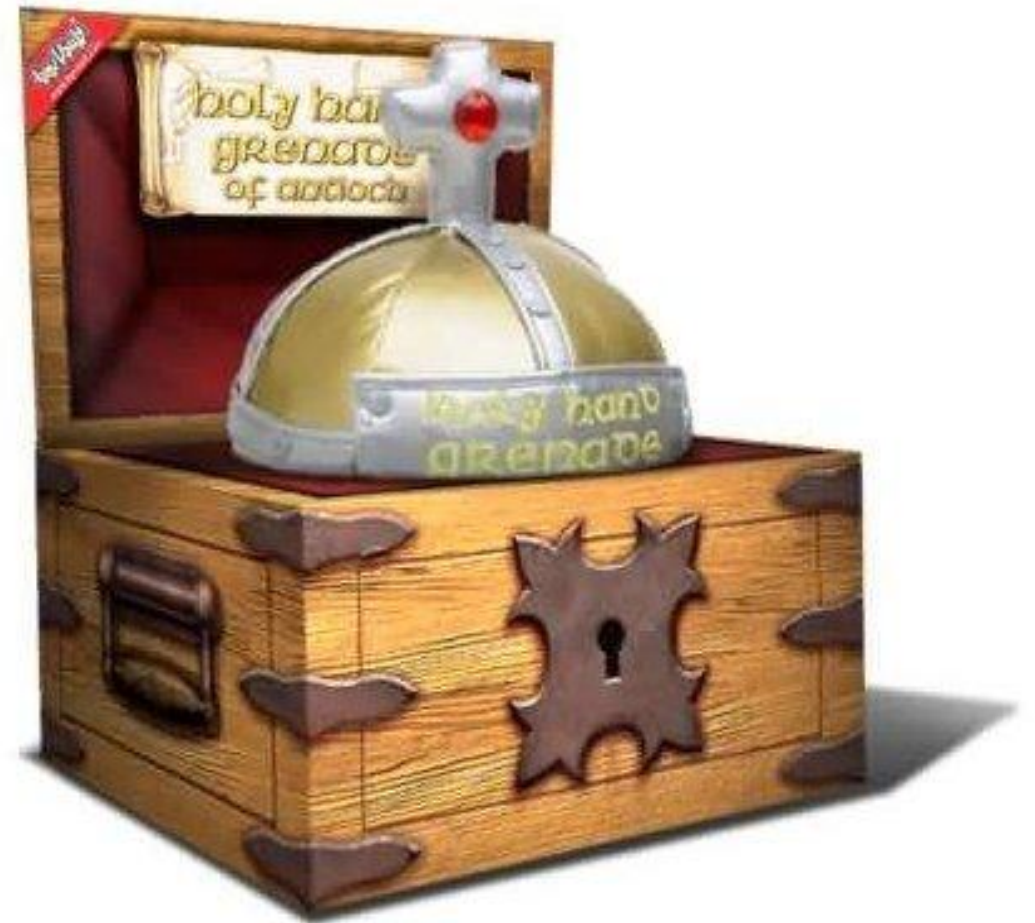
```
conn.commit();
```

```
conn.close();
```



Application Continuity

Unplanned outages should be hidden from applications



Application Continuity

In-flight work continues

- Replays in-flight work on recoverable errors
- Masks most hardware, software, network, storage errors and outages
- Supports JDBC-Thin, UCP, WebLogic Server, 3rd Party Java app servers
- RAC, RAC One, & Active Data Guard
- Improves end user experience

6. ▼ **Estimated Trip Cost**

| | |
|----------------------------------|----------------|
| Flight Total: | 1,536.69 AUD |
| San Francisco, CA - Hotel Total: | 1,800.00 USD ‡ |
| | 1,950.65 AUD |

Trip Total: 3,487.35 AUD ‡
3,218.00 USD

‡ Please note that this total is based on available information. The estimated cost may not include taxes and fees.

- Remember to obtain an original invoice for all your expenses where required under the Global Travel Policy. The invoice should always include the name and address of your Oracle company. Failure to obtain a proper invoice may increase Oracle's costs by up to 25%.

Your order number is 175634. You are protected by Application Continuity

[Purchase Trip](#) [Start Over](#)

Application
Continuity
Demonstration



Phases in Application Continuity

1 – Normal Operation

- Client marks database requests
- Server decides which calls can & cannot be replayed
- Directed, client holds original calls, their inputs, and validation data

2 – Outage Phase 1: Reconnect

- Checks replay is enabled
- Verifies timeliness
- Creates a new connection
- Checks target database is valid
- Uses Transaction Guard to force last outcome

3 – Outage Phase 2: Replay

- Replays captured calls
- Ensures results returned to application match original
- On success, returns control to the application

Exclusions

When replay is not enabled

Application Level

- Default database or default PDB service
- Deprecated, non-standard JDBC classes
- XA in 12.1

Request Level

- Admin actions
 - Alter system
 - Alter database
 - Alter session (subset)
- Active Data Guard with read/write DB links

Target Database

- Databases able to diverge
 - Logical Standby
 - Golden Gate
 - PDB Clone

Steps to use Application Continuity

| Check | What to do |
|-------------------------|---|
| Request Boundaries | UCP, WebLogic, and standard 3 rd Party App servers – return connections to pool |
| JDBC Deprecated Classes | Replace non-standard classes (MOS 1364193.1) |
| Side Effects | Use disable API if a request has a call that should not be replayed |
| Callbacks | Register a callback for applications that change state outside requests For WebLogic and UCP labels – do nothing |
| Mutable Functions | Grant keeping mutable values, e.g. <code>sequence.nextval</code> |

Grant Mutables

Keep original function results at replay

For owned sequences:

```
ALTER SEQUENCE.. [sequence object] [KEEP|NOKEEP];
```

```
CREATE SEQUENCE.. [sequence object] [KEEP|NOKEEP];
```

Grant and Revoke for other users:

```
GRANT [KEEP DATE TIME | KEEP SYSGUID].. [to USER]
```

```
REVOKE [KEEP DATE TIME | KEEP SYSGUID][from USER]
```

```
GRANT KEEP SEQUENCE on [sequence object] [to USER] ;
```

```
REVOKE KEEP SEQUENCE on [sequence object] [from USER]
```


Callbacks

For applications that set state outside database requests

- WebLogic and UCP Connection Labeling
 - Do nothing
- Custom
 - Register Connection Initialization Callback
 - Sets initial state for a session at BOTH runtime and replay
 - Available with WebLogic, UCP, JDBC-Thin

Configuration at Database

Set Service Attributes

FAILOVER_TYPE = TRANSACTION for Application Continuity

Review the service attributes:

COMMIT_OUTCOME = TRUE for Transaction Guard

REPLAY_INITIATION_TIMEOUT = 300 after which replay is canceled

FAILOVER_RETRIES = 30 for the number of connection retries per replay

FAILOVER_DELAY = 3 for delay in seconds between connection retries

Configuration at Client

Use JDBC Replay Data Source

At WebLogic Console or UCP, or your own property file -

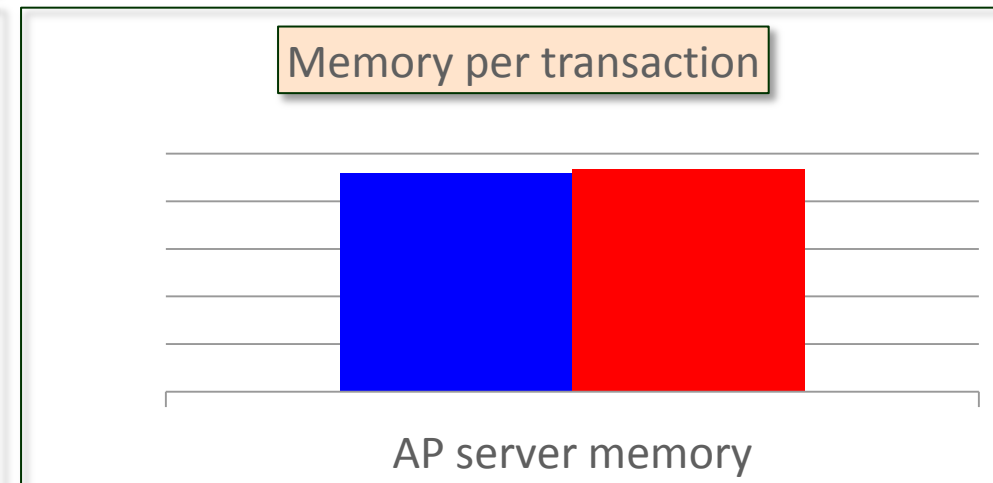
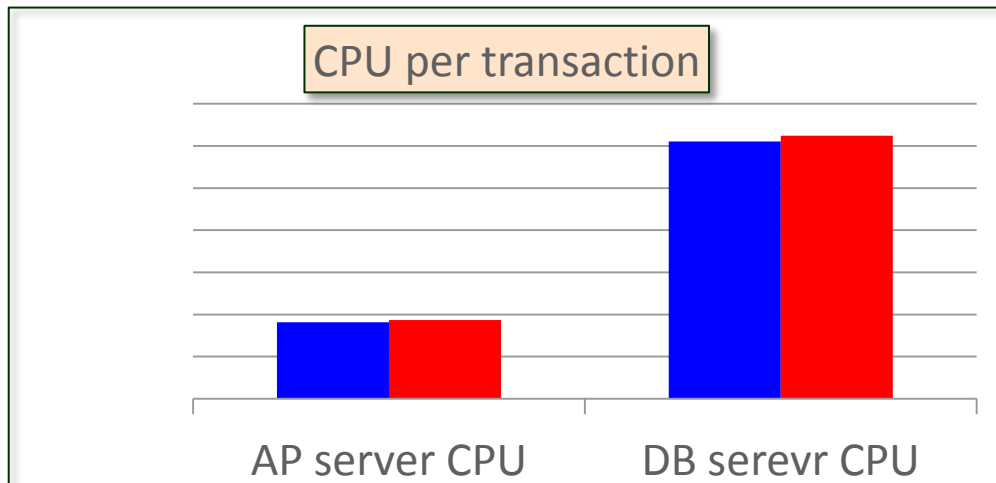
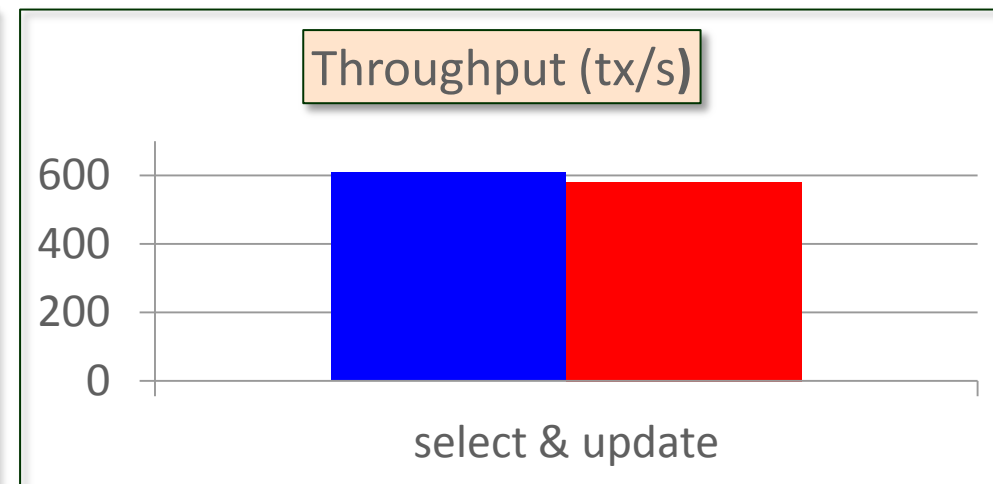
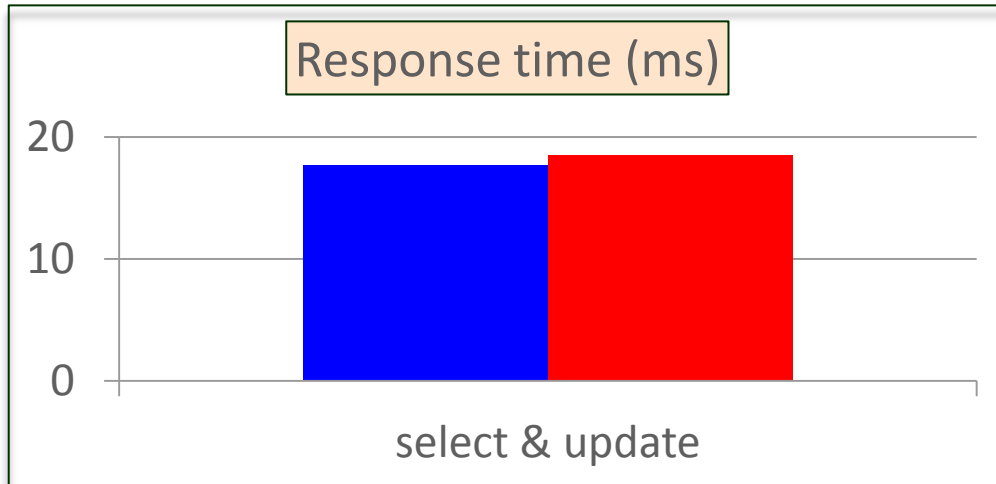
Select new 12.1 datasource

```
replay datasource=oracle.jdbc.replay.OracleDataSourceImpl
```

Application Continuity Performance

WebLogic Server Active GridLink and Real Application Clusters

Empowered by Innovation



■ AC OFF
■ AC ON

MedRec Application



Killing Sessions - Extended

DBA Command

Replays

```
srvctl stop service -db orcl -instance orcl2 -force
```

YES

```
srvctl stop service -db orcl -node rws3 -force
```

YES

```
srvctl stop service -db orcl -instance orcl2 -noreplay -force
```

```
srvctl stop service -db orcl -node rws3 -noreplay -force
```

```
alter system kill session ... immediate
```

YES

```
alter system kill session ... noreplay
```

```
dbms_service.disconnect_session([service], dbms_service. noreplay)
```

AC Statistics

Supported for Oracle JDBC replay driver

Statistics are client-side, cumulative per-connection or total for all pooled connections using `oracle.jdbc.replay.ReplayableConnection`

`ReplayableConnection.getReplayStatistics(FOR_CURRENT_CONNECTION)` returns statistics for current connection

`ReplayableConnection.getReplayStatistics(FOR_ALL_CONNECTIONS)` returns statistics for all connections in the pool

`ReplayableConnection.clearReplayStatistics(StatisticsReportType)` clears replay statistics – per connection or all connections

Runtime

TotalRequests = 1

TotalCompletedRequests = 1

TotalCalls = 19

TotalProtectedCalls = 19

Replay

TotalCallsAffectedByOutages = 3

TotalCallsTriggeringReplay = 3

TotalCallsAffectedByOutagesDuringReplay = 0

SuccessfulReplayCount = 1

FailedReplayCount = 0

ReplayDisablingCount = 0

TotalReplayAttempts = 3

Lessons Learned

- Return connections to the connection pool between requests.
- Set `REPLAY_INITIATION_TIMEOUT`, `RETRY_COUNT`, and `RETRY_DELAY`
- Consider mutable values. Think of these in terms of delayed execution.
- If the application sets values after creating a connection outside the application – repeat these settings in the callback.
- If the application is using XA datasource – check why. Most apps do not need it.
- If experimenting and using `V$instance` etc, put these in the callback.

Transaction Guard

Unplanned outages should be hidden from applications



Transaction Guard

First RDBMS to preserve **COMMIT** Outcome

Reliable transaction outcome after outages

- Allows applications to deal with failures correctly
- Without Transaction Guard, retrying can cause logical corruption
- Application Continuity uses Transaction Guard
- API available with JDBC-thin, OCI/OCCI, ODP.NET

Estimated Trip Cost

| | |
|----------------------------------|-----------------------|
| Flight Total: | 1,536.69 AUD |
| San Francisco, CA - Hotel Total: | 1,800.00 USD ‡ |
| | 1,950.65 AUD |
| Trip Total: | 3,487.35 AUD ‡ |
| | 3,218.00 USD |

Please note that this total is based on available information. The estimated cost may not include taxes and fees.

- Remember to obtain an original invoice for all your expenses where required under the Global Travel Policy. The invoice should always include the name and address of your Oracle company. Failure to obtain a proper invoice may increase Oracle's costs by up to 25%.

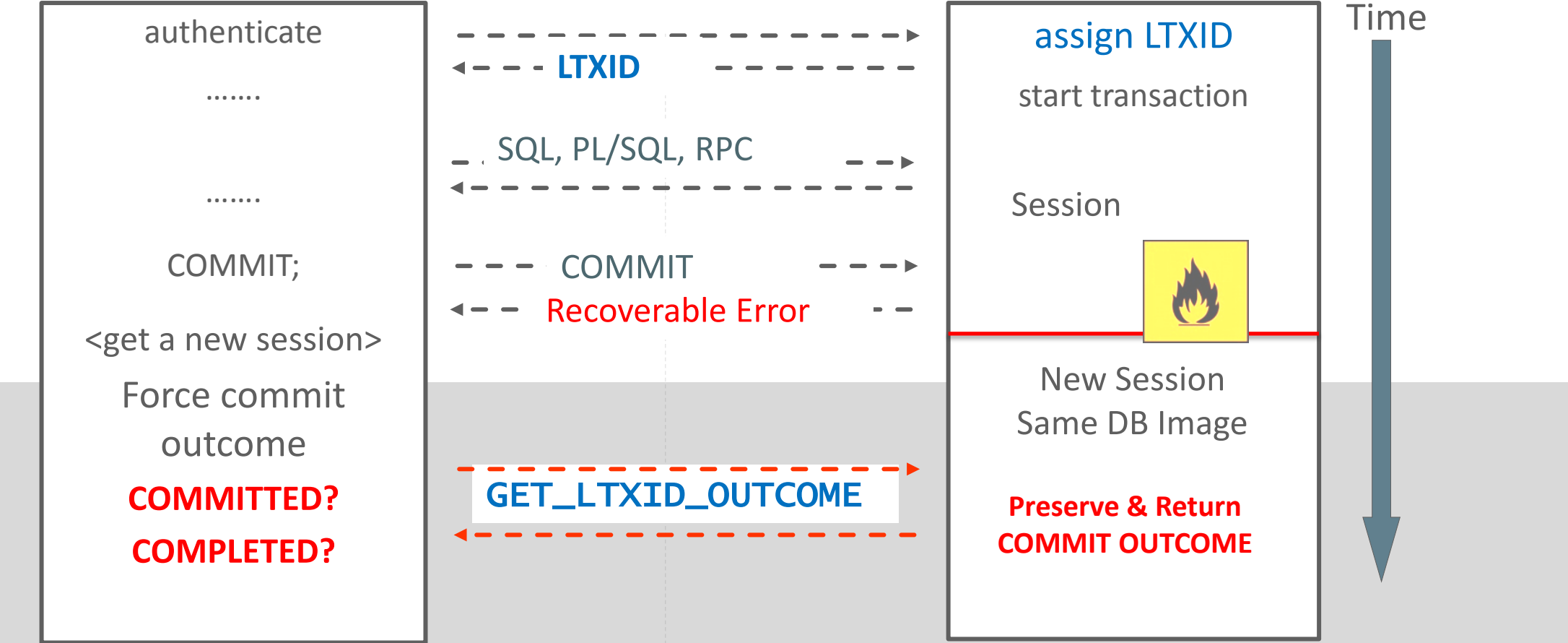
Your order will be processed shortly with a price guarantee. You are protected by Transaction Guard.

[Purchase Trip](#) [Start Over](#)

How Transaction Guard Works

Oracle 12c Drivers

Oracle 12c Database(s)



Transaction Coverage

Inclusions 12.1

Local

Commit on Success (auto-commit)

Distributed and Remote

DDL, DCL, parallel DDL

PL/SQL with embedded COMMIT

PL/SQL with COMMIT as last call

Read-only (allowed for)

Exclusions 12.1

XA

Active Data Guard with R/W links to commit at primary

Database Target - Coverage

Inclusions 12.1

- Single Instance Oracle RDBMS
- RAC One Node
- Real Application Clusters
- Data Guard
- Active Data Guard
- Multitenant including unplug/plug

Exclusions Database Failed Over To -

- Logical Standby
- Golden Gate
- PDB Clones

Forcing Commit Outcome

GET_COMMIT_OUTCOME forces the commit outcome, returning -

- COMMITTED
 - TRUE the user call executed at least one commit
 - FALSE the user call is uncommitted and stays that way
- USER_CALL_COMPLETED
 - TRUE the user call ran to completion.
 - FALSE the user call is not known to have finished
 - e.g. use if expecting return data – commit on success, commit embedded in PL/SQL

Use Case - Unambiguous Outcome

Database session outage

FAN aborts dead session FAST
Application receives an error

If “*recoverable error*” then

Add this part in the error handling routine

Get last LTXID from dead session
Obtain a new database session

// Force commit outcome

execute **DBMS_APP_CONT.GET_LTXID_OUTCOME** with last LTXID

If **committed** then {

 process committed ; // e.g. let user know it committed

 if user_call_completed then application may continue

 else application may not be able to continue}

Else process **uncommitted** // e.g. let user know its safe to resubmit

Server-side settings for Transaction Guard

- **On Service**
 - COMMIT_OUTCOME
 - Values – TRUE and FALSE
 - Default – FALSE
 - Applies to new sessions
- **GRANT EXECUTE ON DBMS_APP_CONT TO <user>;**

Transaction Guard – Key Takeaway

First RDBMS to preserve commit outcome

- Users should not see misleading errors when a transaction really did commit.
- Driver receives an LTXID at authentication and on every commit.
- Once the commit outcome is returned, **the result never changes.**
- Safe for applications and mid-tiers to return success or resubmit themselves.

Success Stories Out of the Box



Unplanned Failover with Application Continuity

WebLogic Active GridLink and Real Application Clusters

Empowered by Innovation



BEFORE

AFTER

DB 11gR2+WLS Generic DS
Error
AP wait time:1s

DB12c+ GridLink+AppCont
No errors, App Continues
AP wait time:1s

DB 11gR2+WLS Generic DS
TIMEOUT
900s (TCP keep-alive)

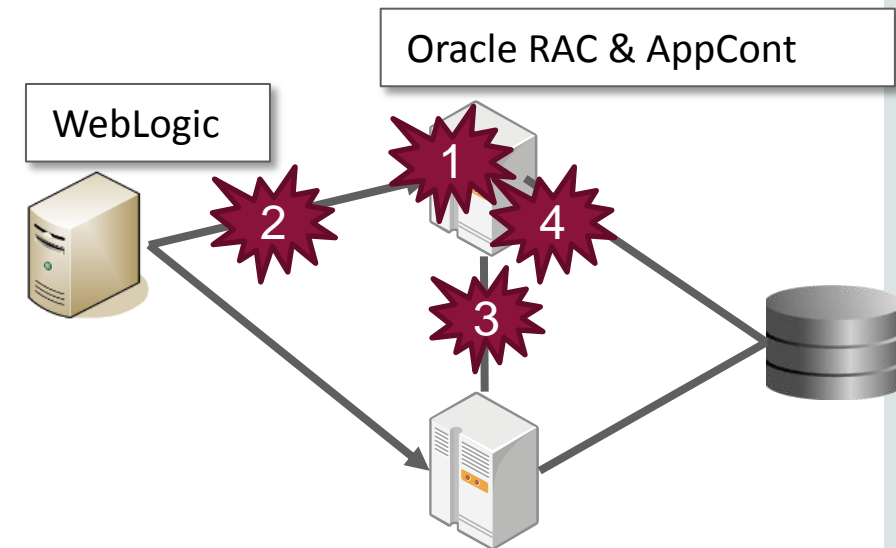
DB12c+ GridLink+AppCont
No errors, App Continues
AP wait time:1s

DB 11gR2+WLS Generic DS
Error
AP wait time: 30s

DB12c+ GridLink+AppCont
No errors, App Continues
AP wait time:30s

DB 11gR2+WLS Generic DS
Hang
AP wait time: minutes

DB12c+ GridLink+AppCont
+ NEC Monitor :
No errors, App Continues



1. Instance down
2. Public network down
3. Interconnect down
4. Background process hang

Planned Failover with FAN

WebLogic Server Active GridLink, RAC and Data Guard

Empowered by Innovation



| DBA Operation | Maintenance | Result | Time to Drain all Sessions |
|-----------------------|---------------------------|--------------------------|----------------------------|
| RAC rolling | PSU apply using opatch | No errors to application | 5s |
| RAC rolling | Instance parameter change | No errors to application | 7s |
| Data Guard switchover | Site maintenance | No errors to application | 29s |
| Data Guard switchover | Site maintenance fallback | No errors to application | 25s |

Planned and Unplanned Failover

RAC One Node, IBM WebSphere, Universal Connection Pool



| Maintenance | Result | Time allowed |
|---|--------------------------|---------------------|
| Planned with FAN + Net | No errors to application | 4 hours |
| Unplanned with Application Continuity + Net | No errors to application | 10 minutes |

Runtime, Planned, & Unplanned

ODP.NET Unmanaged Provider, RAC, and Data Guard

EPSILON®

Change

Improvement

11204 Client to 11204 DB (pending ODAC 12102)

Latest client software

Return connections to ODP.Net pool between requests

Connection lifetime longer than 24 hours

Min and max connections equal

Reduction in connection usage by 40-50% compared to dedicated connection model

TNS names with retry_count and timeouts

No errors for incoming work

No errors to apps during service failover and switchover

Login storms eliminated

FAN planned to drain connections for planned maintenance

No errors to apps at planned maintenance with RAC and with Data Guard

FAN + TAF SELECT to failover. TAF callbacks for transactional to rollback

Errors at unplanned reduced to transactions only

Runtime, Planned, & Unplanned

ODP.NET Unmanaged Provider, RAC, and Data Guard

EPSILON®

| Database Method | Client Method | Example | Result | Time to Drain Sessions |
|----------------------------|--------------------------------------|-----------------------|--------------------------|------------------------|
| RAC rolling upgrade/change | Drain with FAN + TNS | PSU / CPU | No errors to application | 5s |
| Data Guard Switchover | Drain with FAN + TNS | Standby first PSU/CPU | No errors to application | 25s |
| RAC Failover | Failover with FAN + TNS + TAF SELECT | Node outage | Errors for transactions | 5s |
| Data Guard Failover | Failover with FAN + TNS + TAF SELECT | Site outage | Errors for transactions | - |

Anonymous – Unplanned with AC

WebLogic Server Active GridLink and Real Application Clusters

| Workloads | Replay | Reason |
|--|--------------|---|
| concurrent OLTP with DML | succeeds | DML replays concurrently |
| concurrent OLTP query and DML mix | succeeds | Queries replay at original SCN |
| concurrent OLTP with select for update and DML mix | most succeed | Rejections only when unable to restore original |

For Developers : Application Continuity offloads the challenging work of transaction resubmission during failure events, allowing developers to focus on functionality.

Christo Kutrovsky – ATCG Principal Consultant, Oracle ACE

For Enterprise Architects : Application Continuity is a major step towards the holy grail of a continuously available, consistent, and highly performing database cluster

Marc Fielding – ATCG Principal Consultant, Oracle

Empowered by Innovation

NEC

The combinatorial solution with Application Continuity, Real Application Clusters, Data Guard, WebLogic Server Active GridLink and NEC hardware and middleware enables us to provide incredibly high available system for our Mission Critical customers. This solution will become our primary solution for cloud and big data areas.

Yuki Moriyama

Senior Manager, NEC Corporation

Safe Harbor Statement

The preceding is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

Hardware and Software Engineered to Work Together

ORACLE®