No Outages: Transparent Application Continuity

Carol Colrain, Ian Cookson
Oracle Database Development
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. What is Continuous Availability?
2. Maintenance with no Impact
3. Transparent Application Continuity
4. Configuring for Best Results
5. Application Continuity at CME
Continuous Availability

Continuous Availability is not Absolute Availability

Probable outages and maintenance events at the database level are masked from the application, which continues to operate with no errors and within the specified response time objectives while processing these events.

Key points:

• Planned maintenance and likely unplanned outages are hidden from applications
• There is neither data loss nor data inconsistency, guaranteed
• Majority of work (% varies by customer) completes within recovery time SLA
• May appear as a slightly delayed execution

Many customers are achieving Continuous Availability Today
From High Availability to Continuous Availability

**High Availability**
- Minimizes downtime
- In-flight work is lost
- Rolling maintenance at DB
- Predictable runtime performance
- Errors may be visible
- Designed for single failure
- Basic HA building blocks

**Continuous Availability**
- No downtime for users
- In-flight work is preserved
- Maintenance is hidden
- Predictable performance
- Errors visible only if unrecoverable
- Designed for multiple failures
- Builds on top of HA
What kinds of outages?

Planned Maintenance
Patches
Repairs
Upgrades
Changes

Unplanned Outages

Unpredictable Response & Throughput

Site Disasters

Data Corruption

Human Errors

Which outage classes does your business need to handle?
New Concept - Request
All Oracle 12c Pools and in JDK9

PoolDataSource pds = GetPoolDataSource();
Connection conn = pds.getConnection();
PreparedStatement pstmt = ...

... SQL, PL/SQL, local calls, RPC ...

conn.commit();
conn.close();
Maintenance without Impact

There is no reason for users to see downtime during scheduled database maintenance

- Service is unavailable
- Application owners unable to agree maintenance windows
- Long running jobs see errors
- DBA’s and engineers work off hours
- Application and middleware components need to be restarted

Family Holidays website is down
"Our online transaction services are currently unavailable. Our server may be temporarily down or we may be performing routine maintenance functions scheduled every Sunday from 12 a.m. to 5 a.m. (Eastern Standard Time). We apologize for any inconvenience."
Drain sessions before maintenance

Move services first

Drain connections where applications don’t notice

Many Drain Points

- Connection Pools
- Connection tests
- Web requests
- Transaction boundaries
Tip: DBA steps to Drain Sessions – Since 10.2

Repeat for each service allowing time to drain

• Stop service
  
srvctl stop service -db .. -instance .. -service .. (omit service)

• Relocate service
  
srvctl relocate service -db .. -service .. -oldinst .. -newinst
  srvctl relocate service -db .. -service .. -currentnode .. -targetnode

Wait for sessions to drain...

For remaining sessions, stop transactional
  
exec dbms_service.disconnect_session( '\... your service ..\',
  DBMS_SERVICE.POST_TRANSACTION);

Stop the instances using your preferred tool
## Use Oracle Pools – Full Lifecycle

### Drain and Rebalance

<table>
<thead>
<tr>
<th>Applications using ...</th>
<th>Oracle – WebLogic Active GridLink, UCP, ODP.NET managed and unmanaged, OCI Session Pool, Tuxedo, CMAN TDM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3rd party App Servers using UCP: IBM WebSphere, Apache Tomcat, NEC WebOTX, Red Hat JBoss, Spring</td>
</tr>
</tbody>
</table>

| DBA Step               | `srvctl [relocate|stop] service -drain_timeout` |
|------------------------|-----------------------------------------------|

<table>
<thead>
<tr>
<th>Sessions Drain</th>
<th>Immediately new work is redirected</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Gradually</td>
</tr>
<tr>
<td></td>
<td>Active sessions are released when returned to pools</td>
</tr>
</tbody>
</table>

*FAN Planned*
UCP with other Java-based Application Servers
A simple data source replacement

- IBM WebSphere
- Apache Tomcat
- NEC WebOTX
- Red Hat Jboss
- Hibernate
- Spring
- Your own

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
DBA Operations Simplified

• **Group operations** `pdb`, instance, node, or database

• New service attributes

  - `drain_timeout` (seconds)
  - `stopoption` (immediate, transactional)
DBA commands: Group commands

Relocate all services by database/node/pdb

```bash
srvctl relocate service -database -instance -drain_timeout.. -stopoption
[immediate|transactional]
srvctl relocate service -node . -drain_timeout.. -stopoption
srvctl stop service -pdb . -drain_timeout.. -stopoption
```

Start/Stop everything at a node

```bash
srvctl stop service -node <node_name> -drain_timeout.. -stopoption
srvctl stop database -node <node_name> -drain_timeout.. -stopoption
```

Data Guard Switchover

```bash
Switchover to <db_resource_name> [wait [xx]];
```
Drain... Connect... Failover

(1) Drain Node 1
(2) Connect to Node 2
(3) Terminate in-flight requests
(4) Replay in-flight requests

TPM

Node 1

Node 2

Drain Timeout

Time
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.
Application Continuity

In-flight work continues

- Replays in-flight work on recoverable errors
- Masks hardware, software, network, storage, session errors and timeouts
- 12.1 JDBC-Thin, UCP, WebLogic Server, 3rd Party Java application servers
- 12.2 OCI, ODP.NET unmanaged, JDBC Thin on XA, Tuxedo*, SQL*Plus, Pro*
- **18c is Transparent (TAC)**
Demonstration
– Standardize on TAC
Application Continuity Explained

**Normal Operation**
- Client marks requests: explicit and discovered.
- Server *tracks session state*, decides which calls to replay, disables side effects.
- Directed, client holds original calls, their inputs, and validation data.

**Failover Phase 1: Reconnect**
- Checks replay is enabled
- Verifies timeliness
- Creates a new connection
- Checks target database is legal for replay
- Uses Transaction Guard to guarantee commit outcome

**Failover Phase 2: Replay**
- Restores and verifies the session state
- Replays held calls, restores mutables automatically
- Ensures results, states, messages match original.
- On success, returns control to the application

New with 18c
# Using Transparent Application Continuity

<table>
<thead>
<tr>
<th>Request Boundaries</th>
<th>Discovered and Advance Continuously</th>
</tr>
</thead>
<tbody>
<tr>
<td>Session State</td>
<td>Restored and Verified</td>
</tr>
<tr>
<td>Side Effects</td>
<td>Not replayed</td>
</tr>
<tr>
<td>SYSDATE, Sequences..</td>
<td>Automatic for SQL, Grant for PL/SQL</td>
</tr>
<tr>
<td>Coverage</td>
<td>Always know your protection level</td>
</tr>
</tbody>
</table>
Request Boundaries Advance Continuously

• Request boundaries advanced automatically when state is restorable*
• Capture re-enables, if disabled
• Smaller capture set means faster recovery
• **Oracle pool and return to pool are still best practice**

* Transparent Application Continuity for Java, OCI coming
Session State must be Correct to Replay

• Restore session states before replaying
  – `FAILOVER_RESTORE` on your service

• TAC does not replay if session states differ

• AC allows you to add complex states before replay starts
  – AC does not replay if standard states differ
Side Effects Not Replayed

**TAC** – stops capture automatically until next enable point

TAC decides if any requests should not be replayed, e.g.

- UTL_HTTP
- UTL_URL
- DBMS_FILE
- DBMS_FILE_TRANSFER
- UTL_SMTP
- UTL_TCP
- UTL_MAIL
- EXTPROC

Side Effects Disabled

Customized: use AC
Handles SYSDATE, SYSTIMESTAMP, Sequences ...

Keeps and Restores Automatically

Function results kept for SQL
Grant keeping results for PL/SQL

For owned sequences:

ALTER SEQUENCE.. [sequence] [KEEP|NOKEEP]
CREATE SEQUENCE.. [sequence] [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][to USER]
REVOKE KEEP SEQUENCE on [sequence][from USER]
Always Know Your Protection Level

• AWR, system, session, service stats
  – Requests completed per second
  – User calls in request
  – Protected user calls

<table>
<thead>
<tr>
<th>Statistic</th>
<th>Total</th>
<th>per Second</th>
<th>per Trans</th>
</tr>
</thead>
<tbody>
<tr>
<td>cumulative requests</td>
<td>177,406</td>
<td>49.2</td>
<td>5.0</td>
</tr>
<tr>
<td>cumulative user calls in request</td>
<td>493,329</td>
<td>136.8</td>
<td>13.8</td>
</tr>
<tr>
<td>cumulative user calls protected</td>
<td>493,329</td>
<td>136.8</td>
<td>13.8</td>
</tr>
</tbody>
</table>
# Detailed Protection Report when needed

## Application Continuity Summary

<table>
<thead>
<tr>
<th>Outage Type</th>
<th>Status</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coverage checks</td>
<td>TotalRequest = 398&lt;br&gt;PASS = 389&lt;br&gt;WARNING = 0&lt;br&gt;FAIL = 9</td>
<td>[PASS] Trace file name = WEB_ora_124333.trc Row number = 10909&lt;br&gt;SERVICE NAME = (WEB_SSL_SERVICE) MODULE NAME = (JDBC Thin Client) ACTION NAME = POST&lt;br&gt;CLIENT ID = null&lt;br&gt;Coverage(%) = 100 ProtectedCalls = 1 UnProtectedCalls = 0</td>
</tr>
<tr>
<td>PASS</td>
<td>[PASS] Trace file name = WEB_ora_19757.trc Row number = 36978&lt;br&gt;SERVICE NAME = (WEB_SSL_SERVICE) MODULE NAME = (JDBC Thin Client) ACTION NAME = CARD&lt;br&gt;CLIENT ID = null&lt;br&gt;Coverage(%) = 100 ProtectedCalls = 24 UnProtectedCalls = 0</td>
<td></td>
</tr>
<tr>
<td>FAIL</td>
<td>[FAIL] Trace file name = WEB_ora_19757.trc Row number = 481193&lt;br&gt;SERVICE NAME = (WEB_SSL_SERVICE) MODULE NAME = (JDBC Thin Client) ACTION NAME = null&lt;br&gt;CLIENT ID = null&lt;br&gt;Coverage(%) = 20 ProtectedCalls = 1 UnProtectedCalls = 4</td>
<td></td>
</tr>
<tr>
<td>FAIL</td>
<td>[FAIL] Trace file name = WEB_ora_19757.trc Row number = 14203&lt;br&gt;SERVICE NAME = (WEB_SSL_SERVICE) MODULE NAME = (JDBC Thin Client) ACTION NAME = null&lt;br&gt;CLIENT ID = null&lt;br&gt;Coverage(%) = 33 ProtectedCalls = 2 UnProtectedCalls = 1</td>
<td></td>
</tr>
</tbody>
</table>
Applications see no errors during outages

Standardize on Transparent Application Continuity

Hides errors, timeouts, and maintenance

No application knowledge or changes to use

Rebuilds session state & in-flight transactions

Adapts as applications change: protected for the future
Chicago Mercantile Exchange
CME Group Overview

CME Group is the world's leading and most diverse derivatives marketplace bringing together those who need to manage risk or those that want to profit by accepting it.

• Operating Multiple Exchanges – CME, CBOT, Nymex and COMEX
• Trade hundreds of products across the globe on a single platform
• Average daily volume of 15.6 million contracts

• CME Clearing – matches and settles all trades and guarantees the creditworthiness of every transaction
• Cleared more than 3.9 billion contracts with a value exceeding $1 quadrillion
• Highest Volume Day – 44.5 million contracts after the election
CME HIGH AVAILABILITY OVERVIEW

Requirements

• Critical DB’s – 10 second to SLA
• Component Failure Cannot cause DR Event
• 24X7 Application up time
  • Including Planned Maintenance
• RPO – 30 seconds (Disaster Only)
• RTO – 2 hours (Disaster Only)

Solution

• Exadata
  • Addresses Performance
  • Allows Consolidation
  • Reduces recovery time (component failure)
• Active Data Guard
• Application Continuity – Planned/unplanned
WHY CME IS ADOPTING APPLICATION CONTINUITY

• Database Outages cause in-flight work to be lost

• A Database Outage can effect many applications concurrently due to schema consolidation

• Critical Applications are becoming 24x7 – These are referential applications

• Database planned downtime on behalf of patching is exceedingly harder to schedule due to shrinking maintenance windows.

• Avoid dedicating maintenance windows to the database group

• Applications work together as a system. It can take several hours to start and normalize
### Failure Proofing Applications Is Hard

<table>
<thead>
<tr>
<th>What’s Hard</th>
<th>What’s Today’s Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hanging on TCP/IP Timeouts – Application is not aware of an issue because</td>
<td>- FAN – Fast Application Notification</td>
</tr>
<tr>
<td>there has been no ack for the last operation</td>
<td>- FCF – Fast Connection Failover</td>
</tr>
<tr>
<td></td>
<td>- ONS – Notification Services</td>
</tr>
<tr>
<td></td>
<td>These features work together to overcome TCP hangs</td>
</tr>
<tr>
<td>Reconnecting to surviving nodes or standby database after failure</td>
<td>Application Continuity automatically performs connection retries all configurable</td>
</tr>
<tr>
<td></td>
<td>in the connection string</td>
</tr>
<tr>
<td>Assuring any in-flight transactions were committed to the database.</td>
<td>Application Continuity features handle this transparently. Transactions are crosschecked</td>
</tr>
<tr>
<td>Confidence leaving applications live during planned Database Maintenance</td>
<td>and replayed safely</td>
</tr>
<tr>
<td></td>
<td>AC has proven to be resilient at CME.</td>
</tr>
</tbody>
</table>
Normal Operation

- All OLTP services configured as 1 active, rest available
- Over 400 services across environment
- Over 100 applications
- Node capacity actively managed

- Most Application Servers “Lie in Wait”
- Critical Applications are connected in a RO mode
Planned Maintenance

- Exadata Full Stack Patching takes 4 hours at best
- CME does not do rolling patches (duration too long)
- AC allows apps to stay up and undergo updates while patching happens.
Planned Maintenance

- DR is always patched first
- Applications in DR are taken offline
- Normal change window applies
- Application changes in PROD coincide with DR patching
Planned Maintenance

- Local Standby databases are patched after DR
- Patching the local standby database does not impact running application
- Patched during normal maintenance window
- Application changes and testing can continue
Planned Maintenance – Database As A Service

- AC compliant applications stay running and available
- Non compliant applications are stopped and restarted (Transition period)
- A database switchover is performed
- An LDAP job modifies connection strings for non compliant apps

- Non compliant apps are restarted
- Changes and testing continues during maintenance window
- Process repeated for fail back
UNPLANNED OUTAGES

- Node 1 fails
- All services fail to available instance (2 illustrated)
- Application connections follow service location using Application Continuity
UNPLANNED OUTAGES

- What if the whole Exadata Fails?
- At CME – this is not allowed to cause a DR event
UNPLANNED OUTAGES

• Catastrophic Data Center Failure
  • Uncontrolled network outage (All HA FAILS)
  • Physical Damage to building
  • EXA 1 and EXA 2 fail in same week

• Critical Apps Up for customer RO access
  • Databases are converted – Apps convert to RW
  • All apps started - < 2 hours
  • All automated
CME Best Practices

- Good test environment that mirrors production
- Credible HA and DR testing methodology on a mandatory interval commensurate with your change rate
- Application simulation for testing that is realistic
- Capacity Planning – keep utilization of servers <50%
- Client Interrupted using FAN and FCF
- Time Based Failover, supported by Application Continuity
- No Single Points of Failure
- Strong Change Control
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.
Integrated Cloud
Applications & Platform Services