ORACLE CODE ONE

# Break New Ground

**San Francisco**
September 16–19, 2019

# Eliminate Application Downtime with Oracle Database and .NET

**Alex Keh**

Senior Principal Product Manager
Oracle
September 17, 2019

## Agenda

- Application Continuous Availability
- Fast Application Notification
- Planned Maintenance Outages
- Unplanned Outages

# Application Continuous Availability

## From HA to Continuous Availability

| **High Availability** | **Continuous Availability** |
|---|---|
| Minimizes downtime | Removes user perspective downtime |
| In-flight work is lost | In-flight work is preserved |
| Rolling maintenance at DB | Maintenance is hidden |
| Predictable runtime performance | Predictable outage performance |
| Errors may be visible | Errors only if unrecoverable |
| Design for single failure | Handles multiple concurrent failures |
| Basic HA building blocks | Builds on top of HA |

# Continuous Availability

Continuous Availability is not absolute availability.

- Probable DB outages and maintenance events are masked from the app
- App continues with no errors and within specified response time objectives while processing these events

Key points

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

Many customers achieve Continuous Availability today

# .NET Application Continuous Availability

Goal: Make Oracle .NET applications continuously available

- Not just Oracle DB servers

What does that mean?

- Apps handle recoverable errors, such as DB and network failures transparently
- Developers add no to little code to handle failures gracefully

Benefits

- Improved end user service levels
- Greater developer productivity
- Less administrative time spent handling failover

## .NET Application Continuous Availability

Database has numerous availability solutions

- RAC, Data Guard, GoldenGate, Global Data Services, RAC One Node, etc.

What about for the middle-tier for client HA?

- ODP.NET works with each DB availability solution
- Main scenarios: planned maintenance and unplanned outages
- Errors and interruptions automatically masked to end users during outages

## Fast Application Notification

# FAN in a Nut Shell

FAN events specify what changed, where, when

- Server initiated messages
- ODP.NET apps receive these messages
- Adjusts connection behavior automatically

ODP.NET subscribers receive FAN over either

- Oracle Notification Service (ONS): ODP.NET 12c and higher
  - Except unmanaged ODP.NET 12c with Oracle DB 11.2 or earlier
- AQ: ODP.NET 11g and lower

ONS is faster, more scalable, eliminates firewall issue, supports Active Data Guard, and consolidates publish/subscribe service

# FAN

Rapid notification about DB and service state changes

Event types

- Down
  - Received quickly to invoke failover
- Planned Down
  - Drains sessions for planned maintenance with no user interruption
- 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

## Planned Maintenance Outages

## Planned Outage – How Availability Is Maintained

Oracle .NET app uses ODP.NET pooling

FAN alerts ODP.NET of impending downtime or service relocation

- ODP.NET stops allocating new connections to that DB node
- Idle connections closed
- Connections returned to the pool are closed

Pool draining feature

- Part of Fast Connection Failover (FCF)

Shutdown commences when all connections are closed

- Optionally set a maximum timeout upon which shutdown occurs

End user experience: Little to no disruption

## Pool Draining Example and Settings

Relocate service example

- srvctl relocate service –database <database name> –service <service name> –drain_timeout 120 –stopoption IMMEDIATE –oldinst <existing instance>

ODP.NET settings

- "Pooling=true" (default)
- "HA Events = true" – connection pool attribute
  - In ODP.NET 12.2+, set to true by default

Recommend using Oracle Database 11.2.0.4+ and ODP.NET 11.2.0.4+

# Pool Draining for Data Guard Switchovers

Data Guard 12.2+ supports pool draining

- SWITCHOVER TO <database name> [WAIT <timeout in seconds> ];

Switchovers require primary to shut down before standby comes up

- End users may request a connection during that time
- With no available DB to serve the request, users receive timeout errors
- But this can be prevented by “pausing” ODP.NET requests

# Pool Draining for Data Guard Switchovers

ODP.NET ServiceRelocationConnectionTimeout setting blocks incoming connection requests until standby is up or timeout expires

- Eliminates/Reduces connection timeout errors
- Set in .NET config file or OracleConfiguration class
- Default = 90 seconds

Oracle TNSNAMES settings complementary to blocking connection requests

- RETRY_COUNT
  - Number of times ADDRESS list traversed before connection attempt terminates
- RETRY_DELAY
  - Delay in seconds between subsequent retries for a connection

# RETRY_COUNT and RETRY_DELAY

## TNSNAMES.ORA Sample Configuration

```
alias =(DESCRIPTION_LIST =
       (DESCRIPTION=
                (RETRY_COUNT=10)(RETRY_DELAY=5)
               (ADDRESS_LIST=(ADDRESS = .   .   .)(ADDRESS= .  .  .))
      (CONNECT_DATA=(SERVICE_NAME=hr_svc)))                        .
     (DESCRIPTION=
                (RETRY_COUNT=10)(RETRY_DELAY=5)
      (ADDRESS_LIST=(ADDRESS = .   .   .)(ADDRESS=.  .  .))
      (CONNECT_DATA=(SERVICE_NAME=hr_svc2))))
```

Retry while service is unavailable

## Planned Maintenance Outage

For non-pooled scenarios

Use Transparent Application Failover (TAF)

- Upon planned outage, automatically reconnect to an available DB instance
- Restore session state – available in 12.2+
  - i.e. NLS, timezone, action, module, etc.
- Reposition cursors to position they were on prior to failure
- Replay initial transaction statement – available in 12.2+
  - Any DML or stored procedure

TAF is an OCI level feature

- Unmanaged ODP.NET only

## Unplanned Outages

## Use Fast Connection Failover

FAN alerts ODP.NET of immediate DB node, service, etc. failure

- ODP.NET stops allocating new connections to failed entity
- Idle connections closed
- Active connections are closed

Benefits (assuming another DB instance is available)

- New users are not allocated invalid connections
- No need to clear pools explicitly

## Application Continuity (AC)

Recover active queries and transactions seamlessly after unplanned DB outages from a recoverable error

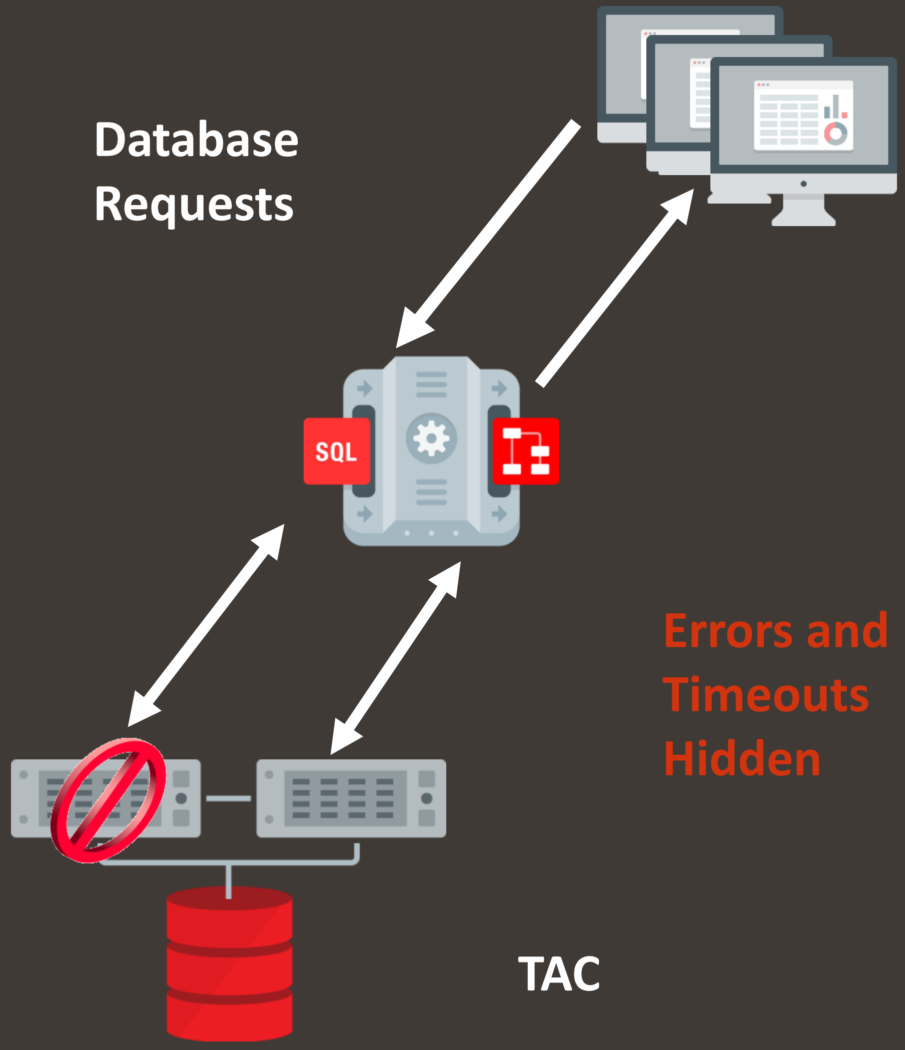
Masks hardware, software, network, storage errors, and timeouts

- Available with RAC, RAC One Node, and Active Data Guard

Benefits

- AC replays in-flight work on recoverable errors
- No additional ODP.NET code required
- Transparent to end users (except for a slight delay)

# Transparent Application Continuity (TAC)



Built in session state tracking

Restores and enforces session state before replay

Discovers and advances request boundaries

Keeps mutables for SQL

Disables side-effects

Adapts as applications change: apps protected for the future

# TAC Explained

## Normal Operation

ODP.NET marks DB requests

Server tracks session state, decides which calls to replay, disables side effects

ODP.NET holds original calls, their inputs, and validation data

🟩 New with TAC

## 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 ODP.NET TAC Features

NEW IN 19c

Each request can contain multiple transactions

- TAC supports all transactions in request, not just the first
- Uses implicit request boundary

Smaller memory usage

- Implicit request boundary ensures more timely replay queue purges

New features require DB 18c+ with unmanaged ODP.NET 19c+

# ODP.NET AC Settings

Set “Pooling=true” (default)

“Application Continuity=true” (default) in the connection string

- Available in ODP.NET 12.2+

AC requires unmanaged ODP.NET and DB 12.2+

TAC requires unmanaged ODP.NET and DB 18c+

## Transaction Guard

ODP.NET determines transaction commit status even upon a DB failure without custom coding

Used by AC

Benefit

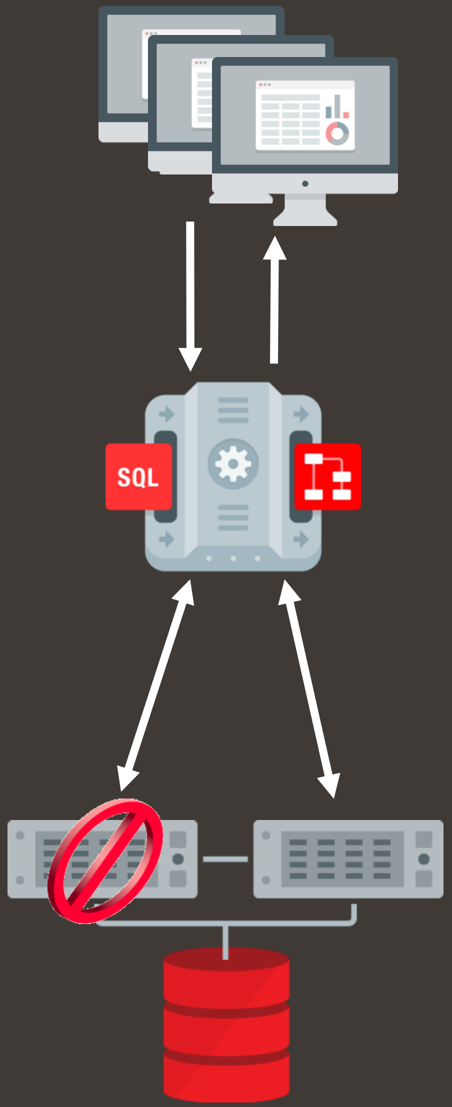
- Ensures accurate knowledge of transaction outcome

App can query transaction outcome

- OracleConnection properties return transaction status
- OracleLogicalTransaction class

Requires Oracle Database 12c and ODP.NET 12c+

# Transaction Guard Scenario



1. ODP.NET receives FAN down event or error

2. IsRecoverable=false → roll back
IsRecoverable=true → re-submit

3. To re-submit, retrieve OracleConnection.OracleLogicalTransaction

4. Retrieve transaction status with OracleLogicalTransaction.GetOutcome

5. If committed and completed, done.
If not committed nor completed, re-submit.

# Transaction Guard

## ODP.NET Catch Block Code Sample

```
catch (Exception ex)
{
      if (ex is OracleException)
      {
            //  It's safe to re-submit the work if the error is
            // recoverable and the transaction has not been committed

            if (ex.IsRecoverable && ex.OracleLogicalTransaction != null &&
!ex.OracleLogicalTransaction.Committed)
            {
                // safe to re-submit work
            }
            else
            {
                // do not re-submit work
            }
       }
}
```

## Questions?

## Hands on Labs

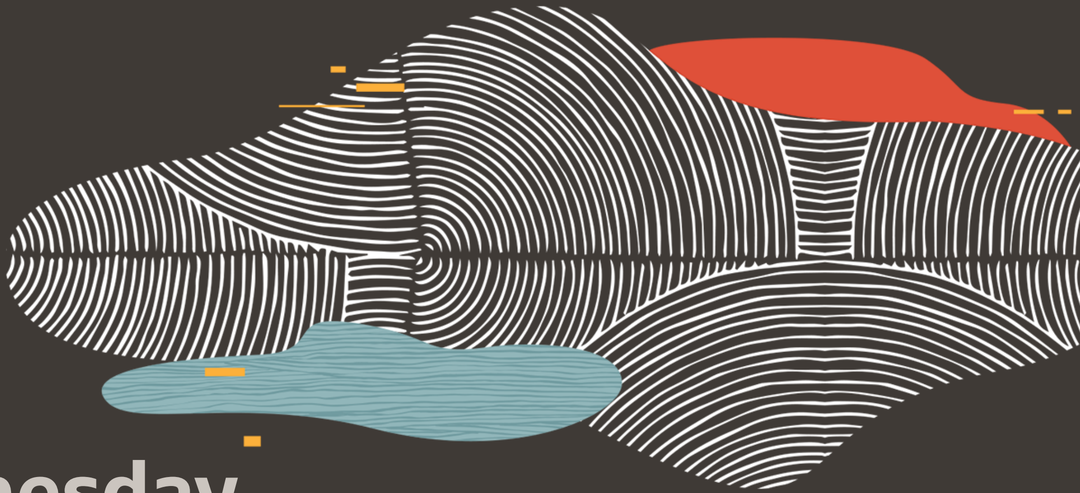
**HOL4642: Building .NET Applications with Oracle Database**

## Tuesday

**2:15-3:15** Moscone West 3019

## Wednesday

**9:00am-10:00** Moscone West 3019



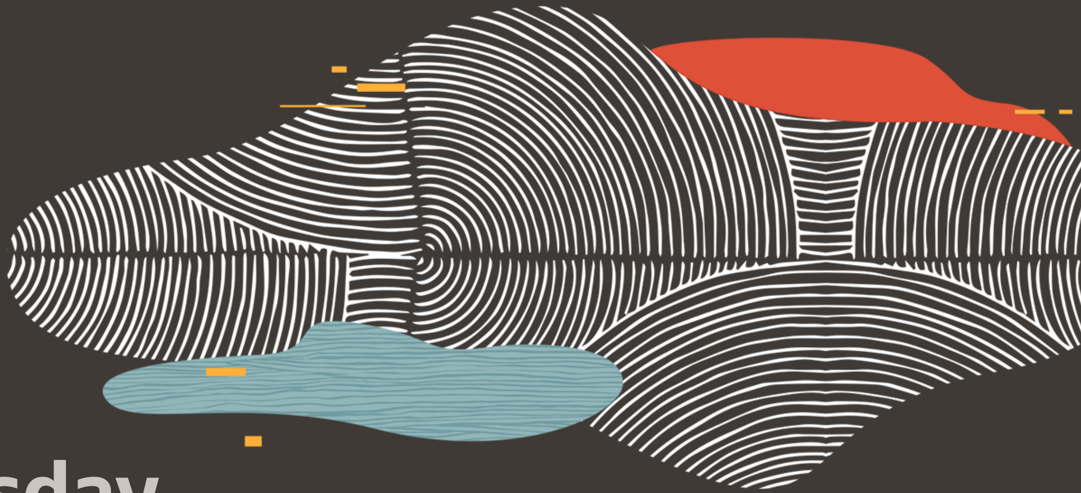
**What's Ahead**

## Wednesday

| | |
|---|---|
| 5:00-5:45 | Developing and Deploying Oracle Database Applications in Kubernetes<br><br>Moscone South 313 |

## Thursday

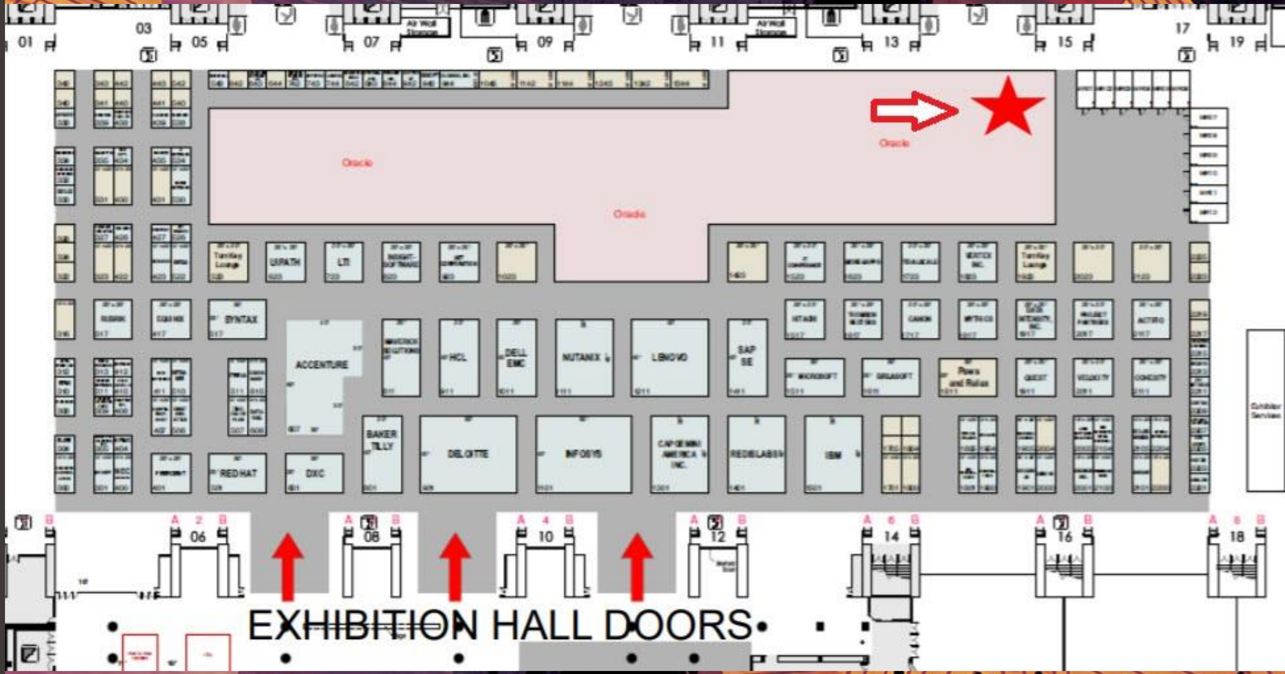
| | |
|---|---|
| 9:00AM-9:45 | Accelerate Oracle Database .NET Application Performance - Moscone South 313 |
| 1:15 – 2:00 | Exploring the Multicloud: Working with Azure and Oracle Autonomous Database - Moscone South 209 |
| 2:15-3:00 | Running Oracle Database and Applications in Docker Containers on Windows - Moscone South 313 |



**Get a personalized demo**

## Visit us in the Moscone South demogrounds

DEV-013: .NET Development and Windows Integration for Oracle Database



## Thank You

**Alex Keh**

Senior Principal Product Manager
Oracle
September 17, 2019

**Session Survey**

Help us make the content even better. Please complete the session survey in the Mobile App.

## Safe Harbor