
ORACLE®

NCOUG 2018
Transparent High Availability for Your Applications

Carol Colrain
Real Application Clusters Development
February, 2018

ORACLE

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. Easy steps for Application Continuity
3. Are your building blocks in place?
4. Customer Stories

How do we define Continuous Availability?

Customer have differing definitions

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:

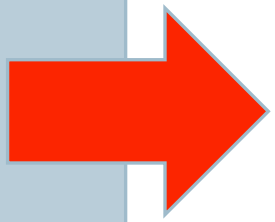
1. Planned maintenance and likely unplanned outages are hidden from applications
2. There is neither data loss nor data inconsistency
3. Majority of work (% varies by customer) completes within recovery time SLA
4. May appear as a slightly delayed execution

Many customers are achieving Continuous Availability Today

Difference between High Availability and Continuous Availability

High Availability

- Minimizes downtime
- Guaranteed data commitment
- In-flight work is lost
- Rolling maintenance at DB
- Predictable runtime performance
- Errors may be visible
- Design for single failure
- Basic HA building blocks



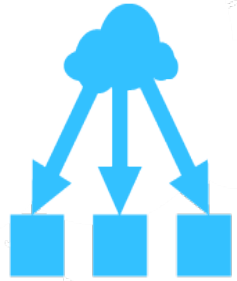
Continuous Availability

- Removes downtime from user perspective
- Runtime data corruption protection
- In-flight work is preserved
- Maintenance is hidden
- Predictable performance during outages
- Errors only if unrecoverable
- Designed for multiple concurrent failures
- Builds on top of HA

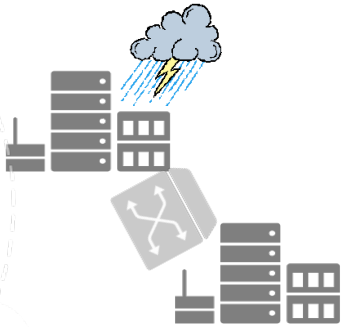
What kind of outages?

Planned
Maintenance
Patches
Repairs
Upgrades
Changes

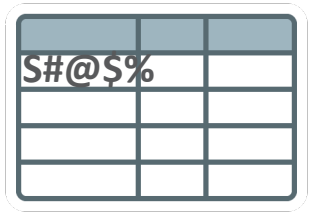
Unplanned Outages



Unpredictable
Response &
Throughput



Site
Disasters



Data
Corruption

Human
Errors

Which outage classes does your business need to handle?

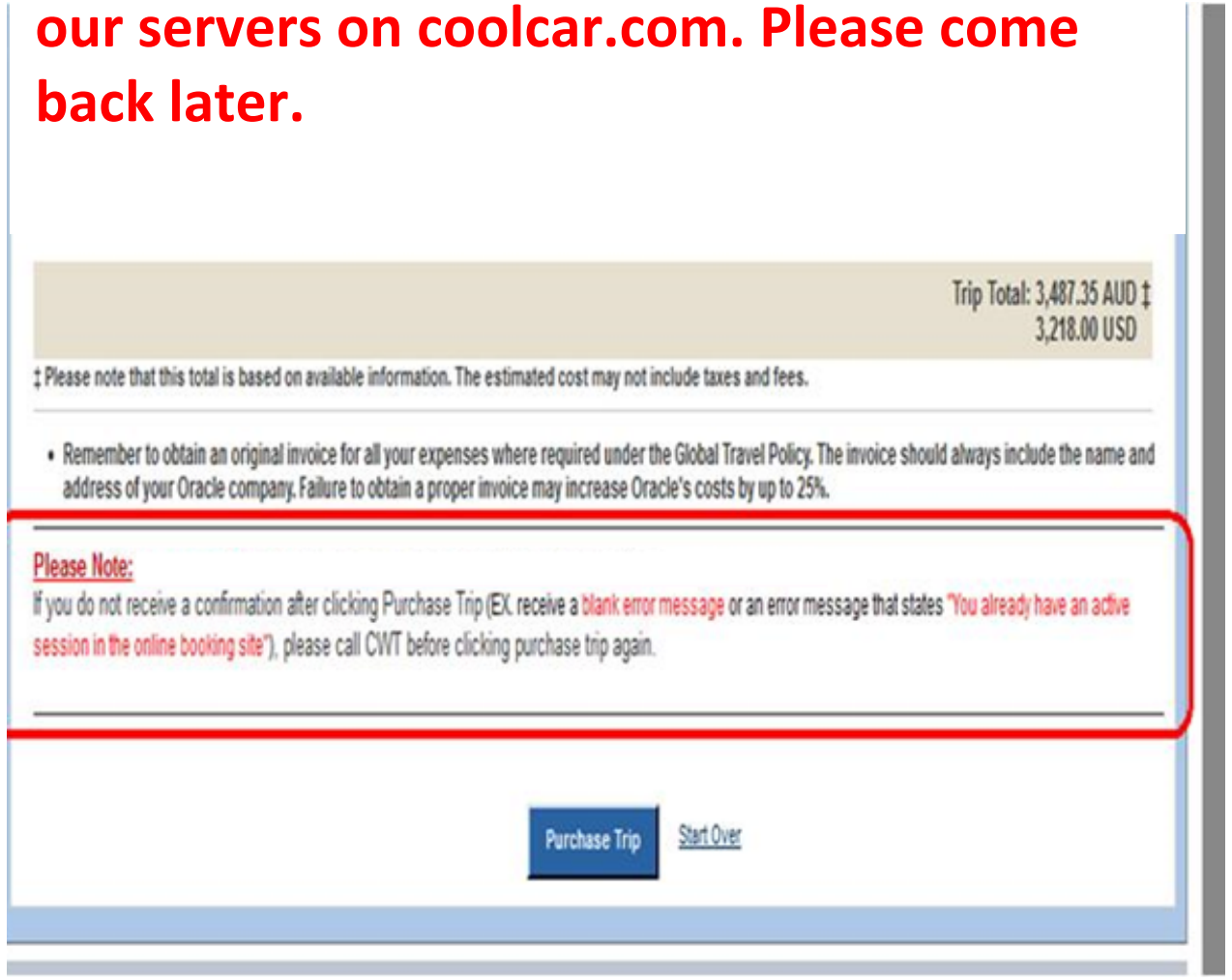
Pre-12^{c} Situation

In-Flight Work Lost

Database outages cause loss of in-flight work, leaving users and applications in-doubt

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

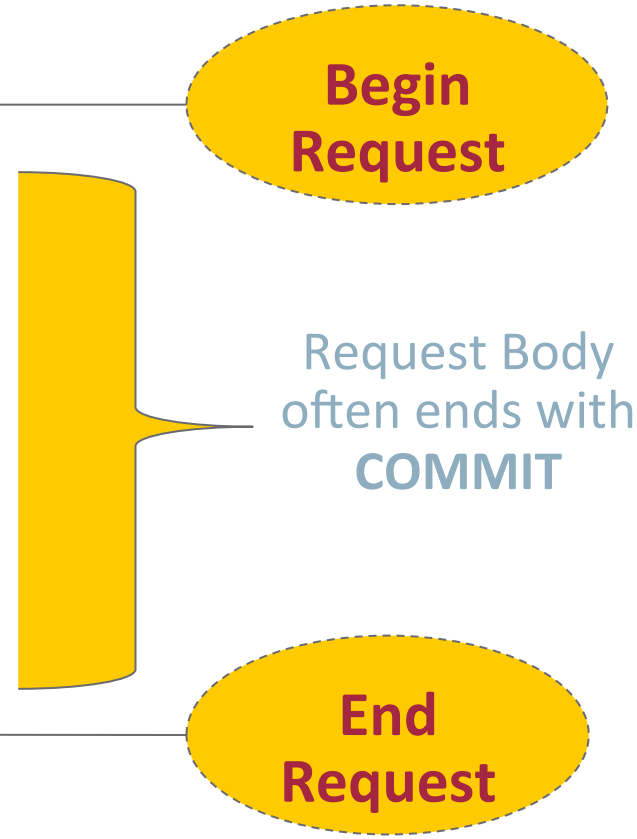
Sorry. Internal Server Error - 400 Error
We are currently experiencing an issue with our servers on coolcar.com. Please come back later.



Database Request - Standard JDK9

New Concept

```
PoolDataSource pds = GetPoolDataSource();
Connection conn = pds.getConnection();
PreparedStatement pstmt = …
   …
    SQL, PL/SQL, local calls, RPC
     …
conn.commit();
conn.close();
```



3

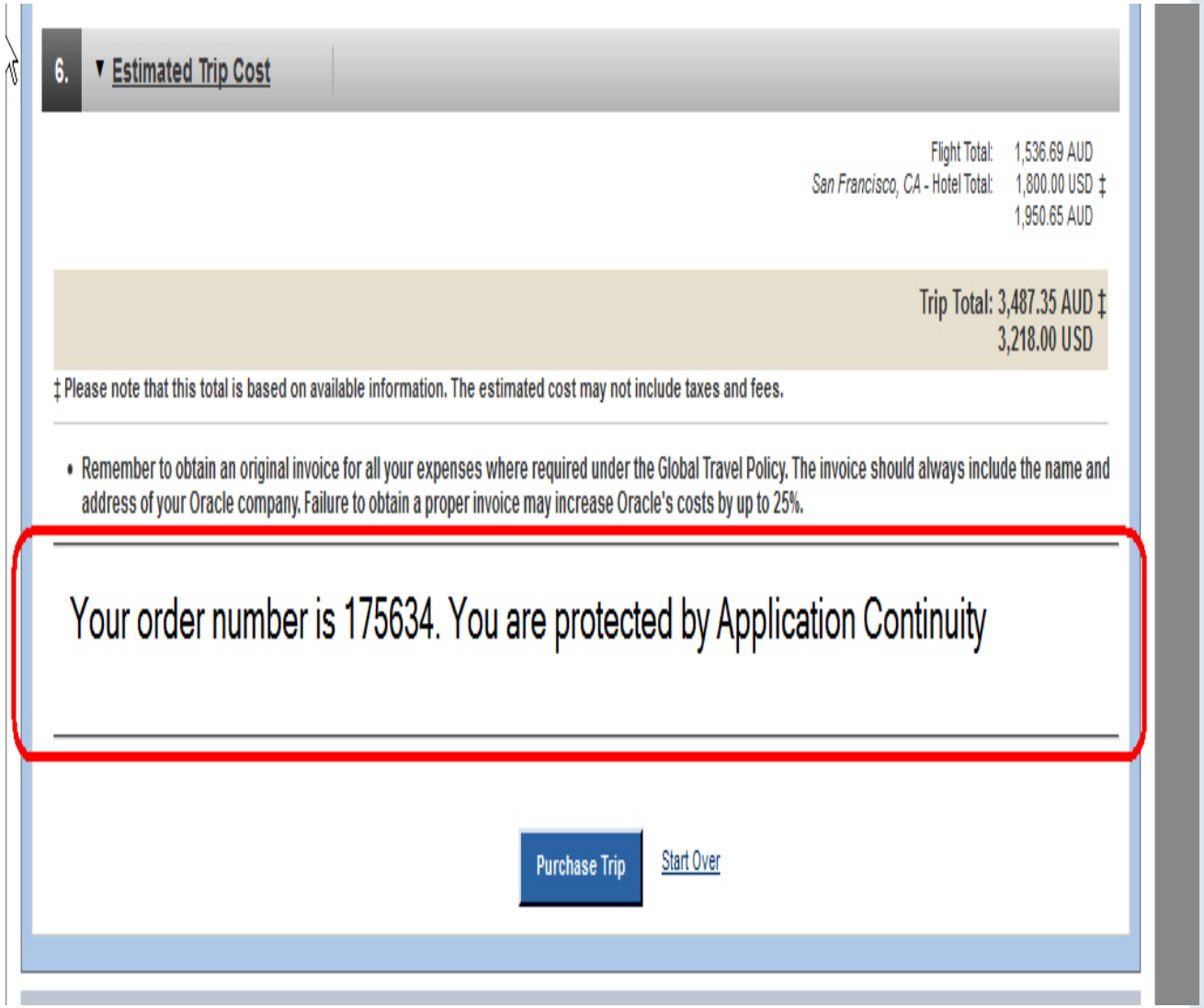
Easy Steps for Application Continuity

Unplanned outages should be hidden from applications

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)**



Transparent Application Continuity

– Make TAC standard for your applications

NEW IN 18c

Under the Hood

Normal Operation

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

New in 18c

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 function values
- Ensures results, client states, messages 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 pre-18c

Remainder of Request

- Alter system, database, session (subset)
- Best effort streaming lobs
- XA after promote
- OCI – old OCIStmtPrepare, ADT's, AQ
- DB links ADG to primary

Target Database

- Different Database
  - Logical Standby
  - Golden Gate
  - 3rd Party Replication

Easy Steps for Application Continuity

Check	What to do
Identify Requests	Use Oracle pools and return the connections, or use TAC for Java
Initial State	Use **FAILOVER_RESTORE** for most applications
Grant Keep	Grant **KEEP**ing mutable values, e.g. seq.nextval, sysdate, systimestamp
Side effects	TAC disables all side effects, AC custom
Check Protection	Protection levels in AWR, V$SYSYSTAT, V$SESSSTAT, detail in acchk
No deprecated Java	Use **acchk** to know (almost nil in 18c)

Request Boundaries, Standard in JDK9

Let the database know that it has a request

Oracle

Return connections

- UCP
- WebLogic Server
- ODP.NET unmanaged
- OCI Session Pool
- Tuxedo
- SQL*Plus

Other Java

Use UCP

Return connections

- IBM WebSphere
- Apache Tomcat
- NEC WebOTX
- RedHat Jboss
- Spring
- your own

Custom Java

- Add Request boundaries
  - beginRequest
  - endRequest

TAC - Request Boundaries Advance Continuously

NEW IN 18c

- Request boundaries advance when state is detected restorable
- Capture is re-enabled, if previously disabled
- Smaller capture set means faster recovery
- Return to pool is still best practice
- Needs first boundary in 18c

Transparent Application Continuity for Java 18c

Initial Session State Before Replaying

- **Failover_Restore on your service (from 12.2)**
  - Restores common session states
- **Logon triggers, Labels, TAF or JDBC callbacks**
  - Allows restoring complex states before replaying
  - Applied after FAILOVER_RESTORE
- **TAC** enforces session matches original before replaying

NEW IN 18c

Side Effects Not Replayed

NEW IN 18c

TAC – disables automatically until next enable point

- **TAC decides if any requests should not be replayed**

  e.g. DBMS_LOGMNR
  DBMS_SCHEDULER
  UTL_HTTP
  UTL_URL
  DBMS_FILE
  DBMS_FILE_TRANSFER
  UTL_SMTP
  UTL_TCP
  UTL_MAIL
  EXTPROC

- **To decide yourself, use AC**

ORACLE

Grant Mutables

Keep original function results at replay

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]

Function Results Restored at Replay

Example Usage

Function	eBusiness Suite	PeopleSoft	Fusion Applications	Siebel
SYSDATE SYSTIMESTAMP	Original	Original	Original	Current
Sequence.nextval Sequence.currval	Original	Original	Original	N/A
SYS_GUID	Original	N/A	Original	N/A
LOB locators (always)	Read consistency	N/A	Read consistency	N/A

When granted, replay restores original function result.

Always Know Your Protection Level

NEW IN 18c

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

Statistic	Total	per Second	per Trans
cumulative requests	177,406	49.2	5.0
cumulative user calls in request	493,329	136.8	13.8
cumulative user calls protected	493,329	136.8	13.8

EPSILON

Detailed Protection Report when Needed

Application Continuity Summary

Outage Type	Status	Message
Coverage checks		**TotalRequest = 398** **PASS = 389** **WARNING = 0** **FAIL = 9**
	PASS	[PASS] Trace file name = WEB_ora_124333.trc Row number = 10909 SERVICE NAME = (WEB_SSL_SERVICE) MODULE NAME = (JDBC Thin Client) ACTION NAME = POST CLIENT ID = null Coverage(%) = 100 ProtectedCalls = 1 UnProtectedCalls = 0
	PASS	[PASS] Trace file name = WEB_ora_19757.trc Row number = 36978 SERVICE NAME = (WEB_SSL_SERVICE) MODULE NAME = (JDBC Thin Client) ACTION NAME = CARD CLIENT ID = null Coverage(%) = 100 ProtectedCalls = 24 UnProtectedCalls = 0
	FAIL	[FAIL] Trace file name = WEB_ora_19757.trc Row number = 481193 SERVICE NAME = (WEB_SSL_SERVICE) MODULE NAME = (JDBC Thin Client) ACTION NAME = null CLIENT ID = null Coverage(%) = 20 ProtectedCalls = 1 UnProtectedCalls = 4
	FAIL	[FAIL] Trace file name = WEB_ora_19757.trc Row number = 14203 SERVICE NAME = (WEB_SSL_SERVICE) MODULE NAME = (JDBC Thin Client) ACTION NAME = null CLIENT ID = null Coverage(%) = 33 ProtectedCalls = 2 UnProtectedCalls = 1

Configuration at Database

Service Attributes same as Application Continuity

- FAILOVER_TYPE = AUTO or TRANSACTION for Application Continuity
- FAILOVER_RESTORE = AUTO or LEVEL1 for initial state
- COMMIT_OUTCOME = TRUE for Transaction Guard

Also verify:

  - AQ_HA_NOTIFICATIONS=True for FAN OCI
  - REPLAY_INITIATION_TIMEOUT = 300 - time in seconds before replay is canceled
  - FAILOVER_RETRIES = 30 for the number of connection retries per replay
  - FAILOVER_DELAY = 3 for delay in seconds between connection retries

- Grant execute on DBMS_APP_CONT

Configuration at Client-Side

Java

```
Use a replay data source (local or XA)

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

OCI, ODP.NET unmanaged, Tuxedo

```
On when enabled on the service
```

SQL*Plus

```
-ac switch (also -dynamic|-static)
```

4 Are your building blocks in place ?

Configuring for Best Results

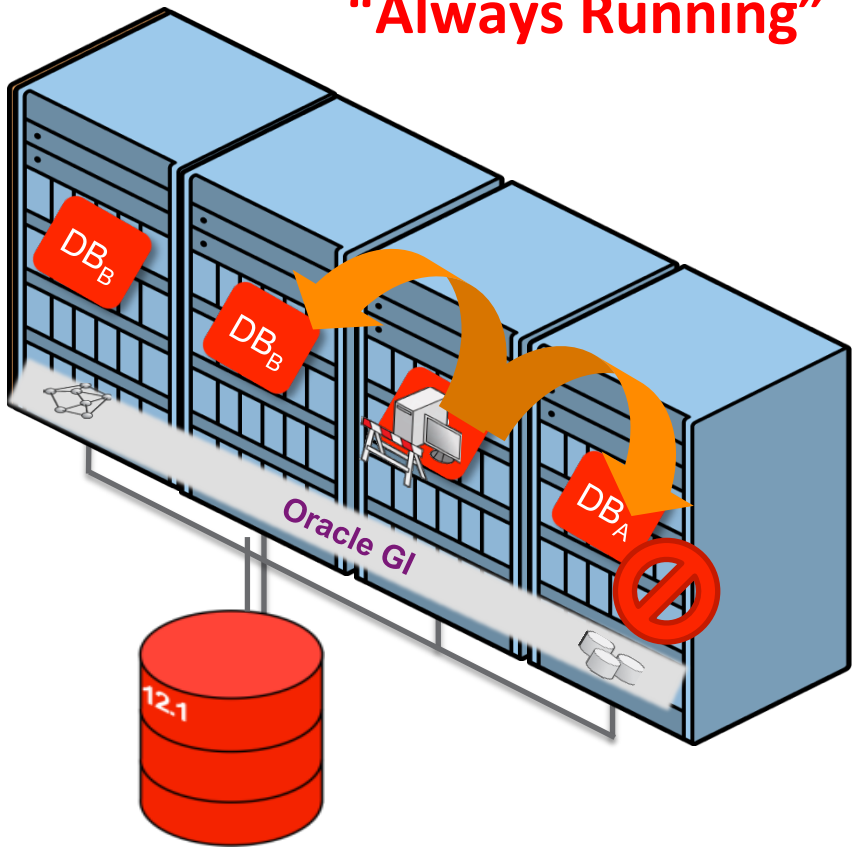
Which Server Stack for me?	• RAC or RAC One, Active Data Guard, GoldenGate, CMAN
Flex ASM	• All databases on Flex ASM or ACFS
Services	• Services for Location Transparency
Continuous Connections	• Connections appear continuous
Outage Notification	• FAN for immediate interrupt
Inflight work	• Application Continuity
Recovery Time SLAs	• Recover in a timely manner

Start Now

Oracle Real Application Clusters

Continuous Availability

"Always Running"



RAC or RAC One Node

Real Application Service Levels

- **Scales** 4000 PDBs 8000 Services NEW IN 18c
- Online Rolling Maintenance
- Drains and balances **gradually**
- DB Recovers in very low seconds
- **App Continuity** recovers inflight
- **Data is always consistent**

Oracle Active Data Guard

Continuous Availability

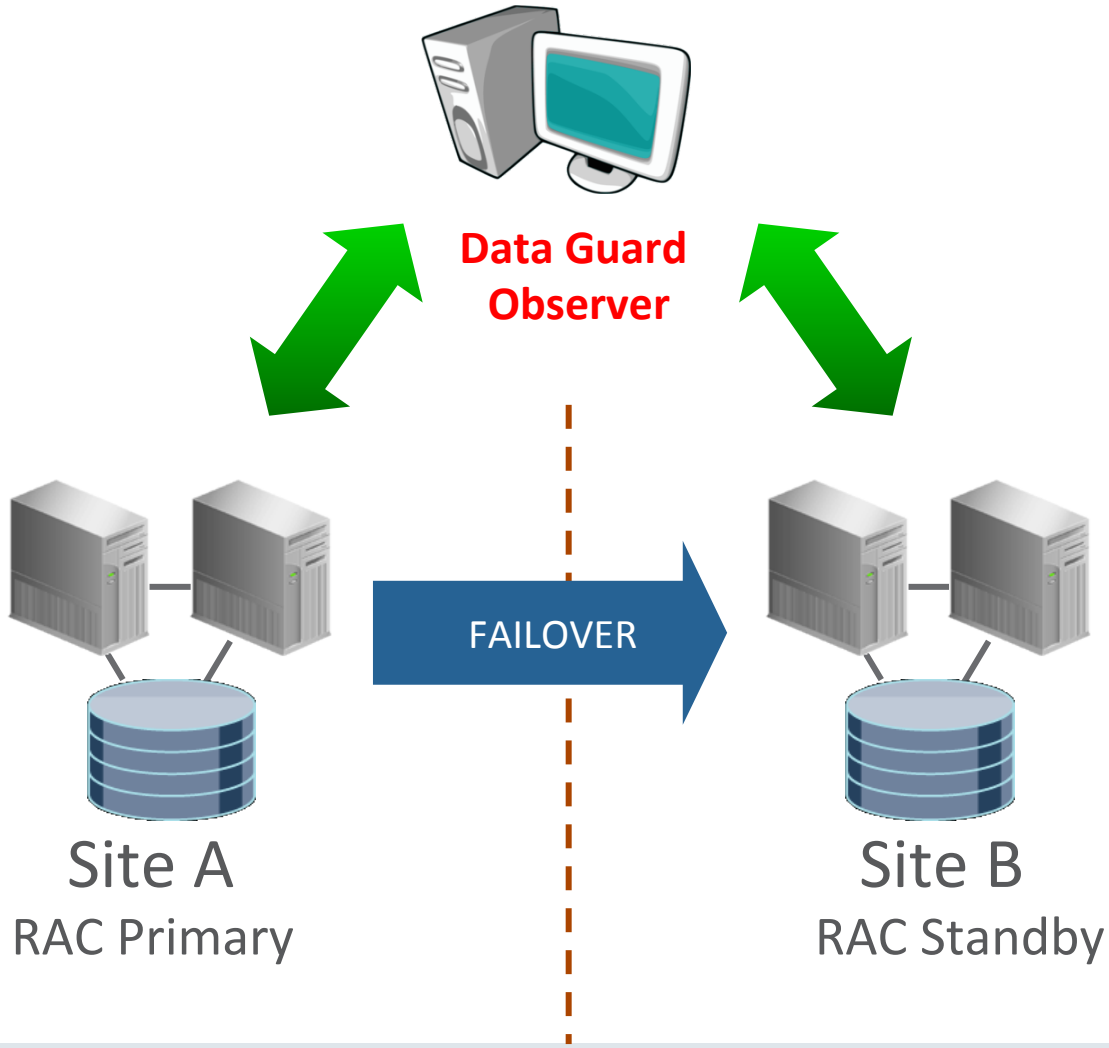
Use max availability/(no)affirm for local DG-ADG, FarSync to remote sites for **no data loss**

Recover in-flight work with Application Continuity

For planned, drain then switchover

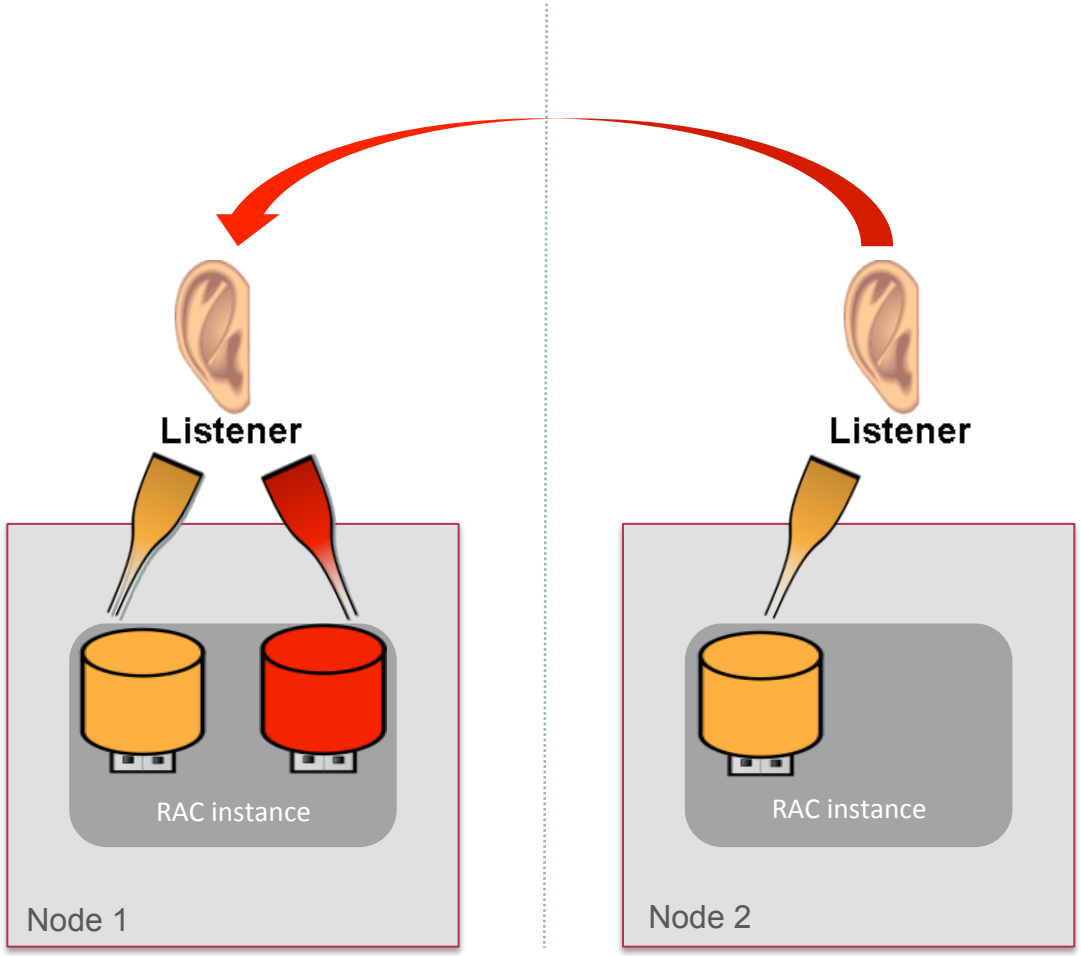
```
Switchover to <DG> [wait]
```

ADG sessions survive role change



Services for Location Transparency

Services provide a “dial in number” for your application



- Regardless of location, application keeps the name
- Moving, reshaping, prioritizing controls how a service is offered
- Batch and OLTP separated
- DB and PDB names for admin only

Connections Appear Continuous

while a service is temporarily unavailable

Configure in One Place

Automatic Retries

```
alias =(DESCRIPTION =
    (CONNECT_TIMEOUT=90) (RETRY_COUNT=20)(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)))
```

No reliance on DNS

ALWAYS use a SERVICE that is NOT DB/PDB name

FAN for INSTANT Interrupt

The dead thing cannot tell you it is dead

Standardized

JDBC Thin Driver
JDBC Universal Connection Pool
OCI/OCCI driver
ODP.NET Unmanaged Provider (OCI)
ODP.NET Managed Provider (C#)
OCI Session Pool
WebLogic Active GridLink
Tuxedo
Listeners

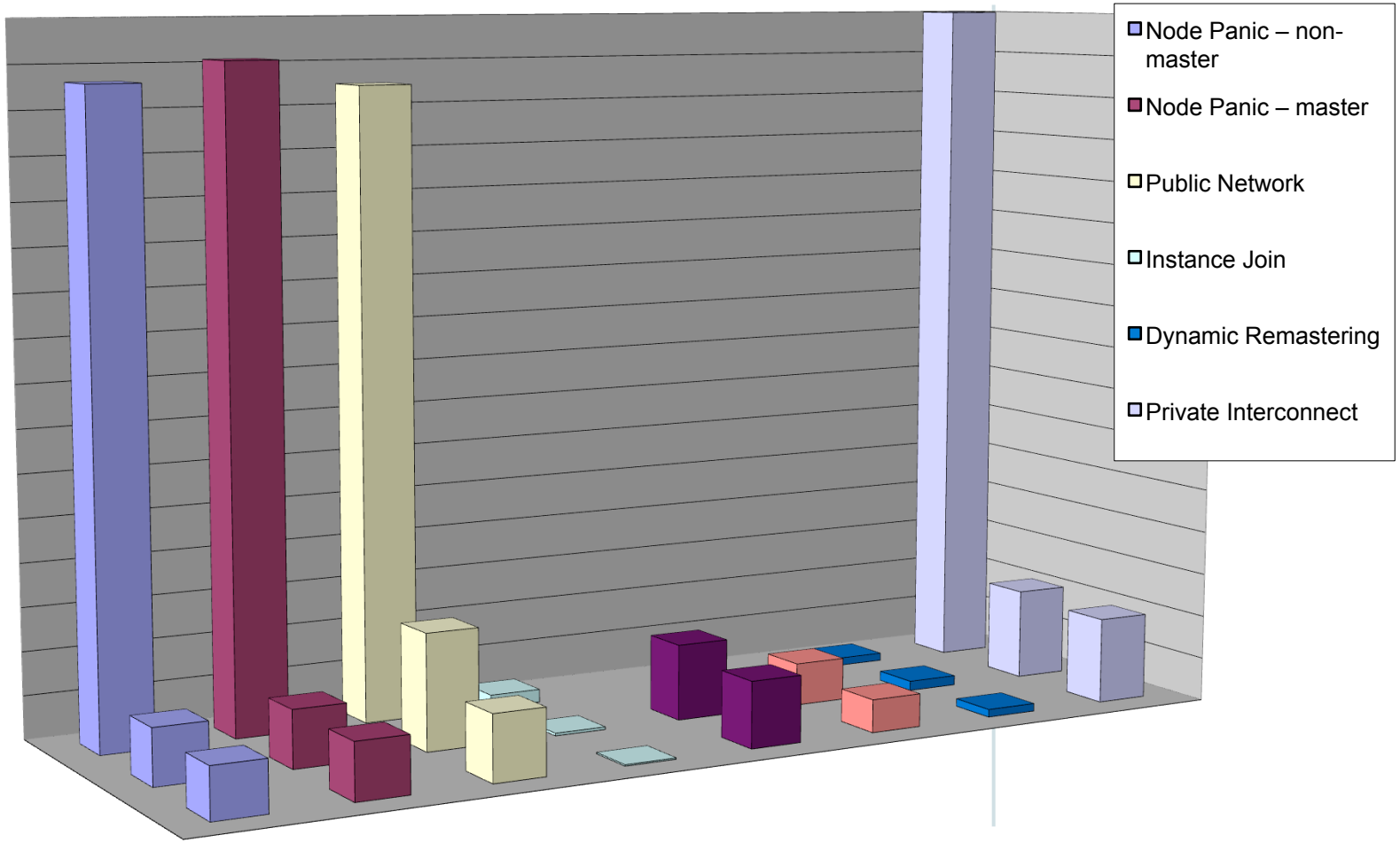
Auto-Configured

```
DESCRIPTION =
    (CONNECT_TIMEOUT=90)
    (RETRY_COUNT=20)(RETRY_DELAY=3)
    (TRANSPORT_CONNECT_TIMEOUT=3)
    (ADDRESS_LIST =
        (LOAD_BALANCE=on)
        ( ADDRESS
        (HOST=prima
    (ADDRESS_LIST =
        (LOAD_BALANCE=on)
        ( ADDRES
        (HOST=seco
(CONNECT_DATA=(SERVICE_NAME=gold)))
```

FAN auto configured

FAN auto configured

Recover in a timely manner



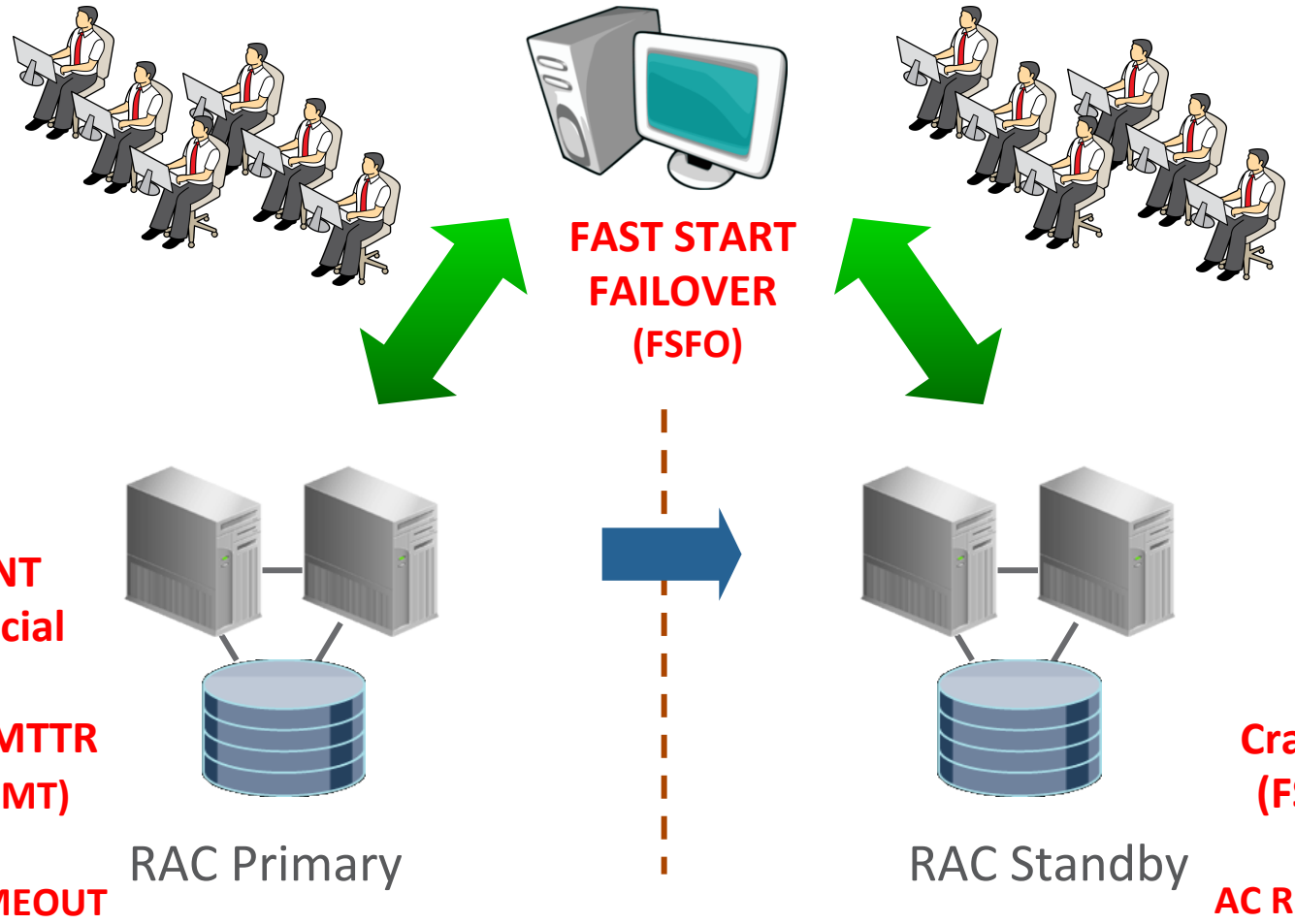
TUNABLES

- Hardware & adjustable misscount
- FAST_START _MTTR_TARGET
- Flex ASM
- SCSI Timeout
- Service Isolation 18^{C}
- PING_TIMEOUT
- FSFO Observer

Align your Timeouts

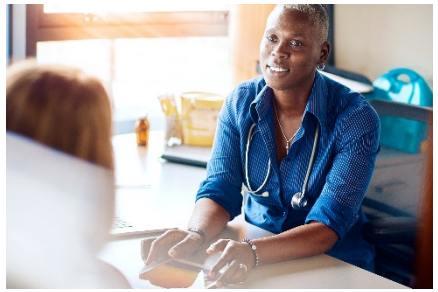
Application Timeout

- **FSFO + Crash Recovery**
- **2 x MISSCOUNT + FSMT**

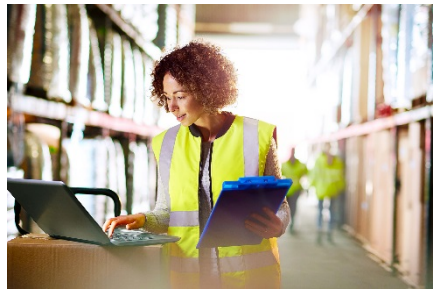


Customer Stories







History of NEC/Oracle alliance

The NEC and Oracle alliance is continuous over a quarter of a century

Won Global Partner Award: Database 2015,2016

2016 **Continuous Availability**

2015 **Zero error solution enhancement for cloud**

2013 **Zero error planned DB maintenance and unplanned DB outage solution**

2012 NEC high available Linux DB platform

2008 NEC's RAC 11gR1 fast failover best practice

2006 NEC's RAC 10gR2 fast failover best practice

2005 STA (Strategic Technology Alliance) started

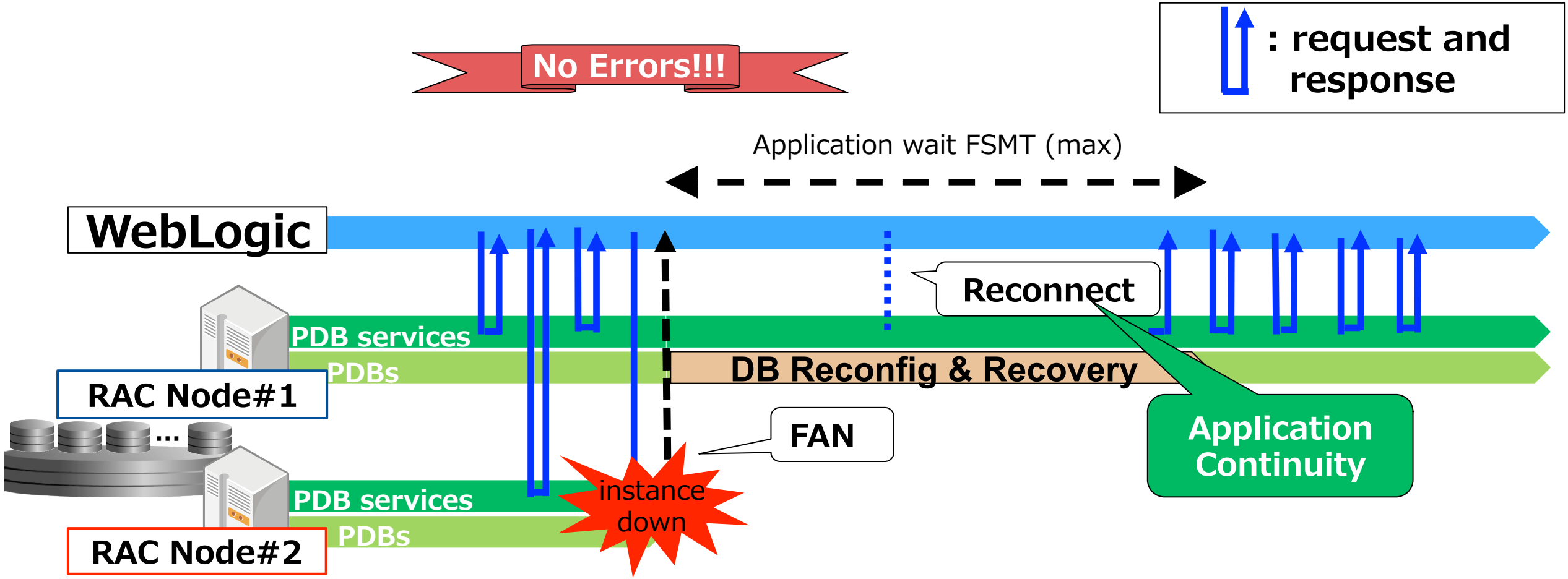
2000 Development alliance for mission critical systems

1997 NEC and BEA alliance started

1987 NEC and Oracle OEM contract started(first in Japan)

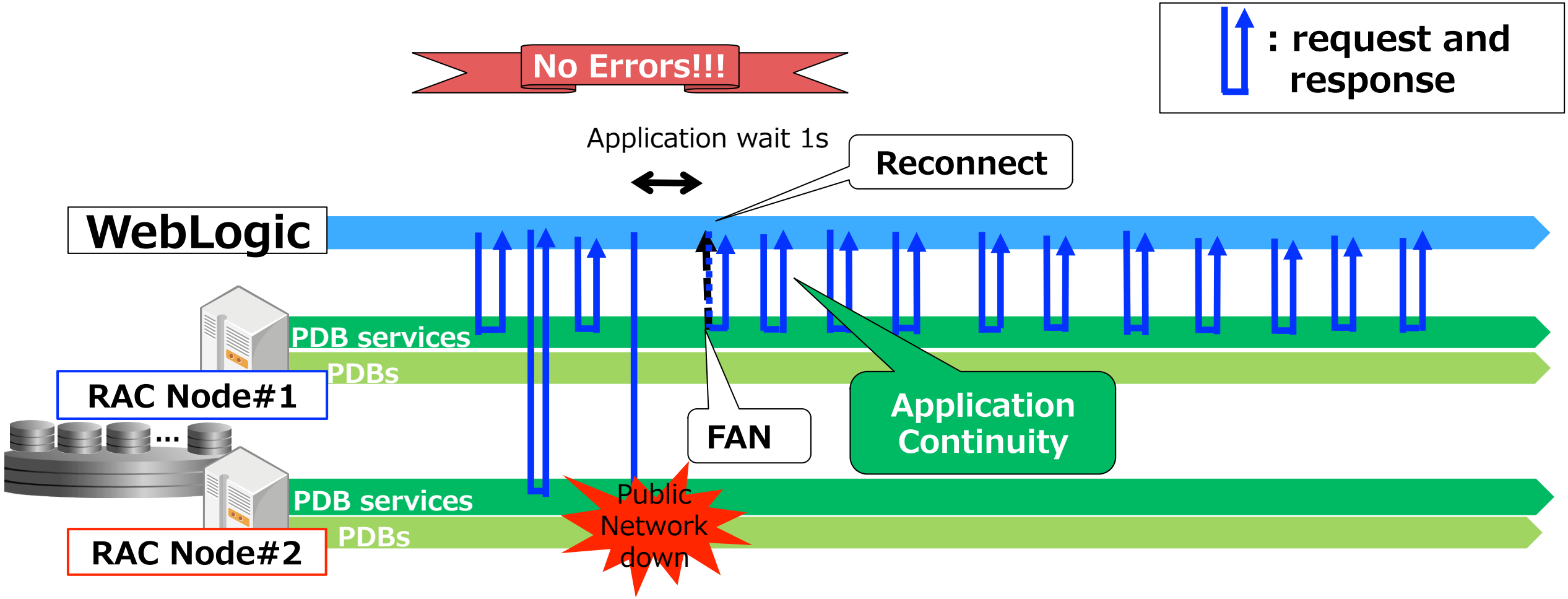
DB Instance Down

Confirmed no errors to all tenant's clients



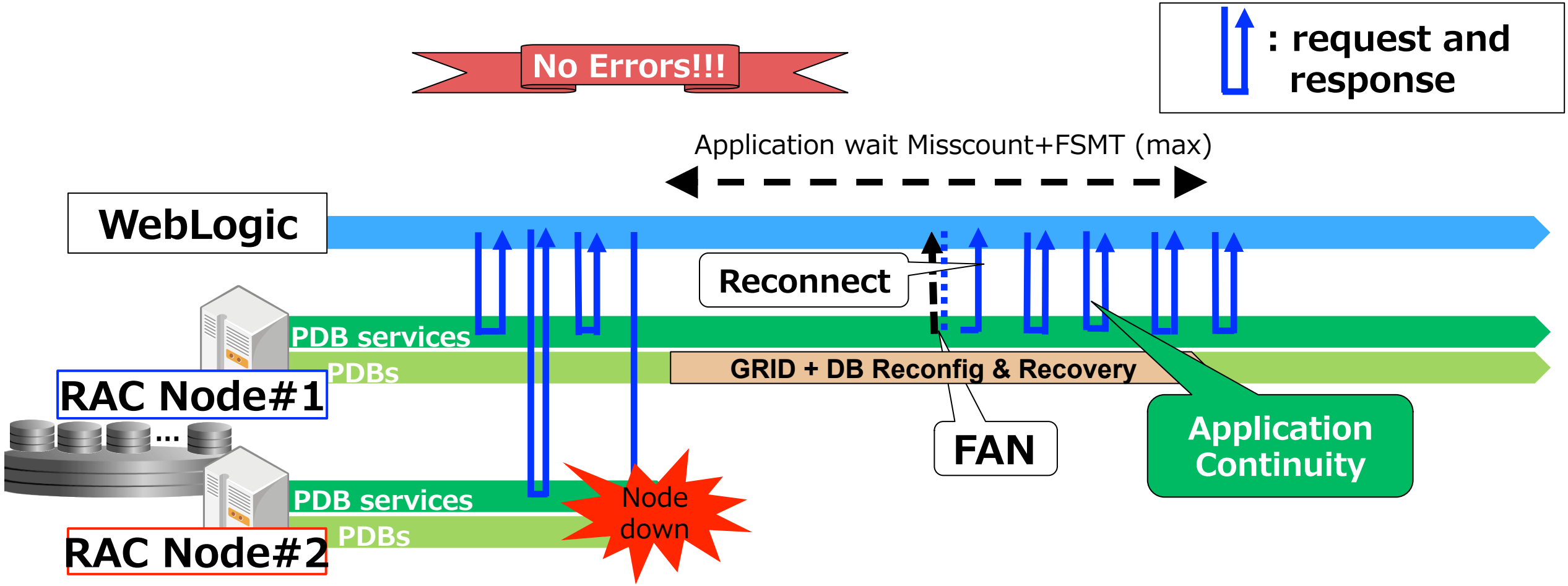
Public Network Down

Confirmed no errors to all tenant's clients



DB Node Down

Confirmed no errors to all tenant's clients



NEC is ready to provide Highly Available and Maintainable Cloud platform with Application Continuity, Real Application Clusters, Oracle Multitenant, Oracle Security options and WebLogic Server Active GridLink on NEC HW.
I believe this platform brings phenomenal value to our cloud customers because it hides unplanned outages and scheduled maintenance from application users.

Yuki Moriyama
Deputy General 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.


ORACLE®