CME Group

Providing Continuous Application Availability using Oracle Application Continuity

Troy Reece – Senior Director in Database Technologies
Cristian Berrios – Lead DBA
Shiraz Kamal – Senior DBA

October, 2017

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

We work around the world, around the clock.

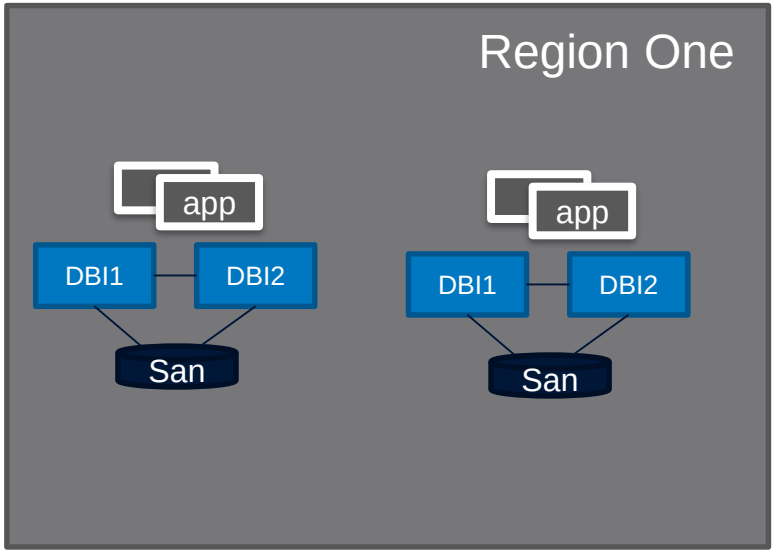
Agenda

- History of Database High Availability At CME
  - Escalating Requirements
  - Maturing Database Architecture
  - Challenges
- Continuous Application Availability
  - Why we're doing it
  - Unplanned outage
  - Planned outage
  - Technical
- Reducing Brownouts
- Test Results

HA First Iteration - Electronic Trading Grows

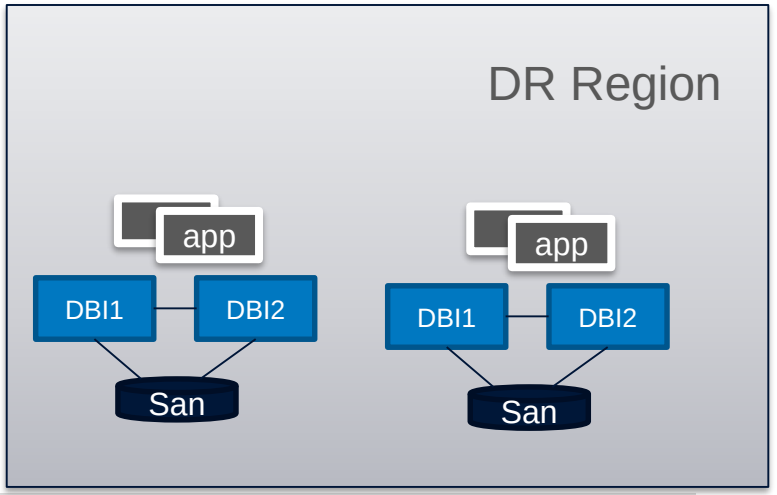
Requirements

- Harden DR environment
- Databases must be available
- Database must be scalable
- RPO/RTO – not well defined
- Application SLA not well defined
- Critical Apps on Main Frame
- Abundant downtime for maintenance



Solution

- Early Adopter of Linux –Replaces Sun Big Box
- Early Adopter of RAC
- Oracle RAC Various Sizes
- Replication VIA Various SAN Technology



Challenges

- Application restarts on db instance failures
- Database failover takes too long
- Transactions not scalable – DB proliferation

HA Continued – Distributed Computing

Requirements

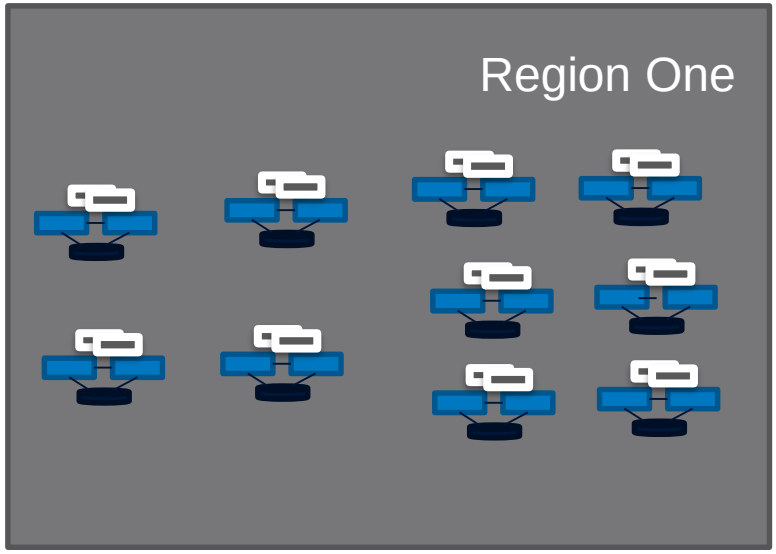
- Critical DB's – 10 second outage to SLA
- RPO – 30 seconds (Disaster Only)
- RTO – 4 hours (Disaster Only)
- Distributed Computing Over Mainframe
- Abundant downtime for maintenance

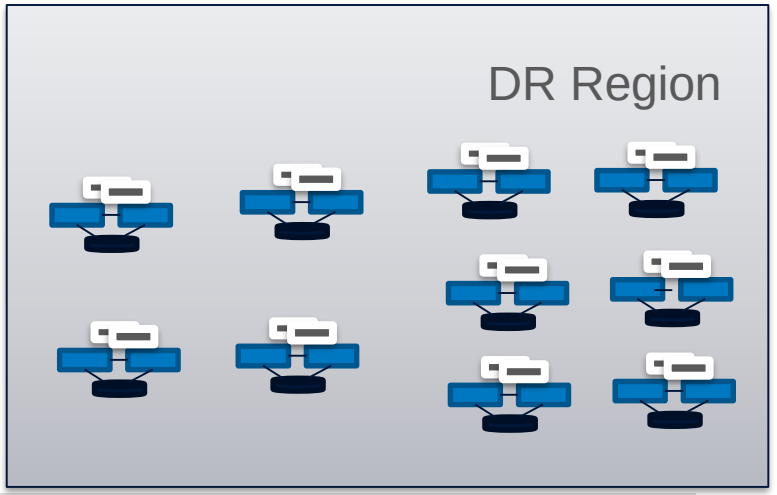
Solution

- Zero Brownout – Reduce node failure time
- ONS/FAN/FCF introduced
- RAC Compliance
- San Replication for DR

Challenges

- Still painful to code and test applications – Technical Debt
- Transactions loss dealt with at app level
- Distributed Computing grows – DB proliferation
- Datasets Grow causing batch processing to exceed SLA

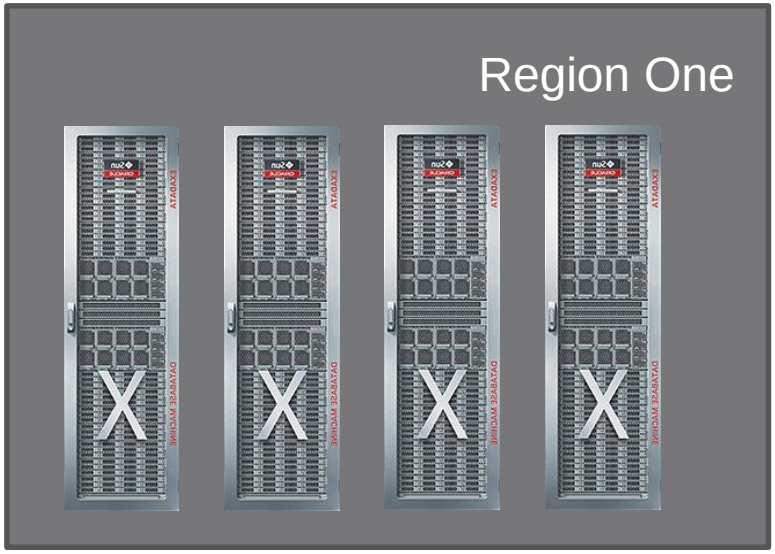




HA Current - Electronic Trading Is Mature

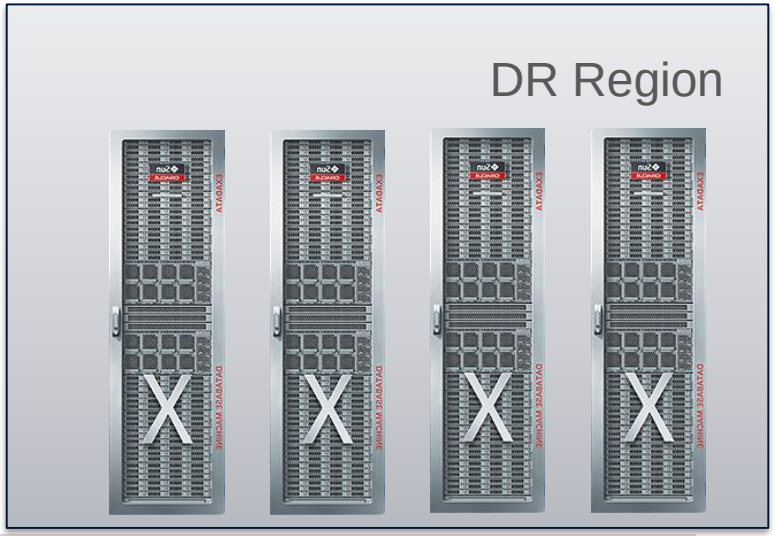
Requirements

- RTO – <10 sec (OLTP)
- RPO – 30 seconds (Disaster Only)
- RTO – 2 hours (Disaster Only)
- Shrinking maintenance windows
- Component Failure Cannot cause DR Event
- 24X7 Planned Maintenance Capability



Solution

- Exadata – Addresses Performance Allows Consolidation
- Data Guard Replication – Active too
- Application Continuity – Planned/unplanned

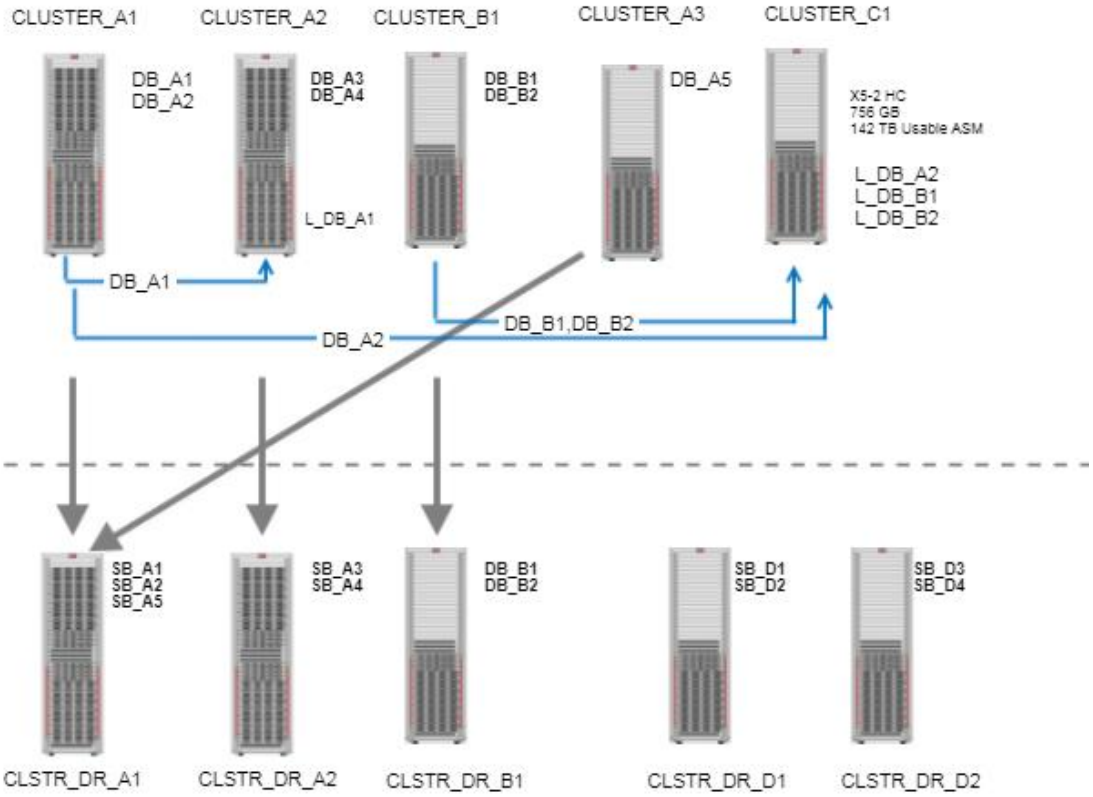


Challenges

- Significant Investment
- Slow Adoption - Priority

CME Exadata HA Architecture

- Multiple Databases / Exadata
- Each Prod Database is replicated locally and remotely
  - BLUE Local (Fast Sync)
  - Gray – Async
- Dedicated Local DG Recipient
- Active DG in DR
- Multiple Complete Exadata Failures need to occur in order for DR event to happen
- Running over 100 apps and more that 200 services



Application Continuity

Why Bother

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 serval hours to start and normalize

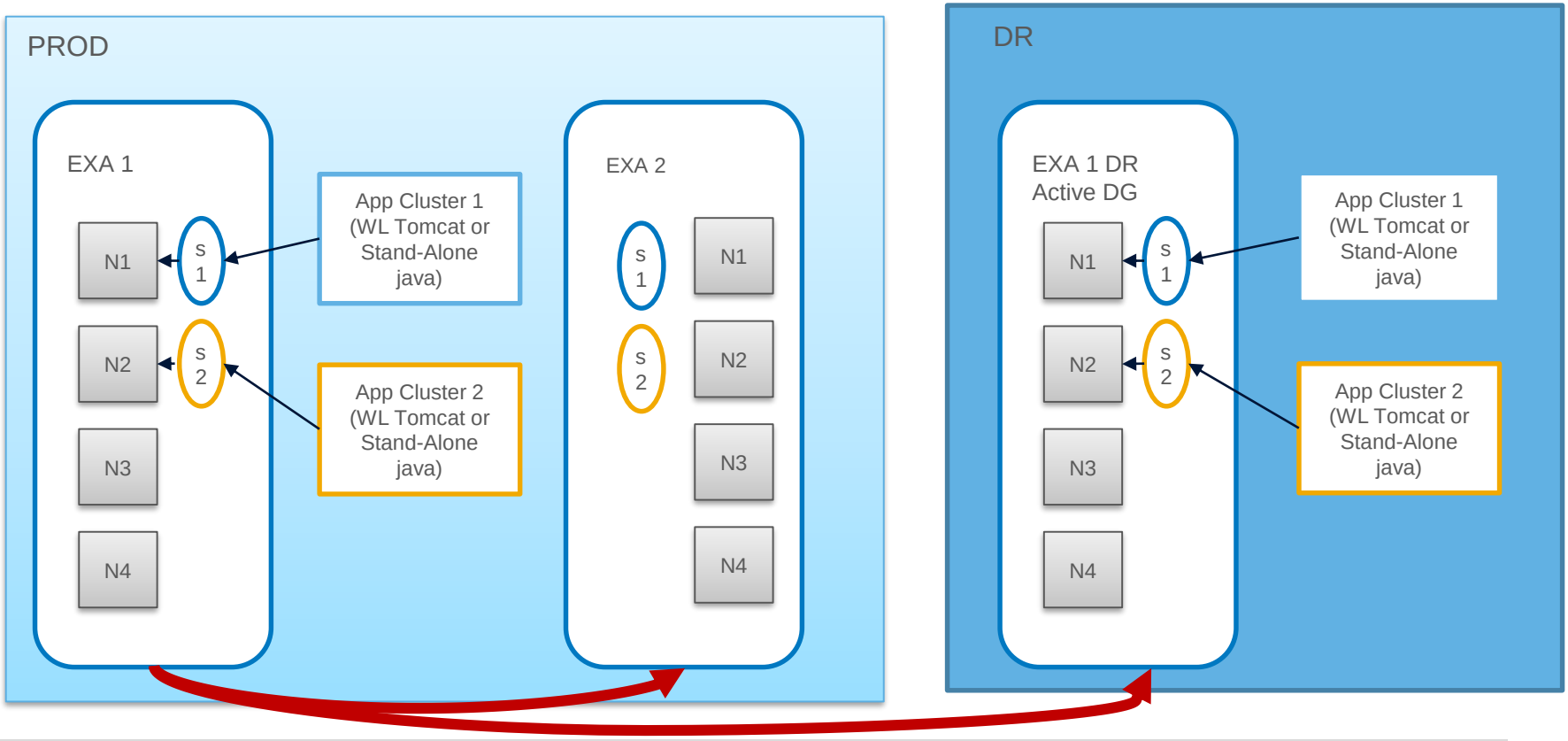
Failure Proofing Applications Is Hard

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

Normal Operation

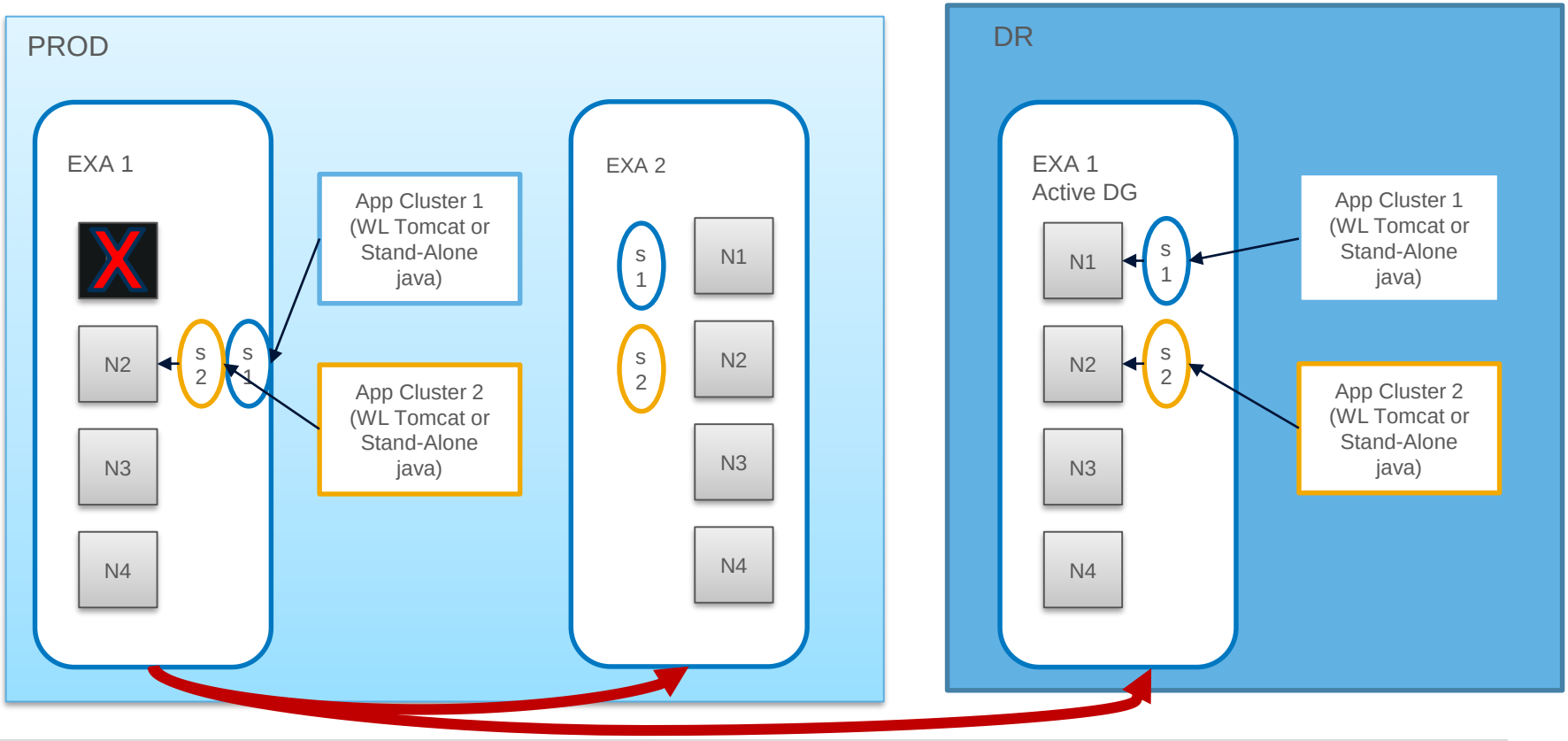
- All OLTP services configured as 1 active, rest available
- Over 400 services across environment
- Over 100 applications
- Node capacity actively managed

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



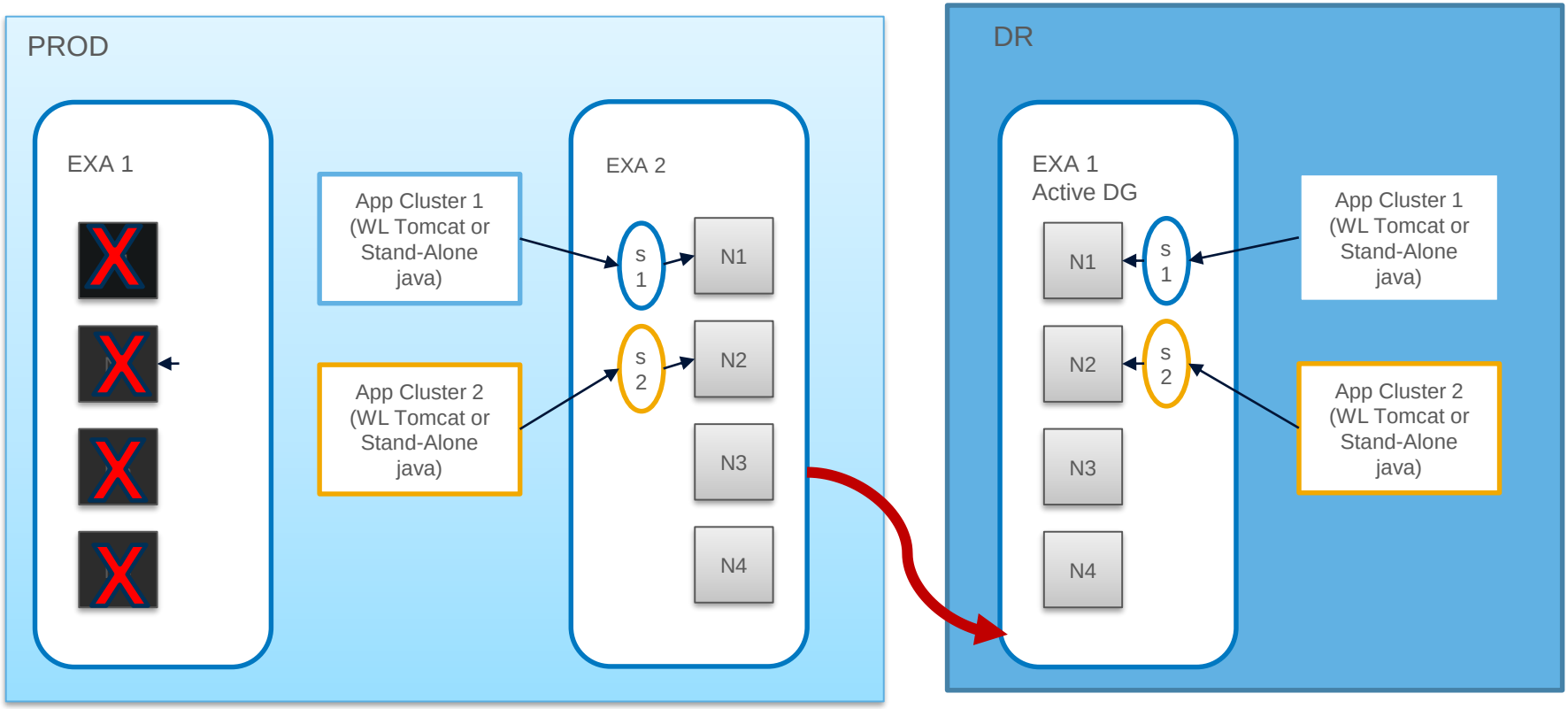
UNPLANNED OUTAGES

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



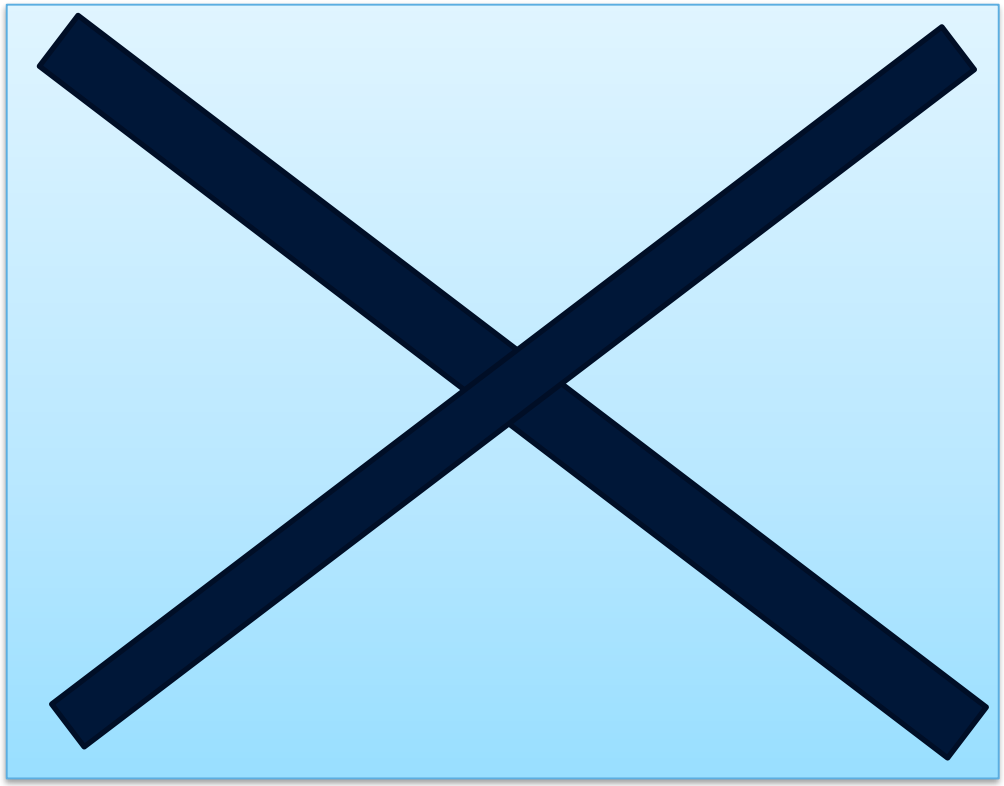
UNPLANNED OUTAGES

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

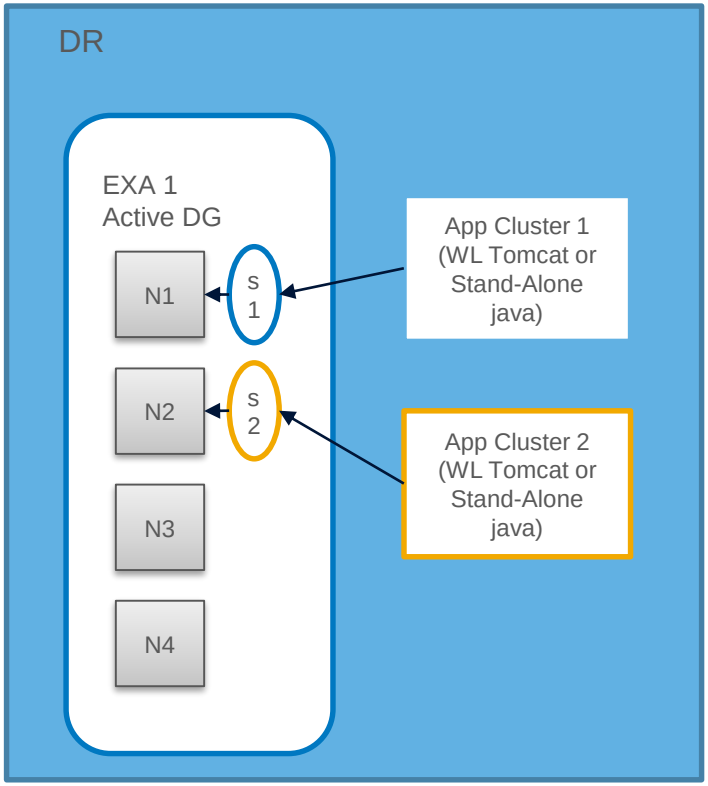


UNPLANNED OUTAGES

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

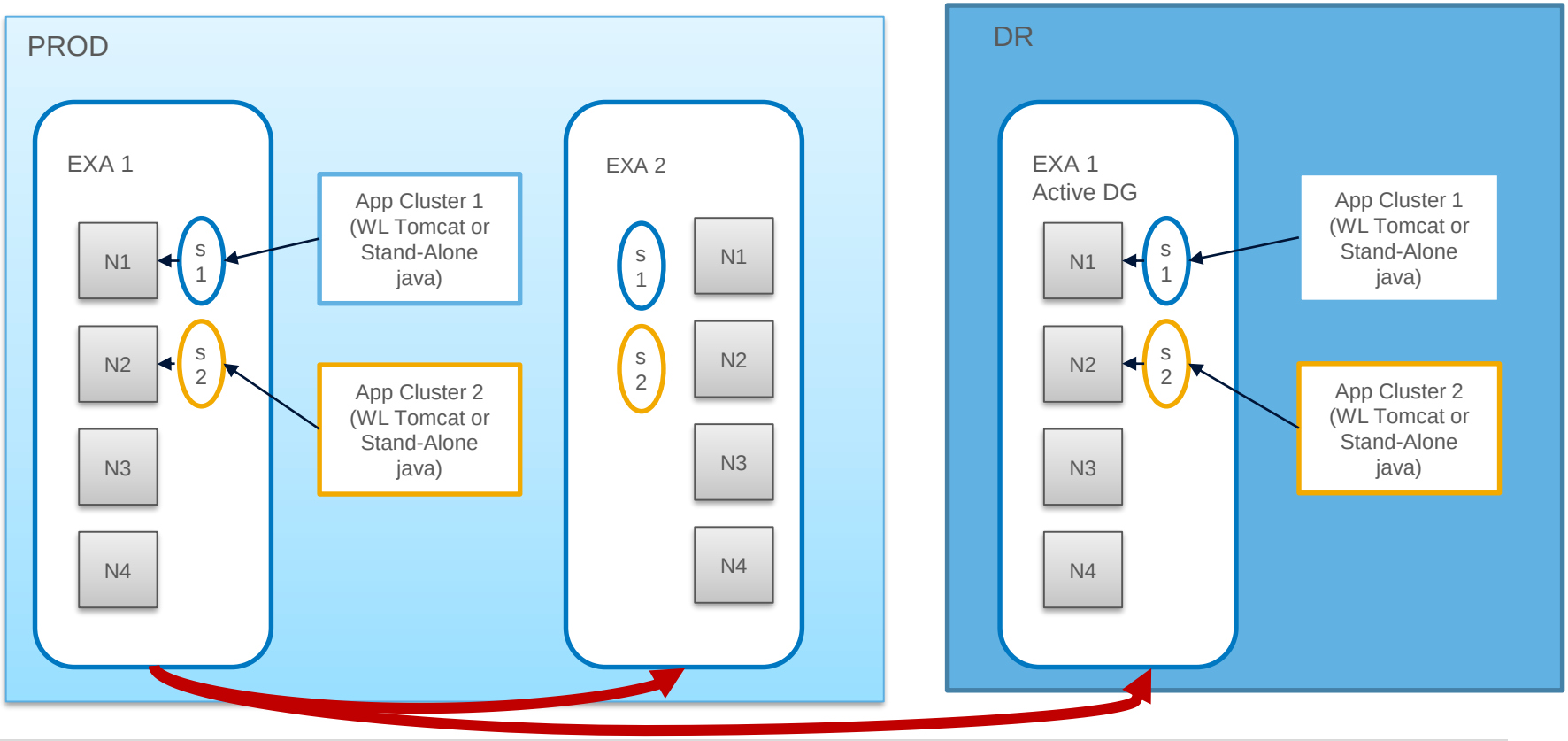


- Critical Apps Up for customer RO access
- Databases are converted – Apps convert to RW
- All apps started - < 2 hours
- All automated



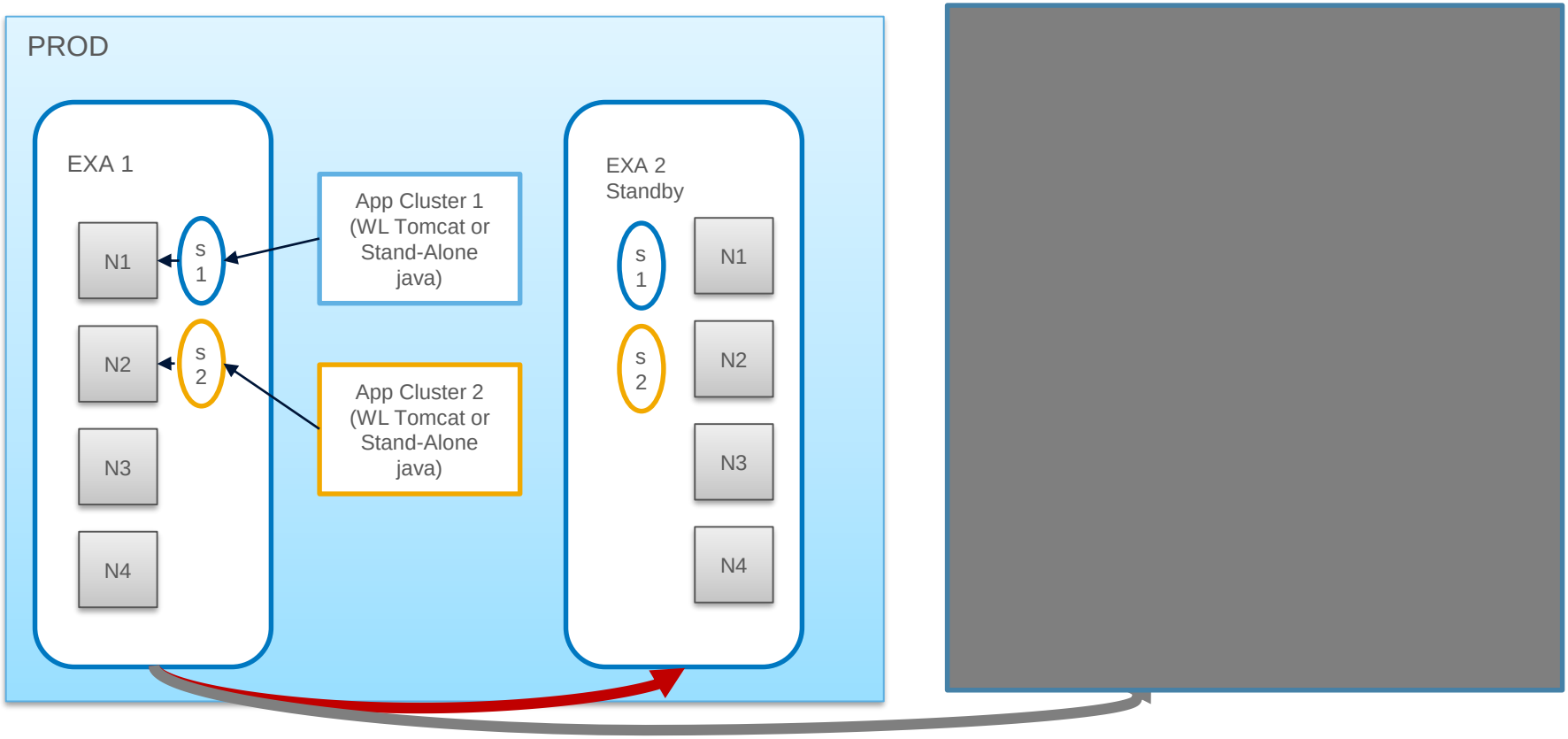
Planned Maintenance

- 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.



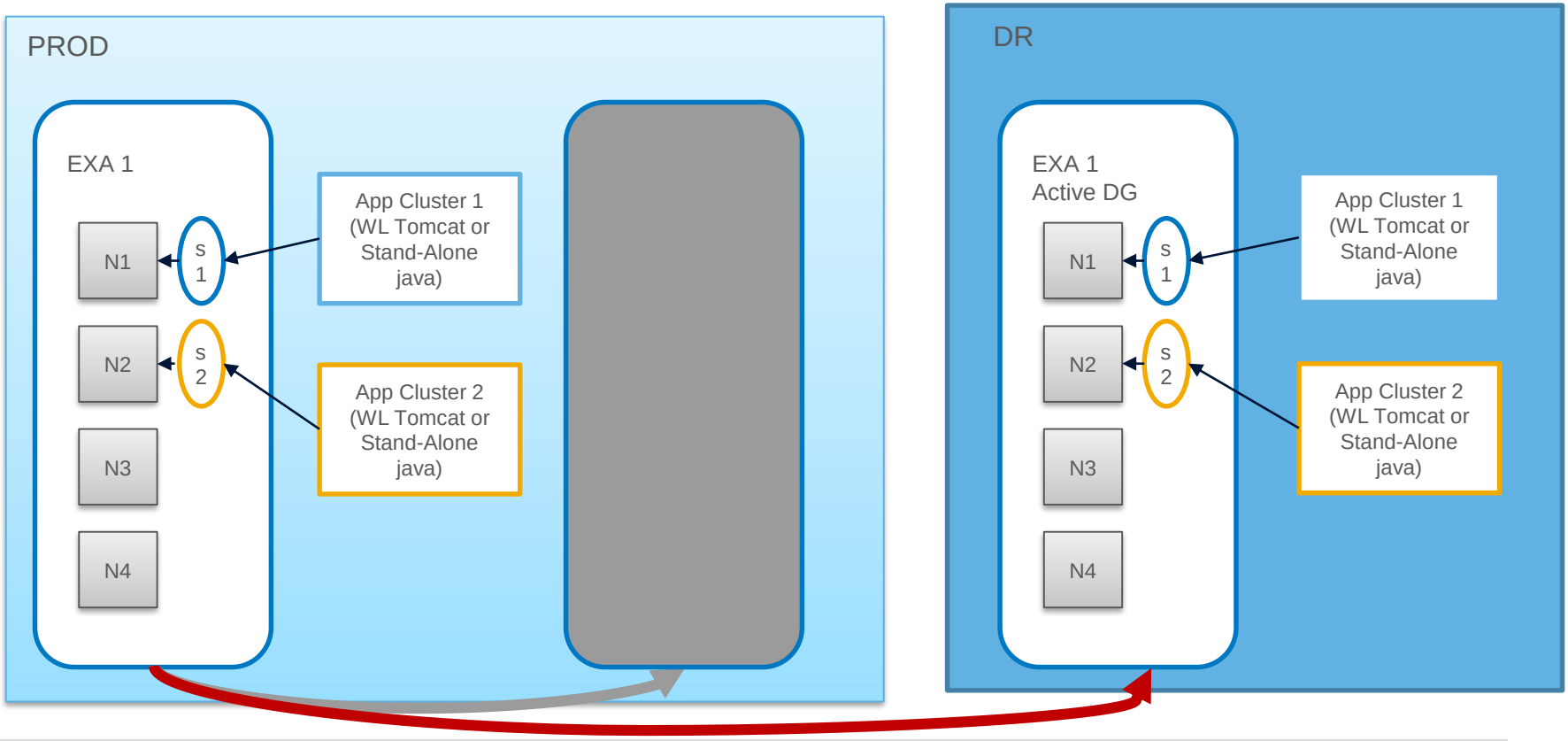
Planned Maintenance

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



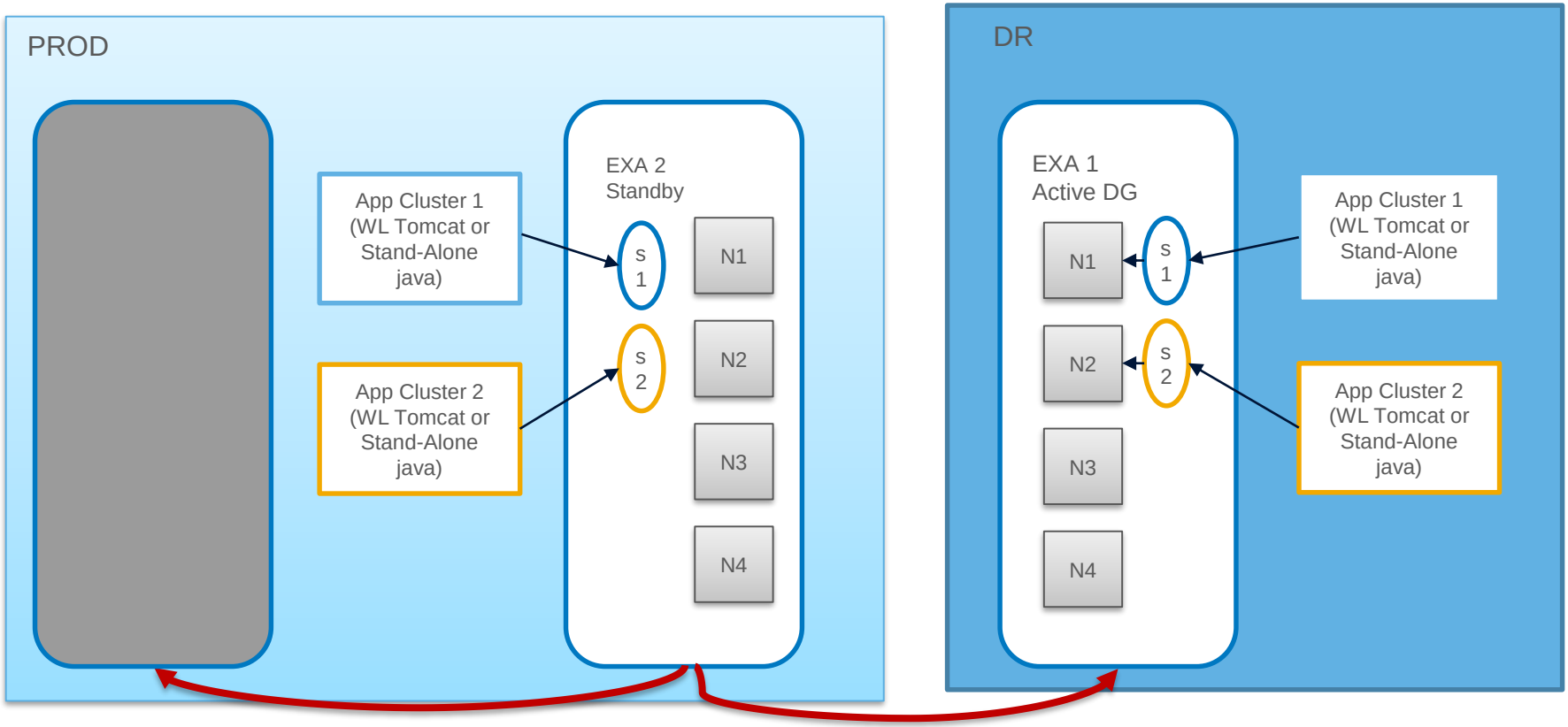
Planned Maintenance

- 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 application stay running
- 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 continue during maintenance window
- Process repeated for fail back



Application Continuity

Configuration

SERVER SIDE CONFIGURATION

Set Service Attributes

- Set Service Attributes in both Primary and standby.
  - srvctl modify service -d DBNAME -s SERVICE -failovertype TRANSACTION -replay_init_time 1800 -failoverretry 60 -failoverdelay 3 -commit_outcome TRUE -retention 86400 -notification TRUE
- Permissions are needed in order to retain mutable values (sequences) during replay
  - GRANT KEEP DATE_TIME, KEEP SYS_GUID to <USER>;
- For each sequence:
  - Alter sequence <name> keep;
  - Grant keep sequences on <name> to <user>;

SQL*NET CONFIGURATION

- Create or alter connection descriptors
- AT CME Oracle LDAP is used so that the DBA staff can control application connection behavior
- TYPICAL APPLICATION CONNECTION STRING

  jdbc:oracle:thin:@ldaps://ORAQALDAP:3131/DCEPSA,cn=OracleContext,dc=world

- LDAP Contents (example command)

  ldap.sh -a DCEPSAC "(DESCRIPTION=(CONNECT_TIMEOUT=240)(RETRY_COUNT=60)(RETRY_DELAY=3)(ADDRESS_LIST=
  (LOAD_BALANCE=ON)(ADDRESS=(PROTOCOL=TCP)(HOST=tddvdb0001d)(PORT=1521)))
  (ADDRESS_LIST=(LOAD_BALANCE=ON)(ADDRESS=(PROTOCOL=TCP)(HOST=tddvdb0002d)(PORT=1521)))
  (CONNECT_DATA=(SERVICE_NAME=DCEPSAC.WORLD)))"

SQL*NET CONFIGURATION

TNS Properties

```
(DESCRIPTION_LIST=
   (LOAD_BALANCE=OFF)
   (FAILOVER=ON)
   (DESCRIPTION=
      (CONNECT_TIMEOUT=90)(RETRY_COUNT=10)(RETRY_DELAY=3)
      (ADDRESS=(PROTOCOL=TCP)(HOST={primary_scan_vip})(PORT={primary_port}))
      (LOAD_BALANCE=yes)
      (CONNECT_DATA=(SERVER=DEDICATED)(SERVICE_NAME={service_name}))
   )
   (DESCRIPTION=
      (CONNECT_TIMEOUT=90)(RETRY_COUNT=10)(RETRY_DELAY=3)
      (ADDRESS=(PROTOCOL=TCP)(HOST={standby_scan_vip})(PORT={standby_port}))
      (LOAD_BALANCE=yes)
      (CONNECT_DATA=(SERVER=DEDICATED)(SERVICE_NAME={service_name}))))
```

CLIENT SIDE CONFIGURATION

Apply Latest Patch

- Use Latest 12.1.0.2 ojdbc7, ons and ucp all matching jars
- Patch 22650072 - MERGE REQUEST ON TOP OF 12.1.0.2.0 includes the fix for the root cause with is Bug 21666072 - UCP connections not drained after DataGuard switchover
- Patch 19154304: JDBC: RETRY_COUNT DOES NOT RETRY WHEN SERVICE DOWN AS REQUIRED

CLIENT SIDE CONFIGURATION

UCP – USE JDBC REPLAY DATA SOURCE

```
import oracle.ucp.jdbc.PoolDataSource;

import oracle.ucp.jdbc.PoolDataSourceFactory;

…

DriverManager.registerDriver(new oracle.jdbc.OracleDriver());

final PoolDataSource pds =
PoolDataSourceFactory.getPoolDataSource();

pds.setConnectionFactoryClassName("oracle.jdbc.replay.OracleDataSourceImpl");

pds.setUser(username);

pds.setPassword(password);

pds.setURL(dburl);
```

CLIENT SIDE CONFIGURATION

SETUP ONS, FCF AND CONNECTION PROPERTIES

~~final String onsConfiguration = “node1:6200,node2:6200,…”;~~

~~pds.setONSConfiguration(onsConfiguration);~~

pds.setFastConnectionFailoverEnabled(true);

final Properties properties = new Properties();

// set connection timeout

properties.setProperty(oracle.net.ns.SQLnetDef.TCP_CONNTIMEOUT_STR, "1000");

// set autocommit off

properties.setProperty(
OracleConnection.CONNECTION_PROPERTY_AUTOCOMMIT, "false");

pds.setConnectionProperties(properties);

CLIENT SIDE CONFIGURATION

ALWAYS RETURN CONNECTIONS TO THE POOL

```
try (final Connection conn = pds.getConnection()) {
    conn.setAutoCommit(false);
    // DO WORK: Prepare Statements, Execute Queries, Execute Procs, etc
    conn.commit();
}
```

CLIENT SIDE CONFIGURATION

Spring Template Configuration

```
<?xml version="1.0" encoding="UTF-8"?>

<beans xmlns="http://www.springframework.org/schema/beans"

       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

       xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-3.0.xsd">


   <!-- Initialization for data source -->

   <bean id="dataSource" class="oracle.ucp.jdbc.PoolDataSourceImpl">

      <property name="connectionFactoryClassName" value="oracle.jdbc.replay.OracleDataSourceImpl"/>
      <property name="URL" value="jdbc:oracle:thin:@ldap://ORAQALDAP:3060/ACTEST,cn=OracleContext,dc=world"/>
      <property name="user" value="TESTUSER"/>
      <property name="password" value="TESTPASSWORD"/>
      <property name="maxPoolSize" value="16"/>
      <property name="initialPoolSize" value="8"/>
      <property name="fastConnectionFailoverEnabled" value="true"/>

   </bean>

   <!-- Definition for EmpJDBCTemplate bean -->
   <bean id="ACJDBCTemplate" class="com.cmegroup.dba.ac.spring.ACJDBCTemplate">
      <property name="dataSource" ref="dataSource"/>
   </bean>
</beans>
```

CLIENT SIDE CONFIGURATION

Spring Template Bean

```
public static void main(String[] args) throws Exception {
    ApplicationContext context = new FileSystemXmlApplicationContext(args[0]);
    final ACJDBCTemplate template = (ACJDBCTemplate)
context.getBean("ACJDBCTemplate");
…
}
 public void setDataSource(DataSource dataSource) {
        this.dataSource = dataSource;
        this.template = new JdbcTemplate(dataSource);
    }
public void run() throws SQLException {
   templace.execute(“insert into …”);
}
```

LESSONS LEARNED

- Use latest jar versions with applied patches
- Set Auto commit Off.
- Return Connections to the Pool.
- Do not use Deprecated JDBC Classes.
- If a request has a call that should not be replayed, replay can be disabled.
- Reinitialize Connection using Callbacks for applications that set state outside database requests.

Reducing Brownout

NODE FAILURE AND RECOVERY PROCESS

- Cluster Resource polling
- Critical Resources
  - Interconnect
  - Disk
- Triggers I/O fencing – begins brownout
- Enhanced on Exadata

- Eviction started
- CRS
- Cluster Reconfiguration
- Services relocate – Active/Passive
- FAN/FCF notifies clients and interrupts processing

- Instance reconfiguration
- Instance Recovery
- Ends brownout

REDUCING THE BROWNOUT

COMPUTE NODE AND INSTANCE FAILURE			
SETTING	DEFAULT	CME	NOTES
CSS MISSCOUNT	30S	3S	Single most impactful change
CSS REBOOTTIME	5s	1s	
PING TIMEOUT			VIP and public network
FAST_START_MTTR_TARGET	0	1	Database instance parameter
Flash Disk	Exadata	Exadata	Greatly speeds instance recovery
DATAGUARD			
Service Role	Primary	PRIMARY,PHYSICAL_STANDBY	Post switchover, standby services start up automatically
LogArchiveMaxProcesses	4	6	Initial for arch processes to run, faster recovery
ReopenSecs	300	30	Time before ARC process should retry access to failed dest
Max availability	SYNC	FASTSYNC	FastSync with Redo-routes

CME TESTS FOR HA

- The testing described here happens whenever database software or hardware changes. It is in addition to application testing
- A micro benchmark program written in Java was used for this testing.
  - Performs continuous DML across 3 tables – variable rates
  - Configurable multi threaded - scales
  - JDBC batching
  - Records / tracks database response time (latency)
  - AC compliant
- The following failure scenarios were tested
  - Instance Failure – A critical background process is killed at the O/S level
  - Node failure – A kernel break command is used to kill the node immediately (not graceful)
  - Node Reboot – Fast graceful shutdown is initiated at O/S
  - CPU/Memory Starvation – purposely cause an increase in system CPU until CSS/CRS stops responding

CME BENCHMARK RESULTS

TEST	DEFAULT	CME	NOTES
Instance Kill	.5ms	.5ms	Default settings
Service bounce	5.228sec	5.228sec	Default settings
Kernel Panic	19.4sec	15.1sec	CME- CSS Miscount=1
Node Reboot	16.353sec	07.427sec	CME- CSS Reboottime -1
Memory overload	Average 1sec delay per thread for 30sec	Average 1 sec delay per thread for 28sec	Ramp up Memory consumption till Crash
Dataguard SwitchOver	2:21mins	41secs	Planned Maintenance


CME Group