Hochverfügbarkeit mit dem Applikationsserver WebLogic Server Cluster

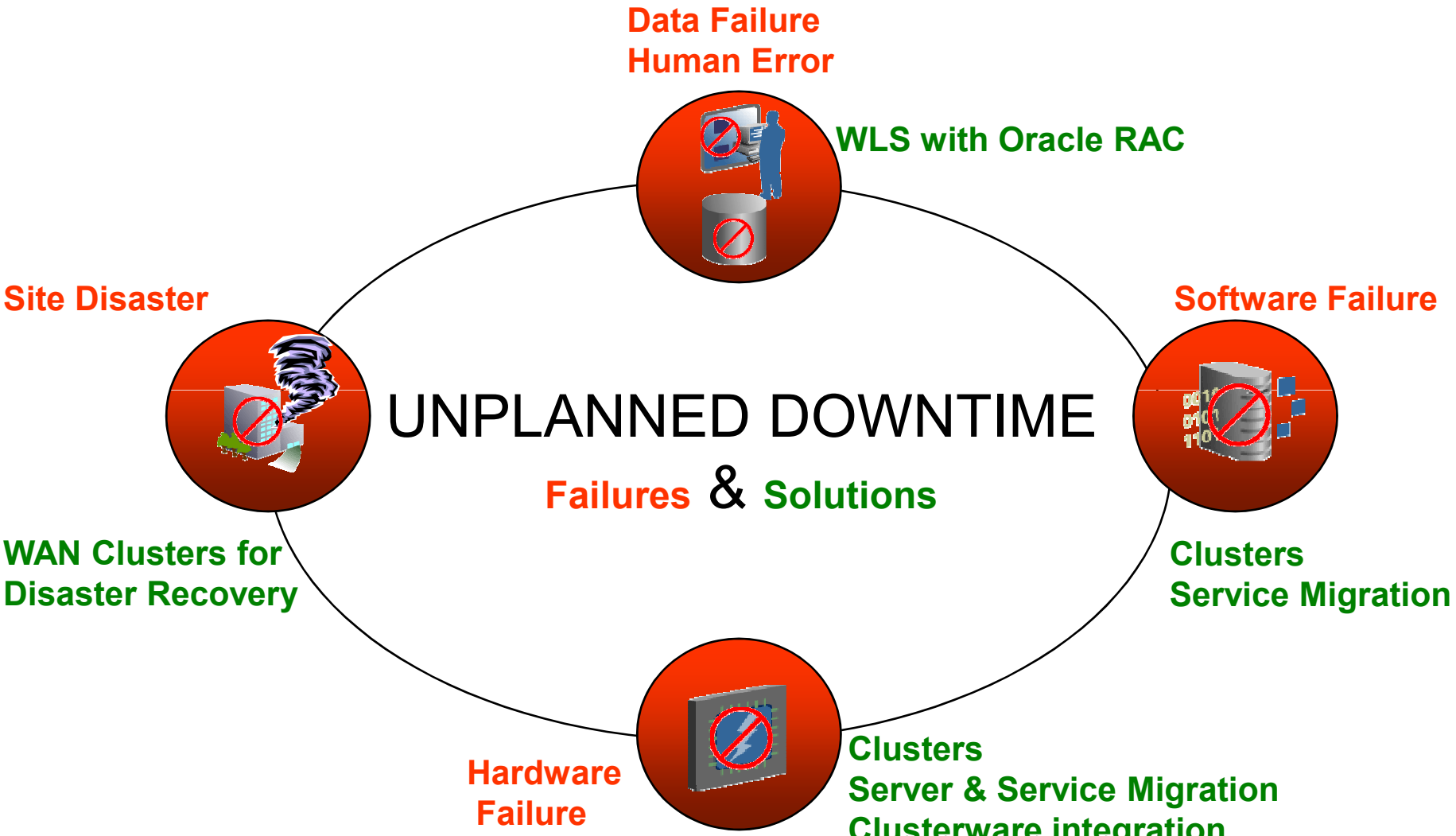
ORACLE®

Wolfgang Weigend
Senior Leitender Systemberater
ORACLE Deutschland GmbH

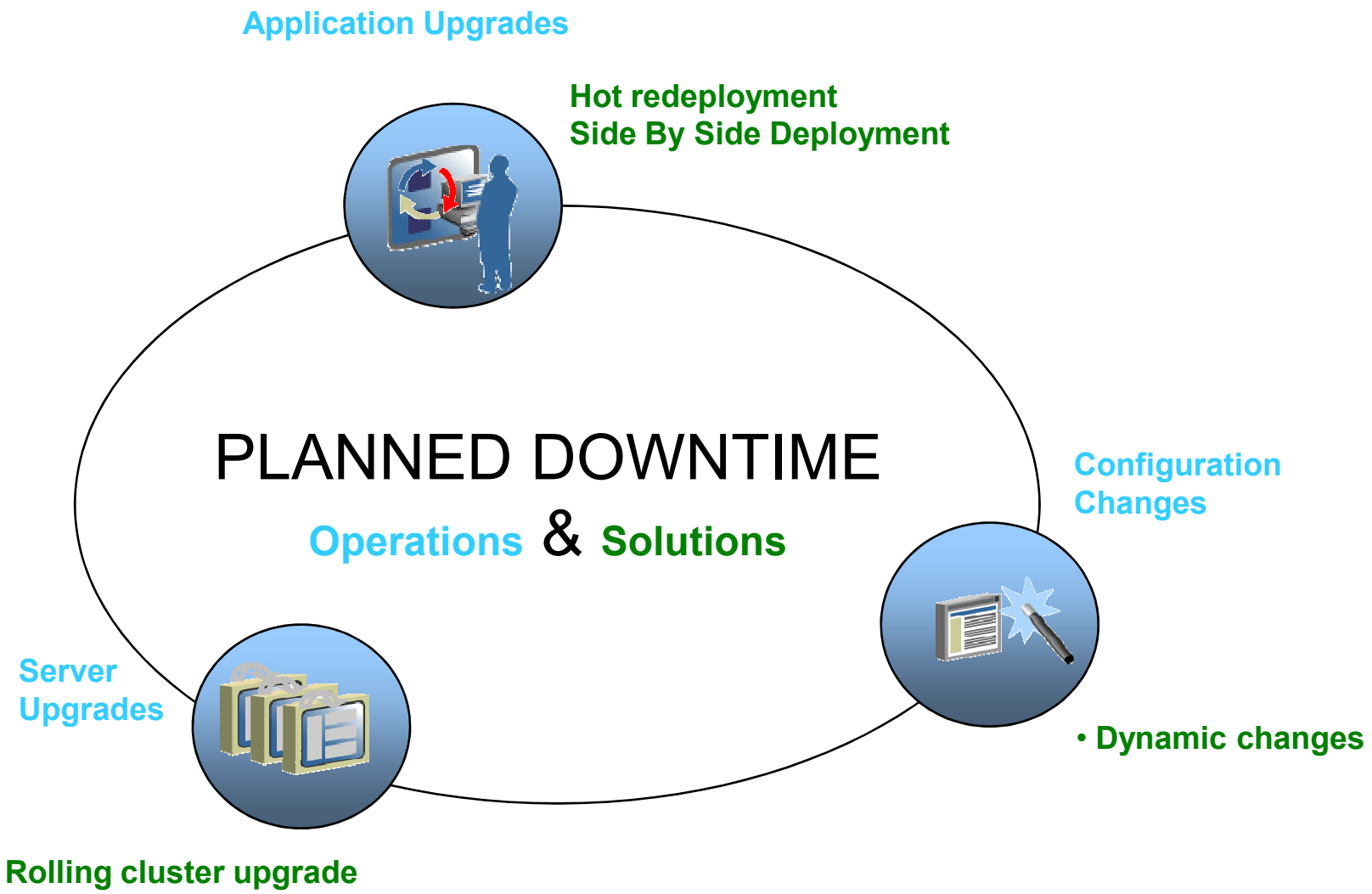
Agenda

- WebLogic Cluster Lösung
  - Hochverfügbarkeit und WebLogic Server Cluster Topology
  - Exemplarische Beispiele und Betriebskonzepte
- Administration im Cluster
  - Node Manager
- Kommunikation im Cluster
  - JNDI
- Verfügbarkeit
  - Was bedeutet das für die Anwendungen?
  - Was bedeutet es für die Entwicklung?
    - Programming Applications for WebLogic Server Clusters
  - Demo
- Architektur Konzepte und Empfehlungen
- Zusammenfassung

WebLogic Server Hochverfügbarkeit

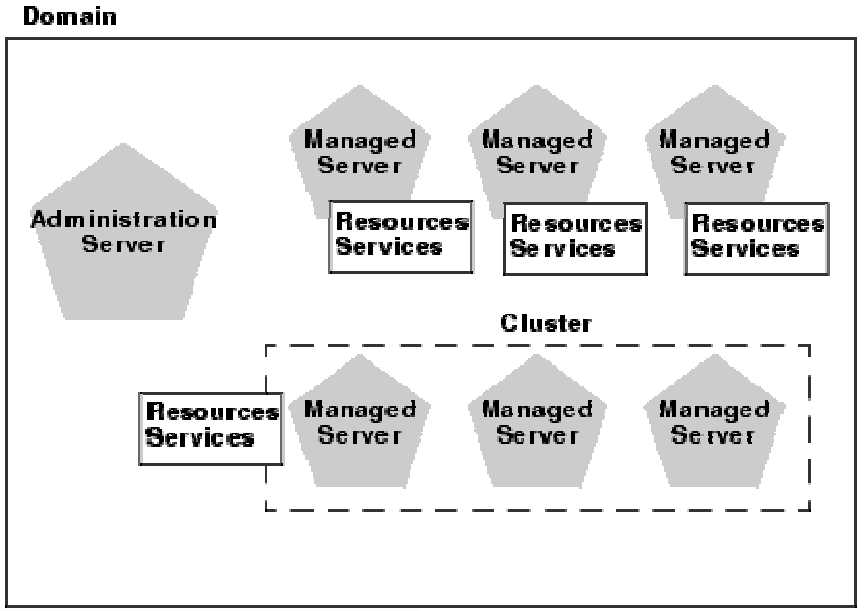


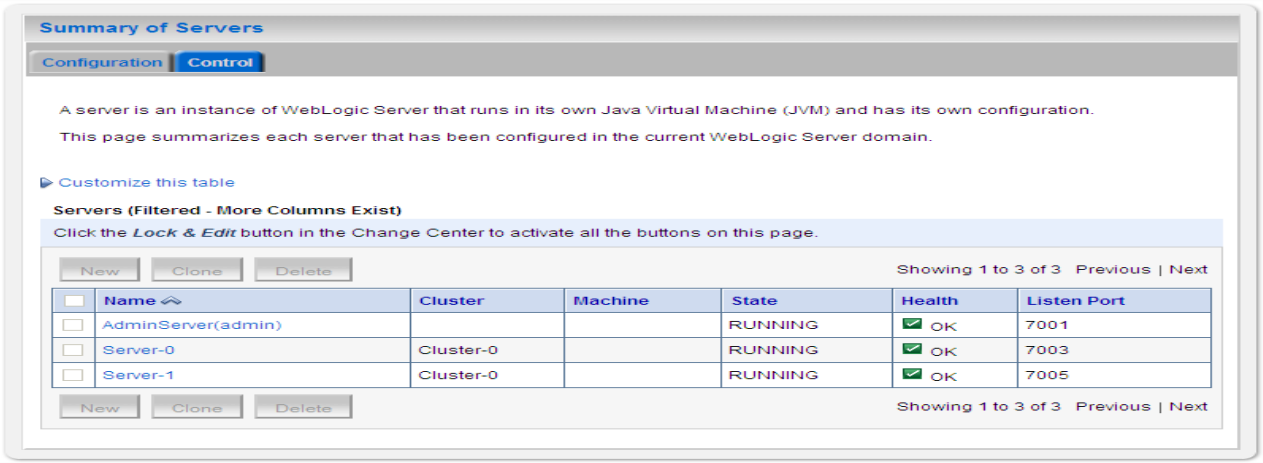
WebLogic Server Hochverfügbarkeit



WebLogic Server Cluster Topology

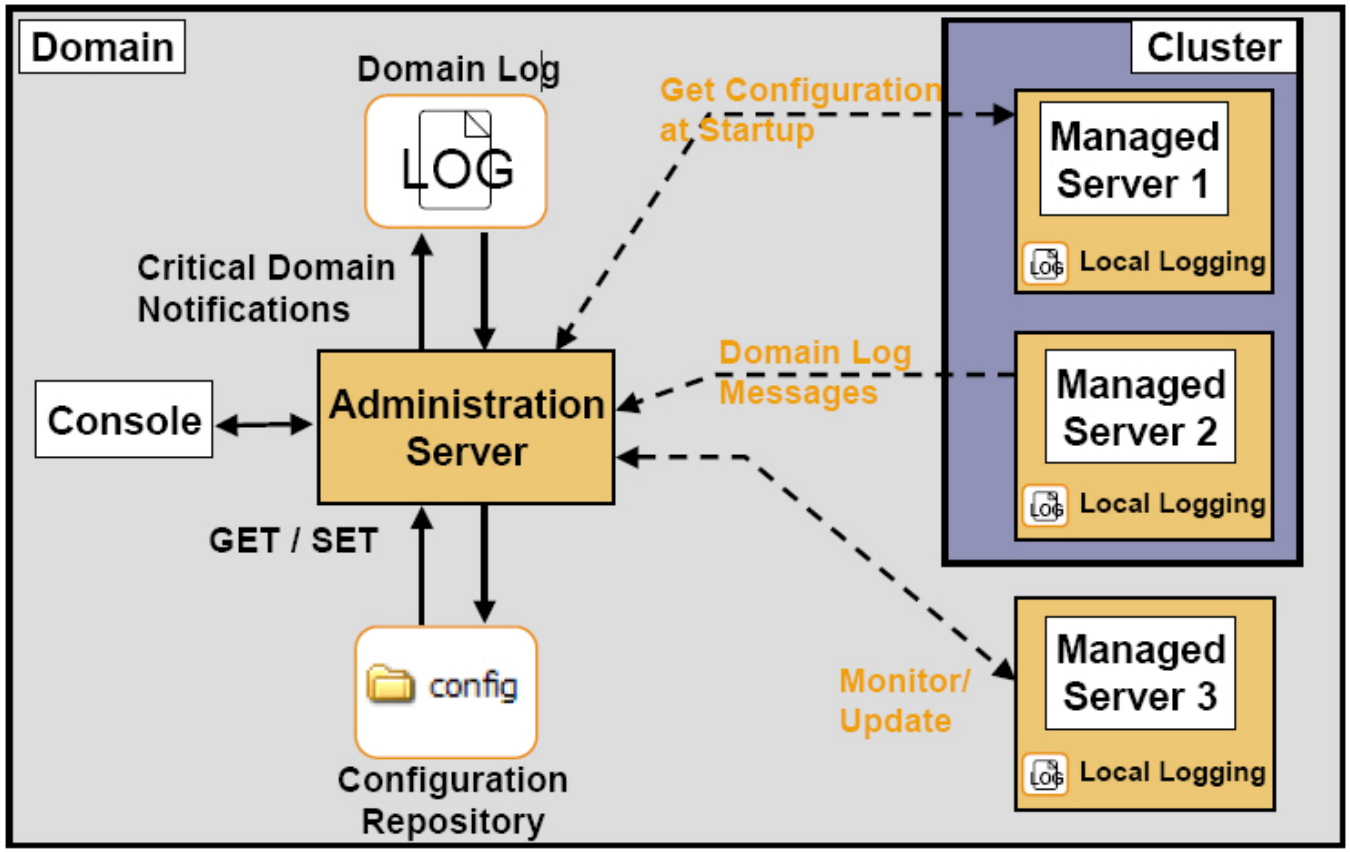
- **Domain** - Gruppe von Instanzen mit einheitlicher Kontrolle
- **Administrations Server** - Zentrale Konfigurations Kontrolle für die Domain
- **Managed Server** - Instanz für Applikationen und notwendigen Ressourcen
- **Cluster** - Gruppe von Managed Servern für erhöhte Skaliebarkeit und Zuverläßigkeit
- **Node Manager** – Prozess pro Maschine zum Starten und Stoppen von Instanzen
- **Flexibele Architecture** – Konfiguration für flexibele Anforderunegn





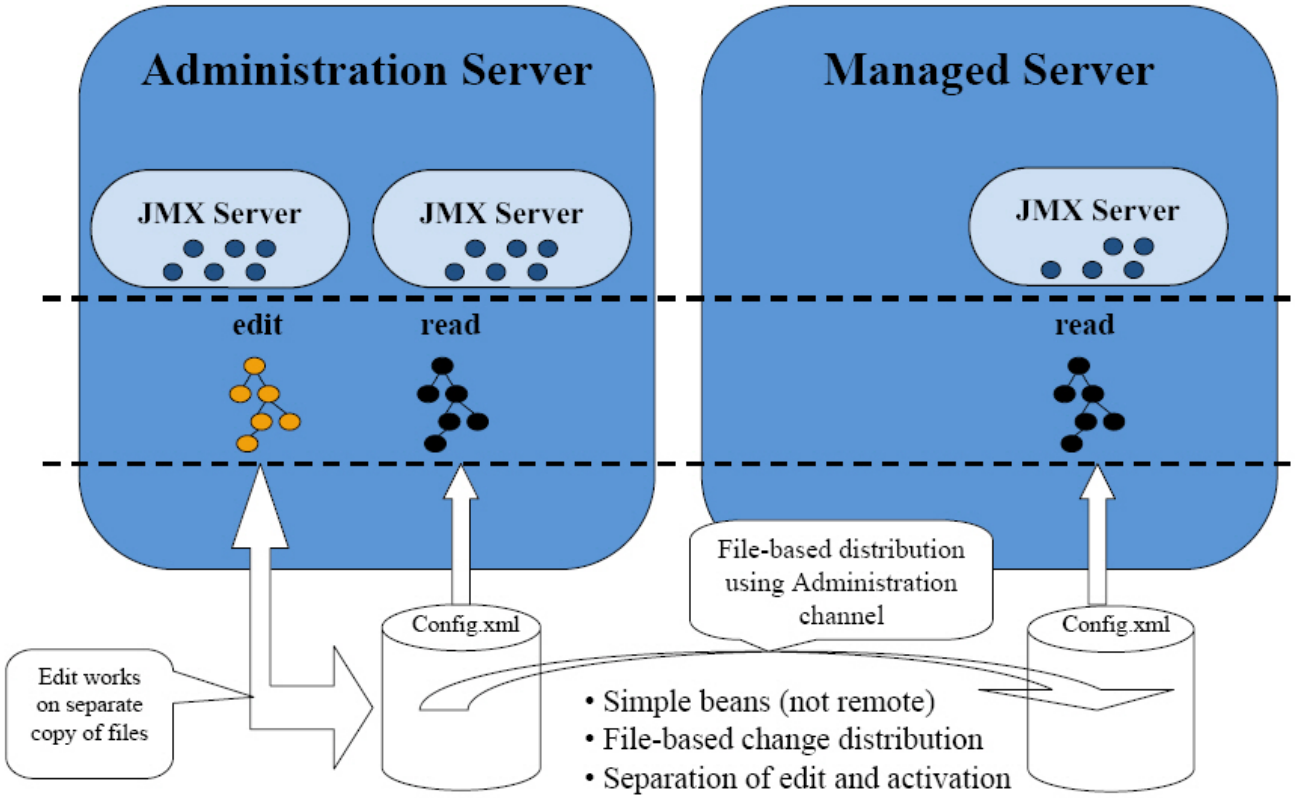
Konfiguration der WebLogic Server Umgebung

Domain Struktur



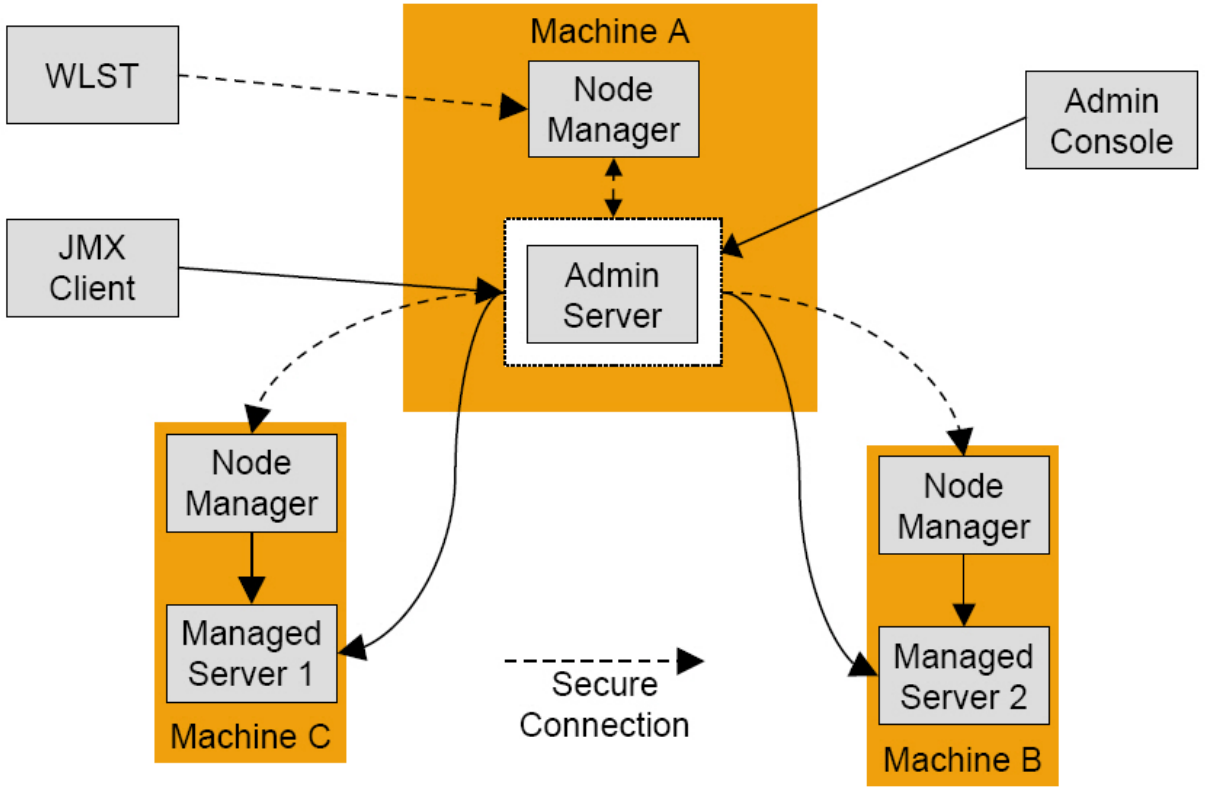
Verwaltung der WebLogic Server Umgebung

Konfigurations Management Architektur



Verwaltung vom WebLogic Server

Die Node Manager Architektur



- Entfernte Administration mit dem Node Manager
  - Remote Start und Stop von Managed Servern, Cluster, und Domänen.
  - Überwacht und verwaltet den Server
  - Windows Service oder Unix Deamon

WebLogic Server Hochverfügbarkeitslösung

- Definition
  - Cluster
  - Benefits
- Cluster communication
  - IP Multicast/Unicast
  - IP Sockets
- Web Cluster
  - Proxy plug-in
  - External load balancer
  - Sync/Async session replication
- EJB/RMI clustering
  - Replica-aware stub
  - Multiple LB algorithms
- Cluster wide JNDI
- MAN/WAN replication
- Whole Server Migration
- Automatic Service Migration
- Hot Redeployment
- Side-by-side deployment
- Dynamic config changes
- Rolling upgrade
- WLS with Oracle RAC

Definition: What is WebLogic Cluster?

- Multiple WLS instances running simultaneously and working together.
- Cluster is part of a *WLS domain*. A domain can have multiple clusters.
- Cluster members can run in same machine or be located on different machines.
- Rolling upgrade of cluster members is supported.
- Clients view a cluster as a single WLS instance.

Definition: Key Benefits of Clustering

- Scalability
  - Load Balance
    - Even distribution of jobs
    - Multiple copies of an object that can do a particular job must be available
- High-Availability
  - Failover
    - When a object processing a job becomes unavailable, a copy of the object elsewhere takes over and finishes the job

Cluster Communication

- Communication among cluster members
  - IP multicast or Unicast
    - Broadcasting heartbeats and availability of services
  - Muxers (for exchanging data within clusters)
    - Clients use Java muxers
    - WLS uses Native muxers
      - Epoll, Devpoll, Posix – Unix
      - NT Muxer - Windows

Cluster Communication
Use of IP Multicast / Unicast

- Each cluster member instance uses multicast or Unicast for
  - Cluster heartbeats
    - Broadcast regular "heartbeat" messages to advertise its availability.
    - Maintains list of live cluster members when “heartbeat” is received from a member.
  - Cluster-wide JNDI updates
    - Announce the availability of clustered objects that are deployed or removed locally.
    - Updates local JNDI after receiving announcements for clustered objects from peers.

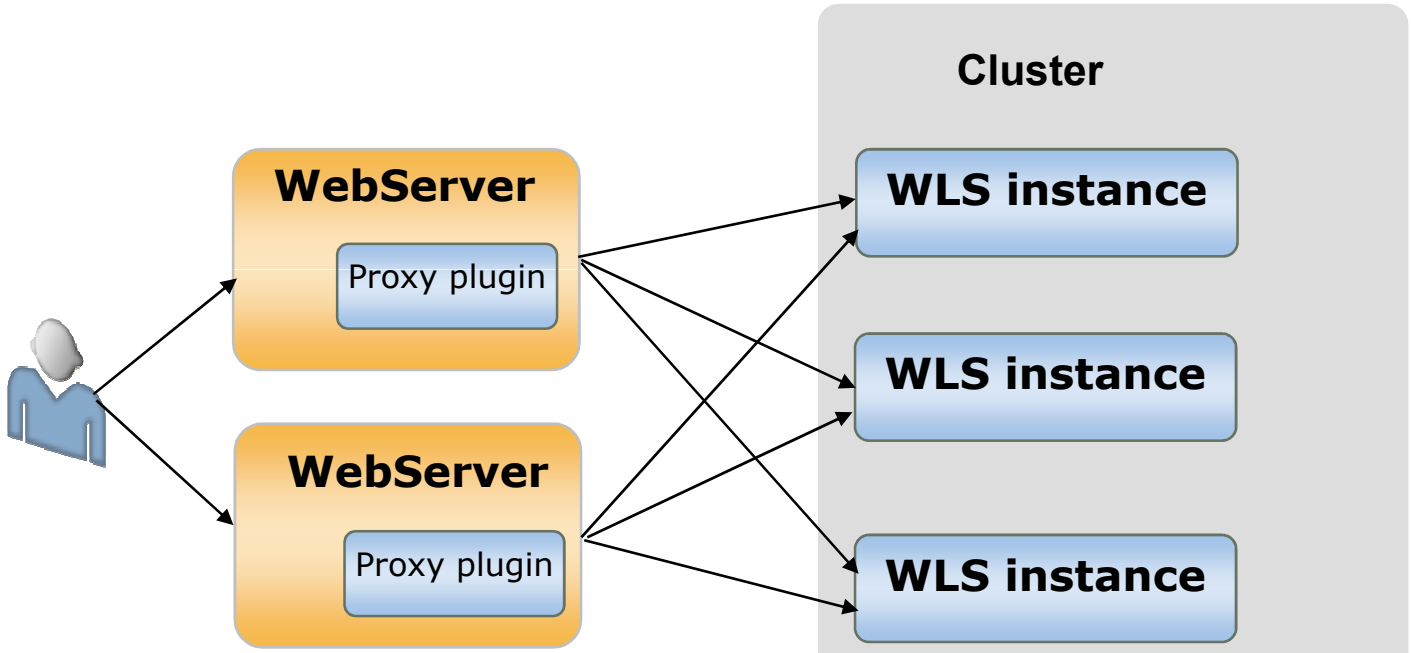
Cluster Communication
Use of IP Multicast / Unicast

- Unicast based cluster messaging
  - WLS clustering can work without Multicast !
  - TCP based
  - Avoids N-way connectivity
  - Designed to reduce message hops
  - Scalable to large node clusters

Web Cluster (JSPs and Servlets)

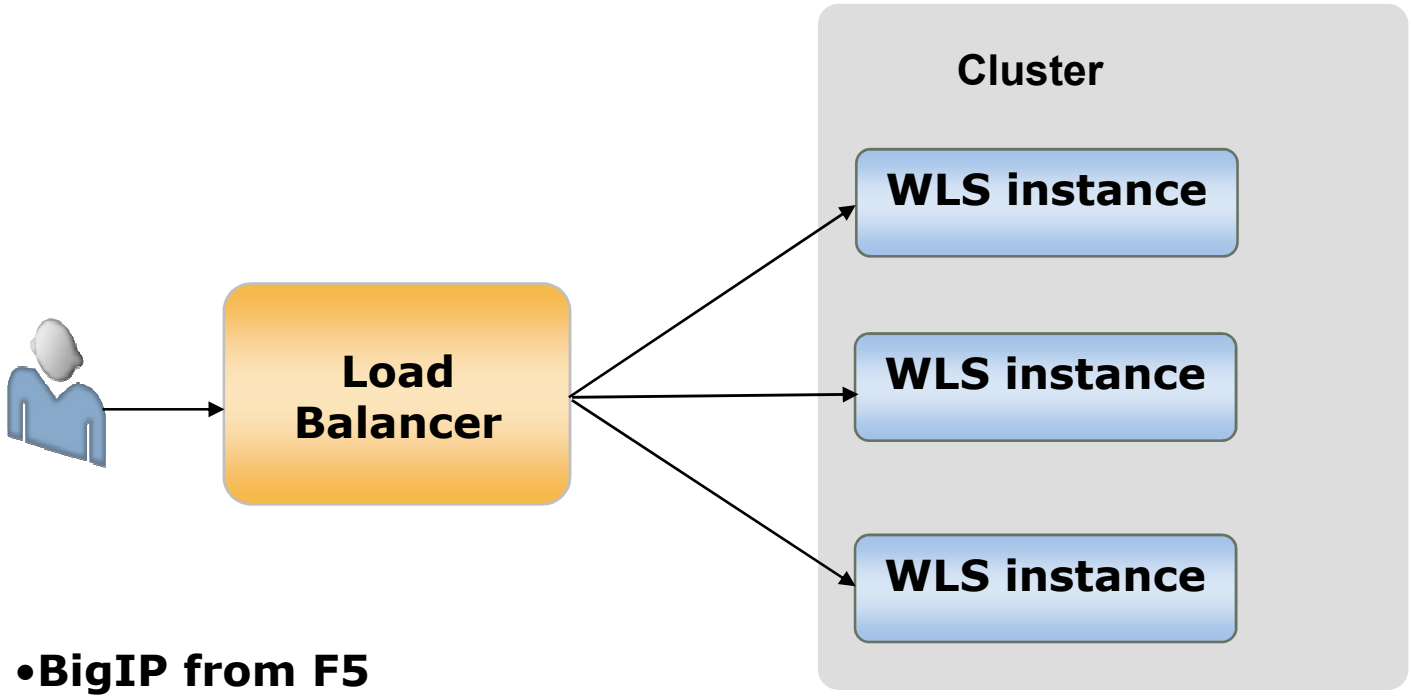
- Replication: HTTP session state of clients
  - Primary replicates session to Secondary (both Sync and Async)
- Replication: Failover
  - Initiated by load balancer after encountering an error
  - Secondary becomes the new Primary and chooses a Secondary
- Load Balance
  - New client sessions are load balanced
  - Must maintain “session affinity” or “sticky” load balance
  - Types of load balancers
    - Proxy plug-in running within a iPlanet, Apache or IIS
    - HttpClusterServlet running within another WLS
    - External load balancer, e.g. BigIP/F5, Alteon/Nortel,Cisco
    - Load balancing algorithms: Round robin

Web Cluster
WebServer with proxy plug-in



- iPlanet/SunOne
- Apache
- IIS
- WLS with HttpCLusterServlet

Web Cluster
External Load Balancer



- BigIP from F5
- Alteon from Nortel
- Cisco

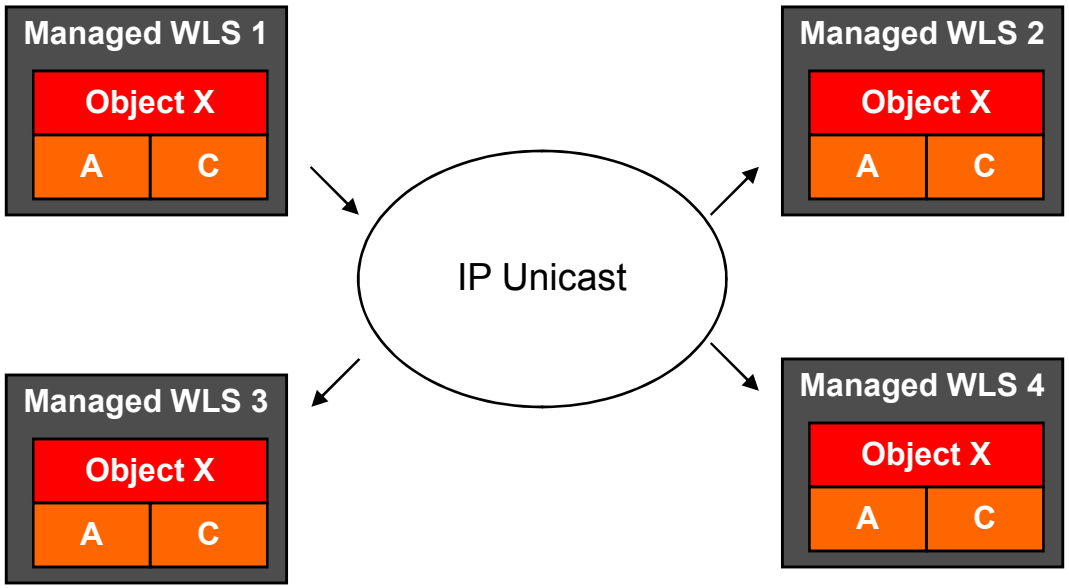
EJB/RMI Object Cluster
Replica-aware stub

- If an Object ((e.g. EJB) is clustered, instances of the object are deployed in all members, called replica.
- The stub that is returned to client is called “replica-aware” stub which represents the collection of replicas.
- The “replica-aware” stub
  - Load-balances method invocations based on load-balance policy (Round robin, weighted, random, server affinity)
  - If error occurred in invocation, fails over to a replica based on whether method is “idempotent”.

Hochverfügbarkeit mit WebLogic JNDI Cluster

Cluster Wide JNDI Service

- Objekt-Clustering für EJBs, JDBC, JMS, und eigene Objekte
- Jeder Server erzeugt und pflegt seine lokale Kopie der Cluster-weiten JNDI Baumstruktur
- Geschäftliche Auswirkungen
  - Kann normal weiterlaufen, auch wenn große Software- und Hardware-Infrastruktur-Ausfälle auftreten



EJB/RMI Object Cluster
EJB invocations

- Stateless Session EJB
  - Invocations are load balanced to all members where the EJB is deployed ("*replicas*").
- Stateful Session EJB
  - If not clustered, they are "pinned" to the member where created
  - If clustered, the state is replicated (to a secondary server instance) and the "*replica aware*" stub is aware of locations (primary and secondary).

- Using JNDI from Within Java EE Components

Although it is possible for Java EE components to use the global environment directly, it is preferable to use the component environment. Each Java EE component within a Java EE application had its own component environment which is set up based on information contained in the component's deployment descriptors.

Java EE components are able to look up their component environments using the following code:

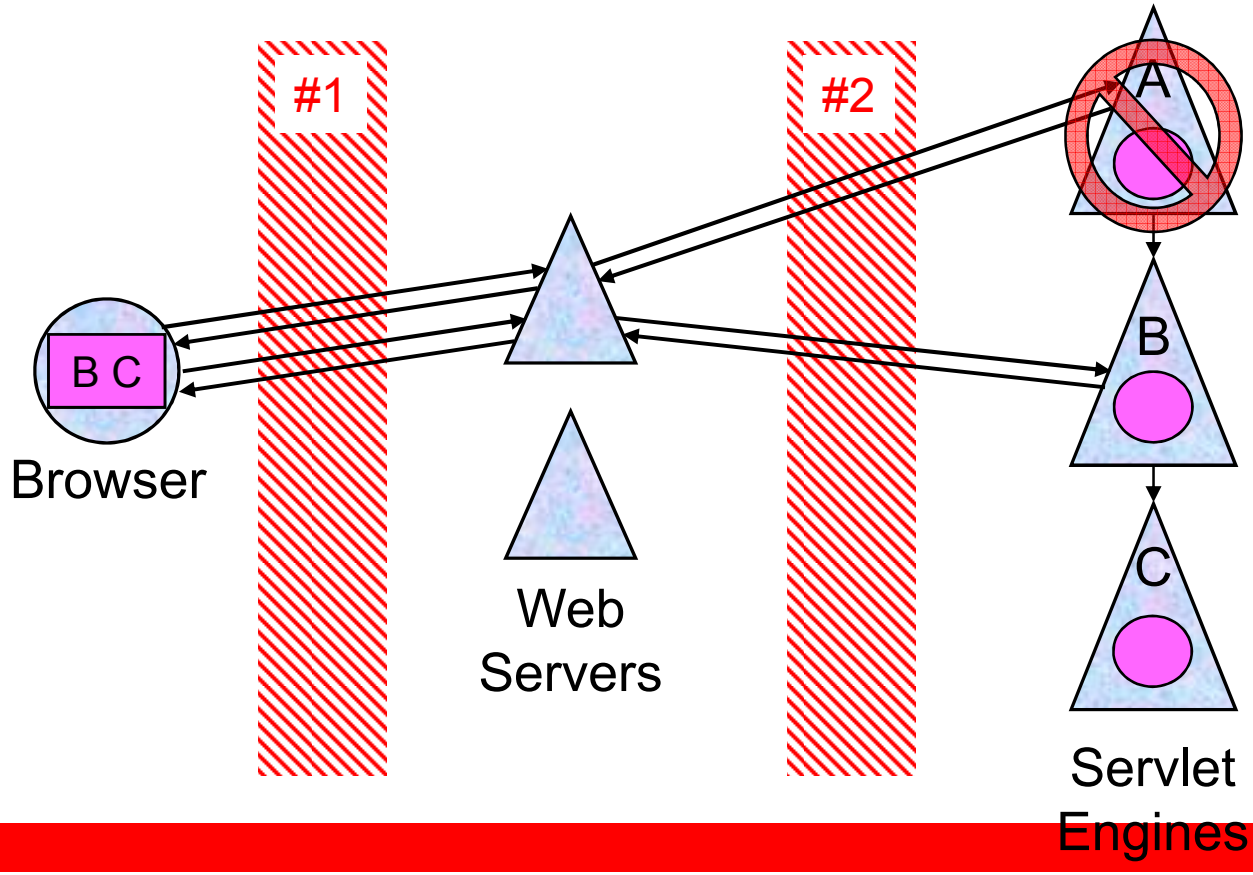
```
Context ctx = new InitailContext();
Context comp_env = (Context)ctx.lookup("java:comp/env");
```

Because you are working within a Java EE component, you do not need to set up the Hashtable or Environment objects to define the connection information

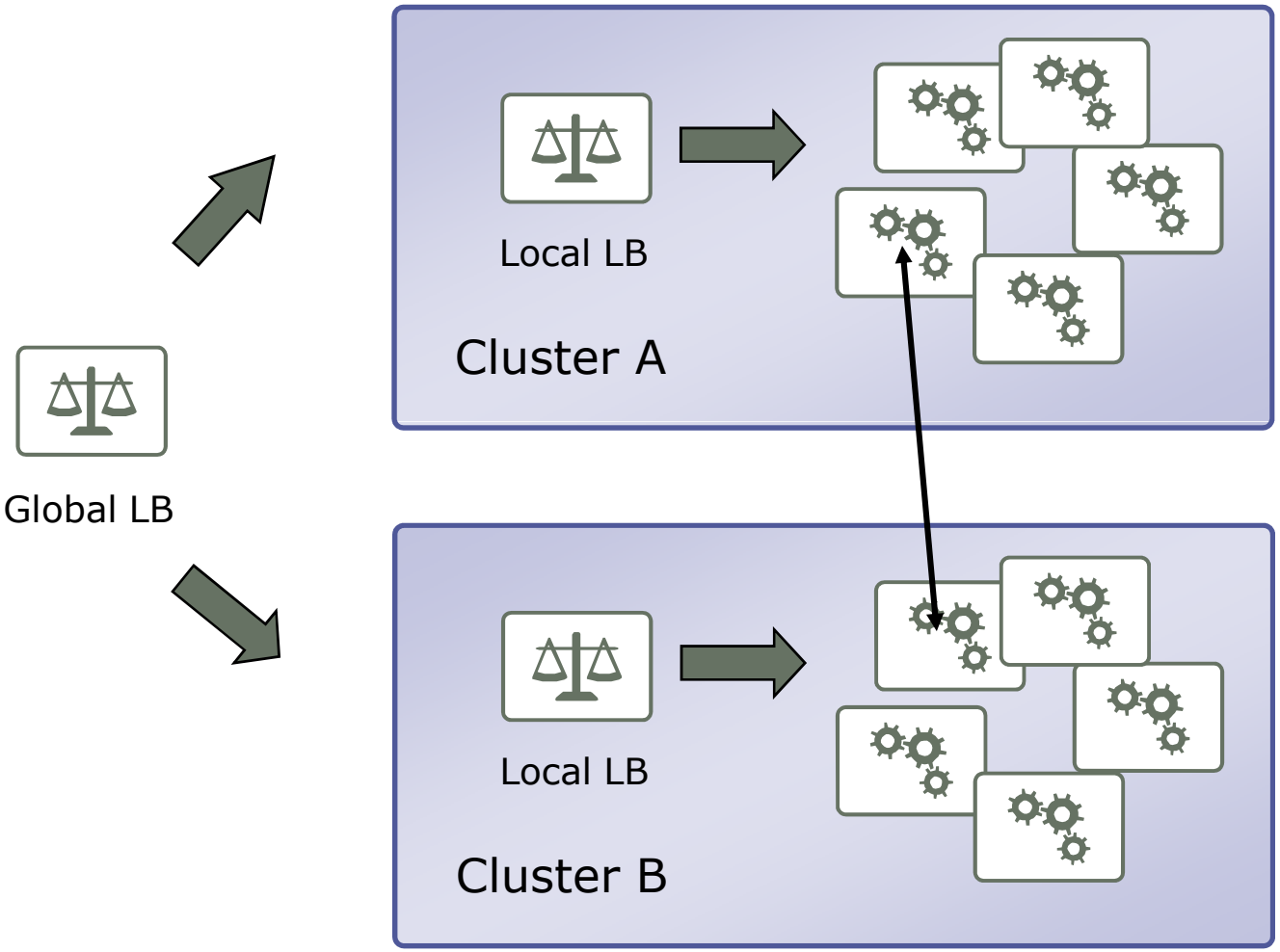
Session State Replication

LAN replication

Availability via synchronous or asynchronous, in-memory replication between primary and secondary

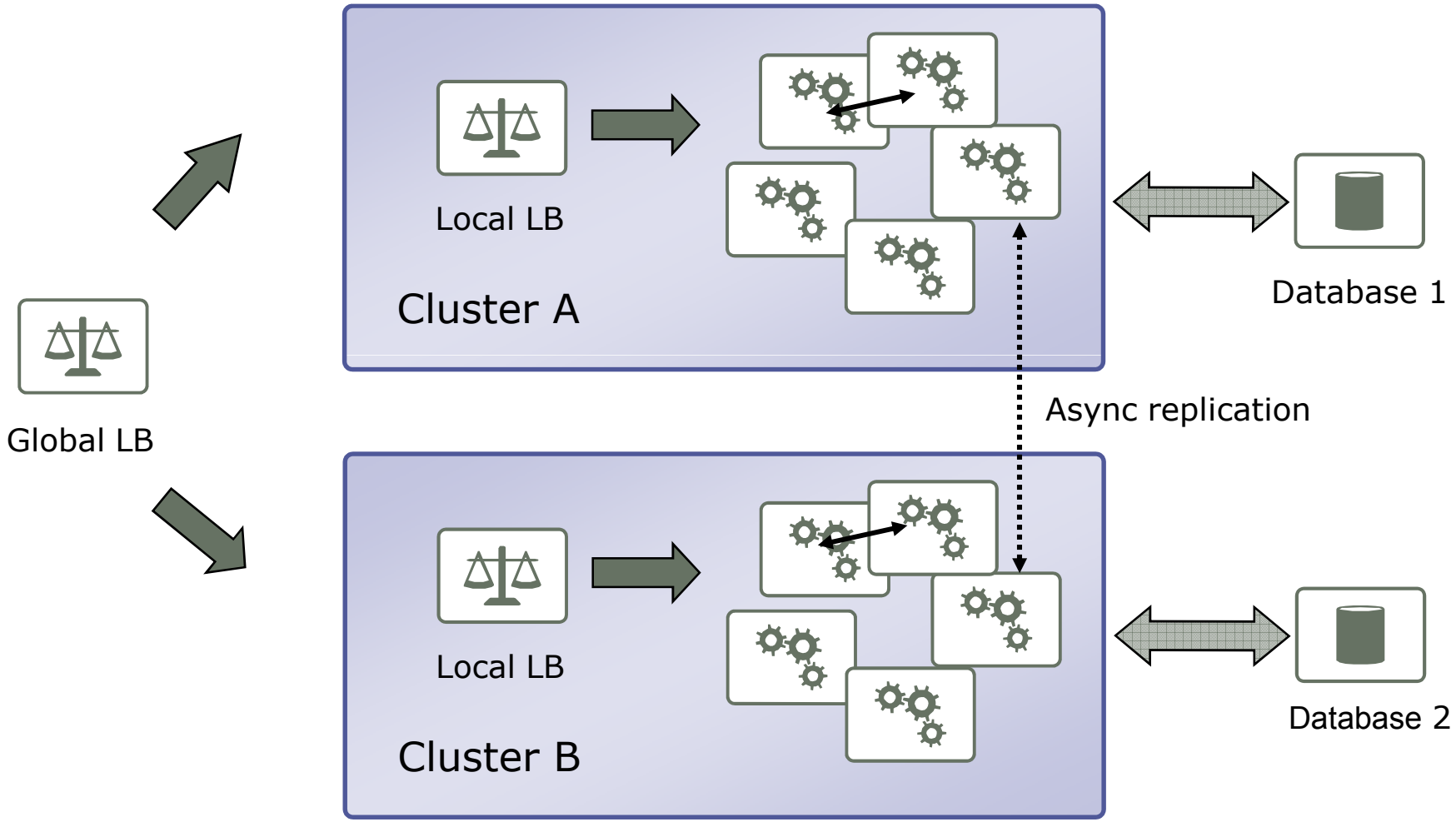


Metro-Area Network Replication



Disaster Recovery - Site Replication

Wide-Area Network Replication

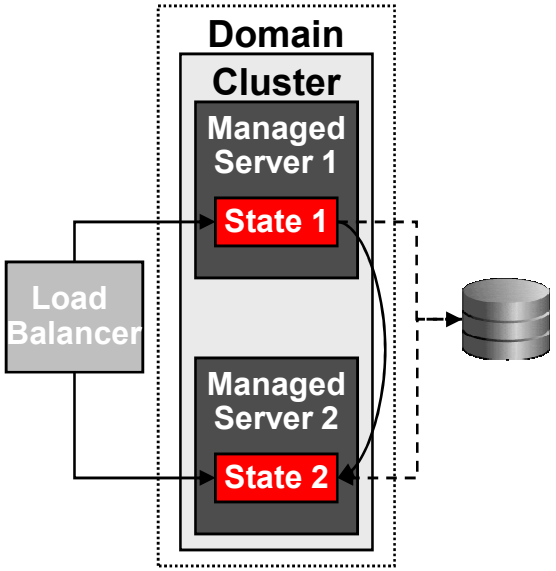


Wide-Area Network Replication
Best Practices

- WAN shouldn’t be treated as MAN+database
  - Over-frequent data flushing will cause it to behave like MAN, but with added synchronization overhead
- WAN can be used without remote cluster, acts as a database backup of session data
- Network configuration is the stumbling block for MAN
- Use WAN for site failover and very high availability
- For non-mission critical options, regular cluster replication may be enough

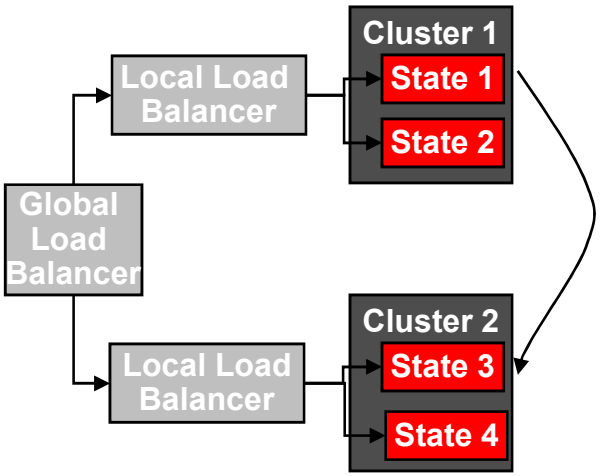
Hochverfügbarkeit mit WebLogic State-Replikation

Domain State Replication



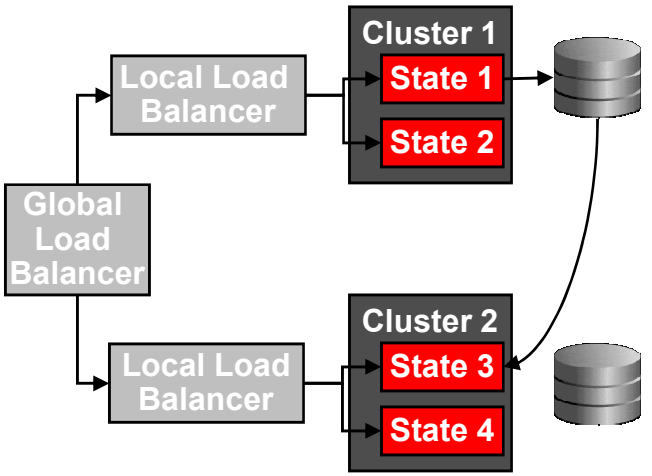
In Memory oder Datenbank asynchron, oder synchron verbunden

MAN State Replication



In Memory

WAN State Replication



Datenbank asynchron verbunden

A Server Fails, Messages are Trapped

Scenario

- Server is hosting JMS destinations; messages are persisted using the WebLogic Persistent Store (file or JDBC)
- Server fails; messages are trapped (messages are only available through the destination)

- Solutions:
  - Restart the server
  - Restart the server in another location (Whole Server Migration)
  - Restart the JMS service in another location (Service Migration)

Whole-Server Migration

General Idea

- Provides high availability for pinned services like JTA, JMS and custom singleton services within a cluster
- Automatic migration of failed servers within a cluster
  - Move server from one machine to another
  - Appears like a server restart on another machine
  - Requires Node Manager with IP migration support
  - Supported on Solaris, Linux and HP-UX
- Based on the notion of leasing – each clustered server instance needs a lease to run
- Servers periodically renew their lease against a lease table
- A single “cluster master” is determined. The cluster master grants leases and keeps track of the hosts that have those leases
- When a server loses its lease, the cluster master then restarts the server either on the same host or on a different host, depending on configuration and conditions

Whole Server Migration

Leasing options

- High-availability database leasing — requires a high-availability database to store leasing information.
- Non-database consensus leasing — stores the leasing information in-memory replicated in multiple cluster members.

Cluster Master – Not a Single Point of Failure

- Cluster Master is responsible for monitoring all servers' liveness

- What happens when the cluster master fails?
  - All servers compete to be the new cluster master
  - New cluster master is determined as the server that can write the cluster master record in the database (DB leasing) or with the earliest start time (consensus leasing)
  - Lease data is available either through replication in memory or in a database

Consensus Leasing

- Hierarchical leasing scheme where the cluster master is elected by majority consensus and gets a primary lease
  - The cluster master has the earliest start time
  - Other cluster members agree to that
- The cluster master grants sub-leases to other servers in the cluster
- Heartbeats are used to detect failures or loss of a lease, including the cluster master's lease
- The cluster master replicates the lease table to other members of the cluster on a best-effort basis.

HA Database Leasing

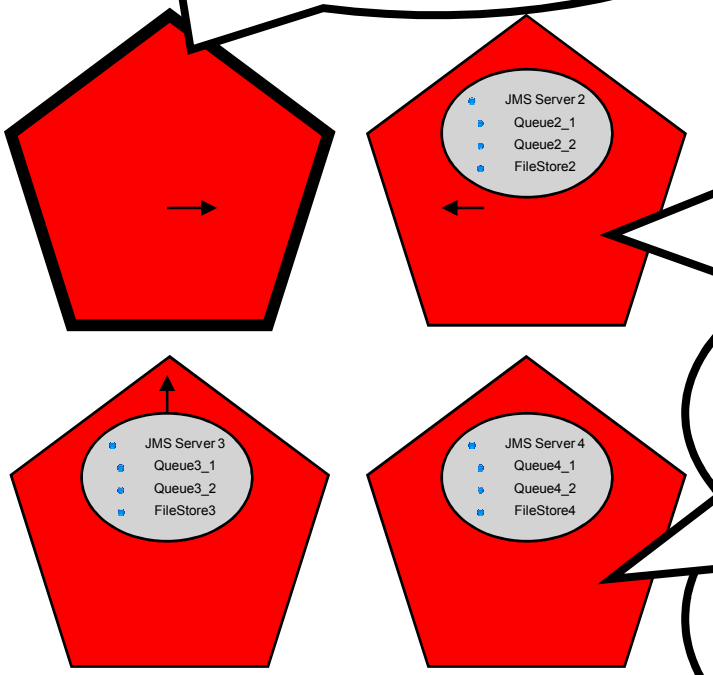
- Leasing table stored in a DB table
- Each server instance writes a record in the table as part of obtaining a lease.
- Each server competes to be the cluster master by trying to write the cluster master record in the table.
- Each server instance updates the record in the table on a periodic basis to renew the lease
- The cluster master checks the table on a periodic basis to make sure that leases are renewed
- If a lease is not renewed, the cluster master takes action on the failed server
  - Restart, if enabled
  - Migrate to another server

Service Migration

- Applies to services that run as singletons in a cluster:
  - JMS servers, their hosted destinations, and related services
  - JTA transaction recovery service for a server
  - User-defined singleton services
- Enables you to restart these services on another running server in the cluster:
  - For JMS, rescue stranded persistent messages
  - For JTA, process incomplete transactions for a failed server
  - For user-defined singleton services, guarantees that the process runs exactly once in the cluster (WLS automatically restarts it somewhere in the cluster)

How a[illegible]MS migration works

Servers in a cluster compete to be the cluster leader. The cluster leader grants leases to servers to host migratable targets



Other servers poll the lease master and compete for the lease to host a migratable target. The lease master grants a lease to the most appropriate host.

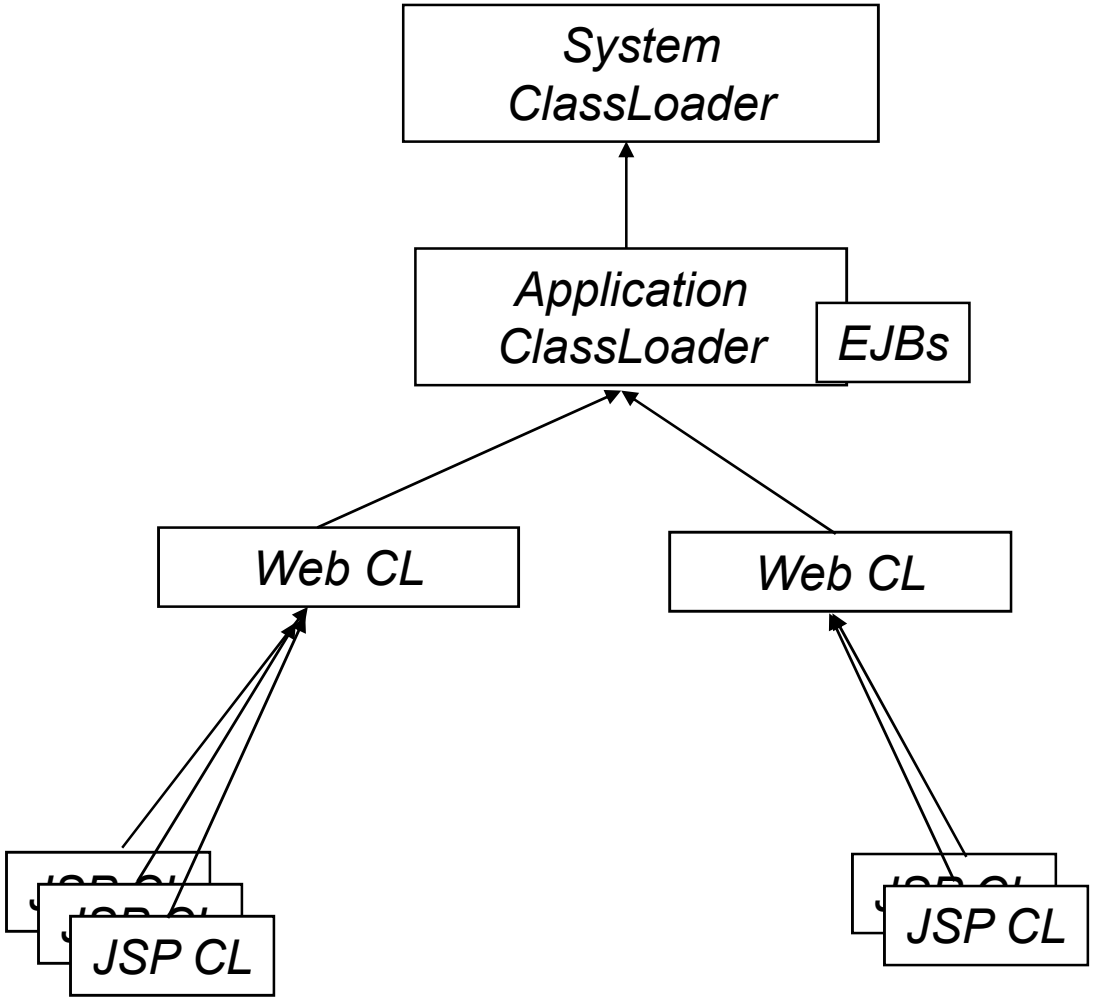
it's le[illegible]
migratable target.

M[illegible]
groups of deployed objects that migrate together as a unit

WLS Hot Redeployment

- Newer versions of application modules such as EJBs can be deployed while the server is running
- Web applications can be redeployed without redeploying the EJB tier
- The JSP class has its own classloader, which is a child of the Web application classloader. This allows JSPs to be individually reloaded.

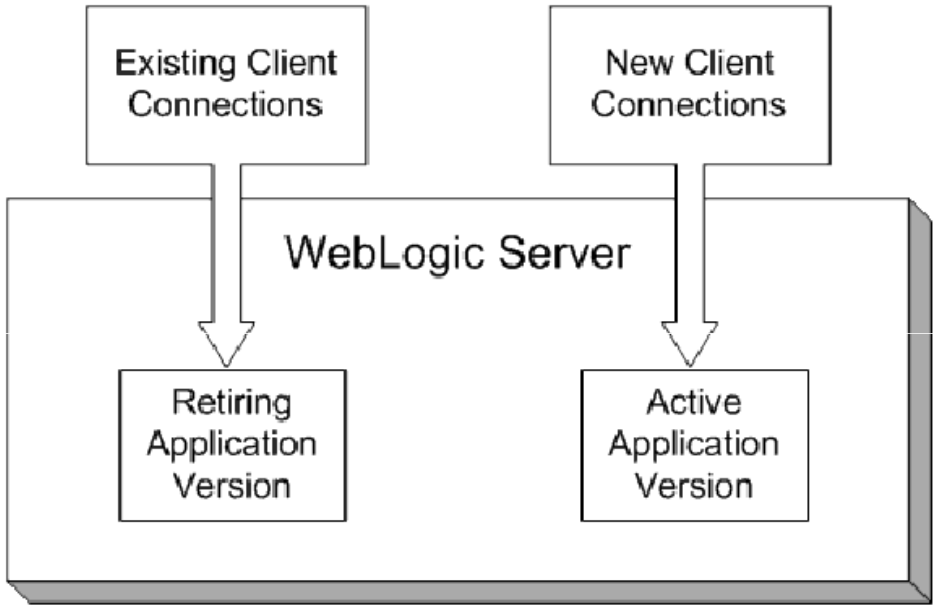
WLS Hot Redeployment
Class Loader Tree (w/o Filtering CL)



Production Redeployment

Side by Side Deployment

- Multiple application versions can co-exist
  - New client requests are routed to active version;
    Existing client requests can finish up with existing version
- Automatic Retirement Policy: Graceful, Timeout
- Test application version before opening up for business
- Rollback to previous application version
- Two versions of the application can be active at any given point of time



WebLogic Server Dynamic Updates

- Batch Updates
  - User obtains a configuration lock
  - Makes multiple config changes and deployments
  - Activates or rolls back changes
  - Previous configurations archived
- Configuration Deployment
  - Configuration changes ‘deployed’ to managed servers
  - Managed servers listen for dynamic settings
  - Static settings reflected on server restart
- Dynamic configuration settings
  - Take effect when changes activated
  - Approximately 1,400 dynamic configuration settings
  - Supports common tunables, channels, scalability, performance settings

WebLogic Server Rolling Upgrade

- Upgrades a running cluster with a patch, maintenance pack, or minor release without shutting down the entire cluster.
- During the rolling upgrade of a cluster, each server in the cluster is individually upgraded and restarted while the other servers in the cluster continue to host your application.
- You can also uninstall a patch, maintenance pack, or minor release in a rolling fashion.

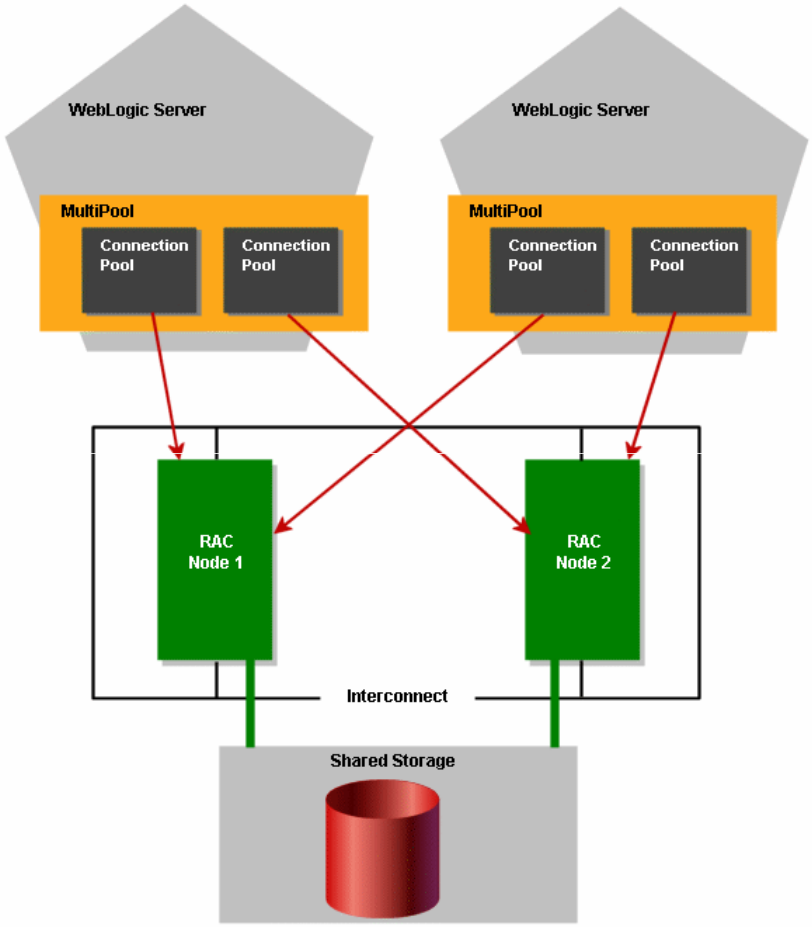
WebLogic Server Rolling Upgrade
Limitations

- Rolling upgrade applies only to upgrades within a product family. For example, you can upgrade from 9.x to 9.y but cannot upgrade from 9.x to 10.x.
- When WebLogic Server is installed on a machine and multiple Managed Servers are run from this same installation, you must shutdown all Managed Servers that use the same installation before you can upgrade.
- During the upgrade, you can use new features only after the entire domain has been upgraded. You should not make configuration changes during the upgrade process until all the servers in the cluster have been upgraded.
- For a minor release, during the rolling upgrade, there must be two entirely separate installation directories. That is, the location of the old installation and the location of the new installation must be two different directories.

WebLogic integrierte Verfügbarkeit

Oracle RAC Datenbank-Unterstützung

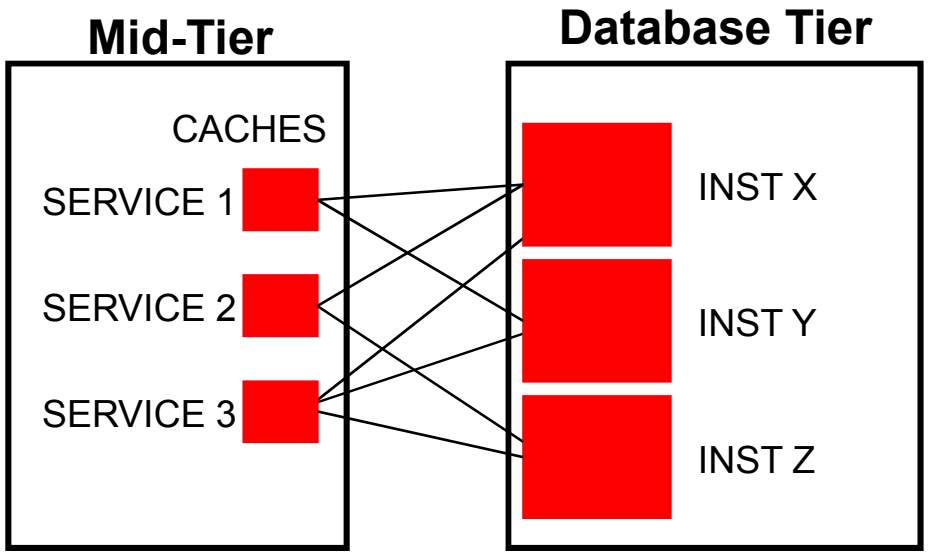
- Schnelle Cluster-Knoten-Übernahme
- Automatische Rück-Übernahme von Cluster-Knoten
- Load Balancing oder Hochverfügbarkeit ist optional konfigurierbar
- Übernahme Anfrage der Knoten
- Periodische Zustandsprüfung
- Gebundene Transaktionen
- Support für schnelle Knoten-Verbindungs-Übernahme



Fast Connection Failover

Best Practice for Application Connections

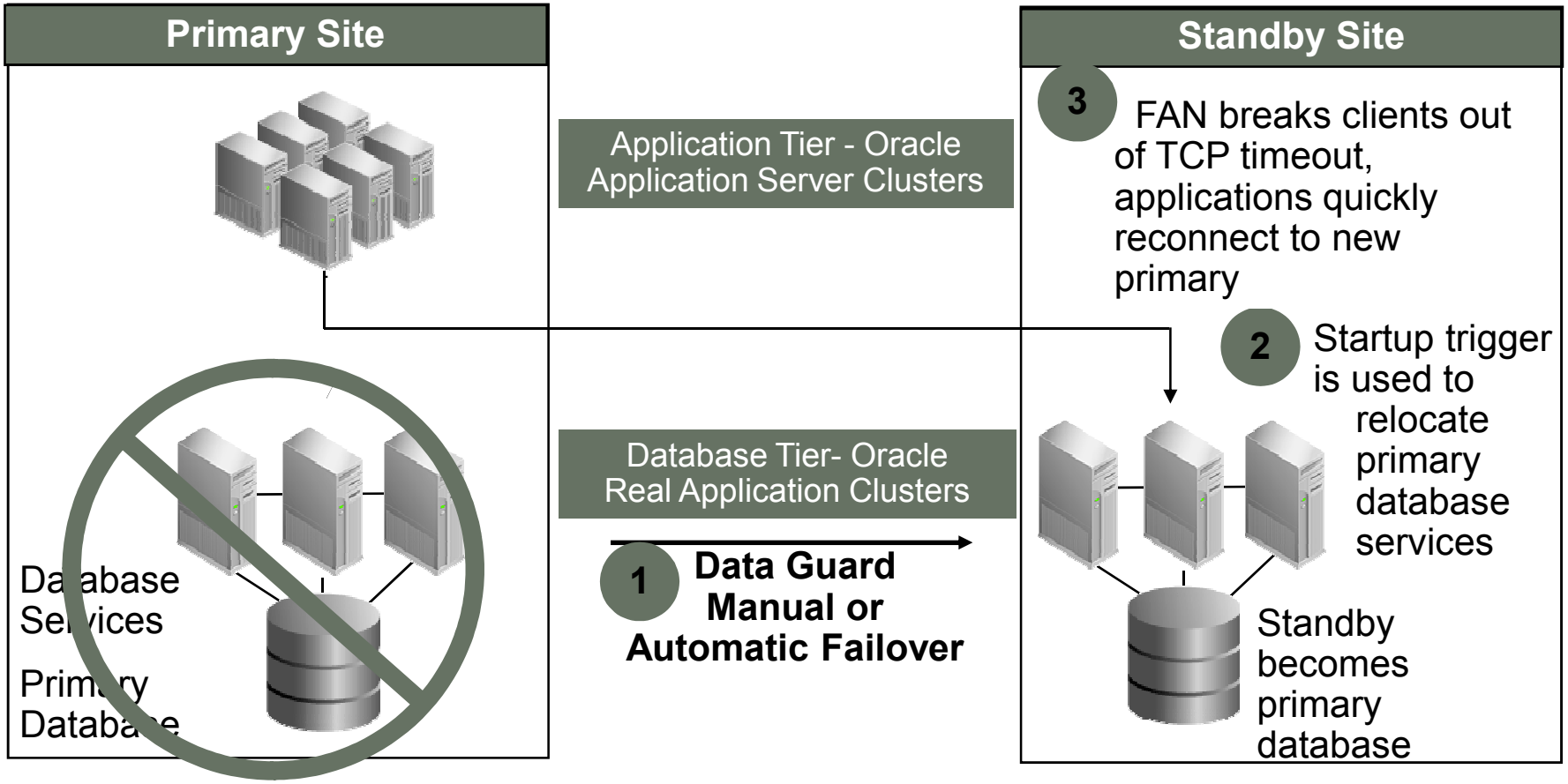
- Supports multiple connection caches
- Datasource for each cache mapped to a cluster managed service
- Keeps track of service and instance for each connection
- Cleans up connections when failures occur
- Distributes new work requests across available instances
- Applications can mask failures from the end user by retrying connections after failure.



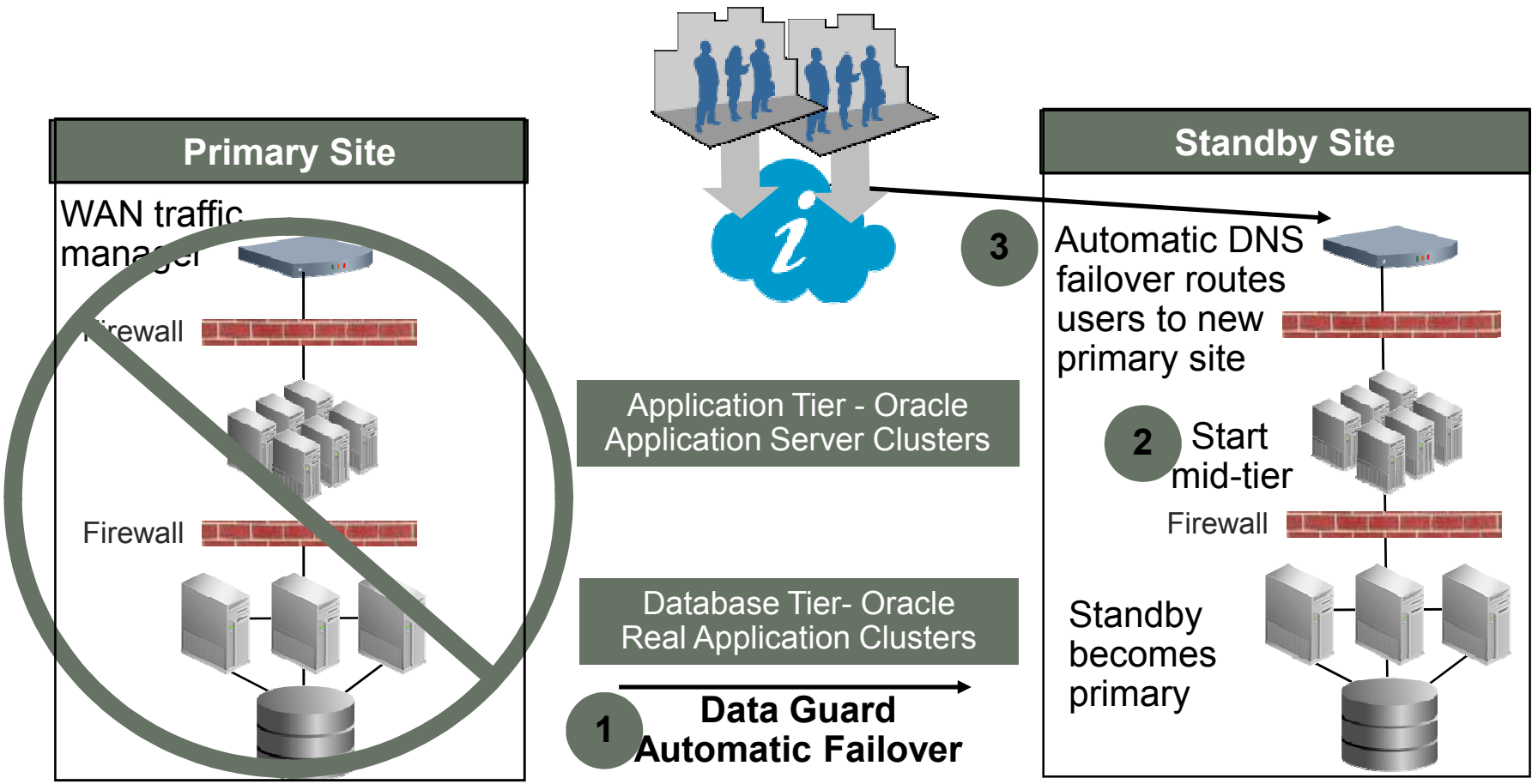
JDBC, OCI

Application Failover: Database & App Server

Primary site and application tier still viable



Complete Site Failure: Database & App Server



Zusammenfassung: Hochverfügbarkeit mit WebLogic Cluster

- Bauen Sie stabile Geschäftsanwendungen (Produktivität)
  - Entwickler schätzen WebLogic wegen der vereinfachten Produktentwicklung
  - Profitieren Sie vom Eclipse-based Community Tooling – durch integrierte Eclipse Plugins
- Bauen Sie zuverlässige Geschäftsanwendungen (24x7 Verfügbarkeit)
  - Reliability, Availability, Scalability & Performance
  - Hohe Verfügbarkeit auch ohne Hardware-Unterstützung
  - Siehe: Forrester 'Cost of Reliability' Tool Whitepaper, Artikel, Blogs, Dokumentation
  - Online – www.oracle.com/appserver
- Bauen Sie etwas Neues auf diesem Fundament (Innovation)
  - Informieren Sie sich über unsere Kunden, die mit neuen Ideen aus dem IT Bereich eine innovative Geschäftsentwicklung eingeleitet haben
  - Verwenden Sie WebLogic um eine IT zu realisieren die ihr Geschäft unterstützt
- Holen Sie das Maximale aus Ihrer IT Infrastruktur (Kosteneffiktivität)
  - Profitieren Sie mit WebLogic durch optimierte Hardwareauslastung and Performance

Vielen Dank für Ihre Aufmerksamkeit!

Wolfgang.Weigend@oracle.com