

Siemens PLM Connection

High Availability of Teamcenter Enterprise

Mark Ludwig

2008

Siemens
PLM Connection



Americas 2008

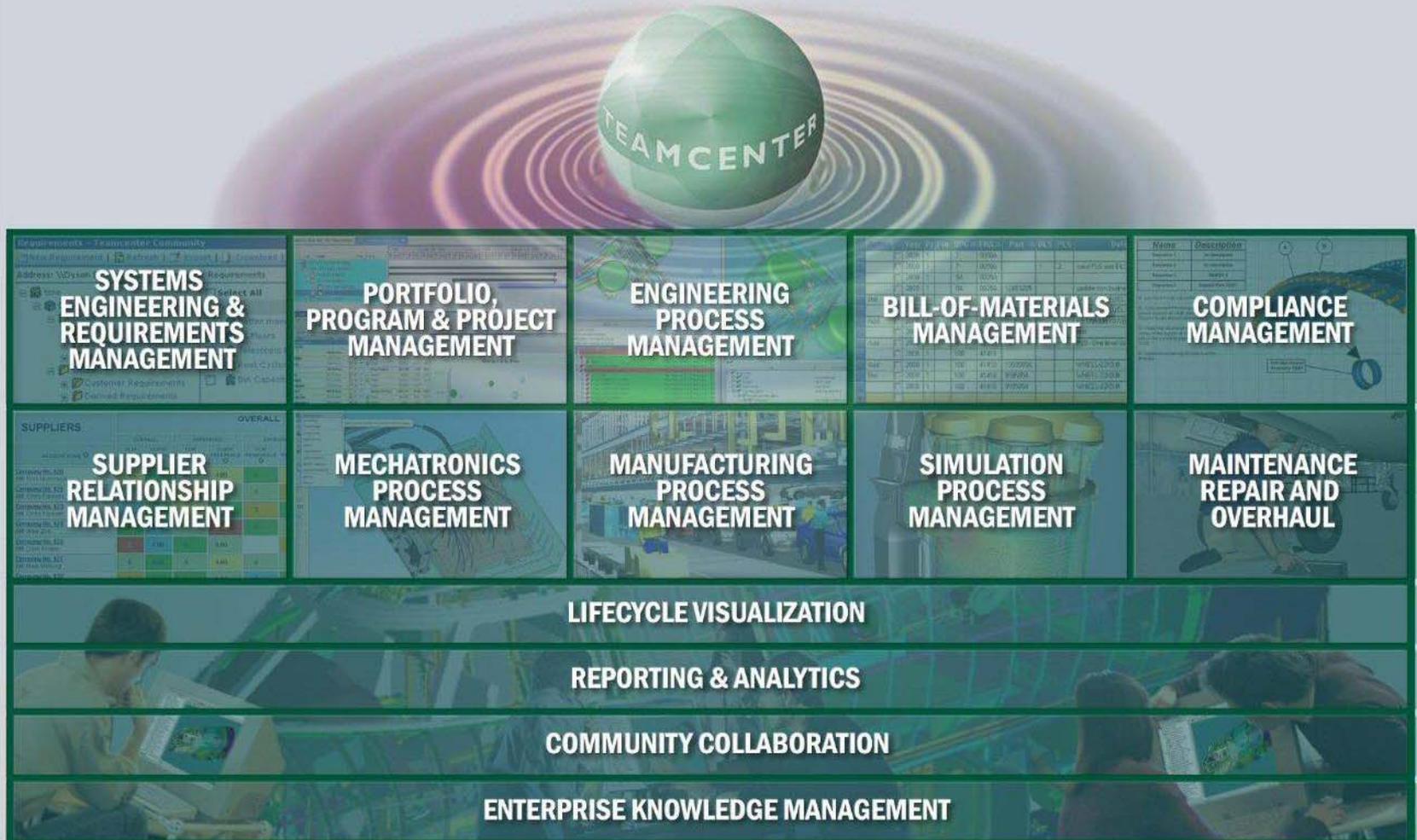
PLM Software

Answers for industry.

SIEMENS

Teamcenter Digital Lifecycle Management Solutions

SIEMENS



Copyright © Siemens PLM Software Inc. 2008. All rights reserved.

Enterprise Knowledge Management

Agenda

Introduction to High Availability

- Defined by *Wikipedia*
- Applied to Teamcenter

Load Balancing Technology

- Hardware versus Software

High Availability Web Tier

High Availability Enterprise Tier

Summary



Introduction to High Availability

What is High Availability?

- Availability
- Downtime
- Continuous Availability

High Availability Clusters

- Node Configurations
- Three-Tier, Two-Node Example

High Availability Components

- Clustering and Computing Clusters
- Load Balancing and Failover

High Availability on the Teamcenter Reference 4-Tier Architecture

Terminology

- **High availability** is a system design protocol and associated implementation that ensures a certain absolute degree of operational continuity during a given measurement period.
- **Availability** refers to the ability of the user community to access the system.... If a user cannot access the system, it is said to be **unavailable**. Generally, the term **downtime** is used to refer to periods when a system is unavailable. A distinction needs to be made between **planned** downtime and **unplanned** downtime.

Terminology

- Many computing sites exclude planned downtime from availability calculations ... [so they] can claim to have phenomenally high availability.... Systems that exhibit truly **continuous availability** are comparatively rare and higher priced, and ... have carefully implemented specialty designs that eliminate any single point of failure....

Terminology

- **High-availability clusters** (also known as **HA Clusters** or **Failover Clusters**) are computer clusters that are implemented primarily for the purpose of improving the availability of services which the cluster provides. They operate by having redundant computers or **nodes** which are then used to provide service when system components fail. Normally, if a server with a particular application crashes, the application will be unavailable until someone fixes the crashed server. HA clustering remedies this situation by detecting hardware/software faults, and immediately restarting the application on another system without requiring administrative intervention.

Terminology

- The most common size for an HA cluster is two nodes, since that's the minimum required to provide redundancy, but many clusters consist of many more, sometimes dozens of nodes. Such configurations can sometimes be categorized into one of the following models:
 - Active/Active
 - Active/Passive
 - N+1
 - N+M
 - N-to-1
 - N-to-N

Most Common Terms

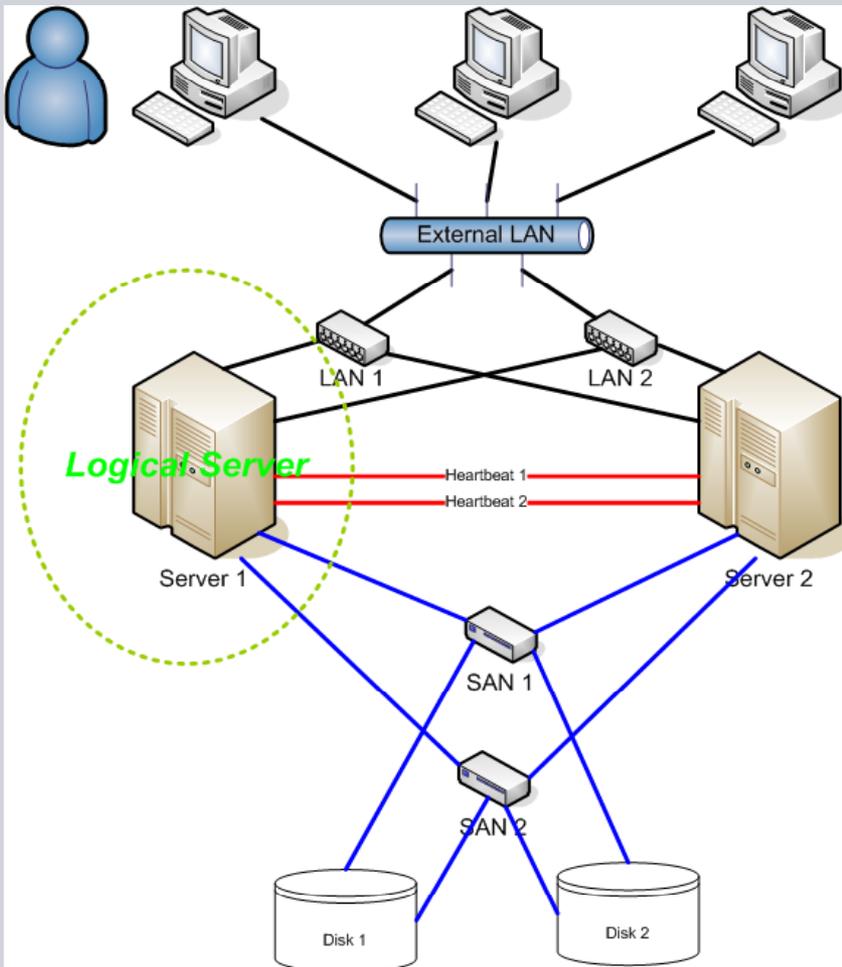
- Active/Active

Traffic intended for the failed node is either passed onto an existing node or load balanced across the remaining nodes. This is usually only possible when the nodes utilize an homogeneous software configuration.

- Active/Passive

Provides a fully redundant instance of each node, which is only brought online when its associated primary node fails. This configuration typically requires the most amount of extra hardware.

Example of High Availability Cluster *Wikipedia*



Two-Node, Three-Tier, Active/Active Layout

- Client
- Server
- Resources

Two instances of each component

- No single point of failure

The two servers manage failover themselves via **heartbeat signals** between them

Less Common Terms

- N+1 – Provides a single extra node that is brought online to take over the role of the node that has failed. In the case of heterogeneous software configuration on each primary node, the extra node must be universally capable of assuming any of the roles of the primary nodes it is responsible for. This normally refers to clusters which have multiple services running simultaneously; in the single service case, this degenerates to Active/Passive.
- N+M – In cases where a single cluster is managing many services, having only one dedicated failover node may not offer sufficient redundancy. In such cases, more than one (M) standby servers are included and available. The number of standby servers is a tradeoff between cost and reliability requirements.

Less Common Terms

- N-to-1 – Allows the failover standby node to become the active one temporarily, until the original node can be restored or brought back online, at which point the services or instances must be failed-back to it in order to restore High Availability.
- N-to-N – A combination of Active/Active and N+M clusters, N to N clusters redistribute the services or instances from the failed node among the remaining active nodes, thus eliminating (as with Active/Active) the need for a 'standby' node, but introducing a need for extra capacity on all active nodes.

Overview of Load Balancing and Failover

Load Balancers

- Load Balancing is performed by Load Balancers
- Load balancers act as logical service contact points for all clients of that service
- Load balancers receive client requests and direct each to one active, functional server node

Overview of Load Balancing and Failover

Types of Load Balancers

- A “hardware load balancer,” is actually a combined hardware/software system
- Hardware load balancers can generally handle high loads and all networking protocols
- “Software load balancers” are less capable – usually only supporting a limited set of networking protocols, and lacking redundancy, can be single points of failure, so are not generally part of a high availability setup
- The best load balancers are internally redundant to avoid becoming single points of failure in an overall high availability deployment

Overview of Load Balancing and Failover

Failover

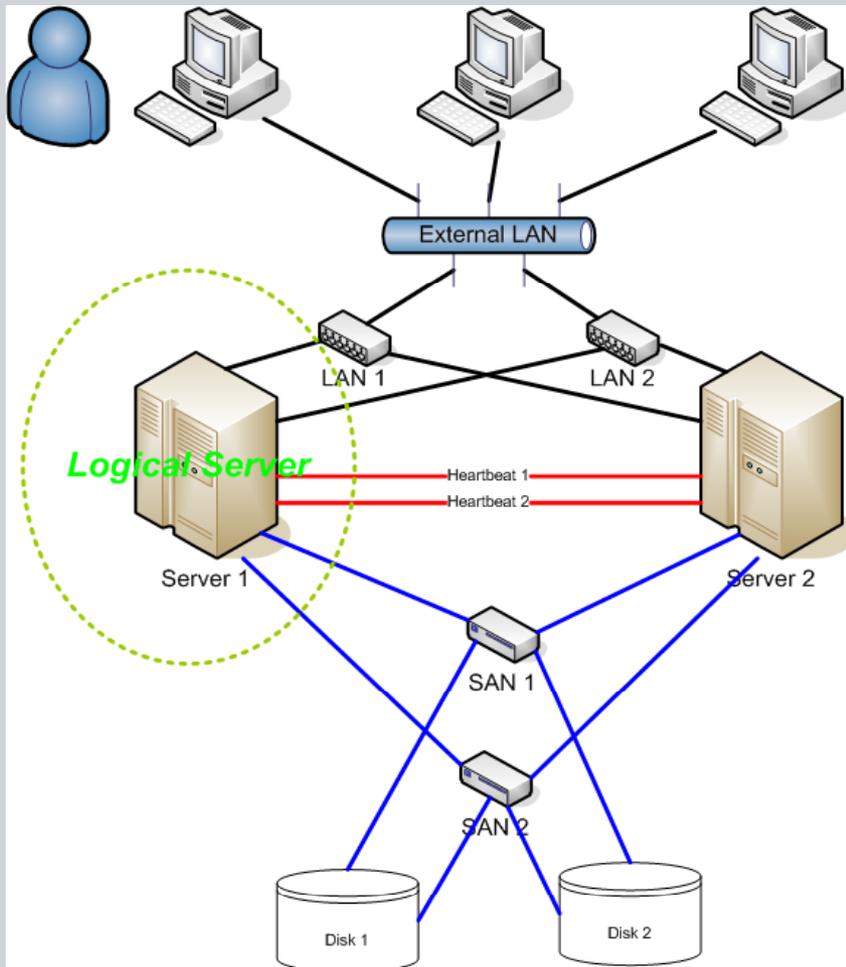
- Failover conceptually applies at multiple levels, from individual components to user sessions
- An individual hardware or software component of a cluster may fail
 - If the system has redundancy for that component, the failure causes failover of whatever service that component provides
 - For example, in an Active/Passive configuration, the system may bring an idle replacement component on line
- At the highest level, user sessions may failover
 - This concept is covered later in this presentation in the section that applies these concepts to the Teamcenter Enterprise Web Tier

Overview of Load Balancing and Failover

Involvement of Load Balancers in Failover

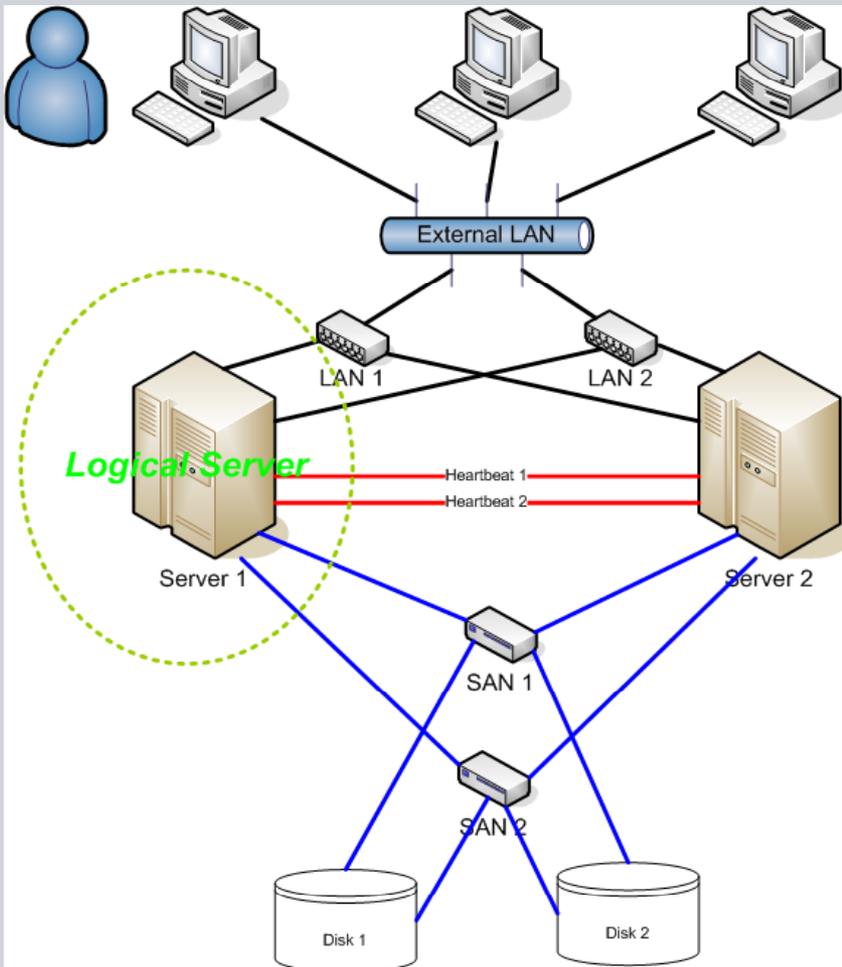
- Load balancers must monitor the health of individual nodes and send each request to a functional node
- For stateless transactional systems, load balancing each request to a functional node is sufficient to provide high availability; no failover is necessary
 - Consider a request to retrieve the contents of a static web page
 - Any functional web server in a cluster can satisfy that request
 - Simply sending the request to any functional web server satisfies the client, so the cluster can provide high availability that way
- A load balancer may send a request to a web server that may stop functioning before servicing that request, so the request may fail and require retry to reach a functional node to satisfy the request

Example of High Availability Cluster *Wikipedia*



Where is the Load Balancer?

Example of High Availability Cluster *Wikipedia*

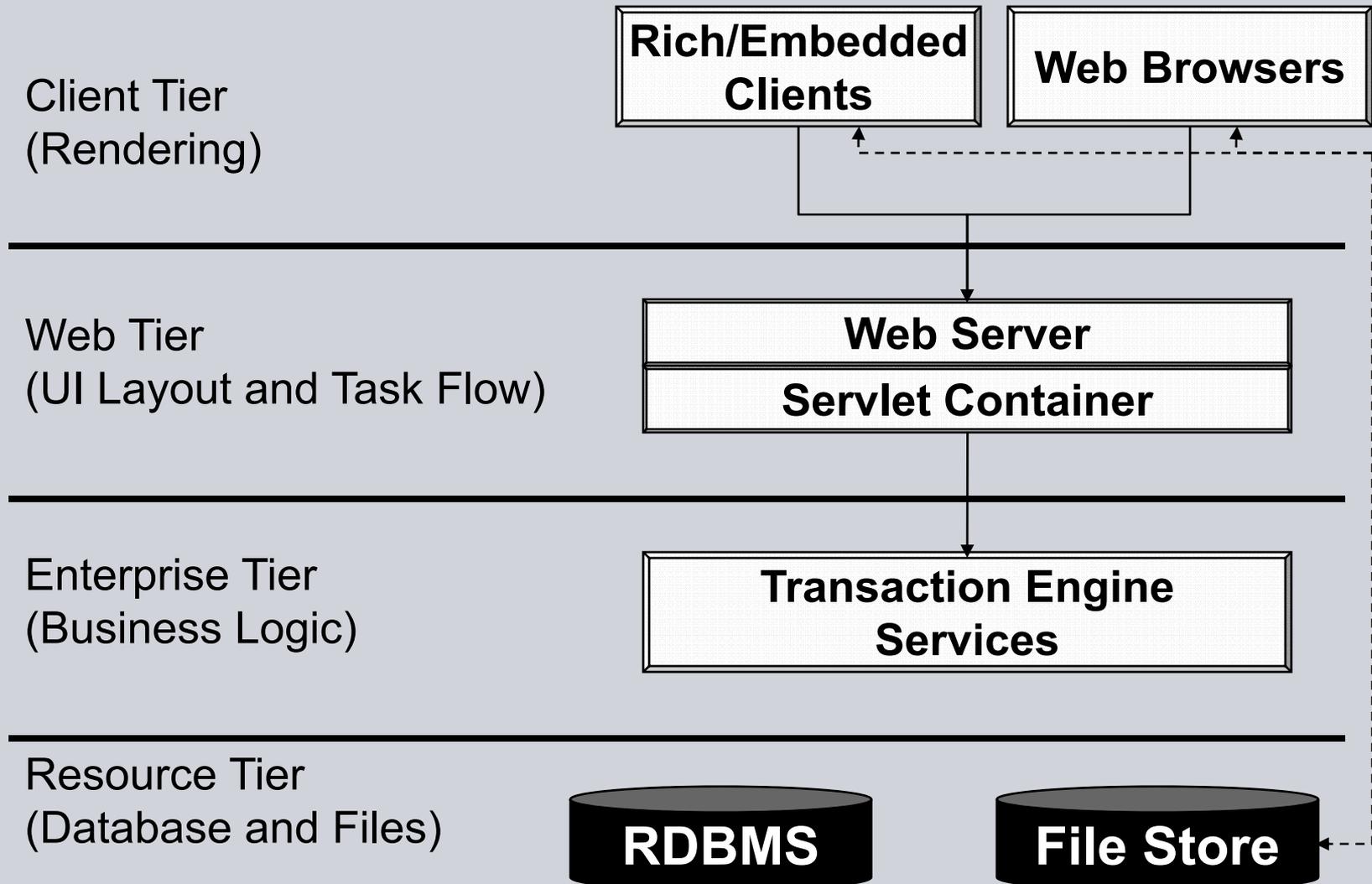


Where is the Load Balancer?

It is truly absent, but must be part of the “External LAN”

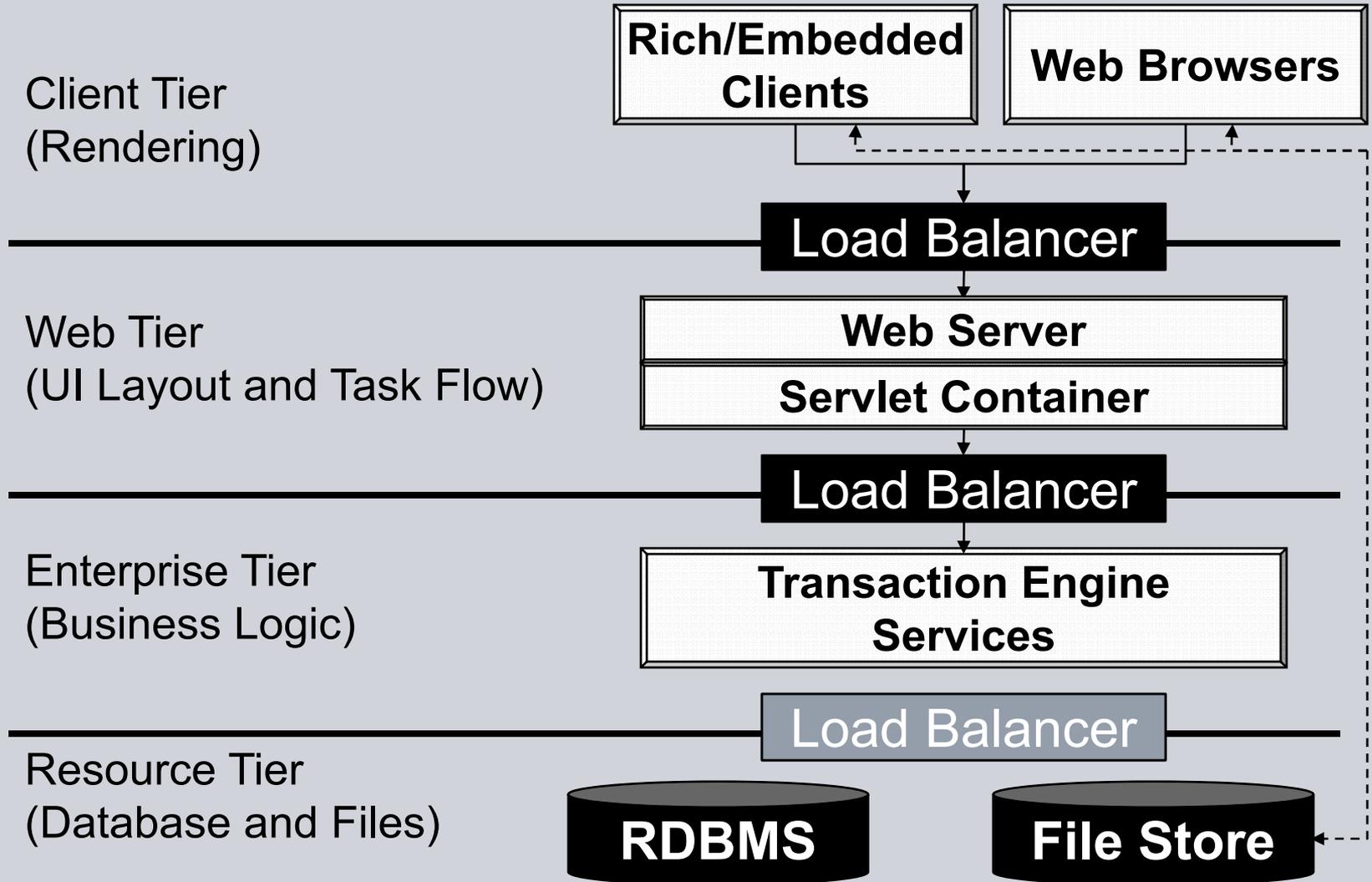
- Some assistance from “LAN 1” and “LAN 2”

Teamcenter 4-Tier Architecture Standard Reference



Teamcenter 4-Tier Architecture

High Availability and Load Balancing Services



Introduction to High Availability

Questions?



Load Balancing Technology

Responsibilities of a Load Balancer

- Fundamentals of Load Balancing
- Load Balancer Involvement in Failover

Teamcenter Tiers of High Availability

Customer Deployments

Commercial Load Balancers

Responsibilities of a Load Balancer

Fundamentals of Load Balancing

- A load balancer at the TCP level needs to forward the network packets to the target server
- At the HTTP level, it needs to read the cookies on each request made by the client
 - Based on this information, it can send the request to the appropriate node in the cluster

Responsibilities of a Load Balancer

Fundamentals of Load Balancing

- For best performance, a load balancer should provide server affinity in HTTP communication to maintain the user's session in that web application server
 - This is more challenging through a secure channel, such as HTTPS
 - In a secure channel, the messages are SSL-encrypted, and this prevents the load balancer from reading the session information
- Some load balancers act as proxies, so they are able to decrypt the request
 - Depending on the configuration, they may forward the HTTP request in the clear or encrypt it again for transport to the web server.

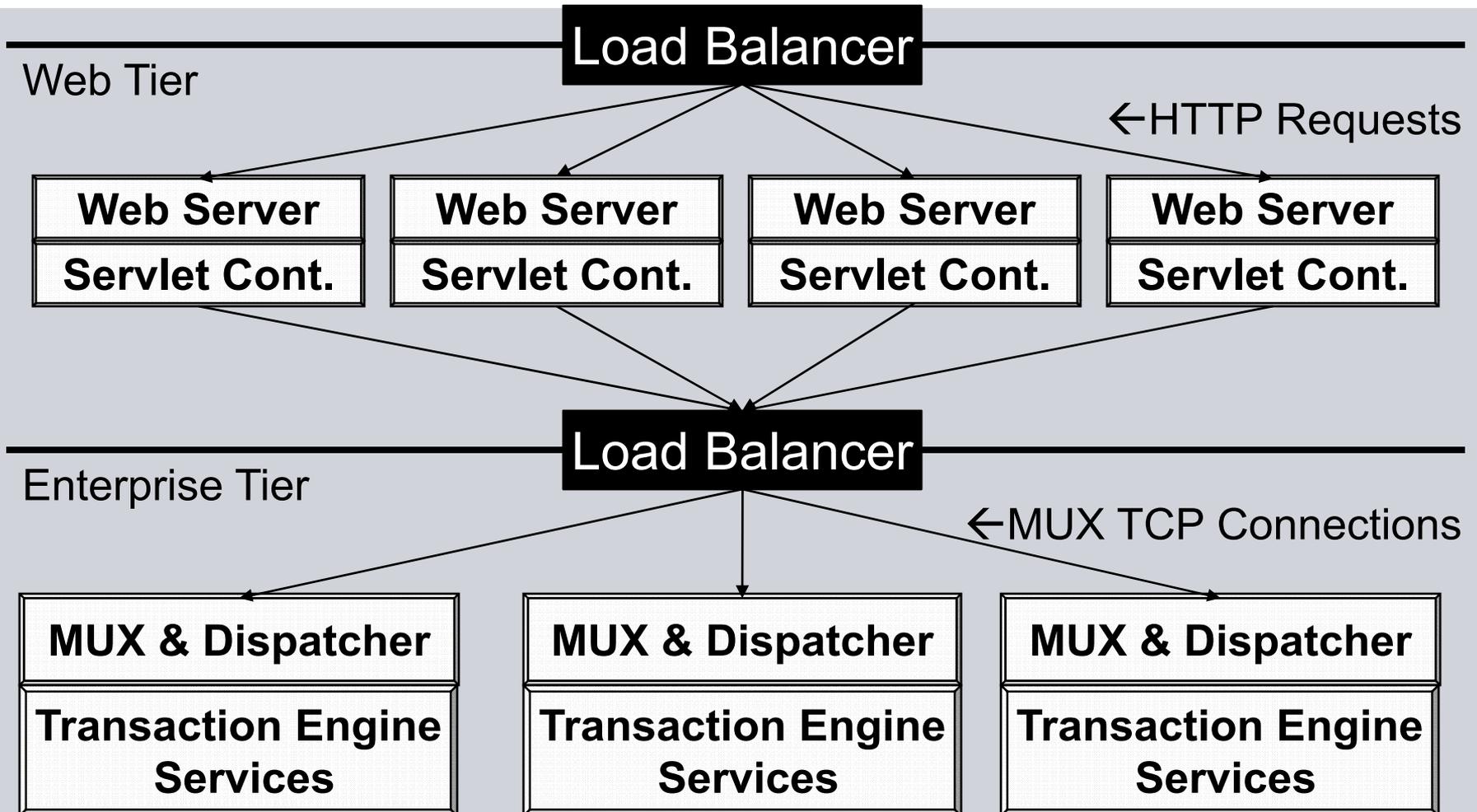
Responsibilities of a Load Balancer

Load Balancer Involvement in Failover

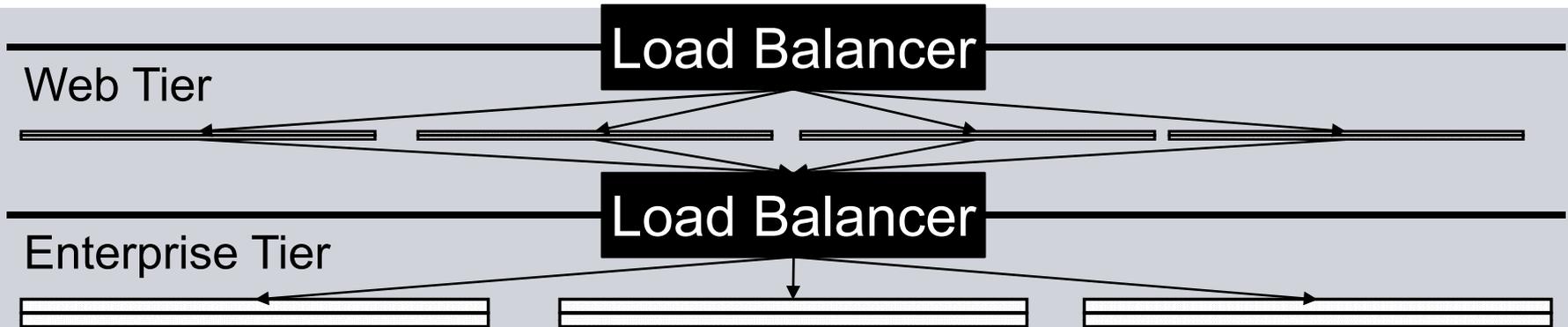
- A load balancer must monitor the health of all of the individual nodes that provide the logical service that it represents
- It must not send the initial request for a session to an unresponsive node
- To provide best performance by maintaining “sticky sessions,” when a load balancer receives a request for a session that was on a node that is not responding, the load balancer must send the request to another node that is responding

Teamcenter Tiers of High Availability

Load Balancing Web Tier and Enterprise Tier



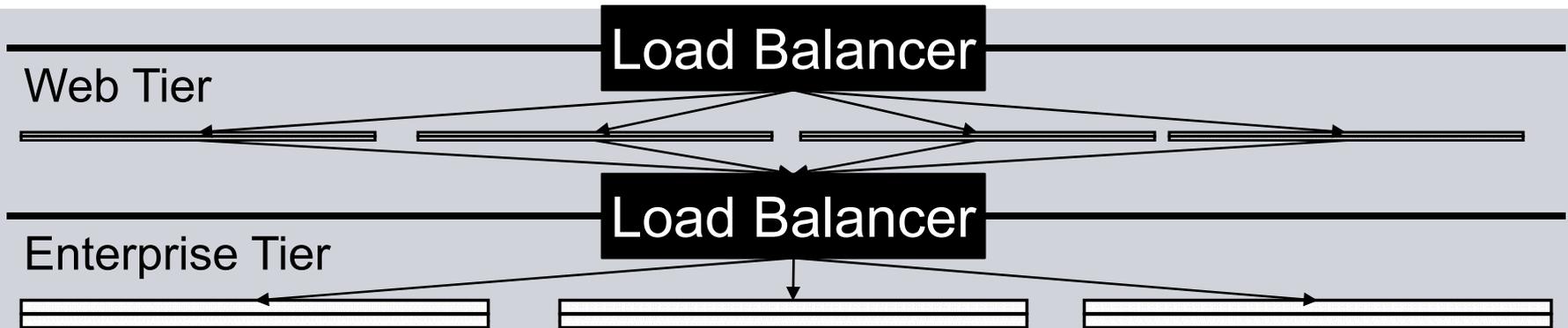
Teamcenter Tiers of High Availability Load Balancing Web Tier and Enterprise Tier



Which Clustering Model is this?

Teamcenter Tiers of High Availability

Load Balancing Web Tier and Enterprise Tier

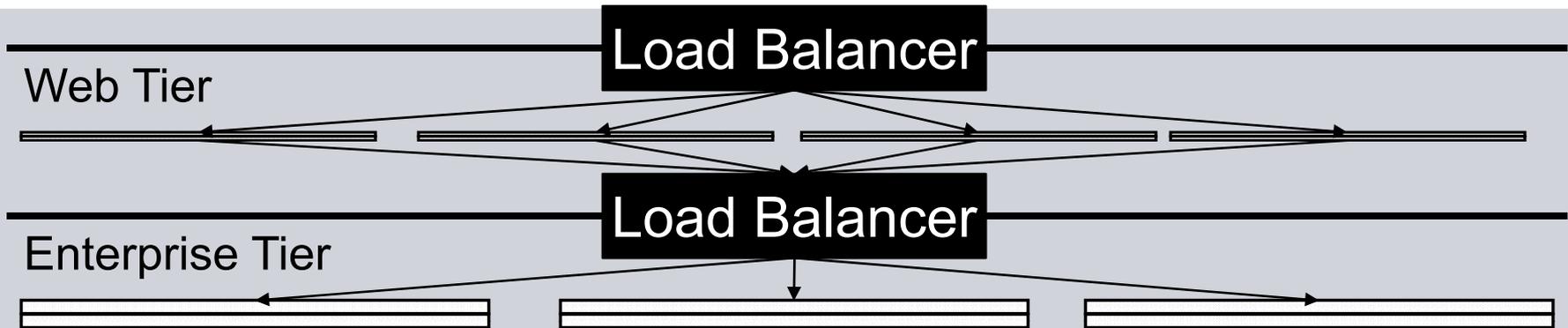


Which Clustering Model is this?

- N-to-N – A combination of Active/Active and N+M clusters, N to N clusters redistribute the services or instances from the failed node among the remaining active nodes, thus eliminating (as with Active/Active) the need for a 'standby' node, but introducing a need for extra capacity on all active nodes

Teamcenter Tiers of High Availability

Load Balancing Web Tier and Enterprise Tier



General Notes on Load Balancers for Teamcenter Enterprise

- As depicted, they look like single points of failure, but are logical entities that can be internally redundant
- Different responsibilities for the Web Tier and Enterprise Tier
 - HTTP **requests** versus
 - MUX TCP **connections**
- Different numbers of “stacks” shown in Web & Enterprise Tiers shows independent load balancing
 - Permits horizontal scalability

Customer Deployments

Customer	Load Balancer	Vendor	Notes
1	Redline	Juniper Networks, Inc.	No longer available (newer products are)
2	ACE (Application Control Engine)	Cisco Systems, Inc.	One device serves three tiers
3	BIG-IP	F5 Networks, Inc.	One device serves two tiers
4	BIG-IP	F5 Networks, Inc.	One device serves three tiers

Commercial Load Balancers

BIG-IP (*F5 Networks, Inc.*)

- Firewall
- Load Balancer
- Combined Hardware/Software System (UNIX is the Operating System)
- Serves largest Web Farms (most expensive)

Redline Appliance (*Juniper Networks, Inc.*)

- Firewall
- Load Balancer
- Redline product line no longer available
 - Newer products available

Application Control Engine (ACE) (*Cisco Systems, Inc.*)

- Firewall
- Load Balancer

Load Balancing Technology

Questions?



High Availability Web Tier

Clustering

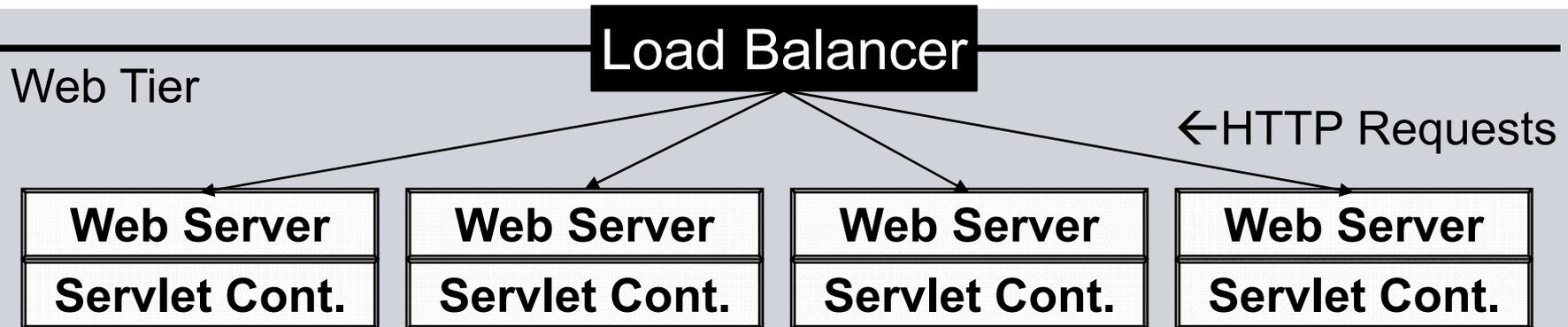
Load Balancing and Failover

Hardware and Software Platforms

Customer Deployments

Constraints

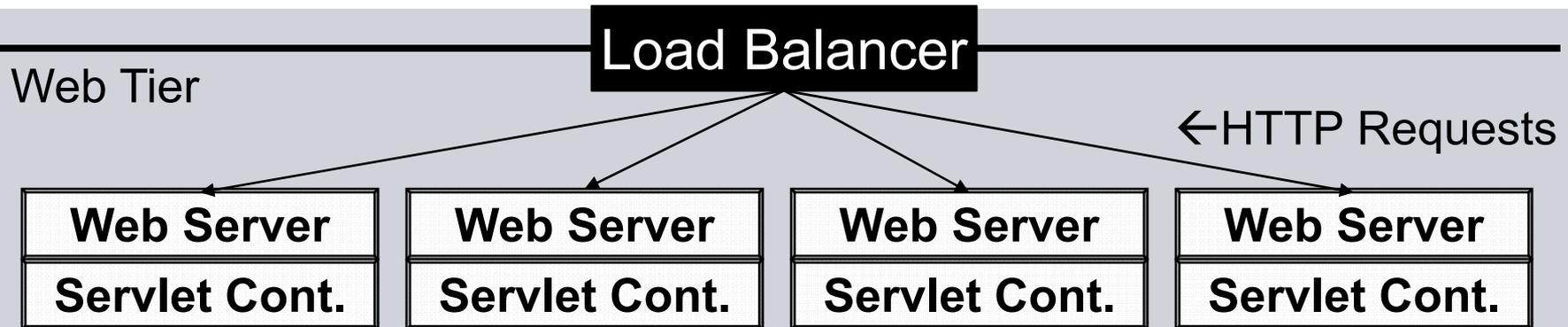
High Availability Web Tier Clustering



Details

- The Load Balancer in front of the Web Tier needs to support HTTP connections from all web clients
- All web clients must use URL(s) that send the HTTP requests to the logical service address supported by the load balancer
- Programmatic web clients accessing web services could use the same URL(s) as interactive web browsers
 - A more advanced configuration might have separate URLs for classes of clients (classes of users, certain programs, ...) to provide different levels of service

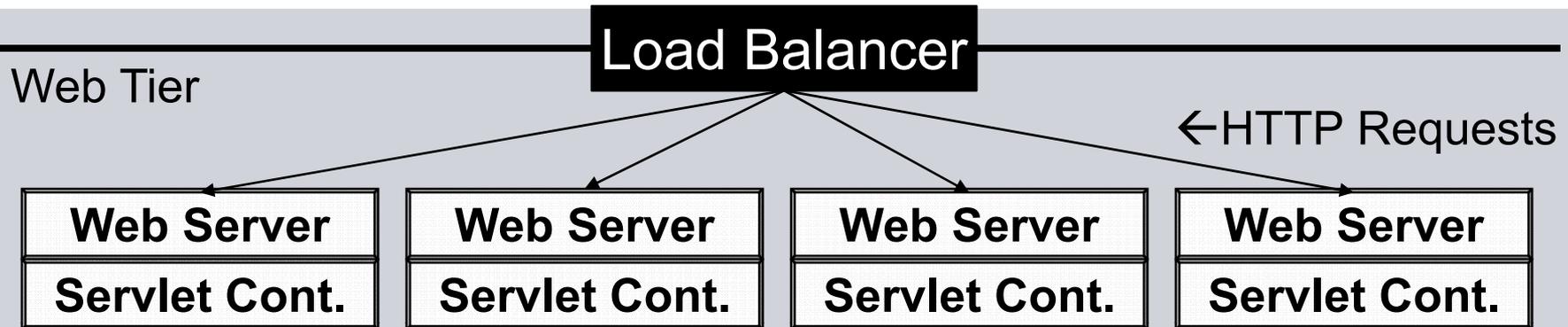
High Availability Web Tier Clustering



Details

- The load balancer **may not** use HTTP Redirect (status codes 301 or 302), because the HTTP 1.1 standard (RFC 2068) **requires** user agents (web browsers) to **interactively** redirect POST requests
- Most Teamcenter Enterprise web requests use POST
 - Carry hidden fields that maintain context across pages
- Generally prohibits use of Apache's **mod_rewrite** software load balancer, because it relies on redirection
- See RFC 2068 sections 10.3, 10.3.2 and 10.3.3

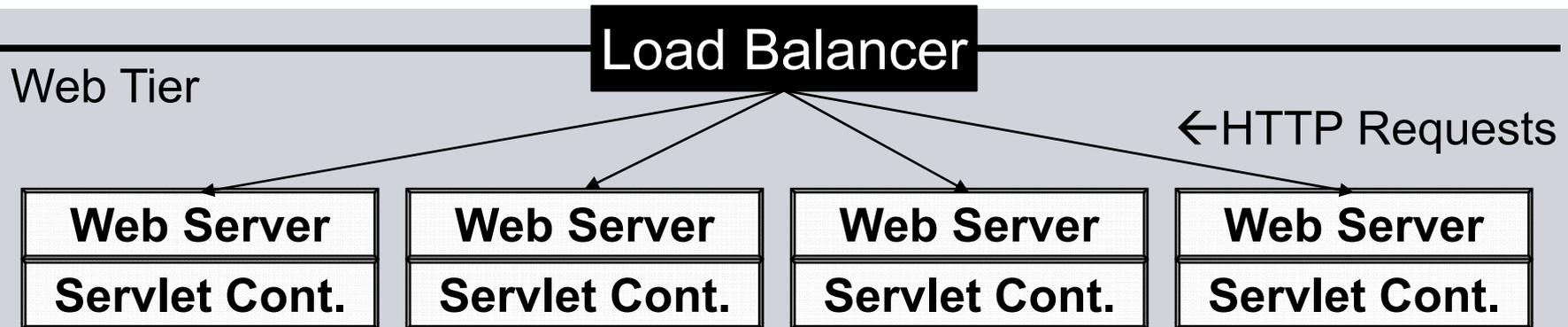
High Availability Web Tier Clustering



Details

- Web Application Servers operate as a Cluster
 - Load balancer directs TCP connections for HTTP requests
 - Sufficient hardware → no single point of failure
 - As many software nodes as necessary for load – allows full horizontal scalability
 - Web Tier WAR files must all have the same MUXHost and MUXPort
 - Point to the Enterprise Tier Load Balancer

High Availability Web Tier Clustering

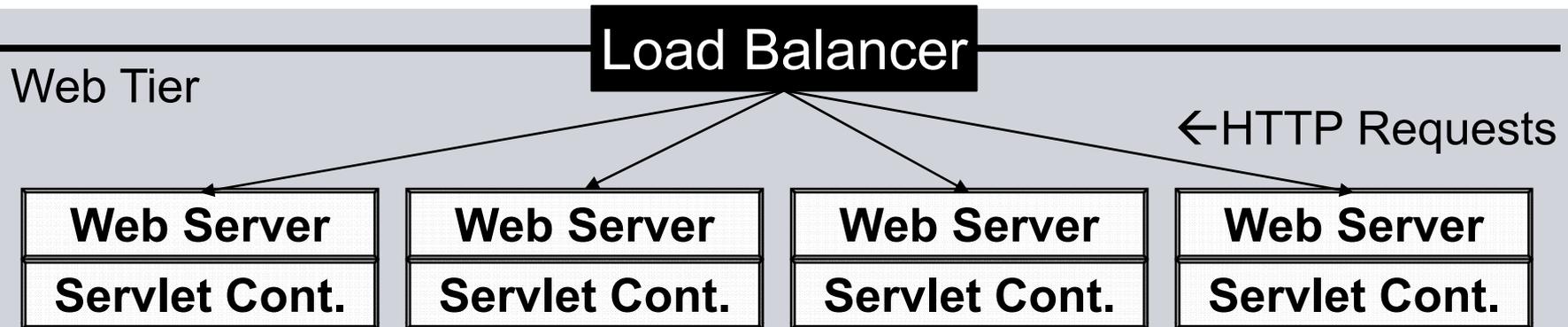


Details

- Web Application Servers operate as a Cluster
 - Identical MUXHost and MUXPort values are required across a Web Tier cluster (i.e., in all WAR files) for proper failover behavior
- Siemens made a technological investment to conform to the Servlet Specification restrictions for **Distributed Sessions**
 - Allows Servlet Container to move user session state among nodes in a cluster without any specific Siemens code involvement

High Availability Web Tier

Load Balancing and Failover



Best Performing Configuration

- Expect a fresh TCP connection for each HTTP request
- Redirect the initial connection for a session
 - Simple round-robin should be satisfactory
 - Generally the user community is large enough to have sufficient diversity so that requests naturally distribute the load
 - No need to sense loads on, or performance of, the individual nodes of the cluster
- Redirect subsequent connections for an established session
 - Be “sticky” – use server chosen for initial connection for session

User Experience in Failover

- Failover occurs when a component failed
- User's experience varies drastically: was the system acting on a user's request at the time of the failure?
 - Yes: user will see an error screen
 - Expect user to use Back button and retry the action, which will usually succeed
 - No: user may have no idea that anything failed
- Activation of the user's session in the newly-chosen node may take a noticeable amount of time, so the user may notice a delay in getting a response from the first request after failover occurs

Servlet Specification defines a Distributable web application

- One WAR file deployed on multiple nodes
- Session holds user context/state information
- Cluster technology manages the node on which any particular user session is “active”
 - Keeping a session “sticky” to a node reduces the need to move a session to make it “active” on new node
 - Data in a session varies with user activity, but can be large enough to take a noticeable amount of time to copy
 - Best performance by minimizing the size of a session

Servlet Specification: Distributable Web Application

- Configuring a Distributable Web Application means the **deployment descriptor** must include the **distributable** element
 - Informs the Servlet Container that the application allows **sessions** to move among different nodes of a cluster
- Must not mark all web applications as **distributable**
 - Cannot assume all Web Tier code will function, because of restrictions for Distributable sessions
 - Replicating session data has a cost – needless if not truly distributable

Servlet Specification: Details of distributable tag

- The **distributable** tag is empty and is placed between the **description** and **context-param** tags:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE ...>
<web-app ...>
  <display-name>Test Web Application</display-name>
  <description>This is a test web application.</description>
  <b>distributable</b>
  <context-param> ... </context-param>
</web-app>
```

- The Teamcenter Enterprise Web Tier installation utility **insweb** optionally generates a distributable tag in the deployment descriptor

Static versus Dynamic Content

- Teamcenter Enterprise Web Applications use a small amount of static content: icons and color bars on the pages
- Vast majority of Teamcenter Enterprise Web Applications' content is dynamic
 - Main focus of high availability effort
- Load balancing and failover of static content is a very different thing with different considerations, different ways to tune it
 - Akamai, et al.

Web Application Servers – General Requirements

- Web Application Servers must support clustering for good user experience during failover
 - “Good user experience” means the user’s session continues
 - No login required after failover
 - Users’ task flows continue
- Clustering is generally supported by high-end editions, and generally not supported by “express” editions
 - WebSphere Application Server only supports clusters with **distributed sessions** with the Network Deployment option
 - WebLogic Server supports clustering
 - WebLogic Express does not

High Availability Web Tier Customer Deployments



Customer	Web Application Server (Web Container)
1	WebLogic
2	WebLogic
3	WebSphere
4	WebSphere

High Availability Web Tier Constraints



None!

High Availability Web Tier

Questions?



High Availability Enterprise Tier

Clustering

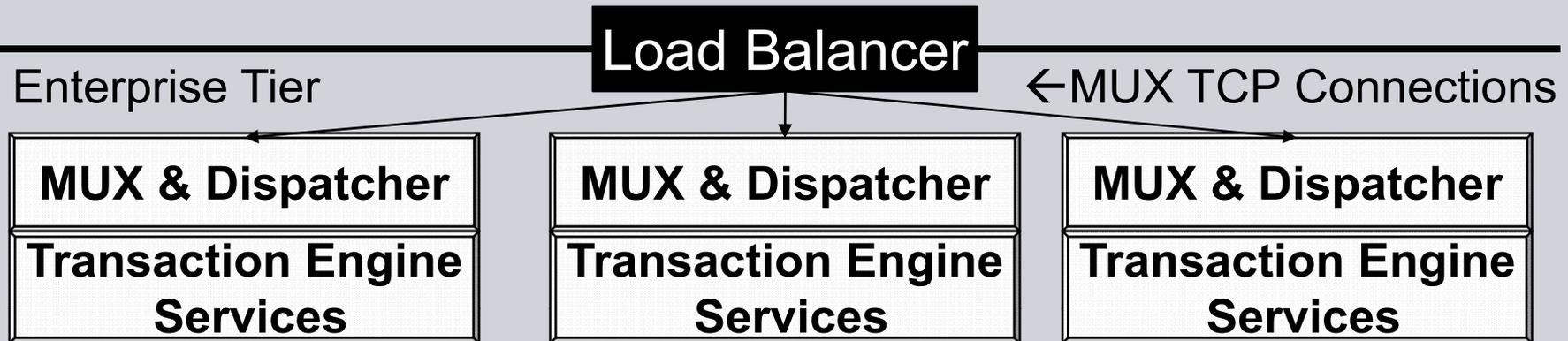
Load Balancing and Failover

Hardware and Software Platforms

Customer Deployments

Constraints

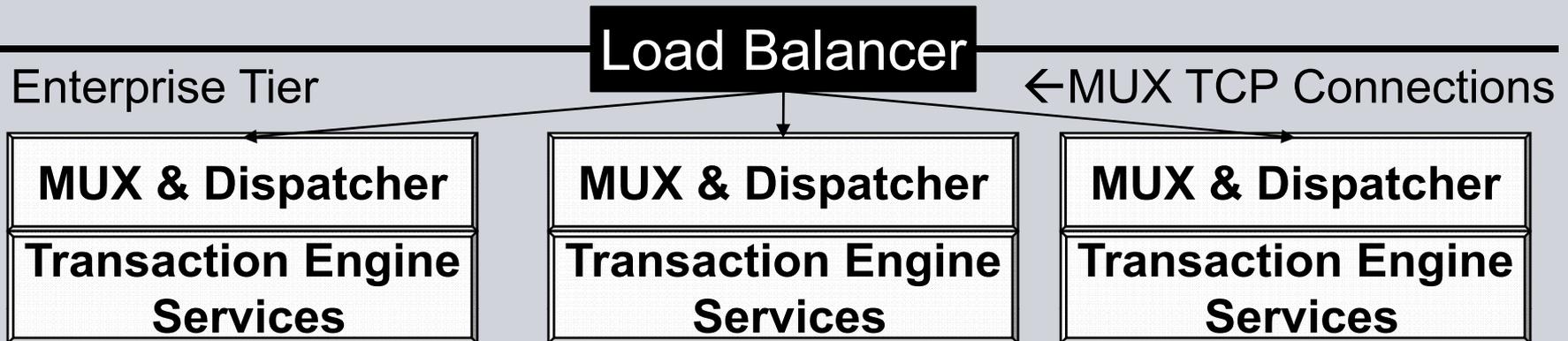
High Availability Enterprise Tier Clustering



Details

- Redirecting TCP connections is less common, but should be possible with most load balancers
- Every MUX in a cluster must have the same name (VC_HOSTNAME)
- Same name implies same services provided by all – the nodes are proper **clones** of each other
- Internally, each MUX has a unique ID (GUID) it generates, so that a MUX outside the cluster can maintain multiple distinct TCP connections to what otherwise appear to be the same MUX
 - Responses go back to correct MUX

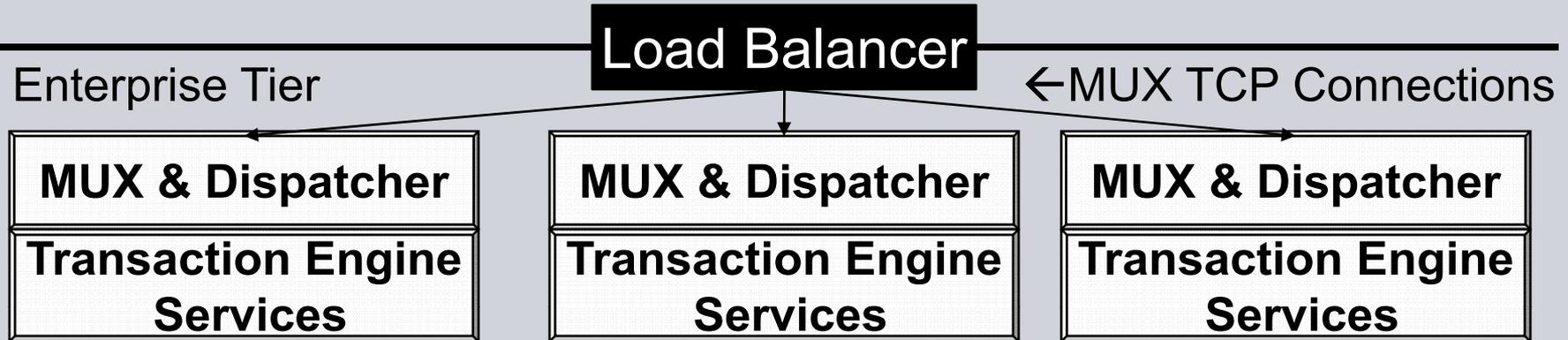
High Availability Enterprise Tier Clustering



Details

- Each MUX and Dispatcher operates as if solitary
 - Database locking provides transactional integrity against concurrency
- Purely stateless user sessions
 - Full session context provided with each request
 - From Web Tier for thin client
 - From Client Tier for Classic Client

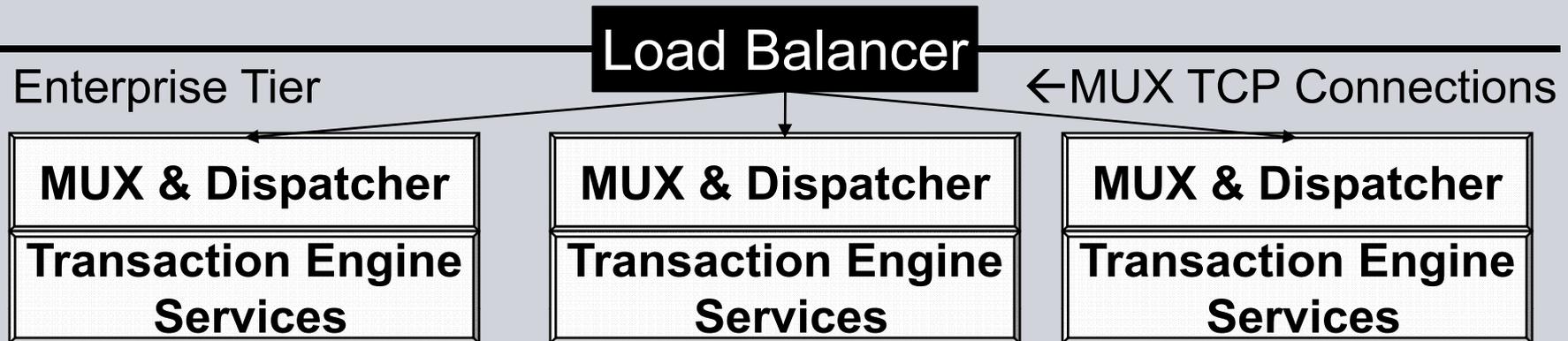
High Availability Enterprise Tier Clustering



Details

- All MUX clients must use a TCP network name/address that directs connections to the load balancer
 - Web Tier WAR files: same MUXHost and MUXPort
 - Client/Enterprise Tier configuration **hosts** map: one, consistent TCP host name and port → the load balancer's entry point
- Enterprise Tier Load Balancer must receive TCP connections on the VC_MUX_IPPORT and redirect **each connection** to an Enterprise Tier MUX on one of the nodes of the Enterprise Tier cluster

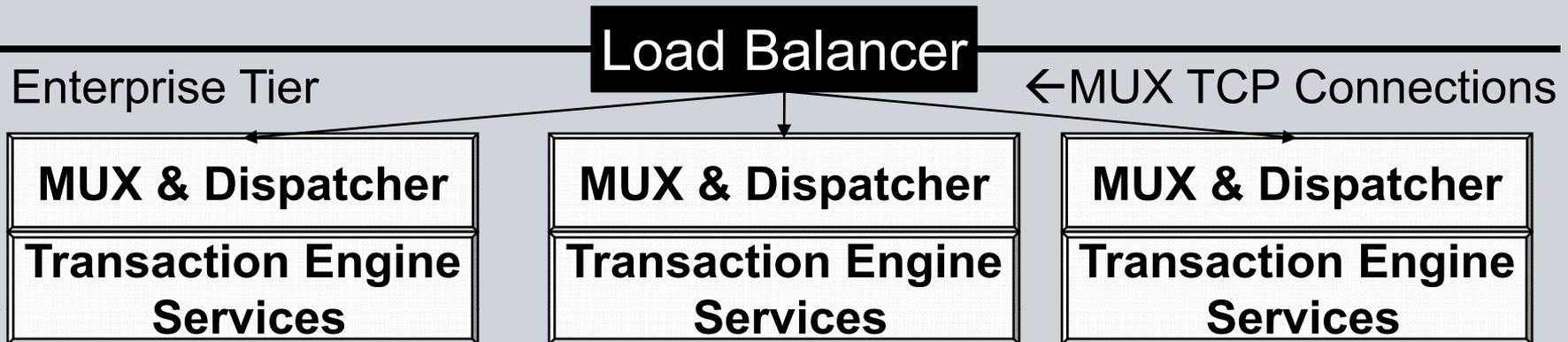
High Availability Enterprise Tier Clustering



Details

- Most general – Load Balancer should accept TCP connections from:
 - Client Tier
 - Not depicted, but necessary for sites that support other clients (Classic Client, e!Vista GUI, I-deas Enterprise, or any custom client based these technologies)
 - Web Tier
 - Another Enterprise Tier
 - For example, a load balancer might be supporting a cluster of “corporate servers” that services remote “work group servers”

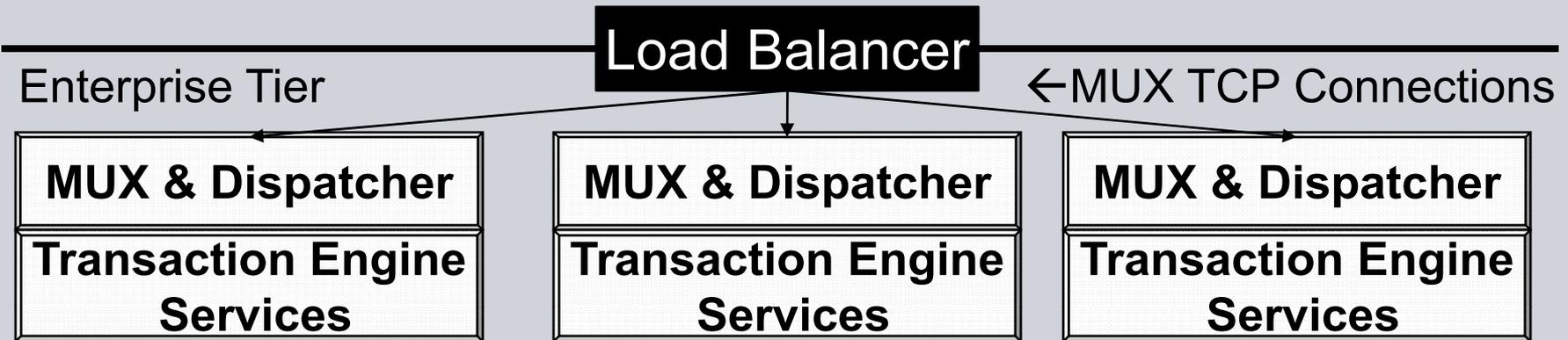
High Availability Enterprise Tier Load Balancing and Failover



Best Performing Configuration

- Duration of MUX TCP connections is ***much*** longer than HTTP connections – often days
 - Unusual configuration for Load Balancers that are usually working with HTTP requests
- Persistence affects Session Registry (see 2025302 and the `ENABLE_CLIENT_SESSION_REGISTRY` configuration variable)
- Persistence depends on traffic over the MUX connection versus the `VC MUX IDLE TIMEOUT` configuration variable setting

High Availability Enterprise Tier Load Balancing and Failover



Best Performing Configuration

- Load Balancer **must not** time out TCP connections unilaterally
 - Expect proper adherence to TCP/IP specifications (RFCs)
 - Expect endpoint hosts to manage the duration of the TCP connection if idle with traditional keepalive implementations (default: two hours)
- Many “firewalls” are configured for one hour timeout (violating TCP keepalive)
 - **Common workaround: shorten keepalive in endpoint TCP stacks to 55 minutes or so**

Hardware Platform – any supported

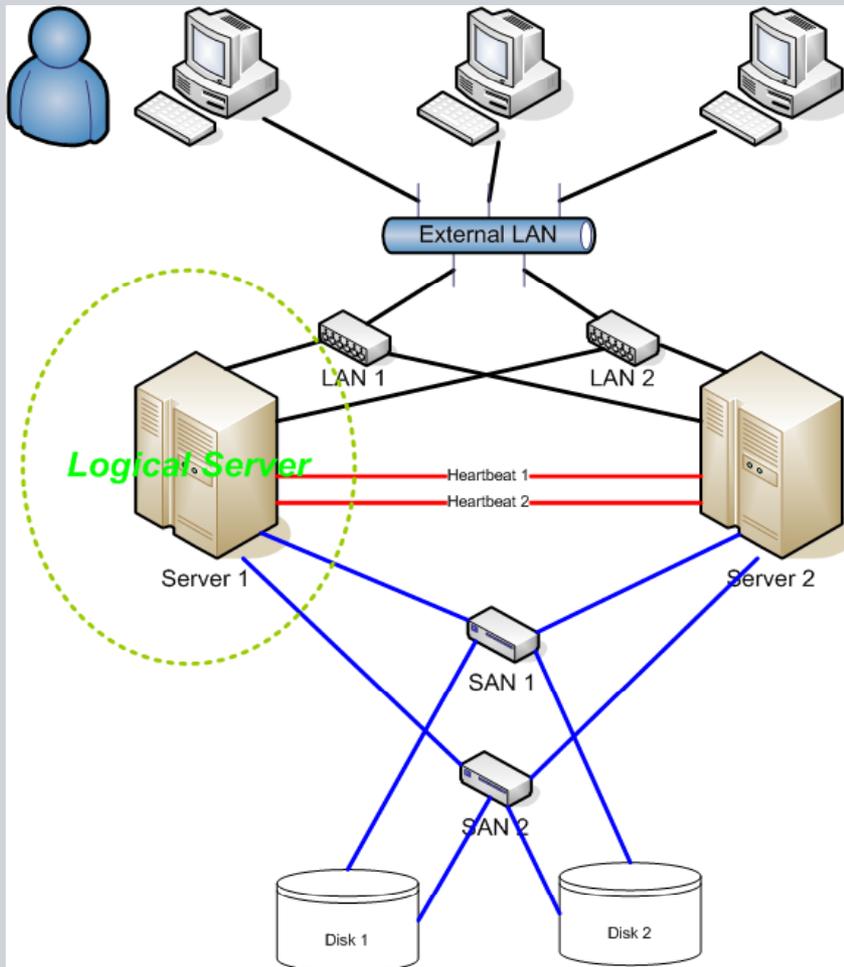
- Some failover technologies require hardware assistance
- For example, Sun Cluster requires a “private interconnect” network between two Sun boxes

Operating Systems – any supported

- Except: if Enterprise Tier failover is required for **software singletons**, this limits the choices
 - Specifically: ***no support for Enterprise Tier failover on Windows***

High Availability Enterprise Tier – Platforms

Wikipedia



Sun Cluster requires a “private interconnect” network between two Sun boxes

- Shown here as “Heartbeat 1” and “Heartbeat 2”

High Availability Enterprise Tier Customer Deployments



Customer	Operating System	Notes
1	Windows	Enterprise Tier Failover not necessary for this customer
2	Windows	Enterprise Tier Failover not necessary for this customer
3	HP-UX	
4	Solaris	

Only one Autonomy Indexer process throughout cluster (singleton)

- Sequential update processing required per object

Event processors triggered locally

- Means that when events are being much more heavily generated on one node, the processors on the other nodes are not alerted
- Usually still distributes work among nodes, because processors drain their queues

Event Queue Broker loses some efficiency

- Competes at RDBMS as individual event processors do without the broker
- Still more efficient, because only one Event Queue Broker per event type per node, versus many event processors of the same event type per node, all competing for events
- Under analysis and investigation for future improvements

Message Access Rules

- Rules files must be shared, but there is no management of file concurrency in “rulefile” when generating a new file
 - Sites must manage concurrency
- Triggers (e.g., Admin Editor, “rulefile -r” or “rulefile -R”) only apply on the specific node
 - Similar to deployment with multiple mloaders: use “rulefile” then “rulefile -R”
 - Difference is that rules files themselves should be shared via SAN, so run “rulefile” once on one node, then “rulefile -R” on all nodes to signal availability of new rules file

High Availability Enterprise Tier

Questions?



Summary

Using Clustering and Load Balancing, sites can achieve High Availability

- No **unplanned** downtime

True horizontal scalability – Web & Enterprise Tiers

- Tiers scale independently
- Just “add a box” to serve increased user activity

Teamcenter Enterprise has been deployed to provide high availability at several customer sites and is available to all of our customers on Teamcenter 2005 MP1, 2005 SR1 and 2007.

Contact

Mark Ludwig
Teamcenter Architect

E-mail: ludwig.mark@siemens.com

www.siemens.com/plm