High Availability Architectures for Linux in a Virtual Environment

Scott Loveland
IBM Systems and Technology Group
Poughkeepsie, NY
d10swl1@us.ibm.com

March 15, 2010
Session 9276

Trademarks

Agenda

- Definitions of High Availability
- Fundamental High Availability Guidelines
- Virtualization considerations for HA with z/VM and LPARs
- Reference Architectures
  - Active / Cold Standby Cluster
  - Active / Active Application Server Cluster with MQ HA cluster
  - Active / Active Application Server Cluster with Database Mirroring on Linux
  - Active / Active Application Server Cluster with Database Sharing on Linux
  - Active / Active Application Server Cluster with Database Sharing on z/OS in a Parallel Sysplex
  - Active / Active Application Server Cluster with Database Sharing on z/OS Across Cities
- How we've tested these Architectures

Definitions of High Availability

- **High Availability (HA) –** Provide service during defined periods, at acceptable or agreed upon levels, and masks *unplanned* outages from end-users. It employs Fault Tolerance; Automated Failure Detection, Recovery, Bypass Reconfiguration, Testing, Problem and Change Management
- **Continuous Operations (CO) --** Continuously operate and mask *planned* outages from end-users. It employs Non-disruptive hardware and software changes, non-disruptive configuration, software coexistence.
- **Continuous Availability (CA) --** Deliver non-disruptive service to the end user 7 days a week, 24 hours a day (there are no planned or unplanned outages).

Definitions adopted from the
IBM HA Center of Competence in Poughkeepsie, NY.

Fundamental High Availability Guidelines

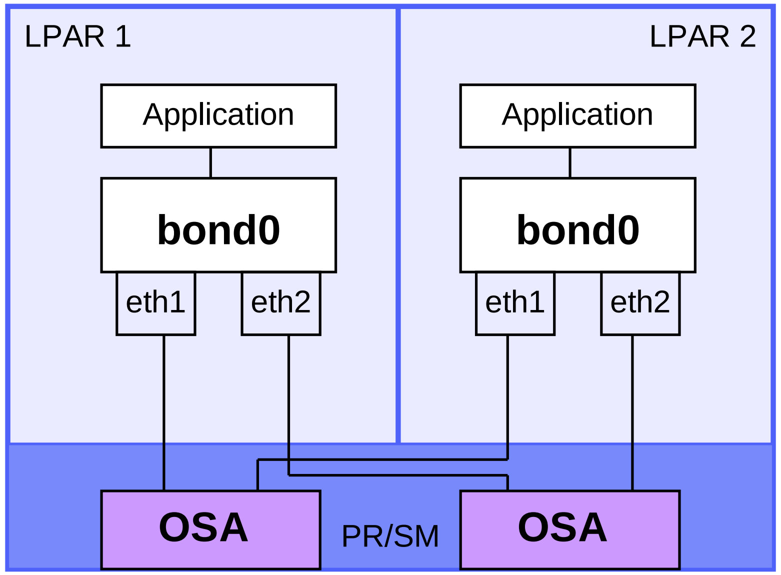
- **Redundancy, Redundancy, Redundancy –** Duplicate everything to eliminate single points of failure. Examples:
  - ***Duplication***
    - LPARs
    - Operating systems
    - Software servers
    - Control Units
    - I/O paths (FICON, FCP)
    - Ports and cards (FICON, FCP, disk host adapters, OSA)
    - FICON Directors
    - Fibre channel switches
    - Network routers
  - ***Mirroring***
    - Disks (system, data)
    - Data

Fundamental High Availability Guidelines (cont.)

- **Protect Data Consistency –** Provide ability for data and file systems to quickly return to a point of consistency after a crash.
  - Journaling databases
  - Journaling file systems
  - Mirroring
- **Automate Detection and Failover --** Let the system do the work in order to minimize outage windows.
  - Multipathing
    - FICON vs. FCP
  - VIPA
  - Monitoring and heart beating
  - Clustered middleware
  - Clustered operating systems

Leveraging Virtualization for Network Interface Redundancy and Automated Failover

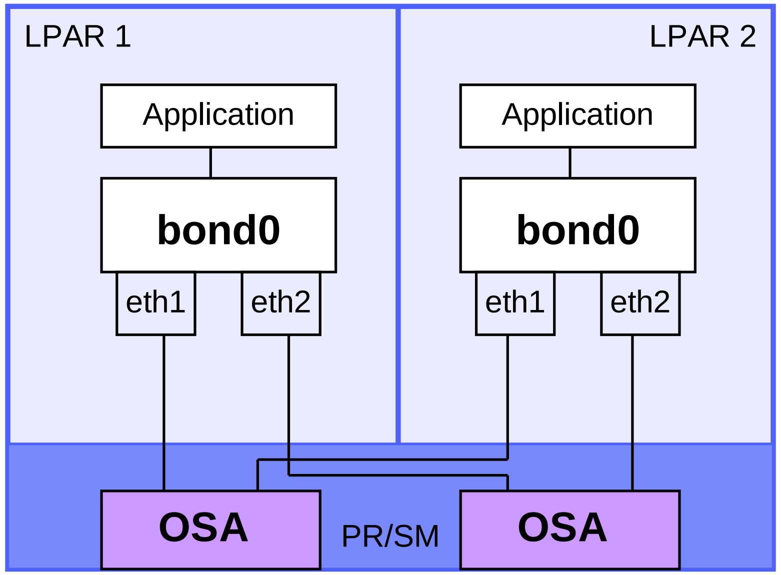
Resource Virtualization:
OSA Channel Bonding



- Linux *bonding* driver enslaves multiple OSA connections to create a single logical network interface card (NIC)
- Detects loss of NIC connectivity and automatically fails over to surviving NIC
- No dynamic routing (OSPF) dependency
- Active/backup & aggregation modes
- **Separately configured for each Linux**

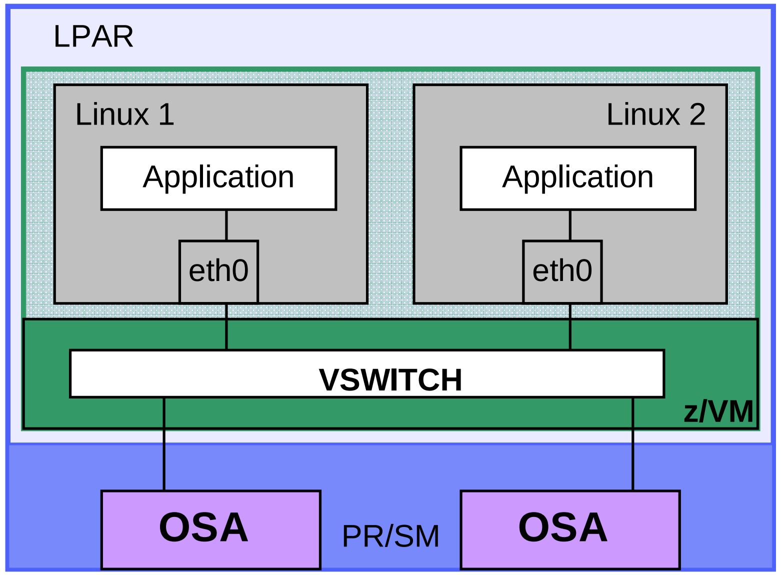
Leveraging Virtualization for Network Interface Redundancy and Automated Failover

Resource Virtualization: OSA Channel Bonding



- Linux *bonding* driver enslaves multiple OSA connections to create a single logical network interface card (NIC)
- Detects loss of NIC connectivity and automatically fails over to surviving NIC
- No dynamic routing (OSPF) dependency
- Active/backup & aggregation modes
- **Separately configured for each Linux**

System Virtualization: z/VM VSWITCH



- z/VM *VSWITCH* enslaves multiple OSA connections. Creates virtual NICs for each Linux guest
- Detects loss of physical NIC connectivity and automatically fails over to surviving NIC
- No dynamic routing (OSPF) dependency
- Active/backup & aggregation modes
- **Centralized configuration benefits all guests**

High Availability considerations of z/VM and LPARs

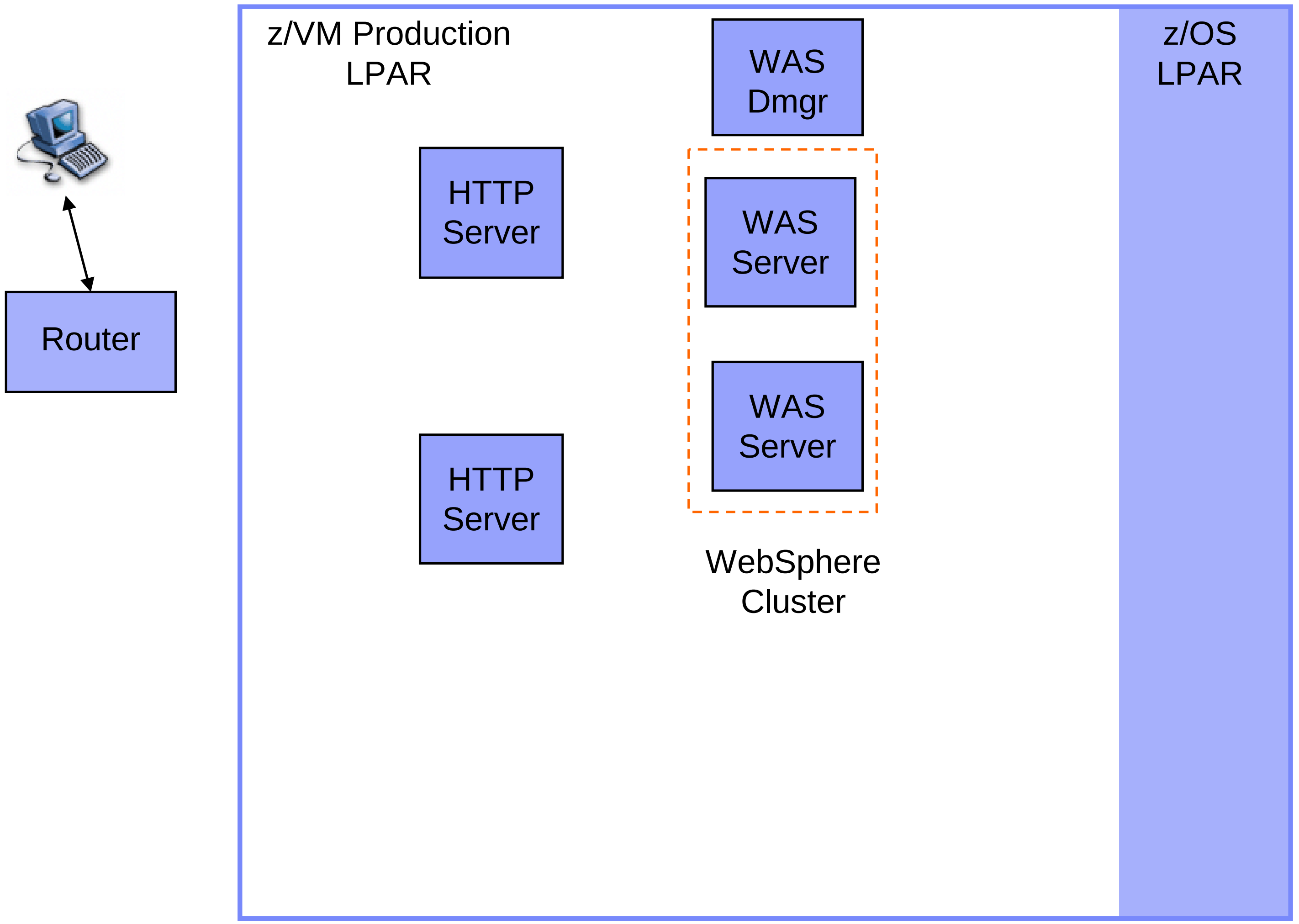
Potential Single Point of Failure	Relative Probability of Failure	Relative cost to duplicate
System z hardware	Very Low	High
Disk Subsystem microcode	Very Low	Medium
LPAR	Very Low	Low
z/VM	Low	Low
Linux	Low	Very Low
Application	High	Very Low

Besides hardware and software failures, what else can cause down time for the application?

- zSeries hardware upgrades requiring POR.
- z/VM maintenance.
- Linux kernel maintenance that requires reboot.
- Application maintenance

Leveraging Virtualization to Help Reduce Points of Failure

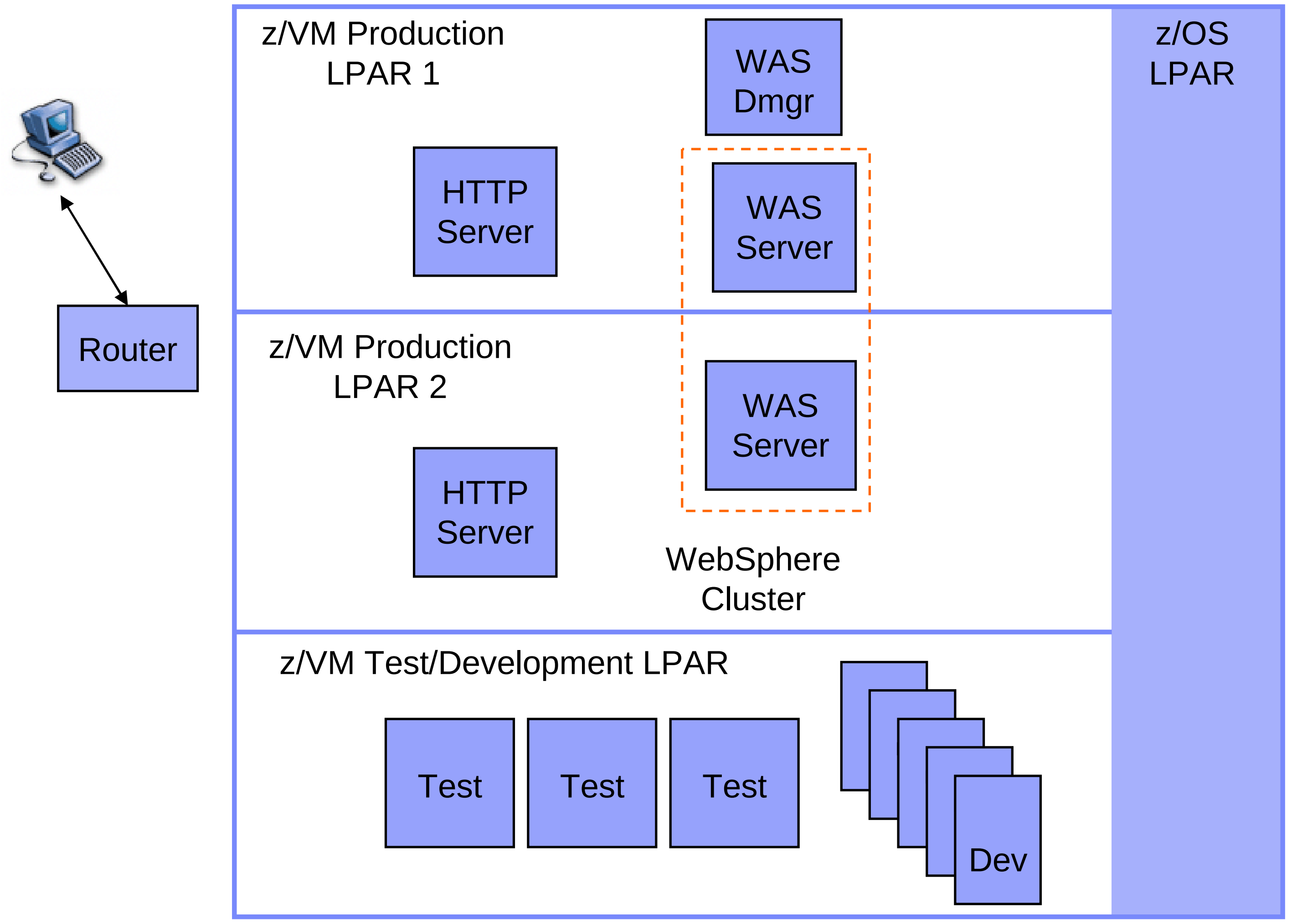
One z/VM LPAR with many Linux virtual servers enables resource sharing



- Applications run in cluster across multiple Linux virtual servers
- All Linux virtual servers draw from a common pool of memory and IFLs.
- Resources from a failed server can flow to surviving servers
- Smaller cluster helps reduce failure points
- Remaining SPoFs include System z hardware, LPAR, z/VM.

Highly Efficient HA Solution for Linux on System z

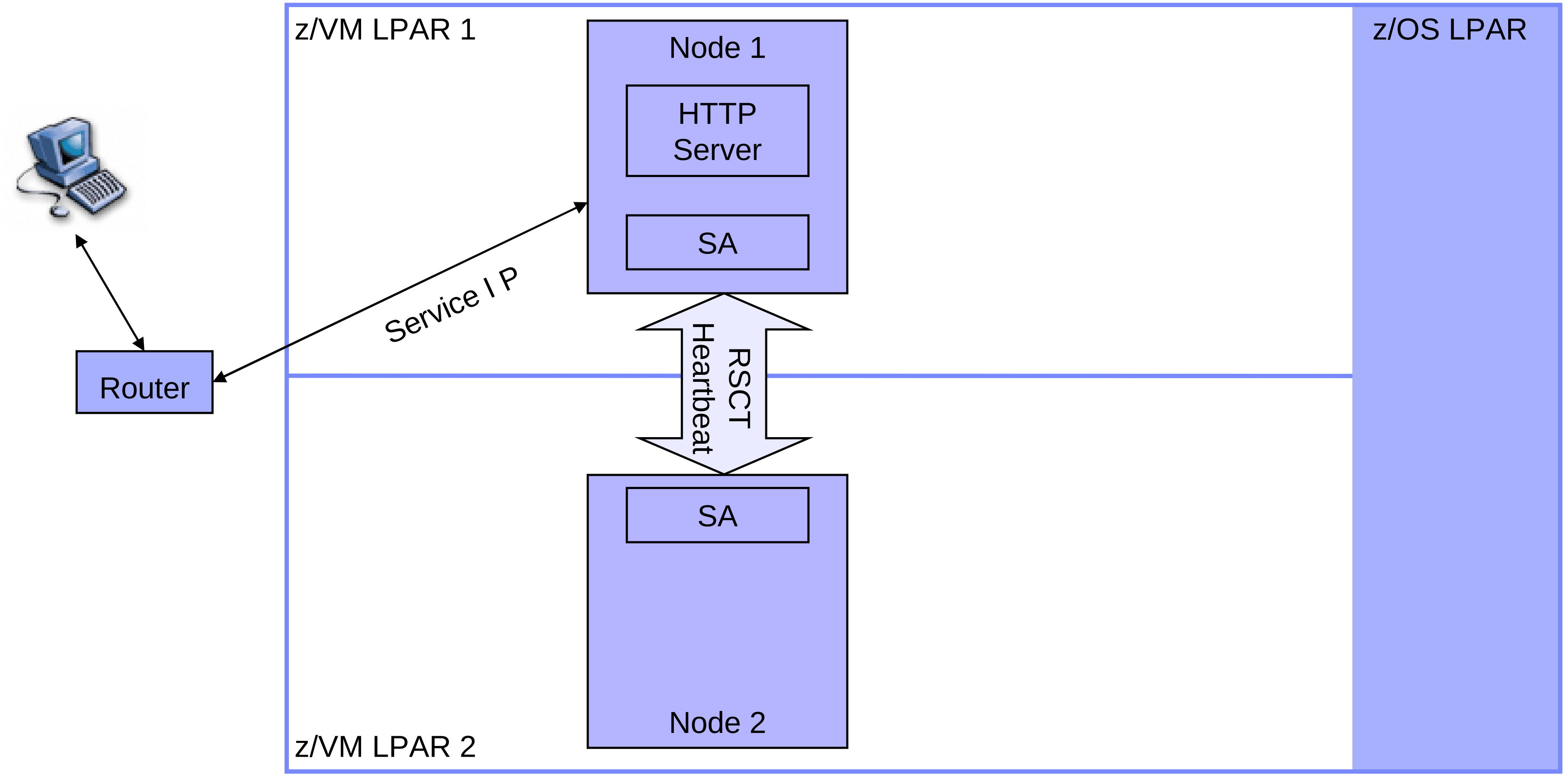
3 z/VM LPARs. This solution is designed to eliminate most failure points, for a reasonable cost.



- Two LPARs run production workload.
- One LPAR runs Test/Dev workload.
- Applications run in clusters split between the prod LPARs.
- All LPARs can share a common pool of IFLs.
- Traffic encryption needs minimized
- Remaining SPoF is System z hardware.

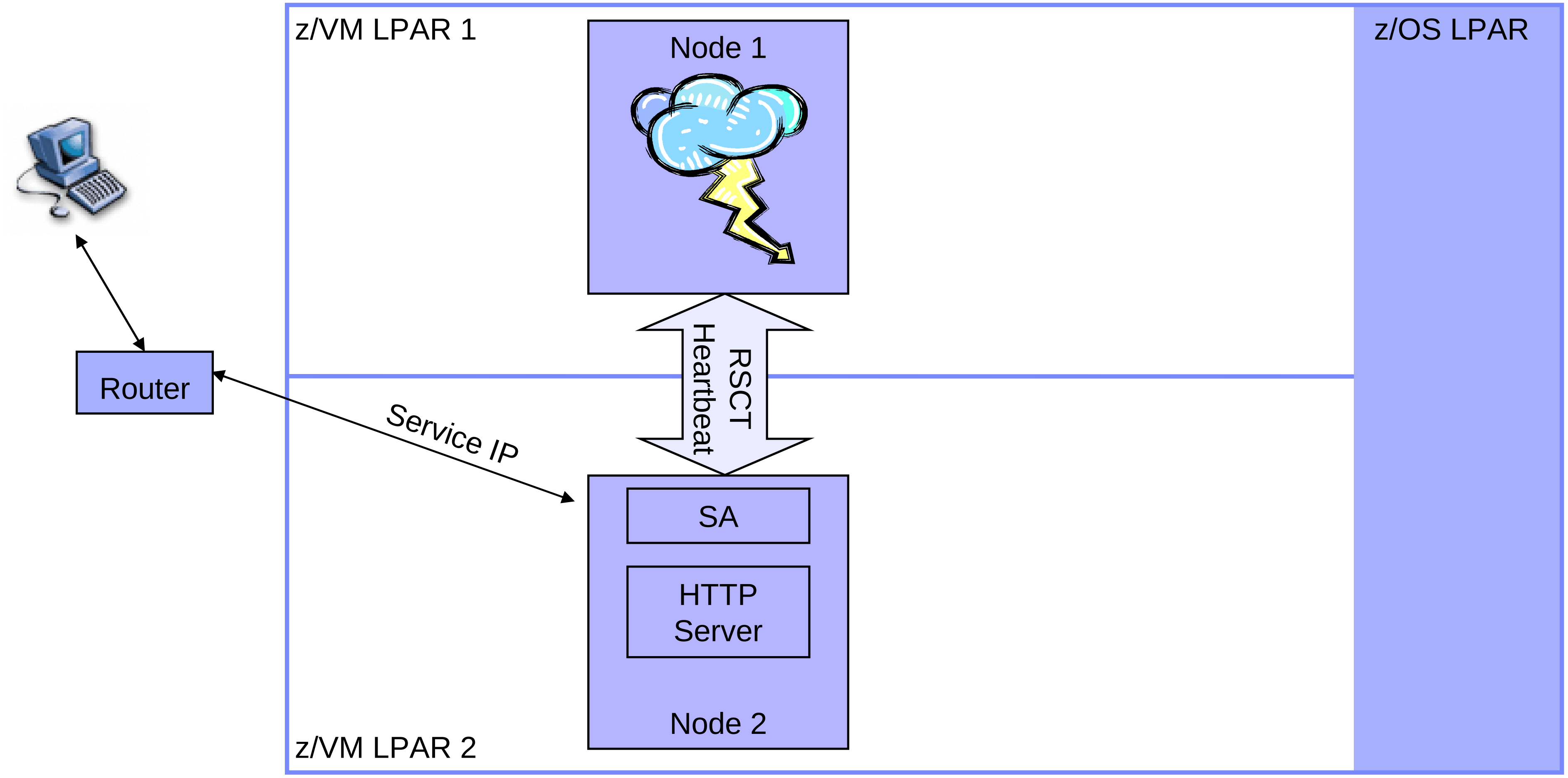
Reference Architecture: Active / Cold Standby Cluster

1. IBM Tivoli System Automation for Multiplatform monitors for outage

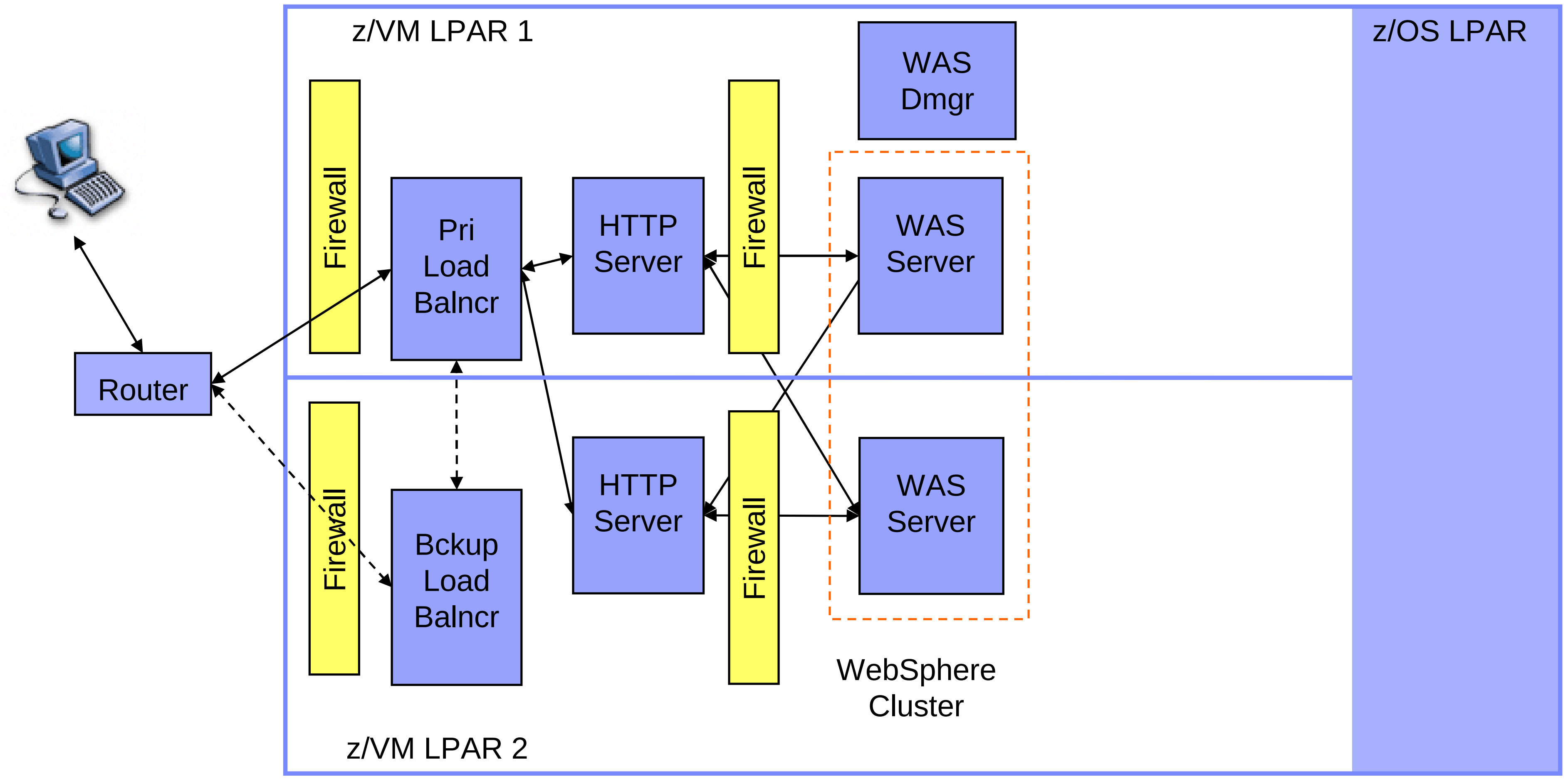


Reference Architecture: Active / Cold Standby Cluster

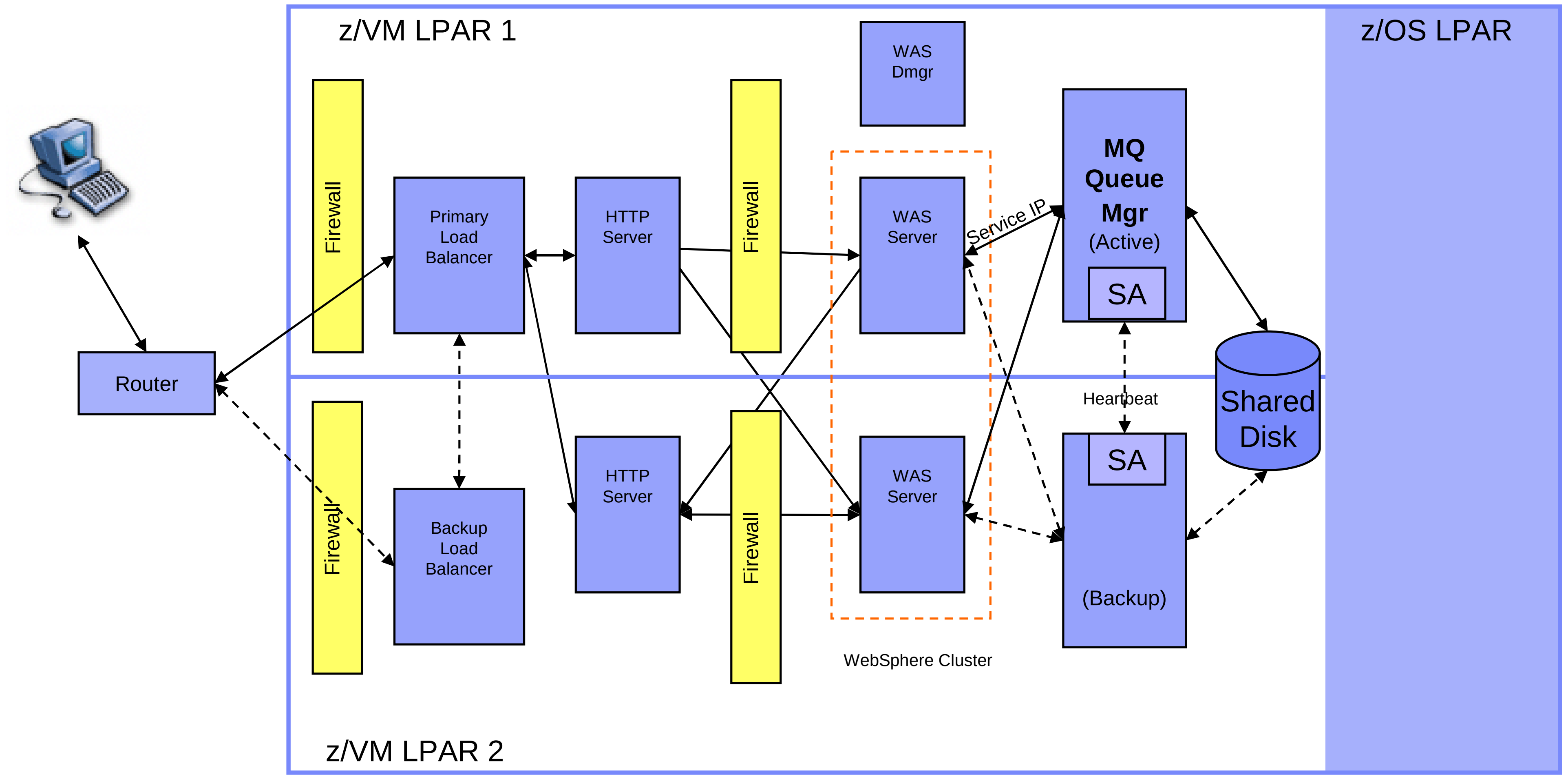
2. IBM Tivoli System Automation detects outage and automates failover



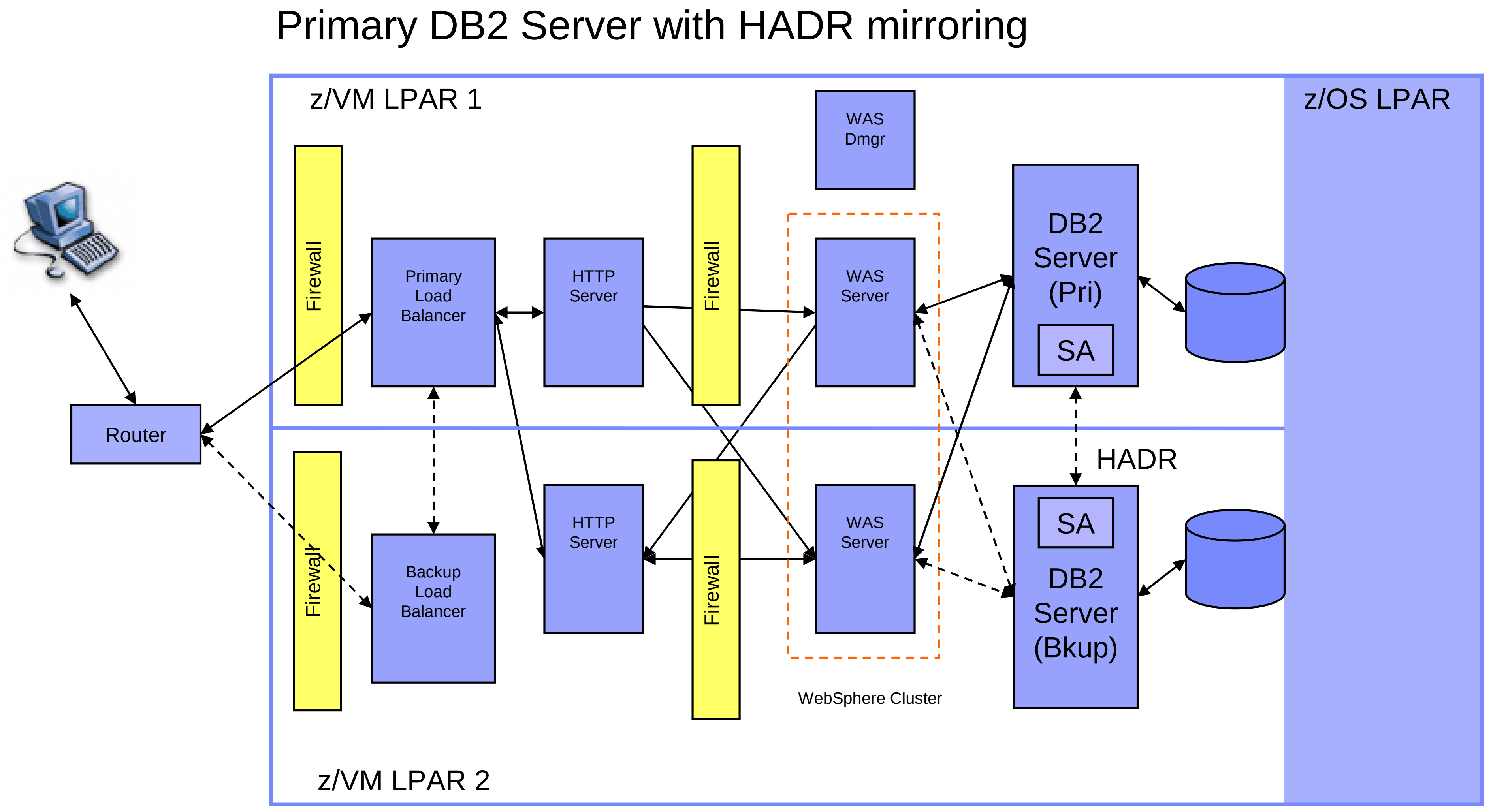
Reference Architecture: Active/Active Application Server Cluster



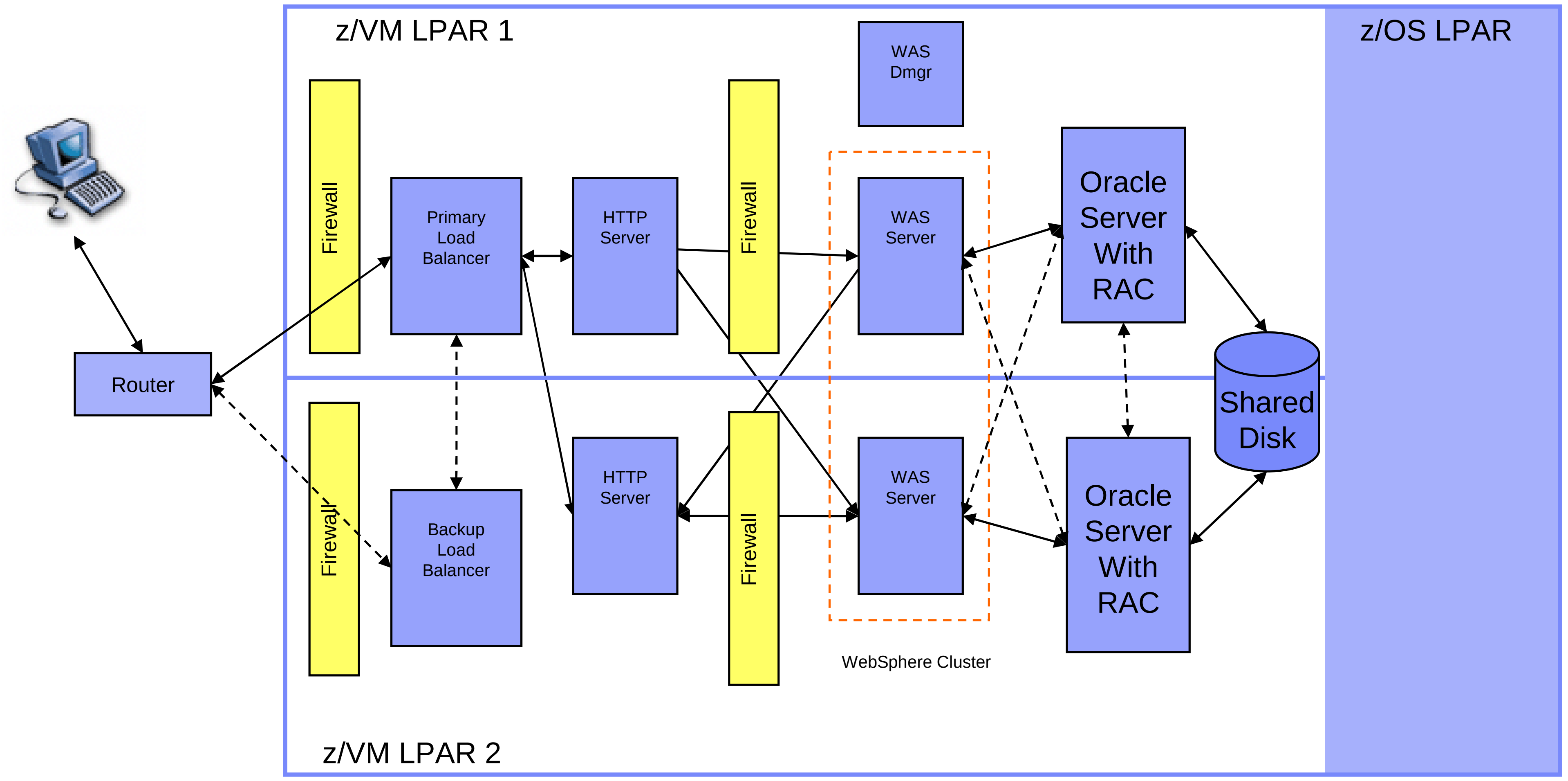
Reference Architecture: Active/Active Application Server Cluster with MQ HA Cluster



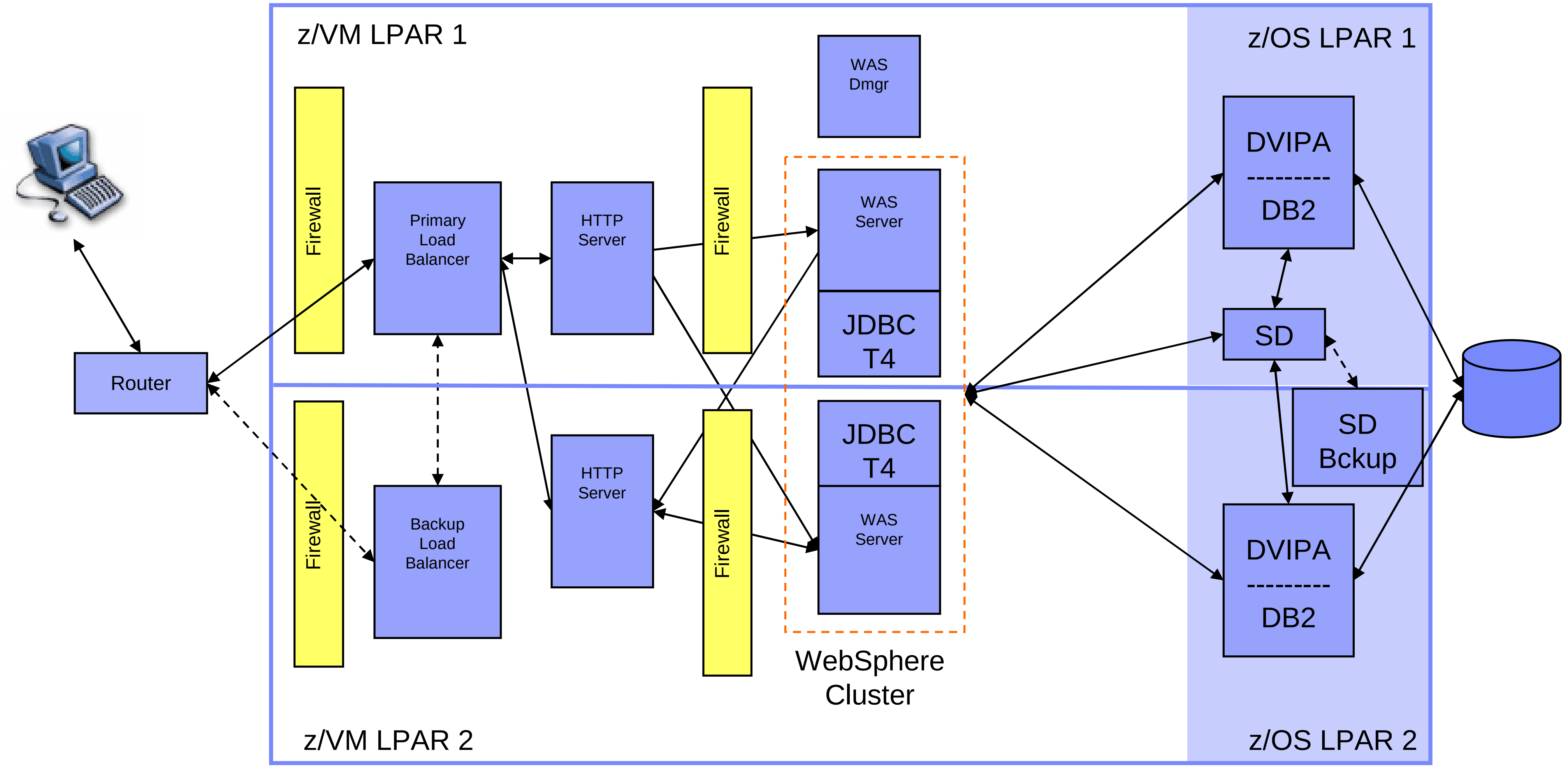
Reference Architecture: Active/Active Application Server Cluster with Database Mirroring on Linux



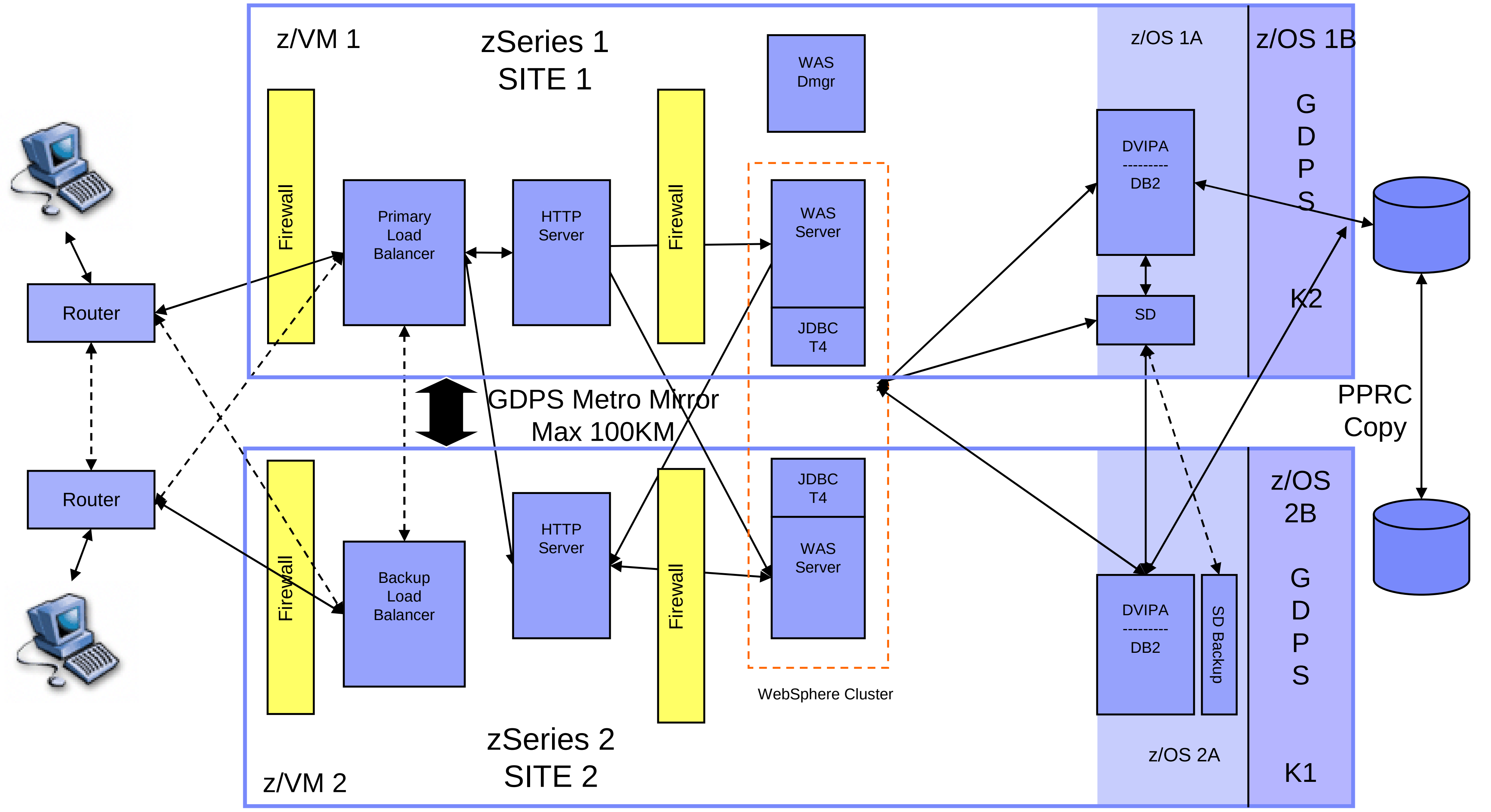
Reference Architecture: Active/Active Application Server Cluster with Database Sharing on Linux



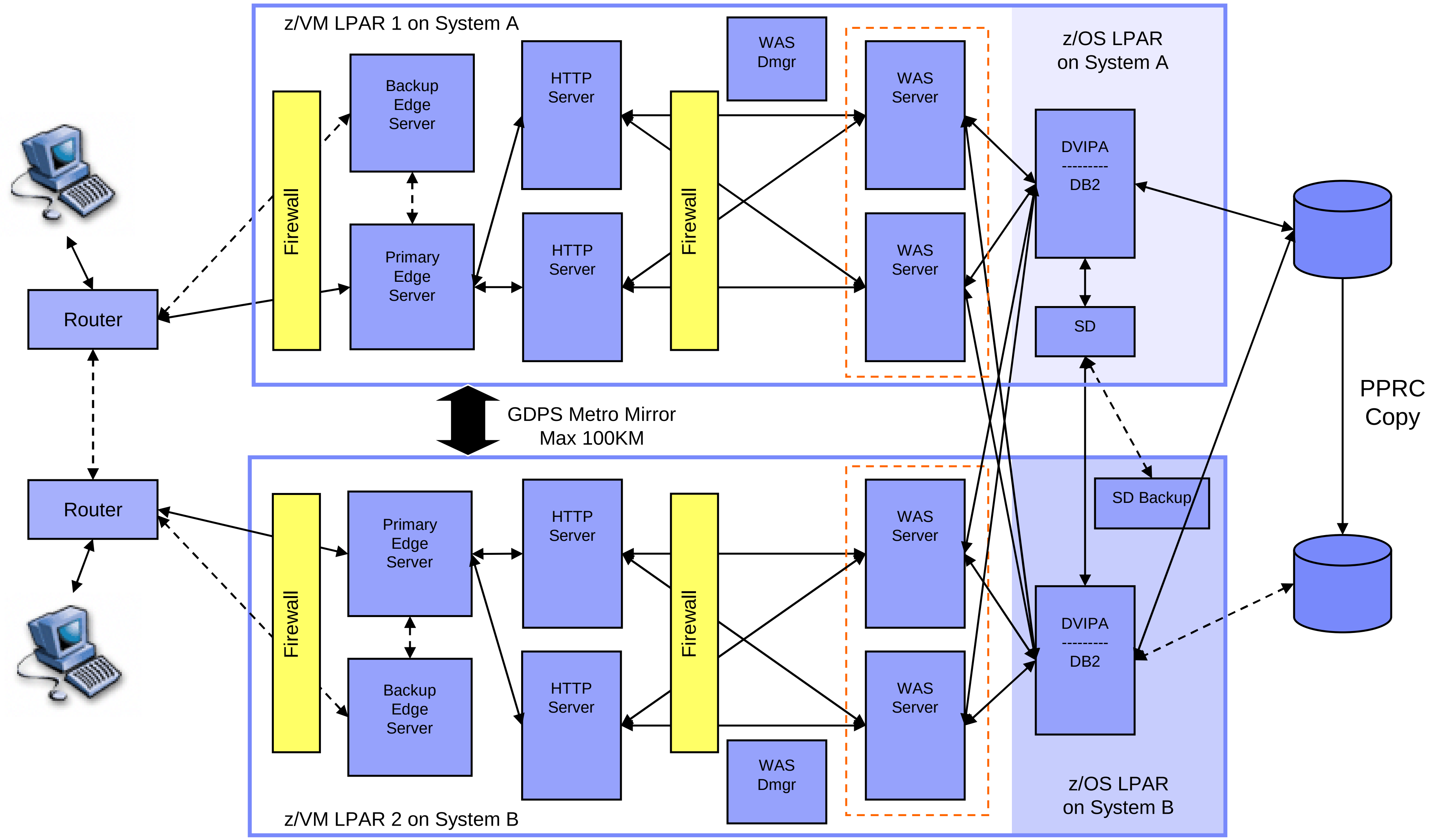
Reference Architecture: Active/Active Application Server with Database Sharing on z/OS in a Parallel Sysplex



Reference Architecture: Active/Active Application Server Cluster with Database Sharing on z/OS Across Cities



Another Option: Two WAS Cells.



Architectural Alternatives (Either Approach)

Each server must have sufficient capacity to run the workload should the other site fail. This can be costly to configure. Alternatives to help optimize that cost:

- **Expendable work.** Workload is normally split between sites. Each site is over configured so it is ready to instantly absorb the other's work if needed. During normal operation, excess capacity at each site is consumed by lower priority, expendable work. In a failover situation, expendable work is stopped to free up resources for the other site's incoming work.
- **Capacity Upgrade on Demand.** Workload is normally split between sites. Each site is only configured with capacity to handle normal operations, but is also setup with Capacity Upgrade on Demand (CUoD). In a failover situation, additional CPUs are brought online at the surviving site. This approach gives lower cost as long as reduced performance can be tolerated during the first few minutes of a failover until the upgrade CPUs are brought online.
- **Active / Passive Deployment**. Workload normally contained at Site 1, standby server capability only at Site 2 (the GDPS K1 z/OS system is the only active system at Site 2). Primary and secondary disk configurations active at both sites. During fail over, CUoD adds resources to surviving site, and standby servers are started. Helps save hardware and software costs, but lengthens recovery time.

How We’ve Tested These Architectures

- Exhaustive world-wide review by the IBM HA Community of Practice, and HA Center in Poughkeepsie.
- Stress testing done by the IBM Test and Integration Center for Linux, in Poughkeepsie.

Summary

- As with other environments, maximizing availability translates to eliminating single points of failure.
- The virtualization and shared-resource capabilities available with System z can help
- A variety of possible HA architectures are available. We've touched on several.
- More details available with the referenced papers …

Getting more Information

Get papers on this subject

- *High Availability Architectures for Linux on IBM System z*
  - Contains detailed information on these architectures and why we chose certain features.
  - Available at http://www-03.ibm.com/servers/eserver/zseries/library/whitepapers/pdf/HA_Architectures_for_Linux_on_System_z.pdf
  - **update coming soon…**
- *zSeries Platform Test Report for z/OS and Linux Virtual Servers, June 2006 Edition (Part 4. Linux Virtual Servers)*
  - Contains detailed information on the setup and testing of these architectures.
  - Available at **http://www-03.ibm.com/systems/services/platformtest/servers/systemz.html**
- *zSeries Platform Test Report for z/OS and Linux Virtual Servers, June 2007 (Chapter 13: zLVS PET Recovery)*
  - Contains detailed information on how to prepare for various potential failures, the indicators that alert you to each failure, and what actions to take to recover
  - Includes GDPS/PPRC Multiplatform Resiliency scenarios
  - Same URL as above

Getting more Information

Redbooks and other White papers on this subject

- *WebSphere Application Server Network Deployment V6: High Availability Solutions*. SG24-6688
- *WebSphere Application Server V6: Scalability and Performance Handbook.* SG24-6392
- WebSphere V5.1 Performance, Scalability, and High Availability, SG24-6198
- *PeopleSoft V8 on zSeries Using Sysplex Data Sharing and Enterprise Storage Systems.* Chapters 3 and 4. SG24-6093
- *IBM DB2 Connect Quick Beginnings for DB2 Connect Enterprise Edition Version 8*, GC09-4833, chapter 15
- *Leveraging z/OS TCP/IP Dynamic VIPAs and Sysplex Distributor for Higher Availability,* IBM, *www-03.ibm.com/servers/eserver/zseries/library/techpapers/gm130165.html*
- *Automating DB2 HADR Failover on Linux using Tivoli System Automation* , IBM, *ftp://ftp.software.ibm.com/software/data/pubs/papers/hadr_tsa.pdf*
- *Technical Comparison of DB2 HADR and Oracle Data Guard,* IBM, *ftp://ftp.software.ibm.com/software/data/pubs/papers/hadr-comp.pdf*
- *Oracle Real Application Clusters,* Oracle, *http://www.oracle.com/technology/products/database/clustering/index.html*
- IBM GDPS Web site: *www.ibm.com/servers/eserver/zseries/gdps.html*

The End

Backup

Reference Architecture: Active / Cold Standby Cluster

Flow

1. **IBM Tivoli System Automation for Multiplatforms** (SA). SA runs on each node in the cluster. It monitors cluster nodes and exchanges information through Reliable Scalable Cluster Technology (RSCT) services.
2. **Service IP.** SA creates a Service IP address as an alias on an appropriate network adapter on Node 1 (where the HTTP server will be started).
3. **HTTP Server.** One and only one instance of the HTTP server is defined to SA to be able to run in either of the two nodes, with a "depends on" relationship to a single IP address (the Service IP). SA starts the HTTP server on Node 1, then at user-defined intervals invokes a script to confirm it is still up and serving pages. It also monitors the Linux node itself to ensure it remains active.
4. **Failure Occurs.** The HTTP Server node crashes. RSCT determines that Node 1 is no longer responding. TSA moves the Service IP over to Node 2, then restarts the HTTP server there.

Product Versions

- Any version of IBM Tivoli System Automation for Multiplatforms

Architectual Alternatives

- **Tie Breaker.** When SAs lose communication with one another, a “split brain” scenario could potentially develop. The subcluster with greater than half of the defined nodes claims “quorum” and takes control (if the subcluster losing quorum contains any critical resources which can only be active on one node in the cluster, it will immediately shut itself down). This works when there are an odd number of nodes. But when the total number of nodes is even, a tie breaker is needed. There are multiple choices:
  - **ECKD tie breaker**. Whoever gets a reserve on a designated shared volume wins. Hardware guarantees only 1 winner. But could grant quorum to a node which cannot access the network.
  - **Network tie breaker.** Whoever can ping a central point (such as a router on the same subnet as the cluster), wins. Ensures the winner can access the network, but are rare situations where both clusters could win the tiebreaker.
  - **Operator tie breaker.** Asks a human to decide.
- **Cluster Management Software.** Some other form of cluster management software could be used. For example, the Linux-HA project offers a popular open source solution

Reference Architecture: Active/Active Application Server Cluster

Flow

1. **Communications within an LPAR.** All communications within each z/VM LPAR are done via a z/VM Virtual Switch (vswitch). One vswitch with two vswitch controllers (VM userids) is used in each LPAR. Each vswitch uses two sets of OSA ports, preferably on two separate OSA features. Should one vswitch controller or OSA feature fail, communications fail over to the other.
2. **Load Balancer.** Requests enter a router that sends traffic to the cluster IP alias of the primary Load Balancer. Should this Load Balancer fail, the backup Load Balancer will detect the failure, claim the cluster IP alias, and take over for the primary Load Balancer.
3. **HTTP Server.** The Load Balancer sprays requests between the two HTTP servers. Should one of the HTTP servers fail, the Load Balancer detects this and will not route requests to it. The HTTP server serves static pages. It also routes WebSphere requests via the WebSphere plugin to the two WebSphere servers in the cluster. Should one of the WebSphere servers fail, the plugin will detect this and not route requests to it.
4. **WebSphere.** The application is deployed to a WebSphere Application Server cluster consisting of two nodes. WebSphere manages the deployment of the application onto the nodes of the cluster, and can upgrade the application on the fly. User session data is replicated among the cluster members so that if a cluster member fails the transaction can typically be continued on the other cluster member.

Product Versions

- WebSphere Application Server Network Deployment V6.0 or newer. This architecture can be done with WebSphere v5 but lacks some of the HA features of WebSphere v6. For these reasons we recommend v6.
- z/VM 4.4 or higher for VSWITCH support

Reference Architecture: Active/Active Application Server Cluster with MQ HA Cluster

Flow

1. **Edge, HTTP, WebSphere** have already been covered and do not change in the scenario.
2. **MQ Client.** JMS connection factories and activation specifications are configured to use client transport mode. The queue manager uses a service IP address which moves as the queue manager fails over. WebSphere Application Server automatically reconnects in the event of JMS connection failures.
3. **MQ Queue Manager**. The primary queue manager is monitored by a Tivoli System Automation agent. The file system containing queue manager files should be mounted only by the Linux guest running the active queue manager. The failure of the primary queue manager is detected by TSA, which initiates the failover by running a (user-created) script which mounts the queue manager’s file system on the backup Linux guest, starts the backup queue manager on the backup Linux guest, and switches the service IP address of the primary queue manager to the backup node

Product Versions

- WebSphere MQ v 6.0.2.7 and above
  - SupportPac MC91 provides instructions for using WMQ in HA clusters (though it doesn’t discuss Linux specifically). It has been withdrawn and is no longer being enhanced but is still available for download: http://www-01.ibm.com/support/docview.wss?rs=171&uid=swg24011869&loc=en_US&cs=utf-8&lang=en
  - See also SupportPac IC91, for Linux-specific information: http://www.ibm.com/support/docview.wss?rs=171&uid=swg24013413&loc=en_US&cs=utf-8&lang=en
- Tivoli System Automation for MultiPlatforms, V1.2 and above.

Architectural Alternatives

- **Standby vs. Takeover configuration.** This architecture shows a standby configuration, where the backup node is not processing any critical work until a failover occurs. The solution, however, is also suitable for a mutual takeover configuration, where both nodes are processing highly available & movable work.
- **WMQ Queue manager clusters.** WMQ Clusters provide a means of reducing administration and providing load balancing of messages across instances of cluster queues. They also offer availability benefits, however a queue manager cluster does not manage the recovery of a failed queue manager.
- **WMQ Multi-instance Queue Manager.** This approach is new with WebSphere MQ v7.0.1 and above. One set of logs and data are shared R/W by an active and a standby queue manager instance, using network-attached storage. It uses built-in MQ services for failover, so it doesn't require an external cluster manager such as TSA. However, if there is a need to coordinate other resources when there is an MQ failure, then external cluster management software would still be needed
- **Cluster Management Software.** Some other cluster management software could be used rather than Tivoli System Automation, such the open source Linux-HA package.

- NOTE: Queue sharing groups are not an alternative, since they are only available for z/OS.

Reference Architecture: Active/Active Application Server Cluster with Database Mirroring on Linux

Flow

1. **Edge, HTTP, WebSphere** have already been covered and do not change in the scenario.
2. **DB2 Client (JDBC).** WebSphere runs the application and sends DB2 data requests to the Primary DB2 Server.
3. **The DB2 HADR** (High Availability Disaster Recovery) feature is used to provide failover in case of a DB2 failure. HADR uses two DB2 servers and two databases to mirror the data from the primary database to the standby.
   1. HADR also communicates to the DB2 clients (the JDBC driver in our case) to inform them of the address of the standby server.
   2. IBM Tivoli System Automation running on both DB2 Servers is designed to automatically detect a failure of the primary and issues commands on the standby for its DB2 to become the primary.
   3. When communication to the primary DB2 server fails, it is designed so that the clients automatically route requests to the standby server (in-flight transactions are rolled back and the application can then continue from where it left off).

Product Versions

- DB2 Universal Database (DB2 UDB) 8.2 is the minimum level required for HADR
- Tivoli System Automation for MultiPlatforms, V1.2 (or higher). This is included with DB2 8.2 at no cost for use with DB2.

Architectural Alternatives

- **HADR Modes.** There are three modes of operation for HADR. Tradeoff is performance vs. potential transaction loss:
  1. Synchronous Mode: Log records written to disk on both primary and standby servers before application receives a successful return code to its commit
  2. Near Synchronous Mode: Log records written to disk on primary server are at least in memory on standby prior to notifying application of successful commit
  3. Asynchronous Mode: Log records written to disk on primary and passed to TCP/IP socket to be sent to standby prior to notifying application of successful commit
- **Cluster Management Software.** Some other cluster management software could be used rather than Tivoli System Automation. However, TSA is already included with DB2 V8.2, as are pre-canned automation scripts in support of HADR.
- **Database Software.** Oracle Data Guard is another version of a database mirroring solution, though with several functional differences compared to DB2 HADR (see white paper referenced at end of presentation)

Reference Architecture: Active/Active Application Server Cluster with Database Sharing on Linux

Flow

1. **Edge, HTTP, WebSphere** have already been covered and do not change in the scenario.
2. **Oracle Client (JDBC).** WebSphere runs the application and sends data requests to the Oracle Server. The Oracle JDBC thin client-side driver (JDBC Type 4) or the Oracle Call Interface (the JDBC type 2 driver plus the Oracle Client) contacts the Oracle database server.
3. **Oracle.** The servers are configured using Oracle RAC in an active/active configuration with 2 database servers. This means that each server is active and updates a common database file. Should one of the Oracle servers fail, the in-flight transactions are lost, but the other server in the RAC can receive all JDBC requests. The failing RAC node's cluster IP address fails over to a surviving node in the cluster to speed client reconnect.

Product Versions

- Oracle 9i or higher with the Real Application Clusters (RAC) option and appropriate client.

Architectural Alternatives

- **Active/Passive cluster.** A two node shared database cluster can be configured in Active/Passive mode, along with Oracle RAC Guard. This configuration has no load balancing, but failover will likely be faster.
- **Failover Mode.** There are options as to how RAC will manage failover
  - **Session vs SELECT failover.** Whether to simply failover sessions, or also failover in-progress queries (TYPE=SESSION/SELECT). Note that SELECT does not failover transactional statements, just queries.
  - **Use of pre-established backup connections**. Pre-establish connections to shadow processes on the backup node, **or** establish them at failover time. Speeds failover at the cost of extra memory usage on each node (METHOD=BASIC/PRECONNECT).

Reference Architecture: Active/Active Application Server with Database Sharing on z/OS in a Parallel Sysplex

Flow

1. **Edge, HTTP, WebSphere** have already been covered and do not change in the scenario.
2. **JDBC T4.** Each WAS server is configured to use the JDBC Type 4 driver. This JDBC driver is used to communicate with the DB2 z/OS data sharing group members. Once properly configured it is sysplex-aware and works cooperatively with DB2 and WLM on z/OS to balance workload across available members of the DB2 data sharing group. It will relocate work away from a failed or overloaded DB2 member to one that is up and underutilized on a transaction-by-transaction (rather than connection) basis.
3. **Sysplex Distributor.** On z/OS, Sysplex Distributor provides an initial contact single cluster IP address (known as a group Dynamic VIPA) for the data sharing group, which has built-in redundancy across network adapters and z/OS images. After initial contact, all subsequent communication is directly between the JDBC T4 client and the DB2 data sharing group members.
4. **DB2.** All DB2 group members in the Parallel Sysplex have equal access to a shared database. Should a DB2 group member fail, in-flight transactions to that DB2 will fail and be rolled back, but subsequent transactions are designed to be automatically rerouted to surviving DB2 group members. z/OS ARM (Automatic Restart Manager) may automatically restart a failed DB2 on the same or a different z/OS system (based on user-defined policy). If DB2 is restarted elsewhere, the sysplex-aware TCP/IP stacks can enable its dynamic VIPA address to automatically follow it to the new system.

Architectual Alternatives

- **DB2 Connect EE vs. JDBC T4.** DB2 Connect EE is also sysplex aware and can be used in place of the JDBC Type 4 driver. It can also be configured with a hot standby server that will take over in the event of a failure. However, using DB2 Connect introduces another layer, with the associated additional overhead and points of failure.
- **Sysplex Distributor Inclusion**. Sysplex distributor is only used for initial contact between the JDBC T4 driver and the DB2 data sharing group members. After that, it steps out of the picture.
  - It could be omitted and an initial DB2 member to contact could be pre-defined to the client instead. However, this requires that particular DB2 member must be active during initialization of the workload.
  - If a DNS has been placed within the Parallel Sysplex, DNS/WLM could also be used in place of Sysplex Distributor.

Product Versions

- JDBC Type 4 Driver sysplex support is available with JDBC driver 2.7 and above. This ships with DB2 UDB V8.2 fix pack 3, a.k.a. DB2 UDB V8.1 fix pack 10
  - To setup sysplex awareness please refer to the topic “DB2 Universal JDBC Driver connection concentrator and Sysplex workload balancing” in this link: http://publib.boulder.ibm.com/infocenter/db2luw/v8/topic/com.ibm.db2.udb.rn.doc/rn/r0012130.htm#wq575.
- DB2 Distributed Data Facility for z/OS V6 and V7 support for DVIPA and Dynamic DVIPA is provided through APAR PQ46659.
- Base Dynamic VIPA support is available with OS/390 V2R8 and higher.
- Base Sysplex Distributor support is available with OS/390 V2R10 and higher. Support for integration of Sysplex Distributor with Cisco MNLB function (not shown in this architecture) is available with z/OS V1R2 and higher. Fast connection reset support is available with z/OS V1R2 and higher.

Reference Architecture: Active/Active Application Server Cluster with Database Sharing on z/OS Across Cities

Flow (Single WAS Cell)

1. **Edge, HTTP, WebSphere, DB2 Connect, Sysplex Distributor, and DB2** have already been covered and do not change in the scenario.
2. **GDPS/PPRC V3.2.** All critical data resides on storage subsystem(s) in site 1 (the primary copy of data) and is mirrored to site 2 (the secondary copy of data) via synchronous **Peer-to-Peer Remote Copy** (PPRC). PPRC is designed to make it possible for a site switch with no data loss.
   The primary Controlling System (K1) running in Site 2:
   - Monitors the Parallel Sysplex cluster, coupling facilities, and storage subsystems and maintains GDPS status.
   - Manages a controlled site switch for outages of z/OS and Parallel Sysplex, z/VM, and Linux for zSeries (as a guest under z/VM).
   - Invokes **HyperSwap** on z/OS and z/VM for a site switch of disk subsystems, which can eliminate the need for an IPL at the recovery site to use the mirrored disks.
   - Works with Tivoli System Automation Multiplatform across z/VM and Linux to understand their state and coordinate their restart during the recovery phase.
   - Invokes network switching, based on user-defined automation scripts

Product Versions

- GDPS/PPRC Multiplatform requires IBM services to provide proper setup and education. It is not sold separately.

Flow (Two WAS Cells)

1. **Edge** flows do not change.
2. **HTTP, WebSphere** are clustered only within their own LPAR. Since each cell is independent, requests do not flow between LPARs.
3. **Sysplex Distributor, DB2, GDPS/PPRC** have already been covered and do not change in the scenario.

Pros (Two WAS Cells)

1. **Avoid communication latency** as stages of a request bounce between datacenters.
2. **Deployment Mgr** is available in the surviving cell.
3. **Simpler application version updates.** New app versions can be deployed in one cell, and if found to have problems, that cell can be fenced off and backed out while the other cell continued running the older app version.

Cons (Two WAS Cells)

1. **Session Affinity.** The routers must now maintain session affinity.
2. **Deployments must be done twice.** Since each cell is independent, all deployments and configuration changes must be made in each cell.

Notes:

- We accept the loss of HTTP session objects. Users having to login again is acceptable when we loose a datacenter.