IBM TXSeries for Multiplatforms Whitepaper

Preparing TXSeries systems for high availability on AIX

IBM TXSeries for Multiplatforms Whitepaper

Preparing TXSeries systems for high availability on AIX

Contents

About this document

This document reflects the IBM® TXSeries for Multiplatforms understanding of many of the questions asked about configuring high availability solutions for TXSeries product. This document explains high availability solutions with various practical scenarios to help you decide on the best suitable high availability option for your requirement.

This document is an attempt to provide a detailed guide into understanding various technologies and solutions to make a TXSeries system highly available and resilient to failures. The application of different solutions detailed in this paper is entirely dependent on the circumstances of use, system design and architecture, as well as your needs and requirements. For instance, if your system only has DPL-based applications, you can configure TXSeries Workload Manager in your system, to balance the load across multiple regions as well as creating an *always-available* processing environment. If you have a critical machine running your entire system including TXSeries, then utilizing PowerHA to make your system highly available should satisfy your needs. It is important to understand that the various examples provided in this paper are only samples which can be used as a template to design your own HA TXSeries environment. Some of the products mentioned in this document, such as PowerHA and Network Dispatcher are not part of TXSeries and are IBM products that are utilized to design highly available and reliable systems. Hence, it is important to understand their functions and uses before deciding the best suited highly-available solution for our system.

This document is presented "As-Is" and IBM does not assume responsibility for the statements expressed herein. These opinions are based on the authors' experiences. If you have questions about the contents of this document, please contact the authors:

- Raghavendran Srinivasan (raghavs1@in.ibm.com)
- Jithesh Moothoor (jmoothoo@in.ibm.com)
- Reshmi George (reshmi.ge@in.ibm.com)
- Govind Chakravarti (govchakr@in.ibm.com)

The authors would also like to express their thanks to:

- Gopalakrishnan. P, Lead Manager, TXSeries for Mutiplatforms and WebSphere eXtended Transaction Runtime
- Hariharan Venkitachalam, Product Architect, TXSeries for Mutiplatforms and WebSphere eXtended Transaction Runtime

Chapter 1. An overview on TXSeries, WLM and HA

This document explores the IBM PowerHA and TXSeries for Multiplatforms Workload Manager as a high availability solution applicable to TXSerieson AIX.

This document covers various configurations for creating high availability solutions involving TXSeries, depending on requirements and procedure to configure PowerHA and TXSeries Workload Manager. It is intended to help IT architects to plan and choose a high availability solution for TXSeries based on the application design framework and its high availability requirements. A working knowledge of TXSeries is required.

Prior to configuring a high availability solution on TXSeries, you might want to consider the following aspects:

- Which option will suit my requirement?
- What kind of client architecture. How does the region receive the requests?
- Whether a particular high availability technique will suit the application architecture?
- How do you want to distribute your CICS system?

High availability

High availability refers to an approach for a system to be operational continuously over a long time without disruption.

The system, in this case can be viewed as a combination of hardware, physical or logical servers and the applications with the capability to support failure takeover. A high availability solution ensures that the failure of any component, which is a part of the solution, does not cause the application and its data to become permanently unavailable.

The common way of achieving a high availability solution is either through *distributed computing environment* or through *on-demand computing environment*.

In a *distributed computing environment*, the load is distributed across multiple systems running in parallel by providing a transparency layer to the client. In such a setup, the systems in the background can be brought down for maintenance without affecting the throughput of the applications. In this kind of high availability design, the systems and applications are actively spread across multiple systems. The load is routed to other systems when a particular system in the environment is not available.

In an *on-demand computing environment*, one of the systems remain active and other remains on a standby or passive mode. When the active system goes down, the passive system becomes active and handles the incoming load until the original system is active and available.

The Figure 1 on page 2 describes the difference between a distributed computing system and an on demand system in a high availability environment. In the figure, the shared data storage can be a hard disk device or a database system.

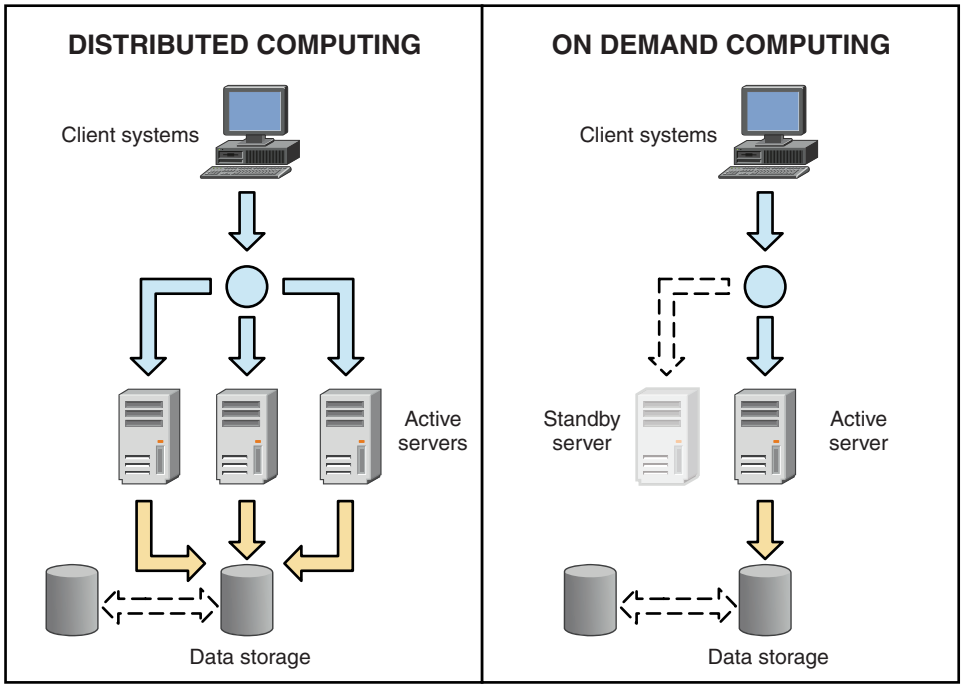


Figure 1. Distributed computing mechanism and on-demand computing environment

Note: The high availability of shared data storage is described in Figure 1 to depict completeness in the environment. High availability of shared storage is beyond the scope of this document.

TXSeries and High availability

TXSeries is a CICS® Online Transaction Processing (OLTP) environment for mixed language applications on distributed platforms. TXSeries is available on IBM AIX®, Windows, HP-UX, HP-IA, and Solaris.

TXSeries integrates the business software services required for online transaction processing applications. TXSeries can be configured to service thousands of parallel clients and provide increased scalability of Transactions per Second (TPS). You can use the high-performing, distributed transactional services of TXSeries in a stand-alone environment, or in support of larger mainframe and Java™ Enterprise Edition (Java EE) application deployments. TXSeries provides the Workload Management (WLM) utility for optimizing the distribution of application load and provide high availability in a CICS environment. WLM has the capability to maintain maximum throughput and ensure high availability of entities for processing the requests. TXSeries can also be configured with IBM PowerHA® to provide failover mechanism. Alternatively, PowerHA can be used in conjunction with WLM to provide high availability feature.

TXSeries Workload Manager

The TXSeries Work Load Manager is a utility that optimizes the distribution of tasks to multiple TXSeries regions capable of processing work requests.

The Work Load Manager has the following features:

- WLM ensures High availability of application processing regions (system that can always accept and execute tasks).

- Maintain maximum throughput and distribute application workload across TXSeries regions.

WLM ensures that requests are routed to ensure maximum throughput. As shown in Figure 2, the WLM listener communicates to regions through TCP. It receives messages from the regions during startup and shutdown, maintaining availability of regions. If the region has a planned or unplanned shutdown, WLM health monitor would identify the issue and list the region as unavailable.

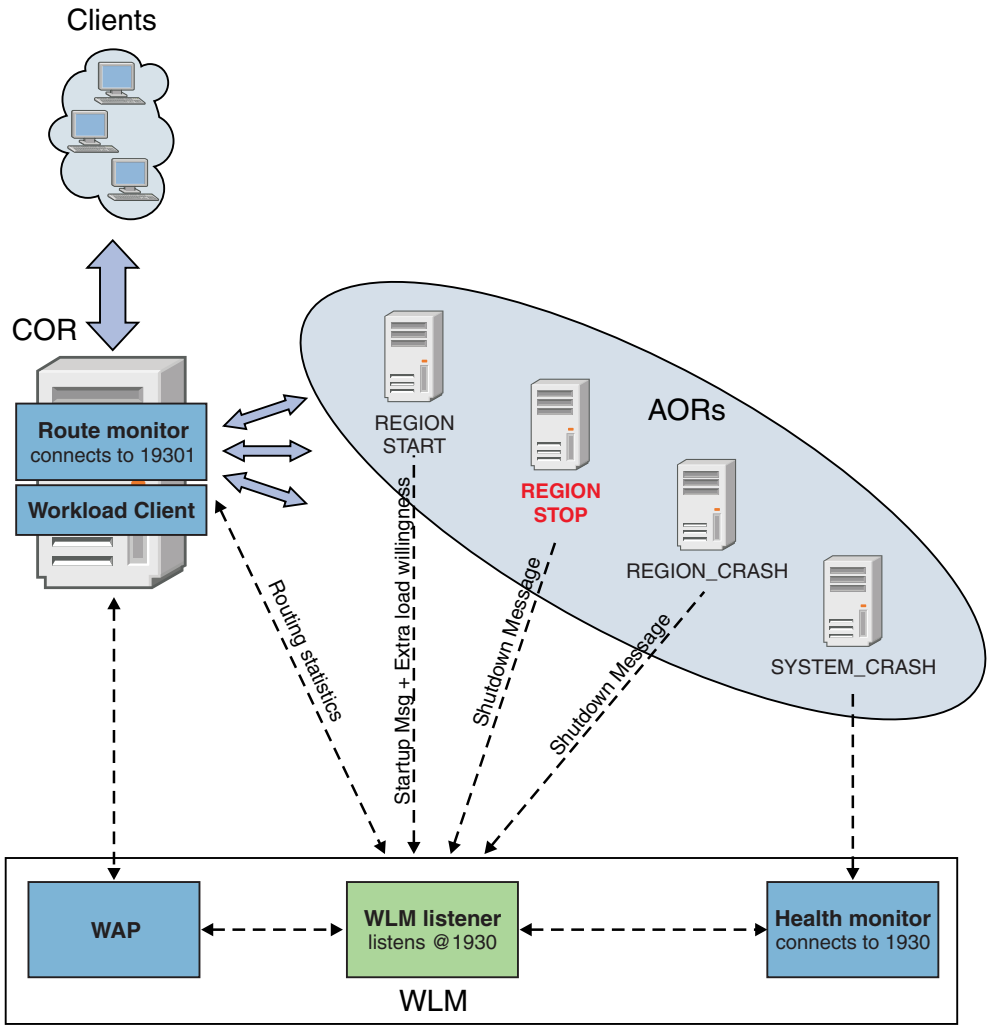


Figure 2. Dynamic load distribution by WLM

Requests are routed to available regions based on WLM configuration, after a region goes unavailable. Another key aspect of WLM is that if any of the application processing regions are slow in responding to transaction request or has higher or repeated rate of transaction failures, the WLM component will route subsequent requests to other application processing regions.

A common way to implement high availability through TXSeries workload manager is shown in Figure 3 on page 4. In TXSeries WLM, the requests reach the Client Owning region (COR). The COR receives the requests and routes the

requests to a configured Application Owning Region (AOR) when it encounters a Distributed Programming Link (DPL) or Distributed Transaction Routing (DTR). The AOR region executes the actual business application. In a WLM setup, all the AORs are usually identical with respect to the program definitions installed (cloned regions).

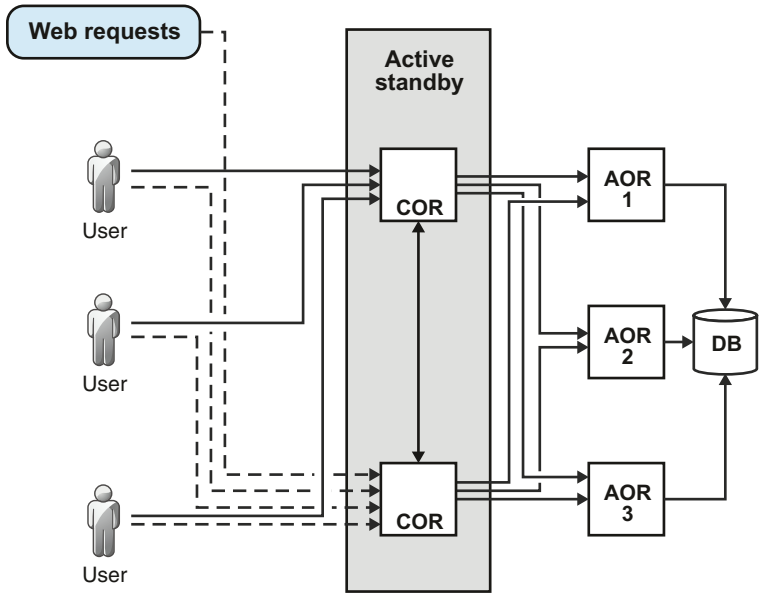


Figure 3. A common TXSeries Workload Manager setup with HA

In Figure 4 on page 5, a COR and AOR can be configured on a same AIX server or it can be spread across multiple servers based on the load and capacity of the server and it can even be run on other platforms on which TXSeries is supported. The COR and AOR can be configured to connect through various CICS ISC (Inter System Communication) protocols such as CICS TCP/IP (cics_tcp), PPC TCP/IP (ppc_tcp) and IPIC depending on the application architecture.

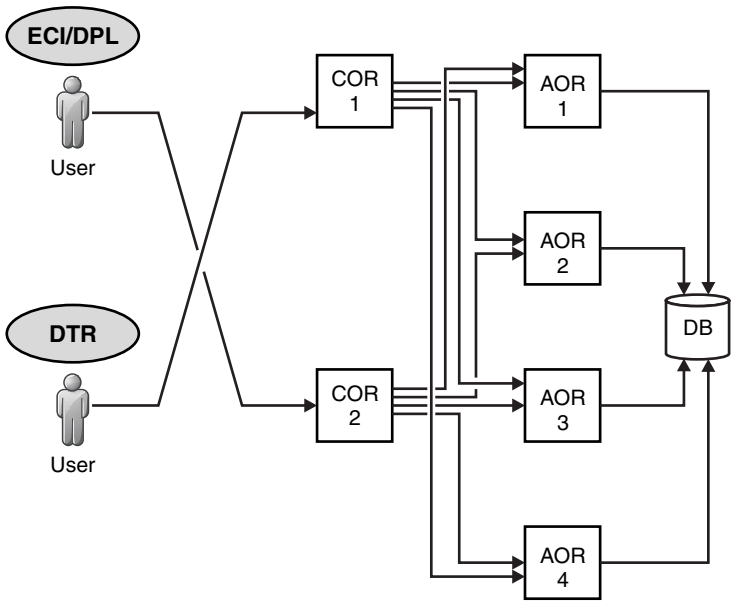


Figure 4. A common TXSeries Workload Manager setup

WLM is ideally suited for the high availability scenario where many instances/entities (AOR) can be made available for executing similar application. However, WLM has its limitations with respect to platform and application design. This section covers the benefits and limitations of using WLM for high availability solution.

WLM provides you the following benefits:

- WLM provides a highly available and customizable TXSeries environment.
- With the WLM setup, the individual systems can be brought down for maintenance and fix upgrades can be without having a downtime.
- Achieve a constantly high throughput. The throughput can be increased by increasing the AORs in the setup.
- WLM setup provides a fault tolerance for the applications with a minimum impact. For example, when a particular AOR in a WLM environment goes down, the unavailability of AOR is immediately detected and the new requests get routed to other AORs automatically. This enables the system to maintain a constant throughput.
- WLM provides customizations with respect to distribution of load across AOR regions. For example, if a particular AOR is placed on a powerful and capable of handling more requests, WLM can be configured to route more requests to that AOR.
- WLM understands the load on the various AORs and takes decisions automatically to distribute the load based on the capacity of an AOR.
- Provides a centralized monitoring capability through RMON. In TXSeries 8.1, a new command line time **cicswlmstat** is provided for monitoring WLM.

Requests are received by COR and routed to different AORs. WLM makes routing decisions by the characteristics of each AOR. Unavailability of one AOR can be adjusted by distributing requests to other AORs.

The TXSeries WLM however, has the following limitations:

- WLM can be used for terminal-based or DPL-based applications. It cannot be used in other forms of client communication supported by TXSeries such as DTP, Function shipping and Asynchronous program starts.
- WLM does not provide affinity to route transactions to a particular region. For example, if a Transaction A runs in region X and the Transaction B is required to run only in region X, then WLM will not be usable in such a case as Transaction B cannot be guaranteed to run on the same region X where Transaction A executed.

Chapter 2. Planning for high availability, HA scenarios and examples

Following are the prerequisites for configuring PowerHA and WLM with TXSeries:

- TXSeries for Multiplatforms Version 7.1 and TXSeries 8.1
- Power® HA Version 7.1.1.2 CF Base server Runtime
- AIX Version 7100-01

Scenarios to illustrate High Availability solutions

Scenario 1: IBM WebSphere® MQ as a load feeder to TXSeries

In this scenario, client requests are routed to TXSeries through IBM WebSphere MQ (WMQ). The TXSeries system waits on queue in WMQ and processes requests based on messages arriving in queue.

WMQ and TXSeries are tightly integrated. Figure 5 explains a possible high availability scenario in such a setup. Both WMQ and TXSeries are hosted in a PowerHA environment. The SYSTEM 1 serves the requests primarily. In case of a failure with SYSTEM 1 (active system), the SYSTEM 2 will be available and continue to serve new requests. SYSTEM 2 can be either in active/standby mode or active/active mode. Continuous availability of TXSeries regions are maintained in this manner.

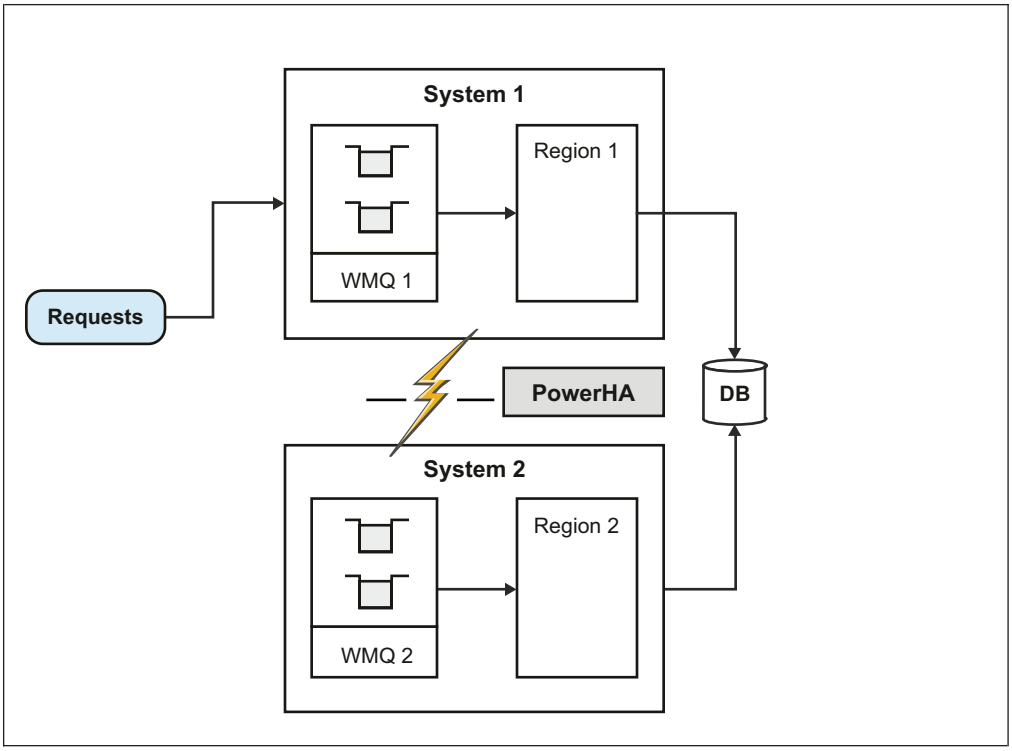


Figure 5. TXSeries handling requests from WMQ.

Note that in this case, high availability is on the system as a whole and not on WMQ or a region. The service will be affected if the region or WMQ goes down.

Scenario 2: IBM WebSphere as a Front-End to TXSeries

In this scenario, the region receives the requests through IBM WebSphere. The WebSphere may act as a server receiving requests from a browser and the business logic gets implemented in a TXSeries system.

The Figure 6 shows a possible high availability solution using TXSeries WLM for the scenario. Web requests are received by WebSphere and applications in TXSeries region are invoked through CICS Transaction Gateway (CTG). The CICS Transaction Gateway invokes the TXSeries side application through DPL. At the TXSeries side, WLM setup is made to provide high availability. The requests from CTG reach the COR and gets routed to appropriate AOR based on the availability. AORs can distribute requests across multiple machines to make the system more highly available. In this setup, region services are not impacted when an AOR goes down or the AOR is brought down for maintenance.

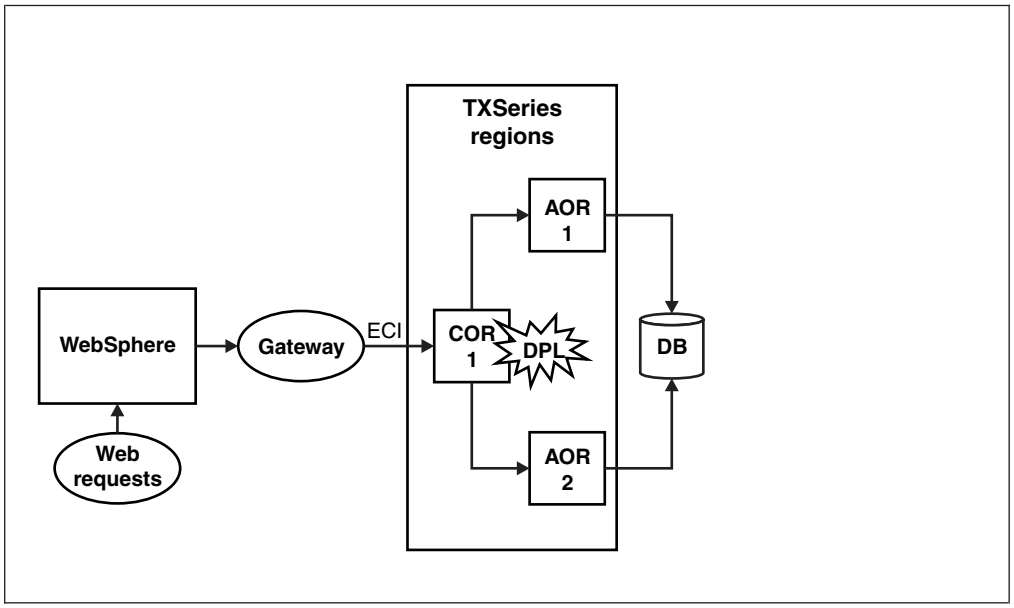


Figure 6. High Availability solution through TXSeries WLM for DPL applications

Scenario 3: Terminal applications invoked through TN3270 supported terminals

In this scenario, the region receives requests through TN3270 client for terminal emulation.

Figure 7 on page 9 shows a possible implementation of a high availability solution using WLM DTR implementation.

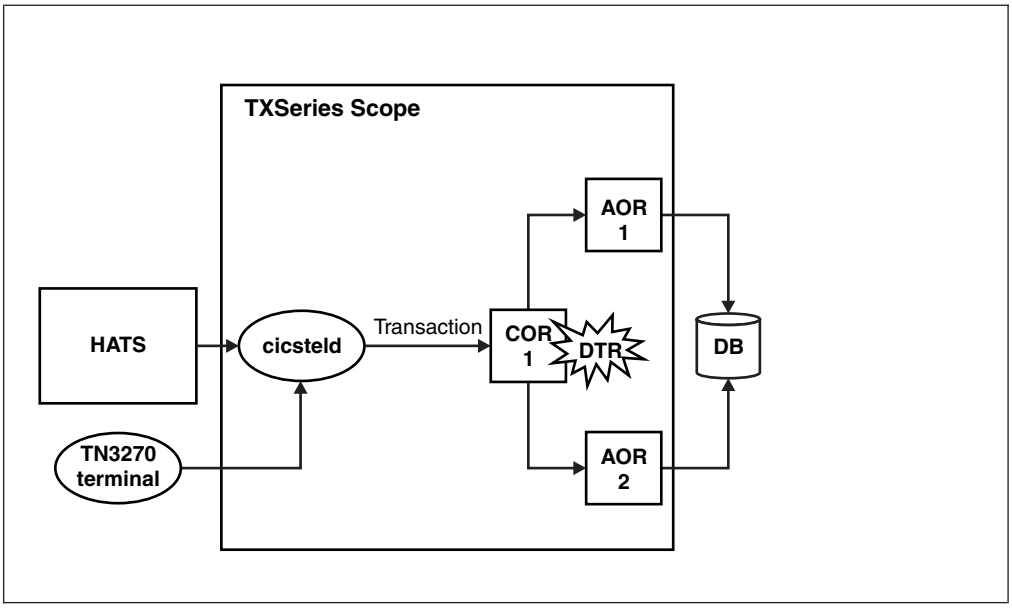


Figure 7. High availability TXSeries in a DTR environment.

Web requests are received by TN3270 terminals, and the transactions are invoked from TXSeries through cicsteld. The requests reach the COR region of TXSeries WLM and gets routed to appropriate AOR's based on the availability. AORs can distribute requests across multiple machines to make the system more highly available. In this scenario, the region services are not impacted when an AOR is not available or if the AOR is brought down for maintenance. The Rational® HATS software can be optionally configured as a terminal modernizer to reuse existing CICS Terminal screens. The IBM Rational Host Access Transformation Services (HATS) software acts as a bridge to interpret 3270 streams to web screens and vice versa.

An example of integrated High Availability system

A high availability solution is shown in Figure 8 on page 10. Business-critical applications are configured into a cluster which typically involves at least two systems (or nodes). The cluster monitors the critical resources for changes that may indicate a failure, a pending failure, or a possible configuration change. In this configuration, there are 2 CORs in two different servers and requests are sent to each of them. If one COR goes down, the other can take it up. All application requests will be routed through the Network Dispatcher machine. The Network Dispatcher is an optional, software load balancer and a component of IBM WebSphere Edge Server. The Network Dispatcher efficiently distributes requests between different CORs. The Network dispatcher is made highly available using PowerHA.

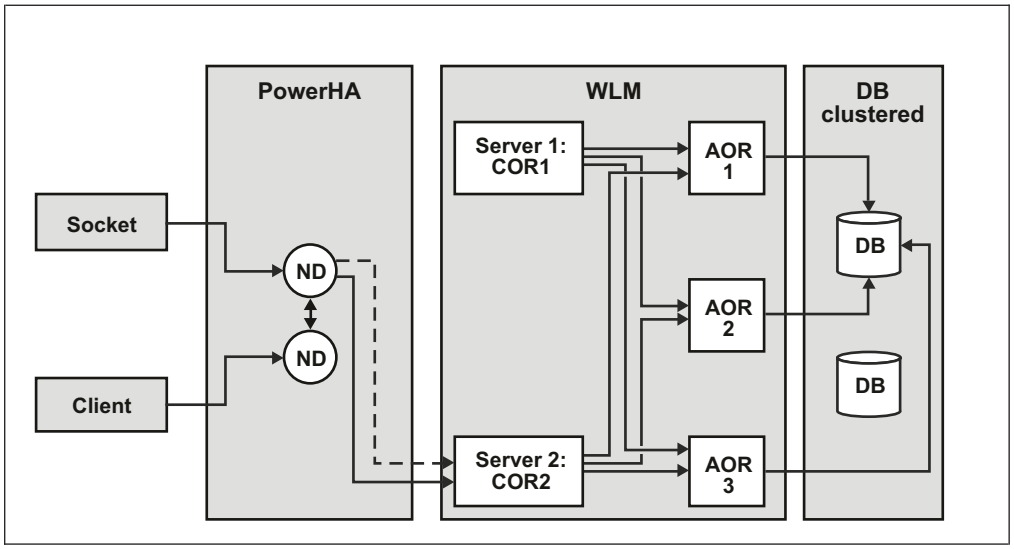


Figure 8. An integrated high availability system

Chapter 3. Configuring PowerHA and WLM with TXSeries

Configuring PowerHAwith TXSeries.

PowerHA ensures availability and avoids single point failure in hardware and software.

PowerHA achieves high availability through the use of redundant hardware, such as power supplies, network interfaces, SAN and mirrored or RAID disks. You need this specified hardware setup to use PowerHA. You need to update the `/etc/host` file before configuring the cluster. It is mandatory to include all interface labels/addresses of all nodes in the cluster which PowerHA refers in this file.

For example:

```
 /etc/hosts
###### Cluster IPs##########
##Boot IPs##
10.10.123.11    primary.in.ibm.com      primary
10.10.123.12    secondary.in.ibm.com    secondary

###Persistent IPs###
A.B.C.27    primarypip
A.B.C.250   secondarypip

###Application IP####
applicationip
```

The PowerHA Cluster configuration involves following steps:

1. Topology configuration
   a. Creating the cluster.
   b. Adding nodes into the cluster.
   c. Adding logical networks.
   d. Adding communication interfaces/devices that are to be part of the cluster.
2. Resource configuration
   a. Creating resources.
   b. Creating Resource Groups.
   c. Adding resources to be part of Resource Groups.

You can configure TXSeries in the PowerHA environment using either of the following modes:

- **Active–Standby mode configuration**

  The Active–Standby mode configuration is explained through Figure 9 on page 12. Node 1 is the backup node for Node 2. Node 1 is being used by other applications so failover cannot be disruptive to Node 1 users. Node 1 is neither running SFS, nor TXSeries region, but TXSeries software is installed on both nodes. Both Node 1 and Node 2 have two network interfaces (tr0 and tr1). The tr0 interface is the service adapter and tr1 is the standby adapter.

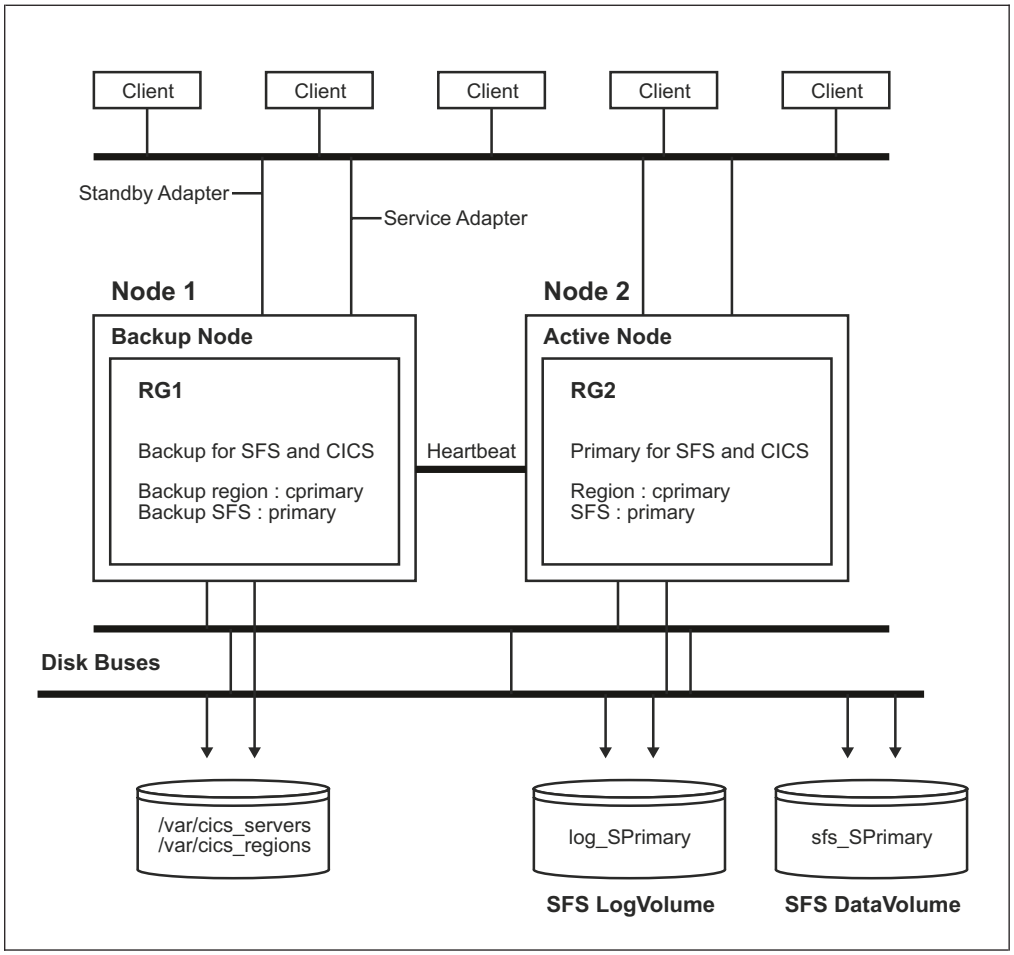


Figure 9. A sample CICS configuration on PowerHA active-standby setup

TXSeries configuration on active node (Node 2) involves the following steps:

1. Configure the TXSeries region and SFS on Node 2. In this example, the TXSeries region name is listed as `cprimary` and SFS name is `/.:/cics/sfs/primary`. The hostname of active node is Node 2.

   Issue command:

   ```
   cicscp –v create sfs_server primary
    cicscp –v  create region cprimary DefaultFileServer="/.:/cics/sfs/primary"
   ```

   To run TXSeries and the SFS on both nodes, certain file systems and volumes need to be shared. You must have a separate shared file system for `/var/cics_regions`, `/var/cics_servers`, and shared raw logical volumes for the SFS.
2. Complete the following steps to add SFS logical volumes in network shared disk.
   a. Remove logical volumes `sfs_Sprimary` and `log_Sprimary` from system `/dev` directory which is created by default with the **`cicscp –v create sfs_server`** command.
   b. Verify shared disk attached to the system. Issue the **`#lsdev | grep disk`** command to verify the shared disk attached to your system. The following example lists the external disks as hdisk:

```
#lsdev | grep disk
   hdisk0    Defined     Virtual SCSI Disk Drive
   hdisk1    Available   Virtual SCSI Disk Drive
   hdisk2    Available   Virtual SCSI Disk Drive
   hdisk3    Avaialble   Virtual SCSI Disk Drive
```

The external disk has been installed on the primary and backup machine using the command.

3. Create a network shared volume with the external disk by completing the following steps:
   a. Create an external volume group on the external disk. Issue the **smitty mkvg** command.

```
      Add an Original Volume Group

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                                      [Entry Fields]

* VOLUME GROUP name                                   [txseriesvg]
* Physical PARTITIONS SIZE in megabytes
  PHYSICAL VOLUME names                               [hdisk3]
  FORCE the creation of volume groups ?               no
  Activate volume group AUTOMATICALLY                 yes
        at system restart ?
Volume group MAJOR NUMBER                             [60]
Create VG Concurrent Capable ?                        no
Infinte Retry Option                                  no

F1=Help              F2=Refresh           F3=Cancel         F4=List
F5=Reset             F6=Command           F7=Edit           F8=Image
F9=Shell             F10=Exit             Enter=Do
```

   The new volume group is **txseriesvg**, the disk devices to be included is hdisk3, and a major number is to be assigned to it. It is also important to specify that the volume group should not be activated (varied on) automatically at system restart. The varyon of shared volume groups need to be under control of high-availability software (PowerHA), so that it is coordinated correctly.

   b. Vary on the volume group just created: **varyonvg txseriesvg**.
   c. Before creating file systems on the shared disk resources, you need to create the `jfslog` logical volume. You need to provide it a unique name, as it is used on all nodes in the cluster to refer to the same log. This method will avoid naming conflicts that might arise between nodes in the cluster.
   d. Add logical volume with `txseriesvg` volume group. Use smitty **mklv** to add the logical volumes `loglv_txseries` for the file systems in a volume group `txseriesvg`.

```
      Add a Logical Volume

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

[TOP]                                                [Entry Fields]
  Logical Volume NAME                                [loglv txseries]
* VOLUME GROUP name                                  txseriesvg
* Number of LOGICAL PARTITIONS                       [1]
  PHYSICAL VOLUME names                              [hdisk3]
  Logical volume TYPE                                [jfslog]
  POSITION on physical volume                        outer_middle
  RANGE of physical volumes                          minimum
  MAXIMUM NUMBER of PHYSICAL VOLUMES                 []
  to use for allocations
Number of COPIES of each logical                     1
  partition
Mirror Write Consistancy ?                           active
Allocate each logical partition copy                 yes
on a SEPEARTE physical volume ?
RELOCATE the logical volume during reorganization ?  yes
Logical volume LABEL                                 []
MAXIMUM NUMBER  og LOGICAL PARTITIONS                [512]
Enable BAD BLOCK relocation ?                        yes
SCHEDULING POLICY for writing/reading                parallel
logical partition copies
[MORE.....8]

F1=Help              F2=Refresh           F3=Cancel         F4=List
F5=Reset             F6=Command           F7=Edit           F8=Image
F9=Shell             F10=Exit             Enter=Do
```

e. After creating the `jfslog` logical volume, ensure that you format the logical volume with the following command:

```
# logform /dev/loglv_txseries
    logform: destroy /dev/loglv_txseries (y)?
```

To destroy old log files, you need to answer yes (y) at the prompt.

f. Create SFS logical volumes (`sfs_Sprimary` and `log_Sprimary`) using volume group, **`txseriesvg`**.

Note: You can also create `log_Sprimary` logical volume using the same method.

```
     Add a Logical Volume

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

[TOP]                                                [Entry Fields]
  Logical Volume NAME                                [sfs_Sprimary]
* VOLUME GROUP name                                  txseriesvg
* Number of LOGICAL PARTITIONS                       [10]
  PHYSICAL VOLUME names                              [hdisk3]
  Logical volume TYPE                                []
  POSITION on physical volume                        outer_middle
  RANGE of physical volumes                          minimum
  MAXIMUM NUMBER of PHYSICAL VOLUMES                 []
  to use for allocations
Number of COPIES of each logical                     1
  partition
Mirror Write Consistancy ?                           active
Allocate each logical partition copy                 yes
on a SEPEARTE physical volume ?
RELOCATE the logical volume during reorganization ?  yes
Logical volume LABEL                                 []
MAXIMUM NUMBER  og LOGICAL PARTITIONS                [512]
Enable BAD BLOCK relocation ?                        yes
SCHEDULING POLICY for writing/reading                parallel
logical partition copies
[MORE.....8]

F1=Help              F2=Refresh           F3=Cancel         F4=List
F5=Reset             F6=Command           F7=Edit           F8=Image
F9=Shell             F10=Exit             Enter=Do
```

Verify the logical volumes in **txseriesvg** volume group.

```
# lsvg -l txseriesvg
txseriesvg
LV NAME          TYPE      LPs   PPs   PVs      LV STATE      MOUNT POINT
loglv_txseries   jfslog    1     1     1      closed/syncd     N/A
log_Sprimary      jfs      10    10    1      closed/syncd     N/A
sfs_Sprimary      jfs      20    20    1      closed/syncd     N/A
```

You need to create two separate logical volumes to mount the two file systems that are used by TXSeries:

– The logical volume `sfsvar`, for the `/var/cics_servers` file system.

– The logical volume `cicsvar`, for the `/var/cics_regions` file system.

This method will help you use the `/var/cics_servers` and `/var/cics_regions` for both nodes configured in PowerHA. This can maintain the data integrity between systems during fail over cases.

To create logical volumes with name `sfsvar` and `cicsvar`, go through 3f on page 14.

Create a file system on the logical volume that was created from the listed step.

```
      Add a standard Journaled File System

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

[TOP]                                                  [Entry Fields]
* LOGICAL VOLUME name                                  sfsvar
* MOUNT POINT                                          [var/cics servers]
  Mount AUTOMATICALLY st system restart?               no
  PERMISSIONS                                          read/write
  Mount OPTIONS                                        []
  Start Disk  Accounting ?                             no
  Fragment Size (bytes)                                4096
  Number of bytes per inode                            4096
  Allocation Group Size (MBytes)                       8
  Logical Volume for Log
  Mount GROUP          []



F1=Help              F2=Refresh           F3=Cancel          F4=List
F5=Reset             F6=Command           F7=Edit            F8=Image
F9=Shell             F10=Exit             Enter=Do
```

```
#lsvg -l txseriesvg
  txseriesvg:
  LV NAME  TYPE LPs PPs PVs LV STATE MOUNT POINT
  loglv_txseries jfslog 1 1  1 closed/syncd N/A
  log_Sprimary jfs  10 10    1 closed/syncd N/A
  sfs_SPrimary jfs  20 20    1 closed/syncd N/A
  sfsvar     jfs  10 10    1 closed/syncd /var/cics_servers
  cicsvar     jfs  10 10    1 closed/syncd /var/cics_regions
```

4. Start the region and SFS on Node 2. Complete the following steps:
   a. Use **cicscp –v start sfs_server primary StartType=cold** command to start SFS on the Node2 machine.
   b. Start the region using the following command on Node 2 machine:
      ```
      cicscp –v start region cprimary StartType=cold
      ```
   c. Check the status of region and SFS using the **cicscp -v status all** command.

CICS configuration on backup node (Node 1) involves the following steps:

1. Stop the region and SFS on Node 2. Run the following command:
   ```
   cicscp –v stop region cprimary
       cicscp –v stop sfs_server primary
   ```
2. Unmount the file systems /var/cics_regions and /var/cics_servers from Node 2. Run the command:
   ```
   umount /var/cics_regions
    umount /var/cics_servers
   ```
3. Varryoff the volume group txseriesvg on Node 2. Issue the **varyoffvg txseriesvg** command.
4. Import the volume group txseriesvg on Node 1. Issue command **smitty importvg**.

```
     Import a Volume Group

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                        [Entry Fields]
  VOLUME GROUP name                     [txseriesvg]
* PHYSICAL VOLUME name                  [hdisk3]
  Volume group MAJOR NUMBER             []




F1=Help              F2=Refresh          F3=Cancel         F4=List
F5=Reset             F6=Command          F7=Edit           F8=Image
F9=Shell             F10=Exit            Enter=Do
```

5. Vary on the volume group txseriesvg on Node 1 using the **varyon** command. Mount /var/cics_regions and /var/cics_servers file systems using the mount command:

   ```
   varyon txseriesvg
       mount /var/cics_regions
       mount /var/cics_servers
   ```

6. Create region and SFS on the server Node1. Issue the commands:

   ```
   cicscp –v create sfs_server primary
    cicscp –v create region cprimary
   ```

7. Ensure that SFS and region can be started on Node1, by running the following commands:

   ```
    cicscp –v start sfs_server primary
     cicscp –v start region cprimary
   ```

- **Active-Active mode configuration**

  The Active-Active mode configuration is explained through Figure 10 on page 18.

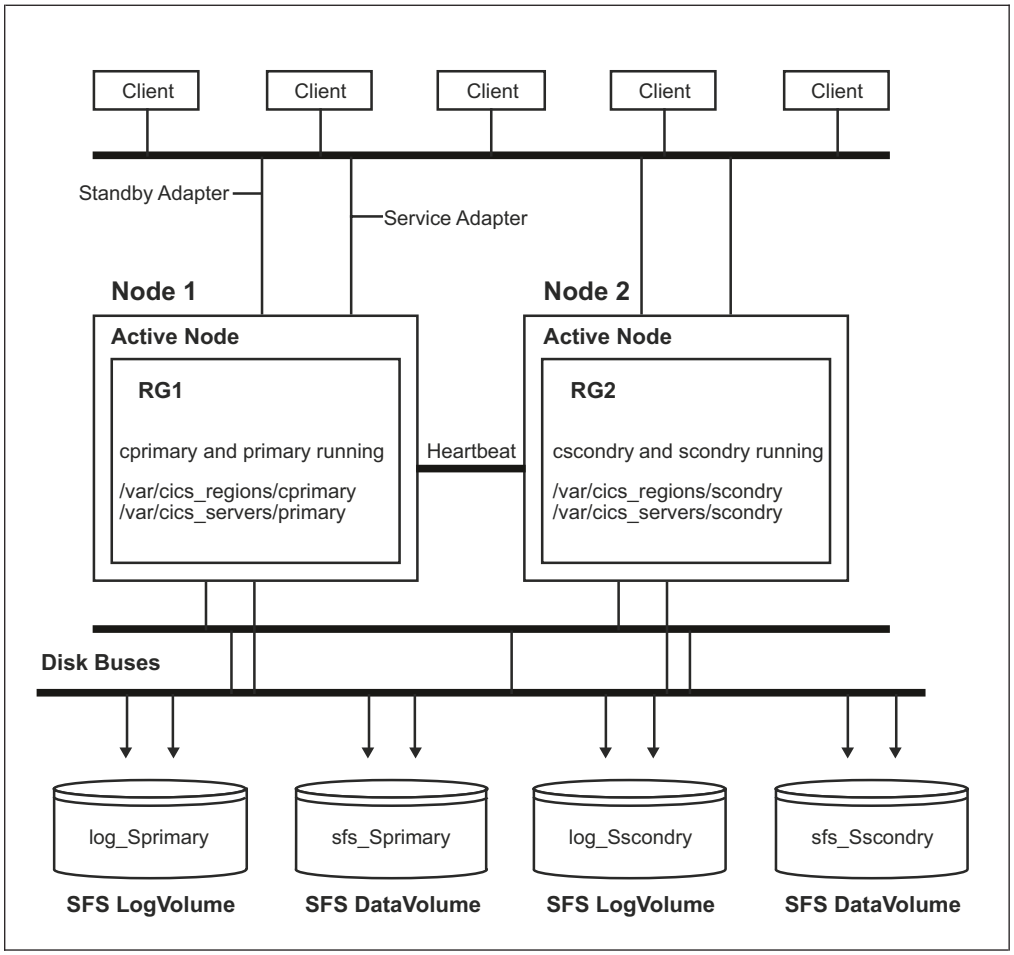


Figure 10. A sample CICS configuration on PowerHA active-active mode setup

The configuration is explained through the following steps:

1. Configure the TXSeries region and SFS on Node1 and Node2. For example:
   - TXSeries region name is `cprimary` and SFS name is `/.:/cics/sfs/primary`. The hostname of active node is Node1.
   - TXSeries region name is `cscondry` and SFS name is `/.:/cics/sfs/scondry`. The hostname of active node is Node2.

   To configure the region and SFS with the listed names, issue the following command:

   ```
   cicscp –v create sfs_server primary
   cicscp-v  create region cprimary DefaultFileServer="/.:/cics/sfs/primary
   cicscp –v create sfs_server scondry
   cicscp -v create region cscondry DefaultFileServer="/.:/cics/sfs/scondry
   ```

   To run TXSeries and the SFS on both nodes, ensure that you keep primary and secondary SFS logical volumes in network shared disk.

   From operating system perspective, mounting the same file system on two servers is not feasible. In order to share the `cics_servers` and `cics_regions` on both LPARs taking part on PowerHA, you need to introduce General Parallel File System (GPFS). This can be mounted on both the LPARs to share `cics_servers` and `cics_regions` folder. If not, you need to keep these directories in local file system on Node1 and Node2 respectively.

To keep the cics_servers and cics_regions directories in local file system, proper management of these directory logs is required. Integrity needs to be maintained between systems.

Note: Details on configuring GPFS is beyond the scope of this document. For explanation purpose, we mention the cics_servers and cics_regions in local filesystem on Node1 and Node2.

2. Complete the following steps to create SFS logical volumes for SFS primary (sfs_Sprimary and log_Sprimary) and scondry (sfs_Sscondry and log_Sscondry), in network shared disk:
   a. Remove the logical volumes sfs_Sprimary and log_Sprimary from system /dev directory that is created by default with the **cicscp –v create sfs_server** command.
   b. Verify the shared disk attached to the system. Issue the following AIX command to verify the shared disk attached to your system. The listed example uses the external disk as hdisk.
   c. The external disk has been installed on the primary and backup machine through the listed command.
3. Create a network shared volume with external disk by completing the following steps:
   a. Create an external volume group on the external disk. Issue the **smitty mkvg** command.

```
      Add an Original Volume Group

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                                      [Entry Fields]

* VOLUME GROUP name                                   [txseriesvg]
* Physical PARTITIONS SIZE in megabytes
  PHYSICAL VOLUME names                               [hdisk3]
  FORCE the creation of volume groups ?               no
  Activate volume group AUTOMATICALLY                 yes
       at system restart ?
Volume group MAJOR NUMBER                             [60]
Create VG Concurrent Capable ?                        no
Infinte Retry Option                                  no



F1=Help             F2=Refresh          F3=Cancel         F4=List
F5=Reset            F6=Command          F7=Edit           F8=Image
F9=Shell            F10=Exit            Enter=Do
```

   Provide name of the new volume group txseriesvg, the disk devices to be included hdisk3, and the major number to be assigned to it. It is also important to specify that we do not want the volume group activated (varied on) automatically at system restart. Later the varyon of shared volume groups has to be under the control of the high-availability software (PowerHA), so it is coordinated correctly.
   b. Vary on the volume group just created:

```
varyonvg txseriesvg
```

   c. Before creating file systems on the shared disk resources, create the jfslog logical volume. Provide it with a unique name that might be used on all nodes in the cluster to refer to the same log. This will avoid name conflicts that might arise between nodes in the cluster.

d. Add logical volume with txseriesvg volume group. Use **smitty mklv** to add the loglv_txseries logical volumes for file systems in volume group txseriesvg.

```
      Add a Logical Volume
Type or select values in entry fields.
Press Enter AFTER making all desired changes.

[TOP]                                                 [Entry Fields]
  Logical Volume NAME                                 [loglv_txseries]
* VOLUME GROUP name                                   txseriesvg
* Number of LOGICAL PARTITIONS                       [1]
  PHYSICAL VOLUME names                               [hdisk3]
  Logical volume TYPE                                 [jfslog]
  POSITION on physical volume                         outer_middle
  RANGE of physical volumes                           minimum
  MAXIMUM NUMBER of PHYSICAL VOLUMES                  []
  to use for allocations
Number of COPIES of each logical                      1
  partition
Mirror Write Consistancy ?                            active
Allocate each logical partition copy                  yes
on a SEPEARTE physical volume ?
RELOCATE the logical volume during reorganization ?   yes
Logical volume LABEL                                  []
MAXIMUM NUMBER  og LOGICAL PARTITIONS                 [512]
Enable BAD BLOCK relocation ?                         yes
SCHEDULING POLICY for writing/reading                 parallel
logical partition copies
[MORE.....8]

F1=Help              F2=Refresh           F3=Cancel         F4=List
F5=Reset             F6=Command           F7=Edit           F8=Image
F9=Shell             F10=Exit             Enter=Do
```

e. After creating the jfslog logical volume, format the logical volume with the following command:

```
# logform /dev/loglv_txseries
            logform: destroy /dev/loglv_txseries (y)?
```

Answer yes (y) to the prompt on whether to destroy the old version of the log.

f. Create SFS logical volumes (sfs_Sprimary and log_Sprimary) using the txseriesvg volume group.

```
      Add a Logical Volume
Type or select values in entry fields.
Press Enter AFTER making all desired changes.

[TOP]                                                  [Entry Fields]
  Logical Volume NAME                                  [sfs_Sprimary]
* VOLUME GROUP name                                    txseriesvg
* Number of LOGICAL PARTITIONS                         [10]
  PHYSICAL VOLUME names                                [hdisk3]
  Logical volume TYPE                                  []
  POSITION on physical volume                          outer_middle
  RANGE of physical volumes                            minimum
  MAXIMUM NUMBER of PHYSICAL VOLUMES                   []
  to use for allocations
Number of COPIES of each logical                       1
  partition
Mirror Write Consistancy ?                             active
Allocate each logical partition copy                   yes
on a SEPEARTE physical volume ?
RELOCATE the logical volume during reorganization ?    yes
Logical volume LABEL                                   []
MAXIMUM NUMBER  og LOGICAL PARTITIONS                  [512]
Enable BAD BLOCK relocation ?                          yes
SCHEDULING POLICY for writing/reading                  parallel
logical partition copies
[MORE.....8]

F1=Help              F2=Refresh           F3=Cancel         F4=List
F5=Reset             F6=Command           F7=Edit           F8=Image
F9=Shell             F10=Exit             Enter=Do
```

g. Go through the step 3f on page 20 to create `log_Sprimary`, `log_Sscondry` and `sfs_Sscondry` logical volumes.

h. Verify the logical volumes in `txseriesvg` volume group:

```
# lsvg -l txseriesvg
       txseriesvg
       LV NAME          TYPE      LPs   PPs   PVs      LV STATE      MOUNT POINT
       loglv_txseries   jfslog    1     1     1      closed/syncd     N/A
       log_Sprimary      jfs      10    10    1      closed/syncd     N/A
       sfs_Sprimary      jfs      20    20    1      closed/syncd     N/A
       log_Sscondary     jfs      10    10    1      closed/syncd     N/A
       sfs_Sscondary     jfs      20    20    1      closed/syncd     N/A
```

4. Start region and SFS on Node1 and Node2. Complete the following substeps:

   a. Use **cicscp –v start sfs_server primary StartType=cold** command to start SFS on the Node1 machine.

   b. Start region using **cicscp –v start region cprimary StartType=cold** command on Node1 machine.

   c. Use the **cicscp –v start sfs_server scondry StartType=cold** command to start SFS on the Node2 machine.

   d. Start region using the **cicscp –v start region cscondry StartType=cold** command on Node2 machine.

   e. Check status of the region and SFS using **cicscp –v status all** command.

Creating a custom script to run TXSeries during failover

Virtually any application that can run on a standalone AIX system can run in a clustered environment protected by PowerHA. The application must be able to be started and stopped by scripts as well be able to be recovered by running a script after an unexpected shutdown.

CICS is an application which is defined to PowerHA as application servers with the following attributes:

- The *start script* should start CICS regions and SFS from both clean and an unexpected shutdown. Output from the script will be logged in the `hacmp.out` log file. The exit code from the script will be monitored by PowerHA. An example start script is as follows:

```
export DB2DBDFT=cicstest
export COBDIR=/opt/microfocus/cobol
export DB2INSTANCE=db2inst1
export CICS_TK_TRACE_REDIRECT=trace=FILE:/tmp/server_trace.out
/usr/lpp/cics/bin/cicscp -v start sfs_server primary
/usr/lpp/cics/bin/cicscp -v start region cprimary StartType=auto
```

- The *stop script* must be able to successfully stop the CICS regions and SFS. Output is also logged in `hacmp.out` and the exit code will be monitored. PowerHA is able to monitor the application itself, not just the required resources. An example stop script is as follows:

```
/usr/lpp/cics/bin/cicscp -v stop sfs_server primary
/usr/lpp/cics/bin/cicscp -v stop region cprimary
```

The full path name of the script must be same on all nodes. However, the contents of the script itself can be different from node to node. If they differ on nodes, it will inhibit your ability to use file collections feature. It is recommended that you have an intelligent script that can determine which node it is running on and start up appropriate. Verify the list of application servers configured including start/stop scripts using below commands:

```
# ./usr/es/sbin/cluster/utilities/cllsserv
 cicsapplication /opt/txseries/startcics.sh  /opt/txseries/stopcics.sh background
```

Verifying the failover of TXSeries systems

You can verify the failover of TXSeries systems by completing the following steps:

1. Create a custom script in PowerHA to start and stop the TXSeries during its failover. This script will automatically start the SFS and region during the failover of resource group from active node to backup node.

```
      Change/show Application Controller Scripts

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                        [Entry fields]
Application Cntroller Name              txseries_controller
New Name                                [txseries_controller]
Start Script                            [/opt/txseries/startcics.sh]
Stop Script                             [/opt/txseries/startcics.sh]
Application Monitor Name(s)
Application startup mode                [background]


F1=Help             F2=Refresh          F3=Cancel         F4=List
F5=Reset            F6=Command          F7=Edit           F8=Image
F9=Shell            F10=Exit            Enter=Do
```

2. The following screen explains the TXSeries resource group characteristics. This will help you understand the resource group configured for TXSeries.

```
      Change/Show All Resources and Attributes for a Custom Resource Group
Type or select values in entry fields.
Press Enter AFTER making all desired changes.

[TOP]                                                [Entry Fields]
  Resource Group NAME                                txseries_resources
* Participating Node (Default Node Priority)         primary secondary

  Startup Policy                                     Online on Home Node Only
  Failover Policy                                    Failover to the Next Priority Node In Th>
  Fallback Policy                                    Never Fallback

  Service IP Labels/Addresse                         [applicationip]
  Application Controllers                            [txseries_controller]

  Volume Groups                                      [txseriesvg]
  Use forced varyon of volume groups, if neccessary  false
  Automatically Import Volume Groups                 false

  Filesystems (empty is ALL for VGs specified)       []
  Filesystems Consistency Check                      fsck
  Filesystems Recovery method                        sequential
  Filesystems mounted before IP configured           false

  Filesystems/Directories to Export(NFSv2/3)         []
  Filesystems/Directories to Export(NFSv4)           []
  Stable Storage Path (NFSv4)                        []
  Filesystems/Directories to NFS Mount               []
[MORE.....11]

F1=Help              F2=Refresh           F3=Cancel         F4=List
F5=Reset             F6=Command           F7=Edit           F8=Image
F9=Shell             F10=Exit             Enter=Do
```

3. Manually test the fallover policy in PowerHA using the following substeps:
   a. Start Cluster services:

      ```
      Smitty sysmirror
      ```

      System Management (C-SPOC) > **PowerHA SystemMirror Services** > **Start Cluster Services**

      The PowerHA and TXSeries startup logs are available at `/var/hacmp/log/hacmp.out`.

      ```
            Start Cluster Services

      Type or select values in entry fields.
      Press Enter AFTER making all desired changes.

                                              [Entry fields]
      Start now, on system restart or both    now
      Start Cluster Services on these nodes   [primary,secondary]
      Manage Resource Groups                  Automatically
      Broadcast message at startup?           false
      Startup Cluster Information Daemon?     false
      Ignore verification errors              false
      Automatically correct errors found      Interactively
      during cluster start?


      F1=Help             F2=Refresh          F3=Cancel         F4=List
      F5=Reset            F6=Command          F7=Edit           F8=Image
      F9=Shell            F10=Exit            Enter=Do
      ```

   b. Move the resource group from active node to backup node using the following command. This will help you verify the PowerHA configuration on Node1 and Node2.

      smit sysmirror > **C-SPOC** > **Resource Groups and Applications** > **Move Resource Groups to Another Node**

Verify the /var/hacmp/log/hacmp.out file to know whether that the resource group moved to backup node.

Chapter 4. Configuring Workload Manager with TXSeries

You can configure a Workload Manager (WLM) environment with TXSeries for high availability as listed.

Ensure that the following prerequisites are in place:

- Access to the *cicssm* user (TXSeries V7.1). WLM configuration and administration commands can only be executed by **cicssm** user. From TXSeries V8.1, it can be any user in *cicssm* OS group.
- Root access is required to be able to create TXSeries regions, configure and start them. In TXSeries V8.1, a non root user can administer TXSeries region. The user has to be a part of cics operating system group.

Setting up the WLM environment

Setting up a WLM environment consists of two aspects that require configuration. These are:

- *WLM design configuration*: This specifies a schema of the environment for WLM to perform effective routing and workload balancing.
- TXSeries region configuration.

The listed steps will work on TXSeries V7.1 and TXSeries V8.1. In TXSeries V8.1, you can also easily configure WLM through WLM IVP sample provided in `/usr/lpp/cicssm/samples/ivp` directory. The directory has sample WLM configuration with a Client Owning Region (COR) and three Application Owning Region (AOR). You can modify the IVP sample to create a WLM configuration of choice.

The WLM environment being configured consists of the following:

- Four different physical systems. (Machine 1, 2, 3, and 4)
- Two CORs. Every COR accepts client requests and re-routes them (based on WLM logic) to an appropriate AOR. Each COR is on a different system. (Machine 1 and 2)
- Four AORs. Everey AOR reside on a different system. (Machine 1, 2, 3 and 4). AOR1 and AOR2 reside on the same systems as COR1 and COR2 respectively.
- All four AORs are considered for *load balancing* by both the CORs.
- AORs are connected to a database. (This is not part of the WLM environment).

A sample COR and AOR configuration is depicted in Figure 11 on page 26.

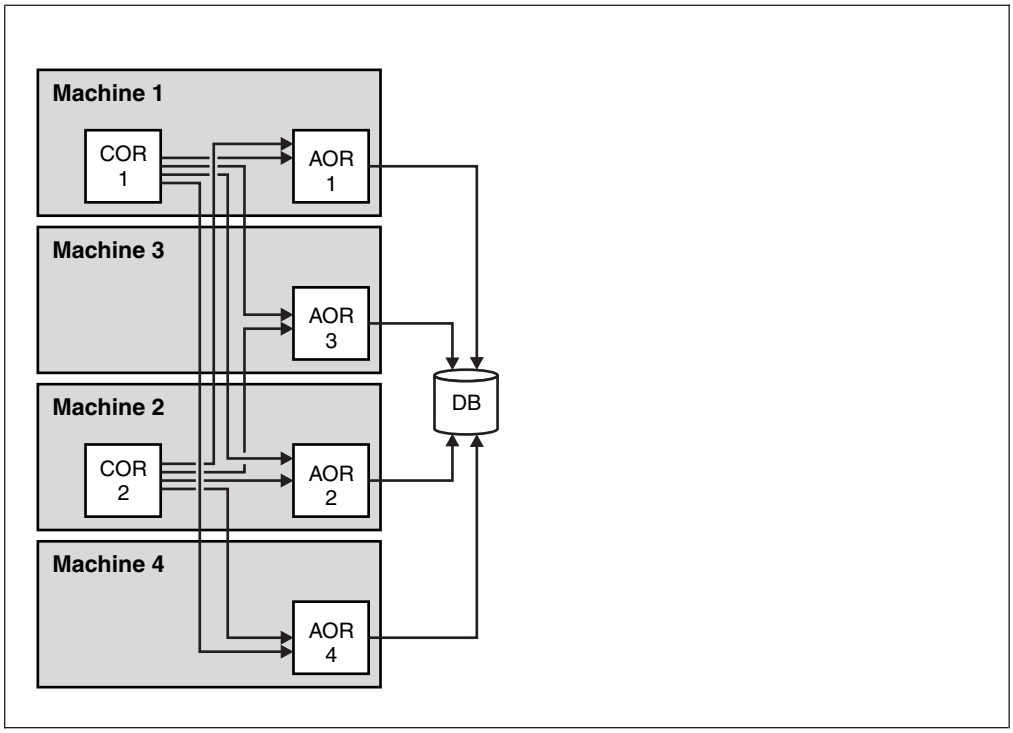


Figure 11. A sample COR and AOR configuration

The WLM configuration requires two artifacts that need to be created. They are:

- WLM configuration file. For example `wlm1.cfg`.
- WAP configuration file. For example `wap.wlm1`.

The files reside in the WLM repository directory in `/var/cicssm/repos`. The WLM configuration is explained based on the following assumptions:

- WLM will run on system: `mac1`. Hence you can do the configuration on `mac1`.
- Configuration name: wlm1
- System names (hostnames): `mac1`, `mac2`, `mac3`, `mac4`.
- TXSeries region names: `COR1`, `COR2`, `AOR1`, `AOR2`, `AOR3`, `AOR4`.
- Program names (which will be called by client): `PRG1`, `PRG2`.
- Transaction name: `TRAN`.

Configuring the WLM plex

To configure the WLM configuration file, you need to use the **`cicswlmcfg`** tool. The tool creates and validates a WLM configuration file. The various steps are explained in detail:

1. Creating the *plex*.

   Plex is logical representation of the entire WLM configuration. To create a plex, run:

   ```
   cicswlmcfg create plexdb wlm1
   ```

   This will create the configuration file, `wlm1.cfg`.

   To create plex *plx1*, run the command:

   ```
   cicswlmcfg create plex plx1 -d wlm1
   ```

   .

2. Creating logical groupings of the regions.

   TXSeries regions must belong to at least one logical group. These groupings are used to distinguish the region based on - requirement, type of applications to run and load capabilities. There are minimum of two groups required, one to group CORs and the other for AORs. The following commands will create two system groups named COR and AOR:

   ```
   cicswlmcfg create group COR -p plx1 -dwlm1
   cicswlmcfg create group AOR -p plx1 -d wlm1
   ```

3. Adding regions to the corresponding groups.

   The following commands will add TXSeries regions COR1 and COR2 to group COR, and add regions AOR1, AOR2, AOR3 and AOR4 to group AOR:

   ```
   cicswlmcfg add region COR1 -g COR -h mac1 -p plx1 -d wlm1
            cicswlmcfg add region COR2 -g COR -h mac2 -p plx1 -d wlm1
            cicswlmcfg add region AOR1 -g AOR -h mac1 -p plx1 -d wlm1
            cicswlmcfg add region AOR2 -g AOR -h mac2 -p plx1 -d wlm1
            cicswlmcfg add region AOR3 -g AOR -h mac3 -p plx1 -d wlm1
            cicswlmcfg add region AOR4 -g AOR -h mac4 -p plx1 -d wlm1
   ```

4. Defining connections between CORs and AORs.

   You need to configure the connectivity between the regions, which would be as depicted:

   - COR1 → connects → AOR1-4
   - COR2 → connects → AOR1-4

   You can configure the connections, by running the following commands:

   ```
   cicswlmcfg add connection AOR1 -r COR1 -S COR1 -R AOR1 -p plx1 -d wlm1 inService=
   cicswlmcfg add connection AOR1 -r COR2 -S COR2 -R AOR1 -p plx1 -d wlm1 inService=1
   cicswlmcfg add connection AOR2 -r COR1 -S COR1 -R AOR2 -p plx1 -d wlm1 inService=1
   cicswlmcfg add connection AOR2 -r COR2 -S COR2 -R AOR2 -p plx1 -d wlm1 inService=1
   cicswlmcfg add connection AOR3 -r COR1 -S COR1 -R AOR3 -p plx1 -d wlm1 inService=1
   cicswlmcfg add connection AOR3 -r COR2 -S COR2 -R AOR3 -p plx1 -d wlm1 inService=1
   cicswlmcfg add connection AOR4 -r COR1 -S COR1 -R AOR4 -p plx1 -d wlm1 inService=1
   cicswlmcfg add connection AOR4 -r COR2 -S COR2 -R AOR4 -p plx1 -d wlm1 inService=1
   ```

   The **inService** argument is set to make the regions available immediately at start-up. The listed commands has three parameters:

   - Name of the connection. (AOR4)
   - Name of the region initiating connection, which is always a COR. (COR2)
   - The connection detail - from starting to terminating region.

     The source region is COR2, and the connecting region is AOR4.

5. Configuring the applications.

   After providing the region information, you need to configure the details of programs and transactions that will be called by clients. The configuration will specify the name of the program or transaction that is defined in the COR/AOR region. It will also specify which group the programs or transactions will execute. (In this case, there is only one AOR group called as AOR).

   ```
   cicswlmcfg add program PRG1 -g AOR -p plx1 -d wlm1
   cicswlmcfg add program PRG2 -g AOR -p plx1 -d wlm1
   cicswlmcfg add transaction TRAN -g AOR -p plx1 -d wlm1
   ```

6. Configuring WAP.

   The WAP configuration requires a file to be created in `/var/cicssm/repos`, with the name, `wap.<wap_name>`. As the WAP name used in the example configuration is `wlm1`, you need to create a file named `wap.wlm1` with the following content:

<hostname> <wap_port>

where:

- *hostname*: Is hostname of the system where WAP will run. Hostname can be obtained by running the command **hostname**.
- *<wap_port>* Is the available port for all WAP communication to take place.

Run the following command to create the `wap.wlm1` file:

```
<cat "hostname 9123" > /var/cicssm/repos/wap.wlm1
```

Configuring the region

To configure the TXSeries regions, complete the following steps:

1. Create the regions – COR1, COR2, AOR1, AOR2, AOR3, and AOR4.
2. Define Listeners in each of the regions. Ensure that the ports used are free for use.
3. Define Connection Definitions in COR1 and COR2 to connect to the AORs.
4. Define the transaction TRAN and programs PRG1, PRG2 in all the regions. In the CORs, these definitions should be set as *Remote* and specify any AOR as default SYSID.
5. Define the WLM User Exit programs in the CORs. Add the following PD entries in COR1 and COR2:
   - Path: **"/usr/lpp/cicssm/bin/bhgdpl.ibmcpp"UserExitNumber=50**
   - Path: **"/usr/lpp/cicssm/bin/bhgdtr.ibmcpp"UserExitNumber=25**

These User Exit programs are used for communication and data exchange between CORs and WLM.

For more information on commands, see the TXSeries for Multiplatforms V8.1 Information Center.

Miscellaneous configuration

The WLM processes communicate and monitor TXSeries regions through TCP communication. This communication takes place using CICS Transaction Gateway (CTG). In order to use WLM, you need to configure CTG as well.

Configure the CTG by doing the following:

1. Add server entries in `ctg.ini` file for all TXSeries regions.
2. Ensure that the ports and hostnames are the same as the ones configured in the respective regions.

You can go through the CICS Transaction Gateway Information Center for information on how to complete the listed steps.

Chapter 5. Starting and verifying the WLM environment

To start using WLM, you first need to start the WLM processes.

Run the **su - cicssm** command. The **cicssm** user must be used to perform any WLM administration. Ensure that you are logged in as **cicssm** user.

Run the following steps:

1. Start WAP named `wlm1` by running **cicswlm start wap wlm1**.
2. Start WCM for WAP `wlm1`. Issue the **cicswlm start wcm wlm1** command.
3. Load the `wlm1.cfg` WLM configuration into the WAP `wlm1` by issuing the **cicswlm load wlm1.cfg wlm1** command.
4. Start the Health Monitor for WAP `wlm1` by issuing command, **cicswlm start hmon wlm1**.

   Check the logs in `/var/cicssm/log` directory for more details of any errors encountered.
5. Start the servers configured in CTG.
6. Start the TXSeries regions using the **cicscp -v start region <region_name> StartType=cold** command.

You can verify the WLM environment by running the client applications and transactions.

The Routing Monitor can be used to verify that WLM is routing the programs properly. To monitor the WLM statistics, run the Routing Monitor (RMON). The RMON screen provides information such as system status, region status, routing statistics and more. A sample screen is depicted in Figure 12.

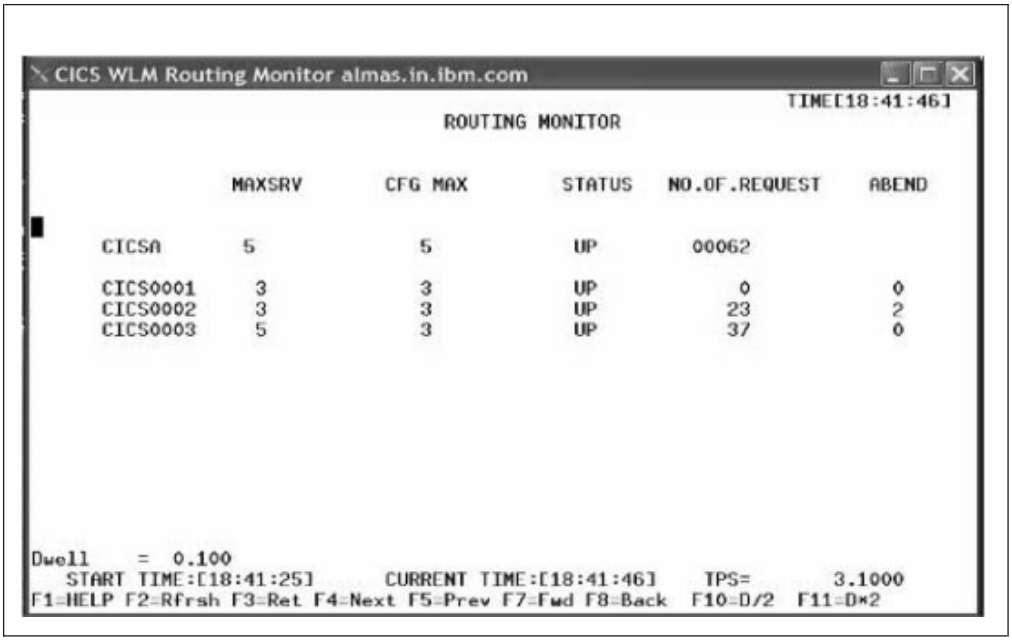
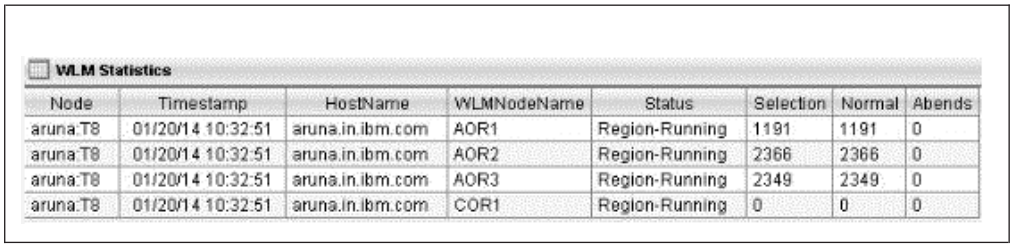


Figure 12. Sample RMON screen

With TXSeries V8.1, you can verify the configuration and load distribution in WLM through the **cicswlmstat** command.

To monitor WLM status, you can use the IBM Tivoli Monitoring Agent. A sample screen is depicted in Figure 13.



WLM Statistics

Node	Timestamp	HostName	WLMNodeName	Status	Selection	Normal	Abends
aruna:T8	01/20/14 10:32:51	aruna.in.ibm.com	AOR1	Region-Running	1191	1191	0
aruna:T8	01/20/14 10:32:51	aruna.in.ibm.com	AOR2	Region-Running	2366	2366	0
aruna:T8	01/20/14 10:32:51	aruna.in.ibm.com	AOR3	Region-Running	2349	2349	0
aruna:T8	01/20/14 10:32:51	aruna.in.ibm.com	COR1	Region-Running	0	0	0

Figure 13. Sample WLM statistics output from IBM Tivoli Monitoring Agent

For further information on WLM, troubleshooting or configuration, see the topics on *Workload Management* in the TXSeries for Multiplatforms V8.1 Information Center.

Chapter 6. References

The following URL's provide you with more details on TXSeries, Power HA and also other related information.

TXSeries for Multiplatforms V7.1 Information Center
http://pic.dhe.ibm.com/infocenter/txformp/v7r1/index.jsp

TXSeries for Multiplatforms V8.1 Information Center
http://pic.dhe.ibm.com/infocenter/txformp/v8r1/index.jsp

Power HA for AIX
http://www.redbooks.ibm.com/abstracts/sg247739.html

Edge components Information Center
http://pic.dhe.ibm.com/infocenter/wasinfo/v7r0/topic/com.ibm.websphere.edge.doc/lb/welcome_edge.html

Appendix. Terminology

Terminologies in Workload Manager:

Application Owning Region (AOR)
AOR is the application owning region (or sometimes referred to as application processing regions in this paper) where the actual business logic application resides. Multiple AORs are configured either in the same server or across multiple servers to provide high availability.

Client Owning Region (COR)
COR is Client Owning region. Client owning region receives the requests from the client and routes it intelligently based on the load, responsiveness and capacity of various AOR regions.

Global Workload Application Program (WAP) Cache
The WAP cache is a global, centralized cache that maintains the properties and details of the CICS regions configured in a WLM setup.

Local Workload Cache Manager (WCM)
The Workload Cache Manager (or WCM) exists for each server where CORs reside. Data is transferred from the WAP cache to the WCM as and when it is needed. If the data is unavailable in the WAP cache and WCM, then default routing takes place based on COR configuration.

Workload Client (WCL)
The WCL is a client application that executes in a CICS COR that is configured to perform routing decisions. The WCL determines the most appropriate CICS AOR which should execute the incoming request. CICS WLM provides a CICS User Exit for Dynamic Transaction Routing (DTR) as well as Dynamic Program Linking (DPL).

WLM Health Monitor (HMON)
The Health Monitor or HMON process detects the status or “health” of the CICS regions. The HMON process is located on the same server as the WAP. It continuously monitors and updates the health of the CICS regions into the routing component of WLM.

WLM listener
The WLM listener (cicswlmlsnr) listens to a port and collects startup and shutdown status of each AOR. The listener updates the AOR status instantly when the AOR starts up, hence ensuring its availability. Similarly, when an AOR shuts down, it is blocked, and removed from the routing selection list.

WLM configuration file
WLM creates an abstract representation of the regions and resources in its environment, called as an object model. This abstract representation is defined in a configuration file. The configuration file can be loaded into memory at runtime which creates the objects and their configuration for use.

Terminologies in PowerHA:

Cluster
Loosely-coupled collection of independent systems (nodes) or LPARs organized into a network for the purpose of sharing resources and communicating with each other.

Node An IBM LPAR is running with AIX and PowerHA that is defined as part of a cluster. Every node has a collection of resources (disks, file systems, IP address (es), and applications such as CICS) that can be transferred to another node in the cluster in case the node fails.

Resource
Resources are logical components of the cluster configuration that can be moved from one node to another.

Resource Group (RG)
Resources are grouped together in resource groups (RGs) which PowerHA keeps highly available as a single unit. In PowerHA configuration, CICS can be a part of resource group. In case of node failure, the components in a resource group including CICS move together from one node to another.

Takeover
The process of the first machine detecting a failure of the second machine and the resource group being moved from the failing machine to the first machine is called takeover.

Shared external disk devices
These are disks that allow multiple nodes to access the disk at the same time. CICS uses the shared external disk to maintain the CICS application data integrity on all nodes. CICS SFS server resides on the external disk in order to be accessible by all required nodes in the cluster.

Service IP Address
It is an IP address used for client access. A service IP Label is a label that matches a service IP address. A service IP label is part of a resource group, which means that PowerHA will monitor it and keep it highly available.

Persistent IP address
A persistent node IP label is an IP alias that can be assigned to a network for a specified node. This IP address is node bounded and this is not any part of resource group.

IP Address Takeover (IPAT)
IP Address takeover is a mechanism of recovering a service IP label by moving it to another adapter on the same node or another node in the cluster. There are two methods of IPAT which differ in the way they will control the service IP:

- *IPAT via aliasing*: The service IP address/Label will be aliased on an existing interface without replacing the base address of the interface.
- *IPAT via replacement*: The service IP address replaces the existing (boot/base) IP address on the network interface.

Fall-over
This terminology explains the movement of resource group(s) from one active node to another node (backup node) as a result of failure on the active node.

Fall-back
This terminology explains the movement of a resource group back from the back up node to previous node, in response to re-integration of previously failed node.

Clients

A client is a system that can access the CICS application running on the cluster nodes.

Printed in USA