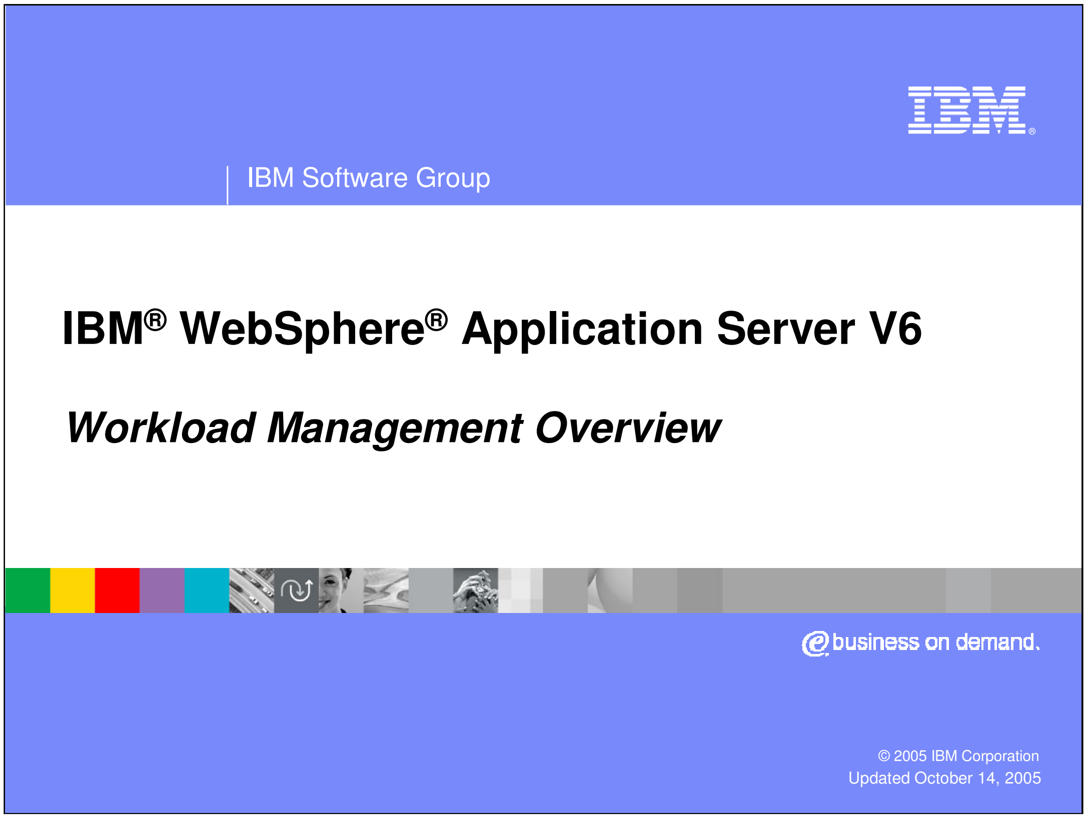


IBM Software Group

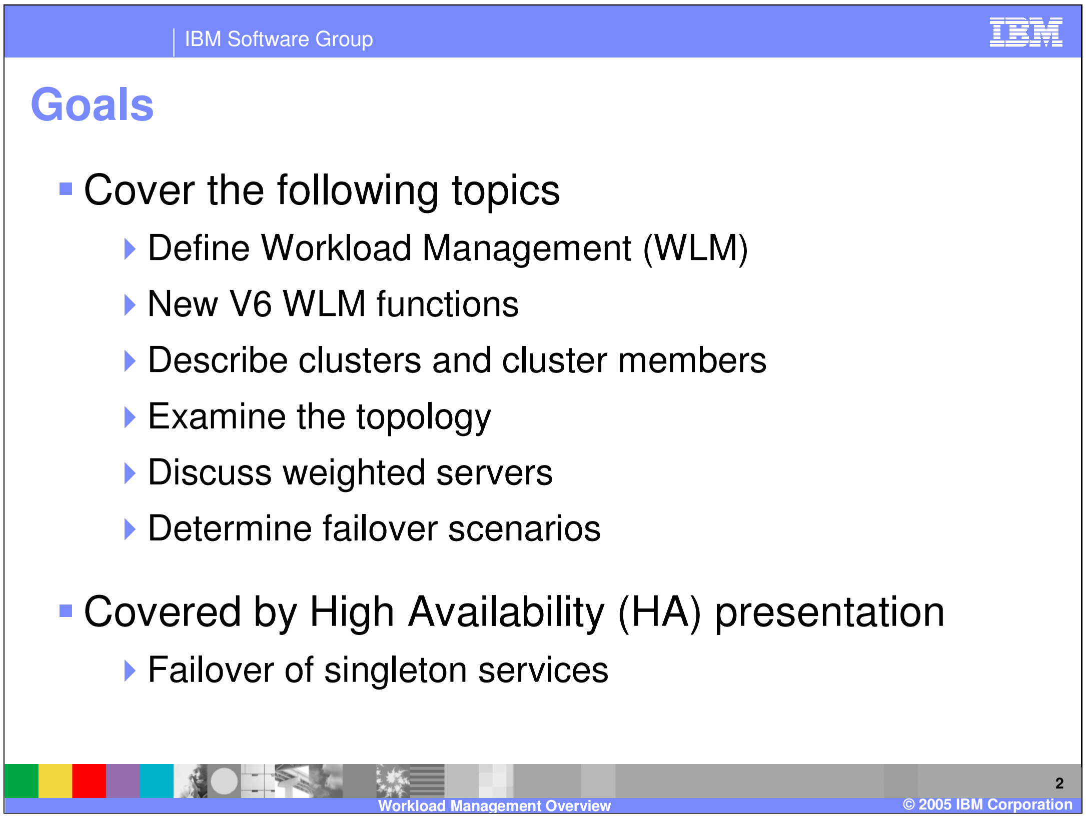
IBM® WebSphere® Application Server V6

Workload Management Overview

business on demand.

Updated October 14, 2005

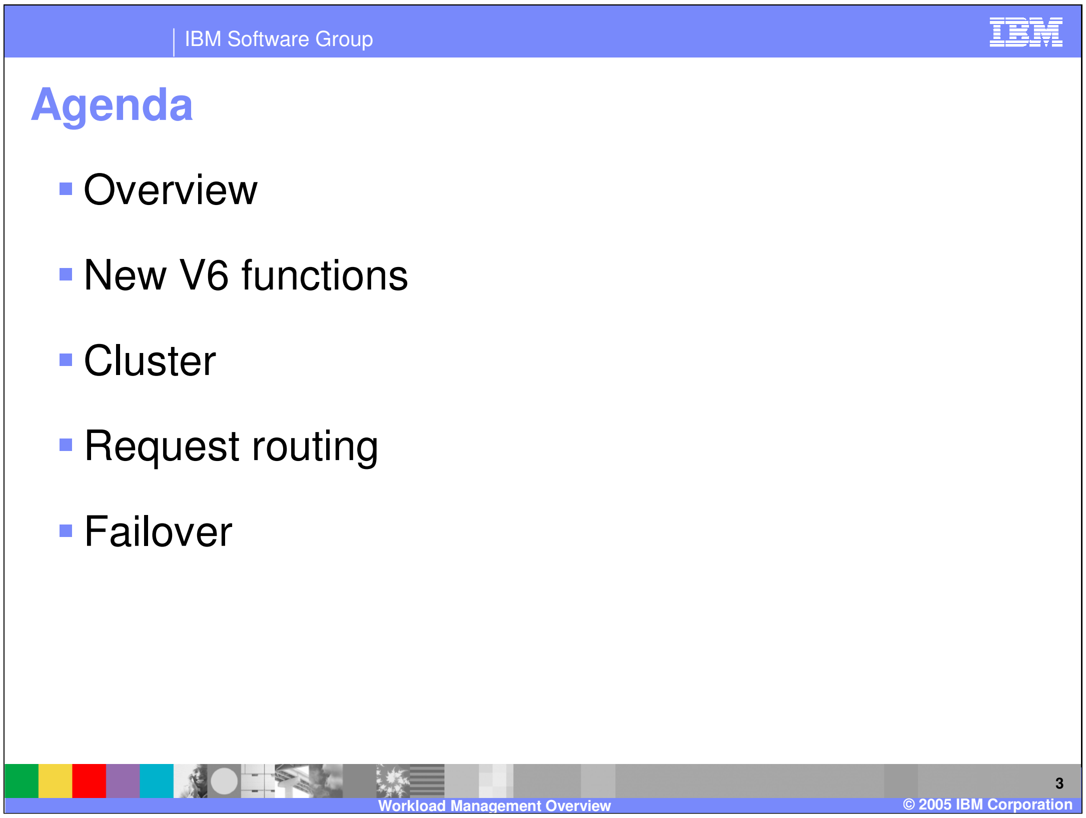
This presentation will provide an overview of Workload Management for WebSphere Application Server V6.



Goals

- Cover the following topics
  - Define Workload Management (WLM)
  - New V6 WLM functions
  - Describe clusters and cluster members
  - Examine the topology
  - Discuss weighted servers
  - Determine failover scenarios
- Covered by High Availability (HA) presentation
  - Failover of singleton services

The goal of this presentation is to define and explain Workload Management. Related High Availability information can be found in separate presentations.



Agenda

- Overview
- New V6 functions
- Cluster
- Request routing
- Failover

The agenda for this presentation is to start with an overview of WLM, then introduce the new features of V6.

IBM Software Group

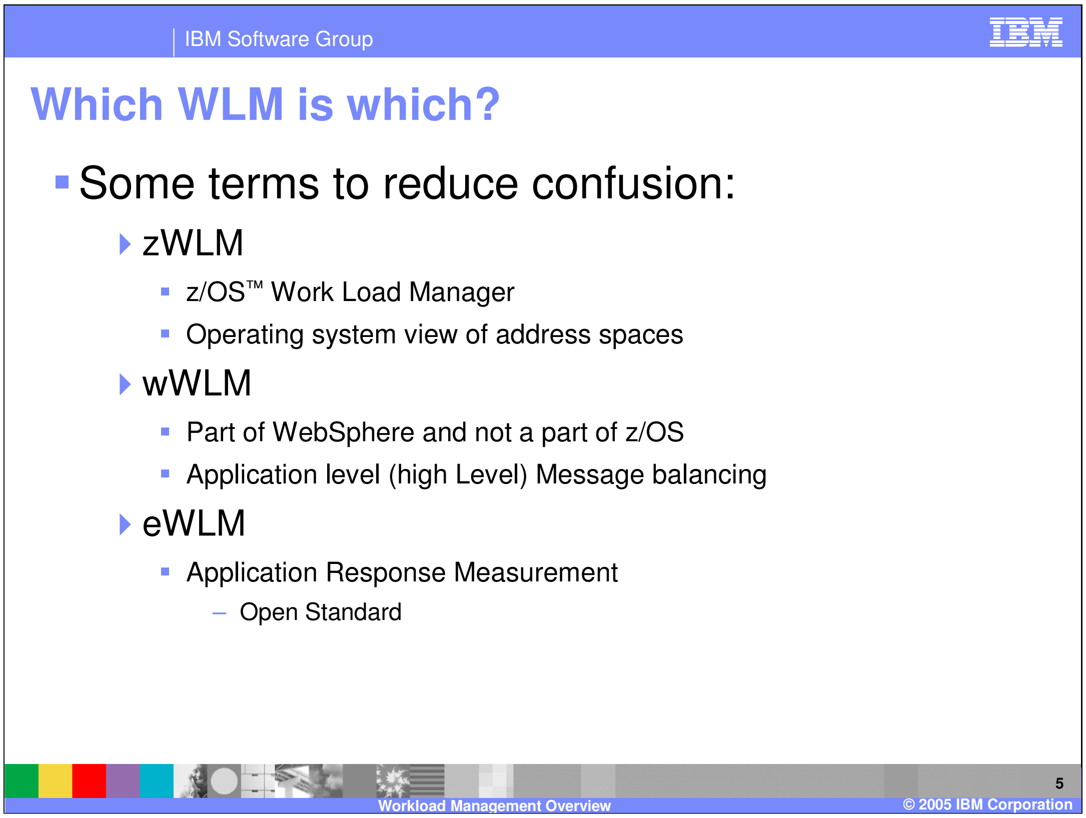
Section

Overview

4

Workload Management Overview

This section is the overview.



IBM Software Group

Which WLM is which?

- Some terms to reduce confusion:
  - zWLM
    - z/OS™ Work Load Manager
    - Operating system view of address spaces
  - wWLM
    - Part of WebSphere and not a part of z/OS
    - Application level (high Level) Message balancing
  - eWLM
    - Application Response Measurement
      - Open Standard

5

Workload Management Overview

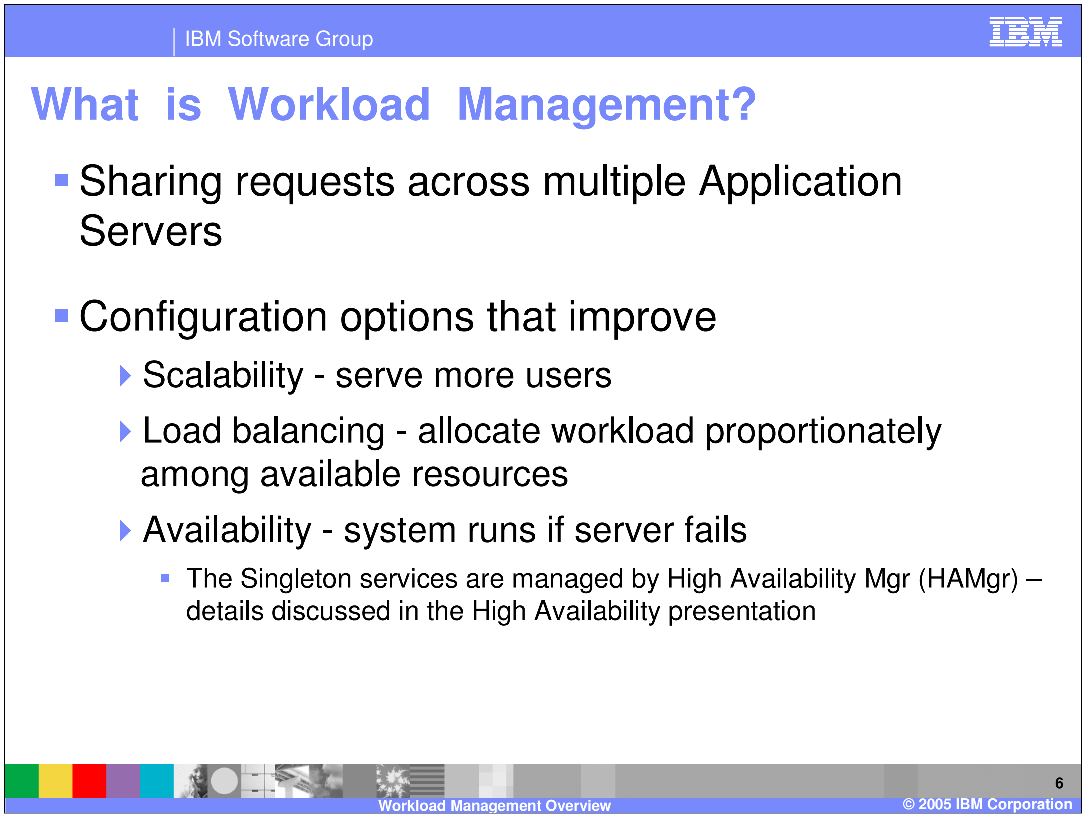
© 2005 IBM Corporation

With WebSphere Application Server version 6.0.1 running on a z/OS platform, there are three kinds of Work Load Managers active. These three Work Load Managers cooperate to make WebSphere perform optimally.

Work Load Manager for z/OS, referred to as zWLM in this presentation, is an integral part of the z/OS operating system. zWLM maintains an operating system view of WebSphere and WebSphere applications. The standard parts of z/OS that were present in WebSphere for z/OS version 5 are still used by v 6.0.1. These include zWLM, Resource Recovery Services (RRS), Peer Restart and Recovery (PRR) and Automatic Restart Manager (ARM). Some of these have some restrictions, which are discussed in other areas of this section of IBM Education Assistant. An example of one of these restrictions is that WebSphere High Availability (HA), if selected, prevents the use of PRR and ARM.

The second WLM is the WebSphere Work Load Manager, referred to as wWLM in this presentation, which is an integral part of WebSphere and maintains an Application view of the requests and processing within WebSphere and WebSphere applications. wWLM is a message router used for load balancing in this context. wWLM is the focus of this presentation. Usually, when people refer to WLM, they are referring to wWLM. This presentation will provide a closer look at wWLM and related functions.

The final form of WLM is enterprise Work Load Manager, referred to as eWLM in this presentation. The core of eWLM is another overloaded abbreviation: Application Response Monitoring (ARM), which is an application programming interface developed by a group of leading technology vendors that can be used to monitor the availability and performance of business transactions within and across diverse applications and systems. An interesting aspect of ARM is that it feeds information back into zWLM in order to provide a wider view of the environment. Finally, ARM is an open standard, the details of which can be viewed at **http://www.opengroup.org/tech/management/arm/**.



IBM Software Group

What is Workload Management?

- Sharing requests across multiple Application Servers
- Configuration options that improve
  - Scalability - serve more users
  - Load balancing - allocate workload proportionately among available resources
  - Availability - system runs if server fails
    - The Singleton services are managed by High Availability Mgr (HAMgr) – details discussed in the High Availability presentation

Workload Management Overview

6

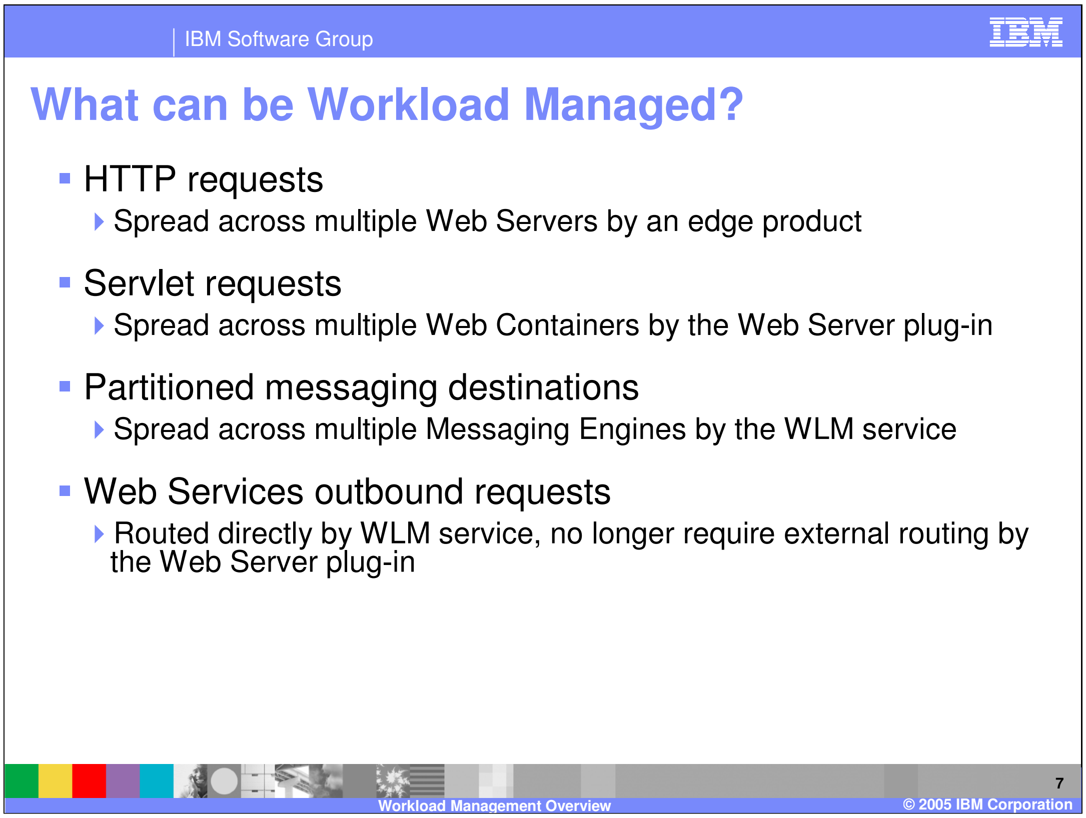
© 2005 IBM Corporation

WebSphere workload management optimizes the distribution of client processing requests.

Incoming work requests are distributed to the application servers, enterprise beans, servlets, and other objects that can most effectively process the requests.

Workload management also provides failover when servers are not available, improving application availability.

In the WebSphere Application Server environment, workload management is implemented by using Clusters of application servers.



What can be Workload Managed?

- HTTP requests
  - Spread across multiple Web Servers by an edge product
- Servlet requests
  - Spread across multiple Web Containers by the Web Server plug-in
- Partitioned messaging destinations
  - Spread across multiple Messaging Engines by the WLM service
- Web Services outbound requests
  - Routed directly by WLM service, no longer require external routing by the Web Server plug-in

Several types of requests can be workload managed

HTTP Requests can be shared across multiple HTTP Servers.

This requires a TCP/IP sprayer to distribute the incoming requests. There are both hardware and software products available to spray TCP/IP requests. Network Dispatcher is a software solution that is part of the WebSphere Edge Server. Network Dispatcher applies intelligent load balancing to HTTP requests.

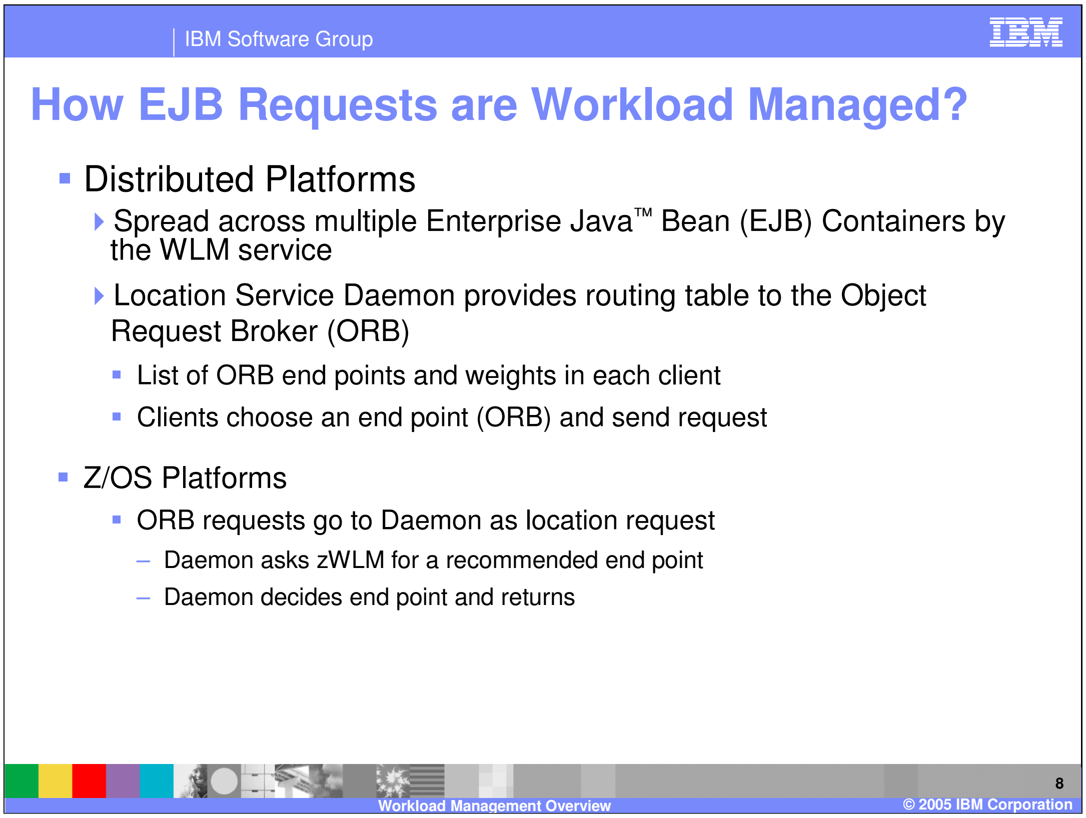
Servlet Requests can be shared across multiple Web Containers.

The WebSphere Plug-in to the HTTP Server distributes Servlet requests.

Web Containers can be configured on the same host system or multiple systems.

Partitioned messaging destinations utilize workload management to distribute messaging workload across multiple Messaging Engines.

Web Services outbound requests are now routed directly, and no longer require an external process, such as the Web Server plug-in to act as an intermediary.



IBM Software Group

How EJB Requests are Workload Managed?

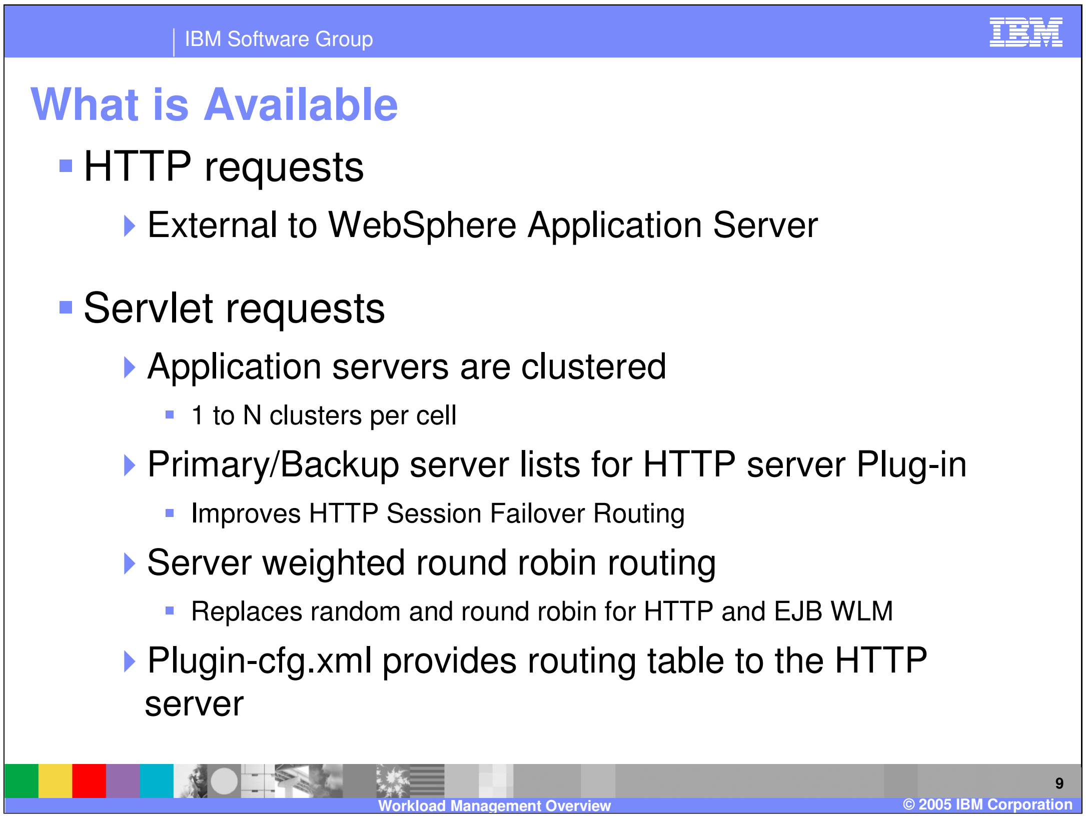
- Distributed Platforms
  - Spread across multiple Enterprise Java™ Bean (EJB) Containers by the WLM service
  - Location Service Daemon provides routing table to the Object Request Broker (ORB)
    - List of ORB end points and weights in each client
    - Clients choose an end point (ORB) and send request
- Z/OS Platforms
  - ORB requests go to Daemon as location request
    - Daemon asks zWLM for a recommended end point
    - Daemon decides end point and returns

8

Workload Management Overview

© 2005 IBM Corporation

Although EJB Requests can be shared across multiple EJB Containers on all platforms, EJB requests are handled differently on distributed and z/OS platforms. On distributed platforms, the Workload Management Plug-in to the Object Request Broker (ORB) distributes EJB requests. EJB requests can come from clients, including Servlets, Java client applications, or another EJB. The clients determine the end points based on the routing information provided by the Location Service Daemon. On z/OS platforms, clients send locate requests to the Daemon and the Daemon, with the advice of zWLM, determines the end point.



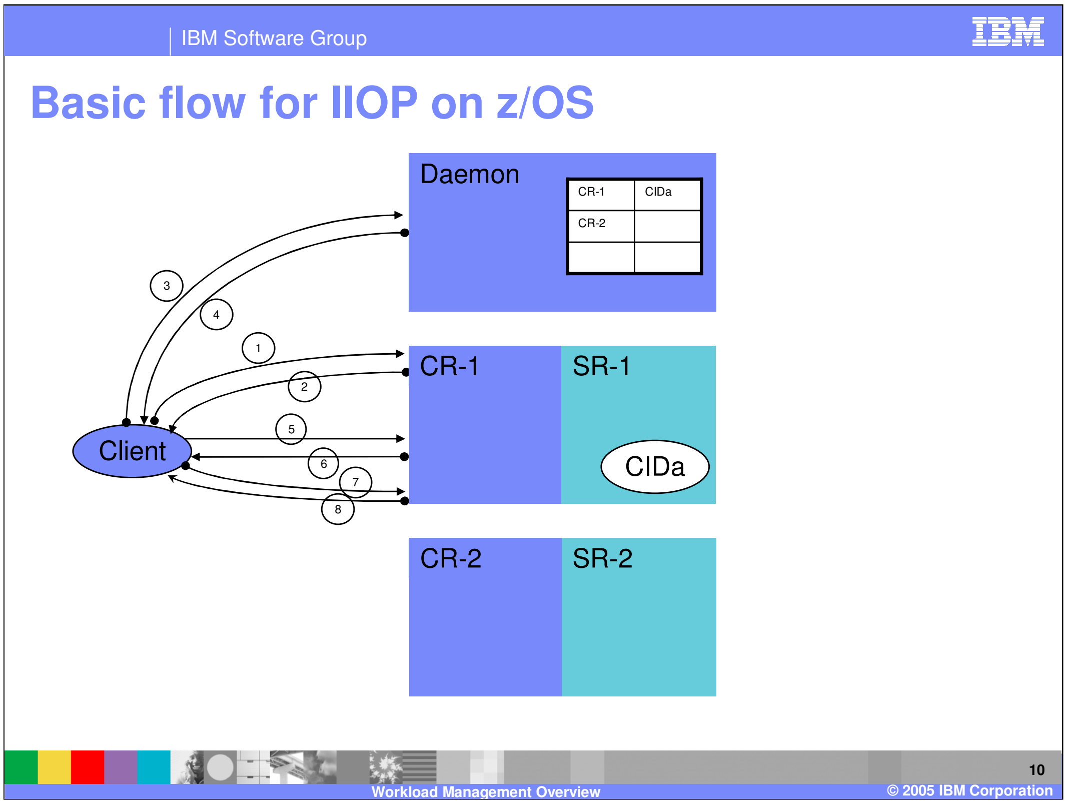
What is Available

- HTTP requests
  - External to WebSphere Application Server
- Servlet requests
  - Application servers are clustered
    - 1 to N clusters per cell
  - Primary/Backup server lists for HTTP server Plug-in
    - Improves HTTP Session Failover Routing
  - Server weighted round robin routing
    - Replaces random and round robin for HTTP and EJB WLM
  - Plugin-cfg.xml provides routing table to the HTTP server

As mentioned previously, HTTP Requests are workload managed externally to WebSphere Application servers.

For Servlet requests, you can configure multiple Clusters in a cell and multiple cluster members within a cluster, as logic dictates for a given scenario. The HTTP server plug-in reads a list of servers it can route to from the plugin-cfg.xml file. In the unlikely event that all the servers fail to respond, the plug-in also has a back-up server list to route to.

Each of those servers also has an associated weight, which will be discussed momentarily. The routing option is a weighted round robin.



IBM Software Group

Basic flow for IIOP on z/OS

Daemon

CR-1	CIDa
CR-2	

10

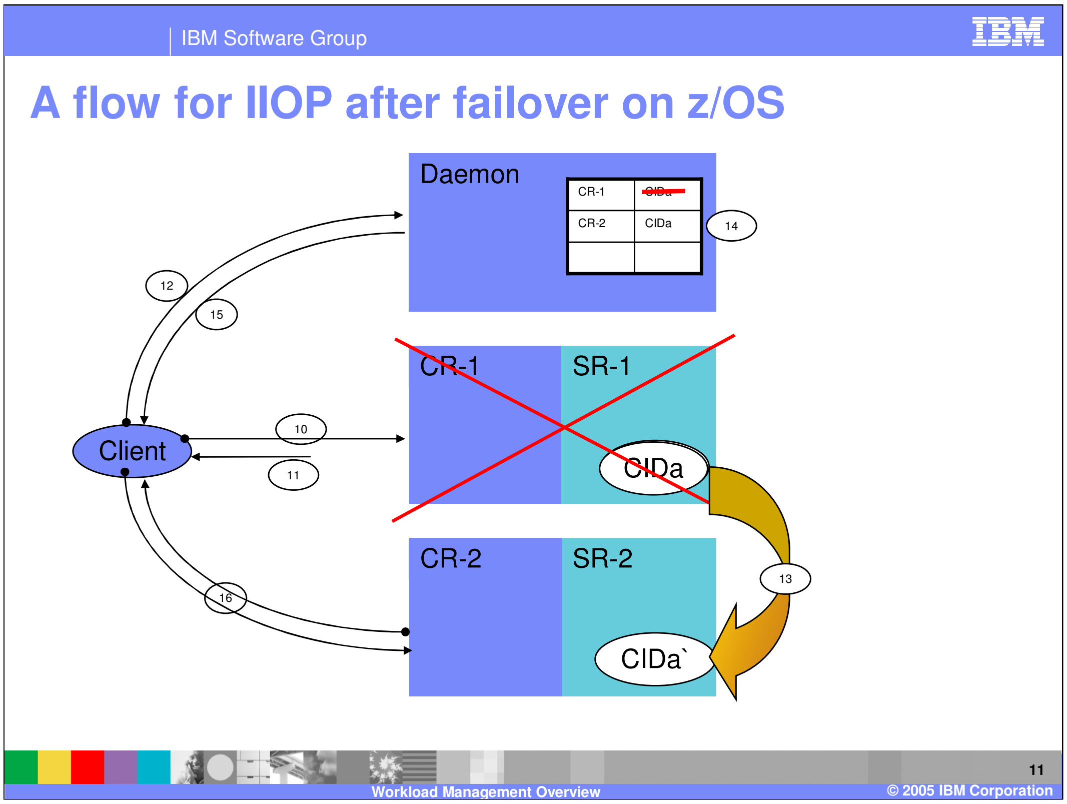
Workload Management Overview

© 2005 IBM Corporation

This graphic shows the a basic flow of an IIOP location request on z/OS and though there are many possible scenarios, this one demonstrates the key points. The client shown can be any object invoking an EJB bean, such as another application server or an edge server. The arrows represent an ordered sequence of events.

1. The client sends a lookup request to a name-server which is located in the server 1 Control Region (CR-1).
2. CR-1 returns an IOR pointing at the Daemon
3. The client sends a Locate request to the Daemon
4. The Daemon returns an IOR Pointing at CR-1
5. The client sends Locate Request to CR-1
6. CR-1 sends an automatic “HERE”
7. The client sends a Create request to CR-1. As shown, CR1 then performs 3 tasks:
   a. CR-1 queues a request to SR-1 (using zWLM) to hydrate an instance of the EJB
   b. SR-1 hydrates an instance with id CIDa (Cluster ID of the Instantiated Object)
   c. SR-1 creates an IOR for the new object containing CIDa
8. CR-1 returns an IOR for the new object to the Client
9. Once the connection is established, the client sends requests to the object and gets responses through CR-1.

Even though there appears to be several messages moving about, this process is quite efficient because the messages are quite small and z/OS handles them efficiently.



IBM Software Group

A flow for IIOP after failover on z/OS

CR-1	~~CIDa~~
CR-2	CIDa

11

Workload Management Overview

© 2005 IBM Corporation

This graphic shows what happens when an object is no longer available. Here we have shown a complete server stopping and the client then sending a request to the object. Again there are several scenarios that might require an object to be moved, but this case demonstrates the key points. Following the numbered arrows as before we have:

10. The request sent by the client
11. (A broken arrow) Failed communication because application or server has failed
12. Client sends a Locate Request to Daemon
13. Failover has detected failure and inflated a new object with state and with CIDa (CIDa' on the previous chart)
14. Daemon updates table (removes CR-1 reference and inserts CR-2 reference)
15. Daemon returns new IOR pointing at CIDa' on CR-2
16. Client resumes conversation with CIDa' on CR-2

IBM Software Group

Section

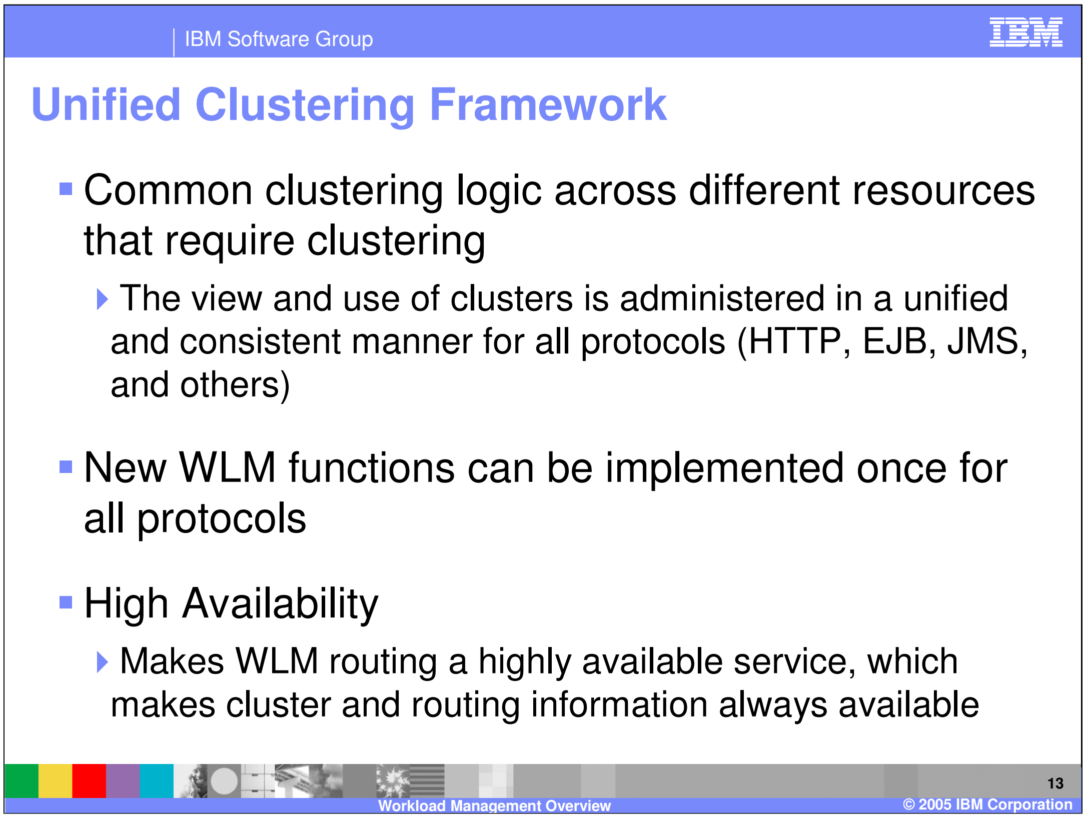
New V6 WLM Functions

12

Workload Management Overview

© 2005 IBM Corporation

This section includes the new workload management functions in V6.



IBM Software Group

Unified Clustering Framework

- Common clustering logic across different resources that require clustering
  - The view and use of clusters is administered in a unified and consistent manner for all protocols (HTTP, EJB, JMS, and others)
- New WLM functions can be implemented once for all protocols
- High Availability
  - Makes WLM routing a highly available service, which makes cluster and routing information always available

This unified view is not limited to application servers. Several kinds of servers can be administered from the Administrative Console and the administration of servers in a given server type has the same appearance. WebSphere V6.0.2 on distributed and z/OS platforms handle IIOP differently as discussed earlier under EJB Requests.

IBM Software Group

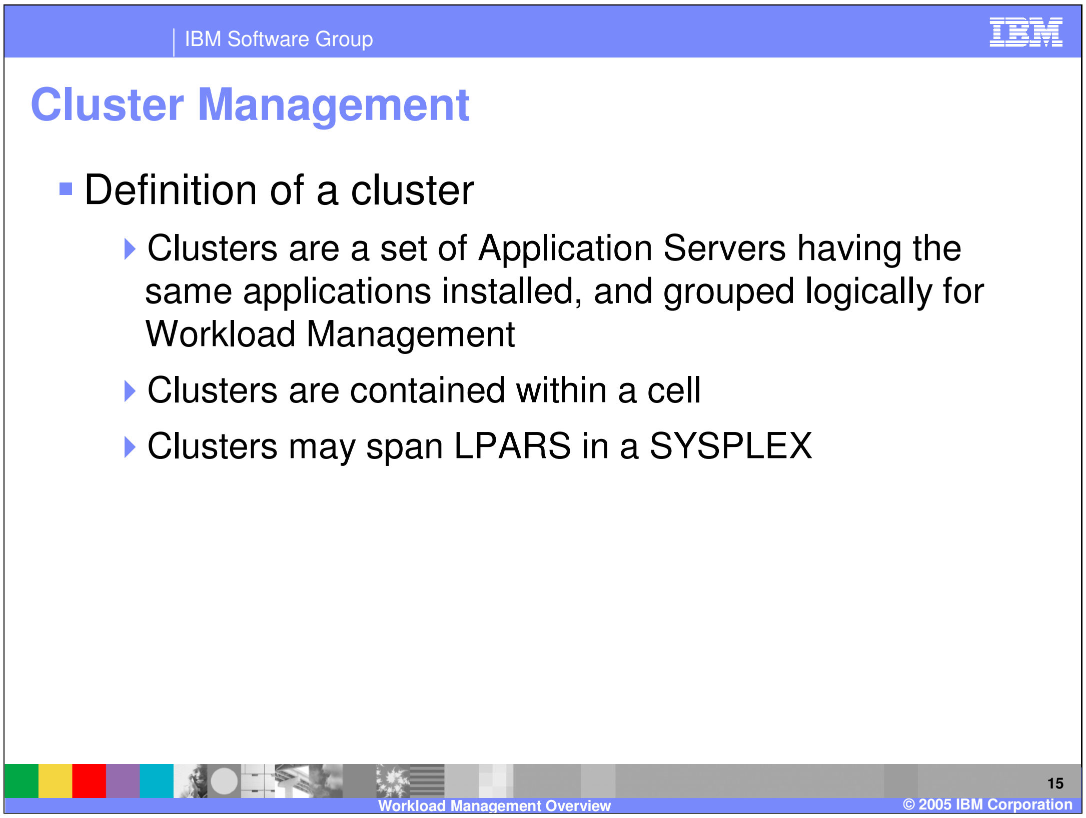
Failover of Stateful Session EJBs

14

Workload Management Overview

© 2005 IBM Corporation

WebSphere Application Server Version 6.0.1 enables you to construct applications with the assumption that your applications using stateful session beans are not limited by unexpected server failures. Version 6.0.1 utilizes the functions of the Data Replication Service (DRS) and Workload Management (WLM) so you can enable stateful session bean failover. Because you might not want to enable failover for every single stateful session bean installed in the EJB container, you can override the EJB container settings at either the application or EJB module level. WLM will failover to a “hot” system that already has the session state and caches that have been DRS enabled.



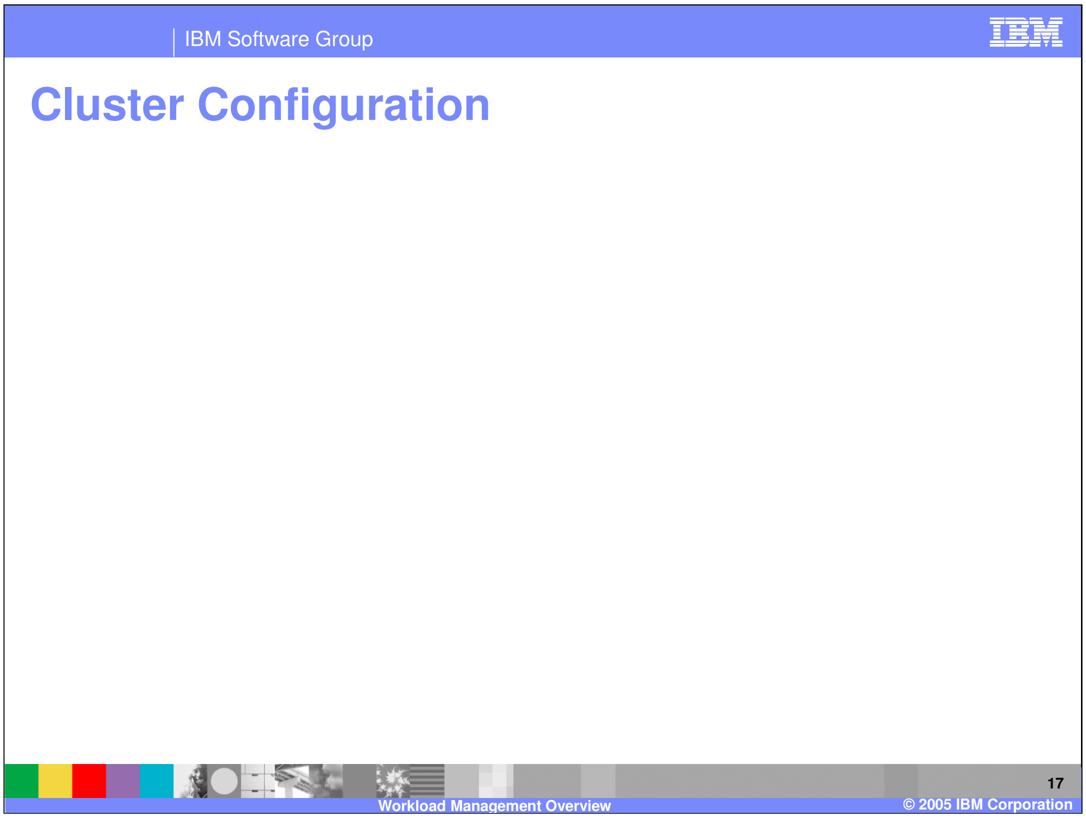
Cluster Management

- Definition of a cluster
  - Clusters are a set of Application Servers having the same applications installed, and grouped logically for Workload Management
  - Clusters are contained within a cell
  - Clusters may span LPARS in a SYSPLEX

Clusters increase availability when they span multiple LPARS with a SYSPLEX. Several application servers can run on a single system, but there is no requirement that they all be in the same cluster. Clustering is a logical grouping, not a physical one. All members of a cluster are nearly identical 'clones' of a common ancestor.

Creating a Cluster

Creating a cluster is a straightforward process. The fist step is to choose a name for the cluster and decide if a replication domain is to be created for this cluster. The next step is to include servers in the new cluster. You can include multiple servers as you create a new cluster, an existing server, or several new servers. If you include an existing server in the cluster and multiple new servers, the new servers will have the same application running on them as the pre-defined one.



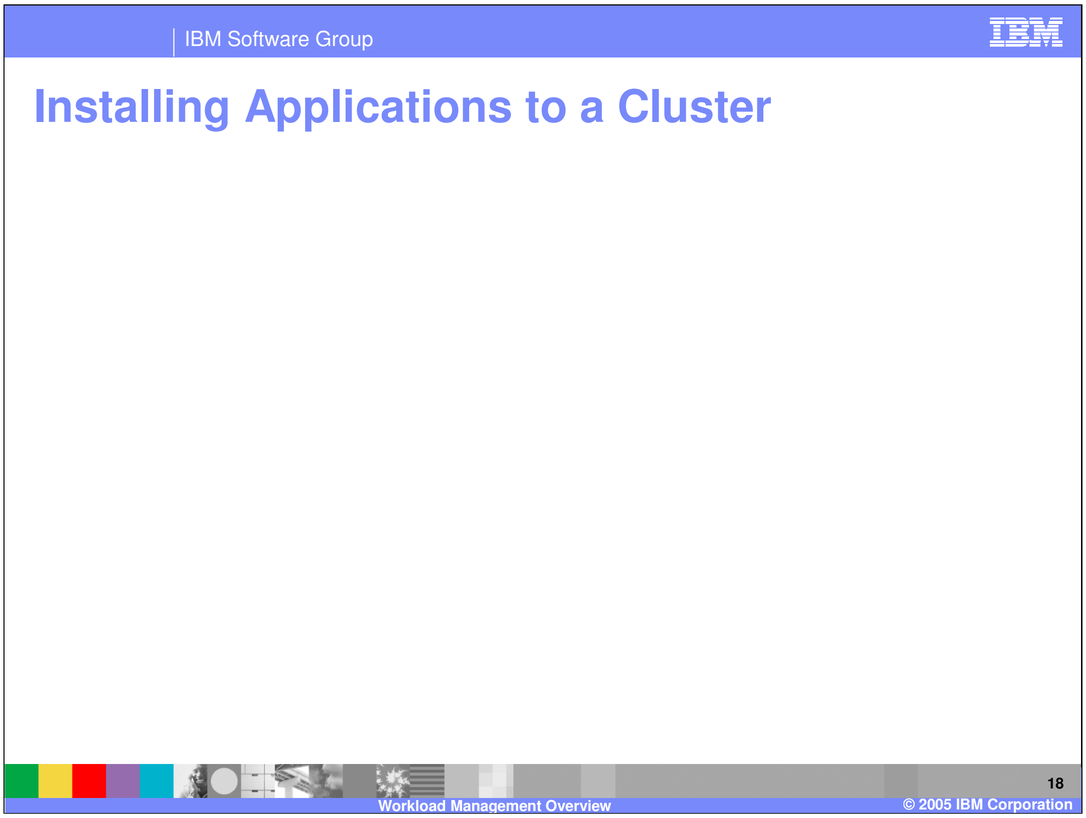
IBM Software Group

Cluster Configuration

17

Workload Management Overview

Once a cluster has been created, the basic fields for cluster administration are shown here. Here you can change several of the properties of the cluster and alter the cluster membership and endpoint listeners. For example, you can specify the node group that bounds this cluster. All application servers that are members of a cluster must be on nodes that are members of the same node group. The enable high availability for persistent services turns HA on or off. If HA is off, Peer Restart and Recovery must be used for high availability. If HA is turned on, Peer Restart and Recovery must not be used. More about HA can be found in the presentation on High Availability in this section of the IBM Education Assistant. Finally, an endpoint listener receives requests from service requester applications within a specific application server or cluster.



IBM Software Group

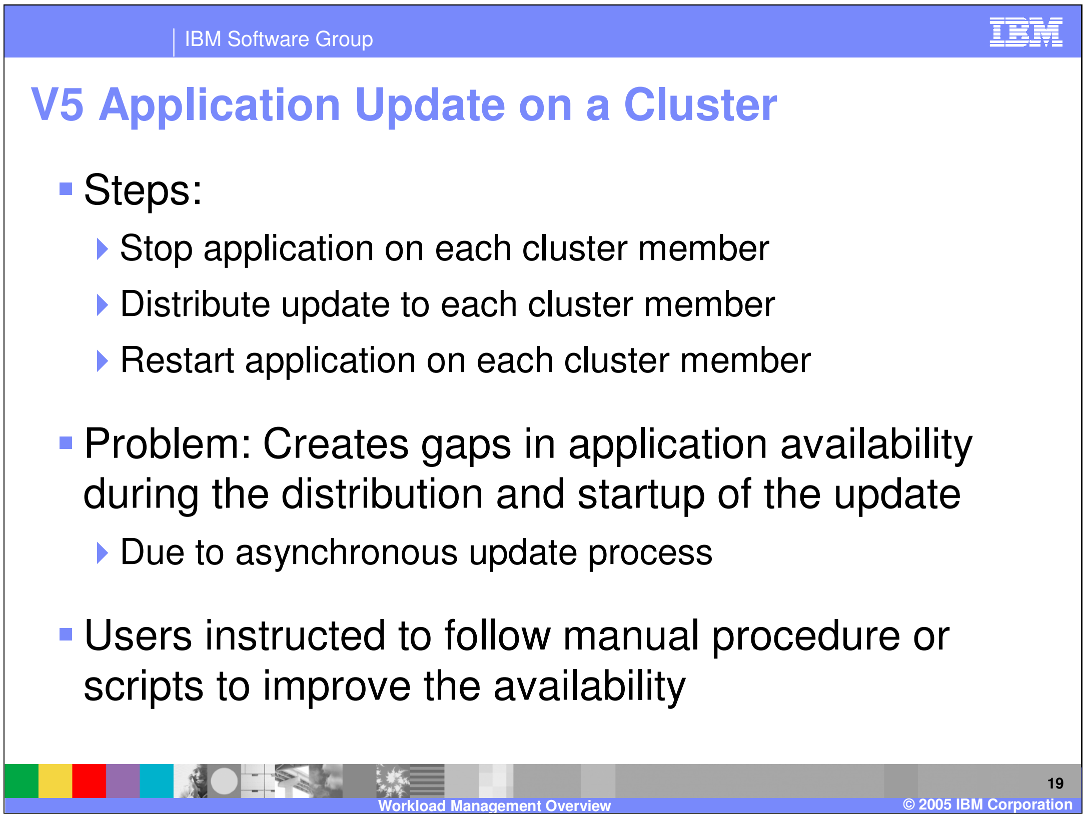
Installing Applications to a Cluster

18

Workload Management Overview

© 2005 IBM Corporation

Updating applications in a cluster is done in the same manner as updating applications in a stand-alone server.



IBM Software Group

V5 Application Update on a Cluster

- Steps:
  - Stop application on each cluster member
  - Distribute update to each cluster member
  - Restart application on each cluster member
- Problem: Creates gaps in application availability during the distribution and startup of the update
  - Due to asynchronous update process
- Users instructed to follow manual procedure or scripts to improve the availability

19

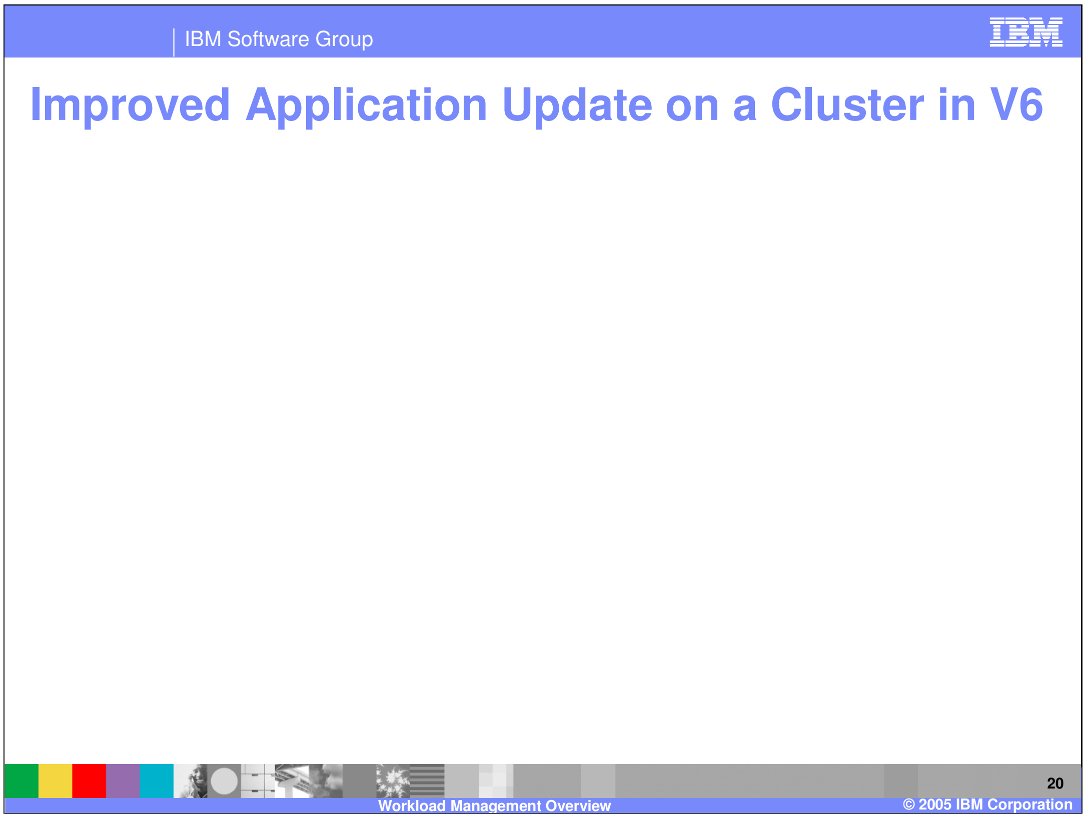
Workload Management Overview

© 2005 IBM Corporation

In version 5, users were instructed to follow manual or scripted procedures to ensure availability during application update. The procedure varied slightly between Distributed and z/OS® platforms, but followed a similar pattern consisting of:

1. Route work away from cluster member
2. Stop application
3. Distribute update to node
4. Re-start application
5. Resume routing work to cluster member

To increase availability, it is necessary to write wsadmin scripts.



IBM Software Group

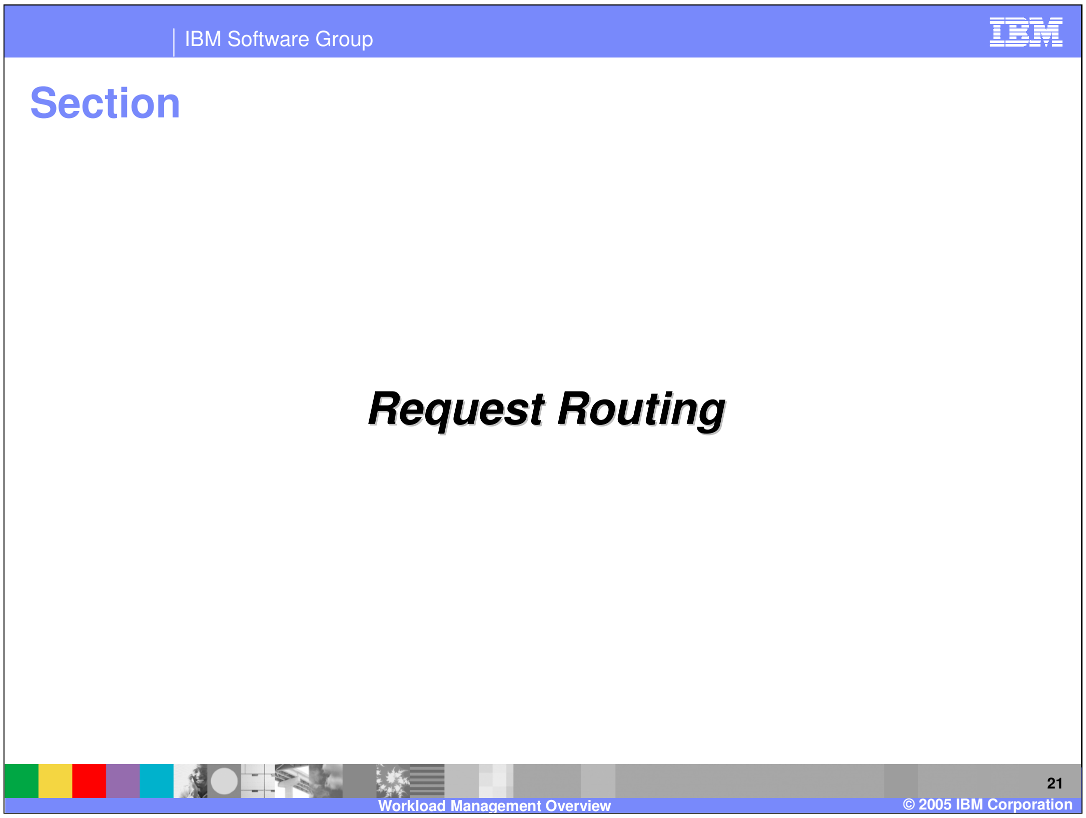
Improved Application Update on a Cluster in V6

20

Workload Management Overview

If a modified application or module is deployed on a cluster, roll out the changes to all cluster members of the cluster on which the application or module is deployed. Click Rollout Update on the Enterprise Applications page to propagate the modified application to all members of the cluster on which the application or module is deployed. Rollout Update sequentially updates the configuration on the nodes that contain cluster members performing the sequence described on this slide.

Tip: At the end of the *Installing* messages displayed by the console during application or module installation, click Manage Applications to go to the Enterprise Applications page. Do not save changes to your configuration until after you roll out the changes.



IBM Software Group

Section

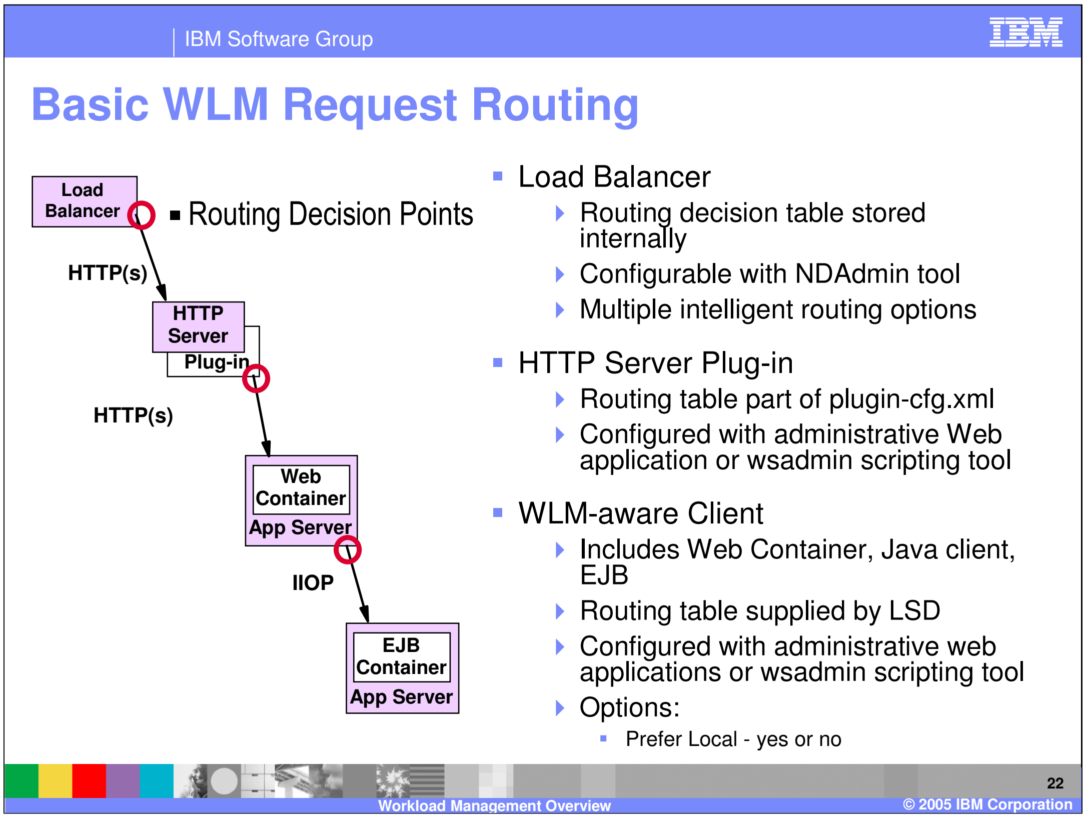
Request Routing

21

Workload Management Overview

© 2005 IBM Corporation

This section describes request routing.



IBM Software Group

Basic WLM Request Routing

Load Balancer

- Routing Decision Points

HTTP(s)

HTTP Server

Plug-in

HTTP(s)

Web Container

App Server

IIOP

EJB Container

App Server

- Load Balancer
  - Routing decision table stored internally
  - Configurable with NDAdmin tool
  - Multiple intelligent routing options
- HTTP Server Plug-in
  - Routing table part of plugin-cfg.xml
  - Configured with administrative Web application or wsadmin scripting tool
- WLM-aware Client
  - Includes Web Container, Java client, EJB
  - Routing table supplied by LSD
  - Configured with administrative web applications or wsadmin scripting tool
  - Options:
    - Prefer Local - yes or no

22

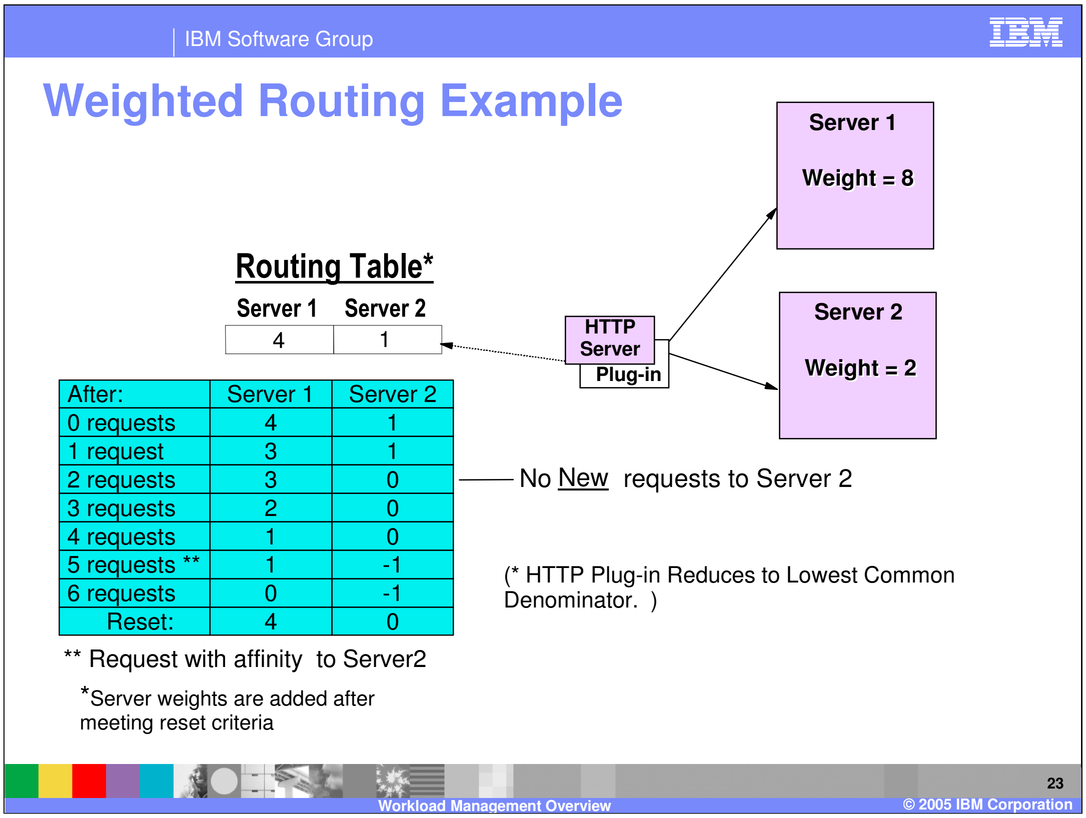
Workload Management Overview

© 2005 IBM Corporation

This section addresses the scenario where everything works as planned. Failover situations will be addressed later.

The Load Balancer is an IP sprayer that makes intelligent load balancing decisions. Using the NDAdmin tool, you can set it up to route to your HTTP servers based on Round Robin, Statistical Round Robin, Best, Custom Advisor, or Content Based Routing.

Once the request arrives at an HTTP server, the routing is Weighted Round Robin. The only configuration option is how much weight to assign each server. The routing information, the list of available servers and their weights, is ultimately stored by the Deployment Manager. This configuration is used to create the plugin-cfg.xml file for the HTTP server plug-in.



IBM Software Group

Weighted Routing Example

Routing Table*

Server 1	Server 2
4	1

HTTP Server

Plug-in

Server 1

Weight = 8

Server 2

Weight = 2

After:	Server 1	Server 2
0 requests	4	1
1 request	3	1
2 requests	3	0
3 requests	2	0
4 requests	1	0
5 requests **	1	-1
6 requests	0	-1
Reset:	4	0

No New requests to Server 2

(* HTTP Plug-in Reduces to Lowest Common Denominator.)

** Request with affinity to Server2

*Server weights are added after meeting reset criteria

23

Workload Management Overview

© 2005 IBM Corporation

When the HTTP Server plug-in is generated, servlet request routing weights are written into the plugin-cfg.xml file, which the HTTP server will reload at configurable intervals.

When a client requests an IOR for an EJB, the Location Service Daemon returns the IOR and a copy of the routing table. The client uses two in-memory copies of the table, one static and one dynamic. The static copy is used only to cache a local copy of the weights.

When the HTTP Server plug-in is generated, servlet request routing weights are written into the plugin-cfg.xml file, which the HTTP server will reload at configurable intervals.

There is a distinct Routing Table for each cluster.

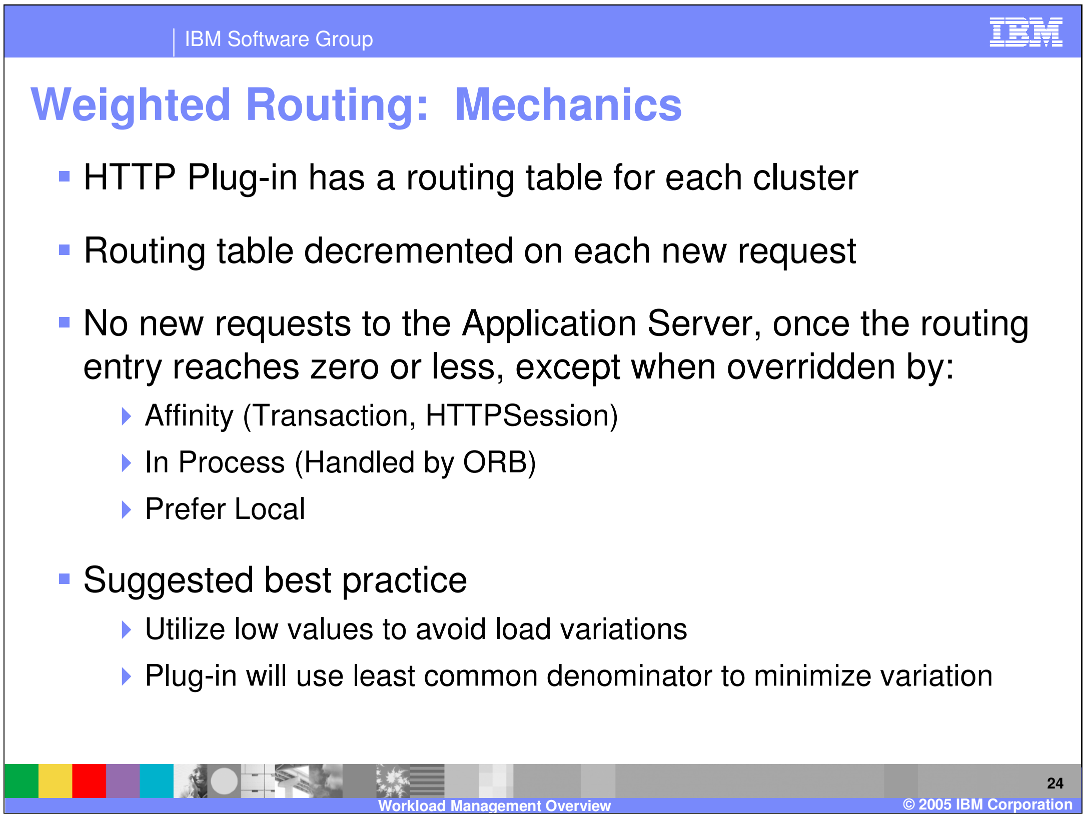
The table illustrated here has two entries, Server 1 and Server 2. The initial values are 4 and 1 (the Least Common Denominator is calculated). The first request is sent to Server 1, and the counter for Server 1 will be decremented by one. The next request will be routed to Server 2, and the count for server 2 will be decremented.

Routing is Round Robin for all servers with non-zero table values.

Requests with affinity are used to decrement the table. As you can see this can cause a server to go "negative",

When all servers have a zero or lower value, the weights are added to the values in the table.

This assures a more even distribution since requests with affinity are part of the calculation.



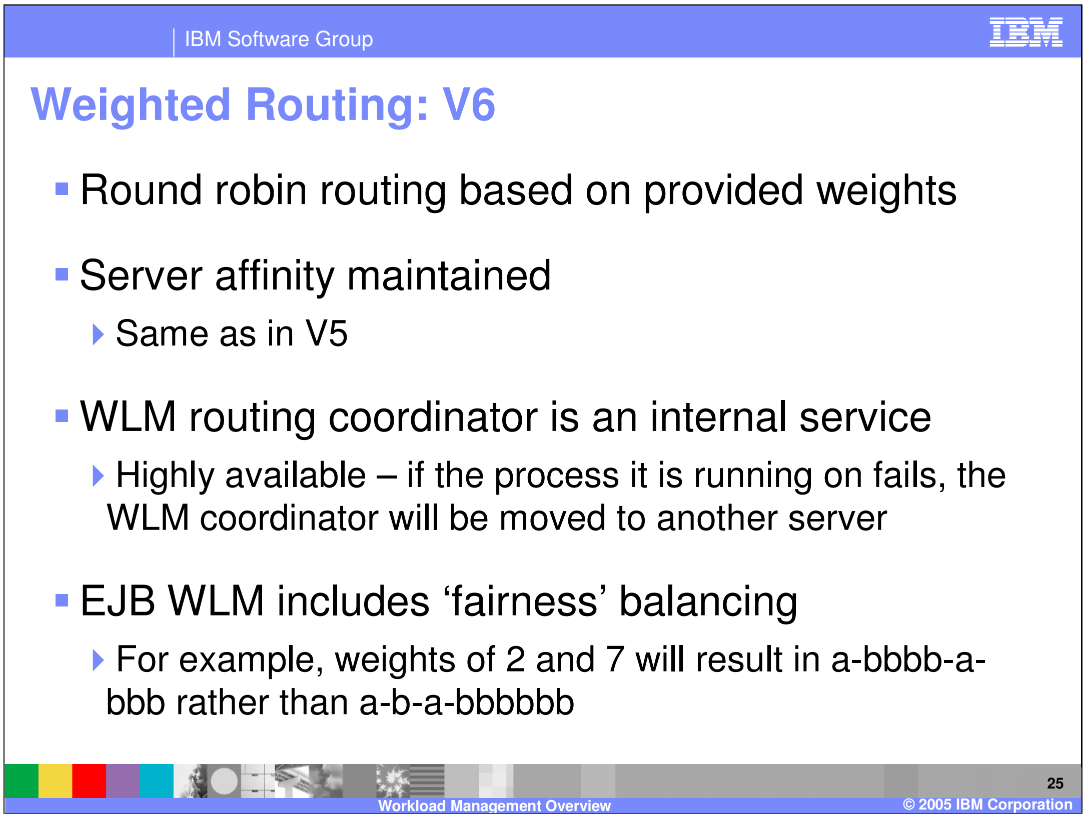
Weighted Routing: Mechanics

- HTTP Plug-in has a routing table for each cluster
- Routing table decremented on each new request
- No new requests to the Application Server, once the routing entry reaches zero or less, except when overridden by:
  - Affinity (Transaction, HTTPSession)
  - In Process (Handled by ORB)
  - Prefer Local
- Suggested best practice
  - Utilize low values to avoid load variations
  - Plug-in will use least common denominator to minimize variation

One thing to emphasize is that only new requests are subject to the weighted routing. Requests that have sessions already in progress will be sent to the server that started the session. Either session Affinity or transaction affinity will override the routing.

There are actually two tables passed when Prefer Local is set. Only the 'Local' table is used, unless all the servers in the Local table have failed. At that point, the other table comes into play.

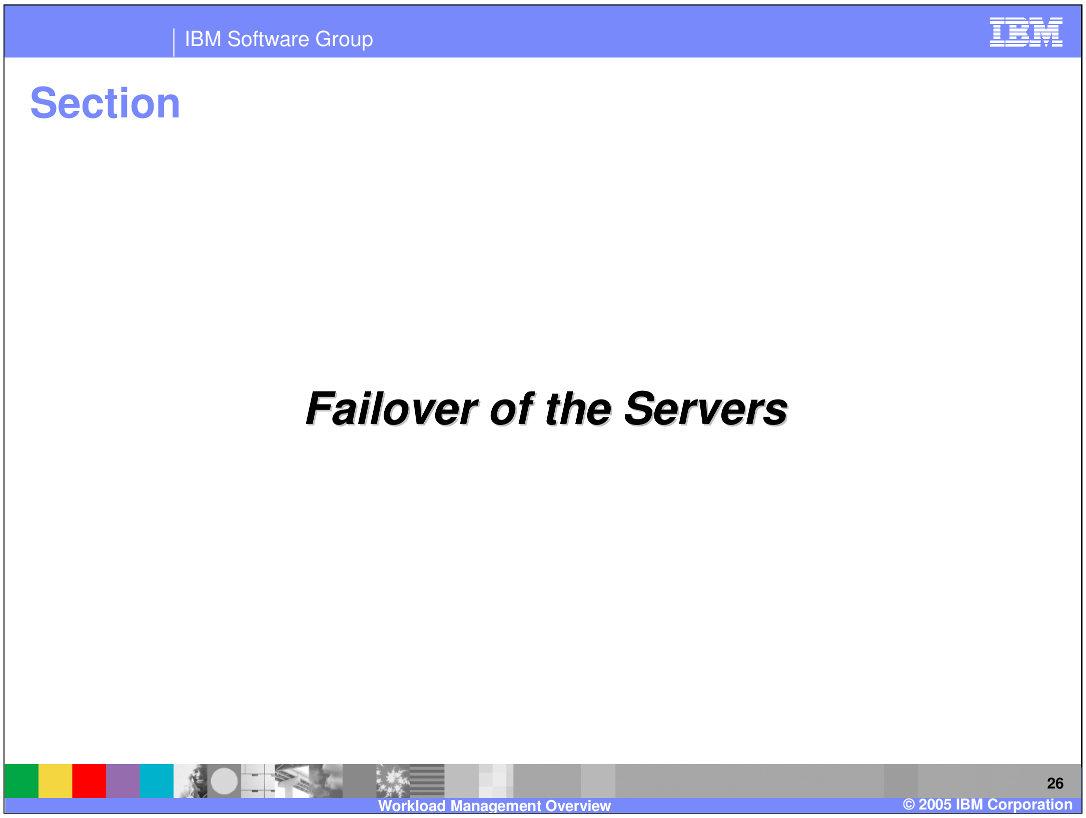
Since the difference between the values in the tables is handled sequentially, it is a good idea to use small numbers for the weights.



Weighted Routing: V6

- Round robin routing based on provided weights
- Server affinity maintained
  - Same as in V5
- WLM routing coordinator is an internal service
  - Highly available – if the process it is running on fails, the WLM coordinator will be moved to another server
- EJB WLM includes ‘fairness’ balancing
  - For example, weights of 2 and 7 will result in a-bbbb-a-bbb rather than a-b-a-bbbbbb

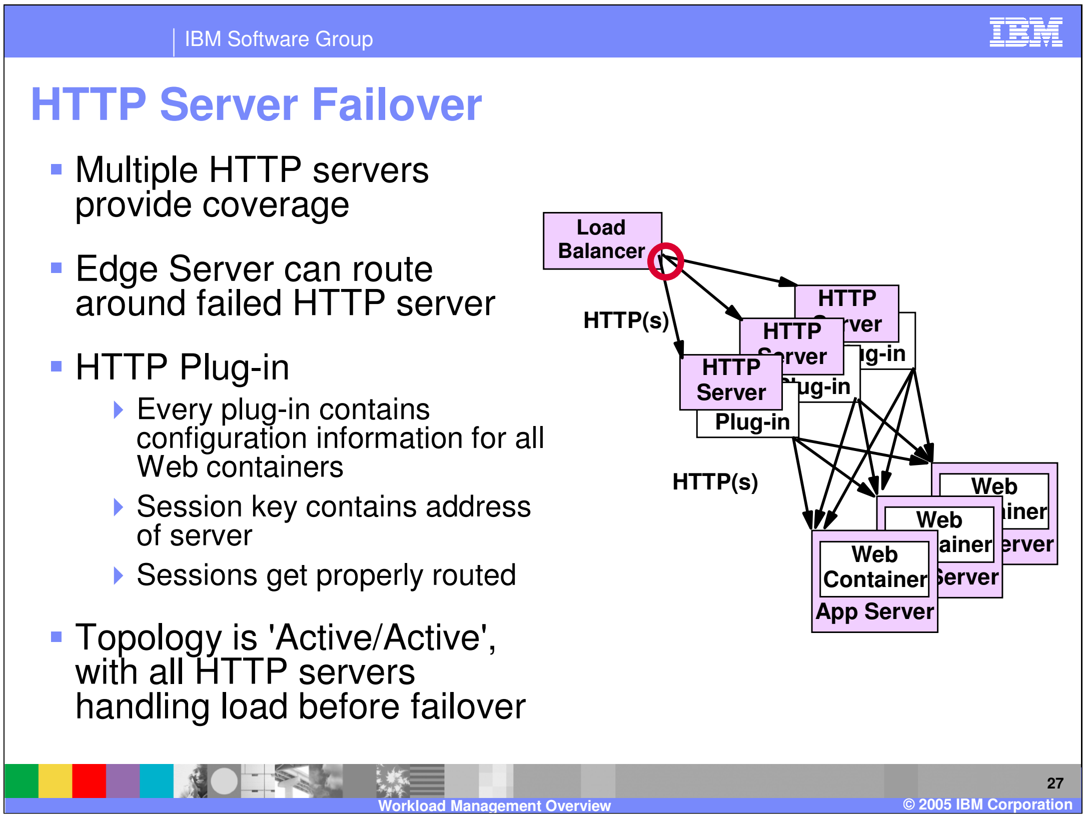
Here is a summary of the Weighted Round robin found in WebSphere v 6.0.1. Remember that EJB routing in z/OS is determined by the Daemon.



Section

Failover of the Servers

This section will cover failover of servers



IBM Software Group

HTTP Server Failover

- Multiple HTTP servers provide coverage
- Edge Server can route around failed HTTP server
- HTTP Plug-in
  - Every plug-in contains configuration information for all Web containers
  - Session key contains address of server
  - Sessions get properly routed
- Topology is 'Active/Active', with all HTTP servers handling load before failover

27

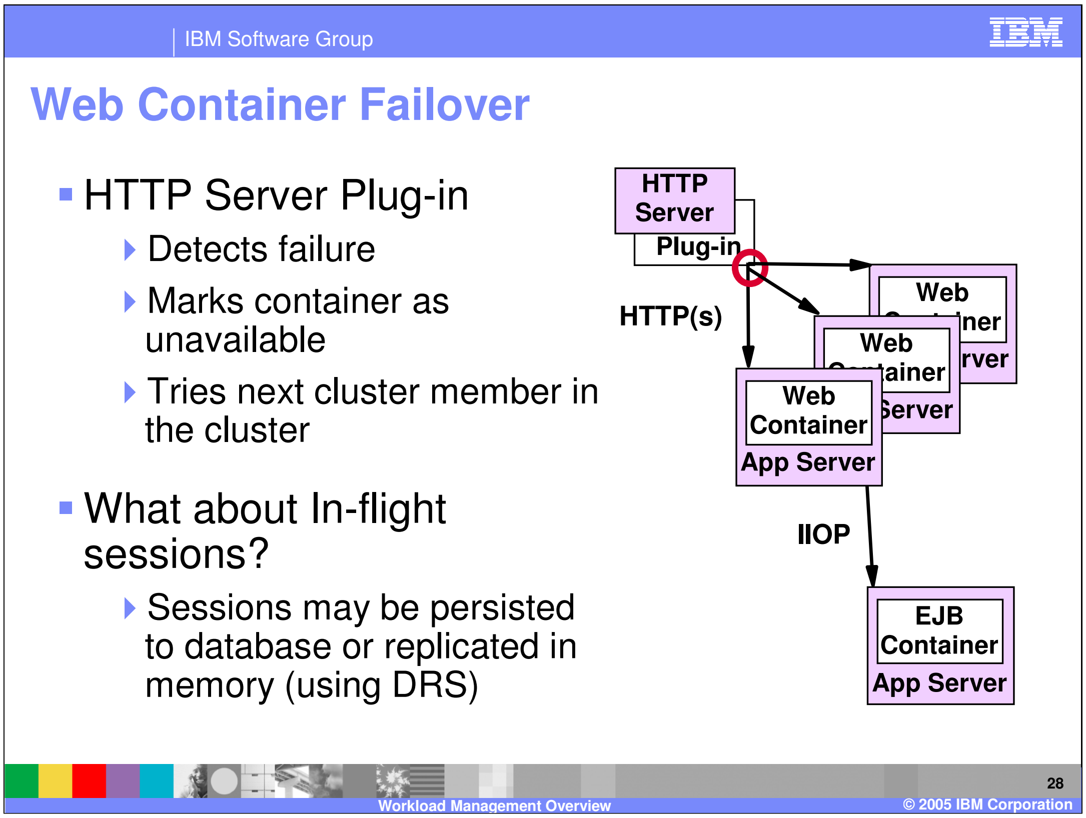
Workload Management Overview

© 2005 IBM Corporation

A typical production environment will have multiple HTTP servers. Each of those HTTP servers will route to multiple WebSphere Application Server instances.

Each plug-in contains configuration information for all of the servers; it has a server list and a back-up server list. If all the servers in the server list are unavailable, it will route to the backup list. This must be manually edited into the plugin-cfg.xml file.

If any HTTP server fails, the Load Balancer will simply route around it. The plug-in reads the cloneID from the session key, and can route the request to its originating server.



Web Container Failover

- HTTP Server Plug-in
  - Detects failure
  - Marks container as unavailable
  - Tries next cluster member in the cluster
- What about In-flight sessions?
  - Sessions may be persisted to database or replicated in memory (using DRS)

If a server process fails, the HTTP server detects the failure and marks that application server as unavailable. It then routes the request to the next cluster member.

Sessions already in progress will have a server ID for that failed server; the HTTP server routes them to the next server. Session data can be handled in two ways. Session Persistence to a Database, or internal messaging of session information.

Database session persistence functions largely as it did in version 4.0.

WebSphere Internal Messaging was new in 5.0, and the details are in the Data Replication Service module. The routing algorithm is a round robin algorithm, so if a server fails, requests for that server will always go to the same 'next' server. This allows you to use the configuration where servers get memory information from only one other server (The 'Buddy' System). This reduces the amount of memory replication necessary in a cluster.

One path through the GUI to session configuration settings is:

Servers->Application Servers -> <Server Name>->Web Container->Session Management->Distributed Environment Settings

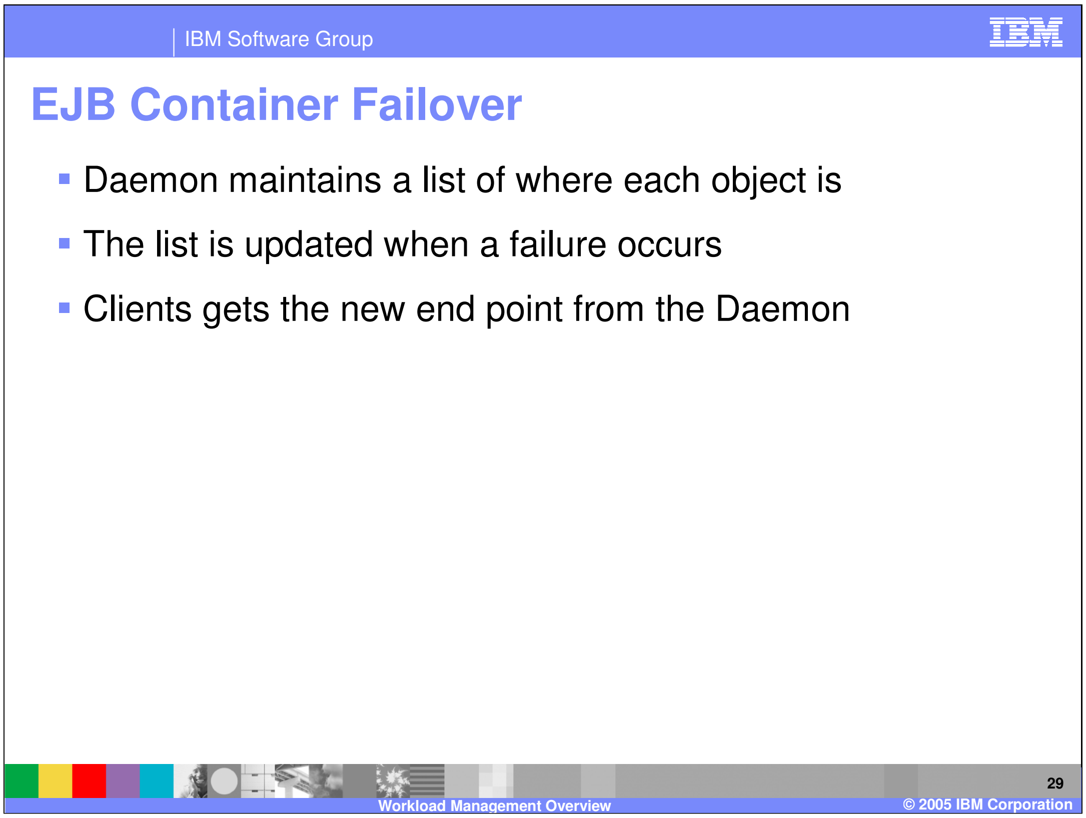
Then either:

->Database

or

->Memory to Memory Replication

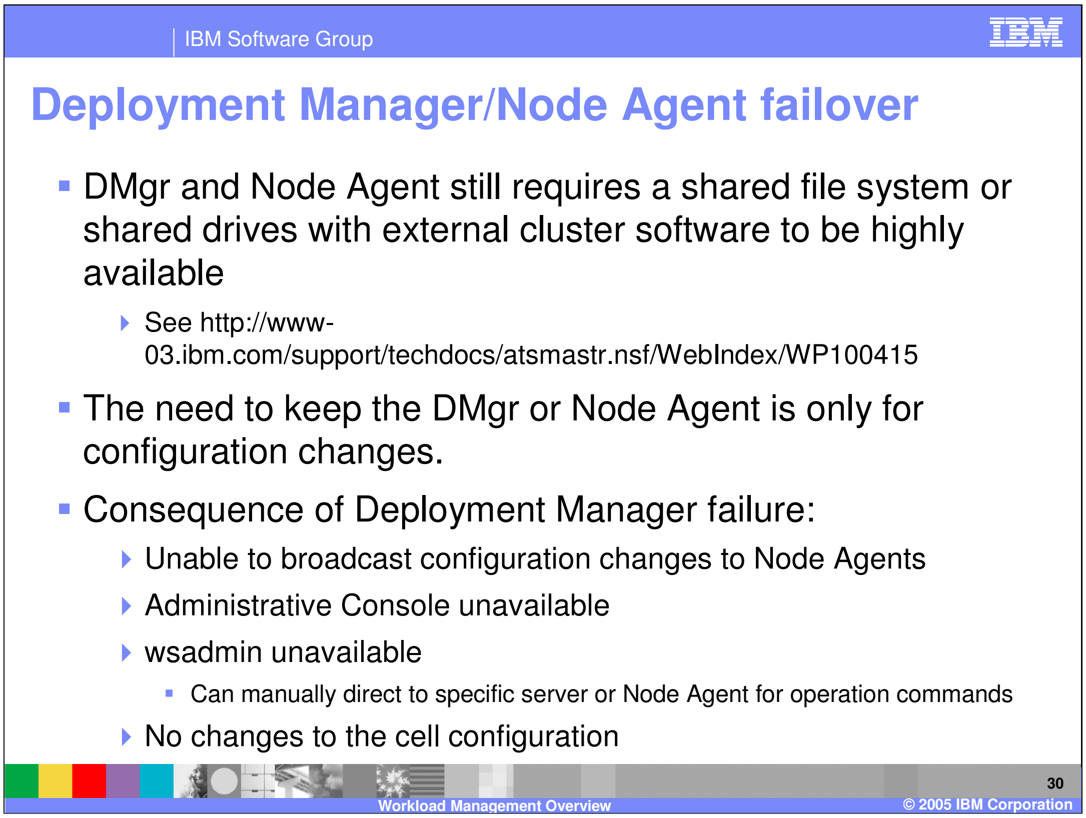
In version 6, the scope of the replication domain is the core group; DRS will coordinate with the WLM component to determine which cluster members should hold backup session information for a specific cluster member.



EJB Container Failover

- Daemon maintains a list of where each object is
- The list is updated when a failure occurs
- Clients gets the new end point from the Daemon

On z/OS platforms, EJB container failover is still handled by the daemon. The system detects a failed server and moves the internal pointers to a new copy of the object on a different server. When an EJB client detects the failure, a locate request is sent to the daemon which returns the new object.

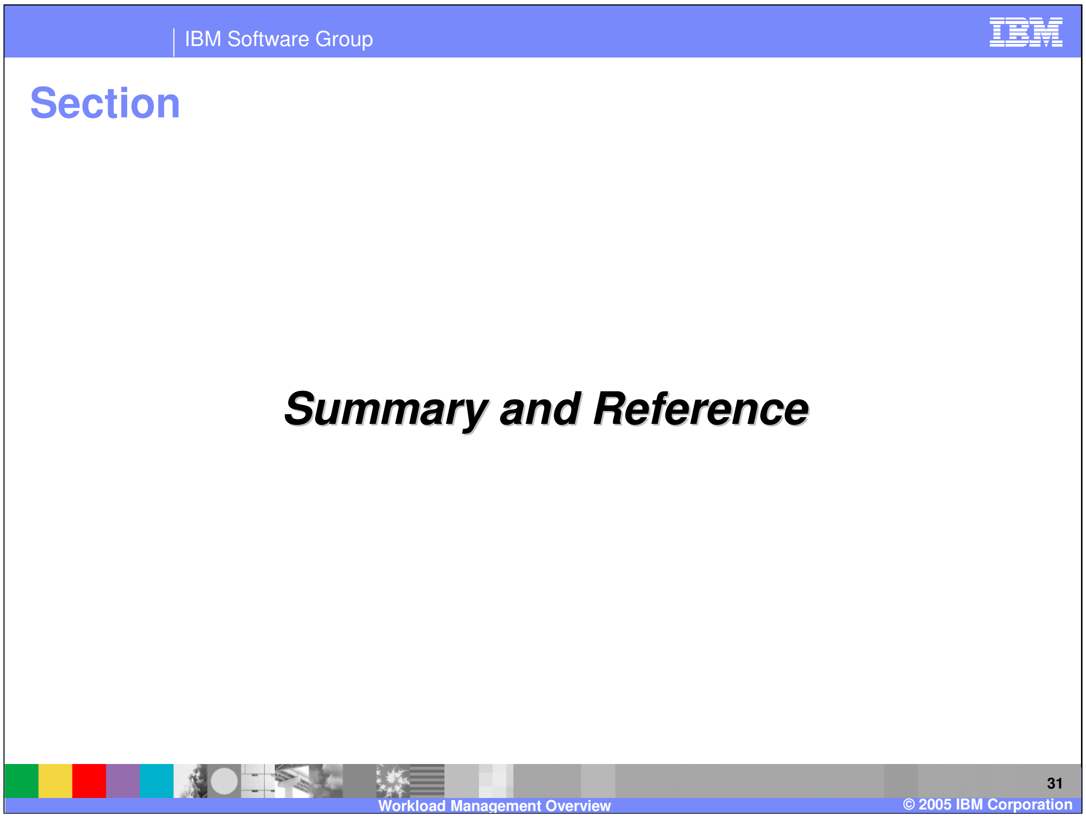


Deployment Manager/Node Agent failover

- DMgr and Node Agent still requires a shared file system or shared drives with external cluster software to be highly available
  - See http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP100415
- The need to keep the DMgr or Node Agent is only for configuration changes.
- Consequence of Deployment Manager failure:
  - Unable to broadcast configuration changes to Node Agents
  - Administrative Console unavailable
  - wsadmin unavailable
    - Can manually direct to specific server or Node Agent for operation commands
  - No changes to the cell configuration

In the event of a Deployment Manager failover.

The Deployment Manager and the Node Agent do not handle client requests, they only handle the configuration repository. Since the configuration information is replicated from the Deployment Manager to the servers through the Node Agents, the only impact of a failed deployment manager or node agent is that administrative configuration changes cannot be passed down to the application servers.



IBM Software Group

Section

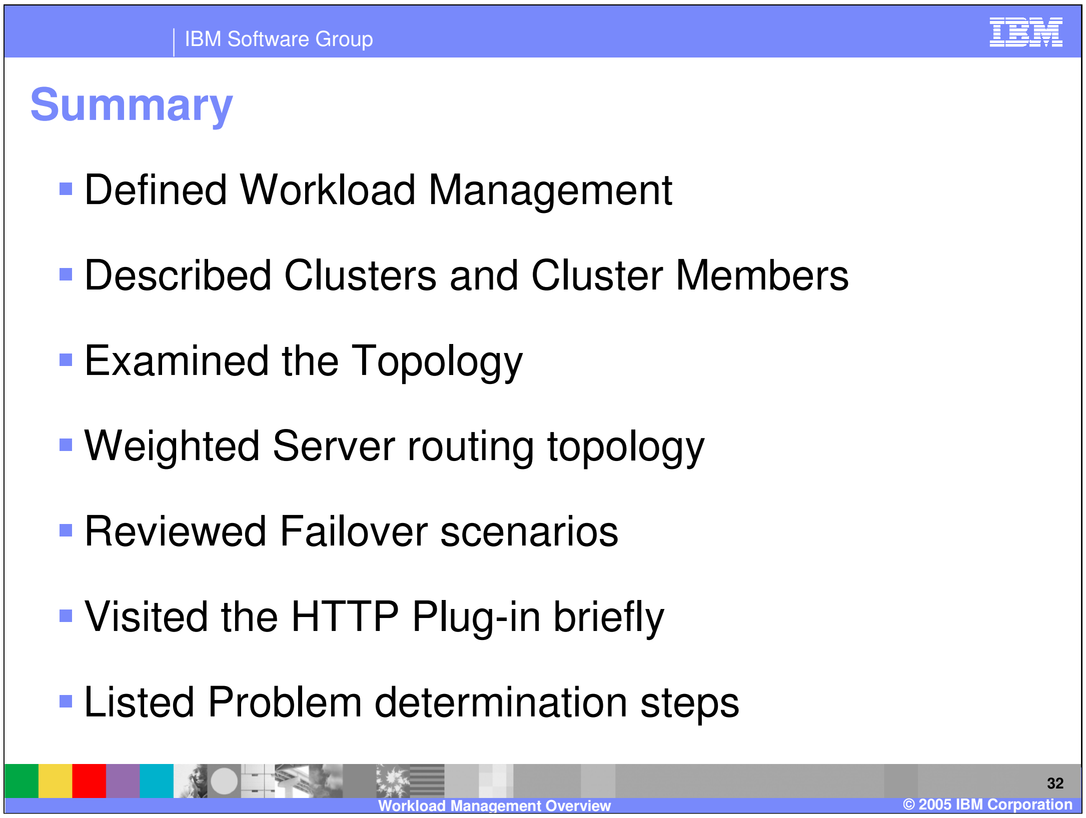
Summary and Reference

31

Workload Management Overview

© 2005 IBM Corporation

This is the summary and reference section.

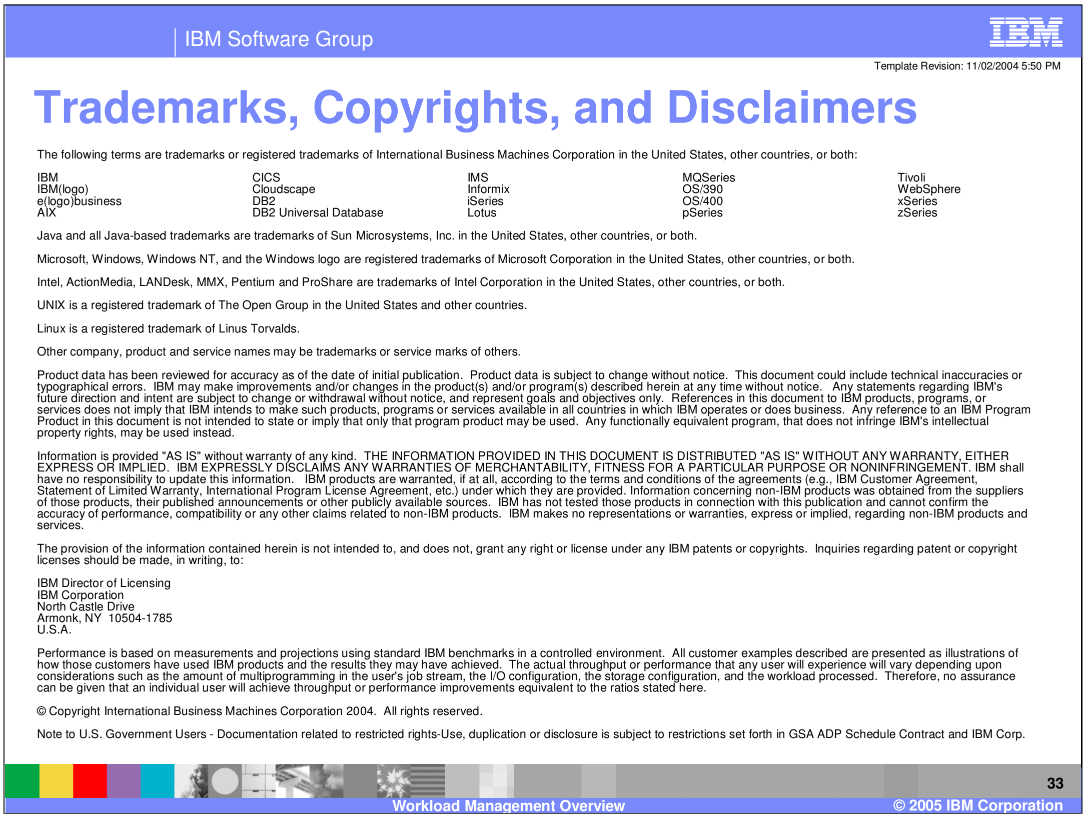


Summary

- Defined Workload Management
- Described Clusters and Cluster Members
- Examined the Topology
- Weighted Server routing topology
- Reviewed Failover scenarios
- Visited the HTTP Plug-in briefly
- Listed Problem determination steps

In summary, this presentation has provided a basic definition of Workload Management. Next, topology was discussed, including where request routing decisions are made. Failover scenarios were also explained, including how various components are configured to avoid a Single Point of Failure (SPOF).

Finally, some problem determination suggestions were offered.



IBM Software Group

Template Revision: 11/02/2004 5:50 PM

Trademarks, Copyrights, and Disclaimers

The following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both:

IBM
IBM(logo)
e(logo)business
AIX
CICS
Cloudscape
DB2
DB2 Universal Database
IMS
Informix
iSeries
Lotus
MQSeries
OS/390
OS/400
pSeries
Tivoli
WebSphere
xSeries
zSeries

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are registered trademarks of Microsoft Corporation in the United States, other countries, or both.

Intel, ActionMedia, LANDesk, MMX, Pentium and ProShare are trademarks of Intel Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Linux is a registered trademark of Linus Torvalds.

Other company, product and service names may be trademarks or service marks of others.

Product data has been reviewed for accuracy as of the date of initial publication. Product data is subject to change without notice. This document could include technical inaccuracies or typographical errors. IBM may make improvements and/or changes in the product(s) and/or program(s) described herein at any time without notice. Any statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only. References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business. Any reference to an IBM Program Product in this document is not intended to state or imply that only that program product may be used. Any functionally equivalent program, that does not infringe IBM's intellectual property rights, may be used instead.

Information is provided "AS IS" without warranty of any kind. THE INFORMATION PROVIDED IN THIS DOCUMENT IS DISTRIBUTED "AS IS" WITHOUT ANY WARRANTY, EITHER EXPRESS OR IMPLIED. IBM EXPRESSLY DISCLAIMS ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NONINFRINGEMENT. IBM shall have no responsibility to update this information. IBM products are warranted, if at all, according to the terms and conditions of the agreements (e.g., IBM Customer Agreement, Statement of Limited Warranty, International Program License Agreement, etc.) under which they are provided. Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products in connection with this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. IBM makes no representations or warranties, express or implied, regarding non-IBM products and services.

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents or copyrights. Inquiries regarding patent or copyright licenses should be made, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. All customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.

Note to U.S. Government Users - Documentation related to restricted rights-Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract and IBM Corp.