



ADI205

## Building Scalable WebSphere Applications

Sumit Ray and George Kosmides  
Noospherics Technologies Inc.

# Who am I?

- ◆ Sumit Ray
  - Sr. WebSphere / J2EE Architect
  - Senior Consultant – Noospherics Technologies
  - WebSphere architecture expertise
  
- ◆ George Kosmides
  - Sr. Enterprise Architect
  - CEO of Noospherics Technologies
  - Extensive experience with:
    - Enterprise architecture and Rational Process
    - J2EE and WebSphere customized training/mentoring
    - Enterprise Integration

# Outline

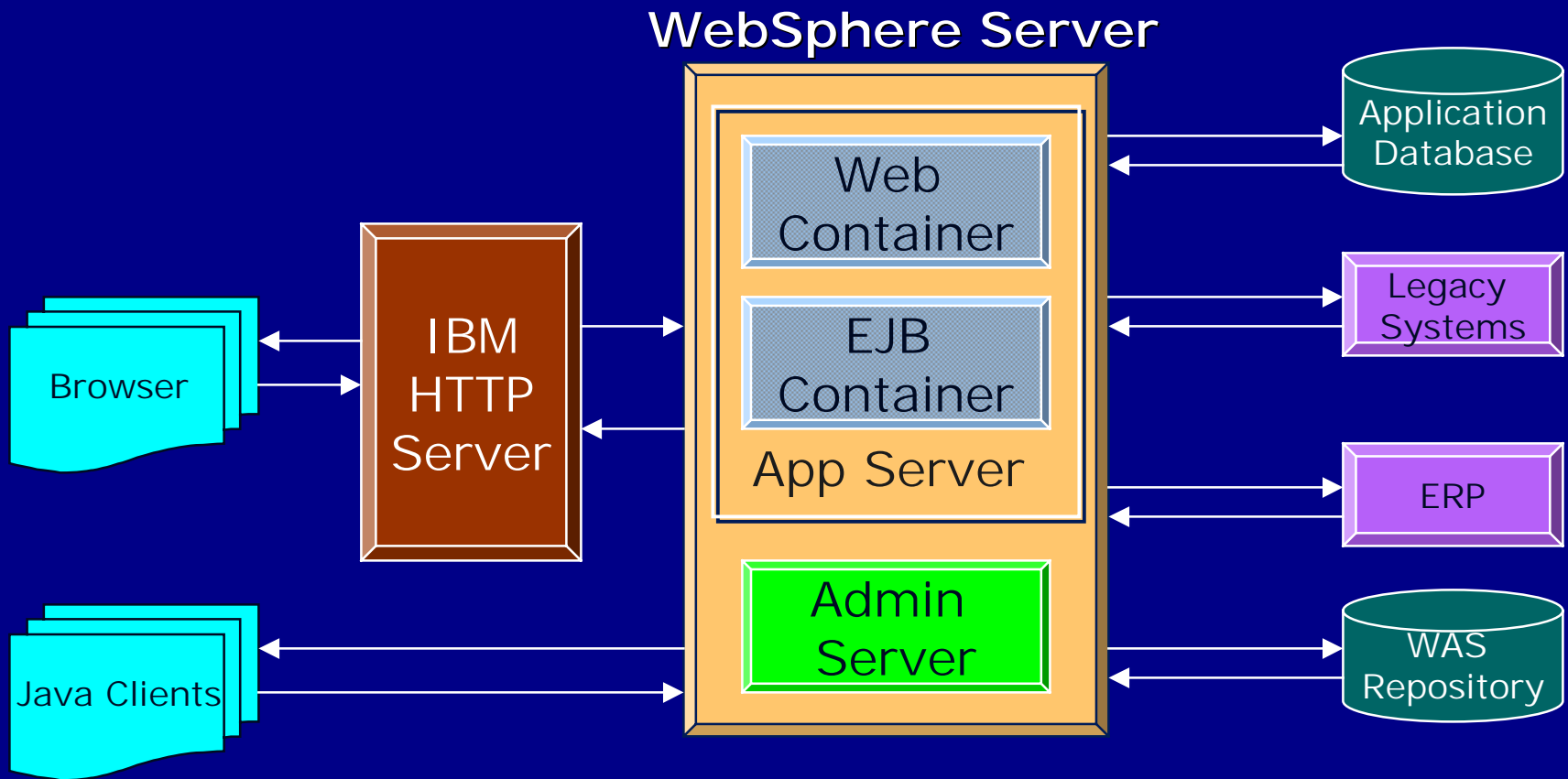
- ◆ Overview of WebSphere Topology Options
  - Domains
  - All-in-one single node applications
  - Distributed tiers – multiple nodes
  - Clusters
  - Model, Clones and Server Groups
  - Horizontal Cloning
  - Vertical Cloning
- ◆ WebSphere Work Load Management
- ◆ WebSphere Performance Best Practices
- ◆ Cluster Administration
  - Domain management

# Outline

- ◆ *Overview of WebSphere Topology Options*
- ◆ WebSphere Workload Management
- ◆ WebSphere Performance Best Practices
- ◆ Cluster Administration

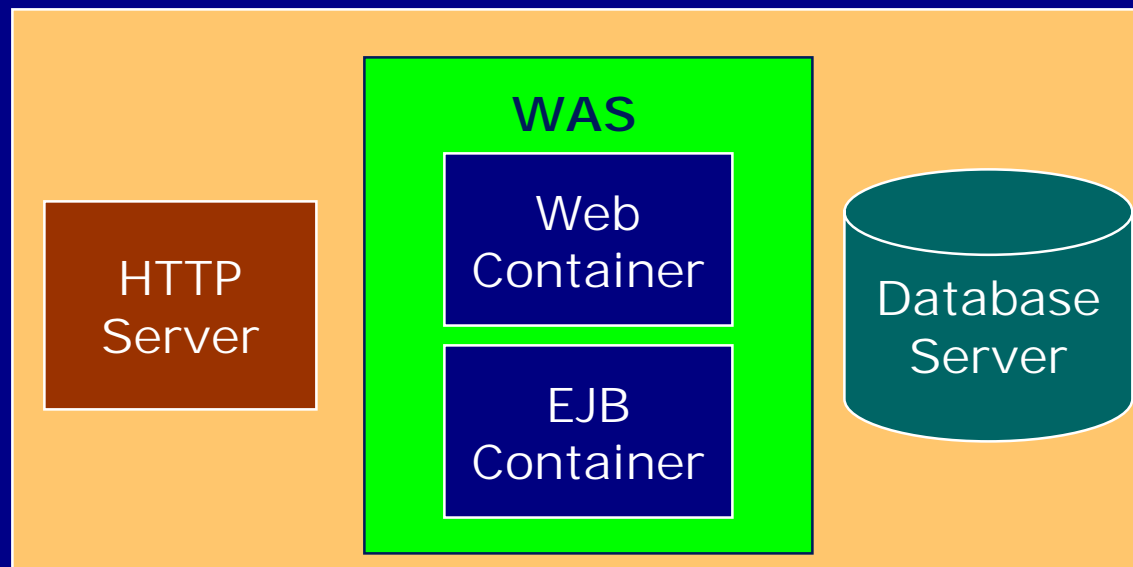


# WebSphere Architecture



# WebSphere Components Distribution

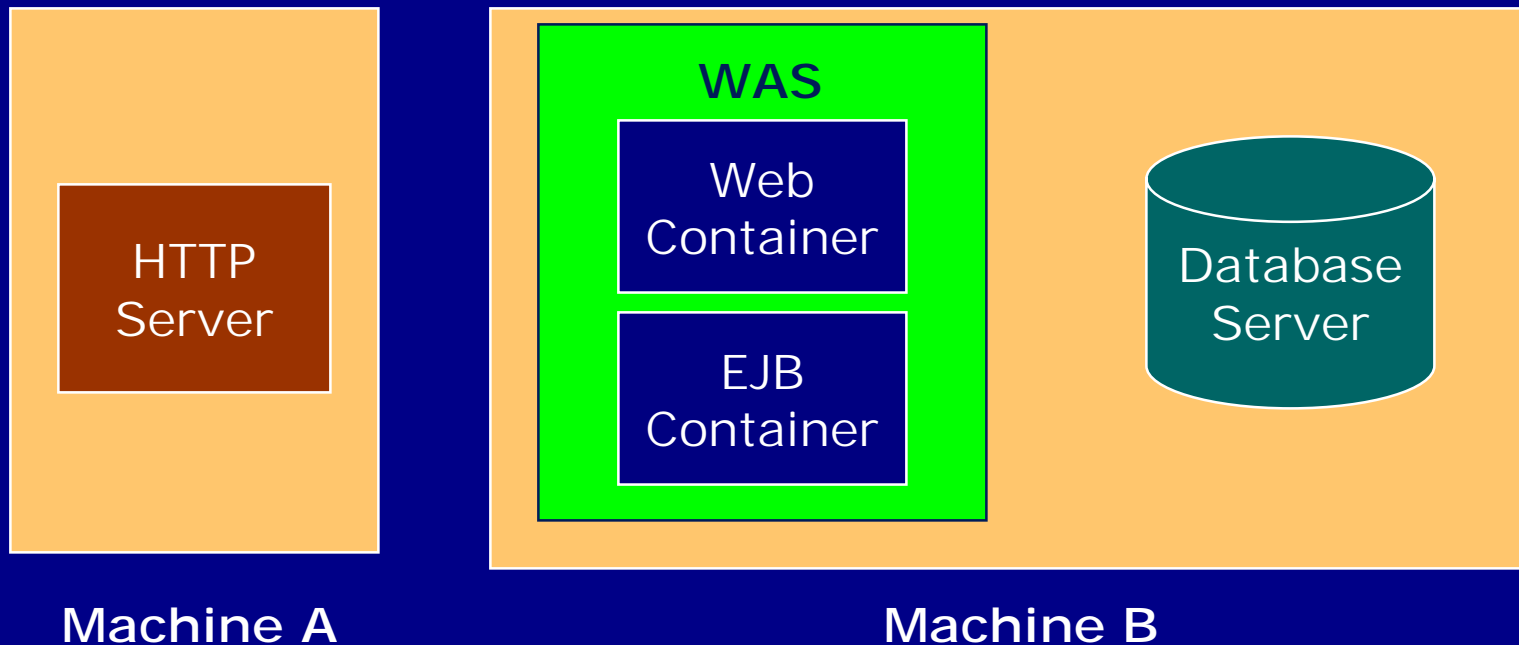
- ◆ Components of WebSphere can be distributed in various ways
- ◆ Simplest is an all-in-one configuration



Machine A

# WebSphere Components Distribution

- ◆ A separate HTTP Server provides added security and better performance since complex processing is delegated to the Application Server

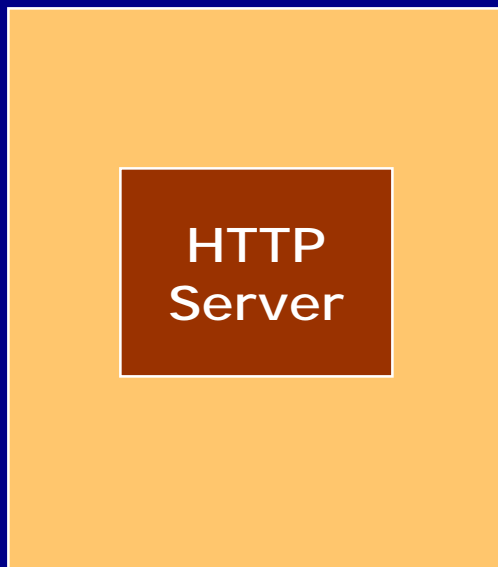


# Remote HTTP Server - Benefits

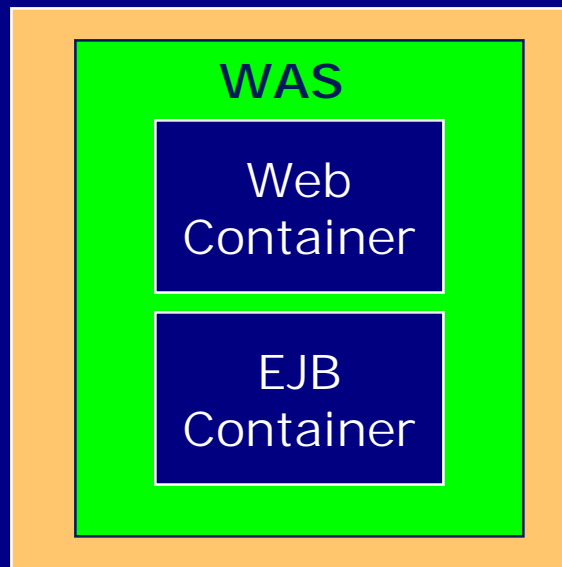
- ◆ Physical separation of the HTTP server from the application server
- ◆ Processes usually across one or more firewalls
- ◆ Very thin WebSphere install on the HTTP server machine
- ◆ Secured Protocol (SSL) for communication between Web Server and Web Container
- ◆ Avoid system resources contention

# WebSphere Components Distribution

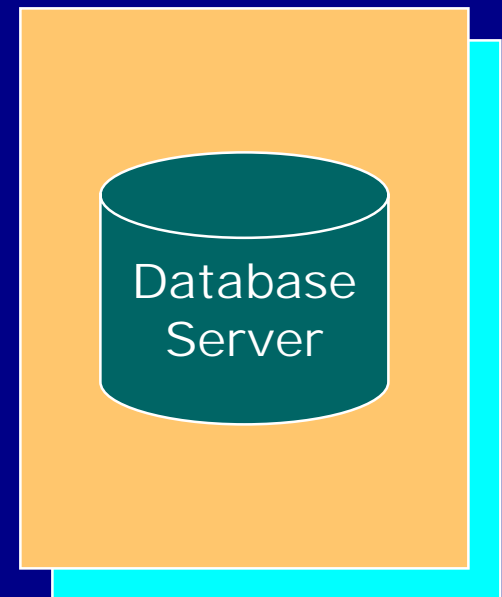
- ◆ It is a good idea to put the administration database on a separate server with application databases where backups are done and hardware based fail-over can be implemented



Machine A



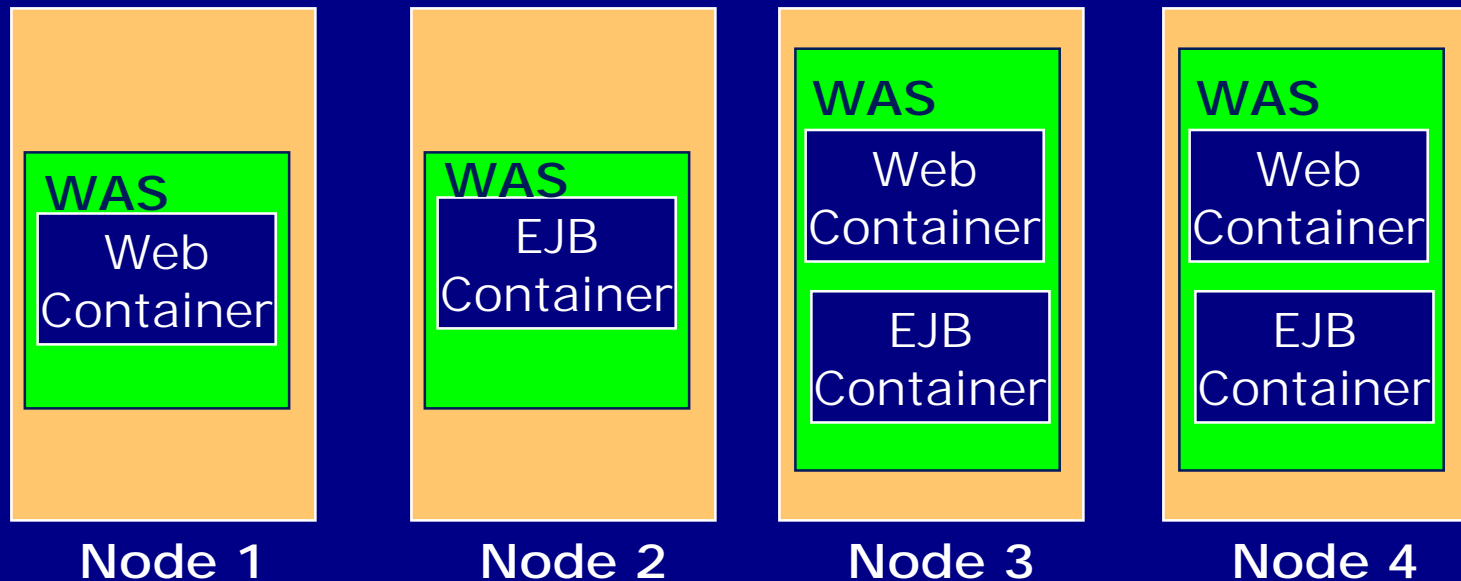
Machine B



Machine C/C'

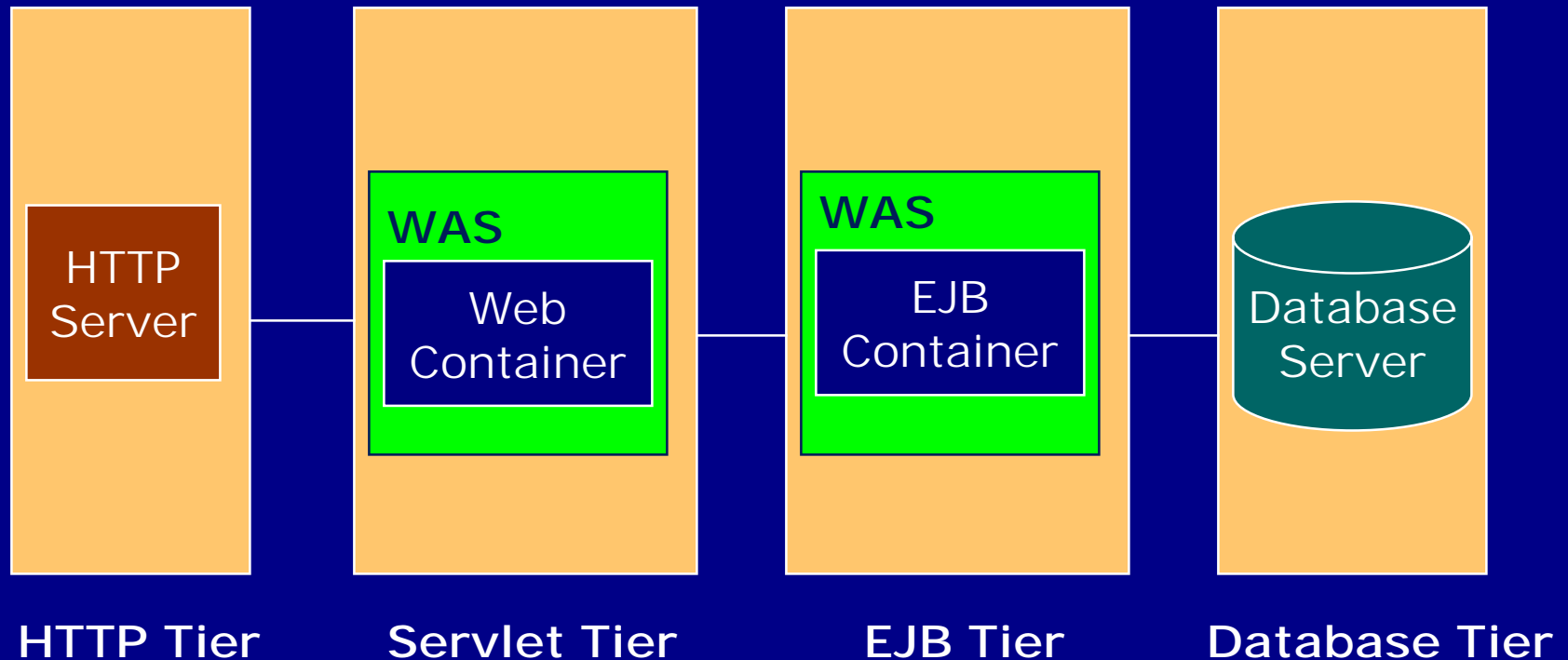
# WebSphere Components Distribution

- ◆ Web and EJB containers can be run on more than one machine
- ◆ Each machine running WAS or one of the containers is referred to as a node



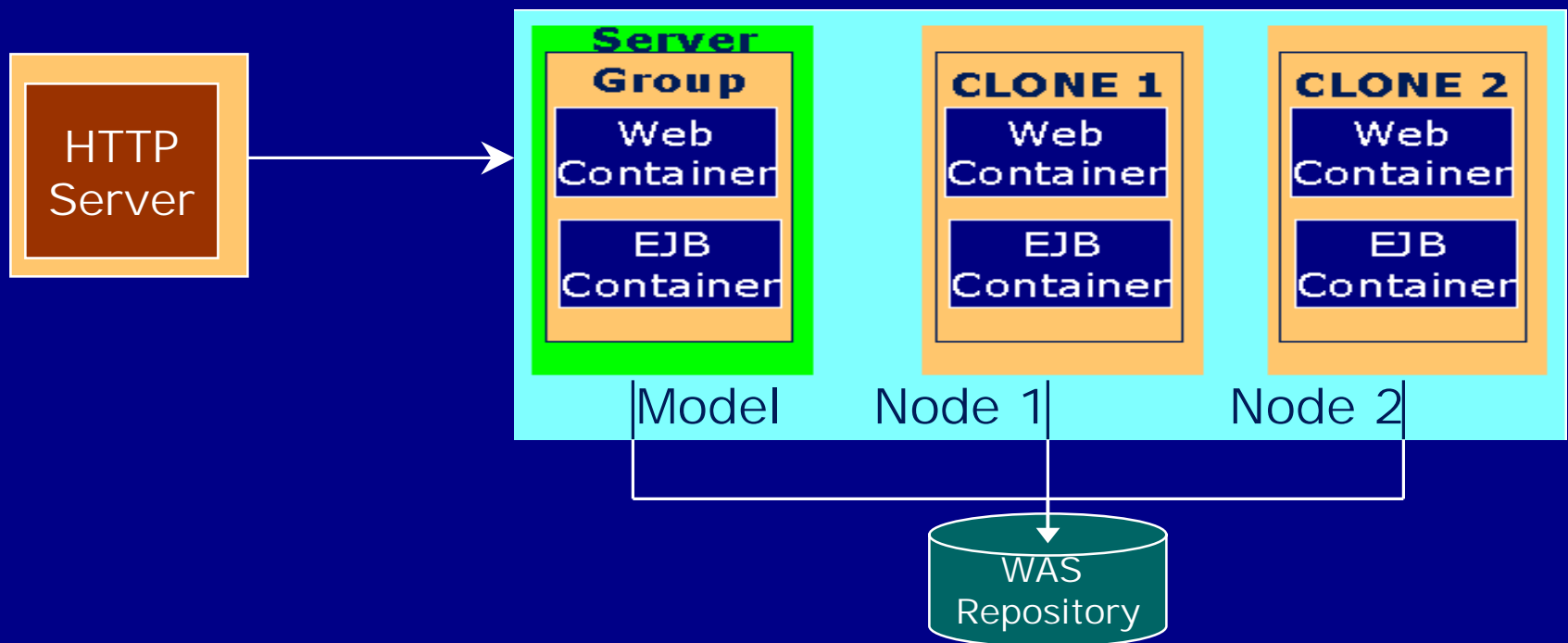
# WebSphere Components Distribution

- ◆ When components are distributed across servers the system is said to be tiered



# Clustering

- ◆ In a tiered system multiple servers can be used at each tier sharing the same WebSphere administrative domain
- ◆ Improves Performance & Throughput
- ◆ Increased Reliability



# Model, Clones, and Server Groups

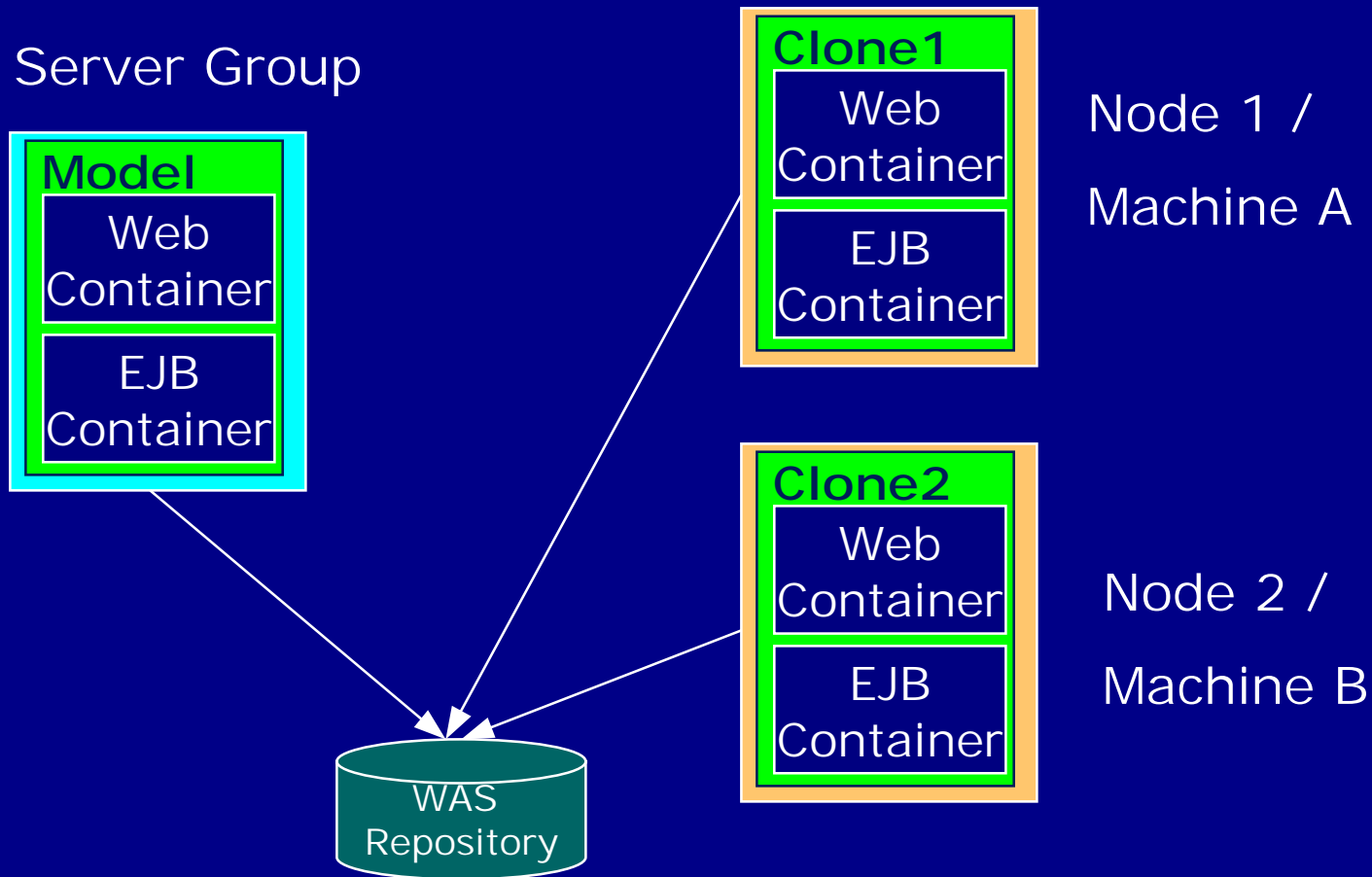
- ◆ Model
  - An application server template for creating copies of a server or process instance (servlet engine or EJB server)
- ◆ Clones
  - Copies of servers or process instances
- ◆ Server groups
  - All clones of a given model comprise a server group
- ◆ Benefits
  - Single Point of Administration
  - Changes Are Propagated to All Clones
  - Start and Stop of Model

# Cloning Considerations

- ◆ Cloning only clones the configuration
  - Executable needs to be migrated to the correct directory on nodes with cloned servers
    - ◆ Share common directory mount (AFS/NFS)
  - Cloned servers must have the same directory structure as the Initial node

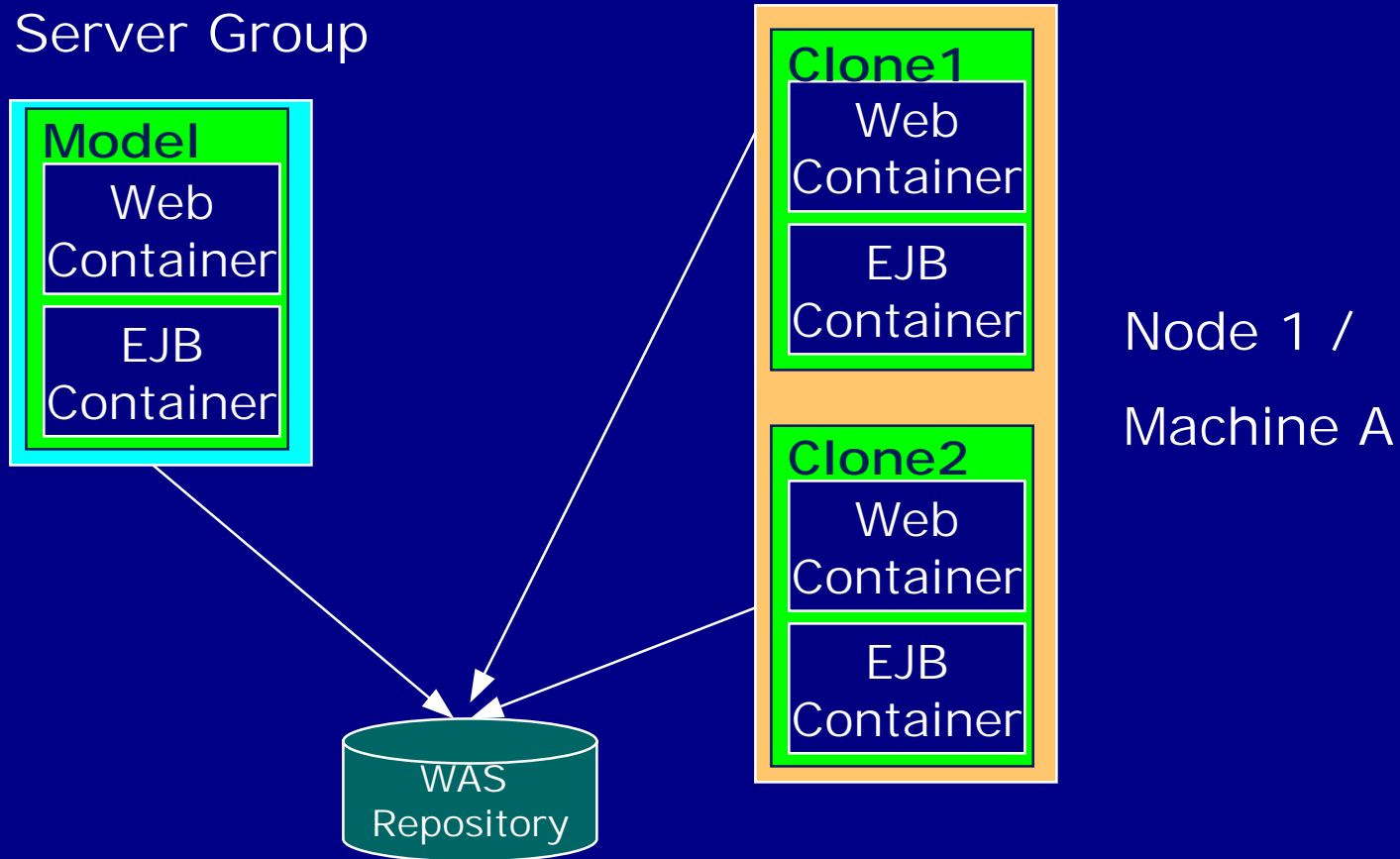
# Horizontal Scaling

- ◆ Failover eliminates Single Point of failure



# Vertical Scaling

- ◆ Multiple JVM's on a Single Large Server Improve Resource Utilization



# Topology Selection Summary

	Performance	Throughput	Maintainability	Availability	Session	Security
Vertical Clones	Limited Benefit	Limited to resources on a single machine	Easiest to maintain	Process isolation	Required	
Horizontal Clones	Best in general	Best in general	Code migration to multiple nodes	Process and hardware redundancy	Required	
HTTP Separate	Better than local	Better than local				Allows for Firewalls/DMZ's
Three Tiers	Typically slower than single JVM	Improved throughput				Most option for Firewall

# Outline

- ◆ Overview of WebSphere Topology Options
- ◆ *WebSphere Load Management*
- ◆ WebSphere Performance Best Practices
- ◆ Cluster Administration



# Objective of Load Management

- ◆ Using WebSphere's built-in capability to build reliable and scalable Applications
- ◆ Point of focus
  - Scalability
    - Performance
    - Throughput
  - Load balancing
  - Failover / Availability

# Scalability

- ◆ Performance

Performance involves minimizing the response time for a given transaction load

- ◆ Throughput

Throughput while related to performance, more precisely defines the number of concurrent transactions that can be accommodated

# Load balancing

- ◆ Load Sharing
  - Fair share of the overall client load that is being processed by the system distributed proportionally across all available servers
- ◆ Load Adaptability
  - If the total load changes over time, the system should naturally adapt itself to maintain this load balancing property

# Failover / Availability

- ◆ Failover
  - Hardware failover
  - Application failover
- ◆ Availability
  - Hardware-based high availability
    - ◆ HACMP
    - ◆ Sun Cluster
    - ◆ MC/ServiceGuard
  - Software-based high availability

# Work Load Management

- ◆ WLM
  - What is it ?
  - What can be WLM'd?
  - Models, Clones and Server Groups
  - WLM - how does it work?
  - Plug-in WLM/ Servlet clustering architecture
  - EJB WLM
  - WLM considerations

# Work Load Management - What Is It?

- ◆ Process of spreading multiple request
  - Effective Use of the available Computing Resources
- ◆ Procedure for improving
  - Scalability
  - Performance
  - Availability / Reliability

# What Can Be WLM'd?

- ◆ Servlet requests
  - Distributed across multiple Web containers
- ◆ EJB requests
  - Distributed across multiple EJB containers
- ◆ Servlet Redirector
  - Not supported in WebSphere V4.0

# WLM - How Does It Work?

- ◆ Plug-in WLM/ Servlet clustering architecture
- ◆ Enterprise Java Services WLM (EJS WLM)/ EJB WLM
- ◆ Session affinity
- ◆ Target server is selected among all available clones
  - Automatically skips clones that are down
- ◆ Always Selects target in same JVM if possible
- ◆ Four basic policies
  - Round-robin
  - Round-robin prefer local
  - Random
  - Random prefer local
- ◆ Slower performance
- ◆ Higher reliability

# WLM - How Does It Work? - Servers

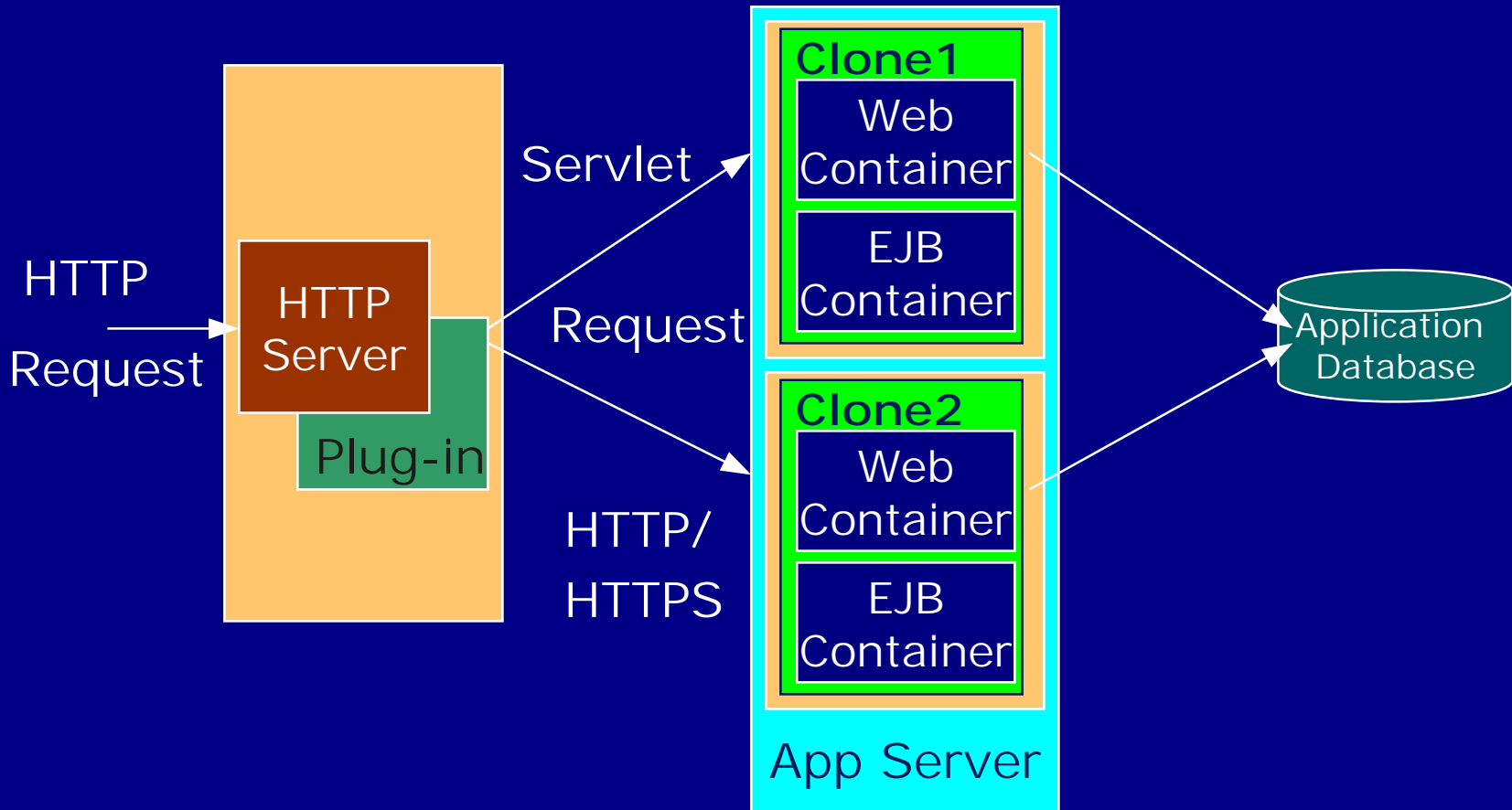
- ◆ WLM Server Runtime
  - Keeps track of the list of available clones by polling the adminserver
  - Checks Model Epoch number on each request from clients
  - Provides client with updated model status whenever necessary

# WLM - How Does It Work? - Clients

- ◆ WLM Client Runtime
  - Server Information Manager keeps track of available servers, using information Piggybacked on response messages from servers
  - Proxy Manager maintains a list of slave proxies for each potential target server
  - Proxy Selector controls target selection
  - Affinity Manager helps in selecting target

# Plug-in WLM/ Servlet clustering architecture

- ◆ Between the HTTP server plugin and the cloned application servers using HTTP

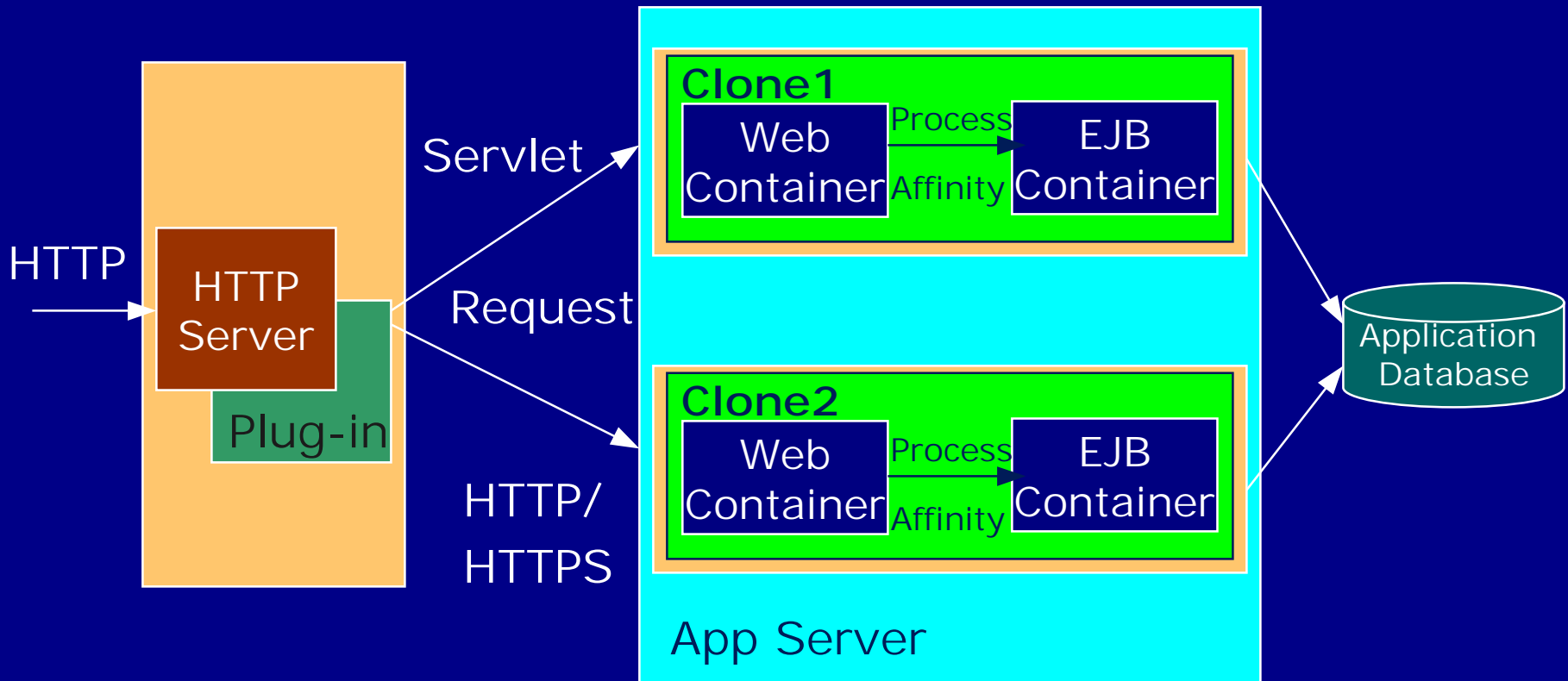


# Plug-in WLM and fail-over

- ◆ Local or Remote HTTP Server
- ◆ Auto-reload of Web module
- ◆ Refresh Interval of the plug-in
- ◆ Session Affinity is provided by plug-in
- ◆ Plug-in workload management and failover policies
  - Round robin
  - Random
- ◆ Process Affinity -
  - make in-process calls to EJBs in the same application server

# EJB Workload Management

- ◆ Between the Web container and the cloned EJB container using RMI/IIOP



# EJB Workload Management

- ◆ The workload management service provides load balancing for the following types of enterprise beans
  - All clones of the home object of an entity or session bean
  - All clones of an instance of a specific entity bean or stateless session bean

Note: The stateful session bean is the only EJB which is not subject to workload management

# EJB Workload Management

- ◆ Stateless session bean served at random from the pool of instances maintained by the container
- ◆ State information for stateful session bean cannot be maintained across multiple application server clones
- ◆ Client request is directed towards an instance of Entity bean for the duration of transaction – Transaction affinity
- ◆ Between transactions the state of Entity bean can be cached

Option	Activate	Call ejbLoad()	WLM
A	Once	Once	No
B	Once	At start of transaction	Yes
C	At start of transaction	At start of transaction	Yes

# EJB Workload Policies

- ◆ EJB Workload Policies
  - Round robin
  - Random
  - Prefer local
- ◆ Process Affinity
  - All in-process requests are redirected to the same client
  - no need of a serialization for method calls
- ◆ Transaction Affinity
  - All the request for a unit of transaction will be routed to the same clone

Note: Whatever the selection policy, the process affinity and transaction affinity will always override the selection policy

# WLM RMI C Compiler

## WLMJAR utility

- Deprecated for WebSphere Application Server V4.0
- Cloned enterprise beans automatically participate in WLM

# WLM Considerations

- ◆ WLM is not completely transparent to the application for failover situations.
  - Comm failure is thrown with a completed "maybe" Minor Code
  - Mapped to `java.rmi.RemoteException`
  - The application has to expect possible failures and initiate the retry
    - ◆ Because WLM cannot know if operation partially completed or not
    - ◆ WLM Will then try to find a surviving server

# Outline

- ◆ Overview of WebSphere Topology Options
- ◆ WebSphere Load Management
- ◆ *WebSphere Performance Best Practices*
- ◆ Cluster Administration

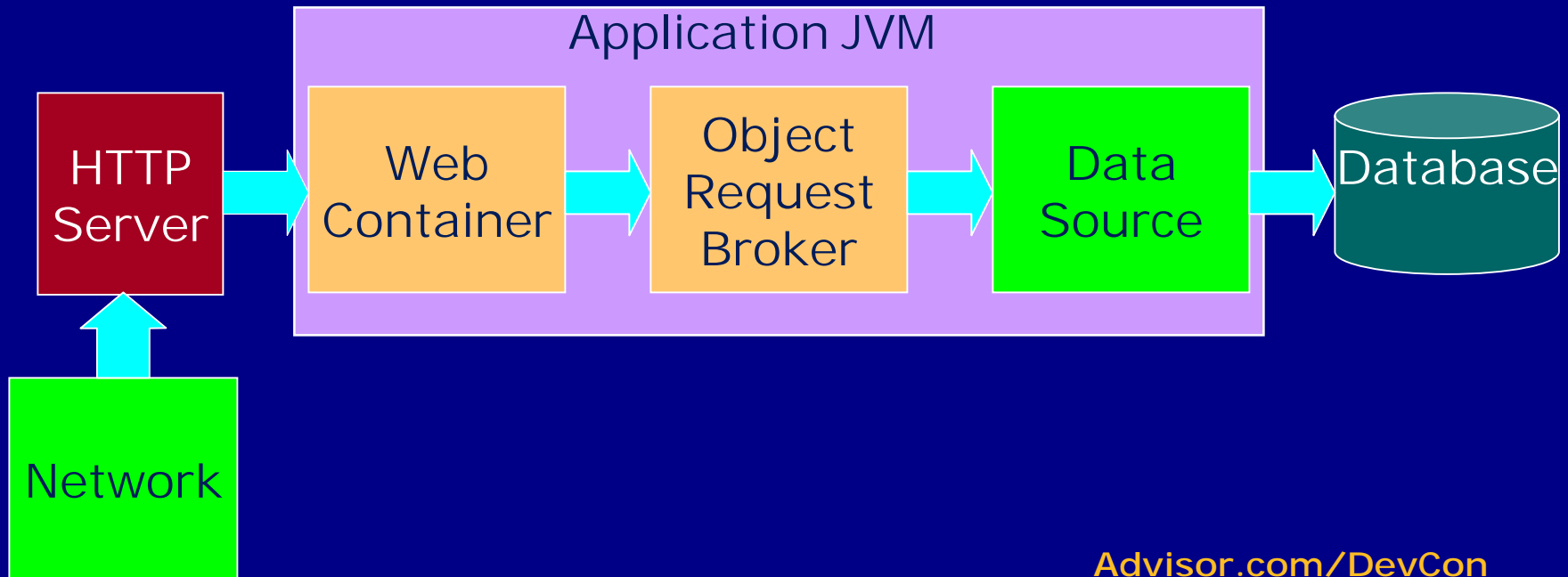


# Performance Tuning

- ◆ Performance Tuning Points
  - Application
  - Operating System
  - Web (HTTP) Server
  - JDK/JVM
  - Web Container
  - Object Request Broker
  - Session Management
  - Database Connection Pool

# Tuning Methodology

- ◆ “Outside In”
  - HTTP Server
  - Application Server
    - ◆ Web Container
    - ◆ Object Request Broker
  - Database and DB Connections



# Application Tuning

- ◆ Biggest Single Factor in Application Performance
  - Use an Application Profiler
    - ◆ Identify and Remove Bottlenecks
  - Reuse Initial Context rather than creating new ones
    - ◆ Each JNDI lookup results in 4 reads of BINDINGBEANTBL as the Admin Server traverses the JNDI namespace.
  - Avoid more than one stateful session EJB per EJB model
  - Client-facing EJB should be stateful with those behind being marked as stateless.
    - ◆ This avoids giving the EJB container unnecessary work to do in managing these instances.

# Operating System Tuning

- ◆ AIX
  - File Descriptors
    - ◆ If too low, you'll receive a "memory allocation error"
    - ◆ Default on AIX is 2000
    - ◆ "*ulimit -a*" (to determine current)
    - ◆ "*ulimit -n*" (to specify a number)
    - ◆ "*ulimit -unlimited*" (for large SMP machines)

# Operating System Tuning

- ◆ Solaris
  - File Descriptors
    - ◆ Set to at least 1024
    - ◆ "ulimit -n 1024"
  - TCP Connection Close
    - ◆ default time wait interval is 240000 ms
    - ◆ set as low as 30000 ms
    - ◆ *ndd -get /dev/tcp tcp\_close\_wait\_interval* (to view the current setting)
    - ◆ *ndd -set /dev/tcp tcp\_close\_wait\_interval 60000* (to change the setting)

# Operating System Tuning

- ◆ NT
  - Virtual Memory
    - ◆ 1.5 to 2 times physical memory for machines with 256Mb or less
    - ◆ Equal to physical memory for machines greater than 256Mb

# HTTP Server Tuning - IBM HTTP Server

- ◆ MaxClients - 50 (default is 150) on UNIX  
ThreadsPerChild on NT
  - Don't increase if CPU is 100%
  - Test with 40 and 60
- ◆ MinSpareServers - 5 (default is 5) on UNIX
- ◆ MaxSpareServers - 50 (same as MaxClients - default is 10) on UNIX
- ◆ StartServers - 50 (same as MaxSpareServers ) on UNIX

# JVM All Platforms

- ◆ Heap Size
  - MX - maximum heap size
  - MS - minimum heap size
  - Set MX to 1/4 Physical Memory
  - Set MS to 1/2 MX
  - MS should not exceed Physical Memory available
- ◆ Use `-verbosegc` to fine tune and analyze heap
- ◆ JIT and Garbage Collection
  - JIT on (default)

# Web Container Properties

- ◆ Set Web module Auto-reload to false
- ◆ Max Thread Size **less than/equal** to the number of threads or processes that are running within the Web server
- ◆ Thread Inactivity Timeout keep low but not zero (Default 10 sec)
- ◆ Do not allow the number of threads to increase beyond the maximum size configured for the pool

# ORB Properties

- ◆ ORB Threads Pool Size
  - A thread is needed for each EJB request
  - Enterprise beans are typically invoked from servlets in another JVM, using RMI/IIOP and remote EJB client applications, using RMI/IIOP. The ORB thread pool size should accommodate both request sources
- ◆ Pass by Reference
  - only if appropriate for your application
  - can break remote transparency, since you can modify objects passed to an EJB method

# EJB Container Properties

- ◆ Deploy EJB's in a Single Container
- ◆ Container Cache
  - Estimate the number of active beans and Cache Size (buckets in hash table)
  - Cache clean-up interval (increase as cache increases)

# Session Management Properties

- ◆ Close finished sessions promptly
  - Invalidate Timeout: 10 mins (default 30 mins)
- ◆ Maximum in Memory session count
  - Set equal to max number of concurrent users
- ◆ Database Persistence
  - Write frequency set to "End of servlet service"
  - Write contents set to "Only updated attributes"
  - Specify session database cleanup schedule to least load periods of the day
  - Configure a separate DB for sessions
  - Configure and tune separate datasource for sessions

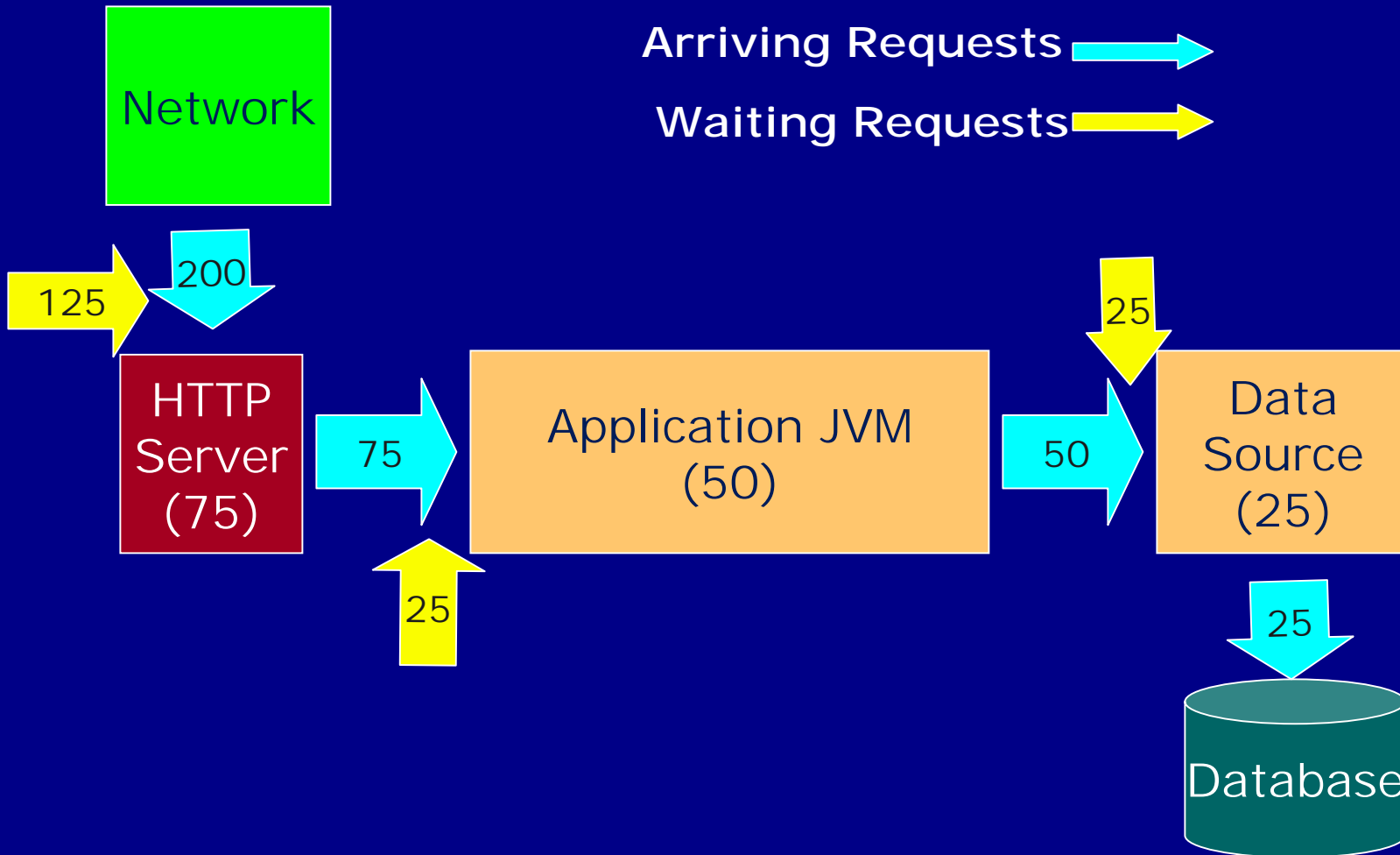
# Database Connection Pool Properties

- ◆ Set Minimum Pool Size to 1 connection
- ◆ Set Maximum Pool Size = Total number of connections used by the Application X Concurrent clients
- ◆ Set Connection Timeout default to 180 seconds
- ◆ Set Idle Timeout to lower like 10 seconds
- ◆ Orphan Timeout to lower like 10 seconds
- ◆ Statement Cache Size large enough for all prepared statements

Statement Cache = The number of SQL prepared statements in Application X maximum number of configured data source connections

- ◆ Enable Auto Connection Cleanup

# WebSphere Queuing - example



# Outline

- ◆ Overview of WebSphere Topology Options
- ◆ WebSphere Load Management
- ◆ WebSphere Performance Best Practices
- ◆ *Cluster Administration*

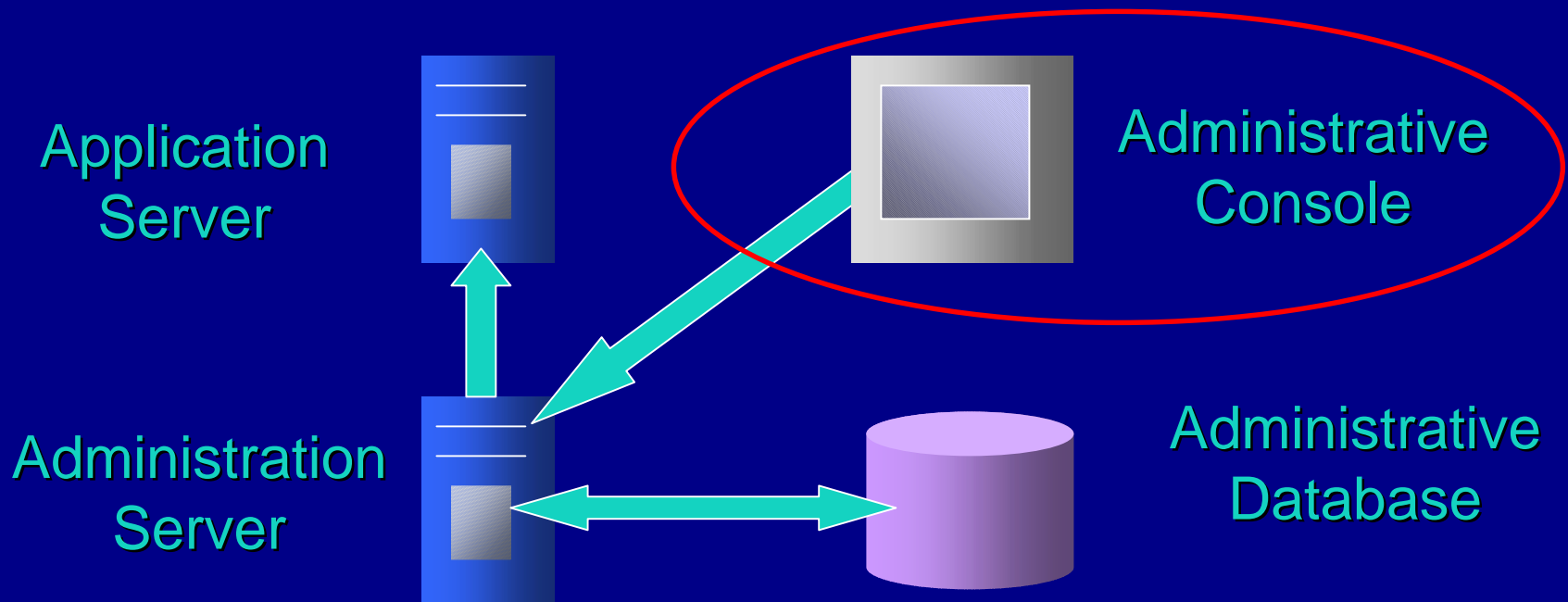


# WebSphere Administration

- ◆ WebSphere Administration involves tracking information such as:
  - Which nodes are running WebSphere
  - What applications are defined in the system
- ◆ And performing tasks such as:
  - Starting and stopping application servers
  - Defining security in the system
  - Creating copies of servers (clones)
  - Deploying applications

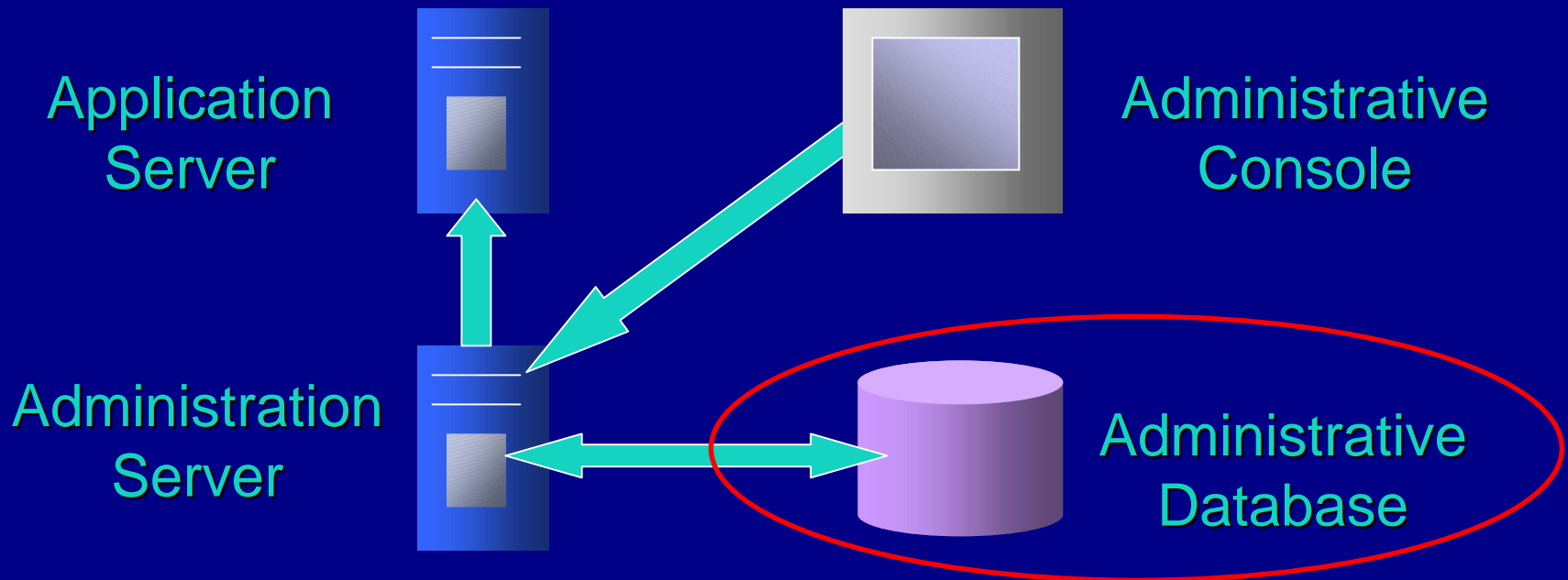
# WebSphere Administration

- ◆ The Administration Console provides the user with a high-level view of this information and an easy way to perform tasks



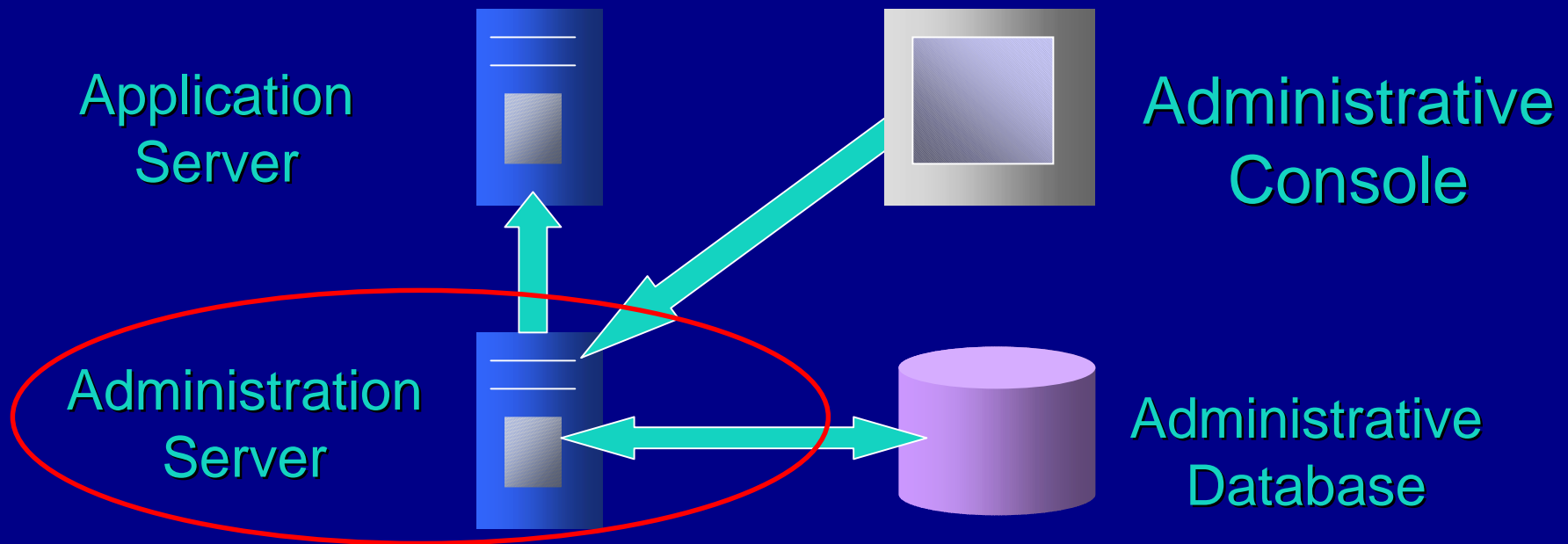
# WebSphere Administration

- ◆ All runtime configuration information is kept in a single persistent database



# WebSphere Administration

- ◆ A separate administration server process accesses the database and the application server or servers



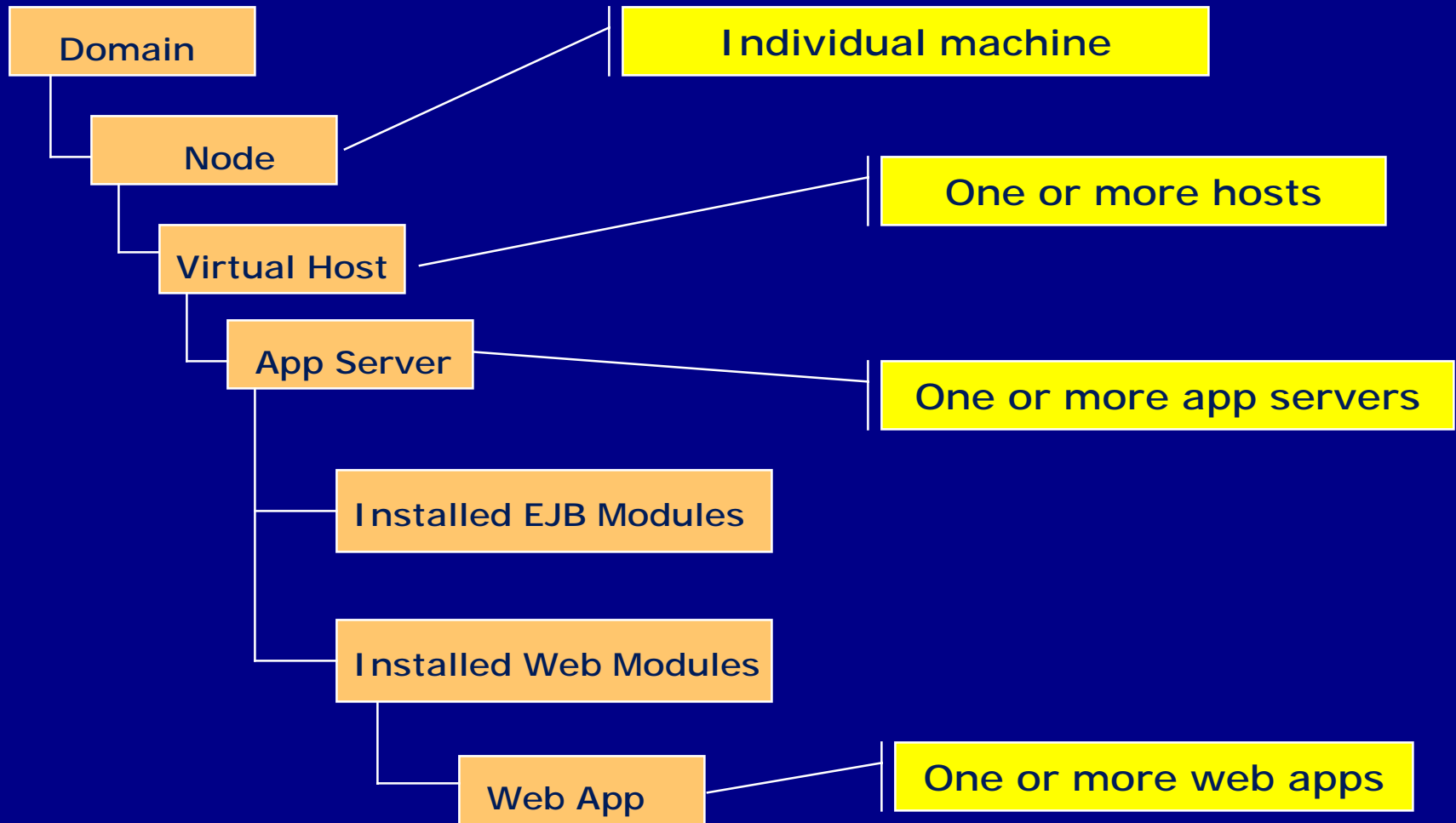
# WebSphere Administration

- ◆ The Administration Console shows a hierarchical view of the information

The screenshot displays the WebSphere Advanced Administrative Console interface. On the left, a tree view shows the hierarchy: WebSphere Administrative Domain > Nodes > jupiter > Application Servers > Default Server. The main panel shows the configuration for the selected server, with tabs for General, Advanced, File, Transaction, JVM Settings, Services, and Custom. The General tab is active, showing fields for Application Server name (Default Server), Node (jupiter), Environment (Environment...), Working directory (C:\WebSphere\AppServer\bin), Node startup state (Last state), Maximum startup attempts (5), and Module visibility (module). Below the configuration panel is an event log table.

Type	Time	Event Message	Source	Options...
Info	4/8/02 1:19 AM	JORB0012: Pass by reference has been set to: true (NoLocalCopies = true)	CRBRas	Options...
Info	4/8/02 1:20 AM	WESVR0023: Server __adminServer open for e-business	com.ibm.wes.runtime.Server	Details...
Info	4/8/02 1:21 AM	Console Ready.		Clear
Info	4/8/02 1:22 AM	Command "Default Server.start" running ...		
Info	4/8/02 1:22 AM	ADM50008: Starting server: Default Server	com.ibm.ejs.sm.active.ActiveServerProcess	
Info	4/8/02 1:22 AM	ADM50032: Started server: Default Server (pid 1784)	com.ibm.ejs.sm.active.ActiveServerProcess	
Info	4/8/02 1:22 AM	Command "Default Server.start" completed successfully.		

# WebSphere Administration



# WebSphere Administration

- ◆ The top level of the Admin. Console shows a Domain

The screenshot displays the WebSphere Advanced Administrative Console interface. The left-hand tree view shows the domain structure, with 'WebSphere Administrative Domain' at the top level, circled in red. Below it are 'Virtual Hosts', 'Server Groups', 'Nodes', and 'Application Servers'. The 'Default Server' is selected under 'Application Servers'. The right-hand pane shows the configuration for the 'Default Server', including fields for 'Application Server name', 'Node', 'Environment', 'Working directory', 'Node startup state', 'Maximum startup attempts', and 'Module visibility'. The bottom pane shows a log of events, including messages about the server starting successfully.

Type	Time	Event Message	Source
Info	4/8/02 1:19 AM	JORB0012: Pass by reference has been set to: true (NoLocalCopies = true)	ORBRas
Info	4/8/02 1:20 AM	WSVR0023: Server __adminServer open for e-business	com.ibm.ws.runtime.Server
Info	4/8/02 1:21 AM	Console Ready.	
Info	4/8/02 1:22 AM	Command "Default Server.start" running ...	
Info	4/8/02 1:22 AM	ADMS0008: Starting server: Default Server	com.ibm.ejs.sm.active.ActiveServerProcess
Info	4/8/02 1:22 AM	ADMS0002: Startat server: Default Server (pid 1784)	com.ibm.ejs.sm.active.ActiveServerProcess
Info	4/8/02 1:22 AM	Command "Default Server.start" completed successfully.	

# WebSphere Administration

- ◆ Domain
  - A domain is the unit of administration in WebSphere
  - All the information in a Domain is kept in a single Administration Database
  - Domains contain nodes, applications and other resources

# WebSphere Administration

- ◆ Multiple Domains
  - Multiple domains can be set up in WebSphere
  - Sometimes this is done for testing vs. production
  - Sometimes for redundancy in production
  - Requires an HTTP sprayer such as IBM's Network Dispatcher

# WebSphere Tools Administration

- ◆ XMLConfig Tool
  - Command line tool to export (full or partial) /import configuration files from and to the WebSphere Repository
  - Repetitive common administrative tasks like start, stop, restart, create, update, delete, enable, disable etc. etc. for resources can be saved as XML files to be used and reused without using the admin console
  - Helps to move configurations from one environment to another e.g. from the development box to the test box

# WebSphere Tools for Administration

- ◆ WSCP (WebSphere Control Program) Tool
  - Command line and scripting interface to administer resources in WebSphere
  - All admin console tasks can be performed
  - Based on Tool Command Language (TCL), a scripting language with simple and easy syntax
  - Automates the administrative tasks and can be saved as scripts for reuse to avoid perform repetitive, long and complex tasks through admin console

# Resources

- ◆ A pdf version of the presentation will be available for download at

[www.noospherics.com/websphere/devcon](http://www.noospherics.com/websphere/devcon)

# ADVISOR DEVCON Web Update Page

[www.Advisor.com/CIW0204p.nsf/w/ciwupd](http://www.Advisor.com/CIW0204p.nsf/w/ciwupd)

# Thank you!

Please remember to fill out your evaluation.