SHARE in Anaheim
March 2011

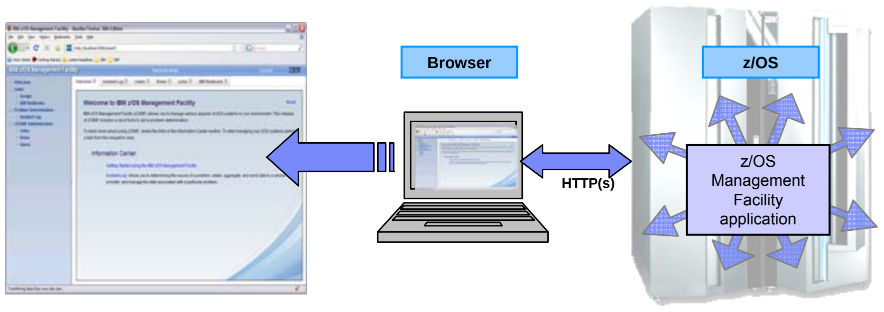
Understanding the WebSphere App Server OEM Edition for z/OSMF Sysprogs

Session 9061

Glenn Anderson, IBM Technical Training

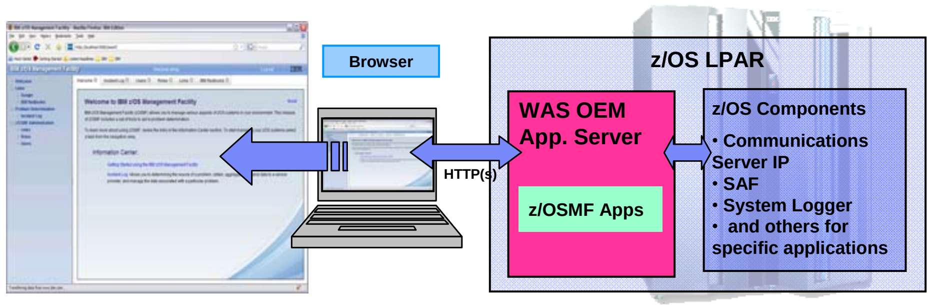
Authorized IBM Training

IBM z/OS Management Facility
z/OS application, browser access



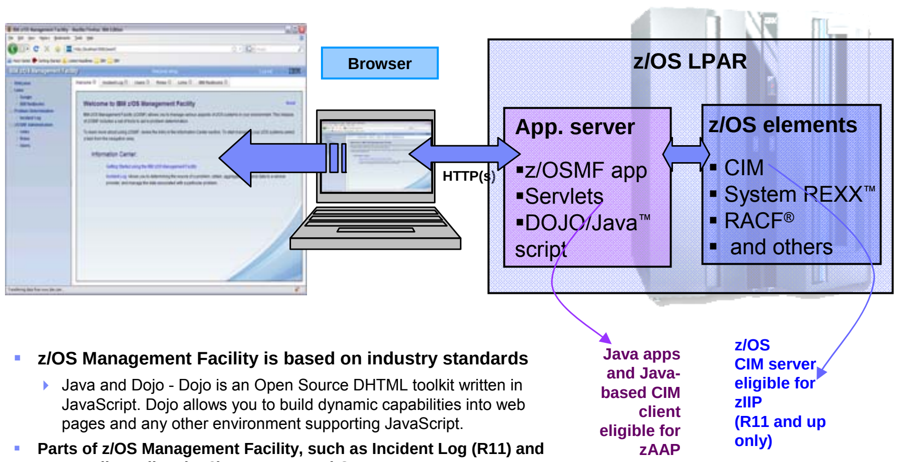
- **z/OS Management Facility is a Web 2.0 application on z/OS**
  - Manages z/OS from z/OS
  - Browser communicates with z/OS MF via secure connection, anywhere, anytime
  - z/OSMF V1R11 is supported on z/OS V1R10 w/maint, z/OSV1R11, and z/OS V1R12
  - z/OSMF V1R12 is supported on z/OS V1R12

IBM z/OS Management Facility
The Application Stack



- The z/OS Management Facility product consists of :
  - WebSphere Application Server OEM Edition
  - z/OSMF applications
- The z/OS Management Facility applications run on the z/OS system and are presented on a PC using a browser
- The z/OS Management Facility requires:
  - z/OS Communications Server
  - Security definitions (SAF)
  - System Logger
  - Other components are required for specific z/OSMF applications

IBM z/OS Management Facility
Industry standards

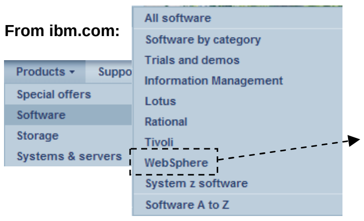


- z/OS Management Facility is based on industry standards
  - Java and Dojo - Dojo is an Open Source DHTML toolkit written in JavaScript. Dojo allows you to build dynamic capabilities into web pages and any other environment supporting JavaScript.
- Parts of z/OS Management Facility, such as Incident Log (R11) and WLM Policy Editor (R12) use JAVA and CIM

WebSphere is a brand; WebSphere Application Server a product

We'll start by clearing up a point of confusion about the term *WebSphere*.

From ibm.com:



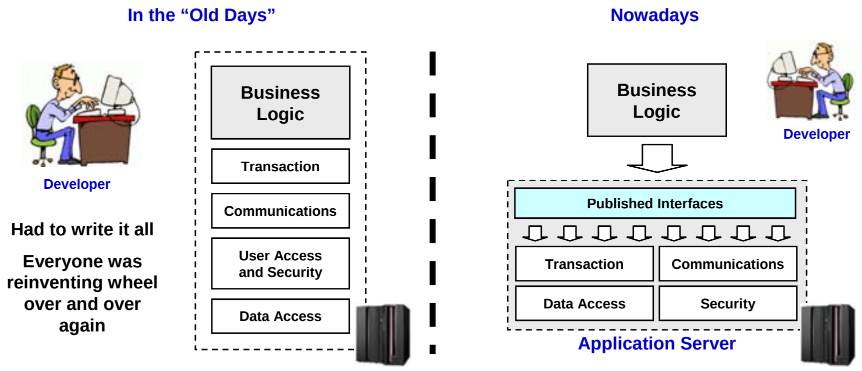
WebSphere Adapters
WebSphere Application Server — Our Focus
Branch Transformation Toolkit for WebSphere Studio
WebSphere Business Events
WebSphere Business Integration Server
:
:
WebSphere Virtual Enterprise
WebSphere Voice Response for AIX
WebSphere Voice Server

Probably close to 100 products carry the "WebSphere" brand name.

For many, the term *WebSphere* means *WebSphere Application Server*. Sometimes the acronym WAS is also used informally.

What an application server provides

WebSphere Application Server is an application server … but what is that?



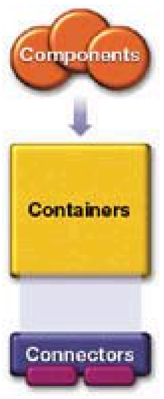
Purpose is to provide pre-packaged application support stuff so that developers can focus on the main business task. No more reinventing the wheel.

This is *not* new with WebSphere … IBM had an application server back in 1968!*

So what's the key difference between WebSphere and past application servers?

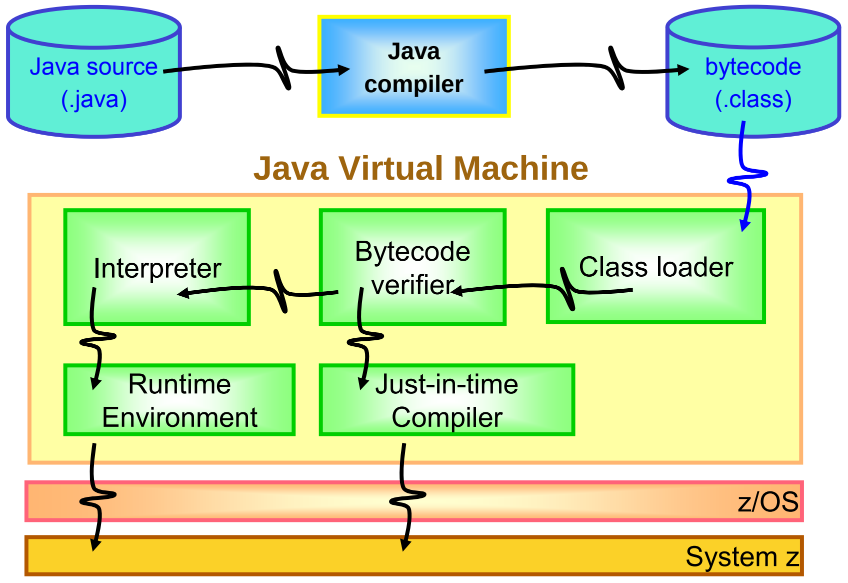
* CICS is an application server

The J2EE application model



- Components
  - The key focus of application developers; these are the EJBs, Servlets, JSPs, and clients.
  - Many component behaviors can be specified at deployment time, rather than in program code.
- Containers
  - These provide services to components transparently, including transaction support and resource pooling.
  - Containers and connectors conceal complexity and promote portability.
- Connectors
  - These sit under the J2EE platform, defining portable service APIs to plug into existing enterprise vendor offerings.
  - Connectors promote flexibility by enabling a variety of implementations of specific services.

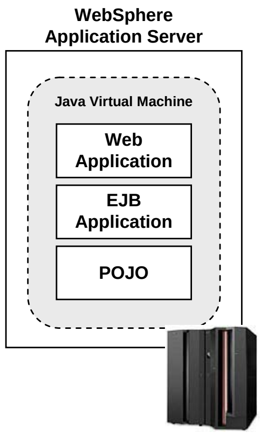
Compile and run Java on z/OS



Different kinds of Java programs

WebSphere Application Server can host (or "run" or "support") several different kinds of applications, all written in Java:

It is okay not to understand the details of these things. It is better at this point just to understand that different kind of programs exist and listen for these terms when others talk about the WebSphere environment.



Web application

An application that is accessed with a browser. This typically consists of static files (HTML, JPG/GIF), and Java programs that generate dynamic output:

- **Servlets**: Java program that contains logic to do things like perform calculations, access data, and format a reply
- **JSPs**: stands for Java Server Pages, it's a way to create a dynamic web page that can be populated with dynamic content

EJB application

Stands for "Enterprise Java Bean," it's a more sophisticated application that's intended for high-end applications. Two flavors:

- **Session Beans**: meant to hold the logic of the application
- **Entity Beans**: meant to represent data as an "object"

Many EJB applications are made up of just session beans -- easier.

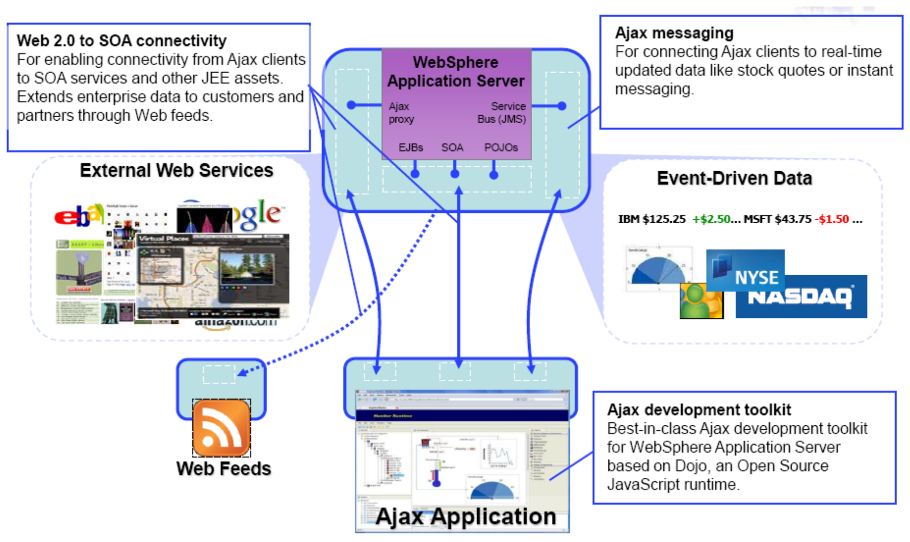
Java EE

Too simple a categorization, but okay for now

POJO

Stands for "Plain Old Java Object." It is the simplest form of a Java program and lately more people are returning to simplicity. (POJO commonly applies to the EJB 3.0 environment and Java Batch environment).

IBM Feature Pack for Web 2.0



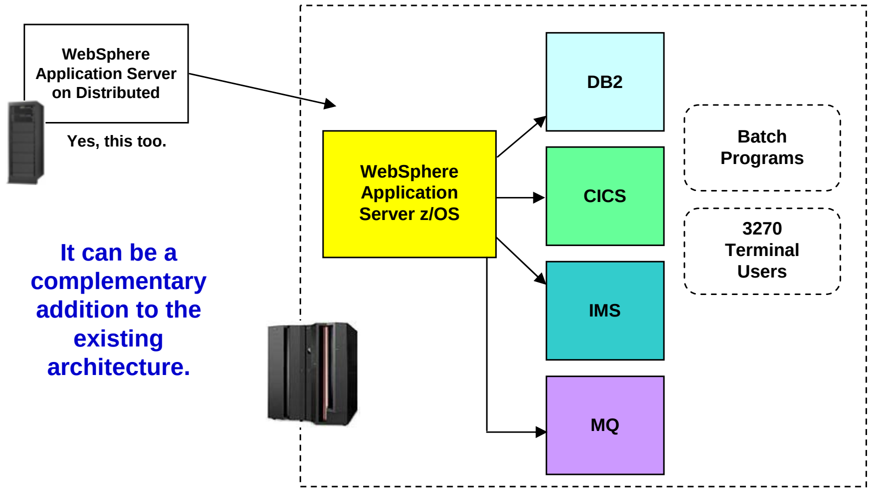
Web 2.0 to SOA connectivity
For enabling connectivity from Ajax clients to SOA services and other JEE assets. Extends enterprise data to customers and partners through Web feeds.

Ajax messaging
For connecting Ajax clients to real-time updated data like stock quotes or instant messaging.

Ajax development toolkit
Best-in-class Ajax development toolkit for WebSphere Application Server based on Dojo, an Open Source JavaScript runtime.

Use WebSphere in combination with other solutions!

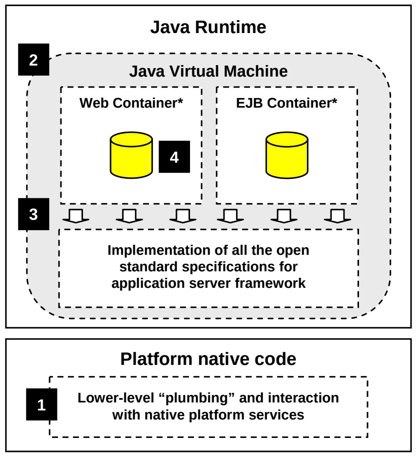
WebSphere Application Server works perfectly well in combination with other traditional systems, such as CICS, IMS, and DB2:


WebSphere Application Server on Distributed
Yes, this too.
WebSphere Application Server z/OS
DB2
CICS
IMS
MQ
Batch Programs
3270 Terminal Users

It can be a complementary addition to the existing architecture.

Schematic diagram of WebSphere Application Server

Here is a semi-conceptual view of what WebSphere Application Server is:

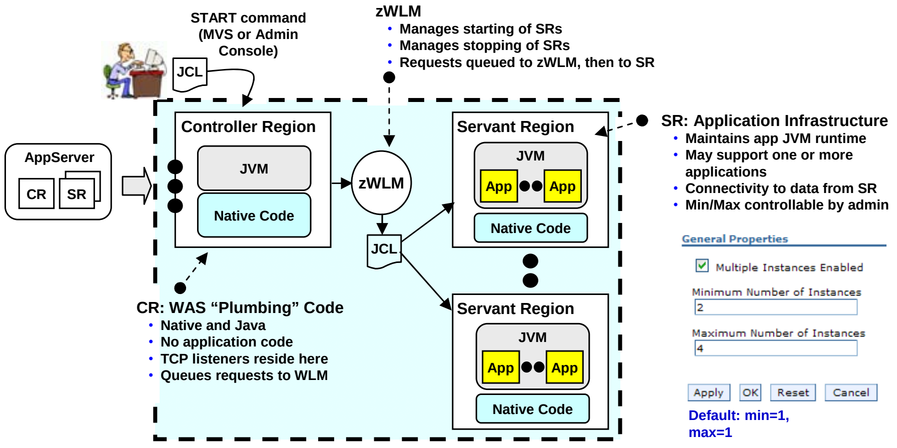

Java Runtime
2
Java Virtual Machine
Web Container*
EJB Container*
4
3
Implementation of all the open standard specifications for application server framework
Platform native code
1
Lower-level "plumbing" and interaction with native platform services

* "Containers" are just logical software constructs inside the JVM that provide services specific to the type of application that runs in them. "Web Container" is for web applications; "EJB Container" is for EJBs.

1. Server is started
   - On z/OS that is done with a START command (more later).
   - This native code is what establishes the lower-level "plumbing" and allows for the invocation of the Java environment.
2. Java Runtime established, including JVM
   - When the native base is ready, it establishes Java runtime environment and launches the JVM.
3. WebSphere Java components loaded into JVM
   - With the JVM launched, WebSphere Application Server can now load the Java components that make up the Java EE environment.
   - This is the "framework" we mentioned earlier.
   - This is why WebSphere Application Server is more than just a JVM.
4. Your applications are loaded and started
   - If they are deployed in the server and configured to start automatically, WebSphere will do that for you.

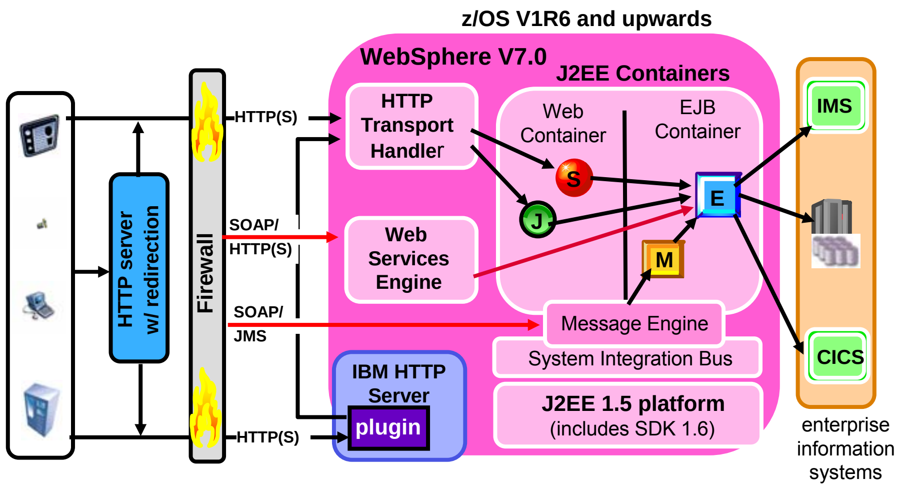
A peek inside the application server architecture

We see that inside our little curved-box picture of the Application Server resides two or more address spaces as well as integration with zWLM:


START command (MVS or Admin Console)
JCL
zWLM
• Manages starting of SRs
• Manages stopping of SRs
• Requests queued to zWLM, then to SR
AppServer
CR
SR
Controller Region
JVM
Native Code
zWLM
JCL
Servant Region
JVM
App
App
Native Code
Servant Region
JVM
App
App
Native Code
CR: WAS "Plumbing" Code
• Native and Java
• No application code
• TCP listeners reside here
• Queues requests to WLM
SR: Application Infrastructure
• Maintains app JVM runtime
• May support one or more applications
• Connectivity to data from SR
• Min/Max controllable by admin
General Properties
Multiple Instances Enabled
Minimum Number of Instances
2
Maximum Number of Instances
4
Apply
OK
Reset
Cancel
Default: min=1, max=1

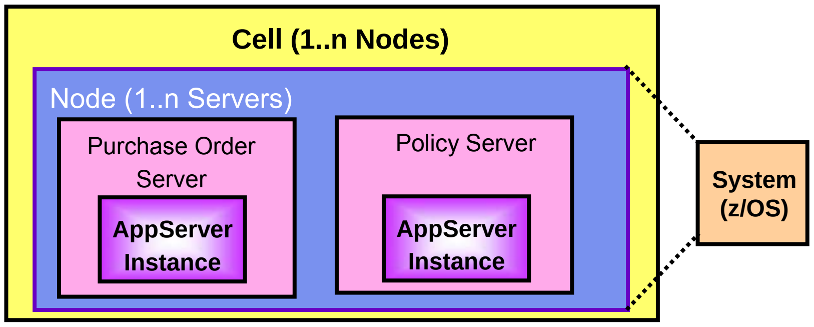
This is a built-in "vertical scaling" mechanism. It also allows for redundancy of application JVM to prevent single point of failure.

WebSphere V7 for z/OS infrastructure



- JMS messaging and Web Services integration are implemented using the Service Integration Technologies.
- System Integration Bus provides common connection bus for J2EE applications.
- Message Engine provides platform for JMS services.

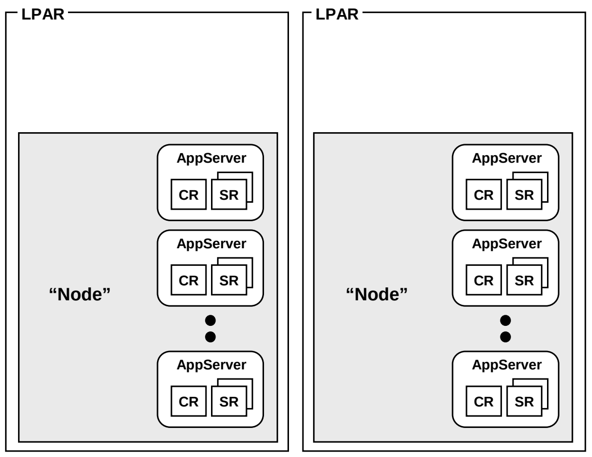
WebSphere V7: Basic architecture



- Application server
  - Logical component comprising a group of J2EE apps
- Application server instance
  - A server process for executing J2EE applications from specific AppServer
  - Contains a Web container, EJB Container, and services tasks
- Node
  - Grouping of servers for configuration and operational management
  - Cannot span the scope of a machine/LPAR
- Cell
  - Network of multiple nodes
  - Single point of administration

Multiple application servers and the concept of a node

There are many reasons* for creating multiple application servers. A "node" is simply the logical collection of applications servers on an LPAR:



Key points:

- **Nodes are a logical thing; They are not a started task.**
- **They logically organize application servers on an LPAR.**
- **No architectural limit to the number of application servers in a node; limited only by system resources.**
- **Rule: Node must stay on an LPAR; it cannot span LPARs in a Sysplex.**

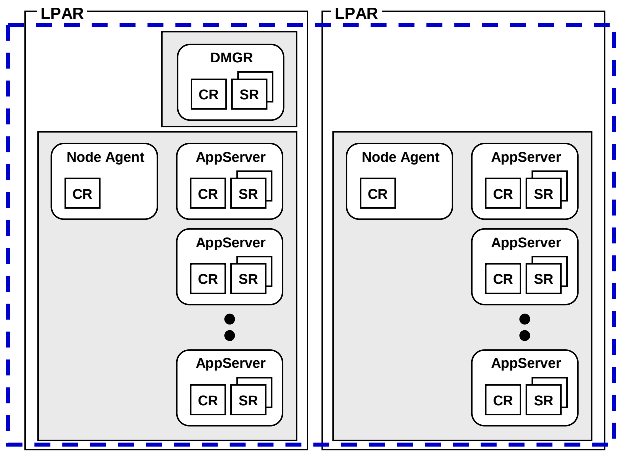
What is the point?

(We will see in a moment.)

*** Requirement for separation of application.
Applications have different custom JVM settings.
Different performance requirements**

Now we can introduce the concept of the cell

The cell is really nothing more than the extent of administrative control a DMGR has. In this example, it controls two nodes on two LPARs that's the cell.

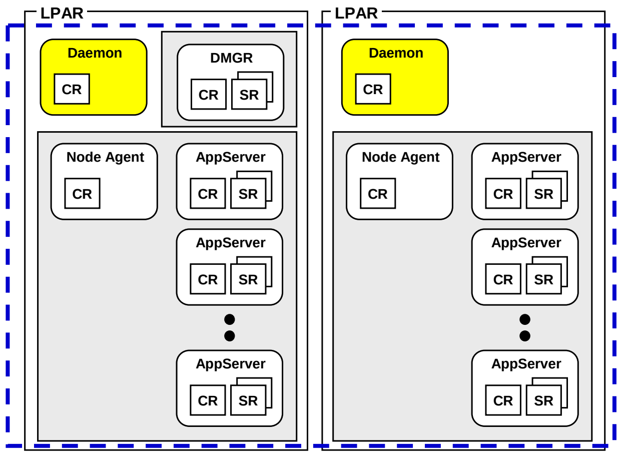


Key Points:

- The Cell is a logical thing; it is not a started task or address space.
- The Cell marks the boundary for administrative isolation. You can limit who has access to modifications to the Cell. This is how QA, Test, and Production are best kept separate.

Wrap-up: Daemon servers

Daemons host two functions: access to modules held in storage, and the "Location Name Service" for remote client IIOP requests



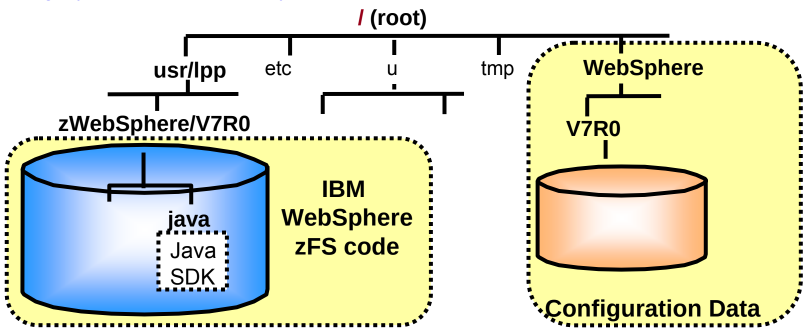
Key Points:

- Perhaps the most confusing server element in the WebSphere configuration
- They are not really part of a node; they are more cell/LPAR based.
- Rule of thumb:

One daemon per cell per z/OS image

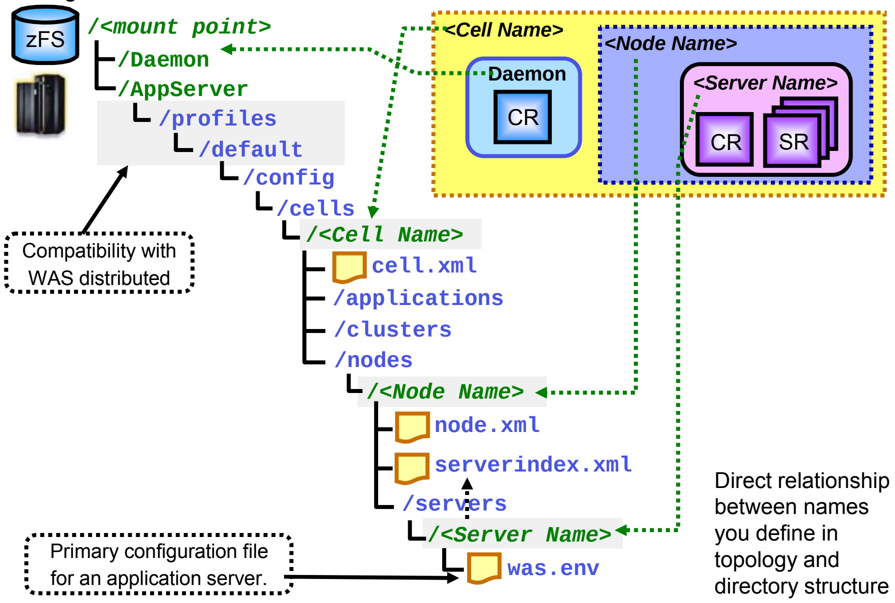
WAS and USS zFS

- WebSphere V7 SMP/E installation target:
  - IBM WebSphere distributed code in `/usr/lpp/zWebSphere/V7R0`
  - Integrated Java SDK in `/usr/lpp/zWebSphere/V7R0/java`
- Server and application configuration and run-time data kept on a separate file system
  - Default mount point for configuration HFS file - `/WebSphere/V7R0...`
  - HFS file sharing not required in a Sysplex; administration tool ensures data integrity over concurrent system updates



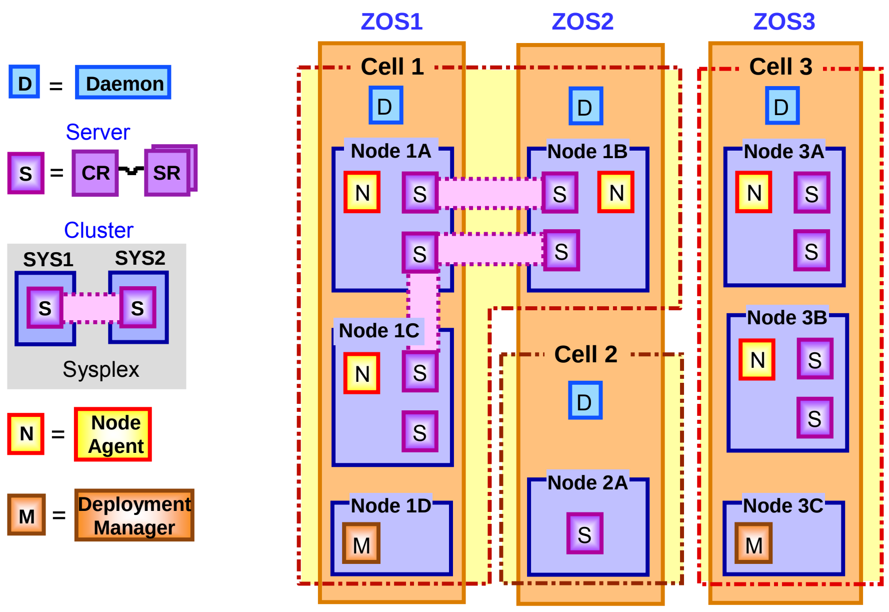
zFS under the base application server node

Configuration zFS

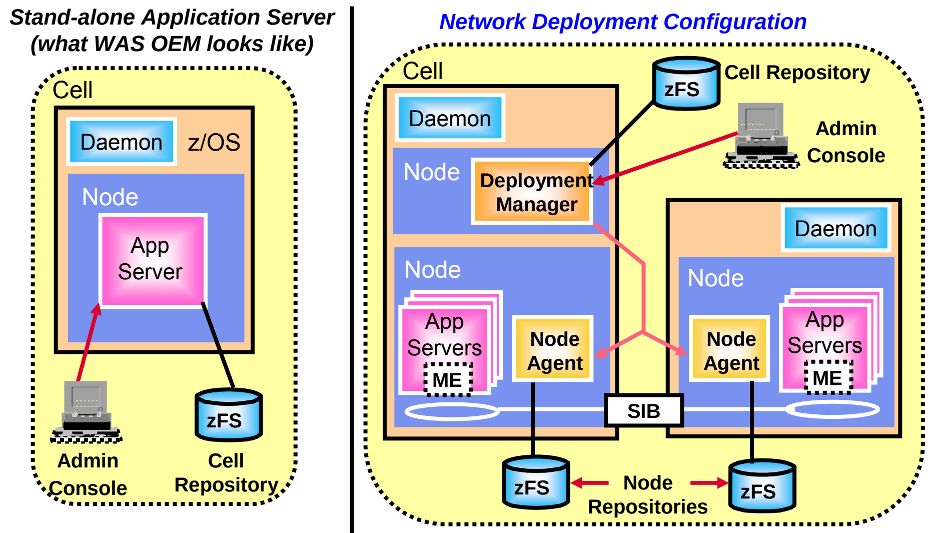


Direct relationship between names you define in topology and directory structure

WebSphere for z/OS v7 topology: Elements view



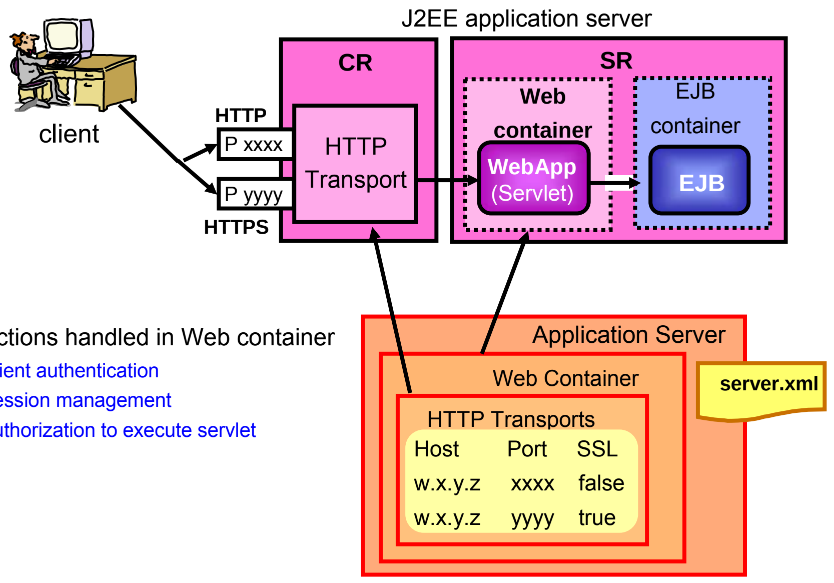
WebSphere for z/OS configuration types



- Server is called an unmanaged application server.
- Servers are called managed application servers.

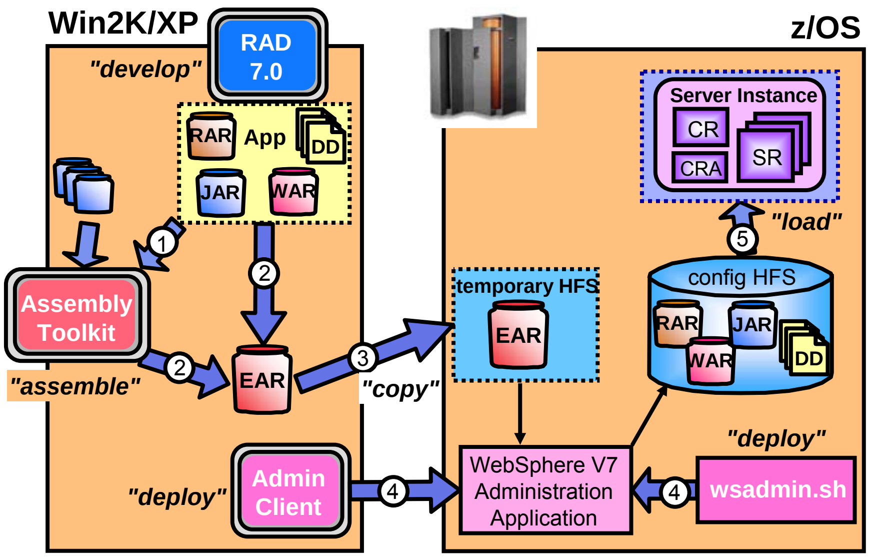
ME = messaging engine, SIB = System Integration Bus

Overview of HTTP transport



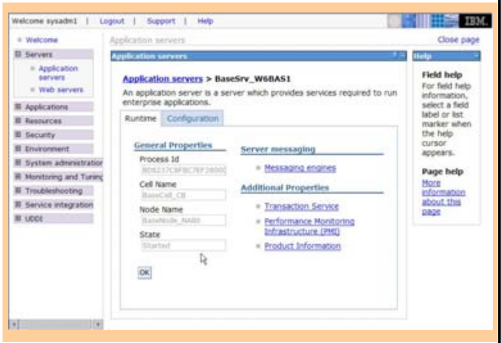
- Functions handled in Web container
  - Client authentication
  - Session management
  - Authorization to execute servlet

WebSphere V7 application deployment



WebSphere V7 administration tool (1 of 2)

- Family wide SM function and topology
  - Managed resources = JMX MBeans
  - Set of JMX Connectors provides choice of remote access protocol
- Administrative domains of cell, node, server
- Multiple administrator support
- Common WebSphere Admin Model/Process/Console
  - Thin client UI – Web browser + applet
  - Install applications directly from WSAD
- Consolidated configuration files
  - JVM configuration + Server Environment Variables + Java Properties + Web Container config
- Common scripting function across all WebSphere platforms
  - Pluggable architecture will support multiple scripting languages, Jacl, and Jython.
- JNDI/CosNaming federation built into SM Admin function


Welcome sysadm1 | Logout | Support | Help

Welcome
Servers
Application servers
Web servers
Applications
Resources
Security
Environment
System administration
Monitoring and Tuning
Troubleshooting
Service integration
UDDI

Application servers
Close page

Application servers

Application servers > BaseSrv_W6BAS1

An application server is a server which provides services required to run enterprise applications.

Runtime | Configuration

General Properties

Process Id
[illegible]

Cell Name
BaseCell_C8

Node Name
BaseNode_NAB0

State
Started

OK

Server messaging
- Messaging engines

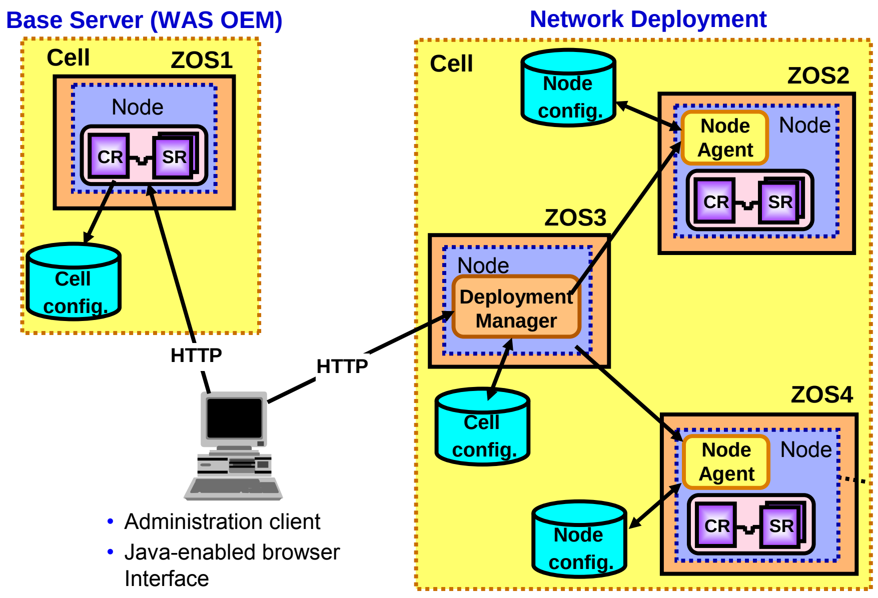
Additional Properties
- Transaction Service
- Performance Monitoring Infrastructure (PMI)
- Product Information

Help

Field help
For field help information, select a field label or list marker when the help cursor appears.

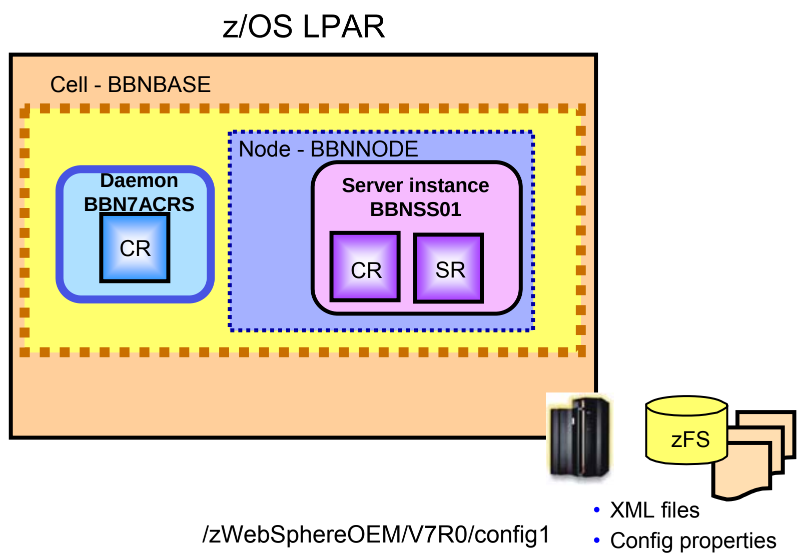
Page help
More information about this page

WebSphere V7 administration tool (2 of 2)



- Administration client
- Java-enabled browser Interface

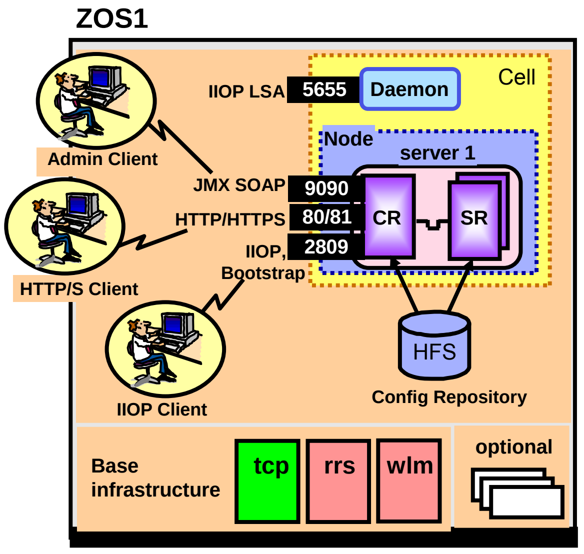
WAS OEM stand-alone server node



- .

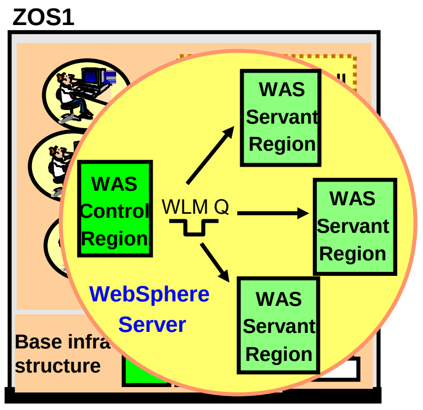
- XML files
- Config properties
- Applications

WebSphere for z/OS V7 system structure



- Infrastructure
  - TCP/IP
  - WLM (goal mode)
  - RRS (LOGR)
  - USS
- WAS for z/OS
  - WAS servers
  - Application servers
- Users
  - Admin Client
  - HTTP/S Client
  - IIOP Client
- Optional products
  - IMS, CICS
  - MQSeries
  - DB2
  - LDAP

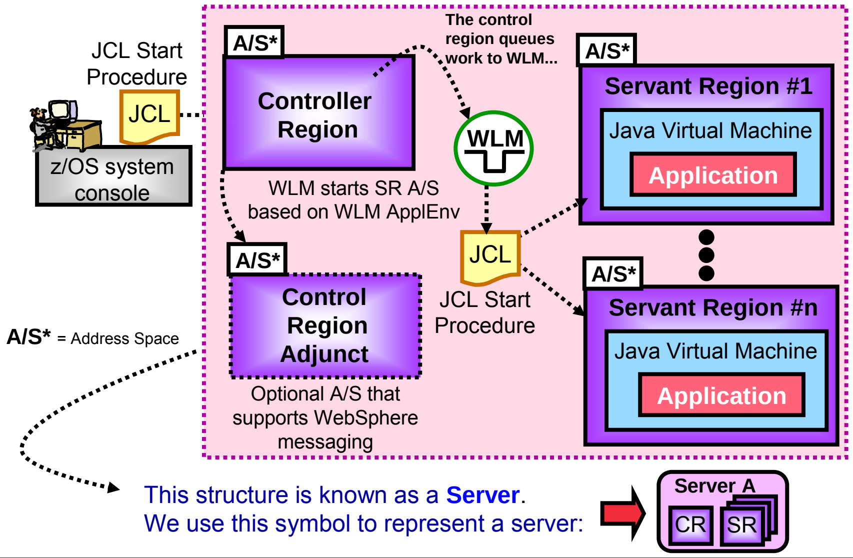
WLM Support



- Control region
  - Receives request
  - Classifies request
  - Queues request
- WLM
  - Manages queued requests
  - Starts and stops server regions as necessary
  - Monitors system resources
  - Manages to installation goals
- Server region
  - Selects work for given service class
  - Application code executes here

Controllers, CRAs, and servants

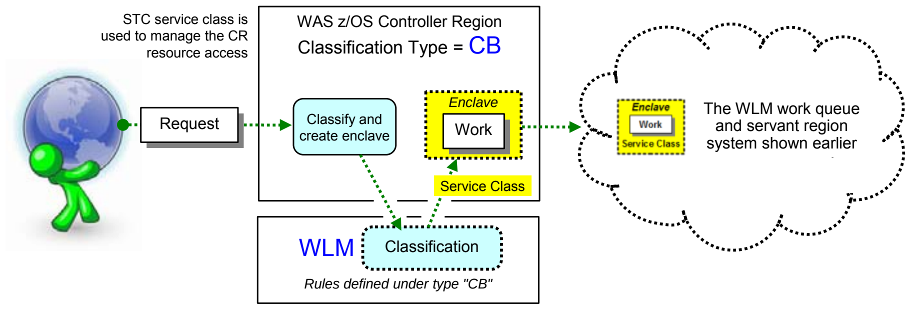
The address space in which the JVM resides is called a **servant**. Multiple servants may be started by WLM, based on work queued by a controller region:



This structure is known as a **Server**.
We use this symbol to represent a server:

The WLM "Enclave"

An "enclave" is a way to identify and manage individual pieces of work *within* the many parts of a running z/OS system

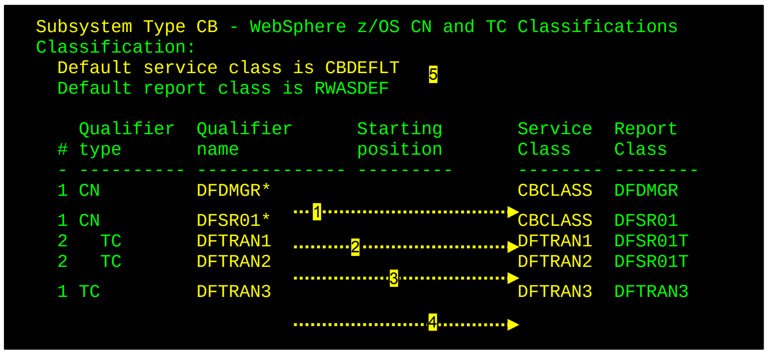

STC service class is used to manage the CR resource access
WAS z/OS Controller Region
Classification Type = CB
Request
Classify and create enclave
Enclave
Work
Service Class
WLM
Classification
Rules defined under type "CB"
Enclave
Work
Service Class
The WLM work queue and servant region system shown earlier

Key points from this chart

- An "enclave" is simply a way for WLM to understand priorities at a work unit level
- WAS does this automatically ... if you do no other configuration it'll still do this with default values

Assigning a Service Class to the Enclave

This is for the work request ... earlier we saw how the CR was classified using the STC type. Now we look at the CB type ...



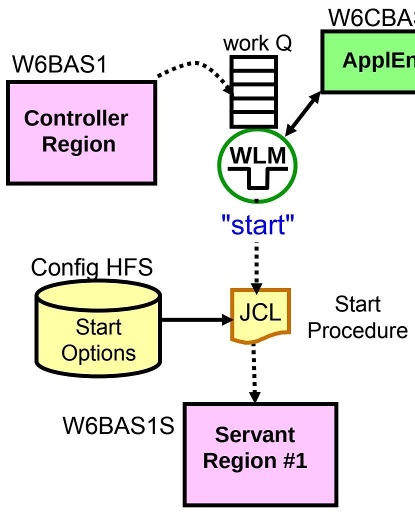
```
Subsystem Type CB - WebSphere z/OS CN and TC Classifications
Classification:
  Default service class is CBDEFLT    5
  Default report class is RWASDEF

    Qualifier  Qualifier      Starting       Service  Report
  # type       name           position       Class    Class
  - ---------- -------------- ---------      -------- --------
  1 CN         DFDMGR*                       CBCLASS  DFDMGR

  1 CN         DFSR01*  ···1·················▶CBCLASS  DFSR01
  2   TC       DFTRAN1  ··········2··········▶DFTRAN1  DFSR01T
  2   TC       DFTRAN2                        DFTRAN2  DFSR01T
                        ··············3······▶
  1 TC         DFTRAN3                        DFTRAN3  DFTRAN3

                        ··················4··▶
```

Enclaves created in WAS CR are classified by rules in CB subsystem type:

1. CN of `DFDMGR*` matches the Deployment Manager. Work there goes to `CBCLASS`.
2. Work in `DFSR01*` cluster *without* a transaction classification gets `CBCLASS` as well.
3. Work in `DFSR01*` cluster *with* TC of `DFTRAN1` or `DFTRAN2` get service classes as shown
4. Work that matches the TC of `DFTRAN3` *regardless of WAS CN* gets service class `DFTRAN3`
5. Anything that doesn't match any specific rules gets the default service class of `CBDEFLT`

WLM application environments

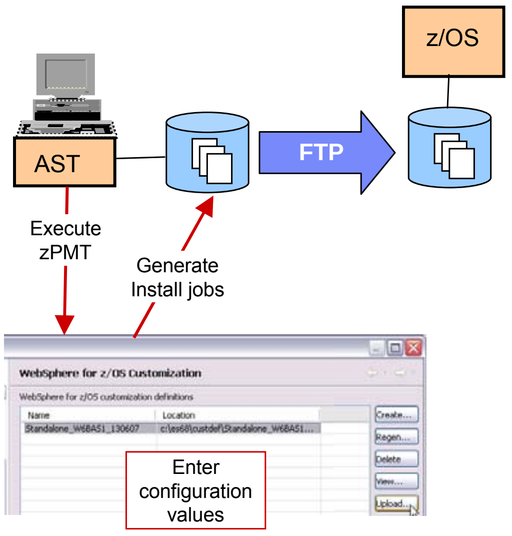
- Application environments are the WLM mechanism for managing each server (application and infrastructure)



- WLM starts servant regions based on workload received.
- ApplEnv provides WLM with the parameters needed to start the servant region.
  - Subsystem type
  - Node name
  - Servant region PROC name specified to WLM
  - Pointer to server configuration variables in config HFS

z/OS Profile Management Tool (not seen with WAS OEM)

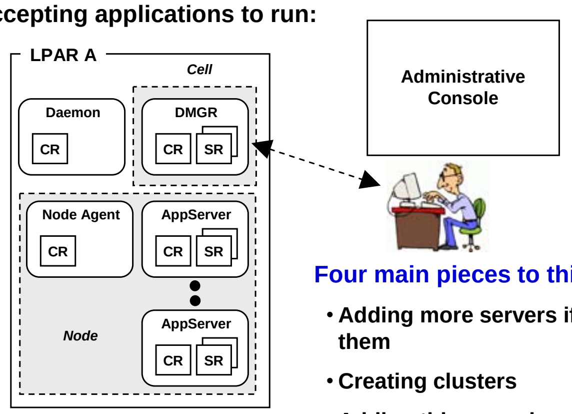
- New WebSphere for z/OS customization tool
- Shipped as part of the Application Server Toolkit
- GUI-based Windows application
- User defines the same configuration values as for the old ISPF tool
- Installation jobstreams are generated on workstation
- Jobstreams transferred to z/OS with FTP
- Strategic tool


z/OS
AST
FTP
Execute zPMT
Generate Install jobs
WebSphere for z/OS Customization
WebSphere for z/OS customization definitions
Create...
Regen...
Delete
View...
Upload...
Enter configuration values

Name	Location
Standalone_W6BAS1_130607	c:\es68\custdef\Standalone_W6BAS1...

What you have after you have built your cell

After you have done all that, you have a configuration that is capable of accepting applications to run:



But your cell will no doubt require more post-creation customization

Four main pieces to this:

- **Adding more servers if you see the need for them**
- **Creating clusters**
- **Adding things such as JDBC, JCA, and MQ**
- **Deploying applications and starting them**

This post-creation customization is common across all platforms; it is not just a z/OS thing.

In fact, it is common across all middleware, such as DB2, CICS, MQ; all require some customization.

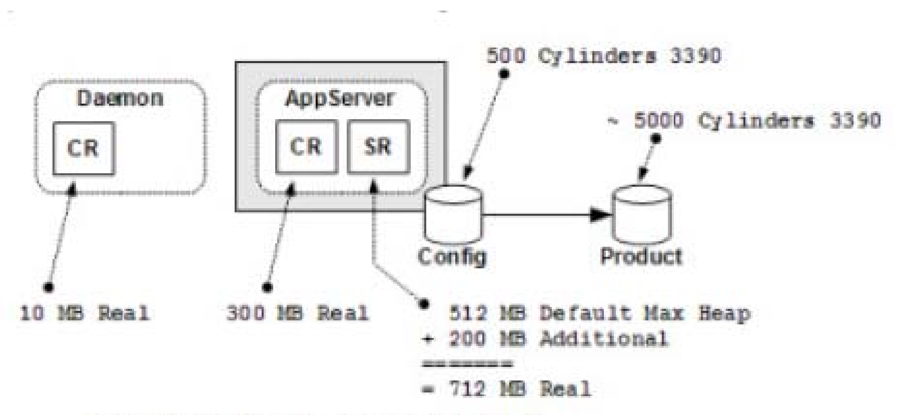
WAS OEM Response File (1of 2)

```
cellName=bbnbase
hostName=@HOSTNAME
nodeName=bbnnode
profileName=default
serverName=server1
zAdjunctProcName=BBN7CRA
zAdminAsynchProcName=BBN7ADM
zAdminAsynchTaskUid=2504
zAdminAsynchTaskUserid=WSADMSH
zAdminConsolePort=32205
zAdminConsoleSecurePort=32206
zAdminLocalPort=32209
zAdminUid=2403
zAdminUnauthenticatedUid=2402
zAdminUnauthenticatedUserid=WSGUEST
zAdminUserid=WOEMADM
```

WAS OEM Response File (2 of 2)

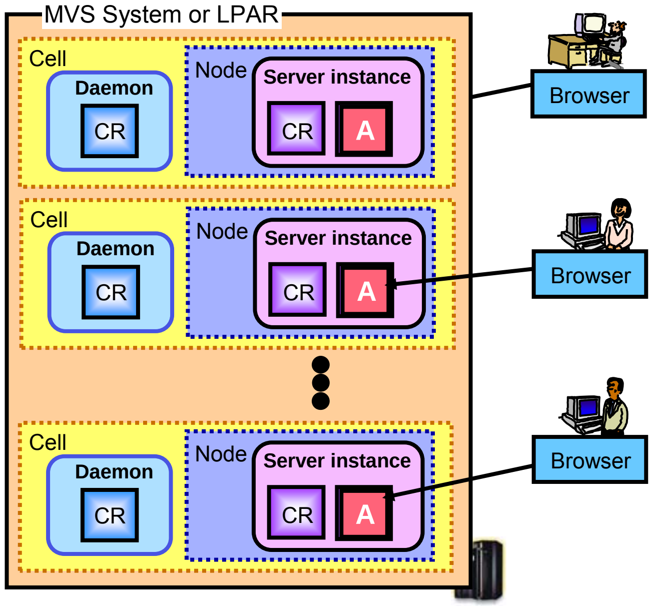
```
zCellShortName=BBNBASE
zClusterTransitionName=BBNC001
zConfigHfsName=BBN.V7R0.CONFIG1.ZFS
zConfigHfsVolume=BBNVOL
zConfigMountPoint=/zWebSphereOEM/V7R0/config1
zConfigurationGroup=WSCFG1
zConfigurationGroupGID=2500
zControlProcName=BBN7ACR
zControlUid=2431
zControlUserid=WSCRU1
zDaemonHomePath=generated
zDaemonIPName=generated
zDaemonJobName=BBN7ACRS
zDaemonPort=32200
```

WebSphere App Server for z/OS Real Storage Requirements



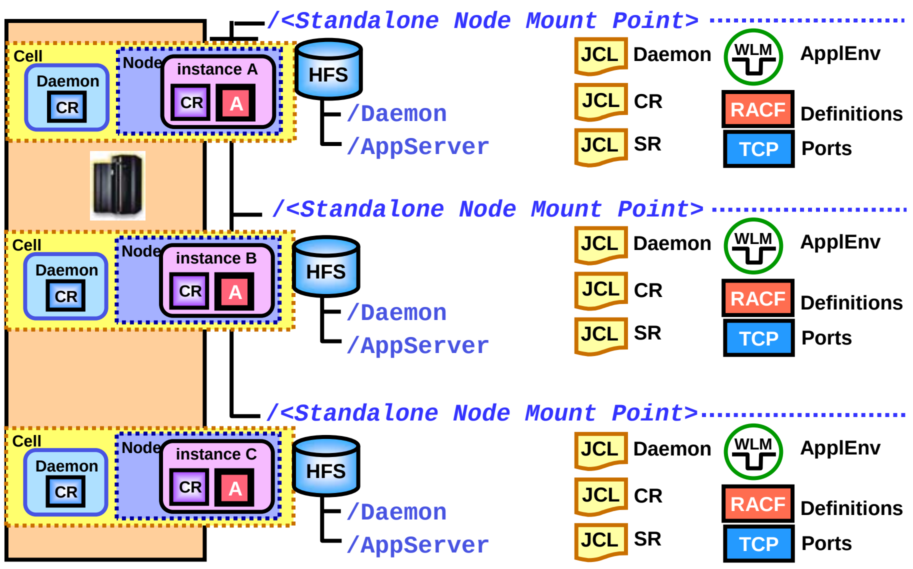
Estimated Storage	1,022 MB of real
Estimated Disk	5,500 cylinders
zAAP	1 additional to support WAS

Multiple stand-alone server nodes



- Multiple stand-alone nodes in one system are acceptable with WebSphere for z/OS Version 7.
- It involves running the dialog for each stand-alone node.
- Each server would have its own:
  - Server root (and HFS)
  - Cell
  - Node
  - TCP/IP ports
  - Admin interface
- Implies great separation and isolation ...

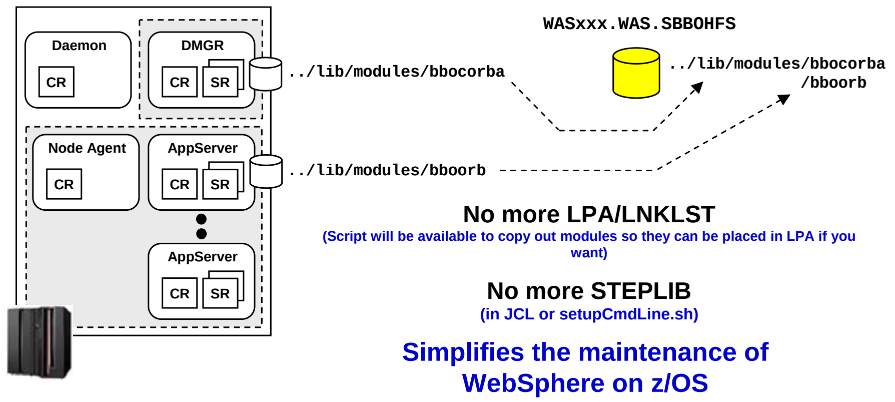
Separate resources for each node



If you are looking for isolated environments, this is good news.

No more load module libraries in V7

The load modules will now be included in the HFS under `/lib/modules`.



No more LPA/LNKLST

(Script will be available to copy out modules so they can be placed in LPA if you want)

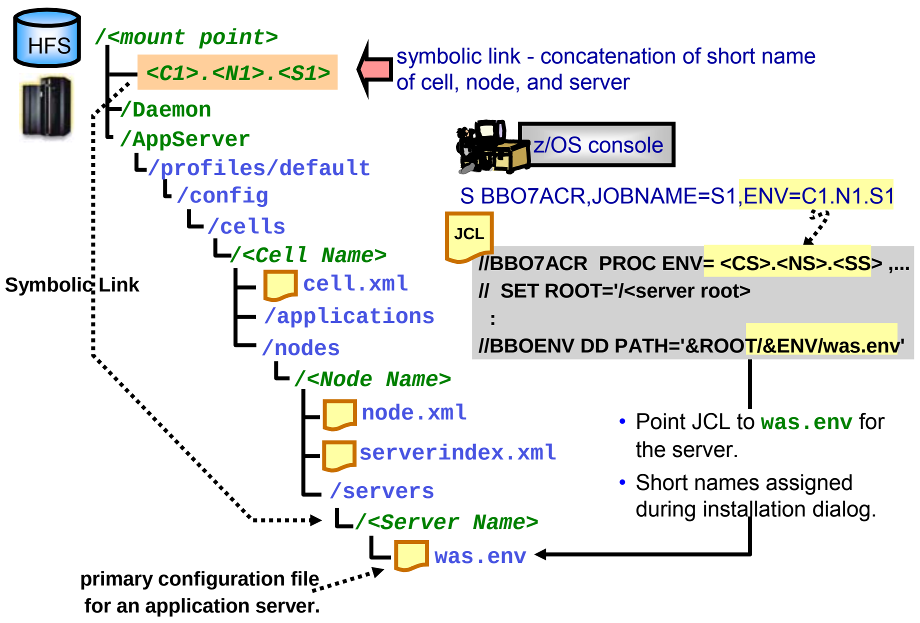
No more STEPLIB

(in JCL or setupCmdLine.sh)

Simplifies the maintenance of WebSphere on z/OS

Eliminates possibility of accidental mismatch between PDSE and HFS

Starting the stand-alone server

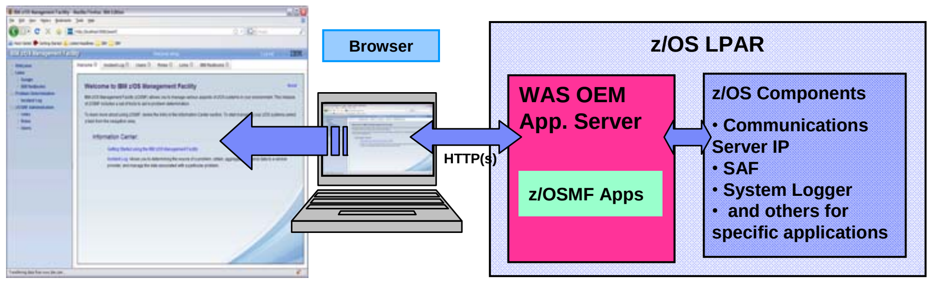


S BBO7ACR,JOBNAME=S1,ENV=C1.N1.S1

```
//BBO7ACR  PROC ENV= <CS>.<NS>.<SS> ,...
//  SET ROOT='/<server root>
  :
//BBOENV DD PATH='&ROOT/&ENV/was.env'
```

- Point JCL to `was.env` for the server.
- Short names assigned during installation dialog.

Starting z/OSMF



- START *appserver_proc_name*,JOBNAME=*server_short_name*,
- ENV=*cell_short_name.node_short_name.server_short_name*

- For example:

- START BBN7ACR,JOBNAME=BBNS001,ENV=BBNBASE.BBNNODE.BBNS001

Open A Web Browser to z/OSMF

- The URL for the Welcome task has the following format:
- https://hostname:port/zosmf/

- where:

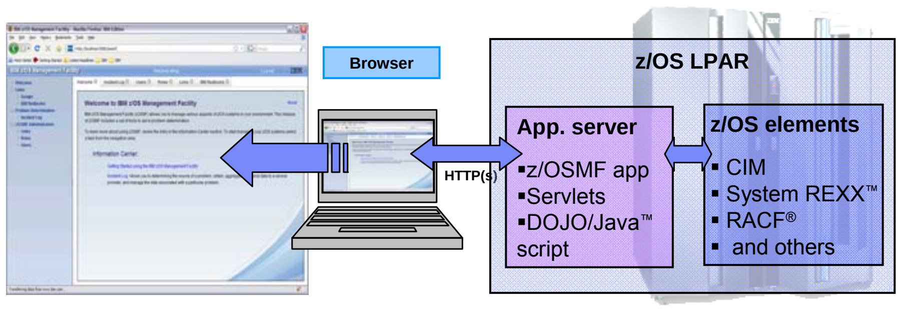
- v *hostname* is the hostname or IP address of the system in which IBM WebSphere Application Server OEM Edition for z/OS is installed

- v *port* is the secure application port for the IBM WebSphere Application Server OEM Edition for z/OS configuration. By default, the port is 32208.

WAS OEM Hostname / Port

- To find the hostname and port number, check the IBM WebSphere Application Server OEM Edition for z/OS response file, which is located by default in the directory:
- /etc/zWebSphereOEM/V7R0/conf/CONFIG1/CONFIG1.responseFile
- In the response file, see the following fields:
  - hostName
  - zHttpTransportSslPort

z/OSMF and WAS OEM


Browser
HTTP(s)
z/OS LPAR
App. server
▪z/OSMF app
▪Servlets
▪DOJO/Java™ script
z/OS elements
▪ CIM
▪ System REXX™
▪ RACF®
▪ and others