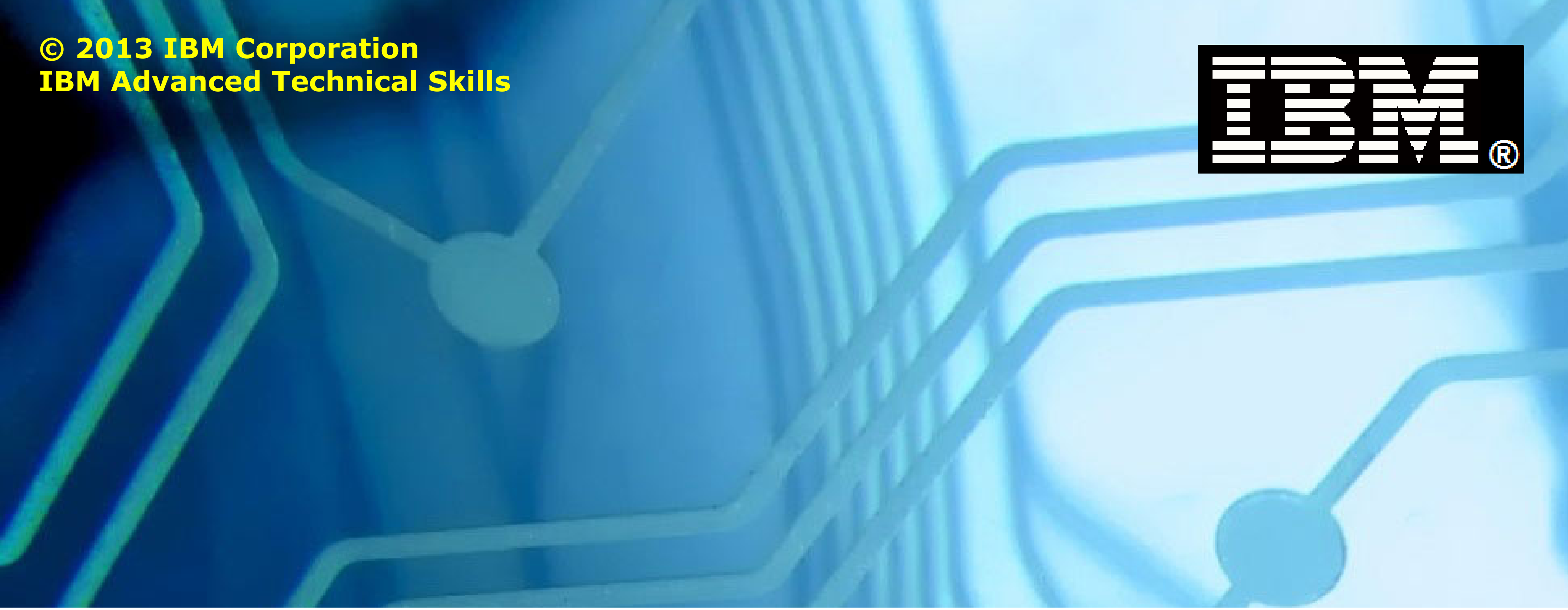


IBM Advanced Technical Skills

WBSR85

WebSphere Application Server z/OS V8.5

Functions and Capabilities

This page intentionally left blank

Agenda

- Introduction and Overview
- Administrative Model

  Hands-On Using the Admin Console, the WSADMIN interface and HPEL
- Understanding the Server Models
  - Multi-JVM model

    Hands-On Configuring, using dynamic MODIFY, and using WLM to classify work into separate servant regions
  - Granular RAS

    Hands-On Extending use of classification XML to control behavior to request level
  - Liberty Profile

    Hands-On New lightweight dynamic server runtime model
- Access Data
  - JDBC and DB2

    Hands-On Type 2/4, the new alternate JNDI failover function, functions unique to WAS z/OS
  - CICS

    Hands-On CTG EXCI and the Gateway Daemon
  - JMS and MQ

    Hands-On MQ as JMS provider using bindings and client mode
- Installation Manager (IM)
- WebSphere Optimized Local Adapers (WOLA)

  Hands-On Inbound batch-to-WAS; outbound WAS-to-CICS; HA functions

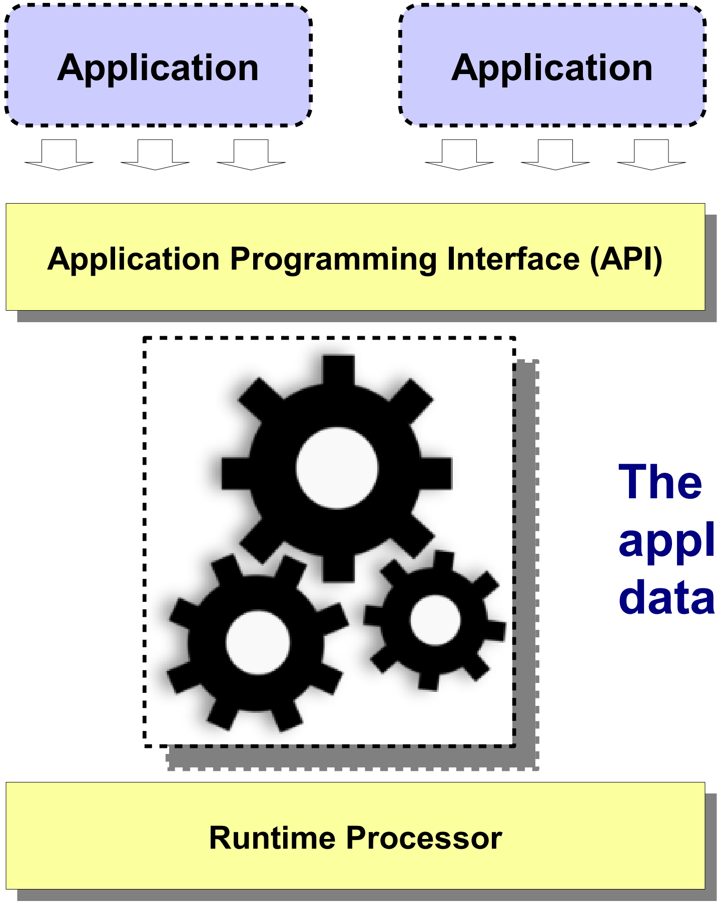
Concepts …

Essential Concepts

We start by getting a few key concepts on the table …

WAS is an "Application Server"

An "application server" provides functions and services to applications so the applications do not themselves do not have to re-invent those functions:



Application developer focuses on the specific business logic, and does *not* worry about writing "plumbing" code

Access to "plumbing code" is through *interfaces*, which are typically published and available to all application developers

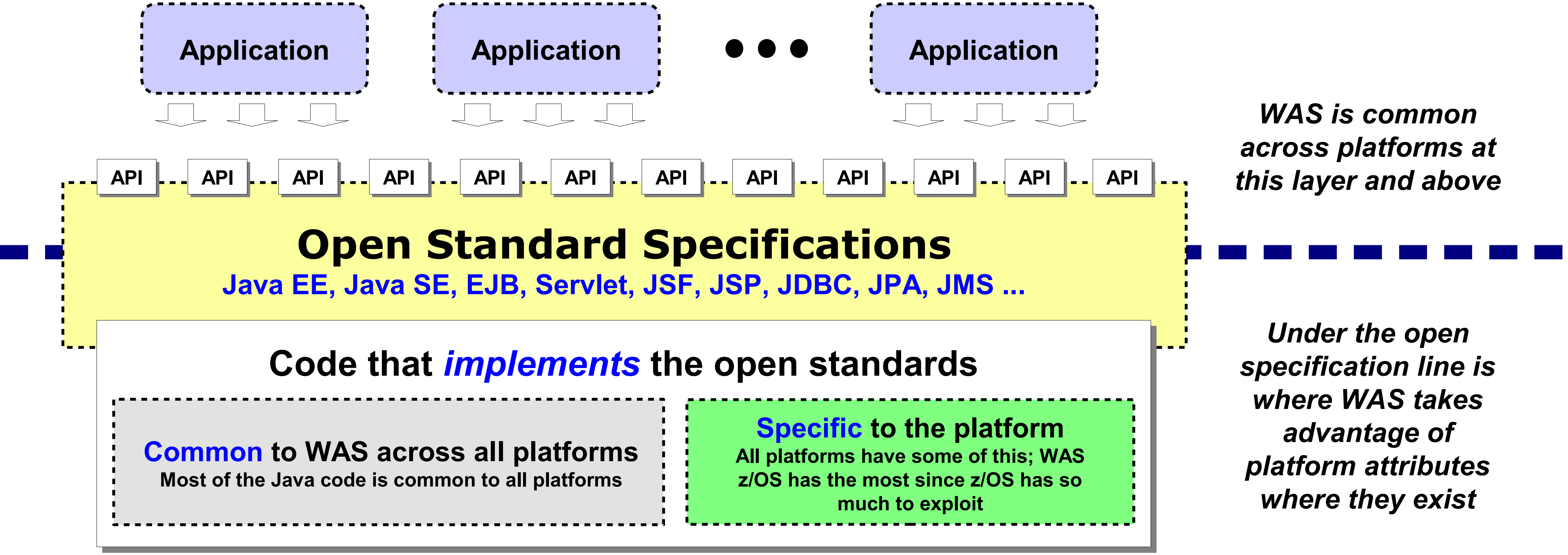
The "Application Server" provides services to the application, such as: communication, security, data access, transaction, etc.

The processor on which the appserver runs may be any of a wide range of possible platforms

WebSphere Application Server and open standard APIs …

"WAS is WAS" -- at Open Specification Layer

This is an important starting concept -- it's what makes application development a platform-neutral consideration:



Application

Application

• • •

Application

API API API API API API API API API API API API

Open Standard Specifications

Java EE, Java SE, EJB, Servlet, JSF, JSP, JDBC, JPA, JMS ...

Code that *implements* the open standards

Common to WAS across all platforms

Most of the Java code is common to all platforms

Specific to the platform

All platforms have some of this; WAS z/OS has the most since z/OS has so much to exploit

WAS is common across platforms at this layer and above

Under the open specification line is where WAS takes advantage of platform attributes where they exist

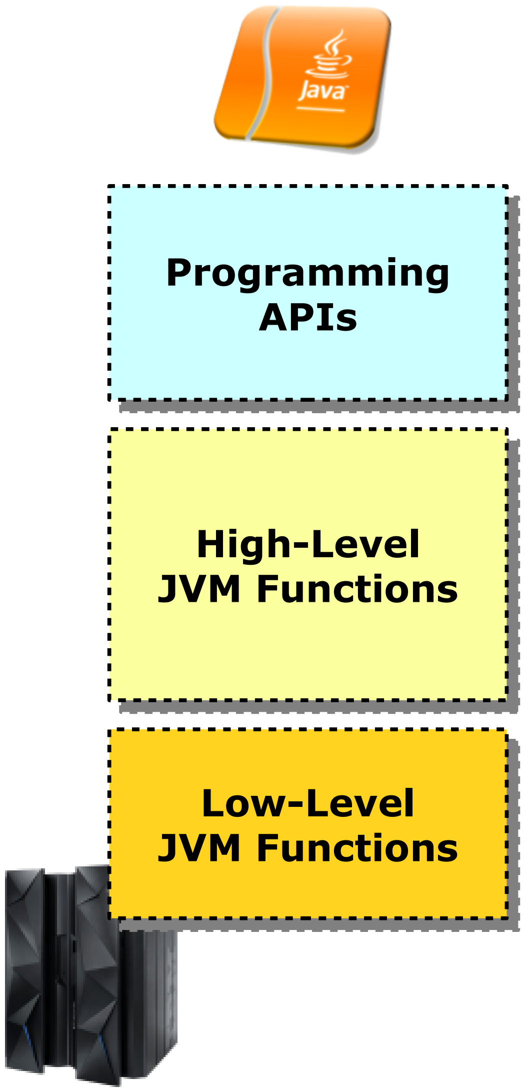
Much of this workshop will focus on what's available to be exploited below the line and how that can be of value

InfoCenter `rovr_specs`

Java at heart …

IBM Java inside WAS z/OS

It's important to understand that while the Java APIs are industry standards, the *implementation* below the APIs becomes increasingly platform-aware:



Programming APIs

High-Level JVM Functions

Low-Level JVM Functions

SDK conforms to the accepted standards

- IBM SDK provides all the required APIs according to the specification at the level being discussed
- IBM z/OS SDK provides *additional* APIs to take advantage of z/OS platform specific functions (such as SAF security)

JVM Functions common across IBM SDKs

- The JVM is entirely IBM's ... first delivered in 2005
- Many features: generational GC, shared classes
- High-level JVM functions common across IBM Java

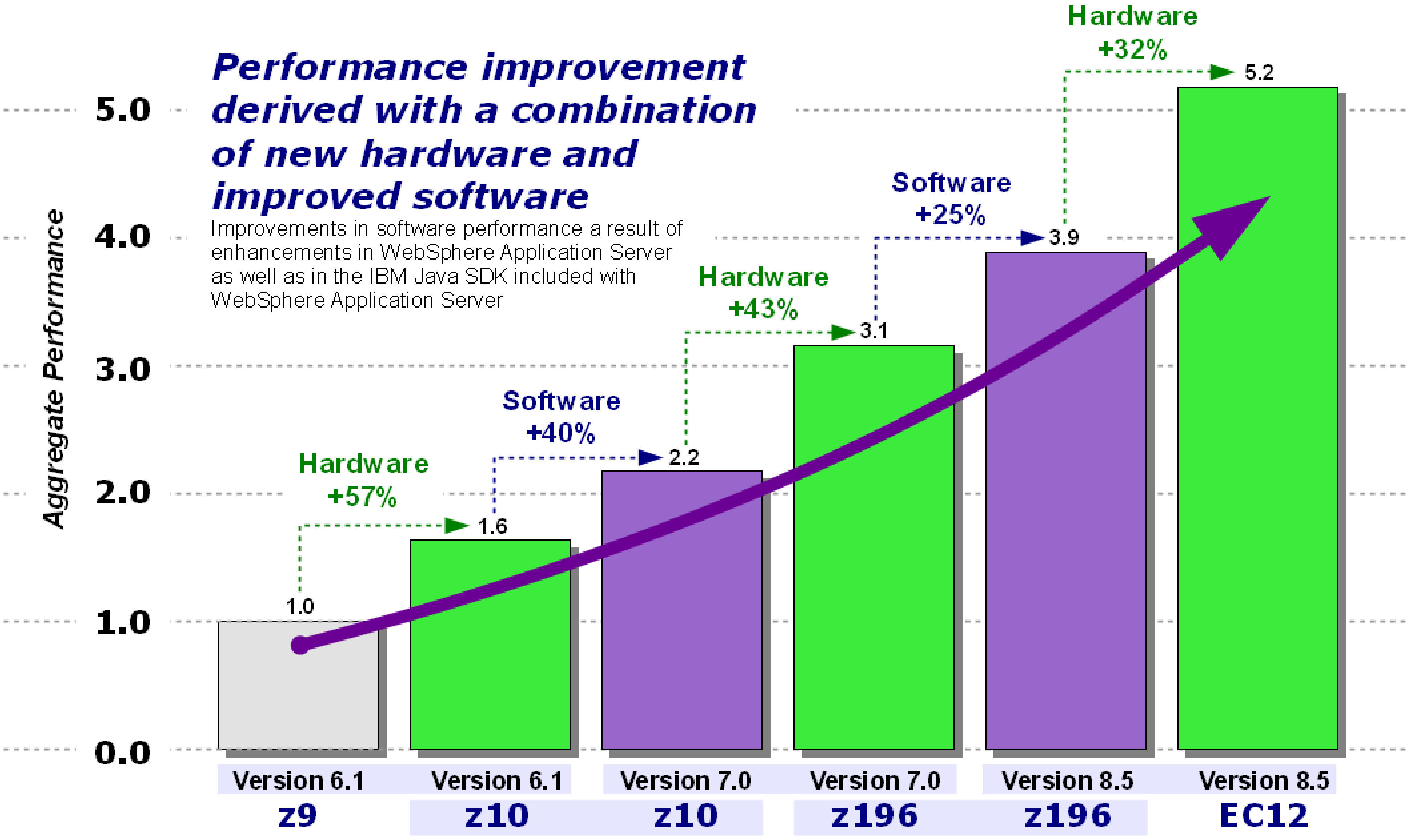
System z and z/OS functions

- Takes specific advantage of platform, including exploitation of new CISC instructions available with new System z: z10, z196, EC12
- Big Decimal, Large Page, Out of Order execution, transactional execution, flash paging ... equals *performance*
- Work with z/OS dispatcher to offload to specialty engines

Performance ...

Performance Over Time

It's a story of improvements in hardware and software:



Performance improvement derived with a combination of new hardware and improved software

Improvements in software performance a result of enhancements in WebSphere Application Server as well as in the IBM Java SDK included with WebSphere Application Server

Controlled test in specific environment. Results vary. This is not a guarantee of performance.

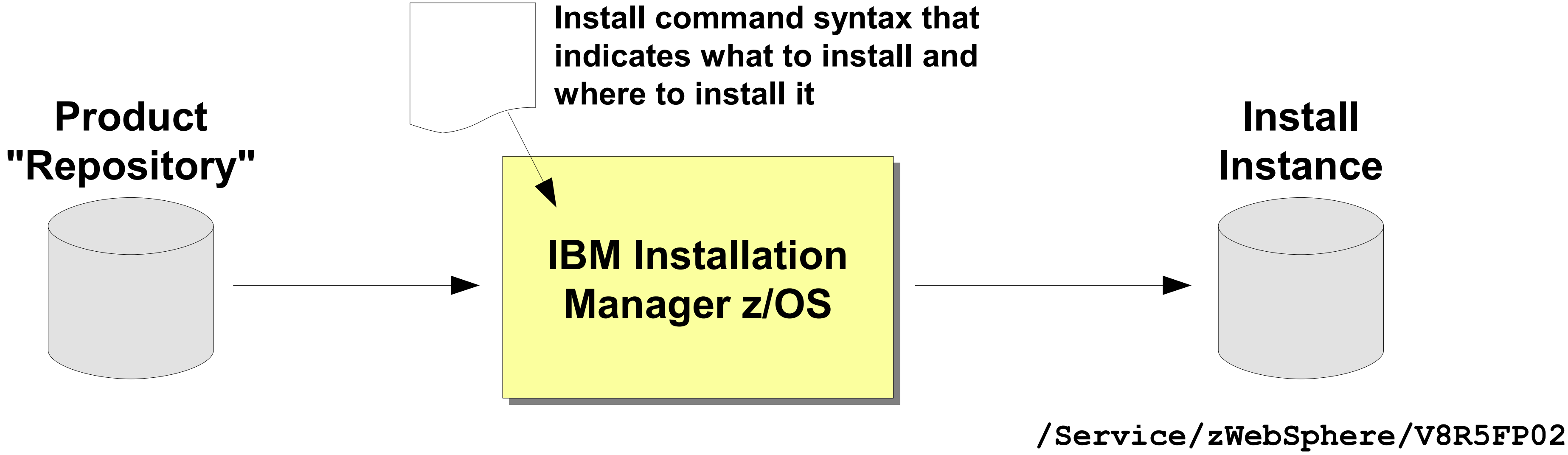
Installation/configuration …

Installation / Configuration

Setting a high-level baseline of how this is accomplished

Overview of Installation

Unit 5 of this workshop covers the details of this. Here we'll provide a very high-level recap of what's involved to install WAS z/OS:


Install command syntax that indicates what to install and where to install it
Product "Repository"
IBM Installation Manager z/OS
Install Instance
/Service/zWebSphere/V8R5FP02

This can be a local repository or the IBM repository hosted "in the cloud"

On z/OS this can be wrappered in JCL and run as a job that performs the install

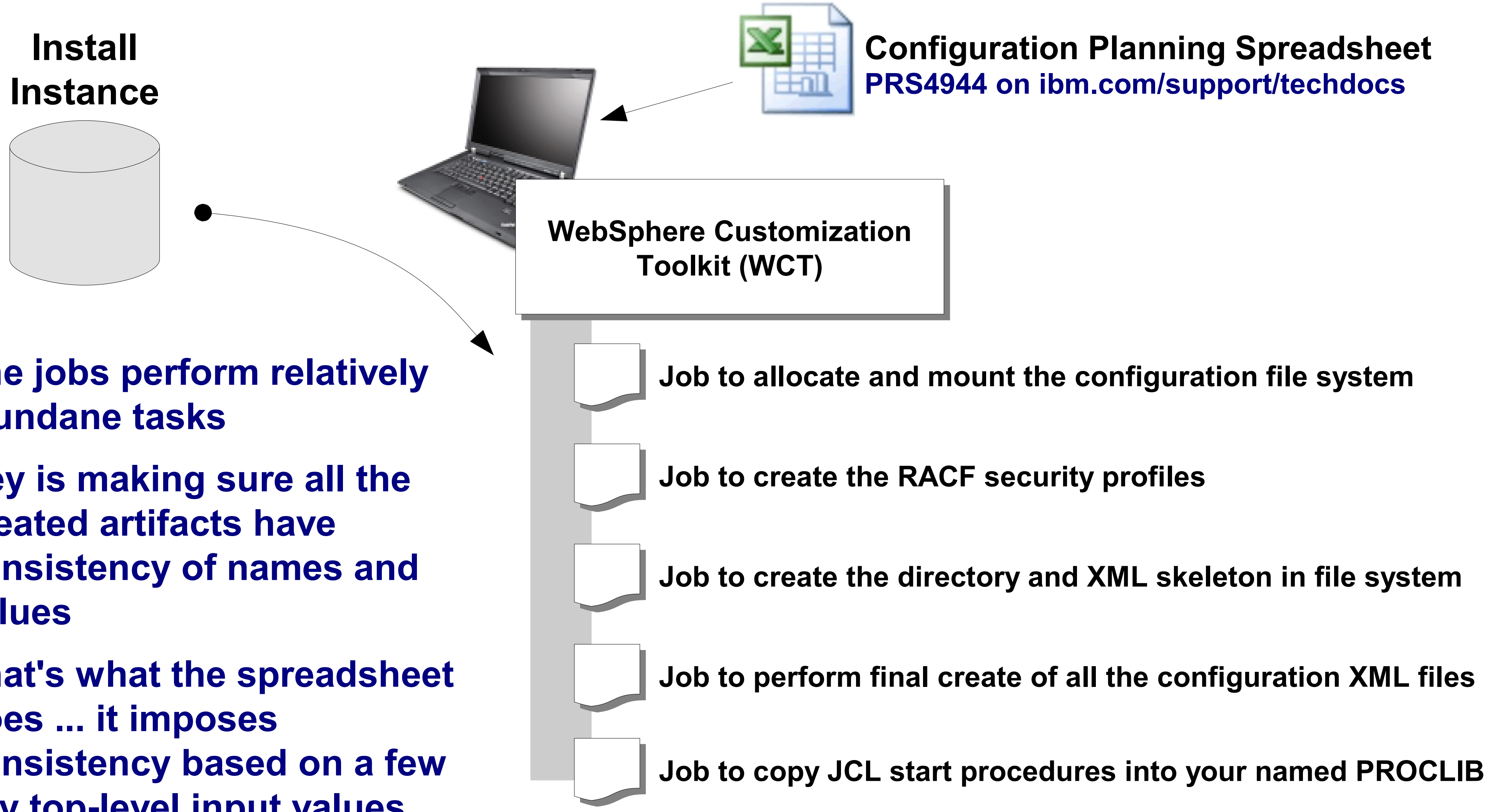
Typical practice is to have a separate install image for every version, release and fixpack you want to use

This is a departure from SMP/E. Unit 5 will discuss why IM was chosen for this and what advantages this brings when installing WAS z/OS

Creating runtime …

Overview of Creating the Runtime

This process has been the same for several versions now. It involves creating a set of customized z/OS jobs, then running those jobs to create the runtime enviornment:


Install Instance
Configuration Planning Spreadsheet
PRS4944 on ibm.com/support/techdocs
WebSphere Customization Toolkit (WCT)
Job to allocate and mount the configuration file system
Job to create the RACF security profiles
Job to create the directory and XML skeleton in file system
Job to perform final create of all the configuration XML files
Job to copy JCL start procedures into your named PROCLIB

The jobs perform relatively mundane tasks

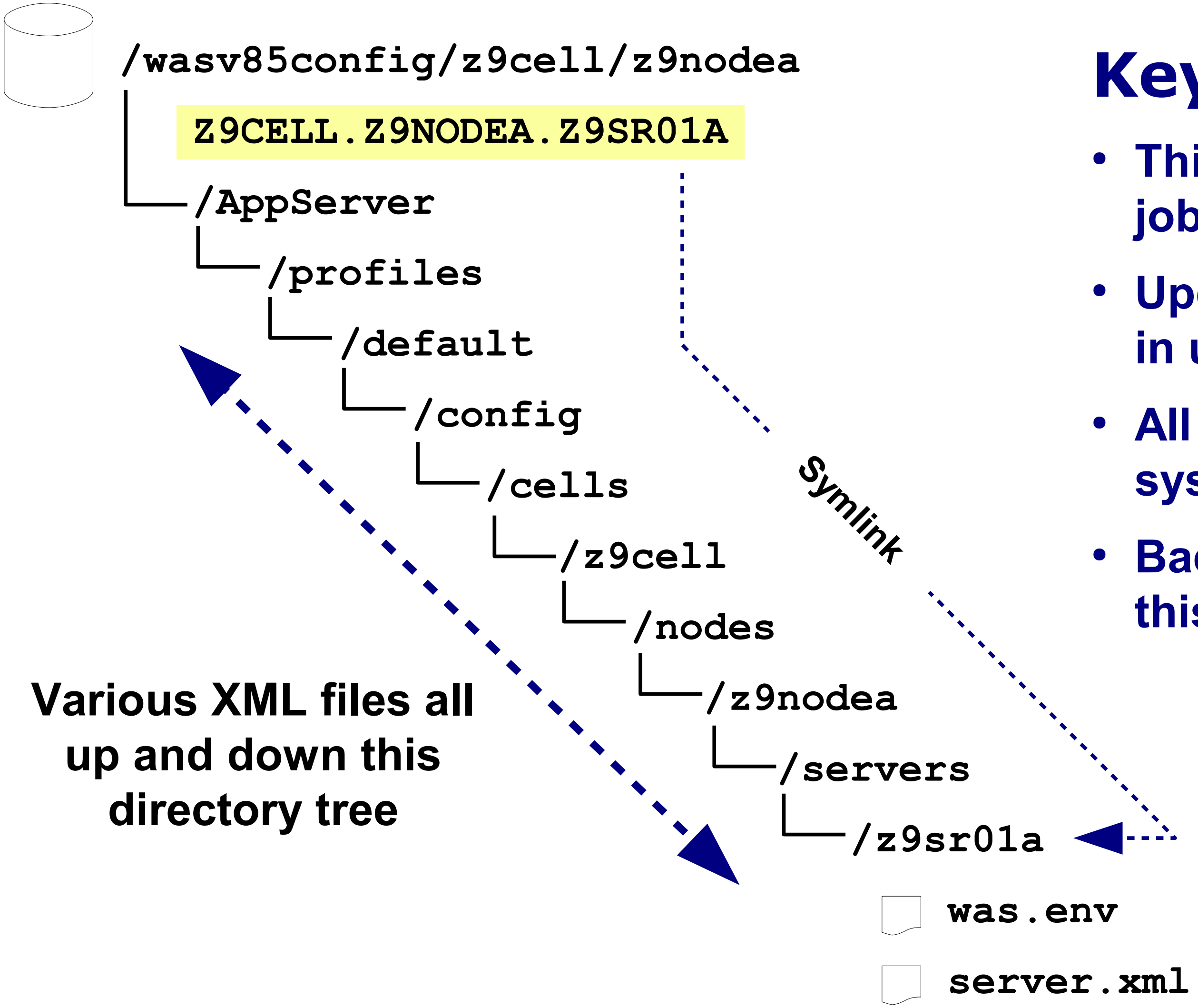
Key is making sure all the created artifacts have consistency of names and values

That's what the spreadsheet does ... it imposes consistency based on a few key top-level input values

Configuration file systems …

Overview of the Configuration File Systems

The configuration file systems contain directories and XML files that represent the runtime. Your customization ends up as changes to these directories and files:



```
/wasv85config/z9cell/z9nodea
  Z9CELL.Z9NODEA.Z9SR01A
  └─/AppServer
     └─/profiles
        └─/default
           └─/config
              └─/cells
                 └─/z9cell
                    └─/nodes
                       └─/z9nodea
                          └─/servers
                             └─/z9sr01a
                                  was.env
                                  server.xml
```

Various XML files all up and down this directory tree

This symlink comes into play on the MVS START command ... next chart

Key Points:

- This is built by customization jobs
- Updates in Admin Console result in updates to various XML files
- All this is contained in a UNIX file system
- Backup and restore is done at this level

Starting and stopping servers …

Starting and Stopping Servers

WAS z/OS servers operate as started tasks. Standard MVS START commands are used:

```
S Z9ACRA,JOBNAME=Z9SR01A,ENV=Z9CELL.Z9NODEA.Z9SR01A
```

JCL Start Procedure

`ENV=` is a pointer to the symlink that resolves to the server directory. This provides a way to overcome length limitations in z/OS for the `PARMS=''` string

One JCL proc may be used to start different servers in the node ... simply by passing a different `ENV=` string

Key Points:

- **In z/OS environment this is largely "business as usual" processing**
- **The server comes up as a started task (multiple address spaces as you'll see)**
- **It is possible to use supplied `startServer.sh` and `stopServer.sh` shell scripts (those end up issuing MVS START and STOP under the covers)**
- **Also use Admin Console to start and stop certain servers**

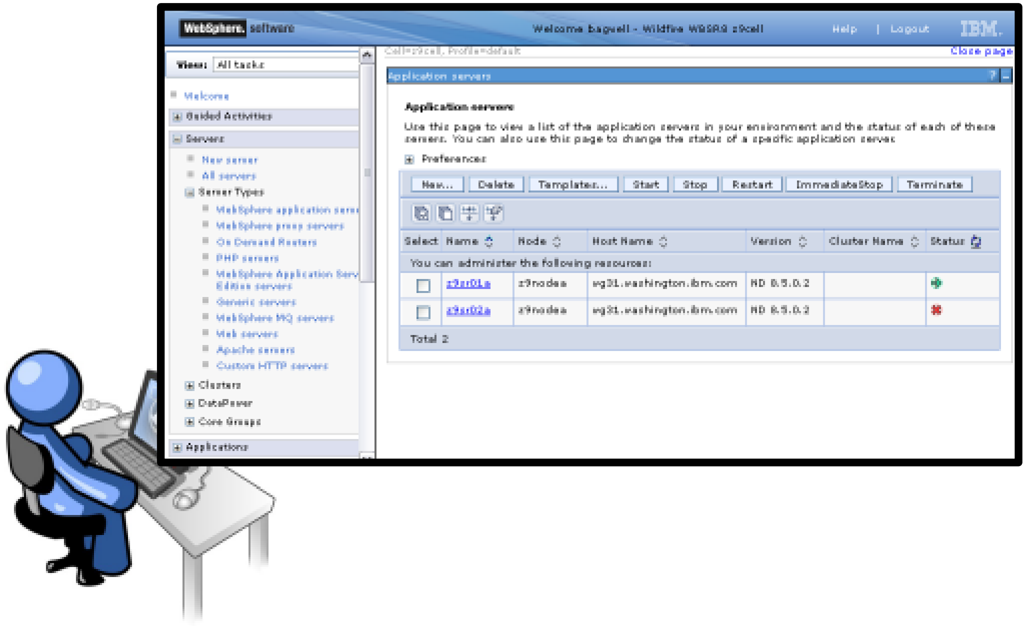
Administration overview …

Administration Overview

High level of the Admin Console and administration of runtime

The Deployment Manager and Admin Console

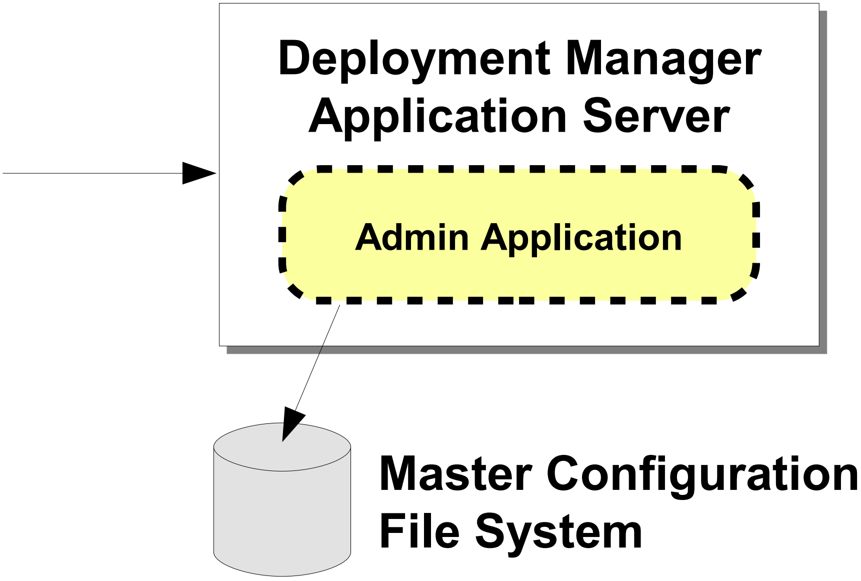
The Deployment Manager is an application server with a dedicated purpose: to run the Administrative Console application:



Application servers

Use this page to view a list of the application servers in your environment and the status of each of these servers. You can also use this page to change the status of a specific application server.

Select	Name	Node	Host Name	Version	Cluster Name	Status
You can administer the following resources:						
□	z9sr01a	z9nodea	wg31.washington.ibm.com	ND 8.5.0.2		➜
□	z9sr02a	z9nodea	wg31.washington.ibm.com	ND 8.5.0.2		✖
Total 2						

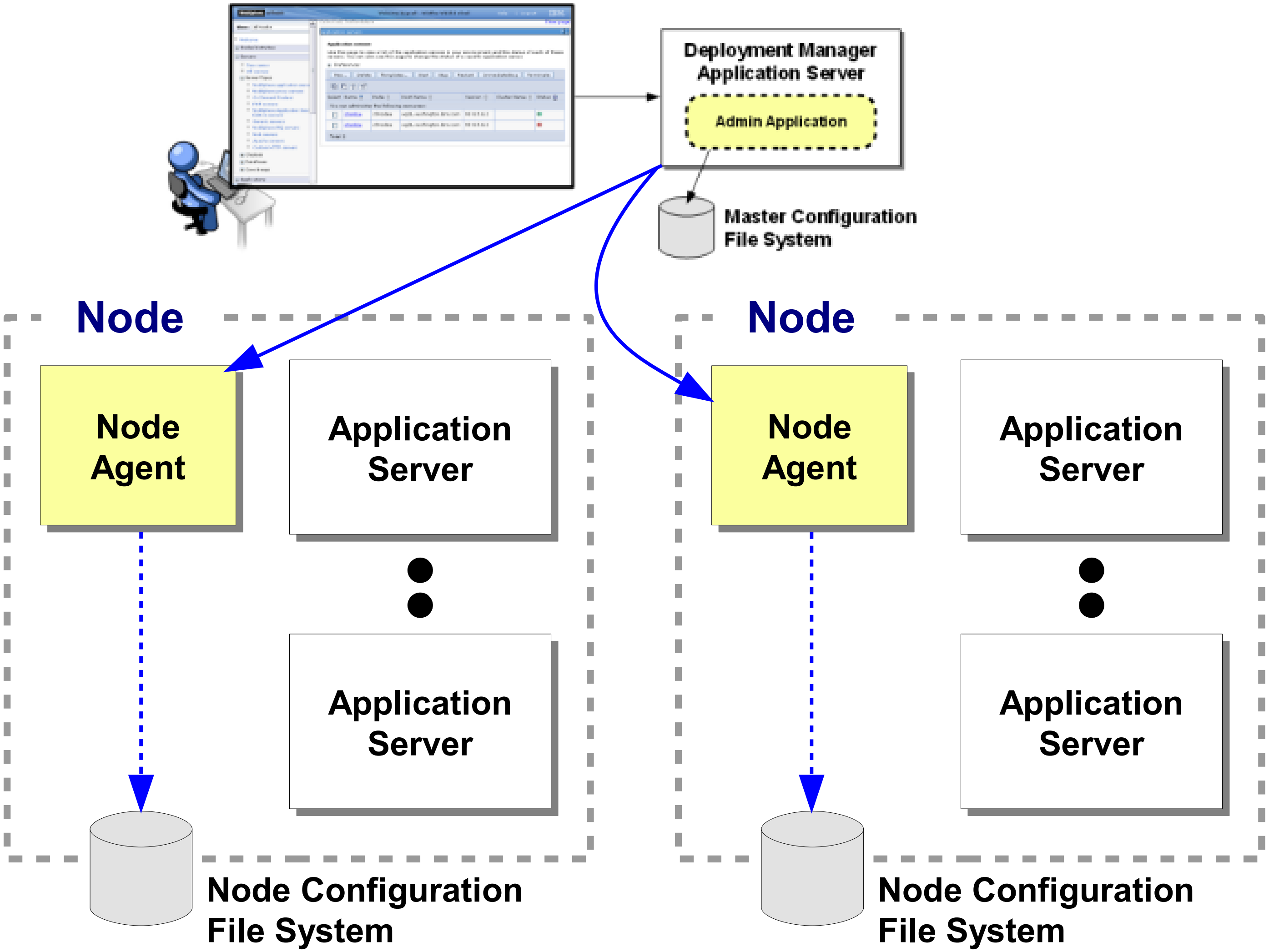


The Administrative Console's role is to turn your mouse clicks and keystrokes into the appropriate updates in the configuration file system XML tree

Nodes, Node Agents …

Nodes, Node Agents and Synchronization

Nodes are way of collecting up application servers on an LPAR. Node Agents are a way to get configuration changes from the master configuration out to the nodes:



Nodes are a collection of servers on an LPAR

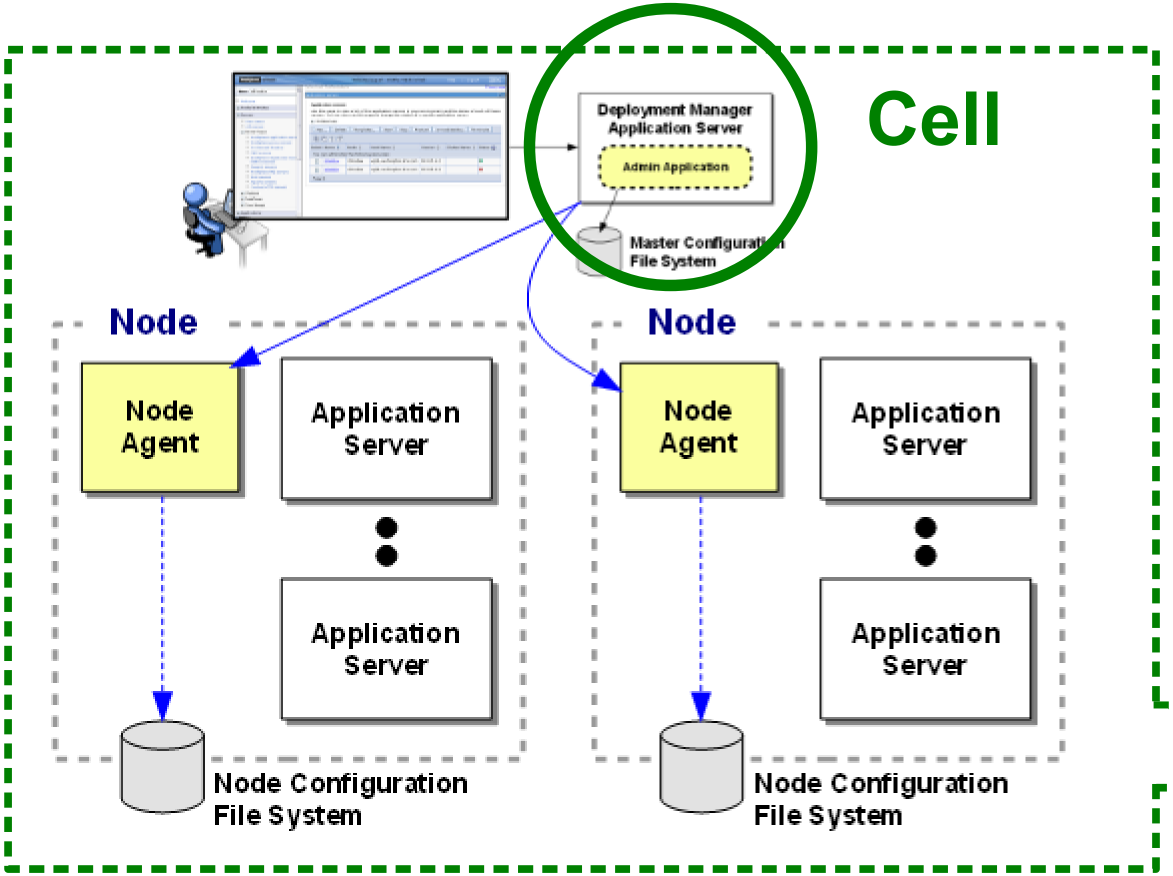
Admin Console updates the Master Configuration File System

Node Agents copy changed XML files from Master down to the node file system

The cell …

The "Cell" -- Boundary of Management Control

The "cell" is the extent of management control for a given Deployment Manager. Most run with multiple cells. And a cell can span platforms if you wish:



A "Cell" is the extent of management control for a given Deployment Manager

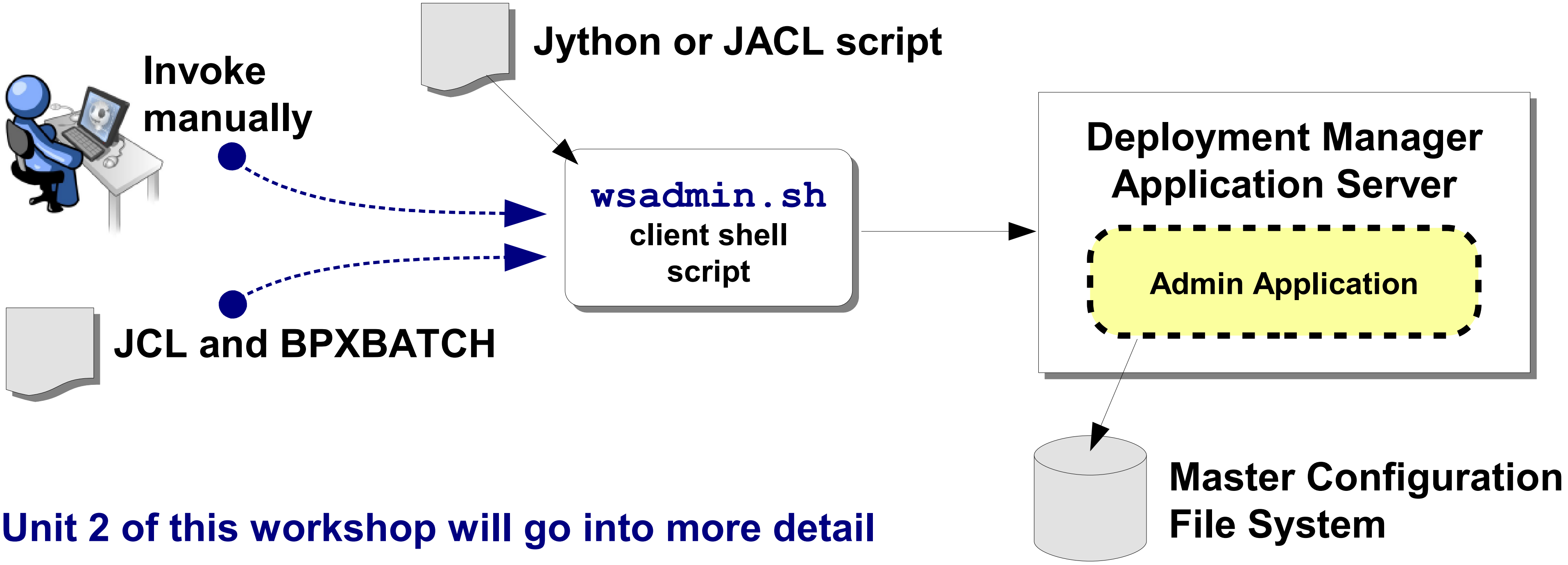
Often used to isolate security for Test, QA, Production

May span platforms ... issue there is simply coordination of SSL certificates

WSADMIN …

WSADMIN -- A Programmatic Interface

WSADMIN is a scripting interface to WAS (all platforms). It provides a way to programmatically perform administration actions:



Unit 2 of this workshop will go into more detail

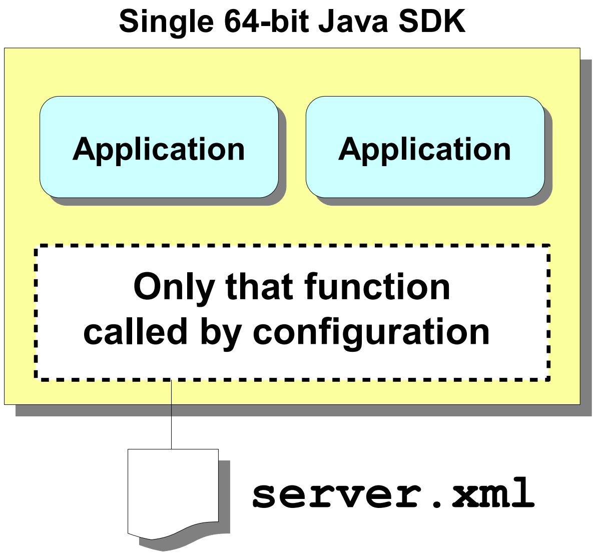
Anything you can do in Admin Console, you can do using WSADMIN scripting

Allows you to automate common tasks such as application deployment ... which provides *consistency* of actions across Test, QA, Prod

Liberty …

Liberty Profile

The Liberty Profile is a lightweight, dynamic, composable, single-JVM server model. It is offered along with "Traditional WAS z/OS" ...



- **Configuration file determines what functions are loaded**
- **Starts very quickly, consumes much less memory than traditional WAS z/OS**
- **Servlets, JSPs, web applications**

  Updated in V8.5.5 with additional features
- **Dynamic -- change server configuration or applications without server restart**
- **Not part of traditional WAS "cell" or "node" structure**

We have a section and lab on this topic

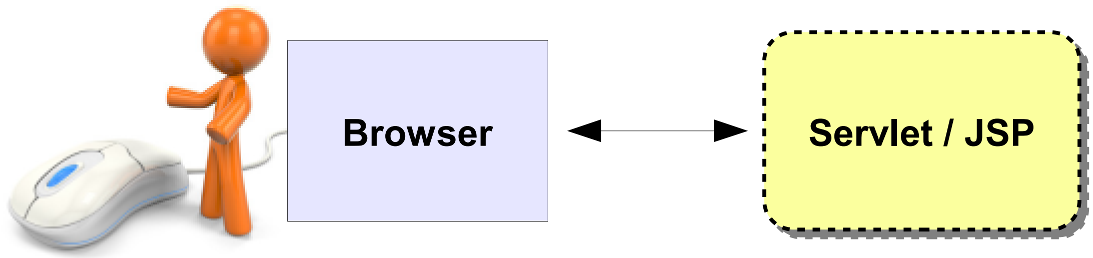
Applications ...

Applications

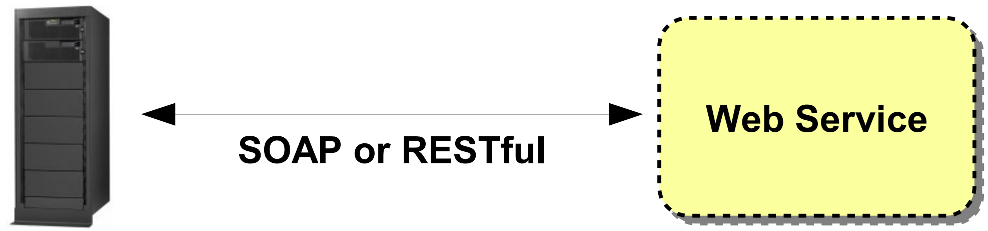
Overview of application development, packaging and deployment

Different Kinds of Applications

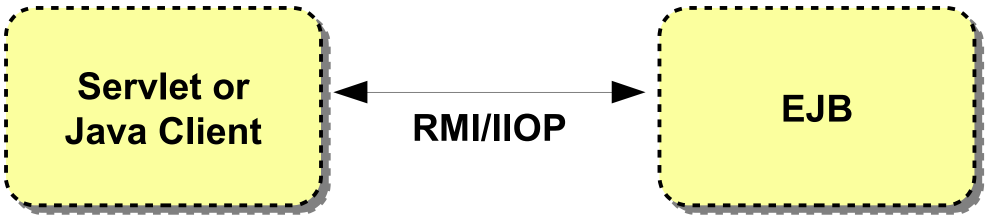
Here's a partial review of some common application "types" ...



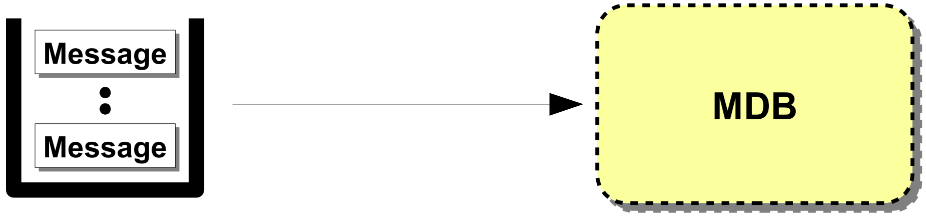
- Used to render user browser screens
- May interact with backend data
- May include JavaServer Faces or Struts (open source frameworks)



- Computer-to-Computer interaction
- Mobile devices
- Request type is HTTP



- Often used as backend to Servlet/JSP
- Often used as business logic and interaction with backend data
- Often packaged with Servlet/JSP

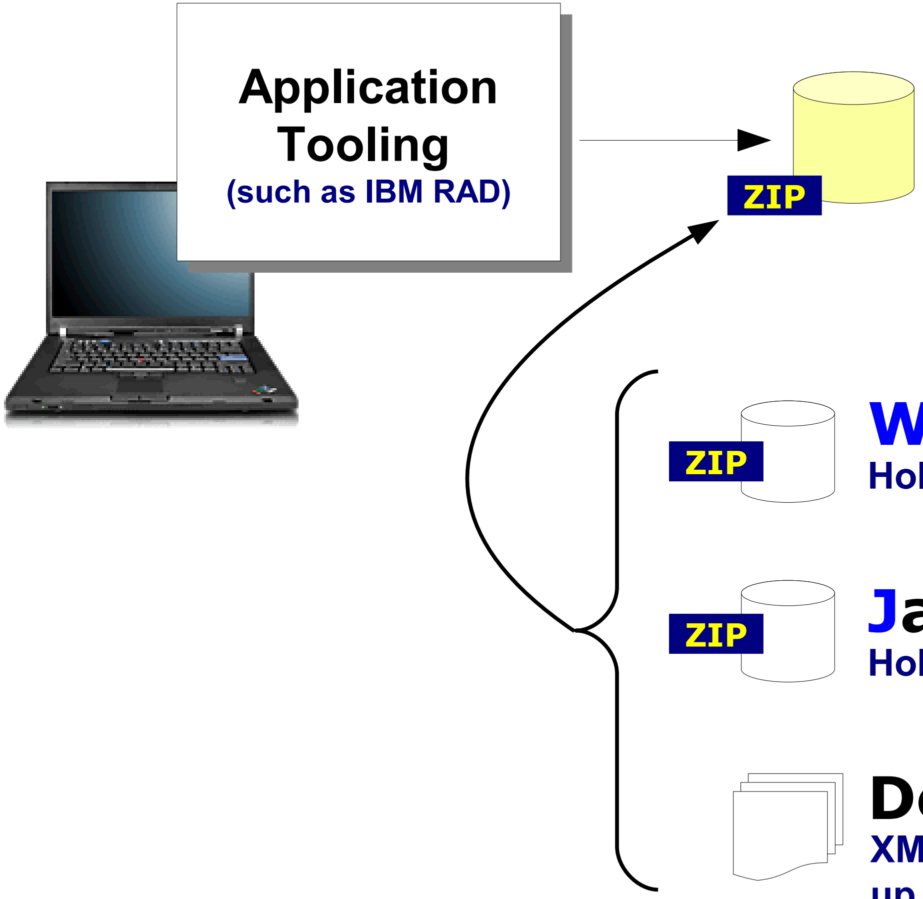


- MDB = Message Drive Bean
- Listens for message arrival on queue
- Picks message up and triggers action of application (often calls other EJBs)

Packaging ...

Application Packaging and Deployment

The unit of deployment is an "EAR" file -- a zip format file -- that the DMGR takes in, determines requirements, then updates XML so appservers understand updates:



Enterprise ARchive File (EAR)

Web ARchive File (WAR)
Holds web applications, such as servlets and JSPs

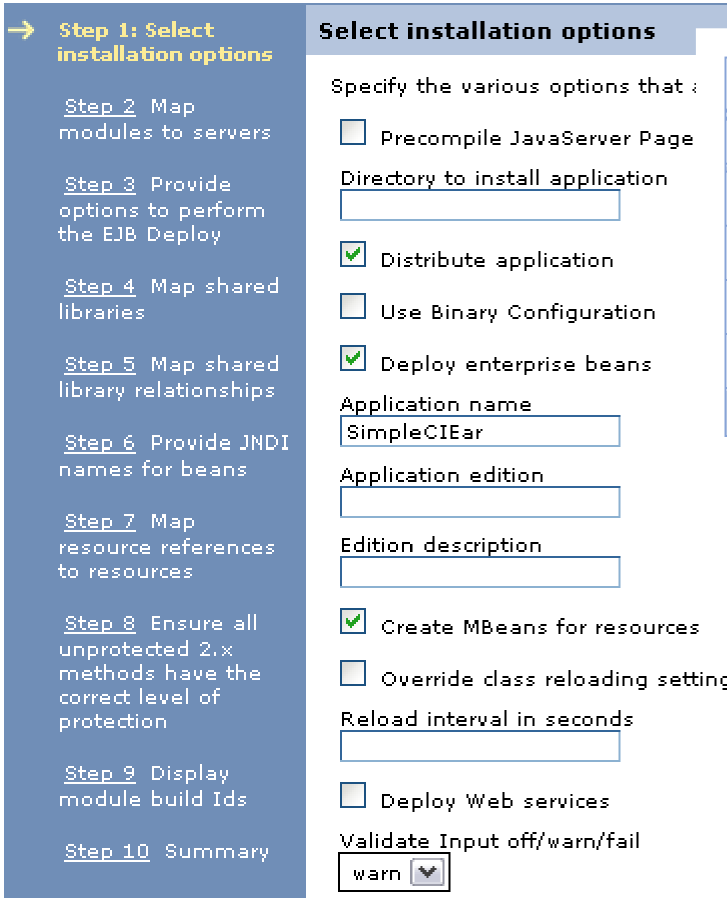
Java ARchive File (JAR)
Holds EJB applications

Deployment Descriptors
XML files that tell the story of the components packaged up in the JARs, WARs and the overall EAR

Deployment …

Applications Deployment in WAS z/OS

Application deployment is the same on WAS z/OS as it is on other WAS platforms. Can be done through Admin Console or WSADMIN. Some things to consider:



→ Step 1: Select installation options

Step 2 Map modules to servers

Step 3 Provide options to perform the EJB Deploy

Step 4 Map shared libraries

Step 5 Map shared library relationships

Step 6 Provide JNDI names for beans

Step 7 Map resource references to resources

Step 8 Ensure all unprotected 2.x methods have the correct level of protection

Step 9 Display module build Ids

Step 10 Summary

Select installation options

Specify the various options that a

□ Precompile JavaServer Page

Directory to install application

☑ Distribute application

□ Use Binary Configuration

☑ Deploy enterprise beans

Application name
SimpleCIEar

Application edition

Edition description

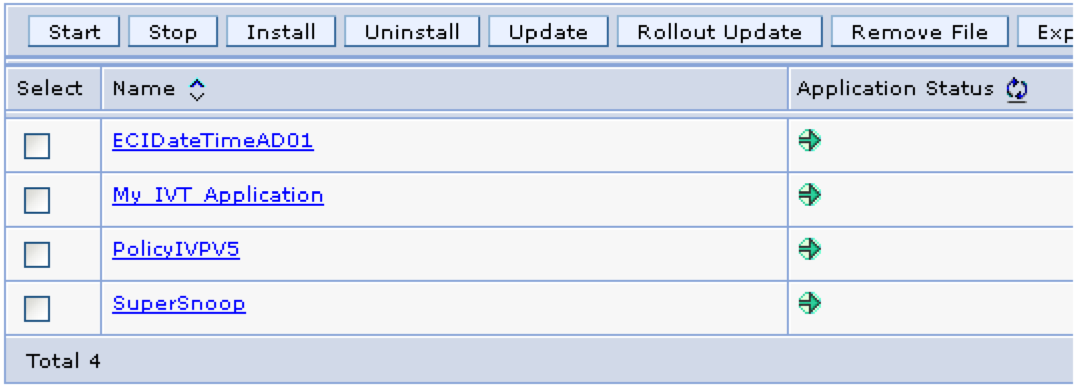
☑ Create MBeans for resources

□ Override class reloading settings for We

Reload interval in seconds

□ Deploy Web services

Validate Input off/warn/fail
warn



Select	Name	Application Status
□	ECIDateTimeAD01	➔
□	My IVT Application	➔
□	PolicyIVPV5	➔
□	SuperSnoop	➔
Total 4		

What backend data is the application seeking to use?

What other dependencies does the application have (other programs, other Java classes)?

Does the application have security requirements that need to be accounted for?

In general it is best if application developers and WAS administrators communicate with each other so deployment is as successful as possible

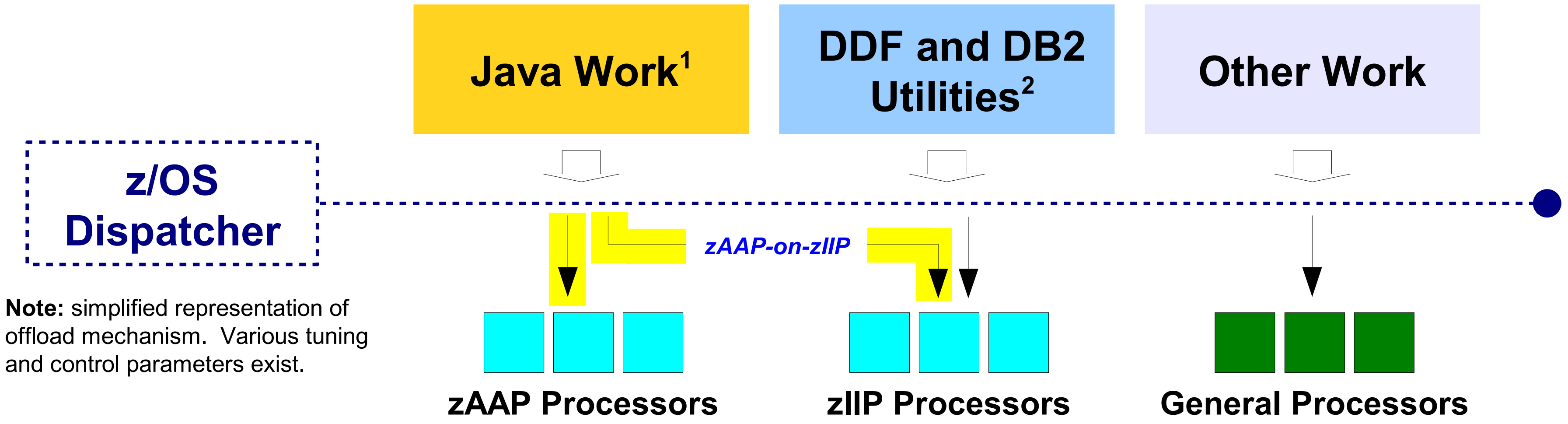
Taking advantage of platform …

Taking Advantage of z/OS

A review of the way WAS z/OS takes advantage of the platform

System z Specialty Engines

Specialty engines provide additional processing capacity with an attractive financial profile: lower acquisition cost, not counted towards software license charges:



Note: simplified representation of offload mechanism. Various tuning and control parameters exist.

zAAP - *System z Application Assist Processor*

Offload of Java and XML parsing work.

zIIP - *System z Integrated Information Processor*

Certain DB2 work and XML parsing services.

zAAP-on-zIIP

A means of more efficiently using specialty engines by defining only a pool of zIIP processors and allowing eligible zAAP work to run on the zIIPs[3].

IFL - *Integrated Facility for Linux*

For running z/VM and Linux. Does not apply to z/OS, but plays strong role in Linux for System z

Note 1 -- See `http://www.ibm.com/systems/z/hardware/features/zaap/`

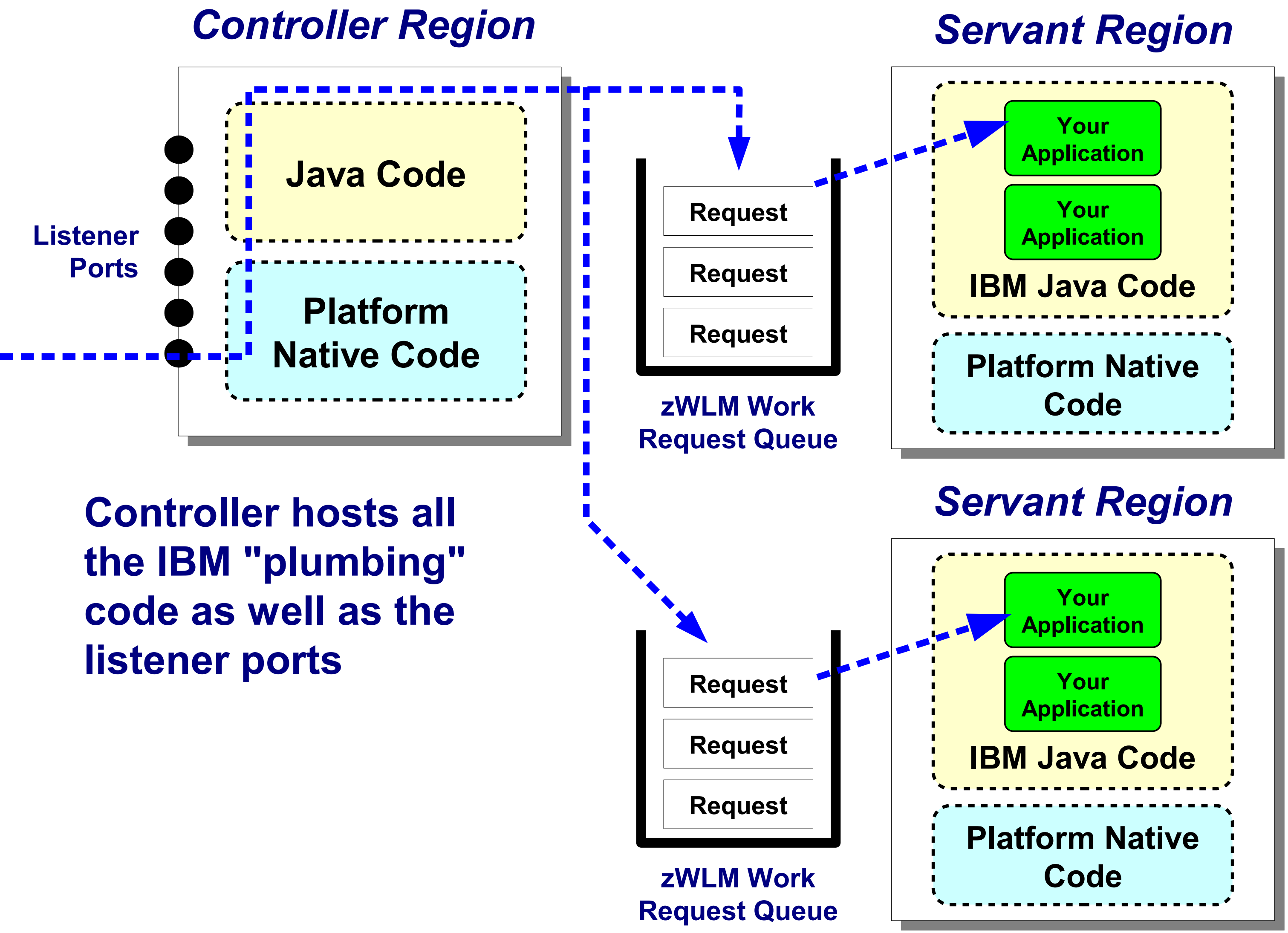
Note 2 -- Plus other work, see `http://www.ibm.com/systems/z/hardware/features/ziip/`

Note 3 -- EC12 planned to be the last system that supports zAAP; after that, zAAP-on-zIIP will be the offload mechanism

Multi-JVM Design …

WAS z/OS and the "Multi-JVM" Design

We have a whole section on exploiting this feature. For now, focus on the essentials:



Servant region hosts applications

zWLM work queue acts as intermediary point for requests

Servants "pull" work

Controller hosts all the IBM "plumbing" code as well as the listener ports

Additional servants may be started ... by your or by zWLM

Provides vertical scaling ...

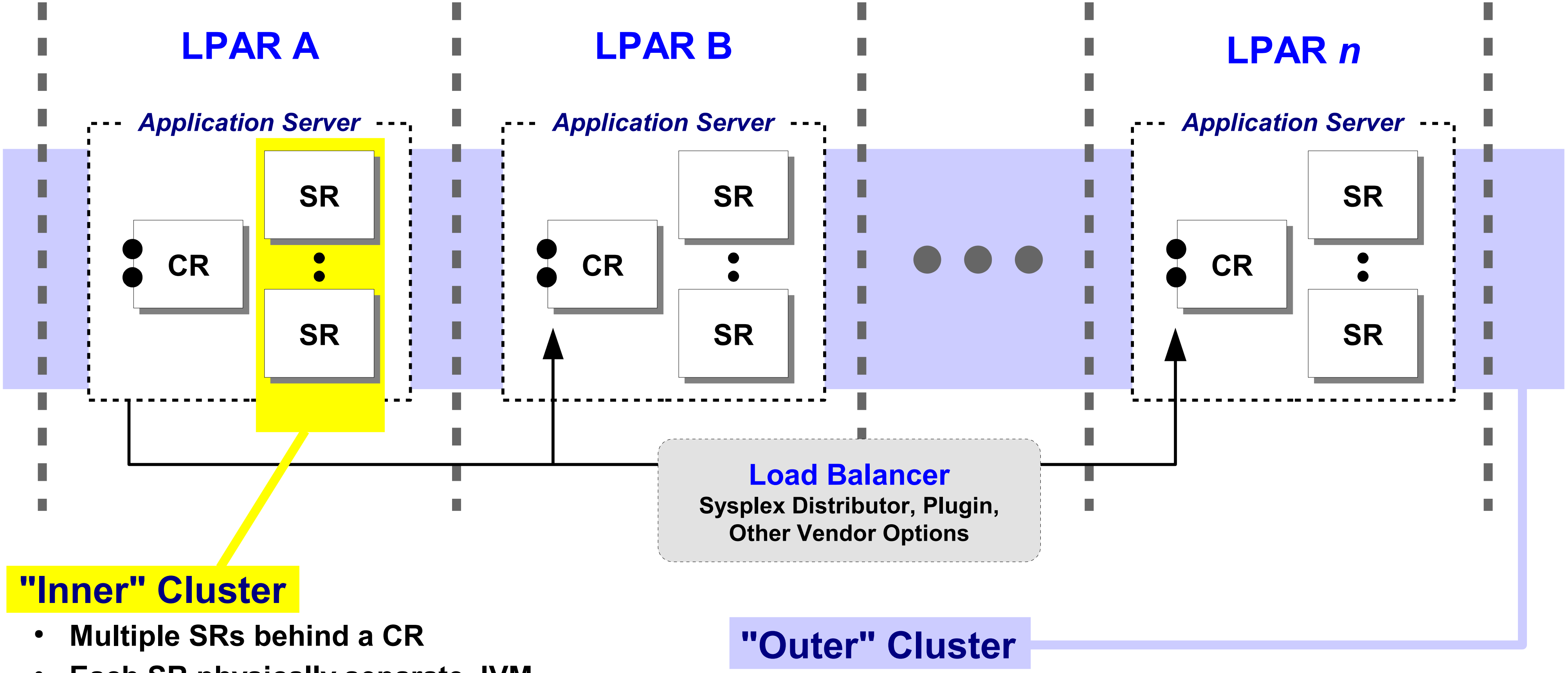
... also classification and work placement

TechDocs WP101740

Clusters ...

Clusters - "Inner" and "Outer"

With WAS z/OS we have two levels of clustering for availability:



"Inner" Cluster

- Multiple SRs behind a CR
- Each SR physically separate JVM
- App binaries in each JVM
- Each SR has own worker thread pool
- WLM will restart failed SR
- WLM will distribute work (Unit 3)
- Stateful replication possible

"Outer" Cluster

- Multiple appservers across LPARs
- WebSphere cluster common across platforms
- App binaries in each appserver's SRs
- Stateful replication possible
- Many options for front-end work distribution

Other examples …

WLM, SAF, SMF, MODIFY, Cross-Memory

Other points of platform exploitation are those summarized here:

★ **WLM**

z/OS Workload Manager

- Controller / Servant structure as discussed on previous chart
- Request classification for separate service classes and reporting classes

SAF

z/OS Security Access Facility

- Sysplex-aware security definition repository and resource access control
- Userids and passwords, SSL certificates, EJBROLE definitions
- Security workshop covers WAS z/OS security in detail (ask for details if interested)

SMF

z/OS System Management Facilities

- SMF 120.9 record to record detailed information about request activity
- Useful for analysis and chargeback
- See WP102205 at ibm.com/support/techdocs for guide to SMF Techdocs

★ **MODIFY**

z/OS MODIFY interface

- Allows dynamic operations against WAS z/OS servers
- Long list of actions to display and act up on server operational behavior

★ **Cross Memory**

z/OS Cross-Memory Exchange

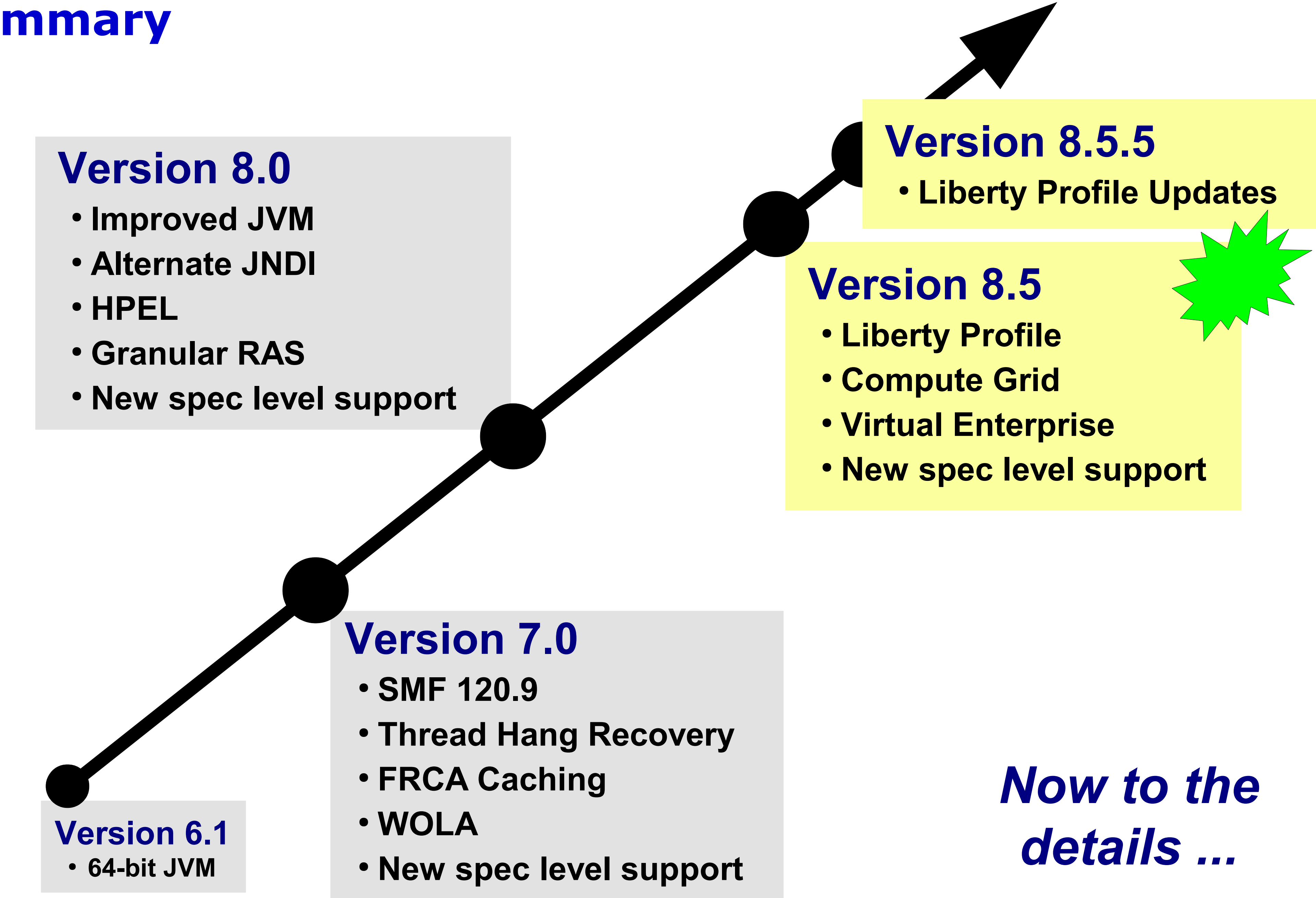
- DB2 Type 2 connector, CICS EXCI, MQ BINDINGS, WOLA
- Low latency, better security

Wrap up …

Wrap-Up

Final thoughts before getting to the rest of the workshop

Summary



Version 6.1

- 64-bit JVM

Version 7.0

- SMF 120.9
- Thread Hang Recovery
- FRCA Caching
- WOLA
- New spec level support

Version 8.0

- Improved JVM
- Alternate JNDI
- HPEL
- Granular RAS
- New spec level support

Version 8.5

- Liberty Profile
- Compute Grid
- Virtual Enterprise
- New spec level support

Version 8.5.5

- Liberty Profile Updates

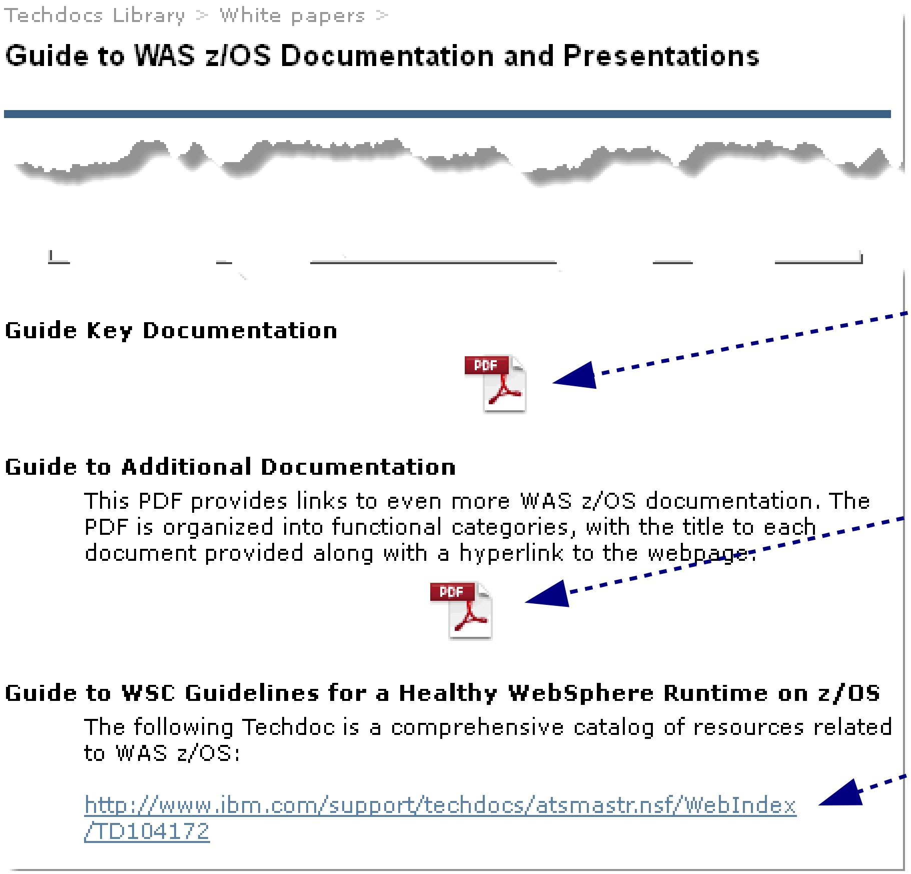
Now to the details ...

Techdocs ...

Techdocs

We've published a great deal of useful information out on the Techdocs site. So many that we decided to publish a "guide" to all the documents ... WP102205

ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP102205



Techdocs Library > White papers >

Guide to WAS z/OS Documentation and Presentations

Guide Key Documentation

Guide to Additional Documentation

This PDF provides links to even more WAS z/OS documentation. The PDF is organized into functional categories, with the title to each document provided along with a hyperlink to the webpage.

Guide to WSC Guidelines for a Healthy WebSphere Runtime on z/OS

The following Techdoc is a comprehensive catalog of resources related to WAS z/OS:

http://www.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/TD104172

PDF with hiperlinks to the key documents for WAS z/OS

PDF with hiperlinks to important documents not listed in the "key documents" PDF

John Hutchinson's "Healthy Runtime" information

The labs ...

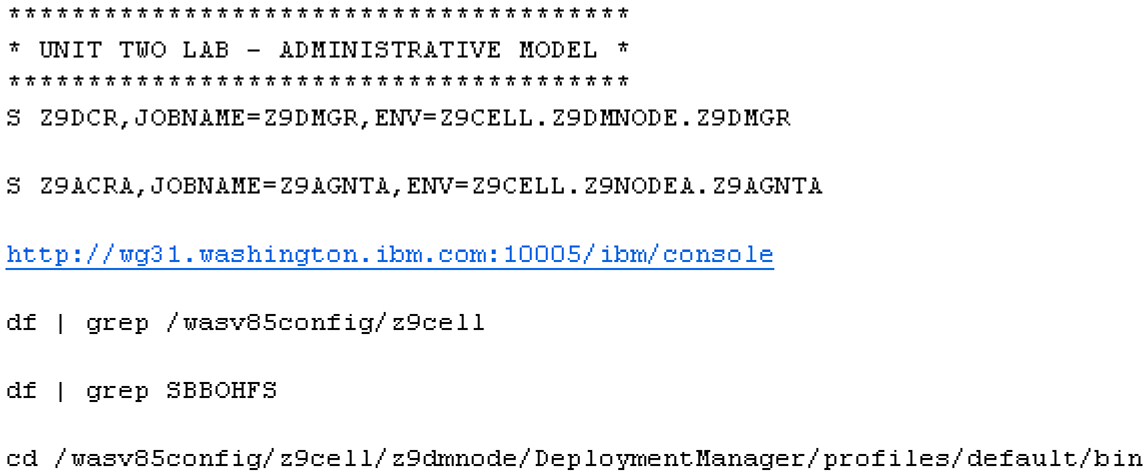
Few Notes About the Labs

Slow and steady … lots of information, so trying to rush usually results in overlooking things

MVS and ISPF usage hints in the back

Cut-and-paste command text file on desktop

WBSR85 Lab Commands for Cut-and-Paste .txt



```
***************************************
* UNIT TWO LAB - ADMINISTRATIVE MODEL *
***************************************
S Z9DCR,JOBNAME=Z9DMGR,ENV=Z9CELL.Z9DMNODE.Z9DMGR

S Z9ACRA,JOBNAME=Z9AGNTA,ENV=Z9CELL.Z9NODEA.Z9AGNTA

http://wg31.washington.ibm.com:10005/ibm/console

df | grep /wasv85config/z9cell

df | grep SBBOHFS

cd /wasv85config/z9cell/z9dmnode/DeploymentManager/profiles/default/bin
```