

*Hendrik Wörner*

*IT-Specialist for WebSphere on System z*

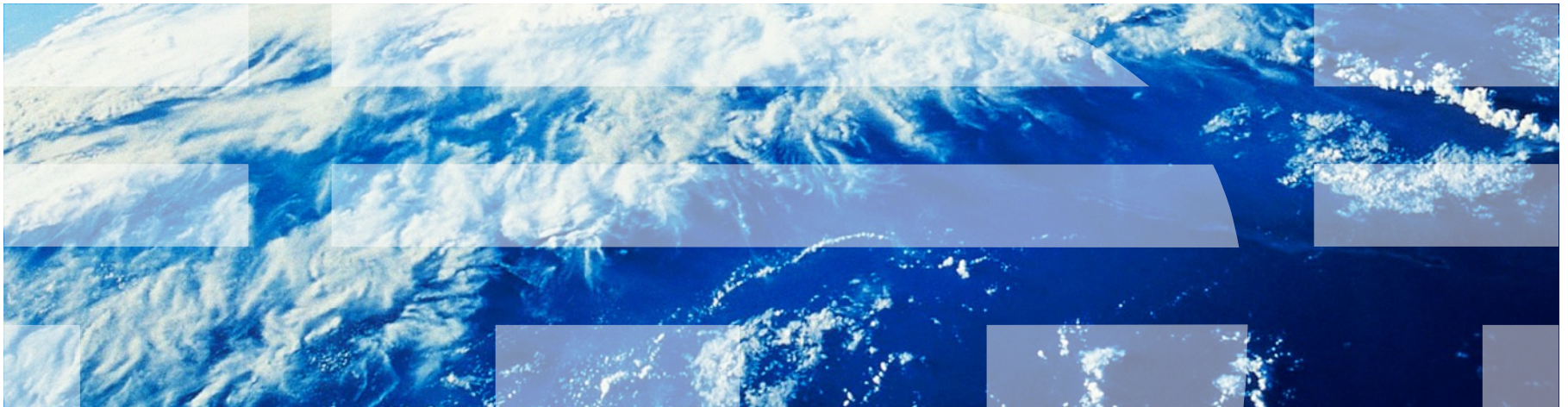
*[hwoerner@de.ibm.com](mailto:hwoerner@de.ibm.com)*

*Joerg-Ulrich Veser*

*IT-Specialist for WebSphere on System z*

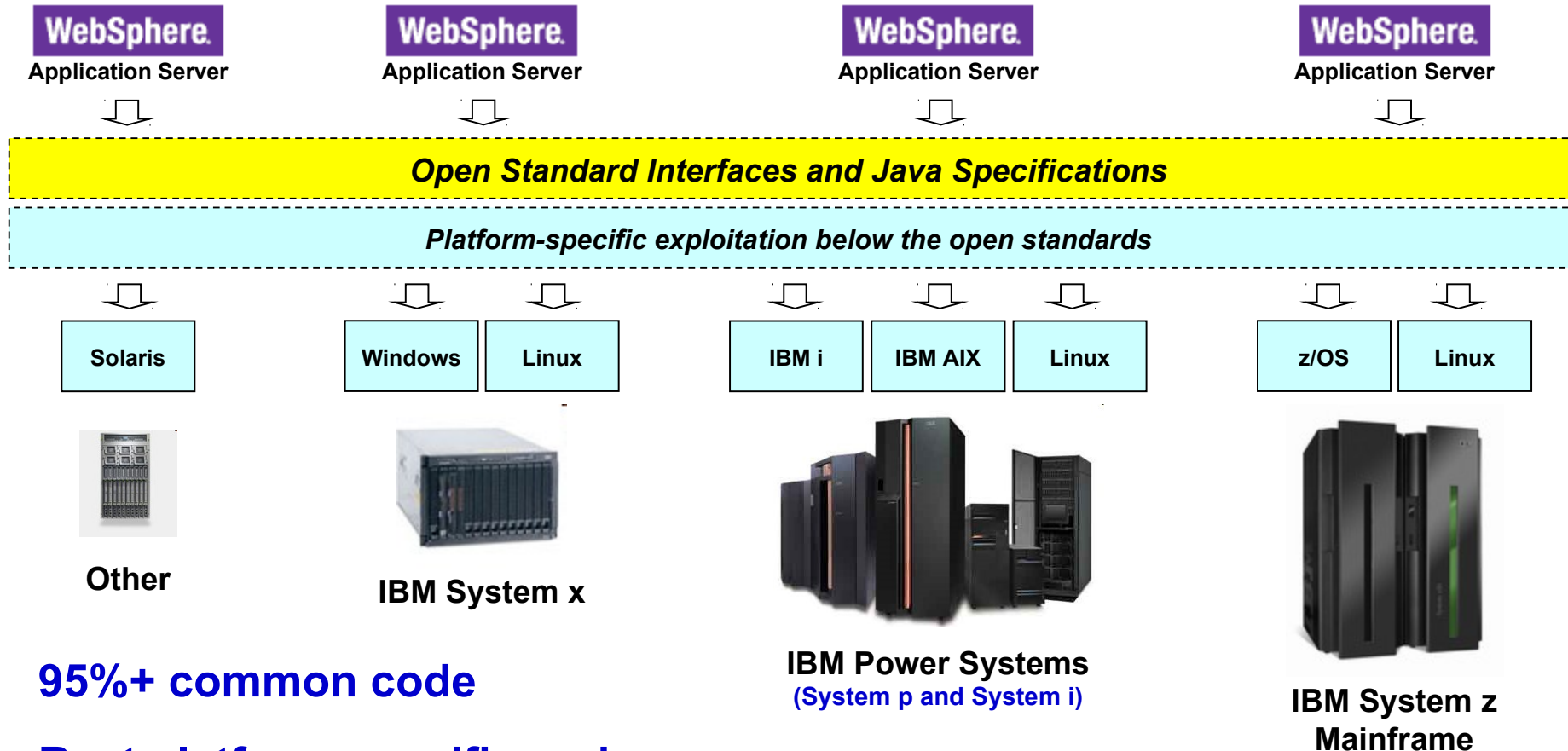
*[jveser@de.ibm.com](mailto:jveser@de.ibm.com)*

# WebSphere Application Server z/OS Update



## WebSphere is WebSphere Above the Specification Layer

This is a key strategic statement from IBM. You will have aligned specifications and delivery schedules for WAS across the platforms



**95%+ common code**

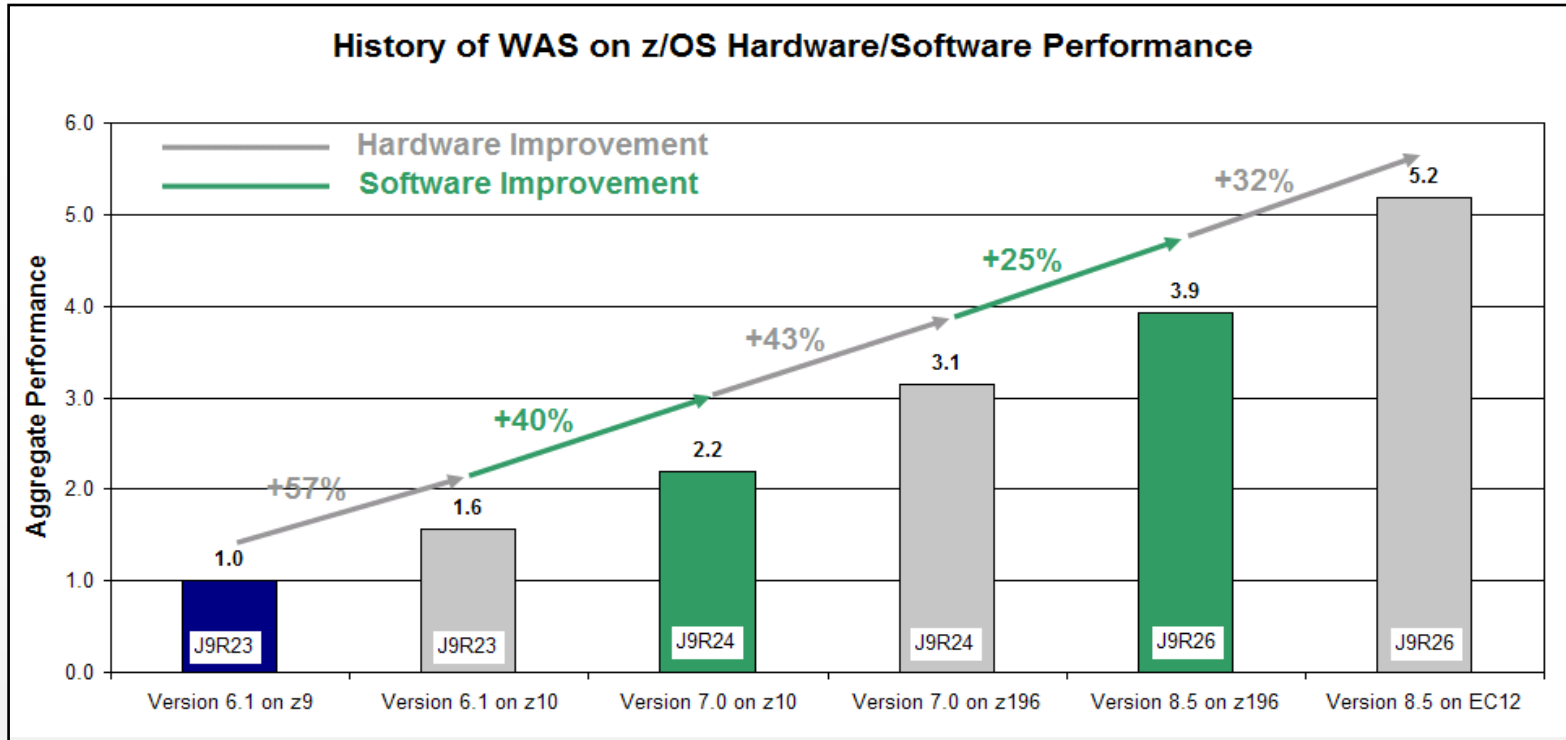
**Rest platform-specific code**

Java and JVM ...

© 2012 IBM Corporation

# WebSphere z/OS Performance

Aggregate HW, SDK and WAS Improvement: WAS 6.1 (Java 5) on z9 to WAS 8.5 (Java 7) on zEC12

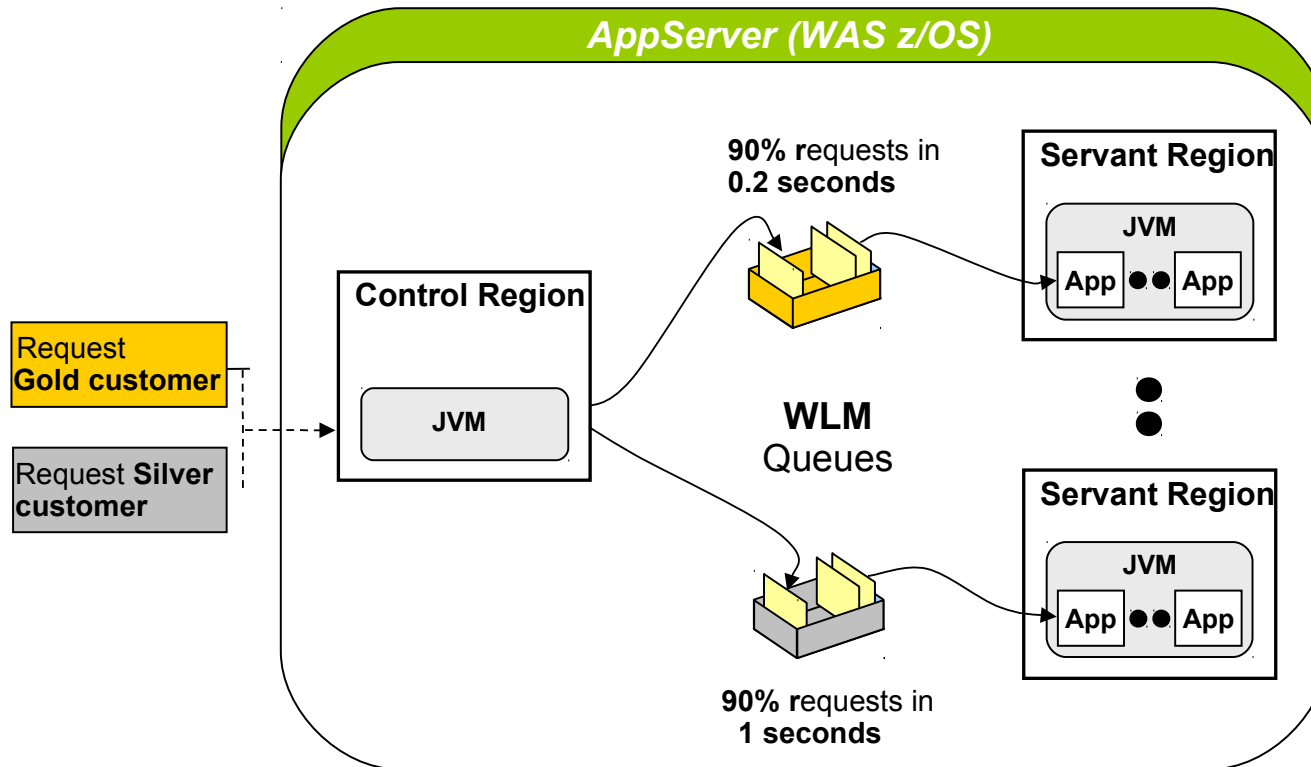


**~5x aggregate hardware and software improvement comparing WAS 6.1 Java5 on z9 to WAS 8.5 Java7 on zEC12**

# Better Availability through built-in Mini-Cluster and z/OS Workload Manager

**Availability and Scalability** - One logical **Application Server** on z/OS can already be a Mini-Cluster, which **consists of multiple JVMs**. Depending on the workload additional JVMs can be started automatically.

**Definition of Service Level Agreements** in terms of WLM goals is possible. For instance it is possible depending on the customer status to assign additional resources.



## Availability:

Already one logical Application Server can consist of multiple Java Virtual Machines (JVM).

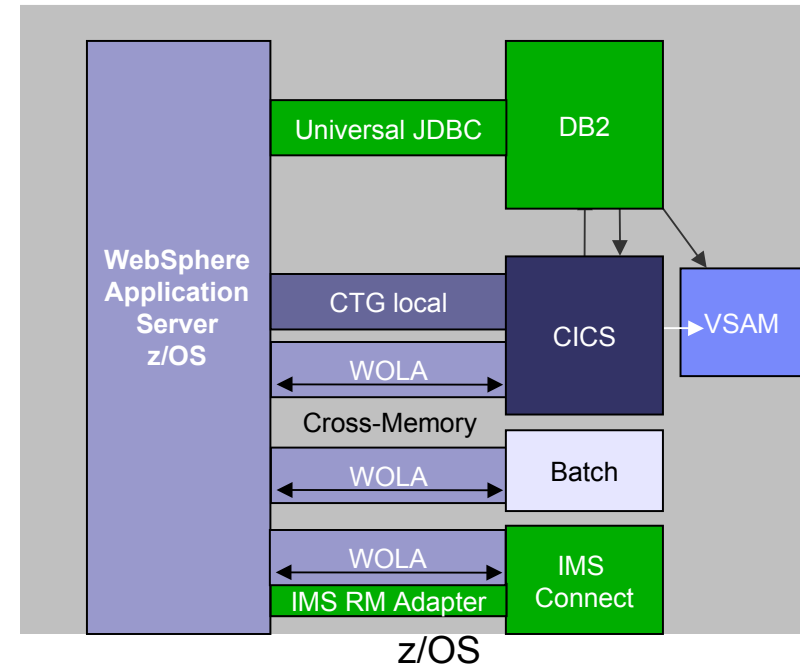
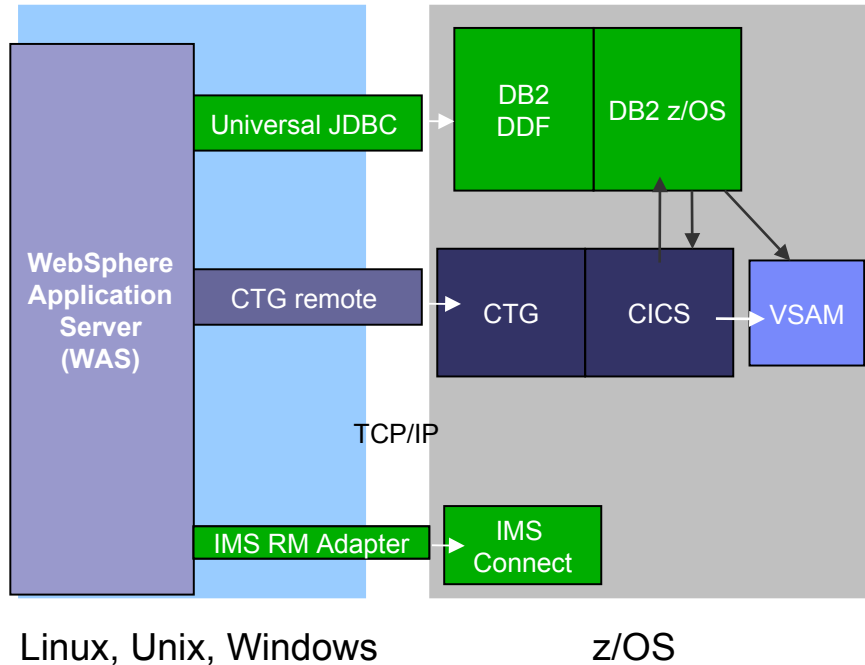
## Scalability:

Depending on the system utilization WLM can start additional JVMs automatically.

## Quality of Services:

WLM can prioritize the workload. Service-Level-agreements can be defined.

## WAS z/OS: Access to data in z/OS using multiple physical Server-Layers compared to one physical layer

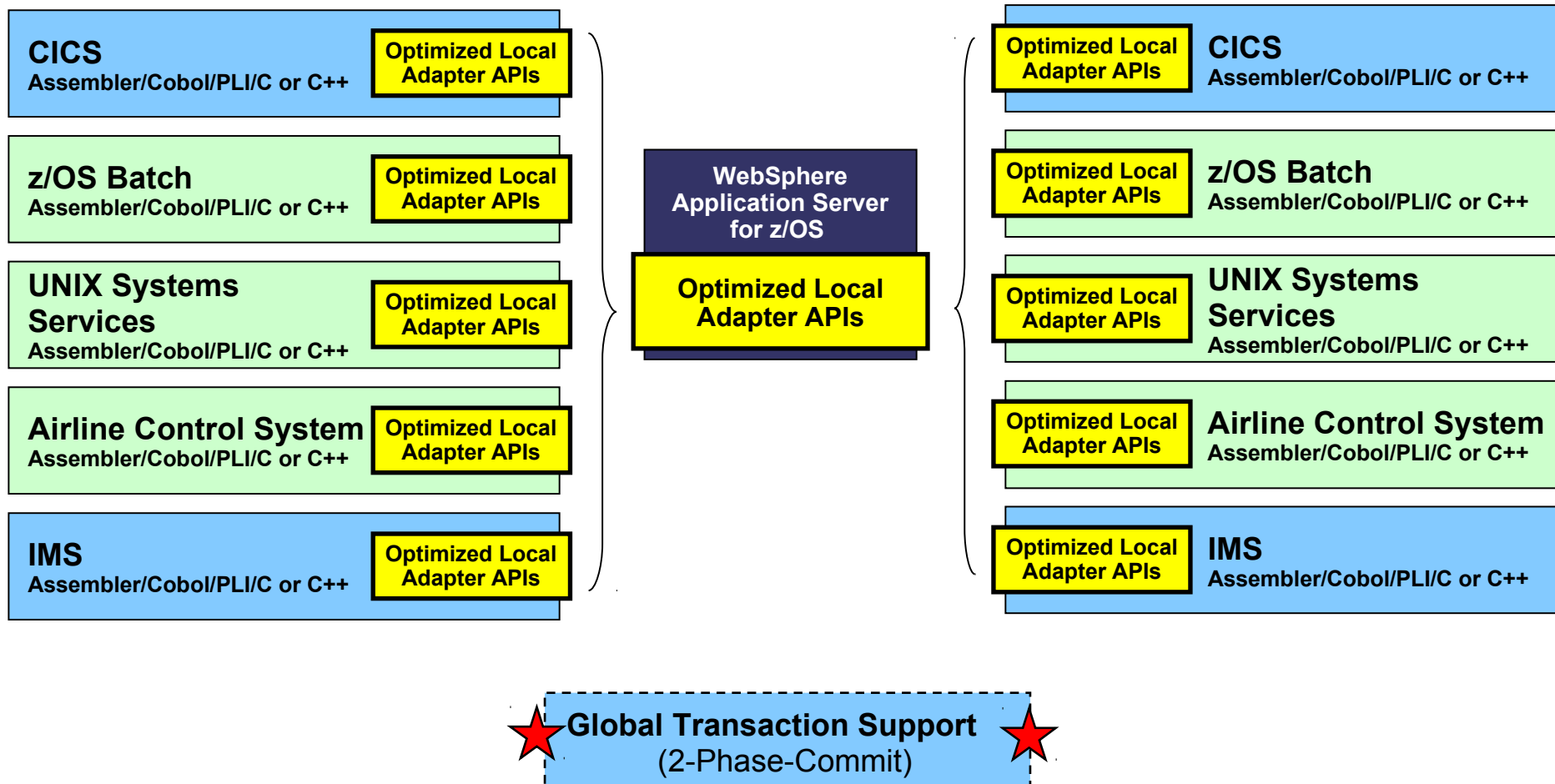


The consolidation of multiple physical Server-Layer has many advantages: increased security, extended and extended and simplified management, lower costs and gain in performance

**The operation of less physical Server-Layers makes it easier, to reach non-functional requirements.**

## Cross-Memory: WebSphere Optimized Local Adapters (WOLA)

The new WOLA Adapters allow a bi-directional Cross-Memory communication between WAS and CICS, IMS, z/OS Batch and USS.

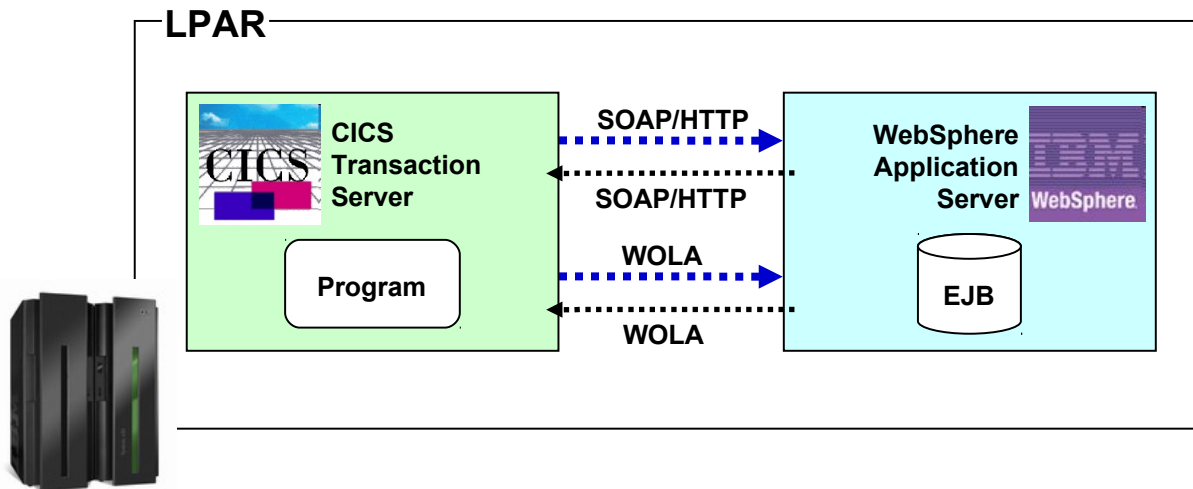


## A Comparison of CICS Invoking WAS Web Services

**A recent benchmark compared the case of web services into WAS against invoking the EJB using WOLA:**

**Testcase:** CICS calls the same EJB in WAS z/OS over

- 1) WebServices
- 2) WOLA

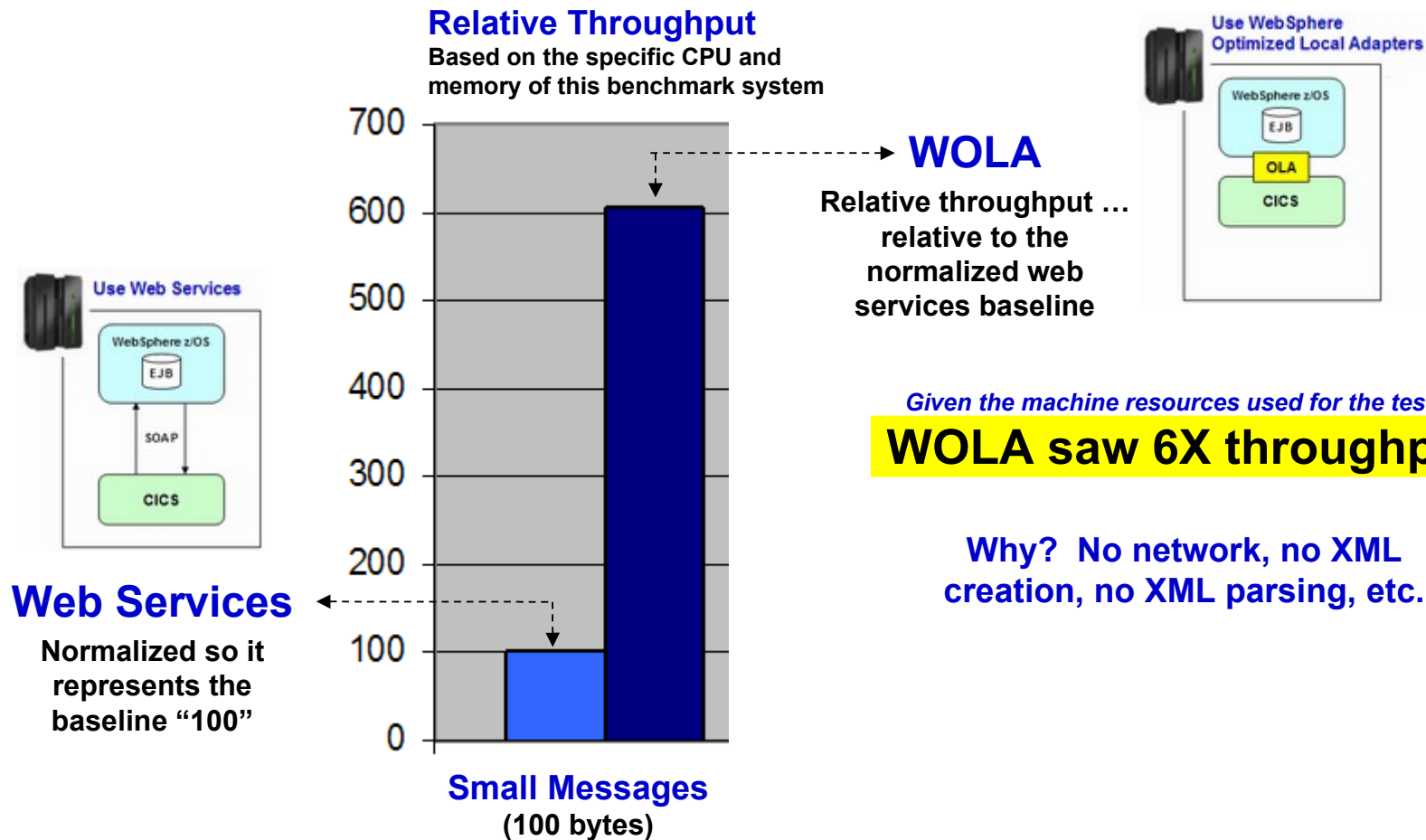


### Some notes:

- We are not suggesting WOLA can or should replace web services in all cases. But in some cases it might well make sense
- In this test CICS and WAS on same LPAR, so TCP is optimized between the two
- The test was a relatively simple echo application

## Relative Throughput for 100 Byte Message

The following chart serves as our starting point ... a comparison of relative throughput based on a 100 byte message exchange:



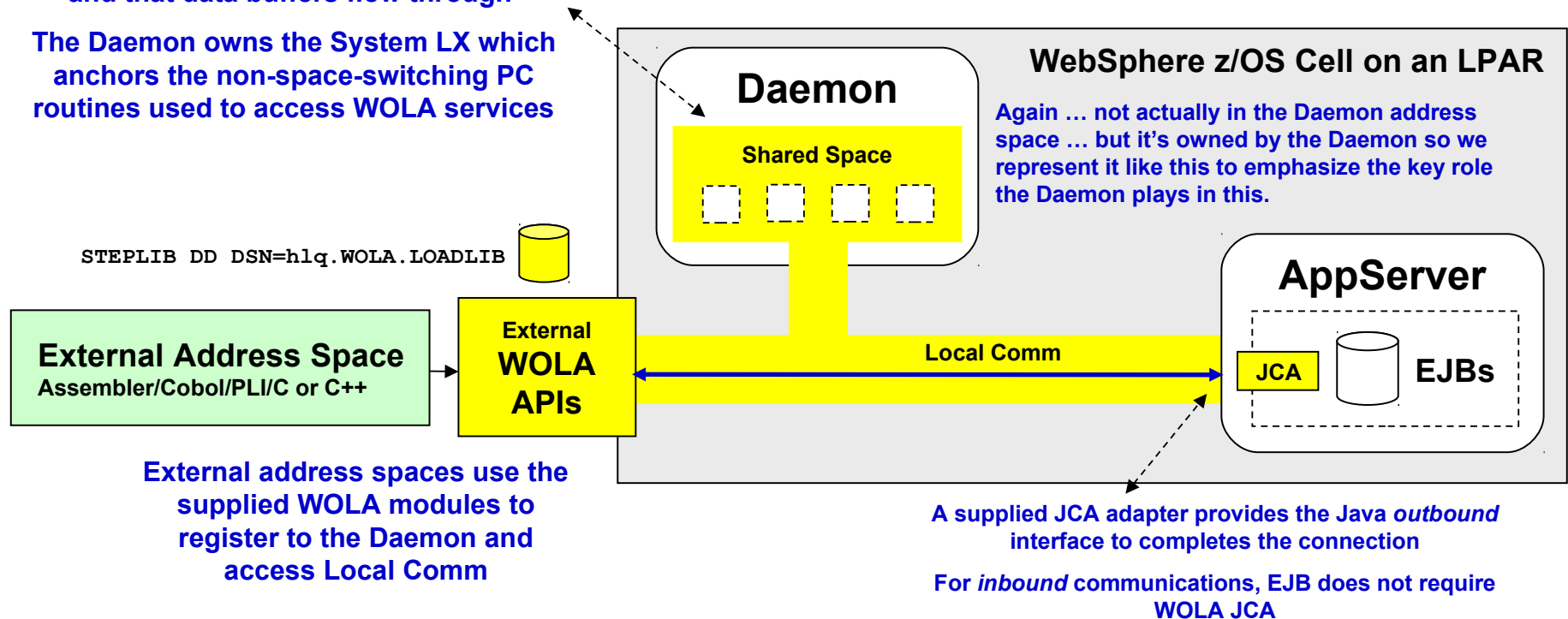


## Very High Level Picture of WOLA

This is just a schematic, but it helps position some key concepts:

The Daemon owns the shared above-the-bar storage where WOLA registrations live and that data buffers flow through

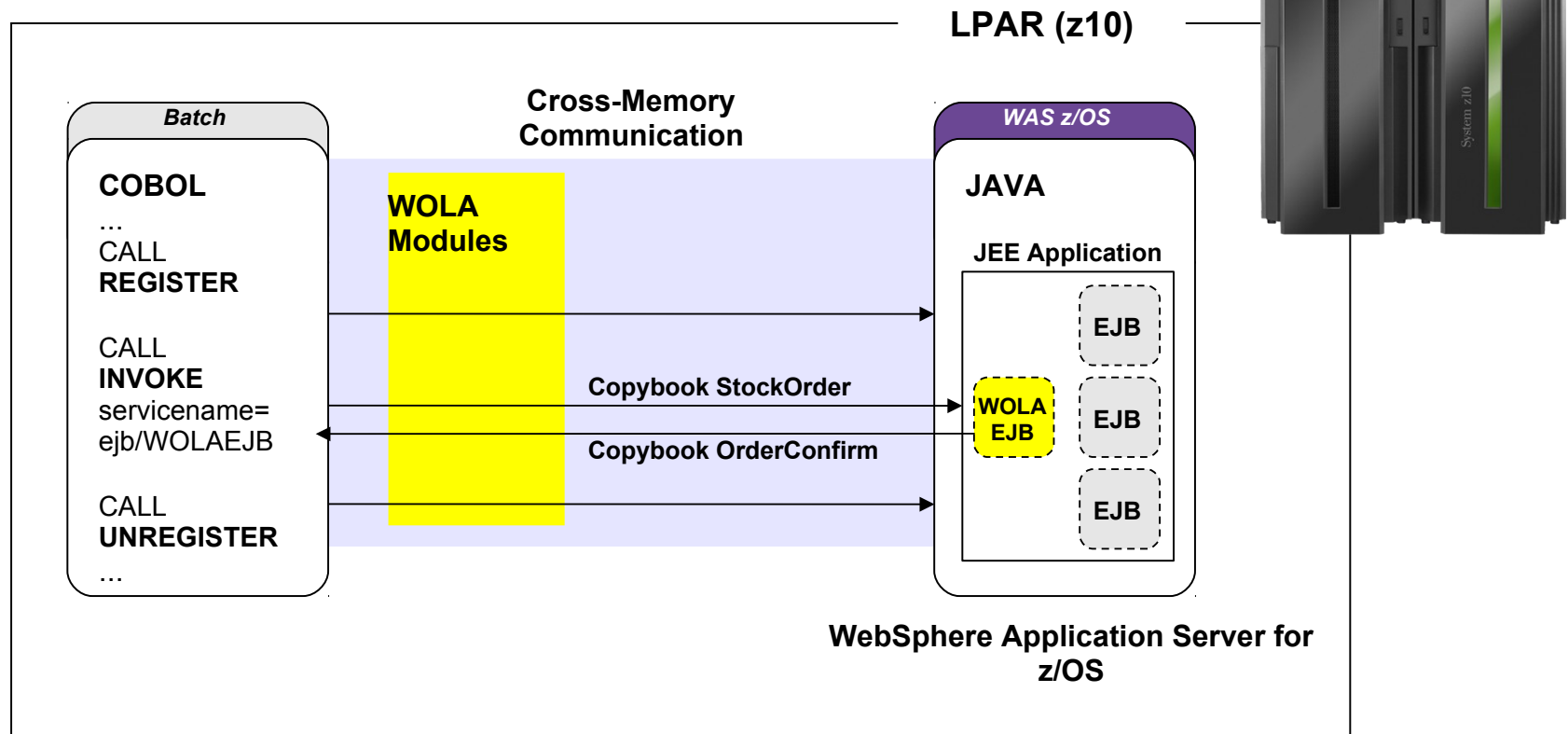
The Daemon owns the System LX which anchors the non-space-switching PC routines used to access WOLA services



**Essentially this is a set of APIs that externalizes the Local Comm function that is already in use within a WebSphere cell on an LPAR**

# Cobol / Java Inter Language Communication with WOLA

WOLA allows simple access to EJBs in WAS z/OS from COBOL (Batch or CICS) with the capability to transmit Cobol Copybooks.



```

01 StockOrder.
03 userID     PIC X(8).
03 stock      PIC X(5).
03 quantity   PIC 9(11).

```

```

01 OrderConfirm.
03 orderID    PIC X(8).
03 price      PIC 9(7)V9(2).
03 orderFee   PIC 9(7)V9(2).
03 ostatus    PIC X(12).
03 quantity   PIC 9(10)V9(1).
03 stock      PIC X(5).

```

# Sparda-Datenverarbeitung eG (SDV) supports growth and offers innovative banking solutions with Web portal software from IBM

## Business challenge:

Sparda-Datenverarbeitung eG (SDV) had been successfully running its Internet Home Banking (IHB) application for several years. Moving forward, the company wanted to capitalize on the most recent version of the underlying IBM WebSphere® Portal software to help deliver innovative capabilities and support an expected transaction growth rate of about 20 percent annually.

## Solution:

SDV upgraded to the most recent versions of IBM WebSphere software to support its IHB application, Web portal environment and Web 2.0 functionality. To increase storage capacity and help deliver more robust transaction capabilities, security and performance, SDV also installed an IBM DB2® data server and implemented IBM CICS® transaction software.

## Benefits:

- Improves IHB application performance and increases storage capacity
- Enables innovation such as new Web 2.0 user interfaces
- Creates a more efficient and security-rich approach for separating different banking customers in one physical server

*“With IBM WebSphere on z/OS we decided in favour of the platform with the lowest TCO for WebSphere.”*

*Sparda-Datenverarbeitung eG*

## Solution components:

- IBM WebSphere® Application Server for z/OS® Version 7.0
- IBM WebSphere Portal Enable Version 6.1
- IBM CICS® Transaction Server Version 3.2
- IBM CICS Transaction Gateway Version 6.0
- IBM DB2® for z/OS® Version 8.1

# Concept of “dedicated Batch” Windows Going Away

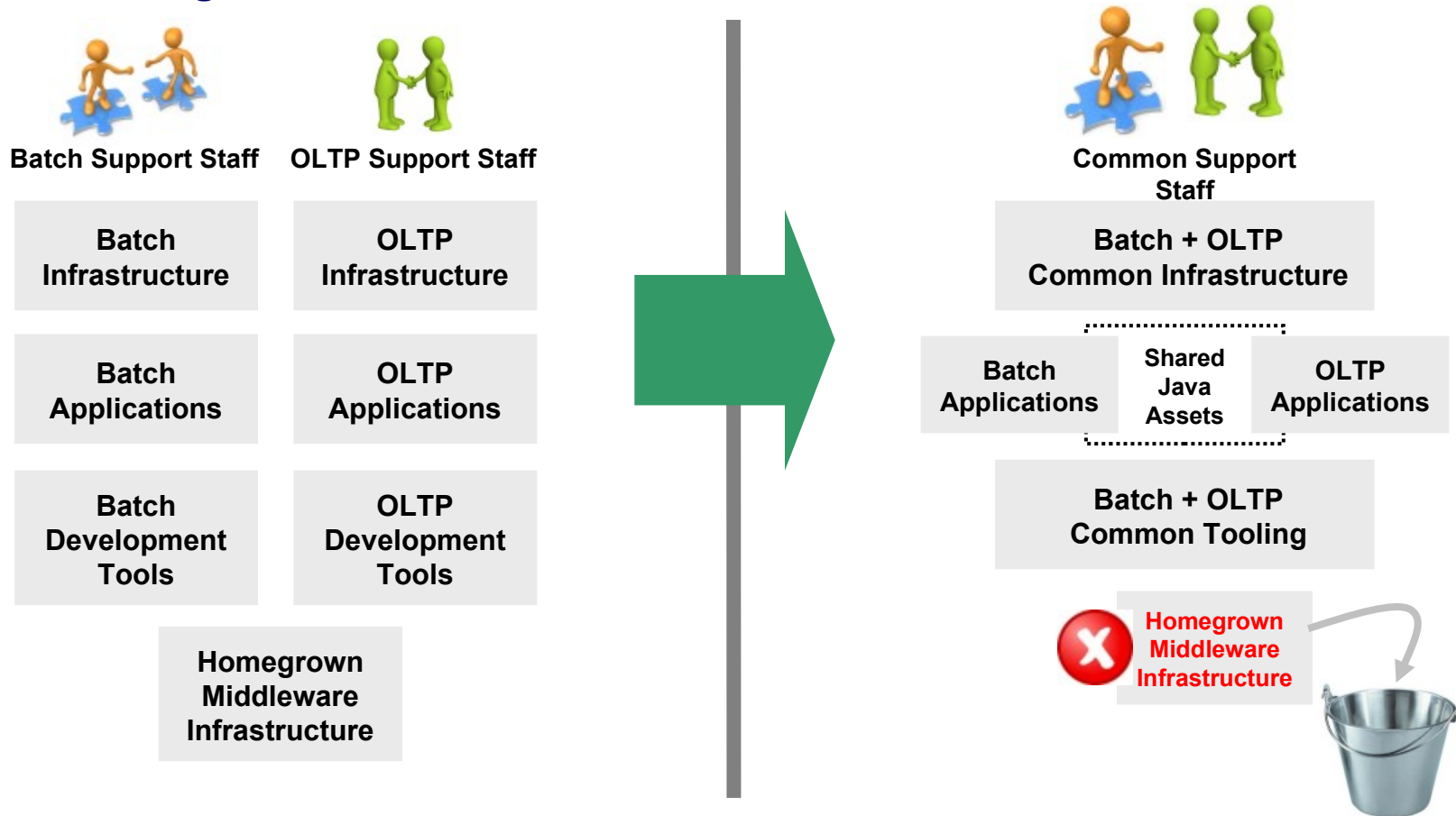
**Windows of time which used to be dedicated to batch processing are shrinking. The demands of online processing require more and more ...**



**The need to process batch work has *not* gone away.**  
**The need to perform the work concurrent with OLTP has emerged.**

## The value of shared services

It's not *just* that the window is shrinking ... it's also the cost pressures on maintaining the batch and OLTP environments:



**Efficiencies through consolidation around common assets**

# Java for Batch Processing?

**Yes ... for many very good reasons:**



## **Availability of Skills**

Java is a programming language with wide adoption in the industry. Skills for Java programming are common and affordable.

## **Tooling Support**

Development tooling for Java has advanced to the point where some tools (IBM Rational Application Developer) are very powerful and sophisticated.

*This also provides an opportunity to consolidate to a common tooling environment for both OLTP and batch development.*

## **z/OS Specialty Engines**

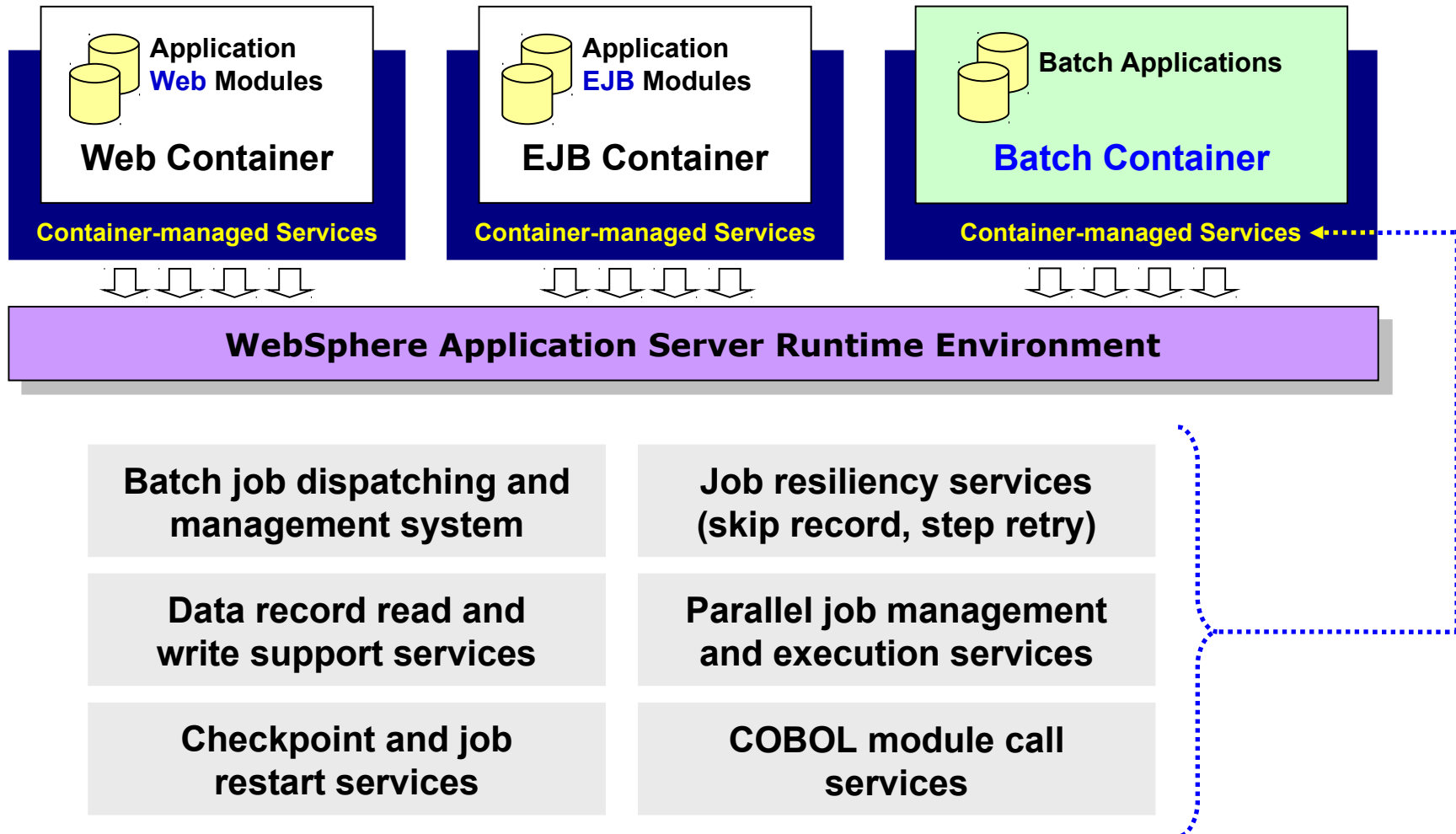
Pressures on cost containment often dictate greater use of z/OS specialty engines. Java offloads to zAAP. Java batch does as well.

## **Processing in OLTP Runtime**

Running Java batch in the same execution runtime as Java OLTP provides an opportunity to mix and manage the two processing types together under the same management model.

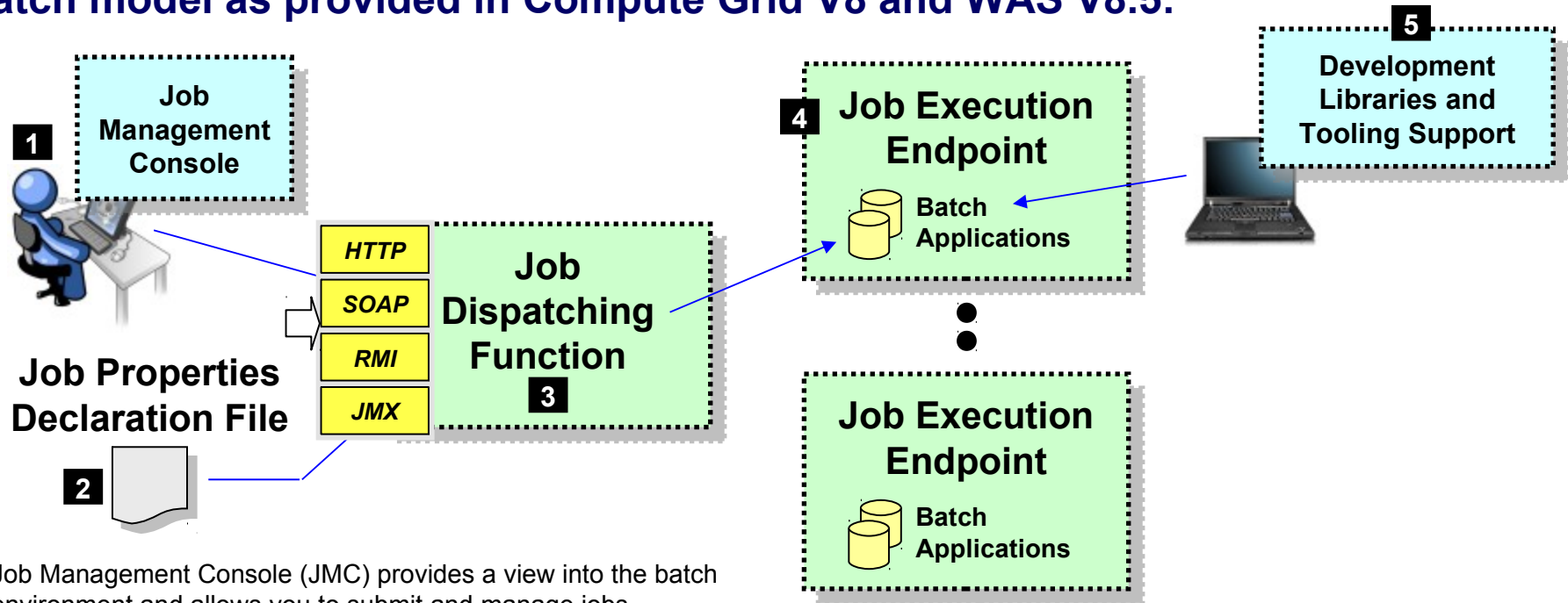
## Batch Container Added to the WAS Runtime

At a very high-level, you may think the IBM WebSphere Java Batch function as a "batch container" operating alongside the other containers of WAS itself:



# Overview of the Management and Execution Model

**This picture illustrates some of the key components of the WebSphere Java Batch model as provided in Compute Grid V8 and WAS V8.5:**



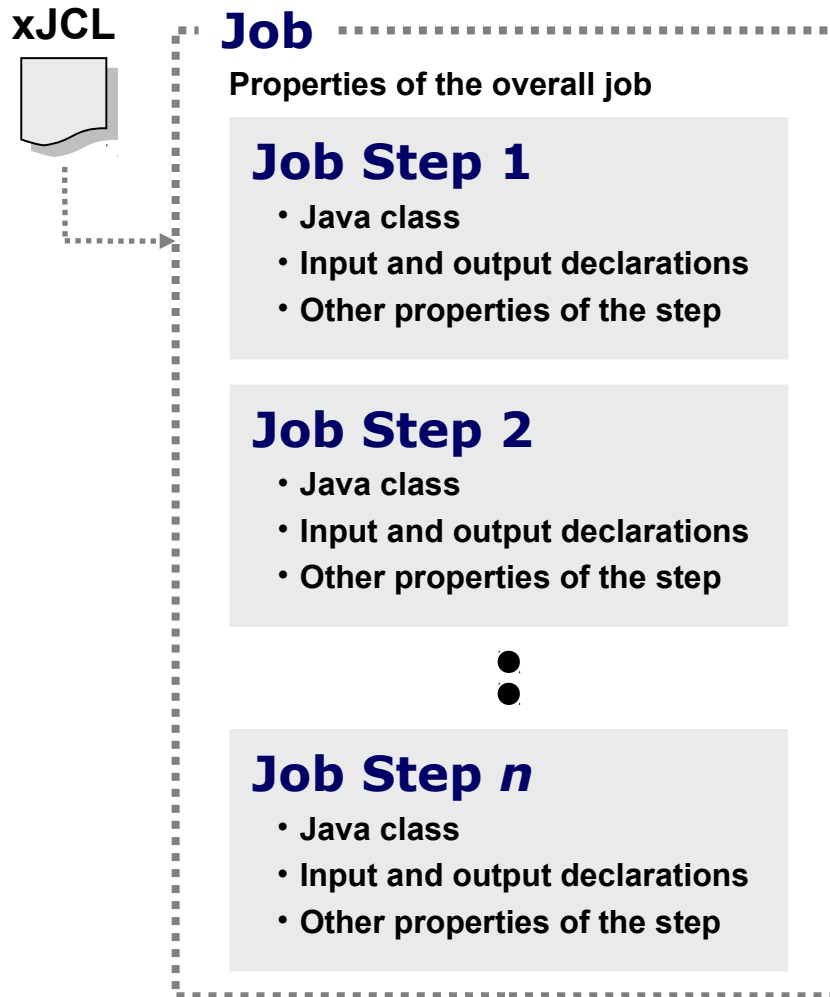
1. Job Management Console (JMC) provides a view into the batch environment and allows you to submit and manage jobs
2. Job declaration file (xJCL) provides information about the job to be run, such as the steps, the data input and output streams and the batch class files to invoke
3. The Job Dispatching function interprets the xJCL, dispatches the job to the endpoint where the batch application resides, and provides ability to stop and restart jobs
4. The Execution Endpoint is a WAS server in which the deployed batch applications run
5. The development libraries and tooling assist in the creation of the batch applications

**A comprehensive Java  
batch execution platform**  
Built on the proven Java runtime environment  
of WebSphere Application Server



## Batch Job and Batch Job Steps

**A batch job consists of one or more steps executed in order specified in xJCL:**



**The xJCL is submitted through the Job Management Console**

Interfaces provided: [HTTP browser](#), [command Line](#), [Web Services](#), [RMI](#)

**The Job Dispatching function interprets xJCL and determines which endpoint has batch application class files deployed**

**Dispatching Function invokes job and passes to the endpoint an object containing all the properties in xJCL**

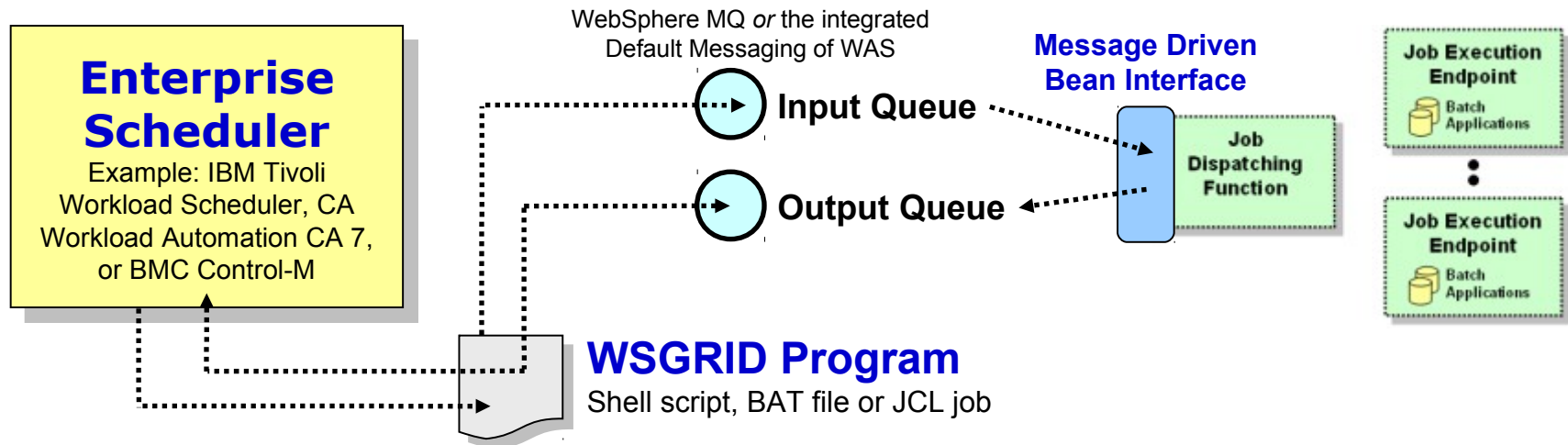
**Steps are executed in order, with conditional step processing if declared**

**Dispatching Function maintains awareness of job state**

**When job ends, job output file accessible through Job Management Console**

## Integration with Enterprise Scheduler Functions

The Job Dispatching Function has a Message Driven Bean (MDB) interface. IBM supplies a program that integrates schedulers with WebSphere Java Batch:



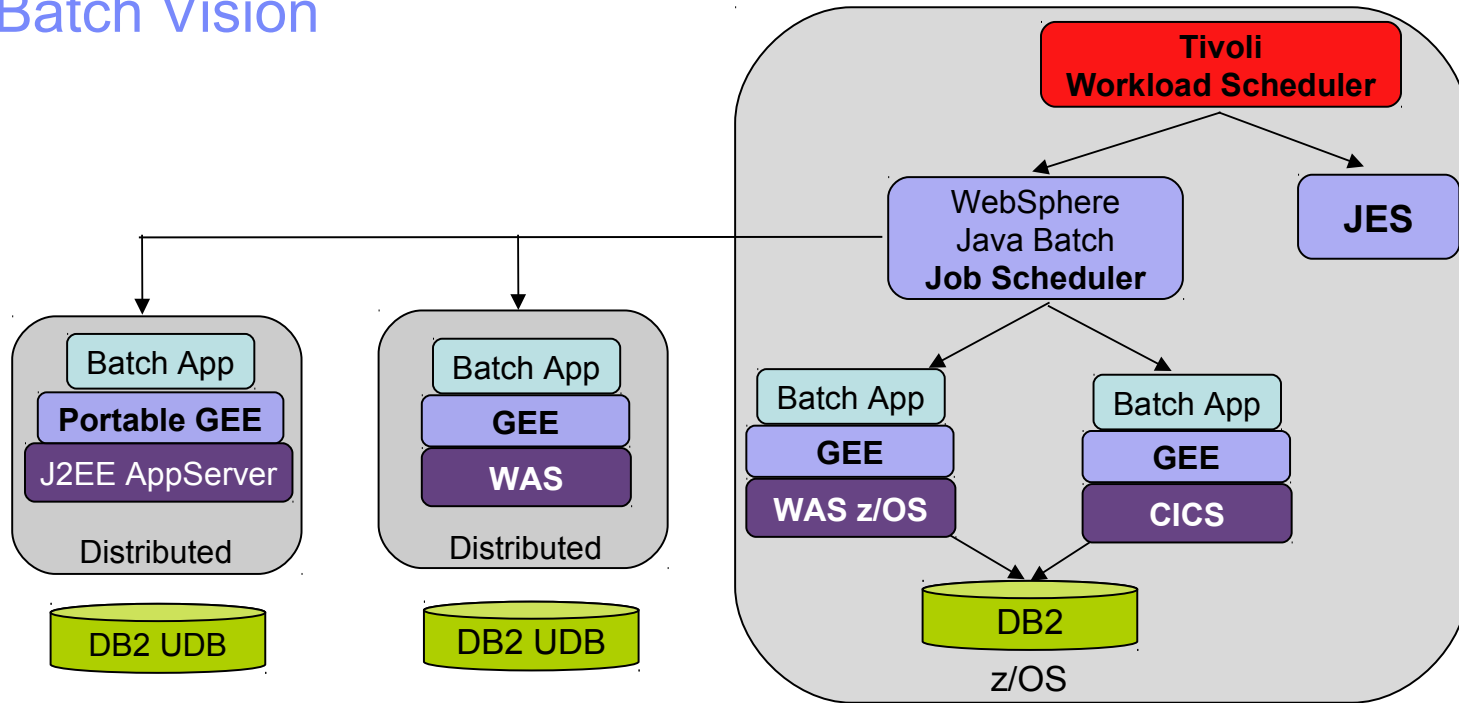
**WSGRID is seen by Scheduler as any other batch job it starts and monitors**

**WSGRID interacts with Job Dispatching, submitting the job and processing Java batch job output back to STDOUT or JES Spool if z/OS**

**WSGRID program stays up for life of job in WebSphere Java Batch**

**To the Scheduler, WGRID is the Java Batch job ... but behind WSGRID is all the WebSphere Java Batch function we'll discuss**

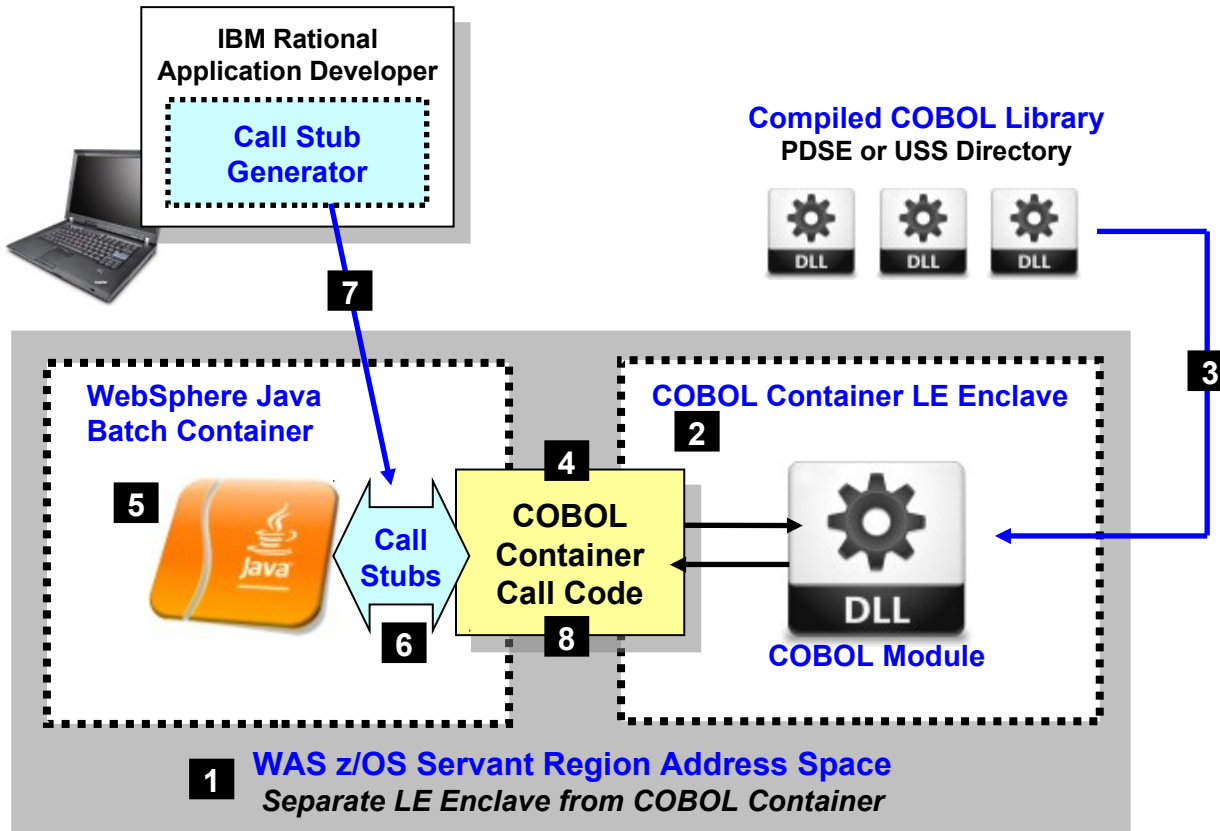
## The Batch Vision



- **Portable Batch applications** across platforms and J2EE vendors
- Location of the data dictates the placement of the batch application
- Flexible programming model, will host Spring Batch, JZOS, Compute Grid apps
- Centrally managed by your enterprise scheduler
- z/OS operational procedures manage batch across all platforms

## Cobol Container

**The COBOL Container provides a way to call and execute COBOL modules in the WAS z/OS server address space ... a very efficient way to call COBOL**

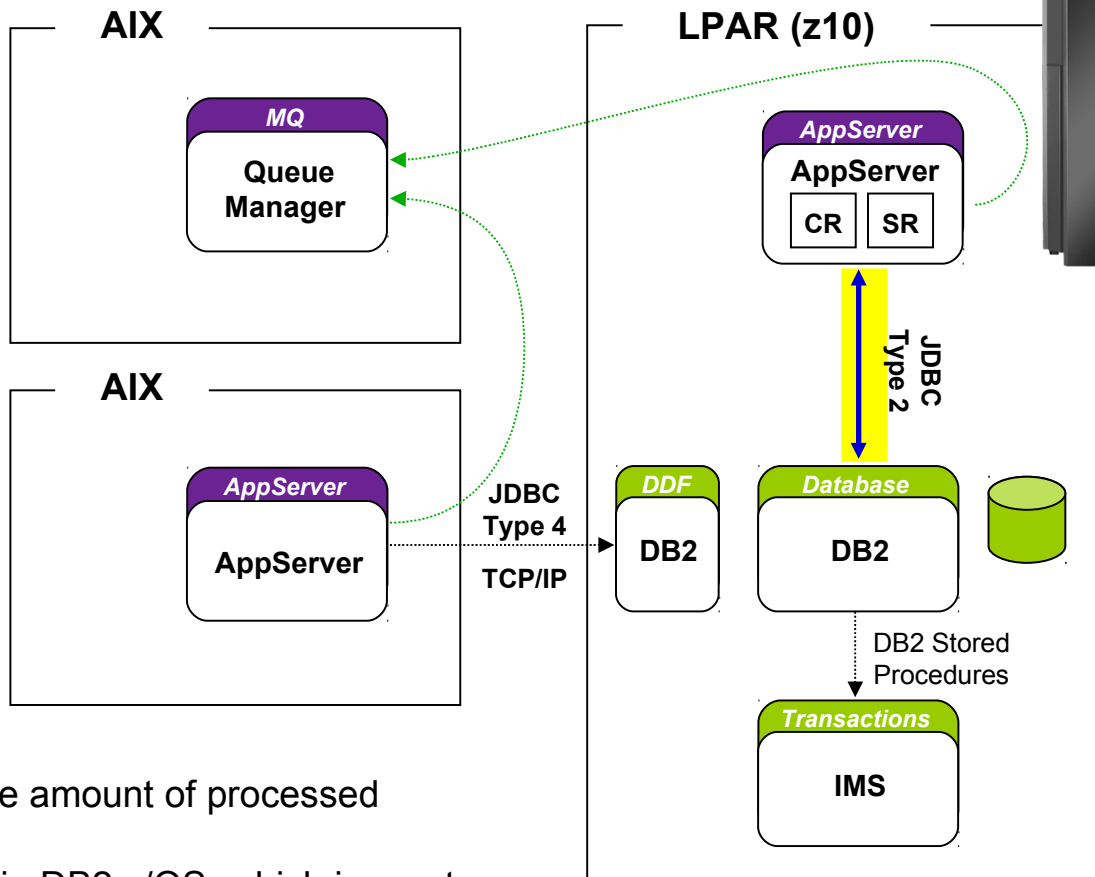


1. Batch application runs in the WAS z/OS servant region address space
2. The COBOL container is created as a separate LE enclave in the address space
3. COBOL DLLs are accessed using STEPLIB or LIBPATH
4. COBOL Container code provides the "glue" between the Java environment and the native COBOL
5. Java batch code uses supplied class methods to create the container and use it
6. Call stubs provide an easy way to call the COBOL DLL and marshal data back and forth
7. The call stubs are generated by a supplied utility that uses COBOL source to understand data bindings
8. JDBC Type 2 connections created in the Java batch program may be shared into the COBOL module in the COBOL Container

**Lines of code needed to invoke COBOL many times less than other means of calling COBOL from Java**

## WAS z/OS Java Batch PoC Architecture

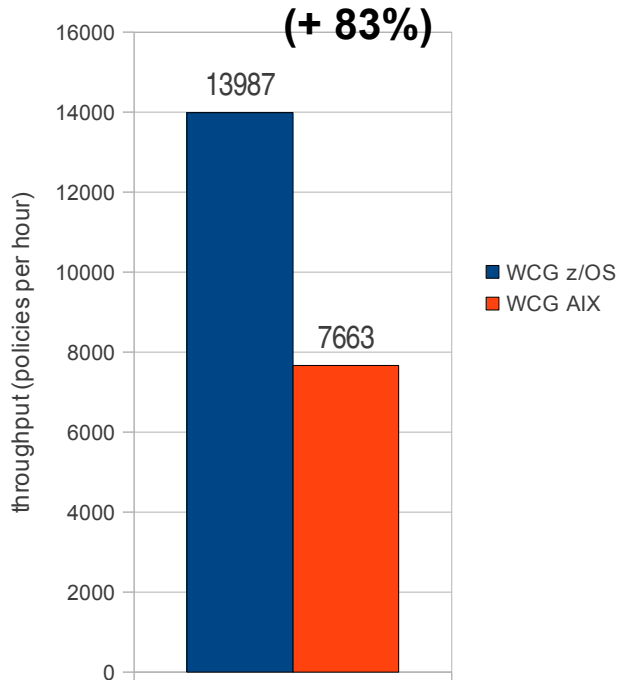
An insurance company has an existing WCG infrastructure on distributed AIX with data intensive JEE applications, which accesses DB2 z/OS via JDBC type 4. Within the scope of this PoC the Java batch job, which calculates the dynamic of the accident insurance is evaluated on WCG z/OS using JDBC type 2 cross memory adapter.



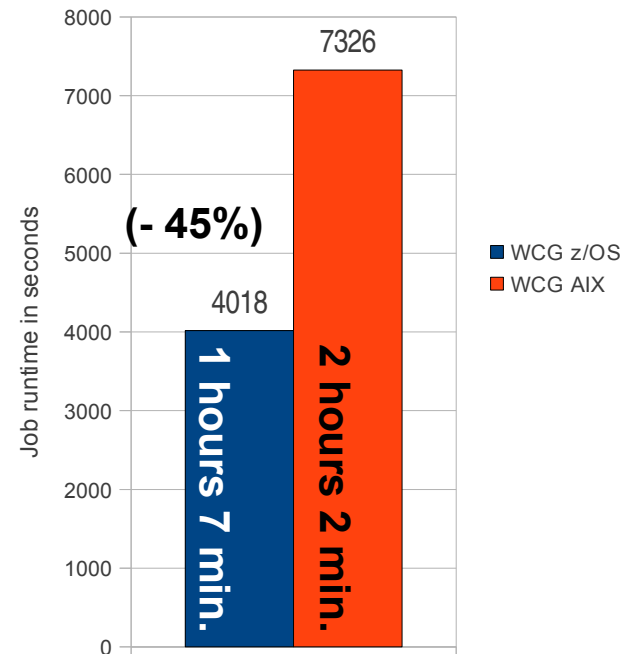
- ✓ Same job with same amount of processed insurance policies
- ✓ Same data basis in DB2 z/OS, which is reset after each run

## WAS z/OS Java Batch PoC Results

Throughput platform comparison  
based on 15592 policies



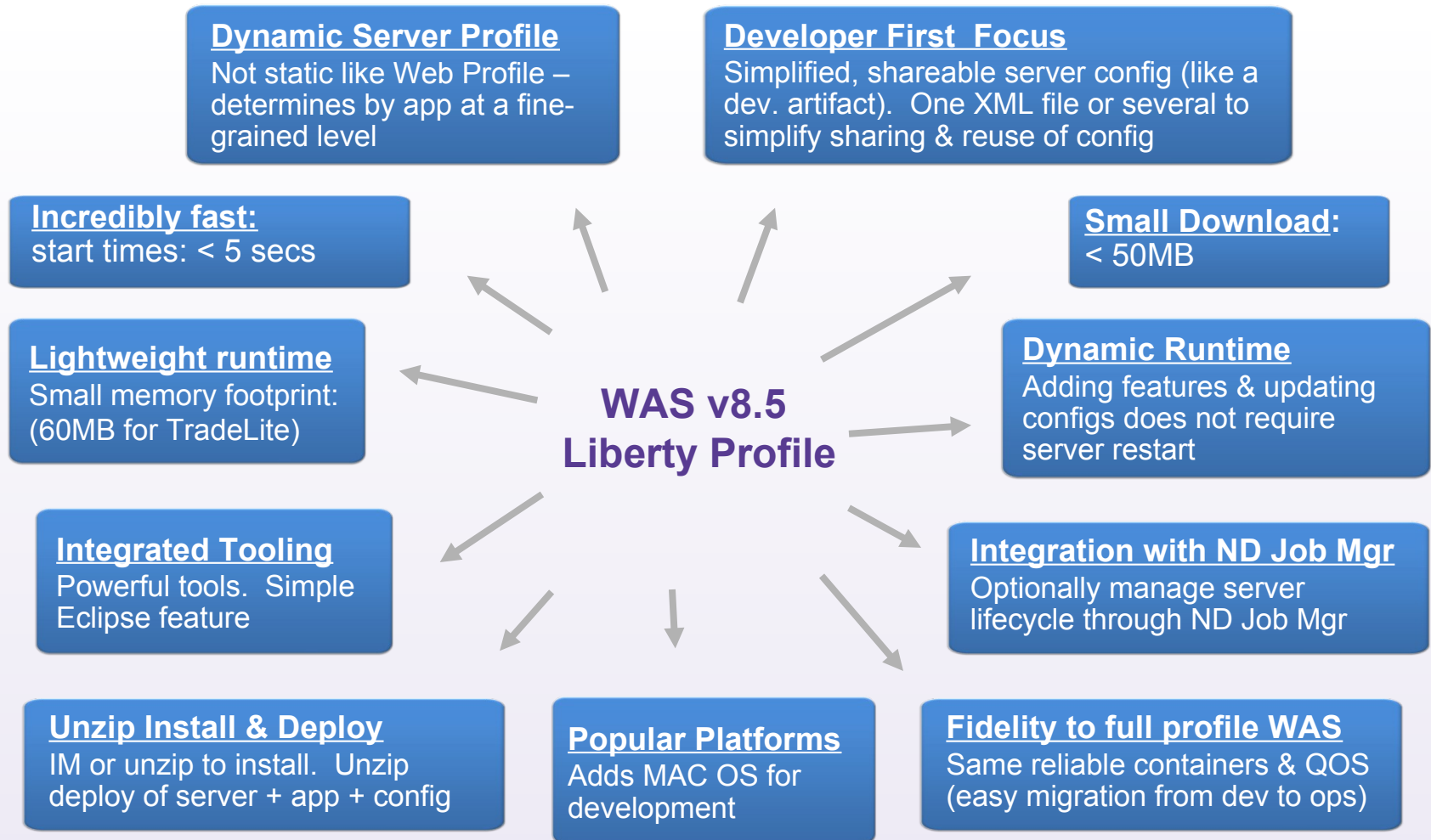
Job runtime platform comparison  
based on 15592 policies



The same job runs on both platforms WCG AIX and WCG z/OS. Because of the proximity to data in DB2 z/O the job, which runs on **WCG z/OS** has **83.5% more throughput** and the **job runtime is shortened by 45%**.

# WAS V8.5: Introduces the Lightweight Liberty Profile

*A highly composable, dynamic Server profile*



# What is the Liberty profile?

## *A lightweight, dynamic, composable runtime*

### **Lightweight**

- Server install is only about 55 MB
- Extremely fast server starts – typically well under 5 seconds

### **Dynamic**

- Available features are user selected and can change at runtime
- Restarts are not required for server configuration changes

### **Composable**

- Features are implemented as loosely coupled components with lazily resolved optional and mandatory dependencies
- The availability of features and components determines what Liberty *can* do and what's available to applications



## What is the Liberty Profile?

### ***An easy to configure runtime environment***

#### **Simple, extensible, and sparse configuration model**

- Configuration can live in a single XML document
- Configuration is by exception
  - Defaults are provided by contributing feature
  - Only modifications to the defaults are required

#### **Flexible configuration structure**

- Include mechanism allows for shared configuration elements
- Variable indirection mechanism allows for customization when distributed across multiple JVMs
- Easily managed by version control systems if desired

## What is the Liberty Profile?

### *A transportable runtime for your applications*

Use “**server package**” to generate an archive that contains a tested, self-contained, pre-configured server instance that includes your application

- Enables an application-centric deployment model that allows for easy scale-out
- Light-touch admin builds on the ND job manager infrastructure to manage Liberty server instances

### *A runtime environment with fidelity to full WAS*

- Liberty *is* WebSphere
- Applications that are developed and tested on Liberty will run on the full profile

## Liberty and traditional profile capabilities

*There are functional differences between traditional WAS and the Liberty profile – Liberty provides a useful subset of traditional WAS*

### Liberty Profile

Bean validation  
Blueprint  
Java API for RESTful Web Services  
Java Database Connectivity (JDBC)  
Java Naming and Directory Interface (JNDI)  
Java Persistence API (JPA)  
Java Server Faces (JSF)  
Java Server Pages (JSP)  
JMX  
Monitoring  
OSGi JPA  
Remote connector  
Secure Sockets Layer (SSL)  
Security  
Servlet  
Session Persistence  
Transaction  
Web application bundle (WAB)  
z/OS Security (SAF)  
z/OS Transactions (RRS)  
z/OS Workload Management

### Traditional WAS Profile

Everything Liberty has...



Enterprise Java Beans (EJBs)  
Messaging (JMS)  
Web Services  
Service Component Arch (SCA)  
Java Connector Architecture (JCA)  
Clustering  
WebSphere Optimized Local Adapters  
Administrative Console  
WSADMIN scripting  
**Multi-JVM Server Model**

*And much more ...*

## What This Means for Developers

- Support for Liberty Profile in Rational Application Developer
  - ▶ Enterprise development - advanced programming, cloud, collaboration, and quality tools
  - ▶ Available standalone or bundled in WAS - Tools Edition and WAS ND - Tools Edition
- Support for Liberty Profile in WAS Developer Tools for Eclipse (WDT)
  - ▶ Subset of RAD focused on core programming models
  - ▶ Simple Eclipse feature update for WTP 3.6 (Helios) and 3.7 (Indigo)
  - ▶ Available unsupported at no charge, or supported for a fee through WAS for Developers – Tools Edition for Eclipse
- Lightweight WAS Liberty profile runtime for dev/test
  - ▶ Focused on the development and operations experience
  - ▶ Small footprint test server runtime
  - ▶ Simplified, shareable config
  - ▶ Fastest possible server start-up
  - ▶ Fidelity with full-profile WAS editions
  - ▶ Free for developer ! No Expiration
  - ▶ Initially focused on web apps (including JPA, transactions, security...)
- Accelerate development time to value
  - ▶ Develop/test with RAD or WDT and WAS v8.5 Liberty server type
  - ▶ Deploy applications unchanged to full profile WAS for production

## Why Liberty on z/OS?

### *Simplification*

- Liberty environments don't need significant z/OS configuration and customization
  - RRS, WLM, and SAF exploitation and configuration is optional
  - No authorized code is **required** to host applications
- Liberty runs in a single process instead of 3+ started tasks
  - Significantly reduced resource consumption
  - No started task definitions are **required**
  - No need to create users and groups for controllers, servants
- Server instances can be quickly created or cloned

```
server create serverName [options]
server package serverName [options]
```

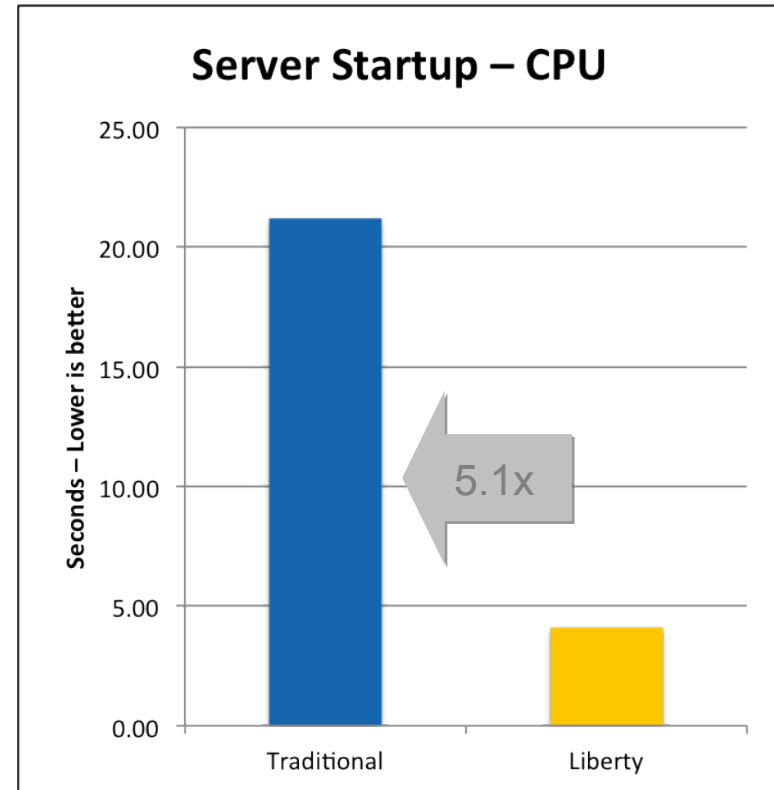
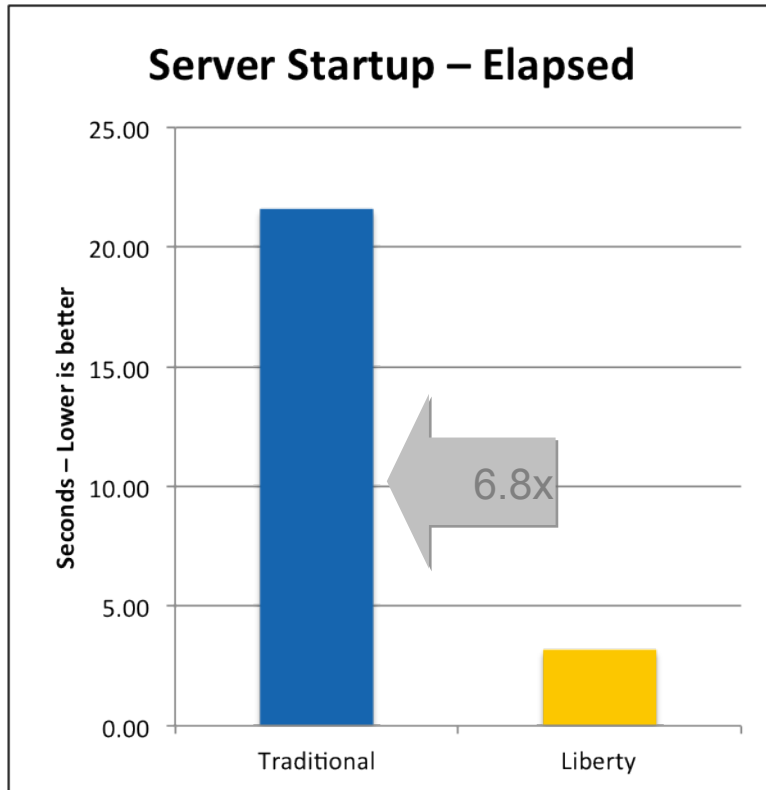
## Why Liberty on z/OS?

### ***Application portability and stack consistency***

- Liberty behaves *exactly the same* on all platforms out of the box
  - z/OS specific behaviour must be configured if desired
- Administration is the same for all platforms out of the box
  - Server operations are controlled by the same server script
  - Logs, trace, and configuration live in the hierarchical file system and are tagged with the appropriate code page for easy viewing and editing
  - Existing server configurations can be brought to z/OS from distributed without modification
- An extremely light-weight, single process runtime
  - Removes deployment and runtime complications introduced by the split process, multi-JVM runtime of traditional WAS for z/OS

## Why Liberty on z/OS?

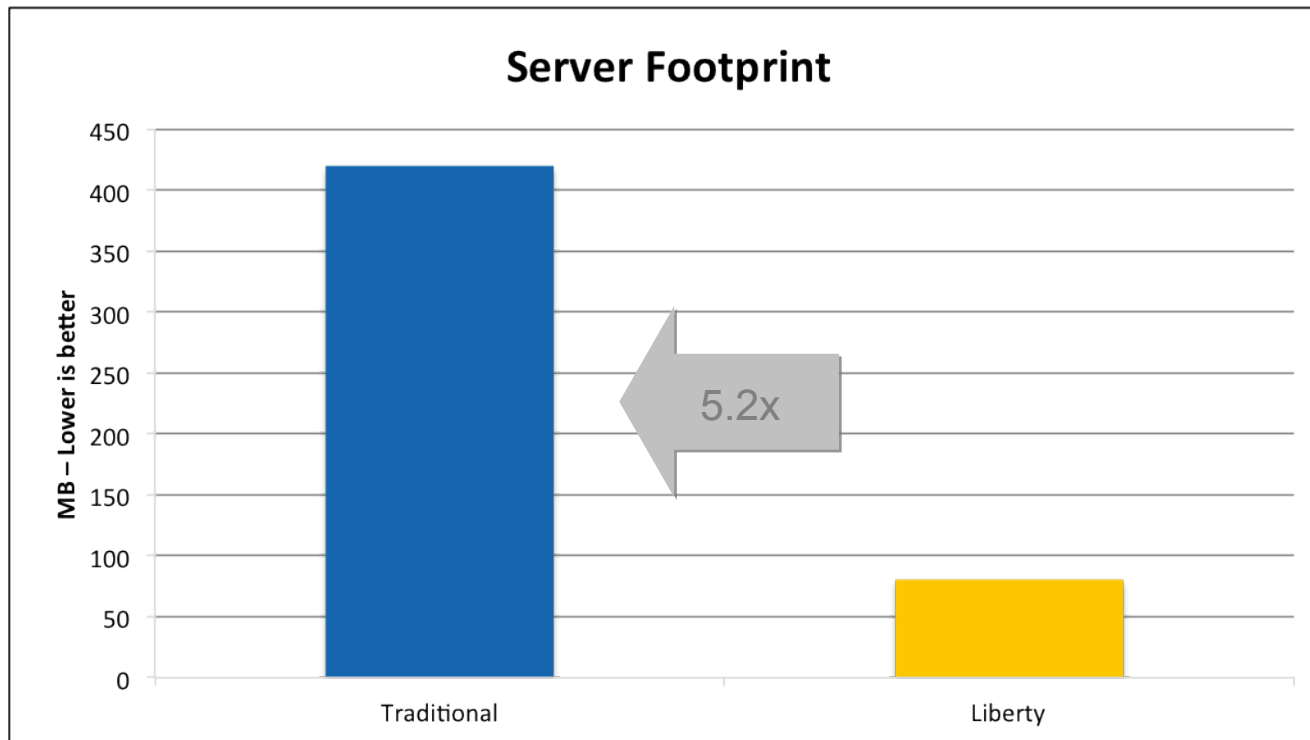
**Performance: Startup time – 3.2 seconds!**



- Liberty – 64bit IBM Java 6.0.1, 64/64MB min/max heap, 60MB shared class cache, TradeLite installed
- Traditional – 64bit IBM Java 6.0.1, 1SR, 128/256MB min/max CR heap, 256/512MB min/max SR heap, 75MB CR shared class cache, 75MB SR shared class cache, no applications installed

## Why Liberty on z/OS?

### ***Performance: Memory footprint – 80% reduction***

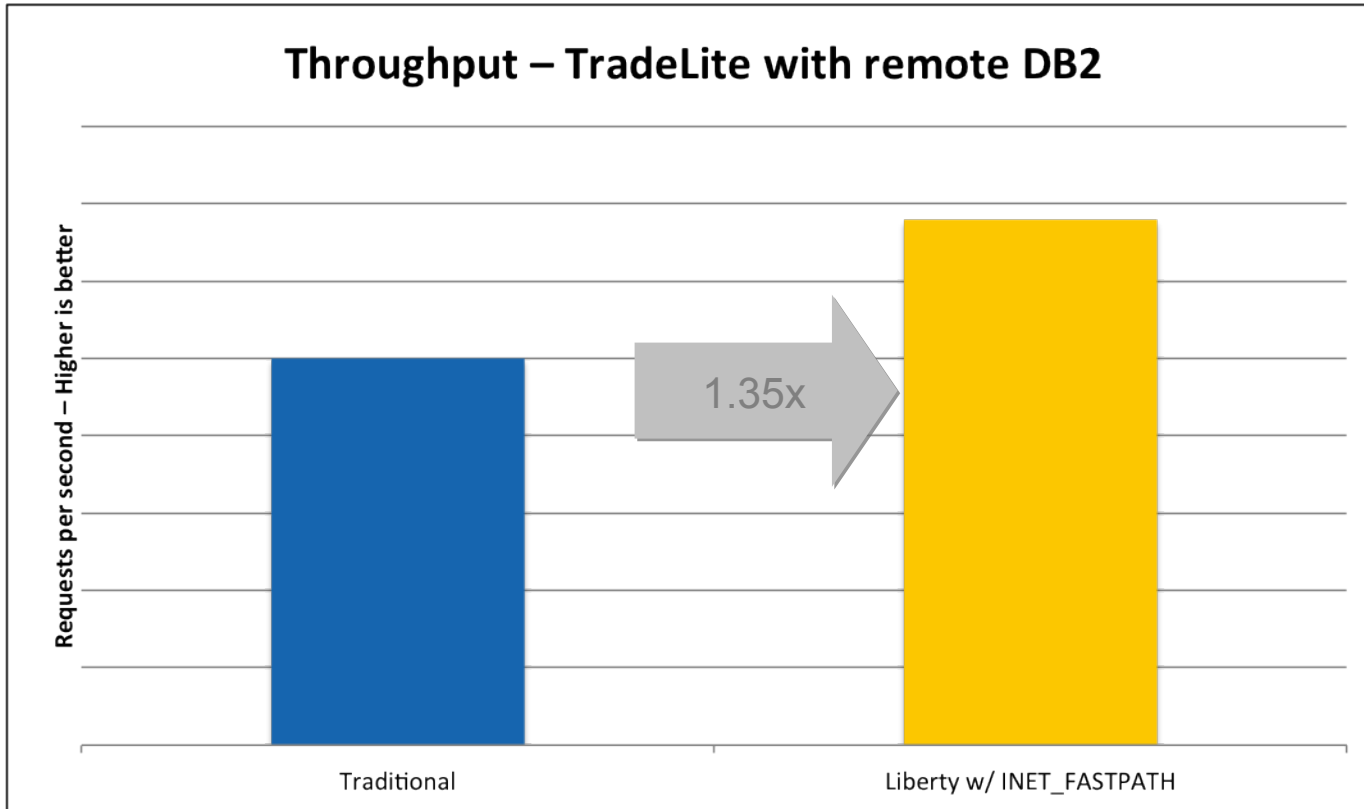


- Liberty – 64bit IBM Java 6.0.1, 64/64MB min/max heap, 60MB shared class cache, TradeLite installed
- Traditional – 64bit IBM Java 6.0.1, 1SR, 128/256MB min/max CR heap, 256/512MB min/max SR heap, 75MB CR shared class cache, 75MB SR shared class cache, no applications installed



## Why Liberty on z/OS?

### *Performance: Throughput – Up to 35% improvement*



- z196, 2-way LPAR running z/OS 1.13
- 64bit IBM Java 6.0.1 with compressed references, 1M large pages, 2GB heap
- IBM DB2 for z/OS v10, JDBC T4 with keepDynamic
- `_BXK_INET_FASTPATH=*` set to enable CommServer “fast path” for Liberty

## Liberty Profile – Startup & Footprint

### ■ The problem of a lightweight development environment in WebSphere has been solved!

- Liberty Profile startup & footprint are on par with Tomcat.
- Liberty Profile starts up in less than half the time of JBoss Web profile.

#### System Info:

Lenovo T60p - 2 x 2.16 GHz Intel Core Duo T2600  
2GB RAM, Windows XP 32-bit

Apache Tomcat 7.0.12

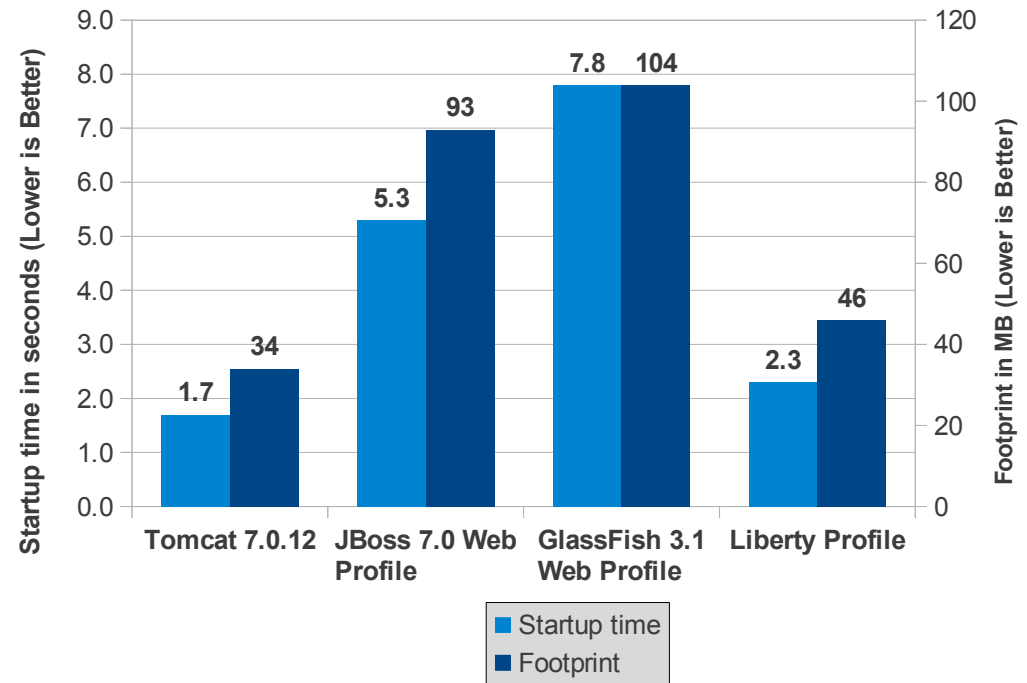
JBoss Community Edition 7.0 Web Profile server

GlassFish Server 3.1 Open Source Edition Web Profile

WAS V8.Next Liberty Profile

(All servers had the TradeLite benchmark application installed)

### Startup & Footprint Comparison of various lightweight servers



*Note: Tomcat , JBoss, and GlassFish were measured with the HotSpot JDK, while Liberty was measured with the IBM JDK.*

## Liberty Profile - Throughput

- **A lightweight server that can service requests with the speed of a full production server!**
  - Liberty Profile provides up to 20% better runtime performance than JBoss and 25% better than Tomcat.

### System Info:

IBM x3550 – 4 x 1.86 GHz Intel Xeon E5320, 8 GB RAM  
RedHat Linux 5.3 32-bit

Apache Tomcat 7.0.12

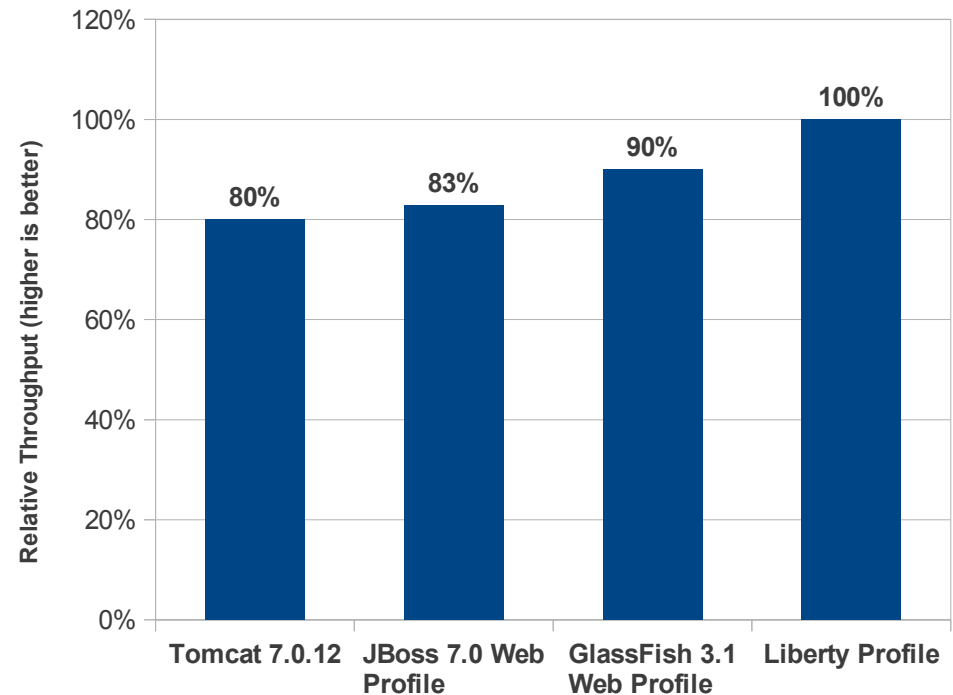
JBoss Community Edition 7.0 Web Profile server

GlassFish Server 3.1 Open Source Edition Web Profile

WAS V8.Next Liberty Profile

(All servers had the TradeLite benchmark application installed)

Throughput Comparison of various lightweight servers



*Note: Tomcat , JBoss, and GlassFish were measured with the HotSpot JDK, while Liberty was measured with the IBM JDK.*

## Liberty for z/OS – Key Use Cases

### ***Accelerate application development and deployment while leveraging z/OS qualities of service***

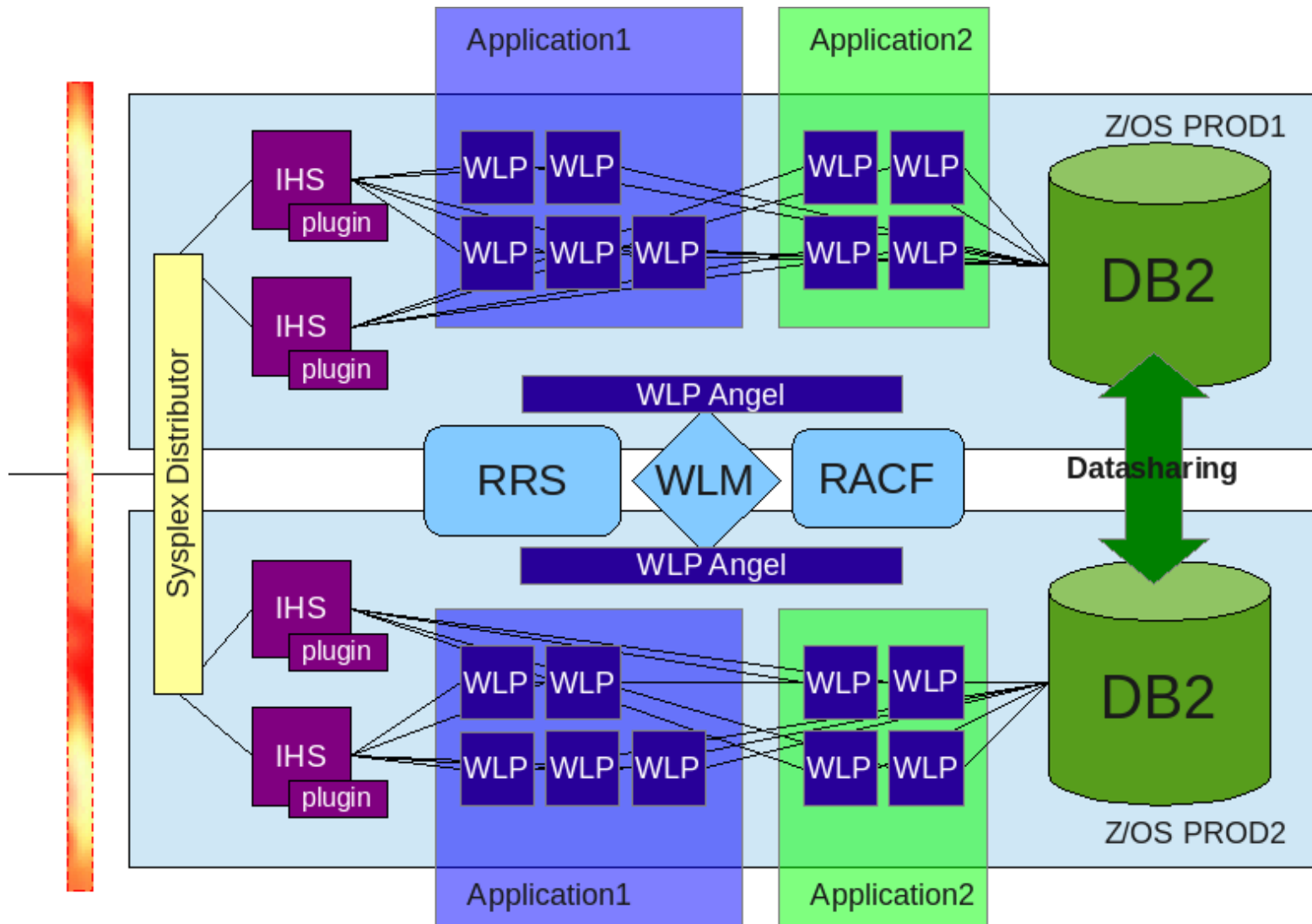
**Lightweight Production** – Where a lightweight application server is appropriate for production web applications, leverage the rapid startup and small footprint of Liberty profile based applications

**Test Web Applications using z/OS Resources** – Easily perform z/OS platform testing of web applications regardless of development platform

**Incremental adoption of unique z/OS extensions** – Enable incremental exploitation of optional z/OS extensions to leverage z/OS qualities of service

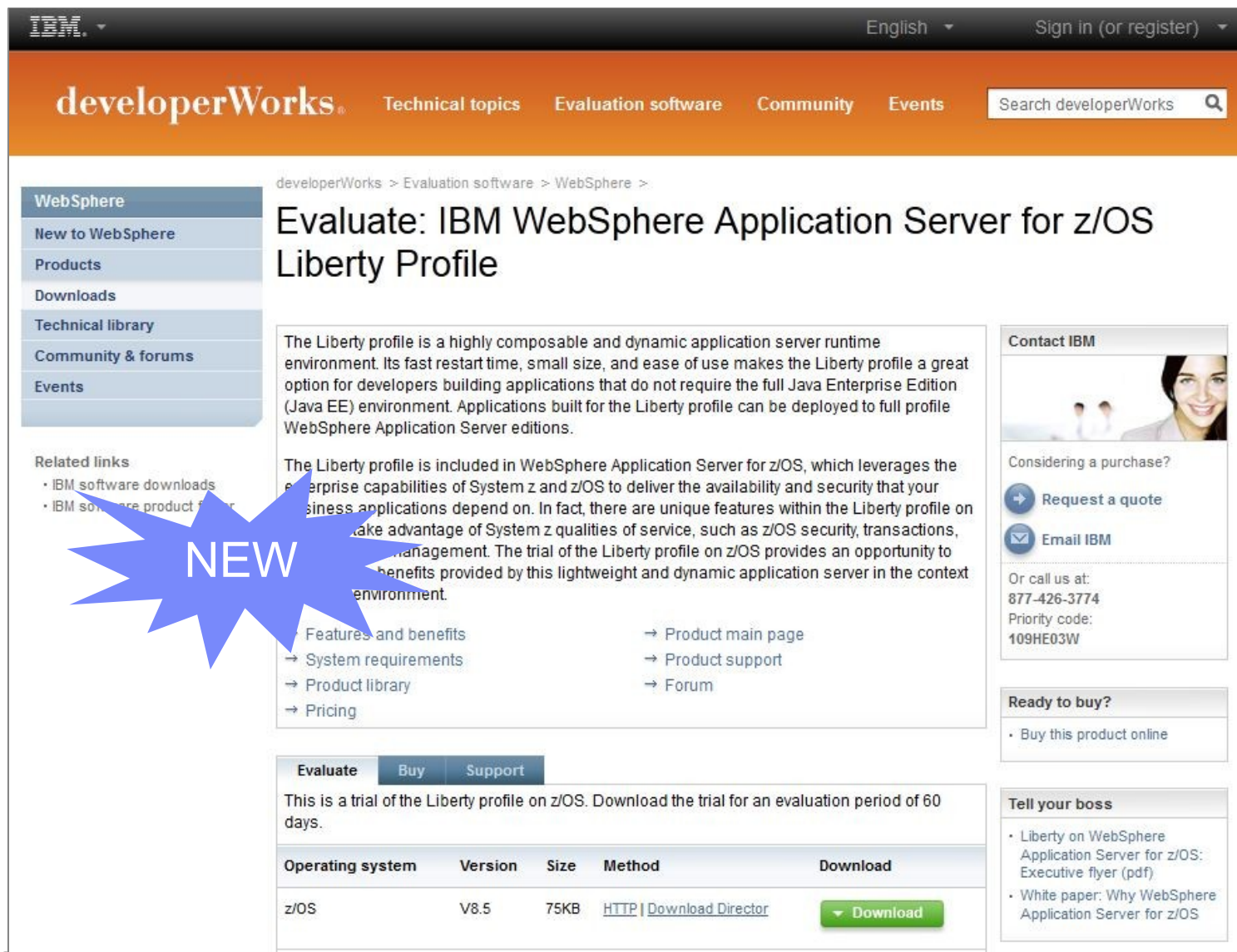
**Efficient packaging and deployment of applications** – Create and deploy Liberty profile applications as packages that include both the application and configuration

# Liberty Production Architecture on z/OS



# Announcing a new Liberty Profile on z/OS Trial!

<http://www.ibm.com/developerworks/downloads/ws/waszosliberty/>



IBM. English Sign in (or register)

developerWorks® Technical topics Evaluation software Community Events Search developerWorks

WebSphere

- New to WebSphere
- Products
- Downloads
- Technical library
- Community & forums
- Events

Related links

- IBM software downloads
- IBM software product finder

developerWorks > Evaluation software > WebSphere >

## Evaluate: IBM WebSphere Application Server for z/OS Liberty Profile

The Liberty profile is a highly composable and dynamic application server runtime environment. Its fast restart time, small size, and ease of use makes the Liberty profile a great option for developers building applications that do not require the full Java Enterprise Edition (Java EE) environment. Applications built for the Liberty profile can be deployed to full profile WebSphere Application Server editions.

The Liberty profile is included in WebSphere Application Server for z/OS, which leverages the enterprise capabilities of System z and z/OS to deliver the availability and security that your business applications depend on. In fact, there are unique features within the Liberty profile on z/OS that take advantage of System z qualities of service, such as z/OS security, transactions, and resource management. The trial of the Liberty profile on z/OS provides an opportunity to experience the benefits provided by this lightweight and dynamic application server in the context of a z/OS environment.


- Features and benefits
- System requirements
- Product library
- Pricing
- Product main page
- Product support
- Forum

**Evaluate Buy Support**

This is a trial of the Liberty profile on z/OS. Download the trial for an evaluation period of 60 days.

Operating system	Version	Size	Method	Download
z/OS	V8.5	75KB	<a href="#">HTTP   Download Director</a>	<a href="#">Download</a>

**Contact IBM**



Considering a purchase?

- Request a quote
- Email IBM

Or call us at:  
877-426-3774  
Priority code:  
109HE03W

**Ready to buy?**

- Buy this product online

**Tell your boss**

- Liberty on WebSphere Application Server for z/OS: Executive flyer (pdf)
- White paper: Why WebSphere Application Server for z/OS



***Ich mag den Mainframe, weil wir ein gemeinsames Talent haben:  
Wir können viele verschiedene Dinge erledigen – gleichzeitig!***

*Isabel Arnold, Technical Sales IBM System z*



# WebSphere z/OS Links Collection

Topic	Link
<b>Guide to WebSphere on z/OS Collateral</b> Updated master list of links to collateral	<a href="http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP102205">http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP102205</a>
<b>WebSphere Java Batch</b> Overview and z/OS Specifics Presentation, whitepaper, videos	<a href="http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP101783">http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP101783</a>
<b>Why WAS for z/OS</b> Executive Brochure History of release enhancements Technical Presentation	<a href="http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP101532">http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP101532</a>
<b>WAS for z/OS Liberty Profile</b> Executive Brochure Quick Start Guide and Samples	<a href="http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP102110">http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP102110</a>
<b>WebSphere Optimized Local Adapters (WOLA)</b> Overview, whitepapers, videos History of WOLA updates	<a href="http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP101490">http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP101490</a> <a href="https://www.ibm.com/developerworks/mydeveloperworks/blogs/wasdev">https://www.ibm.com/developerworks/mydeveloperworks/blogs/wasdev</a>
<b>Training – z/OS Wildfire Workshops</b> WAS for z/OS v8.5, WebSphere Compute Grid	<a href="http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/PRS1778">http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/PRS1778</a>
<b>WebSphere on z Virtual User Group</b> Download Sept webcast – WAS 8.5 / Liberty Register for Dec webcast – WebSphere Batch	<a href="http://www.websphereusergroup.org/zos">http://www.websphereusergroup.org/zos</a>



## WLP 8.5 Programming Models (as of Nov 2012)

The Liberty profile supports a subset of the Java EE 6 stack. The following list note the supported technologies:

Java Servlet 3.0

JavaServer Faces (JSF) 2.0

JavaServer Pages (JSP) 2.2

Java Expression Language 2.2

Standard Tag Library for JavaServer Pages (JSTL) 1.2

Bean Validation 1.0

Java Persistence API (JPA) 2.0

Java Transaction API (JTA) 1.1

Java API for RESTful Web Services (JAX-RS) 1.1

The Liberty profile also supports OSGi applications. The following list shows supported technologies for OSGi applications (with a reference to the specification where appropriate):

Web Application Bundles (OSGi R4.2 Enterprise, Chapter 128)

Blueprint Container (OSGi R4.2 Enterprise, Chapter 121)

– Blueprint Transactions

– Blueprint Managed JPA

JNDI (OSGi R4.2 Enterprise, Chapter 126)

OSGi application of Java EE technologies supported by the profile. A complete list of technologies supported by the Liberty profile can be found at the [information center website](#)..

# Lightweight Configuration of WLP: server.xml

```
<server description="tradeLiteServer">
  <featureManager>
    <feature>jsp-2.2</feature>
    <feature>jdbc-4.0</feature>
  </featureManager>
```

*Features* control what's available in the runtime

```
<logging consoleLogLevel="INFO" />
```

*Singleton* configurations specify properties for runtime services for which there is only one instance

```
<application type="war"
  id="tradelite"
  name="tradelite"
  location="${shared.app.dir}/webcontainer/tradelite.war" />
```

*Instance* configurations allow multiple instances of resources and applications to be declared

```
<include location="jdbc-drivers.xml" />
<include location="${user.home}/custom.xml" optional="true" />
```

*Includes* can be used to implement an extensible configuration model

```
<dataSource id="jdbc/DerbyTradeDataSource"
  jndiName="jdbc/TradeDataSource"
  jdbcDriverRef="DerbyEmbedded">
  <properties databaseName="${shared.resource.dir}/data/tradedb" />
</dataSource>
</server>
```

*References* can be used in multiple elements to point share a common definition

## Composability – Based on *features*

```
<server description="composabilityIsKey">  
  <featureManager>  
    <feature>appSecurity-1.0</feature>  
    <feature>jsp-2.2</feature>  
    <feature>restConnector-1.0</feature>  
    <feature>jpa-2.0</feature>  
  </featureManager>  
</server>
```

