#SHAREorg

Getting Started with WebSphere Batch *(a.k.a. Compute Grid)* on WAS V8.5 for z/OS

John Hutchinson
IBM

August, 2013, Boston
Session Number 13598

SHARE in Boston

WebSphere Application Server on z/OS Sessions in Boston

Day	Time	Room	#	Title	Speaker
Monday	**9:30**	**203**	**13597**	**Getting Started with WebSphere Liberty Profile on z/OS**	**David Follis**
Monday	**4:30**	**203**	**13600**	**Managing Server Output from WAS on z/OS**	**Mike Loos**
Tuesday	**9:30**	**203**	**13644**	**Using WAS Optimized Local Adapters (WOLA) to migrate your COBOL to zAAP-able Java**	**Jim Mulvey**
Tuesday	**11:00**	**203**	**13640**	**Need A Support Assistant? Check Out IBM's! (ISA)**	**Mike Stephen**
Tuesday	**1:30**	**207**	**13953**	**What Would Life Be Like If You Ran Your Internet Applications On z/OS?**	**Ed McCarthy**
Tuesday	**3:00**	**203**	**13641**	**zWAS: In Real Life**	**Rod Feak**
Wednesday	**1:30**	**202**	**13601**	**Lab: WebSphere Liberty Profile on z/OS**	**everybody**
Thursday	**11:00**	**203**	**13598**	**Getting Started with Compute Grid (Batch)**	**John Hutchinson**
Thursday	**3:00**	**203**	**13645**	**Configuring Security for Liberty**	**Mike Loos**

SHARE in Boston

Do you have WebSphere AppServer V8.5 on z/OS?

. . . . Then you have WebSphere Batch ! *("Compute Grid")*

Get started with some basic Batch applications!

- **Java Batch options**
  - JZOS, WAS Feature Packs, z/OS Batch Container, or . . .
  - WebSphere Batch ("Compute Grid") built into WAS V 8.5
- **Important Features in WebSphere Batch**
  - Integration with Schedulers, CICS, COBOL, PJM, WLM, SMF
- **Development Tools**
  - Batch Framework & Supporting Classes
- **Choosing your 1st Application** ("Proof Of Concept")

  a) Develop New Batch Application in Java Batch

  b) Reuse Existing Java Main Batch Applications

  c) Transform traditional JES batch jobs into WAS batch.

Several Different Approaches

Standalone Java Program

- Simple programming model
- Launch with "java" command at the shell
- Programmer responsible for everything

JVM Launcher Solutions

- JZOS, BPXBATCH
- JVM Launcher in a JES Batch Job
- Programmer must code functions not provided by Launcher

z/OS Batch Execution Runtime

- z/OS 1.13 & V2.1 Batch Container
- Batch Container provides useful functions
- Programmer must code functions not provided by Container

WebSphere Batch Container

- WebSphere Compute Grid or WebSphere 8.5
- Provides batch programming function as services of the platform
- Allow programmer to focus on the business logic, not "middleware" functions

All are perfectly good approaches, depending on the nature of your batch processing needs.

Why run Batch with Java on Z?

Extend Development Skill Sets & Programming Resources

- Java programming skills are more prevalent.
- Leverage your OLTP infrastructure.
- Modern development tools increase application agility.

Integration with other Technology Solutions

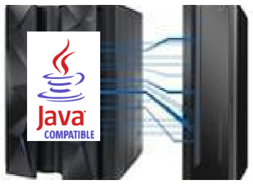
- Extend Enterprise Scheduling with WebSphere Batch
- Java rules execution engines (Ilog, WODM, Drools)

Leverage Specialty Engine Processors on System Z

- Offload Java work to System Z Application Assist Processors (zAAP)
- Lower the overall cost-profile of running batch on the mainframe.

Compress your Batch Window

- Run batch during Online



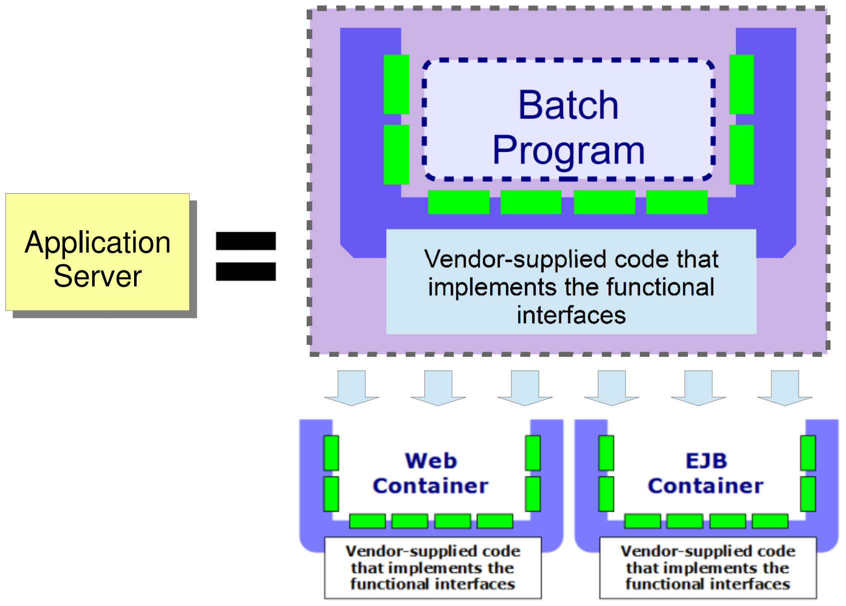
Why Use *WebSphere* Batch?

Benefits beyond a J2SE-based batch implementation:

- ***Long-running re-usable JVM*** to spread the cost of initialization and tear-down and reap the benefits of JIT.
- Built on Java EE platform security, logging, integration, high availability, resource management.
- Programming model for batch in a structured, re-usable, rich manner.
- Application support functions specific to batch processing:
  - Checkpoint / Restart
  - Service Classification & Workload throttling
  - Log management and aggregation
  - Job Parallelism & Distribution across LPARs
  - Scheduler integration (TWS, Ctrl_M, Zeke, etc)

WebSphere Batch Runtime Container

The IBM Batch Container is an extension to the existing container structure of WebSphere Application Server

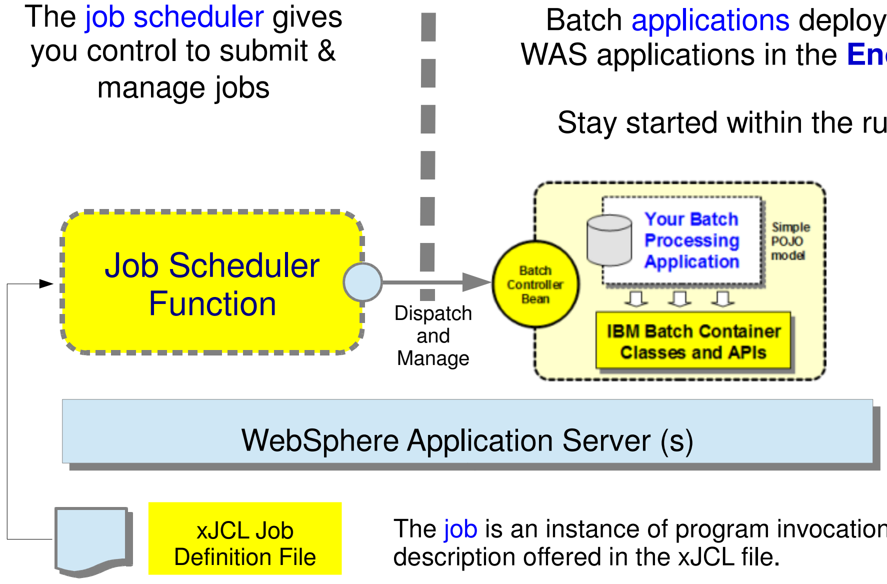

Application Server
Batch Program
Vendor-supplied code that implements the functional interfaces
Web Container
Vendor-supplied code that implements the functional interfaces
EJB Container
Vendor-supplied code that implements the functional interfaces

Separation of Job Scheduler & Batch Endpoint

The job scheduler gives you control to submit & manage jobs

Batch applications deployed like other WAS applications in the **Endpoint** server.

Stay started within the running JVM.



The job is an instance of program invocation based on the description offered in the xJCL file.

* The two may be in the same server, or separate servers. Or clustered. Your choice. ☺

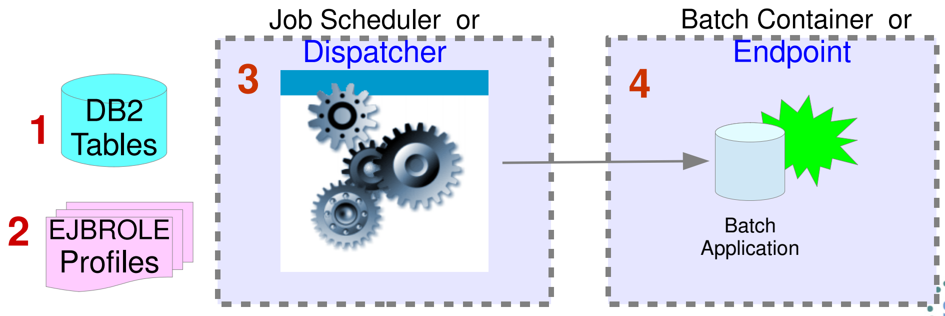
Simple Customization Steps For WebSphere Batch

Batch is Integrated in WAS V 8.5 profiles and "binaries"

(No longer "Augmented" as a "Stacked Product")

1. **Create Database tables & Data sources**
2. **Enable Application Security & Create EJBROLE Profiles**
3. **Configure Job Scheduler (Dispatching) in a Server (ISC)**
4. **Deploy a Sample Batch Application in an Endpoint Server (ISC)**
   - *Mapping to a server/cluster defines the Endpoint cluster.*

See "**WAS 8.5 *Java Batch Runtime Quick Start Guide***" Techdoc WP101783.



xJCL: Declaring the Structure of a Job

Concepts same as traditional // JCL ... syntax different

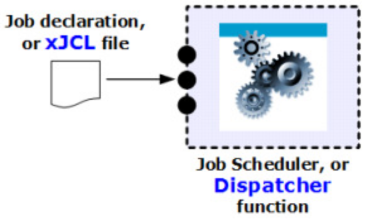
<job name= > ... </job> analogous to the // JOB card

```
<?xml version="1.0" encoding="UTF-8" ?>
<job name="Sample" default-application-name="Sample" ... ">
<jndi-name>ejb/com/ibm/ws/batch/SampleBatchController</jndi-name>
     :
     <substitution-props>
        <prop name="ABC" value="1000" />
        <prop name="XYZ" value="/tmp/Sample.txt" />
     </substitution-props>

   <job-step name="SampleStep1">
   <jndi-name>ejb/SampleModule1</jndi-name>
     :
     <props>
        <prop name="first.name" value="${ABC}"/>
     </props>
   </job-step>

   <job-step name="SampleStep2">
   <jndi-name>ejb/SampleModule2</jndi-name>
     :
     <props>
        <prop name="number" value="${ABC}"/>
     </props>
   </job-step>

</job>
```



<job-step <jndi-name> ... /job-step> analogous to // EXEC PGM= in traditional JCL

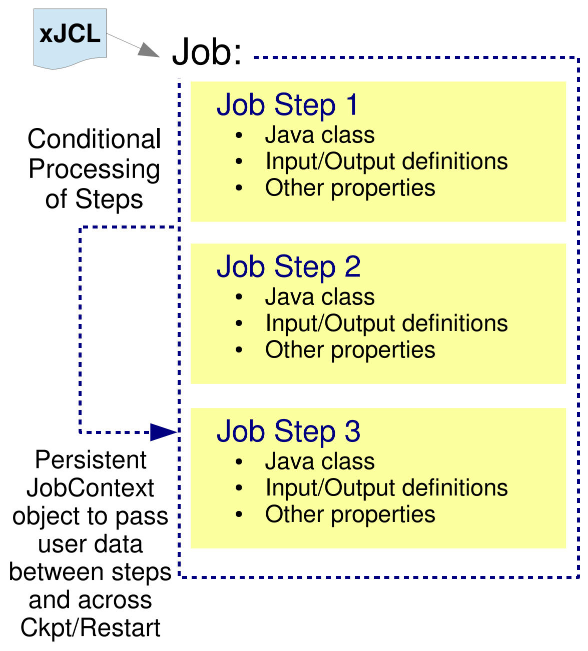
Substitution properties cariied down to variable declarations in XML

Not shown:
- Input/Output declarations
- Checkpoint declaration
- Conditional processing
- Much more

xJCL "describes" the elements of a "job"

Batch Job and Job Steps

A batch job consists of one or more steps executed in order...


xJCL
Job:
Job Step 1
• Java class
• Input/Output definitions
• Other properties
Job Step 2
• Java class
• Input/Output definitions
• Other properties
Job Step 3
• Java class
• Input/Output definitions
• Other properties
Conditional Processing of Steps
Persistent JobContext object to pass user data between steps and across Ckpt/Restart

Dispatcher interprets xJCL & determines which endpoint has batch application deployed

Dispatcher passes job (xJCL) to Endpoint server

• Steps executed in order, with conditional step processing if declared

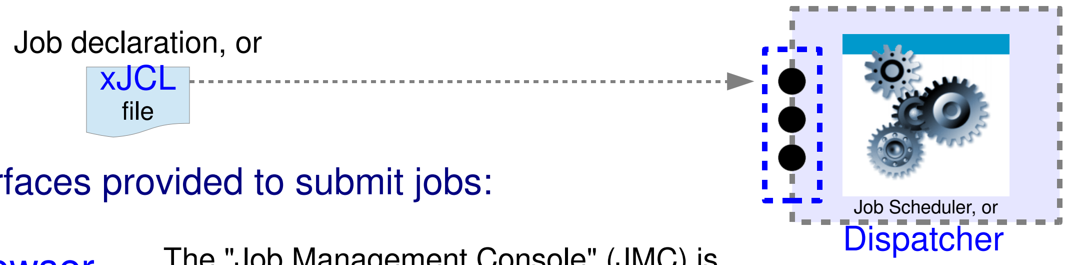
• Dispatcher maintains awareness of the job state

• When the job ends, Joblog accessible to the Submitter

Additional options:
•Step-Retry;
•Skip-Records;
•Restartable

Ways to Submit Jobs

Several different interfaces to the job dispatcher:

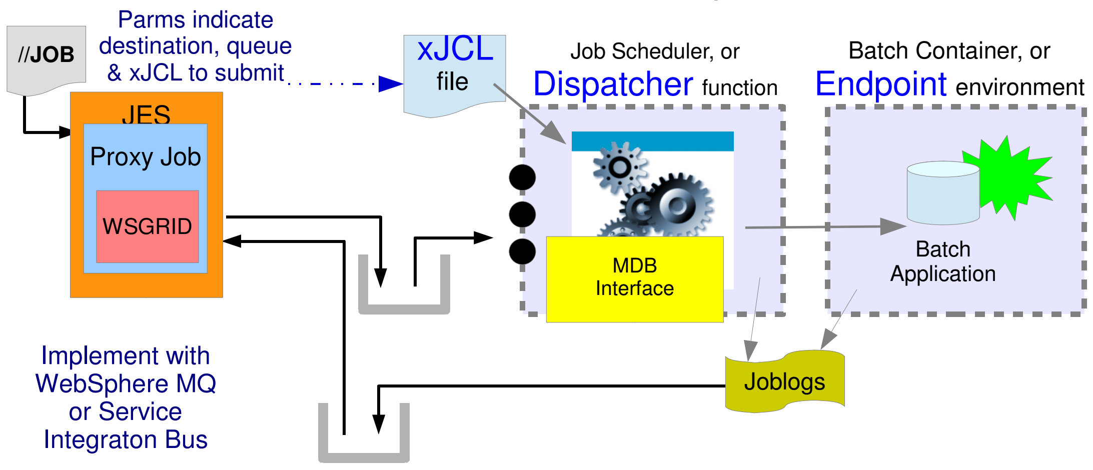


Interfaces provided to submit jobs:

Browser	The "Job Management Console" (JMC) is a simple-to-use browser application
USS Command Line	Using the supplied `lrcmd.sh` client
Web Services	Web Services clients may use this as a "batch service provider" in service oriented architecture
RMI	Java programs may submit jobs using the RMI interface
MDB	The Message Drive Bean interface allows job submission across a messaging queue. Heart of the integration with enterprise schedulers

Integration with Workload Schedulers

It's all about the MDB interface to the Dispatcher...



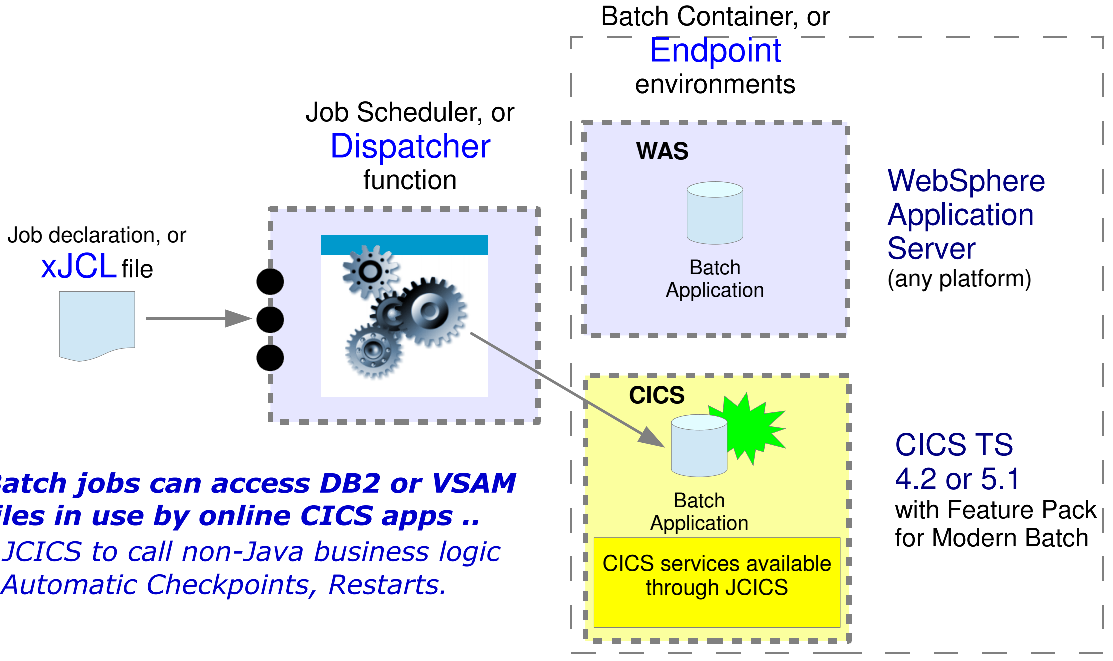
//JOB
Parms indicate destination, queue & xJCL to submit
xJCL file
Job Scheduler, or Dispatcher function
Batch Container, or Endpoint environment
JES
Proxy Job
WSGRID
MDB Interface
Batch Application
Joblogs
Implement with WebSphere MQ or Service Integraton Bus

If workload scheduler is capable of submitting JCL or invoking a shell script, it can submit a job into Compute Grid

The WSGRID utility stays up for duration of job in Compute Grid, and feeds Java batch output to STDOUT or JES

"Batch Containers" not Limited to WebSphere

There's also a Java batch container for **CICS** ...

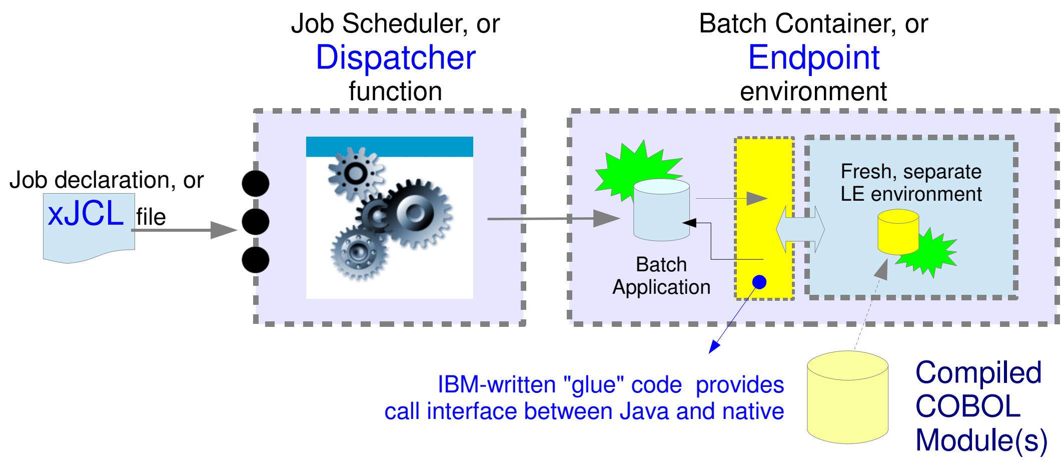


Batch Container, or Endpoint environments
Job Scheduler, or Dispatcher function
Job declaration, or xJCL file
WAS
Batch Application
WebSphere Application Server (any platform)
CICS
Batch Application
CICS services available through JCICS
CICS TS 4.2 or 5.1 with Feature Pack for Modern Batch

Batch jobs can access DB2 or VSAM files in use by online CICS apps ..
- *JCICS to call non-Java business logic*
- *Automatic Checkpoints, Restarts.*

Java Batch + COBOL? Yes ...

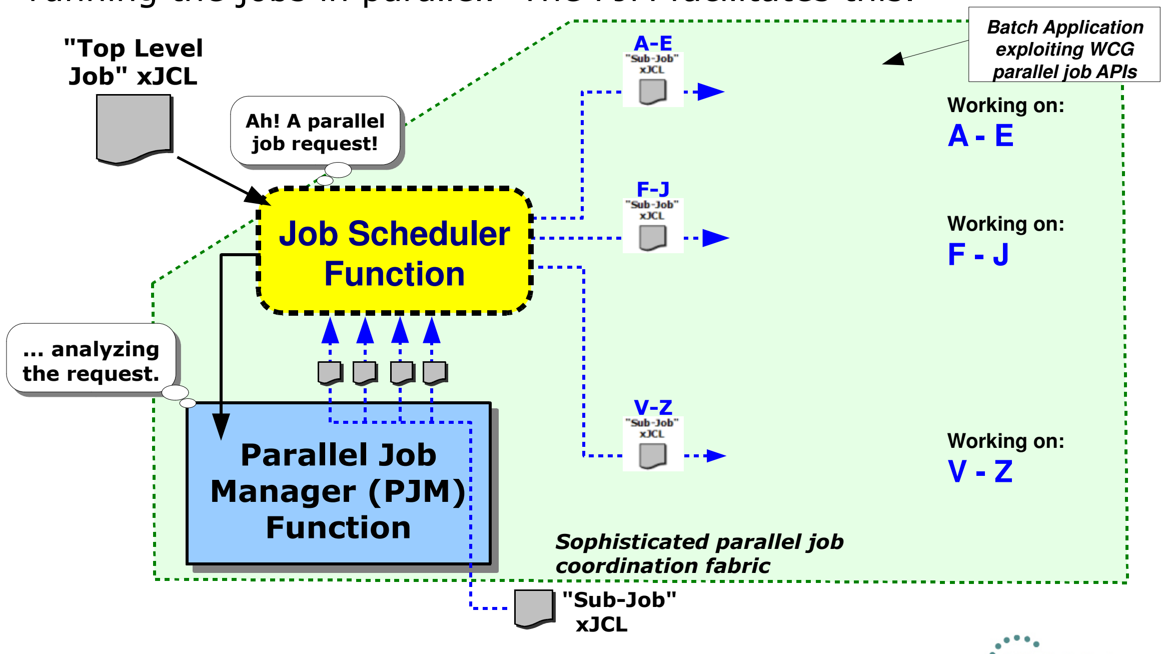
With a new COBOL interoperability function that allows Java batch programs to call COBOL programs directly ...



This provides a way to use (and re-use) COBOL assets as part of a Java batch job within Compute Grid

Parallel Job Manager (PJM)

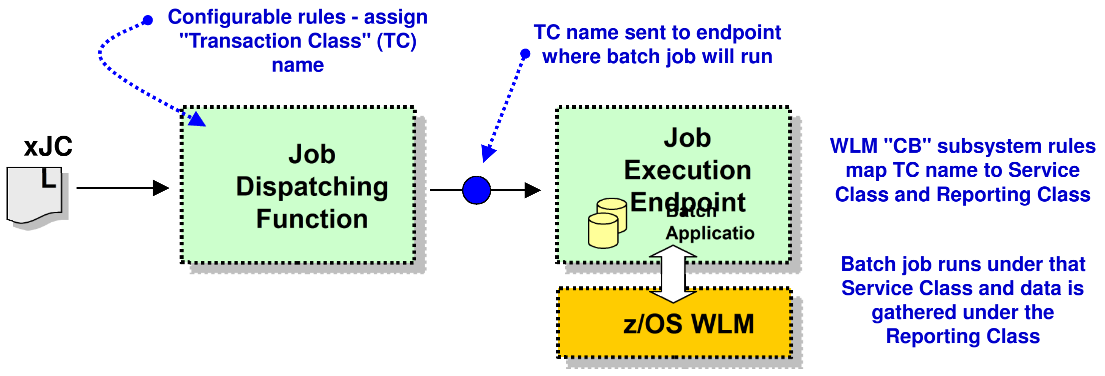
Batch processing often lends itself to data partitioning and running the jobs in parallel. The PJM facilitates this.



"Top Level Job" xJCL
Ah! A parallel job request!
Job Scheduler Function
... analyzing the request.
Parallel Job Manager (PJM) Function
A-E "Sub-Job" xJCL
F-J "Sub-Job" xJCL
V-Z "Sub-Job" xJCL
Batch Application exploiting WCG parallel job APIs
Working on: A - E
Working on: F - J
Working on: V - Z
Sophisticated parallel job coordination fabric
"Sub-Job" xJCL

Classifying Batch Jobs with WLM

Submitted jobs can be tagged with a WLM "transaction class," which may be used to map the batch job to a WLM Service Class or Reporting Class:

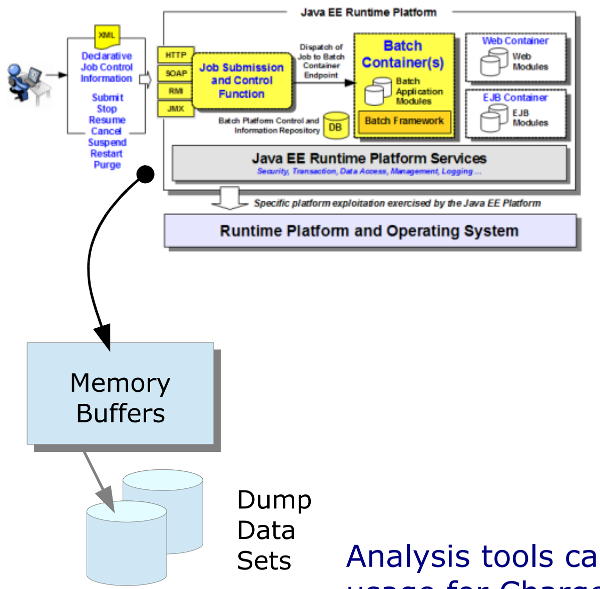

Configurable rules - assign "Transaction Class" (TC) name
TC name sent to endpoint where batch job will run
xJC
Job Dispatching Function
Job Execution Endpoint
Batch Applicatio
z/OS WLM
WLM "CB" subsystem rules map TC name to Service Class and Reporting Class
Batch job runs under that Service Class and data is gathered under the Reporting Class

Classify to a WLM Service or Reporting Class to manage Batch Work and gather system information for all work running in that Class

Unique to z/OS!

SMF Recording for Charge-Back & Capacity Planning

SMF is a powerful (and fast) activity recording subsystem on z/OS. Compute Grid z/OS exploits this with its own SMF record:


Java EE Runtime Platform
XML
Declarative Job Control Information
Submit
Stop
Resume
Cancel
Suspend
Restart
Purge
HTTP
SOAP
RMI
JMX
Job Submission and Control Function
Dispatch of Job to Batch Container Endpoint
Batch Container(s)
Batch Application Modules
Batch Framework
Batch Platform Control and Information Repository
DB
Web Container
Web Modules
EJB Container
EJB Modules
Java EE Runtime Platform Services
Security, Transaction, Data Access, Management, Logging ...
Specific platform exploitation exercised by the Java EE Platform
Runtime Platform and Operating System
Memory Buffers
Dump Data Sets

SMF 120.20 & 120.9 records contain additional info for Batch Jobs:

- Job identifier
- Job submitter
- Final Job state
- Server
- Node
- Accounting information
- Job start time
- Last update time
- General CPU usage
- zAAP or zIIP CPU use

Analysis tools can generate reports and determine usage for Charge-back, Capacity Planning, and Performance Measurement.

- Customer Use Profiles
- Development Tools
- Batch Programming Model
- Framework & Classes
- POC Scenarios

Customer Use Profiles - WAS Batch on z/OS

Large Financial Enterprise

Take advantage of newer development tools and skills

Increase agility -- faster time-to-market for changes.

Usage: Month-end, quarter-end & year-end batch jobs

- WebSphere Java batch spread across 6 LPARs
- WSGRID for integration with existing enterprise scheduler
- Parallel Job Manager for reduced overall completion time
- Heavy interaction with MVS datasets and DB2
- Integrated with WODM for business rules engine
- Co-located with batch in same WAS z/OS runtime servers

More Customer Use Profiles

International Insurance Enterprise

Exploit System z Specialty engines.

Usage: Lots of COBOL batch applications with many interdependencies

- Adopting Java batch incrementally based on batch job interdependencies
- COBOL Container lets Java call existing COBOL with low-level interface.
- Share JDBC T2 between Java and COBOL in same transaction scope
- Using WSGRID to integrate with enterprise scheduler.

Utility Company

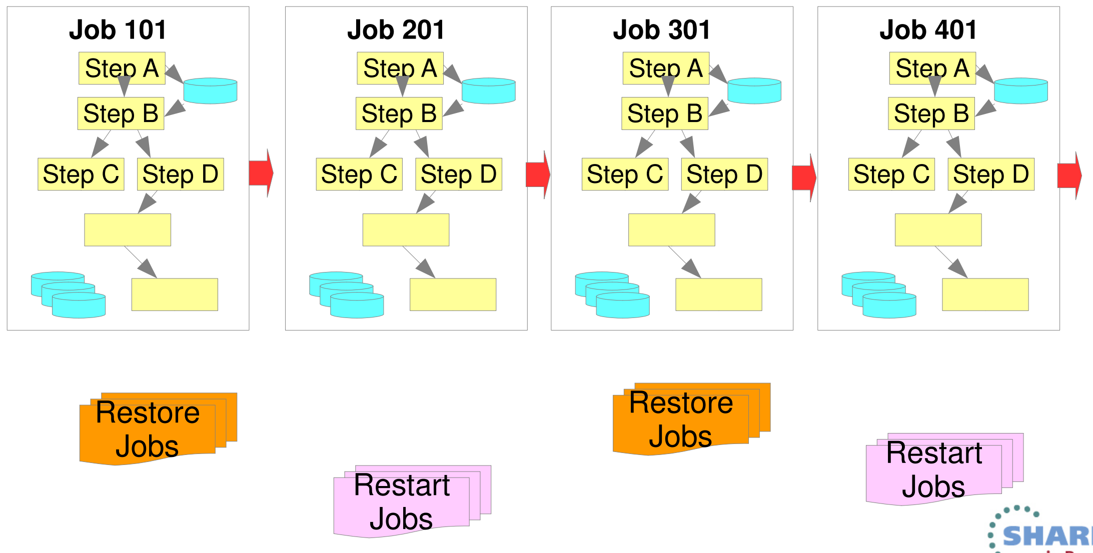
Improve Critical Path in Batch Cycle.

Usage: Cut Bills for 1/20th of the Customers each night

- Allow Batch jobs to be run while CICS online systems are up
  (Previously, online systems owning the database files must be offline so files could be accessed.)

The "Batch Cycle" (Network of batch job-streams)

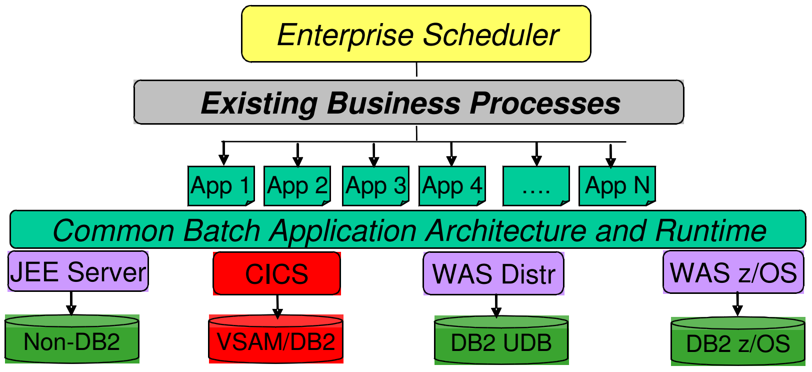
- Strict Sequence with Restore, Restart & Recovery Jobs.
- Every Night, Weekly, Monthly, Quarterly, Year-End.
- Run when Online Applications are Down.
- Strict Time Schedule... Critical Path = "Batch Cycle"



The Batch Vision:

Concurrent Access to Data with Online Systems

- ✔ Run anywhere, any time. Portable across platforms.
- ✔ Locate near data.
- ✔ Centrally managed by enterprise scheduler.
- ✔ Integrated with existing: Disaster Recovery, Auditing, Logging, Archiving.

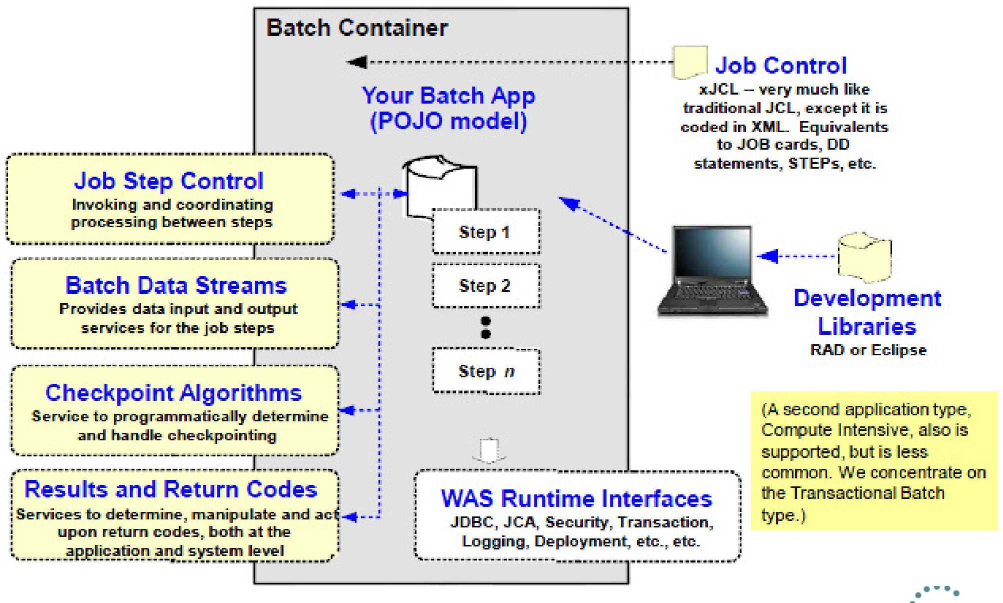


Examples:

- ***CICS Batch Feature Pack lets WebSphere schedule batch jobs in CICS***
- ***Increased availability for CICS during batch processing***

The Batch Programming Model

Functions & class libraries supplied with WebSphere Batch


Batch Container
Your Batch App (POJO model)
Step 1
Step 2
Step n
Job Control
xJCL – very much like traditional JCL, except it is coded in XML. Equivalents to JOB cards, DD statements, STEPs, etc.
Job Step Control
Invoking and coordinating processing between steps
Batch Data Streams
Provides data input and output services for the job steps
Checkpoint Algorithms
Service to programmatically determine and handle checkpointing
Results and Return Codes
Services to determine, manipulate and act upon return codes, both at the application and system level
WAS Runtime Interfaces
JDBC, JCA, Security, Transaction, Logging, Deployment, etc., etc.
Development Libraries
RAD or Eclipse

(A second application type, Compute Intensive, also is supported, but is less common. We concentrate on the Transactional Batch type.)

Function-Rich Programming Framework

Batch Data Streams (BDSF)

- Supplied “Patterns” for JDBC, JPA, J2C, Files, JZOS classes
- BDS maps data fields to Data Objects for Java
- Applications focus on Business Logic, Not data handling & Recovery

Checkpoint Processing

- Interval for Commit processing based on Time / Record numbers
- Restart failed jobs from Checkpoints

Extended Programming Functions

- Skip-Record Processing to tolerate data read/write errors
- Retry-Step Processing – Allow job to continue with errors (with customizable actions)
- Configurable Transaction Modes (Local/Global at Step level)
- Batch Data Stream Timeout configurable at BDS level
- Record Metrics available through JobStepContext object
- Parallel Job Manager, Parallel Steps & Multi-threading or Multi-JVM
- COBOL Container – Share JDBC Type 2 Connectors

WebSphere Batch Programming Education

Topics, including xJCL references & sample code:

1) Job & Step Overview
   - Batch vs. Compute-Intensive
2) Implement the Batch Step
   - Create Java class
3) Batch Data Streams (xJCL)
4) Batch Loop & Checkpoints
5) Compute-Intensive Steps
6) Job Step Context
   - Object providing user data area
7) Setting/Using Step Return Codes
8) Job States
9) Exceptions & Failures
10) Batch Data Streams
11) Transaction Mode
12) Database Cursors
13) Batch Framework
14) Step Retry
15) Skip Record Processing
16) Application Packaging

References:

- www.ibm.com/developerworks/websphere/techjournal/0801_vignola/0801_vignola.html
- www.ibm.com/developerworks/websphere/techjournal/1109_alderman/1109_alderman.html
- WebSphere V8.5 InfoCenter

Some JZOS Classes to help Batch Apps:

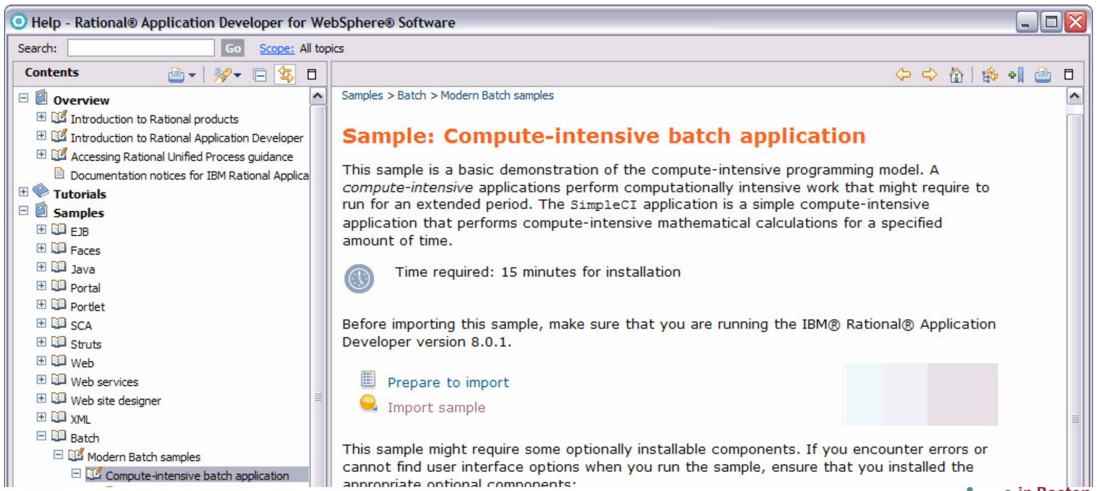
JZOS has functions that make Java on z/OS much easier & more useful.
Use them in your batch application development:

- **DfSort** - Invoke DFSORT to perform high-volume sort and merge operations
- **Exec** - Run external process that buffers output, provides timeout control and stdout/stderr character encoding.
- **File Factory** - Build a BufferedReader, BufferedWriter, InputStream, or OutputStream on a text file or MVS dataset.
- **JzosPermission** - Simple Permission class to allow JZOS to operate with a SecurityManager (such as RACF)
- **MvsConsole** - Class with static methods to interface with the MVS console.
- **WtoMessage** - Data object/bean for holding a WTO message and parameters.
- **MvsJobSubmitter** - Submit batch jobs to JES2 or JES3 from a Java program
- **PdsDirectory** - Opening a PDS directory and iterating over its members.
- **Zfile** - JNI Wrapper for z/OS C-Library IO routines.
- **Zutil** - Static interface to various z/OS native library calls - getCurrentJobId(), getCurrentUser(), getCpuTimeMicros()

See www.ibm.com/developerworks/java/zos/javadoc/jzos

Application Development Environments

- Rational Application Developer tooling for WebSphere Batch (RAD)
  - Wizard based dialogs for creating batch application artifacts
  - Designed around the “Batch Data Stream Framework API”
- Techdoc WP101788 “Beginners guide to developing Java Batch Apps”
- Redbook SG24-7835 “RAD for WebSphere Software V8 Programming Guide”


Help - Rational® Application Developer for WebSphere® Software
Search:
Go
Scope: All topics
Contents
Overview
Introduction to Rational products
Introduction to Rational Application Developer
Accessing Rational Unified Process guidance
Documentation notices for IBM Rational Applica
Tutorials
Samples
EJB
Faces
Java
Portal
Portlet
SCA
Struts
Web
Web services
Web site designer
XML
Batch
Modern Batch samples
Compute-intensive batch application
Samples > Batch > Modern Batch samples

Sample: Compute-intensive batch application

This sample is a basic demonstration of the compute-intensive programming model. A *compute-intensive* applications perform computationally intensive work that might require to run for an extended period. The `SimpleCI` application is a simple compute-intensive application that performs compute-intensive mathematical calculations for a specified amount of time.

Time required: 15 minutes for installation

Before importing this sample, make sure that you are running the IBM® Rational® Application Developer version 8.0.1.

Prepare to import

Import sample

This sample might require some optionally installable components. If you encounter errors or cannot find user interface options when you run the sample, ensure that you installed the appropriate optional components:

Other Appl'n Development Environments

Options for Developers to test Locally on workstations:

- WebSphere Application Server Unit Test Environment (UTE)
  - No charge: WebSphere Application Server Developer Tools and WebSphere Application Server for Developers

  ibm.com/developerworks/downloads/ws/wasdevelopers/index.html

- Batch Simulator
  - Stand-alone J2SE batch simulator to unit-test batch jobs:

  ibm.com/developerworks/websphere/techjournal/0801_vignola/0801_vignola.html

- Batch model simplifies “swapping” platform-dependent components
  - BDS implementation for flat-file for z/OS data-set inputs.

Rational Application Developer tooling for WebSphere Batch still the best!

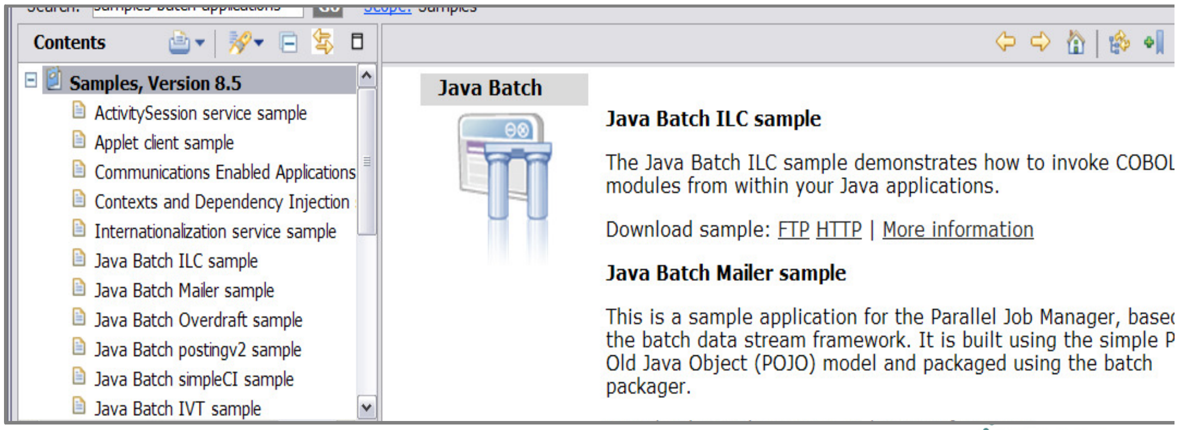
WebSphere V. 8.5 Sample Applications

- Download samples from the “Samples, V 8.5” Infoormation Center:

  http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp
  - Samples no longer packaged with the WebSphere product.

On the Downloads tab, Download compressed file, or individual sample files.

- Samples have an /installableApps directory containing deployable prebuilt archives.
- Contain sample-specific source archives, scripts & instructions for building apps.


Contents
Samples, Version 8.5
ActivitySession service sample
Applet client sample
Communications Enabled Applications
Contexts and Dependency Injection
Internationalization service sample
Java Batch ILC sample
Java Batch Mailer sample
Java Batch Overdraft sample
Java Batch postingv2 sample
Java Batch simpleCI sample
Java Batch IVT sample
Java Batch

Java Batch ILC sample

The Java Batch ILC sample demonstrates how to invoke COBOL modules from within your Java applications.

Download sample: FTP HTTP | More information

Java Batch Mailer sample

This is a sample application for the Parallel Job Manager, based the batch data stream framework. It is built using the simple P Old Java Object (POJO) model and packaged using the batch packager.

Application Selection for a P.O.C.

Choosing the right application:

Workload Profile

- CPU-bound work can demonstrate capabilities of off-load eligibility
- IO-Bound applications still yield good results.

Application Dependencies

- Applications that interact with many other apps make poor POC candidates.
- Select an application that stands-alone to minimize the development effort
- Focus can be on developer tooling training, and performance comparisons.
- New Application or Existing COBOL application?

External System Integration

- WebSphere batch applications can integrate any external system you need.
- Choose an application that accesses DB2 or MVS data to demonstrate common integration patterns.

Success Criteria – Setting Expectations

Functional:

- Demonstrate equivalent capabilities
- MVS and DB2 integration
- Checkpoint / Restart capabilities

Operational:

- Integration with z/OS Job Scheduling (TWS, ZEKE, CA7, etc)
- Test Fail-over scenarios
- SMF 120.9 record generation & reporting

Measurement Criteria:

- CPU time
- Elapsed Time
- % Offload to specialty engines (zIIPs & zAAPs)
- Initial performance profiling

Transforming traditional JES job-streams

Keep Enterprise Scheduler as your Central Controller

- Replace individual job steps with WSGRID Batch Steps
- Keep traditional utilities as-is, or replace with Java / JZOS apps.
  (e.g., DFSORT can be invoked via JZOS services.)
- Use COBOL container to integrate Java-written functions while retaining existing COBOL code.
- Use CICS container to use CICS apps & access VSAM or DB2

Advanced Functions

- Parallel Job Manager can provide more horizontal parallelism.
- Automated Recovery and Restart processing
- WOLA Adaptors for efficient adapters to CICS, IMS, Batch, USS, ALC
- Parallel Sysplex, Reliability, Availability, and Fail-over scenarios.
- Performance tuning & measurement with WLM & SMF.

Where to go for more Help and Education?

- DevWorks
- IEA
- Techdocs
- RedBooks
- Facebook
- Twitter

ibm.com/developerworks/websphere
- Intro to batch programming - techjournal/0801_vignola/0801_vignola.html
- Skip-record processing - techjournal/1109_alderman/1109_alderman.html
- Simple Compute Grid Parallel Batch Application – tutorials/1203_usha
- Enterprise batch processing techjournal/1210_narain/1210_narain.html
- Integration w/ ent.schedulers techjournal/1303_narain/1303_narain.html

ibm.com/support/techdocs/atsmastr.nsf/Web/TechDocs
- WP101783 - WebSphere Modern Batch
- WP101788 - Beginners Guide to Coding Java Batch Jobs
- WP101909 - WebSphere Compute Grid COBOL Container
- WP102231 - WebSphere Compute Grid z/OS Capacity Planning
- WP101532 - Why WAS for z/OS
- WP102110 - WAS for z/OS Liberty Profile
- WP101490 - WebSphere Optimized Local Adapters (WOLA)
- PRS4644 - WebSphere Compute Grid for z/OS Wildfire class mat'ls
- PRS4467 - WebSphere ATS - YouTube Video Flyer with Hiperlinks

- www.websphereusergroup.org/zos
- InfoCenter – http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp

Summary

1. Quick Start with Simple Configuration in WAS 8.5.
2. Experiment with Sample Applications
   (IVT, Mailer, Posting).
3. Integrate with Enterprise Scheduler.
4. Create stand-alone Java applications,
   then inter-operate with COBOL and CICS applications.
5. Incorporate into your Batch Production Cycles.

Appendix

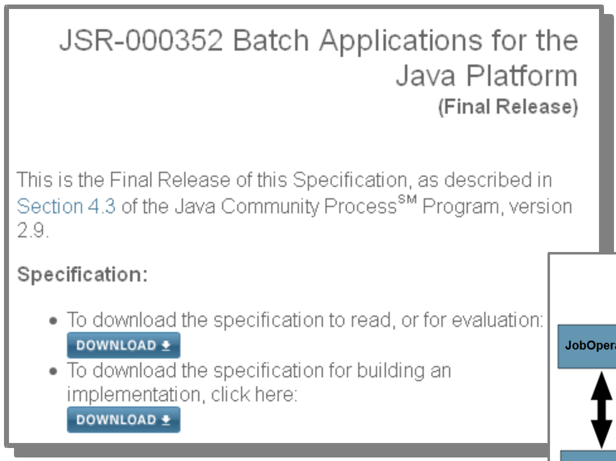
References & Back-up

WebSphere Application Server on z/OS Sessions

Day	Time	Room	#	Title	Speaker
Monday	9:30	203	13597	Getting Started with WebSphere Liberty Profile on z/OS	David Follis
Monday	4:30	203	13600	Managing Server Output from WAS on z/OS	Mike Loos
Tuesday	9:30	203	13644	Using WAS Optimized Local Adapters (WOLA) to migrate your COBOL to zAAP-able Java	Jim Mulvey
Tuesday	11:00	203	13640	Need A Support Assistant? Check Out IBM's! (ISA)	Mike Stephen
Tuesday	1:30	207	13953	What Would Life Be Like If You Ran Your Internet Applications On z/OS?	Ed McCarthy
Tuesday	3:00	203	13641	zWAS: In Real Life	Rod Feak
Wed.	1:30	202	13601	Lab: WebSphere Liberty Profile on z/OS	everybody
Thursday	11:00	203	13598	Getting Started With Compute Grid	John Hutchinson
Thursday	3:00	203	13645	Configuring Security for Liberty	Mike Loos

JSR 352 (Java Specification Requests 352)
- Batch Application for the Java Platform

- The V1.0 final release dated April, 2013, is available for download from the following URL:
  http://jcp.org/aboutJava/communityprocess/final/jsr352/index.html

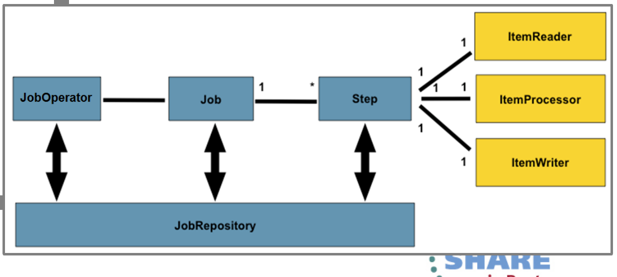

JSR-000352 Batch Applications for the Java Platform
(Final Release)

This is the Final Release of this Specification, as described in Section 4.3 of the Java Community Process^{SM} Program, version 2.9.

Specification:

- To download the specification to read, or for evaluation:
  DOWNLOAD
- To download the specification for building an implementation, click here:
  DOWNLOAD

"Domain Language of Batch" section provides an overview of key concepts & key terms.
+ Detailed sections on API specs.



Sample Application Development Challenges

After the solution design, several development challenges encountered:

- **COBOL CopyBook** conversion to Java Bean
  - EBDIC / Packed Decimal conversion to UTF-8 / 2's Compliment binary
  - Utilized JZOS' RecordFieldGenerator for record interpretation and conversion.
- **Print to JES Spool**
  - Requirement to print job output to JES with unique Job Identifier
  - Segregate output per job submission from within WAS CG
  - Combine PJM and JZOS to submit dynamically created IEBGENER jobs to JES for print to spool.
- **PJM**: Various approaches to job parallelization

Performance Thoughts . . .

Measurement Criteria:

- CPU time
- Elapsed Time
- % Offload to specialty engines (zIIPs & zAAPs)

"It depends..."

Dependent Factors:

- Interpreted vs. JITed Code
- New Javas can take advantage of new zEnterprise Hardware
- What about your compiled COBOL modules?
- Additional Performance Measurement & Tuning activities

Other...

- Parallel Sysplex, Reliability, Availability, WLM, SMF

Batch Programming Model

xJCL files

Essential interfaces:

- BatchJobStepInterface Methods
  - createJobStep, getProperties, processJobStep, destroyJobStep
- BatchDataStream Methods
  - open(), close(), getProperties, position, initialize, externalizeCheckpointInfo, ...

Optional interfaces

- Checkpoint policy
- Results
- Plus many more . . .

Use existing Sample Applications as a Model

System z Social Media Channels

- Top Facebook pages related to System z:
  - IBM System z
  - IBM Academic Initiative System z
  - IBM Master the Mainframe Contest
  - IBM Destination z
  - Millennial Mainframer
  - IBM Smarter Computing
- Top LinkedIn groups related to System z:
  - System z Advocates
  - SAP on System z
  - IBM Mainframe- Unofficial Group
  - IBM System z Events
  - Mainframe Experts Network
  - System z Linux
  - Enterprise Systems
  - Mainframe Security Gurus
- Twitter profiles related to System z:
  - IBM System z
  - IBM System z Events
  - IBM DB2 on System z
  - Millennial Mainframer
  - Destination z
  - IBM Smarter Computing

YouTube accounts related to System z:

- IBM System z
- Destination z
- IBM Smarter Computing

- Top System z blogs to check out:
  - Mainframe Insights
  - Smarter Computing
  - Millennial Mainframer
  - Mainframe & Hybrid Computing
  - The Mainframe Blog
  - Mainframe Watch Belgium
  - Mainframe Update
  - Enterprise Systems Media Blog
  - Dancing Dinosaur
  - DB2 for z/OS
  - IBM Destination z
  - DB2utor

Slide Title (Type Size=28) (no more than two lines)

- First Major Topic (Type Size=24)
  - Subtopic One (Type Size=22)
  - Subtopic Two (Type Size=22)
    - Sub-subtopic (Type Size=20)
- Second Major Topic (Type Size=24)
- Third Major Topic (Type Size=24)
- Fourth Major Topic (Type Size=24)