Batch Modernization:
Batch Improvements in z/OS 1.13

Speaker Name: Bob Rogers
Presentation created by: Gary Puchkoff
Speaker Company: IBM Corporation

Date of Presentation: August 9, 2011
Session Number: 9676

Agenda

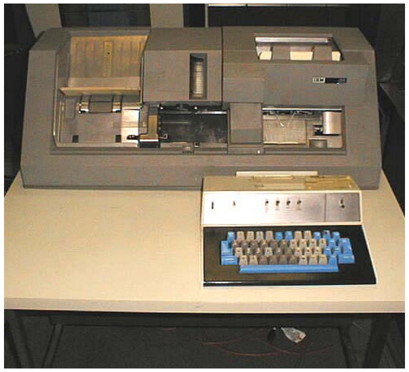
- Why is batch important?
- z/OS batch – part of the OS fabric
- IBM’s Batch Modernization effort
- New features
- Improvements

What is batch computing?

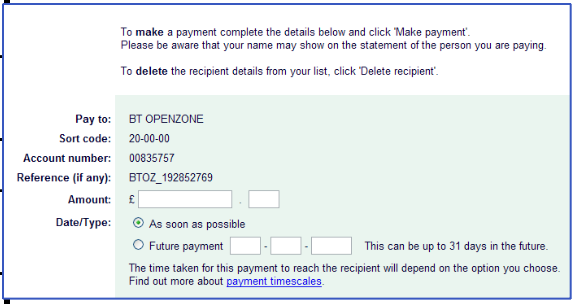
Batch processing *system for processing data with little or no operator intervention. This allows efficient use of the computer and is well suited to applications of a repetitive nature, such as file format conversion, payroll, or the production of utility bills.*

*In **interactive computing**, by contrast, data and instructions are entered while the processing program is running.*

Hutchinson Encyclopedia



Batch / Bulk Processing	**Interactive Computing**
Interface designed for machine	Interface designed for human
Optimized for machine utilization	Optimized for efficiency of human
Many items of data processed per human intervention	Single transaction processed per intervention
Automated	Manual
Served well by legacy languages & technology – COBOL, JCL, JES	Served well by modern languages & frameworks – Java, WebKit
Ceased to be a significant element of University curricula	Significant part of University curricula



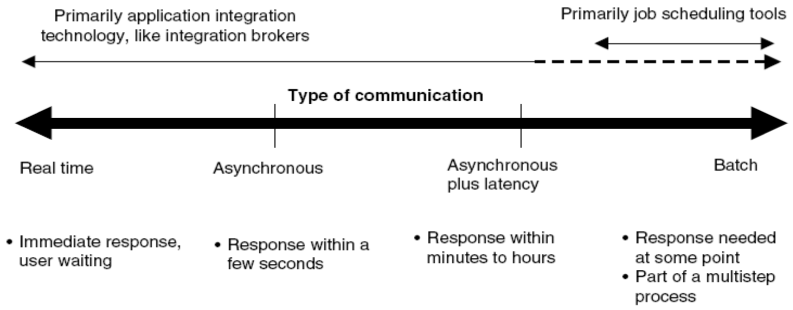
To **make** a payment complete the details below and click 'Make payment'.
Please be aware that your name may show on the statement of the person you are paying.

To **delete** the recipient details from your list, click 'Delete recipient'.

Pay to:	BT OPENZONE
Sort code:	20-00-00
Account number:	00835757
Reference (if any):	BTOZ_192852769
Amount:	£ ____ . __
Date/Type:	● As soon as possible ○ Future payment __ - __ - __ This can be up to 31 days in the future.

The time taken for this payment to reach the recipient will depend on the option you choose. Find out more about payment timescales.

- By eliminating human reaction time, batch can process in one hour what OLTP would take one month (or require 50,000 people)
- Batch is part of a continuous spectrum of workload
- Enterprise clients run a continual mix of online and batch – response to global, 24 x 7 business
- Business events as a trigger for batch e.g. end-of-day are a hallmark of a critical workload


Primarily application integration technology, like integration brokers
Primarily job scheduling tools
Type of communication
Real time
Asynchronous
Asynchronous plus latency
Batch
• Immediate response, user waiting
• Response within a few seconds
• Response within minutes to hours
• Response needed at some point
• Part of a multistep process

Source: "Consider Scheduling Tools for Batch Application Integration", Gartner.

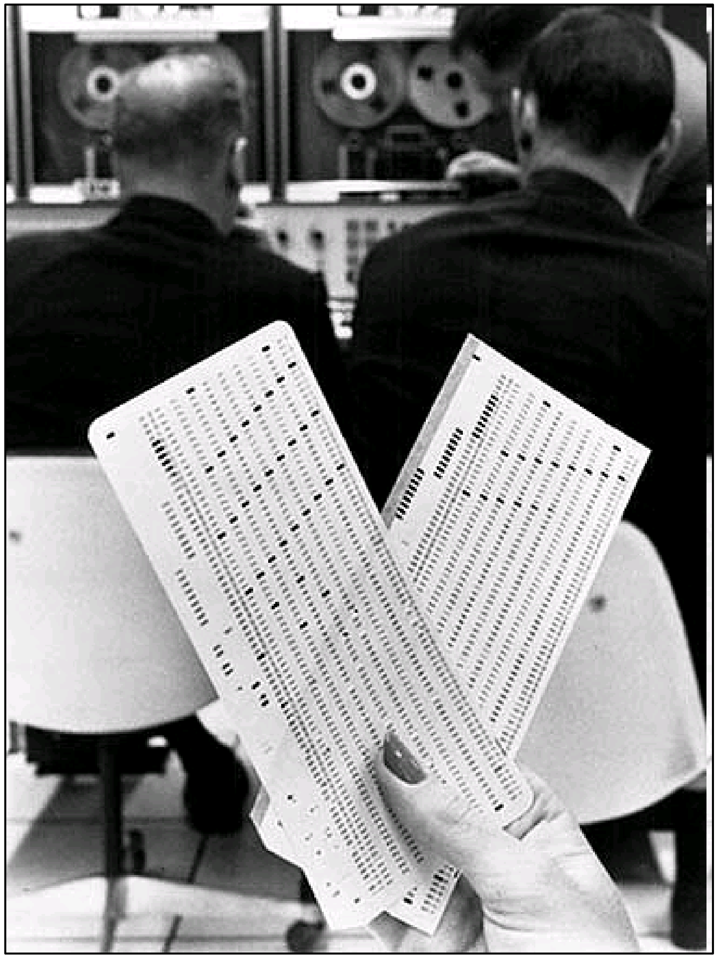
Why Batch?

- Economies of Scale
  - Processing all items in a collection
  - Pre-fetch optimizations
- Period based processing
  - End of day, month, year
  - Consistent reporting
  - Integration with others on a consistent basis

z/OS 45 years of Batch –
From punch cards to Java Batch

45 years of evolution in Batch processing on the IBM Mainframe have provided the foundation for heavy-duty, reliable and efficient Batch for most large companies in the world:

- WLM, WLM Batch initiators
- Batch & Print Subsystem JES2, JES3, PSF
- Job Control Language (JCL)
- Batch Management Interfaces (for example. SDSF)
- Step and Job dependencies by means of Condition Codes and Job Networks
- Online and Batch in parallel
- Time-driven Job execution
- Job / Step Restart functions, Start, Submit, Remote submit, Syntax Scanner
- Accounting based on Job/USER, Job statistics and RMF reports
- Pre-loaded address spaces (initiators)
- All Mainframe programming languages can be used in Batch



z/OS Batch – Integral part of the OS

- OS based batch submission
  - Always there
  - Scheduled
  - Secure
  - Resource Accounting
- OS based resource management
  - Dataset synchronization via enqueues
  - CPU, memory, I/O access based on WLM policy
- OS based device management
  - Disk, Tapes

IBM Batch Modernization initiative

- Ensure that IT architects and the industry recognize
  - Almost all large scale IT projects include batch
  - The web focus of the last decade has focused on online work, response time, transactions
  - Backend processing is often best implemented as batch
- Provide an IBM Batch solution for any platform – including z/OS
  - WebSphere Compute Grid
    - Both java programming model and execution environment
- CICS Compute Grid on z/OS
- Maintain leadership in z/OS batch
  - Improvements to existing function
  - Integrate and support the Compute grid programming model

WebSphere XD Compute Grid

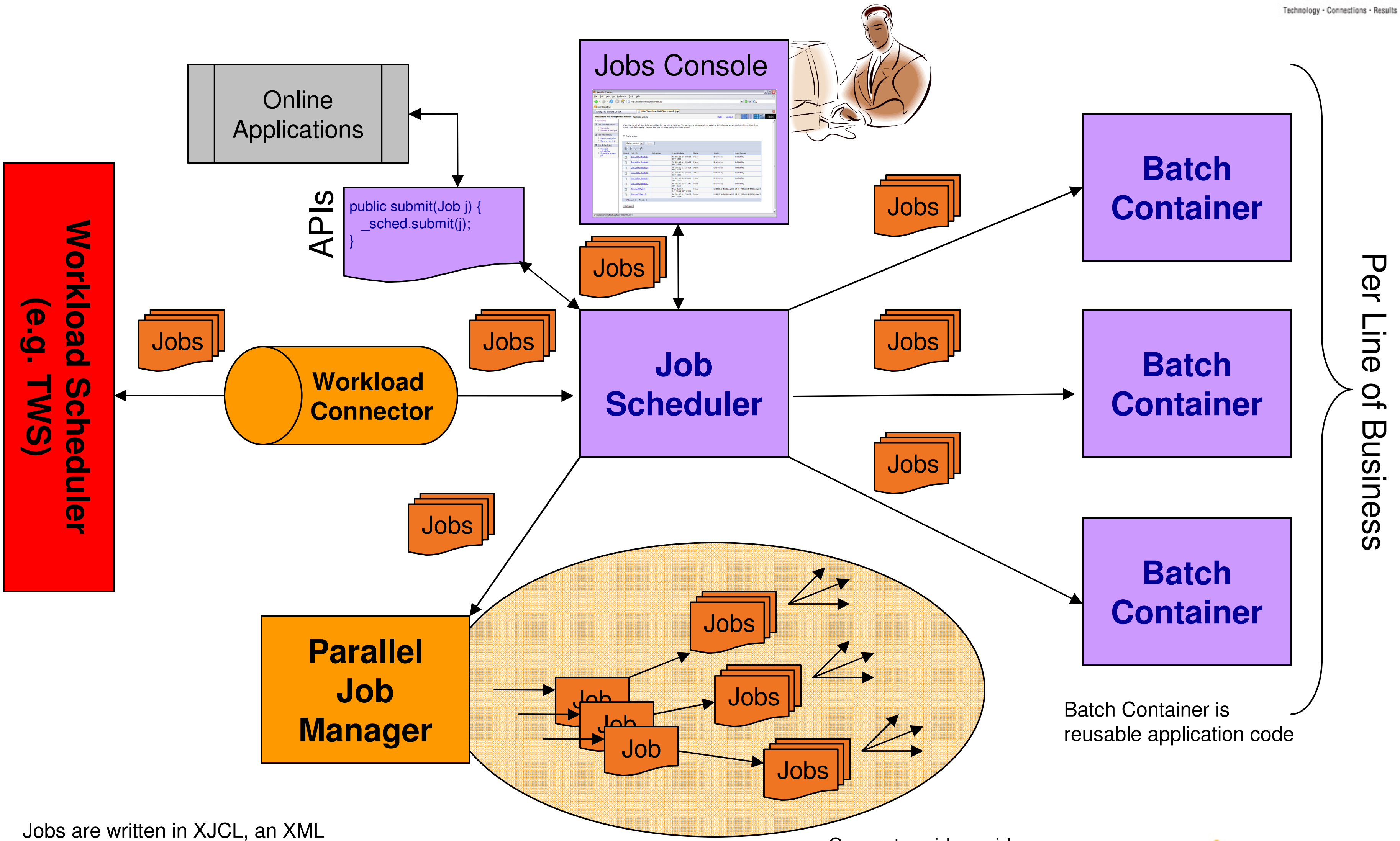
- Java Batch combined with all functionality provided by the WebSphere JEE Container

- Extension on top of WebSphere Application Server
  - Available as a feature pack on WAS V7 (limited function), or a product, WebSphere XD Compute Grid (full function)

- Java stand-alone functionality plus
  - WAS Container management
    - security, transactions and connection management
  - Checkpointing
  - Persistent JVM
  - QoS, such as high availability

- Reuse of OLTP code in WAS XD Compute Grid Batch Container

Compute Grid Overview diagram



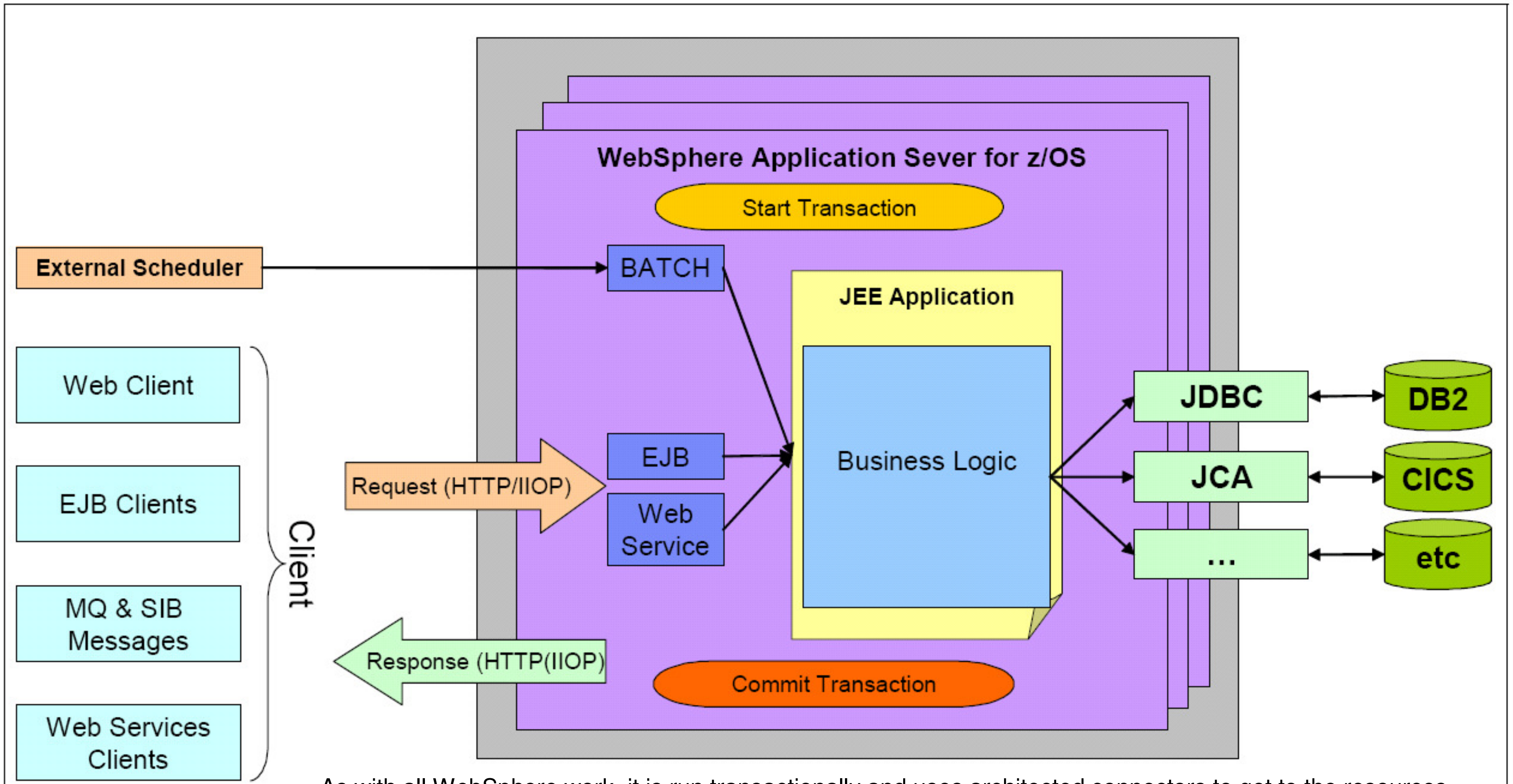
```
public submit(Job j) {
   _sched.submit(j);
}
```

Batch Container is reusable application code

Jobs are written in XJCL, an XML version of a job control language.

Compute grid provides parallel job management

WebSphere XD Compute Grid programming environment



As with all WebSphere work, it is run transactionally and uses architected connectors to get to the resources

Benefits of Java (Batch) on z/OS

- Specific Java Batch APIs for z/OS
  - Dataset and VSAM access
  - Condition Code passing
  - DFSORT support with a Java Class
  - Writing Logstreams
  - Triggering of Jobs from Java
  - RACF APIs
- Local DB2 Database driver for high throughput
- Access to many Java skills
- Effective and efficient development tools available
  - Rational Tools available specific to Batch Container
- Availability of many classes, libraries, frameworks and applications based on open source
- Interoperability with other programming languages on z/OS

z/OS Batch Runtime Environment (New)

- A new option for running batch work in z/OS 1.13
- Provides a managed environment for integration of Java and COBOL
- Consistent with IBM WebSphere based batch
  - A subset of the WebSphere programming model
  - Incorporated in the operating system
- DB2 as a resource manager

Batch Execution Runtime Environment Java COBOL with DB2 Interoperability

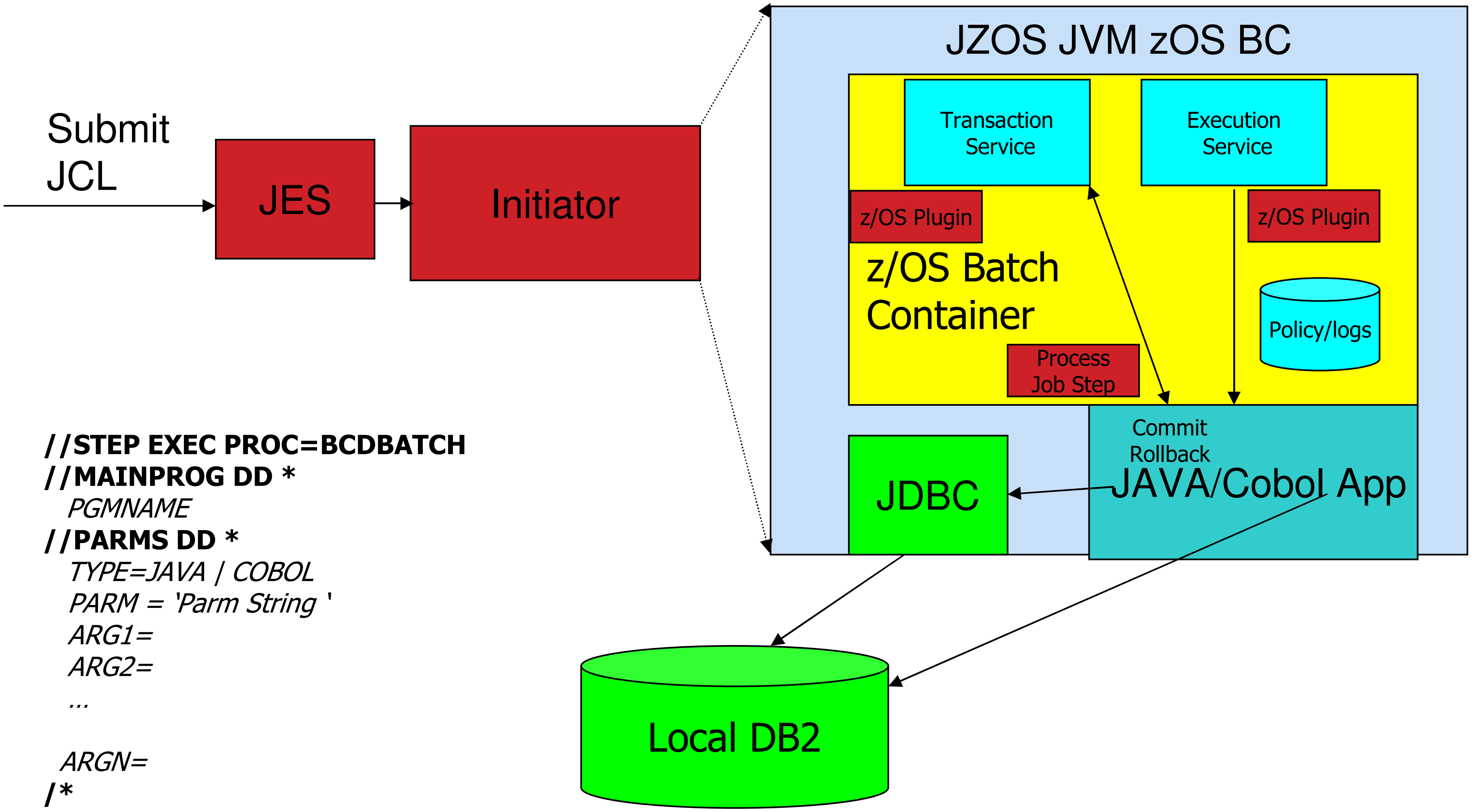
- Ability to extend COBOL/DB2 applications with either COBOL or Java:
  - Requires local attach z/OS DB2 connection sharing for common DB2 access
  - Requires Transactional integrity among the application components

- A generalized solution in a pure batch environment, without requiring a specific run time or middleware. It does not require an application server like CICS or DB2 Stored Procedures.

- Implementation requires little or no changes to existing code!
  - Only requires special callbacks for commit/rollback

- Enables interoperability between COBOL, Java and DB2 in batch

Our z/OS Topology
JES Single Step based

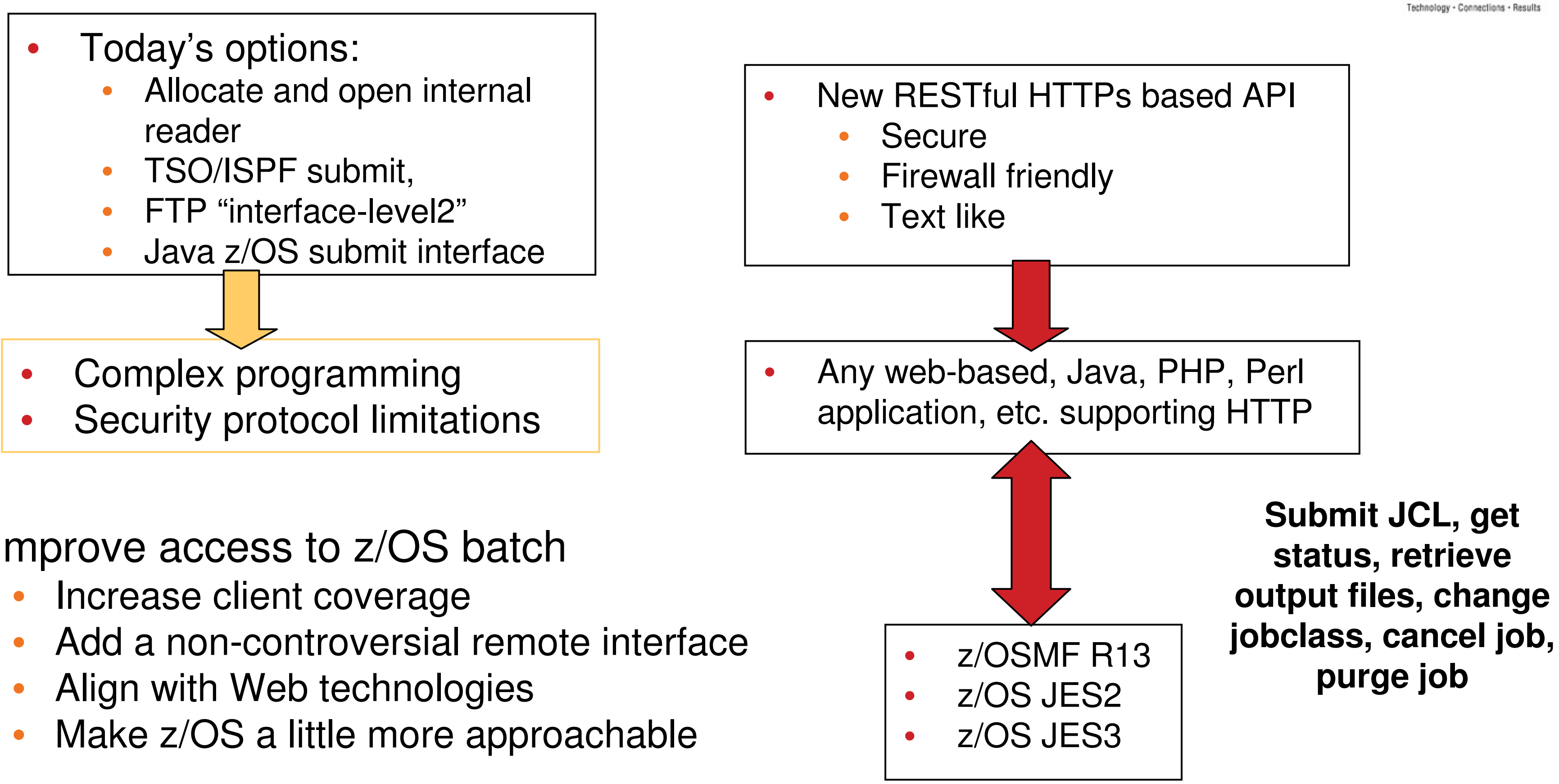


```
//STEP EXEC PROC=BCDBATCH
//MAINPROG DD *
  PGMNAME
//PARMS DD *
  TYPE=JAVA | COBOL
  PARM = 'Parm String '
  ARG1=
  ARG2=
  …

 ARGN=
/*
```

JCL is familiar to operations
code in the batch container is the same as the code one could write anywhere

JOBS REST API for z/OS – use with z/OS BATCH



- Today's options:
  - Allocate and open internal reader
  - TSO/ISPF submit,
  - FTP "interface-level2"
  - Java z/OS submit interface

- Complex programming
- Security protocol limitations

- New RESTful HTTPs based API
  - Secure
  - Firewall friendly
  - Text like

- Any web-based, Java, PHP, Perl application, etc. supporting HTTP

Submit JCL, get status, retrieve output files, change jobclass, cancel job, purge job

- z/OSMF R13
- z/OS JES2
- z/OS JES3

- Improve access to z/OS batch
  - Increase client coverage
  - Add a non-controversial remote interface
  - Align with Web technologies
  - Make z/OS a little more approachable

- Invoke the power of z/OS batch from virtually any non-z/OS application

JOBS REST Interface (cont)

- Four verbs: Get, Post, Put and Delete

- Example
  - A job named G1JAVA1B with job number JOB00023

GET **/zosmf/restjobs/jobs/G1JAVA1B/JOB00023** HTTP/1.1

- Result returned in JSON format

```
HTTP/1.1 200 OK
Date: Thu, 13 Jan 2011 05:39:28 +0000GMT
Content-Type: application/json
Connection: close
{
”jobid”:”JOB00023”,”jobname”:”G1JAVA1B”,”subsystem”:null,”owner”:”G1JAVA1”,
”status”:”OUTPUT”,”type”:”JOB”,”class”:”A”,”retcode”:”CC 0000”,
"url":"https:\/\/host:port\/zosmf\/restjobs\/jobs\/G1JAVA1B\/JOB00023",
"files-url":"https:\/\/host:port\/zosmf\/restjobs\/jobs\/G1JAVA1B\/JOB00023\/files"
}
```

Using the files-url you can retrieve the individual datasets. Data can be retrieve in chunks etc.

Instream data in PROCs and INCLUDEs

- Instream data in procs and includes (JES2)

```
//PROC1  PROC
//ASTEP   EXEC PGM=xyz
//DD1       DD *
   This is instream data
//              PEND
```

  - Prior to this support you could not have instream data in a JCL procedure
  - Support also allows instream data in include statements wherever a DD statement is allowed
  - Requires both z/OS 1.13 and JES2 1.13 on the converting system and the initiating system
  - This is not supported for MSTR subsystems
  - This is not supported by JES3

Instream data in PROCs and INCLUDEs

Instream data in PROC example

```
//HELLO   PROC
//STEP1   EXEC ASMHCLG
//C.SYSIN DD *
TEST     CSECT ,
         STM   14,12,12(13)
         BALR  12,0
         USING *,12
         ST    13,SAVAREA+4
         LA    13,SAVAREA
         SPACE 1
         WTO   'Hello world!'
         SPACE 1
         L     13,SAVAREA+4
         LM    14,12,12(13)
         SR    15,15
         BR    14
         SPACE 1
SAVAREA  DC    18F'0'
         END
//L.TEST  DD DUMMY
//L.SYSXX DD *
//       PEND
```

JOB return code (JES2 only)

- There is no formal definition of the job return code. The defacto standard is JES2 returning the maximum return code of the job.
- New job card keyword to control job return code
  - JOBRC= MAXRC | LASTRC | (STEP,*name.name*)
    - MAXRC is existing processing (default)
    - LASTRC is return code of last step
    - (STEP,*name.name*) is return code of identified step
      - *If step not executed, defaults to MAXRC*
- JOBCLASS JOBRC= MAXRC|LASTRC to affect processing for all jobs in the jobclass
- Affects return code seen in
  - Extended status (e.g. SDSF)
  - ENF 70
  - HASP165 message
  - $DJ,CC= command

Suspend a job at next step (JES2 only)

- If a job is restartable
  - A new operator command is defined to suspend the job at the next step
  - The job is requeued for execution
- Allows a faster and/or less disruptive shutdown
- The job must complete the step it is running.
  - This remains an issue for long running steps
- Can reduce Mean-Time-To-Recovery (MTTR) if the system must be gracefully brought down.

Suspend a job at next step (JES2 only)

- Remove job on step boundary
  - New STEP operand on $EJ command
    - Causes job to exit execution at end of current step
    - Optional HOLD operand makes job held
    - Job is requeued for execution
  - Job must be journaling (JOURNAL=YES on JOBCLASS)
  - Uses existing continue restart function of z/OS
    - Previously used to restart jobs after an IPL
  - Full syntax $EJxxx,STEP [,HOLD]
    - Full cross member support

Spin Any (JES 2 only)

- Ability to specify automatic spin options in JCL
  - Eliminates the requirement to take down a long running process to release spool space
  - Similar to what was done for JESLOG
  - Applies to any data set allocated as SPIN
    - No application code/JCL change needed
  - Provides a time, size, or command based option to spin output
- Update to SPIN= parameter on DD statement

```
SPIN= { NO }
        { (UNALLOC,'hh:mm') }          Spin at specific time
        { (UNALLOC,'+hh:mm') }         Spin every hh:mm interval
        { (UNALLOC,nnn [K|M]) }        Spin every nnn lines
        { (UNALLOC,NOCMND) }           Cannot be spun by command
        { (UNALLOC,CMNDONLY) }         Can be spun via operator command
```

- $TJ*n*,SPIN,DDNAME=*name* command added

Freeing Tape Volumes before end of step

- A new FREEVOL=EOV keyword on the JCL DD statement:
  - This is intended to allow overlapped processing for multivolume data sets, which can reduce batch processing elapsed time.
  - Specifies that a tape that is a part of a multivolume data set become available at end-of-volume rather than at step end.
  - Allows other jobs to use the tape immediately.
  - Allows overlapped processing of multivolume tape data sets.

IEBCOPY

- IEBCOPY performance is improved for copying from a partitioned data set (PDS) to another PDS
- IEBCOPY now exploits 31-bit storage for buffers
- The requirement for APF authorization of IEBCOPY is removed

COPYCNT

- New JCL OUTPUT statement keyword
- Synonym for COPIES=
- COPYCNT=x
  - Where x can be 1-4G
  - Usage depends on print driver… if the print driver is COPYCNT aware it will supercede any specification of COPIES=

- z/OS now allows the specification of copies >255
- No support in JES2 or JES3 for detecting or changing the value of COPYCNT
- SDSF can display and change COPYCNT

Options for improved Batch Elapsed time

- Hardware
  - CPU
    - Faster
    - More
  - I/O
    - zHPF
    - SSD drives
  - Memory
    - Large pages, More
- Software
  - Parallel
    - I/O, setup, execution
  - Optimize
    - Use less resource, exploit new algorithms

References

- Redbook – Batch Modernization for z/OS – SG24-7779

  http://www.redbooks.ibm.com/redbooks/pdfs/sg247779.pdf
- Redpaper – Batch Processing with Websphere Compute Grid: Delivering Business Value to the Enterprise
  http://www.redbooks.ibm.com/redpapers/pdfs/redp4566.pdf
- Java Batch Programming with XD Compute Grid
  http://www.ibm.com/developerworks/websphere/techjournal/0801_vignola/0801_vignola.html

Question

- Where would you like to see us focus on batch modernization?

- Please send e-mail to Gary Puchkoff (puchkoff@us.ibm.com)