

IBM WebSphere Application Server V8.5

Batch overview



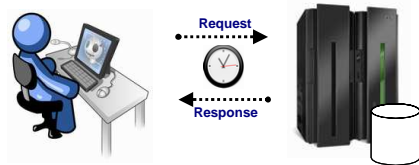
© 2012 IBM Corporation

This module provides an overview of the Batch feature of IBM WebSphere Application Server V8.5.

Batch processing

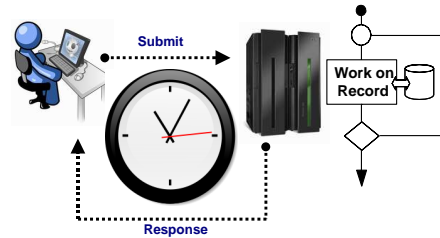
Comparison of online transaction processing and batch processing

Online Transaction Processing



- **A request / receive model**
(Often but not necessarily synchronous)
- **Duration of processing relatively short**
- **Often transactional in nature**
- **Application server runtime enforces timeouts for workload**

Batch Processing (or Long Running)



- **A submit / work / result set model**
(Asynchronous to the submitter)
- **Duration of processing a function of work to be done; can be hours/days**
- **Often transactional in nature**
- **Often multi-step processes**

2

Batch overview

© 2012 IBM Corporation

Online transaction processing is a request/receive model where the duration of the processing is relatively short, and the tasks are typically transactional in nature. In this model, the application server runtime enforces timeouts for the workload.

Batch processing is a submit/work/result set model where the duration of the processing is a function of the tasks to be completed. In some cases, the tasks can require hours or even days to complete. In this model, the work tasks are typically transactional in nature and typically involve multi-step processes.

Batch modernization

Providing greater flexibility / control of batch window processing

- Moving to a continuously available batch model intermixed with OLTP
- Using goal-oriented workload management to enable job completion within deadlines

SOA transformation - Running batch as a service

- Hosting batch processes within SOA-capable application server environments
- Optimally reusing services through efficient co-location

Leveraging Java skills for all applications

- Utilizing Java for batch processing and OLTP

Reducing cost of running batch on z/OS

- Moving batch processing to Java to take advantage of zAAP specialty engines
- Utilizing processor resources available when OLTP workloads are light

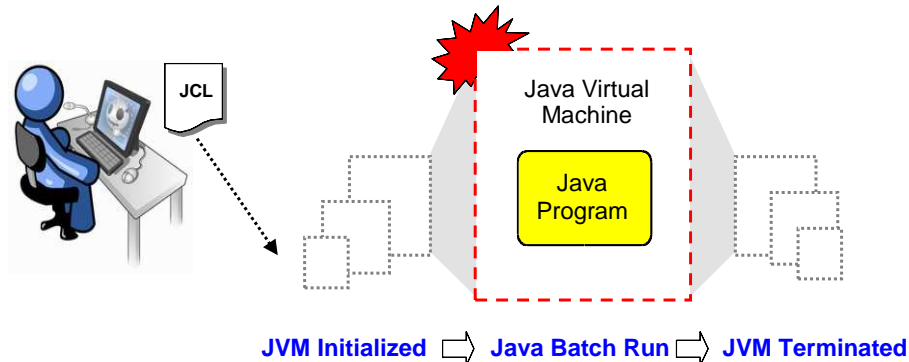
Modernization does not mean wholesale replacement of existing batch function that serves the business needs. Business imperatives drive technical solutions

Batch modernization involves specialized techniques to achieve job completion within deadlines, optimal reuse, leverage of JAVA skills, and reducing cost by way of System z Application Assist Processor (zAAP) specialty engines.

Greater flexibility and control of batch window processing is realized by moving to a continuously available batch model intermixed with online transaction processing. This coupled with using goal oriented workload management enables job completion within deadlines. Hosting batch processes within service oriented architecture (SOA) capable WebSphere Application Server environments lend to optimally reusing services through efficient co-location. Leveraging Java skills for batch and online transaction processing tasks helps reduce cost. Additionally cost reduction is achieved on the z/OS platform by moving batch processing to JAVA and taking advantage of System z Application Assist Processor (zAAP) specialty engines and utilizing processor resources available when online transaction processing workloads are light.

Use JVM launcher

Many attempt to use a JVM launcher to run Java batch programs



This approach works, but has limitations:

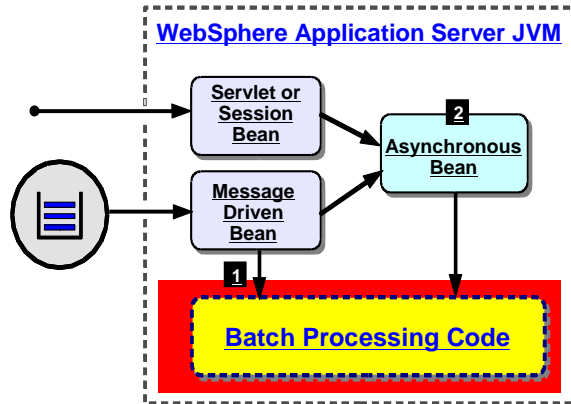
- High overhead associated with repeated cycling of JVMs
- Application must provide most batch processing services

Java Virtual Machine (JVM) launcher technologies are free and easy to use, and the programming interfaces are very useful; however, a new JVM is created for every step, and the resulting overhead is significant. In addition, JVM launcher technologies do not provide a transaction manager. Ultimately, the application developer must author a transaction manager within their application.

On the z/OS platform, WebSphere Application Server complements the Java Batch Toolkit for z/OS (JZOS). The JZOS launcher can be used when only a few Java batch steps must be executed. Once the size of the Java batch infrastructure grows to the point where the overhead of creating and terminating JVMs is an issue, the Batch feature of WebSphere Application Server can be used. The second component of JZOS, it's API's for accessing z/OS resources, can be used from WebSphere Application Server applications.

Use application server runtime

A second approach uses the application server runtime and authors the batch application as a message driven bean or an asynchronous bean

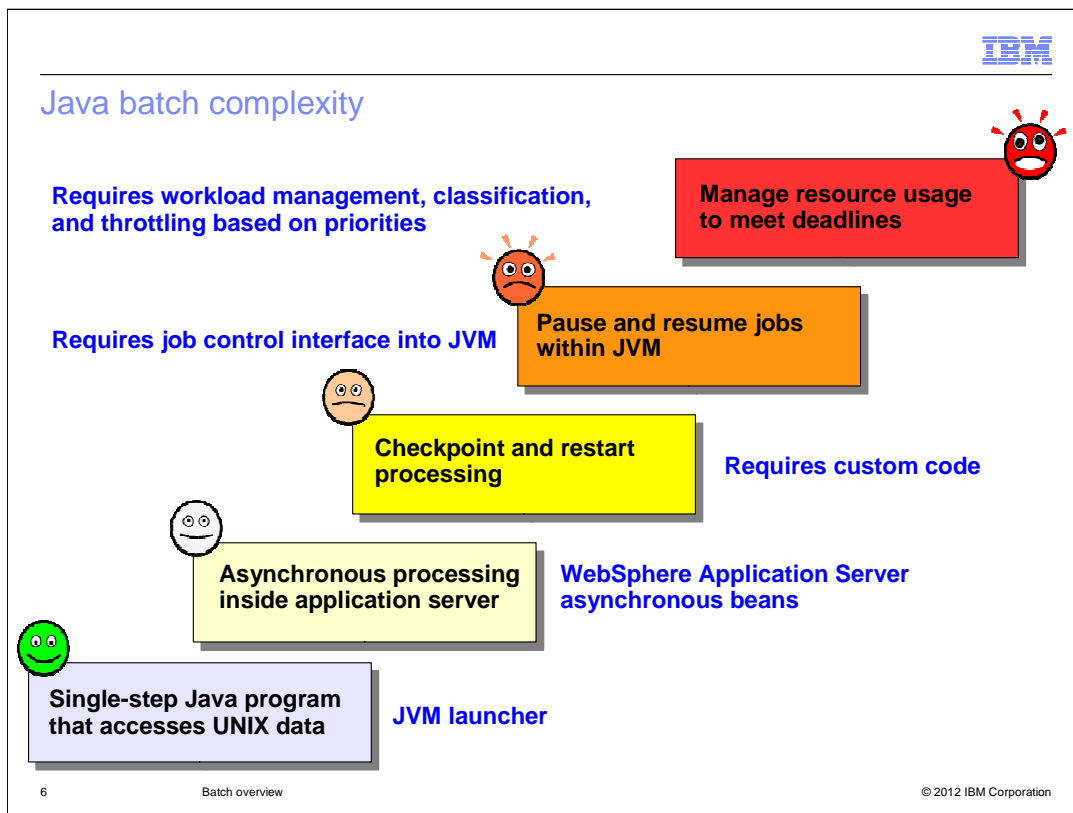


1. When a message driven bean is used, the application must manage the timeout on the message driven bean input
2. The preferred technique (used by WebSphere Application Server and WebSphere Extended Deployment Compute Grid) is to use an asynchronous bean

In both cases, the application developer must author custom middleware to support the batch processing

An application server runtime is another option that can be used to run the batch application. However, this solution is lacking features that need to be implemented as custom middleware.

Examples of custom middleware functionality that would need to be implemented include: A batch programming model and development tooling, interweaving of online transaction processing tasks and batch tasks, check pointing and restarting of jobs, integration with enterprise schedulers, graphical and command line interfaces for job monitoring and management, job usage accounting, and activity logging.



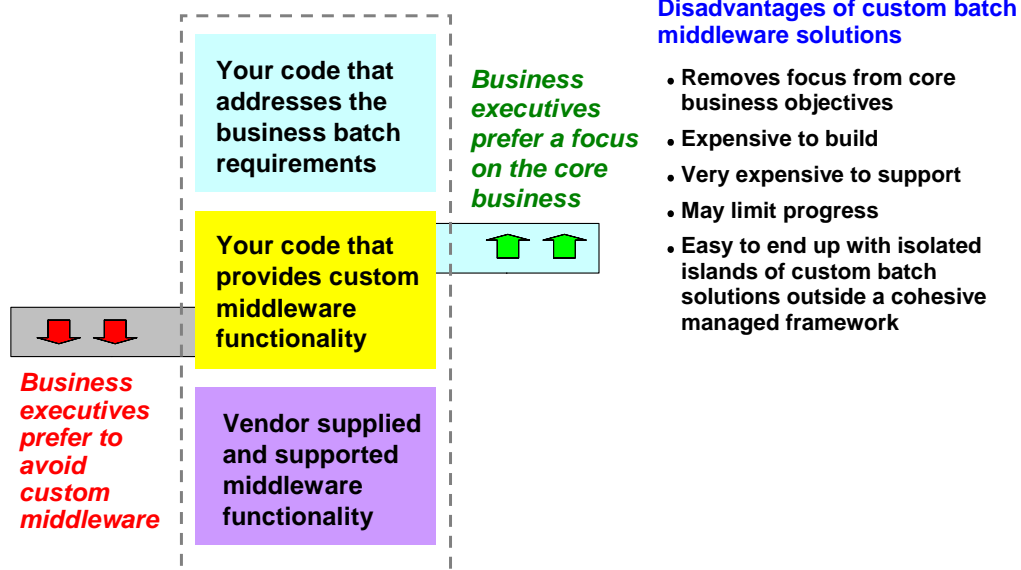
Initial Java batch programming projects typically begin with a relatively simple Java Virtual Machine (JVM) launcher design, but it grows more complex over time as additional management features are required. These projects frequently progress in the following manner:

An organization uses the Java command line interface or the Java Batch Toolkit for z/OS (JZOS) to run a single-step Java program that accesses UNIX data. After realizing that transaction management capabilities are required, the organization moves the application to a Java Enterprise Edition (Java EE) application server.

As the organization begins to process large numbers of records as a single batch job, it finds that it is no longer practical to restart entire jobs when failures occur in the middle of a job. The organization then introduces check pointing and restart capability. A large number of jobs in the batch environment compete for the same resources, and administrators need a way to manage the jobs and handle resource conflicts. The organization then builds a job control interface into the JVM that processes the batch applications.

As batch loads increase even further, it becomes impossible to manually handle resource conflicts; as a result, the organization now requires a batch job management system that can optimally distribute workloads based on priorities and deadlines.

Custom middleware versus core business



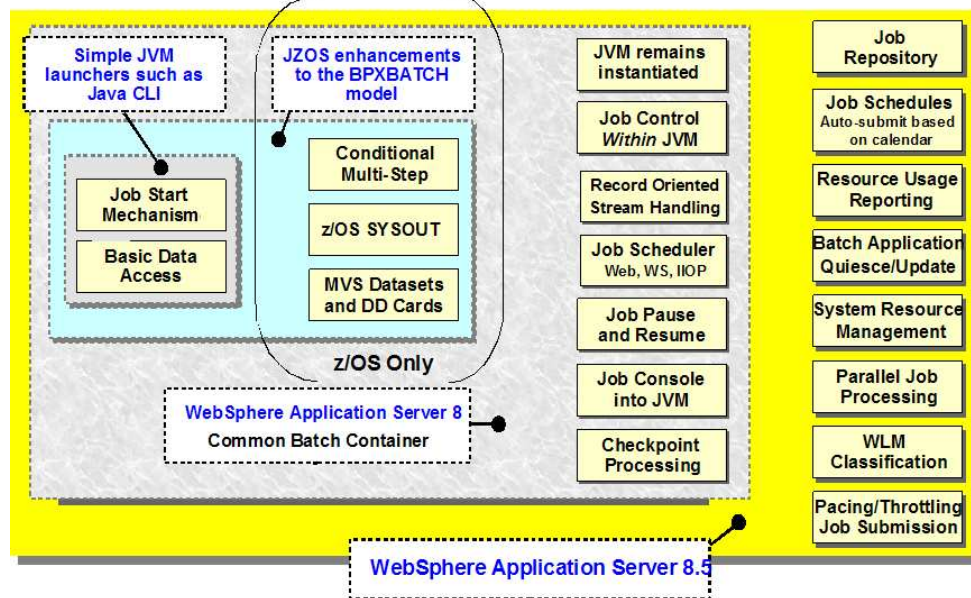
7

Batch overview

© 2012 IBM Corporation

Organizations are frequently tempted to solve tactical issues with custom middleware code; however, the middleware code requires extensive time to develop and maintain, which ultimately decreases an organization's ability to develop custom code that addresses core business objectives.

IBM Java Batch offerings



8

Batch overview

© 2012 IBM Corporation

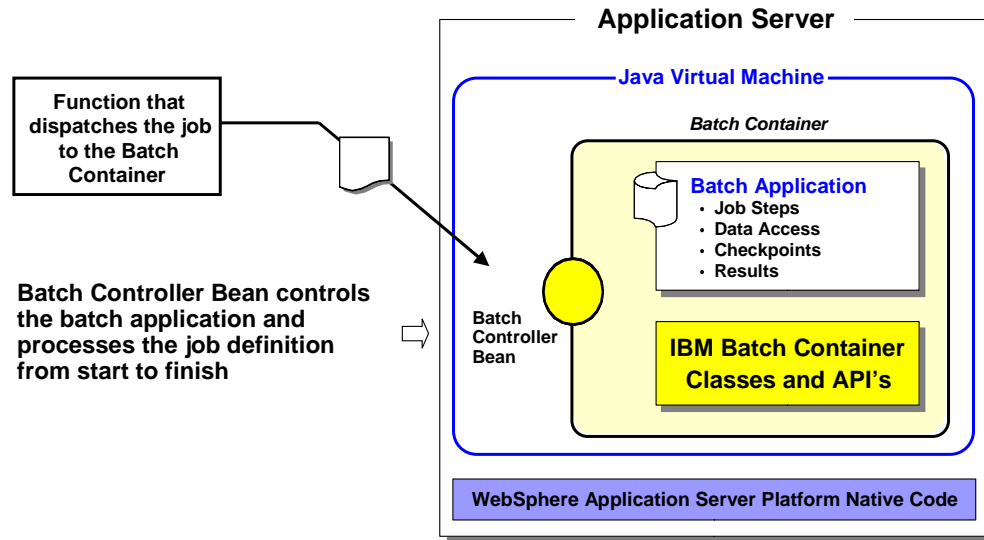
Java batch tasks can be managed in several different ways. For example, simple Java Virtual Machine (JVM) launchers can manage single step batch jobs that require basic data access.

On the z/OS platform, specifically, the Java Batch Toolkit for z/OS (also known as JZOS) enhances the BPXBATCH model by supporting conditional multi-step batch jobs with access to MVS datasets and use of Data Definition (DD) cards.

The prior version of WebSphere provided multi-step job support, a managed container for execution of batch jobs, a job control interface, job checkpoint and restart capability, and a batch application development framework. In WebSphere Application Server version 8.5, the capabilities of the prior release are expanded upon. For example: support for job repository and schedules, workload management, job usage reporting, batch application quiesce and update, parallel job support, and pacing and throttling of jobs.

Batch container

The Batch Container is the heart of the WebSphere Application Server Batch function



9

Batch overview

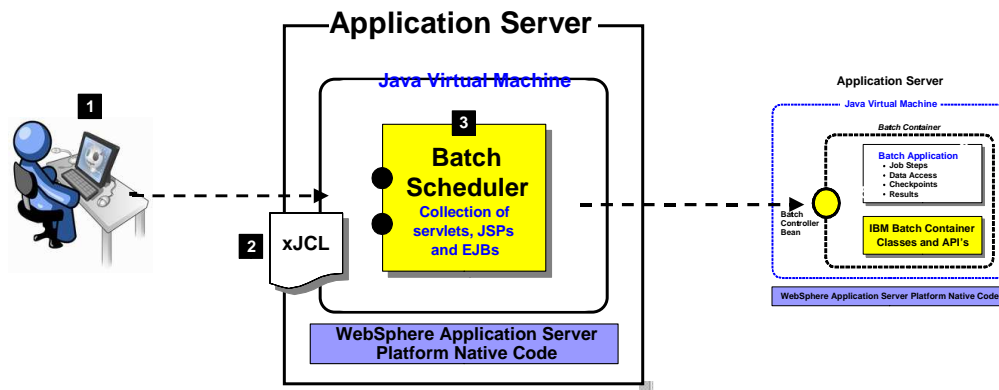
© 2012 IBM Corporation

The Batch Container is the heart of the batch application support provided in WebSphere Application Server. It runs a batch job under the control of an asynchronous bean, which can be thought of as a container-managed thread. The batch container ultimately processes a job definition and carries out the life cycle of a job.

The Batch Container provides these services:

- Check pointing, which involves resuming batch work from a selected position.
- Result processing, which involves intercepting and processing step and job return codes.
- And Batch data stream management, which involves reading, positioning, and repositioning data streams to files, relational databases, native z/OS datasets, and many other different types of input and output resources.

Batch components and workflow



1. Job is submitted
2. Job Control Definition is specified
3. Scheduler analyzes the request
4. Job is dispatched to batch endpoint
5. Batch endpoint begins execution
6. Batch application is invoked

10

Batch overview

© 2012 IBM Corporation

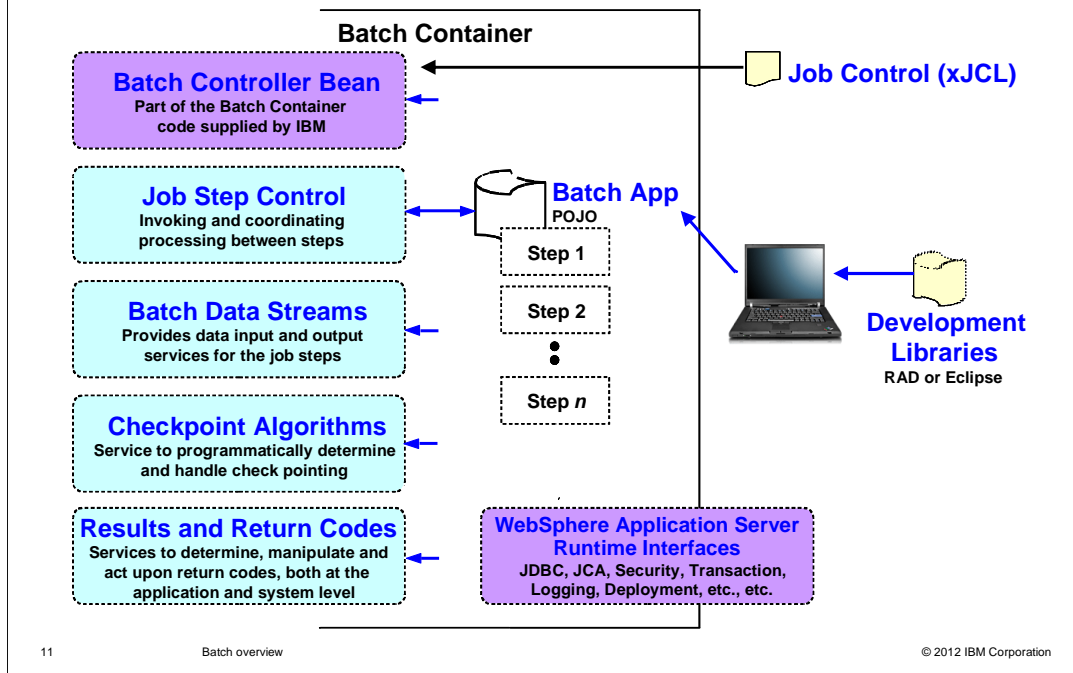
Batch jobs are submitted to the system using the Job Management Console or programmatically by way of Enterprise Java Beans (EJB), Java Message Service (JMS), or web services.

Each job is submitted in the form of an XML Job Control Language (xJCL) document. The Job Dispatcher then selects the best endpoint application server for job execution based on several different metrics.

The endpoint application server sets up the jobs in the Batch Container and executes the batch steps based on the definitions in the xJCL.

While the job is running, the Job Dispatcher aggregates job logs and provides lifecycle management functions such as start, stop, cancel, and so on.

Batch programming model



11

Batch overview

© 2012 IBM Corporation

The batch programming model consists of four principal interfaces, two of which are essential to building a batch application, and two which are optional and intended for advanced scenarios.

The first essential item is the batch job step, which defines the interaction between the batch container and the batch application.

The other essential item is the batch data stream. The batch data stream abstracts a particular input source or output destination for a batch application and defines the interaction between the batch container and a concrete BatchDataStream implementation.

An optional checkpoint policy algorithm defines the interaction between the batch container and a custom checkpoint policy implementation. A checkpoint policy is used to determine when the batch container will checkpoint a running batch job to enable a restart to occur after a planned or unplanned interruption. WebSphere Application Server includes two ready-to-use checkpoint policies.

An optional results algorithm defines the interaction between the batch container and a custom results algorithm. The purpose of the results algorithm is to provide the overall return code for a job. The algorithm has visibility to the return codes from each of the job steps. WebSphere Application Server includes one ready-to-use results algorithm.

Job control definition - xJCL

```
<?xml version="1.0" encoding="UTF-8" ?>
```

```
<job name="name" ... >
  <jndi-name>batch_controller_bean_jndi</jndi-name>
  <substitution-props>
    <prop name="property_name" value="value" />
  </substitution-props>
```

Roughly analogous
to the JOB card on z/OS

```
<job-step name="name">
  <classname>package.class</classname>
  <checkpoint-algorithm-ref name="chkpt"/>
  <results-ref name="jobsum"/>
  <batch-data-streams>
    <bds>
      <logical-name>input_stream</logical-name>
      <props>
        <prop name="name" value="value" />
      </props>
    </bds>
  </batch-data-streams>
</job-step>
```

A job step

Like the EXEC PGM=
statement in JCL

Similar to DD
statements

```
</job-step>
</job>
```

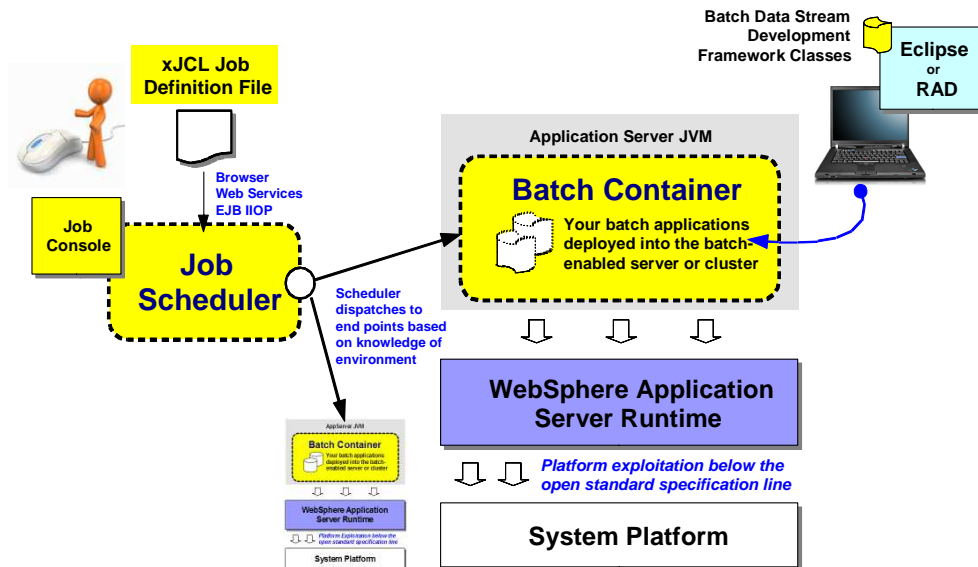
Jobs are expressed using an Extensible Markup Language (XML) dialect called XML Job Control Language, or xJCL.

The xJCL definition is very similar to the traditional JCL.

The xJCL definition of a job is not part of the batch application, but is constructed separately and submitted to the job scheduler to run. The job definition identifies which batch application to run, and its inputs and outputs. It also identifies which checkpoint algorithms and results algorithms to use.

The Job Scheduler uses information in the xJCL to determine where and when the job should run.

Batch environment review



13

Batch overview

© 2012 IBM Corporation

The Job Scheduler provides the job management functions such as submit, cancel, restart, and so on. It maintains a history of all job activity, including waiting jobs, and running jobs, and completed jobs. The job scheduler is hosted in a WebSphere Application Server or a server cluster.

Jobs are described using job control language called XML Job Control Language (xJCL), which identifies the batch application to run and its inputs and outputs. The Batch Container provides the execution environment for batch jobs. There can be multiple Batch Containers in a WebSphere cell. Batch applications are regular WebSphere Java Enterprise Edition (Java EE) applications, deployed as Enterprise Archive (EAR) files.

What is new in this release

WebSphere Application Server 8.5 has updated its Batch support to include new functionality in these areas

- Programming model enhancements
- Operational enhancements

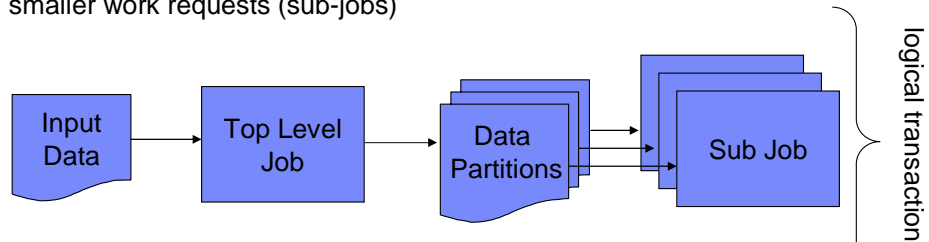
WebSphere Application Server 8.5 contains enhancements to the Batch programming model and operational features.

Programming model enhancements

This section describes the Batch programming model enhancements included in WebSphere Application Server 8.5

Parallel Batch – parallel Job Manager

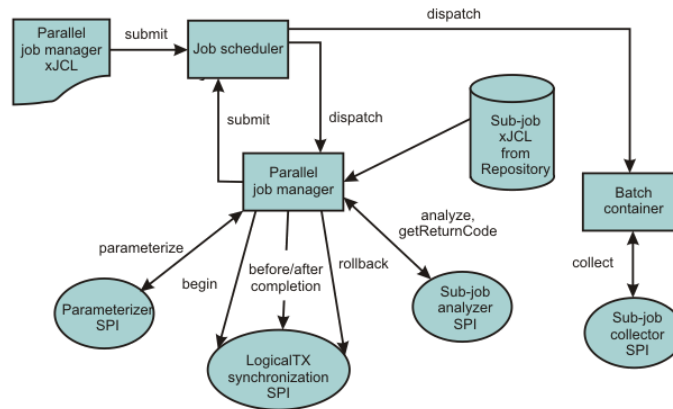
- Parallel Job Manager (PJM) decomposes a large work request into many smaller work requests (sub-jobs)



- PJM then provides operational control over the sub-jobs executing across the job endpoints – note sub-jobs are clones
- Administrator only manages the top-level (logical) job; PJM, in the background, manages the sub-jobs

The Parallel Job Manager provides support for building transactional batch applications as a job, and then divides the job into subordinate jobs. The subordinate jobs can run independently and in parallel. The Parallel Job Manager is used to submit and manage the transactional batch jobs.

Parallel Batch - PJM Application Programming Interfaces



17

Batch overview

© 2012 IBM Corporation

The Batch programming model has been updated to provide APIs for the Parallel Job Manager.

The purpose of the Parameterizer API is to divide the top-level job into multiple sub-jobs. The Parameterize API determines the number of sub-jobs to create, and the input properties passed to each sub-job.

The Synchronization API gives you control during the various life cycle stages of the logical transaction. For example, the Begin, beforeCompletion, and afterCompletion life stages. You can use these control points to roll back the logical transaction if necessary.

The SubJobCollector API collects information related to a subordinate job execution.

The SubJobAnalyzer API is used to analyze information collected previously by using the SubJobCollector API. In a typical implementation, the SubJobAnalyzer API is used to aggregate information obtained from all sub-jobs to determine the consolidated return code for the top-level job.

Parallel Batch – xJCL examples

xJCL:

```
<job name=... >
  <step name=... >
    <run instances="multiple"
      jvm="single" />
    ...
  </step>
  <step name=... >
    <run instances="multiple"
      jvm="multiple" />
    ...
  </step>
</job>
```

- Ability to specify parallelization configuration per job step
- Supports any step type
- Step might be multi-threaded or multi-process (that is, multiple JVMs)

Multi threaded step

Multi process step

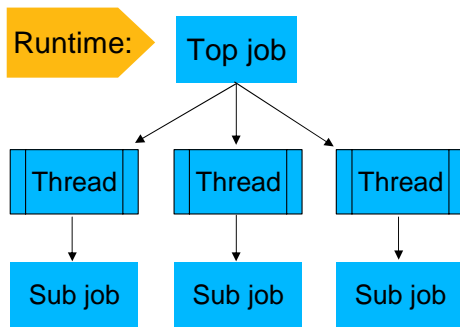
Parallel step support allows you to run some steps of a job as parallel jobs themselves. This approach is different from running two steps concurrently (a capability that is not currently supported). A parallel step can be thought of as a top level job, with the sub-job xJCL generated using the step's own xJCL snippet.

Parallel Batch - multi-threading

xJCL:

```
<job name=... >  
  <run instances="multiple"  
    jvm="single" />  
  <step name=... >  
    ...  
  </step>  
</job>
```

- Option to run parallel job on multiple threads
- Parallel Job Manager local optimization
- Alternative to running parallel job across multiple JVMs
- Optimizes shorter running sub jobs



19

Batch overview

© 2012 IBM Corporation

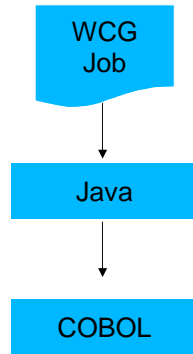
WebSphere Batch allows the running of parallel jobs and their sub-jobs in the same JVM if needed. This capability is especially useful when sub-jobs are typically very short in duration and the overhead of distributing the sub-jobs across servers is much larger compared to the actual work to be completed.

COBOL support

Value

Reuse existing COBOL in new Java Batch Applications!

Batch Application



- Call standard COBOL (z/OS only) modules from Java on same thread in same process
- Java and COBOL run in same transaction scope
- DB2 connections managed by WebSphere Application Server can be shared with COBOL
- COBOL working storage isolation per job step
- RAD tools for Java call stub generation

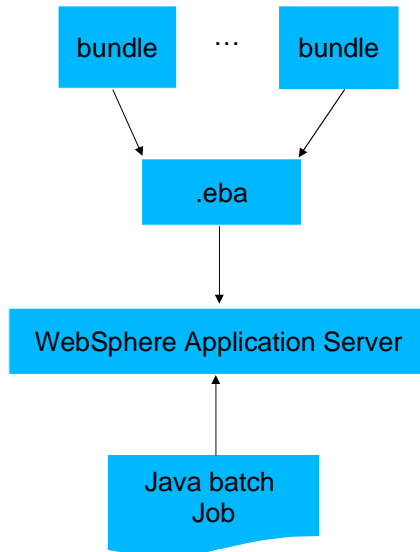
20

Batch overview

© 2012 IBM Corporation

COBOL has been the prevalent language to run batch-style workloads in the past, and even today there is a large existing base of COBOL code. On the z/OS platform, Java batch adds support to easily call into existing COBOL assets to run modern batch workloads.

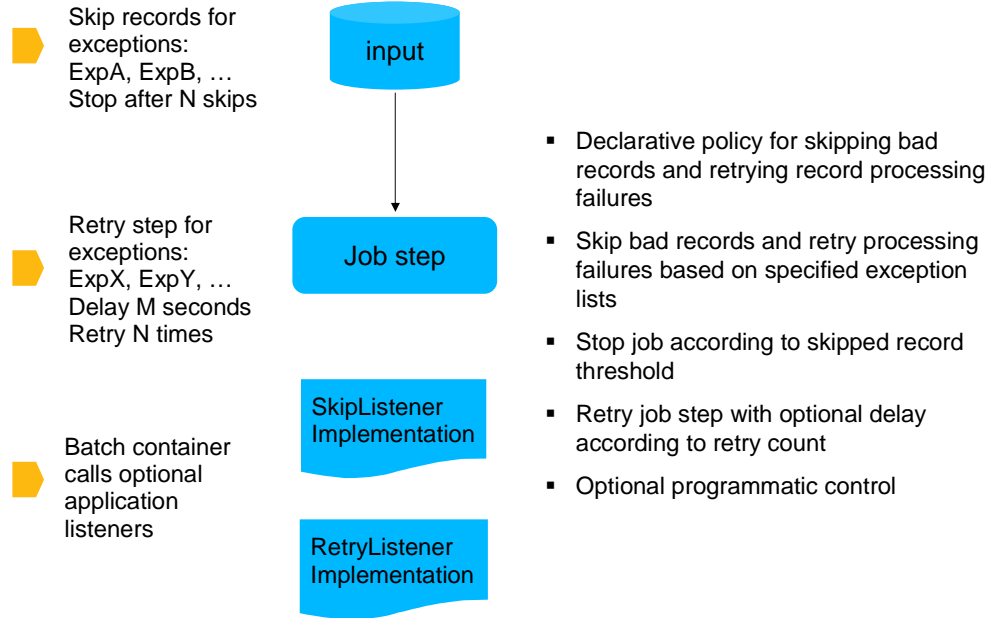
OSGi batch applications



- Enables use of OSGi for Java batch application development
- Full batch programming model available to OSGi framework
- Supports standard and blueprint bundles
- Enterprise Bundle Archive deployment

WebSphere Batch enables you to re-use your existing blueprint OSGi applications to run batch workloads. Batch artifacts are exported as blueprint services which are invoked by the batch container during the job life cycle.

Record processing policy



22

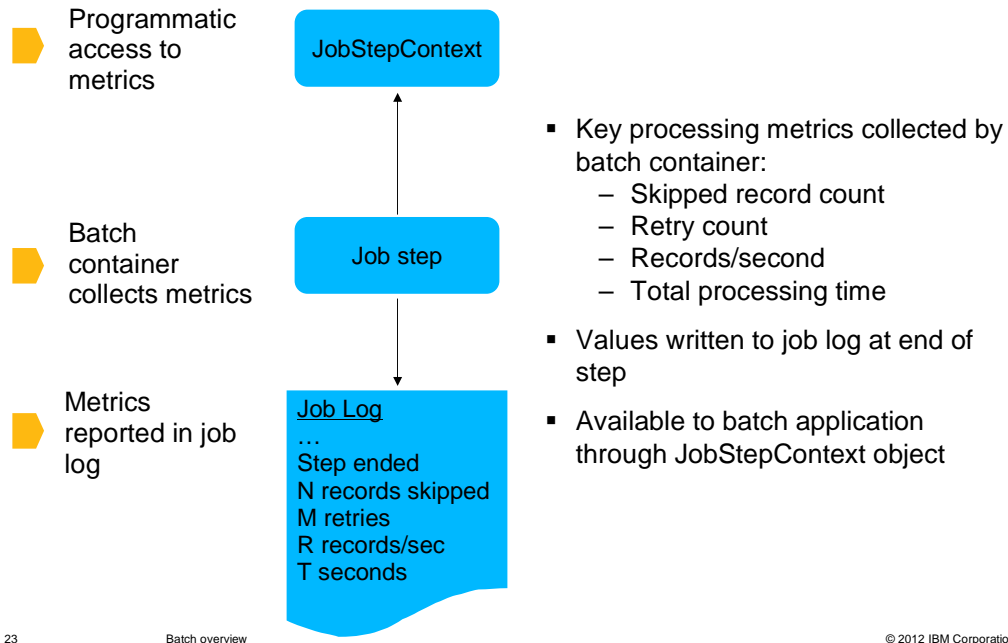
Batch overview

© 2012 IBM Corporation

WebSphere Batch includes two different declarative record processing policies, skip records and step retry. The skip records policy covers the input stream, and the step retry policy covers record processing and the output stream.

The step retry policy is NOT related to skipped records. It will automatically retry a failed step since its last checkpoint.

Record metrics



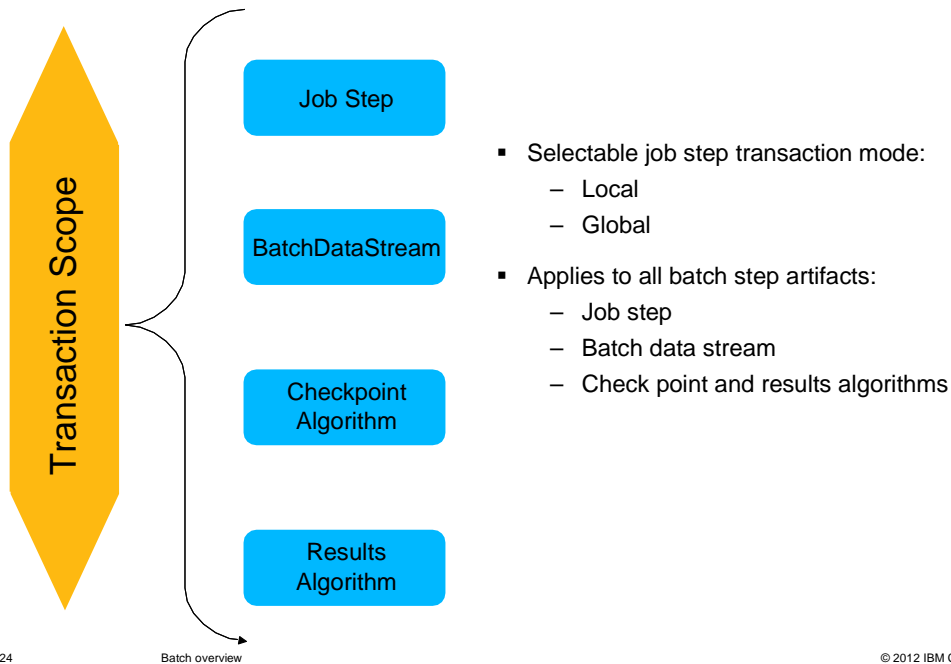
23

Batch overview

© 2012 IBM Corporation

The record metrics feature provides convenient statistics about the records processed by each job step. Key statistics include the skipped record count, retry count, records processed per second, and total processing time. The values are written to the job log at the end of a step, and are available to a batch application through the JobStep Context object.

Configurable transaction model



WebSphere Batch adds support for a local transaction mode of operation. In this mode, no global transaction is started; the connection object is shared between the container and your code through the JobStep Context object. Local transaction mode significantly improves performance by reducing the overhead required to manage global transactions. This mode is especially useful if your data is located on the same database system as the Job Scheduler tables. In addition, large cursors are less likely to be closed when transactions are committed.

Batch data stream timeouts

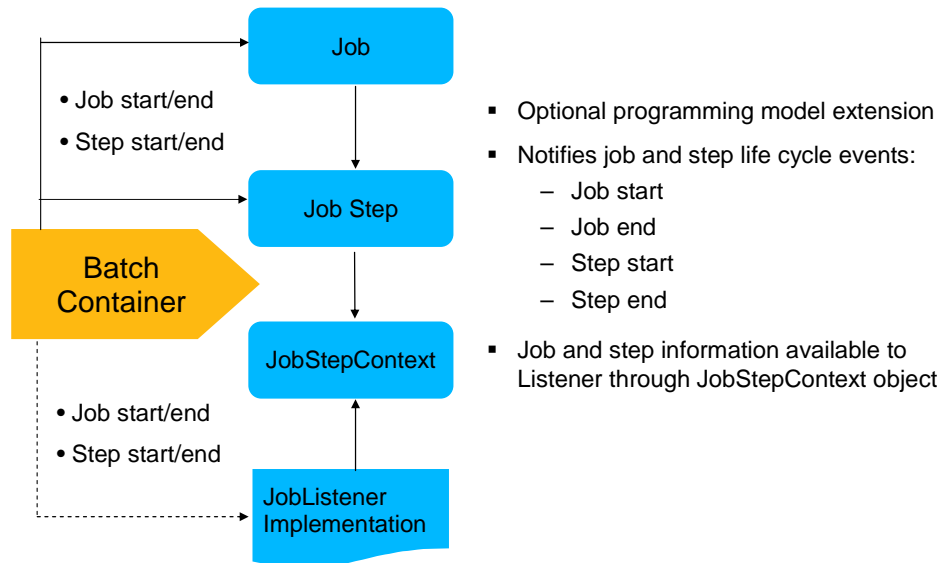
xJCL:

```
<job name=... >
  <step name=...>
    <batch-data-streams timeout=N>
      <bds>...</bds>
      <bds>...</bds>
      <bds>...</bds>
    </batch-data-streams>
  </step>
</job>
```

- Problem
 - Job steps need short transaction timeouts
 - Batch data streams (large results sets) need long timeouts
- Solution
 - Configurable batch data stream timeout
 - Can be specified per job step

WebSphere Batch allows you to configure a batch data stream timeout for each job step; as a result, you can now define short timeouts for transactions, and long timeouts for BatchDataStreams.

Job and step listener



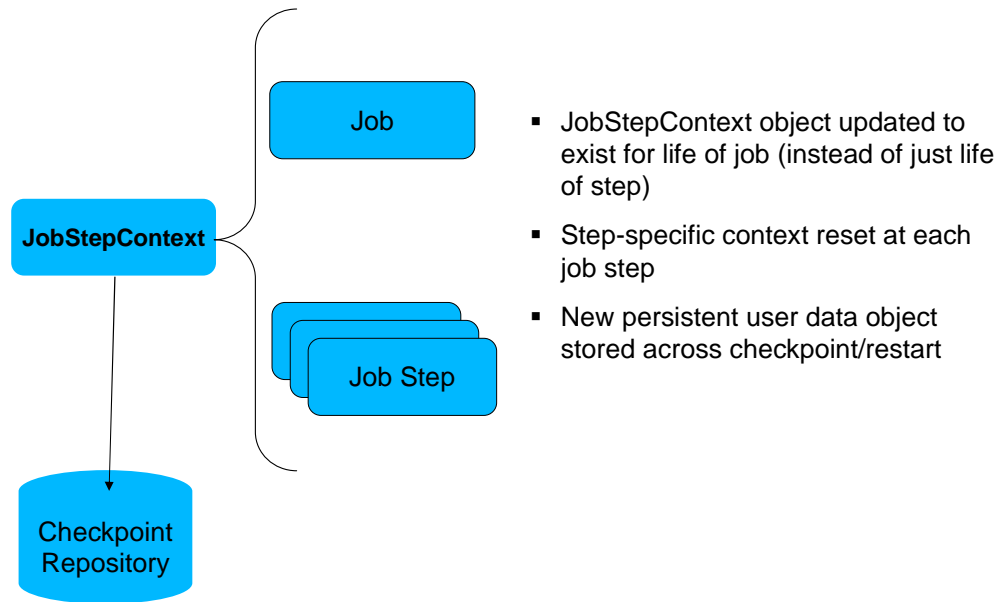
26

Batch overview

© 2012 IBM Corporation

The job and step listener allows application developers to add cross-cutting initialization and clean up routines to jobs and steps. The listener implementation is specified in a new job-level xJCL element. The batch container invokes the listener at specific events: before job start, after job start, before step start, and before step end. Data can be accessed through the JobStep Context object.

Persistent job context



27

Batch overview

© 2012 IBM Corporation

Job and Step context information are available with the updated WebSphere Batch support.

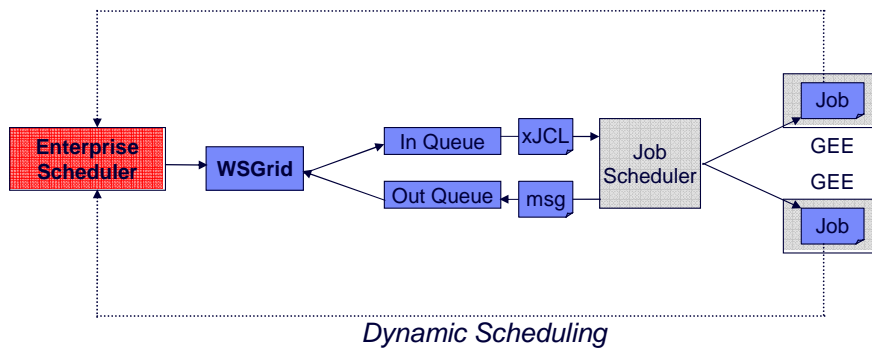
The JobStep Context information is available to all job types and exists for the life of the job, as opposed to the life of a step.

Step specific context information is reset at each job step. Any new step-level user data is persisted to a database at each checkpoint so that it is available at job restart.

Operational enhancements

This section describes the Batch operational enhancements included in WebSphere Application Server 8.5

Integration with an external scheduler

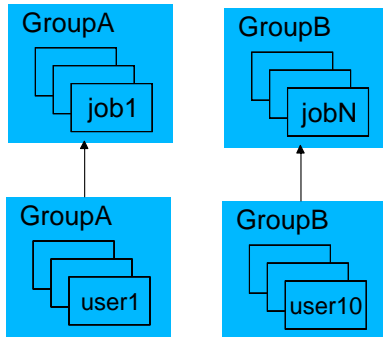


- Central enterprise scheduler, Job Scheduler told what to execute.
- Jobs and commands are submitted from Enterprise Scheduler to Job Scheduler by way of WSGRID
- Jobs can dynamically schedule to Enterprise Scheduler by way of EJB interface

WebSphere Application Server 8.5 Batch provides an integration capability with external workload schedulers, such as Tivoli Workload Scheduler.

An integration layer known as WSGrid enables Tivoli Workload Scheduler (and similar products) to dispatch and monitor batch activities. This integration layer can also be used by vendor workload scheduler products.

Group security (1 of 2)



- User in GroupA can manage jobs that belong to GroupA
- User in GroupB can manage jobs that belong to GroupB

- Group-level security for jobs
 - Leveraging WebSphere security functions
 - Enable in custom properties
 - Configure to work with or without security roles
- Users have access to jobs based on group membership
- Job management console views are customized based group security settings

30

Batch overview

© 2012 IBM Corporation


In the prior version of WebSphere Batch, access to job management tasks and information is controlled through role-based authentication, where each user must be assigned the Iradmin, Irsubmitter, or Irmonitor role. This model is still supported. In an effort to improve flexibility and streamline user management tasks, a new security model has been introduced that enables a group of users to operate on a common subset of jobs. In the graphic, a user from GroupA can manage jobs that belong to GroupA, and a user from GroupB can manage jobs that belong to GroupB. If you belong to both GroupA and GroupB, you can manage jobs from both groups.

Group security can be configured in combination with role-based security. When this approach is used, you can perform a job-related action if and only if you and the job are members of the same group, and your role permits the job action.

The default security model continues to be role-based. To change the security model, you must define several custom properties at the level of the job scheduler.

Group security (2 of 2)

Job management console

Select	Job ID	Submitter	Last Update	State 	Node	Application Server	Group
<input type="checkbox"/>	Echo:00035	vignola	2010-02-04 11:34:53.953	Ended	VIGNOLA-T60Node01	server1	JSYSDFLT
<input type="checkbox"/>	ParallelJobManager:00083	vignola	2010-05-14 01:06:32.890	Restartable	VIGNOLA-T60Node01	server1	JSYSDFLT
<input type="checkbox"/>	ParallelJobManager:00084	vignola	2010-05-14 01:09:37.593	Restartable	VIGNOLA-T60Node01	server1	JSYSDFLT
<input type="checkbox"/>	ParallelJobManager:00085	vignola	2010-05-14 01:19:15.953	Restartable	VIGNOLA-T60Node01	server1	JSYSDFLT

When group-based security is enabled, the job management console includes the group associated with each job. It is also possible to filter this view by group name.

Scheduling a job

Specify information for creating a schedule.

Step 1: Create schedule
Step 2: Specify job
Step 3: Confirm create schedule

Create schedule
Specify the name of the schedule to create. Specify the start date and time for the job to first run.

* Name:

* Start date (yyyy-MM-dd): 2012-03-14

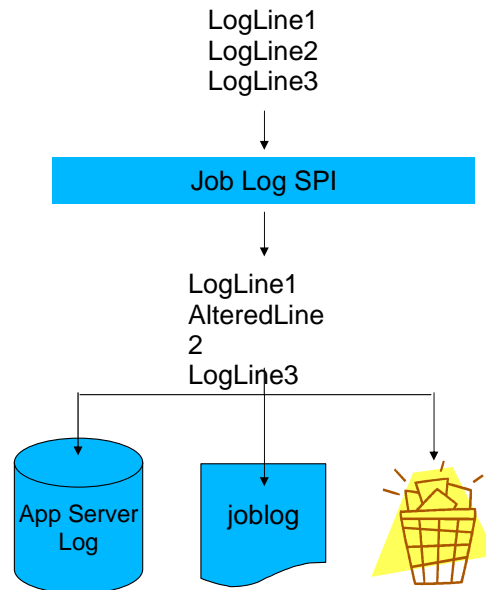
* Start time (HH:mm:ss): 18:42:05

* Interval: Daily

< Back Next > Finish Cancel

The WebSphere Batch support has been updated to include the capability to schedule jobs based on calendar and time.

Job log system programming interface (SPI)



33

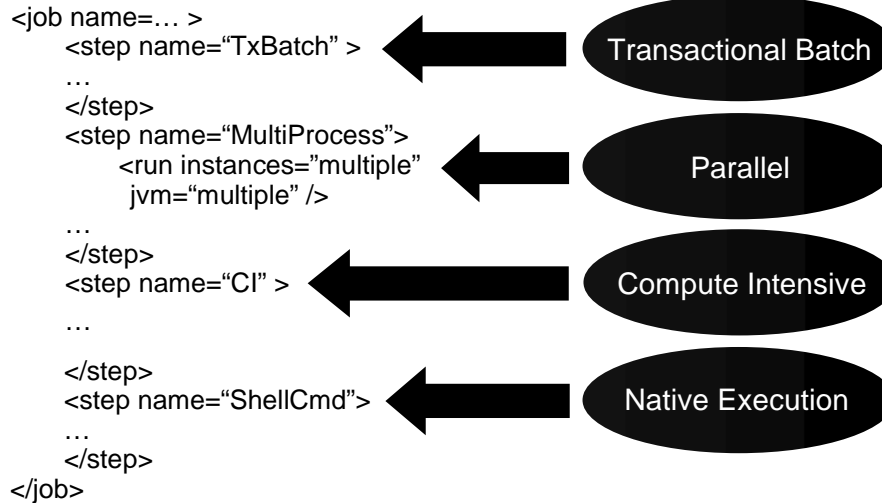
Batch overview

© 2012 IBM Corporation

The job log system programming interface (SPI) allows system-wide customization and control of job log content and destinations. You can use the job log SPI to direct the logging information to only the job log, only the WebSphere Application Server log, both logs, or neither log. The job log SPI can also be used to modify the content of the log line, as shown in the graphic.

Mixed steps

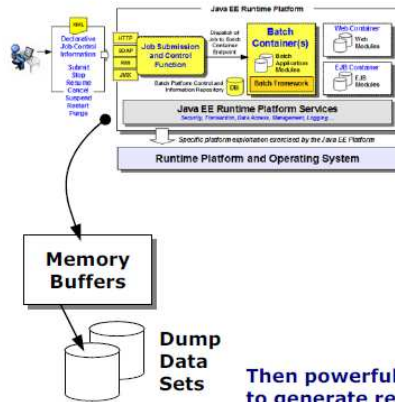
xJCL:



WebSphere Batch allows a job to contain different types of steps. For example, a job can include a mixture of batch, compute intensive, and native job steps. This capability adds immense flexibility to the batch programming model, as different parts of a job can be run whether they are transactional or not.

Java Batch and SMF

SMF is a fast activity recording subsystem on z/OS. The Java Batch support within WebSphere Application Server 8.5 exploits this with its own SMF record:



Information in SMF 120.20 record:

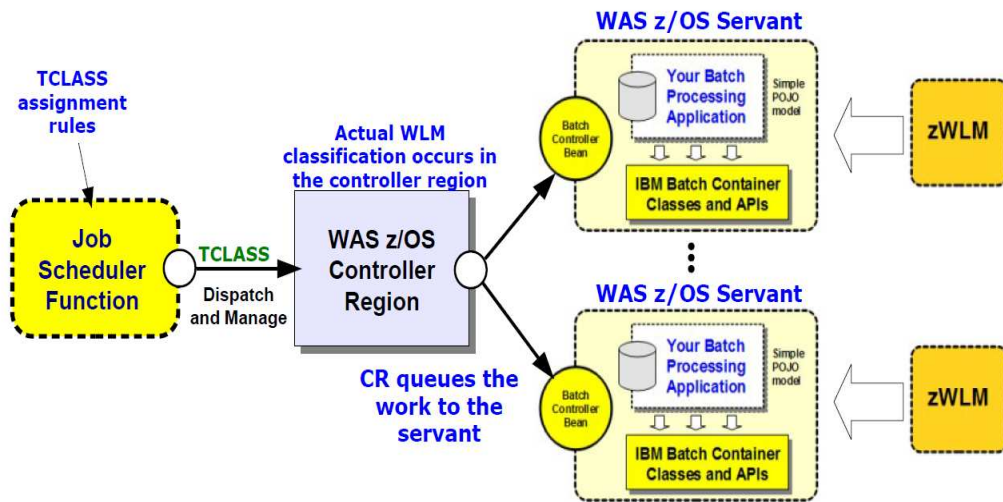
Job identifier
Job submitter
Final Job state
Server
Node
Accounting information
Job start time
Last update time
General CPU
zAAP CPU

Then powerful industry analysis tools can be used to generate reports and determine usage for reasons such as capacity planning and chargeback

SMF is a facility of the z/OS operating system that provides a high-speed activity recording mechanism for programs that want to use it.

WebSphere Application Server for z/OS has an SMF record referred to as SMF 120 subtype 9. The Batch support within WebSphere Application server 8.5 also takes advantage of SMF, and writes its own SMF 120 subtype 20.

WLM and job classification

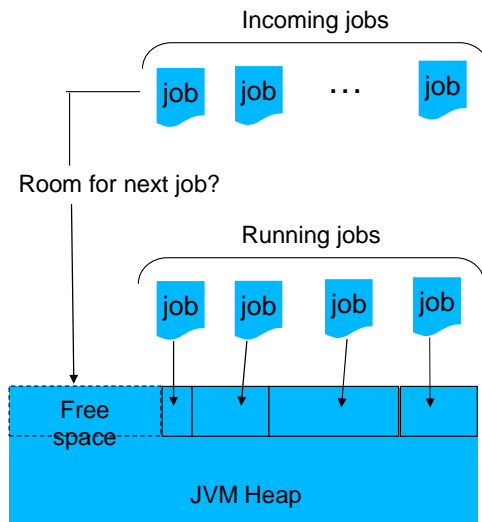


Classify batch **jobs** separately, queue them to separate servants, and allow WLM to manage each according to goals

WebSphere Batch makes use of the WLM classification capabilities of the underlying WebSphere Application Server for z/OS product.

This provides a means by which requests can be isolated to separate servant regions, where WLM can then apply resource allocation measures.

Memory overload protection



- Protection against over-scheduling jobs to an application server
- Batch container monitors job memory demand against available JVM heap space
- Prevents Java OutOfMemoryError
- Automatic real time job memory estimation with declarative xJCL override

xJCL: <job name=... [memory=N] ... >

37

Batch overview

© 2012 IBM Corporation

WebSphere Batch provides a memory overload protection feature that prevents the mix of jobs on an endpoint server from causing a Java heap out of memory condition. A new xJCL attribute, `memory`, allows you to declare the amount of memory required by a job.

The batch container monitors job memory requirements against its available heap space. If a job enters the batch container, and the memory requirements of the job exceed the available free space, the job is placed in wait state. As other jobs exit the BatchContainer and resources are released, the held job is again evaluated against the available free space.

This feature does not affect which endpoint server the job scheduler selects to process the job. It ensures that the batch container on the selected endpoint does not run into an out of memory condition because an excessive number of memory-intensive jobs are being processed concurrently.

Feedback

Your feedback is valuable

You can help improve the quality of IBM Education Assistant content to better meet your needs by providing feedback.

- Did you find this module useful?
- Did it help you solve a problem or answer a question?
- Do you have suggestions for improvements?

Click to send email feedback:

mailto:iea@us.ibm.com?subject=Feedback_about_WASv85_Batch_Overview.ppt

This module is also available in PDF format at: [../WASv85_Batch_Overview.pdf](#)

You can help improve the quality of IBM Education Assistant content by providing feedback.



Trademarks, disclaimer, and copyright information

IBM, the IBM logo, ibm.com, DB2, System z, Tivoli, WebSphere, and z/OS are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of other IBM trademarks is available on the web at "[Copyright and trademark information](http://www.ibm.com/legal/copytrade.shtml)" at <http://www.ibm.com/legal/copytrade.shtml>

THE INFORMATION CONTAINED IN THIS PRESENTATION IS PROVIDED FOR INFORMATIONAL PURPOSES ONLY. WHILE EFFORTS WERE MADE TO VERIFY THE COMPLETENESS AND ACCURACY OF THE INFORMATION CONTAINED IN THIS PRESENTATION, IT IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. IN ADDITION, THIS INFORMATION IS BASED ON IBM'S CURRENT PRODUCT PLANS AND STRATEGY, WHICH ARE SUBJECT TO CHANGE BY IBM WITHOUT NOTICE. IBM SHALL NOT BE RESPONSIBLE FOR ANY DAMAGES ARISING OUT OF THE USE OF, OR OTHERWISE RELATED TO, THIS PRESENTATION OR ANY OTHER DOCUMENTATION. NOTHING CONTAINED IN THIS PRESENTATION IS INTENDED TO, NOR SHALL HAVE THE EFFECT OF, CREATING ANY WARRANTIES OR REPRESENTATIONS FROM IBM (OR ITS SUPPLIERS OR LICENSORS), OR ALTERING THE TERMS AND CONDITIONS OF ANY AGREEMENT OR LICENSE GOVERNING THE USE OF IBM PRODUCTS OR SOFTWARE.

© Copyright International Business Machines Corporation 2012. All rights reserved.