WebSphere Application Server & IBM Operational Decision Manager:
Adapting mainframe applications for mobile workloads

Speaker Name and Title

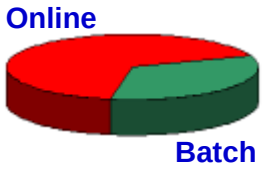
Agenda

- Impact of mobile workloads on enterprises
- Application modernization – an incremental approach
- WebSphere Java Batch Overview
- Operational Decision Manager Overview
- Bringing Batch and Decision Processing Together
- Summary and Wrap-Up

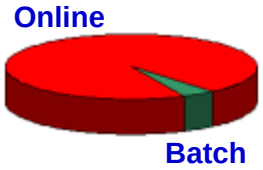
Impacts of mobile workloads

The proliferation of mobile access is driving increased transactions putting stress on traditional enterprise processing

In the past ...



Today ...



24 x 7 x 365 Access

Users of your online systems expect availability at all hours.

Users from other parts of the world means availability is expected around the clock.

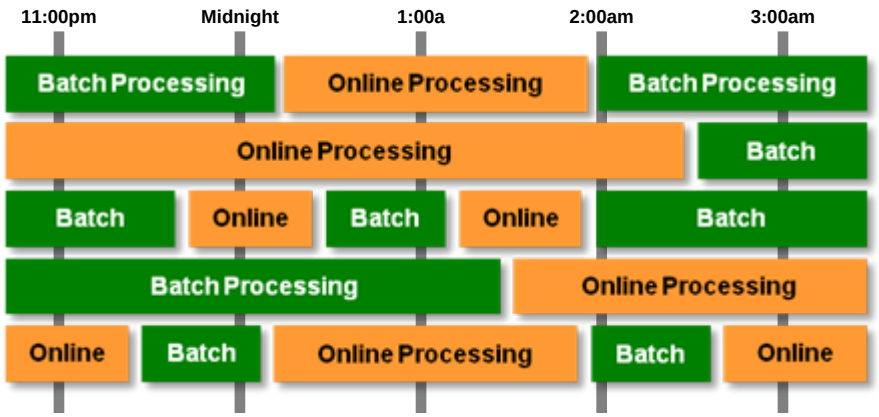
Mobile Users

Users are no longer tied to a desk and a computer. Today users have access to mobile computing devices that are with the user wherever they may be. Day or night, home or office.

Need to process data and decisions faster

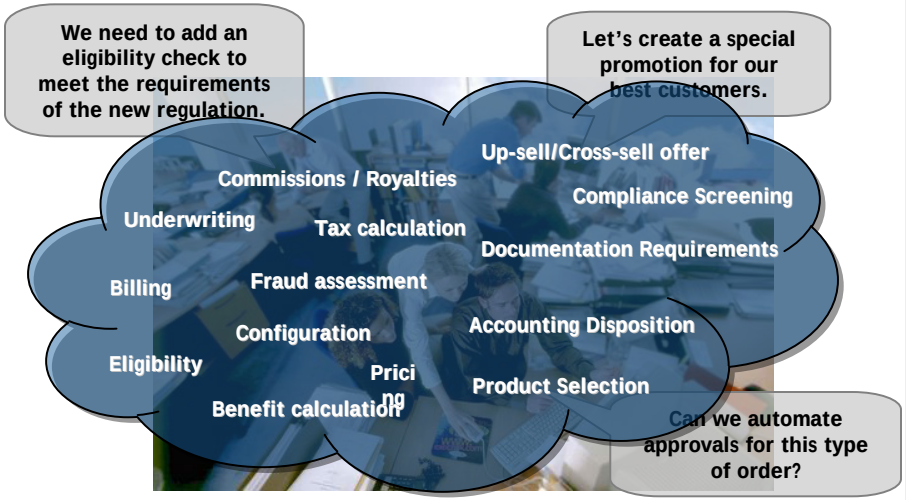
Organizations need to be able to process data and make decisions at the speed of business

Integrated Batch and Online



Windows of time which used to be dedicated to batch processing are shrinking. Online and Batch execution within a common runtime, enables managing workloads by priority to respond to capacity fluctuations and meet SLAs.

Automated Business Decisions

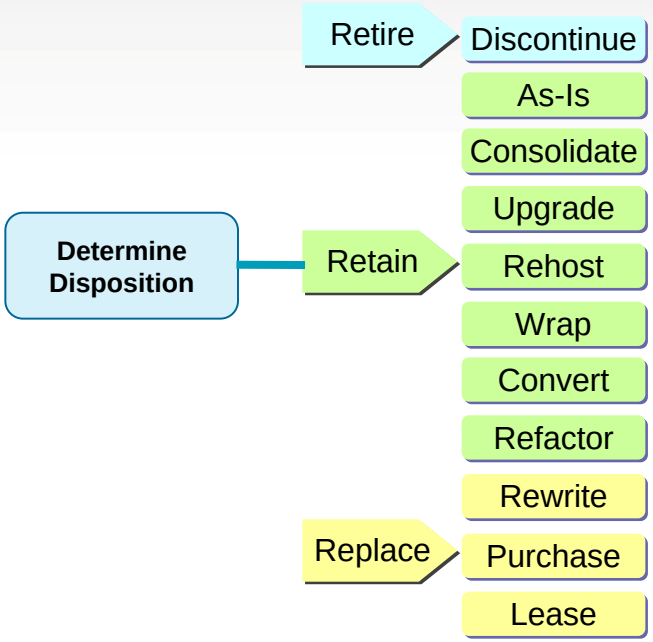


Decisions are everywhere and change frequently

Businesses need to be able to quickly make decisions to respond to customers. This requires the ability to define, execute, and govern decisions in an automated, flexible, yet tightly controlled manner.

Application modernization
what are the modernization options

Decision based modernization offers incremental business agility



- Maintain, Consolidate, Update
- Expand/modernize user access
- SOA modularization/reuse (includes Decision Mgmt based Modernization)
- Modernize language for multiple platforms
- Refactor applications to leverage new Batch execution technologies
- Decision Mgmt based Modernization

Incremental Adoption: Evolution not revolution

A primary benefit of WebSphere solutions is the ability to introduce modernization capabilities in a controlled and incremental fashion

- Organizations must be able to start small and grow as part of application modernization
- In many cases it would not be feasible to replace significant portions of the existing processing at a single time
- Integration with existing processes, applications, operating system services and technologies allows capabilities to introduced incrementally, while still providing immediate value
- Keep the processing co-located with the mainframe applications and data, ensures performance, availability and security

WebSphere Batch Overview

Batch Processing Modernization

- Business need: Consolidation of online workloads with the processing of traditional batch workloads

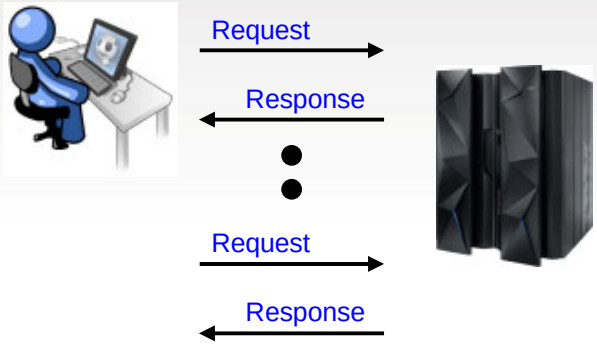
- Drivers:
  - Time sensitivity
    Overnight batch windows are not completing as planned
  - Cost savings
    Moving from COBOL to Java enables offload of workload to specialty engines
    Removal of homegrown batch infrastructures
  - Leverage availability of Java development skills
  - Enable shared services for code re-use and easier maintenance

- Modernization with WebSphere Java batch: Incremental integration of Java batch processing with existing enterprise batch to enable concurrent online and batch workload processing.

What is Batch (or Bulk) Processing?

Many definitions exist ... They have in common that there is minimal human interaction and expectation of results at a future time rather than immediately.

Online Processing



In general:

- Interaction is *one-for-one* ... that is, request with matching response
- Expectation is for response to follow request in a *near-immediate* time frame
- **Examples:** Inventory query, Website shopping transaction, eBanking account withdrawal

Bulk Processing

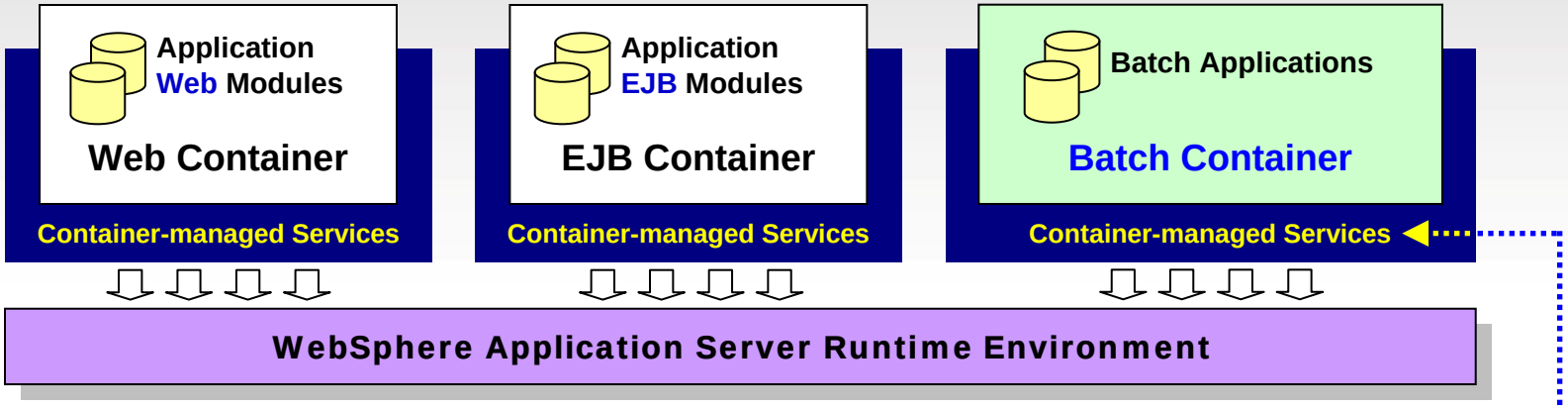
In general:

- Interaction is *one-for-many*... that is, initial request results in many results from processing
- Expectation is for results to finish within some determined *non-immediate* time frame
- **Examples:** Month end tax calculation, Period end statements and reports, Data transformation, Data analysis

High — Interaction and Responsiveness — Low

Batch is fundamental to the WAS Runtime

Think of IBM WebSphere Java Batch function as a "batch container" operating alongside the other containers of WAS itself:

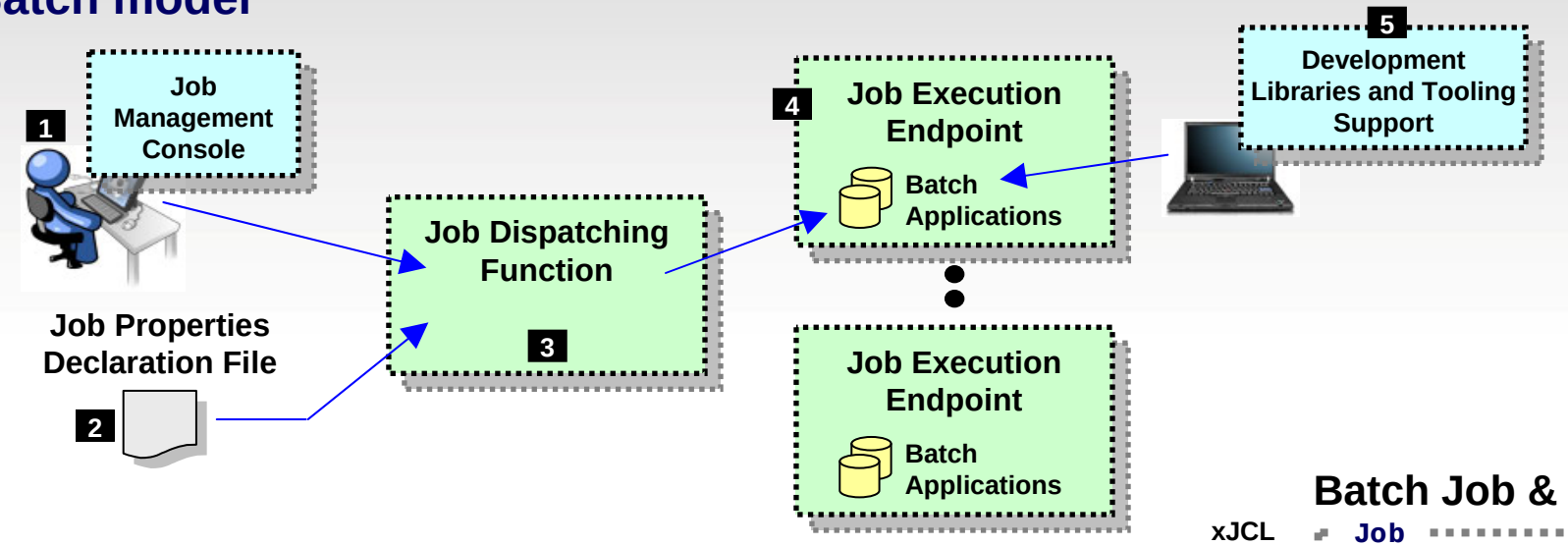


Batch job dispatching and management system	Job resiliency services (skip record, step retry)
Data record read and write support services	Parallel job management and execution services
Checkpoint and job restart services	COBOL module call services

Batch Services

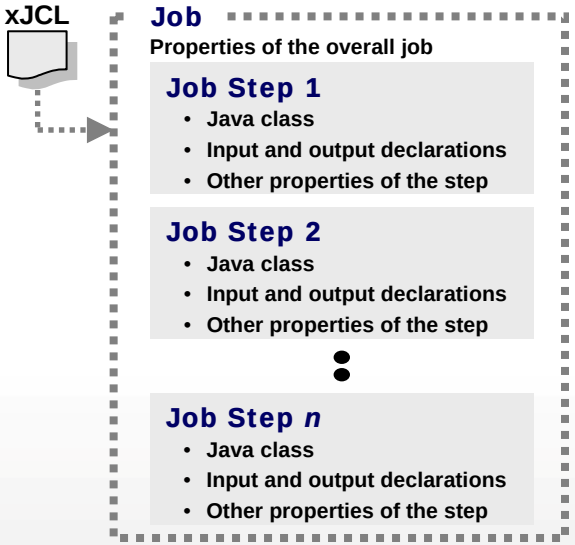
Batch Management and Execution Model

This picture illustrates some of the key components of the WebSphere Java Batch model



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

Batch Job & Job Steps



The Job Definition

Job Declaration File - xJCL

Conceptually, xJCL is just like normal // JCL -- it describes the job to be run and the context in which the job is to operate. The difference is xJCL is written in XML:

```
<?xml version="1.0" encoding="UTF-8" ?>
<job name="MyJob" ... ">
  <substitution-props>
    <prop name="inputDataStream" value="/tmp/input-text.txt" /
    <prop name="outputDataStream" value="/tmp/output-text.txt"
    <prop name="checkPoint" value="10" />
  </substitution-props>

  <job-step name="MyStep1">
    <classname>com.ibm.ws.batch.MyStep1</classname>
    <batch-data-streams>
      <bds>
       <logical-name>inputStream</logical-name>
         <props>
           <prop name="PATTERN_IMPL_CLASS" value="com.ibm.webs
           <prop name="FILENAME" value="${inputDataStream}" />
         </props>
       </bds>
    </batch-data-streams>
  </job-step>
     :
  <job-step ...>
  </job-step>
</job>
```

Comparable to the JCL "JOB" card. It sets job-level information along with some substitution properties Substitution properties may be overridden at submission time.

Comparable to the JCL step. It names the Java class that implements the step function.
The "Batch Data Stream" implementation is specified. In this example it defines the class that implements the input stream.

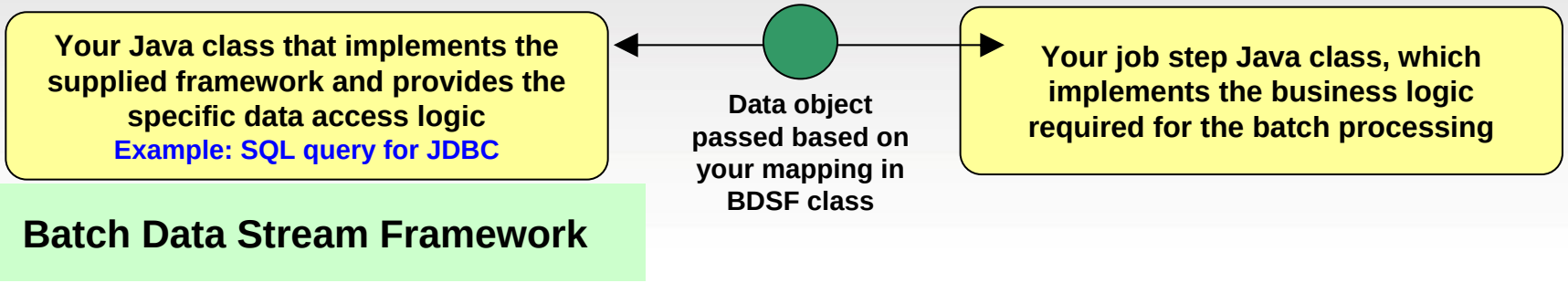
If job consists of more steps they are specified in sequence.

This tells WebSphere Java Batch *what* to run and *how* to run it

Note: this is a trimmed version of actual xJCL

Batch Data Stream Framework (BDSF)

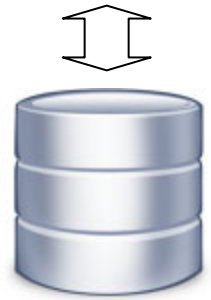
A key service provided by the batch container – it abstracts data read and write operations so your code may focus on the business logic:



Batch Data Stream Framework

Supplied "patterns" for data access:

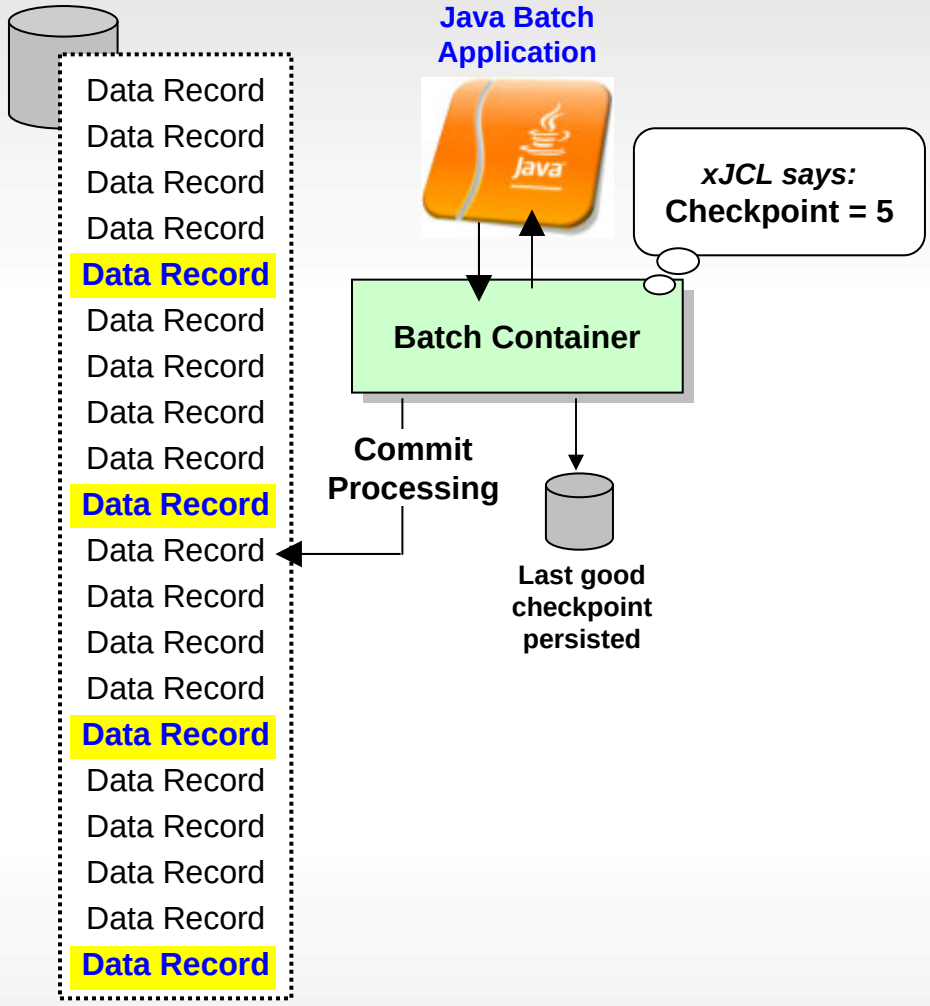
- **JDBC read or write operations**
- **JPA read or write operations**
- **File read or write operations**
- **z/OS Data Set read or write operations**



- Batch Data Stream retrieves result set from data persistence store (DB, file, etc.)
- Batch Data Stream maps data fields to data object
- For each record in result set, BDSF invokes your job step, passing a data object mapped to your specifications
- Your job step code stays focused on business logic, not Java stream handling and data object formatting

Transactional Checkpoint Processing

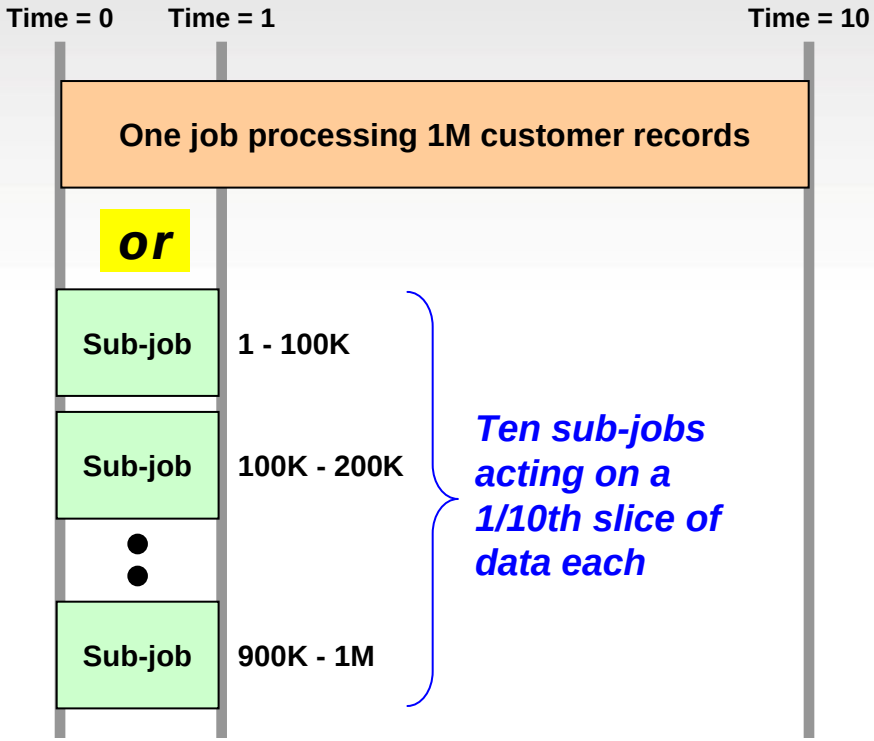
The batch container provides the ability to checkpoint at intervals based on either record count or time. The container keeps track of last checkpoint.



- Checkpoint interval (record or time) specified in the xJCL
- This is a function of the batch container, *not* your application code
- As checkpoint intervals are reached, container commits and records the checkpoint attained
- In the event of a failure, job may be restarted at the last good checkpoint
- Set the checkpoint interval based on your knowledge of balance between recoverability and efficiency

Parallel Job Manager

The Parallel Job Manager (PJM) provides a way to "parameterize" logic so parallel sub-jobs may act on a slice of the overall batch job data:



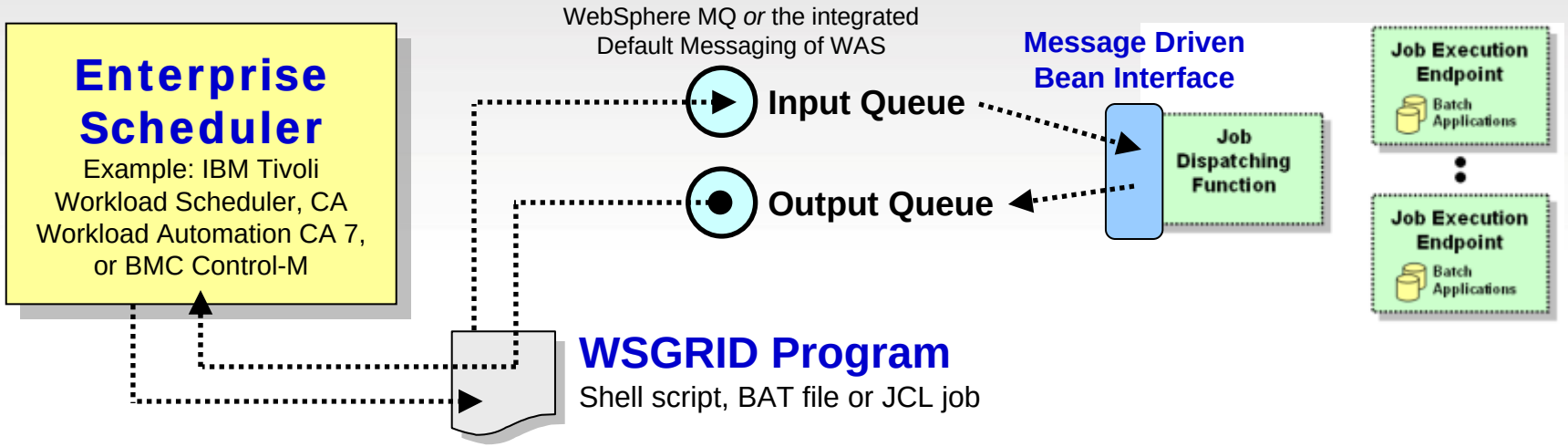
- xJCL specifies whether job is to be run in parallel, and if so how:
  - One JVM, multiple threads
  - Multiple JVMs
- Your "parameterizer" code is called at start so data range may be segmented into sub-job slices
- Job is submitted, then PJM dispatches "sub-jobs" to act on each data range
  - "Parameterizer" code constructs data range query strings to be used by each sub-job
- PJM manages "top-job" and all subordinate "sub-jobs" to completion

Objective is reduction in overall job completion time

Which shortens overall batch window if other jobs are dependent on this job for completion

Integration with Enterprise Schedulers

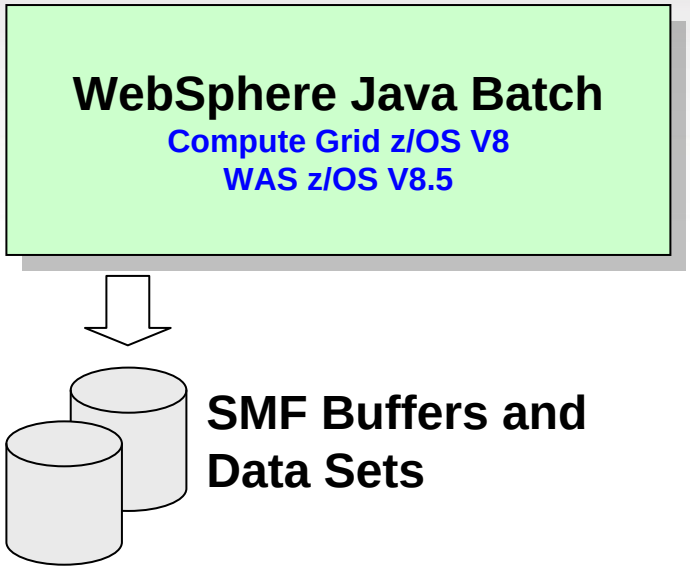
WebSphere batch supplies a program that integrates enterprise schedulers with WebSphere Java Batch, enabling incremental adoption into existing schedulers



- 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

SMF 120.9 Activity Recording

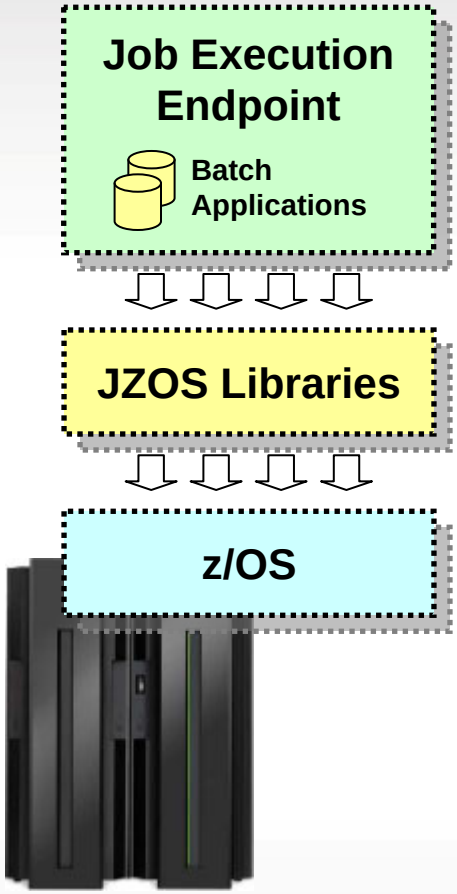
WAS z/OS supports the use of activity recording using the SMF 120.9 record. WebSphere Java Batch extends the record with batch activity information:



- Job activity records allow you to understand how your system is being used and to provide chargeback data
- Activity recording available on all platforms, but only z/OS uses SMF, which is an extremely efficient logging mechanism
- Provides historical records for usage analysis and batch capacity planning
- Information captured:
  - Job submitter
  - Date and time of submission
  - Final job state
  - Total CPU used for job
  - General processor used for job
  - zAAP usage derived: Total - GP = zAAP

Use of JZOS Services

JZOS is a set of functions that make using Java on z/OS much easier and useful. The JZOS class libraries may be used in batch application development:



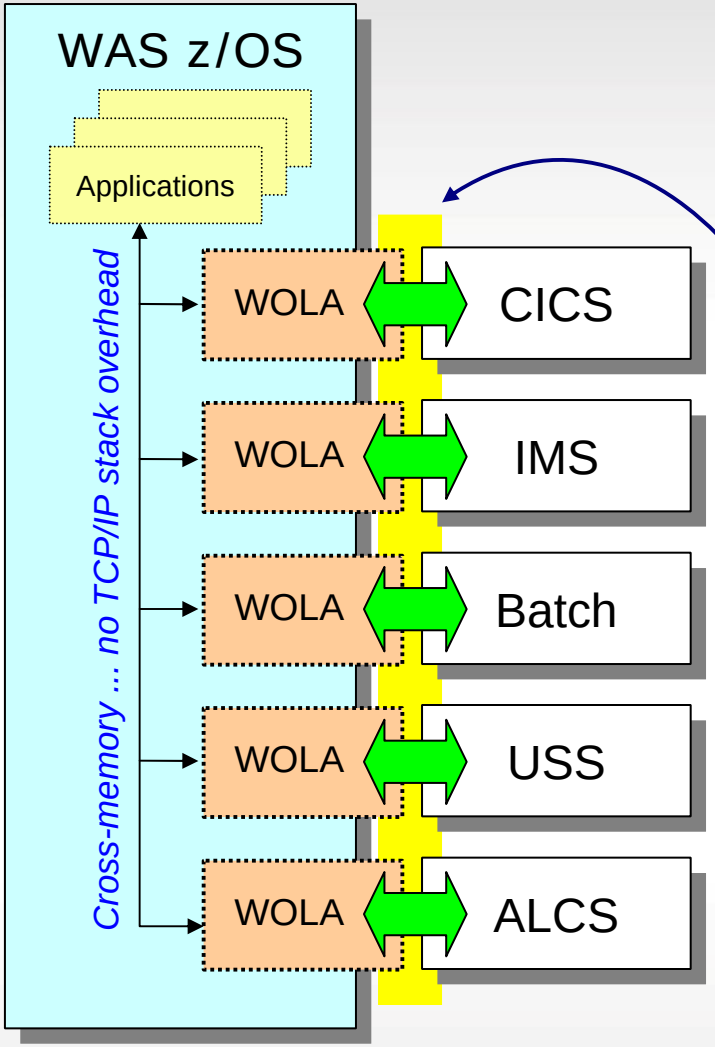
Examples of some z/OS services available:

- **DfSort** - interface for invoking DFSORT
- **MvsConsole** - class with static methods to interface with the MVS console.
- **MvsJobSubmitter** - class for submitting batch jobs to JES2 or JES3 from a Java program
- **PdsDirectory** - class for opening a PDS directory and iterating over its members.
- **WtoMessage** - simple data object/bean for holding a WTO message and its parameters.
- **ZUtil** - static interface to various z/OS native library calls other than I/O.

WebSphere Java Batch and JZOS are not mutually exclusive … the JZOS class libraries may provide exactly what you need for your batch application to access z/OS functions and services

WebSphere Optimized Local Adapters

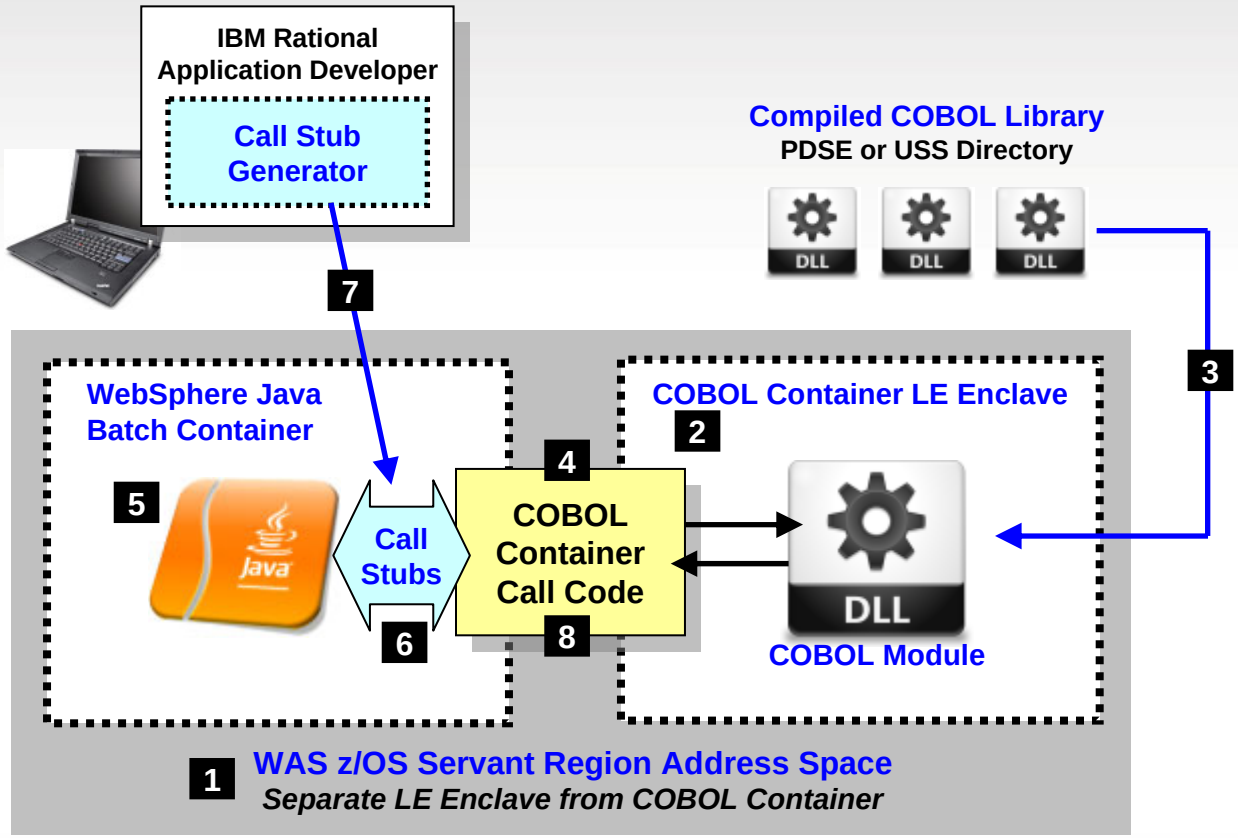
WOLA provides an efficient low-latency mechanism to exchange data bi-directionally between WAS z/OS and other address spaces:



- Leverage WOLA to call out from batch programs to co-located applications
- Very efficient byte-array transfer
- Bi-directional
  - Outbound -- Java in WAS invokes program in external
  - Inbound -- Program in external invokes Java in WAS
- Two phase commit, identity assertion
- Supplied JCA resource adapter for applications going outbound
- Supplied native APIs for cases where their usage is indicated
  - COBOL, C/C++, PL/I, High Level Assembler
  - 31-bit and 64-bit modules

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

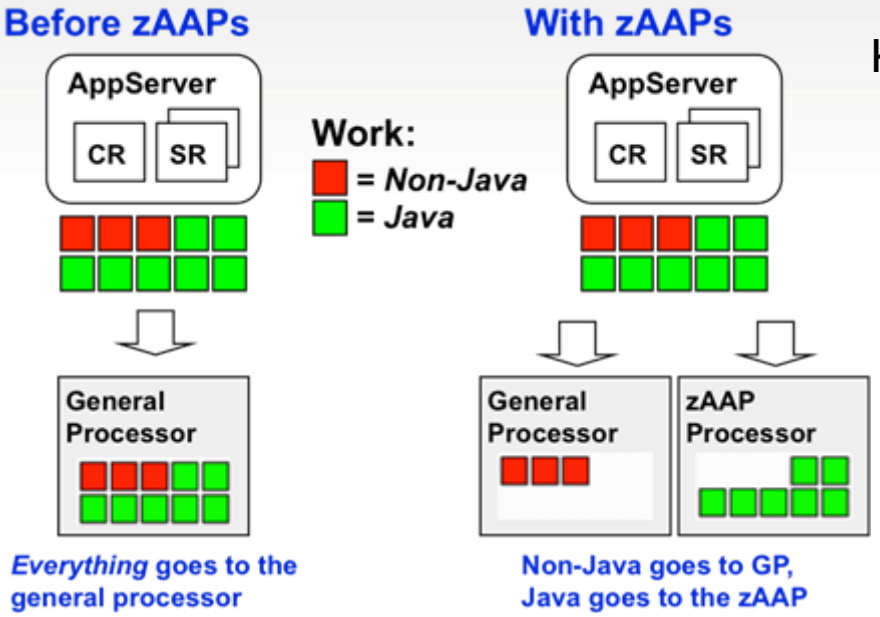


1. Batch application runs in the WAS z/OS servant region address space
2. The COBOL container is created as a separate LE enclave in the address space
3. COBOL DLLs are accessed using STEPLIB or LIBPATH
4. COBOL Container code provides the "glue" between the Java environment and the native COBOL
5. Java 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

Java Offload to zAAP Specialty Engines

zAAP engines are Java offload engines. They enhance the financial picture of the z/OS platform, and they free up GP for other key subsystem processing



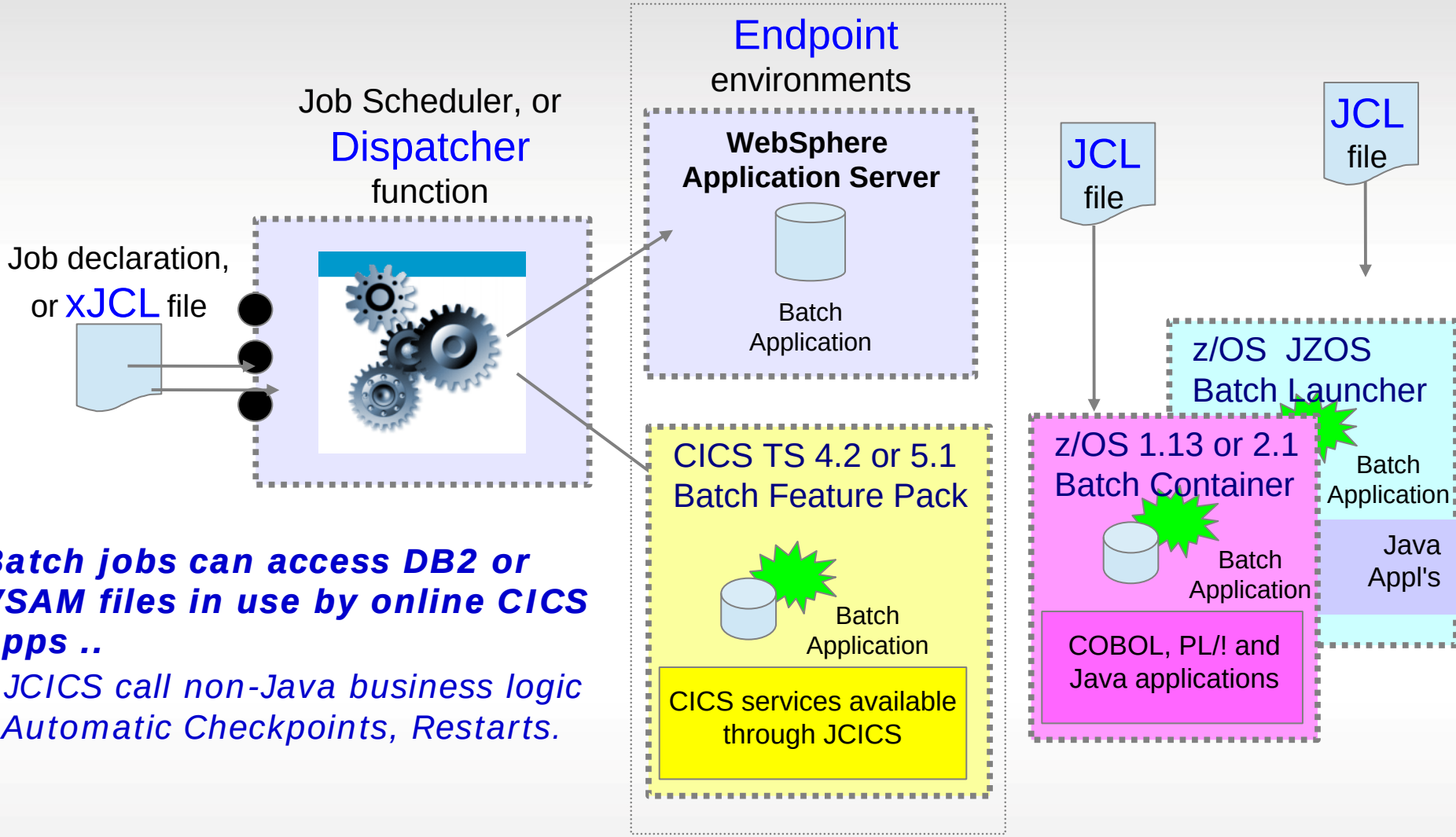
Keys to understanding value of zAAPs:

- zAAP processors have a considerably lower acquisition cost compared to GPs
- Offloading Java to zAAP frequently allows growing non-Java work to live within existing GPs, thus avoiding capital acquisition
- Monthly license charges based on capacity of the system can be influenced by the presence of zAAPs, which do not count towards charges

This is really a function of the Java SDK and the dispatcher of z/OS. The zAAP-enabled Java SDK is packaged with WAS z/OS, so WAS automatically takes advantage of zAAPs if they're present and configured

"Batch Containers" are not Limited to WAS

There's also a Java batch container for **CICS** ... and **z/OS**

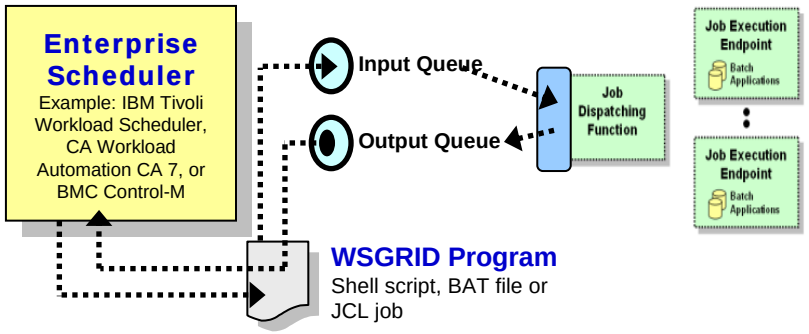


Batch jobs can access DB2 or VSAM files in use by online CICS apps ..

- *JCICS call non-Java business logic*
- *Automatic Checkpoints, Restarts.*

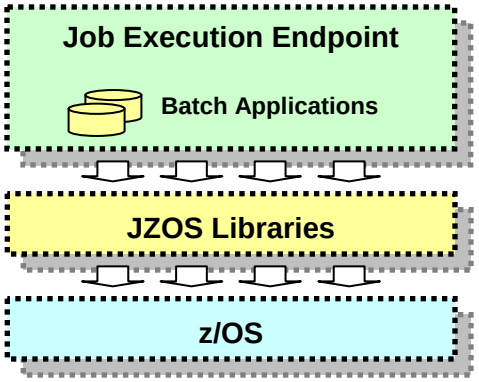
Key capabilities for incremental adoption

Integration with Existing Processing



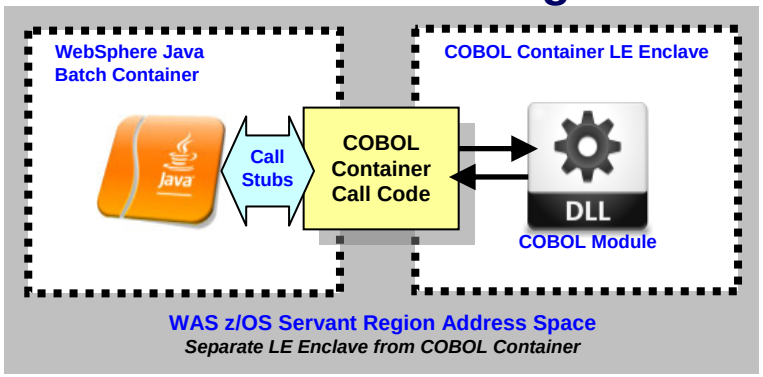
Integration with enterprise schedulers enables WebSphere batch to be introduced without disruption into existing processes

Leverage JZOS Services



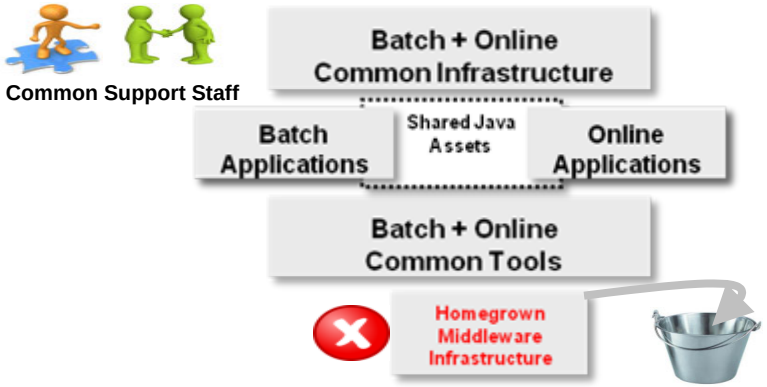
Access existing z/OS resources from WebSphere Java batch applications

Java and COBOL Integration



The close proximity and shared database connections enable an extremely efficient and effective means of calling COBOL from Java

Leveraging Shared Infrastructure



The ability to share services and infrastructure between batch and online processes allows for step-wise convergence of batch and online

WebSphere Batch Summary

Evolve to a single infrastructure for both online and batch that enables you to leverage existing applications and focus resources on business logic

✓ Improve batch execution efficiency

- **Parallel Batch Jobs** – Execute a single large batch job that is broken into chunks and executed concurrently across a grid of resources.
- **Dynamic Online & Batch Runtime** – Dynamically provision resources as capacity changes to meet operational goals.

✓ Leverage common skills and IT resources

- **Replace Homegrown Batch Frameworks** – Migrate from a native batch runtime (e.g. C / C++, PL/I, and COBOL) to Java and reduce costly proprietary batch infrastructures and focus development resources on creating business value.
- **Share business logic across Online and Batch** – Leverage the proven WebSphere platform to share logic across both batch and online, reducing maintenance and development costs.

✓ Optimize the cost of batch and online

- **Leverage System z Specialty Processors** – Offload Java workload from GP to less expensive zAAP processors, to gain processing capacity and control impacts of processor costs
- **Batch as a Service** – Expose business capabilities as a service and leverage usage accounting features for tracking and chargeback

IBM Operational Decision Manager Overview

Decision Management Modernization

Applying technology and process to gain increased “decision making” agility for business applications

- Business need: Business application “decision making” needs to adapt to changes in the marketplace, in time to make a difference

- Application Development drivers:
  - Cost savings
    More effective application development & maintenance with less business risk
    Consolidation/Restructure of existing applications, saving hardware & resources
  - Changing ratio of source inventory to development skills
    Forcing need for formal processes with an on line electronic repository
  - Be able to react to changes requested by business in days, not months

Integrating Decision Management Design/Architecture on the Mainframe

Apply well proven module design constructs to focus and accelerate change

- Isolate the decision calls into a separate "callable function“
  - Minimizes the impact of application change on the data context/API
  - Allows the decision call to be shareable by other parts of the application
- Design the data context as part of the interface
  - Recommend a custom copybook/include as the decision context
  - Allows the data context to be a tailored subset of the application data
- Identify extractable rules decisions
  - Focus on LoB driven changes with any regularity/urgency
  - Define the decision from actual business policy rather than current application behavior
  - Architect an incremental application refactor driven by targeted decisions

Redefined Application Change Cycles

Innovate change cycles provide agility while controlling risk

Application Developer

Application Development

Business Rule Management

Business & IT

Functions / Tasks / Flow changes in Weeks / Months

Functional Requirements

Functional Enhancements / Platform Upgrades

In-Production Business Rule Application

Synchronization

Business policies

Business policy and rule changes

Business policy and rule changes

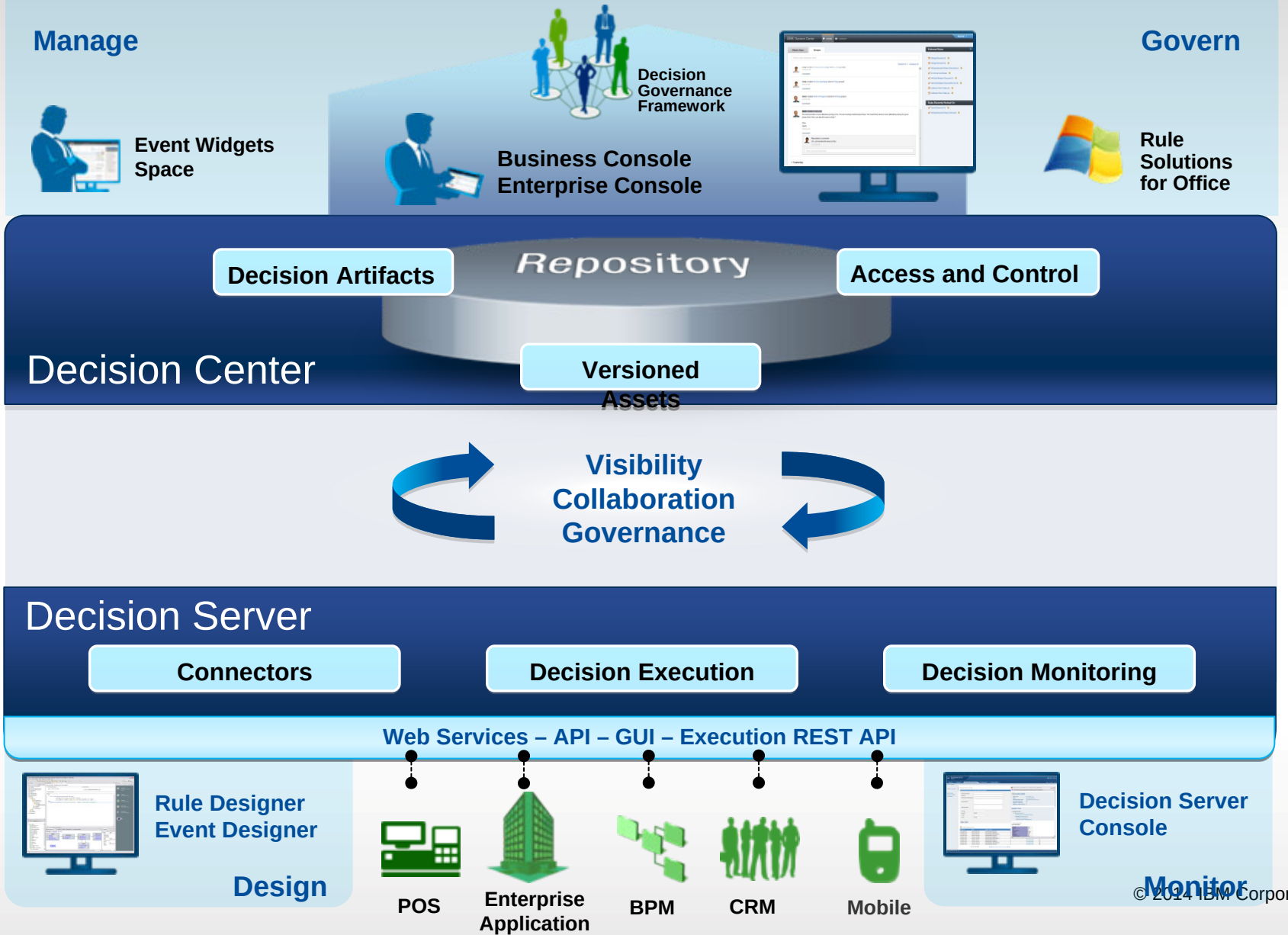
Business policy and rule changes

Business policy and rule changes

Decisions / Policies Changes in Days / Weeks

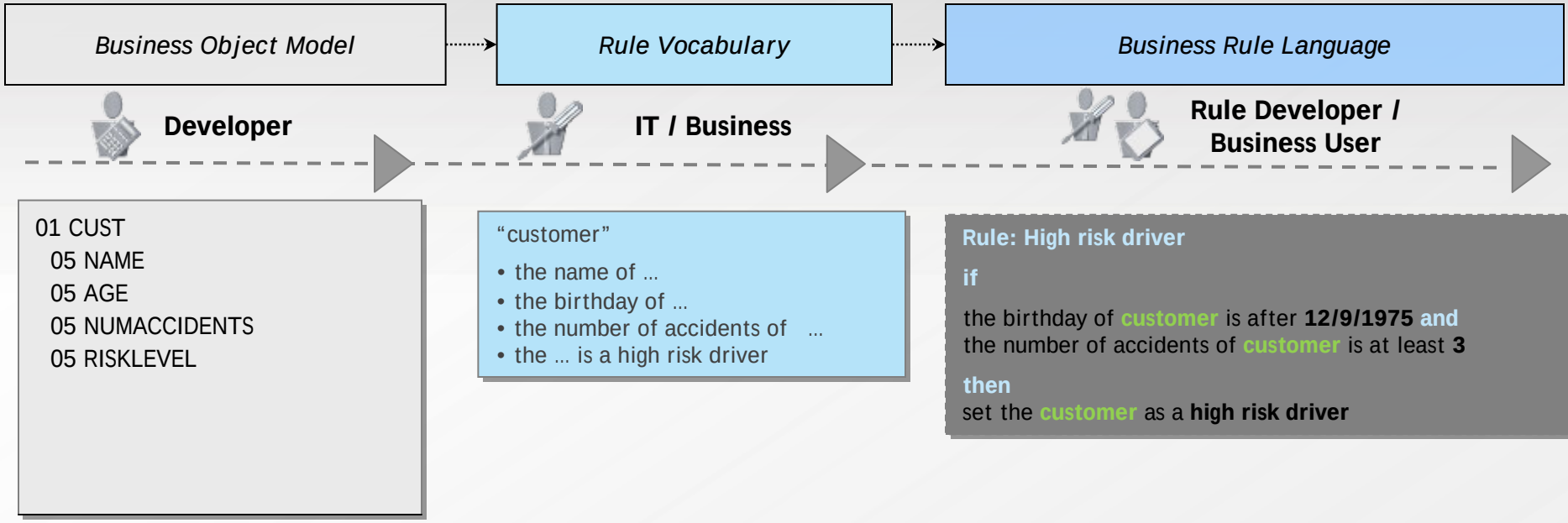
IBM Operational Decision Manager 8.5.1

A comprehensive Decision authoring, governance and runtime offering

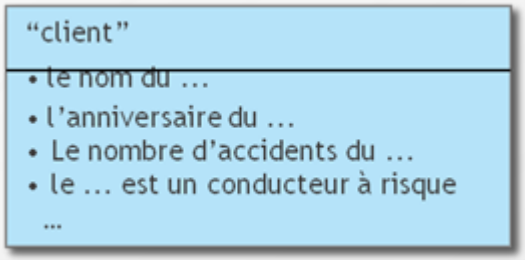


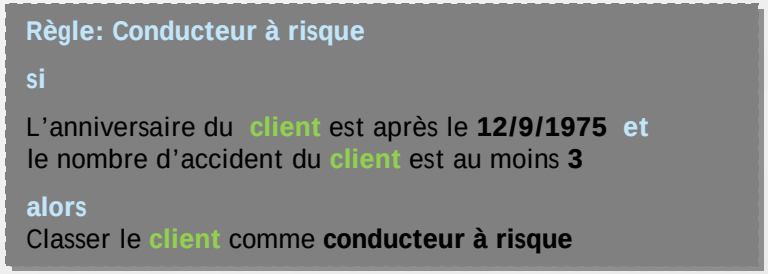
ODM Brings the IT and Business together

This result is more efficient decisions that work well in a mobile centric world



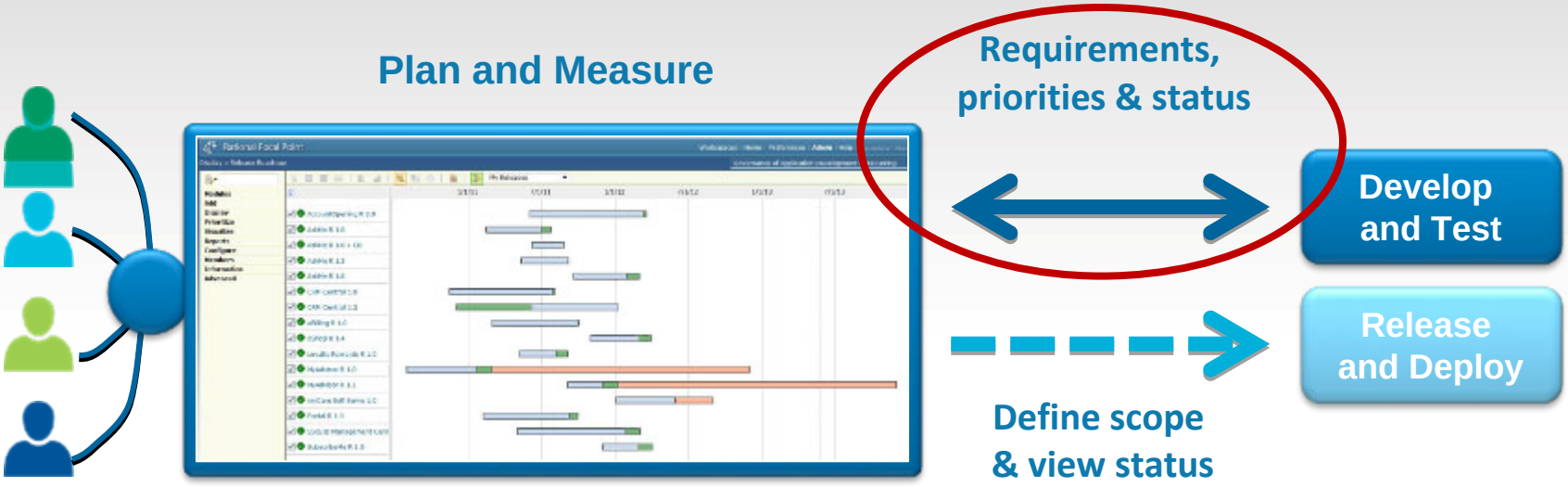
- Automatic generation of the rule vocabulary.
- Comprehensive industry focused business terms to define its data and associated actions.
- Localizable vocabulary





Integrating Decision Management
Adapting the Process on the Mainframe

Enhancing collaborative relationships through a common language



- Adapting the requirement process
  - Reach consensus with the LoB analysts on the decision vocabulary
  - Establish the requirement working language based on the decision vocabulary
  - Express business requirements as decision artifacts
  - Expect the “requirement is the design is the program” efficiencies

Rule and Event Designer

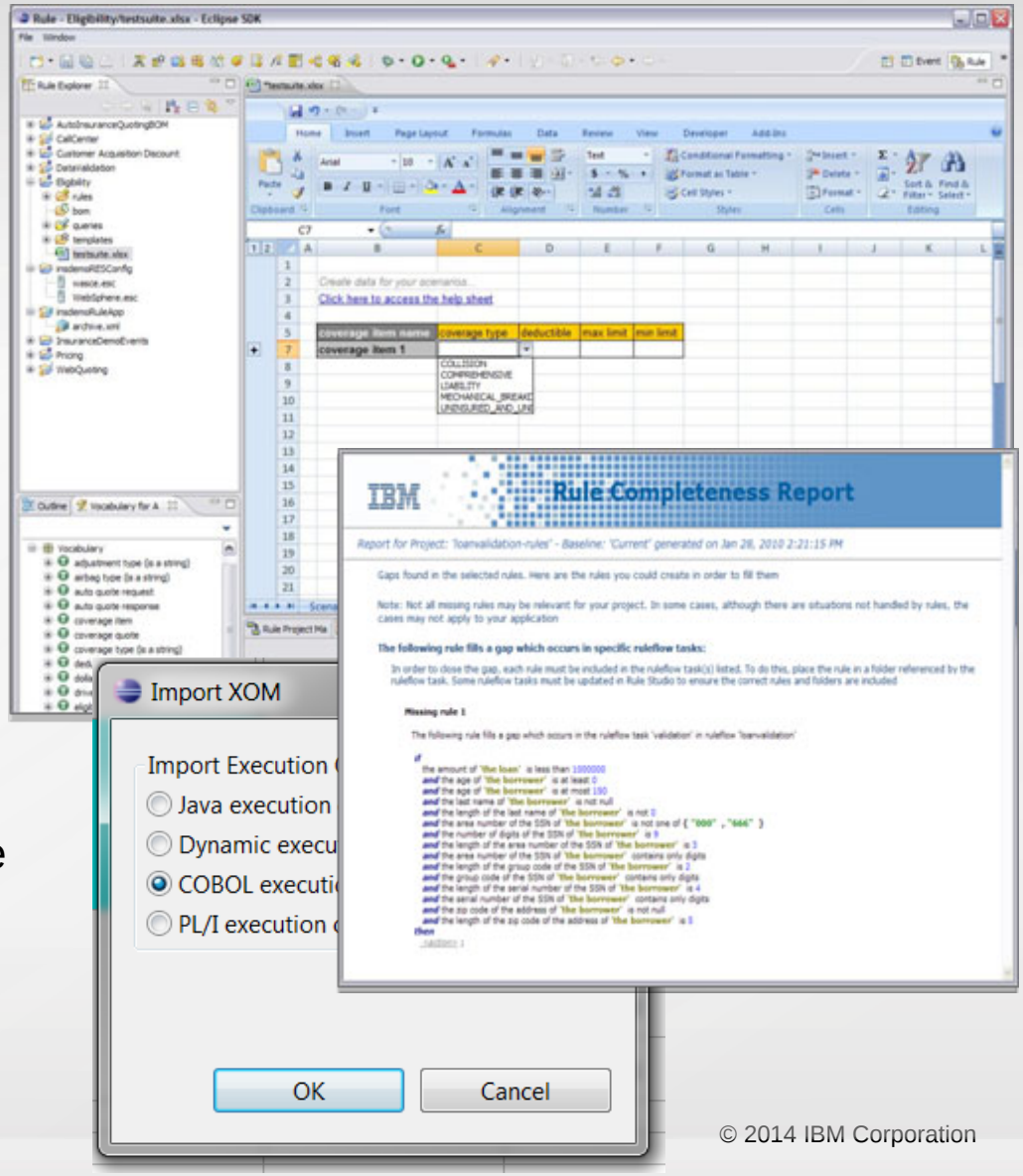
Comprehensive authoring, testing, configuring and deployment facilities

Design
- Rules and events business objects
- Vocabularies
- Projects structure and organization
- Rule Templates

Test
- Step by step debugging
- Value inspectors
- Test and simulation suites
- Completeness reports

Configure
- Business environment (Decision Center)

Deploy
- Rules and events projects to their respective execution environments



Decision Tables

Concise rendering of tabular sets of rules

	Grade	Amount of loan		Insurance required	Insurance rate
		Min	Max		
0	A	< 100,000		false	⊘
1		100,000	300,000	true	0.001
2		300,000	600,000	true	0.003
3		≥ 600,000		true	0.005
4	B	< 100,000		false	⊘
5		100,000	300,001	true	0.0025
6		300,000	600,000	true	0.005
7		≥ 600,000		true	0.0075
8	C	< 100,000		true	0.0035
9		100,000	300,000	true	0.006
10		300,000	600,000	true	0.0085
11		≥ 600,000		true	0.0145
12	*Otherwise*			true	0.022

Actions

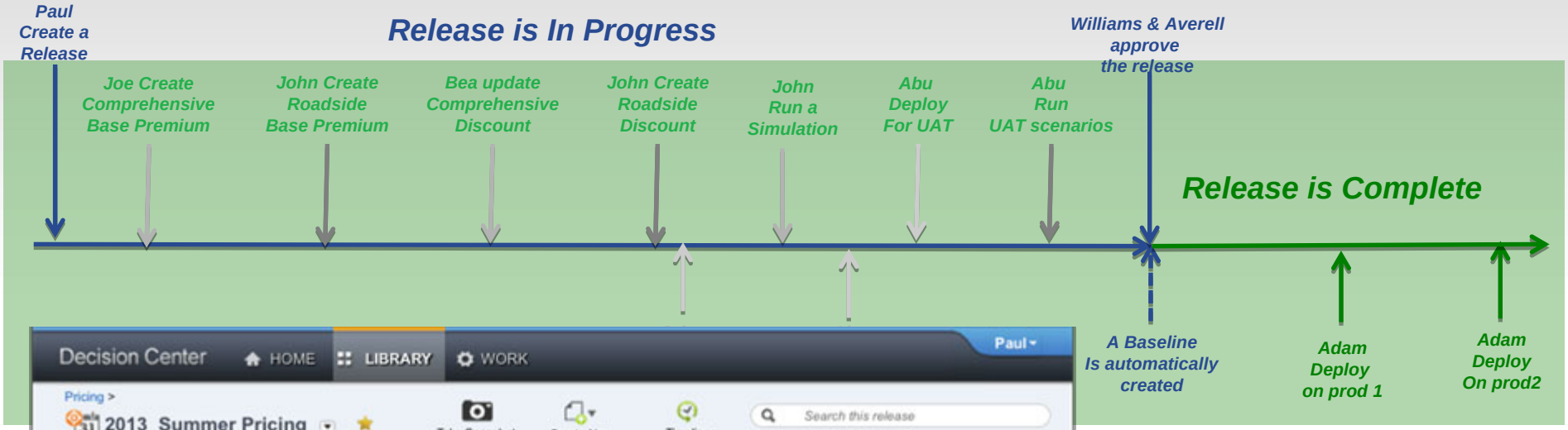
Built-in Gap/Overlap checking

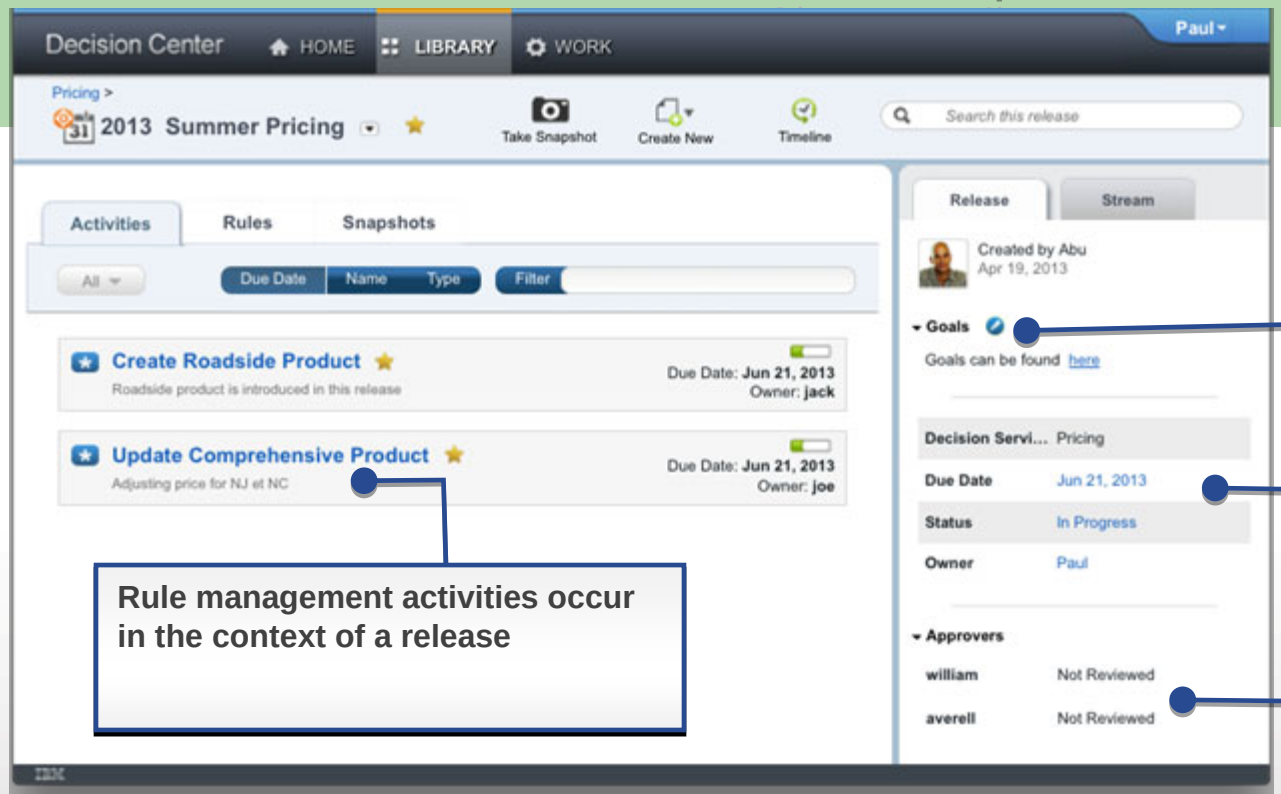
Automatic Rule generation

```
if
   all of the following conditions are true :
         - the loan grade in 'the loan report' is "C"
         - the amount of 'the loan' is at least 600000 ,
then
   set insurance required in 'the loan report' to true ;
   set the insurance rate in 'the loan report' to 0.0145 ;
```

Decision Governance Framework

All Changes and Deployments Relate to a Specific Release





Rule management activities occur in the context of a release

Release objectives are documented

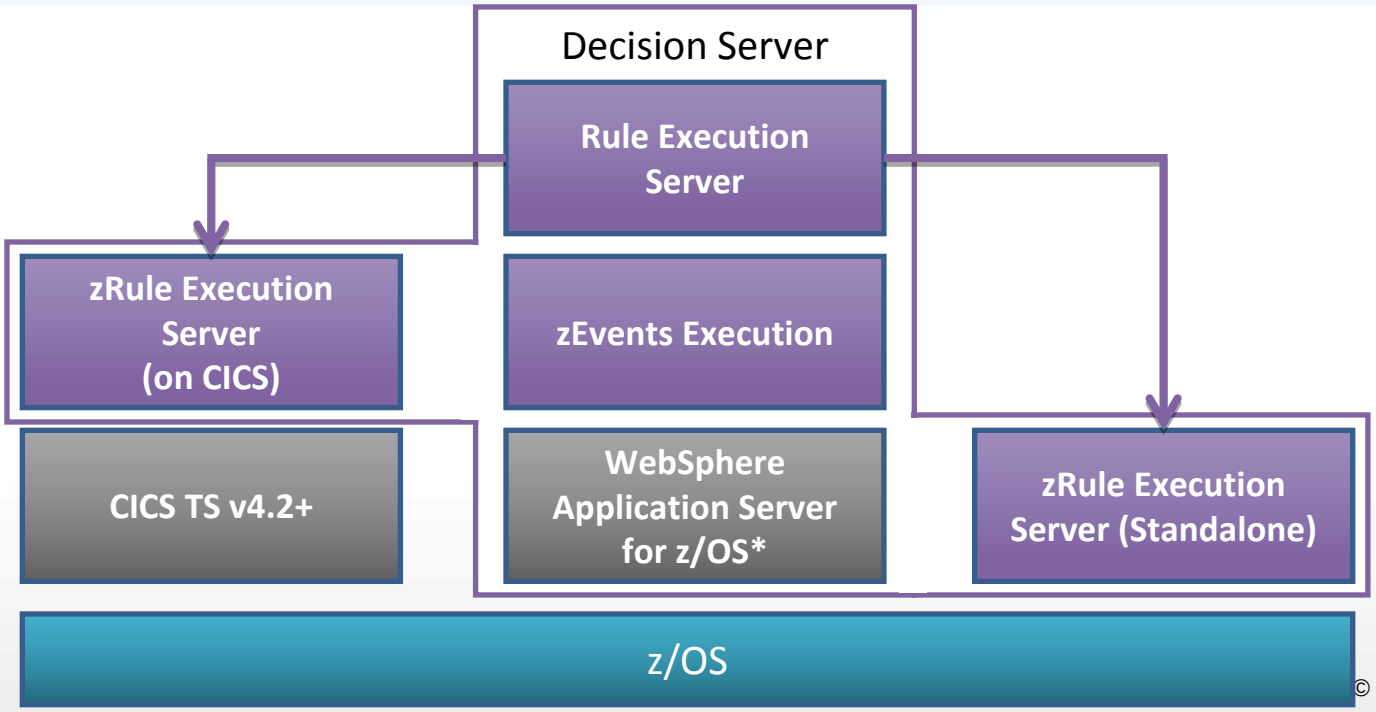
Release has
- A due date
- A status
- An owner

To complete, a Release must be reviewed and approved

Decision Server Runtime Options

Flexible options to get the Decision Server close to the calling application

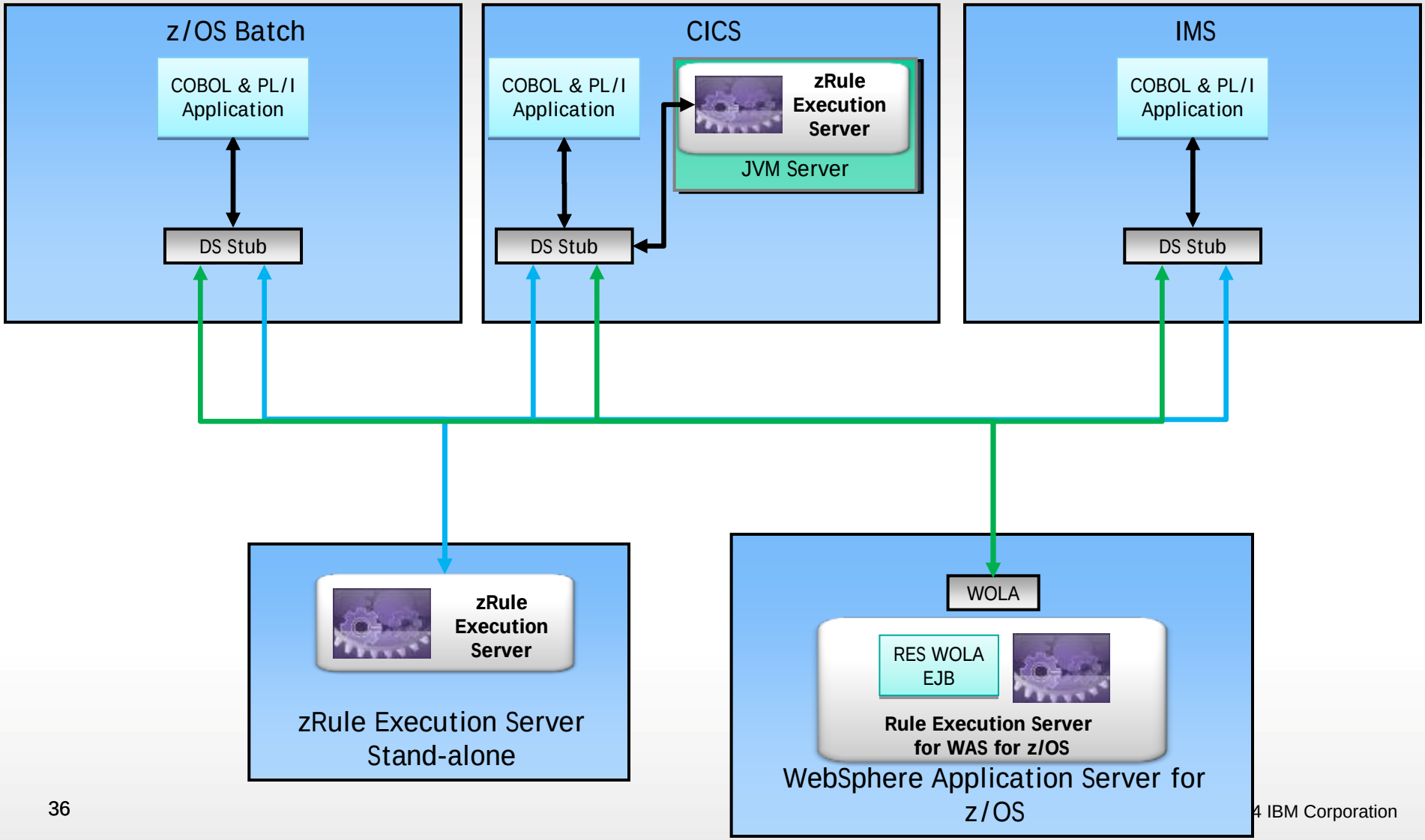
- Decisions can be invoked from existing CICS, batch and IMS applications
- Runtime support for COBOL and PL/I data types
- Flexible runtime deployment to fit any z/OS environment:
  - Deployed on WebSphere Application Server for z/OS
  - Deployed standalone to z/OS
  - Deployed in CICS TS 4.2 and above JVMServer environment



* limited use entitlement included with Decision Server

Decision Invocation Options on z/OS

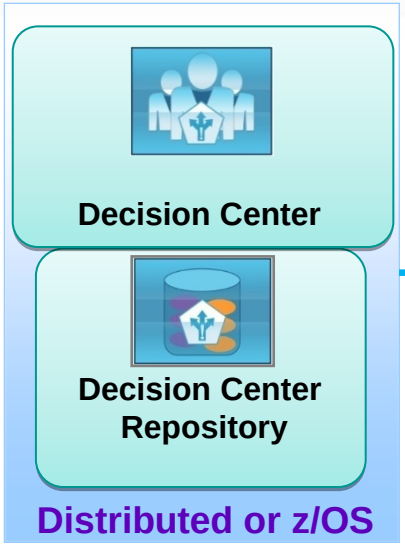
The Rule Execution Server used is transparent to the calling application



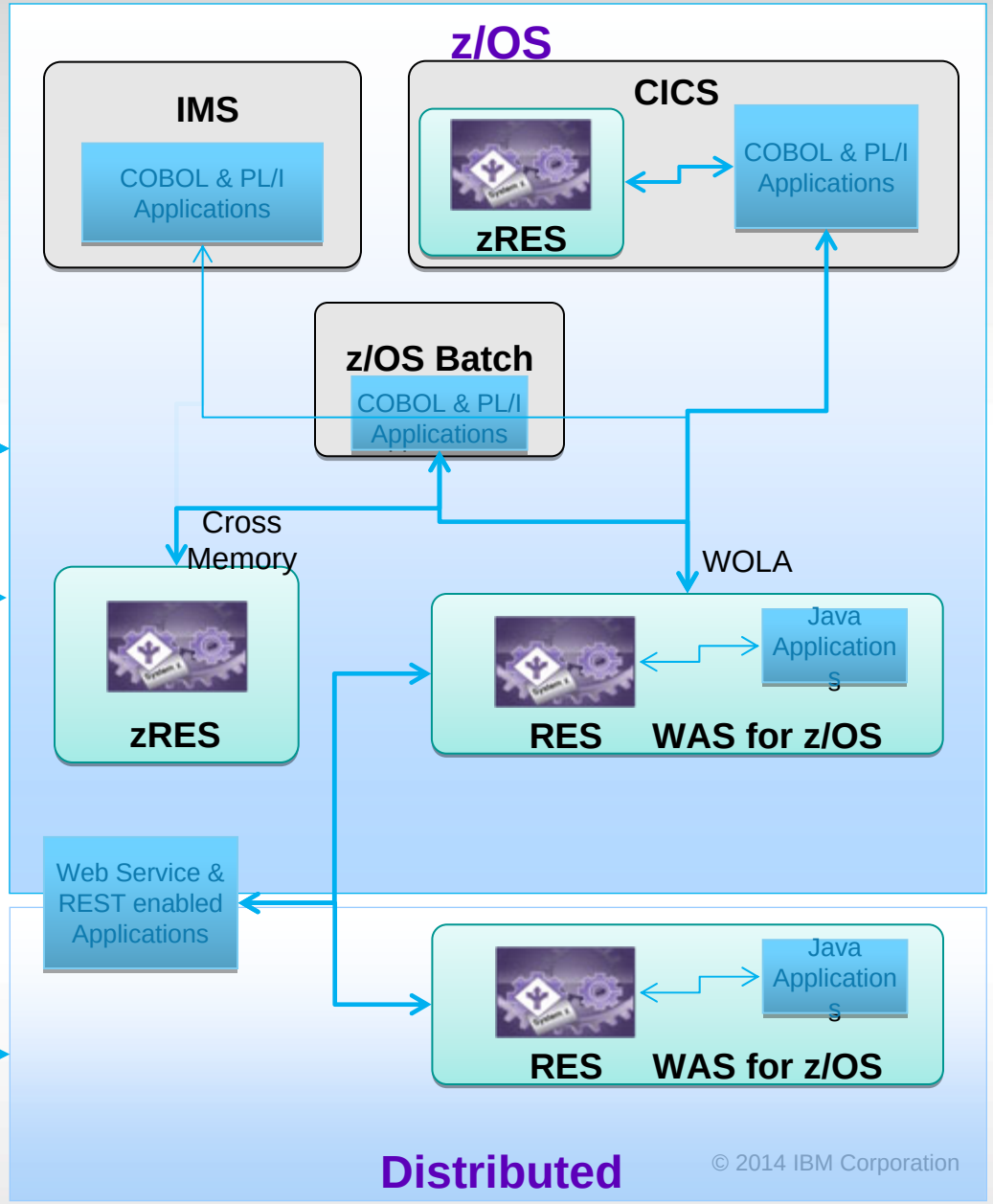
Decision Management: Comprehensive Flexibility

Architect, Application Developer

Business Analyst, Business Manager



Deploy

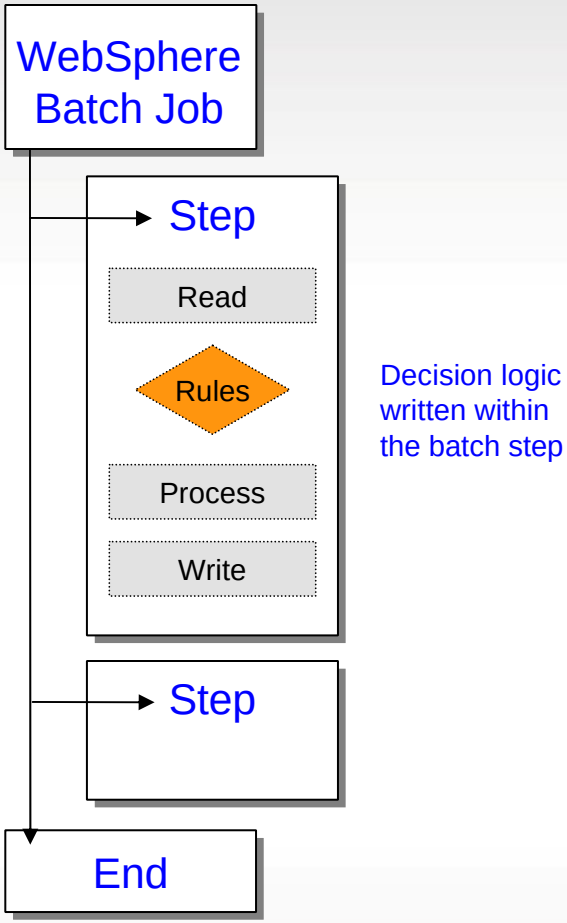


- Decision Server access on z/OS optimized for COBOL & PL/I applications
- Decision services delivering consistent business behavior enterprise wide

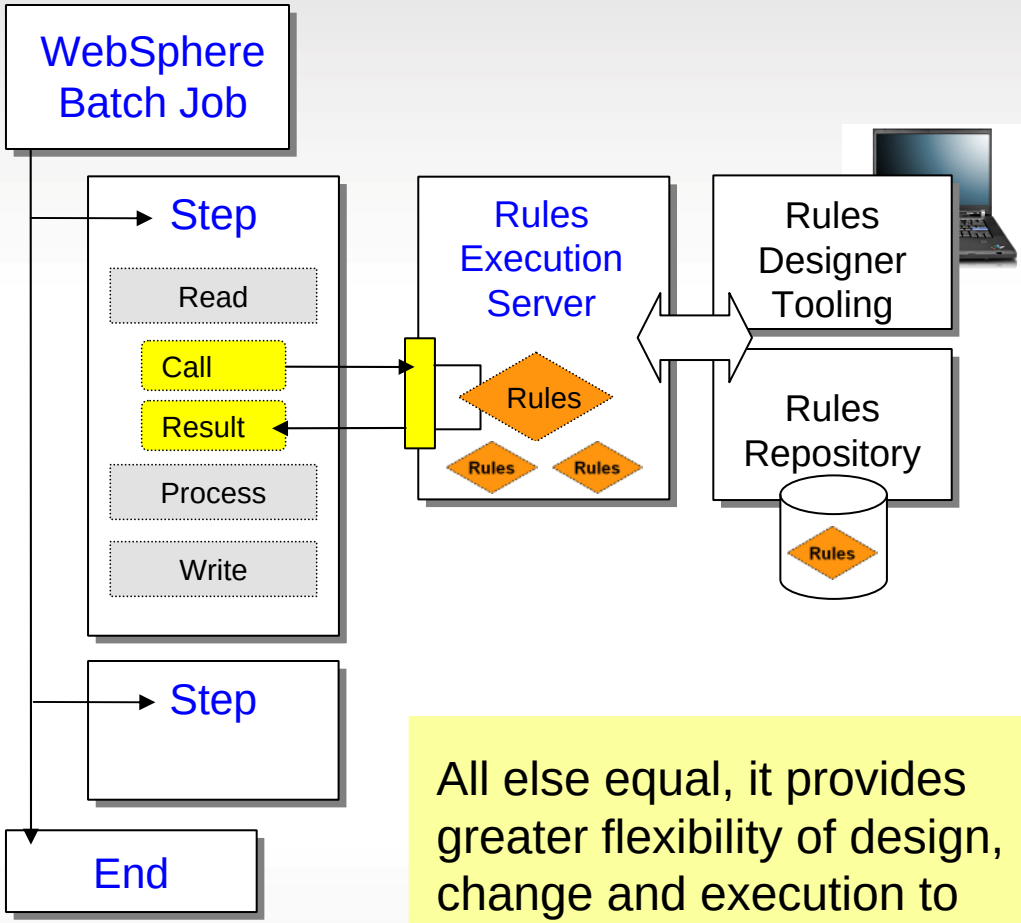
Bringing Batch and Decision Processing Together

Batch Step Decision Logic -- Externalized

Combining the capabilities of WebSphere batch with Operational Decision Manager enables automated decision processing against externalized rules



Decision logic written within the batch step



All else equal, it provides greater flexibility of design, change and execution to externalize the rules

Wrap-Up and Summary

Summary

- ✓ **Mobile access is driving significant increases to the volume of transactions**
- ✓ **Businesses need to efficiently process data and respond in a timely manner in order to retain customers and generate new opportunities**
- ✓ **WebSphere Application Server enables businesses to consolidate batch and online workloads in a prioritized and efficient manner**
- ✓ **IBM Operational Decision Manager enables businesses to streamline the definition and execution of automated business decisions**
- ✓ **Benefits include:**
  - **Faster time to market**
  - **Lower cost of maintenance**
  - **More efficient use of resources and skills**

WAS on z/OS and Java Batch Resources

Topic	Link
Guide to WebSphere on z/OS Collateral – Updated master list of links to collateral	http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP102205
WebSphere Java Batch – Overview, z/OS Specifics, Quick Start – Presentation, whitepaper, videos	http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP101783
Why WebSphere Application Server for z/OS – Executive Brochure – History of release enhancements – Technical Presentation, videos	http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP101532
WAS for z/OS Liberty Profile – Executive Brochure – Quick Start Guide and Samples	http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP102110
WebSphere Optimized Local Adapters (WOLA) – Overview, whitepapers, videos – History of WOLA updates	http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP101490
Training – z/OS Wildfire Workshops – WAS for z/OS v8.5 – WebSphere Compute Grid (WebSphere Batch)	http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/PRS1778
WebSphere on z Virtual User Group – Join the User Group – Download previous webcasts	http://www.websphereusergroup.org/zos

ODM Resources

- http://www.ibm.com/operational-decision-management
  - Shortcut: http://ibm.com/ibmodm
  - IBM Operational Decision Manager for z/OS
- White papers & tech docs
  - WebSphere z/OS – The Value of Co-Location
  - Brief introduction to WebSphere Optimized Local Adapters
  - WebSphere for System z Prescriptive Use Cases (Oct. 28, 2011 Addendum)
- Redbooks
  - Flexible Decision Automation for Your zEnterprise with Business Rules and Events
  - Batch Modernization on z/OS
  - Patterns: Integrating WebSphere ILOG JRules with IBM Software
- IBM Operational Decision Management YouTube demo
- Top 10 Business Use Cases for Operational Decision Management
- Good Decision! Decision Management blog