



Giving It the Beans: Using IBM MQ as the Messaging Provider for JEE Applications in IBM WebSphere Application Server

Chris Leonard

IBM UK

ChrisL@uk.ibm.com

Wednesday 12th August

Session 17892



SHARE is an independent volunteer-run information technology association that provides **education, professional networking and industry influence.**

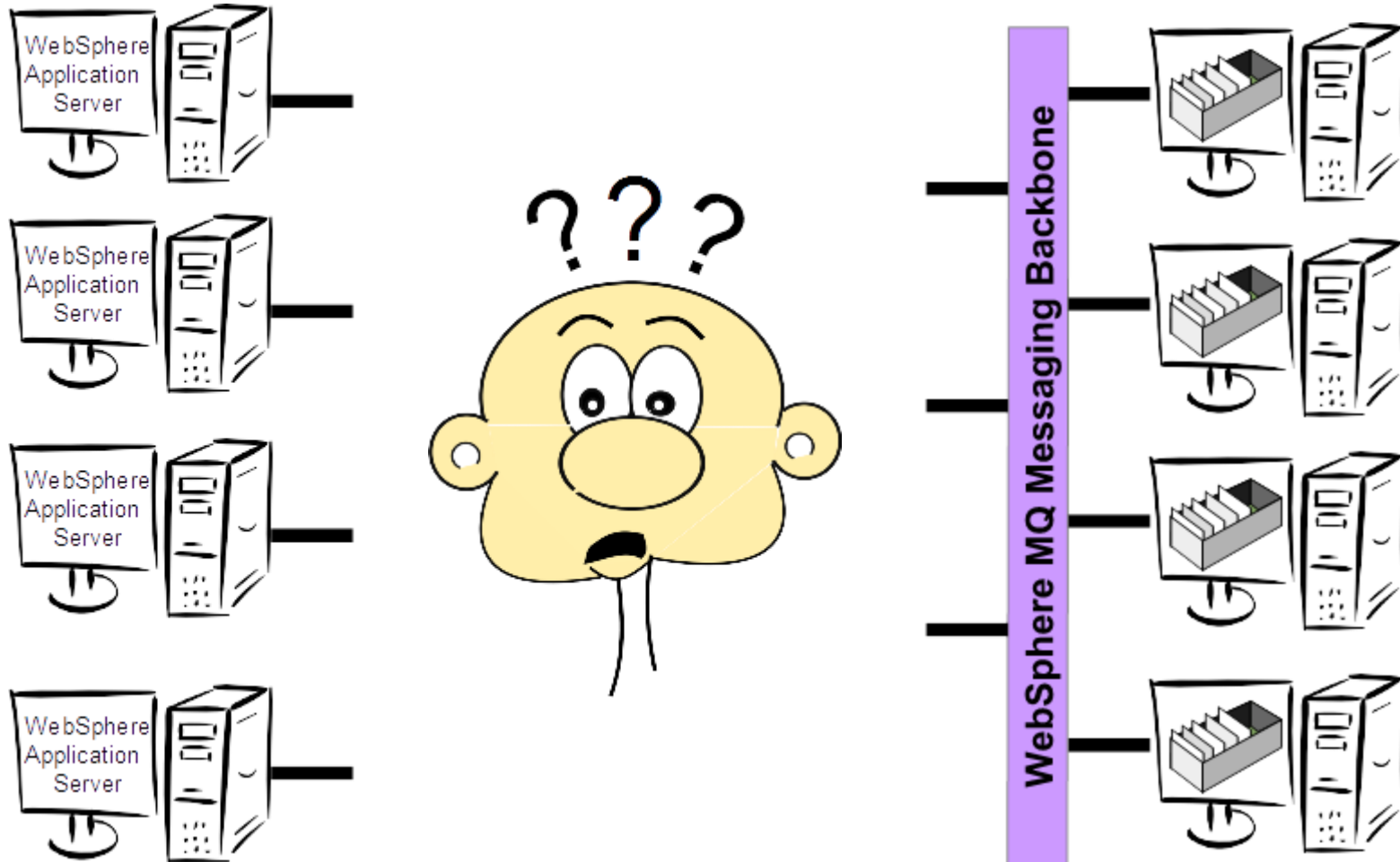
Copyright (c) 2015 by SHARE Inc. Except where otherwise noted, this work is licensed under <http://creativecommons.org/licenses/by-nc-sa/3.0/>



Agenda

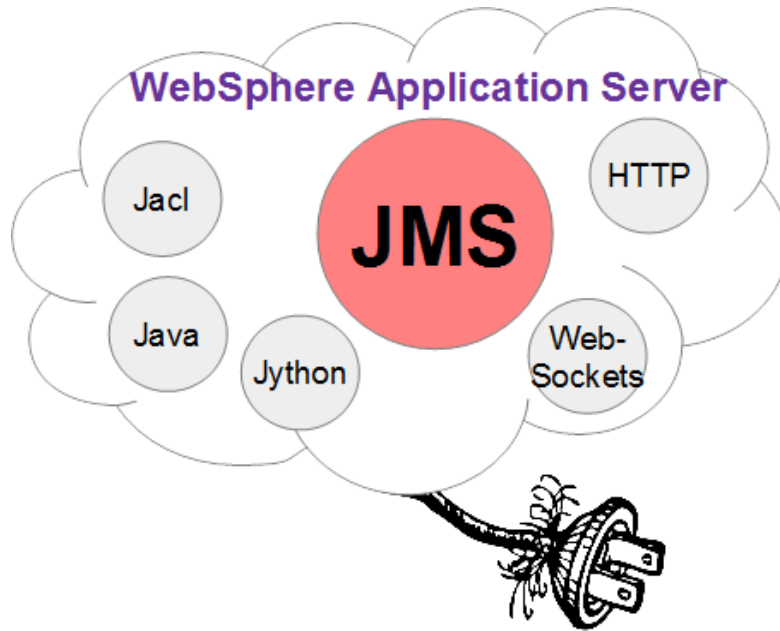
- **Connecting WebSphere Application Server to the WebSphere MQ messaging Infrastructure**
- Configuring the environment for JMS outbound messaging
- Inbound messaging
- Features of WebSphere MQ Resource Adapter
- Common 'Gotchas'
- Reference links

How do we interact with our messaging Infrastructure?



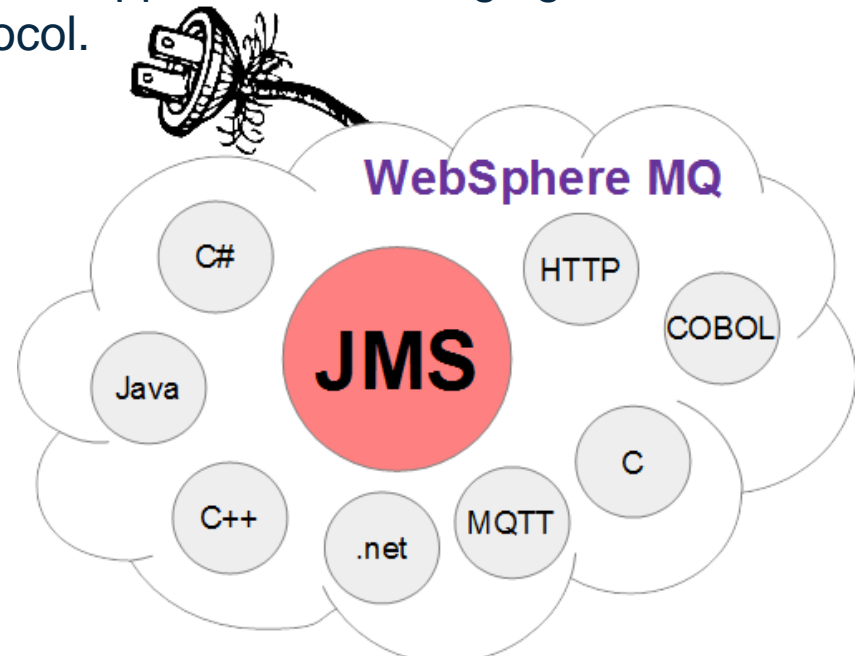
Complete your session evaluations online at www.SHARE.org/Orlando-Eval

Use the most appropriate protocol



WebSphere Application Server is a fully compliant Java Enterprise Edition (JEE) application server.

The Java Message Service (JMS) is the JEE application messaging protocol.

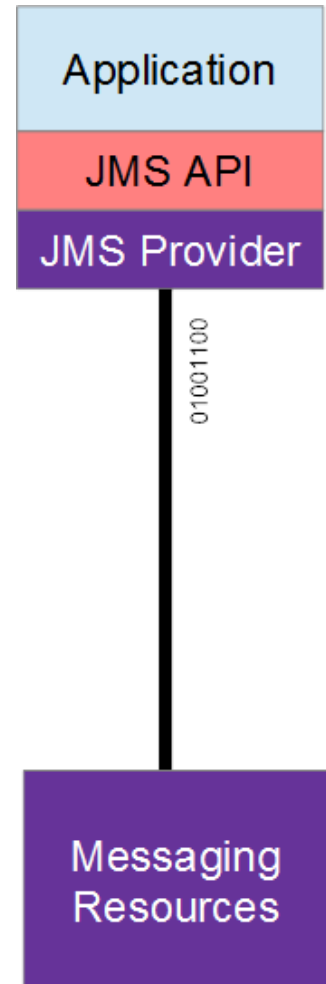


WebSphere MQ provides a fully JMS 1.1 compliant messaging provider.

Therefore, JMS is the answer!

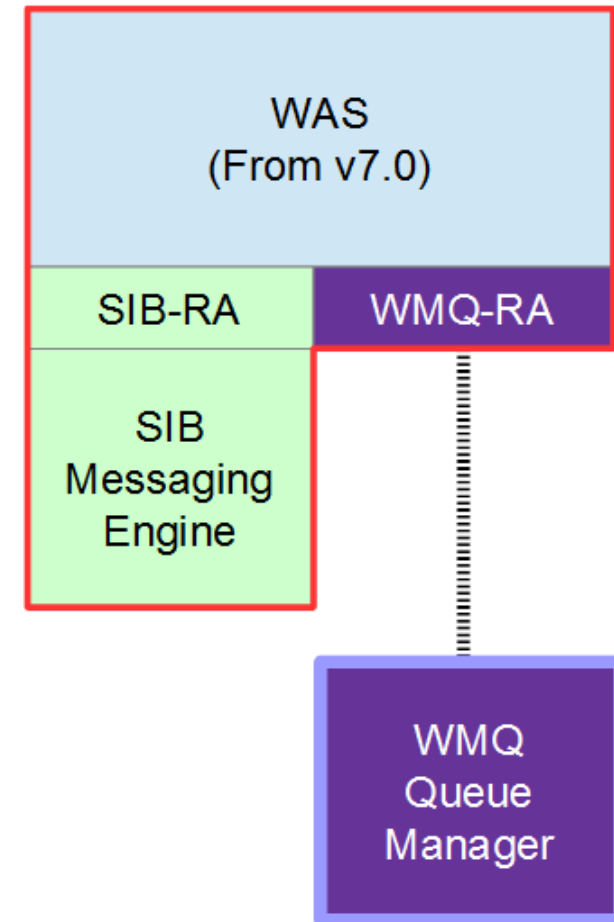
Java Message Service (JMS)

- A standardised Java API which allows applications to utilise messaging.
- JMS makes use of administered objects to keep the application abstracted away from the messaging provider's configuration specifics.
 - This also permits the configuration to be changed without recompiling the application.
- Supports both point-to-point messaging, and publish/subscribe.
- Applications are unaware of the implementation details.
JMS is not a wire protocol.



A choice of JMS Providers in WebSphere Application Server

- JEE compliant application servers must provide support for JMS.
- A WebSphere MQ server is not included within the WAS installation, so (from WAS v7.0 onwards) WAS includes an alternative JMS messaging provider, the Service Integration Bus (SIB)
- This is referred to as the “Default Messaging Provider” within the Administration Console.
- In addition to SIB, a WAS installation also comes with an integrated WebSphere MQ Resource Adapter (WMQ-RA), which provides JMS messaging functionality which uses the capabilities of WebSphere MQ.



WebSphere MQ Resource Adapter

Targeted versions of WebSphere MQ

The table below shows versions of the WebSphere MQ Resource Adapter which have not yet been shipped in a release of WebSphere Application Server. The second column provides information on which WebSphere Application Server fix packs the WebSphere MQ Resource Adapter version has been targeted for.

WebSphere MQ JCA resource adapter Version	Targeted for inclusion in WebSphere Application Server Fix Packs
7.0.1.12	7.0.0.35
7.0.1.12	8.0.0.10
7.1.0.5	8.5.5.4

WebSphere Application Server V8.5

WebSphere Application Server Version 8.5 ships with the WebSphere MQ Version 7.1 Resource Adapter. The table below shows the level of the Resource Adapter that is included with specific releases of the application server.

WebSphere Application Server Version	WebSphere MQ JCA resource adapter Version	Implementation Version, shown in WMSG1703I log entry during server startup
8.5.0.0	7.1.0.0	7.1.0.0-k000-L111005.1
8.5.0.1	7.1.0.1	7.1.0.1-k710-001-120424
8.5.0.2	7.1.0.2	7.1.0.2-k710-002-120928
8.5.5.0	7.1.0.2	7.1.0.2-k710-002-120928
8.5.5.1	7.1.0.2	7.1.0.2-k710-002-120928

<http://www-01.ibm.com/support/docview.wss?rs=171&uid=swg21248089>

- From WAS v7.0 onwards, the WebSphere MQ Resource Adapter (WMQ-RA) is now **supplied with the WAS installation***, and updated by the application of WAS fix packs.
- Therefore each version of WAS is associated with a specific version of the WMQ-RA. This is detailed on a web page (see link above).
- This does not limit the version of the queue manager you are using!
- **Do not** bundle the WMQ classes for Java/JMS .jar files within your applications.
- * A Liberty profile in WAS 8.5.5 does not include the WMQ-RA – and this must be manually added for WMQ-JMS function. This is downloaded from:

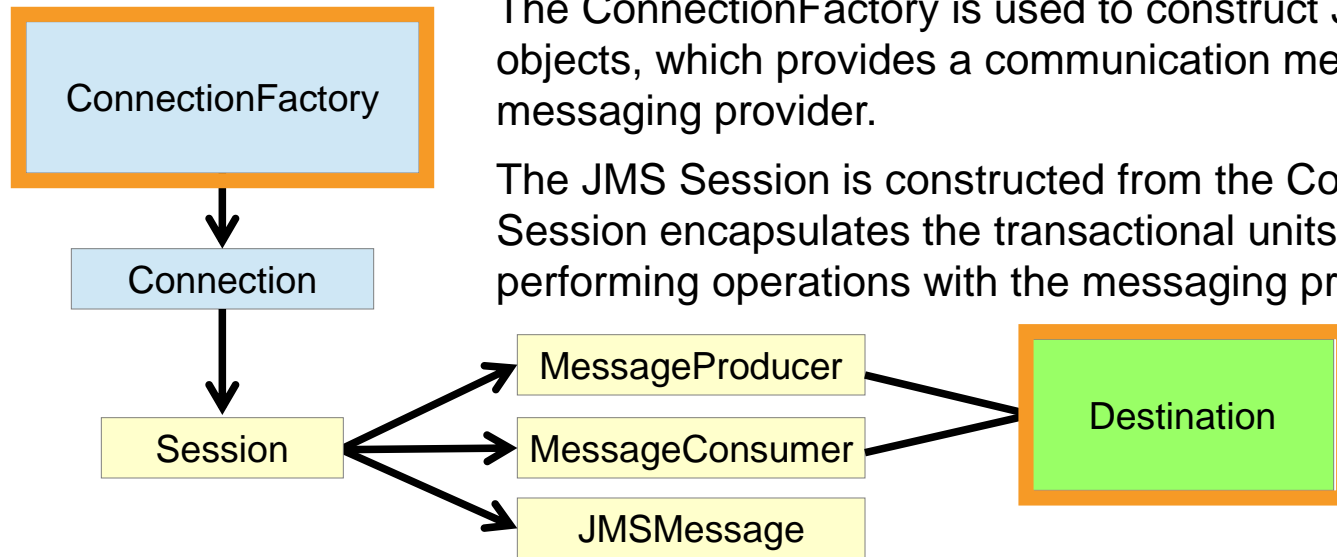
<http://www.ibm.com/support/docview.wss?uid=swg21633761>

Complete your session evaluations online at www.SHARE.org/Orlando-Eval

Agenda

- Connecting WebSphere Application Server to the WebSphere MQ messaging Infrastructure
- **Configuring the environment for JMS outbound messaging**
- Inbound messaging
- Features of WebSphere MQ Resource Adapter
- Common 'Gotchas'
- Reference links

JMS 1.1 Overview



The **ConnectionFactory** is used to construct JMS **Connection** objects, which provides a communication mechanism with the messaging provider.

The JMS **Session** is constructed from the **Connection**. The **Session** encapsulates the transactional units of work when performing operations with the messaging provider.

The **MessageProducer** and **MessageConsumer** provide message sending and receiving function. The **JMSMessage** class contains the user data and meta data which is sent or received from the messaging provider.

The JMS **Destination** holds the configuration of the queues or topics on the messaging provider that the **MessageProducer** and **MessageConsumer** are sending or receiving messages from.

The **ConnectionFactory** and **Destination** objects contain provider specific configuration data. ***This is a problem if we want our JMS application to be provider agnostic.***

JNDI – Java Naming and Directory Interface

- To maintain messaging provider independence, we lookup provider specific objects within a central store, called a JNDI.
- The JNDI is used to store more than just the configuration of JMS objects, for objects associated with database lookups.

Example of use to obtain a ConnectionFactory definition from within an EJB:

```
import javax.naming.Context;
import javax.naming.InitialContext;

... ..

Context jndiContext = new InitialContext();
ConnectionFactory myConnectionFactory =
(ConnectionFactory)jndiContext.lookup("jms/myConnectionFactory");
```

Configuring for WMQ: Connection Factories

Resources > JMS > Connection factories > [New]

Connection factories > MyCF
A unified JMS connection factory can be used to create JMS connections to both queue and topic destination

Configuration

General Properties

Administration

Scope
Node=localhostNode03,Server=server1

Provider
WebSphere MQ messaging provider

+ Name
MyCF

+ JNDI name
jms/MyCF

Description

Connection

Queue manager
MyQM

Transport
Bindings, then client

☐ Enter host and port information in the form of separate hostname and port values

Hostname
localhost

Port
1414

☒ Enter host and port information in the form of a connection name list

+ Connection name list
localhost(1414)

Server connection channel
SYSTEM.DEF.SVRCONN

- Specifies how an application connects to a WMQ Queue Manager
- Requires:
 - Queue manager name
 - Transport type (client or bindings)
 - Hostname and port
 - Channel name
 - A JNDI name – how the object is referenced from within the EJB
- Optional configuration, such as SSL
- Alternatively you can use WMQ **client channel definition table (CCDT) URL**

Configuring for WMQ: Destinations

Resources > JMS > Queues / Topics > [New]

[Queues](#) > Q

Queue destinations provided for point-to-point messaging by the WebSphere MQ destinations for the WebSphere MQ messaging provider.

Configuration

General Properties

Administration

Scope

Node=localhostNode03,Server=server1

Provider

WebSphere MQ messaging provider

✦ Name

Q

✦ JNDI name

jms/Q

Description

WebSphere MQ Queue

✦ Queue name

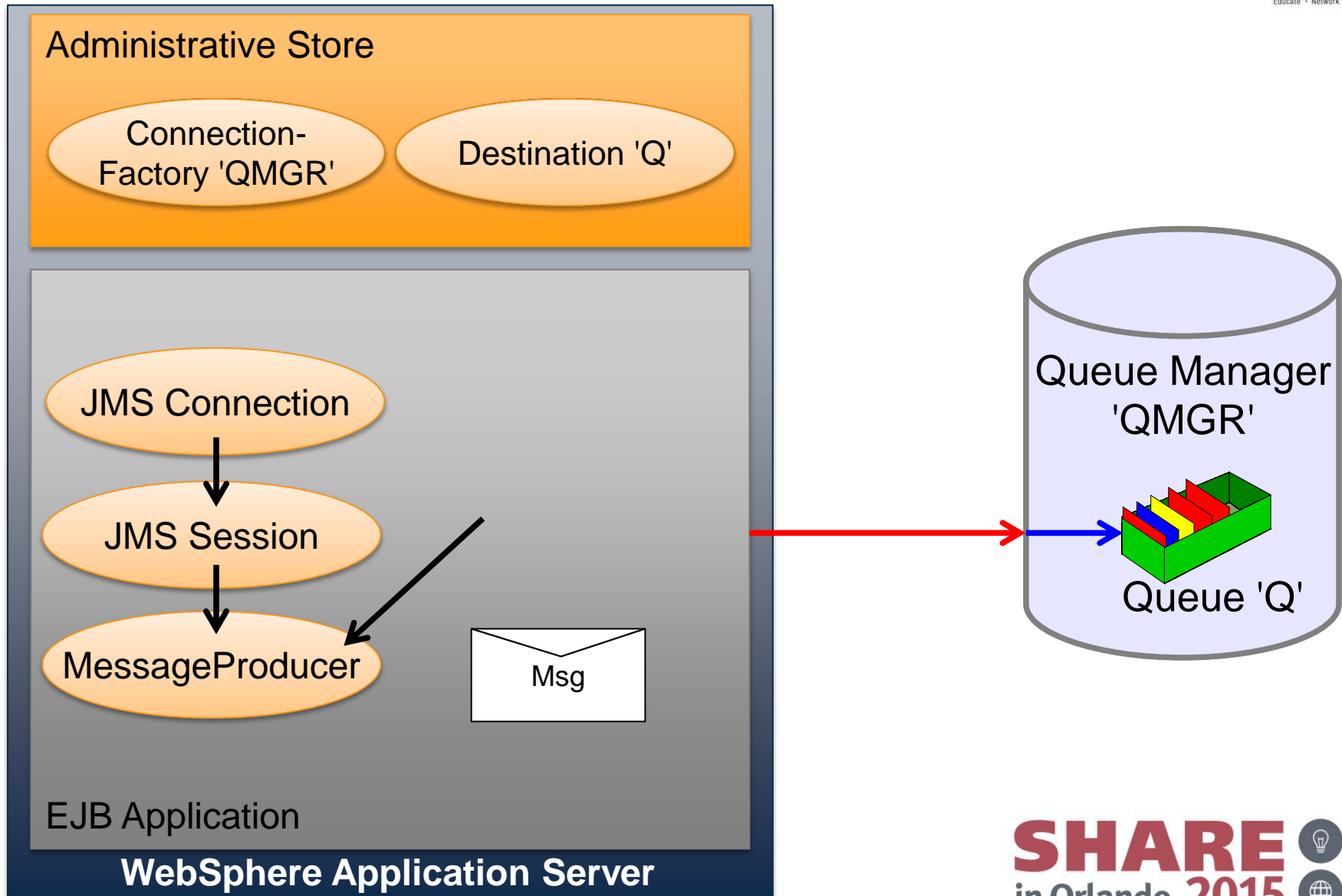
Q

Queue manager or Queue sharing group name

Apply OK Reset Cancel

- Defines references to the resources in WMQ that a JMS application will use
 - The WMQ resources must be created using WMQ administration
- **Queues**
 - Identifies the actual queue in WMQ
 - Can be used to set properties such as persistence, priority, etc.
- **Topics**
 - Defines the WMQ publish/subscribe destination

Putting it all together – sending a message



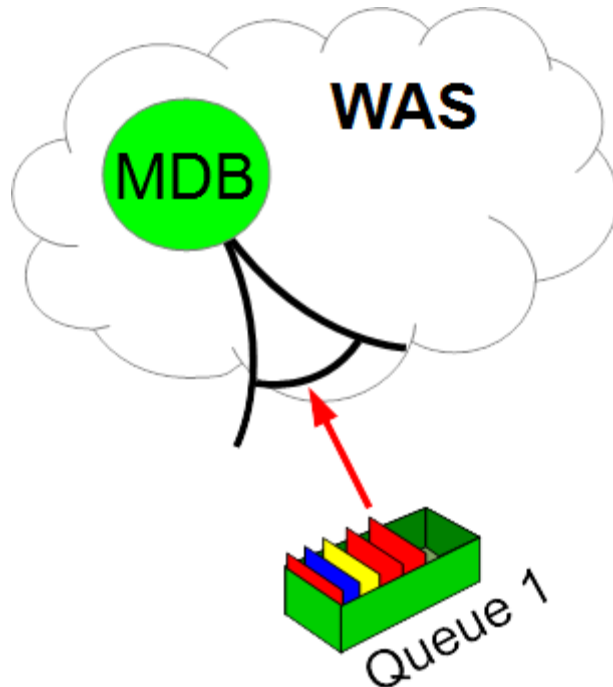
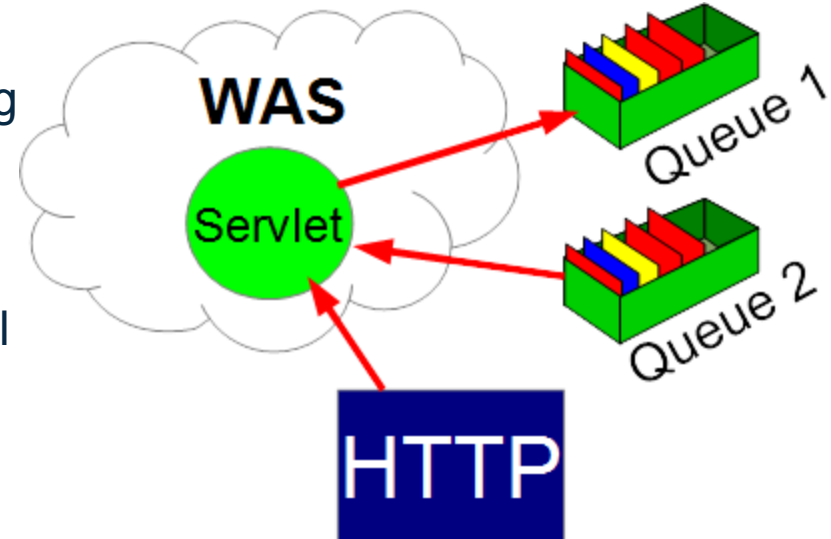
Agenda

- Connecting WebSphere Application Server to the WebSphere MQ messaging Infrastructure
- Configuring the environment for JMS outbound messaging
- **Inbound messaging**
- Features of WebSphere MQ Resource Adapter
- Common 'Gotchas'
- Reference links

Inbound Messaging

What we have looked at so far we term '**outbound messaging**', meaning messaging which was initiated from an application.

This is typically used with Web Servlets for example, which are initiated from an external HTTP request to the application server.



An alternative is '**inbound messaging**', meaning a message on a WMQ queue triggers the running of an application code.

In the JEE world, these types of application are call **Message Driven Beans (MDB)**.

They are configured using an **Activation Specification** (JEE), or a Listener Port (WAS).

Configuring for WMQ: Activation Specifications

Resources > JMS > Activation Specifications > [New]

Activation specifications > AS
WebSphere MQ Activation Specification

Configuration

General Properties

Administration

Scope
Node=localhostNode03,Server=server1

Provider
WebSphere MQ Resource Adapter

+ Name
AS

+ JNDI name
jms/AS

Description

Connection

Queue manager
MyQM

Transport
Bindings, then client ▾

☒ Enter host and port information in the form of separate hostname and port values

+ Hostname
localhost

Port
1414

☐ Enter host and port information in the form of a connection name list

Connection name list

- Activation Specifications are the standardised way of delivering messages to an MDB.
- The WebSphere MQ Resource Adapter provides Activation Specification functionality when used with WebSphere MQ.
- **Listener Ports** provide a similar function but are not part of the JEE standard, and are functionally stabilized within WAS.
- Activation Specifications combine the configuration of connectivity, the JMS Destination where messages are to be consumed from, and the runtime characteristics of the MDB itself
- Activation Specifications can be defined at all WAS configuration scopes, as can be done for ConnectionFactories and Destinations.

Further Activation Specification Specification Configuration

Resources > JMS > Activation Specifications > [New]

Integrated Solutions Console Welcome Help | Logout

View: All tasks

- Welcome
- Guided Activities
- Servers
- Applications
- Services
- Resources
 - Schedulers
 - Object pool managers
 - JMS
 - JMS providers
 - Connection factories
 - Queue connection factories
 - Topic connection factories
 - Queues
 - Topics
 - Activation specifications
 - JDBC
 - Resource Adapters
 - Asynchronous beans
 - Cache instances
 - Mail
 - URL

Specify MDB destination data

Specify MDB destination data

Step 1: Configure basic attributes

→ Step 1.1: Specify MDB destination data

Step 2: Select connection method

Step 3: Test connection

Step 4: Summary

Specify MDB destination data

Enter information about the destination from which messages will be delivered to Message Driven Beans that are associated with the new activation specification.

* Destination JNDI name
jms/myQ

Message selector

Destination type
Queue

Previous Next Cancel

- Message Selectors are used when you only want messages with a set of specific properties to be delivered to the MDB. These properties can be user-defined as in this example, or generic such as the “JMSPriority” property.

Complete your session evaluations online at www.SHARE.org/Orlando-Eval

Further Activation Specification Specification Configuration

Resources > JMS > Activation Specifications > [myA S] > Advanced properties

General Properties

Message compression

☐ Compress message headers

Compression algorithm for message payloads
NONE

Connection consumer

☒ Retain messages, even if no matching consumer is available

Rescan interval
5000 milliseconds

Maximum server sessions
10

Start timeout
10000 milliseconds

Server session pool timeout
300000 milliseconds

Message format

* Coded character set identifier
819

Additional

☒ Fail JMS method calls if the queue manager is quiescing

☒ Stop endpoint if message delivery fails

Number of sequential delivery failures before suspending endpoint
0

Key properties:

• Maximum server sessions

- How many MDB instances to run in parallel.

• Server session pool timeout

- How long an unused Server Session is left in the pool before closing. This is used to reduce system resources in low MDB activity periods, or to circumvent problems with TCP/IP and idle sockets.

• Number of sequential delivery....

- Used to stop the Activation Specification should a series of MDB instances not complete successfully.
- *Note that this behaves in a different way to Listener Ports.*

Activation Specification stopping behaviour

- The message reported in the application server log file when the number of sequential failed MDB deliveries is reached is:

CWWMQ0007W: The message endpoint <MDB name> has been paused by the system. Message delivery failed to the endpoint more than <X> times. The last attempted delivery failed with the following error:
<error details>

- Note that Listener Ports stop using a different algorithm, based on the backout count of the messages being consumed on the MDB source queue, rather than the number of sequential MDB delivery failures.

Message Driven Beans Code Snippet

```
import javax.jms.Message;
import javax.jms.TextMessage;
import javax.jms.JMSEException;

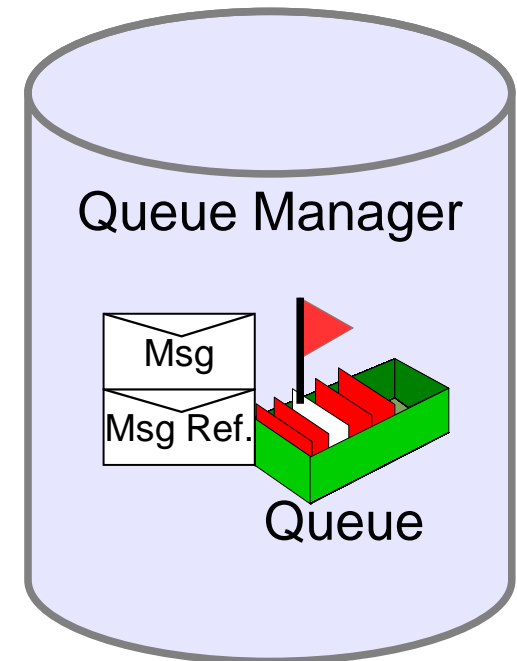
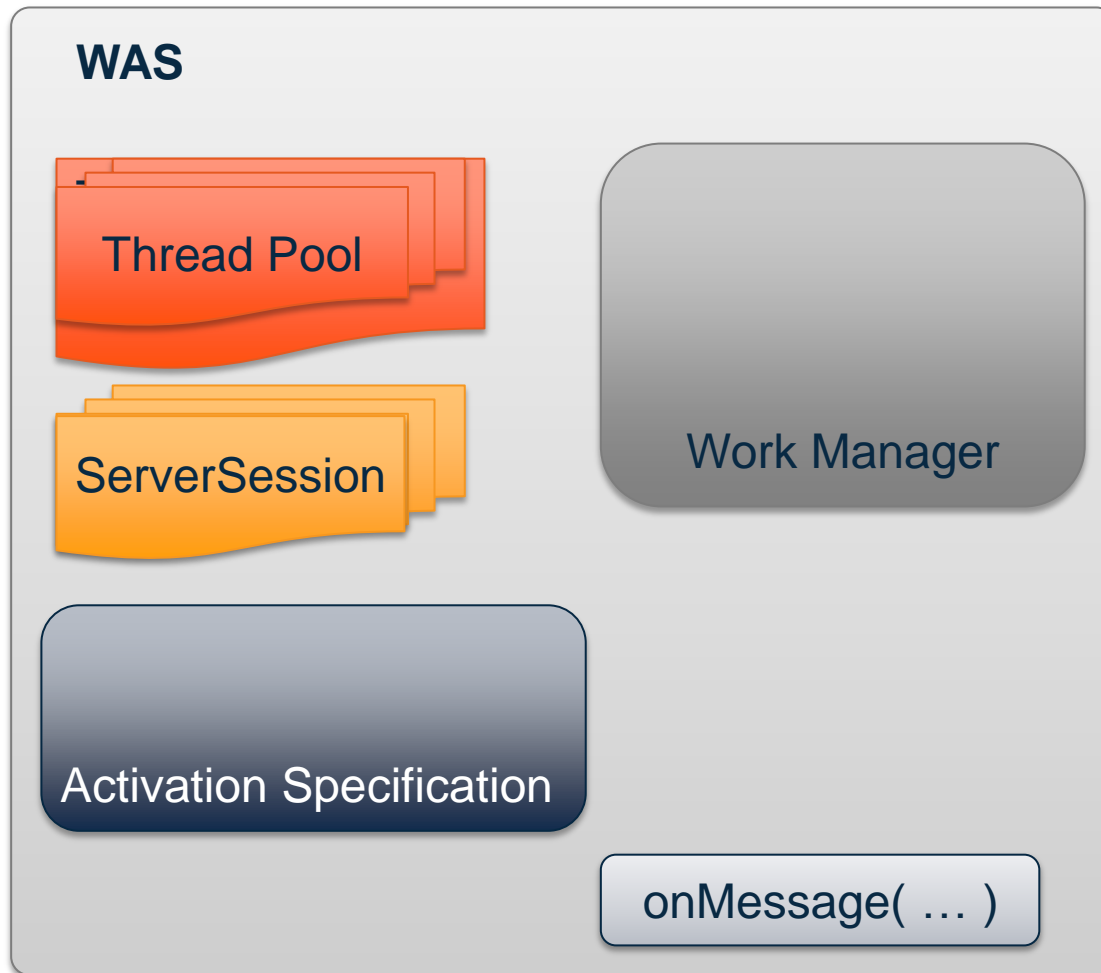
... ..

public void onMessage(Message message) {
    try {
        if (message instanceof TextMessage) {
            TextMessage textMsg = (TextMessage)message;
            System.out.println("Message text is " + textMsg.getText());
        }
    } catch (JMSEException ex) {
        System.out.println("JMSEException occurred : " + ex);
    }
}
```

- This is the application method driven by the Activation Specification (or Listener Port) when a message is available

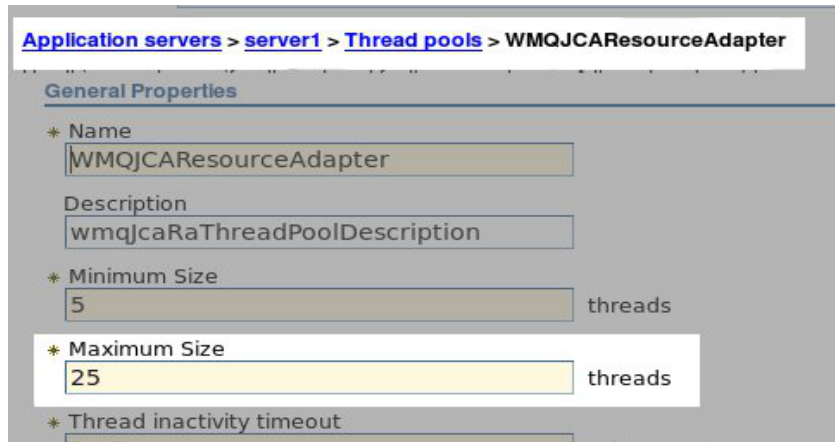
The Inner Workings of a WMQ Activation Specification

- Having an overview of how WebSphere MQ Activation Specifications function may help you to understand how to tune your system.



Activation Specifications – Thread Pooling

- In order to process a message within an MDB, two resources are needed within the WAS environment:
 1. An **available Server Session**, configured on the Activation Specification. (Default = '10')
 2. An **available thread**, configured on the WMQ Resource Adapter for the JVM. (Default = '25')



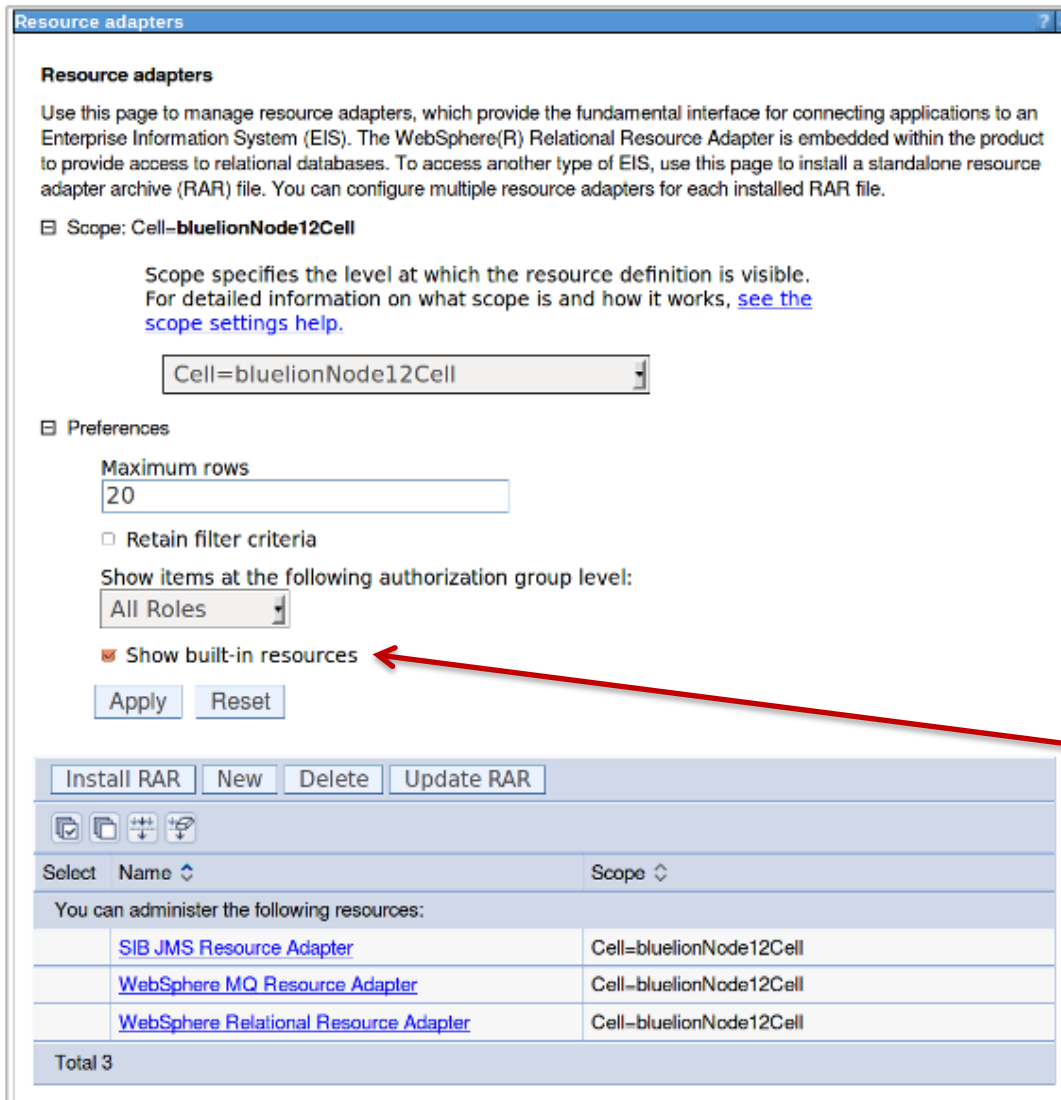
The screenshot shows the configuration page for the **WMQJCAResourceAdapter** under the path **Application servers > server1 > Thread pools > WMQJCAResourceAdapter**. The **General Properties** section includes the following fields:

- Name:** WMQJCAResourceAdapter
- Description:** wmqJcaRaThreadPoolDescription
- Minimum Size:** 5 threads
- Maximum Size:** 25 threads (highlighted in yellow)
- Thread inactivity timeout:** 5000

With the default settings, your system could run out of threads for MDB instances with three Activation Specifications, resulting in slower processing and the potential for 'WorkRejectedException' exceptions to be thrown if the time taken to wait for a thread exceeds the 'Start timeout' value configured on the Activation Specification.

Activation Specifications - Recovery

Resources — Resource Adapters



Resource adapters

Use this page to manage resource adapters, which provide the fundamental interface for connecting applications to an Enterprise Information System (EIS). The WebSphere(R) Relational Resource Adapter is embedded within the product to provide access to relational databases. To access another type of EIS, use this page to install a standalone resource adapter archive (RAR) file. You can configure multiple resource adapters for each installed RAR file.

Scope: Cell=**bluelionNode12Cell**

Scope specifies the level at which the resource definition is visible. For detailed information on what scope is and how it works, [see the scope settings help](#).

Cell=**bluelionNode12Cell**

Preferences

Maximum rows
20

☐ Retain filter criteria

Show items at the following authorization group level:
All Roles

☒ Show built-in resources

Apply Reset

Install RAR New Delete Update RAR

Select	Name	Scope
	You can administer the following resources:	
	SIB JMS Resource Adapter	Cell=bluelionNode12Cell
	WebSphere MQ Resource Adapter	Cell=bluelionNode12Cell
	WebSphere Relational Resource Adapter	Cell=bluelionNode12Cell
Total 3		

- What happens if the connection to the queue manager is broken, for example by a network interruption?
- The Activation Specification will by default go into **recovery**, which is configured on the WebSphere MQ Resource Adapter.
- Select the 'Show built-in resources' check-box and press 'Apply' to be able to see the WebSphere MQ Resource Adapter.

Activation Specifications - Recovery

The default behaviour is to try to reconnect to the Queue Manager 5 times at 300000ms (5 minute) intervals.

Following the initial connection failure, an immediate reconnection attempt is made, then at the configured intervals.

Resources → Resource Adapters →
WebSphere MQ Resource Adapter → Custom properties

Name	Value	Description	Required
You can administer the following resources:			
connectionConcurrency	5	connectionConcurrency	false
maxConnections	10	maxConnections	false
logWriterEnabled	true	logWriterEnabled	false
reconnectionRetryCount	5	reconnectionRetryCount	false
reconnectionRetryInterval	300000	reconnectionRetryInterval	false
traceEnabled	false	traceEnabled	false
traceLevel	3	traceLevel	false
Total 7			

- If the Activation Specification stops, this is reported in the application server's SystemOut.log log file, as a message of the following form:

```
[15:27:10:454] CWSJY0003W: MQJCA4013: A connection to a queue manager failed
for activationSpec 'javax.jms.Queue:jms/myQ@myQMGR <1946317826>'. Check the
queue manager error logs for details.
```

```
[15:27:10:462] CWSJY0003W: MQJCA4003: A recoverable exception occurred in the
JMS layer: 'JMSCMQ0002: The method 'MQCTL' failed.'
```

```
[15:27:30:657] CWSJY0003W: MQJCA4014: Failed to reconnect one or more MDBs
after a connection failure.
```

Agenda

- Connecting WebSphere Application Server to the WebSphere MQ messaging Infrastructure
- Configuring the environment for JMS outbound messaging
- Inbound messaging
- **Features of WebSphere MQ Resource Adapter**
- Common 'Gotchas'
- Reference links

Message Properties and Selectors

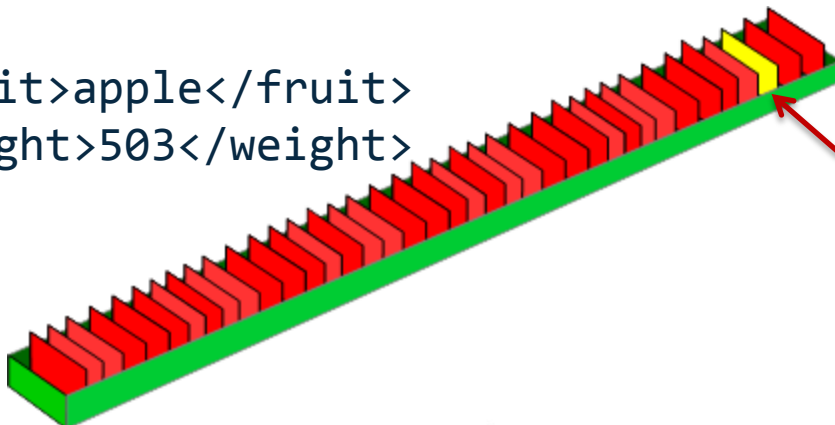
- JMS message selectors allow the filtering of message which are being consumed by an application.
 - As of WebSphere MQ v7, the queue manager understands JMS message properties, and the selection work is performed by the queue manager

Sample JMS snippet code to set user properties:

```
jmsMessage.setStringProperty("fruit", "apple");  
jmsMessage.setIntProperty("weight", 503);
```

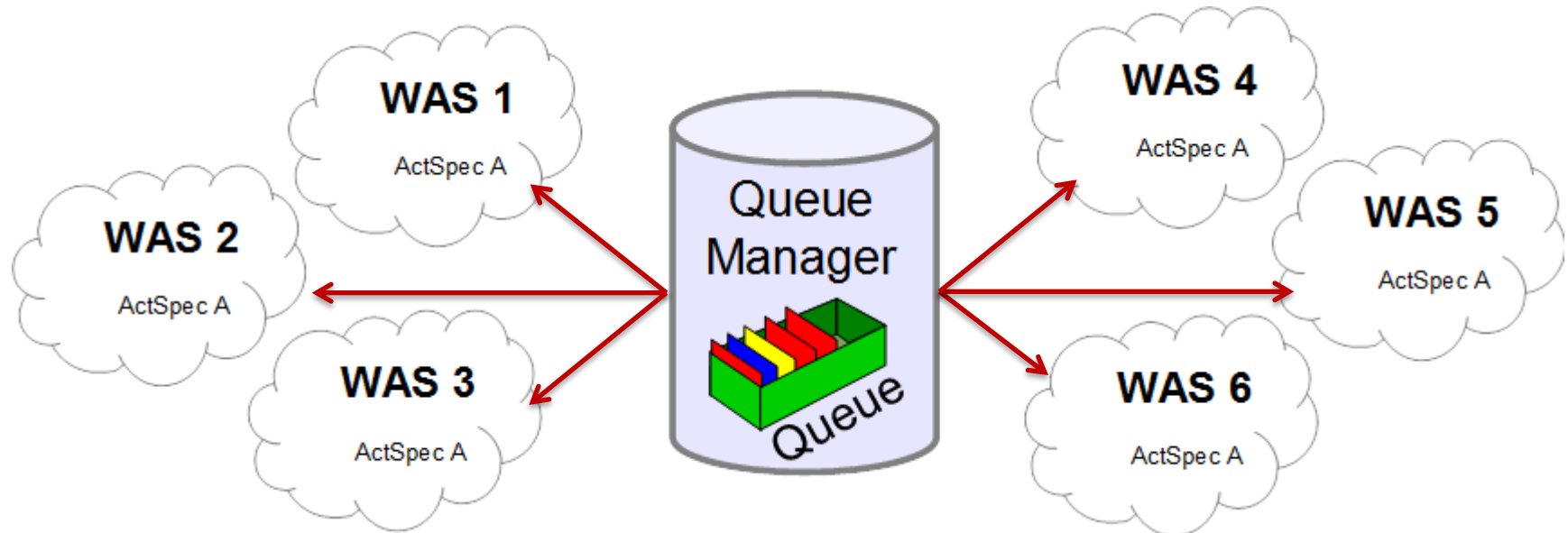
This will produce a WMQ message with the RFH2 structure:

```
<usr>  
  <fruit>apple</fruit>  
  <weight>503</weight>  
</usr>
```



Note: Remember that a queue manager is not a database. Using selectors against deep queues can have performance implications.

Asynchronous Message Consumption and Distribution of Messages



- JMS has had the concept of the asynchronous message consumer since inception:
 - `MessageListener / onMessage`
- Event driven processing is 'natural' within the Java environment.
- WebSphere MQ v7 introduced the ability for a Queue Manager to drive a consumer when a message is available.
- Message 'polling threads' are no longer necessary. Asynchronous consumers do not use `MQGET`.
- The Queue Manager determines the distribution of messages to the consumers.

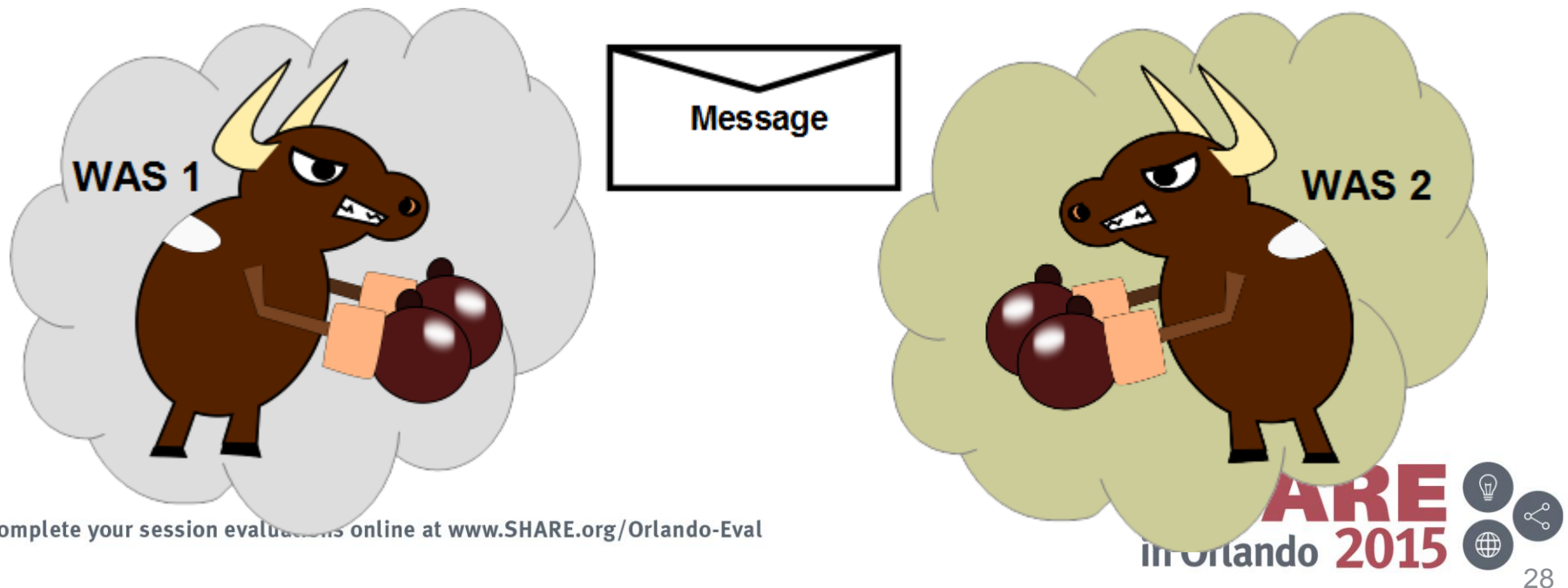
Complete your session evaluations online at www.SHARE.org/Orlando-Eval

Browse-with-Mark

Back in the 'old' days of WebSphere MQ v6 on a distributed platform, activation specifications/Listener Ports in application servers within a WAS cluster would 'fight' over the same message on a queue, due to the two-stage MDB driving method:

1. Browse for a message.
2. Get that message, identified by its CorrelationID and MessageID.

This caused message contention which could significantly slow the MDBs down, and could put a significant load on the queue manager.



Browse-with-Mark

Back in the 'old' days of WebSphere MQ v6 on a distributed platform, activation specifications/Listener Ports in application servers within a WAS cluster would 'fight' over the same message on a queue, due to the two-stage MDB driving method:

1. Browse for a message.
2. Get that message, identified by its CorrelationID and MessageID.

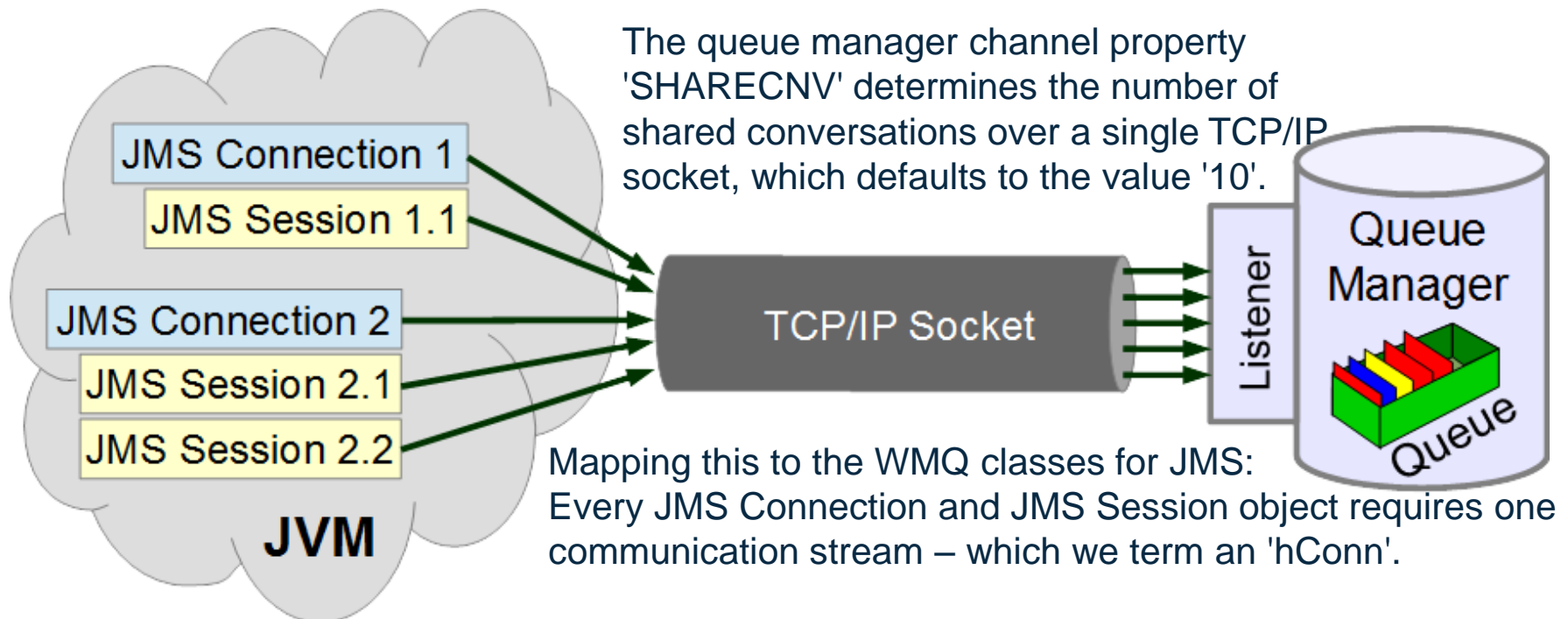
This caused message contention which could significantly slow the MDBs down, and could put a significant load on the queue manager.

This problem was solved in WebSphere MQ v7 on distributed platforms with the browse-with-mark function, where a temporary flag is placed on messages when being browsed. The queue manager property “MARKINT” determines the length of time of the flag. Be aware of the message:

CWSJY0003W: WebSphere classes for JMS attempted to get a message for delivery to an message listener, that had previously been marked using browse-with-mark, but the message was not there.

Conversation Sharing and JMS

WebSphere MQ v7.0 introduced the concept of conversation sharing. This is where multiple objects communicate with the queue manager using a multiplexed TCP/IP socket.



The higher the number of shared conversations, the increased delay there is in a busy system as only one object can communicate at any point in time. However, also the higher the number of allowable shared conversations, the less the need to create an expensive new TCP/IP socket.

Experiment on your systems to find an optimal value for your workload.

Sending Messages from WAS to non-JMS systems

Resources → JMS → Queues →
[Select Queue] → Advanced properties

Message format

Coded character set identifier
1208

☒ Use native encodings

Integer encoding
Normal

Decimal encoding
Normal

Floating point encoding
IEEENormal

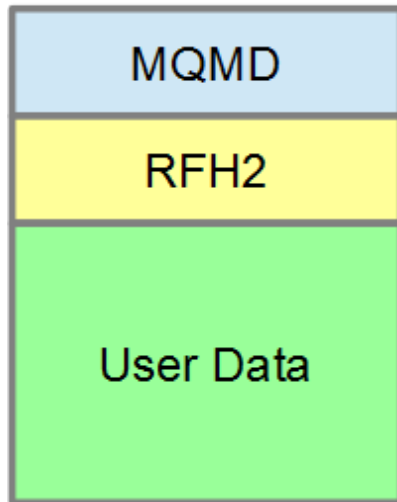
☒ Append RFH version 2 headers to messages sent to this destination

Message body
UNSPECIFIED

Reply to style
DEFAULT

TARGCLIENT switch

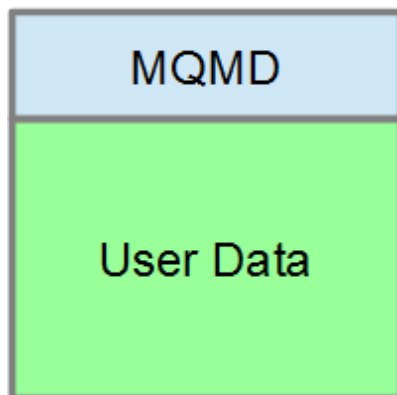
TARGCLIENT=JMS



The default behaviour of the WMQ-JMS classes when sending messages are to store JMS properties within the MQMD and RFH2 headers of the WMQ Message.

If the receiving application is a non-JMS application, it may not be expecting the RFH2. The Destination property 'TARGCLIENT' controls if the RFH2 is created in the sent message.

TARGCLIENT=MQ



JMS provider specific property name	Field and header used for transmission	Set by
JMSMessageID	MsgID in MQMD	Send Method
JMSReplyTo	ReplyToQ/ReplyToQMGR in RFH2	Message Object
JMSRedelivered	Not Applicable	Receive-only

Sending Messages from WAS to non-JMS systems

Resources → JMS → Queues →
[Select Queue] → Advanced properties
(From WAS v8)



Message descriptor

- ☐ MQMD read enabled
- ☐ MQMD write enabled

Additional

MQMD message context
DEFAULT

- When sending messages which are to be consumed by non-JMS applications, you may want to control the format and header data of the WebSphere MQ message.
- To achieve for some properties (those starting with the name “JMS_IBM_”) requires additional configuration of the Destination object to allow reading and writing of the MQMD.
- MQMD property values are accessed via Java properties on the JMS Message, the names of which are detailed in the WebSphere MQ Information Center under the heading “Mapping JMS fields onto WebSphere MQ fields (outgoing messages)”.
- Some properties cannot be set by the application when sending the message. Instead these properties are updated in the JMS message by the sending operation itself.
- Use with caution!***

JMS provider specific property name	Field and header used for transmission	Set by
JMS_IBM_MsgType	MsgType in MQMD	Message Object
JMS_IBM_PutTime	PutTime in MQMD	Send method



Read Ahead

- In general messages are sent to a JMS application when the application requests it – one at a time.
- However, an asynchronous JMS consumer can be configured to receive more than one message, using the feature of 'Read Ahead'.

Resources → JMS → Queues →
[Select Queue] → Advanced properties

Optimizations

Asynchronously send messages to the queue manager

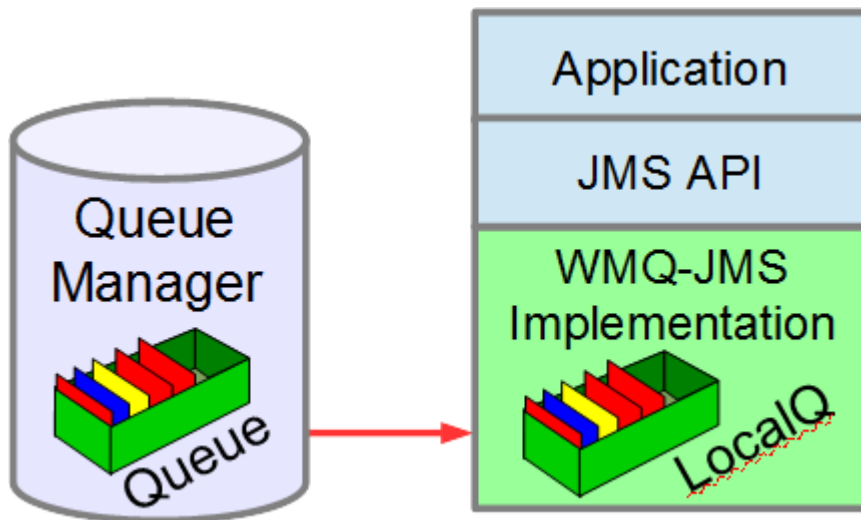
As per queue definition

Read ahead, and cache, non-persistent messages for consumers

As per queue definition

Read ahead consumer close method

Wait for all cached messages to be delivered



- In order for Read Ahead to operate, the following conditions must be met:
 - Using non-persistent messages
 - OR not destructively consuming messages – for example browsing for messages with an Activation Specification.
- Note that if the application terminates unexpectedly, all unconsumed non-persistent messages are discarded.

JMS User Authentication

- In JEE, JMS application-specified user and passwords are not necessarily used. Instead, “Container-managed” authentication is deployed.
 - Activation Specifications / Connection Factories can be associated with authentication data.
 - JAAS – J2C Authentication Data defines username and password details.
 - The application needs to use JEE resource-references to access the connection factory
 - The ***authenticationType*** parameter needs to be set to container for container-managed authentication.
-
- As for other WMQ clients, security exits are required to validate passwords, WMQ only checks user id.
 - User IDs became more important with the introduction WMQ 7.1 channel authentication.
 - Read the following Technote for details:
<http://www.ibm.com/support/docview.wss?uid=swg21580097>

Security settings

Select the authentication values for this resource.

Authentication alias for XA recovery

localhostNode03/MyUserID

Mapping-configuration alias

DefaultPrincipalMapping

Container-managed authentication alias

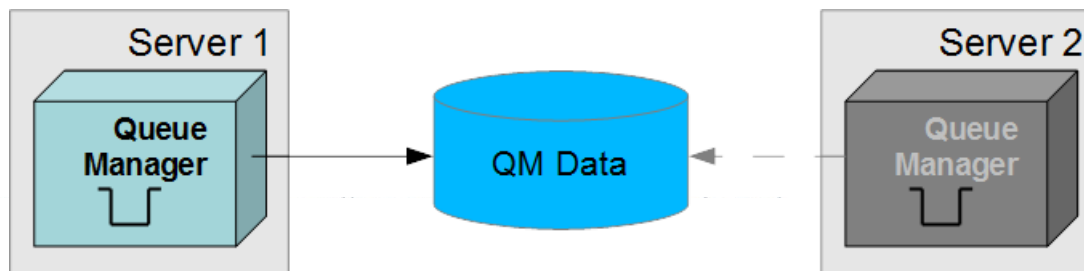
localhostNode03/MyUserID

Related Items

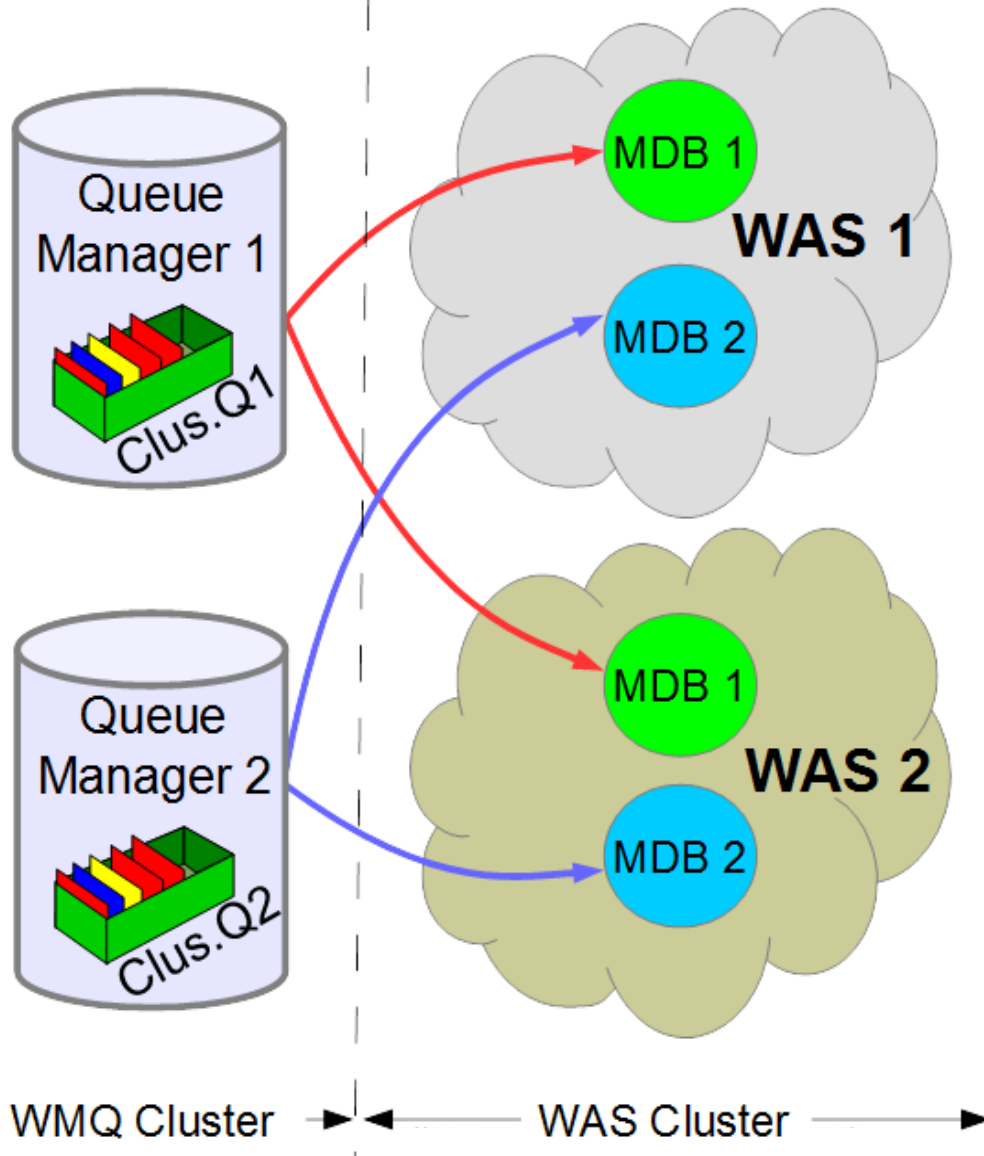
- [JAAS - J2C authentication data](#)

High Availability – Multi-instance Queue Managers

- WMQ's implementation of Active-Passive failover is the Multi-Instance Queue Manager, which utilises queue manager data stored in network storage system. It requires:
 - OS = Linux, UNIX or Windows
 - WMQ 7.0.1 or later Queue Manager
 - WMQ-RA 7.0.1.3 or later (included within WAS 7.0.0.13)
- WAS Connection Factories/Activation Specifications must be configured to locate the queue manager at multiple network addresses. This is achieved using:
 - Client Connection Definition Table (CCDT) or
 - Connection Name Lists (a comma separate list of hostnames and port numbers, of the form: “hostname1(port1), hostname2(port2)”).
- Automatic Client Reconnect is **not supported** from within the EJB/Web container.



High Availability – Resilient Configurations

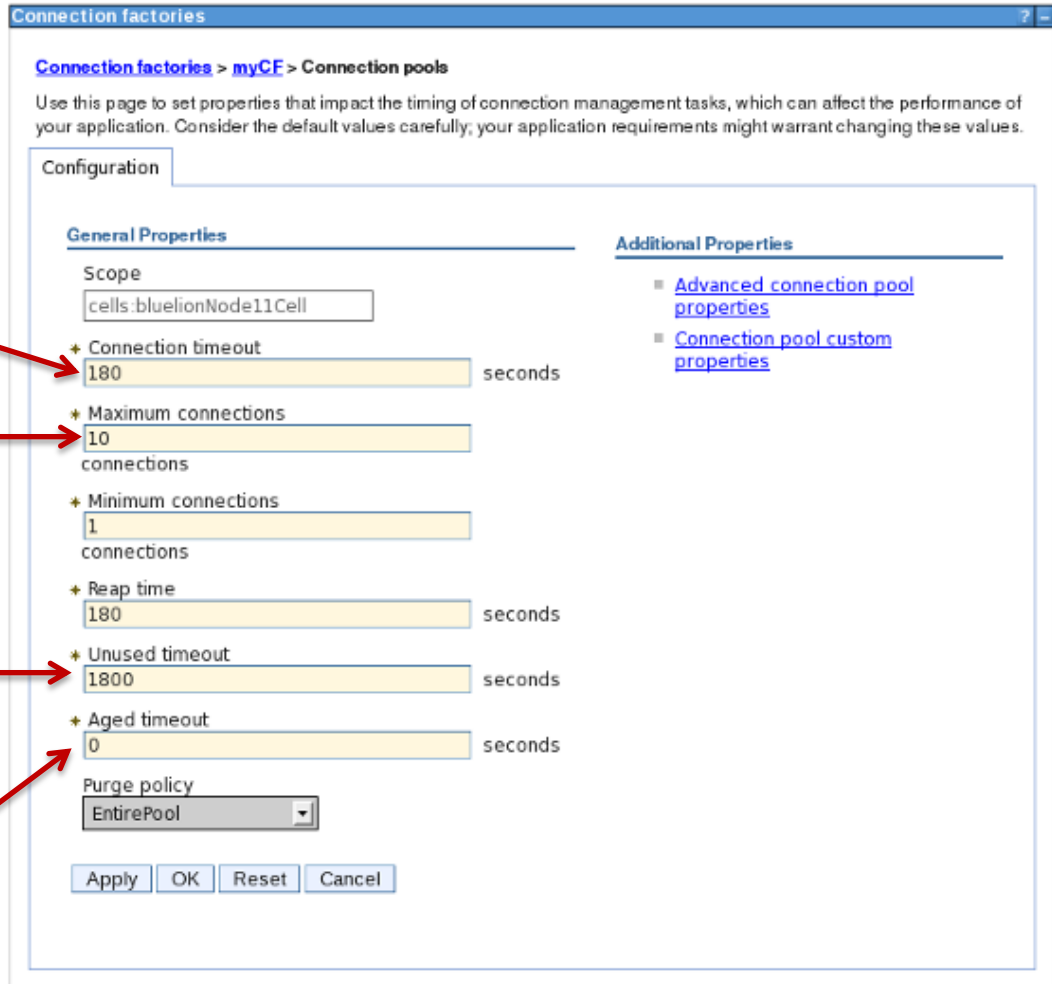


- Design the capacity of the WAS MDBs to be sufficient such that one MDB can manage the entire workload.
- Then any one system can be removed from this design, and messages will continue to be consumed by the MDB with no loss of service – other than the set of messages which become unavailable if one of the Queue Managers goes down

Connection and Session Pooling

- Connection and Session pooling is provided by default by WAS for all Connections and Sessions created from a ConnectionFactory looked up from the WAS JNDI.
- Length of time to wait for a Connection when the pool is full
- Maximum number of Connections which can be created from this ConnectionFactory
- Length of time an idle Connection remains in the pool
- Length of time a Connection can exist for before being discarded when not in use. **Do not use** this property with WebSphere MQ Connection Factories unless you understand the consequences.

Resources → JMS → Connection factories →
[Select ConnectionFactory] → Connection pools



The screenshot shows the 'Connection factories' configuration page in WebSphere. The breadcrumb trail is 'Connection factories > myCF > Connection pools'. The page title is 'Connection pools'. The description states: 'Use this page to set properties that impact the timing of connection management tasks, which can affect the performance of your application. Consider the default values carefully; your application requirements might warrant changing these values.'

The 'Configuration' tab is selected. It contains two sections: 'General Properties' and 'Additional Properties'.

General Properties:

- Scope: cells:blueionNode11Cell
- Connection timeout: 180 seconds
- Maximum connections: 10 connections
- Minimum connections: 1 connections
- Reap time: 180 seconds
- Unused timeout: 1800 seconds
- Aged timeout: 0 seconds
- Purge policy: EntirePool (dropdown menu)

Additional Properties:

- [Advanced connection pool properties](#)
- [Connection pool custom properties](#)

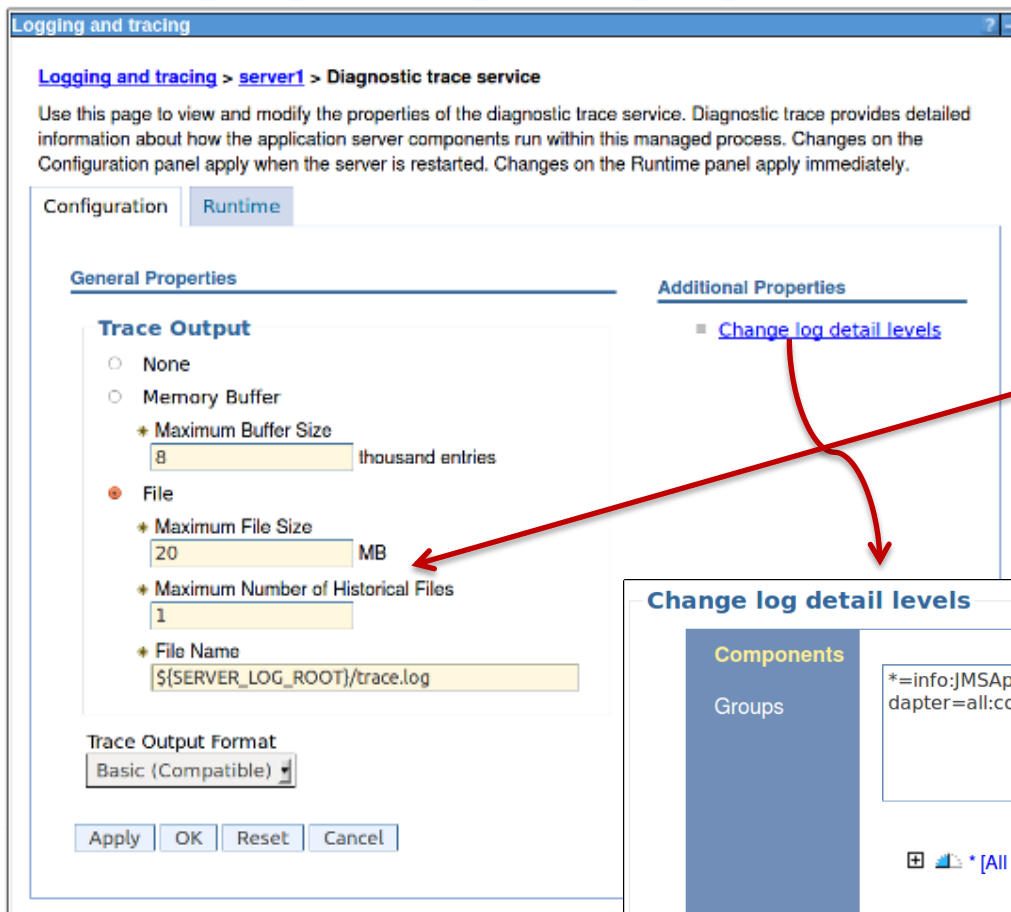
Buttons at the bottom: Apply, OK, Reset, Cancel.

Collecting Diagnostic Data – Enabling Trace

Trace for the WMQ-RA is integrated with the WAS trace system. In general, trace will be request by IBM Support teams when investigating reported problems.

Note the two tabs – “Configuration” and “Runtime”.

Troubleshooting → Logs and trace → [select server] → Diagnostic trace



Logging and tracing

Logging and tracing > server1 > Diagnostic trace service

Use this page to view and modify the properties of the diagnostic trace service. Diagnostic trace provides detailed information about how the application server components run within this managed process. Changes on the Configuration panel apply when the server is restarted. Changes on the Runtime panel apply immediately.

Configuration Runtime

General Properties

Trace Output

- ☐ None
- ☐ Memory Buffer
- ☒ File
 - Maximum Buffer Size: 8 thousand entries
 - Maximum File Size: 20 MB
 - Maximum Number of Historical Files: 1
 - File Name: \${SERVER_LOG_ROOT}/trace.log

Trace Output Format: Basic (Compatible)

Apply OK Reset Cancel

Additional Properties

- [Change log detail levels](#)

A suitable trace string for the WMQ-RA component is:

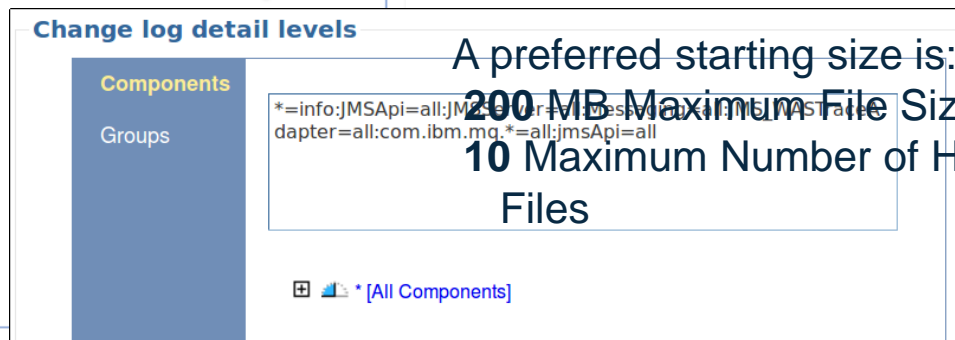
```
*=info:JMSApi=all:JMSServer=all:Messaging=all:JMS_WASTraceAdapter=all:com.ibm.mq.*=all:jmsApi=all
```

The default trace size of 2 log files (1 historical) and 20Mb maximum log size are normally too small to capture WMQ-RA issues.

A preferred starting size is:

200 MB Maximum File Size

10 Maximum Number of Historical Files



Change log detail levels

Components

Groups

=info:JMSApi=all:JMSServer=all:Messaging=all:JMS_WASTraceAdapter=all:com.ibm.mq.=all:jmsApi=all

* [All Components]

Investigating Your Own Applications

- The trace string used on the previous page results in the internals of the WMQ-RA being captured. This output is not intended for the end user to consume.
- Is there another trace string which produces output which might be of more use to the end user?
- Yes! See the information in the following technote:
<http://www.ibm.com/support/docview.wss?uid=swg21663645>

Collecting a trace of the JMS API calls made by an message-driven bean application.

Technote (FAQ)

Question

You have written a message-driven bean (MDB) application that runs inside of WebSphere Application Server. Inside the message-driven bean's `onMessage()` method, you have implemented some logic that uses the WebSphere Application Server WebSphere MQ messaging provider to communicate with a WebSphere MQ queue manager.

The message-driven bean application is not behaving as you expect it to. Are there any diagnostics that you can collect from the application server that shows what JMS API calls your application is making when communicating with WebSphere MQ?

Answer

WebSphere Application Server provides a diagnostic trace facility that can be used to diagnose problems. It is possible to configure this trace facility to generate trace information about the JMS API calls made by a message-driven bean application. This is useful for application developers who want to see the code path taken by their application.

To enable a trace of the JMS API calls made by a message-driven bean application, [set up a WebSphere Application Server trace](#) using the following trace string:

```
*=info:  
com.ibm.ejs.jms.JMSConnectionFactoryHandle=all:  
com.ibm.ejs.jms.JMSConnectionFactoryHandle=all:  
com.ibm.ejs.jms.JMSMessageConsumer=all:
```

Agenda

- Connecting WebSphere Application Server to the WebSphere MQ messaging Infrastructure
- Configuring the environment for JMS outbound messaging
- Inbound messaging
- Features of WebSphere MQ Resource Adapter
- **Common 'Gotchas'**
- Reference links

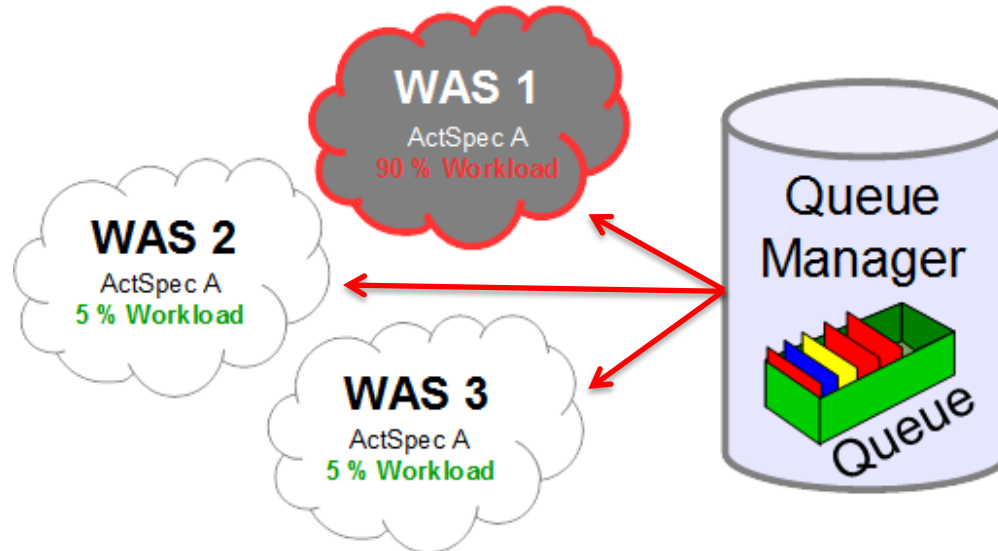
Application Server Hang on Startup

- If you have multiple MDBs configured within the same WAS server, and there are messages on the queues when the Activation Specification starts, WMQ-RA threads hang in a logical deadlock, and messages are not consumed.
- APAR IZ68236 describes this in detail – all WAS servers running with a WMQ-RA v7.0 are affected by this.
- The fix is a configuration change – 'connectionConcurrency' must be set to the value '1' in the WMQ-RA custom properties.
- **THIS MUST BE CONFIGURED AT THE CELL SCOPE.**

Resources → Resource Adapters →
WebSphere MQ Resource Adapter → Custom properties

Name	Value	Description	Required
You can administer the following resources:			
connectionConcurrency	5	connectionConcurrency	false
maxConnections	10	maxConnections	false
logWriterEnabled	true	logWriterEnabled	false
reconnectionRetryCount	5	reconnectionRetryCount	false
reconnectionRetryInterval	300000	reconnectionRetryInterval	false
traceEnabled	false	traceEnabled	false
traceLevel	3	traceLevel	false
Total 7			

Uneven Distribution of Messages



- The original design of the WebSphere MQ v7.0 Queue Manager when working with multiple message consumers was to saturate the first registered consumer before sending messages to the next message consumer.
- In the case of an Activation Specification with 10 Server Sessions defined, this means that the first 10 available messages would go to the one WAS server before other WAS servers received messages.
- **IZ97460** – included in WebSphere MQ 7.0.1.6 – changed this queue manager behaviour to a round-robin distribution.

Poor performance of MDBs

- There are a number of reasons why your MDBs may slow down, and these can be difficult to diagnose.
- Some general tips:
 1. Verify that 'migration mode' is not in use. This is a compatibility layer within the WMQ-RA v7 which allows communication with WMQ v6 Queue Managers. It supports none of the v7 features which enhance MDB performance (browse-with-mark to distributed queue managers).
 - You cannot determine directly if it is activated, however you can check for the following activating conditions:
 - a. A Connection Factory configured with “Provider Version” set to “6”.
 - b. Connecting to a v6 queue manager
 - c. Connection to a v7 queue manager with the CHANNEL property 'SHARECNV=0'
 2. Check the depth of the queue which the MDBs are consuming messages from. If it is deep (for example 10,000+ messages), check to see if message selectors are being used, or if the average message size is greater than 4Kb.

My WAS Connections/Sessions are not behaving as they are supposed to!

- Programmatically creating your ConnectionFactory in your application code means that you will bypass the WAS wrappers – meaning that your transactions will not be seen by the WAS transaction manager, and you will have no pooling of JMS Connections and JMS Sessions

```
Context jndiContext = new InitialContext();  
ConnectionFactory myConnectionFactory =  
    (ConnectionFactory)jndiContext.lookup("jms/myConnectionFactory");  
Connection conn = myConnectionFactory.createConnection();
```



```
MQConnectionFactory myConnectionFactory = new MQConnectionFactory();  
myConnectionFactory.setQueueManager("myQMGR");  
myConnectionFactory.setHostName("localhost");  
myConnectionFactory.setChannel("MY.CHANNEL.SVRCONN");  
myConnectionFactory.setPort(1414);  
myConnectionFactory.setTransportType(WMQConstants.WMQ_CM_CLIENT);  
  
Connection conn = myConnectionFactory.createConnection();
```



Agenda

- Connecting WebSphere Application Server to the WebSphere MQ messaging Infrastructure
- Configuring the environment for JMS outbound messaging
- Inbound messaging
- Features of WebSphere MQ Resource Adapter
- Common 'Gotchas'
- **Reference links**

Further Information

- WAS product information : <http://www.ibm.com/software/webservers/appserv/was/>
- WAS Information Centers :
 - 6.0 <http://publib.boulder.ibm.com/infocenter/wasinfo/v6r0/index.jsp>
 - 6.1 <http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/index.jsp>
 - 7.0 <http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/index.jsp>
 - 8.0 <http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/index.jsp>
 - 8.5 <http://publib.boulder.ibm.com/infocenter/wasinfo/v8r5/index.jsp>
- Product Connectivity Information Center
 - <http://publib.boulder.ibm.com/infocenter/prodconn/v1r0m0/index.jsp>
- Using WebSphere MQ Java Interfaces in J2EE/JEE Environments
 - <http://www.ibm.com/support/docview.wss?rs=171&uid=swg21266535>
- IBM developerWorks : <http://www.ibm.com/developerworks>
 - (Searching on “Service Integration Bus” returns a number of interesting articles)
 - http://www.ibm.com/developerworks/websphere/techjournal/0901_leeming/0901_leeming.html – WASV7
 - http://www.ibm.com/developerworks/websphere/techjournal/0601_ratnasinghe/0601_ratnasinghe.html - Security
 - http://www.ibm.com/developerworks/websphere/techjournal/0601_smithson/0601_smithson.html - Security
- IBM RedBooks : <http://www.redbooks.ibm.com>
 - WebSphere Application Server V7: Messaging Administration Guide SG24-7770-00
 - WebSphere Application Server V7: Concepts, Planning and Design, SG24-7708-00
 - WebSphere Application Server V7: Technical Overview, REDP-4482-00
 - WebSphere Application Server V6.1: JMS Problem Determination, REDP-4330-00
 - WebSphere Application Server V6.1: System Management & Configuration, SG24-7304-00
 - WebSphere Application Server V6 Scalability and Performance Handbook, SG24-6392-00
 - WebSphere Application Server V6.1 Security Handbook, SG24-6316-01
 - WebSphere Application Server V6.1: Technical Overview, REDP-4191-00
 - WebSphere Application Server V6.1: Planning and Design, SG24-7305-00
 - WebSphere Application Server V6.1: Installation Problem Determination, REDP-4305-00

This was Session #17892. The rest of the week ...

	Monday	Tuesday	Wednesday	Thursday	Friday
08:30			MQ for z/OS, Using and Abusing New Hardware and the New v8 Features	Nobody Uses Files Any More Do They? New Technologies for Old Technology, File Processing in MQ MFT and IIB	Monitoring and Auditing MQ
					Securing MQ Initiated CICS Workload
10:00	Introduction to MQ - Can MQ Really Make My Life Easier?	MQ for z/OS: The Insider Story	IBM Integration Bus MQ Flexibility	Common Problems and Problem Determination for MQ z/OS	IBM MQ and IBM Integration Bus - from Migration and Maintenance to Continuous Enhancements, How and Why to Stay Current
11:15	Introduction to IBM Integration Bus on z/OS	Introduction to the New MQ Appliance	MQ V8 Hands-on Labs! MQ V8 with CICS and COBOL! MQ SMF Labs!		
12:15					
1:45	What's New in the Messaging Family - MQ v8 and More		Getting Started with Performance of MQ on z/OS	IBM MQ: Are z/OS & Distributed Platforms Like Oil & Water?	
3:15	What's New in IBM Integration Bus	Live!: End to End Security of My Queue Manager on z/OS	Digging into the MQ SMF Data	MQ Parallel Sysplex Exploitation, Getting the Best Availability from MQ on z/OS by Using Shared Queues	
		Application Programming with MQ Verbs			
4:30	MQ Security: New v8 Features Deep Dive	Live!: What's the Cloud Going to Do to My MQ Network?	Giving It the Beans: Using IBM MQ as the Messaging Provider for JEE Applications in IBM WebSphere Application Server	Challenge the MQ & IIB Experts	
		The Do's and Don'ts of IBM Integration Bus Performance			



Copyright and Trademarks

© IBM Corporation 2015. All Rights Reserved.

IBM, the IBM logo, and ibm.com 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 IBM trademarks is available on the Web at “Copyright and trademark information” at www.ibm.com/legal/copytrade.shtml.