WebSphere

WAS Messaging: overview and direction

SIBus, WMQ and SCA

Graham Wallis, STSM, IBM

March 2009

© 2009 IBM Corporation

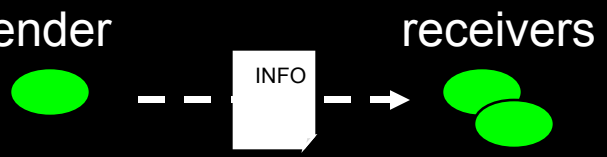
Contents

- JMS messaging
- Using SIB
- Using WMQ
- Thin JMS Clients
- WPS & WESB SCA messaging
- WPS & WESB scaling
- Recent developments
- Future directions

© 2009 IBM Corporation

JMS – Java Messaging Service

- **Allows application interactions to be loosely coupled**
  - decouple sender and receiver
  - temporal decoupling
  - transactional decoupling
  - routing, transformation possible
- **Supports messaging domains**
  - point-to-point - uses a queue
  - publish-subscribe – uses a topic (think of a bulletin board)
- **Supports different types of message payload**
  - Text, Bytes, Object, Map, Stream
- **Programming interface**
  - not a transport protocol
  - JMS interface implemented by a “JMS Provider”

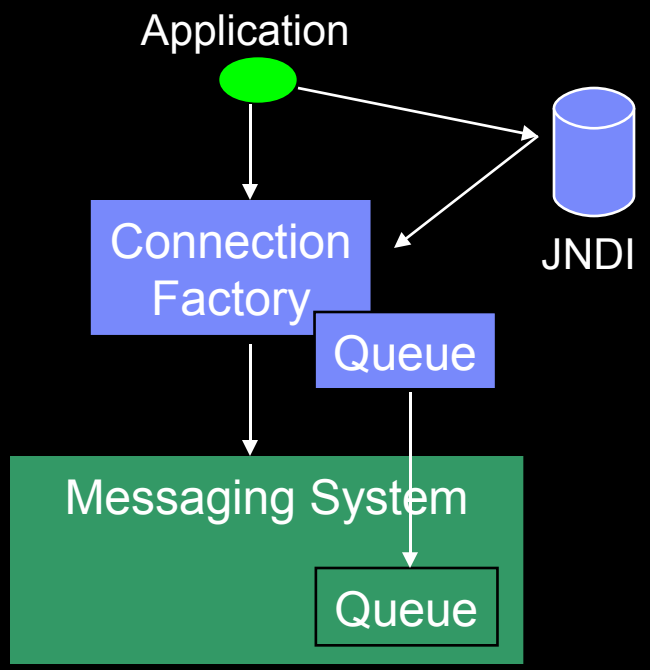


© 2009 IBM Corporation

JMS Resources

- **Access to the messaging system is via JMS resources**
  - administered objects looked up in JNDI
  - they provide an abstraction that hides the provider differences from the app
  - a resource maps from (portable) JMS onto a particular messaging system
  - a resource is provider specific

- JMS Connection Factory
  - To create an outbound connection to the provider
- JMS Activation Specification
  - To create a connection for inbound message-driven processing (MDBs)
- JMS Queue
  - For point to point, queued messaging
- JMS Topic
  - For publish subscribe, topic based messaging



© 2009 IBM Corporation

JMS – Outline of a typical application

```
import javax.naming.*;
import javax.jms.*;

Context messaging = new InitialContext();
ConnectionFactory connectionFactory =
    (ConnectionFactory)messaging.lookup("myConnectionFactory");
Queue stockQueue = (Queue)messaging.lookup("myQueue");

Connection connection = ConnectionFactory.createConnection();
Session session =
    connection.createSession(false,Session.AUTO_ACKNOWLEDGE);
MessageProducer sender = session.createProducer(stockQueue);
Message message = session.createTextMessage();
message.setText(myString);
sender.send(message);
```


© 2009 IBM Corporation

JMS Providers

- **WAS contains a choice of JMS providers**
  - you can connect to different messaging systems
  - choice of JMS Provider is per-application*
  - they are not mutually exclusive
- **[Simple] choice depends on where the queue/topic is…**
  - If it's a SIB destination
    - use the default (SIB) JMS provider (see "Using SIB" slide)
  - If it's a WMQ Queue
    - you can use it directly or indirectly (see "Using WMQ" slide)
  - If you are creating the destination
    - you can choose which messaging system to use (see "Choosing…" slide)
  - If it's something else – use that…

* - actually it's per-connection, but that would be rare

© 2009 IBM Corporation

Choosing a messaging system

SIB	WMQ
Included with WAS (c. 2005) Integrated with WAS runtime, configuration and console	Separate product (c. 1993) Own runtime, admin domain and explorer interface
Growing customer base	Massive installed base
Supports WESB, WPS	Supports WMB, FTE
Provides point-point & publish-subscribe messaging models	
Supports JMS	Supports multiple programming languages
Largest message ~ 40MB	Largest message: **v6=100MB** **v7="unlimited"**
Limited tooling support	Widespread tooling support

© 2009 IBM Corporation

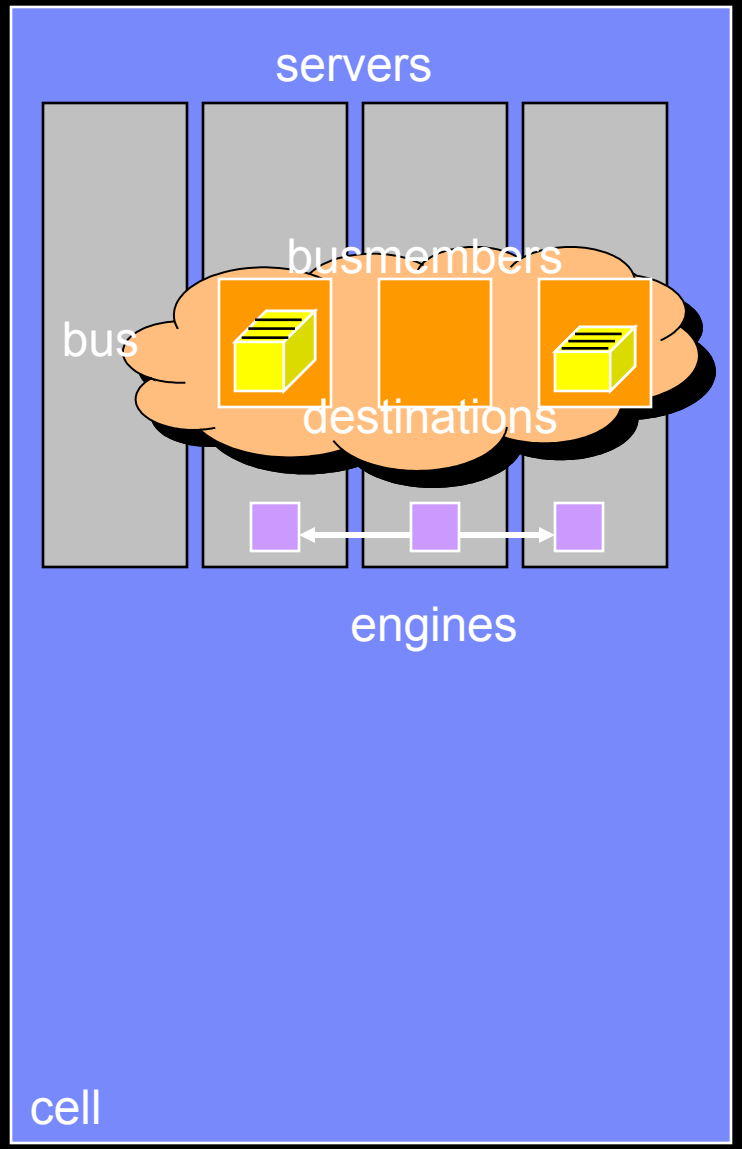
Using SIB

- **The default JMS provider resources refer to SIBus**
- **SIBus:**
  - Integrated messaging system
    - configured in the WAS admin model
    - runs in the WAS runtime
    - exploits WAS clustering model
  - Includes point-to-point and publish subscribe
  - Provides a broad range of QoS options
  - Uses JCA RA for delivery of inbound messages to MDBs
    - configured by activation specs

© 2009 IBM Corporation

SIB concepts

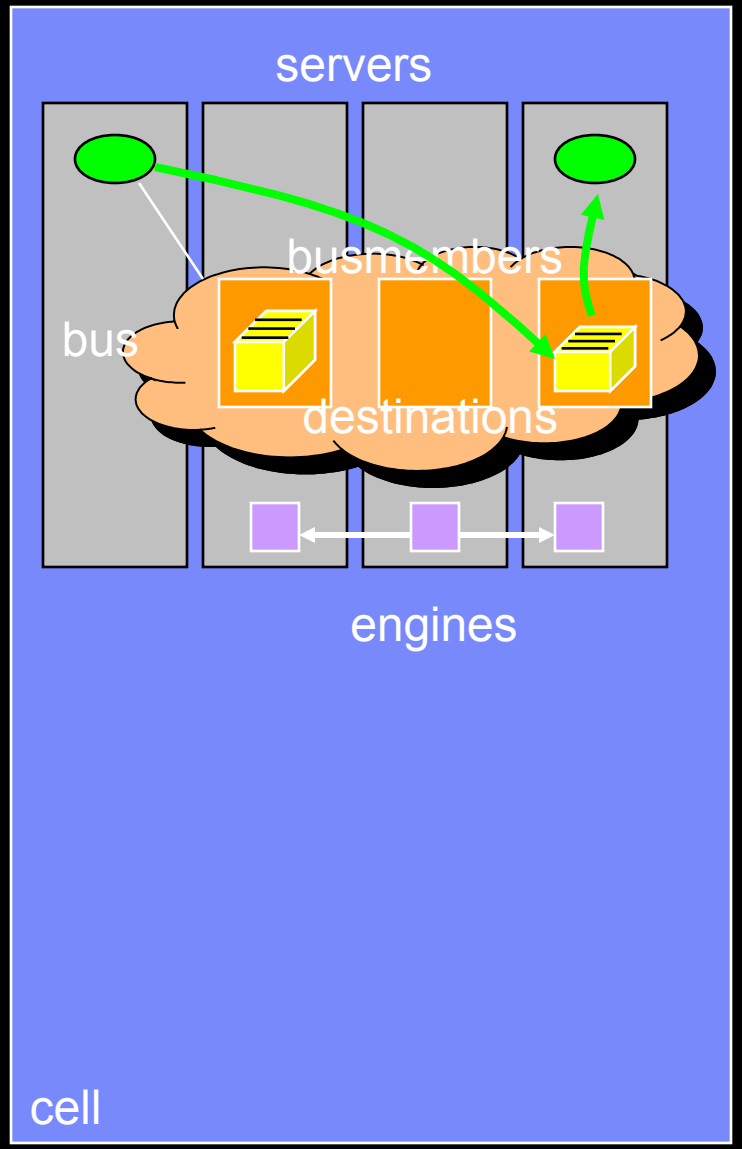
- **Bus**
- **A bus is a logical entity that:**
  - contains destinations
  - is location transparent
- **A bus exists within a cell**
  - It cannot extend beyond the cell
- Physically:
  - the cell contains servers (or clusters)
  - some of which are busmembers
    - a destination is assigned to a busmember
  - busmembers run messaging engines
  - engines manage the runtime states of destinations



© 2009 IBM Corporation

SIB concepts

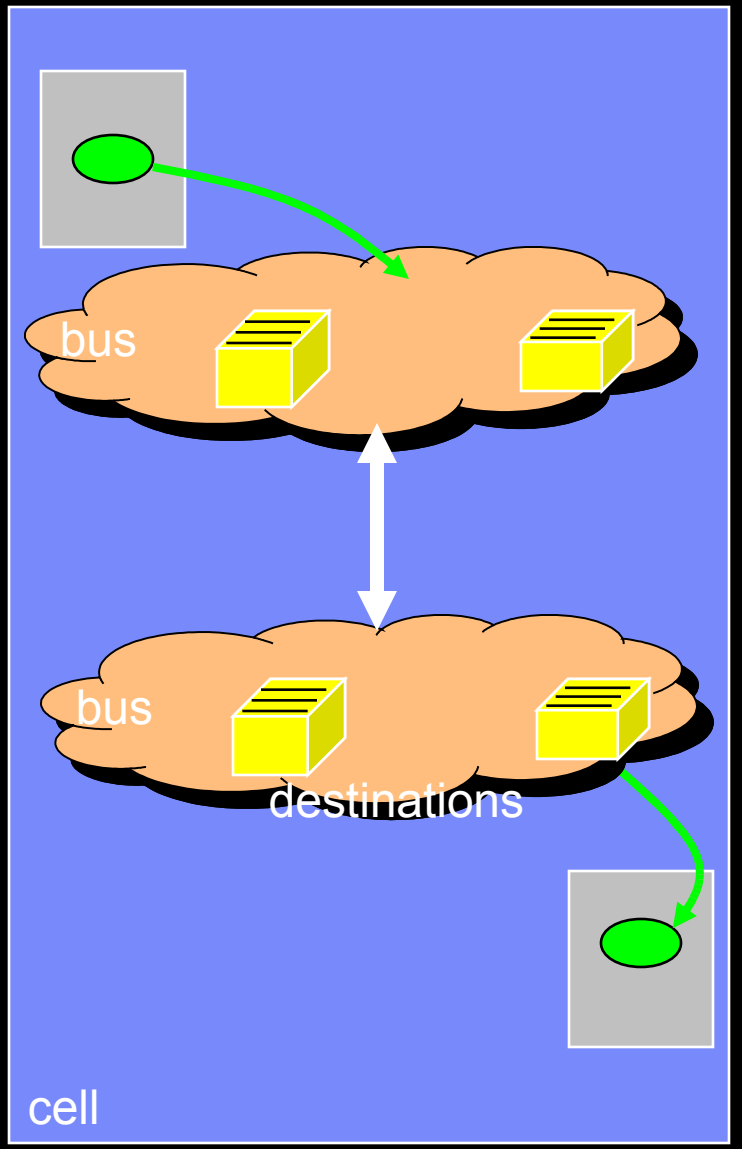
- **Logically, an application:**
  - creates a connection to the bus
    - at any busmember
  - creates a producer and/or consumer session
    - attached to a destination
  - sends and/or receives messages
- Physically:
  - the connection is handled by a messaging engine (in a bus member)
  - the engine routes the message to the destination
  - the engine handles persistence, etc.



© 2009 IBM Corporation

SIB concepts

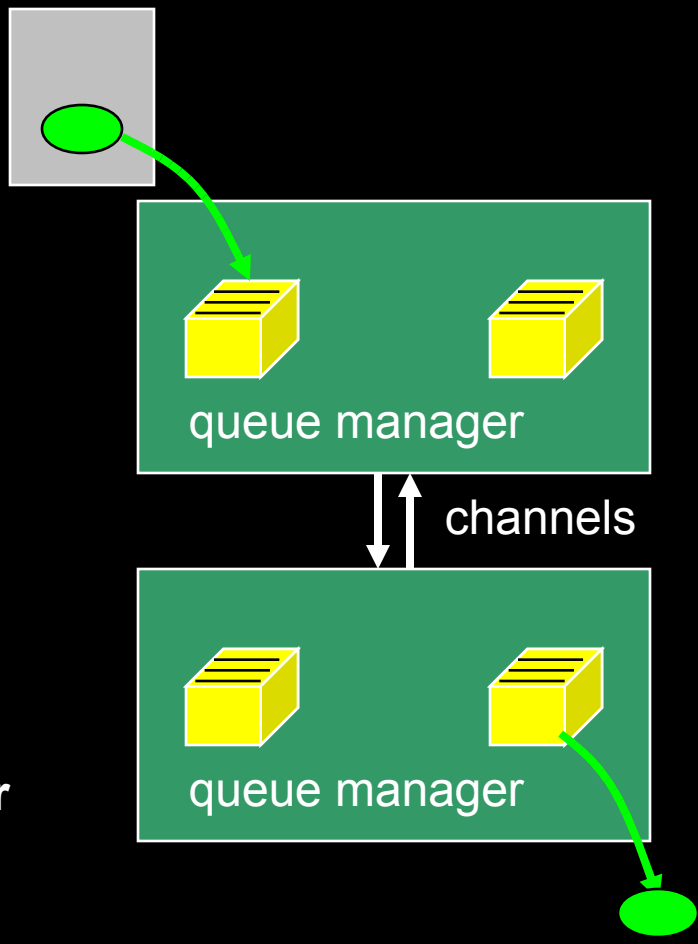
- **A cell can have multiple buses**
  - separate buses provide isolation
- **Buses can be linked**
  - create a Foreign Bus Connection (to another SIBus)
  - buses in the same cell
  - buses in different cells
- **An application can send to a destination in a linked bus**
- **Physically a link is managed by a pair of messaging engines**
- **A bus has a security model**
  - who can connect to the bus
  - who can send to a destination
  - who can receive from a destination



© 2009 IBM Corporation

Using WMQ – what is WMQ?

- **WMQ is an IBM messaging product**
  - Huge installed base and tooling support
  - Can connects just about anything
  - Supports multiple programming languages
- **Queues are managed by queue managers**
- **Queue managers can be connected to one another**
  - Using channels
- **An application connects to a queue manager**
  - Using a client connection
    - local binding (API) or a socket connection

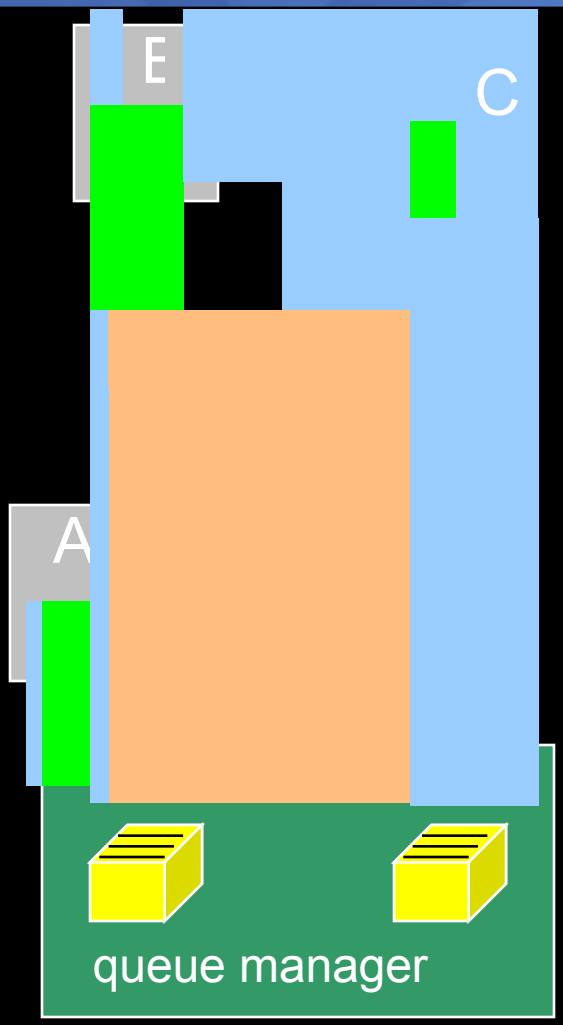


© 2009 IBM Corporation

WAS using WMQ

- **There are three ways of WAS using WMQ:**

	Direct?	Synch?	Connection type	JMS provider
A	Direct	Synch	Client connection	WMQ
B	Indirect			SIB
C		Asynch (store & forward)	Sender & Receiver channels	

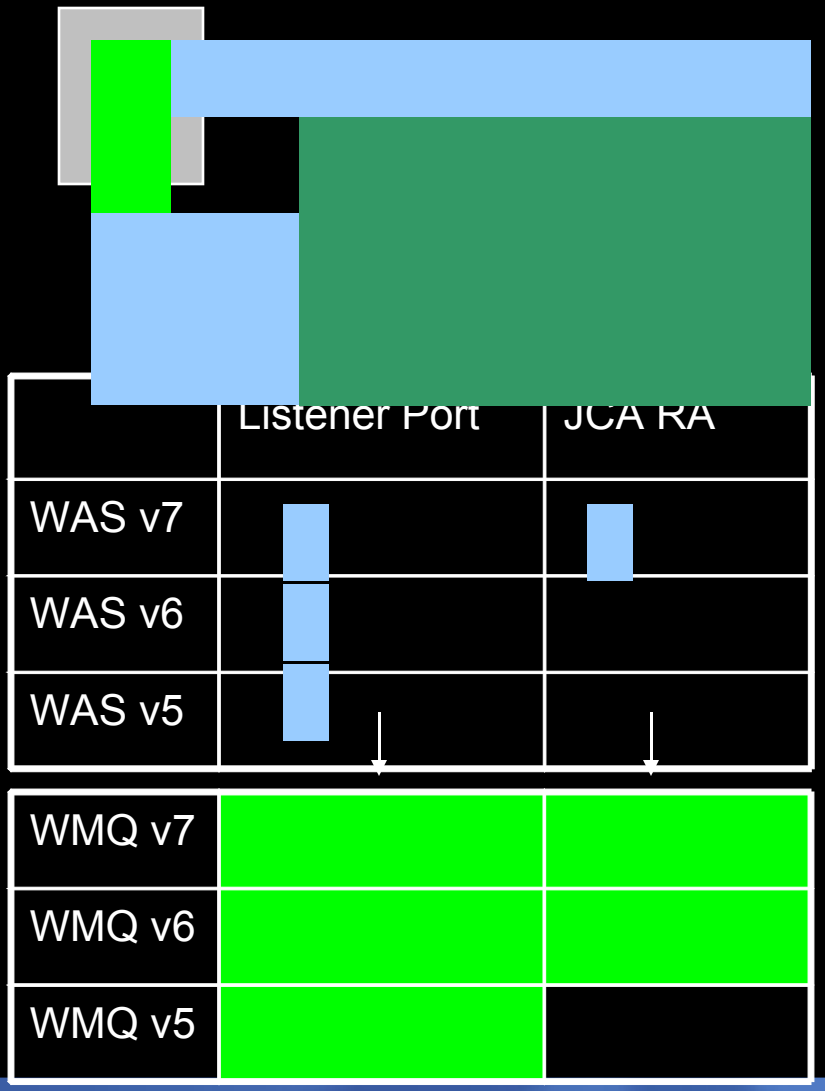


- Explained more on the coming slides…

© 2009 IBM Corporation

Option A: Using WMQ via the WMQ JMS Provider

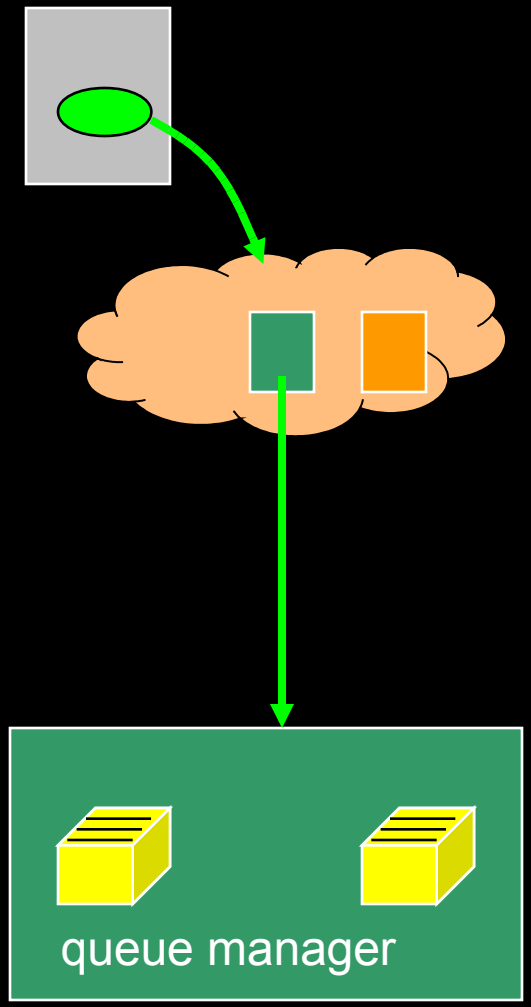
- **Direct – no transcoding of messages**
- **The WAS v7 WMQ provider is based on a JCA 1.5 RA**
  - cognisant of multi-regions in WAS/z and zWLM
- **There are two ways of configuring inbound message delivery to MDBs**
- **Activation Spec (preferred)**
  - JCA 1.5 compliant
  - can be defined at any scope, e.g. cell
  - requires definition of activation spec and destination
- **Listener Port (deprecated)**
  - introduced in WAS v5
  - uses ASF portion of JMS spec
  - server scoped only
  - requires definition of a connection factory, destination and listener port
  - although deprecated, no plans to remove any time soon
  - there is a migration utility (migrateWMQMLP)



© 2009 IBM Corporation

Option B: Using WMQ synchronously via SIB JMS Provider

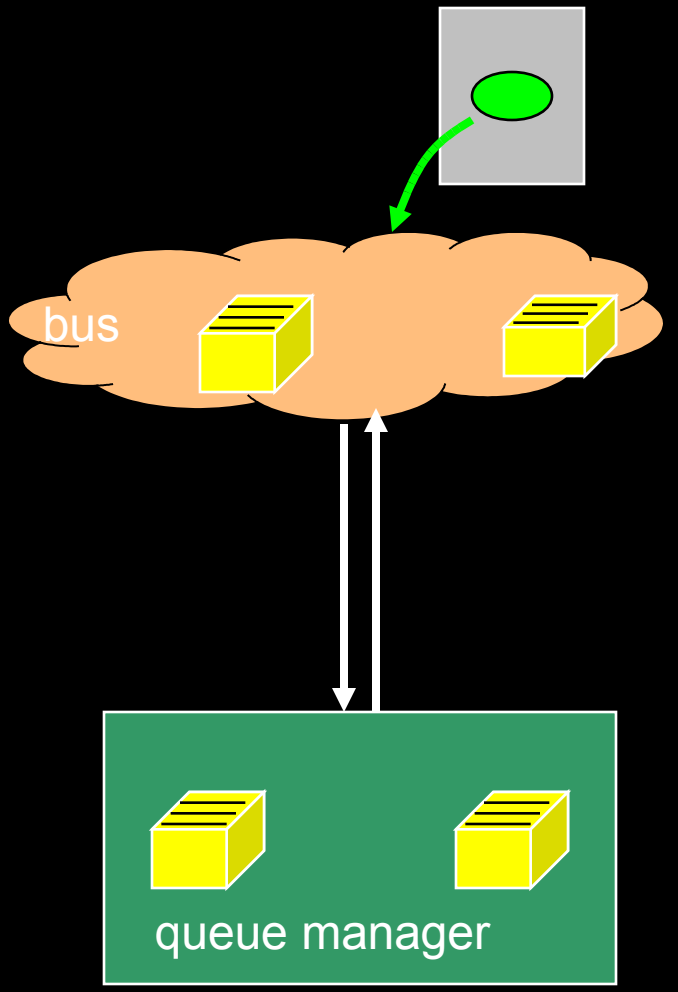
- **Indirect**
  - uses the SIB JMS Provider
  - messages are transcoded
- **WMQ queue manager becomes a virtual busmember**
- **Create a WMQ Server to represent the Queue Manager**
- **Make the WMQ Server a busmember**
- **Supports producers and consumers**
- **Transaction support has affinity, so good for access to WMQ Shared Queue**
- **From WASv7 this is supported with WMQv7 on all platforms**
  - With WASv6.1, WMQ had to be on z/OS



© 2009 IBM Corporation

Option C: Using WMQ asynchronously via SIBus

- **Indirect:**
  - uses the SIB JMS Provider
  - messages are transcoded
- **Uses SIBus**
  - provides store and forward
- **Create a Foreign Bus Connection to WMQ**
  - Connects to WMQ using Sender & Receiver channels
  - Queue Manager looks like another bus
  - SIBus looks like a queue manager
- **Supports producers only**
- **Publish-subscribe uses the PubSubBridge**
- **Implemented in SIBus by "MQLink"**



© 2009 IBM Corporation

JMS – Thin Clients

- **From a J2SE environment**
- **To connect to SIB, use the SIB JMS client**
  - Shipped in the WAS runtimes/ directory
  - Footprint < 2MB
  - v7 client is supported in: Geronimo, WAS CE and JBoss, Lotus Expeditor
- **To connect to WMQ, use the WMQ JMS client**
  - Downloadable from the WMQ web pages
- **There is a legacy capability for connecting a V5 default messaging (WMQ) client to the V6+ default messaging provider**
  - called the “MQClientLink”
  - but you don’t want to use it.

© 2009 IBM Corporation

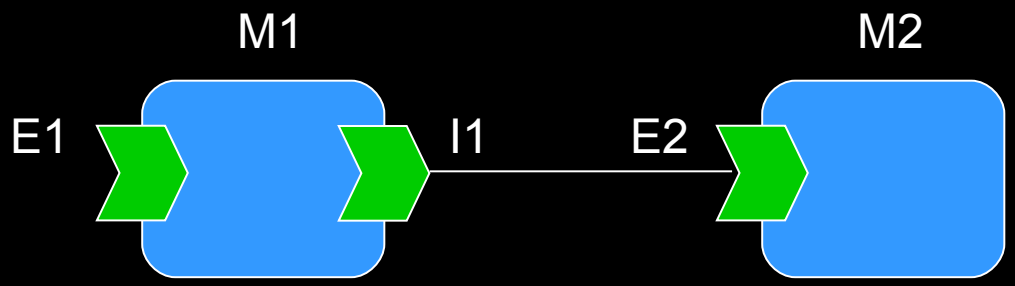
WESB and WPS use of SIBus

- **The WebSphere Process Server and WebSphere ESB products run SCA modules containing SCA components**
- **SIBus provides an asynchronous transport for SCA components**
- **It's used for:**
  - Explicit JMS bindings
  - Asynchronous Message Exchange Patterns
- **Bus configuration is performed by WBI tooling**

© 2009 IBM Corporation

WESB and WPS use of SIBus

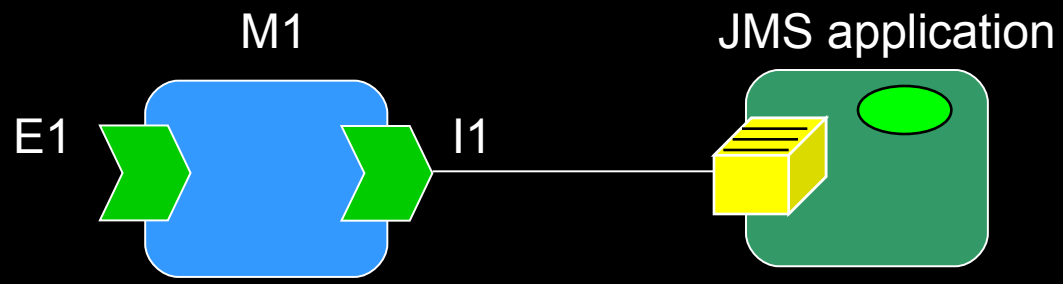
Suppose my business process looks like this:



Default binding (asynch)

Uses SIBus

Or this:



Explicit JMS binding (on the import)

Uses SIB or WMQ…

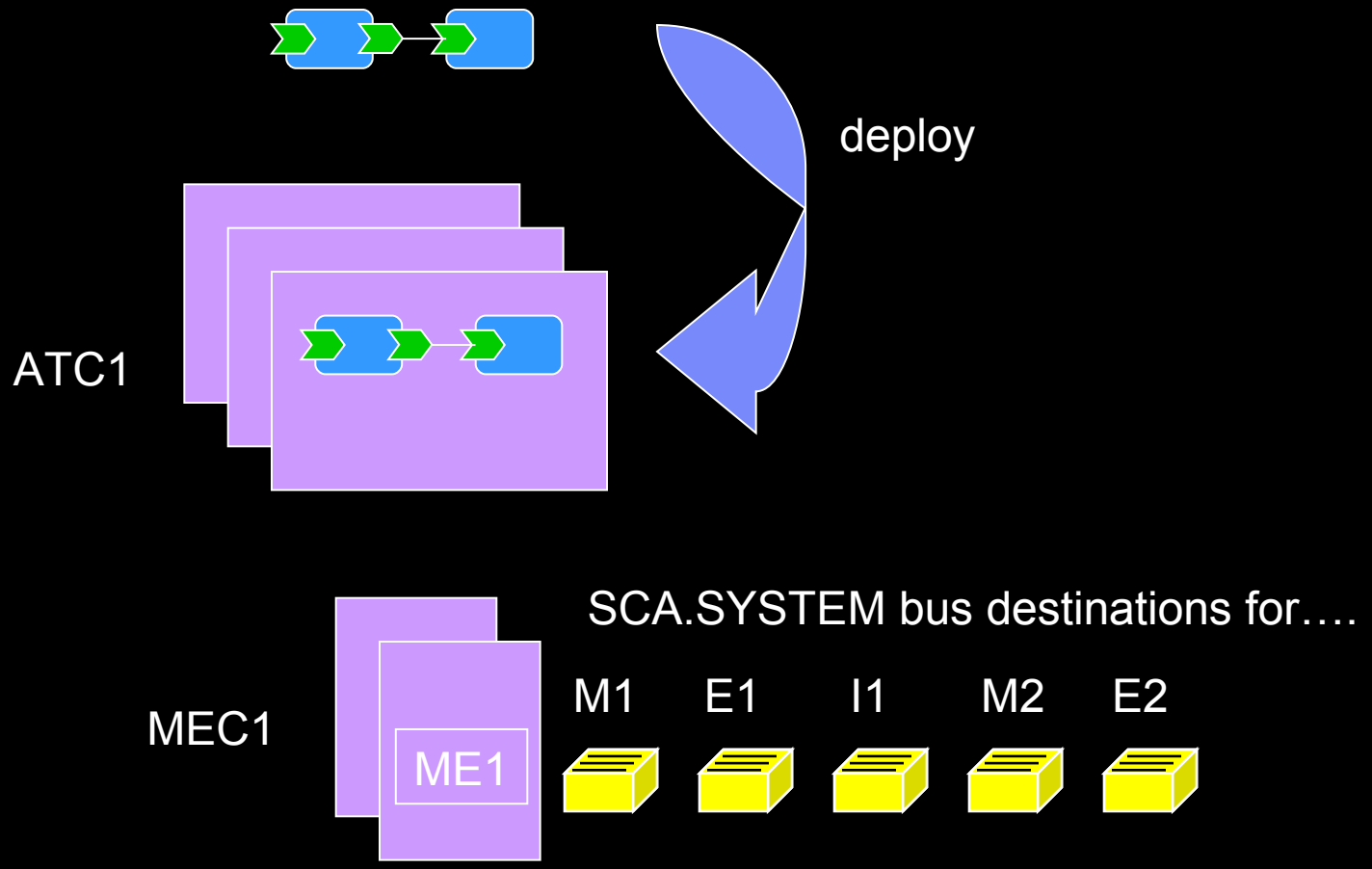
© 2009 IBM Corporation

WESB and WPS use of SIBus

- **Current releases create separate buses for:**
  - SCA Applications
  - SCA System
  - Business Process Monitoring
  - Common Event Infrastructure
- **Early releases created many SIB destinations**
  - But this has been reduced
- **A WPS/WESB topology has:**
  - Application Target Clusters (ATCs)
  - Messaging Engine Clusters (MECs)
  - (It can also have service clusters)
- **Different patterns suit different goals**
  - Collocated with one ME per server; app parallelism (but not message order)
  - Distributed with one ME; app parallelism and message order
  - The “Golden” topology is a distributed one that has one ATC and one MEC with a single ME

© 2009 IBM Corporation

Golden Topology (ATC 1:1 MEC)



© 2009 IBM Corporation

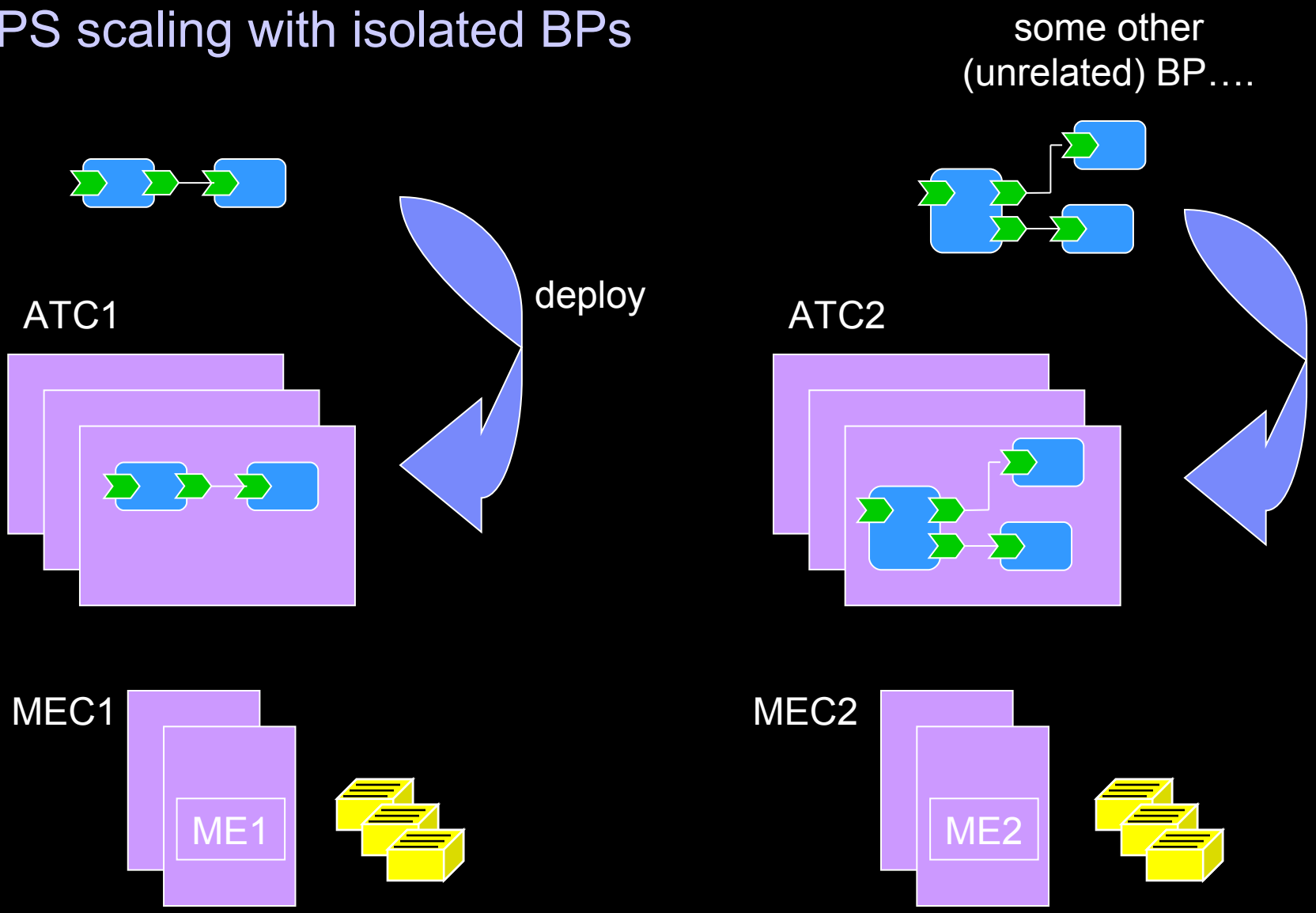
WPS scaling

- **There are two reasons why customers have extended their topologies**
  - Memory consumption in app cluster JVMs
    - Deployment of complex BP flows to one app cluster has led to memory outages
    - One app cluster can host:
      - Lots of small modules
      - Fewer large modules
  - SIB ME startup time
    - Lots of modules/components leads to lots of SIB destinations
    - This affects the startup time for the ME
    - By adding another MEC you can spread the destinations thinner
    - Helps startup time, but can complicate things (later slide)

- **Basic approach is to replicate the Golden Topology**
- **It is also possible to use multiple app clusters with one bus member cluster**
  - From a connectivity viewpoint it is similar to a single ATC and single MEC
  - However - watch that the MEC doesn't become overloaded

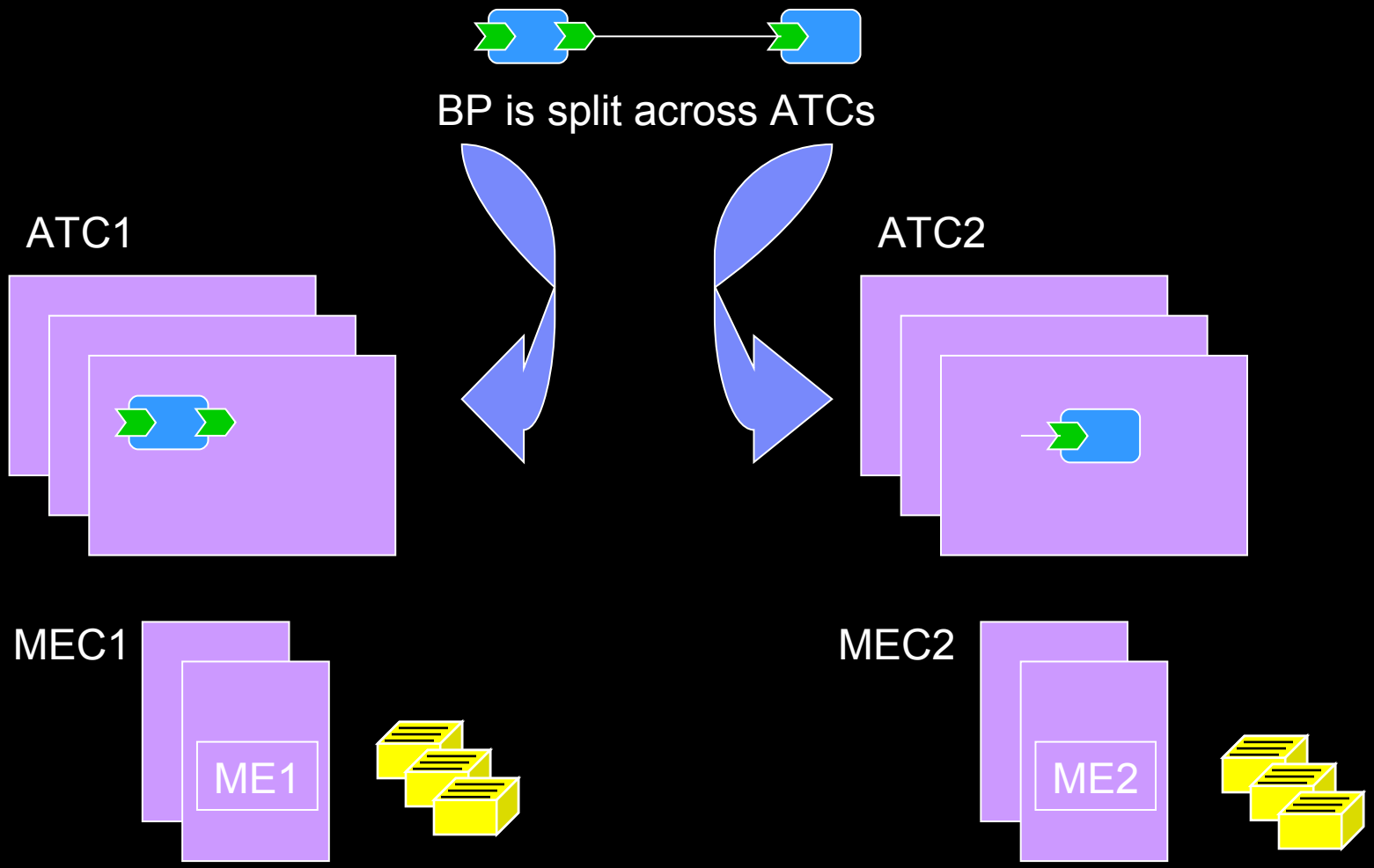
© 2009 IBM Corporation

WPS scaling with isolated BPs



© 2009 IBM Corporation

WPS scaling with complex BPs



© 2009 IBM Corporation

WPS scaling – additional considerations

- **The introduction of a second bus member introduces some additional considerations**
- **Location transparency can surprise people**
  - e.g. people have thought that “the queue point has moved”, when in fact the bus is just doing store and forward
- **Workload balancing of connections can lead to unintended connection patterns**
  - It is possible to control the degree of workload balancing by setting Target properties
  - Configuration of connection factories and activation specs
  - Configuration of custom properties for SCA (core) connections
- **If message order is important, it is necessary to restrict connectivity patterns to maintain message order**
  - Setting of target properties as above for limitation of workload balancing
  - Connection Factories, Activation Specs and SCA bindings should use “required” targets
  - Event Sequencing code itself always sets target properties (no configuration needed)

© 2009 IBM Corporation

WPS scaling – best practice

- **If you exceed capacity of ATC or MEC then divide apps**
  - Load app cluster JVMs as far as possible, then move additional apps to other clusters
  - Segment application workload between different ATCs
- **Retain ATC 1:1 MEC ratio**
  - Stick with this as a repeating pattern in which a ‘unit’ is one appcluster and its ME cluster
  - Add an MEC per ATC and split apps again if MEC overloaded
- **Avoid multi-ME clusters**
  - partitioned destinations make marooned messages more likely and make it impossible to do event sequencing
- **Where possible target communication to local ME for efficiency**
  - Collocate modules that interact
  - Inbound asynch (MDB) should be local (use target significance of “required”)
  - Some ‘outbound’ (send/receive) will go remote
  - e.g. module interacting with a remote module (in another cluster)
  - Use target significance of “preferred” so transaction can complete using S&F
- **Try to avoid creating cells or buses with lots of members**
  - May need to split coregroup or cell if enough JVMs (c. 100)
  - Optionally reduce bus size by splitting cell or rename buses

© 2009 IBM Corporation

WAS v7 messaging update

- **Updates to WMQ provider shipped with WAS v7**
  - Based on a JCA 1.5 RA supporting activation specs
  - New wizard for creating connection factories or activation specs, including
    - better integration of SSL support
    - support for defining a CF or AS based on a CCDT
    - support for message compression (headers and payload)
    - support for channel exits
  - Admin commands e.g. for configuring resources
- **Improvements to "single-bus" WMQ interop**
  - ability to use WMQ Server busmember to WMQv7 on distributed platforms
- **Improved access to clustered destinations**
  - new options for message routing – e.g. scoping of an alias to a subset of partitions
  - new options for message visibility
  - improved response routing
  - more flexible MDB activations
  - improved load balancing
- **Easier configuration of SIBus**
  - Cluster configuration wizard
  - Foreign Bus Connection wizard
  - SIBus security simpler and SIBus authorizations panels
- **Easier monitoring of SIBus**
  - Link transmitters
  - Navigation from Apps to SIB resources
  - Improved notification of MDB message retries; MDB can be stopped as a result of sequential failed messages
- **Performance improvements to SIBus**
  - Especially for large, persistent messages

© 2009 IBM Corporation

What's in store…

- **SIBus Resilience: failure handling, exception destination utilities, cleanup**
- **SIBus Performance: footprint, startup time, throughput, tuning guidance**
- **Extended SIBus topologies: tunnelling, scalability, flexible naming**
- **Ease of configuration and management of ME persistent stores**
- **Improved monitoring of SIBus**
- **Better synchronisation during startup & shutdown**
- **WMQ interoperability, updates and functional improvements**
- **Security context propagation**
- **Administrative improvements, simplification, versatility**
- **SCA JMS bindings**
  - Current releases use an MDB; in future expect it to be more direct
- **SCA scaling: improvements to destination scalability**

© 2009 IBM Corporation

Additional information

- **WAS Information Center**
  - http://www.ibm.com/software/webservers/appserv/was/library
- **Types of JMS provider**
  - http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/index.jsp?topic=/com.ibm.websphere.nd.doc/info/ae/ae/cmj_jmsp.html
- **Using WMQ with WASv7**
  - http://www.ibm.com/developerworks/websphere/techjournal/0901_leming/0901_leming.html
- **Scaling WPS**
  - http://www.ibm.com/developerworks/websphere/library/techarticles/0901_herness/0901_herness.html

© 2009 IBM Corporation