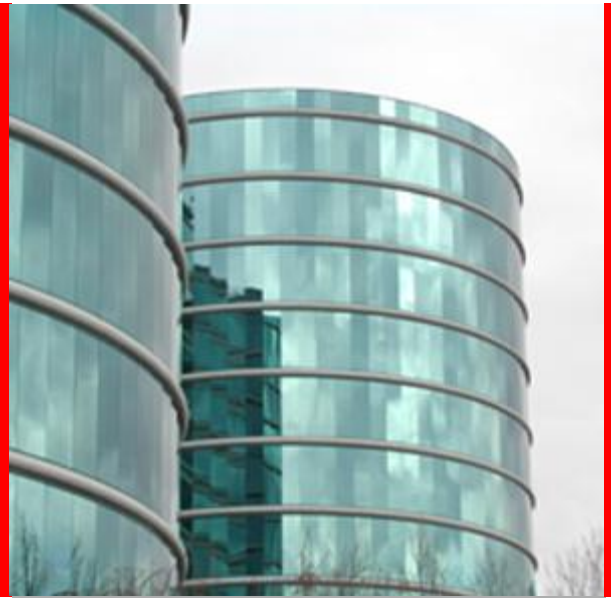


WebLogic JMS Messaging Infrastructure
WebLogic Server 11gR1 Labs

Messaging Basics

Built-in Best-of-Breed Messaging (JMS) Engine

Years of hardening. Strong performance.

- WLS embeds within it a full-function high-performance Messaging Engine that is on-par or superior to Messaging "pure-plays"

- Using WLS JMS eliminates the need to acquire and manage a 3rd party Messaging product,
  - reducing Infrastructure Costs
  - Reducing licensing costs
  - taking advantage of superior capabilities offered by this engine

Basics – Advantages of “Built-in”

- No additional installation: runs in same process space as WLS
- Integrated infrastructure
  - Leverages core WLS protocols and services (RMI, thread pooling)
  - WLS supplies Web Services, Servlets, and EJBs which work in concert with JMS
- Integrated security
  - Uses same user identities
  - Leverages WLS role-based security model
- Integrated administration
  - Unified administration console
  - Unified configuration
- Robust, proven built-in transaction manager

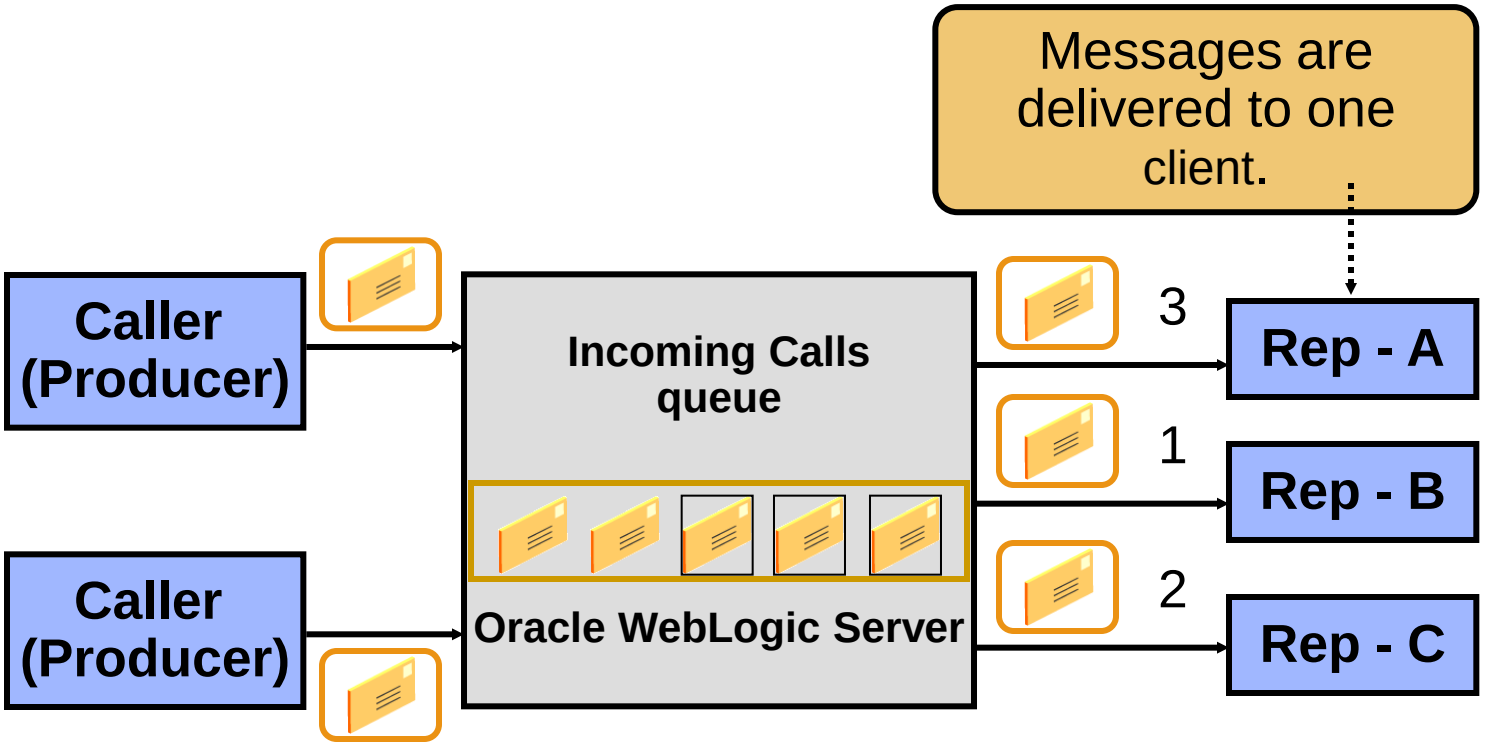
- Optimal performance and scalability
  - Applications can access JMS locally without a network call
  - No need to serialize/de-serialize messages
  - Connection pooling when used inside EJBs and Servlets

Basics – Standards and QoS

- Standards and protocol support
  - Fully JMS 1.0.2 and 1.1 compliant (pub/sub and queuing)
  - Fast, multicast-capable pub/sub
  - File or Database persistence (both fully XA-capable)
  - Enhanced XML message support
- Reliability and QoS
  - Error destinations and retry counts to handle failed messages
  - Message paging to support large sets of messages
  - Timer services to reliably schedule future message delivery

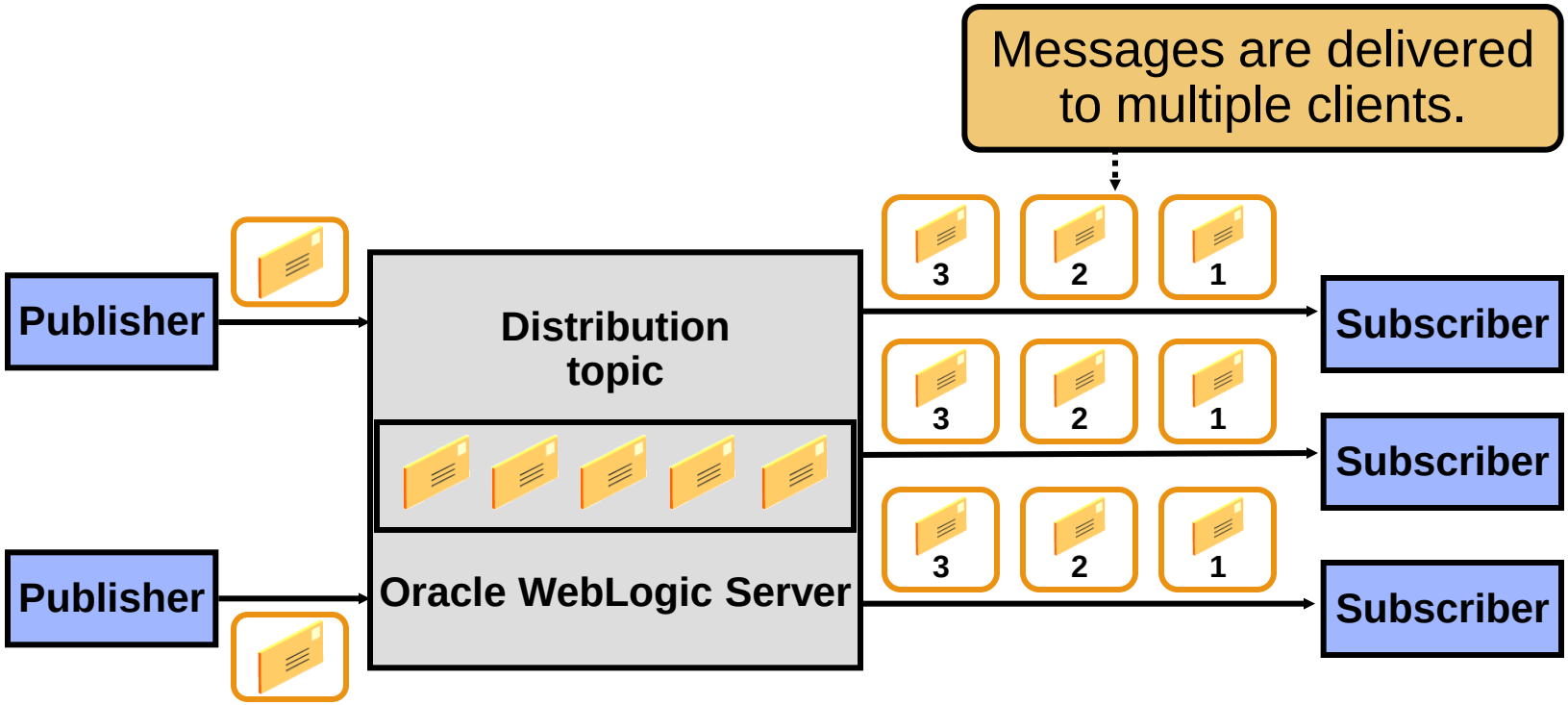
Point-To-Point Queue

- Many message producers can serialize messages to multiple receivers in a queue.



Publish/Subscribe Topics

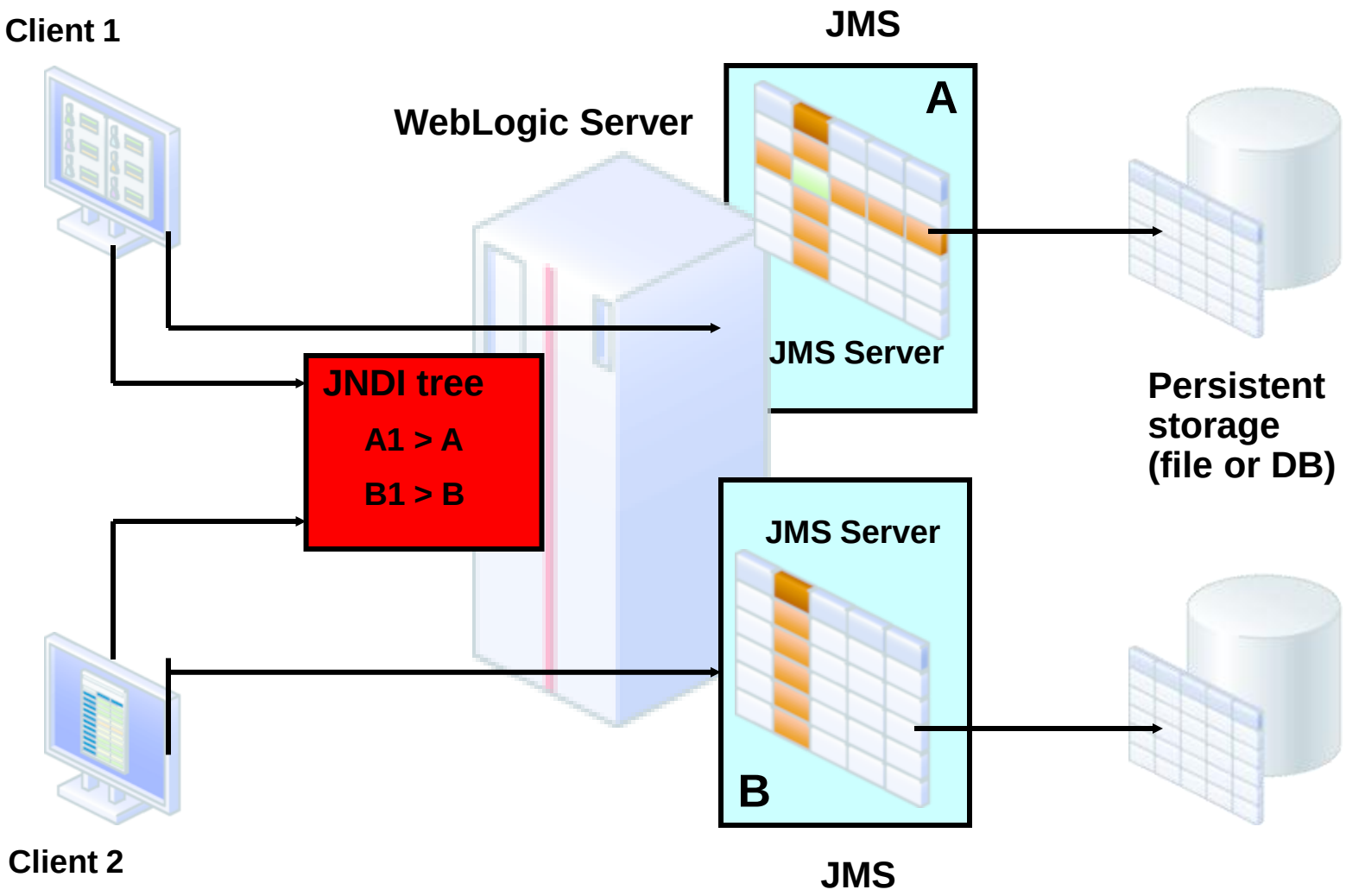
- Publishing and subscribing to a topic decouples producers from consumers.



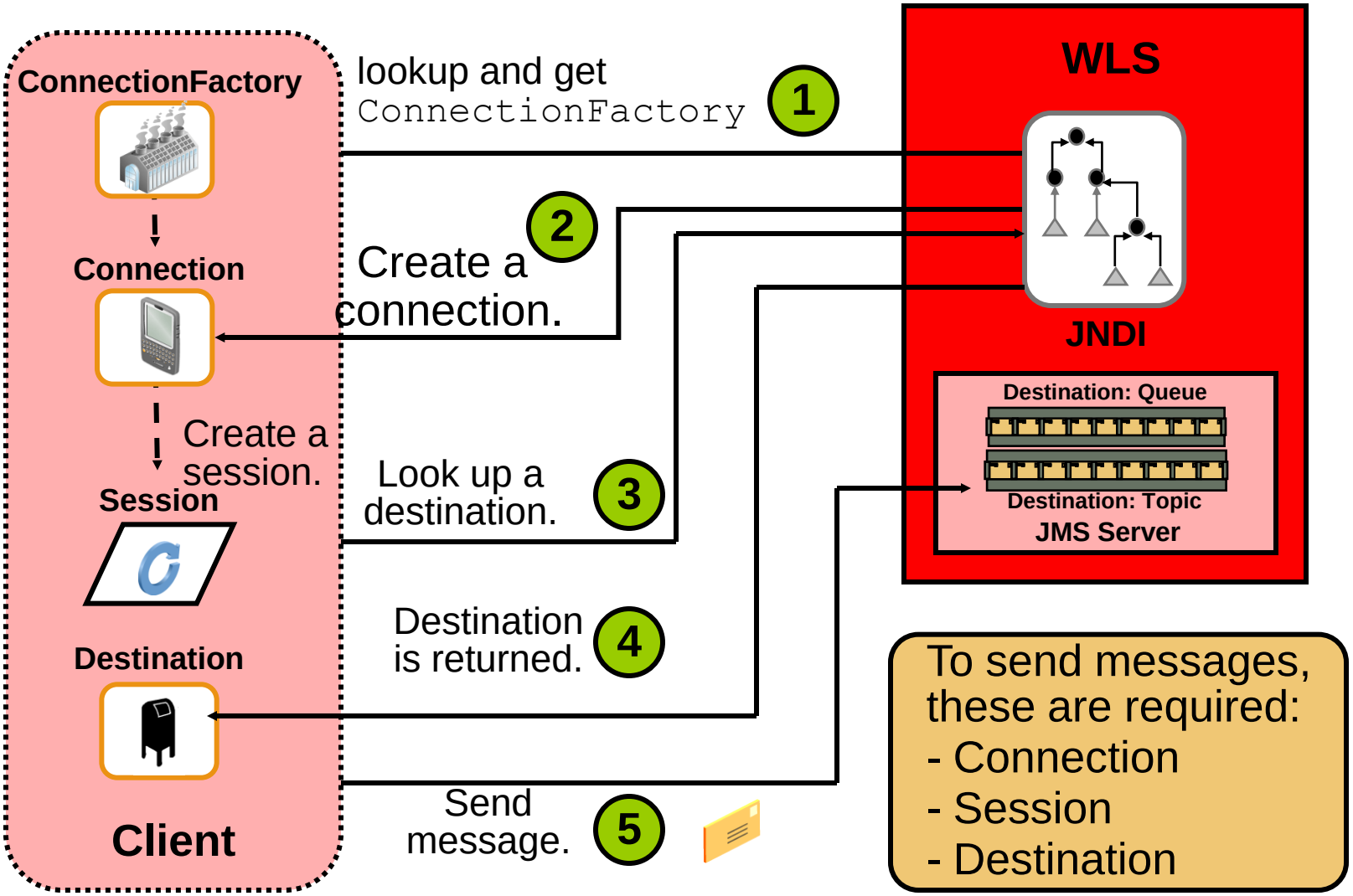
Oracle WebLogic Server JMS Features

- Oracle WebLogic Server JMS supports:
  - Both the point-to-point and Publish/Subscribe JMS models
  - Acknowledgement-based guaranteed delivery
  - Transactional message delivery
  - Durable subscribers
  - Distributed destinations
  - Recovery from failed servers

Oracle WLS JMS Architecture

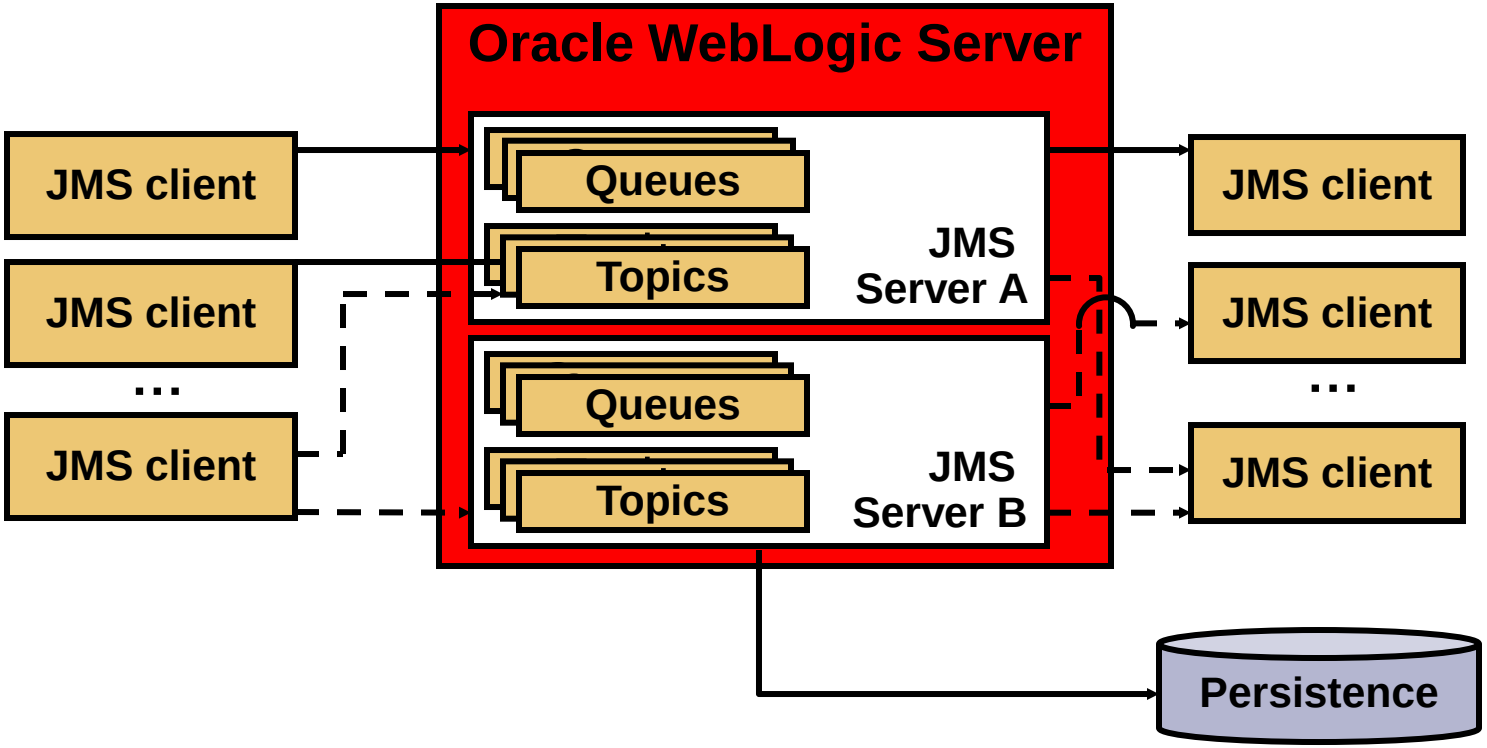


Typical JMS Messaging Process



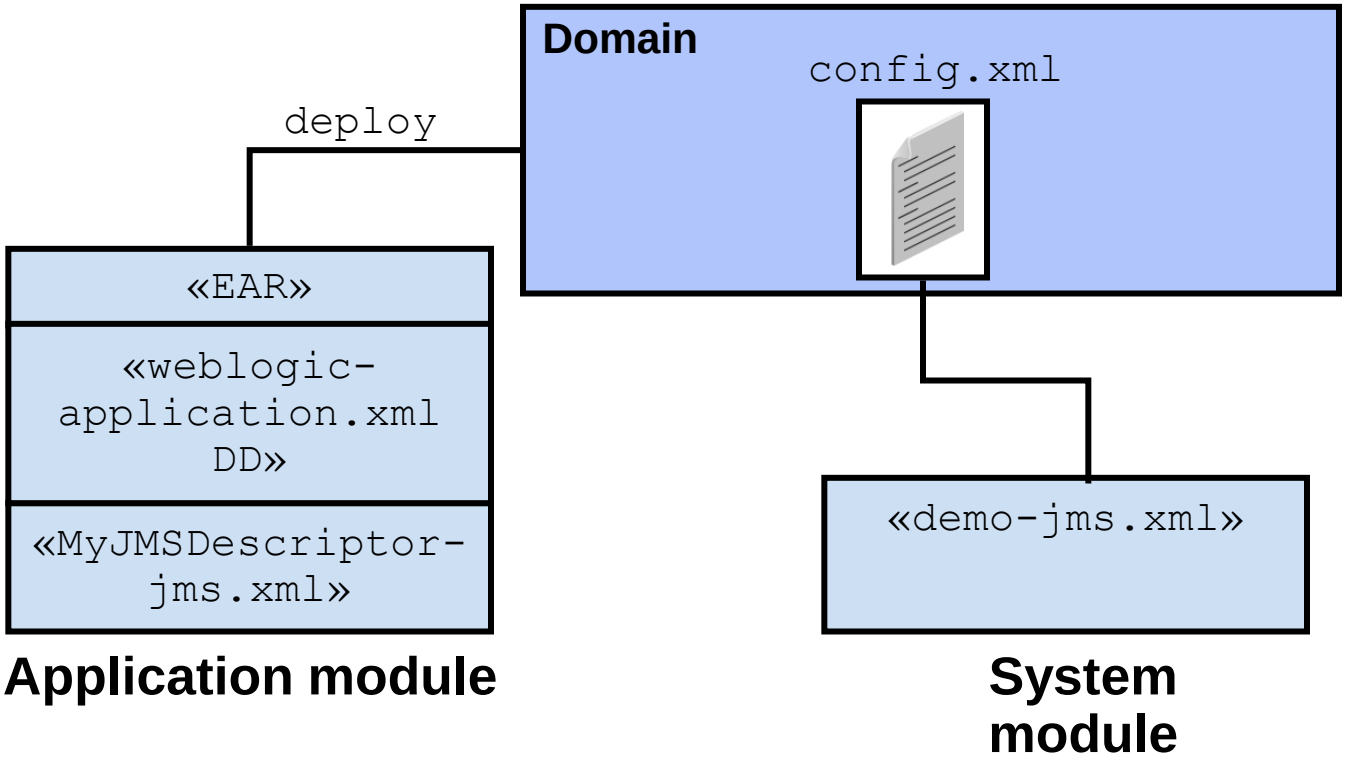
Oracle WLS JMS Server

- In Oracle WLS, the messaging service is implemented through a JMS server.
- A JMS server receives and distributes messages.



JMS Modules

- JMS resources can be configured as:
  - System modules
  - application modules.



Connection Factories

- JMS connection factories are used to set default client connection parameters, including:
  - Message priority
  - Message time-to-live (TTL)
  - Message persistence
  - Transactional behavior
  - Acknowledgement policy
  - Flow control
- WLS provides a default client connection factory that:
  - Uses WebLogic's default connection settings
  - Is located on the server JNDI tree at `weblogic.jms.ConnectionFactory`

Destination

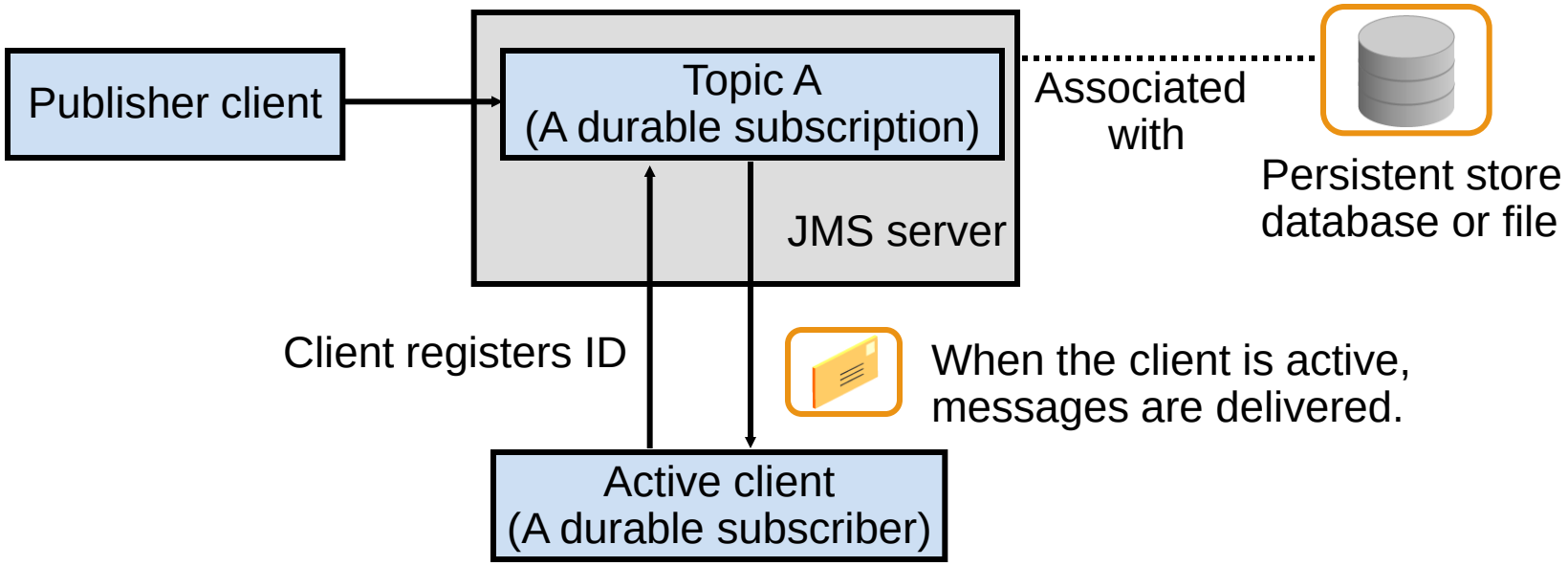
- A destination is a lightweight object that is stored in JNDI.
- It is the target on a JMS server for sending messages and the location from where messages will be consumed.
- The JMS destination types are:
  - Queue (for the point-to-point model)
  - Topic ((for the Publish/Subscribe model)

Durable Subscribers and Subscriptions

- Durable subscribers register durable subscriptions for guaranteed message delivery even if the subscribers are inactive.
- A subscriber is considered active if the Java object that represents it exists.
- By default, subscribers are nondurable.

How a Durable Subscription Works

- Durable subscription is effective only when the client is inactive during the time that the message is published.
- When the client becomes active again, its ID is used to retrieve and redeliver messages.



Configuring a Durable Subscription

- To configure durable subscriptions, an administrator must:
  - Create and configure a JMS store
  - Configure connection factories or destinations as persistent
  - Associate the JMS store with the JMS server
- The JMS store can be configured to use either of the following:
  - A file store
  - A JDBC Store (a connection pool)

Messaging Manageability

Message Management

- Comprehensive monitoring statistics
  - Stats for clients, destinations, JMS servers, pooling, transactions, …
- Message management
  - View / browse *all* messages including invisible messages (birth-time, transactional, retry delay)
  - Delete, move, import, export messages (to XML files)
- Pause/resume destinations
  - Prevent sends and/or consumes
- Fully dynamic
  - Rarely requires manual XML editing or restarts
- Available via console, JMX, WLST (scripting)
- Reminder:
  - Integrated infrastructure with WLS
  - Integrated security with WLS
  - Integrated administration with WLS

Monitoring JMS Servers

- Statistics are provided for the following JMS objects:
  - JMS servers
  - Connections
  - Destinations

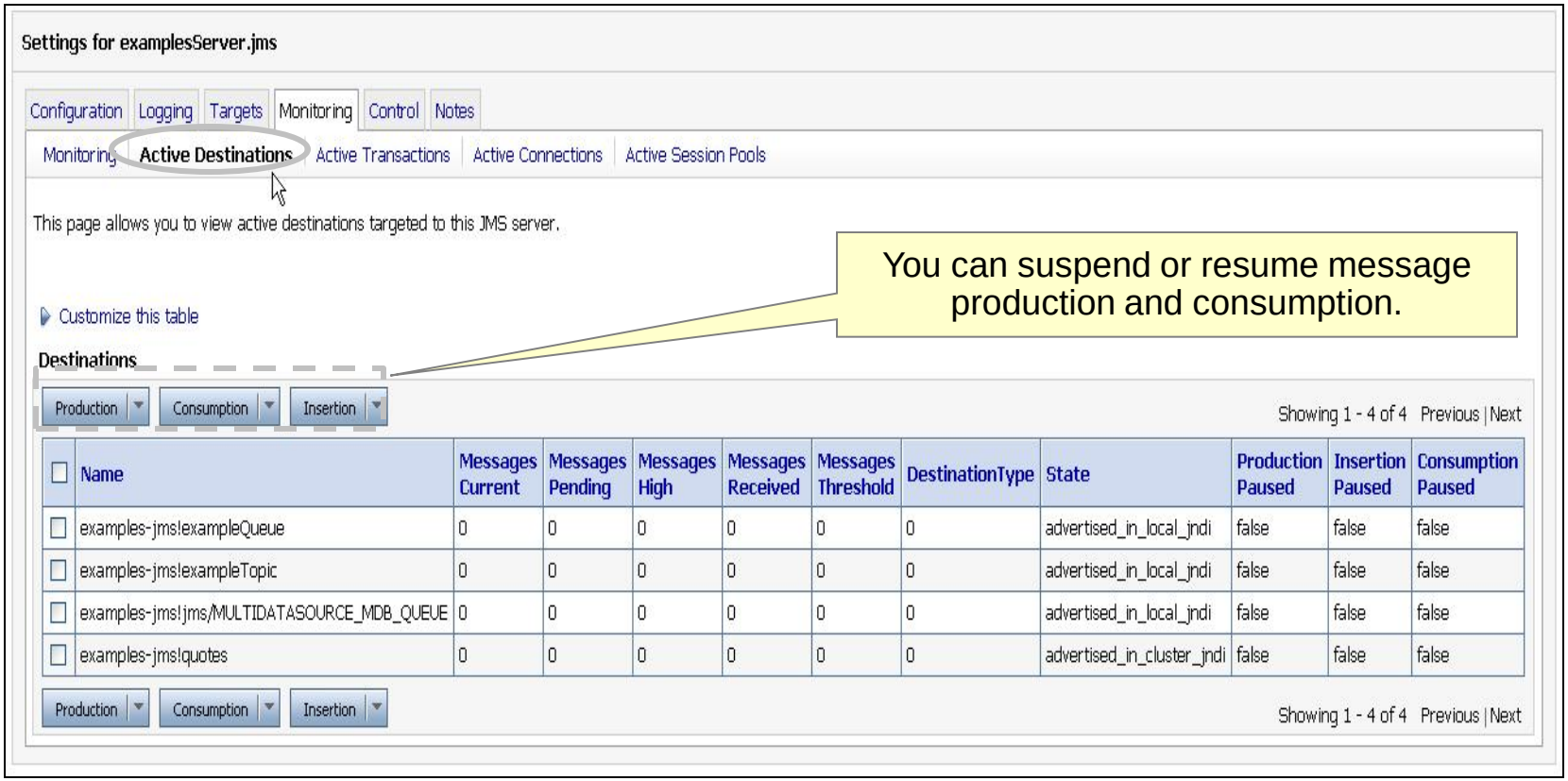
Customize this table

Statistics(Filtered - More Columns Exist)

Showing 1 to 1 of 1 Previous | Next

Name	Messages Current	Messages High	Bytes Current	Session Pools Current
PayrollJMSServer	3	3	36	0

Monitoring and Managing Destinations



You can suspend or resume message production and consumption.

Monitoring Queues

- In the Administration console, navigate to Services > Messaging > JMS Modules.
- In the JMS Modules table, click the JMS module you have created.
- In the Summary of Resources table, click the link to your queue, and then click the Monitoring tab.
- The Messages High and Messages Total columns show nonzero values indicating that messages have been received.

Destinations(Filtered - More Columns Exist)

Show Messages

Showing 1 to 1 of 1 Previous | Next

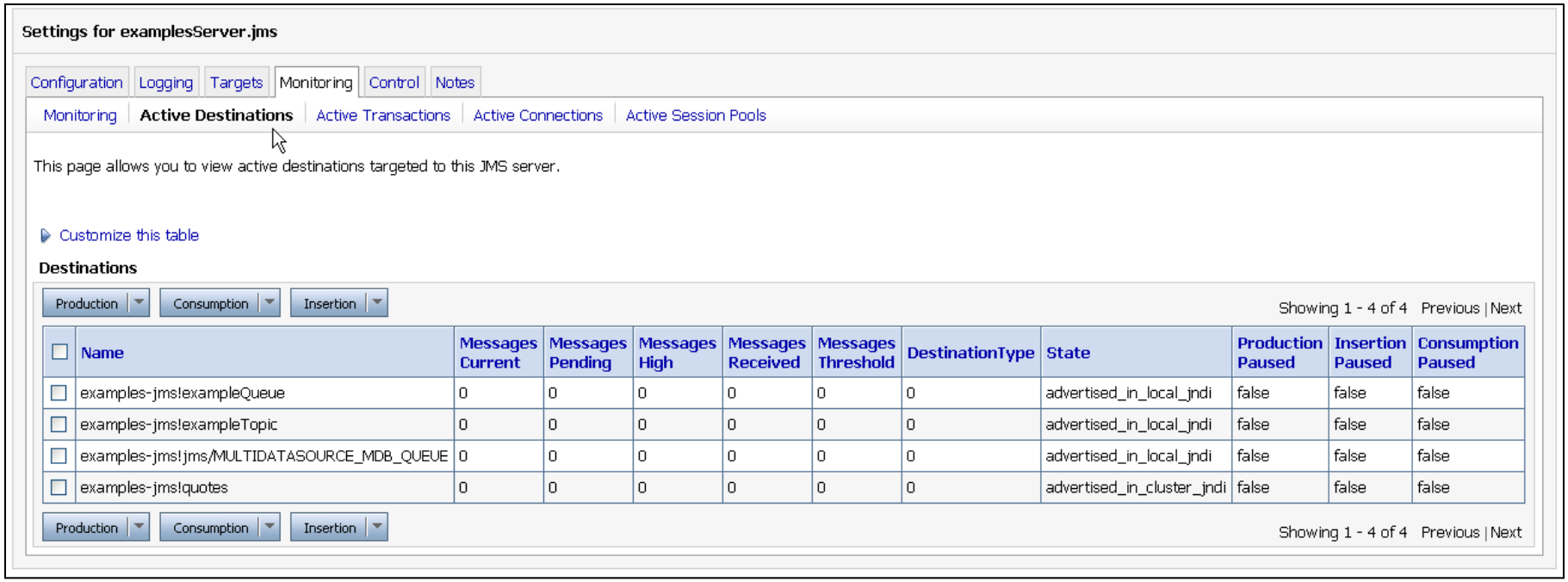
Name	Consumers Current	Consumers High	Consumers Total	Messages High	Messages Total
dizzyworldModule!dizzyworldQueue	0	0	0	1	1

Show Messages

Showing 1 to 1 of 1 Previous | Next

Viewing Active Queues and Topics

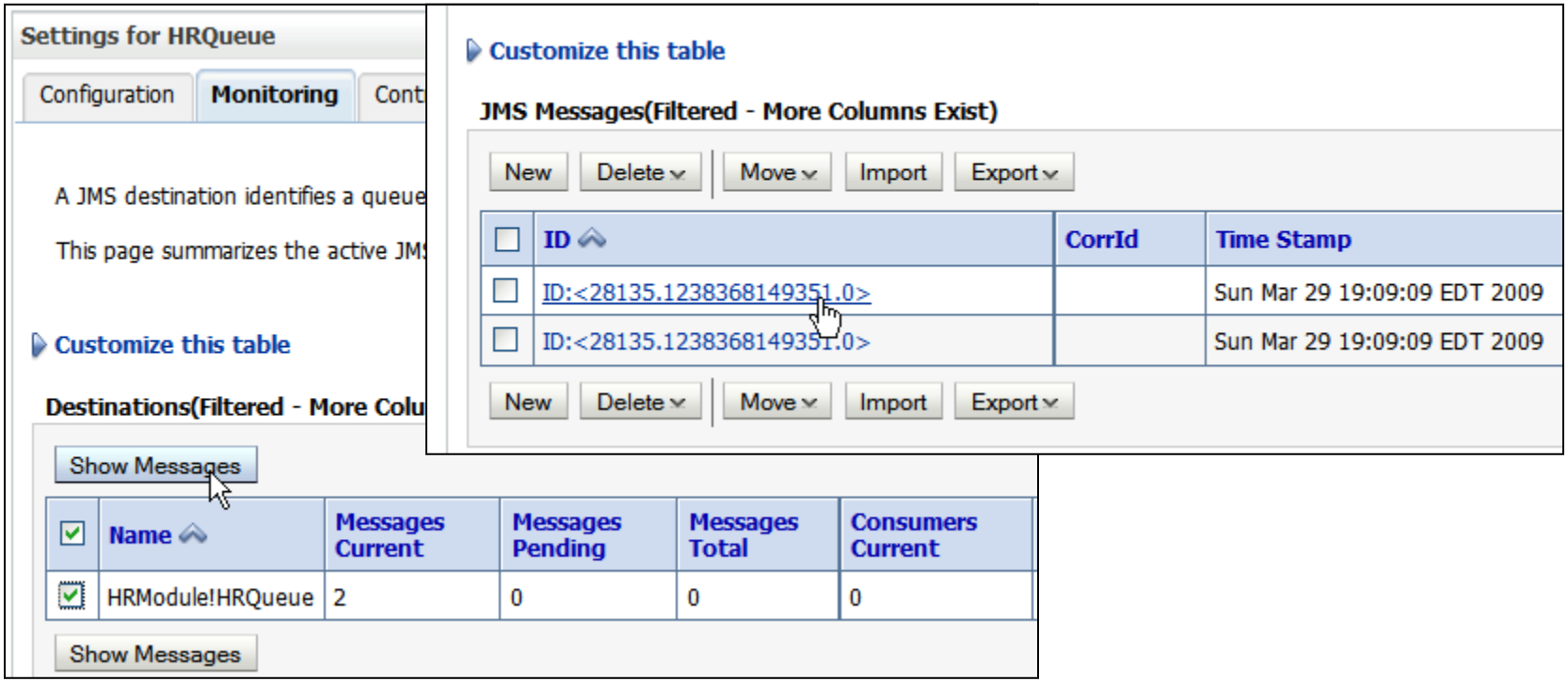
- In the Administration Console, navigate to the JMS Modules and click the Active Destinations tab.



Name	Messages Current	Messages Pending	Messages High	Messages Received	Messages Threshold	DestinationType	State	Production Paused	Insertion Paused	Consumption Paused
examples-jms!exampleQueue	0	0	0	0	0	0	advertised_in_local_jndi	false	false	false
examples-jms!exampleTopic	0	0	0	0	0	0	advertised_in_local_jndi	false	false	false
examples-jms!jms/MULTIDATASOURCE_MDB_QUEUE	0	0	0	0	0	0	advertised_in_local_jndi	false	false	false
examples-jms!quotes	0	0	0	0	0	0	advertised_in_cluster_jndi	false	false	false

Managing Messages in a Queue

- You can enable messages to be viewed in the Administration Console.
- After they are enabled, you can view and manage the messages in a queue using the Administration Console.



Messaging High Availability

What is High Availability for JMS?

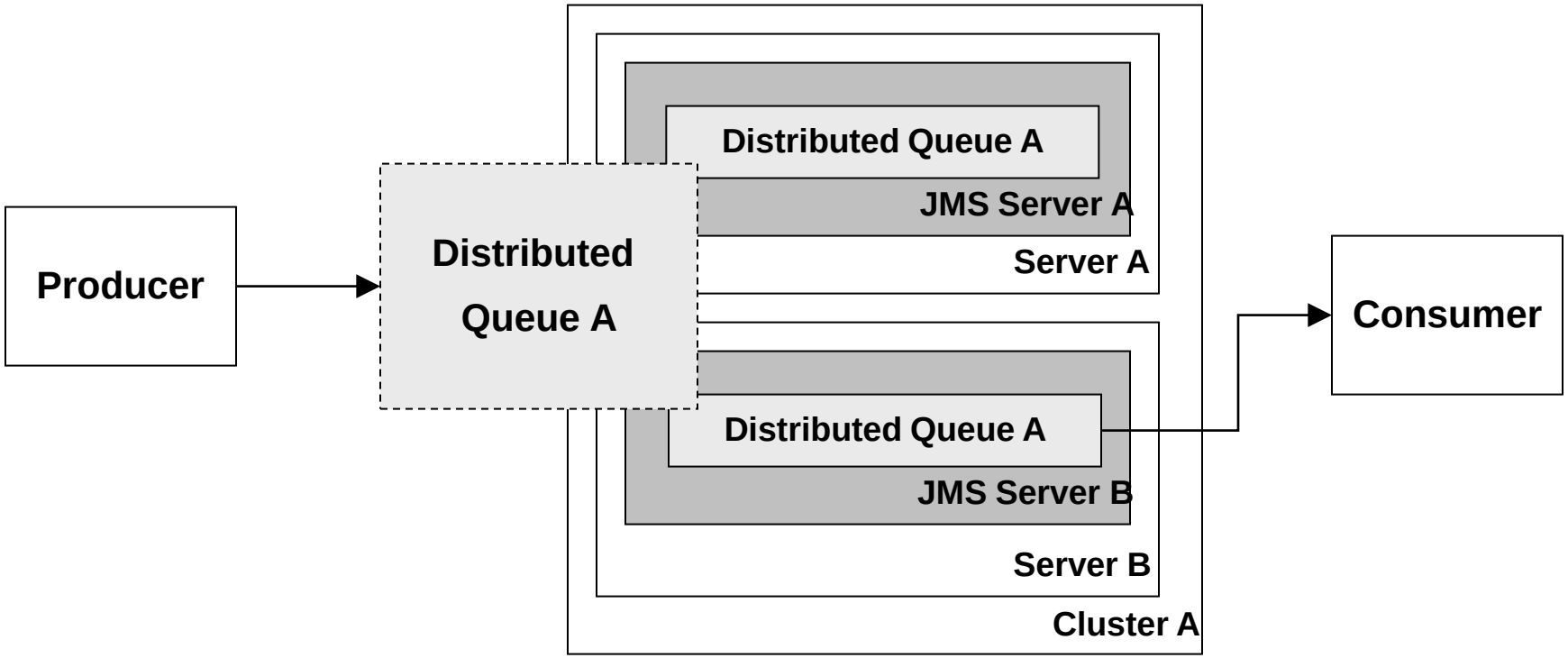
Continued ability to send and receive messages	Distributed Destinations
All messages sent are processed	Whole Server and Service Migration
Seamless client failover	Automatic Reconnect
Continued ability to send when no remote servers are available	Store and Forward Client SAF

Distributed Destinations

aka "Clustered Destinations"

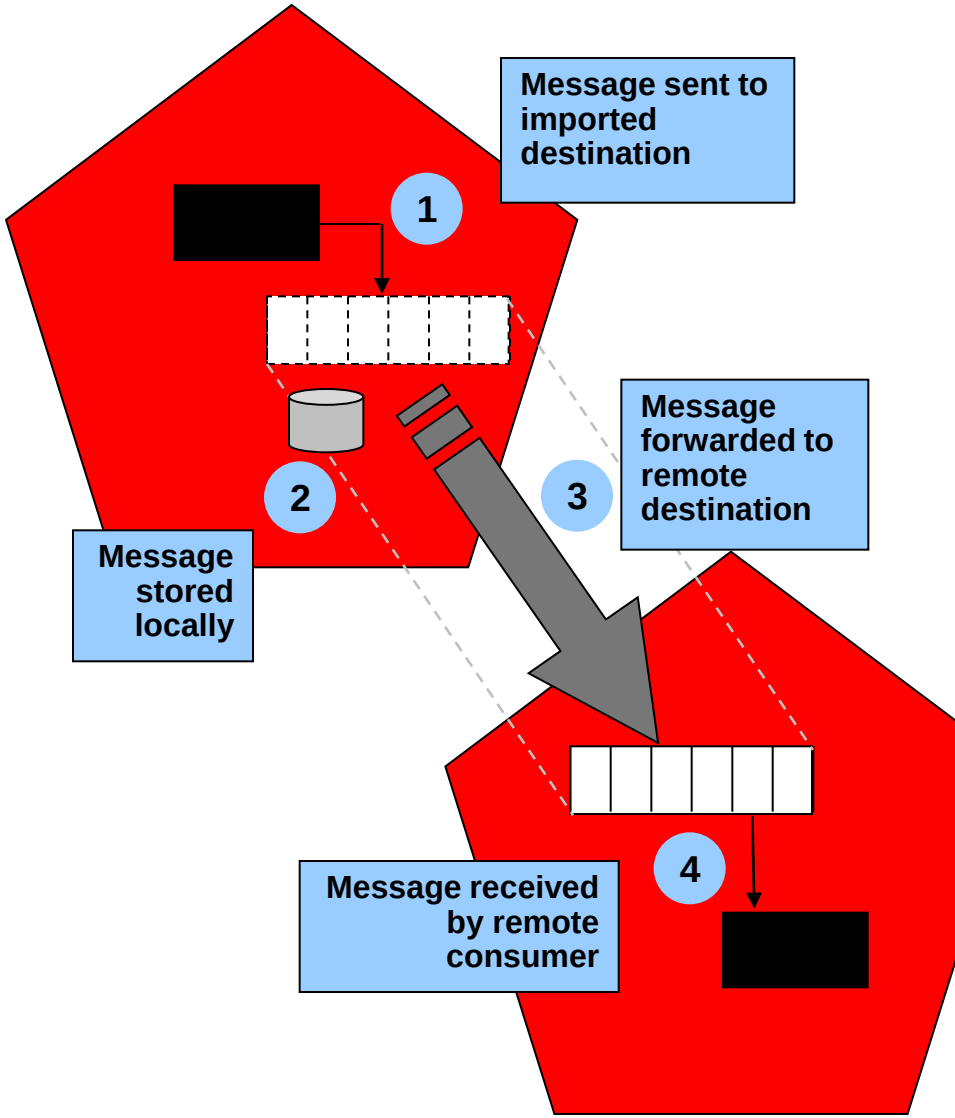
- Umbrella for a group of "member" queues or topics in a cluster
- Appear as a single unit
- Provides high availability and scalability
  - Multiple physical instances
  - Parallel processing
  - Scalability
  - HA
  - Load balancing and failover, with fine-grained control

Distributed Destinations

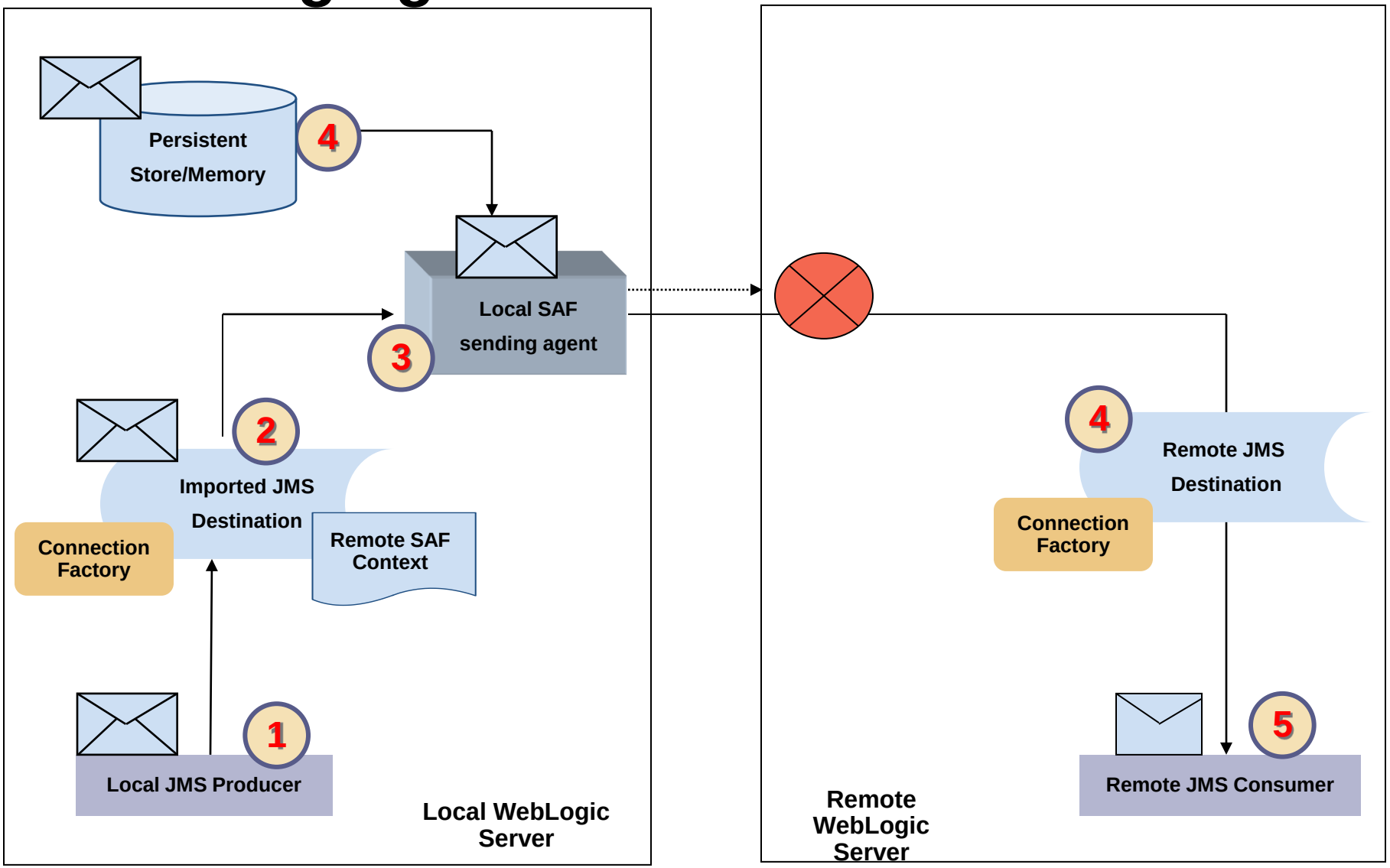


Store and Forward (SAF)

- Store messages on local server and forwards to remote server when endpoint is available
- Increases reliability of communication
  - Forwarding between domains, clusters, and servers
  - Preserves message ordering
- Improvement over Messaging Bridge
  - SAF is faster and more scalable for WLS-WLS connectivity
  - Clusterable
  - Messaging Bridge still supported and useful for non-WLS connectivity or pre-WLS 9.0 destinations



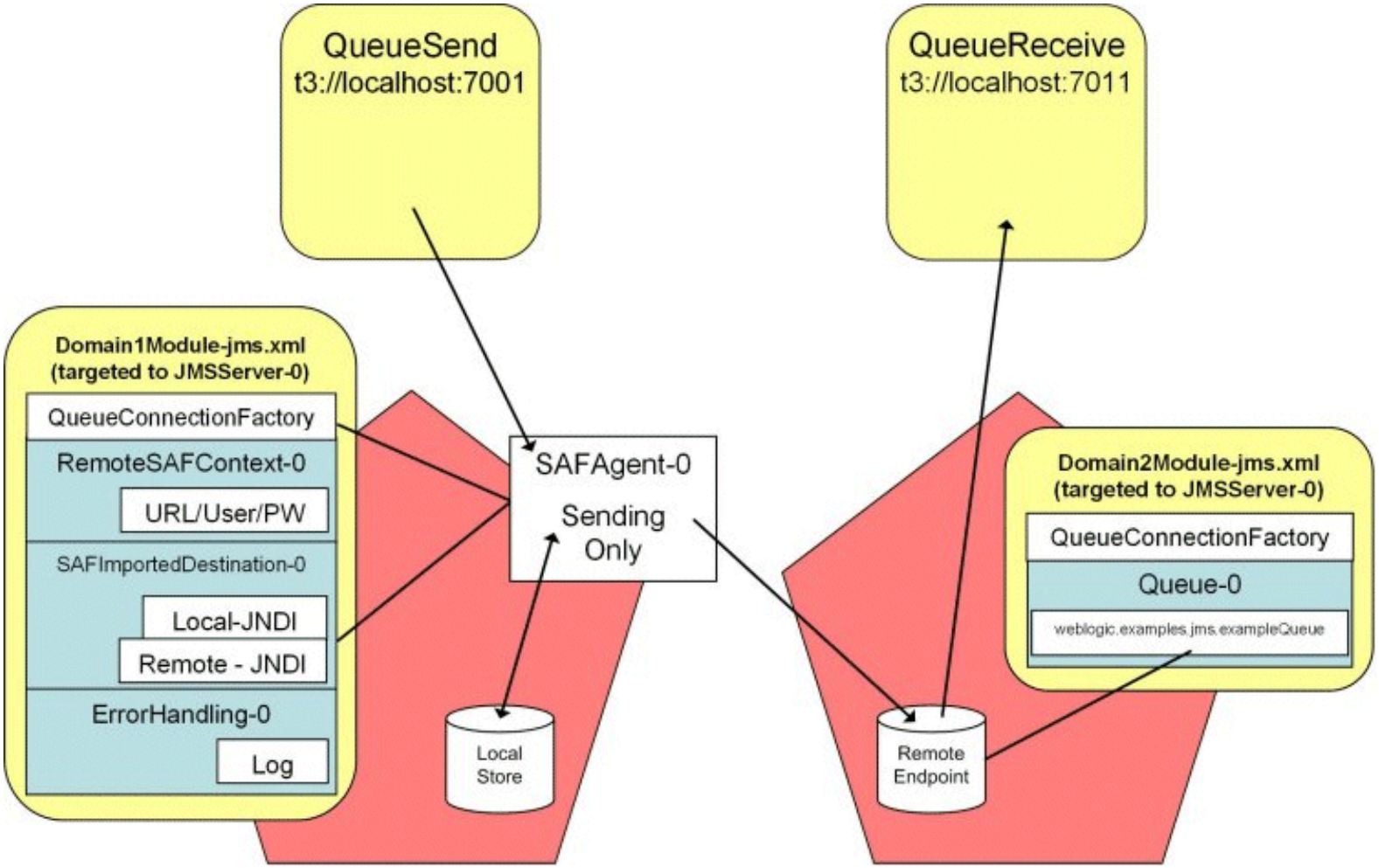
Messaging flow in SAF



SAF Resources In a JMS Module

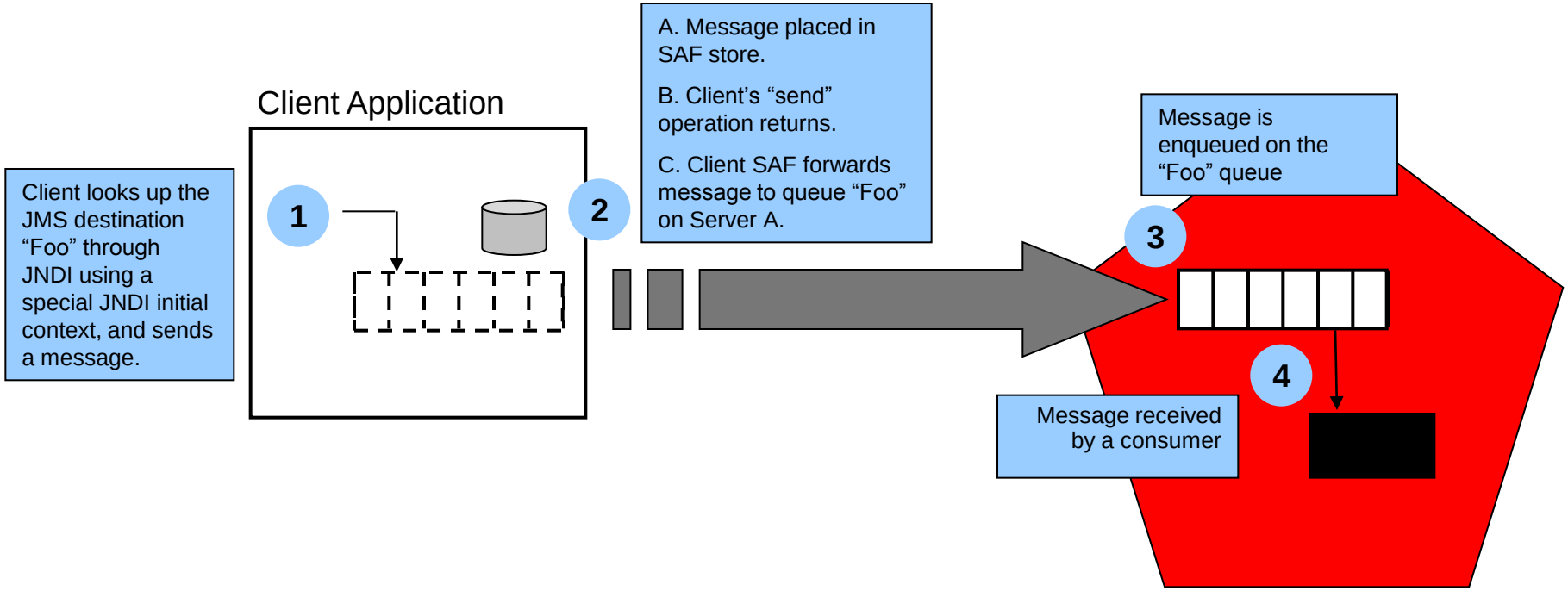
- When configuring SAF resources for a JMS module, you need to configure the following resources in a JMS system module or application module:
  - **Imported SAF Destinations** - local representation of a JMS destinations (queues or topics) in a JMS module that is associated with a remote server instance or cluster
  - **Remote SAF Context** - URL of the remote server instance or cluster where the JMS destination is exported from
  - **SAF Error Handling** - define the action to be taken when the SAF service fails to forward messages to a remote destination

SAF JMS Picture



Client Store And Forward

- Same concept as Store and Forward, except the messages are stored on the client and are forwarded when the client is connected.
- Enables reliable messaging over unreliable network links.
- Small footprint on client side.



Enterprise Features

Interoperability: Mapping, Wrapping, & Bridges

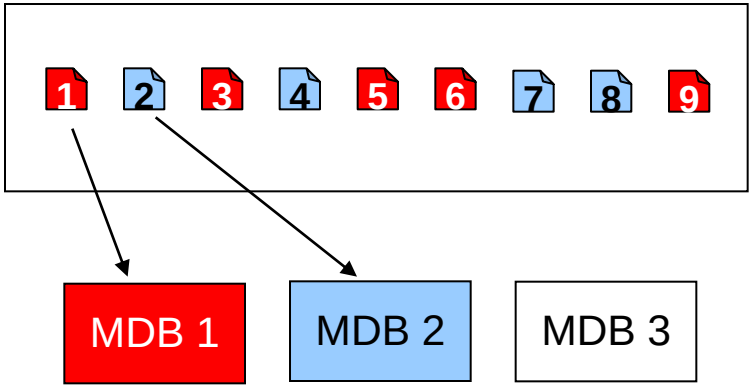
- Foreign JMS Servers and Destinations (Mapping)
  - Optional configured mapping of remote JNDI resources to local JNDI
  - Avoids hard-coding in app or descriptors
- Standard EJB or servlet *resource references* (Wrapping)
  - ***Automatically pool*** referenced JMS resources when they are closed
  - ***Automatically enlists*** JMS resources with the current transaction
- MDBs can directly consume from any JMS vendor
- Messaging Bridge
  - Forwards from a “source” destination to a “target” destination.
- Store and Forward

JMS Unit of Order

- **Problem Description**
  - Certain applications require strictly ordered processing of messages
  - Typically implemented by serializing processing of ALL messages (kills performance) or adding application complexity (detect or prevent out of order processing)
- **Solution: Unit of Order**
- **How this feature works**
  - Messages tagged with same Unit of Order (UOO) are "**processed**" in order
  - *Applies across a cluster, too: messages routed to DD member*
  - Concurrency through Multiple UOOs
  - Stronger ordering semantics than the JMS specification
- **Benefits**
  - IT can support complex Business workflow requirements without building major and costly complexities into the apps or compromising performance
  - Reduces or eliminates DB lock contention

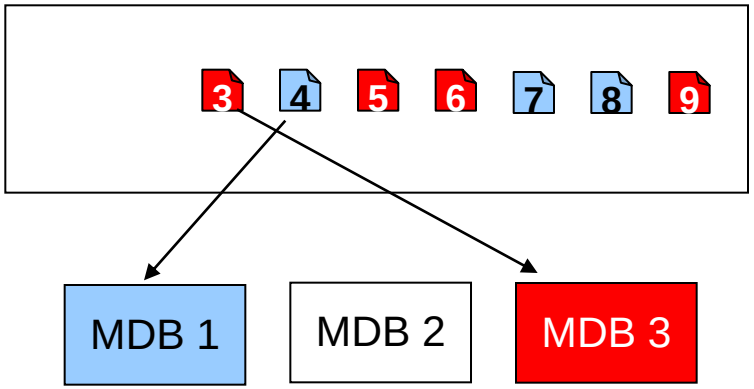
State 1

- Msg 1 consumed by MDB 1
- Other UOO Red messages unavailable
- Msg 2 consumed by MDB 2
- Other UOO Blue messages unavailable



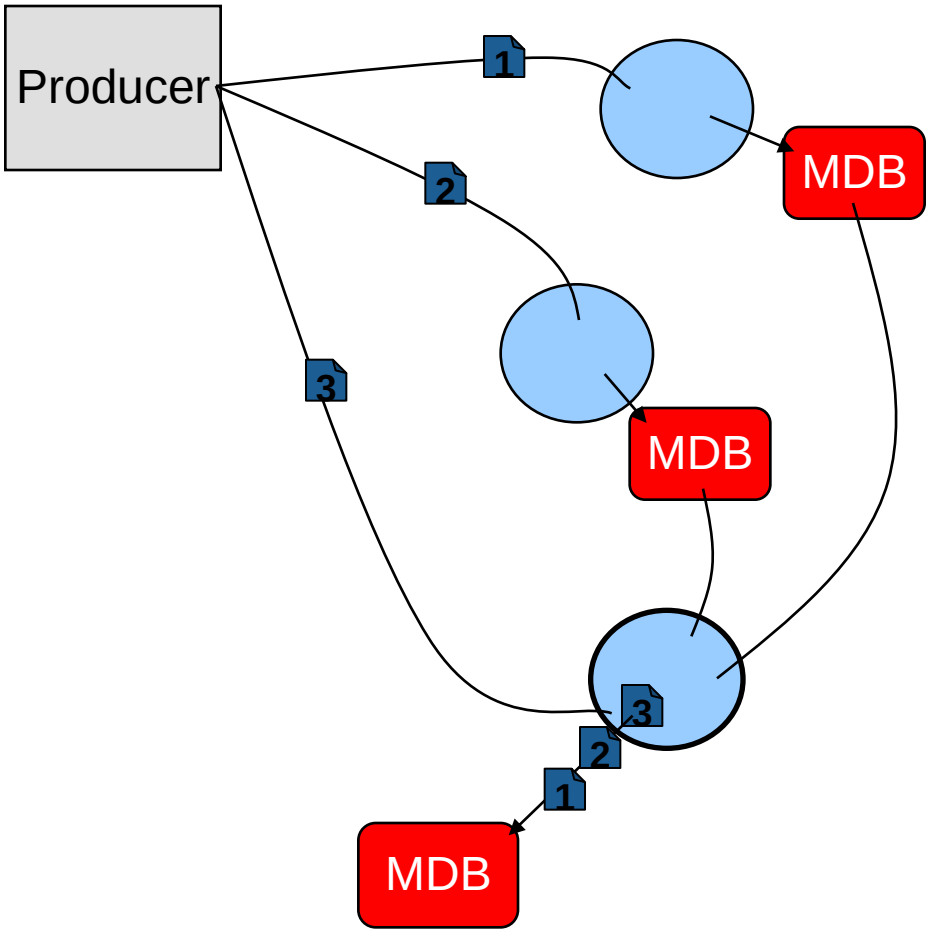
State 2

- MDB 1 completes processing message 1
- Next UOO Red message becomes available
- Msg 3 consumed by MDB 3
- Other UOO Red messages unavailable
- …



Unit of Work

- Messages are grouped with a group identifier
- At the final destination, messages become available only when all messages in the group have arrived
- Messages are reordered as specified by the UOW producer, regardless of the order in which they arrived
- Messages are received by a single consumer with no gaps between messages in the group
- “Intermediate Destinations” – stops along the way for some messages; UOW is ignored at those destinations



New in 10.3: WebLogic Messaging .Net Client

- Brings together two worlds: .Net and Java
  - Alternative to existing C client
- Allows front end to be coded with .Net and back end coded with WebLogic
- Fully managed code
  - Single DLL, no JNI
- Based on JMS 1.1 API
  - Many WebLogic extensions supported
  - SAF, DD, Automatic Failover – all of our Messaging Engine benefits
- Direct access to WebLogic JMS
  - Uses existing socket configuration: t3://WebLogicServer:port
  - **No third party bridging**

Other WebLogic JMS Features

- **Deployable Configuration:** Optionally put configuration (destinations, etc) in XML descriptor and deploy with application.
- **Timed messages**: Send a message that is not delivered to consumers until a specified time
- **Automatic Client Reconnect**: Best effort to transparently reconnect clients to cluster after a network outage (configurable).
- **Sorted queues**: Sort the messages on a queue based on message header fields and/or user-defined properties; FIFO is the default
- **Username in message**: Optional.
- **Message lifecycle logging**: Text logging of fine grained events in message life-cycle
- **Multicast topics**: Delivers messages to topic consumers using a multicast protocol (fast but unreliable)
- **XML messages**: Store XML as Dom tree instead of text, filter topic subscribers and queue messages using an XPath expression
- **Logging Last Resource**: Transactionally safe (ACID) 2PC tx optimization for non-XA DB connections

Oracle Advanced Queuing Integration

- Enables easy interop with existing AQ destinations
  - This was a significant challenge in previous releases!
- Leverages new JNDI provider in AQ JMS client
- Uses standard WLS integration features
  - Foreign JMS Servers
  - JDBC Data Sources
  - MDBs
- Fully supports JTA transactions