IBM DataPower SOA Appliances & MQ Interoperability

Joel Gauci - Certified IT Specialist ,

IBM DataPower & Connectivity Appliances

gauci@fr.ibm.com

MQ Guide Share
France

Agenda

- Part 1: DataPower SOA Appliances
  - Basic Concepts
- Part 2: DataPower & MQ Interoperability
  - Introduction
  - Basic MQ Architecture
  - Message Workflow
    - HTTP to MQ
    - MQ to HTTP
  - Routing
  - Using MQ with a Web-Service Proxy
  - MQ and JMS
  - Units of Work
    - DataPower UoW Implementation
- Conclusion

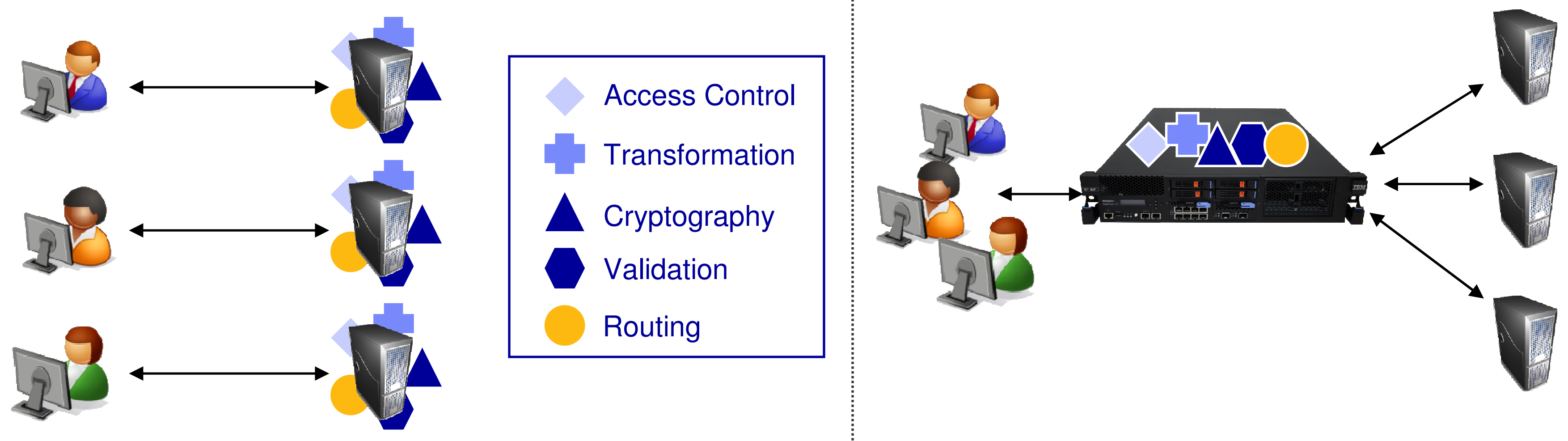
Agenda

SOA Appliances Centralize and Simplify Key Functions

- **Route, transform, and help secure multiple applications without code changes**
- **Lower cost and complexity**
- **Enable new business with unmatched performance**

Before SOA Appliance…

…After SOA Appliances

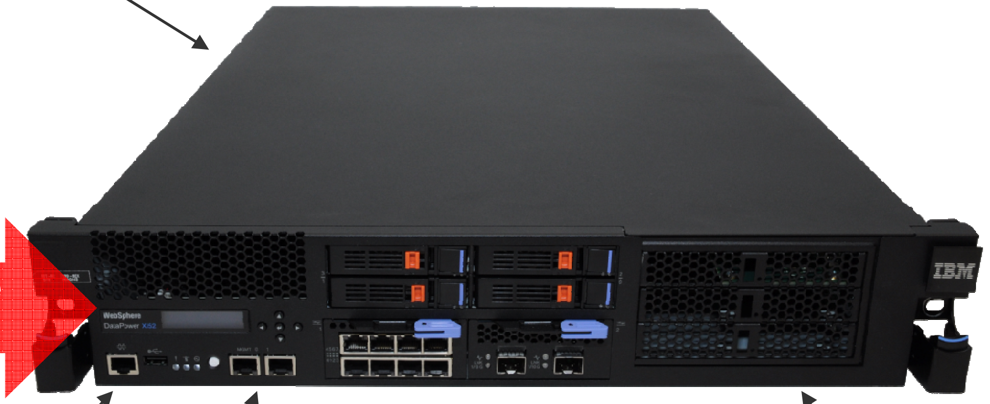


IBM WebSphere DataPower Appliances

XML Accelerator Hardware Component

Crypto Treatements Hardware Component

Replacable Units:
- HDD (RAID-1 / 600Go)
- Power Supply (2)
- Fans (2)
- Battery (2)
- Network Modules



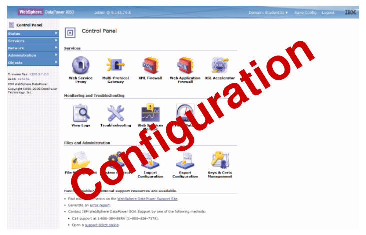
Firmware 6.0.0

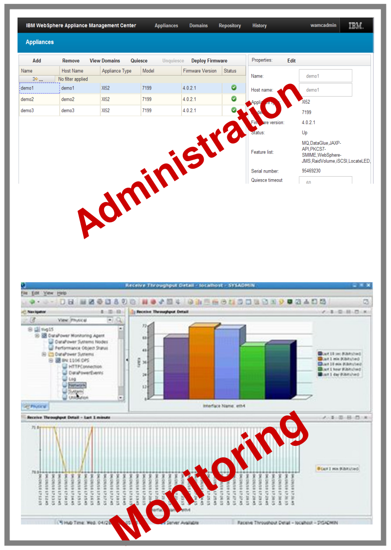
Slot for Hardware Security Module (HSM)

Management Ports

RJ45 Console Port

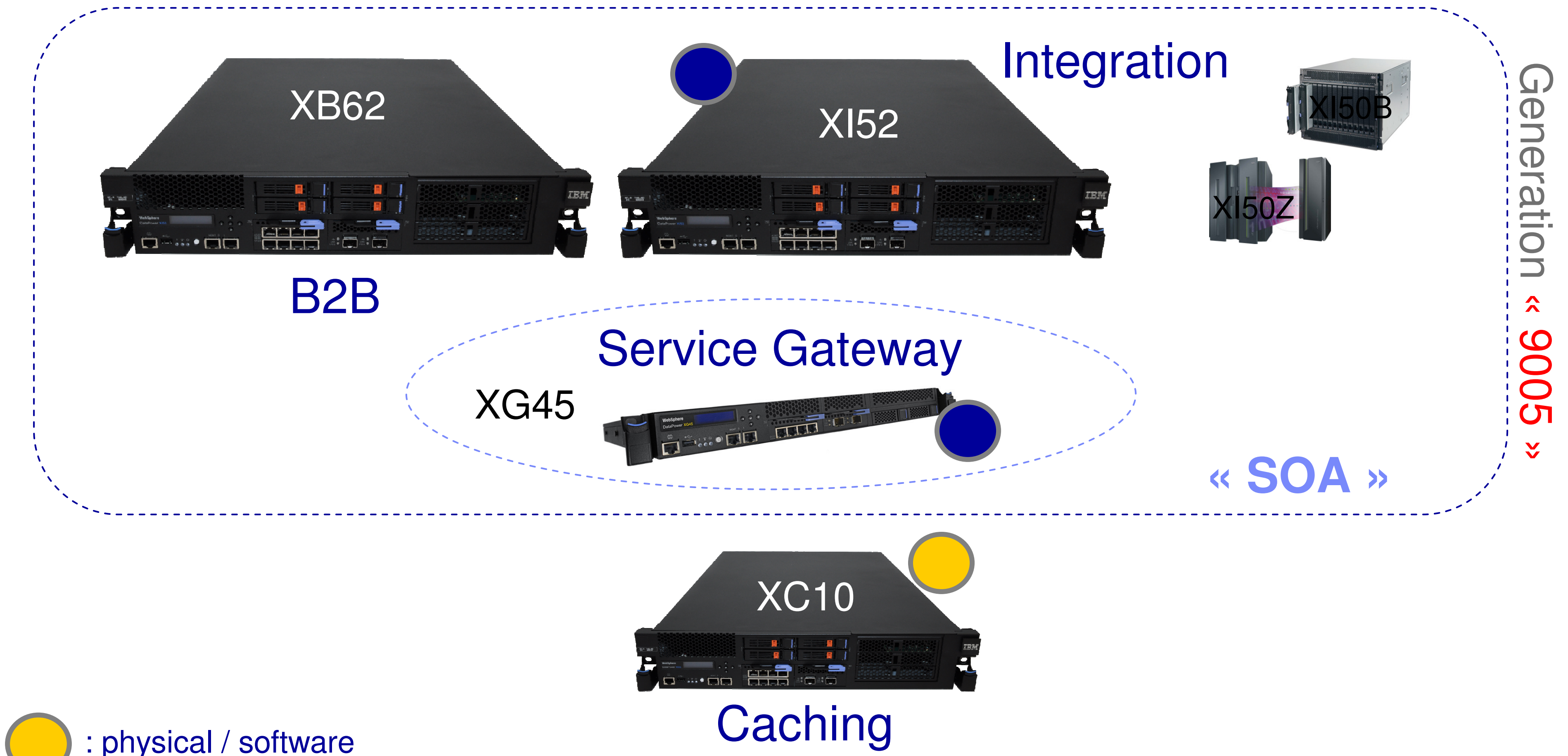
Network Modules





WAMC « 5.0 »

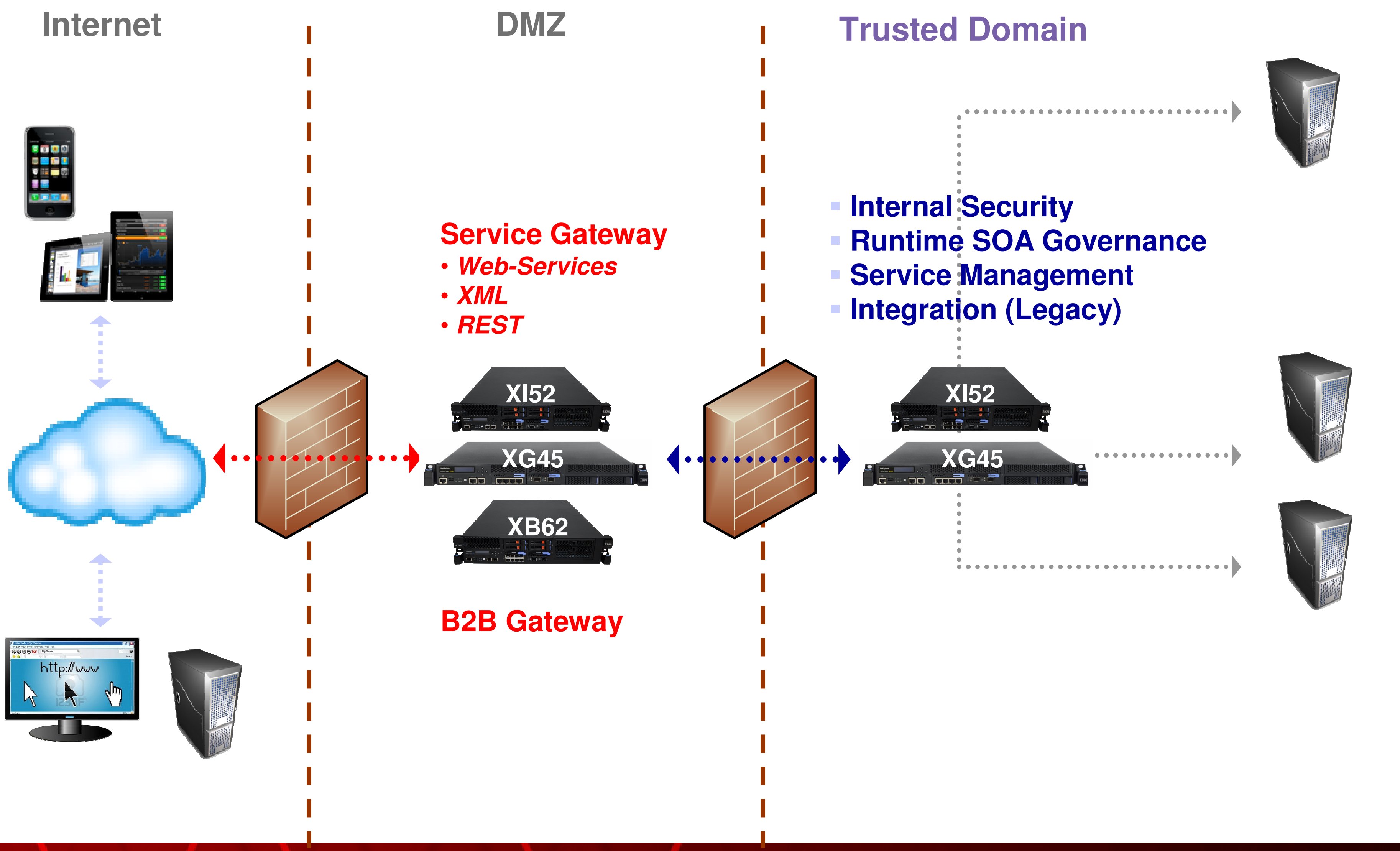
IBM WebSphere DataPower Portfolio



🟡 : physical / software

🔵 : physical / virtual

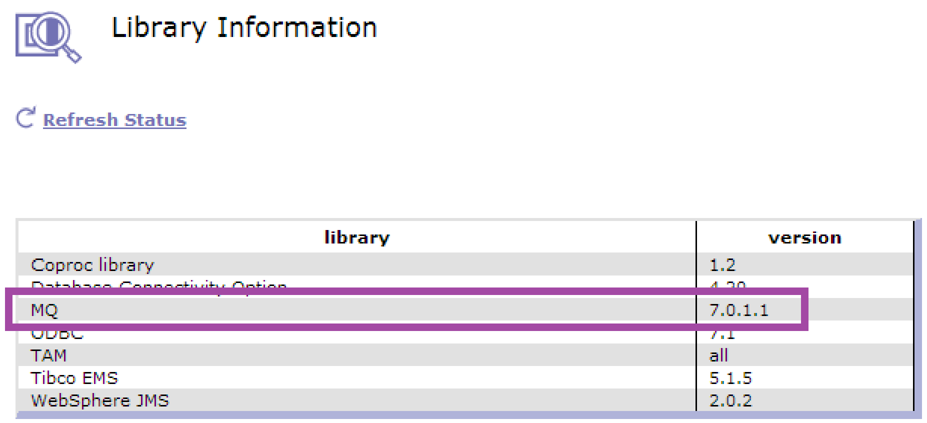
DataPower Appliance Topology



- Internal Security
- Runtime SOA Governance
- Service Management
- Integration (Legacy)

MQ Support in DataPower Appliances

- An MQ Client is automatically integrated in the following appliances
  - XI50 (/ XI50B / XI50Z)
  - XG45 (φ and ϑ)
  - XI52 (φ and ϑ)
  - XB62
- In firmware 6.0.0.1: MQ Client 7.0.1.1



Library Information

Refresh Status

library	version
Coproc library	1.2
Database Connectivity Option	4.20
MQ	7.0.1.1
ODBC	7.1
TAM	all
Tibco EMS	5.1.5
WebSphere JMS	2.0.2

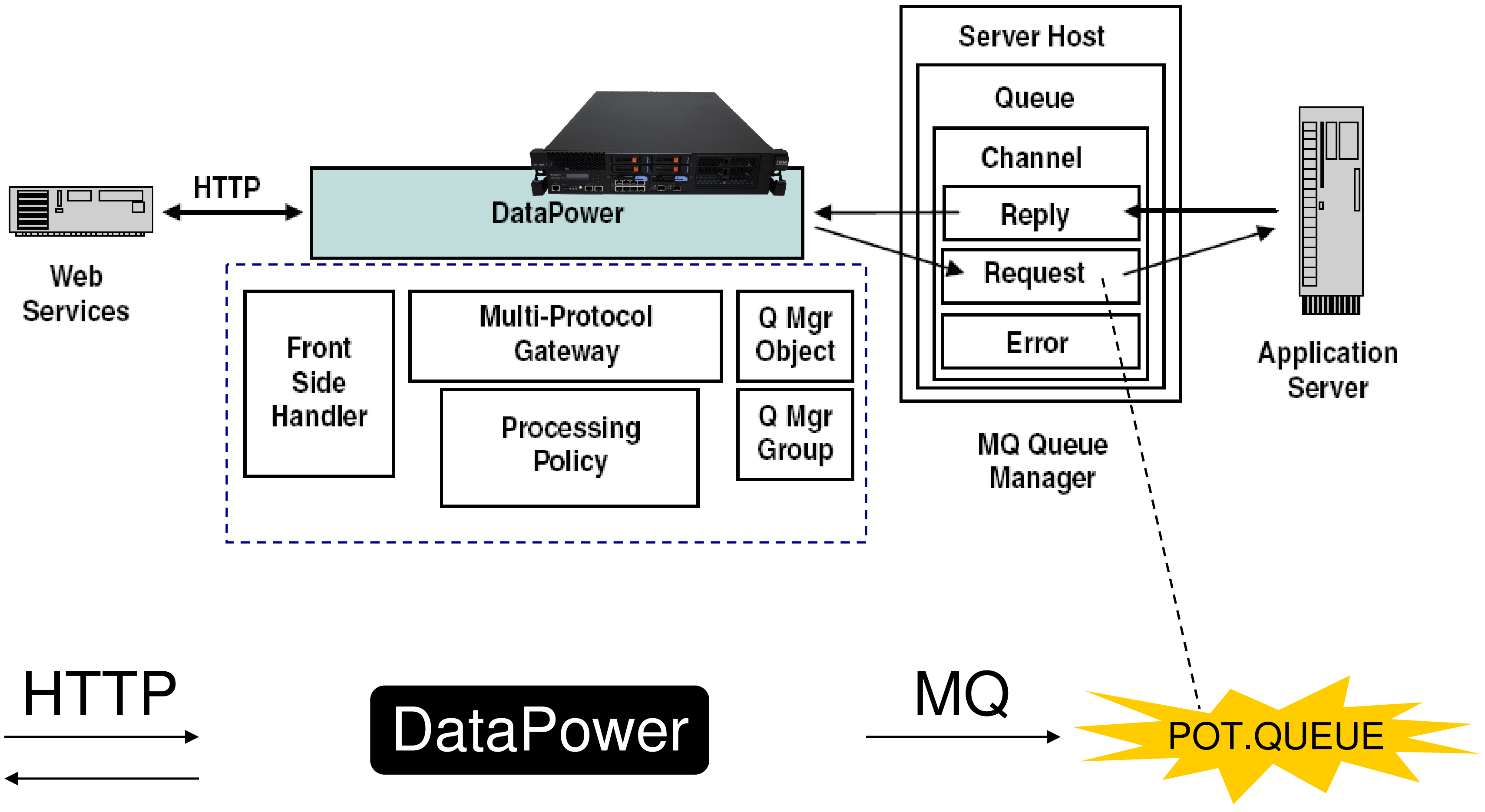
Agenda

Introduction

- DataPower(*) can exchange messages with WebSphere MQ systems by acting as an MQ client node.
- This capability allows the DataPower device to bridge disparate messaging and transport protocols, such as HTTP or TIBCO EMS, WebSphere JMS to WebSphere MQ.
- Messages originating within or outside of an MQ messaging bus can flow easily to and from another MQ messaging bus or other messaging system, such as HTTP, TIBCO EMS or WebSphere JMS.
- It is the Multi-Protocol Gateway service running on the DataPower device that makes this possible.

(*): DataPower appliances that support MQ

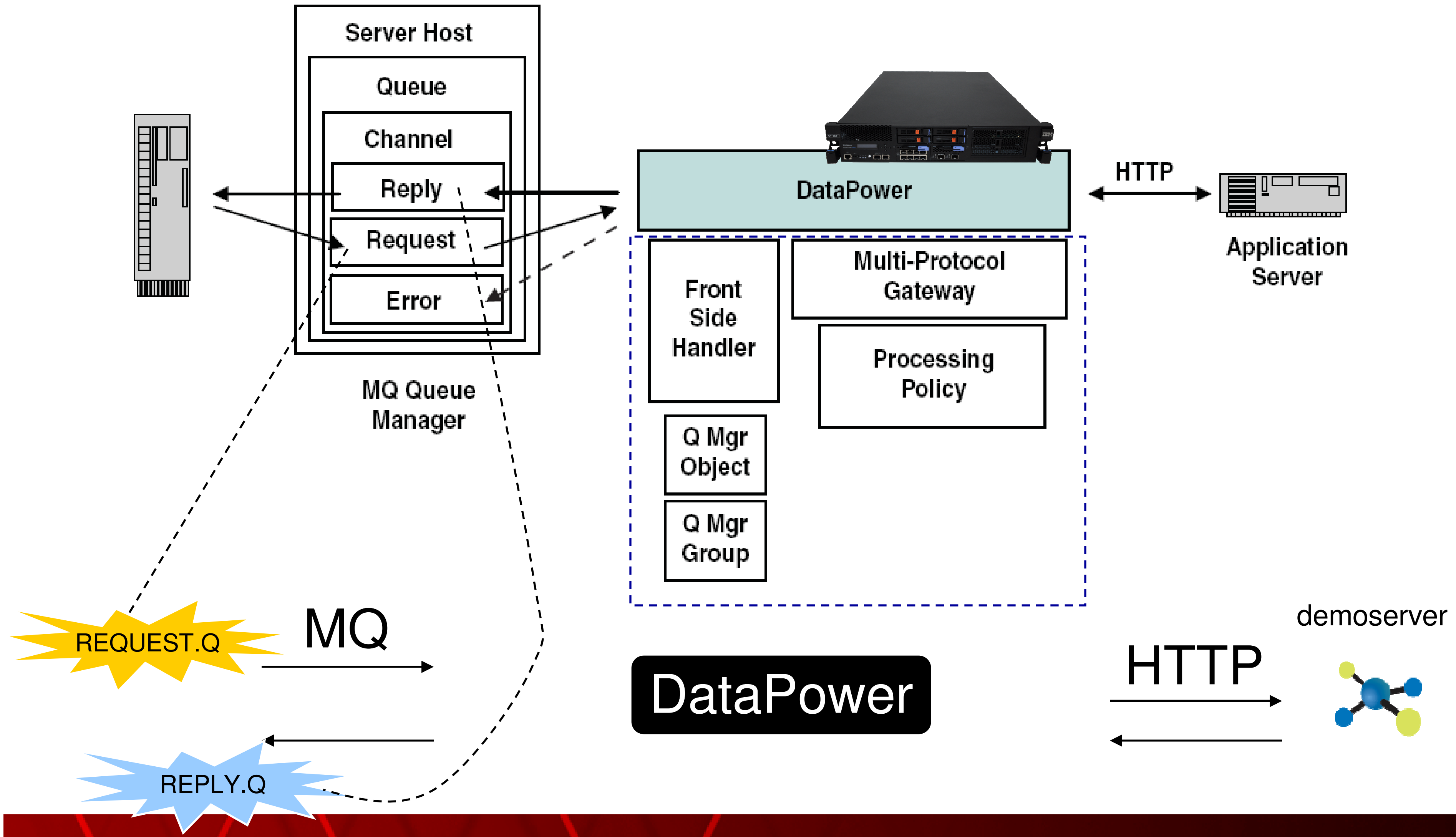
Basic MQ Architecture – HTTP to MQ



Message WorkFlow – HTTP to MQ

1. The HTTP client sends an HTTP-based request (typically an HTTP Post containing a SOAP XML document, but might contain binary data) to the DataPower device. An HTTP Front Side Protocol Handler listens on an assigned port for incoming requests
2. The Front Side Handler passes the message to the Multi-Protocol Gateway service object. The Multi-Protocol Gateway then applies any and all relevant Processing Policy actions on the message
3. The Multi-Protocol Gateway can dynamically determine the appropriate destination to which to route the message, or can route all messages statically to a particular destination. In either case, in this architecture, the destination is a particular queue managed by a particular MQ Queue Manager. The DataPower MQ Queue Manager object contains the necessary information to establish a connection to the MQ Queue Manager
4. The message is placed on the destination queue with MQPUT.
5. The DataPower device polls the Reply-To queue specified in the Destination URL to find a correlated response message.
6. The Multi-Protocol Gateway examines the Correlation ID value in the MQ header of messages on the Reply-to queue; when this ID is the same as the Message ID assigned to the request, the Multi-Protocol Gateway takes the message as the response

Basic MQ Architecture – MQ to HTTP



Message Workflow – MQ to HTTP

1. An MQ Front Side Protocol Handler polls the configured Request queue, managed by the referenced MQ Queue Manager, for incoming requests. All messages found on the queue are copied from the queue
2. The Front Side Handler passes the message to the Multi-Protocol Gateway service object. The Multi-Protocol Gateway then applies any and all relevant Processing Policy actions on the message
3. The Multi-Protocol Gateway gets the response from the back end application server. In this scenario, the HTTP protocol requires a response (which might contain an error message) or the Multi-Protocol Gateway will register an error
4. The response message can be placed on the Reply-to queue specified in the Front Side Protocol Handler. The Front Side Handler sets the MQMD CorrelID to the saved MsgID of the original request

Routing – Back side Request Routing

- A Multi-Protocol Gateway can route messages to and from MQ queues just as messages are routed to and from HTTP destinations. The Multi-Protocol Gateway can use static or dynamically determined routes
- **Back side Request routing**
  - A Multi-Protocol Gateway can use either static or dynamic routing to a back side destination.
    - When using a static route, the **Backend URL** property of the Gateway determines the route.
    - When using dynamic routing, the Gateway Processing Policy must set the destination during processing.
- The Gateway can examine the following inputs to help determine the desired message routing:
  - **Message content** The Multi-Protocol Gateway can dynamically determine the destination queue by employing an XPath routing map to examine the content of the message, or by using a custom stylesheet.
  - **Protocol header** The Multi-Protocol Gateway can also dynamically determine destination queue by examining the value of the MQ protocol headers. For a list of all headers, use the var://service/header-manifest service variable, which contains a nodeset of all supported headers found in the message, including the required MQMD header

Routing – Back side Destinations

- If the Multi-Protocol Gateway dynamically determines the back end destination, then the Multi-Protocol Gateway Processing Policy must set a back end target at some point using the Route action (or by using either the set-target() or xset-target() DataPower Extension Function calls in a custom stylesheet)
- Messages can also be sent, or routed to one or more alternative destinations using the Results (or Results Async) processing actions, just as with HTTP messages.
  - *For example*, a single request message might contain a number of attachments. These attachments could be separated from the original request and routed individually to a particular destination (that might not be an MQ queue).

    The processing policy of the Multi-Protocol Gateway could collect the responses and construct a reply message, could ignore the responses, or could send a response message that does nothing more than acknowledge receipt of the original request

Routing – URL

- The DataPower device employs an MQ URL to express a destination queue to which messages are put. An MQ URL is similar to the following example:
  - **dpmq**://QueueManager/URI?RequestQueue=PUTQ;ReplyQueue=GETQ;Sync=true;Model=true;Timeout=1000;PMO=2048

- You can also construct a dynamic MQ URL which points to an MQ Queue Manager which has not been defined by a static MQ Queue Manager object on the IBM WebSphere DataPower XML Integration Appliance XI50.

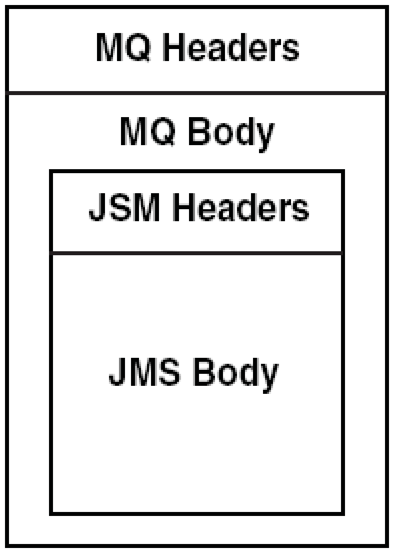
  Such dynamic MQ URLs take the form:
  - **mq**://ipAddress:Port/example?Channel=chanName;QueueManager=qmName;UserName=uName;ChannelLimit=maxChannels;ChannelTimeout=timeout;RequestQueue=requestQueueName;ReplyQueue=replyQueueName

Using MQ with a Web-Service Proxy

- The Web Service Proxy service can also interoperate with the MQ messaging system. Here are some of the methods available to create such a configuration
- The Proxy could employ an MQ Front Side Protocol Handler to get SOAP request messages from an MQ queue, and optionally place any response on a corresponding reply queue
- The Proxy could employ a statically-defined backend that uses MQ queues to put requests and get responses. As most WSDL files indicate an HTTP endpoint, you might need to manually change the Proxy type property to *static backend* from the typical default of *static-from-wsdl*. You will then need to specify the backend URL using the MQ URL syntax
- The Proxy could employ a Results (or Results Async) action in the Processing Policy that sends messages to an MQ queue

MQ & JMS

- It is possible to transport JMS messages over an MQ system. In such a case, some architectures allow for the selection of MQ messages based on values contained in the JMS message headers
- The DataPower device views JMS messages over MQ as illustrated here -- -- -- →
- It is possible to examine the nodeset contained in the *var://service/header-manifest* variable, which contains the parsed JMS headers

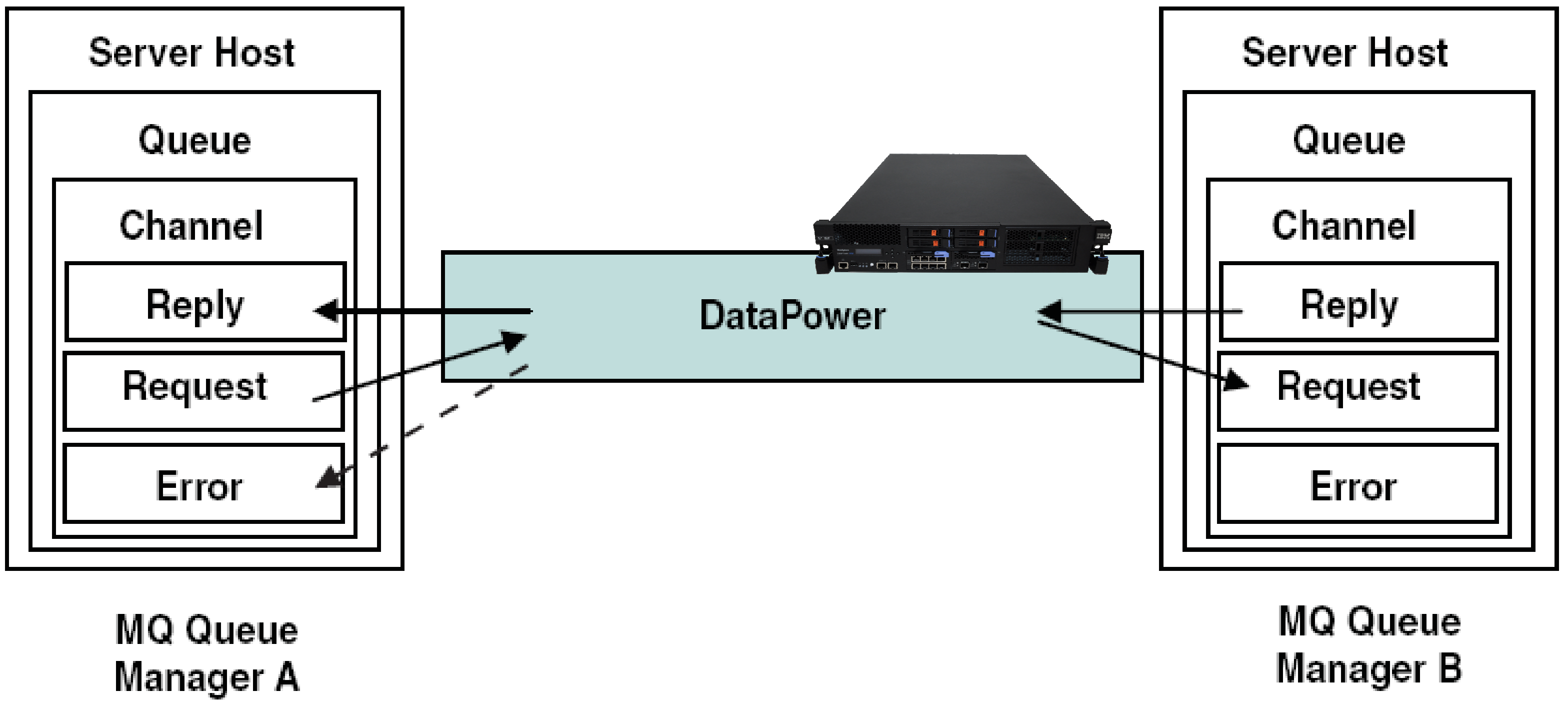


Units of Work

- The MQ protocol includes the concept of Units of Work - that is, the ability to remove messages from a queue only when the application that removed the message has successfully processed the message
- This concept also includes the ability to roll back transactions, much like a database system
- The standard MQ Client library supports the concept of *local Units of Work*.
  - A Unit of Work begins when an MQ client retrieves (or MQGETs) a message from a queue and ends when that same client successfully places (or MQPUTs) a response message on a queue managed by the same Queue Manager
- The Queue Manager managing the queue from which the request message is retrieved automatically holds a copy of the message on the queue and allows no other client to retrieve the same message until the Queue Manager receives either an MQCOMIT or an MQBACK signal on the same client connection
  - Upon receiving an MQCOMIT, the Queue Manager deletes the message
  - Upon receiving an MQBACK, the Queue Manager makes the message again available to any client for retrieval

DataPower Units of Work Implementation 1/4

- How does the DataPower XI52 device implement MQ Units of Work ?



DataPower Units of Work Implementation 2/4

1. MQ Front Side Handler connects to the QueueManager (MQCONNX) or gets a connection from the connection cache
2. MQ Front Side Handler performs an MQOPEN for the Get queue
3. If the DataPower device Queue Manager's **Units of Work** property is set to 1 (i.e. enabled), the MQGET is issued with SYNCPOINT=true to mark the start of a Unit of Work
4. The Front Side Handler passes the message returned by MQGET to the Multi-Protocol Gateway's Processing Policy
5. When the Request Rule completes, the Multi-Protocol Gateway sends the possibly altered message to the back end server
6. It is important to note that the back end MQPUT typically goes to a different Queue Manager than the front, and even if it is to the same Queue Manager as the front it would use a different connection. Therefore, the Unit of Work for the back end connection is not the same as the Unit of Work for the front side connection

DataPower Units of Work Implementation 3/4

7. The Multi-Protocol Gateway retrieves a response from the back end server using an MQGET issued to the queue specified by the ReplyToQueue of the destination URL
8. When the Processing Policy completes, MQ Front Side Handler is signaled and any response data from the backside is MQPUT to a front queue using one of the following methods:
   1. the Reply-to-Q in the MQMD of the reply message (which may be altered during response processing);
   2. the Reply-to-Q in MQMD of the initial request message;
   3. the statically configured Put Queue in MQ Front Side Handler which uses the *same* connection as the Get Queue of the MQ Front Side Handler
9. The MQ Front Side Handler issues an MQCOMIT
10. When a message remains on a request (GET) queue after a failure, the MQ Front Side Handler will again retrieve the same message. It is possible that this loop could continue indefinitely (although many Queue Managers detect messages left on a queue for a long time and remove them). It is possible to configure the MQ Queue Manager object used by the MQ Front Side Handler on the DataPower device to handle this case automatically. Set the **Automatic Backout** flag to On and then set the **BackoutThreshold** and **BackoutQueueName** properties of the Queue Manager to the appropriate values

DataPower Units of Work Implementation 4/4

- The MQ Front Side Handler using the Queue Manager tracks the MQMD.BackoutCount header value of the MQGET of the request (GET) queue.

- If the MQMD.BackoutCount reaches the configured backout threshold, the MQ Front Side Handler will move (MQPUT) the message on the queue identified by the **BackoutQueueName** property (typically the dead letter queue) and issue MQCMIT to the request queue to break the loop.

- Common Message Delivery Patterns
  - **Independent Asynchronous Delivery**
  - **Synchronous Delivery**

Conclusion

- Wrap-Up
- Questions ?