

---

MQvNext beta.... IBM MQ function candidates

# Notices and Disclaimers

---

Copyright © 2016 by International Business Machines Corporation (IBM). No part of this document may be reproduced or transmitted in any form without written permission from IBM.

## **U.S. Government Users Restricted Rights - Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM.**

Information in these presentations (including information relating to products that have not yet been announced by IBM) has been reviewed for accuracy as of the date of initial publication and could include unintentional technical or typographical errors. IBM shall have no responsibility to update this information. THIS DOCUMENT IS DISTRIBUTED "AS IS" WITHOUT ANY WARRANTY, EITHER EXPRESS OR IMPLIED. IN NO EVENT SHALL IBM BE LIABLE FOR ANY DAMAGE ARISING FROM THE USE OF THIS INFORMATION, INCLUDING BUT NOT LIMITED TO, LOSS OF DATA, BUSINESS INTERRUPTION, LOSS OF PROFIT OR LOSS OF OPPORTUNITY. IBM products and services are warranted according to the terms and conditions of the agreements under which they are provided.

## **Any statements regarding IBM's future direction, intent or product plans are subject to change or withdrawal without notice.**

Performance data contained herein was generally obtained in a controlled, isolated environments. Customer examples are presented as illustrations of how those customers have used IBM products and the results they may have achieved. Actual performance, cost, savings or other results in other operating environments may vary.

References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business.

Workshops, sessions and associated materials may have been prepared by independent session speakers, and do not necessarily reflect the views of IBM. All materials and discussions are provided for informational purposes only, and are neither intended to, nor shall constitute legal or other guidance or advice to any individual participant or their specific situation.

It is the customer's responsibility to insure its own compliance with legal requirements and to obtain advice of competent legal counsel as to the identification and interpretation of any relevant laws and regulatory requirements that may affect the customer's business and any actions the customer may need to take to comply with such laws. IBM does not provide legal advice or represent or warrant that its services or products will ensure that the customer is in compliance with any law

# Notices and Disclaimers Con't.

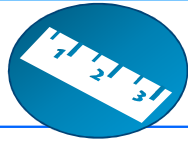
---

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products in connection with this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products. IBM does not warrant the quality of any third-party products, or the ability of any such third-party products to interoperate with IBM's products. IBM EXPRESSLY DISCLAIMS ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents, copyrights, trademarks or other intellectual property right.

IBM, the IBM logo, ibm.com, Aspera®, Bluemix, Blueworks Live, CICS, Clearcase, Cognos®, DOORS®, Emptoris®, Enterprise Document Management System™, FASP®, FileNet®, Global Business Services®, Global Technology Services®, IBM ExperienceOne™, IBM SmartCloud®, IBM Social Business®, Information on Demand, ILOG, Maximo®, MQIntegrator®, MQSeries®, Netcool®, OMEGAMON, OpenPower, PureAnalytics™, PureApplication®, pureCluster™, PureCoverage®, PureData®, PureExperience®, PureFlex®, pureQuery®, pureScale®, PureSystems®, QRadar®, Rational®, Rhapsody®, Smarter Commerce®, SoDA, SPSS, Sterling Commerce®, StoredIQ, Tealeaf®, Tivoli®, Trusteer®, Unica®, urban{code}®, Watson, WebSphere®, Worklight®, X-Force® and System z® Z/OS, are trademarks of International Business Machines Corporation, 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](http://www.ibm.com/legal/copytrade.shtml).

# IBM MQ V8 delivering best in class enterprise messaging



## Platforms & Standards

## Security

## Scalability

## System z exploitation

64-bit for all platforms

Userid authentication via OS & LDAP

Multiplexed client performance

64-bit buffer pools in MQ for z/OS means less paging, more performance

Multiple Cluster Transmit Queue on all platforms

User-based authorisation for Unix

Queue manager vertical scaling

Performance and capacity, larger log capacity.

Support for JMS 2.0

AMS integration for IBM i & z/OS

Publish/Subscribe improvements

Performance enhancements for IBM Information Replicator (QRep)

Improved support for .Net and WCF

DNS Hostnames in CHLAUTH records

Routed publish/subscribe

Exploit zEDC compression accelerator & FlashExpress

SHA-2 for z, i & NSS

Multiple certificates per queue manager

Announced  
22 April 2014  
GA Distrib 23 May  
GA z/OS 13 June

SMF enhancements

# MQvNext themes

---

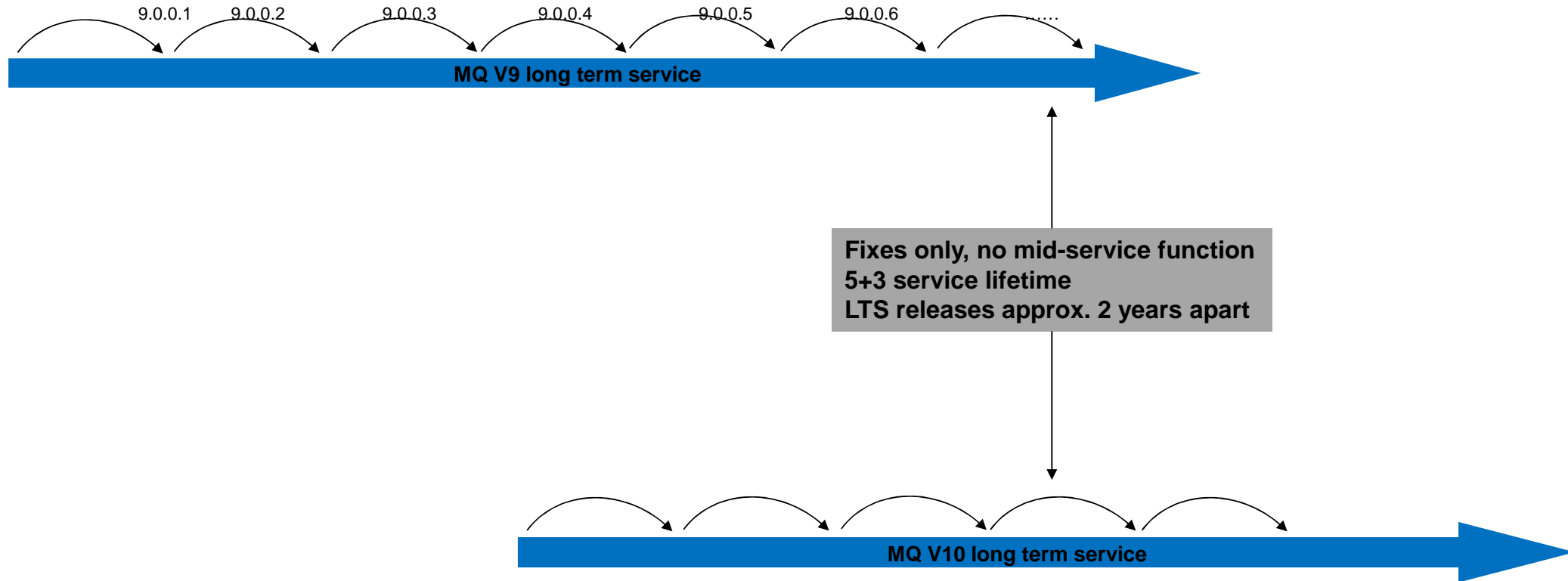
- MQ Continuous Delivery
- Move towards MQ “self-service”
- Cloud-enable MQ
- Reduce complexity of MQ
  - Development
  - Deploying
  - Administration
  - Elastic scaling
  - Location transparency
- As always, leverage latest technology enhancements

# Post MQv8 enhancements already delivered

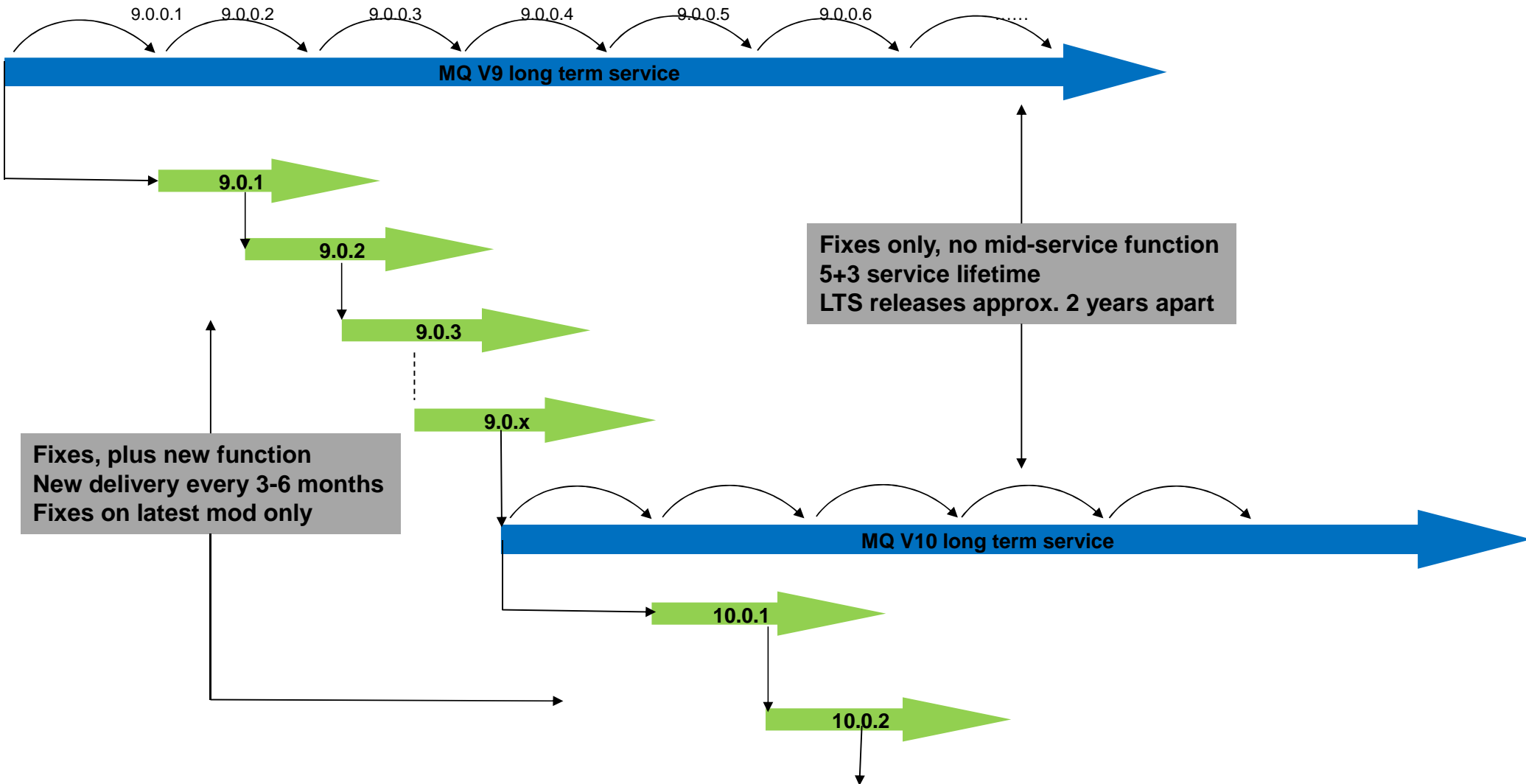
---

- Capped Message Expiry for queues (CAPEXPY)
- CICS java programs can use the IBM MQ classes for JMS in the CICS® Open Services Gateway initiative (OSGi) Java™ Virtual Machine (JVM) server
- IMS java programs can use the IBM MQ classes for JMS, JMS 2.0 spec

# Traditional MQ releases



# MQ Continuous Delivery Releases - Stable and Rapid



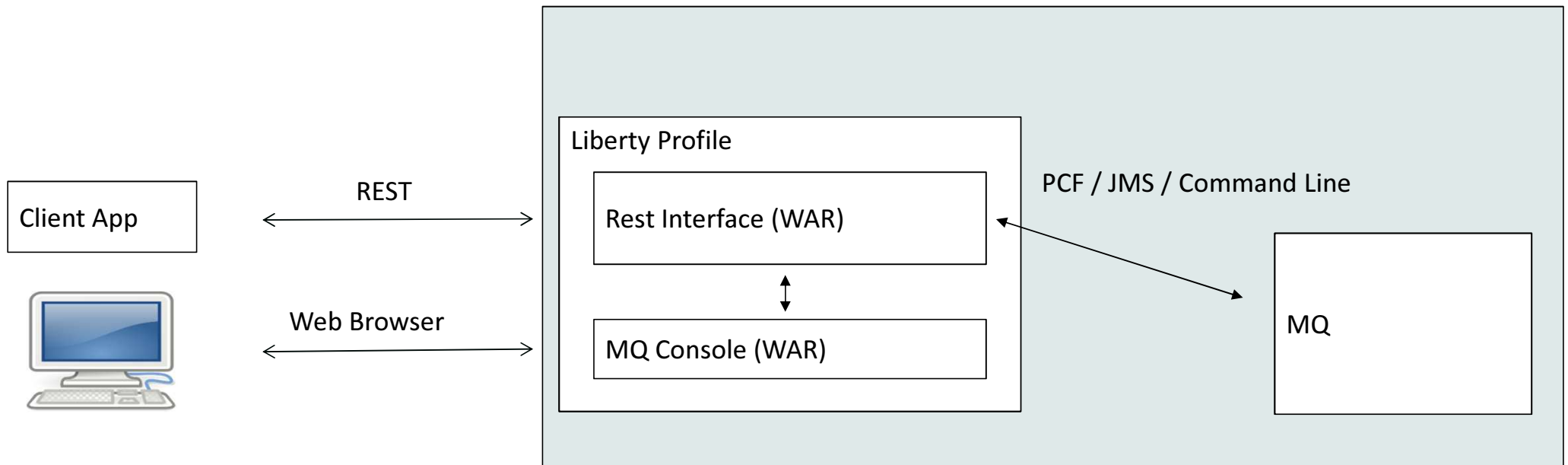


# MQvNext possible features in beta programs

---

- Channel partner information in channel exits
  - Allow administrators to detect, and potentially reject, older, unsupported MQ Clients
- Buffer pool exploitation of 1M fixed pages
  - Large performance gains in particular for non-persistent messages; leverages “mega-memory” of latest z hardware
- MQ CCDT updates (XML instead of binary, support for URL to remotely obtain remote info)
  - Render MQ architecture “location transparent” for MQ Clients – users connect to MQ, not to a queue manager
- Support for AMQP protocol (eg. MQ Light API) on MQ z/OS
  - New CHLTYPE(AMQP) on CHIN, and motor running in Java; MQ Light clients connect to defined IP port
- MQ z/OS Provisioning
  - z/OSMF scripts; facilitate developer testing, simplify MQ sys.prog tasks
- MQ Admin REST interface
  - Simplify development of administration interfaces; simplify integration of MQ administration
- MQ Web Console
  - Reduce footprint for administrators
- z/OS Connect MQ Service Provider
  - Simplify development of mobile applications for MQ z/OS
  - Leverage common z/OS subsystem interface for exposing z/OS resources

# MQ Admin REST interface



- The REST Interface and MQ Console sits inside of the Liberty Profile and can provide support to an MQ installation on the box.
- Packaged as an EAR file hosted by the Liberty Profile

# MQ Console

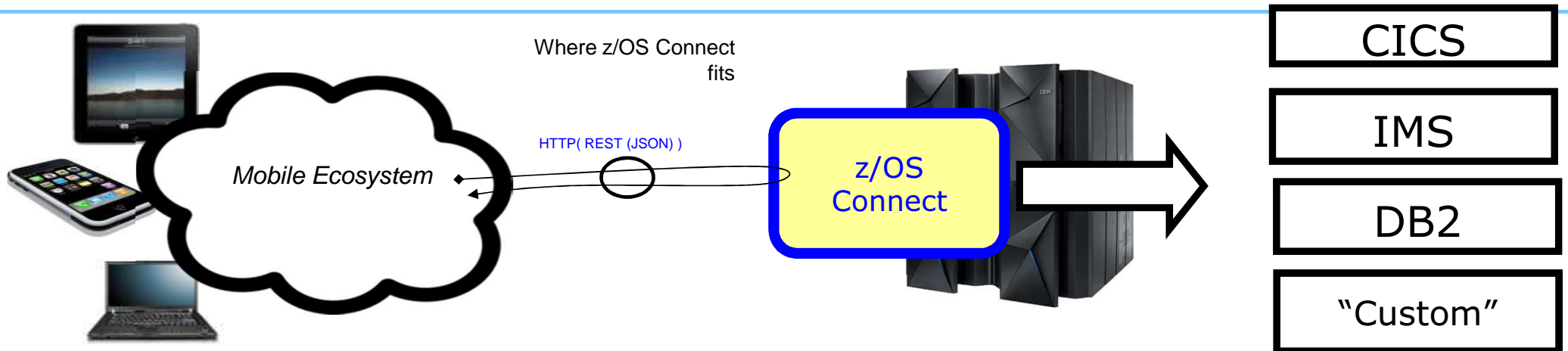
- Web Interface

- Based upon work done for the MQ Appliance
- Dashboard approach
- Collections of objects as widgets
- Each user can have a different dashboard with their own collection of interested objects
- Under the covers it is separate from the REST interface, it's just another client

The screenshot displays the IBM MQ Console web interface for QM1. The dashboard includes several widgets:

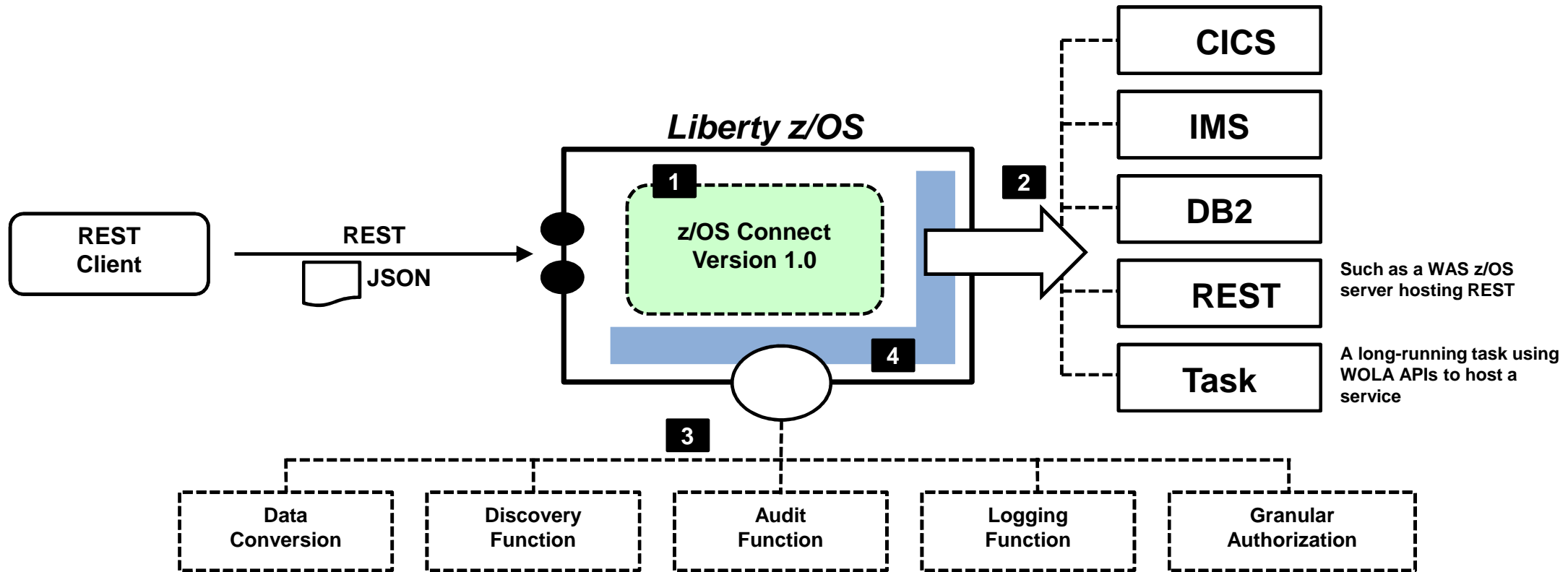
- Queues on QM1:** A table listing system queues with columns for Name, Queue type, and Queue depth. Total: 57 Selected: 0. Updated: 4:33:36 PM.
- Client-connection Channels on QM1:** A table listing client-connection channels. Total: 1 Selected: 0. Updated: 4:33:28 PM.
- Channels on QM1:** A table listing various channels with columns for Name, Type, and Overall channel status. Total: 9 Selected: 0. Updated: 4:33:33 PM.
- Listeners on QM1:** A table listing listeners with columns for Name, Port, and Listener status. Total: 5 Selected: 0. Updated: 4:33:31 PM.
- Subscriptions on QM1:** A widget with a Filter... field.
- Topics on QM1:** A widget with More... and Filter... fields.

## So what is z/OS Connect?



- A z/OS mobile gateway: Consistent entry point for mobile access to one or more backend systems. Runs only on z/OS.
- Shields backend systems from requiring awareness of RESTful URIs and JSON data formatting
- WAS Liberty based: Quick to install, configure. Lightweight and modular.
- Java, so runs on specialty engines (zAAP, zIIP)
- Provides point for capturing usage information using SMF
- Introspection capability for rapid development and API management
- Fast, scalable and maintainable in an HA environment

# z/OS Connect components



1. z/OS Connect a “feature” of Liberty

2. “Service Provider” = backend connectivity

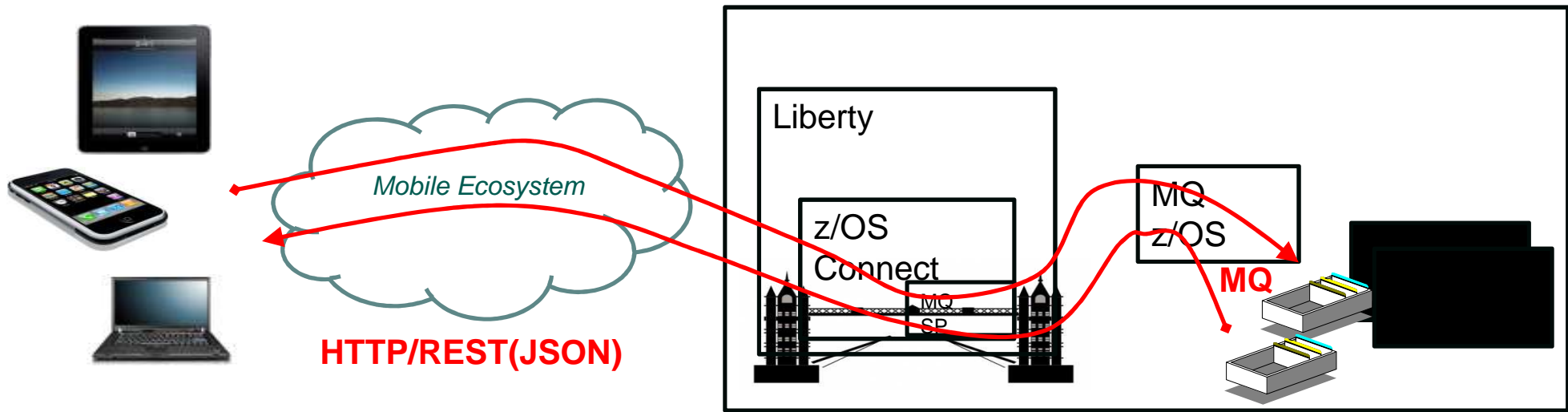
3. “Interceptors” = configurable function

4. Extensible interface = flexibility

# z/OS Connect: MQ Service Provider

- A new z/OS Connect Service Provider for MQ; becomes an MQ z/OS “gateway”.
- Remember: MQ has already made available for several years the “IBM MQ Bridge for HTTP” which is also a REST-MQ gateway (search “q033190\_” in the KC). This is a simple JEE servlet that IBM supports.
- This new z/OS Connect Service Provider is a plug-in for z/OS Connect, so takes advantage of the richer z/OS Connect framework (interface definition, introspection, SMF, security model, etc.).
- The basic services provided:
  - ✓ HTTP POST → MQPUT (queue or topic)
  - ✓ HTTP DELETE → MQGET (destructive, queue)
  - ✓ HTTP GET → MQGET (non-destructive, queue)
- Advanced services:
  - ✓ HTTP POST → MQPUT to a queue and the SP waits for a reply message that gets returned to the client
- Payloads are JSON only; EBCDIC <> ASCII translation supported.
- Support for numerous basic MQ options (eg. Expiry, Wait interval, Persistence....).
- Caveat: whilst this z/OS Connect interface to MQ provides a nice function, just remember that applications built upon this REST interface to MQ are losing part of that asynchronous, robust (eg. Transactional) advantage of MQ.

# z/OS Connect MQ Service Provider



- Simple “one way” MQPUT (POST), MQGET (GET or DELETE), PUBLISH (POST) or “round-trip” request/reply (MQPUT & MQGET with wait)
- Supports Queues or Topics
- Payload JSON downstream, but z/OS Connect transformation can be done upstream
- Many MQ options (eg. Persistence, replyQ, etc.) supported

Warning: MQ v.Next beta – this may never become part of MQ!

# Testing z/OS Connect with MQ SP

## 1. Preliminaries

- Set up and test a basic Liberty with z/OS Connect (v1 only for now). Check WP102439 if you need hints for this.
- Have a running local MQ z/OS queue manager (any supported version should be ok)
- Get the code (com.ibm.ws.zos.connect.mq\_1.0.0.jar) (only in beta program today)

## 2. Basic enablement for LPAR

- Copy jar into Liberty library, eg. /usr/lpp/zWebSphere/Liberty/V8557/lib
- Copy manifest (zosConnectMQ-1.0.mf) into Liberty features, eg.  
/usr/lpp/zWebSphere/Liberty/V8557/lib/features

## 3. Specific Liberty server.xml configuration

- Enable the feature:  

```
<feature>wmqJmsClient-1.1</feature>  
<feature>zosConnectMQ-1.0</feature>
```
- Enable JMS
- Define my z/OS Connect MQ services

## 4. Test



# server.xml for MQ SP tests

```

:
<variable name="wmqJmsClient.rar.location" value="/u/farkas/testDir/WMQ7505/wmq.jmsra.rar" />
<wmqJmsClient nativeLibraryPath="/usr/lpp/mqm/V8R0M0/java/lib" />

<jmsConnectionFactory connectionManagerRef="JMSConnMgr" id="TestQM" jndiName="jms/qcfTestQM">
  <properties.wmqJms queueManager="ZTMQ" transportType="BINDINGS" />
</jmsConnectionFactory>
<connectionManager id="JMSConnMgr" maxPoolSize="2"/>

<jmsQueue jndiName="jms/myQueue"><properties.wmqJms
  baseQueueManagerName="ZTMQ" baseQueueName="FARKAS.Test.Q" /></jmsQueue>
<jmsQueue jndiName="jms/myReplyQ"><properties.wmqJms
  baseQueueManagerName="ZTMQ" baseQueueName="FARKAS.REPLY.Q" /></jmsQueue>

<zosConnectService id="zosconnMQ1" invokeURI="/oneWay" serviceName="oneWay" serviceRef="oneWay" />
  <mqzOSConnectService id="oneWay" connectionFactory="jms/qcfTestQM" destination="jms/myQueue" />

<zosConnectService id="zosconnMQ2" invokeURI="/roundTrip" serviceName="roundTrip" serviceRef="rounder" />
  <mqzOSConnectService id="rounder" connectionFactory="jms/qcfTestQM" destination="jms/myQueue"
    replyDestination="jms/myReplyQ" waitInterval="20000" replySelection="none" persistence="false" />
:

```

Defs for using MQ  
JMSv1.1

Defs for using MQ in  
Bindings mode

Define a few queues

Define z/OSC MQ  
services

# A few MQ SP tests

MQPUT with <https://9.212.143.123:20668/oneWay>

MQGET with <https://9.212.143.123:20668/oneWay>

The screenshot shows the Advanced Rest Client interface for an MQPUT request. The URL is `https://9.212.143.123:20668/oneWay`. The request method is set to PUT. The payload is a JSON object: `{ "DFH0XCINOperation": { "outer": { "test": "1" } } }`. The response status is `204: No Content` with a loading time of 66ms. The response headers are: `Content-Language: en-US`, `X-Powered-By: Servlet/3.0`, `Access-Control-Allow-Origin: chrome-extension://hgml0ofddfdnphfgcellkdfbfjeloo/`, `Access-Control-Allow-Credentials: true`, `Content-Type: application/json`, `Content-Length: 0`, and `Date: Wed, 27 Jan 2016 08:53:35 GMT`.

The screenshot shows the Advanced Rest Client interface for an MQGET request. The URL is `https://9.212.143.123:20668/oneWay`. The request method is set to GET. The response status is `200: OK` with a loading time of 30ms. The response headers are: `Content-Language: en-US`, `X-Powered-By: Servlet/3.0`, `Content-Type: application/json`, `Content-Length: 44`, and `Date: Wed, 27 Jan 2016 08:53:30 GMT`. The response body is a JSON object: `{ "DFH0XCINOperation": { "outer": { "test": "1" } } }`. A yellow callout box with the text "Note the HTTP GET here" points to the GET method selection in the interface.

# MQ SP property setting test

The screenshot shows the Advanced Rest Client interface. The URL is `https://9.212.143.123:20668/oneWay`. The request method is `POST`. The headers section shows `Content-Type: application/json` and `ibm-mq-user: myProp1;"CarlString";string`. The payload is a JSON object: `{ "DFH0XCMNOperation": "outer": { "test": "msg prop test1" } }`. A yellow callout points to the `ibm-mq-user` header with the text "I set a MQ message property here".

The response window shows the "Named Properties" section with the following table:

Name	Value
JMSDeliveryMode	1
JMSDestination	queue://ZTMQ/FARKAS.Test.Q
JMSTimestamp	1453886022504
mcd.Msd	jms_text
myProp1	CarlString

A yellow callout points to the `myProp1` property in the table with the text "MQ Explorer display of property here".

# MQ SP round-trip

The screenshot displays the Advanced Rest Client interface. The URL bar shows `https://9.212.143.123:20668/roundTrip`. The request method is set to `POST`. The headers section shows `Content-Type: application/json` and `ibm-mq-usr: myProp1;CarlString;string`. The payload is a JSON object: `{ "DFH0X(PWOperation)": { "outer": { "test": "request" } } }`. The status bar indicates `200: OK` with a loading time of `3252ms`. The response headers section lists: `X-Powered-By: Servlet/3.0`, `Access-Control-Allow-Origin: chrome-extension://hgmloofddfdnphfgcellkdfbfjelo`, `Access-Control-Allow-Credentials: true`, `Content-Type: application/json`, `Content-Language: en-US`, `Content-Length: 21`, and `Date: Wed, 27 Jan 2016 09:44:53 GMT`. The response body is a JSON object: `{ "reply": "the reply" }`.

Note the POST here

Different service (URL) here which has WAIT defined

And here's the reply (synchronous)