MQvNext beta…. IBM MQ function candidates

Notices and Disclaimers

Notices and Disclaimers Con't.

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		SMF enhancements

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

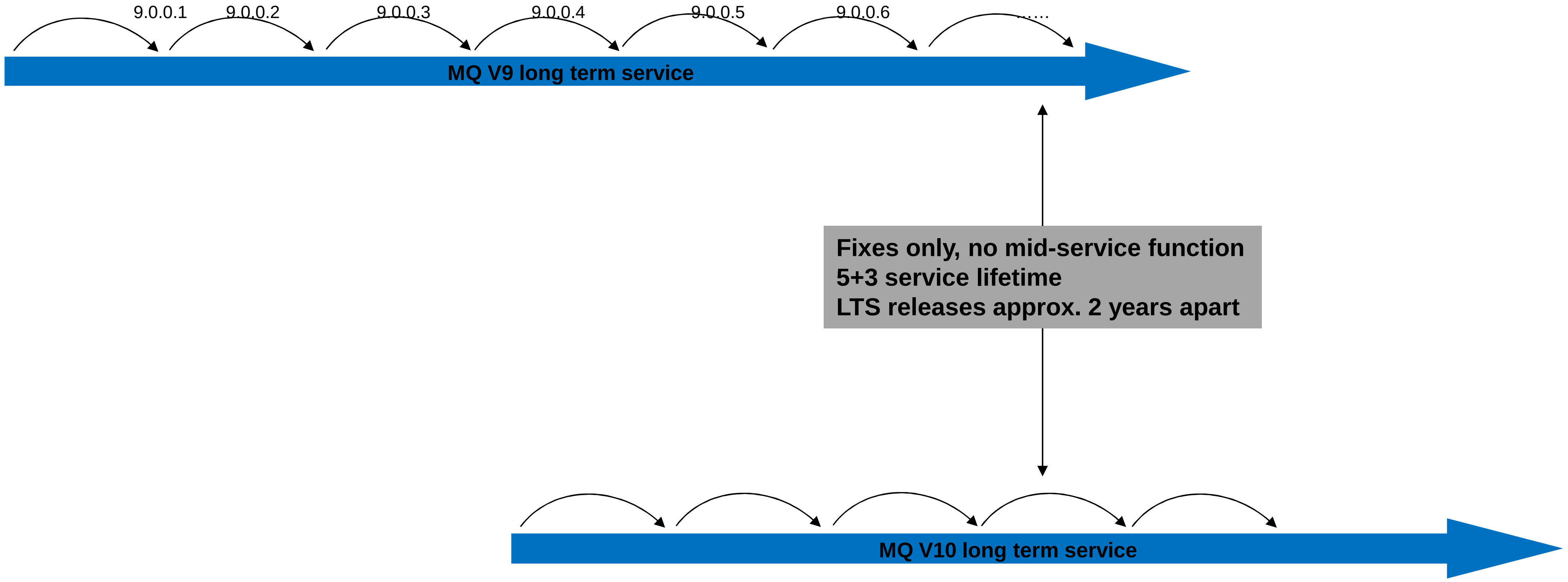
MQvNext themes

- 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

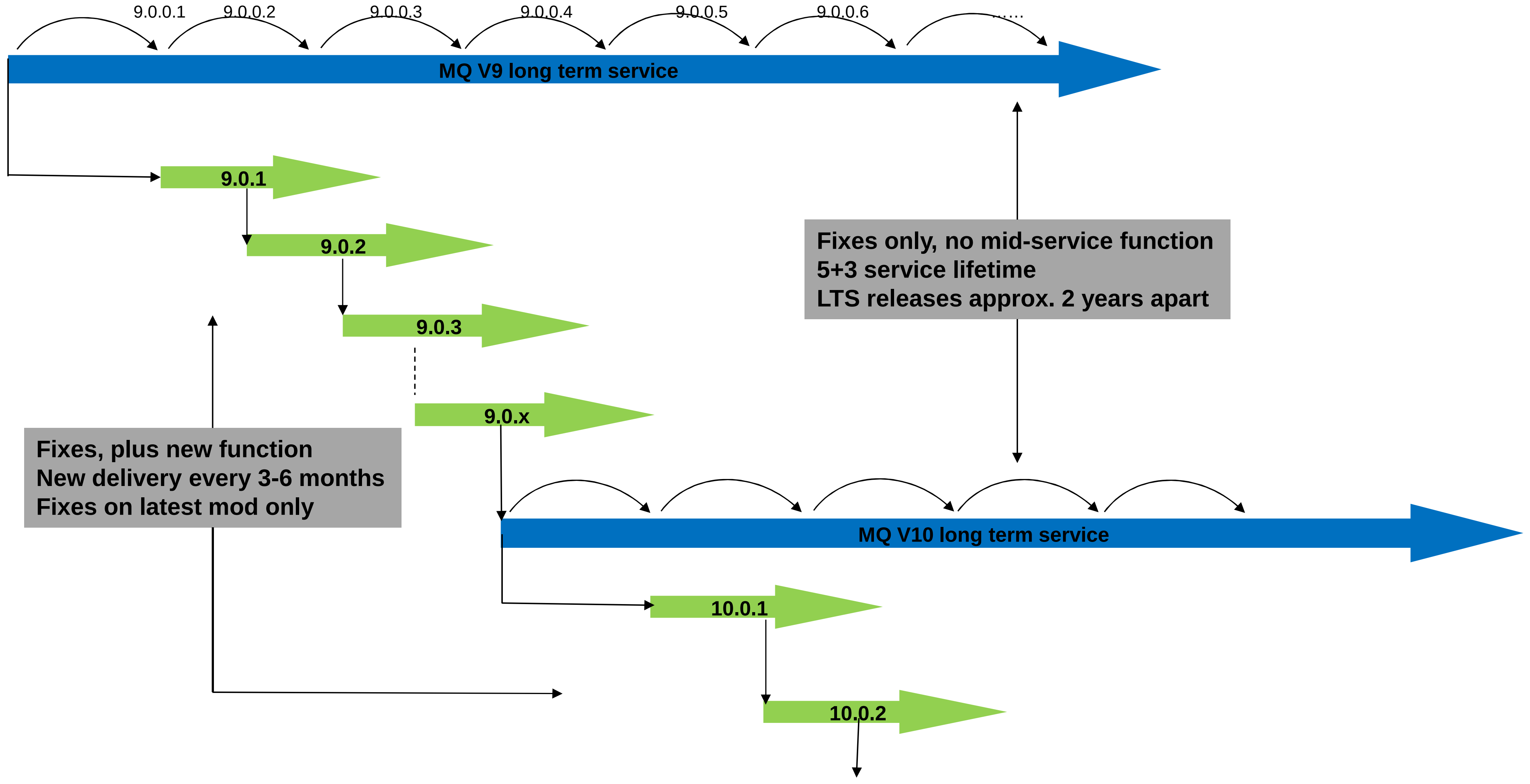
Post MQv8 enhancements already delivered

- Capped Message Expiry for queues (CAPEXPRY)
- CICS java programs san 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



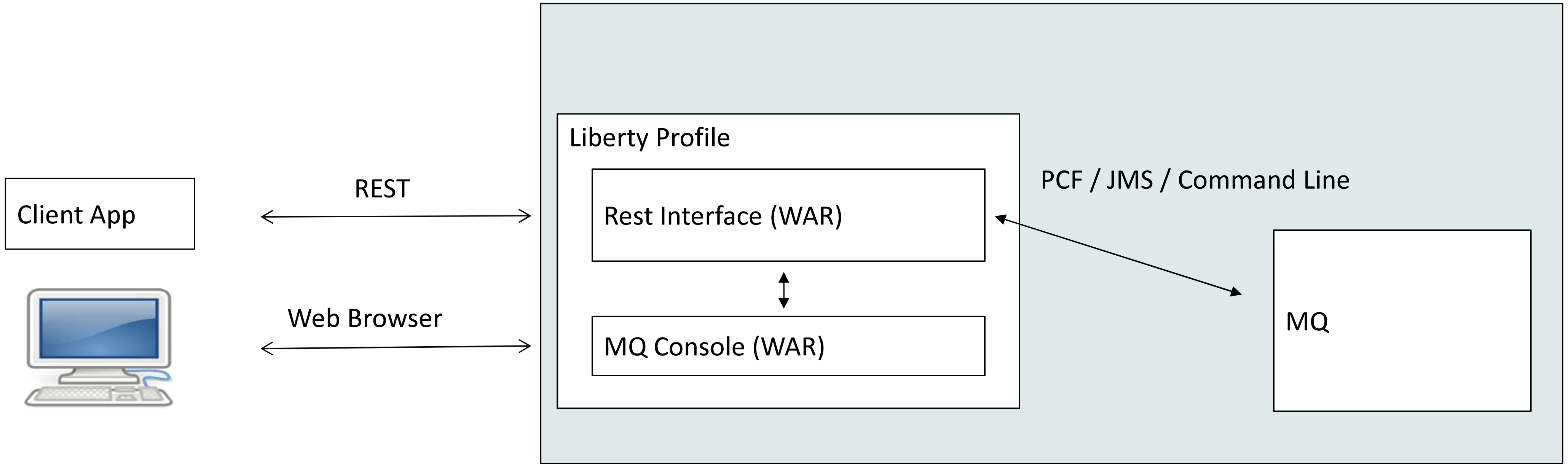
Fixes only, no mid-service function
5+3 service lifetime
LTS releases approx. 2 years apart

Fixes, plus new function
New delivery every 3-6 months
Fixes on latest mod only

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

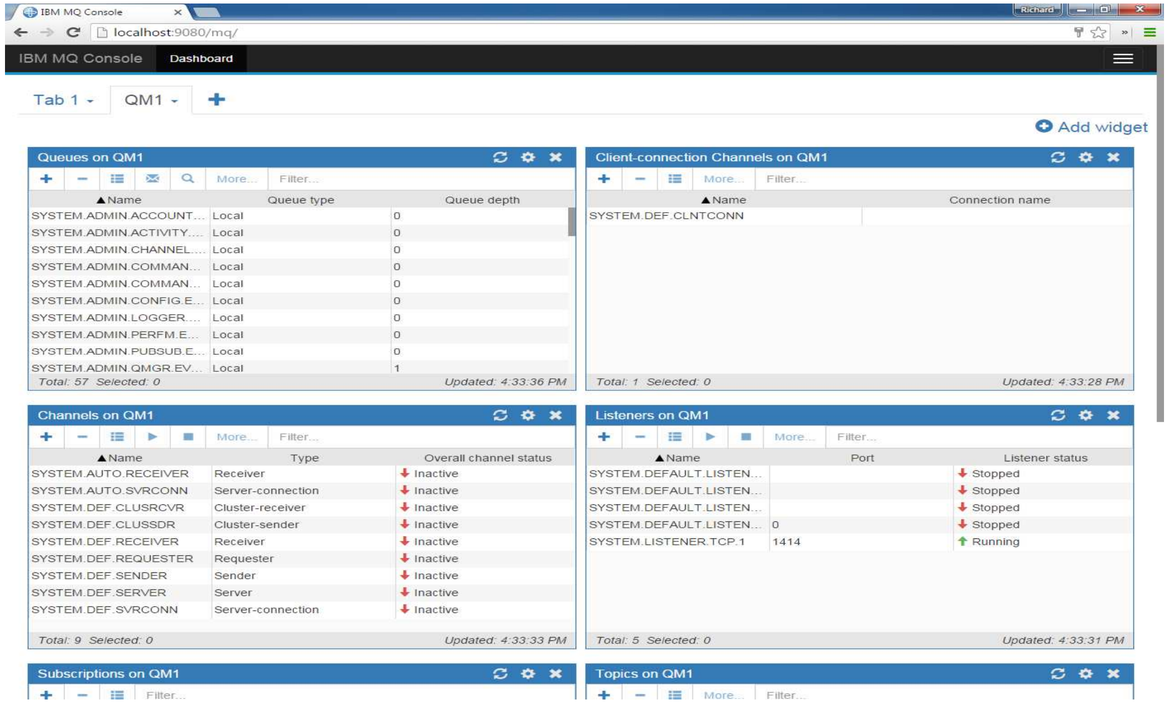
MQ Admin REST interface



- 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



Queues on QM1

▲Name	Queue type	Queue depth
SYSTEM.ADMIN.ACCOUNT...	Local	0
SYSTEM.ADMIN.ACTIVITY....	Local	0
SYSTEM.ADMIN.CHANNEL....	Local	0
SYSTEM.ADMIN.COMMAN...	Local	0
SYSTEM.ADMIN.COMMAN...	Local	0
SYSTEM.ADMIN.CONFIG.E...	Local	0
SYSTEM.ADMIN.LOGGER....	Local	0
SYSTEM.ADMIN.PERFM.E...	Local	0
SYSTEM.ADMIN.PUBSUB.E...	Local	0
SYSTEM.ADMIN.QMGR.EV...	Local	1

Total: 57 Selected: 0 *Updated: 4:33:36 PM*

Client-connection Channels on QM1

▲Name	Connection name
SYSTEM.DEF.CLNTCONN	

Total: 1 Selected: 0 *Updated: 4:33:28 PM*

Channels on QM1

▲Name	Type	Overall channel status
SYSTEM.AUTO.RECEIVER	Receiver	↓ Inactive
SYSTEM.AUTO.SVRCONN	Server-connection	↓ Inactive
SYSTEM.DEF.CLUSRCVR	Cluster-receiver	↓ Inactive
SYSTEM.DEF.CLUSSDR	Cluster-sender	↓ Inactive
SYSTEM.DEF.RECEIVER	Receiver	↓ Inactive
SYSTEM.DEF.REQUESTER	Requester	↓ Inactive
SYSTEM.DEF.SENDER	Sender	↓ Inactive
SYSTEM.DEF.SERVER	Server	↓ Inactive
SYSTEM.DEF.SVRCONN	Server-connection	↓ Inactive

Total: 9 Selected: 0 *Updated: 4:33:33 PM*

Listeners on QM1

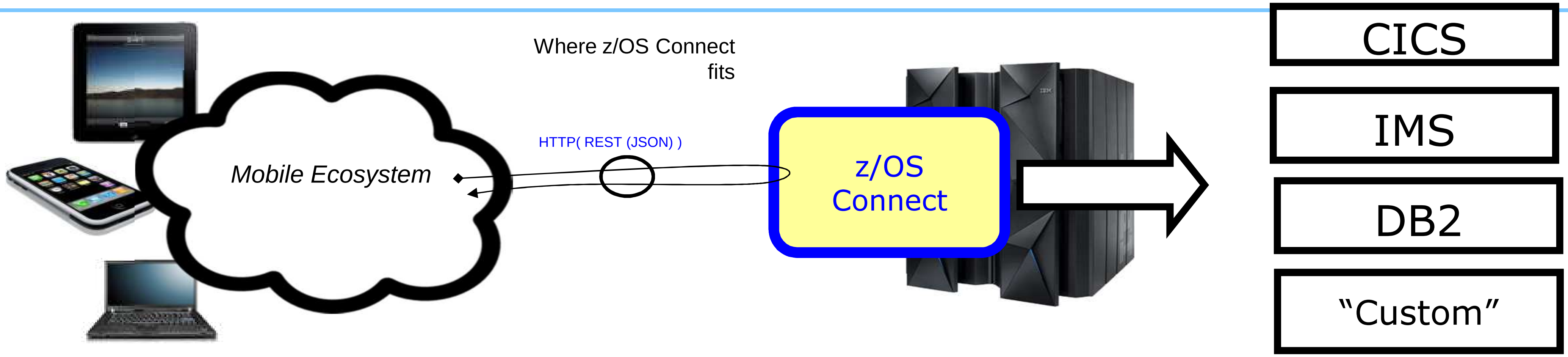
▲Name	Port	Listener status
SYSTEM.DEFAULT.LISTEN...		↓ Stopped
SYSTEM.DEFAULT.LISTEN...		↓ Stopped
SYSTEM.DEFAULT.LISTEN...		↓ Stopped
SYSTEM.DEFAULT.LISTEN...	0	↓ Stopped
SYSTEM.LISTENER.TCP.1	1414	↑ Running

Total: 5 Selected: 0 *Updated: 4:33:31 PM*

Subscriptions on QM1

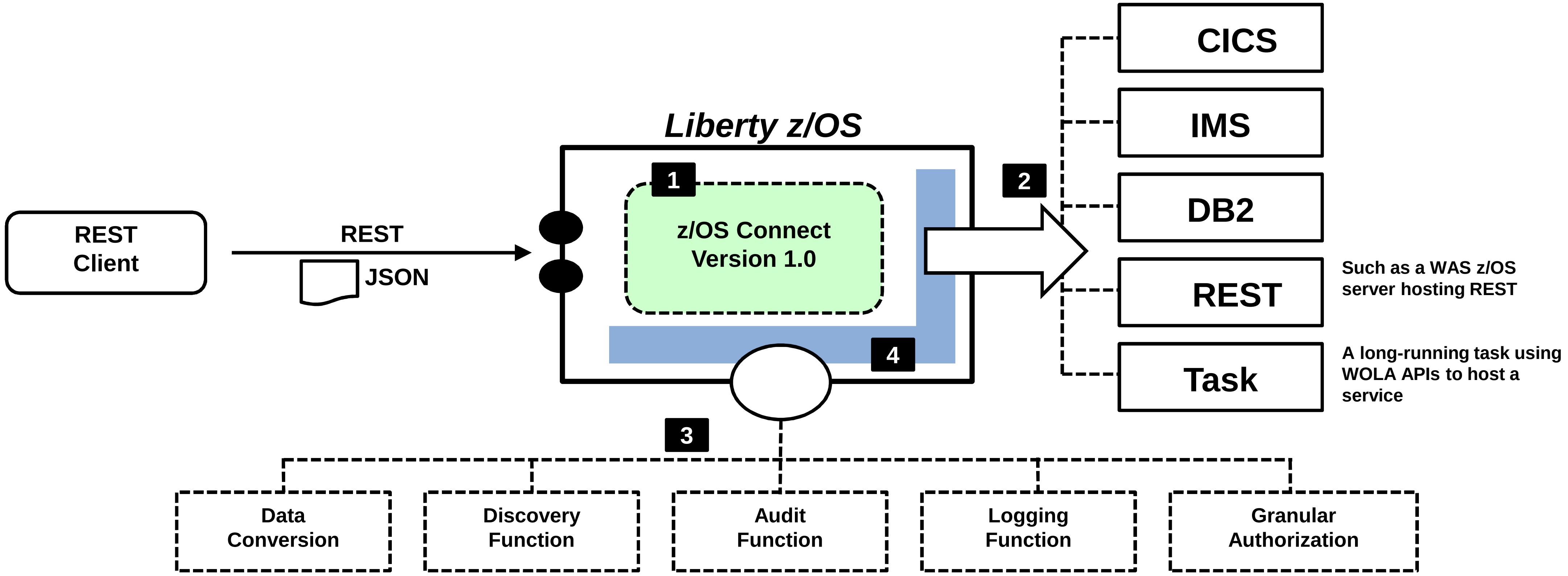
Topics on QM1

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

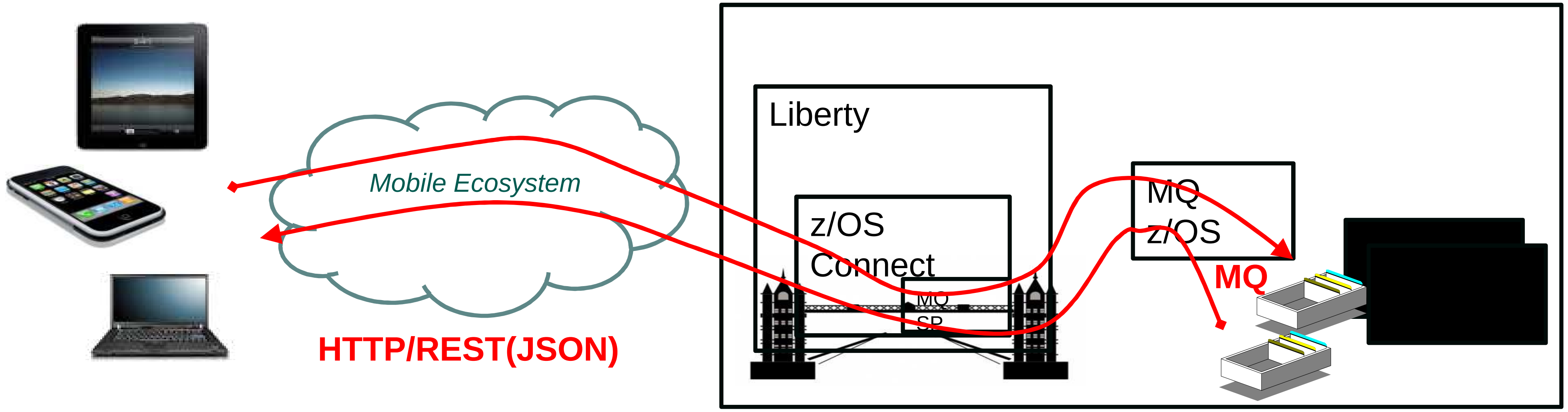


1. z/OS Connect a “feature” of Liberty
2. “Service Provider” = backend connectivity
3. “Interceptors” = configurable function
4. Extensible interface = flexibility

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

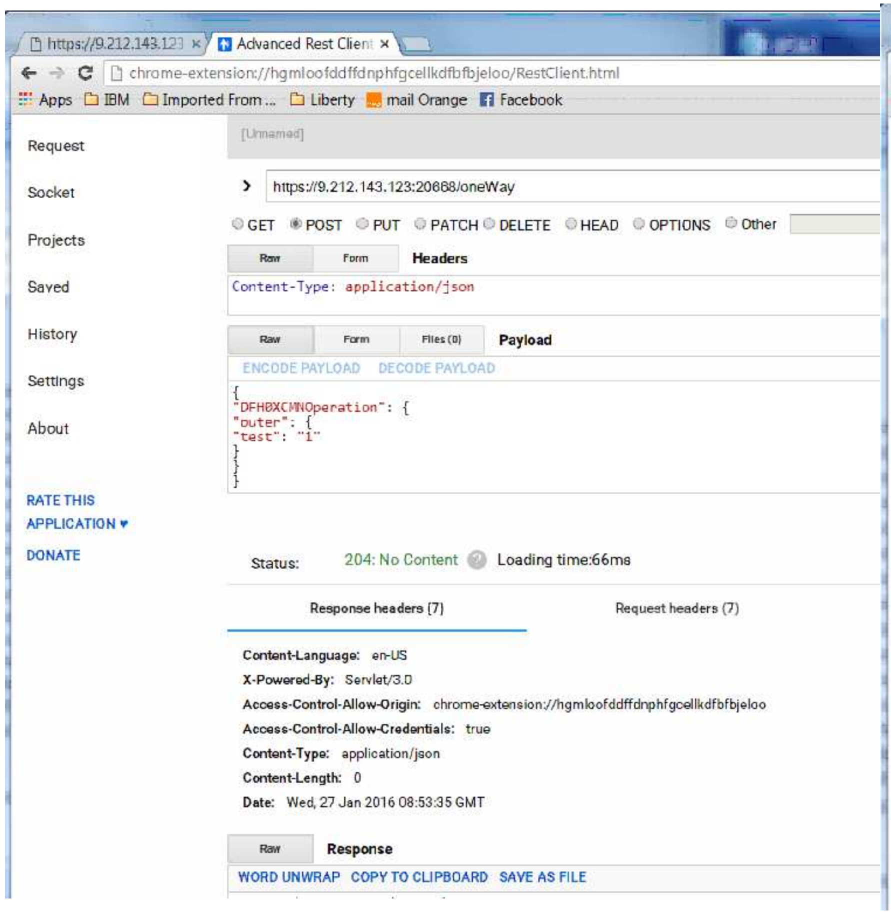
Defs for using MQ in Bindings mode

Define a few queues

Define z/OSC MQ services

A few MQ SP tests

MQPUT with https://9.212.143.123:20668/oneWay



https://9.212.143.123:20668/oneWay

GET [✓ POST] PUT PATCH DELETE HEAD OPTIONS Other

Headers

```
Content-Type: application/json
```

Payload

```
{
"DFH0XCMNOperation": {
"outer": {
"test": "1"
}
}
}
```

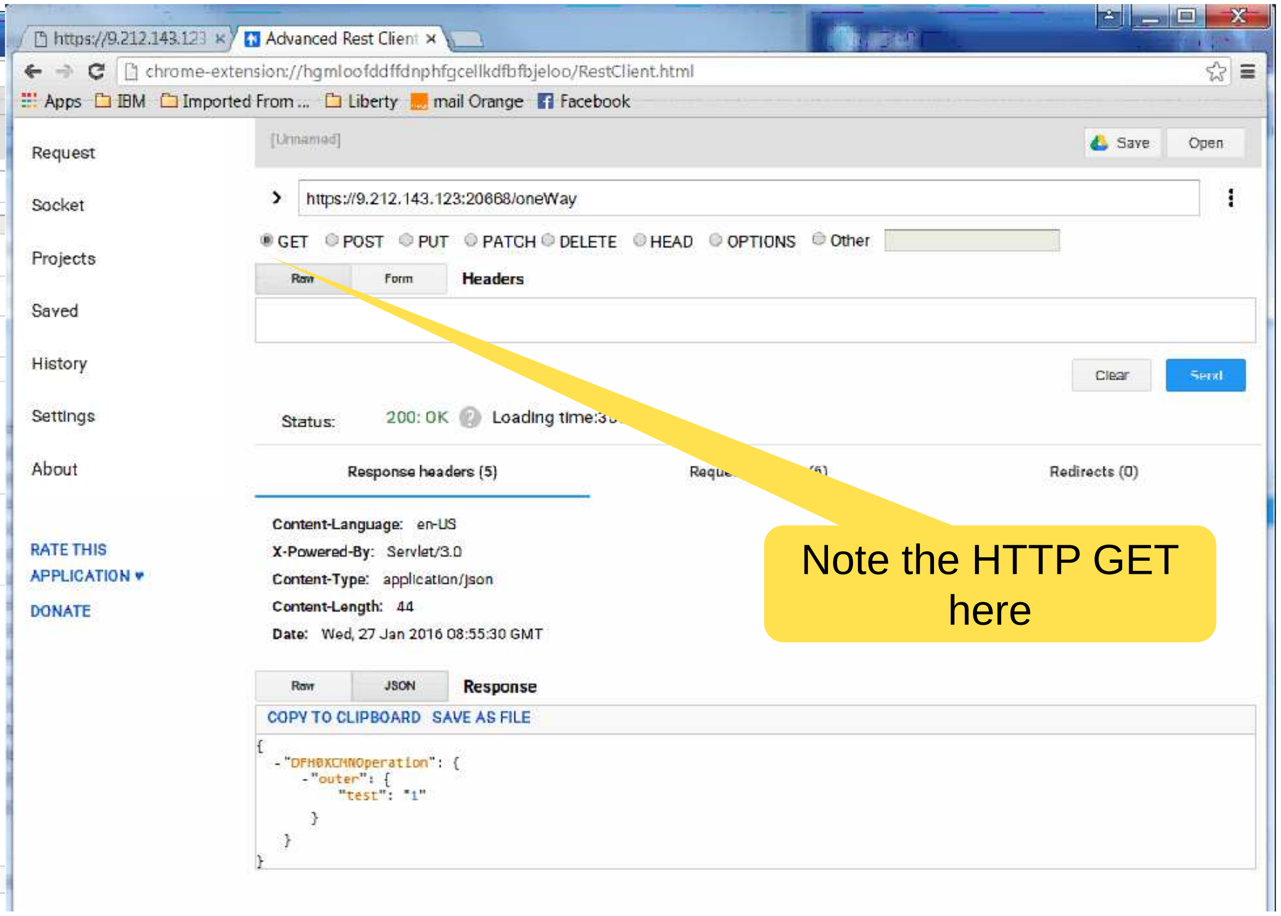
Status: 204: No Content Loading time:66ms

Response headers (7)

```
Content-Language: en-US
X-Powered-By: Servlet/3.0
Access-Control-Allow-Origin: chrome-extension://hgmloofddffdnphfgcellkdfbfbjeloo
Access-Control-Allow-Credentials: true
Content-Type: application/json
Content-Length: 0
Date: Wed, 27 Jan 2016 08:53:35 GMT
```

https://9.212.143.123
Advanced Rest Client
chrome-extension://hgmloofddffdnphfgcellkdfbfbjeloo/RestClient.html
Apps IBM Imported From ... Liberty mail Orange Facebook
Request
Socket
Projects
Saved
History
Settings
About
RATE THIS APPLICATION ♥
DONATE
[Unnamed]
Raw Form Headers
Raw Form Files (0) Payload
ENCODE PAYLOAD DECODE PAYLOAD
Response headers (7) Request headers (7)
Raw Response
WORD UNWRAP COPY TO CLIPBOARD SAVE AS FILE

MQGET with https://9.212.143.123:20668/oneWay



https://9.212.143.123:20668/oneWay

[✓ GET] POST PUT PATCH DELETE HEAD OPTIONS Other

Status: 200: OK Loading time:3

Response headers (5)

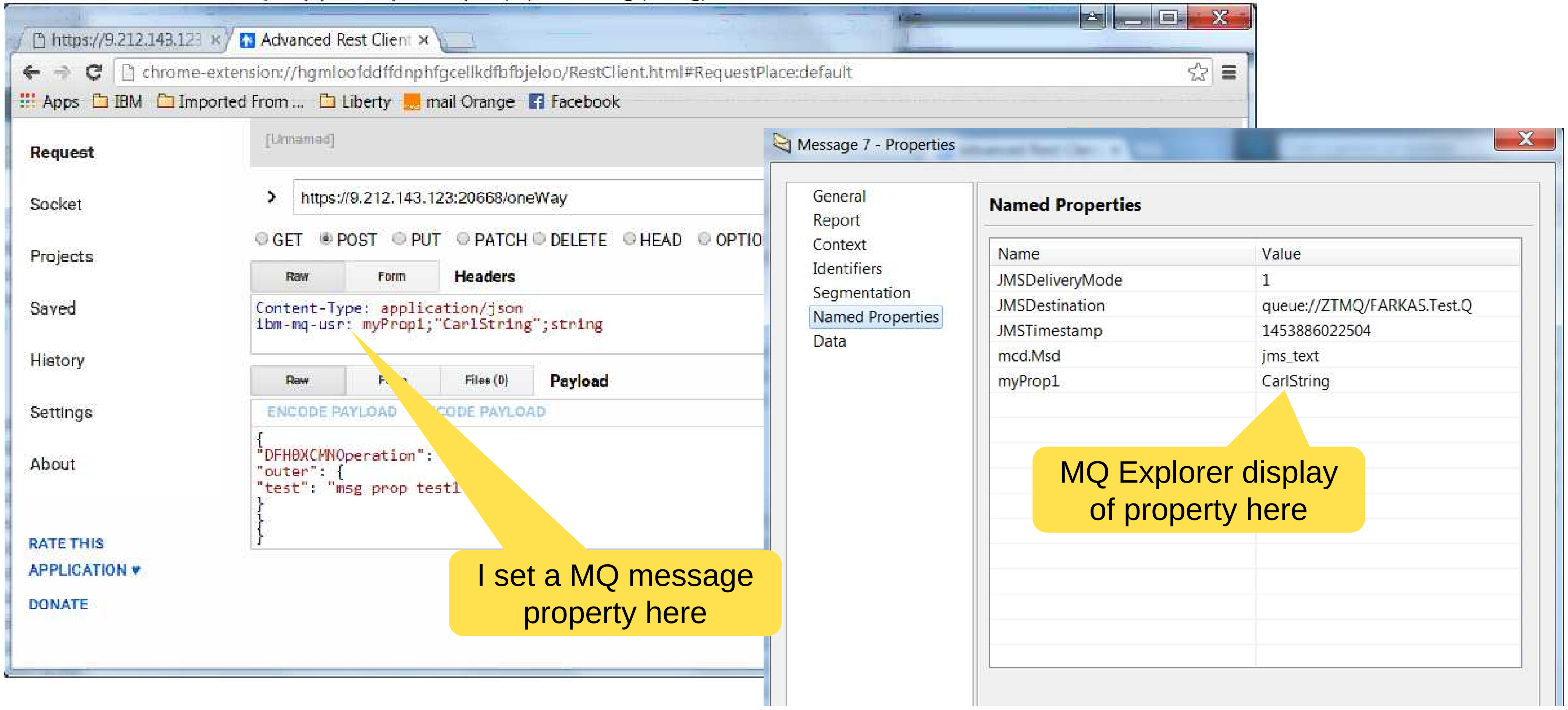
```
Content-Language: en-US
X-Powered-By: Servlet/3.0
Content-Type: application/json
Content-Length: 44
Date: Wed, 27 Jan 2016 08:55:30 GMT
```

Response

```
{
  - "DFH0XCMNOperation": {
      - "outer": {
            "test": "1"
        }
    }
}
```

Note the HTTP GET here

MQ SP property setting test



https://9.212.143.123:20668/oneWay

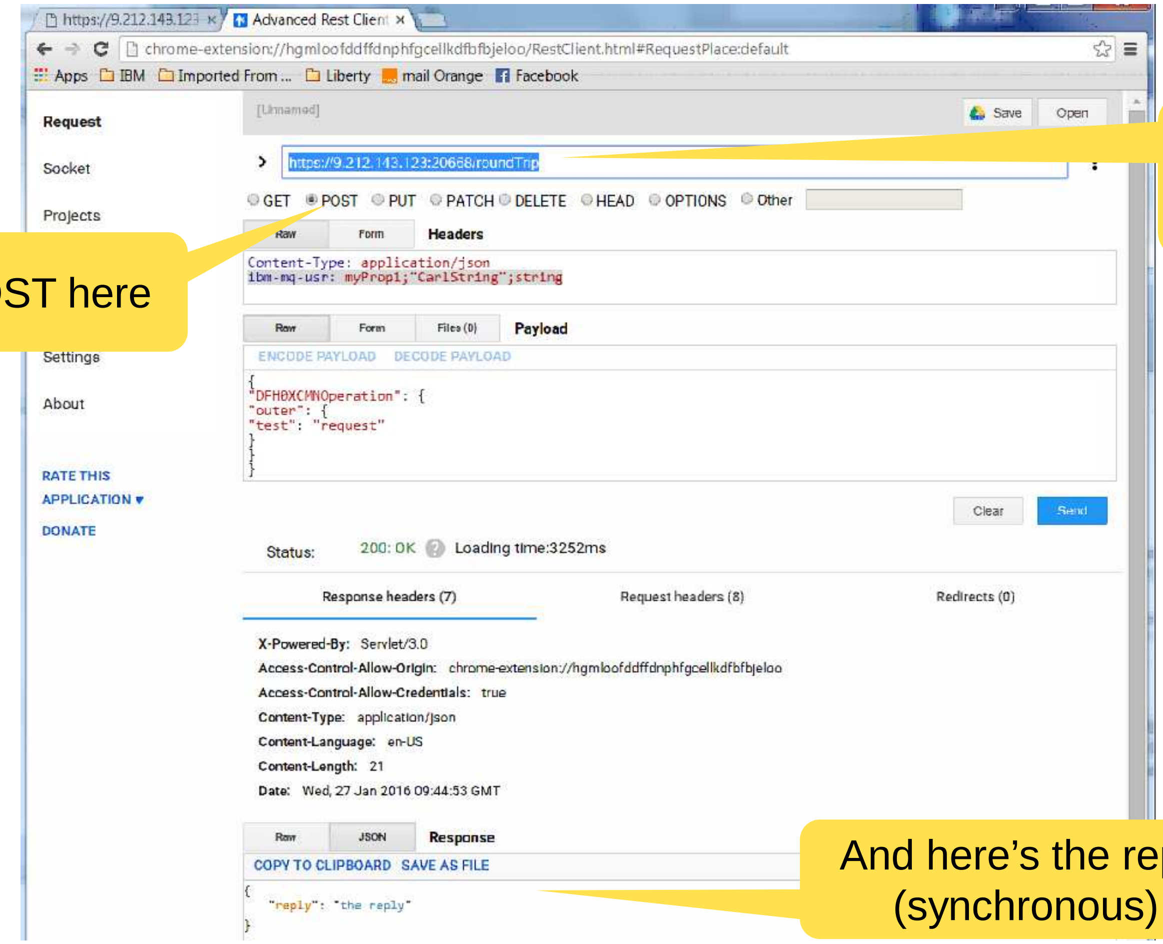
```
Content-Type: application/json
ibm-mq-usr: myProp1;"CarlString";string
```

```
{
"DFH0XCMNOperation":
"outer": {
"test": "msg prop test1
}
}
```

Named Properties

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

MQ SP round-trip



https://9.212.143.123:20668/roundTrip

GET [✓ POST] PUT PATCH DELETE HEAD OPTIONS Other

```
Content-Type: application/json
ibm-mq-usr: myProp1;"CarlString";string
```

```
{
"DFH0XCMNOperation": {
"outer": {
"test": "request"
}
}
}
```

Status: 200: OK Loading time:3252ms

```
X-Powered-By: Servlet/3.0
Access-Control-Allow-Origin: chrome-extension://hgmloofddffdnphfgcellkdfbfbjeloo
Access-Control-Allow-Credentials: true
Content-Type: application/json
Content-Language: en-US
Content-Length: 21
Date: Wed, 27 Jan 2016 09:44:53 GMT
```

```
{
  "reply": "the reply"
}
```

Different service (URL) here which has WAIT defined

Note the POST here

And here’s the reply (synchronous)