MQ Light, Bluemix, and MQ

WebSphere Integration User Group 2015

Matt Whitehead
MQ/MQ Light Dev Team

Please Note:

- IBM's statements regarding its plans, directions, and intent are subject to change or withdrawal without notice at IBM's sole discretion.

- Information regarding potential future products is intended to outline our general product direction and it should not be relied on in making a purchasing decision.

- The information mentioned regarding potential future products is not a commitment, promise, or legal obligation to deliver any material, code or functionality. Information about potential future products may not be incorporated into any contract.

- The development, release, and timing of any future features or functionality described for our products remains at our sole discretion.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput or performance that any user will experience will vary depending upon many factors, including considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve results similar to those stated here.

Agenda

- An overview of MQ Light 
- How MQ Light works in Bluemix 
- Running MQ Light applications in your MQ environment

IBM MQ

Agenda

- An overview of MQ Light 
- How MQ Light works in Bluemix
- Running MQ Light applications in your MQ environment

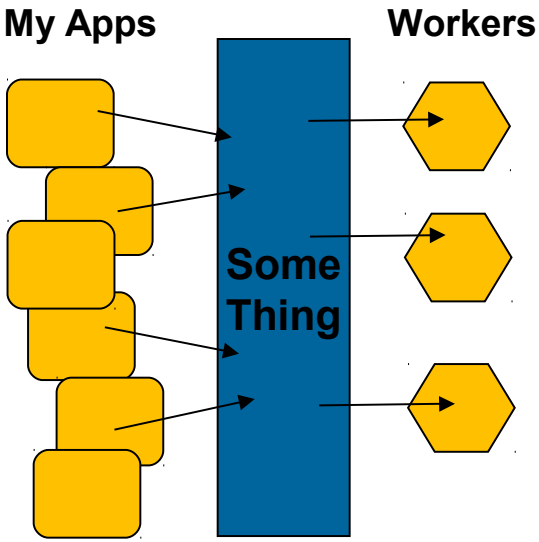
IBM MQ

The journey that got us here

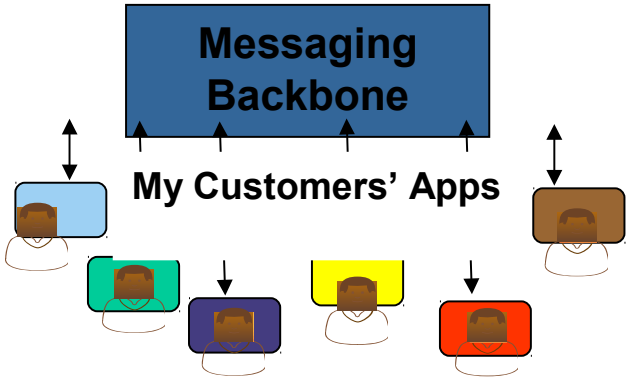
Andy
Developer

Iain
Infrastructure Guy

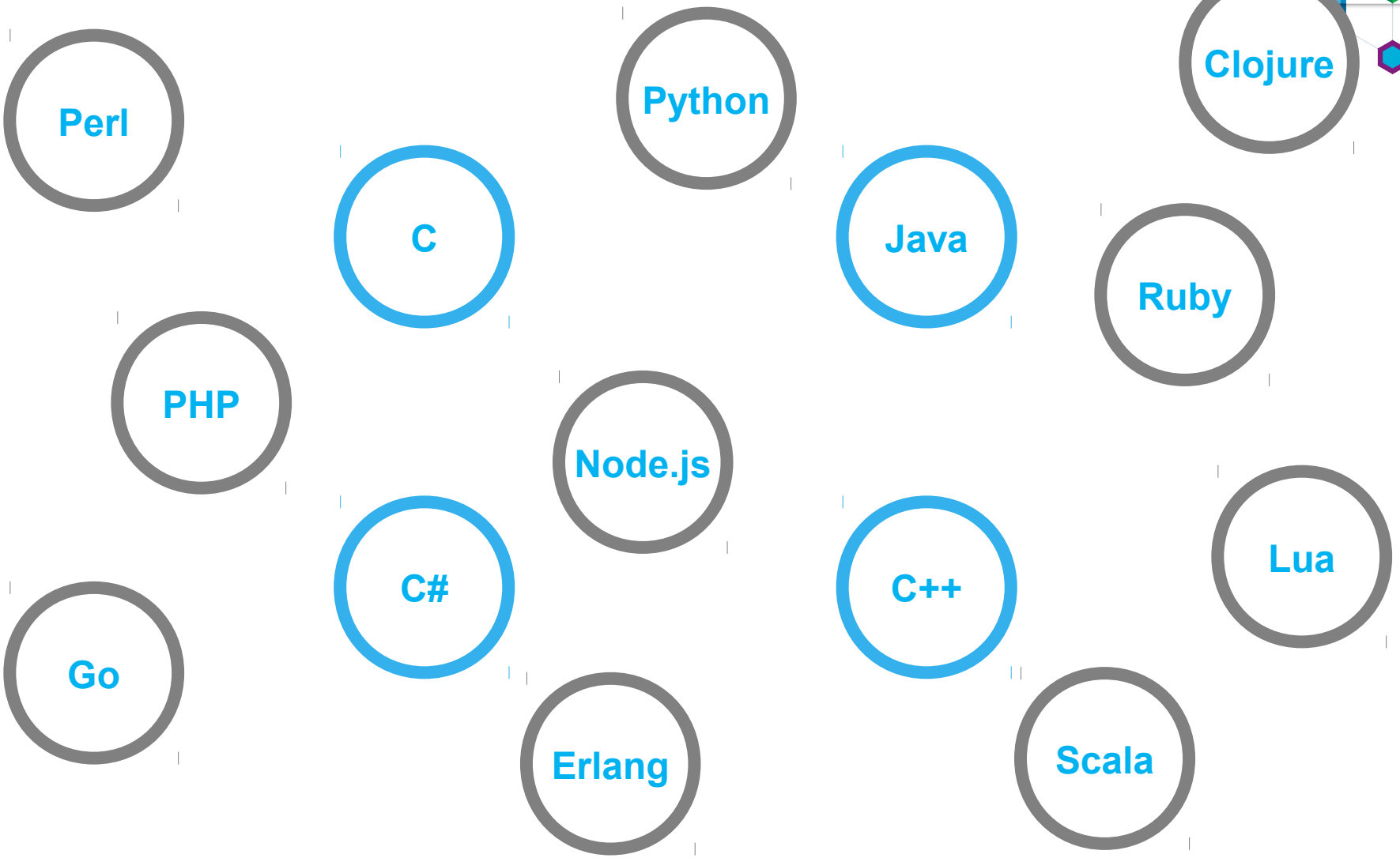
I want to execute code without taxing my Web app processes



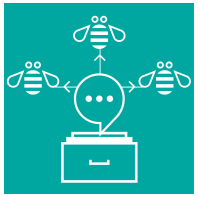
My job is to run a communications service for my customers' apps



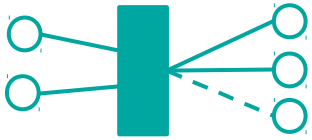
The journey that got us here…



Use cases

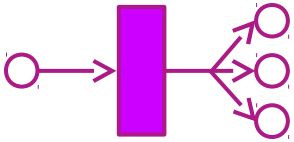


Worker offload



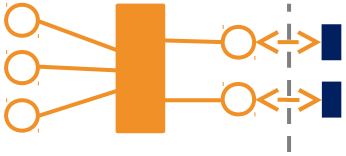
- Image processing
- Text analytics

Event driven



- Posting video to multiple social sites after transcoding
- Respond to external events

Connecting external systems



- Updating external booking app
- Posting updates to twitter

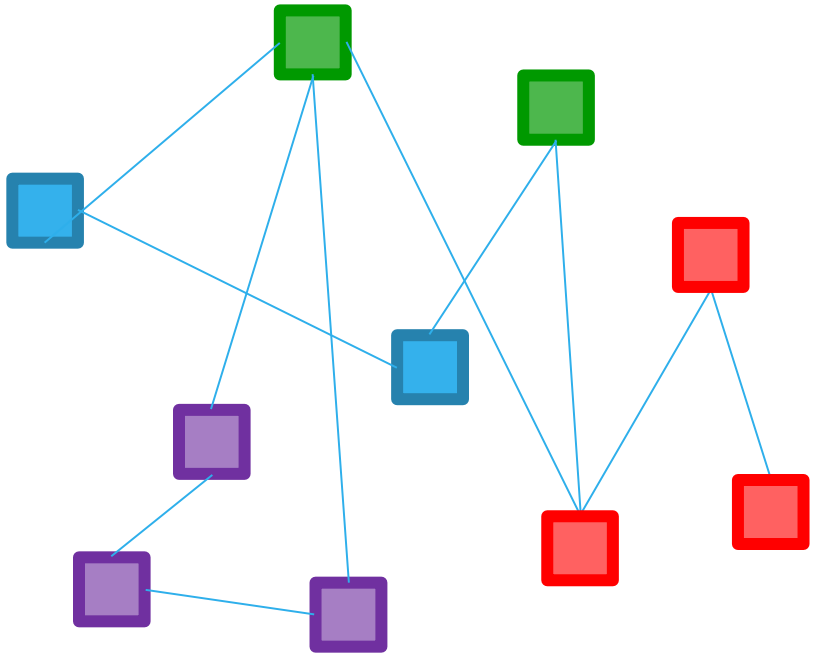
Trend in Cloud Application Architecture

Moving away from…
Large applications with single develop, test, deploy cycles using single technology stacks

Large monolithic Applications

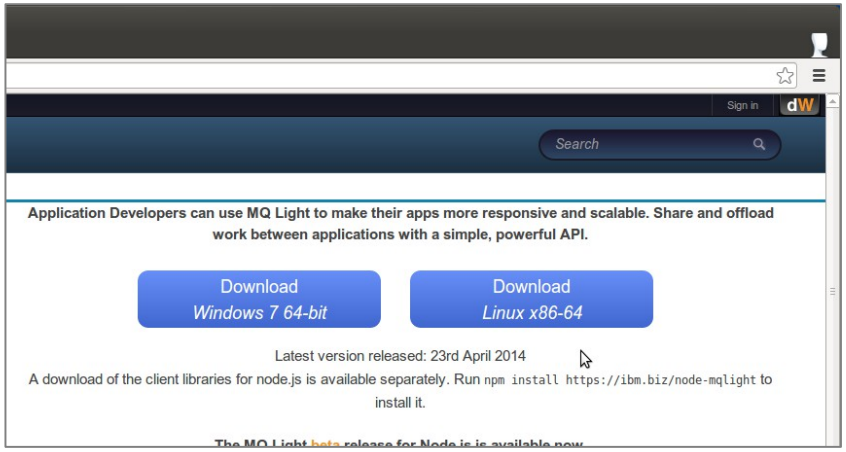
...towards…
Delivering global scale applications with unprecedented agility by de-composing apps into a microservices architecture

(2-pizza sized teams)



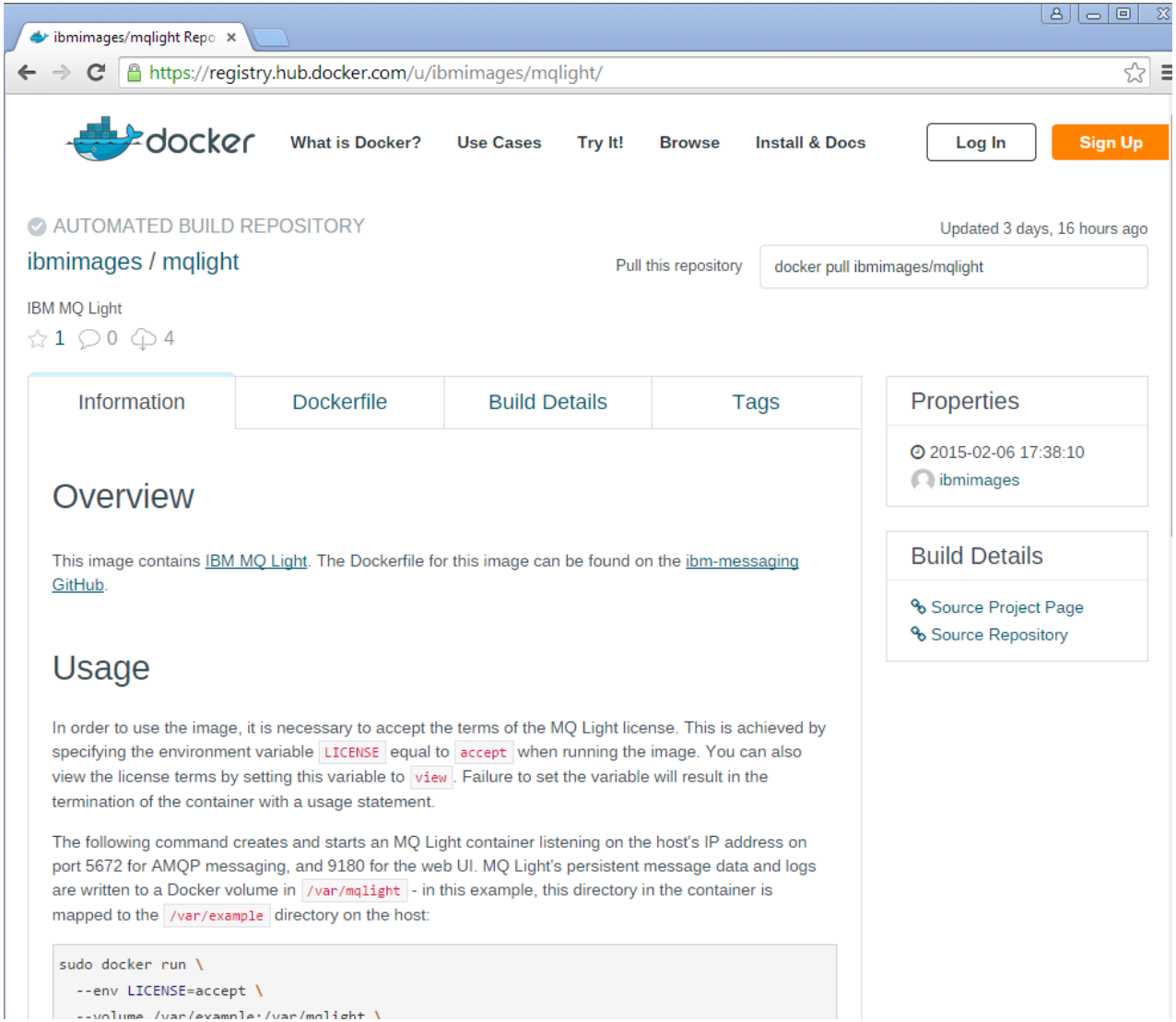
First Five Minute Experience

- Download and get coding within 5 minutes
  - Linux-x86-64, Windows7 64 bit, Mac OSX
  - Unzip install 
  - Unlimited time developer license (unsupported).



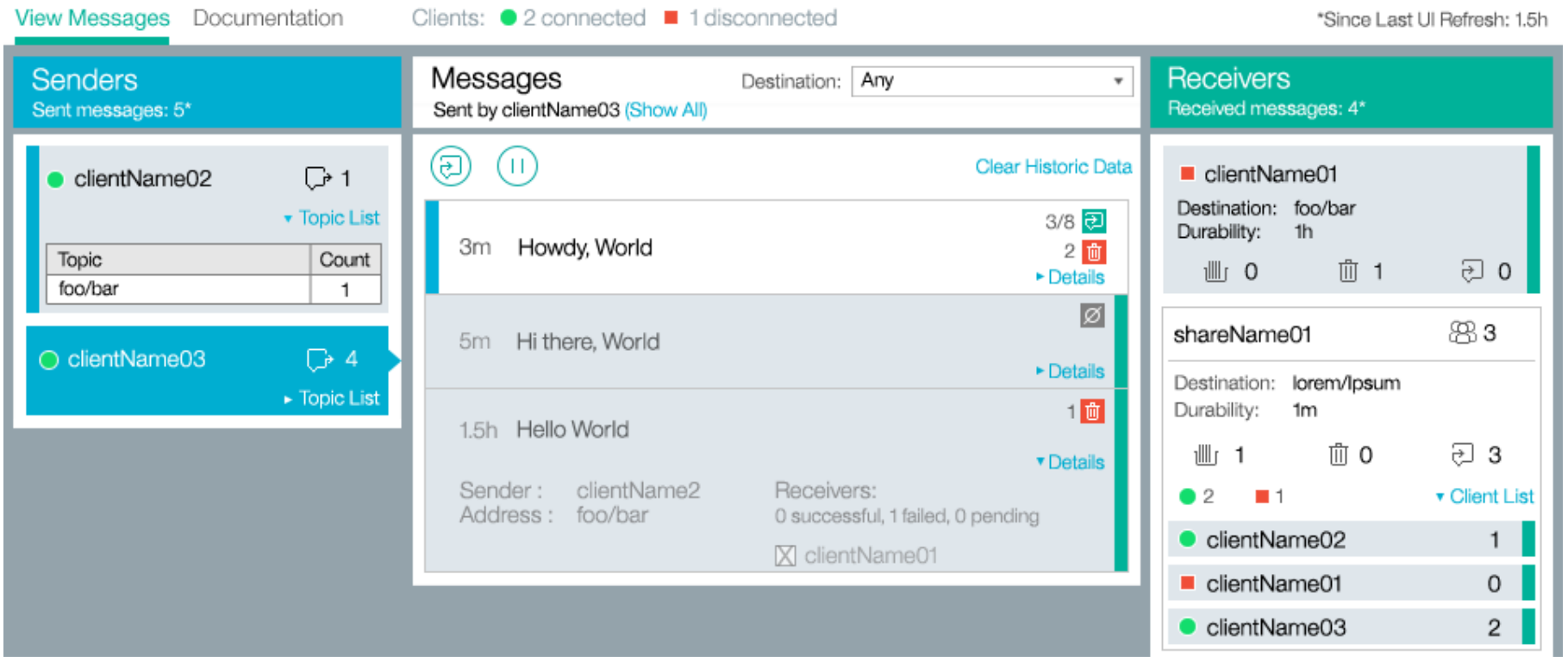
- No administration; just code and go
- API client libraries installed using language package manager
- Tutorials and examples in their languages, relevant to actual use

MQ Light Image DockerHub Registry

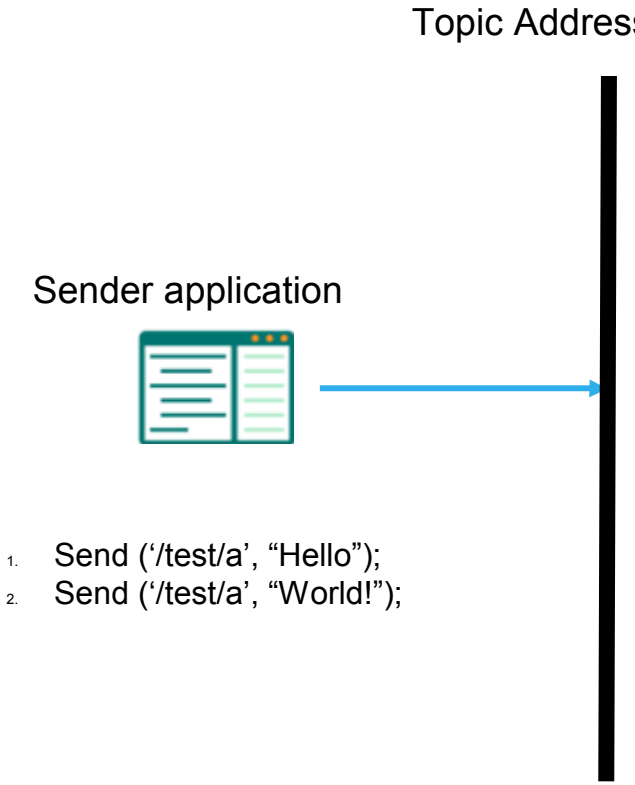


Tooling to assist development

- Used to answer questions like :
  - Where did this message come from?
  - Where did that message go?

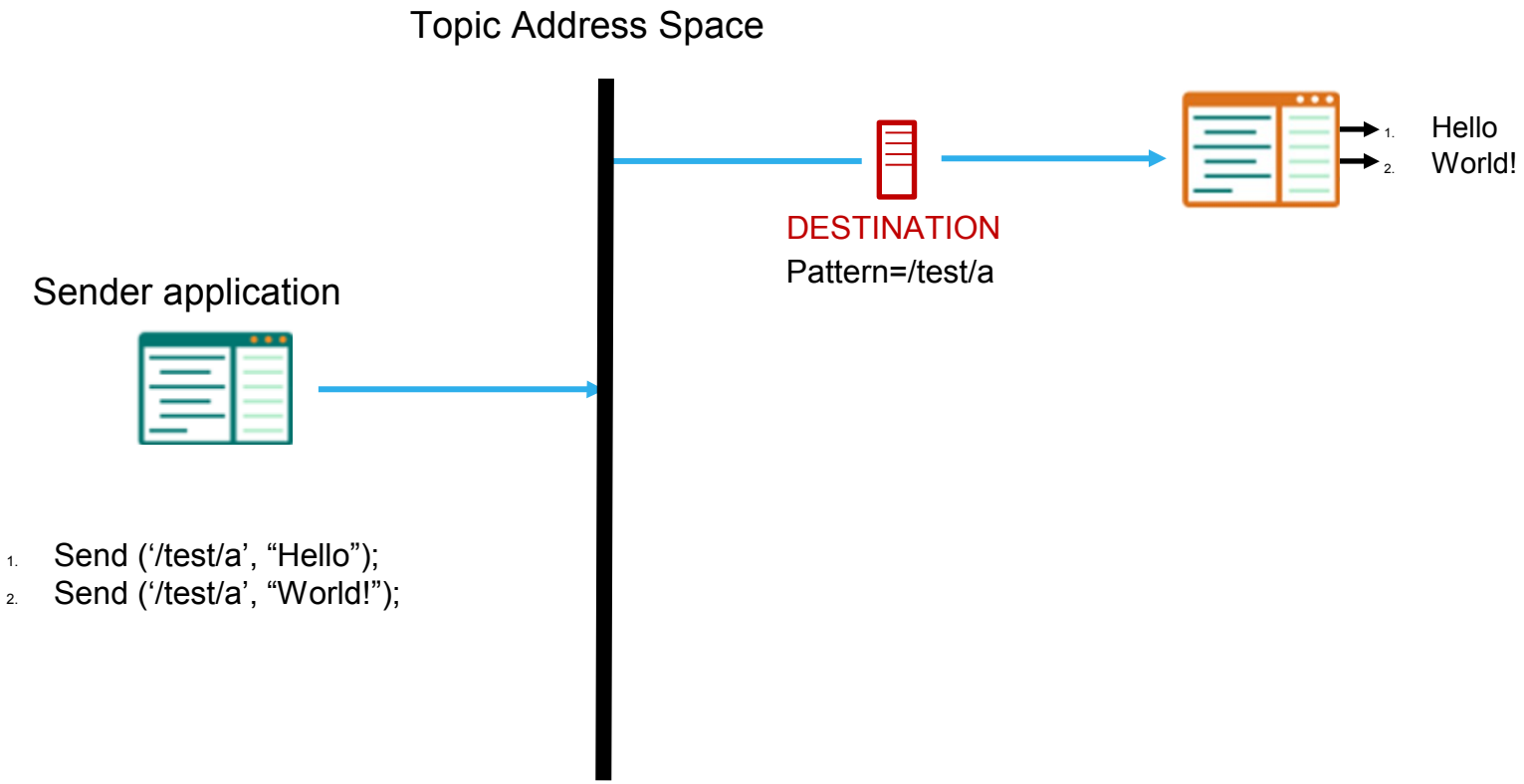


MQ Light Messaging Model – Send Messages



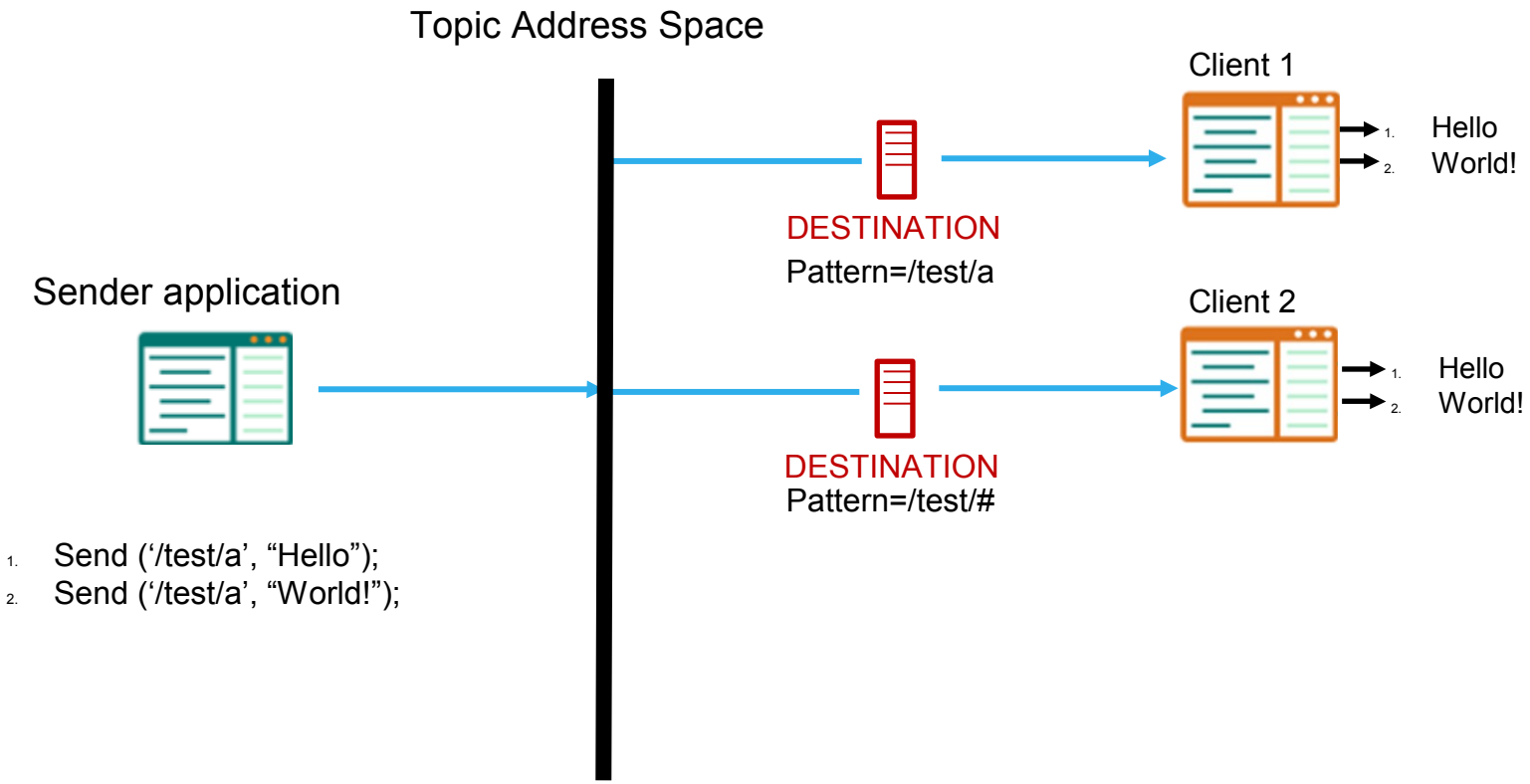
Applications send messages to a ***topic***.
A topic is an address in the topic space
either flat or arranged hierarchically.

MQ Light Messaging Model – Simple Receive



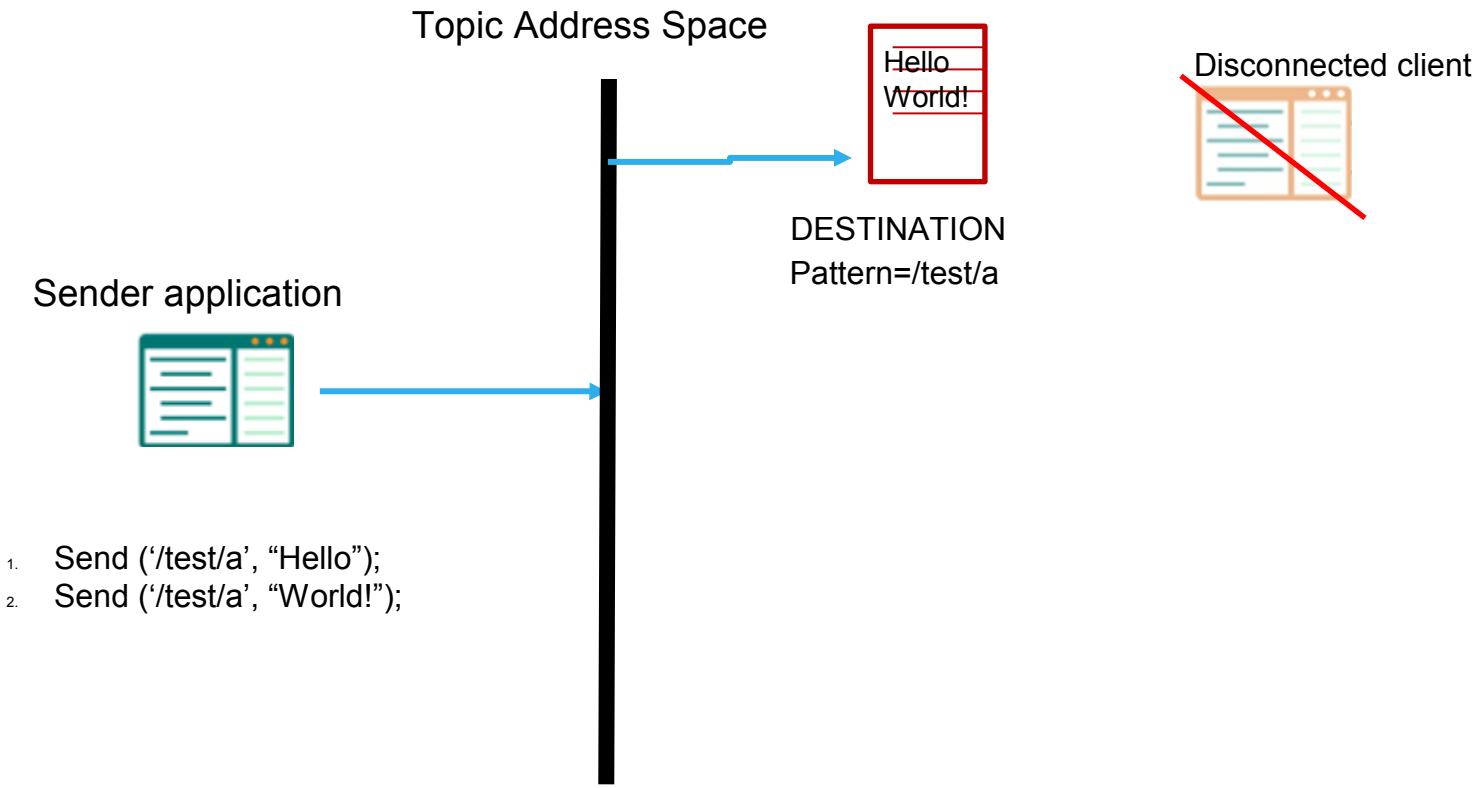
- Applications receive messages by creating a **destination** with a pattern which matches the topics they are interested in.
- Pattern matching scheme based on WMQ.

MQ Light Messaging Model – Pub/Sub



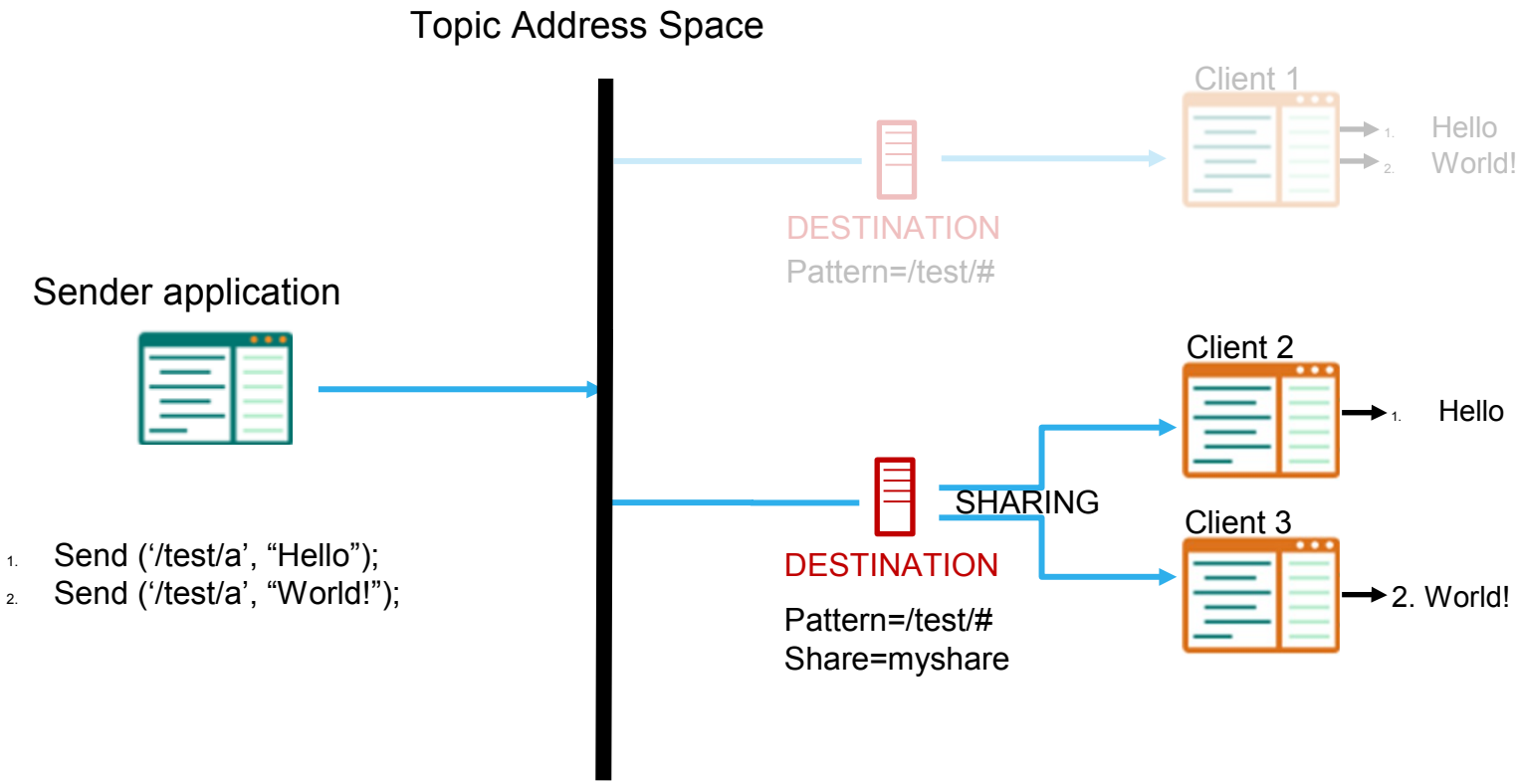
- Multiple destinations can be created which match the same topic
  - Pub/Sub style.

MQ Light Messaging Model – Persistent destinations



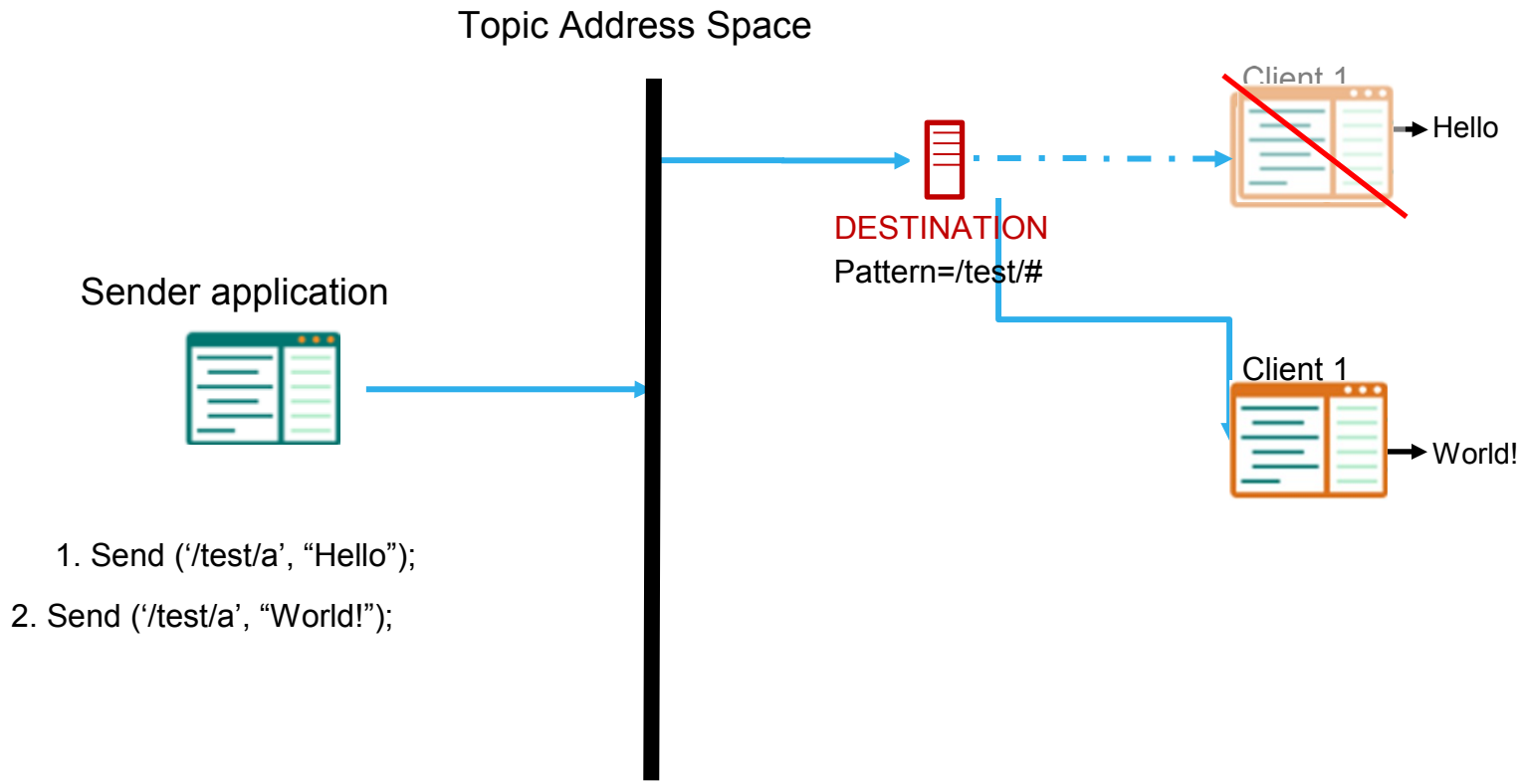
- Destinations persist for a defined "time to live" after receiver detaches.

MQ Light Messaging Model – Sharing



- Clients attaching to the same topic pattern and share name attach to the same shared destination.

MQ Light Messaging Model – Client takeover



- Applications connect to MQ Light service specify (optional) client ID.
- Re-using the same client ID pre-empts the original connection.
  - Ideal for worker takeover in the cloud.

MQ Light Messaging Model

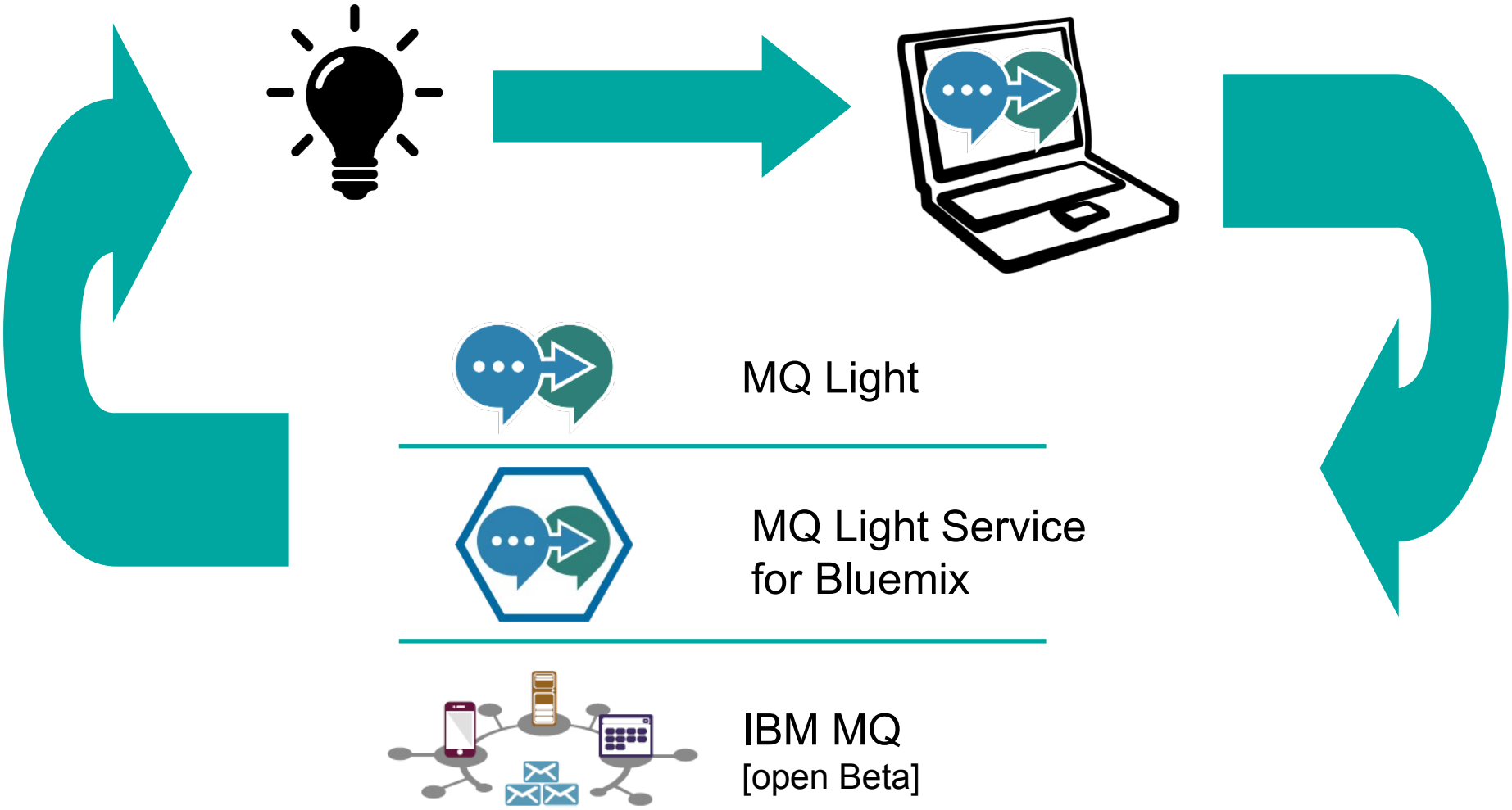
- Messages
  - Payload is either Text or Binary.
  - Content-type is used by clients to transfer JSON
  - Per message time to live.

- Message delivery model
  - At most once delivery (QoS 0)
  - At least once delivery (QoS 1)
  - Acknowledge & Reject messages
  - Control over the number of unacknowledged messages delivered

Programming languages

Language	Availability	Notes
Node.js	Supported IBM Client.	
Ruby	IBM Client in Beta	
Java (Non Blocking)	IBM Client in Beta	
Python	IBM Client in Beta	
PHP	Qpid Proton	
Scala	Can use Java non Blocking client.	Forum users report this working.
C/C++	Qpid Proton	
Perl	Qpid Proton	

Deployment Options



Agenda

- An overview of MQ Light

- How MQ Light works in Bluemix

- Running MQ Light applications in your MQ environment

IBM MQ

IBM Bluemix

Bluemix is an **open-standards**, cloud-based platform for **building**, **running, and managing applications.**

Build your apps, your way

Use the most prominent compute technologies to power your app: Cloud Foundry, Docker, OpenStack.

Scale more than just instances

Development, monitoring, deployment, and logging tools allow the developer to run and manage the entire application.

Extend apps with services

A catalog of IBM, third party, and open source services allow the developer to stitch an application together quickly.

Deploy and manage hybrid apps seamlessly

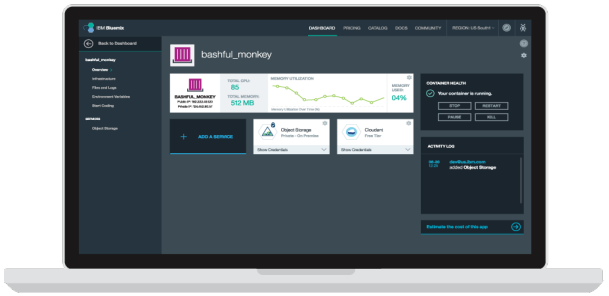
Get a seamless dev and management experience across a number of hybrid implementations options.

Layered Security

IBM secures the platform and infrastructure and provides you with the tools to secure your apps.

Flexible Pricing

Try compute options and services for free and, when you're ready, pay only for what you use. Pay as you go and subscription models offer choice and flexibility.

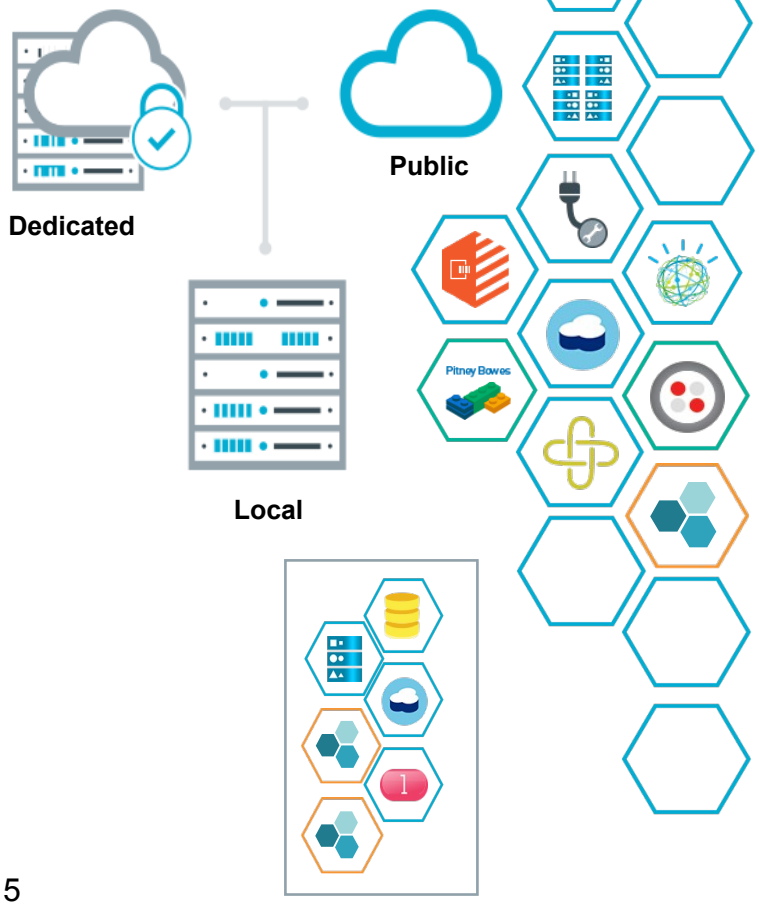


Dedicated to you

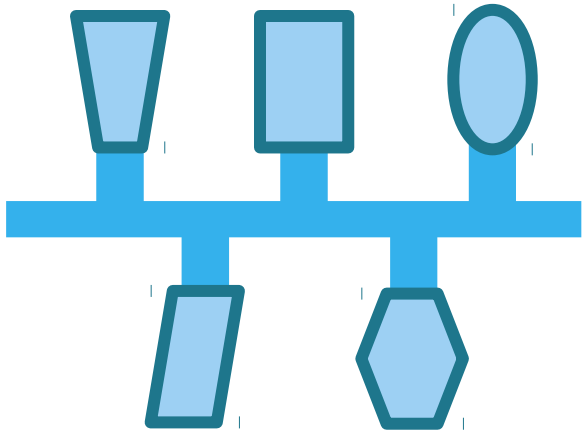
Single tenant hardware that's completely dedicated to you – allowing you to satisfy regulatory & legal compliance.

What it means

- The Bluemix platform and dedicated runtimes and services sit on SoftLayer hardware that is dedicated to you
- You still have the ability to connect to all multi-tenant services in the "public" catalog
- Integrated to your LDAP for developer authentication
- Elastic capacity based on your demands.

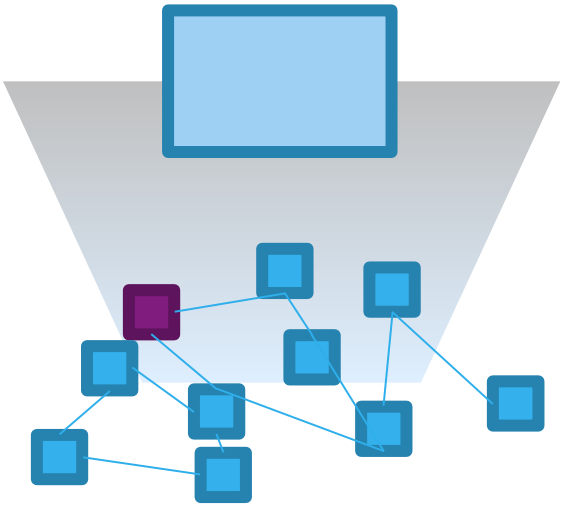


Different messaging requirements



Messaging for enterprise central IT

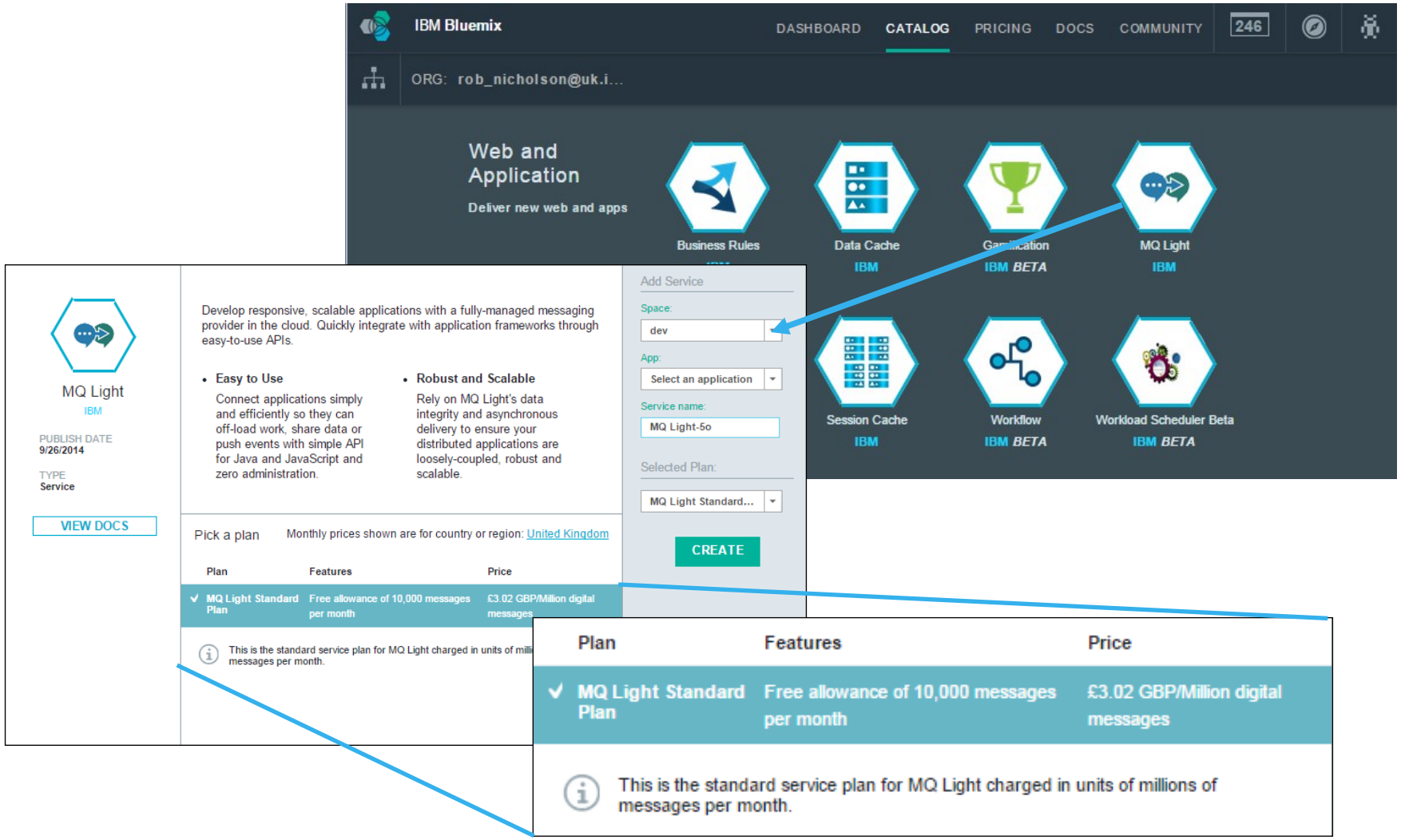
- Support for enterprise language and frameworks
- XA 2 phase transactions and bulletproof QoS options eliminates app compensation logic.
- Translation between different platform types
- Admin policy controlled behaviour
- Comprehensive configuration options for fine grained control



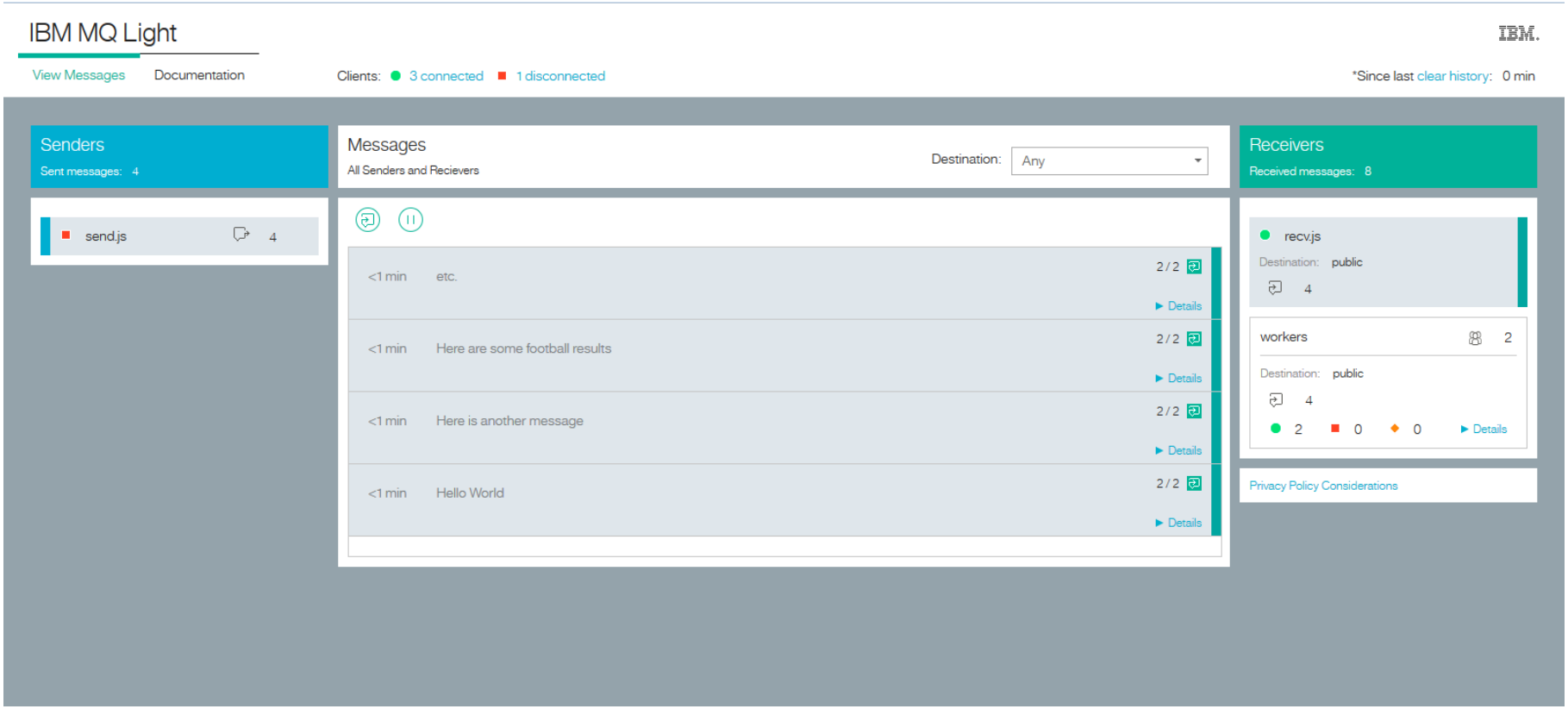
Messaging for microservice devs

- Reactive scaling
- Open and standard protocols
- Completely self service
- Application controlled behaviour
- Ability to quickly unit-test microservices
- Minimal concepts for “full stack architects”

Introduction to MQ Light Service



The Bluemix UI is identical to the standalone MQ Light development UI...



MQ Light Bluemix - Language support

In Bluemix the MQ Light service supports all of the MQ Light clients...

Language	Availability
Node.js	Supported IBM Client.
Ruby	IBM Client in Beta
Java (Non Blocking)	IBM Client in Beta
Python	IBM Client in Beta
PHP	Qpid Proton
Scala	Can use Java non Blocking client.
C/C++	Qpid Proton
Perl	Qpid Proton

...plus JMS 1.1

Pub/Sub & queueing, local transactions only, JSE or JEE (MDBs)

Agenda

- An overview of MQ Light

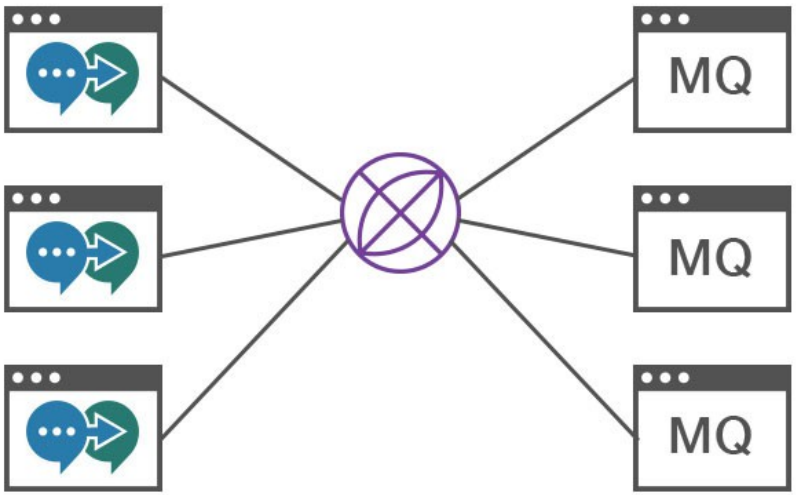
- How MQ Light works in Bluemix

- Running MQ Light applications in your MQ environment

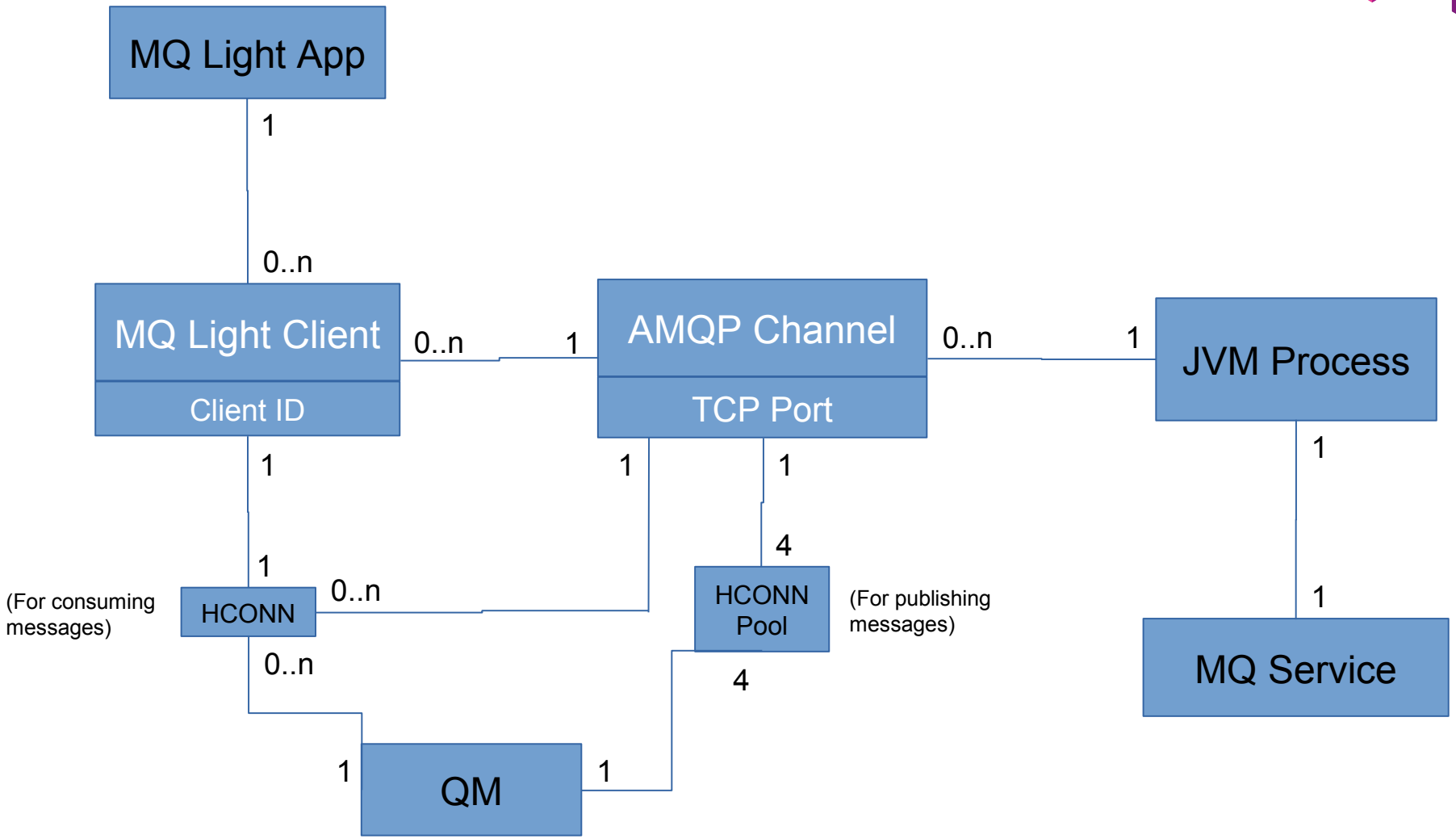
IBM MQ

New AMQP channel type

- Adds a channel type of “AMQP”
- Support a subset of the AMQP 1.0 Oasis specification
- Interoperable with MQ FAP and MQTT applications (see later slides for details)



The architecture of an MQ Light channel



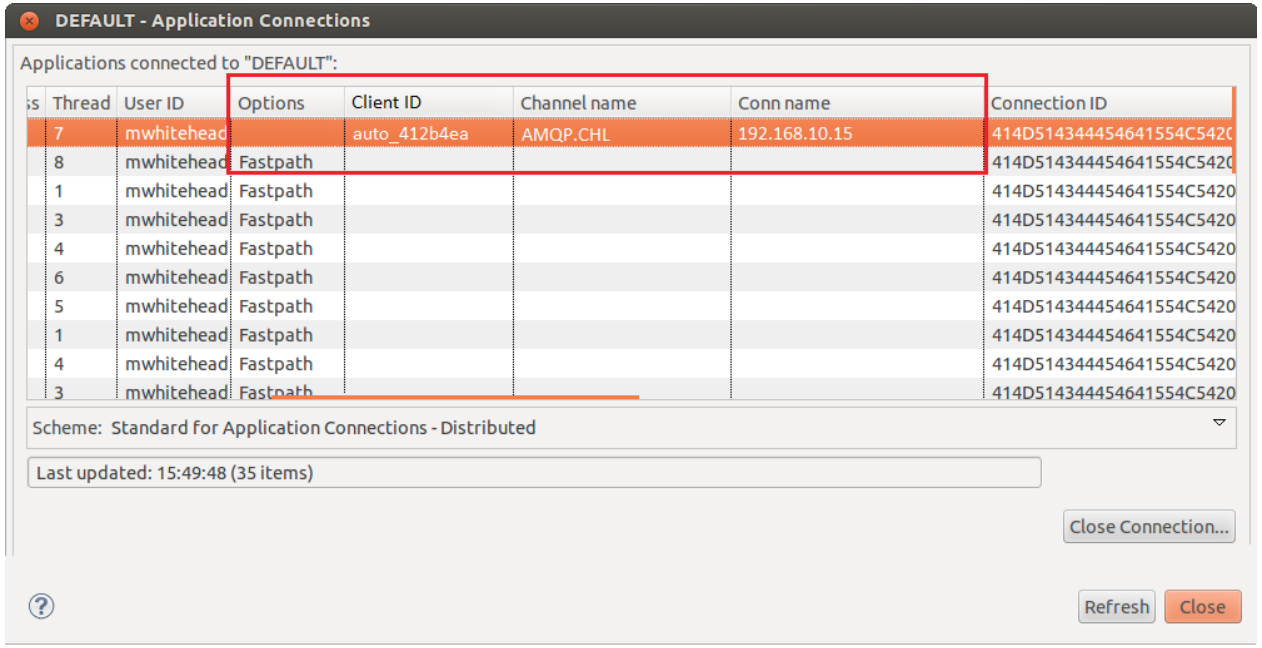
AMQP Channels

- Configuration model
  - MQSC and PCF updates allow you to administer AMQP channels in much the same way as other MQ objects

    - DISPLAY CHANNEL(*) CHLTYPE(AMQP)
    - DEFINE CHANNEL(MY.AMQP.CHANNEL) CHLTYPE(AMQP) PORT(5673)
    - START CHANNEL(MY.AMQP.CHANNEL)
    - STOP CHANNEL(MY.AMQP.CHANNEL)
    - DISPLAY CHSTATUS(*) CHLTYPE(AMQP)

Administration

- Explorer
  - Not available in the current version of the beta. Some ideas about how it could look...



- Connection name and channel name populated to show where the client has connected from
- New **Client ID** attribute in the Application Connections view

Monitoring

- Events
  - MQ has support for events for monitoring different activities
  - Some are already available to try in the beta
    - Command events (e.g. request to start a channel)
    - Configuration events (e.g. request to change channel attrs)
  - Some are ones we'd like to do
    - Security events (e.g. an AMQP client failed an authority check)

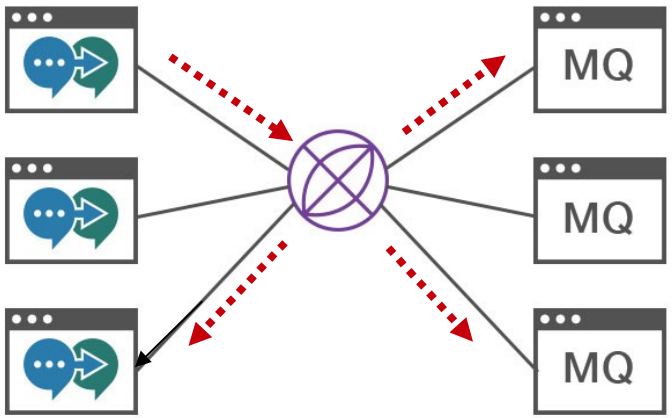
Interoperability

- AMQP to MQ FAP and MQTT

warehouse/item/372837

warehouse/item/#

Published AMQP messages MQPUT to an MQ topic



MQ apps can consume AMQP publishes by subscribing to matching topic pattern

Consumed AMQP messages MQGET from an MQSUB

MQ apps can publish to AMQP clients by MQPUT to the same topic string

orders/electrical/#

orders/electrical/wiring

Detail on Interoperability – **AMQP** to **MQ**

Some AMQP **headers** are set as MQMD fields:

AMQP header.ttl	set on MQ message as MQMD.expiry (converted to 10ths of a second)
AMQP header.priority	set on MQ message as MQMD.priority (max value of 9)
AMQP properties.correlation-id	set on MQ message as MQMD.correlid

All AMQP **headers** are set as MQ message properties with a mapped name:

header.durable	set on MQ message as MQ property	AMQPDurable
header.priority	set on MQ message as MQ property	AMQPPriority
header.ttl	set on MQ message as MQ property	AMQPTtl
header.first-acquirer	set on MQ message as MQ property	AMQPFirstAcquirer
header.delivery-count	set on MQ message as MQ property	AMQPDeliveryCount

All AMQP **properties** are also set as MQ message properties, e.g.

properties.user-id	set on MQ message as MQ property	AMQPUserId
properties.to	set on MQ message as MQ property	AMQPTo
properties.subject	set on MQ message as MQ property	AMQPSubject
properties.reply-to	set on MQ message as MQ property	AMQPReplyTo
properties.content-type	set on MQ message as MQ property	AMQPContentType
properties.content-encoding	set on MQ message as MQ property	AMQPContentEncoding
properties.creation-time	set on MQ message as MQ property	AMQPCreationTime
properties.group-id	set on MQ message as MQ property	AMQPGroupId
properties.message-id	set on MQ message as MQ property	AMQPMessageId
Properties.group-sequence	set on MQ message as MQ property	AMQPGroupSequence
Properties.absolute-expiry-time	set on MQ message as MQ property	AMQPAbsoluteExpiryTime
Properties.reply-to-group-id	set on MQ message as MQ property	AMQPReplyToGroupId

Finally, all AMQP **application-properties** are copied into the MQ message properties in the user space (usr.*) using similar naming conventions with some restrictions on the property length, characters used, and certain keyword restrictions e.g. “JMS”.

Detail on Interoperability – **MQ** to **AMQP**

The following **MQMD fields** are set on the AMQP message as headers, **if and only if** the value in the MQ message is not the same as the AMQP default value for that property.

MQMD.persistence	set on AMQP message as header.durable
MQMD.expiry	set on AMQP message as header.ttl
MQMD.priority	set on AMQP message as header.priority

Some **MQ message properties**, if they exist, are set as AMQP headers:

MQ message property AMQPFirstAcquirer	set on AMQP message as header.first-acquirer
MQ message property AMQPDeliveryCount	set on AMQP message as header.delivery-count

Some **MQ message properties**are set as AMQP properties:

AMQPUserId	set on the AMQP message as properties.user-id
AMQPTo	set on the AMQP message as properties.to
AMQPSubject	set on the AMQP message as properties.subject
AMQPReplyTo	set on the AMQP message as properties.reply-to
AMQPContentType	set on the AMQP message as properties.content-type
AMQPContentEncoding	set on the AMQP message as properties.content-encoding
AMQPCreationTime	set on the AMQP message as properties.creation-time
AMQPGroupId	set on the AMQP message as properties.group-id
AMQPMessageId	set on the AMQP message as properties.message-id
AMQPGroupSequence	set on the AMQP message as properties.group-sequence
AMQPAbsoluteExpiryTime	set on the AMQP message as properties.absolute-expiry-time
AMQPReplyToGroupId	set on the AMQP message as properties.reply-to-group-id

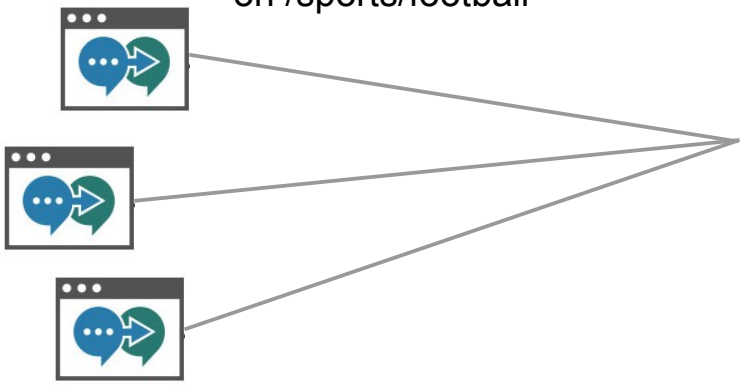
Finally, all **MQ message properties in the user space** (i.e. those which start usr.*) are copied into the AMQP message as application properties.

Development Backlog

- These features aren't yet available in the beta, but they're on our backlog
  - MQ Explorer support
  - Parity with MQ Distributed Platform coverage
  - Channel Authentication Rules
  - Connection Authentication Rules
  - Enhanced Monitoring
  - TLS Administration using MQ Channel definition
  - KnowledgeCenterdocumentation
  - Administering shared destinations
  - Segregating AMQP/MQ Light messages from other MQ apps

Managing MQ Light Applications in a MQ Environment

(Not available in the beta yet – another feature on our backlog – give us feedback)

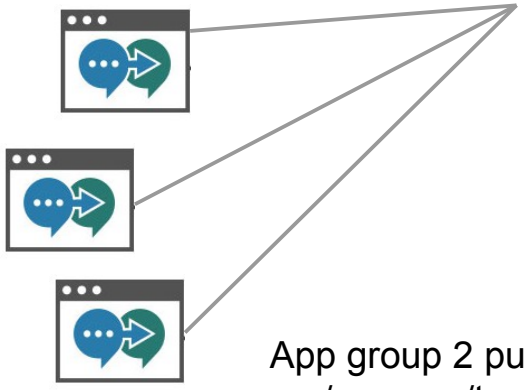


App group 1 publishing
on /sports/football

Queue manager

```
CHANNEL(AMQP.CHL.1)      CHLTYPE(AMQP)
PORT(5672)               TOPROOT(GROUP1.TOPIC)
...
```

If GROUP1.TOPIC has the topic string /groups/1, messages end up on /groups/1/sports/football



```
CHANNEL(AMQP.CHL.2)      CHLTYPE(AMQP)
PORT(5673)               TOPROOT(GROUP2.TOPIC)
...
```

If GROUP2.TOPIC has the topic string /groups/2, messages end up on /groups/2/sensors/temp

App group 2 publishing
on /sensors/temp

Thank You

Questions?

MQ Light website

developer.ibm.com/messaging/mq-light

IBM Bluemix website

console.ng.bluemix.net

MQ beta for MQ Light (AMQP channel)

www.ibm.com/developerworks/community/blogs/messaging

Watch the demo - Google “MQ Light Walk Through You Tube”