

Connecting MQ to the World: New Bridges

Mark Taylor
marke_taylor@uk.ibm.com
IBM Hursley

Introduction

- MQ has always included links to other application technologies
 - Not just a set of APIs
 - Not always devolved to MessageBroker/IIB
- For example
 - CICS, IMS Bridges
 - SAP R/3 Link (obsolete)
 - Resource Adapter for JEE
 - AMQP
- Now has product-provided bridges to two new environments:
 - **Salesforce**
 - **Blockchain**
- And open-source connectors for **Kafka**



Objectives

- Provide simple way for MQ apps to integrate with these new systems
- Make use of existing MQ infrastructure
- Build on MQ skills
- Complement capabilities in other products such as IIB

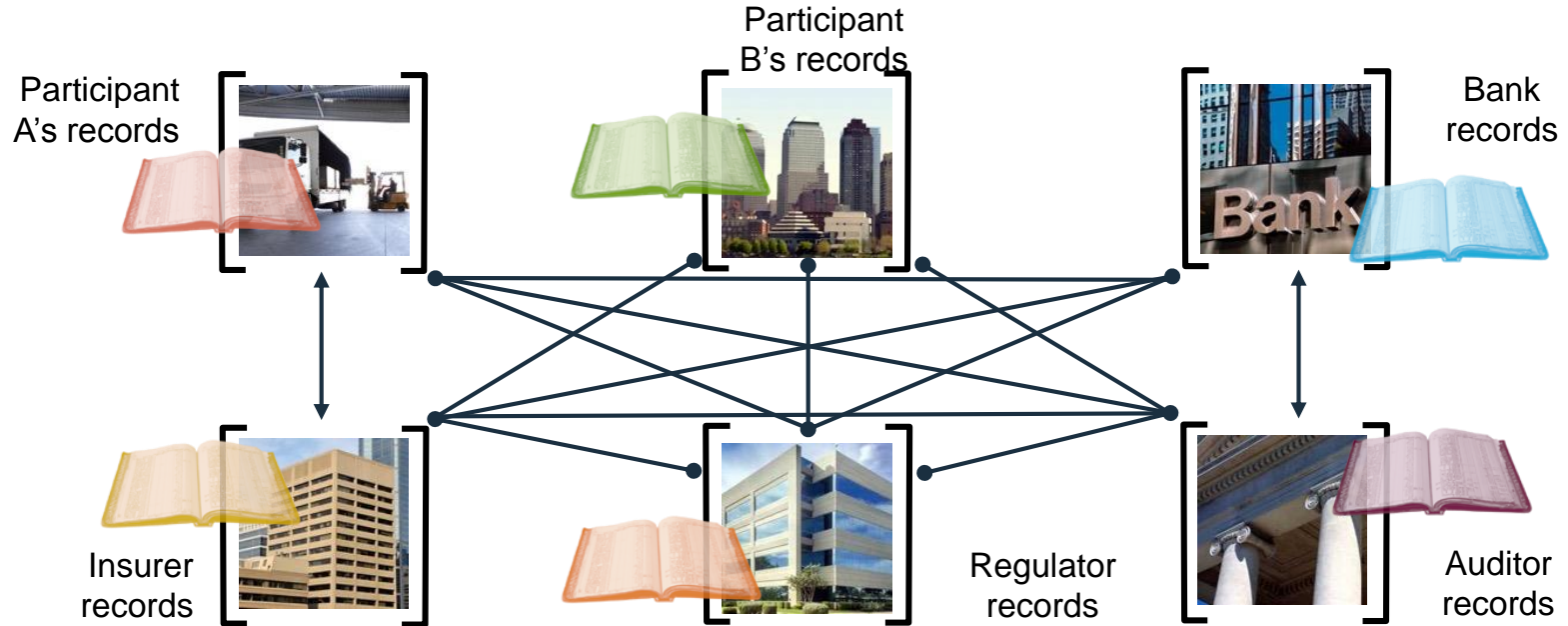
A decorative graphic consisting of several overlapping, wavy, horizontal bands of blue and light blue, creating a sense of motion and depth, spans the width of the slide above the main title.

MQ BRIDGE FOR BLOCKCHAIN

A decorative graphic consisting of several overlapping, wavy, horizontal bands of blue and light blue, creating a sense of motion and depth, spans the width of the slide above the title.

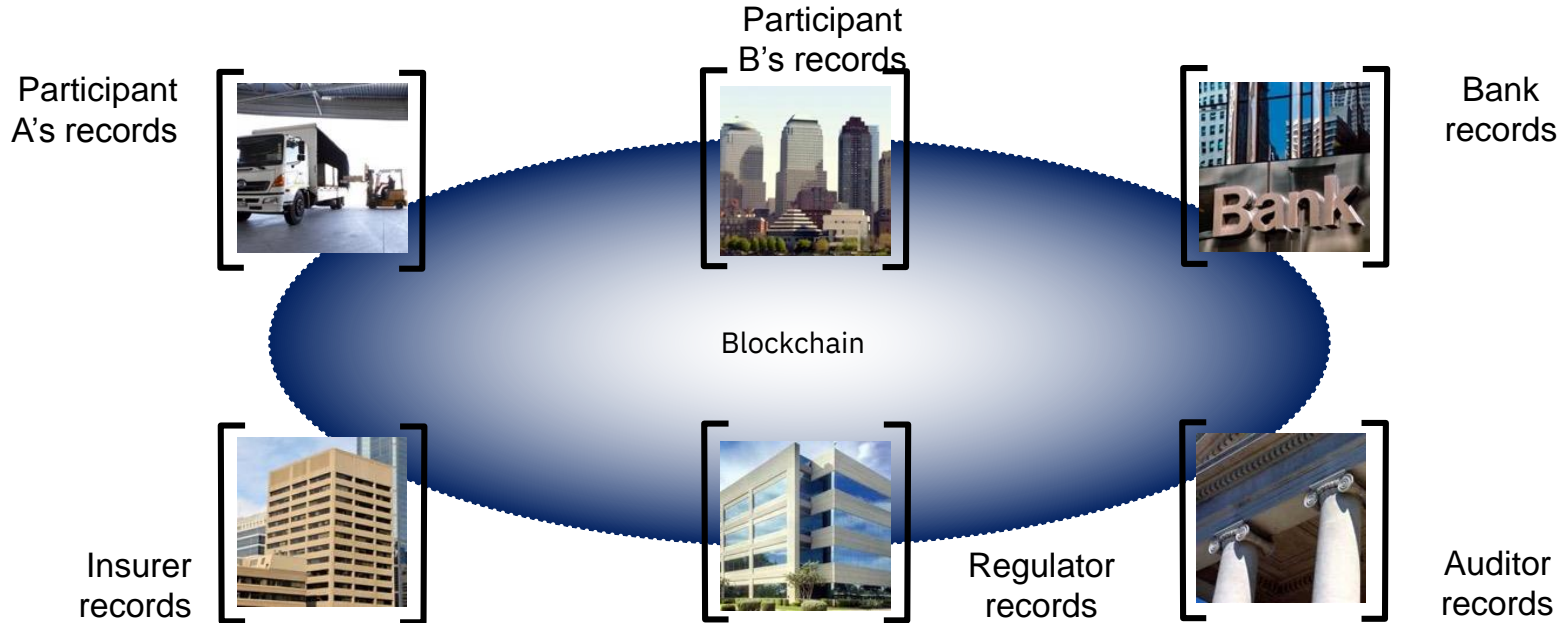
WHAT IS BLOCKCHAIN

Problem ...



... inefficient, expensive, vulnerable

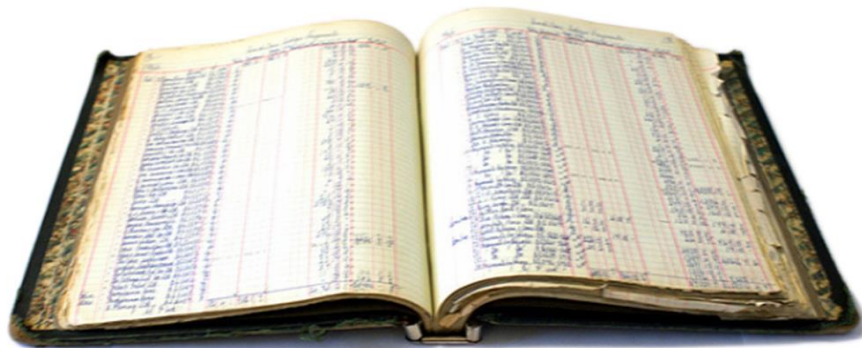
A shared, replicated, permissioned ledger ...



... with consensus, provenance, immutability and finality

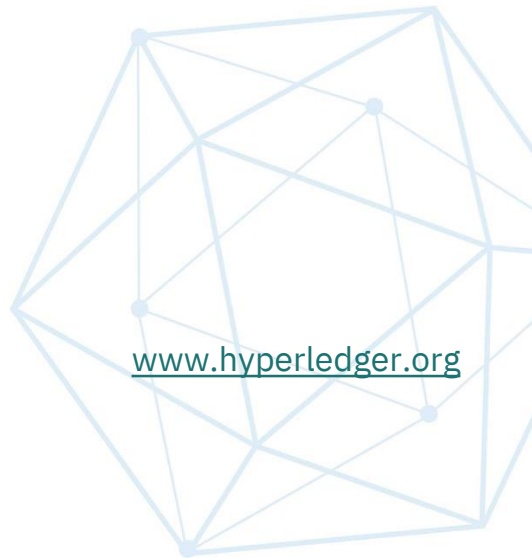
Shared ledger

- Shared between participants
- Participants have own copy through replication
- Permissioned, so participants see only appropriate transactions
- THE shared system of record

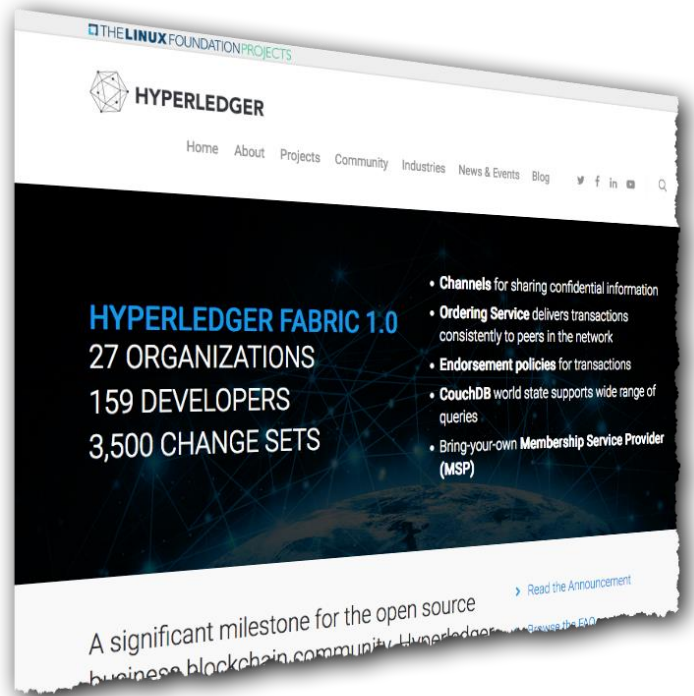


Hyperledger: A Linux Foundation project

- A collaborative effort created to advance cross-industry blockchain technologies for business
- Announced December 2015, now around 150 members
- Open source, open standards, open governance
- Five frameworks and three tools projects
- IBM is a premier member of Hyperledger



Hyperledger Fabric: Distributed ledger platform



- An implementation of blockchain technology that is a foundation for developing blockchain applications
- Emphasis on ledger, smart contracts, consensus, confidentiality, resiliency and scalability.
- V1.0 released July 2017
 - 159 developers from 27 organizations
 - IBM is one contributor of code, IP and development effort to Hyperledger Fabric

<http://hyperledger-fabric.readthedocs.io/>

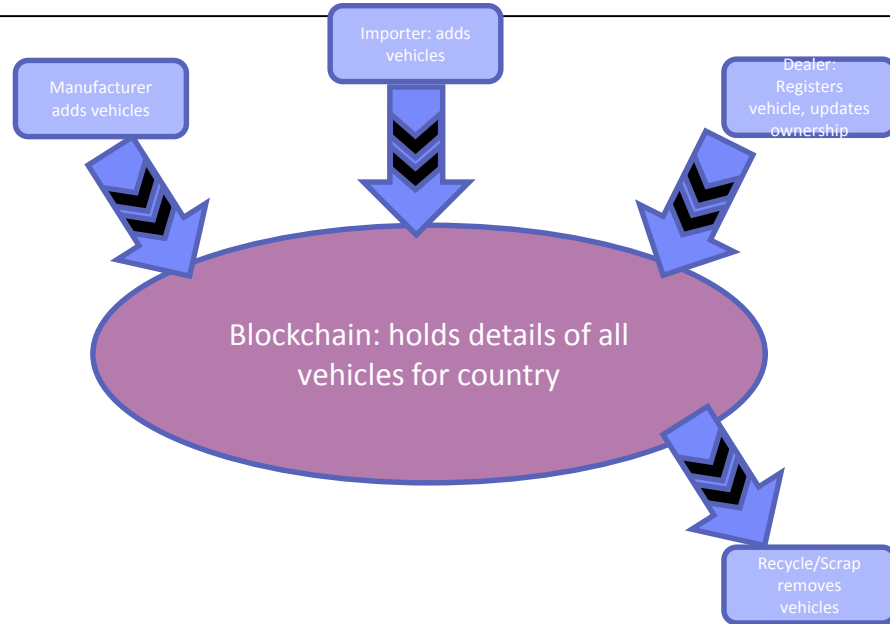
A decorative graphic consisting of several overlapping, wavy blue lines that sweep across the upper half of the slide, creating a sense of motion and depth.

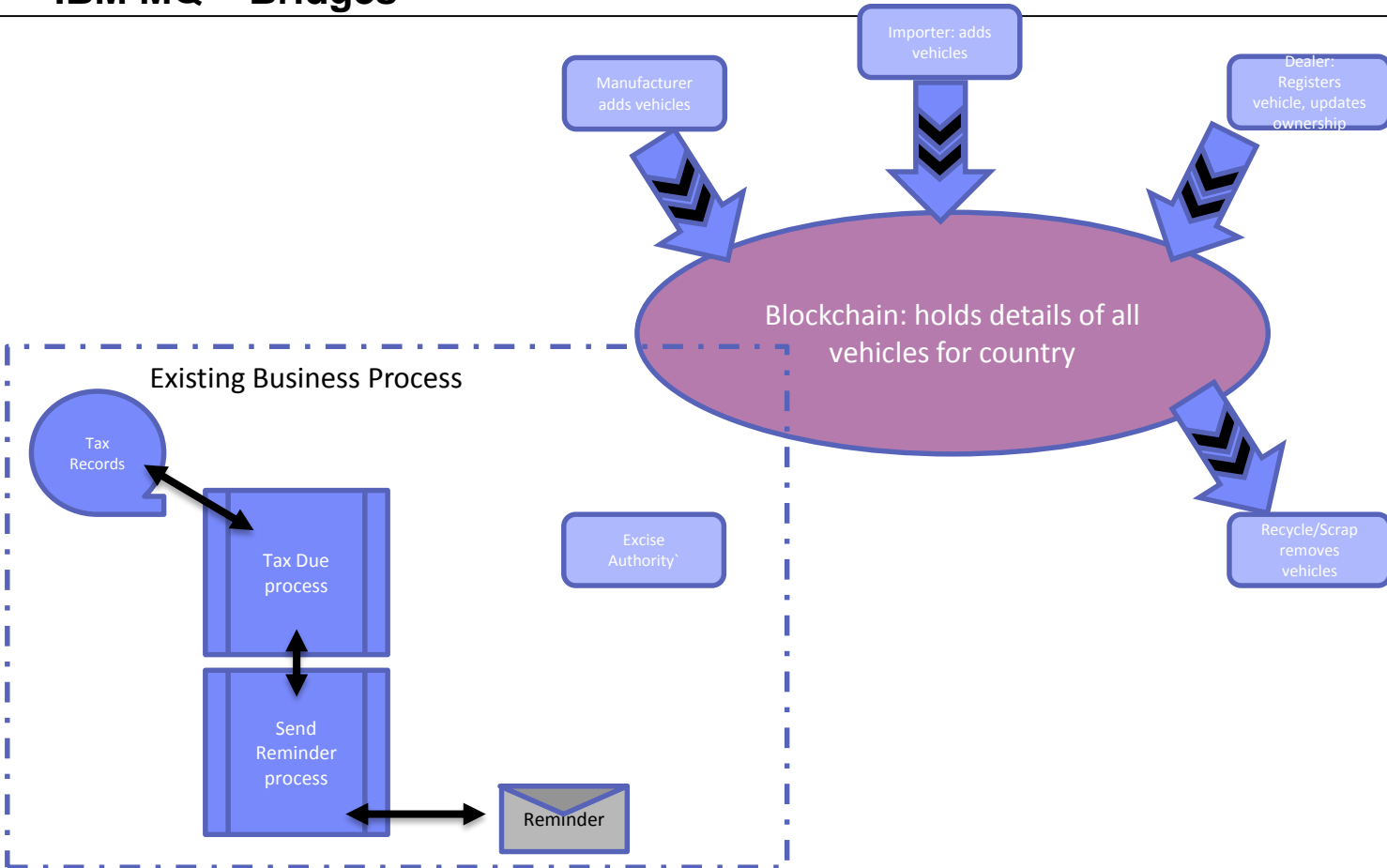
THE MQ BRIDGE TO BLOCKCHAIN

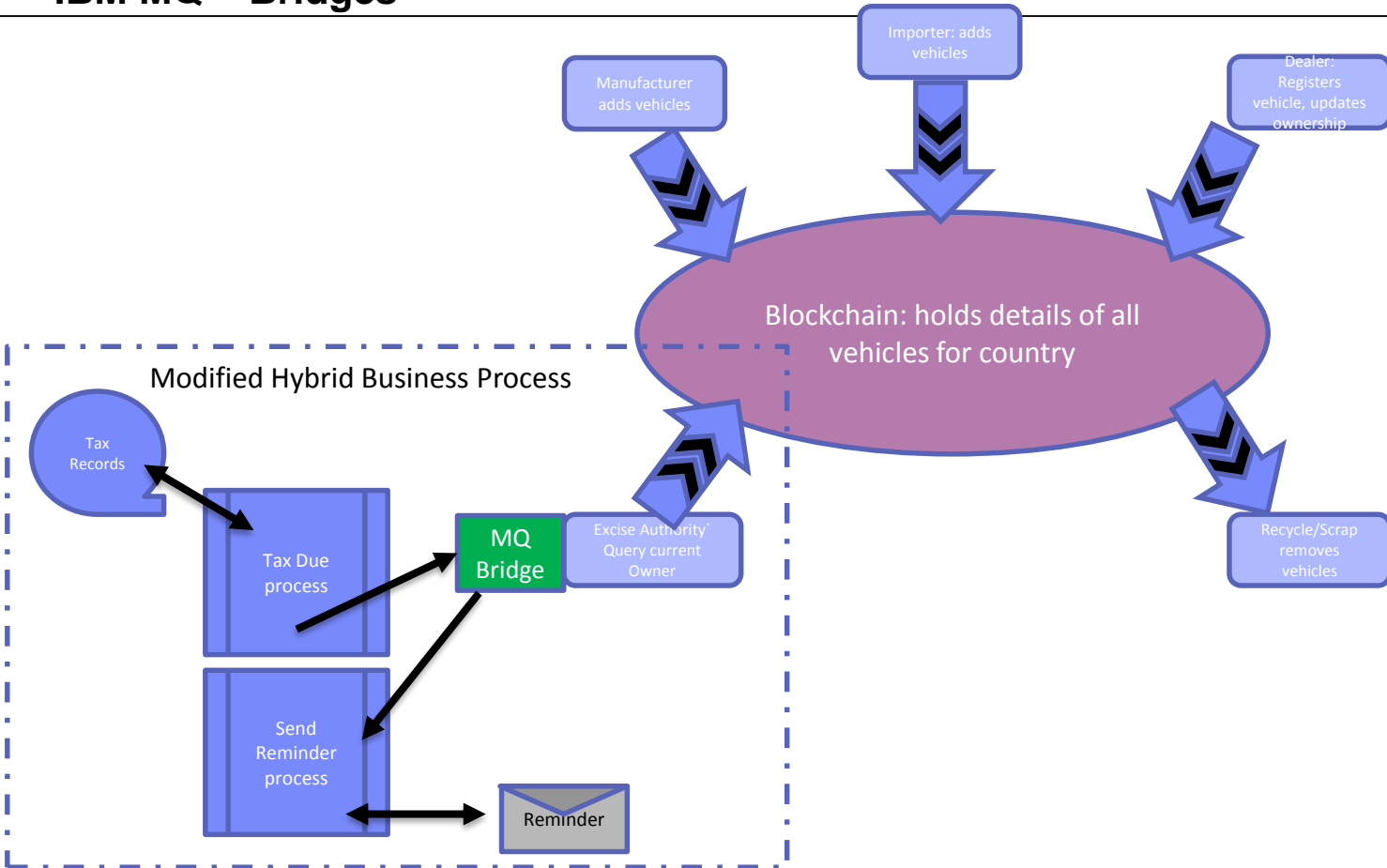
Blockchain Bridge

- MQ is a natural fit to connect existing business transaction systems to remote intra-business ledger services
- Asynchronous request/reply MQ message flow for applications to request information from Blockchain ("what is the value of the balance on this account") over MQ queues









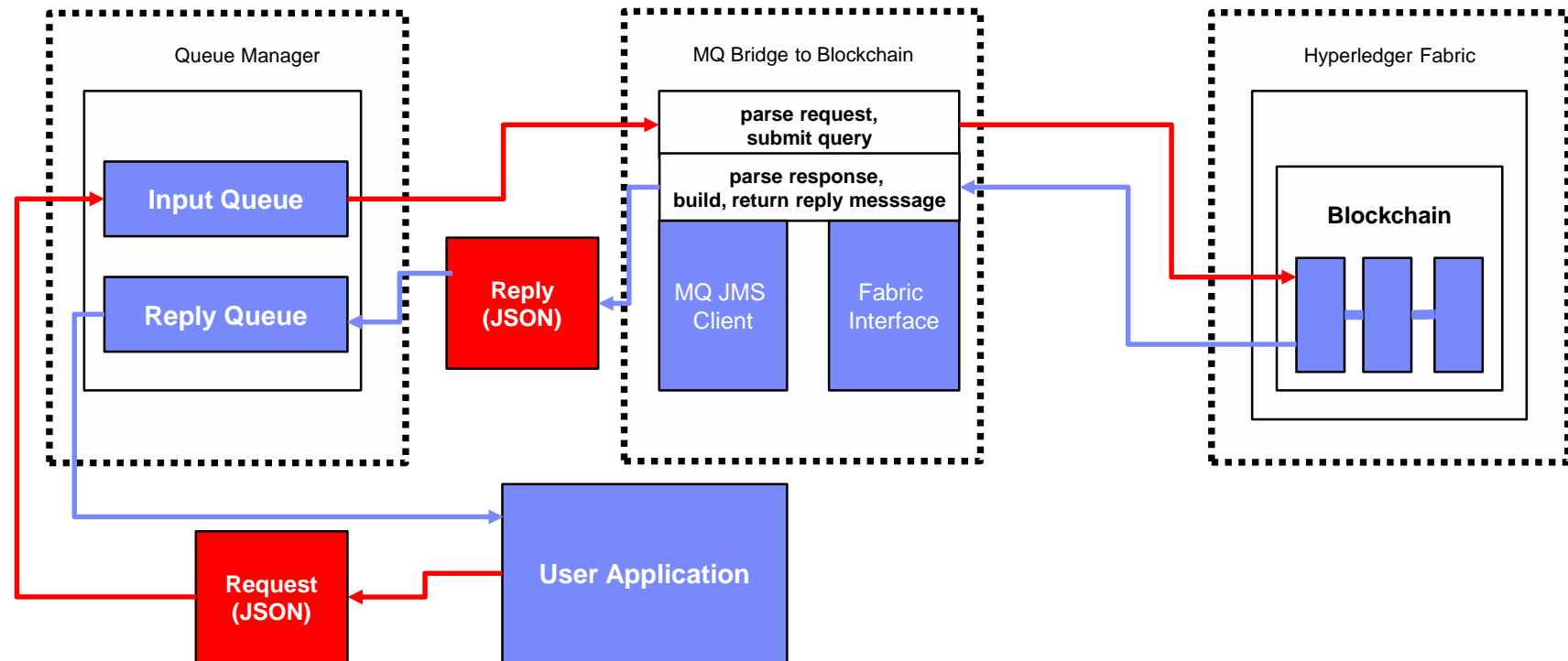
A decorative graphic consisting of several overlapping, wavy blue lines that sweep across the upper half of the slide, creating a sense of motion and depth.

DETAILS

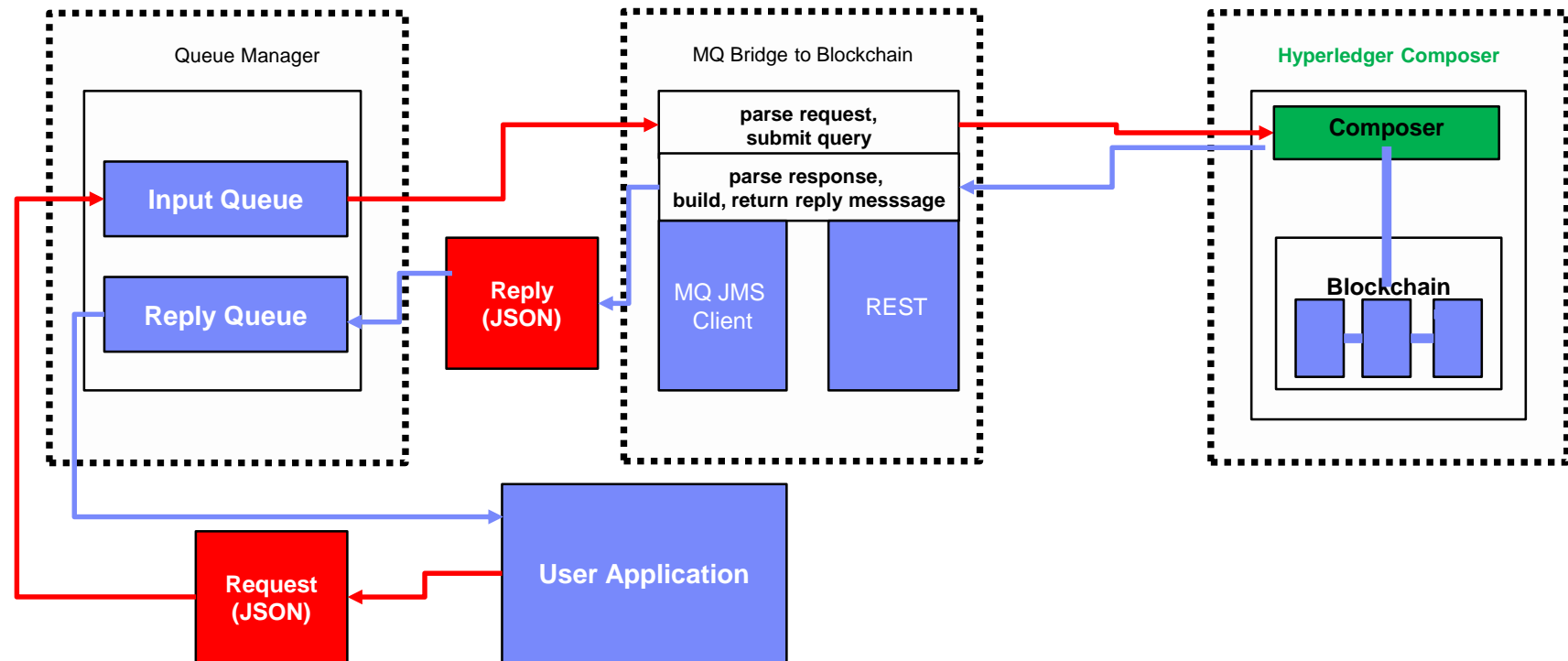
Overview

- The bridge allows an MQ app to query or update data held in a Hyperledger Fabric system
 - Without needing to know anything about the ledger configuration or controls
 - No special APIs
- Support:
 - Only supported for connections to a queue manager with "Advanced" license
 - Including z/OS VUE Advanced, MQ Appliance
 - Bridge itself supported only on xLinux and z/OS
- Initial release in 9.0.3
 - Modified and extended as Hyperledger has changed
 - Likely to change further as Hyperledger evolves
- V9.1 bridge based on Hyperledger Composer
 - A simplified, "business-level" view of resources

Initial release



V9.1



Installation/configuration

- Bridge package is shipped as installable fileset for xLinux, and as part of z/OS Adv product
 - Includes bridge code, sample code and definitions, and prereq 3rd party jars
- Config asks for connection information to MQ and to Blockchain
 - For Blockchain, need address of REST server, a userid and password
 - For MQ, needs client connection information, optionally userid and password
 - TLS configuration
- Configure qmgr with necessary queues and security
- Can then start in normal runtime mode
 - Connections made to both sides at startup, so configuration errors are rapidly discovered

Configuration (partial)

```
-----
Queue Manager                : []QM1
Bridge Input Queue           : [SYSTEM.BLOCKCHAIN.INPUT.QUEUE]APPL1.BLOCKCHAIN.INPUT.QUEUE
MQ Channel                   : []
MQ Conname                   : []
MQ CCDT URL                  : []
JNDI implementation class     : [com.sun.jndi.fscontext.RefFSContextFactory]
JNDI provider URL            : []
MQ Userid                    : []
MQ Password                   : []
```

User Identification

```
-----
Userid                       : []
Password                      : []
API path for login           : [auth/users/login]
```

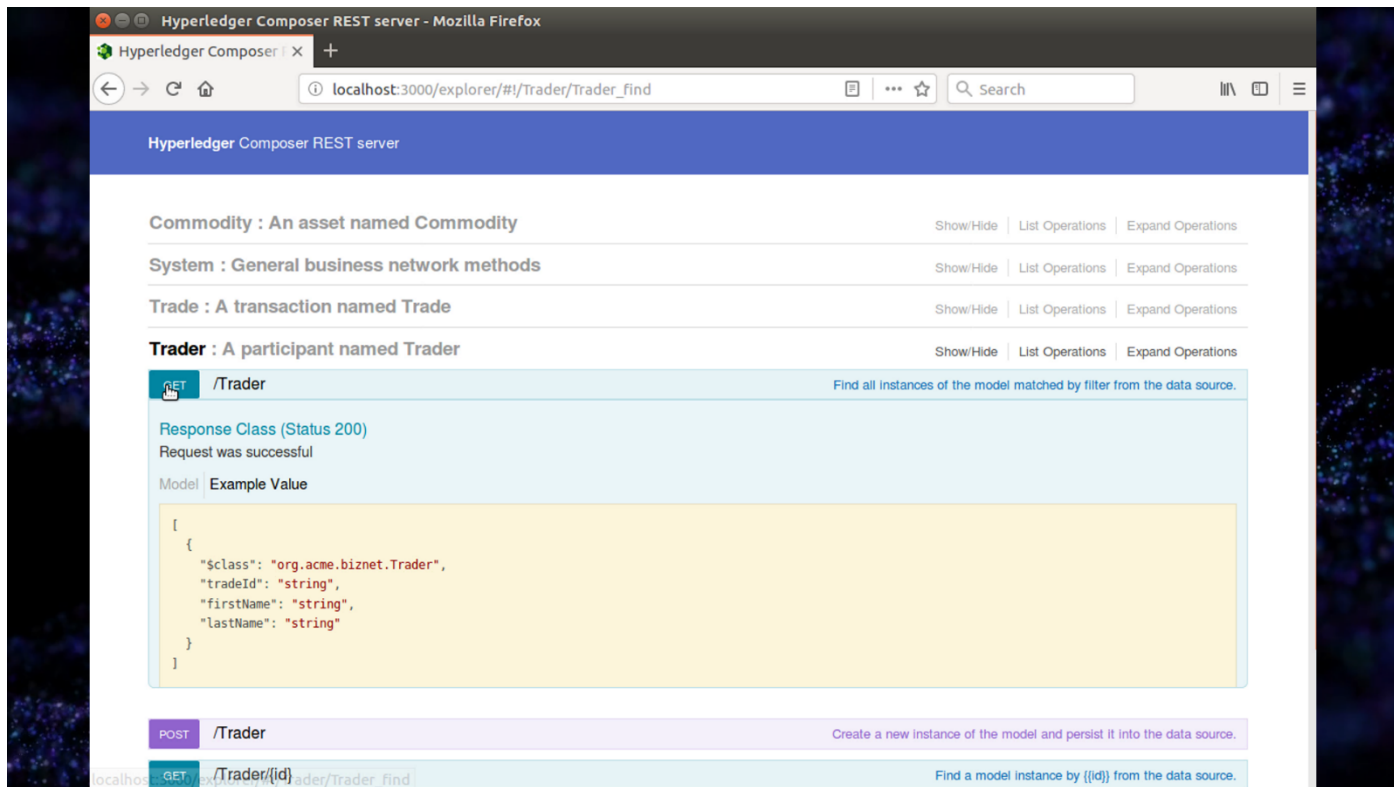
REST Server

```
-----
Address for Composer REST server : []localhost:3000
```

Certificate stores for TLS connections

```
-----
Personal keystore             : []
```

Composer REST API



Hyperledger Composer REST server - Mozilla Firefox

Hyperledger Composer | X

localhost:3000/explorer/#!/Trader/Trader_find

Hyperledger Composer REST server

Commodity : An asset named Commodity Show/Hide | List Operations | Expand Operations

System : General business network methods Show/Hide | List Operations | Expand Operations

Trade : A transaction named Trade Show/Hide | List Operations | Expand Operations

Trader : A participant named Trader Show/Hide | List Operations | Expand Operations

GET /Trader Find all instances of the model matched by filter from the data source.

Response Class (Status 200)
Request was successful

Model	Example Value
	<pre>[{ "\$class": "org.acme.biznet.Trader", "tradeId": "string", "firstName": "string", "lastName": "string" }]</pre>

POST /Trader Create a new instance of the model and persist it into the data source.

GET /Trader/{id} Find a model instance by {id} from the data source.

Application interface

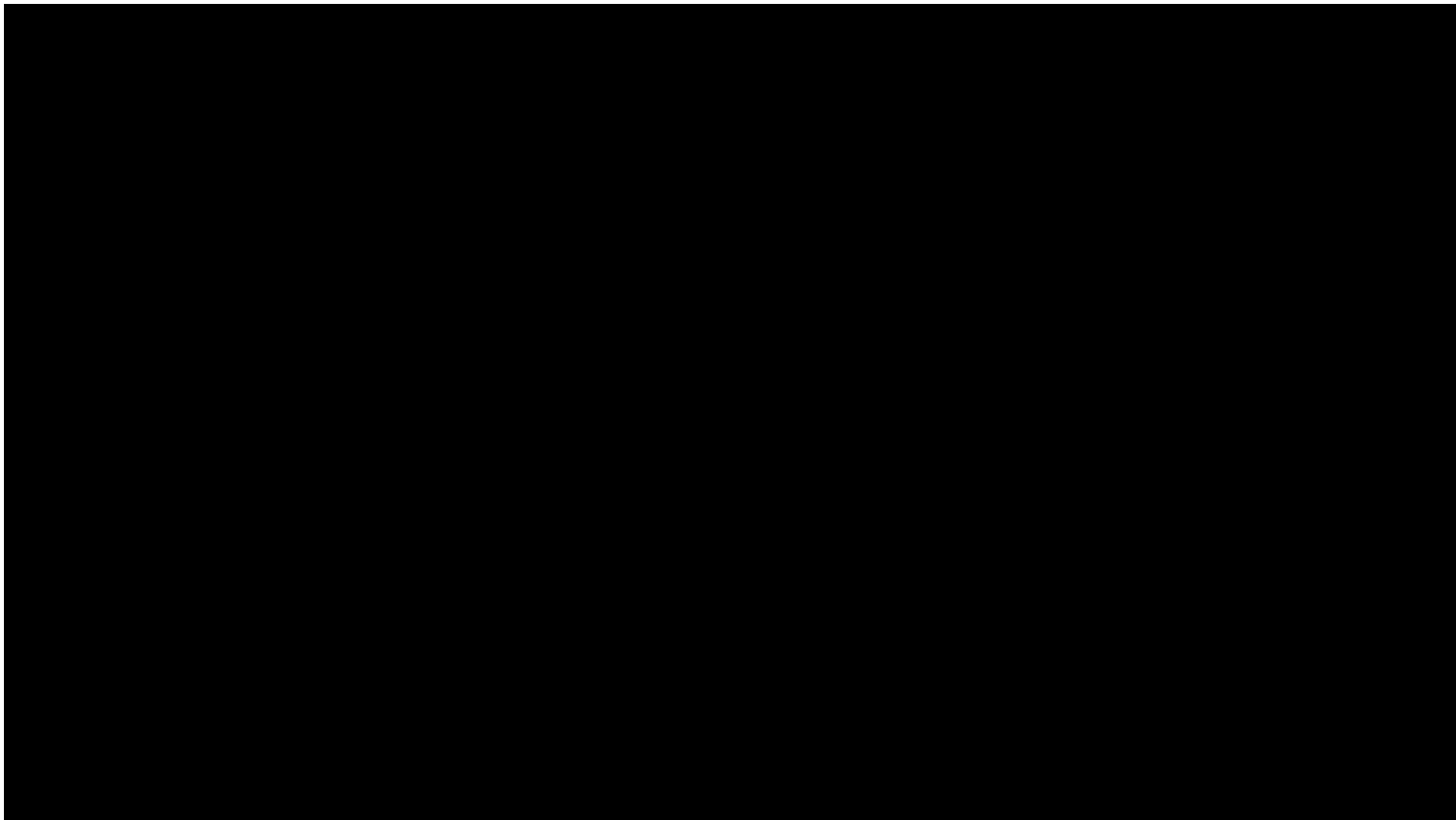
- Applications interact with the bridge by putting a JSON-formatted message on the queue
 - Sample app provided to show construction of fields

```
{ "method": "POST",  
  "path": "api/Trader",  
  "body": { "$class" : "Trader",  
            "tradeId" : "Trader2",  
            "firstName": "Jane",  
            "lastName" : "Doe" }}
```

- Response contains requested data and indication of success or failure)
 - Errors may come directly from the bridge (eg badly formatted input message), from Hyperledger fabric (eg unknown method), or from the chaincode itself (eg bad function parameters)

```
{ "statusCode": 200,  
  "statusType": "SUCCESS",  
  "data": [ {  
    "$class": "Trader",  
    "firstName": "John",  
    "lastName": "Doe",  
    "tradeId": "Trader1"  } ] }
```

```
{ "statusCode": 500,  
  "statusType": "FAILURE",  
  "message": "Error trying invoke business  
network. Error: chaincode error (status: 500,  
message: Error: Failed to add object with ID  
'Trader1' as the object already exists)\n"  
}
```



A decorative graphic consisting of several overlapping, wavy blue lines that sweep across the upper half of the slide, creating a sense of motion and flow.

IBM MQ BRIDGE TO SALESFORCE

IBM and Salesforce partnership

IBM and Salesforce
announce a new landmark
partnership

[Read the press release](#)



Accelerate decision making and
drive greater customer success

IBM and Salesforce will deliver joint solutions designed to leverage artificial intelligence and enable companies to make smarter decisions, faster than ever before. The partnership will bring insights from Watson directly into the Salesforce Intelligent Customer Success Platform, combining deep customer insights from Salesforce Einstein with Watson's structured and unstructured data across many industries including healthcare, financial services, retail and weather.

Sign up for our webinar to learn how you can bring IBM and Salesforce to your organization.

[Register for webinar](#)

MQ Bridge to Salesforce

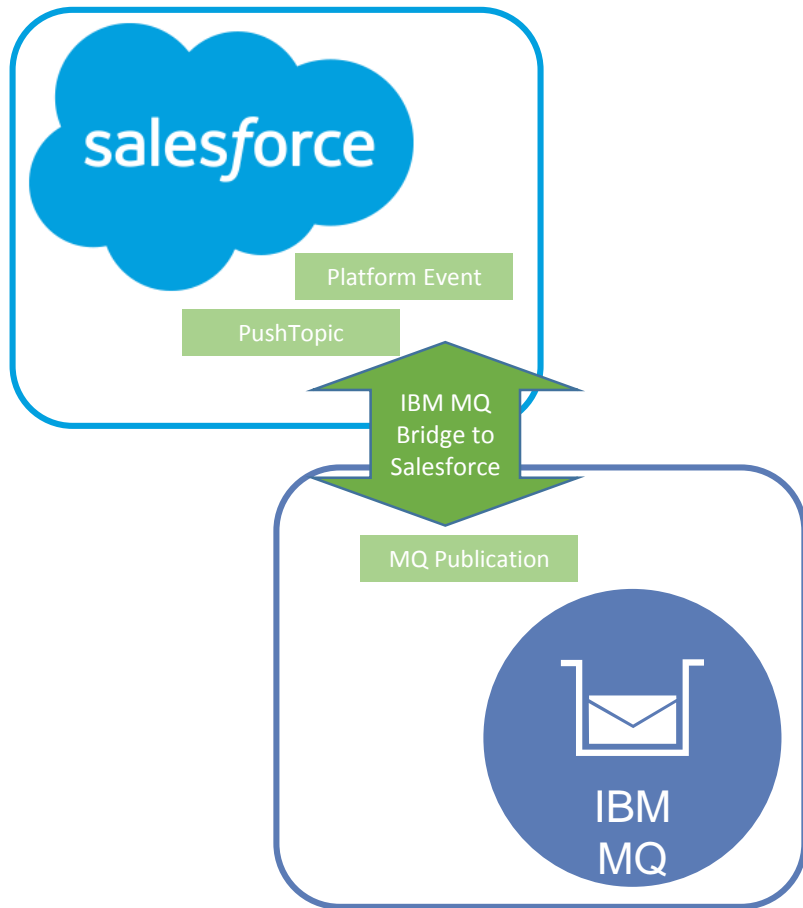
Salesforce's cloud-based CRM platform enables events to be emitted when changes are made to data, or when applications run. It allows apps to consume these events.

- You can inject these Salesforce events into your MQ systems using the new MQ Bridge for Salesforce with no need for your backend applications to connect to Salesforce
- You can create events in your MQ applications that are published to applications running in Salesforce

Supports Salesforce Platform Events and PushTopics

Uses MQ's publish/subscribe capabilities

The bridge runs on xLinux, but connects to any queue manager and is enabled for monitoring with system topic metrics

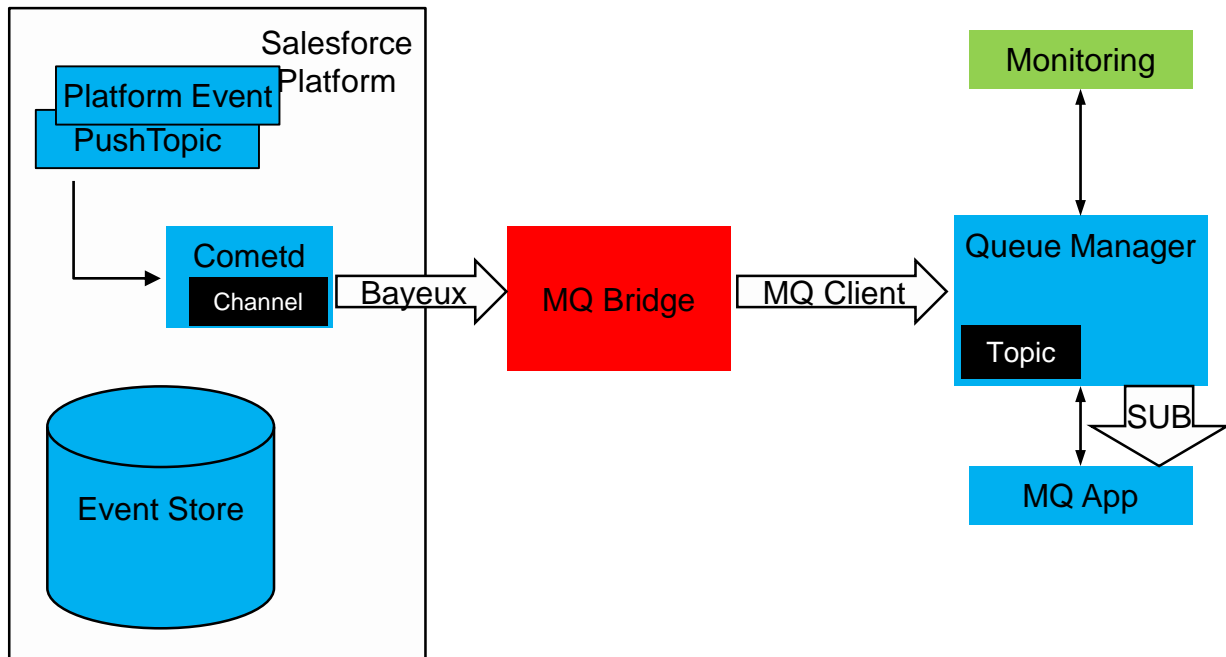


IBM MQ to Salesforce bridge

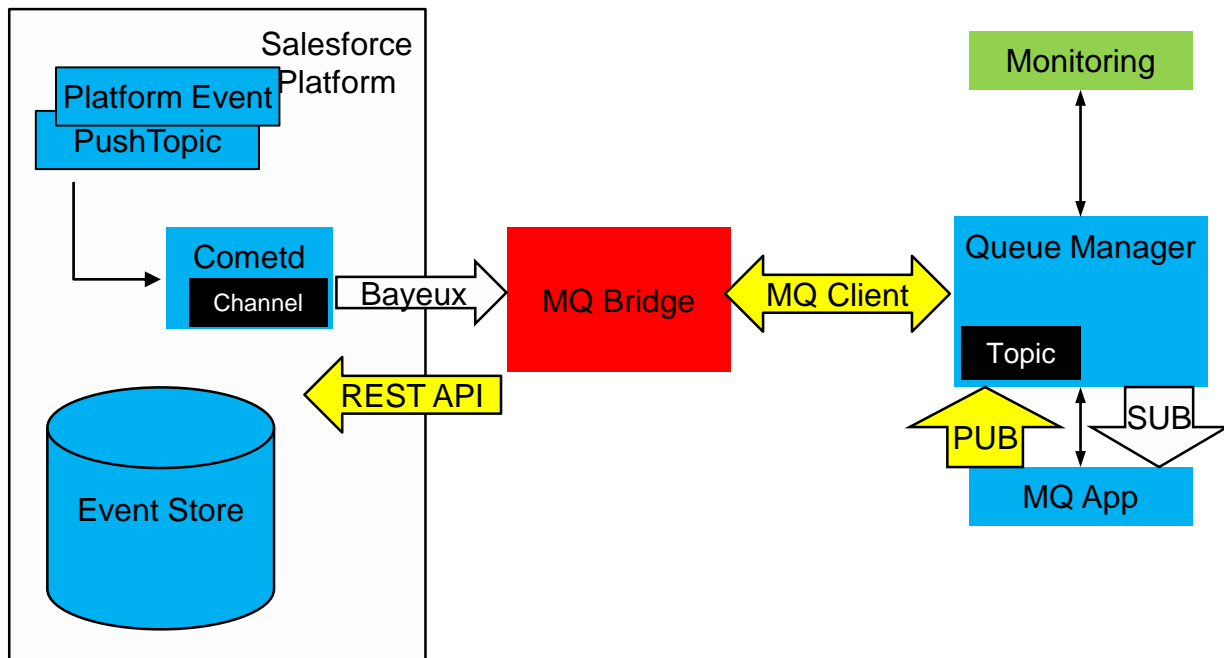
initial availability in MQ 9.0.2
updated for MQ 9.0.4



Design



Design





DETAILS

What are PushTopics and Platform Events

- PushTopics are "implicit" events generated by changes to Salesforce database tables
- Caused by a SOQL (SQL-ish) statement
- Whenever a table is changed, the SOQL statement is executed to see if it is satisfied
 - If so, PushTopic is created containing requested fields
- Platform Events are "explicit" events, created by user-written application code
- New capability in Salesforce since March 2017
- Administrator defines the name of the event, and the fields it should contain
- Both are made available via a pub/sub API and protocol to any application

TABLE: "CASE"

ID	NAME	STATUS	SUBJECT
AAAAAA	Customer1	Normal	Paper jam

TABLE: "CASE"

ID	NAME	STATUS	SUBJECT
AAAAAA	Customer1	Normal	Paper jam
AAABBB	Customer2	Normal	Phone won't stop ringing

TABLE: "CASE"

ID	NAME	STATUS	SUBJECT
AAAAAA	Customer1	Normal	Paper jam
AAABBB	Customer2	Normal	Phone won't stop ringing
BBBCCC	Customer3	Escalated	Printer on fire



GENERATE PUSHTOPIC

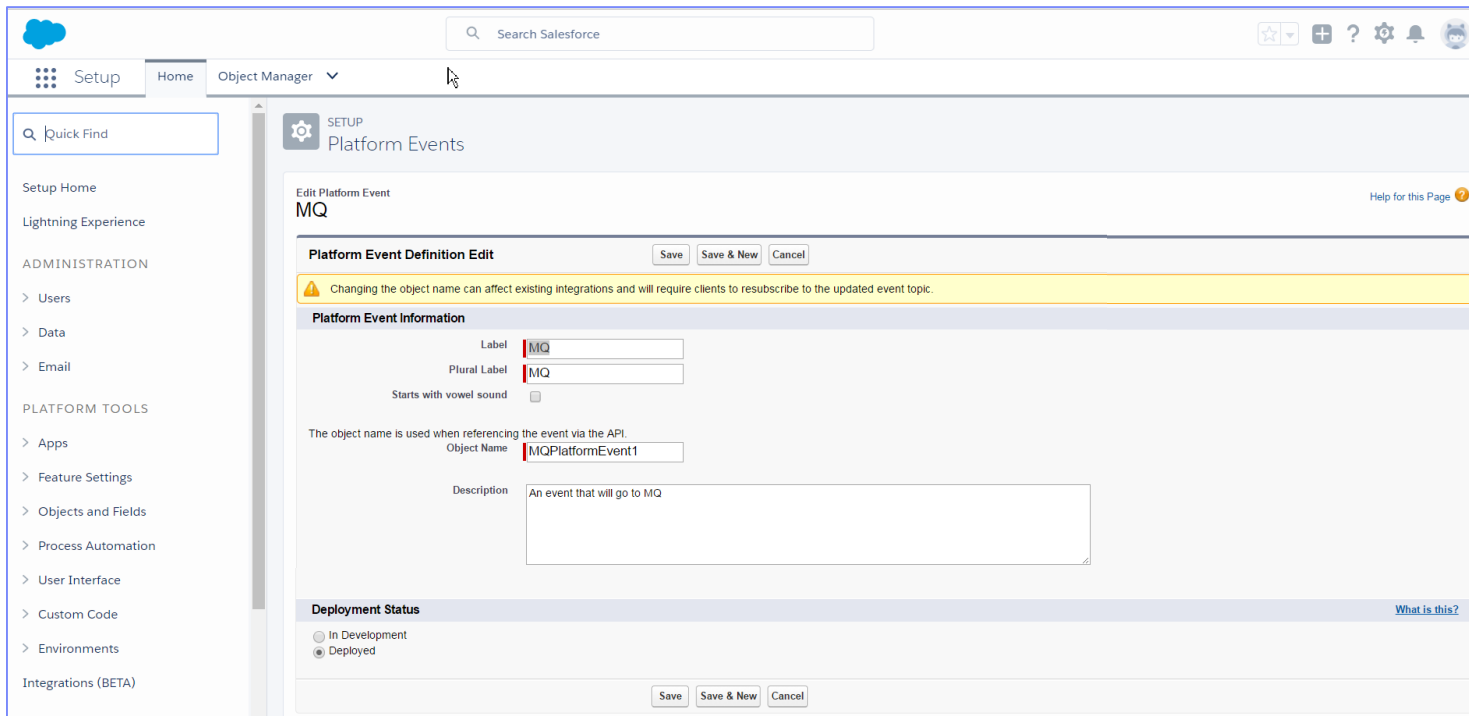
Defining a PushTopic

- Can be done by executing APEX code
 - Others APIs exist

```
PushTopic pushTopic = new PushTopic();  
pushTopic.Name = 'EscalatedCaseUpdates';  
pushTopic.Query = 'SELECT Id, AccountId, Subject FROM CASE  
                    WHERE Status = \'Escalated\' AND AccountId != NULL';  
pushTopic.ApiVersion = 39.0;  
pushTopic.NotifyForOperationCreate =true;  
pushTopic.NotifyForOperationUpdate =true ;  
pushTopic.NotifyForOperationUndelete =true ;  
pushTopic.NotifyForOperationDelete = true ;  
pushTopic.NotifyForFields = 'Referenced';  
insert pushTopic;
```

- PushTopic will now be generated automatically whenever the Query is satisfied

Defining a Platform Event



The screenshot shows the Salesforce Setup interface for defining a Platform Event. The left sidebar contains navigation links: Setup Home, Lightning Experience, ADMINISTRATION (Users, Data, Email), PLATFORM TOOLS (Apps, Feature Settings, Objects and Fields, Process Automation, User Interface, Custom Code, Environments), and Integrations (BETA). The main content area is titled "Platform Events" and shows the "Edit Platform Event" page for an event named "MQ".

Platform Event Definition Edit [Save] [Save & New] [Cancel]

Platform Event Information

Label

Plural Label

Starts with vowel sound ☐

The object name is used when referencing the event via the API.

Object Name

Description


Deployment Status [What is this?](#)


☐ In Development

☒ Deployed

[Save] [Save & New] [Cancel]

Platform Event definition in Salesforce

 **SETUP**
Platform Events

 Platform Event
MQ

[Standard Fields \[3\]](#) | [Custom Fields & Relationships \[2\]](#)

Platform Event Definition Detail [Edit](#) [Delete](#)

Singular Label	MQ	
Plural Label	MQ	Deploy
Object Name	MQPlatformEvent1	
API Name	MQPlatformEvent1__e	
Created By	Mark Taylor	03/01/2017 11:19

Standard Fields

Action	Field Label	Field Name	Data Type
Created By		CreatedBy	Lookup(User)
Created Date		CreatedDate	Date/Time
Replay ID		ReplayId	External Lookup

Custom Fields & Relationships [New](#)

Action	Field Label	API Name	Data Type	Indexed	Controlling Field
Edit Del	MyText	MyText__c	Text(64)		
Edit Del	Name	Name__c	Text(32)		

Creating a Platform Event

- Can be done by executing APEX code
 - Others APIs exist

```
MQPlatformEvent1__e pe;  
  
pe = new MQPlatformevent1__e(Name__c='Mark');  
  
Database.SaveResult result = EventBus.publish(pe);  
  
if (result.isSuccess()) {  
    System.debug('Published OK');  
} else {  
    for (Database.Error err : result.getErrors()) {  
        System.debug('Failed with ' + err.getStatusCode() + err.getMessage());  
    }  
}
```

Getting started with the bridge

- Configure the bridge
 - Connection details for MQ (queue manager name, channel, userid etc)
 - Connection details for Salesforce (id, password, OAuth secret, URL etc)
 - TLS keystores (uses same stores for both sides of bridge)
 - Topic information (which topics to listen to from Salesforce, publication root for MQ, QoS)
- Configure Salesforce
 - Define pushtopic rules, define platform event names, add any event generation code
- Configure MQ
 - Define queue for synchronisation values
 - Define topic root, set security rules
- And then start the bridge so that any MQ application can send and receive publications

MQ Applications - Subscribing

- Any MQ application can subscribe to the topics
 - Standard MQ APIs for accessing these
- "Application" could be IIB message flow that exploits Salesforce node for further processing
- Data is JSON format (option to republish full data or just subject/payload)

```
{  "clientId": "w211mr7nytfdadqzulpq4sbovyh",
  "data": {
    "event": {
      "createdDate": "2016-11-12T19:09:31.384Z", "replayId": 10, "type": "created"
    },
    "subject": {
      "AccountId": "Y000009Yxb", "Id": "Y000001QGp", "Subject": "Acct updated"
    }
  },
  "channel": "/topic/EscalatedCaseUpdates"
}
```

Qualities of Service

- The bridge supports two QoS
 - At **most** once delivery
 - At **least** once delivery
- All events from Salesforce have a sequence number (replayId) unique to the event name
 - Events are held inside Salesforce for potential replay for up to 24 hours
 - The sequence number of last event processed is persisted in MQ
 - On restart, bridge can use that sequence number to request all events since the last one processed
 - At **least** once QoS
 - Or can just ask for only new events
 - At **most** once QoS
- If the bridge out of contact with Salesforce for more than 24 hours, old events are not available
 - Even with at least once QoS

MQ Applications - Publishing

- As well as subscribing to Salesforce topics, bridge subscribes to MQ topics
- MQ applications publish to that tree
- Bridge sends them to Salesforce
 - Then available to any Salesforce application
 - Including any MQ apps listening via the bridge
- Publications contain JSON corresponding to Platform Event fields
 - MQ topic name indicates Salesforce event name

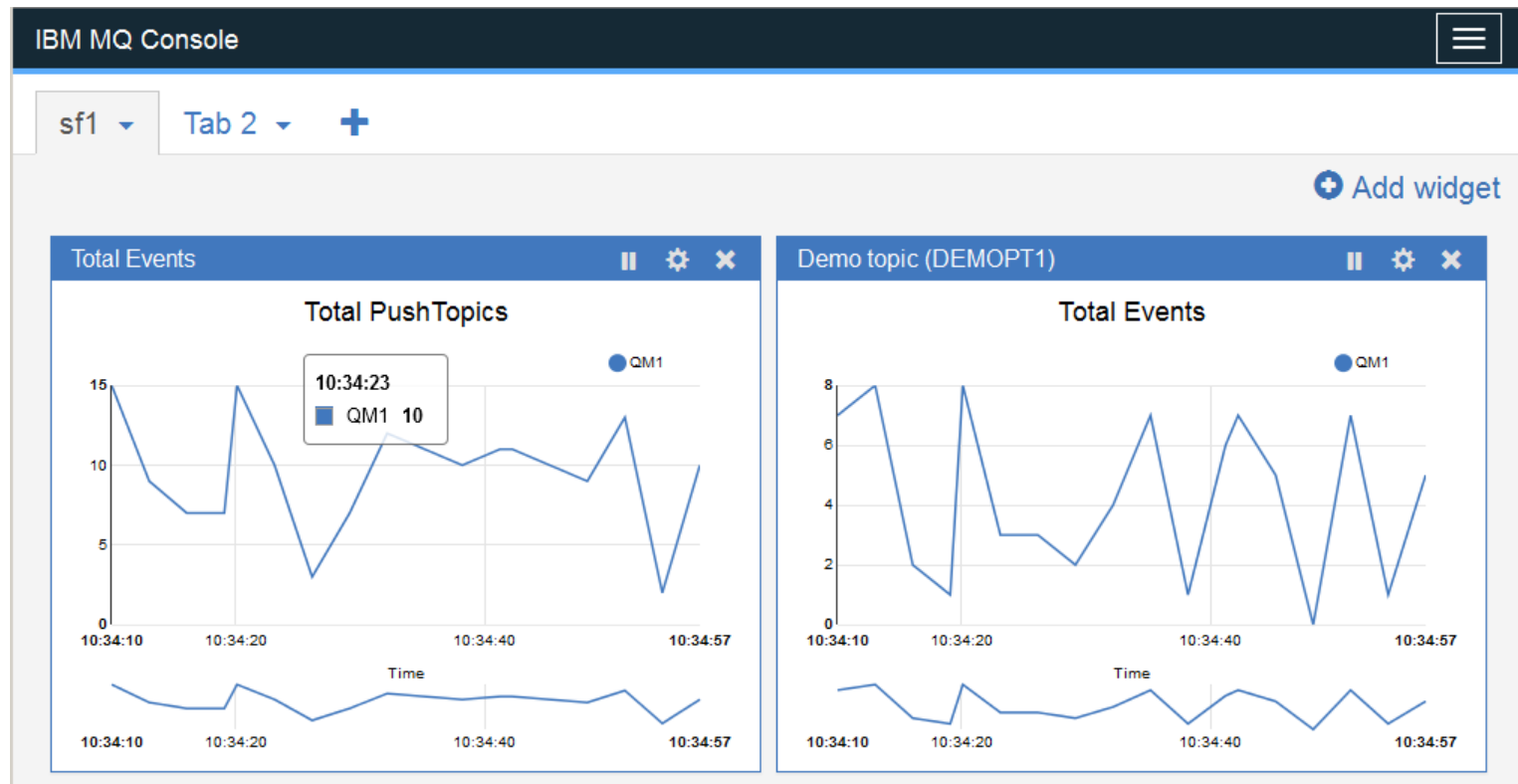
Topic:

```
/<root>/mqtosfb/event/MQPlatformEvent1__e
```

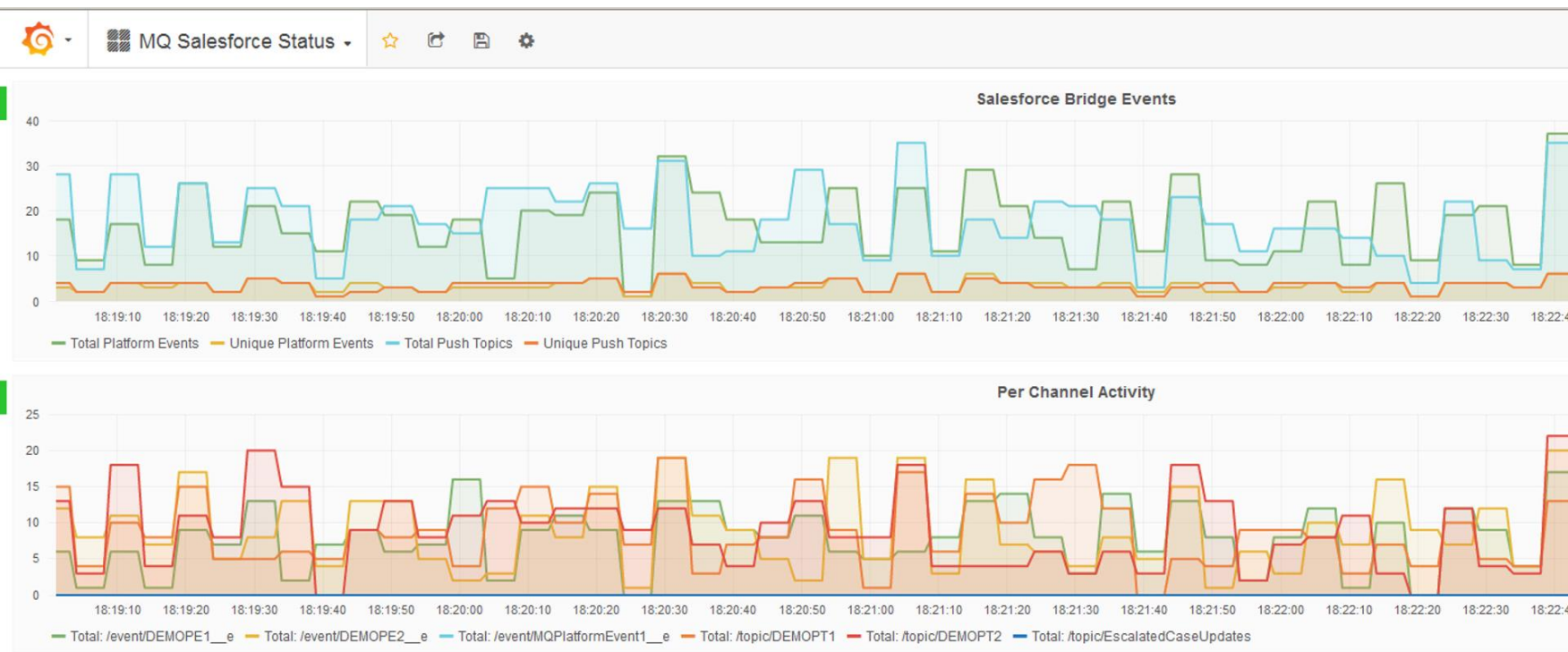
Message Body:

```
{ "MyText__c" : "Some text here", "Name__c" : "Bob Smith" }
```

Monitoring using the MQ Console



Monitoring using Grafana



A decorative graphic consisting of several overlapping, wavy, horizontal bands of blue and light blue, creating a sense of motion and depth, positioned above the main title.

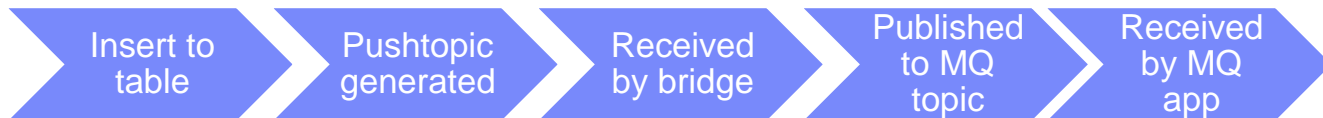
POSSIBLE ENHANCEMENTS

Possible enhancements

- RFEs have been raised that would, if implemented, lead to features such as
- Multiple bridges on a queue manager
- Different QoS for inbound (SF to MQ) and outbound (MQ to SF) publications
- Workload sharing for outbound work

Demo Outline

- A change is made to a Salesforce table
- Resulting in an pushtopic being generated
- Picked up by the bridge
- And republished to an MQ application that is subscribed to the topic



- Round trip publication



A decorative graphic consisting of several overlapping, wavy horizontal bands in various shades of blue, spanning the width of the slide and positioned above the word DEMO.

DEMO

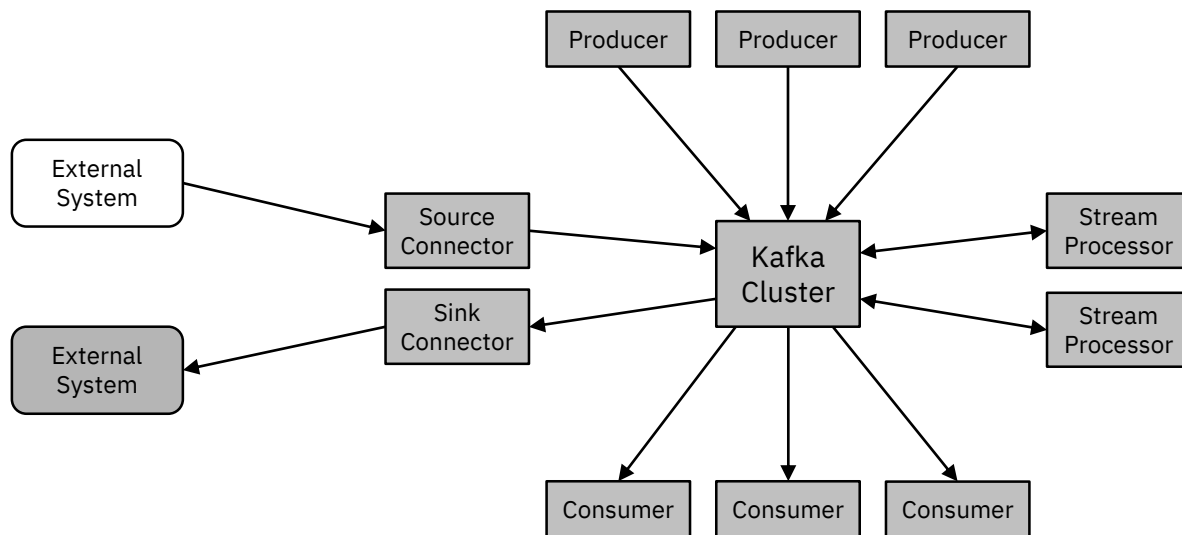
For more info on Salesforce event processing

- MQ Knowledge Centre
 - https://www.ibm.com/support/knowledgecenter/en/SSFKSJ_9.0.0/com.ibm.mq.con.doc/q129310_.htm
- MQ blog entry (and link to video) at
 - https://www.ibm.com/developerworks/community/blogs/messaging/entry/DRAFT_IBM_MQ_and_Salesforce_messaging?lang=en
- PushTopics described at
 - https://developer.salesforce.com/docs/atlas.en-us.api_streaming.meta/api_streaming/intro_stream.htm
- Platform Events at
 - https://developer.salesforce.com/docs/atlas.en-us.platform_events.meta/platform_events/platform_events_intro.htm
- Testing with Salesforce – Can try these out using **FREE** developer accounts

A decorative graphic consisting of several overlapping, wavy, horizontal bands of blue and light blue, creating a sense of motion and depth, spanning the width of the slide above the title.

KAFKA CONNECTORS

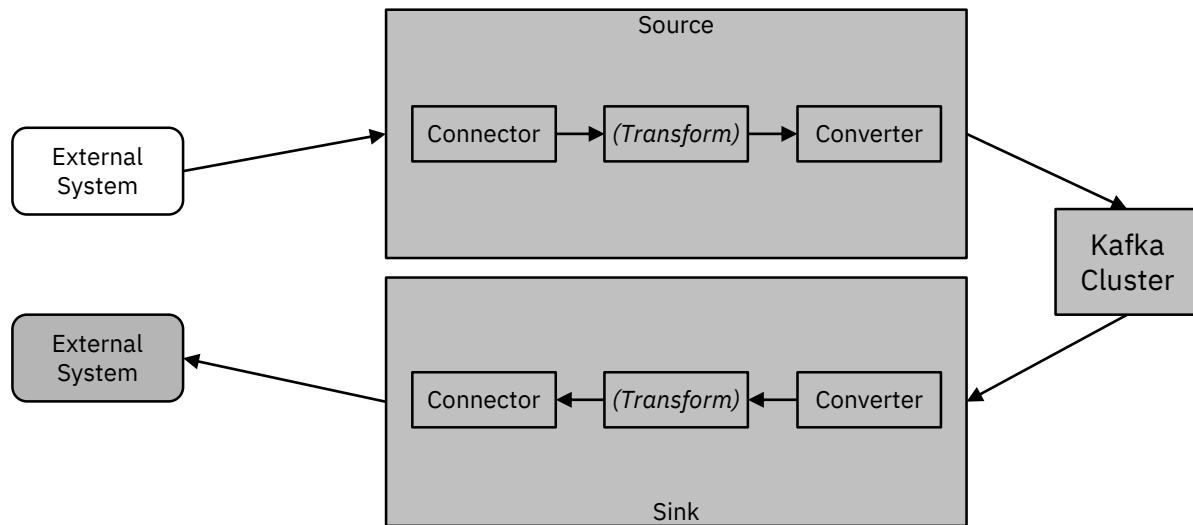
Apache Kafka is an Open-Source Streaming Platform



Use cases

Messaging
Website activity tracking
Metrics
Log aggregation
Stream processing
Event sourcing
Commit log

Kafka Connect



Over 80 connectors

HDFS

Elasticsearch

MySQL

JDBC

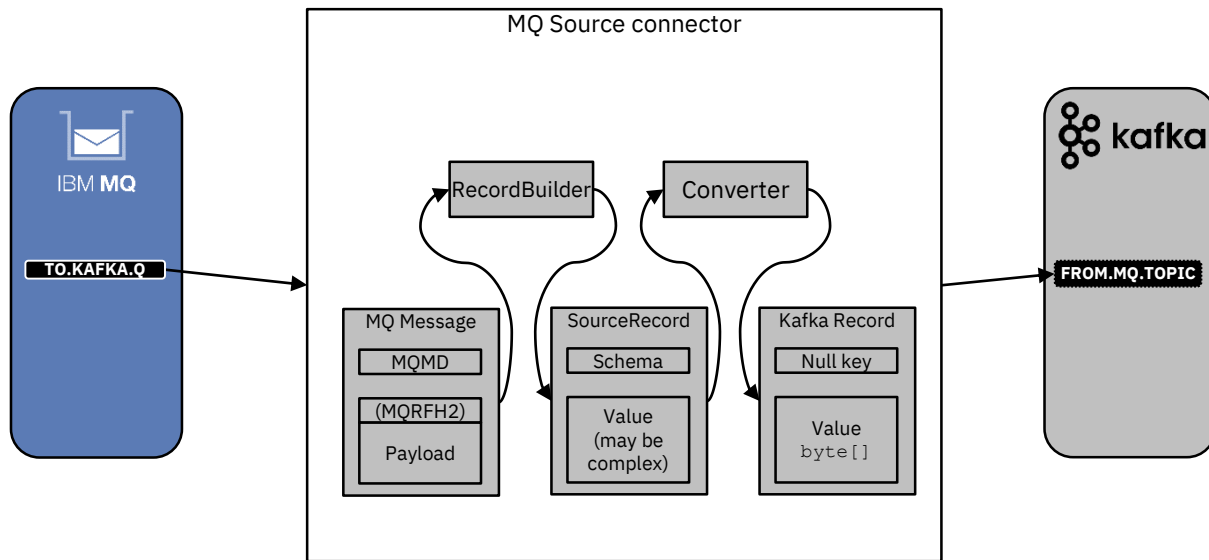
IBM MQ

MQTT

CoAP

+ many others

Kafka Connect MQ Source connector



Open-source – build it yourself

Use any supported MQ release

Uses JMS client internally

Client connections

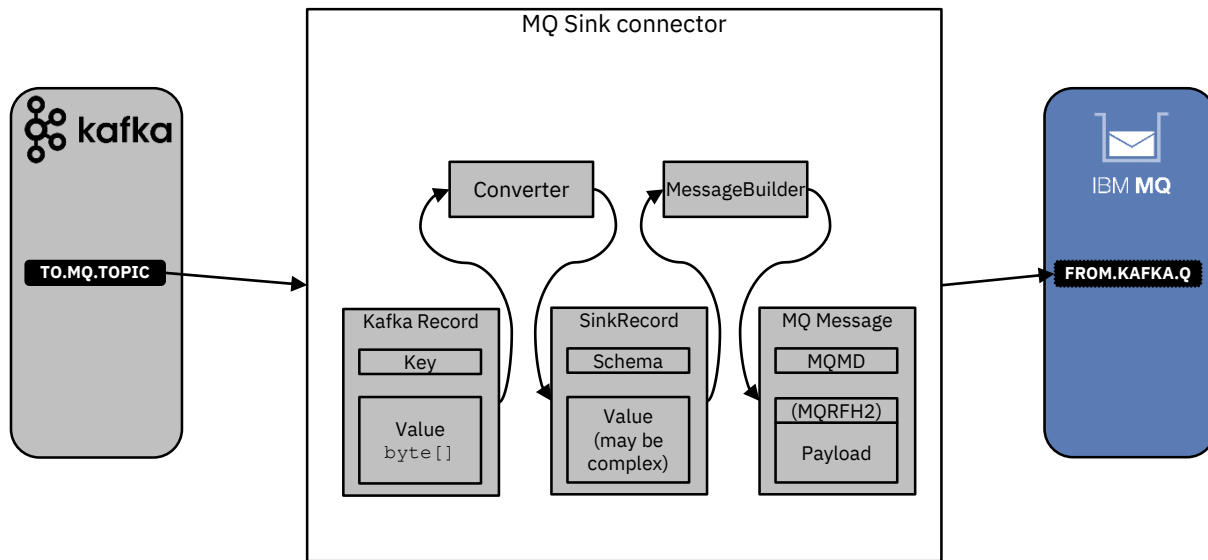
Supports TLS, authentication

MQ queue->Kafka topic

Support for binary, text, JSON

Easy to extend

Kafka Connect MQ Sink connector



- Open-source – build it yourself
- Use any supported MQ release
- Uses JMS client internally
- Client connections
- Supports TLS, authentication
- Kafka topic -> MQ queue
- Support for binary, text, JSON
- Easy to extend

Summary

- Simple extension allows MQ apps to exploit new services
 - Event-driven Salesforce applications
 - Integration with Hyperledger Fabric Blockchain
 - Integration with Kafka event streaming
- Follows existing MQ application models
- Easy configuration
- Easy to extend existing processes



Any questions?