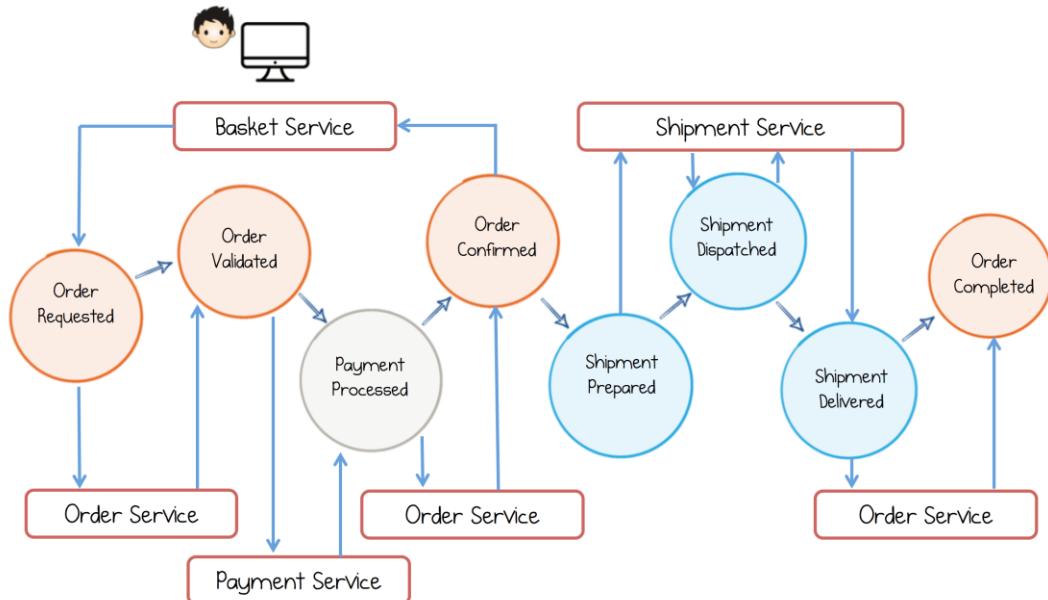
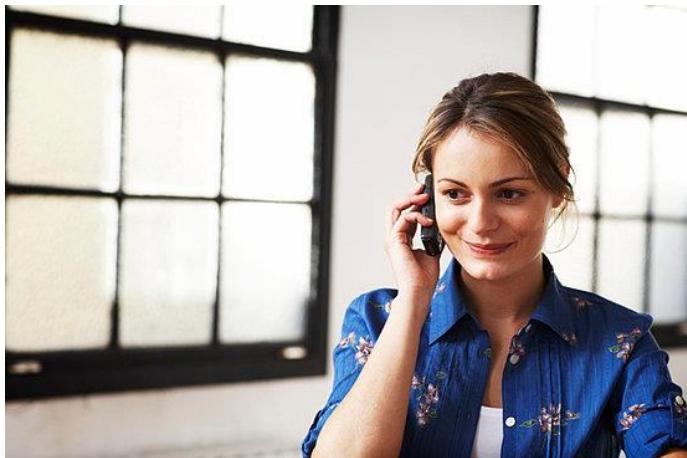


Messaging and brokers



Synchronous vs assynchronous



Synchronous vs assynchronous



“Do MVC like it’s 1979”

REMEMBER?

Models

“Models represent knowledge. A model could be a single object (rather uninteresting), or it could be some structure of objects.” *The original MVC report, 1979.*

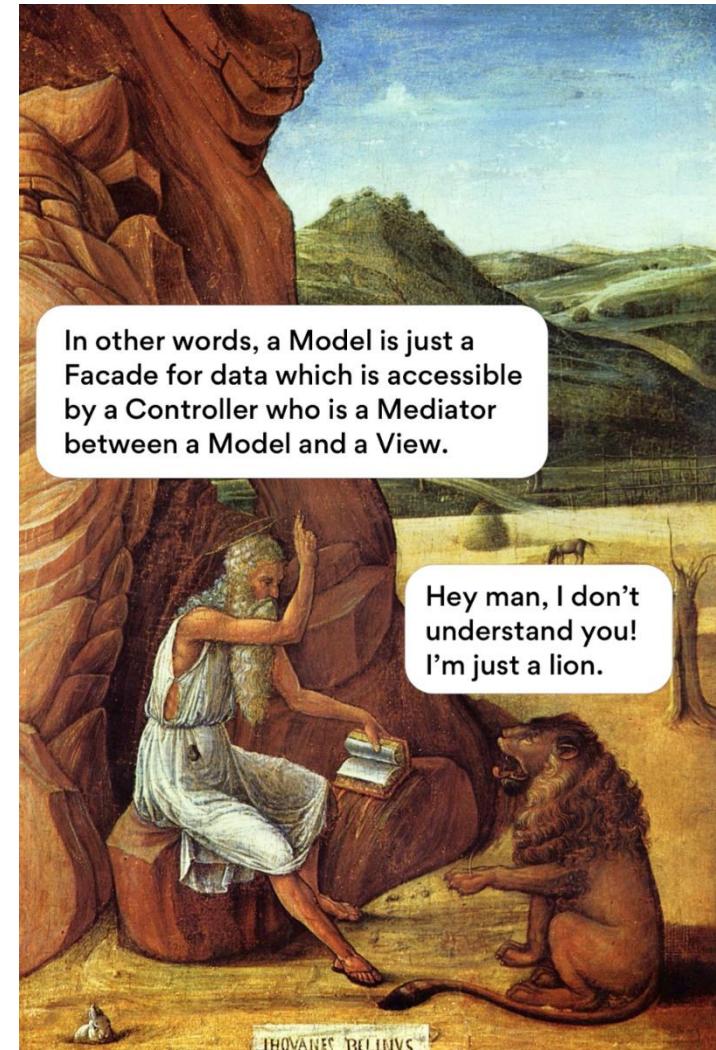
Views

“A view is a (visual) representation of its model. It would ordinarily highlight certain attributes of the model and suppress others.” *The original MVC report, 1979.*

Controllers

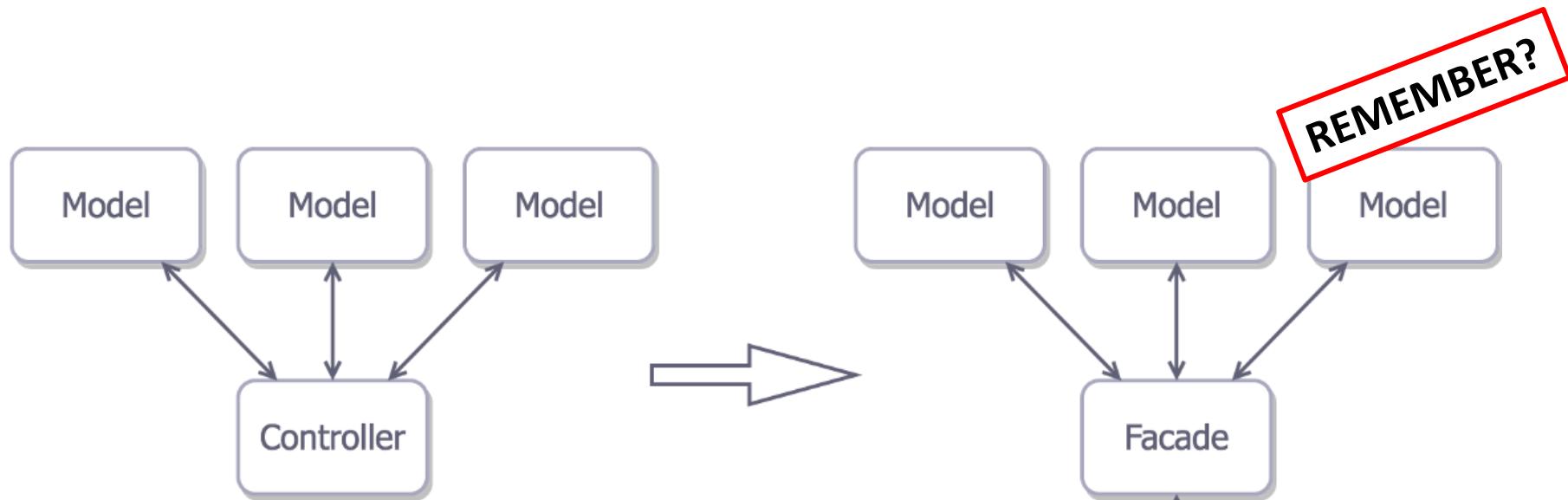
“A controller is a link between a user and the system. It provides the user with input by arranging for relevant views to present themselves in appropriate places on the screen.” *The original MVC report, 1979.*

Note: Controllers were initially named Editors, and then the name was changed.

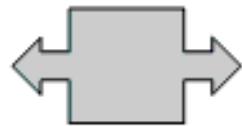


Do MVC like it's 1979

<https://badoootech.badoo.com/do-mvc-like-its-1979-da62304f6568>

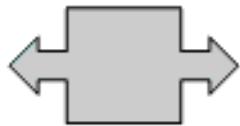


Adapter



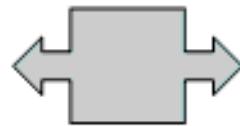
new
interface

Facade



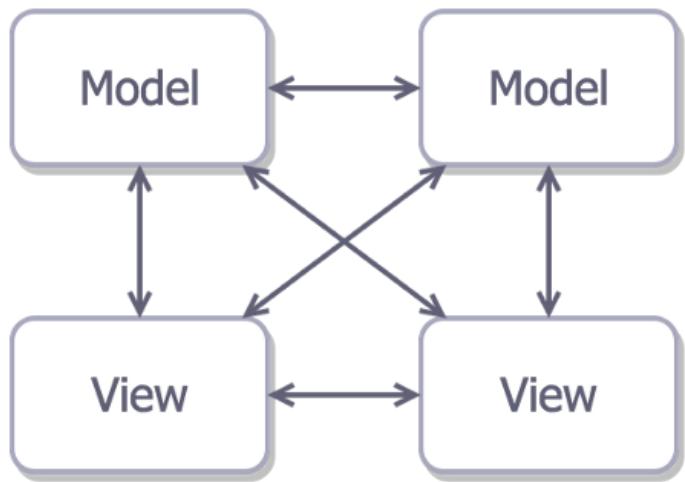
simplified
interface

Proxy

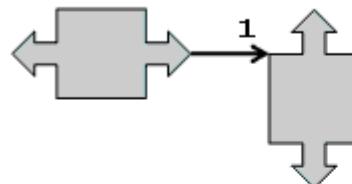


same
interface

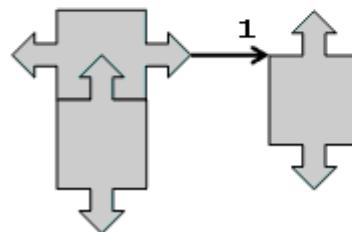
REMEMBER?



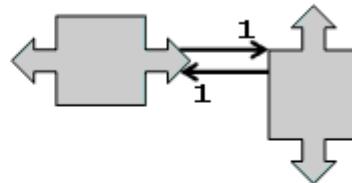
Builder



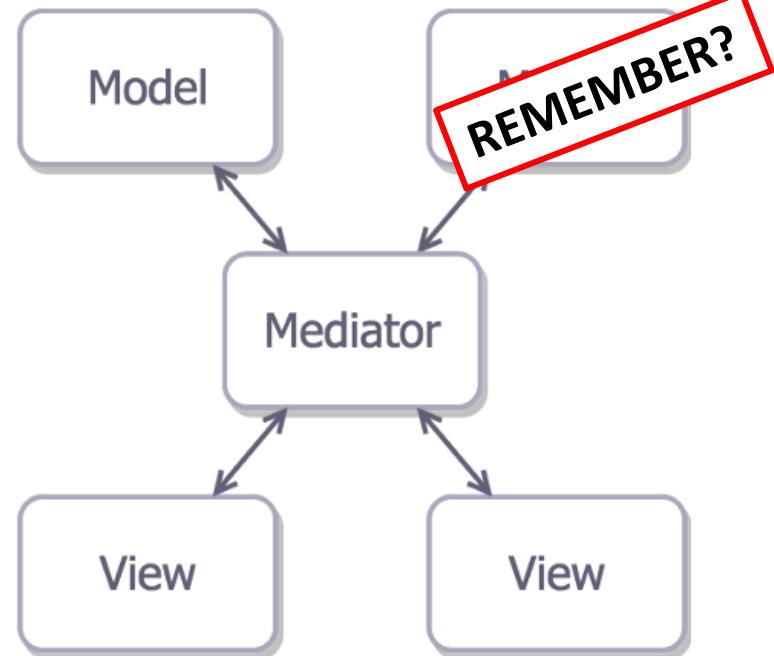
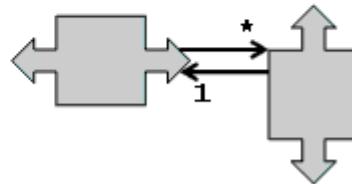
Bridge



State



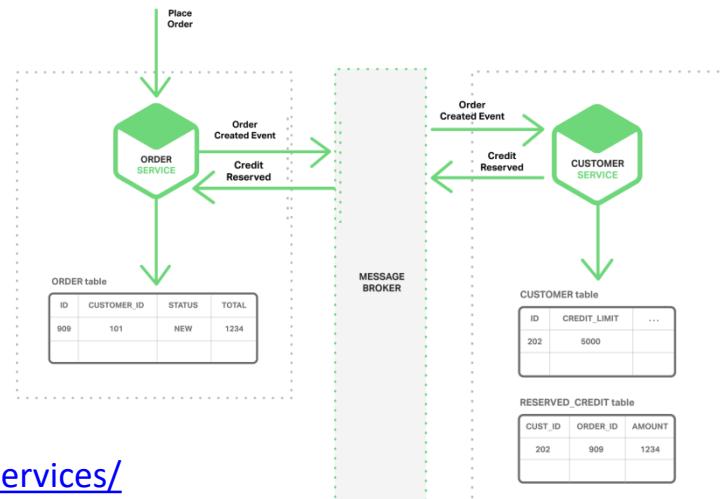
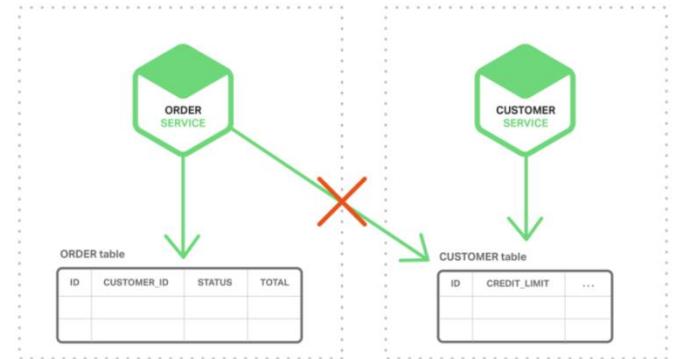
Observer



Design Patterns by Vince Huston
<http://www.vincehuston.org/dp/>

Messaging

- Asynchronous
- Focus on message
- Reduce coupling
- Scalable & Fault tolerant

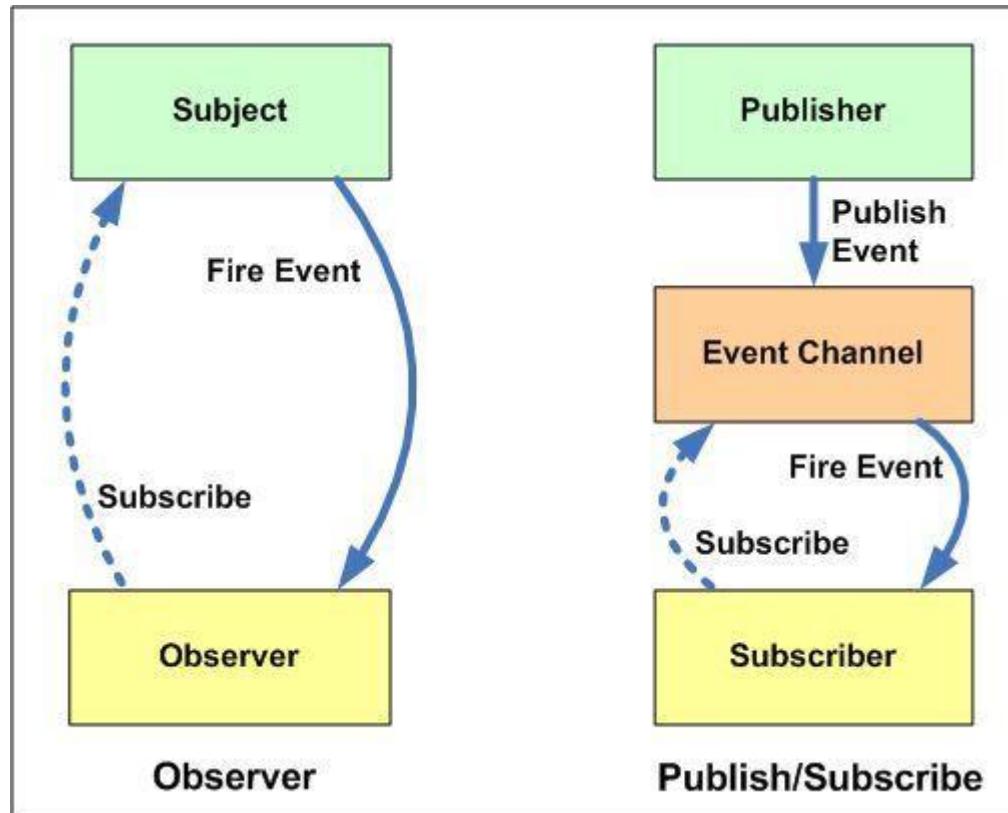


<https://www.nginx.com/blog/event-driven-data-management-microservices/>

Messaging

- Conceptual
 - Reduce coupling
 - Focus on domain
 - Scalable
- Patterns and styles associated
 - Publisher / subscriber
 - Cqrs, Event sourcing
 - Event Driven Architecture
 - ...
- **Concept is technology agnostic**
- **BUT commonly associated with concrete implementations**

Observer vs Pub-Sub pattern



Observer vs Pub-Sub pattern

<https://hackernoon.com/observer-vs-pub-sub-pattern-50d3b27f838c>

Observer vs Pub-Sub pattern

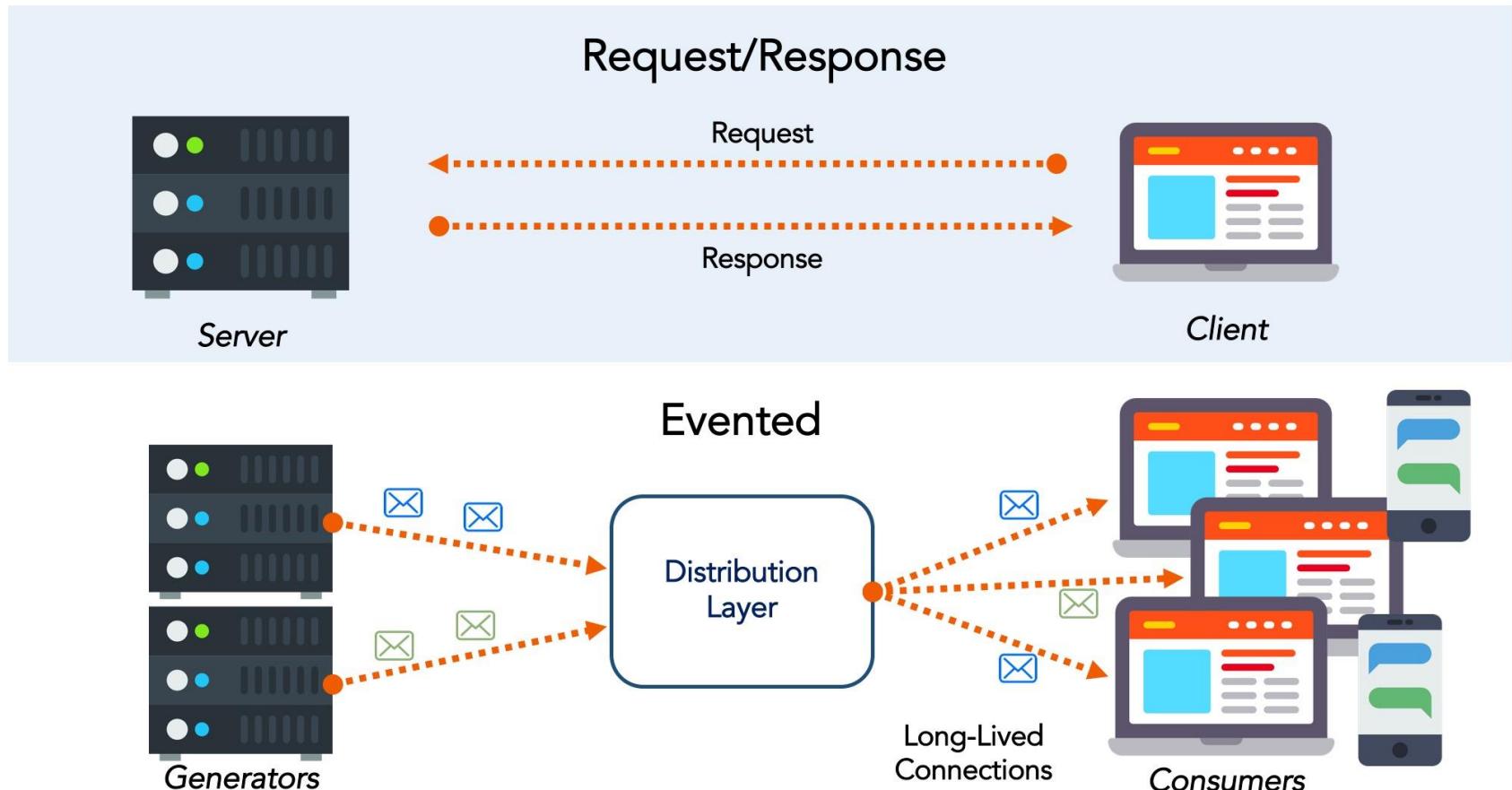
Observer

- *Observers are aware of the Subject, also the Subject maintains a record of the Observers.*
- pattern components **are tightly coupled**
- mostly implemented in a **synchronous** way, i.e. the Subject calls the appropriate method of all its observers when some event occurs.
- pattern need to be implemented in a **single application address space.**

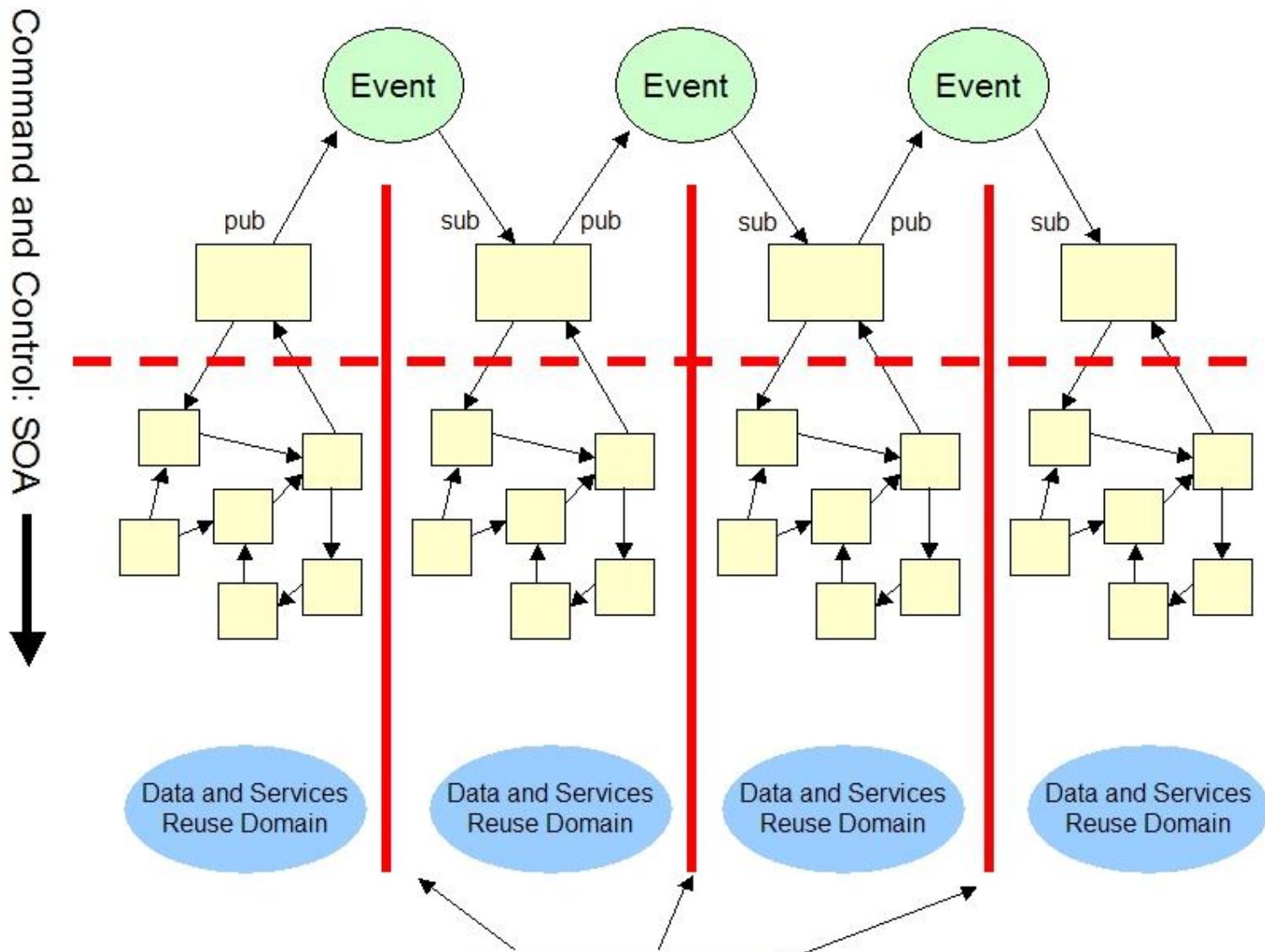
Publisher/Subscriber,

- publishers and subscribers **don't need to know each other**. They simply communicate with the help of message queues or broker.
- pattern, components are **loosely coupled**
- pattern is mostly implemented in an **asynchronous** way (using message queue).
- pattern is more of a **cross application pattern.**

Messaging vs Request / response



Business Process Chain: EDA

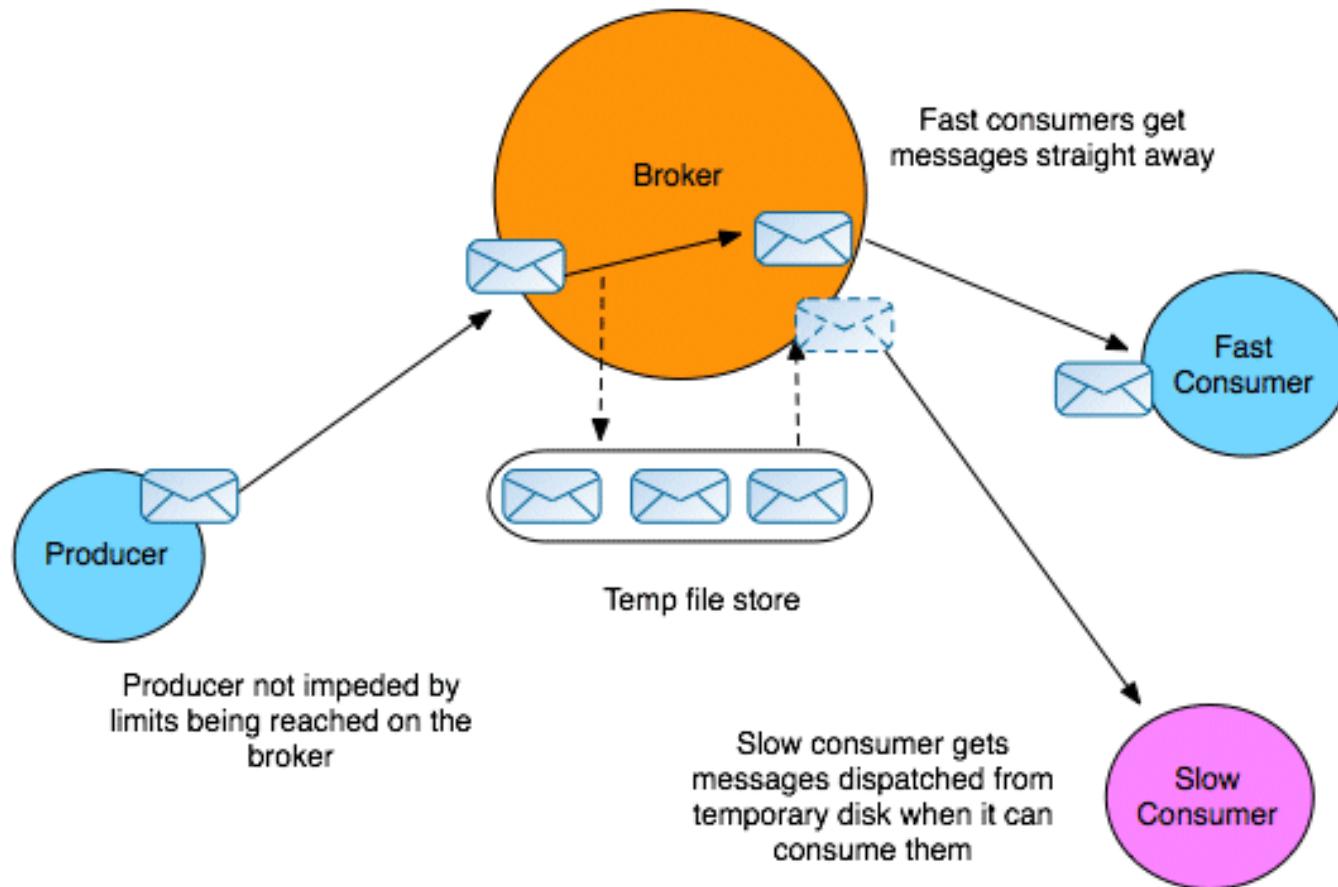


How EDA extends SOA and why it is important

<http://soa-eda.blogspot.pt/2006/11/how-eda-extends-soa-and-why-it-is.html>



A solution is a broker

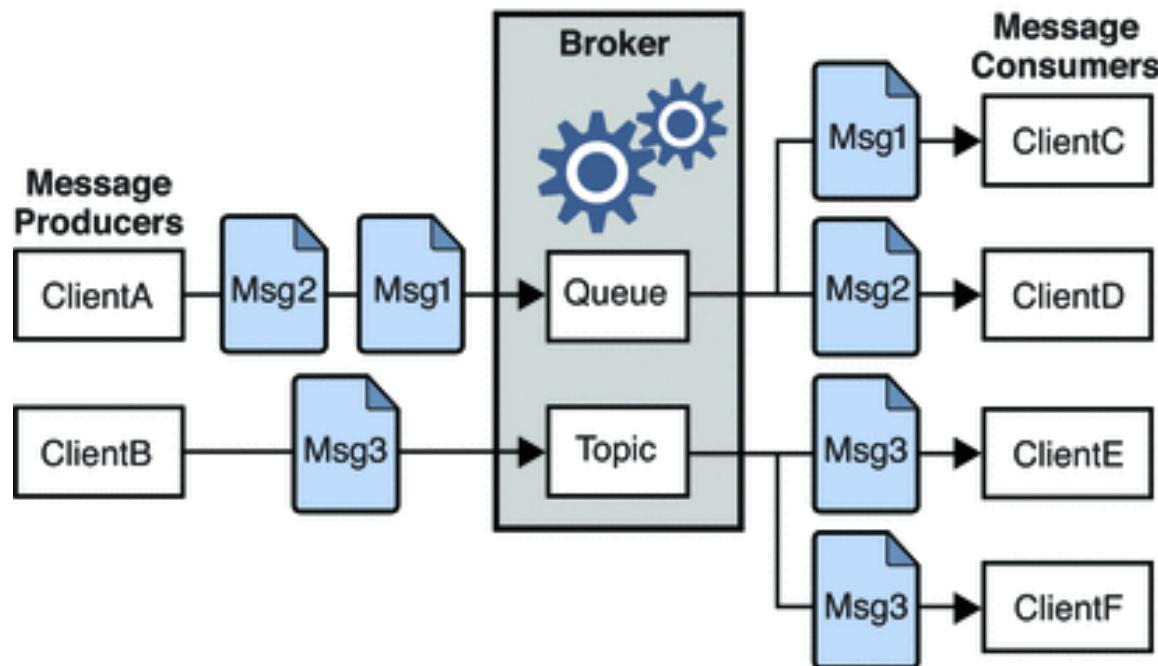


A solution is a broker

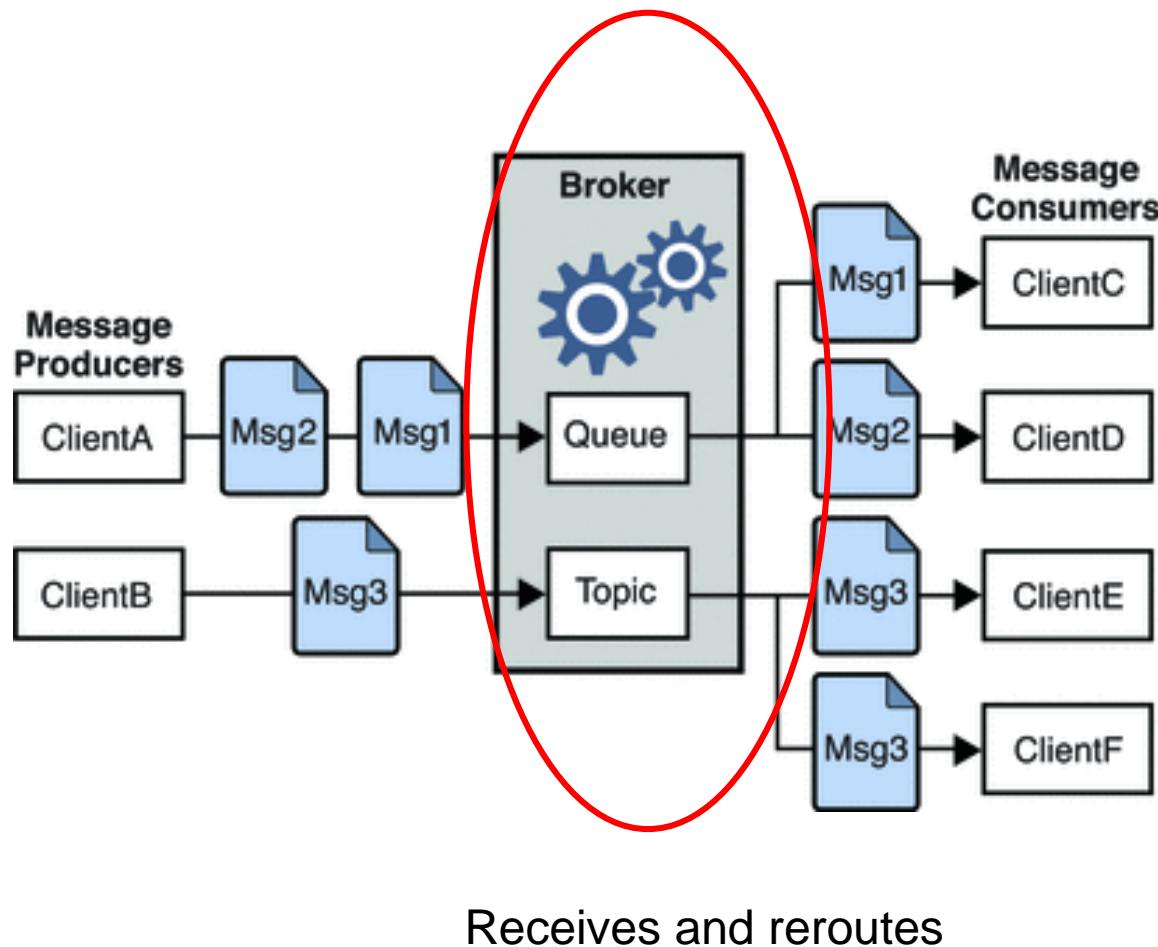
- Receives
 - Input
- Re routes
 - Transport
- Deliver
 - Finds



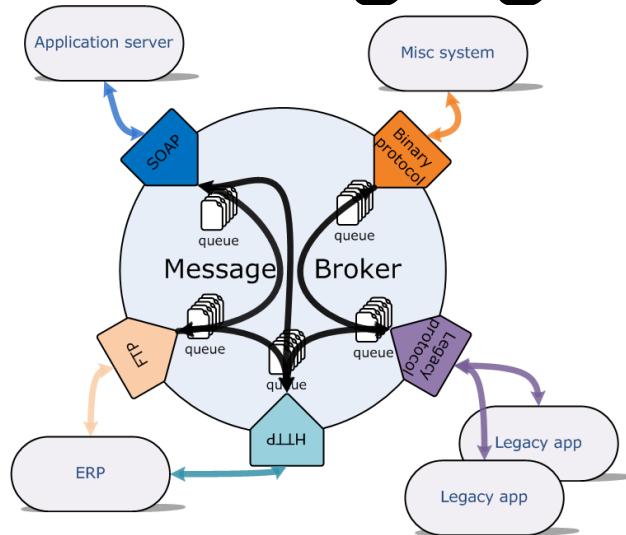
Brokers, Messages, queues, topics ...



Brokers, Messages, queues, topics ...



Messaging as a integration solution



QoS 0
at most once

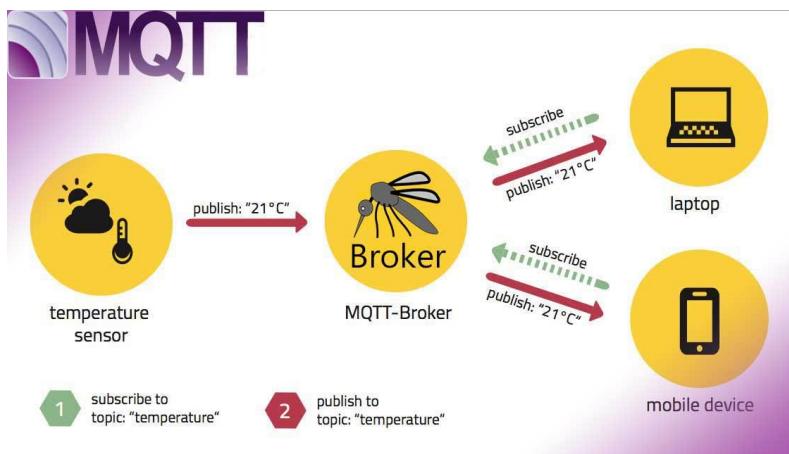
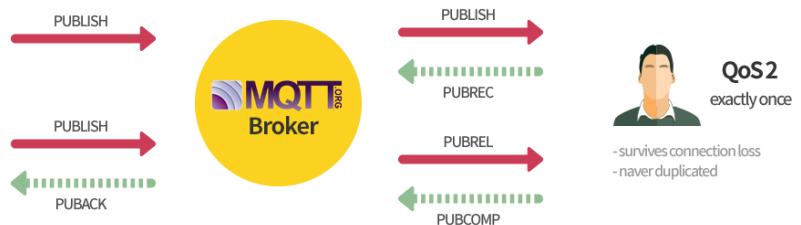
- doesn't survive failures
- never duplicated



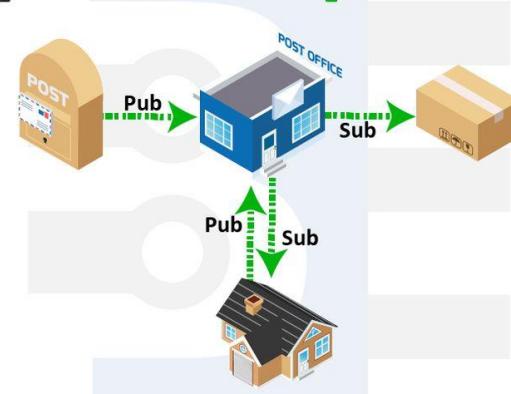
QoS 1
at least once

- survives connection loss
- can be duplicated

Quality of Service for reliable messaging



`mqtt://broker/topic/message`





Wiki Loves Monuments: Photograph a monument, help Wikipedia and win!



Message broker

From Wikipedia, the free encyclopedia

A **message broker** (also known as an **integration broker** or **interface engine**^[1]) is an intermediary computer program module that translates a message from the formal messaging protocol of the sender to the formal messaging protocol of the receiver. Message brokers are elements in telecommunication or computer networks where software applications communicate by exchanging formally-defined messages.^[1] Message brokers are a building block of message-oriented middleware (MOM) but are typically not a replacement for traditional middleware like MOM and remote procedure call (RPC).^{[2][3]}

Contents [hide]

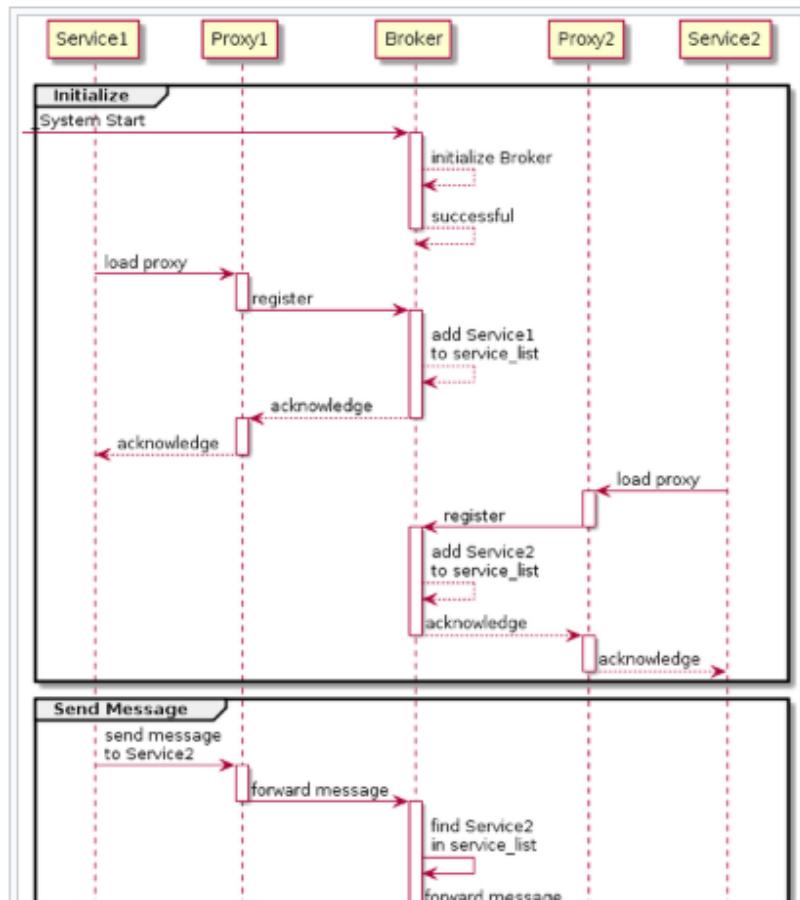
- 1 Purpose, functionality, and architecture
- 2 List of message broker software
- 3 See also
- 4 References

Purpose, functionality, and architecture [edit]

A message broker is an architectural pattern for message validation

https://en.wikipedia.org/wiki/Message_broker

minimizing the mutual awareness that applications should have of each





Wiki Loves Monuments: Photograph a monument, help Wikipedia and win!



X

Message broker

From Wikipedia, the free encyclopedia

A **message broker** (also known as an **integration broker** or **interface engine**^[1]) is an intermediary computer program module that translates a message from the formal messaging protocol of the sender to the formal messaging protocol of the receiver. Message brokers are elements in telecommu

communic
brokers are
typically no
procedure

A message broker is an architectural pattern for message validation, transformation, and routing. It mediates communication among applications, minimizing the mutual awareness that applications should have of each other in order to be able to exchange messages, effectively implementing decoupling.^[4]

1 Purpos

2 List of

3 See als

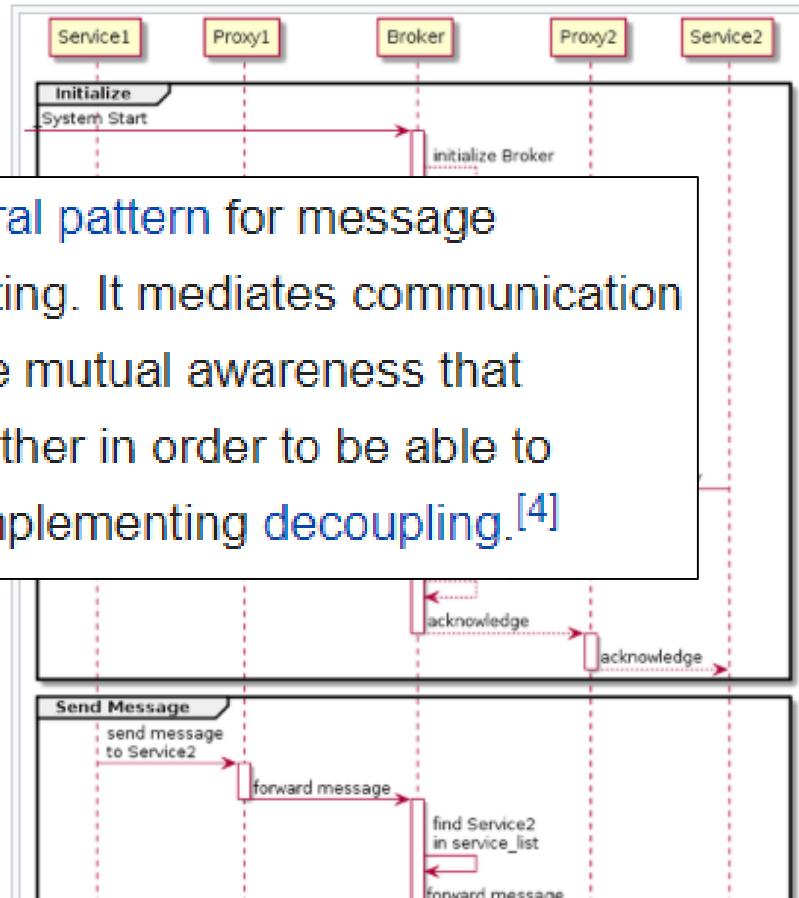
4 References

Purpose, functionality, and architecture [edit]

A message broker is an architectural pattern for message validation

https://en.wikipedia.org/wiki/Message_broker

minimizing the mutual awareness that applications should have of each



Messaging and protocols...

The screenshot shows a Wikipedia page titled "Message queue". The page has a navigation bar at the top with tabs for Article, Talk, Read, Edit, View history, and Search. The main content area starts with a section titled "See also" with a link to "[edit]". Below this, there are two columns of links. The left column is labeled "From W" and the right column is labeled "In com". Both columns list various messaging protocols and services, such as Advanced Message Queuing Protocol, Amazon Simple Queue Service, Celery Task Queue, Gearman, IBM WebSphere Message Broker, IBM WebSphere MQ, IronMQ, Java Message Service, Message-oriented middleware, Microsoft Message Queuing (MSMQ), Microsoft Azure Services Platform, QDB queues, RabbitMQ, StormMQ, ØMQ, SnakeMQ, and HornetQ.

/

http://en.wikipedia.org/wiki/Message_queue

http://en.wikipedia.org/wiki/Advanced_Message_Queueing_Protocol

Messaging and protocols

- AMQP, which stands for Advanced Message Queuing Protocol
- MQTT (Message Queue Telemetry Transport)
- STOMP (Simple/Streaming Text Oriented Messaging Protocol)
- CoAP (Constrained Application Protocol)



IoT Protocol Wars: MQTT vs CoAP vs ...

MQTT

- Field devices with cellular or satellite backhaul – every Kb matters, traffic is expensive.
- Two-way communications over unreliable networks.
- Battery powered devices with low power consumption.
- Devices may sleep, but not 95% of the time. Otherwise please see MQTT-S or CoAP.
- NAT traversal to be addressed as an afterthought – important, but not critical.

MQTT-S (over UDP)

- Pretty much the same as MQTT, but with really sleepy devices (sensor networks).
- Potential to scale 10x times more devices comparing to MQTT – question about UDP vs TCP scalability.
- NAT traversal might become a larger issue here comparing to MQTT. So must be addressed during the planning stage.

CoAP

- Pretty similar to MQTT-S choice, but with several additions.
- Web services oriented architecture, rather than messaging oriented.
- When you're building a platform for the open development communities in the internet space.

REST API

- One-way communications from devices to the cloud.
- NAT traversal around the globe is among the top priorities.



MQTT: just a mention

- “*MQTT is a machine-to-machine (M2M)/"Internet of Things" connectivity protocol. It was designed as an extremely **lightweight publish/subscribe messaging** transport. It is useful for connections with remote locations where a small code footprint is required and/or network bandwidth is at a premium*”
- <http://mqtt.org/>
- *Several brokers, low fingerprint*
 - *Mosquitto most popular*
 - *Popular in IoT*
- *Several drivers & integrations*
 - ActiveMQ , RabbitMQ, Spring, ...



News D

MQTT is a machine-to-machine (M2M)/"Internet of Things" messaging protocol based on an extremely lightweight publish/subscribe messaging transport. It has been designed for use in locations where a small code footprint is required and/or network bandwidth is at a premium. It has been used in sensors communicating to a broker via GPRS, with healthcare providers, and in a range of home automation applications because of its small size, low power consumption and the distribution of information to one or many receivers ([more](#))

News

Servers/Brokers

MQTT v3.1.1 now an OASIS Standard

November 7th, 2014 - [5 Comments](#)

<https://github.com/mqtt/mqtt.github.io/wiki/standard>

Servers

Good news everyone! MQTT v3.1.1 has now become

This marks not just the result of 18 months hard work, but also 15 years of work started by Andy and Andy. You can

You can find the standard specification as either [single file](#) or [zip](#)

Interoperability testing at EclipseCon 2014

January 22nd, 2014 - [No Comments](#)

The [Eclipse Paho](#) project is rapidly becoming a source of MQTT brokers. It contains implementations in C, Java, Javascript, Python, Lua, C++, embedded/minimal C, Go... and an Object Oriented Broker. The popular mosquitto broker recently moved under the [project](#). The project contains both mosquitto, and a fully open-source MQTT broker which also happens to support MQTT-SN.

Server	QoS 0	QoS 1	QoS 2	auth	bridge	\$SYS	SSL	dynamic topics	cluster
Zlemetry	✓	✓	✓	✓	✓	§	✓	✓	✓
Apache ActiveMQ	✓	✓	✓	✓	X	X	✓	✓	✓
Apache Apollo	✓	✓	✓	✓	X	X	✓	✓	?
Bevywise IoT Platform	✓	✓	✓	✓	rm	✓	✓	✓	✓
emitter	✓	§	X	✓	X	X	✓	✓	✓
emqttd	✓	✓	✓	✓	✓	✓	✓	✓	✓
GnatMQ	✓	✓	✓	✓	X	X	X	✓	X
Eclipse	✓	✓	✓	✓	X	✓	✓	✓	X
HiveMQ	✓	✓	✓	✓	✓	✓	✓	✓	✓
IBM MessageSight	✓	✓	✓	✓	X	✓	✓	✓	§
JoramMQ	✓	✓	✓	✓	✓	✓	✓	✓	✓
Longdoz	✓	?	?	?	?	?	?	?	?
mosquette	✓	✓	✓	✓	?	?	✓	?	X
mosca	✓	✓	X	✓	?	?	?	?	?
mosquitto	✓	✓	✓	✓	✓	✓	✓	✓	§
MOTT is	✓	✓	✓	§	X	X	✓	✓	X
PubSub+ (PubSubHubBuddy)	✓	✓	✓	✓	✓	✓	✓	✓	✓
Paho	✓	✓	✓	✓	✓	✓	X	✓	X
Software AG Universal Messaging	✓	✓	✓	✓	X	X	✓	✓	✓
Solace	✓	✓	X	✓	§	✓	✓	✓	✓
Trafero Tstack	✓	✓	✓	✓	X	X	✓	✓	X
VerneMQ	✓	✓	✓	✓	✓	✓	✓	✓	✓
WebSphere MQ	✓	✓	✓	✓	✓	✓	✓	✓	?





List of message broker software [edit]



- Amazon Web Services (AWS) Simple Queue Service (SQS)

- Apache ActiveMQ

- Apache Kafka

- Apache Qpid

- Celery

- Cloverleaf ([E-Novation Lifeline](#))

- Converse Message Broker ([Converse Technology](#))

- Enduro/X Transactional Message Queue (TMQ)

- Financial Fusion Message Broker ([Sybase](#))

- Fuse Message Broker ([enterprise ActiveMQ](#))

- Gearman

- HornetQ ([Red Hat](#))

- IBM Integration Bus

- IBM Message Queues / IBM WebSphere MQ

- JBoss Messaging ([JBoss](#))

- JORAM

- Microsoft Azure Service Bus ([Microsoft](#))

- Microsoft BizTalk Server ([Microsoft](#))

- NATS ([MIT Open Source License](#), written in Go)

- Open Message Queue

- Oracle Message Broker ([Oracle Corporation](#))

Message

From Wikipedia, the free encyclopedia

A **message broker** [1] is an application that receives a message from the producer, converts it to another messaging protocol, and then forwards it to the consumer. Telecommunications companies communicate by brokers. Brokers are built typically not as a separate procedure call (F

Contents

- 1 Purpose, function
- 2 List of message brokers
- 3 See also
- 4 References

Purpose, function

A message broker minimizes the number of components in a system.

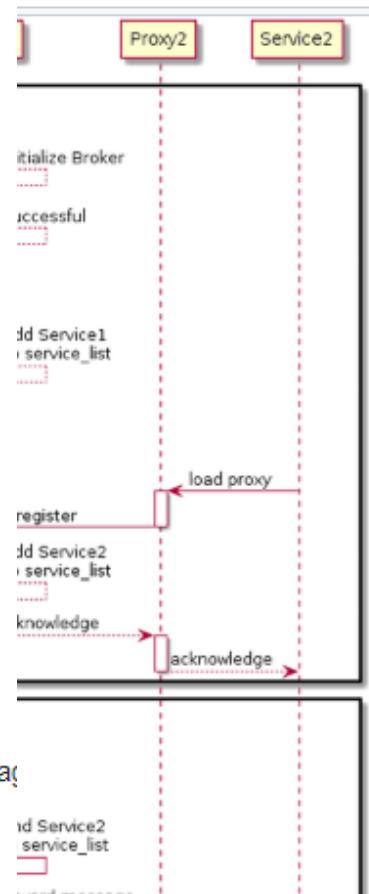
https://en.wikipedia.org/w/index.php?title=Message_broker&oldid=900000000

- RabbitMQ ([Mozilla Public License](#), written in Erlang)

- Redis An open source, in-memory data structure store, used as a database, cache and message broker.

- SAP PI ([SAP AG](#))

- Solace PubSub+



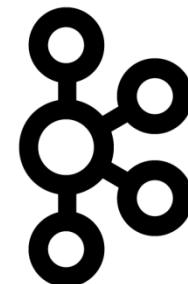
Messaging brokers

ActiveMQ



logstash

 redis

APACHE  kafka®


mosquitto



Messaging Protocols & clients

	ActiveMQ	Apollo	HornetQ	Qpid	RabbitMQ	ZeroMQ
AMQP	1.0	1.0	announced	1.0	0-8, 0-9, 0-9-1	-
MQTT	✓	✓	-	-	✓	-
OpenWire	✓	✓	-	-	-	-
REST	✓	✓	✓	-	✓	-
STOMP	✓	✓	✓	-	✓	-
STOMP over Websockets	✓	✓	✓	-	✓	-
XMPP	✓	-	-	-	Over Gateway	-

Table 1: Support for Messaging Protocols

ActiveMQ, Qpid, HornetQ and RabbitMQ in Comparison

<https://www.predic8.com/activemq-hornetq-rabbitmq-apollo-qpid-comparison.htm>

Messaging Protocols & clients

	ActiveMQ	Apollo	HornetQ	Qpid	RabbitMQ	ZeroMQ	
AMQP	1.0	1.0	announced	1.0	0-8, 0-9, 0-9-1	-	
MQTT	✓	✓	-	-	✓	-	
OpenWire		ActiveMQ	Apollo	HornetQ	Qpid	RabbitMQ	ZeroQ
REST	C	✓	-	-	✓	✓	✓
STOMP	C++	-	-	-	✓	✓	✓
STOMP over	Erlang	-	-	-	-	✓	✓
XMPP	Haskell	-	-	-	-	✓	✓
Table 1: Summary of messaging protocols and clients supported by various brokers.	Java JMS	✓	-	✓	✓	-	-
	Java proprietary	✓	-	✓	-	✓	✓
	.NET	-	-	-	✓	✓	✓
	Objective-C	-	-	-	-	-	✓
	Perl	-	-	-	-	✓	✓
	PHP	-	-	-	-	✓	✓
	Python	-	-	-	✓	✓	✓
	Ruby	-	-	-	✓	✓	✓

ActiveMQ, Qpid, HornetQ and RabbitMQ in Comparison

<https://www.predic8.com/activemq-hornetq-rabbitmq-apollo-qpid-comparison.htm>

Some examples: just wildfly



DISTRIBUTION	BROKER	VERSION	PROTOCOL SUPPORT
Artemis standalone	Artemis	1.5	AMQP, STOMP, MQTT, OpenWire, HornetQ Core
ActiveMQ 5 standalone	ActiveMQ 5	5.14	AMQP, STOMP, MQTT, OpenWire
WildFly 8, 9	HornetQ	2.4	AMQP, STOMP, HornetQ Core
JBoss EAP 6.4	HornetQ	2.3	AMQP, STOMP, HornetQ Core
WildFly 10	Artemis	1.1	AMQP, STOMP, HornetQ Core
JBoss EAP 7	Artemis	1.1	HornetQ Core

<https://blog.akquinet.de/2017/02/22/activemq-confusion-and-what-comes-with-your-jboss-eap-wildfly/>

Brokered solutions

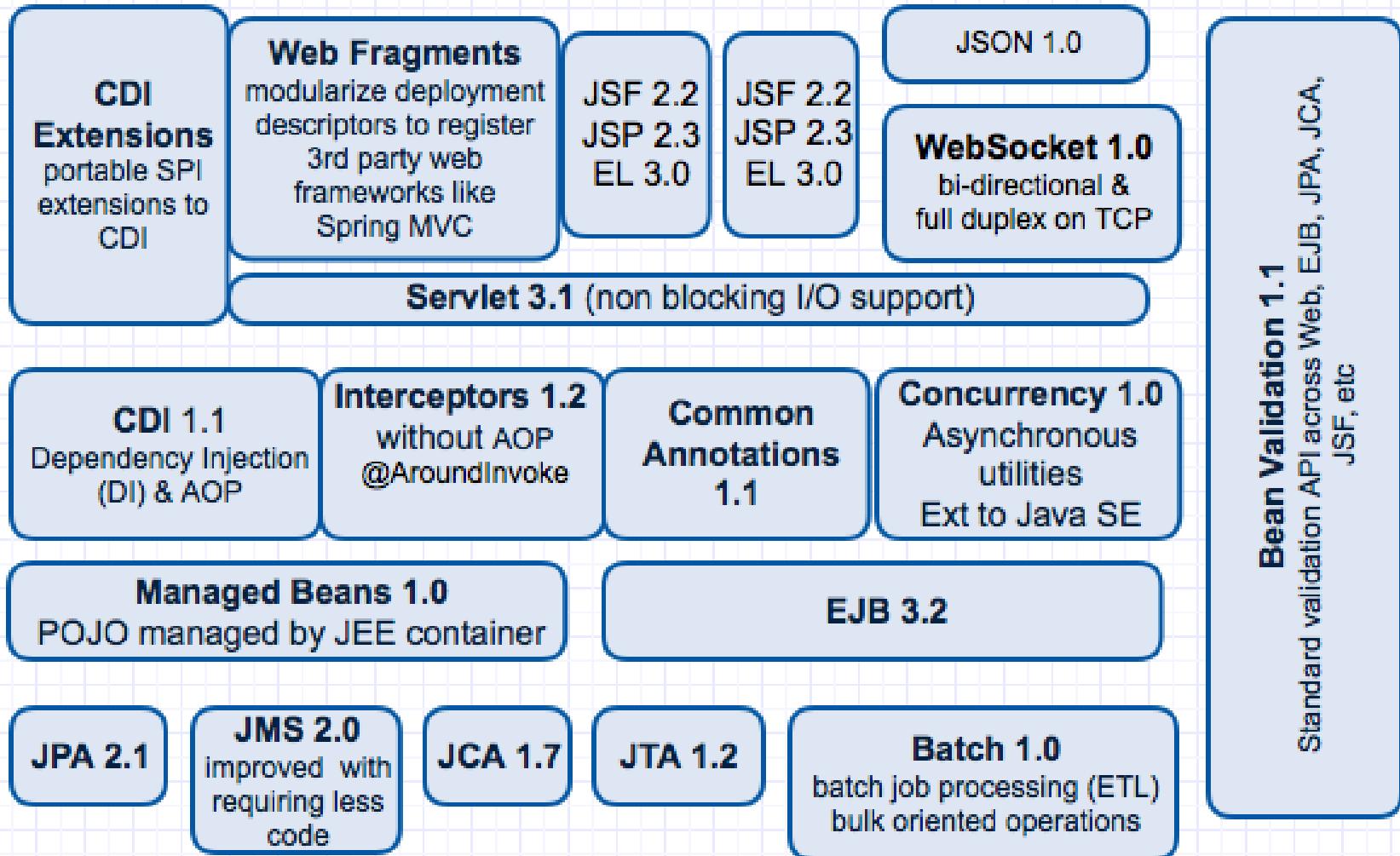
- Several options
 - **EAI and SOA platforms**, such as [IBM WebSphere MQ](#), [TIBCO](#), [Vitria](#), [Oracle Service Bus](#), [WebMethods](#) (now Software AG), [Microsoft BizTalk](#), or [Fiorano](#).
 - **Open source ESB's** like [Mule ESB](#), [JBoss Fuse](#), [Open ESB](#), [WSo2](#), [Spring Integration](#), or [Talend ESB](#)
 - **Message Brokers** like [ActiveMQ](#), [Apache Kafka](#), or [RabbitMQ](#)
 - **Web service- or REST-based integration**, including [Amazon Simple Queue Service \(SQS\)](#) or [Google Cloud Pub/Sub](#)
 - **JMS-based messaging systems**
 - **Microsoft technologies** like [MSMQ](#) or [Windows Communication Foundation \(WCF\)](#)

<http://www.enterpriseintegrationpatterns.com/patterns/messaging/index.html>

Java Messaging system (JMS): a high level messaging endpoint

Jfernан@ua.pt

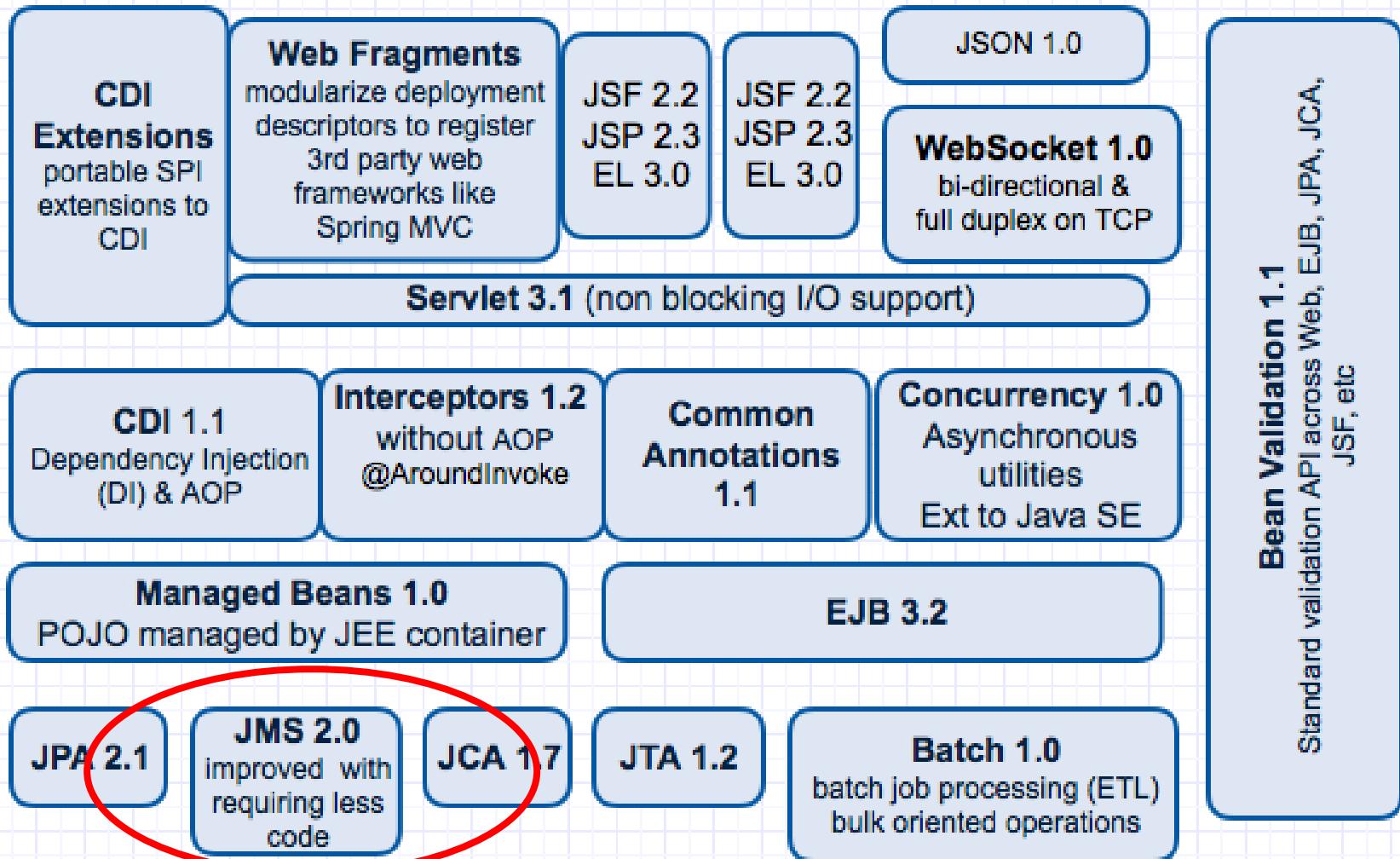
JEE 7.0 Technologies



♥ ♦ 8 JEE (i.e. Java EE) Overview interview questions & answers

<https://www.java-success.com/jee-overview-interview-questions-and-answers/>

JEE 7.0 Technologies



♥ ♦ 8 JEE (i.e. Java EE) Overview interview questions & answers

<https://www.java-success.com/jee-overview-interview-questions-and-answers/>

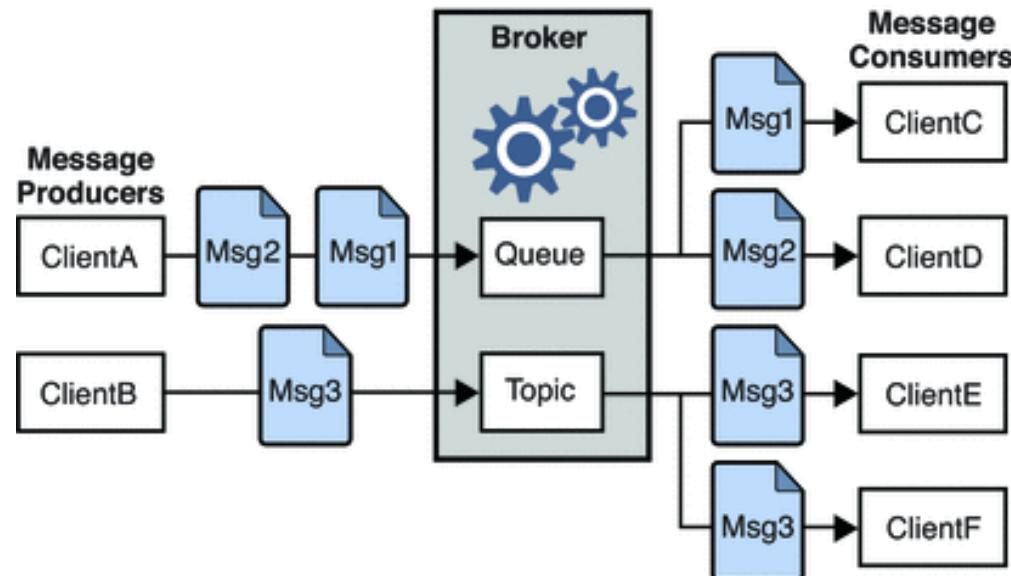
Java Messaging Service

Java Message Service

From Wikipedia, the free encyclopedia

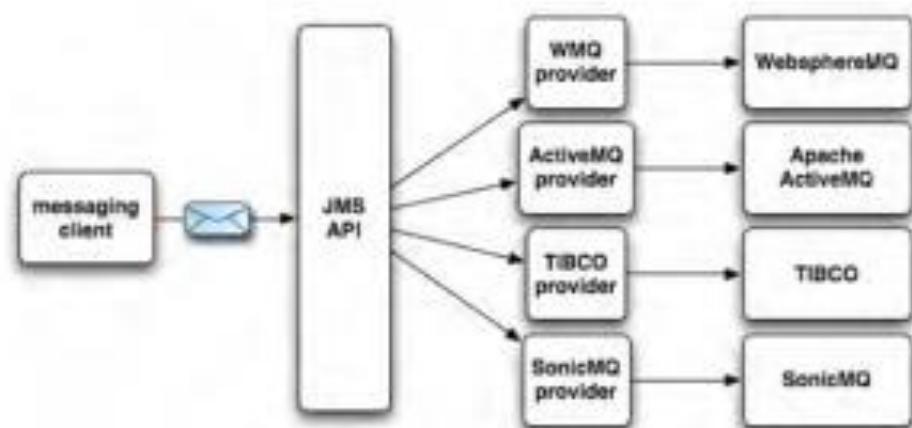
The Java Message Service (JMS) API is a Java Message Oriented Middleware (MOM) API^[1] for sending messages between two or more clients. JMS is a part of the Java Platform, Enterprise Edition, and is defined by a specification developed under the Java Community Process as JSR 914.^[2] It is a messaging standard that allows application components based on the Java Enterprise Edition (Java EE) to create, send, receive, and read messages. It allows the communication between different components of a *distributed application* to be loosely coupled, reliable, and asynchronous.^[3]

Contents [\[hide\]](#)



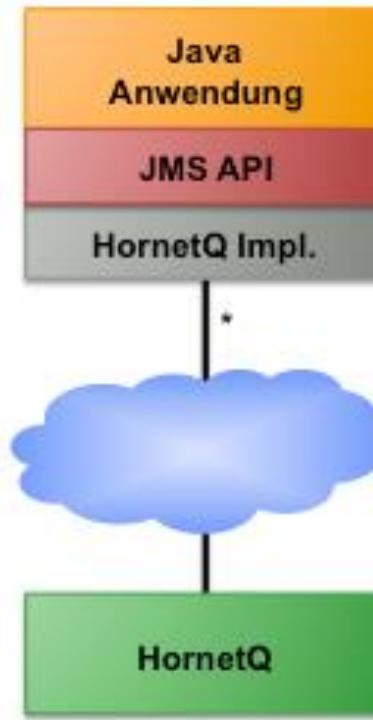
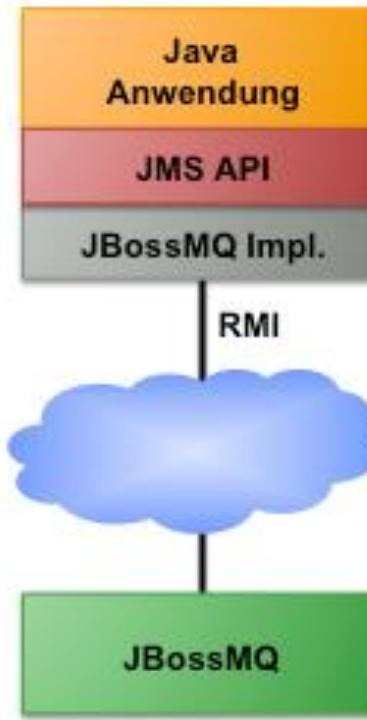
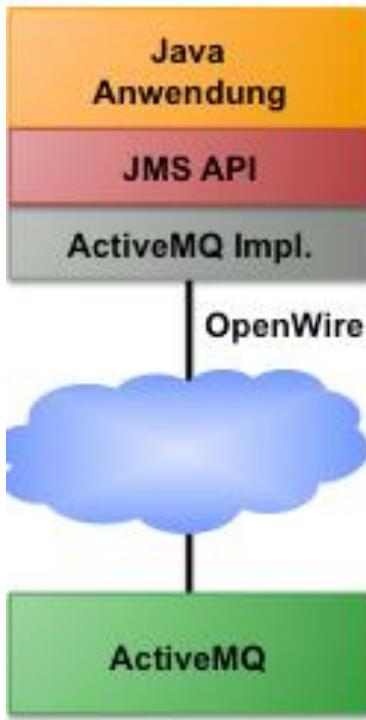
JMS: middleware with concrete roles

- **JMS client —**
 - An application using Java to send and receive messages.
- **Non-JMS client —**
 - An application a JMS provider's native API to send and receive messages
- **JMS producer —**
 - A client application that creates and sends JMS messages.
- **JMS consumer —**
 - A client application that receives and processes JMS messages.
- **JMS provider —**
 - The implementation of the JMS interfaces.,.
- **JMS message —**
 - The most fundamental concept of JMS;
- **JMS domains —**
 - include point-to-point and publish/subscribe.
- Others...



<https://anwaarlabs.wordpress.com/2014/04/28/message-queue-part-2-jms/>

JMS



<https://www.predic8.com/activemq-hornetq-rabbitmq-apollo-qpidd-comparison.htm>

Java Messaging Service

Provider implementations [edit]

To use JMS, one must have a JMS provider that can manage the sessions and queues. Starting from Java EE version 1.4, JMS provider has to be contained in all Java EE application servers. This can be implemented using the message inflow management of the Java EE Connector Architecture, which was first made available in that version.

Java M

From Wikipedia

The Java Me part of the Java standard that communicates

Conti

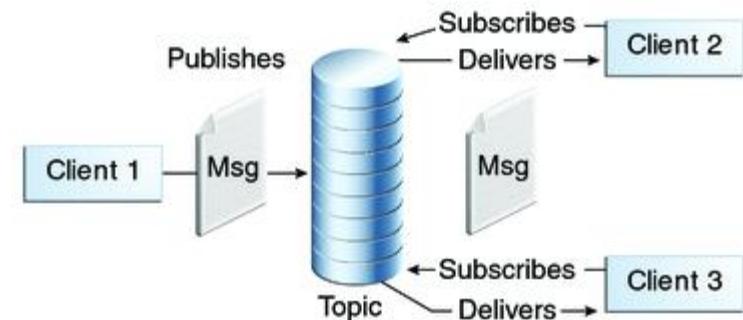
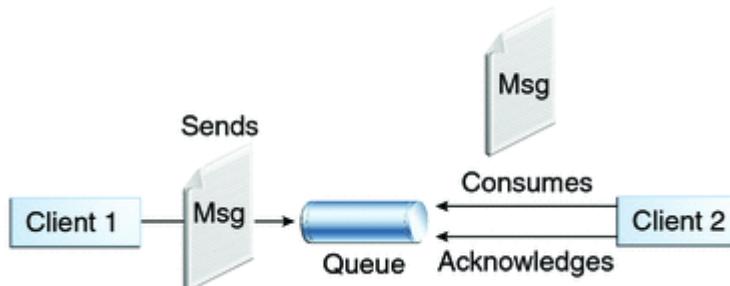
The following is a list of JMS providers:

- Apache ActiveMQ
- Apache Qpid, using AMQP^[5]
- Oracle Weblogic (part of the Fusion Middleware suite) and Oracle AQ from Oracle
- EMS from TIBCO
- FFMQ, GNU LGPL licensed
- JBoss Messaging and HornetQ from JBoss
- JORAM, from the OW2 Consortium
- Open Message Queue, from Oracle
- OpenJMS, from The OpenJMS Group
- Solace JMS from Solace Systems
- RabbitMQ by Rabbit Technologies Ltd., acquired by SpringSource
- SAP Process Integration ESB
- SonicMQ from Progress Software
- SwiftMQ
- Tervela
- Ultra Messaging from 29 West (acquired by Informatica)
- webMethods from Software AG
- WebSphere Application Server from IBM, which provides an inbuilt default messaging provider known as the Service Integration Bus (SIBus), or which connects to WebSphere MQ as a JMS provider^[6]
- WebSphere MQ (formerly MQSeries) from IBM
- FioranoMQ

A historical comparison matrix of JMS providers from 2005 is available at <http://www.theserverside.com/reviews/matrix.tss> ↗
http://javakeexample.blogspot.in/2012/12/integrating-spring-with-jms_19.html ↗

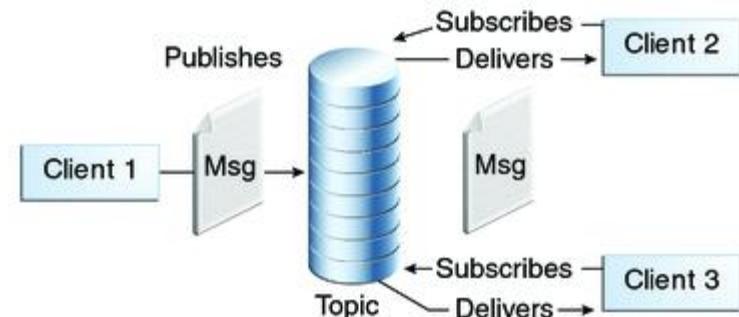
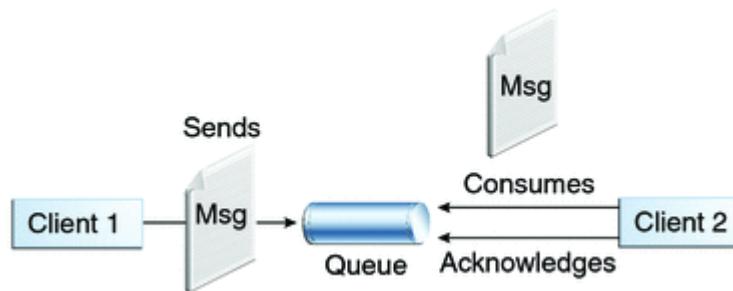
JMS supports basic patterns

- **Point-to-Point
Messaging Domain**
- **Publish/Subscribe
Messaging Domain**



Getting the JMS resources

- Point-to-Point Messaging Domain
- Publish/Subscribe Messaging Domain



```
@Resource(lookup = "jms/ConnectionFactory")
private static ConnectionFactory connectionFactory;

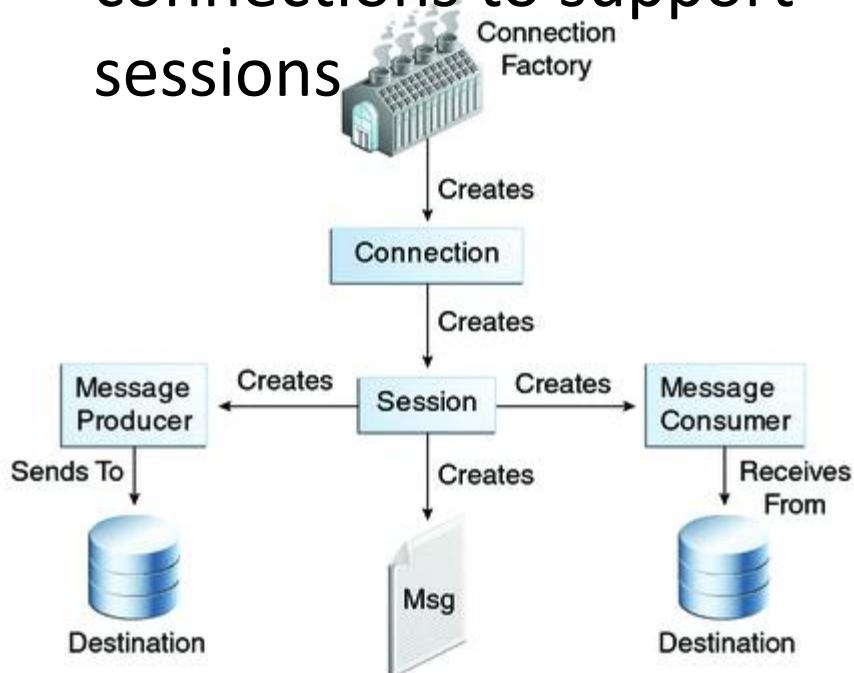
@Resource(lookup = "jms/Queue")
private static Queue queue;
```

```
@Resource(lookup = "jms/ConnectionFactory")
private ConnectionFactory connectionFactory;

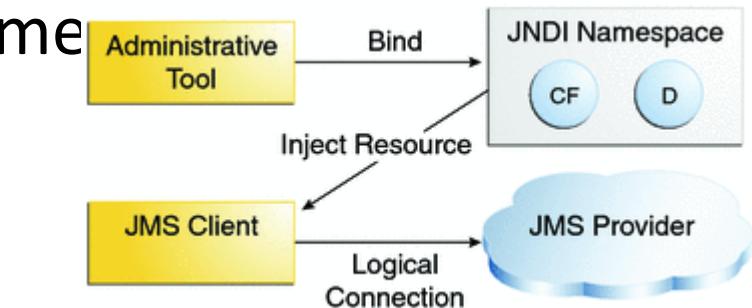
@Resource(lookup = "jms/Topic")
private Topic topic;
```

JMS hides connection management

- The factory creates connections to support sessions

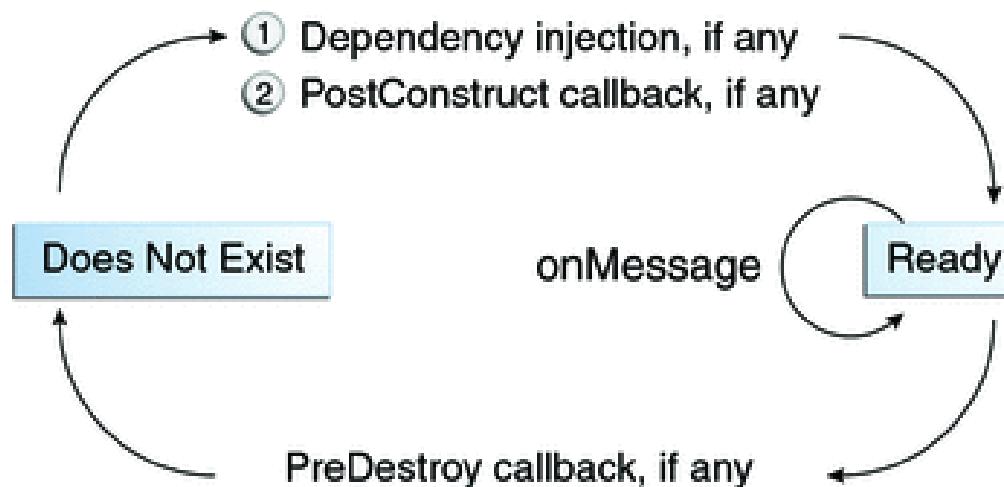


- JMS binding are kept in a JNDI namespace to help JMS clients find the JMS provider



Message Driven Bean (MDB)

- Hides the complexity
- Handles to listen to the Java Message Service API



MDB: A simple example

- The application must perform:
 - Creating a connection and a session
 - Creating message producers and consumers
 - Sending and receiving messages

A Message Driven Bean (MDB)...

```
1 package test;
2
3 import java.util.logging.Logger;
4 import javax.ejb.ActivationConfigProperty;
5 import javax.ejb.MessageDriven;
6 import javax.jms.JMSException;
7 import javax.jms.Message;
8 import javax.jms.MessageListener;
9
10 @MessageDriven(mappedName="testQueue2",
11     activationConfig = { @ActivationConfigProperty(
12         propertyName="destinationType", propertyValue="javax.jms.Queue")})
13 public class MessageBean implements MessageListener {
14     Logger logger = Logger.getLogger("test");
15
16     public void onMessage(Message msg) {
17         try {
18             String name = msg.getStringProperty("name");
19             logger.info("Received msg " + msg + ", from " + name);
20         } catch (JMSException e) {
21             throw new RuntimeException(e);
22         }
23     }
24 }
```

From <http://javahowto.blogspot.pt/2010/04/message-driven-bean-example-with.html>

Is like a Server managed callback

```
1 package test;
2
3 import java.util.logging.Logger;
4 import javax.ejb.ActivationConfigProperty;
5 import javax.ejb.MessageDriven;
6 import javax.jms.JMSException;
7 import javax.jms.Message;
8 import javax.jms.MessageListener;
9
10 @MessageDriven(mappedName="testQueue2",
11     activationConfig = { @ActivationConfigProperty(
12         propertyName="destinationType", propertyValue="javax.jms.Queue")})
13 public class MessageBean implements MessageListener {
14     Logger logger = Logger.getLogger("test");
15
16     public void onMessage(Message msg) {
17         try {
18             String name = msg.getStringProperty("name");
19             logger.info("Received msg " + msg + ", from " + name);
20         } catch (JMSException e) {
21             throw new RuntimeException(e);
22         }
23     }
24 }
```

```
1 ./asadmin create-jms-resource --restype javax.jms.Queue testQueue2
2 ./asadmin create-jms-resource --restype javax.jms.QueueConnectionFactory jms/QueueConnectionFactory
```

From <http://javahowto.blogspot.pt/2010/04/message-driven-bean-example-with.html>

MDB can be requested

```
9
10 @WebServlet(urlPatterns = "/MessageServlet")
11 public class MessageServlet extends HttpServlet {
12     @Resource(mappedName = "testQueue2")
13     private Queue queue;
14
15     @Resource(mappedName = "jms/QueueConnectionFactory")
16     private QueueConnectionFactory queueConnectionFactory;
17
18     @Override
19     protected void doGet(HttpServletRequest request, HttpServletResponse response)
20         throws ServletException, IOException {
21         QueueConnection queueConnection = null;
22         try {
23             queueConnection = queueConnectionFactory.createQueueConnection();
24             queueConnection.start();
25             QueueSession queueSession = queueConnection.createQueueSession(false,
26                             Session.AUTO_ACKNOWLEDGE);
27             QueueSender sender = queueSession.createSender(queue);
28
29             TextMessage msg = queueSession.createTextMessage();
30             msg.setText("A message from MessageServlet");
31             msg.setStringProperty("name", "MessageServlet");
32
33             sender.send(msg);
34         } catch (JMSException e) {
35             throw new RuntimeException(e);
36         } finally {
37             try {
38                 if (queueConnection != null) {
39                     queueConnection.close();
40                 }
41             } catch (JMSException e) { //ignore
42             }
43         }
44     }
45 }
```

From <http://javahowto.blogspot.pt/2010/04/message-driven-bean-example-with.html>

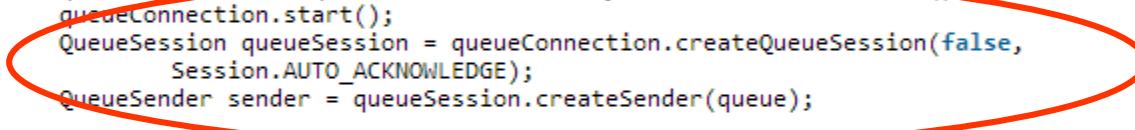
... To the server

```
9
10 @WebServlet(urlPatterns = "/MessageServlet")
11 public class MessageServlet extends HttpServlet {
12     @Resource(mappedName = "testQueue2")
13     private Queue queue;
14
15     @Resource(mappedName = "jms/QueueConnectionFactory")
16     private QueueConnectionFactory queueConnectionFactory;
17
18     @Override
19     protected void doGet(HttpServletRequest request, HttpServletResponse response)
20         throws ServletException, IOException {
21         QueueConnection queueConnection = null;
22         try {
23             queueConnection = queueConnectionFactory.createQueueConnection();
24             queueConnection.start();
25             QueueSession queueSession = queueConnection.createQueueSession(false,
26                             Session.AUTO_ACKNOWLEDGE);
27             QueueSender sender = queueSession.createSender(queue);
28
29             TextMessage msg = queueSession.createTextMessage();
30             msg.setText("A message from MessageServlet");
31             msg.setStringProperty("name", "MessageServlet");
32
33             sender.send(msg);
34         } catch (JMSException e) {
35             throw new RuntimeException(e);
36         } finally {
37             try {
38                 if (queueConnection != null) {
39                     queueConnection.close();
40                 }
41             } catch (JMSException e) { //ignore
42             }
43         }
44     }
45 }
```

From <http://javahowto.blogspot.pt/2010/04/message-driven-bean-example-with.html>

... To the server

```
9
10 @WebServlet(urlPatterns = "/MessageServlet")
11 public class MessageServlet extends HttpServlet {
12     @Resource(mappedName = "testQueue2")
13     private Queue queue;
14
15     @Resource(mappedName = "jms/QueueConnectionFactory")
16     private QueueConnectionFactory queueConnectionFactory;
17
18     @Override
19     protected void doGet(HttpServletRequest request, HttpServletResponse response)
20         throws ServletException, IOException {
21         QueueConnection queueConnection = null;
22         try {
23             queueConnection = queueConnectionFactory.createQueueConnection();
24             queueConnection.start();
25             QueueSession queueSession = queueConnection.createQueueSession(false,
26                 Session.AUTO_ACKNOWLEDGE);
27             QueueSender sender = queueSession.createSender(queue);
28
29             TextMessage msg = queueSession.createTextMessage();
30             msg.setText("A message from MessageServlet");
31             msg.setStringProperty("name", "MessageServlet");
32
33             sender.send(msg);
34         } catch (JMSException e) {
35             throw new RuntimeException(e);
36         } finally {
37             try {
38                 if (queueConnection != null) {
39                     queueConnection.close();
40                 }
41             } catch (JMSException e) { //ignore
42             }
43         }
44     }
45 }
```



From <http://javahowto.blogspot.pt/2010/04/message-driven-bean-example-with.html>

... to pass messages

```
9
10 @WebServlet(urlPatterns = "/MessageServlet")
11 public class MessageServlet extends HttpServlet {
12     @Resource(mappedName = "testQueue2")
13     private Queue queue;
14
15     @Resource(mappedName = "jms/QueueConnectionFactory")
16     private QueueConnectionFactory queueConnectionFactory;
17
18     @Override
19     protected void doGet(HttpServletRequest request, HttpServletResponse response)
20         throws ServletException, IOException {
21         QueueConnection queueConnection = null;
22         try {
23             queueConnection = queueConnectionFactory.createQueueConnection();
24             queueConnection.start();
25             QueueSession queueSession = queueConnection.createQueueSession(false,
26                 Session.AUTO_ACKNOWLEDGE);
27             QueueSender sender = queueSession.createSender(queue);
28
29             TextMessage msg = queueSession.createTextMessage();
30             msg.setText("A message from MessageServlet");
31             msg.setStringProperty("name", "MessageServlet");
32
33             sender.send(msg);
34         } catch (JMSException e) {
35             throw new RuntimeException(e);
36         } finally {
37             try {
38                 if (queueConnection != null) {
39                     queueConnection.close();
40                 }
41             } catch (JMSException e) { //ignore
42             }
43         }
44     }
45 }
```

From <http://javahowto.blogspot.pt/2010/04/message-driven-bean-example-with.html>

... That will “trigger” the MDB

```
9
10 @WebServlet(urlPatterns = "/MessageServlet")
11 public class MessageServlet extends HttpServlet {
12     @Resource(mappedName = "testQueue2")
13     private Queue queue,
14
15     @Resource(mappedName = "jms/QueueConnectionFactory")
16     private QueueConnectionFactory queueConnectionFactory;
17
18     @Override
19     protected void doGet(HttpServletRequest request, HttpServletResponse response)
20         throws ServletException, IOException {
21         QueueConnection queueConnection = null;
22         try {
23             queueConnection = queueConnectionFactory.createQueueConnection();
24             queueConnection.start();
25             QueueSession queueSession = queueConnection.createQueueSession(false,
26                 Session.AUTO_ACKNOWLEDGE);
27             QueueSender sender = queueSession.createSender(queue);
28
29             TextMessage msg = queueSession.createTextMessage();
30             msg.setText("A message from MessageServlet");
31             msg.setStringProperty("name", "MessageServlet");
32
33             sender.send(msg);
34         } catch (JMSException e) {
35             throw new RuntimeException(e);
36         } finally {
37             try {
38                 if (queueConnection != null) {
39                     queueConnection.close();
40                 }
41             } catch (JMSException e) { //ignore
42             }
43         }
44     }
45 }
```

From <http://javahowto.blogspot.pt/2010/04/message-driven-bean-example-with.html>



... That will “trigger” the MDB

```
9
10 @WebServlet(urlPatterns = "/MessageServlet")
11 public class MessageServlet extends HttpServlet {
12     @Resource(mappedName = "testQueue2")
13     private Queue queue;
14
15     @Resource(mappedName = "jms/QueueConnectionFactory")
16     private QueueConnectionF
17
18     @Override
19     protected void doGet(Htt
20         throws ServletException
21         QueueConnection queu
22         try {
23             queueConnection
24             queueConnection.
25             QueueSession que
26                 Session.
27                 QueueSender send
28
29                 TextMessage msg
30                 msg.setText("A m
31                 msg.setStringPro
32
33                 sender.send(msg)
34             } catch (JMSException
35             throw new Runtime
36             } finally {
37                 try {
38                     if (queueCon
39                     queueCon
40                 }
41             } catch (JMSException e) { //ignore
42             }
43         }
44     }
45 }
```

```
1 package test;
2
3 import java.util.logging.Logger;
4 import javax.ejb.ActivationConfigProperty;
5 import javax.ejb.MessageDriven;
6 import javax.jms.JMSException;
7 import javax.jms.Message;
8 import javax.jms.MessageListener;
9
10 @MessageDriven(mappedName="testQueue2",
11     activationConfig = { @ActivationConfigProperty(
12         propertyName="destinationType", propertyValue="javax.jms.Queue")})
13 public class MessageBean implements MessageListener {
14     Logger logger = Logger.getLogger("test");
15
16     public void onMessage(Message msg) {
17         try {
18             String name = msg.getStringProperty("name");
19             logger.info("Received msg " + msg + ", from " + name);
20         } catch (JMSException e) {
21             throw new RuntimeException(e);
22         }
23     }
24 }
```

From <http://javahowto.blogspot.pt/2010/04/message-driven-bean-example-with.html>

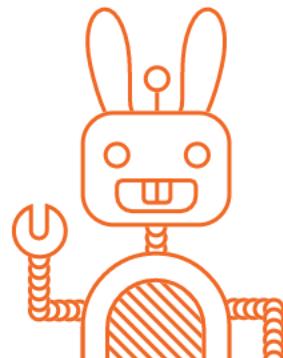
Note...

- This is a snippet
- Configurations, deployments , ...
- Several subjects not described
 - Transactions
 - Quality of service
 - Connection type
 - Synchronous / asynchronous
 - ...

JMS is a interface for listener

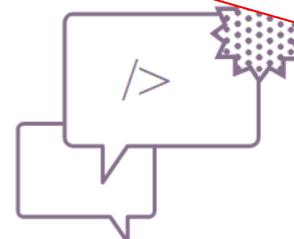
- Need to use a “real broker”
- Application servers have embedded brokers
 - several options
 - Some information not available/ clear
 - No full compatibility

CAN BE USEFULL IN SOME PROJECTS



Announcing [RabbitMQ 3.7](#)

More automation and operator friendly than ever before.



RabbitMQ is the most widely deployed open source message broker.

With more than 35,000 production deployments of RabbitMQ world-wide at small startups and large enterprises, RabbitMQ is the most popular open source message broker.

RabbitMQ is lightweight and easy to deploy on premises and in the cloud. It supports multiple messaging protocols. RabbitMQ can be deployed in distributed and federated configurations to meet high-scale, high-availability requirements.

and provides

<https://www.rabbitmq.com/#features>

Updates

1. [RabbitMQ 3.7.3](#) 30 Jan 2018
2. [Impact of Meltdown mitigation on RabbitMQ performance](#) 26 Jan 2018
3. [Securing RabbitMQ against the ROBOT TLS attack](#) 26 Jan 2018

[More updates →](#)

Tweets



RabbitMQ
@RabbitMQ
RabbitMQ 3.7.4 is almost here. Help us test this RC! groups.google.com/forum/#!topic/rabbitmq/RabbitMQ_3.7.4_is_almost_here_Help_us_test_this_RC/...



1 "Hello World!"

The simplest thing that does something



Python | Java | Ruby | PHP | C#

2 Work queues

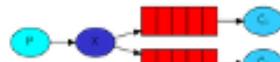
Distributing tasks among workers



Python | Java | Ruby | PHP | C#

3 Publish/Subscribe

Sending messages to many consumers at once



Python | Java | Ruby | PHP | C#

4 Routing

Receiving messages selectively



Python | Java | Ruby | PHP | C#

5 Topics

Receiving messages based on a pattern



Python | Java | Ruby | PHP | C#

6 RPC

Remote procedure call implementation



Python | Java | Ruby | PHP | C#

Rabbit deployment

messaging

With more than 10,000 startups and large enterprises using RabbitMQ as their message broker.

RabbitMQ is lightweight and easy to deploy on premises and in the cloud. It supports multiple messaging protocols. RabbitMQ can be deployed in distributed and federated configurations to meet high-scale, high-availability requirements.

<https://www.rabbitmq.com/#features>

loud environments, and provides

<https://www.rabbitmq.com/devtools.html>

Tweets



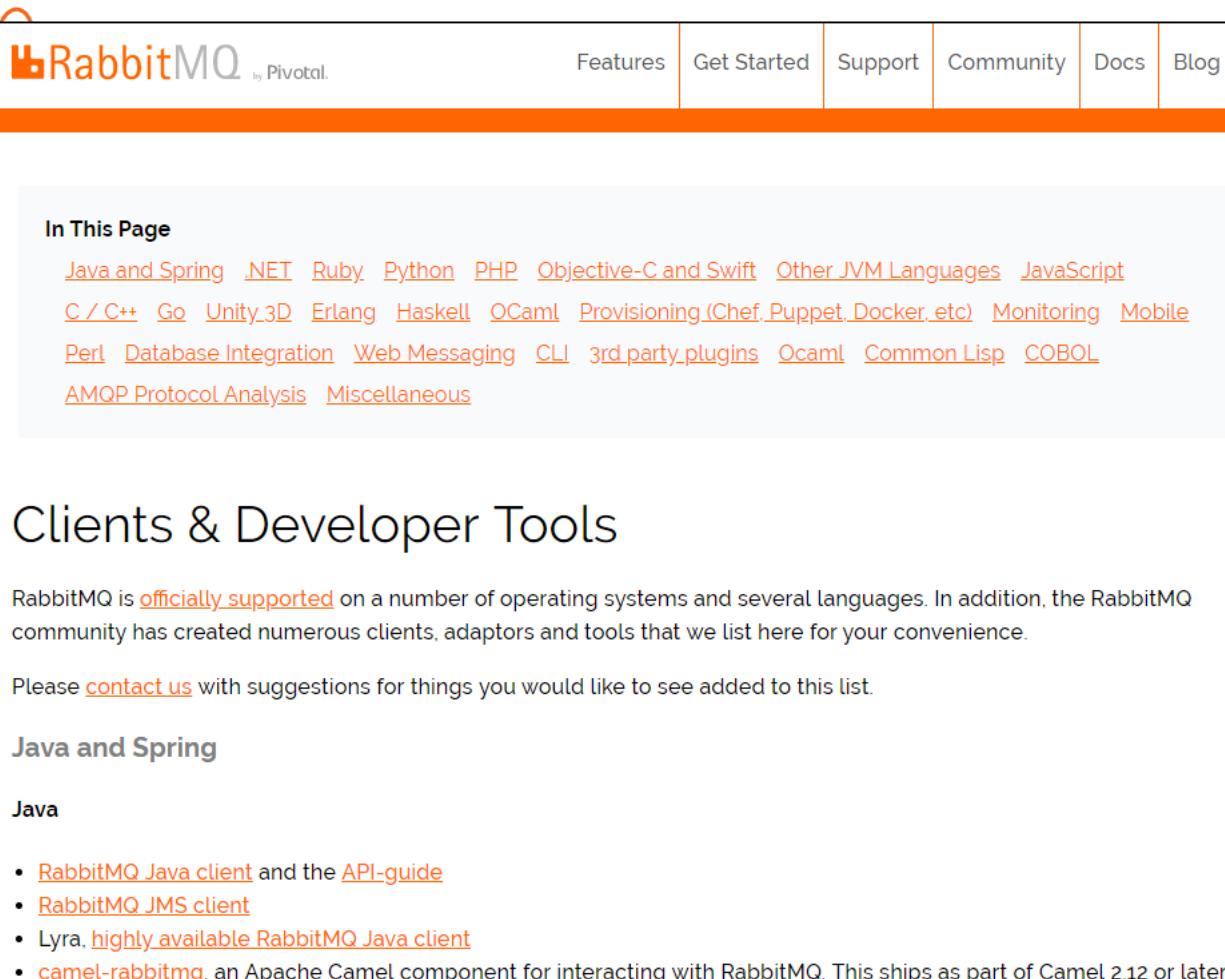
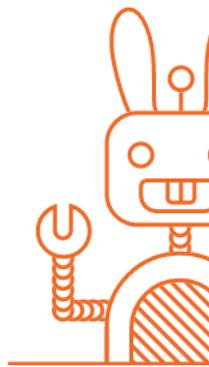
RabbitMQ
@RabbitMQ



RabbitMQ 3.7.4 is almost here. Help us test this RC! [groups.google.com/forum/#topic/...](https://groups.google.com/forum/#topic/)



universidade de aveiro



The screenshot shows the RabbitMQ website's "Clients & Developer Tools" page. On the left, there is a sidebar with the text "RabbitMQ deployed message" and "With more than 35.0 startups and large enterprises using RabbitMQ as their message broker." Below this, it says "RabbitMQ is lightweight, reliable, and supports multiple message formats and federated configurations." To the right, the main content area has a header "In This Page" followed by a list of supported languages and tools: Java and Spring, .NET, Ruby, Python, PHP, Objective-C and Swift, Other JVM Languages, JavaScript, C/C++, Go, Unity 3D, Erlang, Haskell, OCaml, Provisioning (Chef, Puppet, Docker, etc), Monitoring, Mobile, Perl, Database Integration, Web Messaging, CLI, 3rd party plugins, Ocaml, Common Lisp, COBOL, AMQP Protocol Analysis, and Miscellaneous.

<https://www.rabbitmq.com/#features>

<https://www.rabbitmq.com/devtools.html>

loud environments, and provides

this KC: [groups.google.com/forum/#topic/...](https://groups.google.com/forum/#topic/)

Plugins

Overview

RabbitMQ supports plugins. Plugins extend core broker functionality in a variety of ways: with support for more protocols, system state [monitoring](#), additional AMQP 0-9-1 exchange types, node [federation](#), and more. A number of features are implemented as plugins that ship in the core distribution.

This guide covers the plugin mechanism and plugins that ship with RabbitMQ 3.7.8. 3rd party plugins can be installed separately. A set of [curated plugins](#) is also available.

Plugins are activated when a node is started or at runtime when a [CLI tool](#) is used. For a plugin to be activated at boot, it must be enabled. To enable a plugin, use the [rabbitmq-plugins](#):

```
rabbitmq-plugins enable plugin-name
```

And to disable a plugin, use:

```
rabbitmq-plugins disable plugin-name
```

A list of plugins available locally (in the node's [plugins directory](#)) as well as their status (enabled or disabled) can be obtained using :

In This Section

[Server Documentation](#)

[Client Documentation](#)

Plugins

[Management plugin](#)

[Federation plugin](#)

[Shovel plugin](#)

[STOMP plugin](#)

[STOMP-over-
WebSockets](#)

[MQTT plugin](#)

[MQTT-over-
WebSockets](#)

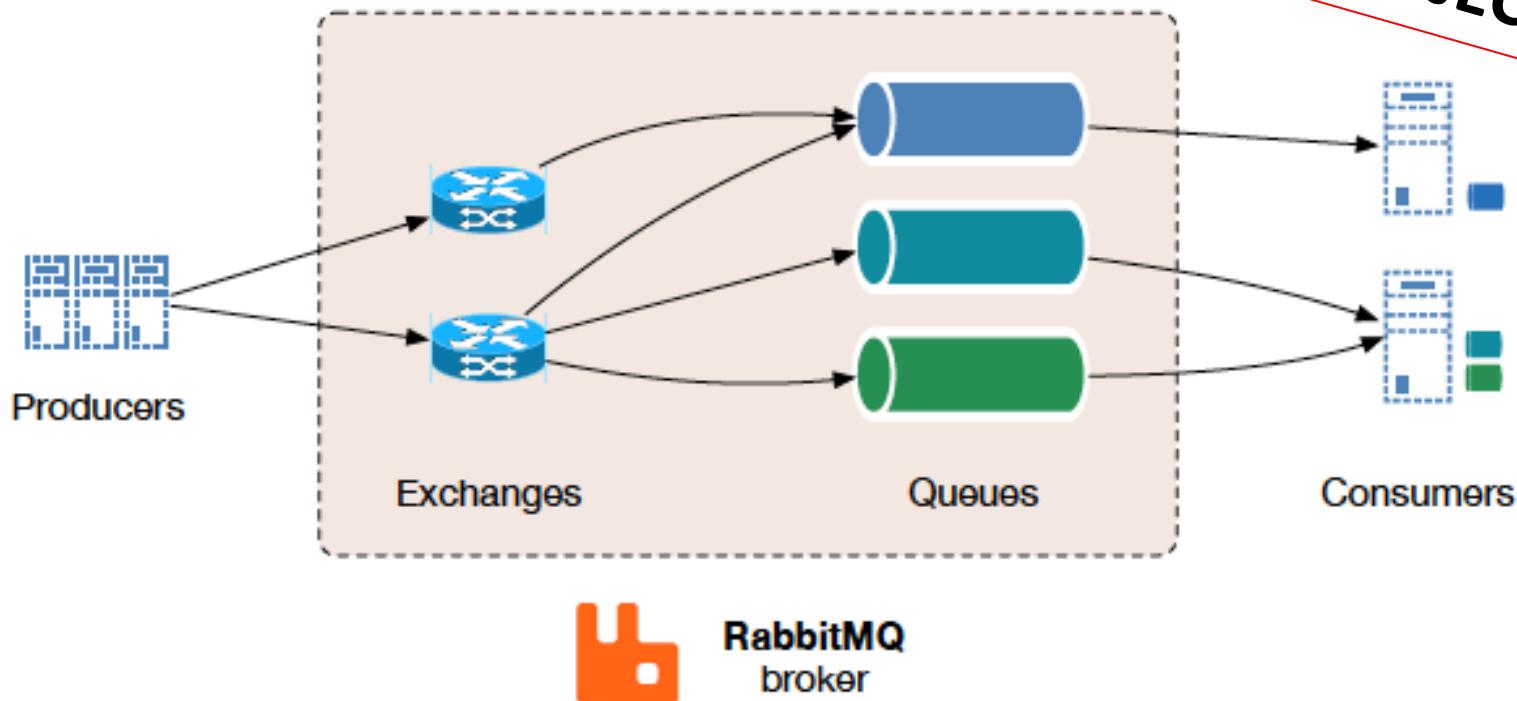
[LDAP plugin](#)

[Configuring HTTP-
based \(Web\) plugins](#)

[Installing plugins](#)

[Plugin development](#)

CAN BE USEFULL IN SOME PROJECTS



Understanding When to use RabbitMQ or Apache Kafka
<https://content.pivotal.io/blog/understanding-when-to-use-rabbitmq-or-apache-kafka>

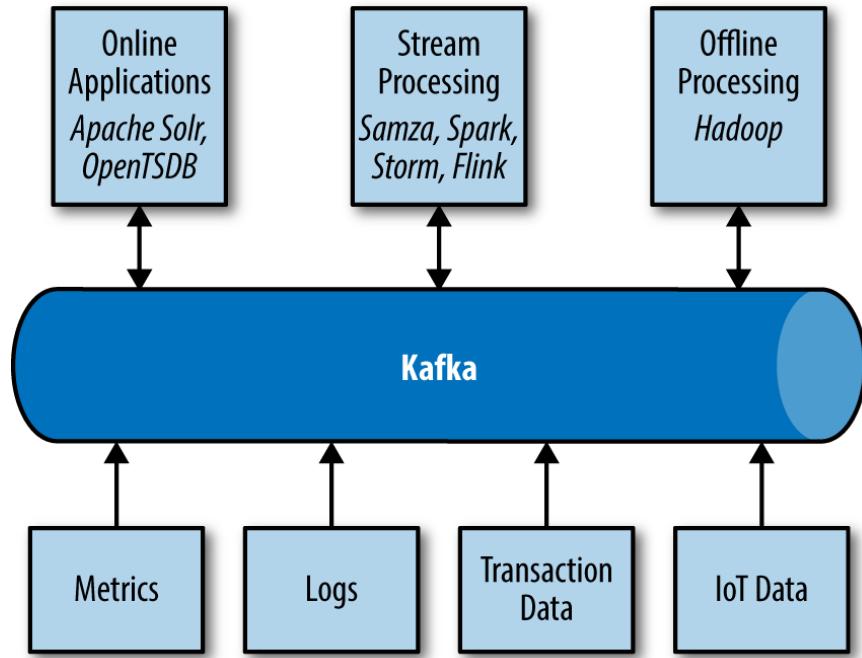
Kafka: a worthy option



José Maria Fernandes (jfernand@ua.pt) - August 2018

Kafka

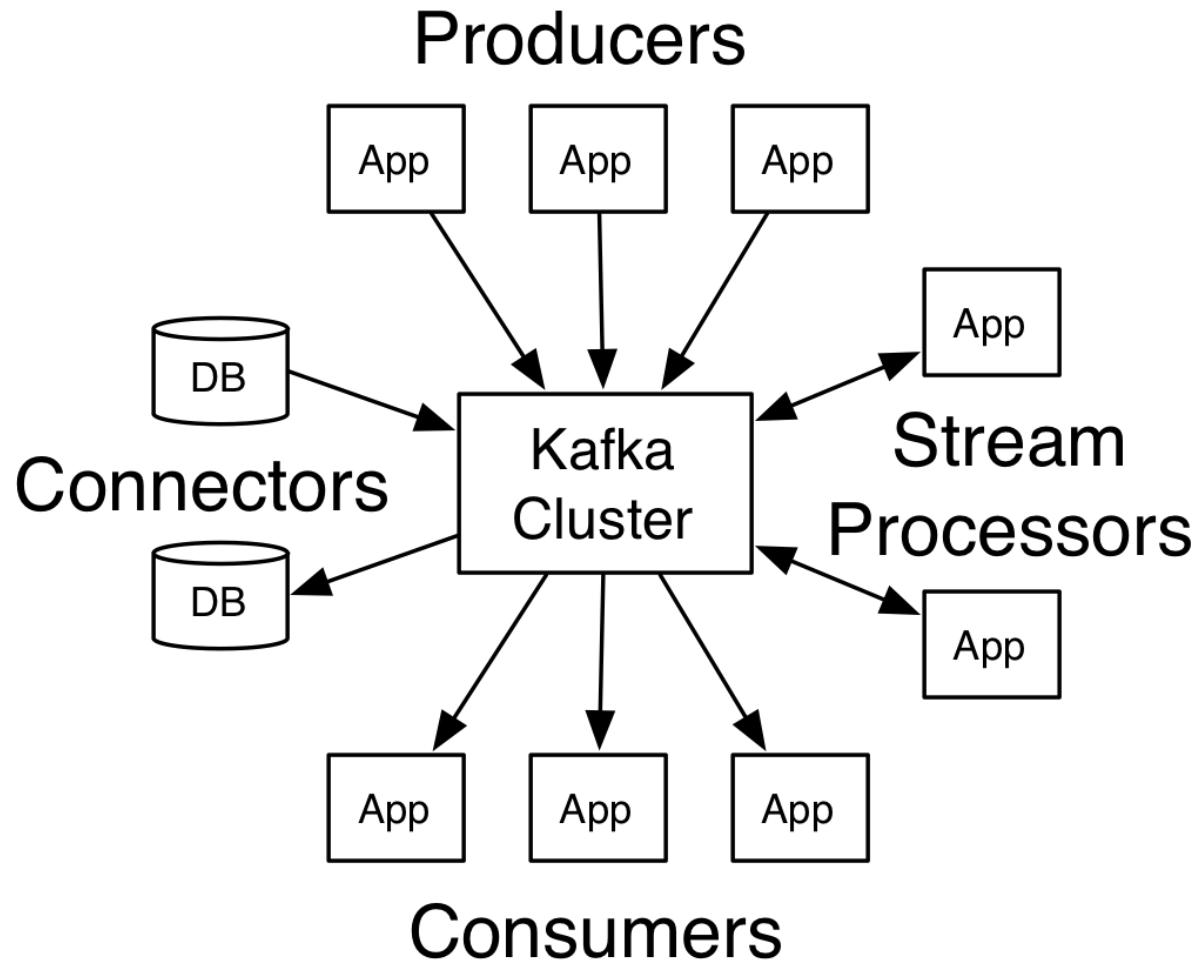
- As a Messaging System
- As a Storage System
- for Stream Processing



<https://balamaci.ro/kafka-streams-for-stream-processing/>

<https://www.safaribooksonline.com/library/view/kafka-the-definitive/9781491936153/ch01.html>

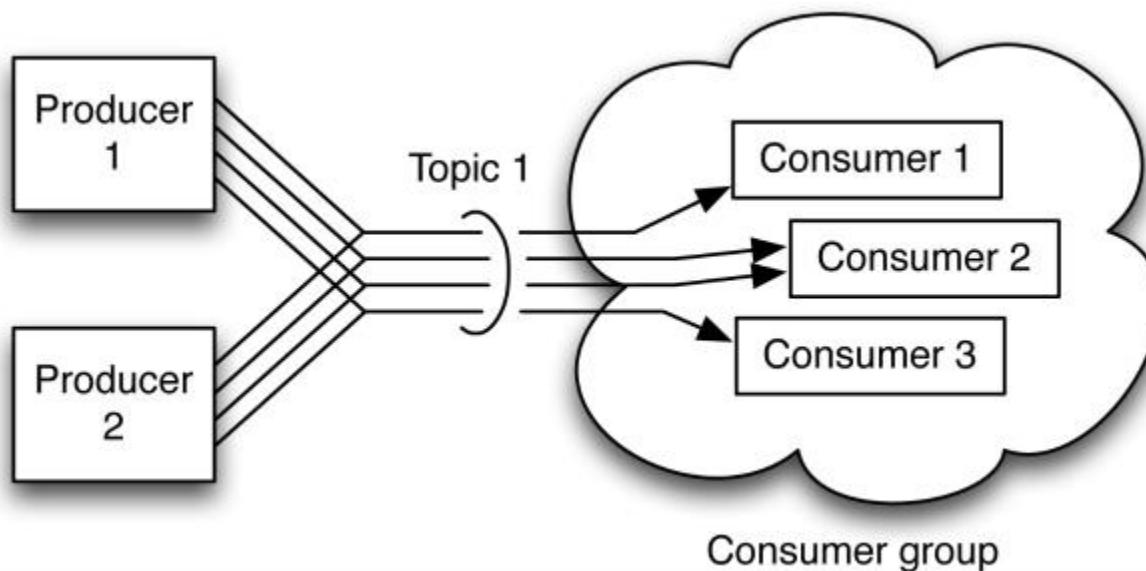
Kafka is a broker



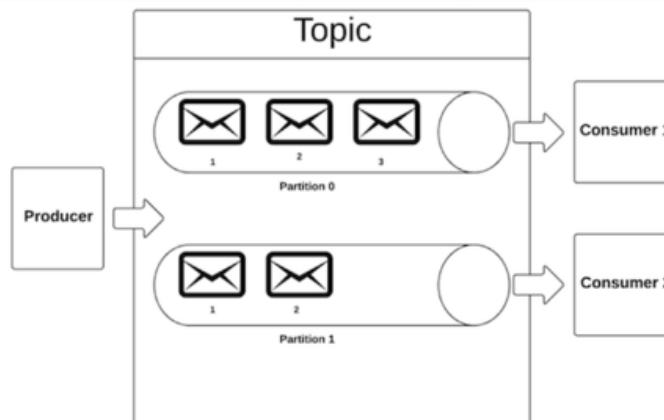
<http://kafka.apache.org/intro>

José Maria Fernandes (jfernand@ua.pt) - August 2018

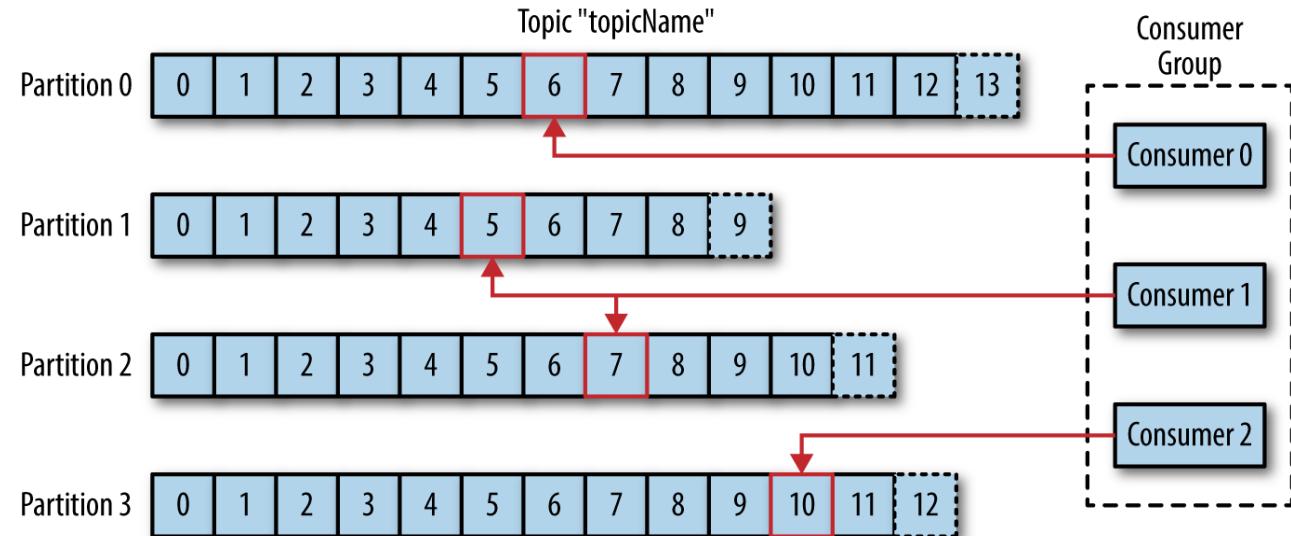
It has topic to publish “events”



Uses topic partition to scale

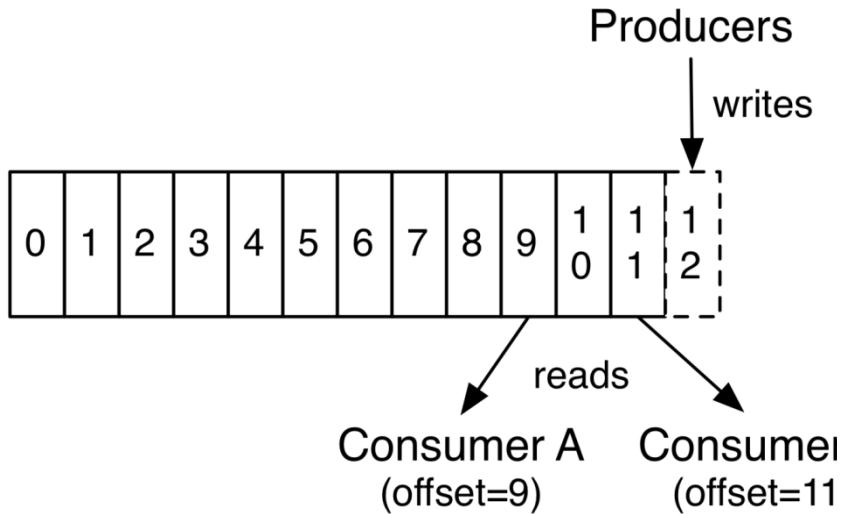


<https://www.javaworld.com/article/3066873/big-data/big-data-messaging-with-kafka-part-2.html>



<https://www.safaribooksonline.com/library/view/kafka-the-definitive/9781491936153/ch01.html>

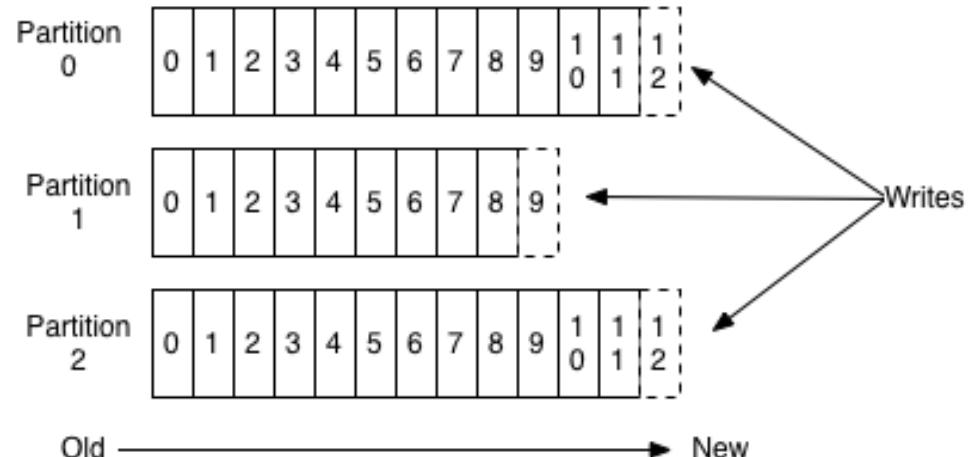
Uses memory files to store



Message agnostic i.e. appends message, pointers are file offsets.
No message processing

Concurrent access depends on file concurrent OS access support

Anatomy of a Topic





Apache Kafka

 Pages Blog

SPACE SHORTCUTS

 Retrospectives

CHILD PAGES

 Index

Clients

 Space tools ▾

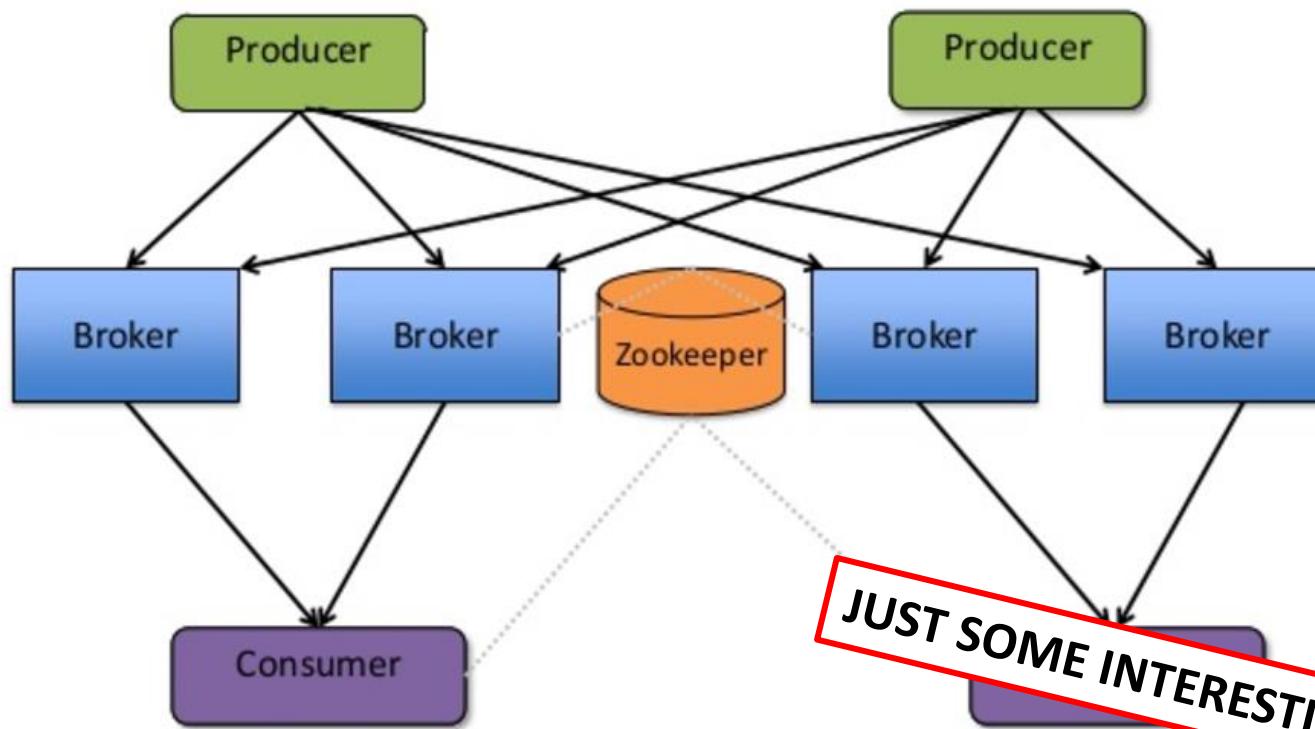
Pages / Index

Clients

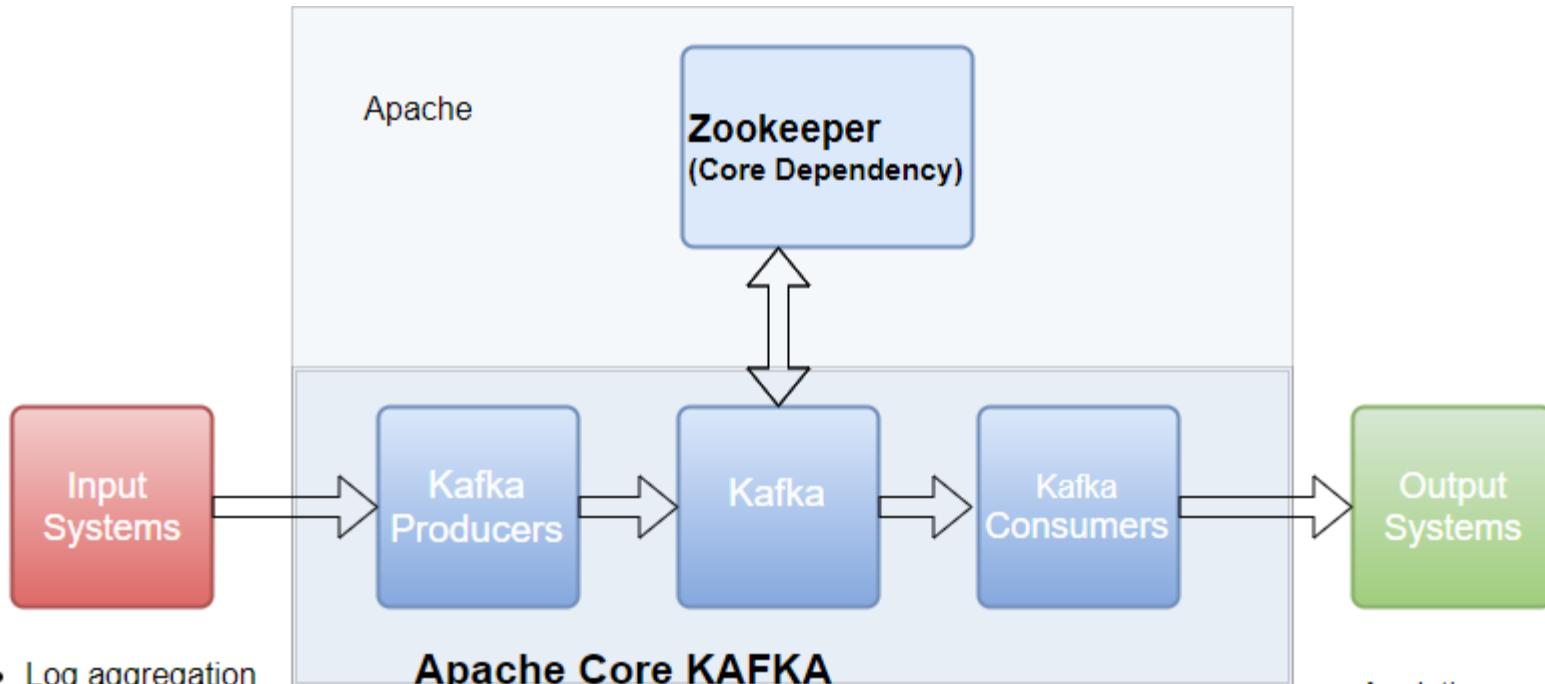
Created by Jun Rao, last modified by Leo Gorodinski on Dec 19, 2017

- How The Kafka Project Handles Clients
- C/C++
- Python
- Go (AKA golang)
- Erlang
- .NET
- Clojure
- Ruby
- Node.js
- Proxy (HTTP REST, etc)
- Perl
- stdin/stdout
- PHP
- Rust
- Alternative Java
- Storm
- Scala DSL
- Clojure
- Swift
- Client Libraries Previously Supported

Kafka Architecture



Kafka architecture



- Log aggregation
- Metrics
- KPIs
- Batch imports
- Audit trail
- User activity logs
- Web logs

Not part of core

- Schema Registry
- Avro
- Kafka REST Proxy
- Kafka Connect
- Kafka Streams

Apache Kafka Core

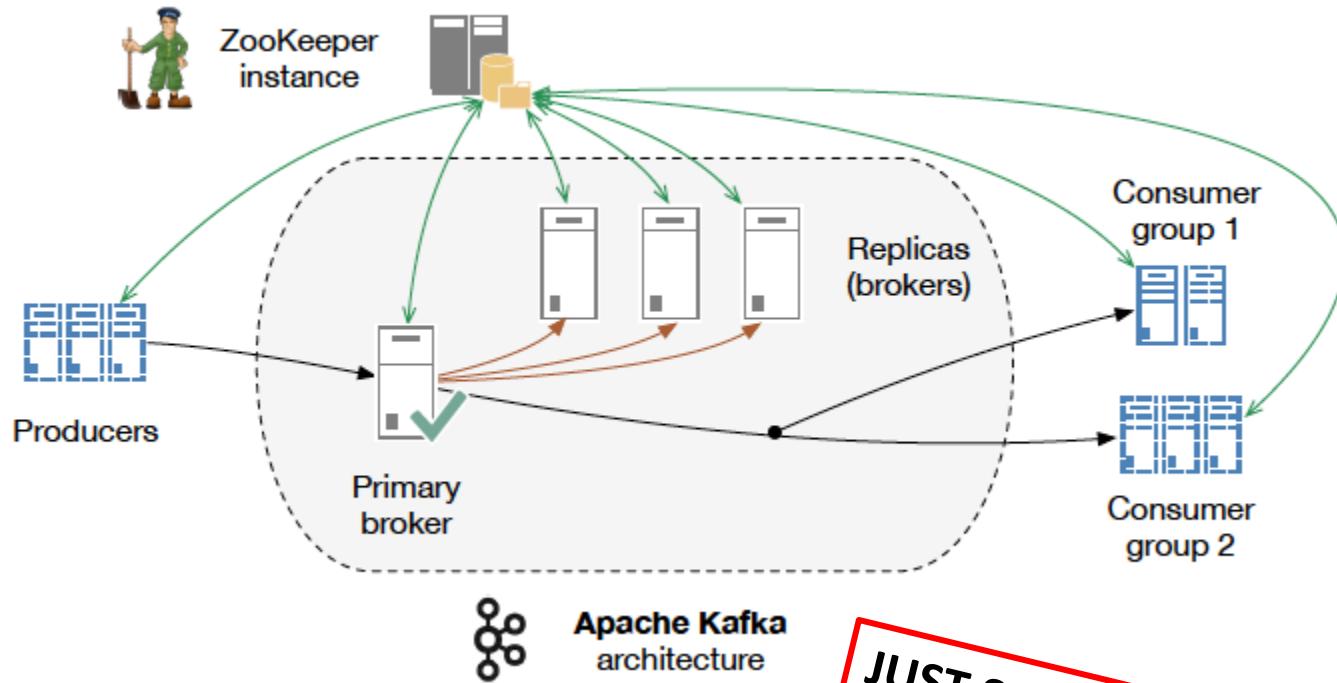
- Server/Broker
- Scripts to start libs
- Script to start up Zookeeper
- Utils to create topics
- Utils to monitor stats

- Analytics
- Databases
- Machine Learning
- Dashboards
- Indexed for Search
- Business Intelligence

<https://dzone.com/articles/kafka-architecture>

José Maria Fernandes (jfernand@ua.pt) - August 2018

Support clustering



Understanding When to use RabbitMQ or Apache Kafka

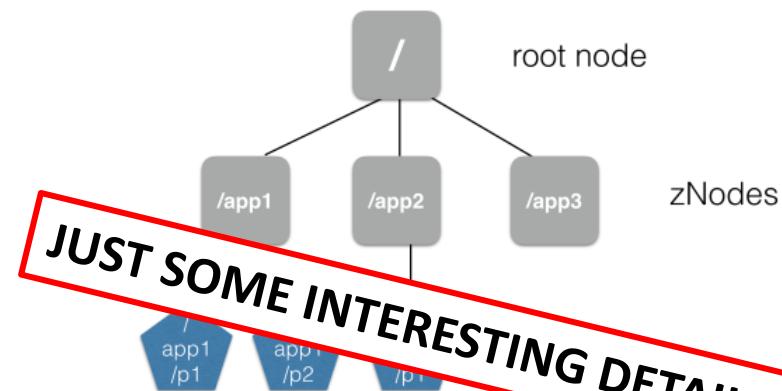
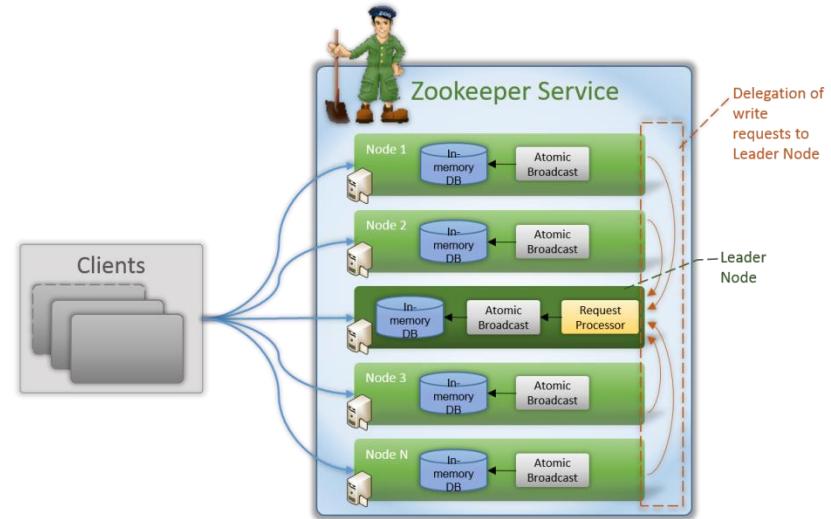
<https://content.pivotal.io/blog/understanding-when-to-use-rabbitmq-or-apache-kafka>

José Maria Fernandes (jfernand@ua.pt) - August 2018



Apache Zookeeper

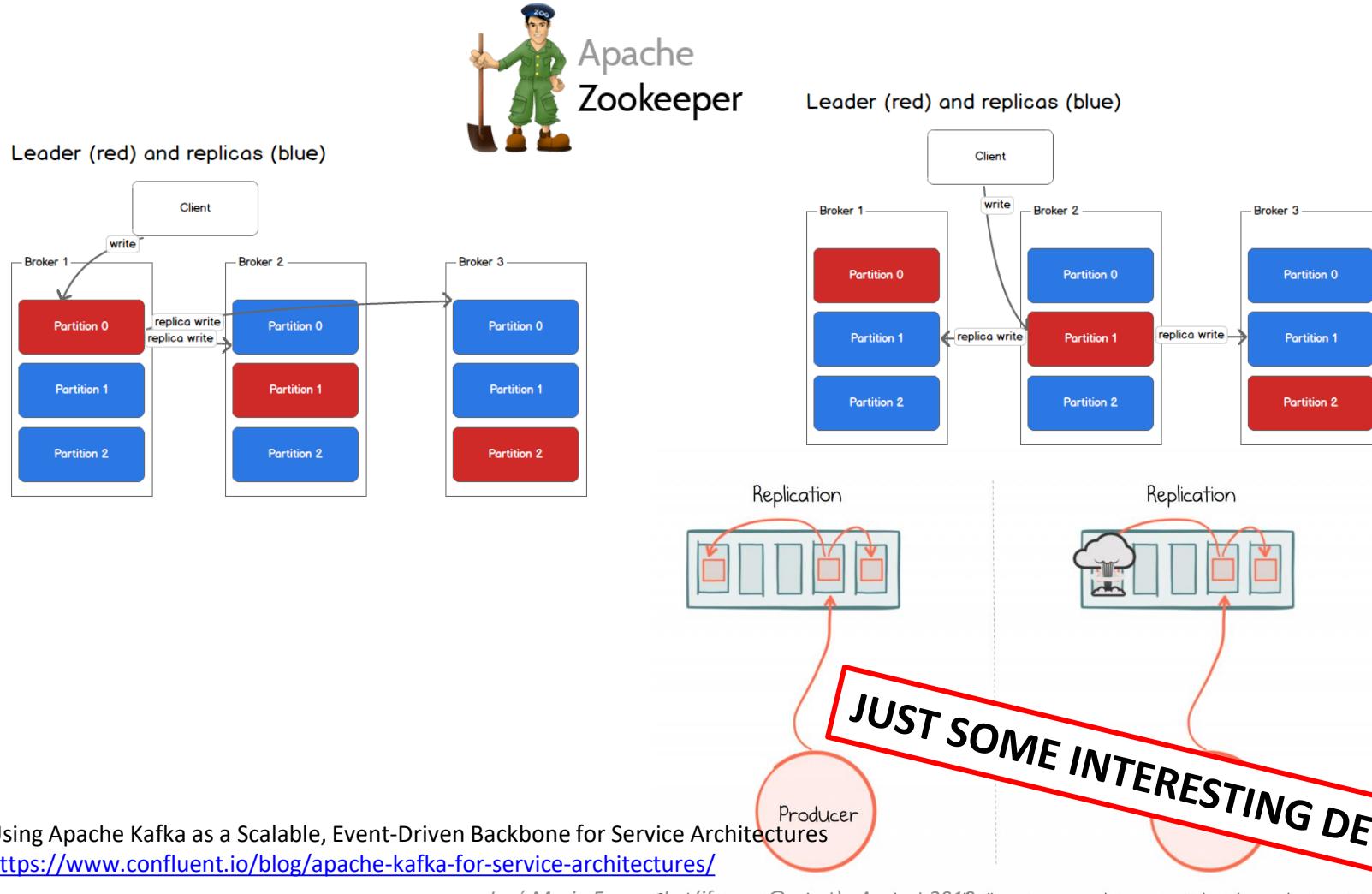
- Apache open source
- Distributed file system
 - Focused of being a repository for configuration files and the services needed around that.
 - Support co-ordinating services for distributed solutions.
- it is itself, a distributed system.
- used by Hadoop components
 - HDFS file system, HBase, Kafka, and others.



<http://www.allprogrammingtutorials.com/tutorials/introduction-to-apache-zookeeper.php>

José Maria Fernandes (jffernand@ua.pt) - August 2018

Kafka consistency and failover

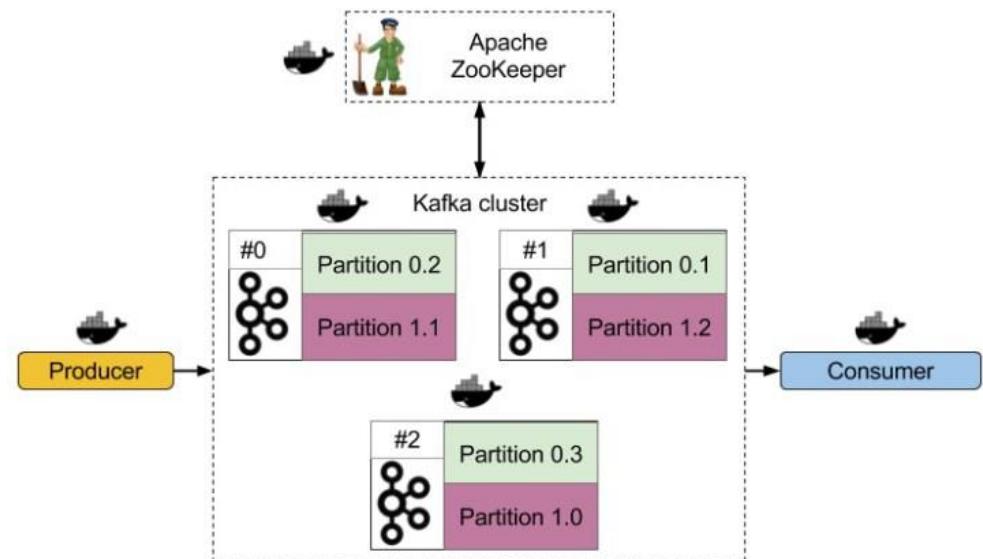


José Maria Fernandes (jfernandez@ua.pt) - August 2018

Replication over three machines will allow two machines to fail without losing data

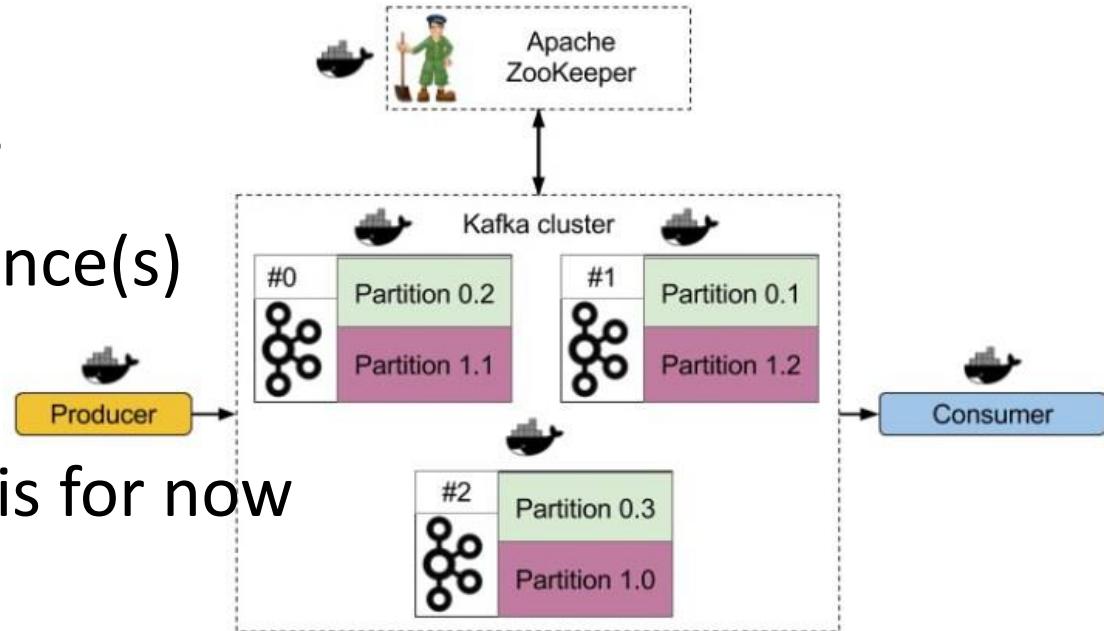


Deploying kafka



Deploying kafka

- Basic concern
 - access Zookeeper
 - access kafka instance(s)
- Native OS
 - Probably using this for now
- Docker
 - several options



<https://codeblog.dotsandbrackets.com/highly-available-kafka-cluster-docker/>

Start ZK and kafka



- in linux
 - bin/kafka-server-start.sh config/server.properties
 - bin/zookeeper-server-start.sh config/zookeeper.properties
- in windows
 - bin\windows\kafka-server-start.bat config\server.properties
 - bin\windows\zookeeper-server-start.bat config\zookeeper.properties

<https://kafka.apache.org/quickstart>



Search

Explore Help Sign up Sign in

PUBLIC | AUTOMATED BUILD

wurstmeister/kafka ☆

Last pushed: 16 hours ago

Repo Info Tags Dockerfile Build Details

Short Description

Multi-Broker Apache Kafka Image

Full Description

docker pulls 26M
docker stars 642
version 2.11-2.0.0
0B 20 layers

kafka-docker

Dockerfile for Apache Kafka

The image is available directly from [Docker Hub](#)

Announcements

- 03-Apr-2018 - **BREAKING** - `KAFKA_ADVERTISED_PROTOCOL_NAME` and `KAFKA_PROTOCOL_NAME` removed. Please update to canonical `kafka` settings.
- 03-Apr-2018 - **BREAKING** - `KAFKA_ZOOKEEPER_CONNECT` is now a mandatory environment var.

Pre-Requisites

- install docker-compose <https://docs.docker.com/compose/install/>
- modify the `KAFKA_ADVERTISED_HOST_NAME` in `docker-compose.yml` to match your docker host IP / Note:
Do not use local <https://hub.docker.com/r/wurstmeister/kafka/>
- if you want to change `message.max.bytes` in `docker-compose.yml`, e.g. in order to increase the message.max.bytes parameter. Fernández (fernando@ua.pt) - August 2018

YTES: 2000000. To turn off automatic topic creation set

CREATE_ENABLE: false



Search

Explore Help Sign up Sign in

PUBLIC | AUTOMATED BUILD

wurstmeister/kafka ☆

Last pushed: 16 hours ago

Repo Info Tags Dockerfile Build Details

Short Description

Multi-Broker Apache Kafka Image

Full Description

docker pulls 26M
docker stars 642
version 2.11-2.0.0
0B 20 layers

kafka-docker

Dockerfile for Apache Kafka

The image is available directly from [Docker Hub](#)

Announcements

- 03-Apr-2018 - BREAKING - KAFKA_ADVERTISED_PROTOCOL Please update to canonical kafka settings.
- 03-Apr-2018 - BREAKING - KAFKA_ZOOKEEPER_CONNECT is now a mandatory environment var.

Pre-Requisites

- install docker-compose <https://docs.docker.com/compose/install/>
- modify the KAFKA_ADVERTISED_HOST_NAME in `docker-compose.yml` to match your docker host IP / Note: Do not use local IP
- if you want to change the `message.max.bytes` parameter (from <https://github.com/wurstmeister/docker-kafka/pull/15>) in `docker-compose.yml`, e.g. in order to increase the message.max.bytes parameter (from <https://github.com/wurstmeister/docker-kafka/pull/15>)

Usage

Start a cluster:

- `docker-compose up -d`

Add more brokers:

- `docker-compose scale kafka=3`

Destroy a cluster:

- `docker-compose stop`

and

wurstmeister/kafka

ter

kafka-docker

August 2018





Connectivity

sscaling edited this page on Mar 29 · 1 revision

Kafka connectivity

A large percentage of questions and issues opened against the kafka-docker project concern configuring Kafka networking. This is often a case of not understanding the Kafka requirements, or not understanding docker networking.

This page aims to explain some of the basic requirements to help people resolve any initial issues they may encounter. It is not an exhaustive guide to configuring Kafka and/or Docker.

Kafka requirements

There are three main requirements for configuring Kafka networking.

1. Each Broker must be able to talk to Zookeeper - for leader election etc.
2. Each Broker must be able to talk to every other Broker - for replication etc.
3. Each Consumer/Producer must be able to talk to every Broker - for reading/writing data etc.

The following diagram represents the different communication paths:

<https://github.com/wurstmeister/kafka-docker/wiki/Connectivity>



Pages 3

Find a Page...

[Home](#)

[Connectivity](#)

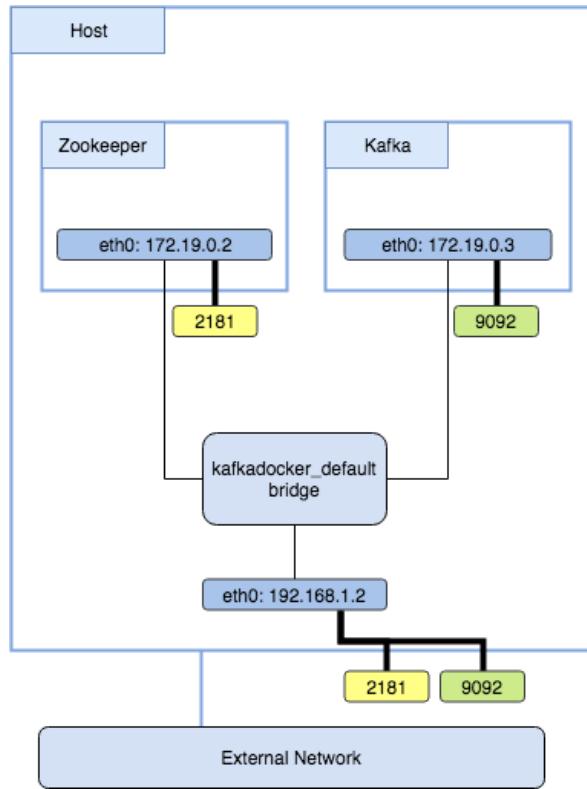
[Swarm](#)

Clone this wiki locally

<https://github.com/wurstmeister/kafka-docker>



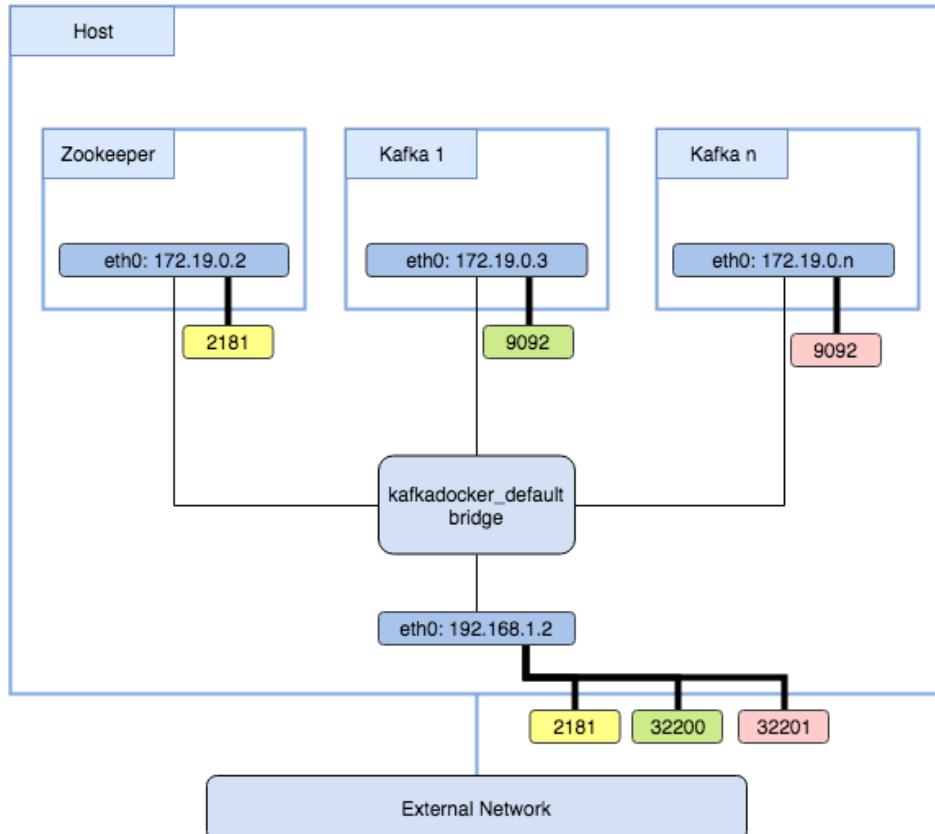
[Clone in Desktop](#)



```
# KAFKA_ADVERTISED_HOST_NAME: localhost
# ZOOKEEPER_CONNECT: zookeeper:2181
docker-compose -f docker-compose-single-broker.yml up -d
```

Kafka connectivity

<https://github.com/wurstmeister/kafka-docker/wiki/Connectivity>



```
# KAFKA_ADVERTISED_HOST_NAME: 192.168.1.2
# ZOOKEEPER_CONNET: zookeeper:2181
docker-compose up -d zookeeper
docker-compose scale kafka=2
```

Kafka connectivity

<https://github.com/wurstmeister/kafka-docker/wiki/Connectivity>



Search

Explore Help Sign up Sign in

PUBLIC | AUTOMATED BUILD

spotify/kafka

Last pushed: 2 years ago

[Repo Info](#) [Tags](#) [Dockerfile](#) [Build Details](#)

Short Description

A simple docker image with both Kafka and Zookeeper.

Full Description

Kafka in Docker

This repository provides everything you need to run Kafka in Docker.

For convenience also contains a packaged proxy that can be used to get data from a legacy Kafka 7 cluster into a dockerized Kafka 8.

Why?

The main hurdle of running Kafka in Docker is that it depends on Zookeeper.

Compared to other Kafka docker images, this one runs both Zookeeper and Kafka in the same container. This means:

<https://hub.docker.com/r/spotify/kafka/>

José Maria Fernandes (jffern@ua.pt) - August 2018



PUBLIC | AUTOMATED BUILD

spotify/kafka

Last pushed: 2 years ago

Repo Info

Tags

Short Description

A simple docker

Full Description

Kafka in Docker

Why?

The main hurdle of running Kafka in Docker is that it depends on Zookeeper. Compared to other Kafka docker images, this one runs both Zookeeper and Kafka in the same container. This means:

- No dependency on an external Zookeeper host, or linking to another container
- Zookeeper and Kafka are configured to work together out of the box

This repository provides everything you need to run Kafka in Docker.

For convenience also contains a packaged proxy that can be used to get data from

a legacy `docker run -p 2181:2181 -p 9092:9092 --env`
`ADVERTISED_HOST=localhost --env ADVERTISED_PORT=9092`
`spotify/kafka`

The main hurdle of running Kafka in Docker is that it depends on Zookeeper.

Compared to other Kafka docker images, this one runs both Zookeeper and Kafka in the same container. This means:

<https://hub.docker.com/r/spotify/kafka/>

José Maria Fernandes (jjfernand@ua.pt) - August 2018

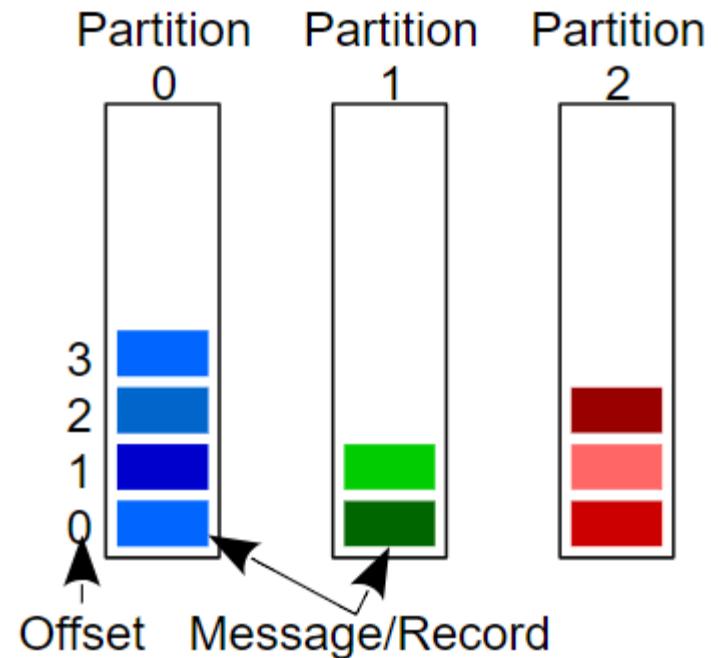


Producer / Consumer using shell i.e. no code

Create topic

```
bin/kafka-topics.sh --create  
--zookeeper localhost:2181  
--replication-factor 1 --partitions 1  
--topic test
```

```
bin\windows\kafka-topics.bat --create  
--zookeeper localhost:2181  
--replication-factor 1 --partitions 1  
--topic test
```

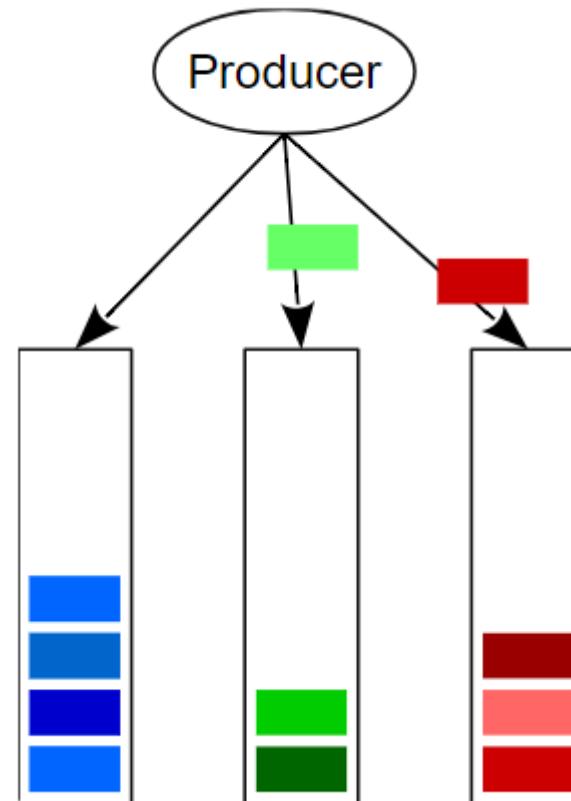


<https://kafka.apache.org/quickstart>

Run producer

```
bin/kafka-console-producer.sh  
--broker-list localhost:9092  
--topic test
```

```
bin\windows\kafka-console-producer.bat  
--broker-list localhost:9092  
--topic test
```

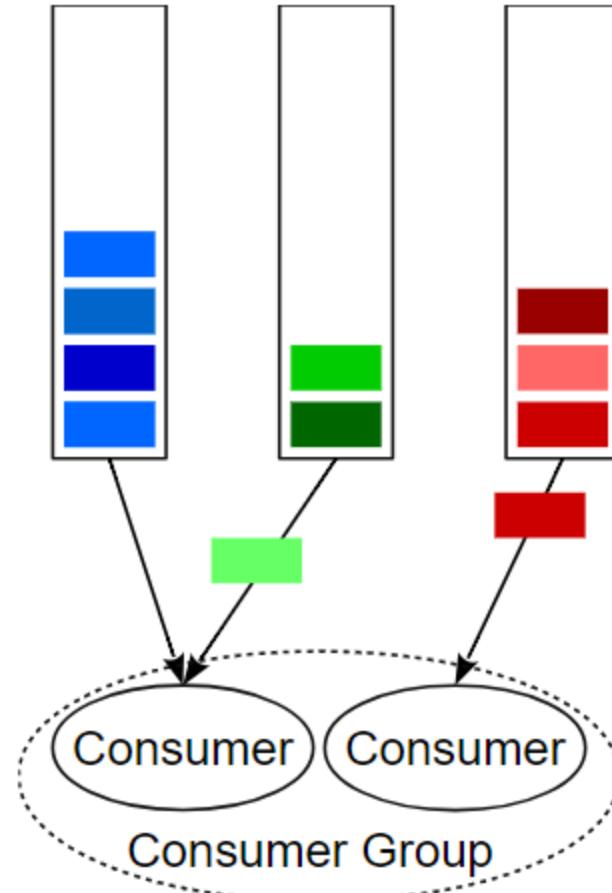


<https://kafka.apache.org/quickstart>

Run consumer

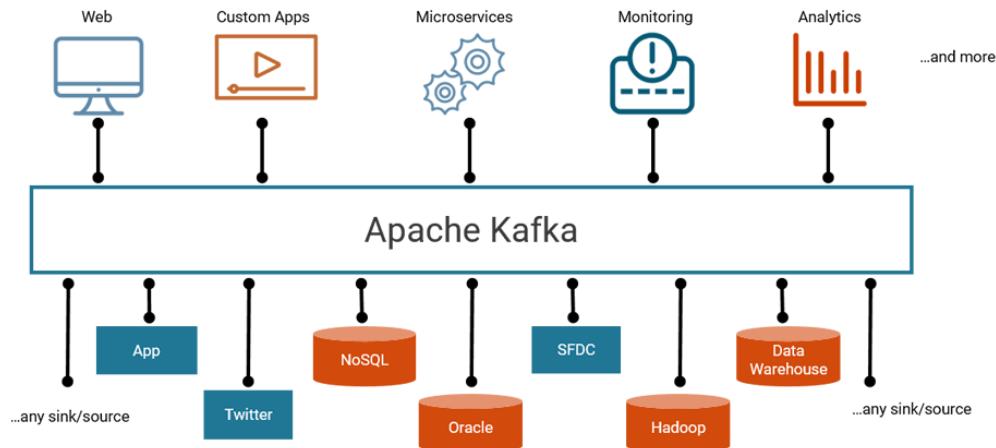
```
bin/kafka-console-consumer.sh  
--bootstrap-server localhost:9092  
--topic test --from-beginning
```

```
bin\windows\kafka-console-consumer.bat  
--bootstrap-server localhost:9092  
--topic test --from-beginning
```

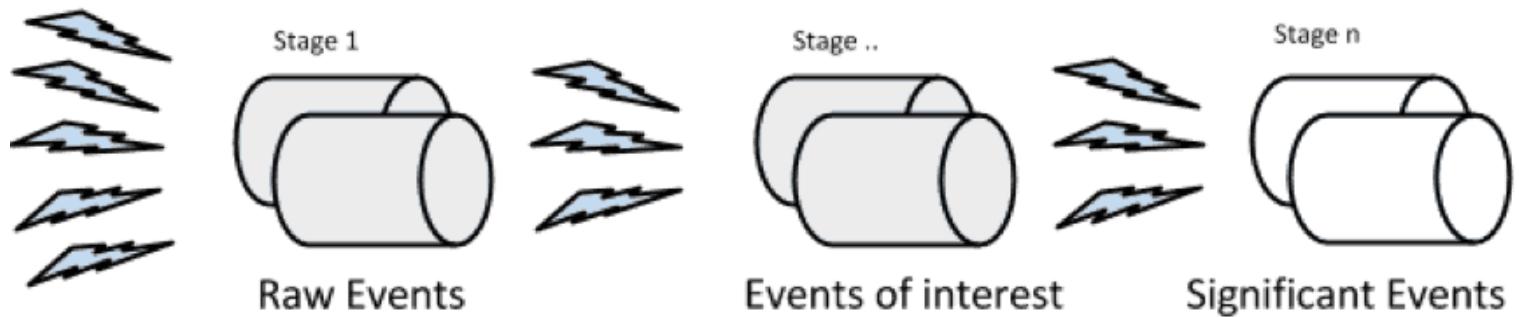


<https://kafka.apache.org/quickstart>

Pipeline & integration



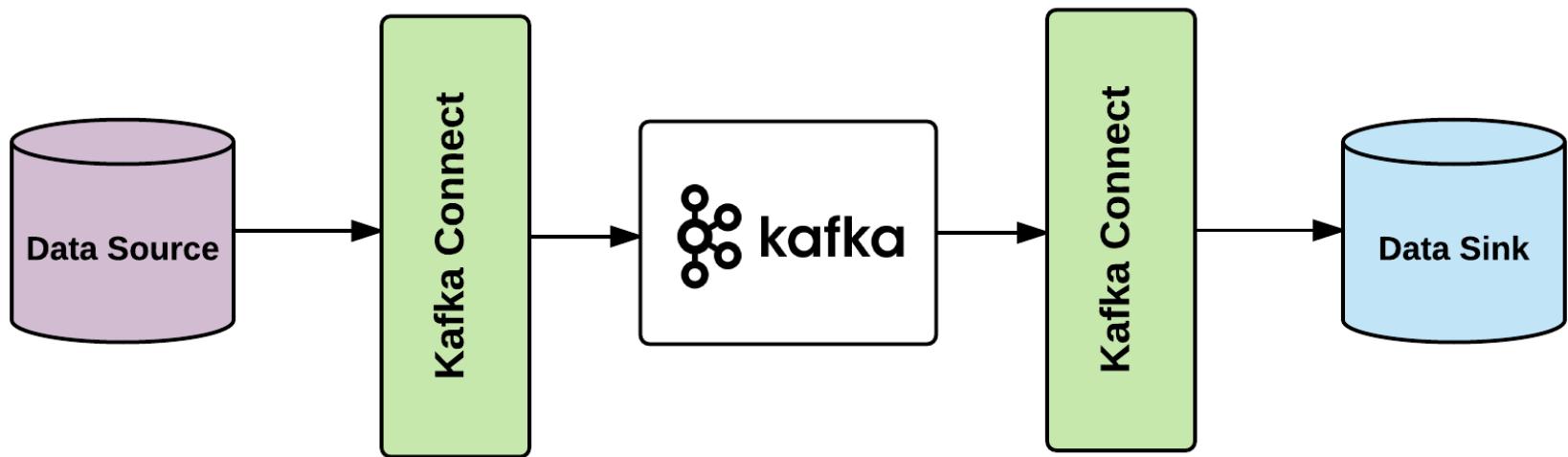
Real-time Multi-stage Data Refinery Pipelines



<https://hortonworks.com/blog/storm-kafka-together-real-time-data-refinery/>

José Maria Fernandes (jfernand@ua.pt) - August 2018

Kafka connect



<https://kafka.apache.org/documentation/#connect>

<https://www.confluent.io/product/connectors/>

<https://docs.confluent.io/current/connect/quickstart.html>

From file to kafka

sink

```
name=local-file-source
connector.class=FileStreamSource
tasks.max=1
file=c:\\tmp\\my-test.txt
topic=my-connect-test1
```

```
bin\\windows\\connect-standalone.bat my-standalone1.properties my-file-source.properties
bin\\windows\\kafka-console-consumer.bat --bootstrap-server localhost:9092 --topic my-connect-test1 --from-beginning
```

“connector”

```
key.converter=org.apache.kafka.connect.storage.StringConverter
value.converter=org.apache.kafka.connect.storage.StringConverter
```

```
# always want to use the built-in default. Offset  
and config data is never visible outside of Kafka  
Connect in this format.
```

```
internal.key.converter=org.apache.kafka.connect.json.JsonConverter
internal.value.converter=org.apache.kafka.connect.json.JsonConverter
internal.key.converter.schemas.enable=false
internal.value.converter.schemas.enable=false
```

```
offset.storage.file.filename=c:\\tmp\\connect.offsets
offset.flush.interval.ms=1000
```

From kafka to file

sink

```
name=local-file-sink  
connector.class=FileStreamSink  
tasks.max=1  
file=c:\tmp\test.sink.txt  
topics=my-connect-test
```

“connector”

```
bootstrap.servers=localhost:9092  
  
key.converter=org.apache.kafka.connect.storage.StringConverter  
value.converter=org.apache.kafka.connect.storage.StringConverter  
  
internal.key.converter=org.apache.kafka.connect.json.JsonConverter  
internal.value.converter=org.apache.kafka.connect.json.JsonConverter  
internal.key.converter.schemas.enable=false  
internal.value.converter.schemas.enable=false  
  
offset.storage.file.filename=c:\tmp\connect.offsets  
# Flush much faster than normal, which is useful  
for testing/debugging  
offset.flush.interval.ms=10000
```

```
bin\windows\kafka-console-producer.bat --broker-list localhost:9092 --topic my-connect-test  
bin\windows\connect-standalone.bat my-standalone.properties my-file-sink.properties
```

Simple producer

```
public class Producer {  
  
    public static void main(String[] args){  
        Properties properties = new Properties();  
        properties.put("bootstrap.servers", "localhost:9092");  
        properties.put("key.serializer", "org.apache.kafka.common.serialization.StringSerializer");  
        properties.put("value.serializer", "org.apache.kafka.common.serialization.StringSerializer");  
  
        KafkaProducer kafkaProducer = new KafkaProducer(properties);  
        try{  
            for(int i = 0; i < 100; i++){  
                System.out.println(i);  
                kafkaProducer.send(new ProducerRecord("devglan-test", Integer.toString(i), "test message - " +  
                    i));  
            }  
        }catch (Exception e){  
            e.printStackTrace();  
        }finally {  
            kafkaProducer.close();  
        }  
    }  
}
```

<https://www.devglan.com/corejava/apache-kafka-java-example>

jose ivaria fernandes (jffernanc@ua.pt) - August 2018

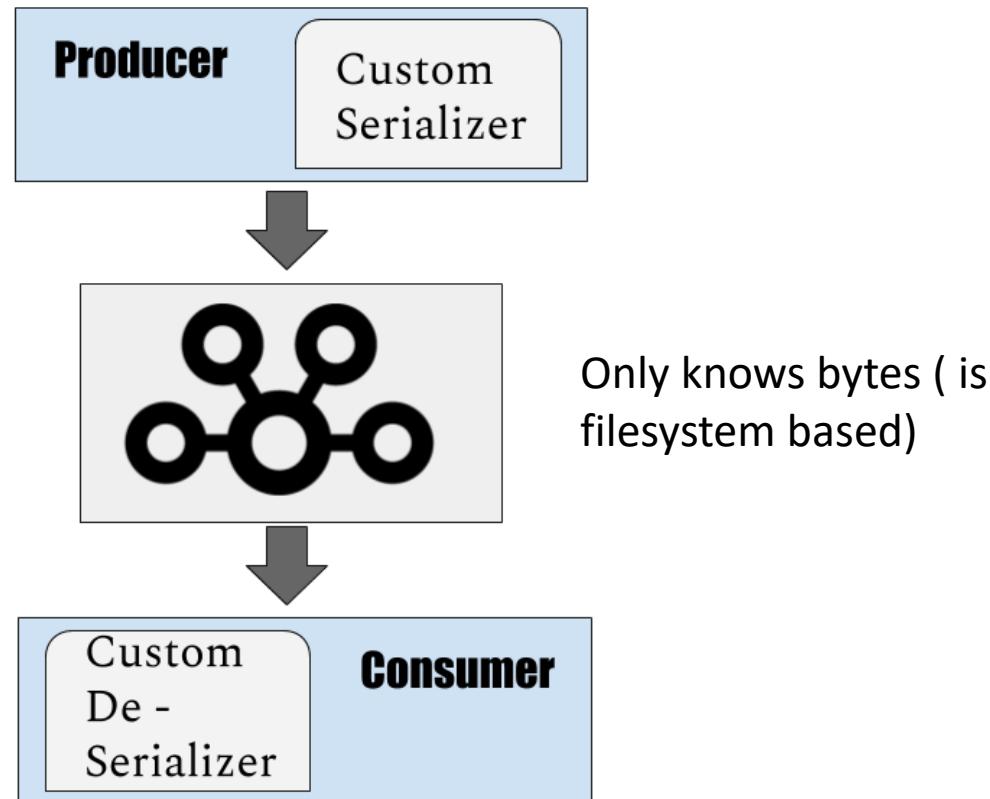
Simple consumer

```
public class Consumer {  
  
    public static void main(String[] args) {  
        Properties properties = new Properties();  
        properties.put("bootstrap.servers", "localhost:9092");  
        properties.put("key.deserializer", "org.apache.kafka.common.serialization.StringDeserializer");  
        properties.put("value.deserializer", "org.apache.kafka.common.serialization.StringDeserializer");  
        properties.put("group.id", "test-group");  
  
        KafkaConsumer kafkaConsumer = new KafkaConsumer(properties);  
        List topics = new ArrayList();  
        topics.add("devglan-test");  
        kafkaConsumer.subscribe(topics);  
        try{  
            while (true){  
                ConsumerRecords records = kafkaConsumer.poll(10);  
                for (ConsumerRecord record: records){  
                    System.out.println(String.format("Topic - %s, Partition - %d, Value: %s", record.topic(), record.partition(), record.value()));  
                }  
            }  
        }catch (Exception e){  
            System.out.println(e.getMessage());  
        }finally {  
            kafkaConsumer.close();  
        }  
    }  
}
```

<https://www.devglan.com/corejava/apache-kafka-java-example>

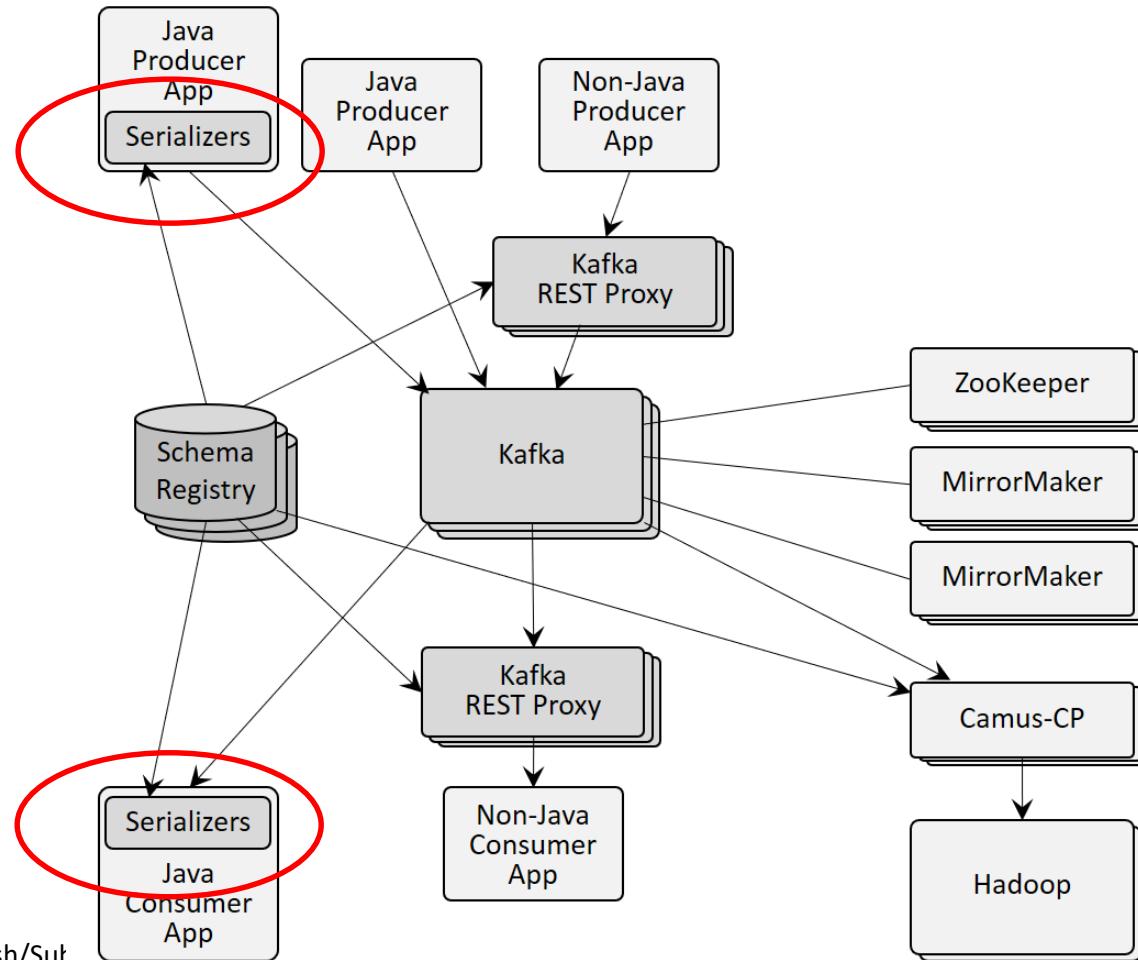


Kafka is data agnostic





Components architecture



Comparing Kafka and Solace for Publish/Suk
<http://dev.solace.com/kafka/product-architecture>

You need to provide SerDes

- SerDes (Serializer/Deserializer)
 - Needed in kafka
 - Mandatory (i.e. explicit) when using streams
- for the data types of record keys and record values
- Several existing
 - <https://kafka.apache.org/10/docs/developer-guide/datatypes>
- Can define your own

- [Available SerDes](#)
 - [Primitive and basic types](#)
 - [Avro](#)
 - [JSON](#)
 - [Further serdes](#)

SERDES: Key & value (de)serialization

```
public class Producer {  
  
    public static void main(String[] args){  
        Properties properties = new Properties();  
        properties.put("bootstrap.servers", "localhost:9092");  
        properties.put("key.serializer", "org.apache.kafka.common.serialization.StringSerializer");  
        properties.put("value.serializer", "org.apache.kafka.common.serialization.StringSerializer");  
  
        KafkaProducer kafkaProducer = new KafkaProducer(properties);  
        try{  
            for(int i = 0; i < 10; i++) {  
                System.out.println("Producing message " + i);  
                kafkaProducer.send(new ProducerRecord("devglan-test", "Hello World " + i));  
            }  
        }catch (Exception e){  
            e.printStackTrace();  
        }finally {  
            kafkaProducer.close();  
        }  
    }  
}
```

It (de/en) codes the KEY and VALUE

```
public class Consumer {  
  
    public static void main(String[] args) {  
        Properties properties = new Properties();  
        properties.put("bootstrap.servers", "localhost:9092");  
        properties.put("key.deserializer", "org.apache.kafka.common.serialization.StringDeserializer");  
        properties.put("value.deserializer", "org.apache.kafka.common.serialization.StringDeserializer");  
        properties.put("group.id", "test-group");  
  
        KafkaConsumer kafkaConsumer = new KafkaConsumer(properties);  
        List topics = new ArrayList();  
        topics.add("devglan-test");  
        kafkaConsumer.subscribe(topics);  
        try{  
            while (true){  
                ConsumerRecords records = kafkaConsumer.poll(10);  
                for (ConsumerRecord record: records){  
                    System.out.println(String.format("Topic - %s, Partition - %d, Value: %s", record.topic(), record.partition(), record.value()));  
                }  
            }  
        }
```

SIMPLE... using just strings



THE DZONE GUIDE TO
Internet of Things
Harnessing Device Data

INCLUDING:
An Introduction to IoT Architecture
Learn How to Create an IoT Data Processing Pipeline
The Ups and Downs of Smart Cities

DOWNLOAD GUIDE

How to Create Serializers With Kafka

Kafka provides the ability to publish and subscribe to streams of records. You can create your own custom serializer and deserializer.



by Prabhat Kashyap · Jan. 31, 17 · Integration Zone · Tutorial

Like (6) Comment (1) Save Tweet

20.71k Views

Join the DZone community and get the full member experience.

JOIN FOR FREE

SnapLogic is the leading self-service enterprise-grade integration platform. Download the [2018 GartnerMagic Quadrant for Enterprise iPaaS](#) or [play around on the platform, risk free, for 30 days](#).

Kafka lets us publish and subscribe to streams of records and the records can be of any type (JSON, String, POJO, etc.) Kafka gives users the ability to create our own serializer and deserializer so that we can transmit different data type using it.

In this article, I will demonstrate how to create a custom serializer and deserializer — but first, let's understand what serialization is and why we would serialize.

Serialization and Deserialization

How to Create Serializers With Kafka

<https://dzone.com/articles/kafka-sending-object-as-a-message>

transmission.





```
public class UserSerializer implements Serializer {  
  
    @Override public void configure(Map<String, ?> map, boolean b) {}  
  
    @Override public byte[] serialize(ObjectMapper obj, String s) throws IOException {  
        byte[] retVal = null;  
        ObjectMapper mapper = new ObjectMapper();  
        try {  
            retVal = mapper.writeValueAsBytes(obj);  
        } catch (Exception e) {  
            e.printStackTrace();  
        }  
        return retVal;  
    }  
  
    @Override public Object deserialize(String arg0, byte[] arg1) throws IOException {  
        ObjectMapper mapper = new ObjectMapper();  
        User user = null;  
        try {  
            user = mapper.readValue(arg1, User.class);  
        } catch (Exception e) {  
            e.printStackTrace();  
        }  
        return user;  
    }  
}
```

How

Kafka
your



Join

SnapL
for Ent

Kafka lets us publish and subsc

POJO, etc.) Kafka gives users the

transmit different data type usi

}

In this article, I will demonstrate how to create a custom serializer and deserializer — but first, let's understand what serialization is and why we need it.

Serialization and Deserialization

How to Create Serializers With Kafka

<https://dzone.com/articles/kafka-sending-object-as-a-message>

transmission.



If curious... or need...

- How to Create Serializers With Kafka
 - <https://dzone.com/articles/kafka-sending-object-as-a-message>
- Kafka tutorial #3 - JSON SerDes
 - <https://aseigneurin.github.io/2018/08/01/kafka-tutorial-3-json-serdes.html>
- Kafka tutorial #7 - Kafka Streams SerDes and Avro
 - <https://aseigneurin.github.io/2018/08/06/kafka-tutorial-7-kafka-streams-serdes-and-avro.html>
- Robust Message Serialization in Apache Kafka Using Apache Avro, Part 1
 - <http://blog.cloudera.com/blog/2018/07/robust-message-serialization-in-apache-kafka-using-apache-avro-part-1/>

Some notes

- Issues due to access to port or correct ip address
 - Zookeeper, kafka
 - Within network, network vs container...
- Typical: Leader Not Available Kafka in Console Producer
 - <https://stackoverflow.com/questions/35788697/leader-not-available-kafka-in-console-producer>
 - (...)
 - It could be related to advertised.host.name setting in your server.properties.
 - (...)
 - ...what worked for me was this
advertised.listeners=PLAINTEXT://my.ip:9092



Home About Services Community



Blog Archive Blog Español 日本のブログ Beginner's Guides Production

Payara Micro JCA Adapters - Apache Kafka

08 Jun 2017

[Tweet](#) [Share](#) [Like 0](#) [Share](#) [G+](#)



by Matthew Gill

In this blog, which follows on from the [introduction to Cloud Connectors in Payara Micro](#), we'll guide you through the process of setting up Payara Micro as a *Kafka Client*, which can produce and consume data from Apache Kafka.

What is Apache Kafka?

Put simply, [Apache Kafka](#) is a publish-subscribe messaging system. It is similar to a

<https://blog.payara.fish/payara-micro-jca-adapters-apache-kafka>

priorities are message log over consumers, and because of this is optimised to process an extremely high volume of messages. This also means that it lacks a few features,



[Blog Archive](#) [Blog Español](#) [日本のブログ](#) [Beginner's Guides](#) [Production](#)

Payara

08 Jun 2017

[Tweet](#) [in Share](#)

In this blog, we'll

Micro, we'll g

Client, which

```
@MessageDriven(activationConfig = {  
    @ActivationConfigProperty(propertyName = "clientId",  
        propertyValue = "testClient"),  
    @ActivationConfigProperty(propertyName = "groupIdConfig",  
        propertyValue = "test-consumer-group"),  
    @ActivationConfigProperty(propertyName = "topics",  
        propertyValue = "testing"),  
    @ActivationConfigProperty(propertyName = "bootstrapServersConfig",  
        propertyValue = "192.168.29.139:9092"),  
    @ActivationConfigProperty(propertyName = "autoCommitInterval",  
        propertyValue = "100"),  
    @ActivationConfigProperty(propertyName = "retryBackoff",  
        propertyValue = "1000"),  
    @ActivationConfigProperty(propertyName = "keyDeserializer",  
        propertyValue = "org.apache.kafka.common.serialization.StringDeserializer"),  
    @ActivationConfigProperty(propertyName = "valueDeserializer",  
        propertyValue = "org.apache.kafka.common.serialization.StringDeserializer"),  
    @ActivationConfigProperty(propertyName = "pollInterval", propertyValue = "1000"),  
})  
public class KafkaMDB implements KafkaListener {  
    @OnRecord( topics={"testing"})  
    public void getMessageTest(ConsumerRecord record) {  
        System.out.println("Got record on topic testing " + record);  
    }  
}
```

What is Apache Kafka:

Put simply, **Apache Kafka** is a publish-subscribe messaging system. It is similar to a

<https://blog.payara.fish/payara-micro-jca-adapters-apache-kafka>

priorities are message log over consumers, and because of this is optimised to process an extremely high volume of messages. This also means that it loses a few features.

Deploying to Payara Micro

You need the following three files to make this work:

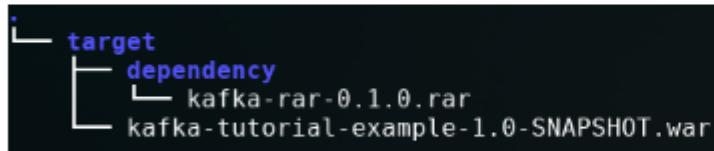
1. Payara Micro Jar (obviously).
2. Kafka Example Jar (the project we've created).
3. Kafka RAR Adapter (the JCA connector project).

Pay

08 Jun

 Tweet

The latter two should be in the directory tree shown below (if you can't find these files then you may not have built the projects with "`mvn package`").



In this
Micrc
Clien

To deploy these to Payara Micro, run the following command.

```
java -jar payara-micro.jar --deploy target/dependency/kafka-rar-0.1.0.rar --dep
```

What is Apache Kafka?

Put simply, **Apache Kafka** is a publish-subscribe messaging system. It is similar to a

<https://blog.payara.fish/payara-micro-jca-adapters-apache-kafka>

priorities are message log over consumers, and because of this is optimised to process an extremely high volume of messages. This also means that it loses a few features.

```
@ConnectionFactoryDefinition(name = "java:module/env/KafkaConnectionFactory",
    description = "Kafka Connection Factory",
    interfaceName = "fish.payara.cloud.connectors.kafka.KafkaConnectionFactory",
    resourceAdapter = "kafka-rar-0.1.0",
    minPoolSize = 2,
    maxPoolSize = 2,
    transactionSupport = TransactionSupport.TransactionSupportLevel.NoTransaction,
    properties = {
        "bootstrapServersConfig=192.168.29.139:9092",
        "clientId=PayaraMicroMessenger"
    })
@Stateless
public class KafkaMessenger {

    @Resource(lookup = "java:module/env/KafkaConnectionFactory")
    KafkaConnectionFactory factory;

    @Schedule(hour = "*", minute = "*", second = "*/5", persistent = false)
    public void sendMessage() {
        try (KafkaConnection conn = factory.createConnection()) {
            conn.send(new ProducerRecord("testing", "Sent from Payara Micro."));
        } catch (Exception ex) {
            Logger.getLogger(getClass().getName()).log(Level.SEVERE, null, ex);
        }
    }
}
```

What is Apache Kafka?

<https://github.com/payara/Cloud-Connectors/tree/master/Kafka>

At simply, Apache Kafka is a publish-subscribe messaging system. It is similar to a

<https://blog.payara.fish/payara-micro-jca-adapters-apache-kafka>

priorities are message log over consumers, and because of this is optimised to process an extremely high volume of messages. This also means that it loses a few features.

[Code](#)[Issues 6](#)[Pull requests 1](#)[Projects 1](#)[Insights](#)

Branch: master

Cloud-Connectors / Kafka /

[Create new file](#)[Find file](#)[History](#)

Pandrex247 Increment version number to SNAPSHOT

Latest commit [5a449dc](#) on Jun 15

KafkaExample	Increment version number to SNAPSHOT	4 months ago
KafkaJCAPI	Increment version number to SNAPSHOT	4 months ago
KafkaRAR	Increment version number to SNAPSHOT	4 months ago
kafka-rest-example	Increment version number to SNAPSHOT	4 months ago
README.md	Increment version number to SNAPSHOT	4 months ago

README.md

Apache Kafka

These modules form the basis of the Apache Kafka JCA connector. The code is in three modules

- KafkaExample is an EJB jar module that shows a Timer Bean which sends a message periodically and an MDB that receives the message
- KafkaRAR is the bulk of the JCA code and the jar file which must be used as a provided dependency for any code using the JCA module
- KafkaRARPackaging is a maven module that assembles the rar file. The rar file should be deployed to your container.

To use the JCA adapter the KafkaRAR.rar should be deployed to your application server.

To deploy the JCA adapter on Payara Micro use the following commands.

```
java -jar payara-micro.jar --deploy kafka-rar-0.1.0.rar --deploy kafka-example-0.1.0.jar
```

Inbound MDB

The KafkaExample module shows an example MDB that receives messages from a Kafka topic. To receive messages you must implement the KafkaListener interface.

<https://github.com/payara/Cloud-Connectors/tree/master/Kafka>

Also you must set the ActivationConfigProperty values suitable for your MDB.

Valid properties are below. On Payara all properties can be replaced via System properties using the syntax





Code

Branch: master

Pandrex247

-

KafkaExample

KafkaJCAF

KafkaRAR

kafka-rest

README.md

README.m

```
@MessageDriven(activationConfig = {  
    @ActivationConfigProperty(propertyName = "clientId", propertyValue = "testClient"),  
    @ActivationConfigProperty(propertyName = "groupIdConfig", propertyValue = "testGroup"),  
    @ActivationConfigProperty(propertyName = "topics", propertyValue = "test,test2"),  
    @ActivationConfigProperty(propertyName = "bootstrapServersConfig", propertyValue = "localhost:9092"),  
    @ActivationConfigProperty(propertyName = "autoCommitInterval", propertyValue = "100"),  
    @ActivationConfigProperty(propertyName = "retryBackoff", propertyValue = "1000"),  
    @ActivationConfigProperty(propertyName = "keyDeserializer", propertyValue = "org.apache.kafka.common.  
    @ActivationConfigProperty(propertyName = "valueDeserializer", propertyValue = "org.apache.kafka.common.  
    @ActivationConfigProperty(propertyName = "pollInterval", propertyValue = "3000"),  
})  
public class KafkaMDB implements KafkaListener {
```

Apa

These m

- Kaf
- the
- Kaf
- JCA
- Kaf

To use th

To deplo

java -

Inbu

Th

im

ht

pu

pu

Also you

```
public KafkaMDB() {  
}  
  
@OnRecords( matchOtherMethods = true)  
public void getMessageTest2(ConsumerRecords records) {  
    System.out.println("Bulk processing called with " + records.count() + " records");  
}  
  
@OnRecord( topics={"test"})  
public void getMessageTest(ConsumerRecord record) {  
    System.out.println("Got record on topic test " + record);  
}
```

```
@OnRecord( topics={"test2"})  
public void getMessageTest2(ConsumerRecord record) {  
    System.out.println("Got record on topic test2 " + record);  
}
```



Spring for Apache Kafka

2.1.10

 [Overview](#) [Learn](#)

The Spring for Apache Kafka (spring-kafka) project applies core Spring concepts to the development of Kafka-based messaging solutions. It provides a "template" as a high-level abstraction for sending messages. It also provides support for Message-driven POJOs with `@KafkaListener` annotations and a "listener container". These libraries promote the use of dependency injection and declarative. In all of these cases, you will see similarities to the JMS support in the Spring Framework and RabbitMQ support in Spring AMQP.

Features

- `KafkaTemplate`
- `KafkaMessageListenerContainer`
- `@KafkaListener`
- `KafkaTransactionManager`
- `spring-kafka-test` jar with embedded kafka server

Kafka Client Compatibility

Spring for Apache Kafka is based on the pure java `kafka-clients` jar. The following is the compatibility matrix:

Spring for Apache Kafka Version	Spring Integration for Apache Kafka Version	<code>kafka-clients</code>
2.2.x	3.1.x	2.0.0
2.2.0.M1	3.1.0.M1	1.1.x
2.1.x	3.0.x	1.0.x, 1.1.x
2.0.x	2.9.x	0.11.0.x, 1.0.x
1.2.x	2.2.x	0.11.0.x, 1.0.x
		0.10.2.x

https://spring.io/projects/spring-kafka

Getting Started with Kafka in Spring Boot



Bartosz Jedrzejewski

May 21, 2018

Choreography, Microservices,
Spring Boot

Kafka seems to only be gaining in popularity. A few years ago you could mostly see it in Big Data engineering context. These days, Kafka is starting to power more common message-oriented architectures. In this article, I want to give you a basic introduction to working with Spring Boot and Kafka.

Installing Kafka on your machine

One common barrier to entry for people to hack around on their machines with Kafka is how tricky the installation can be. For that very reason, I have written "[How to easily run Kafka with Docker for development](#)" – I think you will find it especially useful if you are on Windows!

If you are running macOS or Linux, you can still follow the aforementioned tutorial, but you could also run Kafka and Zookeeper without Docker. To do that I recommend [Confluent.io](#) platform and their [Quick Start tutorial](#).

Enabling Kafka in Spring Boot

Assuming that you have Kafka accessible on `kafka:9092` what follows is basic instruction on integrating your Spring Boot application with Kafka.

With Spring Boot, to use Kafka, you need a single dependency added to your POM file

<https://www.e4developer.com/2018/05/21/getting-started-with-kafka-in-spring-boot/>



ABOUT

E4developer is a place where I share my open and honest views on software development, technology and working with people. The name – e4, comes from a chess move, this is how I start most of my games. Follow me on twitter – [@e4developer](#)

I also blog at [Scott Logic](#) – a great consulting company where I work as a Lead Developer.

When it comes to my chess passion, I also run [chesscollecting.com](#) where you can see some beautiful chess sets.

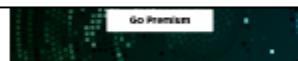
CATEGORIES

- Algorithms (3)
- Architecture (20)
- Books (10)
- Building teams (11)
- Career (22)
- Chess (2)
- Choreography (8)
- DevOps (4)
- Java (13)
- Microservices (47)
- Newsletter (5)
- Orchestration (4)
- Personal (13)
- Public speaking (3)
- Reactive (7)
- Spring Boot (17)
- Spring Cloud (11)
- Spring Cloud Data Flow (4)

Getting Started with



```
Application.properties
spring.kafka.consumer.group-id=kafka-intro
spring.kafka.bootstrap-servers=kafka:9092
```



ABOUT

E4developer is a place where I share my open


Bartosz Jedrzejewski
May 21, 2018
Choreography, Microservices, Spring Boot

```
@Autowired
private KafkaTemplate<String, String> kafkaTemplate;

public void send(String topic, String payload) {
    kafkaTemplate.send(topic, payload);
    System.out.println("Message: "+payload+" sent to topic: "+topic);
}
```

how tricky the installation can be. For that very reason, I have written "[How to easily run Kafka with Docker for development](#)" – I think you will find it especially useful if you are on Windows!

If you are running macOS or Linux, you can still follow the aforementioned tutorial, but you could also run Kafka and Zookeeper without Docker. To do that I recommend

- Algorithms (3)
- Architecture (20)
- Books (10)
- Building teams (11)
- Career (22)
- Chess (2)
- Choreography (8)
- Cloud (1)

```
@KafkaListener(topics = "topic1")
public void receiveTopic1(ConsumerRecord<?, ?> consumerRecord) {
    System.out.println("Receiver on topic1: "+consumerRecord.toString());
}
```

Spring Cloud Data Flow (4)

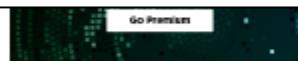
With Spring Boot, to use Kafka, you need a single dependency added to your POM file

<https://www.e4developer.com/2018/05/21/getting-started-with-kafka-in-spring-boot/>

Getting Started with



```
Application.properties
spring.kafka.consumer.group-id=kafka-intro
spring.kafka.bootstrap-servers=kafka:9092
```



ABOUT

E4developer is a place where I share my open

Bartosz Jedrzejewski
May 21, 2018
Choreography, Microservices, Spring Boot

```
@Autowired
private KafkaTemplate<String, String> kafkaTemplate;

public void send(String topic, String payload) {
    kafkaTemplate.send(topic, payload);
    System.out.println("Message: "+payload+" sent to topic: "+topic);
}
```

how tricky the installation can be. For that very reason, I have written "[How to easily run Kafka with Docker for development](#)" – I think you will find it especially useful if you are on Windows!

If you are running macOS or Linux, you can still follow the aforementioned tutorial, but you could also run Kafka and Zookeeper without Docker. To do that I recommend

- Algorithms (3)
- Architecture (20)
- Books (10)
- Building teams (11)
- Career (22)
- Chess (2)
- Choreography (8)
- Cloud (1)

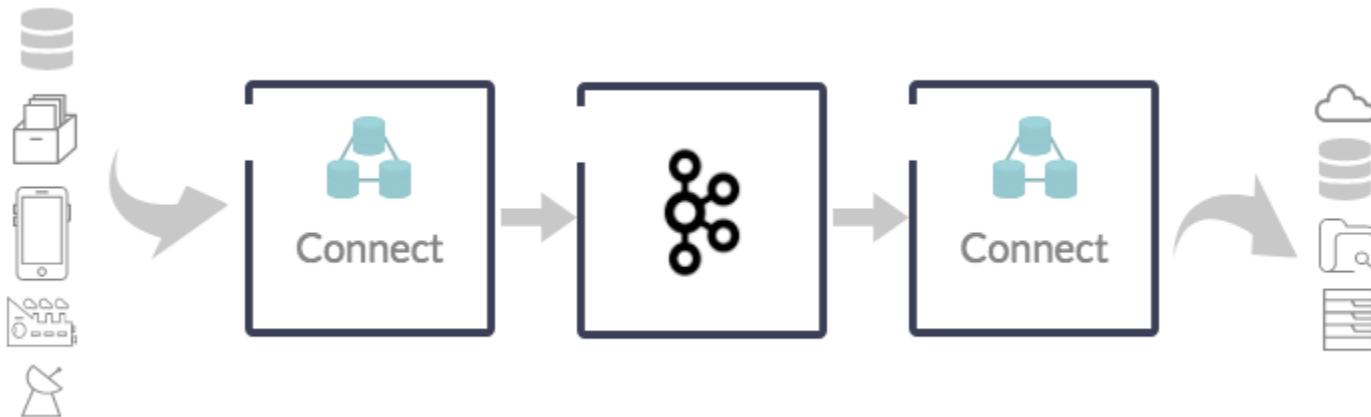
```
@KafkaListener(topics = "topic1")
public void receiveTopic1(ConsumerRecord<?, ?> consumerRecord) {
    System.out.println("Receiver on topic1: "+consumerRecord.toString());
}
```

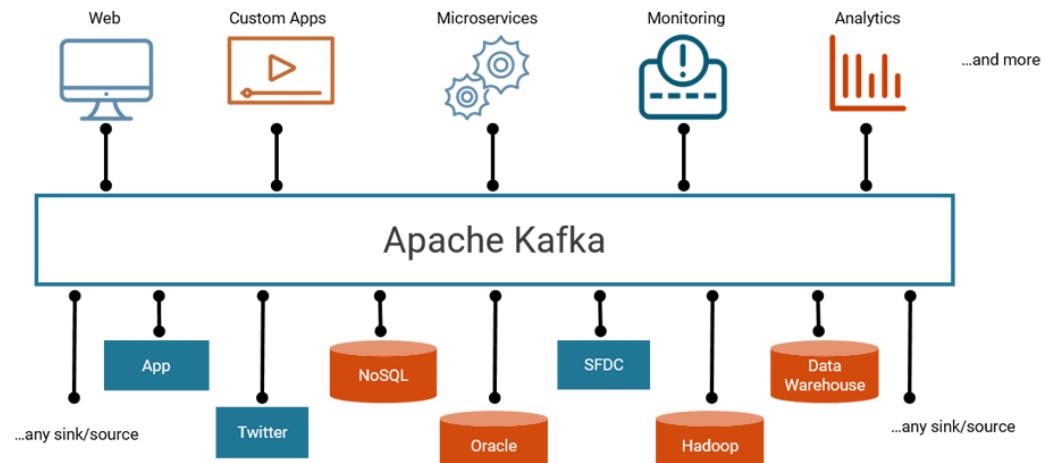
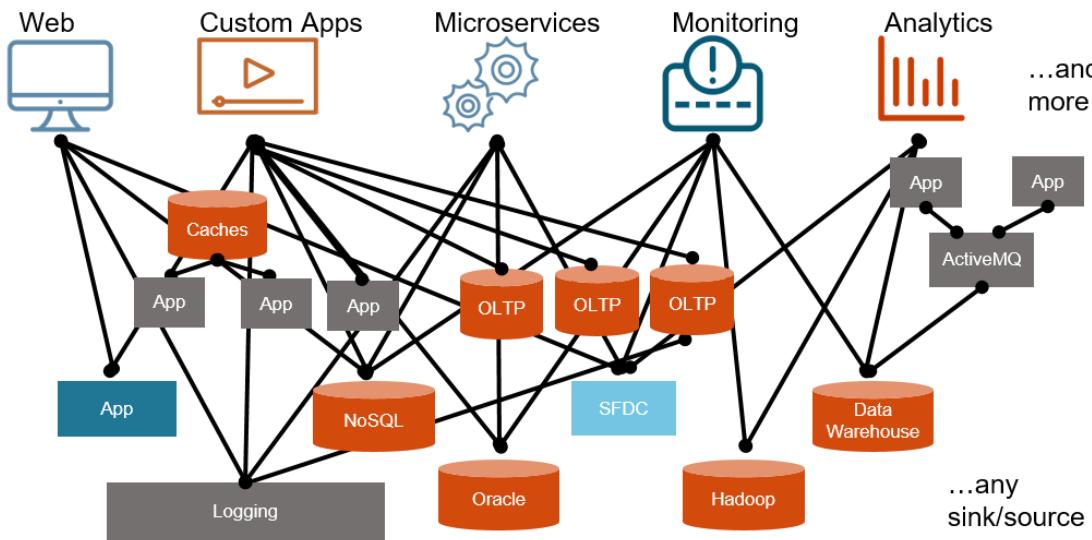
Spring Cloud Data Flow (4)

With Spring Boot, to use Kafka, you need a single dependency added to your POM file

<https://www.e4developer.com/2018/05/21/getting-started-with-kafka-in-spring-boot/>

“connectors”





<https://www.confluent.io/product/confluent-platform/>

José Maria Fernandes (jffernан@ua.pt) - August 2018



Open Studio for ESR

Open Source Downloads

Data Integration
Jumpstart your ETL projects and integrate data
[VIEW DETAILS →](#)
[DOWNLOAD NOW](#)

Big Data
Simplify ETL for large and diverse data sets
[VIEW DETAILS →](#)
[DOWNLOAD NOW](#)

Data Preparation
Enable users to discover, blend and clean data
[VIEW DETAILS →](#)
[WINDOWS DOWNLOAD](#)
[MAC DOWNLOAD](#)

Data Quality
Assess the accuracy and integrity of data
[VIEW DETAILS →](#)
[DOWNLOAD NOW](#)

Enterprise Service Bus
Speed up orchestration of applications and APIs
[VIEW DETAILS →](#)
[DOWNLOAD NOW](#)

Master Data Management
Generate a single "version of the truth" for data
[VIEW DETAILS →](#)
[DOWNLOAD NOW](#)

Speed up integration service reuse

Unlock the potential of more applications and resources than ever with Talend Open Studio for Enterprise Service Bus. It provides extensible open source technologies for building enterprise service-oriented architectures.

<https://www.talend.com/products/application-integration/esb-open-studio/>

Confluent Open Source

A developer-optimized distribution of Apache Kafka®. 100% open source.

Confluent, founded by the original creators of Apache Kafka® at LinkedIn, created a more complete distribution of Kafka, adding schema management, connectors, clients, and the highest level of testing in the industry.

[Download](#)

Why Confluent Open Source?



<https://www.confluent.io/product/confluent-open-source/>

More Languages

The majority of teams use two or

Better Data

Schema registry sets schemas for the

Fewer Problems

Packaged by the Kafka experts at

0

universidade de aveiro



The screenshot shows the Confluent Connectors page. At the top, there's a search bar labeled "Search connectors". Below it, a navigation bar includes "All", "Verified" (which is highlighted in bold), "Sources", "Sinks", and "Community". The main content area displays a grid of connector cards:

Category	Connector Name	Provider	Downloads	Rating (%)
Confluent	Kafka Connect Cassandra	Confluent, Inc.	31	100%
Confluent	Kafka Connect Avro Converter	Confluent, Inc.	27	100%
Confluent	Confluent Kafka Replicator	Confluent, Inc.	26	67%
Confluent	Kafka Connect IBM MQ	Confluent, Inc.	10	100%
Confluent	Kafka Connect ActiveMQ	Confluent, Inc.	9	100%
Confluent	Kafka Connect JMS	Confluent, Inc.	16	60%
Confluent	Kafka Connect S3	Confluent, Inc.	32	80%
Confluent Preview	Kafka Connect Salesforce	Confluent, Inc.	31	100%
Confluent Preview	(Placeholder)	(Placeholder)	(Placeholder)	(Placeholder)
Confluent Preview	(Placeholder)	(Placeholder)	(Placeholder)	(Placeholder)
Confluent	Java DB	(Placeholder)	(Placeholder)	(Placeholder)

On the left side of the page, there's a sidebar with the text "A developer's guide to Kafka Connect" and a "Download" button. At the bottom, there are icons for a browser, binary code, and a terminal window, along with the text "Better Data", "Fewer Problems", and "Packaged by the Kafka experts at".

<https://www.confluent.io/product/confluent-open-source/>

More Languages

The majority of teams use two or

Better Data

Schema registry sets schemas for the

Fewer Problems

Packaged by the Kafka experts at



UNLOCK YOUR DATA.

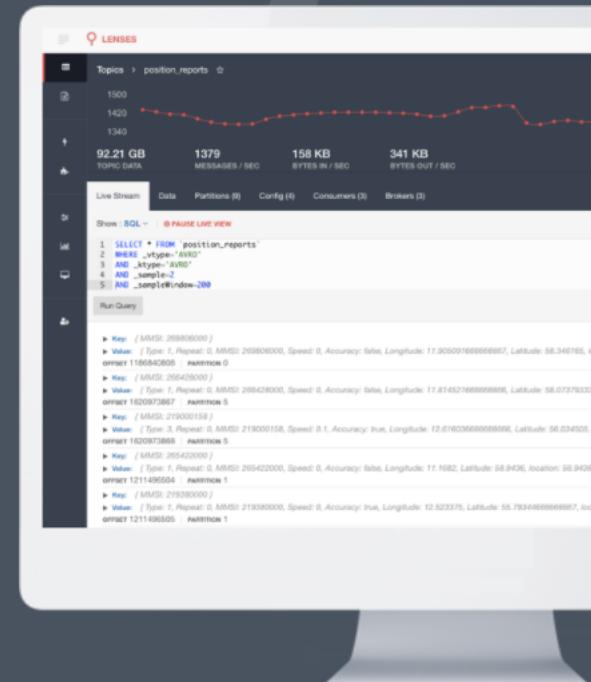
Stream. Analyze. React.

Quick Insights.

- ✓ Demystify complexity
- ✓ Access & Integrate your data in motion
- ✓ Connect to various systems with 25+ open source Kafka connectors

UPGRADE TO STREAMING ETL IN MINUTES

- ✓ Build and deploy real-time Kafka based pipelines in minutes
- ✓ Native scale with Kubernetes - the powerful container orchestrator
- ✓ No code required

[DOWNLOAD](#)[REQUEST DEMO](#)

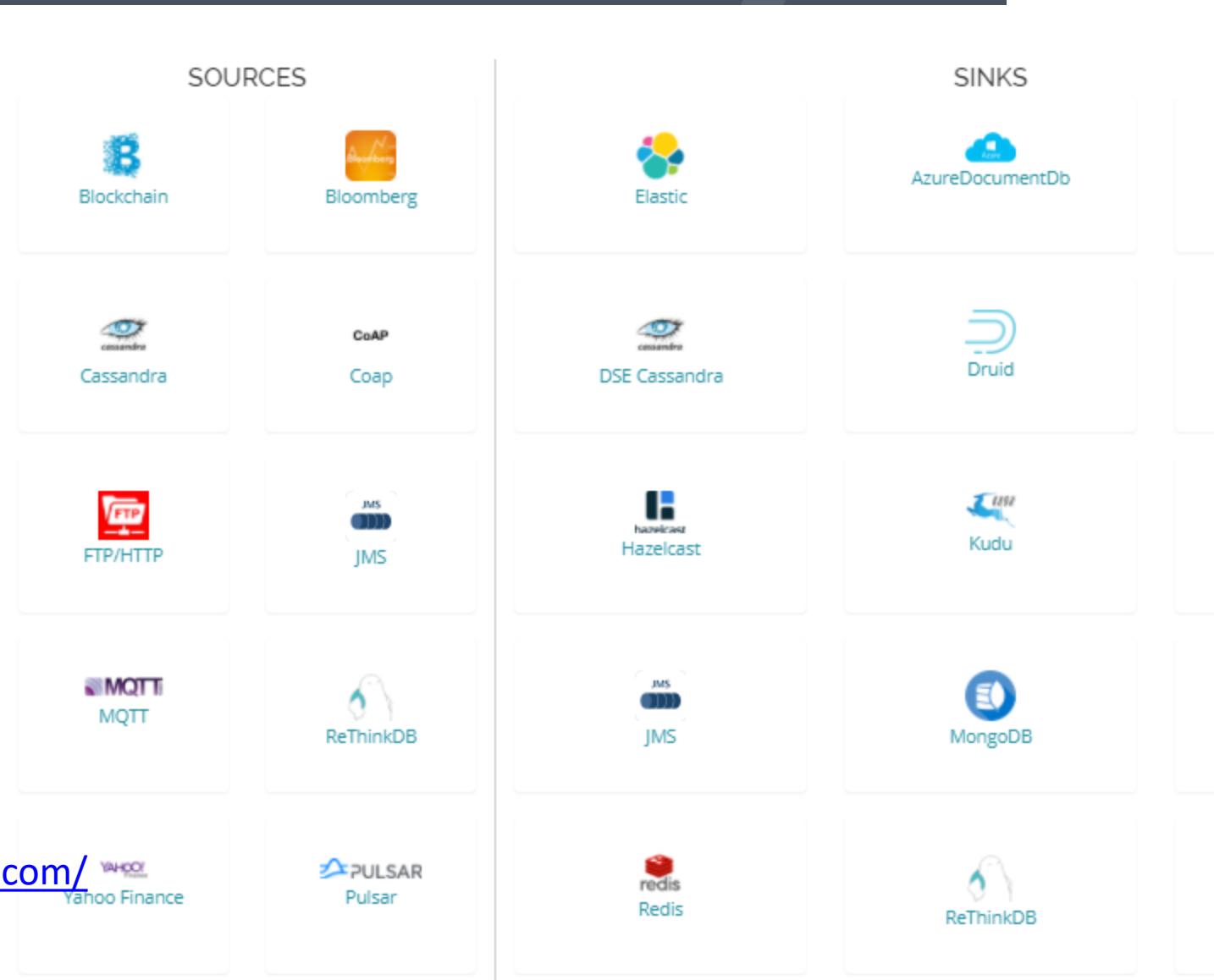
<http://www.landoop.com/>

<http://www.landoop.com/kafka-lenses/>

UNLOCK YOUR DATA.

Stream. Analyze. Read.
Quick Insights.

- ✓ Demystify complexity
- ✓ Access & Integrate you in motion
- ✓ Connect to various systems with 25+ open source kind connectors

[DOWNLOAD](#)[REQUEST](#)

<http://www.landoop.com/>
YAHOO Finance



1.1

Search docs

Installation & Setup

User Guide

Developer Guide

Lenses SQL Engine

Connectors

Connect CLI

Source Connectors

Sink Connectors

Cloudera Integration

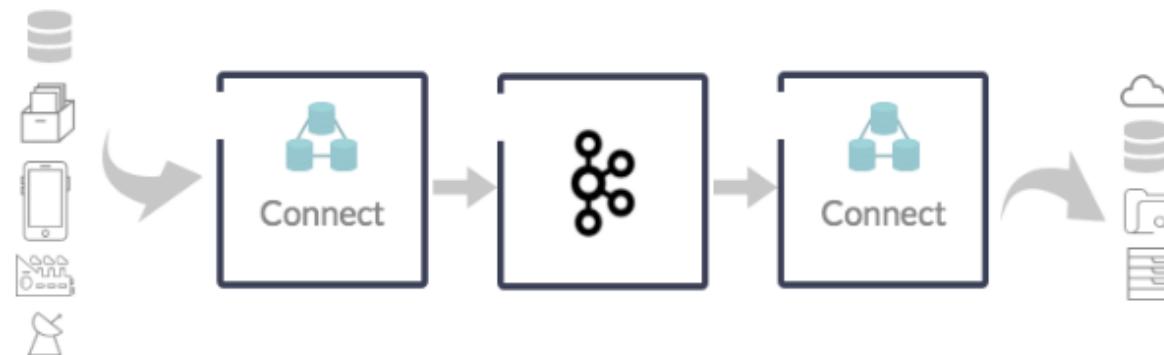
Connector Install

FAQ

Release Notes

Kafka Connect is a tool to rapidly stream events in and out of Kafka. It has a narrow focus on data ingress in and egress out of the central nervous system of modern streaming frameworks. Connect allows developers to quickly build robust, durable and scalable pipelines in and out of Kafka.

Landoop provides a large collection of Connectors bundled in the Stream Reactor. This open source set of components can be used as the entry point for building your streaming data flows.



Kafka Connect can run in standalone or distributed mode. The former is targeting testing or one-off jobs whereas the latter provides a scalable, fault-tolerant service supporting an entire organization. Connect can scale down for development, to support production workloads.

<https://lenses.stream/1.1/connectors/index.html>

v1.1 v2.1 v2.0 v1.0



1.1

Search docs

Installation & Setup

User Guide

Developer Guide

Lenses SQL Engine

Connectors

Connect CLI

Source Connectors

Sink Connectors

Cloudera Integration

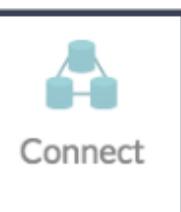
Connector Install

FAQ

Release Notes

Source connectors load

- Blockchain Source
- Bloomberg Source
- Cassandra Source
- Cassandra CDC
- CoAP Source
- FTP Source
- JMS Source
- Mqtt Source
- Pulsar Source
- ReThink Source



Connect

Kafka Connect can run in standalone mode for testing or as an one-off jobs whereas

Sink connectors stream data f

- Azure DocumentDB Sink
- Cassandra Sink
- CoAP Sink
- Elastic Sink
- Elastic 5 Sink
- HazelCast Sink
- HBase Sink
- Influx Sink
- JMS Sink
- Kudu Sink
- Mongo Sink
- MQTT Sink
- Pulsar Sink
- Redis Sink
- ReThink Sink
- VoltDB Sink



getting
nt

to support production workloads.

Blog Article

Cloud Connectors

02 Jun 2017

 [Tweet](#) [Share](#) 

Currently we have JCA adapters for:

- **Apache Kafka** - sending messages using a Connection Factory and receiving messages via an MDB.
- **Amazon SQS** - sending messages using a Connection Factory and receiving messages via an MDB to Amazon Simple Queue Service queues.
- **MQTT** - sending messages using a Connection Factory and receiving messages via an MDB to an MQTT broker or to IOT hubs that support MQTT.
- **Azure Service Bus** - Sending and receiving messages to/from Azure Service Bus Queues using ConnectionFactory via MDB.

Payara Micro 172 brings with it support for JCA adapters, meaning it can be

used at <http://blog.payara.fish/cloud-connectors-in-payara-micro>

[Subscribe](#)



Code

Issues 6

Pull requests 1

Projects 1

Insights



Join GitHub today

GitHub is home to over 28 million developers working together to host and review code, manage projects, and build software together.

[Sign up](#)[Dismiss](#)

JCA Connectors for Java EE7 that connect to Cloud Services

[payara](#) [javaee](#) [kafka](#) [mqtt](#) [sq](#) [payaramicro](#) [jca](#) [rar](#) [amazon-sqs](#) [azure-service-bus](#)

52 commits

1 branch

3 releases

6 contributors

[View license](#)Branch: master ▾ [New pull request](#)[Find file](#)[Clone or download](#) ▾

Pandrex247 Merge pull request #52 from Pandrex247/Increment-Version-Numbers ...

Latest commit f6ff9e7 on Jun 15

[AmazonSQS](#)[AzureServiceBu](#)[Kafka](#)[MQTT](#)[.gitignore](#)[License.md](#)[README.md](#)[pom.xml](#)

Currently we have JCA adapters for:

- Apache Kafka - sending messages using a Connection Factory and receiving messages via an MDB
- Amazon SQS - sending messages using a Connection Factory and receiving messages via an MDB to Amazon Simple Queue Service queues.
- MQTT - sending messages using a Connection Factory and receiving messages via an MDB to an MQTT broker or to IOT hubs that support MQTT.
- Azure Service Bus - Sending and receiving messages to/from Azure Service Bus Queues using ConnectionFactory via MDB

[README.md](#)

Payara Cloud Connectors

Payara Cloud Connectors is a project to provide JavaEE standards based connectivity to common Cloud infrastructure. Utilising ICA we provide connectivity to many different services provided by the leading cloud providers and open source technologies.



PROJECTS

Spring Integration



Extends the Spring programming model to support the well-known [Enterprise Integration Patterns](#). Spring Integration enables lightweight messaging within Spring-based applications and supports integration with external systems via declarative adapters. Those adapters provide a higher-level of abstraction over Spring's support for remoting, messaging, and scheduling. Spring Integration's primary goal is to provide a simple model for building enterprise integration solutions while maintaining the separation of concerns that is essential for producing maintainable, testable code.

[QUICK START](#)

<https://projects.spring.io/spring-integration/>

Guides

Whatever you're building,
releases and techniques

Have a suggestion for a

Find a guide...



Getting Started
Design Patterns
World Class

Building a RESTful Web Service
Learn how to create a RESTful web service with Spring.

Building a RESTful Web Service

Learn how to create a RESTful web service with Spring.

Building Java Projects with Gradle

Learn how to build a Java project with Gradle.

Uploading Files

Learn how to build a Spring application that accepts multi-part file uploads.

Messaging with Redis

Learn how to use Redis as a message broker.

Accessing Twitter Data

Learn how to access user data from Twitter.

Validating Form Input

Learn how to perform form validation with Spring.

Scheduling Tasks

Learn how to schedule tasks with Spring.

Building Java Projects with Maven

Learn how to build a Java project with Maven.

Authenticating a User with LDAP

Learn how to secure an application with LDAP.

Registering an Application with Twitter

Learn how to register apps with Twitter.

Accessing Facebook Data

Learn how to access Facebook information from an application.

Building a RESTful Web Service with Spring Boot Actuator

Learn how to create a RESTful Web service with Spring Boot Actuator.

Consuming a RESTful Web Service

Learn how to retrieve web page data with Spring's RestTemplate.

Accessing Relational Data using JDBC with Spring

Learn how to access relational data with Spring.

Registering an Application with Facebook

Learn how to register an application to integrate with Facebook.

Messaging with RabbitMQ

Learn how to create a simple publish-and-subscribe application with Spring and RabbitMQ.

Accessing Data with Neo4j

Learn how to persist objects and relationships in Neo4j's NoSQL data store.

Messaging with JMS

Learn how to publish and subscribe to messages using a JMS broker.

<https://spring.io/guides>

Creating a Batch Service

Securing a Web Application

Building a Hypermedia-Driven

Take home

- Messaging
 - High level solution, not only brokers
 - Supports reactive, events , decoupling
- Same (new) patterns
 - Mostly different scope
 - Conceptually equivalent
 - Cloud, soa, integration
- Don't mix architecture & implementation
 - Mix if need asynch, synch , message , REST, ...

Messaging

- Conceptual
 - Reduce coupling
 - Focus on domain
 - Scalable
- Patterns and styles associated
 - Publisher / subscriber
 - Cqrs, Event sourcing
 - Event Driven Architecture
 - ...
- Technology agnostic
 - However commonly associated with concrete broker solutions

Take home...

- Messaging
 - A (good) trend
 - Brokers, message, events, ...
- Kafka
 - One option
 - is a worthy tool:
 - Integration – data & developer
 - Many roles and some to be invented
- Connectors
 - New trend, not only for messaging
 - Look and don't reinvent the wheel

Just a mention

- Runtime cannot be ignored
 - limited resources Kafka runtime may be too “heavy”
 - Different versions / protocols
- Websocket
 - P2P solutions may enough
 - Similar reasoning, different solution
- MQTT solutions may be interesting
 - Mosquitto and plenty of lightweight clients
- RabbitMQ
 - Multi protocol support
 - Multi platform support
- Addressed on a need to know basis (mostly)

ActiveMQ Confusion and What comes with your JBoss EAP / WildFly

Datetime: 2017-02-22 05:26:58

Topic: Jboss

Share

[Original >>](#)

Here to See The Original Article!!!

[Test](#)[Option](#)[Applications](#)[Performance Management](#)[Actually](#)[infolinks](#)

Oftentimes
to different
their name a
a broker with
name 'Activ

So there are

- 'ActiveMQ project is used many config'
- 'ActiveMQ Artemis' Active

DISTRIBUTION	BROKER	VERSION	PROTOCOL SUPPORT
Artemis standalone	Artemis	1.5	AMQP, STOMP, MQTT, OpenWire, HornetQ Core
ActiveMQ 5 standalone	ActiveMQ 5	5.14	AMQP, STOMP, MQTT, OpenWire
WildFly 8, 9	HornetQ	2.4	AMQP, STOMP, HornetQ Core
JBoss EAP 6.4	HornetQ	2.3	AMQP, STOMP, HornetQ Core
WildFly 10	Artemis	1.1	AMQP, STOMP, HornetQ Core
JBoss EAP 7	Artemis	1.1	HornetQ Core

ActiveMQ 5 with some radical architecture changes to improve performance, reliability and maintainability. This project seems to have become dormant since the introduction of the 3rd ActiveMQ project:

- 'ActiveMQ Artemis' is not a fork or a reimplementation but rather started as the HornetQ broker. HornetQ's code was donated to the ActiveMQ project and is now developed by the developers of HornetQ under the new name ActiveMQ Artemis. So basically it's a completely different and field proven broker. Artemis is known to be rather fast for a JMS broker. Artemis received and is still receiving a lot of changes to build it up to the feature set of ActiveMQ 5. The last official statement I heard is that Artemis might at some point become the successor of ActiveMQ 5, but that is not decided yet. The [Wikipedia](#) article for

ActiveMQ Confusion and What comes with your JBoss EAP / WildFly

<http://126kr.com/article/6vk09hqaq7n>

The project I usually think of when I hear ActiveMQ is the main project ActiveMQ 5. However, the latest WildFly, as well as RedHat's EAP 7, distribute ActiveMQ Artemis as their JMS broker.



Eclipse Mosquitto™

An open source MQTT broker

Eclipse Mosquitto is an open source (EPL/EDL licensed) message broker that implements the MQTT protocol versions 3.1 and 3.1.1.

Mosquitto is lightweight

The MQTT protocol provides a publish/subscribe message delivery model designed for Internet of Things messages, sensors, actuators and microcontrollers.

The Mosquitto project also includes a command line MQTT client and a test harness.

Mosquitto is part of the Eclipse Project.

Downloads

Mosquitto is highly portable and available for a wide range of platforms. Go to the Downloads page to learn more.



The screenshot shows the Eclipse Paho website homepage. At the top, there is a navigation bar with the Eclipse Paho logo, a search bar, and links for Components, Documentation, and Community. The main feature is a large, stylized "paho" logo where each letter is a different color (yellow, orange, green, brown) and the "o" is replaced by a white speech bubble icon. Below the logo, a text block reads: "The Eclipse Paho project provides open-source client implementations of MQTT and MQTT-SN messaging protocols aimed at new, existing, and emerging applications for the Internet of Things (IoT)." A prominent blue button at the bottom right says "Download Now »".

<https://mosquitto.org/>

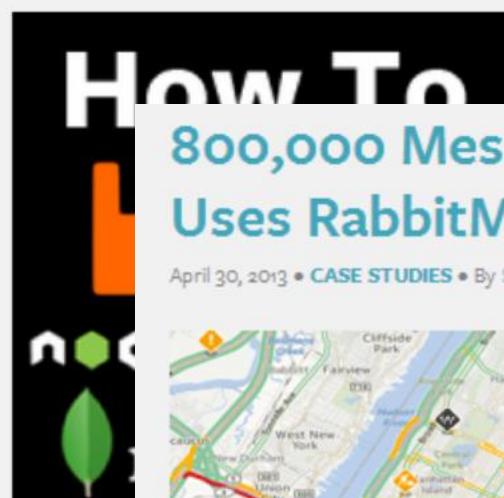
<https://www.eclipse.org/paho/clients/js/>

RabbitMQ, ActiveMQ,...



How to Apply Messaging to Cloud Apps with RabbitMQ, Node.js, Redis, and More

August 19, 2013 • PRODUCTS • By Adam Bloom



below.

Recently, Cloud Foundry developer advocate, Alvaro Videla, gave a talk at the Norwegian Developer's

800,000 Messages/Minute: How Nokia's Uses RabbitMQ to Make Real-time Traffic Maps

April 30, 2013 • CASE STUDIES • By Stacey Schneider



Back in 2008, Nokia acquired HERE Maps which has since become part of Nokia's Location and Navigation business, renamed HERE.

Based in Chicago, IL, the HERE team ingests tens of thousands of data points every second from sensors around the globe including

<https://www.rabbitmq.com/>
<http://activemq.apache.org/>
<http://www.gridshore.nl/2011/07/28/combining-java-and-nodejs/>
<http://blog.gopivotal.com/products/how-to-apply-messaging-to-cloud-applications>
<http://blog.cloudfoundry.com/tag/redis/>

Why present RabbitMQ?

- A good example, not the only one
- Multilanguage and framework solution
- Good solution to integrate software

The screenshot shows the official RabbitMQ website. At the top is the logo 'RabbitMQ' with the tagline 'Messaging that just works'. To the right, it says 'by Pivotal™'. Below the logo is a navigation bar with links: Features, Installation, Documentation, Tutorials, Services, Community, and Blog. There is also a search bar labeled 'Search RabbitMQ' with a magnifying glass icon.

This section is titled '1 "Hello World!"'. It describes it as 'The simplest thing that does something'. It includes a diagram showing a blue circle labeled 'P' connected by an arrow to a red rectangle representing a queue, which then has an arrow pointing to a blue circle labeled 'C'. Below the diagram, there are language options: Python | Java | Ruby | PHP | C#.

Introduction

RabbitMQ is a message broker. In essence, it accepts messages from *producers*, and delivers them to *consumers*. In-between, it can route, buffer, and persist the messages according to rules you give it.

RabbitMQ, and messaging in general, uses some jargon.

› *Producing* means nothing more than sending. A program that sends messages is a *producer*. We'll draw it like that, with "P":



This section is titled '2 Work queues'. It describes it as 'Distributing tasks among workers'. It includes a diagram showing a blue circle labeled 'P' connected by an arrow to a red rectangle representing a queue, which then branches into two arrows pointing to two blue circles labeled 'C1' and 'C2'. Below the diagram, there are language options: Python | Java | Ruby | PHP | C#.

› A *queue* is the name for a mailbox. It lives inside RabbitMQ. Although messages flow through RabbitMQ and your applications, they can be stored only inside a *queue*. A *queue* is not bound by any limits, it can store as many messages as you like - it's essentially an infinite buffer. Many *producers* can send messages that go to one *queue* - many *consumers* can try to receive data from one *queue*. A *queue* will be drawn like this, with its name above it:



Understanding When to Use RabbitMQ or Apache Kafka

Understanding When to Use RabbitMQ or Apache Kafka

RabbitMQ and Apache Kafka are two of the most popular messaging technologies on the market today. Get the insight you need to choose the right software for you.

by Pieter Humphrey R IBM · May, 01, 17 · Integration Zone

Like (26) Comment (4) Save Tweet 13.12k Views

Share, secure, distribute, control, and monetize your APIs with the platform built with performance, time-to-value, and growth in mind. Free 90-day trial of 3Scale by Red Hat

How do humans make decisions? In everyday life, emotion is often the circuit-breaking factor in pulling the trigger on a complex or overwhelming decision. But for experts making complex decisions that have long term consequences, it can't be pure impulse. High performers typically use the circuit breaker of "instinct," "gut feeling," or other emotions only once their expert, unconscious mind has absorbed all the facts required to make a decision.

Today there are dozens of messaging technologies, countless ESBs, and nearly 100 iPaaS vendors in the market. Naturally, this leads to questions about how to choose the right messaging technology for your needs - particularly for those already invested in a particular choice. Do we switch wholesale? Just use the right tool for the right job? Have we correctly framed the job at hand for the business need? Given that, what is the right tool for me? Worse, an exhaustive market analysis might never finish, but due diligence is critical given the average lifespan of integration code.

This post endeavors to give the unconscious, expert mind some even-handed treatment to consider, starting with the most modern, popular choices today: RabbitMQ and Apache Kafka. Each has its own origin story, design intent, uses cases where it shines, integration capabilities, and developer experience. Origins are revealing about the overall design intent for any piece of software, and make a good starting point.

Origins

RabbitMQ is a "traditional" message broker that implements a variety of messaging protocols. It was one of the first open source message brokers to achieve a reasonable level of features, client libraries, dev tools, and quality documentation. RabbitMQ was originally developed to implement AMQP, an open wire protocol for messaging with powerful routing features. While Java has messaging standards like JMS, it's not helpful for non-Java applications that need distributed messaging, which is severely limiting to any integration scenario, microservices or monolithic. With the advent of AMQP, cross-language flexibility became real for open source message brokers.

Apache Kafka is developed in Scala and started out at LinkedIn as a way to make data ingest to Hadoop from Apache Flume easier. Ingesting and exporting from multiple data sources and destinations with tools like Flume meant writing separate data pipelines for each source and destination pairing. Kafka helped LinkedIn standardize the data pipelines and allowed getting data out of each system once and into each system once, making pipelines (and operation) simpler. Kafka is well adopted today within the Apache Software Foundation ecosystem of projects. In particular, it is well integrated with Hadoop, Spark, Flink, and Storm.

While it is a cluster

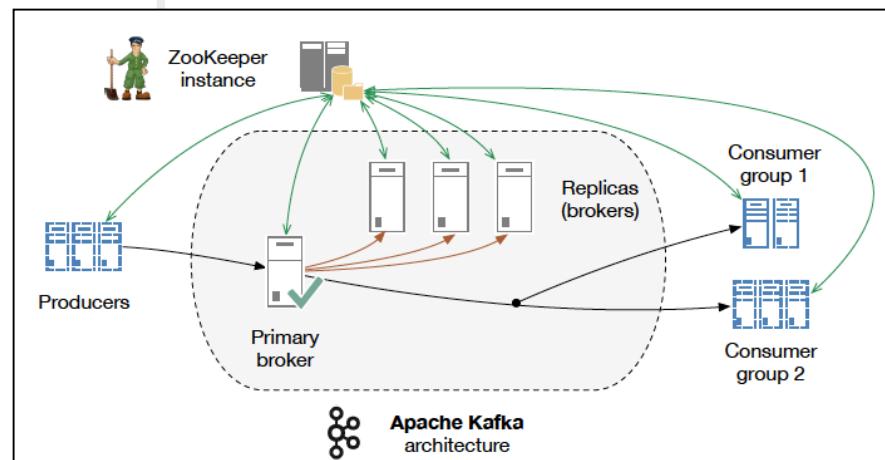
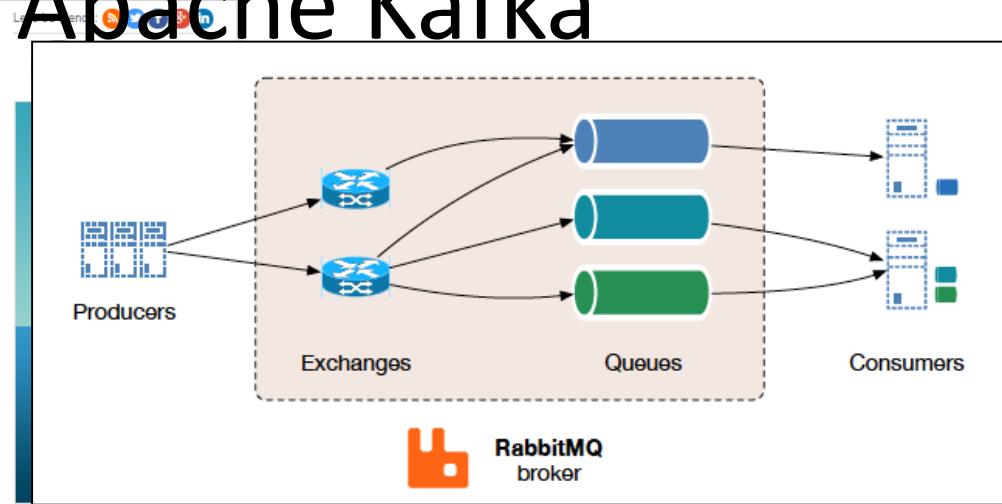
Developer Experience Security and Operations Performance

Arch

RabbitMQ
point, r
consum
similar
correct
langua
scenari

...

<https://dzone.com/articles/understanding-when-to-use-rabbitmq-or-apache-kafka>



The END