Messaging using AMQP

Rahul Agrawal

Advanced Message Queuing Protocol

- Why should you care ?
  - Many significant IT efforts include a messaging and integration component (10%-30% of project cost) .

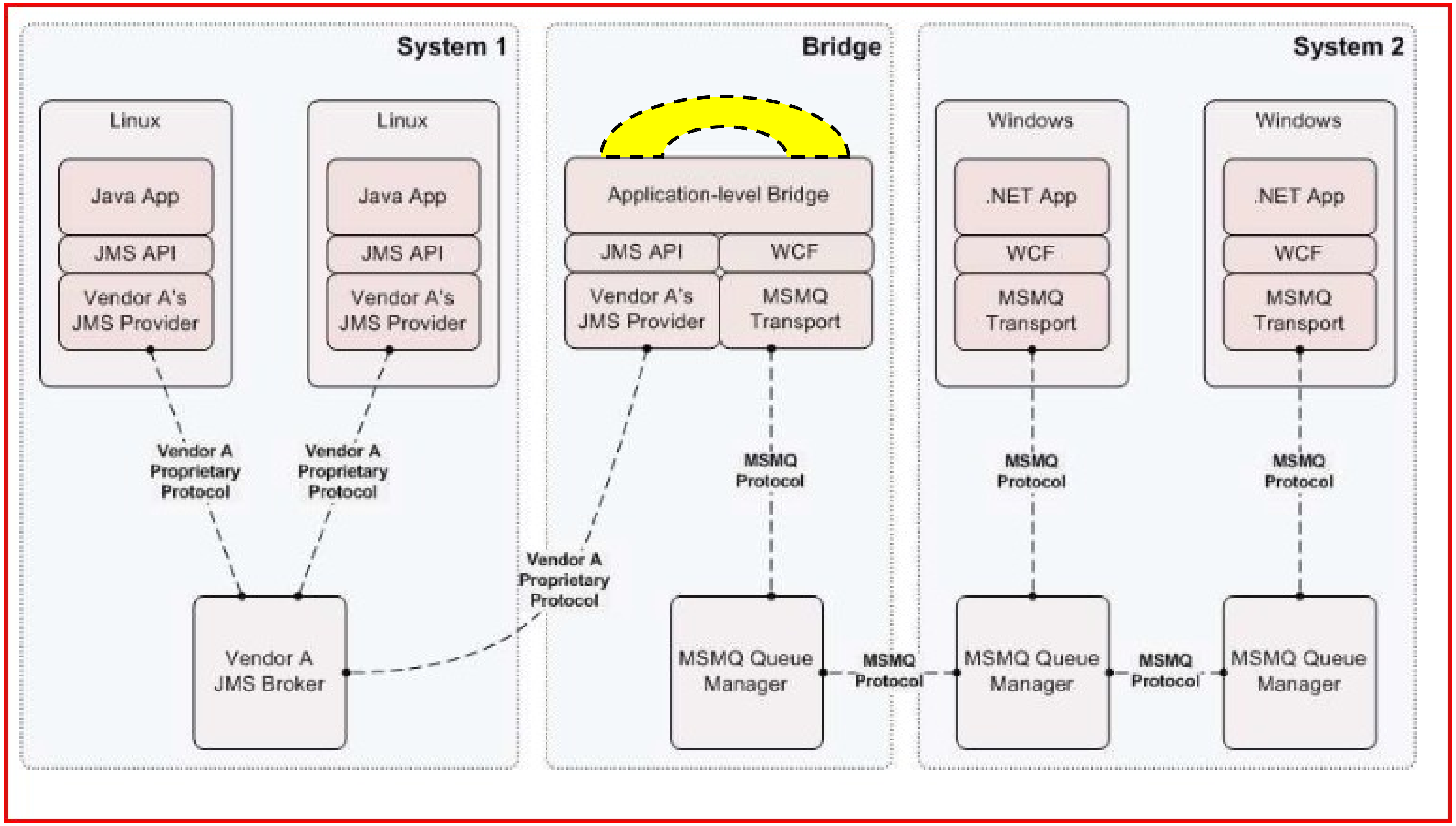
Why Messaging ?

- Transfer data from point A to point B.
- Asynchronous.
- Decouple publishers and consumers.
- Queuing for later delivery.

Limitations of exiting MOMs

- Proprietary middleware has been a source of lock-in.
  - IBM MQ, Tibco Rendezvous , Sonic MQ.
- Interoperability is more difficult than it need be.
  - MQ Series and Tibco RV cannot natively interoperate with each other or other middleware products.
- Language and platform independence is still an issue.
  - JMS is not technology agnostic and only legitimately supports Java platforms.

Application Level Bridging



AMQP

- Conceived by JP Morgan around 2006.
- Goal was to provide a vendor-neutral protocol for managing the flow of messages across enterprise's business systems.

How does AMQP solve the problem?

- AMQP is a wire level protocol and not an API.
  - JMS is an API.
  - Just like HTTP is for Internet, AMQP is for messaging.

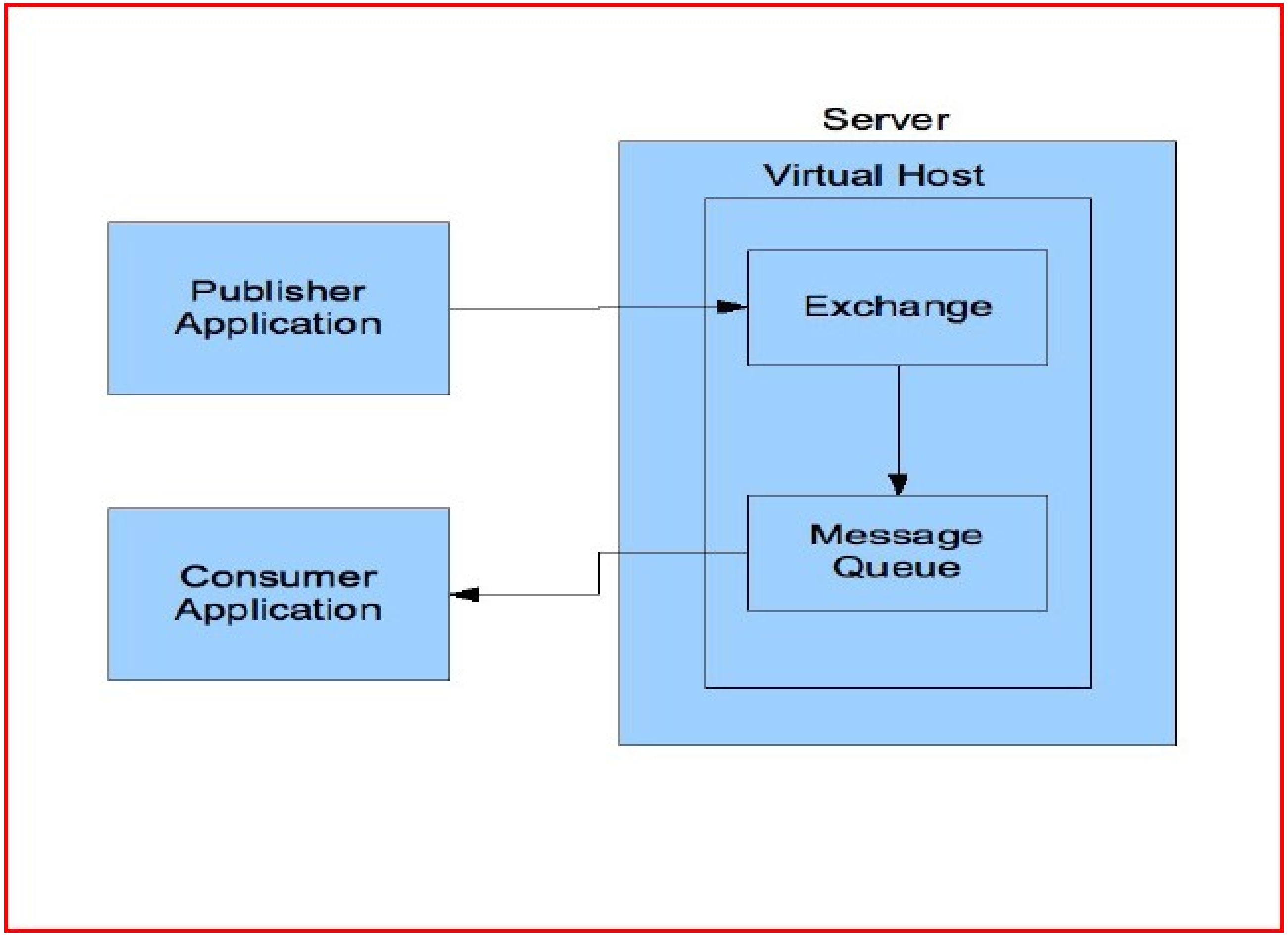
- When a protocol is specified at the wire-level and published, most technologies can use it.
- Compare this to an API, where the actual implementation is specific to the platform.

AMQP : Industry Support

- Cisco Systems
- Goldman Sachs
- IONA
- Novell
- Redhat
- WSO2
- Microsoft
- Credit Suisse
- Envoy Technologies
- iMatix
- JPMorgan Chase
- Rabbit Technologies
- TWIST
- 29West

AMQP Concepts

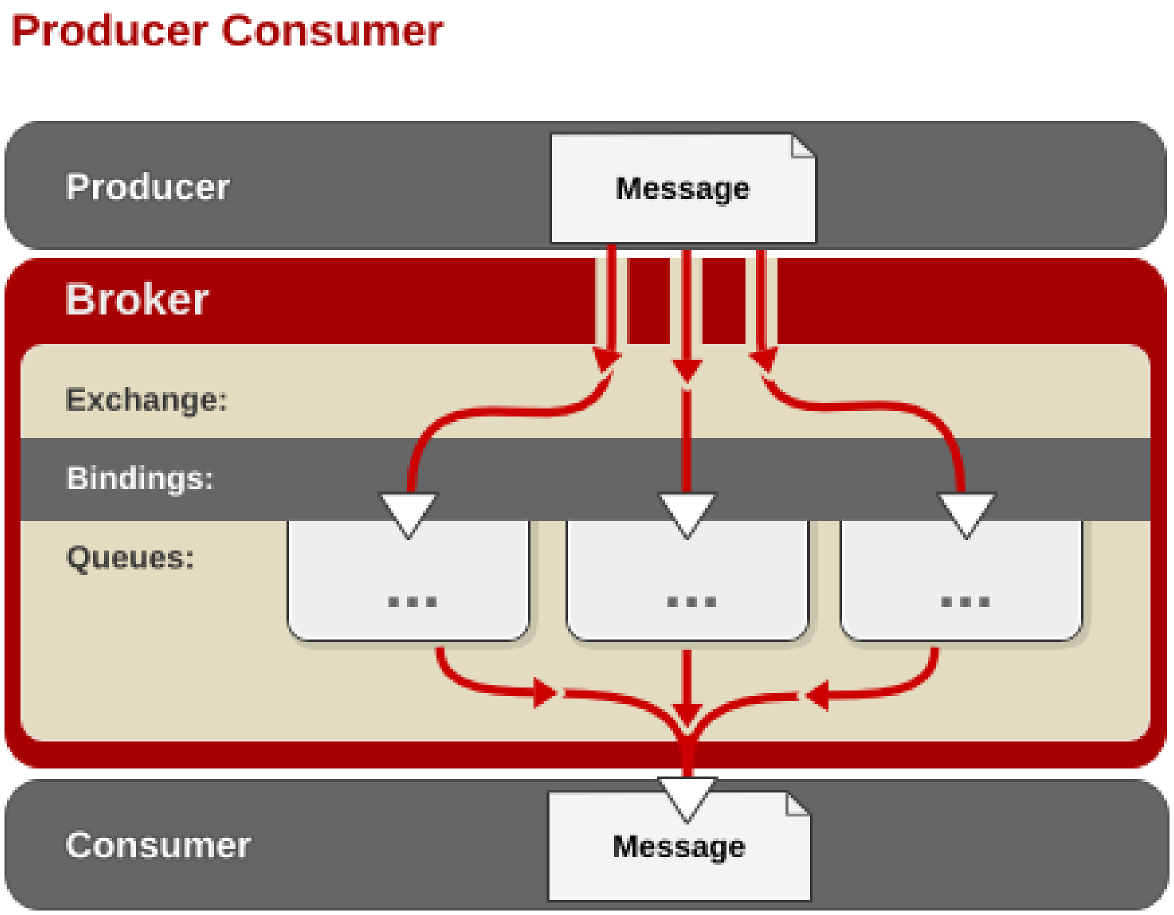
Virtual Hosts



Virtual Hosts

- Each virtual host comprises its own name space, a set of exchanges, message queues and all associated objects.

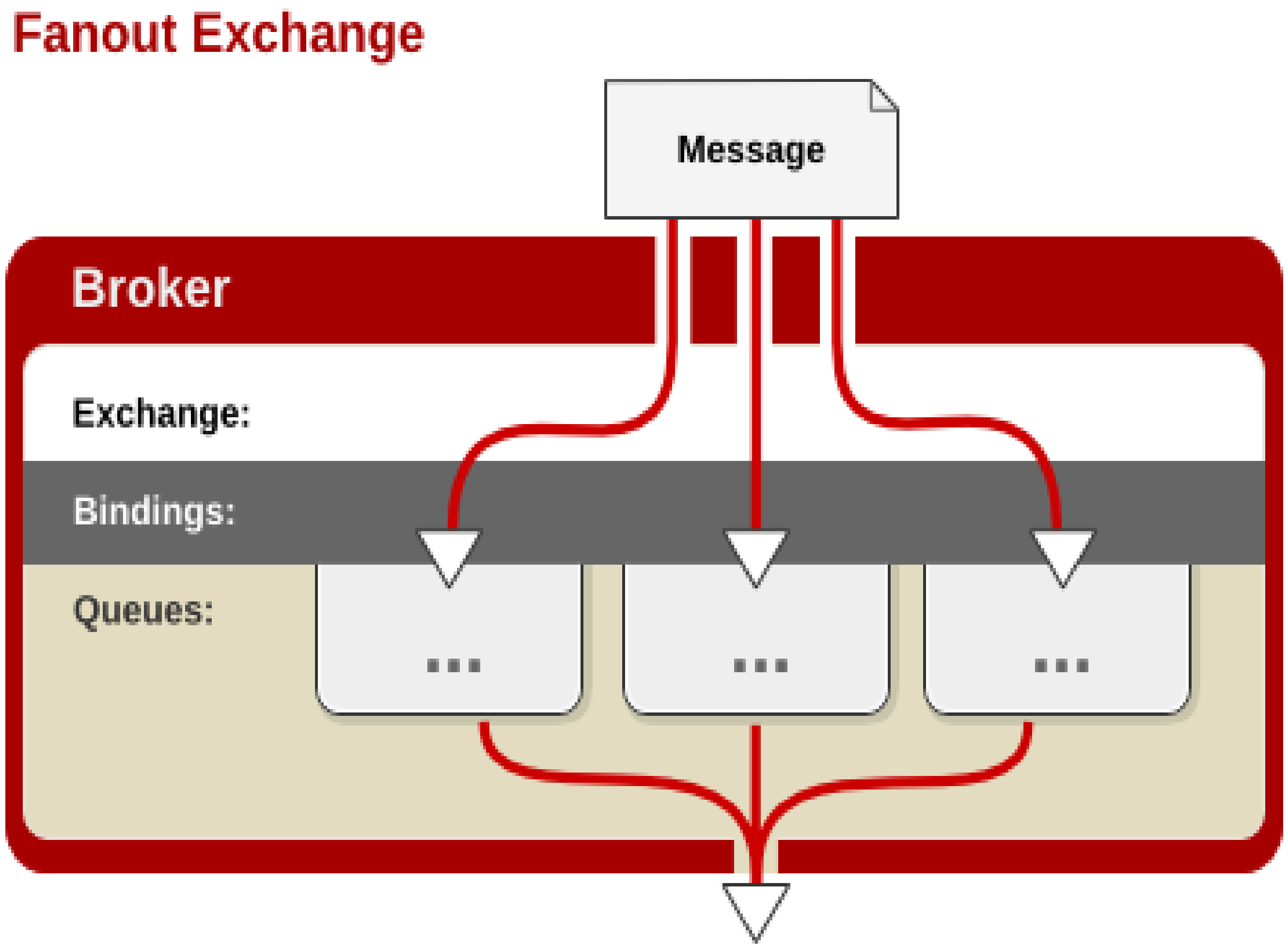
AMQP concepts



Exchange, Binding and Queues

- The "exchange" receives messages from publisher and routes these to “Queues".
- The "binding" defines the relationship between a message queue and an exchange and provides the message routing criteria.

Exchange Types

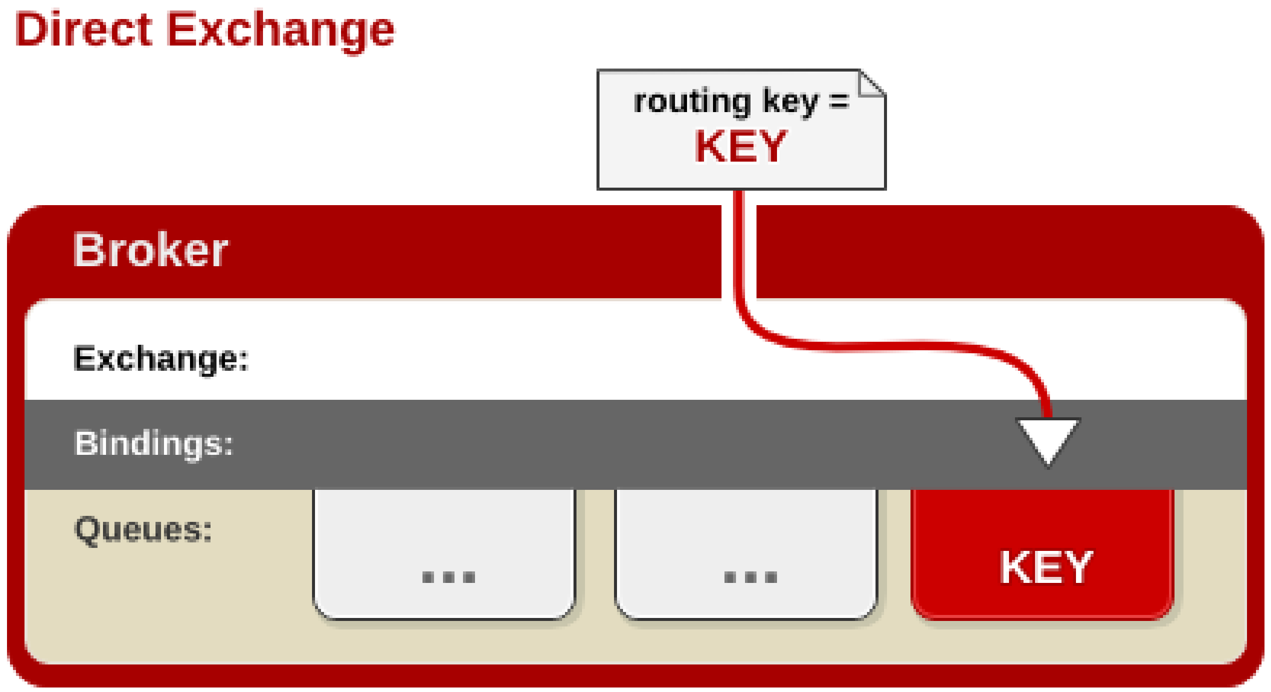


A Fanout exchange sends messages to every queue bound to the exchange

Fanout Exchange Type

- No routing key.
- What goes in must go out.

Exchange Types

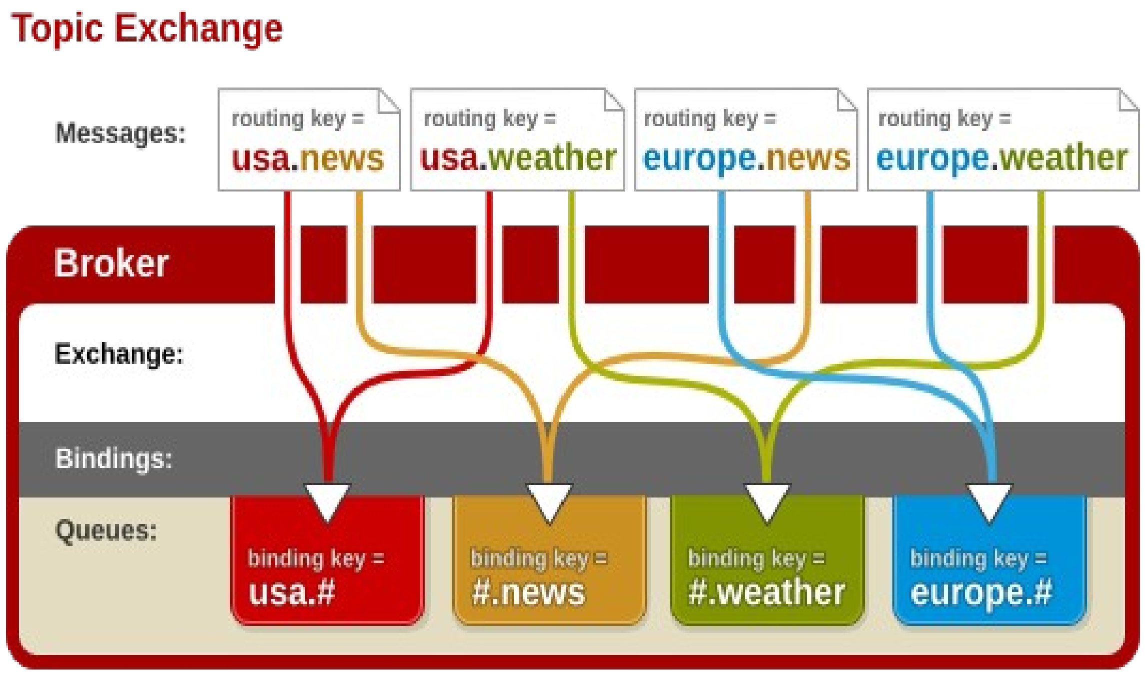


A Direct exchange sends a message to a queue if the message's routing key is identical to the binding key for the queue.

Direct Exchange Type

- Uses String as routing key.
- Queue binds to exchange with key K.
- Publisher sends a message with key R.
- Message is passed to this queue if K=R.
- Direct exchange named “amq.direct” is always pre-created in each virtual host

Exchange Types



Topic Exchange Type

- Uses pattern as routing key (“a.b.c.d”).
- Queues can use wildcard characters in binding key.
- * matches a single word.
- # matches zero or more words.
- “amq.topic” is pre-created in each virtual host.
- *.stock.# matches usd.stock and eur.stock.db but not stock.nasdaq

AMQP Broker Implementations

- Apache Qpid : http://qpid.apache.org
  - Java & C++ Implementation.
- Rabbit MQ : http://www.rabbitmq.com
  - Erlang Implementation.
- Red Hat's MRG (Messaging, Realtime & Grid) : http://www.redhat.com/mrg/
  - C++ Implementation.

Apache Qpid: Open Source AMQP Messaging

- **AMQP Messaging Brokers**
  - C++ implementation
  - Java implementation
- **AMQP Client APIs: C++, Java, JMS, Ruby, Python, and C#**

Demo

- Java implementation of the broker.
- JMS clients interacting with Python clients.