Next Generation Open Source Messaging with **Apache Apollo**

Hiram Chirino
Red Hat Engineer
Blog: http://hiramchirino.com/blog/
Twitter: @hiramchirino
GitHub: https://github.com/chirino

About me

Hiram Chirino

Blog: http://hiramchirino.com/blog/
Twitter: @hiramchirino
GitHub: https://github.com/chirino

- Engineer at Red Hat
- Apache Committer on: ActiveMQ, Camel, Karaf, ServiceMix, Geronimo, Felix, and Aries
- Apache Member and ActiveMQ PMC Chair
- Co-Founder of many other OS projects:
  - HawtDispatch, Scalate, LevelDBJNI, Jansi, And many more!

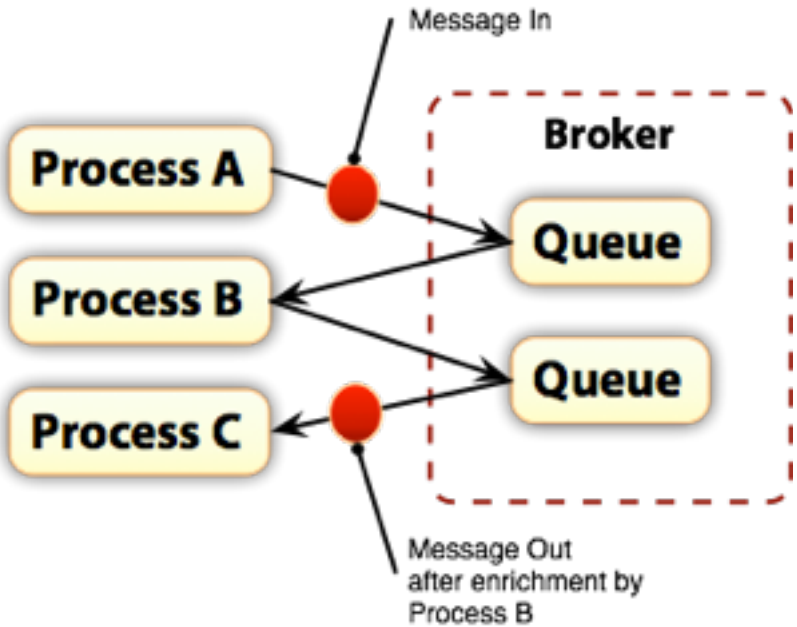
Outline

- What is Apache Apollo?
- What makes it different?
- What's the trajectory?

What is Apache Apollo?

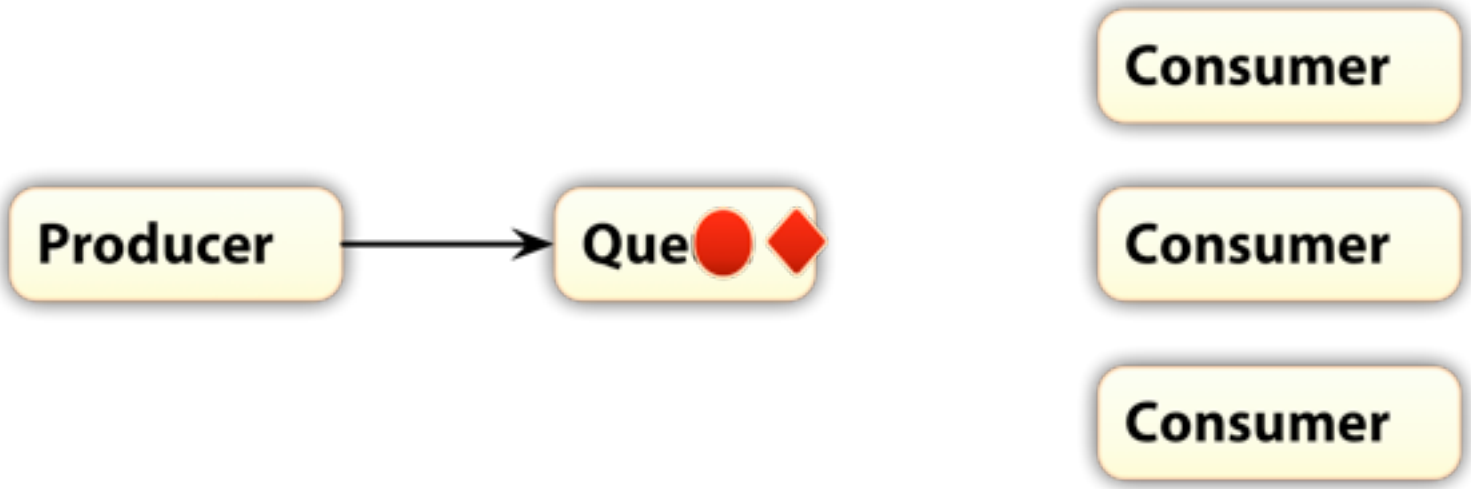
Messaging Server:

- Queues
- Topics
- Transactions
- Reliable Messaging

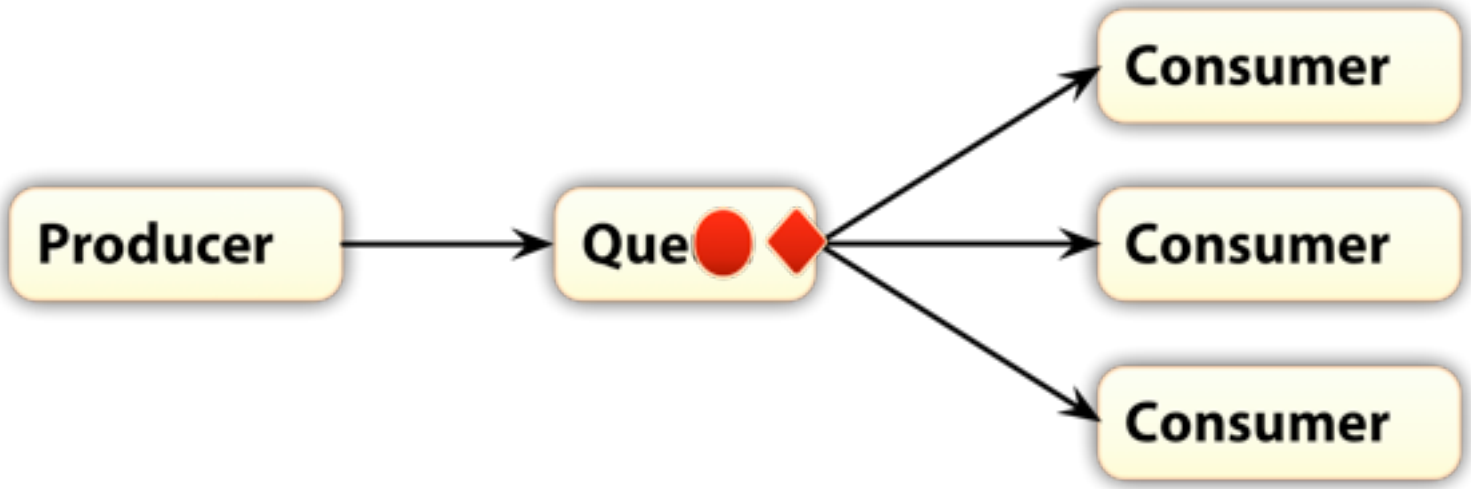


Queues: Point-to-Point

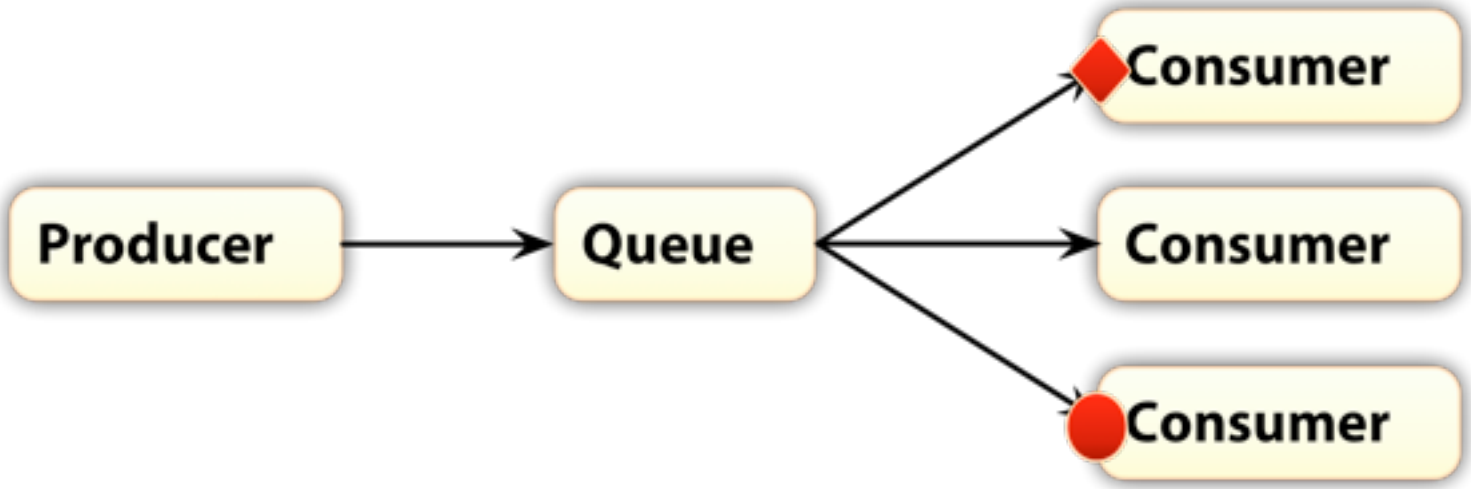
Queues: Point-to-Point



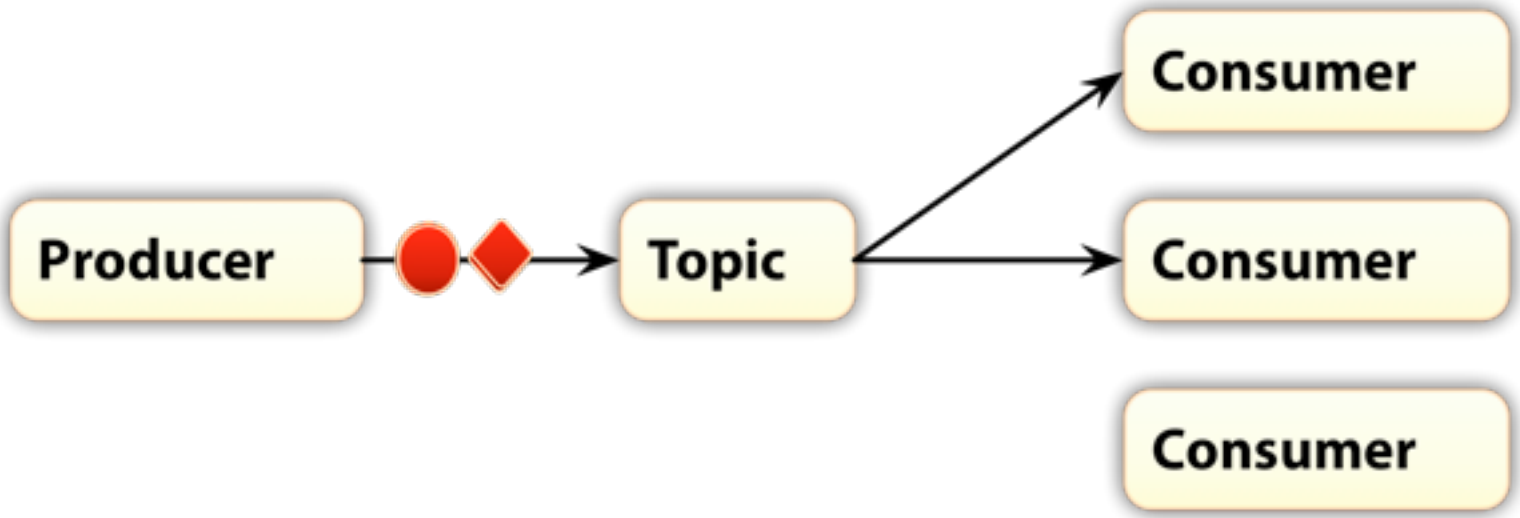
Queues: Point-to-Point



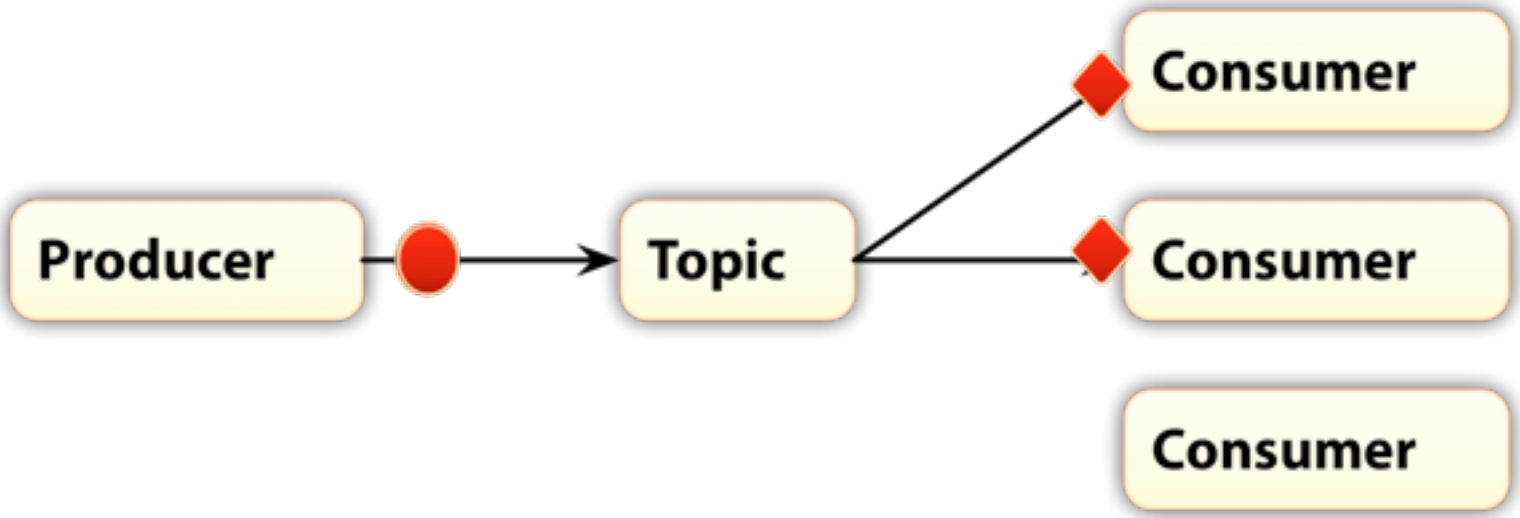
Queues: Point-to-Point



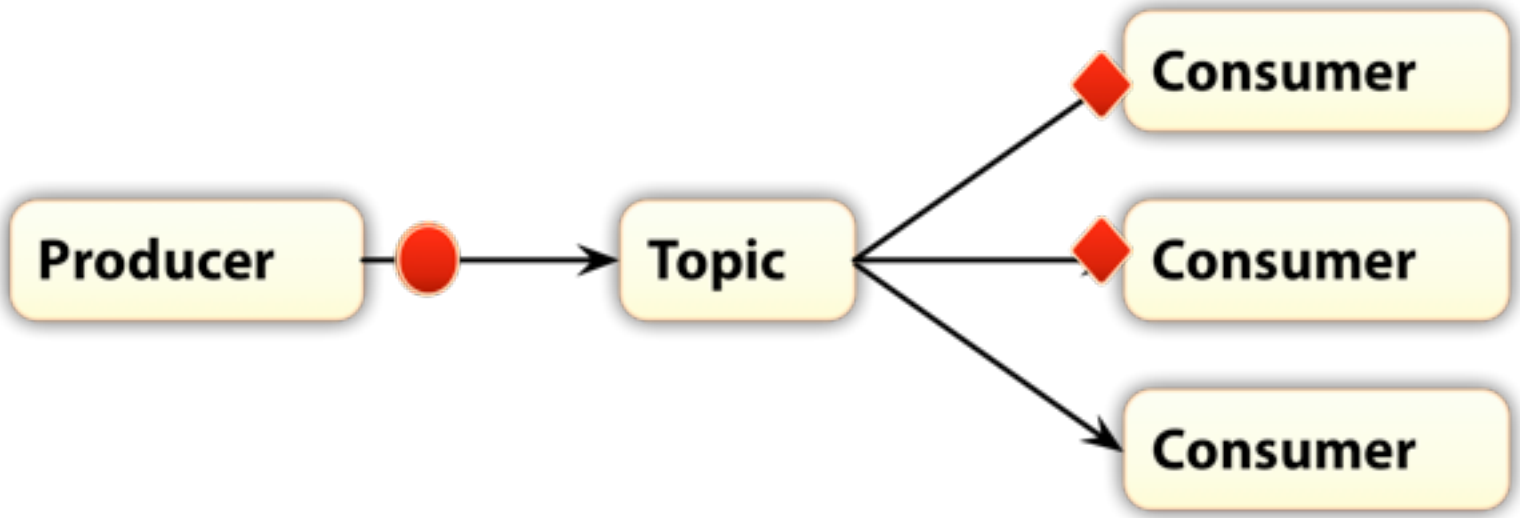
Topics: Publish/Subscribe



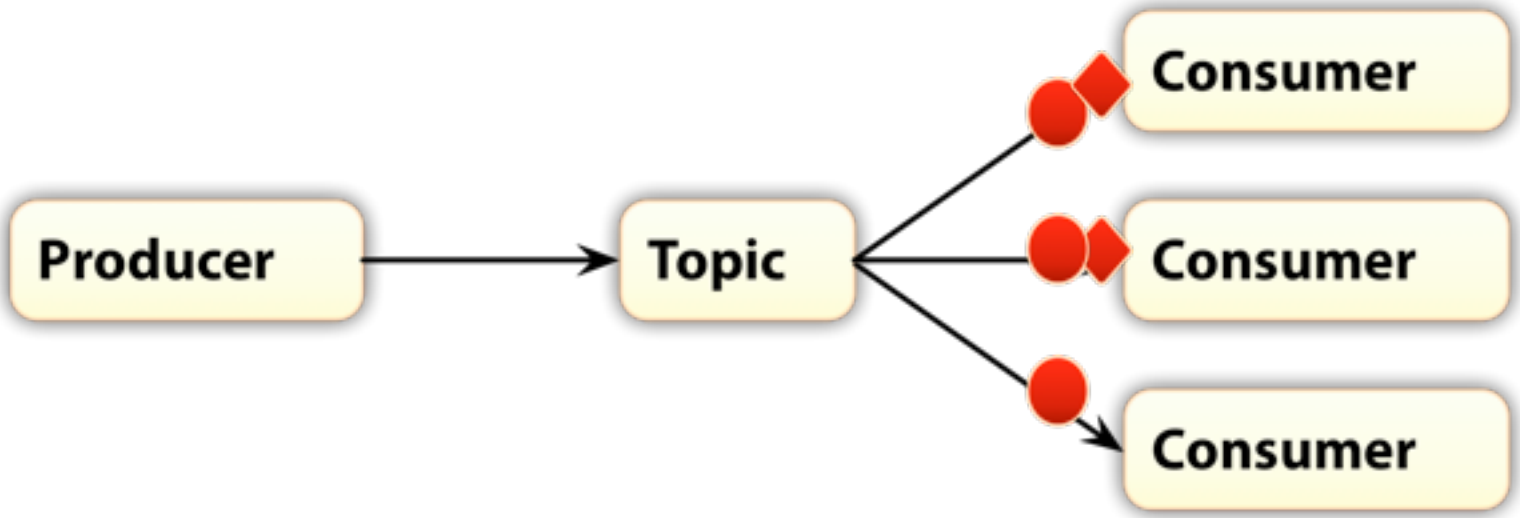
Topics: Publish/Subscribe



Topics: Publish/Subscribe



Topics: Publish/Subscribe



But your happy /w ActiveMQ?
Yay! Stick with it!

ActiveMQ will be supported for many more years to come!

It will be a long time before Apollo:

- Supplies all of ActiveMQ's features
- Provides migration tools

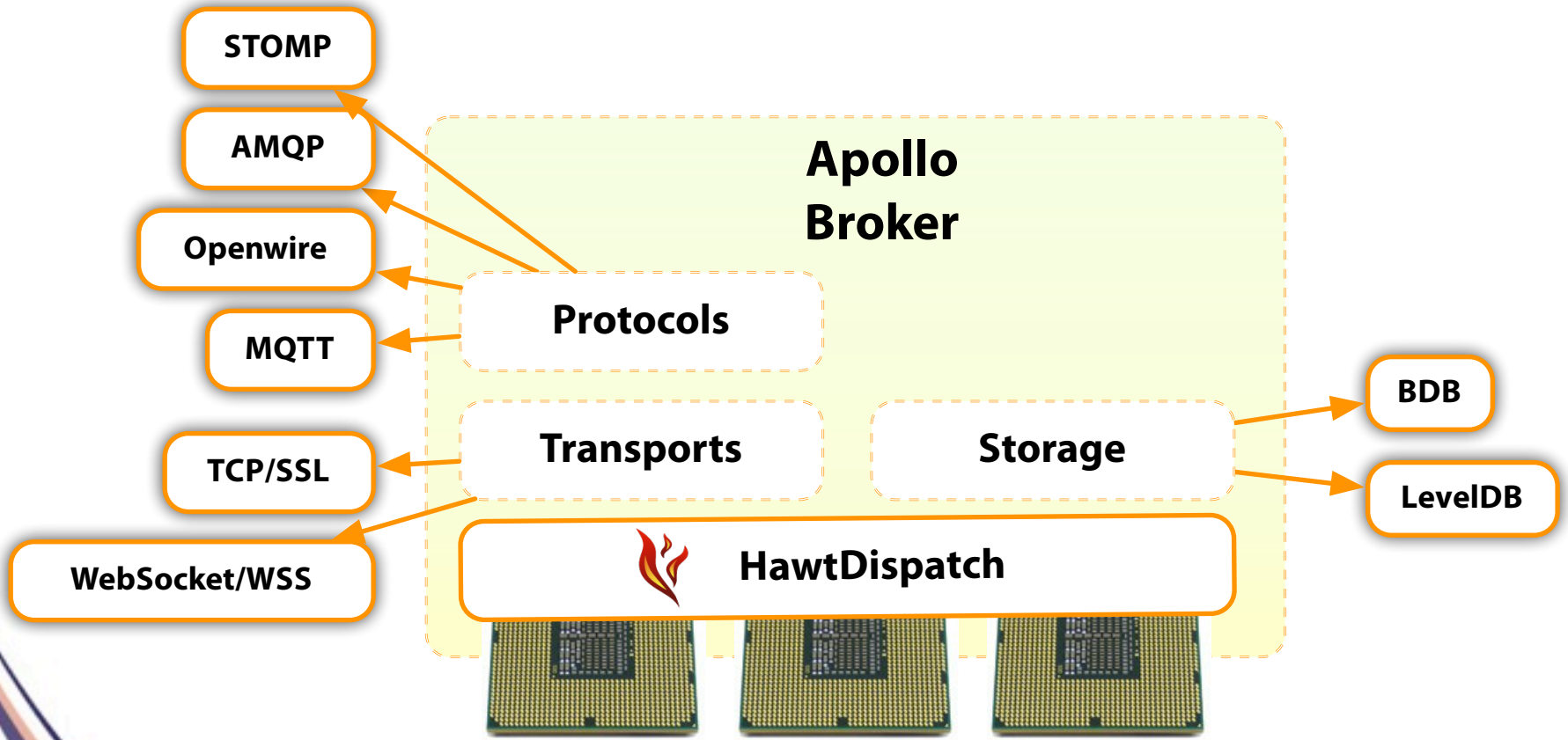
Apollo bits are being back ported

Why use Apollo?

Do you want:

Lower CPU overhead
A reduced memory footprint
Runtime configuration reloading
REST based management API
Simpler Configuration Options

Plugin Architecture



Transports

- TCP
- SSL
- WebSockets
- Secure WebSockets
- UDP

Protocols

- STOMP 1.0/1.1/1.2
- MQTT v3.1
- AMQP 1.0
- Openwire

Protocol: STOMP

http://stomp.github.com/

Simple **T**ext **O**rientated **M**essaging **P**rotocol

Uses Text Headers like HTTP

Many Clients APIs in Java, C#, C, Ruby, Python, JS, PHP, etc.

Interoperates with ActiveMQ, RabbitMQ, HornetQ, …

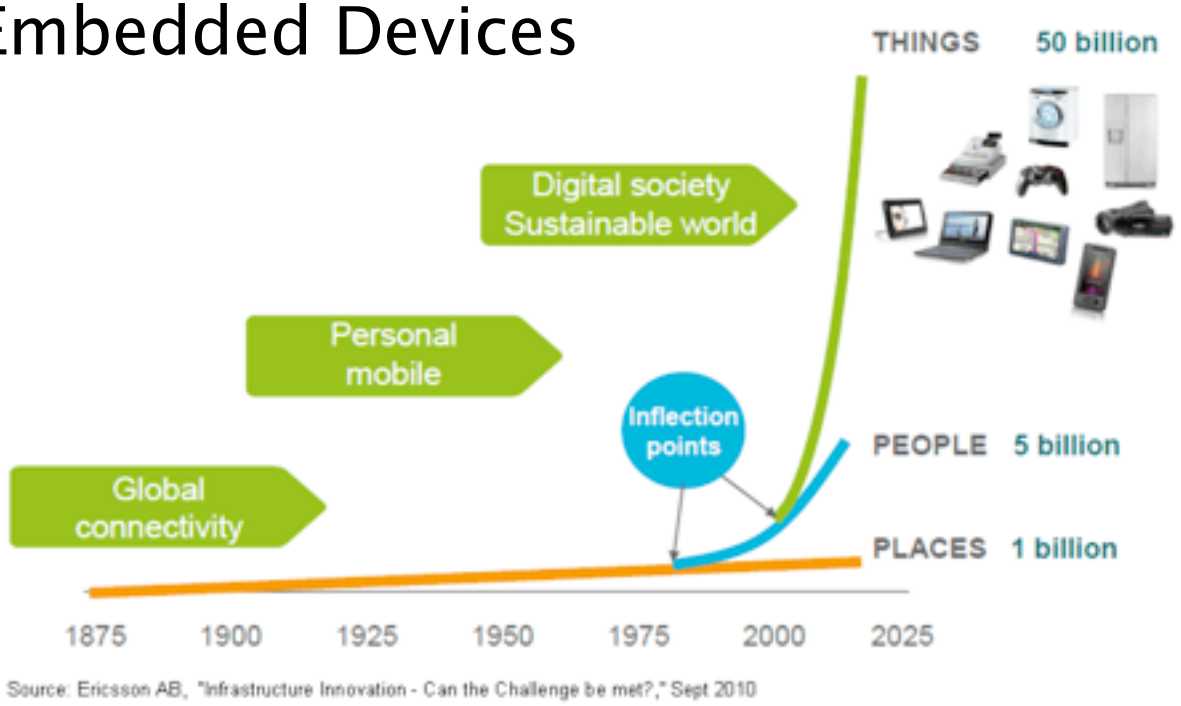
Protocol: MQTT

Get at https://github.com/fusesource/fuse-extra/

Focused on:

- Pub/Sub
- Unreliable, low bandwidth networks
- Small footprint / Embedded Devices

Interoperates with WebsphereMQ, Mosquitto, …



Protocol: AMQP 1.0

AMQP is a **binary** wire protocol which was designed for interoperability between different vendors.

Supports Queue and Topic semantics.

Many native client libraries are available.

Interoperates with ActiveMQ, Qpid, SwitfMQ...

Protocol: Openwire

- Openwire is the native binary protocol implemented by ActiveMQ
- API options:
  - JMS 1.1 Client of ActiveMQ 5.x
  - NMS Client for C# Apps
  - CMS Client for C++ Apps
- Not Yet Supported
  - XA Transactions (distributed transactions)

Message Stores

Are Plugins

Ships with 2 Options

- LevelDB Store
- BDB Store

Used to store

- persistent messages
- non-persistent messages that needs to be swapped out of memory

Also used to swap out non-persistent messages.

LevelDB Store

- A Journal + LevelDB based index
- The pure ASL 2.0 licensed option
- Uses a JNI implementation on Linux and OS X
  - Fastest Store available
- On all other platforms a pure Java implementation is used
  - Not used as much as the JNI version
- LevelDB indexes are awesome for sequential r/w access patterns

BDB Store

Not ASL 2.0! You have to Agree to the BDB license & download from Oracle.

Pure Java implementation

Very robust.

What makes Apollo **Different**?

HawtDispatch

- Event Processing System
- Multithreaded Reactor Model
- Fixed size Thread Pool /w NIO event support

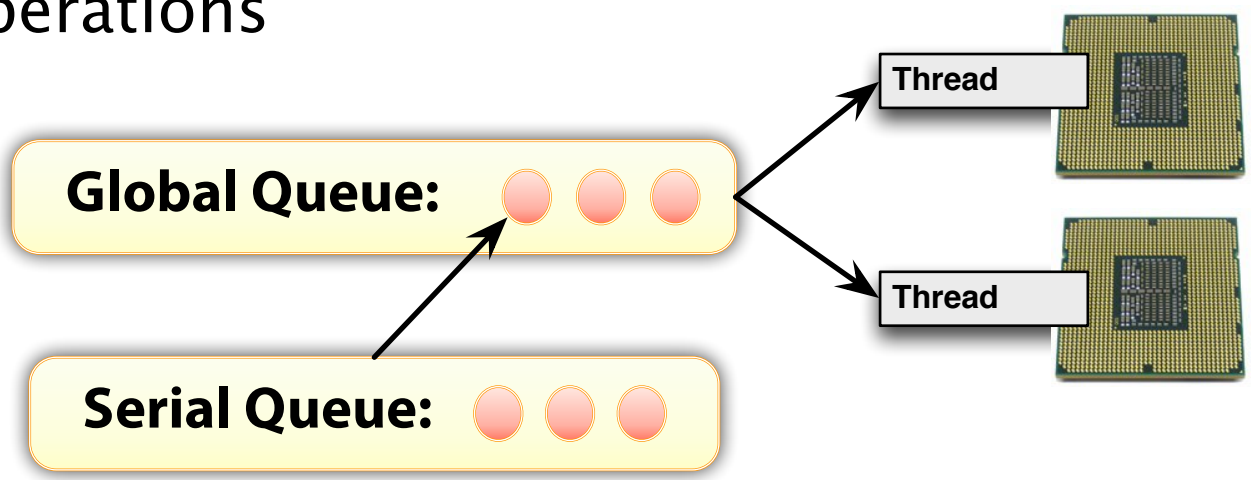
Dispatch Queues

Global Dispatch Queue

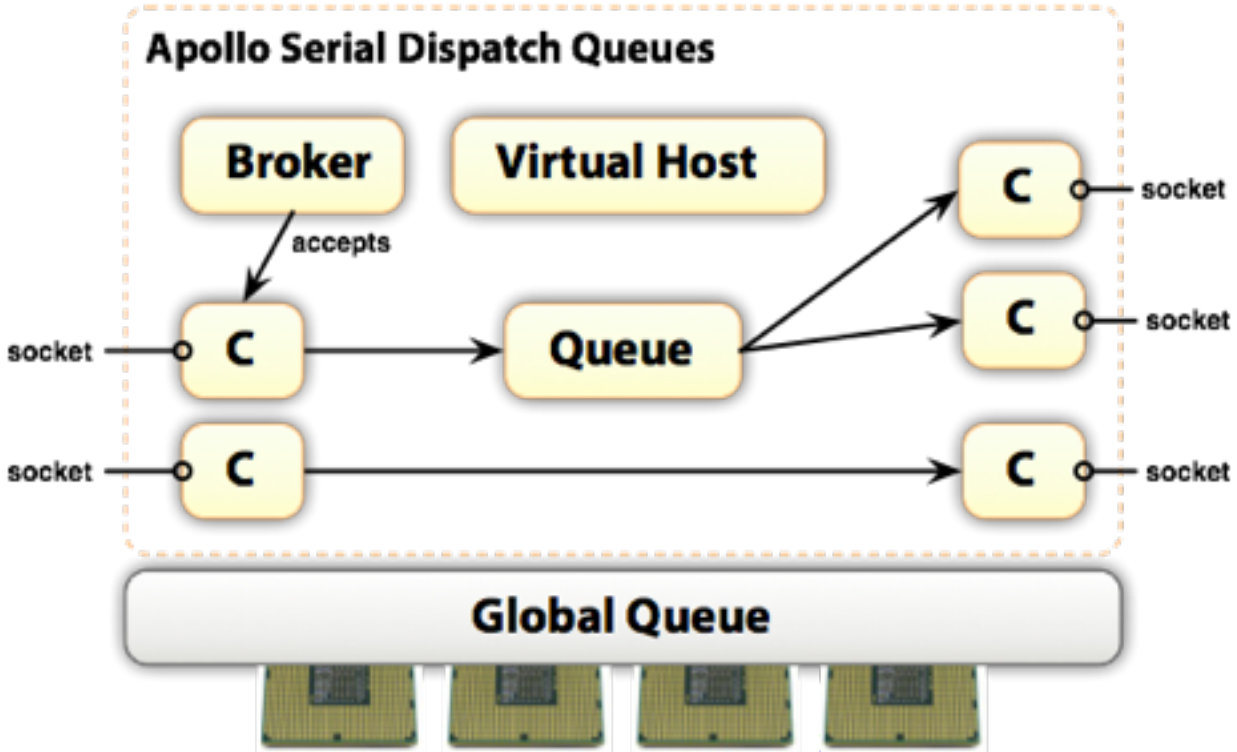
- Concurrently executes Runnable objects
- Only 1 in the system

Serial Dispatch Queue

- Serially executes Runnable objects
- Use CAS operations

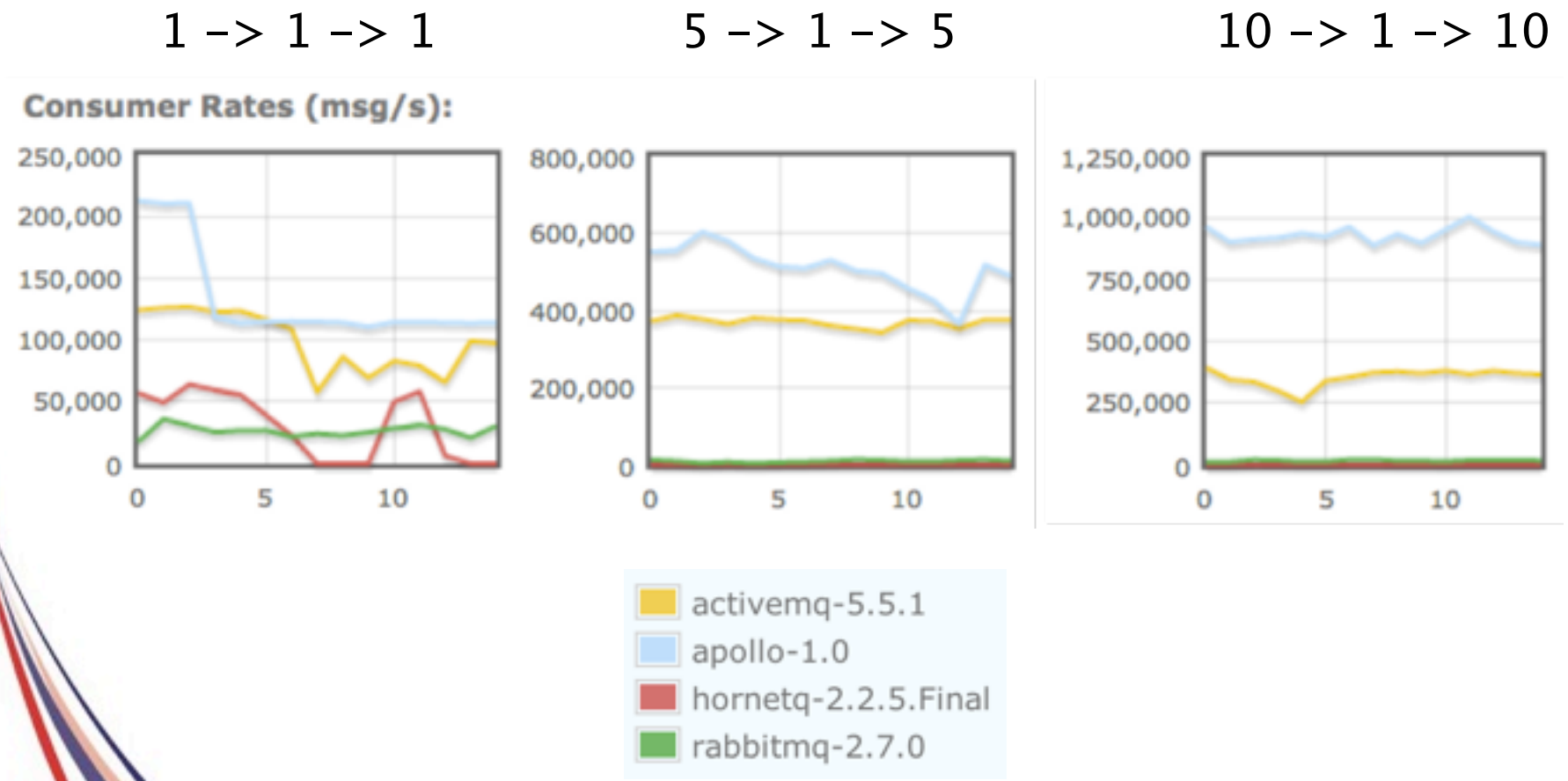


Serial Dispatch Queues



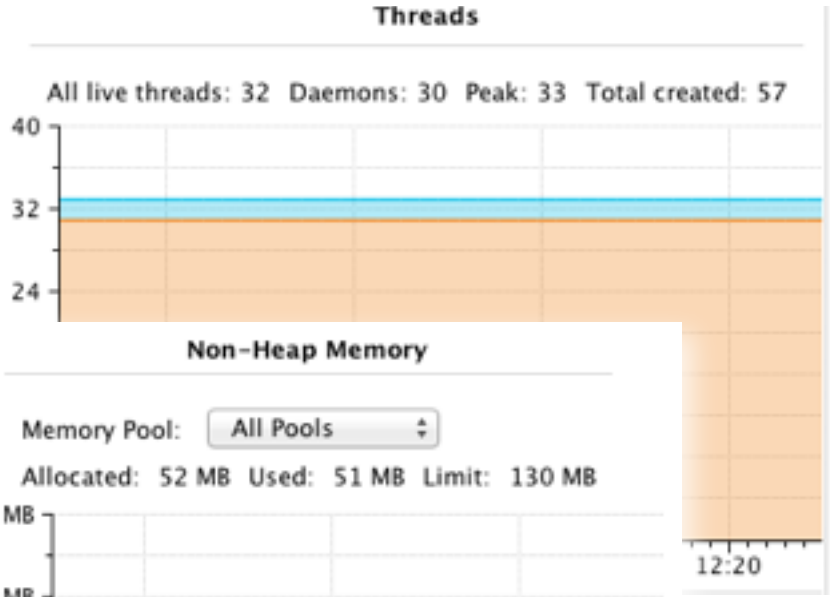
C = Client Connection

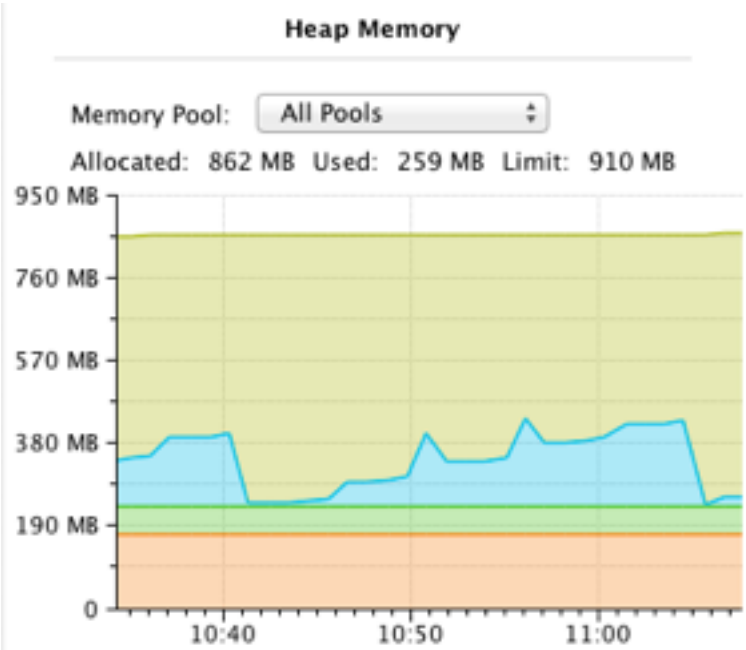
Low Thread Contention...

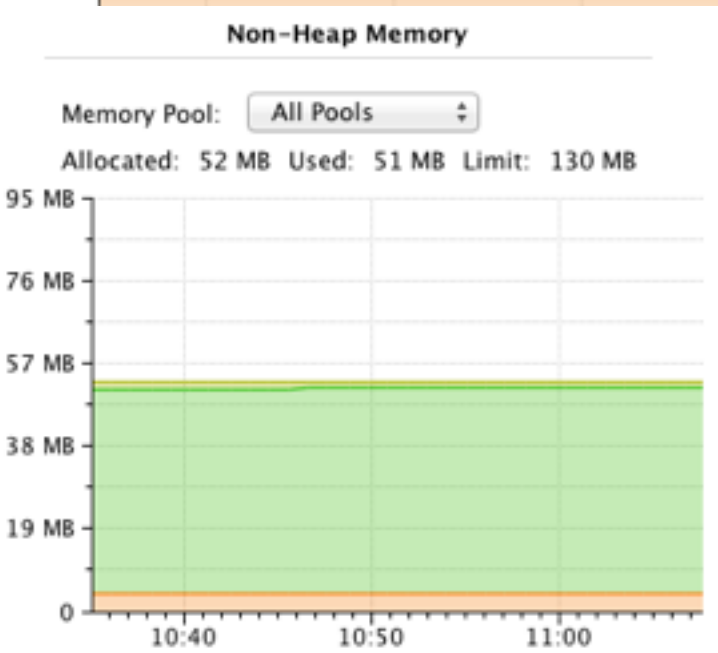


Low Memory Overhead...

- 1000 Producer Connections
- 1000 Topics
- 5000 Consumer Connections







Why is Apollo using Scala?

Java API example:

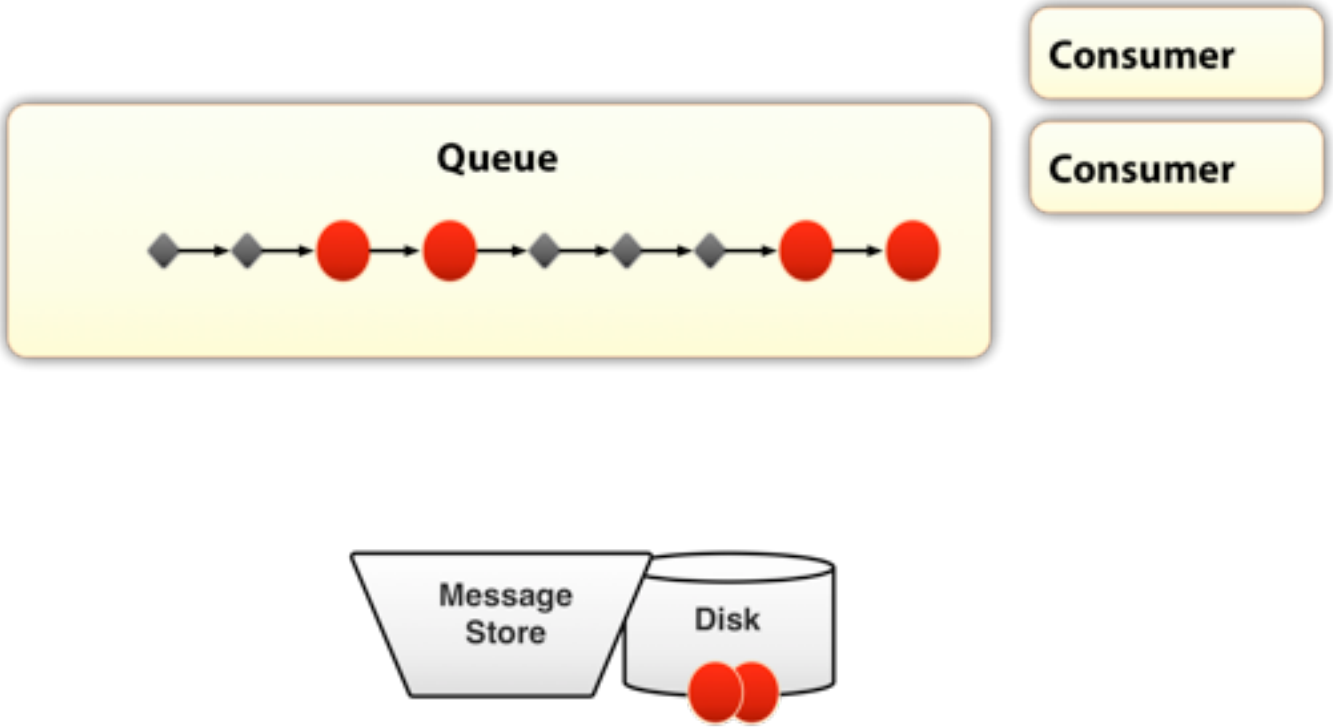
```
queue.execute(new Runnable(){
  public void run() {
    System.out.println("Hi!");
  }
});
```

Same thing in the Scala API:

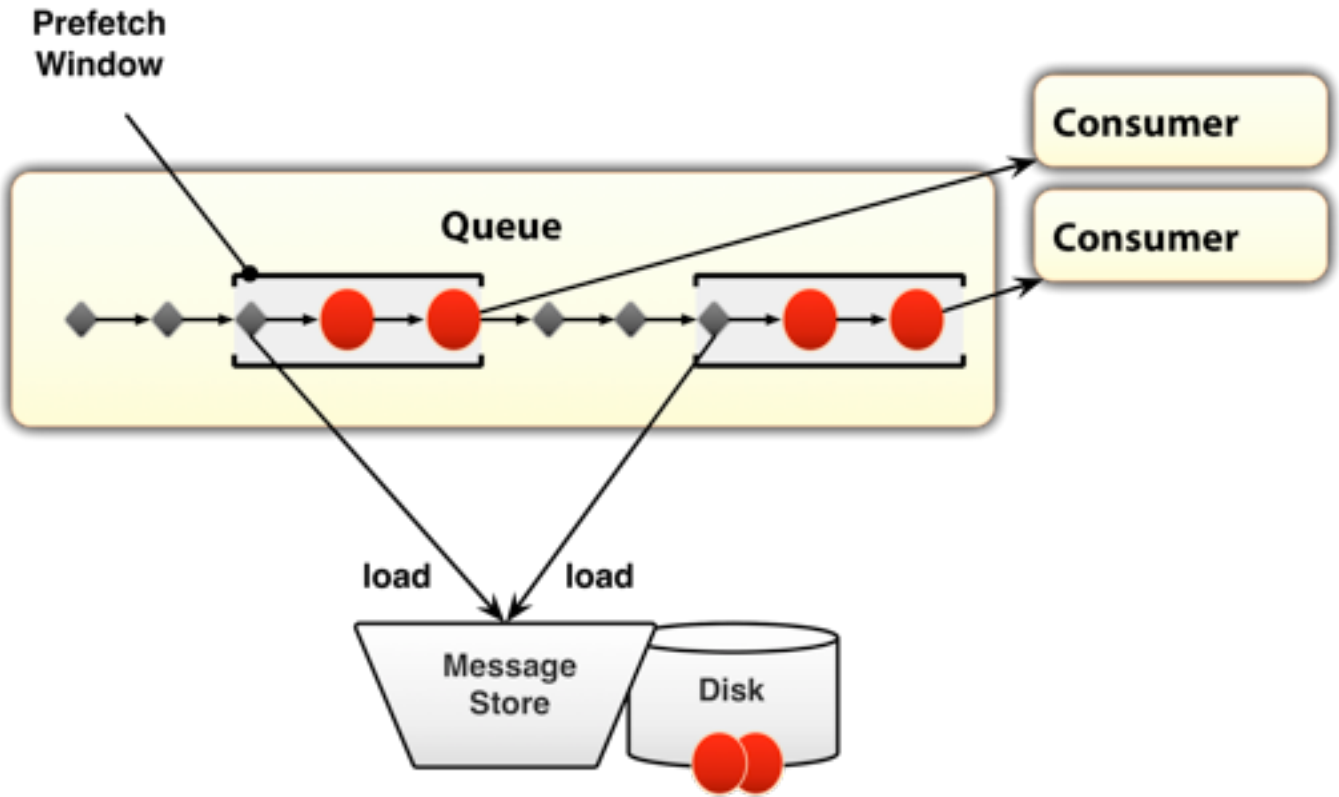
```
queue {
  System.out.println("Hi!");
}
```

Terse closures FTW!

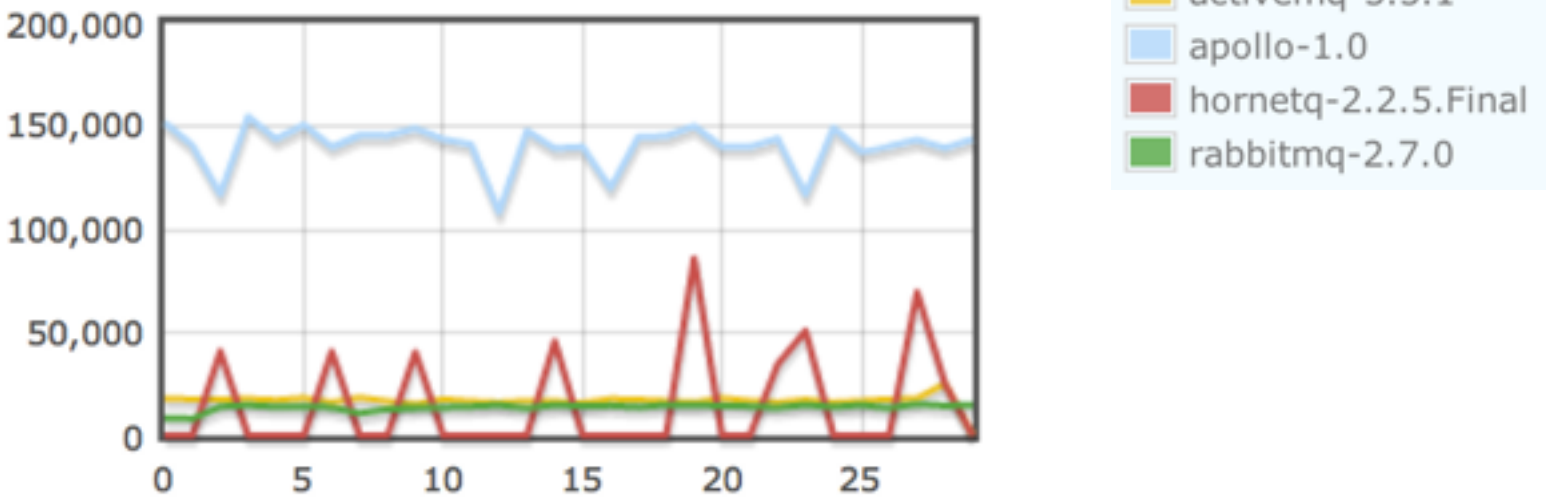
Per Consumer Store Prefetch



Per Consumer Store Prefetch

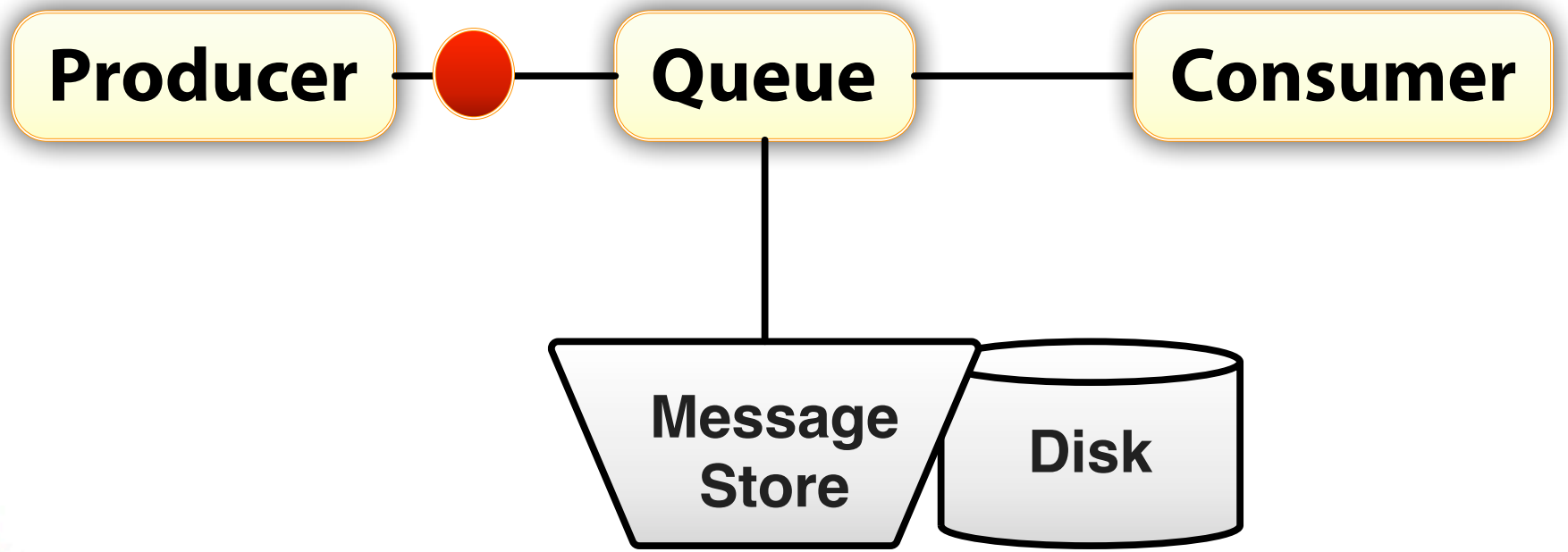


Per Consumer Store Prefetch

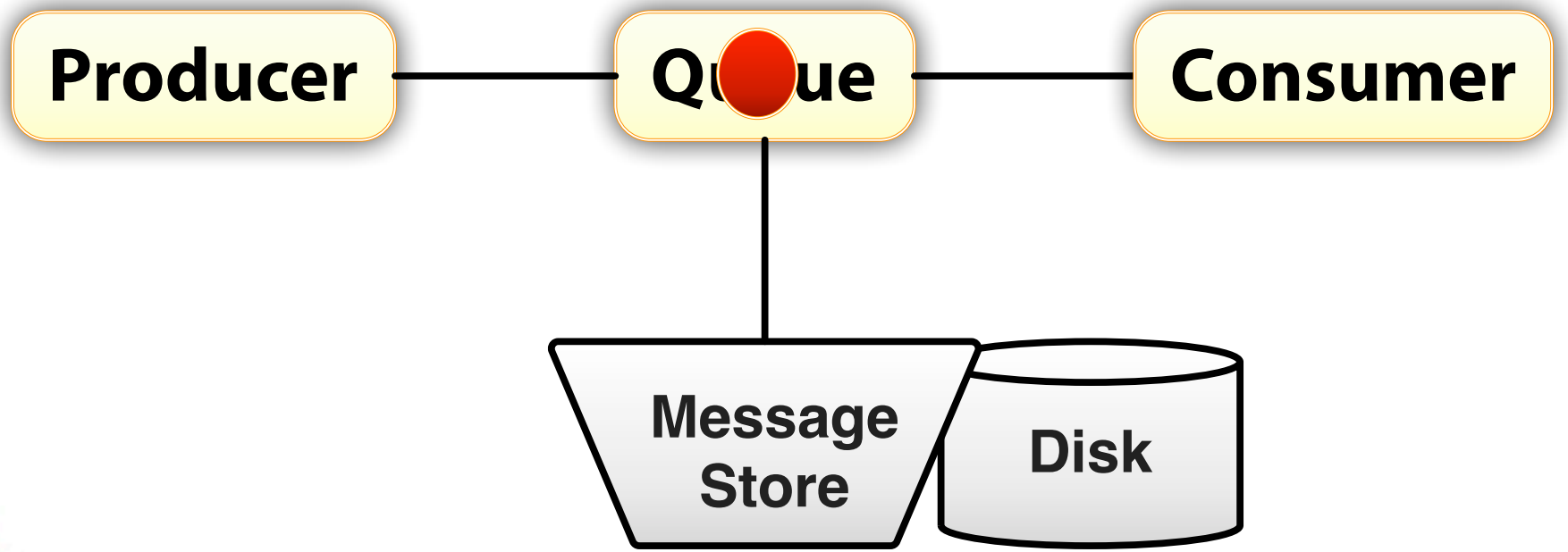


Source:
http://hiramchirino.com/stomp-benchmark/ubuntu-2600k/index.html

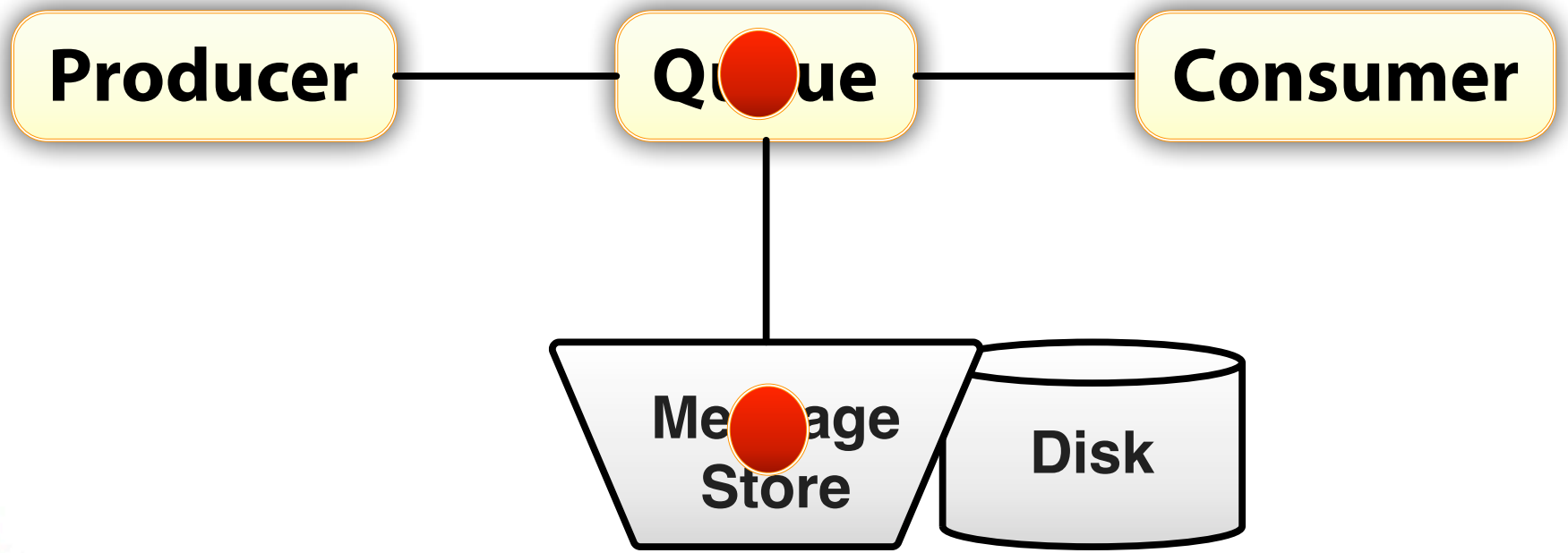
Message Store: Store and Dispatch



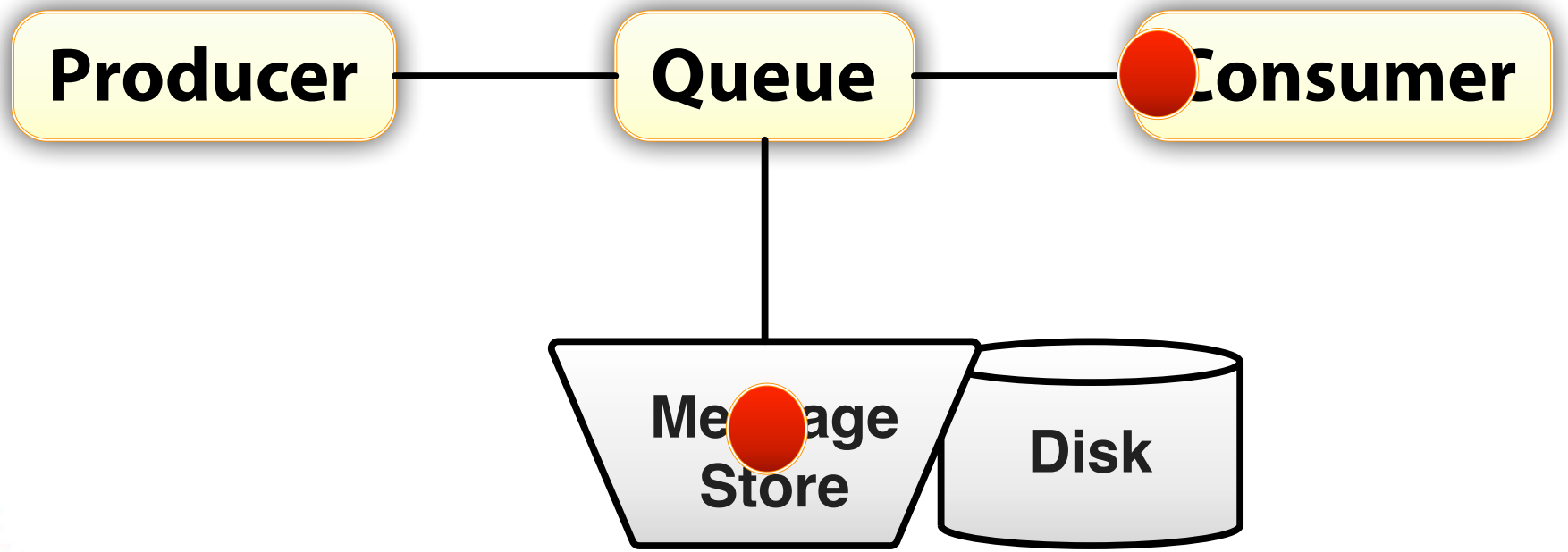
Message Store: Store and Dispatch



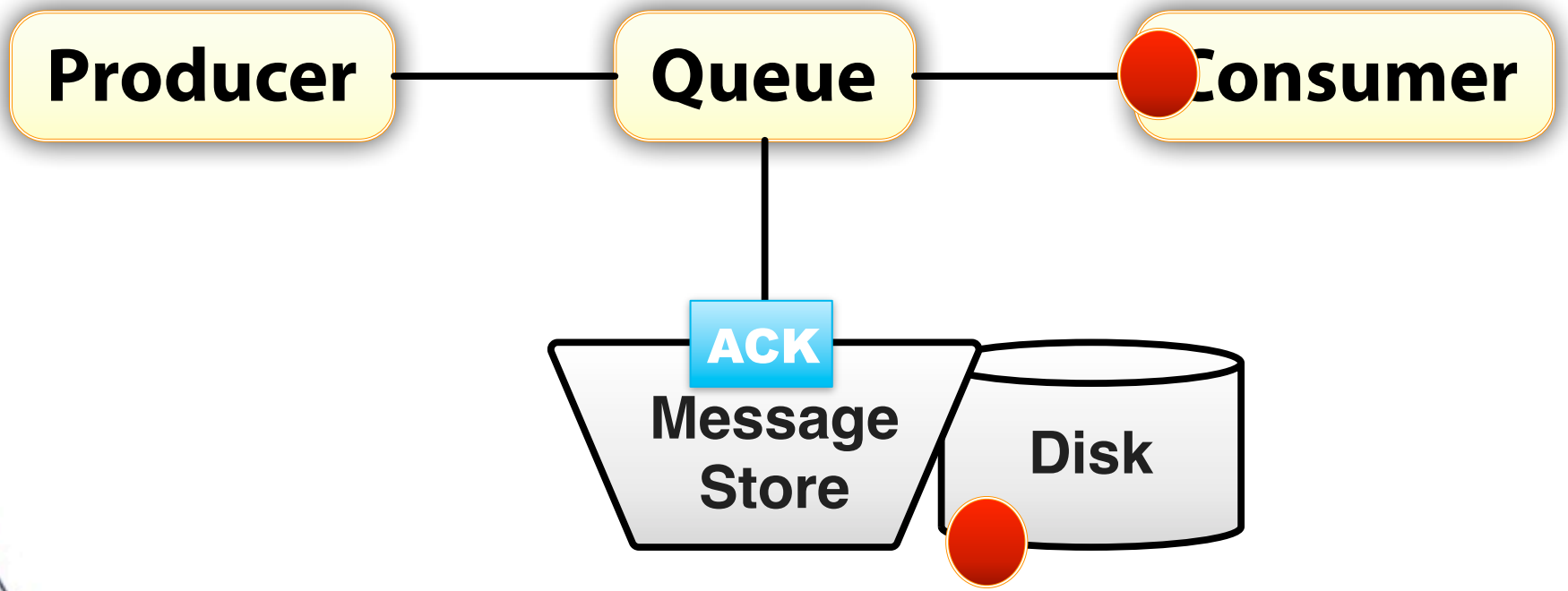
Message Store: Store and Dispatch



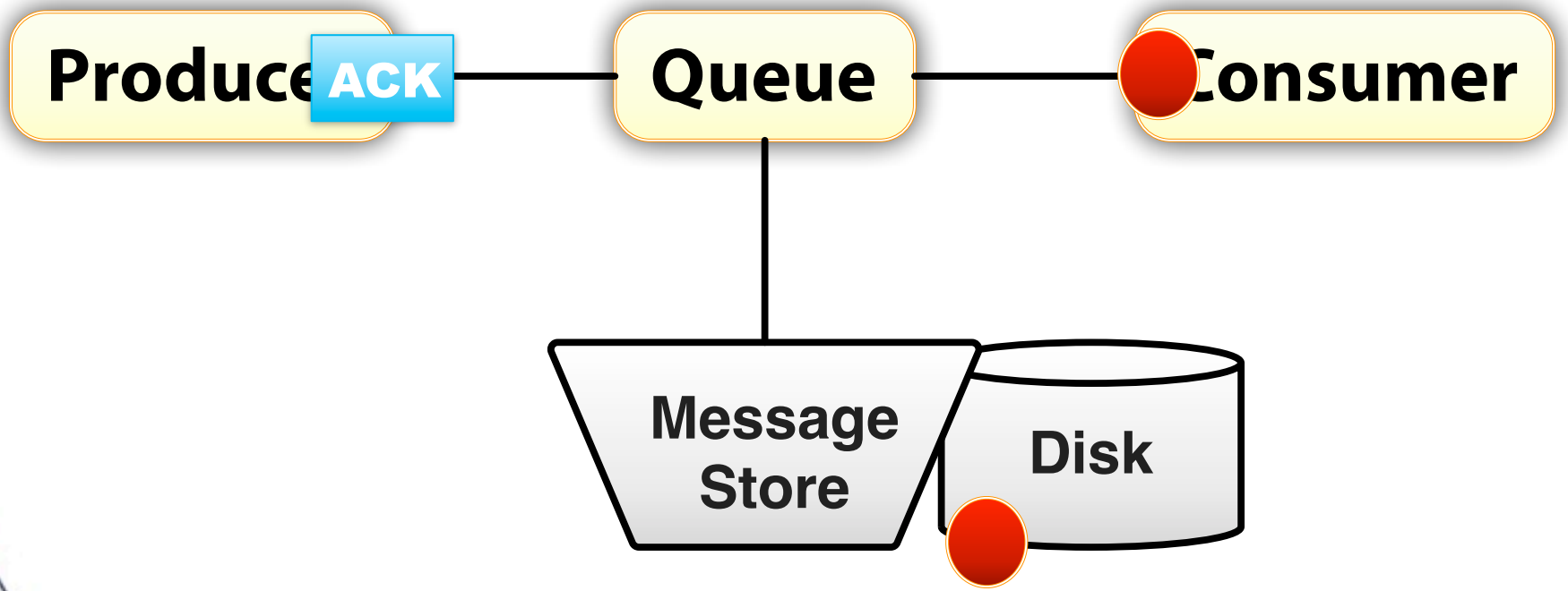
Message Store: Store with No Delay



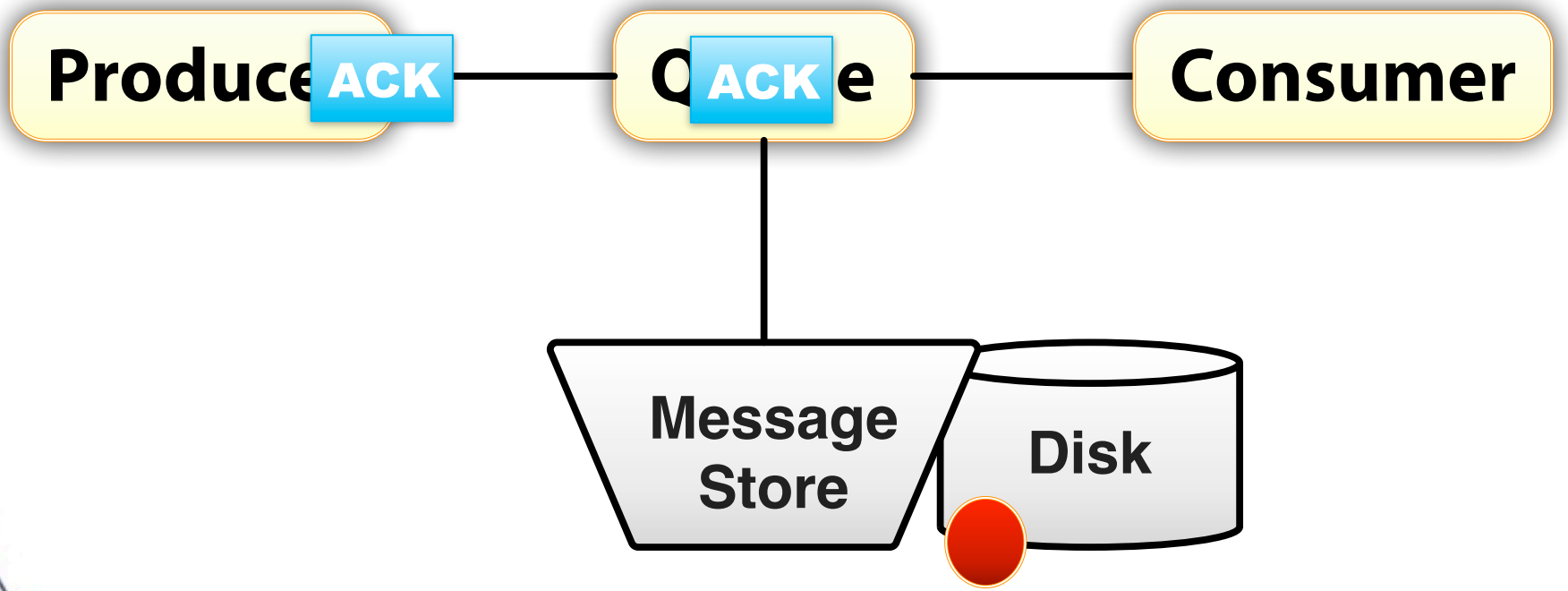
Message Store: Store with No Delay



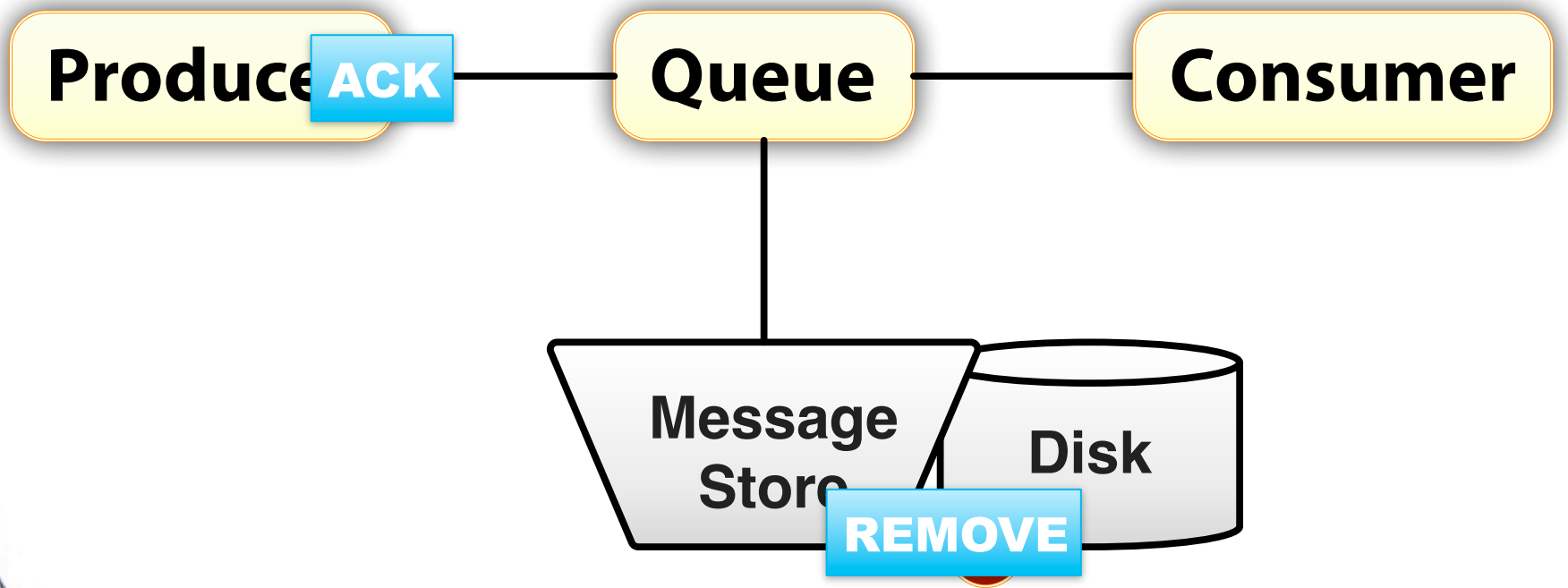
Message Store: Store with No Delay



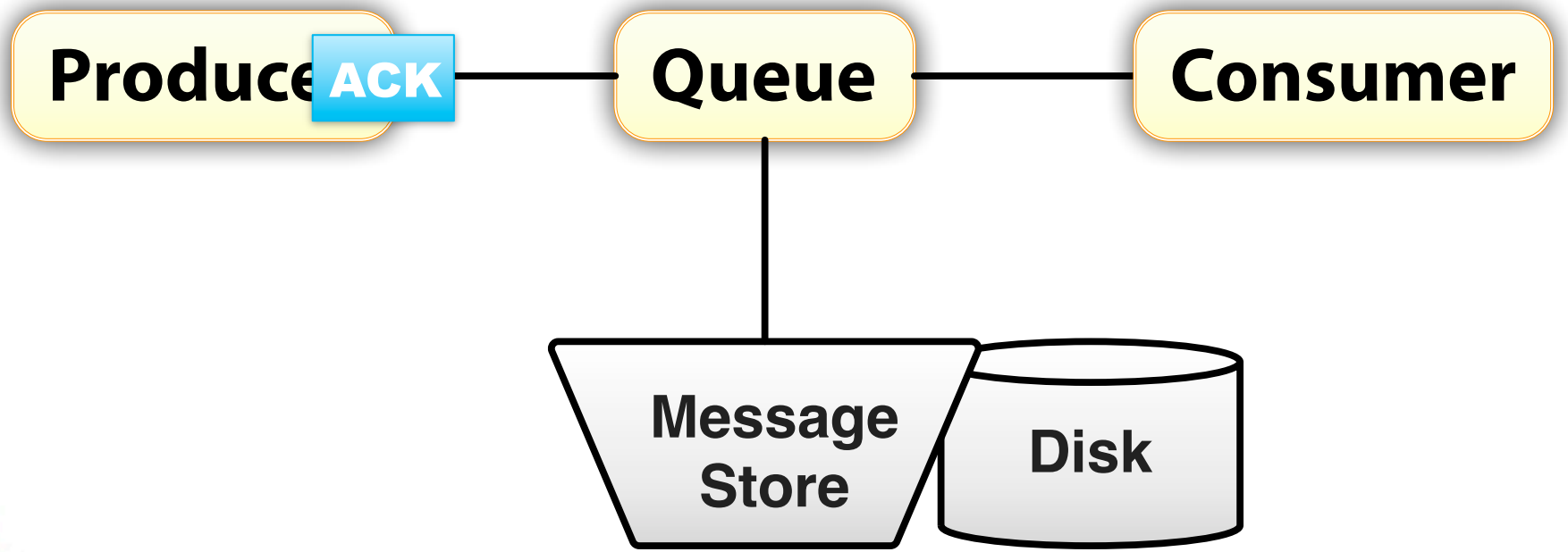
Message Store: Store with No Delay



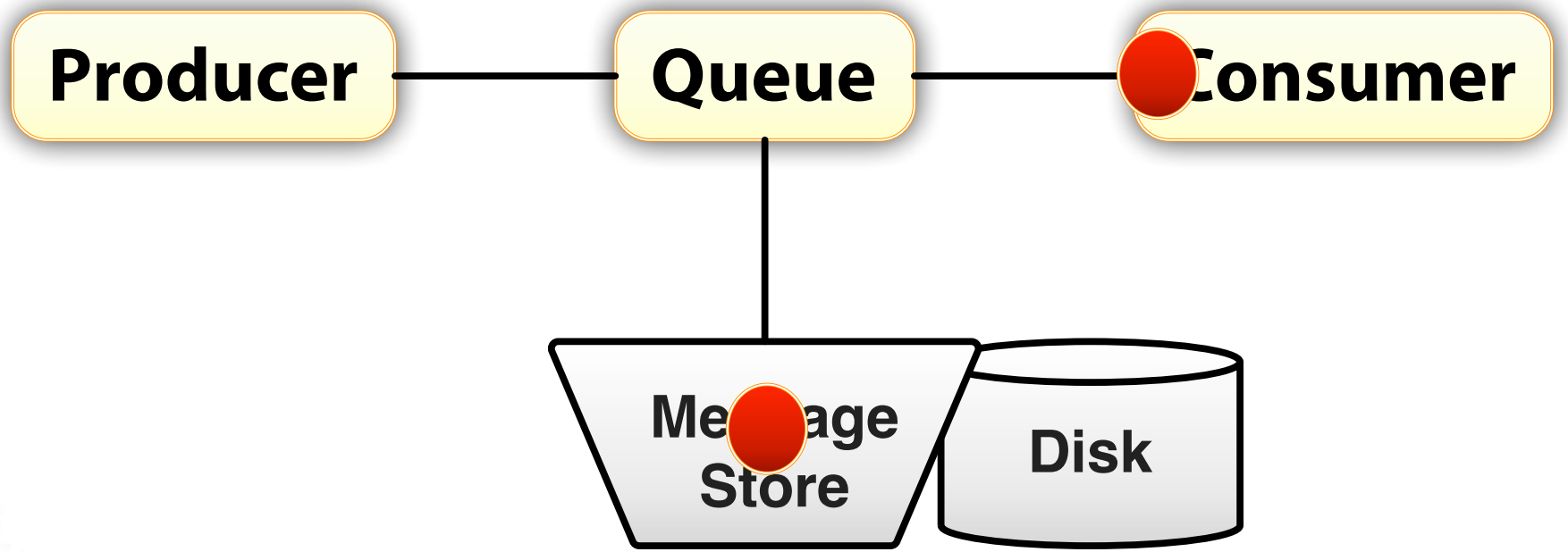
Message Store: Store with No Delay



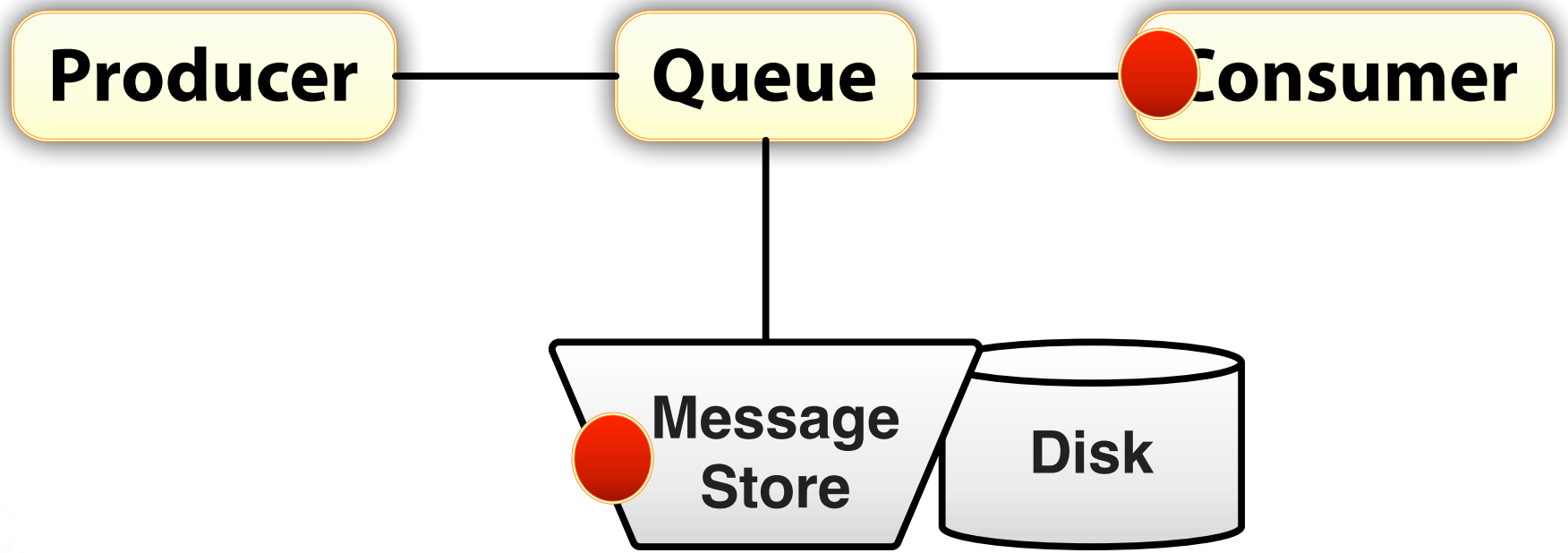
Message Store: Store with No Delay



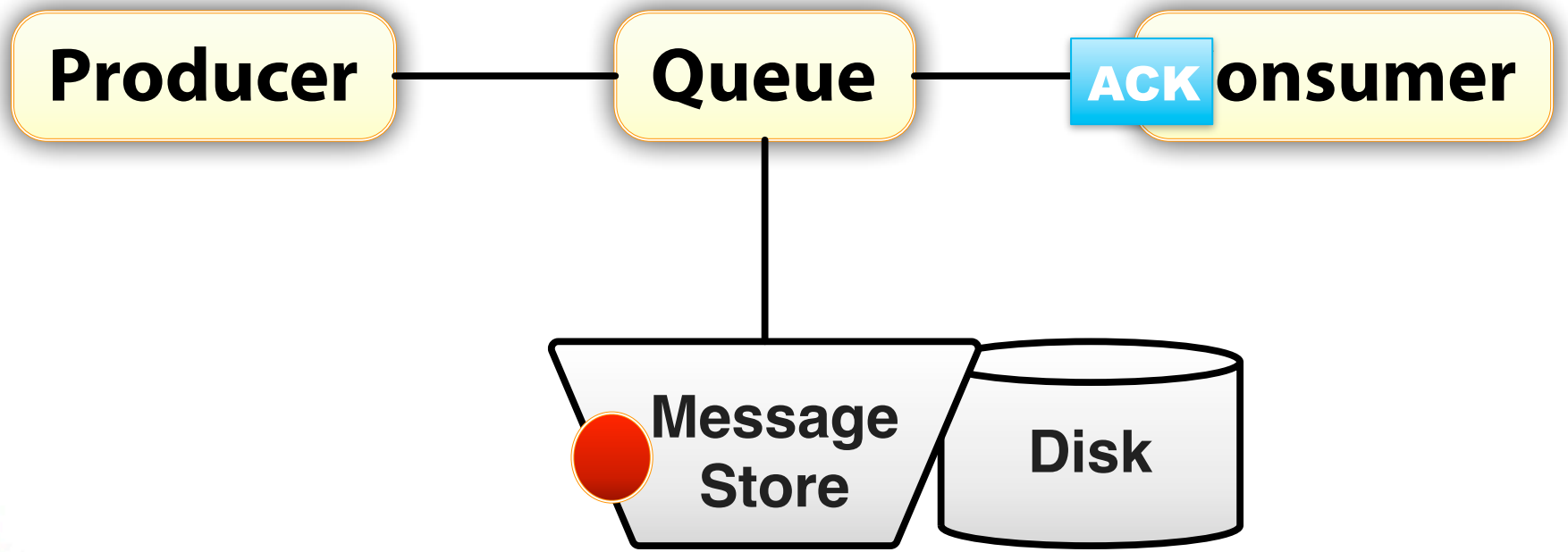
Message Store: Store with Delay



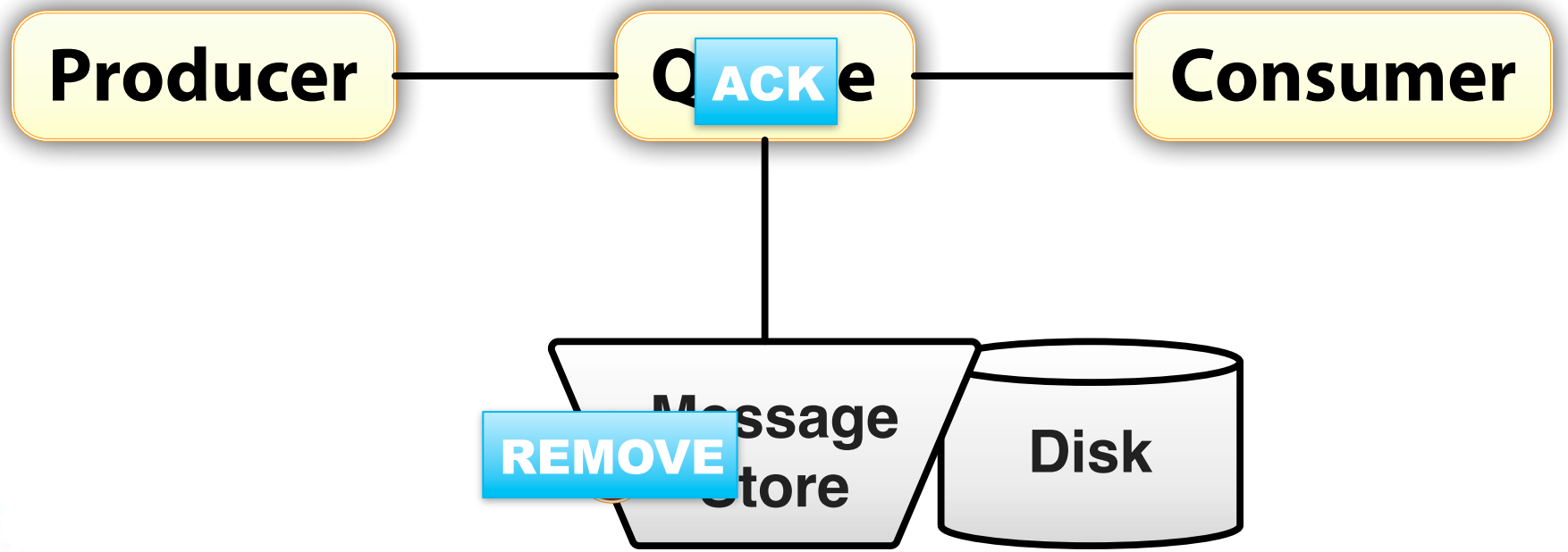
Message Store: Store with Delay



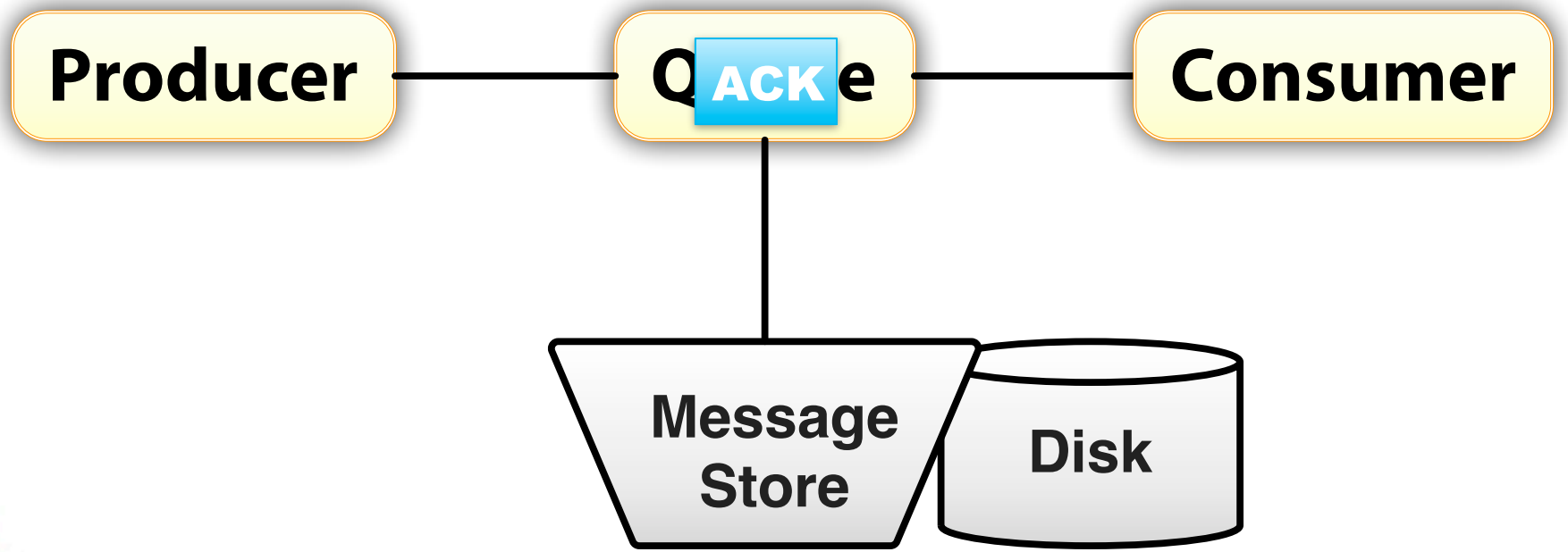
Message Store: Store with Delay



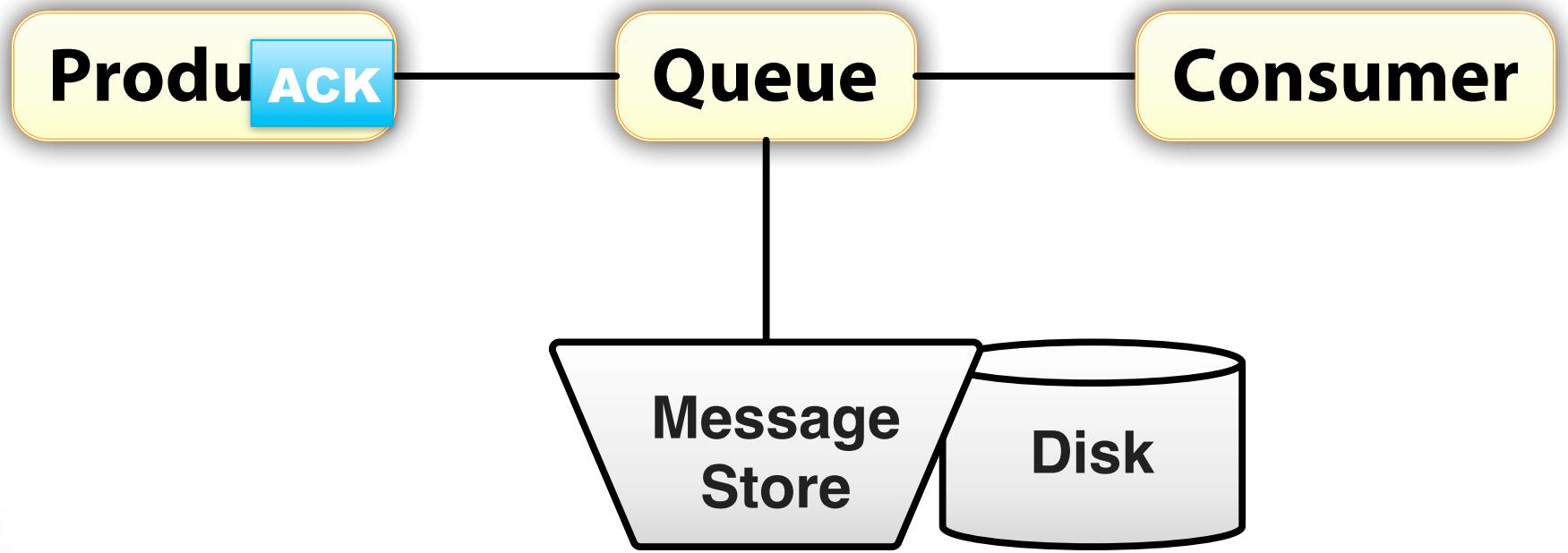
Message Store: Store with Delay



Message Store: Store with Delay



Message Store: Store with Delay



Forget Virtual Destinations!

- Durable subscriptions are implemented with queues.

- Used a mirrored queue to send a copy to the topic with the same name.

Apollo's Trajectory

FuseSource
A Progress Software Company

Features! Features! Features!

Road Map Features

- Networks of Brokers
- Priority Support
- Message Scheduling
- XA Transactions
- JMX Management API

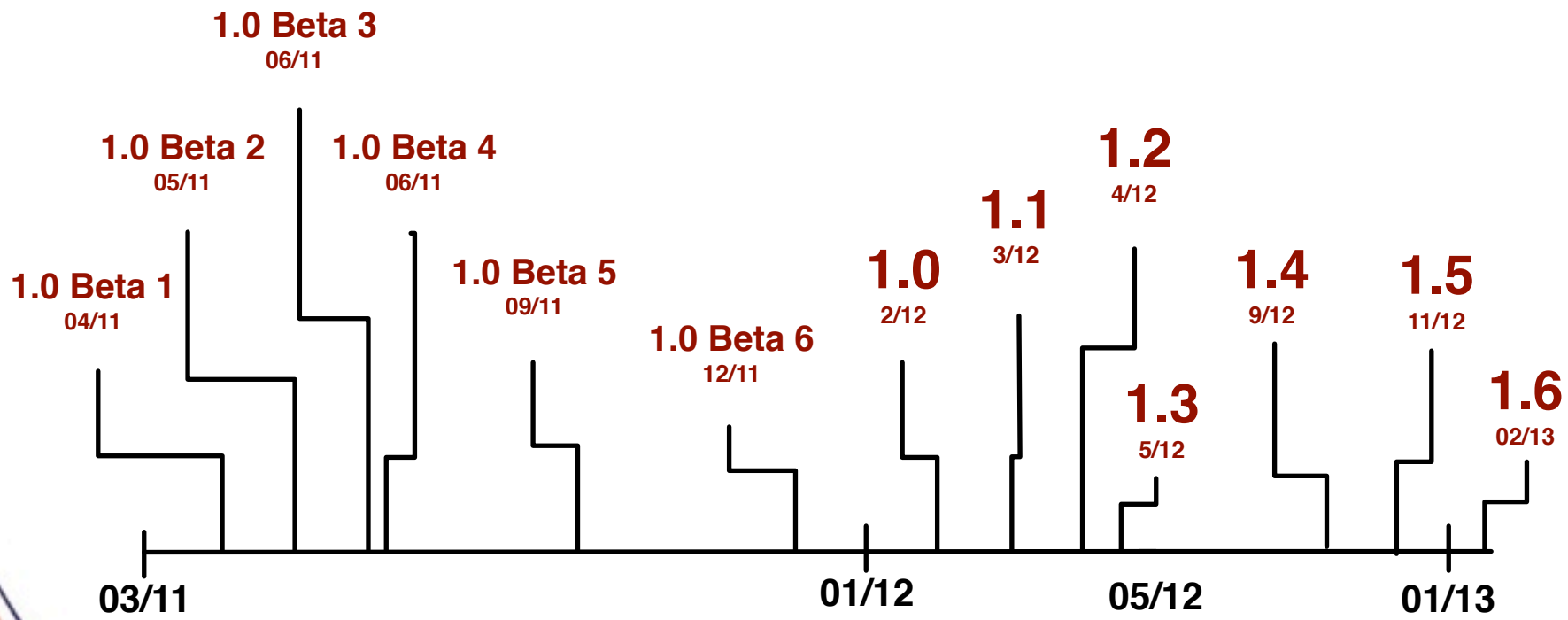
Back Ported Apollo Features

- LevelDB Store
- MQTT Protocol
- STOMP 1.1 Support

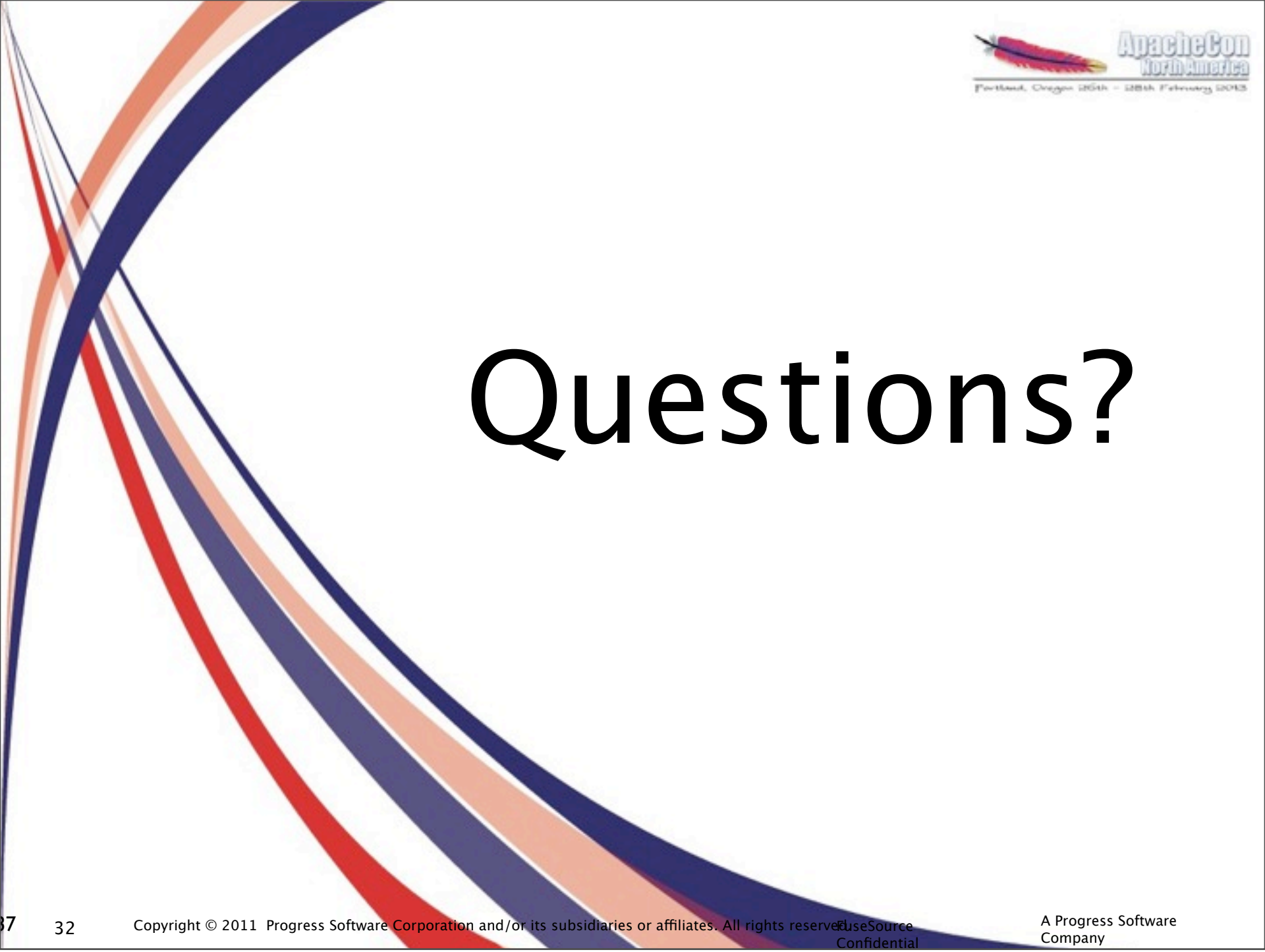
Pending Back Port

- Store Delays

Release Velocity




DEMO



The Link Bonanza

Apache Apollo
http://activemq.apache.org/apollo/

STOMP Benchmarks
http://hiramchirino.com/stomp-benchmark/

MQTT Protocol Plugin for Apollo
https://github.com/fusesource/fuse-extra

HawtDispatch
http://hawtdispatch.fusesource.org/

StompJMS
https://github.com/fusesource/stompjms