redhat.

Asynchronous Messaging with Apache ActiveMQ


ActiveMQ

What is Apache ActiveMQ?

- Top level Apache Software Foundation project
- Wildly popular, high performance, reliable message broker
  - Clustering and Fault Tolerance
  - Supports publish/subscribe, point to point, message groups, out of band messaging and streaming, distributed transactions, …
- Myriad of connectivity options
  - Native Java, C/C++, and .NET
  - AMQP 1.0, MQTT 3.1.1, STOMP 2.0, and OpenWire
  - STOMP protocol enables Ruby, JS, Perl, Python, PHP, ActionScript, ...
- Embedded and standalone deployment options
  - Pre-integrated with open source integration and application frameworks
  - Deep integration with Spring Framework, OSGi, and Java EE

Configuring Transport Connectors

Configured in broker for client connections

TCP - most used; socket connections using binary OpenWire protocol

NIO - like TCP, except uses Java NIO to reduce number of threads managing all connections

SSL - secure TCP connection

STOMP - text based protocol; facilitates multiple language integration

MQTT - lightweight publish / subscribe protocol

VM - enables efficient in-process connections for embedded broker

Examples:

```
<transportConnector uri="tcp://0.0.0.0:61616"/>

<transportConnector uri="nio://0.0.0.0:61616"/>

<transportConnector uri="stomp://0.0.0.0:61617"/>

<transportConnector uri="stomp+nio://0.0.0.0:61617"/>
```

Configuring Client Connections

Matches configuration for transport connectors in the broker

Set as broker url on JMS connection factory

Options can be set in the url as key/value params or directly on the connection factory

Format:

tcp://hostname:port?key=value

Examples:

tcp://myhost:61616?trace=false&soTimeout=60000

Lot more details at:

http://activemq.apache.org/configuring-transports.html

Configuring Client Connections - Wrapper Transports

Augment / wrap client side connections

Failover - automatic reconnection from connection failures

Fanout - simultaneously replicate commands and message to multiple brokers

Discovery - automatic discovery of brokers

Format:

wrapper:(tcp://hostname:port?key=value)

Examples:

failover:(tcp://master:61616,tcp://slave:61616)

failover:(tcp://virtualIp:61616)

fanout:(static:(tcp://host1:61616,tcp://host2:61616))

Configuring Persistence Adapters

File system based

- kahaDB - recommended; improved scalability and quick recovery
- levelDB - high throughput, quick recovery, better indexing

RDBMS based

- jdbcPersistenceAdapter - quick and easy to setup
- journaledJDBC - faster than pure JDBC; file journaling with long term JDBC storage

Memory based

- memoryPersistenceAdapter – testing only; same as <broker persistent="false">

Configuring Network Connectors

Used to connect a broker to other brokers in the network

Matches configuration for transport connectors in the broker

Format:

tcp://hostname:port?key=value

Examples:

<networkconnector uri="static:(tcp://myhost:61616)"/>

<networkconnector

uri="failover:(tcp://master:61616?soTimeout=60000,tcp://slave:61616)"/>

Lot more details at:

http://activemq.apache.org/networks-of-brokers.html

High Availability

Two complementary approaches:

Master/Slave - access to persistent messages after broker failure

A given message is in one and only one broker (persistence store)

If a broker instance fails, all persistent messages are recoverable upon broker restart

Master/Slave allows a 2nd broker instance (slave) to be ready to process persistent messages upon master (1st broker) failure

Clients should use failover transport to automatically connect to slave

```
failover:(tcp://master:61616,tcp://slave:61616)?randomize=false
```

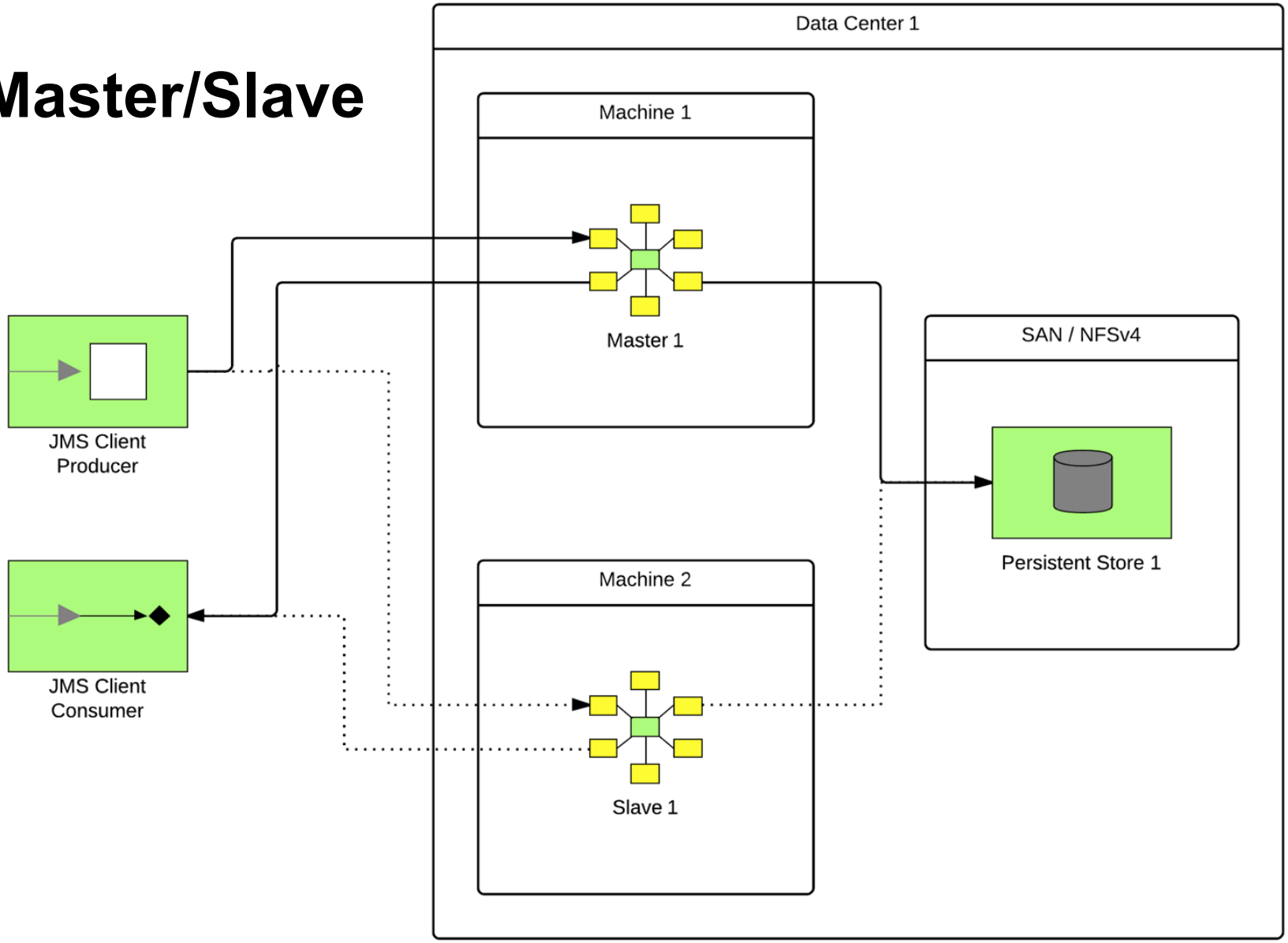
Network of Brokers - scale out message processes

Messages can be load balanced to consumers across multiple brokers

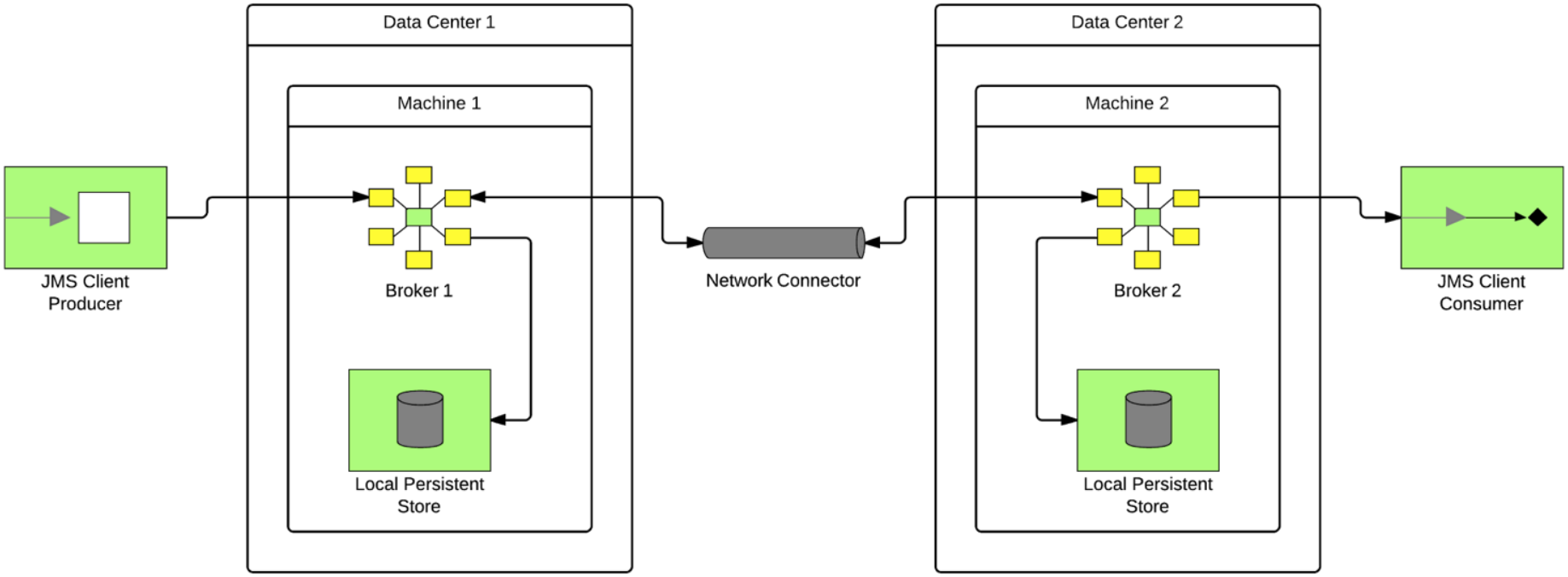
A message can be forwarded to another broker when a valid consumer is present

Brokers can be configured to prioritize local & nearby brokers to reduce network traffic

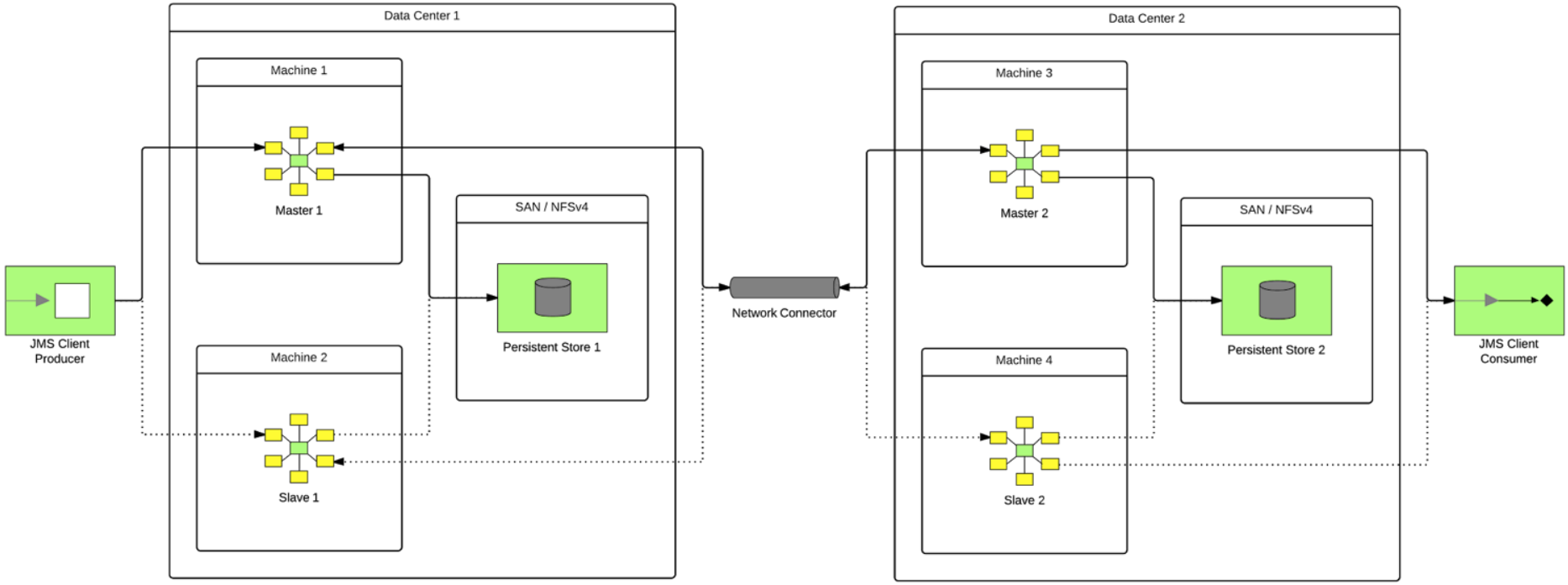
Master/Slave

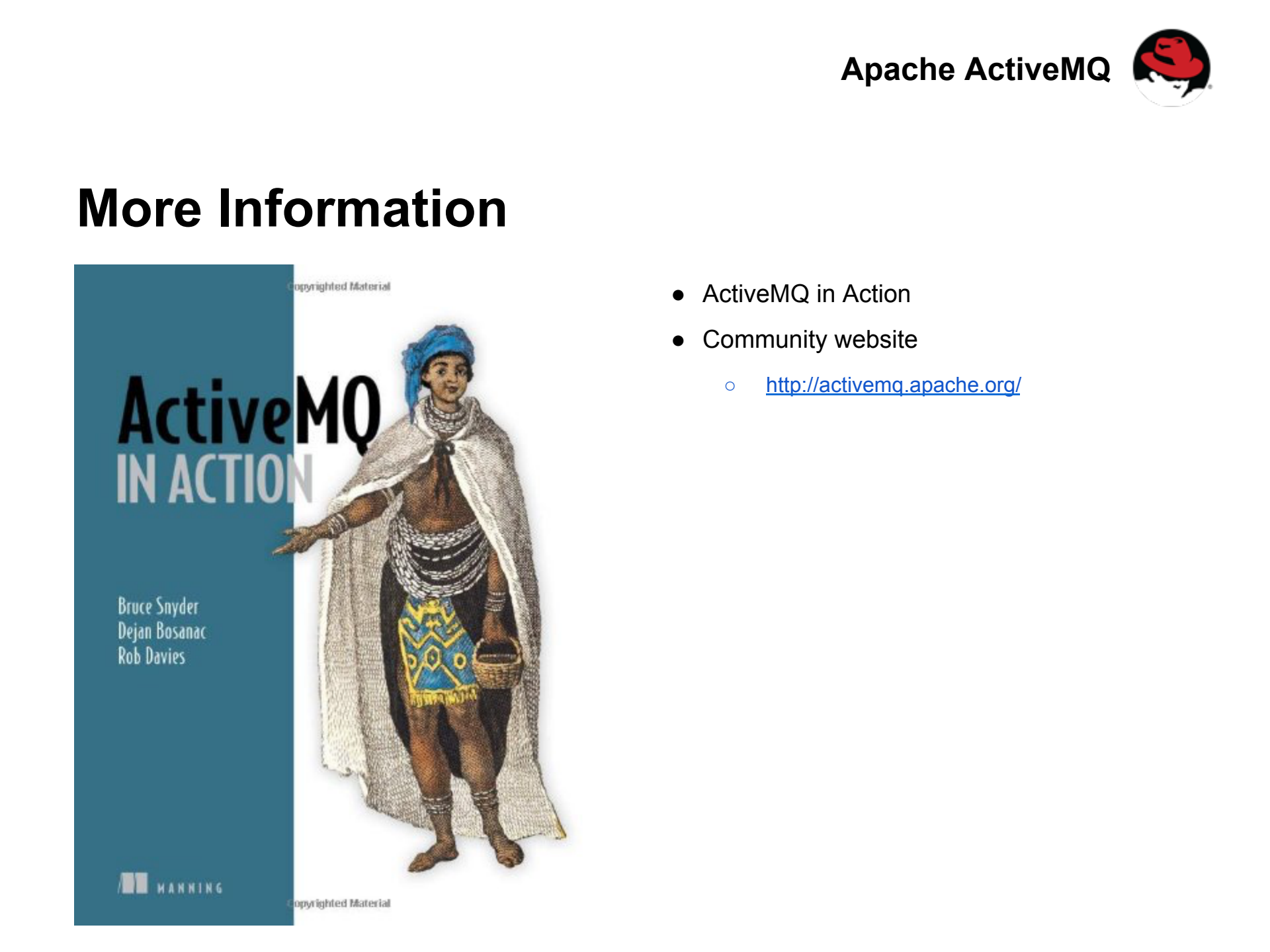


Network of Brokers



Network of Master/Slave





More Information

- ActiveMQ in Action
- Community website
  - http://activemq.apache.org/