

Messaging with Ruby

using Redis & zeromq









advanced k/v store ➡ data structure server

disk-backed, in-memory store

is Redis a NOSQL DB, a cache or a messaging server?

Redis as queue server

Lists

- lists of binary-safe strings (length prefixed byte array)
- push/pop on both ends: can be used as queues
- LPUSH + RPOP / RPUSH + LPOP
- BRPOP/ BLPOP blocks until items are available
- blocking queue means queue without polling

Redis as pubsub server

Publish / Subscribe

- does not involve key/value storage
 - realtime, channeled messaging
 - (UN)SUBSCRIBE to one or more channels
 - P(UN)SUBSCRIBE to channels matching pattern
 - PUBLISH message to channel
-
- buffering for slow connected clients (warning!)

Redis Ruby Libraries

- Ruby client library
<https://github.com/ezmobius/redis-rb>
- Evented Ruby client library
<https://github.com/madsimian/em-redis>

Resque

Resque (pronounced like "rescue") is a Redis-backed library for creating background jobs, placing those jobs on multiple queues, and processing them later.

Resque is:

- a Ruby library for creating, querying, and processing jobs
- a Rake task for starting a worker which processes jobs
- a Sinatra app for monitoring queues, jobs, and workers

<https://github.com/defunkt/resque>

ØMQ \zeromq\ The Intelligent Transport Layer

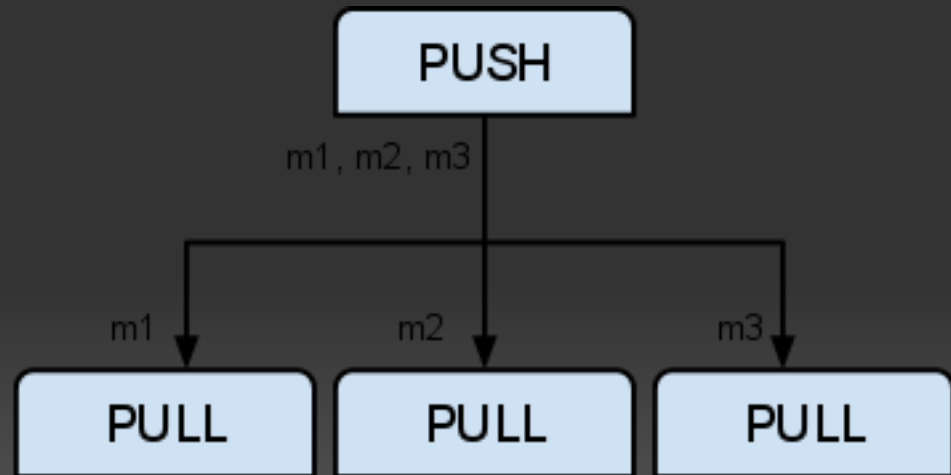
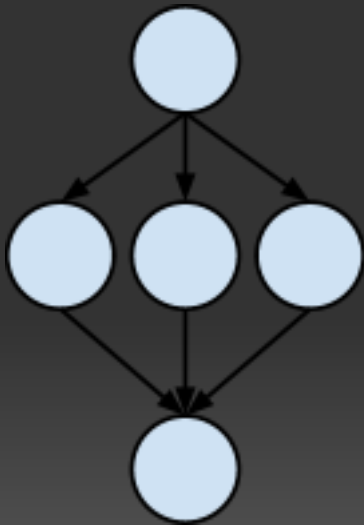
- Ø The socket library that acts as a concurrency framework.
- Ø Faster than TCP, for clustered products and supercomputing.
- Ø Carries messages across inproc, IPC, TCP, and multicast.
- Ø Connect N-to-N via fanout, pubsub, pipeline, request-reply.
- Ø Asynch I/O for scalable multicore message-passing apps.
- Ø Large and active open source community.
- Ø 20+ languages including C, C++, Java, .NET, Python.
- Ø Most OSes including Linux, Windows, OS X.
- Ø LGPL free software with full commercial support from iMatix.

ØMQ \zeromq\

- messaging client library
- support most messaging patterns
 - request/reply, pub/sub, push/pull, pair
- serverless, no intermediate broker
- simple API, similar to sockets
- ØMQ is to sockets what Ruby is to C
- blazing fast! orders of magnitude faster than most AMQP systems
- TCP, MULTICAST, IPC, INPROC transports
- the zero in ØMQ was meant as "zero broker" and "zero latency"

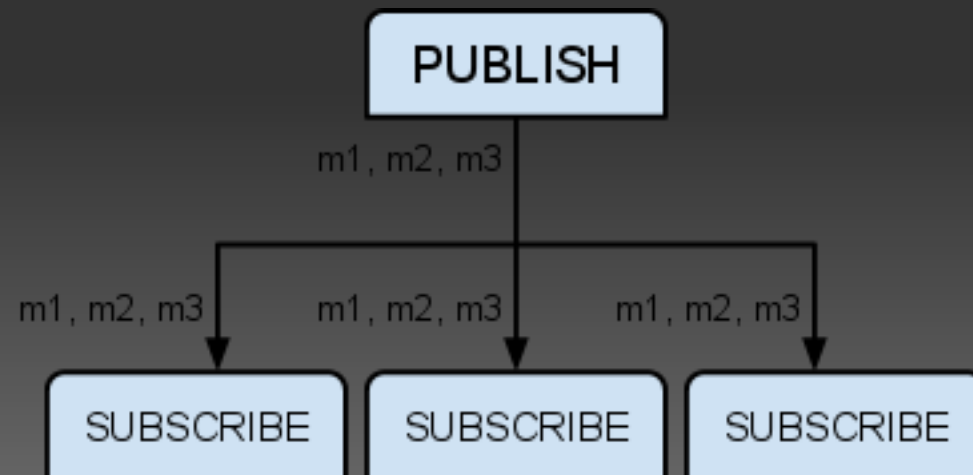
ØMQ PUSH/PULL

- pipelining: connect nodes in a fan-out / fan-in pattern
- load balancing/fair queuing
- parallel tasks distribution



ØMQ PUB/SUB

- connect set of publishers to set of subscribers
- loosely coupled
- *volatile* published stream
- publish on topic
- subscribe on topics / topic patterns



0MQ Ruby Libraries

- Ruby bindings
<https://github.com/zeromq/rbzmq>
- FFI Ruby binding
<https://github.com/chuckremes/ffi-rzmq>
- FFI Evented Ruby bindings
<https://github.com/andrewvc/em-zeromq>

0MQ plugin for RabbitMQ

you can use RabbitMQ as a device in a 0MQ network, gaining rabbitry goodness such as persistence and monitoring

<https://github.com/rabbitmq/rmq-0mq/wiki/>