Messaging at CERN

Lionel Cons – CERN IT/CM

In the Beginning…

- messaging started at CERN ~10 years ago
- goal was to simplify grid middleware
- initiator was the Operations Automation Team (OAT) of the Enabling Grids for E-Science in Europe (EGEE) project
- driving force has been the European Middleware Initiative (EMI) project
- ☞ "*Using ActiveMQ at CERN for the Large Hadron Collider*" (FUSE day, 2010)
- messaging proved to be useful so its use grew…

Main Use Case

- messaging is used to decouple information producers from consumers
  - using different software stacks
  - managed by different teams
  - only sharing the “schema” of the JSON payload
  - published to topic, consumed from virtual queues
- much more WAN than LAN
  - code change could take months to get deployed
- mostly STOMP with very few OpenWire or AMQP
- frequent use of X.509 authentication (grid)

STOMP

- CERN pushed for standardization (STOMP 1.2)
- advantages
  - supported by most brokers
  - decent client libraries available for all languages
  - lightweight (e.g. can publish from PL/SQL)
- drawbacks
  - none for us?

Other Use Case

- messaging is used inside an application
- control on which messaging solution is used (and how!) is very limited
- applications used at CERN:
    - Celery with RabbitMQ
    - MCollective with ActiveMQ
    - OpenStack with RabbitMQ
    - …

MCollective Use Case

- MCollective needs a network of brokers
- initial requirements are challenging:
    - 2 data centers 1000 km apart
    - 30k concurrent connections
    - 300k subscriptions
- worked with Puppet Labs to reduce the number of subscriptions: now only 150k
- works fine (except an abnormally large number of connection timeout warnings)
- must scale with the growth of nodes we manage

IT Managed Messaging Services

- 17 different clusters (test and production)
- 44 brokers
- 267 applications
- average message rates: ~1k Hz in and ~5kHz out
- ~6k destinations (topics and queues)
- ~25k concurrent connections
- ~120k subscriptions

- all run Red Hat A-MQ 6

Messaging Monitoring

- all log files analyzed
- 103 different metrics collected
  - messaging metrics collected through Jolokia
- 1350 checks every minute
  - e.g. per-client messages received per second too low/high
- expert system named Metis using Esper
  - time aggregations like min/max/average
  - other aggregations like “all brokers in a cluster”
  - hysteresis
  - patterns
  - …

Accelerators Controls (1/2)

- transport data from middle tier servers to GUIs and storage system but also log messages, infrastructure monitoring & audit information
- broad usage pattern in message size & frequency
- criticality service: **No JMS, No Beam**
- ☞ "*Large Scale Messaging with ActiveMQ for Particle Accelerators at CERN*" (CamelOne 2012)

courtesy of Felix Ehm (BE/CO)

Accelerators Controls (2/2)

- 25 brokers organized in dedicated services:
  5 HA clusters and 15 single instances
- running on *physical* machines
- up to: >270GB per day, 8k messages per second
- currently: Apache ActiveMQ 5.12.2
- middleware team reviewing the usage of JMS: the idea is to simplify the environment and extend the use of ØMQ which is already being used for Remote Device Access (RDA)

courtesy of Felix Ehm (BE/CO)

Pushing the Limits

- creative network of brokers topology (MCollective)
- peaks of several thousands of messages per second (from a single client)
- some large messages, up to 100MB
- sometimes huge backlogs (several days) while some consumers are down
- tens of new connections per second
- X.509 authentication (JAAS) with ~70k entries

Possible Evolutions (1/2)

- Red Hat A-MQ 6.x:
  - end of full support: Jan 2018
- Apache ActiveMQ 5.x:
  - will be supported as long as it is widely used
- Red Hat A-MQ 7.x:
  - only one alpha and one beta released so far
  - beta1 based on Artemis 1.3.0
- Apache ActiveMQ 6.x (aka Artemis):
  - latest version is 1.5.1
  - major changes coming with 2.0.0
  - ~50 unresolved bugs in Jira important for our use cases

Possible Evolutions (2/2)

- ØMQ: speed and low latency
  - already used in accelerators controls
  - CERN is even quoted in ØMQ's "*Learn the Basics*"
- Kafka: high volumes and scalability
  - big overlap with traditional messaging
  - seen more as *complement* rather than *replacement*
- RabbitMQ: another widely used messaging broker

Summary

- messaging started at CERN ~10 years ago
- one use was the simplification of grid middleware
- messaging was also used in accelerators controls
- it also spread to several very different areas
- messaging still is widely used at CERN
- some use cases are quite challenging
- some overlapping technologies are appearing
- messaging at CERN will evolve to adapt to changes both in requirements and solutions