Sirius: Distributing and Coordinating Application Reference Data

Michael Bevilacqua-Linn, Maulan Byron, Peter Cline, Jon Moore, and Steve Muir

Comcast Cable

@jon_moore
What is “Reference Data” anyway?

4.2 GB!
wikipedia

43 GB!

Uncompressed!
App Server

ORM

Reference Data

“Obscuring the Real Mechanism”
App Server

- IOExceptions
- Timeouts
- Cache misses
- Cxn pools
App Server

- Data structures
- Algorithms
- Unit tests
- Profilers
Application

enqueue\{Put,Delete\}

Sirius

Request Handler

handle

apply

Paxos

persist

txn log
ingest

client service

client service

Photo: http://www.flickr.com/photos/37080143@N08/3409059834
by SimCity Research Laboratory: http://www.flickr.com/photos/37080143@N08/
License: http://creativecommons.org/licenses/by-sa/2.0/
App Server

write

segment
App Server

write

segment

segment
write
App Server

write

compact

segment

segment

segment
App Server

segment

compact

write

segment

segment
Implementation
def receive = {
  case Request(command: Command) =>
    propose(command)

  case decision @ Decision(slot, decisionCommand) =>
    decisions.put(slot, decisionCommand)
    reproposeIfClobbered(slot, decisionCommand)
    try {
      performFun(decision)
    } catch {
      case t: Throwable =>
        logger.warning("...")
    }

  case decisionHint @ DecisionHint(decisionHintSlotNum) =>
    slotNum = decisionHintSlotNum + 1
    outstandingProposals.filter((k, _) => k > decisionHintSlotNum)
    decisions.filter((k, _) => k > decisionHintSlotNum)
    localLeader forward decisionHint

  case Reap =>
    reapStagnantProposals()
}

Conclusions
Sirius provides access to reference data with:

- Native, arbitrary data structures
- Eventually consistent replication
- Persistence and replay
- Convenient library interface

http://comcast.github.io/sirius/