Big Data Platform at Pinterest

Mao Ye

Data Architecture

Design Choices for Hadoop Platform

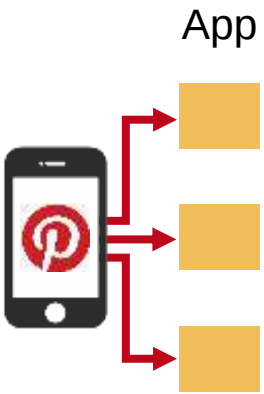
Pinball for Workflow Management

Data Architecture

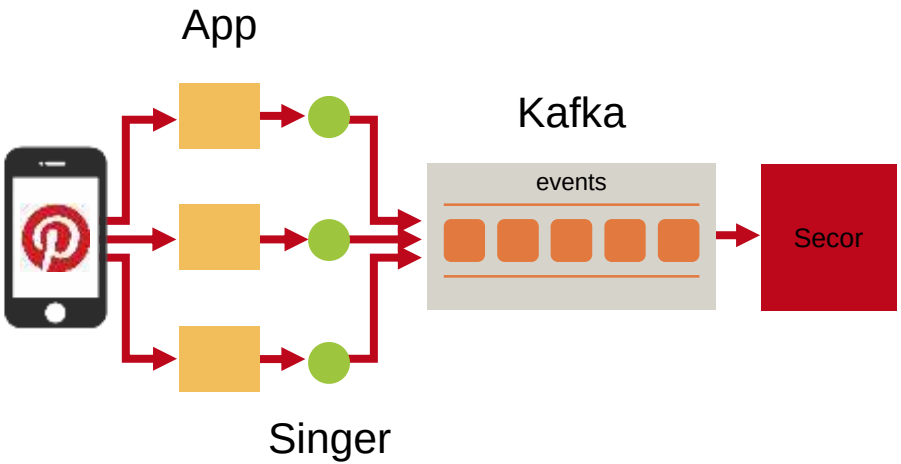
Data at Pinterest

- 60 Billion Pins
- 1 Billion boards
- 100M MAU
- 60 PB of data on S3
- 3 PB processed every day
- 2000 node Hadoop cluster
- 250 engineers

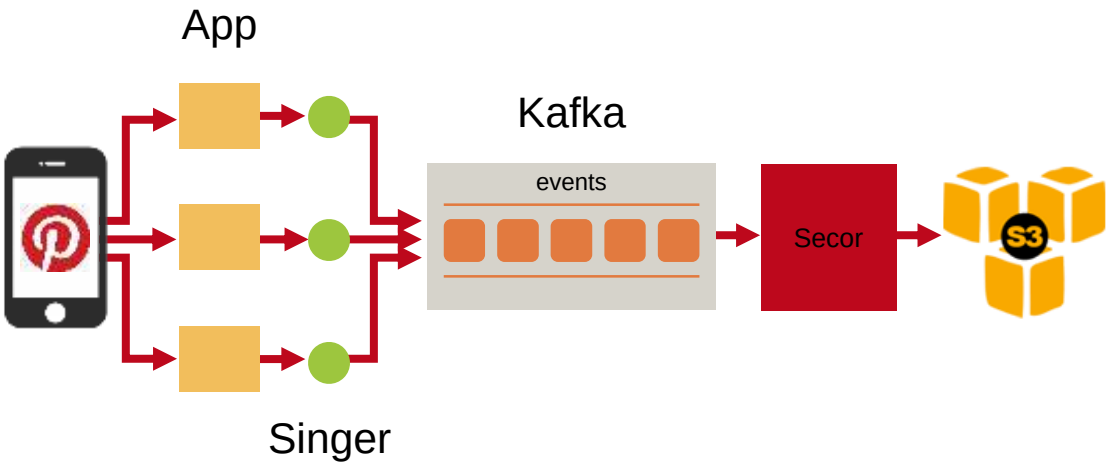
Pinterest Data Architecture



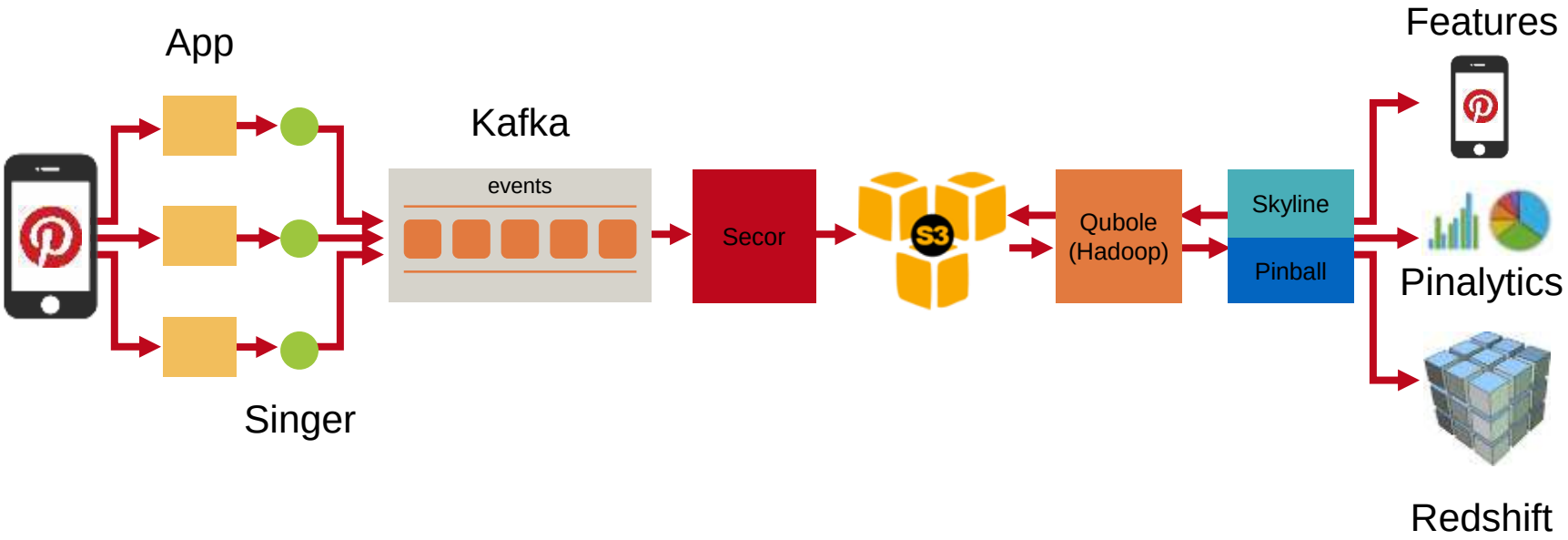
Pinterest Data Architecture



Pinterest Data Architecture



Pinterest Data Architecture

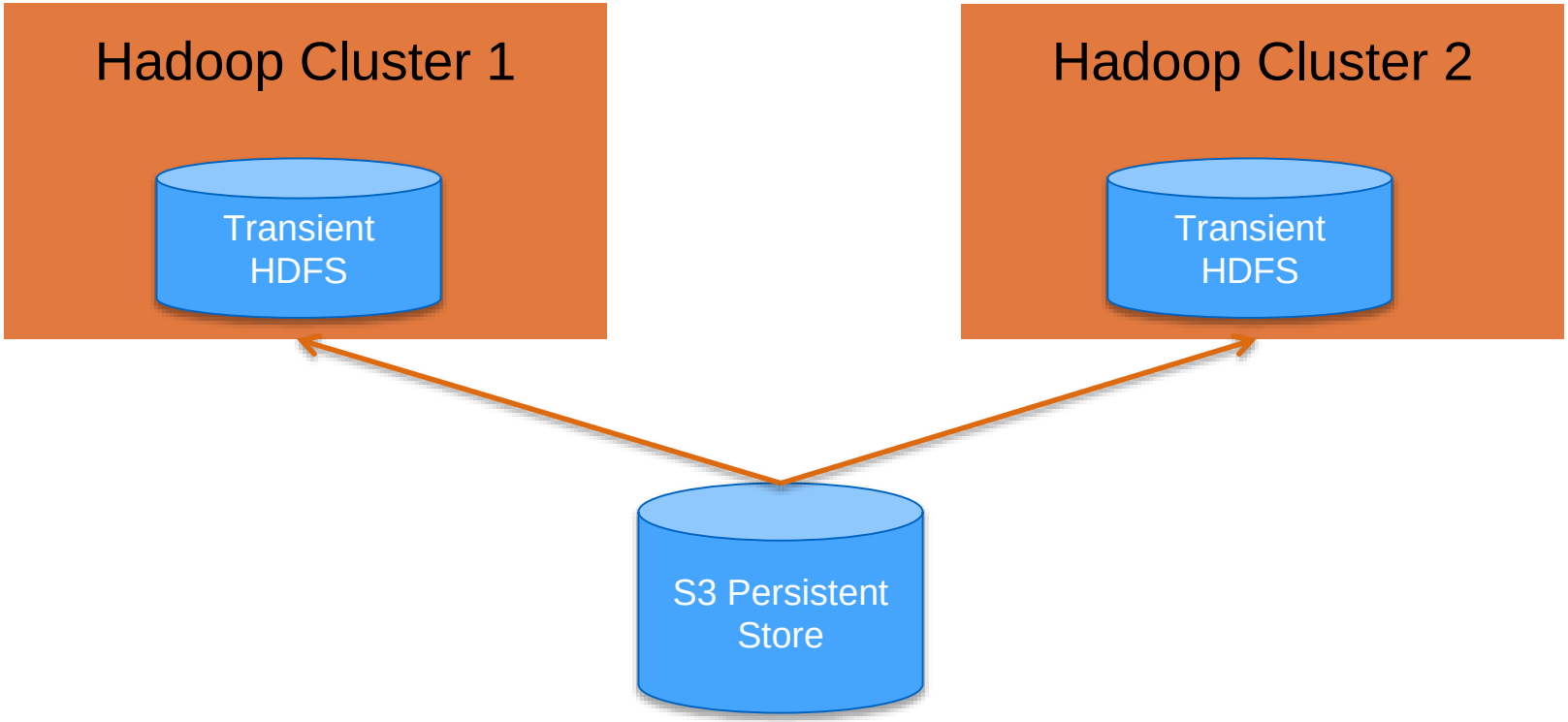


Design Choices for Hadoop Platform

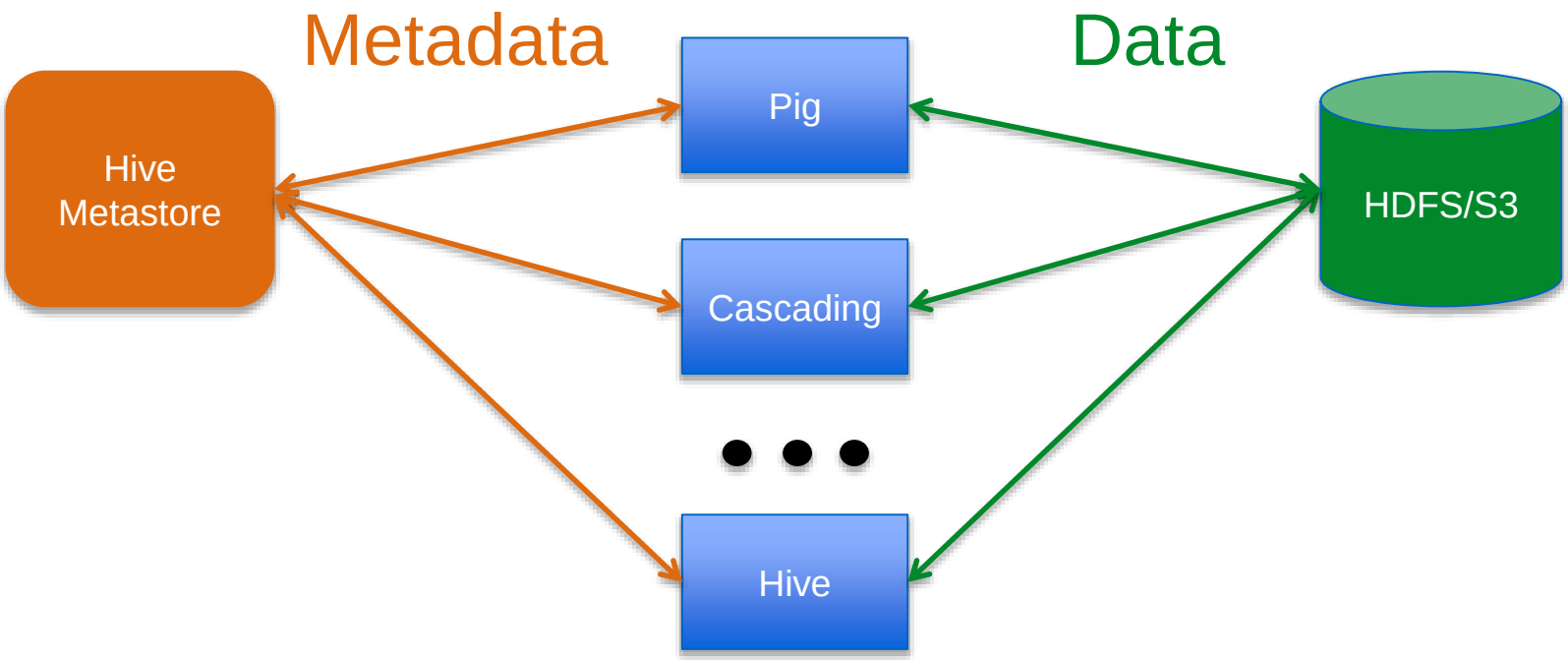
Hadoop Platform Requirements

- Isolated multi-tenancy
- Elasticity
- Support multiple clusters
- Ephemeral clusters
- Access control layer
- Shared data store
- Easy deployment

Decoupling compute & storage



Centralized Hive Metastore

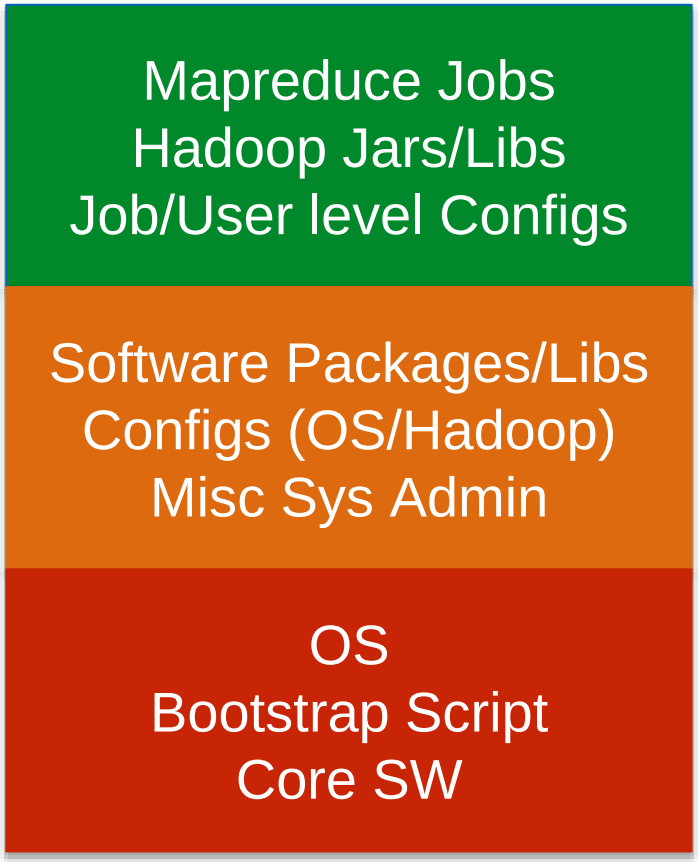


Multi-layered Packaging

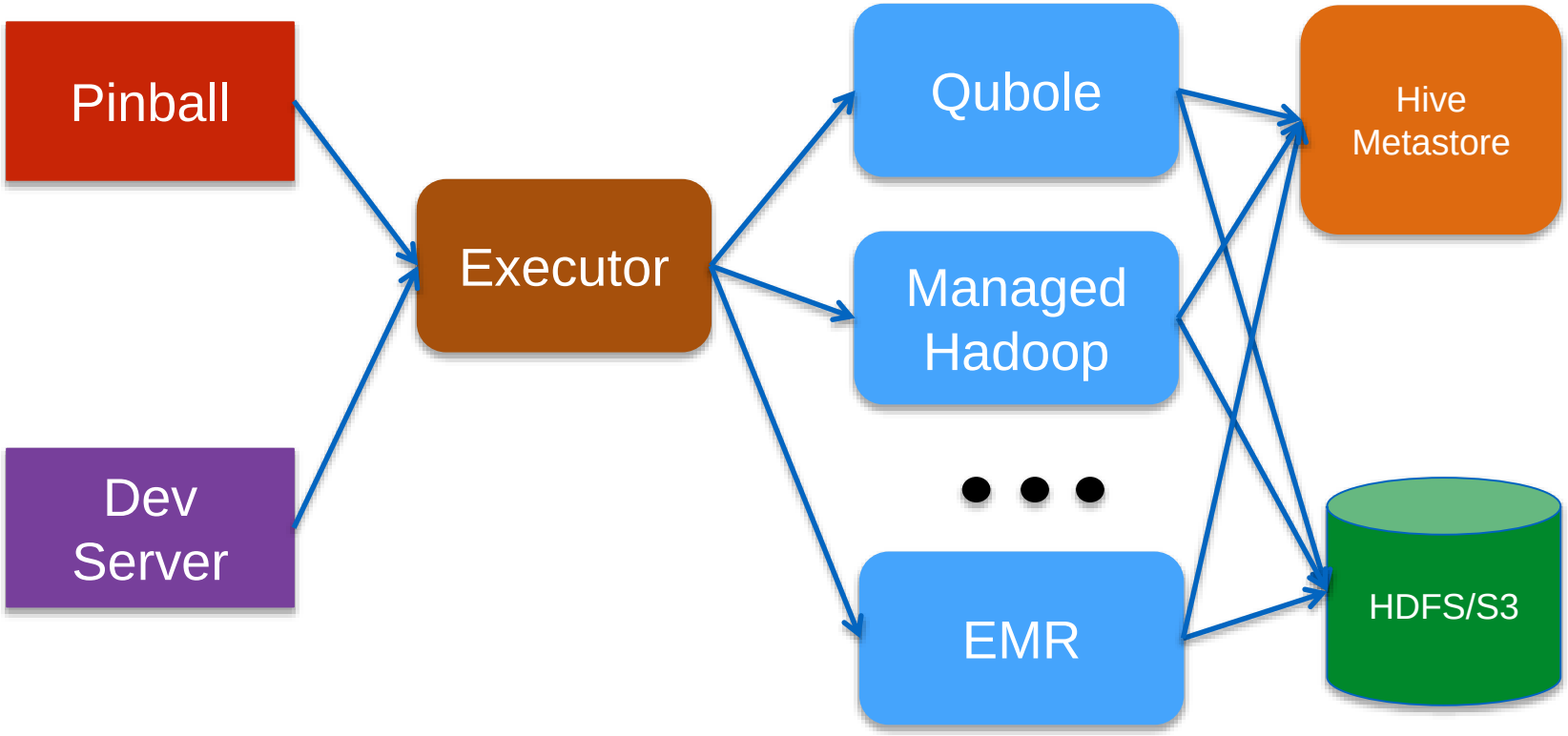
Runtime Staging
(on S3)

Automated Configuration
(Masterless Puppet)

Baked AMI



Executor Abstraction Layer



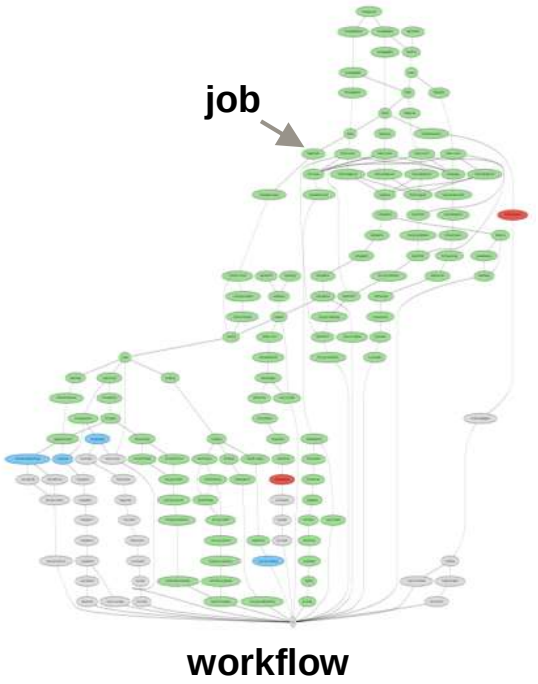
Why Qubole?

- Hadoop & Spark as managed services
- Tight integration with Hive
- Graceful cluster scaling
- API for simplified executor abstraction
- Advanced support for spot instances
- Baked AMI customization

Pinball for Workflow Management

Scale of Processing

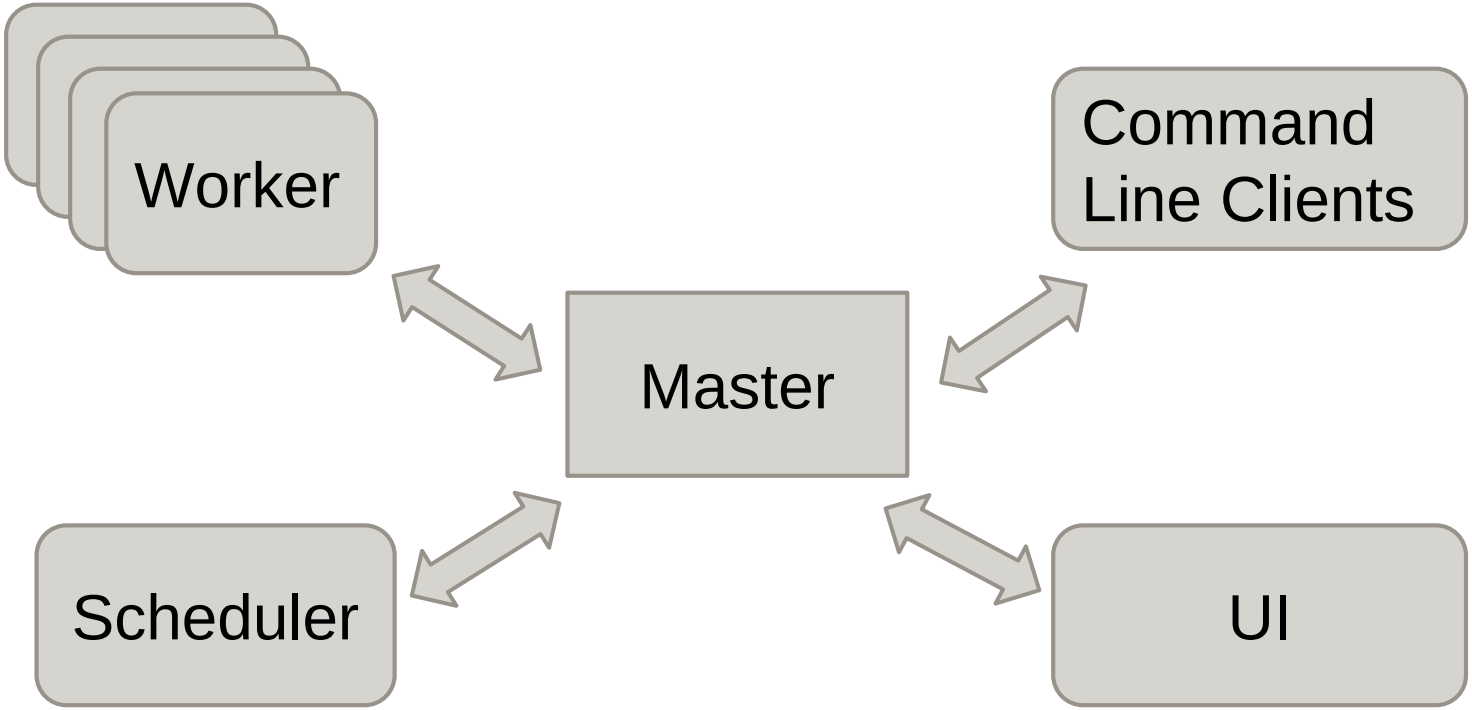
- **Scale:**
  - 60 Billion Pins
  - Hundreds of workflows
  - Thousands of jobs
  - 500+ jobs in a workflow
  - 3 petabytes processed daily
- **Support:**
  - Hadoop, Cascading, Hive, Spark …



Why Pinball?

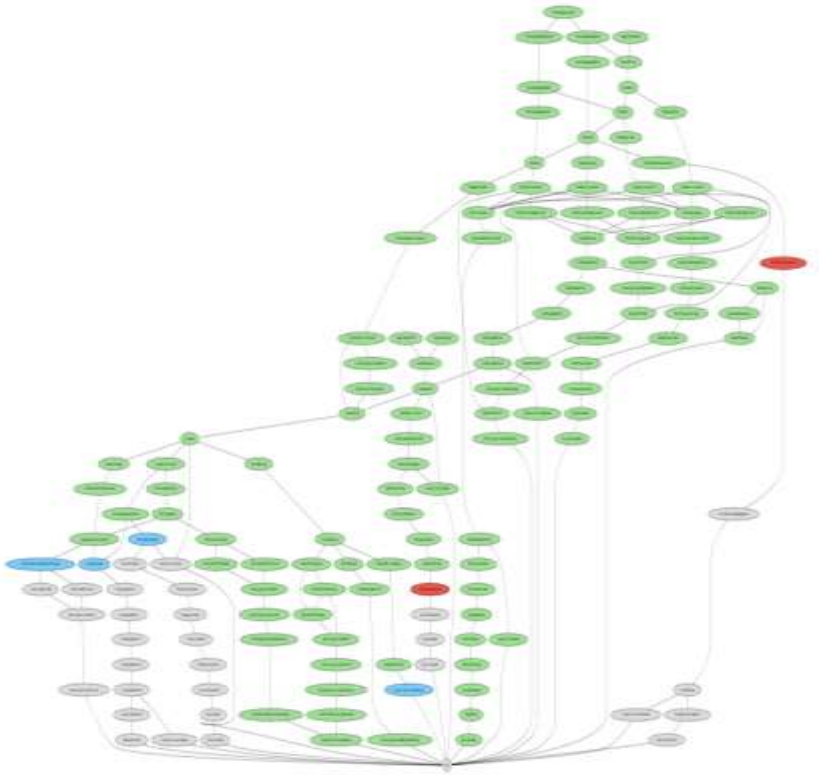
- **Requirements**
  - Simple abstractions
  - Extensible in future
  - Reliable stateless computing
  - Easy to debug
  - Scales horizontally
  - Can be upgraded w/o aborting workflows
  - Rich features like auto-retries, per-job emails, overrun policies…
- **Options**
  - Apache Oozie, Azkaban, Luigi

Pinball Design



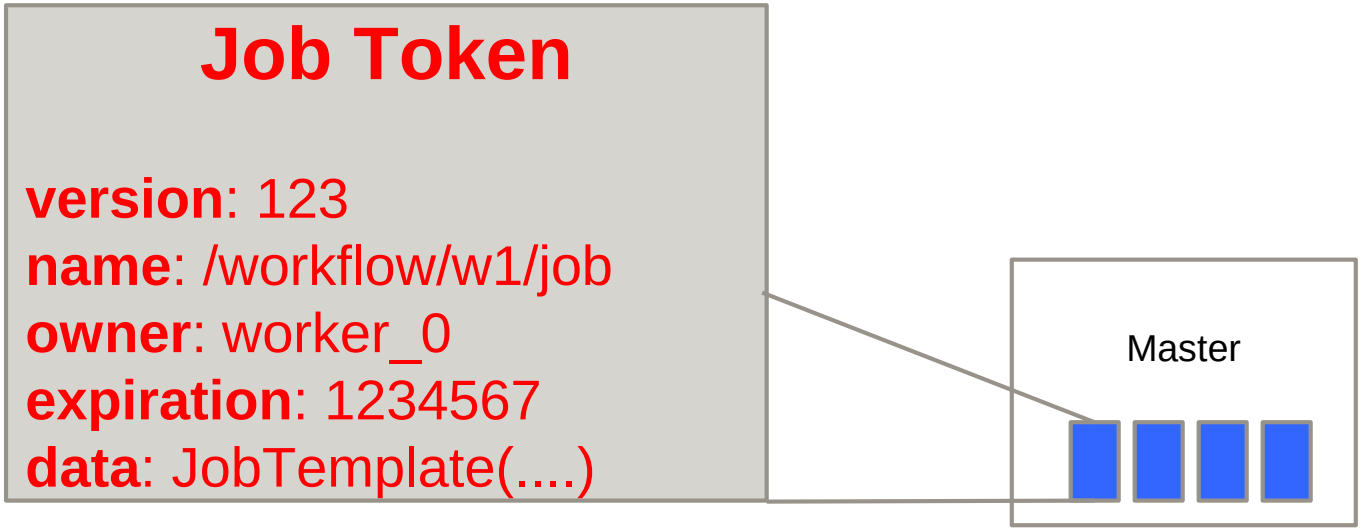
Workflow Model

- **Workflow**
  - A directed graph of nodes called jobs
- **Edge**
  - Run after dependence
- **Node**
  - Job is a node

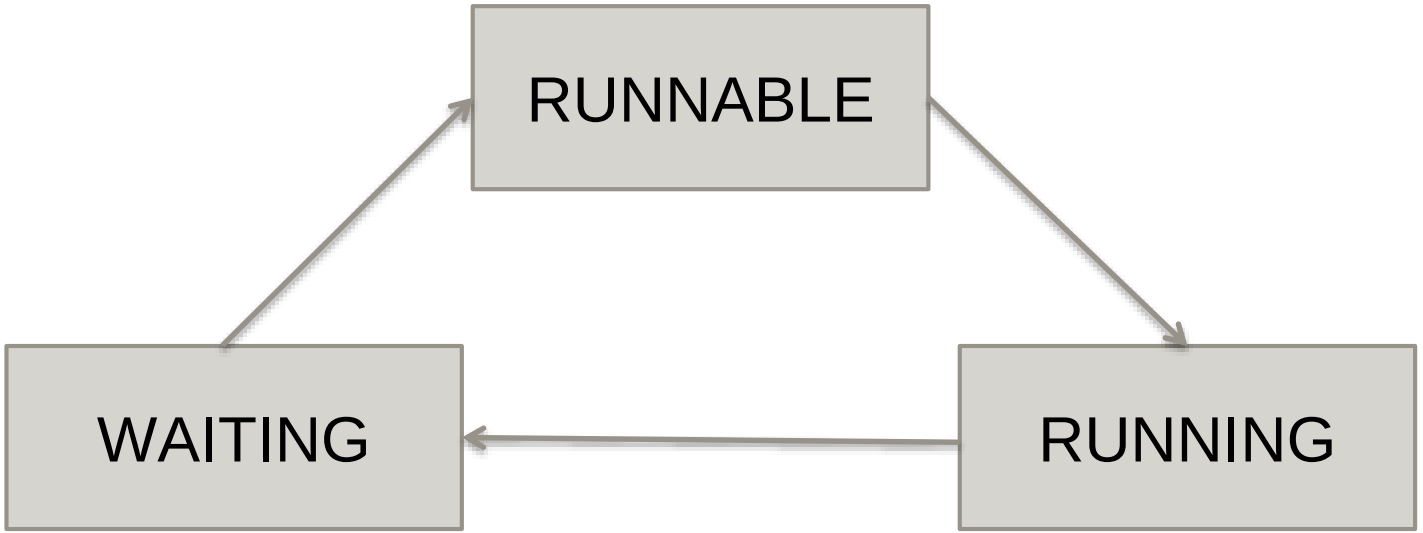


Job State

- Job state is captured in a token
- Tokens are named hierarchically

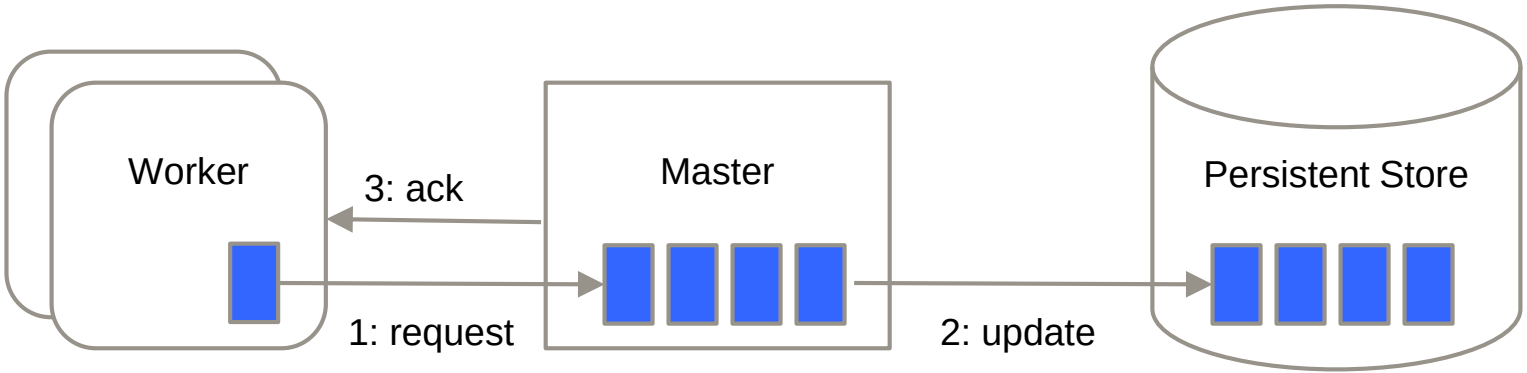


Job State Machine



Master Worker Interaction

- Master keeps the state
- Workers claim and execute tasks
- Horizontally scalable



Master

- Entire state is kept in memory
- Each state update is synchronously persisted before master replies to client
- Master runs on a single thread – no concurrency issues

Worker

```
while (true) {
  job = master.get_RUNNABLE_job();
  if (job) {
    change job state to RUNNING;
    run the job;
    if job failed {
      depending on the config, either abort the workflow or do nothing (i.e., continue with other jobs);
    } else {
      post events to downstream jobs and check if some of them can be made RUNNABLE;
      if all inputs of the job are satisfied {
        make the job RUNNABLE
      } else {
        make the job WAITING
      }
    }
  }
}
```

Open Source

Git repo:

https://github.com/pinterest/pinball

Mailing list:

https://groups.google.com/forum/#!forum/pinball-users

Thank You