Multi-tenant Machine Learning
Apache Aurora & Apache Mesos

Stephan Erb
2016.11.15

serb@apache.org
@ErbStephan

blueyonder

Apache Aurora

https://aurora.apache.org

Mesos framework for the deployment and scaling of stateless and fault tolerant services in a datacenter

Apache Mesos

https://mesos.apache.org

Cluster manager providing fault-tolerant, fine-grained multitenancy via containers

Apache Aurora

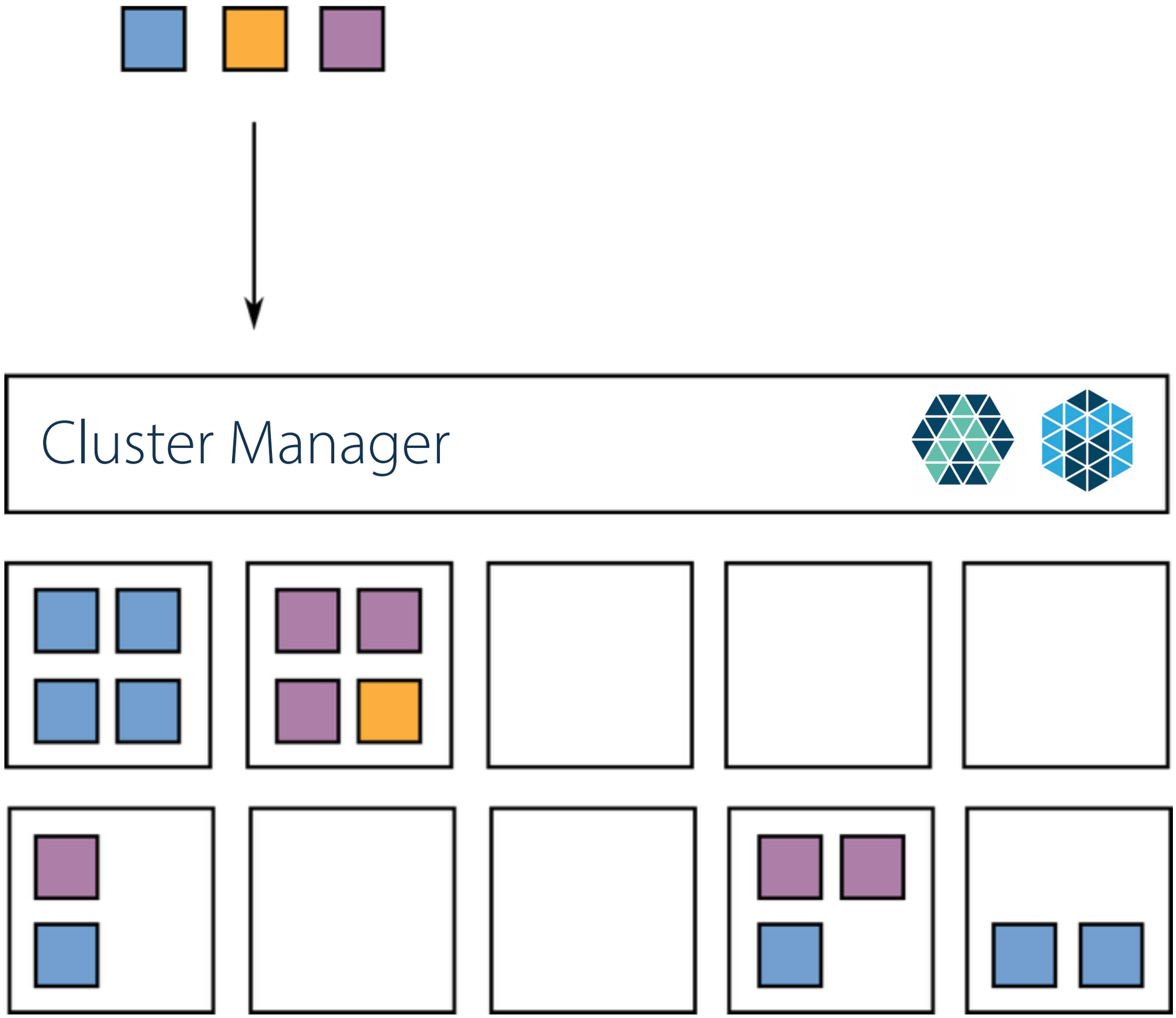
https://aurora.apache.org

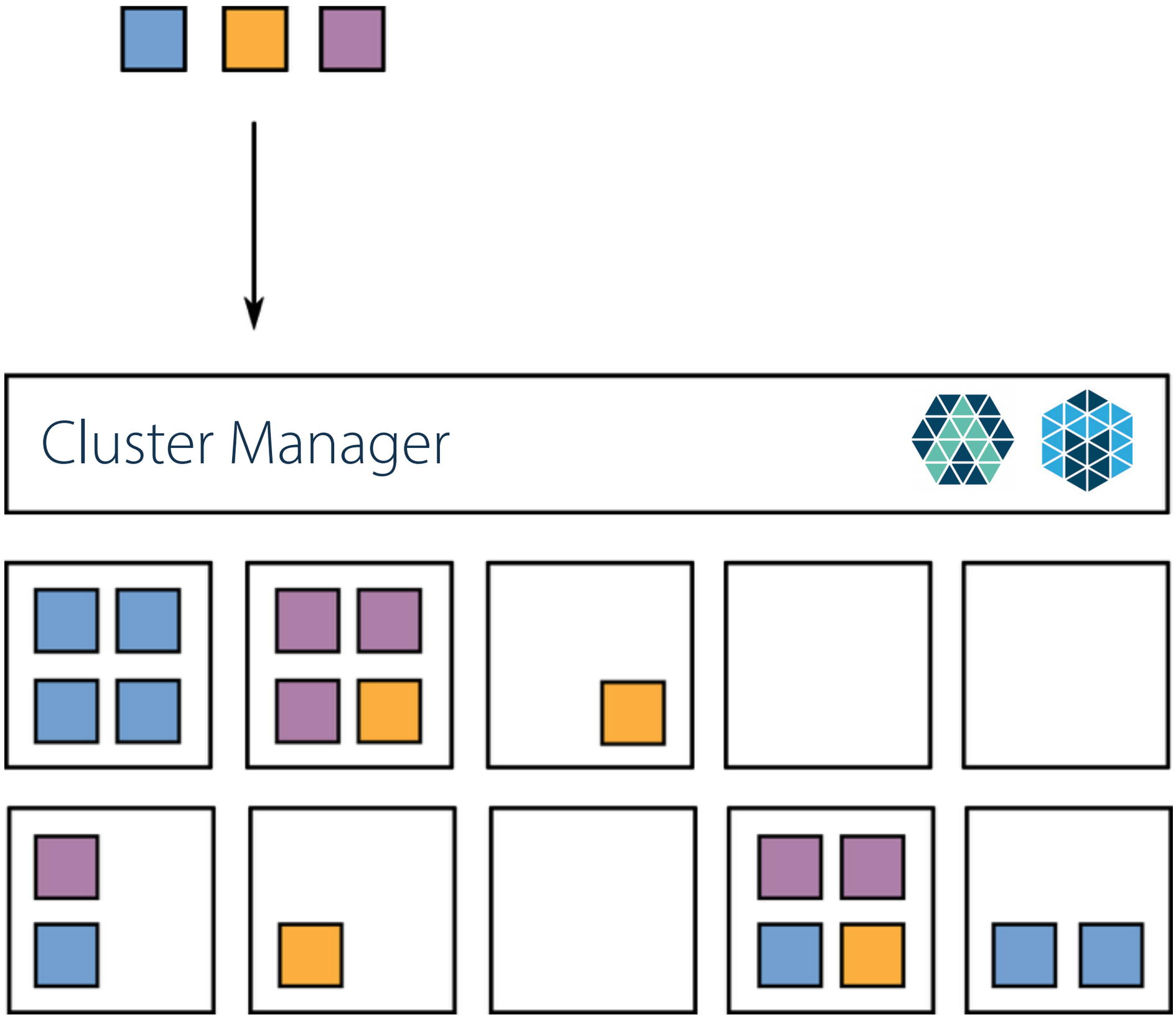
“distributed supervisord"

Apache Mesos

https://mesos.apache.org

“plumbing”


Cluster Manager


Cluster Manager

Aurora Example

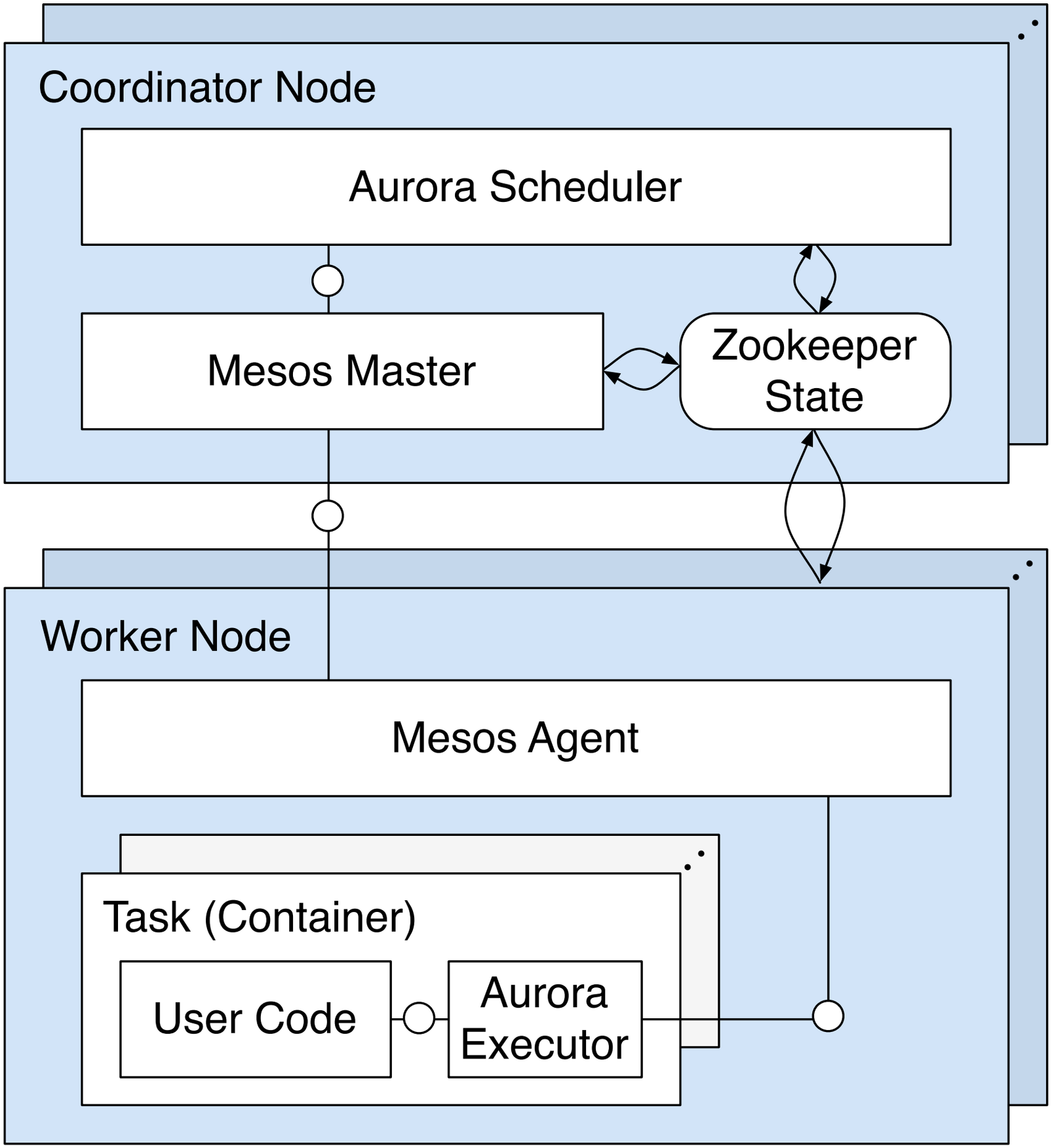
```
webservice = Process(
  name = 'webservice',
  cmdline = ‘./run_my_webservice.py’)

task = Task(
  processes = [webservice],
  resources = Resources(cpu=4, ram=4*GB, disk=8*GB))

jobs = [
  Job(
    task=task,
    instances=4,
    constraints = {'host': 'limit:1'},
    service=True,
    cluster=‘rz1', role=‘www’, environment=‘prod’,
    name=‘webserver’),
]
```

Aurora Example

```
$ aurora update start rz1/www/prod/webserver \
    webserver.aurora
```


Coordinator Node
Aurora Scheduler
Mesos Master
Zookeeper
State
Worker Node
Mesos Agent
Task (Container)
User Code
Aurora
Executor

devcluster / www-data / prod / hello

Update In Progress

2 of 3

STARTED BY **AURORA**
2 MINUTES AGO

Active tasks (3) | Completed tasks (0) | All tasks

Configuration Overview

0-2

show config

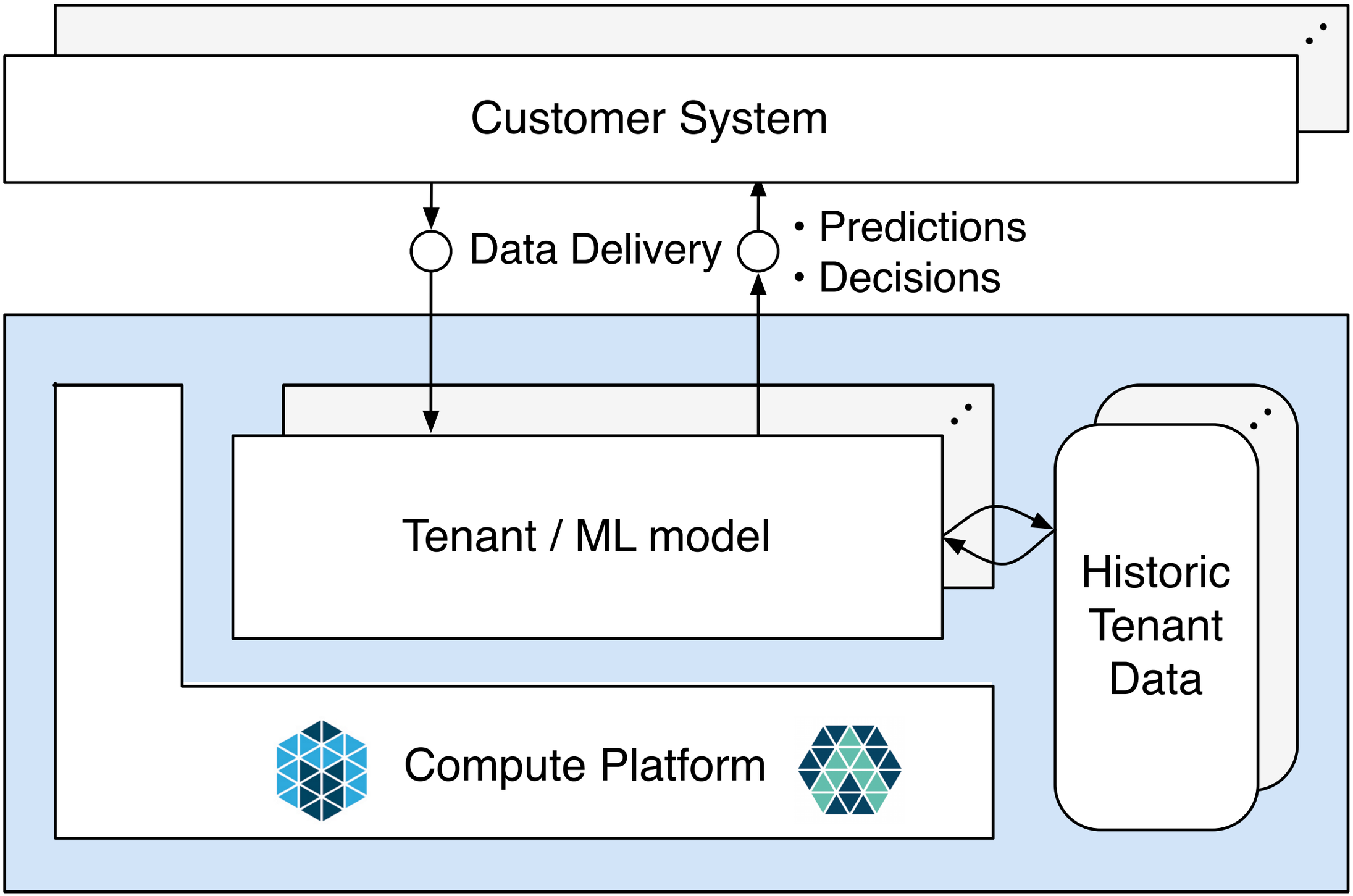
Instance	Status	Host
0	✚ 8 minutes ago - RUNNING	192.168.33.7
1	✚ 2 minutes ago - RUNNING	192.168.33.7
2	✚ a minute ago - RUNNING	192.168.33.7

Update History

id	status	started	last modified	user
0b51787d-ccd5-4c82-a888-1d37b500b0ce	IN PROGRESS	2 minutes ago	a minute ago	aurora


blue yonder

Photo by liz west https://flic.kr/p/7qYh21


Customer System
Data Delivery
• Predictions
• Decisions
Tenant / ML model
Historic Tenant Data
Compute Platform

Key Achievement

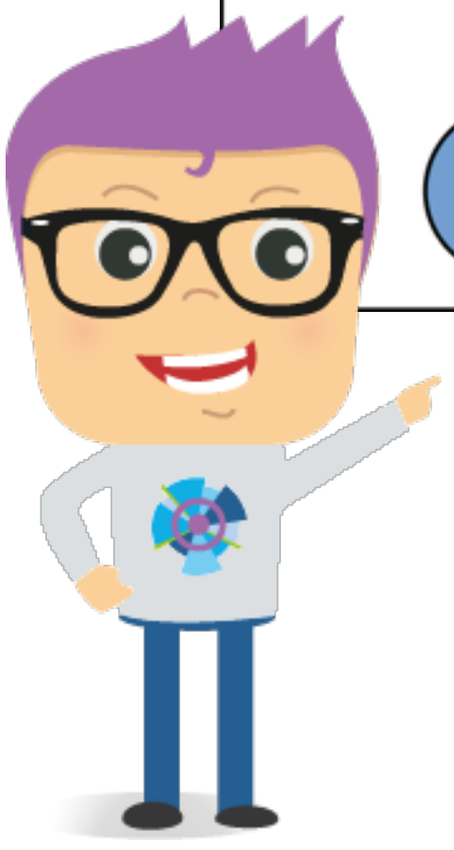
Data scientists deploy to production.

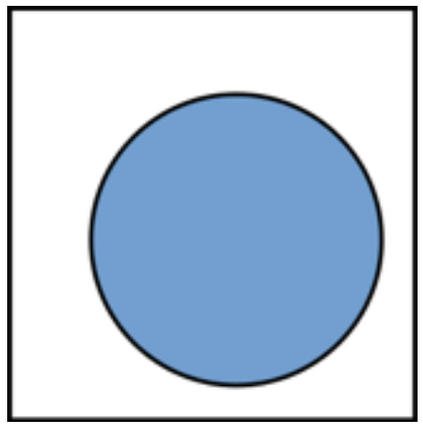
Key Achievement

Data scientists deploy to production.

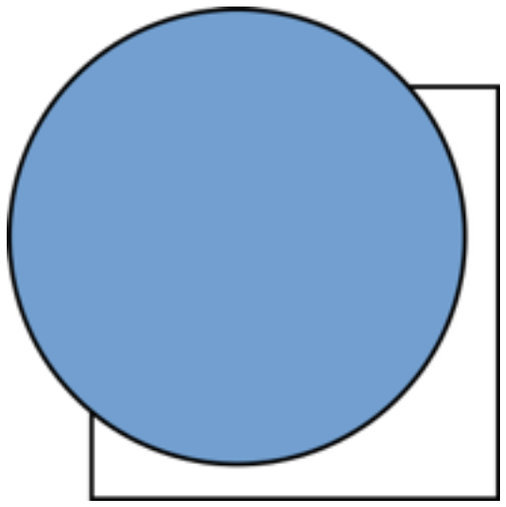
Key Achievement

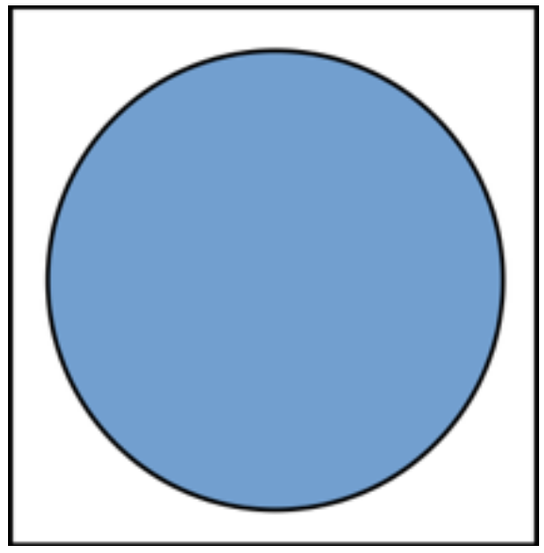
Data scientists deploy to production.





VM/Host





bigger
VM/Host

Data larger than RAM

Implementation Choices:

- semi-**external** implementation (out-of-core)
- communication-efficient **distributed** memory implementation
- **streaming** (aka “large data volumes are hard, infinite data is easy”)

Data larger than RAM

Implementation Choices:

- semi-**external** implementation (out-of-core)
- communication-efficient **distributed** memory implementation
- **streaming** (aka "large data volumes are hard, infinite data is easy")

Domain-specific Problem Decomposition

```
# Compute on whole data set
#
compute_prediction(data)
```

```
# Compute on partitioned data
#
# (this is rather restrictive but tends to
#  work great for many usecases)
#
for chunk in partition(data):
    compute_prediction(chunk)
```

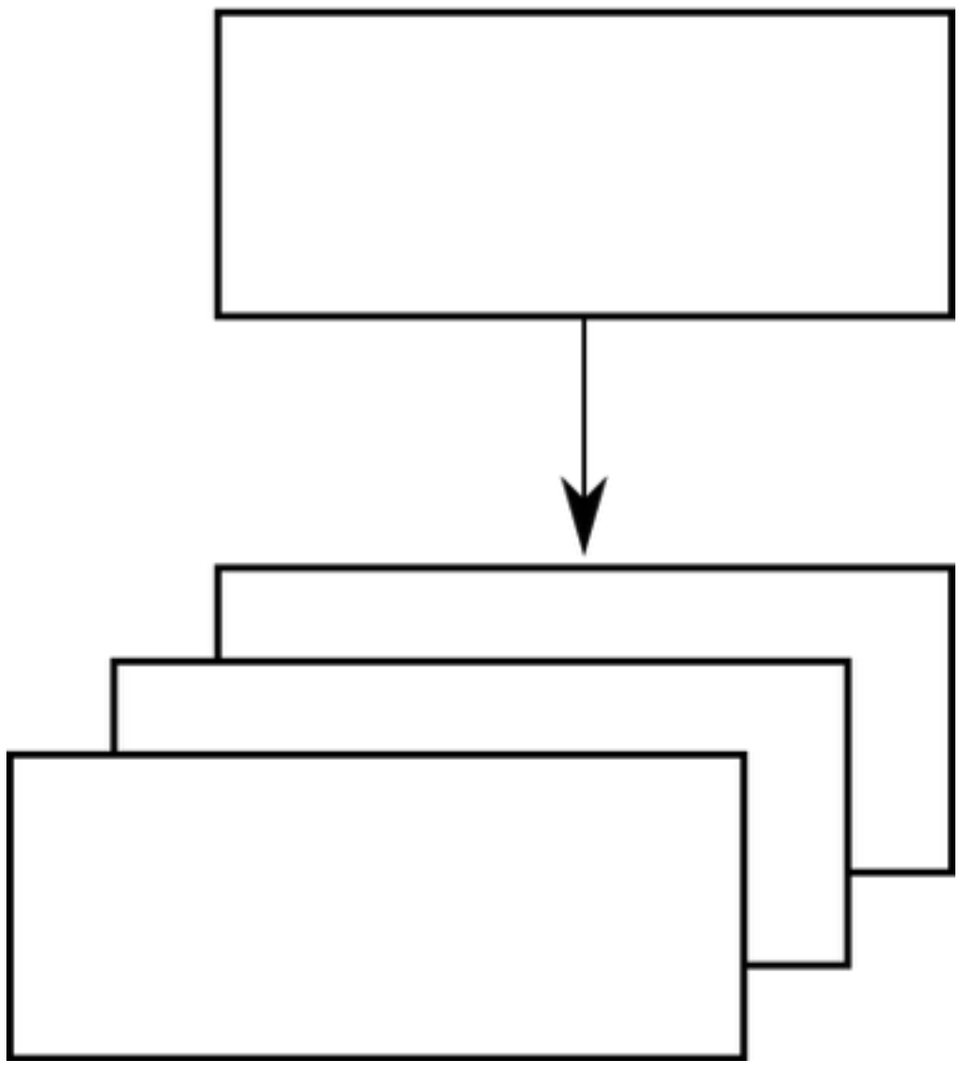
Python Scheduling

Master

- manages job graphs
- guarantees fault tolerance

Workers

- run python functions
- distributable
- dynamic worker count



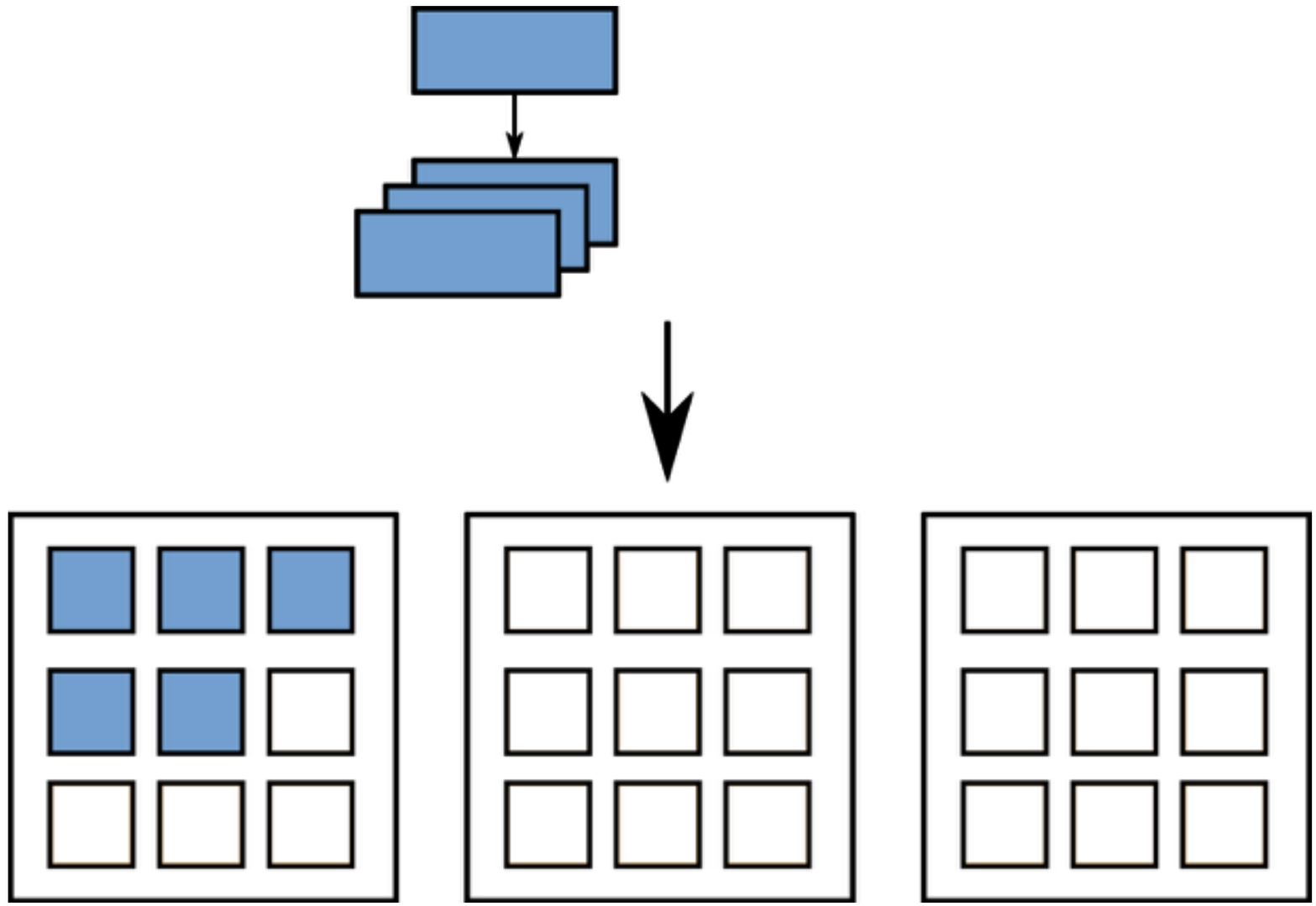
http://www.celeryproject.org/

http://distributed.readthedocs.io/en/latest/

Cluster Scheduling

Project/ Tenant

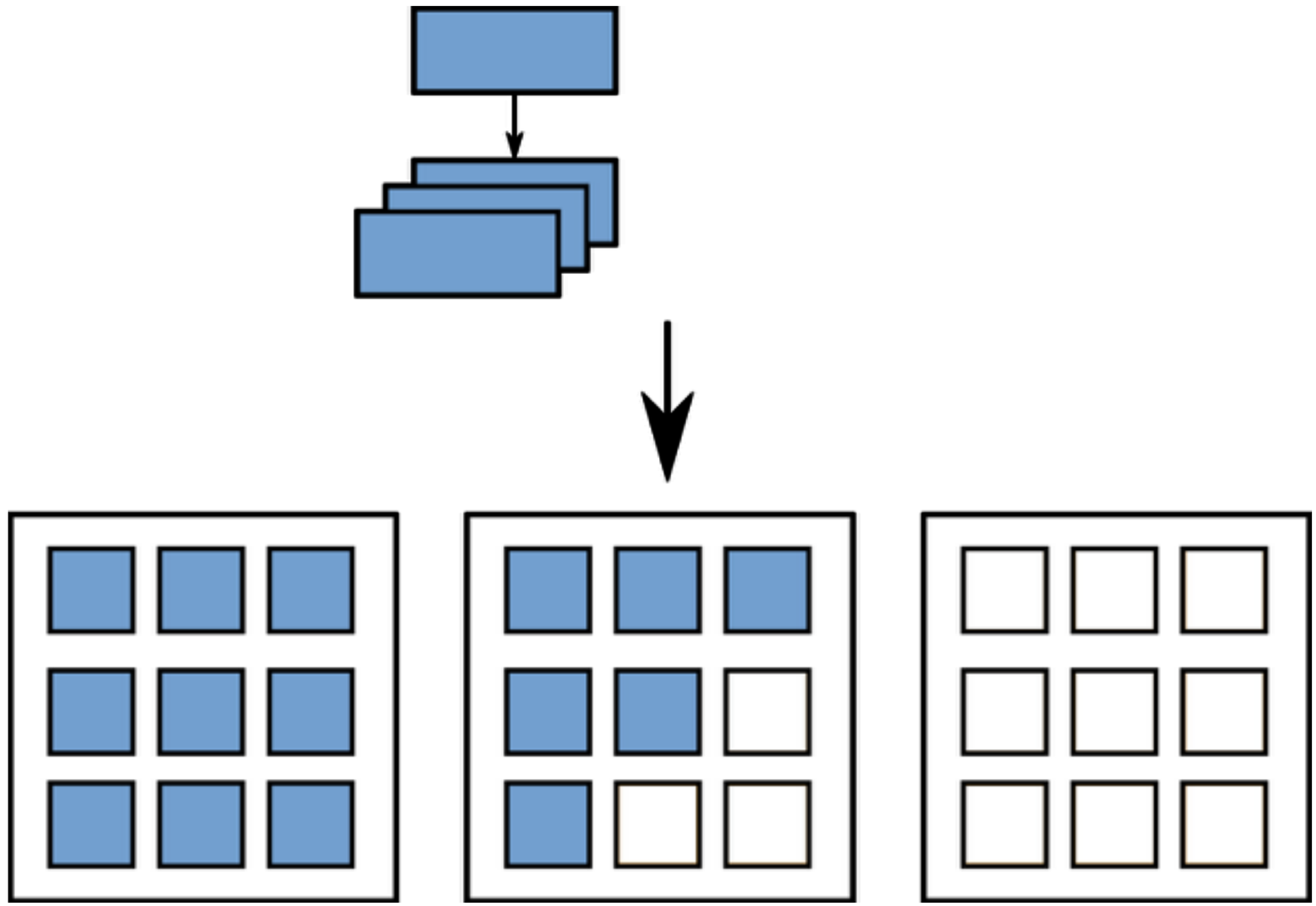
Compute Cluster



Cluster Scheduling

Project/ Tenant

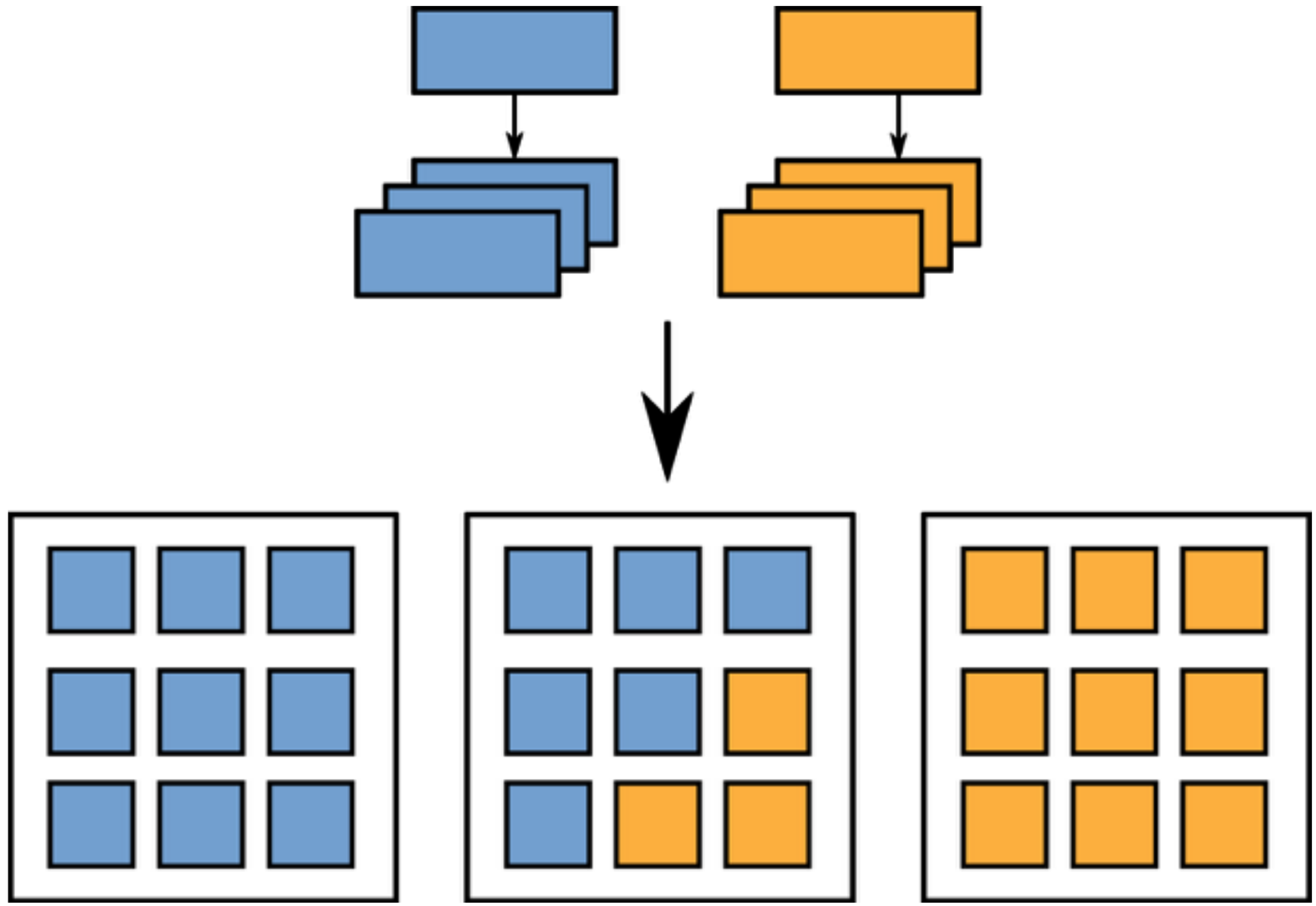
Compute Cluster



Cluster Scheduling

Project/ Tenant

Compute Cluster



Key Idea

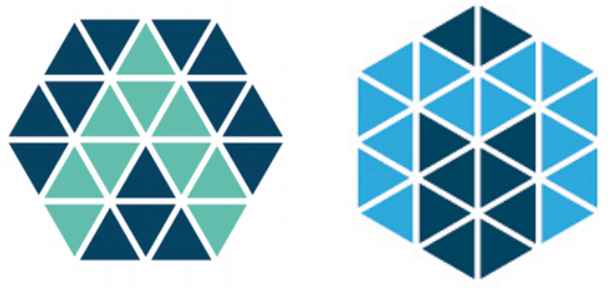
Multi-tenancy via multi-instance deployments

Good multi-tenancy is hard enough that it just doesn't happen by accident.

— Jay Kreps

https://www.confluent.io/blog/sharing-is-caring-multi-tenancy-in-distributed-data-systems

Multi-tenant Features



Aurora

- Structured job keys
  - role (tenant01, …)
  - environments (devel, …)
  - name
- Job tiers/priorities
- Quota & preemption

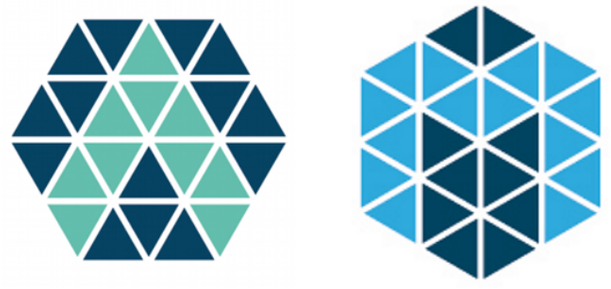
Mesos

- Linux users
- Filesystem isolation via Docker/Appc containers
- CPU/RAM isolation via cgroups
- Linux namespaces (pid, network, …)
- Multi-framework support

Merits and Pitfalls?

Multiple frameworks on the same Mesos cluster

Feature Dimensions



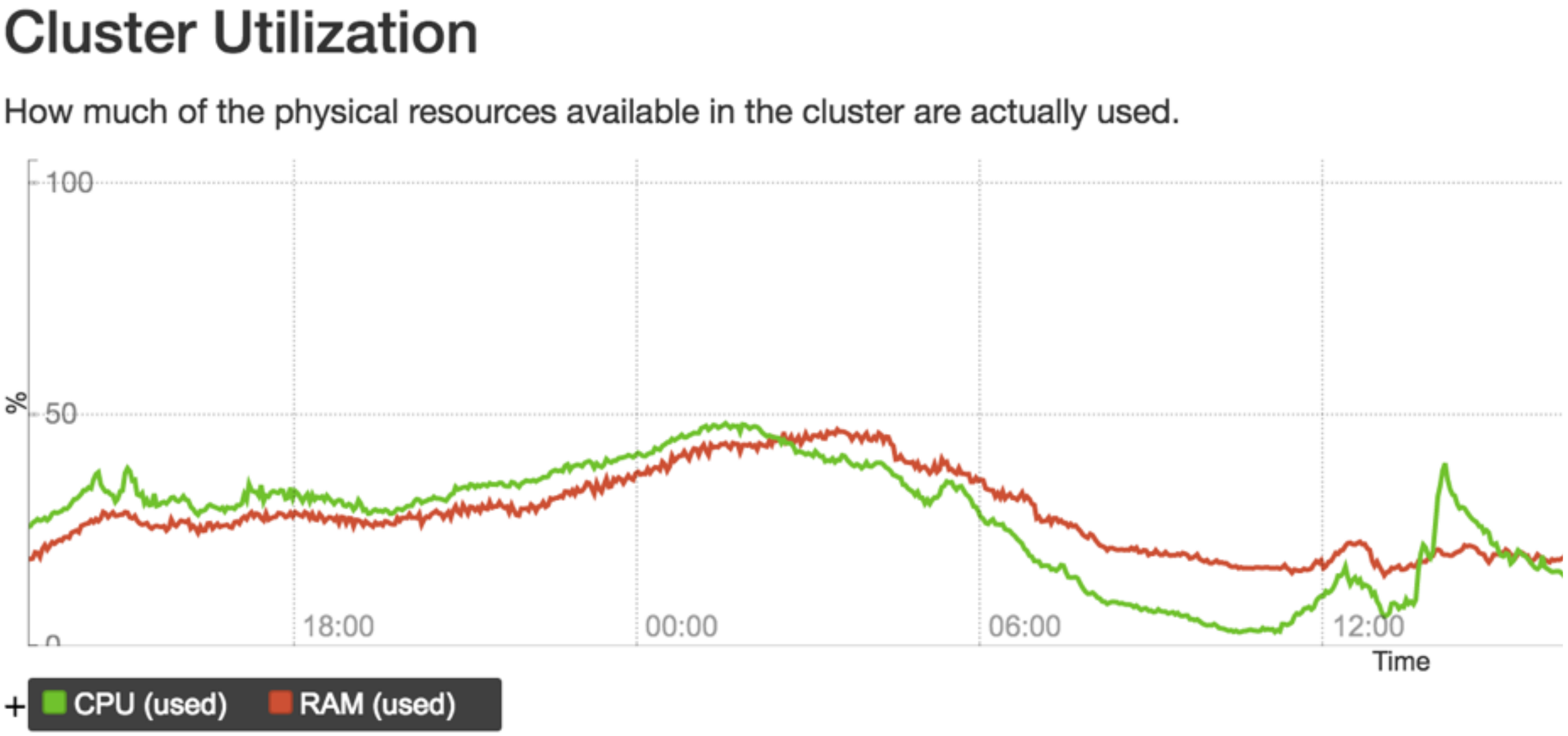
User

- long-running services
- cron jobs & adhoc jobs
- rolling job updates, with automatic rollback
- service announcement in ZooKeeper
- scheduling constraints
- Docker/Appc support
- self-service UIs

Operator

- high-availability
- maintenance primitives
- resource quotas and preemption
- instrumented for monitoring and debugging
- oversubscription

Oversubscription



https://github.com/blue-yonder/mesos-threshold-oversubscription

Executive Summary

In this talk, we have seen:

- Aurora & Mesos provide excellent support for heterogenous workloads.
- They can even be used by data scientists to ship machine learning models into production.
- All without major headache for your operations team.

Thank you!

Stephan Erb
2016.11.15

serb@apache.org
@ErbStephan

blueyonder