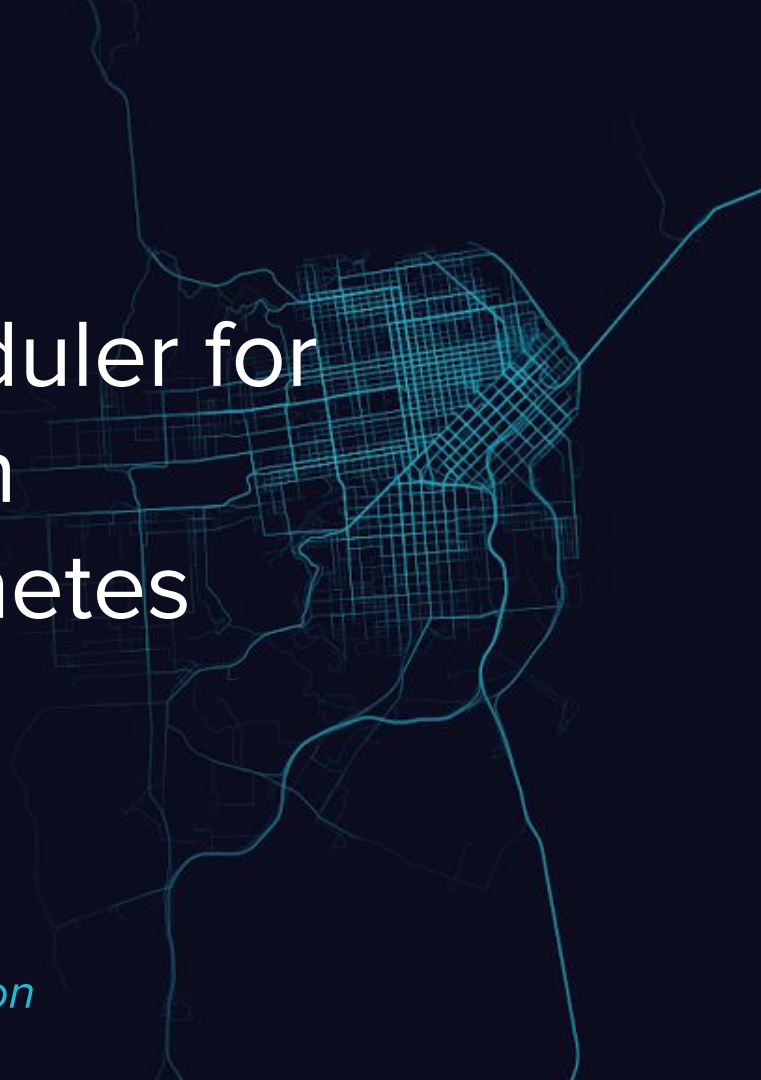
Uber

Peloton: A Unified Scheduler for Web-scale Workloads on Apache Mesos & Kubernetes

Min Cai, Uber

Nitin Bahadur, Uber

Igniting opportunity by setting the world in motion



Uber

10+ billion trips

15M+ trips per day

6 continents, 65 countries and 600+ cities

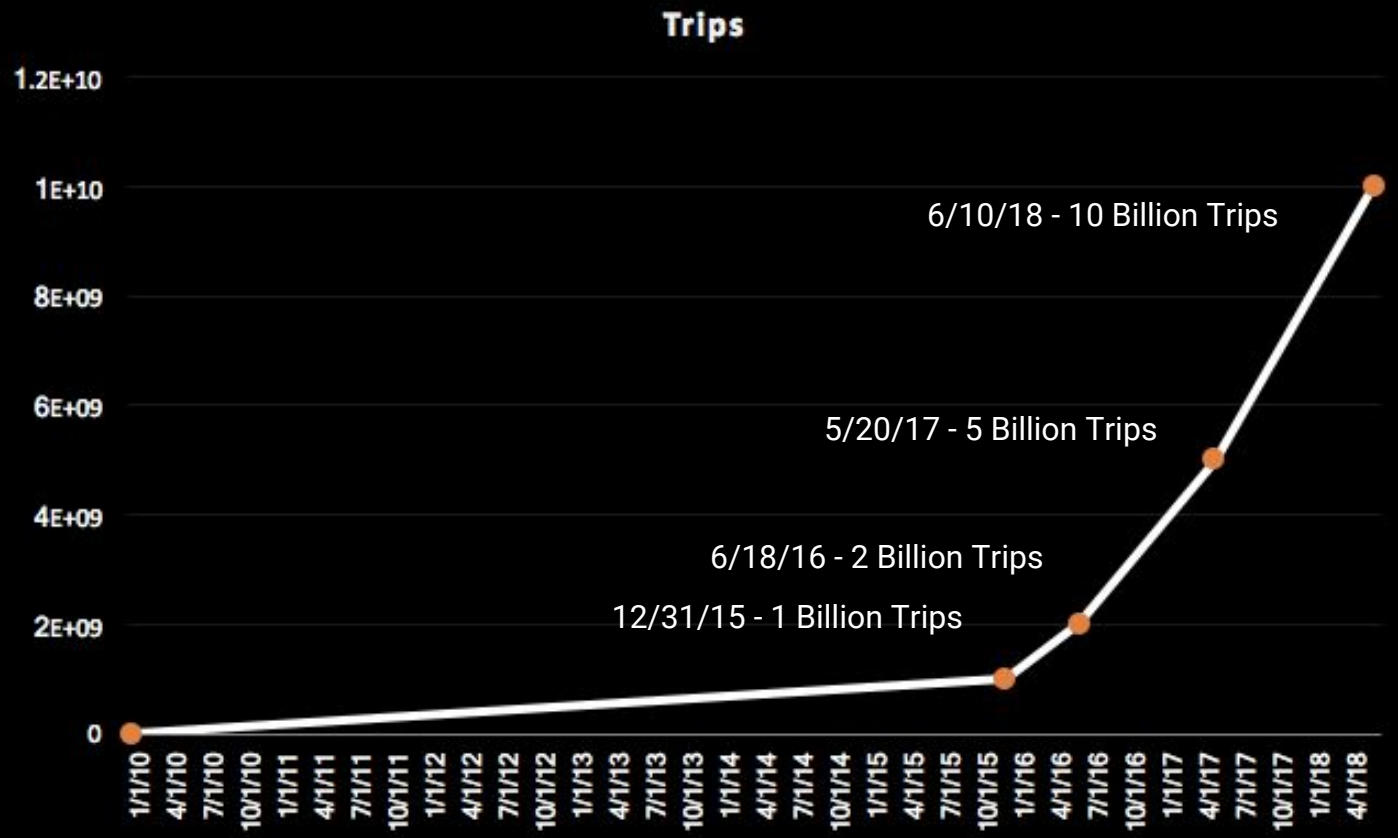
75M active monthly users

3M+ active drivers

16,000+ employees worldwide

3000+ developers worldwide

Business


Trips
1.2E+10
1E+10
8E+09
6E+09
4E+09
2E+09
0
6/10/18 - 10 Billion Trips
5/20/17 - 5 Billion Trips
6/18/16 - 2 Billion Trips
12/31/15 - 1 Billion Trips
1/1/10
4/1/10
7/1/10
10/1/10
1/1/11
4/1/11
7/1/11
10/1/11
1/1/12
4/1/12
7/1/12
10/1/12
1/1/13
4/1/13
7/1/13
10/1/13
1/1/14
4/1/14
7/1/14
10/1/14
1/1/15
4/1/15
7/1/15
10/1/15
1/1/16
4/1/16
7/1/16
10/1/16
1/1/17
4/1/17
7/1/17
10/1/17
1/1/18
4/1/18

Compute Infrastructure Scale

1000s of Microservices

1000s of Builds per day

10,000+ instances deployed per day

100K+ Service containers per cluster

~1M Batch containers per day

~1000s GPUs per cluster

25+ clusters

Uber stateless services run on Mesos Today

New Compute Cluster Use Cases

- Large scale batch jobs for autonomous vehicle use-cases
    - 100K tasks per job and millions tasks per day
- Elastic resource sharing among organizations and teams
- Co-locating mixed workloads on shared clusters
- Distributed deep learning on GPUs

Uber Cluster Workloads

Co-locate Cluster Workloads

Why

- Improve cluster utilization
- Reduce the need to buy extra capacity for big spikes like NYE
- Use DR capacity for batch jobs in All-Active setup
- Batch jobs are ideal for resource overcommit

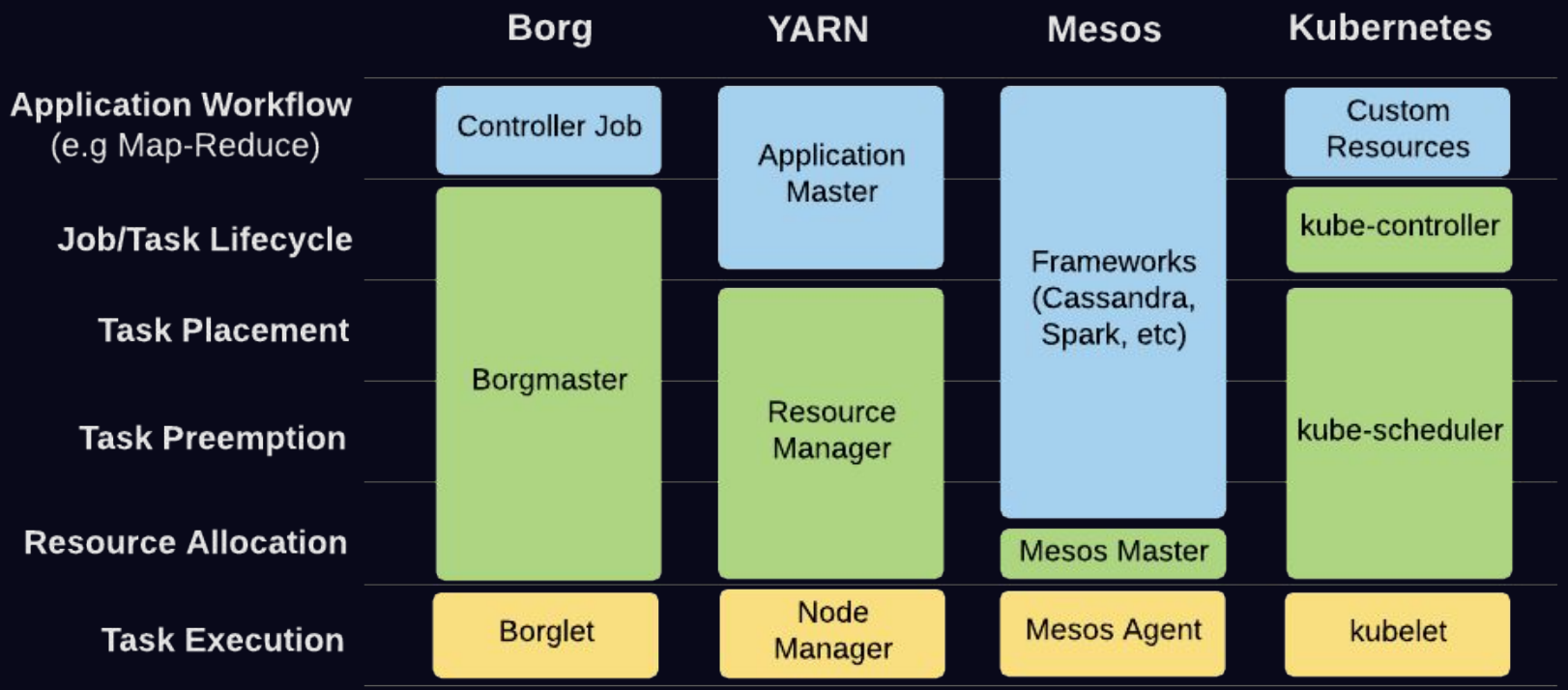
Issues

- Expensive to preempt online jobs that are latency sensitive

Existing Cluster Management Solutions

Image source: https://www.doyouconvert.com/

Comparison of Cluster Manager Architectures



Why Not Use Other Existing Schedulers

- **Borg** is not an open source solution

- **YARN** is a batch scheduler for Hadoop with no or very limited support for stateless, stateful, and daemon jobs.

- **Kubernetes**
  - It hasn’t been able to scale to the large clusters that Uber requires, i.e. 10,000 plus. Federation is still in infancy.
  - Elastic resource sharing is not supported.
  - Not ideal for batch workloads, due to the high-churn nature of batch jobs.

Introducing Peloton

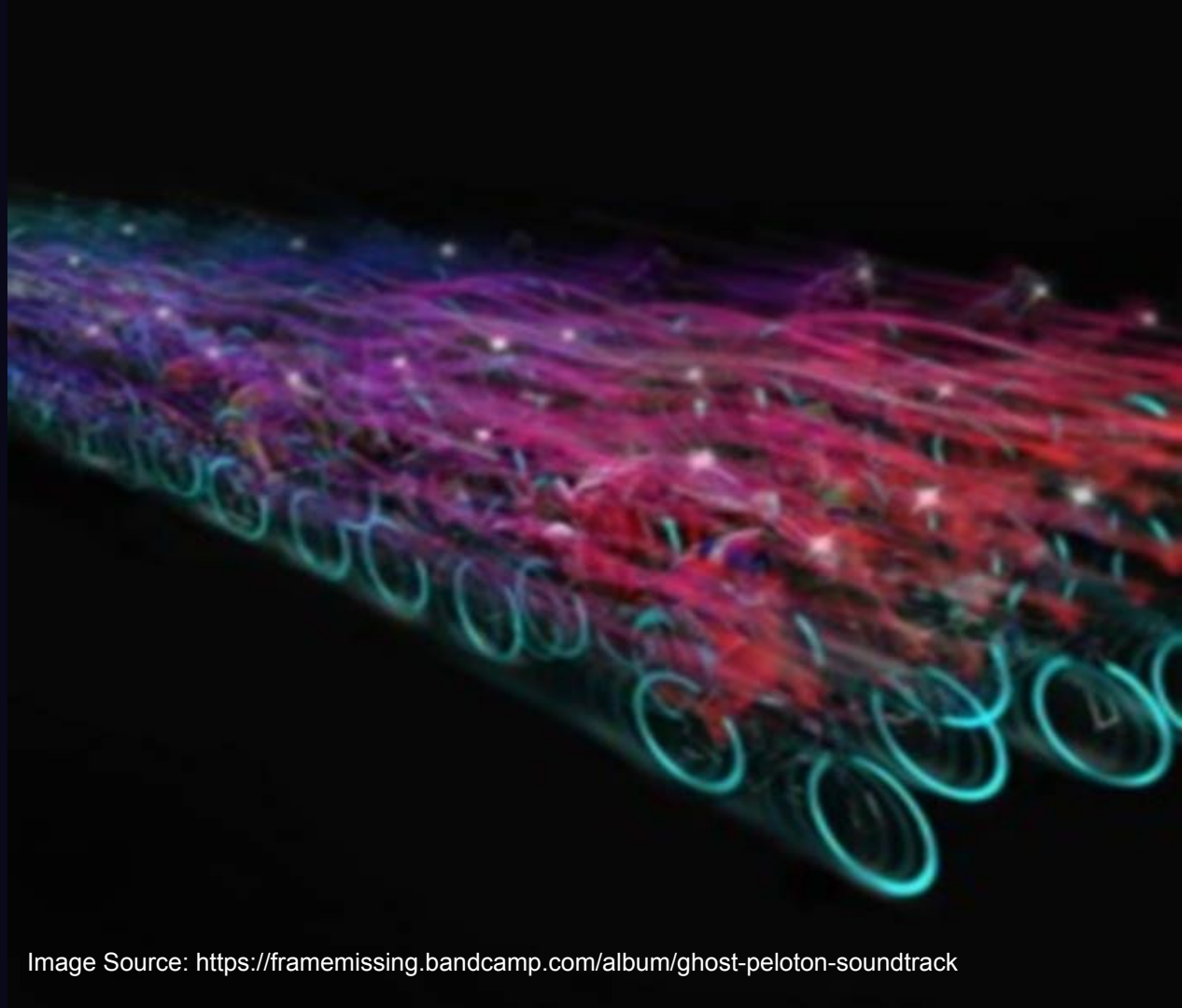
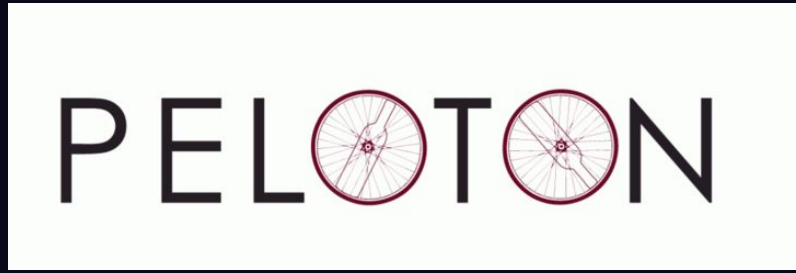


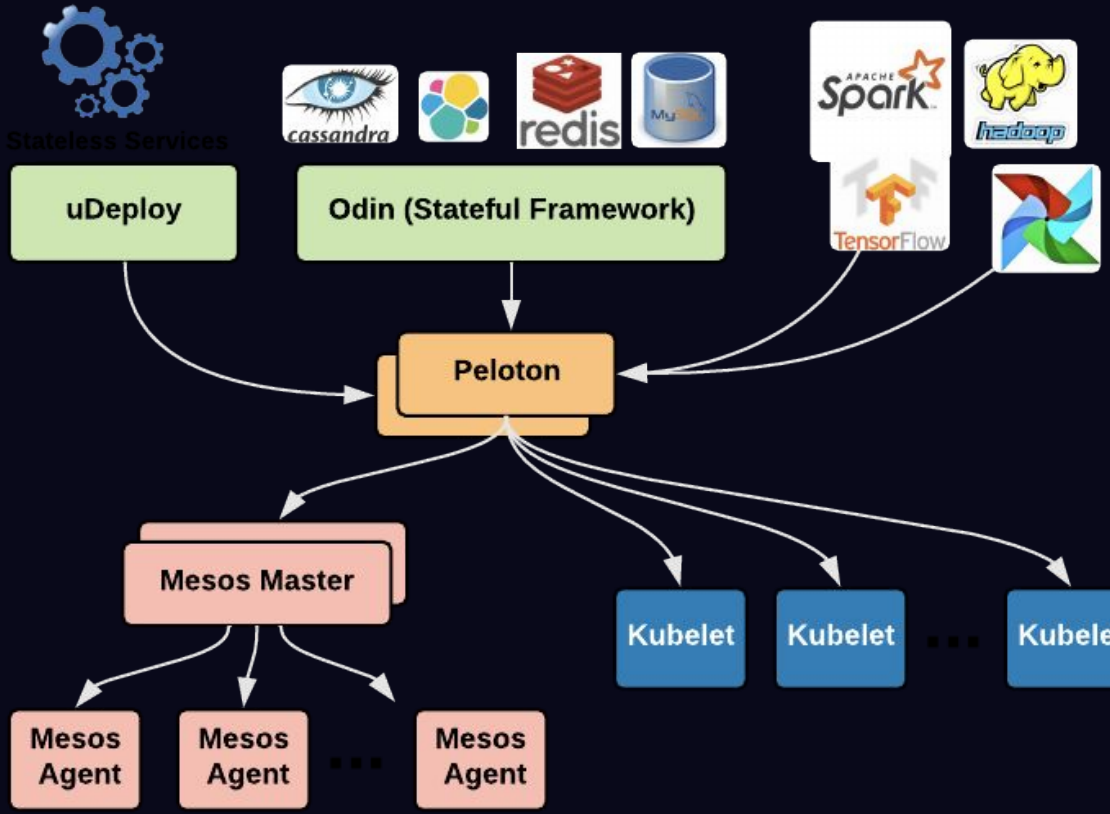
Image Source: https://framemissing.bandcamp.com/album/ghost-peloton-soundtrack

What is Peloton?

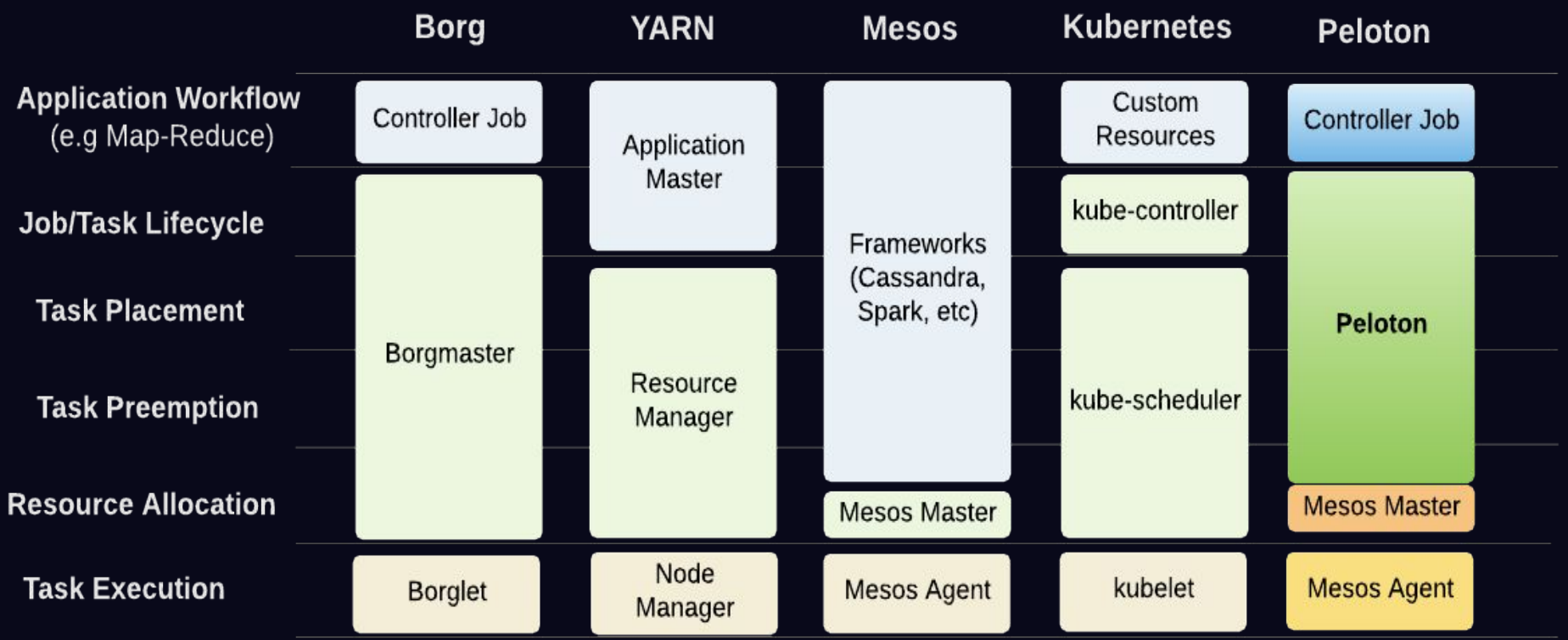
- *Unified Resource Scheduler* for co-locating mixed workload on compute clusters @ Uber
- Integrates with Spark, TensorFlow, YARN, uDeploy, etc.
- Can be run on-premise or in the Cloud



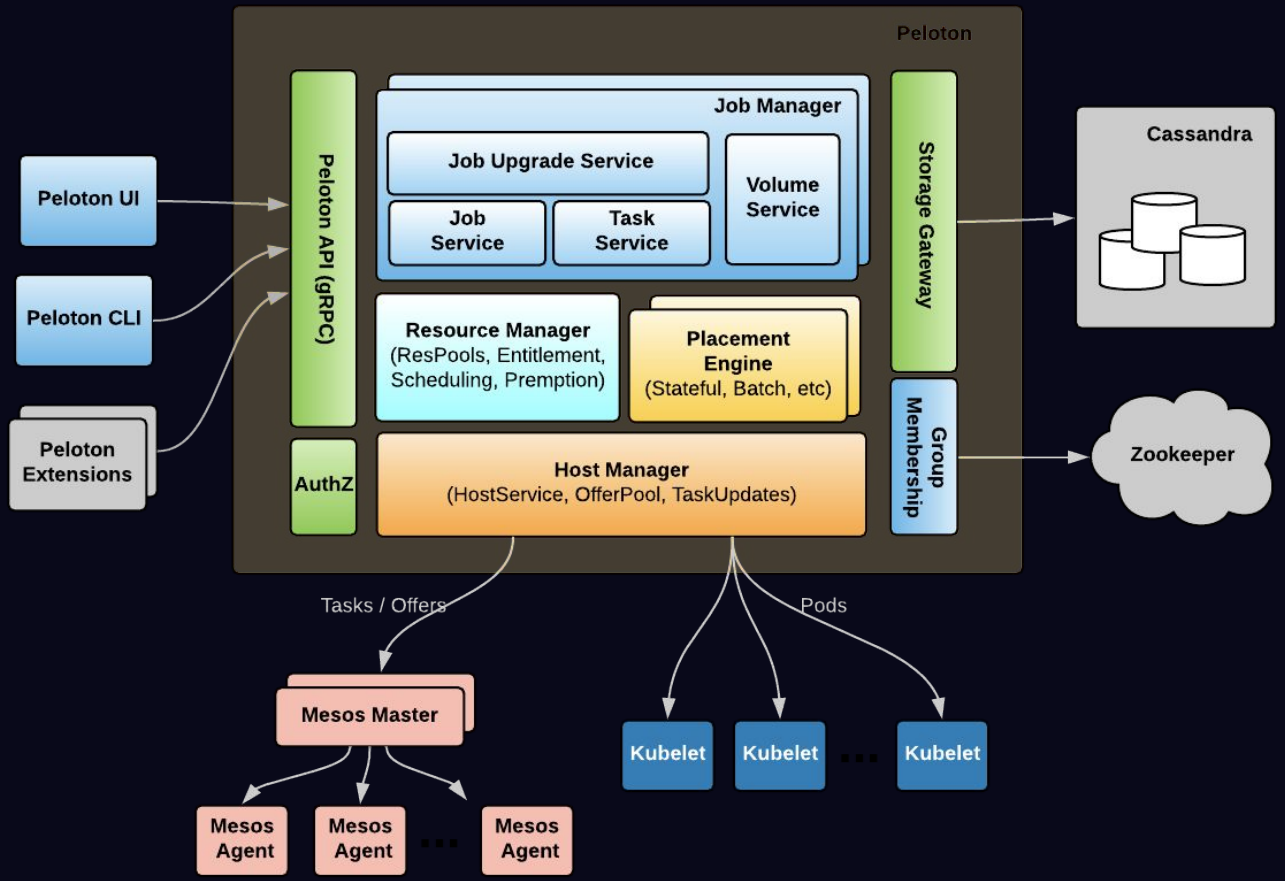
Peloton Overview



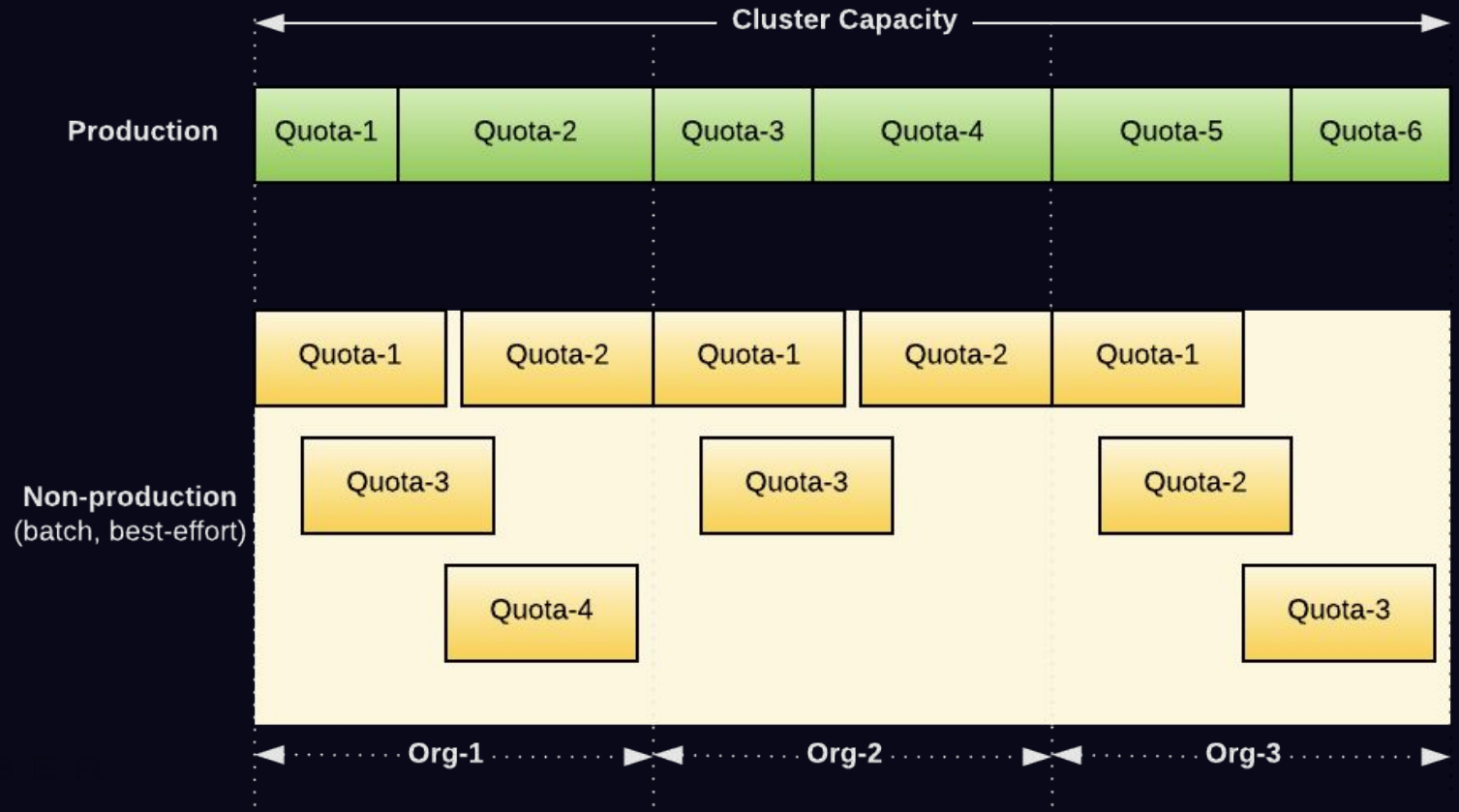
Comparison of Cluster Scheduler Architectures



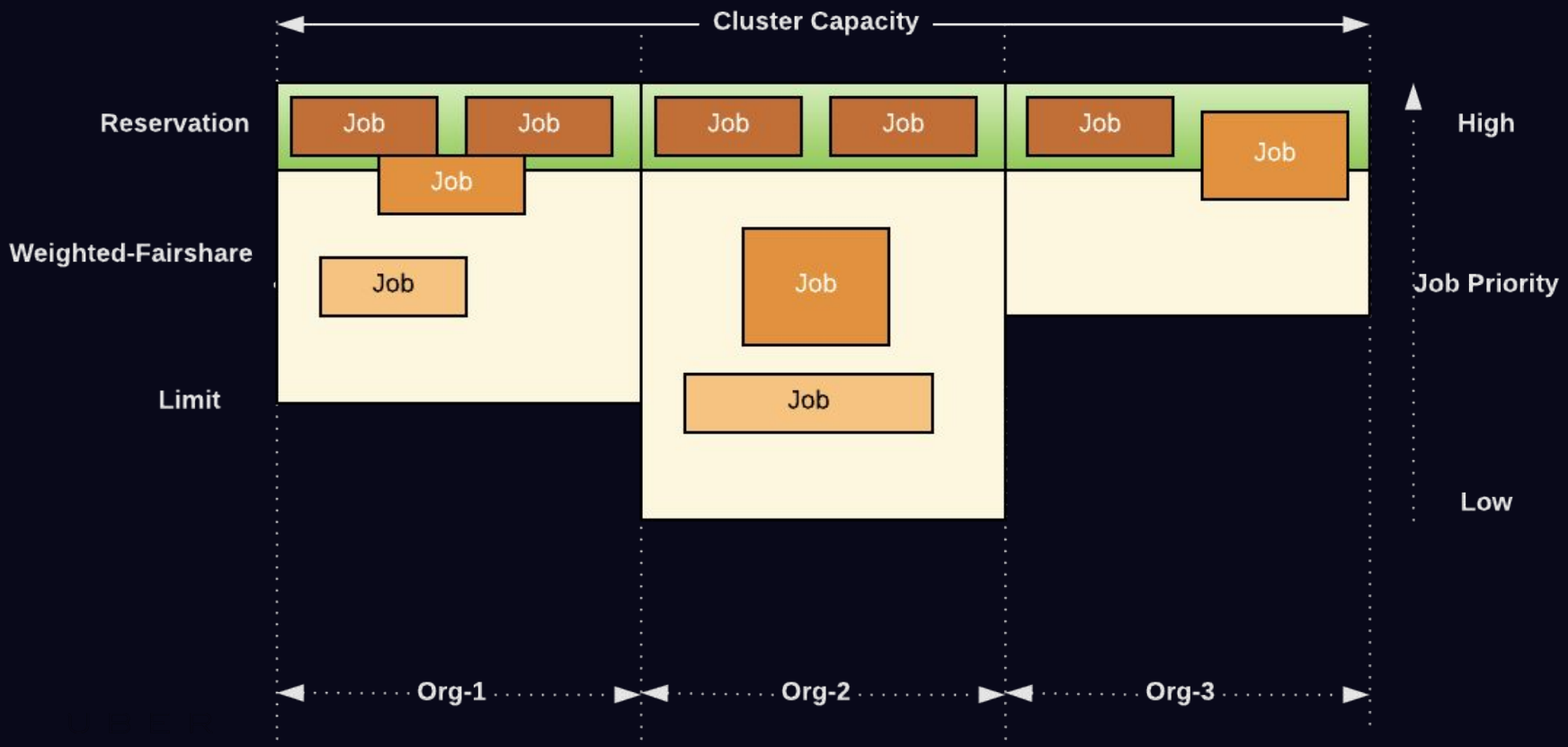
Peloton Architecture



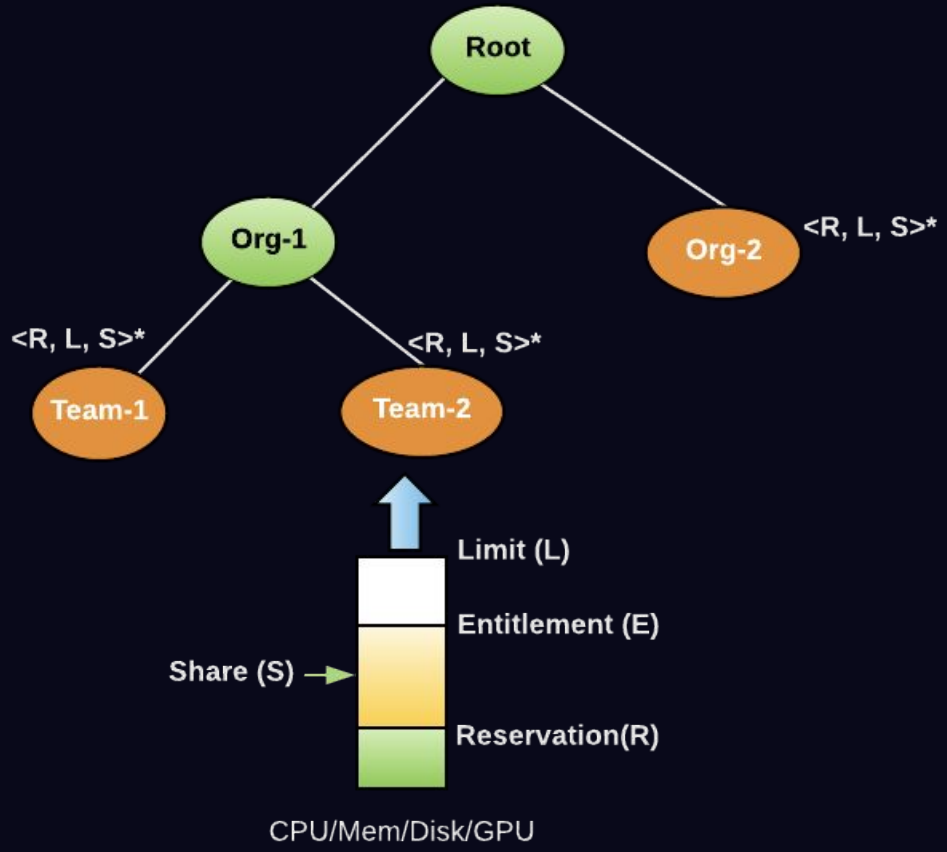
Priority-based Quota (Borg Model)



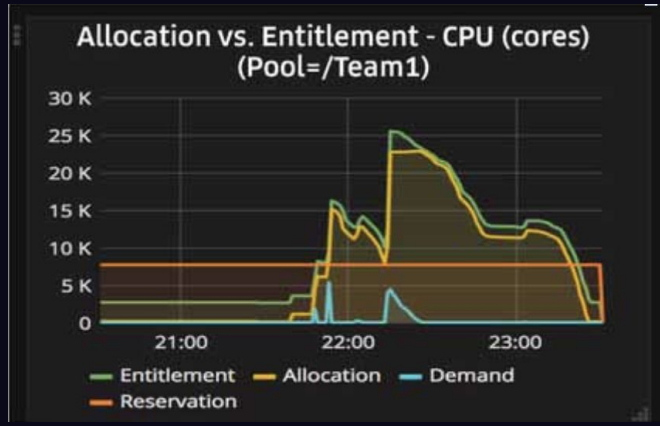
Hierarchical Max-Min Fairness

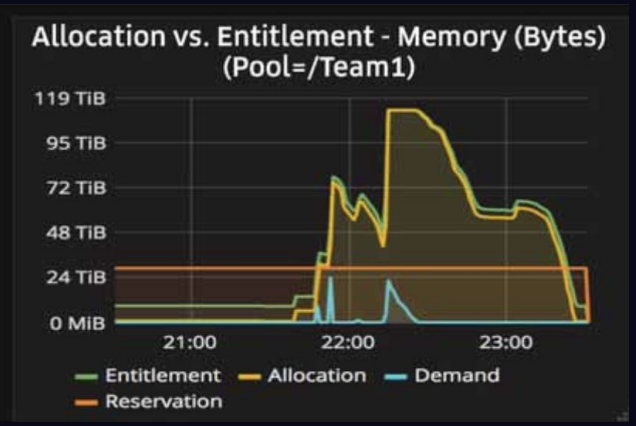


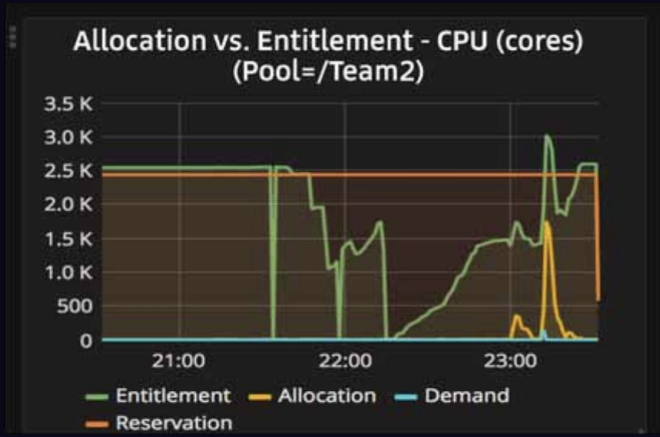
Hierarchical Resource Pools

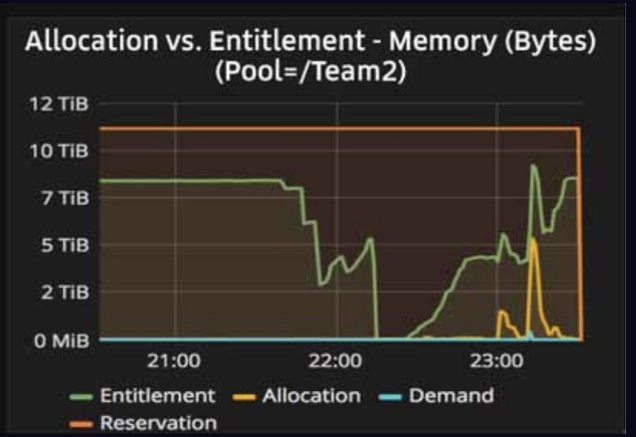


Resource Pool Elasticity

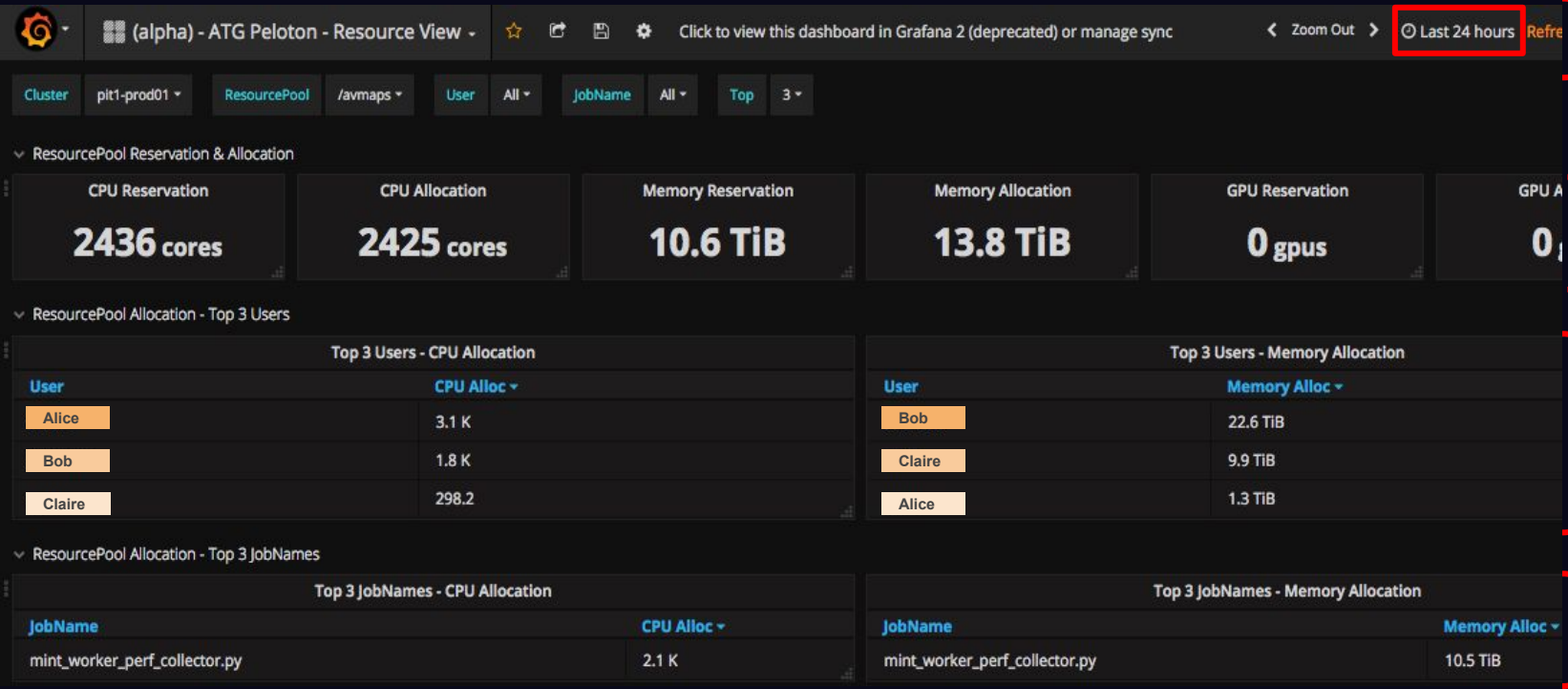








Resource Mgmt Features



For the last 24h

Reservation & Avg Allocation

Top 3 Users in CPU & Mem Allocation

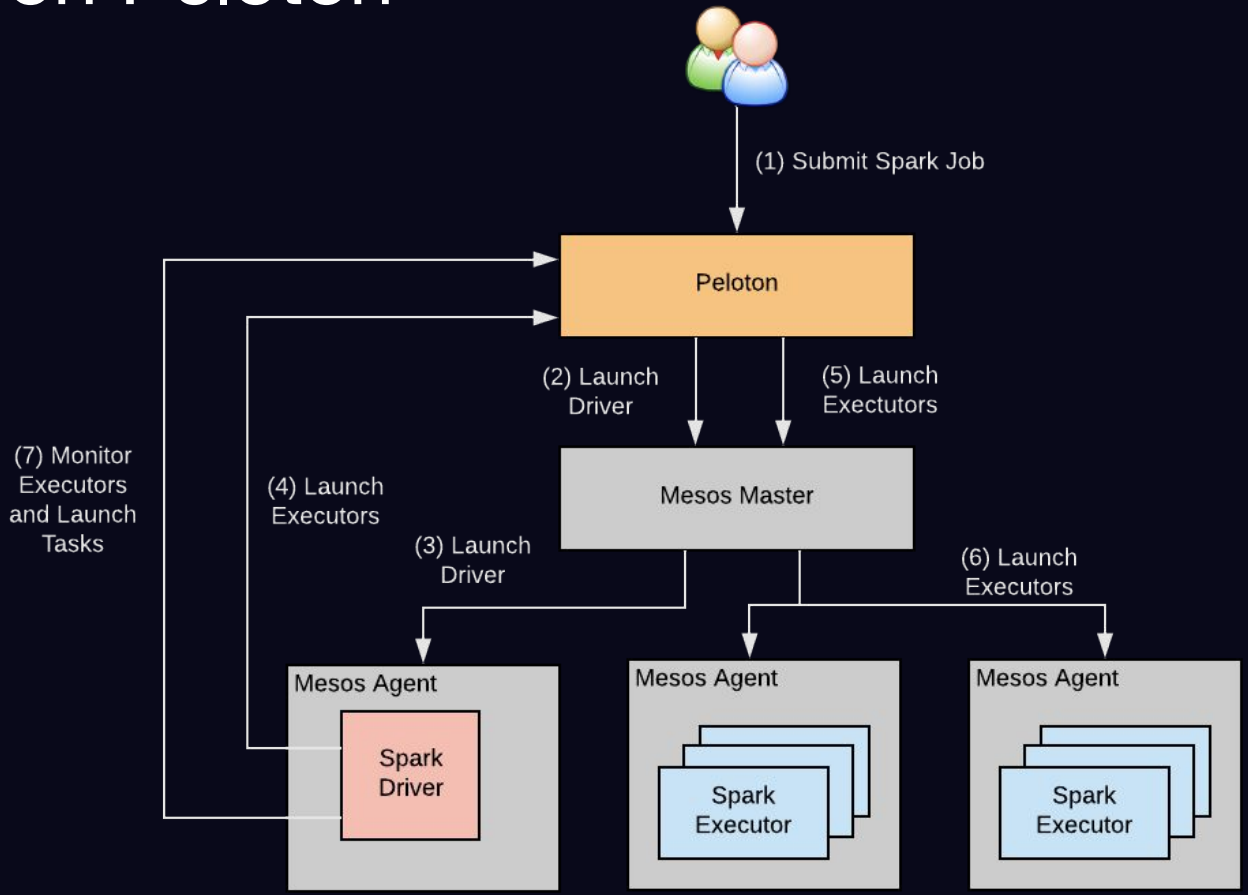
Top 3 JobNames In CPU & Mem Allocation

Peloton Workloads @Uber

Batch Jobs on Peloton

- Running Batch Jobs in multiple datacenters
  - Spark
  - Distributed TensorFlow
  - Large Scale Maps workloads
  - Large scale Autonomous batch workloads
  - Feature parity with YARN for Uber workloads
- Scale
  - **8K**+ Hosts, ~**2.5K+** GPUs
  - **3M+** jobs monthly, **36M+** containers monthly

Spark on Peloton



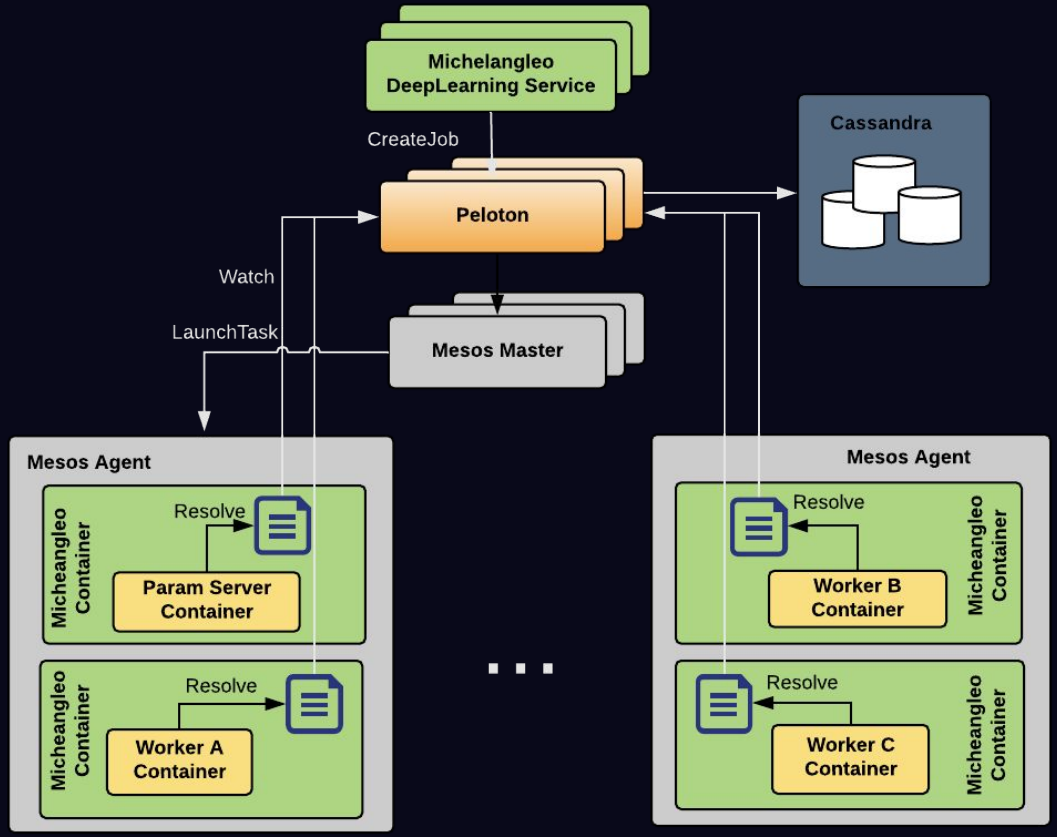
GPUs & Deep Learning

- Self-Driving Vehicles
- Trip Forecasting
- Fraud Detection
- More ...

Distributed TensorFlow Challenges

- Elastic GPU Resource Management
- Locality and Network-aware Placement
- Gang Scheduling
- Task Discovery
- Failure Handling

Distributed Tensorflow on Peloton



Stateless Services

1000s of microservices, growing day by day

- Over-allocated & under-utilized
- Resource over-commit & pre-emption in-use

Team is actively working on migrating services off Apache Aurora to Peloton.

Stateful services

Large scale Storage & Data applications

- Currently running on dedicated bare metal clusters

Peloton target => 2020 H1

Kubernetes & Peloton

Best of both worlds?

Kubernetes

- Widely adopted container orchestration system

Peloton

- Intelligent scheduler built for web-scale workloads

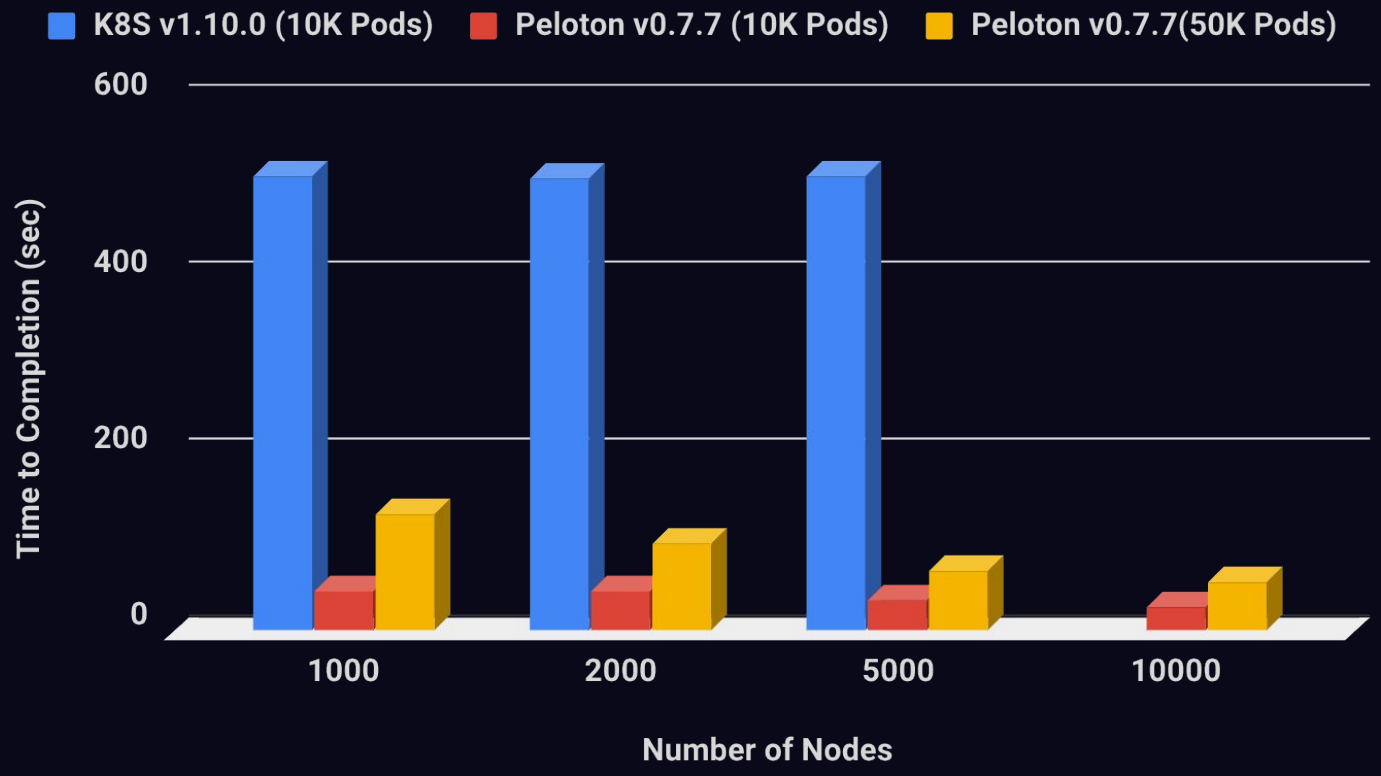
Why consider Kubernetes?

- Lots of features and extensions for mixed workloads
  - Pod, Deployment, StatefulSet, Job, DaemonSet, etc
- Growing community and ecosystem support
- More adoption and native integration from many open source projects
  - E.g. Spark, Flink, Kafka, Tensorflow etc
- Cloud native support in AWS, GCP, and Azure as managed clusters
- Fill the gap for features unavailable in today's Uber Compute offerings
  - StatefulSet
  - Auto-scaling
- Feasible extension model that allows other Uber teams such as Software Networking, Storage, Data, and Security teams to build extensions.

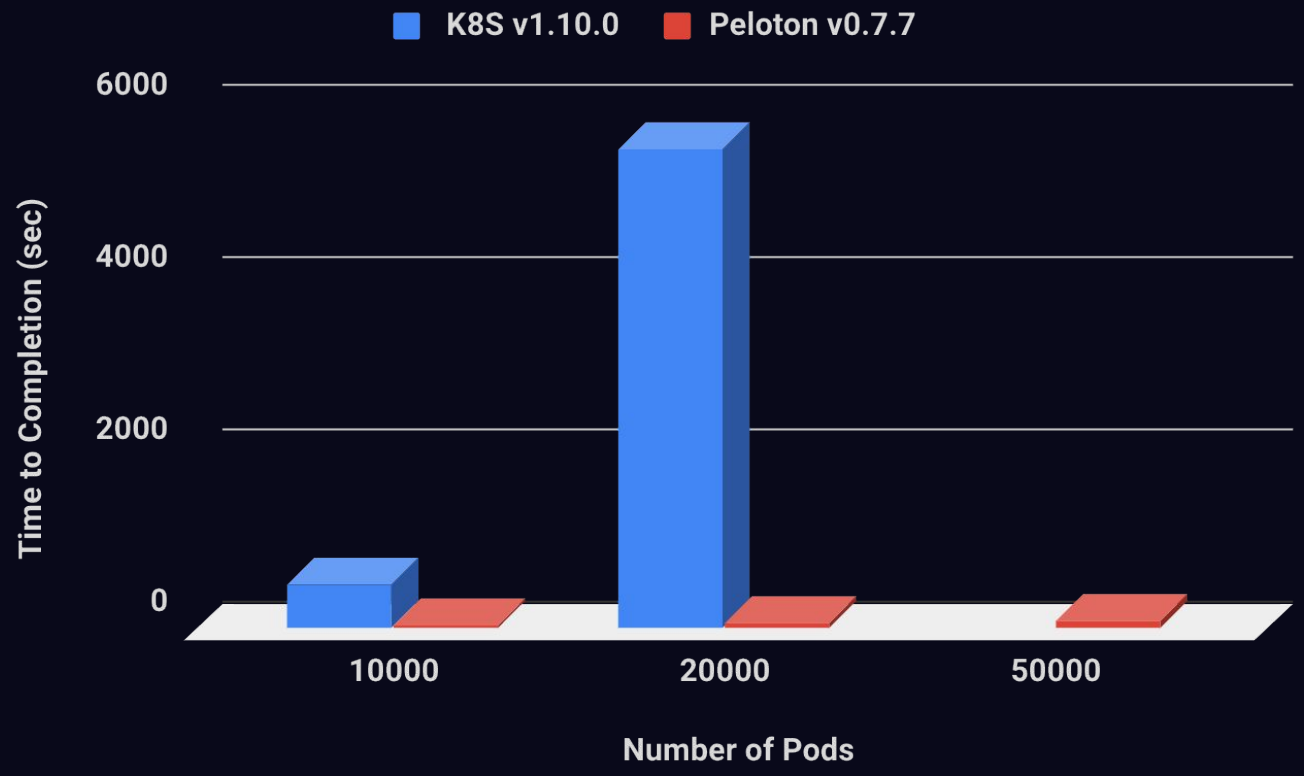
Benchmarking Peloton and Kubernetes

- Running Kubernetes and Peloton as ***virtual clusters*** on top of Peloton
- Scale the cluster sizes from 1K to 10K nodes
- Scale the batch and stateless jobs from 10K to 100K containers
- Measure the performance for the following scenarios:
  - Total time to completion for a batch job
  - Total time to rolling upgrade a stateless job

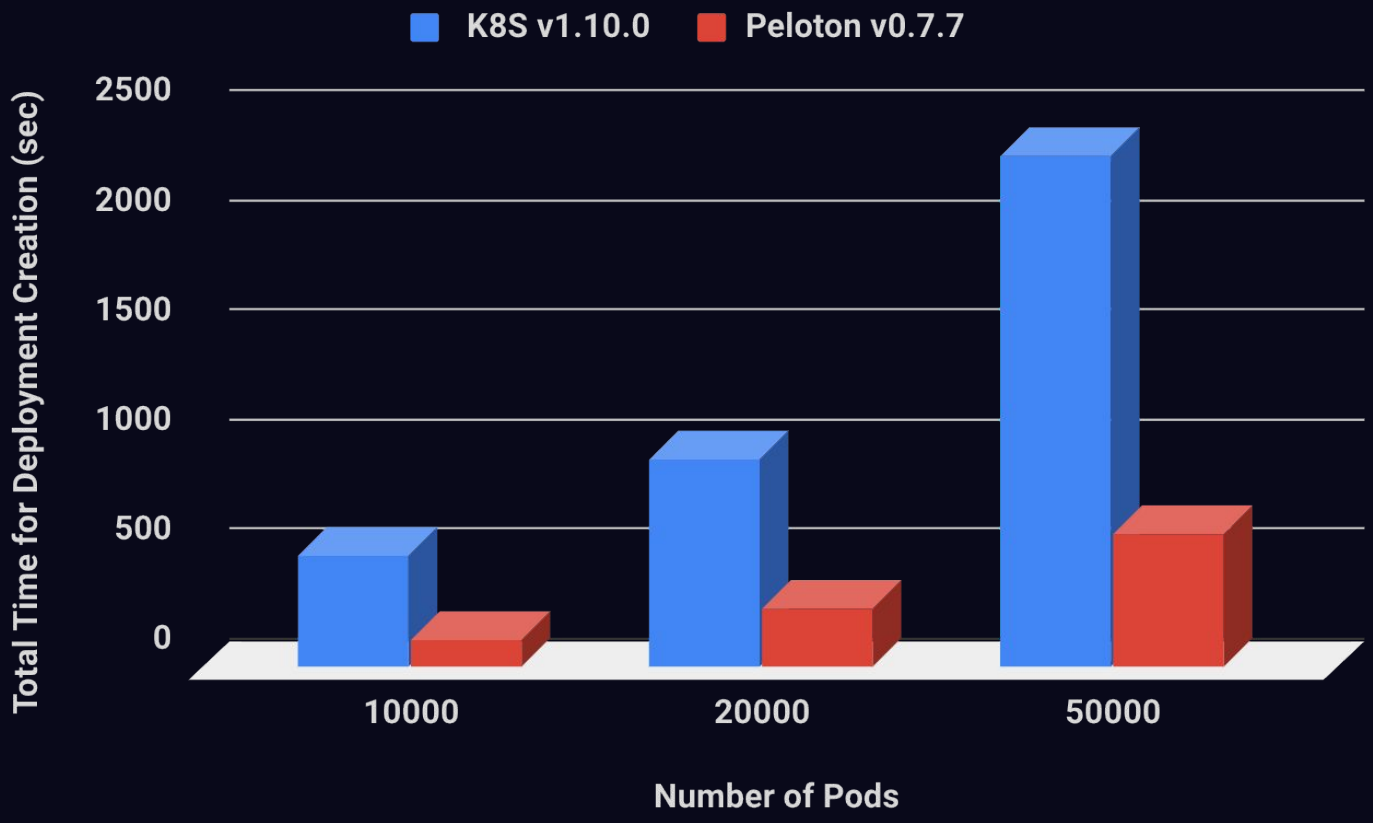
Time to Completion for Batch Jobs



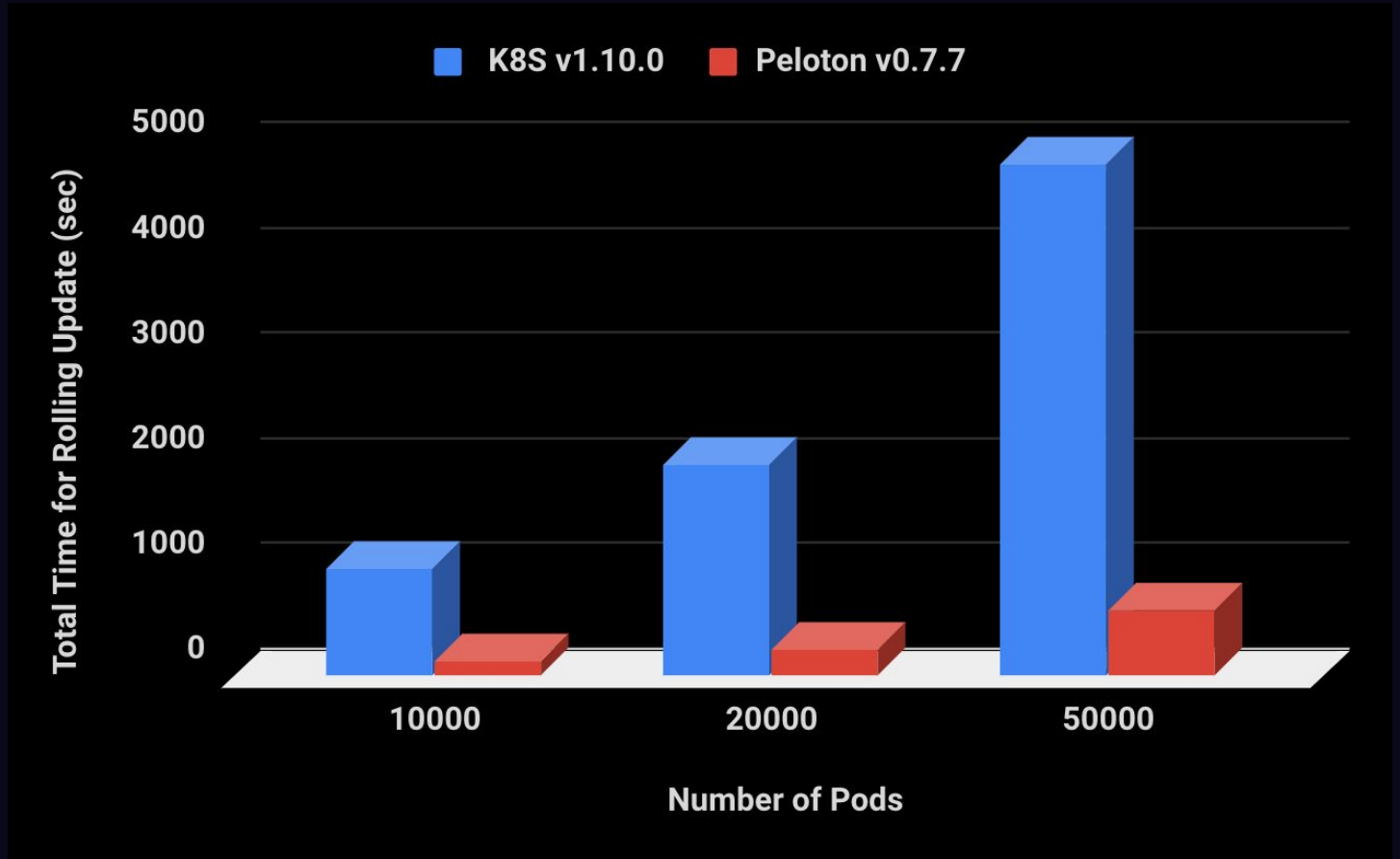
Time to Completion for Batch Jobs (2K Nodes)



Time for Deployment Creation (2K Nodes)



Time for Deployment Rolling Update (2K Nodes)



Peloton & Kubernetes Integration

Why K8s?

- Enables Uber to stay with current technology trends and leverage open-source

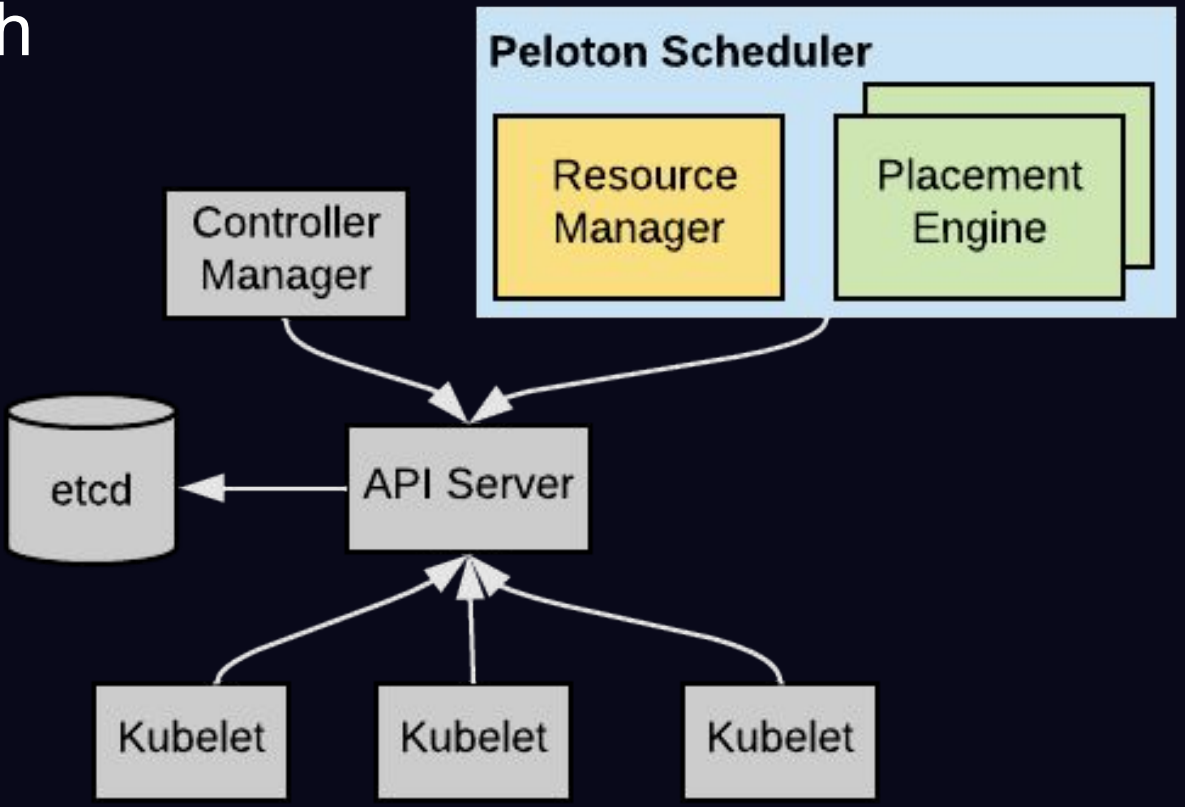
Why Peloton?

- Meets Uber specific scale & customization needs
- Provides a migration path from Mesos to Kubernetes without impacting Uber workloads

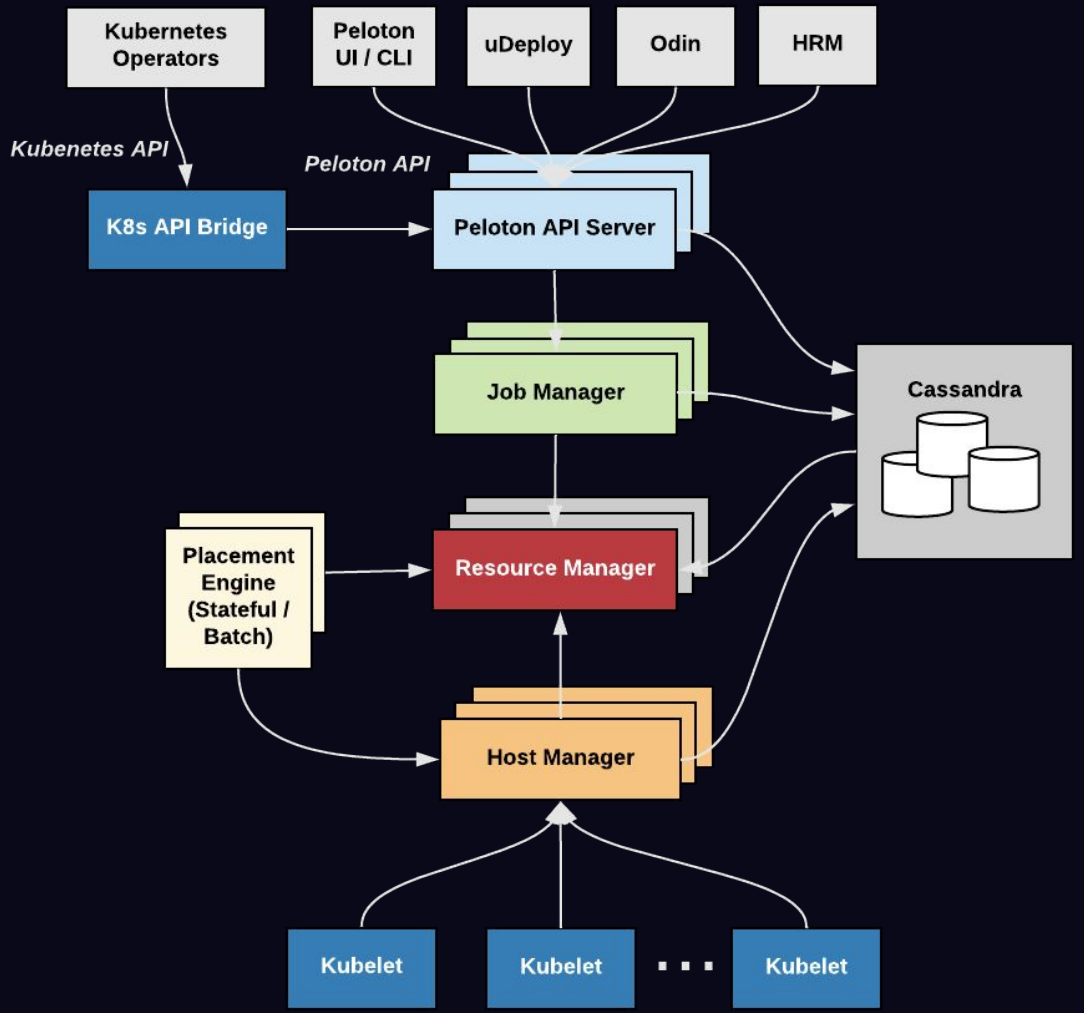
Icing on the cake

- Enables other large Mesos-based companies a way to transparently migrate from Mesos to K8s

Kubernetes with Custom Scheduler



Kubernetize Peloton



Summary

- Peloton has been deployed in production at Uber for over an year
- It’s designed from day-1 to run alongside any container orchestration system
- Engineering blog - eng.uber.com/peloton