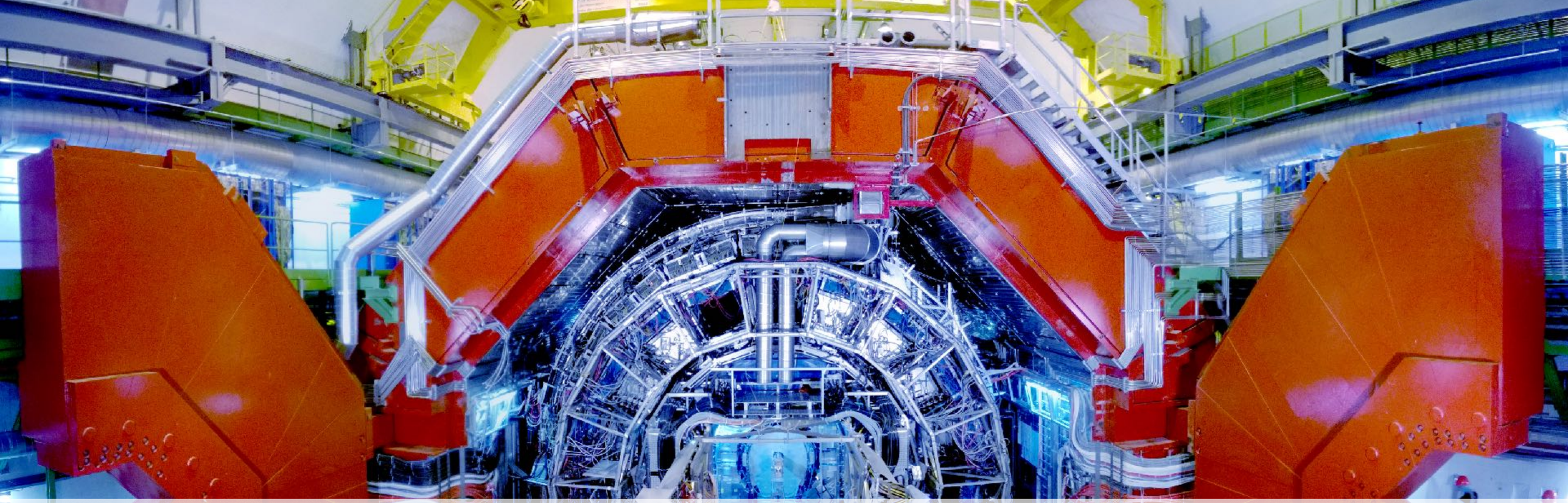


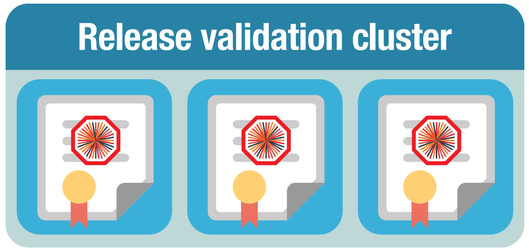
Experiences with the ALICE Mesos infrastructure

Dario Berzano · Giulio Eulisse

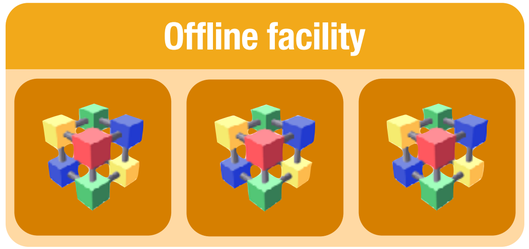
for the ALICE collaboration

Computing in High-Energy and Nuclear Physics - October 12, 2016

ALICE computing infrastructures at CERN

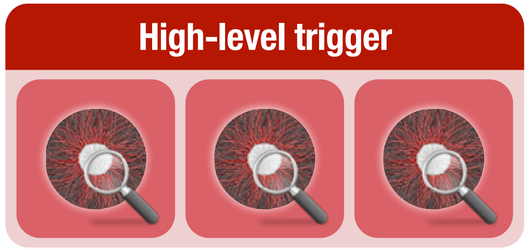


Dynamic on CERN OpenStack
(Slow-ish) IaaS approach



Powerful and not fully used
Could use it opportunistically quite continuously

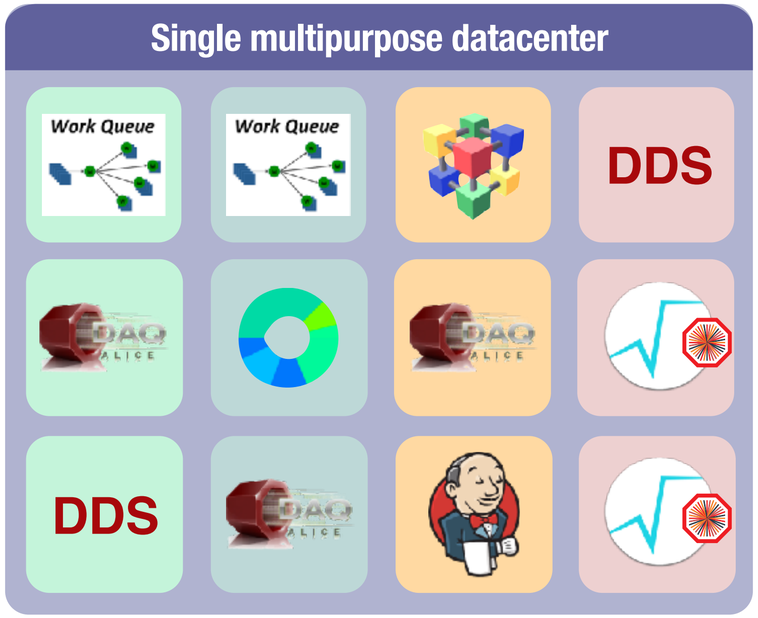
Tests for ALICE upgrade combining Offline/Online
Design phase: needs a way to be fully exploited



Made for data taking, used opportunistically as Grid node
Data taking on bare metal, Grid in dynamically deployed VMs

- So far: dynamic control using VMs and IaaS
  Dynamically start VMs by looking at batch system queues: has eased management but not always snappy
- We still have some resources underutilized
  VM deployment not always viable (see DAQ), not all use cases are batch queues
- Unify infrastructures and cover more use cases

Design of a modern datacenter



- Manage different infrastructures from a single point
- Never have unused resources
- Quickly scale horizontally - within minutes, not hours
- High availability - automatic failover and availability zones
- Handle production vs. development safely

Apache Mesos in production

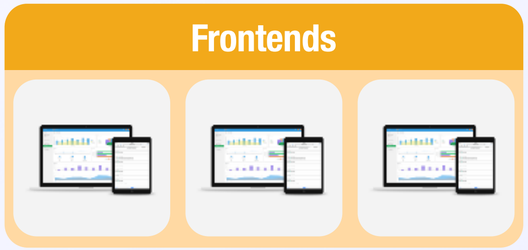
- **Apache Mesos: “program against your datacenter like it’s a single pool of resources”**
- Born in research; used by many industry major players
  Originally developed at UC Berkeley, now it’s an Apache Foundation open source project.
  Used in production at Twitter, Apple, Netflix, NASA JPL, Airbnb…
- Open source backed by many companies
  Plug-n-play solutions available from Mesosphere, Cisco, Rancher Labs…
- An operating system for large clouds
  Schedules tasks like an OS does with processes, known to work for 10k nodes

Our Mesos infrastructure setup at CERN for high availability

Pets

the important nodes

- Different availability zones
- Automatic leader selection
- Control/debug web interface
- Service discovery via DNS

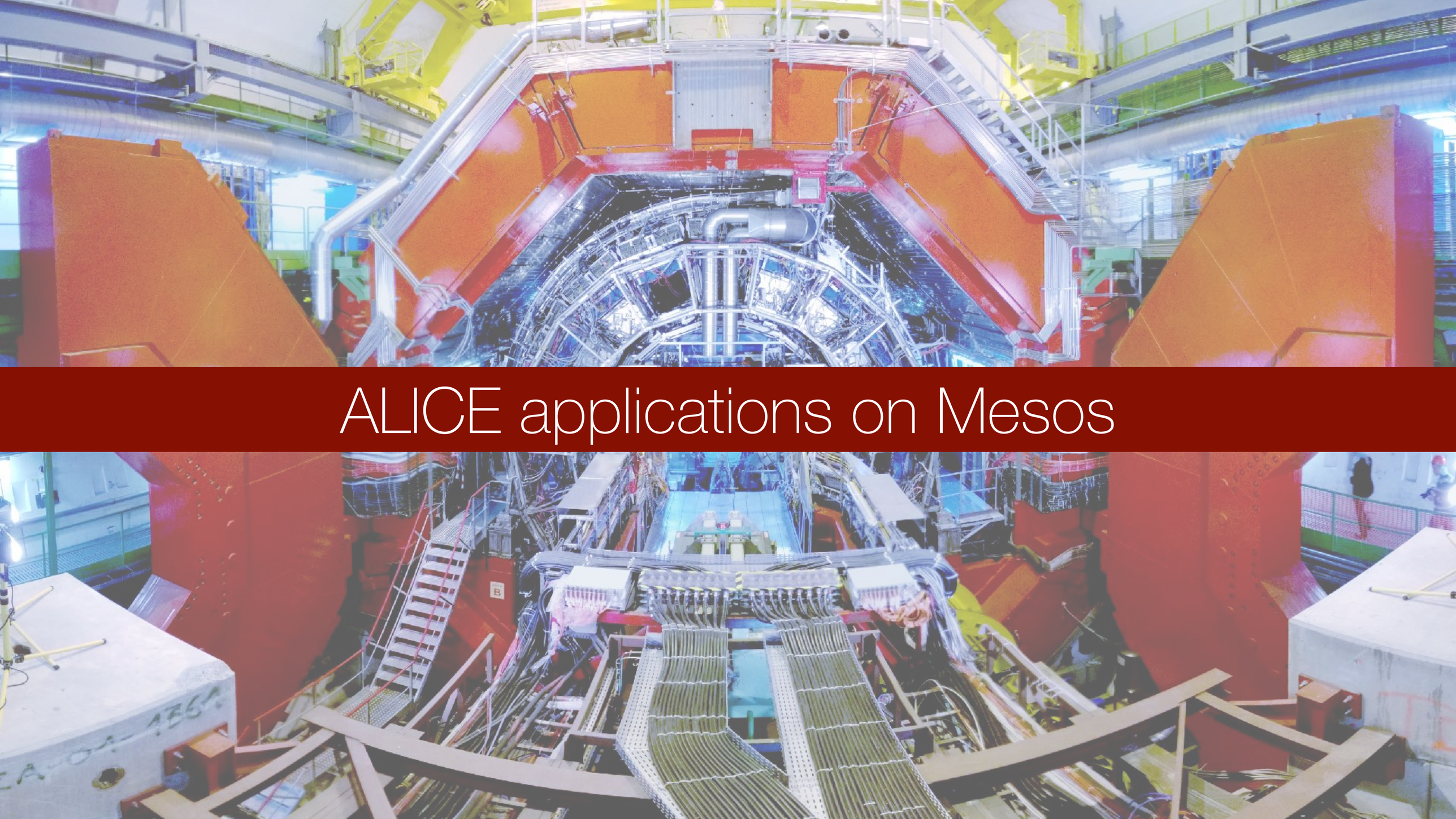


- Single sign-on
- Route to internal services
- Fixed nodes needed as internal services roam freely

Cattle

replaceable if they fail

- Run (mainly) containerized tasks on hosts
- Manage actual resources
- Update possible with no service disruption



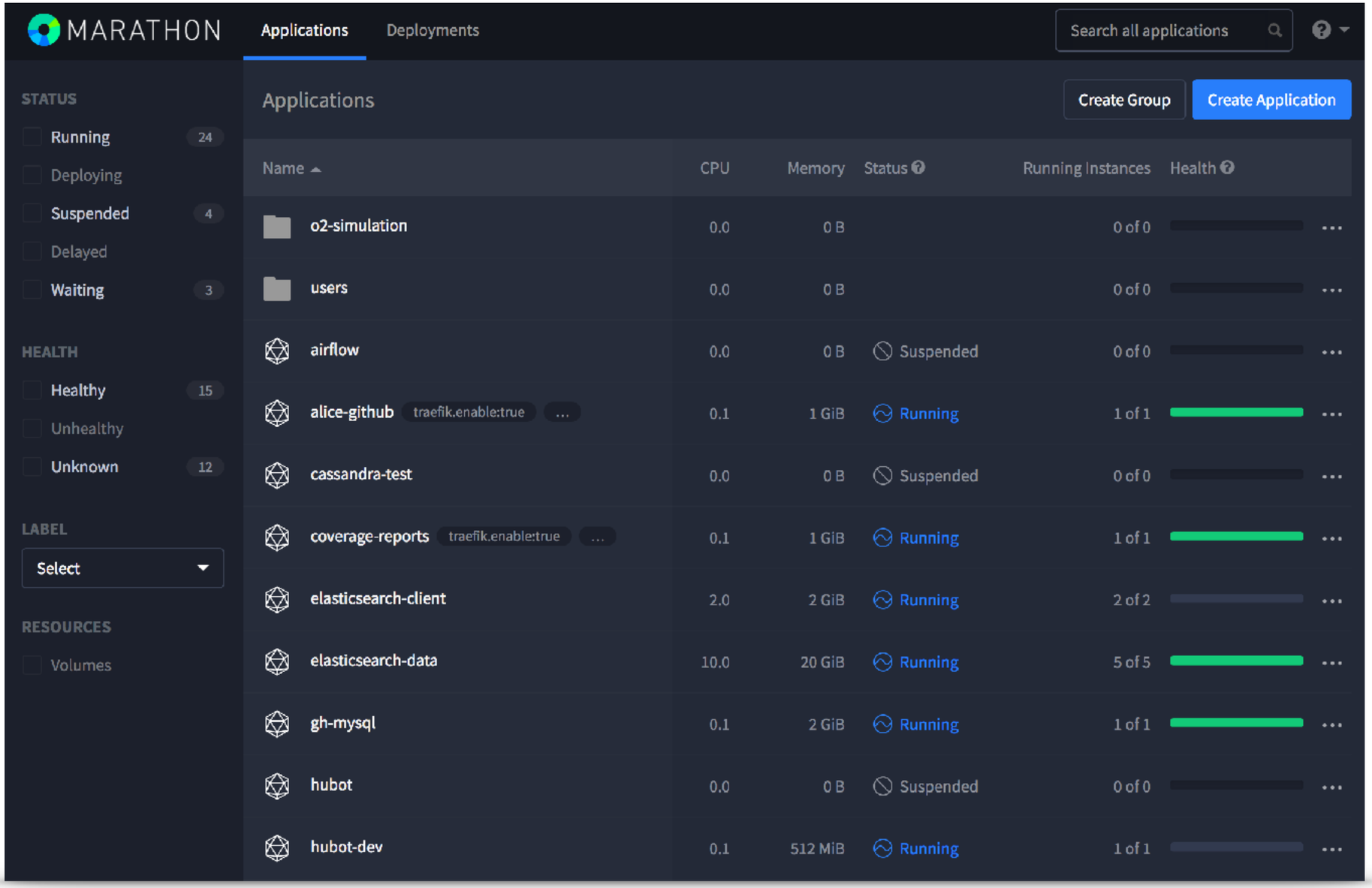
ALICE applications on Mesos

Three ways to run applications on Mesos

- **Mesos is an approach rather than a tool: brings resource knowledge to your application and guarantees fair share**
- Base approach: use an existing tool to deploy your existing applications
  Zero coding: run/scale tasks with *e.g.* Marathon, Aurora, Jenkins and get started quickly. We use Marathon extensively for many microservices (web servers, bots…)
- Write a custom tool (framework) to integrate an existing use case
  A “framework” receives resource offers from Mesos either via the HTTP interface or via C++/Python/Java bindings and decides what to do with them
- Write a fully fledged application scheduler yourself
  Schedule different tasks within your application using resource offers from Mesos. Apple does it with J.A.R.V.I.S., used to drive Siri
- All those use cases can coexist: Mesos is made to support multiple frameworks

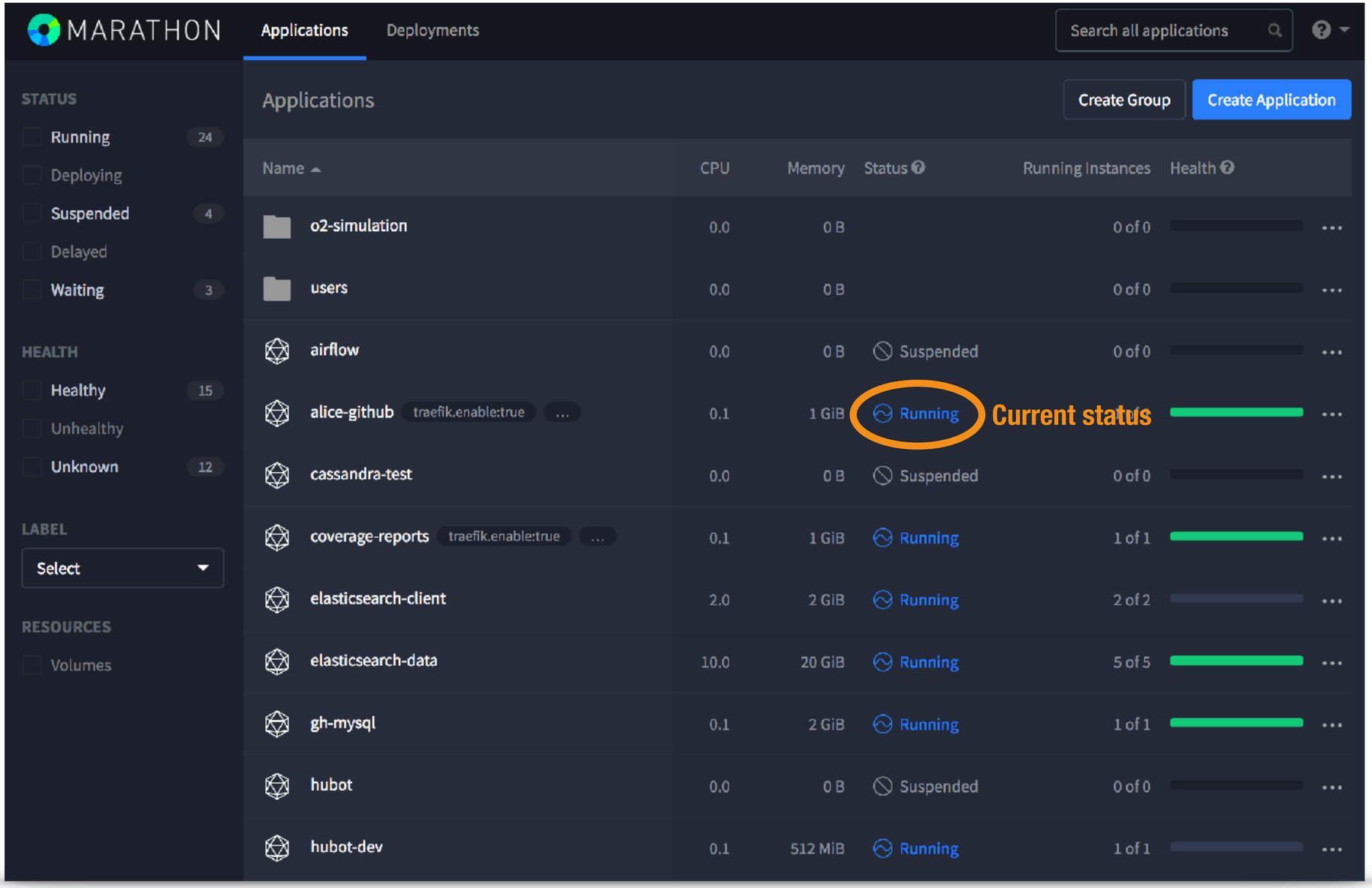
Long-running services with Marathon

- Migrated some existing services to Mesos for HA
- Existing and new services are now shipped as Docker containers
- Fixed frontend nodes dynamically mapped to internal Marathon services



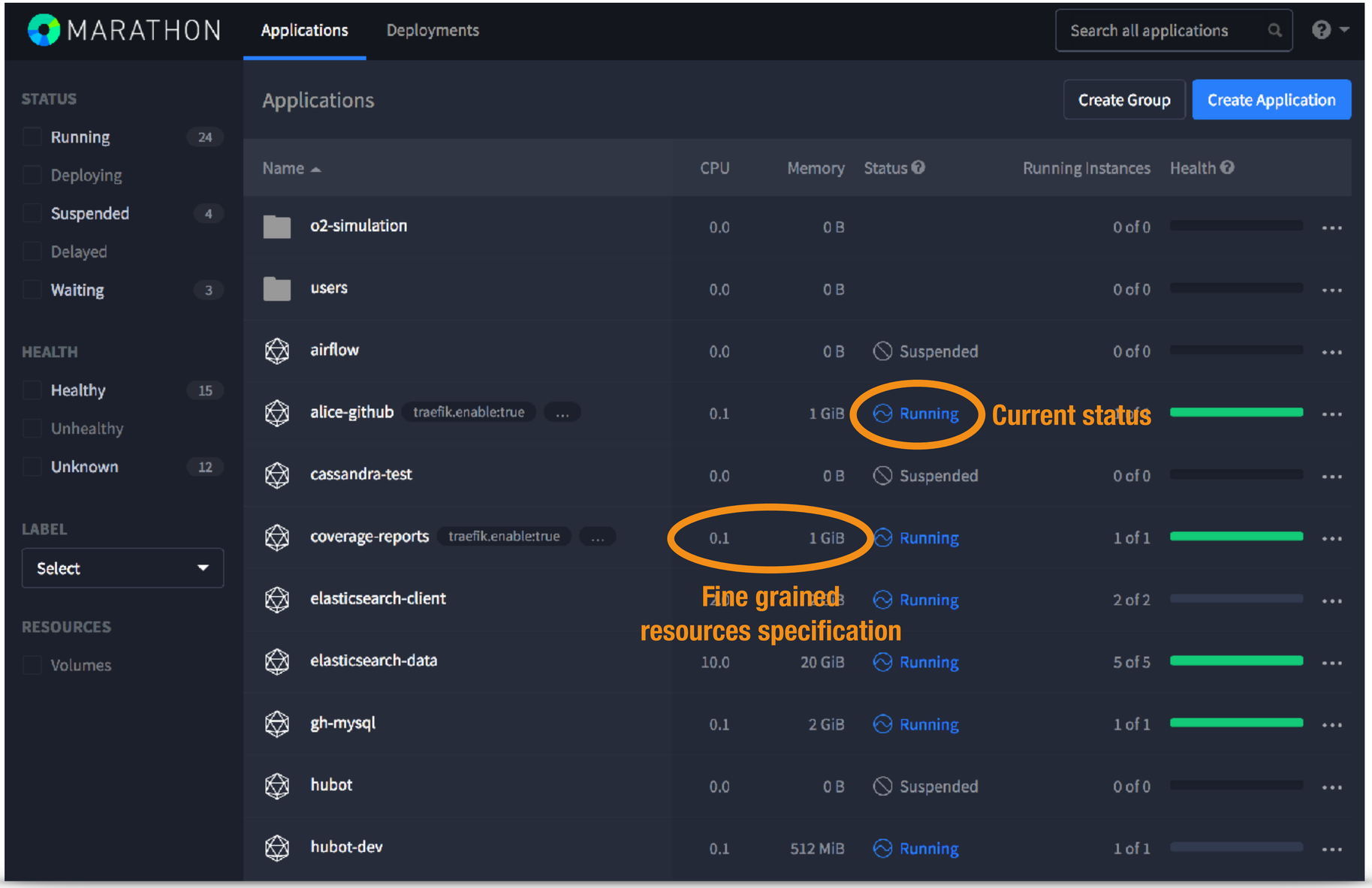
Long-running services with Marathon

- Migrated some existing services to Mesos for HA
- Existing and new services are now shipped as Docker containers
- Fixed frontend nodes dynamically mapped to internal Marathon services



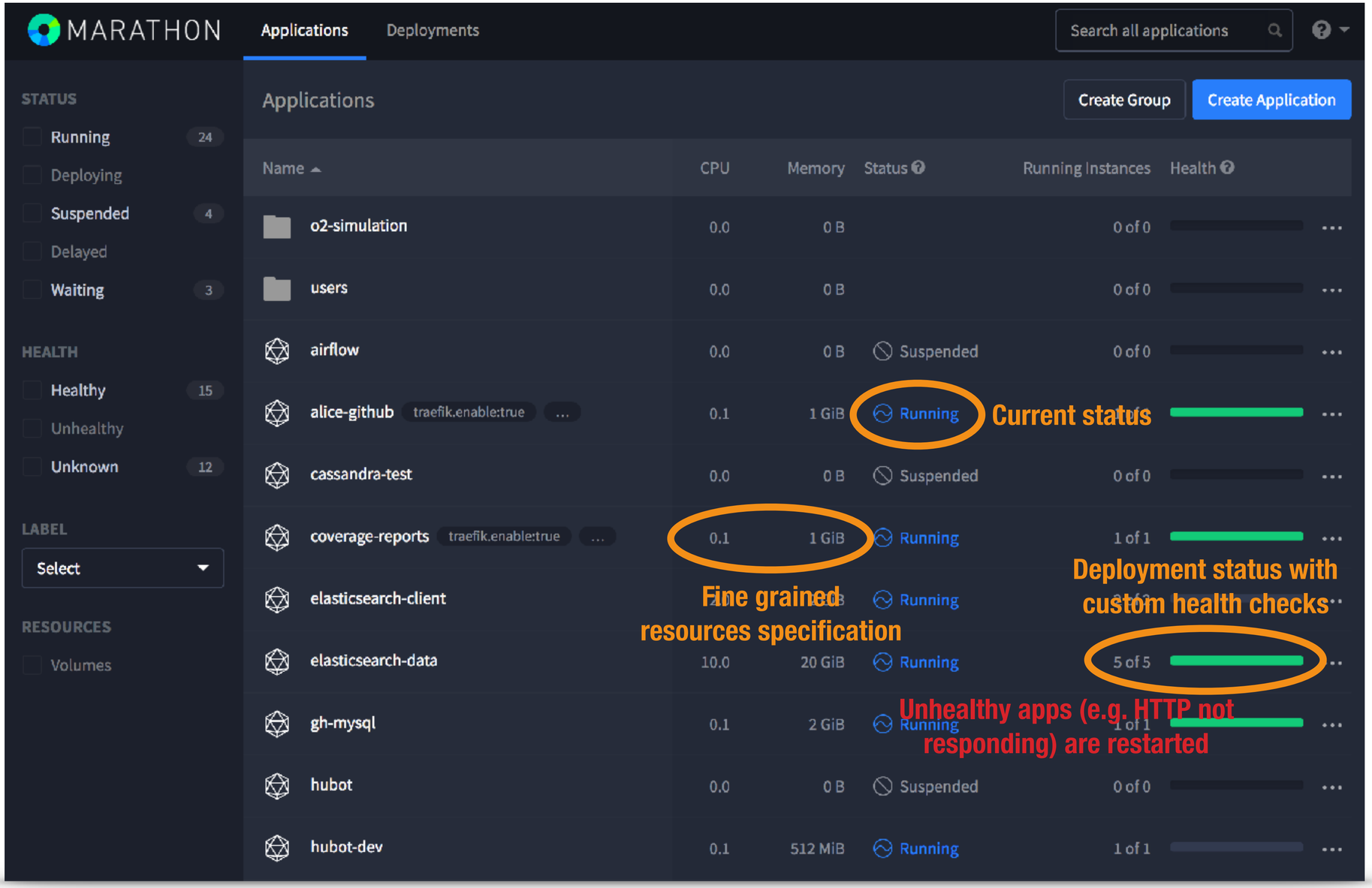
Long-running services with Marathon

- Migrated some existing services to Mesos for HA
- Existing and new services are now shipped as Docker containers
- Fixed frontend nodes dynamically mapped to internal Marathon services



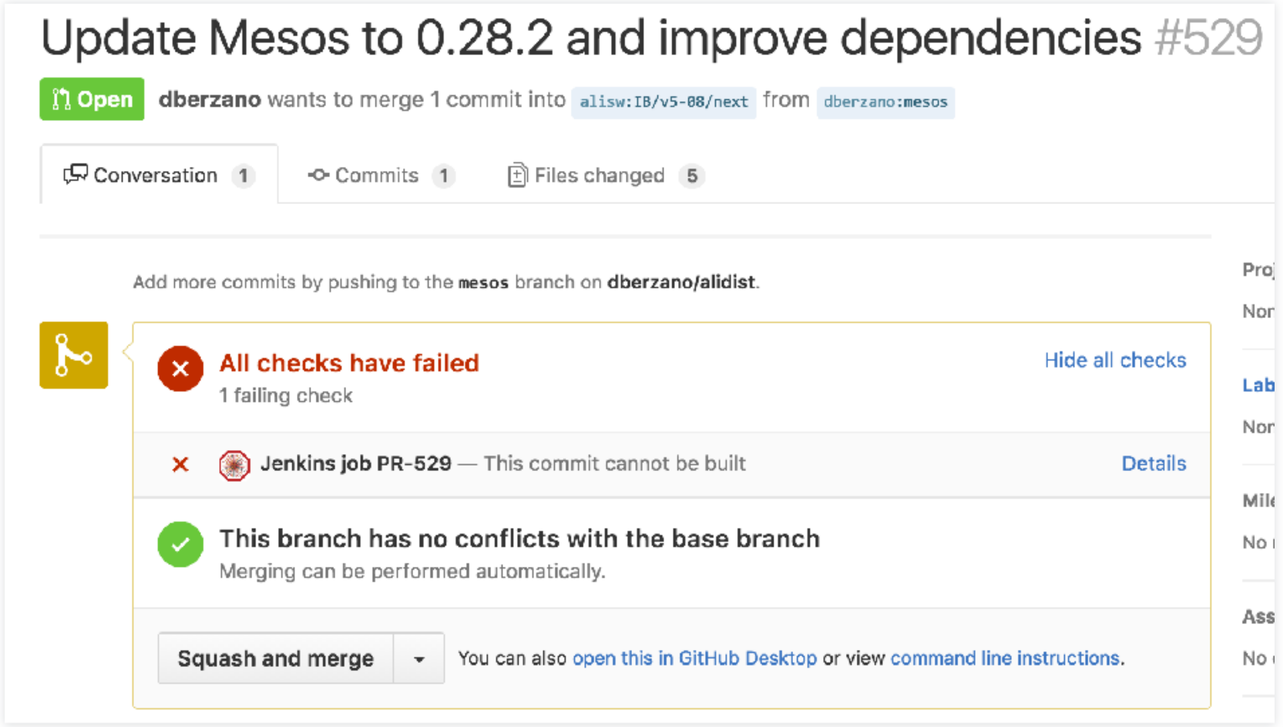
Long-running services with Marathon

- Migrated some existing services to Mesos for HA
- Existing and new services are now shipped as Docker containers
- Fixed frontend nodes dynamically mapped to internal Marathon services



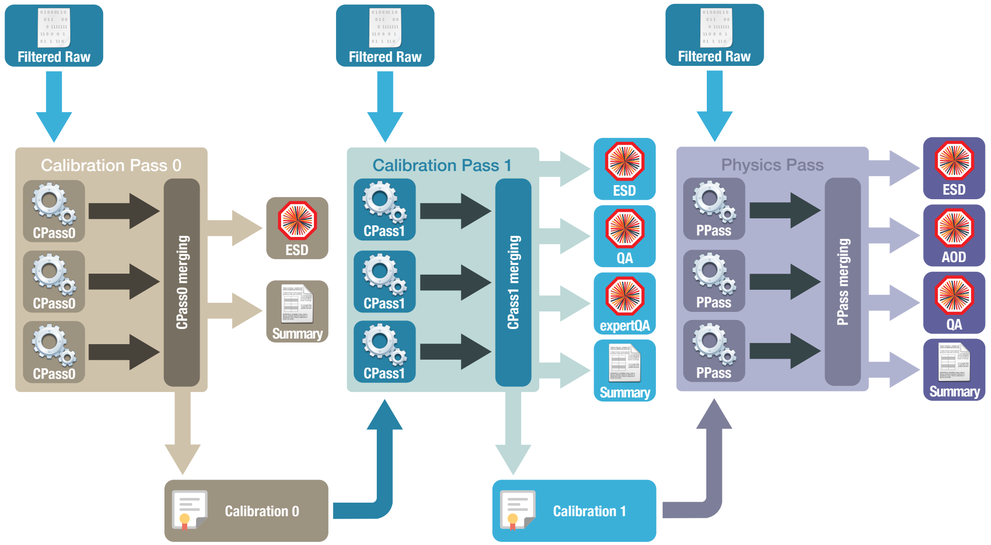
ALICE Continuous Integration

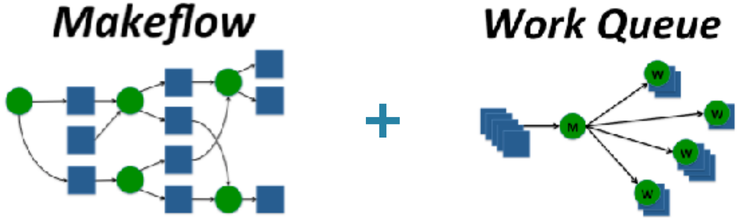
- **Builds and continuous integration with the existing Jenkins Mesos plugin**
- Good testbed for Mesos: we need high availability under unpredictable load
  Automatic pull requests test: they might come at any time in various amounts and we must be able to process all of them
- Decouple build environment from cluster nodes architecture
  Running obsolete SLC5 build containers on modern CERN CentOS 7 nodes
- Friendly to end users
  User interacts only via Git commands and GitHub web interface: the underlying infrastructure's complexity is invisible



ALICE Release Validation: workflow

- Full validation jobs on a Release Candidate
  Define job dependencies with Makeflow, submit to a lightweight agents system powered by Work Queue (usually ran on a batch system)
- Mesos gives resources quickly when needed
  We don't always run it, but when we do we need our resources promptly
- **Pre-existing use case integrated with Mesos**
  Makeflow+Work Queue used already: running on Mesos thanks to our custom C++ framework

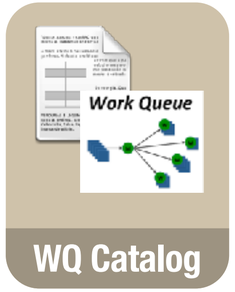


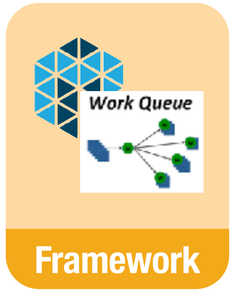


Part of cctools
ccl.cse.nd.edu

The Mesos Work Queue framework

- Makeflow job submitted as before
- Work Queue catalog knows how many resources are needed
- Work Queue code not modified



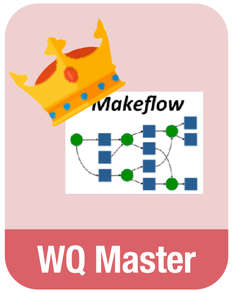


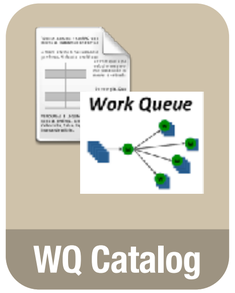
github.com/alisw/mesos-workqueue

The Mesos Work Queue framework

- Makeflow job submitted as before
- Work Queue catalog knows how many resources are needed
- Work Queue code not modified

Submits

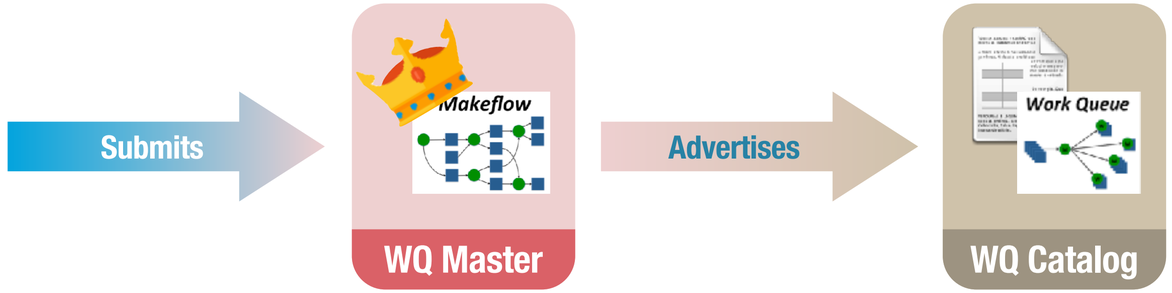




github.com/alisw/mesos-workqueue

The Mesos Work Queue framework

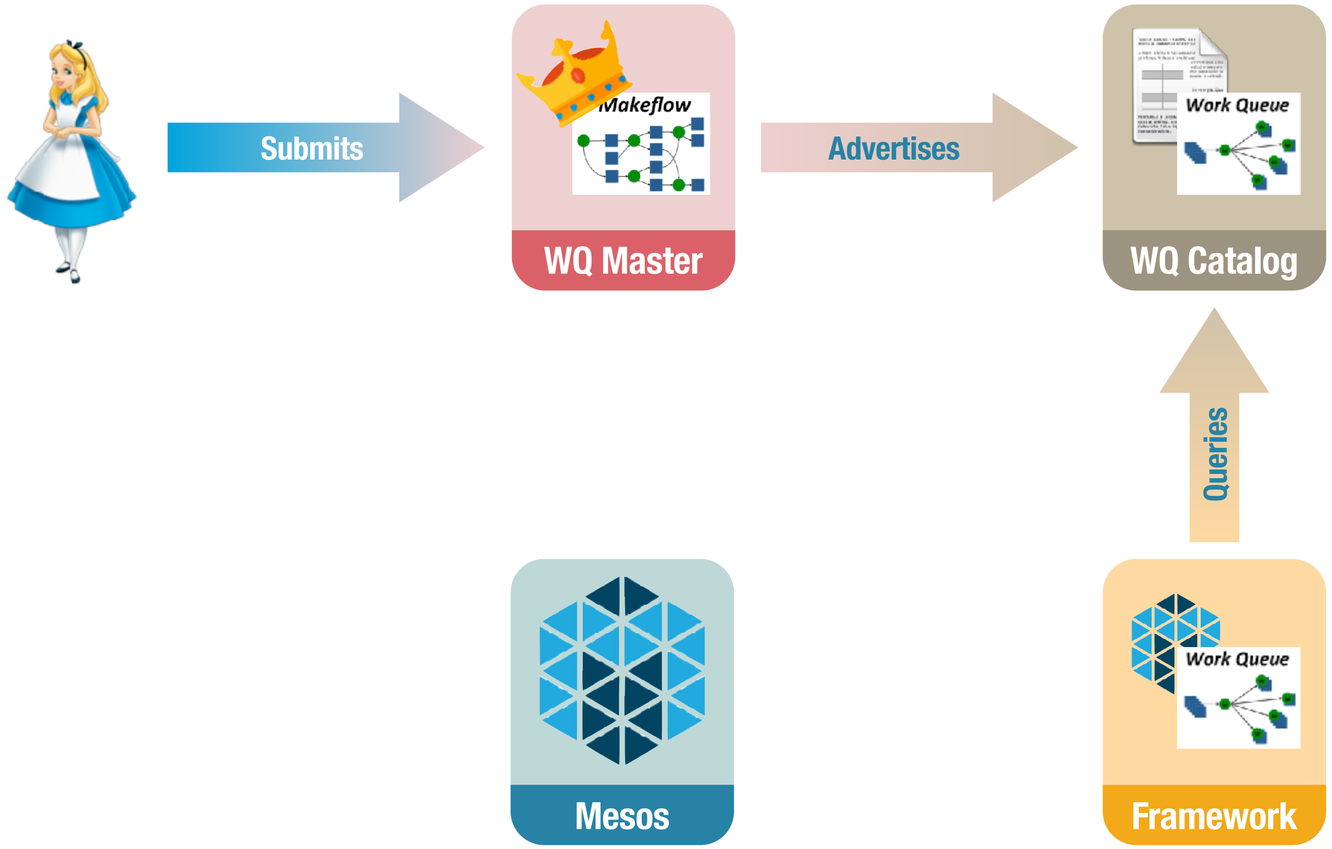
- Makeflow job submitted as before
- Work Queue catalog knows how many resources are needed
- Work Queue code not modified



github.com/alisw/mesos-workqueue

The Mesos Work Queue framework

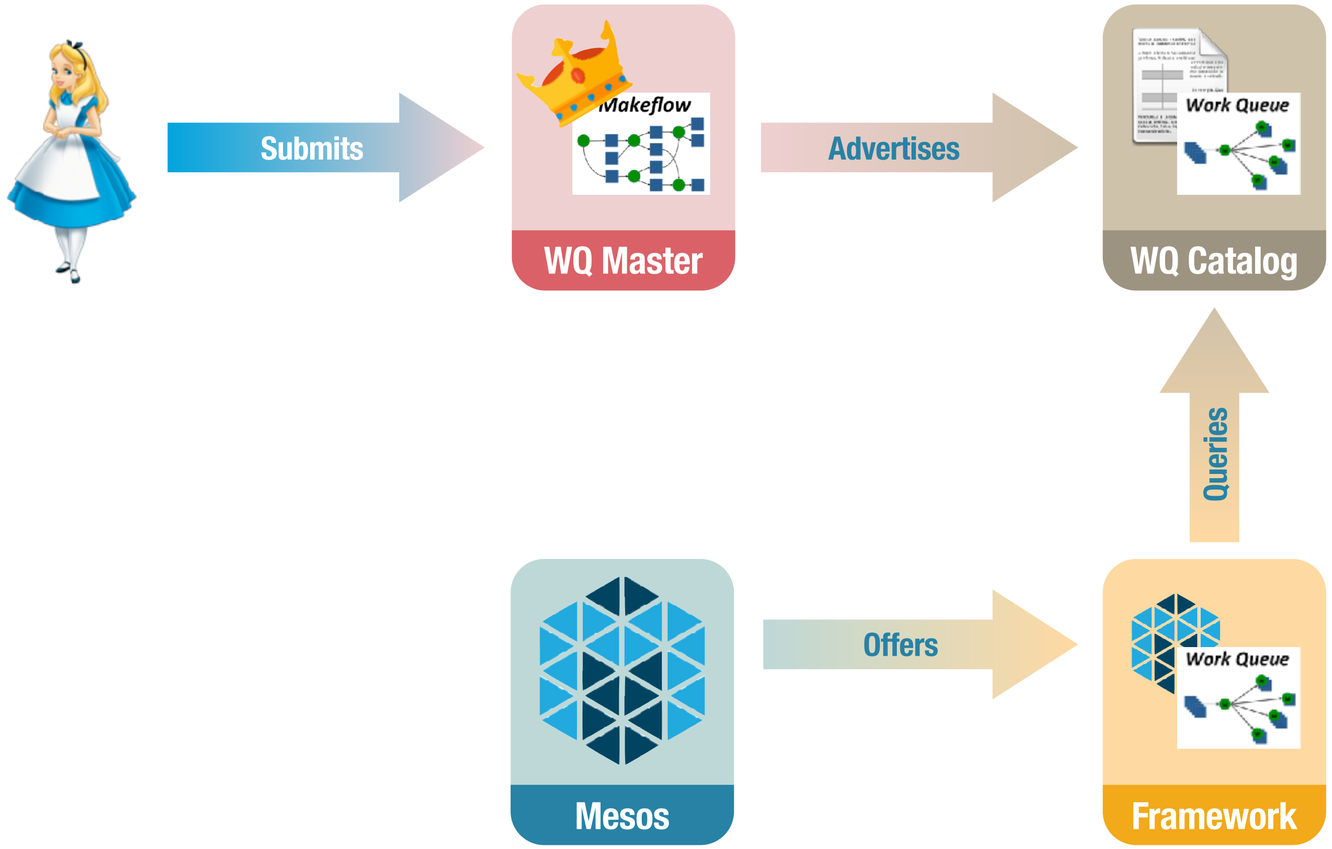
- Makeflow job submitted as before
- Work Queue catalog knows how many resources are needed
- Work Queue code not modified



github.com/alisw/mesos-workqueue

The Mesos Work Queue framework

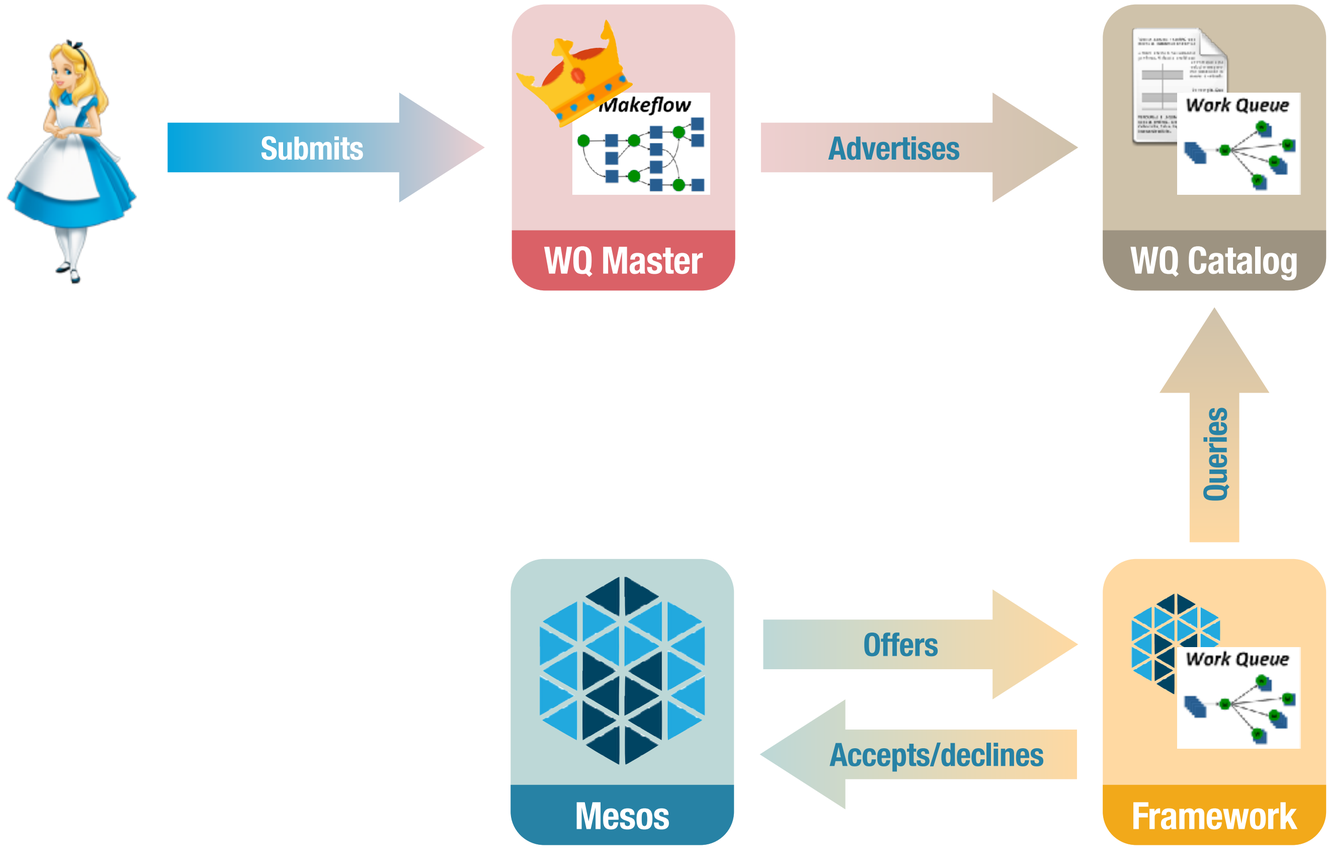
- Makeflow job submitted as before
- Work Queue catalog knows how many resources are needed
- Work Queue code not modified



github.com/alisw/mesos-workqueue

The Mesos Work Queue framework

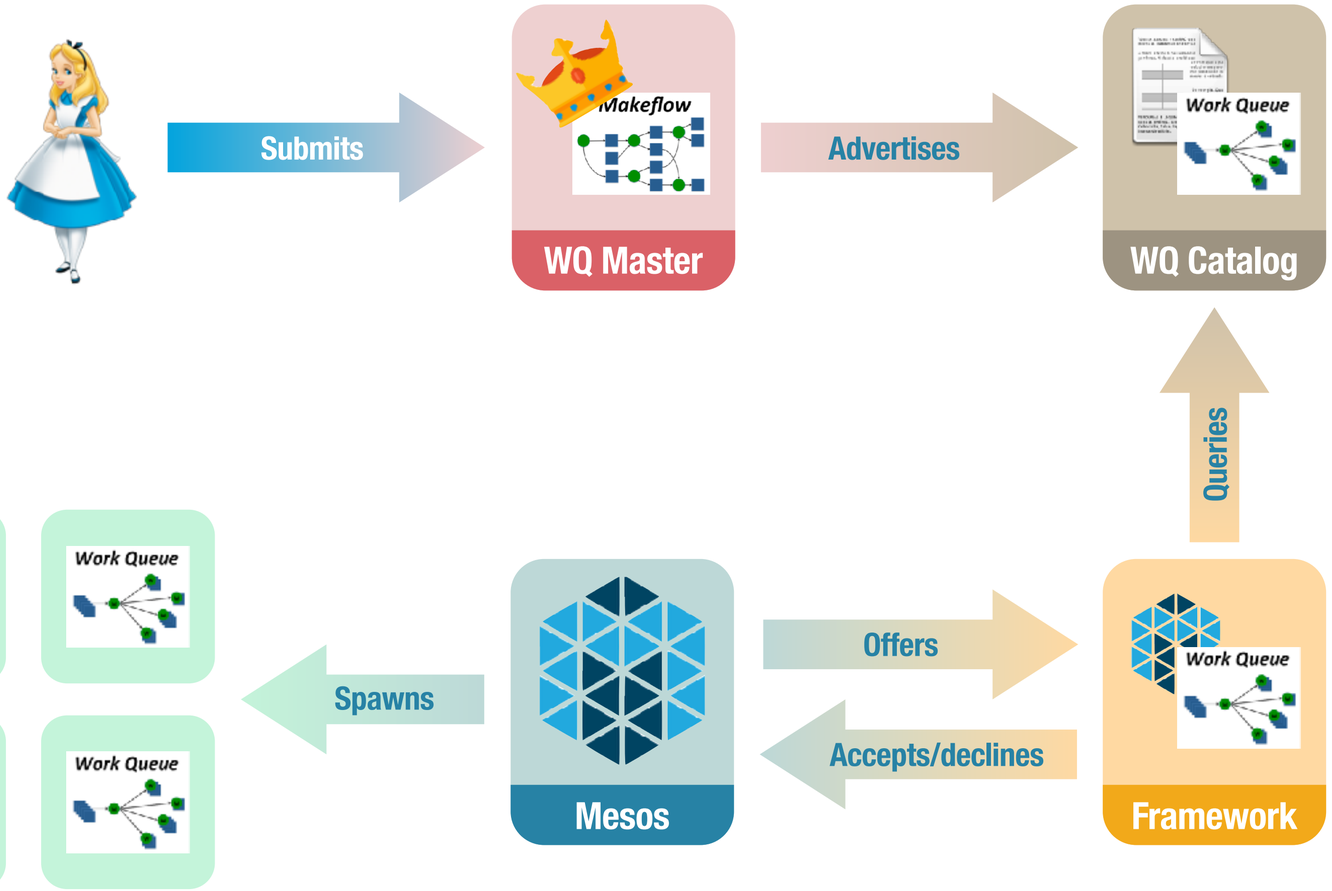
- Makeflow job submitted as before
- Work Queue catalog knows how many resources are needed
- Work Queue code not modified



github.com/alisw/mesos-workqueue

The Mesos Work Queue framework

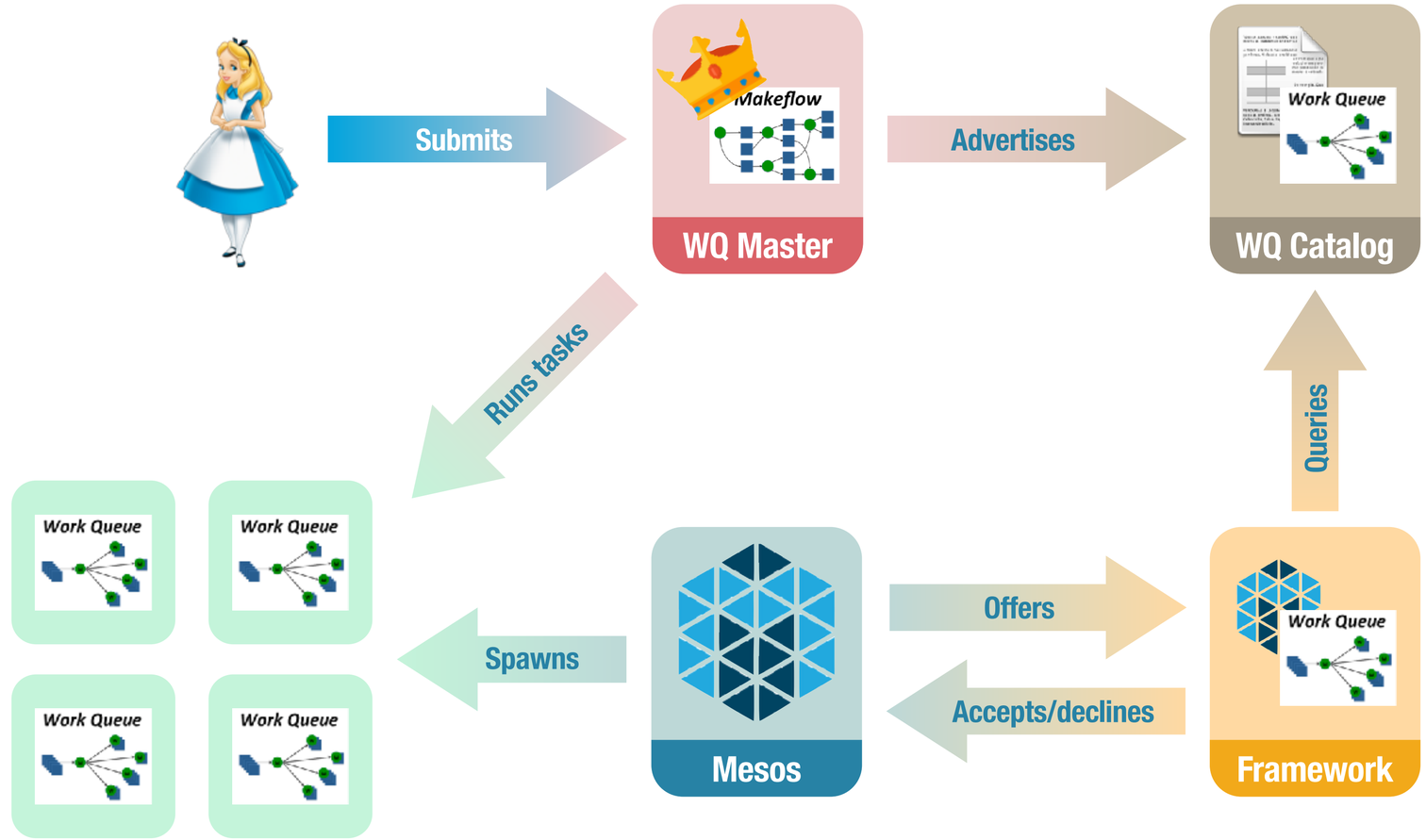
- Makeflow job submitted as before
- Work Queue catalog knows how many resources are needed
- Work Queue code not modified



github.com/alisw/mesos-workqueue

The Mesos Work Queue framework

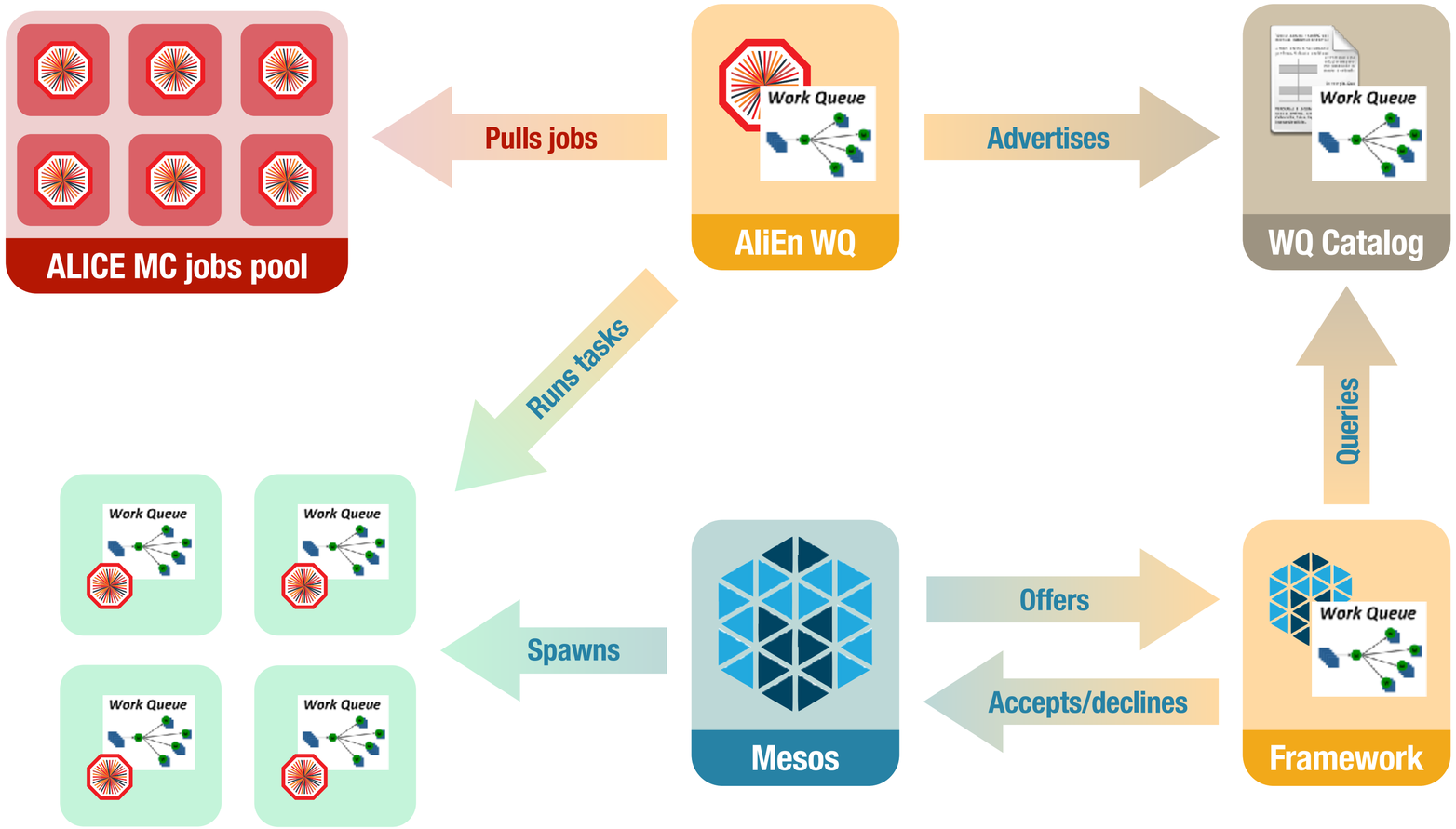
- Makeflow job submitted as before
- Work Queue catalog knows how many resources are needed
- Work Queue code not modified



github.com/alisw/mesos-workqueue

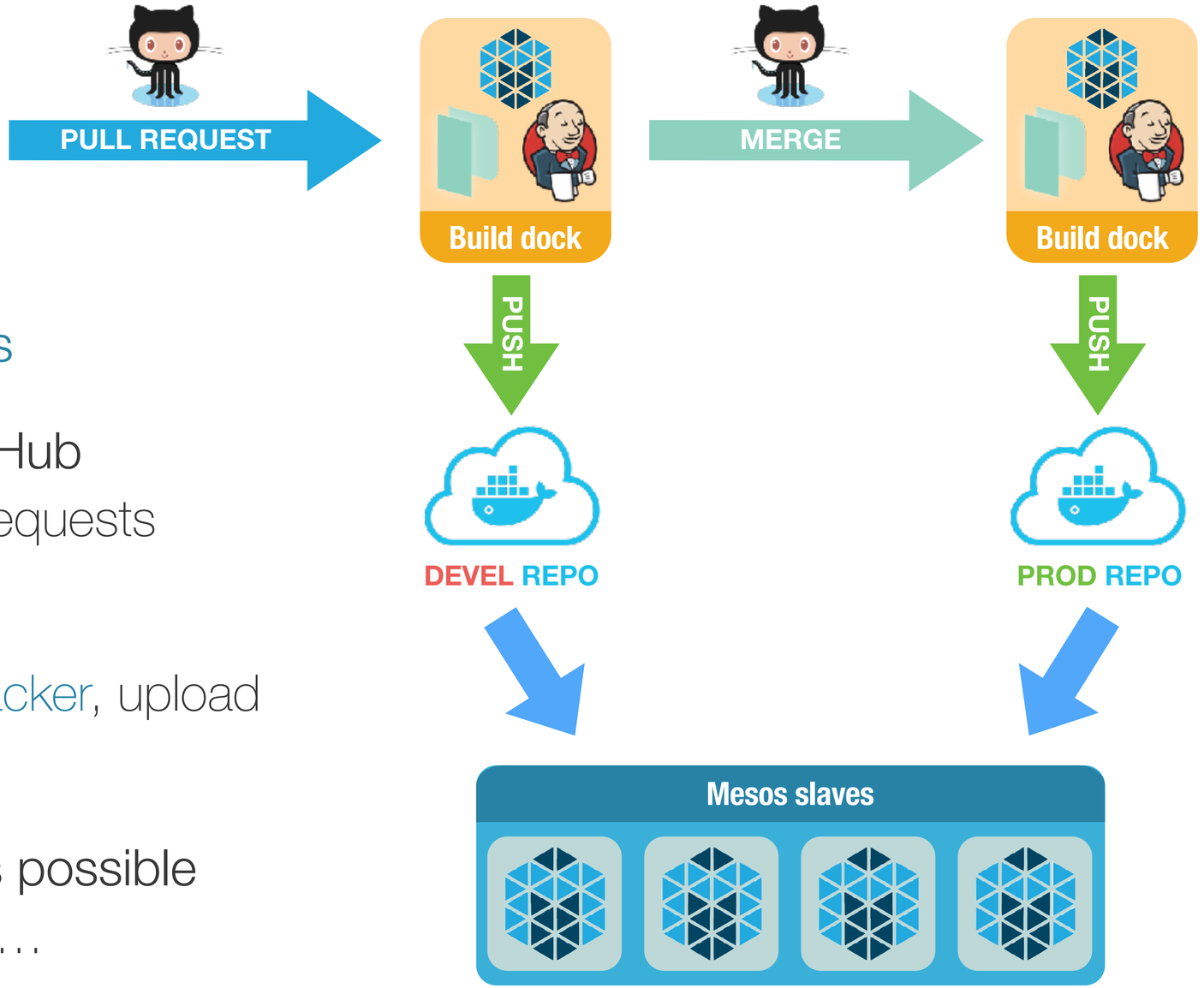
Monte Carlo Grid jobs on Mesos

- Custom C++ adapter enables Mesos to run ALICE Grid jobs
- Reusing the Mesos Work Queue framework
- Limit to Monte Carlo: CPU intensive, fill the Mesos cluster when idle
- Original ALICE Grid software (AliEn) was not modified



github.com/alisw/alien-workqueue

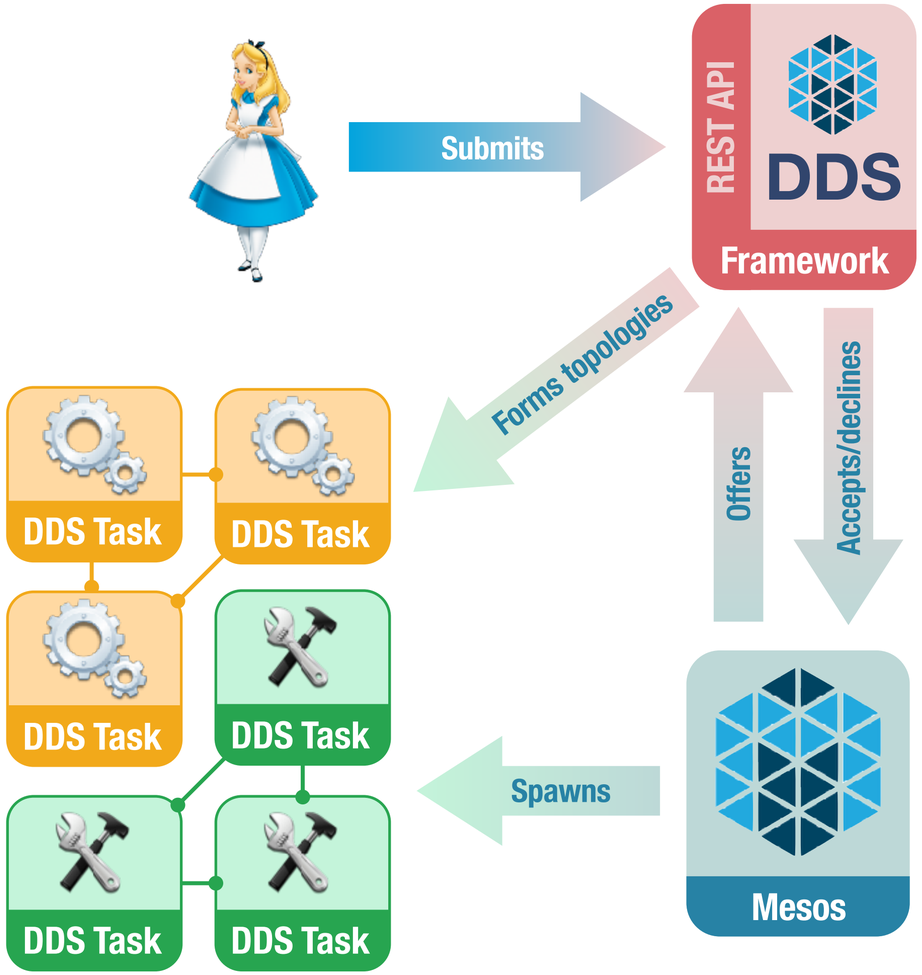
Testing and deploying container updates



- Microservices as containers
- Container manifests on GitHub
  Natural interaction via Pull Requests
- Continuous integration
  Build test containers with Packer, upload to the devel repository
- Multiple switchover policies possible
  Hard switch, rolling updates…

ALICE Run 3 Online/Offline on Mesos

Dynamic Deployment System on Mesos

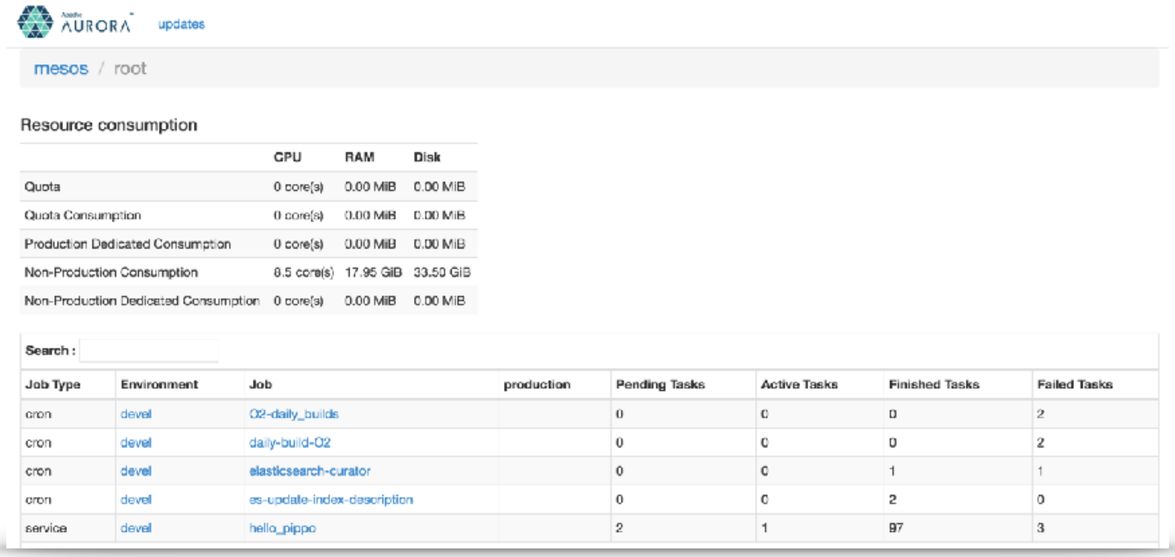


github.com/alisw/mesos-dds

- **ALICE Run 3: Online and Offline processing on the same cluster with defined task topologies**
- Dynamic Deployment System: dds.gsi.de
  Baseline for ALICE Run 3: we use it to define and launch tasks interconnected by a certain topology
- Run DDS workers as Mesos jobs
  Leverage Mesos to bring resource knowledge to DDS and retain the same user tasks submission interface
- The DDS Mesos plugin
  We have written a C++ plugin using Mesos as a “scheduler for our scheduler” without modifying DDS: we can share DDS resources with all other use cases

Future of ALICE software: datacenter with Aurora

- **Aurora is a Mesos framework for long-running services and cron jobs**
- Aurora: multiple users and environments on top of Mesos
  Allows to define priorities and preemption policies based on users and prod/devel environment
- Our challenge: single entry point for future ALICE Run 3 operations on a single datacenter
  Users can launch tasks on a single datacenter: Aurora makes sure production use cases always have top priority and makes possible to seamlessly switch development to production



Developed and used by Twitter with up to 10^5 containers: production and development seamlessly integrated on a single datacenter

Conclusions and outlook

- **We are happy with Mesos and we are constantly working to do more with it**
- Many ALICE Run 2 central Offline services migrated to it
  - Reasonable effort: zero to little code to write, no modifications in current use cases
  - Tangible results: resources used better, high availability, stability
- Currently testing it on a larger scale for Run 3 operations
  - Aurora seems ideal to allow users to use the same production cluster
  - Aurora and DDS on Mesos will possibly deliver infrastructure reliability for critical Run 3 operations where Online and Offline will be together

Thank you!