
KubeCon
CloudNativeCon
North America 2023


KubeCon | CloudNativeCon

North America 2023

WG Batch: What's New and What's Next?

Marcin Wielgus (@mwielgus)
Maciej Szulik (@soltysh)

Contacts

Meetings:

Biweekly on Thursday

at 10:00 AM EDT / 07:00 AM PDT / 04:00 PM CEST

Slack Channel:

#wg-batch

Email group:

wg-batch@kubernetes.io

Mission

Forum to discuss enhancements to **better support batch workloads** in core Kubernetes (eg. HPC, AI/ML, data analytics, CI)

A goal is to **reduce fragmentation** in the Kubernetes batch ecosystem

Stakeholders:

- SIG Scheduling
- SIG Apps
- SIG Node
- SIG Autoscaling

Scope

Additions to the **batch APIs** (Job, CronJob)

Job **queuing** primitives

Tools to maximize **clusters utilization**

Support for **specialized hardware**

Jobs

Elastic Indexed Jobs

Autoscaling Indexed Jobs

Modify **.spec.completions with .spec.parallelism**

Beta (ie. **on by default**) since **v1.27**

https://kep.k8s.io/3715

Pod failure policy for Jobs

Configure how to handle **pod failure**

Works **only** with Pod's **restartPolicy=Never**

https://kep.k8s.io/3329

```yaml
apiVersion: v1
kind: Job
spec:
  template:
    spec:
      containers:
      - name: job-container
        image: job-image
        command: ["./program"]
  backoffLimit: 6
  podFailurePolicy:
    rules:
    - action: FailJob
      onExitCodes:
        operator: NotIn
        values: [40, 41, 42]
    - action: Ignore
      onPodConditions:
      - type: DisruptionTarget
```

Pod replacements for Jobs

Flexibility in **waiting for pod's termination**

.spec.podReplacementPolicy TerminatingOrFailed or Failed

TerminatingOrFailed is the **default**

https://kep.k8s.io/3939

JobSets

Optional add-on

Allows **grouping jobs** to unify their lifecycle

Supports **varying templates** for jobs

Configurable **success policy** and **network options**

https://sigs.k8s.io/jobset

```yaml
apiVersion: jobset.x-k8s.io/v1alpha2
kind: JobSet
metadata:
  name: success-policy
spec:
  successPolicy:
    operator: All
    targetReplicatedJobs:
    - workers
  replicatedJobs:
  - name: leader
    replicas: 1
    template:
      spec:
        completions: 1
        parallelism: 1
        template:
          spec:
            containers:
            - name: leader
              image: bash:latest
              command: …
  - name: workers
    replicas: 1
    template:
      spec:
        completions: 2
        parallelism: 2
        template:
          spec:
            containers:
            - name: worker
              image: bash:latest
              command: …
```

leader job

worker job

Kueue

Problems

- Which of 100 training and data processing jobs should be running at the given time on limited resources?
- How to ensure that all pods of a job will quickly schedule before actually starting the job?
- How to allow a user to use as many of the spot instances they want but limit their on-demand reserved capacity?
- How to do all above, without replacing regular k8s components, in an autoscaled cloud environment?

What is Kueue?

Batch jobs scheduling and admission system that:

- Decides which jobs should run at a given moment and on what type of machines.
- Provides advanced resource controls like:
  - hardware-specific quota
  - quota sharing and borrowing
  - different policies for preemption and quota reclamation
  - job-level priorities
- Doesn't replace any of K8S components.

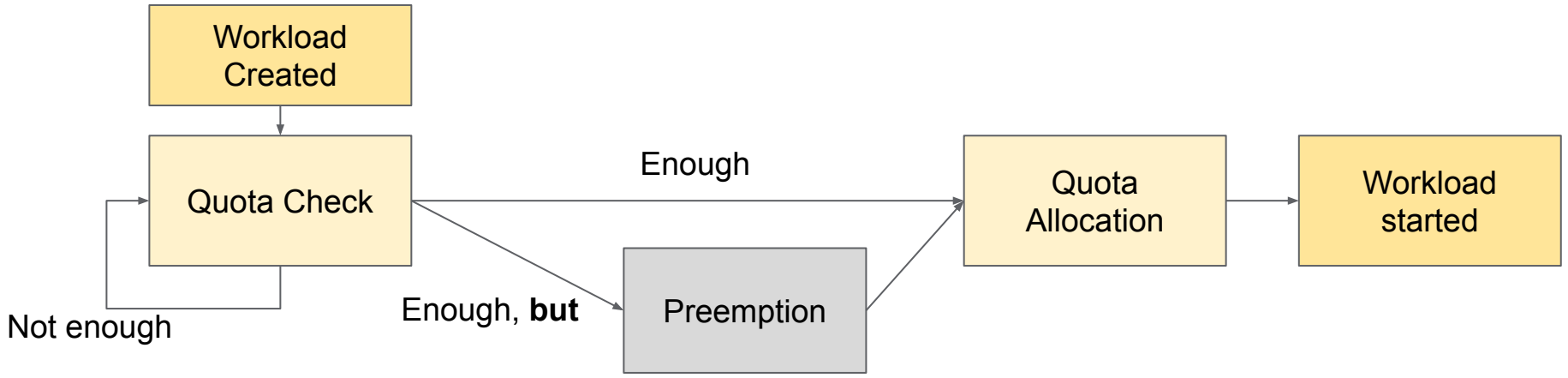
What is Kueue?

Meets users where they are, provides integrations with:

- Kuberentes Job
- Kubeflow Jobs portfolio (MPIJobs, PyTorchJob, TFJob, etC).
- RayJob
- JobSet
- Standalone pods

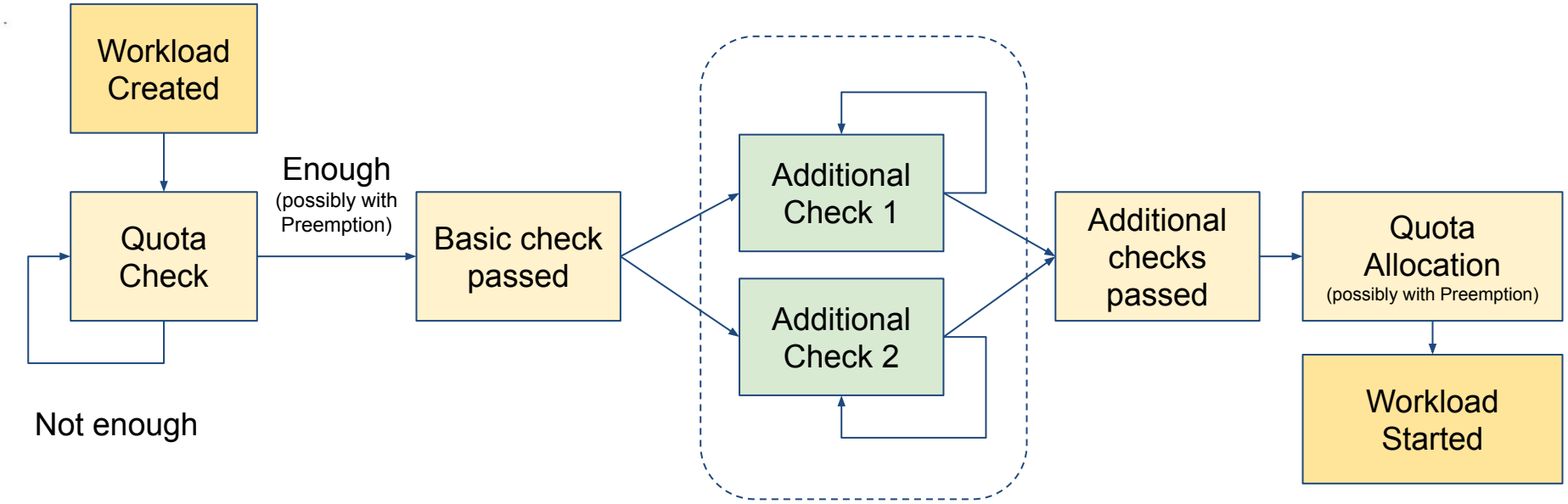
Kueue resource management

- Kueue admits Jobs via queues.
- Each queue can specify a quota for a particular set/kind of resources. For example in a queue there can be 50 cpus, 20 A100 gpus and 200 GB of ram.



New admission mechanism

- Allow users to define additional admission checks.
- The controller can be external to Kueue - no need to fork code to adjust it to you individual needs.



Provisioning Request

- Open Source API to ask Cluster Autoscaler (or any other autoscaling controller) to ensure space for the given set of pods.
- The request is not completed until the resources are available to be consumed.
- The exact details depends on the chosen provisioning class.

```
apiVersion: autoscaling.x-k8s.io/v1beta1
kind: ProvisioningRequest
metadata:
  name: provreq-gpu
  namespace: default
spec:
  provisioningClassName:
    generic-scale-up.k8s.io
  podSets:
  - count: 4
    podTemplateRef:
      name: pod-template-gpu
```

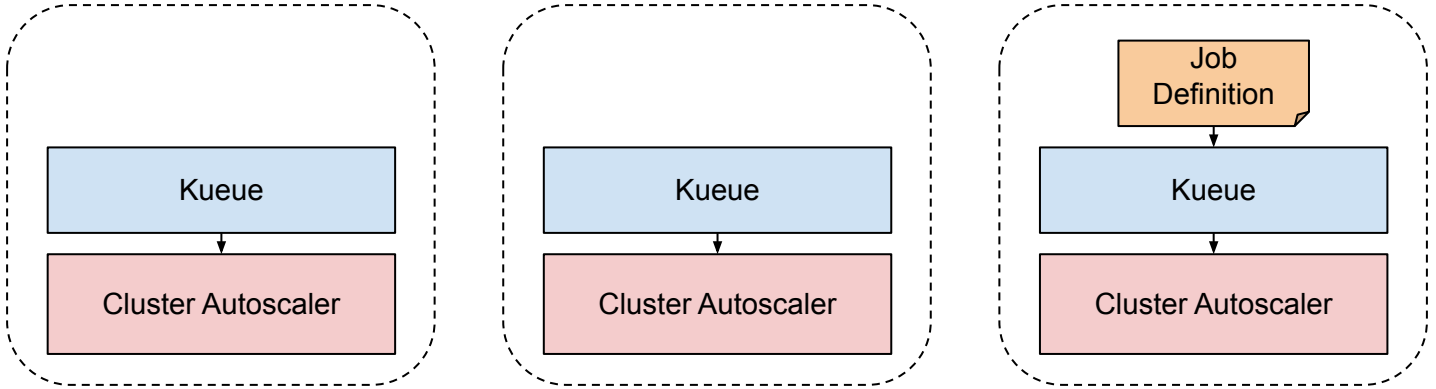
Provisioning Request

- Strengthens atomicity guarantees around gang scheduling.
- Provides gang scheduling with autoscaling.
- Classes are/will be available shortly in CA:
  - check-capacity.k8s.io
  - generic-scale-up.k8s.io
  - queued-provisioning.gke.io

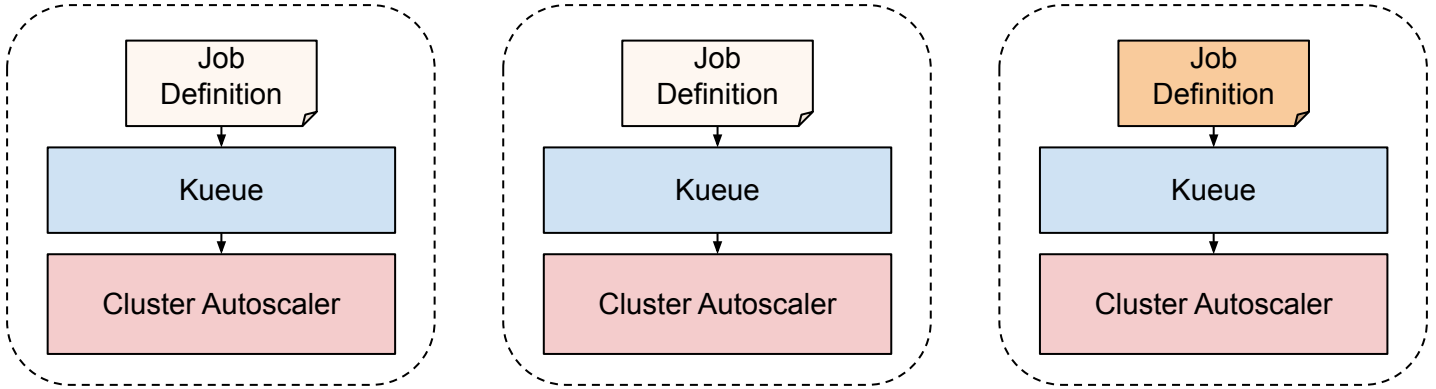
Need for multicluster

- Solve GPU obtainability problems (spots, on-demands are available at different locations at different time).
- Help users having clusters:
  - In a single region.
  - In multiple regions.
  - On different cloud and on prem.

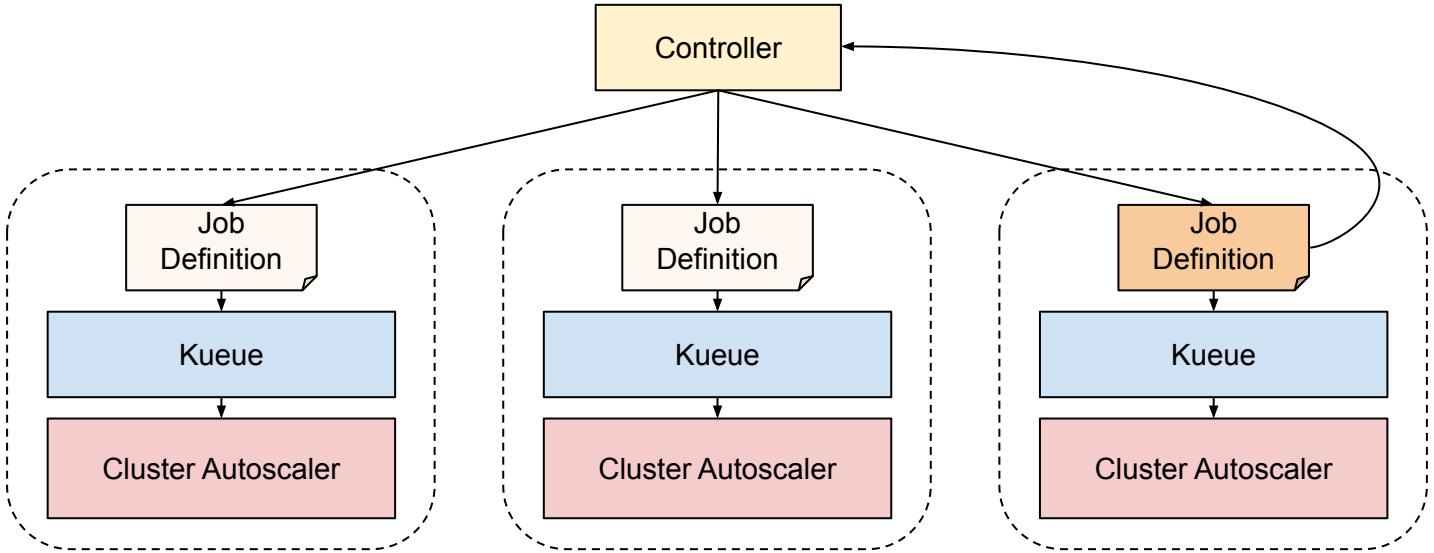
Going multicluster



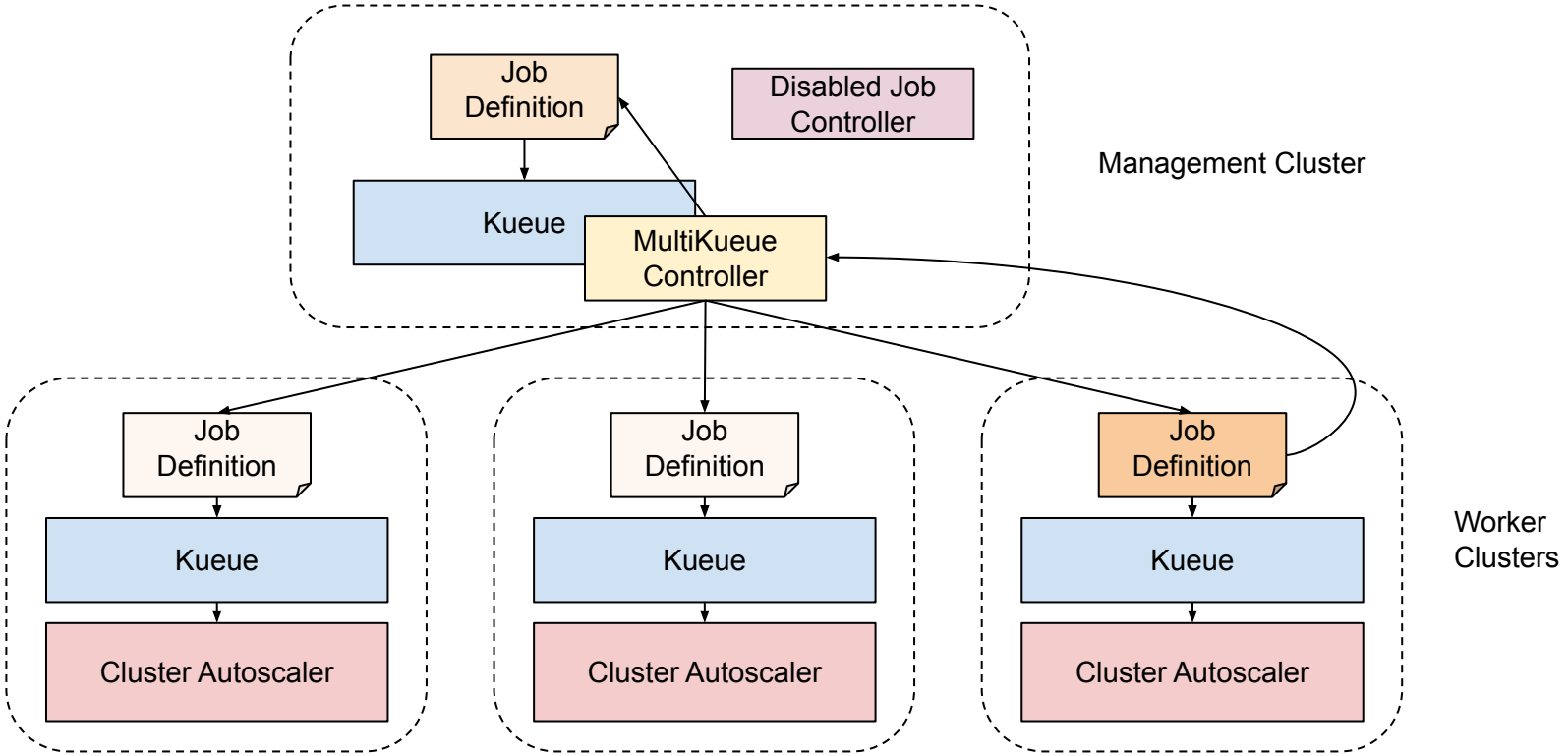
Going multicluster



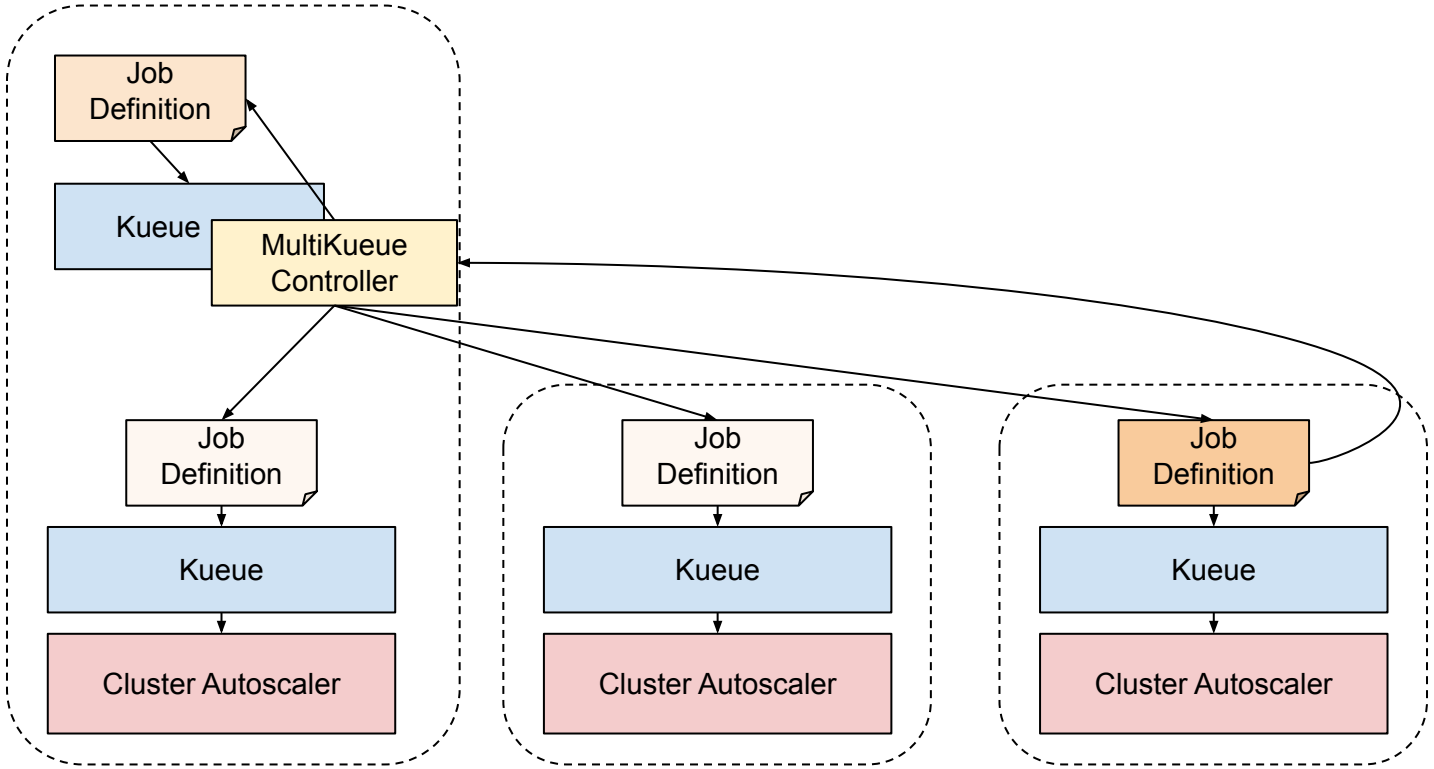
Going multicluster



Going multicluster



Going multicluster



MultiKueue

Pros:

- No new APIs for running Jobs - works with most of Kueue integrations.
- Same binary and functionality on both execution and management cluster.
- Works with Autoscaling via Provisioning Request.
- Works across regions, clouds and on-prems.

Cons:

- Doesn’t address storage.
- Need for management cluster.
- Need to set up roles and authentication between clusters.
- Need to create appropriate queues and namespaces in all clusters.

MultiKueue

Tell us what you think!

What else is coming to Kueue?

- Hierarchical quota structure
- Dedicated command line tools
- Hybrid resource assignments
- Budgets
- Enhanced visibility and dashboards
- ... and lots of other features

Where to find more

- https://kueue.sh
- https://github.com/kubernetes-sigs/kueue
- #wg-batch on Slack
- Batch WG meetings - biweekly on Thursday at 10:00 AM EDT / 07:00 AM PDT / 04:00 PM CEST

Q&A time

Please scan the QR Code above
to leave feedback on this session