
KubeCon
CloudNativeCon
North America 2024

KubeCon | CloudNativeCon

North America 2024

Bloomberg's Journey to Manage a Multi-Cluster Training Application with Karmada

Wei-Cheng Lai, Software Engineer
Yifan Zhang, Software Engineer

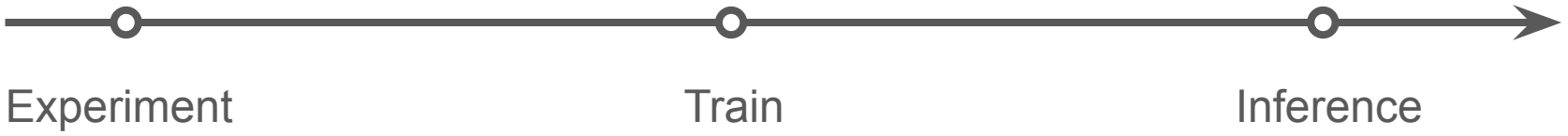
Data Science Platform Engineering @ Bloomberg

Agenda

- Data Science Platform at Bloomberg
- What is Karmada?
- Karmada in use at Bloomberg
- Future Roadmap
- Q&A

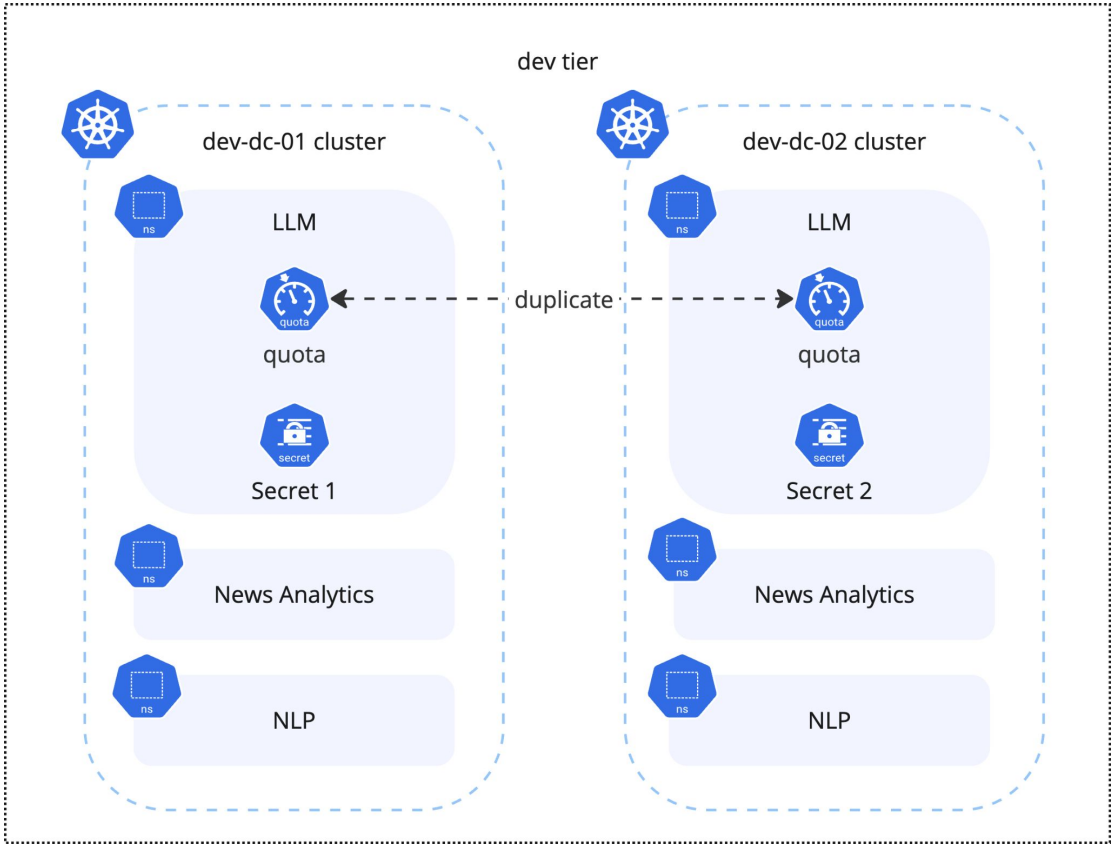
Data Science Platform at Bloomberg

- On-prem bare metal Kubernetes infrastructure for the full machine learning lifecycle

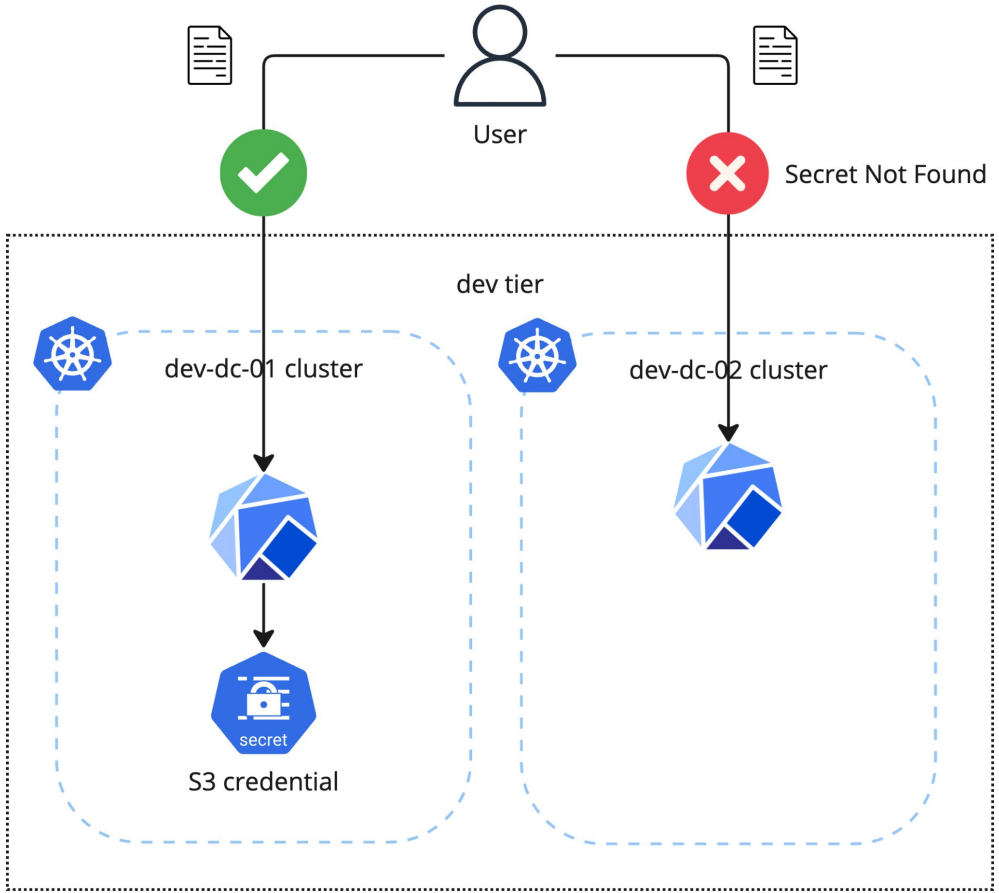


Data Science Platform at Bloomberg

- Namespace-based multi-tenancy
- Duplicate quota across clusters for high availability
- Separate configuration/ secret management

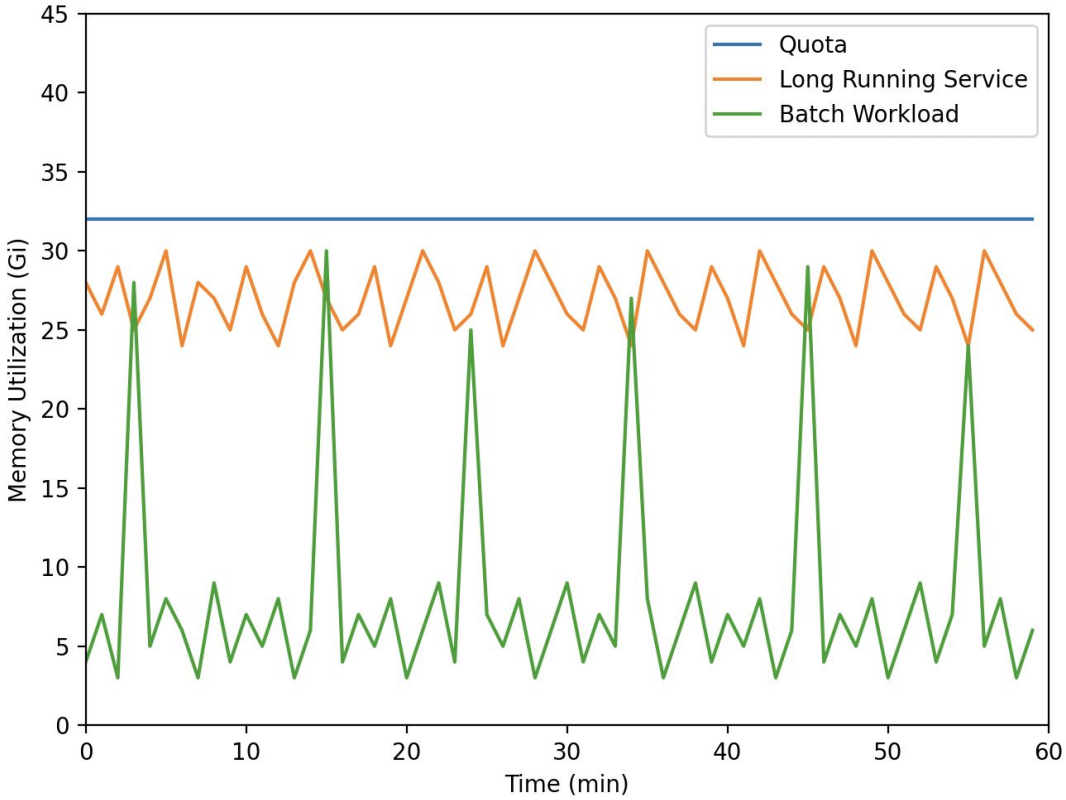


Challenges: Config/Credential Management



💡 Copy configurations and credentials to all clusters

Challenges: Over Budgeting

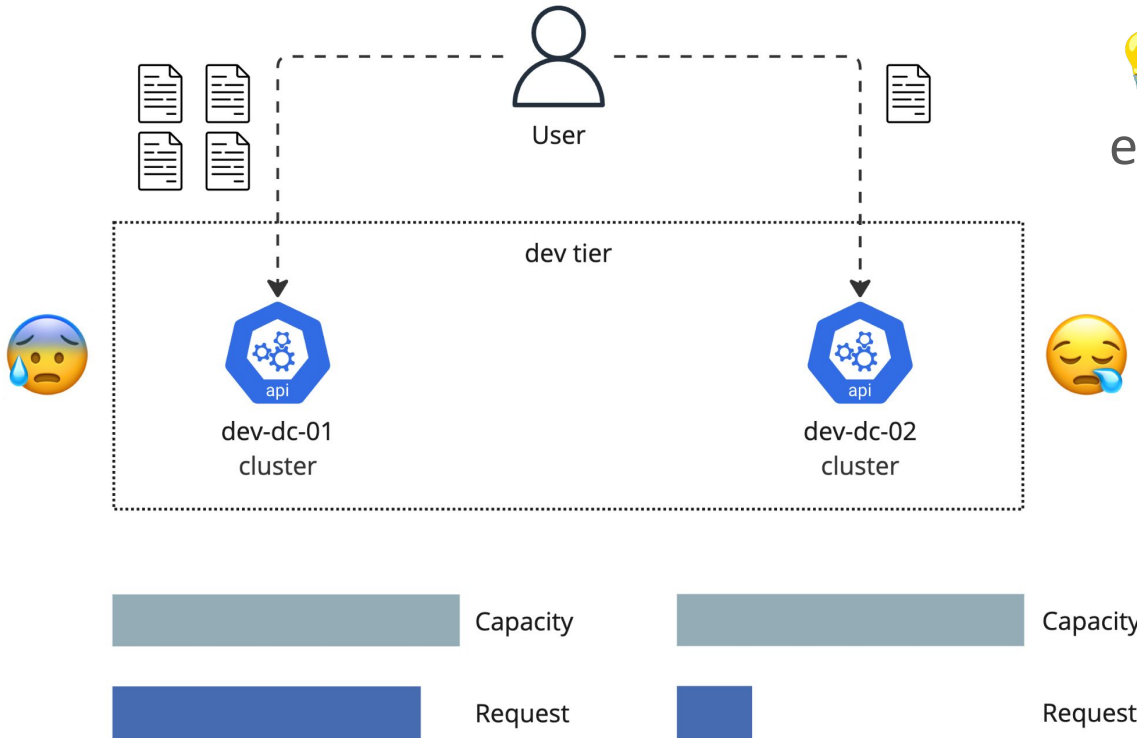


💡 Budget for maximum usage

-> Over-budgeting

-> Waste of resources

Challenges: Unbalanced Utilization



💡 Submit ML training jobs evenly to different clusters

Karmada

KARMADA

What is Karmada?

Open, Multi-Cloud, and Multi-Cluster Container Orchestration Platform

1

Compatible with Kubernetes native APIs

- Upgrade from single-cluster to multi-cluster deployments without code refactoring
- Seamlessly integrated with the Kubernetes single-cluster tool chain

2

Out-of-the-box

- Built-in policy sets for multiple scenarios,such as geo-redundancy, intra-city active-active, and remote DR

3

No vendor lock-in & Centralized Management

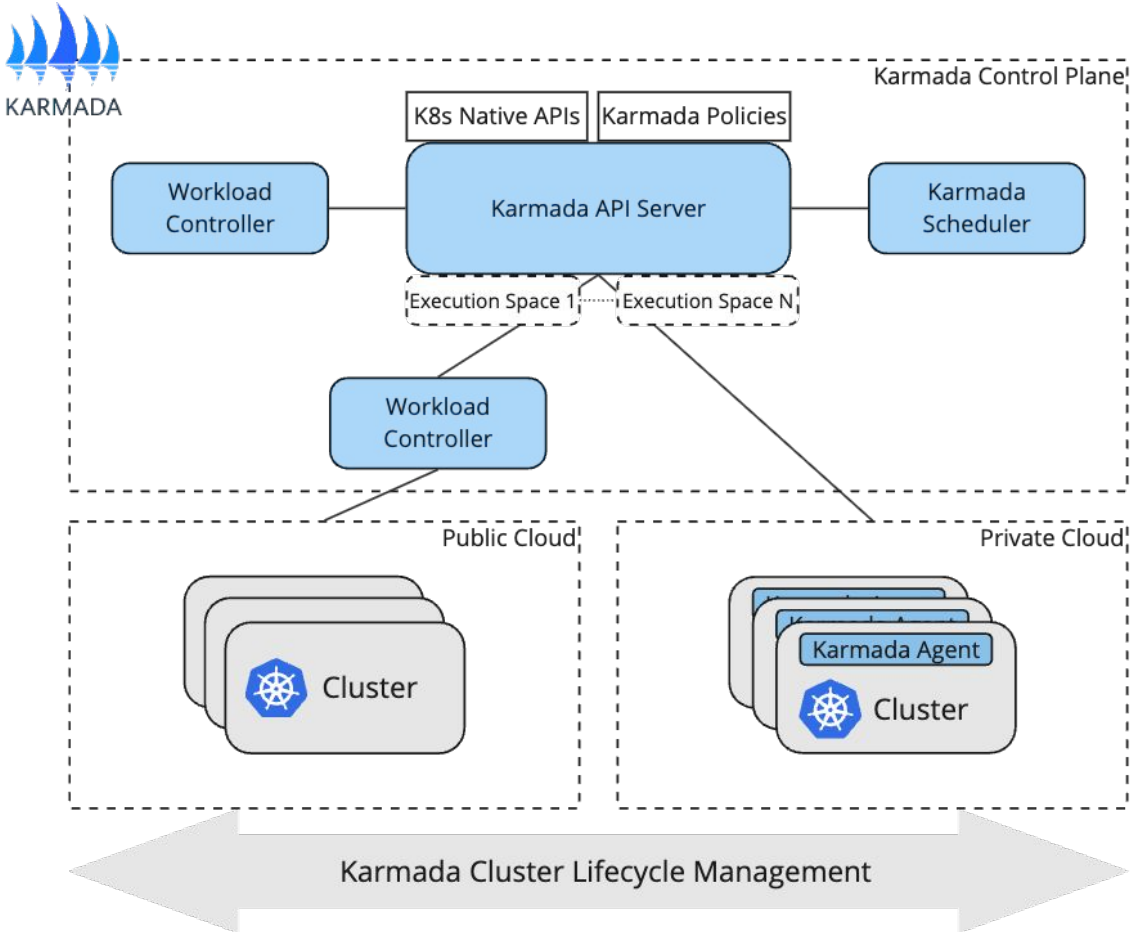
- Support for multi-cloud platforms, auto resource allocation, and free migration
- Not bound to any commercial products from cloud vendors
- Support public clouds, private clouds, and edge clouds

4

Various Multi-Cluster Scheduling Policies

- Cluster scheduling based on affinity and multi-cluster
- HA deployment across regions, AZs, clusters, and vendors

Architecture Overview



Multi-Cluster Management

Significantly reducing repeated management operations

Workload Propagation Across Clusters

Advanced policies meet various scheduling requirements

Cross-Cluster Application Failover

Graceful migration ensures uninterrupted services

Unified Authentication

Significantly reducing the effort required for authentication/audit

Global Resource View

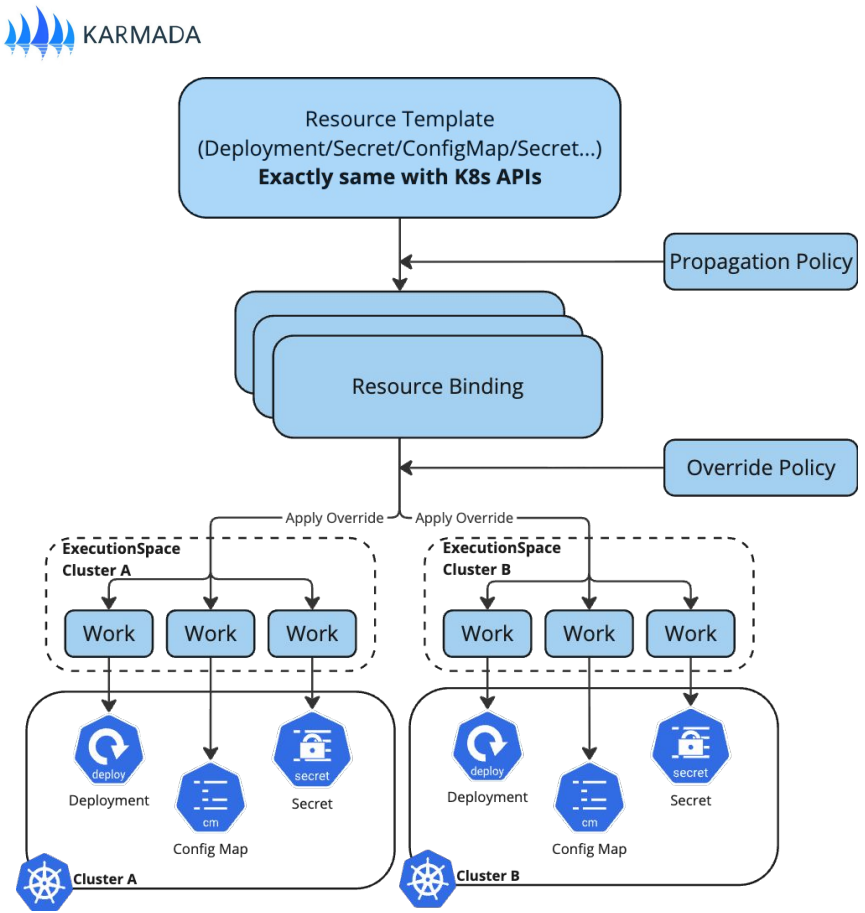
Unified query entry with efficient search support

Multi-Cluster Service Discovery

Effortless cross-cluster service communication

KARMADA

Core Concepts of Karmada



User-Facing APIs

Resource Template

- Same as native Kubernetes API definitions, including CRDs
- Used to create multi-cluster applications without modification

Propagation Policy

- Widely applicable policy for multi-cluster application scheduling

Override Policy

- Differentiated configuration policy applicable across clusters

Resource Binding

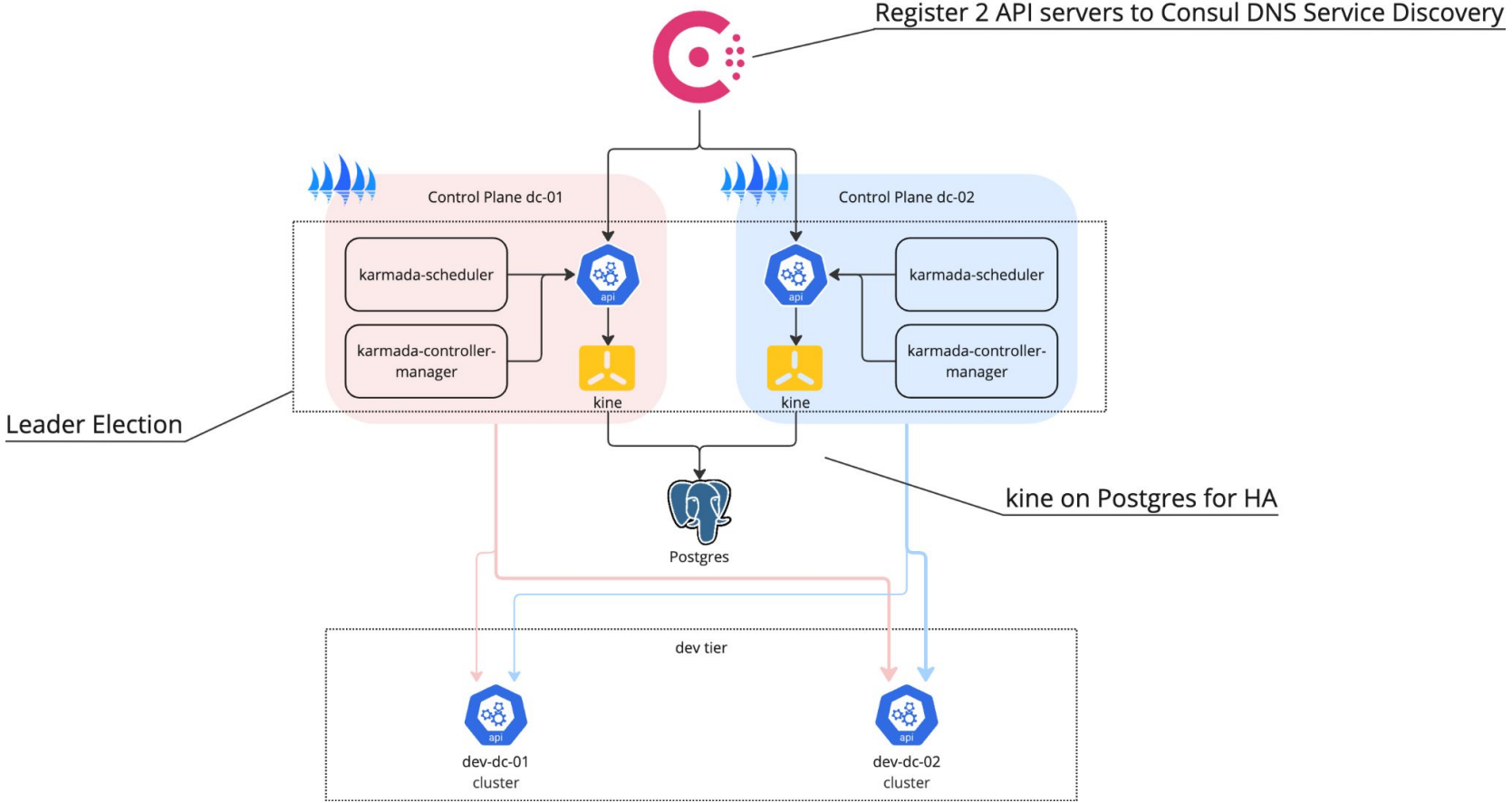
- Unified abstraction, which drives internal processes

Work

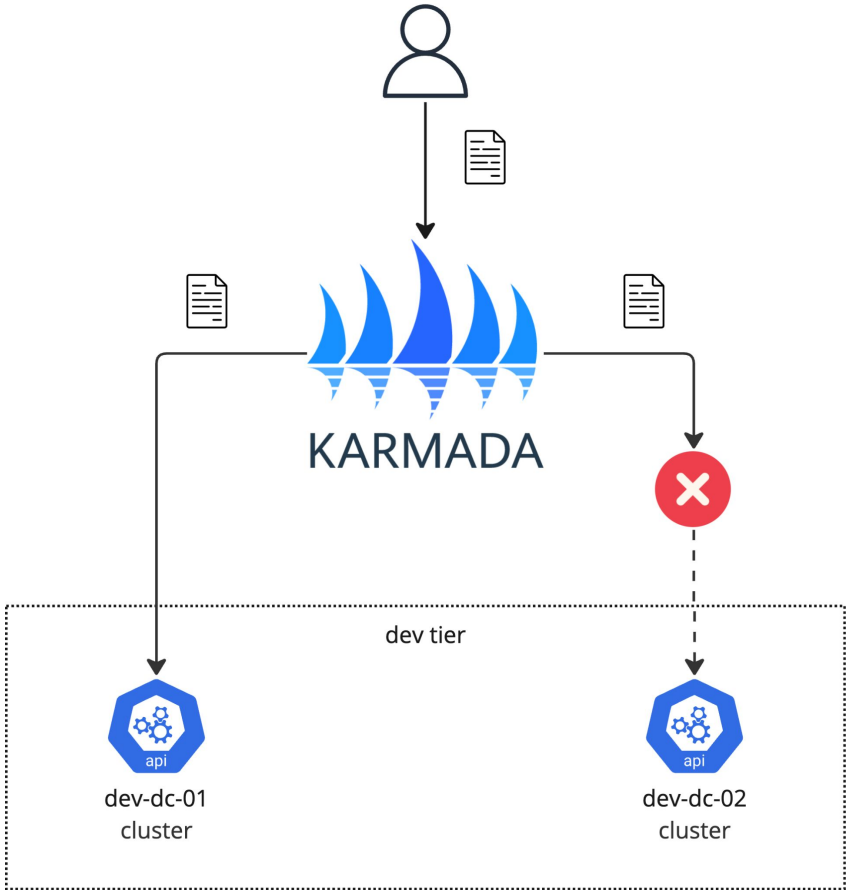
- Object at the federation layer to present a resource in a member cluster

Karmada in use at Bloomberg

Control Plane Setup

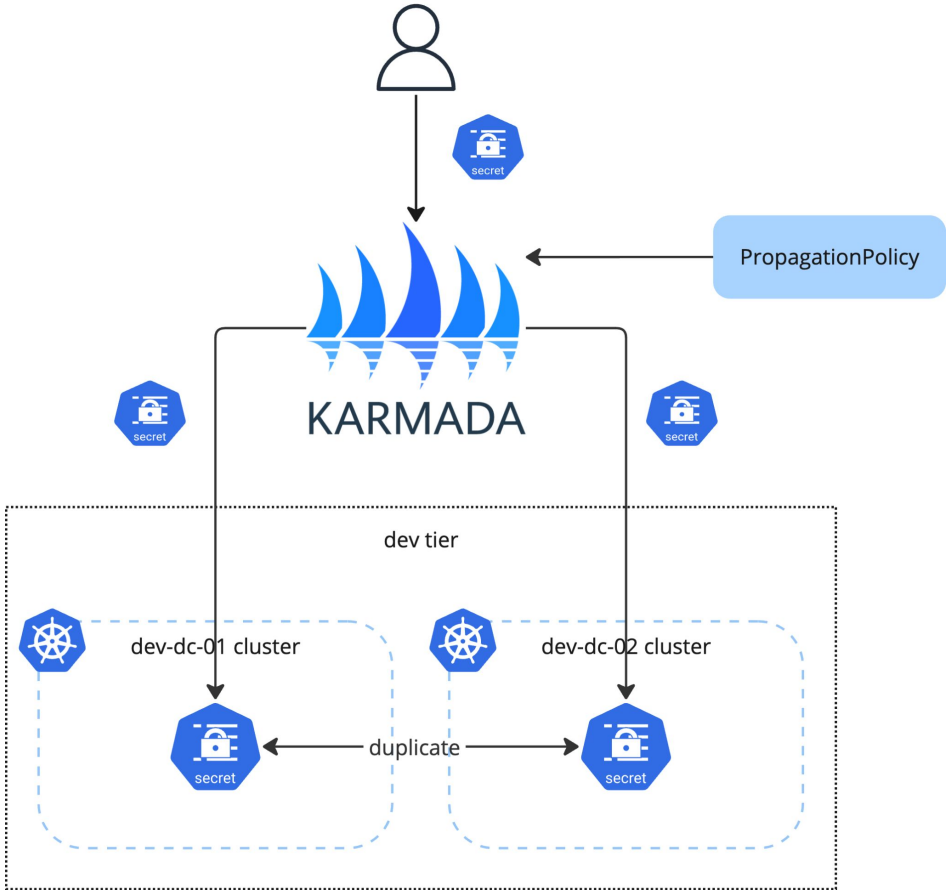


Automatic Failover



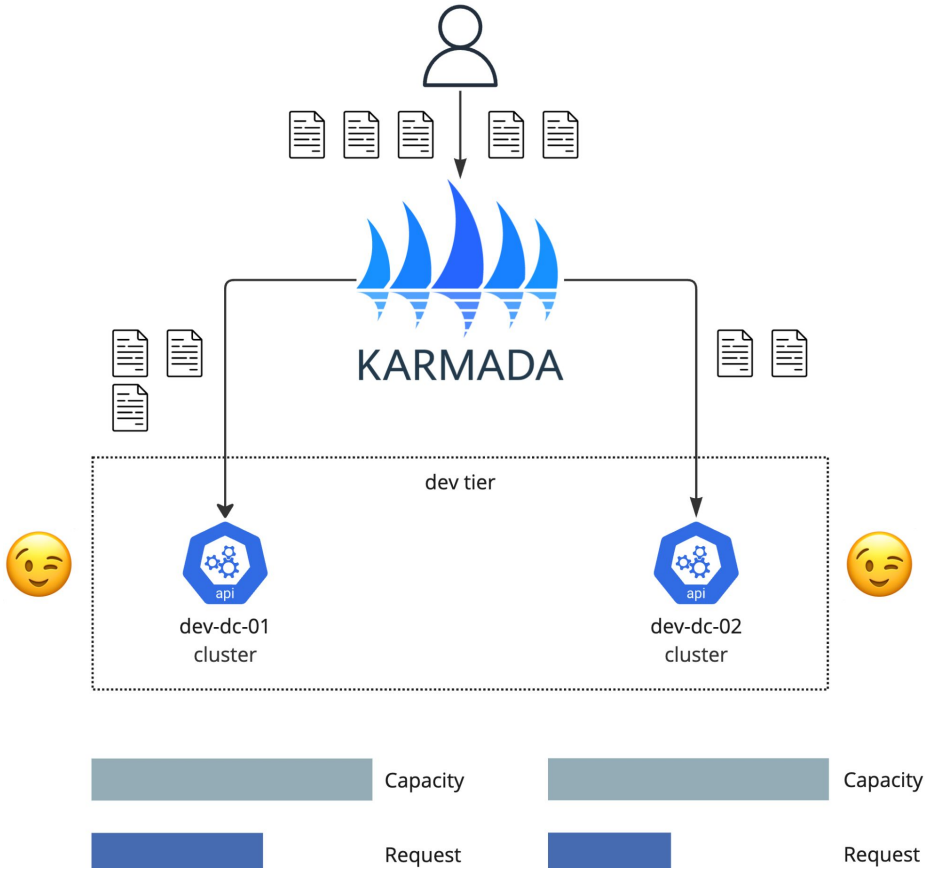
- Select available clusters for new jobs
- Re-schedule running jobs from failed clusters to available clusters

Configuration/Credential Propagation



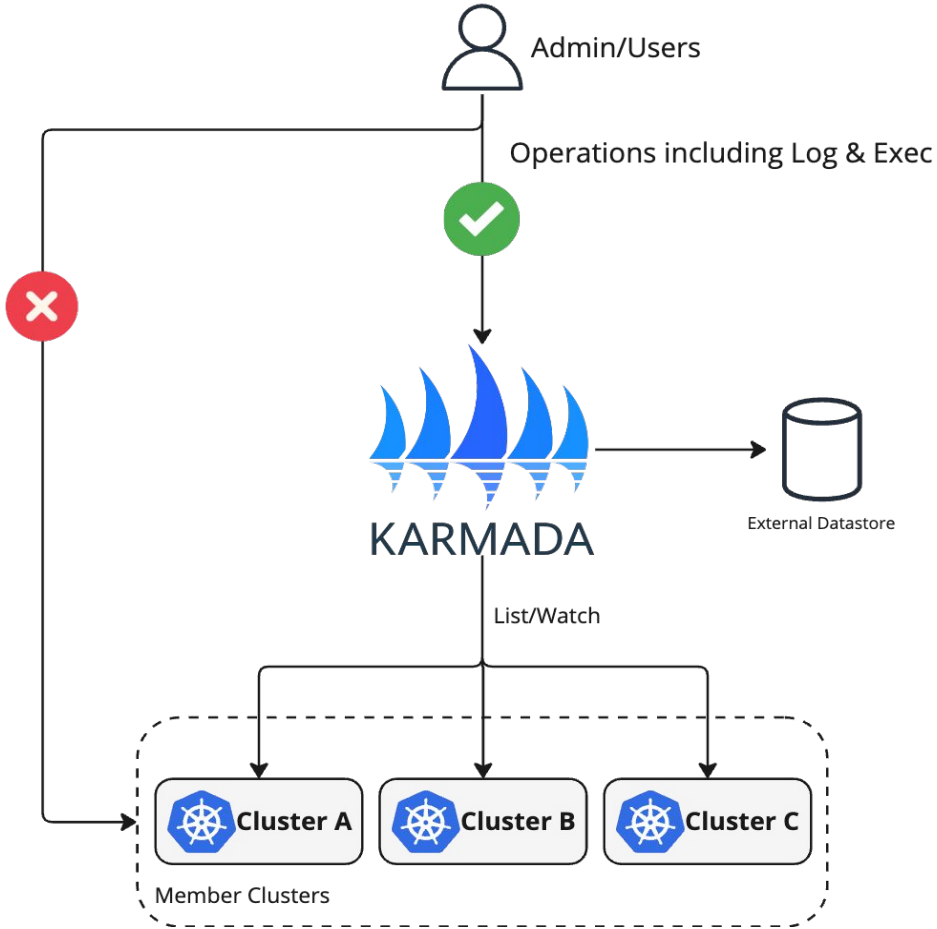
- Automatically propagate configurations and credentials to all clusters under the same tier
- One-time creation, access anywhere
- Auto-propagate to new member clusters

Balanced Scheduling



- Automatically balance workloads across different clusters in the same tier

Global Uniform Resource View

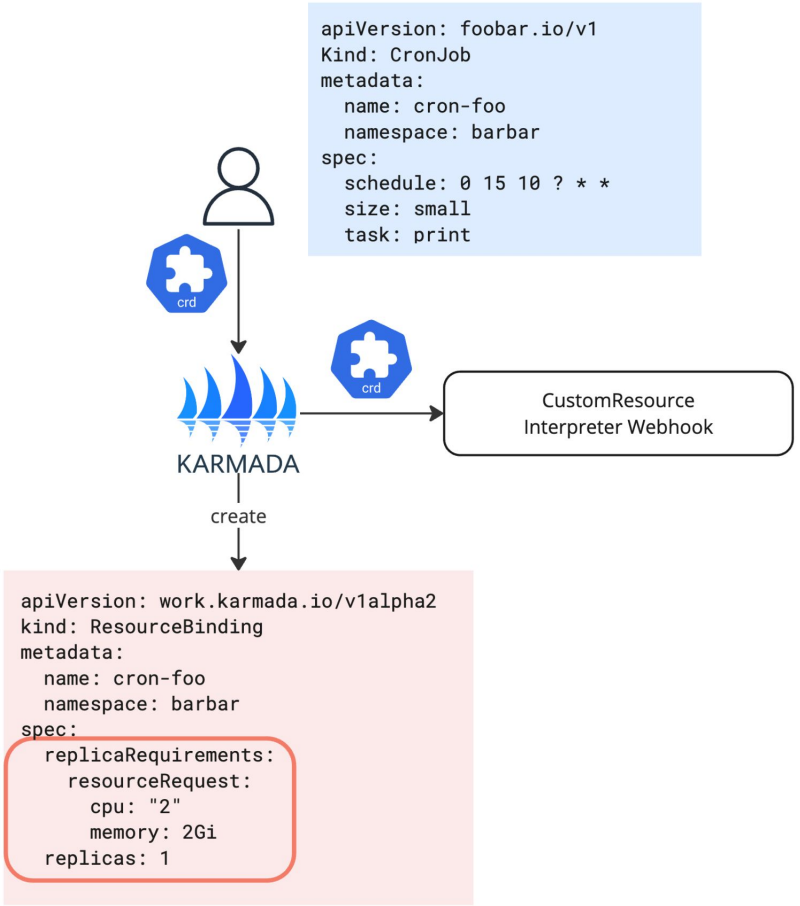


- Admins/Users are able to perform operations against the Resources in member clusters without needing to access member clusters, including **log** and **exec**.

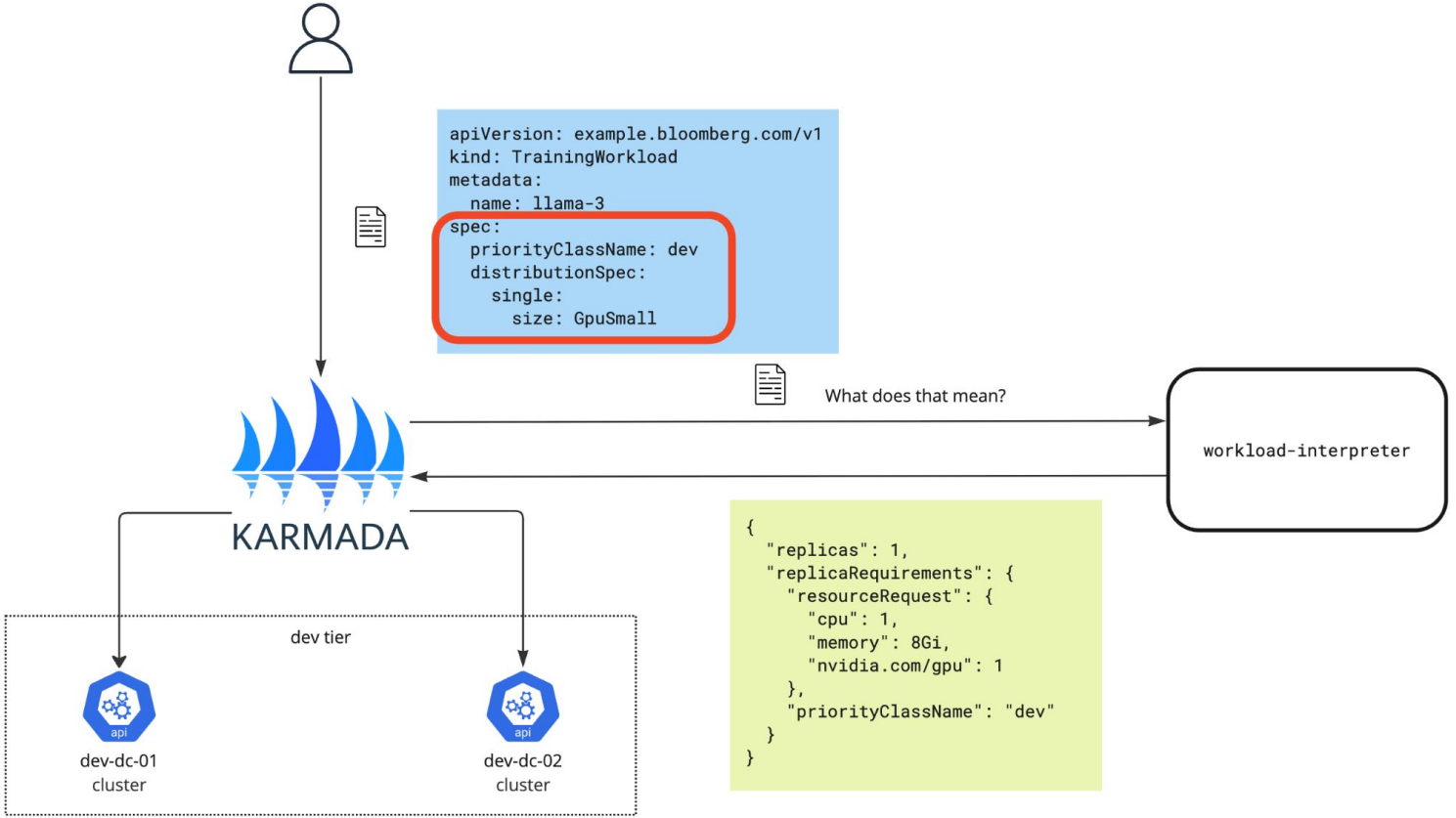
Custom Resource Interpreter

How does Karmada understand your Custom Resource?

→ **Resource Interpreter Webhook**

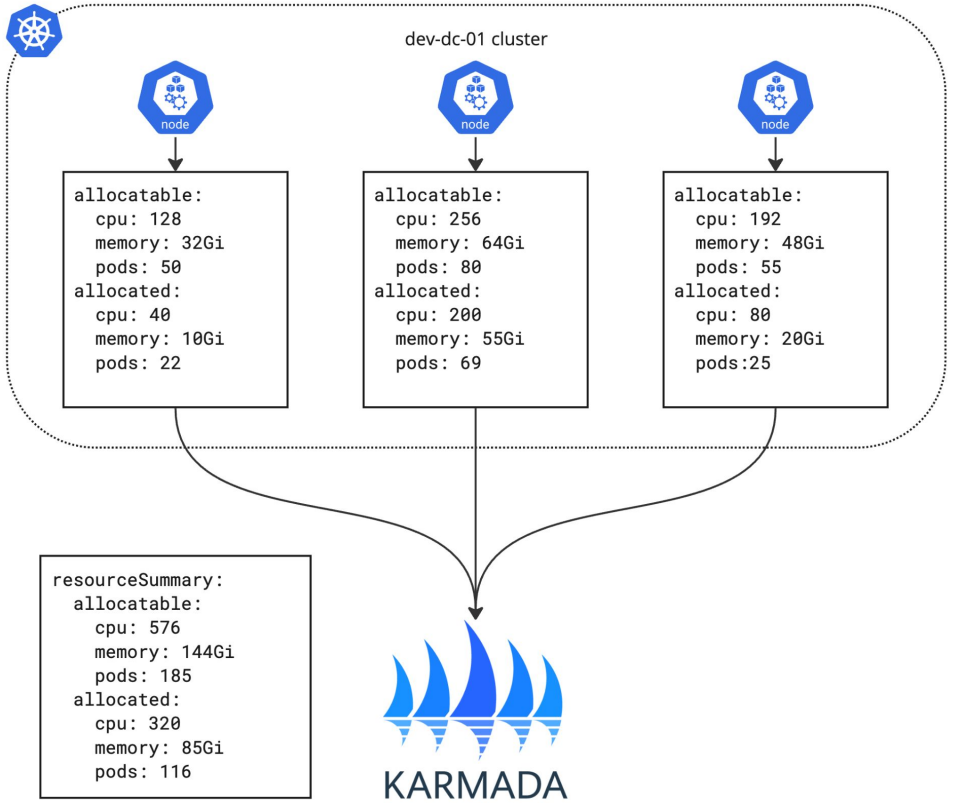


Customized Workload Interpreter



Resource Modeling

- Calculates free resources on th entire cluster
- Fast, but not accurate

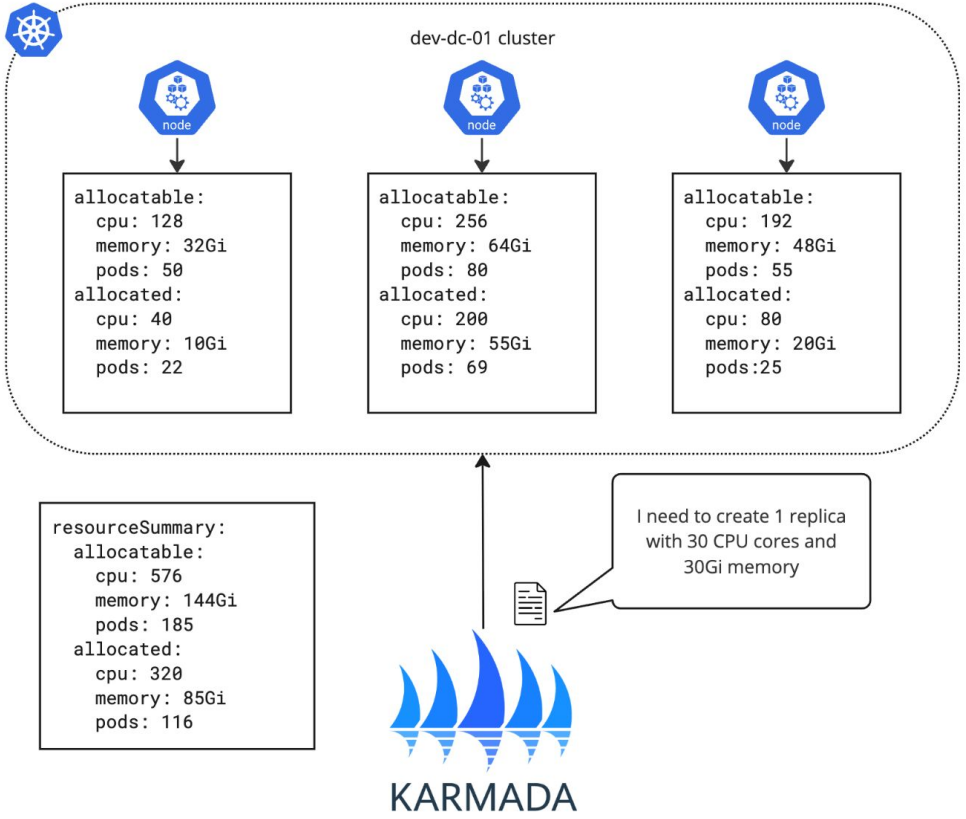


Resource Modeling

⛔ No node can run this workload

✅ Free Resource on the cluster:

- CPU: 256
- Memory: 59Gi
- Pods: 69



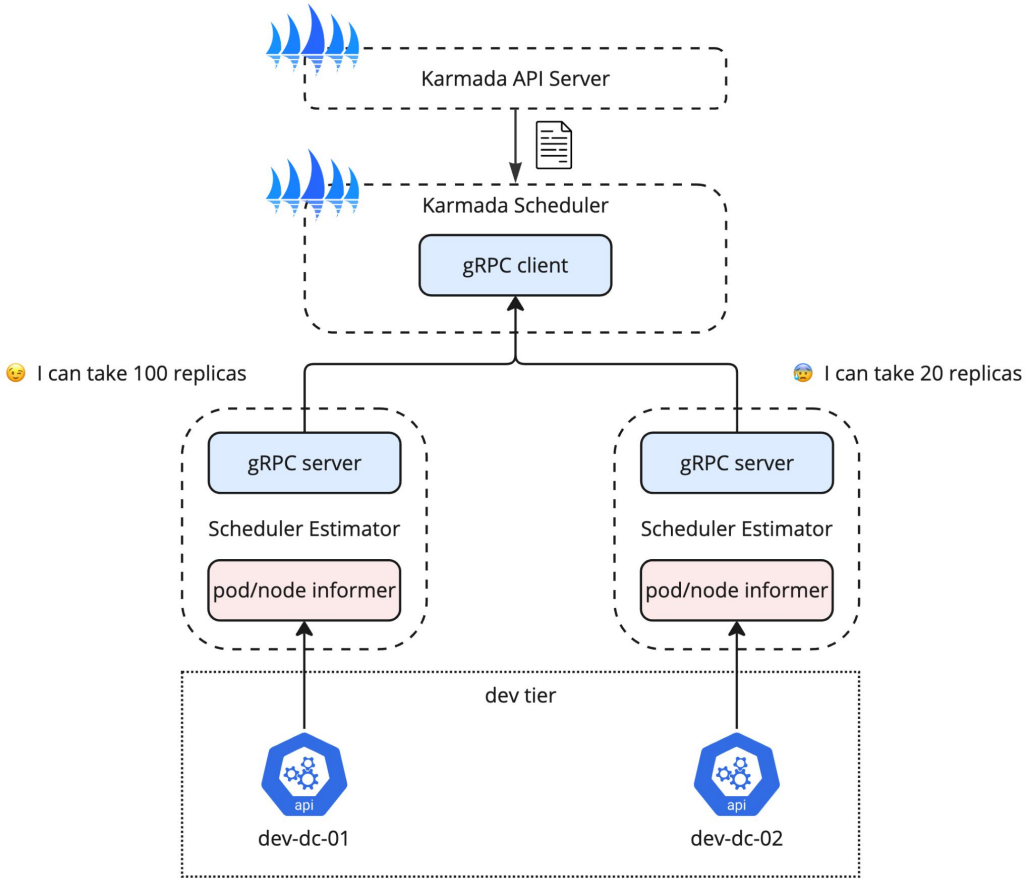
Scheduler Estimator

Provide accurate information for Karmada-Scheduler

`R1`: #replicas that a node can be divided into in terms of hardware resource
`R2`: the maximum remaining pods that the node allows

Amount of replicas that can be taken:
`sumByNode(min(R1, R2) per node)`

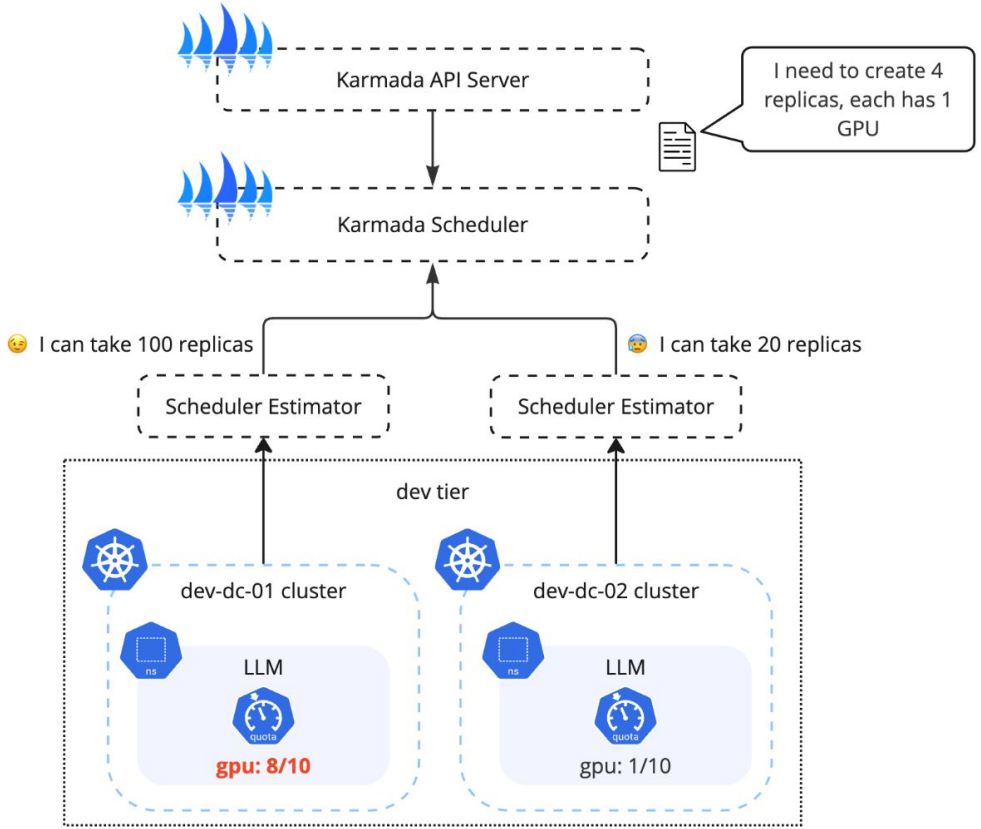
Proposal #521



Scheduler Estimator

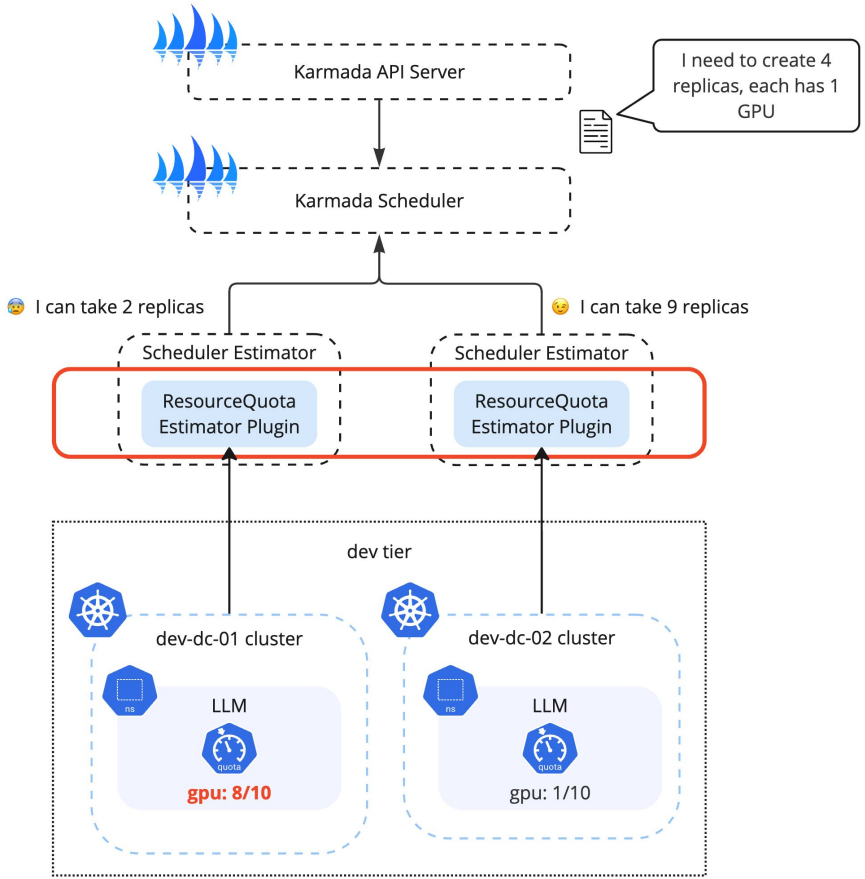
✅ Karmada chooses dc-01
More resources from cluster's perspective

⛔ insufficient quota in the LLM namespace



ResourceQuota Estimator Plugin

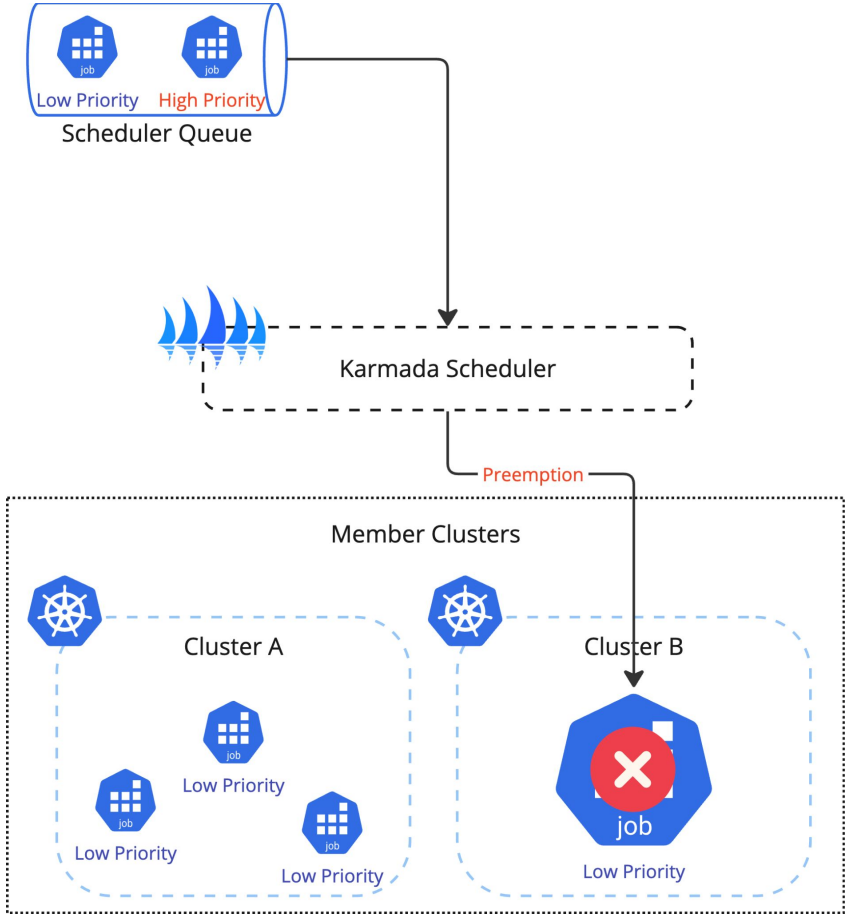
- Checks the remaining ResourceQuota in the target namespace
- Applies to PriorityClass-scoped ResourceQuota
- [PR #4566: [MVP] add resourcequota plugin in scheduler-estimator: add resourcequota plugin]



Future Roadmap

Priority & Preemption

- Ensure that critical workloads have preferential access to resources
- Ensure optimal resource utilization and availability for essential workloads
- [PR #4993: Propose binding priority and preemption]




KubeCon
CloudNativeCon
North America 2024
Thanks!
Questions?