



KubeCon



CloudNativeCon

Europe 2026

#KubeCon #CloudNativeCon

SIG-Multicluster Intro & Deep Dive

- What this SIG is about
- Roll Call of SIG Projects
- How to follow and contribute





KubeCon



CloudNativeCon

Europe 2026

What this SIG is about



What this SIG is about



KubeCon



CloudNativeCon

Europe 2026

- Solving common challenges related to the management of multiple Kubernetes clusters, and applications that exist therein.
- Multi-cluster is everywhere
 - Environment isolation; Tenancy; Multi cloud; Fault tolerance; Data locality; Policy; Capacity; Performance
- Multi cluster applications (split across multiple clusters) can be powerful
- Kubernetes was built with the Cluster as the Universe
 - We've made some progress, but there's a long way to go

Our approach



KubeCon



CloudNativeCon

Europe 2026

- Recognize diversity of the ecosystem
 - Where workloads run (cloud, on-prem)
 - Environmental constraints
 - Recognize the ecosystem when designing APIs – focus on APIs that work for everyone
 - Lift the standards existing or needed in the ecosystem
- Avoid solving any optional problems
 - “This could be useful” is an engineering fly trap!
- Consistency with existing APIs
- Composable building blocks

Project roll call!!!

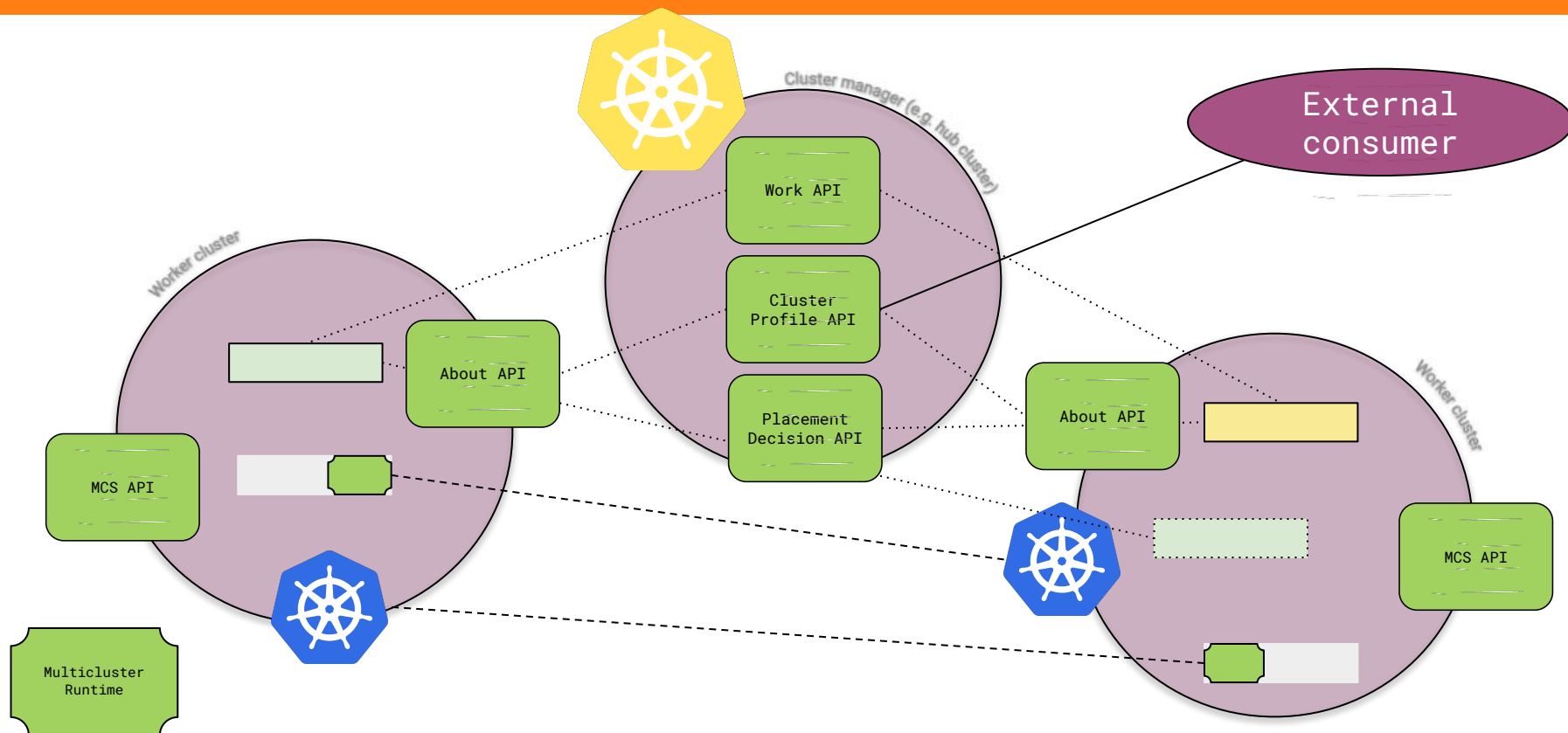


KubeCon



CloudNativeCon

Europe 2026





KubeCon



CloudNativeCon

Europe 2026

Placement Decision API



Expected flow for placements

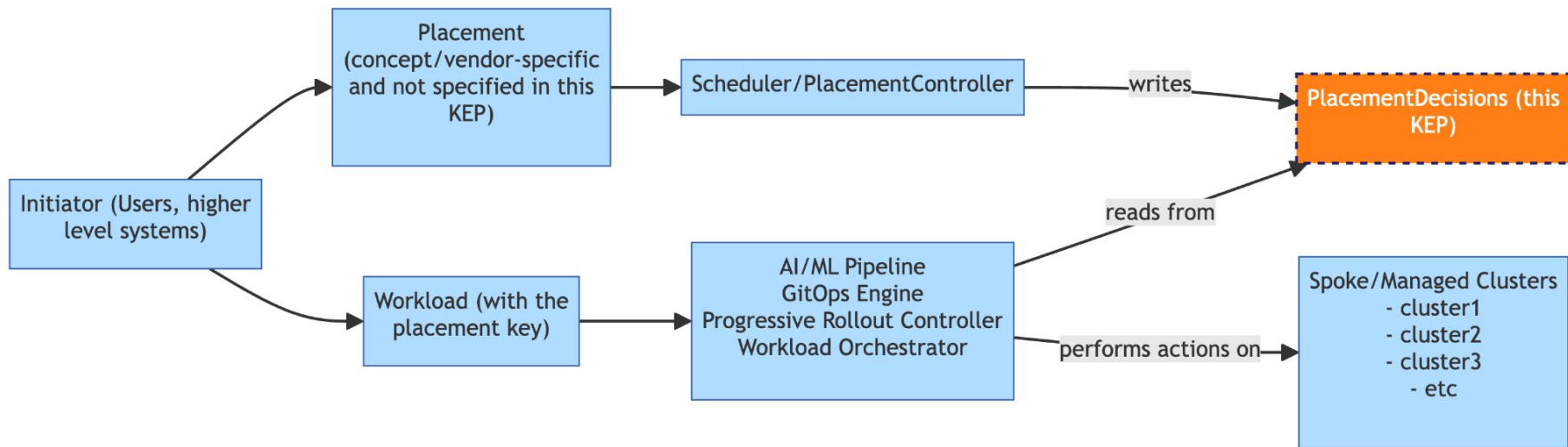


KubeCon



CloudNativeCon

Europe 2026



PlacementDecision API example



KubeCon



CloudNativeCon

Europe 2026

```
apiVersion: multicluster.x-k8s.io/v1alpha1
kind: PlacementDecision
metadata:
  name: app-placement-decision-0
  namespace: argocd
  labels:
    # Optional: present when the decision is tied to a workload
    multicluster.x-k8s.io/placement-key: "my-app"
    # Optional: if this logical decision spans multiple slices
    multicluster.x-k8s.io/decision-key: "argocd-app-placement-decision"
    # Optional: ordering hint when order matters across slices
    multicluster.x-k8s.io/decision-index: "0"
schedulerName: multicluster-placement-controller
decisions:
- clusterProfileRef:
  apiVersion: multicluster.x-k8s.io/v1alpha1
  kind: ClusterProfile
  namespace: fleet1
  name: cluster1
  reason: "GPUs available"
- clusterProfileRef:
  apiVersion: multicluster.x-k8s.io/v1alpha1
  kind: ClusterProfile
  namespace: fleet1
  name: cluster2
  reason: "GPUs available"
```

Story 1: GPU-aware AI training

- Initiator: ML platform / pipeline for a specific training job
- Workload: PyTorch training job requiring 8 GPUs

Story 2: Progressive rollout

- Initiator: Progressive rollout controller managing canary deployment of my-service v2.0
- Workload: Kubernetes Deployment with new version (release: "v2.0-canary")

Use cases



KubeCon



CloudNativeCon

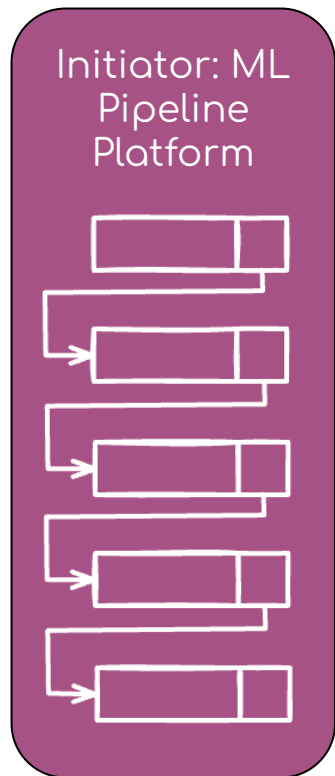
Europe 2026

Story 1: GPU-aware AI training

- Initiator: ML platform / pipeline for a specific training job
- Workload: PyTorch training job requiring 8 GPUs

Story 2: Progressive rollout

- Initiator: Progressive rollout controller managing canary deployment of my-service v2.0
- Workload: Kubernetes Deployment with new version (release: "v2.0-canary")



Labels the manifest

```
multicloud.x-k8s.io/placement-key  
= "training-job-resnet50-123"
```

```
apiVersion: batch/v1  
kind: Job  
metadata:  
  name: training-job  
spec:  
  template:  
    # [ . . . ]
```

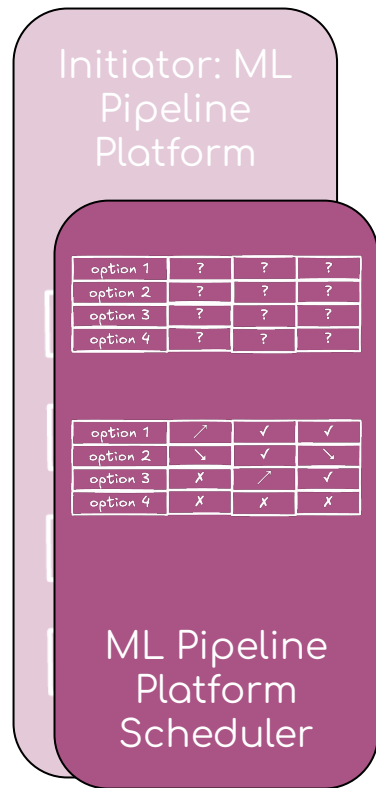
Creates the PlacementRequest

(API not defined)

Cluster manager (e.g. hub cluster)

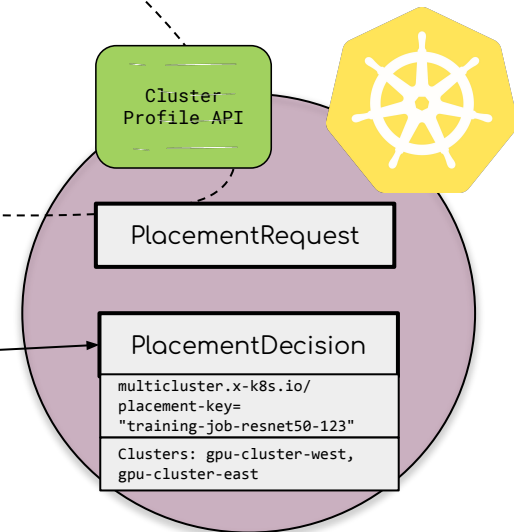
PlacementRequest

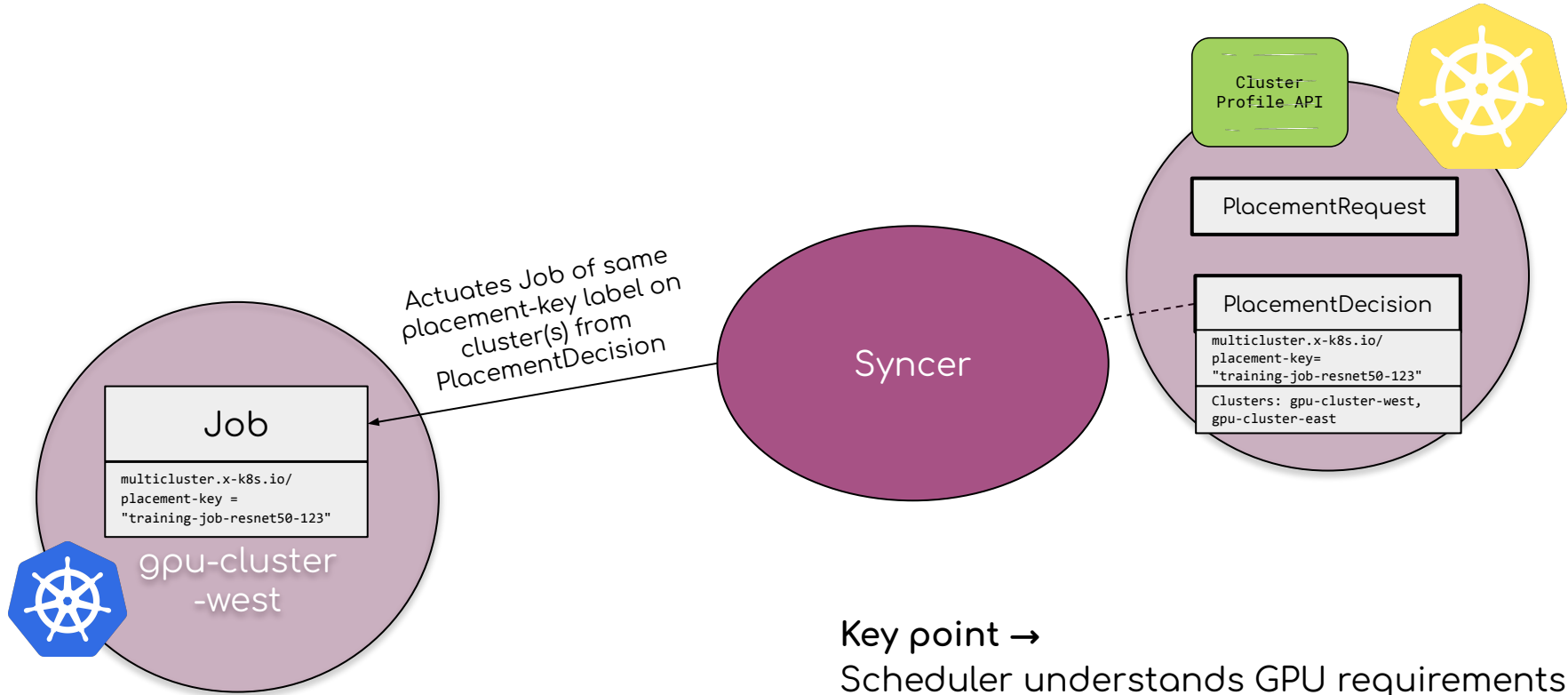
Cluster
Profile API



Scores ClusterProfiles in ClusterInventory based on PlacementRequest

Creates PlacementDecision with same placement-key label and with decided clusters





Key point →

Scheduler understands GPU requirements (workload-aware), consumer ("syncer") just reads "deploy to these clusters"

Use cases



KubeCon



CloudNativeCon

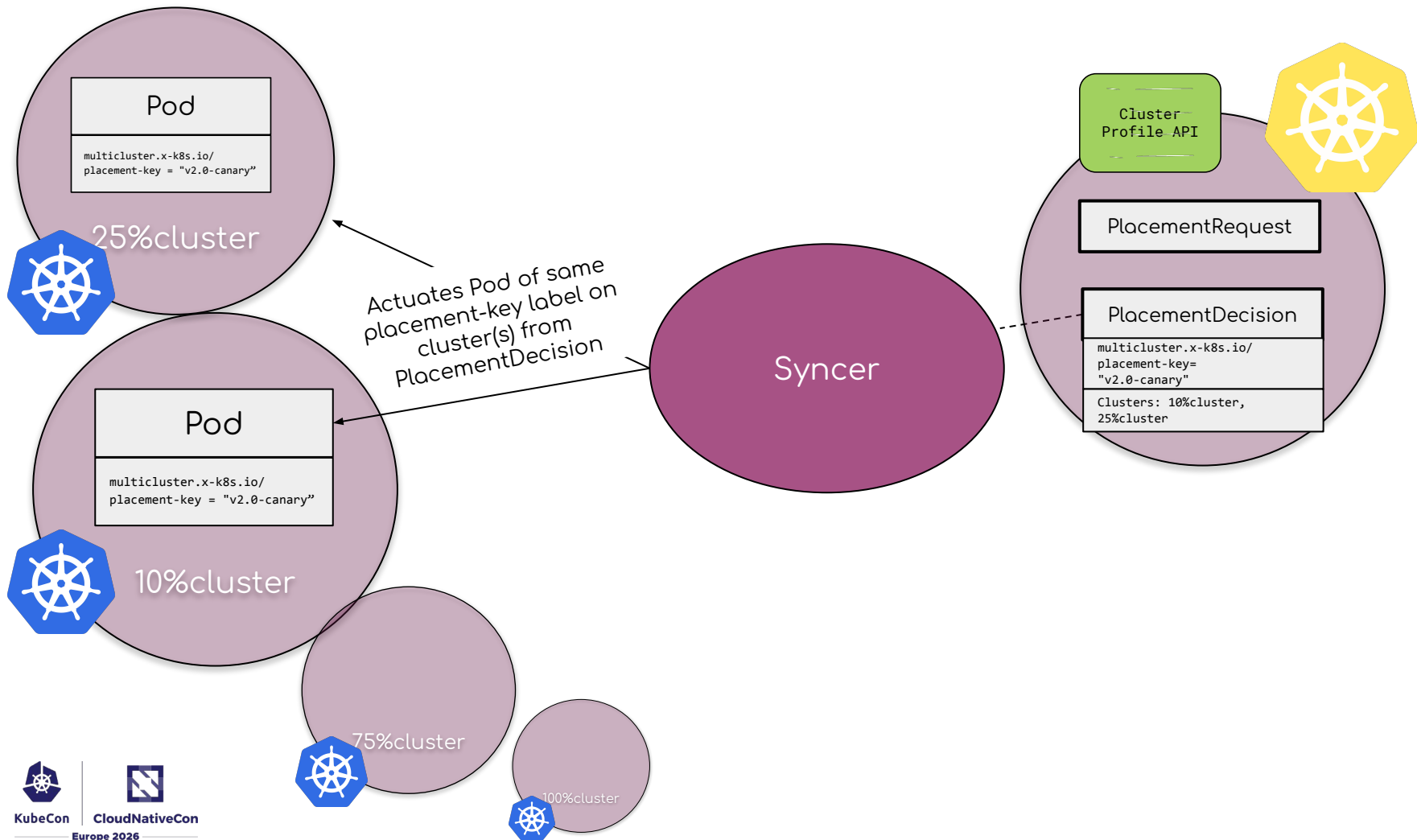
Europe 2026

Story 1: GPU-aware AI training

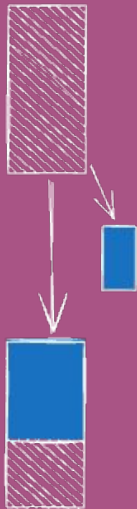
- Initiator: ML platform / pipeline for a specific training job
- Workload: PyTorch training job requiring 8 GPUs

Story 2: Progressive rollout

- Initiator: Progressive rollout controller managing canary deployment of my-service v2.0
- Workload: Kubernetes Deployment with new version (release: "v2.0-canary")



Initiator:
Progressive
rollout
controller



10%cluster
/metrics

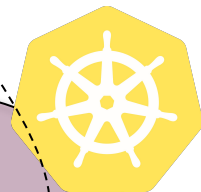


25%cluster
/metrics

Determines progressing rollout is safe
and identifies next candidate clusters

****Updates**** existing PlacementDecision
with same placement-key label
with more clusters

Cluster
Profile API



PlacementRequest

PlacementDecision

multicluster.x-k8s.io/
placement-key=
"v2.0-canary"

Clusters: 10%cluster, 25%
cluster, 75%cluster

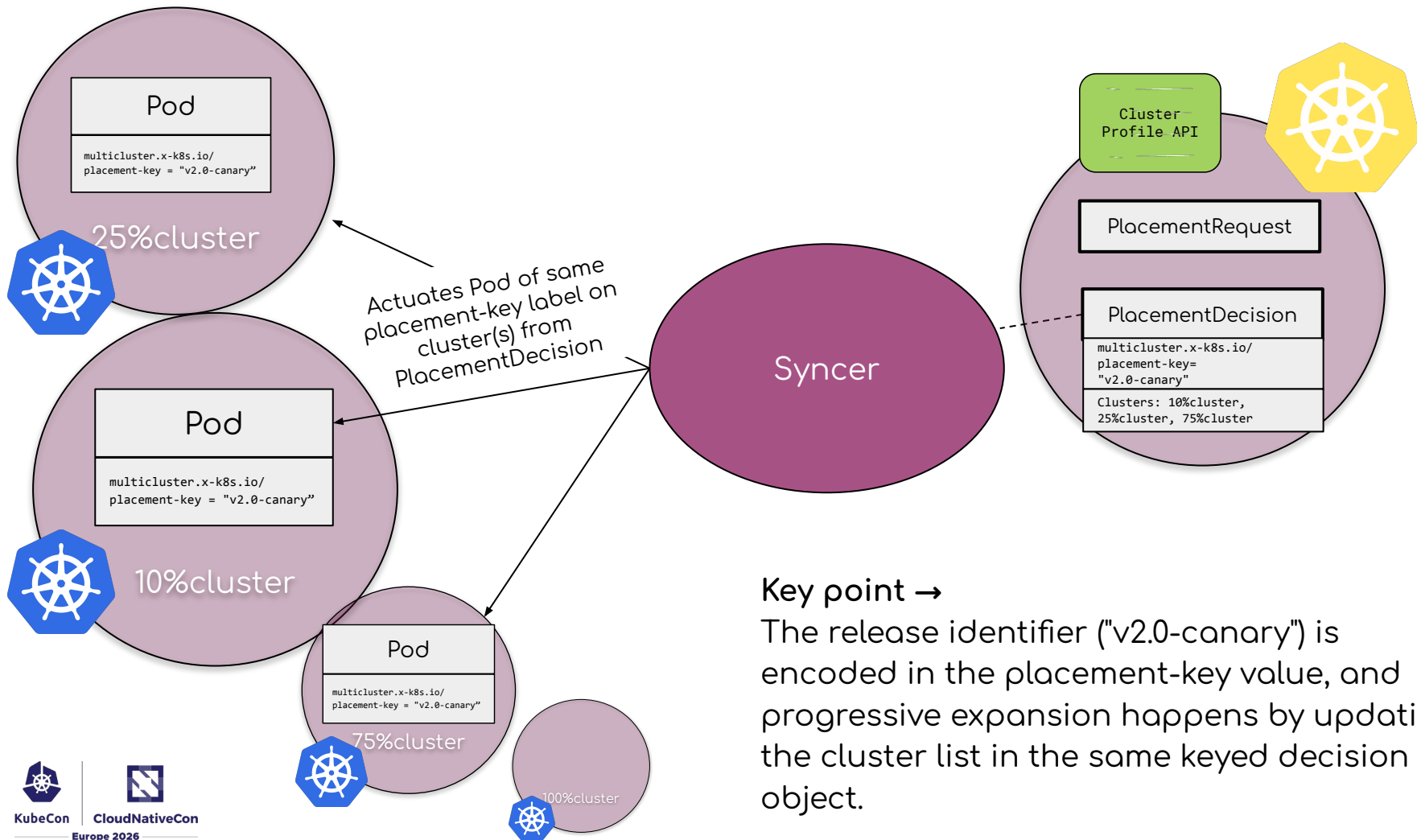


KubeCon



CloudNativeCon

Europe 2026





KubeCon



CloudNativeCon

Europe 2026

Multicluster UXR



About the User Research



KubeCon



CloudNativeCon

Europe 2026

The Problem: Deploying, managing and monitoring multiple Kubernetes clusters is challenging. Our user research aims to capture engineers' experiences and expectations with multi-cluster setups.

Goal: Make complex systems easy to use, Turn difficult operations into simple human friendly.

User Research Method: Survey

Survey Design: Engineers working with multi clusters | Problem areas: Environment, Tooling, observability, troubleshooting, automation, collaboration | Qualitative & Quantitative

Survey Reach: Kubernetes SIG | 170 respondents



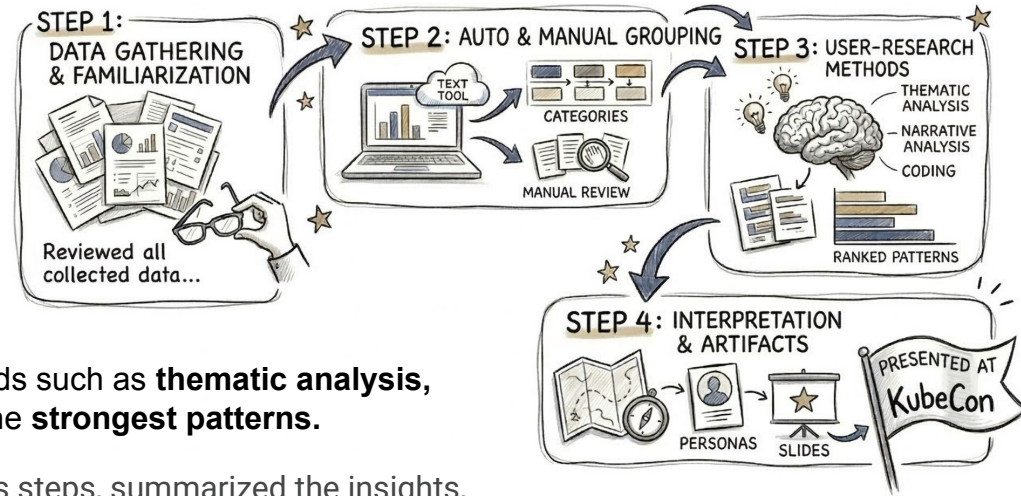
Process & Methods for qualitative data analysis

Step 1: Reviewed all **collected data** to understand and **familiarize ourselves with the responses**.

Step 2: We used a **text-analysis tool** by pasting all responses to automatically group them into categories and identify the most frequent ones. We then **manually reviewed** each response to ensure accurate grouping.

Step 3: Analyzed the data using user-research methods such as **thematic analysis**, **narrative analysis**, and **coding** to identify and rank the **strongest patterns**.

Step 4: We **interpreted the results** from the previous steps, summarized the insights, and organized them into **final artifacts** (e.g., personas, slides) presented at KubeCon



Text Analyser Tools: <https://www.online-utility.org/text/analyzer.jsp>, <https://chatgpt.com/g/g-86OiPigfl-ai-text-analyzer>

From Survey Insights to Personas



KubeCon



CloudNativeCon

Europe 2026

Key Insights	Recommendations
1. Documentation gaps	→ Provide clear up to date documentation with use cases.
2. Centralized monitoring	→ Implement centralized observability solution.
3. AI-driven automation	→ Leverage AI driven automation on tasks.
4. Operational complexity	→ Reduce complexity, provide training.
5. Developer UX	→ UI/UX & Developer Experience Improvement Plan.

We conducted a survey on how engineers manage and monitor multiple Kubernetes clusters, revealing key challenges, workflows, and expectations. See the KubeCon + CloudNativeCon North America 2025 slides and video for reference: <https://sched.co/27Nlj>

In this second phase, we synthesized these insights to create an **Engineer Persona & Journey**, which helps translate research findings into a clear representation of the target user and their goals.

Persona



KubeCon



CloudNativeCon

Europe 2026

- A **user persona** is a research-based representation of a user group built from patterns in real data, interviews, and observations
- The persona helps us identify the **right starting point** for engineers **adopting multi-cluster setups**.
- It provides a **clear, research-backed** representation of real engineers, replacing assumptions about an “**imagined**” user and enabling project discussions and decisions to be based on real insights, ensuring we are **solving genuine problems for engineers**.
- Personas guide our team to focus on the **most impactful challenges engineers face in multi-cluster** environments. They help evaluate: **Does this feature support real workflows? Are we improving the developer experience in the right areas?**
- Documentation can be structured around engineers’ actual workflows, tasks, and goals. Since persona will be useful as a **decision making tool**.

USER PERSONA TEMPLATE (SKETCHED!)

1. PERSONA SKETCH  Name: _____ Age: _____ Job Title: _____		4. BIO/BACKGROUND & STORY  _____ _____	
2. KEY GOALS • To achieve [Goal 1] _____ • To feel [Goal 2] _____ • To find [Goal 3] _____		5. SKILLS & BEHAVIORS Tech Savvy (low-high) _____ Attention to Detail _____ Decision Speed _____	
KEY GOALS & MOTIVATIONS		6. PREFERRED CHANNELS & TOUCHPOINTS    	
FRUSTRATIONS & PAIN POINTS     		7. TECHNOLOGY STACK    Write to: _____ Browser Spreadsheet Industry software	
FRUSTRATIONS & PAIN POINTS  Time-consuming... _____  Expensive! _____  Confusing interface... _____		8. THE IDEAL EXPERIENCE         	

Journey mapping



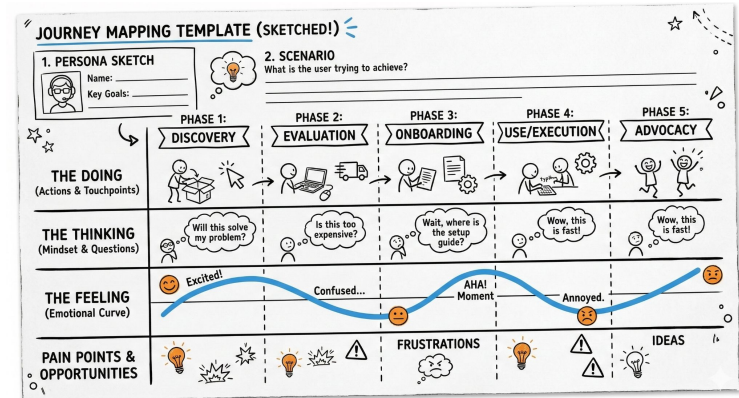
KubeCon



CloudNativeCon

Europe 2026

- A **User Journey Map** visualizes the steps a user takes while completing a task or interacting with a system.
- It highlights the user's **actions, goals, pain points, and experiences** throughout the **workflow**.
- engineers, and product managers can all understand how users interact with the system.
- Journey maps help teams understand how users experience a product over time and **reveal opportunities to improve the overall experience**.
- Product teams use them to decide **which problems to solve first**.
- They help explain research insights in a **clear story format**.
- They ensure engineers, designers, and product managers understand **how users actually work**.



Step by step process

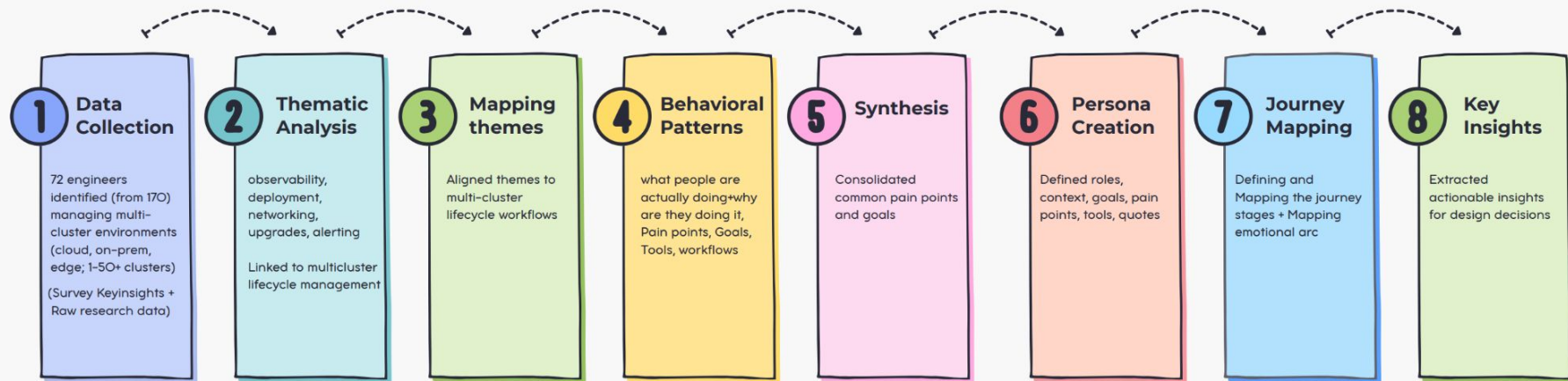


KubeCon



CloudNativeCon

Europe 2026



Hybrid Engineer Persona



KubeCon



CloudNativeCon

Alex is a **hybrid engineer** who **deploys, manages, and maintains observability** across multiple Kubernetes clusters on-prem, in edge locations, and across cloud environments.

	Scenario: The journey of an engineer deploying 16 workloads across multiple Kubernetes clusters spanning AWS, GKE, Azure, on-premises data centers, and edge locations		
Hybrid Engineer		Pain points	lack of purpose-built multi-cluster tooling, repetitive manual work, and the day-to-day operational burden of managing multiclusters
			Limited skilled engineers, shallow vendor support, and the high barrier to entry that leaves teams isolated during complex issues.
			On-prem vs. cloud architecture differences, config inconsistencies, identity system mismatches, and network/sync challenges across environments.
Demographics	Platform Engineer, SRE Engineer, Software developer, Cloud engineer, DevOps Engineer		Cross-cluster visibility gaps, no unified observability view, and the manual triage burden when alerts fire.
			Managing and auditing access across clusters and environments, RBAC complexity at scale, identity federation challenges, and secure baseline provisioning.
Approach	(Primary) A platform team owns the clusters, gives app teams a namespace (single or multi-cluster), and app teams deploy via Git repositories using allowed resources (Secondary) A centralized infra team manages the clusters, app teams get a folder in a Git repo, and deploy using CRDs provided by the platform team		
		Goals	Enable purpose-built and automation-driven multi-cluster tooling with AI-powered self-service and intuitive developer experience to reduce operational burden and cognitive load.
			Build accessible support systems and simplified workflows that reduce reliance on limited skilled engineers, strengthen vendor support, and lower the barrier to managing complex issues."
			Ensure consistent architecture, configuration, identity management, and seamless network synchronization across on-prem and cloud environments.
Primary Use-case	Different environments for different stages of software development life cycle and Environment separation. Production and non-production Distribute workloads across clusters and regions to align with fault domains, improve robustness, ensure high availability, and maintain uninterrupted services during Kubernetes, storage, or network upgrades.		Provide unified cross-cluster visibility and centralized observability to reduce manual triage when alerts occur
			Simplify and standardize access management across clusters and environments by streamlining RBAC, improving identity federation, and enabling secure baseline provisioning at scale."

Google Sheets version at bit.ly/sig-mc-hybrid-eng-persona

Pain points:

- Lack of purpose-built multi-cluster tooling, repetitive manual work, and the day-to-day operational burden of managing multi clusters
- Limited skilled engineers, shallow vendor support, and the high barrier to entry that leaves teams isolated during complex issues.
- On-prem vs. cloud architecture differences, config inconsistencies, identity system mismatches, and network/sync challenges across environments.
- Cross-cluster visibility gaps, no unified observability view, and the manual triage burden when alerts fire.
- Managing and auditing access across clusters and environments, RBAC complexity at scale, identity federation challenges, and secure baseline provisioning.

Their **primary use of multi-cluster** setups is to separate environments (production vs. non-production) and distribute workloads across clusters and regions for fault isolation.

Hybrid Engineer Persona



KubeCon



CloudNativeCon

Europe 2026

How These Personas Are Useful

- The persona helps identify the **right starting point for engineers adopting multi-cluster setups**.
- The persona provides a **clear, research-backed representation of engineers**, replacing the **idea of an imagined user**.
- Discussions and decisions about the project can be based on real engineer research insights rather than assumptions.
- The persona helps teams focus on **solving the most impactful problems engineers face in multi-cluster environments**.
- Documentation can be organized around engineers' real workflows, tasks, and goals.

Goals:

- Enable purpose-built and **automation-driven multi-cluster tooling** with **AI-powered self-service** and **intuitive developer experience** to **reduce operational burden** and **cognitive load**.
- Build accessible support systems and **simplified workflows** that reduce reliance on limited skilled engineers, **strengthen vendor support**, and **lower the barrier to managing complex issues**."
- Ensure consistent **architecture, configuration, identity management**, and **seamless network** synchronization across **on-prem and cloud** environments.
- Provide **unified cross-cluster visibility** and **centralized observability** to **reduce manual triage** when alerts occurs.
- Simplify and **standardize access management** across clusters and environments by streamlining RBAC, improving identity federation, and enabling secure baseline provisioning at scale."

Hybrid Engineer Journey

Key insights obtained from Hybrid engineer mapping:

- Engineers deploying workloads across multi-cluster environments often start with **confidence**, believing they can manage the **complexity** with familiar **tools and architecture, methods, frame work, documentation**.
- The reality of cross-cluster operations — **Spending hours on deploying and managing and upgrading multi-clusters, on prem and Cloud architecture difference, network fragmentation, and configuration sprawl, observability, complex UI/UX tooling, manual triage while critical alerts occur** — reveals itself gradually.
- The turning point comes when teams quantify the operational burden and **adopt unified management approaches** with automation, transforming chaos into a scalable, maintainable architecture.

Google Sheets version at bit.ly/sig-mc-hybrid-eng-persona

Hybrid Engineer

Alex's multi-cluster journey

1

Planning & deployment

16 clusters across 5 environments — GKE, EKS, Azure, on-prem, and edge

Challenges

Complex configs, access control, networking, constant context switching, and cloud vs on-prem inconsistencies

2

Monitoring & observability

Responsible for health and performance across all clusters

Challenges

Fragmented tooling and poor UI/UX make it hard to get a unified view of system health

3

Lifecycle management & upgrades

Version upgrades, workload bursting, and cross-cluster failovers

Challenges

- Manual, repetitive changes across clusters, complex migrations, and alert investigation increase error risk and burnout
- Workload bursting between cloud and on-prem adds operational complexity and unpredictability

4

Alex's vision

What multi-cluster ops should feel like

Goals

- Unified observability and networking across all clusters
- Consistent, standardized workflows for hybrid and multi-cloud
- Better documentation with real-world examples
- Training for engineers on multi-cluster operations
- No more constant tool switching or manual coordination



Sad



Difficult



Overwhelmed



Frustrated



Ideate



Possible Outcome

Join us for User Interviews!

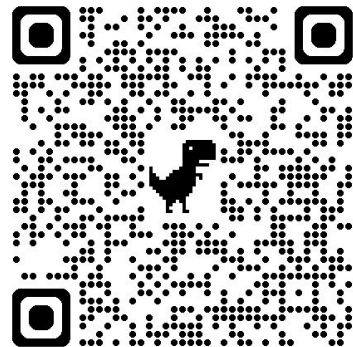
We are conducting user research to continue to understand engineers' real challenges in multi-cluster operations.

Why Participate

- Share your multi-cluster management experience.
- Help identify gaps and opportunities for better solutions.

Interview Info

- 45–60 minute interview, optional screen sharing.
- Participation is voluntary and all responses will be anonymized.
 - a. More research information at:
<https://github.com/pavanipriyasajja/SIG-MC-User-Research>





KubeCon



CloudNativeCon

Europe 2026

Follow and get involved



Be part of the SIG



KubeCon



CloudNativeCon

Europe 2026

- What are you working on?
 - ◇ APIs you're building, concepts you're missing — agenda is always open for demos!
- What problems do you or your customers have?
- Do you have unique needs that have been overlooked?
- Do you want to help?
 - ◇ Contribute to the test and conformance suites
 - ◇ Add information to the web site
 - ◇ Join the UX team for more research and analysis
 - ◇ Provide stakeholder input on KEPs
 - ◇ Try our projects

Be part of the SIG



KubeCon



CloudNativeCon

Europe 2026

Home page: <https://multicluster.sigs.k8s.io>

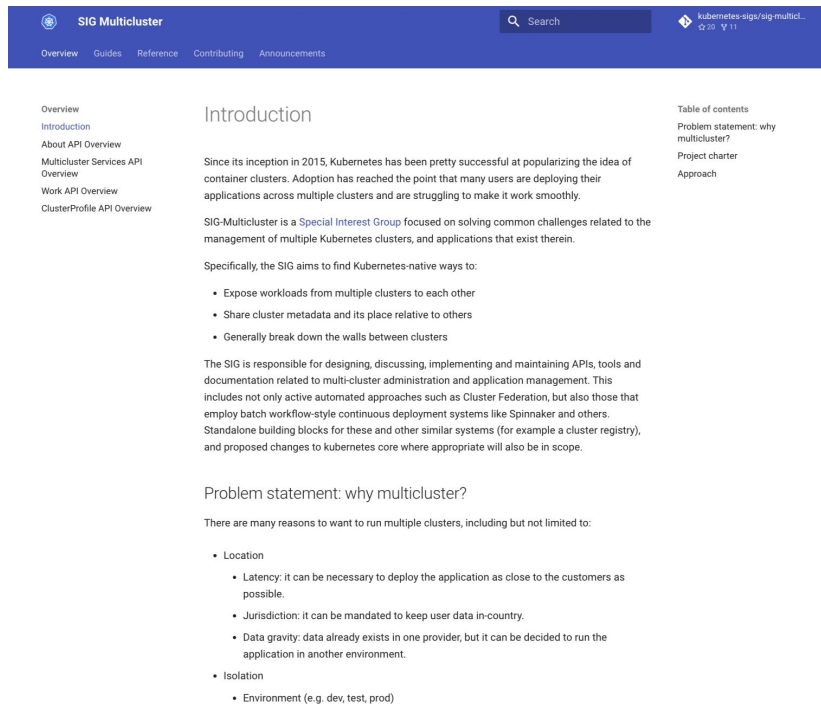
Slack channel:

<https://kubernetes.slack.com/messages/sig-multicluster>

List:

<https://groups.google.com/forum/#!forum/kubernetes-sig-multicluster>

Meetings are weekly Tuesdays, 12:30 Eastern,
9:30 Pacific, 16:30 UTC (AMER/EMEA timeslot)
and biweekly Thursdays at 7 AM UTC
(APAC/EMEA timeslot)



The screenshot shows the SIG Multicluster website. The header is blue with the SIG Multicluster logo and navigation links: Overview, Guides, Reference, Contributing, and Announcements. A search bar is on the right. The main content area has a left sidebar with links: Overview, Introduction, About API Overview, Multicluster Services API Overview, Work API Overview, and ClusterProfile API Overview. The main content area is titled 'Introduction' and contains text about the SIG's purpose and goals. A right sidebar contains links: Table of contents, Problem statement: why multicluster?, Project charter, and Approach.

Introduction

Since its inception in 2015, Kubernetes has been pretty successful at popularizing the idea of container clusters. Adoption has reached the point that many users are deploying their applications across multiple clusters and are struggling to make it work smoothly.

SIG-Multicluster is a [Special Interest Group](#) focused on solving common challenges related to the management of multiple Kubernetes clusters, and applications that exist therein.

Specifically, the SIG aims to find Kubernetes-native ways to:

- Expose workloads from multiple clusters to each other
- Share cluster metadata and its place relative to others
- Generally break down the walls between clusters

The SIG is responsible for designing, discussing, implementing and maintaining APIs, tools and documentation related to multi-cluster administration and application management. This includes not only active automated approaches such as Cluster Federation, but also those that employ batch workflow-style continuous deployment systems like Spinnaker and others. Standalone building blocks for these and other similar systems (for example a cluster registry), and proposed changes to Kubernetes core where appropriate will also be in scope.

Problem statement: why multicluster?

There are many reasons to want to run multiple clusters, including but not limited to:

- Location
 - Latency: it can be necessary to deploy the application as close to the customers as possible.
 - Jurisdiction: it can be mandated to keep user data in-country.
 - Data gravity: data already exists in one provider, but it can be decided to run the application in another environment.
- Isolation
 - Environment (e.g. dev, test, prod)



KubeCon



CloudNativeCon

Europe 2026

Thank you!



<https://sched.co/2EF5b>

