



KubeCon



CloudNativeCon

Europe 2026

#KubeCon #CloudNativeCon

DRAmatically simple

On demand GPUs for MLOps





Who we are



Doug Smith

- MLOps Engineer (currently) working on vLLM (it's an inference server)
- With a long history with containerization, especially for networking workloads!
- Network Plumbing Working Group Emeritus
- Blog: <https://dougbtv.com/>



Miguel Duarte

- OpenShift Virt network engineer
- Network Plumbing Working Group member
- Blog: <https://maiqueb.github.io>

Agenda



KubeCon



CloudNativeCon

Europe 2026

- **Intro**
 - Motivation
 - Device plugins
 - DRA
 - The RPG party: Get to know the characters
 - DRA specific features
- **Hands on tutorial**
 - Requirements
 - Problem statements
 - Walk through



KubeCon



CloudNativeCon

Europe 2026

Motivation



Motivation



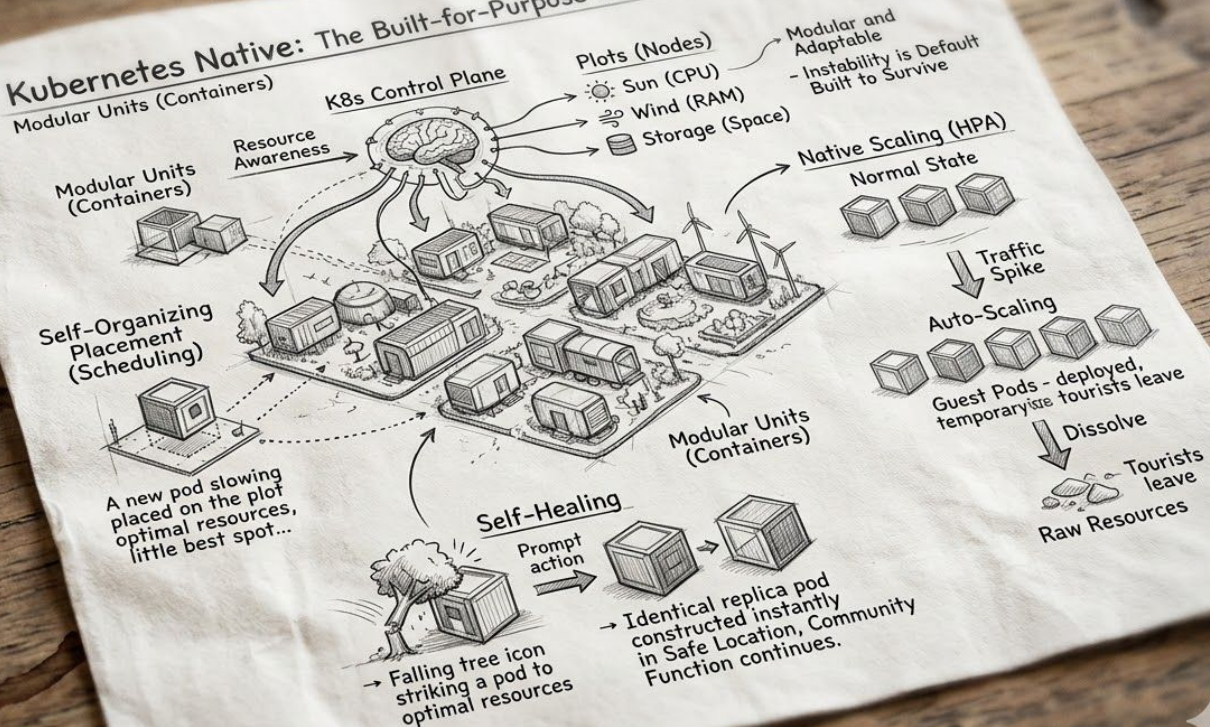
KubeCon



CloudNativeCon

Europe 2026

Kubernetes Native: The Built-for-Purpose Eco-Village



Semantics

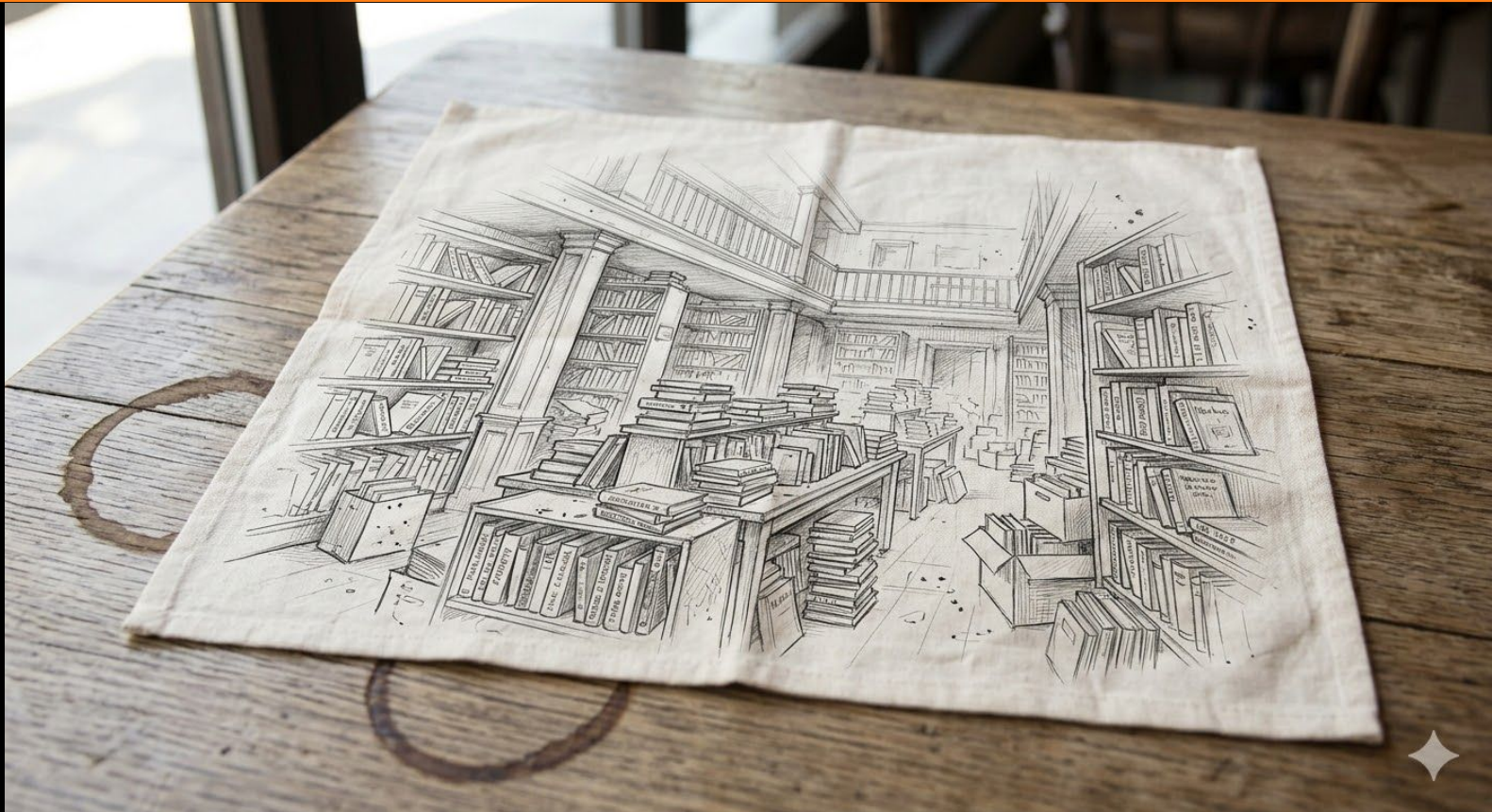


KubeCon



CloudNativeCon

Europe 2026



Resource Efficiency

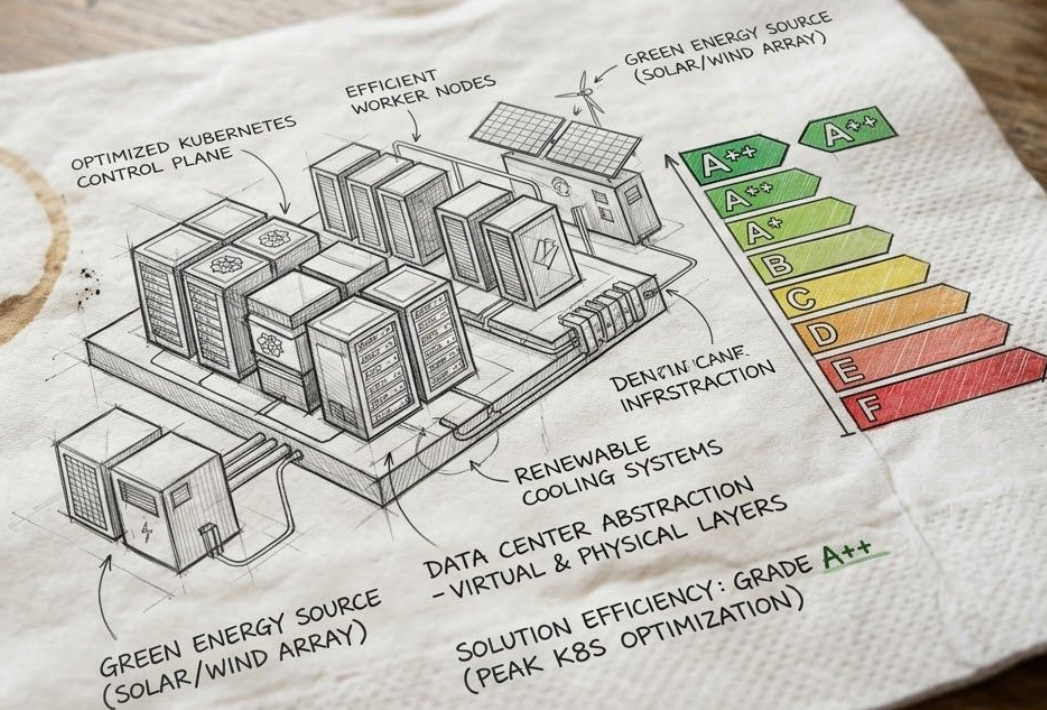


KubeCon



CloudNativeCon

Europe 2026



Simpler Management & Administration

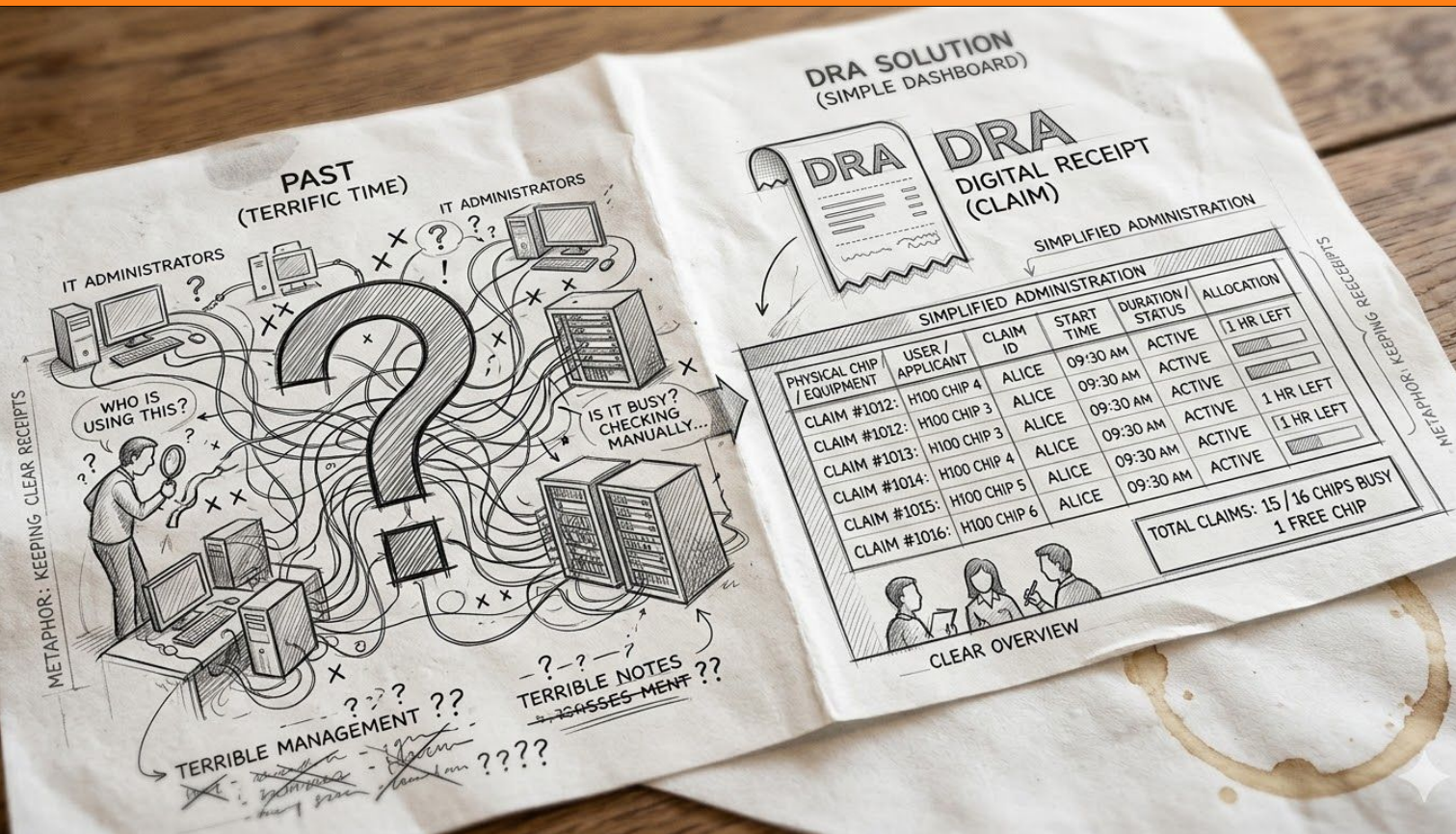


KubeCon



CloudNativeCon

Europe 2026





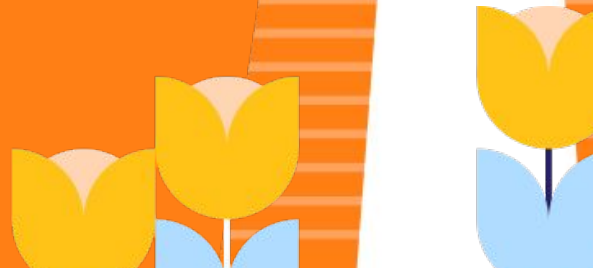
KubeCon



CloudNativeCon

Europe 2026

Device Plugins



Device Plugins



KubeCon



CloudNativeCon

Europe 2026



Device Plugins

(introduced ~Kubernetes 1.8)

- Original Kubernetes GPU integration
- Works great for **whole device allocation**
- Scheduling tied to **nodes and labels**
- Feature discovery often requires **custom labeling**

Operational Reality

- Node labels
- nodeSelectors
- Device plugin configs
- Cluster-specific conventions



Dynamic Resource Allocation (DRA) (Now GA in Kubernetes 1.35)

- **Capability-based resource requests**
- Request **features**, not nodes

Built-in support for:

- GPU partitioning (MiG)
- Complex device topologies
- Vendor-specific parameters



KubeCon



CloudNativeCon

Europe 2026

DRA



DRA in a nutshell



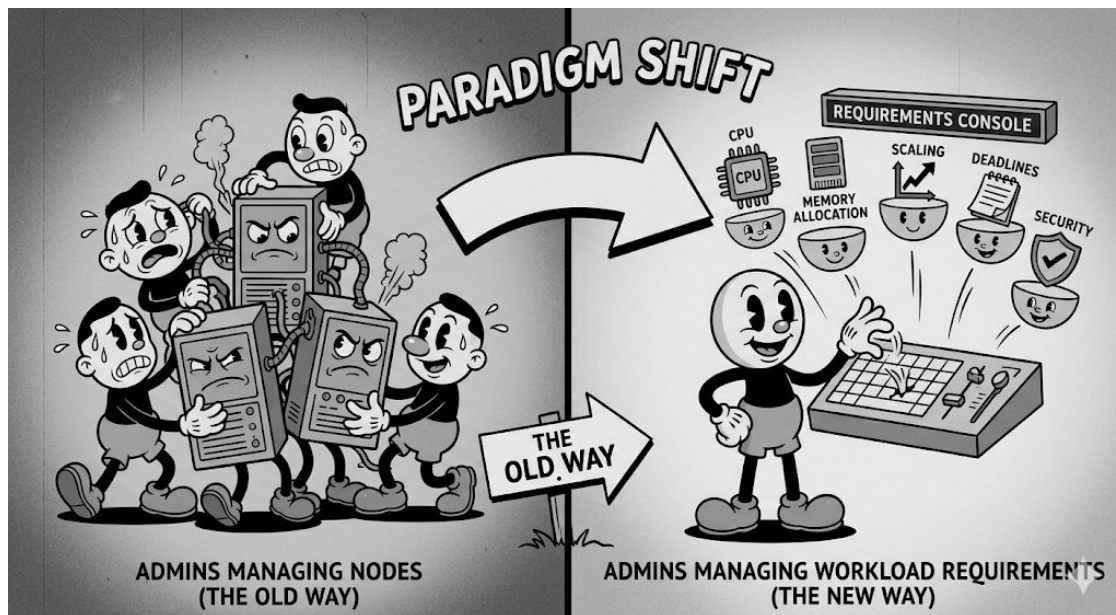
KubeCon



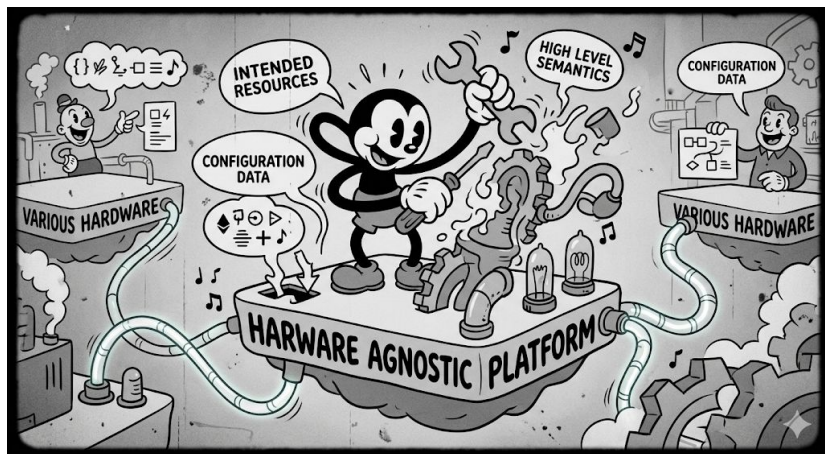
CloudNativeCon

Europe 2026

Start managing requirements



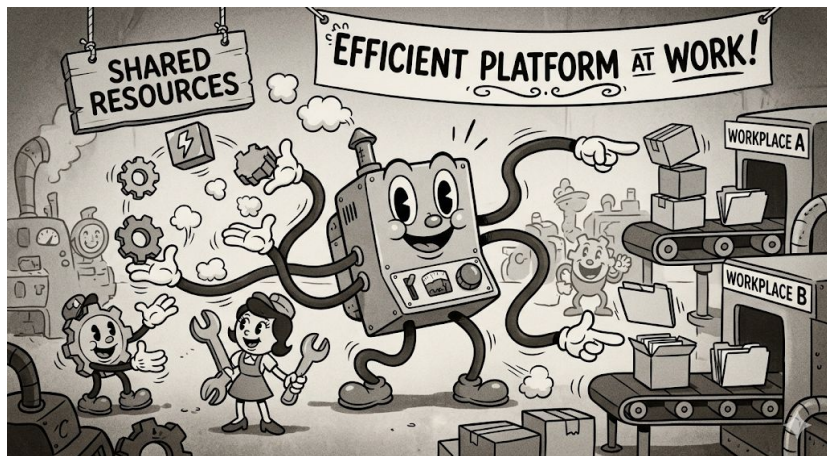
Hardware agnostic



- CEL expressions
 - Request capabilities, not HW models
- Scheduler integrated

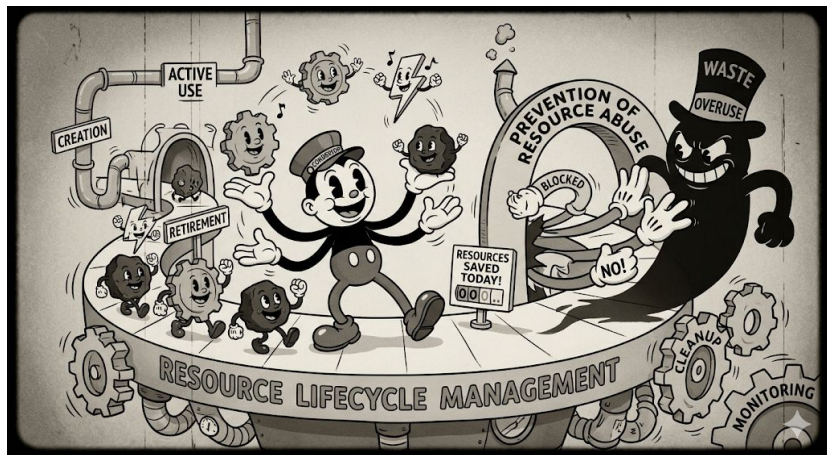
`device.attributes["memory"] >= 24000`
would match *either* a A100, H100, or T4 GPUs

Efficient resource usage



- Dedicated features
 - **Consumable capacity**
 - resource sharing
 - **Influence placement**
 - improve performance & resiliency

Managed lifecycle Resource abuse prevention



- Resources are decoupled from the Pods
 - A claim can:
 - exist before a Pod starts
 - persist after it dies
- Resource requests are subject to quotas
- Users don't request *resources*
 - they request a curated abstraction created by the admin

Superior Observability Simplify the admin's life



- Transparent state
 - Status sub-resource
- Precise error reporting
- Device Health monitoring

Three different actor, in different stages



- Separation of concerns
- Standardized workflow
- Three different, required roles
 - DRA driver
 - Admin
 - App developer



KubeCon



CloudNativeCon

Europe 2026

The Kubernetes DRA Driver Developer



DRA Driver Developer



KubeCon



CloudNativeCon

Europe 2026



Goals:

- Stocking the inventory
 - Introspect the nodes
 - Generate / publish **ResourceSlices**
- Define Attributes / Capacity
 - Decide which attributes to expose, e.g.
 - Firmware versions / memory definitions / topology info
- Implement the DRA gRPC API
 - NodePrepareResources
 - NodeUnprepareResources

Driver => ResourceSlice

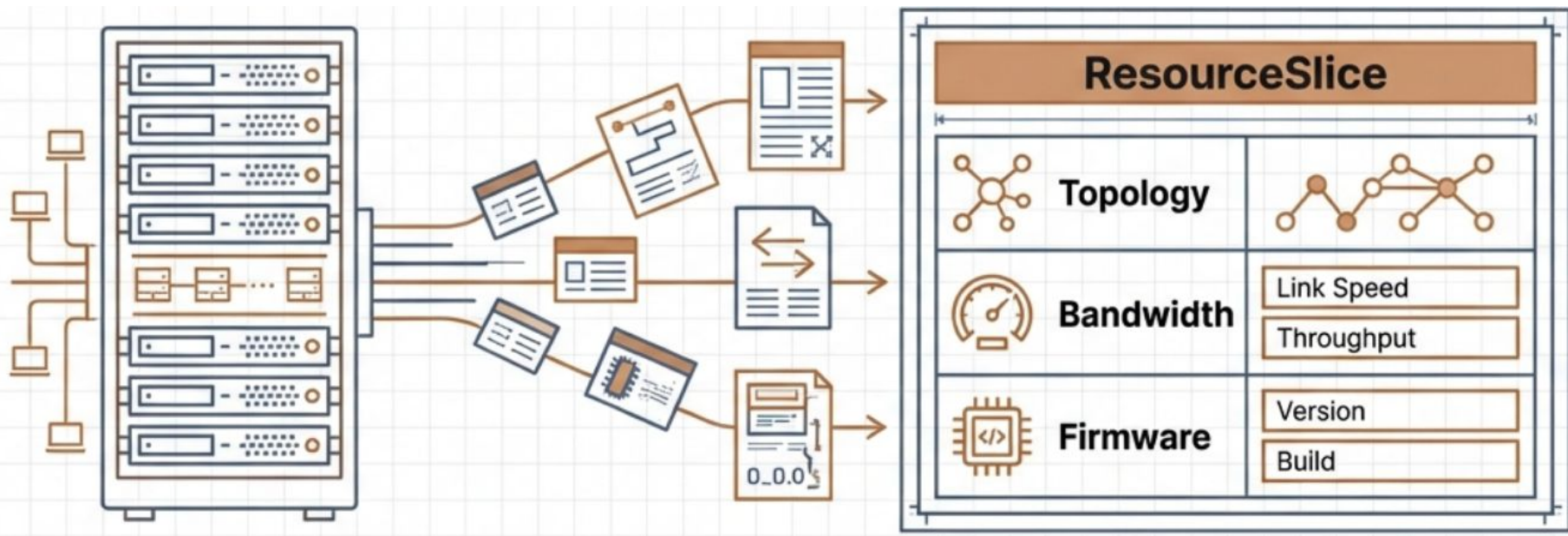


KubeCon



CloudNativeCon

Europe 2026





KubeCon



CloudNativeCon

Europe 2026

The Kubernetes Admin



Kubernetes Admin Role



KubeCon



CloudNativeCon

Europe 2026



Goals:

- Curate the “menu”
 - Create **DeviceClasses**
 - This is the resource “users” select
- Audit existing inventory
 - Inspect **ResourceSlices**
- Monitor / Troubleshoot “orders”
 - Track usage by listing **ResourceClaims**

Admin defines DeviceClass



KubeCon



CloudNativeCon

Europe 2026

- **DeviceClass**
 - Abstracts HW details
 - Policy definition
- This is what the claim requests
 - “Users” can only request admin-curated resources



Kubernetes Admin curating DeviceClass

DeviceClass example



KubeCon



CloudNativeCon

Europe 2026

```
apiVersion: resource.k8s.io/v1alpha3
kind: DeviceClass
metadata:
  name: high-end-training
spec:
  # 1. SELECTORS: Define which hardware fits this class using CEL
  selectors:
    - cel:
        expression: device.attributes["nvidia.com"].memory >= 40000

  # 2. CONFIG: Optional parameters passed to the driver upon allocation
  config:
    - opaque:
        driver: gpu-driver.nvidia.com
        parameters:
          # Example: Enforce a specific sharing strategy or driver setting
          sharingStrategy: "TimeSlicing"
          computeMode: "Exclusive"
```



KubeCon



CloudNativeCon

Europe 2026



The application developer



K8s User role



KubeCon



CloudNativeCon

Europe 2026



Goals:

- Select a **DeviceClass**
- Define the **ResourceClaim**
 - Specify attributes (CEL)
 - Attributes
 - e.g. "VRAM > 24GB"
 - Capacity
 - Configure the driver

User requests resources for the pod

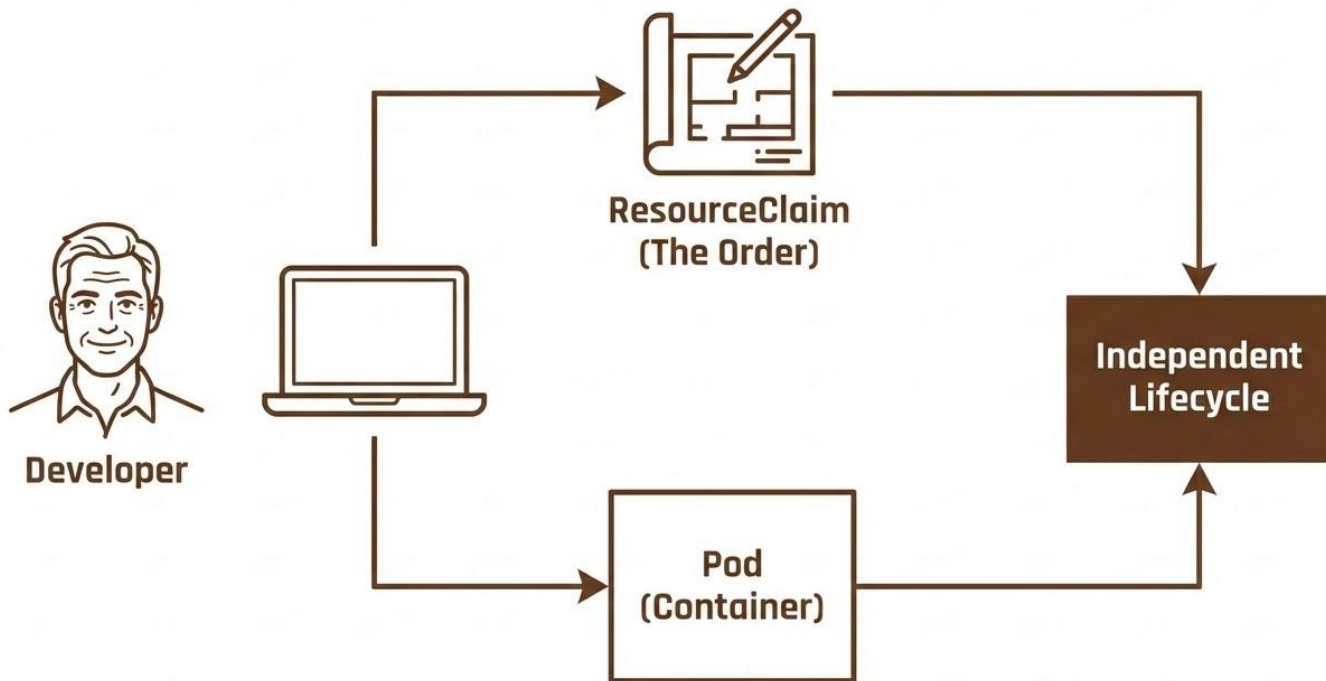


KubeCon



CloudNativeCon

Europe 2026



Putting everything together ...



KubeCon



CloudNativeCon

Europe 2026



The Menu (DeviceClass)

Curated options defined by management (e.g., High-Perf GPU).



The Order (ResourceClaim)

Specific diner request (e.g., Medium-rare, shared).



The Inventory (ResourceSlice)

Live hardware attributes published by drivers.



The Staff (Driver/Scheduler)

Matching orders to inventory dynamically.



KubeCon



CloudNativeCon

Europe 2026

DRA unique features



Consumable capacity



KubeCon



CloudNativeCon

Europe 2026



Consumable capacity



KubeCon



CloudNativeCon

Europe 2026

- **Problem:**

- The app requires 20 GB
- The GPU has 100 GB
- Device plugins would allocate the *whole* card (80 % waste ...)

- **Solution:**

- Drivers *advertise* “capacity”
- Schedulers do the math
- A device *serves* multiple claims, until it is fully used

- **Mechanism:**

- **RequestPolicy:** drivers define steps (e.g. 5 GB increments ...)
- **DistinctAttribute:** prevent double booking (for redundancy)

Requesting a GPU “slice”



KubeCon



CloudNativeCon

Europe 2026

```
apiVersion: resource.k8s.io/v1
kind: ResourceClaim
metadata:
  name: fractional-gpu-claim
spec:
  devices:
    requests:
      - name: req0
        exactly:
          deviceClassName: resource.example.com
          # 1. Request a specific portion of the device's capacity
          capacity:
            requests:
              memory: 10Gi
          # 2. Ensure the selected device explicitly allows sharing
        selectors:
          - cel:
              expression: |-
                device.allowMultipleAllocations == true
```

Influencing placement

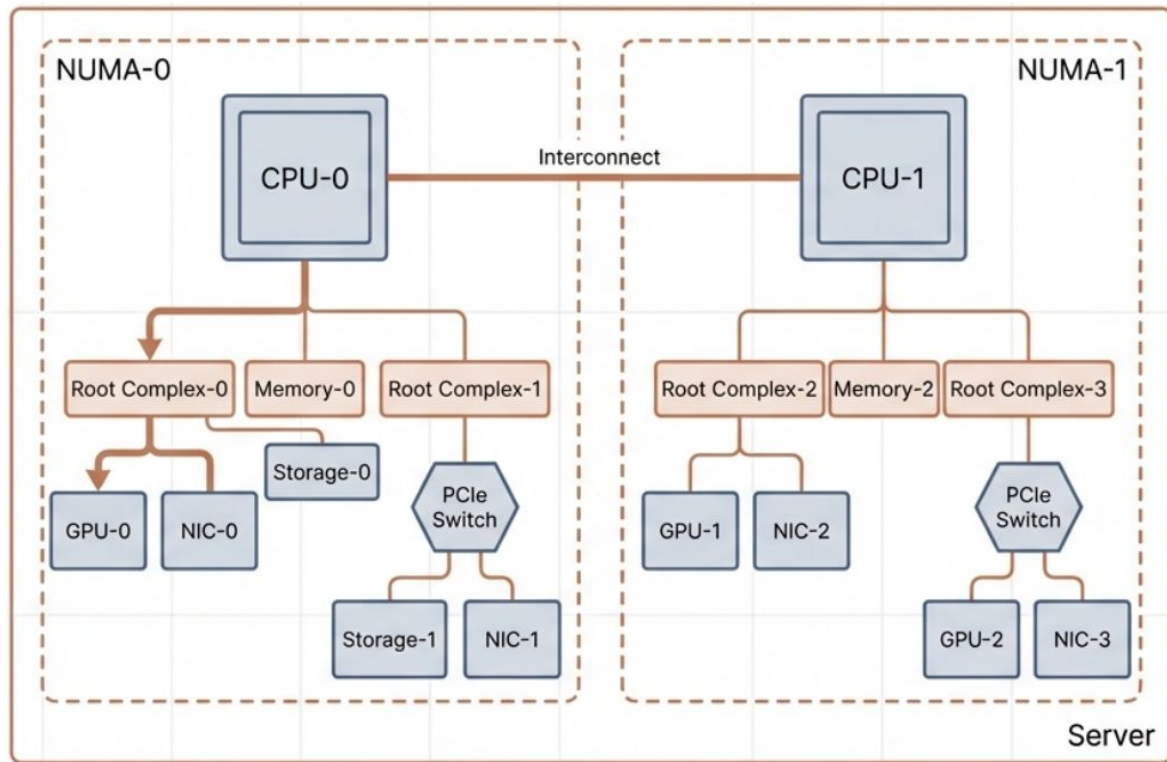


KubeCon



CloudNativeCon

Europe 2026



Influencing placement

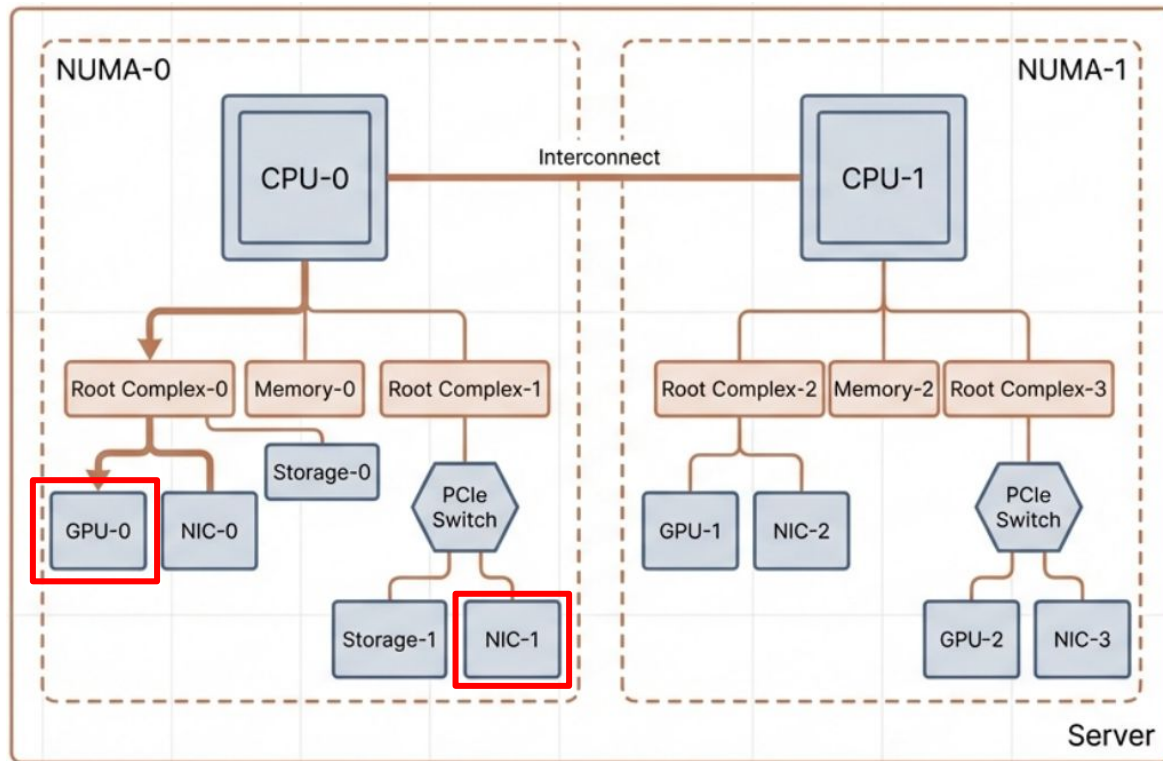


KubeCon



CloudNativeCon

Europe 2026



Influencing placement

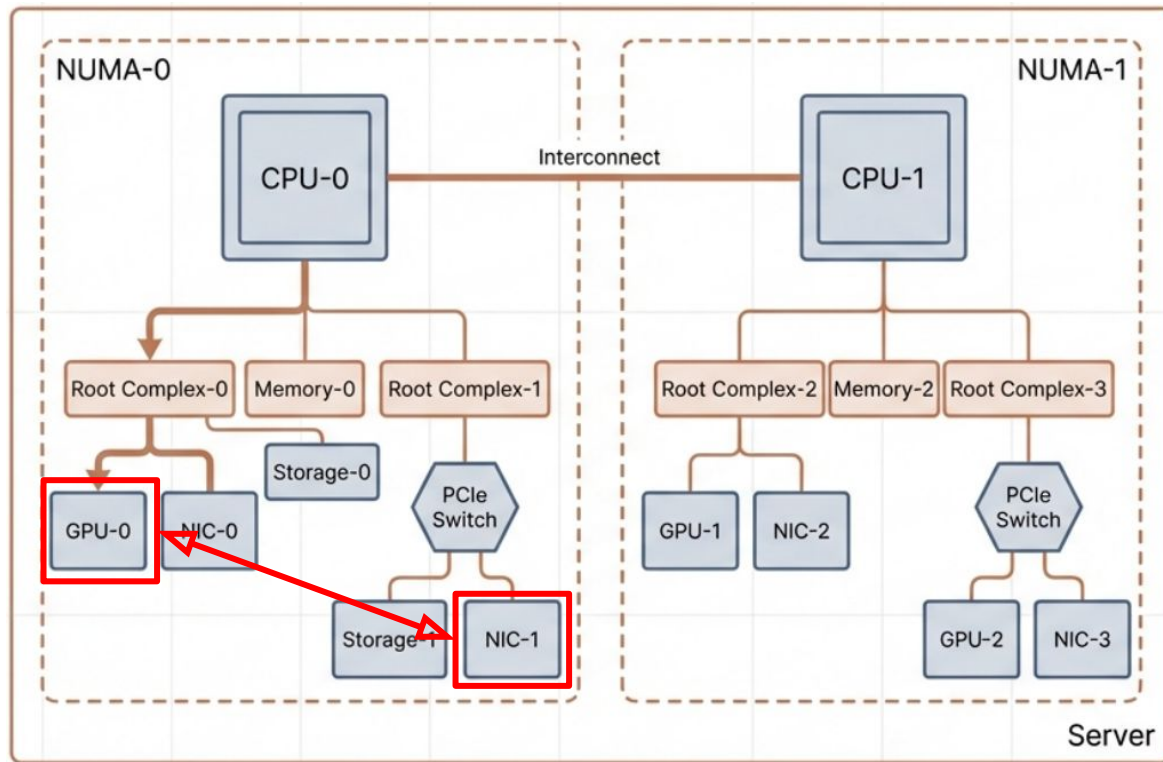


KubeCon



CloudNativeCon

Europe 2026



Influencing placement

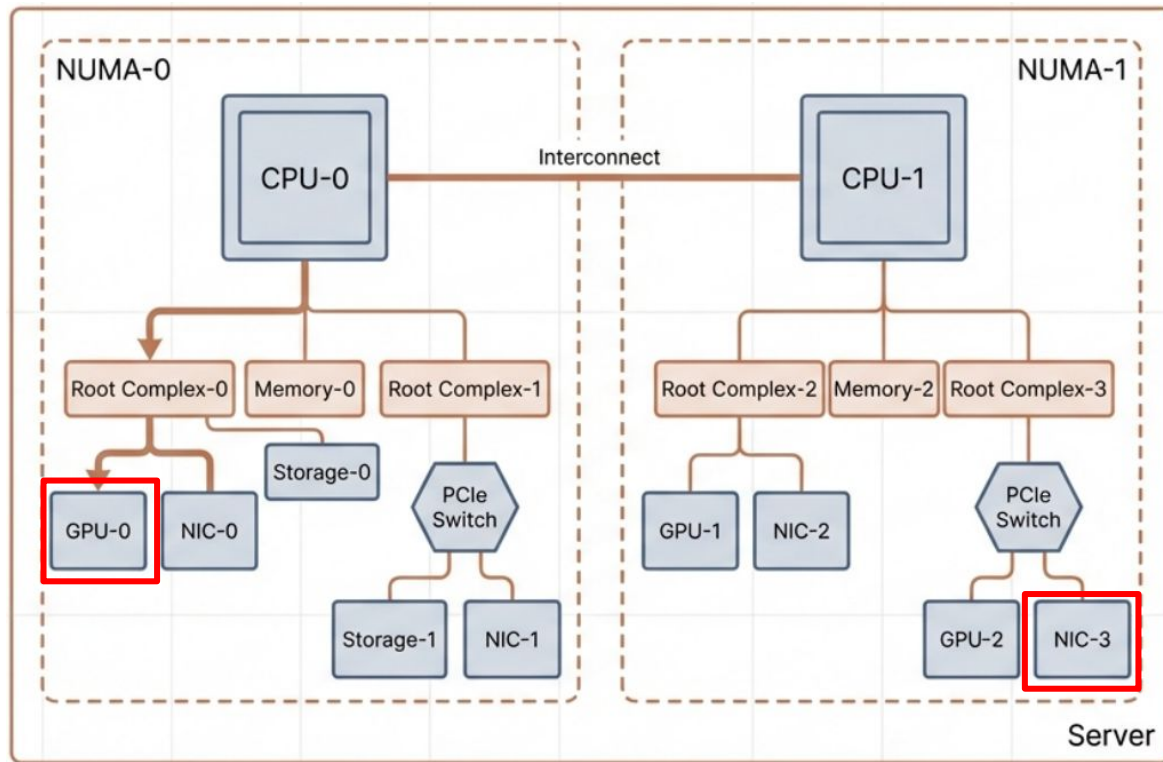


KubeCon



CloudNativeCon

Europe 2026



Influencing placement

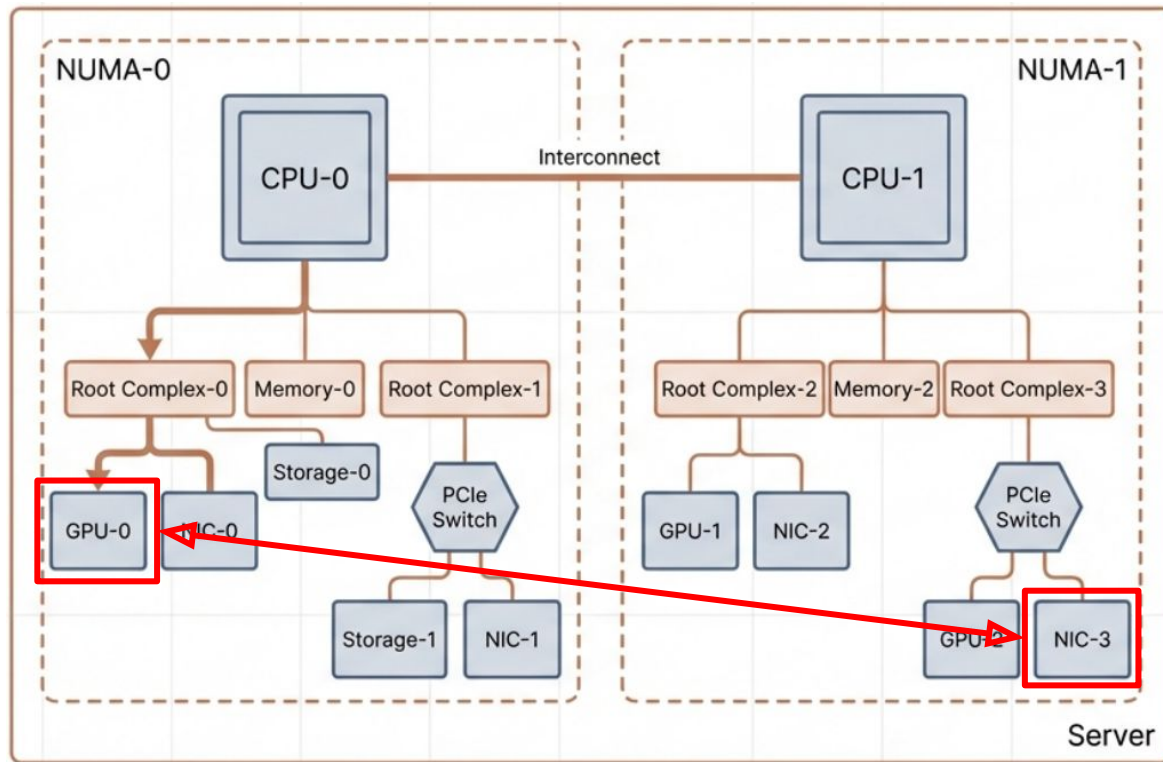


KubeCon



CloudNativeCon

Europe 2026



Influencing placement

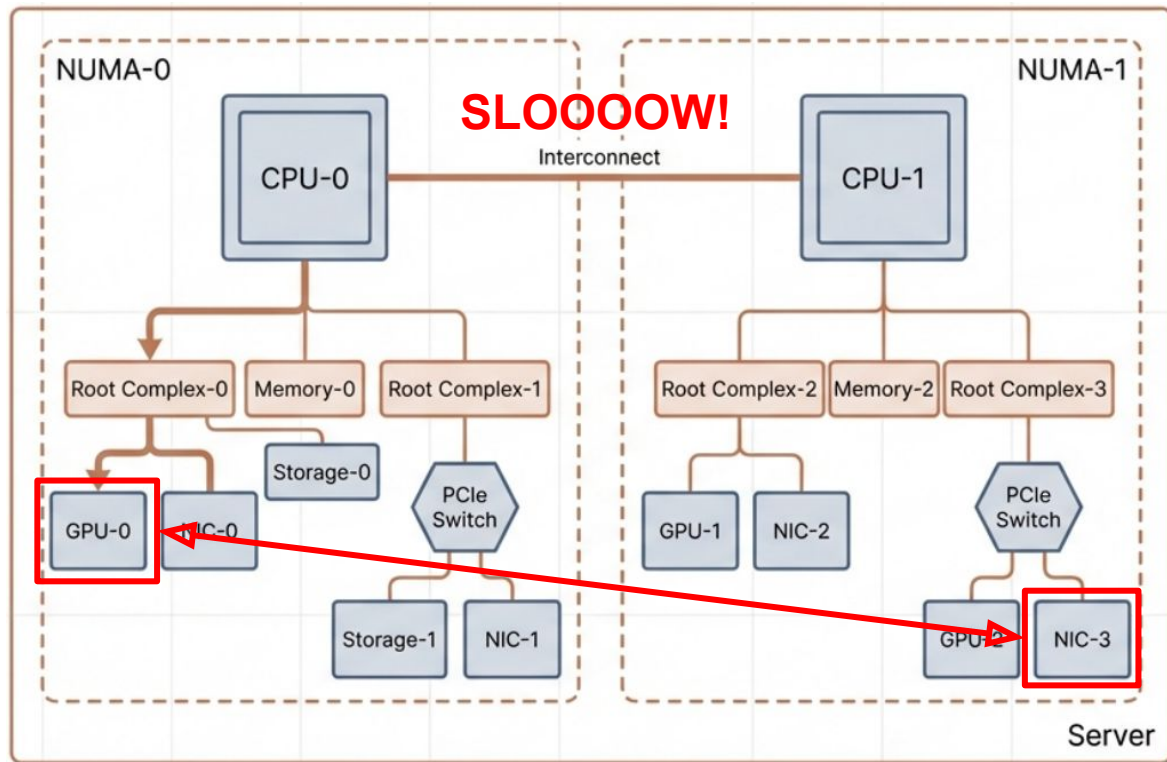


KubeCon



CloudNativeCon

Europe 2026



Influencing placement

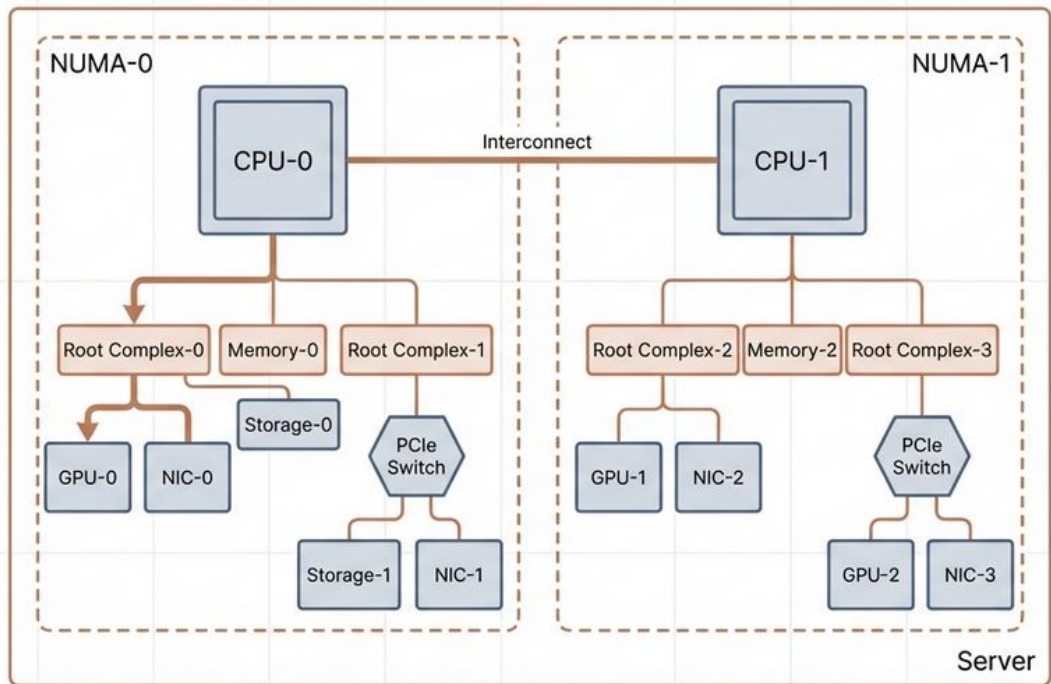


KubeCon



CloudNativeCon

Europe 2026



- **Problem:**

Misaligned GPUs/NICs cause 10-60% performance drops in AI workloads.

- **Solution:**

DRA exposes topology (NUMA IDs, PCIe roots), enabling Topology-Aware Scheduling.

- **Result:**

GPUs and NICs are aligned on the same NUMA node for maximum throughput.

Co-located requests



KubeCon



CloudNativeCon

Europe 2026

```
apiVersion: resource.k8s.io/v1
kind: ResourceClaim
metadata:
  name: gpu-nic-aligned-claim
spec:
  devices:
    requests:
      # 1. Request the GPUs
      - name: gpus
        exactly:
          deviceClassName: gpu
          count: 2
          selectors:
            - cel:
                expression: device.attributes["res.k8s.io"].productName.lowerAscii().matches('^.*a100.*$')
      # 2. Request the Accelerated NIC
      - name: accelerated-nic
        exactly:
          deviceClassName: accelerated-nic
          selectors:
            - cel:
                expression: device.attributes["res.k8s.io"].rdma && device.attributes["res.k8s.io"].linkLayer
== "InfiniBand"
      # 3. Apply the NUMA Node Constraint
    constraints:
      - matchAttribute: numaNode
    requests: ["accelerated-nic", "gpus"]
```



KubeCon



CloudNativeCon

Europe 2026

Workshop time !!!





KubeCon



CloudNativeCon

Europe 2026



Choose your adventure !



Gather 'round the dumpster fire for story time...



KubeCon



CloudNativeCon

Europe 2026



Class: Data Scientist



KubeCon



CloudNativeCon

Europe 2026



Class: Data Scientist

Used to:

Hog GPUs on shared servers.

Now:

Uses “canhazgpu” to reserve GPUs on a single host.

Motivation:

Run experiments fast and not fight coworkers for GPUs.

Stats:

AI/ML: 10/10

Kubernetes: 1/10

CLI Kung-fu: 4/10

Cluster Ops: 0/10

Strengths:

Knows the model, the data, and the math.

Weaknesses:

Will absolutely forget to release the GPU.

Bonus ability:

Long-running screen session

Class: vLLM Developer



KubeCon



CloudNativeCon

Europe 2026



Class: vLLM Developer

Used to:

Reserve GPUs with `canhazgpu` and run experiments on a single machine.

Now:

Uses “`k8shazgpu`” to run GPU workloads across the Kubernetes cluster.

Motivation:

Run code at scale without learning Kubernetes YAML.

Stats:

AI/ML: 9/10

Kubernetes: 2/10

CLI Kung-fu: 7/10

Python: 10/10

Strengths:

Ships code quickly and understands the latest open weight models.

Weaknesses:

If you show them a PodSpec they will panic.

Class: Kube Admin



KubeCon



CloudNativeCon

Europe 2026



Class: Kubernetes Admin

Used to:

Explain why GPU nodes were always full.

Now:

Uses Kubernetes + DRA to manage GPU allocation across the cluster.

Motivation:

Make GPUs schedulable resources instead of social agreements.

Stats:

AI/ML internals: 3/10 (and growing!)

Kubernetes: 10/10

CLI Kung-fu: 9/10

YAML: 10/10

Strengths:

Understands scheduling, nodes, and cluster behavior.

Weaknesses:

Will try to debug everything with ``kubectl describe``.

Bonus ability:

Spot a misconfigured ResourceClaim from 2 nautical miles away.

Sign up for a shell account!



KubeCon



CloudNativeCon

Europe 2026



<https://tinyurl.com/kubecondra>

(it's to google forms, not a rickroll [hopefully])

Do you have GitHub SSH keys setup?



KubeCon



CloudNativeCon

Europe 2026

<https://github.com/torvalds.keys>

**To check, just put your username instead of Linus
Torvald's username.**

All the course materials!



KubeCon



CloudNativeCon

Europe 2026



<https://github.com/dougbtv/dra-matically-simply-dra-lab>

or

<https://tinyurl.com/kubecondrarepo>



KubeCon



CloudNativeCon

Europe 2026



Connecting Your Laptop to Present



KubeCon



CloudNativeCon

Europe 2026

If you plan to use speaker notes, please follow the steps below to ensure your laptop is set up correctly. This will allow you to view your notes while the audience sees only your slides.

Before Connecting

- Close all unnecessary apps & tabs
- Turn off notifications
- Disable Night Mode / TrueTone
- Don't enter full-screen presentation mode yet

At the podium

- Plug in the HDMI cable
- If needed, use your USB-C to HDMI adapter
- On Mac, click "Allow" if prompted

USE "EXTENDED DISPLAY" MODE (NOT MIRRORED)

Using MAC

- Go to System Settings > Displays
- Click the "Arrangement" tab
- Make sure "Mirror Displays" is unchecked

Using PC

- Press $\square$ Win + P, then select "Extend"
- Or: Right-click on the Desktop > Display Settings
- Under Multiple Displays, select "Extend these displays"

Connecting Your Laptop to Present



KubeCon



CloudNativeCon

Europe 2026

When Using PowerPoint

- Start slideshow: ⌘Cmd + Enter (Mac) or F5 (PC)
- Or: Click Slide Show > Play from Start
- If notes show on the wrong screen, click “Swap Displays”
- <https://www.youtube.com/watch?v=gQ3D4m-5pww>

Google Slides

- Use Google Chrome as your browser
- Click the dropdown next to ‘Slideshow’ > select Presentation Display Options
- Check “Presenter View” and “Full Screen”
- If prompted, select "Allow"
- Choose the external display (may have a strange name like PanasonicXYZ)
- Click Start Slideshow
- <https://www.youtube.com/watch?v=-GT7WCvPcys>