International Symposium on Grids & Clouds (ISGC)

19 March 2026 - Taipei

Automating Distributed Cloud Infrastructures for AI-Driven Earth Observation Services

Marica Antonacci

European Center for Medium-Range Weather Forecasts (ECMWF)

ECMWF

Talk roadmap

- **Context**: ECMWF and the European initiatives driving the platform
- **The Challenge**: Running AI-ready services across distributed cloud infrastructures
- **Our Platform**: A GitOps-driven multi-cluster Kubernetes architecture
- **Implementation**: Automating the platform with Terraform, Ansible, Rancher and Fleet
- **Next Generation**: Evolving towards a Kubernetes-native control plane (Cluster API, Crossplane, Sveltos)
- **Takeaways**: Lessons learned from building distributed scientific platforms

European Center for Medium-Range Weather Forecasts

- Founded 1975, supported by 35 States (23 Members + 12 Co-operating)
- Sites in Reading (UK), Bologna (IT) & Bonn (DE)

Core Mission

- Produce global numerical weather forecasts (up to seasonal scales)
- Advance research & modelling to improve forecast skill
- Maintain one of the largest meteorological data archives worldwide

Operational Services

- Forecasts produced four times per day and delivered globally
- Provides: weather, climate, air quality, hydrology & environmental analyses
- Implements Copernicus services (CAMS & C3S) and contributes to Earth monitoring

HPC & Data Infrastructure

- Operates a world-class high-performance computing facility for NWP
- Meteorological archive holds hundreds of petabytes of data and grows daily
- Supports cloud platforms (e.g., European Weather Cloud, Copernicus Data Stores)

Driving Innovation through ECMWF EU Projects

Our External Funded Projects team:

- Leads **European collaborations** to deliver innovative services
- Brings expertise in **cloud infrastructure**, **AI/ML**, and **Earth Observation**

Representative projects:

https://www.eo4eu.eu/

Simplifying access to Earth Observation data & AI analytics

https://buildspaceproject.eu/

Creating digital twins for energy-efficient and resilient buildings

https://climres.eu/

Enhancing climate resilience in buildings and urban environments

https://meditwin-project.eu/

Modelling Mediterranean climate extremes with AI-driven digital twins

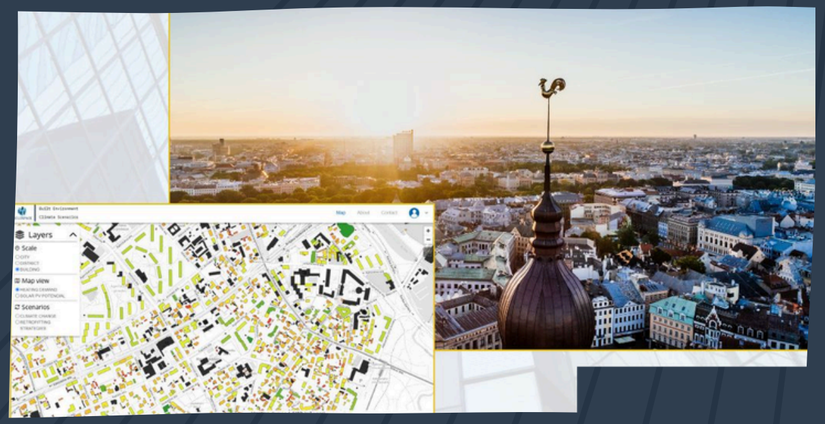
Project Goals

Provide user-oriented services across multiple domains

Simplify access to Earth Observation data

Leverage AI/ML for decision-support and analytics

➡ ***This requires scalable and reproducible cloud infrastructures.***



ECMWF

Infrastructure Challenges

Running these services introduces challenges:

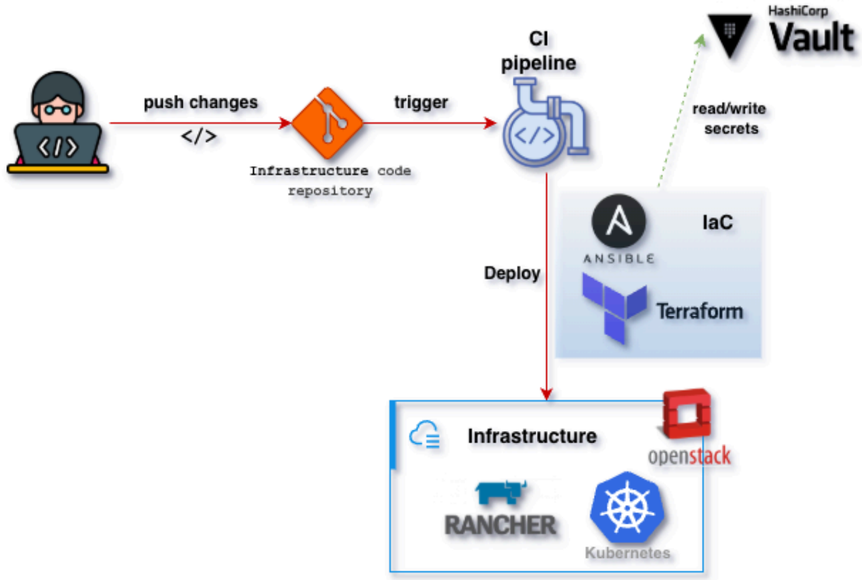
- **multi-cloud** environments
- geographically **distributed infrastructure**
- heterogeneous OpenStack deployments
- need for **secure and reproducible deployments**
- support for **GPU workloads** and data analysis environments

Requirements:

- **automation**
- **reproducibility**
- **scalability**
- **portability**

GitOps Workflow

- **Git as the source of truth:** Every infrastructure and configuration change starts as a Git commit - ensuring full traceability and reproducibility
- **Automation through CI/CD**: GitLab pipelines automatically trigger deployment workflows whenever changes are pushed, reducing manual operations
- **Infrastructure as Code (IaC)**: Terraform provisions the underlying infrastructure (OpenStack VMs, Networks, FIPs, etc.) and RKE2 clusters, while Ansible configures Rancher and other advanced setups. GitLab is used as a backend for Terraform state files.
- **Secrets management integration**: Terraform interacts securely with a **HashiCorp Vault** to read/write sensitive credentials (e.g. OpenStack application credentials, AWS credentials for Route53, SSH keys, etc.)

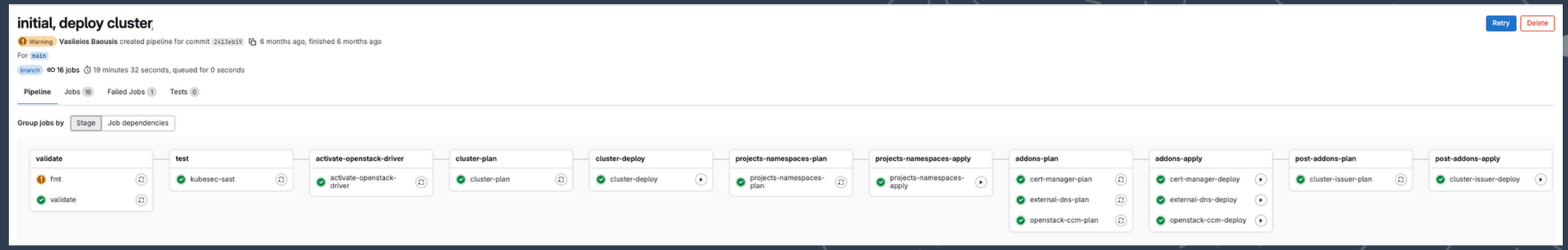


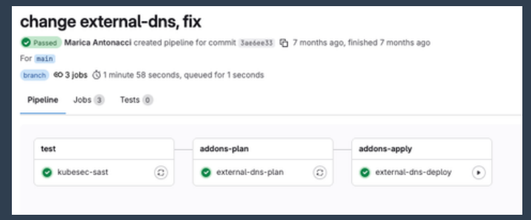
Rancher Management Cluster and Downstream RKE2* Clusters Deployment Workflow

(*) RKE2 is a secure, production-ready Kubernetes distribution designed for enterprise and regulated environments, developed by SUSE, which maintains Rancher.

Repositories and CI/CD pipelines

- A dedicated Git repository stores **the Terraform code** for deploying the **Rancher management (admin) cluster**, while **each downstream RKE2 cluster** has its own separate repository.
- The same GitOps workflow is applied across all repos, but **downstream cluster pipelines are triggered by commit comments**, which determine whether to **create**, **update**, or **destroy** a cluster - executing the corresponding *terraform apply* or *terraform destroy* commands.





Limitations of the Terraform-based model

The Terraform-based workflow works well but introduces issues:

- **code duplication** across cluster repositories
- maintenance overhead
- difficult **cross-cluster updates**
- cluster lifecycle not fully **Kubernetes-native**

Example:

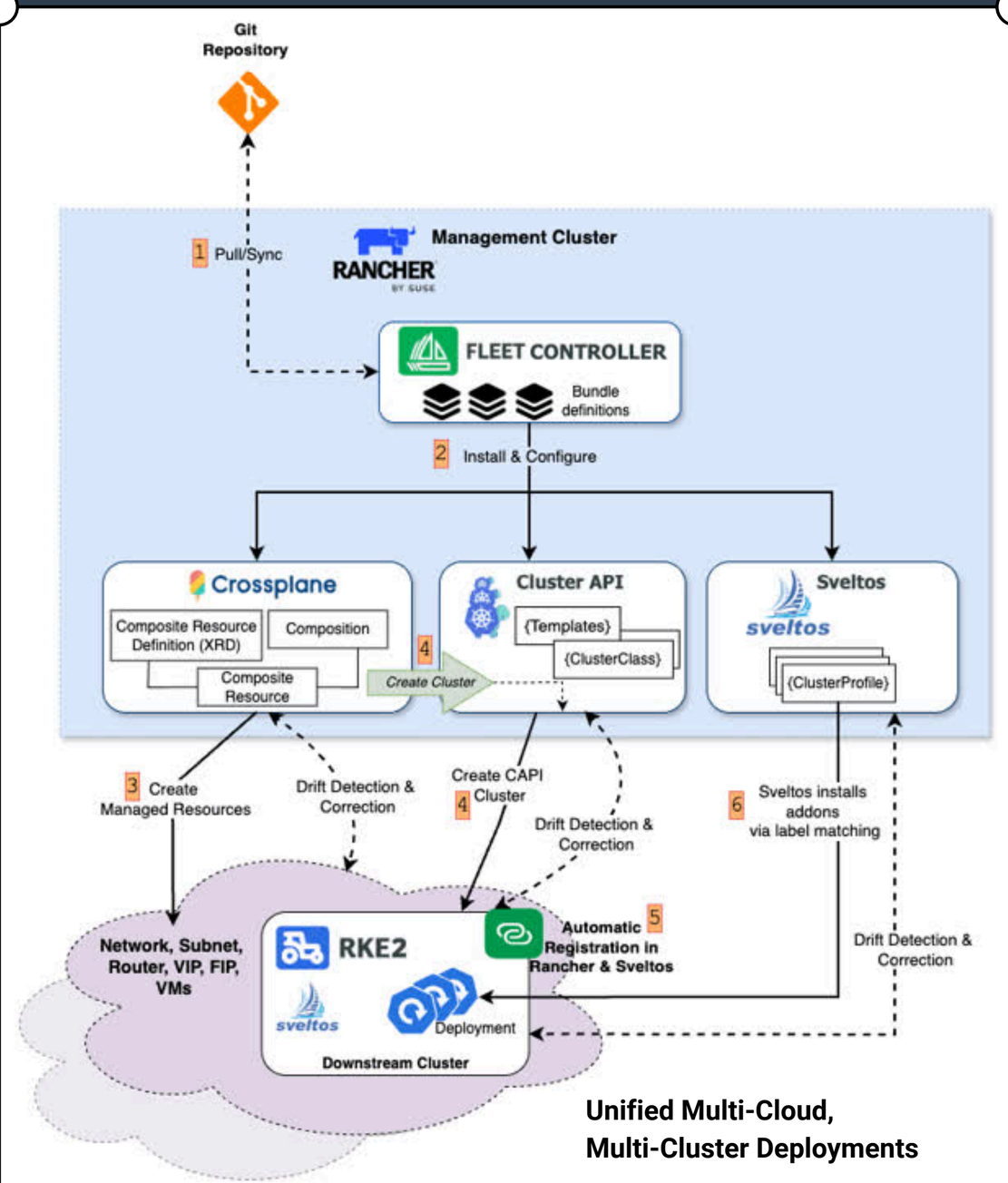
Fixing a bug in ingress configuration

→ requires updates in **every cluster repository!**

This motivated exploring new provisioning models

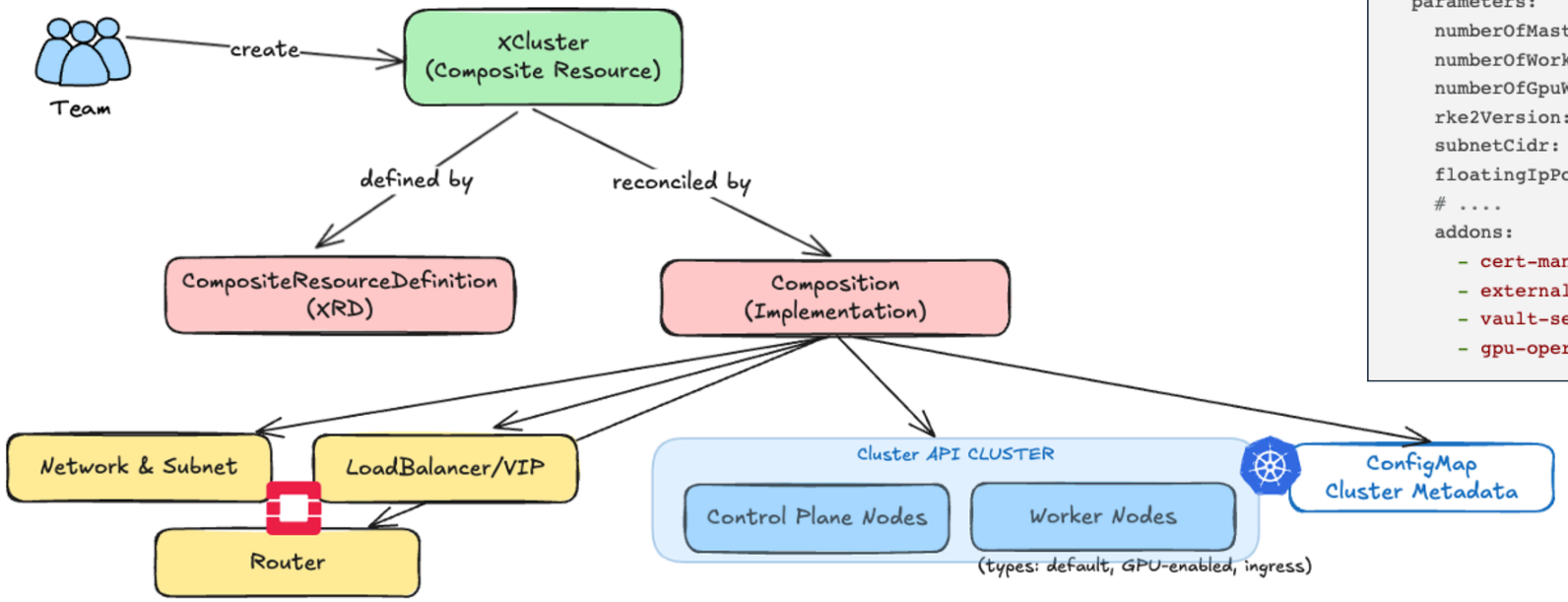
Towards a Kubernetes-Native Platform

- **Management cluster bootstrap**: A management cluster is created and configured with all required controllers and operators.
- **Unified cluster abstraction**: **Crossplane** abstracts infrastructure and cluster creation into a single declarative resource.
- **Kubernetes lifecycle management**: **Cluster API** handles provisioning, scaling, upgrades, and healing of workload clusters.
- **Label-driven add-ons**: **Sveltos** deploys and manages add-ons automatically based on cluster labels.
- **End-to-end GitOps**: The entire workflow is declarative, automated, and continuously reconciled.



ECMWF

Declarative Cluster Provisioning




```yaml
apiVersion: crossplane.ecmwf.int/v1alpha1
kind: XCluster
metadata:
  name: ecmwf-ai-platform
spec:
  parameters:
    numberOfMasters: 3
    numberOfWorkers: 6
    numberOfGpuWorkers: 2
    rke2Version: v1.33.3+rke2r1
    subnetCidr: 10.0.50.0/24
    floatingIpPool: external-internet
    # ....
    addons:
      - cert-manager
      - external-dns
      - vault-secrets-operator
      - gpu-operator
```

A single resource defines:

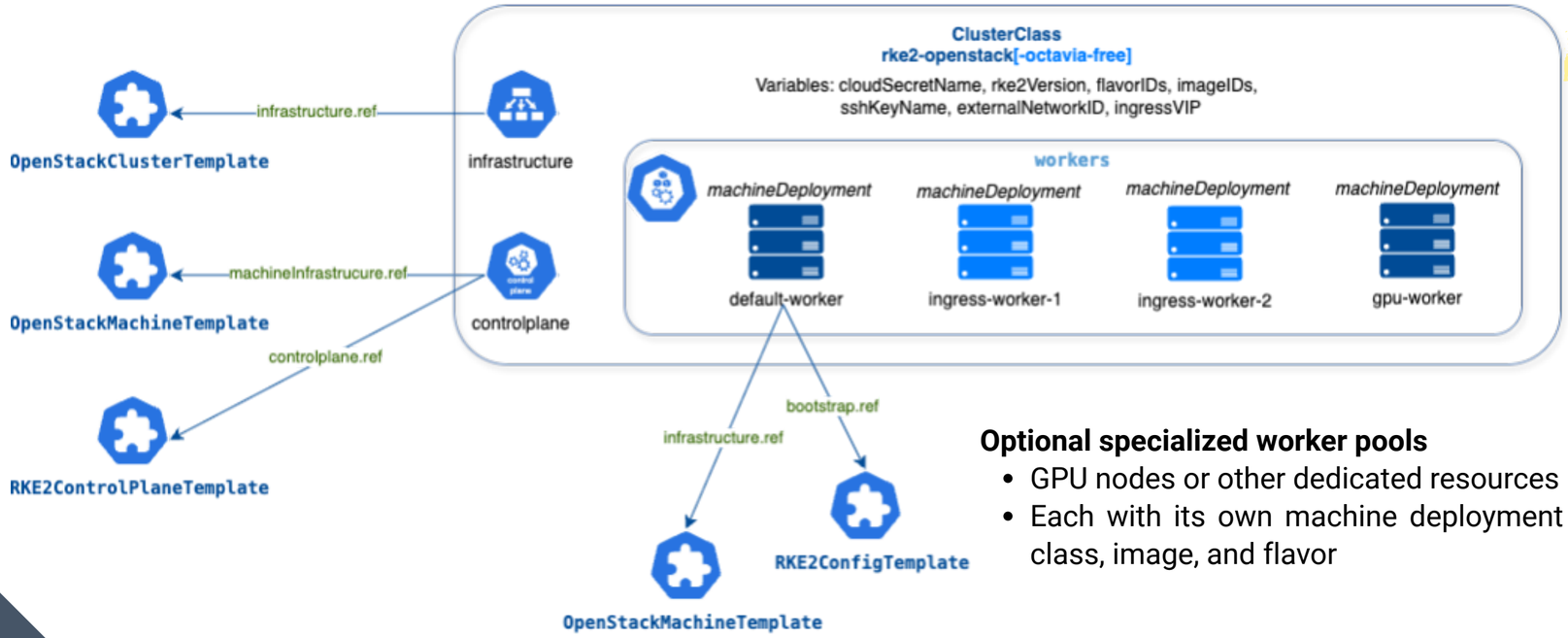
- **infrastructure**
- **Kubernetes cluster topology**
- **platform add-ons (Sveltos)**

One high-level API (*XCluster*) drives networking, compute, and cluster provisioning.

Crossplane abstracts and orchestrates Kubernetes cluster deployments together with all required infrastructure resources, acting as a **glue between infra and cluster**.

Flexible Cluster Design with ClusterClass

Cluster API *ClusterClass* defines the **cluster blueprint**; *templates* provide **reusable** infrastructure and bootstrap **building blocks** that are specialized through patches and variables.



Optional specialized worker pools

- GPU nodes or other dedicated resources
- Each with its own machine deployment class, image, and flavor

One reusable cluster blueprint, two interchangeable ingress topologies

Standard Deployment (Octavia available)

- OpenStack Load Balancer for Kubernetes API
- Ingress services exposed via LoadBalancer
- Fully managed integration with OpenStack networking

Octavia-Free Deployment

When load balancing is not available:

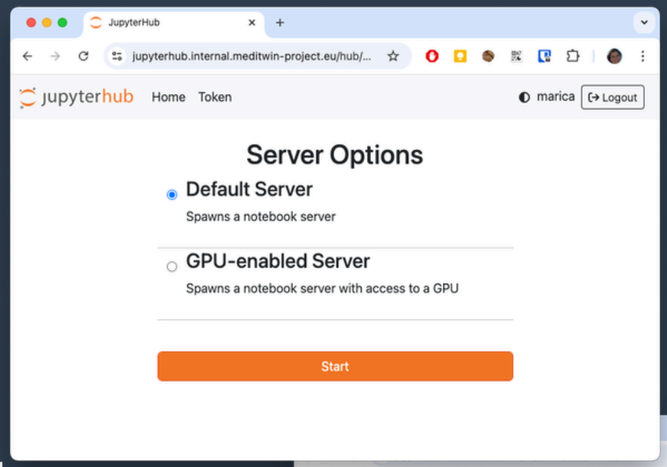
- Dedicated ingress nodes + shared VIP
- Keepalived-based failover for HA
- Crossplane provisions VIP & Floating IP

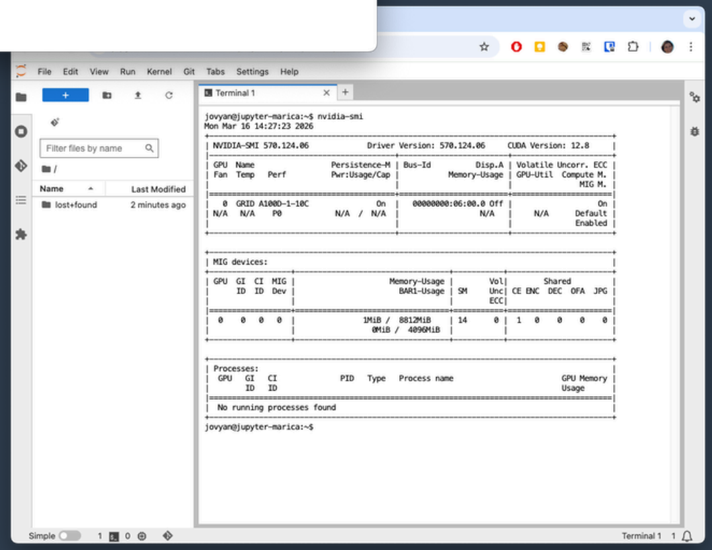
Sveltos configures addons according to the selected topology.

AI/ML Workloads on GPU-Enabled Clusters

Nvidia GPU Operator

- **Automatic deployment via Sveltos**
  - Targets clusters labeled *addons.gpu-operator: enabled*
- **Automatic GPU discovery**: detects all GPU nodes & their properties (model, memory, compute capability)
- **Container runtime configuration**: sets NVIDIA runtime in containerd
- **Toolkit management**: installs NVIDIA drivers & CUDA libraries (optional)





Example: JupyterHub on GPU Nodes

- GitOps-managed deployment via Helm chart
- **Server profiles**:
  - **Default Server**: standard CPU-only notebook
  - **GPU-enabled Server**: PyTorch/CUDA notebook, **nvidia.com/gpu: 1**

Takeaway: Single-click AI/ML environment leveraging GPU nodes, fully automated & reproducible

Key Takeways

Terraform (Pipeline-driven)	Kubernetes-native (Controller-driven)
Provisioning via external pipelines	Declarative resources inside Kubernetes
Infrastructure state managed outside K8s	Continuous reconciliation by controllers
Limited lifecycle automation	Fully GitOps-driven platform
Event-based, pipeline-driven	Controller-based, fully automated lifecycle

It's a paradigm shift — not just different tools, but a tighter, automated, GitOps-integrated lifecycle.

Looking Ahead: Agentic IaC Automation

AI is transforming how we build software

But infrastructure still relies on human-driven workflows

»»» A new paradigm is emerging where AI meets infrastructure automation

Kubernetes-Native Foundation

- Declarative APIs for everything
- Continuous reconciliation of desired state
- Fully programmable and observable infrastructure

»»» The ideal foundation for agent-driven automation

Agentic AI-powered IaC

AI agents can observe, learn, reason, and adapt.

Key capabilities:

- Intelligent Provisioning
- Self-Healing Infrastructure
- Predictive Resource Optimization

From declarative infrastructure to self-driving platforms

ECMWF

Thank you for your attention

The team:
Mohanad Albughdadi - mohanad.albughdadi@ecmwf.int
Marica Antonacci - marica.antonacci@ecmwf.int
Vasileios Baousis - vasileios.baousis@ecmwf.int
Federico Fornari - federico.fornari@ecmwf.int
Tolga Kaprol - tolga.kaprol@ecmwf.int
Claudio Pisa - claudio.pisa@ecmwf.int