AI_INFN
Artificial Intelligence technologies for INFN research

INFN CLOUD

FONDAZIONE CR FIRENZE

Developing Artificial Intelligence in the Cloud: The AI_INFN platform

INFN

Lucio Anderlini
lucio.anderlini@fi.infn.it

Giulio Bianchini
giulio.bianchini@pg.infn.it

Diego Ciangottini
diego.ciangottini@pg.infn.it

Stefano Dal Pra
stefano.dalpra@cnaf.infn.it

Rosa Petrini
rosa.petrini@fi.infn.it

Daniele Spiga
daniele.spiga@pg.infn.it

Scope and objectives

The provisioning of a **common**, **stable**, and **reliable** ground for researchers involved in ML to develop, review and share their applications, **crossing the borders between different communities**, INFN units, experiments and research domains

Provide a **centrally maintained cloud-based infrastructure** for interactive and batch ML fast prototyping, with access to modern hardware accelerators (GPU, FPGA…) and systems tuned for ML performance

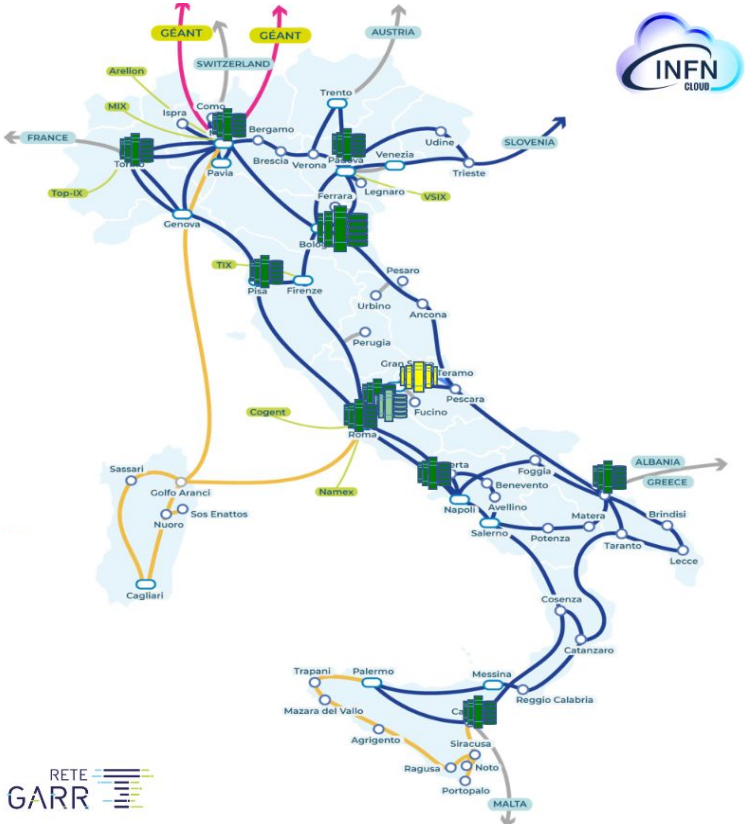
Outline

1. The AI_INFN Platform we are using today:
   https://hub.ai.cloud.infn.it
2. The ongoing developments:
   distributed computing on *virtual kubelets* with interLink
3. Roadmap towards maturity:
   automation, *documentation* and *security*

DataCloud

In the framework of the current NRRP projects (ICSC, TeRABIT) INFN has a leading role in the creation of the **Italian Cloud Federation**, to access all Italian scientific computing resources through uniform interfaces

- **Tier-1** (CNAF)
- **Tier-2** (BA, CT, LNF, LNL/PD, NA, MI, PI, RM1, TO)
- Backbone and federated clouds
- HPC4DR (LNGS)
- **INFN Cloud**:
  - a data lake-centric, heterogeneous federated Cloud infrastructure spanning multiple sites across Italy, providing an extensible portfolio of solutions tailored to multidisciplinary scientific communities



INFN Cloud Resources: Infrastructure

ML_INFN has been among the first and most enthusiastic users of INFN Cloud.

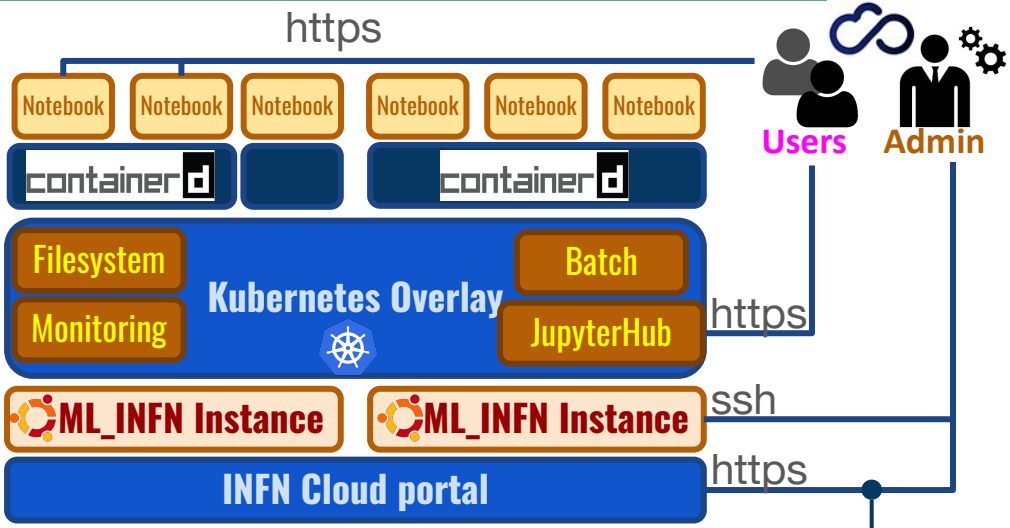
Computing resources available to AI_INFN are located in Room Tier-1 of CNAF and managed through a virtualization layer (**OpenStack of Cloud@CNAF**) in INFN Cloud.

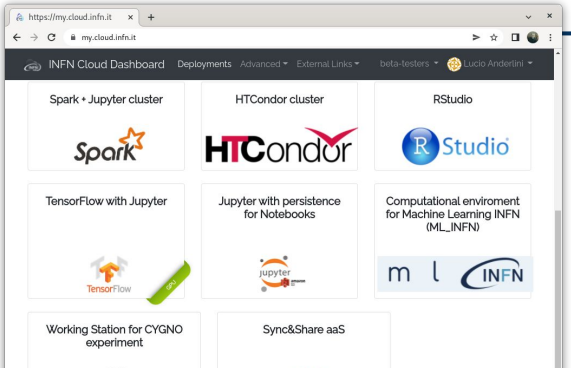
- **Server 1**: 8 nVidia ***Tesla T4*** (**CSN5**) + 5 nVidia ***RTX 5000*** (**ML_CLOUD**, Firenze)
- **Server 2**: 1 ***A100*** (**CSN5**) + 1 A30 (Dip. di Fisica, **UniFi**)
- **Server 3**: 3 ***A100*** (**CNAF**)

Partitioning A100 GPUs with **MIG** (*Multi Instance GPU*) technology, we manage to serve up to **42 GPUs** for interactive development.

INFN Cloud Resources: Architecture

- The ML_INFN outcome: **"sharing precious GPUs through the Cloud is feasible and effective!"**
- With **AI_INFN,** we improved on sharing by **decoupling data from computing resources,** with a **filesystem shared** across the VMs
- An additional abstract, elastic overlay is added on top of multiple VMs **Kubernetes Overlay**:
  - login via AAI → **INDIGO IAM**
  - Monitoring & Accounting
  - Managed software environments for ML
- Adding and removing VMs enables manual horizontal scaling

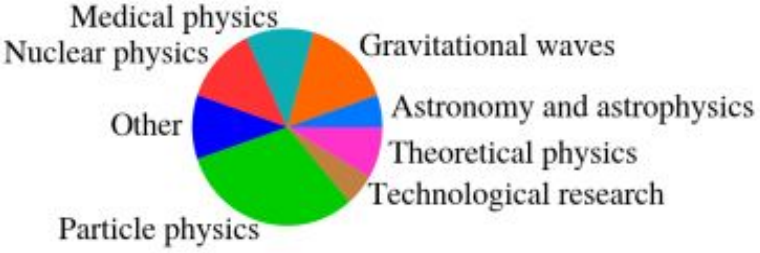




A stress test: the ML_INFN Hackathon

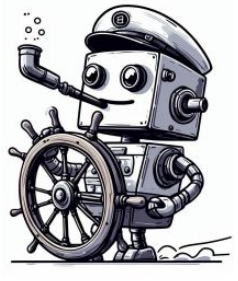
WP Leader: *Francesca Lizzi (INFN Pisa)*

- ML_INFN organized training events (*"hackathons"*), targeting **entry level** (june 2021, december 2021, june 2023) and **advanced** (Bari in november 2022, Pisa in november 2023) audience.



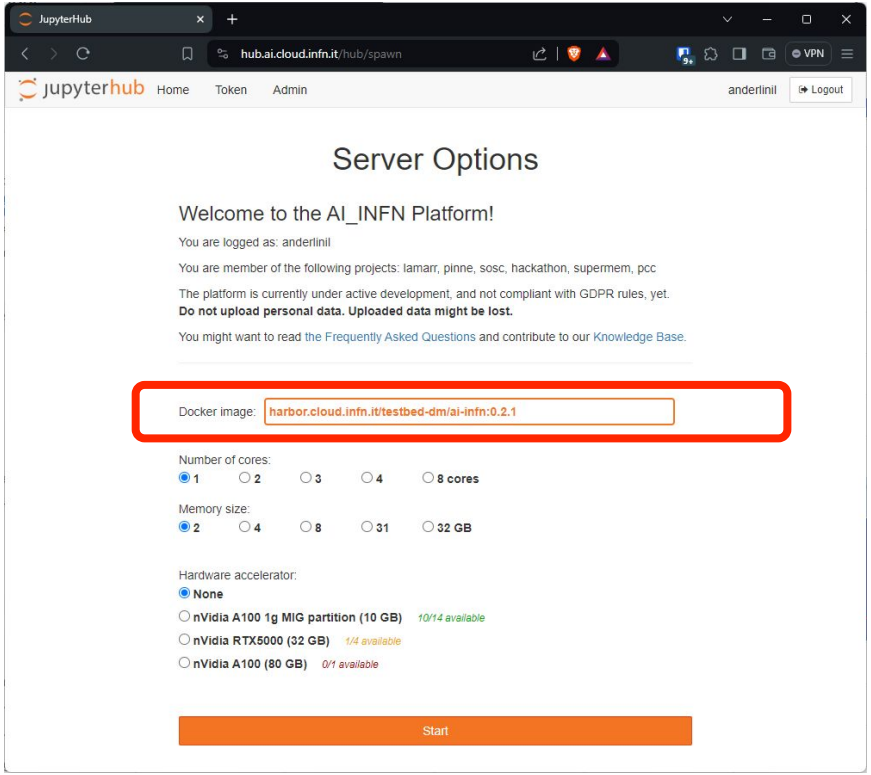
- In the latest event, the AI_INFN's new platform was stress-tested:
  - at **Cloud@CNAF** (using 2 × A100 GPUs for up to 14 participants)
  - at **ReCaS-Bari** (using 4 × A100 GPUs for up to 28 participants)

- Independent networks and file-systems
- Shared IAM authentication
- Synchronized software environments
- Intensive use of the GPUs



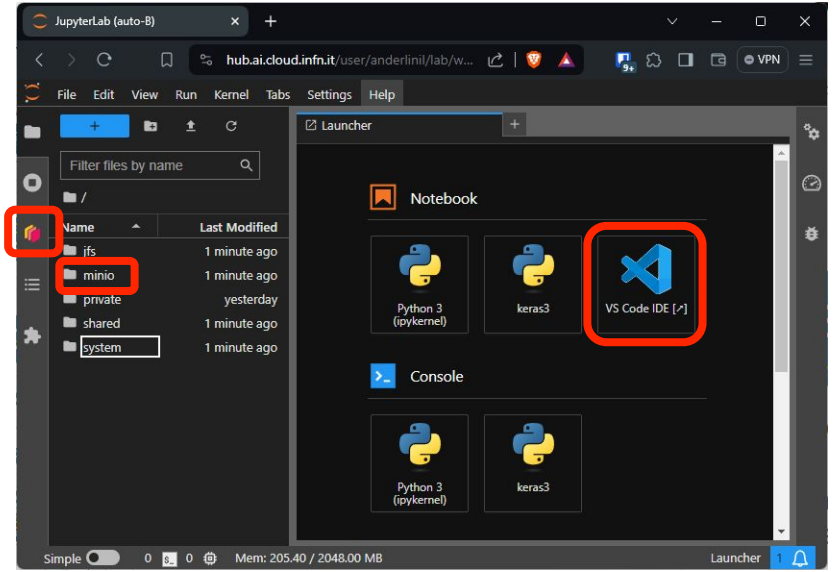
Managed software environments: docker

AI_INFN – User support contact person: *Matteo Barbetti (CNAF)*



The **customizable** docker image defines the user interface.

Default: *VS Code, Dask, MinIO (soon Rados)*



Managed software environments: conda

Configuring the Python software stack to properly control the GPU is sometimes challenging and requires time and expertise.
Sometimes, projects require multiple environments in the same JupyterLab session: picking the right docker image is not a viable option.

A cross-platform and language agnostic packa ge and environment manager, which solves **p ortability** between collaborators and is adopted especially when **python external tools** are used.

Conda utilization on **JupyterLab**:

- Allows to manage **dependencies** of Python projects efficiently.
- Provides **isolated environments** to execute Python code and Jupyter notebooks, independent of the underlying docker image.
- **Users are encouraged to clone and customize the managed conda environments to add their project's dependendencies.**

Managed software environments: conda

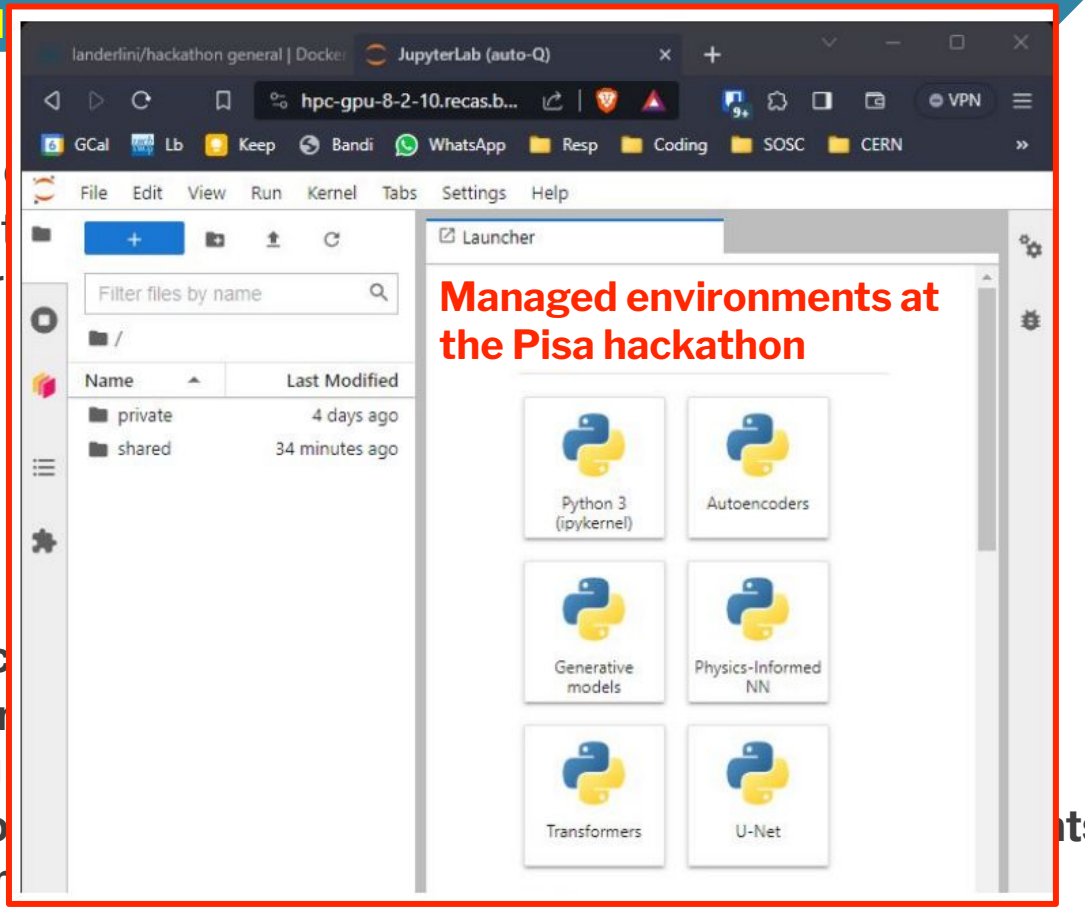
AI_INFN

Configuring the Python software
challenging and requires time an
Sometimes, projects require mul
session: picking the right docker

CONDA

Conda utilization on **JupyterLab**:

- Allows to manage **dependenc**
- Provides **isolated environmer**
  independent of the underlying
- **Users are encouraged to clo** **nts to**
  add their project's depender



Managed software: apptainer

Main problem with conda: it generates environments with 10000+ files, bad for any file system.

A nightmare when distributed.

APPTAINER

- Apptainer is a containerization platform offering an isolated, reproducible environment for application execution.
- Allows to pack an application and all its dependencies in a container, granting portability and consistency of the execution environment.

Advantages of Conda + Apptainer:

- **Conda** is what developers expect, **Apptainer** (squashfs) delivers envs as a single file.
- **Reproducibility:** By using Conda for development and Apptainer for execution, it's possible to ensure complete reproducibility of the environment both during development and distribution.

Monitoring & Accounting with GPU

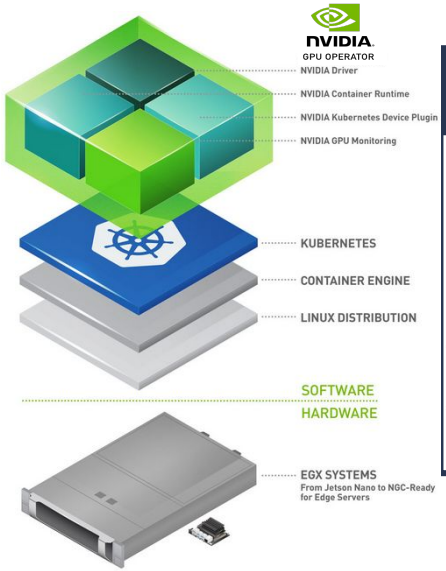
Contact person: *Rosa Petrini (INFN Firenze)*

Three levels of monitoring & accounting:

- **Resource provisioning accounting**: report on resource usage
- **Resource provisioning monitoring**: check if allocated resources are in use or idle
- **Service accounting:** to have vision of the balance and distribution of the resources among projects and, in case of high load, to enforce/guarantee fair access to resources between users.
    - This is to have control over who is using the AI_INFN platform and to do what. In this way we can estimate how much we could shrink the CPU and RAM resources allocated to a single-accelerator task without an evident penalty in performance

Monitoring & Accounting with GPU

Contact person: *Rosa Petrini (INFN Firenze)*



EXPORTER

NVIDIA DCGM-Exporter + Kube Eagle

Prometheus

DASHBOARD

Grafana

Accounting: Configuration of a PostgreSQL server through Ansible

(Nadir Marcelli & Stefano Stalio)

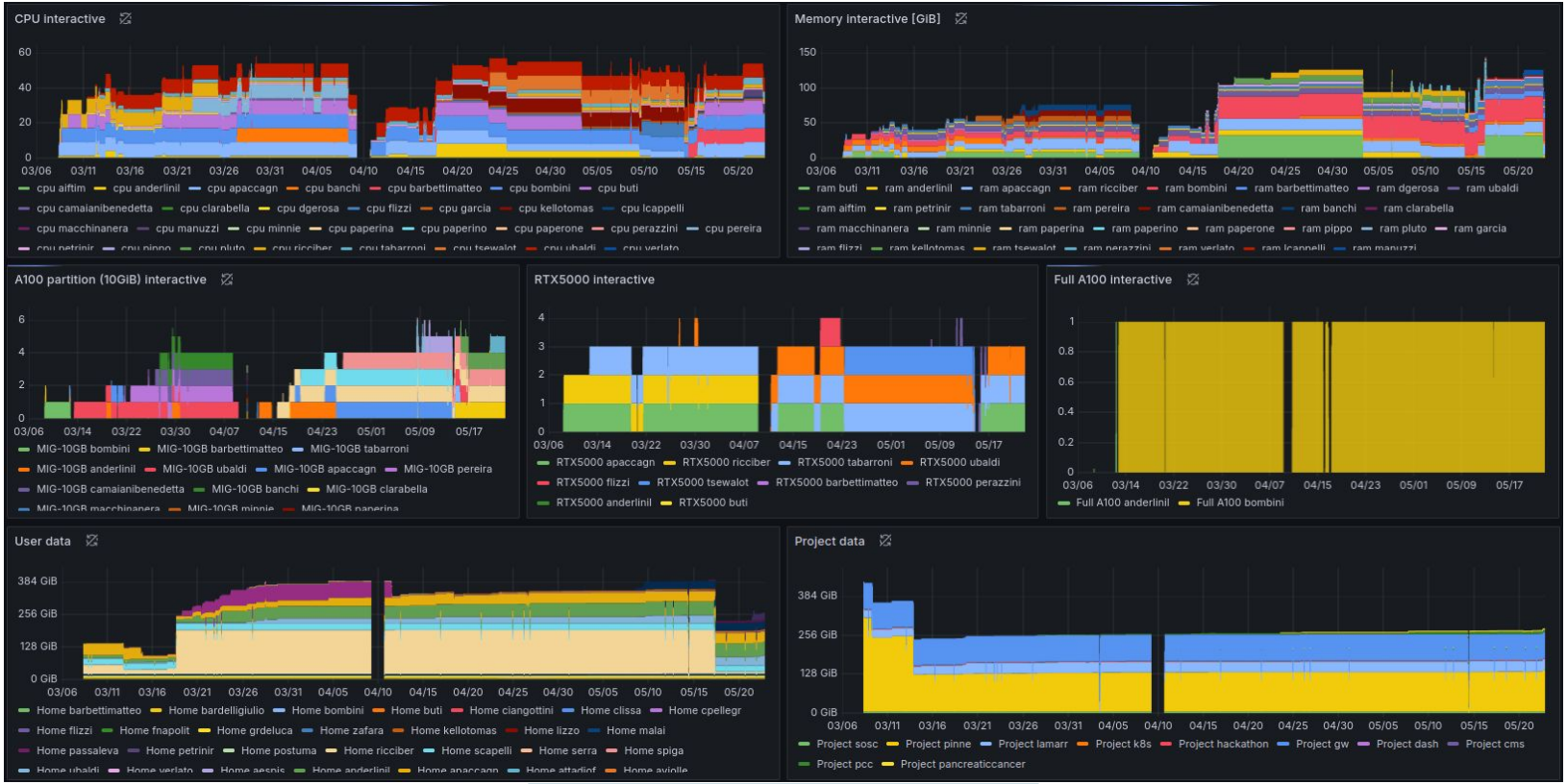
- Allows synchronous replication on one or more secondary servers
- Configuration of an **SSL connection** to ensure a secure communication channel for replication
- Includes configuration of pgbackrest for **periodic backup**
- Installation of repmgr for **automatic failover management**.

Monitoring & Accounting with GPU:
Grafana: Monitoring

Contact person: *Rosa Petrini (INFN Firenze)*

Monitoring & Accounting with GPU:
Grafana: Accounting

Contact person: *Rosa Petrini (INFN Firenze)*



Last 3 months

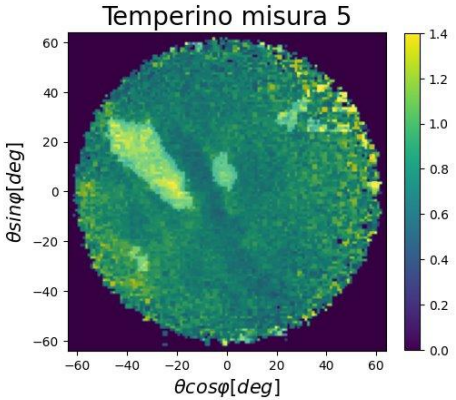
Use case: Unet for the identification of caverns in muography

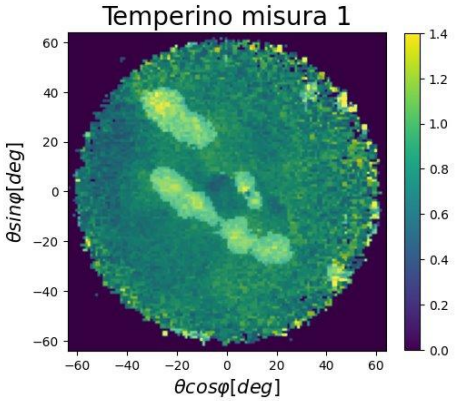
Analysts: *A. Paccagnella, V. Ciulli, C. Frosin (UniFi and INFN Firenze)*

Muography: Imaging technique uses muons to make a radiograph of objects that may be very large

Objective:

- Creating a software capable of detecting and mapping the cavities inside a mine: given a percentage of accuracy.
- Isolating anomalies within a muon radiography
- For this purpose a neural network on the **AI_INFN platform** was created:
  - a very large Dataset has been created for the training (~20K simulated images)
  - a neural network has been developed: a U-Net architecture based on CNN designed for segmenting biomedical images.
  - GPU resources of the platform were used to train and test the NN.
  - Finally, the neural network was tested on real measurements
  - Identification of cavities on a transmission map (target/free-sky).





Ongoing developments

Distributed computing on ***virtual kubelets*** with ***interLink***

From interactive to batch jobs

- Once an analysis or the development of a model is mature, analysts want to scale it on more resources:
  - longer training time than available interactively;
  - freeing interactive resources for development;
  - parallel execution of multiple trials…

vk-dispatcher

➢ We are developing a microservice (vk-dispatcher) translating an interactive session into a Kubernetes Job, executed on the cluster resources.

Development is our priority!
Batch workloads must not affect the interactive use of the platform.

Need for a batch management system, **instantaneously evicting batch** jobs opportunistically.

Kubernetes-native batch system: *Kueue*

Kueue is a set of APIs and a controller meant to simplify and improve job queue management in Kubernetes.

- **Queue management:** Provides a solid infrastructure for job queue management, allowing reliable and scalable execution of jobs inside the Kubernetes cluster.
- **Integration with Kubernetes resources:** Kueue integrates natively with Kubernetes' resources and functionality, making use of orchestration and management features of the cluster.
- **Monitoring and Scalability:** Thanks to dedicated controllers, Kueue simplifies monitoring of job state and allows to scale resources automatically based on workload.

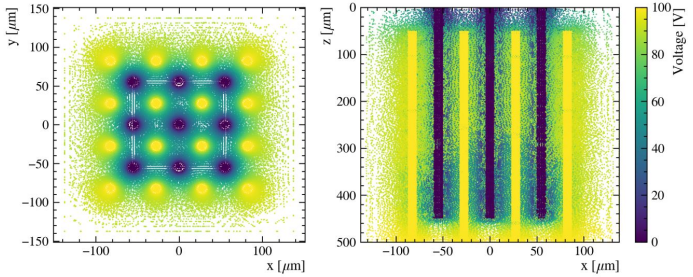
vk-dispatcher + Kueue were alpha-tested with three different applications.
Effective for analysis workflows combining CPU-only and GPU-powered steps.

Optimizing the fabrication of 3D diamond detectors with Physics Informed Neural Networks on Kubernetes

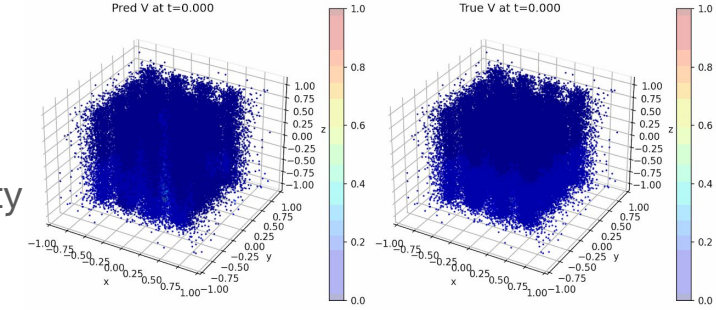
Analysts: *Clarissa Buti and Alessandro Bombini (INFN Firenze)*

- An extension of the Ramo-Shockley theorem is used to study the effect of induced currents on resistive electrodes
- Creation and study of a neural network for the resolution of differential equations (PINN) to compute time-dependent potential maps (ICSC-Spoke 2, partnership with ENI) using:
  - Python scripts with **NVIDIA Modulus**: a framework for building, training, and fine-tuning Physics-ML models with a simple Python interface
- Conversion of the models into a C++ simulation of the 3D diamond detectors based on the ROOT-based **Garfield++** software packages for the detailed simulation of gas and semiconductor detectors
- Use of the simulation to study the contribution to the uncertainty of the timing measurement of the 3D diamond detectors from highly-resistive electrodes

Potential of the simulated sensor



Use of batch features (vkd)



Up to **50 CPU cores, 100 GB of RAM and 6 GPUs** are used opportunistically

Offloading: InterLink & Virtual Kubelet

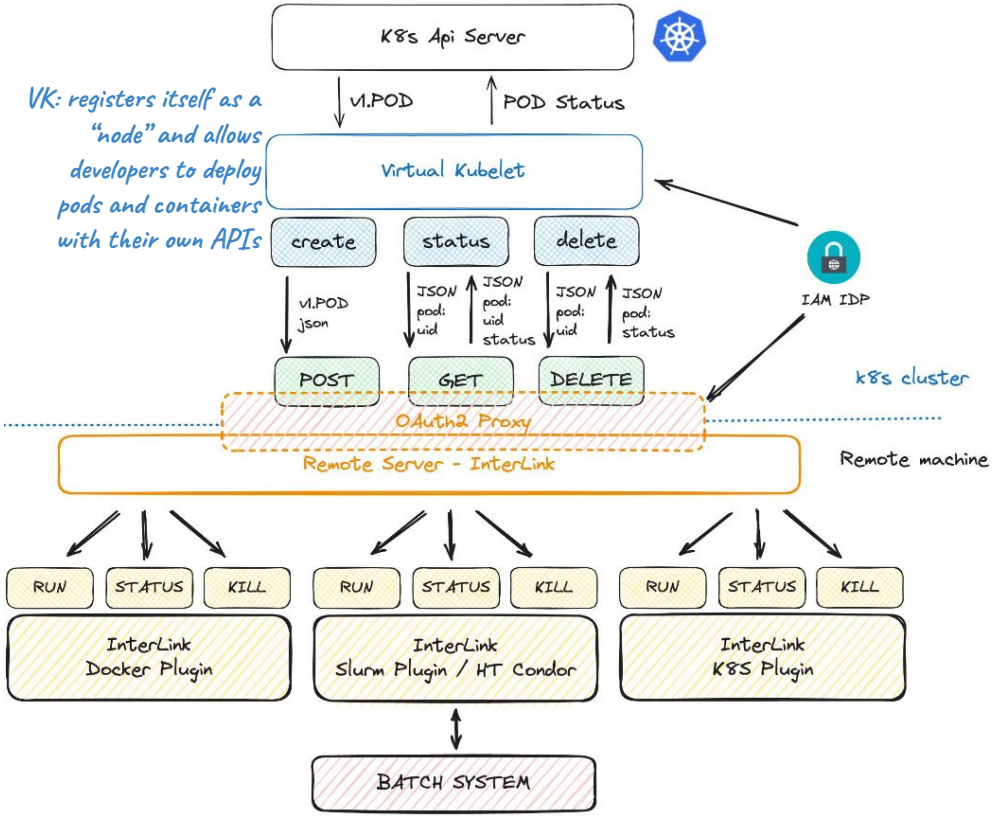
Contact persons: *Giulio Bianchini, Diego Ciangottini, Daniele Spiga et al. (INFN Perugia)*

interLink

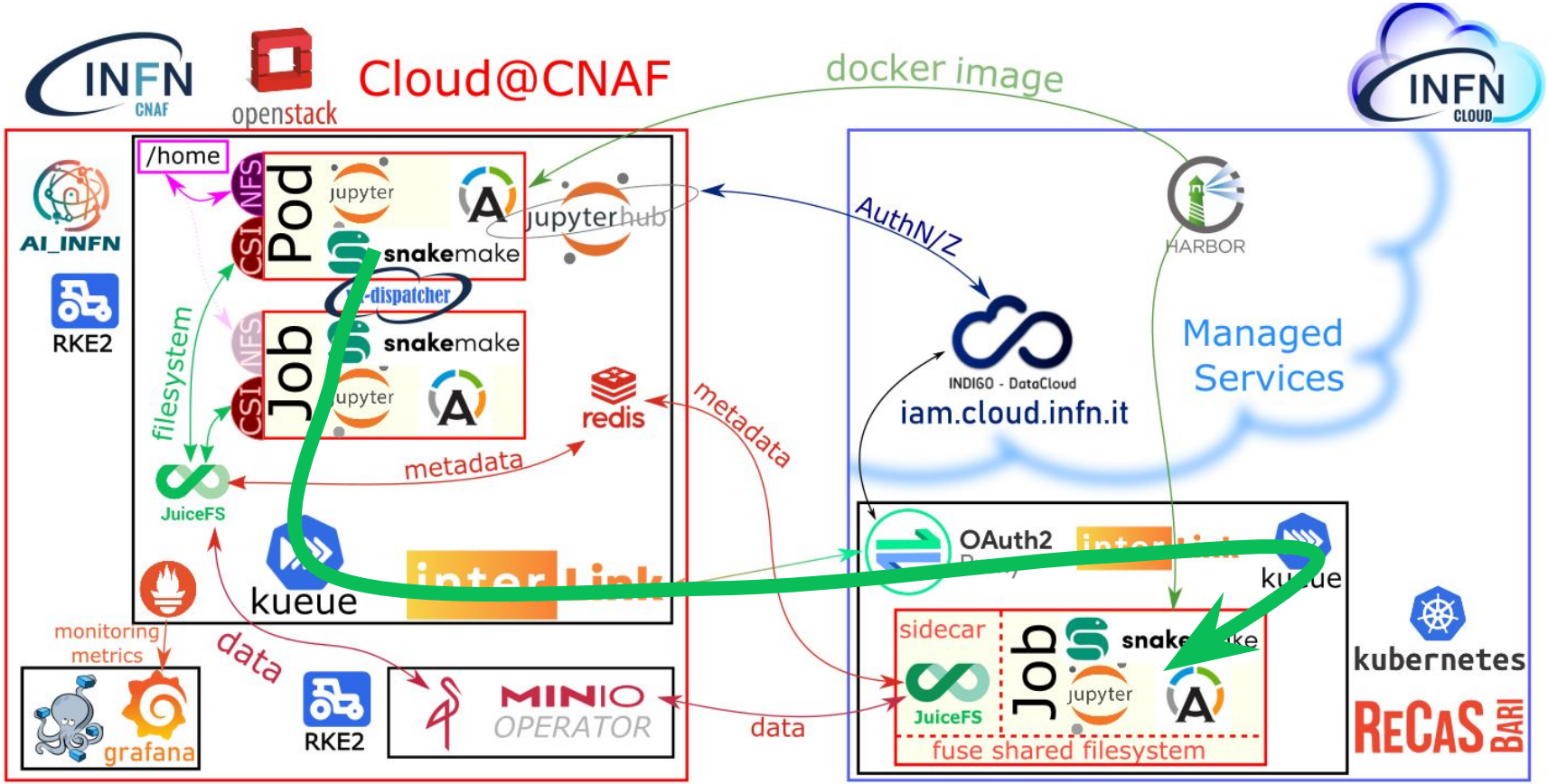
Provide an abstraction for the execution of a Kubernetes pod on any remote resource capable of managing a Container execution lifecycle.

The project consists of 3 main components:

- **Kubernetes Virtual Node**: based on the VirtualKubelet technology. Translating request for a kubernetes pod execution into a remote call to the interLink API server.
- **InterLink API server**: a modular and pluggable REST server that can handle requests coming from the VK and forwarding them to the sidecar;
- **Sidecar**: runs the containers on the infrastructure and returns the result. It can also communicates with the InterLink server.



Offloading the workload with interLink
A proof-of-concept tested on May 16 (3 week ago)



Enabling Technology for *virtual kublets*

The proof of concept shows that:

- We can interface *virtual kubelets* with complicated **workflow management tools** such as **Snakemake**;
- We can distribute a **filesystem through jobs** (rather than through computing nodes) using the **sidecar mechanism**;
- **Combining local and remote resources** in a workflow is feasible and (with some more work) can be made transparent to the user.

Next step: **distribute some realistic, CPU-intensive, workloads**.
Natural candidate: ***LHCb Flash Simulation (Lamarr).***

Conclusion & Outcomes

Automation, documentation and security

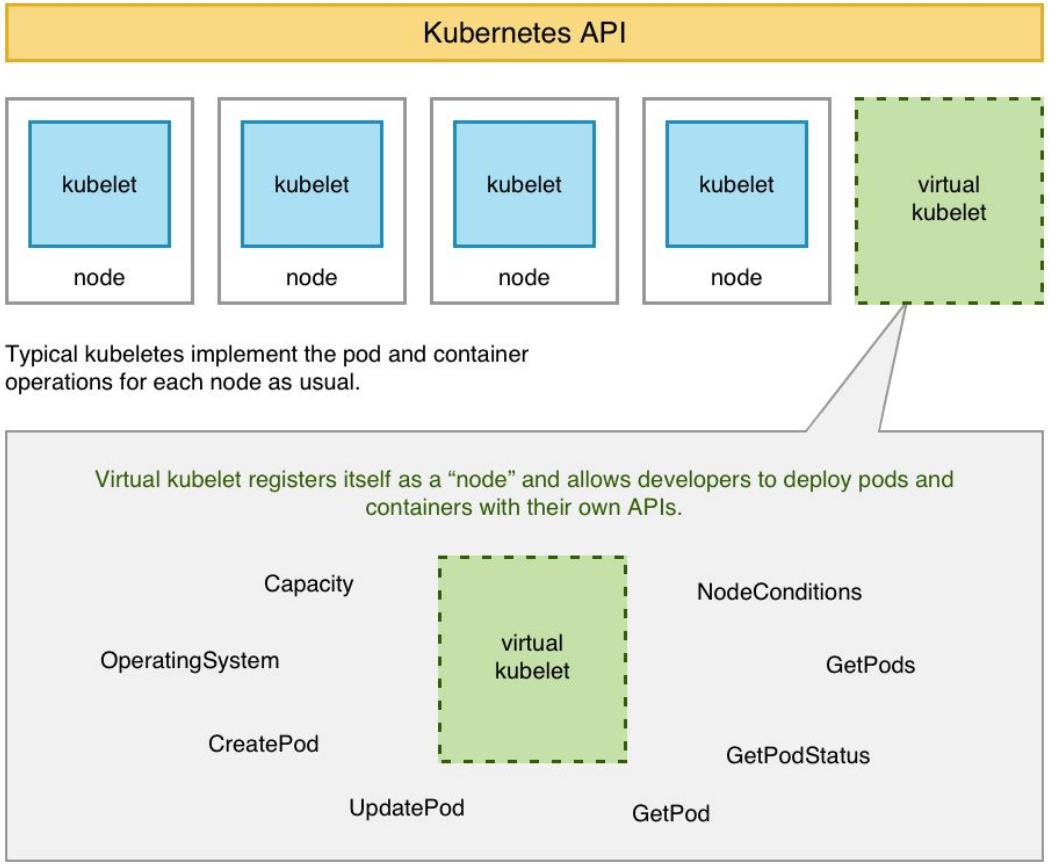
Conclusion & Outcomes

The AI_INFN platform is an intense R&D program with the aim of provide a more effective tool for developing Machine Learning and Artificial Intelligence for INFN researches.

- We aim for an Infrastructure-as-Code approach to replicate the platform easily on multiple setup and keep it updated;
- Our FAQ page will be evolved in a more complete documentation;
- More attention must be devoted to user's data management, improving our backup solutions and reviewing encryption of data transferred through multiple sites;
- We need to set up more safety procedures to update the various components without compromise the service.

Thank you for
your attention!

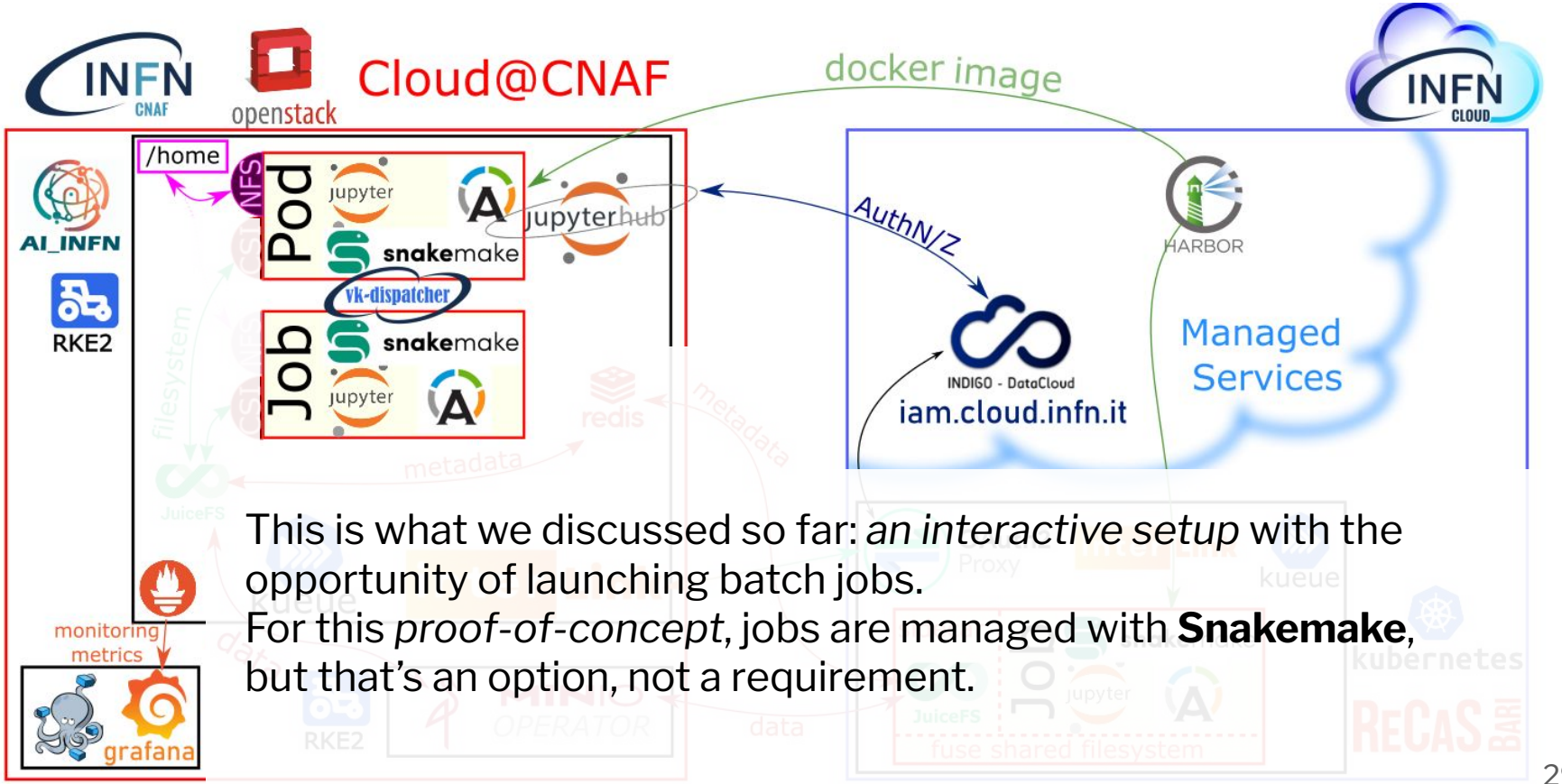
Offloading: InterLink & Virtual Kubelet



Conclusion

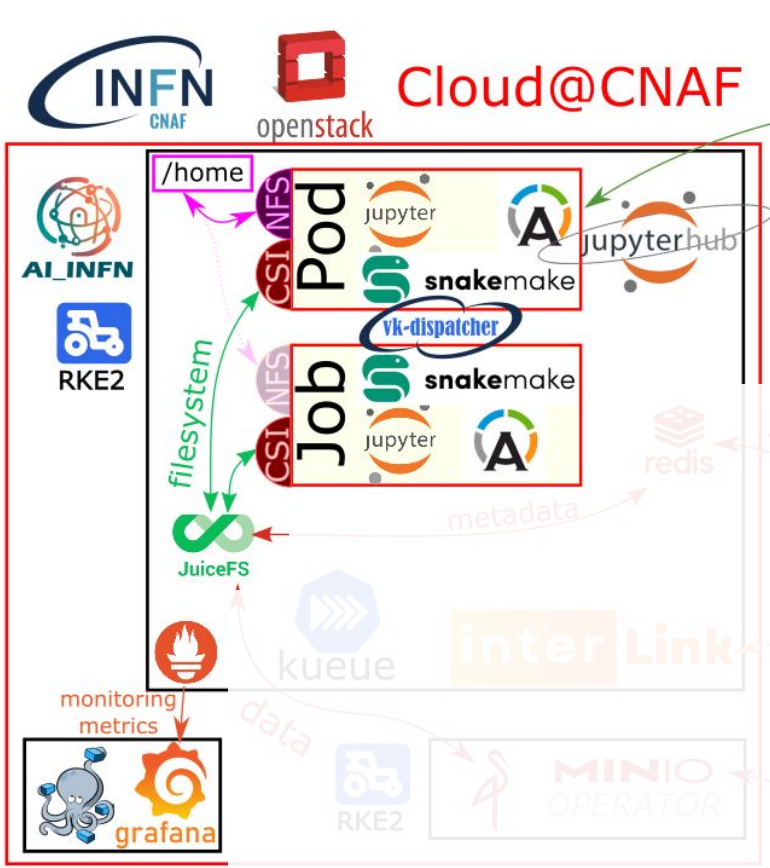
Feature	Proof of concept	Beta-tested in hub.ai	Available for all users	Ready for DataCloud
Interactive development (GPU)	2023-05-18	2023-12-13	2024-03-08	⏳
Interactive develop. (QC/FPGA)	*QC coming soon*	⏳	⏳	⏳
Monitoring	2024-03-18	2024-04-22	2024-05-13	⏳
Accounting	2024-03-18	*coming soon*	⏳	⏳
Batch job submission	2023-12-19	2024-04-18	⏳	⏳
Offloading towards Kueue	2024-05-16	⏳	⏳	⏳
Offloading to Docker (GPU)	*coming soon*	⏳	⏳	⏳

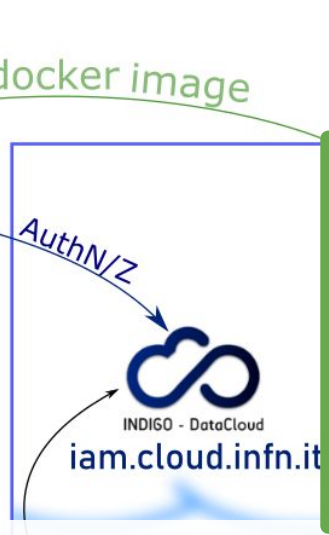
Offloading the workload with interLink
A proof-of-concept tested on May 16 (3 week ago)

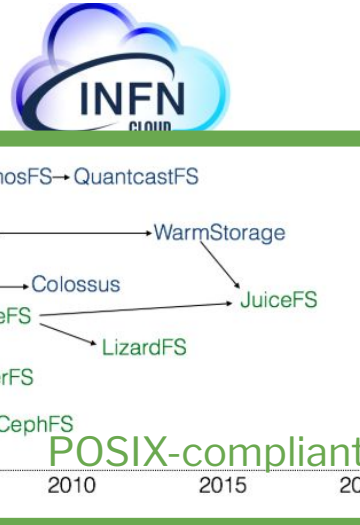


This is what we discussed so far: *an interactive setup* with the opportunity of launching batch jobs.
For this *proof-of-concept*, jobs are managed with **Snakemake**, but that's an option, not a requirement.

Offloading the workload with interLink
A proof-of-concept tested on May 16 (3 week ago)

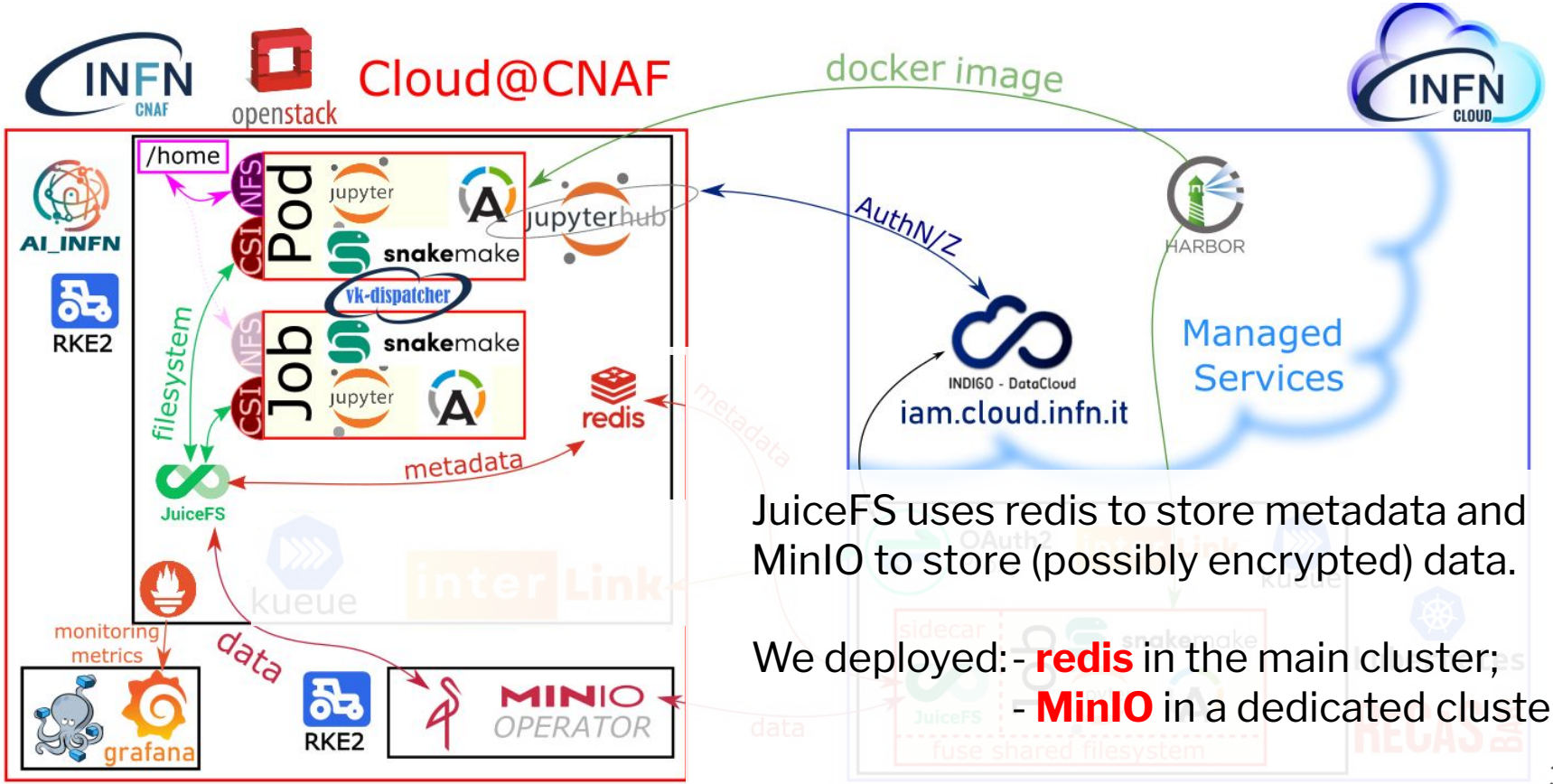






To get Snakemake cross the borders of the clusters, a distributed filesystem is needed. We picked **JuiceFS** which is simple to configure *(e.g. Container Storage Interface available in helm)* and suitable for serveless container applications.

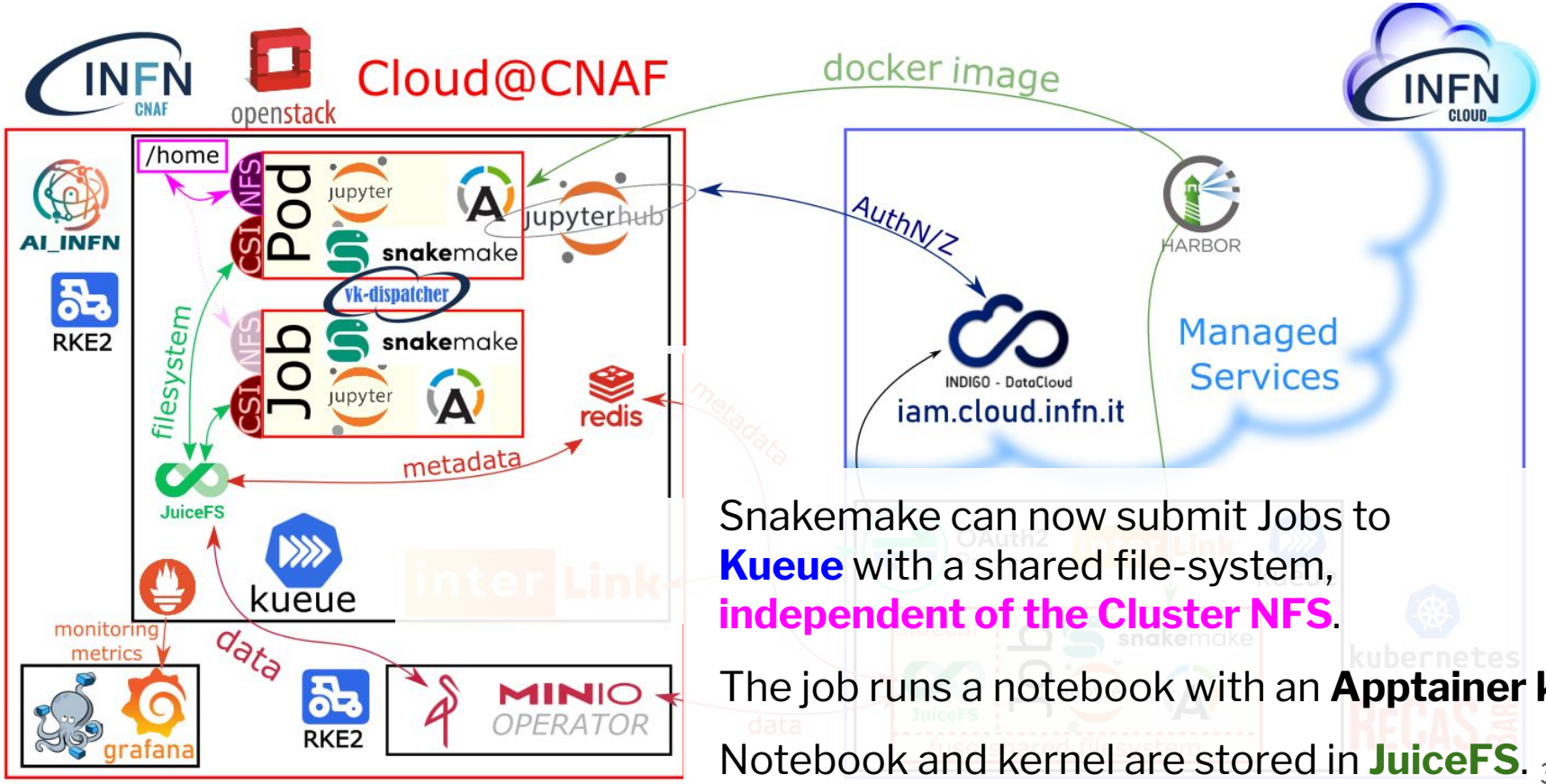
Offloading the workload with interLink
A proof-of-concept tested on May 16 (3 week ago)



JuiceFS uses redis to store metadata and MinIO to store (possibly encrypted) data.

We deployed: - **redis** in the main cluster;
- **MinIO** in a dedicated cluster.

Offloading the workload with interLink
A proof-of-concept tested on May 16 (3 week ago)

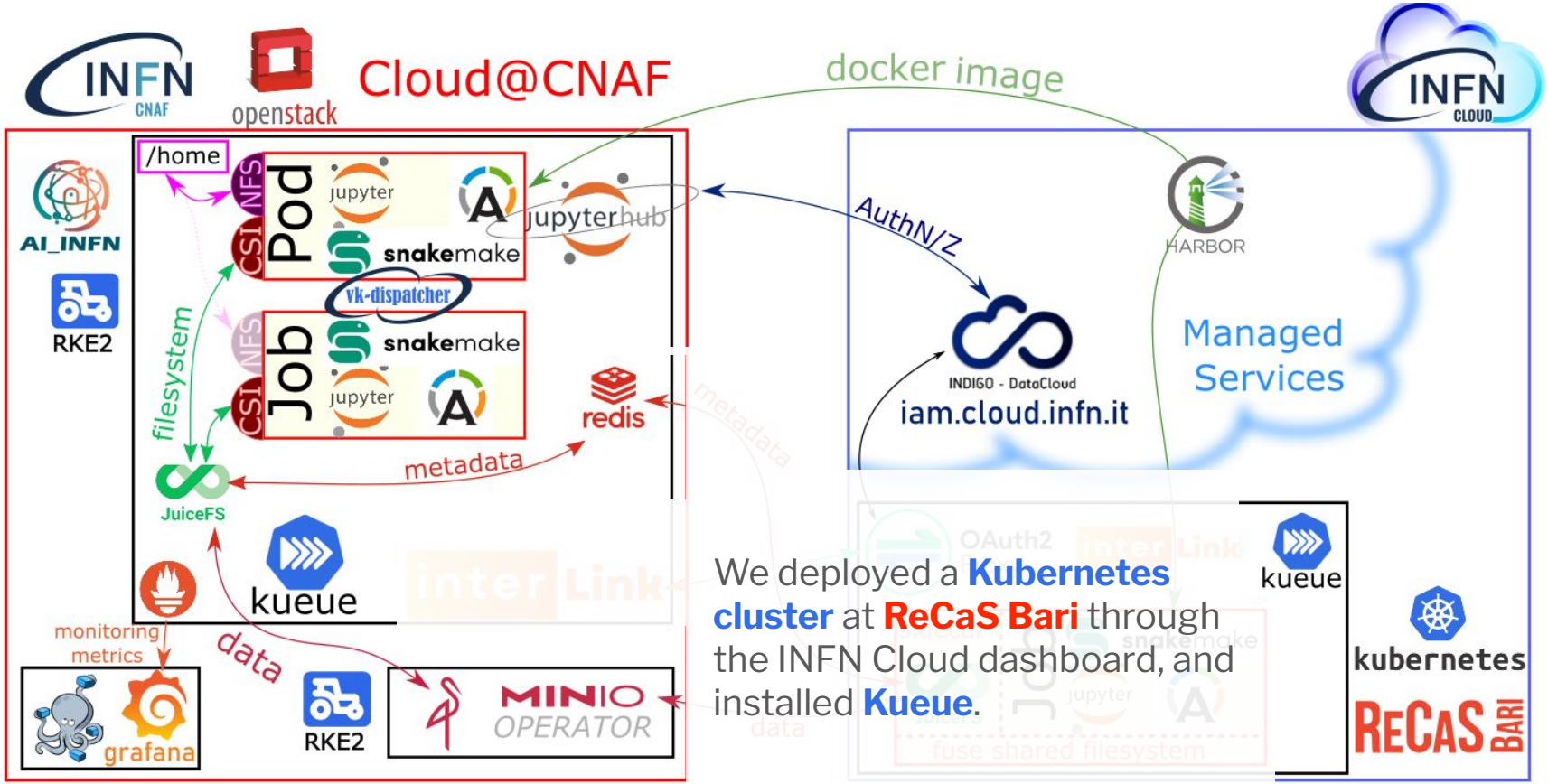


Snakemake can now submit Jobs to **Kueue** with a shared file-system, **independent of the Cluster NFS**.

The job runs a notebook with an **Apptainer kernel**.

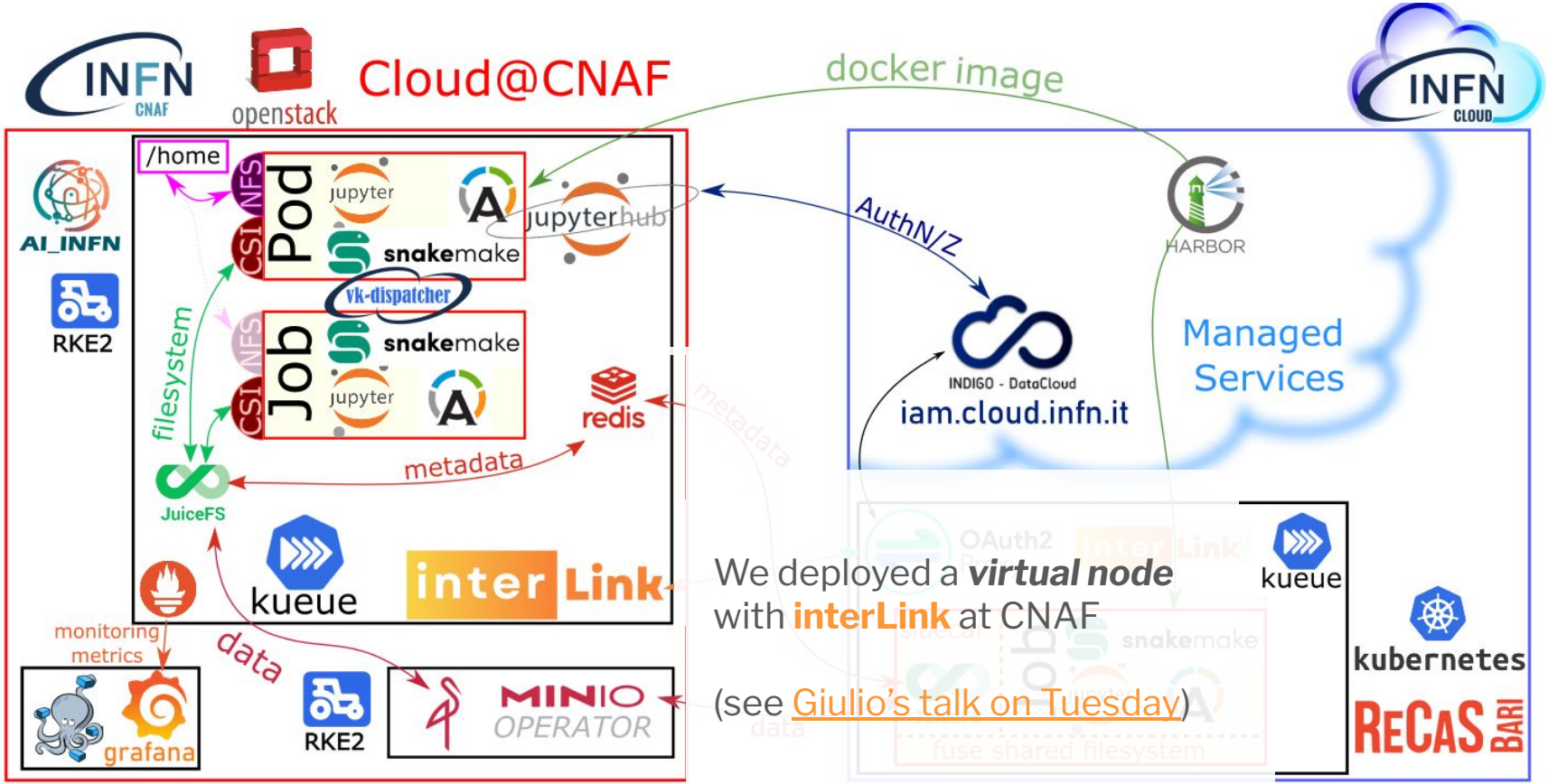
Notebook and kernel are stored in **JuiceFS**.

Offloading the workload with interLink
A proof-of-concept tested on May 16 (3 week ago)



We deployed a **Kubernetes cluster** at **ReCaS Bari** through the INFN Cloud dashboard, and installed **Kueue**.

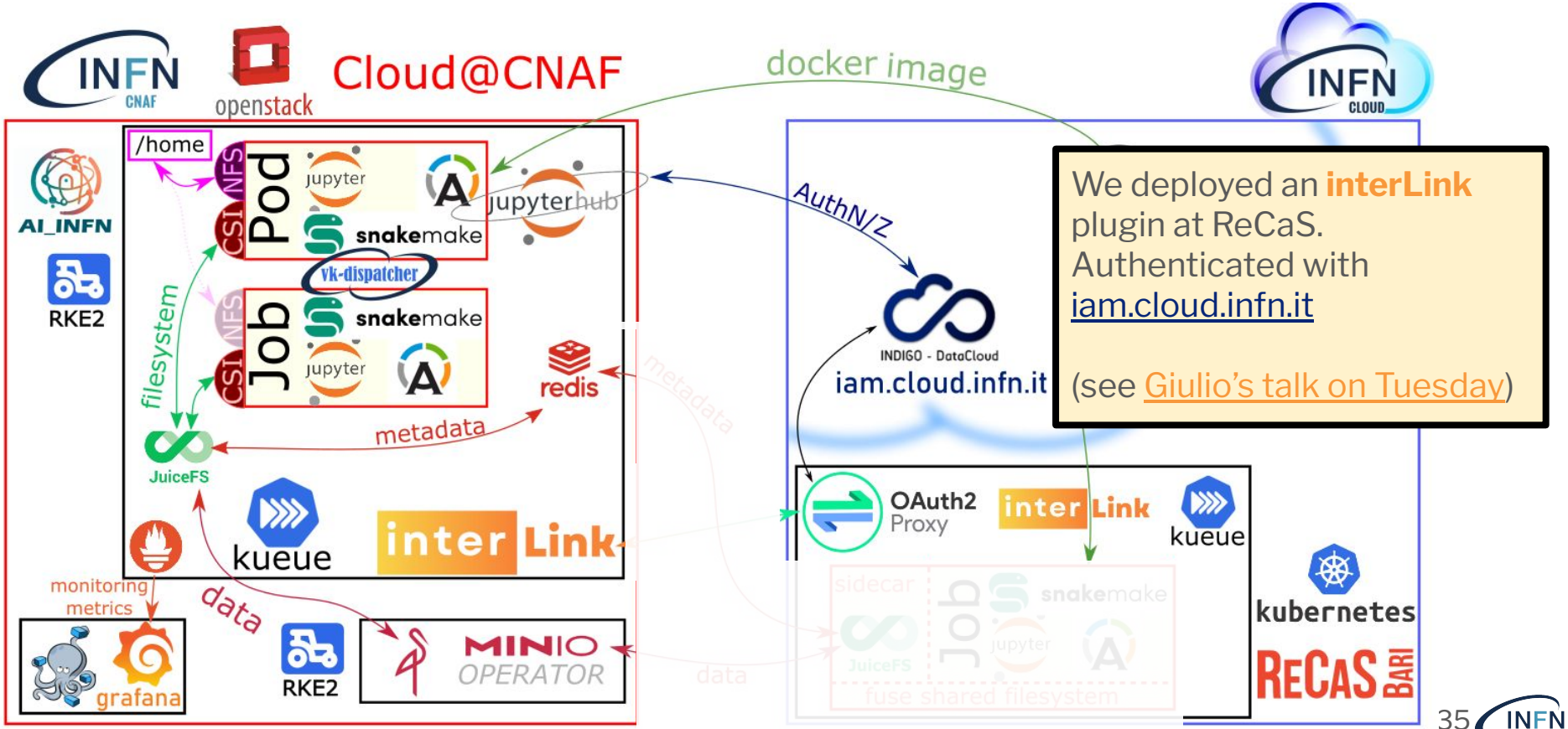
Offloading the workload with interLink
A proof-of-concept tested on May 16 (3 week ago)



We deployed a ***virtual node*** with **interLink** at CNAF

(see Giulio's talk on Tuesday)

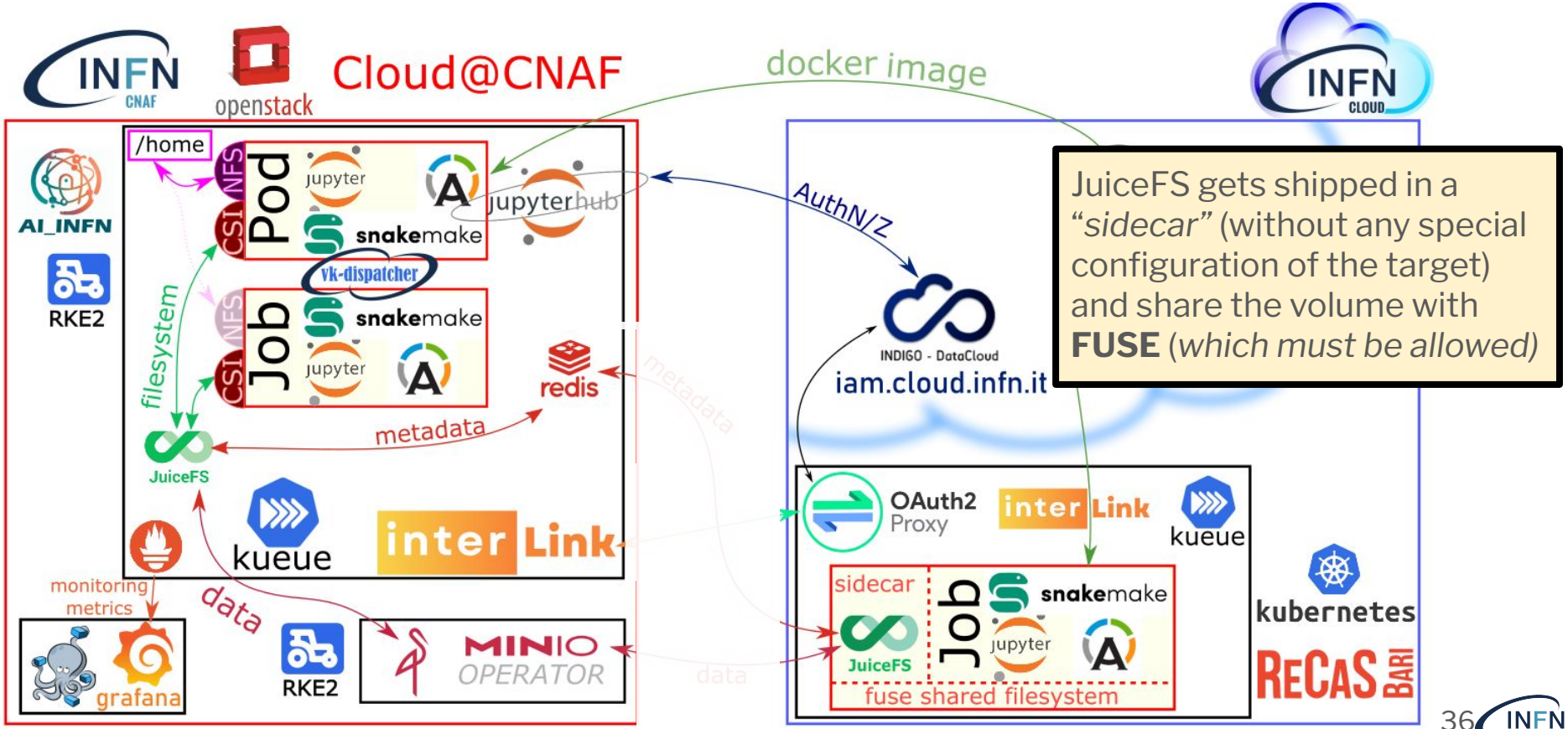
Offloading the workload with interLink
A proof-of-concept tested on May 16 (3 week ago)



We deployed an **interLink** plugin at ReCaS. Authenticated with iam.cloud.infn.it

(see Giulio's talk on Tuesday)

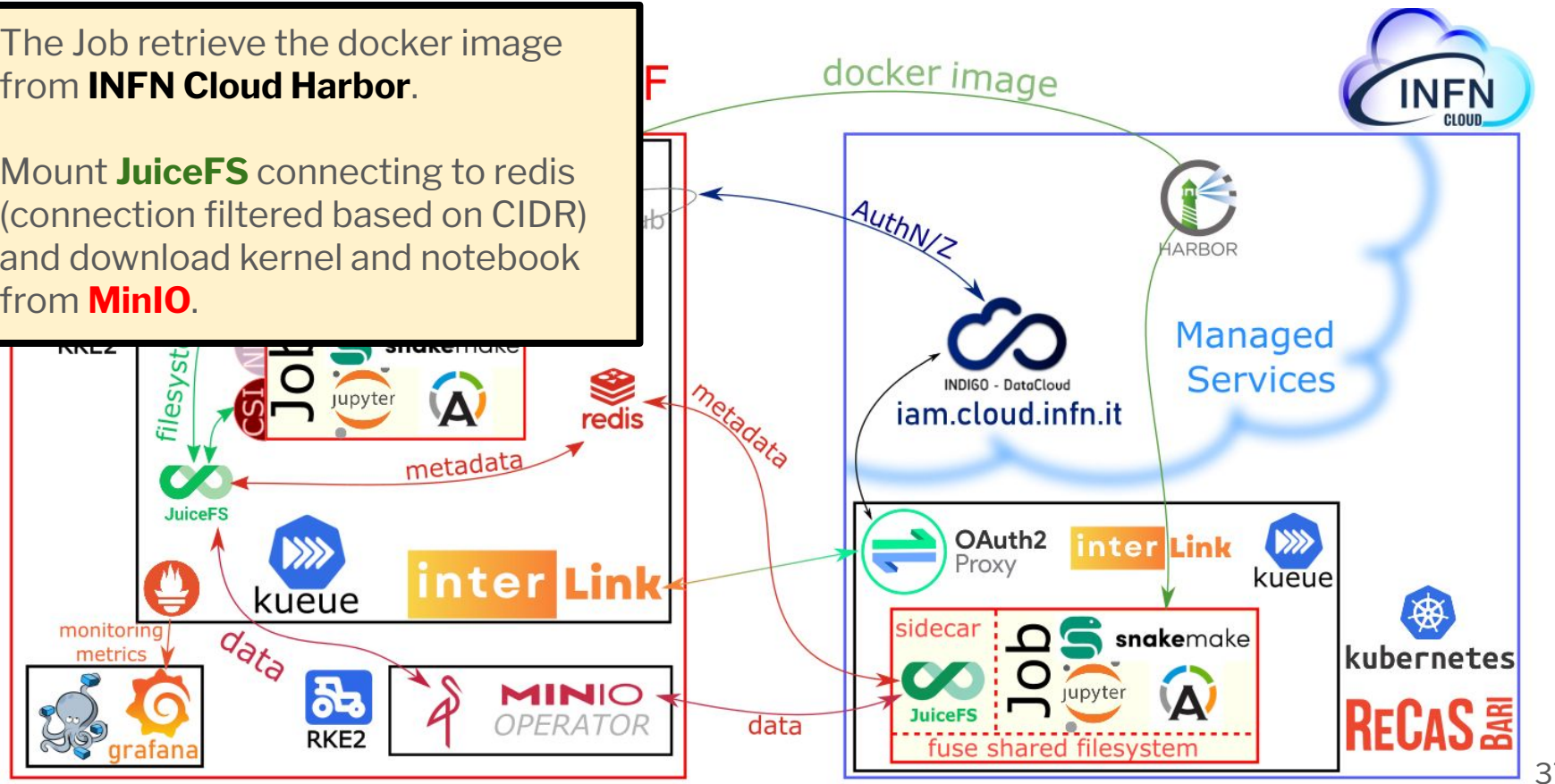
Offloading the workload with interLink
A proof-of-concept tested on May 16 (3 week ago)



Offloading the workload with interLink
A proof-of-concept tested on May 16 (3 week ago)

The Job retrieve the docker image from **INFN Cloud Harbor**.

Mount **JuiceFS** connecting to redis (connection filtered based on CIDR) and download kernel and notebook from **MinIO**.



Offloading the workload with interLink
A proof-of-concept tested on May 16 (3 week ag

Once completed, the job **uploads the output** and **updates the job status** via **JuiceFS**, and **Snakemake** marks the task as succeeded and submit another job (if any).