

# Creating a rucio service for km3net

7th Rucio Community Workshop

## Contact Persons



Bouwe Andela  
Senior Research Software Engineer  
[b.andela@esciencecenter.nl](mailto:b.andela@esciencecenter.nl)



Victor Azizi  
Research Software Engineer  
[v.azizi@esciencecenter.nl](mailto:v.azizi@esciencecenter.nl)

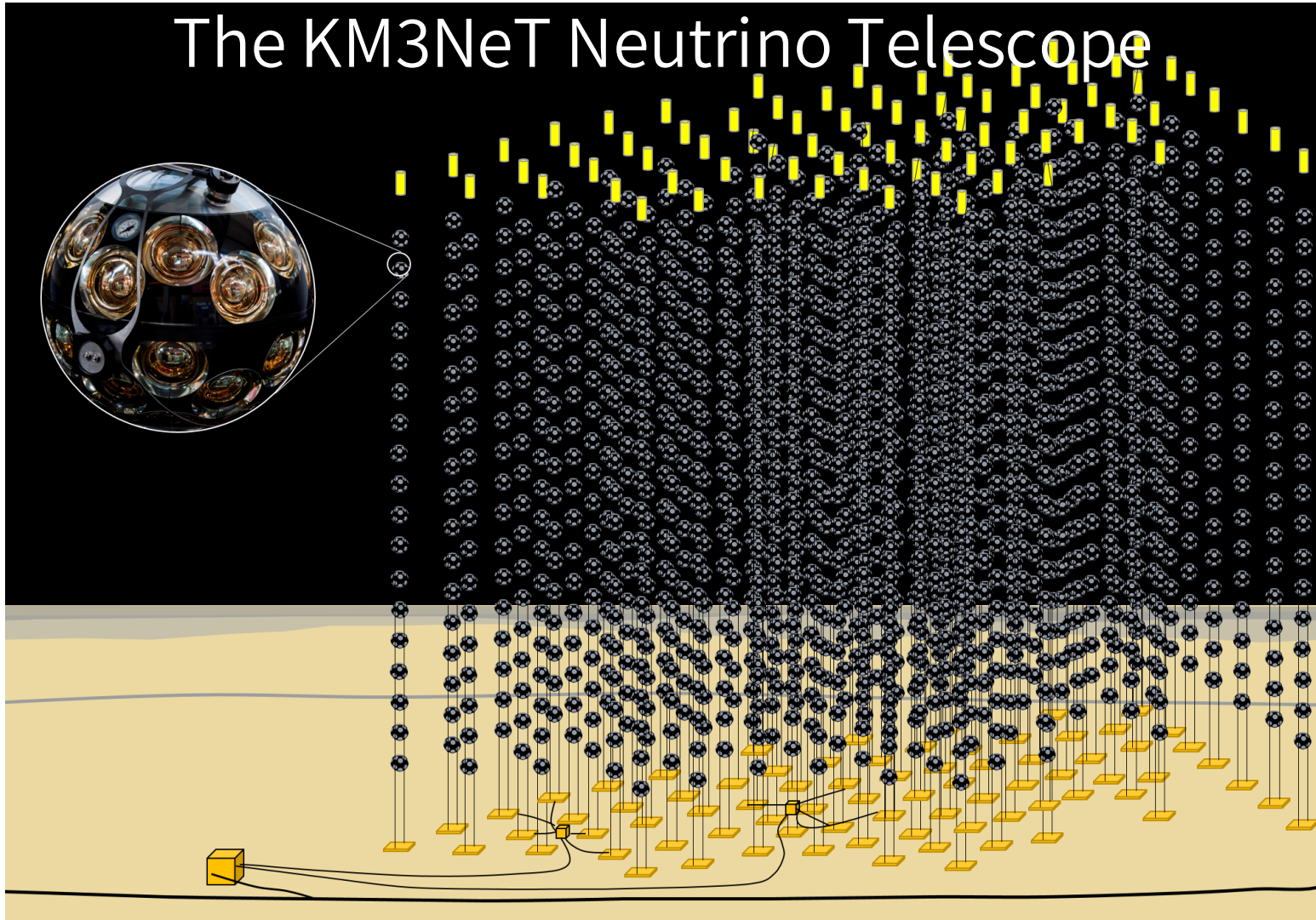
netherlands  
**eScience center**

# Overview

1. What is KM3NeT?
  - How does KM3NET want to use rucio
2. Set up of rucio for KM3NeT
3. Future work for rucio with KM3NeT
4. Questions

**What is KM3NeT**

# The KM3NeT Neutrino Telescope



Distributed infrastructure  
in the Mediterranean Sea

FR:  $10^7 \text{ m}^3$

IT:  $1 \text{ km}^3$

Modular telescope array

345 vertical strings

200.000 light-sensitive  
photo-multiplier tubes

Today:

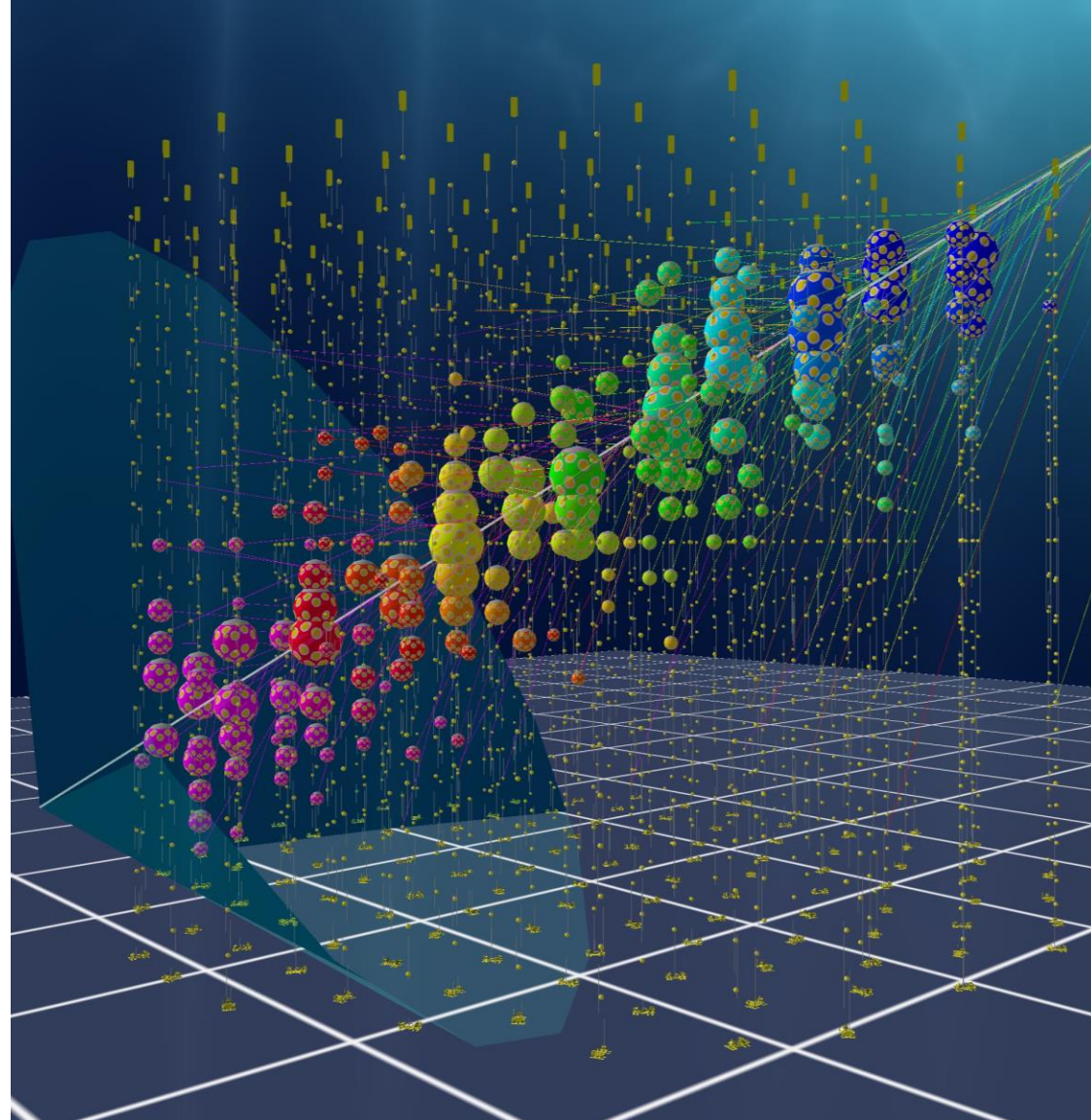
FR: 10% operational

IT: 5% operational



# Simulations in KM3NeT

- Events are
  - Atmospheric muons 200 Hz
  - Atmospheric neutrinos 10-100 day<sup>-1</sup>
  - Cosmic neutrinos 10-100 year<sup>-1</sup>
- Expected "raw" data volume 0.5PB year<sup>-1</sup>
- Simulations 10x data volume
  - Expected data volume of 2PB year<sup>-1</sup>
- Rucio will be used for data management of real data and simulated data on the grid



# Some More Numbers

- Once detector is fully built, computing needs expected to grow into an average of 1-2k cores / jobs running simultaneously, with peaks when doing mass-reprocessings of data.
- The new version of the data processing, which uses snakemake as a workflow manager, is integrated with Rucio for input and output handling on the Grid.





# The KM3NeT Computing Model

Tier-2

data  
analysis

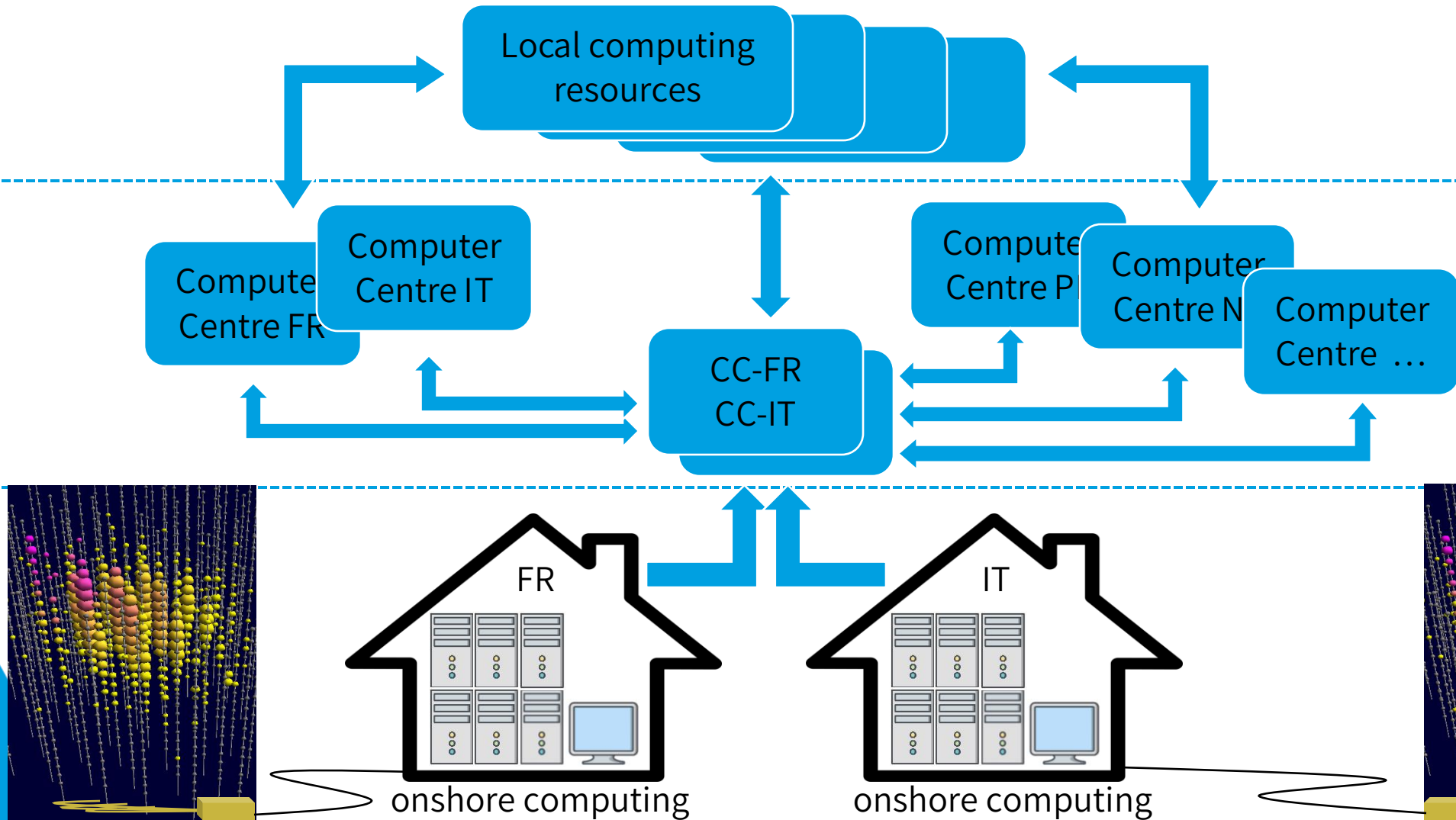
Tier-1

data  
processing

data  
storage

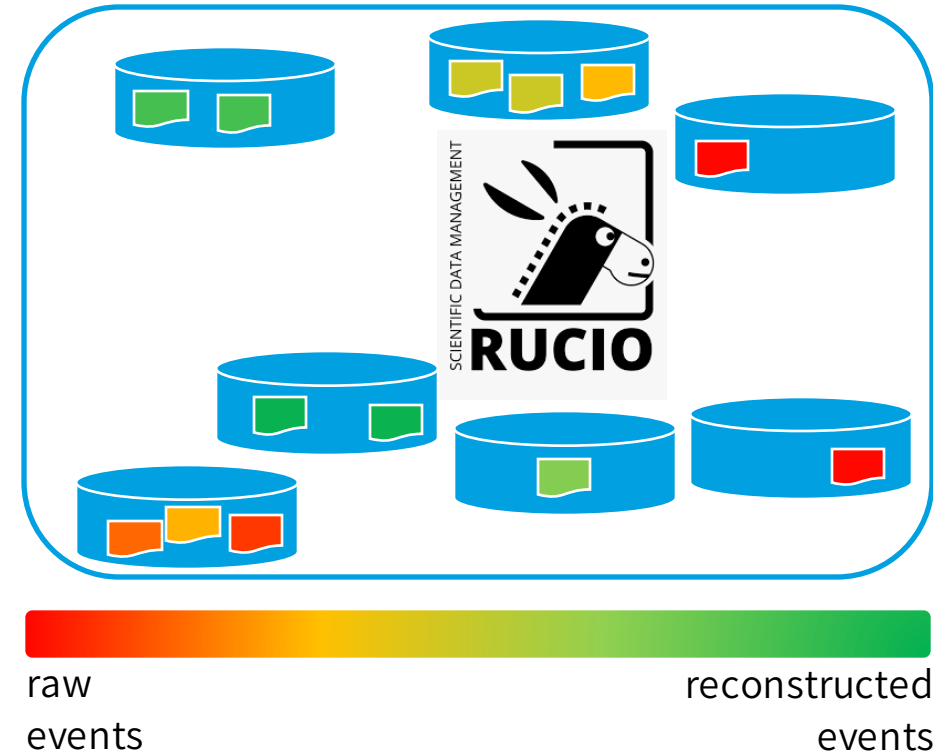
Tier-0

data  
acquisition



# KM3NeT Tier-1 Level

- Calibration of raw events
- “Reconstruction” of calibrated events
- Rucio should handle storage of calibrated and reconstructed data
  - Accessible to all KM3NeT partners
  - Distribution of the data processing
  - Organization in data sets





# The role of the eScience Center

- Set up a data management service
- Help with integration into their processing workflows
- Develop training material and deliver user trainings
- Project runs until end of 2025



# Set up of rucio for KM3NeT



# Document everything

<https://rucio.pages.km3net.de/rucio-documentation/>

## Installation instructions for Rucio

Rucio can be installed on a Kubernetes cluster. This document describes how to set up a Kubernetes cluster for testing and how to install Rucio on it.

### Setting up a Kubernetes cluster for testing purposes

There are several Kubernetes clusters available that can be run on a laptop or a single machine. Popular choices are:

- [microk8s](#)
- [k3s](#)
- [minikube](#) (requires Docker)

### Installing and configuring microk8s

Install microk8s according to the instructions [here](#).

Ensure container images are stored somewhere with enough disk space. They are stored in `/var/snap/microk8s/common/var/lib/containerd/`. If there is not enough space at that location, move it somewhere else and create a symlink from the old location to the new location.

#### Start the cluster

#### Table of contents

Setting up a Kubernetes cluster for testing purposes

Installing and configuring microk8s

Start the cluster

Check status of cluster

Configure the firewall, can be skipped usually

Enable local storage and ingress

Install Rucio and related software using flux

Working with the Rucio database

Bootstrapping the database

Backing up and restoring an existing database

Updating an existing database to a newer version of Rucio

Test if the server works

Accessing Rucio

Update who has access to the root account

Enable SSL passthrough

Enable outside access to 5000 port for authentication and 5001 for webui



# Repositories

<https://git.km3net.de/rucio>

Everything is publicly accessible, except the Secrets repository.



KM3NeT

Search or go to...

Group

- R Rucio
- Pinned
- Issues 19
- Merge requests 0
- Manage
- Plan
- Code
- Build
- Deploy
- Operate
- Settings

Rucio

## R Rucio

Home for repositories related to the KM3NeT Rucio deployment.

Subgroups and projects Shared projects Inactive

Search (3 character minimum)

R Rucio Deployment

K KM3NeT Rucio Containers

R Rucio Deployment Secrets Owner

R Rucio documentation

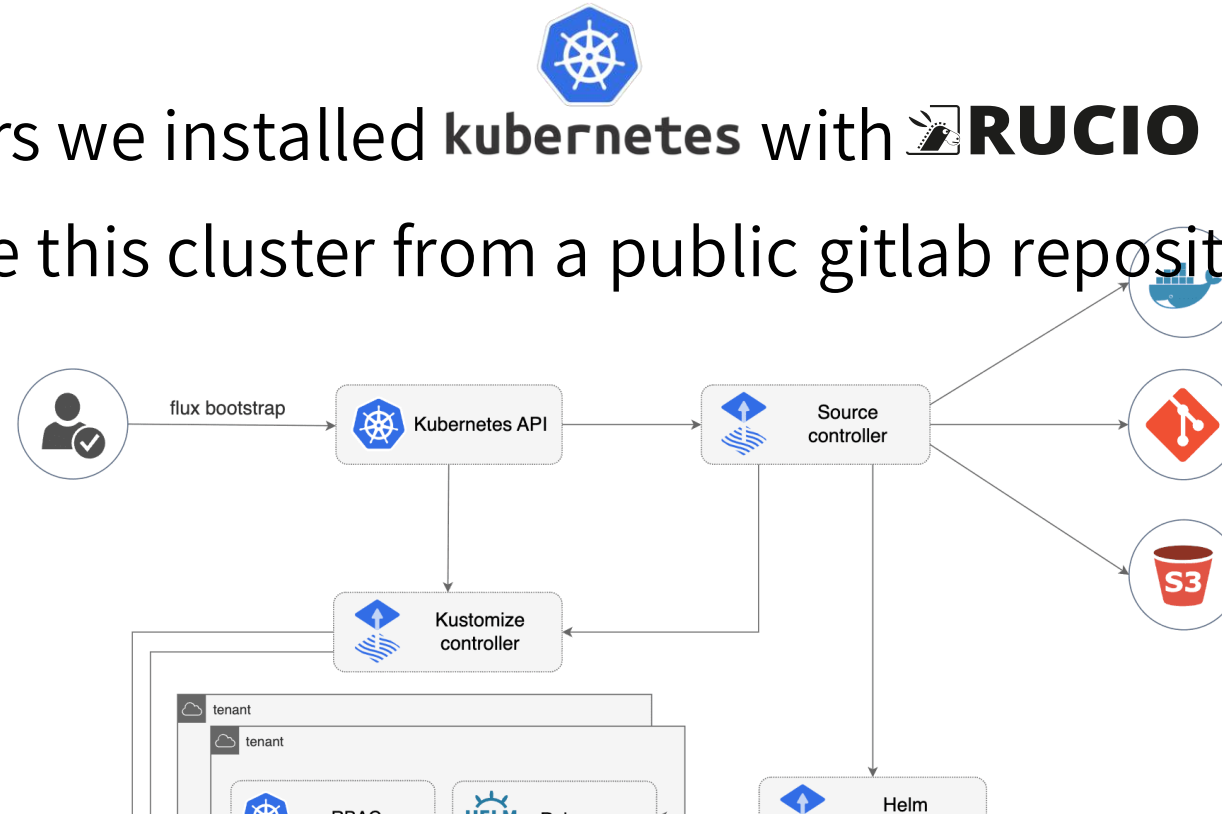
K km3net\_rucio\_policy

# Setting up the data management service

- We got 2 very nice (virtual) servers from Nikhef
  - One for testing, one for production



- On these servers we installed **kubernetes** with **RUCIO**
- We auto update this cluster from a public gitlab repository with flux



# Using rucio helm charts to get started

- Update the documentation to include easy database generation


- How to manage the helm charts?


- There are many helm charts that need the same values


- **Solution:** Just put the helm charts in a helm chart!


- We created a helm chart that generates all our rucio helmcharts (and everything else we need)

 helmrelease-postgresql.yaml

 helmrelease-rucio-daemons.yaml

 helmrelease-rucio-server.yaml

 helmrelease-rucio-ui.yaml

 helmrelease-rucio-webui.yaml

 helmrelease-sealed-secrets.yaml





# Our rucio helmcharts chart values

helmrelease-rucio-deployment.yaml 1.28 KIB

```

1  apiVersion: helm.toolkit.fluxcd.io/v2beta2
2  kind: HelmRelease
3  metadata:
4    name: rucio-deployment
5    namespace: default
6  spec:
7    dependsOn:
8      - name: cert-manager
9    chart:
10     spec:
11       chart: rucio-deployment-chart
12       reconcileStrategy: ChartVersion
13       sourceRef:
14         kind: GitRepository
15         name: flux-system
16         namespace: flux-system
17     interval: 10m0s
18     values:
19       # Important! First host must match the auth_host used by the rucio config, the other will
20       hostnames:
21         - host: rucio.km3net.org
22           issuer: sectigo-issuer
23         - host: rucio.km3net.de
24           issuer: letsencrypt-issuer
25       postgresql_version: 15.6.0
26       postgresql_backup_version: postgres:15.6-alpine
27       rucio_server_version: release-34.4.3
28       rucio_daemons_version: release-34.4.3
29       rucio_webui_version: release-34.0.0
30       rucio_ui_version: release-34.4.3
31       rucio_server_helm_chart_version: "rucio-server-34.0.3"
32       rucio_daemons_helm_chart_version: "rucio-daemons-34.0.4"
33       rucio_webui_helm_chart_version: "rucio-webui-34.0.3"
34       rucio_ui_helm_chart_version: "ui_additionalenvs"
35       update_grid_certificates: true
36       policy_package_location: "git+https://git.km3net.de/rucio/km3net_rucio_policy.git@main"

```

ui helm-chart as template:

```

7  spec:
8    chart:
9      spec:
10       chart: charts/rucio-ui
11       reconcileStrategy: ChartVersion
12       sourceRef:
13         kind: GitRepository
14         name: rucio-ui
15       interval: 12h0m0s
16       values:
17
18       proxy:
19       {{- with (first .Values.hostnames) }}
20         rucioProxy: {{ .host }}
21         rucioProxyScheme: "https"
22         rucioAuthProxy: {{ .host }}:5000
23         rucioAuthProxyScheme: "https"
24       {{- end }}
25       replicaCount: 1
26       image:
27         tag: {{ .Values.rucio_ui_version }}
28       service:

```

# What about automatic deployments

## Candidates:

- gitlab actions (push approach)
- via [flux](#) (pull approach)



# Setting up flux

1. Run `flux bootstrap` to install flux in the cluster
2. Flux will apply our helmrelease and check for updates every 5 minutes
3. Our helmrelease installs all our services, including rucio

## Benefits

- If we push to our deployment repository, the cluster tries to update
- If the upgrade fails, the cluster stays at the latest working version
- We can easily create multiple deployments in the same git repository
- Flux can follow a specific branch,
  - o We can test on a `develop` branch and a test deployment before merging to `main`

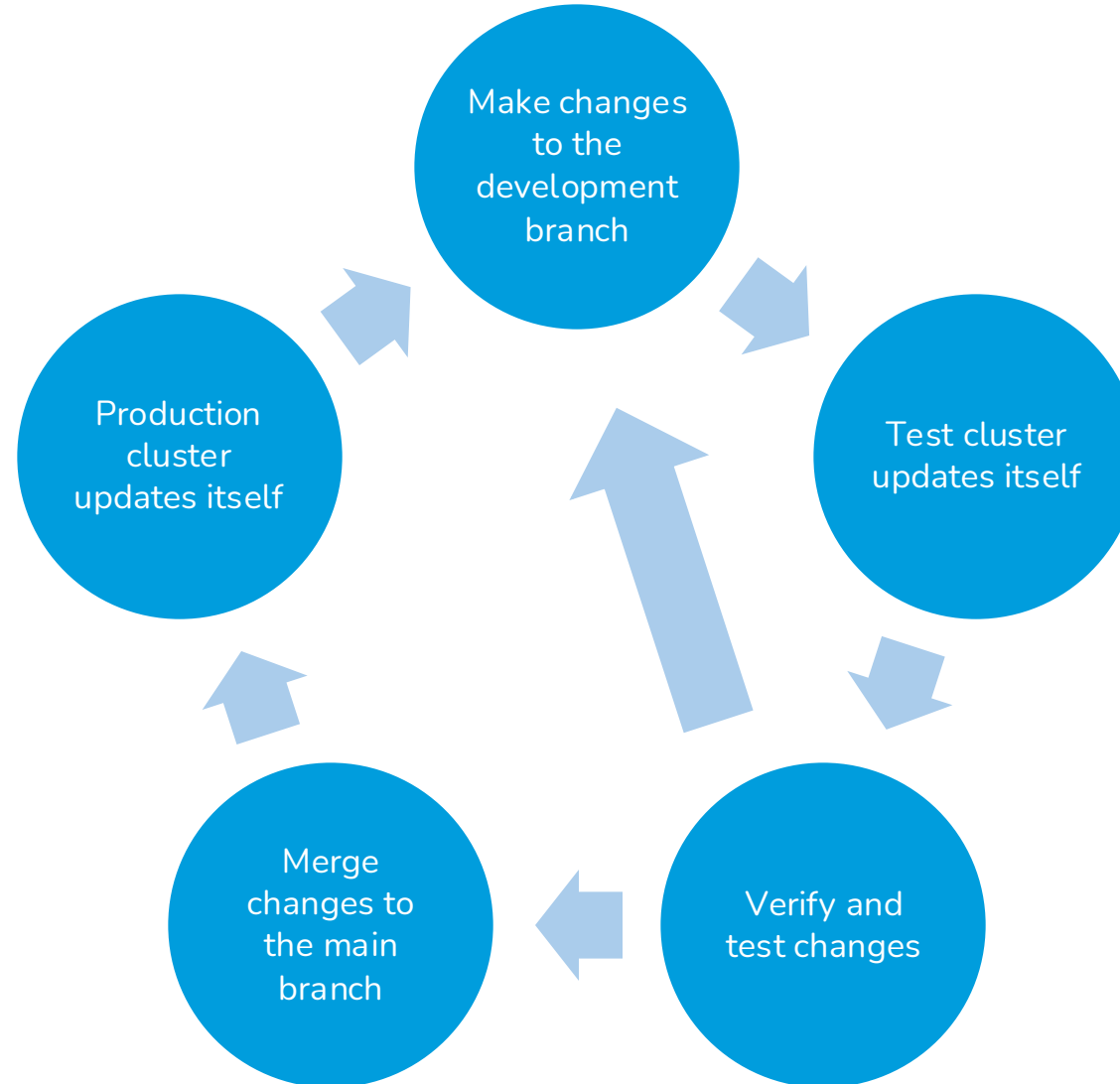
```
clusters
├── production
│   ├── cert-manager.yaml
│   ├── flux-system
│   │   ├── gotk-components.yaml
│   │   ├── gotk-sync.yaml
│   │   └── kustomization.yaml
│   ├── helmrelease-rucio-deployment.yaml
│   └── volumes.yaml
├── testing
│   ├── cert-manager.yaml
│   ├── flux-system
│   │   ├── gotk-components.yaml
│   │   ├── gotk-sync.yaml
│   │   └── kustomization.yaml
│   ├── helmrelease-rucio-deployment.yaml
│   └── volumes.yaml
├── docs
│   └── index.md
├── LICENSE
├── mkdocs.yaml
├── README.md
├── rucio-deployment-chart
│   ├── Chart.yaml
│   └── templates
│       ├── cert-manager.yaml
│       ├── cron-job-backup-cleanup.yaml
│       ├── cron-job-grid-certificates.yaml
│       ├── cron-job-register-new-data.yaml
│       ├── cron-job-update-policy-package.yaml
│       ├── helmrelease-postgresql.yaml
│       ├── helmrelease-rucio-daemons.yaml
│       ├── helmrelease-rucio-server.yaml
│       ├── helmrelease-rucio-ui.yaml
│       ├── helmrelease-rucio-webui.yaml
│       ├── helmrelease-sealed-secrets.yaml
│       ├── nginx.yaml
│       ├── source-charts-bitnami.yaml
│       ├── source-charts-rucio.yaml
│       └── source-charts-sealed-secrets.yaml
└── values.yaml
```

# Secrets management

- Separate private repository with all sensitive information.
  - FTS certificate
  - SSL certificate renewal credentials
  - Database passwords
  - Etc ...
- Script to push these secrets to the kubernetes cluster, separate from flux.



# Development flow



# Future work





# (possible) Integration with DIRAC

- DIRAC (the grid) will be used to distribute compute necessary for KM3NeT
- Currently there is a workflow to make data available on the grid via rucio, **but not necessarily on the compute site!**
- **Benefit:** Easy to implement
- **Drawback:** Extra file transfers between compute sites. Extra energy and time spent.
- **Solution:** Make DIRAC rucio-aware. But we have no DIRAC access, so must be done by DIRAC owner (currently EGI)



# Learn more

- Documentation available: <https://rucio.pages.km3net.de/rucio-documentation/>
- Send us an email: [v.azizi@esciencecenter.nl](mailto:v.azizi@esciencecenter.nl) & [b.andela@esciencecenter.nl](mailto:b.andela@esciencecenter.nl)



# Questions?



e “Empowering  
researchers across  
all disciplines  
through innovative  
research software”

netherlands  
eScience center

## Contact Persons



Bouwe Andela  
Senior Research Software Engineer  
b.andela@esciencecenter.nl



Victor Azizi  
Research Software Engineer  
v.azizi@esciencecenter.nl