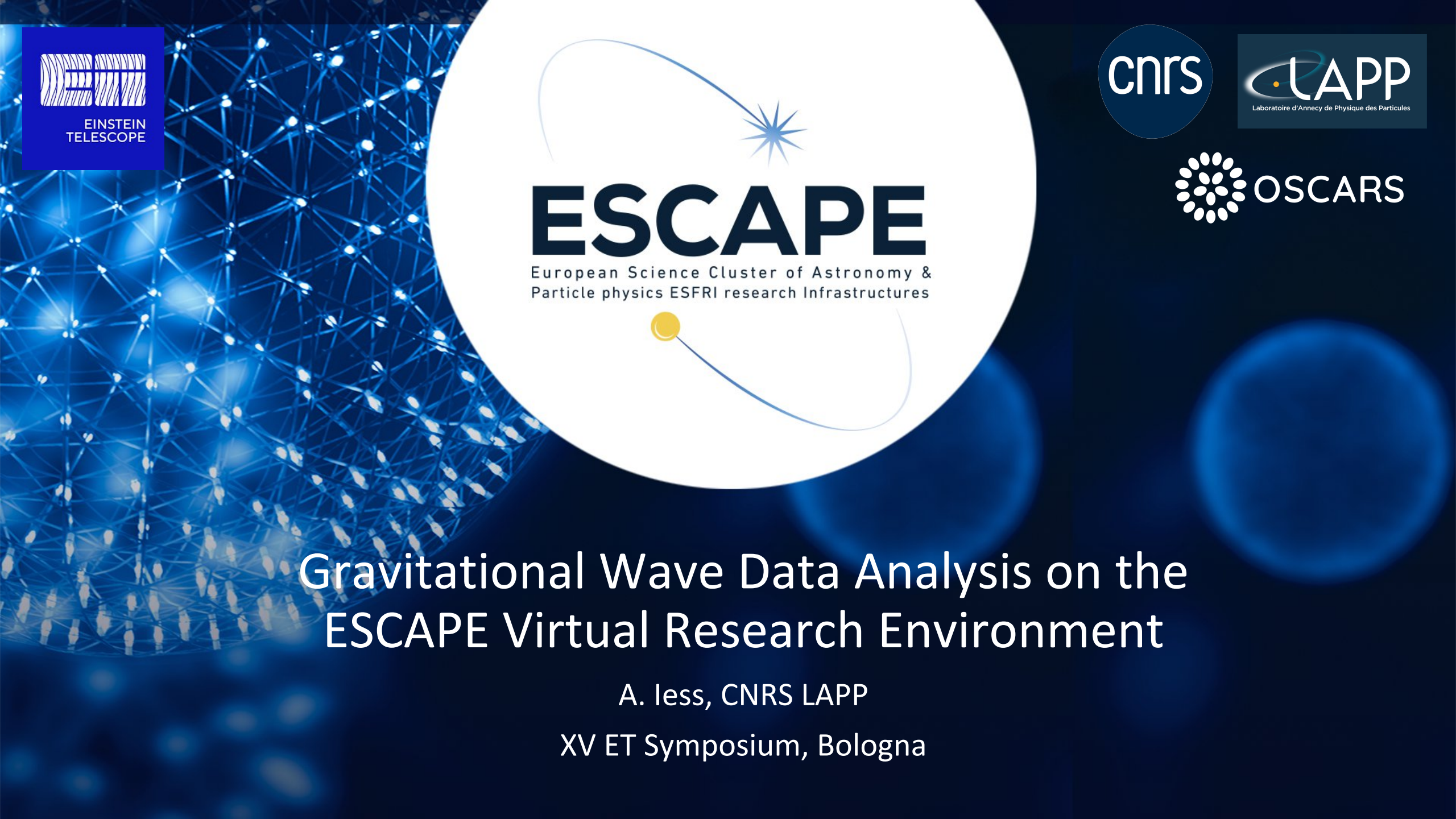


Gravitational Wave Data Analysis on the ESCAPE Virtual Research Environment

A. Iess, CNRS LAPP

XV ET Symposium, Bologna

The ESCAPE Open Collaboration

The ESCAPE Virtual Research Environment (VRE)

The **Virtual Research Environment** was developed by the VRE Team at CERN as part of the ESCAPE Project, under EU Horizon 2020 Grant Agreement no. 824064.

References

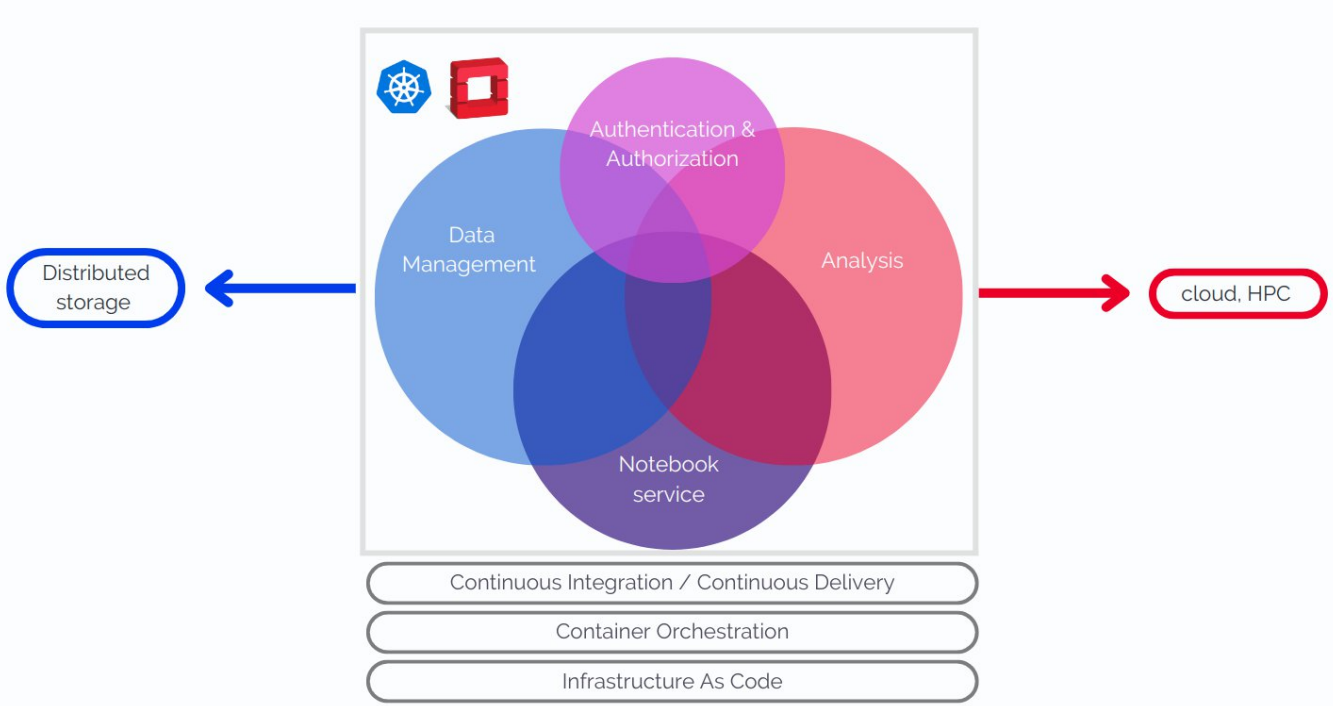
- CHEP2024 proceedings: E. Garcia-Garcia, G. Guerrieri, et al. https://arxiv.org/pdf/2503.02483.
- CHEP2024 contribution: https://indico.cern.ch/event/1338689/contributions/6010696/
- VRE Documentation: https://vre-hub.github.io
- Github Profile: https://github.com/vre-hub

CERN VRE Hub: https://jhub-vre.cern.ch

Mattermost Channel: https://mattermost.web.cern.ch/escape/channels/vre-support

VRE components

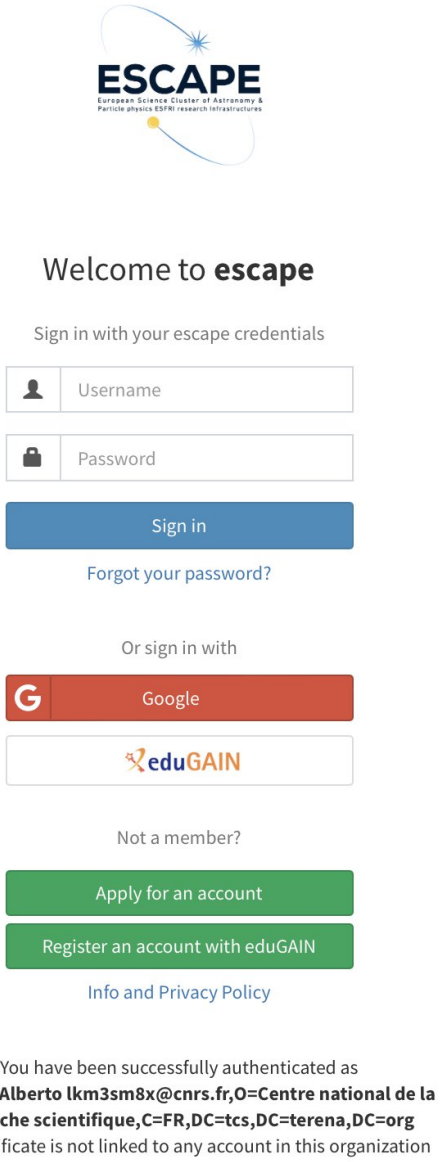
- Federated AAI
- ESCAPE Datalake for federated distributed storage .
- Computing cluster supplying the processing power to run full analyses.
- JupyterHub Interface with containerised environments.



ESCAPE AAI

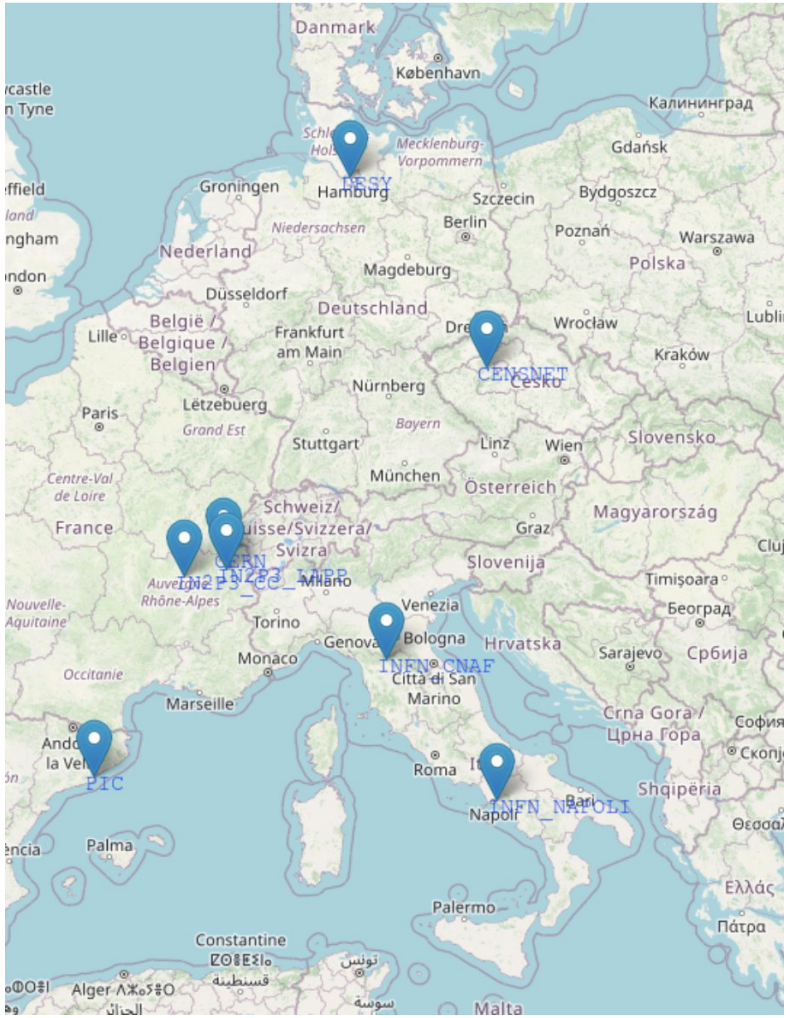
- ESCAPE AAI is based on INDIGO Identity and Access Management (IAM).
- Request an account and wait for approval.
- Add escape group.
- Support Usr+pwd, JSON Web Token OIDC, x.509 certificates.

All ESCAPE Virtual Research Environment (VRE) services and resources are federated through the ESCAPE IAM service.



Rucio for Federated Distributed Data Management

- Distributed Rucio Storage Elements (RSEs).
- Rucio distributed data management system (upload, replication rules, transfer).
- Interaction through docker container or rucio client.
- Authentication through X.509 certificates and OIDC tokens.

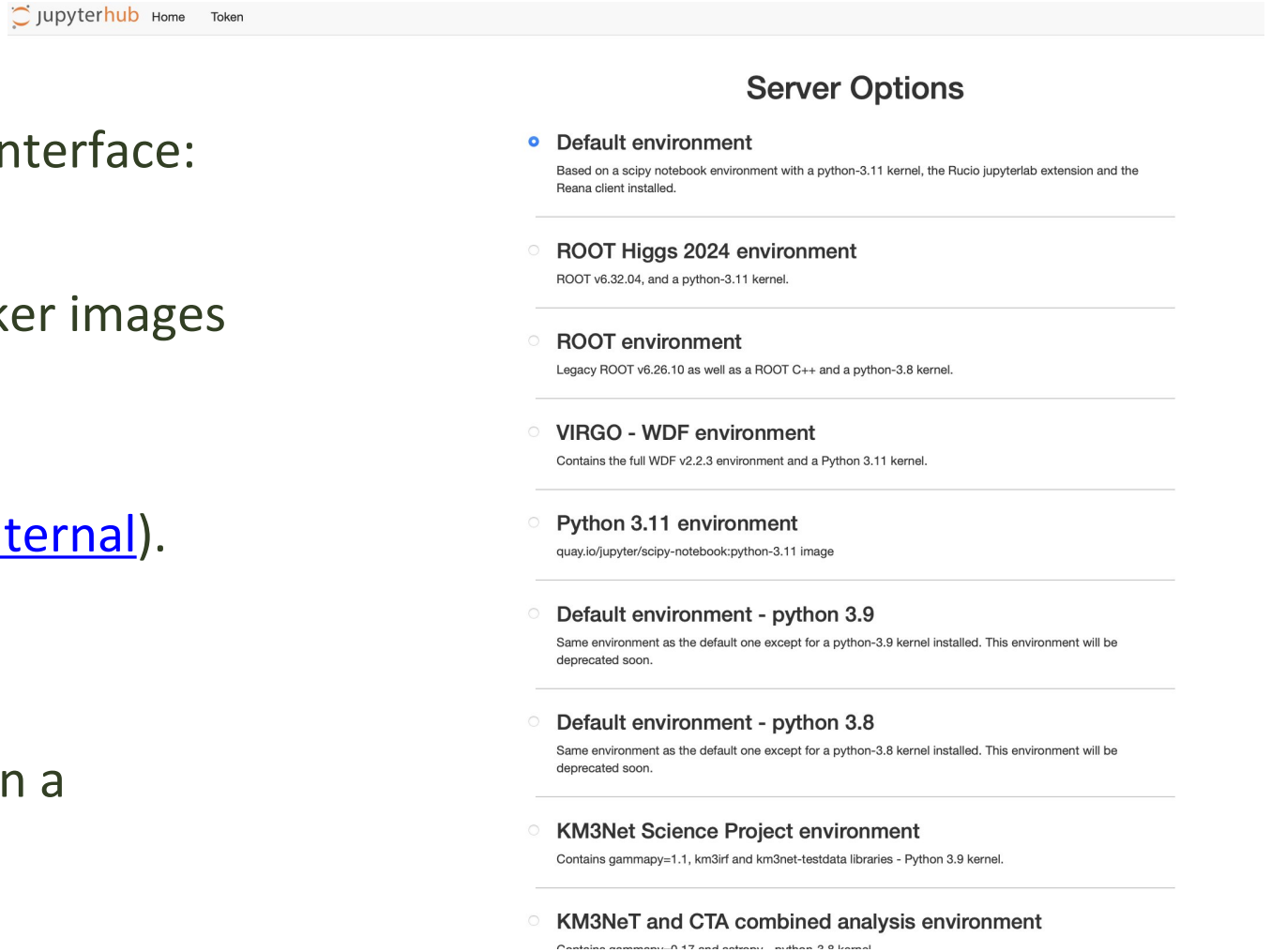


JupyterHub Interface

The ESCAPE VRE offers a JupyterHub interface:

- AAI (credentials, x.509,OpenID)
- Environments encapsulated in Docker images and run as containers.
- Rucio, REANA plugins.
- VREs at CERN and at LAPP (EOSC, internal).
- Documentation.

You can select an environment and run a notebook interactively.



Environments

Alternatively, add an environment with a pull request, which will be reviewed and eventually merged by the VRE team.

README MIT license

Docker automatic build and publish passing

Environments

VRE user environment images for workflows and notebooks available in the VRE JupyterHub service.

VRE user environments

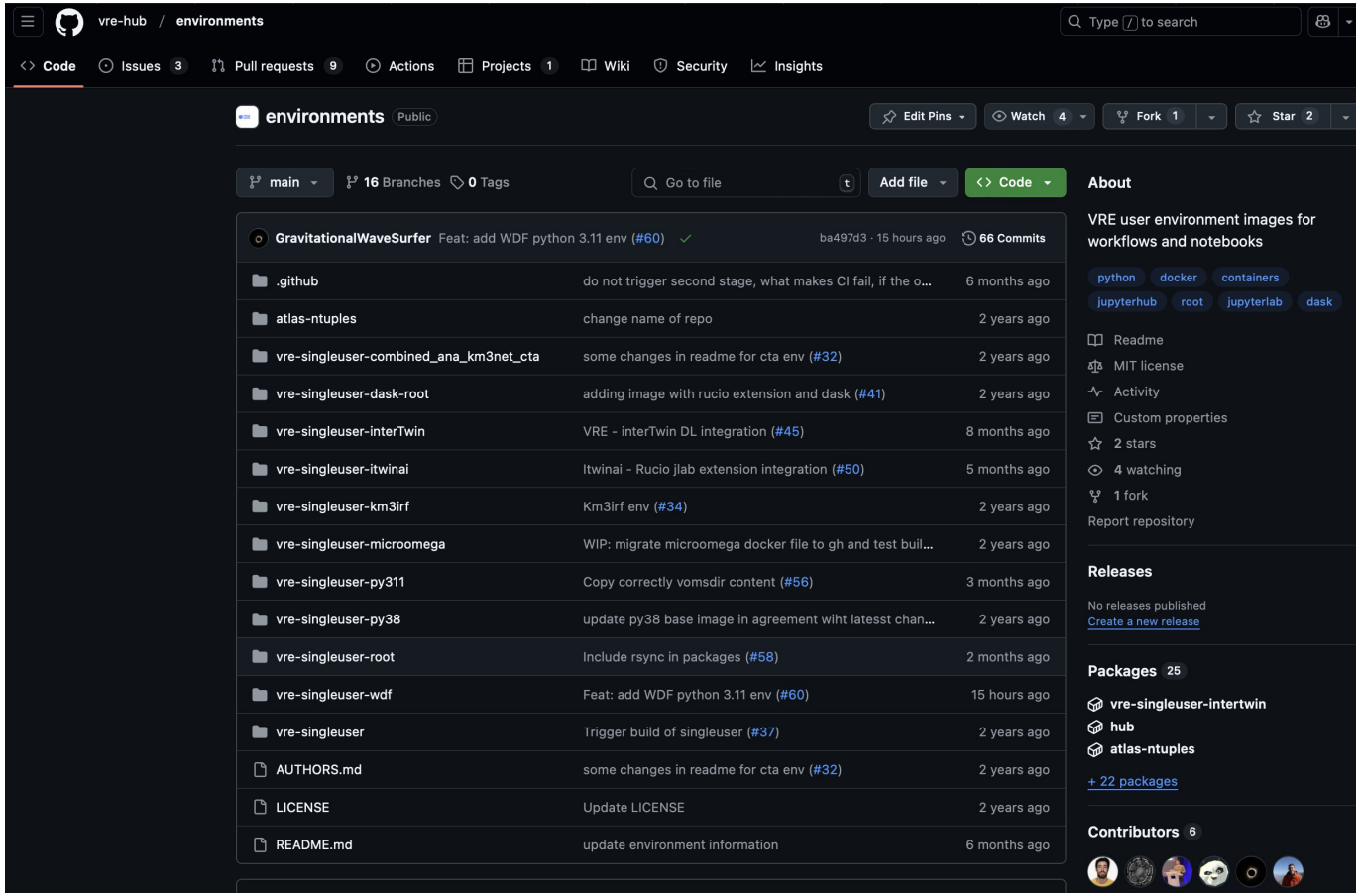
Custom user environments

If you want to extend or modify any image, find below some suggestions:

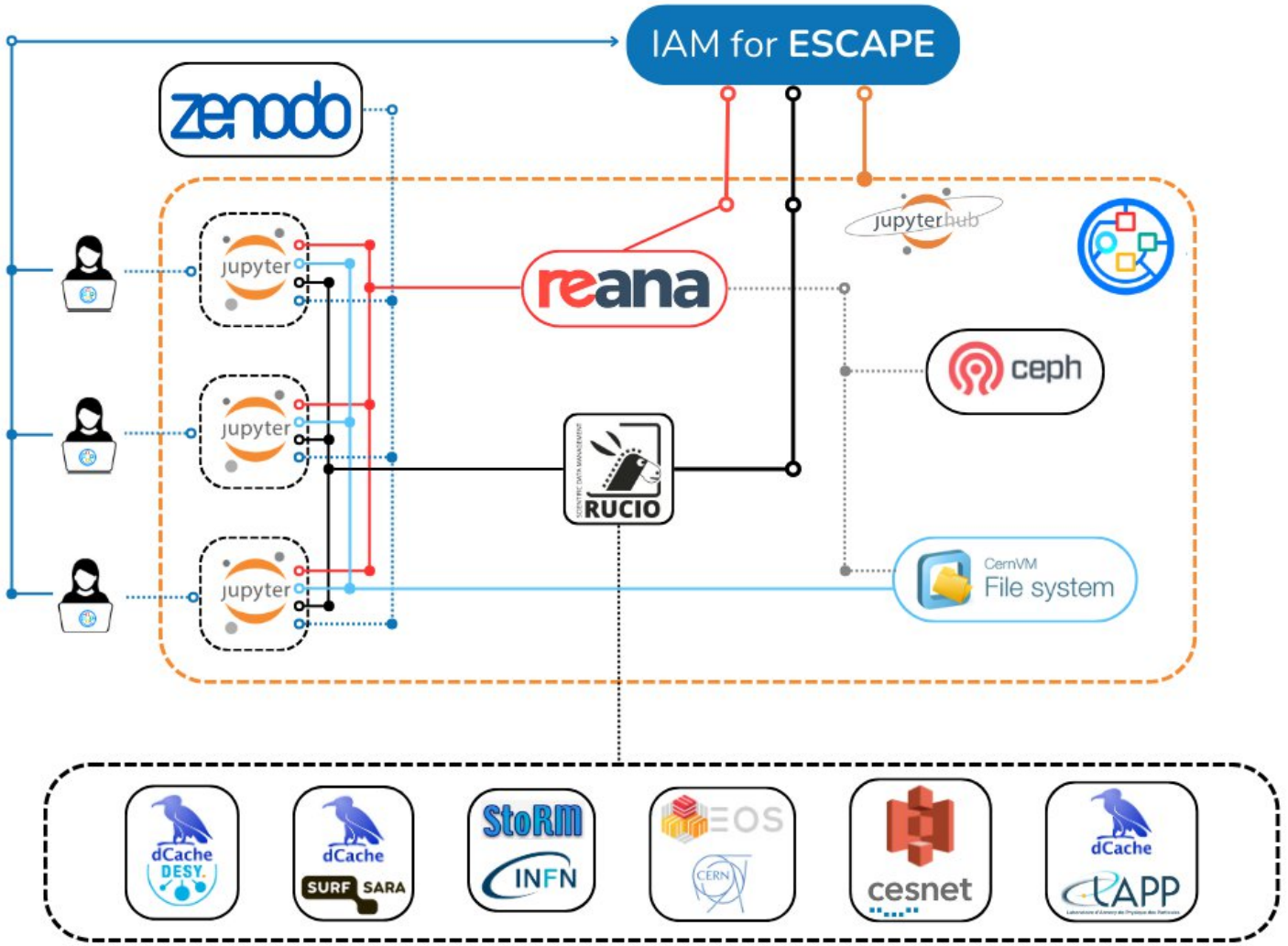
We strongly recommend that every user enviroment image uses the latests `vre-singleuser-py311` image as a base layer (the `FROM` command in the `Dockerfile` - see below).

- Check the latest version of the `vre-singleuser-py311` image here.
- To custimize an image please have a look to the Startup Docker Stack documentation Hooks.
- Create one folder per environments. The directory would need to contain a valid `Dockerfile`. Once your PR is merged, the CI will detect the changes in the repository and will trigger the building of the image.
- Add a `README.md` file describing the environment and its usage.
- We strongly recommend to add the following lines to any new user environment (thus in a `Dockerfile`), so that the image would labelled with the building date.

```
#  Beginning of the suggested Dockerfile
FROM ghcr.io/vre-hub/vre-singleuser-py311:sha-5106e39
LABEL maintainer=<your_name>
ARG BUILD_DATE
LABEL org.label-schema.build-date=$BUILD_DATE

...
```

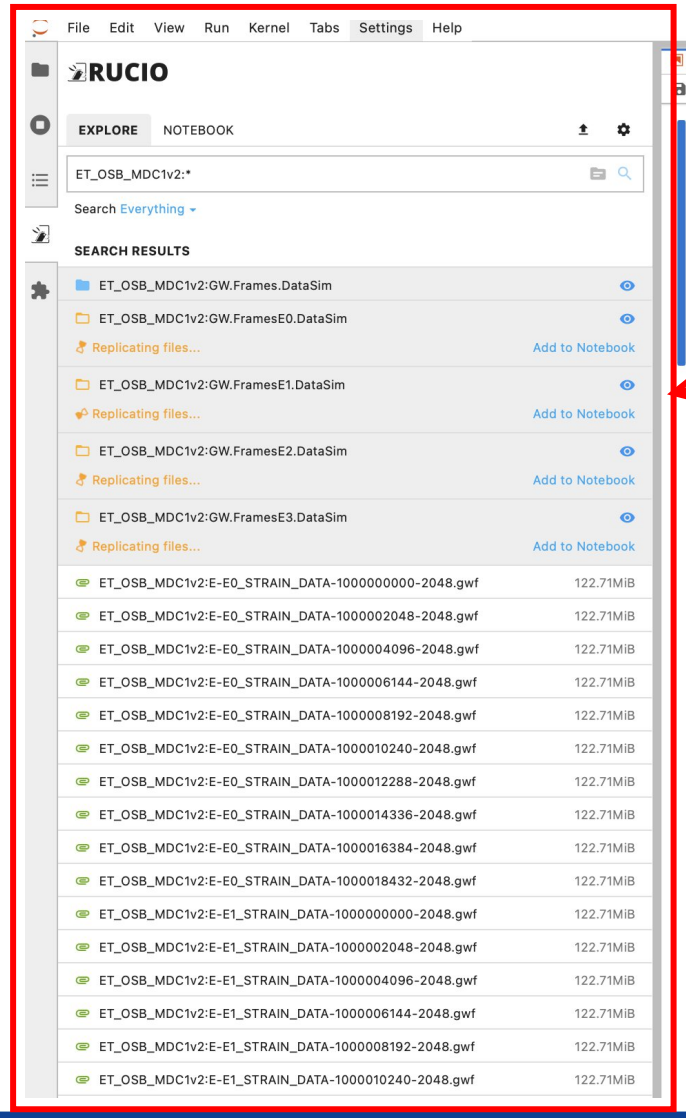


The Full Picture



Running an analysis on the ESCAPE VRE

The JupyterHub interface with Rucio extension facilitates access to data, allowing direct upload to Jupyter Notebooks!



Rucio Data Lake Extension

Untitled.ipynb | Terminal 2

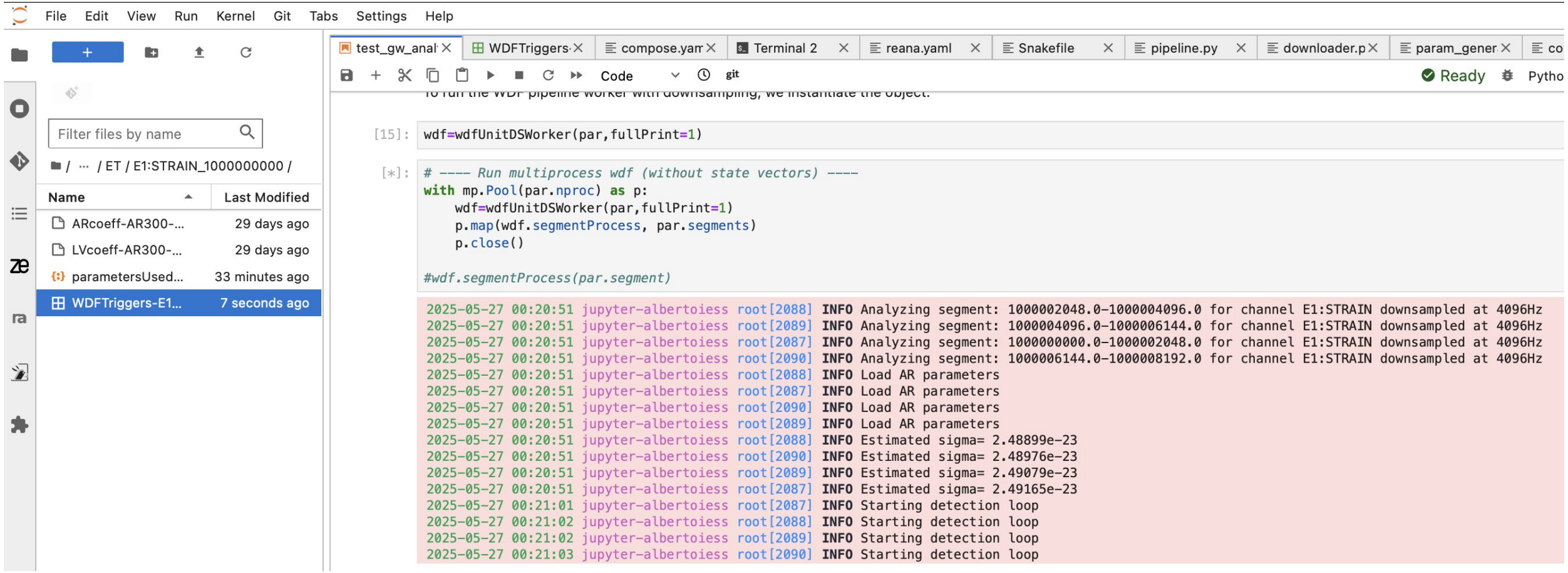
Code | Python 3 (ipykernel)

```
[2]: # import libraries
import time
import os
from pytsa.tsa import *
from pytsa.tsa import SeqView_double_t as SV
from wdf.config.Parameters import *
from wdf.processes.wdfUnitDSWorker import *
from wdf.processes.wdfUnitWorker import *
import logging
import coloredlogs
#select level of logging
coloredlogs.install(isatty=True)

logging.basicConfig(level=logging.DEBUG)
```

```
[ ]: ew_json_config_file = True    # set to True if you want to create new Configuration

if new_json_config_file==True:
    configuration = {
  "window":1024,
  "overlap":768,
  "threshold": 0.2,
  "file": "./data/test.gwf",
  "channel": "H1:GWOSC-4KHZ_R1_STRAIN",
  "run":"offLine",
  "len":10.0,
  "gps":1167559608,
  "segments":[[1167559008,1167559408],[1167559408,1167560008],[1167560008,1167560308]],
  #"segments":[[1167559608,1167560008] ],
  "outdir": "local_dir/",
  "dir":"local_dir/",
  "ID":"WDF_test",
  "ARorder": 1000,
  "learn": 200,
  "preWhite":2,
  "ResamplingFactor":2,
  "LowFrequencyCut":12,
  "FilterOrder":6,
  "nproc":4
}

    filejson = os.path.join(os.getcwd(),"inputWDF.json")
    file_json = open(filejson, "w+")
    json.dump(configuration, file_json)
    file_json.close()

logging.info("read parameters from JSON file")
par = Parameters()

filejson = "inputWDF.json"
try:
    par.load(filejson)
except IOError:
    logging.error("Cannot find resource file " + filejson)
    quit()

par.print()
```

Running an analysis on the ESCAPE VRE

Example: notebook running multiprocess Wavelet Detection Filter pipeline for burst signal detection (Cuoco et al. 2018, Cuoco et al. 2001)



Running an analysis on the ESCAPE VRE

The .csv event trigger file generated by running Wavelet Detection Filter pipeline on the ESCAPE VRE

test_gw_anal | WDFTriggers | compose.yam | Terminal 2 | reana.yaml | Snakefile | pipeline.py | downloader.p | param_gener | config.yaml

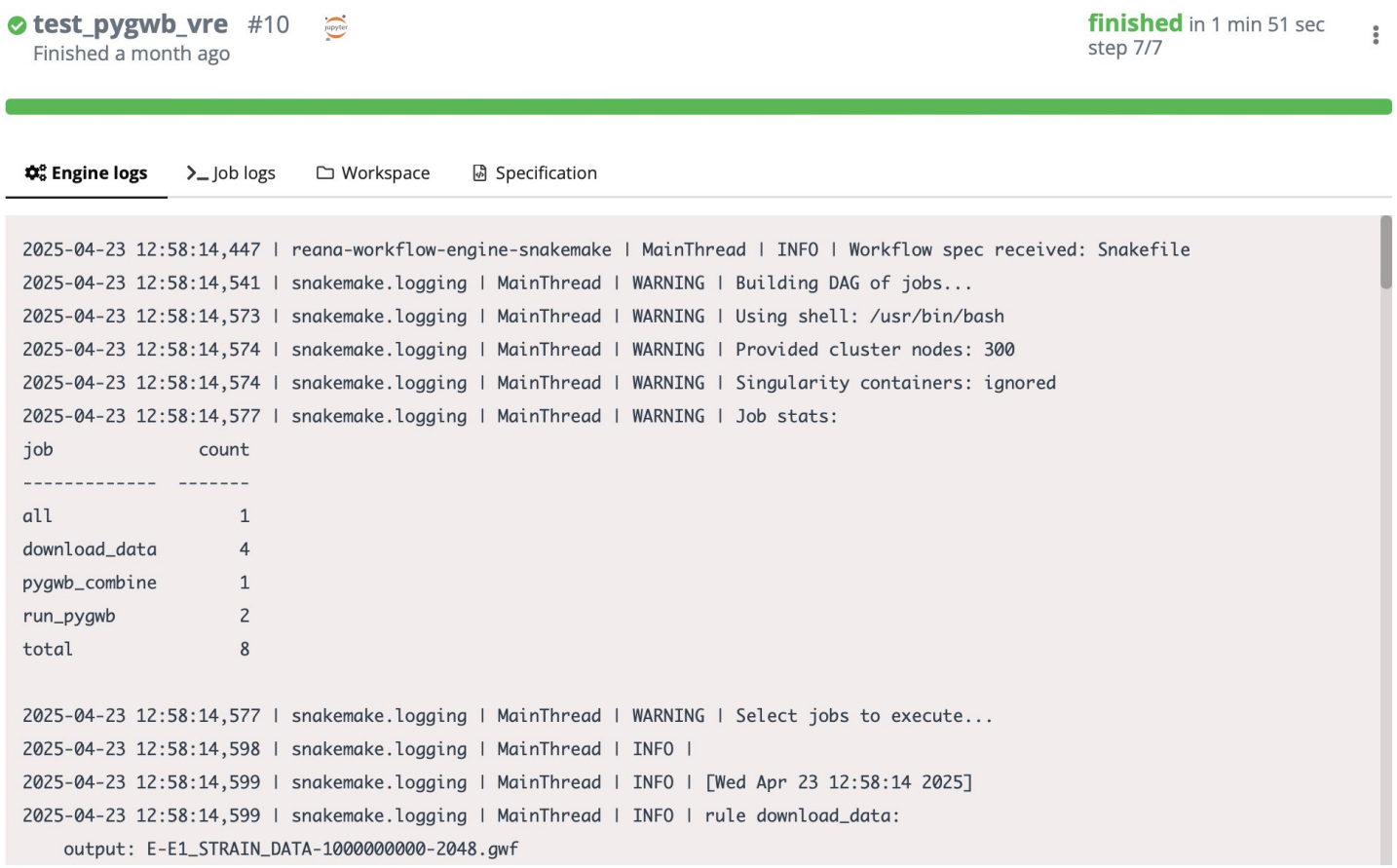
Delimiter: ,

	gps	gpsPeak	duration	EnWDF	snrMean	snrPeak	freqMin	freqMean	freqMa
1	1000000002.1875	1000000002.3349609	0.249755859375	0.5090096328597434	0.3304358309852966	1.5626154136668986	64.0	265.15384615384613	492
2	1000000002.5625	1000000002.590332	0.249755859375	0.5150112307452843	0.32948690698212135	1.3899681034144495	88.0	302.53846153846155	532
3	1000000005.5625	1000000005.7119141	0.242919921875	0.5260332433303789	0.38063813106407723	1.5293947286777794	56.0	251.69230769230768	488
4	1000000005.6875	1000000005.7119141	0.249755859375	0.5084603101105939	0.3091762745273994	1.623545971548338	72.0	248.30769230769232	488
5	1000000009.0625	1000000009.2770996	0.249755859375	0.5321581103463993	0.348202901526786	1.735695712904407	60.0	251.46153846153845	548
6	1000000009.125	1000000009.2770996	0.248046875	0.515084243562685	0.32840622654898083	1.7356709540608741	76.0	260.15384615384613	548
7	1000000012.375	1000000012.6000977	0.249755859375	0.5812086311386826	0.32150536887667847	1.855986213599196	72.0	270.6923076923077	560
8	1000000012.4375	1000000012.6000977	0.24560546875	0.5355878675796901	0.3224722419252462	1.8164082288500996	56.0	258.9230769230769	556
9	1000000012.5	1000000012.6000977	0.219482421875	0.5283963368159178	0.31108966417654255	1.8323184161283326	56.0	278.46153846153845	556
10	1000000014.8125	1000000015.032959	0.235595703125	0.6082811202060369	0.3924666378908149	2.2688403490803326	60.0	256.2307692307692	500
11	1000000014.875	1000000015.032959	0.24169921875	0.6437043479061679	0.4213356644600656	2.2444956126386786	68.0	289.3076923076923	524
12	1000000014.9375	1000000015.032959	0.241943359375	0.6161986213791932	0.4098125173637369	2.227456621365669	64.0	282.53846153846155	500
13	1000000015.0	1000000015.032959	0.249755859375	0.5287611498376082	0.3541891159173277	2.2030764128831555	76.0	286.0769230769231	548
14	1000000015.25	1000000015.3498535	0.249755859375	0.5485104527401753	0.34145573075477914	1.6717384709040886	100.0	252.3846153846154	452
15	1000000019.6875	1000000019.7192383	0.249755859375	0.5225201461512395	0.3410631793993525	1.8502531300496663	68.0	269.0769230769231	504
16	1000000019.75	1000000019.8903809	0.243896484375	0.5007517928950685	0.3286354440597249	1.540387518531809	56.0	228.69230769230768	496
17	1000000021.0625	1000000021.2419434	0.24072265625	0.5262832802322703	0.3412362372352981	2.013707989954107	28.0	251.84615384615384	544
18	1000000021.4375	1000000021.6296387	0.249755859375	0.5581610663642006	0.32336850379768367	1.3170298836277476	60.0	219.46153846153845	428
19	1000000021.5	1000000021.6813965	0.248779296875	0.598113337899521	0.3634672927054459	1.5205310713493543	48.0	209.84615384615384	436
20	1000000021.5625	1000000021.6813965	0.238037109375	0.5097732791391584	0.3210342210444631	1.4873077394455962	52.0	212.0	392

Running an analysis on the ESCAPE VRE

For complex reproducible analyses workflows, you can dispatch to REANA cluster.

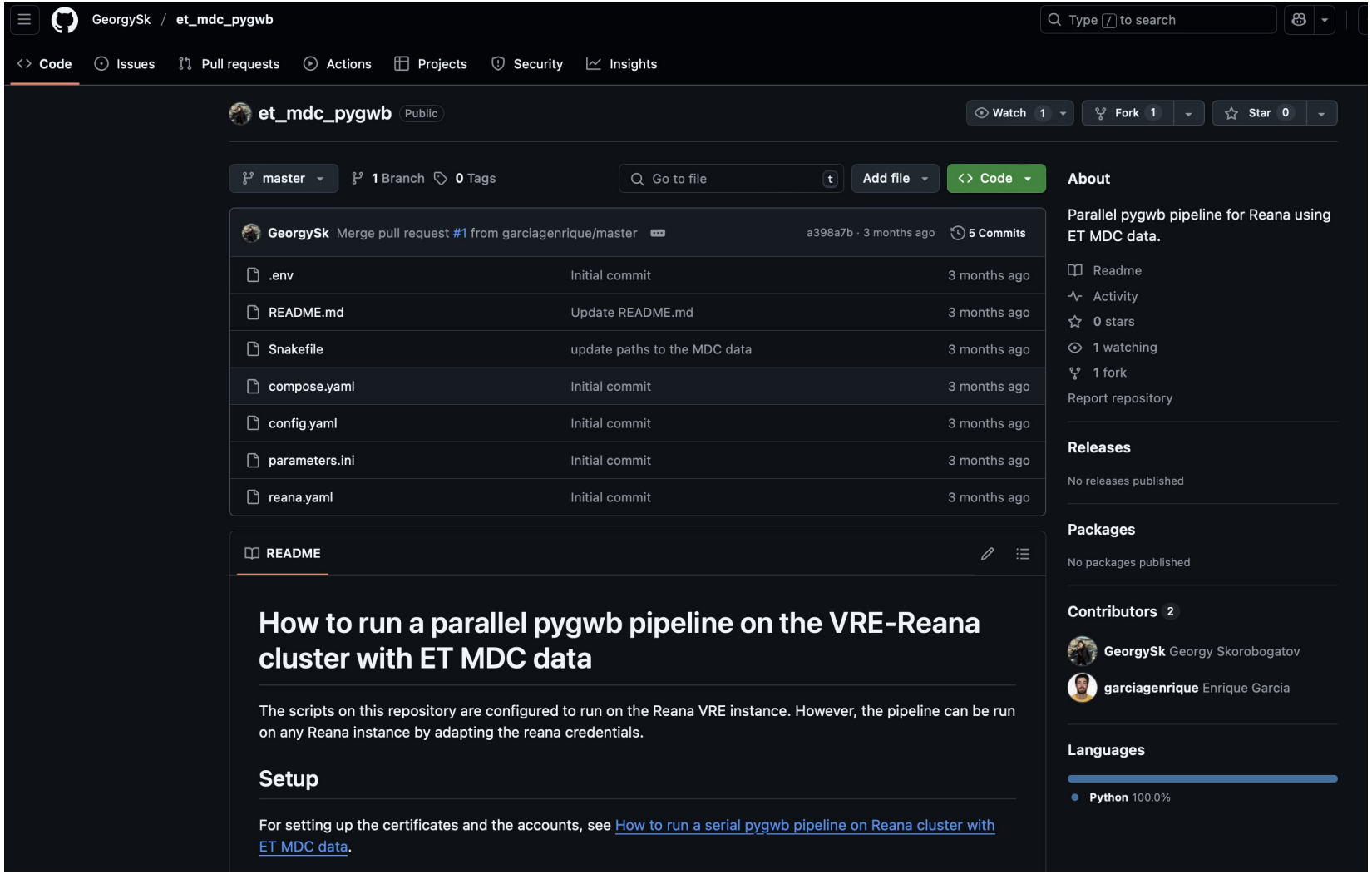
- The REANA JupyterLab extension allow users to interact with the workflow management system from within JupyterLab.
- Reana - Rucio integration to directly upload files from a Rucio RSE to the Reana workspace. No need to download files locally from Rucio and then upload them to the Reana workspace.



Running an analysis on the ESCAPE VRE

Nice example: https://github.com/GeorgySk/et_mdc_pygwb/tree/master

More to come...



Conclusions

- The ESCAPE VRE facilitates the development of end-to-end physics workflows, providing researchers with access to an infrastructure and to the digital content necessary to produce and preserve a scientific result in compliance with FAIR principles.
- Lower the entry barrier for young researchers through JupyterHub and straightforward data access, while allowing experienced researchers to run complex workflows through REANA.
- Open-source and modular, in order to make it easily reproducible by partner institutions.
- Experienced team, willing to help setting up and customizing for diverse scientific communities.
- More in upcoming talks on MADDEN and ETAP OSCARS funded projects.

EXTRA SLIDE: REANA

Reproducible analysis platform for containerised data analysis pipelines on remote compute clouds.

- Supported workflow systems: CWLSerial, Snakemake, Yadage
- Supported compute backends: HTCondor, Kubernetes, Slurm
- Supported source code and storage systems: GitLab, CVMFS, EOS