

Module 6.2

How to run a test using a workflow manager

Workflow manager

Workflow manager

Manages experiments across different platforms

Automatic handling of job submission, dependencies, and error recovery.

Improves workflow scalability and traceability.

EDITO entrypoint to HPC clusters



Think workflow!

When solving big computational problems, it is common to think of your top-level **main function** as a **workflow** that is run through a **workflow manager**.

Think workflow!

When solving big computational problems, it is common to think of your top-level **main function** as a **workflow** that is run through a **workflow manager**.

```
def fetch_inputs():
    ...
def run_simulation():
    ...
def transfer_data():
    ...
def analyze_data():
    ...
def publish_paper():
    ...
def main():
    fetch_inputs()
    run_simulation()
    transfer_data()
    analyze_data()
    publish_paper()

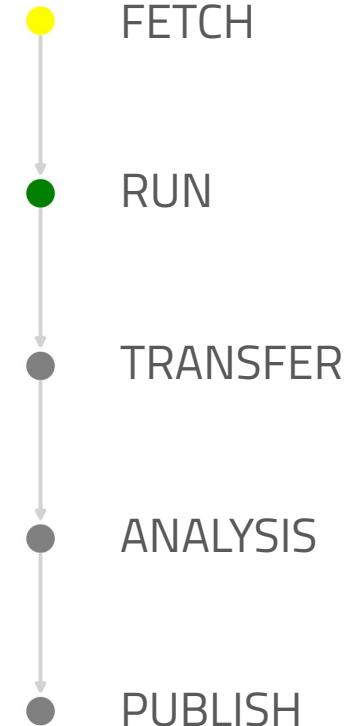
if __name__ == "__main__":
    main()
```

Think workflow!

When solving big computational problems, it is common to think of your top-level **main function** as a **workflow** that is run through a **workflow manager**.

```
def fetch_inputs():
    ...
def run_simulation():
    ...
def transfer_data():
    ...
def analyze_data():
    ...
def publish_paper():
    ...
def main():
    fetch_inputs()
    run_simulation()
    transfer_data()
    analyze_data()
    publish_paper()

if __name__ == "__main__":
    main()
```



EDITO Ecosystem

-  [Reduce](#)
-  [Home](#)
-  [My account](#)
-  [Project settings](#)
-  [Service catalog](#)
-  [My Services](#)
-  [Process catalog](#)
-  [My processes](#)
-  [My Secrets](#)
-  [My Files](#)
-  [Data Explorer](#)



Welcome Ivan to the Datalab

More information about the platform is available in the [Trainings and tutorials](#) section.

[Trainings and tutorials](#)

Run your application as a service

Tune and deploy your application to provide it as a service to your community. Analyze data interactively, benefit from a growing catalog of services, create your own tools, What-If applications and focus applications. Work with the languages and environments you prefer and reserve the resources you need.

[Browse the service catalog](#)

Launch on-demand processes

Execute and configure remote functions to compute scientific processes, such as data transformation, pre/post-processing, reanalysis, forecasts, detections, What-If scenarios, quality controls. Configure runtime inputs to generate data that have never been generated before within the data storage of your choice.

[Browse the process catalog](#)

[Reduce](#)[Home](#)[My account](#)[Project settings](#)[Service catalog](#)[My Services](#)[Process catalog](#)[My processes](#)[My Secrets](#)[My Files](#)[Data Explorer](#)

Service catalog

All catalogs

Search

All

Ocean modelling

Ocean data quality

Data visualization

What-if applications

Focus applications

IDE

Databases

Automation

Playground

**Jupyterlab-autosubmit-bsc***Ocean modelling*

Launch a jupyter python 3.10.12 with Autosubmit installed

[Learn more](#)[Launch](#)**Autosubmit-computing-node-ssh***Ocean modelling*

An autosubmit computing node service (openssh-server) to run alongside Jupyterlab-autosubmit-bsc

[Launch](#)**Delft3dfm-modelbuilder***Ocean modelling*

Set up a Delft3D FM model from scratch with the dfm_tools modelbuilder via the JupyterLab IDE and Python 3.11/3.12.

[Launch](#)**Jupyter-python-ocean-science***Ocean modelling*

A JupyterLab service with Python 3.12 and a collection of standard ocean science packages (nco, cdo, python-cdo, copernicusmarine, pybstac, OWSLib, ...).

[Launch](#)**Autosubmit-demo***Ocean modelling*

Runs a demo of Autosubmit (with JupyterLab), API, and GUI

[Launch](#)**Turbiditymapping-4dvarnet***Ocean modelling*

Launch a Turbidity Mapping - 4DVarNet JupyterLab

[Launch](#)**Surf-nemo***Ocean modelling*

Structured and Unstructured-grid Relocatable ocean platform for Forecasting (SURF), running the Nemo model.

[Launch](#)**Nemo-demo***Ocean modelling*

A Helm chart that runs NEMO and serves its output data through WMS

[Launch](#)

- [Reduce](#)
- [Home](#)
- [My account](#)
- [Project settings](#)
- [Service catalog](#)
- [My Services](#)
- [Process catalog](#)
- [My processes](#)
- [My Secrets](#)
- [My Files](#)
- [Data Explorer](#)

Service catalog

All catalogs

Search

All

Ocean modelling Ocean data quality

Data visualization

What-if applications

Focus applications

IDE

Databases

Automation

Playground



Jupyterlab-autosubmit-bsc

Ocean modelling

Launch a jupyter python 3.10.12 with Autosubmit installed

[Learn more](#) [Launch](#)



Autosubmit-computing-node-ssh

Ocean modelling

An autosubmit computing node service (openssh-server) to run alongside Jupyterlab-autosubmit-bsc

[Launch](#)



Delft3dfm-modelbuilder

Ocean modelling

Set up a Delft3D FM model from scratch with the dfm_tools modelbuilder via the JupyterLab IDE and Python 3.11/3.12.

[Launch](#)



Jupyter-python-ocean-science

Ocean modelling

A JupyterLab service with Python 3.12 and a collection of standard ocean science packages (nco, cdo, python-cdo, copernicusmarine, pystac, OWSLib, ...).

[Launch](#)



Autosubmit-demo

Ocean modelling

Runs a demo of Autosubmit (with JupyterLab), API, and GUI

[Launch](#)



Turbiditymapping-4dvarnet

Ocean modelling

Launch a Turbidity Mapping - 4DVarNet JupyterLab

[Launch](#)



Surf-nemo

Ocean modelling

Structured and Unstructured-grid Relocatable ocean platform for Forecasting (SURF), running the Nemo model.

[Launch](#)



Nemo-demo

Ocean modelling

A Helm chart that runs NEMO and serves its output data through WMS

[Launch](#)

- [Reduce](#)
- [Home](#)
- [My account](#)
- [Project settings](#)
- [Service catalog](#)
- [My Services](#)
- [Process catalog](#)
- [My processes](#)
- [My Secrets](#)
- [My Files](#)
- [Data Explorer](#)

My Services

Access your running services

Services are supposed to be shut down as soon as you stop using them actively.

Refresh + New service Delete all Events 1

Running services



Autosubmit-demo



Service

Autosubmit-demo

Started:

20 minutes ago



Open

\$ helm get notes autosubmit-demo-364932 --namespace user-ialsina



Resource usage quotas

Show more (5)

✓ Your current resource usage is reasonable.

Your Autosubmit Demo App is being deployed.

This service have the following tools:

- [Jupyter Lab](#)
 - Your access token is zcqnoo7bcb415bvn6vi9
- [Autosubmit GUI](#)

Return

Click to copy the password...

- [Reduce](#)
- [Home](#)
- [My account](#)
- [Project settings](#)
- [Service catalog](#)
- [My Services](#)
- [Process catalog](#)
- [My processes](#)
- [My Secrets](#)
- [My Files](#)
- [Data Explorer](#)

My Services

Access your running services

Services are supposed to be shut down as soon as you stop using them actively.

Refresh + New service Delete all Events 1

Running services



Autosubmit-demo

Service

Autosubmit-demo

Started:

20 minutes ago

Open

Your Autosubmit Demo App is being deployed.

This service have the following tools:

- Jupyter Lab
 - Your access token is zcqnoo7bcb415bvn6vi9
- Autosubmit GUI

[Return](#) [Click to copy the password...](#)

```
$ helm get notes autosubmit-demo-364932 --namespace user-ialsina
```

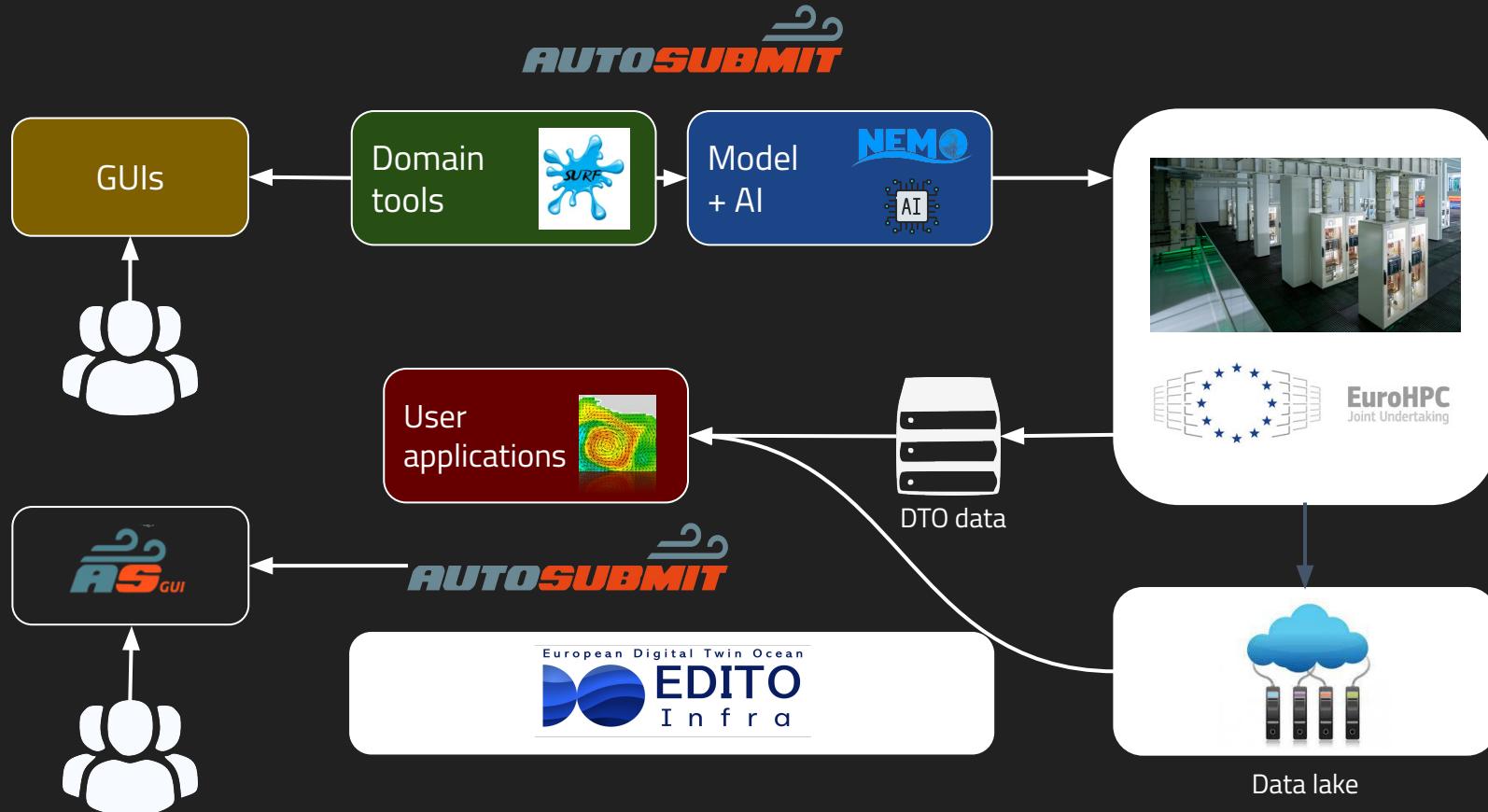
Resource usage quotas

Show more (5)

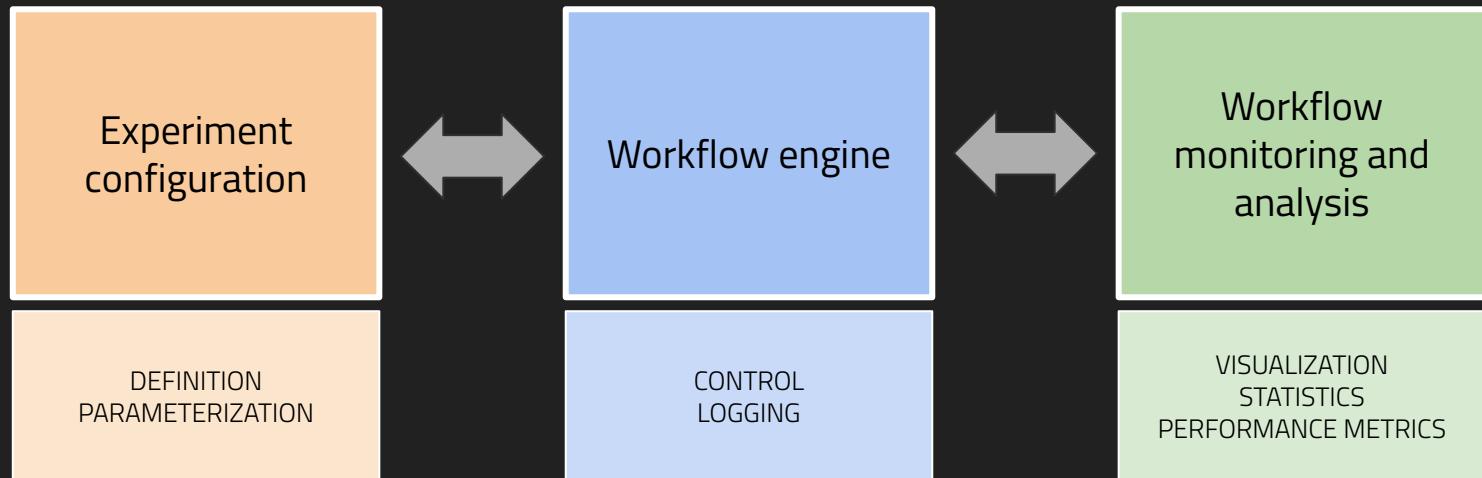
✓ Your current resource usage is reasonable.

Jupyter Lab: service with a notebook with Autosubmit instance running, and access to HPC

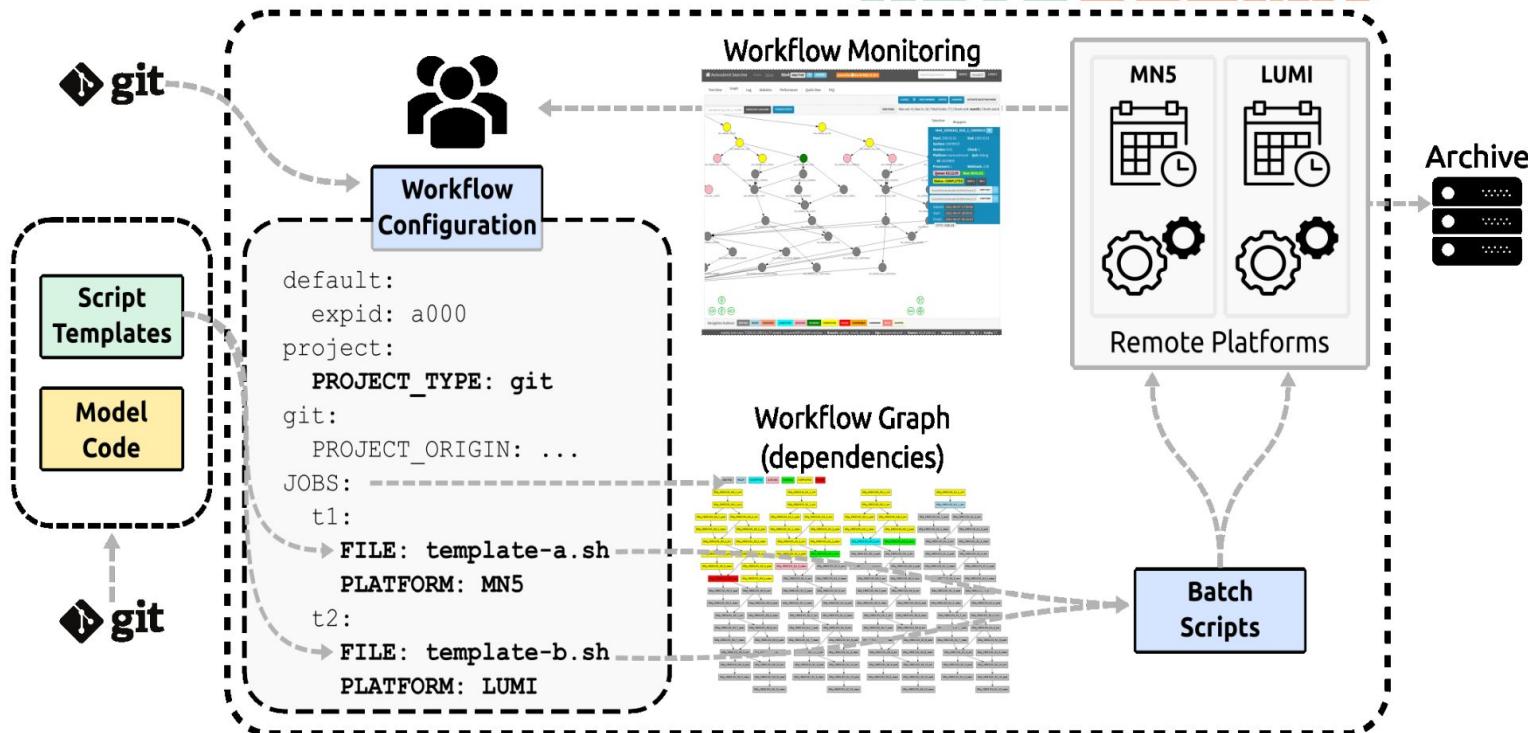
Autosubmit GUI: web app to monitor Autosubmit experiments

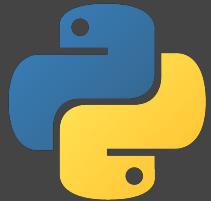


Autosubmit



AUTOSUBMIT





Coupled Model
Intercomparison
Project



Destination Earth





Declarative

YAML

Validation

Inheritance

Imports



Declarative

Consistent

YAML

Validation

Inheritance

Imports

Start Dates

Members

Chunks

Job Splits

Variables



Declarative

YAML
Validation
Inheritance
Imports

Consistent

Start Dates
Members
Chunks
Job Splits
Variables

Composable

Standardised IDs
Create – Copy
Migrate – Monitor
Workflow ↳ model code
RO-Crate

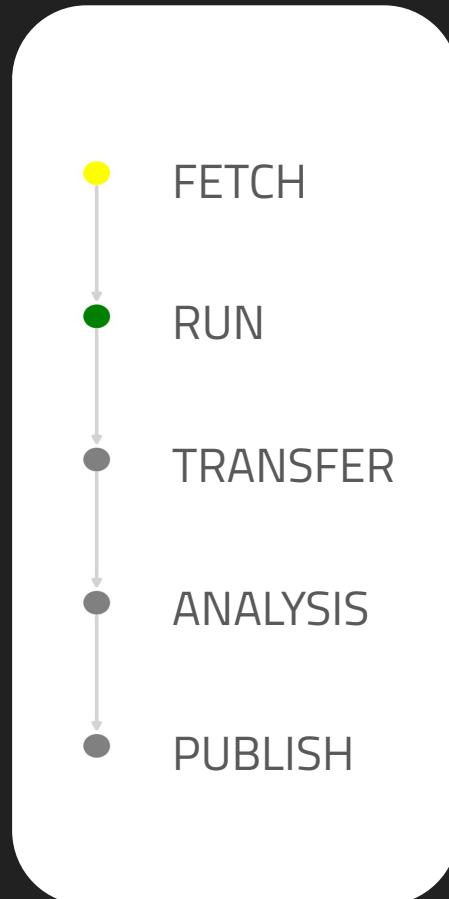
Configuration

Workflow Dependencies

jobs.yml

```
JOBS:  
  FETCH:  
    FILE: Fetch.sh  
  RUN:  
    FILE: Run.sh  
    DEPENDENCIES: FETCH  
  TRANSFER:  
    FILE: Transfer.sh  
    DEPENDENCIES: RUN  
  ANALYSIS:  
    FILE: Chunk.py  
    DEPENDENCIES: TRANSFER  
  PUBLISH:  
    FILE: Publish.sh  
    DEPENDENCIES: ANALYSIS
```

YAML

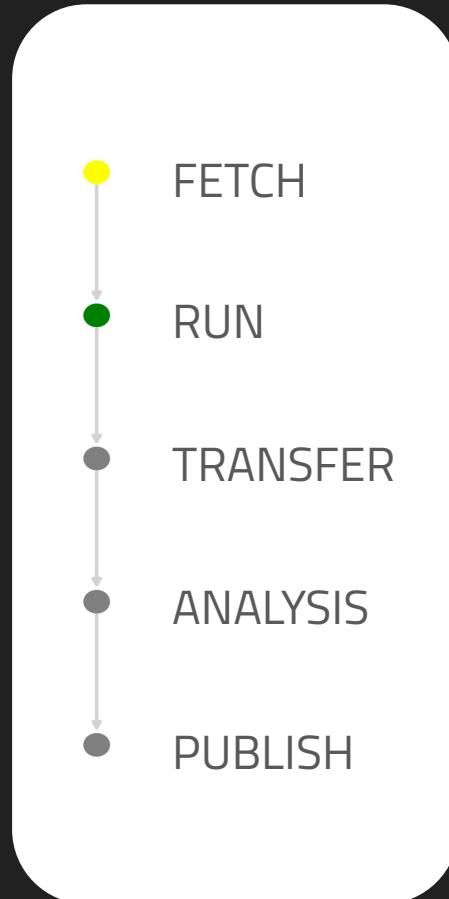


Workflow Dependencies

jobs.yml

```
JOBS:  
  FETCH:  
    FILE: Fetch.sh  
    PLATFORM: LOCAL  
  RUN:  
    FILE: Run.sh  
    DEPENDENCIES: FETCH  
    PLATFORM: REMOTE  
  TRANSFER:  
    FILE: Transfer.sh  
    DEPENDENCIES: RUN  
    PLATFORM: REMOTE  
  ANALYSIS:  
    FILE: Chunk.py  
    DEPENDENCIES: TRANSFER  
    PLATFORM: LOCAL  
  PUBLISH:  
    FILE: Publish.sh  
    DEPENDENCIES: ANALYSIS  
    PLATFORM: LOCAL
```

YAML

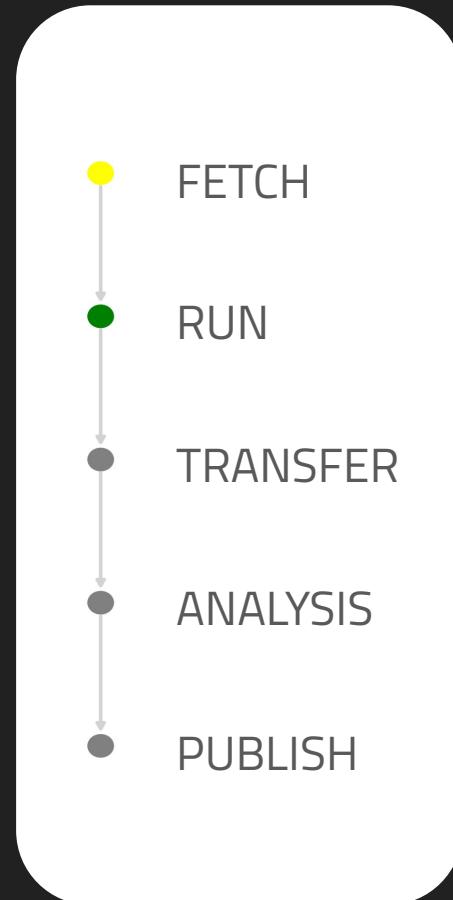


Workflow Platforms

~/platforms.yml

```
PLATFORMS:  
  REMOTE:  
    USER: my-remote-user
```

YAML



Workflow Platforms

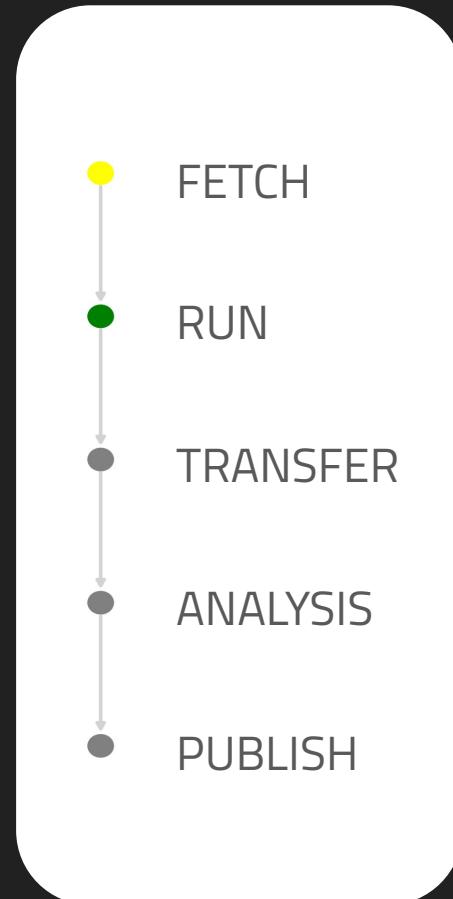
~/platforms.yml

```
PLATFORMS:  
  REMOTE:  
    USER: my-remote-user
```

YAML

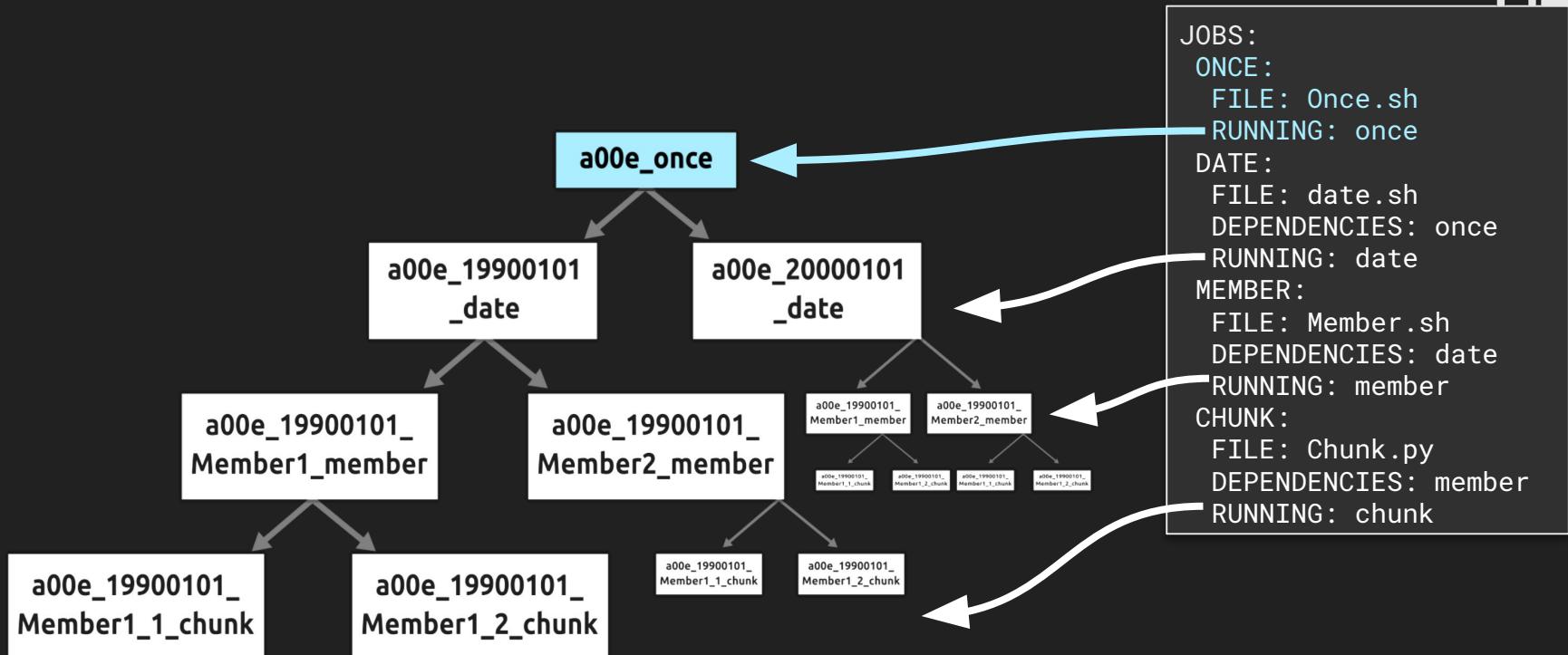
~/ .ssh/config

```
PLATFORMS:  
Host mn5login  
  HostName glogin1.bsc.es  
  User my-remote-user  
  IdentityFile ~/.ssh/id_rsa  
  ForwardX11 yes
```



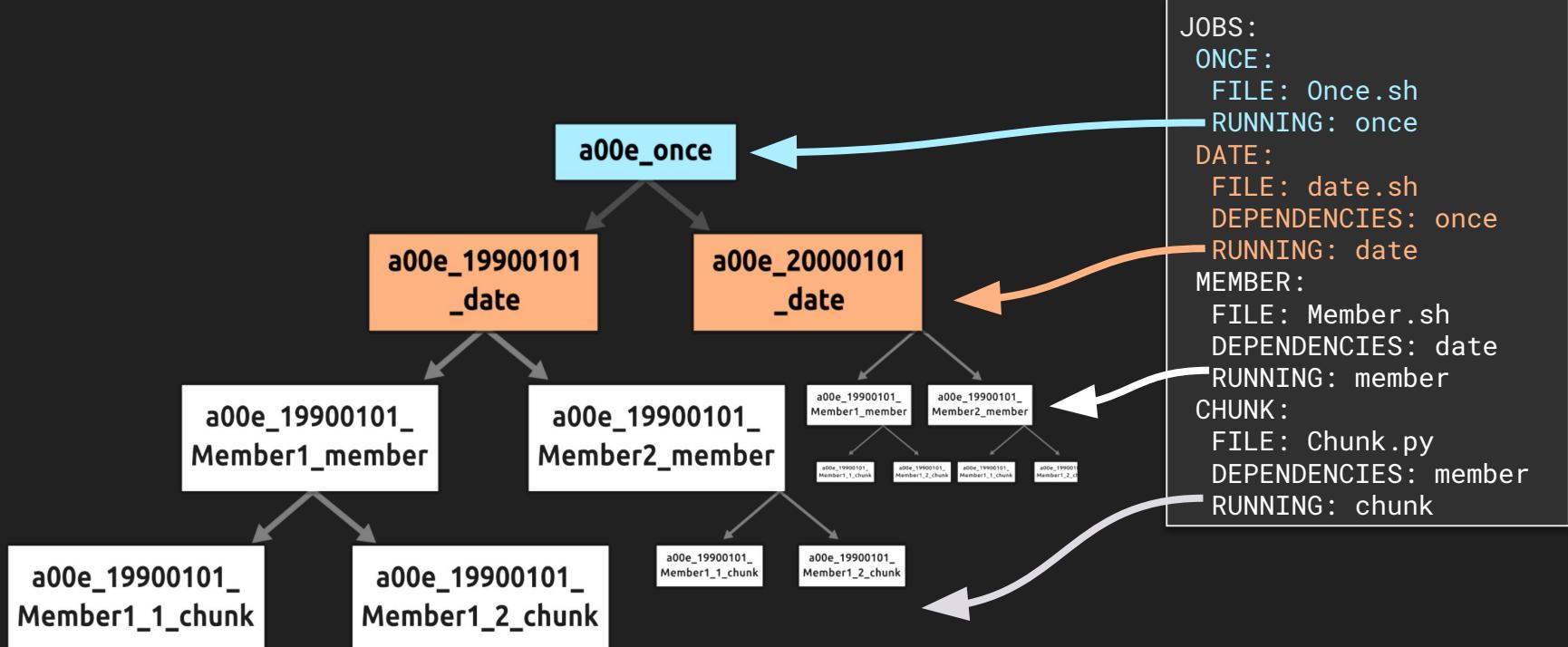
Workflow Hierarchy

YAML



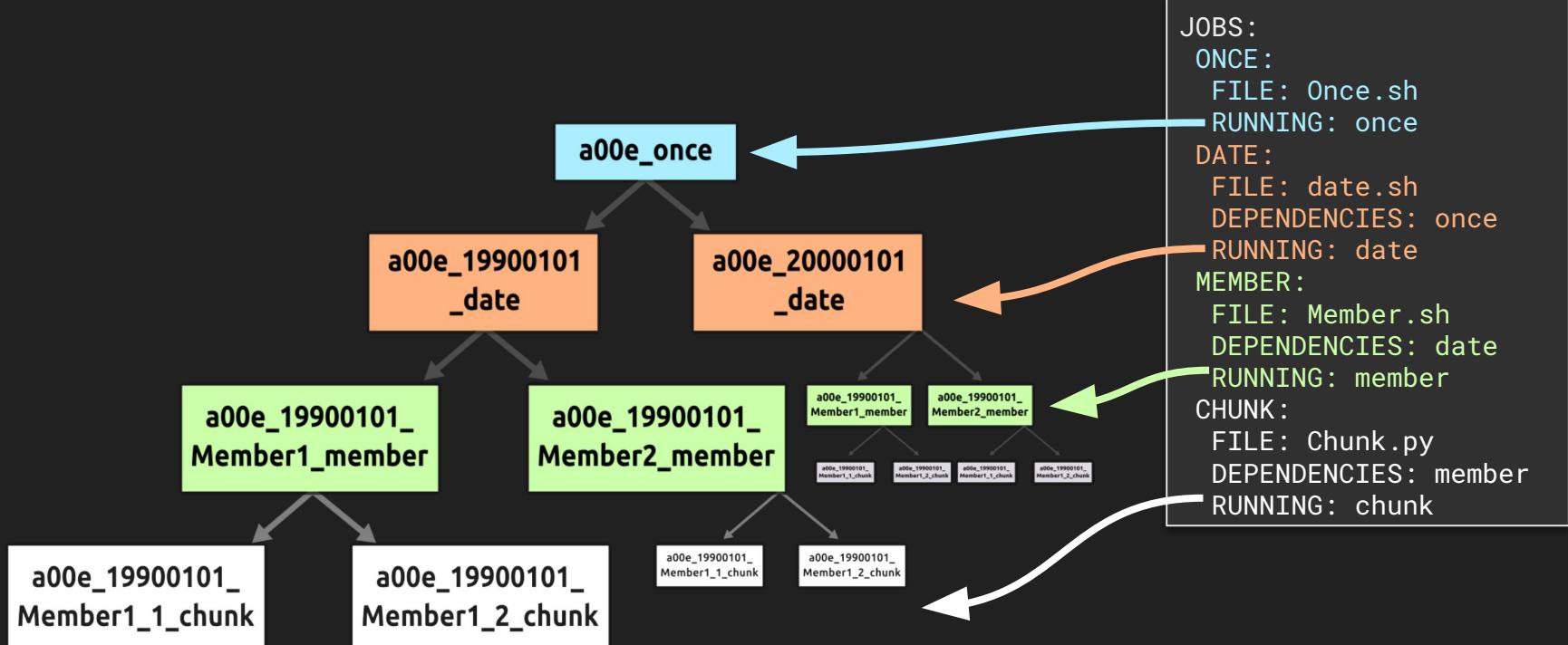
Workflow Hierarchy

YAML



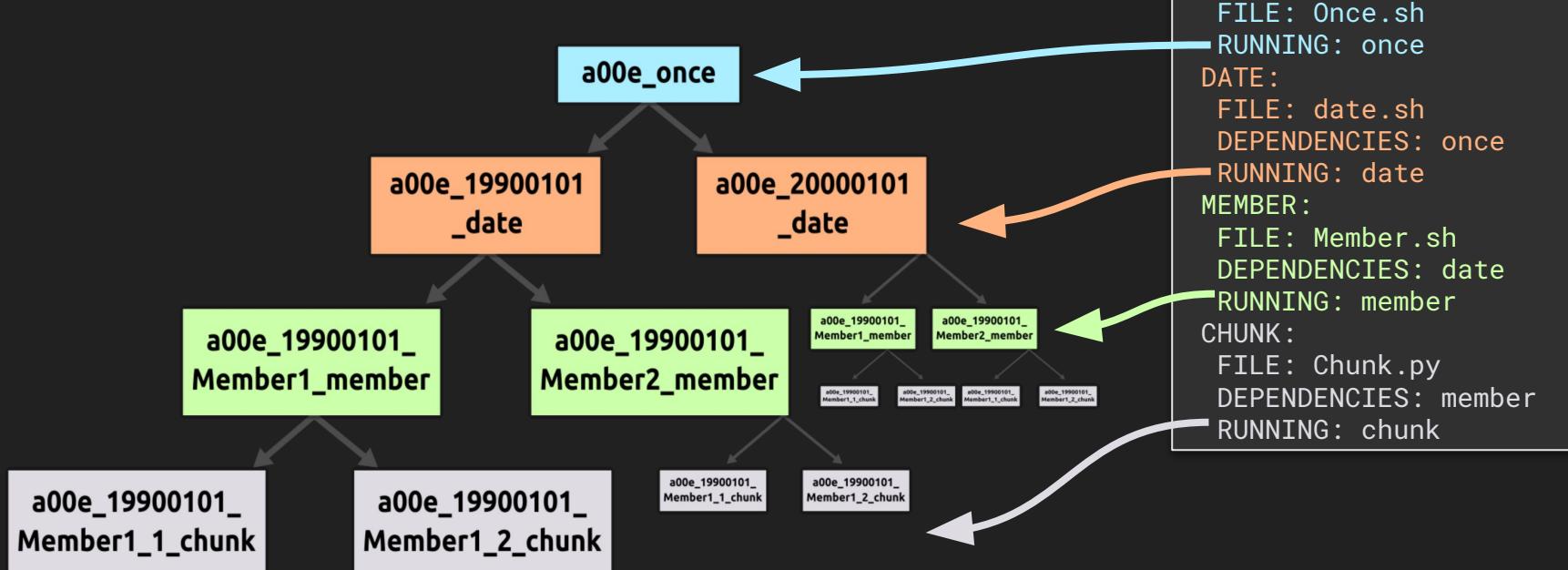
Workflow Hierarchy

YAML



Workflow Hierarchy

YAML



Running

CREATE

CONFIGURE

PREPARE

RUN

MONITOR

CREATE NEW EXPERIMENT

```
autosubmit expid -H "local" -d "My new experiment"
```

CREATE

CONFIGURE

PREPARE

RUN

MONITOR

CREATE NEW EXPERIMENT

```
autosubmit expid -H "local" -d "My new experiment"
```

COPY EXISTING EXPERIMENT

```
autosubmit expid -H "local" -d "My new copy" --copy a00x
```

CREATE

CONFIGURE

PREPARE

RUN

MONITOR

CREATE NEW EXPERIMENT

```
autosubmit expid -H "local" -d "My new experiment"
```

COPY EXISTING EXPERIMENT

```
autosubmit expid -H "local" -d "My new copy" --copy a00x
```

OUTPUT: **Experiment a01a created**

CREATE

CONFIGURE

PREPARE

RUN

MONITOR

AUTOSUBMIT_DATA/a01a/

- |- **conf/** Experiment configuration
- |- **proj/** Project (workflow scripts, config., ...)
- |- **plot/** Visualizations
- |- **tmp/** Logs, templates
- |- **pk1/** Workflow database

CREATE

CONFIGURE

PREPARE

RUN

MONITOR

AUTOSUBMIT_DATA/a01a/

In Autosubmit, the project is a code base external to the experiment configuration that contains the workflow logic as external resources that must be copied.

- **conf/** Experiment configuration
- **proj/** Project (workflow scripts, config., ...)
- **plot/** Visualizations
- **tmp/** Logs, templates
- **pk1/** Workflow database

CREATE

CONFIGURE

PREPARE

RUN

MONITOR

AUTOSUBMIT_DATA/a01a/

In Autosubmit, the **project is a code base external to the experiment configuration** that contains the workflow logic as external resources that must be copied.

configuration files
namelists
YAML files
bash scripts
Python code
...

conf/	Experiment configuration
proj/	Project (workflow scripts, config., ...)
plot/	Visualizations
tmp/	Logs, templates
pk1/	Workflow database

CREATE

CONFIGURE

PREPARE

RUN

MONITOR

AUTOSUBMIT_DATA/a01a/conf/minimal.yml

CONFIG:

 AUTOSUBMIT_VERSION: "4.x.x"

DEFAULT:

 HPCARCH: "local" # -H "local"

PROJECT:

 PROJECT_TYPE: "local" # or git

 PROJECT_DESTINATION: ""

CREATE

CONFIGURE

PREPARE

RUN

MONITOR

AUTOSUBMIT_DATA/a01a/conf/minimal.yml

```
CONFIG:  
  AUTOSUBMIT_VERSION: "4.x.x"  
DEFAULT:  
  HPCARCH: "local"                      # -H "local"  
PROJECT:  
  PROJECT_TYPE: "git"  
  PROJECT_DESTINATION: ""  
GIT:                                         # Because PROJECT_TYPE is "git"  
  PROJECT_ORIGIN: "https://gitlab.bsc.es/..."  
  PROJECT_BRANCH: "v1.2.3"  
  PROJECT_COMMIT: "123456..."
```

CREATE

CONFIGURE

PREPARE

RUN

MONITOR

/appl/AS/AUTOSUBMIT_DATA/a01a/conf/minimal.yml

```
CONFIG:  
  AUTOSUBMIT_VERSION: "4.x.x"  
DEFAULT:  
  EXPID: $expid                                # From `autosubmit expid`  
  HPCARCH: "local"                             # -H "local"  
  CUSTOM_CONFIG: "%PROJDIR%/"                  # Other files?  
PROJECT:  
  PROJECT_TYPE: "git"  
  PROJECT_DESTINATION: ""  
GIT:                                              # Because PROJECT_TYPE is "git"  
  PROJECT_ORIGIN: "https://gitlab.bsc.es/..."  
  PROJECT_BRANCH: "v1.2.3"  
  PROJECT_COMMIT: "123456..."
```

CREATE

CONFIGURE

PREPARE

RUN

MONITOR

/appl/AS/AUTOSUBMIT_DATA/a01a/conf/minimal.yml

Autosubmit will try to load other files located at the **CUSTOM_CONFIG** value.

```
CONFIG:  
    AUTOSUBMIT_VERSION: "4.x.x"  
DEFAULT:  
    EXPID: $expid                                # From `autosubmit expid`  
    HPCARCH: "local"                             # -H "local"  
    CUSTOM_CONFIG: "%PROJDIR%/"                  # Other files?  
PROJECT:  
    PROJECT_TYPE: "git"                          # or local, svn  
    PROJECT_DESTINATION: ""  
GIT:                                              # Because PROJECT_TYPE is "git"  
    PROJECT_ORIGIN: "https://gitlab.bsc.es/..."  
    PROJECT_BRANCH: "v1.2.3"  
    PROJECT_COMMIT: "123456..."
```

CREATE

CONFIGURE

PREPARE

RUN

MONITOR

/appl/AS/AUTOSUBMIT_DATA/a01a/conf/minimal.yml

Autosubmit will try to load other files located at the **CUSTOM_CONFIG** value.

%PROJDIR% is a placeholder variable. Autosubmit replaces this by a value from the workflow configuration.

```
CONFIG:
  AUTOSUBMIT_VERSION: "4.x.x"

DEFAULT:
  EXPID: $expid                                # From `autosubmit expid`
  HPCARCH: "local"                             # -H "local"
  CUSTOM_CONFIG: "%PROJDIR%/"                  # Other files?

PROJECT:
  PROJECT_TYPE: "git"                           # or local, svn
  PROJECT_DESTINATION: ""

GIT:
  PROJECT_ORIGIN: "https://gitlab.bsc.es/..."   # Because PROJECT_TYPE is "git"
  PROJECT_BRANCH: "v1.2.3"
  PROJECT_COMMIT: "123456..."
```

CREATE

CONFIGURE

PREPARE

RUN

MONITOR

```
autosubmit create a01a
```

The first time, it copies the project content into

AUTOSUBMIT_DATA/a01a/proj/

Generates the experiment workflow

Generates

AUTOSUBMIT_DATA/a01a/plot/a01a_\${timestamp}.pdf

CREATE

CONFIGURE

PREPARE

RUN

MONITOR

```
autosubmit run a01a
```

Runs the experiment.

Generates template logs for traceability

```
AUTOSUBMIT_DATA/a01a/tmp/LOG_a01a/
```

Expected output:

```
[local] Connection successful to host localhost
```

```
[local] Correct user privileges for host localhost
```

CREATE

CONFIGURE

PREPARE

RUN

MONITOR

```
autosubmit run a01a
```

NO HANG-UP MODE

```
nohup autosubmit monitor a01a &
```

Visualize console output:

```
tail -f nohup.out
```

CREATE

CONFIGURE

PREPARE

RUN

MONITOR

```
autosubmit monitor a01a
```

CREATE

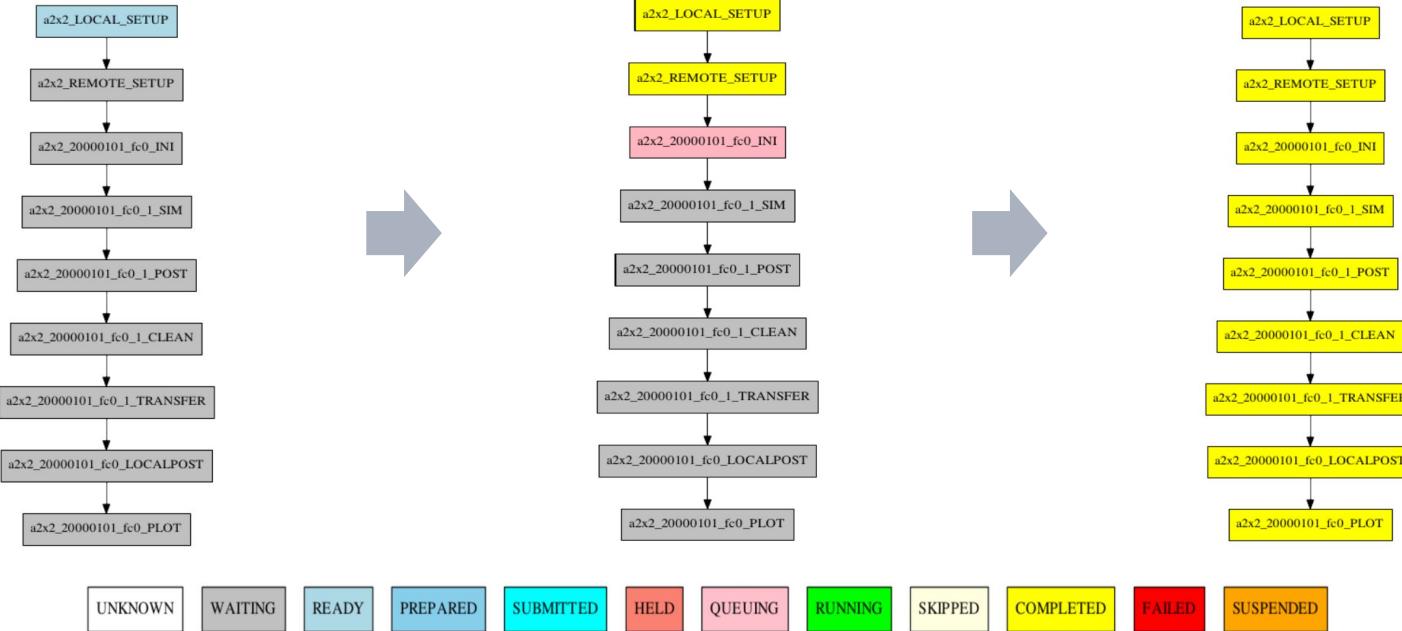
CONFIGURE

PREPARE

RUN

MONITOR

autosubmit monitor a01a



CREATE

CONFIGURE

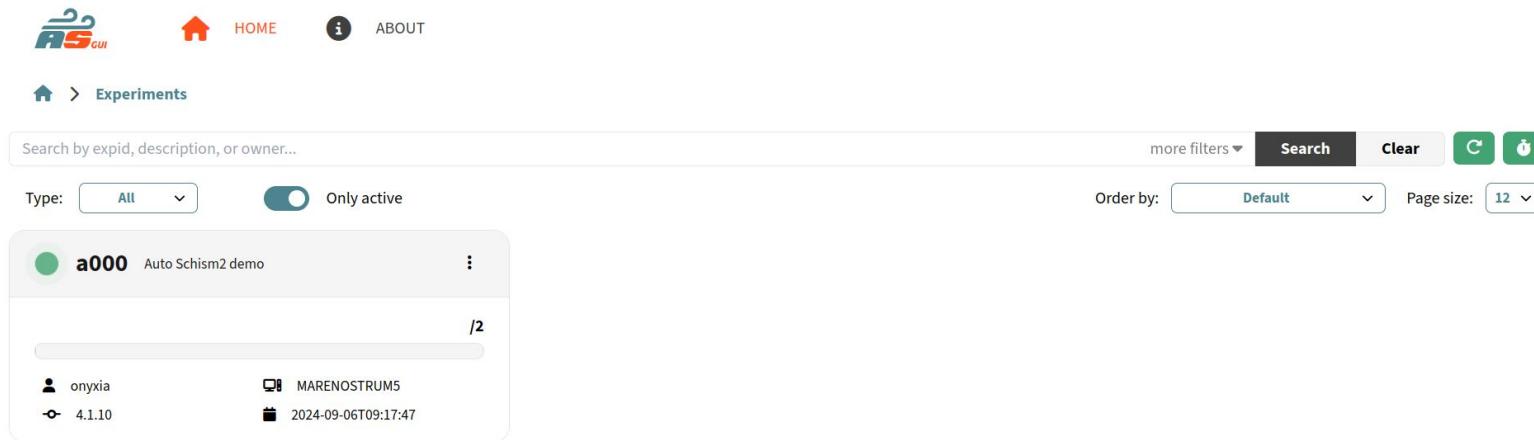
PREPARE

RUN

MONITOR

Monitoring

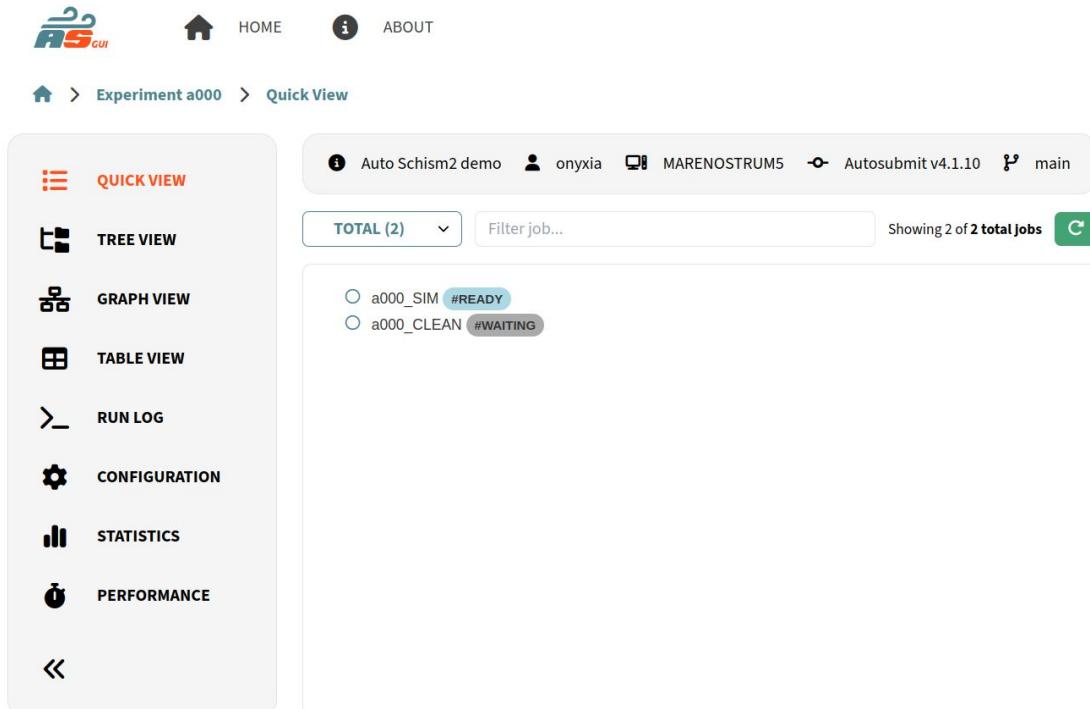
Autosubmit GUI



The screenshot shows the Autosubmit GUI interface. At the top, there is a navigation bar with a logo, a 'HOME' button, and an 'ABOUT' button. Below the navigation bar, the page title is 'Experiments'. A search bar is present with the placeholder 'Search by expid, description, or owner..'. To the right of the search bar are buttons for 'more filters', 'Search', 'Clear', and two green buttons with icons (one with a 'C' and one with a 'D'). Below the search bar are filters for 'Type' (set to 'All'), 'Only active' (a toggle switch), 'Order by' (set to 'Default'), and 'Page size' (set to '12'). The main content area displays a list of experiments. The first experiment in the list is 'a000 Auto Schism2 demo', which has a green status indicator. Below the experiment name is a progress bar showing '/2'. To the left of the progress bar is a user icon (person) and the name 'onyxia'. To the right is a monitor icon and the name 'MARENOSTRUM5'. At the bottom of the experiment card are two more details: a '4.1.10' icon and a timestamp '2024-09-06T09:17:47'.

Autosubmit GUI

QUICK VIEW



The screenshot shows the Autosubmit GUI interface. At the top, there is a navigation bar with a logo, a home icon, and an 'ABOUT' link. Below the navigation bar, the breadcrumb navigation shows 'Experiment a000 > Quick View'. The main content area is divided into two sections: a sidebar on the left and a main panel on the right.

SIDEBAR (Left):

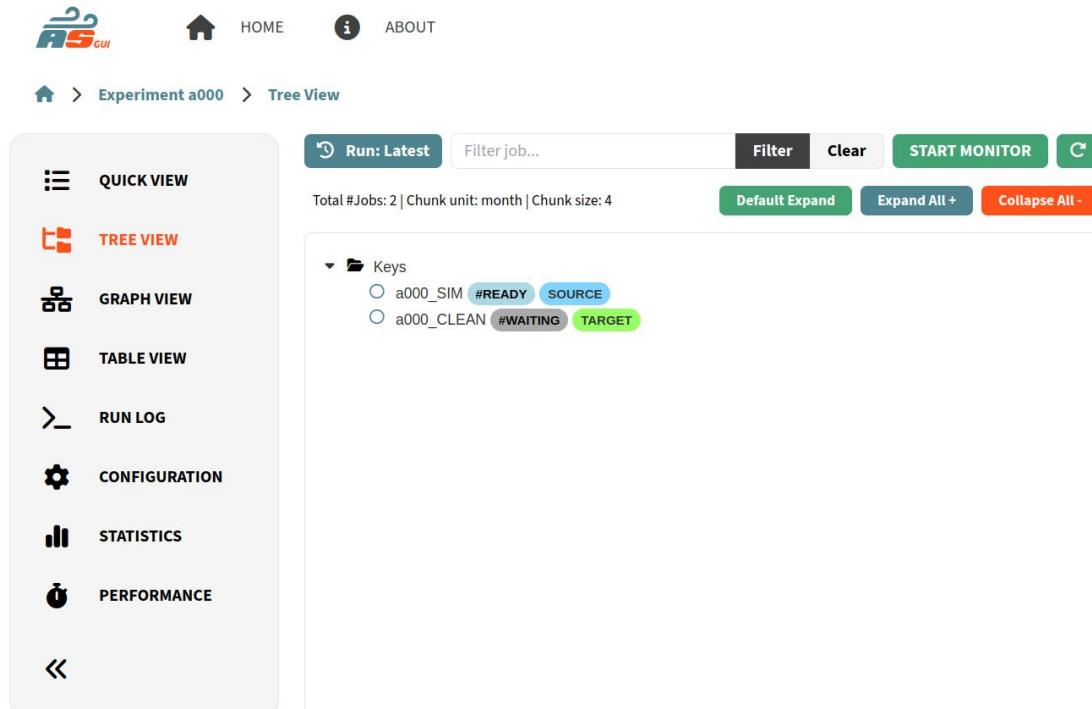
- QUICK VIEW** (selected, highlighted in red)
- TREE VIEW**
- GRAPH VIEW**
- TABLE VIEW**
- RUN LOG**
- CONFIGURATION**
- STATISTICS**
- PERFORMANCE**

MAIN PANEL (Right):

- Header:** Auto Schism2 demo, onyxia, MARENOSTRUM5, Autosubmit v4.1.10, main
- Job Summary:** TOTAL (2) (dropdown), Filter job..., Showing 2 of 2 total jobs, C
- Job List:** a000_SIM (#READY) and a000_CLEAN (#WAITING)

Autosubmit GUI

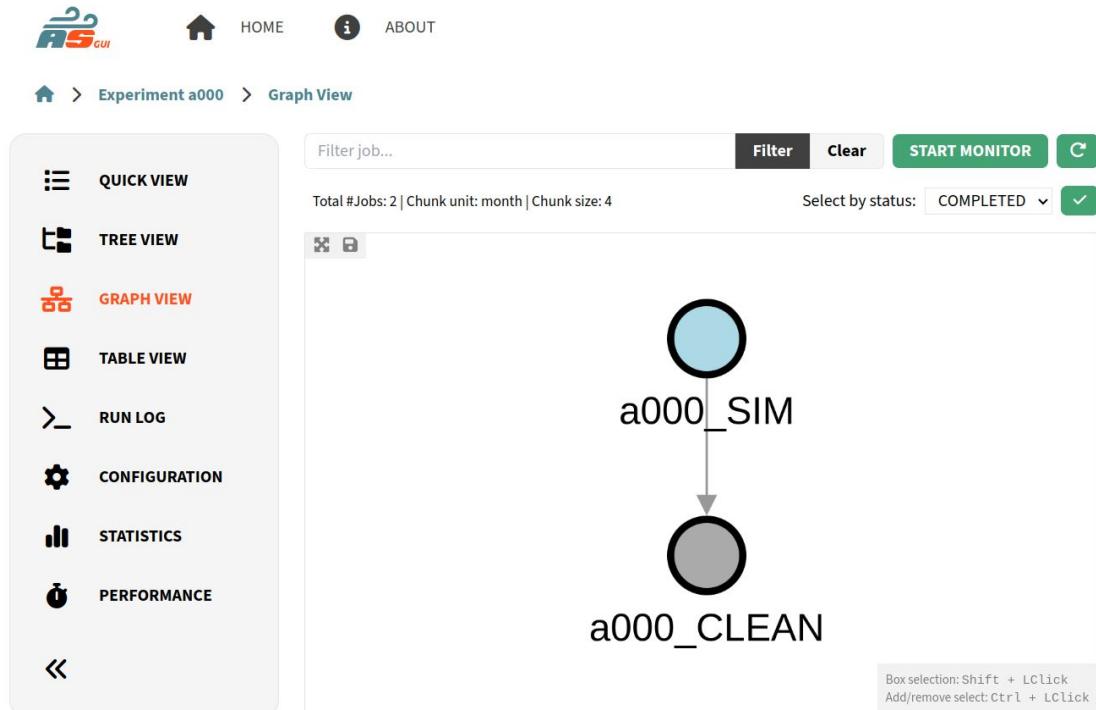
TREE VIEW



The screenshot shows the Autosubmit GUI interface. At the top, there is a navigation bar with a logo, a home icon, and an 'ABOUT' link. Below the navigation bar, the breadcrumb navigation shows 'Experiment a000 > Tree View'. On the left, a sidebar menu lists various options: QUICK VIEW, TREE VIEW (which is selected and highlighted in orange), GRAPH VIEW, TABLE VIEW, RUN LOG, CONFIGURATION, STATISTICS, and PERFORMANCE. At the bottom of the sidebar is a double-left arrow icon. The main content area has a header with a 'Run: Latest' button, a 'Filter job...' input field, and buttons for 'Filter', 'Clear', 'START MONITOR', and a refresh icon. Below this, a message states 'Total #Jobs: 2 | Chunk unit: month | Chunk size: 4' with buttons for 'Default Expand', 'Expand All +', and 'Collapse All -'. The main content area displays a tree view under a 'Keys' folder. The tree view shows two items: 'a000_SIM' and 'a000_CLEAN'. 'a000_SIM' is marked with a blue '#READY' status and a blue 'SOURCE' status. 'a000_CLEAN' is marked with a grey '#WAITING' status and a green 'TARGET' status. The entire interface has a clean, modern design with a light gray background and blue and green accents for active items.

Autosubmit GUI

GRAPH VIEW



Autosubmit GUI

CONFIGURATION

The screenshot shows the Autosubmit GUI configuration interface. The top navigation bar includes a logo, a home icon, 'HOME', and an 'ABOUT' icon. Below the navigation is a breadcrumb trail: 'Experiment a000 > Configuration'. The left sidebar contains several icons: a list, a grid, a file, a search, a graph, a gear (highlighted in orange), a bar chart, and a double arrow. The main content area is divided into several sections:

- CURRENT RUN CONFIGURATION (HISTORICAL DATABASE)**

Setting	Value
PROJDIR	/home/onyxia/autosubmit/a000/proj/project_files
ROOTDIR	/home/onyxia/autosubmit/a00
contains_nones	false
- CURRENT FILESYSTEM CONFIGURATION**

Setting	Value
PROJDIR	/home/onyxia/autosubmit/a000/proj/project_files
ROOTDIR	/home/onyxia/autosubmit/a00
contains_nones	false
- [CONFIG]**

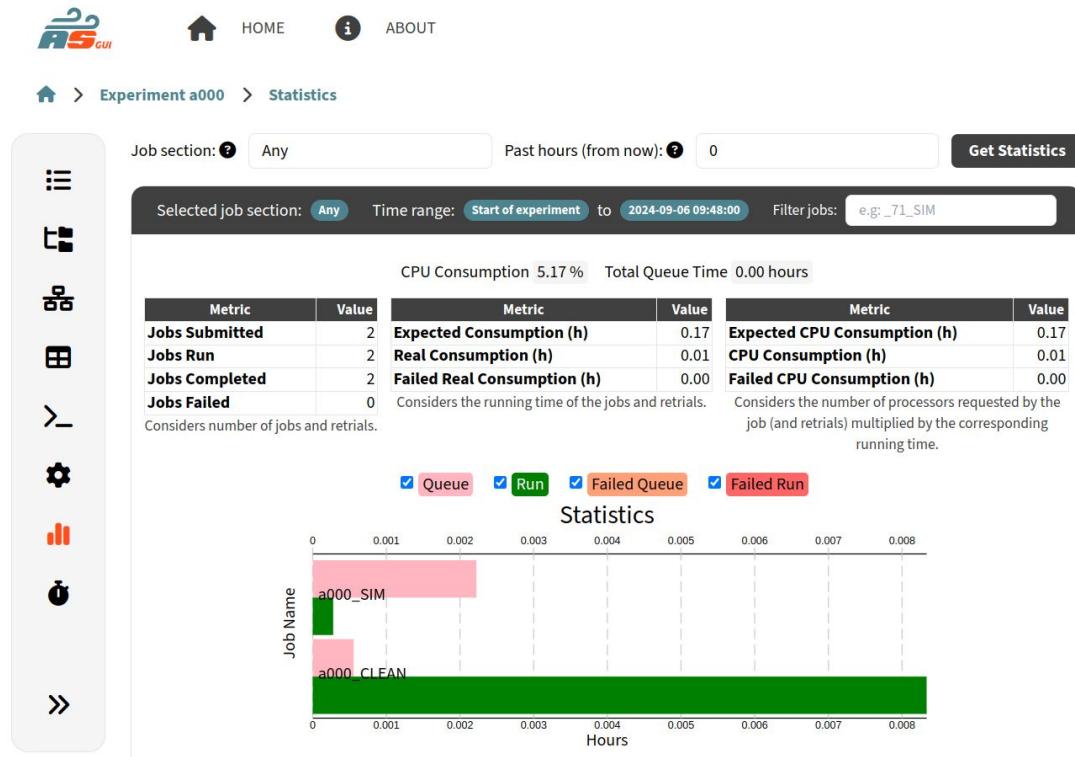
Setting	Value
AUTOSUBMIT_VERSION	4.1.10
MAXWAITINGJOBS	20
TOTALJOBS	20
- [DEFAULT]**

Setting	Value
EXPID	a000
HPCARCH	MARENOSTRUM5
- [EXPERIMENTAL]**

Setting	Value
PROJDIR	/home/onyxia/autosubmit/a000/proj/project_files
ROOTDIR	/home/onyxia/autosubmit/a00
contains_nones	false

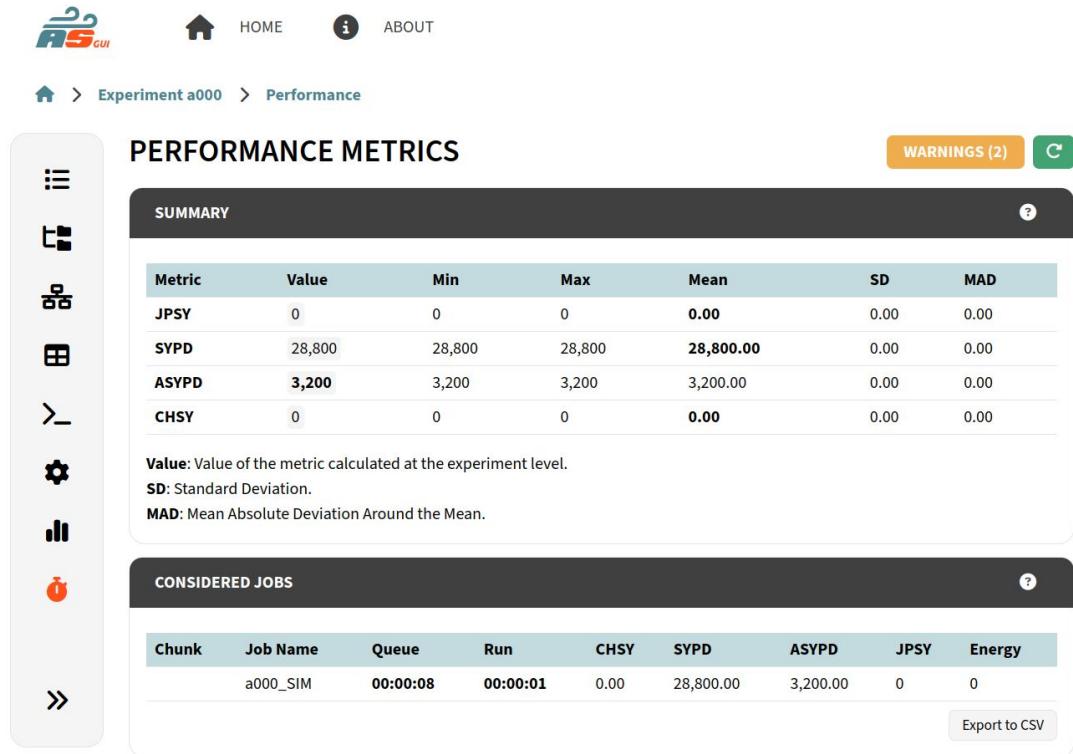
Autosubmit GUI

STATISTICS



Autosubmit GUI

PERFORMANCE



The screenshot shows the Autosubmit GUI interface. At the top, there is a navigation bar with a logo, a home icon, and 'HOME' and 'ABOUT' buttons. Below the navigation bar, the breadcrumb navigation shows 'Experiment a000 > Performance'. The main content area is titled 'PERFORMANCE METRICS' and contains a table for 'SUMMARY' metrics. The table includes columns for Metric, Value, Min, Max, Mean, SD, and MAD. The 'Value' column for JPSY is 0, for SYPD is 28,800, for ASYPD is 3,200, and for CHSY is 0. The 'Mean' column for all metrics is 0.00. Below the table, definitions for Value, SD, and MAD are provided. The 'Value' is the calculated metric at the experiment level, 'SD' is Standard Deviation, and 'MAD' is Mean Absolute Deviation Around the Mean. The 'CONSIDERED JOBS' section shows a table with columns for Chunk, Job Name, Queue, Run, CHSY, SYPD, ASYPD, JPSY, and Energy. The 'a000_SIM' job has a Run time of 00:00:08, CHSY of 0.00, SYPD of 28,800.00, ASYPD of 3,200.00, and both JPSY and Energy at 0. At the bottom right, there is a 'Export to CSV' button.

AS GUI

HOME ABOUT

Experiment a000 > Performance

PERFORMANCE METRICS

WARNINGS (2) C

SUMMARY

Metric	Value	Min	Max	Mean	SD	MAD
JPSY	0	0	0	0.00	0.00	0.00
SYPD	28,800	28,800	28,800	28,800.00	0.00	0.00
ASYPD	3,200	3,200	3,200	3,200.00	0.00	0.00
CHSY	0	0	0	0.00	0.00	0.00

Value: Value of the metric calculated at the experiment level.
SD: Standard Deviation.
MAD: Mean Absolute Deviation Around the Mean.

CONSIDERED JOBS

Chunk	Job Name	Queue	Run	CHSY	SYPD	ASYPD	JPSY	Energy
	a000_SIM	00:00:08	00:00:01	0.00	28,800.00	3,200.00	0	0

Export to CSV

Autosubmit API

A *comprehensive waypoint* to the
Autosubmit persistence layer

Autosubmit API

GET /v3/expinfo/a75t

Autosubmit API

GET /v3/expinfo/a75t

```
{  
  "branch": "FORCeS-RaFSIPv2-DURF-MELUXINA",  
  "chunk_size": 12,  
  "chunk_unit": "month",  
  "completed_jobs": 801,  
  "db_historic_version": 18,  
  "description": "[DURF] EC-Earth3-FORCeS AMIP-IFS-TM5 1850-2000 histSST BASELINE",  
  "error": false,  
  "error_message": "",  
  "expid": "a75t",  
  "hpc": "marenostrom5",  
  "model": "https://earth.bsc.es/gitlab/es/auto-ecearth3.git",  
  "owner": "mariag",  
  "owner_id": 7464,  
  "path": "/esarchive/autosubmit/a75t",  
  "pkl_timestamp": 1718884227,  
  "running": true,  
  "time_last_access": "2024-06-20 02:04:43",  
  "time_last_mod": "2024-05-09 13:40:29",  
  "total_jobs": 1056,  
  "updateTime": 10,  
  "version": "3.15.14"  
}
```

Autosubmit API

RESTful Interface



Open API
Specification



Swagger

Autosubmit API 4.0.0 OAS3

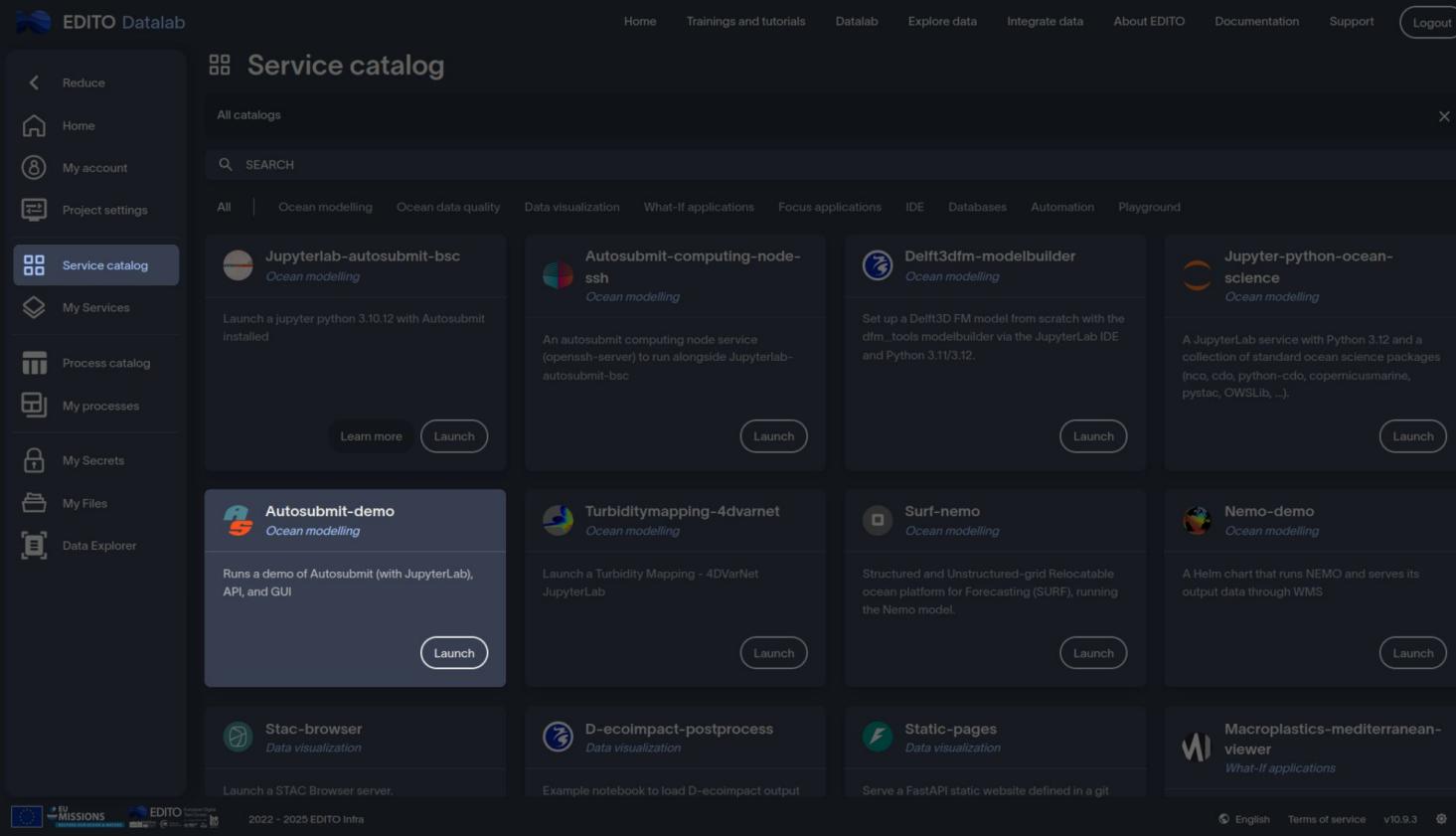
GNU General Public License

v3

<code>POST</code>	<code>/v3/login</code> Login	<code>post_v3_login</code> ▾
<code>GET</code>	<code>/v3/tokentest</code> Token test	<code>get_v3_tokentest</code> ▾
<code>POST</code>	<code>/v3/tokentest</code> Token test	<code>post_v3_tokentest</code> ▾
<code>GET</code>	<code>/v3/updatedesc</code> Update description	<code>get_v3_updatedesc</code> ▾
<code>POST</code>	<code>/v3/updatedesc</code> Update description	<code>post_v3_updatedesc</code> ▾
<code>GET</code>	<code>/v3/cconfig/{expid}</code> Get current configuration	<code>get_v3_cconfig_expid</code> ▾
<code>GET</code>	<code>/v3/expinfo/{expid}</code> Get experiment info	<code>get_v3_expinfo_expid</code> ▾
<code>GET</code>	<code>/v3/expcount/{expid}</code> Get experiment counters	<code>get_v3_expcount_expid</code> ▾
<code>GET</code>	<code>/v3/searchowner/{owner}/{exptype}/{onlyactive}</code> Search owner	<code>get_v3_searchowner_owner_exptype_onlyactive</code> ▾
<code>GET</code>	<code>/v3/searchowner/{owner}</code> Search owner	<code>get_v3_searchowner_owner</code> ▾
<code>GET</code>	<code>/v3/search/{expid}/{exptype}/{onlyactive}</code> Search experiment	<code>get_v3_search_expid_exptype_onlyactive</code> ▾
<code>GET</code>	<code>/v3/search/{expid}</code> Search experiment	<code>get_v3_search_expid</code> ▾
<code>GET</code>	<code>/v3/running</code> Running experiments	<code>get_v3_running</code> ▾
<code>GET</code>	<code>/v3/runs/{expid}</code> Get runs	<code>get_v3_runs_expid</code> ▾

Hands On

Step 1. Find the Autosubmit-demo service card in the Service Catalog



The screenshot shows the EDITO Datalab Service Catalog interface. The left sidebar contains navigation links: Reduce, Home, My account, Project settings, Service catalog (selected), My Services, Process catalog, My processes, My Secrets, My Files, and Data Explorer. The main area is titled "Service catalog" and shows a grid of service cards. The "Autosubmit-demo" service card is highlighted with a dark blue background and white text. It has a "Launch" button. Other visible service cards include "Jupyterlab-autosubmit-bsc", "Autosubmit-computing-node-ssh", "Delft3dfm-modelbuilder", "Jupyter-python-ocean-science", "Turbiditymapping-4dvarnet", "Surf-nemo", "Nemo-demo", "Stac-browser", "D-ecolmpact-postprocess", "Static-pages", and "Macroplastics-mediterranean-viewer". The bottom of the page includes footer links for EU MISSIONS, EDITO Infra, and various documentation links.

EDITO Datalab

Service catalog

All catalogs

SEARCH

Home Trainings and tutorials Datalab Explore data Integrate data About EDITO Documentation Support Logout

Autosubmit-demo

Ocean modelling

Launch a demo of Autosubmit (with JupyterLab), API, and GUI

Launch

Autosubmit-computing-node-ssh

Ocean modelling

An autosubmit computing node service (openssh-server) to run alongside Jupyterlab-autosubmit-bsc

Launch

Delft3dfm-modelbuilder

Ocean modelling

Set up a Delft3D FM model from scratch with the dflm... tools modelbuilder via the JupyterLab IDE and Python 3.11/3.12.

Launch

Jupyter-python-ocean-science

Ocean modelling

A JupyterLab service with Python 3.12 and a collection of standard ocean science packages (rco, cdo, python-cdo, copernicusmarine, pystac, OWSLib, ...).

Launch

Autosubmit-demo

Ocean modelling

Runs a demo of Autosubmit (with JupyterLab), API, and GUI

Launch

Turbiditymapping-4dvarnet

Ocean modelling

Launch a Turbidity Mapping - 4DVarNet JupyterLab

Launch

Surf-nemo

Ocean modelling

Structured and Unstructured-grid Relocatable ocean platform for Forecasting (SURF), running the Nemo model.

Launch

Nemo-demo

Ocean modelling

A Helm chart that runs NEMO and serves its output data through WMS

Launch

Stac-browser

Data visualization

Launch a STAC Browser server.

Launch

D-ecolmpact-postprocess

Data visualization

Example notebook to load D-ecolmpact output

Launch

Static-pages

Data visualization

Serve a FastAPI static website defined in a git

Macroplastics-mediterranean-viewer

What-If applications

EU MISSIONS EDITO Infra

2022 - 2025 EDITO Infra

English Terms of service v10.9.3

Step 2. Review your configuration (if desired) and click Launch.

The screenshot shows the EDITO Datalab interface. The left sidebar contains navigation links: Reduce, Home, My account, Project settings, Service catalog, My Services, Process catalog, My processes, My Secrets, My Files, and Data Explorer. The main content area is titled 'Autosubmit-demo' and shows a Helm chart configuration. At the top, a command line interface (CLI) bar displays the command: '\$ helm install autosubmit-demo-920428 ocean-modelling/autosubmit-demo -f values.yaml'. Below this, the configuration details are listed:

- Friendly name:** autosubmit-demo
- Version:** 0.1.36
- Form** (radio button selected)
- Text Editor** (radio button)
- Security** (security specific configuration)
- Resources** (Your service will have at least the requested resources and never more than its limits. No limit for a resource and you can consume everything left on the host machine.)
- Ingress Details**
- Vault** (Configuration of vault client)
- S3 Configuration** (Configuration of temporary Identity for AWS S3)

At the bottom right of the configuration panel are 'Cancel' and 'Launch' buttons. The 'Launch' button is highlighted with a blue border. The top navigation bar includes links for Home, Trainings and tutorials, Datalab, Explore data, Integrate data, About EDITO, Documentation, Support, and Logout. The bottom of the page features logos for EU MISSIONS, EDITO Infra, and the European Data Infrastructure, along with links for English, Terms of service, and v10.9.3.

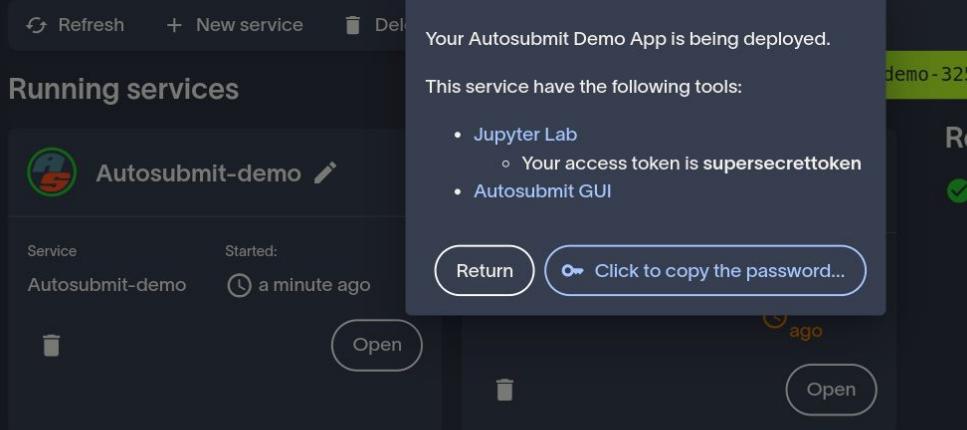
Step 3. Wait until your new Autosubmit personal service is deployed

The screenshot shows the EDITO Datalab interface with a dark theme. The top navigation bar includes links for Home, Trainings and tutorials, Datalab, Explore data, Integrate data, About EDITO, Documentation, Support, and Logout. A dropdown menu for the current project, "Itenorio personal project", is open. The left sidebar contains links for Reduce, Home, My account, Project settings, Service catalog, My services, Process catalog, My processes, My secrets, and My files. The main content area is titled "My Services" and displays a message: "Access your running services. Services are supposed to be shut down." Below this, a "Running services" section shows a service named "Autosubmit-demo" with a status of "Container starting". A modal window is open, stating "Your Autosubmit Demo App is being deployed." and listing "This service have the following tools: Jupyter Lab (Your access token is supersecrettoken) and Autosubmit GUI". It also shows the command "Pulling image "autosubmit/gui:edito-demo"" being executed in a terminal window. The terminal window title is "demo-325530 --namespace user-ltenorio". The bottom right corner of the screen shows resource usage quotas with a green checkmark indicating "Your current resource usage is reasonable." The footer includes logos for EU MISSIONS, RESTORE OUR OCEAN & WATERS, and EDITO, along with copyright information: "2022 - 2024 EDITO Infra". The footer also includes links for English, Terms of service, and version information: "v8.270".

❖ My Services

😊 Access your running services

Services are supposed to be shut down as soon as you stop using them actively.



The screenshot shows the 'My Services' interface. At the top, there are buttons for 'Refresh', '+ New service', and 'Delete'. Below this, a section titled 'Running services' lists a service named 'Autosubmit-demo'. The service details show it was started 'a minute ago'. To the right of the service list is a modal window with the following content:

Your Autosubmit Demo App is being deployed.

This service have the following tools:

- Jupyter Lab
 - Your access token is **supersecrettoken**
- Autosubmit GUI

Buttons at the bottom of the modal include 'Return' and 'Click to copy the password...'. To the right of the modal, there is a green box labeled 'demo - 325' with a 'Re' button and a green checkmark icon.

Jupyter Lab: service with a notebook with Autosubmit instance running, and access to HPC

Autosubmit GUI: web app to monitor Autosubmit experiments

Step 4. On Jupyter Lab, you'll have to use your access token from the previous page

The image shows a screenshot of the EDITO Datalab web interface. On the left, a sidebar menu includes options like 'Reduce', 'Home', 'My account', 'Project settings', 'Service catalog', 'My Services' (which is selected and highlighted in blue), 'Process catalog', 'My processes', 'My Secrets', 'My Files', and 'Data Explorer'. The main content area is titled 'My Services' and shows a list of 'Running services'. One service, 'Autosubmit-demo', is listed twice: once with a status of 'Started: 6 minutes ago' and once with 'Started: 4 days ago'. Each service entry has an 'Open' button. To the right of the services, there is a 'Resource usage quota' section and a 'Token authentication is enabled' section. The 'Token authentication is enabled' section contains a 'Password or token:' input field and a 'Log in' button. A modal dialog box is overlaid on the page, titled 'jupyter'. It contains a 'Password or token:' input field and a 'Log in' button. The modal also contains text about token authentication, a command line example, and a 'Setup a Password' section with fields for 'Token' and 'New Password'.

My Services

Access your running services

Services are supposed to be shut down as soon as you stop using them actively.

Refresh + New service Delete all Events

Running services

Autosubmit-demo

Service Started: Autosubmit-demo 6 minutes ago

Autosubmit-computing-node-ssh

Your Autosubmit Demo App is being deployed.

This service have the following tools:

- Jupyter Lab
 - Your access token is `supersecrettoken`
- Autosubmit GUI

Return Click to copy the password...

Resource usage quota

Your current resource usage

Token authentication is enabled

If no password has been configured, you need to open the server with its login token in the URL or paste it above. This requirement will be lifted if you enable a password.

The command:

```
jupyter server list
```

will show you the URLs of running servers with their tokens, which you can copy and paste into your browser. For example:

```
Currently running servers:  
http://localhost:8888/?token=chde54fa... :1 /Users/you/notebooks
```

or you can paste just the token value into the password field on this page.

See the documentation on how to enable a password in place of token authentication, if you would like to avoid dealing with random tokens.

Cookies are required for authenticated access to the Jupyter server.

Setup a Password

You can also setup a password by entering your token and a new password on the fields below:

Token

New Password

Log in and set new password

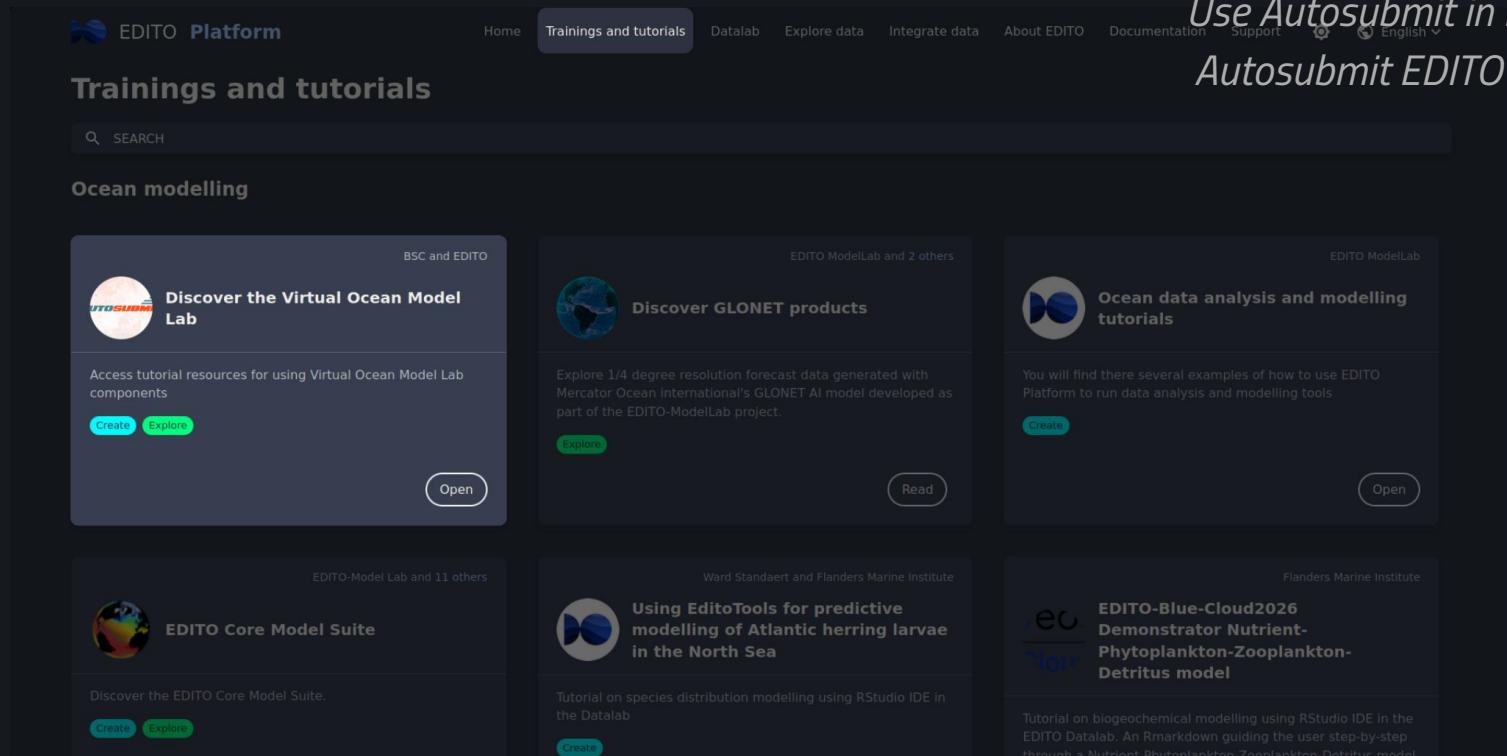
EU MISSIONS EDITO European Data Infra-structure

2022 - 2025 EDITO Infra

English Terms of service v10.9.3

Step 5. Find the "Autosubmit EDITO training"

*Trainings and Tutorials >
Discover the Virtual Ocean Model Lab >
Use Autosubmit in EDITO >
Autosubmit EDITO training*



The screenshot shows the 'Trainings and tutorials' section of the EDITO Platform. The top navigation bar includes links for Home, Trainings and tutorials (which is the active tab), Databab, Explore data, Integrate data, About EDITO, Documentation, Support, and a language dropdown set to English. Below the navigation is a search bar with the placeholder 'SEARCH'. The main content area is titled 'Trainings and tutorials' and features a section for 'Ocean modelling'.

Trainings and tutorials

Ocean modelling

Discover the Virtual Ocean Model Lab (BSC and EDITO) - Access tutorial resources for using Virtual Ocean Model Lab components. Includes 'Create' and 'Explore' buttons and an 'Open' button.

Discover GLONET products (EDITO ModelLab and 2 others) - Explore 1/4 degree resolution forecast data generated with Mercator Ocean International's GLONET AI model developed as part of the EDITO-ModelLab project. Includes an 'Explore' button and an 'Open' button.

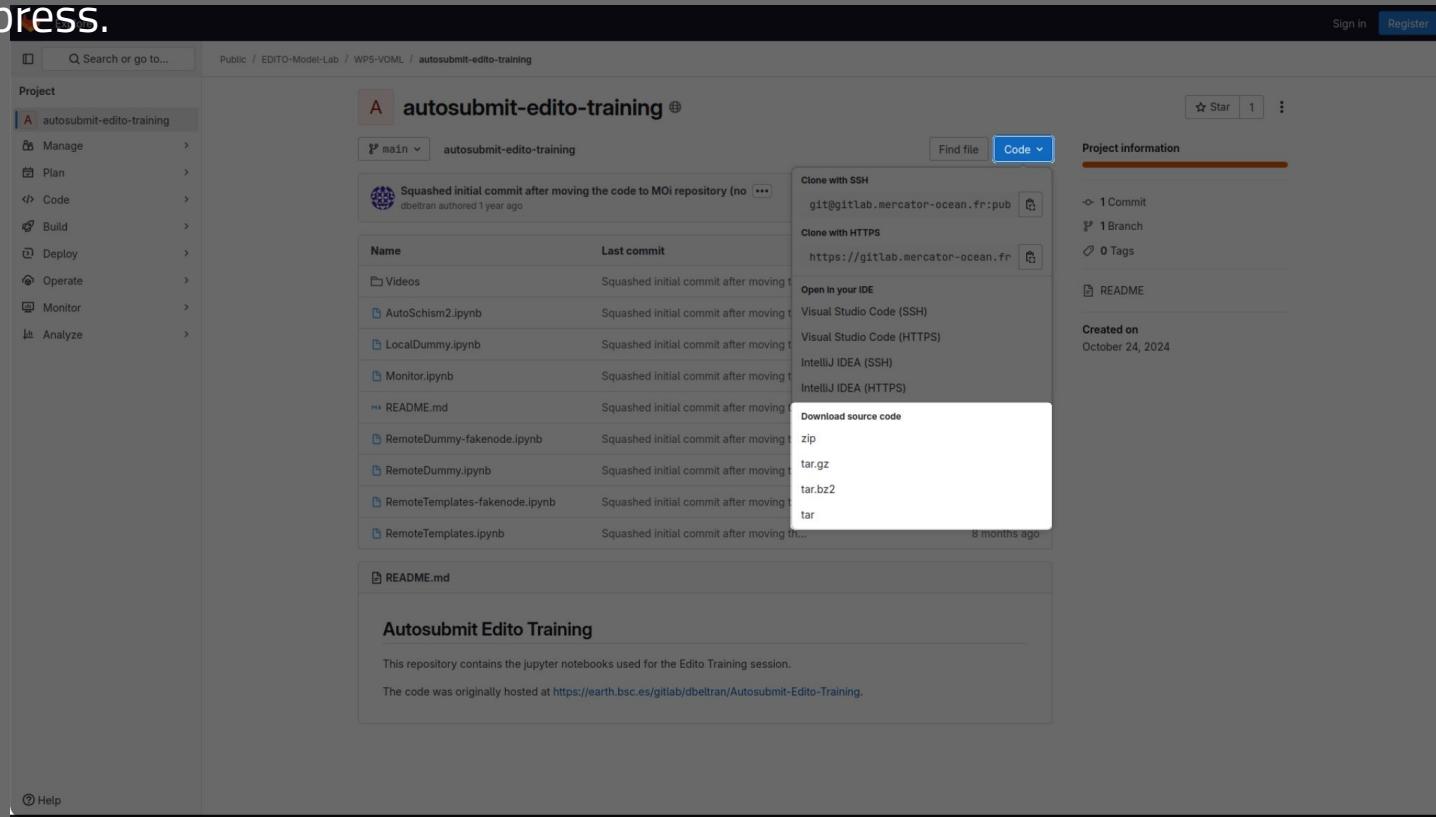
Ocean data analysis and modelling tutorials (EDITO ModelLab) - You will find there several examples of how to use EDITO Platform to run data analysis and modelling tools. Includes a 'Create' button and an 'Open' button.

EDITO Core Model Suite (EDITO-Model Lab and 11 others) - Discover the EDITO Core Model Suite. Includes 'Create' and 'Explore' buttons and an 'Open' button.

Using EditoTools for predictive modelling of Atlantic herring larvae in the North Sea (Ward Standaert and Flanders Marine Institute) - Tutorial on species distribution modelling using RStudio IDE in the Databab. Includes a 'Create' button.

EDITO-Blue-Cloud2026 Demonstrator Nutrient-Phytoplankton-Zooplankton-Detritus model (Flanders Marine Institute) - Tutorial on biogeochemical modelling using RStudio IDE in the EDITO Databab. An Rmarkdown guiding the user step-by-step through a Nutrient-Phytoplankton-Zooplankton-Detritus model. Includes an 'Open' button.

Step 6. Download the repository code in your preferred compression format and decompress.



The screenshot shows a GitLab repository page for 'autosubmit-edito-training'. The repository has 1 commit, 1 branch, and 0 tags. The README file is present. A context menu is open over the repository information, showing options to 'Clone with SSH' and 'Clone with HTTPS', as well as 'Open in your IDE' (Visual Studio Code, IntelliJ IDEA) and 'Download source code' (zip, tar.gz, tar.bz2, tar). The 'Code' tab is selected in the top navigation bar.

Project

autosubmit-edito-training

Code

Find file

Code

Project information

1 Commit

1 Branch

0 Tags

README

Created on

October 24, 2024

Clone with SSH

git@gitlab.mercator-ocean.fr:pub

Clone with HTTPS

<https://gitlab.mercator-ocean.fr>

Open in your IDE

Visual Studio Code (SSH)

Visual Studio Code (HTTPS)

IntelliJ IDEA (SSH)

IntelliJ IDEA (HTTPS)

Download source code

zip

tar.gz

tar.bz2

tar

8 months ago

README.md

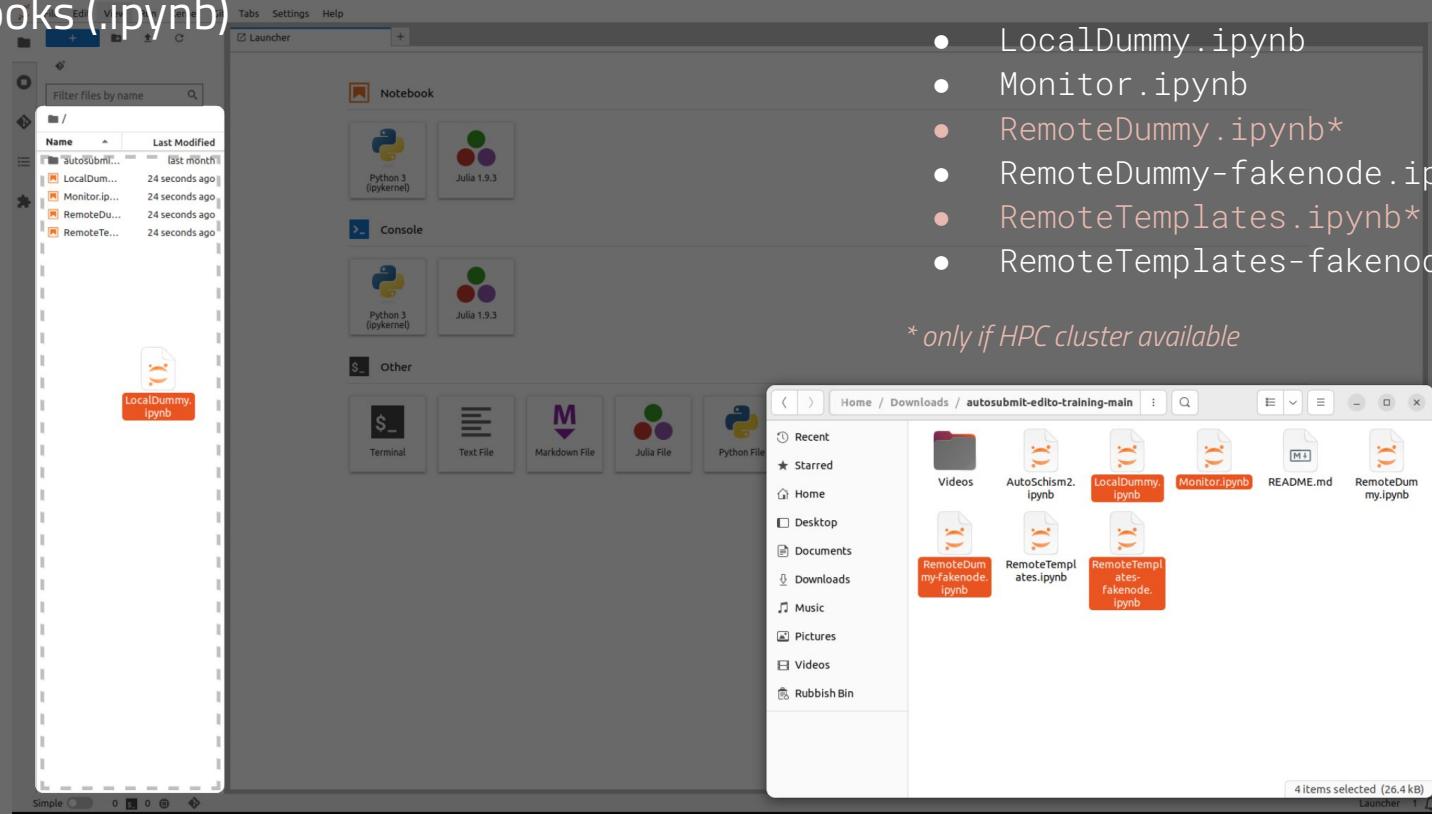
Autosubmit Edito Training

This repository contains the jupyter notebooks used for the Edito Training session.

The code was originally hosted at <https://earth.bsc.es/gitlab/dbeltran/Autosubmit-Edito-Training>.

Help

Step 7. Go to the Jupyter Lab open in Step 4, drag and drop the compressed Jupyter Notebooks (.ipynb)



Step 8. In the Service Catalog, launch the *Autosubmit-computing-node-ssh*

EDITO Datalab

Home Trainings and tutorials Datalab Explore data Integrate data About EDITO Documentation Support Logout

Service catalog

All catalogs SEARCH

All Ocean modelling Ocean data quality Data visualization What-if applications Focus applications IDE Databases Automation Playground

Autosubmit-computing-node-ssh
Ocean modelling

An autosubmit computing node service (openssh-server) to run alongside JupyterLab-autosubmit-bsc

Launch

Jupyterlab-autosubmit-bsc
Ocean modelling

Launch a jupyter python 3.10.12 with Autosubmit installed

Learn more Launch

Delft3dfm-modelbuilder
Ocean modelling

Set up a Delft3D FM model from scratch with the dflm_tools modelbuilder via the JupyterLab IDE and Python 3.11/3.12.

Launch

Jupyter-python-ocean-science
Ocean modelling

A JupyterLab service with Python 3.12 and a collection of standard ocean science packages (rco, cdo, python-cdo, copernicusmarine, pystac, OWSLib, ...).

Launch

Autosubmit-demo
Ocean modelling

Runs a demo of Autosubmit (with JupyterLab), API, and GUI

Launch

Turbiditymapping-4dvarnet
Ocean modelling

Launch a Turbidity Mapping - 4DVarNet JupyterLab

Launch

Surf-nemo
Ocean modelling

Structured and Unstructured-grid Relocatable ocean platform for Forecasting (SURF), running the Nemo model.

Launch

Nemo-demo
Ocean modelling

A Helm chart that runs NEMO and serves its output data through WMS

Launch

Stac-browser
Data visualization

Launch a STAC Browser server.

Launch

D-ecolmpact-postprocess
Data visualization

Example notebook to load D-ecolmpact output

Launch

Static-pages
Data visualization

Serve a FastAPI static website defined in a git

Launch

Macroplastics-mediterranean-viewer
What-if applications

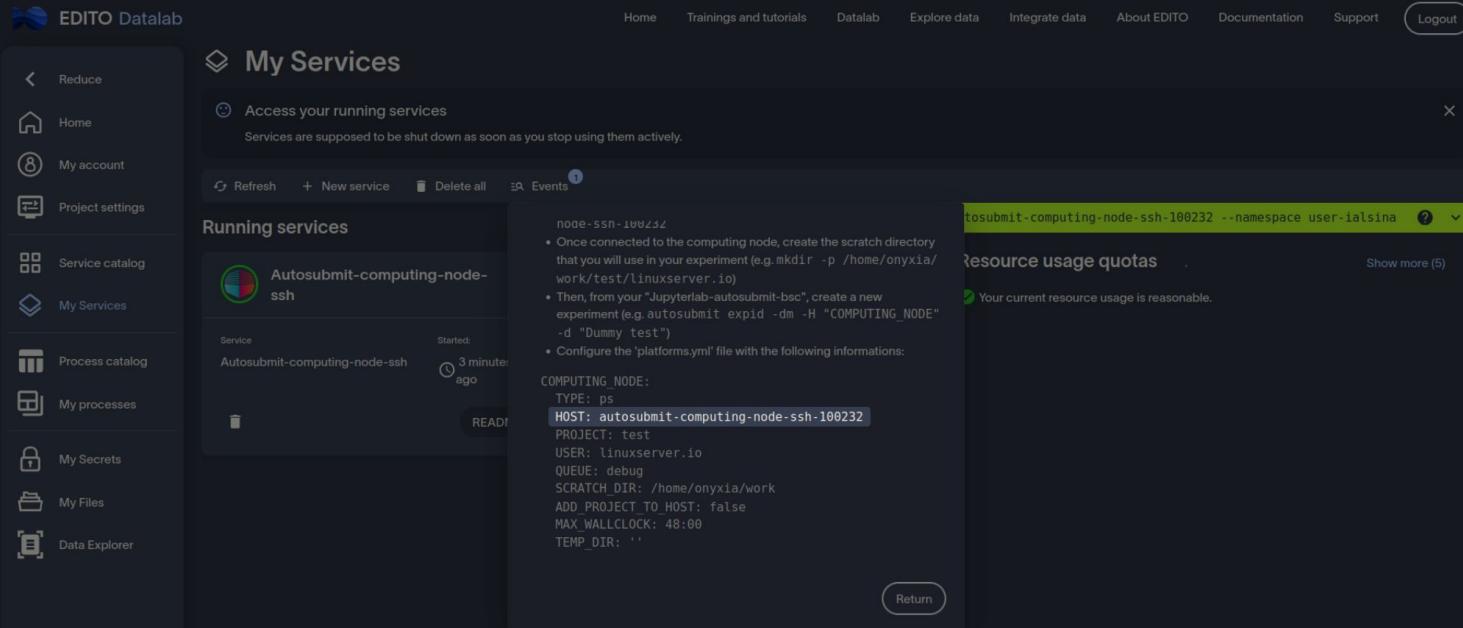
Launch

EU MISSIONS EDITO European Data Infra Infra

2022 - 2025 EDITO Infra

English Terms of service v10.9.3

Step 9. Once the service is launched, take note of the *HOST* key.



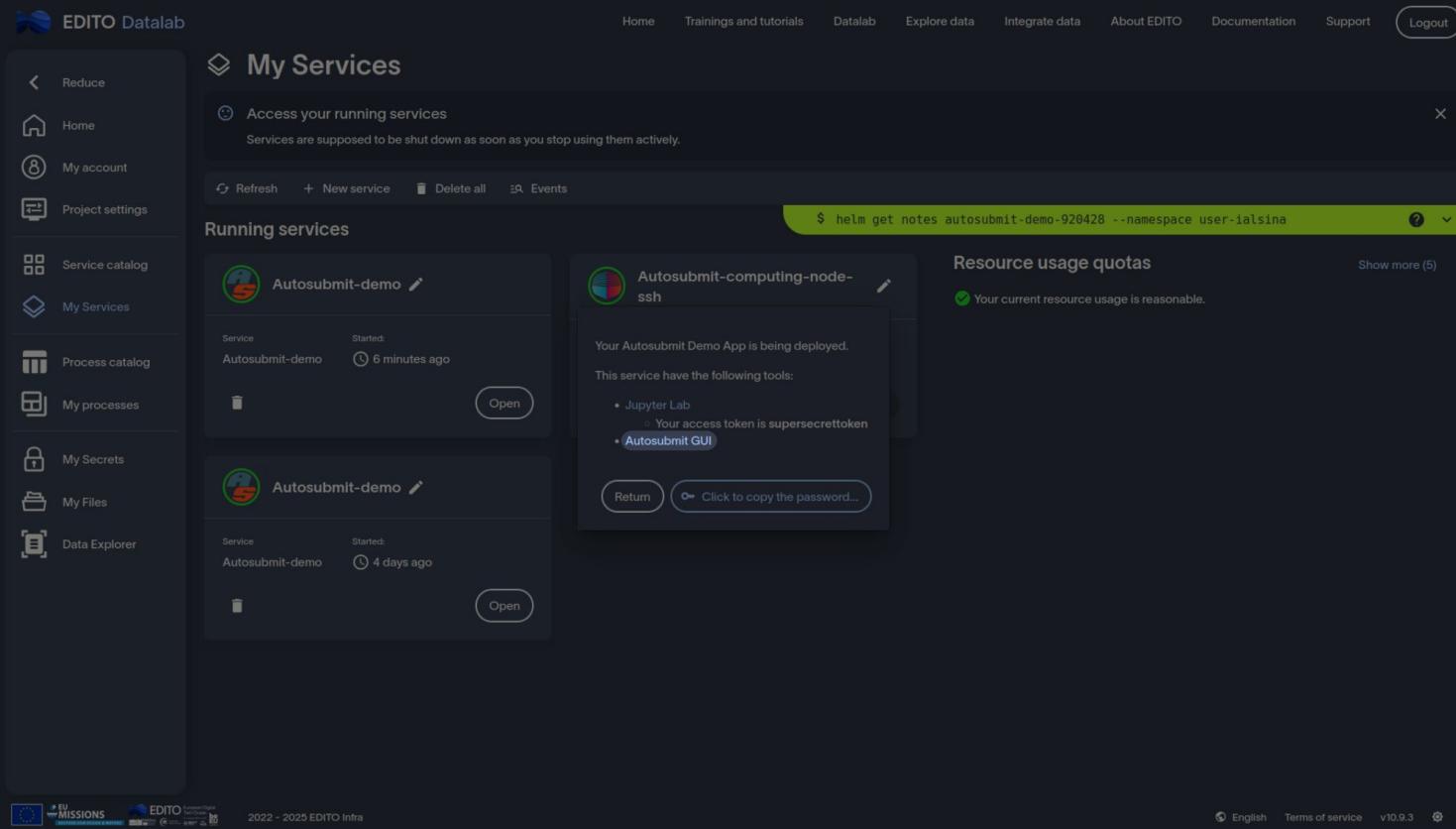
The screenshot shows the 'My Services' page in the EDITO Datalab interface. On the left, a sidebar lists various navigation options: Reduce, Home, My account, Project settings, Service catalog, My Services (which is selected and highlighted in blue), Process catalog, My processes, My Secrets, My Files, and Data Explorer. The main content area is titled 'My Services' and contains a section for 'Access your running services'. It shows a table of 'Running services' with one entry: 'Autosubmit-computing-node-ssh'. The service details show it was started 3 minutes ago and is currently 'READY'. To the right of the service table, a large box displays the configuration for the 'Autosubmit-computing-node-ssh' service. The configuration includes:

```
node-ssh-100232
Once connected to the computing node, create the scratch directory that you will use in your experiment (e.g. mkdir -p /home/onyxia/work/test/linuxserver.io)
Then, from your "Jupyterlab-autosubmit-bsc", create a new experiment (e.g.autosubmit expid -dm -H "COMPUTING_NODE" -d "Dummy test")
Configure the 'platforms.yml' file with the following informations:
COMPUTING_NODE:
  TYPE: ps
  HOST: autosubmit-computing-node-ssh-100232
  PROJECT: test
  USER: linuxserver.io
  QUEUE: debug
  SCRATCH_DIR: /home/onyxia/work
  ADD_PROJECT_TO_HOST: false
  MAX_WALLCLOCK: 48:00
  TEMP_DIR: ''
```

Below the configuration, a green box indicates: 'Your current resource usage is reasonable.'

Step 10. Follow the Jupyter Notebooks, changing the remote user, EDITO user and hostname where needed.

Step 11. Go back to the Autosubmit-demo service, and open the Autosubmit GUI.



The screenshot shows the EDITO Datalab interface with the 'My Services' section selected. The 'Running services' table lists two entries for 'Autosubmit-demo'. The first entry, 'Autosubmit-demo' (started 6 minutes ago), has an 'Open' button. The second entry, 'Autosubmit-demo' (started 4 days ago), also has an 'Open' button. To the right of the table, a modal window for the 'Autosubmit-computing-node-ssh' service is open. It displays deployment notes: '\$ helm get notes autosubmit-demo-920428 --namespace user-ialsina'. The modal also lists tools: 'Jupyter Lab' (with a note about the access token) and 'Autosubmit GUI'. A 'Resource usage quotas' section indicates that usage is reasonable. The bottom right of the modal has a 'Click to copy the password...' button. The left sidebar contains links for 'Reduce', 'Home', 'My account', 'Project settings', 'Service catalog', 'My Services' (selected), 'Process catalog', 'My processes', 'My Secrets', 'My Files', and 'Data Explorer'. The top navigation bar includes 'Home', 'Trainings and tutorials', 'Datalab', 'Explore data', 'Integrate data', 'About EDITO', 'Documentation', 'Support', and a 'Logout' button.

My Services

Access your running services

Services are supposed to be shut down as soon as you stop using them actively.

Refresh + New service Delete all Events

Running services

 Autosubmit-demo	Service: Autosubmit-demo	Started: 6 minutes ago	Open
 Autosubmit-demo	Service: Autosubmit-demo	Started: 4 days ago	Open

Autosubmit-computing-node-ssh

Your Autosubmit Demo App is being deployed.

This service have the following tools:

- Jupyter Lab
 - Your access token is `supersecrettoken`
- Autosubmit GUI

Return [Click to copy the password...](#)

Resource usage quotas

Show more (5)

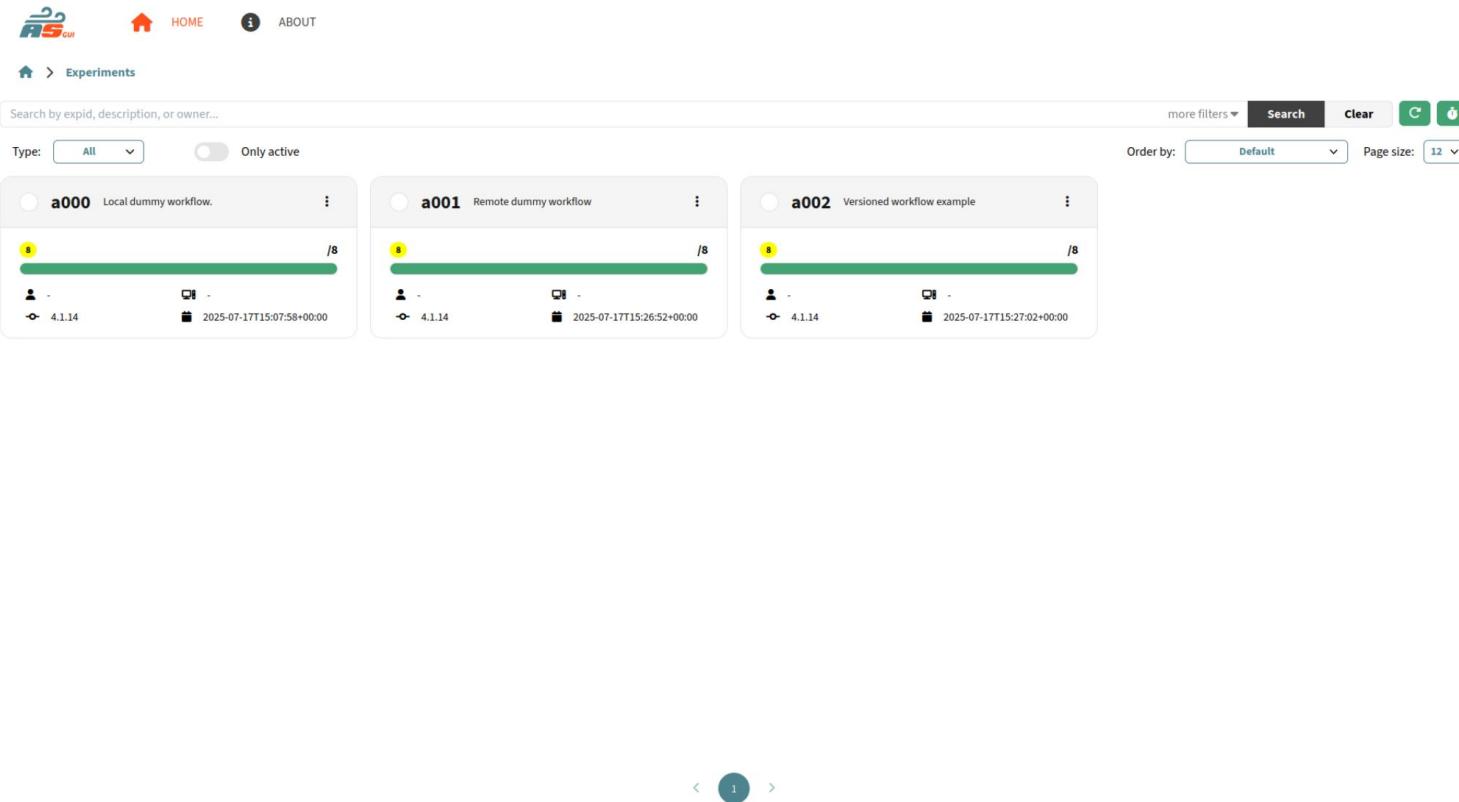
Your current resource usage is reasonable.

EU MISSIONS EDITO European Data Infra-structure for the Environment

2022 - 2025 EDITO Infra

English Terms of service v10.9.3

Step 12. Explore your active experiments. Notice the “Only active” switch.



The screenshot shows the ASwF (Apache Airflow) web interface for managing experiments. The top navigation bar includes links for HOME and ABOUT. Below the navigation is a breadcrumb trail: Experiments. A search bar and a "more filters" dropdown are available. A "Type" dropdown is set to "All", and a "Only active" switch is turned on. The search bar contains the placeholder "Search by expid, description, or owner...". On the right, there are buttons for "Search", "Clear", and "CSV/JSON". The main content area displays three experiment cards:

- a000 Local dummy workflow**: Status: 8/8. Last updated: 2025-07-17T15:07:58+00:00. Version: 4.1.14.
- a001 Remote dummy workflow**: Status: 8/8. Last updated: 2025-07-17T15:26:52+00:00. Version: 4.1.14.
- a002 Versioned workflow example**: Status: 8/8. Last updated: 2025-07-17T15:27:02+00:00. Version: 4.1.14.

Below the cards, there is a page navigation bar with a single page indicator.

Useful links

- Autosubmit (AS)
 - Pip install and docs: <https://autosubmit.readthedocs.io/en/master/installation/index.html>
 - Repository: <https://github.com/BSC-ES/autosubmit-gui>
 - Contact: support-autosubmit@bsc.es
- Autosubmit GUI
 - Public mockup: <https://earth.bsc.es/gitlab/wuruchi/autosubmitreact>
 - Demo: <https://autosubmitgui.bsc.es/presentation/>