

# 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!

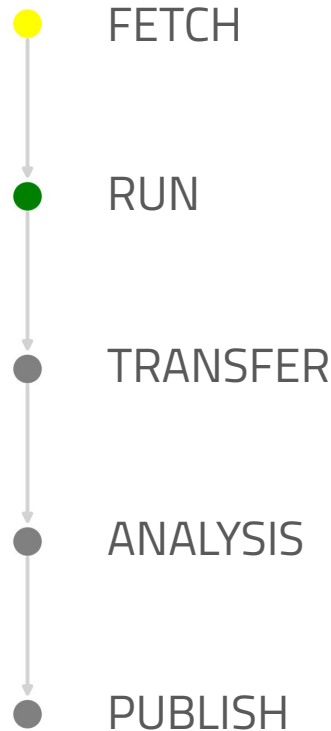
When solving big computational problems, it is common to think of your top-level **main function** as a **workflow** that is run through a **workflow manager**.

```
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, 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

## 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...](#)

&lt; 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 <sup>1</sup>

### 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](#)

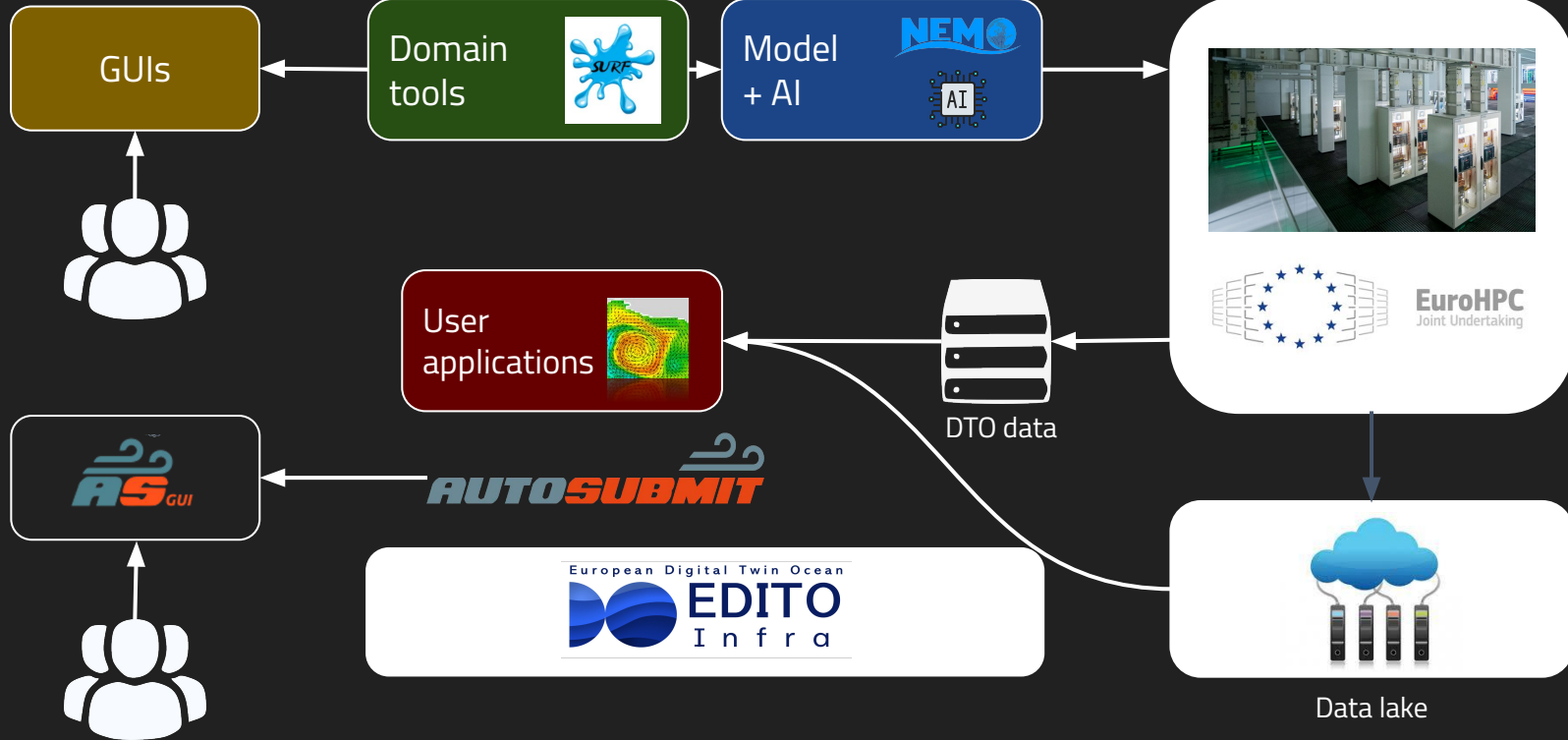
Return

Click to copy the password...

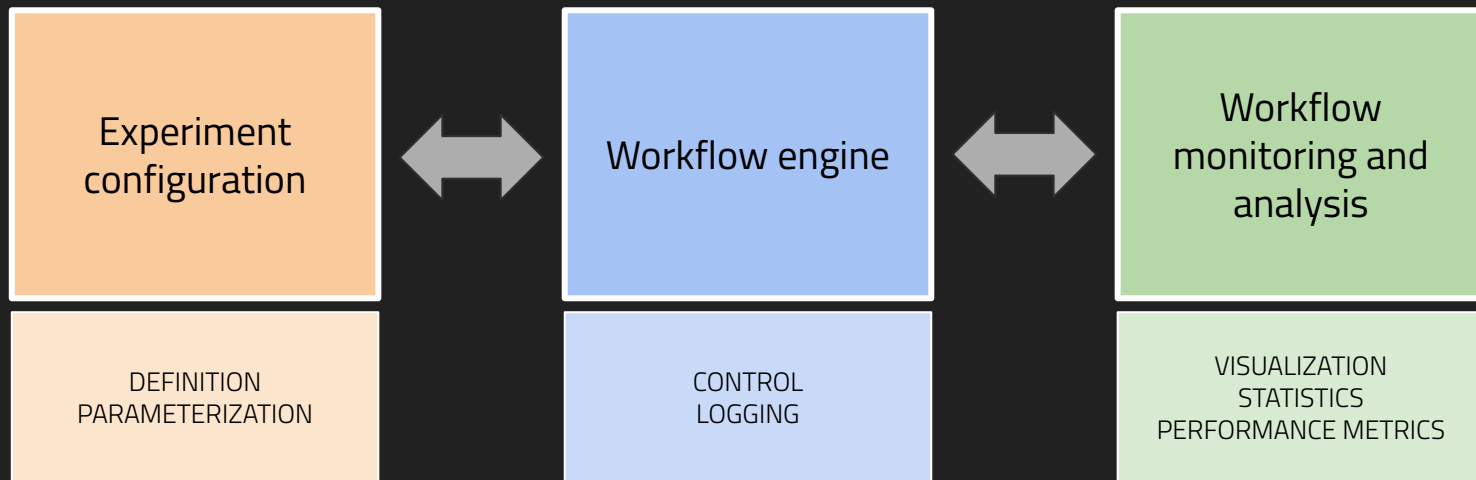
**Jupyter Lab:** service with a notebook with Autosubmit instance running, and access to HPC

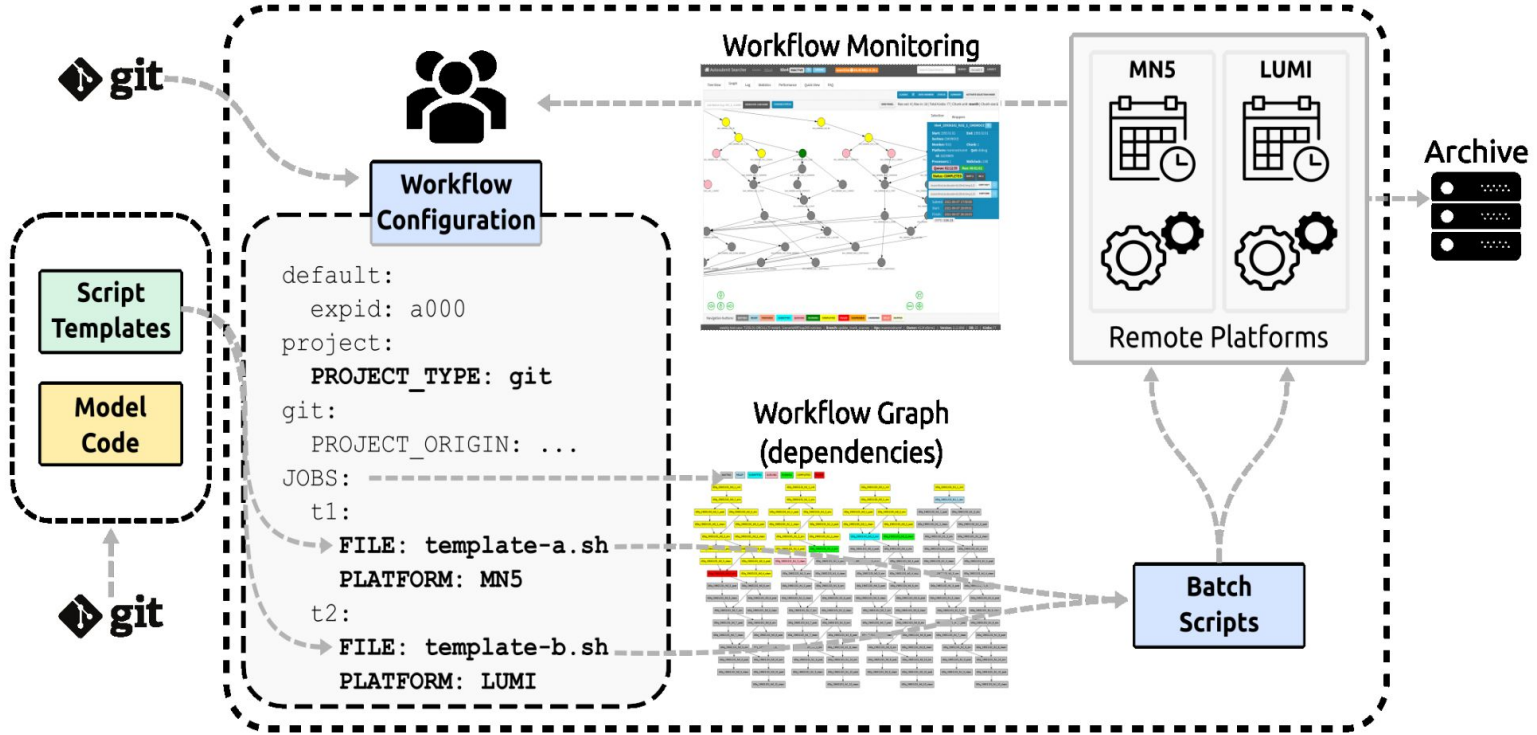
**Autosubmit GUI:** web app to monitor Autosubmit experiments

**AUTOSUBMIT**



Autosubmit







**AUTO***SUBMIT*

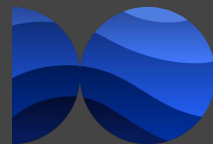


**CMIP** Coupled Model  
Intercomparison  
Project

**WCRP**



**Destination Earth**



**EDITO**



# Declarative

YAML

Validation

Inheritance

Imports



## Declarative

YAML

Validation

Inheritance

Imports

## Consistent

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

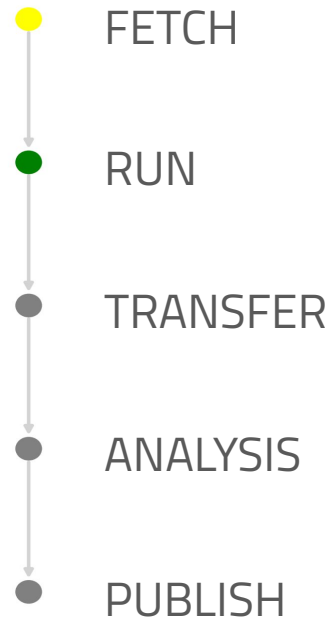
# Configuration

# Workflow Dependencies

jobs.yml

```
JOB:
  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

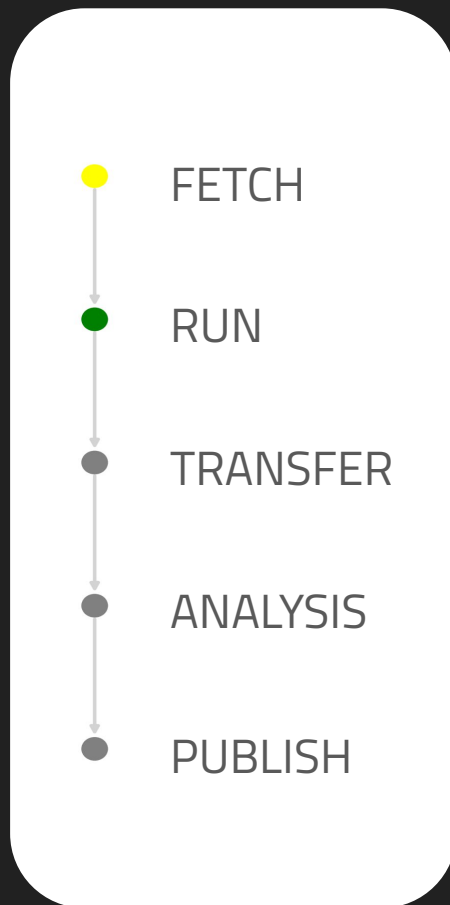


# Workflow Dependencies

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

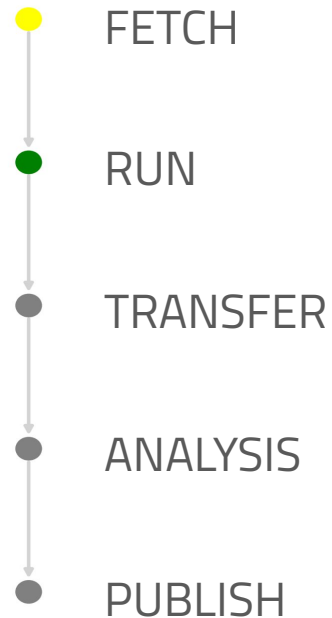


# Workflow Platforms

~/platforms.yml

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

YA  
ML





# Workflow Platforms

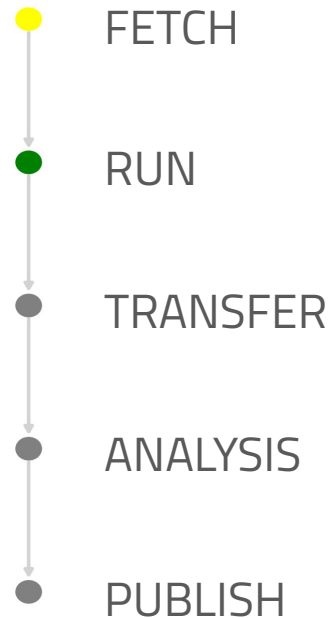
~/platforms.yml

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

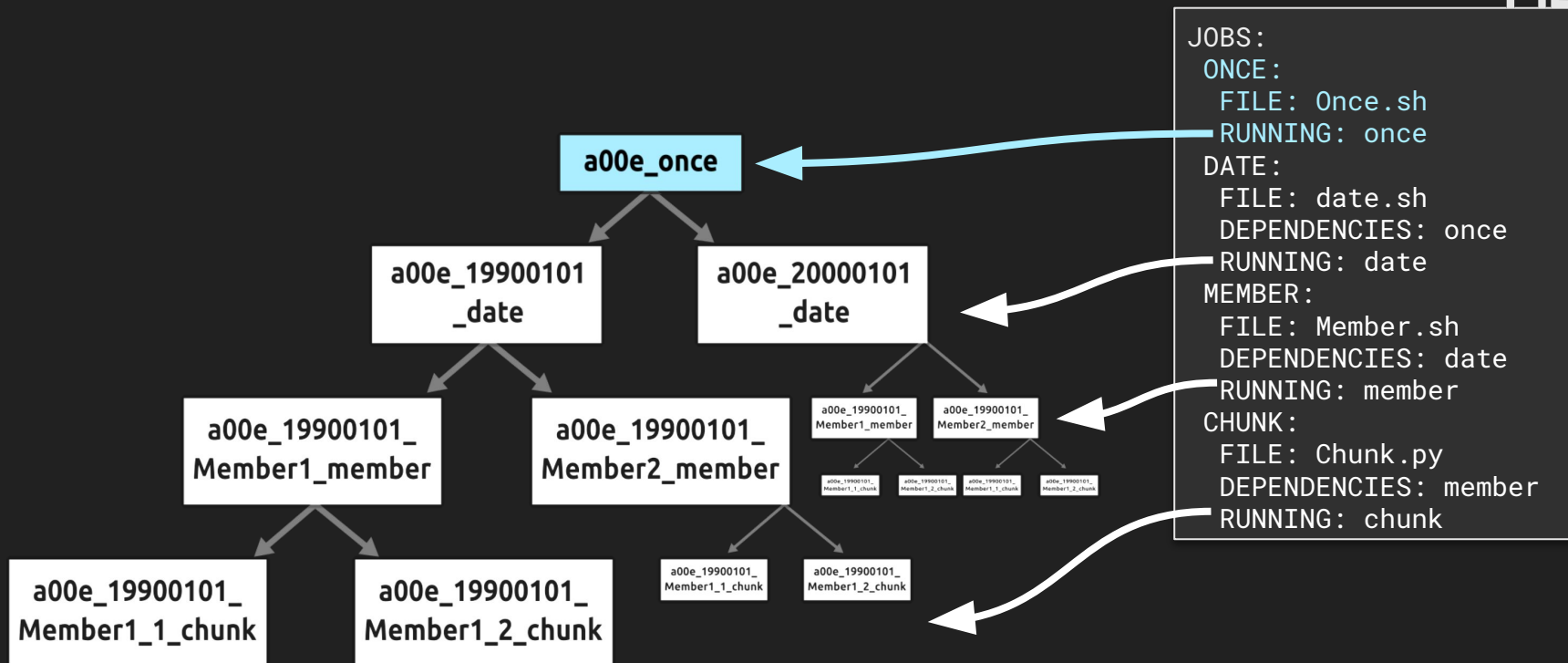
YAML

~/ssh/config

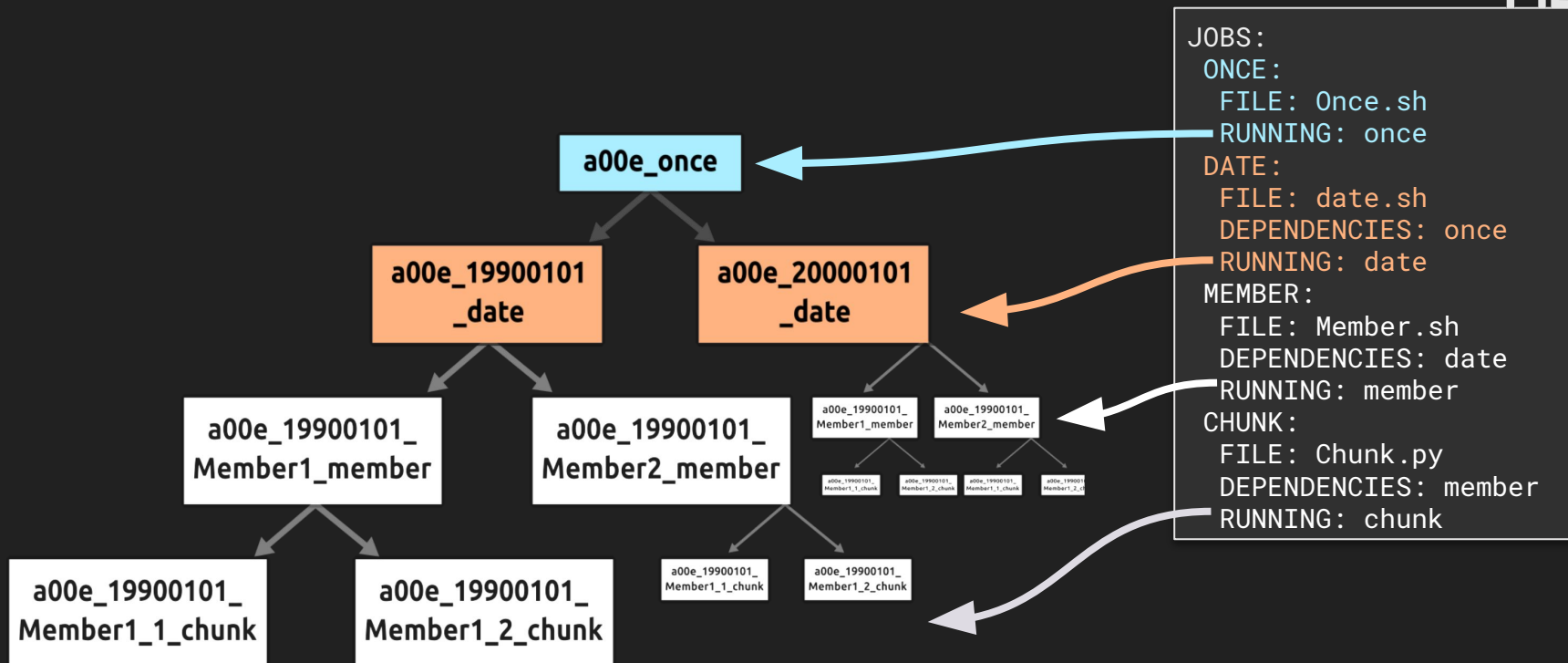
```
PLATFORMS:  
Host mn5login  
  HostName glogin1.bsc.es  
  User my-remote-user  
  IdentityFile ~/.ssh/id_rsa  
  ForwardX11 yes
```



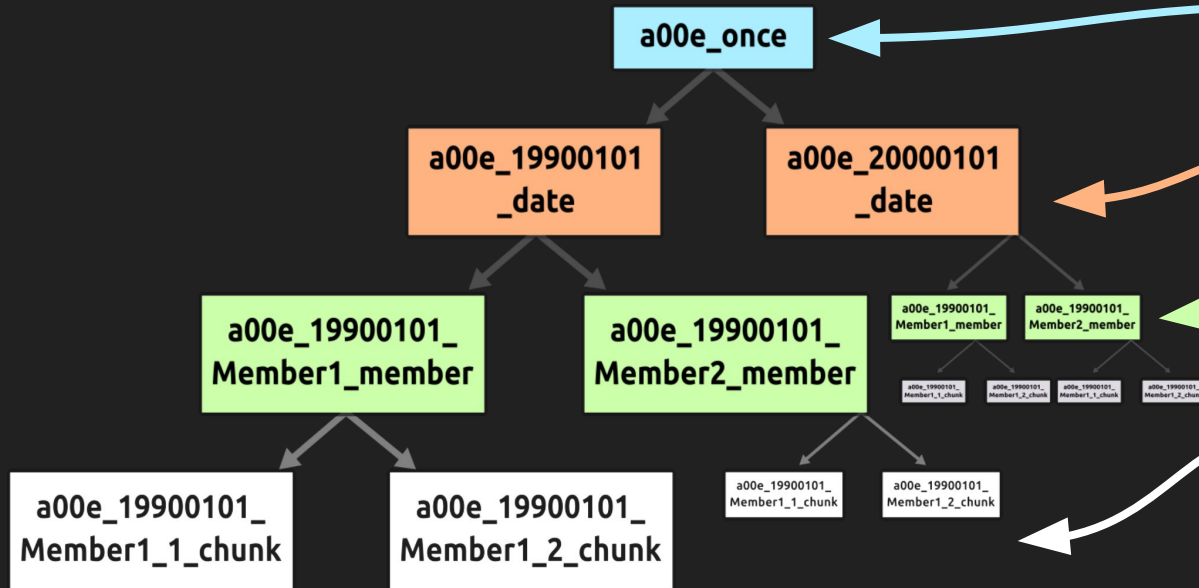
# Workflow Hierarchy



# Workflow Hierarchy

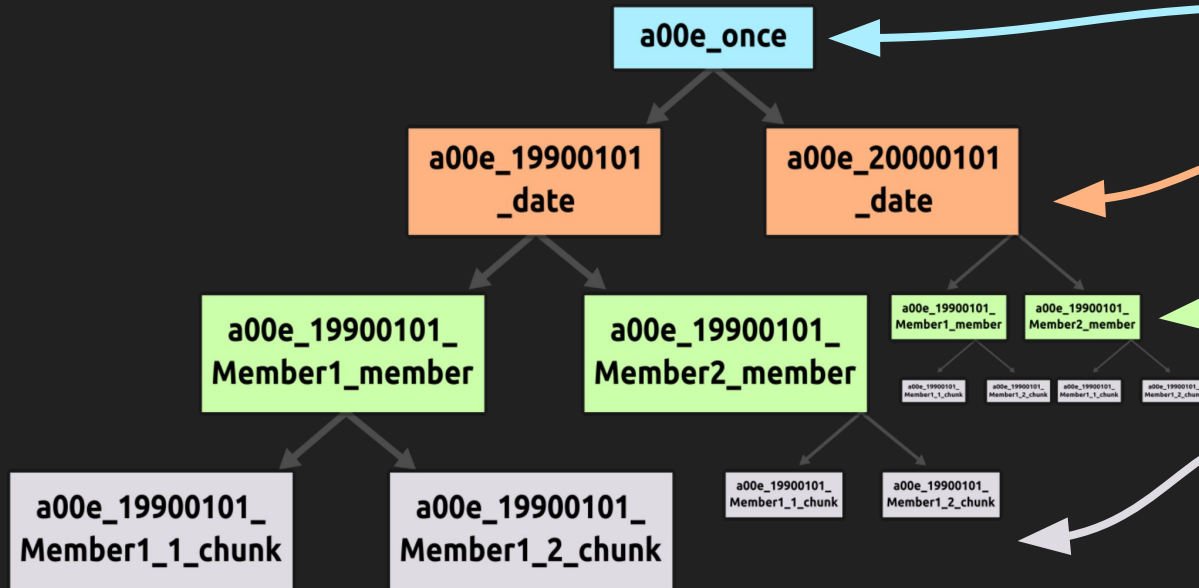


# Workflow Hierarchy



**JOBS:**  
**ONCE:**  
FILE: Once.sh  
RUNNING: once  
**DATE:**  
FILE: date.sh  
DEPENDENCIES: once  
RUNNING: date  
**MEMBER:**  
FILE: Member.sh  
DEPENDENCIES: date  
RUNNING: member  
**CHUNK:**  
FILE: Chunk.py  
DEPENDENCIES: member  
RUNNING: chunk

# Workflow Hierarchy



**JOBS:**  
**ONCE:**  
FILE: Once.sh  
RUNNING: once  
**DATE:**  
FILE: date.sh  
DEPENDENCIES: once  
RUNNING: date  
**MEMBER:**  
FILE: Member.sh  
DEPENDENCIES: date  
RUNNING: member  
**CHUNK:**  
FILE: Chunk.py  
DEPENDENCIES: member  
RUNNING: chunk

Running



```
graph LR; A[CREATE] --> B[CONFIGURE]; B --> C[PREPARE]; C --> D[RUN]; D --> E[MONITOR];
```

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/**

	<b>conf/</b>	Experiment configuration
	<b>proj/</b>	Project (workflow scripts, config., ...)
	<b>plot/</b>	Visualizations
	<b>tmp/</b>	Logs, templates
	<b>pk1/</b>	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.

	<b>conf/</b>	Experiment configuration
	<b>proj/</b>	Project (workflow scripts, config., ...)
	<b>plot/</b>	Visualizations
	<b>tmp/</b>	Logs, templates
	<b>pk1/</b>	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  
...

<b>conf/</b>	Experiment configuration
<b>proj/</b>	Project (workflow scripts, config., ...)
<b>plot/</b>	Visualizations
<b>tmp/</b>	Logs, templates
<b>pk1/</b>	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

/app1/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



/app1/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

/app1/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"          # 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..."
```

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.

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

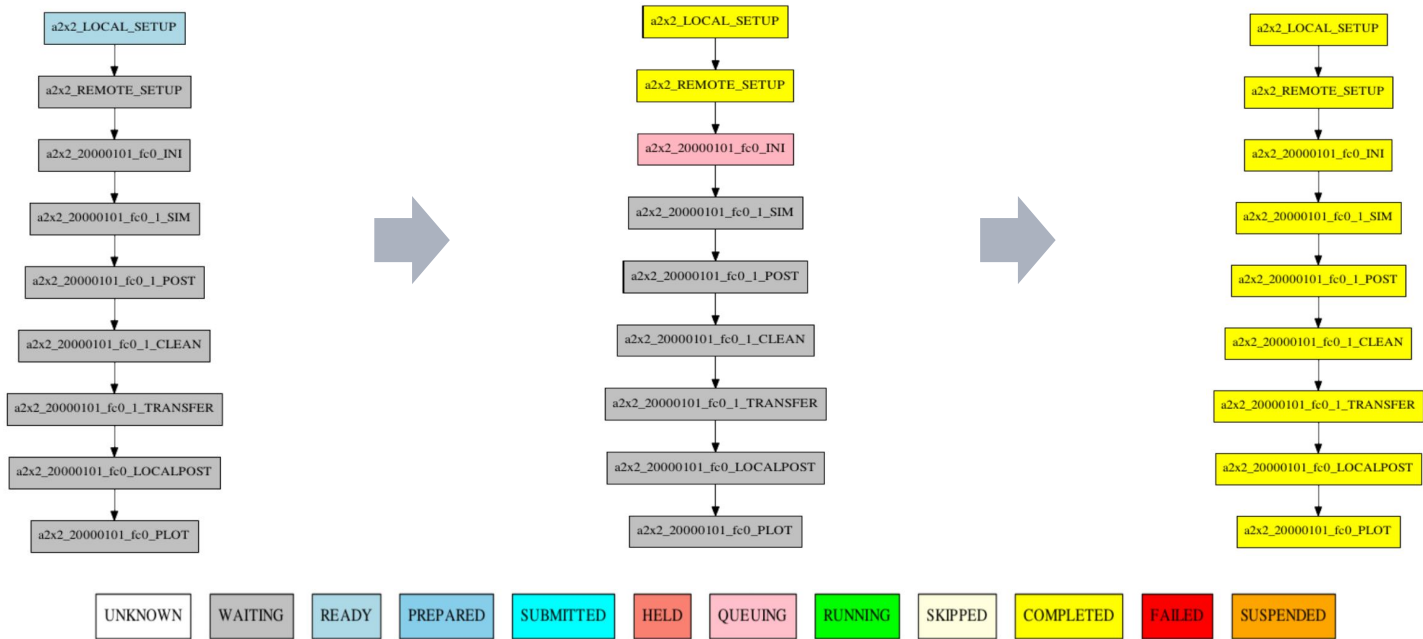
CONFIGURE

PREPARE

RUN

MONITOR

autosubmit monitor a01a



CREATE

CONFIGURE

PREPARE

RUN

MONITOR

# Monitoring



# Autosubmit GUI

[HOME](#)[ABOUT](#)[Experiments](#)[more filters ▼](#)[Search](#)[Clear](#)

Type:

[All](#) ▼

Only active

Order by:

[Default](#) ▼

Page size:

[12](#) ▼**a000**

Auto Schism2 demo



/2



onyxia



MARENOSTRUM5



4.1.10



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 the Autosubmit logo, a home icon, and links for HOME and ABOUT. Below this is a breadcrumb trail: Home > Experiment a000 > Quick View. The main interface is divided into a left sidebar and a main content area. The sidebar contains a list of view options: QUICK VIEW (selected), TREE VIEW, GRAPH VIEW, TABLE VIEW, RUN LOG, CONFIGURATION, STATISTICS, and PERFORMANCE. The main content area has a top bar with system information: Auto Schism2 demo, user onyxia, host MARENOSTRUM5, version Autosubmit v4.1.10, and a main menu icon. Below this is a summary bar showing 'TOTAL (2)' jobs, a filter input field, and a status 'Showing 2 of 2 total Jobs' with a refresh button. The job list below shows two entries: 'a000\_SIM' with status '#READY' and 'a000\_CLEAN' with status '#WAITING'.

AS GUI

HOME ABOUT

Home > Experiment a000 > Quick View

Auto Schism2 demo onyxia MARENOSTRUM5 Autosubmit v4.1.10 main

TOTAL (2) Filter job... Showing 2 of 2 total Jobs

- a000\_SIM #READY
- a000\_CLEAN #WAITING

# Autosubmit GUI

## TREE VIEW

The screenshot displays the Autosubmit GUI interface in the Tree View mode. At the top, there is a navigation bar with the Autosubmit logo, a home icon, and links for HOME and ABOUT. Below this is a breadcrumb trail: Home > Experiment a000 > Tree View.

The left sidebar contains a list of view options: QUICK VIEW, TREE VIEW (which is highlighted in red), 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 at the top features a 'Run: Latest' button, a search bar labeled 'Filter job...', and buttons for 'Filter', 'Clear', 'START MONITOR', and a refresh icon. Below these buttons, it shows 'Total #Jobs: 2 | Chunk unit: month | Chunk size: 4' and three buttons: 'Default Expand', 'Expand All +', and 'Collapse All -'.

The main content area displays a tree structure under the heading 'Keys'. It contains two entries:

- a000\_SIM with status #READY and SOURCE
- a000\_CLEAN with status #WAITING and TARGET

# Autosubmit GUI

## GRAPH VIEW

The screenshot displays the Autosubmit GUI interface. At the top, there is a navigation bar with a logo, a home icon, and links for HOME and ABOUT. Below this is a breadcrumb trail: Home > Experiment a000 > Graph View. The main interface is divided into a left sidebar and a main content area. The sidebar contains a list of view options: QUICK VIEW, TREE VIEW, GRAPH VIEW (highlighted in red), TABLE VIEW, RUN LOG, CONFIGURATION, STATISTICS, PERFORMANCE, and a double arrow icon. The main content area features a search bar labeled 'Filter job...' with 'Filter' and 'Clear' buttons. To the right of the search bar are two green buttons: 'START MONITOR' and a circular refresh icon. Below the search bar, it shows 'Total #Jobs: 2 | Chunk unit: month | Chunk size: 4' and a 'Select by status:' dropdown menu set to 'COMPLETED' with a green checkmark icon. The central part of the main content area displays a graph with two nodes: a light blue circle at the top labeled 'a000\_SIM' and a grey circle at the bottom labeled 'a000\_CLEAN', connected by a downward-pointing arrow. In the bottom right corner of the graph area, a small grey box contains the text: 'Box selection: Shift + LClick' and 'Add/remove select: Ctrl + LClick'.

AS GUI

HOME ABOUT

Home > Experiment a000 > Graph View


Filter job... Filter Clear START MONITOR ↻

Total #Jobs: 2 | Chunk unit: month | Chunk size: 4 Select by status: COMPLETED ✓

Box selection: Shift + LClick  
Add/remove select: Ctrl + LClick

# Autosubmit GUI

## CONFIGURATION

[HOME](#)[ABOUT](#)

[Home](#) > [Experiment a000](#) > [Configuration](#)

☰

🗂️

🔧

📊

>\_

⚙️

📈

🕒

»

**CURRENT RUN CONFIGURATION (HISTORICAL DATABASE)**

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

**[CONFIG]**

Setting	Value
AUTOSUBMIT_VERSION	4.1.10
MAXWAITINGJOBS	20
TOTALJOBS	20

**[DEFAULT]**

Setting	Value
EXPID	a000
HPCARCH	MARENOSTRUM5

**CURRENT FILESYSTEM CONFIGURATION**

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

**[CONFIG]**

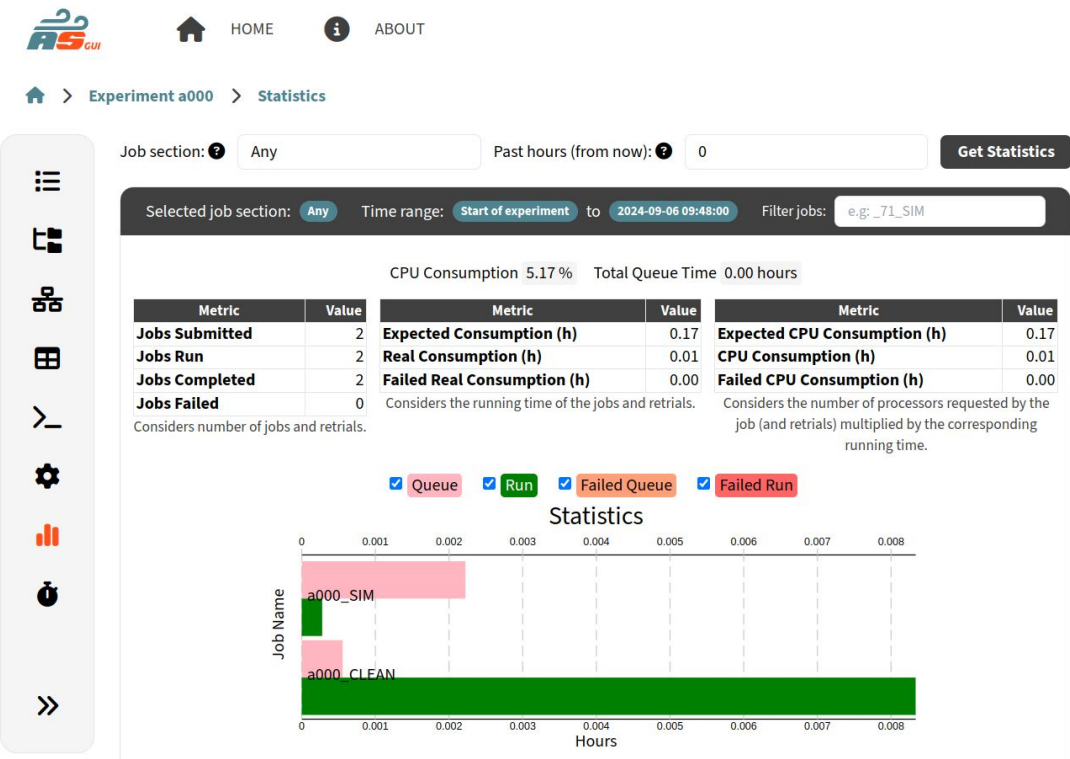
Setting	Value
AUTOSUBMIT_VERSION	4.1.10
MAXWAITINGJOBS	20
TOTALJOBS	20

**[DEFAULT]**

Setting	Value
EXPID	a000
HPCARCH	MARENOSTRUM5

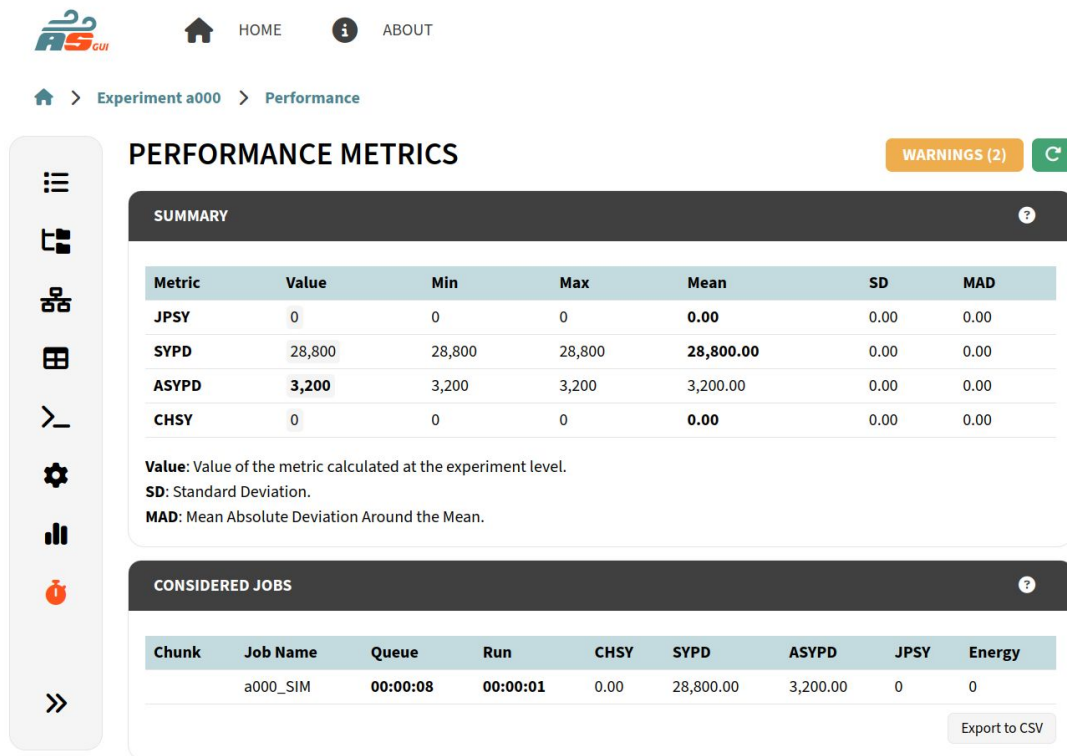
# Autosubmit GUI

## STATISTICS



# Autosubmit GUI

## PERFORMANCE



# 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": "marenostrum5",
  "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

POST	/v3/login	Login	post_v3_login	▼
GET	/v3/tokentest	Token test	get_v3_tokentest	▼
POST	/v3/tokentest	Token test	post_v3_tokentest	▼
GET	/v3/updatedesc	Update description	get_v3_updatedesc	▼
POST	/v3/updatedesc	Update description	post_v3_updatedesc	▼
GET	/v3/cconfig/{expid}	Get current configuration	get_v3_cconfig_exp_id_	▼
GET	/v3/expinfo/{expid}	Get experiment info	get_v3_expinfo_exp_id_	▼
GET	/v3/expcount/{expid}	Get experiment counters	get_v3_expcount_exp_id_	▼
GET	/v3/searchowner/{owner}/{exptype}/{onlyactive}	Search owner	get_v3_searchowner_owner_exptype_onlyactive_	▼
GET	/v3/searchowner/{owner}	Search owner	get_v3_searchowner_owner_	▼
GET	/v3/search/{expid}/{exptype}/{onlyactive}	Search experiment	get_v3_search_exp_id_exptype_onlyactive_	▼
GET	/v3/search/{expid}	Search experiment	get_v3_search_exp_id_	▼
GET	/v3/running	Running experiments	get-v3-running	▼
GET	/v3/runs/{expid}	Get runs	get_v3_runs_exp_id_	▼

Hands On

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

The screenshot displays the EDITO Datalab Service Catalog interface. The top navigation bar includes links for Home, Trainings and tutorials, Datalab, Explore data, Integrate data, About EDITO, Documentation, Support, and a Logout button. A left sidebar contains navigation options: Reduce, Home, My account, Project settings, Service catalog (highlighted), My Services, Process catalog, My processes, My Secrets, My Files, and Data Explorer. The main content area is titled 'Service catalog' and features a search bar and a filter menu with categories: All, Ocean modelling, Ocean data quality, Data visualization, What-If applications, Focus applications, IDE, Databases, Automation, and Playground. A grid of service cards is shown, with the 'Autosubmit-demo' card highlighted in blue. This card is categorized under 'Ocean modelling' and describes a demo of Autosubmit (with JupyterLab), API, and GUI, with a 'Launch' button. Other visible cards include 'Jupyterlab-autosubmit-bsc', 'Autosubmit-computing-node-ssh', 'Delft3dfm-modelbuilder', 'Jupyter-python-ocean-science', 'Turbiditymapping-4dvarnet', 'Surf-nemo', 'Nemo-demo', 'Stac-browser', 'D-ecoincact-postprocess', 'Static-pages', and 'Macroplastics-mediterranean-viewer'.

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

**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

**Stac-browser**  
Data visualization  
Launch a STAC Browser server.  
Launch

**D-ecoincact-postprocess**  
Data visualization  
Example notebook to load D-ecoincact output  
Launch

**Static-pages**  
Data visualization  
Serve a FastAPI static website defined in a git  
Launch

**Macroplastics-mediterranean-viewer**  
What-If applications  
Launch

EUROPEAN COMMISSIONS EDITO 2022 - 2025 EDITO India English Terms of service v10.9.3

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

The screenshot displays the EDITO Datalab web interface. On the left is a dark sidebar with navigation icons and labels: 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 has a top navigation bar with links: Home, Trainings and tutorials, Datalab, Explore data, Integrate data, About EDITO, Documentation, Support, and a Logout button. Below this is a green terminal-style bar showing the command: `$ helm install autosubmit-demo-928428 ocean-modelling/autosubmit-demo -f values.yaml`. The central panel is titled 'Autosubmit-demo' and includes a description: 'The Helm chart autosubmit-demo belongs to the Helm chart repository Ocean modelling.' It features a 'Friendly name' field with 'autosubmit-demo' and a 'Version' dropdown menu set to '0.1.36'. 'Cancel' and 'Launch' buttons are positioned to the right. Below this, there are two tabs: 'Form' (selected) and 'Text Editor'. The 'Form' tab contains a list of configuration sections, each with a title, a brief description, and a dropdown arrow: 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), and S3 Configuration (Configuration of temporary identity for AWS S3).

EDITO Datalab

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

`$ helm install autosubmit-demo-928428 ocean-modelling/autosubmit-demo -f values.yaml`

**Autosubmit-demo**

The Helm chart autosubmit-demo belongs to the Helm chart repository Ocean modelling.

Friendly name: autosubmit-demo Version: 0.1.36

Cancel Launch

☒ Form ☐ Text Editor

- 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

MISSIONS EDITO 2022 - 2025 EDITO Italia English Terms of service v10.9.3

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

The screenshot displays the EDITO Datalab web interface. At the top, the navigation bar includes the EDITO Datalab logo, a project dropdown menu set to 'ltenorio personal project', and links for Home, Trainings and tutorials, Datalab, Explore data, Integrate data, About EDITO, Documentation, Support, and a Logout button.

The left sidebar contains a navigation menu with the following items: Reduce, Home, My account, Project settings, Service catalog, My services (highlighted), Process catalog, My processes, My secrets, and My files.

The main content area is titled 'My Services' and features a status message: 'Access your running services. Services are supposed to be shut down.' Below this, there are buttons for Refresh, New service, and Delete. A section titled 'Running services' lists the 'Autosubmit-demo' service, which is in a 'Container starting' state. An 'Open' button is visible next to the service name.

A modal dialog is overlaid on the 'Autosubmit-demo' service, indicating deployment progress. The modal text reads: 'Your Autosubmit Demo App is being deployed. This service have the following tools: Jupyter Lab (Your access token is supersecrettoken), Autosubmit GUI. Pulling image "autosubmit/gui:edito-demo".' A progress bar is shown at the bottom of the modal, and a 'Return' button is located at the bottom right.

Below the modal, the 'Resource usage quotas' section is visible, showing a green checkmark and the message: 'Your current resource usage is reasonable.' A 'Show more (5)' link is also present.

The footer of the page includes the EU MISSIONS logo, the EDITO logo, the text '2022 - 2024 EDITO Infra', and links for English, Terms of service, v8.27.0, and a settings icon.

## My Services

😊 Access your running services

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

↻ Refresh

+ New service

🗑 Del

### Running services



Autosubmit-demo ✎

Service

Started:

Autosubmit-demo

🕒 a minute ago



Open



Open

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...

**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 screenshot displays the EDITO Datalab web interface. The top navigation bar includes links for Home, Trainings and tutorials, Datalab, Explore data, Integrate data, About EDITO, Documentation, Support, and a Logout button. The left sidebar contains a menu with options: Reduce, Home, My account, Project settings, Service catalog, My Services (selected), 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. Two services are visible: "Autosubmit-demo" (started 6 minutes ago) and "Autosubmit-computing-node-ssh" (started 4 days ago). A terminal window is open, showing the command `$ helm get notes autosubmit-demo-920428 --` and its output, which includes the URL `http://localhost:8888/?token=chda5Eda...` and the instruction to use the token in the Jupyter Lab password field.

Overlaid on the terminal is a Jupyter Lab login dialog. It states "Token authentication is enabled" and provides instructions on how to use the token. It includes a "Password or token:" input field and a "Log in" button. Below this, it shows the "Currently running servers:" section with the same URL as the terminal output. At the bottom, there is a "Setup a Password" section with input fields for "Token" and "New Password", and a "Log in and set new password" button.

The footer of the page includes the European Union flag, the text "EU MISSIONS", the EDITO logo, and the copyright notice "2022 - 2025 EDITO Infra". On the right side of the footer, there are links for "English", "Terms of service", and the version "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*


**EDITO Platform** Home **Trainings and tutorials** Datalab Explore data Integrate data About EDITO Documentation Support English

### Trainings and tutorials

Q SEARCH

#### Ocean modelling

BSC and EDITO




#### Discover the Virtual Ocean Model Lab

Access tutorial resources for using Virtual Ocean Model Lab components

Create Explore

Open

EDITO ModelLab and 2 others




#### Discover GLONET products

Explore 1/4 degree resolution forecast data generated with Mercator Ocean international's GLONET AI model developed as part of the EDITO-ModelLab project.

Explore

Read

EDITO ModelLab




#### Ocean data analysis and modelling tutorials

You will find there several examples of how to use EDITO Platform to run data analysis and modelling tools

Create

Open

EDITO-Model Lab and 11 others




#### EDITO Core Model Suite

Discover the EDITO Core Model Suite.

Create Explore

Ward Standaert and Flanders Marine Institute




#### Using EditoTools for predictive modelling of Atlantic herring larvae in the North Sea

Tutorial on species distribution modelling using RStudio IDE in the Datalab

Create

Flanders Marine Institute



#### EDITO-Blue-Cloud2026 Demonstrator Nutrient-Phytoplankton-Zooplankton-Detritus model

Tutorial on biogeochemical modelling using RStudio IDE in the EDITO Datalab. An Rmarkdown guiding the user step-by-step through a Nutrient-Phytoplankton-Zooplankton-Detritus model

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

The screenshot shows the GitLab web interface for the repository 'autosubmit-edito-training'. The left sidebar contains a 'Project' menu with options: Manage, Plan, Code, Build, Deploy, Operate, Monitor, and Analyze. The main content area displays the repository details, including a commit history table and a README file. A modal window titled 'Download source code' is open, showing options to download the code in various formats: zip, tar.gz, tar.bz2, and tar. The modal also includes options to clone the repository with SSH or HTTPS, and to open it in your IDE (Visual Studio Code, IntelliJ IDEA).

Public / EDITO-Model-Lab / WP5-VOML / autosubmit-edito-training

Project

- autosubmit-edito-training
- Manage
- Plan
- Code
- Build
- Deploy
- Operate
- Monitor
- Analyze

autosubmit-edito-training

main autosubmit-edito-training

Squashed initial commit after moving the code to MOI repository (no dbeltran authored 1 year ago)

Name	Last commit
Videos	Squashed initial commit after moving
AutoSchism2.ipynb	Squashed initial commit after moving
LocalDummy.ipynb	Squashed initial commit after moving
Monitor.ipynb	Squashed initial commit after moving
README.md	Squashed initial commit after moving
RemoteDummy-fakenode.ipynb	Squashed initial commit after moving
RemoteDummy.ipynb	Squashed initial commit after moving
RemoteTemplates-fakenode.ipynb	Squashed initial commit after moving
RemoteTemplates.ipynb	Squashed initial commit after moving

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>.

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

Project information

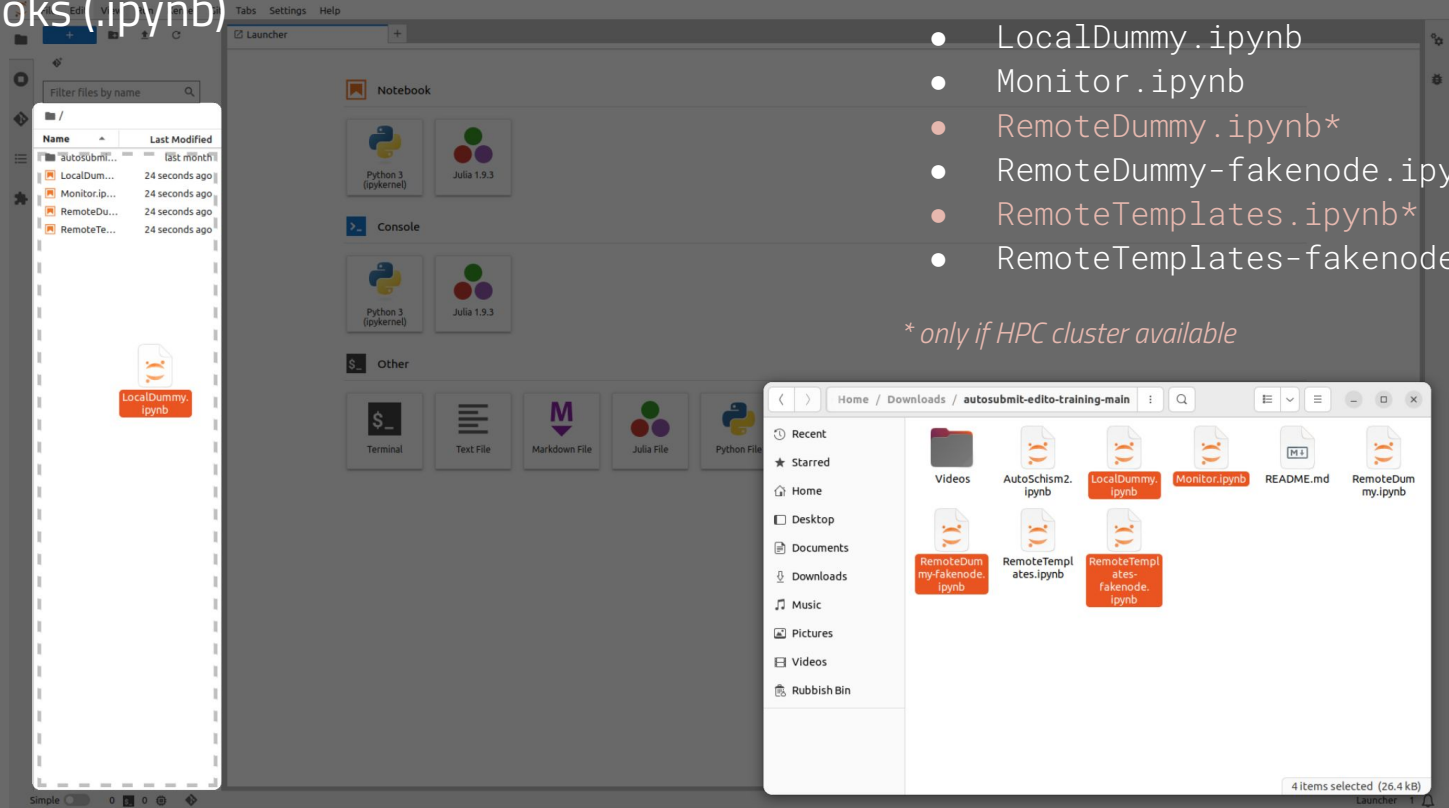
- 1 Commit
- 1 Branch
- 0 Tags
- README

Created on

October 24, 2024

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*

The screenshot displays the EDITO Datalab Service Catalog interface. The top navigation bar includes links for Home, Trainings and tutorials, Datalab, Explore data, Integrate data, About EDITO, Documentation, Support, and a Logout button. A left sidebar contains navigation options: Reduce, Home, My account, Project settings, Service catalog (selected), My Services, Process catalog, My processes, My Secrets, My Files, and Data Explorer. The main content area is titled 'Service catalog' and features a search bar and a filter menu with categories: All, Ocean modelling, Ocean data quality, Data visualization, What-If applications, Focus applications, IDE, Databases, Automation, and Playground. A grid of service cards is shown, with 'Autosubmit-computing-node-ssh' highlighted. This service is categorized under 'Ocean modelling' and is described as an 'An autosubmit computing node service (openssh-server) to run alongside Jupyterlab-autosubmit-bsc'. It includes a 'Launch' button. Other visible services include 'Jupyterlab-autosubmit-bsc', 'Delft3dfm-modelbuilder', 'Jupyter-python-ocean-science', 'Autosubmit-demo', 'Turbiditymapping-4dvarnet', 'Surf-nemo', 'Nemo-demo', 'Stac-browser', 'D-ecoincact-postprocess', 'Static-pages', and 'Macroplastics-mediterranean-viewer'. The footer contains logos for the European Union, the French Republic, and the French Navy, along with the text '2022 - 2025 EDITO India' and a language selector set to 'English'.

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

**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

**Stac-browser**  
Data visualization

Launch a STAC Browser server.

**D-ecoincact-postprocess**  
Data visualization

Example notebook to load D-ecoincact output

**Static-pages**  
Data visualization

Serve a FastAPI static website defined in a git

**Macroplastics-mediterranean-viewer**  
What-If applications

2022 - 2025 EDITO India

English Terms of service v10.9.3

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

The screenshot shows the EDITO Datalab interface. The top navigation bar includes links for Home, Trainings and tutorials, Datalab, Explore data, Integrate data, About EDITO, Documentation, Support, and a Logout button. The left sidebar contains a navigation menu with options: Reduce, Home, My account, Project settings, Service catalog, My Services (selected), Process catalog, My processes, My Secrets, My Files, and Data Explorer.

The main content area is titled 'My Services'. It features a sub-header 'Access your running services' with a note: 'Services are supposed to be shut down as soon as you stop using them actively.' Below this, there are buttons for Refresh, New service, Delete all, and Events. A 'Running services' section lists the 'Autosubmit-computing-node-ssh' service, which started 3 minutes ago and is in a 'READY' state.

A modal window is open, displaying the service's configuration. It includes a terminal output for 'node-ssh-100232' and a list of instructions for connecting to the computing node. The 'COMPUTING\_NODE' section shows the following configuration:

```
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: ''
```

The 'HOST' key is highlighted in the configuration. A 'Return' button is located at the bottom right of the modal.

Below the modal, a 'Resource usage quotas' section is visible, showing a message: 'Your current resource usage is reasonable.' and a 'Show more (5)' link.

The footer of the interface includes the European Union flag, the 'EU MISSIONS' logo, the 'EDITO' logo, and the text '2022 - 2025 EDITO India'. On the right side of the footer, there are links for English, Terms of service, and version v10.9.3.

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. The top navigation bar includes links for Home, Trainings and tutorials, Datalab, Explore data, Integrate data, About EDITO, Documentation, Support, and a Logout button. A left sidebar contains navigation options: Reduce, Home, My account, Project settings, Service catalog, My Services (selected), Process catalog, My processes, My Secrets, My Files, and Data Explorer.

The main content area is titled "My Services" and features a notification banner: "Access your running services. Services are supposed to be shut down as soon as you stop using them actively." Below this is a toolbar with "Refresh", "+ New service", "Delete all", and "Events" buttons.

The "Running services" section displays two service cards for "Autosubmit-demo". The first card indicates it started "6 minutes ago" and has an "Open" button. The second card indicates it started "4 days ago" and also has an "Open" button. A modal dialog is open over the first card, titled "Autosubmit-computing-node-ssh". The modal contains the text: "Your Autosubmit Demo App is being deployed. This service have the following tools:" followed by a list: "Jupyter Lab" (with a sub-note "Your access token is supersecrettoken") and "Autosubmit GUI". At the bottom of the modal are "Return" and "Click to copy the password..." buttons.

A terminal bar at the top of the services section shows the command: `$ helm get notes autosubmit-demo-920428 --namespace user-ialsina`. To the right, a "Resource usage quotas" section shows a green checkmark and the text: "Your current resource usage is reasonable." with a "Show more (5)" link.

The footer contains the "EU MISSIONS" logo, the "EDITO Datalab" logo, the text "2022 - 2025 EDITO Infra", and links for "English", "Terms of service", and "v10.9.3".



Step 12. Explore your active experiments. Notice the *"Only active"* switch.

The screenshot displays the 'Experiments' section of the RS GUI. At the top, there is a navigation bar with the RS GUI logo, a 'HOME' button, and an 'ABOUT' button. Below the navigation bar, the breadcrumb 'Experiments' is shown. A search bar is present with the placeholder text 'Search by expid, description, or owner...'. To the right of the search bar are buttons for 'more filters', 'Search', 'Clear', and two green circular icons. Below the search bar, there is a 'Type:' dropdown menu set to 'All' and a toggle switch for 'Only active' which is currently turned on. To the right of the toggle are 'Order by:' and 'Page size:' dropdown menus. The main content area displays three experiment cards, each with a title, a progress bar, and a status section.

Experiment ID	Description	Progress	Status	Owner	Version	Start Time
a000	Local dummy workflow.	100%	Active	-	4.1.14	2025-07-17T15:07:58+00:00
a001	Remote dummy workflow	100%	Active	-	4.1.14	2025-07-17T15:26:52+00:00
a002	Versioned workflow example	100%	Active	-	4.1.14	2025-07-17T15:27:02+00:00

At the bottom of the page, there is a pagination bar with a circular icon containing the number '1' and arrows for navigation.

# 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](mailto:support-autosubmit@bsc.es)
- Autosubmit GUI
  - Public mockup: <https://earth.bsc.es/gitlab/wuruchi/autosubmitreact>
  - Demo: <https://autosubmitgui.bsc.es/presentation/>