# Airflow and multi-cluster Slurm working together

Eloi Codina-Torras

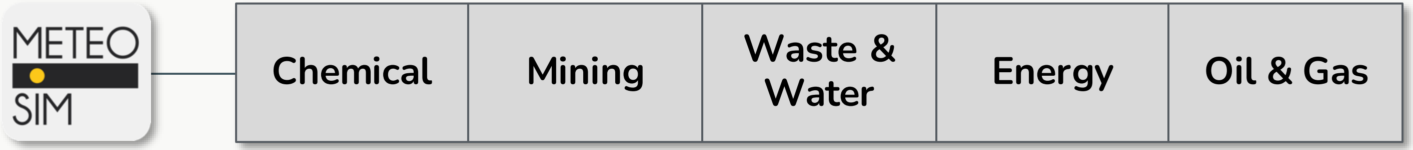
# About me

Eloi Codina-Torras

- Born near Barcelona
- Studied: Industrial Engineering
- Currently: Product Owner @ Meteosim
- Extra: pursuing a PhD in renewable energy
- 2nd Airflow Summit!

# About Meteosim

We offer **meteorological** and **air quality** services to many sectors:



We’re experts in helping our customers

- Evaluate and minimize the environmental impact of their operations
- React to pollution complaints
- Fulfill public administration requisites

# Our use-case

We run **computationally expensive** meteorological and air quality simulations / pipelines:

- Data acquisition
- Pre-processing
- Simulation
- Post-processing

We use:

- A bare-metal machine on-prem
- Virtual machines on the cloud

All the machines are managed with **Slurm**

# Meteosim before Airflow

Hundreds of pipelines were introduced in the **crontab** file

Headaches:

- Bad monitoring. Difficult to know which jobs failed
- Difficult to find the log file for each job
- Difficult to relaunch jobs at the step they failed (a task in a DAG)
- No common practices when writing the pipelines
- Difficult to find the pipeline in the crontab file
- Pipelines running even after they weren’t needed

**All this changed in 2021, when we introduced Airflow**

# Creating the Slurm integration

# Overview

**Computing**

*HPC on-prem*

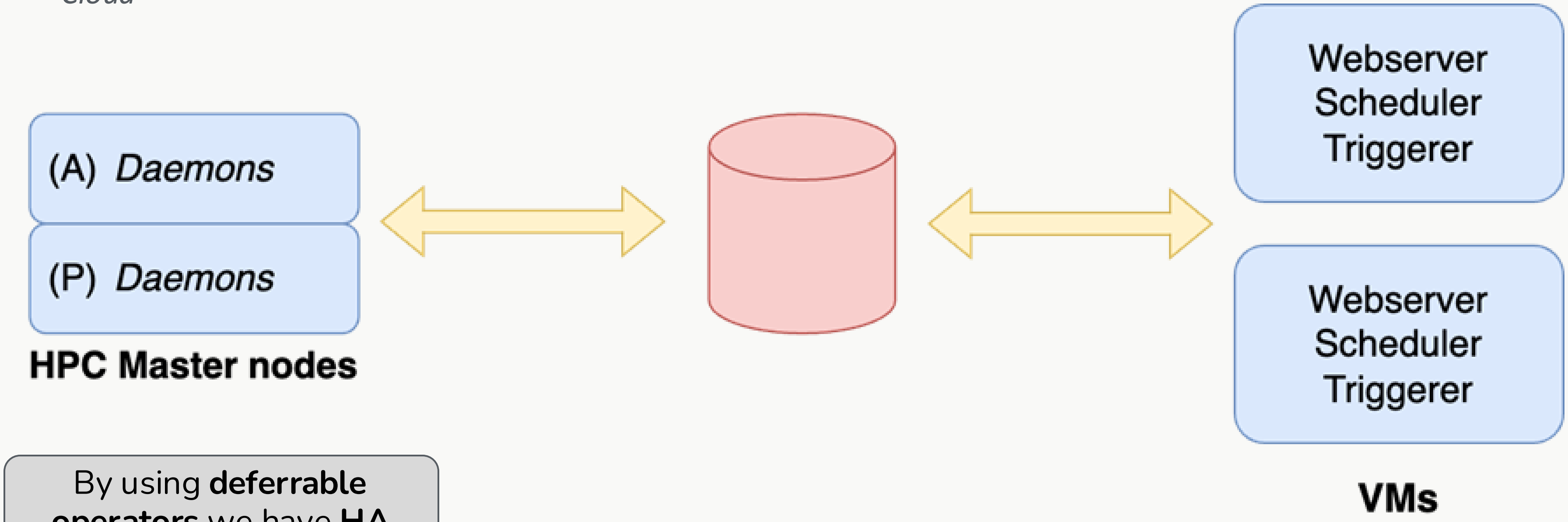
*Cloud*

**Communication layer**

*Redis*

**Airflow**

*In 2 VM*



By using **deferrable operators** we have **HA**

# Example of a message

The message contains information about:

- Which cluster to run the job
- The script
- Resource configuration
- Environment variables the job needs

Daemon #1 adds:

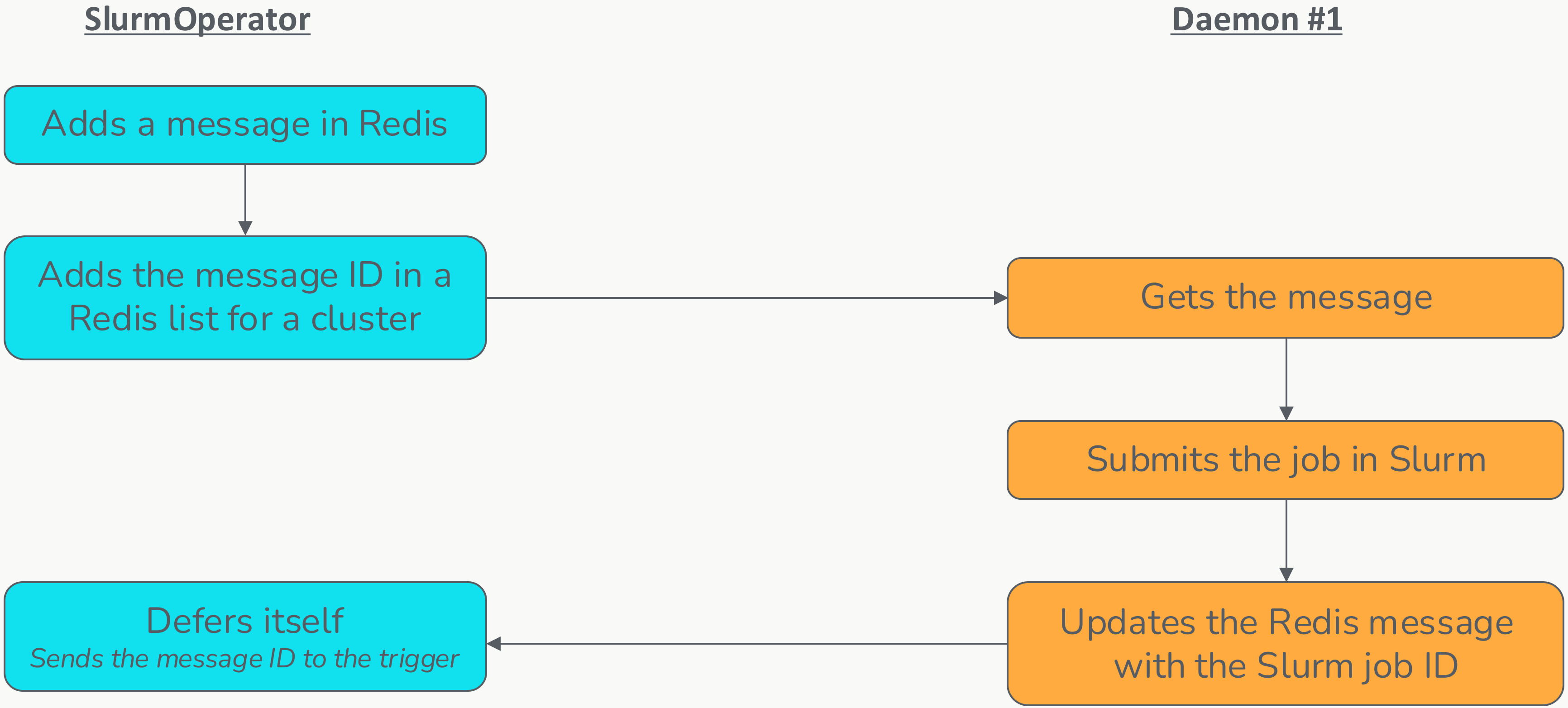
- Information about submission

Daemon #2 adds:

- Information about job state

```
{
  "cluster": "onprem",
  "command": "/path/to/script",
  "slurm_options": {
    "NODES": 1,
    "NTASKS": 1
  },
  "env": {
    "SBATCH_PARTITION": "high",
    "SBATCH_TIMELIMIT": "00:30:00",
    "SBATCH_MEM_PER_NODE": "20G"
  },
  "result": {
    "exit_code": 0,
    "job_id": 123456,
    "message": "reason_why_submit_failed"
  },
  "sacct_result": {
    "state": "COMPLETED",
    "reason": "reason_why_current_state"
  }
}
```

# How do we submit a job?



**SlurmOperator**

Adds a message in Redis

Adds the message ID in a Redis list for a cluster

Defers itself

*Sends the message ID to the trigger*

**Daemon #1**

Gets the message

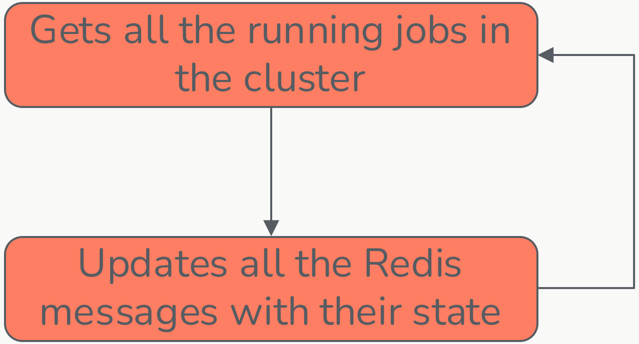
Submits the job in Slurm

Updates the Redis message with the Slurm job ID

# How do we monitor jobs?

**Daemon #2**

*Every 5 seconds*

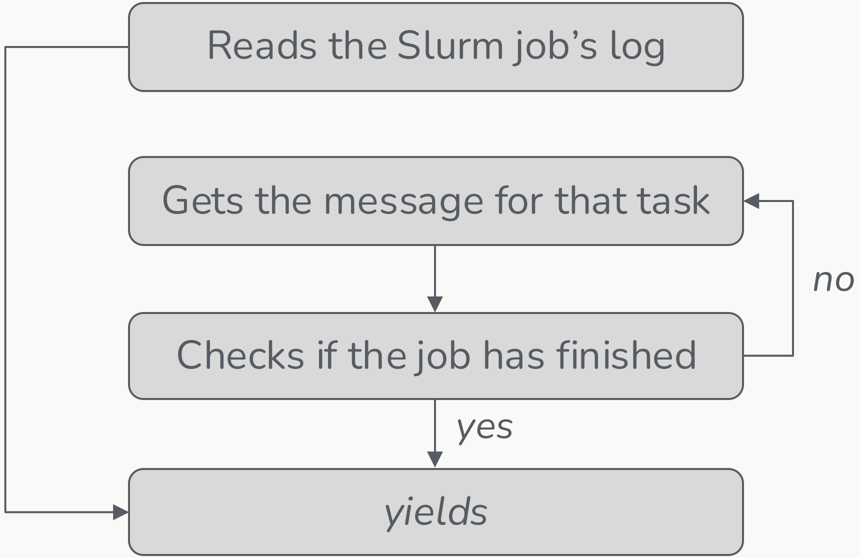


Gets all the running jobs in the cluster

Updates all the Redis messages with their state

**SlurmTrigger**

*Every 5 - 60 seconds*



Reads the Slurm job's log

Gets the message for that task

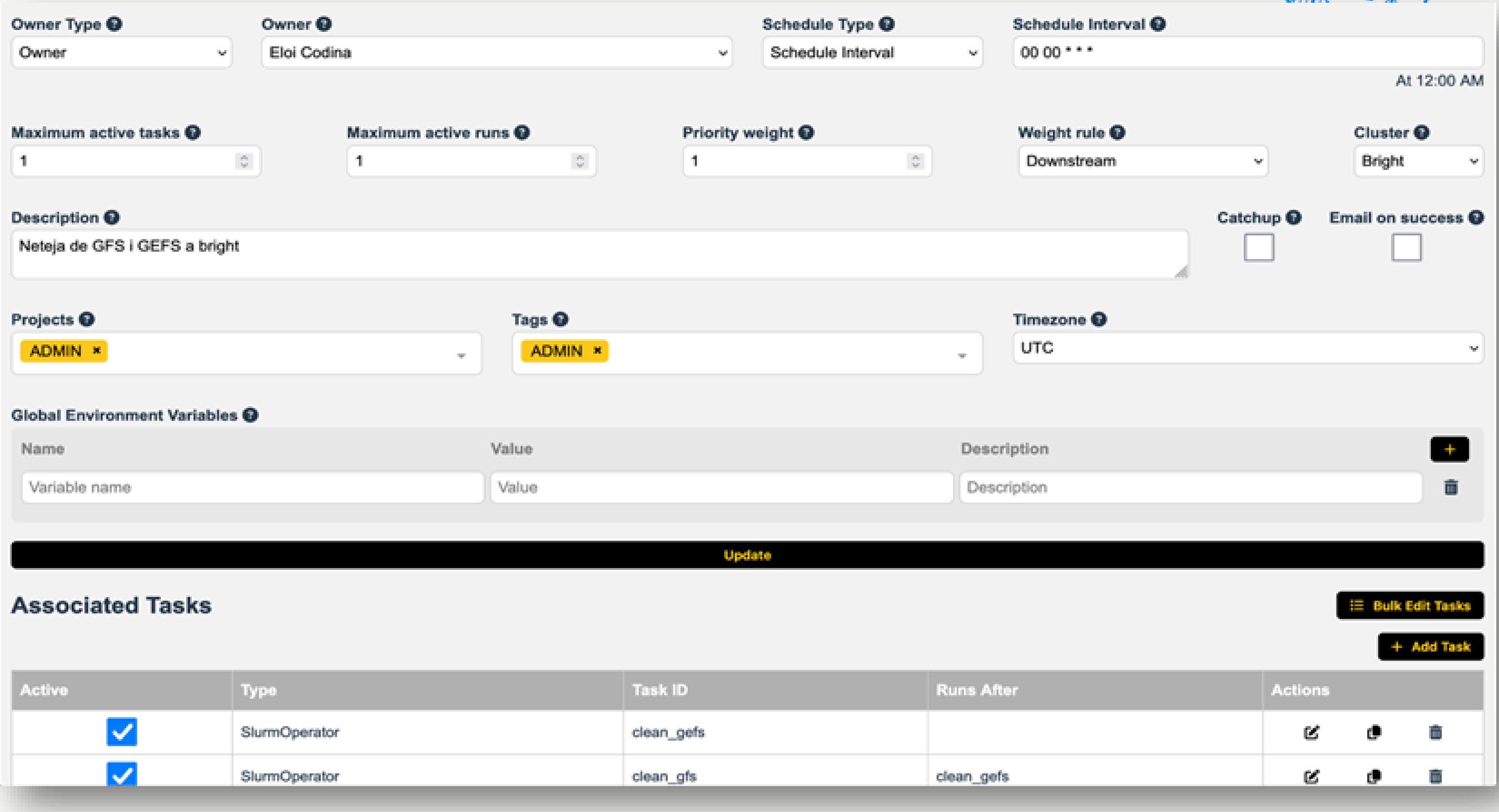
Checks if the job has finished

*no*

*yes*

*yields*

# Manage DAGs



Owner Type
Owner

Owner
Eloi Codina

Schedule Type
Schedule Interval

Schedule Interval
00 00 * * *
At 12:00 AM

Maximum active tasks
1

Maximum active runs
1

Priority weight
1

Weight rule
Downstream

Cluster
Bright

Description
Neteja de GFS i GEFS a bright

Catchup
□

Email on success
□

Projects
ADMIN ×

Tags
ADMIN ×

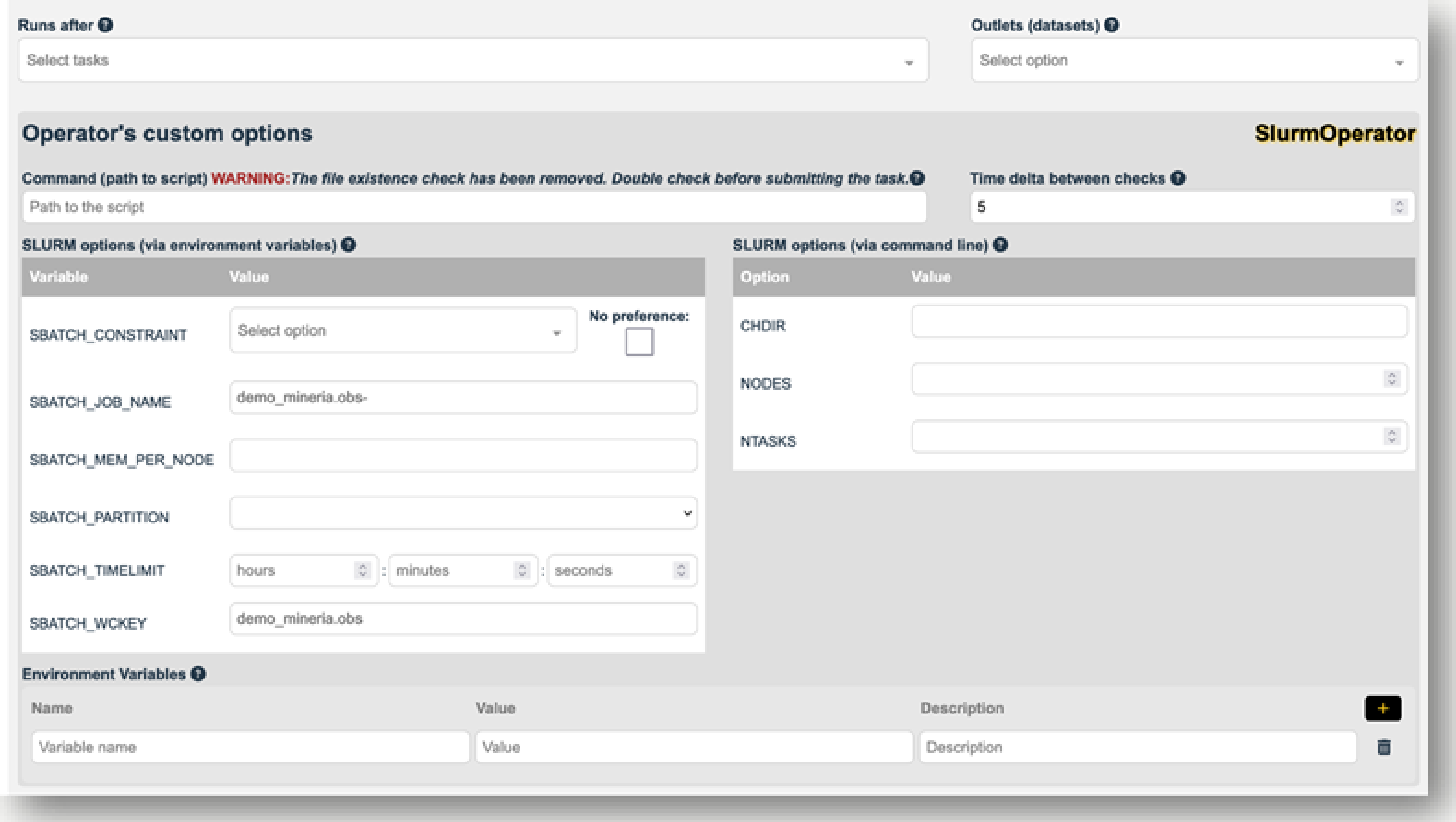
Timezone
UTC

Global Environment Variables

| Name | Value | Description |
|---|---|---|
| Variable name | Value | Description |

## Associated Tasks

| Active | Type | Task ID | Runs After | Actions |
|---|---|---|---|---|
| ☑ | SlurmOperator | clean_gefs | | |
| ☑ | SlurmOperator | clean_gfs | clean_gefs | |



Runs after ❓

Select tasks ▾

Outlets (datasets) ❓

Select option ▾

## Operator's custom options

**SlurmOperator**

Command (path to script) **WARNING:** *The file existence check has been removed. Double check before submitting the task.* ❓

Path to the script

Time delta between checks ❓

5

SLURM options (via environment variables) ❓

| Variable | Value |
|---|---|
| SBATCH_CONSTRAINT | Select option ▾ No preference: □ |
| SBATCH_JOB_NAME | demo_mineria.obs- |
| SBATCH_MEM_PER_NODE | |
| SBATCH_PARTITION | ⌄ |
| SBATCH_TIMELIMIT | hours : minutes : seconds |
| SBATCH_WCKEY | demo_mineria.obs |

SLURM options (via command line) ❓

| Option | Value |
|---|---|
| CHDIR | |
| NODES | |
| NTASKS | |

Environment Variables ❓

| Name | Value | Description | + |
|---|---|---|---|
| Variable name | Value | Description | 🗑 |

# Conclusion

# A success story

**6000** **runs / day**

**0% failure**
*due to the integration*

We can now:

- Let **Slurm manage resource** and **Airflow dependencies and schedules**
- **Run** jobs in multiple clusters with a **single source of truth**
- **Read logs** from all the jobs in one single platform
- Restart any component of the integration: it has **high availability**!

Moreover:

- Creating DAGs is as easy as configuring a form on a webpage
- Every DAG is stored as a YAML file

# Questions?

Eloi Codina-Torras

*eloi-codina*