
# Introduction to the workshop

F. Vazquez de Sola

KM3NeT Grid Tutorial & Workshop, February 2025

netherlands eScience center

Nikhef

NWO

# Introduction to workshop program

Monday

- Morning: Introduction to workshop, setup
- Afternoon: Using Rucio for grid storage
- Visit to DOM assembly hall, workshop dinner!

Tuesday

- Morning: Using Dirac for grid computing
- Afternoon: Using snakemake for complex workflows

Wednesday

- Dirac, Snakemake and Rucio for KM3NeT’s data processing

Each session starts with an introduction to the main topics, followed by a hands-on tutorial, and finishes with a discussion on next steps or missing functionality

# Motivation for grid usage

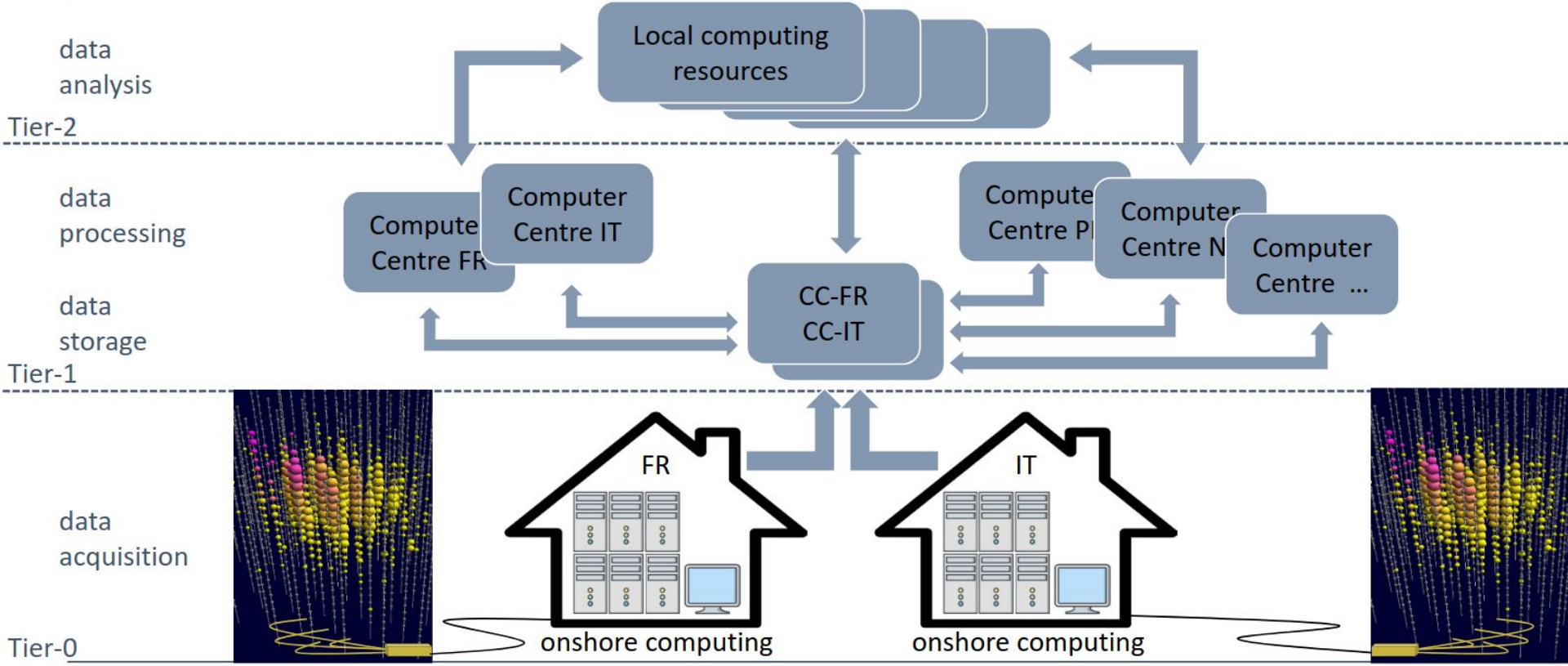
Benefits:

- Scalability of CPU / storage
- Simplifying data sharing
- Resilience to site downtime

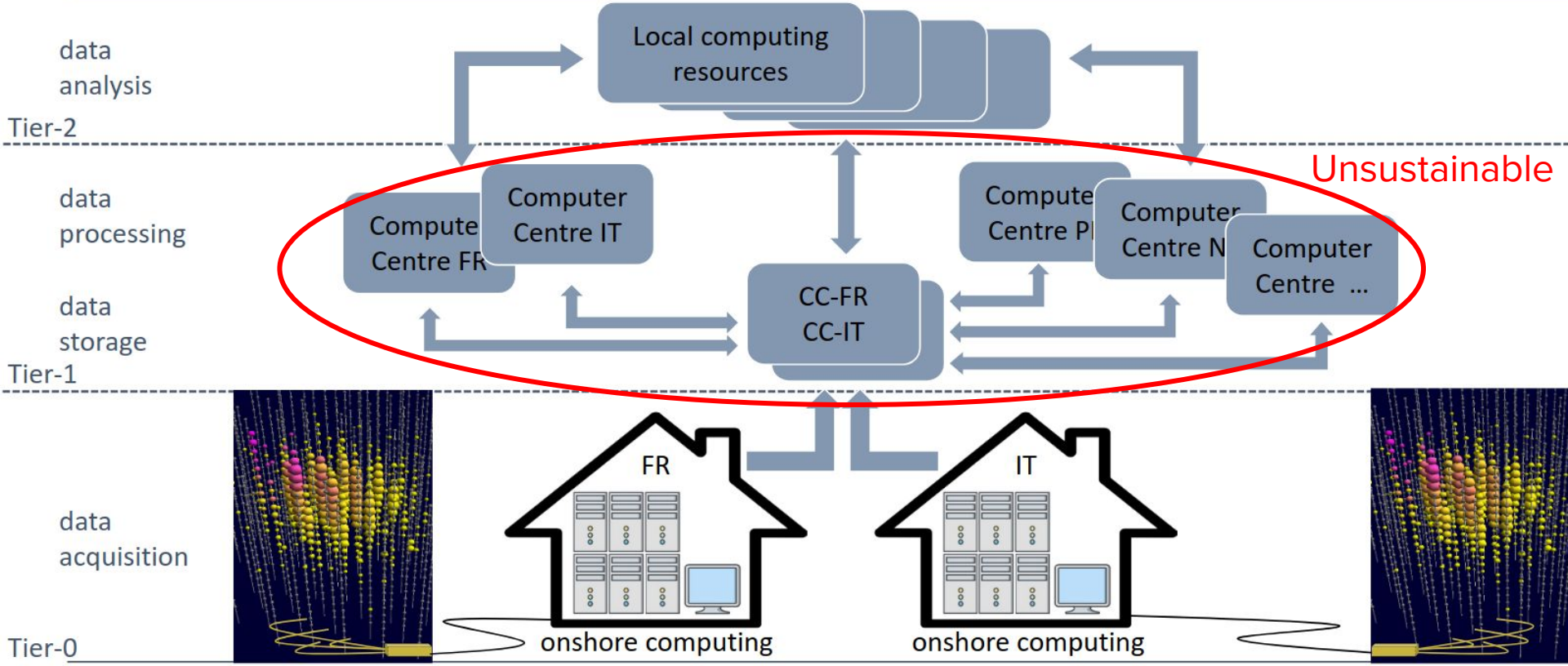
Goals:

- Run “standard” productions and simulations
- Abstract infrastructure for user / shifter

# KM3NeT (previous) Computing Model

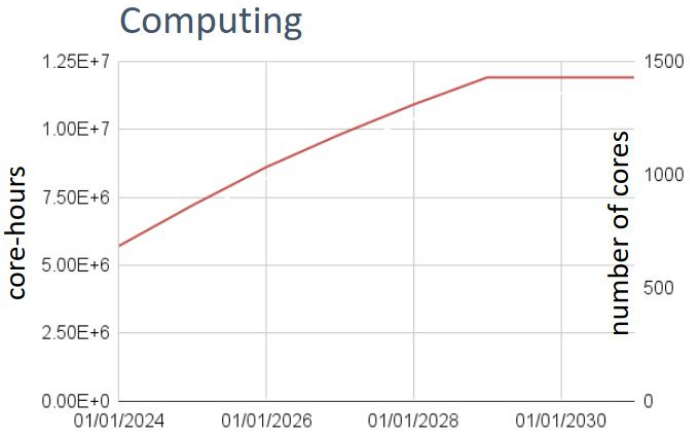


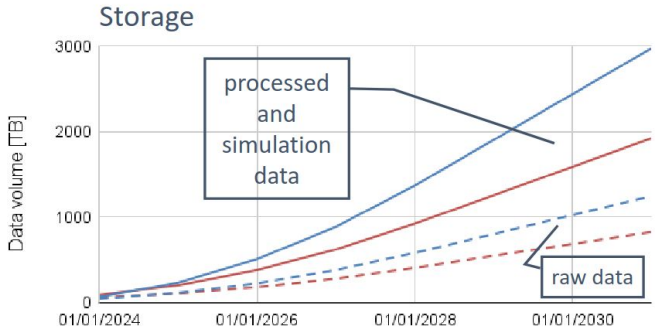
# KM3NeT (previous) Computing Model



# Why unsustainable?

- In practice, most data and computing ended up centralized in a single site, CC-IN2P3 (common environments, shared file system)…
  - Simulation / processing expected to grow to take O(1000) cores, and double/triple storage needs from raw data;
  - IN2P3 resources will not scale like our needs;
  - CC-IN2P3 downtime means collaboration stand-still.
- Attempting to manually split the processing across partner sites lead to large overhead: authentication, environments, file transfers, bookkeeping…



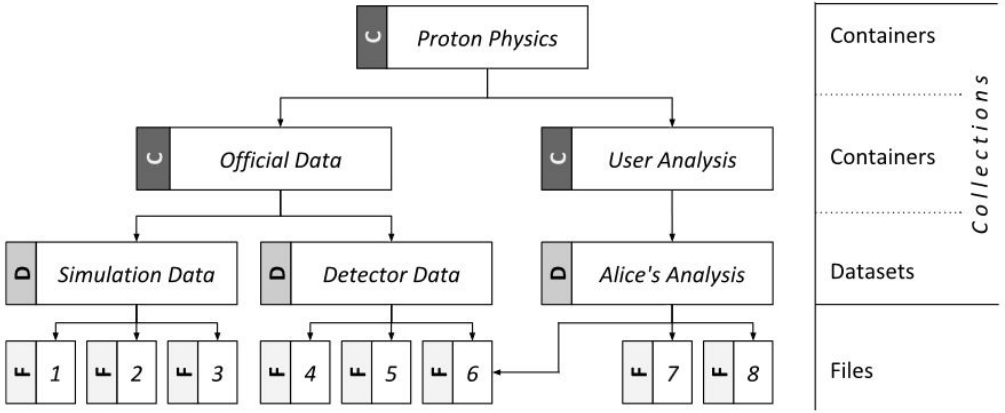

**Must transition to distributed storage and computing**

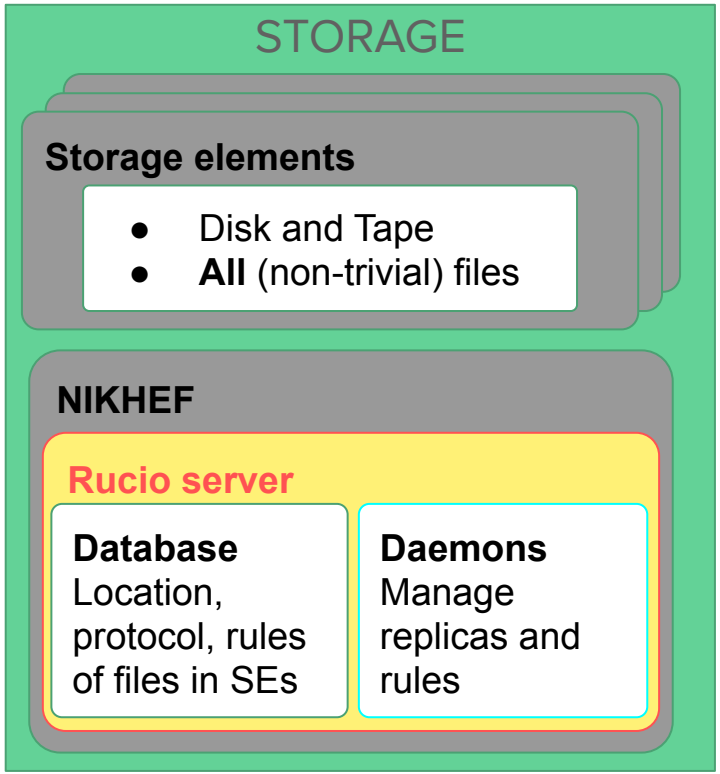
# The different elements

https://link.springer.com/article/10.1007/s41781-019-0026-3

# Grid Storage

- Regular access to grid storage requires knowing physical location of every file:
  - Host address and port
  - Path to file within site
  - File transfer protocol (gfal, xrootd, webdav...)
- Enter RUCIO!



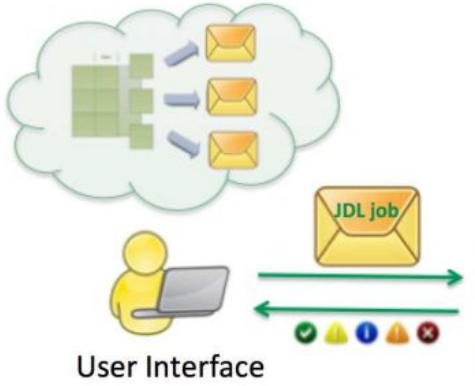

Thanks to Victor A. and Bouwe A. from eScience Center

- Rucio is an interface software to grid storage.
- Provides user-chosen Data Identifiers (DIDs, effectively “aliases”) instead of true paths and protocols to files.
- Can organize files with datasets and containers, and add metadata.
- Convenient way to share files between sites / collaborators.
- Manages replicas automatically through replication rules.

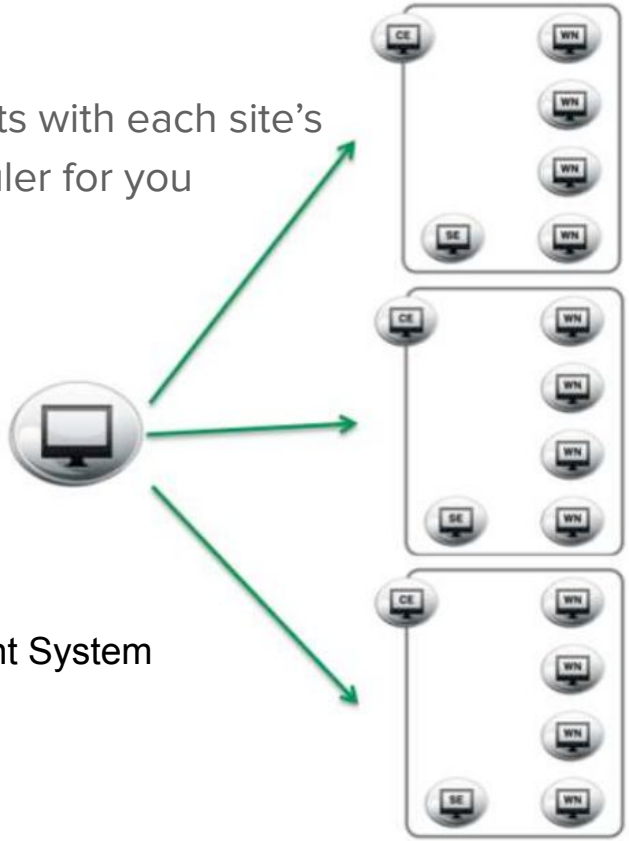
https://dirac.readthedocs.io/en/latest/index.html

# Grid Computing

- Interacting with different
- Enter Dirac!




Interacts with each site's scheduler for you

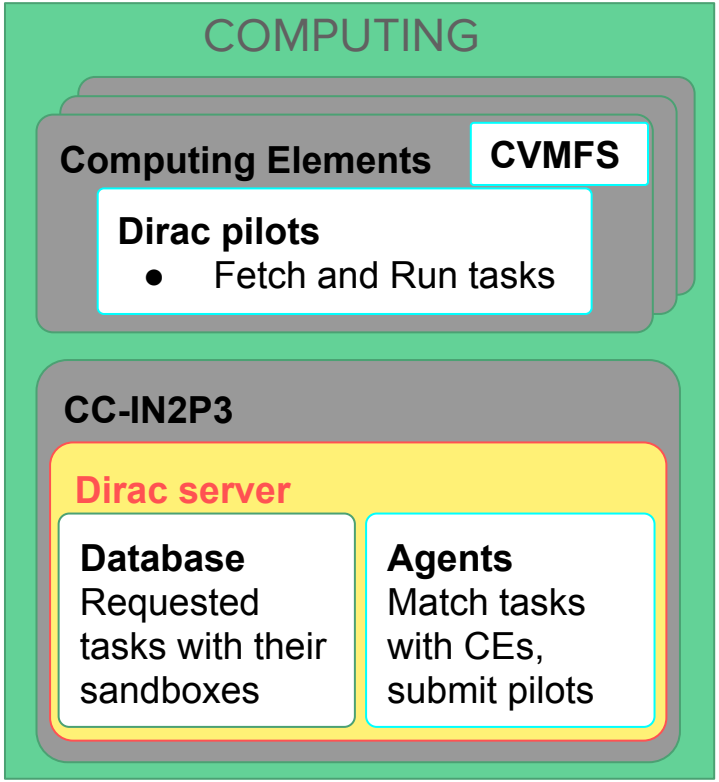



WMS

Worker Management System

Distributed Infrastructure with Remote Agent Control

- Dirac is the middleware interface to computing resources, originally developed by LHCb.
- Mediates between user and site schedulers, propagates user authentication, monitors job progress... (and much more, that we don't use)
- Computing Elements are equipped with the CernVM File System (CVMFS), allowing access to our software for processing.




Thanks to Andrei T. (and many others) from EGI

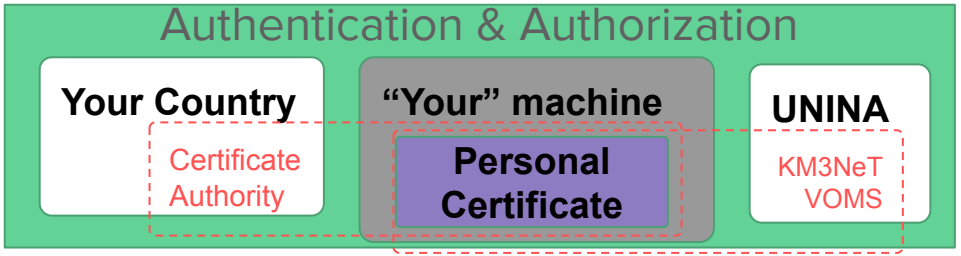
# Authentication & Authorization: X509 Certificates

- **Authentication** from Certification Authority: National entity that issues certificates

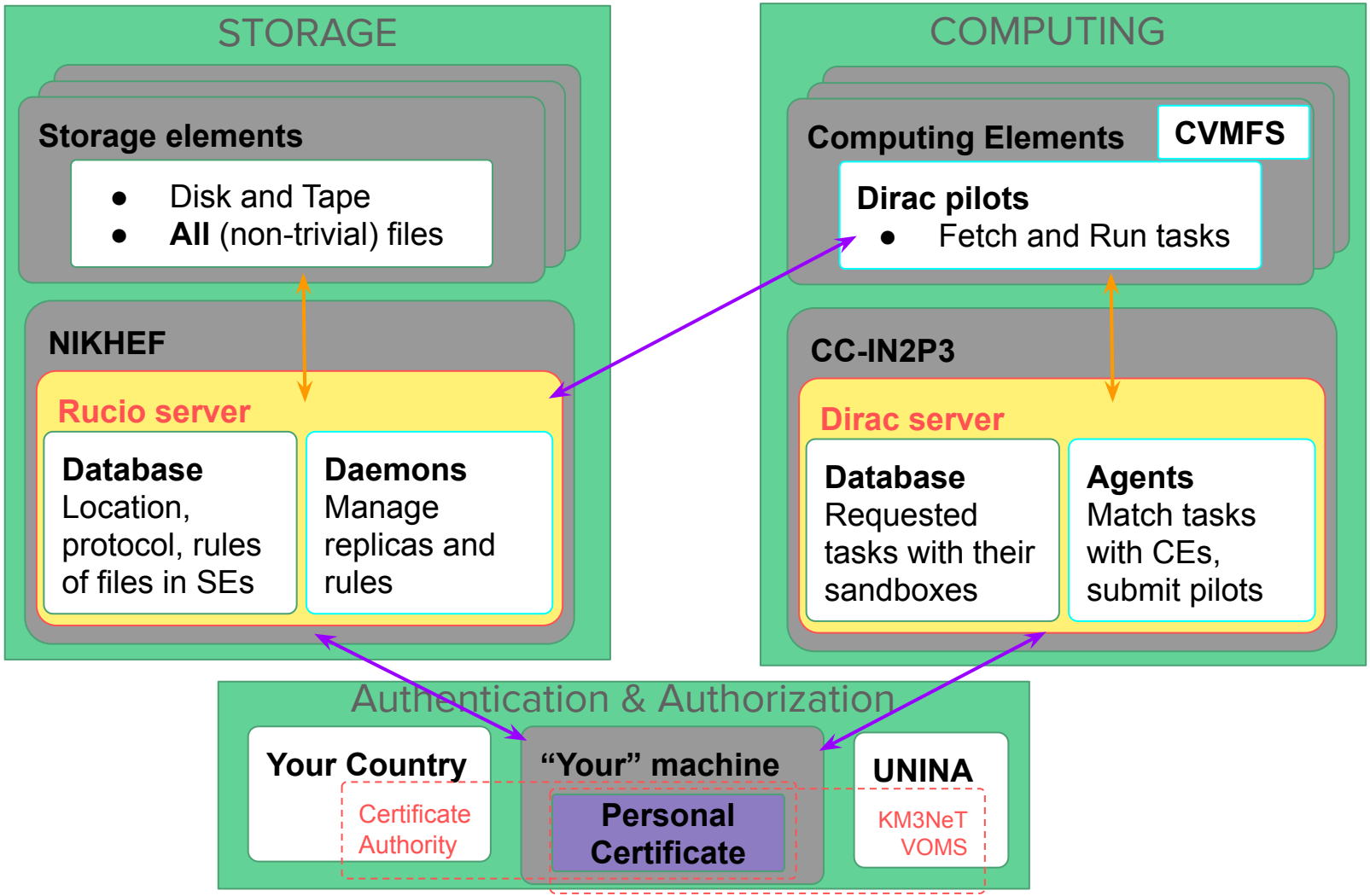
- **Authorization** from “Virtual Organisation”: Group sharing scientific field and research interests (e.g. lsgrid (lifesciences), escape, lofar, km3net.org...)

    - Determine which resources (compute/storage) you can access via extensions to your certificate
    - VOMS (VO Management Service) sets user roles and privileges in a VO, maintains server that returns certificate attributes (eg. membership)

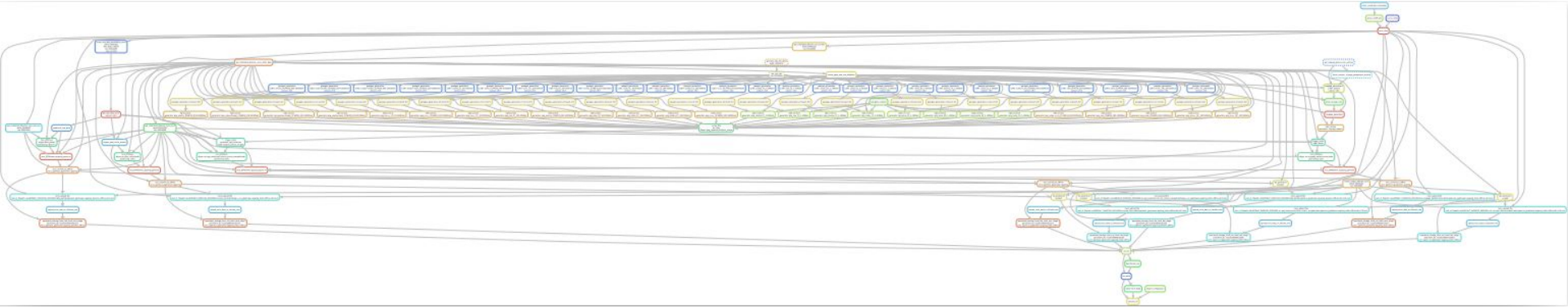

- Authentication & Authorization is what allows you to use Dirac and Rucio, and many other grid resources.
- Currently using X509 certificates, which requires both a country-specific and a collaboration-specific procedure.
  - Currently the most annoying part of working with grid resources.
- UNINA stopping support of KM3NeT VOMS, but we still have a couple of years left. Want to move to alternative (i.e. tokens?) before then.




More details on setup here


STORAGE
Storage elements
Disk and Tape
All (non-trivial) files
NIKHEF
Rucio server
Database
Location, protocol, rules of files in SEs
Daemons
Manage replicas and rules
COMPUTING
Computing Elements
CVMFS
Dirac pilots
Fetch and Run tasks
CC-IN2P3
Dirac server
Database
Requested tasks with their sandboxes
Agents
Match tasks with CEs, submit pilots
Authentication & Authorization
Your Country
Certificate Authority
"Your" machine
Personal Certificate
UNINA
KM3NeT VOMS

# Last but not least:
# What do you do when your workflow looks like this?



# Last but not least:
# What do you do when your workflow looks like this?

You automate it!

- Snakemake automatically generates workflows from output files based on (generic) rules
- Reproducible! Use containerized software, automatic monitoring and logging...

# Status of gridification

# Status of tasks

Rucio

- Raw & calibration & processed data available through Rucio
- Datasets/Containers and metadata used to organize files
- Access to storage at IN2P3, SURF and Nikhef

Dirac

- CORSIKA simulations and Run-Based Data Processing adapted for Dirac
- Access to computing resources at IN2P3, SURfF, Nikhef and CPPM

No Memoranda of Understanding or Service-Level Agreements with any sites or institutions. These are required before scaling up resource use.

# Development & Maintenance

| Task | Description | Number of FTE |
|---|---|---|
| Maintaining Workflow Management System | Monitoring, interaction with Computing Sites, resource allocation | 2 |
| Maintaining Data Management System | Site interaction, troubleshooting | 1 |
| General User support Grid | User training, troubleshooting, support of code migration | 2 |

Two scenarios:

- Power users maintaining and running the processing
  - Only a handful of people are responsible for running and monitoring production. Can be different from the code maintainers.
- Community processing:
  - Everyone can run productions. Still need code maintainers. Require an additional helpdesk, but no dedicated “producers”.

Estimate three to eight people needed long term (depending on scenario and how much help we can get from sites / EGI / DIRAC)

# Upcoming tasks

- **Short term:**
  - Increase user base (hi!)
  - Update CVMFS containers to run at all sites
  - Write unit tests / gitlab CI
  - Test within DP/DQ group
  - Improve documentation
- **Medium and long term** (from my point of view):
  - https://git.km3net.de/workflow-management/grid-tools/-/issues/10
- The *big* tickets are :
  - Implementing a new AAI solution, including a “shared” production account
  - Transition DPDQ to use grid processing / storage
  - Managing the large (and growing!) walltime needed to process one run by subdividing it
  - Move to snakemake 8 to use Rucio plugin / metadata, integrate calibrations & other workflows

# Summary

- First stage of transition to the grid almost complete.
  - The two central elements, Rucio and Dirac, are available for use!
  - Gridification of CORSIKA demonstrated during the summer.
  - Run-Based Data Processing functional on the grid, but not tested to the same extent.
- Grid tools are not a silver bullet. A lot of work left before we are ready for full detector size, and maintenance on top of that. We need more people!

# Hands-on / Setup!

# Setup

- Follow the instructions in here.
- If you are using a temporary proxy (instead of your own certificate) for this workshop, then instead do:

```
$ mkdir $HOME/.globus
$ cd $HOME/.globus
$ ln -s /path/to/proxy usercert.pem
$ ln -s /path/to/proxy userkey.pem
```

# Setup

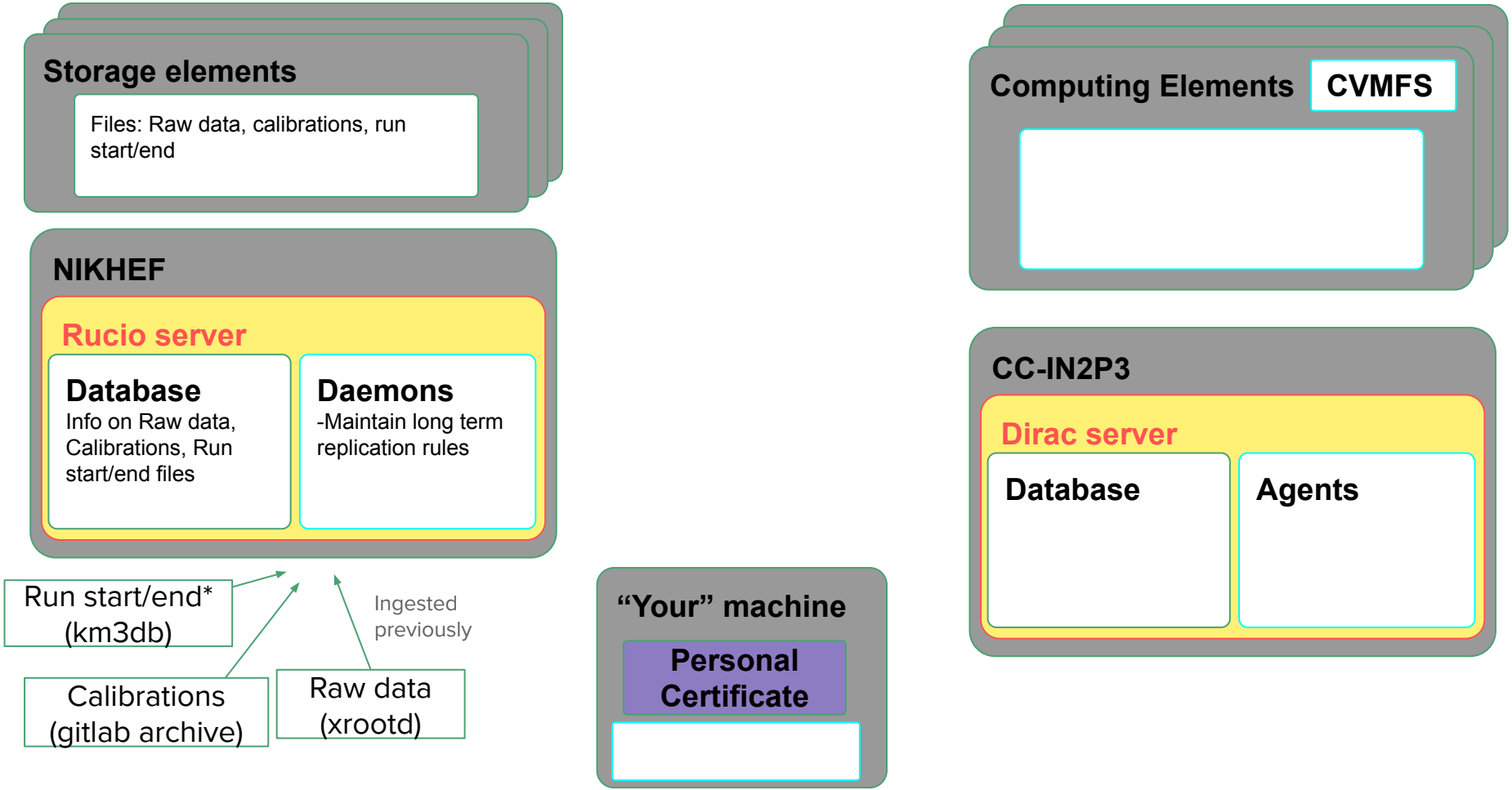
By the end of setup, you should be able to run the following commands without errors:

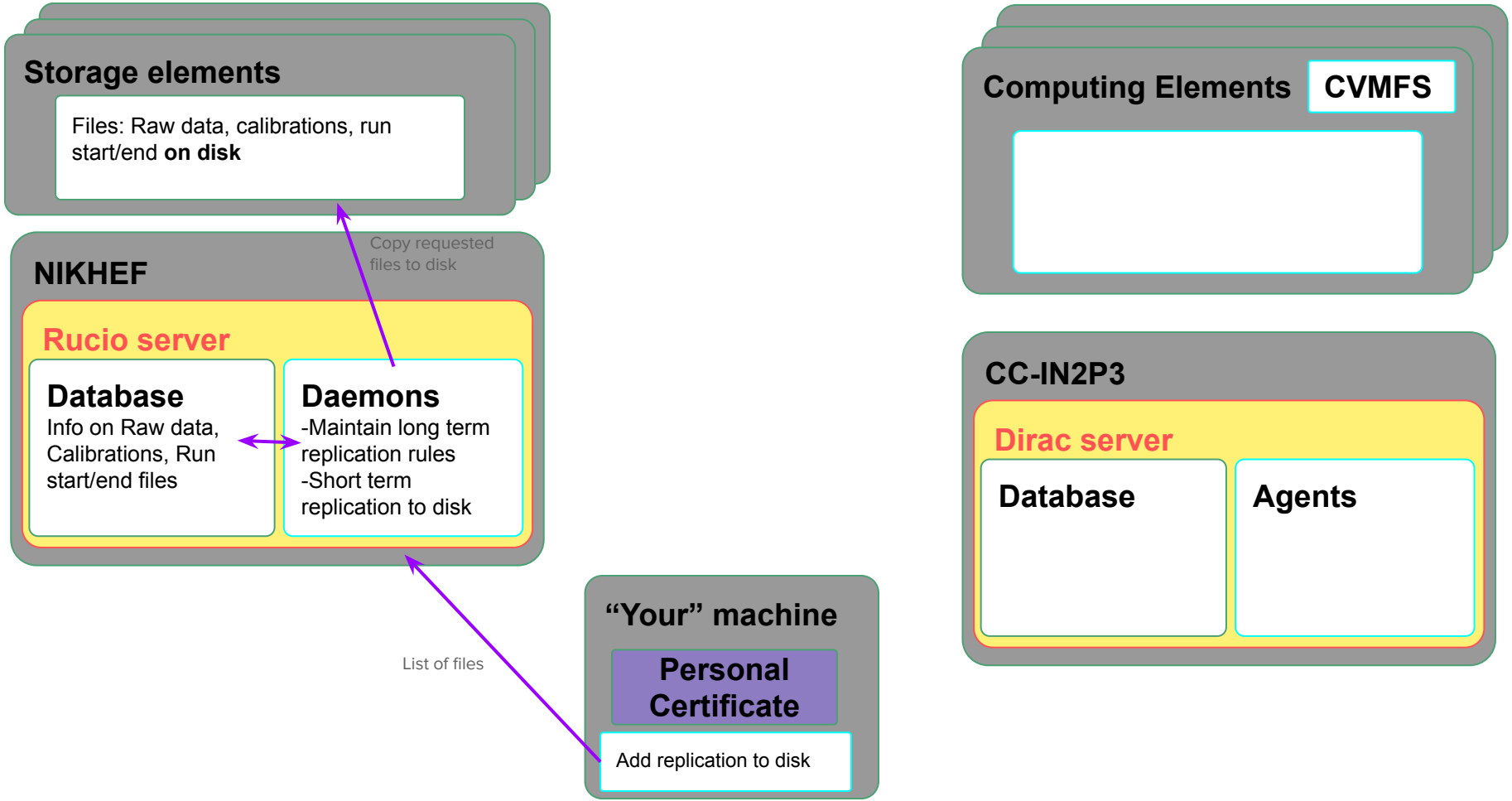
$ voms-proxy-init --voms km3net.org
$ xrdfs km3net.dcache.nikhef.nl ls /pnfs/nikhef.nl/data/km3net/testing
$ rucio whoami
$ rucio download -d testing:hello_dataset
$ source /cvmfs/dirac.egi.eu/dirac/bashrc_egi *
$ dirac-proxy-init -g km3net_user -M -b 2048 --valid 168:00

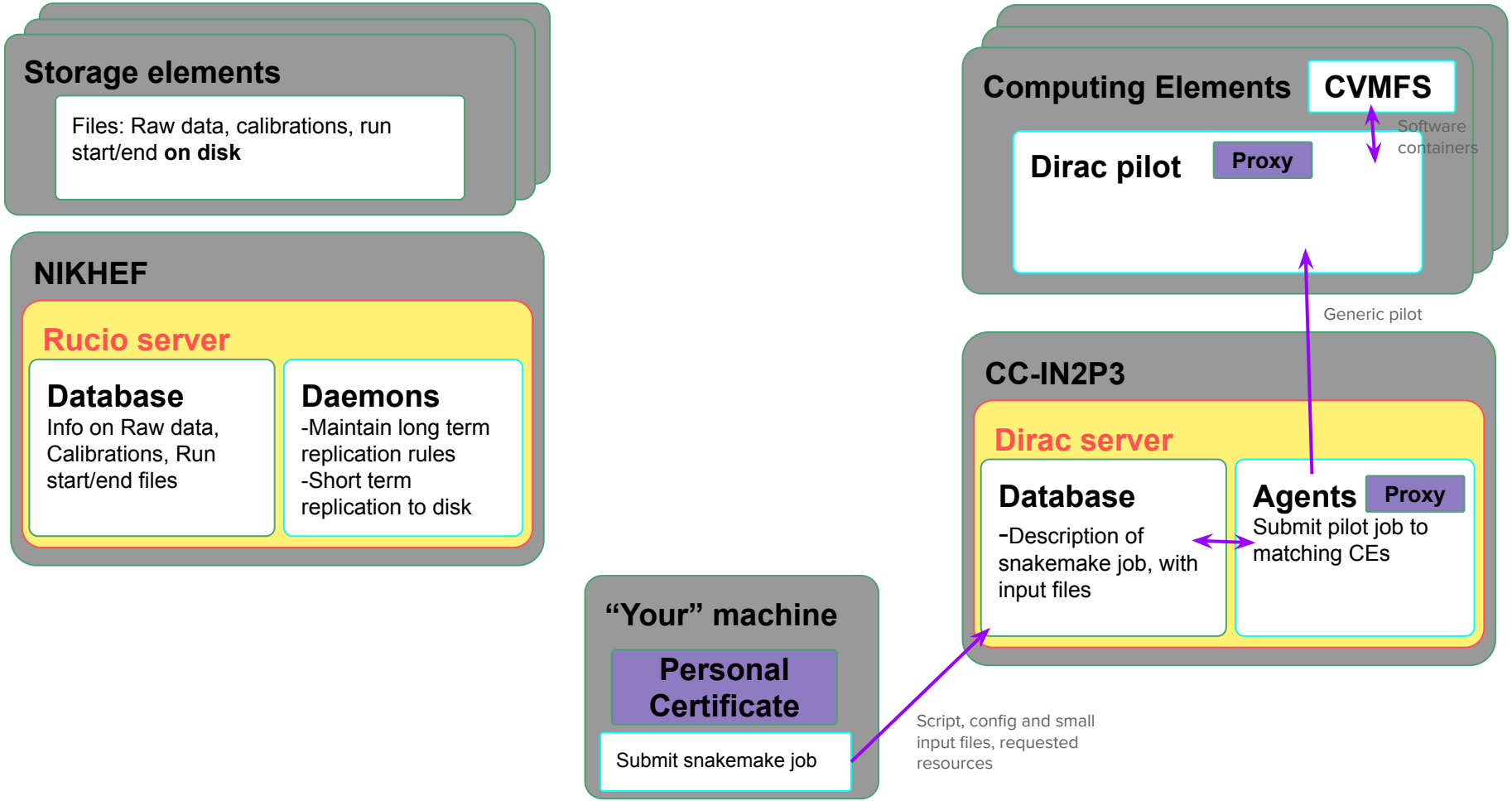
* not necessary if using the dirac client

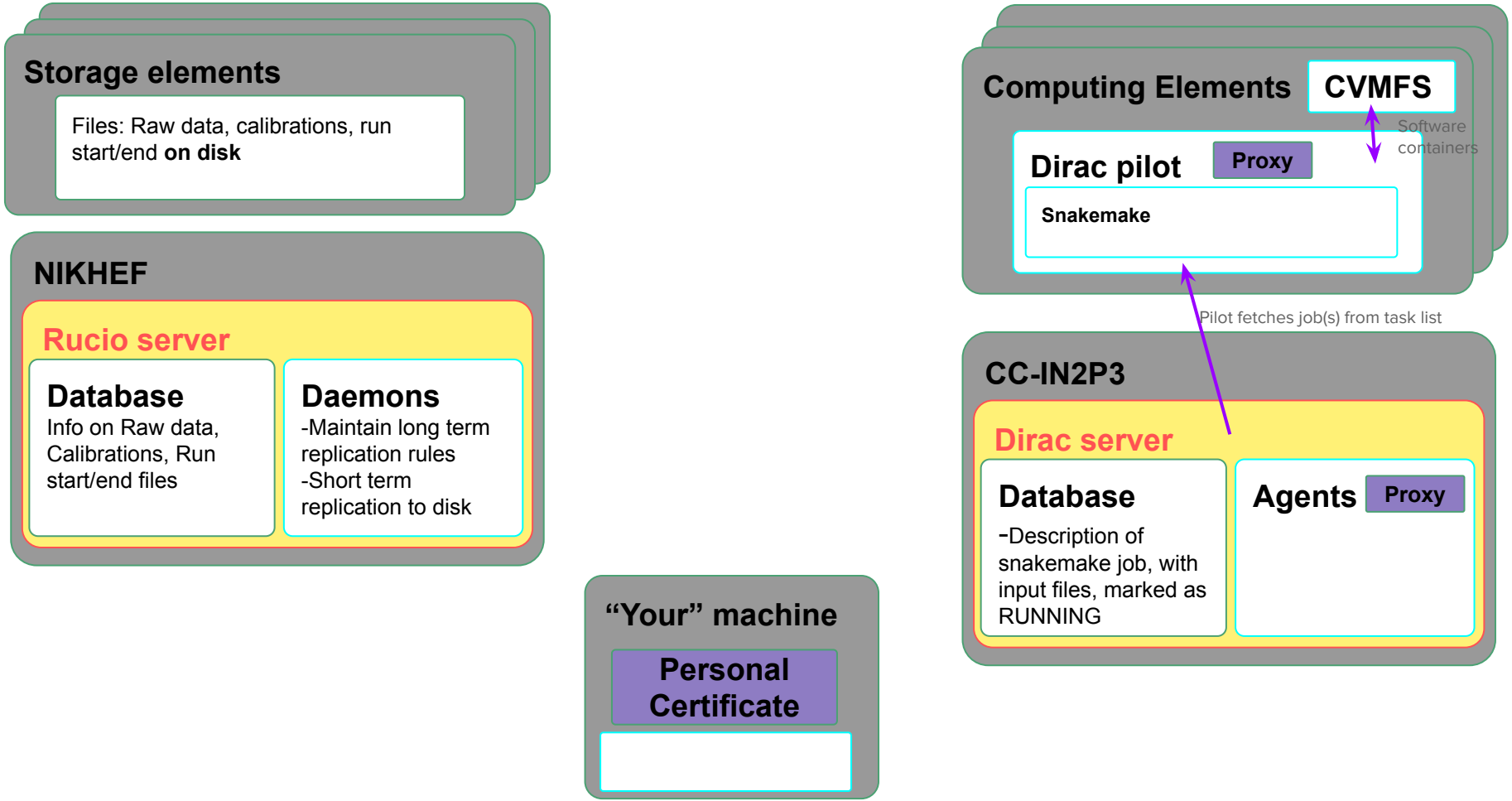
# Backup

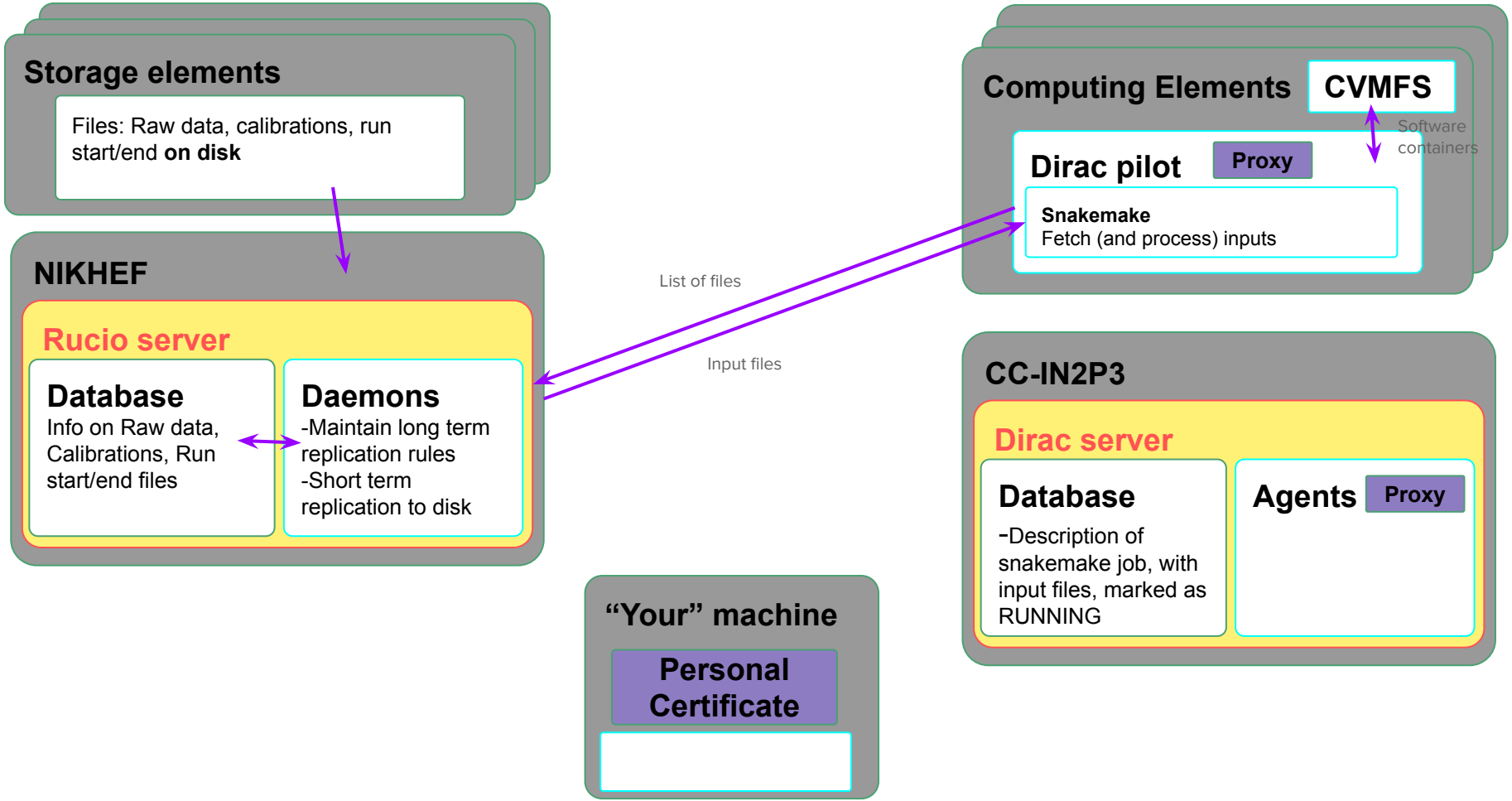
# How does it work?

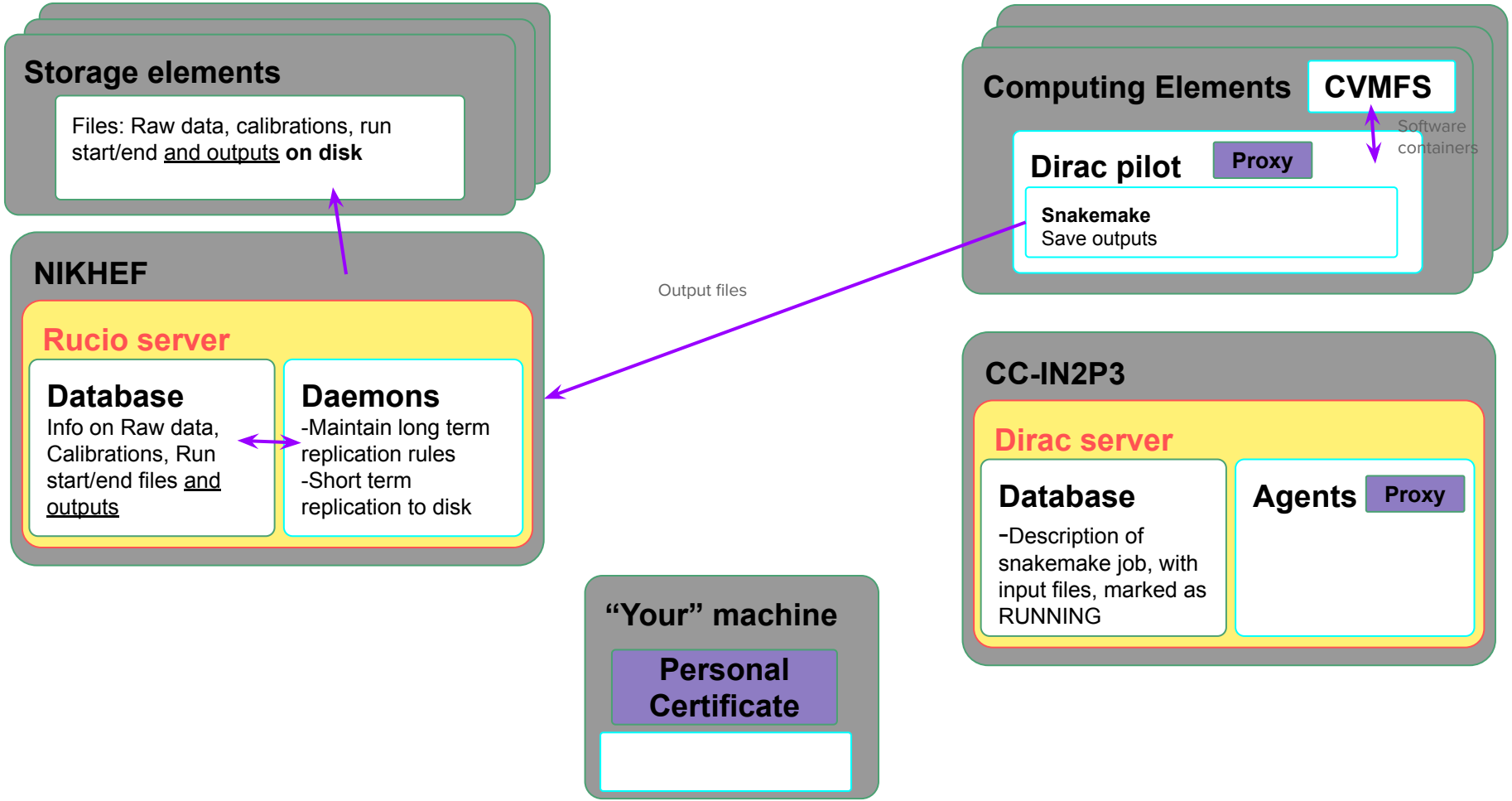

Storage elements
Files: Raw data, calibrations, run start/end
NIKHEF
Rucio server
Database
Info on Raw data, Calibrations, Run start/end files
Daemons
-Maintain long term replication rules
Run start/end* (km3db)
Calibrations (gitlab archive)
Raw data (xrootd)
Ingested previously
"Your" machine
Personal Certificate
Computing Elements
CVMFS
CC-IN2P3
Dirac server
Database
Agents

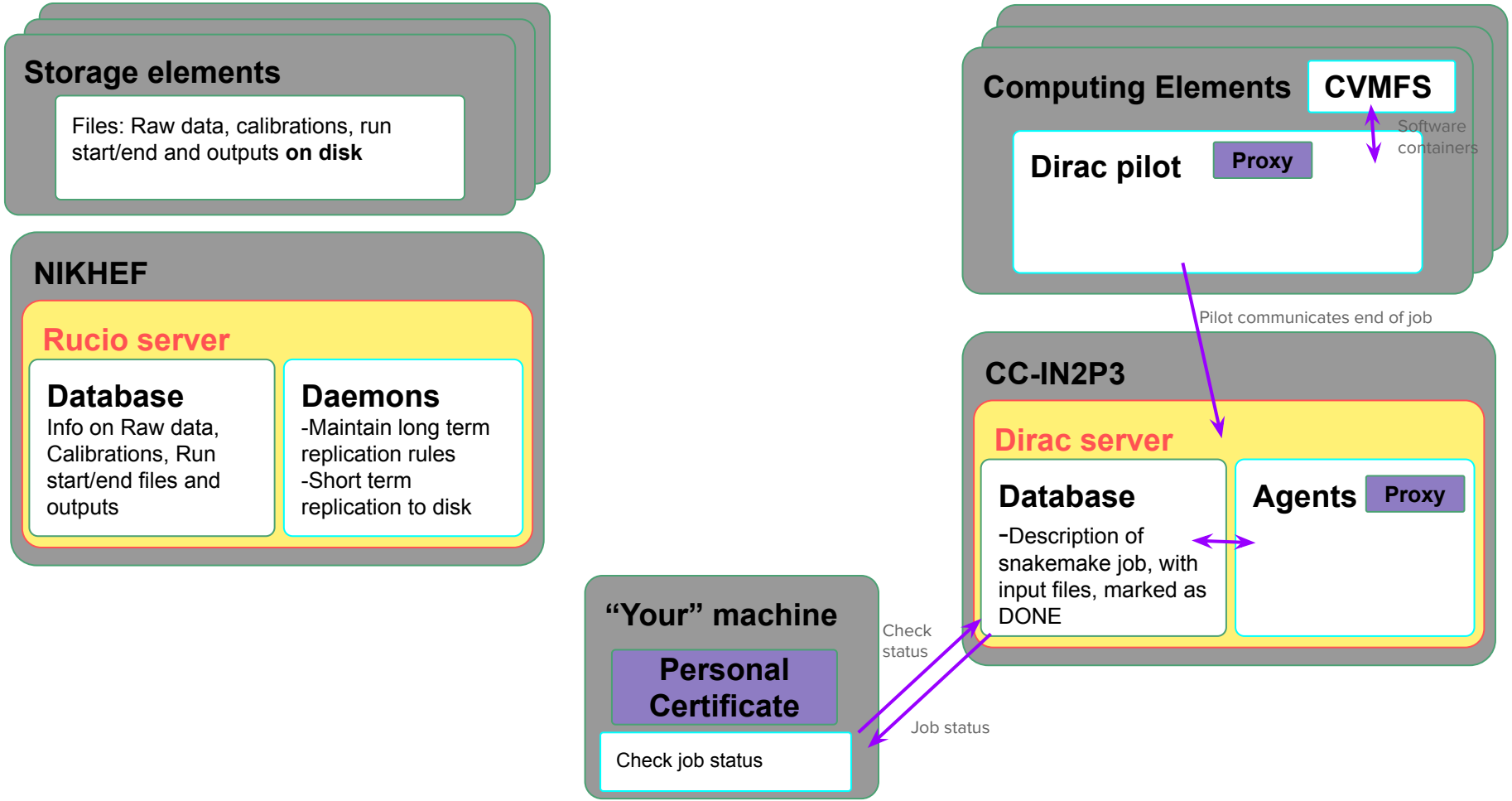

Storage elements
Files: Raw data, calibrations, run start/end on disk
Computing Elements
CVMFS
Copy requested files to disk
NIKHEF
Rucio server
Database
Info on Raw data, Calibrations, Run start/end files
Daemons
-Maintain long term replication rules
-Short term replication to disk
CC-IN2P3
Dirac server
Database
Agents
List of files
“Your” machine
Personal Certificate
Add replication to disk

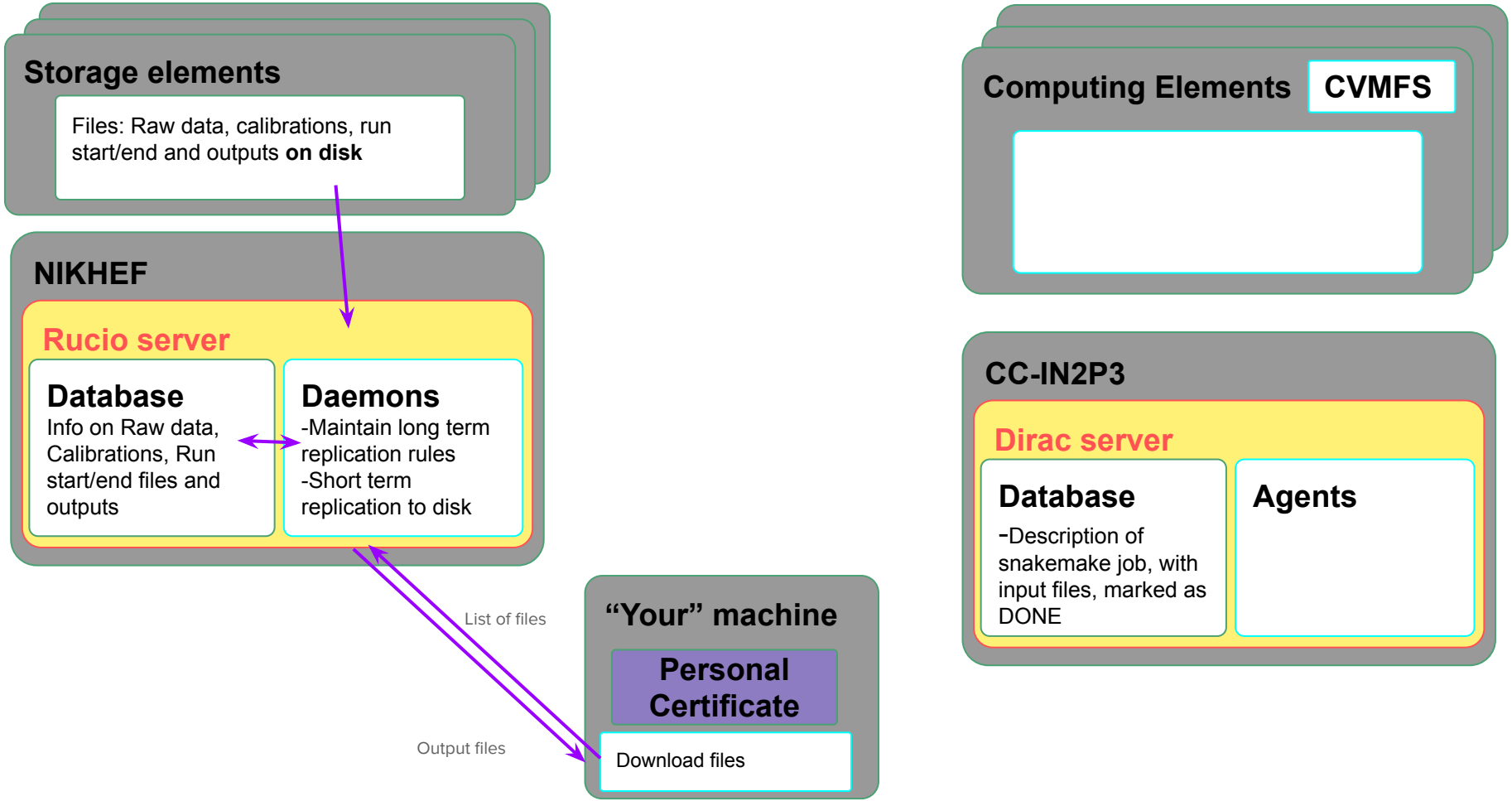

Storage elements
Files: Raw data, calibrations, run start/end on disk
NIKHEF
Rucio server
Database
Info on Raw data, Calibrations, Run start/end files
Daemons
-Maintain long term replication rules
-Short term replication to disk
Computing Elements
CVMFS
Software containers
Dirac pilot
Proxy
Generic pilot
CC-IN2P3
Dirac server
Database
-Description of snakemake job, with input files
Agents
Proxy
Submit pilot job to matching CEs
"Your" machine
Personal Certificate
Submit snakemake job
Script, config and small input files, requested resources


Storage elements
Files: Raw data, calibrations, run start/end on disk
NIKHEF
Rucio server
Database
Info on Raw data, Calibrations, Run start/end files
Daemons
-Maintain long term replication rules
-Short term replication to disk
"Your" machine
Personal Certificate
Computing Elements
CVMFS
Software containers
Dirac pilot
Proxy
Snakemake
Pilot fetches job(s) from task list
CC-IN2P3
Dirac server
Database
-Description of snakemake job, with input files, marked as RUNNING
Agents
Proxy


Storage elements
Files: Raw data, calibrations, run start/end on disk
NIKHEF
Rucio server
Database
Info on Raw data, Calibrations, Run start/end files
Daemons
-Maintain long term replication rules
-Short term replication to disk
List of files
Input files
Computing Elements
CVMFS
Software containers
Dirac pilot
Proxy
Snakemake
Fetch (and process) inputs
CC-IN2P3
Dirac server
Database
-Description of snakemake job, with input files, marked as RUNNING
Agents
Proxy
"Your" machine
Personal Certificate


Storage elements
Files: Raw data, calibrations, run start/end and outputs on disk
NIKHEF
Rucio server
Database
Info on Raw data, Calibrations, Run start/end files and outputs
Daemons
-Maintain long term replication rules
-Short term replication to disk
Output files
Computing Elements
CVMFS
Software containers
Dirac pilot
Proxy
Snakemake
Save outputs
CC-IN2P3
Dirac server
Database
-Description of snakemake job, with input files, marked as RUNNING
Agents
Proxy
"Your" machine
Personal Certificate


Storage elements
Files: Raw data, calibrations, run start/end and outputs on disk
NIKHEF
Rucio server
Database
Info on Raw data, Calibrations, Run start/end files and outputs
Daemons
-Maintain long term replication rules
-Short term replication to disk
Computing Elements
CVMFS
Software containers
Dirac pilot
Proxy
Pilot communicates end of job
CC-IN2P3
Dirac server
Database
-Description of snakemake job, with input files, marked as DONE
Agents
Proxy
"Your" machine
Personal Certificate
Check job status
Check status
Job status


Storage elements
Files: Raw data, calibrations, run start/end and outputs **on disk**
NIKHEF
Rucio server
Database
Info on Raw data, Calibrations, Run start/end files and outputs
Daemons
-Maintain long term replication rules
-Short term replication to disk
List of files
Output files
"Your" machine
Personal Certificate
Download files
Computing Elements
CVMFS
CC-IN2P3
Dirac server
Database
-Description of snakemake job, with input files, marked as DONE
Agents

# Examples

# Dirac’s Job Description Language (JDL)

- A file describing what to execute on the grid
- Contains following information (possibly more, or less):
  - Executable to run
  - Executable parameters
  - WN requirements (e.g. cores, memory)
  - Max runtime requirement (**in s@HS06**)
  - File to redirect stdout & stderr
  - **In- & output sandbox:** only “direct” data exchange with UI. <10MB

For files above 10MB, need to set up up/download inside job itself (more on that later)

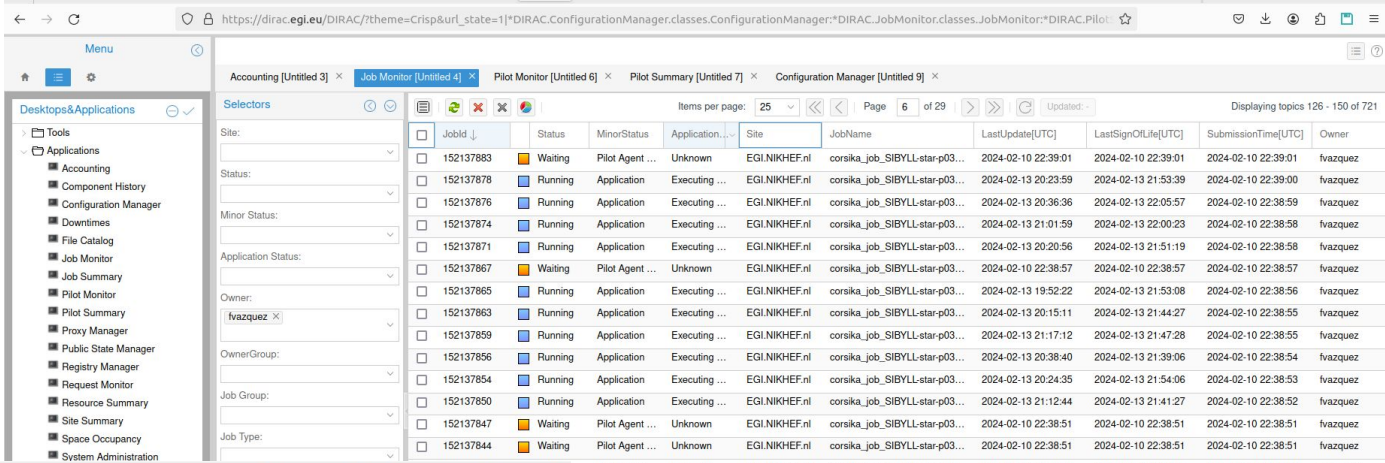
Job.jdl

```
Executable="runstuff.sh inputfile.txt";
StdOutput="stdout.txt";
StdError="stderr.txt";
InputSandbox={"runstuff.sh","inputfile.txt"};
OutputSandbox={"stdout.txt","stderr.txt"};
CPUTime=10000;
NumberOfProcessors=4;
```

In practice, let the Dirac Python API generate the JDL for you

# DIRAC: Dashboard

Web interface for monitoring and logging your jobs




https://dirac.egi.eu/DIRAC/

# RUCIO command examples

**$ rucio list-dids CORSIKA_testing: --filter type=DATASET**

```
+-------------------------------------------------------------------------+-----------------+
| SCOPE:NAME                                                              | [DID TYPE]      |
|-------------------------------------------------------------------------+-----------------|
| CORSIKA_testing:SIBYLL_DefaultAtmo_TeV_low_C_20240328-1103              | DIDType.DATASET |
| CORSIKA_testing:SIBYLL_DefaultAtmo_TeV_low_p_20240328-1536              | DIDType.DATASET |
| CORSIKA_testing:SIBYLL_DefaultAtmo_TeV_low_p_20240328-1552              | DIDType.DATASET |
| CORSIKA_testing:SIBYLL_DefaultAtmo_TeV_low_p_20240402-0204              | DIDType.DATASET |
| CORSIKA_testing:SIBYLL-star-p03_DefaultAtmo_EeV_p_20240402-1553         | DIDType.DATASET |
| CORSIKA_testing:SIBYLL-star-p03_DefaultAtmo_EeV_p_20240404-1732         | DIDType.DATASET |
| CORSIKA_testing:SIBYLL-star-p03_DefaultAtmo_EeV_p_20240405-1357         | DIDType.DATASET |
| CORSIKA_testing:SIBYLL-star-p03_DefaultAtmo_TeV_low_p_20240406-1926     | DIDType.DATASET |
| CORSIKA_testing:SIBYLL-star-p03_DefaultAtmo_TeV_low_p_20240408-0051     | DIDType.DATASET |
| CORSIKA_testing:SIBYLL-star-p03_DefaultAtmo_TeV_low_p_20240408-0052     | DIDType.DATASET |
+-------------------------------------------------------------------------+-----------------+
```

**$ rucio list-dids CORSIKA_testing: --filter Production=TeV_low,NumberShowers.gte=2000**

```
+-----------------------------------------------------------+--------------+
| SCOPE:NAME                                                | [DID TYPE]   |
|-----------------------------------------------------------+--------------|
| CORSIKA_testing:SIBYLL_DefaultAtmo_TeV_low_p_20240328-1536 |              |
| CORSIKA_testing:SIBYLL_DefaultAtmo_TeV_low_p_20240328-1552 |              |
| CORSIKA_testing:SIBYLL_DefaultAtmo_TeV_low_p_20240402-0204 |              |
+-----------------------------------------------------------+--------------+
```

**$ rucio list-files CORSIKA_testing:SIBYLL-star-p03_DefaultAtmo_TeV_low_p_20240406-1926**

```
+-------------------------------------------------------------------------------------+--------------------------------------+------------+------------+----------+
| SCOPE:NAME                                                                          | GUID                                 | ADLER32    | FILESIZE   | EVENTS   |
|-------------------------------------------------------------------------------------+--------------------------------------+------------+------------+----------|
| CORSIKA_testing:DAT_SIBYLL-star-p03_DefaultAtmo_TeV_low_p_20240406-1926_000500.gz   | A77054B3-2737-43D6-9148-B82E5FB44C9F | ad:62b4e0f6 | 66.120 kB  |          |
| CORSIKA_testing:LOG_SIBYLL-star-p03_DefaultAtmo_TeV_low_p_20240406-1926_000500.tar.gz | 1961FE8B-4B25-4558-98FC-A51AB18429C7 | ad:c71c6de7 | 37.762 kB  |          |
+-------------------------------------------------------------------------------------+--------------------------------------+------------+------------+----------+
```

**$ rucio download CORSIKA_testing:SIBYLL-star-p03_DefaultAtmo_TeV_low_p_20240406-1926**

```
---------------------------------
Download summary
---------------------------------------
DID CORSIKA_testing:SIBYLL-star-p03_DefaultAtmo_TeV_low_p_20240406-1926
Total files (DID):                      2
Total files (filtered):                 2
Downloaded files:                       2
Files already found locally:            0
Files that cannot be downloaded:        0
```

# RUCIO setup

- Check https://wiki.km3net.de/index.php/Rucio_data_management
- You can also check the following notebook for examples: https://rucio.pages.km3net.de/rucio-documentation/notebooks/Tutorial/
- And you can check the official documentation: https://rucio.github.io/documentation/

# Dirac and running on the grid

- Check Richard R.'s repository for a local Dirac setup https://git.km3net.de/rrandriatoamanana/grid-on-client
- Examples of grid job submission (Tests/Corsika and Run-Based Data Processing): https://git.km3net.de/workflow-management/grid-tools, https://git.km3net.de/rucio/grid-snakemake

```python
#!/usr/bin/env python

from DIRAC.Core.Base import Script
script_pcl = Script.initialize()

from DIRAC.Interfaces.API.Job import Job
from DIRAC.Interfaces.API.Dirac import Dirac

dirac = Dirac()
j = Job()

j.setInputSandbox(['runstuff.sh', 'inputfile.txt'])
j.setOutputSandbox(['stdout.txt', 'stderr.txt'])
j.setCPUTime(10000)
j.setNumberOfProcessors(4)
j.setExecutable('runstuff.sh', arguments='inputfile.txt')

jobSubmission = dirac.submitJob(j) # mode='local'
print( jobSubmission['Value'] )
```

# Glossary

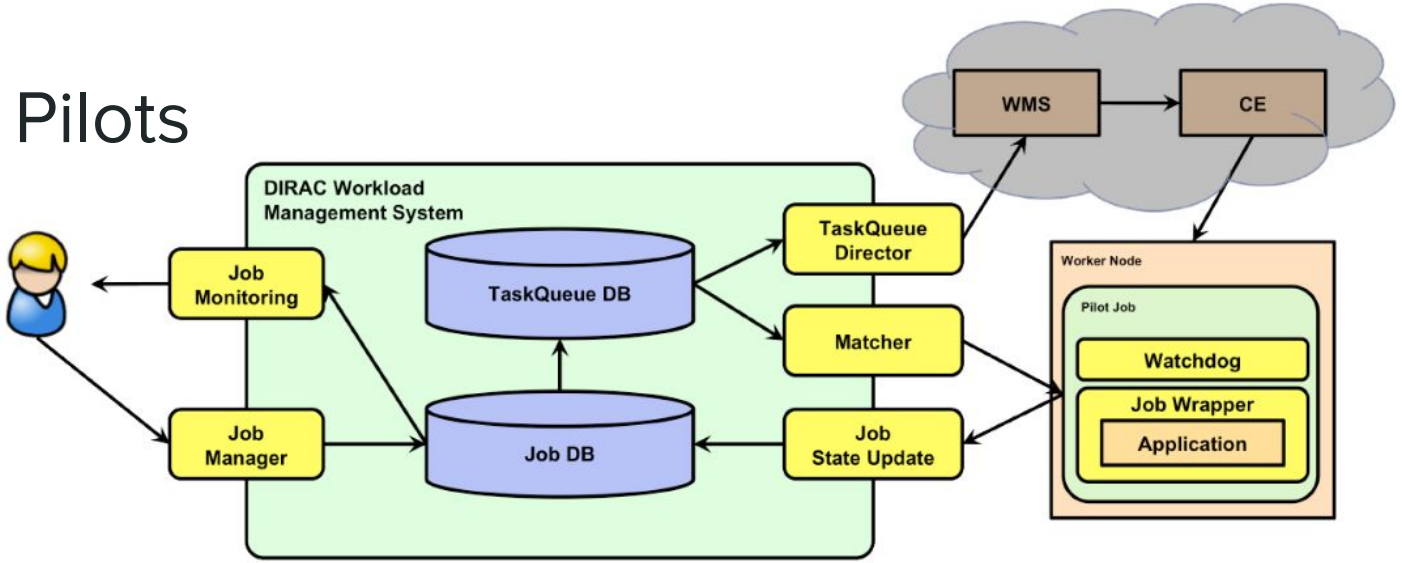
# Glossary : Infrastructure

- **UI (User Interface):** your access point to the Grid. From here you submit the jobs and retrieve logging information.
- **CE (Computing Element):** is the interface to the cluster. Accepts jobs via a batch system and dispatches them to a collection of site Worker Nodes. The jobs are submitted by the users either through a WMS or directly.
- **WN (Worker Node):** the machines that do the actual work & execute jobs.
- **Middleware:** software that mediates between users and Grid resources. Covers a variety of roles (DIRAC, Globus, gLite, dCache...)

# Glossary : Jobs

- **WMS (Workload Management System):** distributes and manages tasks across computing and storage Grid resources.
- **Job:** a program that will run somewhere on a Grid machine.
- **JDL (Job Description Language):** describes job, parameters and requirements.
- **In/Output Sandbox:** Input Sandbox defines any names of files to be uploaded. Output Sandbox contains filenames of data to retrieve after the job is done. Limited to <10MB files.
- **Pilot:** job that runs on a WN to set up the environment and fetch jobs from a central task queue as required (see DIRAC, PanDA, PiCaS)

# DIRAC Pilots




Pilots are wrapper jobs sent by DIRAC to Worker Nodes. They setup the environment and then request “tasks” (jobs) from DIRAC server until they expire.

- Minimize resources spent setting up
- Simplifies discovery of free resources (pull vs push system)
- Allows advanced workflow management

# Monitoring

Dashboard for short term monitoring, eg. EGI DIRAC server: https://dirac.egi.eu/DIRAC/

EGI accounting portal for long term use, eg. Netherlands: https://accounting.egi.eu/egi/country/Netherlands/