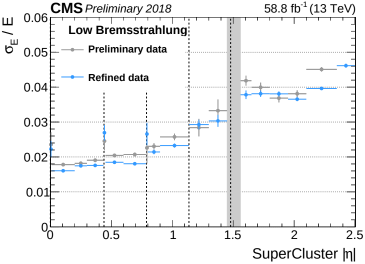
# Automatic data processing for prompt calibration of the CMS ECAL

**Simone Pigazzini**
On behalf of the CMS collaboration

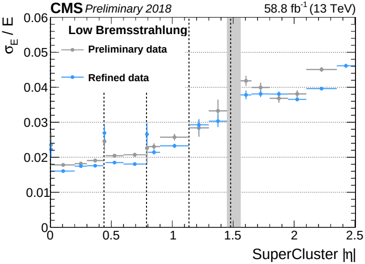
**ACAT 2022**

# The problem, breifly.



- **Radiation damage** requires a **constant monitoring and correction** of the detector response to maintain top performance for physics measurements.
- The ultimate performance $\sim$ 40% (**= 1 year of work**) better than "prompt" one. **Tracking the response evolution over time** being the main challenge.

# The problem, breifly.

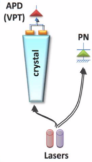


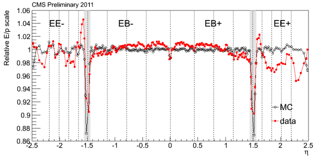
- **Radiation damage** requires a **constant monitoring and correction** of the detector response to maintain top performance for physics measurements.
- The ultimate performance ∼ 40% (**= 1 year of work**) better than "prompt" one. **Tracking the response evolution over time** being the main challenge.

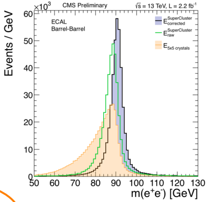
- **Freeing up resources is crucial to tackle recurring and new challenges.**
- The **main focus in Run3 and beyond** will be to improve the mitigation of **noise** and **pile-up** related effects (more in Davide's poster and Abhirami's talk )

# Context: e/$\gamma$ reconstruction in CMS

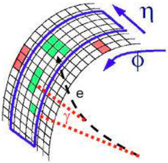
- alignment.
- signal time delay adjustment.

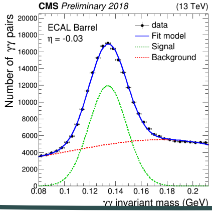


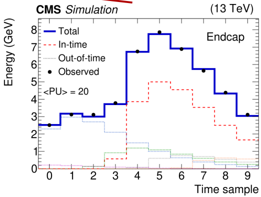




$$E_{e,\gamma} = \Sigma_i [L_i(t) \cdot C_i(t) \cdot A_i(t)] \cdot G(\eta) \cdot F_{e,\gamma}$$

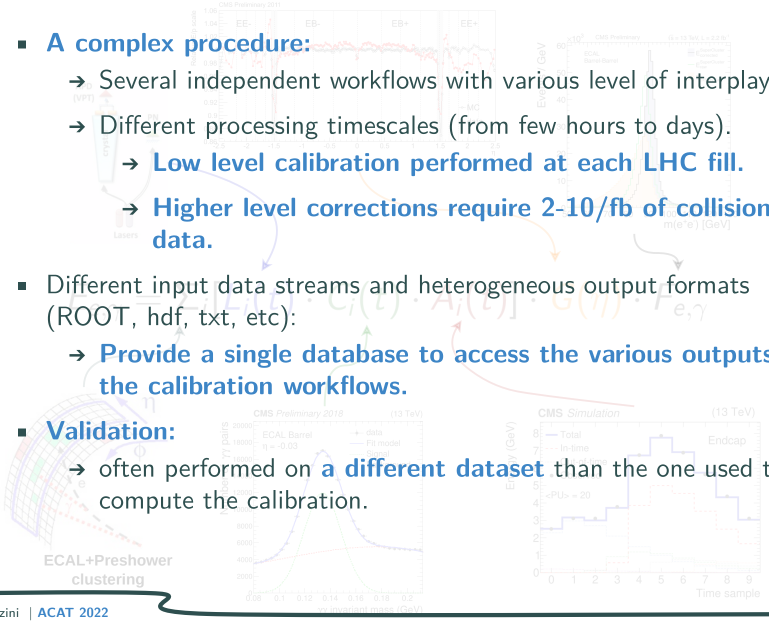






# Context: e/$\gamma$ reconstruction in CMS

- **A complex procedure:**
  - → Several independent workflows with various level of interplay.
  - → Different processing timescales (from few hours to days).
    - → **Low level calibration performed at each LHC fill.**
    - → **Higher level corrections require 2-10/fb of collision data.**
- Different input data streams and heterogeneous output formats (ROOT, hdf, txt, etc):
  - → **Provide a single database to access the various outputs of the calibration workflows.**
- **Validation:**
  - → often performed on **a different dataset** than the one used to compute the calibration.



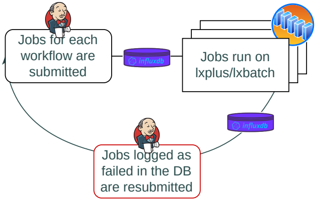
# Context: CMS prompt data processing.

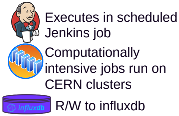
- Data acquired by the CMS experiment is streamed from the local farm to **Tier-0** computing center located at CERN.
- Different data streams are processed at different stages, ultimately producing the "prompt-reconstructions" that is **used in physics analysis**.
- The prompt reconstruction is usually started **48h** after data-taking.
- Before that the **Prompt Calibration Loop** ref ↗ provide a system to compute a new calibration using a reduced/dedicated set of events.
  - ➔ **This new system is inspired to the PCL model, extending its functionalities to allow for greater flexibility.**

# System overview

- **The requirements**:
  - **A system based on industry tools** (less development, easier maintenance).
  - Re-use existing custom code.
  - Process data "live" as well as bulk reprocessing at a later stage.
  - Enforce traceability of jobs configuration (which cfg did I use?).
- **The solution:**
  - → A framework of finite state machine implemented through **Jenkins**, **Influxdb** and **Grafana** for monitoring.
  - → Deployed with the **Openshift** instance provided by CERN-IT.
  - → A **small python package** ↗ to provide the **interface** between the CMS ecosystem, the user jobs and the framework.

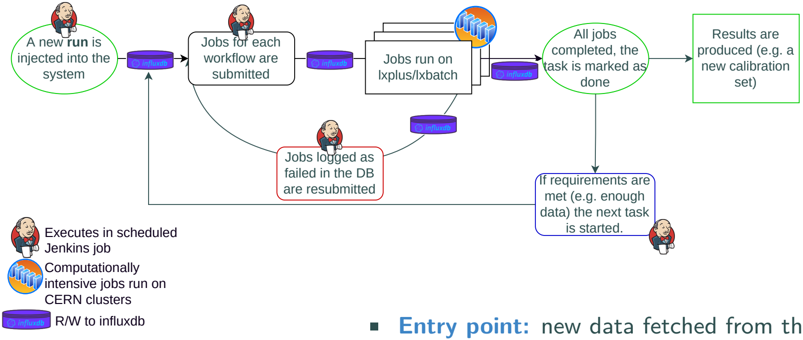
# Processing flow


Jobs for each workflow are submitted
influxdb
Jobs run on lxplus/lxbatch
influxdb
Jobs logged as failed in the DB are resubmitted


Executes in scheduled Jenkins job
Computationally intensive jobs run on CERN clusters
influxdb
R/W to influxdb

- Each **workflow** can run multiple jobs in parallel (local center, GRID, ...).
- Each **job** logs its status independently.
- **Influxdb** does not have fixed tables, allowing each workflow to write for each job a set of **custom fields**.

# Processing flow



A new **run** is injected into the system

Jobs for each workflow are submitted

Jobs run on lxplus/lxbatch

All jobs completed, the task is marked as done

Results are produced (e.g. a new calibration set)

Jobs logged as failed in the DB are resubmitted

If requirements are met (e.g. enough data) the next task is started.

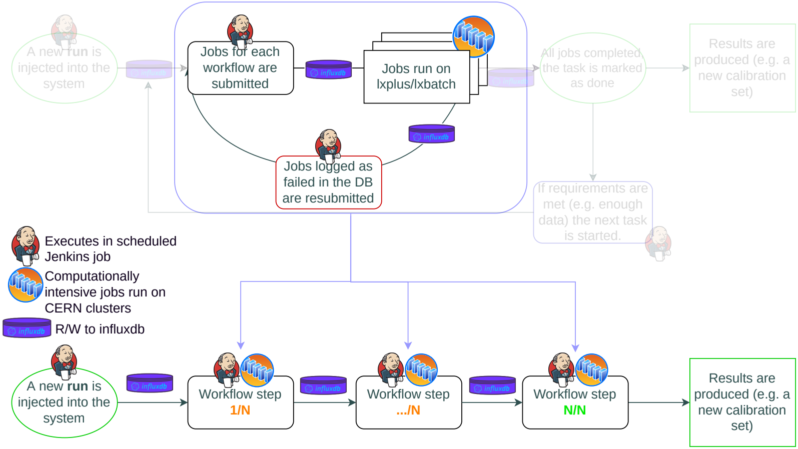
Executes in scheduled Jenkins job

Computationally intensive jobs run on CERN clusters

R/W to influxdb

- **Entry point:** new data fetched from the CMS run registry.
- **Polling started by Jenkins periodically** (1 hour)
- Results are usually collected at the next pass.

# Processing flow



A new **run** is injected into the system

Jobs for each workflow are submitted

Jobs run on lxplus/lxbatch

All jobs completed the task is marked as done

Results are produced (e.g. a new calibration set)

Jobs logged as failed in the DB are resubmitted

If requirements are met (e.g. enough data) the next task is started.

Executes in scheduled Jenkins job

Computationally intensive jobs run on CERN clusters

influxdb R/W to influxdb

A new **run** is injected into the system

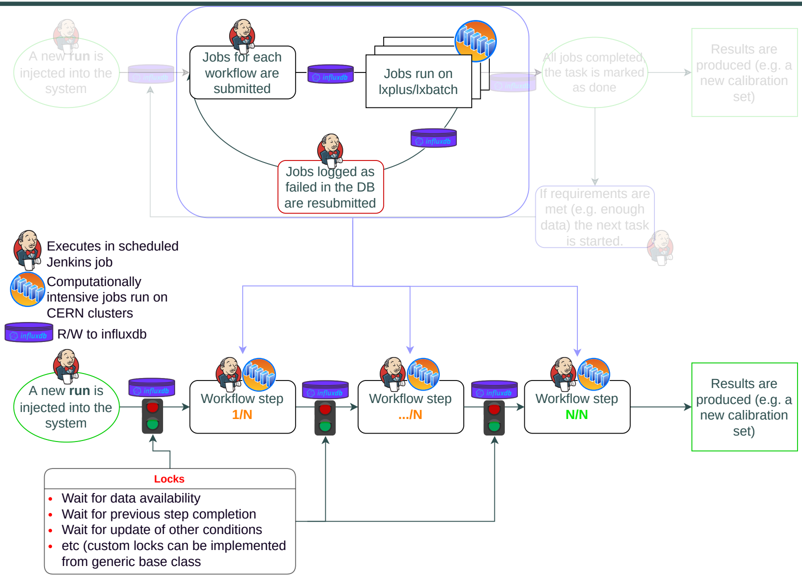
Workflow step **1/N**

Workflow step **.../N**

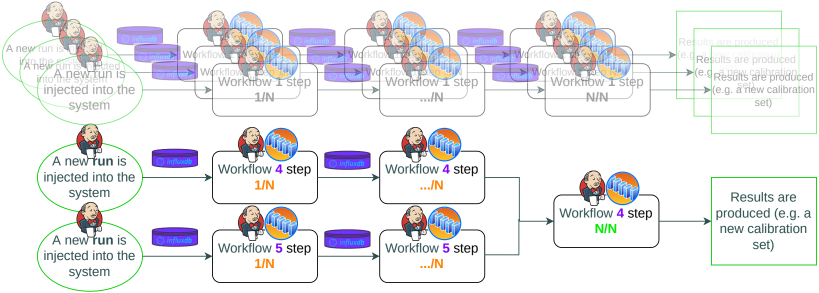
Workflow step **N/N**

Results are produced (e.g. a new calibration set)

# Processing flow

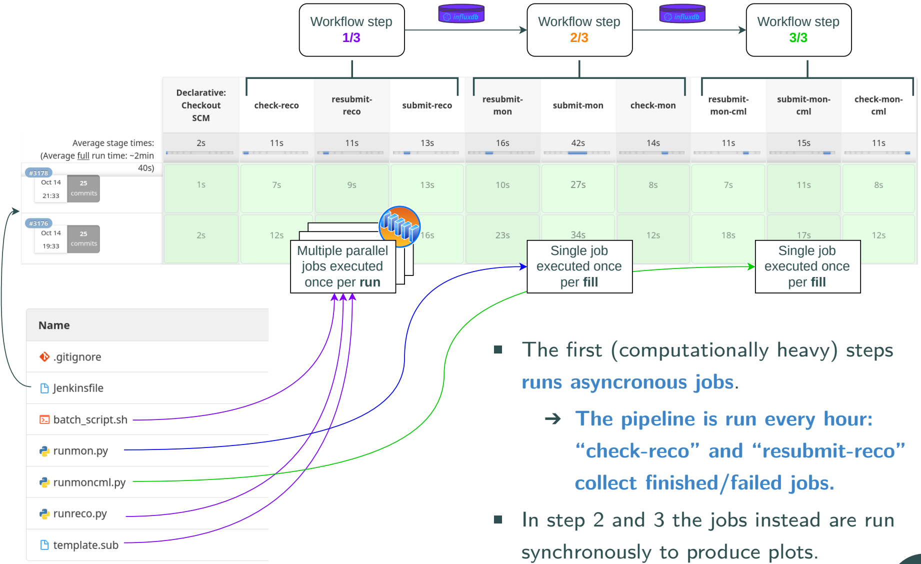

A new run is injected into the system
influxdb
Jobs for each workflow are submitted
influxdb
Jobs run on lxplus/lxbatch
influxdb
All jobs completed, the task is marked as done
Results are produced (e.g. a new calibration set)
influxdb
Jobs logged as failed in the DB are resubmitted
If requirements are met (e.g. enough data) the next task is started.
Executes in scheduled Jenkins job
Computationally intensive jobs run on CERN clusters
influxdb
R/W to influxdb
A new run is injected into the system
influxdb
Workflow step 1/N
influxdb
Workflow step .../N
influxdb
Workflow step N/N
Results are produced (e.g. a new calibration set)
Locks
Wait for data availability
Wait for previous step completion
Wait for update of other conditions
etc (custom locks can be implemented from generic base class

# Processing flow


A new run is injected into the system
influxdb
Workflow 1 step 1/N
Workflow 1 step .../N
Workflow 1 step N/N
Results are produced (e.g. a new calibration set)
A new run is injected into the system
Workflow 4 step 1/N
Workflow 4 step .../N
A new run is injected into the system
Workflow 5 step 1/N
Workflow 5 step .../N
Workflow 4 step N/N
Results are produced (e.g. a new calibration set)

- **Workflow chains runs independently**, started periodically by Jenkins.
- Chians can be joined/split to feed multiple outputs to a single subsequent workflow.
  - ➔ **Convenient for validation (test vs ref)**
  - ➔ Similar to DAG but without the requirement that different branches runs synchronously.

# Workflow example



Workflow step 1/3
influxdb
Workflow step 2/3
influxdb
Workflow step 3/3
Multiple parallel jobs executed once per **run**
Single job executed once per **fill**
Single job executed once per **fill**

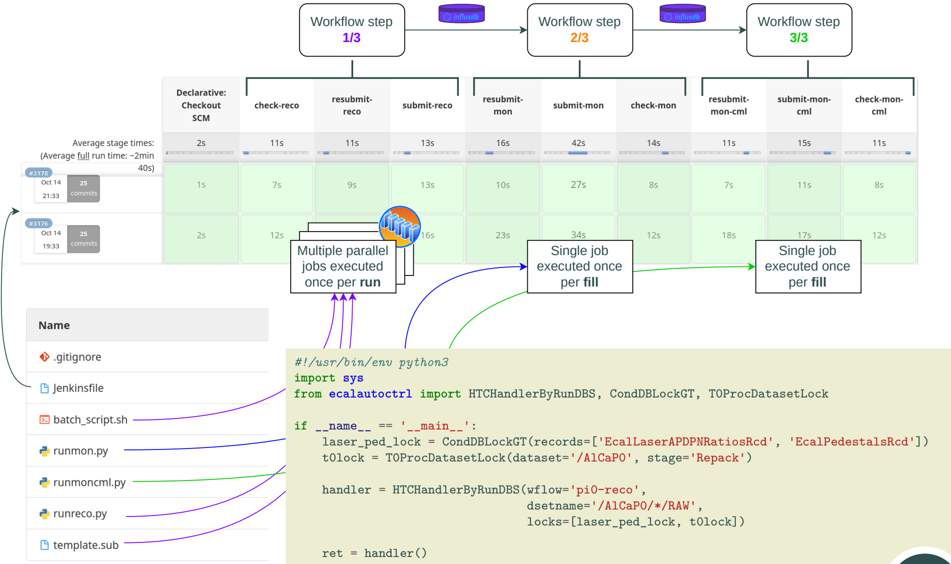
| | Declarative: Checkout SCM | check-reco | resubmit-reco | submit-reco | resubmit-mon | submit-mon | check-mon | resubmit-mon-cml | submit-mon-cml | check-mon-cml |
|---|---|---|---|---|---|---|---|---|---|---|
| Average stage times:<br>(Average full run time: ~2min 40s) | 2s | 11s | 11s | 13s | 16s | 42s | 14s | 11s | 15s | 11s |
| #3178 Oct 14 21:33 25 commits | 1s | 7s | 9s | 13s | 10s | 27s | 8s | 7s | 11s | 8s |
| #3176 Oct 14 19:33 25 commits | 2s | 12s | | 16s | 23s | 34s | 12s | 18s | 17s | 12s |

**Name**

- .gitignore
- Jenkinsfile
- batch_script.sh
- runmon.py
- runmoncml.py
- runreco.py
- template.sub

- The first (computationally heavy) steps **runs asyncronous jobs.**
  - ➔ **The pipeline is run every hour: "check-reco" and "resubmit-reco" collect finished/failed jobs.**
- In step 2 and 3 the jobs instead are run synchronously to produce plots.

# Workflow example



Workflow step **1/3** → Workflow step **2/3** → Workflow step **3/3**

| | Declarative: Checkout SCM | check-reco | resubmit-reco | submit-reco | resubmit-mon | submit-mon | check-mon | resubmit-mon-cml | submit-mon-cml | check-mon-cml |
|---|---|---|---|---|---|---|---|---|---|---|
| Average stage times:<br>(Average full run time: ~2min 40s) | 2s | 11s | 11s | 13s | 16s | 42s | 14s | 11s | 15s | 11s |
| #3178 Oct 14 21:33 25 commits | 1s | 7s | 9s | 13s | 10s | 27s | 8s | 7s | 11s | 8s |
| #3176 Oct 14 19:33 25 commits | 2s | 12s | | 16s | 23s | 34s | 12s | 18s | 17s | 12s |

Multiple parallel jobs executed once per **run**

Single job executed once per **fill**

Single job executed once per **fill**

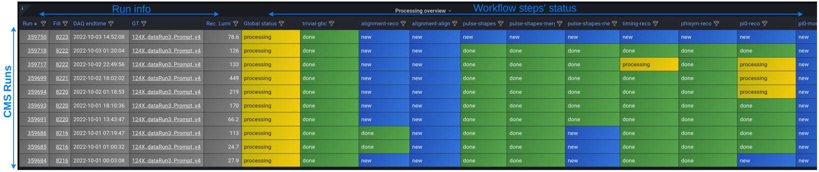
**Name**

- .gitignore
- Jenkinsfile
- batch_script.sh
- runmon.py
- runmoncml.py
- runreco.py
- template.sub

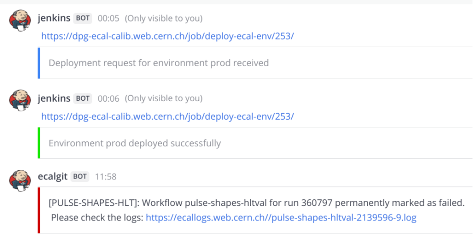
```python
#!/usr/bin/env python3
import sys
from ecalautoctrl import HTCHandlerByRunDBS, CondDBLockGT, T0ProcDatasetLock

if __name__ == '__main__':
    laser_ped_lock = CondDBLockGT(records=['EcalLaserAPDPNRatiosRcd', 'EcalPedestalsRcd'])
    t0lock = T0ProcDatasetLock(dataset='/AlCaP0', stage='Repack')

    handler = HTCHandlerByRunDBS(wflow='pi0-reco',
                                 dsetname='/AlCaP0/*/RAW',
                                 locks=[laser_ped_lock, t0lock])

    ret = handler()

    sys.exit(ret)
```

# Monitoring

## Data processing overview page



| Run | Fill | DAQ endtime | GT | Rec. Lumi | Global status | trivial-ghc | alignment-reco | alignment-align | pulse-shapes | pulse-shapes-merg | pulse-shapes-me | timing-reco | phisym-reco | pi0-reco | pi0-mon |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 359750 | 8223 | 2022-10-03 14:52:08 | 124X_dataRun3_Prompt_v4 | 78.6 | processing | done | new | new | new | new | new | new | new | new | new |
| 359718 | 8222 | 2022-10-03 01:20:04 | 124X_dataRun3_Prompt_v4 | 126 | processing | done | new | new | done | done | done | done | done | done | new |
| 359717 | 8222 | 2022-10-02 22:49:56 | 124X_dataRun3_Prompt_v4 | 133 | processing | done | new | new | done | done | done | processing | done | processing | new |
| 359699 | 8221 | 2022-10-02 18:02:02 | 124X_dataRun3_Prompt_v4 | 449 | processing | done | new | new | done | done | done | done | done | processing | new |
| 359694 | 8220 | 2022-10-02 01:18:53 | 124X_dataRun3_Prompt_v4 | 219 | processing | done | new | new | done | done | done | done | done | processing | new |
| 359693 | 8220 | 2022-10-01 18:10:36 | 124X_dataRun3_Prompt_v4 | 170 | processing | done | new | new | done | done | done | done | done | done | new |
| 359691 | 8220 | 2022-10-01 13:43:47 | 124X_dataRun3_Prompt_v4 | 66.2 | processing | done | new | new | done | done | done | done | done | done | new |
| 359686 | 8216 | 2022-10-01 07:19:47 | 124X_dataRun3_Prompt_v4 | 113 | processing | done | done | new | done | done | new | done | done | done | new |
| 359685 | 8216 | 2022-10-01 01:00:32 | 124X_dataRun3_Prompt_v4 | 24.7 | processing | done | done | new | done | done | new | done | done | done | new |
| 359684 | 8216 | 2022-10-01 00:03:08 | 124X_dataRun3_Prompt_v4 | 27.9 | processing | done | new | new | done | done | new | done | done | new | new |



jenkins BOT 00:05 (Only visible to you)

https://dpg-ecal-calib.web.cern.ch/job/deploy-ecal-env/253/

Deployment request for environment prod received

jenkins BOT 00:06 (Only visible to you)

https://dpg-ecal-calib.web.cern.ch/job/deploy-ecal-env/253/

Environment prod deployed successfully

ecalgit BOT 11:58

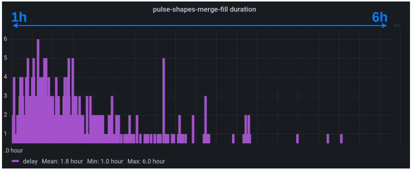
[PULSE-SHAPES-HLT]: Workflow pulse-shapes-hltval for run 360797 permanently marked as failed. Please check the logs: https://ecallogs.web.cern.ch//pulse-shapes-hltval-2139596-9.log

Monitoring of several system parameters.

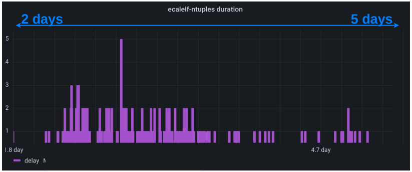
- → Monitoring of resources: **EOS and AFS quotas**.
- → Execution monitoring through webpages and dedicated **Mattermost alerts and slash commands**.
- → Development and debug in **“mirror” system**: execute workflows from HEAD of each branch rather than production tag.

# Review of the operation in 2022

- **The 2022 LHC p-p run was supposed to be a commissioning period.**
  - ➔ The update of new calibrations has not been performed automatically.
  - ➔ Integration of all workflows still ongoing.
  - ➔ **The system had no downtime during 2022 and has been used to successfully monitor the evolution of the ECAL response as well as produce low level calibration factors.**
- **We are already in the process of integrating more CMS subsystems into the framework.**
- **Latency in workflows completion has been mostly driven by external factors** (availability of input: data, other detector calibration factors, etc).



Example of workflow running on RAW data



Example of workflow running on reconstructed, pre-calibrated data.

# Summary and outlook

- We have developed a **framework** ↗ to integrate **independent workflows** into a **coherent and automatic** data processing system.
- **The framework concepts could serve equally well other projects:**
  - → Beside the interaction with the CMS ecosystems (run register, conditions database, etc), the concept of a system based on **Jenkins+influxdb+python** for a **flexible and continuous** data processing can be easily exported to other projects.
- Planned developments:
  - → **Containers** naturally fit the framework structure.
  - → Currently using the CMS software environment on a shared file system (AFS+EOS+CVMFS). + job-dependent software loaded from git repos at runtime.
  - → More monitoring, more continuos testing, ...

# Additional material

# The tools

- **Jenkins** : an open source automation server. It helps automate the parts of software development related to building, testing, and deploying, facilitating continuous integration and continuous delivery. **In this context it is used to automate the execution of data processing jobs.**
- **influxdb** : a NoSQL database. Optimized for time series.
- **Grafana** : a powerful database visualization tool.
- **Openshift:** a family of containerization software products developed by Red Hat.