# Managing workloads and workflows with DIRAC for SWIFT-HEP

Update

Janusz Martyniak, Daniela Bauer & Simon Fayer for Imperial College

Imperial College London

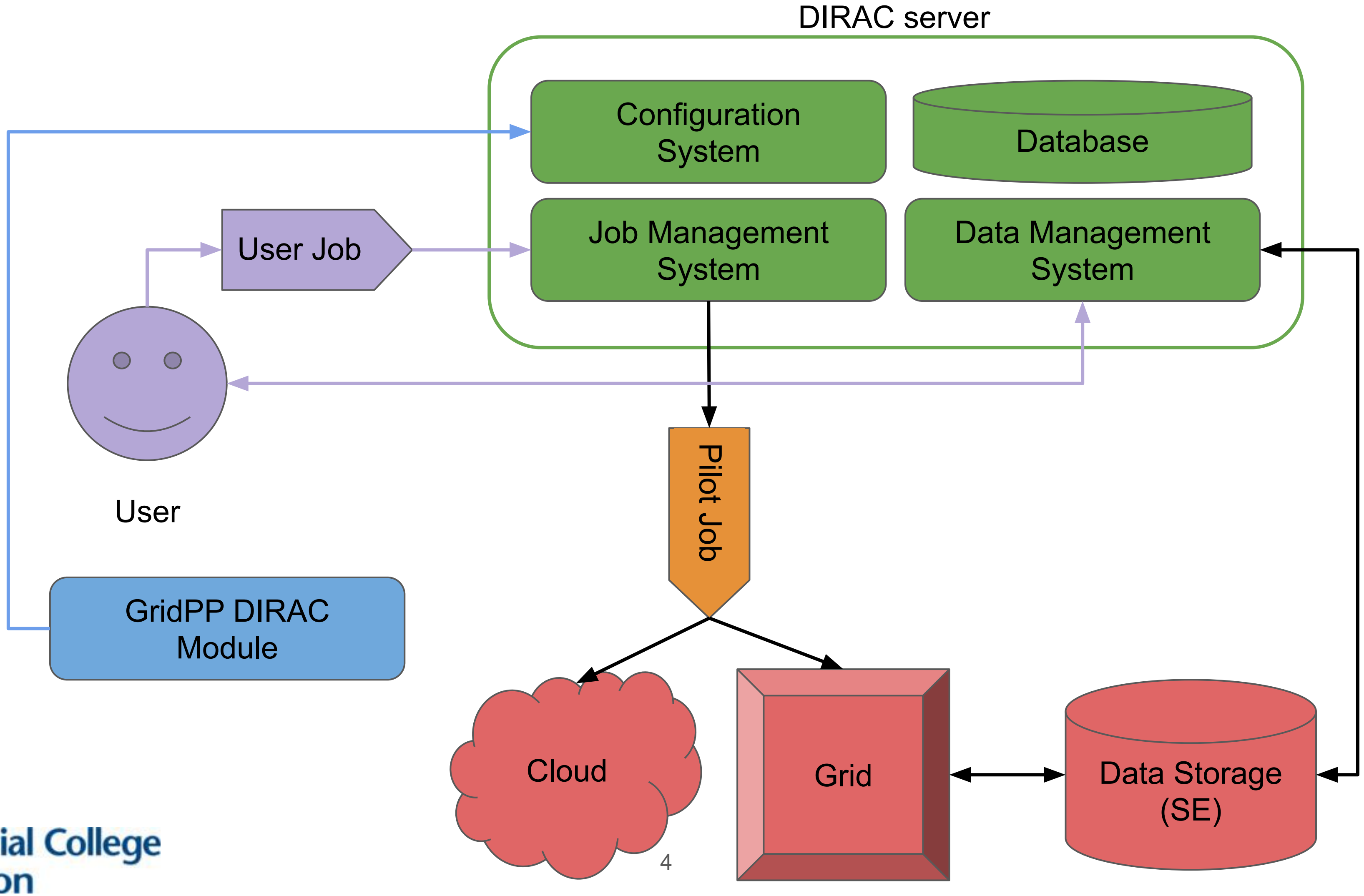
# Overview

- Recap: What is DIRAC and how does it fit in within the SWIFT-HEP remit?
- Status

# DIRAC

- DIRAC is a software originally developed by LHCb. The DIRAC consortium was founded in 2014 to enable adoption by other communities. (The UK is a member of this consortium via Imperial College.)
- DIRAC comprises of:
  - Workload Management System
  - File Catalog/Data Management System
  - Workflow Management System
  - Documentation: https://dirac.readthedocs.io/en/latest/
- Provides a standardised user interface to multiple compute (grid & cloud) and storage resources.
  - It has always had an auto stage-in for input data  from other sites ("the original data lake")
- Written in Python (for Linux)
  - Open Source: https://github.com/DIRACGrid/DIRAC

# DIRAC Schematic (for reference)

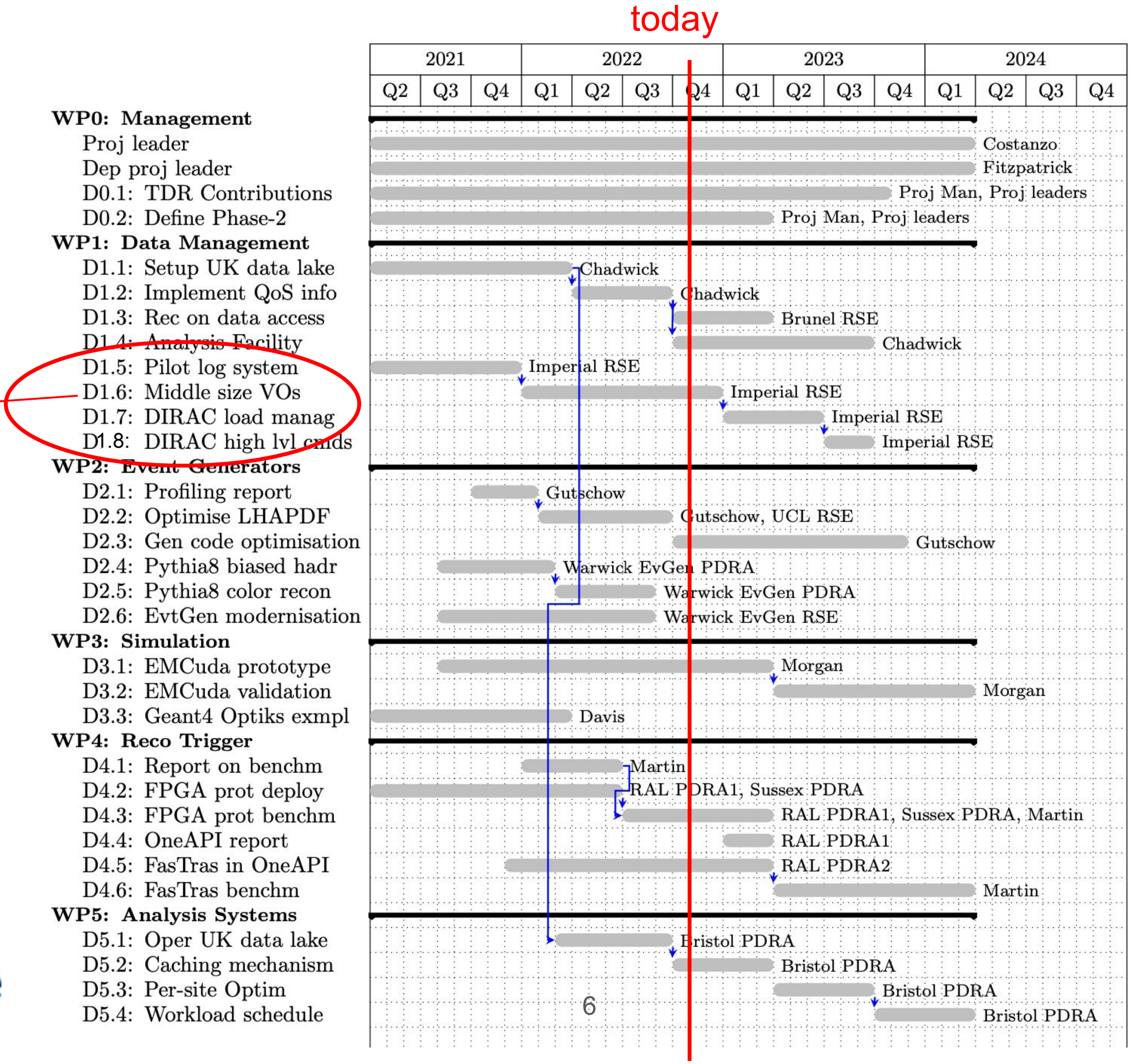


Imperial College London

# DIRAC Users

- Comprised of HEP and non-HEP communities:
  - HEP: **LHCb**, NA62, Belle2, ILC/Calice, mu3e
  - Neutrinos: T2K, HyperK**,** JUNO, SoLid
  - Phenomenology: Pheno (Durham)
  - Dark Matter: LZ
  - Astronomy: CTA, LSST, Auger
  - Biological sciences
- Swift-HEP work is on the **DIRAC core software**, which is used by all communities.
- GridPP provides a DIRAC instance as a service to the non-LHC communities it supports.
- DIRAC is very much a here and now project, but it needs adapting for the future.
  - SWIFT-HEP only represents a subset of ongoing work.

# SWIFT-HEP: In the grand scheme of things



Workflow Management

# WP1.5 Pilot Log System

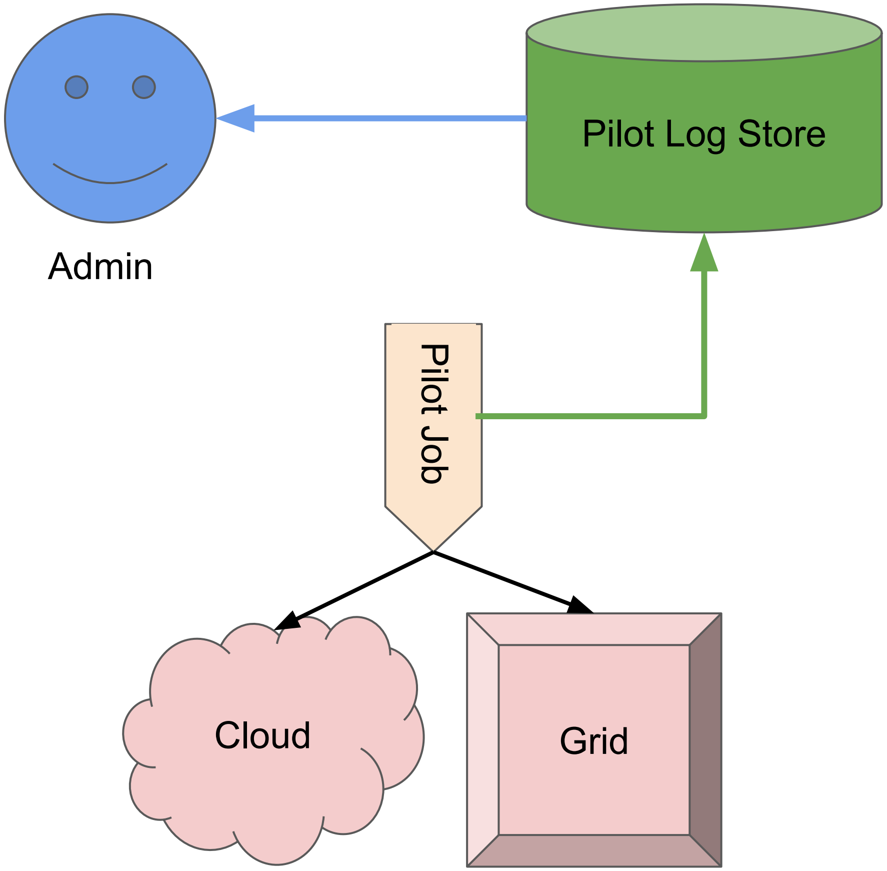
- Pilot jobs:
  - Check the worker-node environment
  - Can stage required input/output files
  - Start and supervise the user job (record memory usage, efficiency, etc.)
- The pilot logs are crucial for diagnosing problems.

- Pilot job logs are stored in an technology dependent way at the execution resources
- Retention policies vary by technology and site:
  - Some logs only kept while job running!
  - Others kept 3 days - 1 month depending on configuration.
  - Transient (cloud) resources may not have space suitable for archiving these logs.
- Log can be completely lost in cases where job crashes (i.e. exceeding batch limits).
- Retaining pilot job logs in a reliable, resource independent manner was identified as a high priority issue by LHCb and other communities.

# WP1.5 Pilot Log System: Implementation

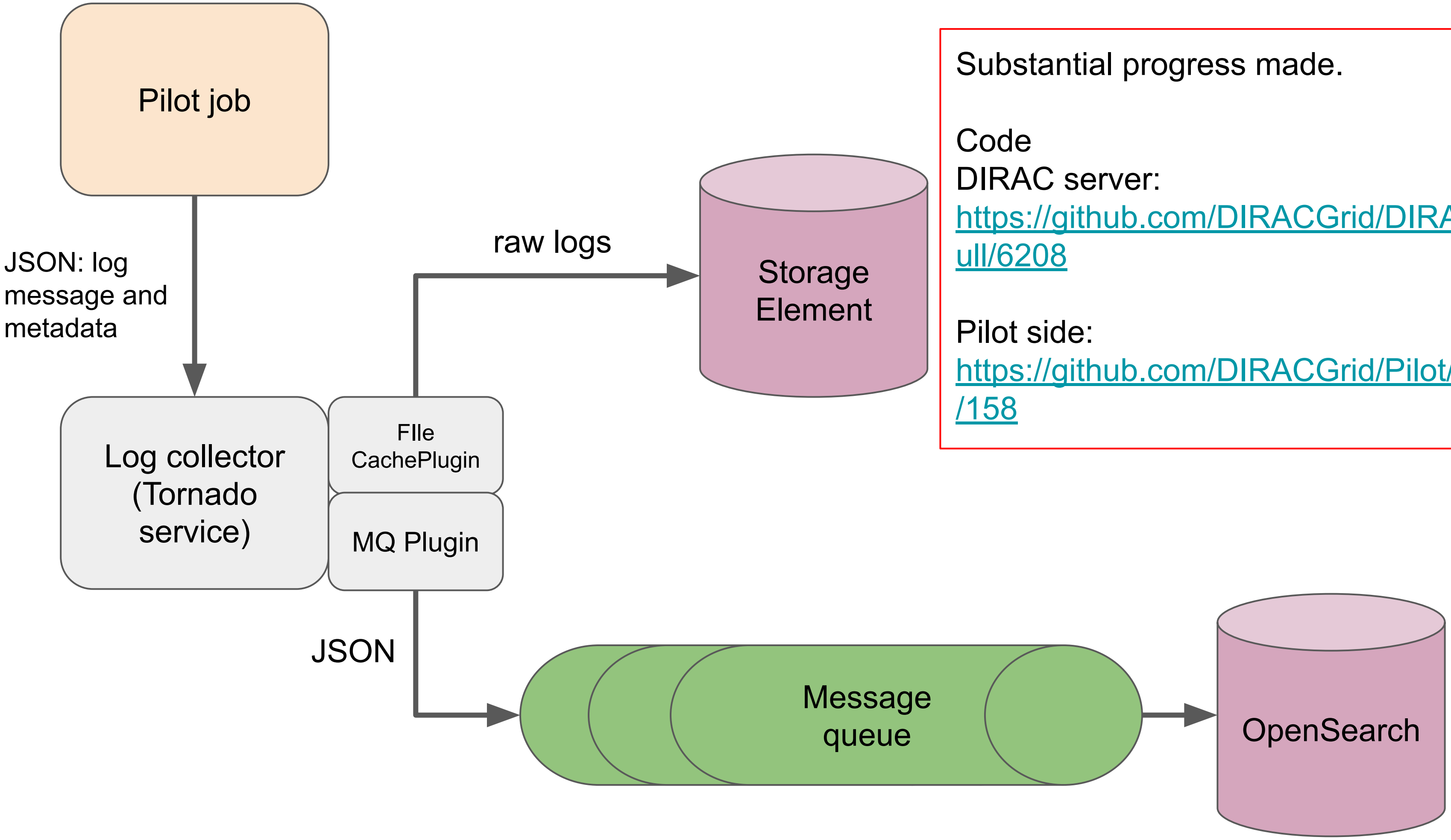
- Develop a central pilot log store and allow the pilot jobs to write logs there directly, therefore removing any resource dependencies.

- At peak times this service needs to cope with a large amount of traffic in a fault-tolerant way.



Imperial College London

# Pilot Log System Status - technical



Substantial progress made.

Code
DIRAC server: https://github.com/DIRACGrid/DIRAC/pull/6208

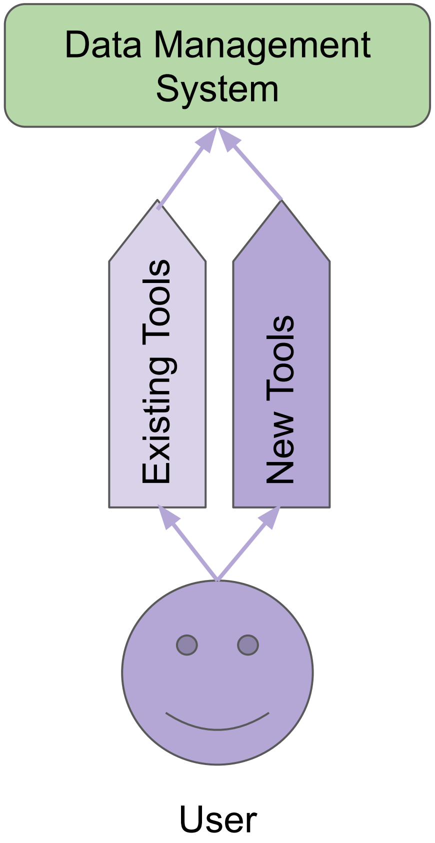
Pilot side: https://github.com/DIRACGrid/Pilot/pull/158

Imperial College London

# WP 1.8 DIRAC High Level Commands

- Target is medium size communities without (much) dedicated computing support.
- These communities often already use the DIRAC File Catalogue and basic DIRAC data management tools, so the threshold for adoption is quite low.

- Develop tools for the most common use cases and make them available to all users as part of DIRAC, e.g.
  - Importing existing data into the file catalogue.
  - Copying directories from one storage element to another.



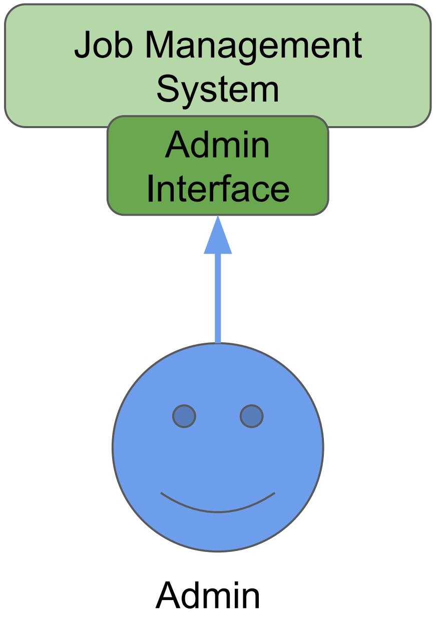
Imperial College London

# WP 1.8 DIRAC High Level Commands

- Turns out we aren’t the only ones whose users would like these facilities.
- Teamed up with EGI to merge the various bits of code and integrate them into core DIRAC
- https://github.com/DIRACGrid/DIRAC/pull/6403
- Close to being merged.

# WP1.7 DIRAC Workload Management - Plans

- Current load management system fairly basic:
  - Jobs bound to sites quite early in submission process.
  - Target site immutable after submission and binding.
- Not flexible enough for large infrastructures, e.g.:
  - Unexpected changes in target site capacities (both up and down).
  - Misunderstandings lead to users submitting large batches of jobs to unsuitable target.



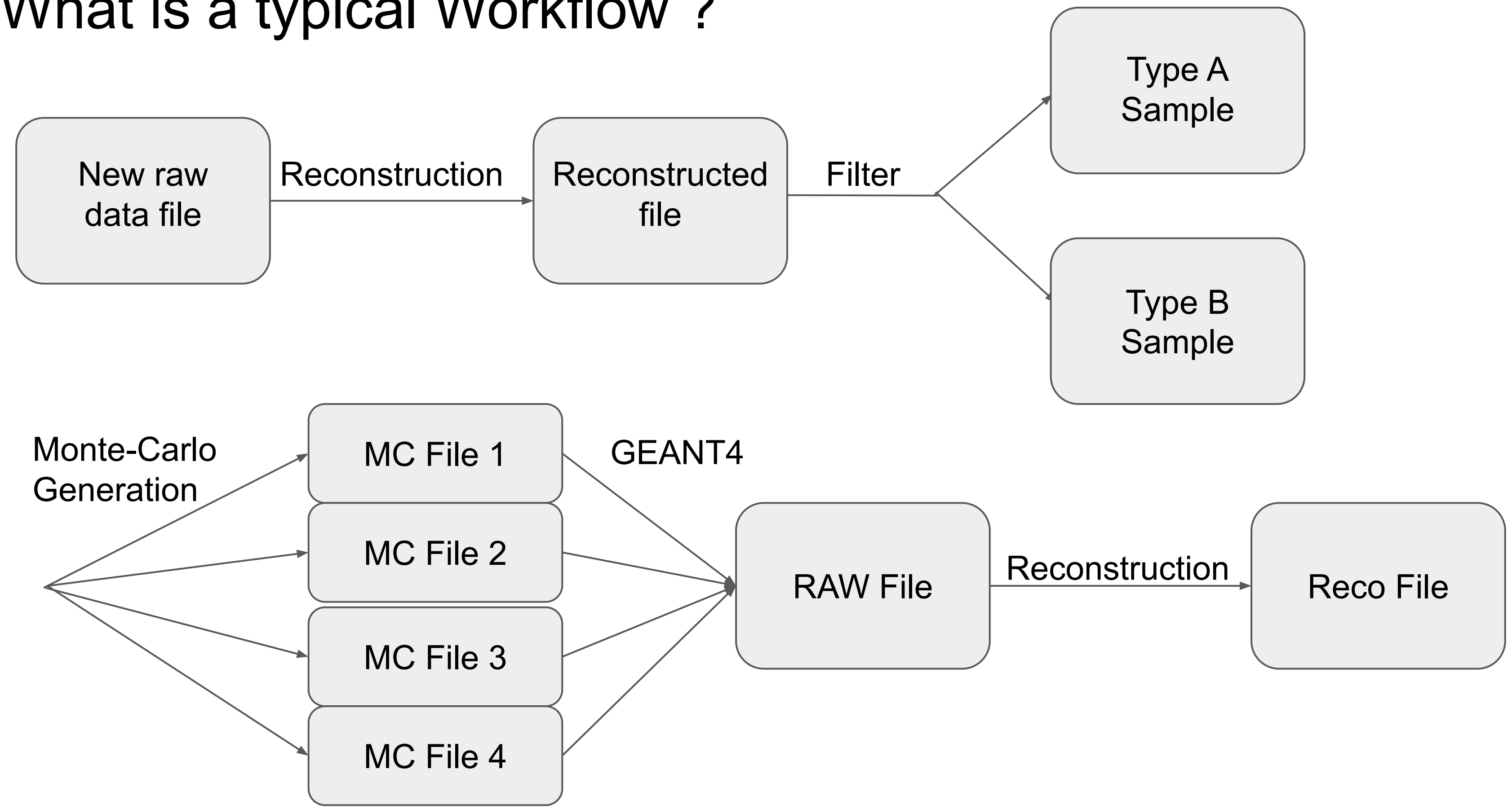
- Develop a manual control for admin with a view to automate this in the future.

# WP1.7 DIRAC Workload Management

- Low(er) priority:
  - no recent problems seen
- Status
  - might not be implementable as originally envisaged
  - code state best described as “collection of hacks”
- We might have to rethink that one:
  - Looking at solutions we can implement as an add-on to the core DIRAC project, rather than integrating it

# WP 1.6 Workflow Management

## What is a typical Workflow ?



Imperial College London

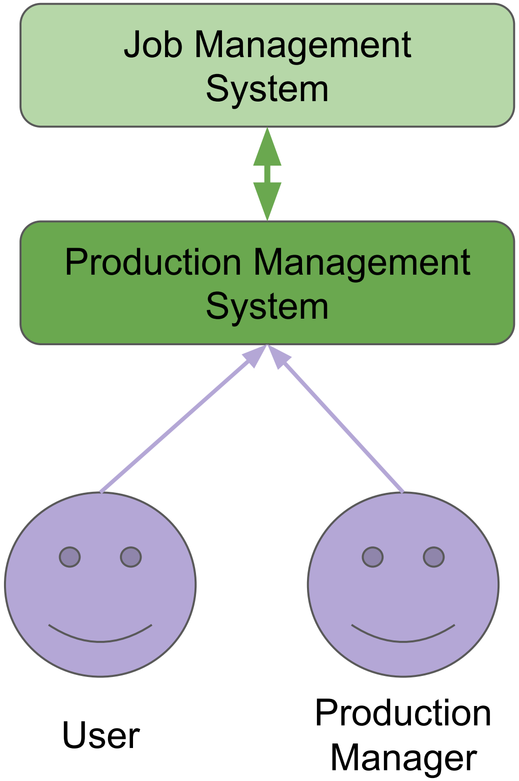
# Status of Workflow Management in DIRAC

- A number of medium sized communities have reached the limit of what can be done in an ad hoc way.
- The core target for this are large production runs, not one-off analyses.
- Basic Workflow management exist in DIRAC core.
- Build on UK work done as part of IRIS digital asset to make code multi-VO compatible.

# WP 1.6 Workflow Management

Planned work:

- Deploy Workflow Software on production server and ensure proper separation of VOs.
  Deployed on pre-prod server.
  Available for testing.
  Now tested as a standard of GridPP pre-prod certification.
- Todo: Increase user friendliness:
  - Error messages.
  - Easier recovery from failures (e.g. rerun part of a workflow).
  - Make existing Web based interface more user/admin friendly.



Imperial College London

# Conclusion

- We are more or less on track with the proposed SWIFT-HEP work
  - The pilot logging is the most substantial bit of work and has already undergone several iterations.