SWIFT HEP

SoftWare InFrastructure and Technology for High Energy Physics

SwiftHep workshop #4

WP5 Analysis Systems

Luke Kreczko for SWIFT-HEP WP5

University of BRISTOL

Outline

News since Workshop #3

Roadmap for the next few months

Since Workshop #3

- New hire - Sam Eriksen (LZ/SWIFT-HEP)
  - After initial difficulties, finally managed to find a suitable candidate
  - Going to start in the coming months
- GridPP DIRAC
  - Upgraded to 7.3 in June
  - JobManager, JobMonitor, JobStateUpdate available via HTTP
- Brunel site volunteered for Analysis Workflow
  - Will be first site we test via DIRAC

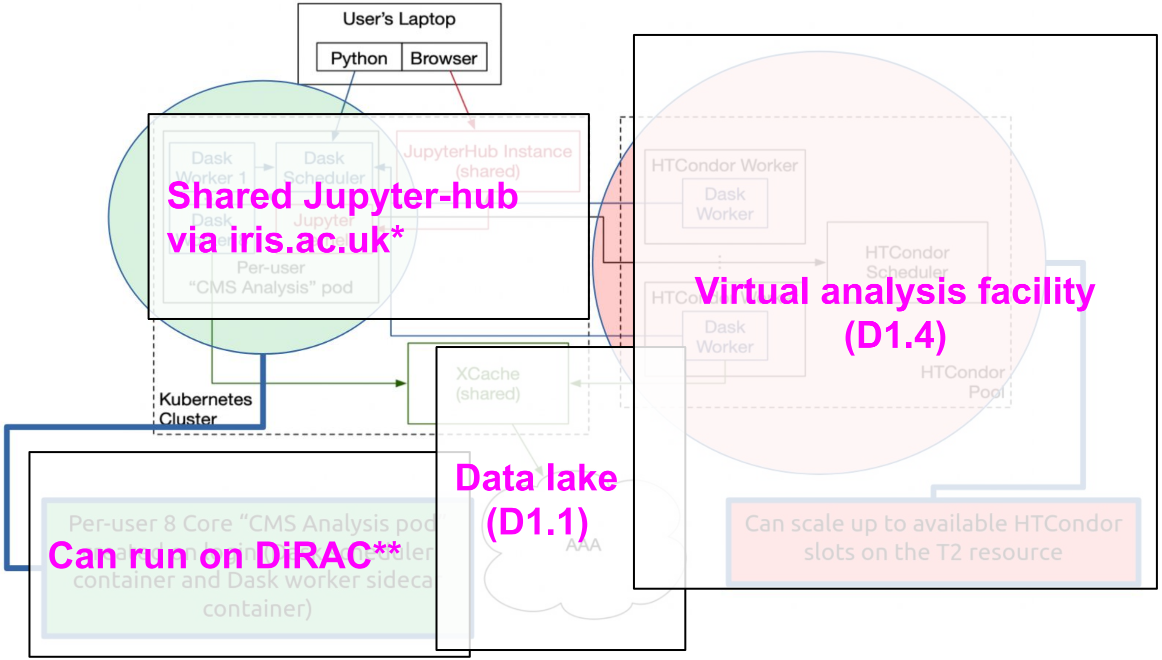
WP5 in a nutshell: run analysis workloads optimally** on distributed (GridPP) resources

** balanced between user-experience and computing efficiency

SWIFT-HEP

“Adaptation” of Coffea-casa

As simple as adding DIRAC jobqueue to dask-jobqueue?



User’s Laptop
Python
Browser
Dask Worker 1
Dask Scheduler
JupyterHub Instance (shared)
Dask
Jupyter
Per-user “CMS Analysis” pod
Kubernetes Cluster
XCache (shared)
HTCondor Worker
Dask Worker
HTCondor Scheduler
HTCondor Worker
Dask Worker
HTCondor Pool
Per-user 8 Core “CMS Analysis pod”
container and Dask worker sidecar container)
AAA
Can scale up to available HTCondor slots on the T2 resource

Shared Jupyter-hub via iris.ac.uk*

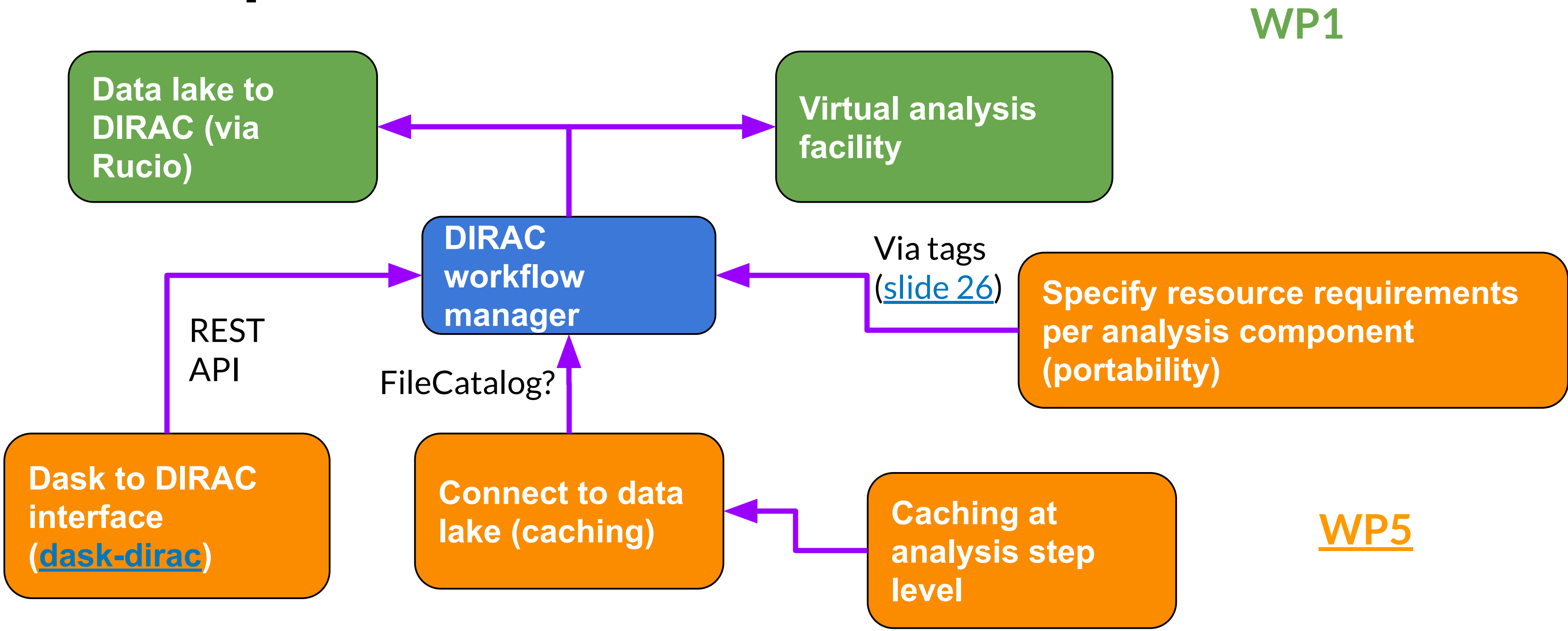
Virtual analysis facility (D1.4)

Data lake (D1.1)

Can run on DiRAC**

*no relation to IRIS-HEP; **no relation to DIRAC

Roadmap overview



WP1

Data lake to DIRAC (via Rucio)

Virtual analysis facility

DIRAC workflow manager

Via tags (slide 26)

Specify resource requirements per analysis component (portability)

REST API

FileCatalog?

Dask to DIRAC interface (dask-dirac)

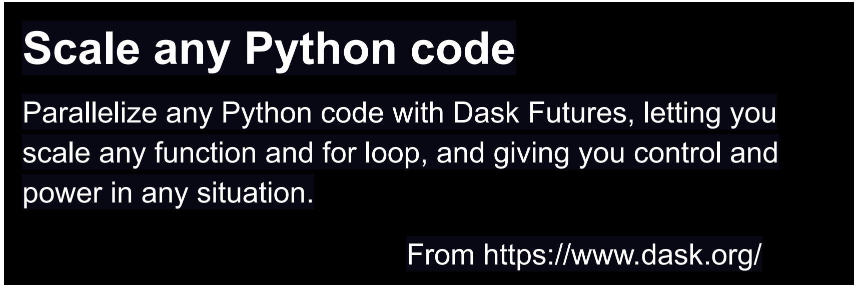
Connect to data lake (caching)

Caching at analysis step level

WP5

Closes example of what we want to achieve: Dask-based Distributed Analysis Facility (kubernetes slides)

Intermission: Dask



Scale any Python code

Parallelize any Python code with Dask Futures, letting you scale any function and for loop, and giving you control and power in any situation.

From https://www.dask.org/

Dask can submit to most batch systems all the same - fantastic from users' perspective

Can we use Dask in HEP?

Coffea (Analysis Grand Challenges) ✓

RDataframe ✓

Awkward-array (native support) WORK IN PROGRESS

FAST-HEP (custom graphs) WORK IN PROGRESS

If infrastructure can be used via Dask → wide use is possible

Dask-dirac

Step 1 of WP5:
Build connector between
Dask and GridPP DIRAC
(Github repo)

Pre-requisites:

- No extra dependencies (DIRAC is hard* to add to standard python installs)
- DIRAC should “just be an extra batch system Dask supports”

We envision this through the HTTP

- Support starts in version 7.3
- HTTP dask-client can be a free useful outcome

* try `pip install DIRAC`

Analysis Grand Challenges

Step 2: Run Analysis Grand Challenges at Brunel via DIRAC

(Github repo)

IRIS-HEP currently provides

- ATLAS H→ZZ
- CMS ttbar

What SWIFT-HEP could provide

- CMS Higgs analysis (Imperial)
- LZ Analysis (Bristol)

To start with analyses need to be able to use Dask.

Later also custom graphs (caching, portability).

Next steps

1. Caching of analysis output
2. Caching of individual analysis steps
3. Portability: on-availability matching of specialised resources
4. Test different use cases and improve setup

Basic test of D1.1

Complex test of D1.1

Test of D1.4 and D1.7

Tying it all together

Discussion topics

- Hiring difficulties have caused delays → how can we minimize the impact on other deliverables
- AGC are based on open data → do we want an opendata.gridpp.ac.uk for non-LHC VOs?
    - I will try to convince LZ to release some of their MDC3 (simulation) data
- Anything to add on the previous talks (ROOT, Analysis Facilities)?
- Anything you would like to raise & discuss?

Backup

Analysis key points

Physics

Last mile of long chain of data recording and processing.

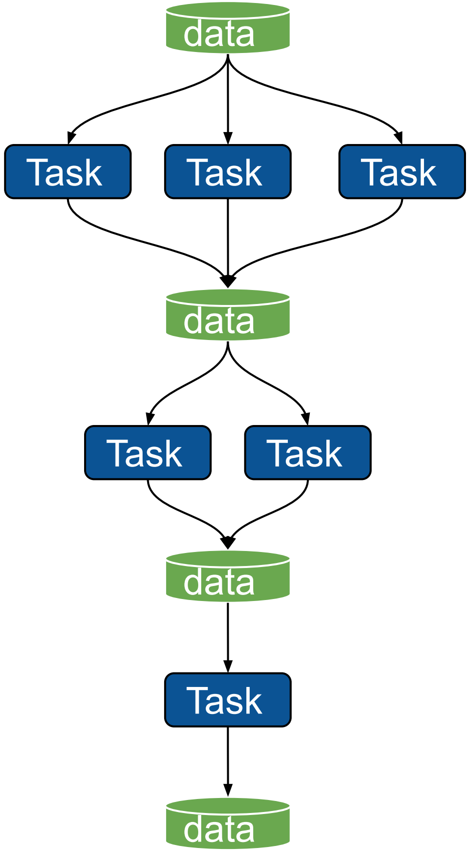
Goals: gain insight and create new knowledge

Computing

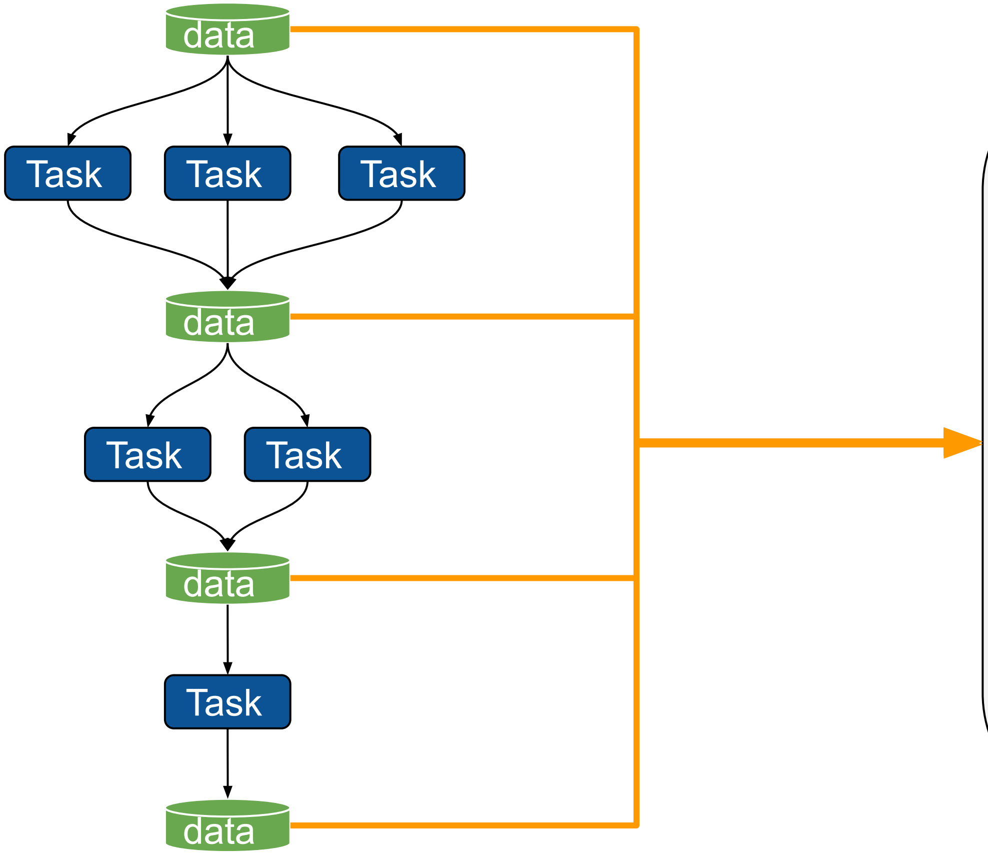
Analysis workflow (data + software) depends on experiment, analysis group, subset of data (signal + relevant backgrounds), analysis iteration.

Flexibility is paramount.

Anatomy of an analysis workflow



Anatomy of an analysis workflow

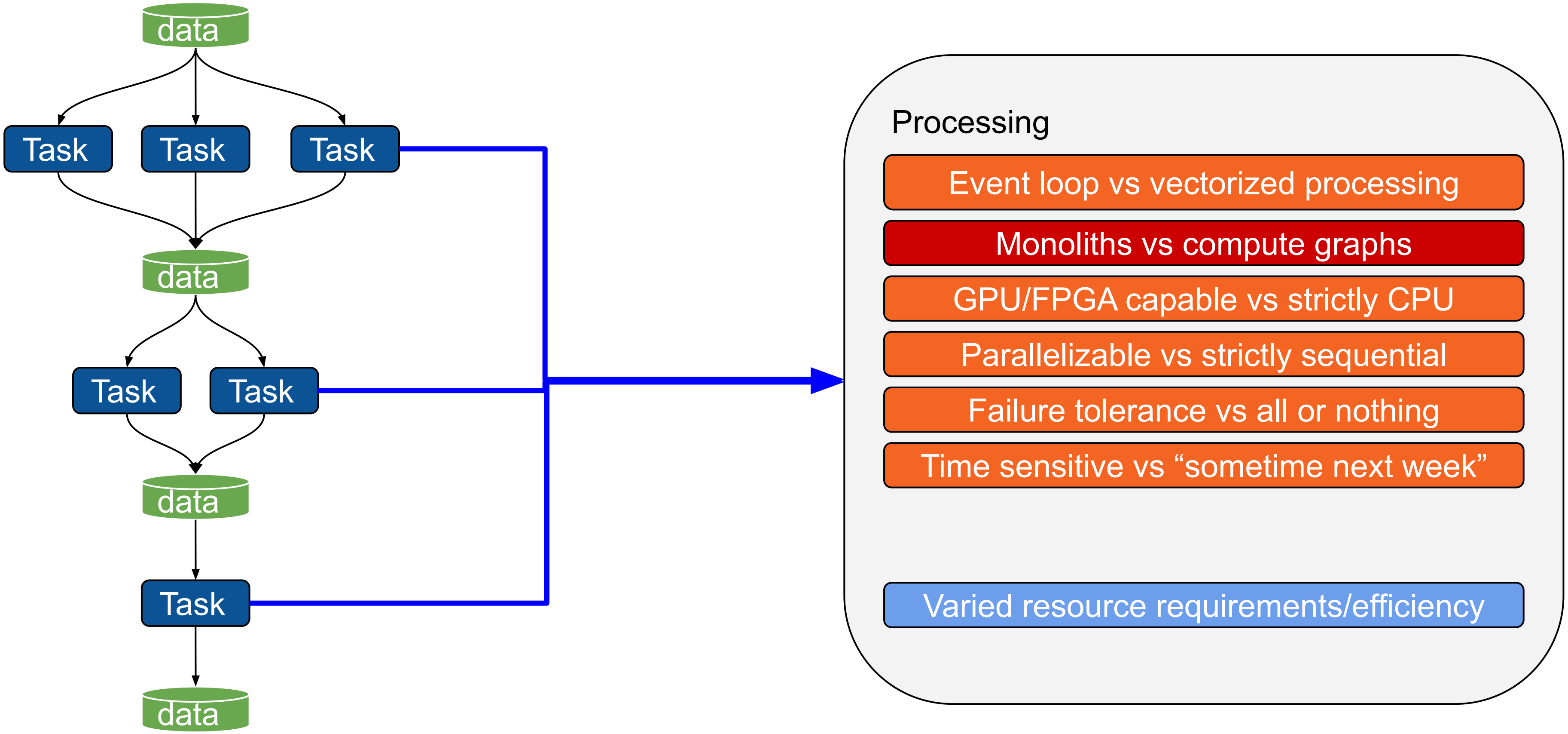


Data Formats

ROOT files: standard for input/output
Internally: Experiment/analysis specific

HDF5	npz
pandas	datacard
parquet	other?

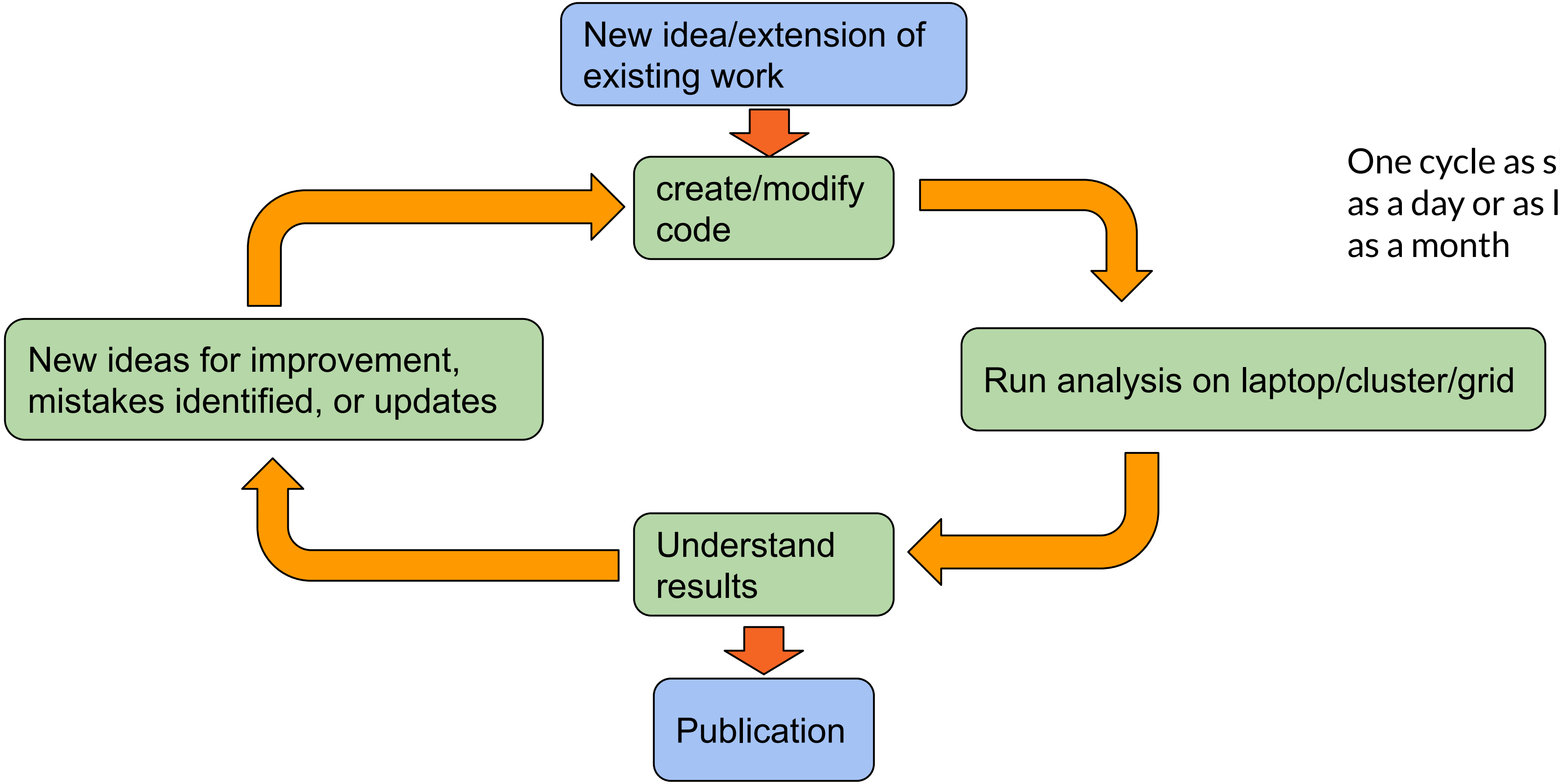
Anatomy of an analysis workflow



Processing

- Event loop vs vectorized processing
- Monoliths vs compute graphs
- GPU/FPGA capable vs strictly CPU
- Parallelizable vs strictly sequential
- Failure tolerance vs all or nothing
- Time sensitive vs “sometime next week”
- Varied resource requirements/efficiency

The cycle of analysis (an oversimplified view)



New idea/extension of existing work

create/modify code

Run analysis on laptop/cluster/grid

Understand results

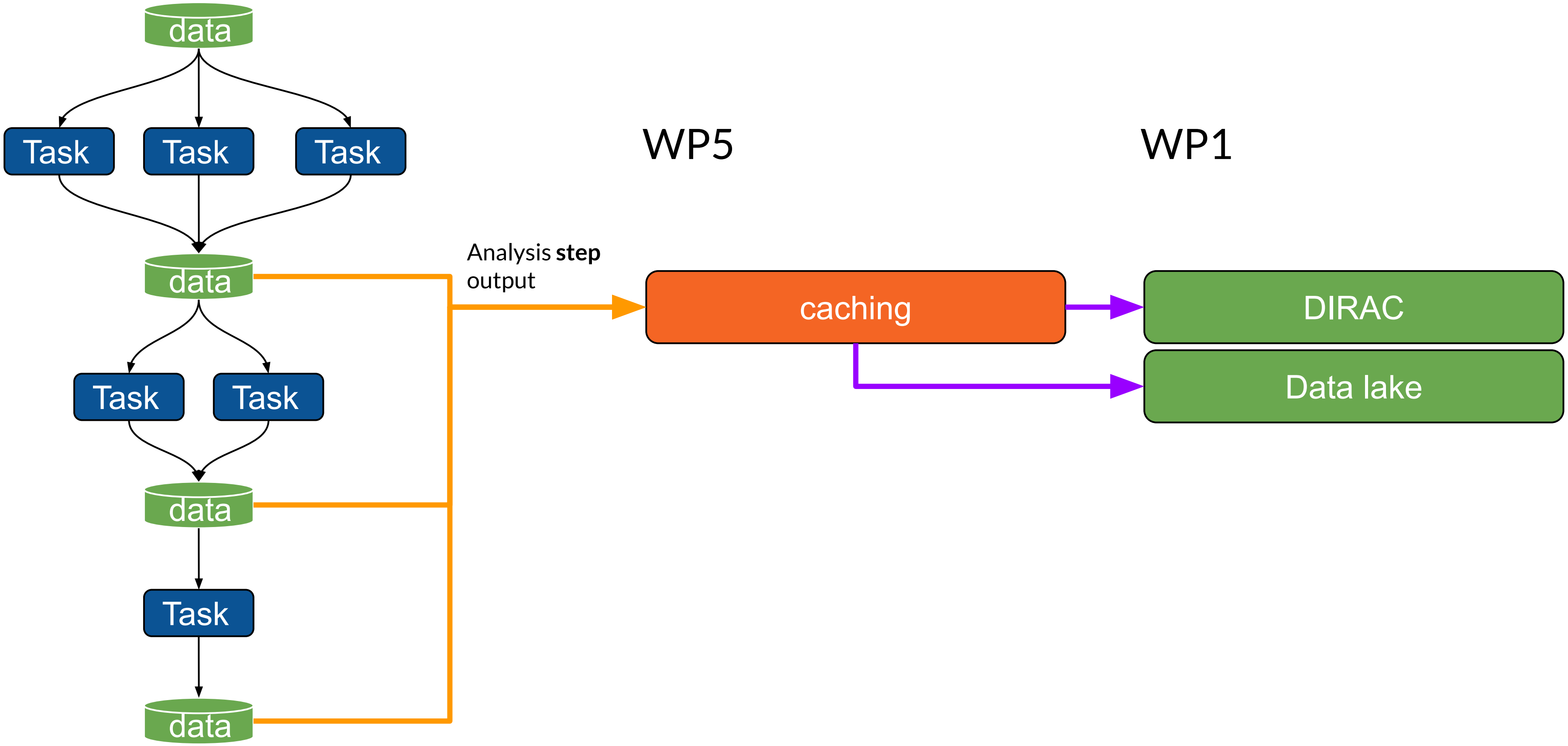
New ideas for improvement, mistakes identified, or updates

Publication

One cycle as short as a day or as long as a month

Work Package 5: Analysis

Analysis workflow



data

Task Task Task

data

Task Task

data

Task

data

Analysis **step** output

WP5

caching

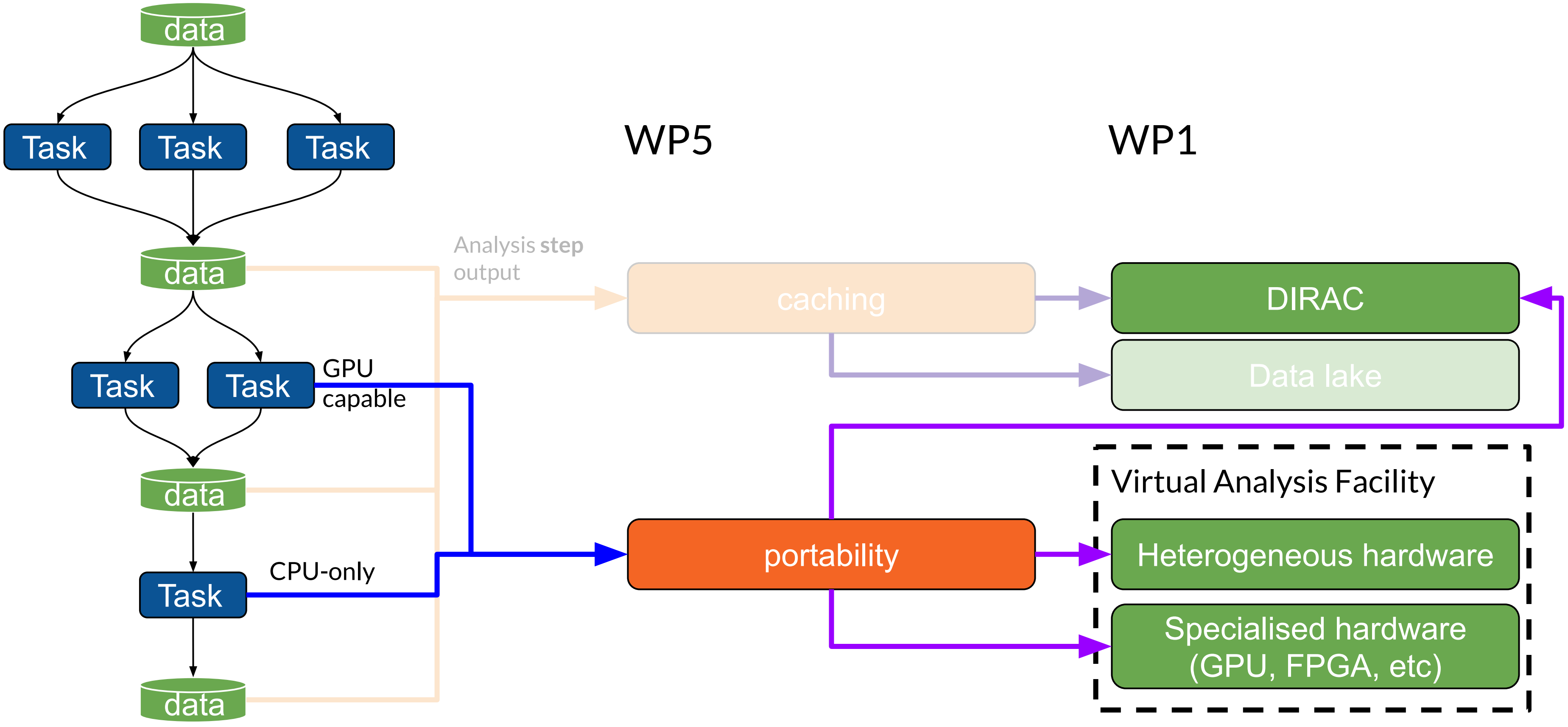
WP1

DIRAC

Data lake

Work Package 5: Analysis

Analysis workflow



data

Task Task Task

data

Task Task

GPU capable

data

Task

CPU-only

data

Analysis **step** output

WP5

WP1

caching

DIRAC

Data lake

portability

Virtual Analysis Facility

Heterogeneous hardware

Specialised hardware (GPU, FPGA, etc)