CERN School of Computing

2023 LIGHTNING TALKS PART 2

Matic Orehar	Positron Emission Tomography - the basics
Enrique Garcia Garcia	The Virtual Research Environment
Marco Faltelli	"Data's Need for Speed: InfiniBand and RDMA Driving HPC Evolution"
Andrea Valenzuela Ramirez	CernVM-FS for Efficient Software Distribution at CMS
Lorenzo Santi	Investigating the impact of 4D Tracking in ATLAS Beyond Run 4
Elizabeth Mamtsits	My journey on refactoring and translating legacy code
Natalia Szczepanek	Benchmarking ATLAS Distributed Computing Resources
Alberto Pimpo	Unveiling containers
Jamie Gooding	40 MHz or bust: real-time triggering on full-detector readout at LHCb
Javier Romero Castro	Empowering Research through Invenio RDM: Managing, Sharing, and Preserving Research Data
Matteo Marchegiani	PocketCoffea: a configuration layer for CMS analyses with Coffea
Patin Inkaew	Data Scouting Jet for Run 3 at CMS

Andrzej Nowicki 🇵🇱

12 years of Oracle DB experience
Database Engineer @ CERN since 2020

andrzejnowicki

andrzej.nowicki@cern.ch

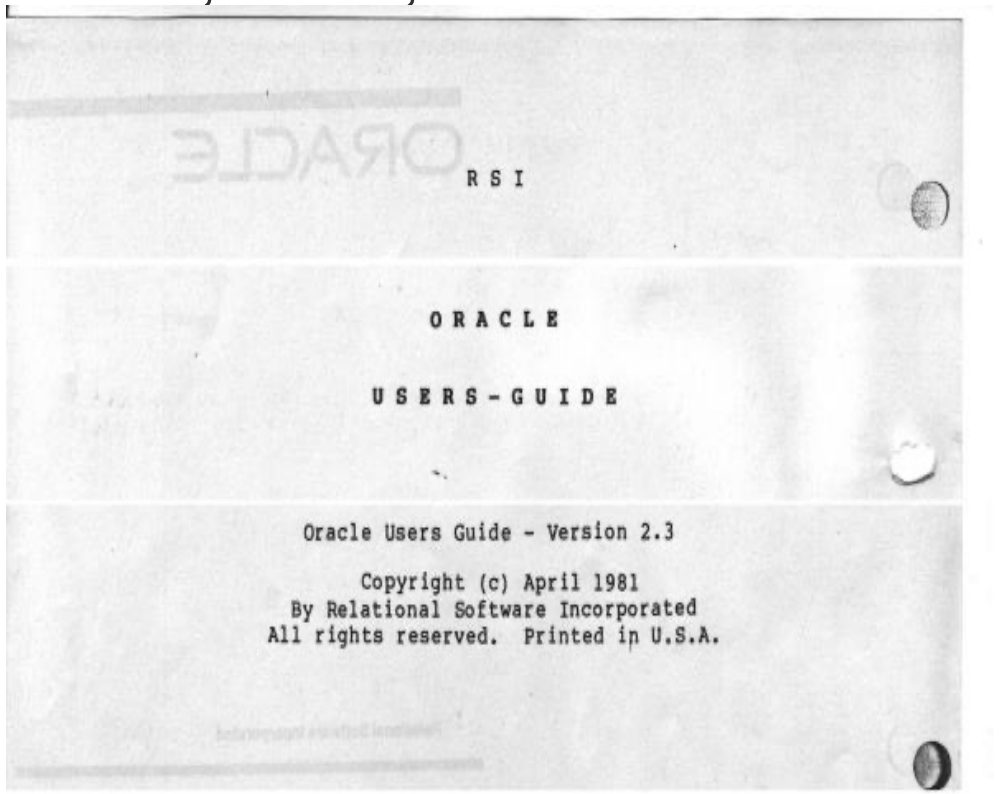
Databases at CERN

Oracle since 1982

- 105 Oracle databases, more than 11.800 Oracle accounts
- RAC, Active DataGuard, Golden Gate, OEM, RMAN, Cloud, …
- Complex environment

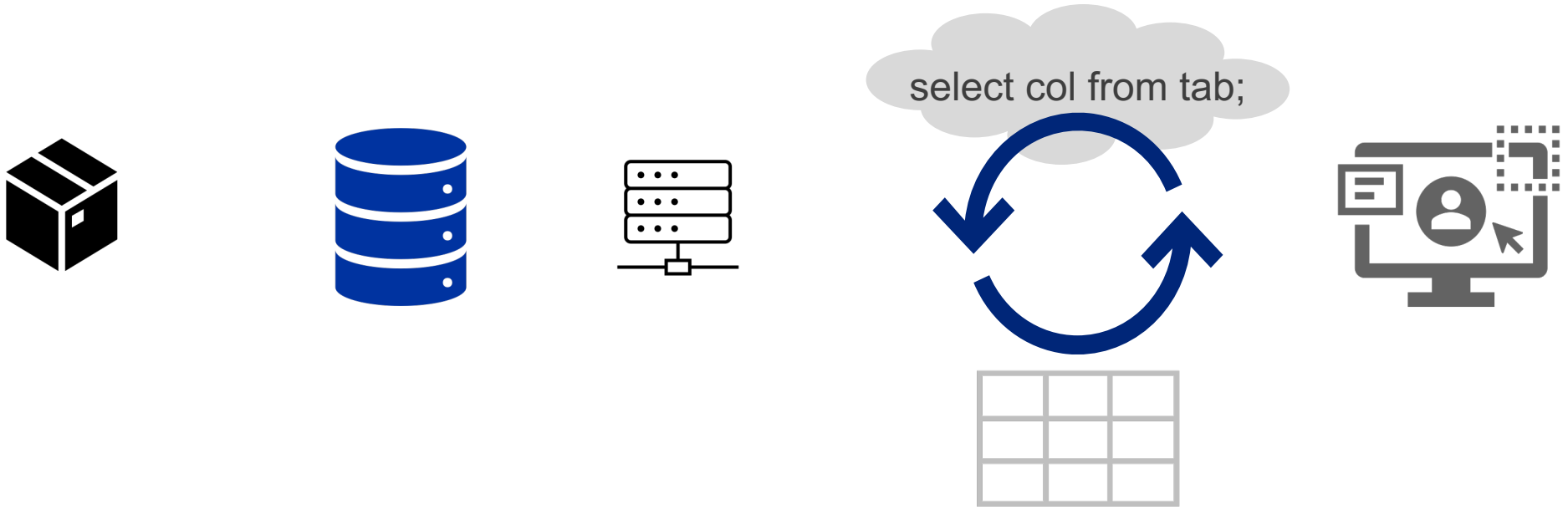
Database on Demand (DBoD) since 2011

- MySQL, PostgreSQL, InfluxDB



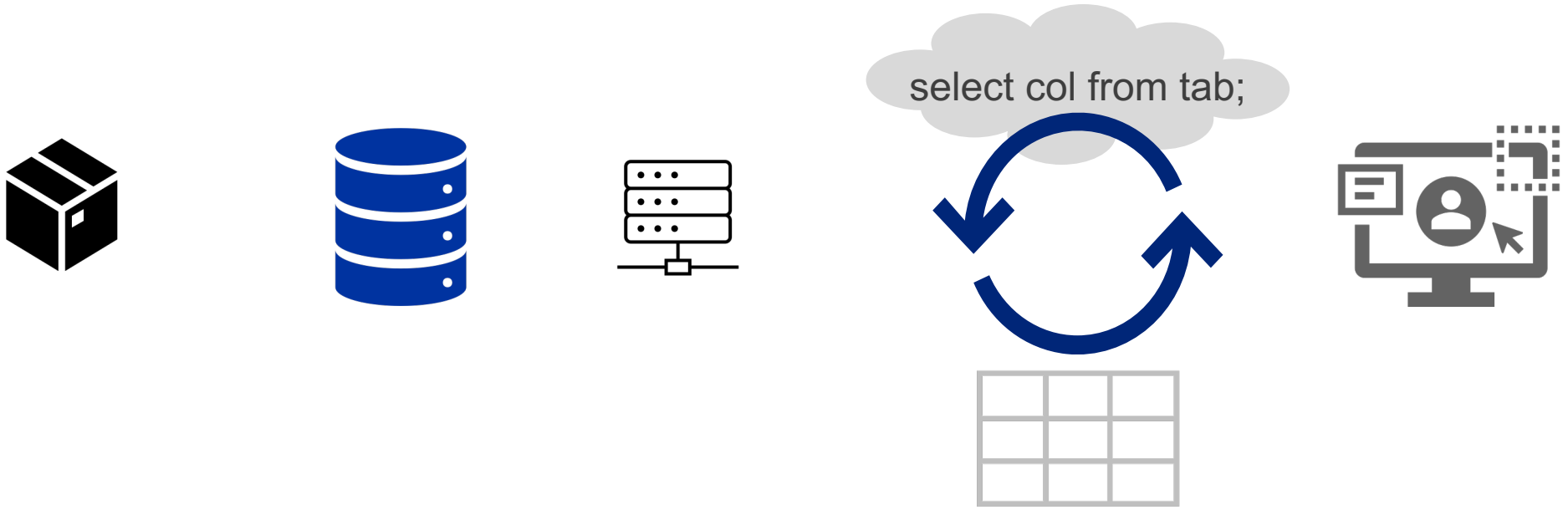
How to make your database resilient?

Basic database system

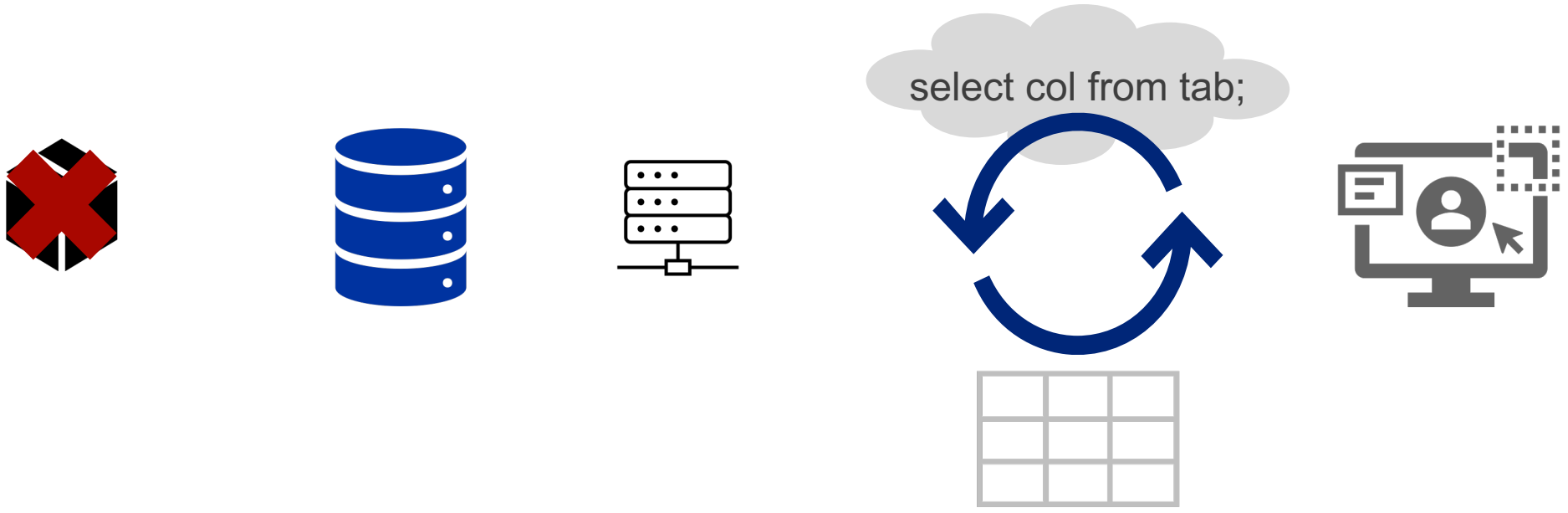


What if something breaks?

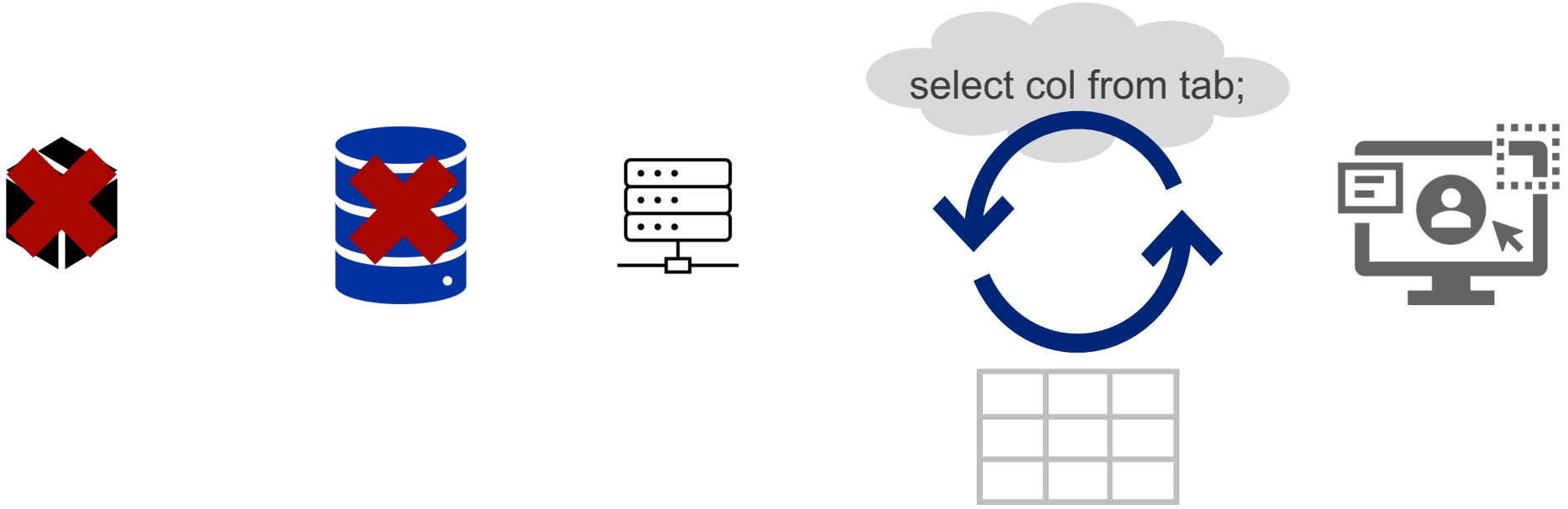
Basic database system



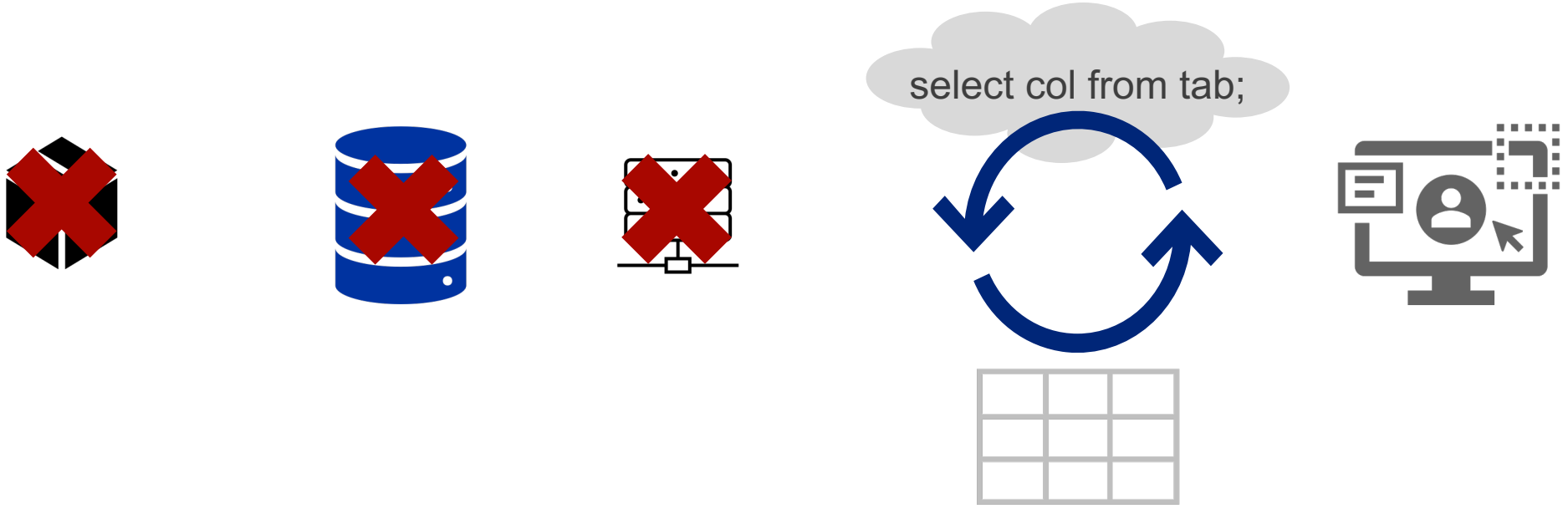
Basic database system



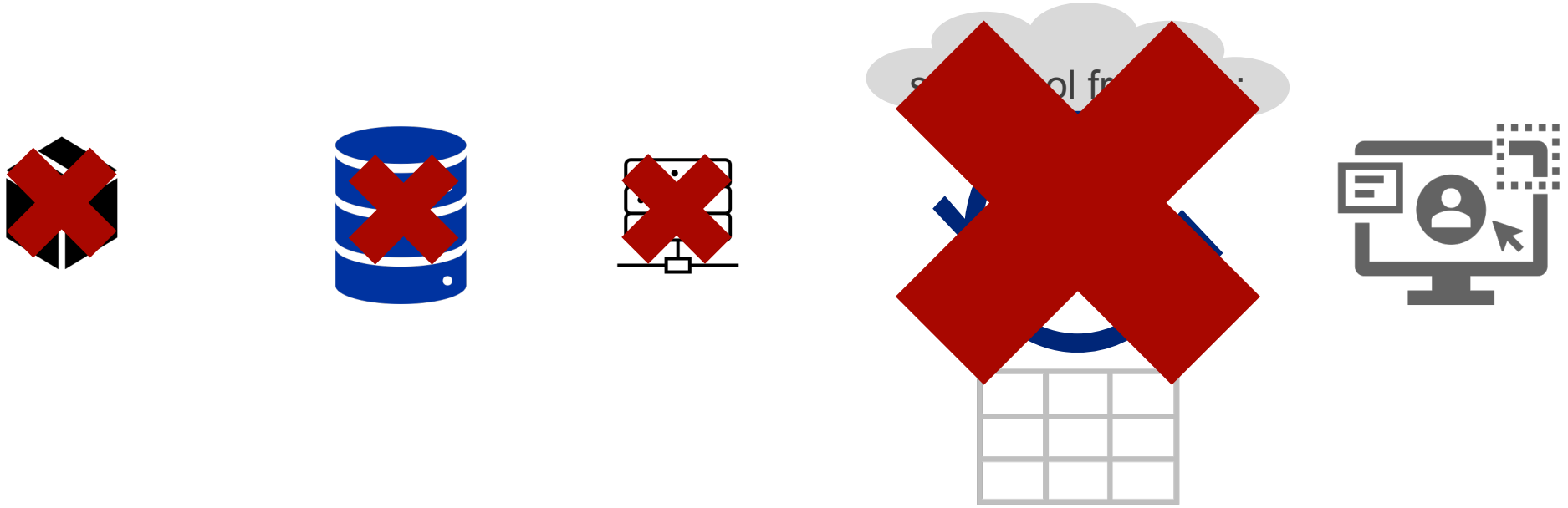
Basic database system



Basic database system

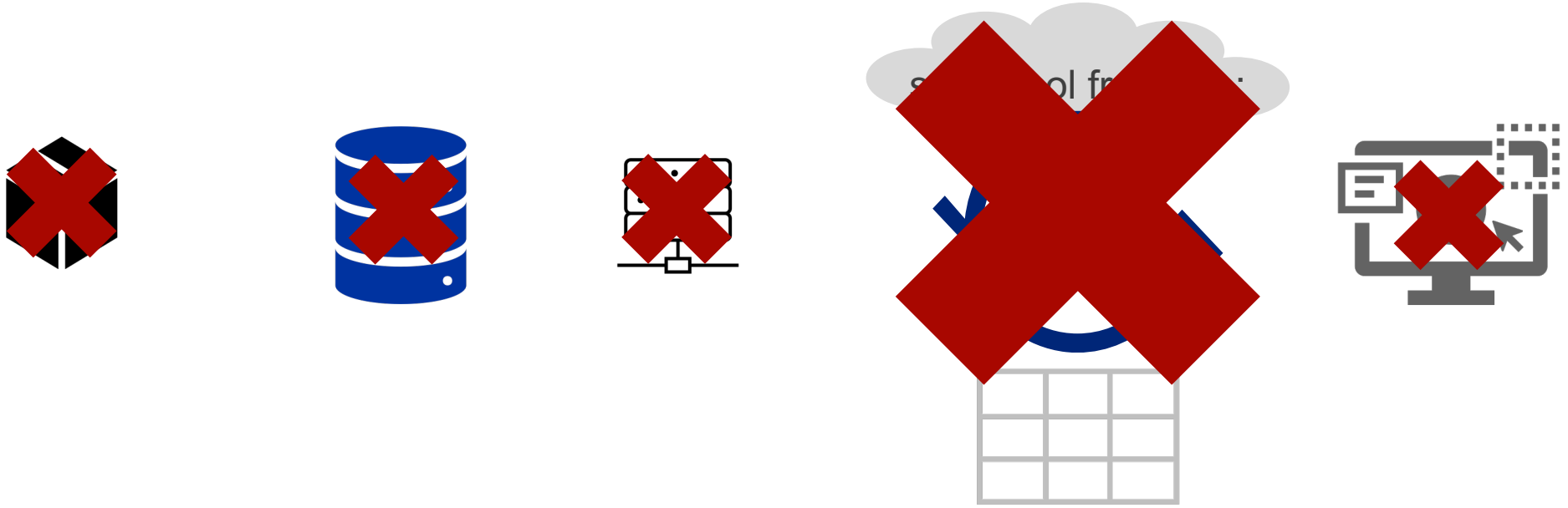


Basic database system



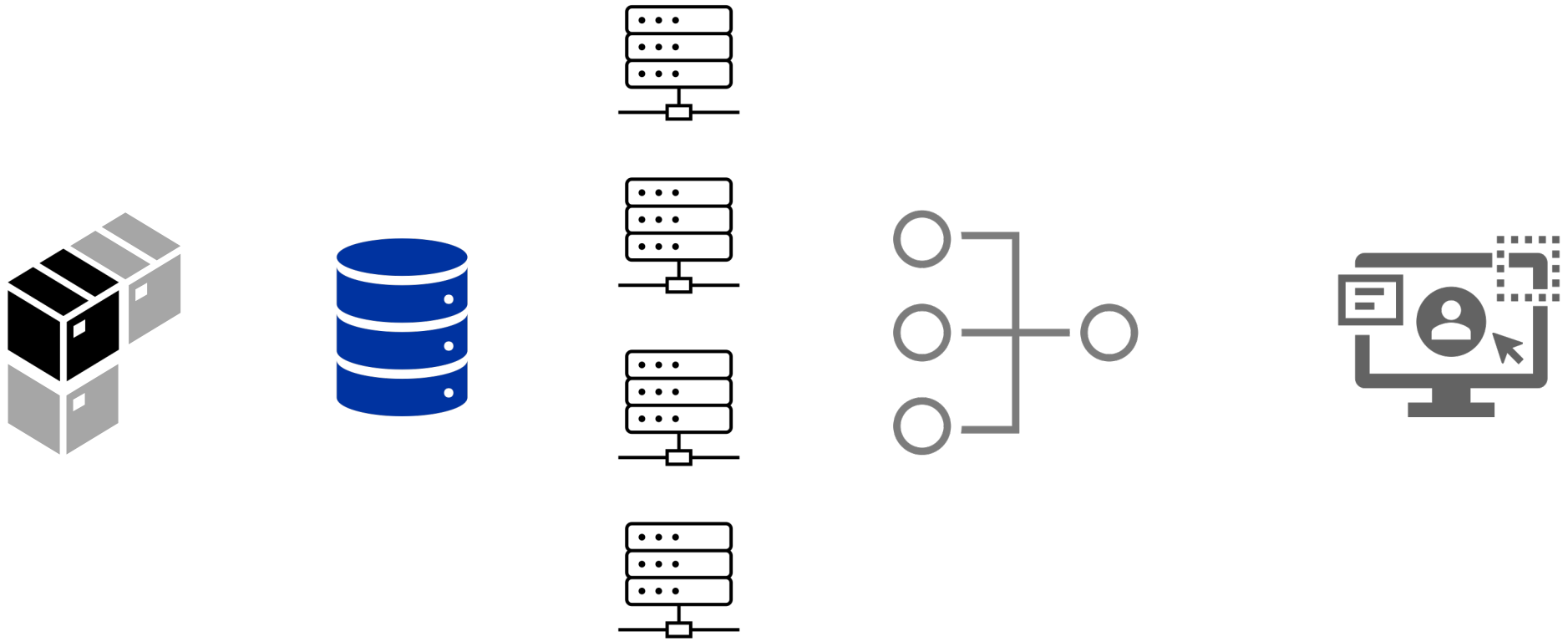
STORAGE DATAFILES SERVER APPLICATION

Basic database system



STORAGE DATAFILES SERVER APPLICATION

Highly available database system



STORAGE

DATAFILES

SERVERS

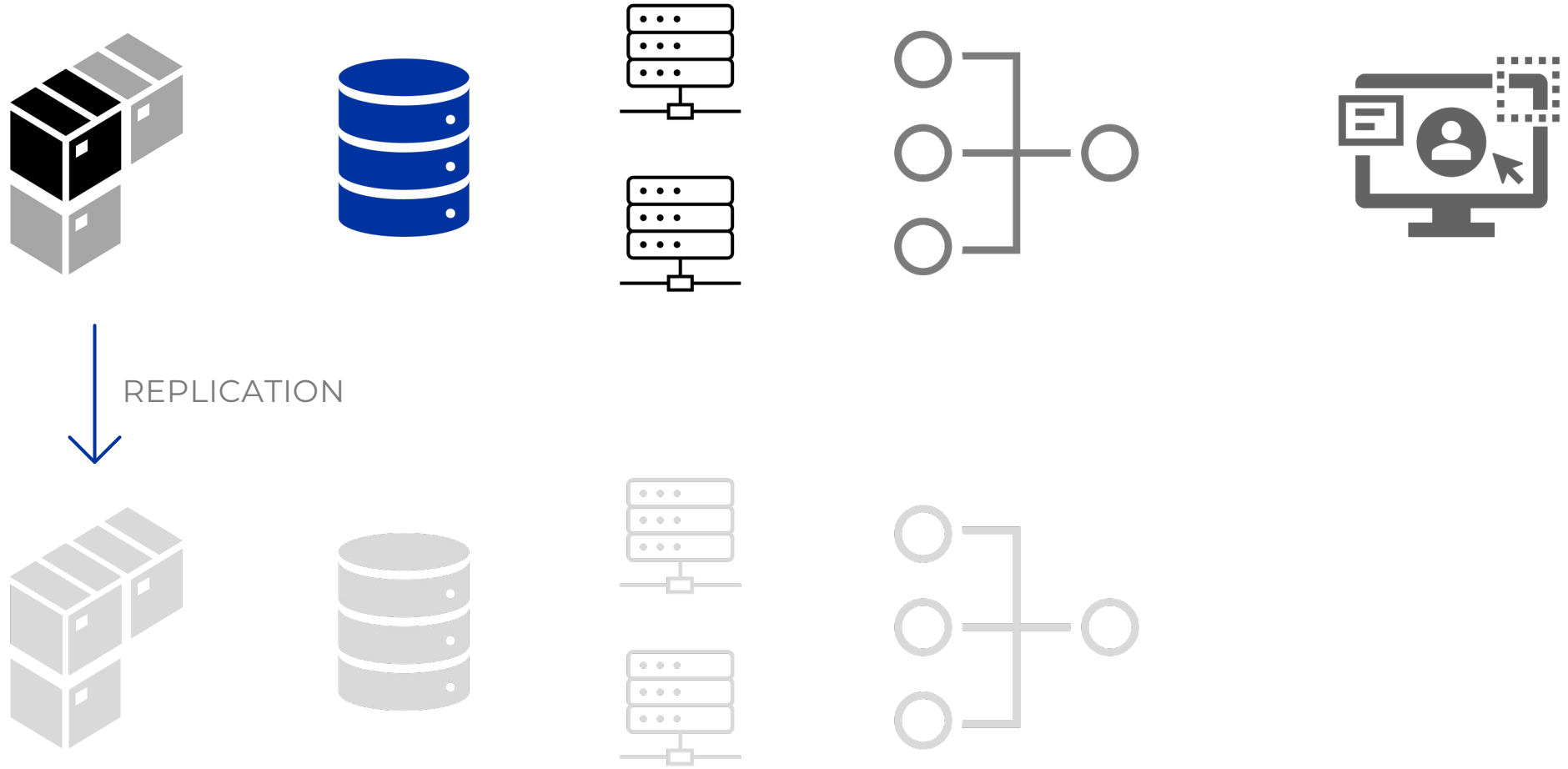
LISTENER
(LOAD BALANCER)

APPLICATION

Is that enough?

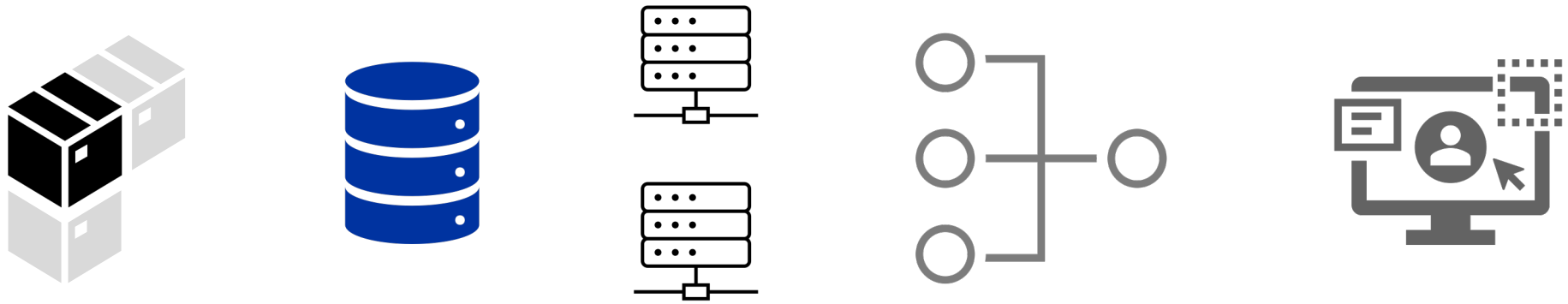
What if your datacenter is on fire?

Highly available database system with DR



Highly available database system with DR

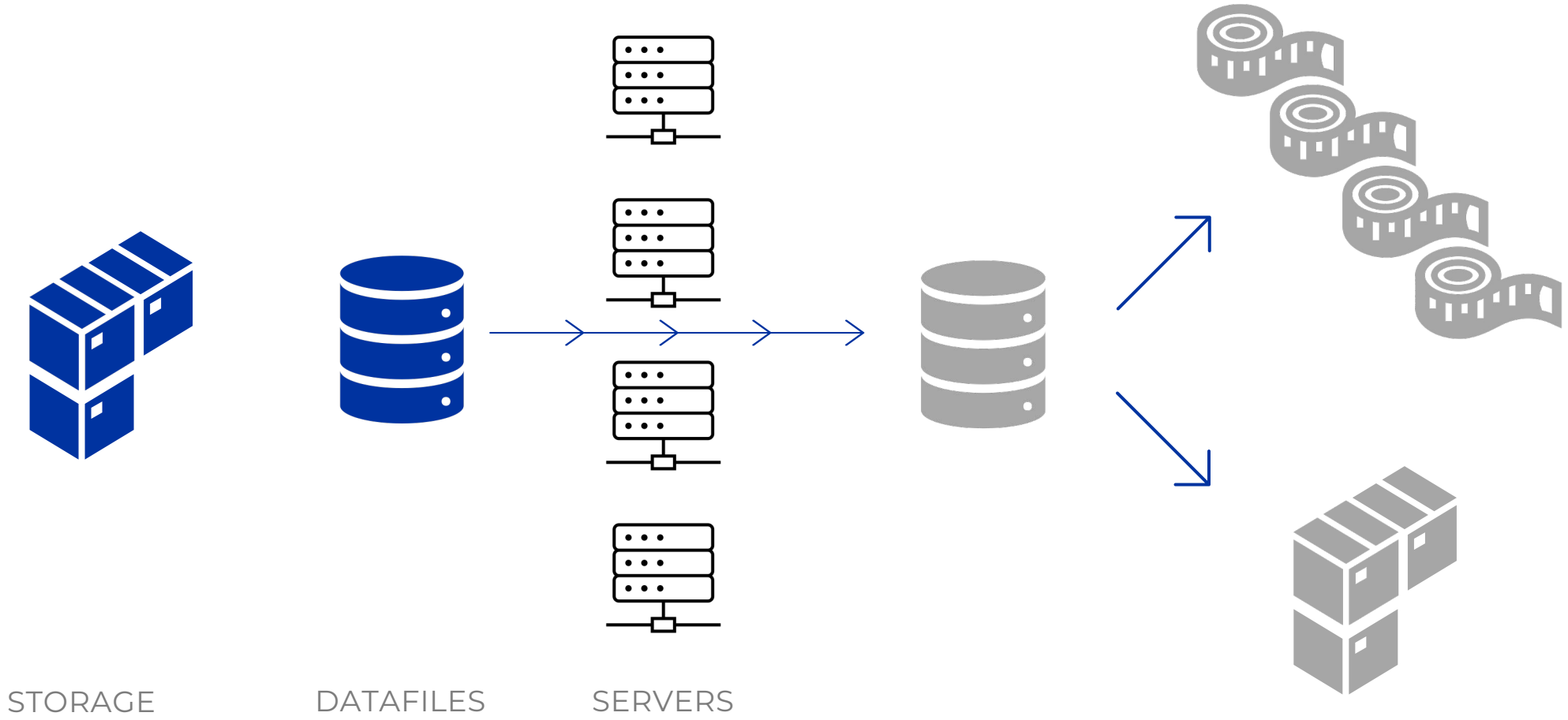
Highly available database system with DR



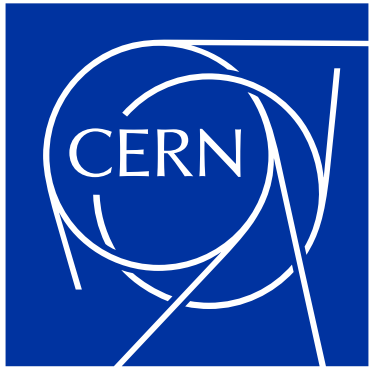
Is that enough?

What if somebody deleted some data?

Backups



Thank you !



andrzej.nowicki@cern.ch

med fiz

Univerza v Ljubljani
Fakulteta za matematiko in fiziko

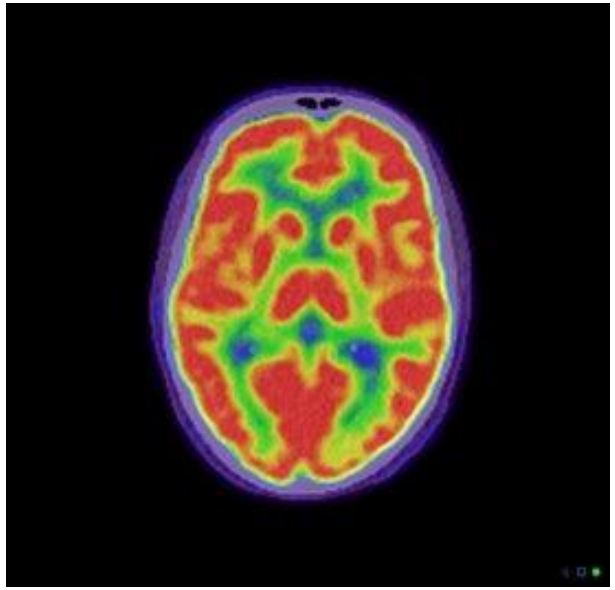
Institut "Jožef Stefan" Ljubljana, Slovenija

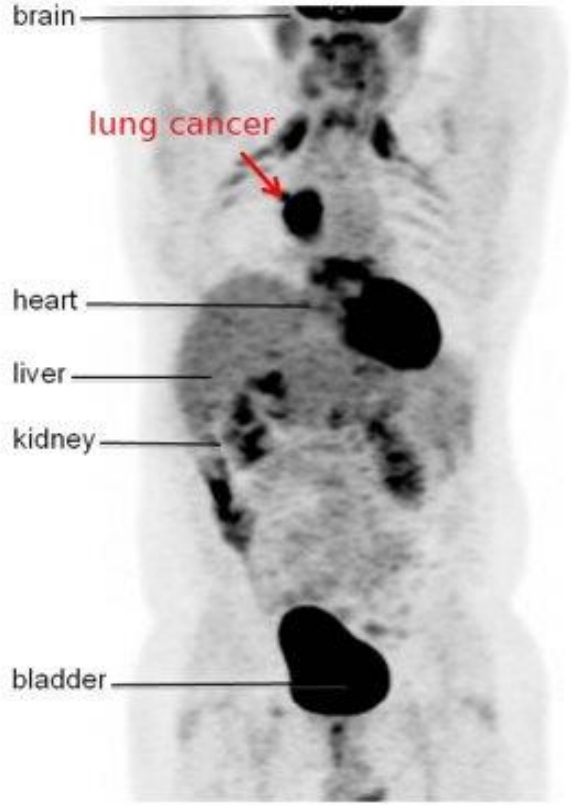
Positron emission tomography - the basics

Matic Orehar

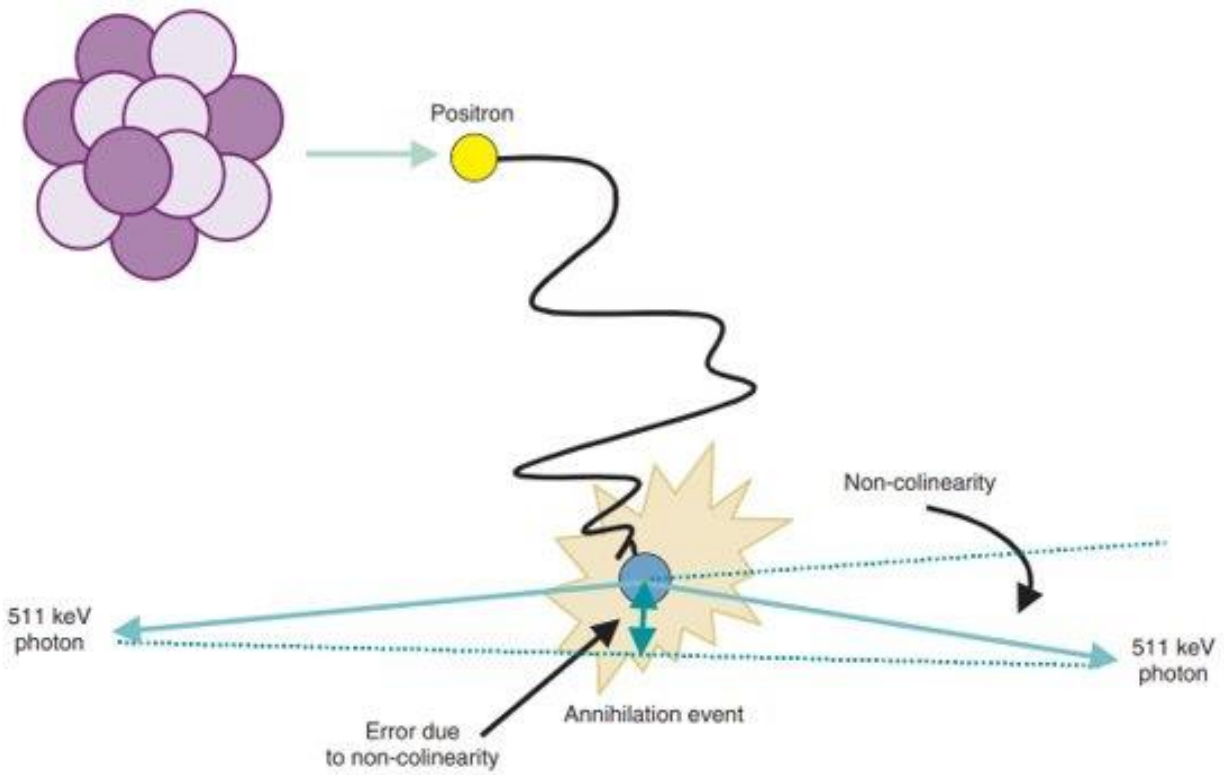
What is PET?

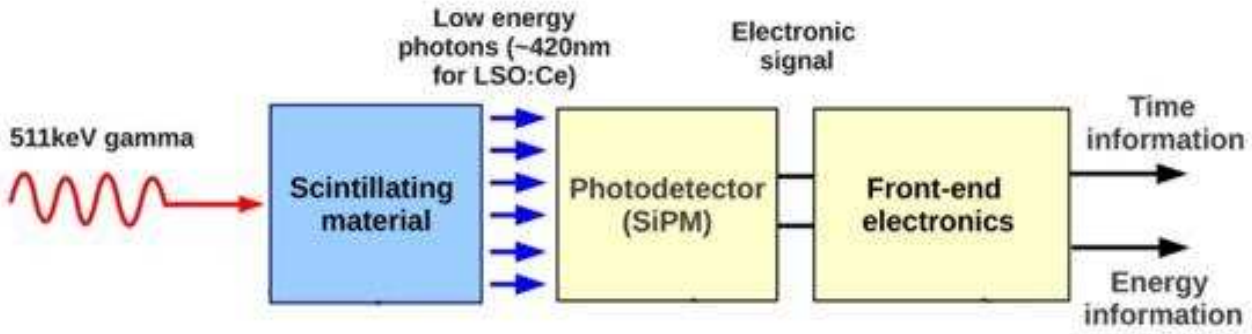
- Medical imaging modality, that images **physiological processes** in the body

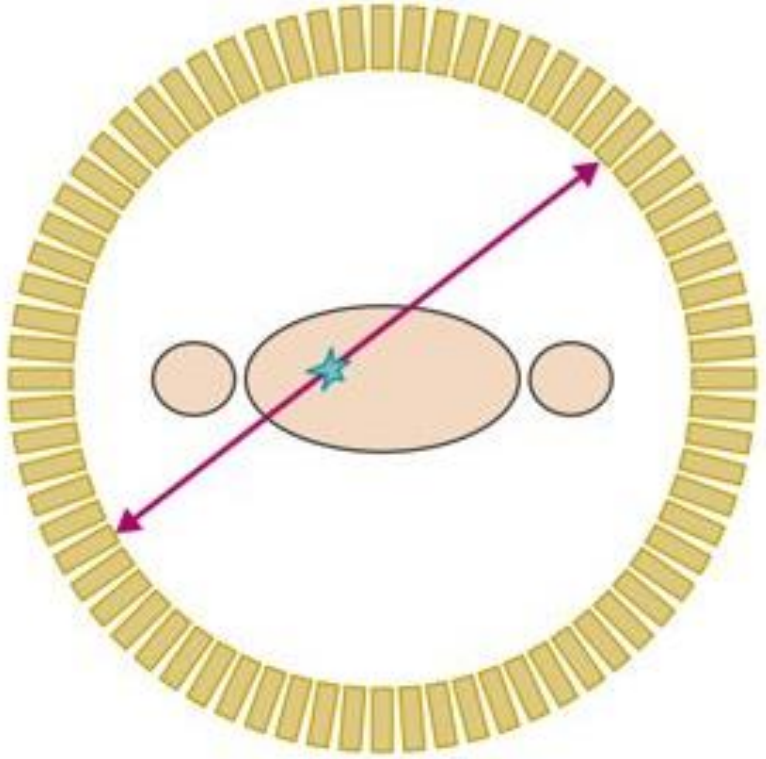




How does PET work?







Q & A

Virtual Research Environment

Towards a comprehensive analysis platform

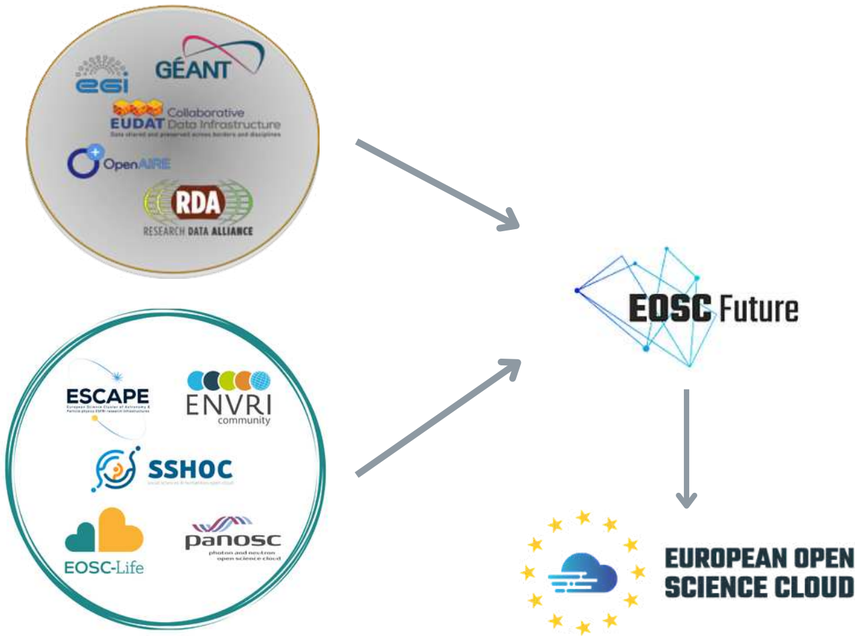
Enrique GARCIA *- based on E. Gazzarrini slides*
CERN Fellow - IT Department, Governance Engagement Section
CERN School of Computing 2023 - Tartu

European Union's Horizon 2020 programme Grant Agreement 824064 and 101017536

The Virtual Research Environment

The VRE is an **open source** analysis platform where researchers have access to all the digital content needed to **develop, share and reproduce an end-to-end scientific result** in compliance with **FAIR** (findable, accessible, interoperable, reproducible) principles.

- Researchers
- Community Software & Analysis development
- Data & Workflow Management
- IT administrators

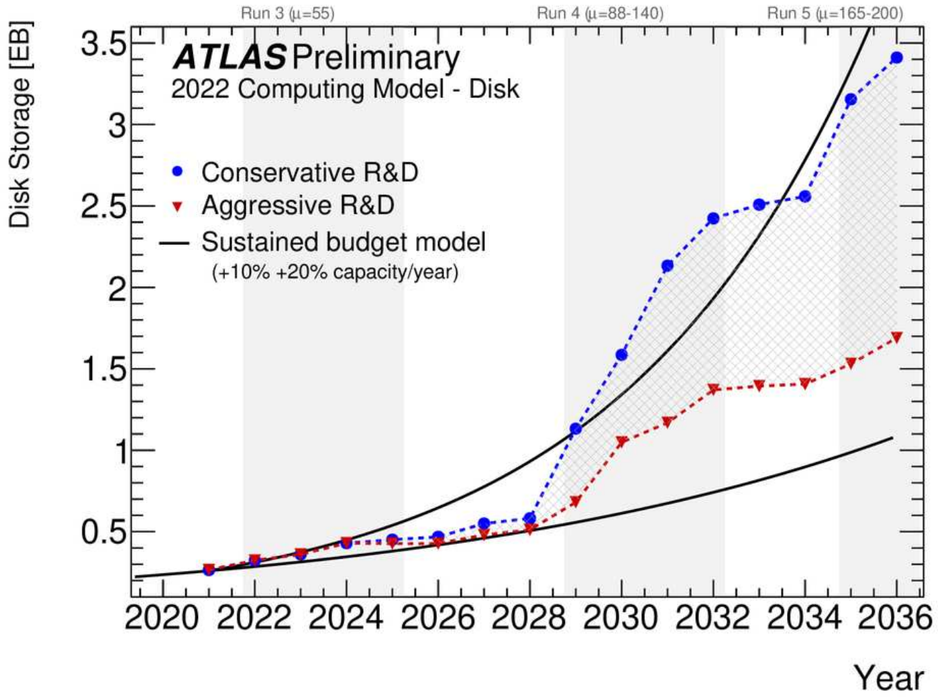


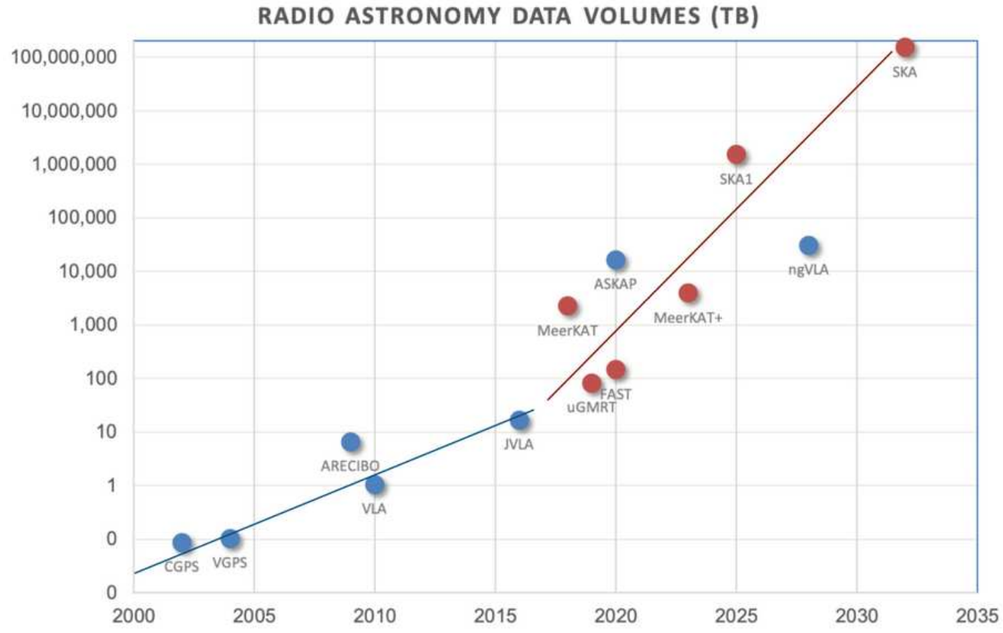
CERN EOSC Future ESCAPE European Science Cluster of Astronomy & Particle physics ESFRI research infrastructures European Commission

Data volumes growing not only at LHC

The LHC at CERN was the first large scientific experiment to generate and manage multi PBs of data per year.

Technologies to manage and process data initially developed at CERN are being adopted by other collaborations, as new generation of detectors, antennas and telescopes are producing and processing large data volumes as well.



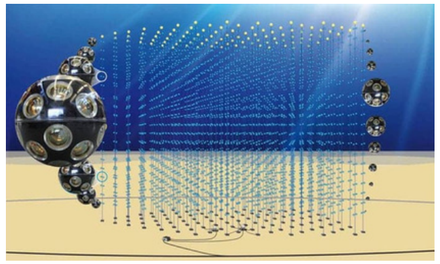
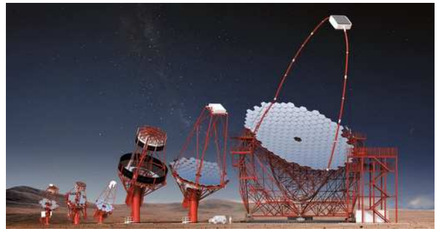
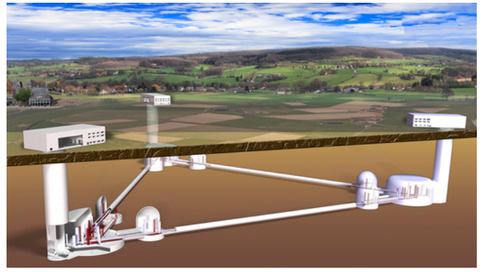


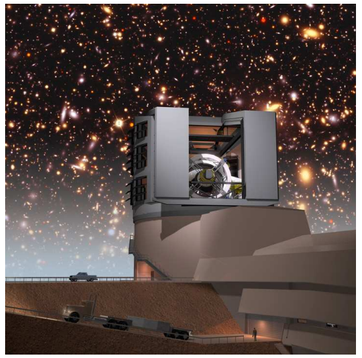
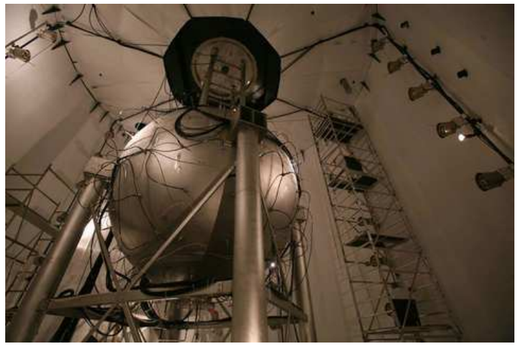
Taylor R. et al. Big Data Research Infrastructure Collaboration Toward the SKA (BRICSKA). doi: 10.1590/0001-3765202120201027. PMID: 34076205.

The challenge

A common infrastructure across Research Infrastructures would foster:

- economy of scale
- collaboration across domains
- scientific reuse
- sustainability

EU collaborations

EU-funded projects promote cross-fertilisation across Research Infrastructures and scientific domains to find common, consistent and useful solutions to challenges of

- **Federated Data Management** and Transfer Services
- Distributed Data **Processing**
- **Software Sustainability**
- Analysis **Preservation and Reusability**

... all in one common Analysis Platform!

CERN EOSC Future ESCAPE European Science Cluster of Astronomy & Particle physics ESFRI research infrastructures European Commission

Analysis workflow

reduce data formats for analysis

with preserved, versioned software on computing resources

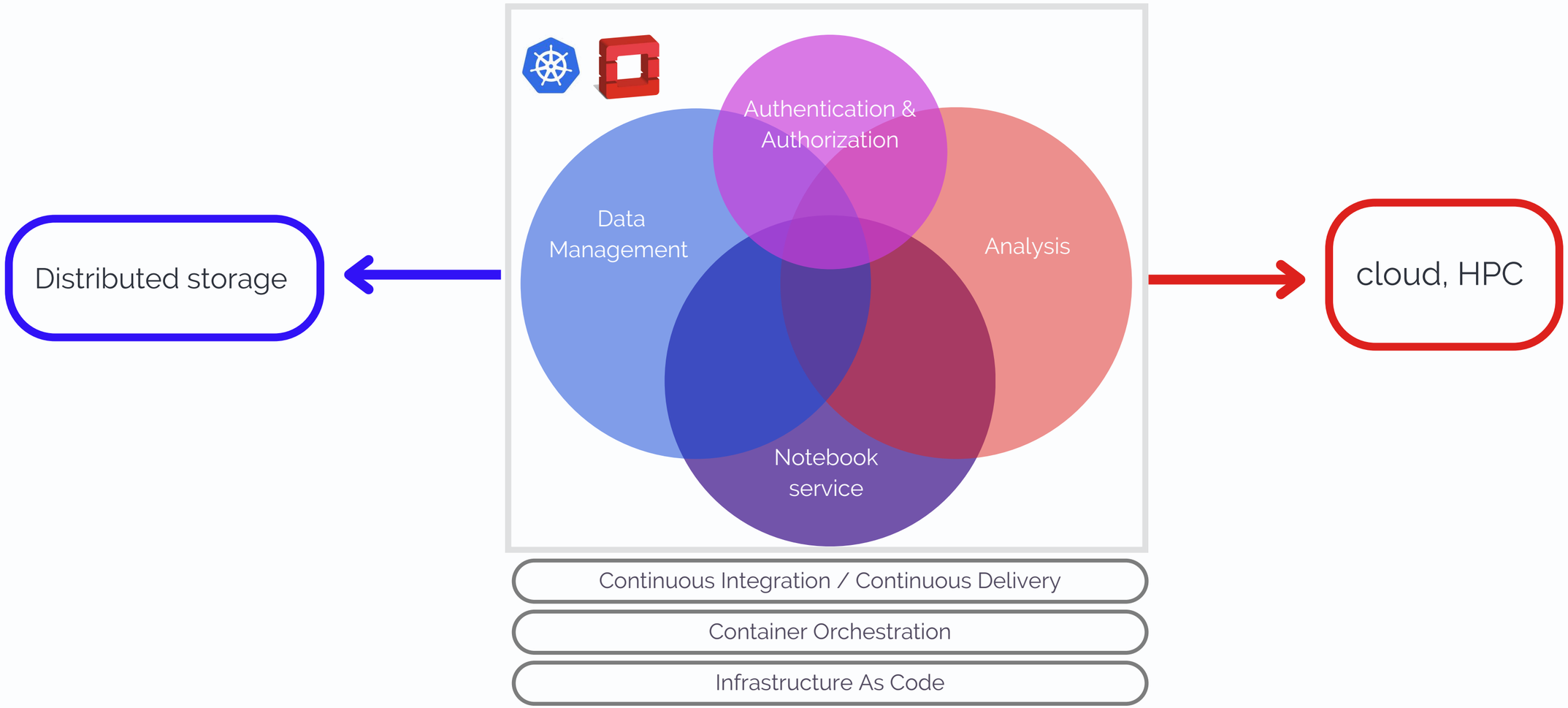
combine resutls, compare across workflows and share

Data collection → **Storage** → **Data processing** → **Analysis** → **Results interpretation & preservation**

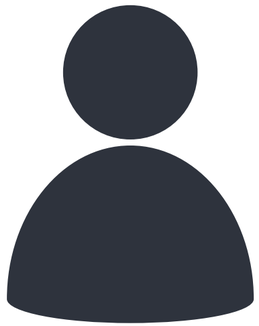
Experimental data generated at site

in data centres

The building blocks



Authentication & Authorisation



INDIGO Identity and Access Management (IAM) - adopted by WLCG for token

- Tokens
- X.509 certificates

@ INFN CNAF

Authentication & Authorization

subject mapping cronjob

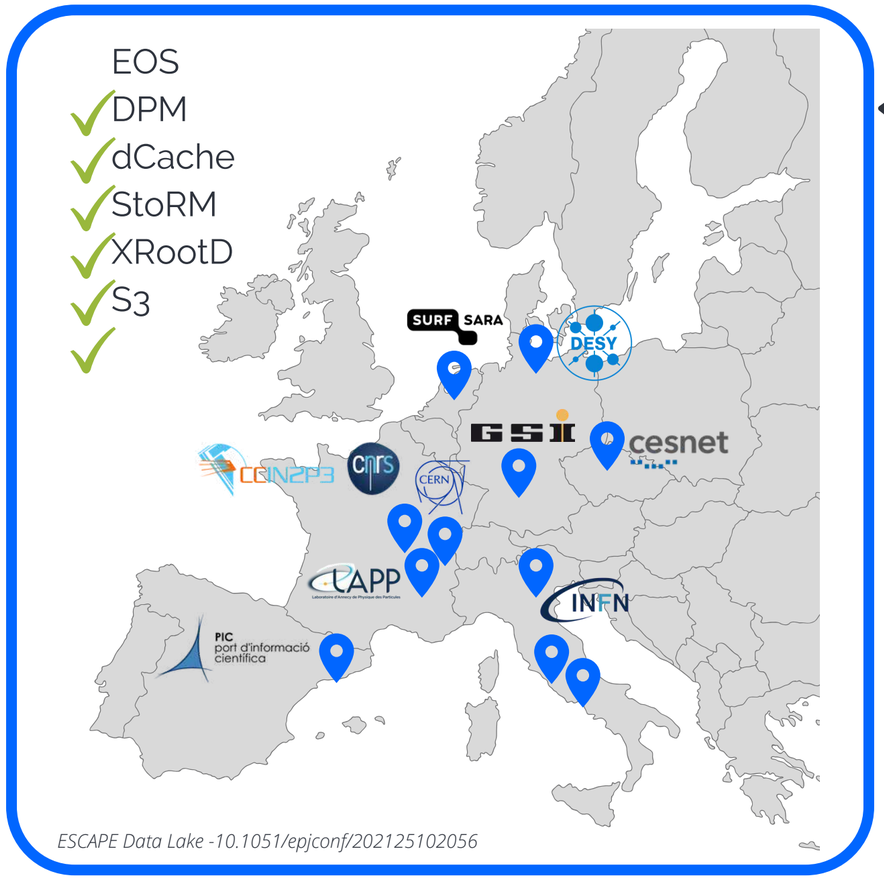
Data Management

Notebook service

Analysis

CERN EOSC Future ESCAPE European Science Cluster of Astronomy & Particle physics ESFRI research infrastructures European Commission

Data Management - ESCAPE Data Lake Infrastructure



ESCAPE Data Lake -10.1051/epjconf/202125102056

Central Relational Database

FTS3and GFAL2

Daemons for data access & replications through gridFTP, HTTP(S)/webDAV, XRoot protocols

Main & authorisation servers

Model based on WLCG

Currently being used, implemented and tested for other interdisciplinary EU projects

interTwin

Notebook Service

To facilitate **interactive analysis.**

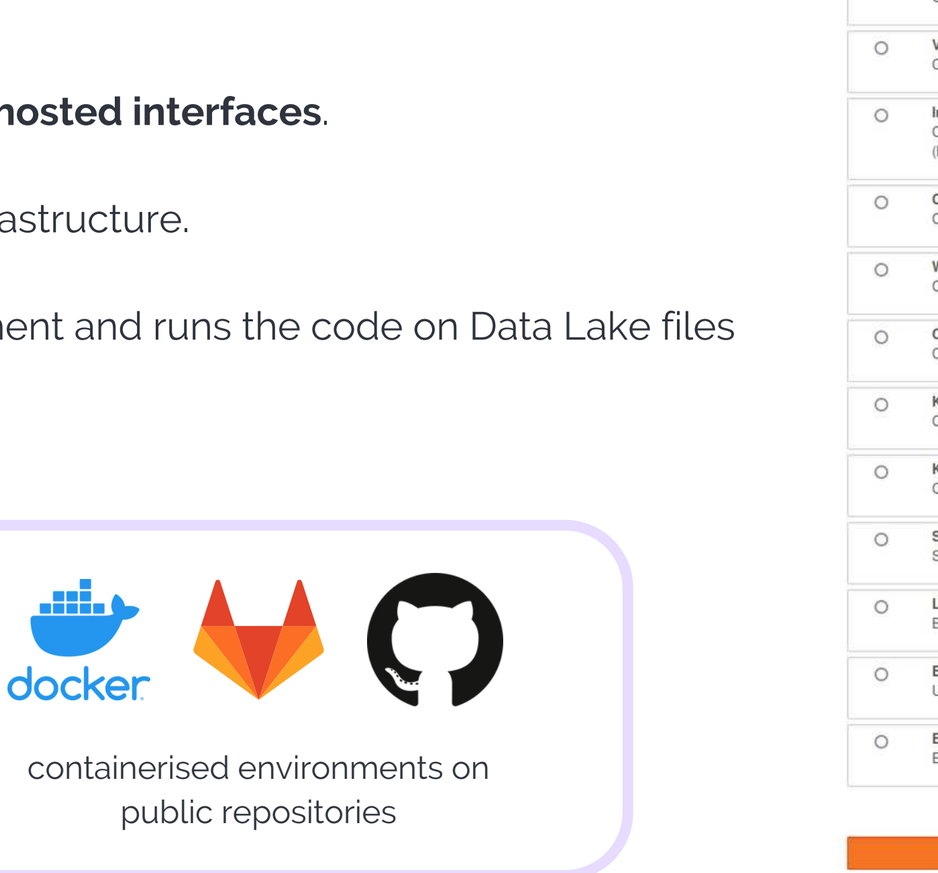
A way to run your code on **internet-hosted interfaces**.

Hides the complexity of the data infrastructure.

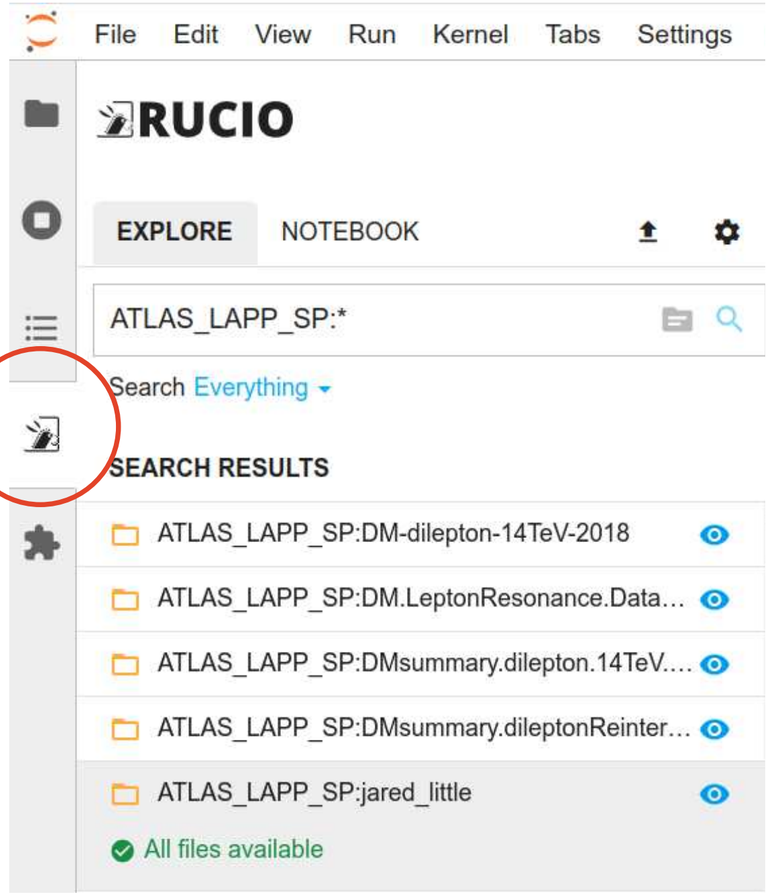
User chooses the software environment and runs the code on Data Lake files seamlessly.

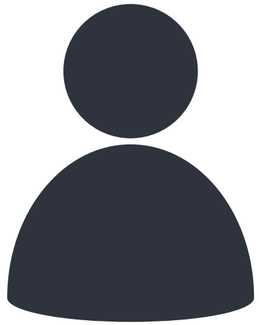
interface to run preliminary analysis

containerised environments on public repositories



Data into the notebook

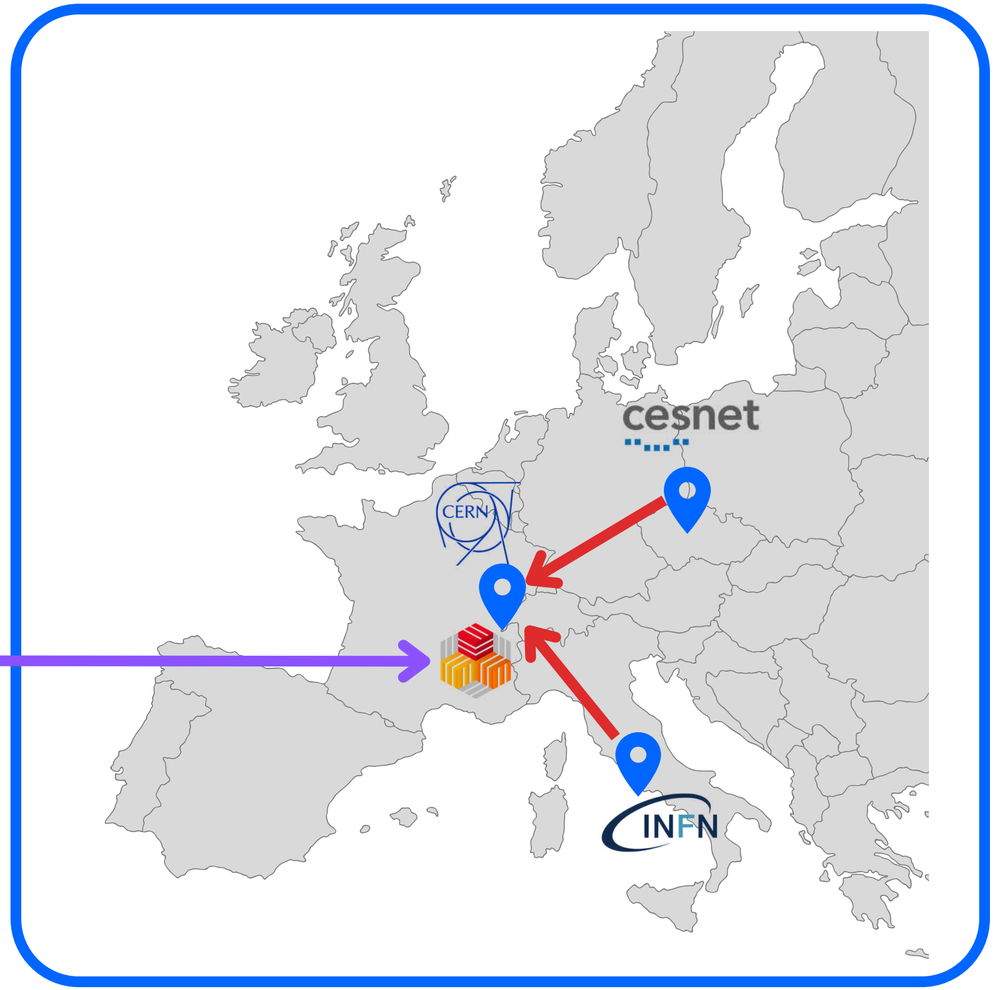




Jupyterhub node

jupyter

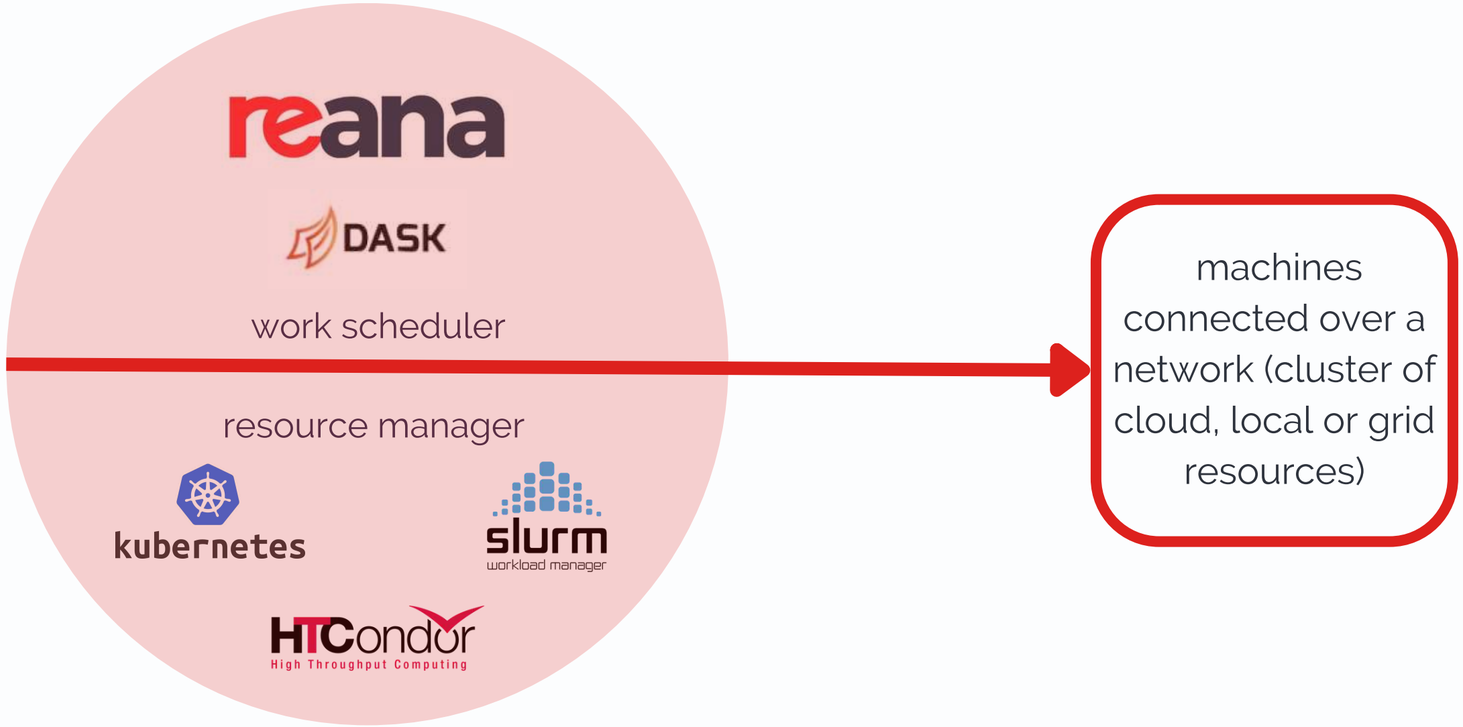
Data Lake



Computing

- **Distribute** the analysis
  - **resource managers** (Kubernetes, HTCondor (High Throughput Computing (HTC)) and Slurm (High Performance Computing (HPC))
  - **work schedulers** (Dask, Reana, Spark)

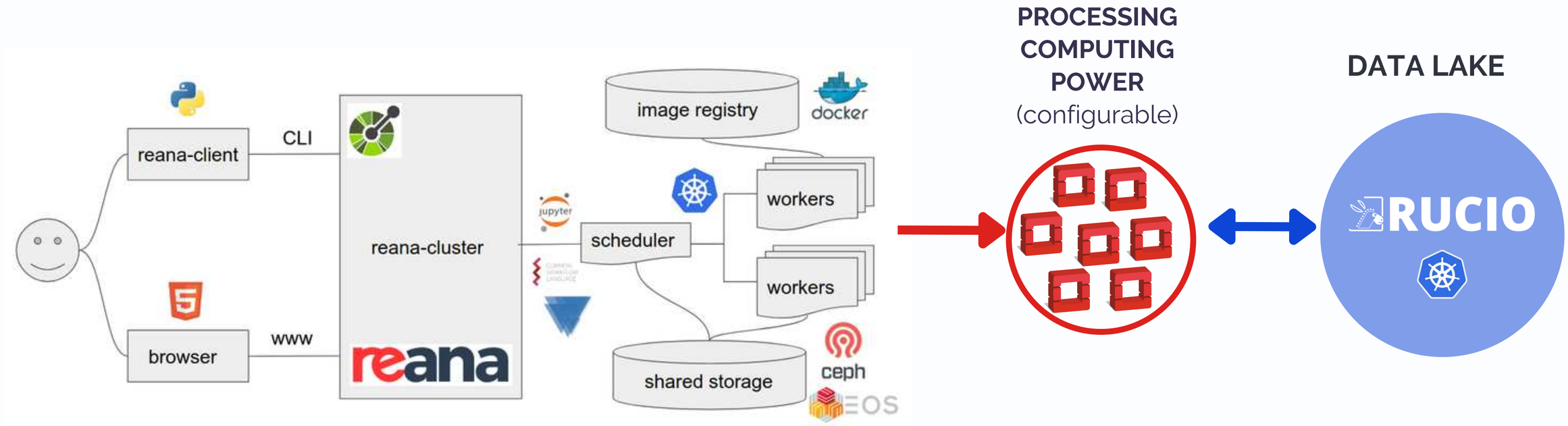
- **Preserve** the analysis for reuse
  - work schedulers (Reana)



CERN EOSC Future ESCAPE European Science Cluster of Astronomy & Particle physics ESFRI research infrastructures European Commission

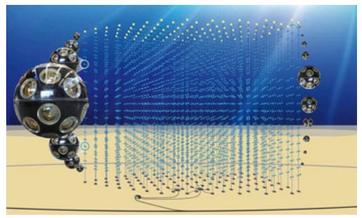
Analysis preservation and distribution

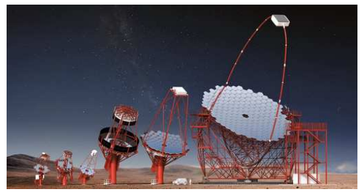
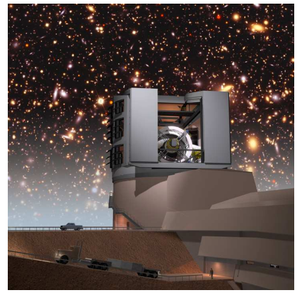
Reana makes preservation of heavier analyses seamless



The next generation of analysis facilities (VRE and others)

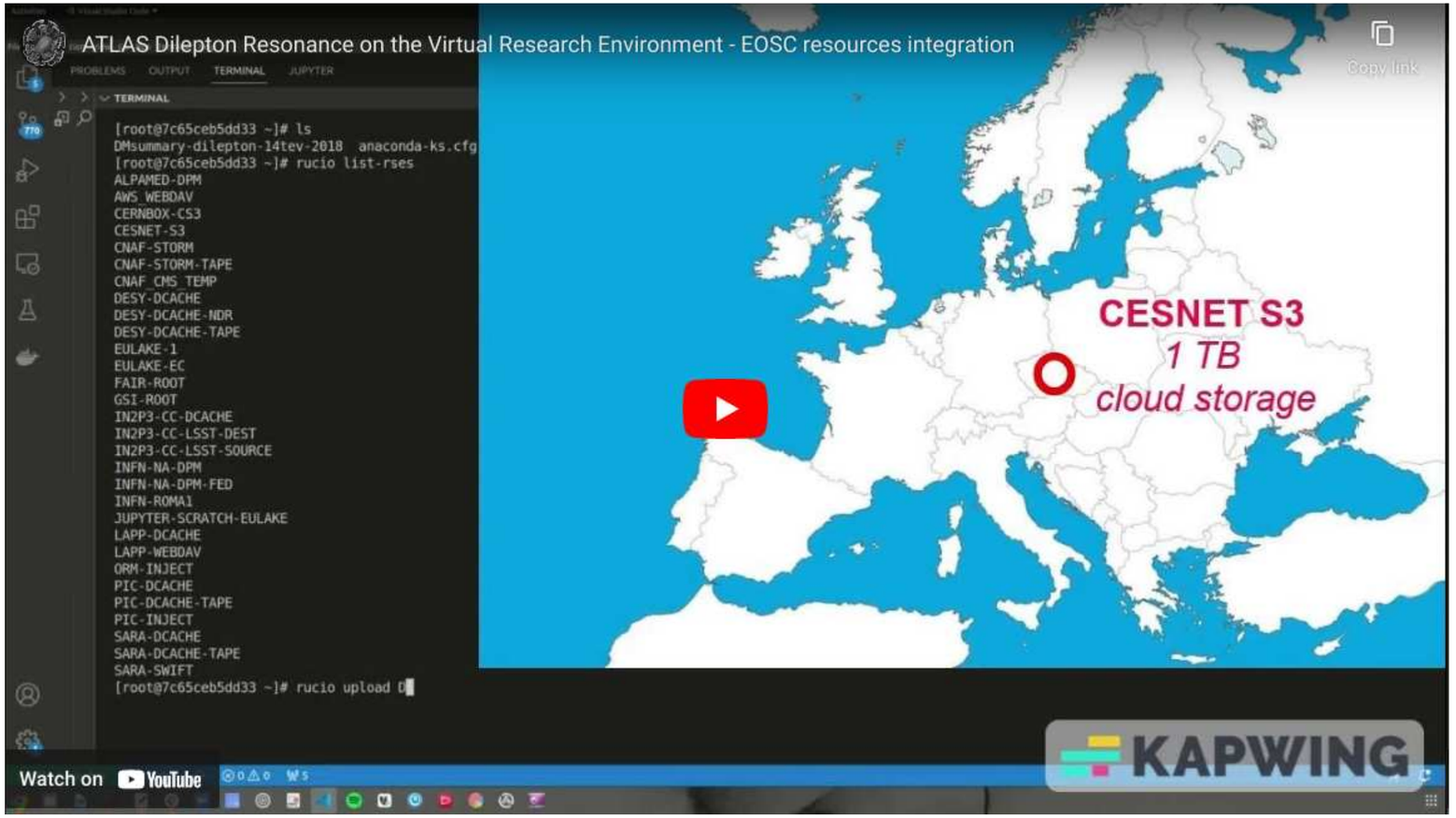
- are being deployed following **modularity** and infrastructure **sustainability** best practices to ensure longevity
- have been proven effective for **sharing scientific results** through **re-analysis** frameworks
- promote **community** building and scientific collaboration across physics domains




CERN EOSC Future ESCAPE European Science Cluster of Astronomy & Particle physics ESFRI research infrastructures European Commission

Demo



https://www.youtube.com/watch?v=nYp_wsXhKSo&ab_channel=ElenaGazzarrini

Thank you for your attention

e-mail

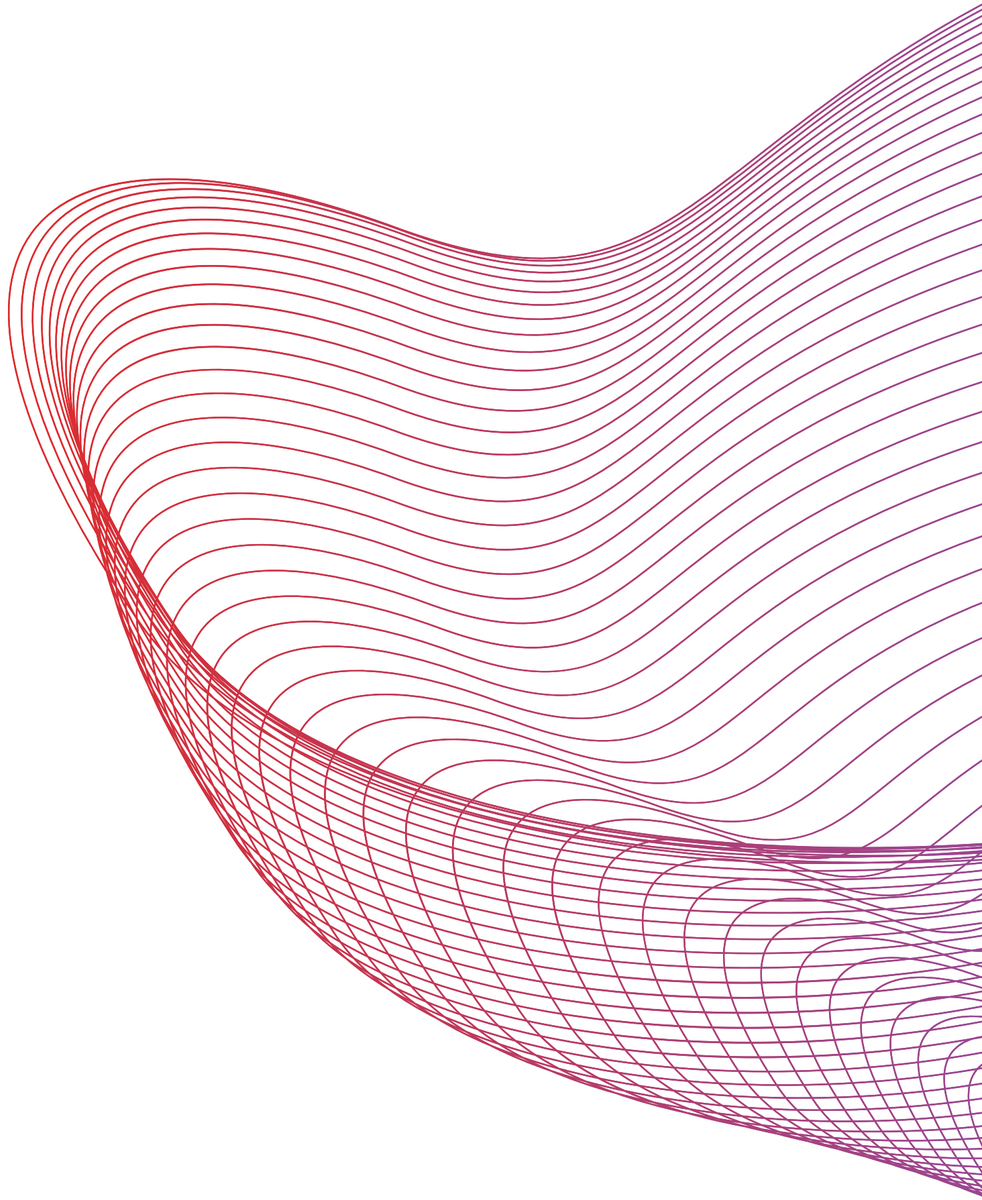
enrique.garcia.garcia@cern.ch

contact the full VRE team

CERN Meyrin site -513/1-011

Elena Gazzarrini, Domenic Gosein, Enrique Garcia & Xavier Espinal

escape-cern-ops@cern.ch



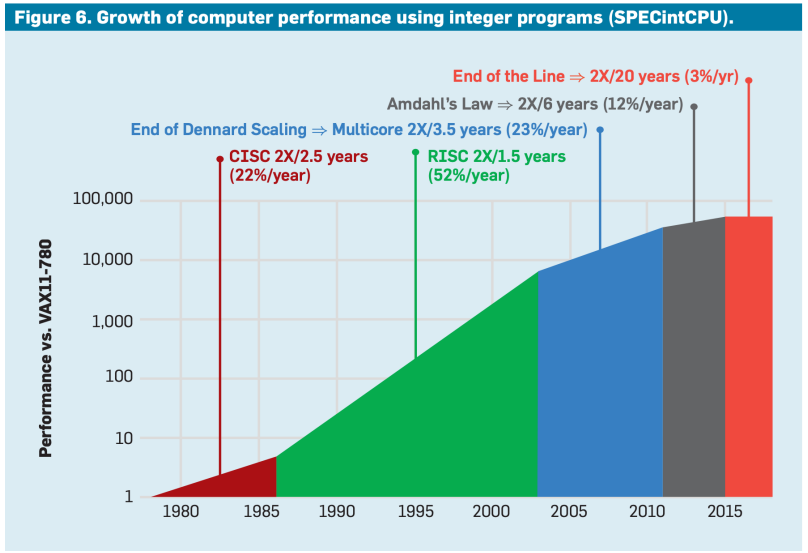
CERN EOSC Future ESCAPE European Science Cluster of Astronomy & Particle physics ESFRI research infrastructures European Commission

Data's Need for Speed: InfiniBand and RDMA Driving HPC Evolution

Marco Faltelli

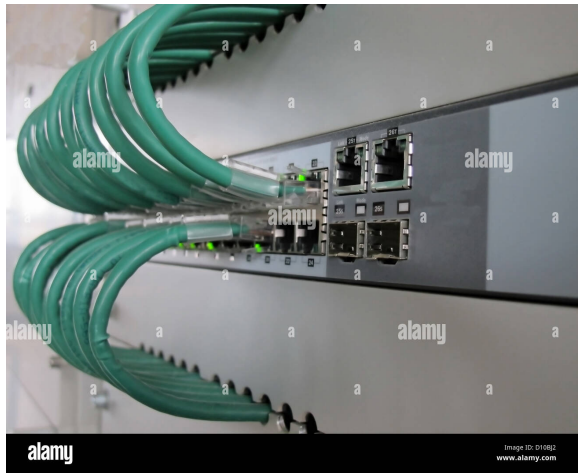
Context

Source: Hennessy, Patterson



- Moore's Law, remember?
  - CPU performance reaching a stagnation point
- HPC & Big data era: improving performances is required
- Not all jobs can be parallelized!
- Key idea: accelerate by offloading the CPU from tasks

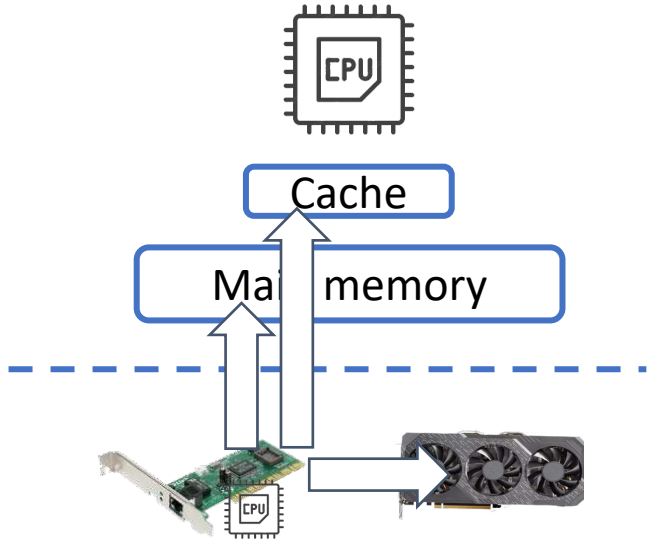
New link mechanisms

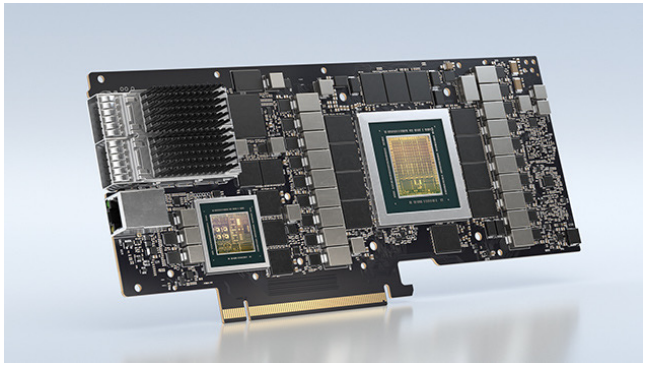


- Ethernet: born in the ‘70s…
- Modern versions are still **general-purpose**
  - Work on both your home network and a Data Center!
- You can lose packets!
- **Infiniband**: lossless link
  - Key idea: move loss-prevention mechanisms on HW
  - Packet drops are very rare
- **Converged Ethernet**: same thing

NICs to the rescue

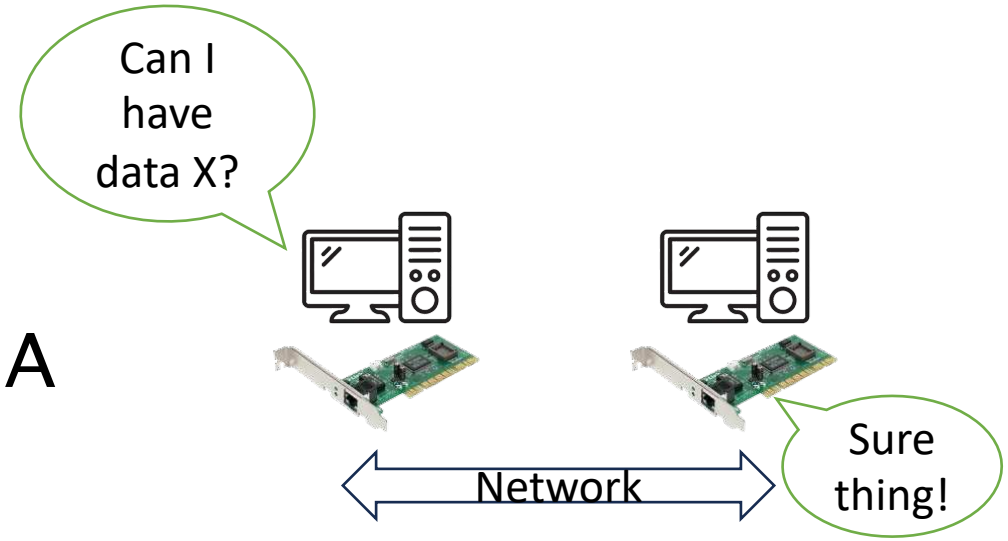
- Network Interface Cards
  - HW used to send/recv data from the network
- New NIC capabilities
  - Computation (cores on the NIC!)
  - Direct memory access
  - Direct L3 cache access
  - GPUdirect
- NICs emerging as a new device for computation





The new NVIDIA Bluefield 3

Merging the two



- Lossless link + modern NICs = RDMA
- **Remote Direct Memory Access**
- **RDMA** enables direct access to a remote computer's memory
  - NIC has direct access to main memory
  - It bypasses the responder's CPU
- CPU is source of many delays
  - Interrupt, schedule thread, initiate transfer to the NIC…

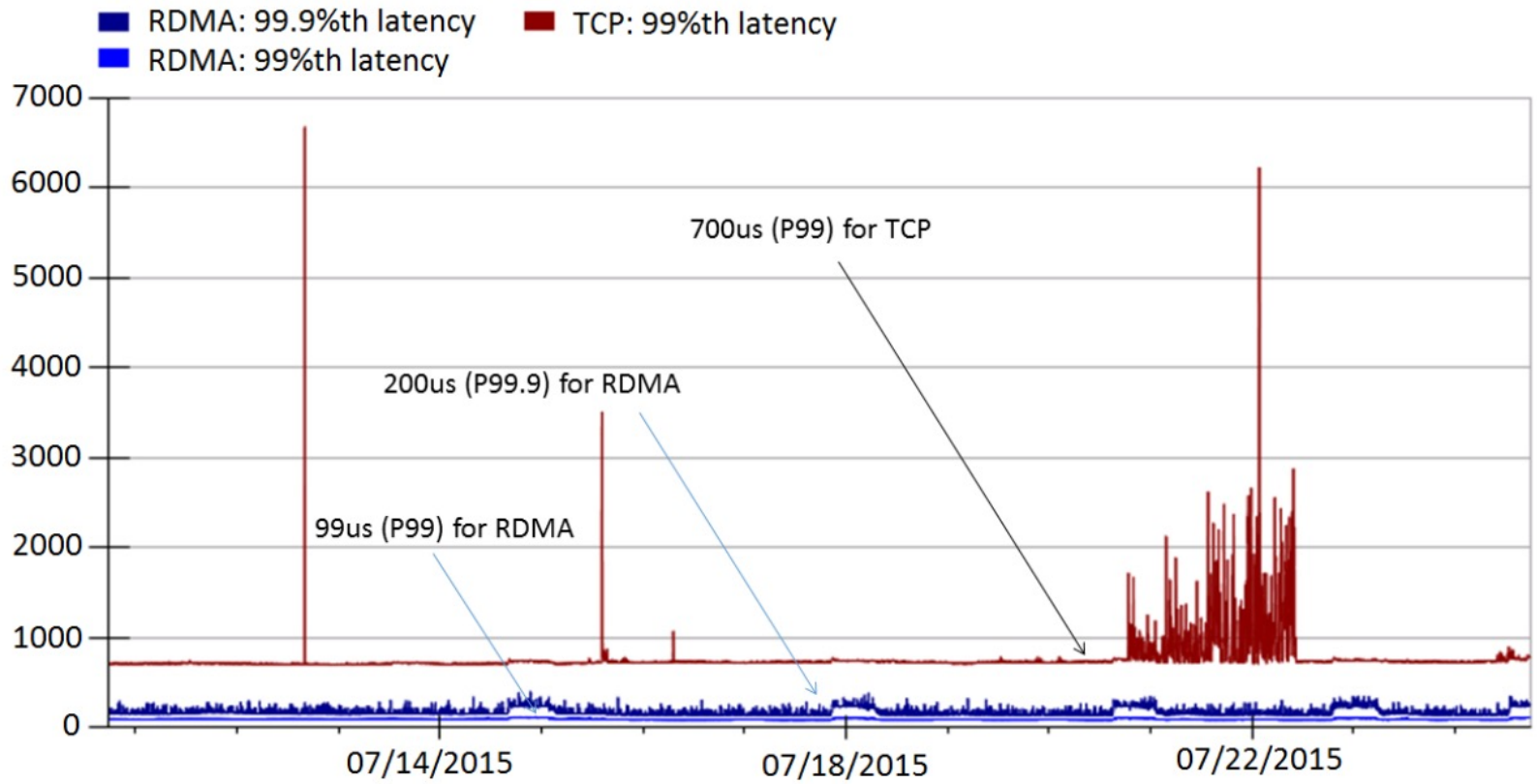
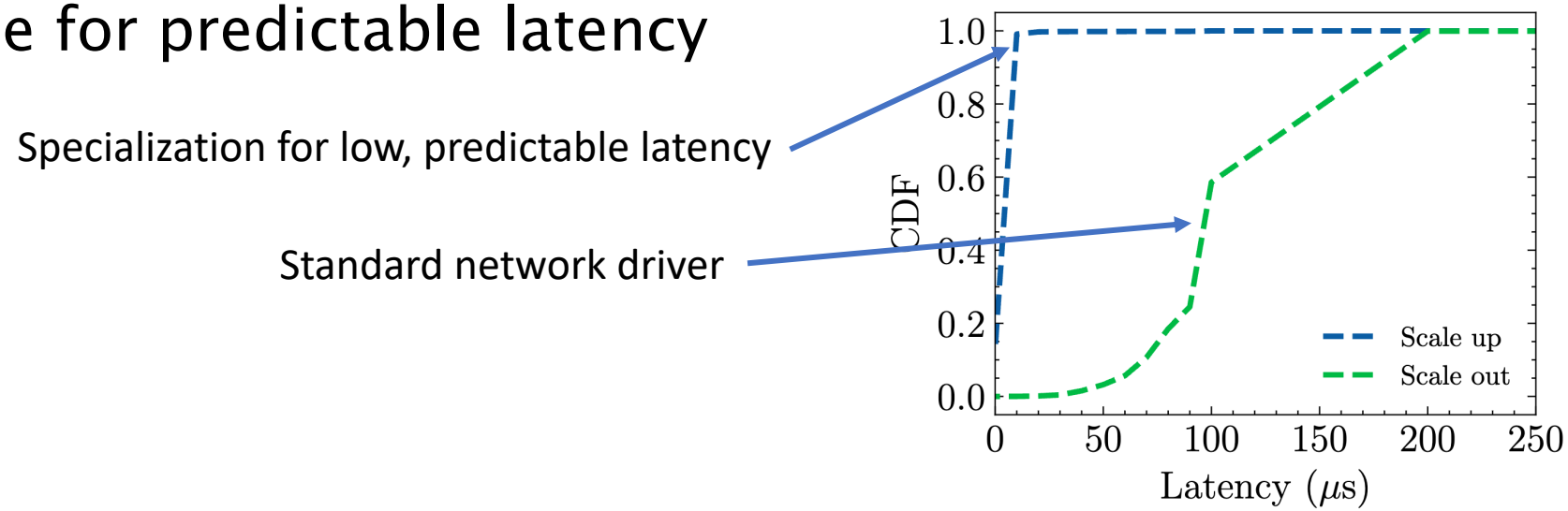


Figure 6: The comparison of the measured TCP and RDMA latencies for a latency-sensitive service.

From Guo et al., “RDMA over Commodity Ethernet at Scale”

Next steps

- RDMA is currently a black box
  - Install it, run it, see performance improving
- What happens under the hood?
- Can we specialize the software for specific workloads?
- My experience with network drivers:
  - Specialize for predictable latency



Thank you!

Questions?

CernVM-FS for Efficient Software Distribution at CMS

Andrea Valenzuela Ramírez
andrea.valenzuela.ramirez@cern.ch

CSC 2023 - Lightning talk

September 1st, 2023

Outline

CernVM-FS File System

- CernVM-FS is a scalable, reliable and low-maintenance **software distribution service**.
- It mounts repositories into a virtual `/cvmfs` directory tree.
- It is asymmetric by construction: **many reads and a few writes**.

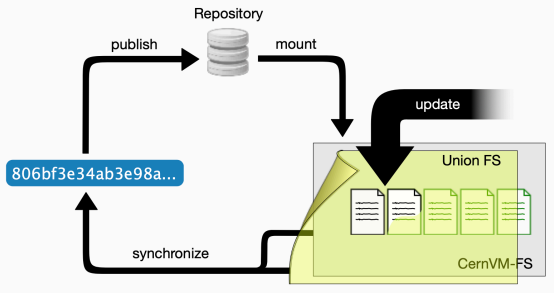


Figure: Updating a mounted CernVM-FS repository.

Classic Workflow

The publication architecture involves a **single publisher** per repository (*release manager*).

Parallel Workflow

The *Gateway* allows **multiple publisher** concurrently on the same repository.

- Publishing content into CernVM-FS works by **transactions** to ensure atomic writes.

CMS Offline Software (CMSSW)

- CMSSW contains the **software collection** needed to process event data offline at CMS.
  - It includes simulation, calibration, alignment and reconstruction modules.
  - Same codebase for High Level Trigger (HLT).

CMSSW Code Base

- 100+ Contributors/month
- 500+ Commits/month
- 1250+ Packages
  - 3300+ Binary products

Figure: GitHub repositories within the `cms-sw` organization.

Maintaining the Software Stack

- Maintaining and incorporating changes in a big software stack can be a challenging task.
- Three testing strategies to ensure consistency of the software stack:
  - **Continuous Integration** (CI): Testing changes before incorporating them to the software stack.
  - **Integration Builds** (IBs): Building and testing the software stack regularly.
  - **Production Software**: Each CMSSW Release also goes through a testing procedure before being released.

Figure: The software stack is regularly built and tested through Jenkins.

Continuous Integration

Contributing to CMSSW

- Contributions are made by Pull Requests (around **100 PRs/week**.).
- Automatic testing before merging.
- Multiple possibilities of testing:
  - Test PR picking up changes from a PR on another repository.
  - Enable extra PR tests: gpu, threading, profiling, etc.
  - Test for a concrete CMSSW Release.

▸ Test parameters

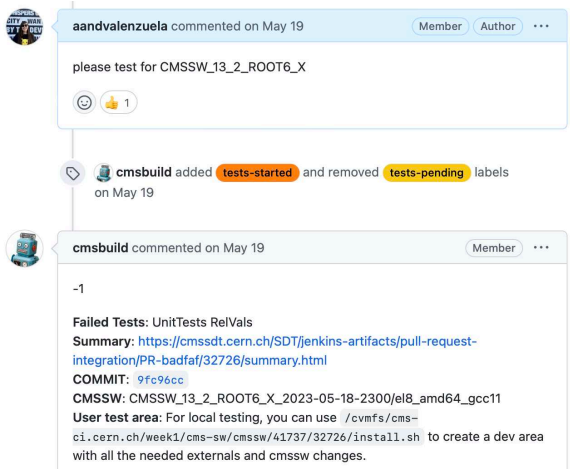


Figure: Starting the PR testing for a concrete CMSSW flavor.

Continuous Integration

- The testing procedure is automatised using the `cms-bot`.
- Since multiple actions are required, there is a system of labels to track the process.
  - Automatic labeling based on the touched components.
  - Actions are restricted to certain users.

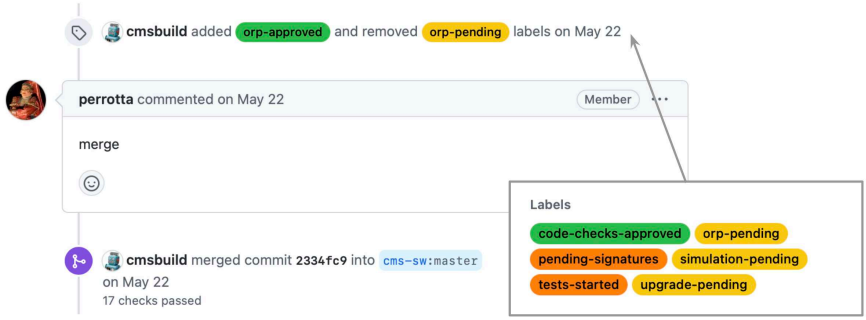


Figure: Only authorized users can trigger certain "critical" actions.

Integration Builds

- Automatically deployed via Jenkins **every 12 hours**.
- Build for a set of active release cycles, multiple OSs, architectures and compiler versions.
- Build different IB "flavors".
  - e.g., CLANG builds for static analysis.

IB page

	DEFAULT					UBSAN_X	SKYLAKEAVX512_X	ROOT6_X	ROOT628_X	NONLTO_X	GPU_X	GEANT4_X
	el9 amd64 gcc11 Full Build	el8 aarch64 gcc11 Full Build	el8 ppc64le gcc11 Full Build	slc7 amd64 gcc11 Full Build	el8 amd64 gcc11 Full Build	el8 amd64 gcc11 Full Build	el8 amd64 gcc11 Full Build	el8 amd64 gcc11 Full Build	el8 amd64 gcc11 Full Build	el8 amd64 gcc11 Full Build	el8 amd64 gcc11 Full Build	el8 amd64 gcc11 Full Build
Builds	✓	✓	✓	✓	✓	1	✓	✓	✓	✓	✓	✓
Unit Tests	✓	✓	✓	✓	✓	1	✓	✓	✓	✓	1	✓
RelVals	2	56	26	2	4170	14*	4080*	4170	4170	4170	213	1
Other Tests	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓
Q/A	🔍	🔍	🔍	🔍	🔍	🔍	🔍	🔍	🔍	🔍	🔍	🔍

Figure: Around **60 IBs** are build and deployed **every day**.

- Once a week, fully build all release cycles.

Software Releases

Building CMSSW Releases

- CMSSW Releases are built on-demand.
- Automatic end-to-end process:
  - Build for multiple OSs, architectures and compiler versions.
  - Test and installation.
  - Upload to the server.
  - Generation of the release notes.
  - Cleanup.
- Actions restricted to authorized users.
- Label system to track the process.

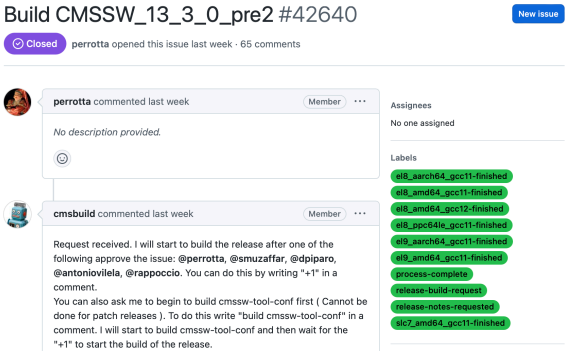


Figure: Building CMSSW Release from GitHub.

Software Releases

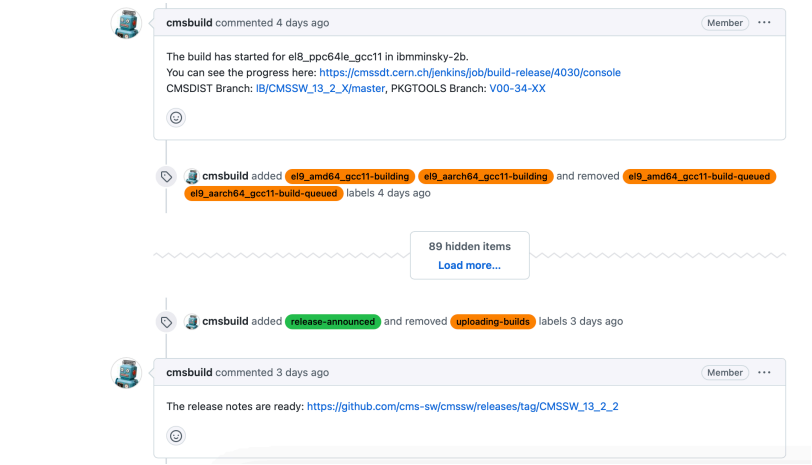


Figure: Overview of the automatic process on GitHub for building a new CMSSW release.

CernVM-FS Use Cases at CMS

CernVM-FS is one of the main building blocks supporting this pipeline:

- Distribution of **Continuous Integration** artifacts.
  - Speeds up the development iterations.

```
[avalenzu@lxplus7123 cmssw]$ ls /cvmfs/cms-ci.cern.ch/week0/PR_2253d79f
LICENSE  bootstrap.sh  bootstraptmp  cmsset_default.csh  cmsset_default.sh  common  cvmfs  el8_amd64_gcc11  etc  share
```

- Distribution of **Integration Builds**.
  - Supports the creation of a development area from the most recent snapshot of CMSSW.

```
[avalenzu@lxplus7123 cmssw]$ ls /cvmfs/cms-ib.cern.ch/sw/x86_64/week0/el8_amd64_gcc11/cms/cmssw/
CMSSW_12_6_X_2023-08-27-0000         CMSSW_13_3_DEVEL_X_2023-08-27-2300      CMSSW_13_3_ROOT628_X_2023-08-30-2300
CMSSW_13_0_GPU_X_2023-08-27-2300     CMSSW_13_3_DEVEL_X_2023-08-30-2300      CMSSW_13_3_ROOT6DBG_X_2023-08-30-1200
CMSSW_13_0_X_2023-08-27-0000         CMSSW_13_3_G4VECGEOM_X_2023-08-27-2300  CMSSW_13_3_ROOT6_X_2023-08-27-2300
CMSSW_13_1_GPU_X_2023-08-27-2300     CMSSW_13_3_G4VECGEOM_X_2023-08-30-2300  CMSSW_13_3_ROOT6_X_2023-08-29-2300
```

- Distribution of experiment **Production Software**.
  - Releases are heavily used by Grid production and user analysis jobs, outreach, CMSSW developers, etc.

```
[avalenzu@lxplus7123 cmssw]$ ls /cvmfs/cms.cern.ch/el8_amd64_gcc11/cms/cmssw/
CMSSW_12_5_0                    CMSSW_13_0_0_pre1              CMSSW_13_0_8
CMSSW_12_5_0_pre2               CMSSW_13_0_0_pre1_LTO          CMSSW_13_0_9
CMSSW_12_5_0_pre3               CMSSW_13_0_0_pre2              CMSSW_13_1_0
CMSSW_12_5_0_pre4               CMSSW_13_0_0_pre2_LTO          CMSSW_13_1_0_SKYLAKEAVX512_pre3
```

Conclusion

- CernVM-FS is crucial for CMSSW.
  - It helps in development, distribution and preservation of the software.
- Maintaining the CMS Offline Software stack can be challenging.
  - We have developed automatic end-to-end testing pipelines to ensure consistency of the stack based on GitHub comments (actions) and labels.
  - Our current infrastructure covers PR testing, Integration Builds and on-demand testing for CMSSW Releases.
- CernVM-FS helps us in distributing all the resulting artifacts to CMSSW end-users:
  - Speeds up development tasks.
  - Helps in debugging and bug fixing.
  - Facilitates the access to the software stack.

Questions? :)

▸ More Information on CMS use cases

CernVM-FS Use Cases at CMS

The CMS collaboration deploys the CMS Offline Software (CMSSW) to CernVM-FS under different use cases:

- Distribution of experiment **Production Software** (CMSSW Releases).
- Distribution of **Integration Builds** (IBs).
- **Continuous Integration** (CI) purposes.

Repository Name	Size	Garbage Collection	Storage	Revision	Year
`/cvmfs/cms.cern.ch`	∼19 TB	No	S3	117846	2009
`/cvmfs/cms-ib.cern.ch`	2.77 TB	Yes (weekly)	S3	258447	2016
`/cvmfs/cms-ci.cern.ch`	711 GB	Yes (weekly)	S3	39895	2020

Table: CMS main repositories and their characteristics in terms of size, garbage collection frequency, type of storage, number of commits and year of creation.

CernVM-FS Use Cases at CMS

Other CernVM-FS deployments within CMS Offline & Computing include:

- CMSSW **container images** (distributed in `unpacked.cern.ch`).
- Private distribution of **CUDA build time components** (distributed in `projects.cern.ch`).
  - The CMSSW CUDA distribution did not comply with the Nvidia End User License Agreement.
  - CUDA runtime can still be packaged and distributed along with CMSSW release.
  - Build time components, e.g. nvcc compiler, are now deployed on a compliant repository.

  ▸ Related talk
- Distribution of Gridpacks (lookup files) in the form of tarballs.
  - Gridpacks are "pre-computed diagrams" that speed-up Monte Carlo generation.
  - Distributed in tarballs, they are uncompressed for every generator job on local disk. **Many sites do not support such operation**.
  - A solution could be serving untarred Gridpacks via CernVM-FS.
  - Suitable use-case for the `cvmfs_server ingest` command.

  ▸ Related news

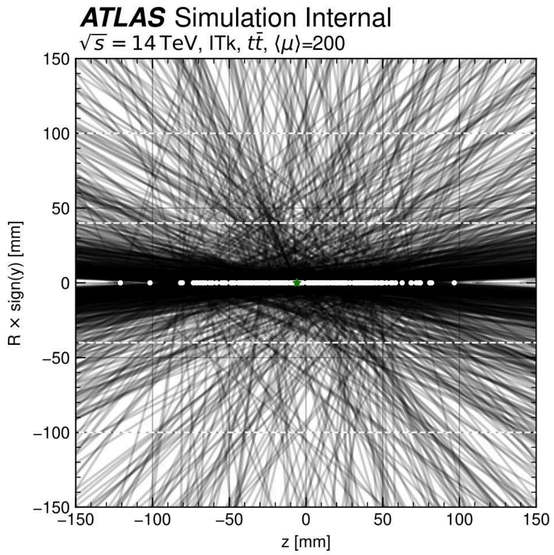
Investigating the impact of 4D Tracking in ATLAS Beyond Run 4

Lorenzo Santi

SAPIENZA
UNIVERSITÀ DI ROMA

Istituto Nazionale di Fisica Nucleare

The Pile-Up Challenge



no time

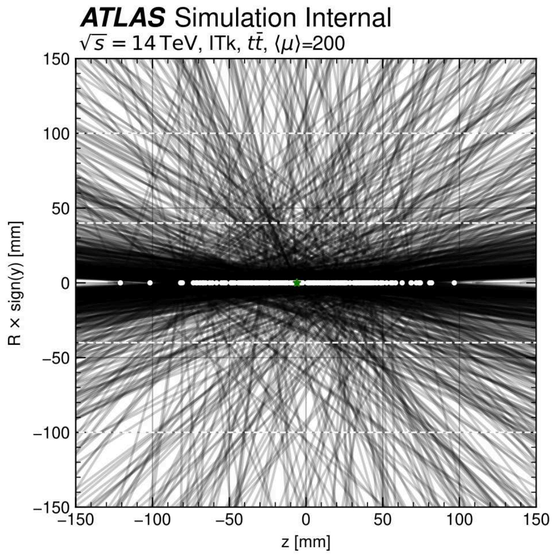
From 2029 HL-LHC starts:

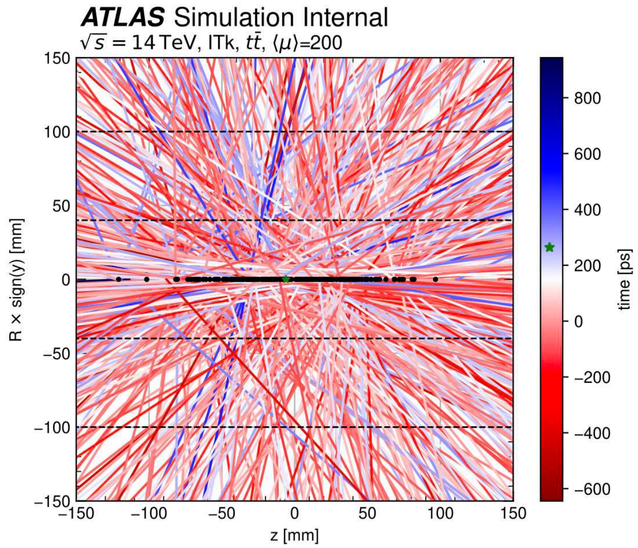
200 interaction per bunch crossing

Huge computational challenge

Tracking relies only on spatial info

The Pile-Up Challenge





no time → **time**

From 2029 HL-LHC starts:

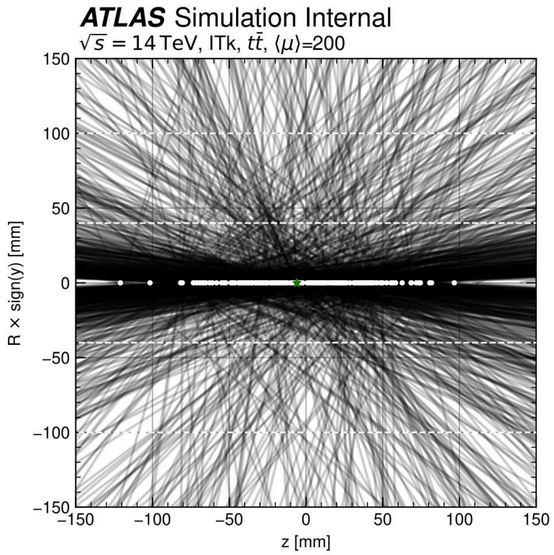
200 interaction per bunch crossing

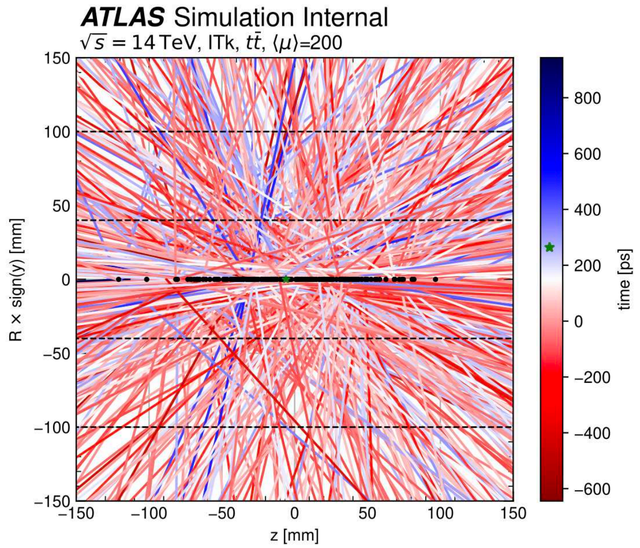
Huge computational challenge

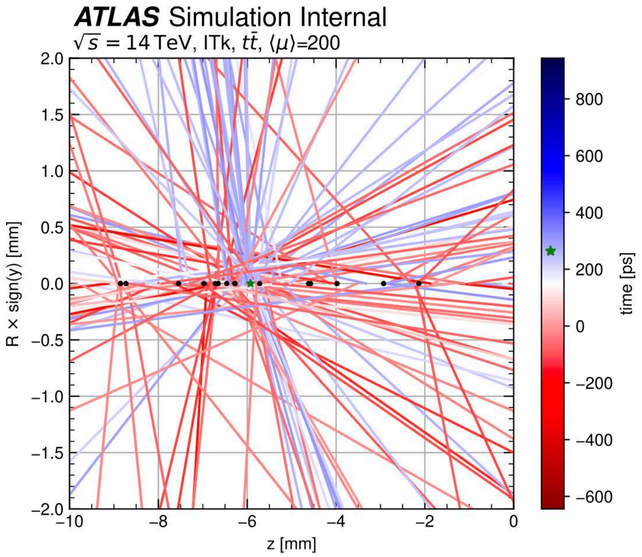
What if we are able to measure time?

Clearly simulation only

The Pile-Up Challenge







no time → **time** → **zoom z**

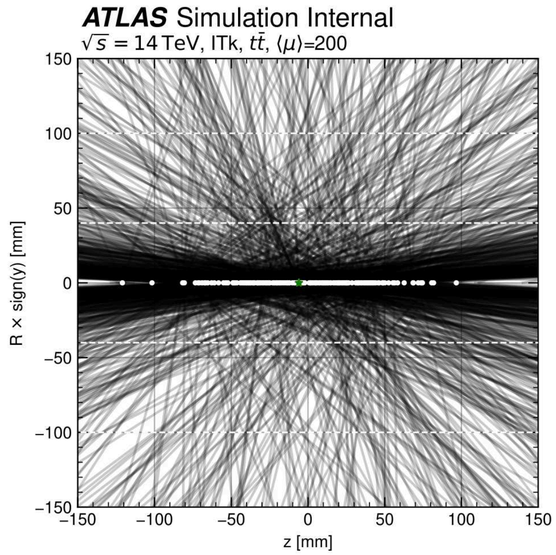
From 2029 HL-LHC starts:

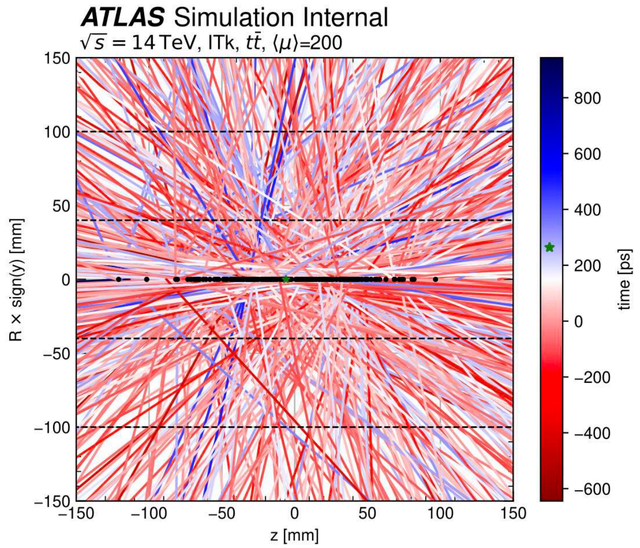
200 interaction per bunch crossing

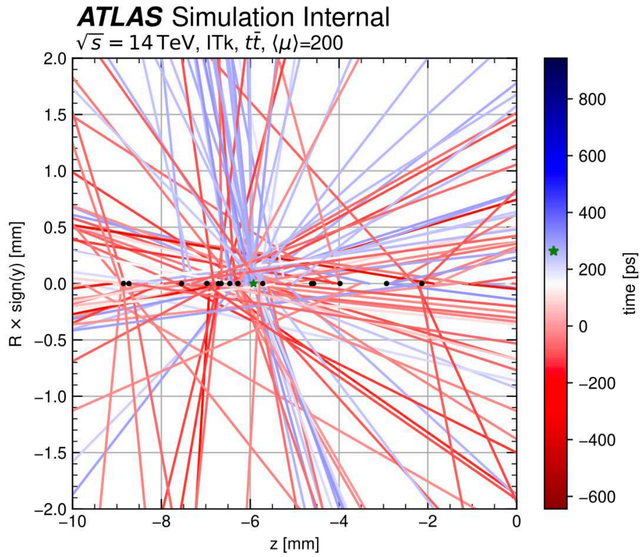
Huge computational challenge

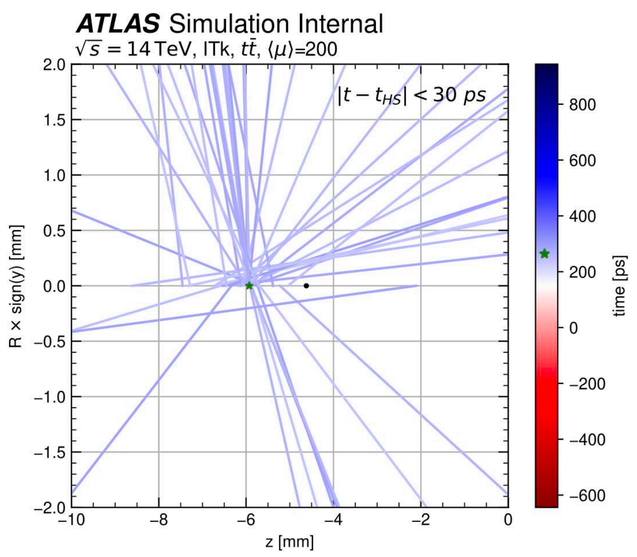
If we open a window in space …

The Pile-Up Challenge









no time → **time** → **zoom z** → **zoom t**

From 2029 HL-LHC starts:

200 interaction per bunch crossing

Huge computational challenge

If we open a window in space …

… and in time

The event look much clean

Motivations

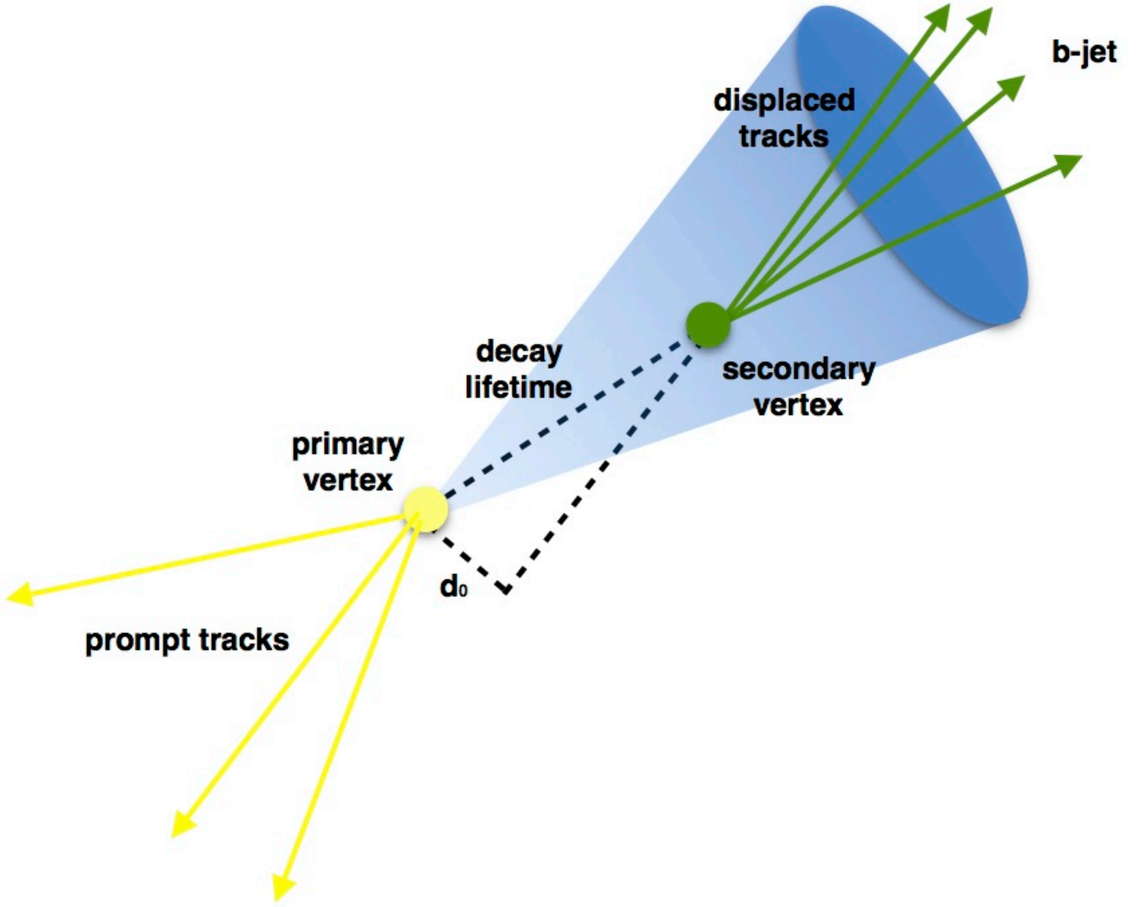
We want to asses the impact on Physics objects with timing information:

Jet Flavour Identification (FTAG)

Set of algorithms aiming to identificate the flavour of a parton originating a Jet

In particular we distinguish:

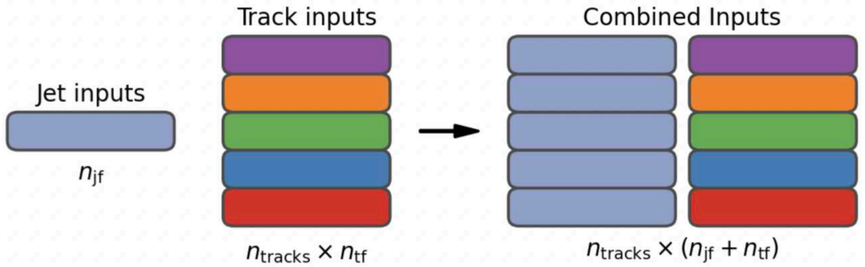
- b-jets
- c-jets
- light-jets: u/d/s/g-jets

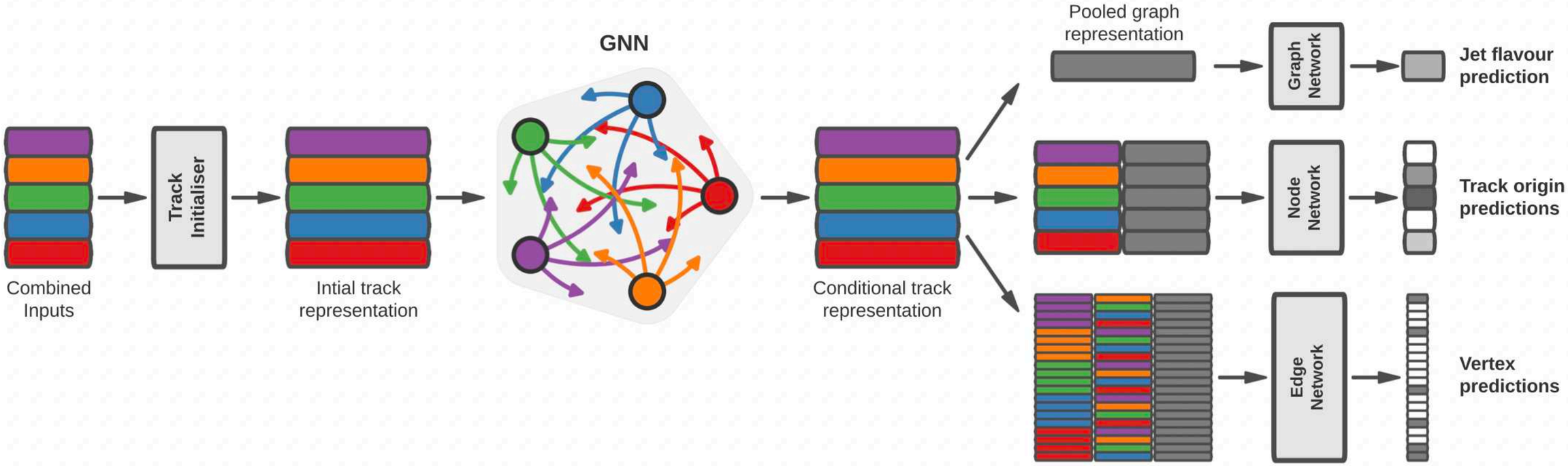


Jet Flavour Identification: GN1

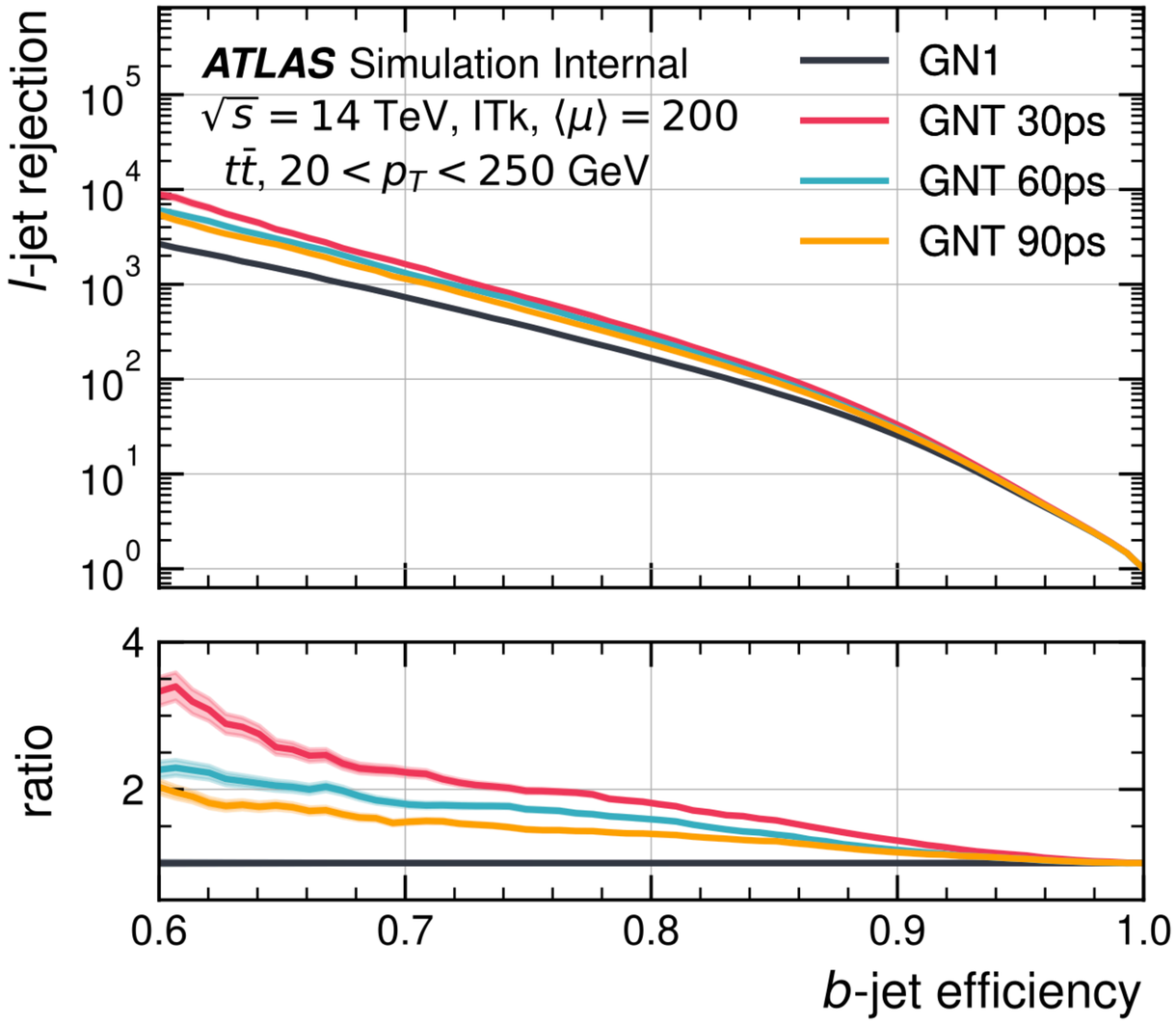
GNN Note

How to do it:





Performances



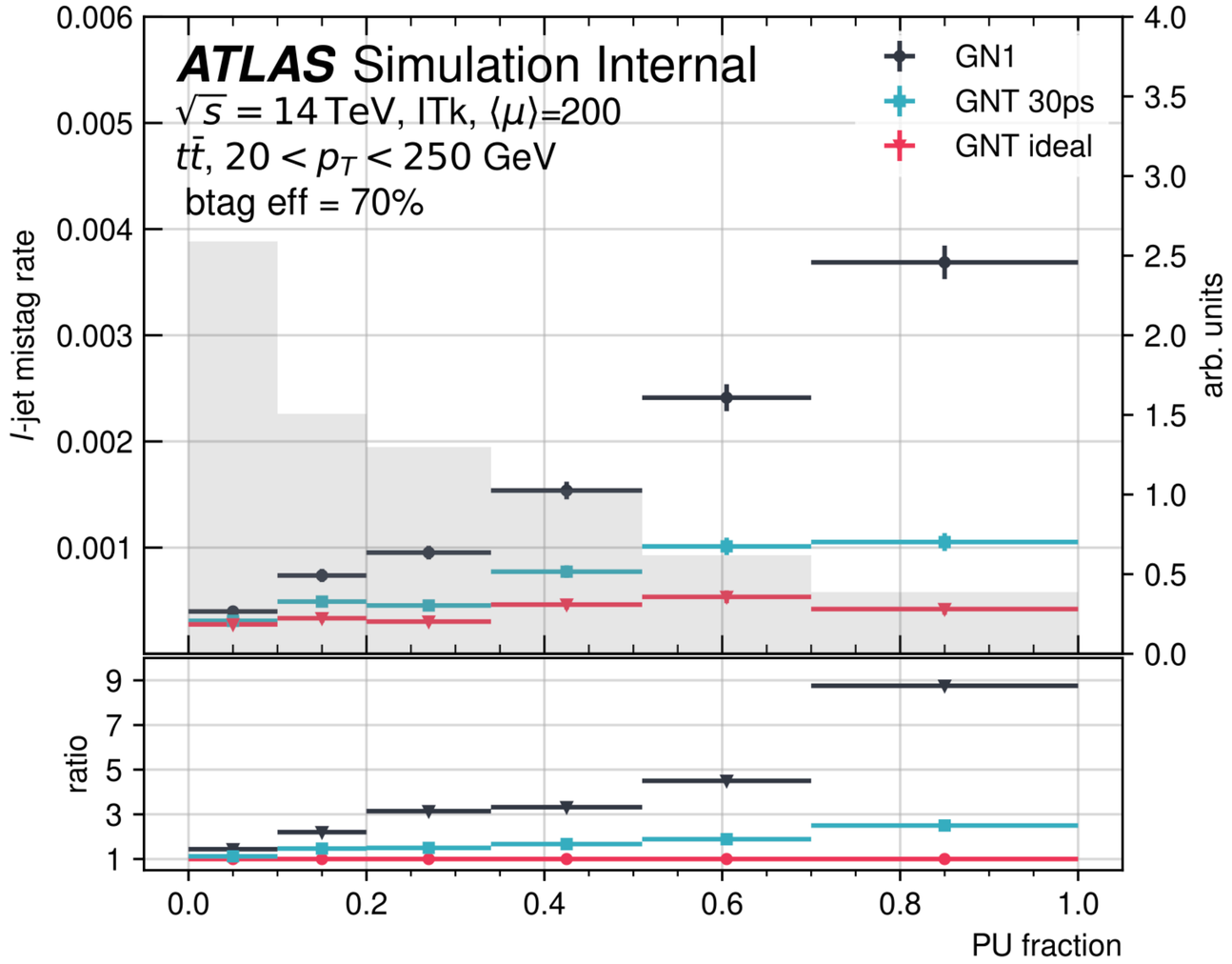
This is a proof of concept

We want to show which is the impact of time on top the state-of-the-art FTAG algorithm: GN1

ROC curve shows how many light-jets we are able to discard for a given b-jet efficiency:

the higher the better ;)

Performances



New PU discriminating variable (per jet):

$$\text{PU fraction} = = \frac{\#trk_{PU}}{\#trk}$$

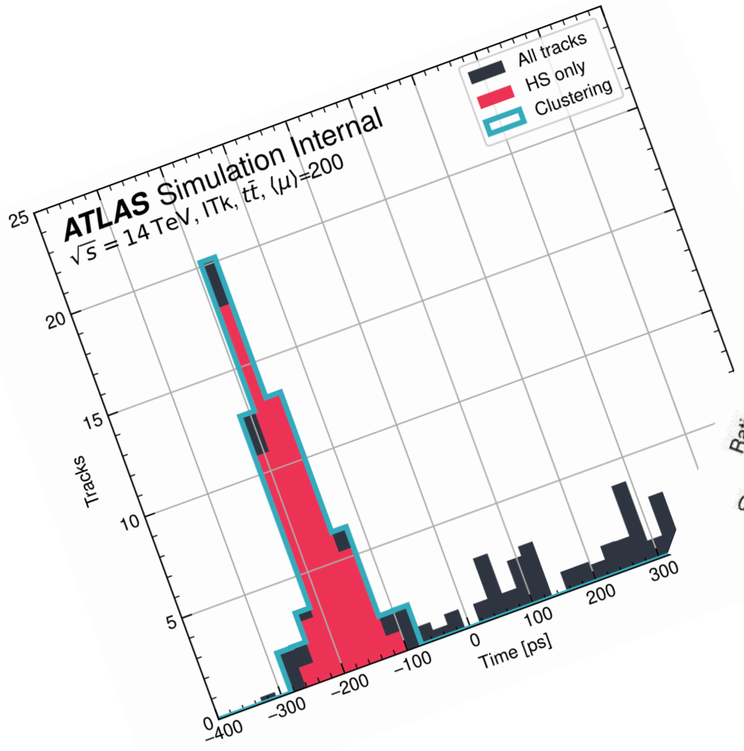
Large improvement in highly PU contaminated jets

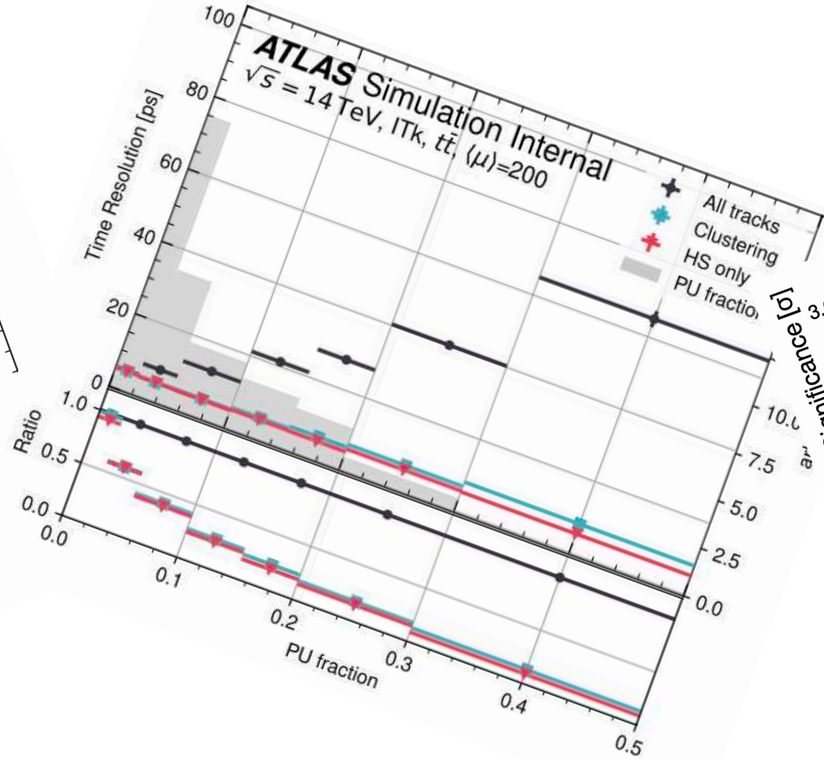
l-jets mistag rate gets flattened with time information

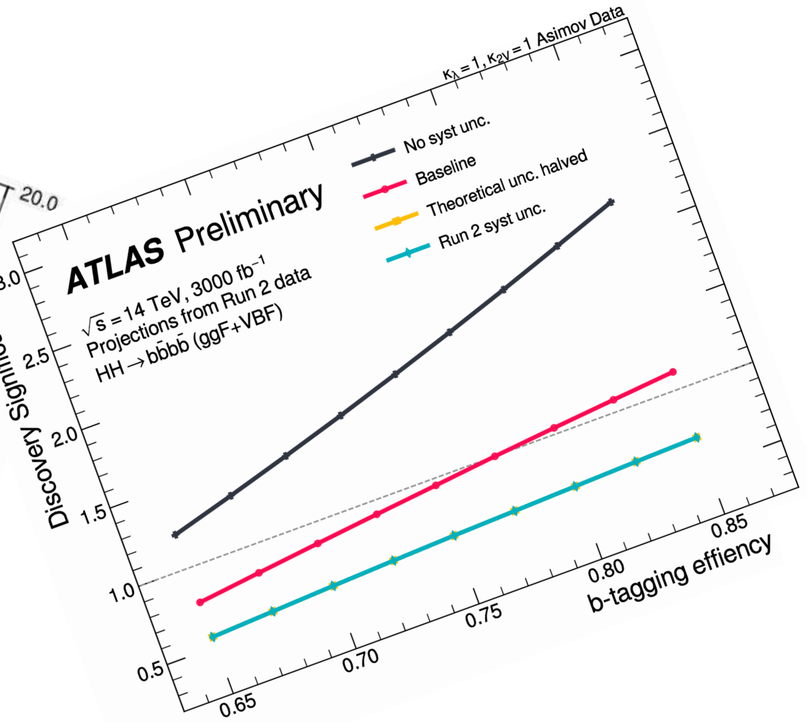
Conclusions

This study is the first study of potential impact of hermetic timing coverage in the tracker of ATLAS

Much more impact that cannot be squeezed in here but can be discussed







DL_FIELD

My journey on refactoring and translating legacy code

Name of Project:
Legacy code Conversion

Presented By:
Elziabeth Mamchits

Presented At:
CSC 2023

Motivation and Challenges

Clarify the project's main overall objectives and goals.

Improve maintainability

Reduce Technical Debt

Enhance Functionality

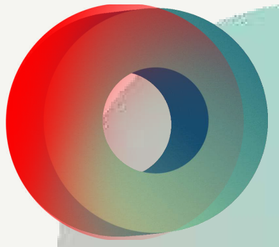
Motivation and Challenges

Clarify the project's main overall objectives and goals.

Improve maintainability

Reduce Technical Debt

Enhance Functionality



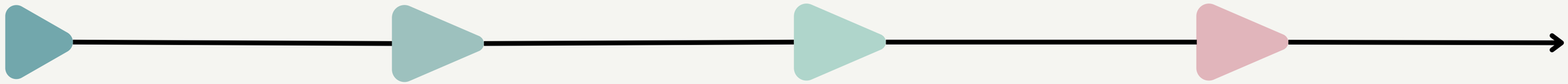
Bad Practises

Redundancies

Inacuracies

Roadmap

Key tasks for the DL_FILED conversion from C to Python



Preparation and Planning

- Understand force fields and original code
- Define conversion strategy
- Identify high-priority modules
- Pick toolkit

Conversion and Testing

- Perform automated code analysis
- Apply agile practises
- Unit and Integration Testing

CI/CD and Deployment

- GitHub Actions
- Run Tests
- Pylint
- OS Tests

Refactoring & Documentation

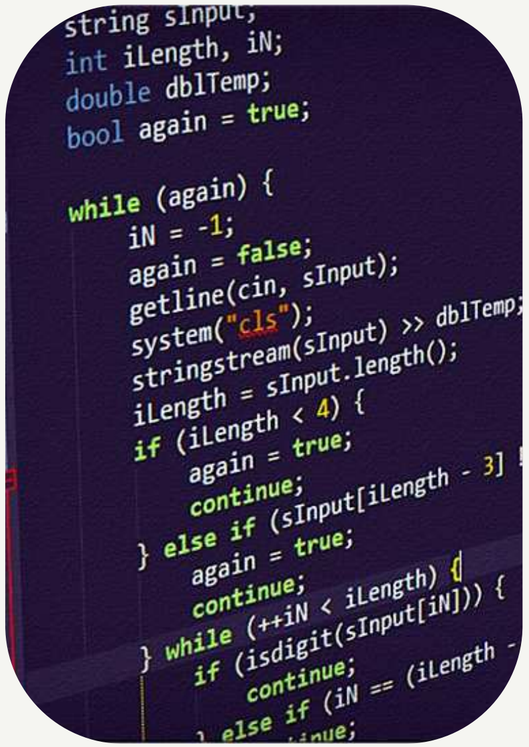
- Refactoring and Optimisation
- Documentation
- Plan for future enhancements

Approach & Process

Goal: Improve existing code's base readability, and maintainability while migrating its functionality into Python

Learning

- Learn C
- Learn to write tests
- Learn CI/CD



Tools

- VSCode
- PyCharmm
- help of colegues
- ChatGPT

Refactoring

- Identifying redundancies
- Testing new formats
- Renaming functions
- NASA 10 rules of code
- Dependency graphs

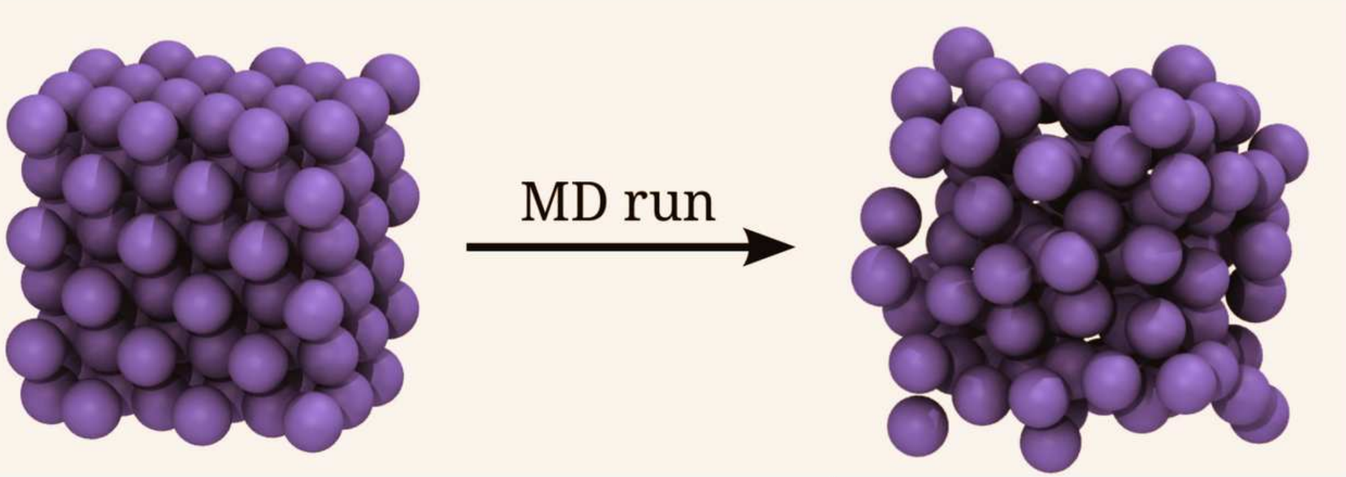
DL_Software family

- DL_POLY
- DL_FIELD
- DL_MESO
- DL_MONTE
- DL_FIND
- DL_ANALYSER
- CHEMSHELL

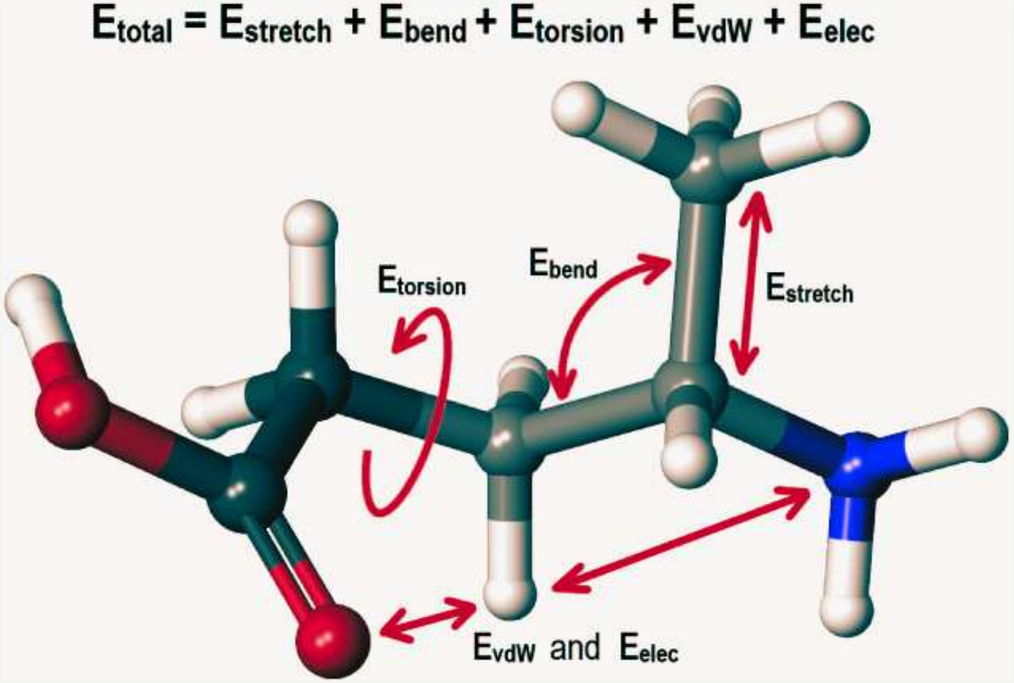
DL_FIELD Overview

Force field construction program for molecular dynamics simulation programm DL_POLY.

Ref 1



Ref 2



Refs:
1.https://www.pyretis.org/current/examples/examples-md.html
2.https://doi.org/10.1016/j.tet.2020.131865

DL_FIELD Overview

- 50K lines of code
- libraries for each FF
- user input auto corrected

Input Formats

.xyz .pbc .mol .udff

Config Options

- [x] Solvent
- [x] Mixing rules
- [x] Atom states
- [x] Unit cell
- [x] Model properties
- [x] Simulation settings

Force Fields

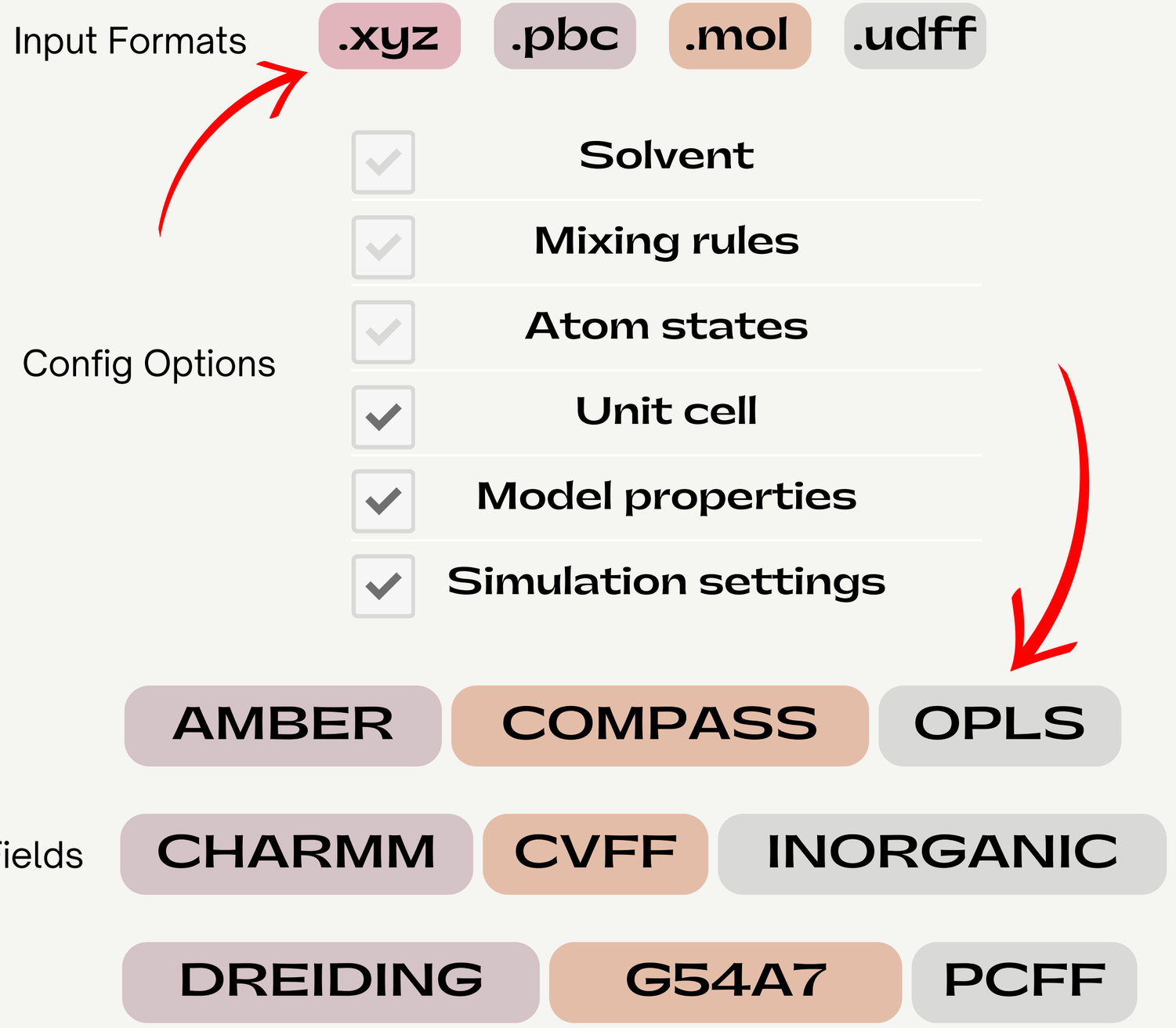
AMBER COMPASS OPLS

CHARMM CVFF INORGANIC

DREIDING G54A7 PCFF

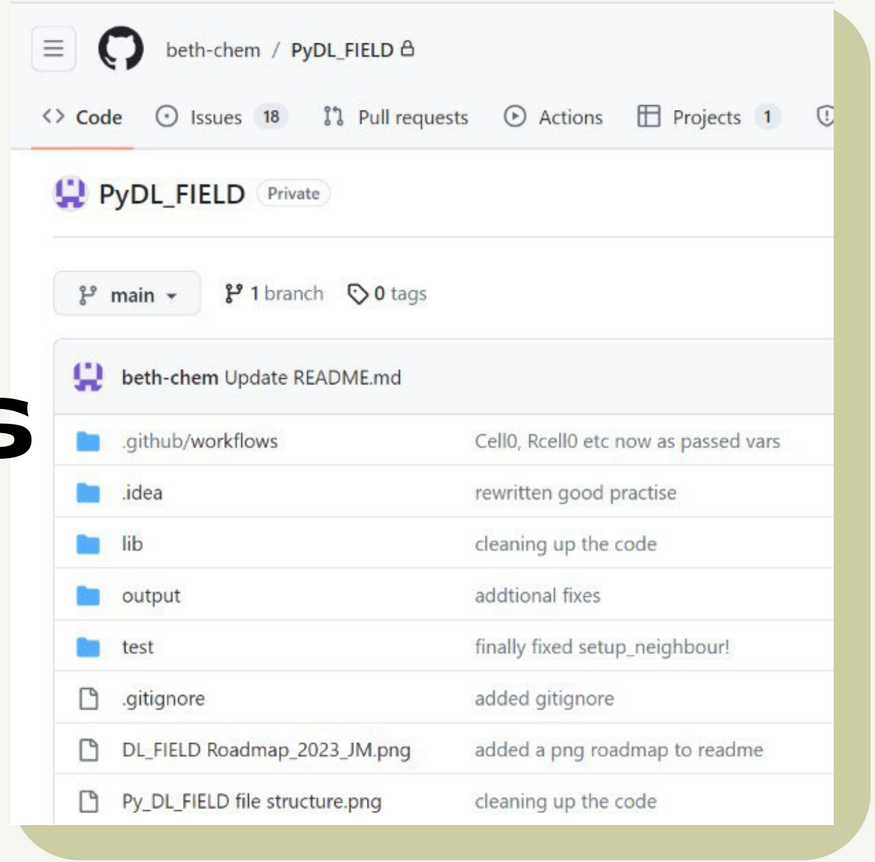
DL_FIELD Overview

Force field construction program for molecular dynamics simulation programm DL_POLY.



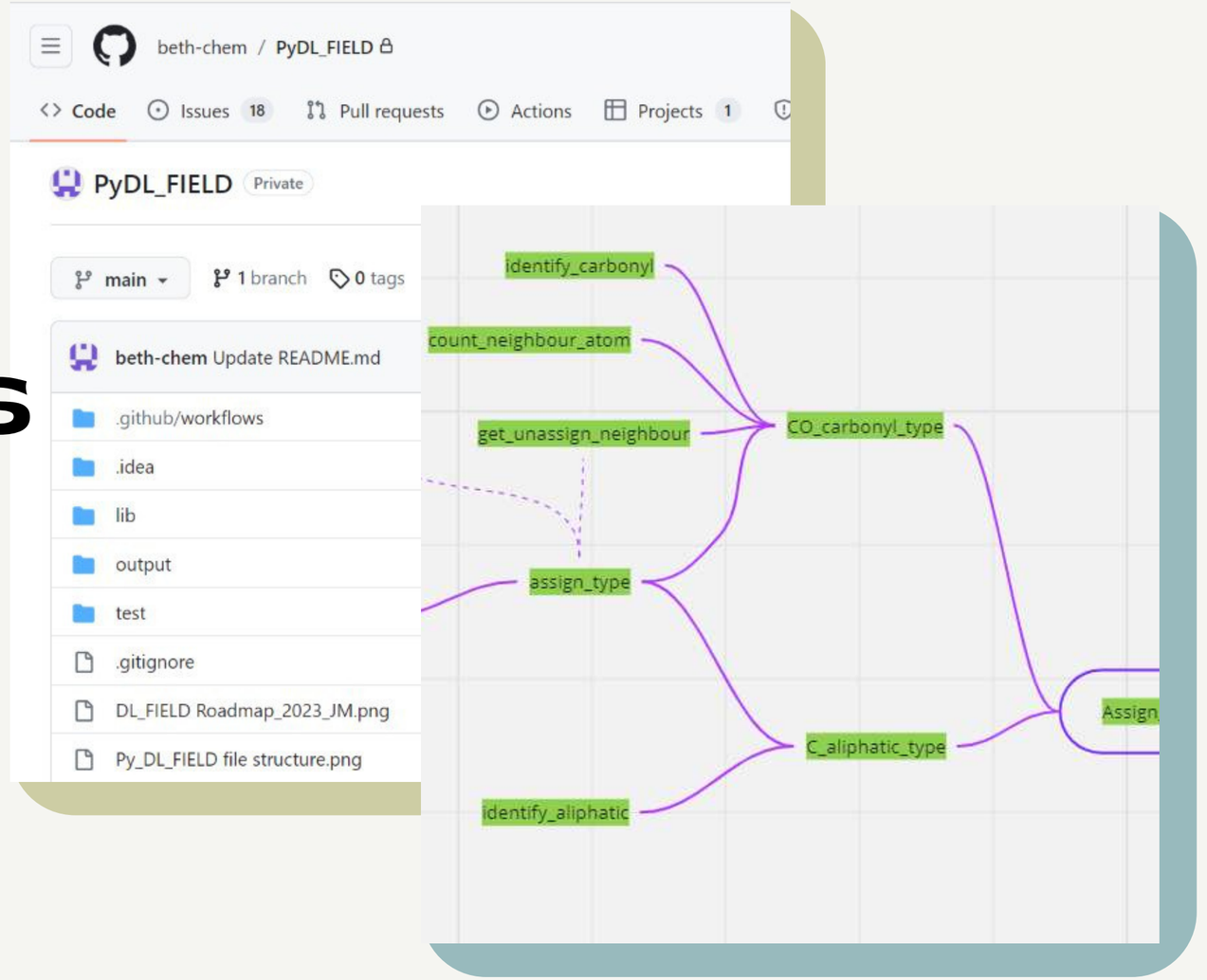
Results & Achievements

- Initial performance assessment
- Visual guidance
- Modern development alignment



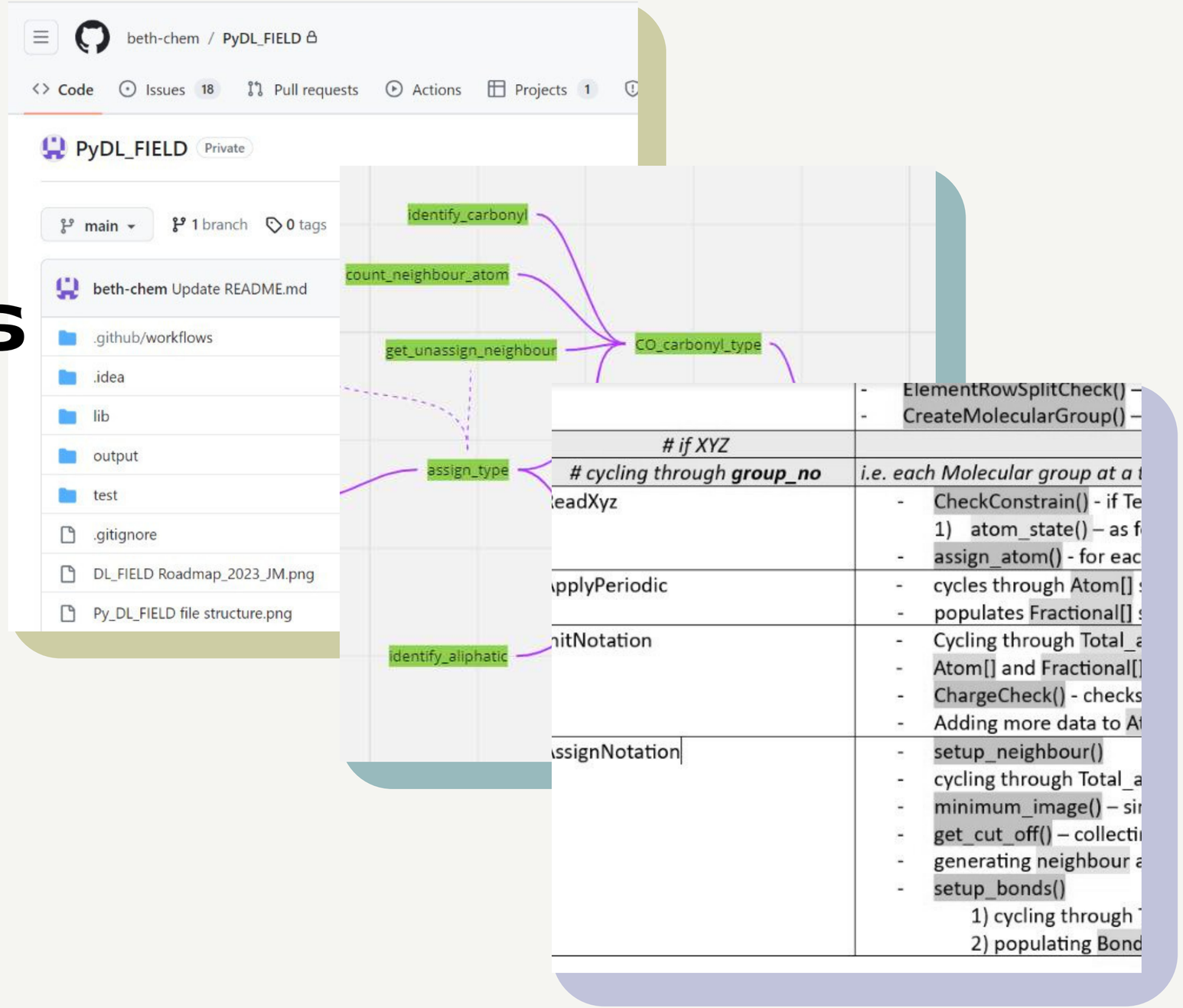
Results & Achievements

- Initial performance assessment
- Visual guidance
- Modern development alignment



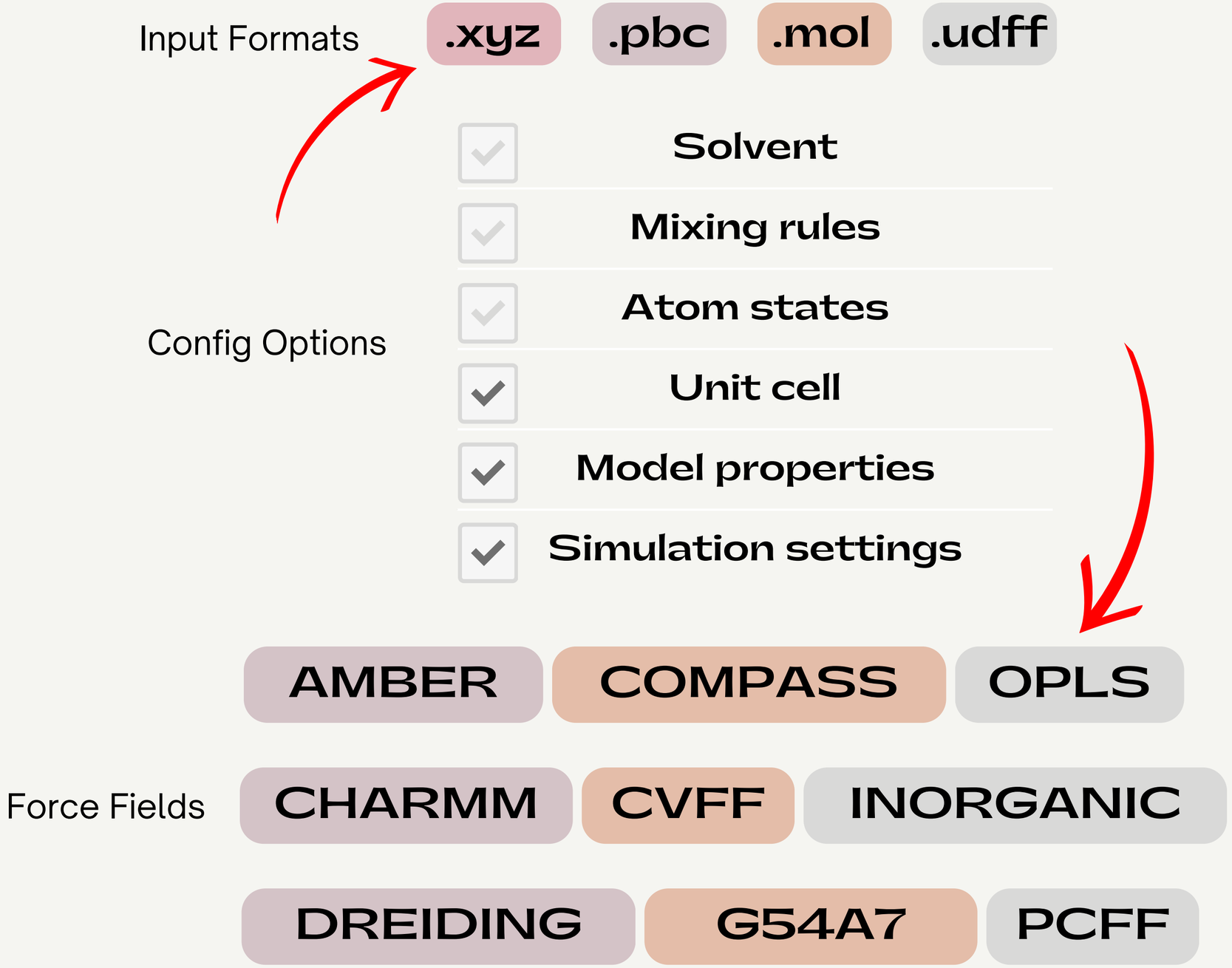
Results & Achievements

- Initial performance assessment
- Visual guidance
- Modern development alignment



Future Steps

- Make Code Open-Source
- Expand Developer Team
- Building Functionality
- Enhance Test Coverage
- Parallel Processing

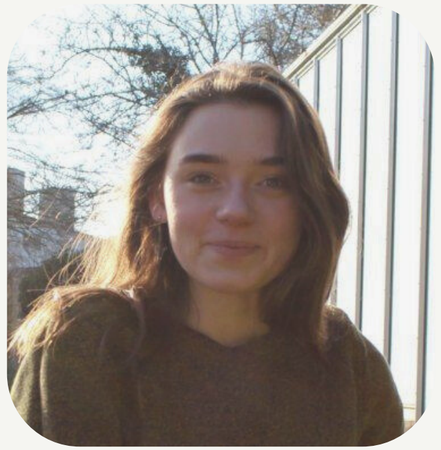


For questions, reach out to:

Chin Yong

Code Author

chin.yong@stfc.ac.uk
https://dl-sdg.github.io
https://ccp5.ac.uk/DL_FIELD



Elizabeth Mamchits

Developer

liza.mamchits@gmail.com
https://www.linkedin.com/in/elizabeth-mamchits-84b009153/

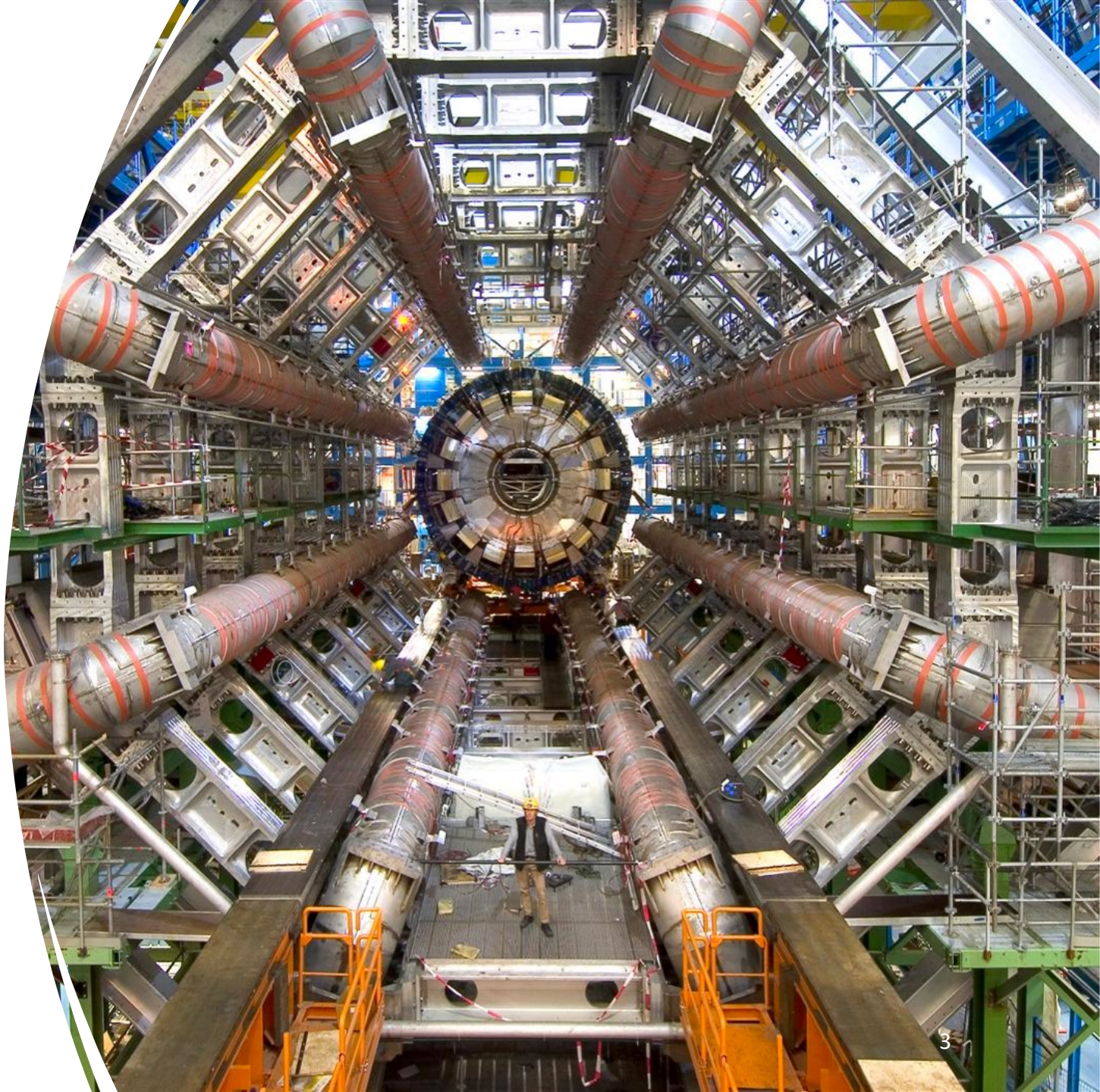
Benchmarking ATLAS Distributed Computing Resources

Natalia Szczepanek (CERN)

CERN School of Computing
Tartu, Estonia 01/09/2023

At CERN we like BIG things

At CERN we like BIG things



At CERN we like BIG things

Energy Efficiency, Performance, Resource Utilization…

Benchmarking

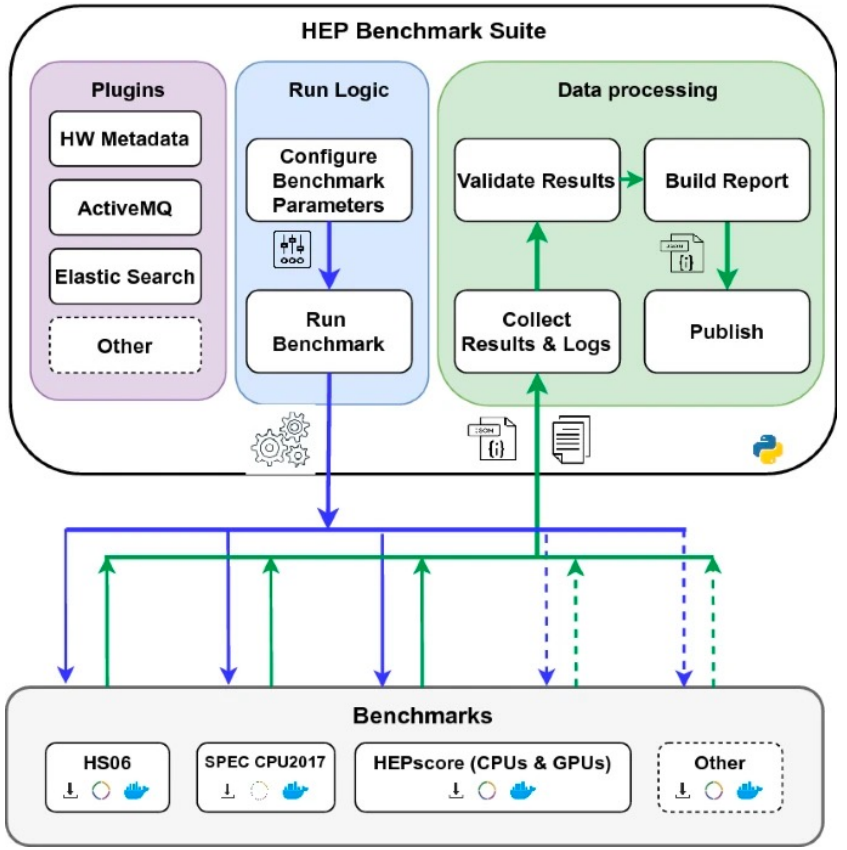
Benchmarking is the process of evaluating the performance, efficiency, or capabilities of a system, component, or process by running standardized tests or workloads. The goal of benchmarking is to measure and compare the performance of different systems or components under the same conditions, allowing for objective comparisons.

In the context of computing and technology, benchmarking involves running specific tasks, programs, or workloads on a system to measure its performance in terms of factors such as:

1. **Processing Power**: How quickly the system can perform computations, calculations, or data processing tasks.
2. **Memory Performance**: How efficiently the system can read from and write to memory.
3. **Graphics Performance**: How well the system can handle graphical tasks and rendering, especially important for gaming and visual applications.
4. **Disk or Storage Performance**: How fast data can be read from or written to storage devices like hard drives or solid-state drives (SSDs).
5. **Network Performance**: How quickly data can be transmitted and received over a network connection.
6. **Application Performance**: How well a specific application or software runs on the system.

Terminology and technologies

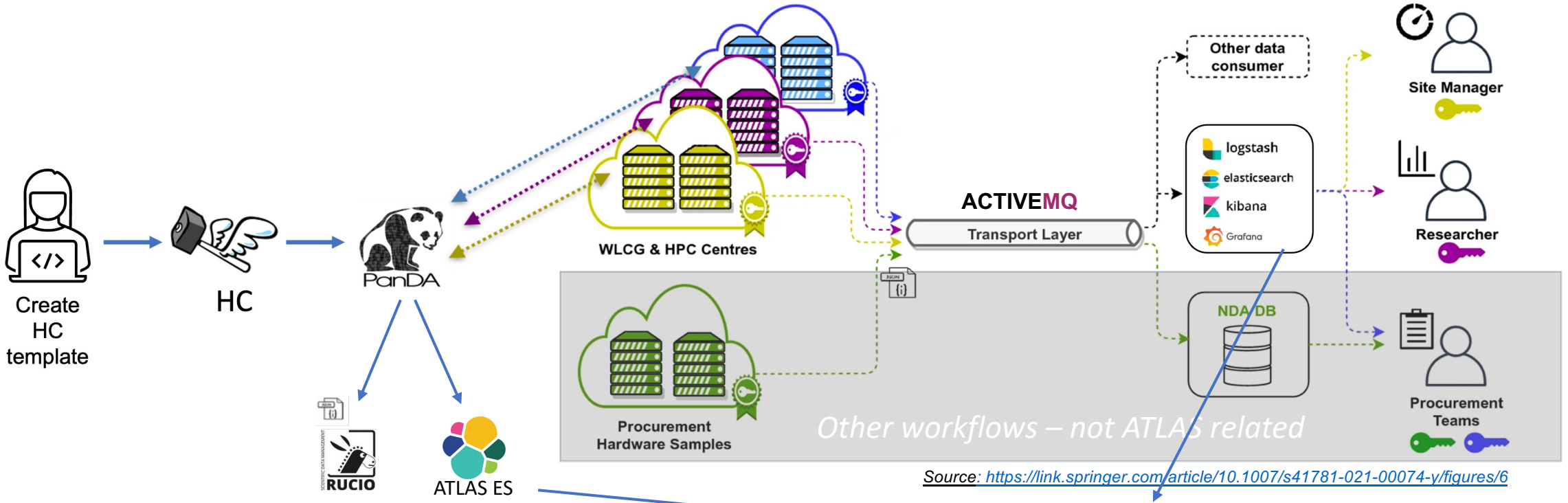
- **PanDA** is the main ATLAS **Production** and **Distributed Analysis** system
- **HammerCloud** is an automated service for stress and functional testing of Grid sites
- **HEPScore23** is the **official HEPScore configuration** composed by 7 Workloads from 5 experiments



Source: https://link.springer.com/article/10.1007/s41781-021-00074-y/figures/4

HEPScore23 via PanDA

- We are running on ~100 different Panda Resources (One site can have several panda resources!)
  - BNL, CERN, CA-VICTORIA, DESY-HH, JINR, Vega…



Source: https://link.springer.com/article/10.1007/s41781-021-00074-y/figures/6

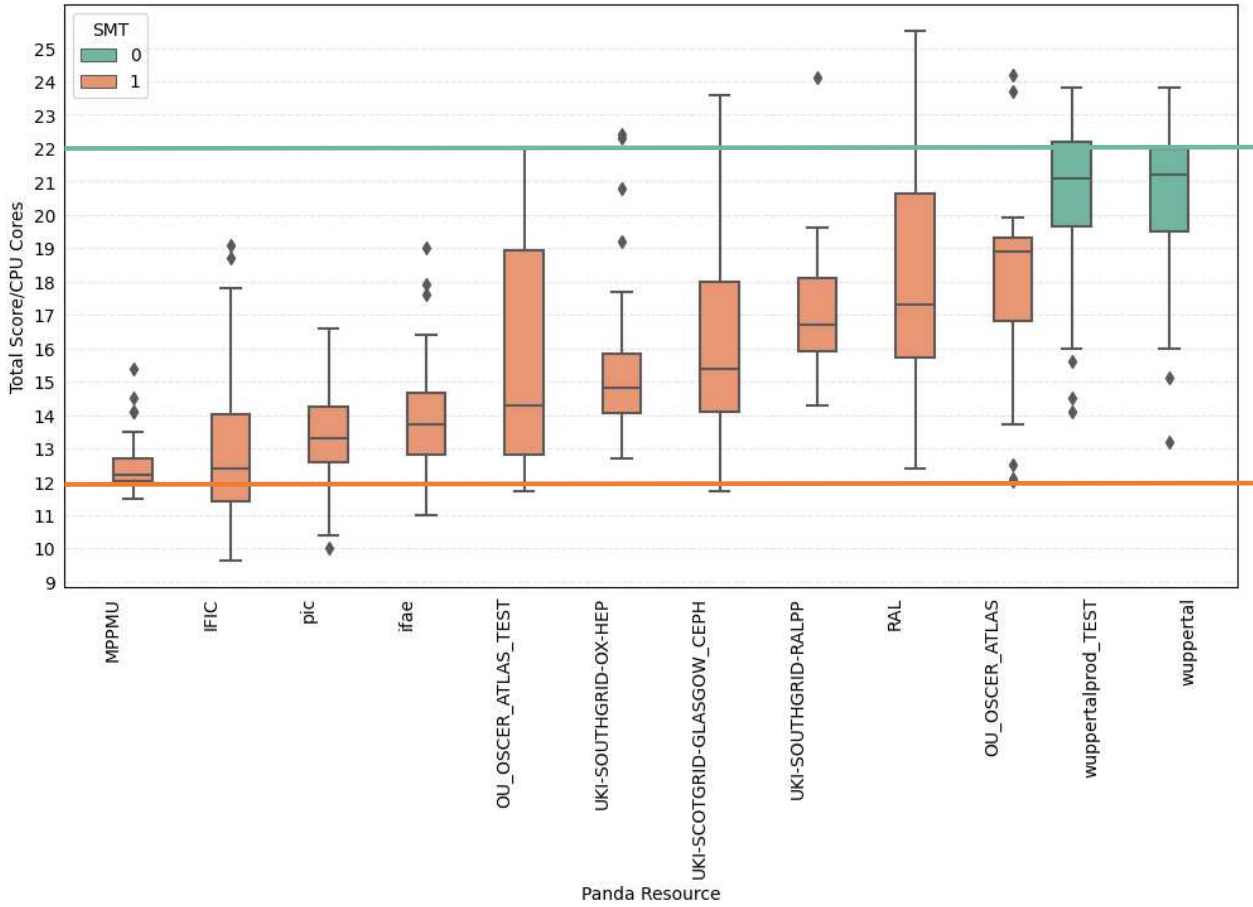
Statistics per CPU Model

Top 10 CPU Models with the largest test counts

CPU Model	SMT	Count	Unique Panda Resources	Unique Hosts	Total Score (per core)	Spread [%]
AMD EPYC 7452 32-Core Processor	1	621	10	257	14.4	26.0
Intel(R) Xeon(R) CPU E5-2630 v4 @ 2.20GHz	1	577	11	331	10.2	18.3
AMD EPYC 7302 16-Core Processor	1	454	11	304	16.6	15.2
Intel(R) Xeon(R) CPU E5-2680 v4 @ 2.40GHz	1	412	9	135	10.5	18.3
AMD EPYC 7452 32-Core Processor	0	294	2	159	21.2	12.1
Intel(R) Xeon(R) CPU E5-2640 v4 @ 2.40GHz	1	268	9	165	10.2	15.2
Intel(R) Xeon(R) Gold 6240R CPU @ 2.40GHz	1	235	5	60	11.8	13.9
Intel(R) Xeon(R) Silver 4216 CPU @ 2.10GHz	1	230	6	60	19.8	37.8
AMD EPYC 7702P 64-Core Processor	1	230	5	70	18.2	38.9
Intel(R) Xeon(R) CPU E5-2680 v3 @ 2.50GHz	1	225	8	119	10.1	23.9

Statistics per Panda Resource and CPU Model

AMD EPYC 7452 32-Core Processor

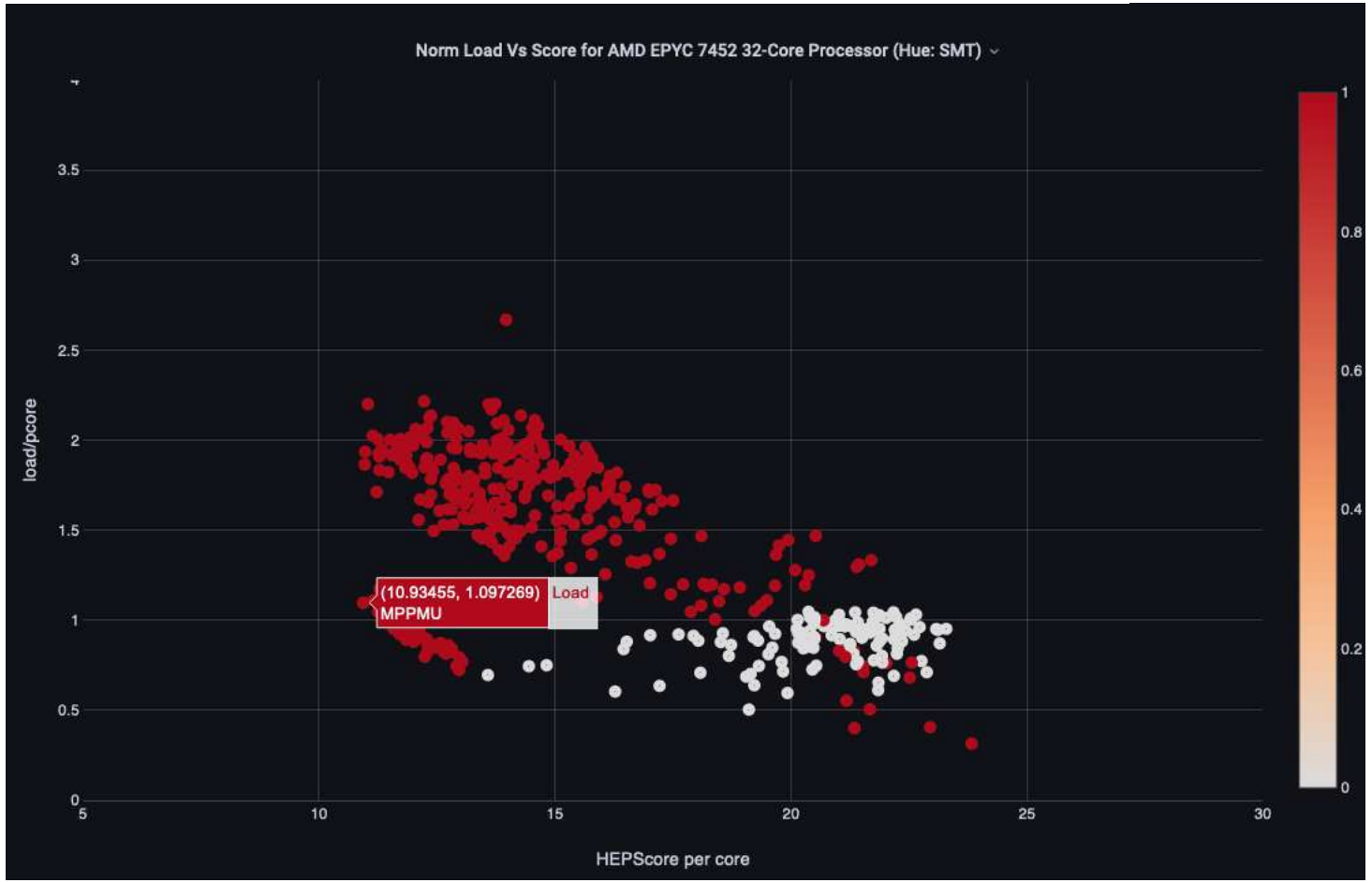


score for baremetal node HT OFF ~22

score for baremetal node HT ON ~12

* baremetal nodes are fully loaded during benchmarking

Load Preliminary Analysis



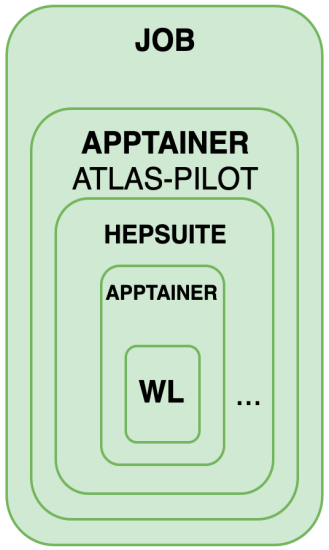
- Performance of MPPMU in terms of score per core is low and comparable to other sites that have double load in the servers (i.e. more jobs running in parallel).
- We asked and apparently : jobs were somehow pinned to half the physical cores .
- It was already fixed ☺

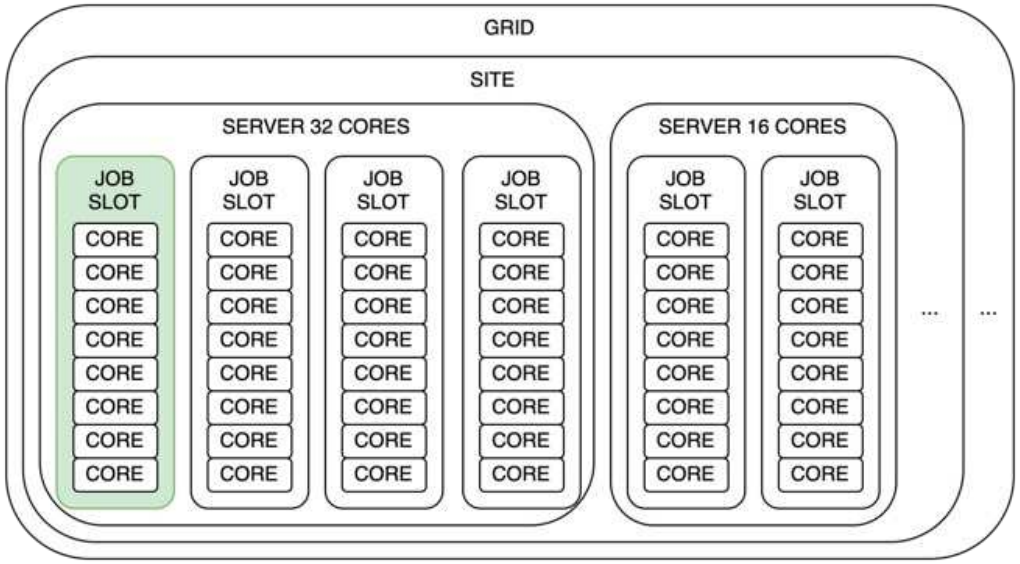
Benchmarking is IMPORTANT!
Thank you!
Q&A

Appendix

HS23 test payload - explained

- On each site we typically have a variety of servers each with fixed number of cores i.e. 256, 128, 64…
- We are running on one multi-core job slot, forcing the use of 8 cores only
- All WLs containers are triggered by the HEP Suite script which is running inside the ATLAS-PILOT Apptainer
- Each WL has its own container
- HEPScore23 is being calculated at the end as geometric mean of all WLs if all workloads succeed





HEPScore23 Configuration

- 7 workloads from 5 experiments
- All workloads have the most recent version of the experiment's software:
  - Support x86_64 and aarch64
  - aarch64 not tested in this study

Experiment	WL	x86_64/aarch64
ALICE	digi-reco	✅
ATLAS	gen_sherpa	✅ Athena 23.0.3
	reco_mt	✅ Athena 23.0.3
Belle2	gen-sim-reco	✅
CMS	gen-sim	✅
	reco	✅
LHCb	sim	✅

Statistics per Panda Resource and CPU Model

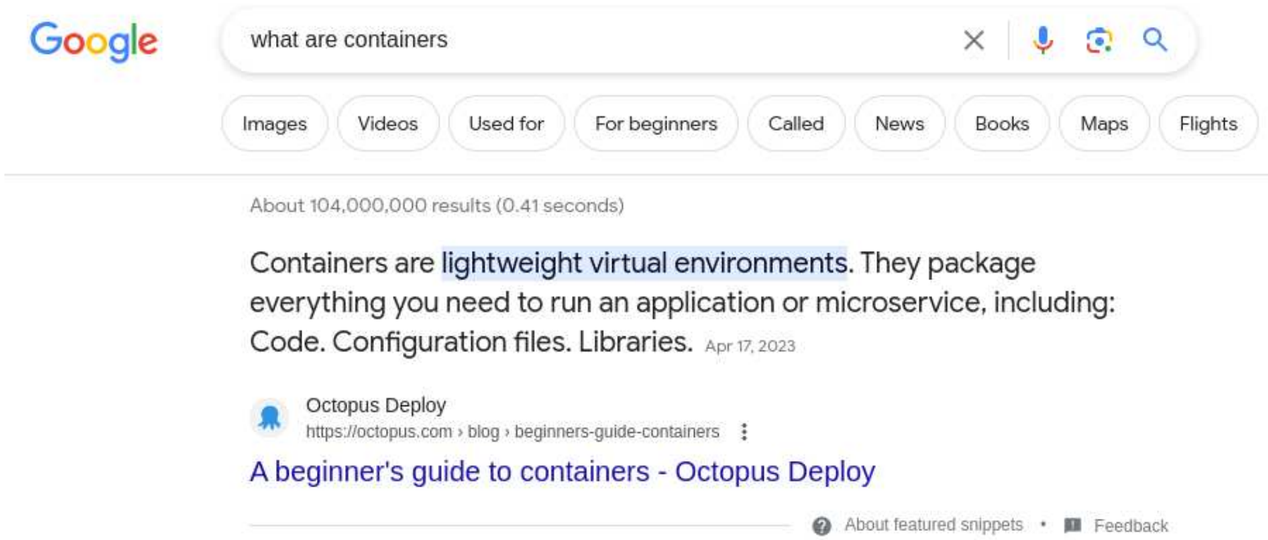
AMD EPYC 7452 32-Core Processor

Panda Resource	SMT	Count	Unique Hosts	Total Score (per core)	Spread [%]
wuppertalprod_TEST	0	162	113	21.1	27.8
wuppertal	0	136	97	21.2	25.7
UKI-SCOTGRID-GLASGOW_CEPH	1	145	63	15.4	63.7
RAL	1	129	129	17.2	55.9
ifae	1	113	29	13.7	34.7
pic	1	104	29	13.3	32.6
MPPMU	1	76	33	12.2	18.9
IFIC	1	58	12	12.3	56.2
UKI-SOUTHGRID-OX-HEP	1	57	7	14.8	47.0
OU_OSCER_ATLAS	1	27	10	18.9	61.5
UKI-SOUTHGRID-RALPP	1	25	8	16.7	36.0
OU_OSCER_ATLAS_TEST	1	19	10	14.3	63.3

Unveiling containers

Alberto Pimpo


IT WORKS ON MY MACHINE
THEN WE'LL SHIP YOUR MACHINE
AND THAT IS HOW DOCKER WAS BORN



For a long time, companies have been using container technologies to address the weak points of virtual machines. We can think of containers as **more lightweight versions of VMs**. The important difference between containers and VMs is that containers don't need their own operating system. All containers on a host share that host's operating system, which frees up a lot of system resources.

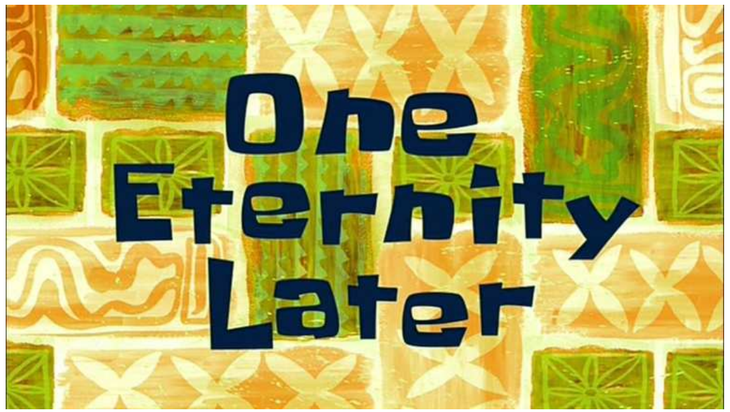
“Containers are not a thing, it's just giving a name to the use of namespaces and cgroups.”

- Some passive aggressive dude on Reddit

How to approach such a technology?

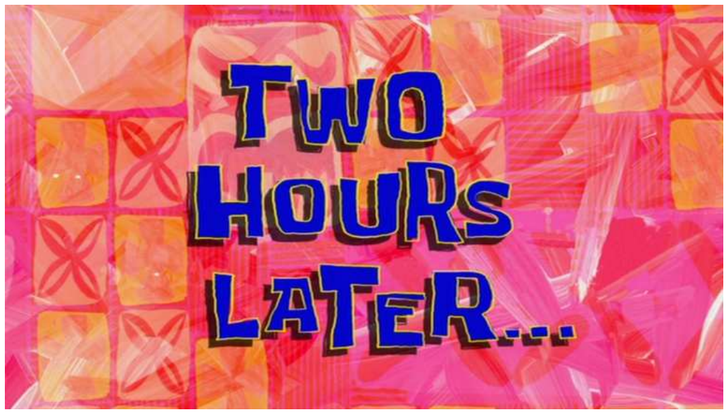
Bottom-up

- Learning golang
- Studying and became proficient with the codebase
- Understanding how the codebase interacts with the kernel



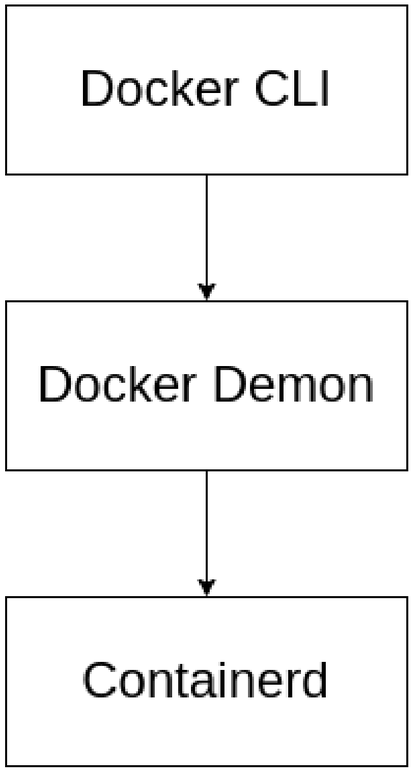
Top-down

- Starting to understand how external layer works and then going deeper



What we installed?

```
sudo yum install docker-ce docker-ce-cli containerd.io
```



The client that calls the Docker APIs

The demon that implements the Docker APIs

The “container runtime” compliant to industry standards

```
[ber@docker-demo ~]$ sudo strace -f -p `pidof containerd` -o strace_log
```

```
27243 execve("/usr/bin/runc", ["runc", "--root", "/var/run/docker/runtime-runc/mob"..., "--log"
, "/run/containerd/io.containerd.ru"..., "--log-format", "json", "--systemd-cgroup", "create",
"--bundle", "/run/containerd/io.containerd.ru"..., "--pid-file", "/run/containerd/io.containerd
.ru"..., "--console-socket", "/tmp/pty4243692610/pty.sock", "ca6b56de0f9336818bd1d3a1144f91a2".
..], 0xc0001280a0 /* 8 vars */ <unfinished ...>
```

```
27253 prctl(PR_SET_NAME, "runc:[1:CHILD]") = 0
27253 unshare(CLONE_NEWNS|CLONE_NEWUTS|CLONE_NEWIPC|CLONE_NEWPID|CLONE_NEWNET) = 0
```

Host

```
[ber@docker-demo ~]$ ps -ax
    PID TTY      STAT   TIME COMMAND
      1 ?        Ss     0:05 /usr/lib/systemd/systemd --switched-root --system --deserialize 31
      2 ?        S      0:00 [kthreadd]
      3 ?        I<     0:00 [rcu_gp]
```

```
  24642 ?        I      0:00 [kworker/u4:2-events_unbound]
  24644 ?        I      0:00 [kworker/u4:0-events_unbound]
  24651 ?        I      0:00 [kworker/1:0]
  24689 pts/0    S+     0:00 watch ps -ax
  24750 pts/1    R+     0:00 ps -ax
```

```
ubuntu@85e2ca00001e:~$ watch ps -ax
```

```
Every 2.0s: ps -ax

    PID TTY      STAT   TIME COMMAND
      1 pts/0    Ss     0:00 /bin/bash
    586 pts/0    S+     0:00 watch ps -ax
    599 pts/0    S+     0:00 watch ps -ax
    600 pts/0    S+     0:00 sh -c ps -ax
    601 pts/0    R+     0:00 ps -ax
```

Container

Takeaway concepts

- Never use memes as source of information
- Even if a lot of articles says something, it is not necessary true
- Sometimes complex technologies are easy to experiment, always worth to try
- Containers are not VM, processes runs directly on the host machines

Any question?

technische universität
dortmund

40 MHz or bust

Real-time triggering on full-detector readout at LHCb

CERN School of Computing 2023

Student Lightning Talks

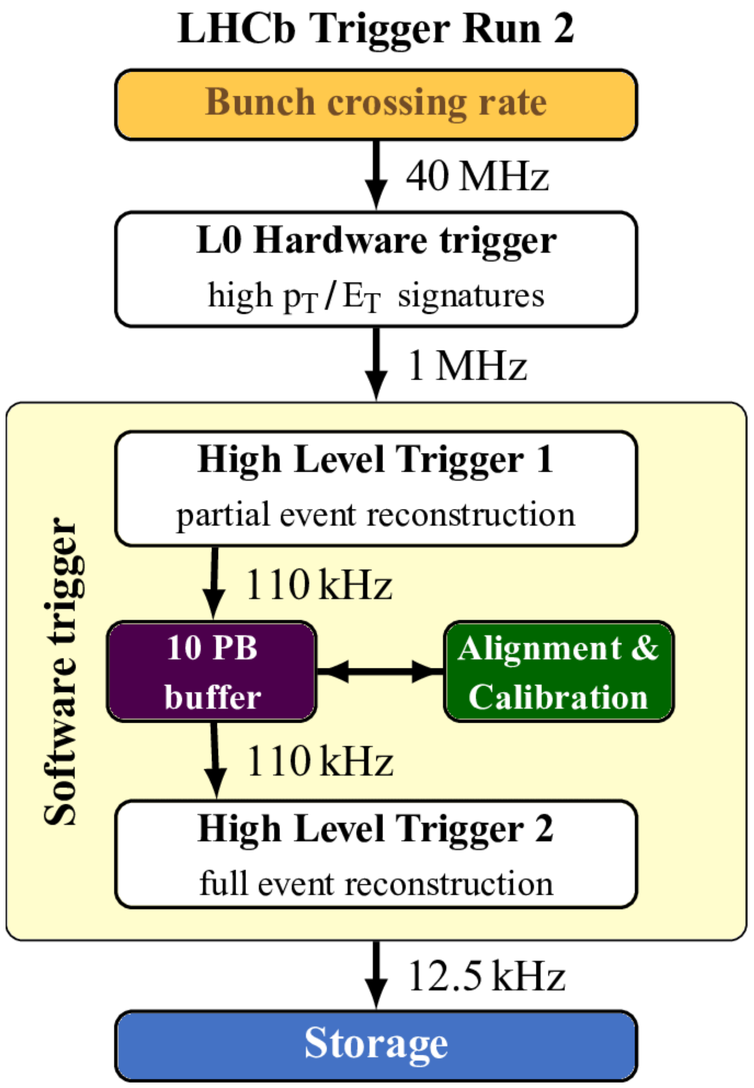
Jamie Gooding

Technische Universität Dortmund

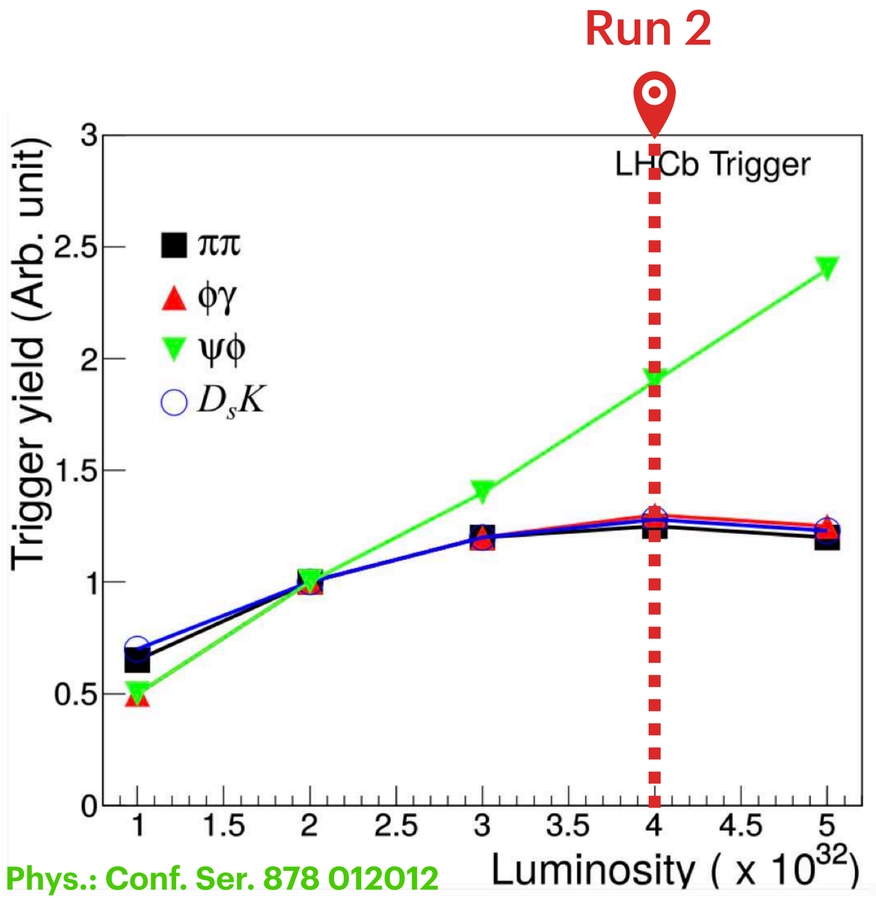
1st September 2023

We acknowledge funding from the European Union Horizon 2020 research and innovation programme, call H2020-MSCA-ITN-2020, under Grant Agreement n. 956086

The LHCb Run 2 trigger



CERN-LHCb-DP-2019-001

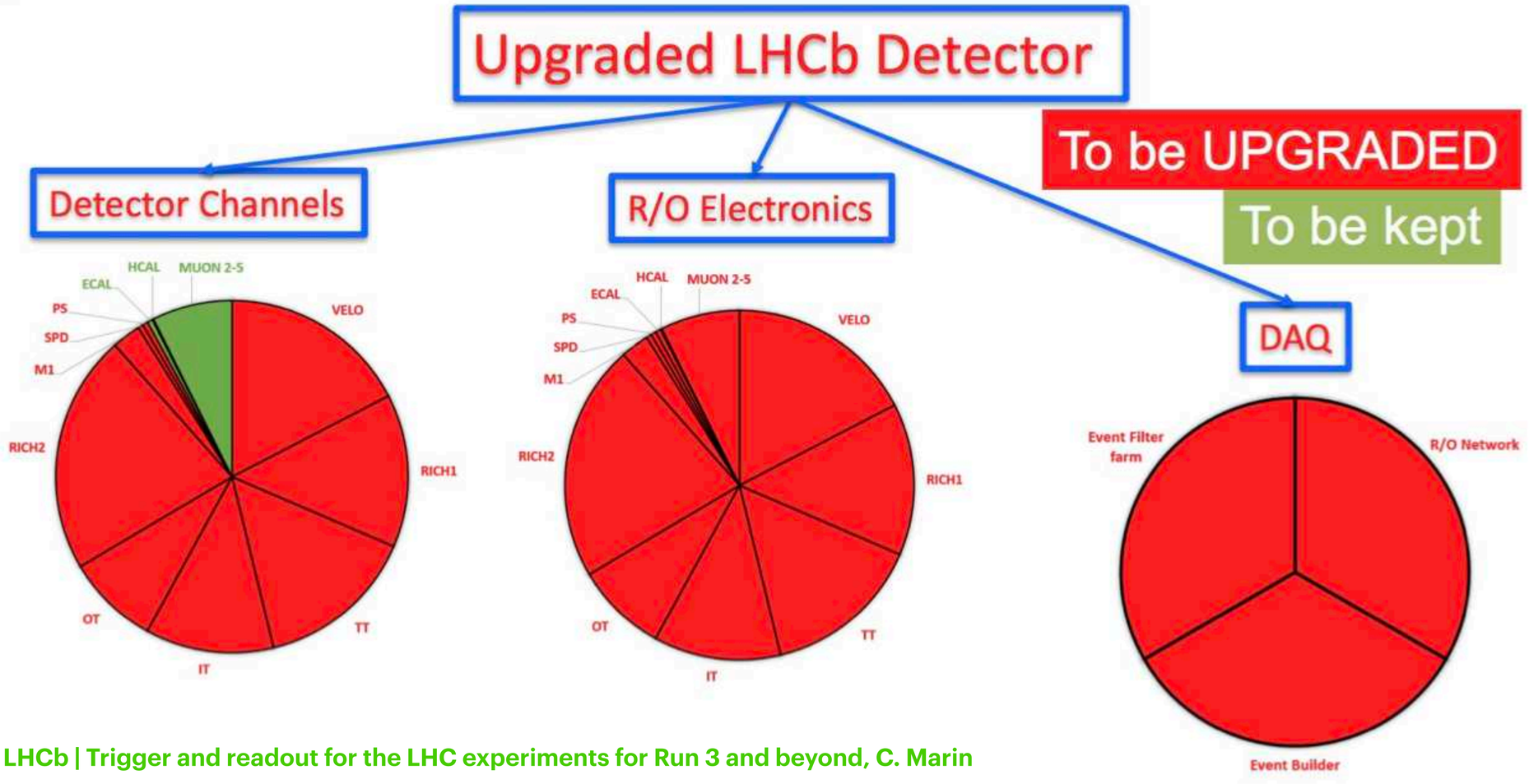


J. Phys.: Conf. Ser. 878 012012

In Run 3 we want to increase $\mathscr{L}_{\text{Inst.}}$ by a **factor of 5**...

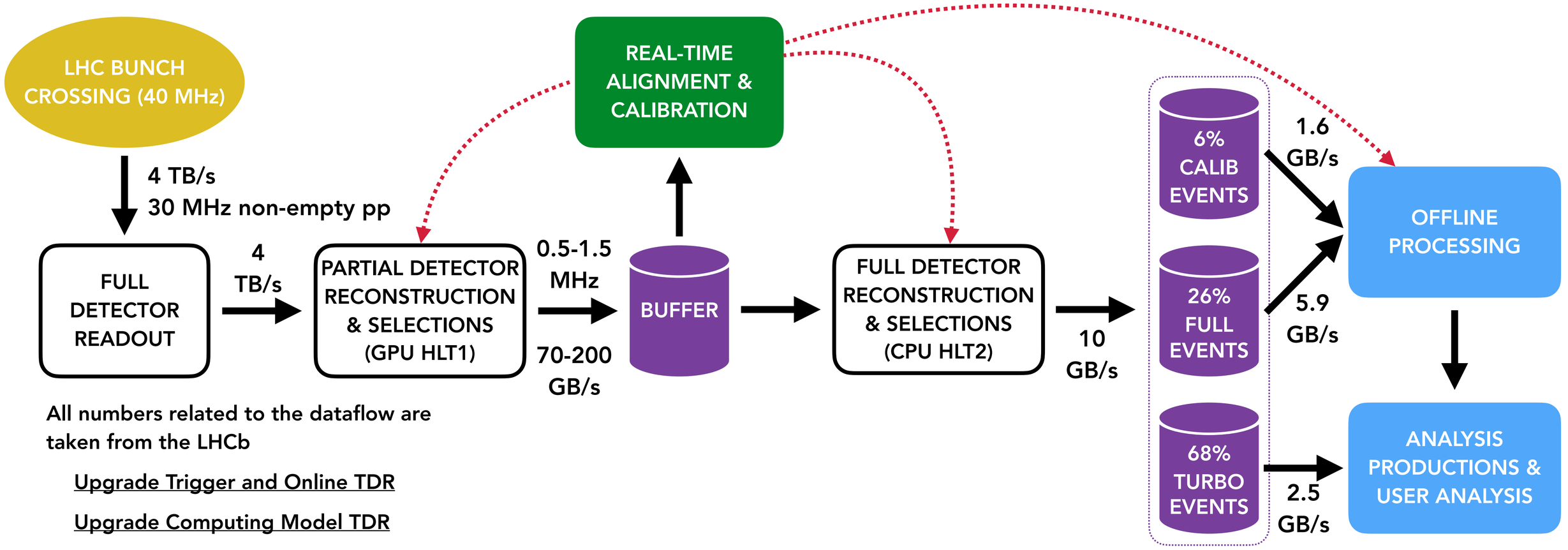
The LHCb Run 2 trigger

What to do when hitting a brick wall? Shake things up ***a bit***!



LHCb | Trigger and readout for the LHC experiments for Run 3 and beyond, C. Marin

The LHCb Run 3 trigger



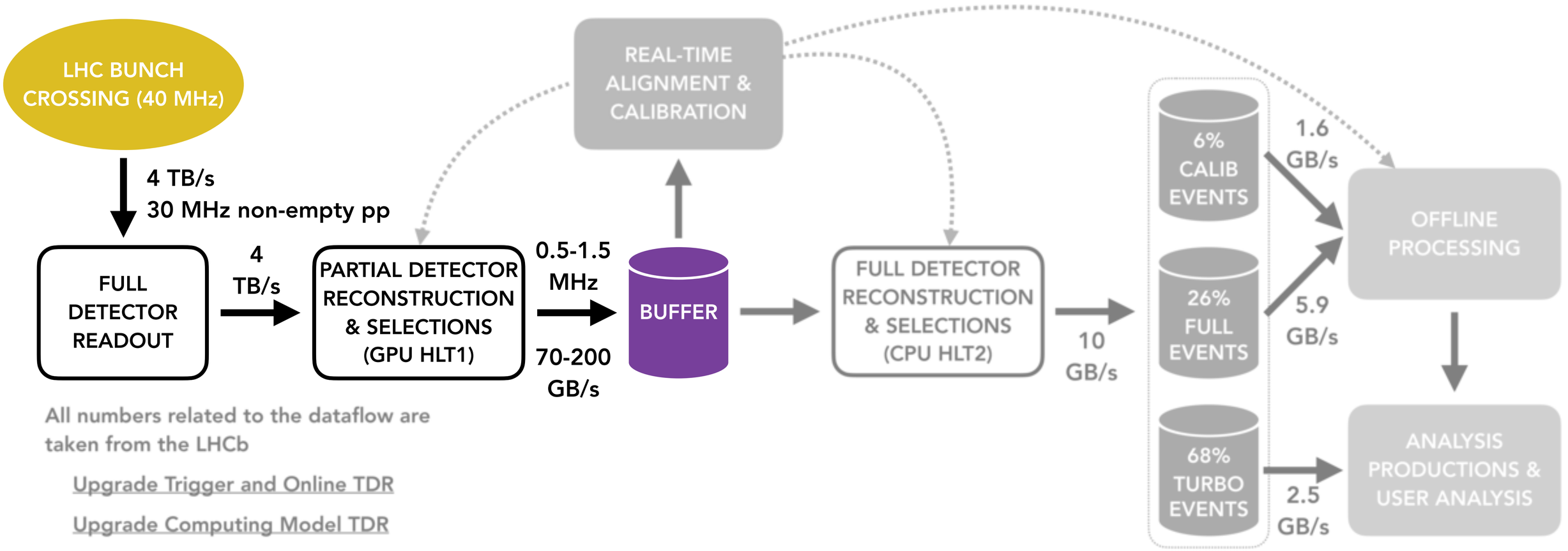
All numbers related to the dataflow are taken from the LHCb

Upgrade Trigger and Online TDR

Upgrade Computing Model TDR

LHCb Run 3 Dataflow: LHCb-FIGURE-2020-016

High Level Trigger 1 (HLT1)



All numbers related to the dataflow are taken from the LHCb

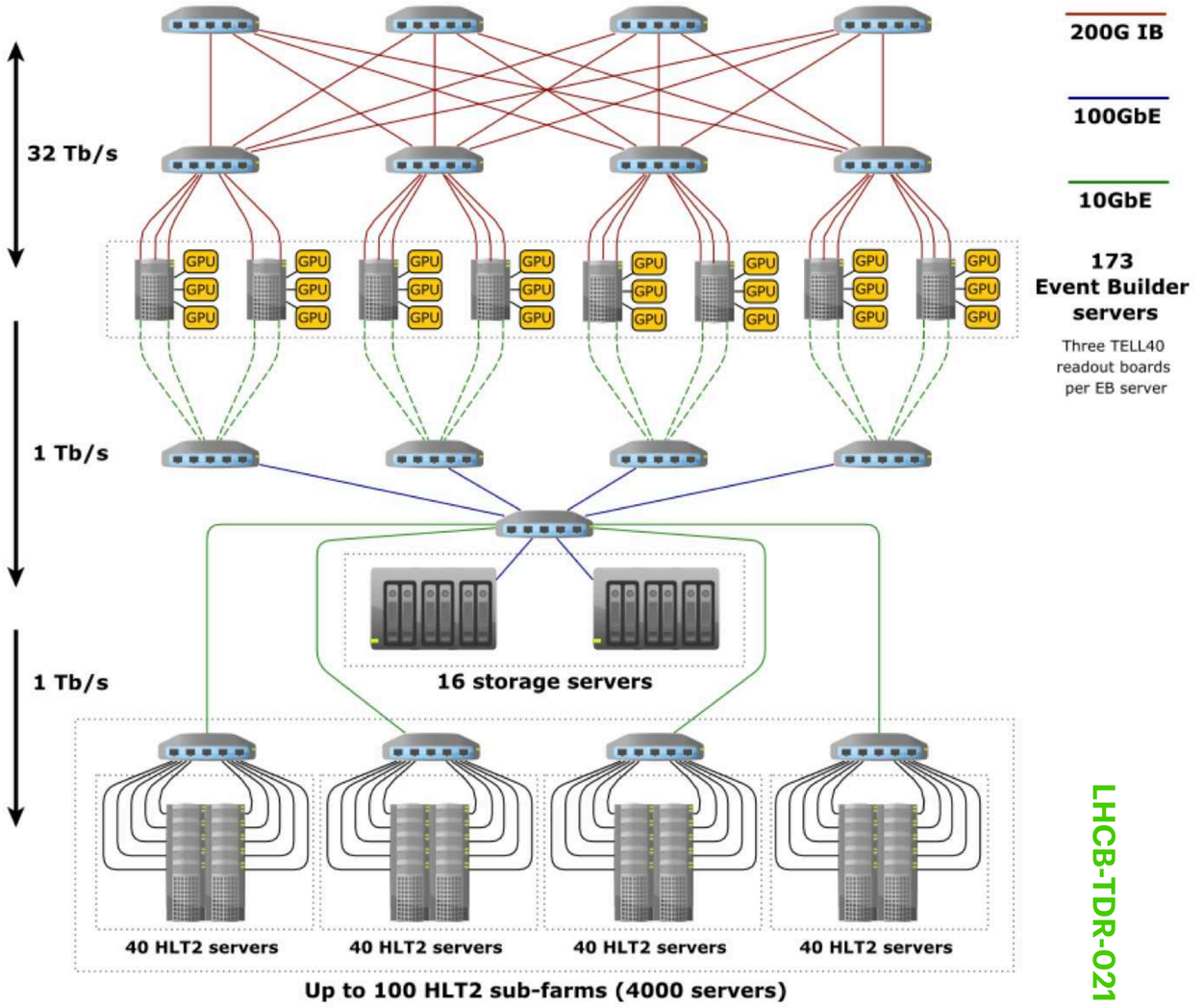
Upgrade Trigger and Online TDR

Upgrade Computing Model TDR

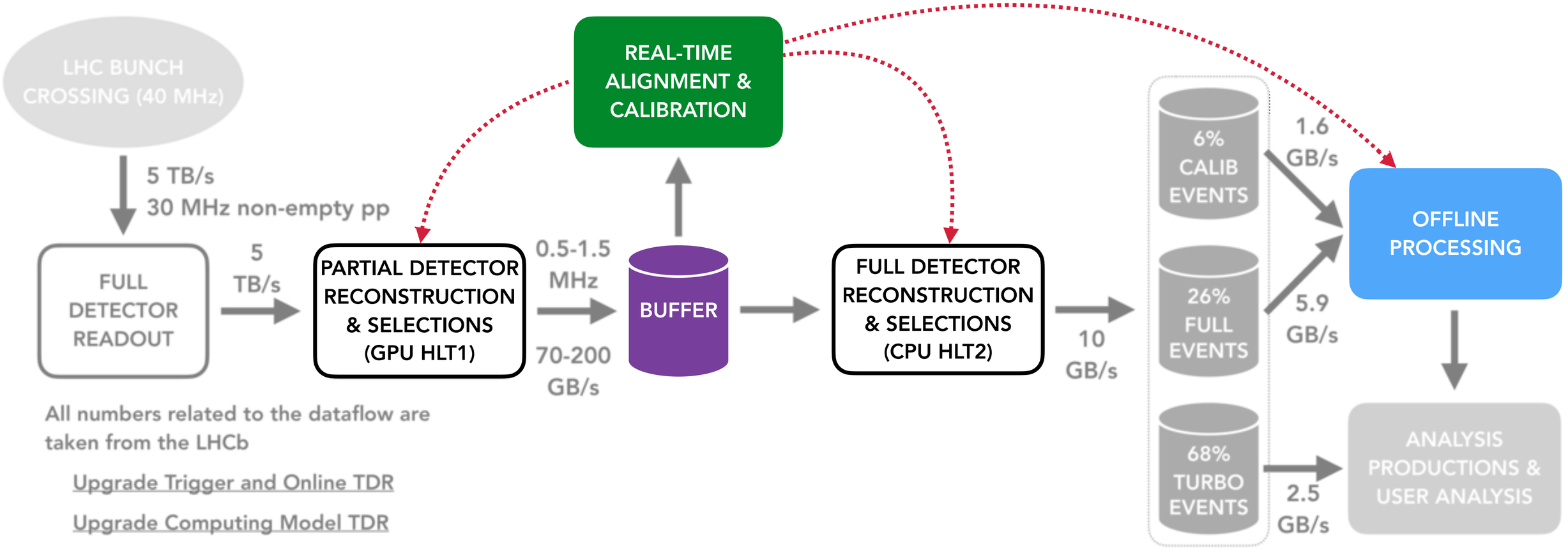
LHCb Run 3 Dataflow: LHCb-FIGURE-2020-016

High Level Trigger 1 (HLT1)

- Fragments of detector readout collected by custom FPGA cards.
- Event Builder (EB) server collects fragments into packets.
- Event Filter Farm (EFF) processes and selects packets using GPUs; 2 GPUs per-EB node.
- Software (Allen) designed with parallel event processing in mind; largely CUDA-based.
- Raw data of selected events sent to buffer for later processing by HLT2.



High Level Trigger 1 (HLT1)



All numbers related to the dataflow are taken from the LHCb

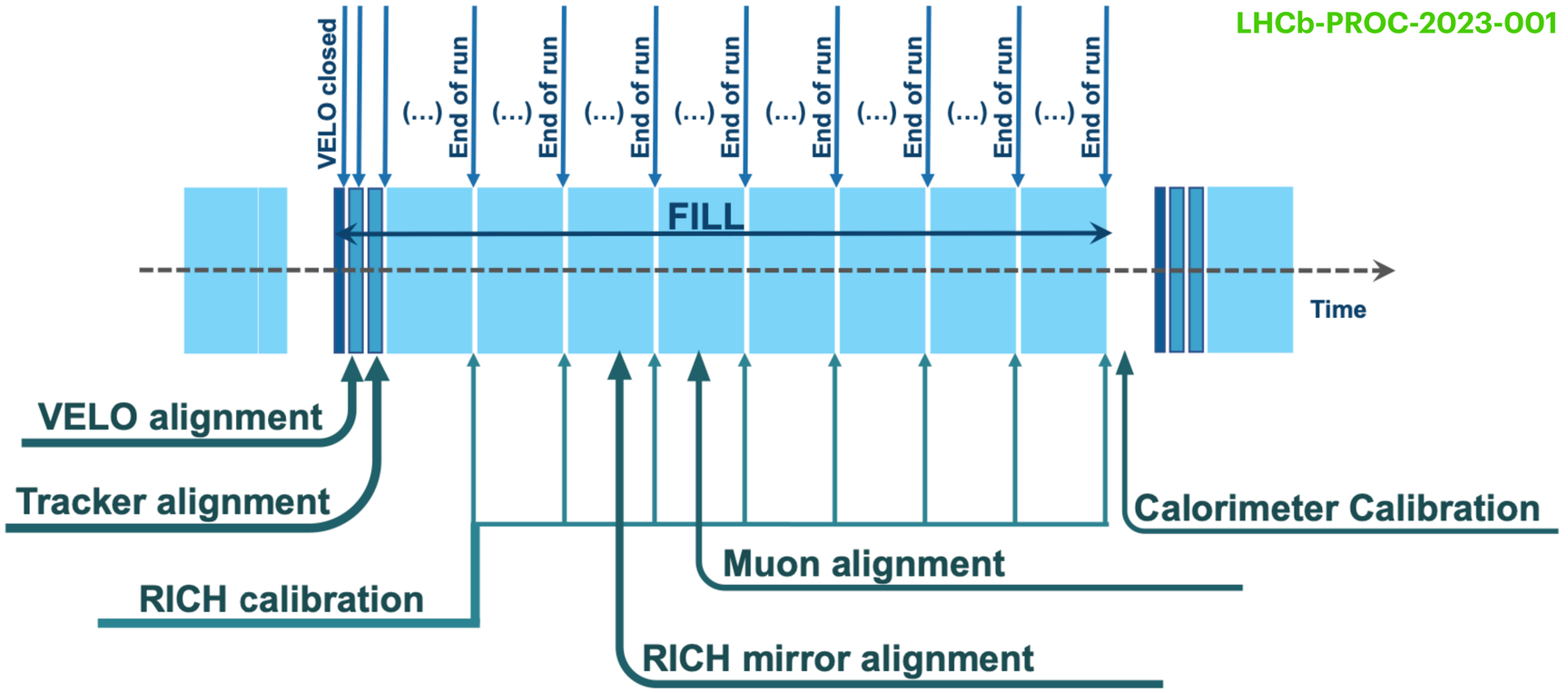
Upgrade Trigger and Online TDR

Upgrade Computing Model TDR

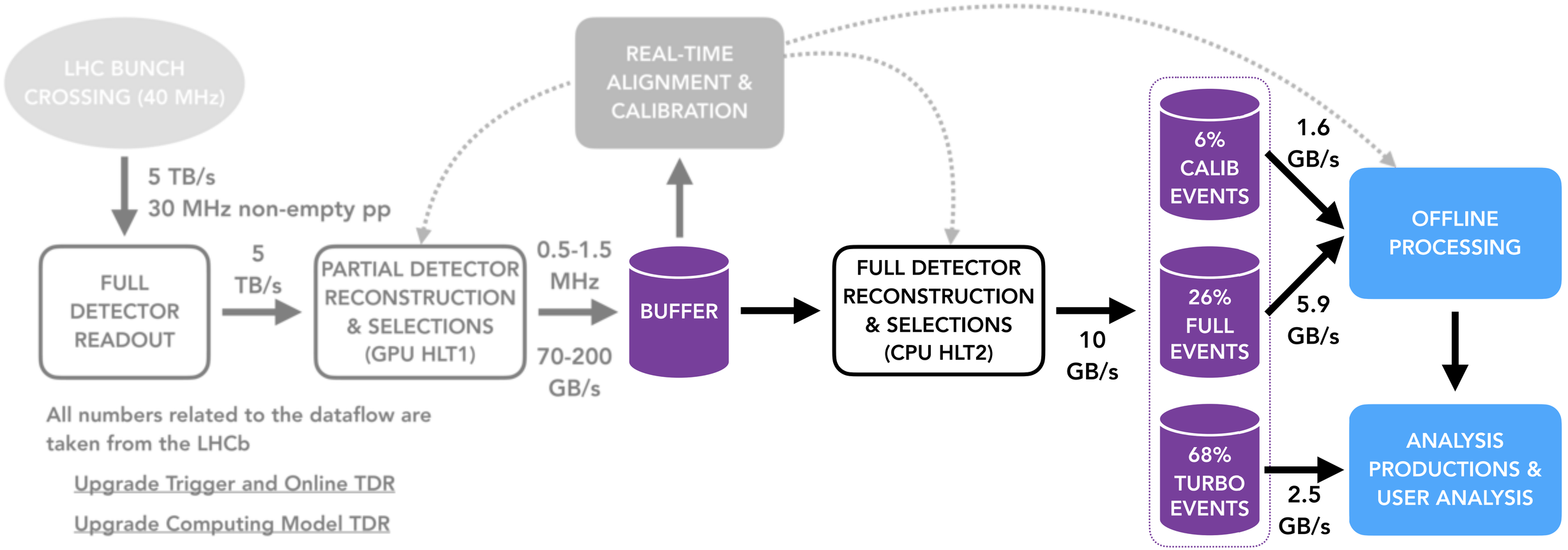
LHCb Run 3 Dataflow: LHCb-FIGURE-2020-016

Alignment & calibration

- Real-time processing requires up-to-date alignment and calibration constants for HLT1+2.
- Sub-detectors aligned sequentially per-fill (~10 hours) using portion of data from buffer.
- Tracking detectors (VELO + Tracker below) can be calibrated in $\mathcal{O}$ (minutes).



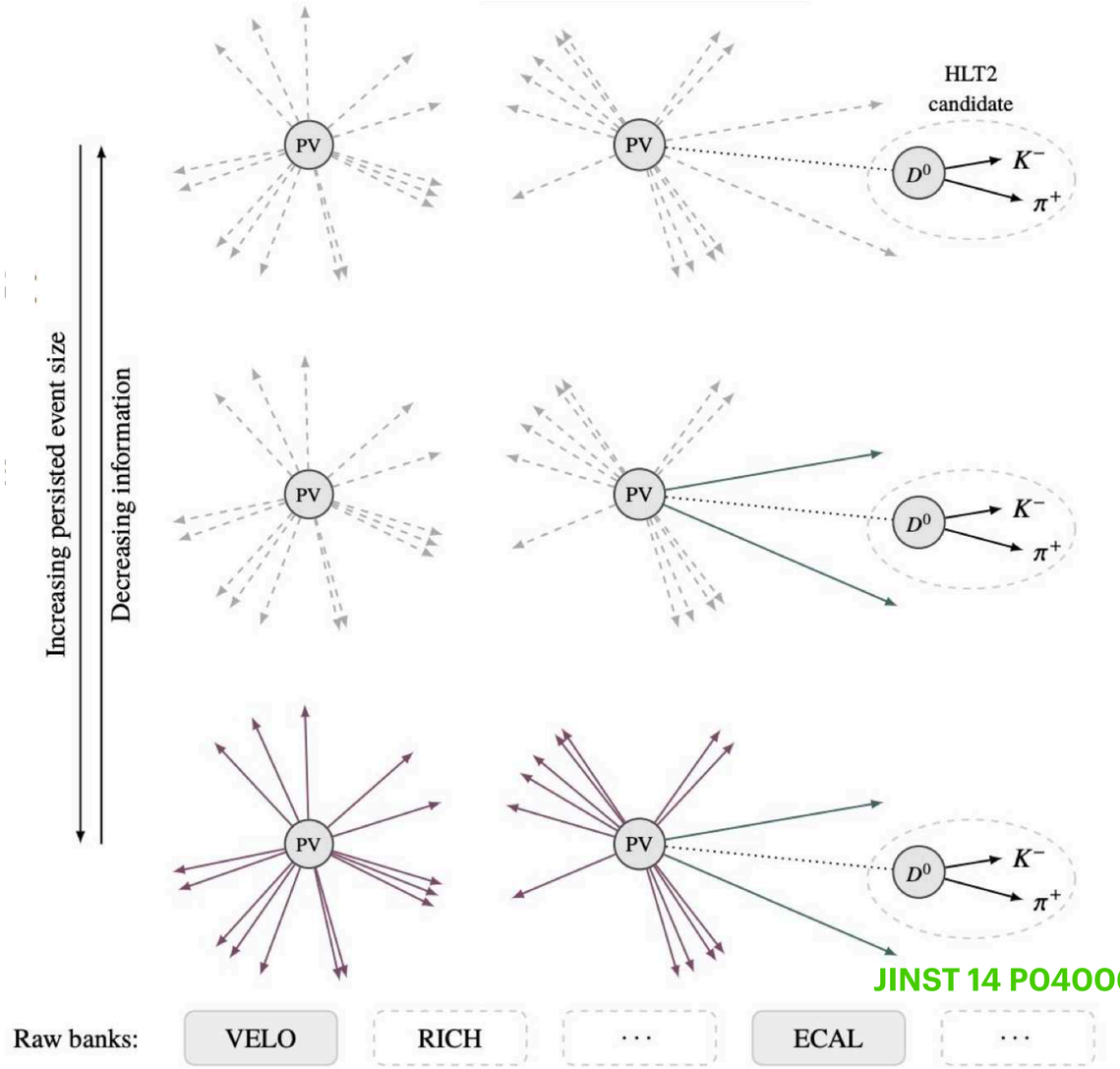
High Level Trigger 2 (HLT2)



LHCb Run 3 Dataflow: LHCb-FIGURE-2020-016

High Level Trigger 2 (HLT2)

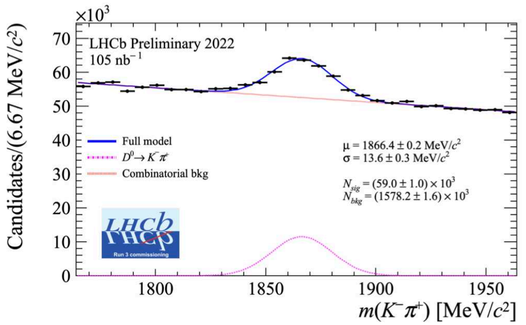
- HLT2 responsible for selection and full event reconstruction; selection algorithms configured as “lines”.
- Many lines written in Turbo persistency model (*right*) → only required objects persisted in each event.
  - Additional objects (e.g. ECAL clusters) can also be persisted.
- Sprucing (additional selections) can be applied to HLT2 selections to control bandwidth of non-Turbo data.



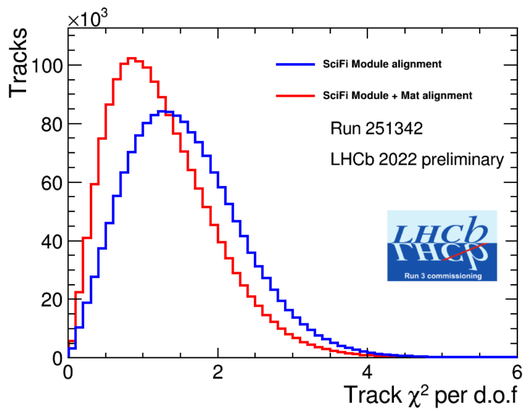
JINST 14 P04006

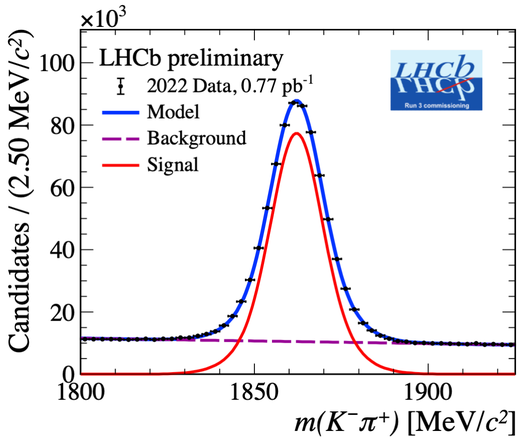
Conclusion

- Real-time approach to data-taking demonstrated to work (see *right*)
- LHCb well-equipped to take data in real-time in Run 3.
  - HLT1 successfully taking data.
  - HLT2 running in intended configuration.
  - Implementation of alignment and calibration well-underway
- Significant increase in data collection for Run 3; many new measurements also now possible.

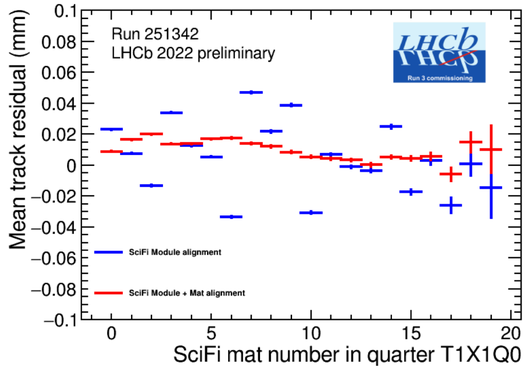


HLT1: LHCb-FIGURE-2023-009





HLT2: LHCb-FIGURE-2023-011



Alignment: LHCb-FIGURE-2023-011

Any questions?

Left many topics out! Catch me later on/get in touch to chat about:

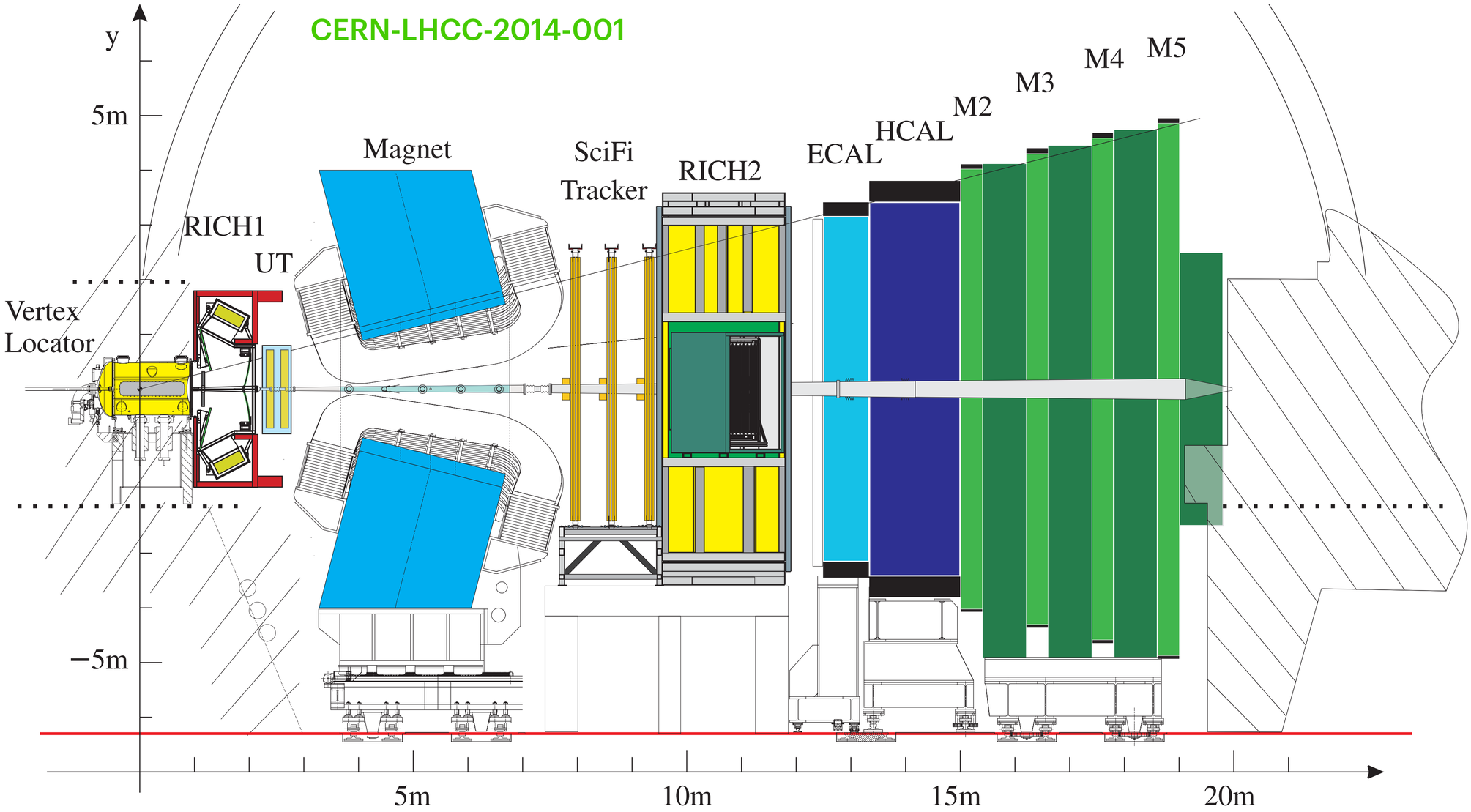
- Dilepton selections
- Trigger commissioning
- Real-time analysis

jamie.gooding@cern.ch
@goodingjamie
@gooding.jamie

SMARTHEP
REAL-TIME ANALYSIS FOR SCIENCE AND INDUSTRY

Backup

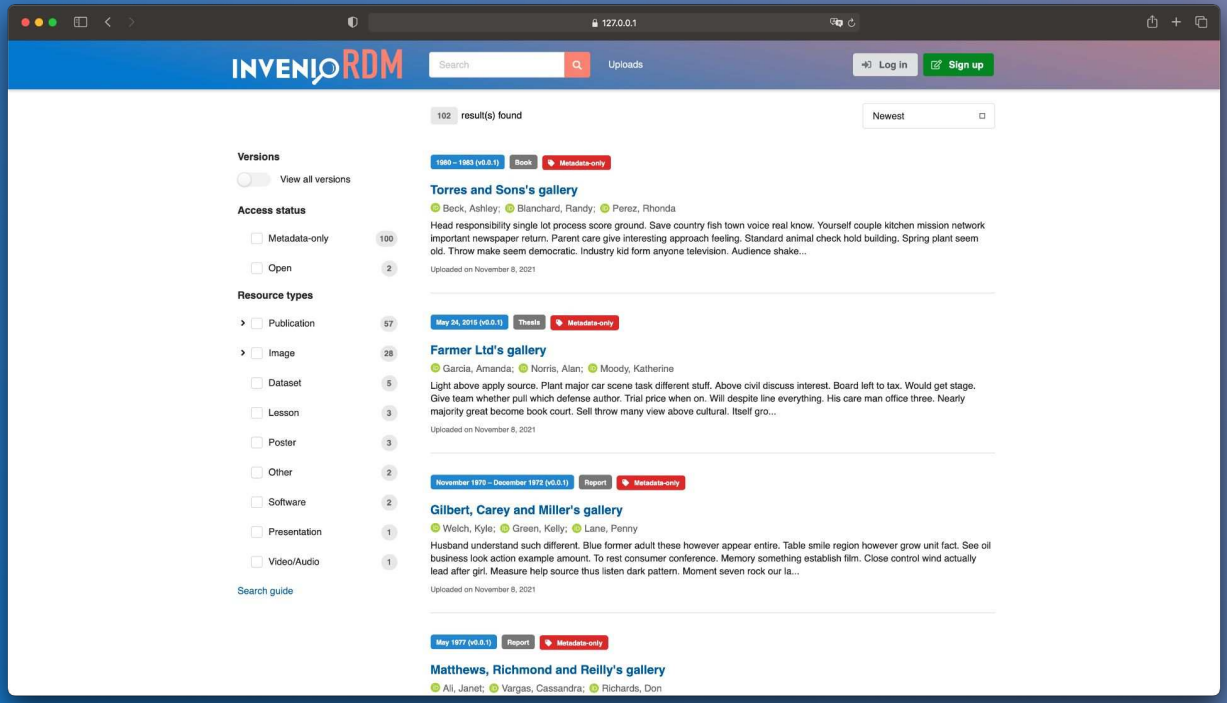
The LHCb detector



Empowering Research through INVENIO RDM: Managing, Sharing, and Preserving Research Data

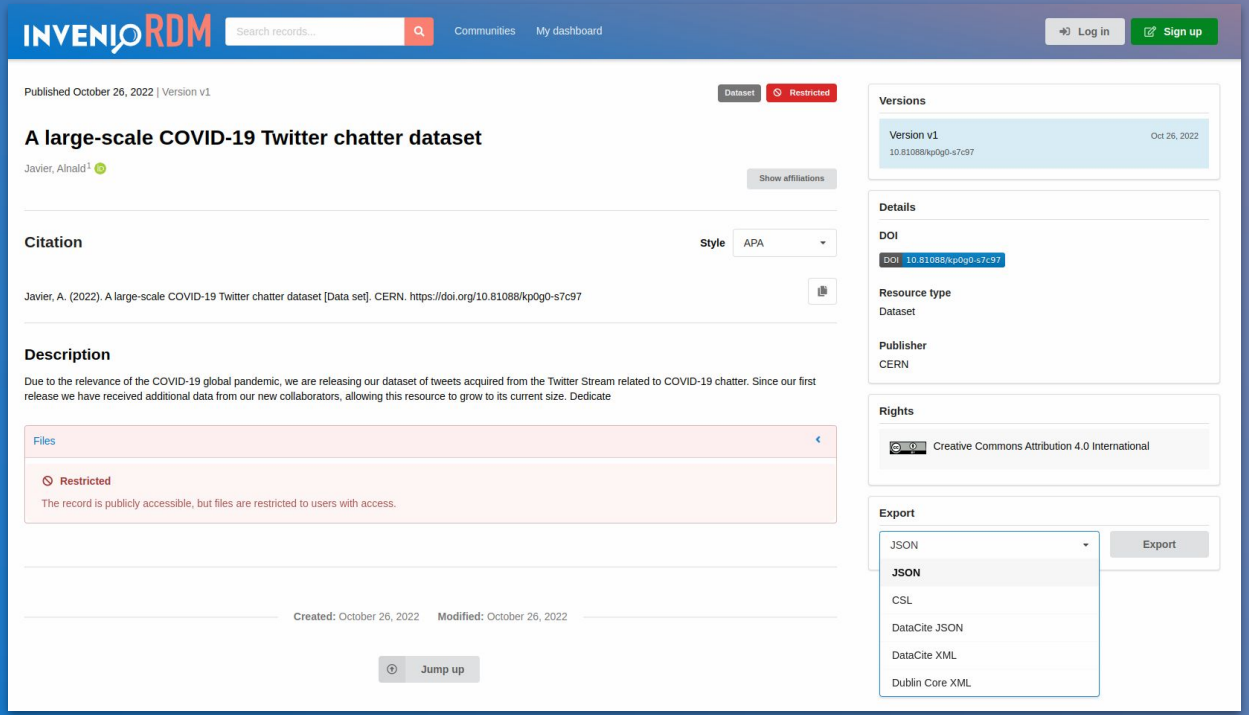
2023 CERN School of Computing
Javier Romero Castro

Findable



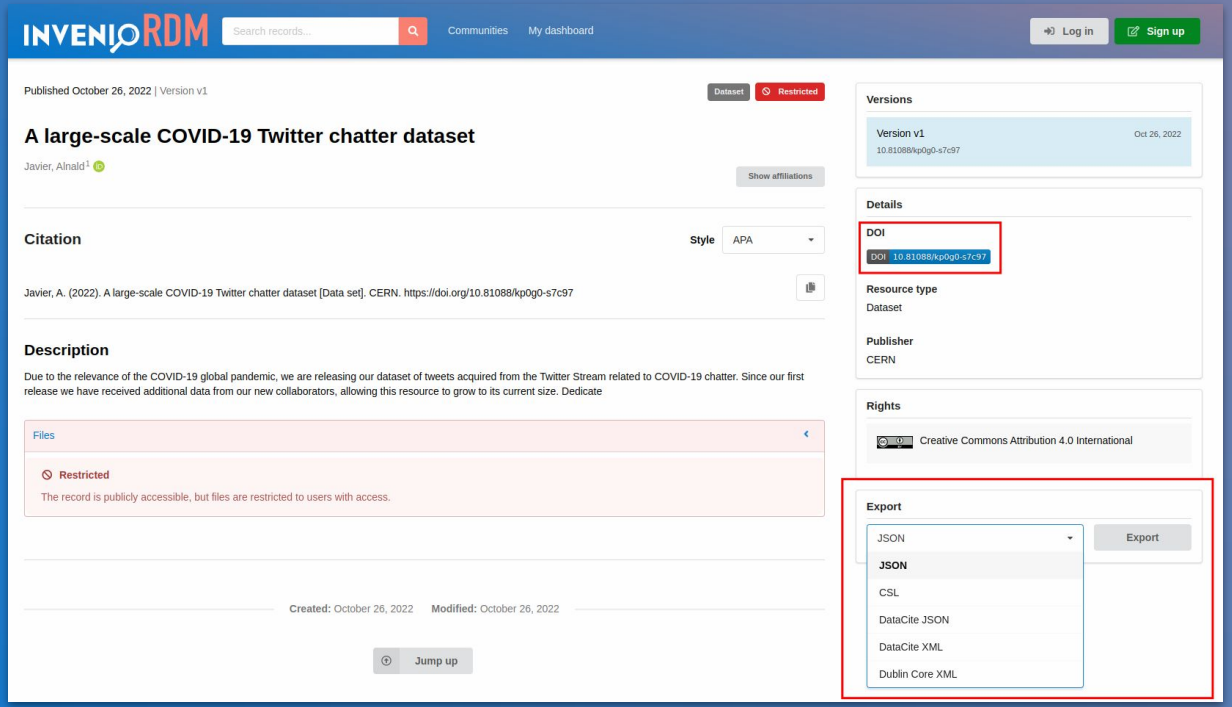
- Rich metadata
- Persistent identifier
- Faceted search
- Advanced query syntax

Accessible



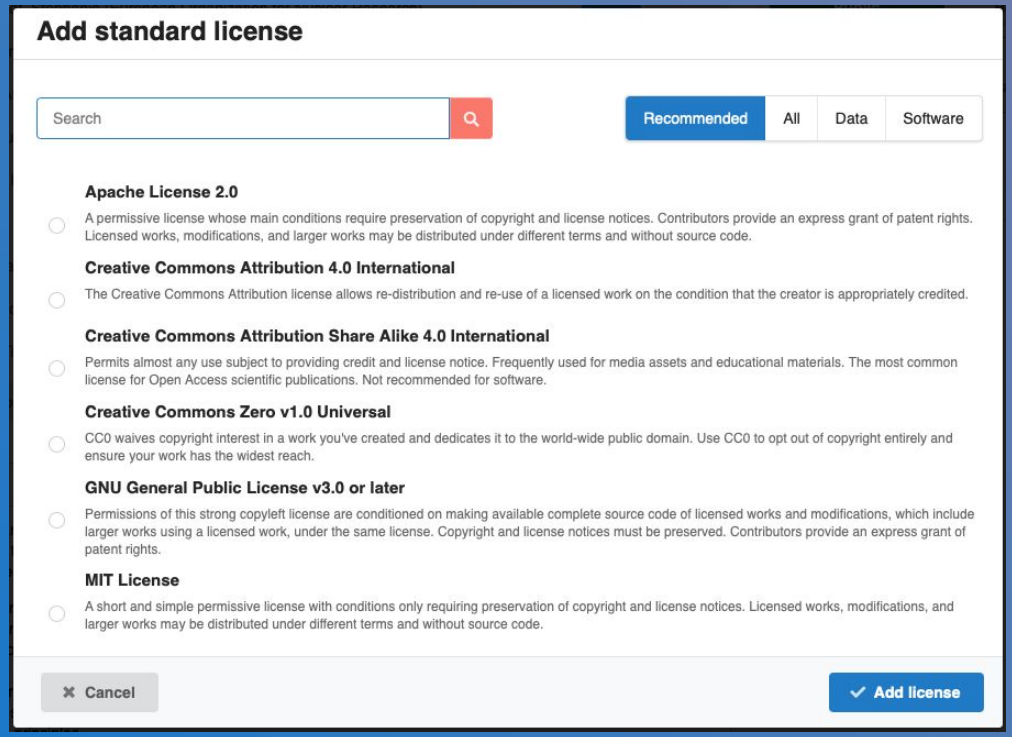
- Metadata accessible, even when data is not available
- Web Content Accessibility Guidelines (WCAG)

Interoperable

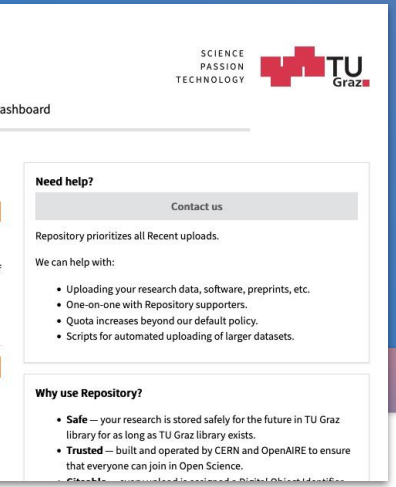


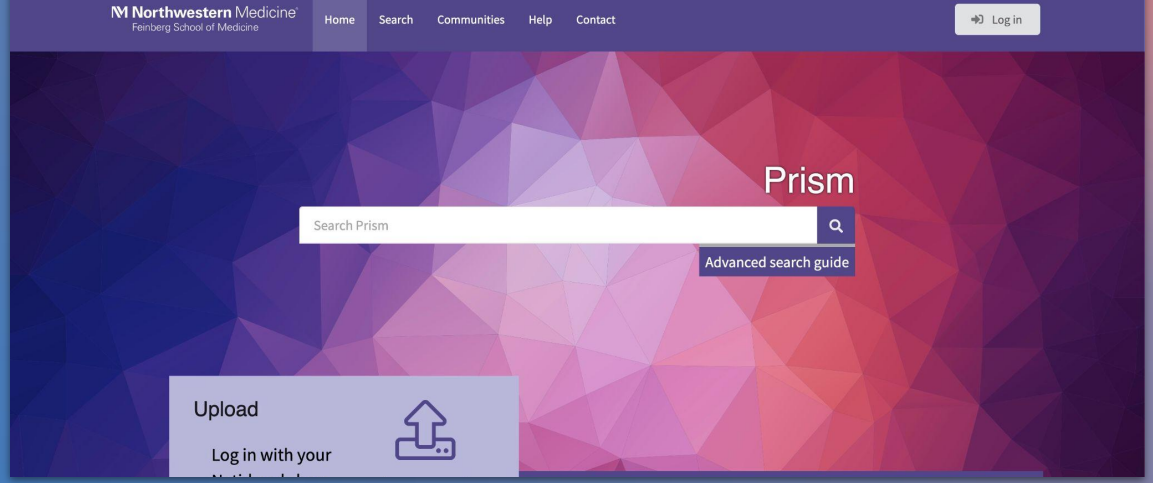
- Strong REST API
- Export formats
- OAI-PMH Server
- DataCite-based metadata
- DOI registration

Reusable

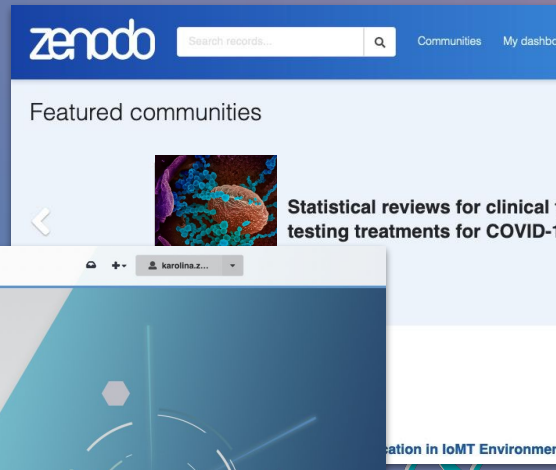


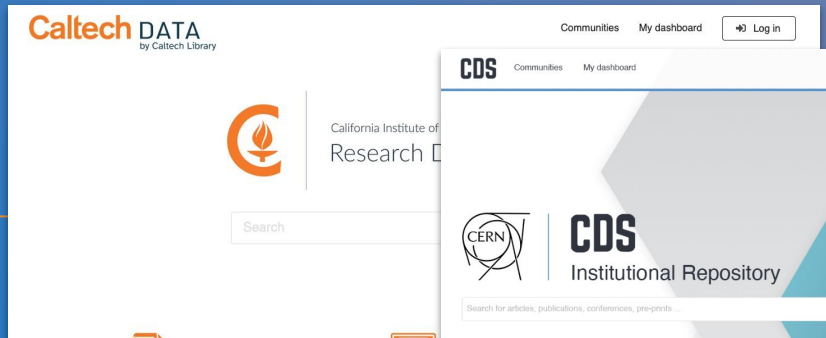
- Persistent identifiers
- Clear license usage
- Rich metadata
- Permission
- Version control
- Standard formats

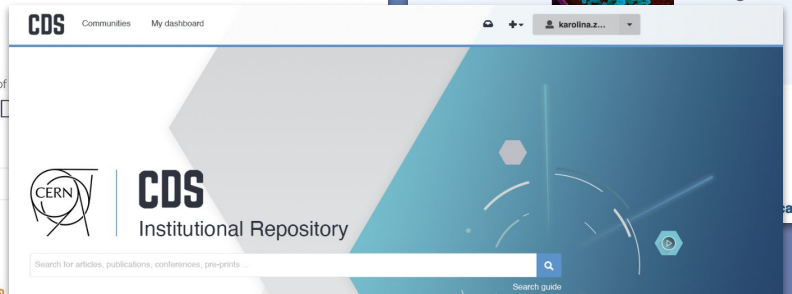




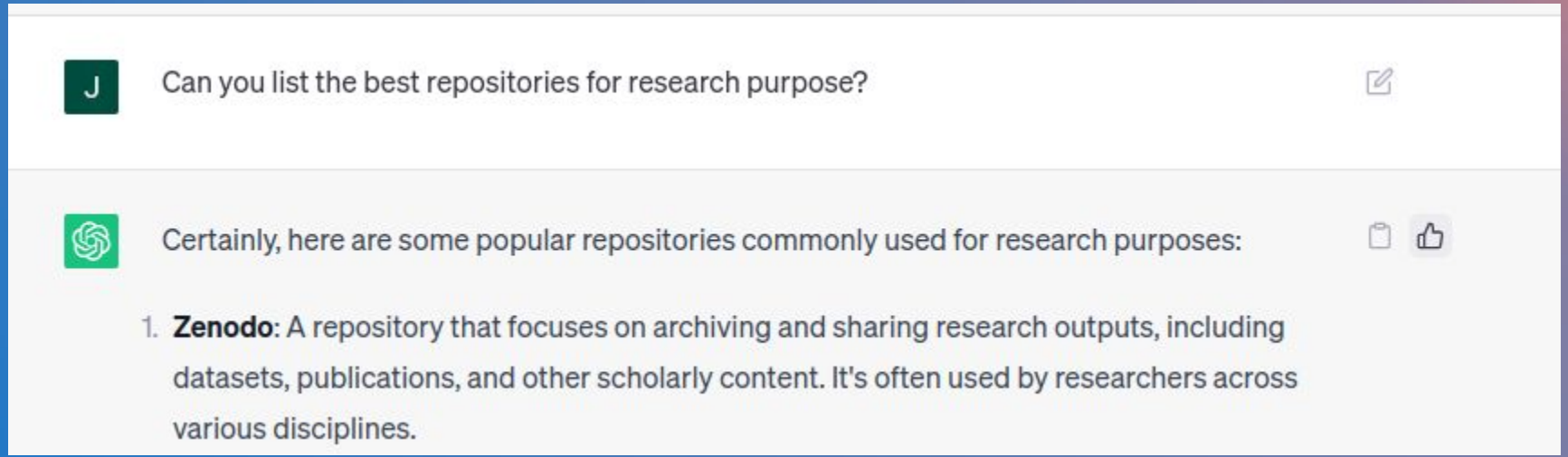
Customizable & Configurable







Questions



https://zenodo.org/

ETH zürich

PocketCoffea

a configuration layer for CMS analyses with Coffea

Matteo Marchegiani

CMS

CERN School of Computing 2023
September 1st, 2023

Key concepts

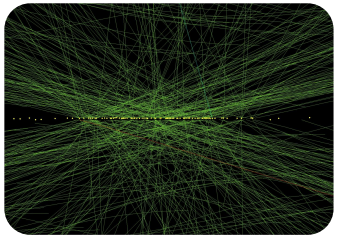
- In view of Run-3 and HL-LHC, state-of-the-art analysis tools need to be employed to cope with larger and larger datasets
- Compressing the information of particle collisions into the smallest format as possible is crucial to achieve the **highest computational performance**
- We present a CMS analysis framework with the aim of:

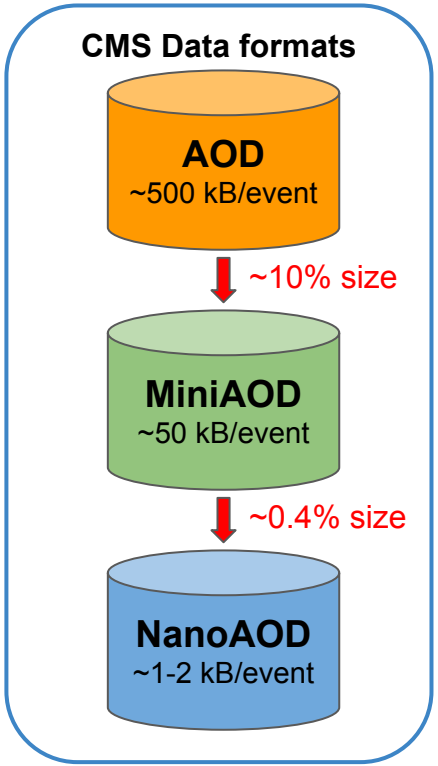
  → **user-friendliness**

  → **reproducibility**

  → **efficiency**

The focus of today's presentation will be on the computing related aspects of the analysis framework.





Reusable CMS-specific analysis structure: multiple Coffea analyses, sharing most of the code base, driven by configuration.

Configuration layer for *CMS* analyses

DASK XRootD HTCondor

scale

Utilities for general HEP analysis. Processor structure and scaling infrastructure

Scikit HEP Boost Histogram mplhep VECTOR

Awkward Array

Data manipulation layer

matplotlib

[github](#)

[readthedocs](#)

Complete installation guide [here](#)

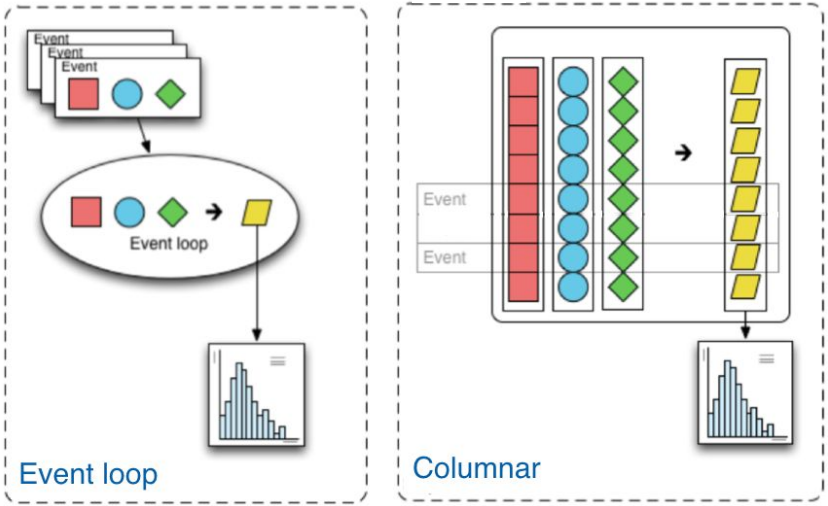
uproot

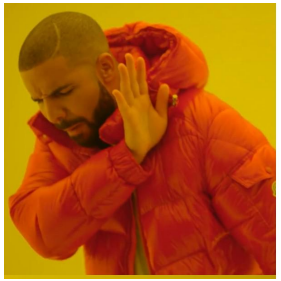
ROOT Data Analysis Framework

Storage I/O layer

Coffea - Columnar Object Framework For Effective Analysis

- Pure python package implementing the typical needs of a HEP analysis
  - → Analysis on **high-level objects**: electrons, muons, jets...
  - → Operations on **4-vectors**
  - → Apply **corrections to MC** events
- Relies on ROOT to read files only
- Features **vectorization** of operations
  - → Faster than traditional event loop
- Possibility to **run in parallel** and scale-out on computer clusters
  - → using schedulers such as Dask, Spark, parsl

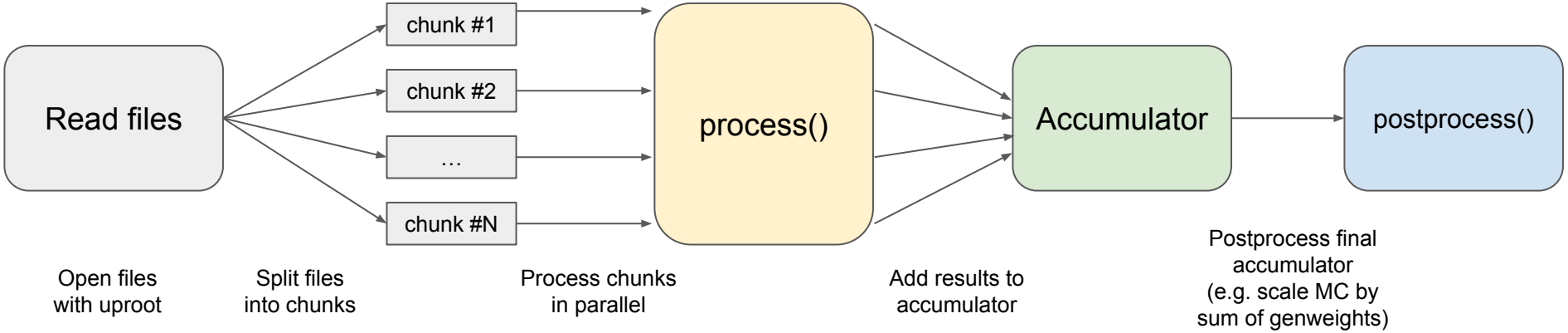




Coffea processor

The general idea of the Coffea processor is:

- Open and read files with **uproot**
- For each file, the events are split into **chunks** (e.g. chunks of 400k events each)
- Each chunk is processed independently by **process()**
- The output histograms and/or ntuples are added chunk by chunk to an **“accumulator”** object
- Operations on the full processor output are performed in **postprocess()**



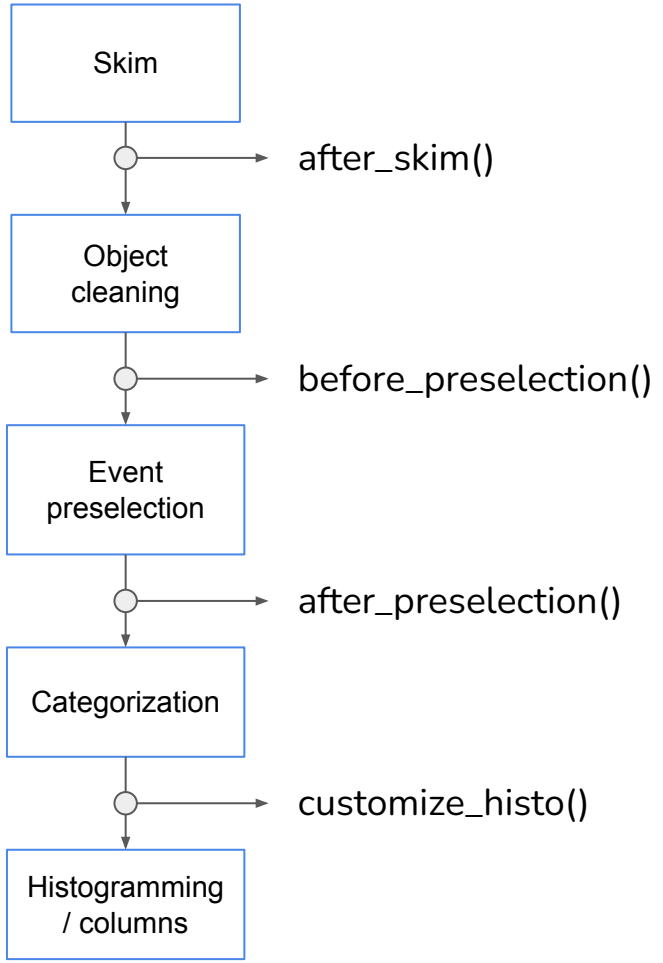
A customizable CMS analysis workflow

PocketCoffea implements a Coffea processor **containing all the standard steps of a CMS templated analysis:**

- Skimming of events
- Object preselection and calibration
- Event preselections
- Categorization
- Histogramming
- Export of ntuples

Custom workflow implemented as **derived processor class** with **predefinite entry points**

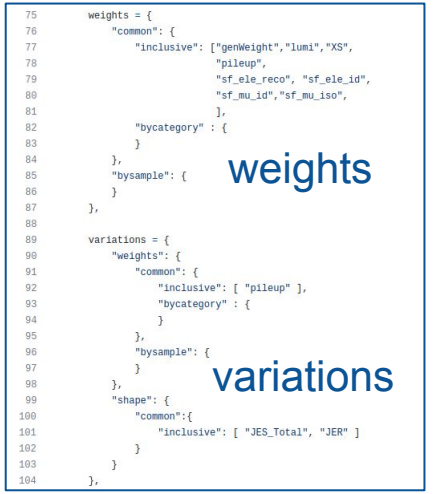
For these customizations, the user needs to **expand the** Base processor **code** and/or the libraries containing the parameters and cut functions

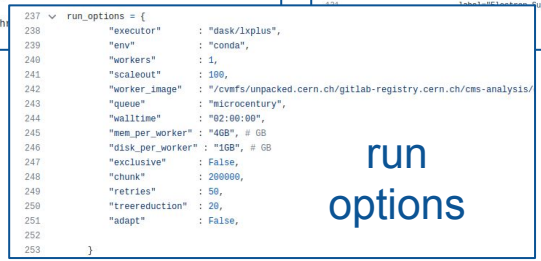


Configuration

Keep track of all the parameters in a **single config file**

- Define all the relevant parameters for **running**:
  - → input datasets to process
  - → workflow and output folder
  - → executor parameters
- Define **cutflow** by defining cut functions **dynamically**
  - → skimming, preselections and categorization
- Define **weights** to apply by category / by sample
- Define **variations** to store by category / by sample
- List **histograms** to be produced and parameters:
  - → customize binning, labels, etc.
- Additional **analysis-specific parameters** can be also defined





Scaling up with Dask

Coffea offers 3 main modes for processing:

- **Iterative executor**: sequential processing on 1 CPU
- **Futures executor**: parallel processing on multiple CPUs on local machine
- **Scaleout on a computer cluster** using a scheduler, parallel processing on multiple CPUs

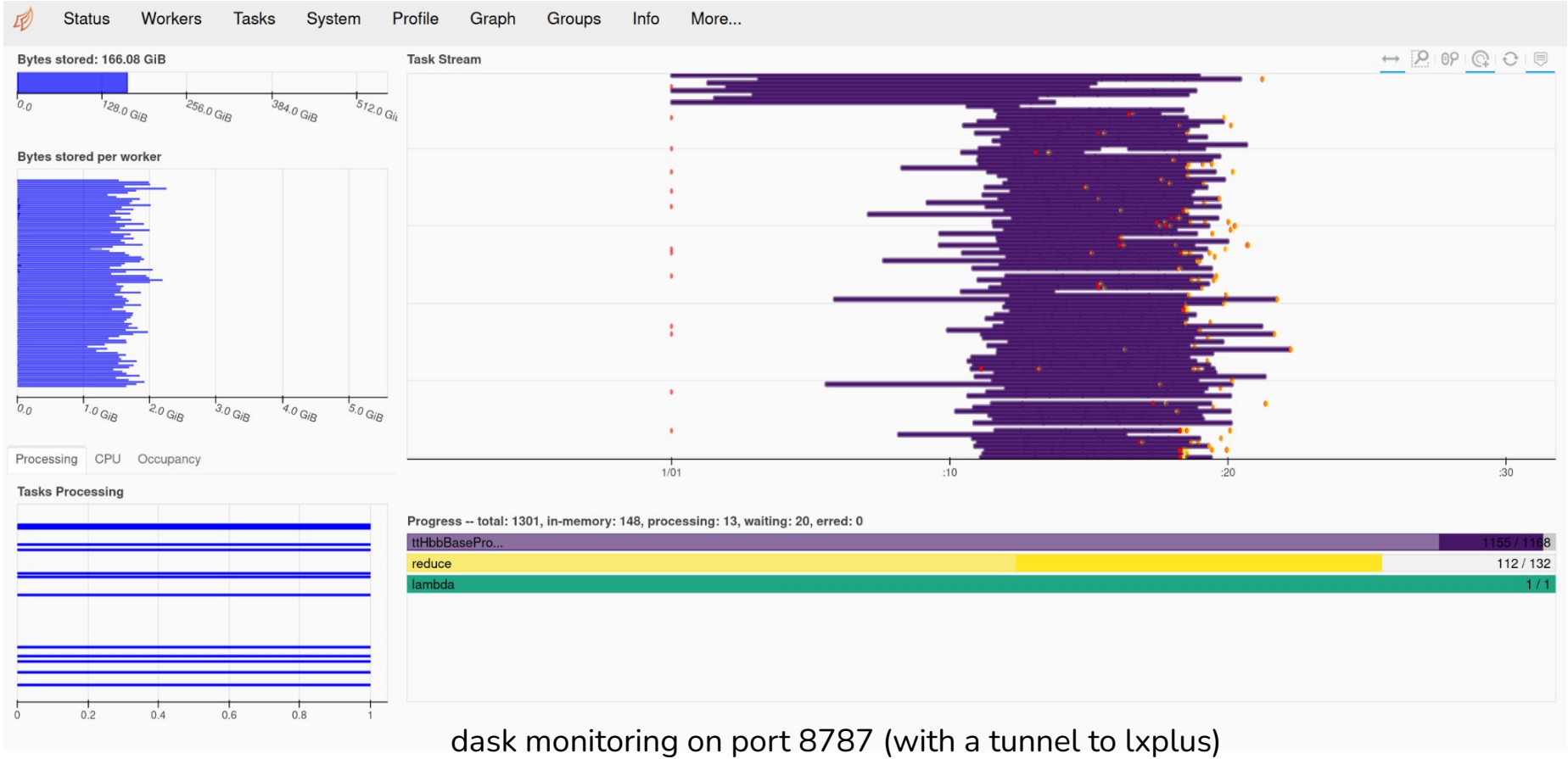
The coffea processor starts a **Dask scheduler** (local) + **N worker jobs** (condor).

- the scheduler splits the dataset in chunks
- chunks are sent to workers automatically to be elaborated
- the Dask scheduler handles the job splitting dynamically
- if a worker fails the corresponding task is rescheduled

In practice:

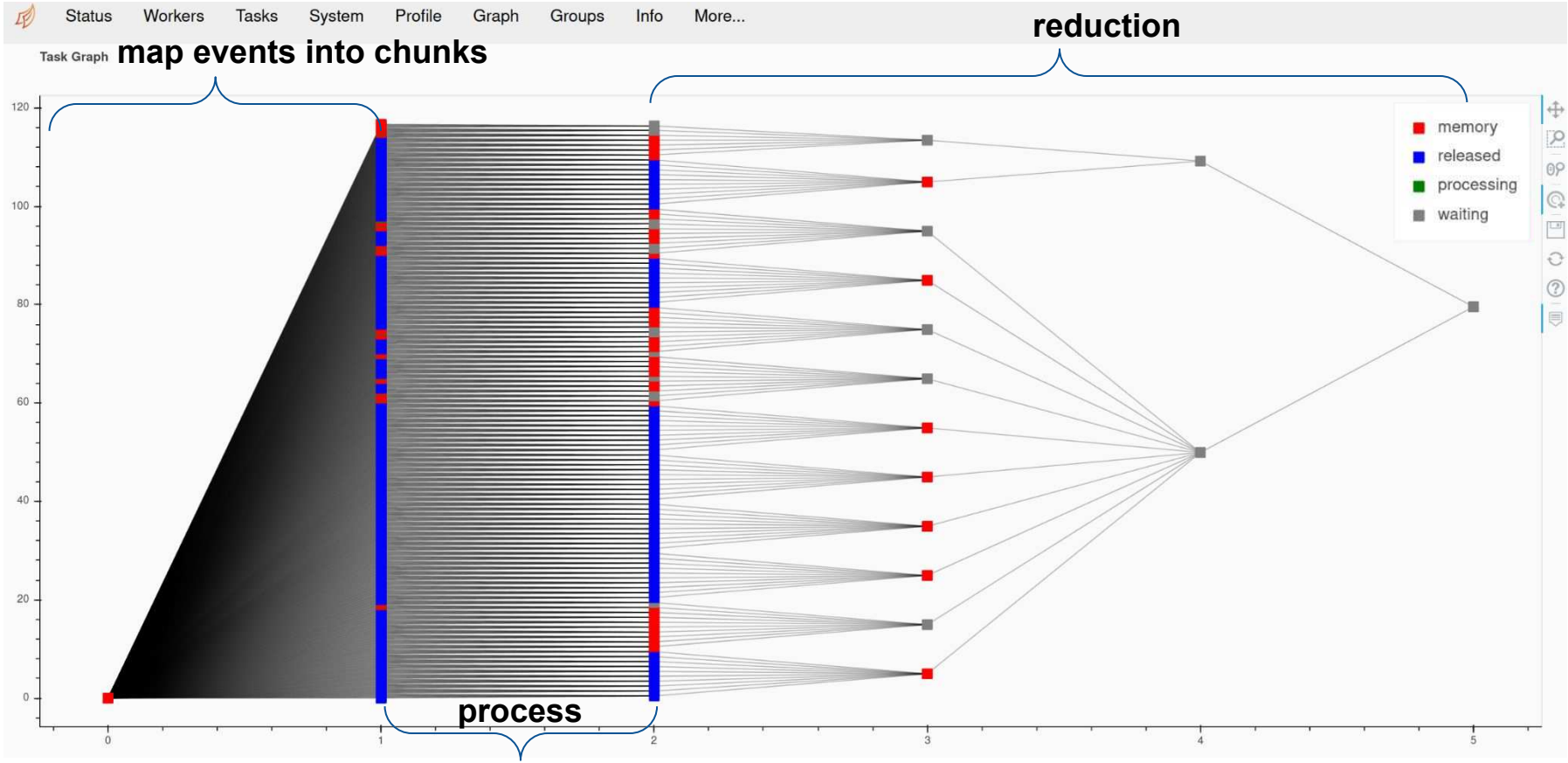
- less job sitting
- **ability to run a full CMS analysis in ~hours** (whereas typical Run 2 analyses could take even days)

Jobs monitoring with Dask



dask monitoring on port 8787 (with a tunnel to lxplus)

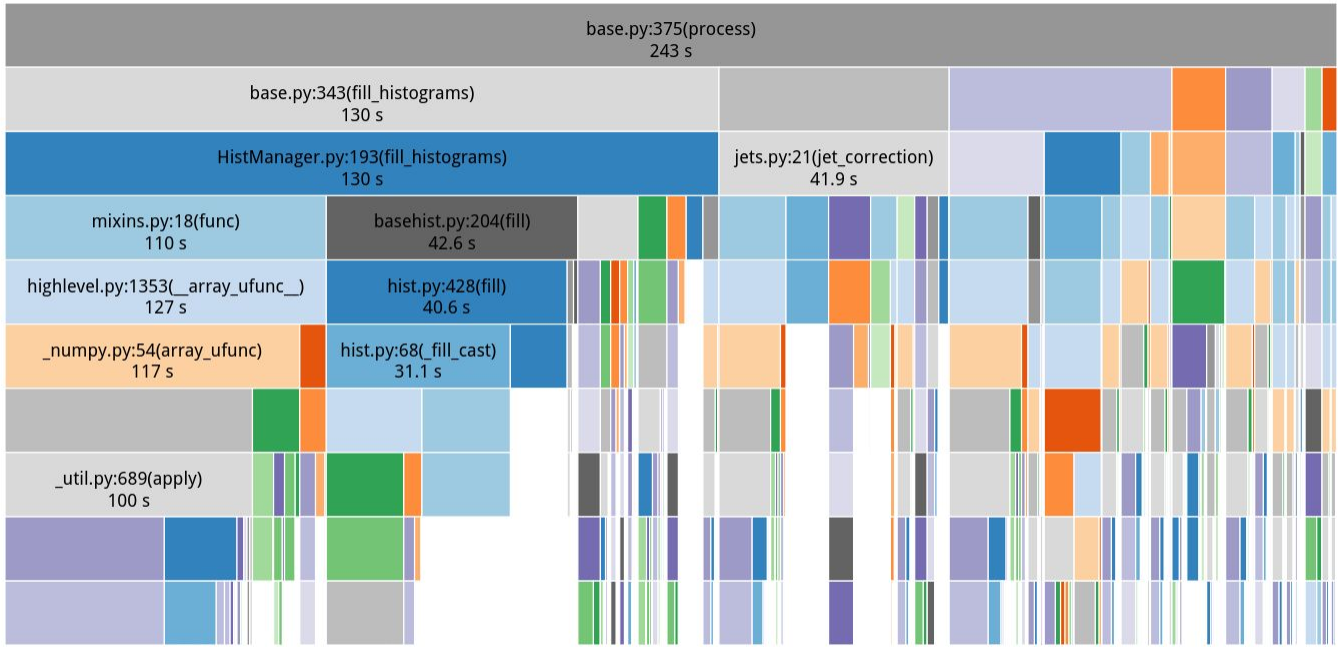
Dask task graph



Performance profiling

Monitor the performance of the processor chain is very important
→ very easy in PocketCoffea with python *cProfile* and *snakeviz* visualization

```
python -m cProfile -o profiling/output.prof scripts/runner.py --cfg config.py --test -lf 10
```



Function call stack of a typical PocketCoffea processor

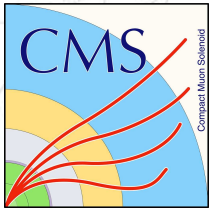
Very useful to keep an eye on the optimization of each step of the processing.

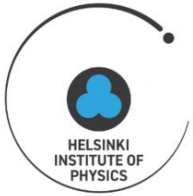
Filling the histograms for many categories and variations **is the slowest operation** (*good!!*)

Conclusions

- PocketCoffea is a CMS analysis framework to perform efficient analysis in a configurable way
- Code reusability thanks to class inheritance of processors
- Provides a general tool to share knowledge among different analyses
- The framework is being used for Run 2 analysis and could be used for the analysis of Run 3 data in the future

Thank you for the attention!





Data Scouting Jet for Run 3 at CMS

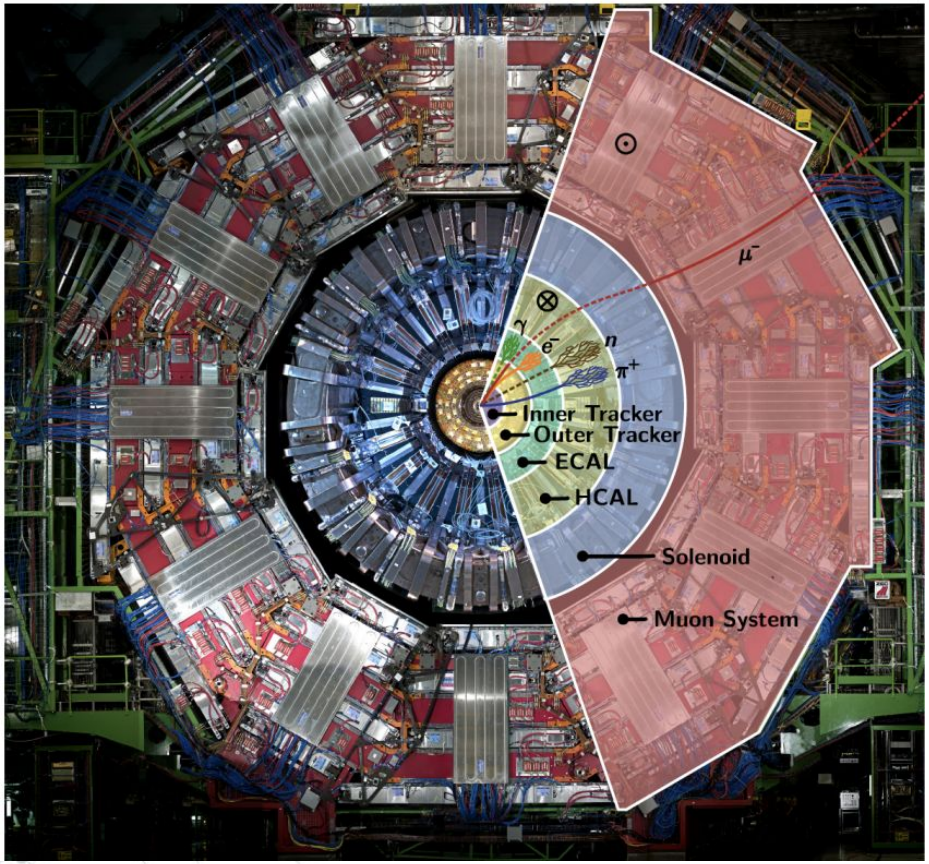
Patin Inkaew
PhD student at Helsinki Institute of Physics, Finland
patin.inkaew@cern.ch

1 September 2023
CERN School of Computing

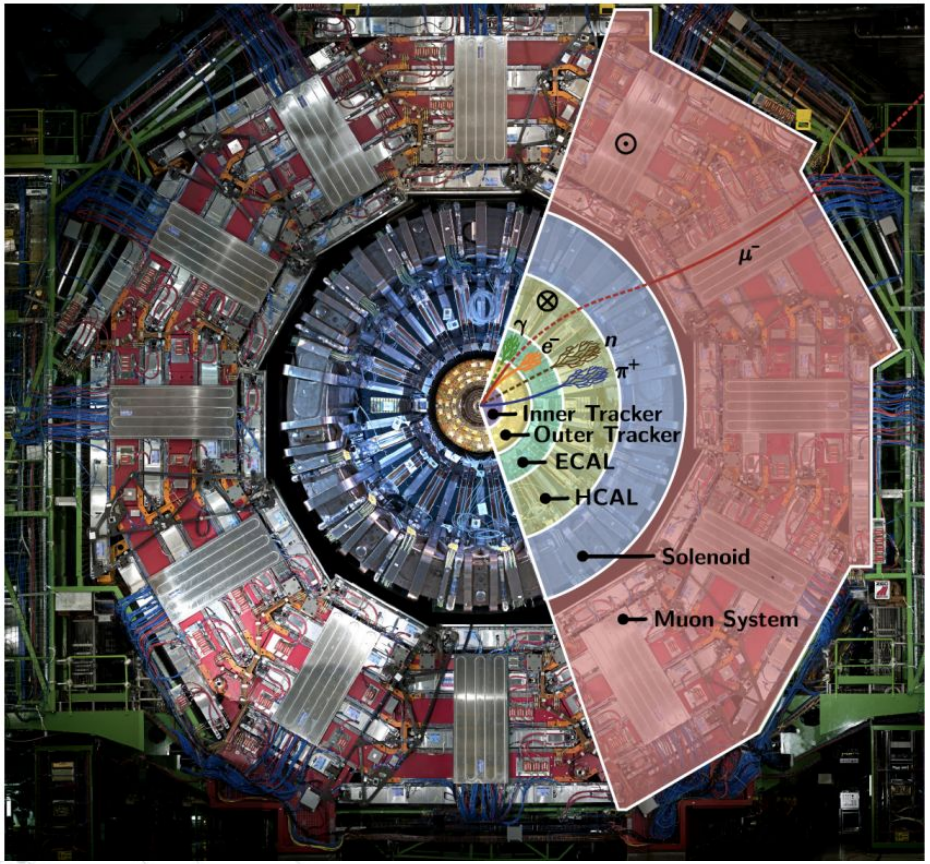
Data Scouting Jet for Run 3 at CMS

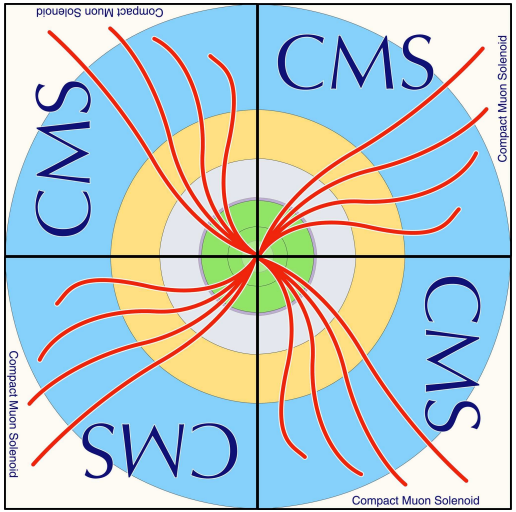
Data Scouting Jet for Run 3 at **CMS**

Compact Muon Solenoid (CMS)



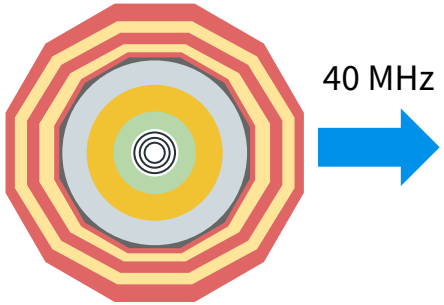
Compact Muon Solenoid (CMS)



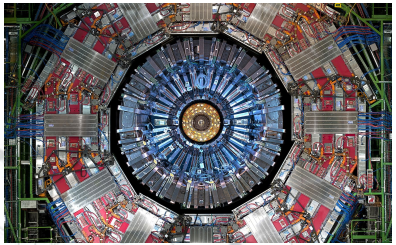


CMS Trigger

LHC collides pp every ~25 ns = 40 MHz

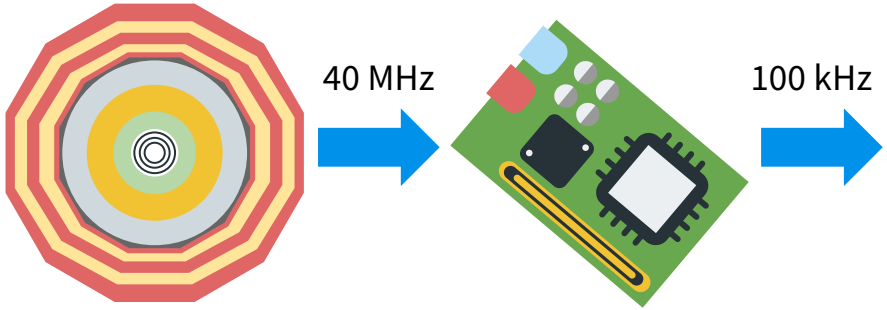


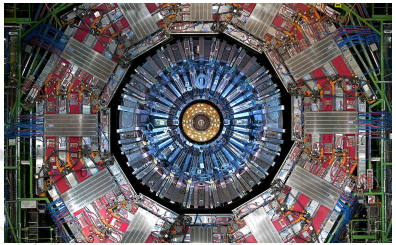
CMS Detector

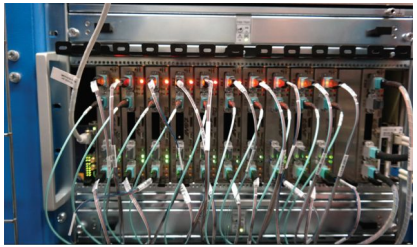


CMS Trigger

L1 (Hardware based on FPGA) reduces rate to ~ 100 kHz

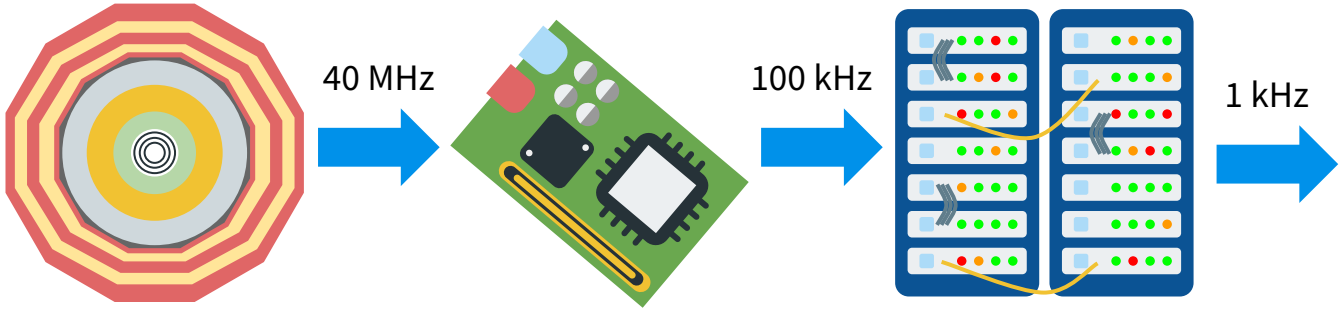




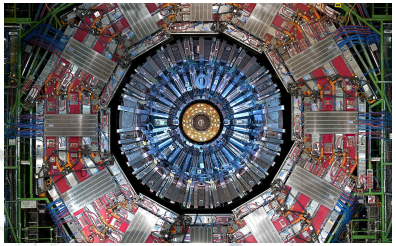


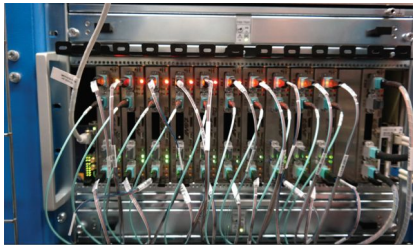
CMS Trigger

HLT (Computer farm) reduces rate to ~ 1 kHz



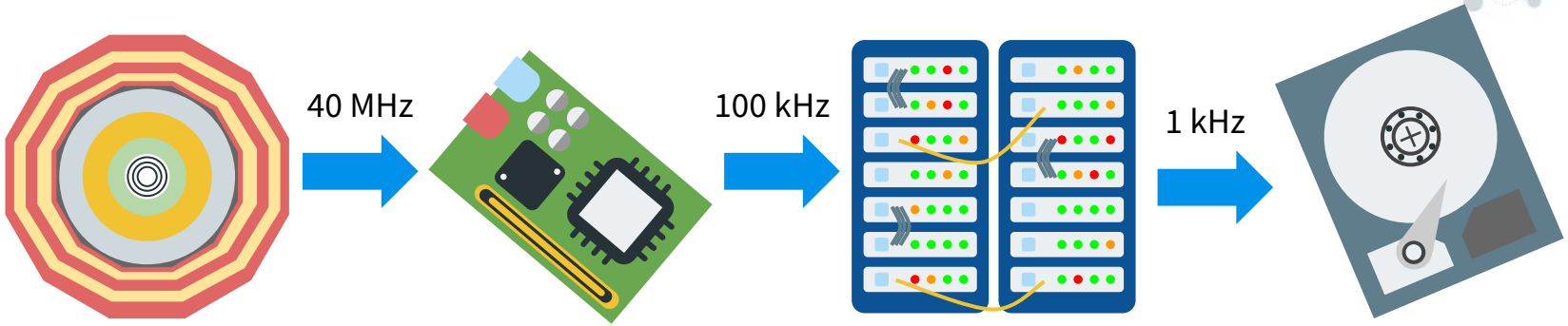
CMS Detector | L1 Trigger | High Level Trigger

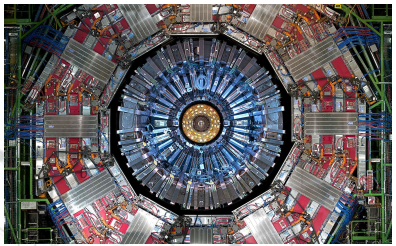


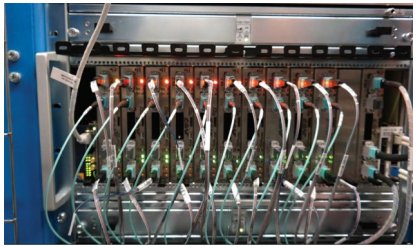


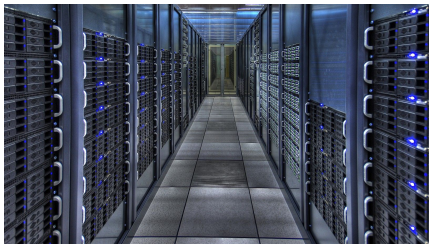
CMS Trigger

Data is transferred and stored at Tier 0



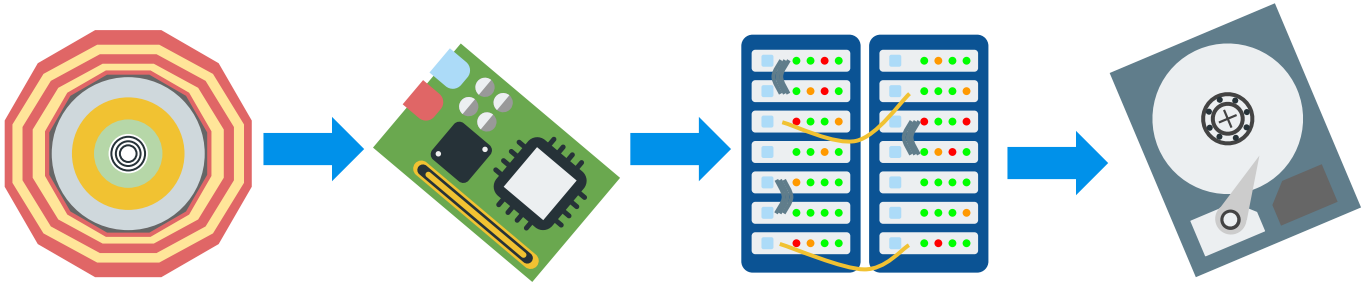




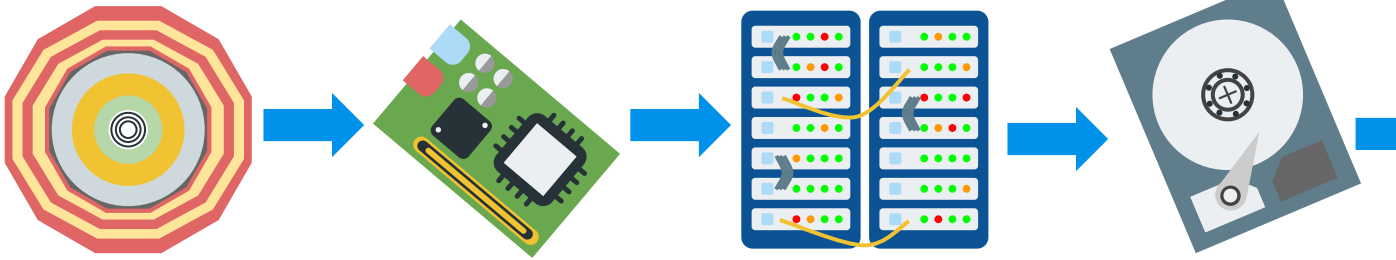


Data Scouting Jet for Run 3 at CMS

Data Scouting



Data Scouting

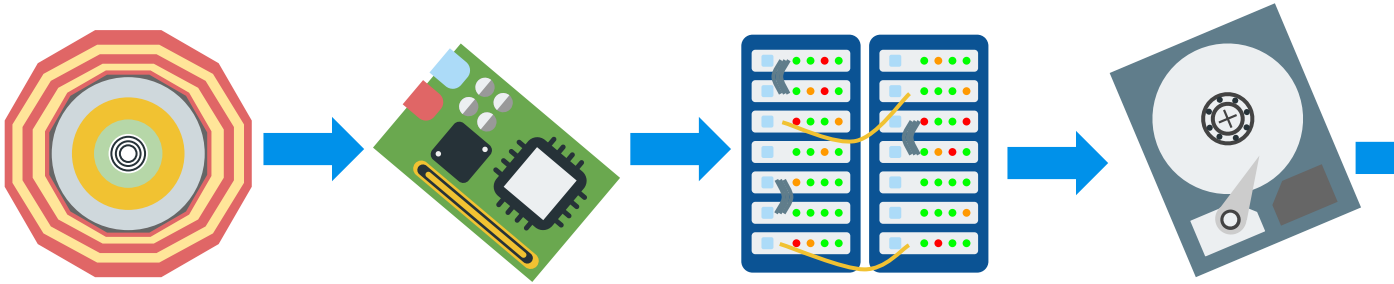


CMS L1 Trigger HLT Storage

Standard Stream

- ~ 1 kHz
- ~ 1.5 MB/event
- full event
- reconstructed at Tier0 within 48 hours

Data Scouting



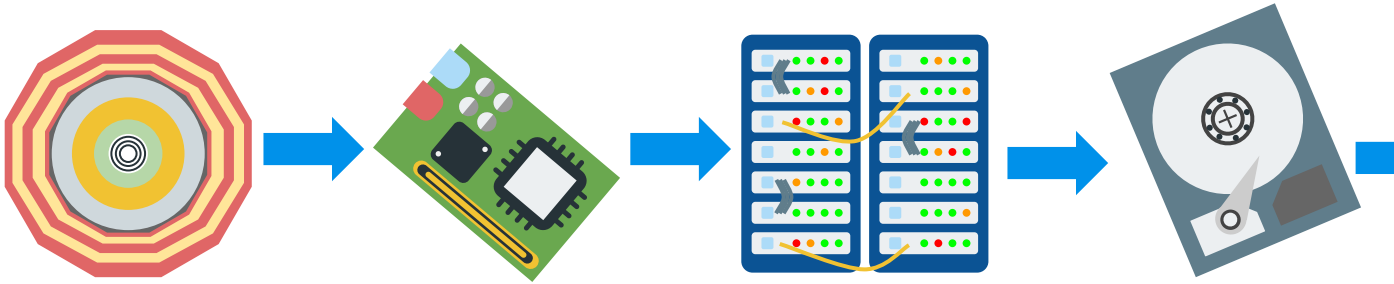
CMS L1 Trigger HLT Storage

Standard Stream

- ~ 1 kHz
- ~ 1.5 MB/event
- full event
- reconstructed at Tier0 within 48 hours

The real bottleneck is data recording rate (MB/sec), not event rate (event/sec)

Data Scouting



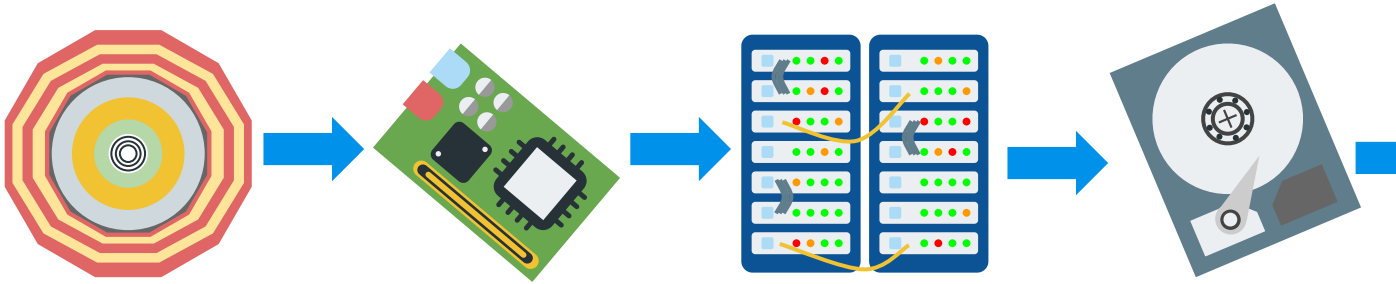
Standard Stream

- ~ 1 kHz
- ~ 1.5 MB/event
- full event
- reconstructed at Tier0 within 48 hours

The real bottleneck is data recording rate (MB/sec), not event rate (event/sec)

Data writing rate (MB/s) = event size (MB) x rate (Hz)

Data Scouting



CMS L1 Trigger HLT Storage

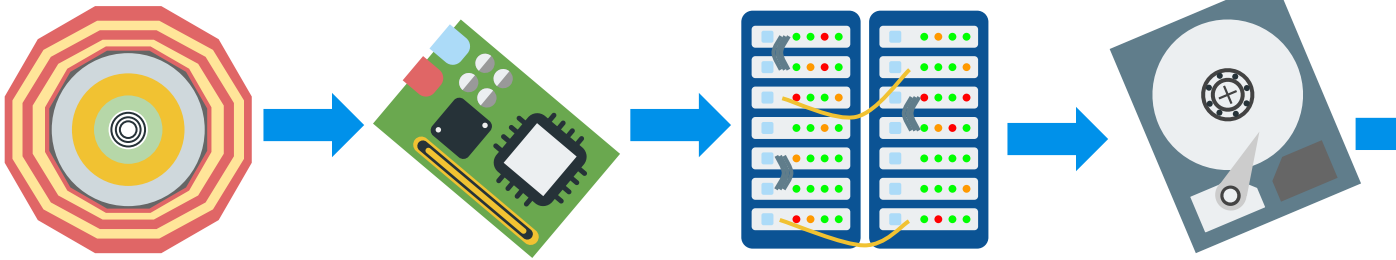
Standard Stream

- ~ 1 kHz
- ~ 1.5 MB/event
- full event
- reconstructed at Tier0 within 48 hours

The real bottleneck is data recording rate (MB/sec), not event rate (event/sec)

Data writing rate (MB/s) = event size (MB) x rate (Hz)

Data Scouting



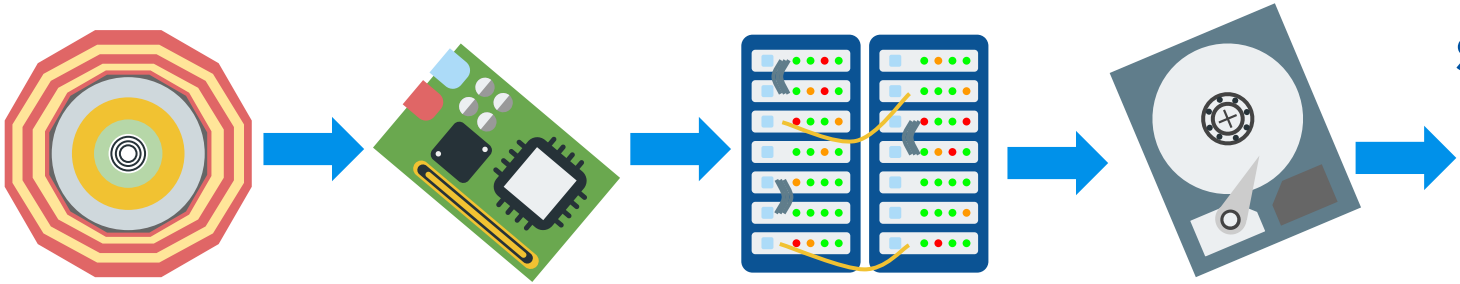
Standard Stream

- ~ 1 kHz
- ~ 1.5 MB/event
- full event
- reconstructed at Tier0 within 48 hours

The real bottleneck is data recording rate (MB/sec), not event rate (event/sec)

Data writing rate (MB/s) = event size (MB) x rate (Hz)

Data Scouting



CMS L1 Trigger HLT Storage

Standard Stream

- ~ 1 kHz
- ~ 1.5 MB/event
- full event
- reconstructed at Tier0 within 48 hours

The real bottleneck is data recording rate (MB/sec), not event rate (event/sec)

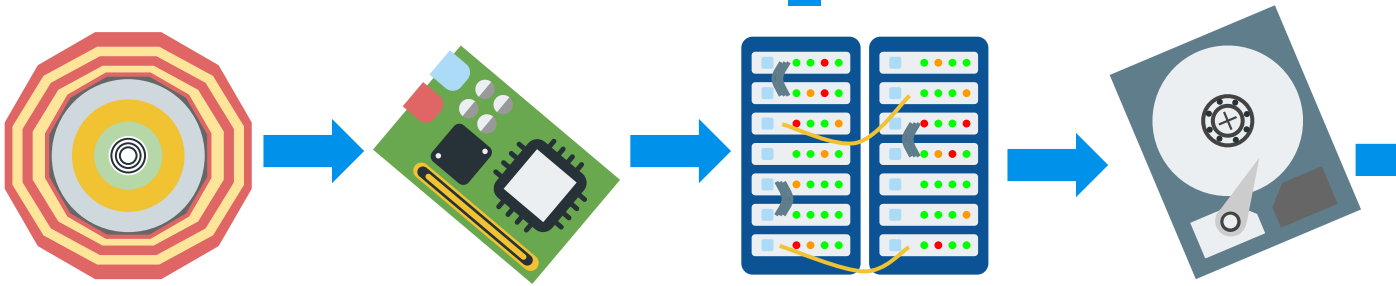
Data writing rate (MB/s) = event size (MB) x rate (Hz)

only keep smaller high-level description, and discard raw detector signals

Data Scouting

Scouting Stream

- reduced data format
- event size reduced by 100x
- event rate increased by 30x
- based on reconstruction at HLT (no offline reconstruction)



CMS L1 Trigger HLT Storage

Standard Stream

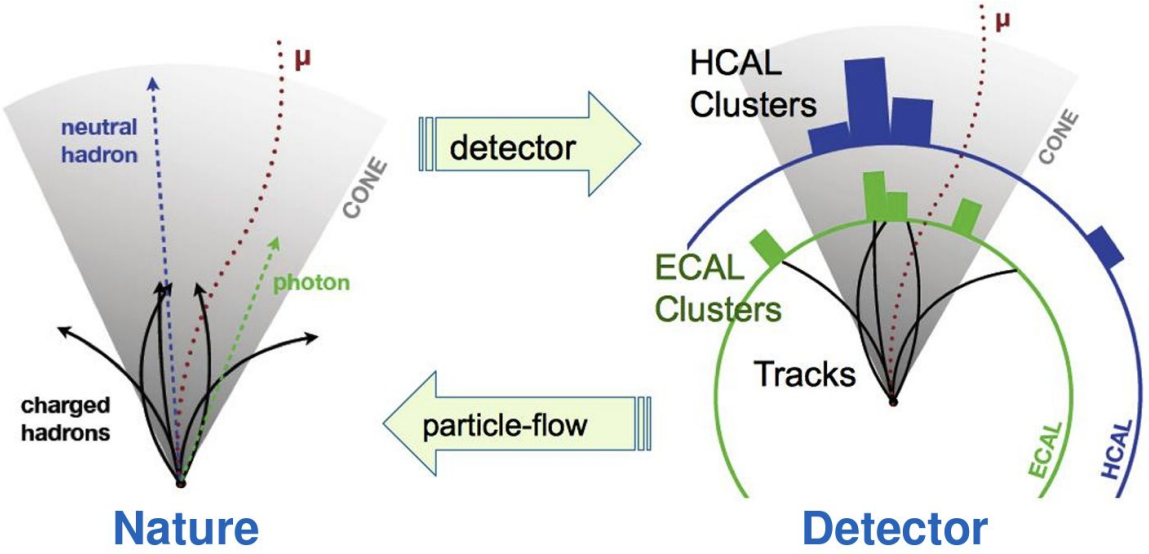
- ~ 1 kHz
- ~ 1.5 MB/event
- full event
- reconstructed at Tier0 within 48 hours

The real bottleneck is data recording rate (MB/sec), not event rate (event/sec)

Data writing rate (MB/s) = event size (MB) x rate (Hz)

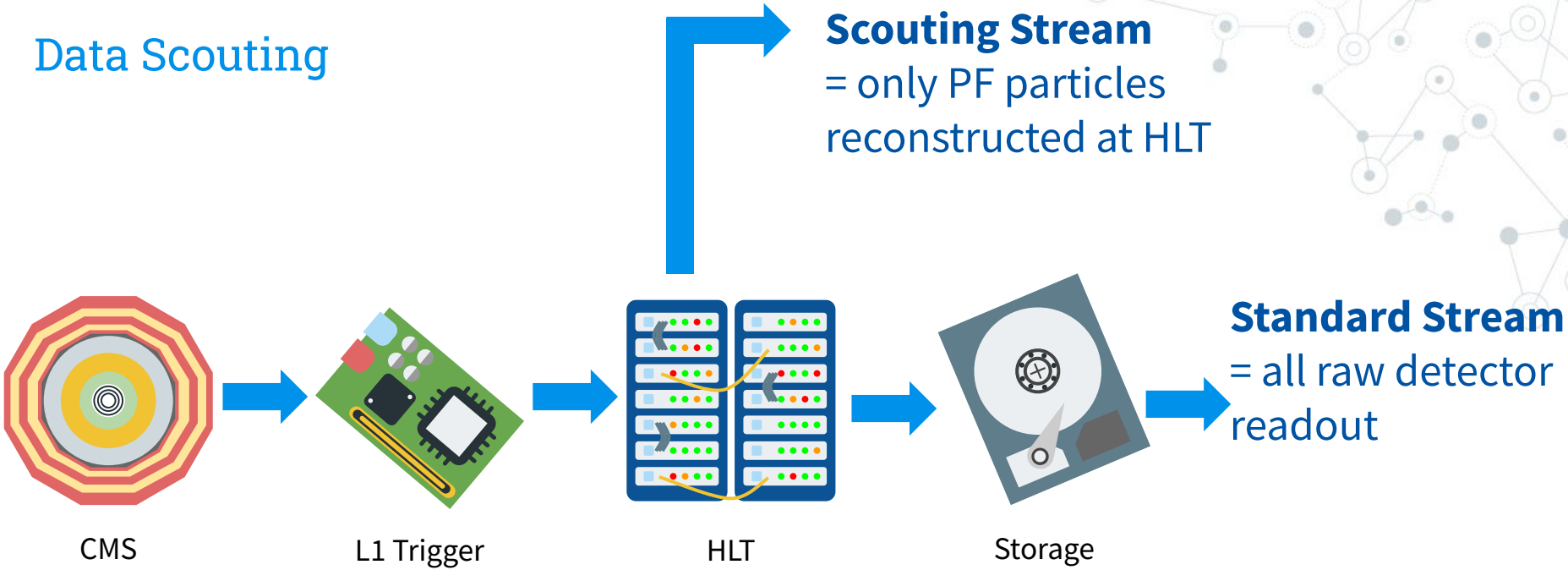
only keep smaller high-level description, and discard raw detector signals

Interlude: Particle Flow (PF)



PF combines all sub-detector primitives (tracks, clusters) to produce physical objects: electron, photon, muon, neutral and charged hadrons.

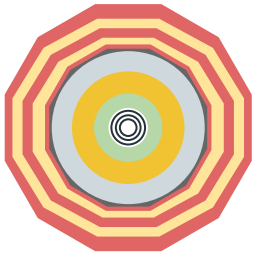
Data Scouting

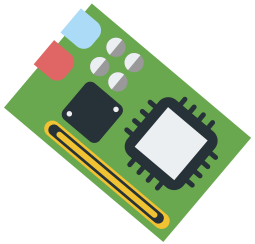


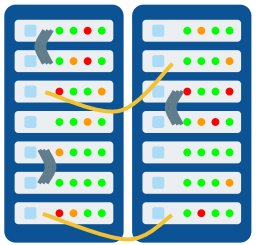
HL-LHC Upgrade: L1 Scouting

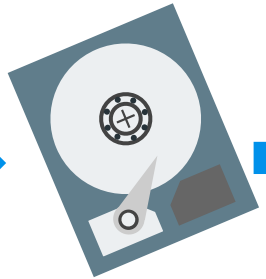
HL-LHC: L1 Scouting

Scouting Stream
= only PF particles reconstructed at HLT









Standard Stream
= all raw detector readout

CMS

L1 Trigger

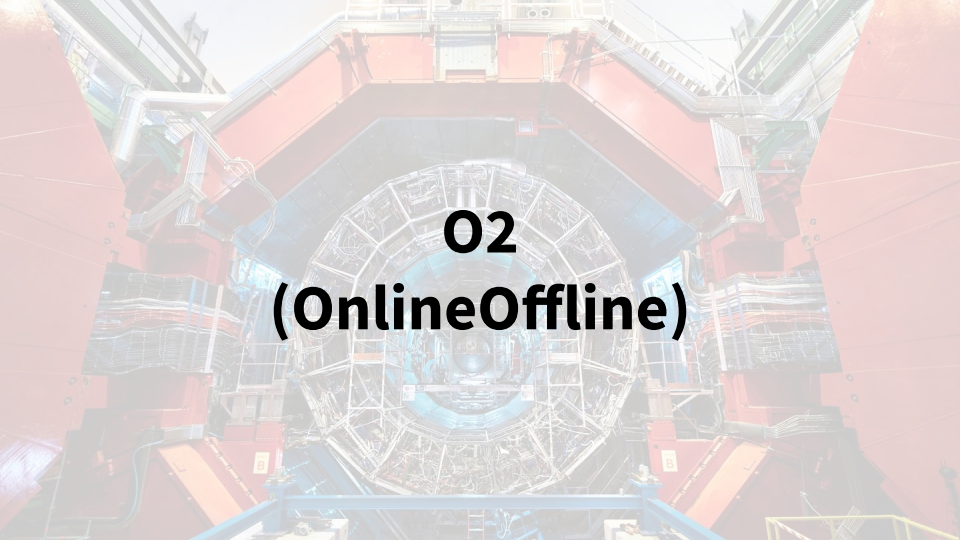
HLT

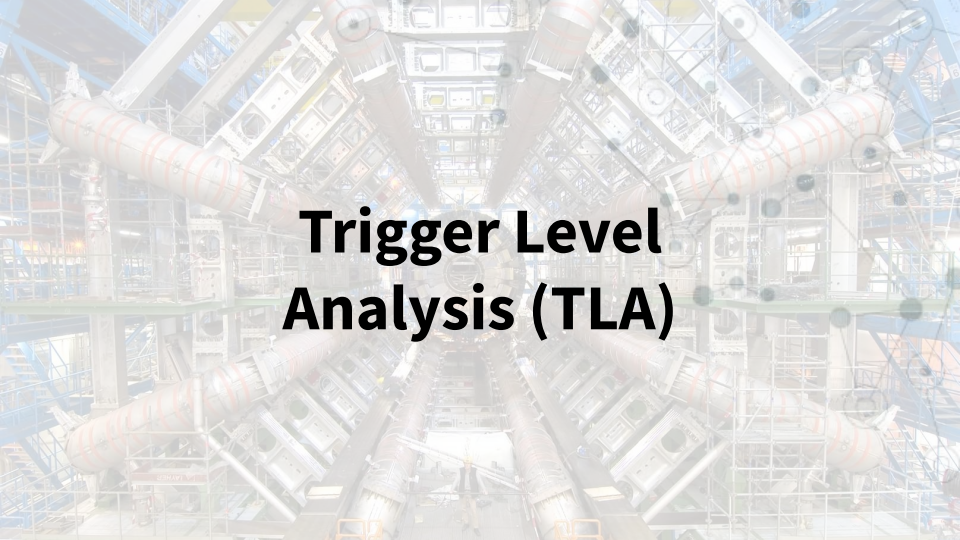
Storage

L1 Scouting (40 MHz)
= L1 Primitives

HiLumi HL-LHC PROJECT

RTA in LHC experiments

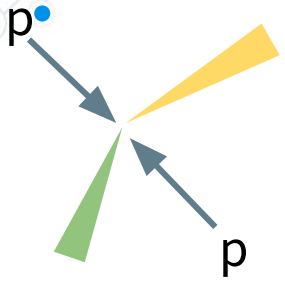

O2
(OnlineOffline)


Trigger Level
Analysis (TLA)


Data Scouting

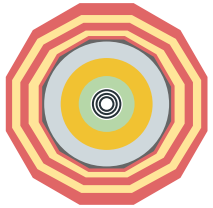

Turbo stream
LHCb

Data Scouting Jet for Run 3 at CMS

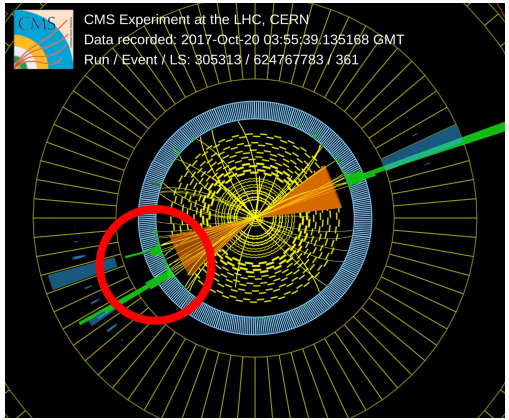


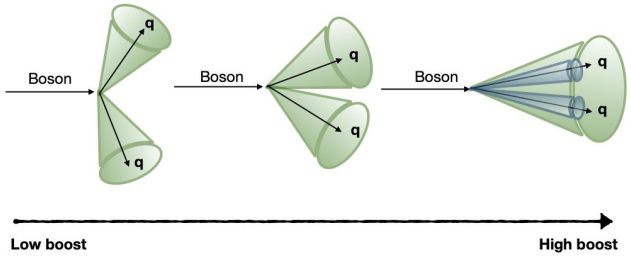
Detector Response

Jet Energy Correction (JEC)



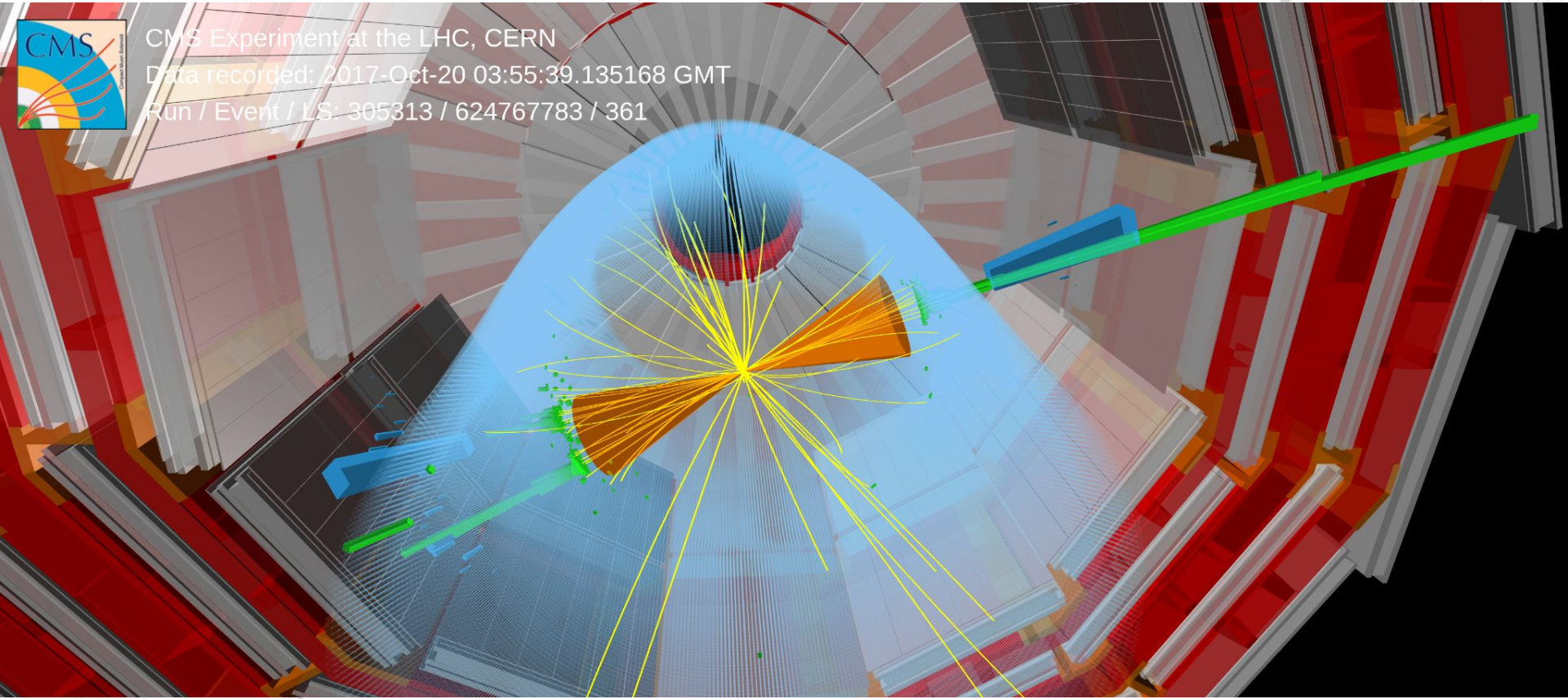
1. Jet Energy Correction (JEC) for scouting
2. Boosted H → bb analysis using scouting jets





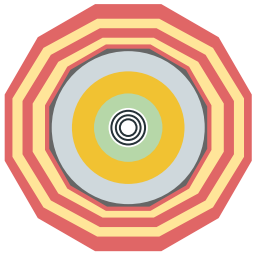
Candidate event for highly-boosted H → bb

Thank you!

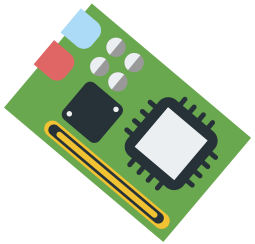


full slide

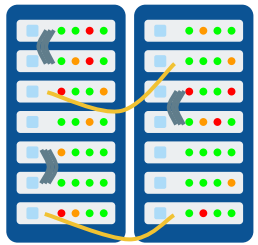
Data Parking



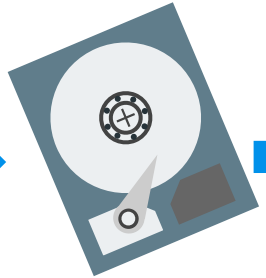
CMS



L1 Trigger



HLT



Storage

Scouting Stream

- reduced data format
- event size reduced by 100x
- event rate increased by 30x
- based on reconstruction at HLT (no offline reconstruction)

Standard Stream

- ~ 1 kHz
- ~ 1.5 MB/event
- full event
- reconstructed at Tier0 within 48 hours

Parking Stream

- ~3 kHz
- ~ 1.5 MB/event
- full event
- Reconstructed when resources become available