

The Grid on the Cloud

A Vision of Analysis Facilities on 100k Cores

Lukas Heinrich, TUM



Showcasing work by

**F. Megino, Tibor Simko, Nikolai Hartmann, Alessandra Forti, A. Hartin, G. Duckeck, R. Walker
P. Gadow, J. Elmsheuser, A. Klimmentor, K De, M. Lassnig and many more**

ATLAS / LHC centric, but hopefully with wider applicability

ALICE

LHC

[Large Hadron Collider]

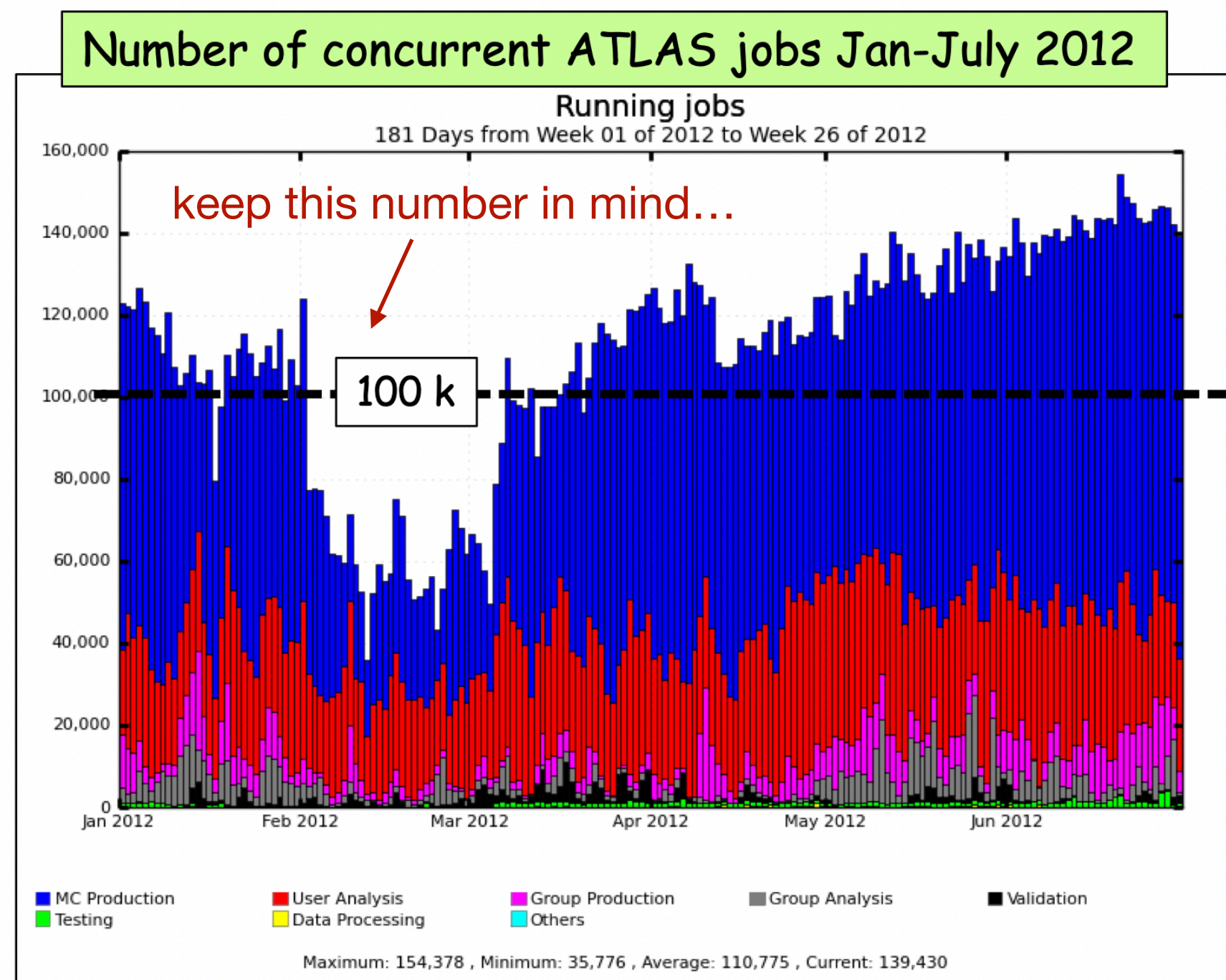
LHCb

ATLAS

LHC Computing is a big success

... after all it gave us this 10 years go ...

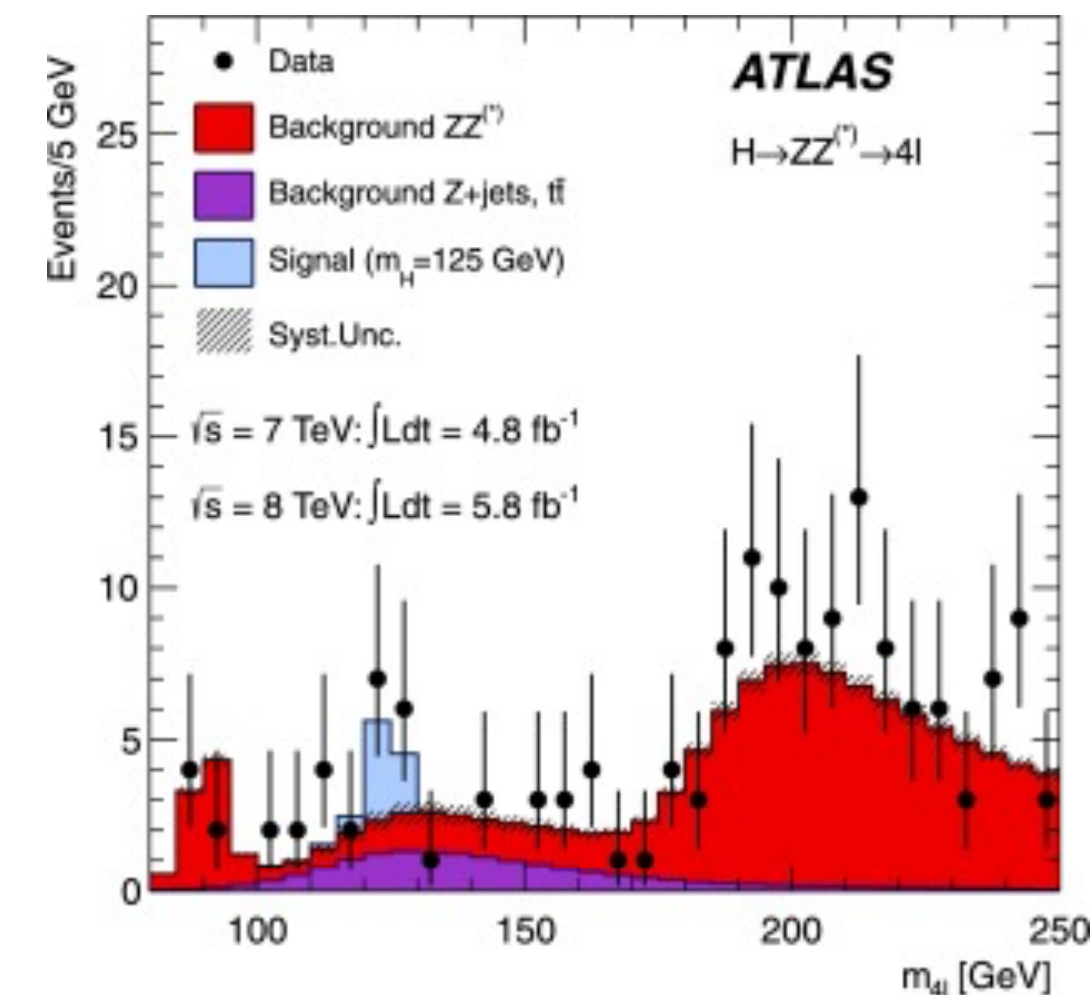
It would have been impossible to release physics results so quickly without the outstanding performance of the Grid (including the CERN Tier-0)



Includes MC production, user and group analysis at CERN, 10 Tier1-s, ~ 70 Tier-2 federations → > 80 sites

> 1500 distinct ATLAS users do analysis on the GRID

- ❑ Available resources fully used/stressed (beyond pledges in some cases)
- ❑ Massive production of 8 TeV Monte Carlo samples
- ❑ Very effective and flexible Computing Model and Operation team → accommodate high trigger rates and pile-up, intense MC simulation, analysis demands from worldwide users (through e.g. dynamic data placement)





The Worldwide LHC Computing Grid

... the cloud before there was a cloud

- Globally federated computing long before there was a cloud
- Commodity x86 platform
- Linux as a common platform choice
- Tight software control to bridge heterogeneity (CVMFS)
- Focused solely on Batch computing
- Distribute, process and analyse LHC data

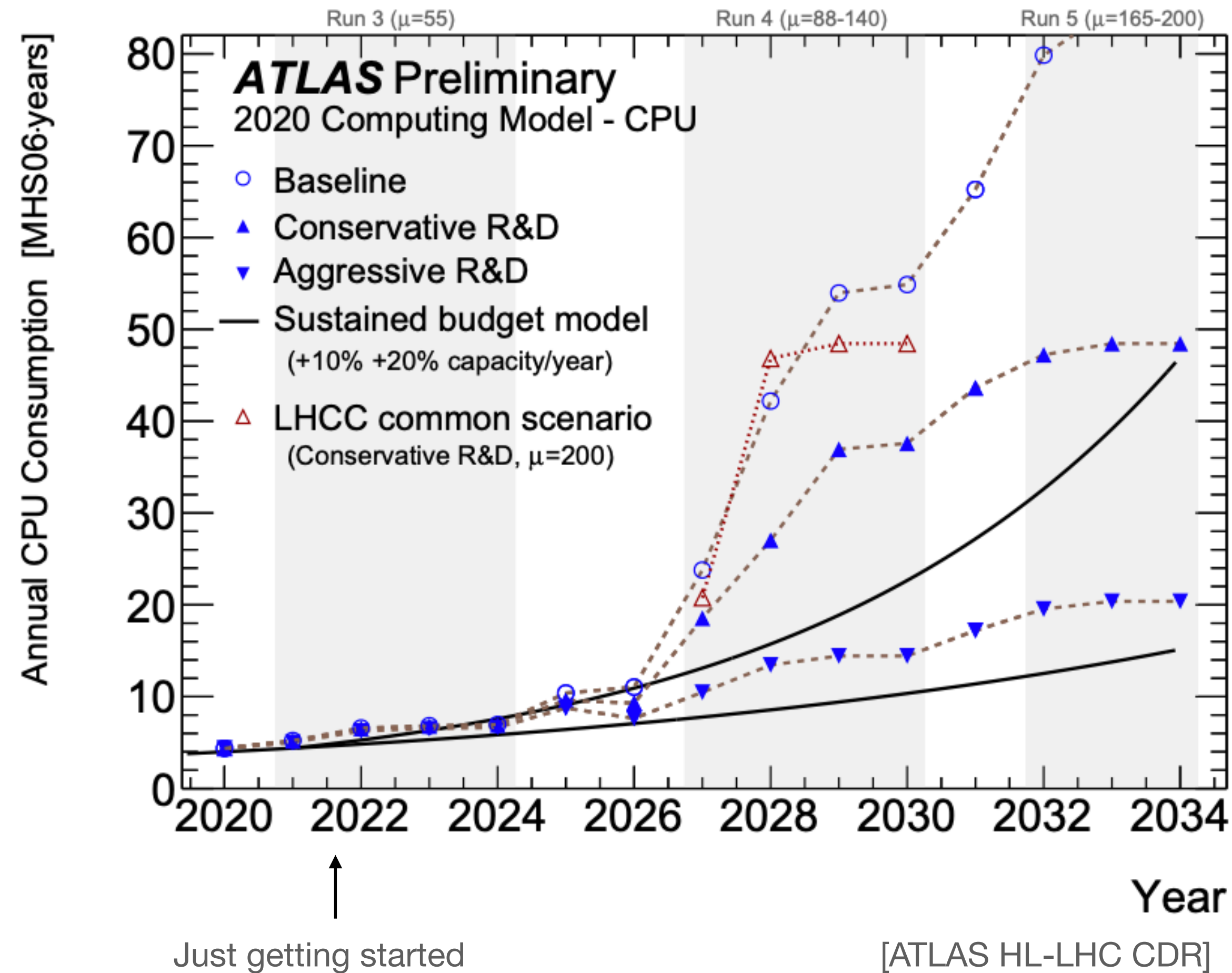
ATLAS stats:
165 data centers
40 countries
800k vCPUs
700 PB managed data
3 Tbps network

Running jobs: 365118
Active CPU cores: 795836
Transfer rate: 18.35 GiB/sec



The Run-3 is here

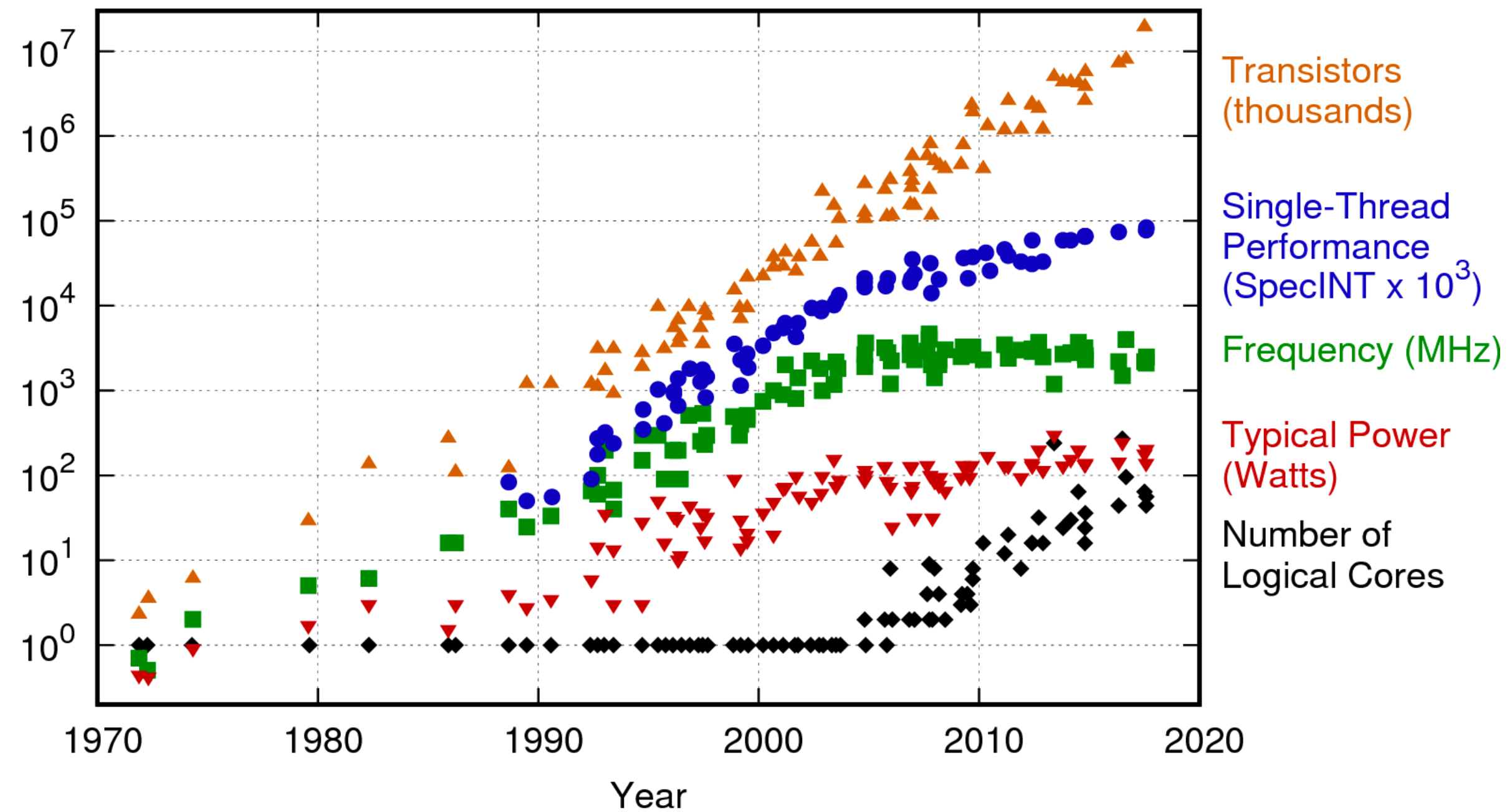
A glimpse into the future



We will collect 3x more data in Run-3, O(10x) more at the HL-LHC
What will Analysis Look like in that Future?

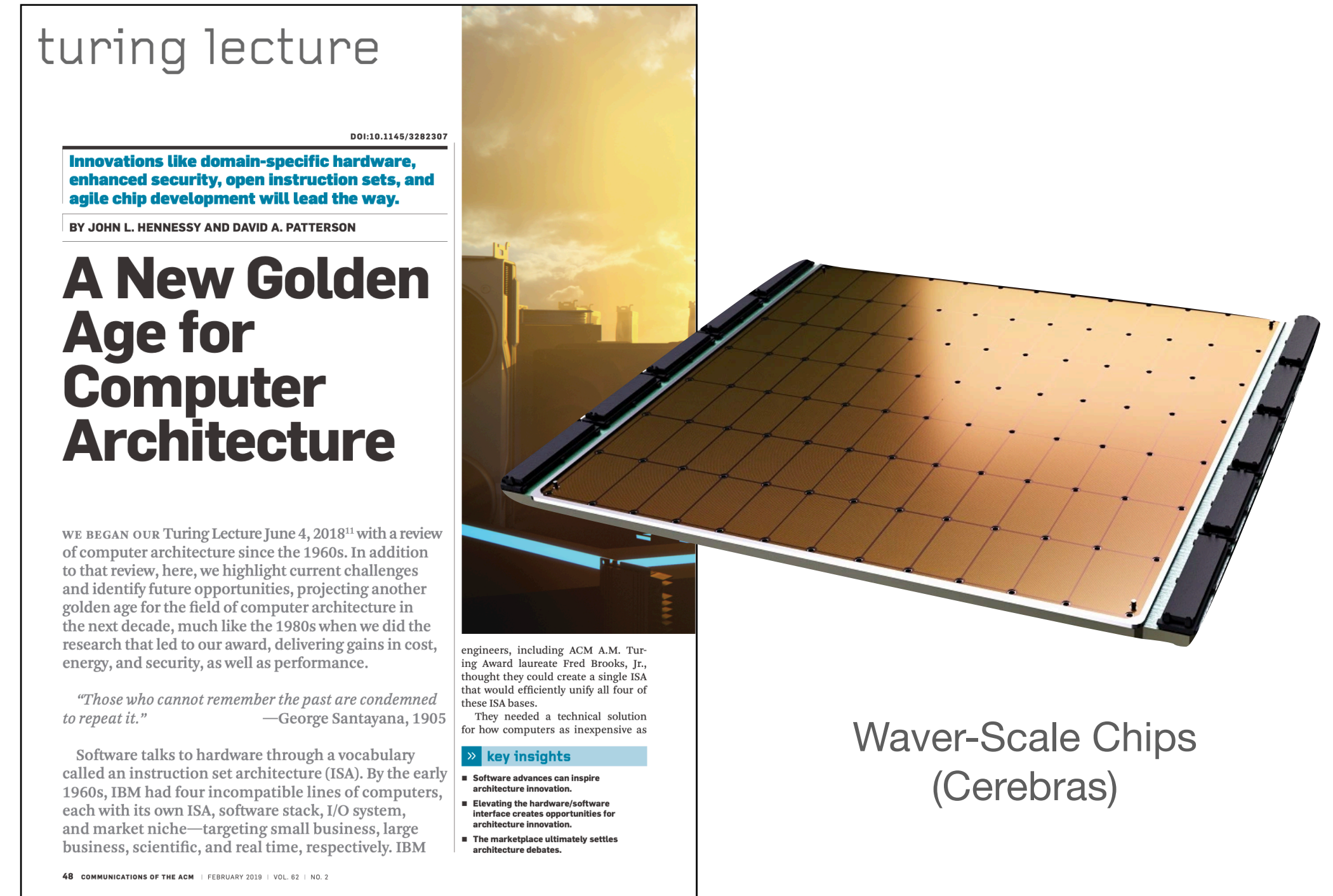
Hardware is changing

42 Years of Microprocessor Trend Data



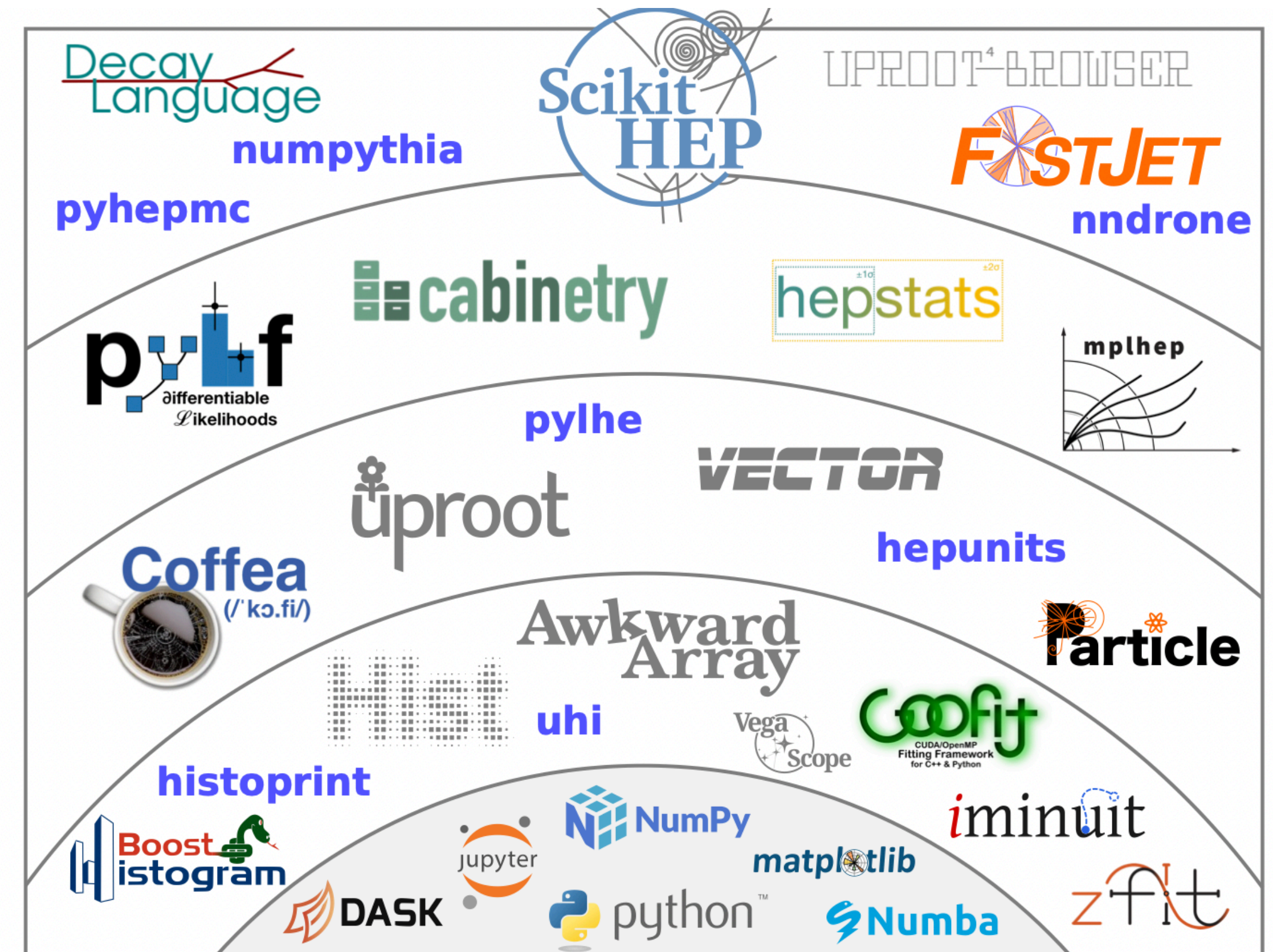
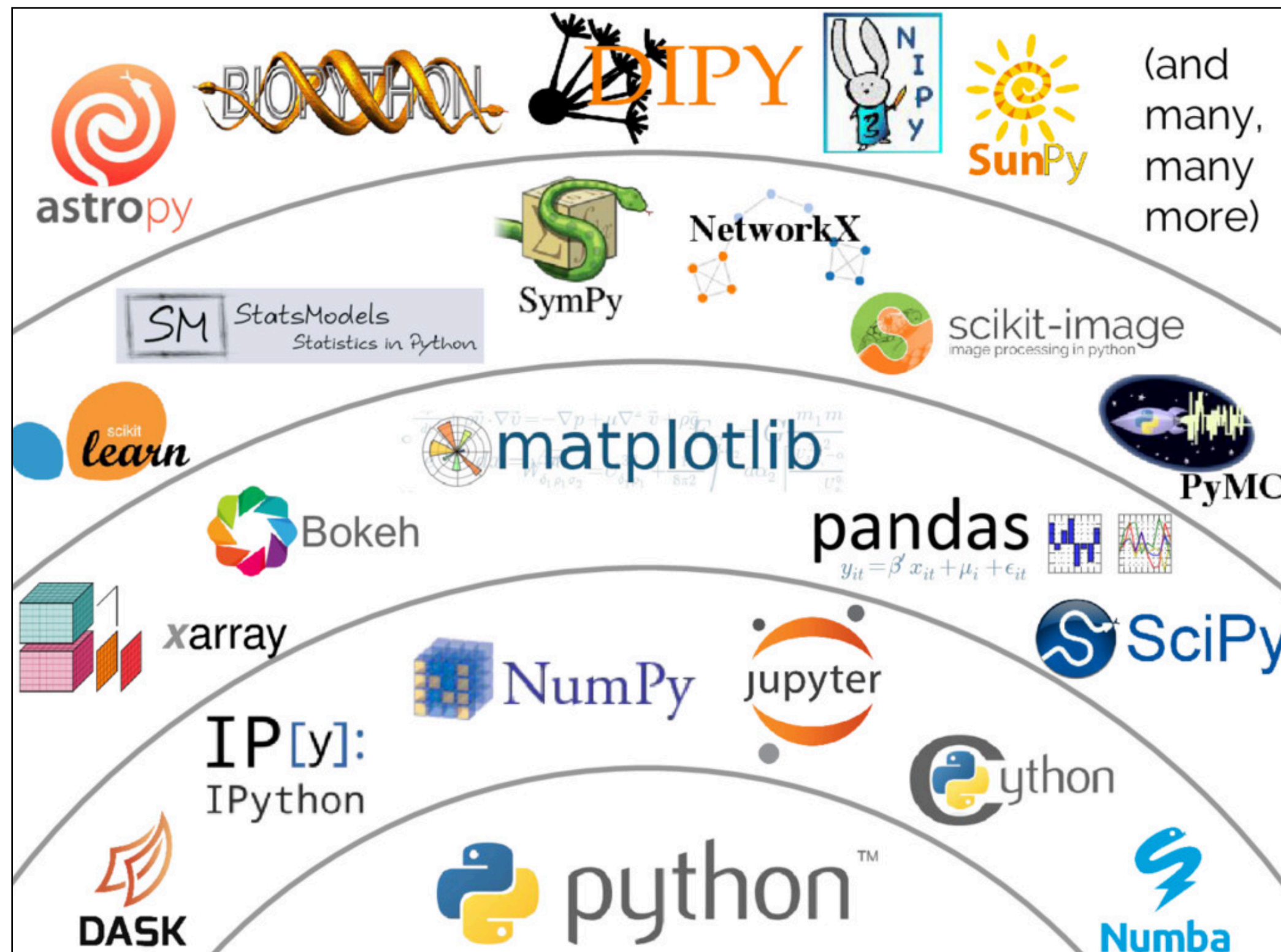
Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten
New plot and data collected for 2010-2017 by K. Rupp

[K. Rupp]



The age of general-purpose computer architectures is ending. Forced towards much more specialized chips. How will we deal with this?

Software is changing



Data Science is much bigger than HNEP. Huge interest to use common tools for interoperability (the “I” in FAIR) & sustainability
Python currently dominant in that space.

Software is changing, maybe fundamentally

... the *other* breakthrough event that happened in 2012

Deep Learning takes over ML thanks to big datasets and GPUs

ImageNet Classification with Deep Convolutional Neural Networks

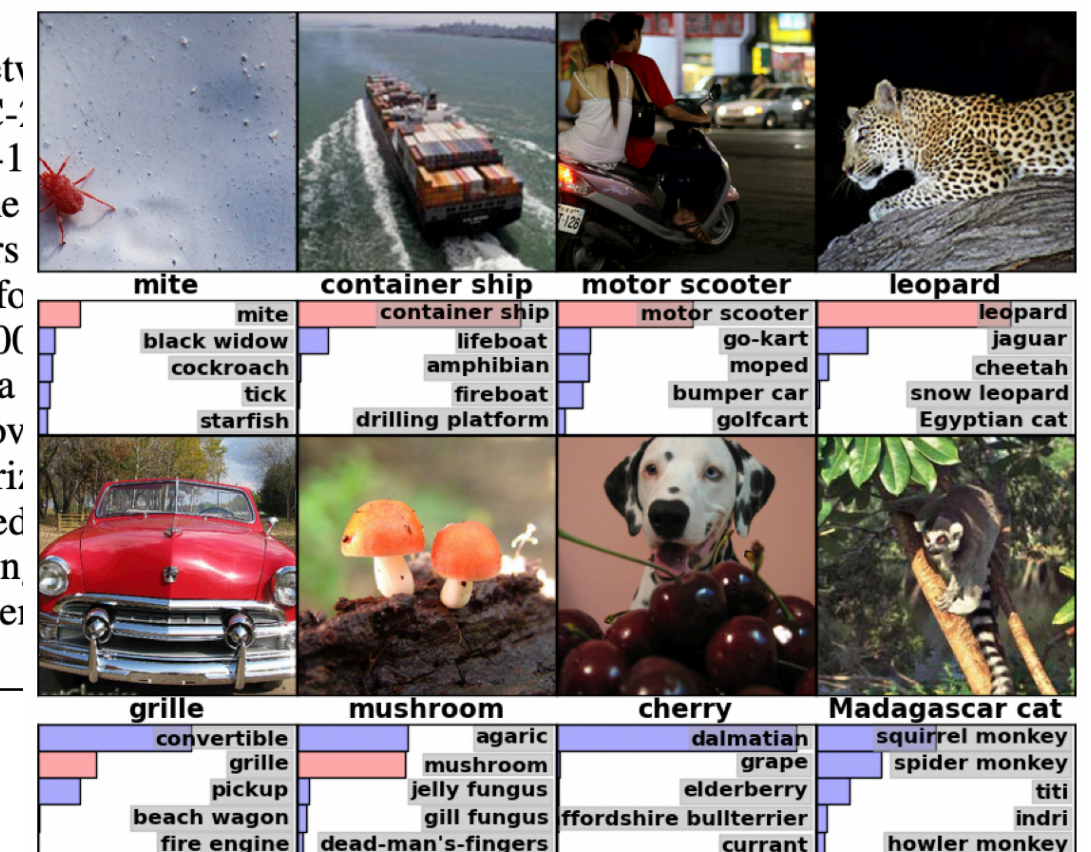
Alex Krizhevsky
University of Toronto
kriz@cs.utoronto.ca

Ilya Sutskever
University of Toronto
ilya@cs.utoronto.ca

Geoffrey E. Hinton
University of Toronto
hinton@cs.utoronto.ca

Abstract

We trained a large, deep convolutional neural network on high-resolution images in the ImageNet LSVRC-2011 dataset. On the test data, we achieved top-1 and top-5 error rates of 37.5% and 17.0% which is considerably better than the previous state-of-the-art. The network, which has 60 million parameters of five convolutional layers, some of which are fully-connected, and three fully-connected layers with a final 1000-unit softmax. To speed up training, we used non-saturating neurons and a modification of the convolution operation. To reduce overfitting, we employed a recently-developed regularization technique that proved to be very effective. We also entered the ILSVRC-2012 competition and achieved a winning performance, compared to 26.2% achieved by the second-best entry.



ML as a **new way to write software** by providing data rather than code
What is the “software development” workflow w/o source code?

Infrastructure is changing

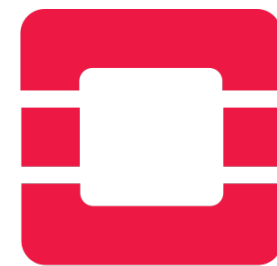
We're not alone in global, federated computation anymore



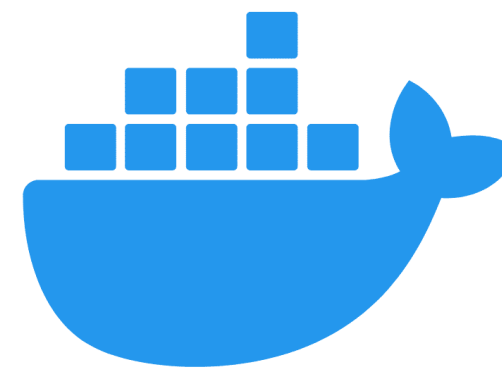
Google Cloud



Alongside by very successful open source software projects



openstack®



docker®



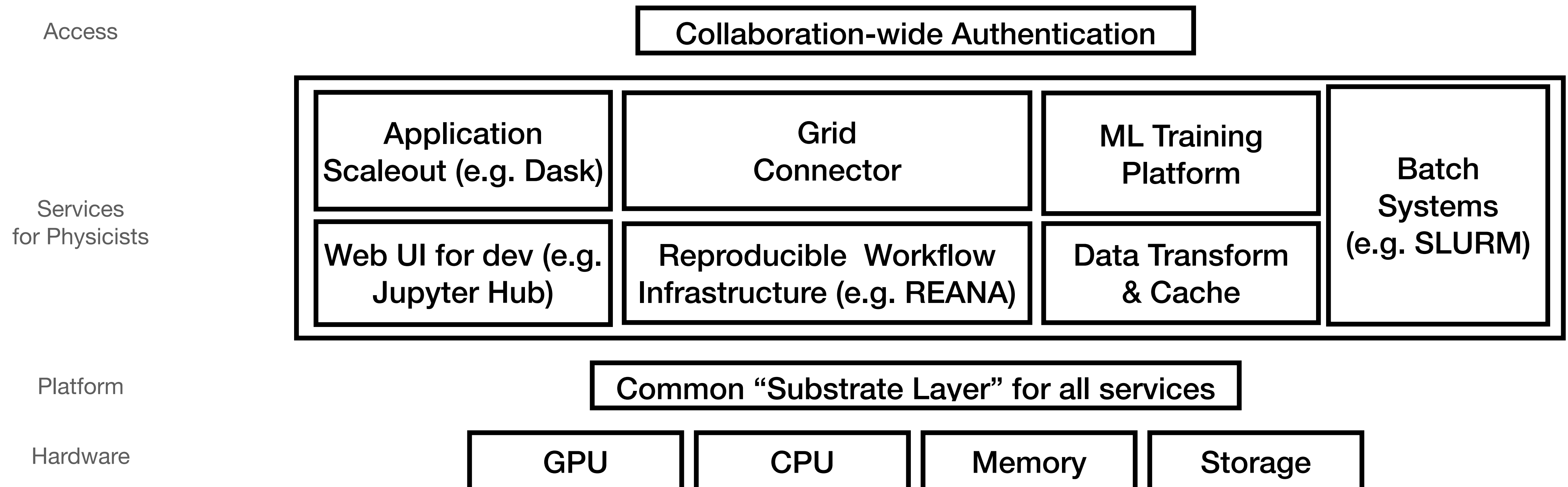
kubernetes

Can we use this in a way that's favourable to us?

Analysis Facilities, next-Gen Grid Sites?

In light of these, it's clear that infrastructure in HEP & other fundamental sciences will shift. A possible wish-list:

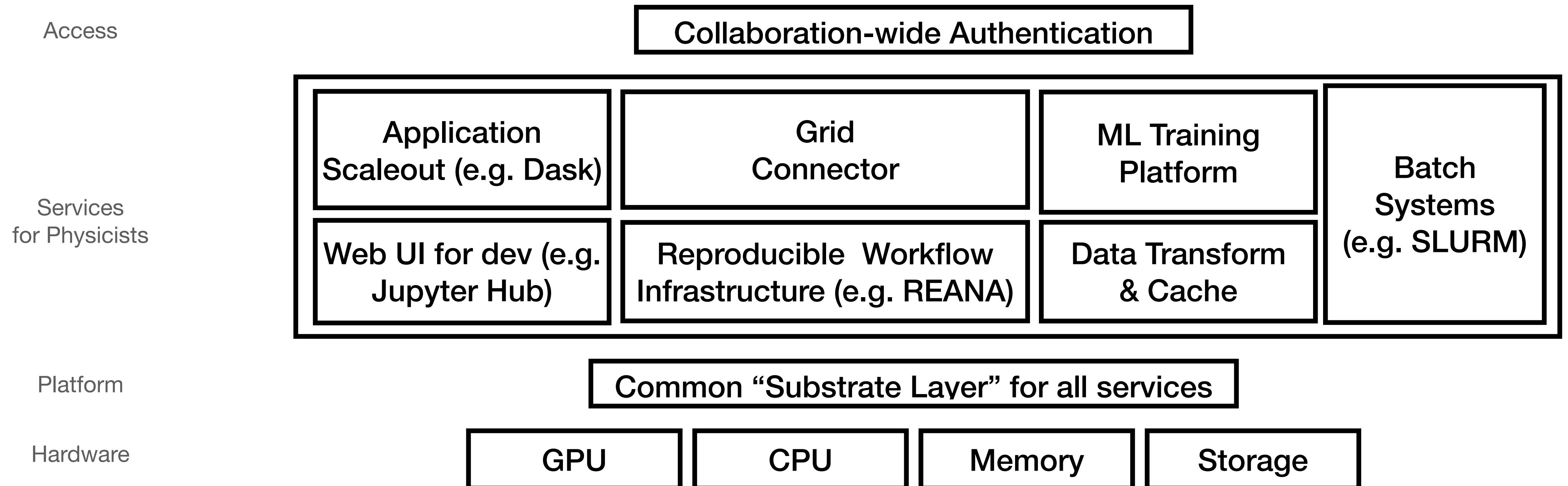
- is this deployable at scale? reproducibility at infrastructure level?



ATLAS-Google Cloud Project

Cloud Advantage: can test before committing large in-house resources

- collaboration between ATLAS & Google to explore what new infrastructure could look like



Cloud Characteristics

Important to keep in mind key differences to on-premise computing

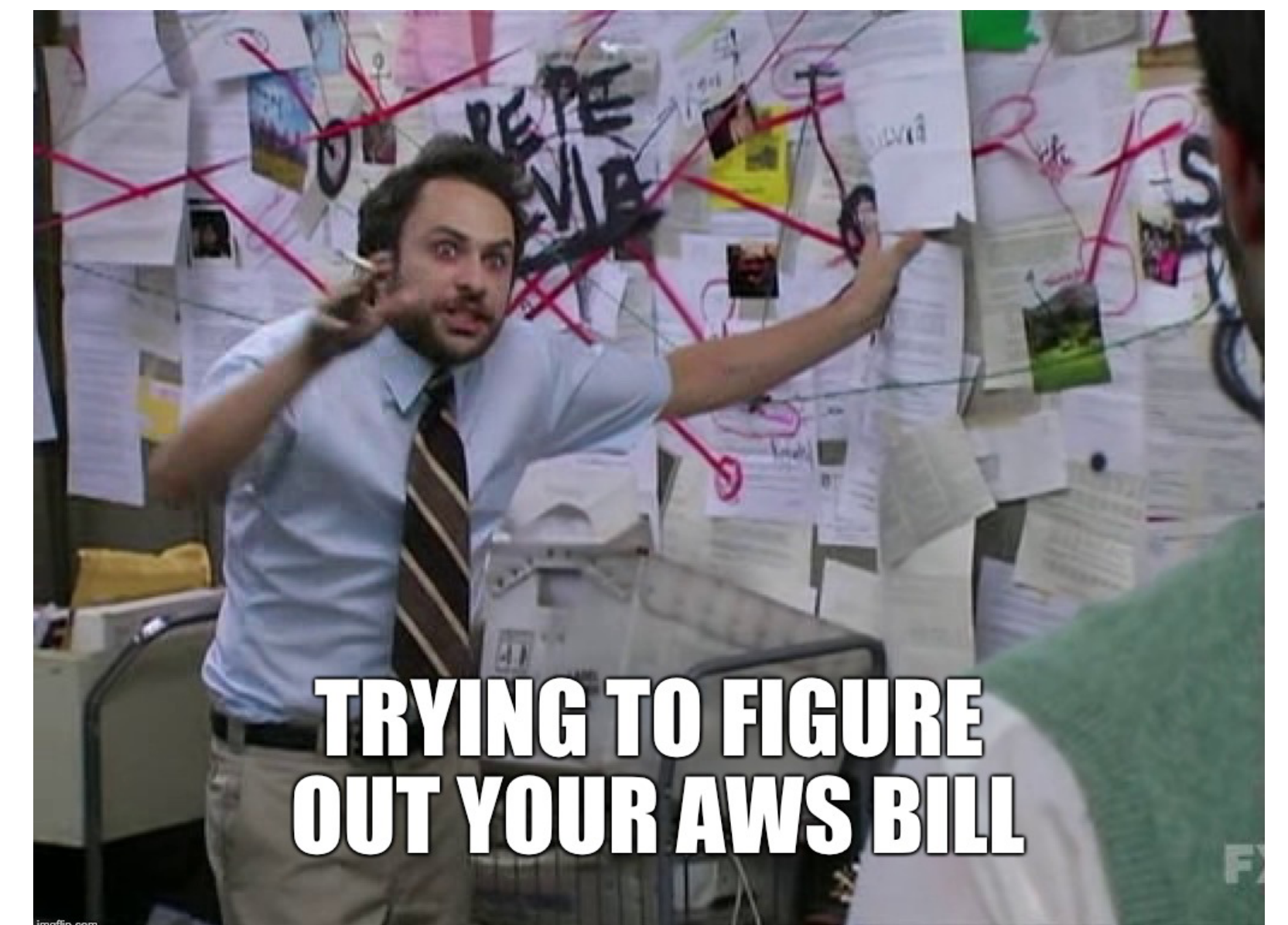
- cloud is very elastic: 50k cores for 1h == 1k cores for 50h
- esp. useful for applications where wall clock time important (e.g. interactive analysis, sequential algorithms, ...)



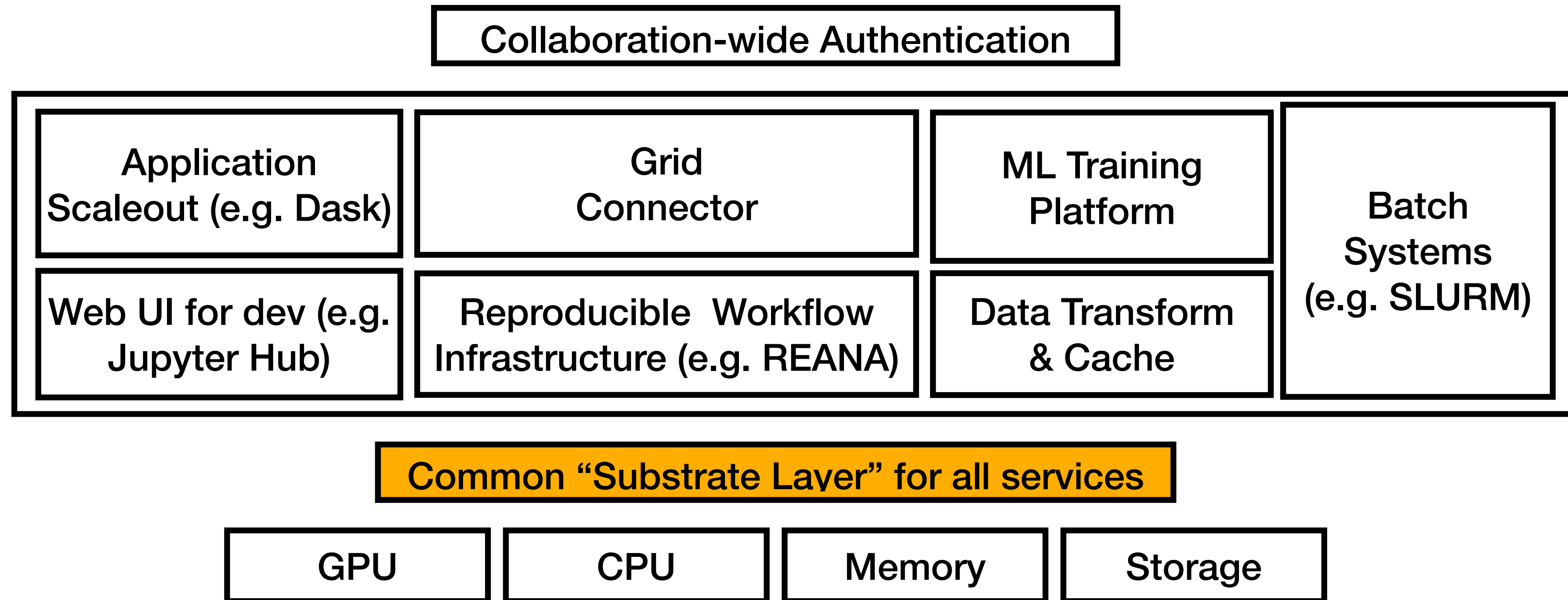
Cloud Characteristics

Important to keep in mind key differences to on-premise computing

- cloud is (can be) very expensive if you don't watch out
- use elasticity to scale on-demand, minimize data transfer
- avoid proprietary offerings to avoid lock in - stay open source



Taking Steps toward the Vision



Platform Choice

Kubernetes is likely the best choice for new infrastructure projects.

- “Lingua Franca” of the cloud, transferable to our own infrastructure
- Target platform for many new projects starting today
- Similar to how we settled on Linux for the original Grid?



CJ 29 Jun 2021

Coffea-casa: an analysis facility prototype
Matous Adamec¹, Garhan Attebury¹, Kenneth Bloom¹, Brian Bockelman², Carl Lundstedt¹, Oksana Shadura¹, and John...
¹University of Nebraska-Lincoln
²Morgridge Institute for Research

ScienceBox
Converging to Kubernetes containers in production for on-premise and hybrid clouds for CERNBox, SWAN, and EOS

(I6.1) Why is the LSST Science Platform built on Kubernetes?

Christine Banek,¹ Adam Thornton,¹ Frossie Economou,¹ Angelo Fausti,¹ K. Simon Krughoff,¹ and Jonathan Sick¹

¹AURA/LSST, Tucson, AZ, USA; cbanek@lsst.org

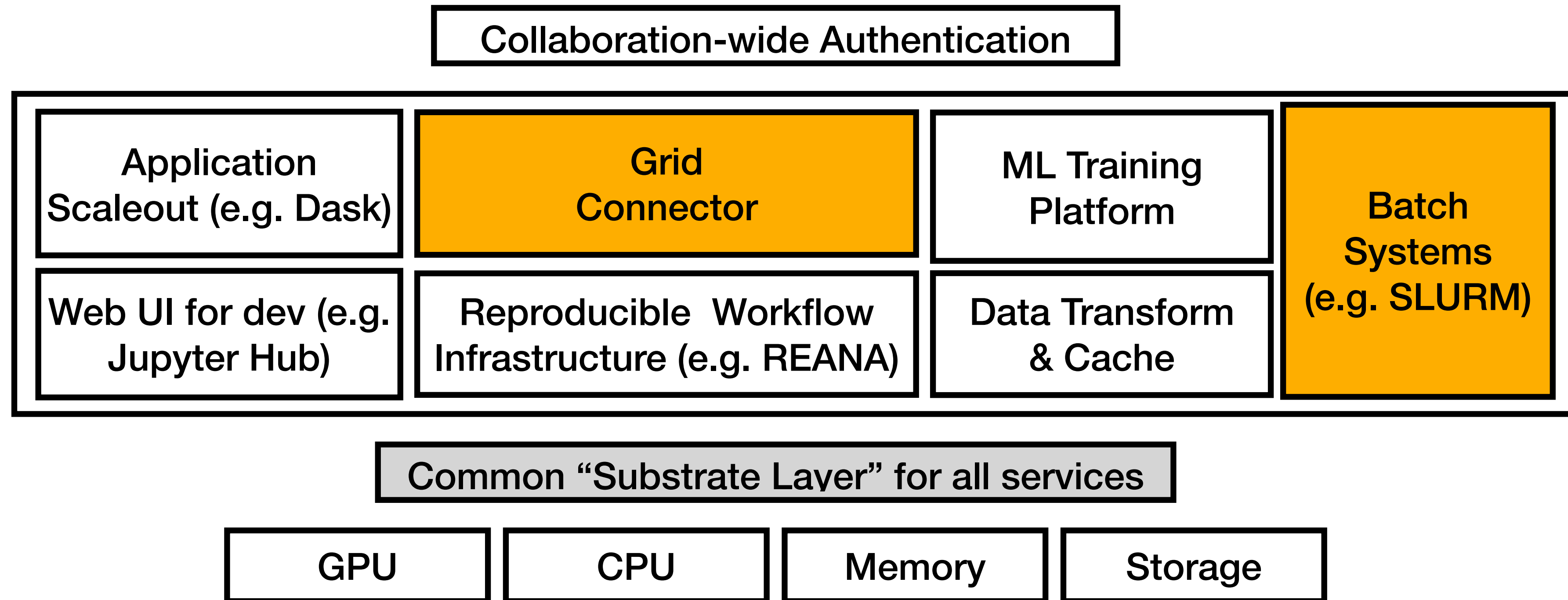
Abstract. LSST has chosen Kubernetes as the platform for deploying and operating the LSST Science Platform. We first present the background reasoning behind this decision, including both instrument-agnostic as well as LSST-specific requirements. We then discuss the basic principles of Kubernetes and Helm, and how they are used as the deployment base for the LSST Science Platform. Furthermore, we provide an

10.1051/epjconf/202024507047

Gonzalez

Poland

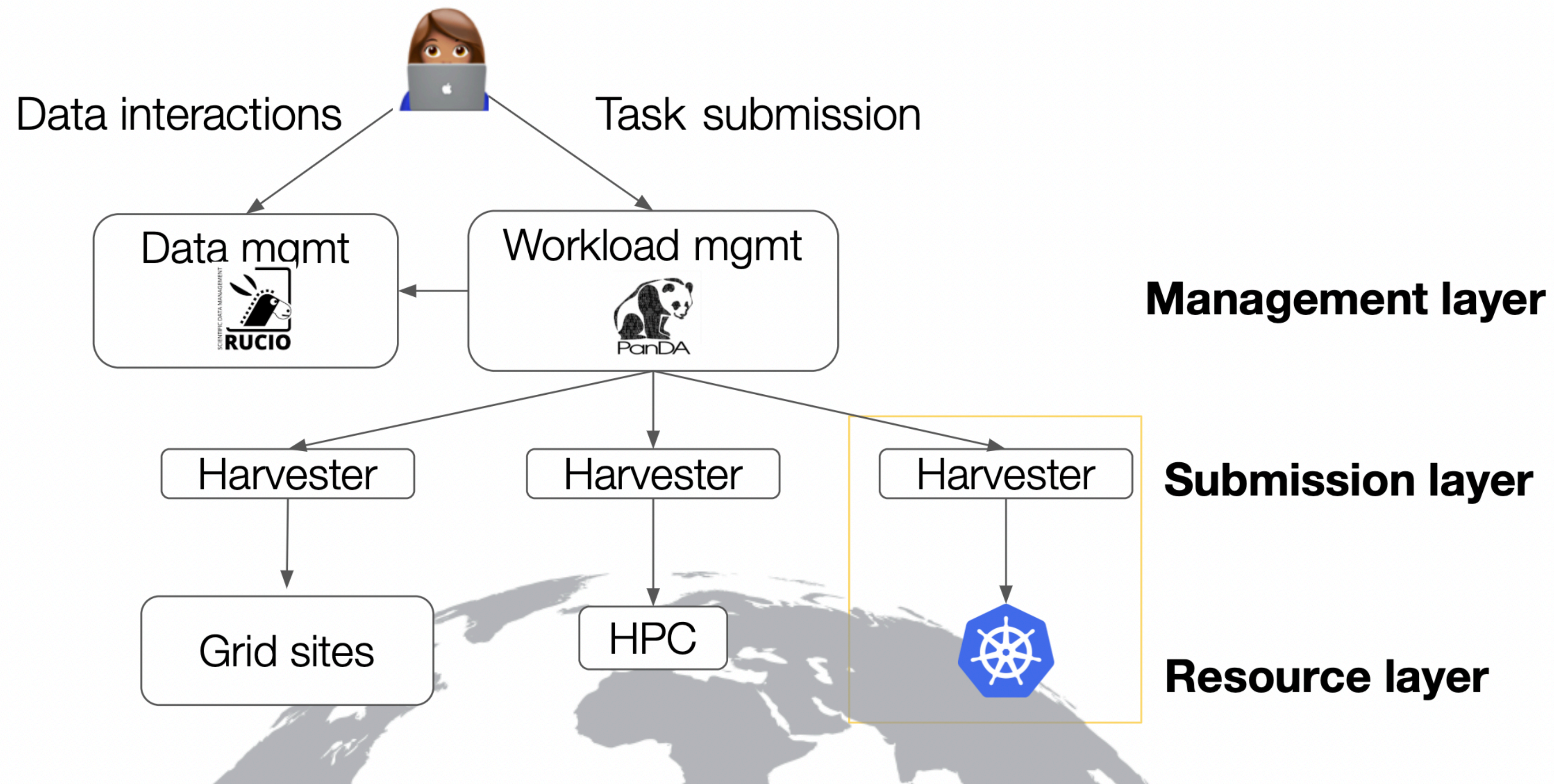
Bringing the Grid to the Cloud



First things first: Grid Sites

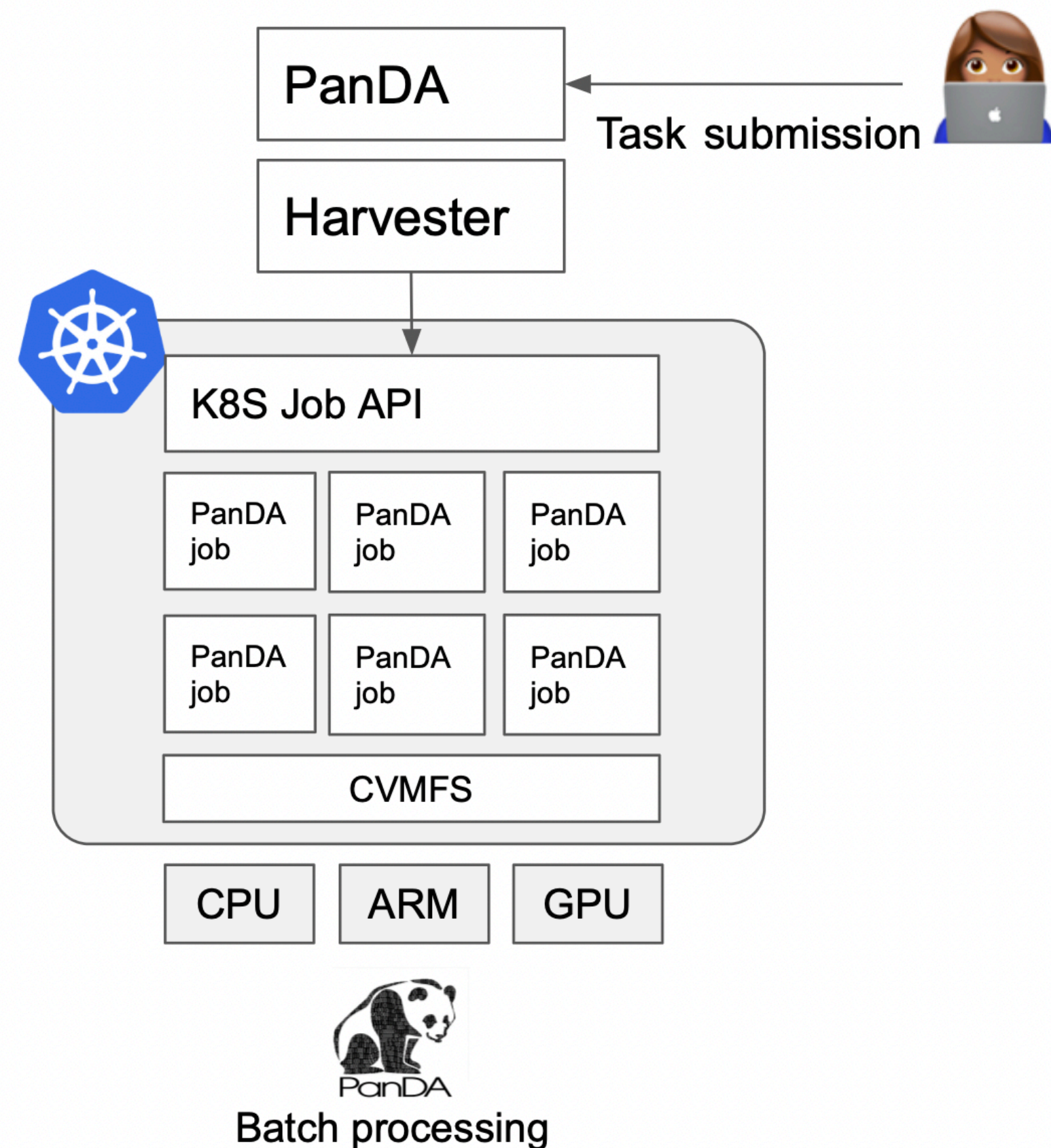
Show we can deploy our existing tools on new infrastructure

Advantage: we were always set up to integrate new systems.



First things first: Grid Sites

Kubernetes as just another resource, receive grid jobs & run them

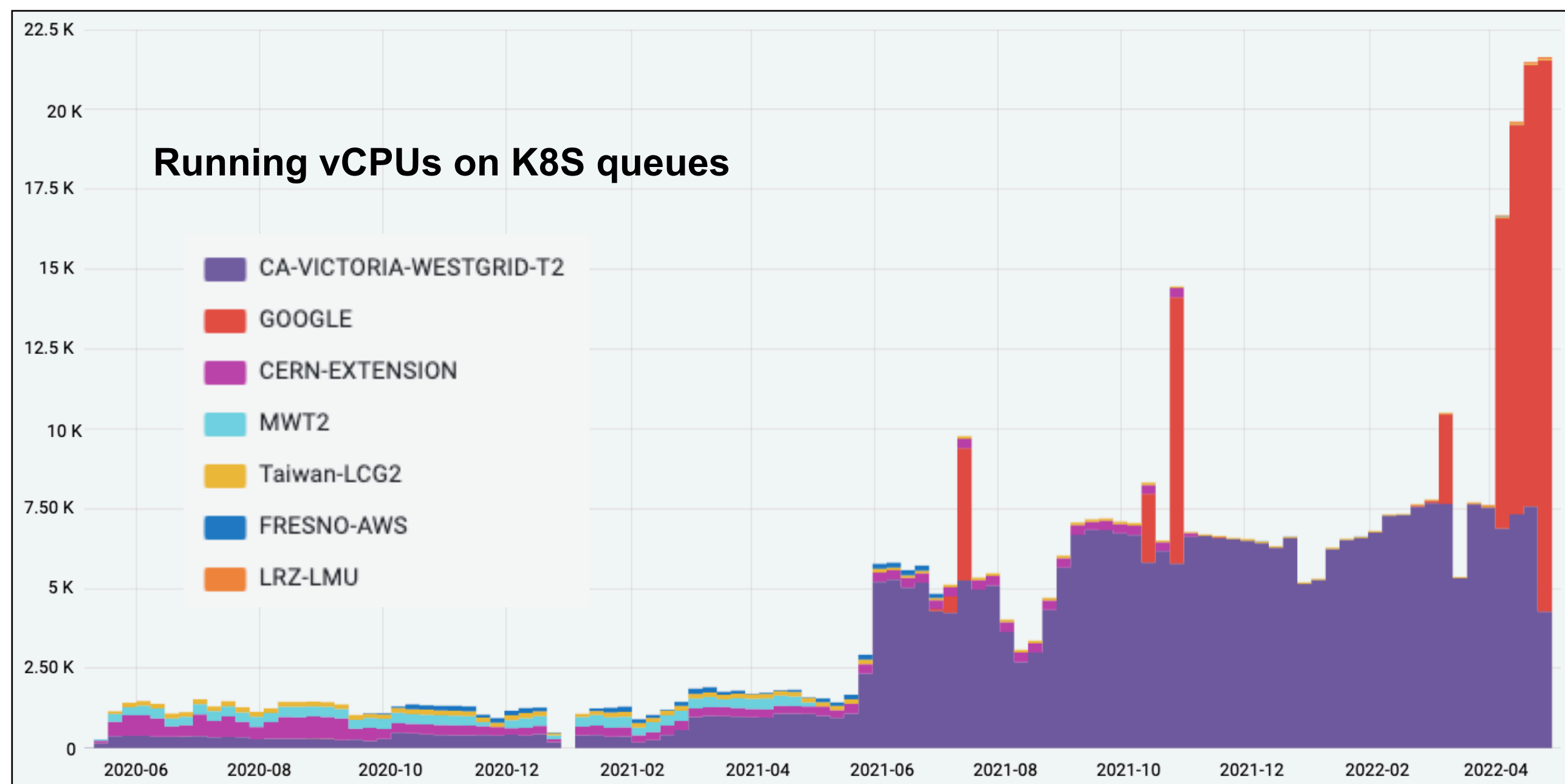


- no external batch system
- dynamic job placement across nodes to maximize utilization
- in cloud it's easy to add heterogeneous resources such as GPU or ARM

First things first: Grid Sites

Successfully running in production O(10k cores) across multiple institute cloud **and** clusters with the same Kubernetes-based setup.

Growing fraction of the grid. How far can we go with a single cluster?



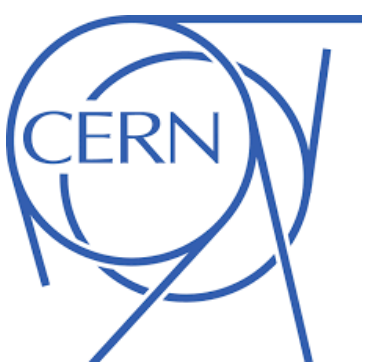
FRESNO STATE



Leibniz-Rechenzentrum
der Bayerischen Akademie der Wissenschaften



THE UNIVERSITY OF
CHICAGO



University
of Victoria



Google Cloud

Cloud-specifics

In order to seriously scale & benefit from cloud elasticity a few adjustments to the setup

- Only a single small node is on 24/7
- HEP is embarrassingly parallel
can just resubmit job → use
pre-emptible VMs 5x cheaper
- to scale use “big nodes” e.g.
80 core: less room for failure
(e.g. mounting filesystems)

1 node: 8vCPU, 32 GB RAM, 160 GB standard pdd



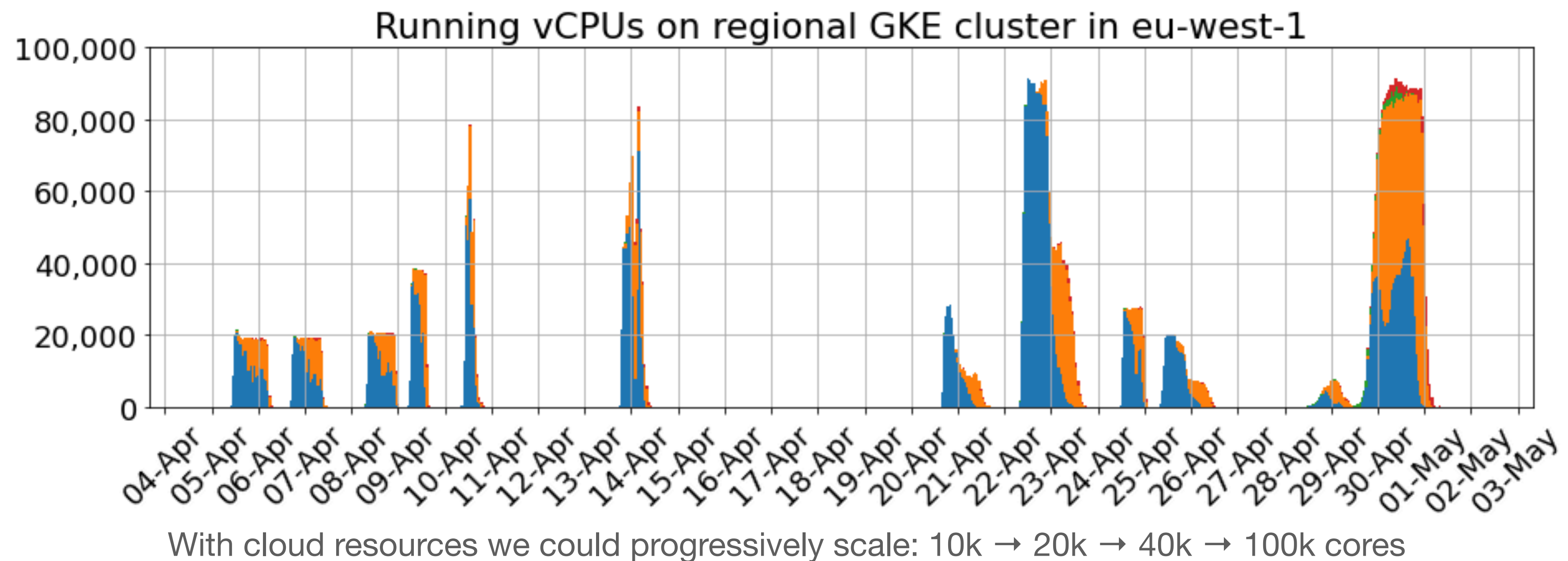
Autoscaled nodes: 80 vCPU, 320 GB RAM, 1.6 TB standard pdd



Scaling Up on the Cloud

With adjustments in place, we can comfortably scale to 100k cores on a single Kubernetes cluster running standard Grid workflows.

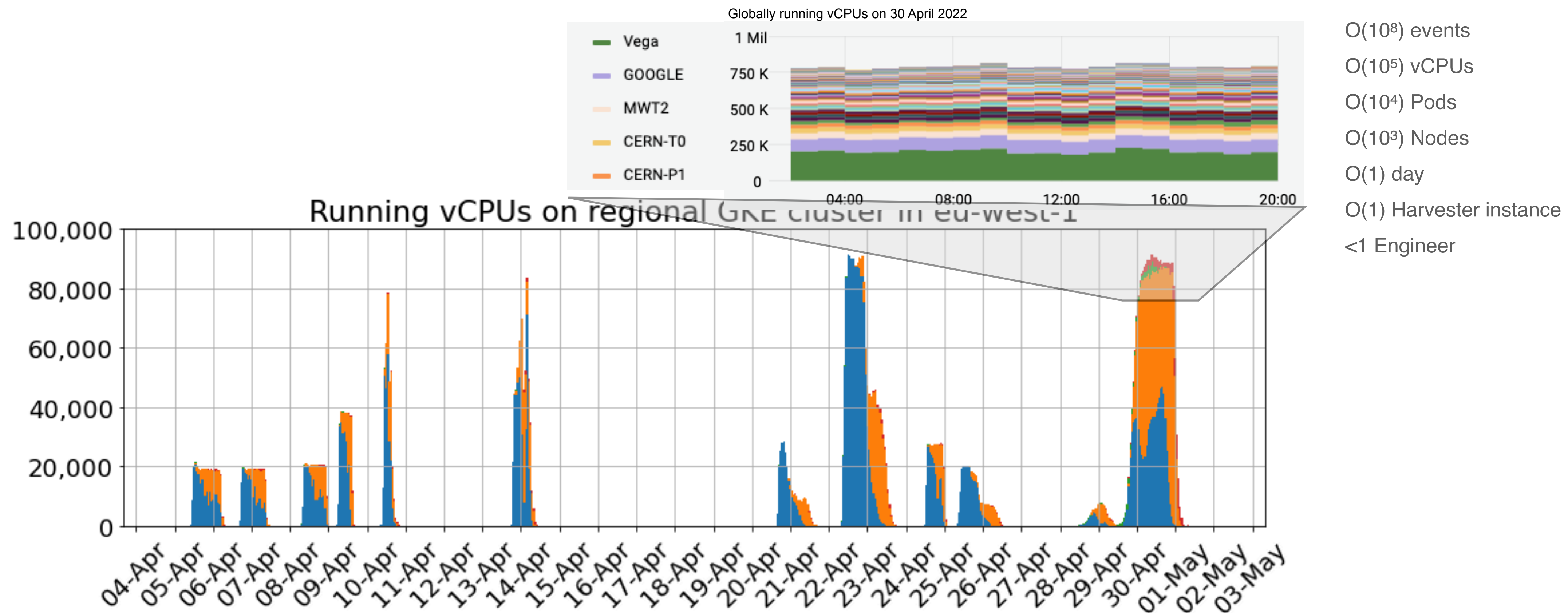
- 10k vs 100k makes little difference
- Only 1% job failure rate even on pre-emptible VMs



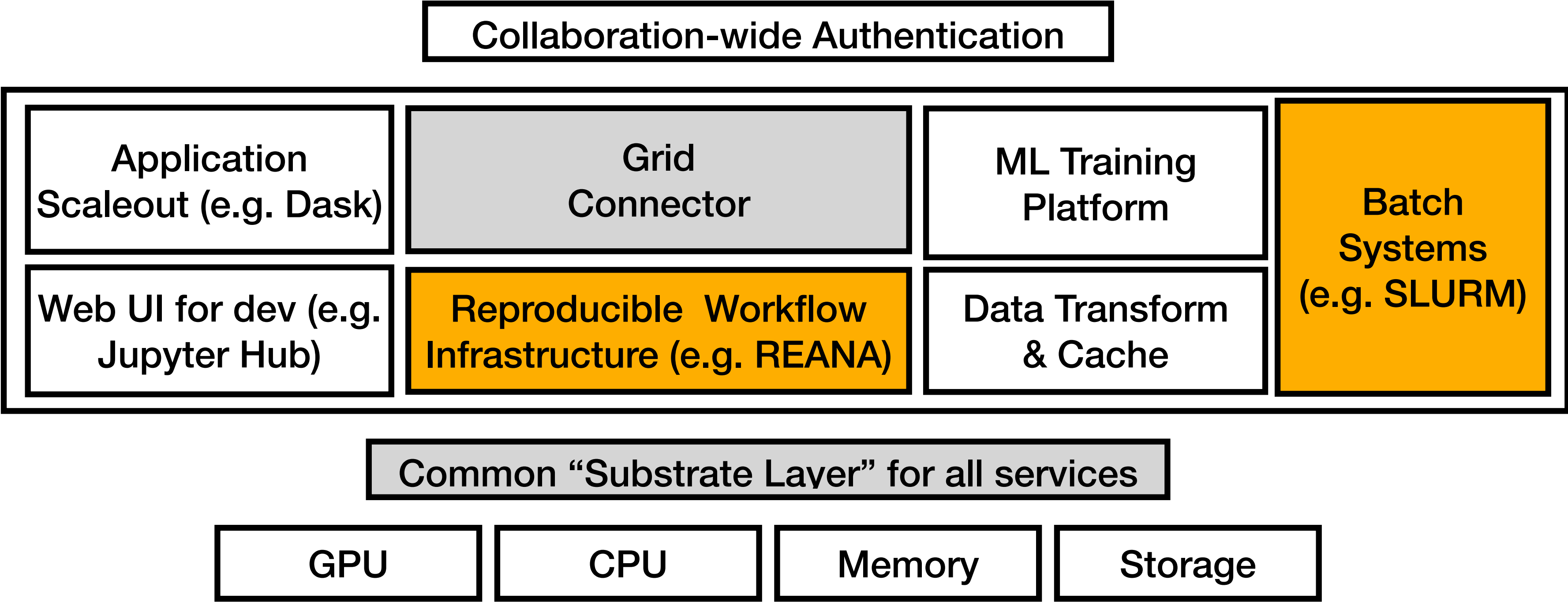
Scaling Up on the Cloud

At 100k cores, a single cluster is a sizable fraction to the overall grid

- should be able to scale more, but started to see instabilities > 100k



Beyond Simple Batch

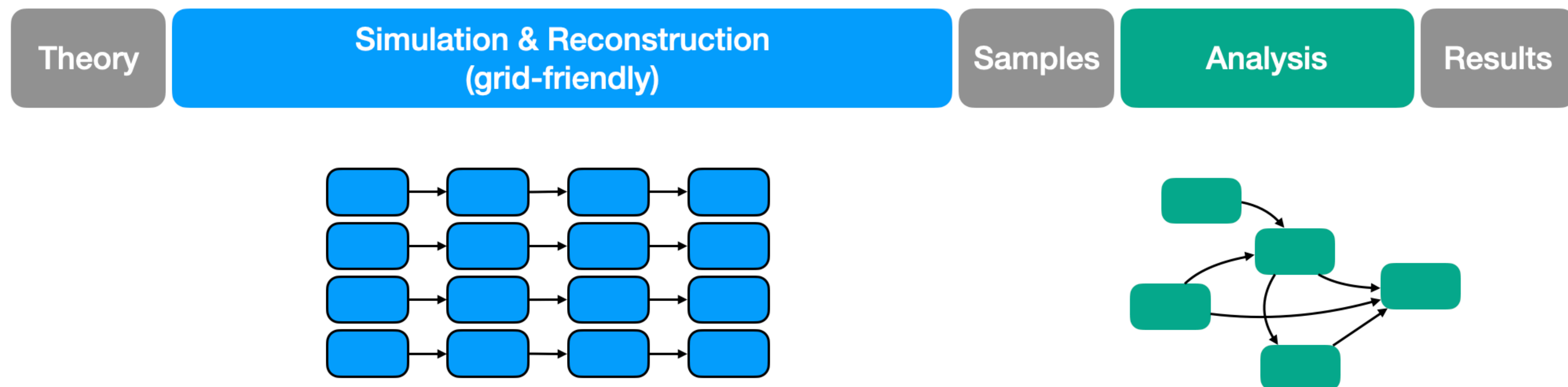


Adding Reproducible Workflows

Grid as it is today very optimized for “collaboration-wide” tasks (simulation and reconstruction) very uniform data processing

User analysis is much more heterogeneous: complex workflows, each analysis unique: *can we map this onto the same infrastructure?*

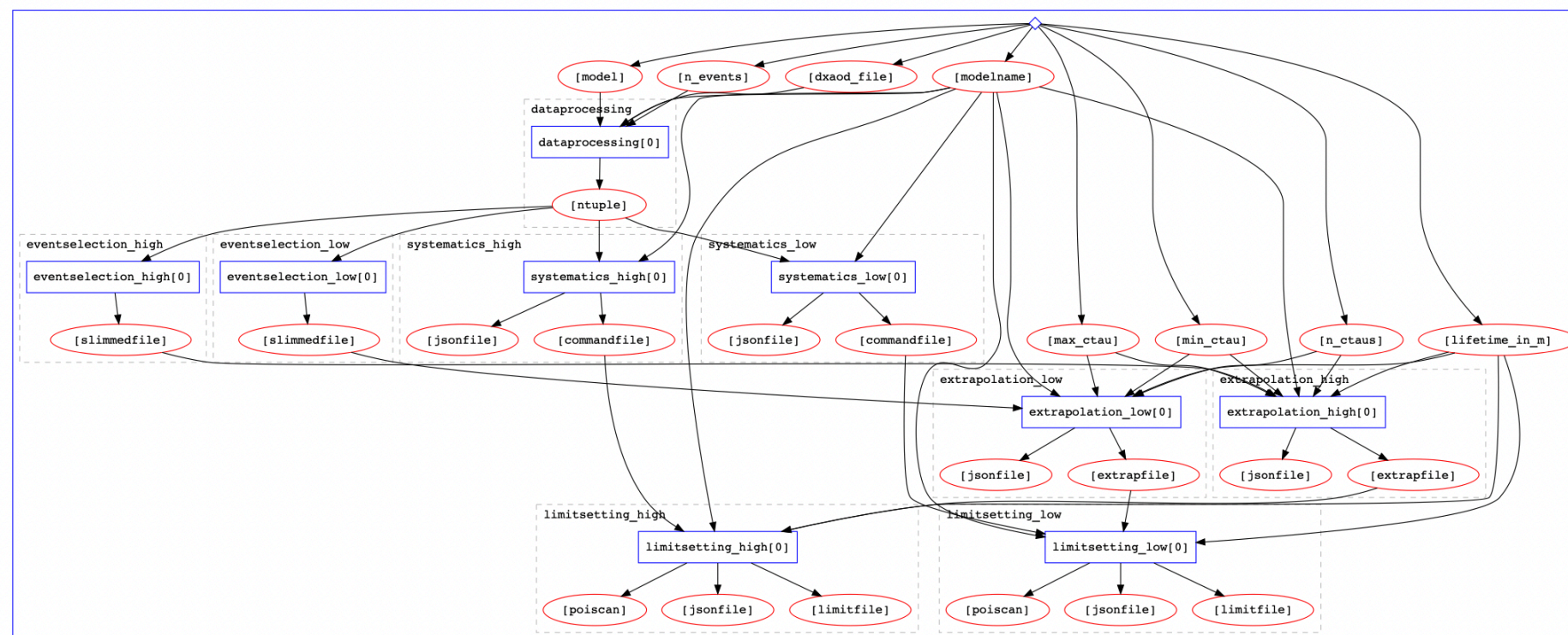
- opportunity to expand what we mean by “the grid”



REANA

Platform for reusable (the “R” in FAIR) analysis workflows on cloud infrastructure.

Idea: describe analysis as a workflow of containerized tasks, let workflow system handle specifics of submission



Many workflow systems in use, REANA aims to support many of them on a unified level

Home Examples Get Started Documentation News Roadmap Contact Blog



Reproducible research data analysis platform

Flexible

Run many computational workflow engines.



Scalable

Support for remote compute clouds.



Reusable

Containerise once, reuse elsewhere. Cloud-native.



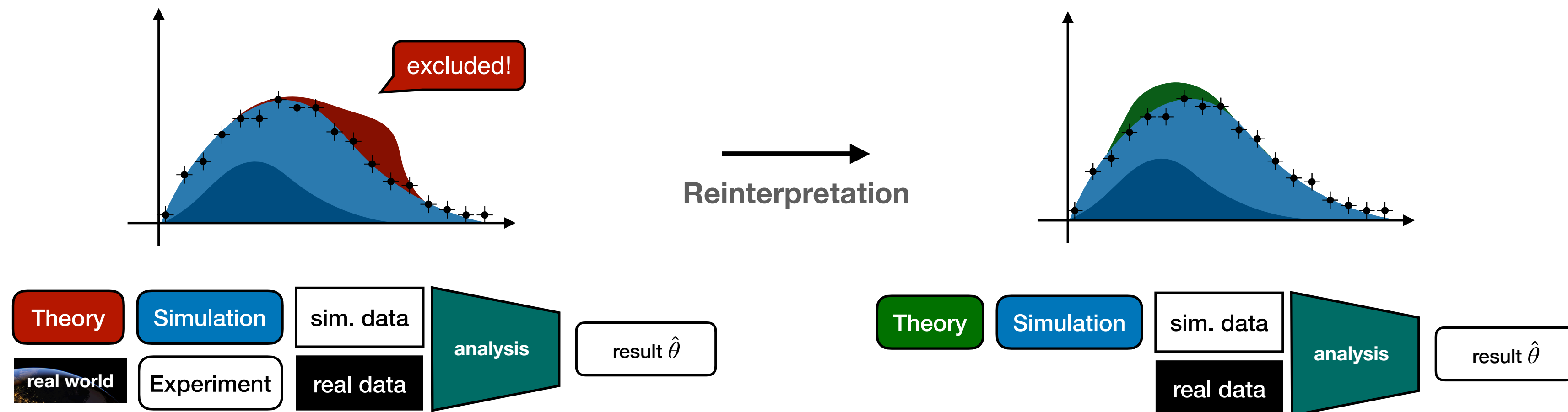
Free

Free Software. MIT licence. Made with ❤️ at CERN.



Key Use Case: RECAST

Idea: most analyses are sensitive to more physics than they originally targeted. To get the most physics out, we should “reuse” them:

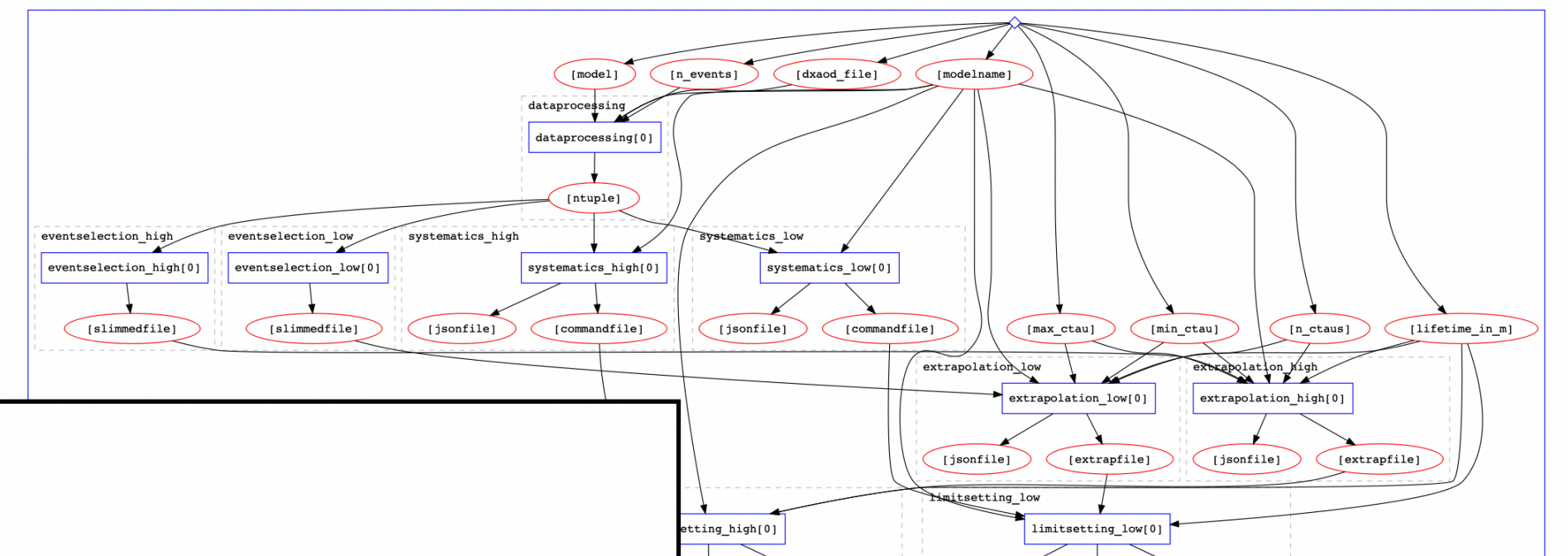


For this we need ability to

- preserve analyses as re-suable objects → containerized workflows
- catalogue them (i.e. findable - the “F” in FAIR)
- re-run them on common infrastructure

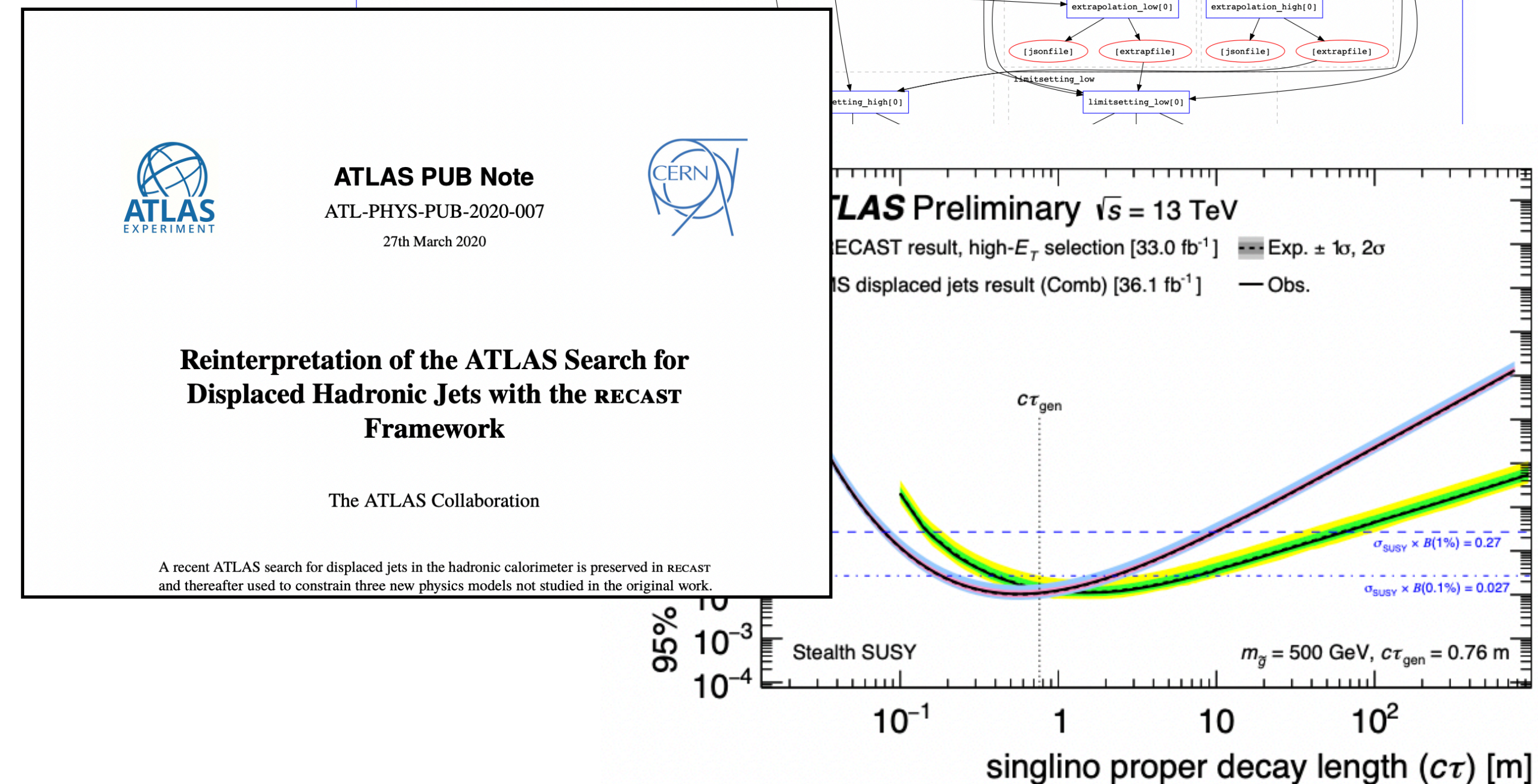
Examples of Reinterpretations

Starting to get new physics results from these efforts for “hand-picked” analyses and individual new theories.



Key question: can we scale this to a general paradigm?

- e.g. study high-dimensional SUSY parameter space such as pMSSM

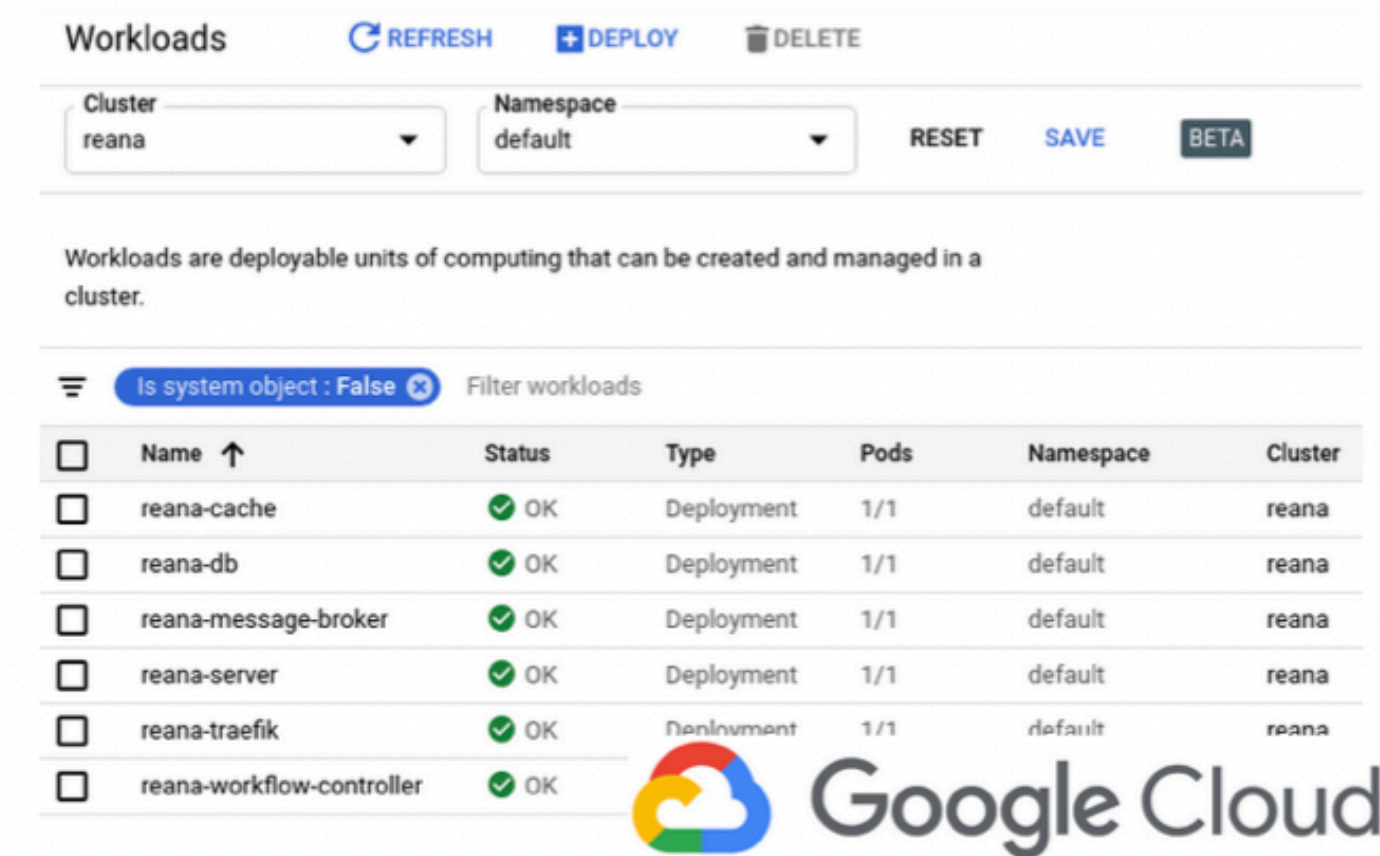


Services as Kubernetes Applications

From a Kubernetes point-of-view complex services like REANA are just another “application”: easy to deploy

```
helm repo add reanahub https://reanahub.github.io/reana  
helm install --devel reana reanahub/reana --wait
```

Same installation at CERN
and Google Cloud thanks to uniformity
of Kubernetes APIs



The screenshot shows the Google Cloud Workloads management interface. At the top, there are buttons for 'REFRESH', 'DEPLOY', and 'DELETE'. Below these are dropdown menus for 'Cluster' (set to 'reana') and 'Namespace' (set to 'default'), along with 'RESET', 'SAVE', and 'BETA' buttons. A text block states: 'Workloads are deployable units of computing that can be created and managed in a cluster.' Below this is a filter bar with 'Is system object : False' and a 'Filter workloads' button. The main table lists the following workloads:

☐	Name ↑	Status	Type	Pods	Namespace	Cluster
☐	reana-cache	✓ OK	Deployment	1/1	default	reana
☐	reana-db	✓ OK	Deployment	1/1	default	reana
☐	reana-message-broker	✓ OK	Deployment	1/1	default	reana
☐	reana-server	✓ OK	Deployment	1/1	default	reana
☐	reana-traefik	✓ OK	Deployment	1/1	default	reana
☐	reana-workflow-controller	✓ OK	Deployment	1/1	default	reana

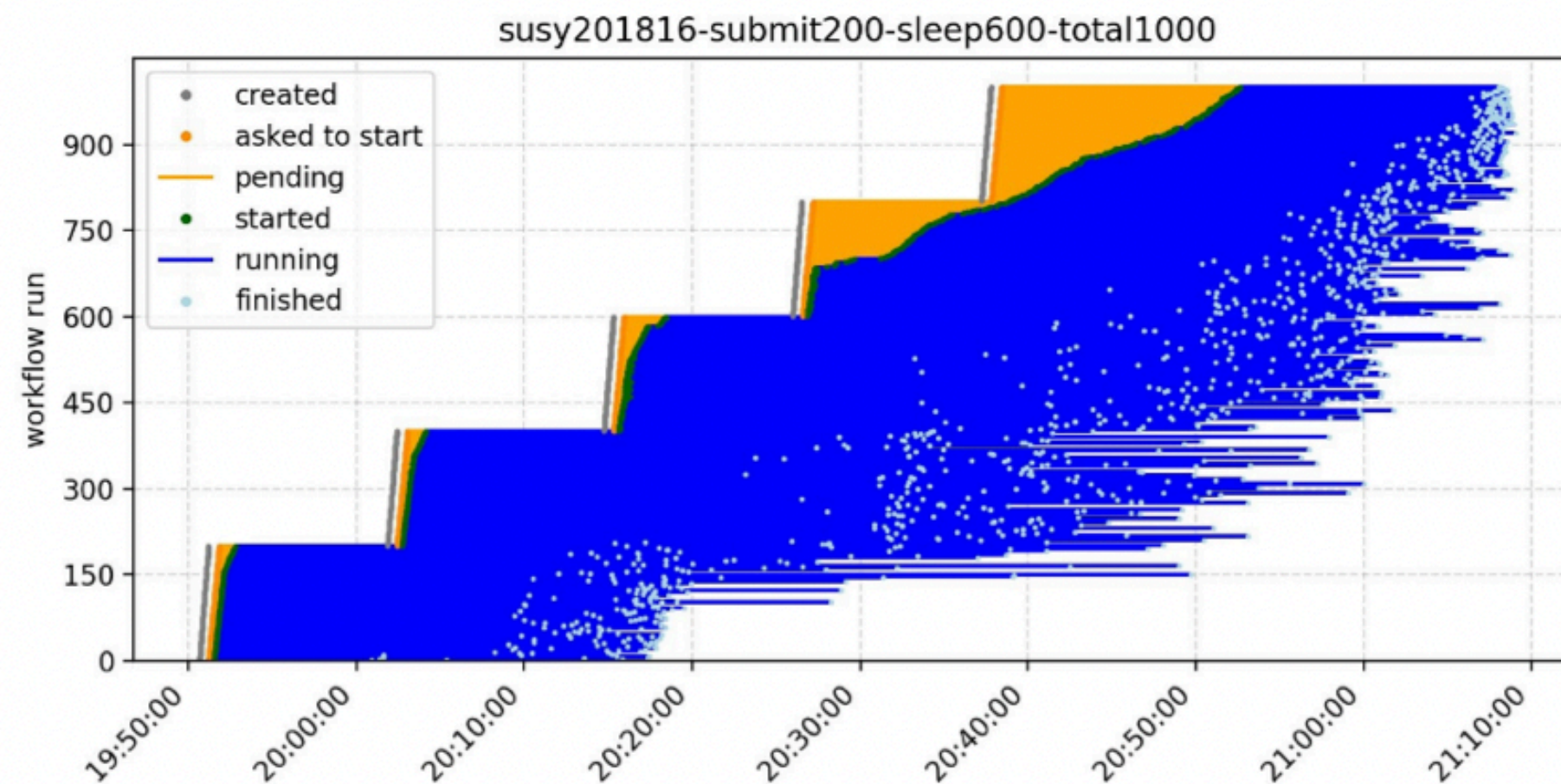
REANA on Google Cloud



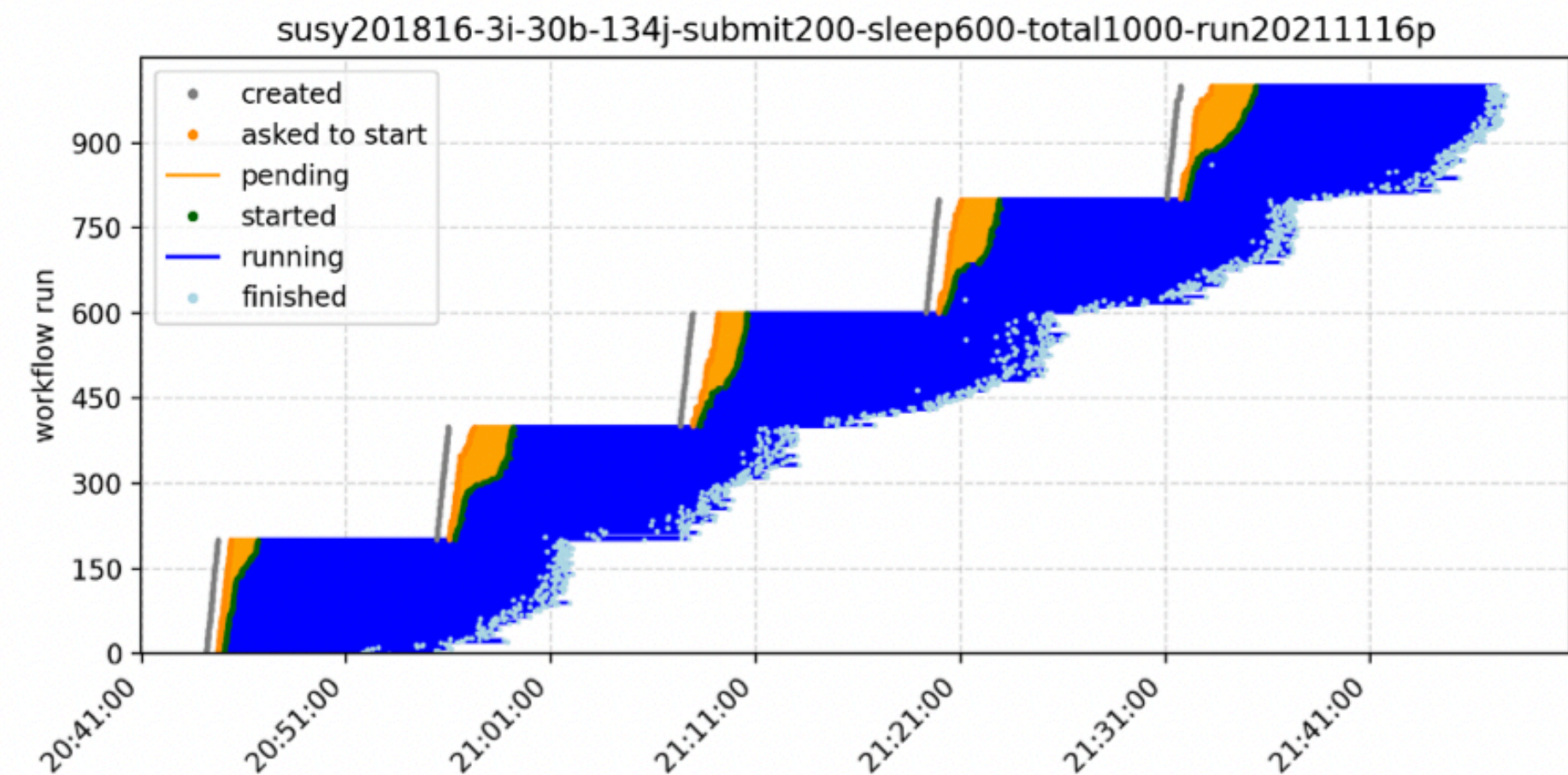
Scaling REANA

To study 19-D pMSSM one would need to run $O(100k)$ analysis runs

- processing many samples, complex fits.. great stress test for REANA
- scaled at CERN to $O(1k)$ cores) looking to push further (a la grid)
- develop basis for production use of REANA inside of collaborations

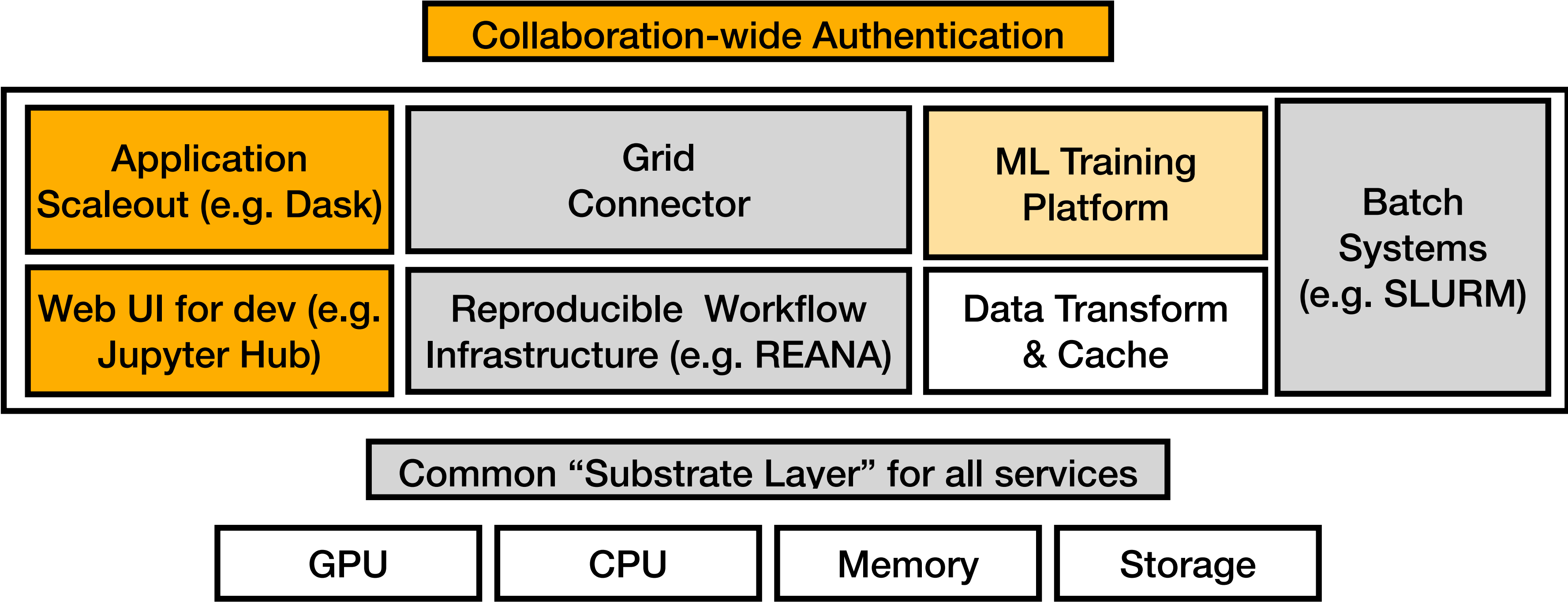


Old cluster (448 cores)



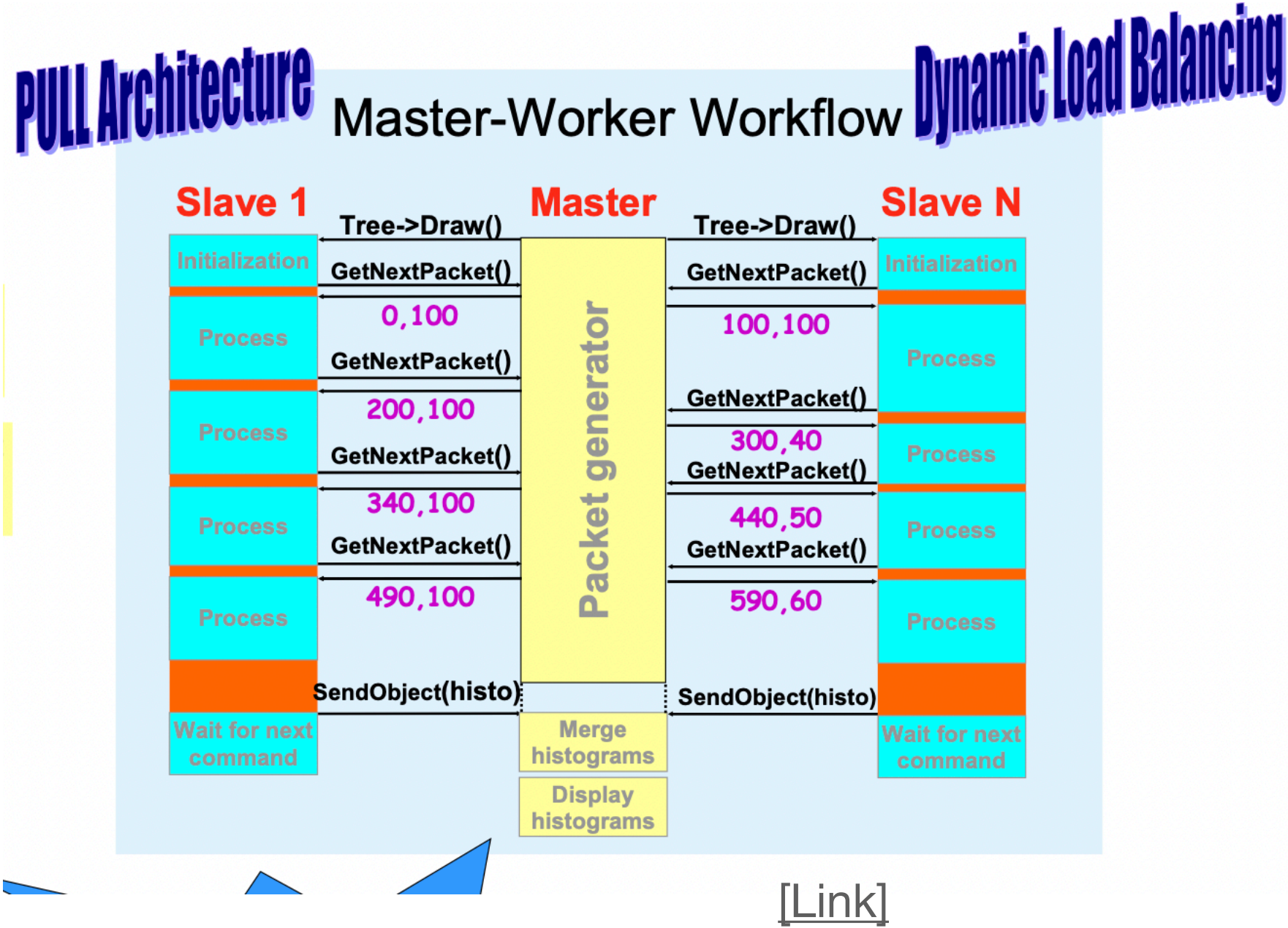
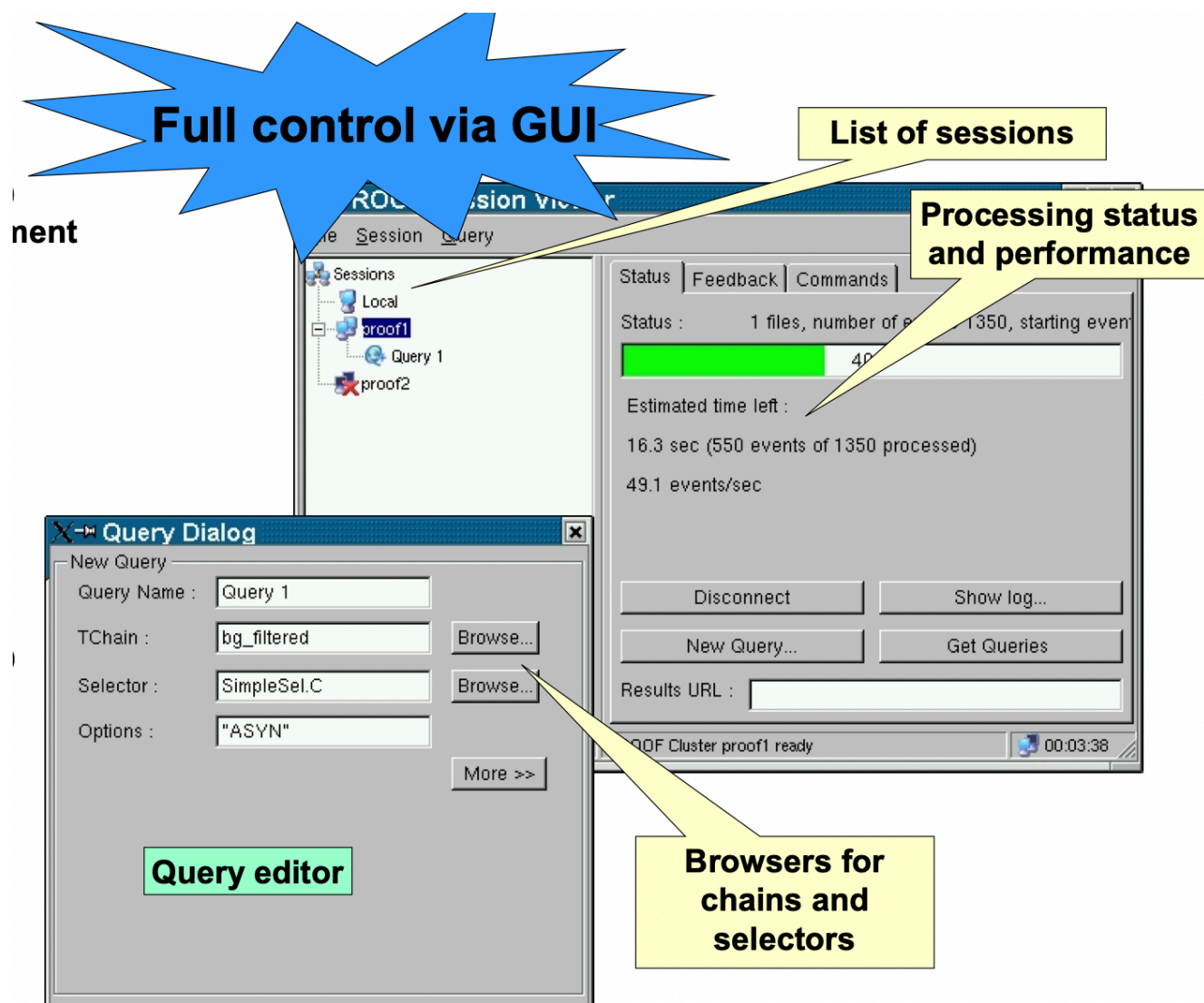
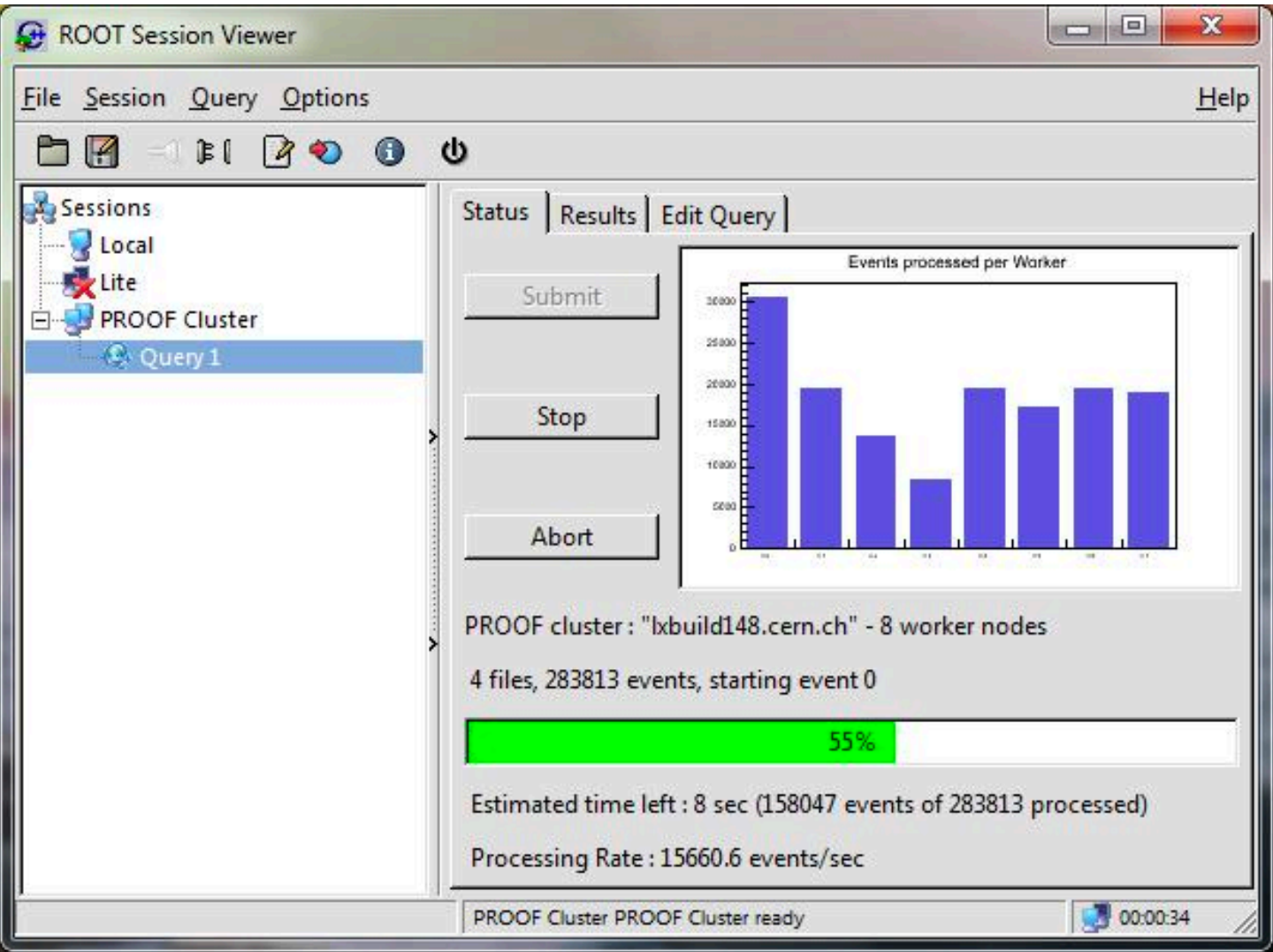
New cluster (1072 cores)

Beyond Simple Batch



Interactive Analysis

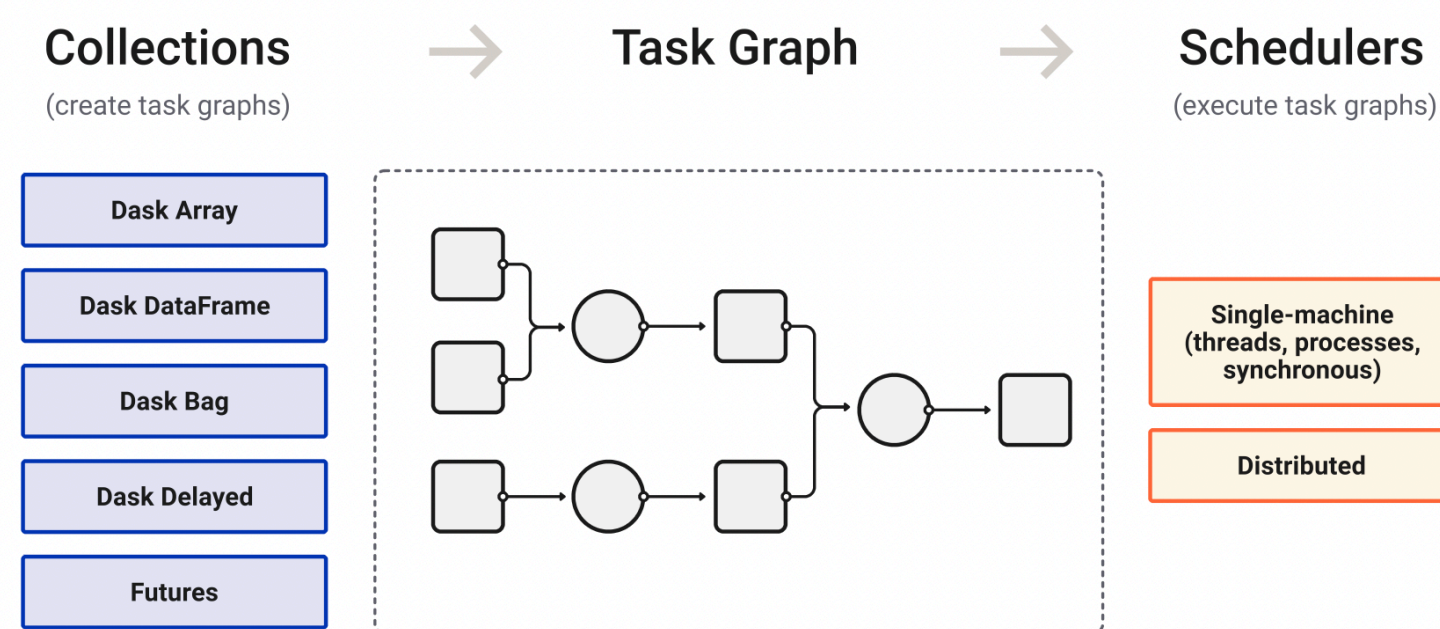
So far, everything was batch (either direct submission or workflows)
But to analysis should be explorative, interactive & fast turnaround at HL-LHC scales. Blast from the Past:



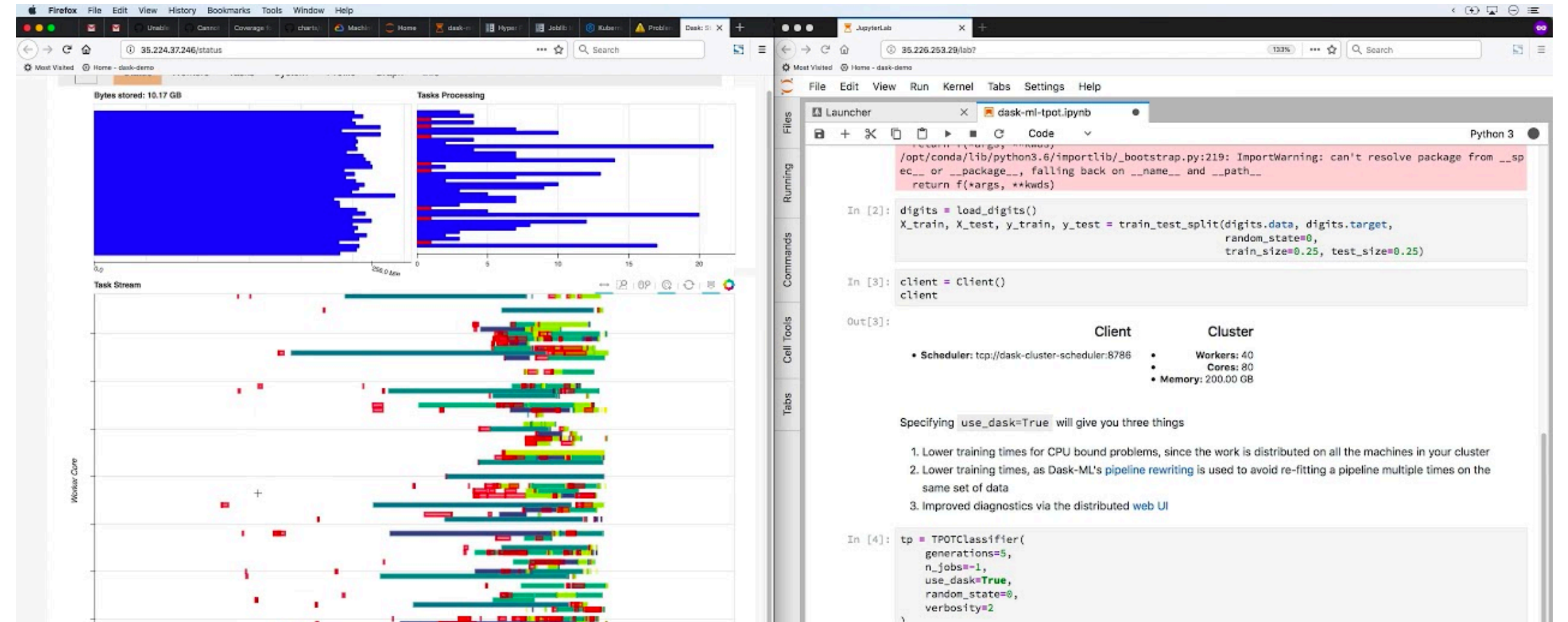
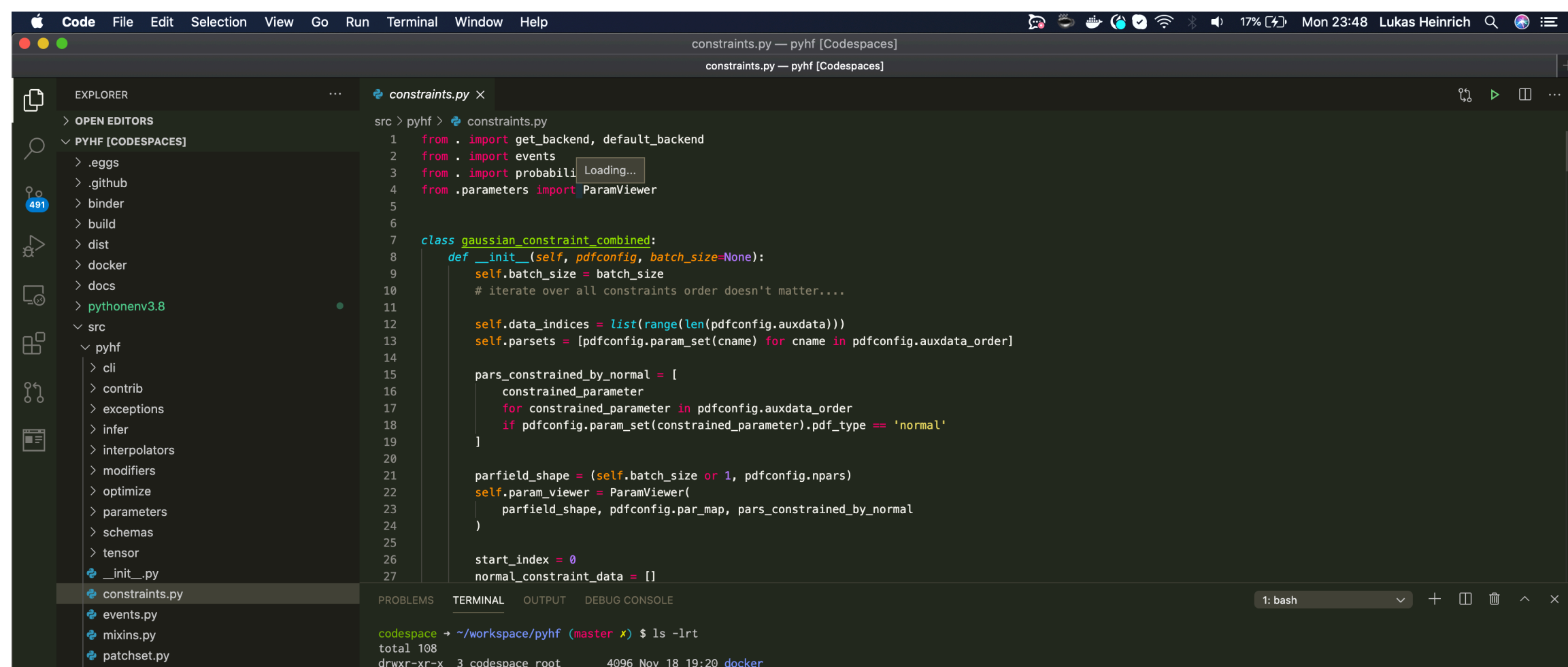
PROOF had the right vision, but technology wasn't really there

A modern take on interactive analysis

Data Science community is investing a lot in “distributed data frame” processors for large-scale data analysis. Good example: Dask



Web-based technologies instead of Native GUIs: Jupyter, VS Code

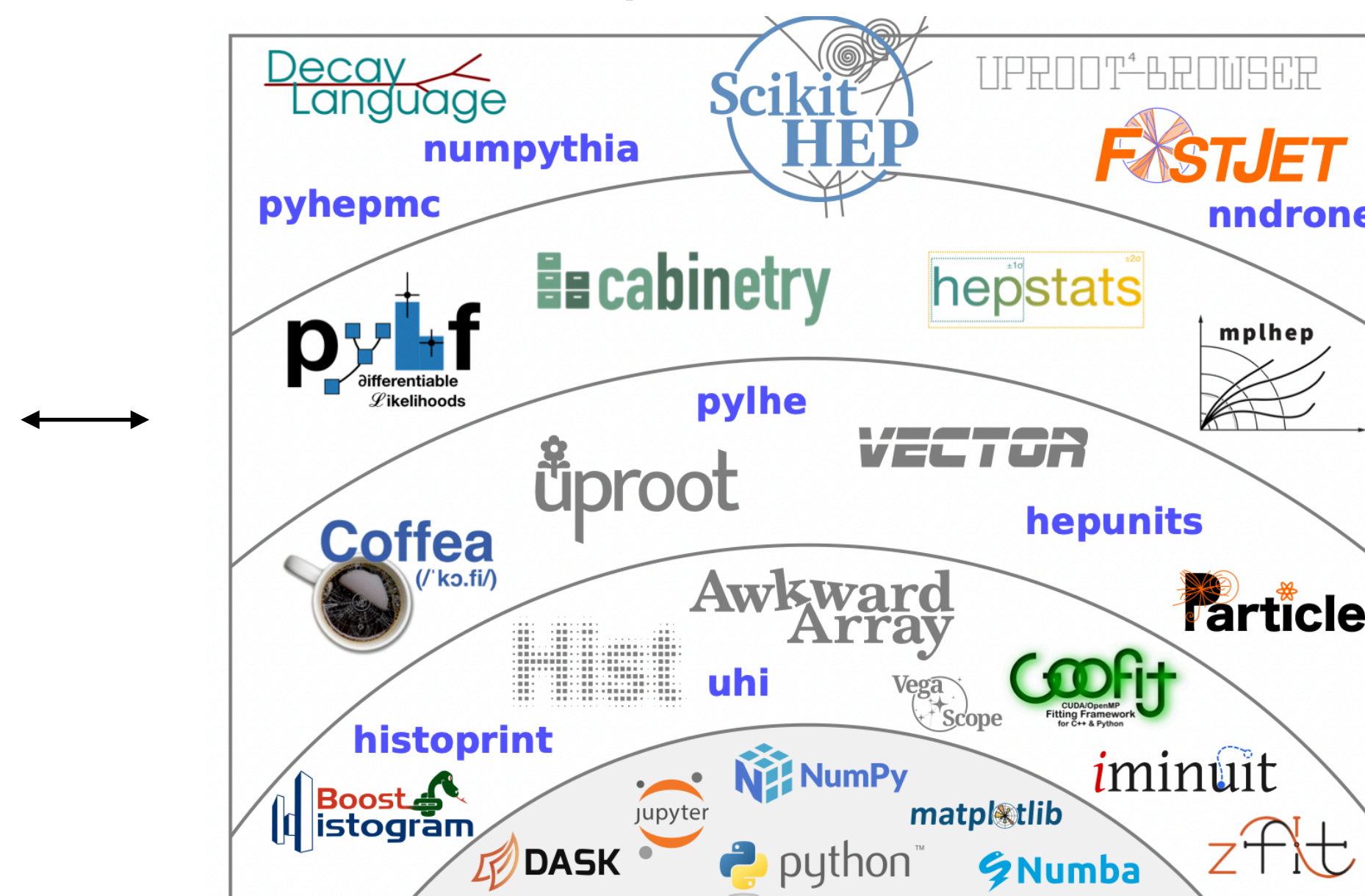
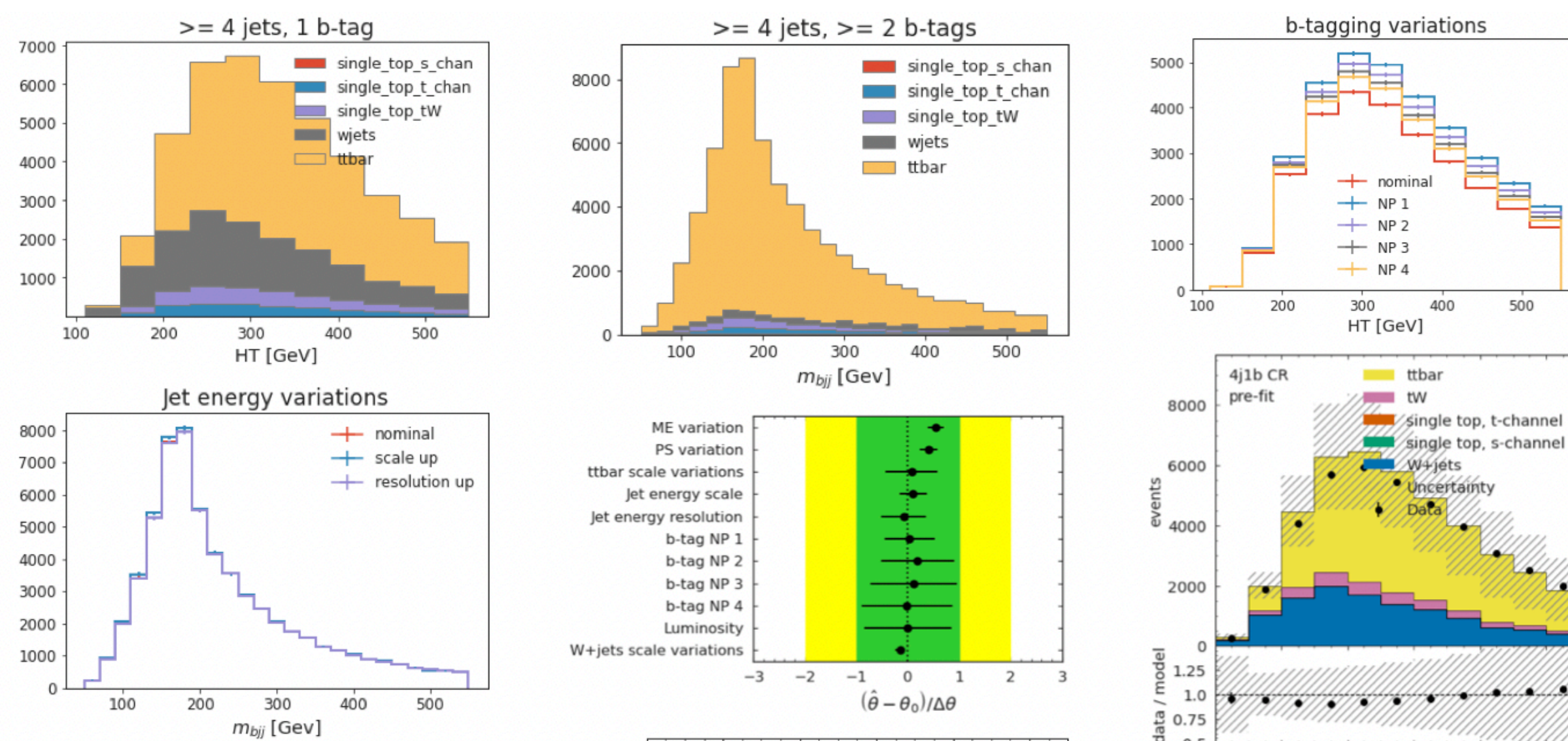


Interactive Analysis for HEP

Closely linked to developments for pythonic HEP analysis

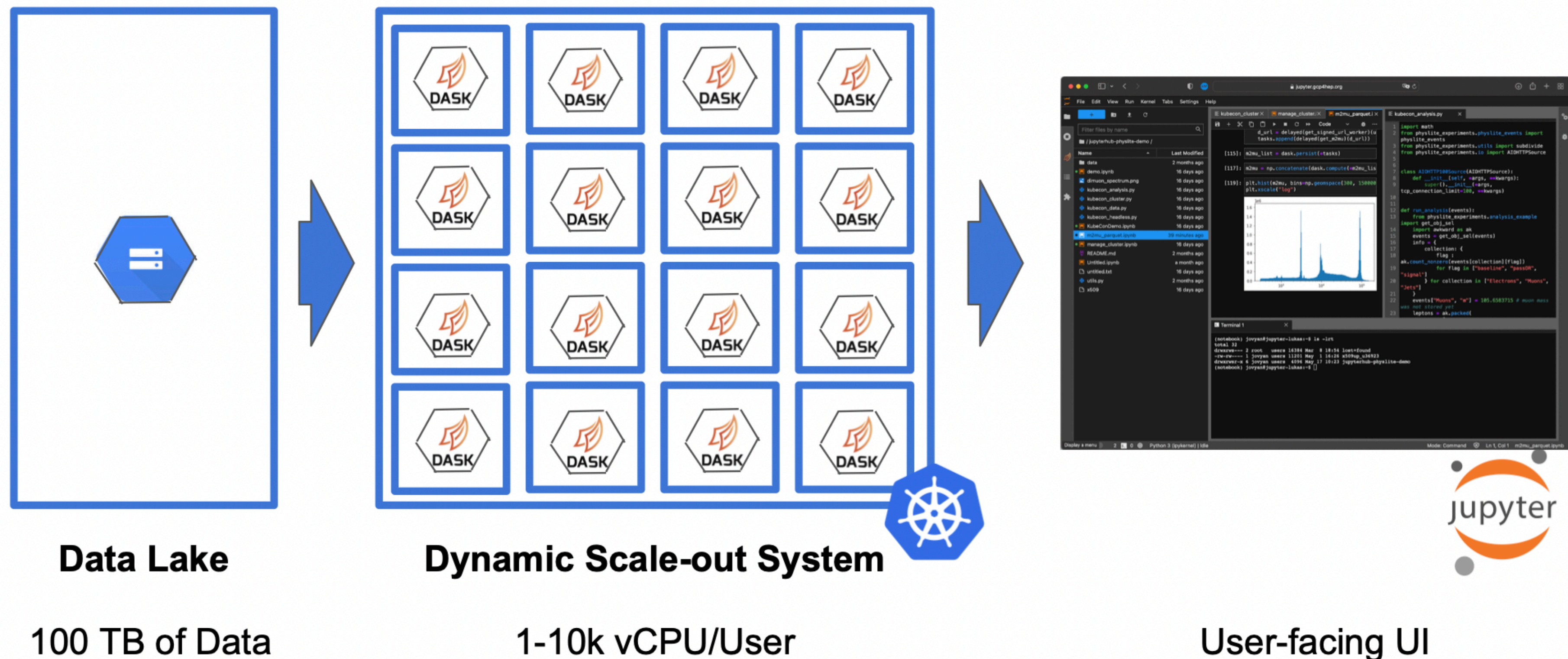
- easy to do “demos”, but for realistic analysis require a full solution for systematic uncertainties, ...

IRIS-HEP Analysis Grand Challenge: demonstrate large-scale and realistic data analysis on Open Data (the “A” in FAIR”)



A recurring Picture

Again, “Jupyter” and “Dask” are just add. applications for Kubernetes like REANA and Harvester: Install via Helm & Scale dynamically



Scale on Google Cloud

Deploy Jupyter + Dask with VO-wide authentication.

- All ATLAS users could get access
- Data via Rucio-on-Google (to minimize data transfer in/out of cloud)
- Custom environments for ML (GPU + ML frameworks)

The image displays the JupyterLab interface for scaling on Google Cloud. On the left, a sidebar shows the 'jupyterhub' logo and a 'Sign in with ATLAS IAM' button. The main area is titled 'Server Options' and lists five environments:

- ☐ **PHYSLITE environment [prod]**
Validated environment for experiments with DAOD_PHYSLITE with uproot, awkward etc.
- ☐ **ML environment**
keras, flatbuffers, joblib, pillow, pytz, scikit-learn, scipy, uproot, root-numpy
- ☐ **ML environment with T4 GPU**
Similar to the previous ML environment, but it boots a VM with a GPU just for YOU. Only to be used when working on ML-GPU applications. Setting up your notebook will take 10 minutes or more.
- ☐ **ML environment with P100 GPU**
Similar to the previous ML environment, but it boots a VM with a GPU just for YOU. Only to be used when working on ML-GPU applications. Setting up your notebook will take 10 minutes or more.
- ☐ **ML environment with K80 GPU**
Similar to the previous ML environment, but it boots a VM with a GPU just for YOU. Only to be used when working on ML-GPU applications. Setting up your notebook will take 10 minutes or more.

At the bottom of the 'Server Options' panel is an orange 'Start' button. An arrow points from this button to the right, where a JupyterLab notebook is open. The notebook has a sidebar with a 'Task Stream' plot showing a green area under a curve, and a 'Progress' panel showing the status of tasks. The main code area contains Python code for scaling and running a histogram.

```
def get_signed_url_worker(proxy_data, scope, name, rucio_account="fbarreir", ca_cert=False):
    import rucio.client
    setup_rucio_and_proxy(proxy_data, rucio_account=rucio_account, proxy_path="XS09")
    rucio_client = rucio.client.Client(ca_cert=ca_cert)
    return get_signed_url(rucio_client, scope, name)

def get_m2mu(url, row_groups=None):
    ak.behavior.update(vector.behavior)
    with fsspec.open(url, cache_type="none") as f:
        events = ak.from_parquet(f, lazy=True, row_groups=row_groups)
        p4 = ak.zip({k: events.Nuons[k] for k in ["pt", "eta", "phi"]}, with_name="PtEtaPhiMLorentzVector")
        p4["mass"] = 185.6583715
        mu1, mu2 = ak.unzip(ak.combinations(p4, 2))
        return ak.to_numpy(ak.flatten((mu1 + mu2).mass))

Run the m2mu computation in parallel, then aggregate the results

[12]: client = cluster.get_client()
      cluster.scale(465)

Define distributed methods

[13]: def get_signed_url_worker(proxy_data, scope, name, rucio_account="fbarreir", ca_cert=False):
      import rucio.client
      setup_rucio_and_proxy(proxy_data, rucio_account=rucio_account, proxy_path="XS09")
      rucio_client = rucio.client.Client(ca_cert=ca_cert)
      return get_signed_url(rucio_client, scope, name)

      def get_m2mu(url, row_groups=None):
          ak.behavior.update(vector.behavior)
          with fsspec.open(url, cache_type="none") as f:
              events = ak.from_parquet(f, lazy=True, row_groups=row_groups)
              p4 = ak.zip({k: events.Nuons[k] for k in ["pt", "eta", "phi"]}, with_name="PtEtaPhiMLorentzVector")
              p4["mass"] = 185.6583715
              mu1, mu2 = ak.unzip(ak.combinations(p4, 2))
              return ak.to_numpy(ak.flatten((mu1 + mu2).mass))

      Run the m2mu computation in parallel, then aggregate the results

[14]: tasks = []
      for file in files:
          d_url = delayed(get_signed_url_worker)(upload.data[-1], file["scope"], file["name"])
          tasks.append(delayed(get_m2mu)(d_url))
      m2mu_list = dask.persist(*tasks)
      m2mu = np.concatenate(dask.compute(*m2mu_list))

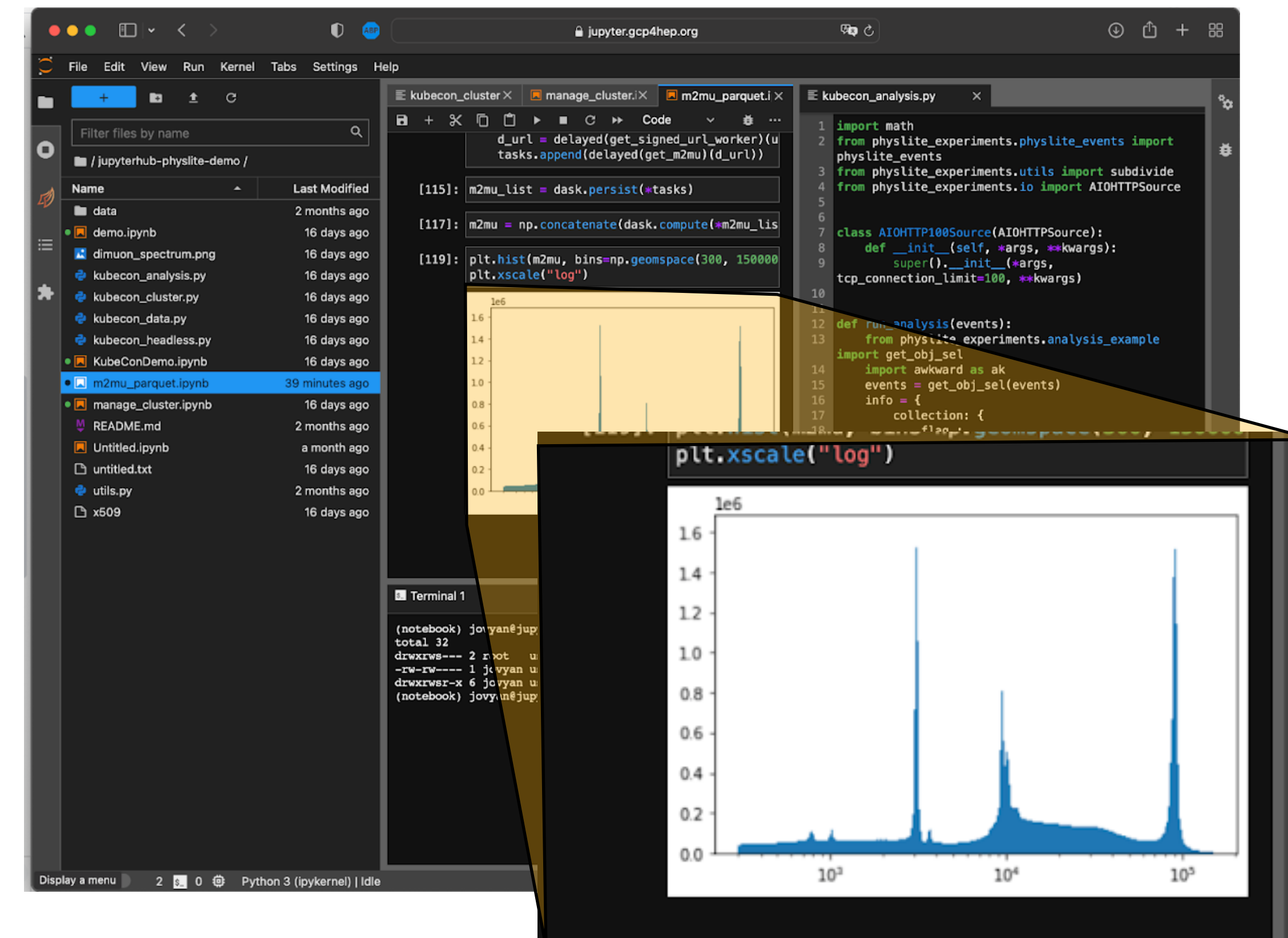
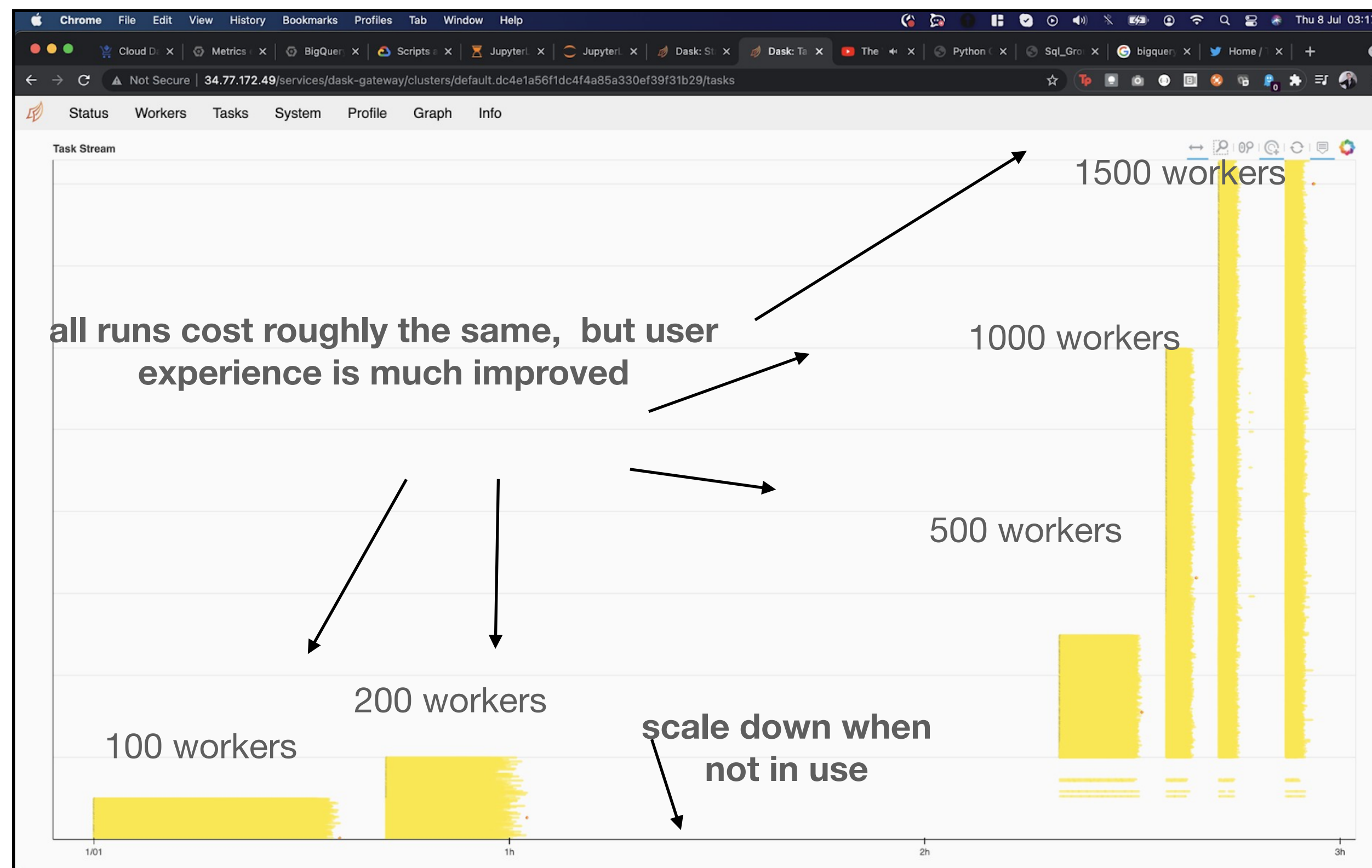
      Plot the results

[9]: plt.hist(m2mu, bins=np.geomspace(300, 150000, 500))
     plt.xscale("log")
```

The 'Task Stream' plot shows a green area under a curve, and the 'Progress' panel shows the status of tasks: 'get_m2mu' (461 / 465) and 'get_signed_u...' (465 / 465).

A modern Take

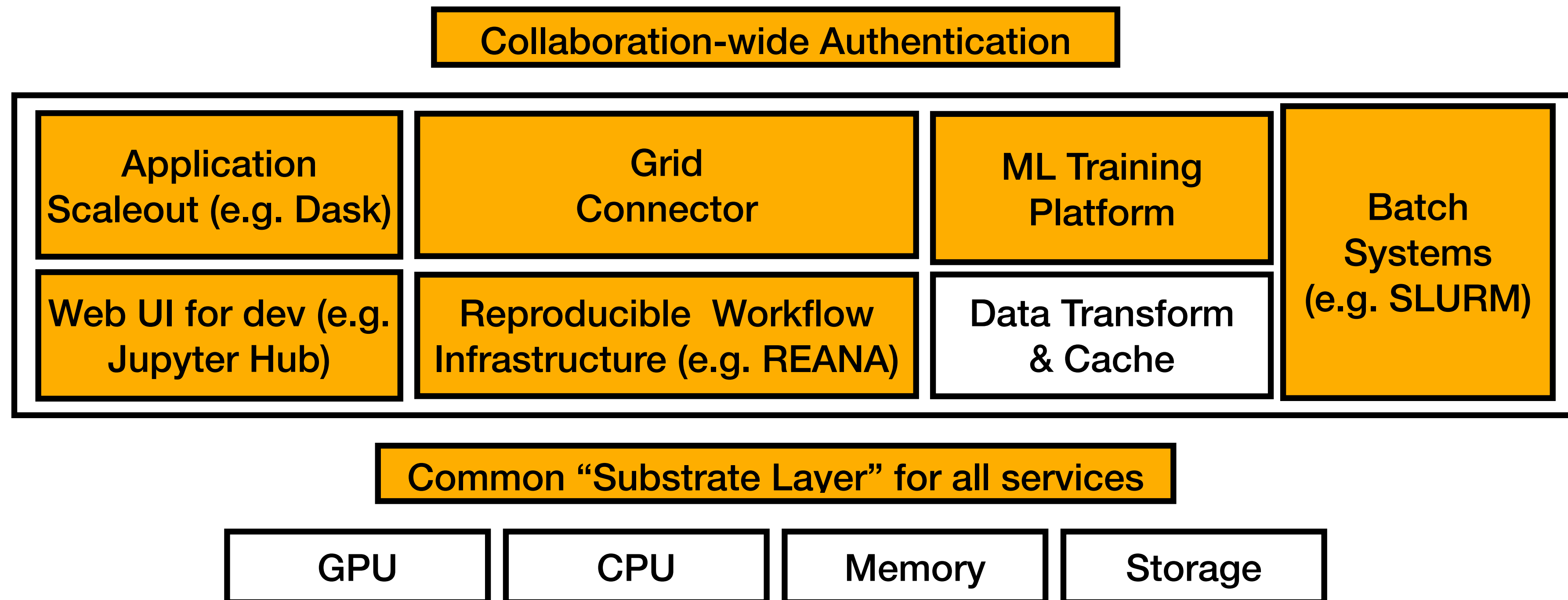
Exploit elasticity of cloud: scale for $O(1000)$ cores for a user analysis
Example: Throughput test for $\mu\mu$ -invariant mass of full Run-2
on ATLAS data in minutes



Wrapping Up

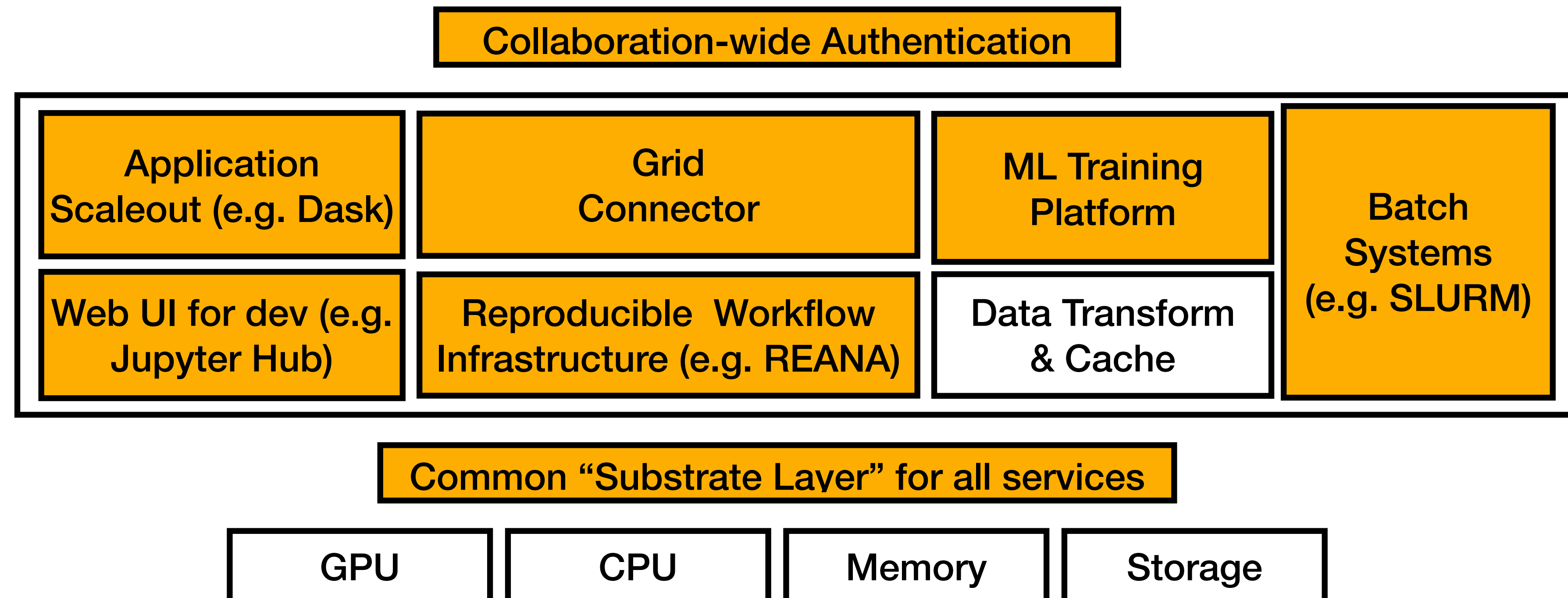
We have all the pieces together to build future analysis facilities & know we can scale: Opportunity for close collaboration across communities

“Forschungsdateninfrastruktur!”



Wrapping Up

Kubernetes seems to be a great option, but we need to work more on our ability to deploy it within an academic context (HPC, Labs, Universities) → question of policy and training



Wrapping Up

HEP Software Foundation is providing a Forum to exchange ideas, experience as we work towards this future



Analysis Facilities Forum Kick-off Meeting

25 March 2022
Europe/Zurich timezone

[Overview](#)[Timetable](#)[Registration](#)[Participant List](#)[Code of Conduct](#)

We are pleased to announce a kick-off meeting of the HSF "Analysis Facilities Forum" organised in collaboration with IRIS-HEP, which will be held virtually on March 25, 2022.

The Analysis Facilities (AF) Forum provides a community platform for those interested in contributing to the development of analysis facilities for use within HEP experiments to develop and exchange ideas. One should interpret "development" in the broad sense, including contribution of ideas from potential end users for functionality to support the analysis of HEP data, specification and planning of the facilities themselves, and technical developments needed to realize AFs. Of course, HEP experiments have their own internal processes for developing and deploying AFs. This AF Forum is intended to support and

Thursdays at 17:00
biweekly meetings