CERN

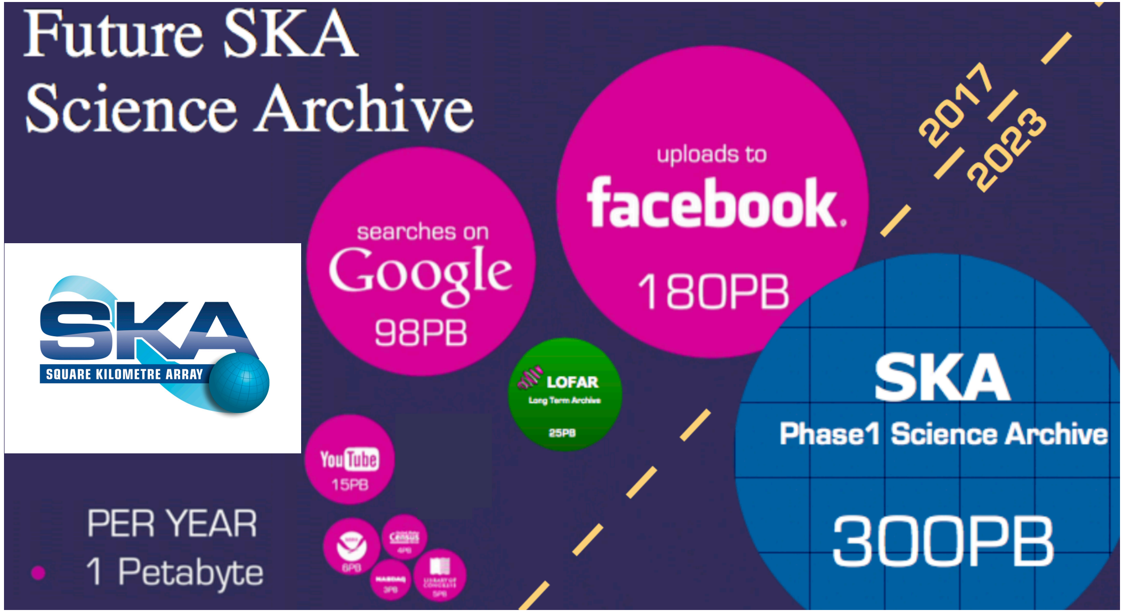
Data Storage, Management and Access evolution

3rd GÉANT SIG-CISS (Cloudy Interoperable Software Stacks)

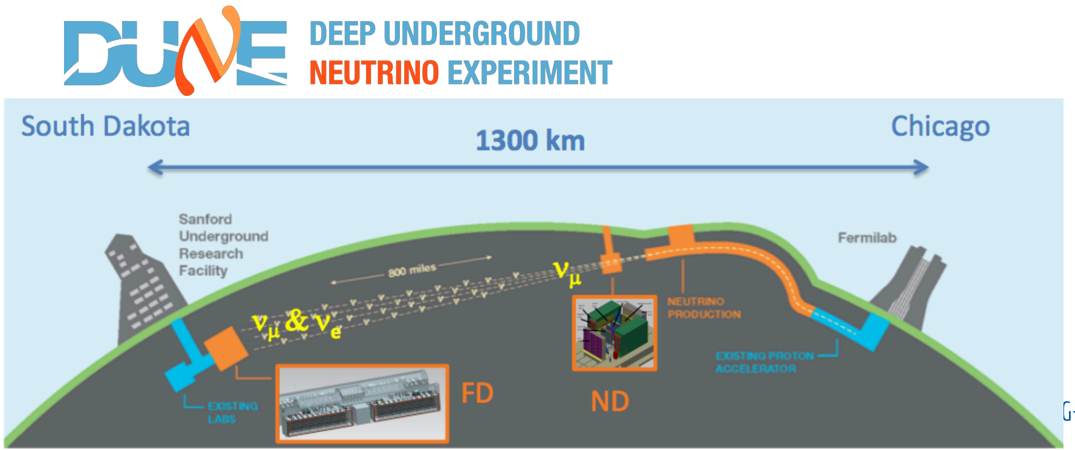
Xavier Espinal (CERN)

The motivation

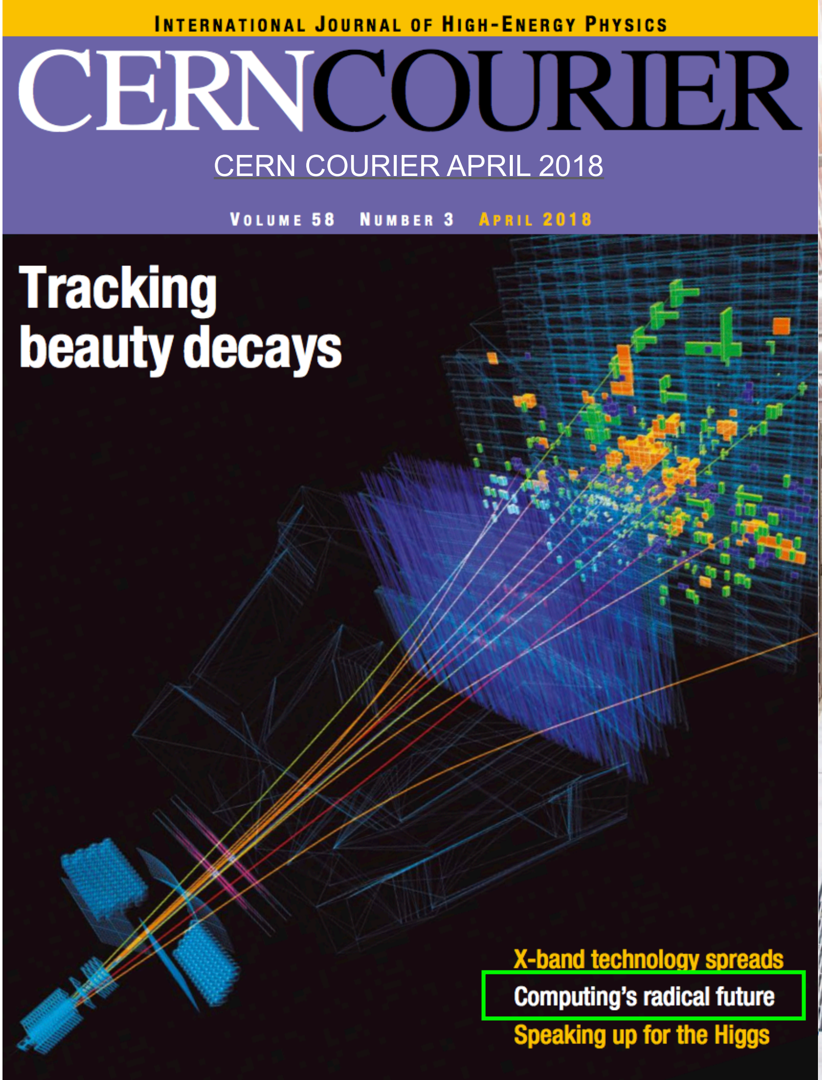
- Change of scale in data volumes is common to all scientific communities: physics, astrophysics, cosmology
- More data not only means more bytes. Classic scaling solutions do not apply anymore

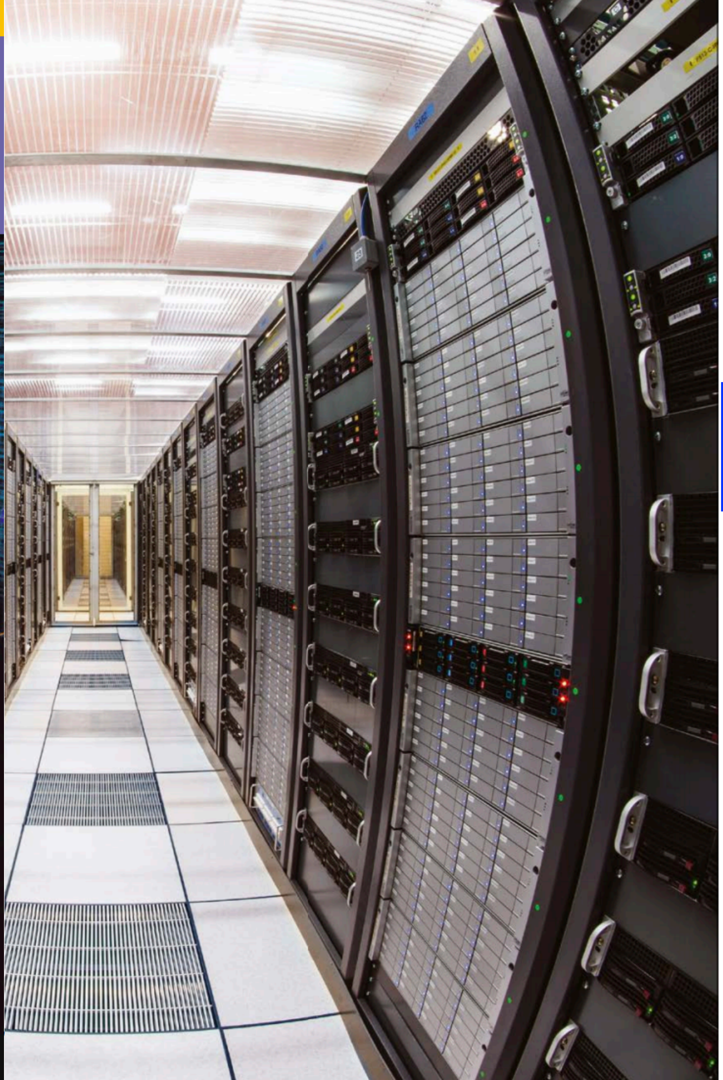

Future SKA
Science Archive
SKA
SQUARE KILOMETRE ARRAY
searches on
Google
98PB
uploads to
facebook.
180PB
2017
2023
LOFAR
Long Term Archive
25PB
You Tube
15PB
SKA
Phase1 Science Archive
300PB
PER YEAR
1 Petabyte


LSST
20PB/night


DUNE
DEEP UNDERGROUND
NEUTRINO EXPERIMENT
South Dakota
1300 km
Chicago
Sanford
Underground
Research
Facility
800 miles
Fermilab
ν_μ & ν_e
ν_μ
NEUTRINO
PRODUCTION
EXISTING PROTON
ACCELERATOR
EXISTING
LABS
FD
ND


SKA
SQUARE KILOMETRE ARRAY
Low Frequency Antenna





Software and computing

Time to adapt for big data

Radical changes in computing and software are required to ensure the success of the LHC and other high-energy physics experiments into the 2020s, argues a new report.

It would be impossible for anyone to conceive of carrying out a particle-physics experiment today without the use of computers and software. Since the 1960s, high-energy physicists have pioneered the use of computers for data acquisition, simulation and analysis. This hasn't just accelerated progress in the field, but driven computing technology generally – from the development of the World Wide Web at CERN to the massive distributed resources of the Worldwide LHC Computing Grid (WLCG) that supports the LHC experiments. For many years these developments and the increasing complexity of data analysis rode a wave of hardware improvements that saw computers get faster every year. However, those blissful days of relying on Moore's law are now well behind us (see panel overleaf), and this has major ramifications for our field.

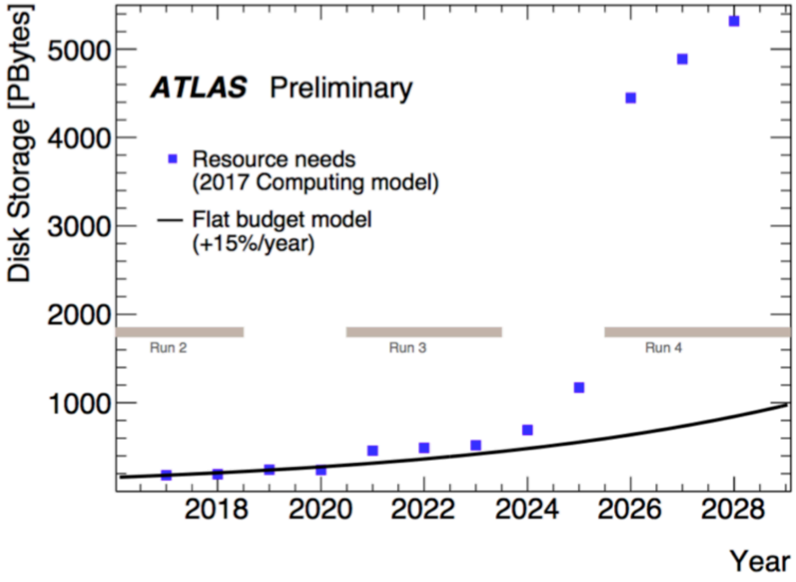
The high-luminosity upgrade of the LHC (HL-LHC), due to enter operation in the mid-2020s, will push the frontiers of accelerator and detector technology, bringing enormous challenges to software and computing (*CERN Courier* October 2017 p5). The scale of the HL-LHC data challenge is staggering: the machine will collect almost 25 times more data than the LHC has produced up to now, and the total LHC dataset (which already stands at almost 1 exabyte) will grow many times larger. If the LHC's ATLAS and CMS experiments project their current computing models to Run 4 of the LHC in 2026, the CPU and disk space required will jump by between a factor of 20 to 40 (figures 1 and 2).

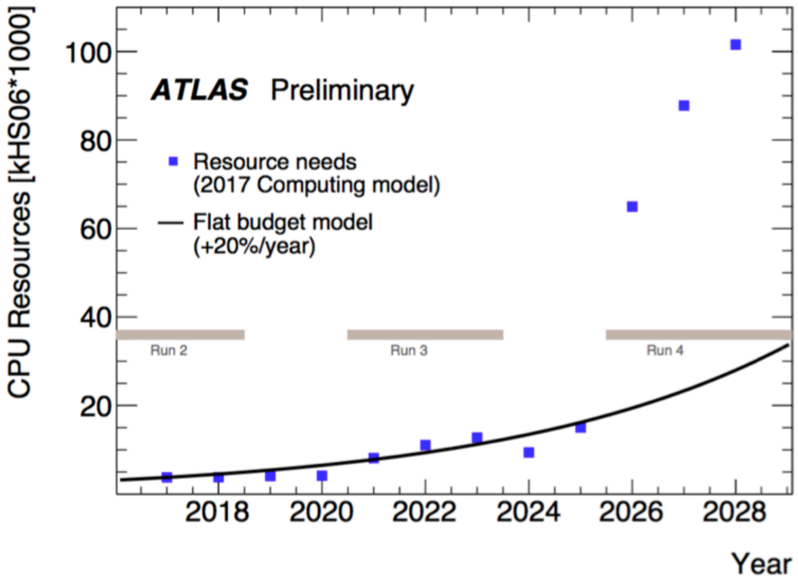
Even with optimistic projections of technological improvements there would be a huge shortfall in computing resources. The WLCG hardware budget is already around 100 million Swiss francs per year and, given the changing nature of computing hardware and slowing technological gains, it is out of the question to simply throw

Inside the CERN computer centre in 2017. (Image credit: J Ordan/CERN.)

The motivation

- Future storage needs are above the expected technology evolution (15%/yr) and funding (flat)





Evolution of federated storage (1/4)

- Redundancy:
  - RAIDs are dead. Market want big disks and redundancy on a single server not a solution anymore. High rebuilt times pose a risk for data loss and also impacts overall performance
  - Full replica duplication solves the single-location problem but cost increases
  - **E**rasure **C**oding (RAIN) could be a potential solution. But at which cost?
    - *Fat* disk servers and increased LAN traffic impact NICs, TORs and Routers
- Time to re-evaluate (or give-up) on redundancy?
  - Eliminate extra costs from: RAID, duplication, EC
  - Data can be reproduced.
    - Except RAW data (primary data coming from the detectors) which is anyway *custodial* (tape or cost-equivalent archive)
  - Reproducing data costs money (CPU cycles) but how much in comparison with the potential gain in storing more data?
    - ~1% of annual disks failure rate (for 100k disks installation -> 3 disks failures per day)

Evolution of federated storage (2/4)

- Data *auction*
  - Need to know what our stakeholders want: less data and more reliable or more date but less reliable?
    - 100PB of data at 10^{-5} annual reliability or 200PB at 10^{-4} annual reliability? … **or a mix of both?**
  - Data gets cold with time. Likelihood to be accessed decreases rapidly. Shouldn't the cost evolve accordingly?
- Leverage *byte-costs:* QoS (Quality of Service)
  - Does it makes sense to continue referring to *disk* and *tape* when we want to refer to *qualities* of the underlying storage services
    - Consumer disks vs. Enterprise disks vs. Tape vs. SSDs vs. RAIN
  - Shouldn't we give the flexibility to the sites? up to the users to choose what they need for their data in terms of:
    - Expected reliability (custodial data vs. transient files)
    - Expected access patterns (latency, IOPS)
    - Expected bandwidth
    - Expected cost
- File workflows: time evolving QoS
  - Data(set) evolves from 2 replicas to EC (8+3) to tape (or cost equivalent) backup

Evolution of federated storage (3/4)

- Large scale storage is complex and likely to worsen to maintain/operate
  - Data volumes moving towards the **EB** scale
  - Disks getting **big** (20TB+). **IOPS** falling. Disk server market favouring **high density** servers (1PB+/4U)
  - Adding **capacity** is a **routine**: should not be a scalability limit in the number of disk/servers.
    - Lightweight namespace disk server orchestration (messaging, notification, journaling,…)
  - Hardware **lifecycle** is **aggressive**: space density (TB/m^2) and power efficiency (TB/kW) keep increasing
    - Disk server replacements as standard operations and transparent to users: keeping data available with efficient draining and rebalancing mechanisms
- Concentrate big storage services on few sites (=data lakes)… and push for more high performance processing centres (=data caching+latency hiding) ?
  - Maintain caches require less effort (stateless service) and resources could be re-oriented to computing infrastructure
- Shouldn't the sites concentrate on what they have a chance to excel and take the most out of the resources?
  - Isn't better to have 1000 cores turning than 1PB of unaccessed data?

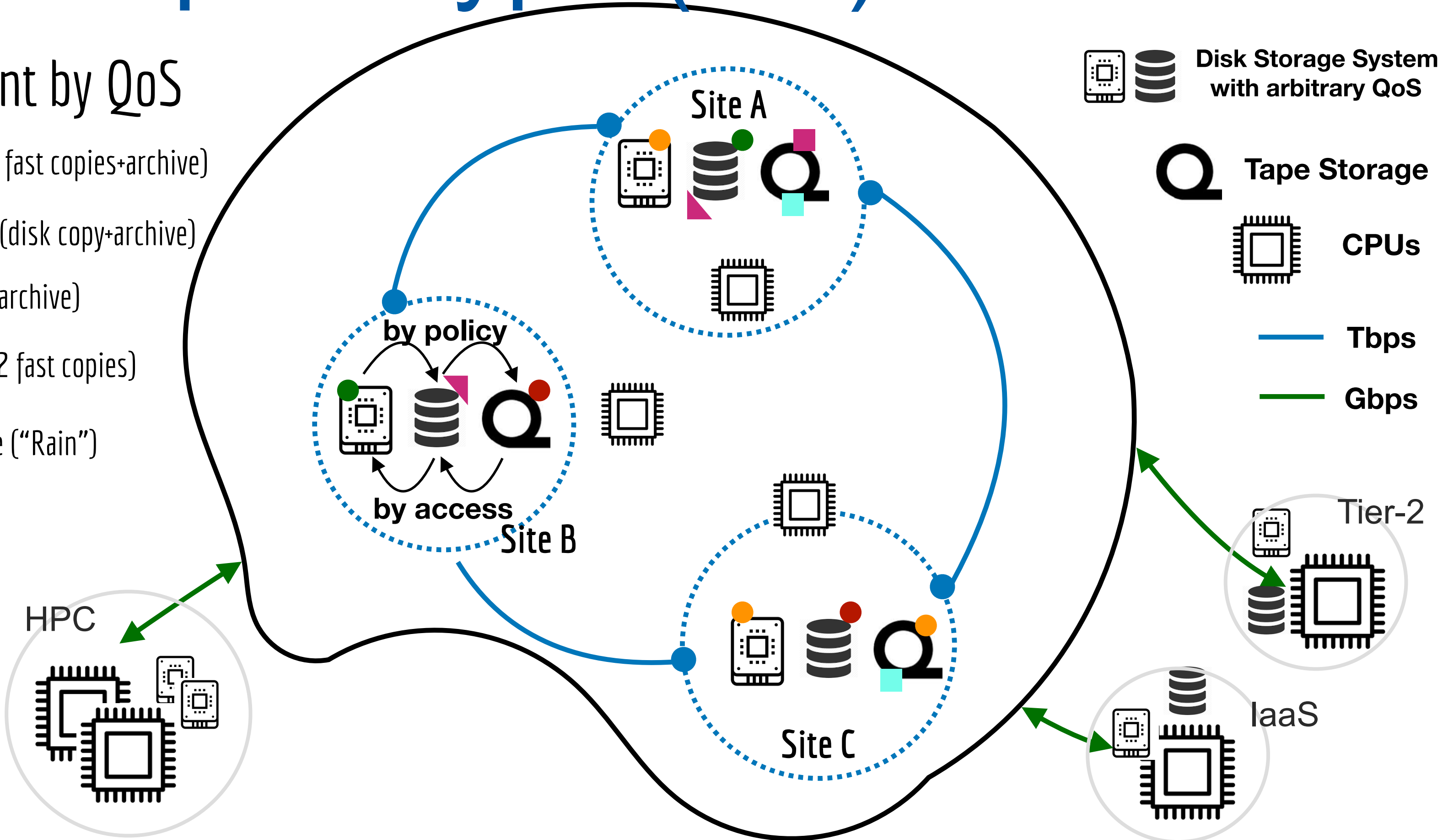
Evolution of federated storage (4/4)

- Expectation management
  - Understanding the access **patterns** is fundamental to tailor a service, ie. HPC centres invest a lot to align code to maximise resources exploitation
  - Many different **workflows** are needed in HEP before getting the final data products for scientists
    - And access patterns are very different: from nearly zero I/O and pure CPU for montecarlo (*HPC-like*) to intense I/O for reconstruction (*HTC-like*)
  - Can a single storage **system** provide High Throughput (HT) and High IOPS?
  - Can a single **hardware** provide HT and High IOPS (keeping costs under control)?
  - Should shared **filesystems** be treated different?
    - Home directories requiring high posix compliance, checkpointing capabilities and “infinite” uptime

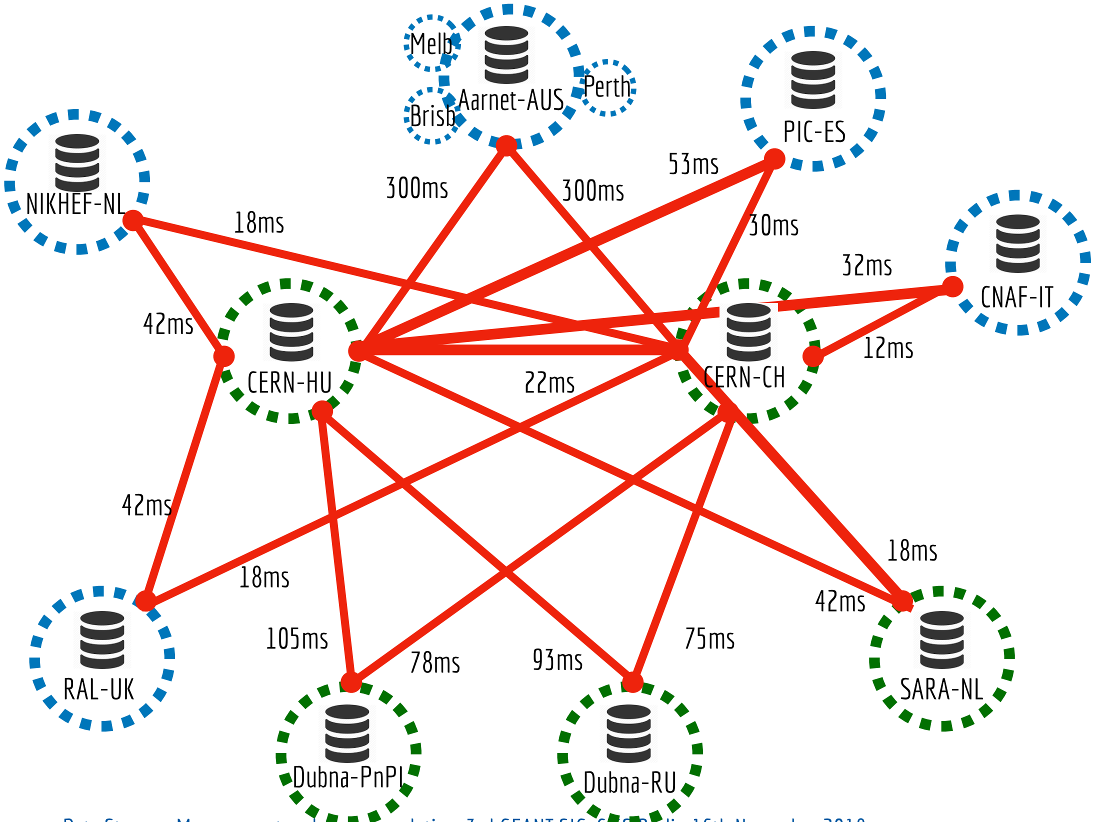
eulake prototype (1/4)

File placement by QoS

- Hot custodial file (2 fast copies+archive)
- Warm custodial file (disk copy+archive)
- Cold custodial file (archive)
- Hot ephemeral file (2 fast copies)
- Warm ephemeral file (“Rain”)

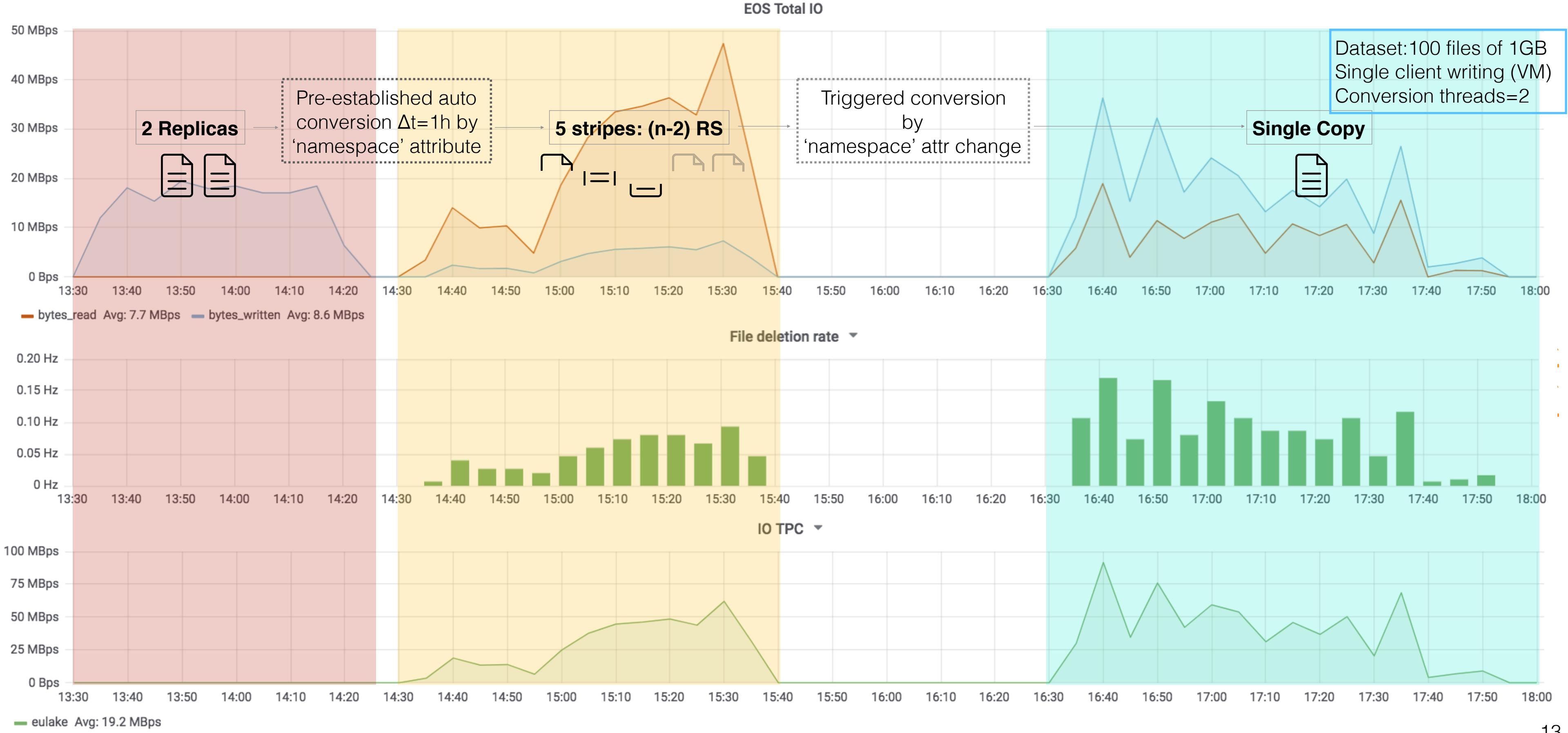


eulake prototype (2/4)



eulake prototype (3/4)

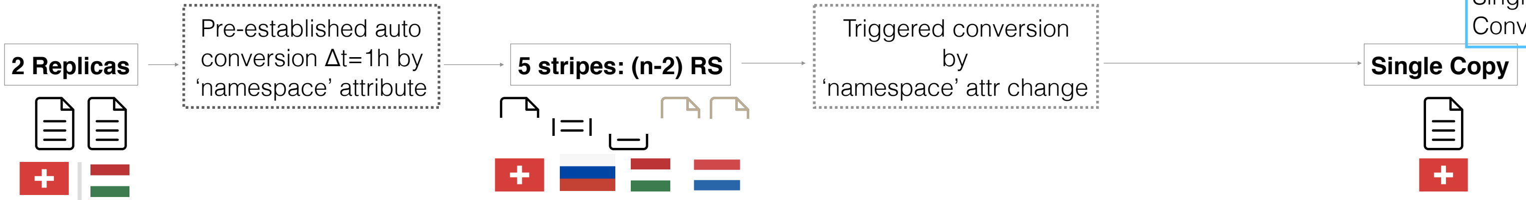
Distributed redundancy and QoS example



eulake prototype (4/4)

Distributed redundancy and QoS example

Dataset:100 files of 1GB
Single client writing (VM)
Conversion threads=2



180315 14:04:36 func=open path=/eulake/lcg/test/conversion/2replicas-to-rain32/file-workflow-2r-rain32.175.file
op=write target[0]=(**p05799459m56401.cern.ch**,33) target[1]=(**p05798818t49625.cern.ch**,80)

180315 15:04:58 time=1521123718.328306 func=open path=/eulake/lcg/test/conversion/2replicas-to-rain32/file-workflow-2r-rain32.175.file
op=read target[0]=(p05799459m56401.cern.ch,33) target[1]=(p05798818t49625.cern.ch,80)

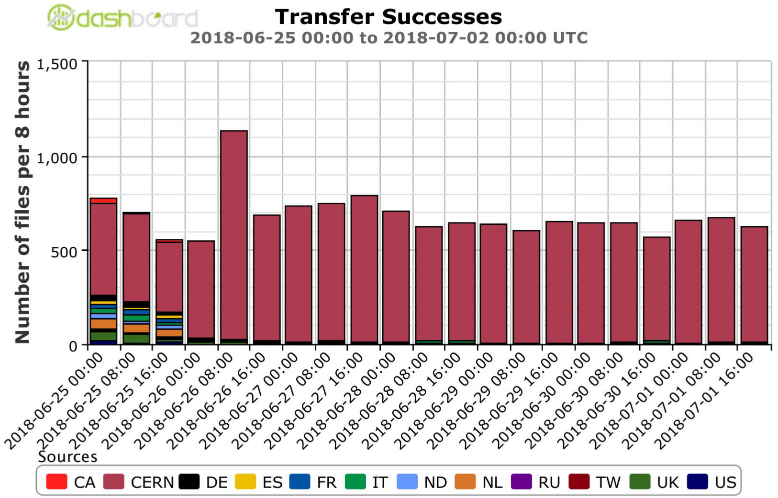
180315 15:04:58 func=open path=/eos/eulake/proc/conversion/000000000001819:default#20640442
op=write eos.layout.nstripes=5&eos.layout.type=raid6
target[0]=(**fst2.grid.surfsara.n**l,130) target[1]=**(p05496644k62259.cern.ch**,1) target[2]=(**dvl-mb01.jinr.ru**,122) target[3]=(**p05798818t49625.cern.ch**,97)
target[4]=(**fst1.grid.surfsara.nl**,124)

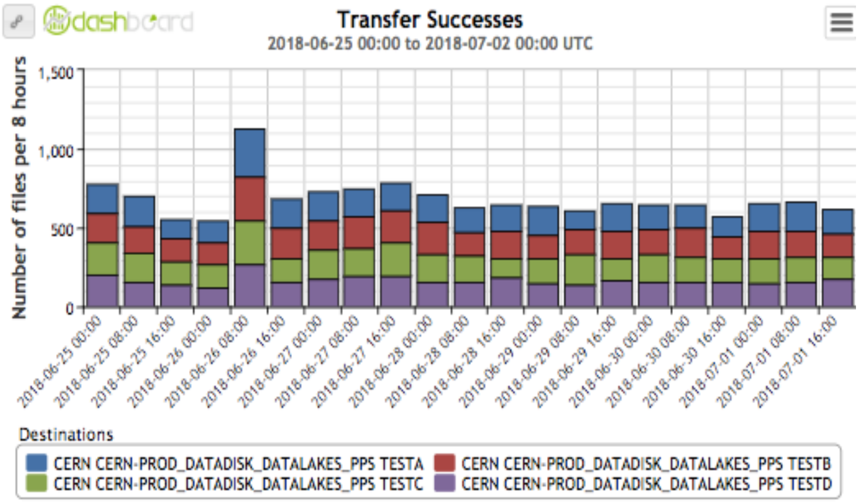
180315 17:22:17 func=open path=/eulake/lcg/test/conversion/2replicas-to-rain32/file-workflow-2r-rain32.175.file
op=read target[0]=(fst2.grid.surfsara.nl,130) target[1]=(p05496644k62259.cern.ch,1) target[2]=(dvl-mb01.jinr.ru,122)
target[3]=(p05798818t49625.cern.ch,97)

180315 17:22:17 func=open path=/eos/eulake/proc/conversion/00000000000018e2:default#00100001
op=write eos.layout.nstripes=1&eos.layout.type=plain tpc.stage=copy redirection=**p05799459m56401.cern.ch**?

eulake integration with ATLAS and CMS Data Management

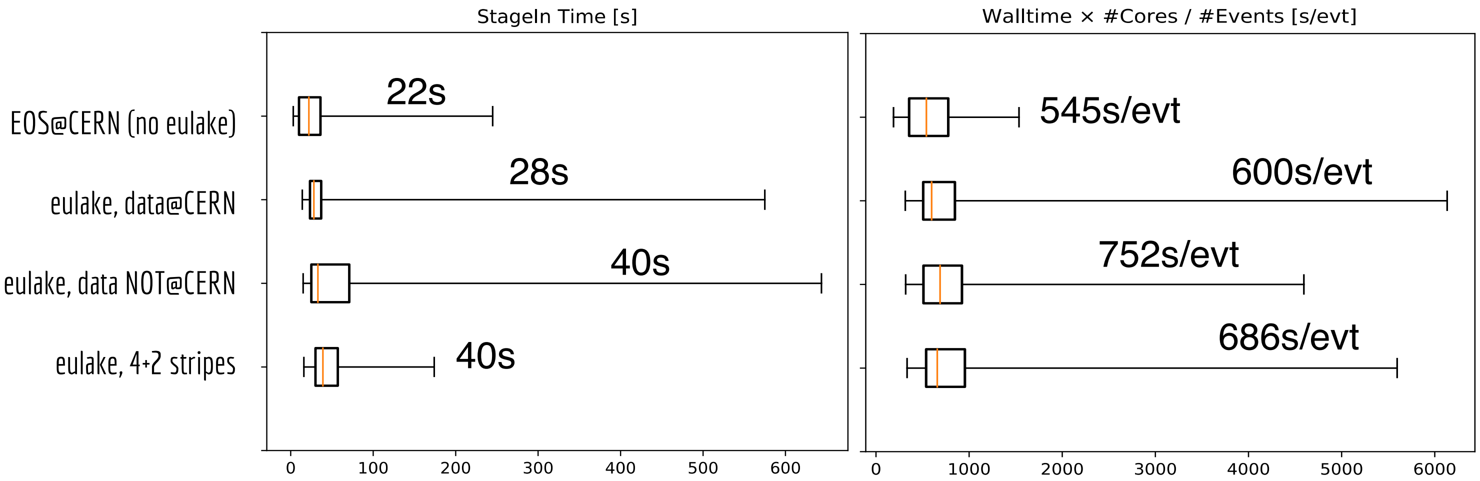
- eulake exposed to ATLAS and CMS data management system as storage endpoint
- Data can be transferred from any site into eulake (see ATLAS below)
- Stored input samples in different eulake areas for testing



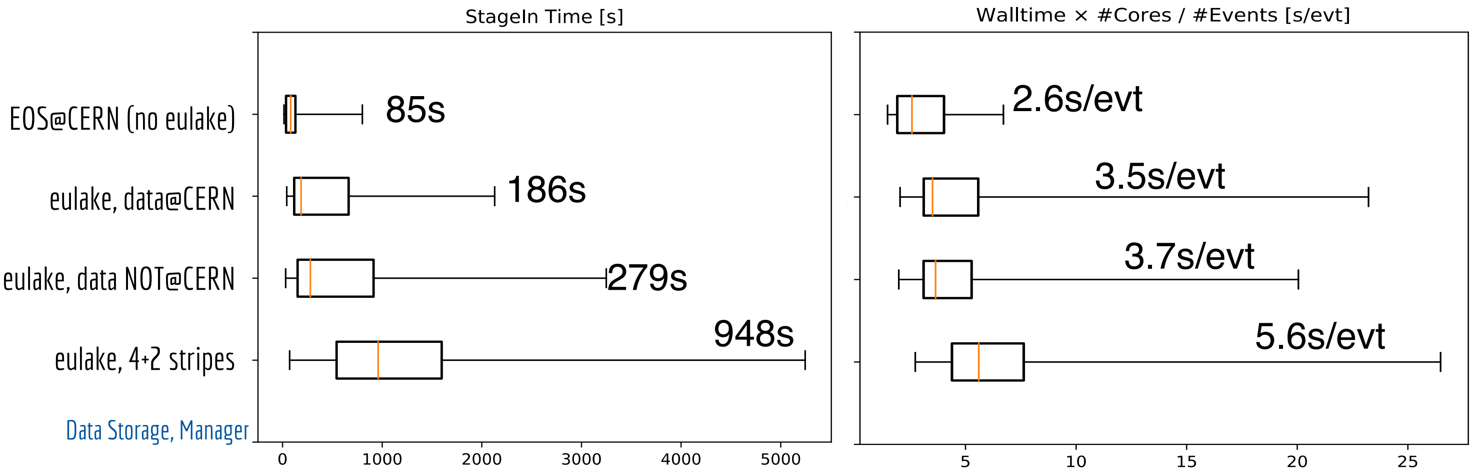


Low I/O intensity workflow (simulation)
~40MB input (1 file), 2 events,
~5 mins/event

Jun 2018

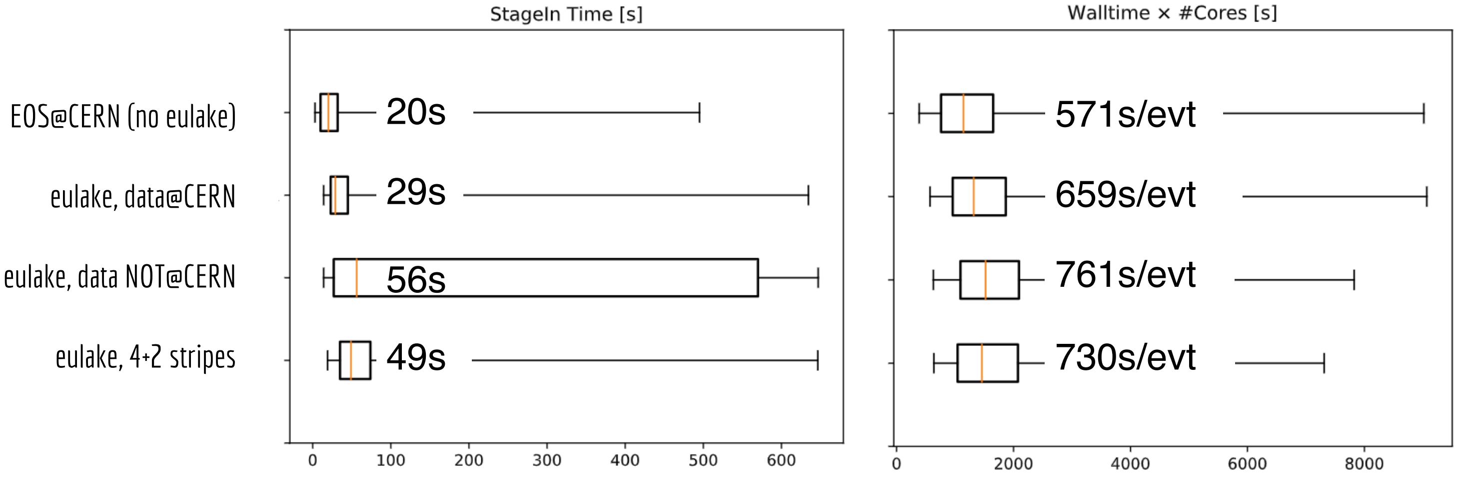


High I/O intensity workflow (DigiReco)
~6GB input (1 file), 1000 events
~2 seconds/event

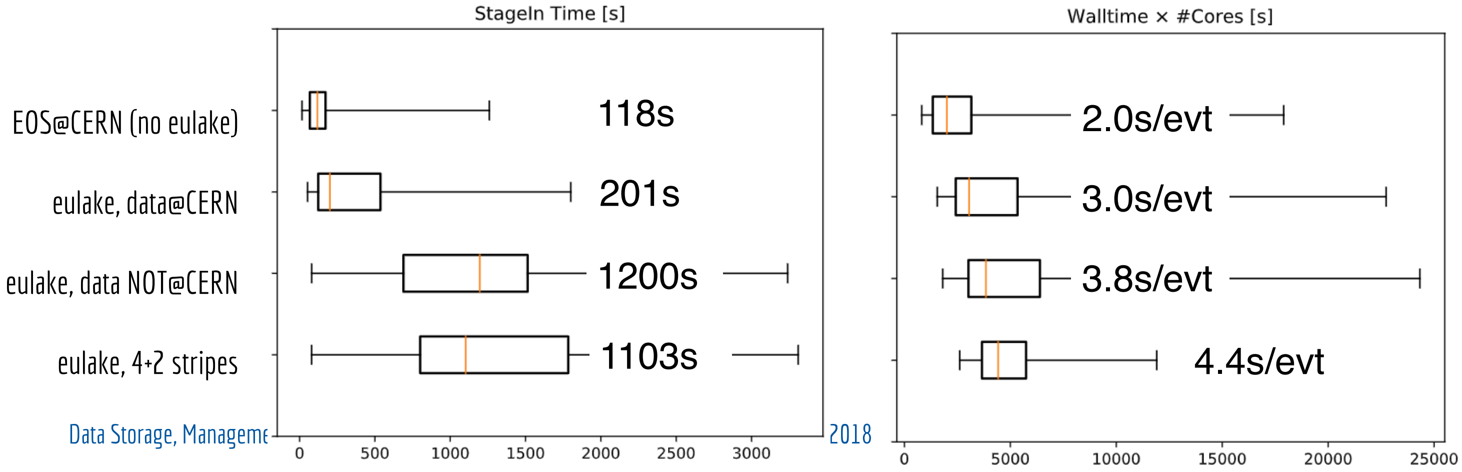


Low I/O intensity workflow (simulation)
~40MB input (1 file), 2 events,
~5 mins/event

Sept 2018



High I/O intensity workflow (DigiReco)
~6GB input (1 file), 1000 events
~2 seconds/event

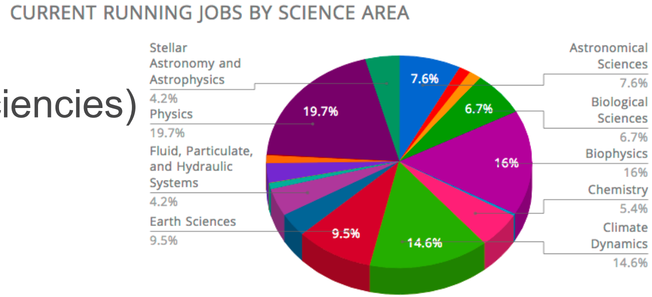


IaaS: could this be the solution?

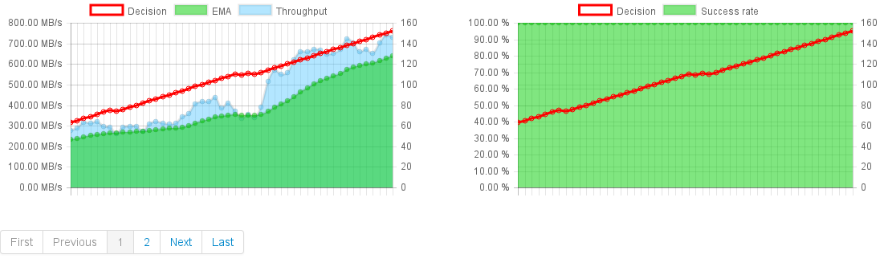
- Evaluated and continue being evaluated in HEP community
- Successful projects with main LHC experiments
  - Interoperability is ready (HTCondor integration)
- Perceived as a good mechanism for handling unforeseen workloads
  - Maximal exploitation of local resources remains the priority
  - IaaS reserved instances could be an option for expected (if any) computing capacity gaps
  - On-demand IaaS (*stock market*) could be an option for emergency computing
- IaaS benefits depend on: providers, type of workflows, performance and market evolution. But need to be ready to use them

HPC and HTC: Bringing T closer to P

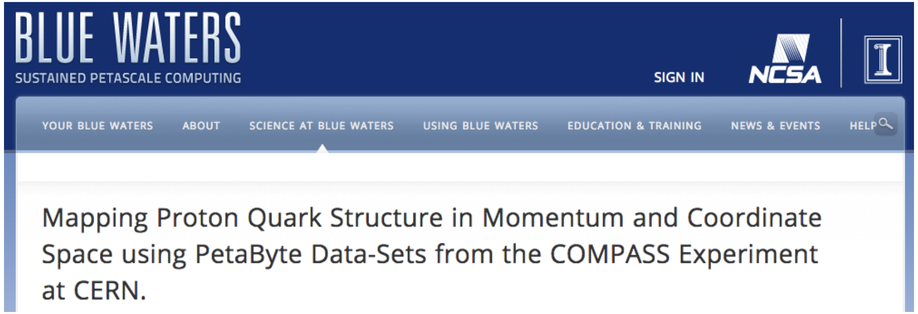
- Common interest and implication from experiments and HPC centres
- Proven for simulation/montecarlo. What about data intensive workloads?
  - Active caching for latency hiding
  - Smart application access by optimising data structures
  - Efficient workload orchestration (maximising cache efficiencies)



Details for srm://castorpublic.cern.ch → gsiftp://ie15.ncsa.illinois.edu



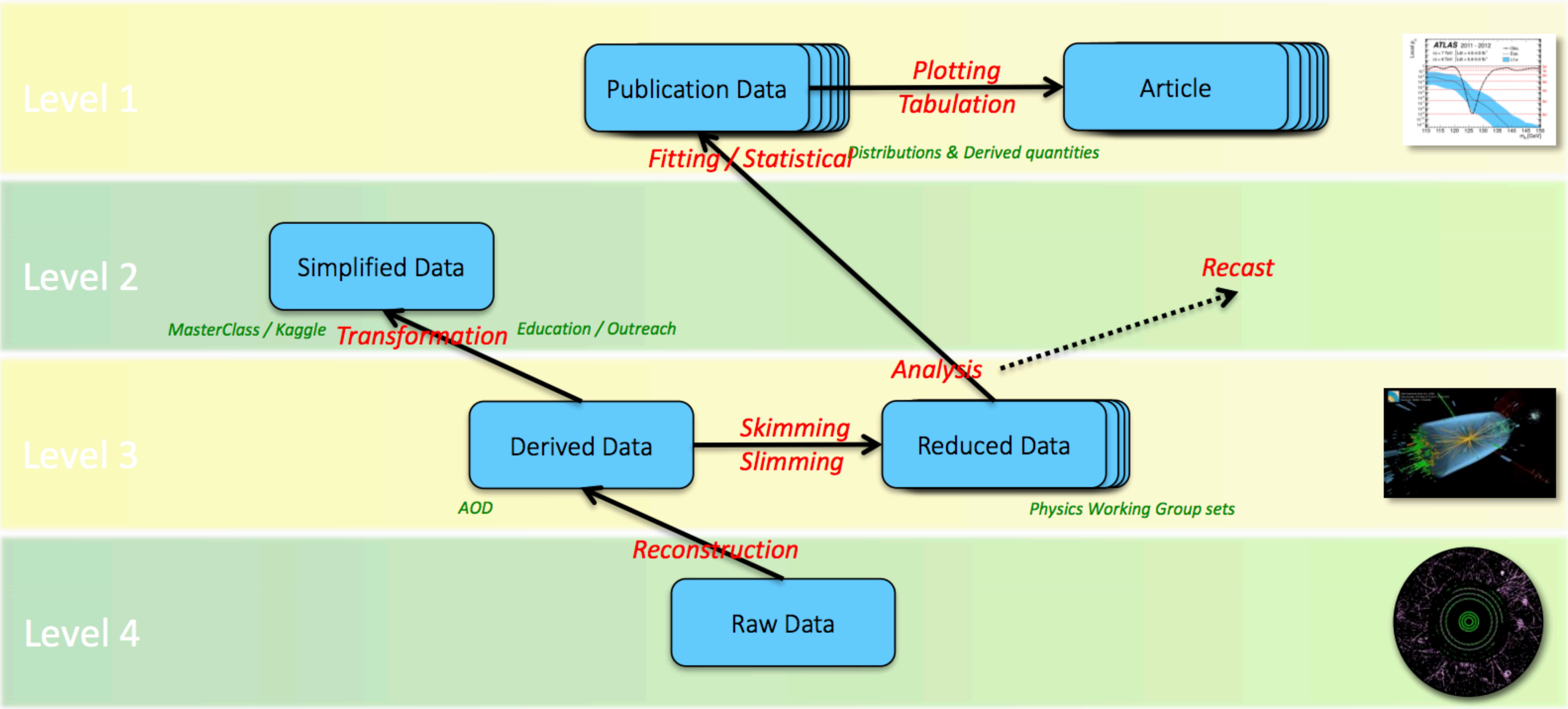
Timestamp	Decision	Running	Queue	Success rate (last 1min)	Throughput	EMA
2016-08-05T13:57:24	154	152	1898	100.00%	735.688 MB/s	648.032 MB/s



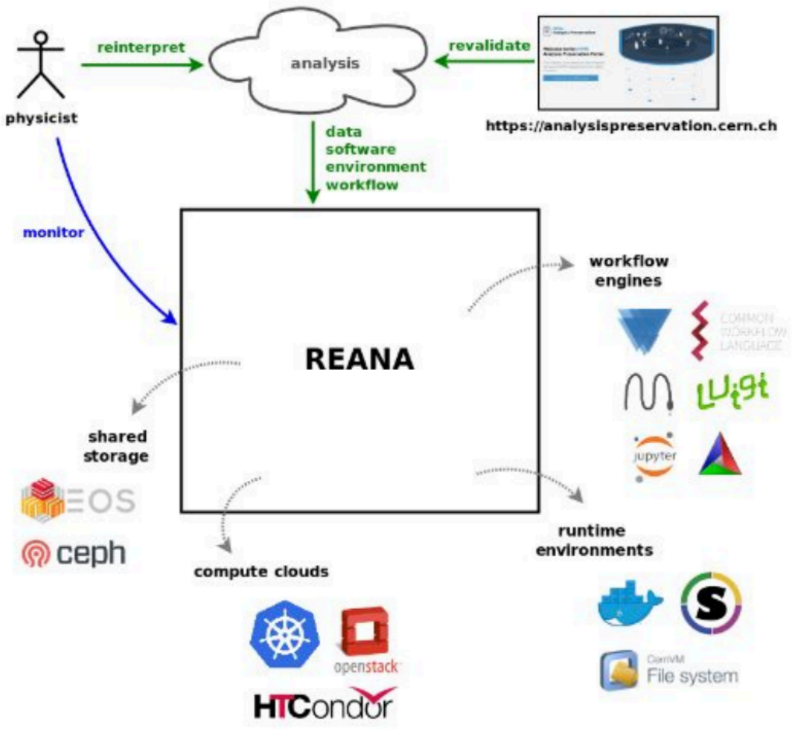
(re)analysis and knowledge preservation

- Preservation of data
- Reusability of data
- Reproducibility of results

(re)analysis and knowledge preservation



(re)analysis and knowledge preservation



http://www.reanahub.io/

reana

Reproducible research data analysis platform

Flexible	Scalable	Reusable	Free
Run many computational workflow engines.	Support for remote compute clouds.	Containerise once, reuse elsewhere. Cloud-native.	Free Software. GPL licence. Made with ❤ at CERN.

COMMON WORKFLOW LANGUAGE

kubernetes

CERN Analysis Preservation & REANA Workshop (30/06/2018)
https://indico.cern.ch/event/720455/

New ways of accessing data

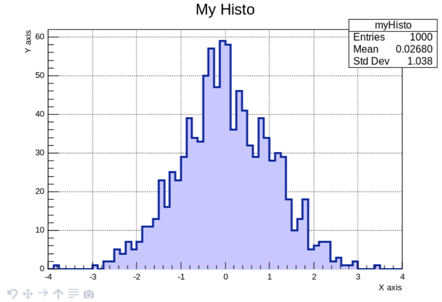
https://swan.web.cern.ch/

```
c.Dra();
```

```
input_line_62:5:3: error: no member named 'Dra' in 'TCanvas'
c.Dra()
~ ^
```

Oops! We mispelled a method. Luckily ROOT informed us about the typo. Let's draw the canvas properly:

In [11]:
```
c.Draw();
```



ROOT
Data Analysis Framework

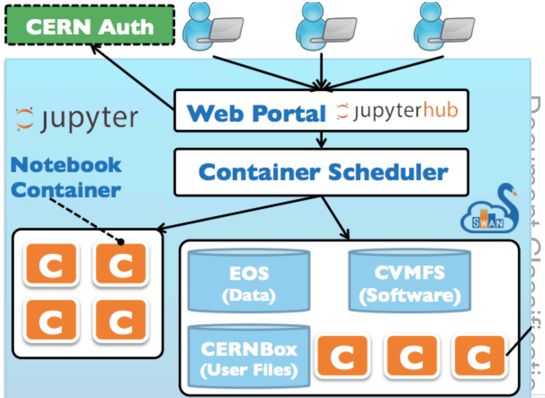
CERNBox

jupyterhub

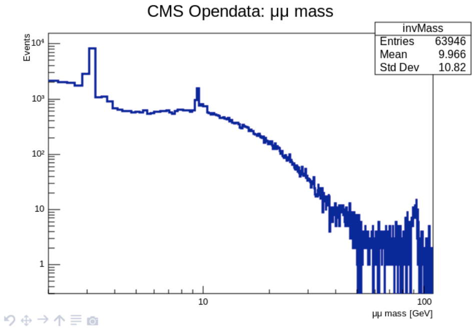
EOS

CERN

SWAN



In [5]:
```
invMass = ROOT.TH1F("invMass","CMS Opendata: #mu#mu mass;#mu#mu mass [GeV];Events",512, 2, 110)
invMassFormula = "sqrt((E1 + E2)^2 - ((px1 + px2)^2 + (py1 + py2)^2 + (pz1 + pz2)^2))"
cut = "Q1*Q2==-1"
c = ROOT.TCanvas()
dimuons.Draw(invMassFormula + " >> invMass",cut,"hist")
c.SetLogx()
c.SetLogy()
c.Draw()
```



That might have been too fast. We now make the analysis above more explicit producing a plot also for the J/Psi particle.

Web based **computing interface** combining: **data**, **code**, **equations**, text and **visualisation**

Summary

- Future scientific computing scenario force us to **re-evaluate** the current model
  - How we understand data storage
  - How we understand data access
  - How we understand data preservation
- Storage technology trends and funding not helping
- Revisiting **redundancy**, **caching, interoperability** and **reproducibility** should give us some of the hints to address the future of data storage in scientific computing
- Dedicated working groups starting **now** in WLCG to **set direction** and coordinate **R&D** projects:
  - **Content delivery and caching** (latency hiding, bandwidth and space optimisation)
  - **Protocols** (http/xrootd/tpc) and networks (tcp/udp, DTNs)
  - **Interoperability** and **Quality of Service** in storage systems