ATLAS Workflow and Data Management

Kaushik De, Alexei Klimentov

US ATLAS / CSI Workshop
July 26, 2018
BNL

The Large Hadron Collider

Exploration of a new frontier in Energy & Data

Today's LHC experiments managed data volume ~1 Exabyte

LHC ring:
27 km circumference

- General Purpose Detectors (ATLAS,CMS), proton-proton, heavy ions. Discovery of new physics: Higgs, SuperSymmetry
- LHCb : pp, B-Physics, CP Violation (matter-antimatter symmetry)
- ALICE : Heavy ions, pp (state of matter of early universe)

Basic Definitions

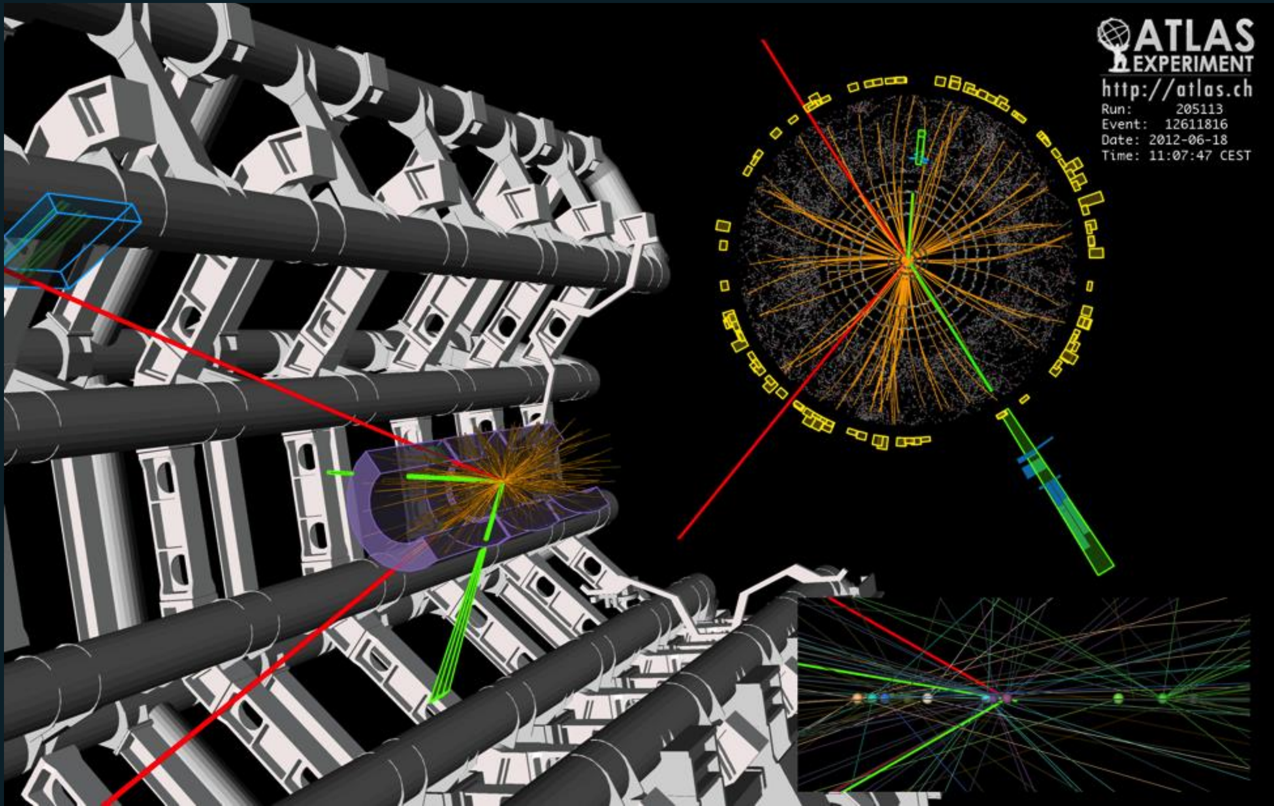
What is this data?

- Raw data:
  - Was a detector element hit?
  - How much energy?
  - What time?

- Reconstructed data:
  - Momentum of tracks (4-vectors)
  - Origin
  - Energy in clusters (jets)
  - Particle type
  - Calibration information
  - …

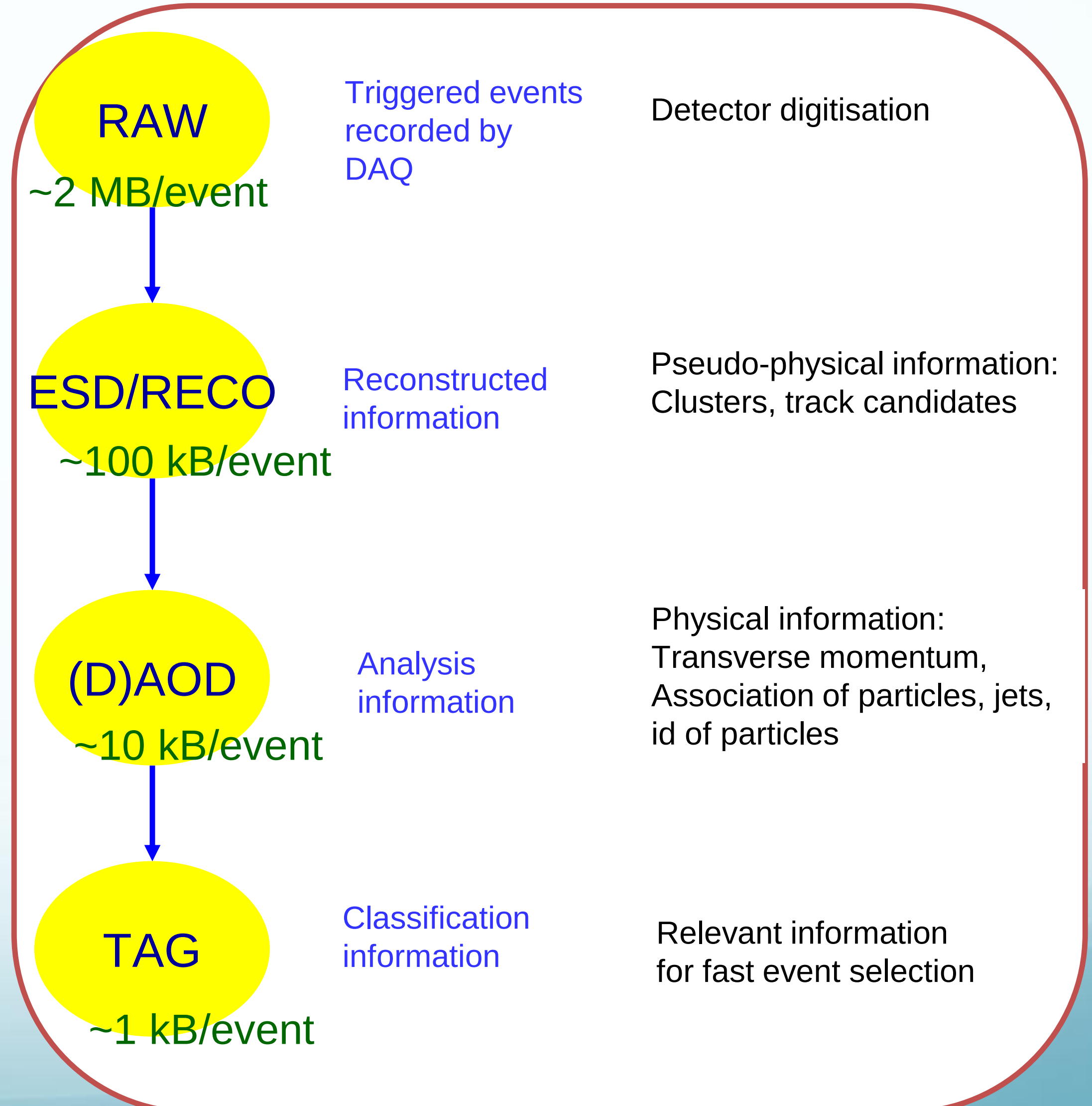
- 150 Million sensors deliver data … ~ 40 Million times per second

- Up to 6 GB/s to be stored and analysed after filtering



Data and Algorithms

- HEP data are organized as *Events* (particle collisions)
- Simulation, Reconstruction and Analysis programs process "one event at a time"
  - Events are fairly independent → Trivial parallel processing
- Event processing programs are composed of a number of algorithms selecting and transforming "raw" event data into"processed" (reconstructed) event data and statistics
- *ATLAS reconstruction and simulation code 5M LOC*
- *1000 software developers*



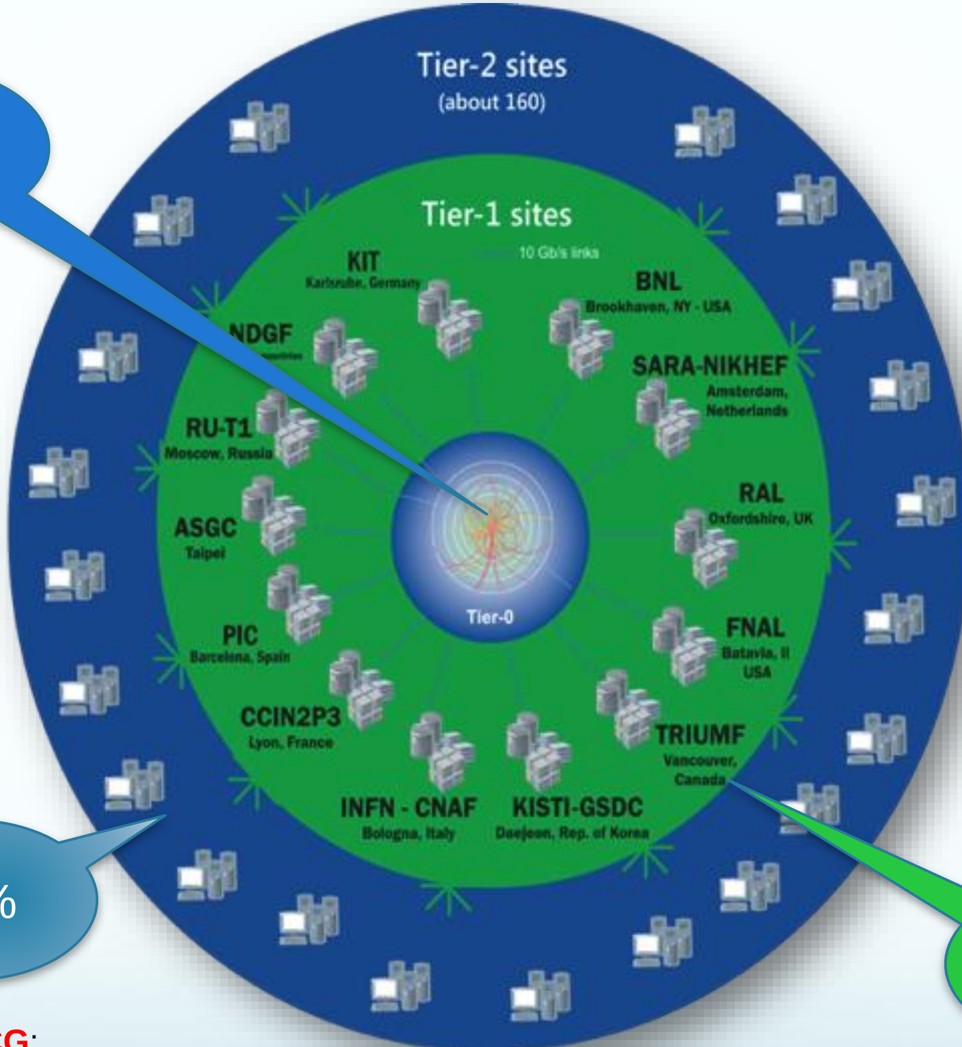
Distributed Computing

LHC Computing. The Worldwide LHC Computing Grid

Tier-0
(CERN and Hungary):
data recording,
reconstruction and
distribution

Tier-1: permanent
storage, re-processing,
analysis

Tier-2: Simulation,
end-user analysis



~170 sites,
42 countries

~750k CPU cores

~1 EB of storage

> 3 million jobs/day

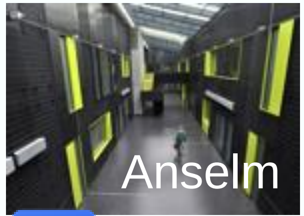
10-100 Gb links

WLCG:
An International collaboration to distribute and analyse LHC data

Integrates computer centres worldwide that provide computing and storage resource into a single infrastructure accessible by all LHC physicists

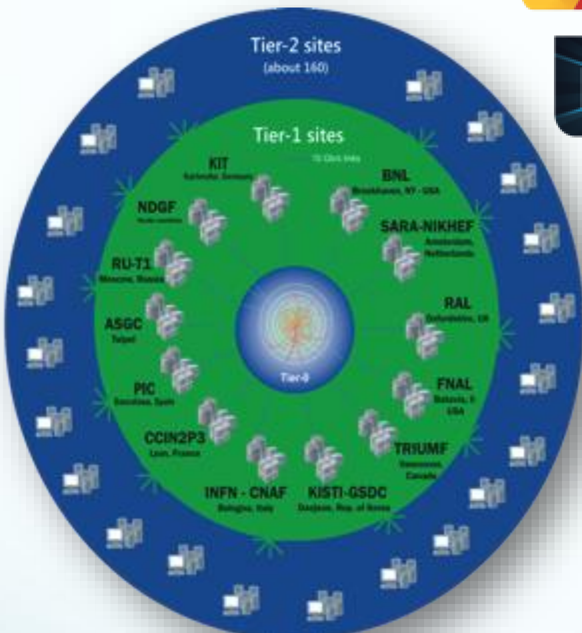
From HTC to HPC

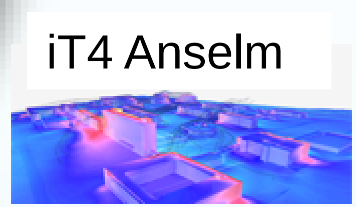
- With highly successful Run 2 (~x2 data delivered), and looking ahead to Runs 3, 4 at the LHC (2023+)
- ATLAS started looking at traditional HPC systems
  - Almost 50% of ATLAS CPU cycles used for simulation
  - HPC architectures are well suited to run simulations
  - However, they need to be integrated into production and data management systems – not standalone
- HTC/Grid + Clouds + HPC == Truly Heterogeneous and distributed computing integrated seamlessly



Google cloud computing

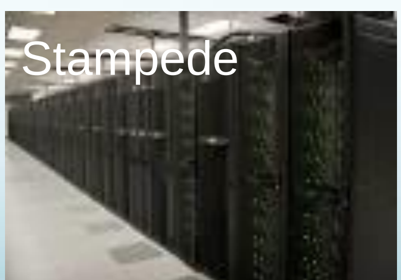
NERSC National Energy Research Scientific Computing Center



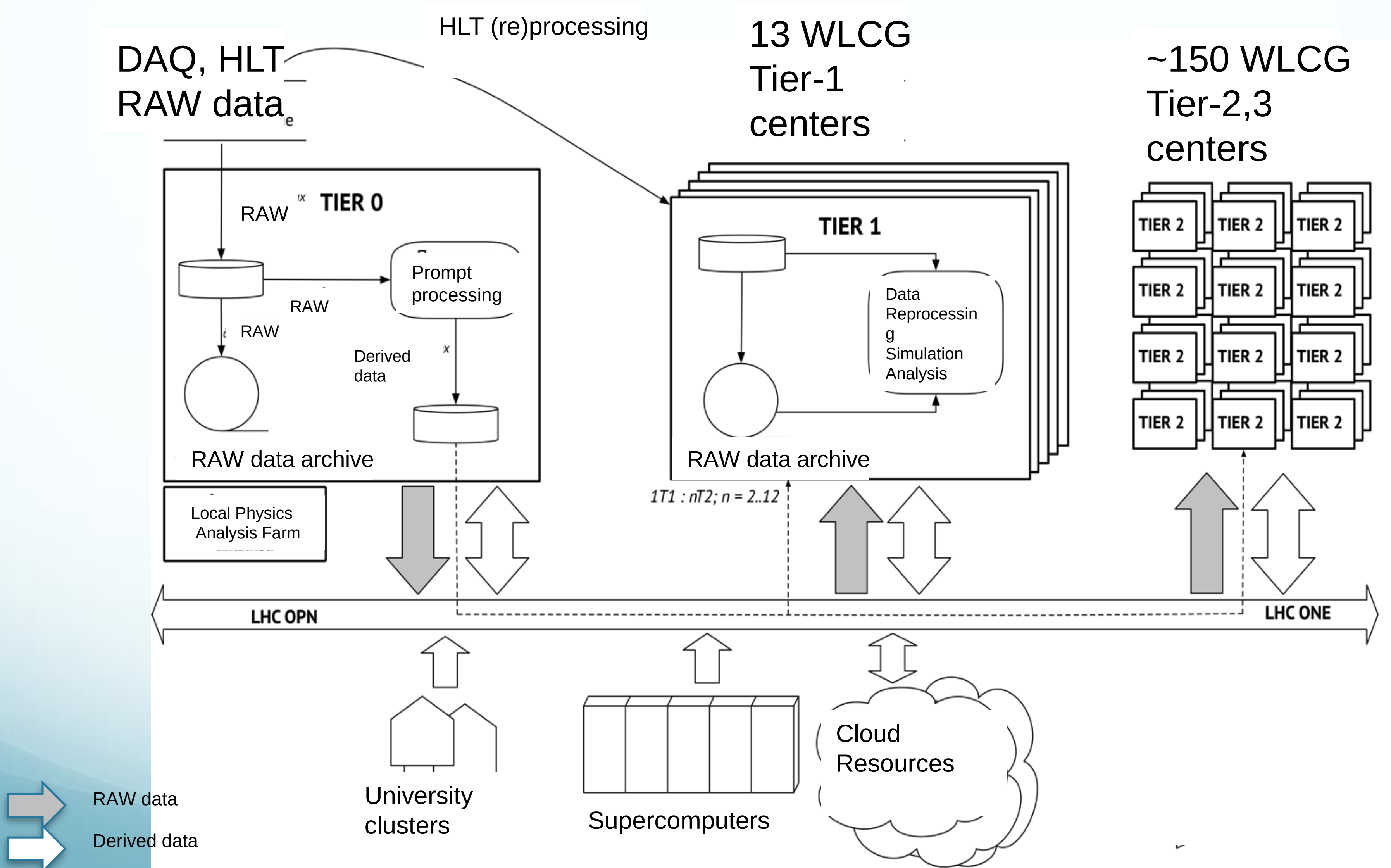


Titan System (Cray XK7)			
Peak Performance	27.1 PF 18,688 compute nodes	24.5 PF GPU	2.6 PF CPU
System memory	710 TB total memory		
Interconnect	Gemini High Speed Interconnect	3D Torus	
Storage	Lustre Filesystem	32 PB	
Archive	High-Performance Storage System (HPSS)	29 PB	
I/O Nodes	512 Service and I/O nodes		

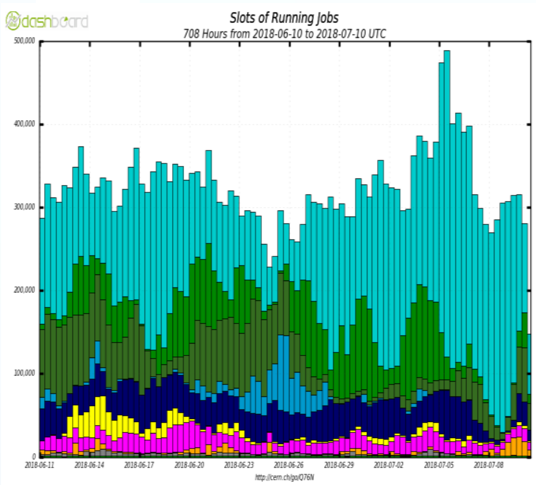
ATLAS Grid would be around #30 from Top100

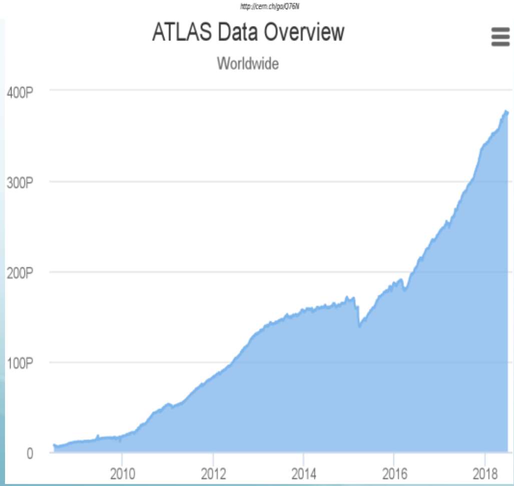


LHC Computing Model



ATLAS Distributed Computing. WMS and DDM





Workload and Workflow Management — PanDA & Production System (ProdSys2)

- Schedules and executes computational tasks
- Interacts with compute systems

Data Management — Rucio

- In charge of all experiment data
- Interacts with storage systems

Operations and Support

Operations teams run the experiment

Databases, Monitoring, Analytics, …

Workflow and Workload Management

Basic Definitions

- Request - high level layer for Production managers ('reprocess 2017 PeriodA data')
  - ProdSys2 translates request to basket of tasks or task chain
  - Chain : event generation -> simulation -> reconstruction -> derivation
- Task : group of associated jobs, it is formed according to request
  - With the same production Tag
    - Production step
    - SW release
    - May have input(s) - dataset(s)
    - Produce outputs - datasets
  - Current scale 2M tasks / year
  - Task chain
  - Task busket
- Job : basic unit of work
  - Executed on a CPU resource/slot
  - May have inputs (files)
  - Produces outputs (files)
  - Current scale 365+M jobs /year
- Pilot job
  - Lightweight execution environment to prepare Computing Element (CE), request actual payload, execute payload and clean up
- Dataset - group of files taken/produced under the same conditions
- Container - group of datasets

Task states :

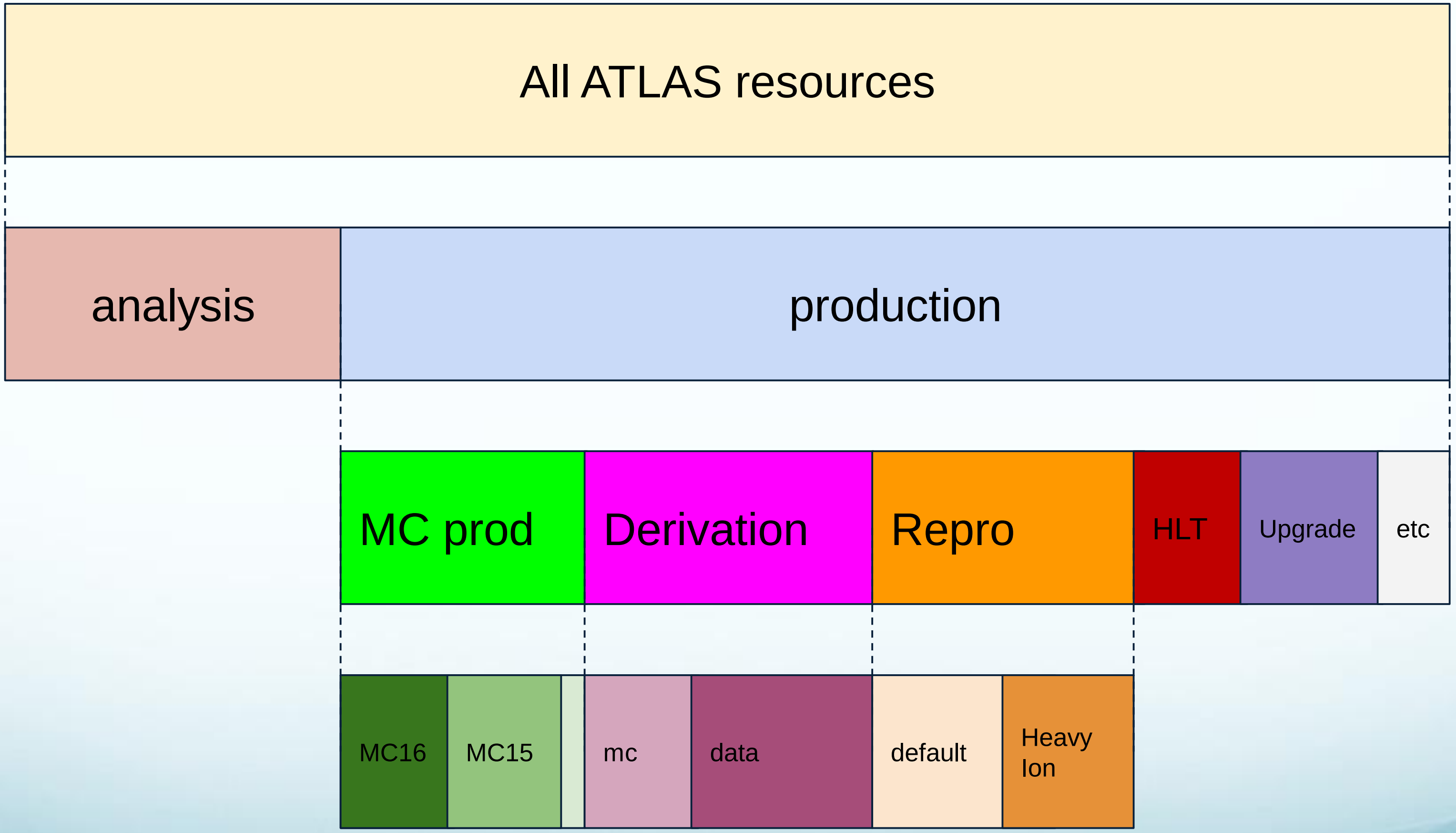
Waiting : the task information is inserted to the DEFT task table (t_production_task) and task is waiting to be processed by JEDI
Registered : the task information is inserted to the JEDI task tables
Assigning : the task brokerage is assigning the task to a cloud
Submitting : the task is running scouts jobs
Running : the task is running jobs
Exhausted: task can go to the exhausted state from running if all attempts have been used, but not all jobs are done, usually it means that some task parameters (for instance, ram count) should be tuned. From the exhausted state task can go to final state : finished, aborted, failed or number of attempts can be manually increased and task can go to running state.
Done : all jobs are successfully finished
Finished : some inputs of task are not finished (or not executed), but task is considered as finished
Broken : task cannot be executed, task definition has problems
Failed : task failed in execution time and it should be aborted
Aborted : the task is killed, all outputs will be erased
Obsolete :the task is obsolete and all outputs will be erased

Main ATLAS workflows

- Monte-Carlo Production (months)
  - Organized in campaigns
- Data (Re)processing (weeks)
  - Organized in campaigns
- High Level Trigger Processing (<24h)
- Tier-0 spill-over (24h-36h)
- SW Validation (days)
- Physics groups production (week)
- Derivation production in trains
- Open-ended production
- Users Analysis (asap)

Resources are shared according to scientific goals between ATLAS & Physics Groups & Physicists

Resource allocation



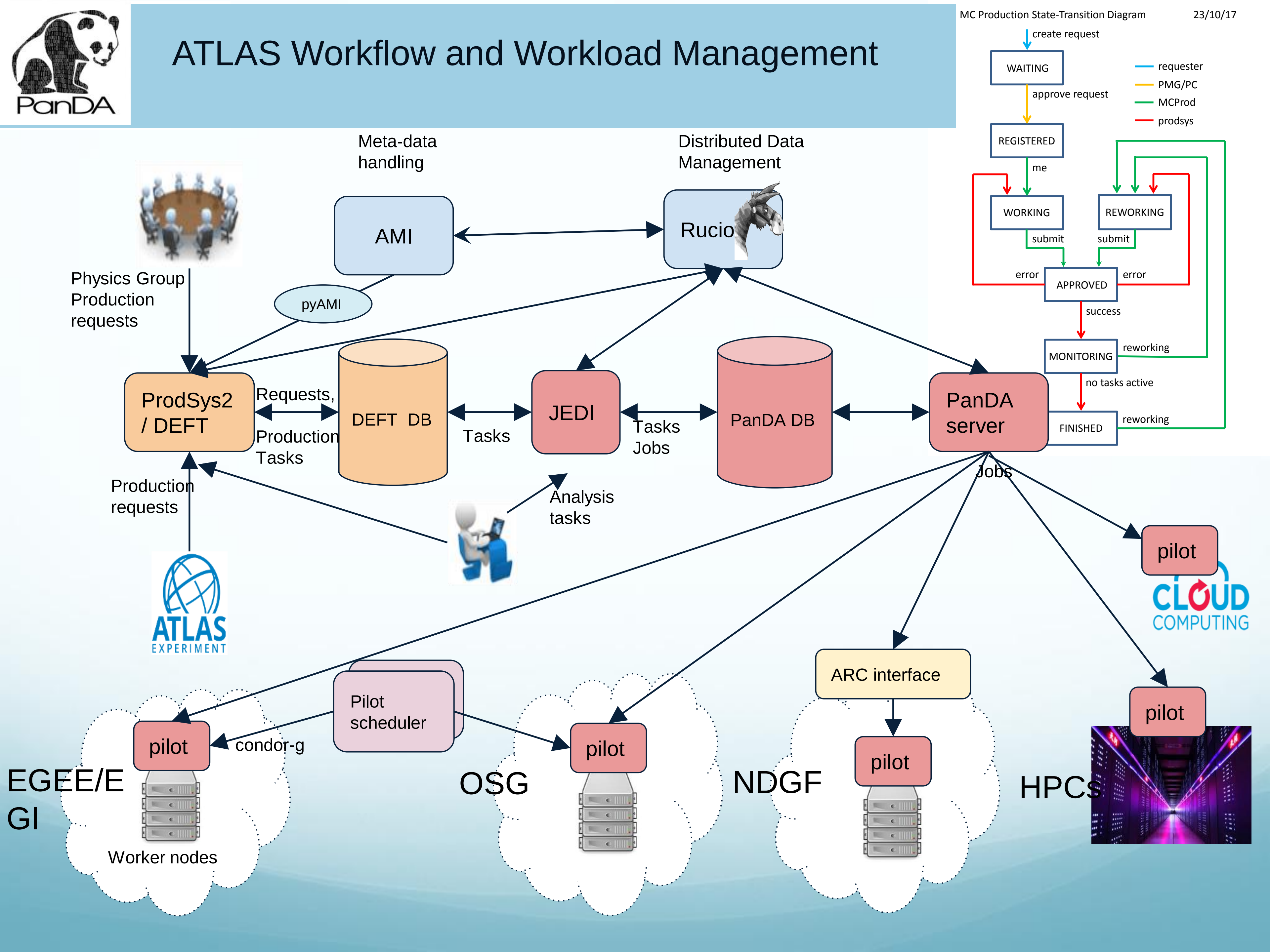
Resource allocation

- Static partitioning between Production and Analysis
- Production
  - Dynamic partitioning by Global share mechanism
  - Shares and allocation defined based on physics needs
    - E.g., large allocation to physics groups before a conference
- Analysis
  - Normal user analysis using personal certificate and group analysis using group production role
  - The same allocation for all users and groups
  - No priority boost for groups by default
  - Higher priorities to a user and/or group if requested by Physics Coordination

Global Shares: hierarchical fair share mechanism

- Used to split processing resources on the grid between activities
  - E.g to allocate 20% of overall CPUs to data reprocessing
- Measured in currently used HS06 (=*ncores x corepower*)
  - It is not a quota system, i.e. we do not keep the history
- Shares are nestable: they will use the sibling's unused share
- Shares are assigned to a task at creation time and propagated to jobs
  - Rules based on prodsourcelabel, working group, campaign and processingtype
- They are restricted within certain limits and can not always be fully satisfied
  - We are working on improving the system and reduce the boundaries

ATLAS Workflow and Workload Management

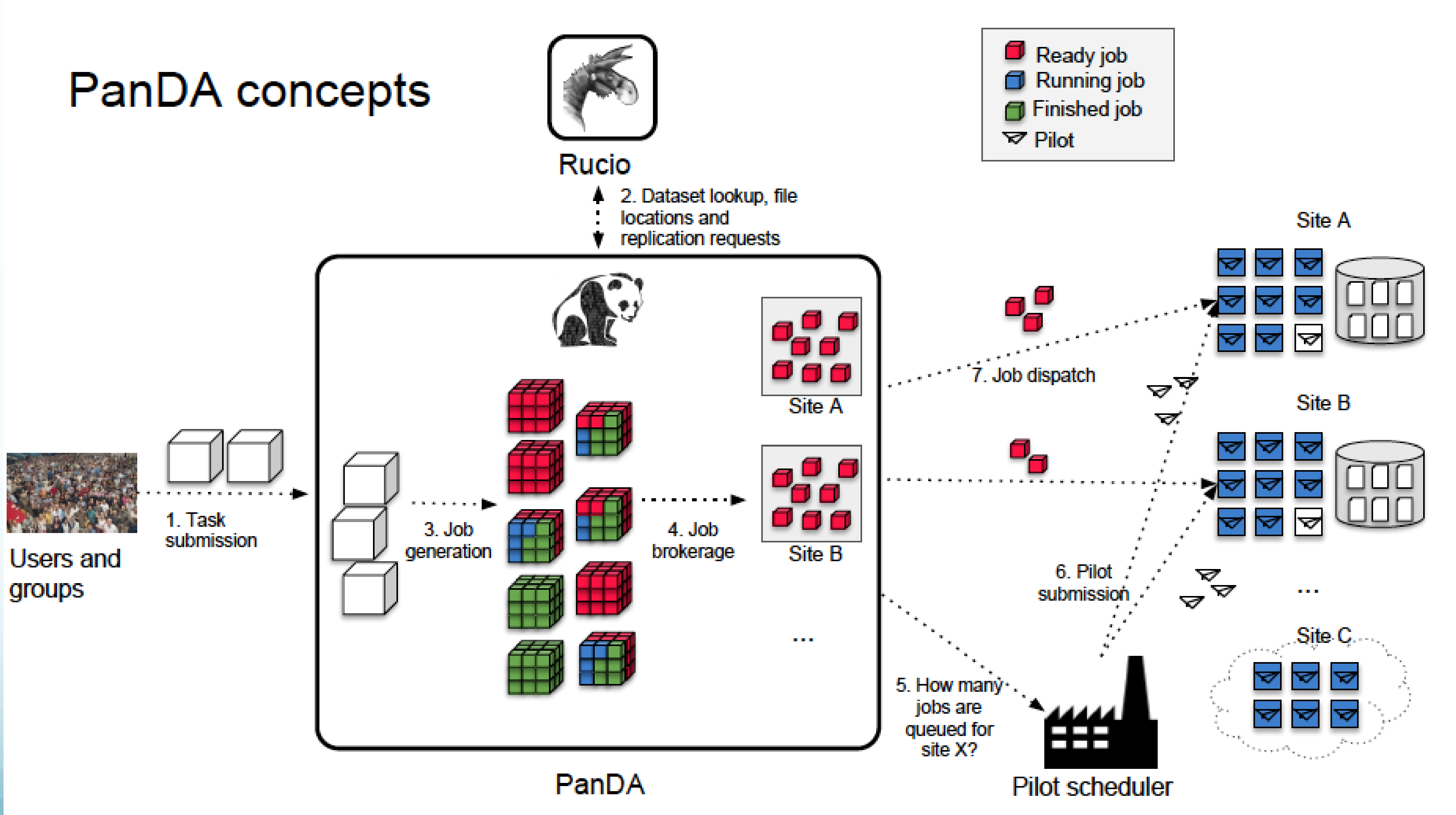


PanDA Workload Management System

- The PanDA workload management system was developed for the ATLAS experiment at the Large Hadron Collider. A new approach to distributed computing
  - A huge hierarchy of computing centers and opportunistic resources working together
  - Main challenge – how to provide efficient automated performance
  - Auxiliary challenge – make resources easily accessible to all users

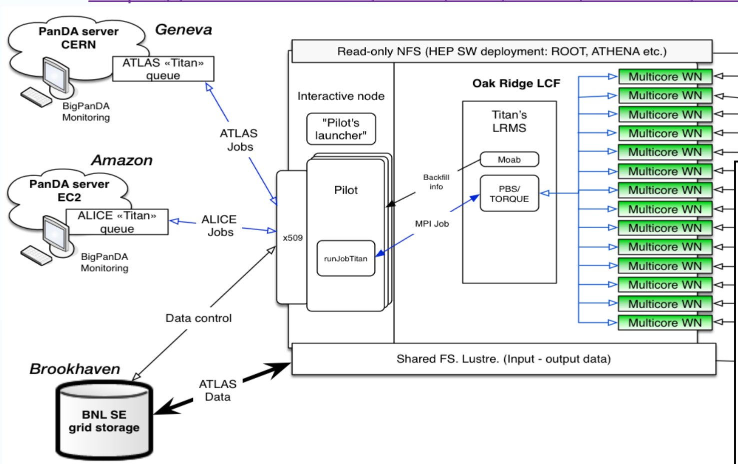
- Core ideas :
  - Make hundreds of distributed sites appear as local
    - Provide a central queue for users – similar to local batch systems
  - Reduce site related errors and reduce latency
    - Build a pilot job system – late transfer of user payloads
    - Crucial for distributed infrastructure maintained by local experts
  - Hide middleware while supporting diversity and evolution
    - PanDA interacts with middleware – users see high level workflow
  - Hide variations in infrastructure
    - PanDA presents uniform 'job' slots to user (with minimal sub-types)
    - Easy to integrate grid sites, clouds, HPC sites …
  - Data processing, MC Production and Physics Analysis users see same PanDA system
    - Same set of distributed resources available to all users
    - Highly flexible – instantaneous control of global priorities by experiment

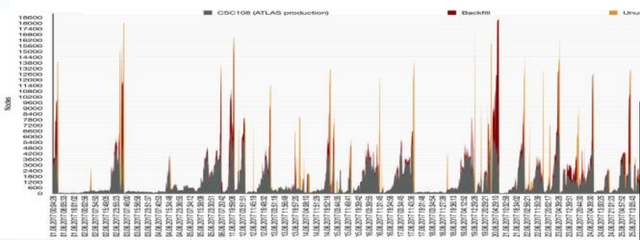
PanDA Workload Management System



PanDA. Production and Distributed Analysis Workload Management System

https://twiki.cern.ch/twiki/bin/view/PanDA/PanDA





Global ATLAS operations

Up to ~800k concurrent jobs
25-30M jobs/month
at >250 sites
~1400 ATLAS users

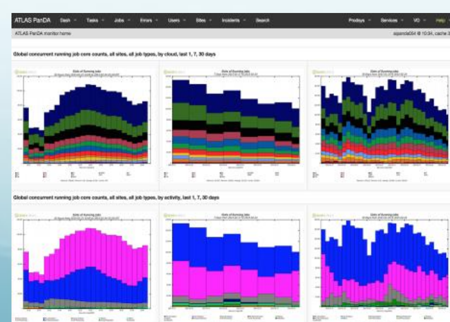
First exascale workload manager in HENP
1.3+ Exabytes processed in 2014 and in 2016-2018
Exascale scientific data processing today

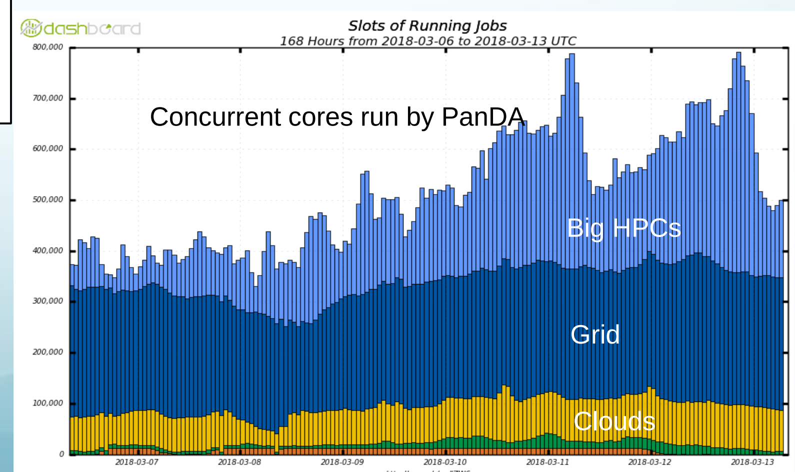
PanDA Brief Story

2005: Initiated for US ATLAS (BNL and UTA)
2006: Support for analysis
2008: Adopted ATLAS-wide
2009: First use beyond ATLAS
2011: Dynamic data caching based on usage and demand
2012: ASCR/HEP BigPanDA project
2014: Network-aware brokerage
2014 : Job Execution and Definition I/F (JEDI) adds complex task management and fine grained dynamic job management
2014: JEDI- based Event Service
2014:megaPanDA project supported by RF Ministry of Science and Education
2015: New ATLAS Production System, based on PanDA/JEDI
2015 :Manage Heterogeneous Computing Resources
2016: DOE ASCR BigPanDA@Titan project
2016:PanDA for bioinformatics
2017:COMPASS adopted PanDA , NICA (JINR)
PanDA beyond HEP : BlueBrain, IceCube, LQCD

BigPanDA Monitor

http://bigpanda.cern.ch/



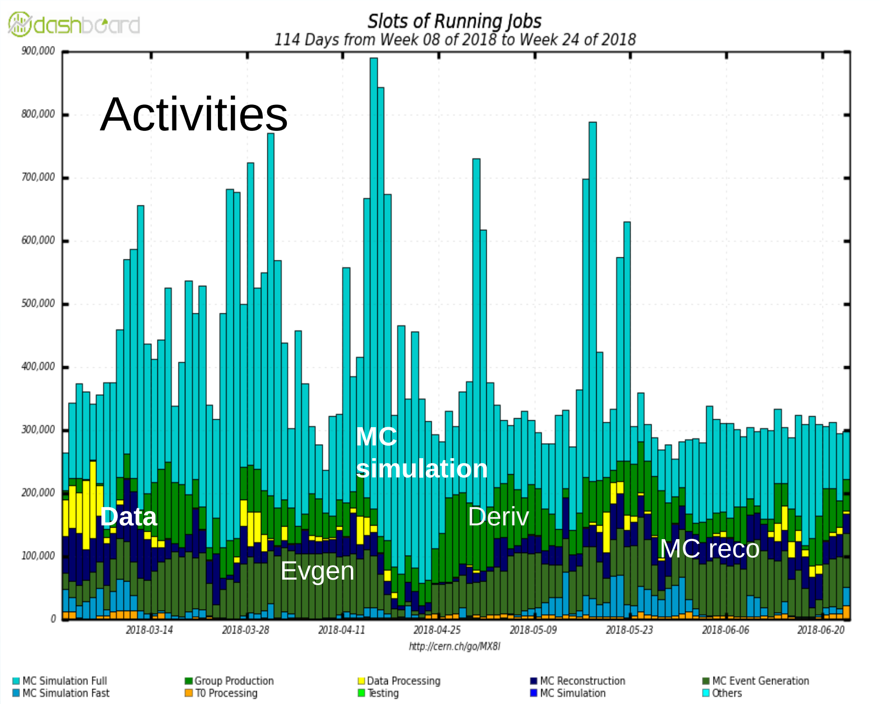


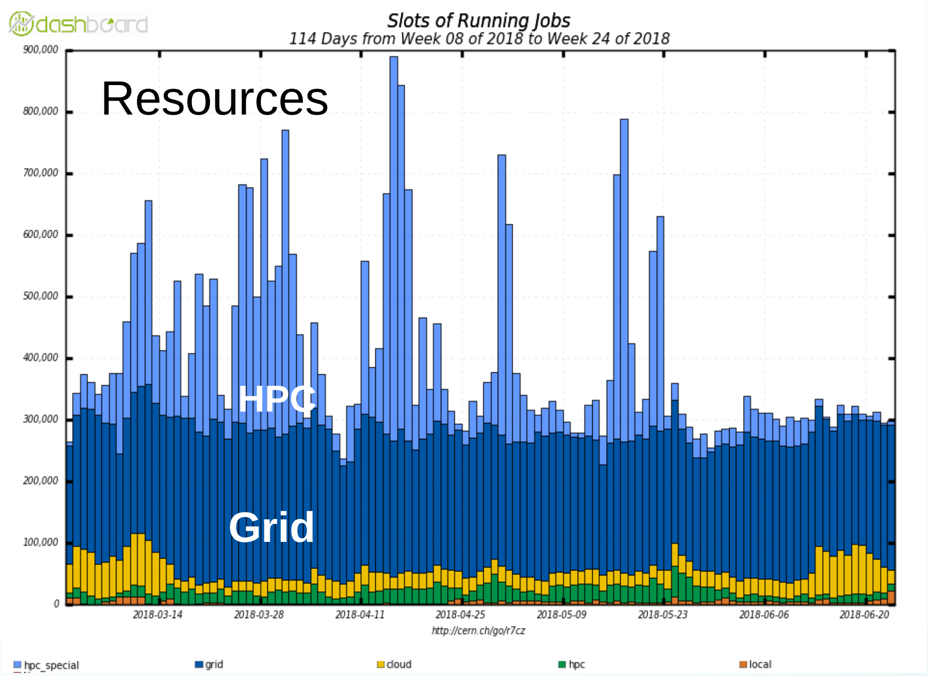
Paradigm Shift in HEP Computing

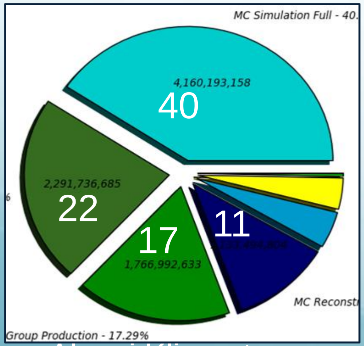
- New ideas from PanDA
  - Distributed resources are seamlessly integrated
  - All users have access to resources worldwide through a single submission system
  - Uniform fair share, priorities and policies allow efficient management of resources
  - Automation, error handling, and other features in PanDA improve user experience
  - All users have access to same resources
- Old HEP paradigm
  - Distributed resources are independent entities
  - Groups of users utilize specific resources (whether locally or remotely)
  - Fair shares, priorities and policies are managed locally, for each resource
  - Uneven user experience at different sites, based on local support and experience
  - Privileged users have access to special resources

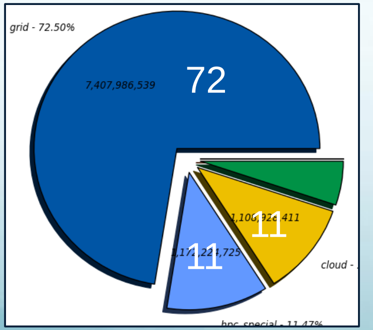
The story of PanDA has parallel in industry – the growth of Cloud Computing

ATLAS Data Processing and Simulation. March - June 2018







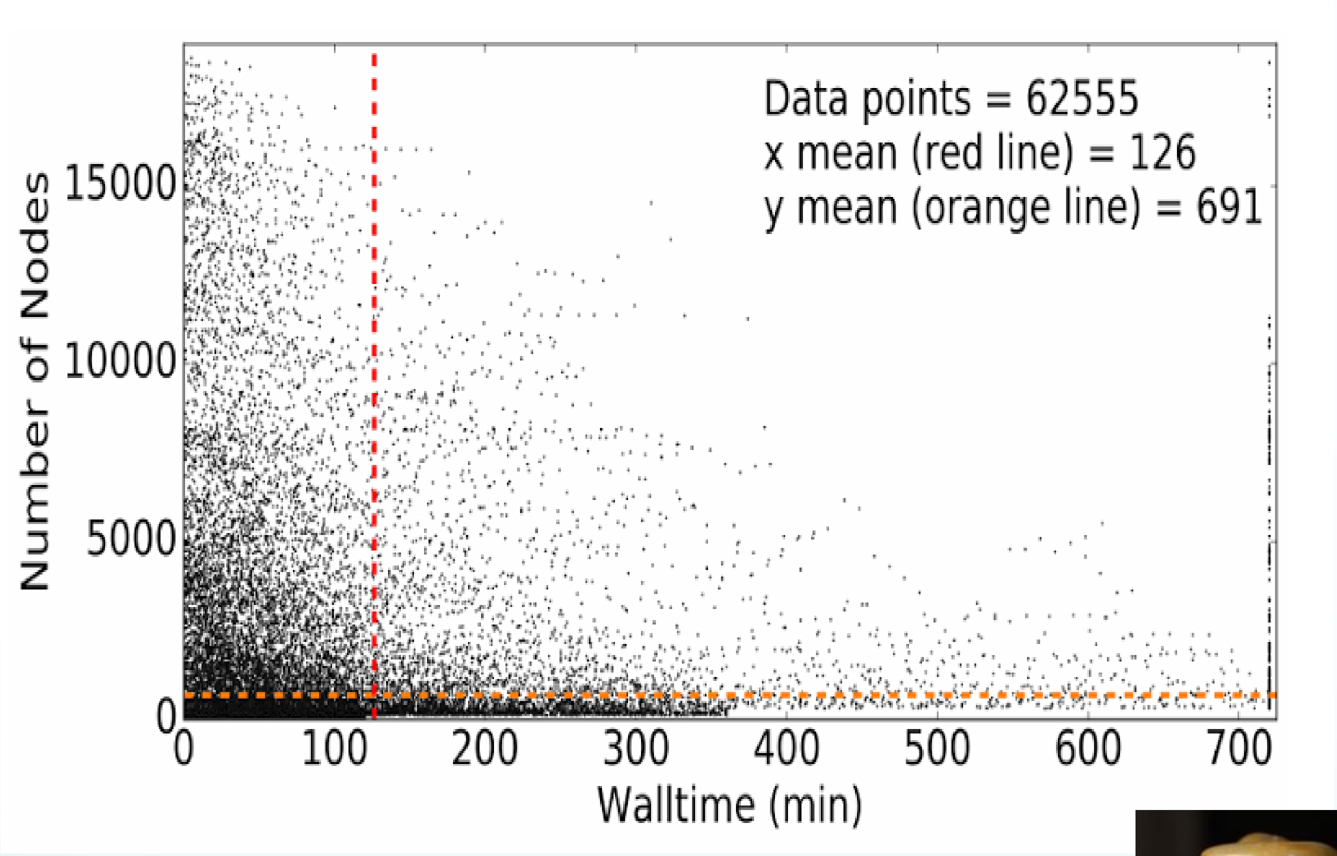


- Full utilization with smooth ops, ~300-350k cores, peaking to ~1M with HPCs
- Moving >1 PB, >20 GB/s, 1.5-2M files per day

BigPanDA Workflow Management on Titan for High Energy and Nuclear Physics and for Future Extreme Scale Scientific Applications

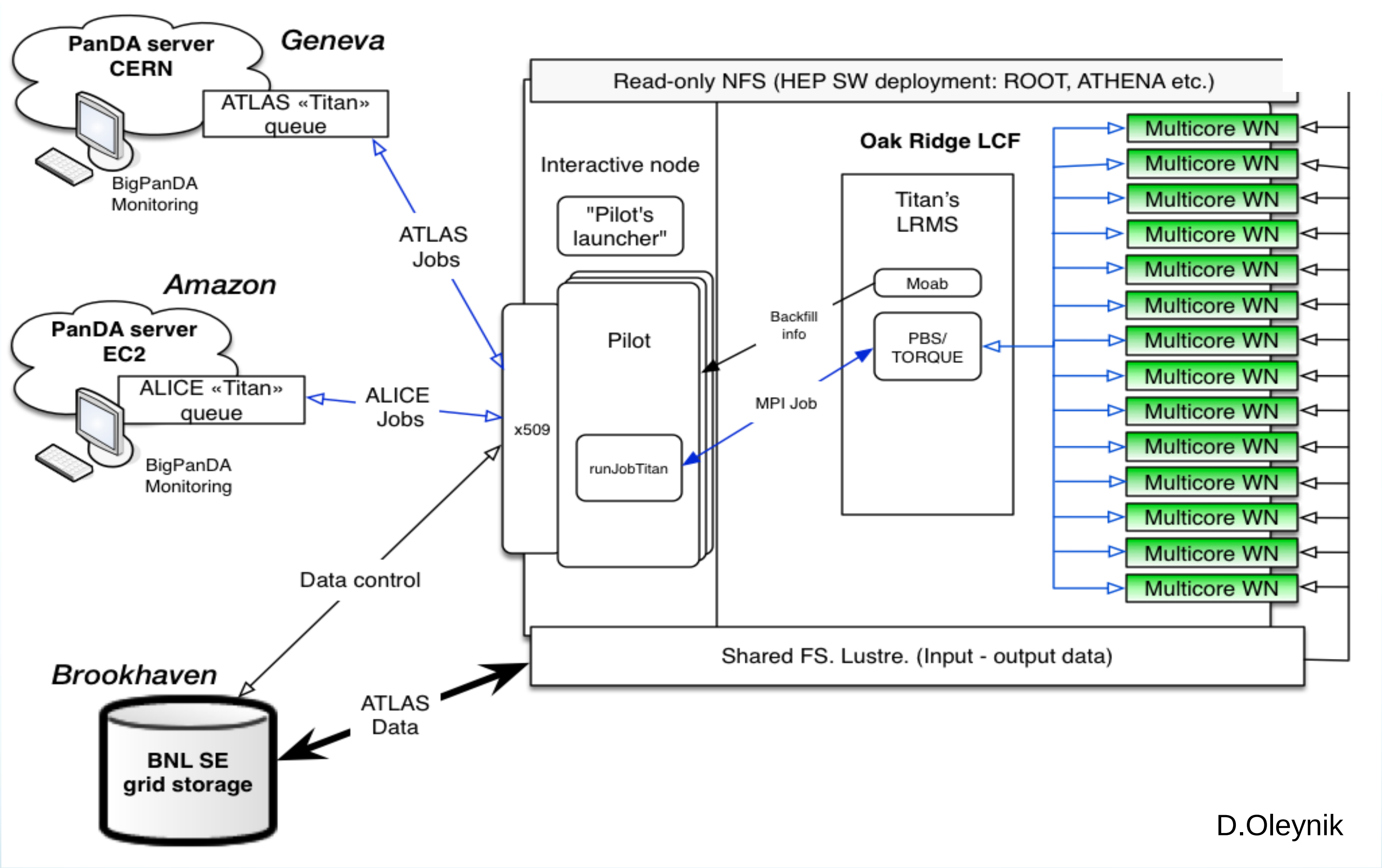
- BigPanDA project: an extension of PanDA beyond the grid and HEP as well as use of PanDA for projects and experiments beyond ATLAS and HEP
- A DOE ASCR and HEP funded project since 2012; a collaboration between BNL, UTA, ORNL and Rutgers University since 2015 (BigPanDA++)

OLCF. Understanding Backfill Slot Availability



- Mean Backfill availability: 691 worker nodes for 126 minutes.
- Up to 15K nodes for 30-100 minutes
- Large margin of optimization

OLCF Titan Integration with PanDA

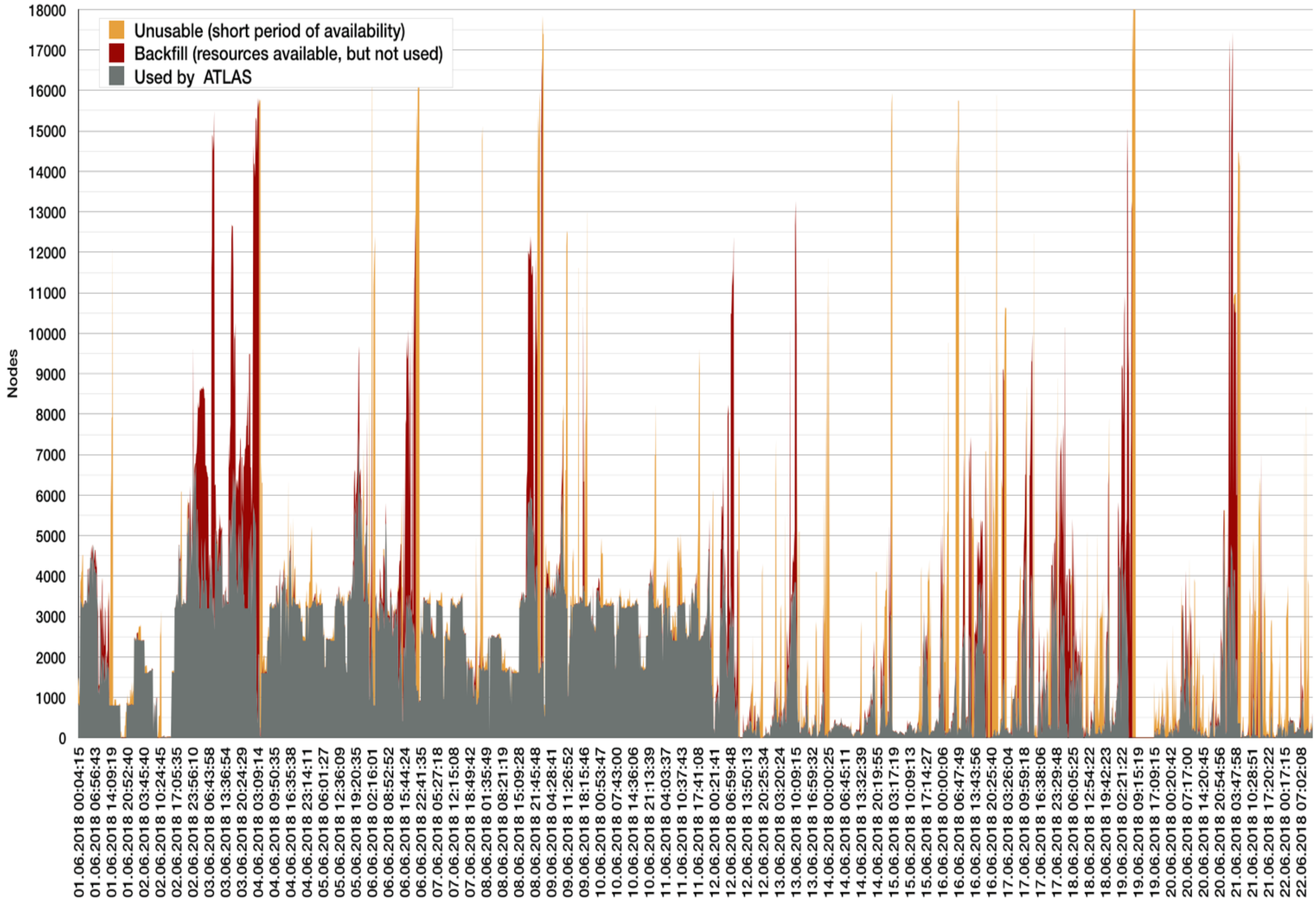


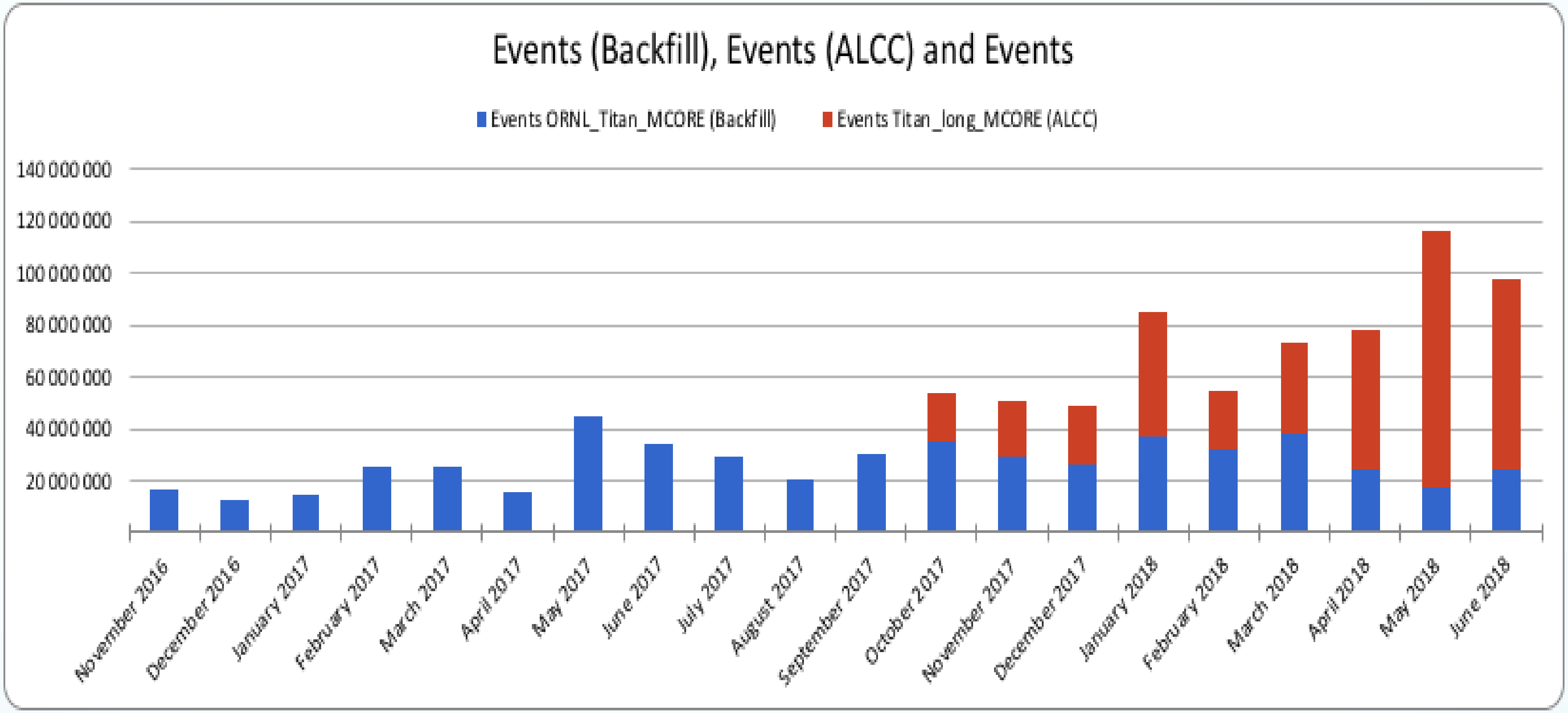
First large scale HPC integrated into ATLAS distributed computing through the BigPanDA project funded by DOE-ASCR
Team leaders: A.Klimentov (BNL), J. Wells (ORNL), S.Jha (Rutgers U), K.De (U of Texas-Arlington)
300 million TITAN core hours in past 12 months, both backfill usage and ALCC allocation

D. Oleynik, S. Panitkin, M. Turilli, A. Angius, S. Oral, K. De, A. Klimentov, J. C. Wells and S. Jha, "High-Throughput Computing on High-Performance Platforms: A Case Study", IEEE e-Science (2017) available as: https://arxiv.org/abs/1704.00978

ATLAS@OLCF: Batch Queue Submission & Active Backfill

- Backfill utilization in 1 June through 22 June 2018, 10-min data frequency





- Since Nov. 2016, 508 million ATLAS events computed via backfill
- Since Oct. 2017, 395 million TLAS events computed via ”normal” batch queue
  - Increases in batch queue event generation beginning in Feb. 2018 show the impact of Harvester

HPC: internal scheduling

- HPC allocations usually awarded by n million node-hours over a period
- HPC internal scheduling policies optimize the usage of their infrastructures while honouring users' fair shares
  - Usually only multi-node slots
  - Large requests often prioritized
  - Max walltime can depend on the size of the request
  - Backfill opportunities outside your allocation
    - Fill out leftovers with limitation on running time
- However ATLAS workloads are loosely coupled (pleasantly parallel)
  - Typically each job needs 1-16 cores, 2-4 GB RAM/core
  - Runs over a file with few hundred events over several hours

HPC: data management

- Not always storage element present at HPC
- HPCs with external I/O can use a remote grid storage element
- Restrictive HPCs require data pre-placement to local storage or shared filesystem
  - Download
  - 3rd party transfers managed by Rucio
    - FTS
    - Globus Online
  - Difficult to converge on one solution

BigPanDA. PanDA beyond High Energy and Nuclear Physics

- PanDA designed to support MultiVO
  - Different VO (Experiments) may share same PanDA server instance
  - Server and Pilot plugins allows to tune pre/post-processing VO specific procedures
  - Monitoring is not VO specific

- If VO requires high scalability (hundreds of thousands jobs per day, on wide range of resources) dedicated instance may be deployed

- Beyond HENP
  - Biology / Genomics: Center for Bioenergy Innovation at ORNL
  - Molecular Dynamics: Prof. K. Nam (U. Texas-Arlington)
  - IceCube Experiment
  - Blue Brain Project (BBP), EPFL
  - LSST (Large Synoptic Survey Telescope) project/DESC collaboration
  - LQCD, US LQCD Project
  - nEDM (neutron Electric Dipole Moment Experiment), ORNL

Data Management

Data Management Tools

- At time of inception, no global/commercial solution for the distributed computing available for our 'Big Data' handling
    - A data intensive instrument which generates unprecedented data volumes
    - Facilities are distributed at multiple locations under different administrative domains
    - Data is produced at many locations where it is neither stored, nor analyzed by researchers nor archived
- ATLAS developed its own tools
    - The first implementation of the data management system was Don Quijote 2 (DQ2)
    - In production from 2006 : Originally designed as a transfer system
    - 2007-2013: Many new features added during LHC Run-1

Data Management. Rucio

EL INGENIOSO
HIDALGO DON QVI-
XOTE DE LA MANCHA,
Compueſto por Miguel de Ceruantes Saauedra.
DIRIGIDO AL DVQVE DE BEIAR,
Marques de Gibraleon, Conde de Benalcaçar, y Baña-
res, Vizconde de la Puebla de Alcozer, Señor de
las villas de Capilla, Curiel, y
Burguillos.

Año, 1605.

CON PRIVILEGIO,
EN MADRID Por Iuan de la Cueſta.
Vendeſe en caſa de Franciſco de Robles, librero del Rey nro ſeñor.

THE DON AND THE WINDMILLS.

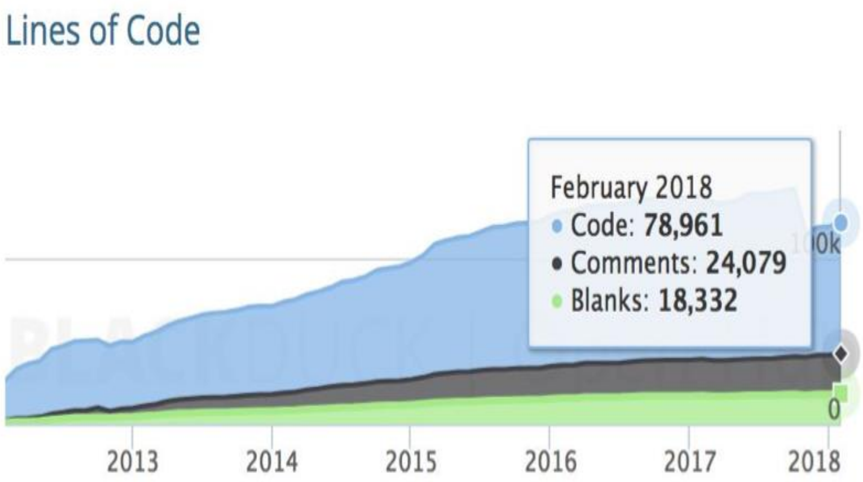
Rucio workshop 2018 : https://indico.cern.ch/event/676472/

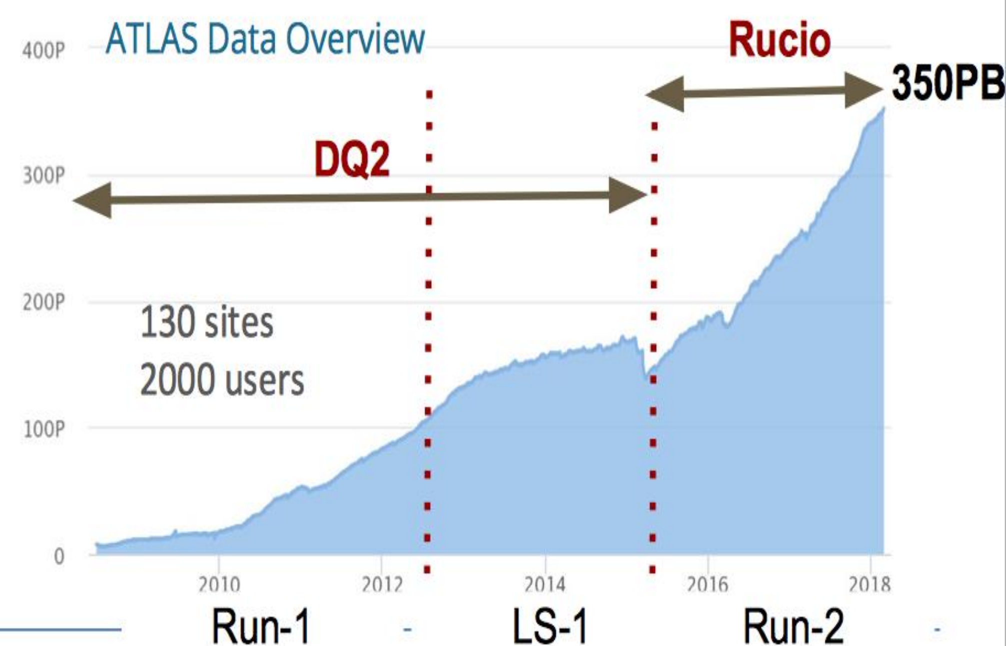
Rucio in a Nutshell

- Rucio provides a complete and generic scientific data management service
  - Designed with more than 10 years of operational experience in large-scale data management!
- Rucio manages multi-location data in a heterogeneous distributed environment
  - Creation, location, transfer, and deletion of replicas of data
  - Orchestration according to both low-level and high-level driven data management policies (usage policies, access control, and data lifetime)
  - Interfaces with workflow management systems
  - Supports a rich set of advanced features, use cases, and requirements
  - Large-scale and repetitive operational tasks can be automated

Rucio Development and Commissioning

- Long initial process:
  - 2012: User surveys, technical studies & design phase ~1 year
  - 2012-2014: Initial development ~2 years
  - 2015: Commissioning & gradual migration from predecessor system DQ2 ~1 year





The Rucio data management system

SCIENTIFIC DATA MANAGEMENT RUCIO

https://rucio.cern.ch/

Fact check

- FOSS Apache Licensed
- Python powered
- Oracle/MariaDB/PostgreSQL
- Component-based
- REST/JSON interface and API
- Built for heterogeneous scenarios
- Horizontally scalable
- Multi-Experiment proven

Tailored to complex science workflows

Global namespace to federate across different storage systems

Control & accounting of data and users

Declarative data management with policies and rules

Transfer orchestration with priorities, shares and activities

Popularity-based replication, caching and deletion

Events & messages for synchronisation with other tools

Consistency & repair of broken and missing data

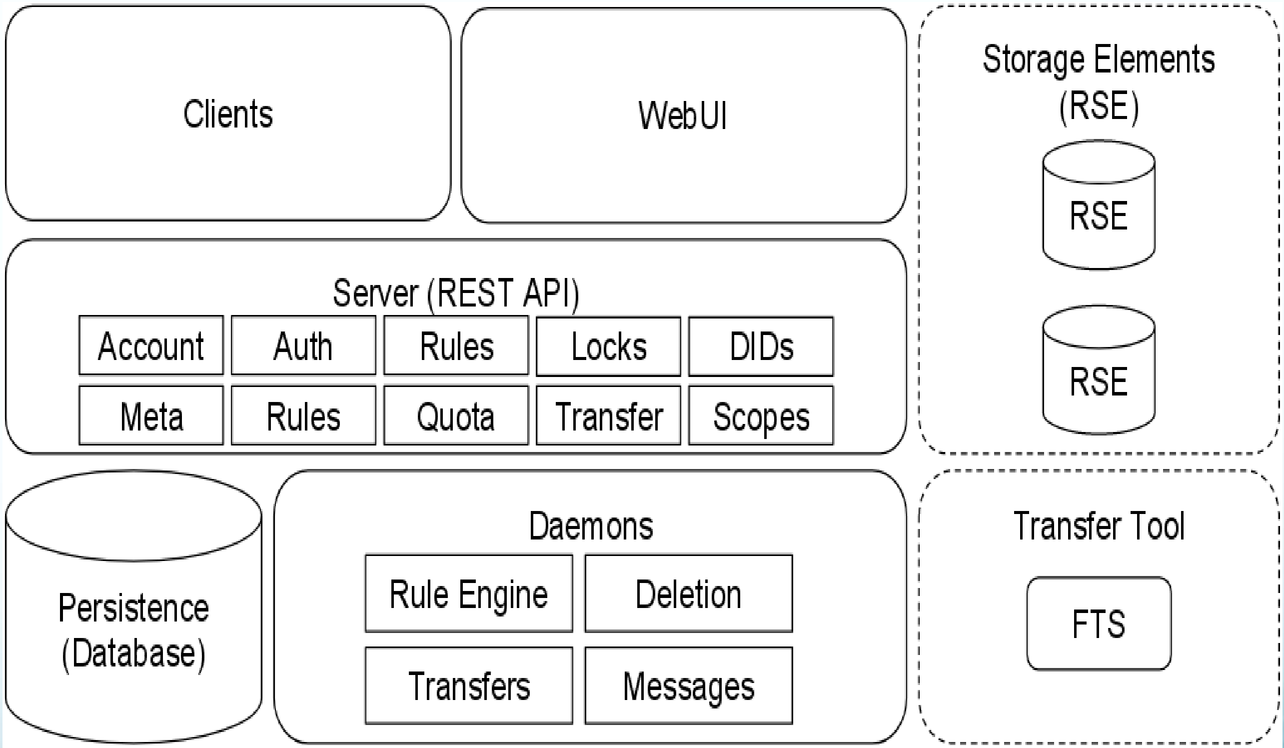
and much more …

The Rucio data management system

https://rucio.cern.ch/

Fact check

- FOSS Apache Licensed
- Python powered
- Oracle/MariaDB/PostgreSQL
- Component-based
- REST/JSON interface and API
- Built for heterogeneous scenarios
- Horizontally scalable
- Multi-Experiment proven



The Rucio data management system

https://rucio.cern.ch/

Science workflows

Global namespace

Control & accounting

Policies & rules

Transfer orchestration

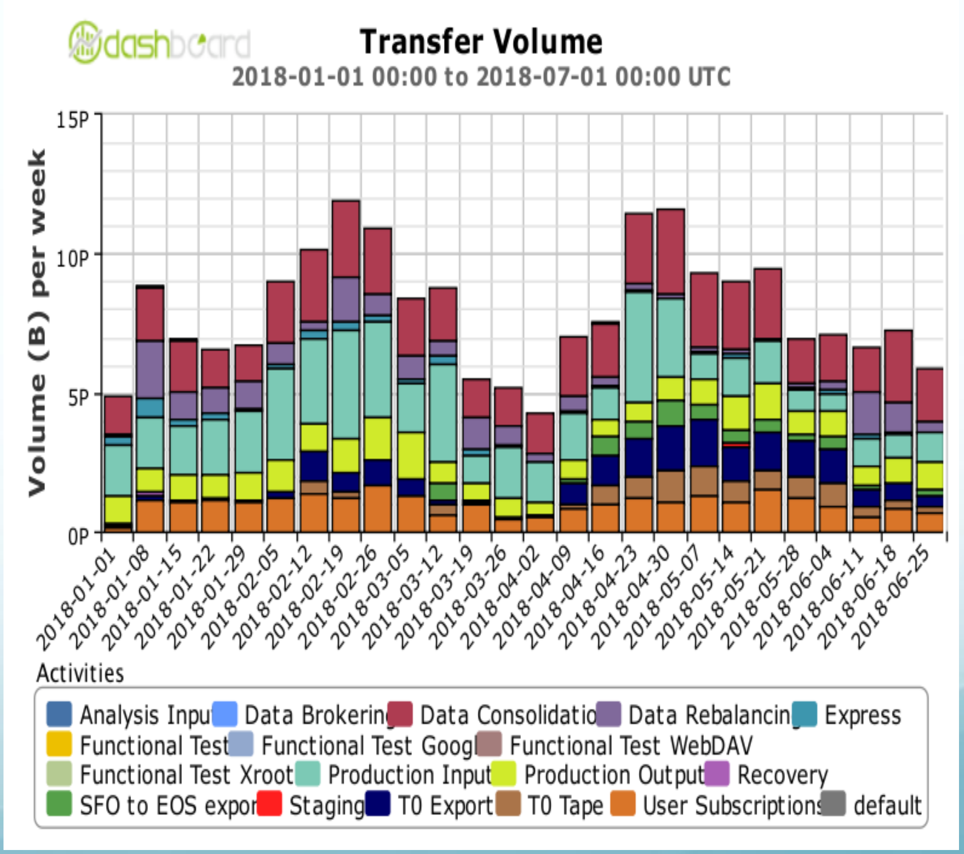
Caching & deletion

Events & messages

Consistency & repair

and much more …

Orchestrated storage-to-storage activities



The Rucio data management system

https://rucio.cern.ch/

Science workflows

Global namespace

Control & accounting

Policies & rules

Transfer orchestration

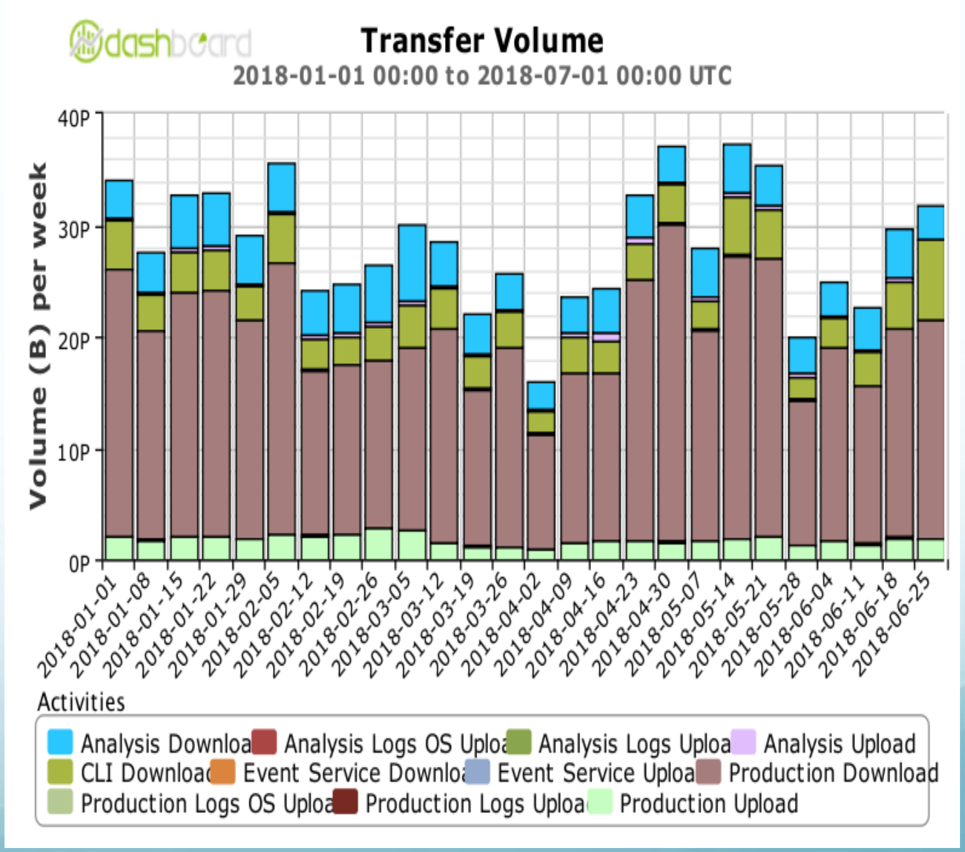
Caching & deletion

Events & messages

Consistency & repair

and much more …

"Chaotic" user access / Job IO



NEXT '18

Lessons Learned

- WMS and DDM are designed by and serve the physics community
- New features are driven by experiment operational needs
- Computing model and computing landscape in general has changed
  - Tiers hierarchy relaxed (~not exist)
  - Computing resources are becoming heterogeneous
    - Dedicated (grid) sites, HPCs, commercial and academic clouds …
    - HPCs and clouds are successfully integrated for Run 2/3
    - The mix of site capabilities and architectures
      - The mix will change with time - though all will be needed
- There are several systems with very well defined roles which are integrated for distributed computing : Information system (AGIS), DDM (Rucio), WMS (ProdSys2/PanDA), meta-data (AMI), and middleware (HTCondor, Globus…). We managed to have a good integration of all of them in ATLAS.
  - Combine all functionalities in one system or separate them between systems ?
    - Catalogs, layers, ….flexibility to add new features and to evaluate new technologies
- Monitoring and accounting are key components of Distributed SW
- Errors handling
- Scalability
  - WMS
  - Database technology
  - Monitoring
- WMS functionality is important as scalability
- Edge service is (should) be an additional layer to serve all heterogeneous resources

Future Development

Revised WMS architecture: PanDA Server - Harvester - Pilot

Harvester as edge service, capable of integrating heterogeneous resources through plugin interface

HPC

Run on edge node of each HPC, or potentially centrally if HPC provides a CE

- Data pre-placement and output transfer through download/upload or 3rd party transfer
- Job management
  - Combine jobs into multi node submission
  - Jumbo jobs management with Yoda
- Exploited in US DOE HPC facilities and available (installed) for other HPCs

Cloud

Can run anywhere, usually centrally in shared instance

- VM lifecycle management: create, monitor and delete VMs
- Plugins existing for Google Compute Engine and Openstack

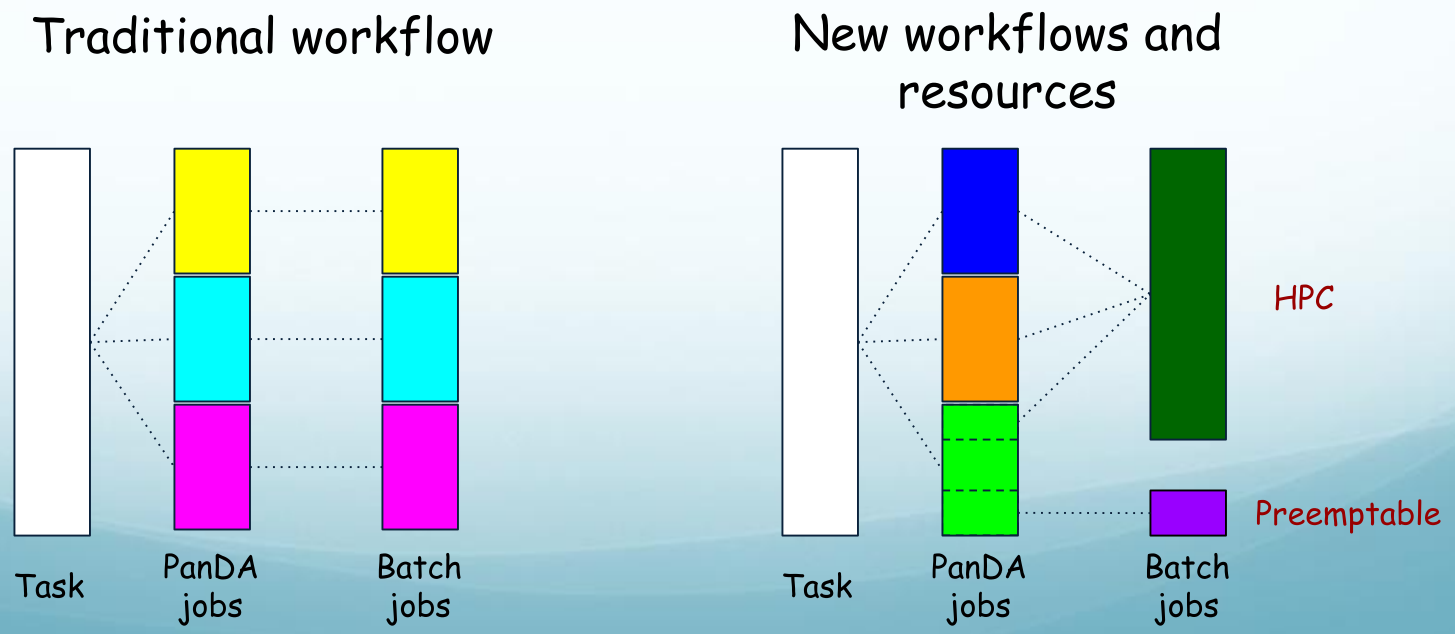
Grid

Can run anywhere, usually centrally in shared instance

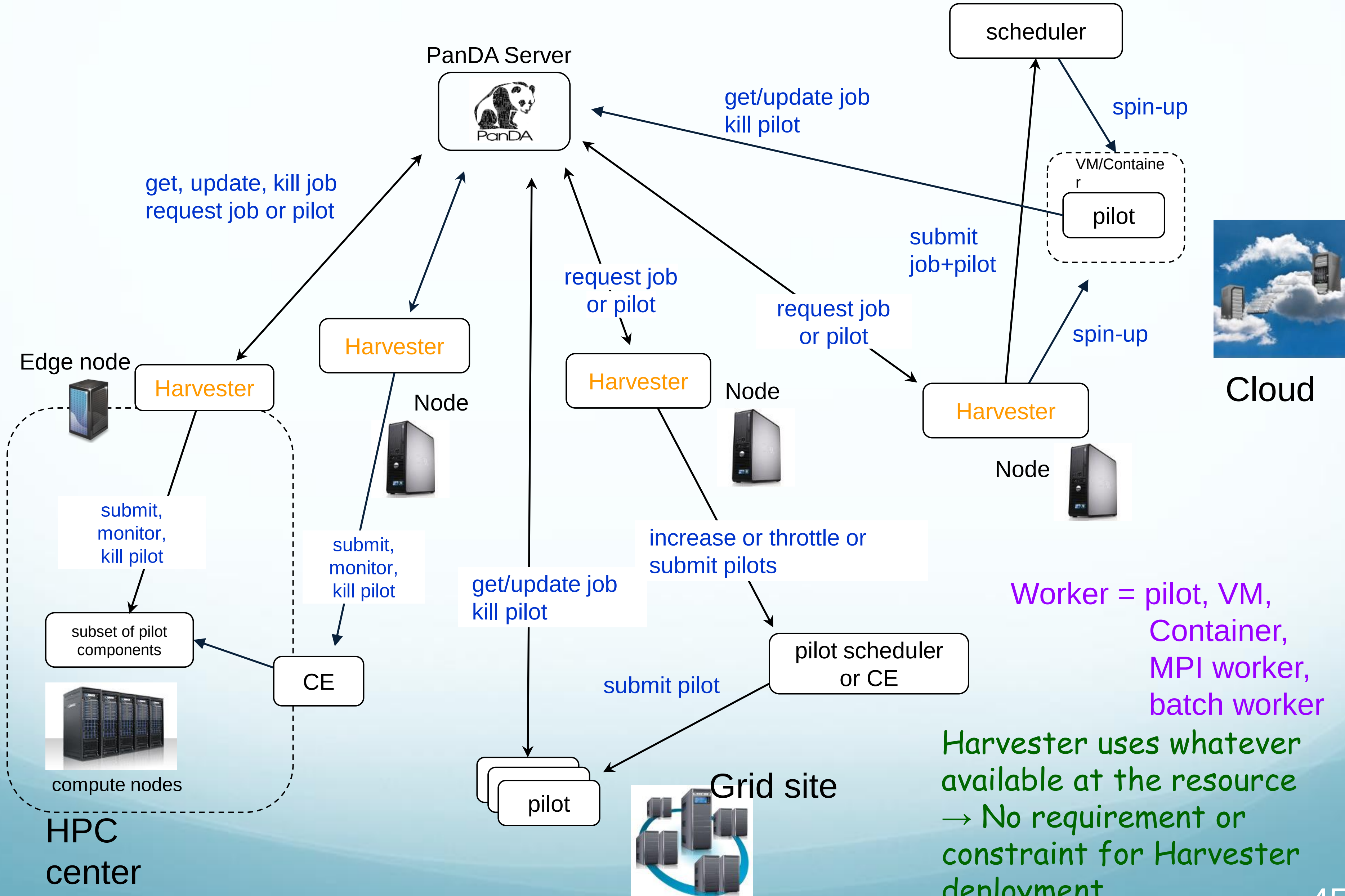
- Standard Pilot submission in different modes
  - Push/pull
  - Closer integration with PanDA server and can receive commands for e.g. Unified PanDa queues

Why Harvester

- Traditional workflow is good for WLCG grid resources since they are almost the same in terms of architecture and OS
  - One PanDA job (entity of production/PanDA system based on physics and/or processing needs) = an immutable collection of events = one batch job (entity of the batch system)
  - Pros and cons of push and pull without crucial advantages
- Not the case for emerging resources and workflows
  - MPI, preemption, fluctuation of availability, fine-grained bookkeeping, …
  - Complicated mapping among PanDA jobs, event collections, and batch jobs
- Also the Grid is well matured, but still has a room for improvement
  - Too many PanDA queues, lost-heartbeat, empty pilots, ...



Harvester in the System



Worker = pilot, VM, Container, MPI worker, batch worker

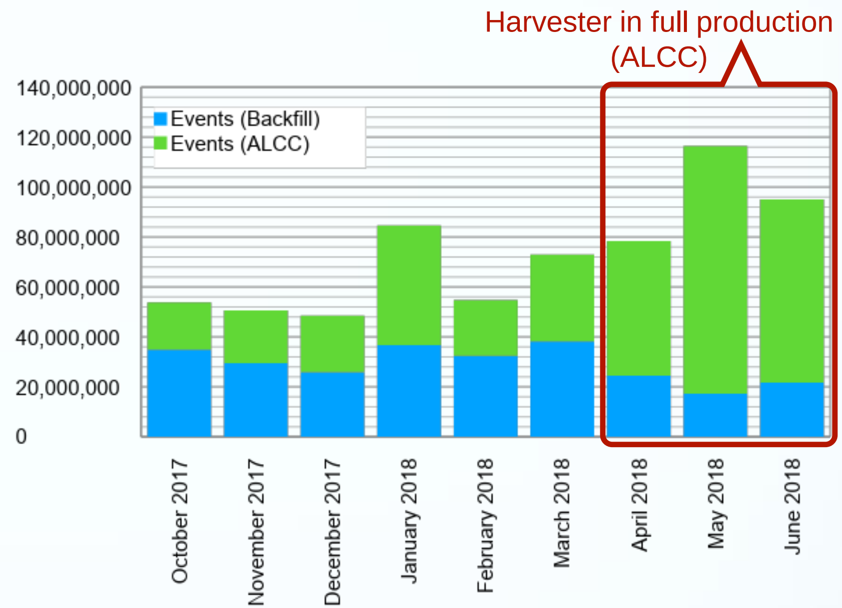
Harvester uses whatever available at the resource → No requirement or constraint for Harvester deployment

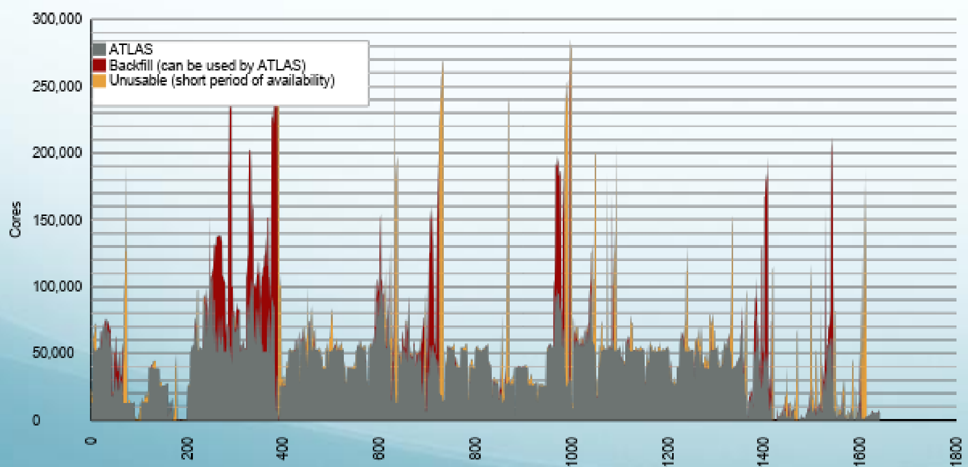
Harvester Commissioning Status

- Architecture designed and implemented
- Harvester for cloud
  - In production : CERN+Leibniz+Edinburgh resources (1.2k CPU cores
  - Work in progress : HLT farm @ LHC Point1, Google Cloud Platform
- Harvester for HPC
  - In production :
    - Theta/ALCF, Titan (OLCF)
    - ASGC (non-ATLAS Vos)
    - Cori+Edison / NERSC
    - KNL@BNL
- Harvester for Grid
  - Core SW is ready
  - Many scalability tests are already conducted in 2018
    - Harvester is currently running on ~200 Production Queues.
    - Harvester scalability is proven
    - Full migration to harvester this year
- 6 harvester instances configured and to be used for non-HEP experiments
  - Harvester instance @JLAB (LQCD)
  - Harvester instance @ORNL (nEDM, LSST)
  - Harvester and NGE (Next Generation Executor)

ATLAS production at OLCF

- Stable day by day operations
  - **550M** Events from 01.01.2018
    - ALCC allocation: **354M**
    - Backfill: **196M**
  - 20K slots AVG (120K reached MAX)
- Significant improvement with starting of using of Harvester against ALCC allocation
  - Harvester allows to serve more running jobs (supports «bigger» batch submissions)

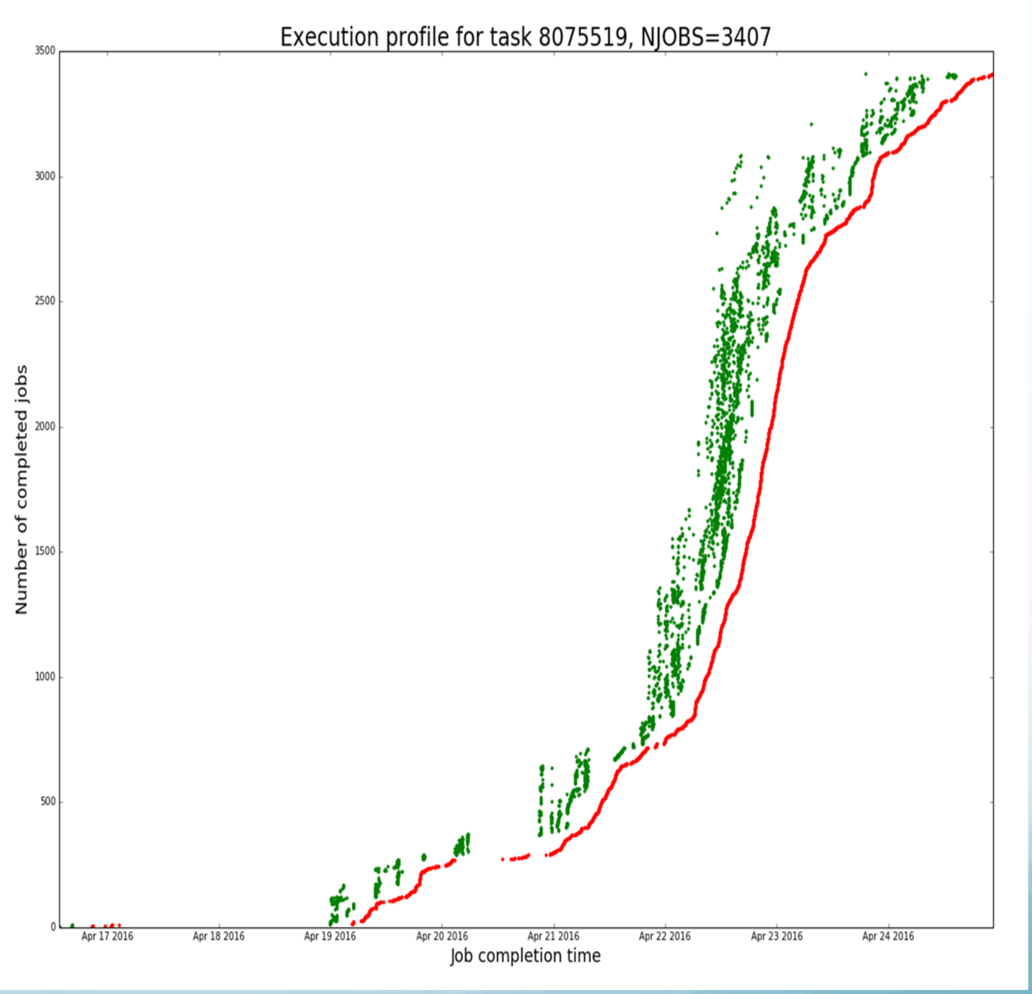




- Still some room for improvements for backfill consumption (red and yellow zones on charts)
  - Harvester will help with allocation of more nodes per one batch submission (red zones)
  - AES may help with efficient walltime utilisation (yellow zones)

Analytics and Machine Learning: Task Time to Complete and anomaly detection

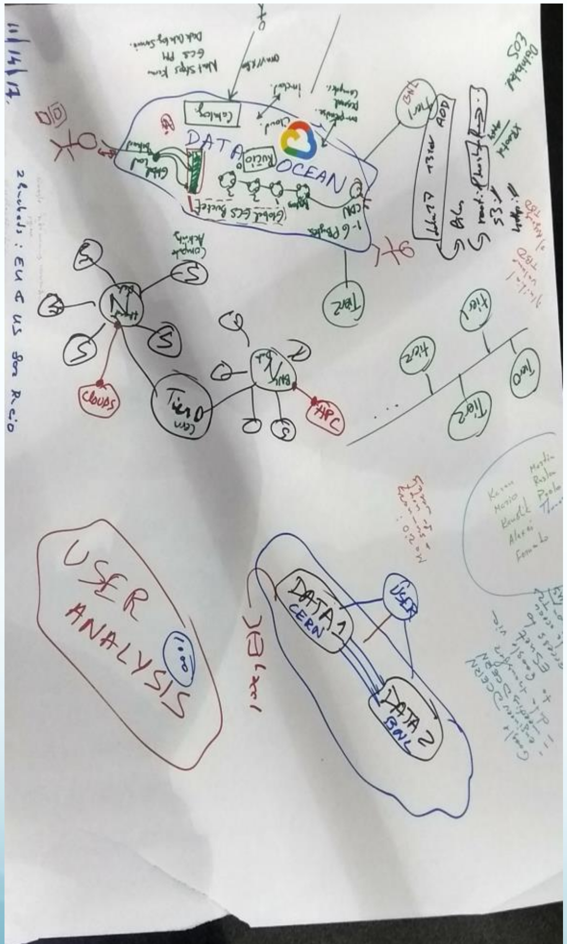
- Several prediction models:
  - Static (“cold”) model
  - Basic dynamic model
  - ML-based dynamic model
- Static “cold” predictions are implemented and being tested
- Profiles for basic dynamic model are being developed
- Application: validate if changes in the system have been favorable
- Develop ML-based model for task duration prediction:
  - Use available data (task parameters, resources state at task submission time) to predict TTC
  - Use predictors as inputs for ML models
  - Test models on historical data
  - Test models on real data

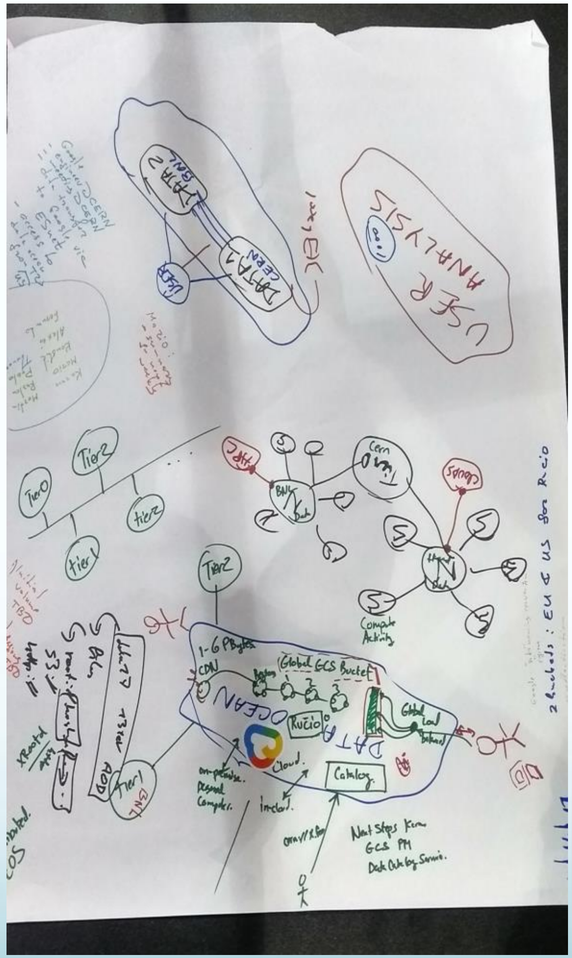


ATLAS Google Data Ocean Project

- Storage becoming a driving cost factor for High Luminosity LHC
  - ATLAS-Google common project to evaluate more dynamic use of storage
  - Store ATLAS data on Google Cloud Storage and access anywhere in the world
- First ATLAS attempt to run both storage and compute on a commercial cloud
- **Data** management: Google Cloud **Storage** like any other storage element for data transfer and accounting
  - Based on signed URLs
  - Third party transfer through FTS
    - Possible from all recent DPM and dCache WebDav endpoints
  - Download and upload of files through Rucio clients
- **Workload** management: manage Google **Compute** Engine resources through Harvester
  - Running a queue for simulation and a queue for analysis

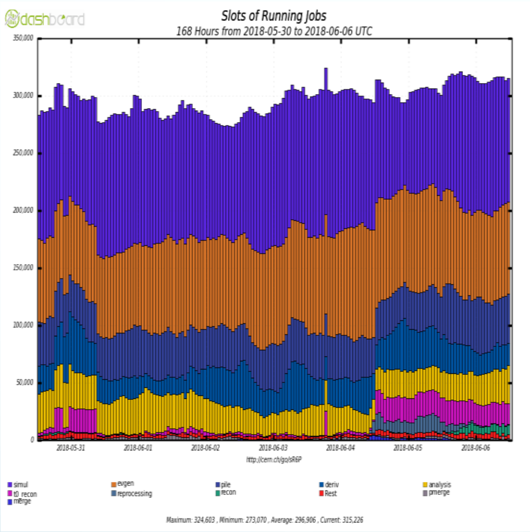
Time to collaborate with Google Cloud!





SC17 Denver, CO | hpc connects.

The first use cases



User analysis

Ensure 100% output availability

Overflow CPU to cloud compute

Data placement, replication, and popularity

Dynamically expand experiment storage capacity

Use cloud networks for increased throughput

Use cloud internal replication for popular data

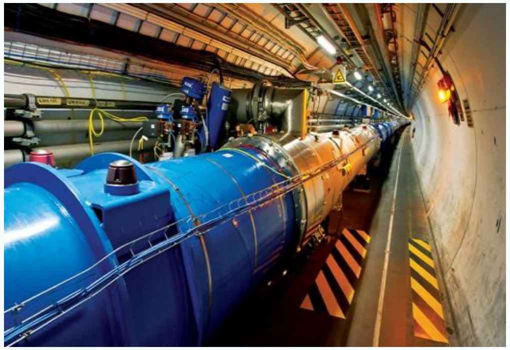
Data formats and streaming

Unravel experiment data format into constituents

Cloud-based marshalling of events from files

NEXT '18

Getting data into Google Cloud Storage



Necessary first step, but ...

...LHC is running!

- Must integrate transparently and on-the-fly
- Downtimes cause a lot of extra costs

Make GCS look like "just another data centre" in the WLCG

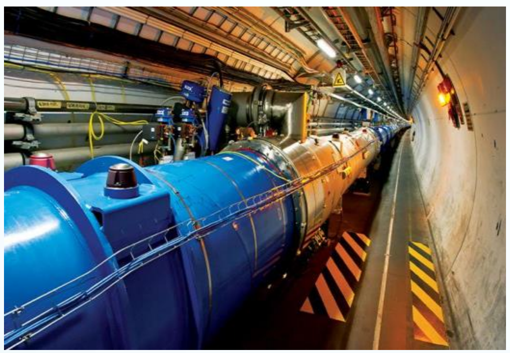
Must support data policy evaluation for organised activities

Must support user data access via existing authN & authZ

Must support existing protocols (WebDAV, gsiftp, root, S3, …)

Must support existing toolchain (ROOT, GFAL, FTS)

Getting data into Google Cloud Storage

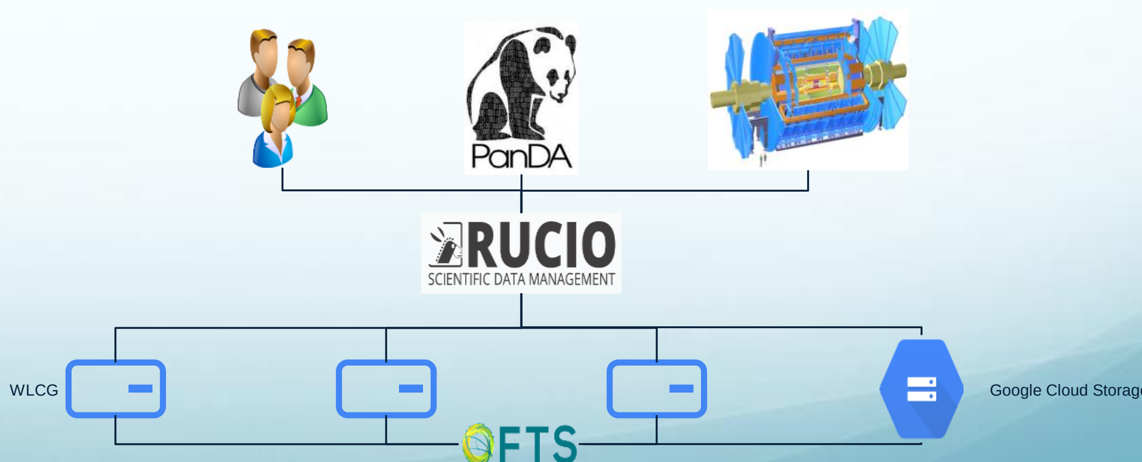


S3 used in first iteration due to full stack support

Rate-limited throughput at ~1 Gbps

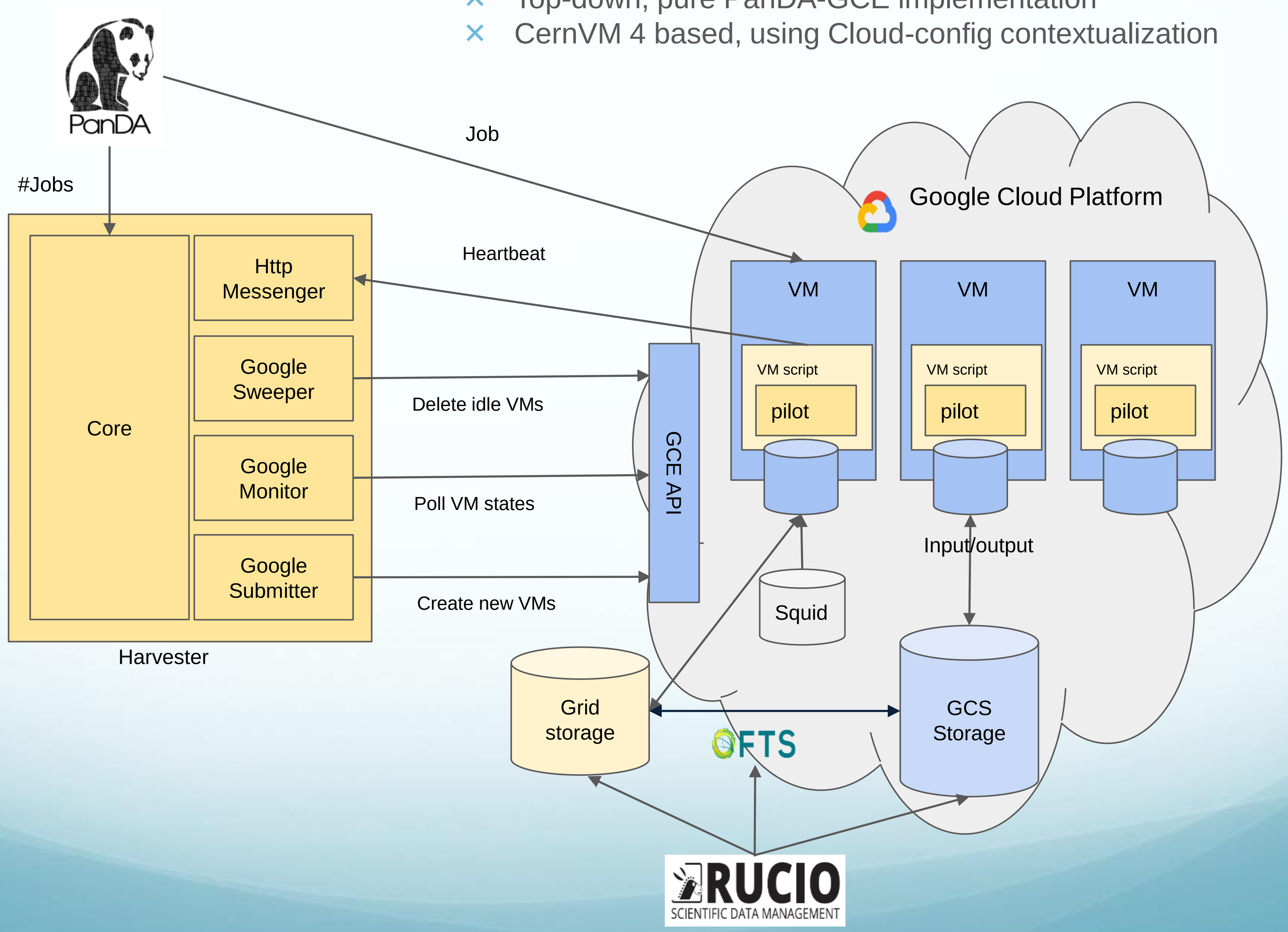
Key distribution problematic for user access

Decision to move to GCP-native client-side signed URLs



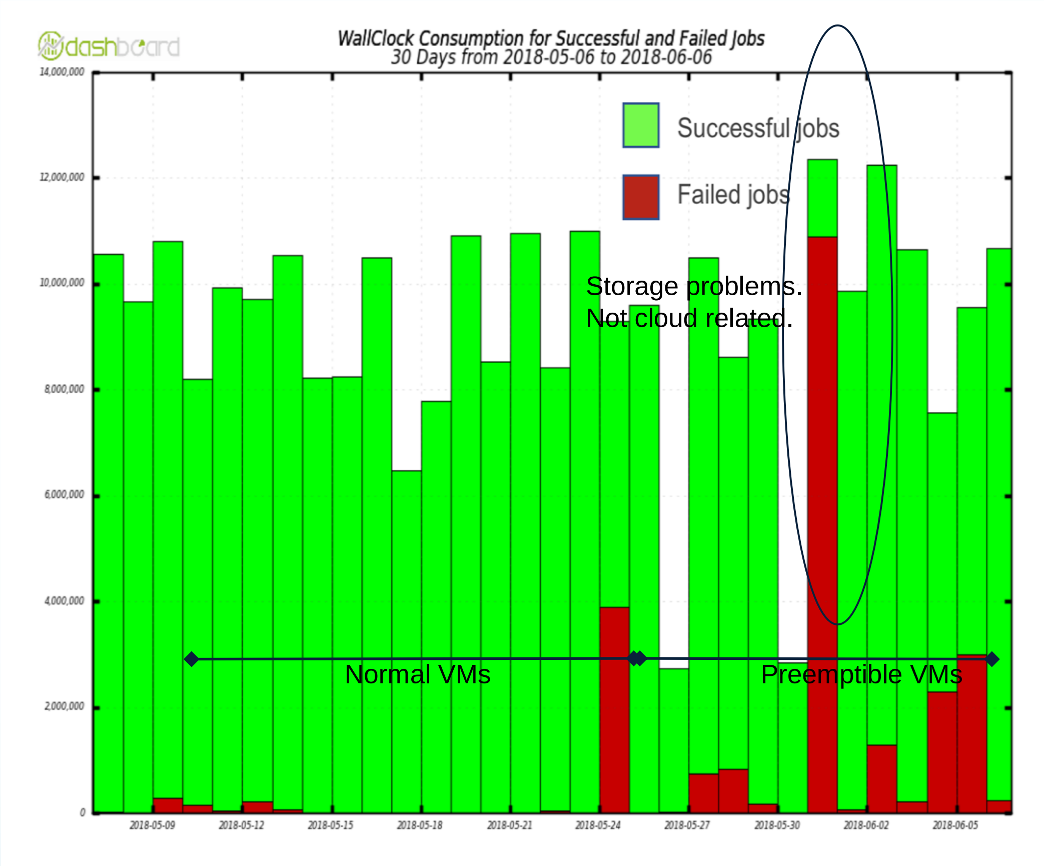
WFM. Block diagram

- Top-down, pure PanDA-GCE implementation
- CernVM 4 based, using Cloud-config contextualization



Very First Results

- Google Cloud Platform completely integrated in Rucio for data and PanDA for workload management
- Analysis use case in progress using cloud storage
- Expand on performance, scalability and cost studies



Efficiency of preemptible VMs can be optimized through usage of Event Service

Future Challenges

- New physics workflows
  - also new ways how Monte-Carlo campaigns are organized
- New strategies
  - “provisioning for peak”
- Integration with networks (via DDM, via IS and directly)
- Data popularity -> event popularity
- Address new computing model
- Address future complexities in workflow handling
  - Machine learning and Task Time To Complete and anomalies detection
  - Monitoring, analytics, accounting and visualization
  - Granularity and data streaming

Future Challenges. Cont'd

- Incorporating new architectures (like TPU, GPU, RISC, FPGA, ARM…)
- Adding new workflows (machine learning training, parallelization, vectorization…)
- Leveraging new technologies (containerization, no-SQL analysis models, high data reduction frameworks, tracking…)
- we have experience to enable large scale data projects for other communities
  - Some components of WMS and DDM software stack could be used by others
- Event Service and Event Streaming Service
- WMS – DDM coupled optimizations
  - WMS will evolve to enable new data models
  - Data lakes, data ocean, caching services, SDN, DDN,…
  - Another level of granularity (from datasets to events)
  - Distributed datasets

Acknowledgements

- Thanks to members of the BigPanDA and PanDA projects, and colleagues from the ATLAS experiment at the LHC

- Special thanks for materials and slides : Fernando Barreiro, Ian Bird, Shantenu Jha, Maksim Gubin, Mario Lassnig, Tadashi Maeno, Danila Oleynik, Markus Schulz, Torre Wenaus, Jack Wells