The ATLAS Production System Evolution

M. Borodin (U of Iowa)
On behalf of F. Barreiro, K. De, D. Golubkov, A. Klimentov,
T. Korchuganova, T. Maeno, P. Nilsson, S. Padolski, T. Wenaus and
ATLAS collaboration

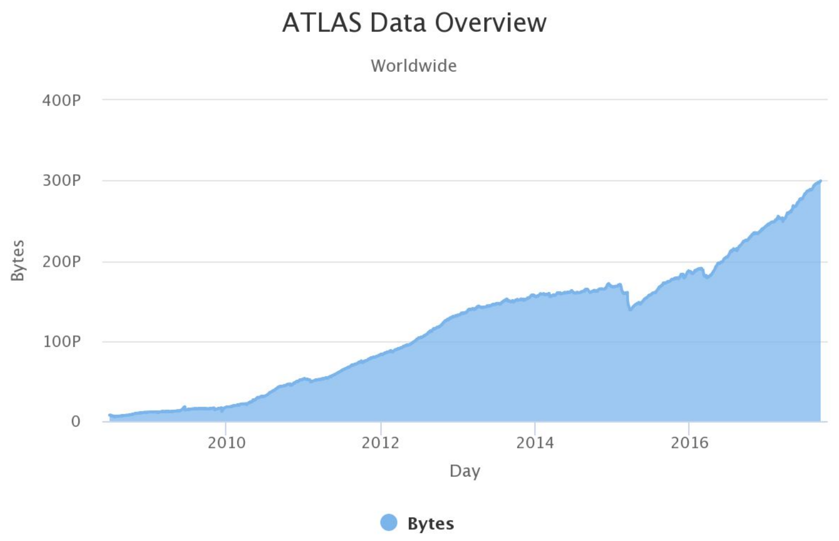
Introduction

- PanDA - **P**roduction **an**d **D**istributed **A**nalysis System
  - Designed to meet ATLAS production/analysis requirements for a data-driven workload management system capable of operating at LHC data processing scale
- New generation of ATLAS production system was developed for Run 2 and beyond – **ProdSys2**
  - Improved resource utilization
  - New types of computing resources: HPC, Clouds
  - Improved usability and robustness

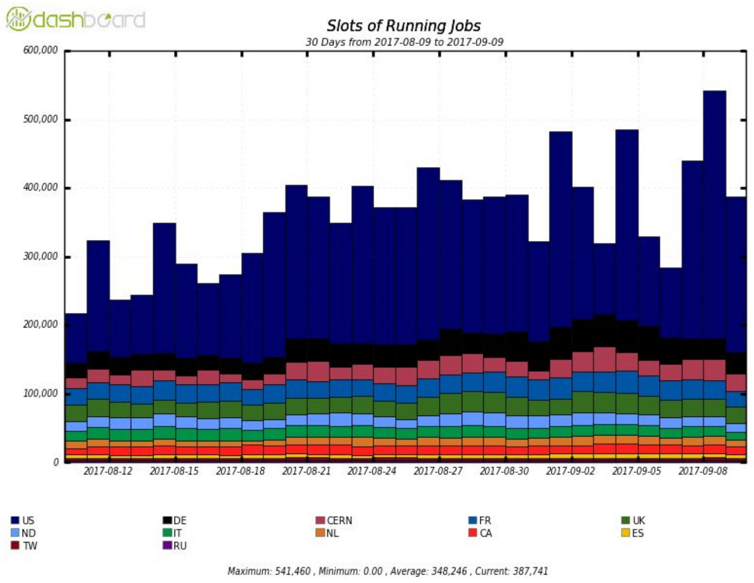
ATLAS production system design goals

- Deliver transparency of data processing in a distributed computing environment
- Achieve high level of automation to reduce operational effort
- Flexibility in adapting to evolving hardware, computing technologies and network configurations
- Scalable to the experiment requirements
- Support diverse and changing middleware
- Insulate user from hardware, middleware, and all other complexities of the underlying system
- Support custom workflow of individual physicis groups
- Incremental and adaptive software development

Orders of magnitude



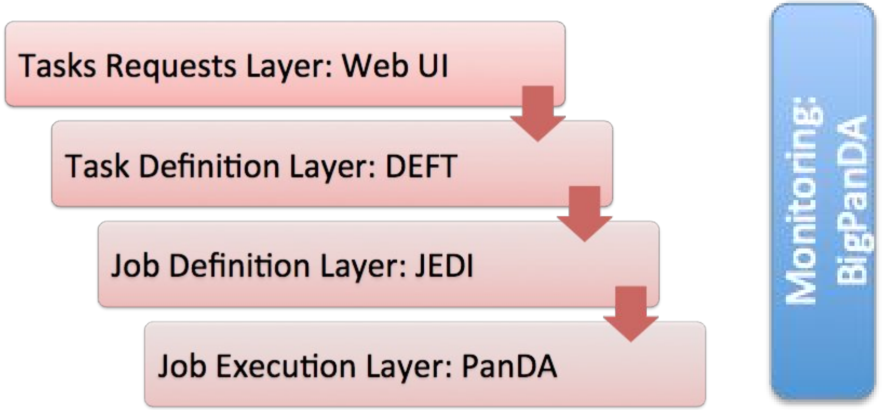
300 PB of data is managed by ATLAS DDM system (Rucio)



More than 300K cores used by simultaneously running jobs in the system

ATLAS production system components

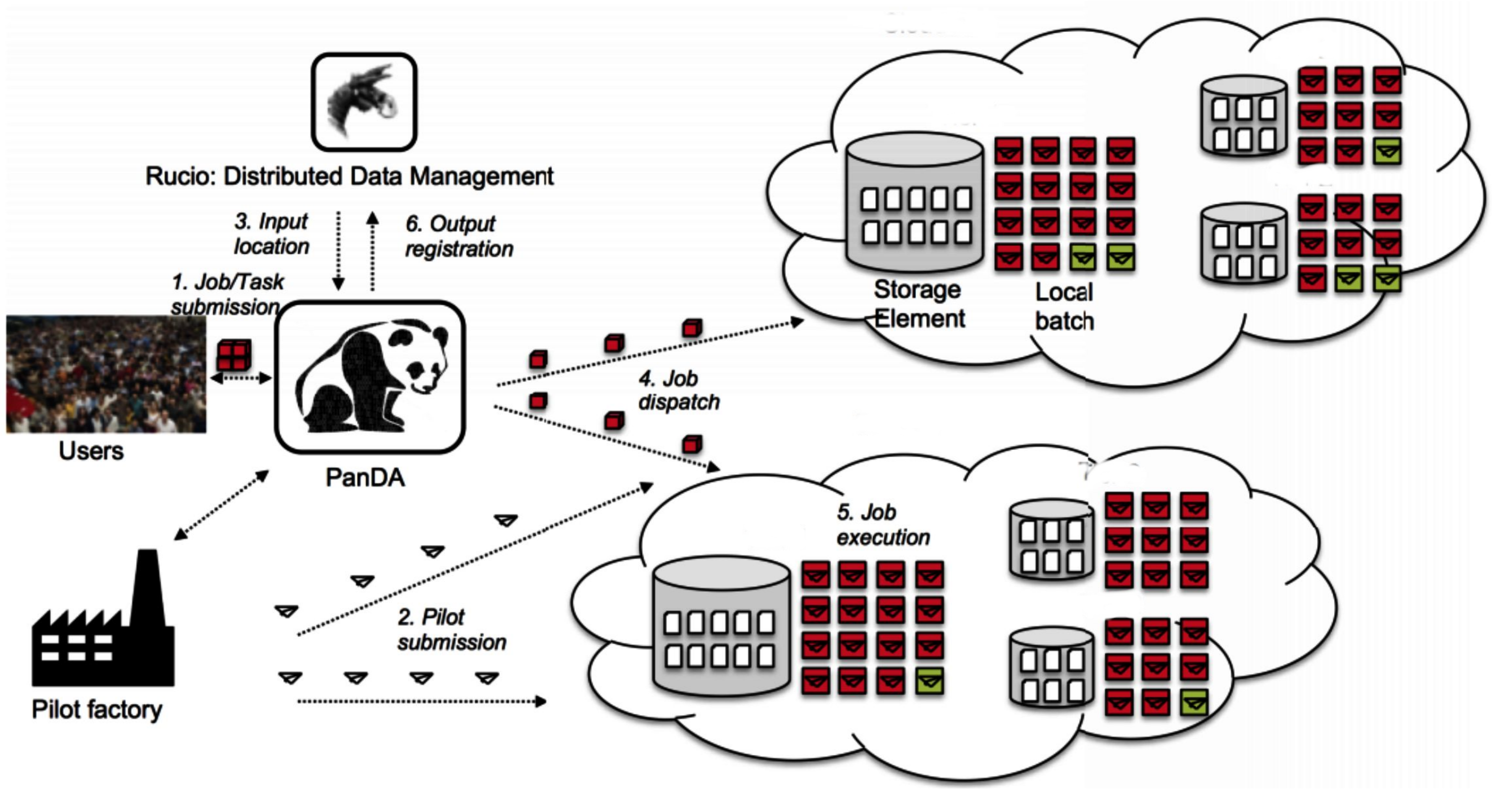
- **Web UI** for Managers and Users provides the interface for task* and production request managing and monitoring at the higher level
- Database Engine for Tasks (**DEFT**): is responsible for formulating the tasks, chains of tasks and also task groups (production request), complete with all necessary parameters
  - It also keeps track of the state of production requests, chains and their constituent tasks


Tasks Requests Layer: Web UI
Task Definition Layer: DEFT
Job Definition Layer: JEDI
Job Execution Layer: PanDA
Monitoring: BigPanDA

- Job Execution and Definition Interface (**JEDI**): is an intelligent component in the **PanDA** server to have capability for **task-level** workload management.
  - Key part of it is **'Dynamic'** job definition, which highly optimizes resources usage compared to 'Static' model used in ProdSys1.
    - Dynamic job definition in JEDI is also crucial for multi-core, HPCs and other new requirements
- Monitoring (**BigPanDA**): progress, status and error diagnostics for all components.
- The PanDA **pilot** is an execution environment used to prepare the computing element, request the actual payload (a production or user analysis job), execute it, and clean up when the payload has finished. Input and output are transferred from/to storage elements, including object stores.

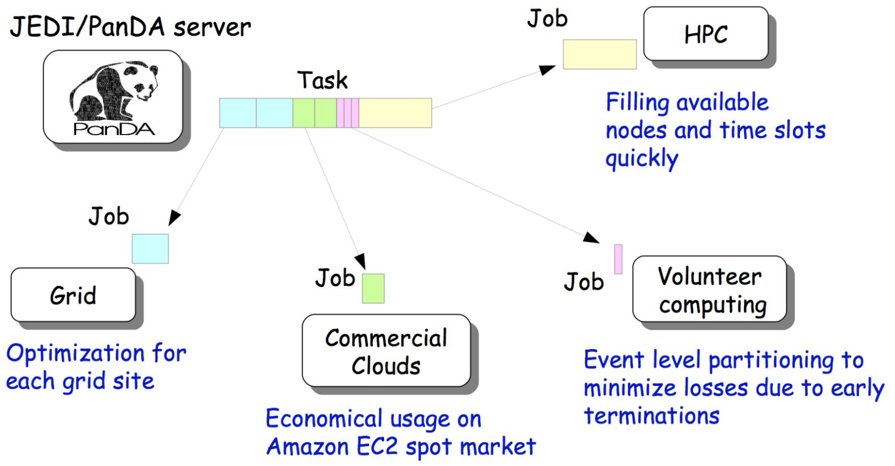
*Task consists of jobs that all run the same program.

High level overview



Dynamic job definition

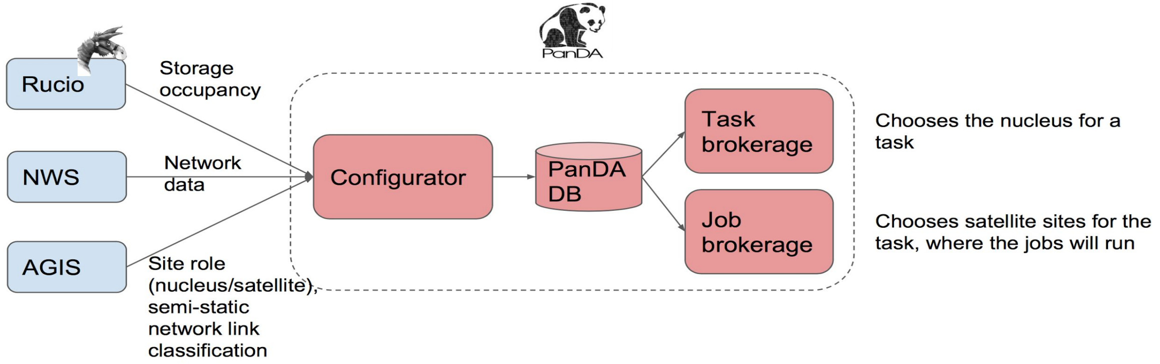
- Excluding requirements from users of detailed knowledge on computing resources
  - Especially for heterogeneous resources, e.g., many CPU cores, very short walltime limit, etc
- Self-optimization of job parameters
  - Real job metrics are collected using scout jobs
    - A small number (~10) of jobs (= scout jobs) are generated for each task with minimum input chunks
  - Job parameters are optimized using job metrics for the rest of input


JEDI/PanDA server
PanDA
Task
Job
HPC
Filling available nodes and time slots quickly
Job
Grid
Optimization for each grid site
Job
Commercial Clouds
Economical usage on Amazon EC2 spot market
Job
Volunteer computing
Event level partitioning to minimize losses due to early terminations

- More intelligence to the brokerage based on
  - Job retry history
  - Network forecast
  - Cache hit rate

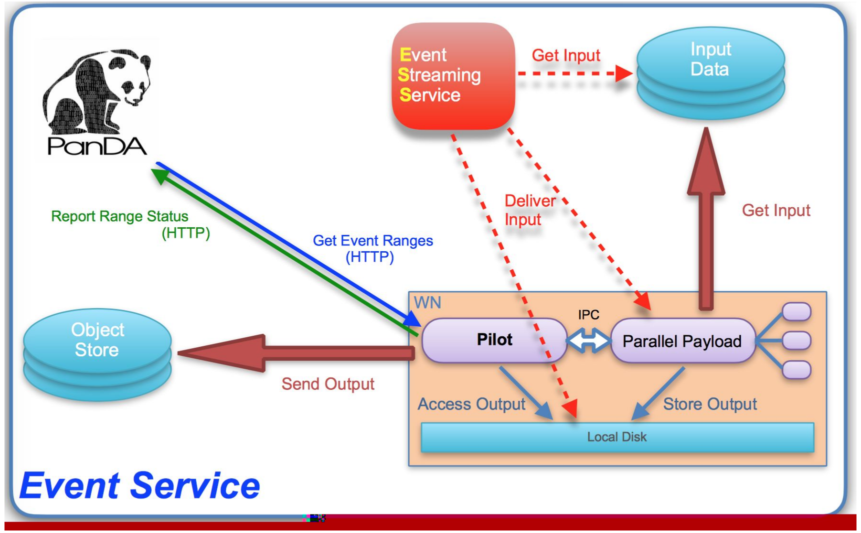
New model with no hierarchy (world cloud)

- Task nucleus:
  - Task brokerage will choose a nucleus for each task based on various criteria
  - The task output will be aggregated in a nucleus
  - The capability of a site to be a nucleus is defined manually in AGIS (ATLAS Grid Information System): Tier 1s and the bigger Tier 2s are defined as nuclei
- Task satellites:
  - Run jobs and ship the output to a nucleus
  - Job brokerage selects satellites for each task, based on usual criteria (e.g. number of jobs and data availability)
  - Satellites are selected across the globe: a network weight will bias towards well connected nuclei and satellites


PanDA
Rucio
Storage occupancy
NWS
Network data
AGIS
Site role (nucleus/satellite), semi-static network link classification
Configurator
PanDA DB
Task brokerage
Job brokerage
Chooses the nucleus for a task
Chooses satellite sites for the task, where the jobs will run

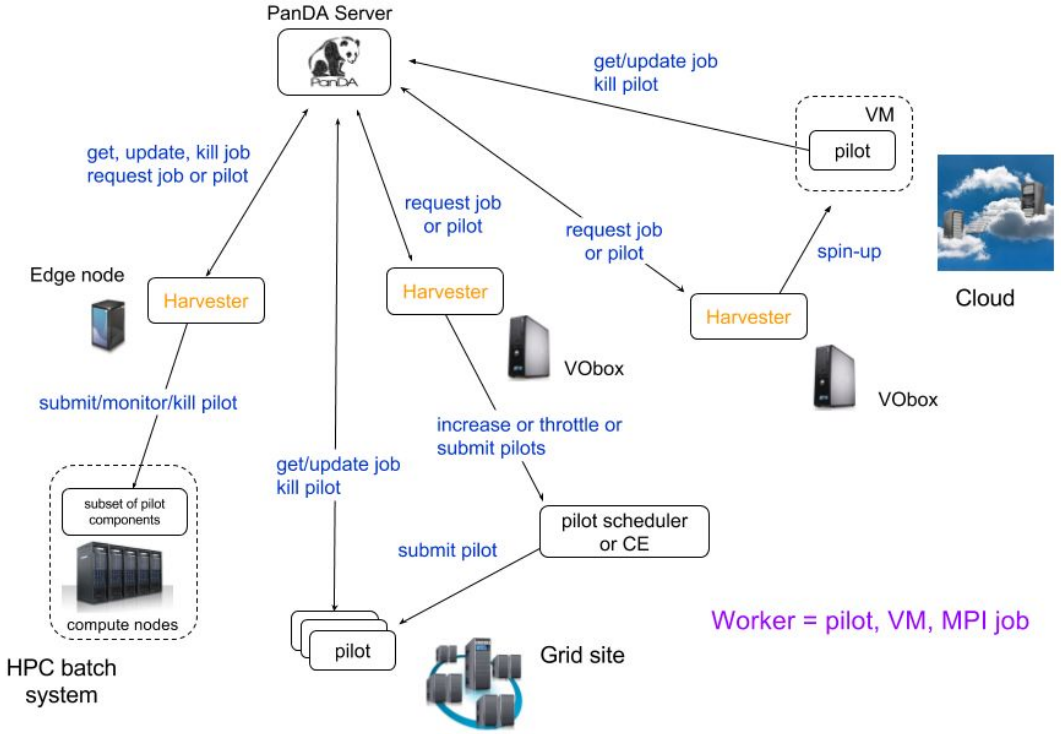
Event service

- A fine-grained approach to event processing. Designed for exploiting diverse, distributed and potentially short-lived resources
  - Quasi-continuous event streaming through worker nodes
- Exploit event processors fully and efficiently through their lifetime
  - Real-time delivery of fine-grained workloads to running application
  - Be robust against disappearance of compute node on short notice
- Decouple processing from chunkiness of files, from data locality considerations and from WAN latency
- Stream outputs away quickly
  - Negligible losses if the worker node vanishes
  - Minimal demands for the local storage



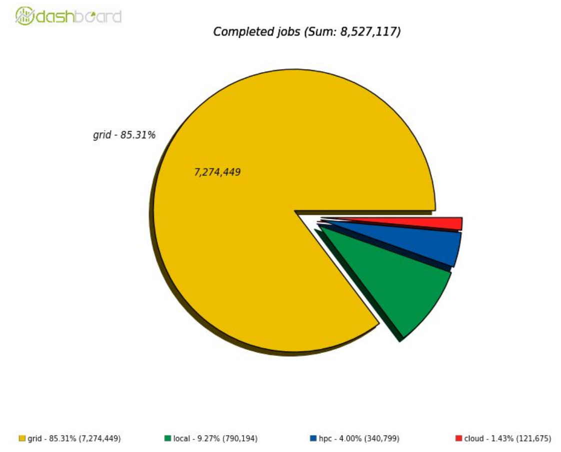
Harvester

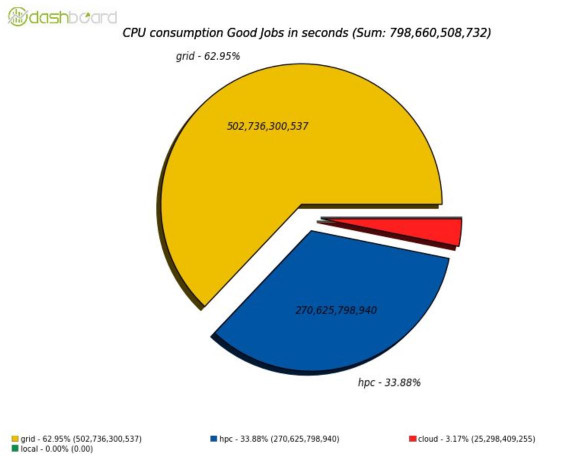
- Harvester is a resource-facing service between the PanDA server and collection of pilots for resource provisioning and workload shaping. It is a lightweight stateless service running on a VObox or an edge node of HPC centers to provide a uniform view for various resources. The following picture shows how harvester interacts with PanDA and resources.



HPC

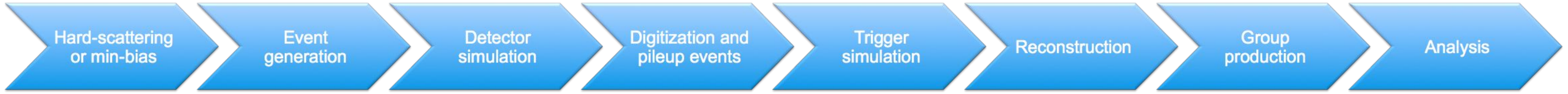
- HPC’s are integrated to the ATLAS production system
  - Titan at OLCF
  - Edison/Cori at NERSC
  - SuperMUC at LRZ
  - HPC2 at NRC-KI
  - ...





DEFT data model and workflows

- Model is represented by multilevel relational instances:
  - **Request** -> **Slice**(chain of steps) -> **Step** -> **Task**
  - Depending on workflow each instance could play a role of a template
  - Tasks are created by initiating a step instance.
- ATLAS production workflows were implemented in chosen model
  - **MC simulation** is composed of many steps: generate hard-processes, hadronize signal and minimum-bias events, simulate energy deposition in the ATLAS detector, digitize electronics response, simulate triggers, reconstruct data, transform the reconstructed data into reduced forms for physics analysis



- **Data Reprocessing** workflow has a tree structure, where output of one task can be an input for several more tasks
- **Derivation** is using so called “train” model, there each input runs on some of many predefined outputs
- **Tier-0** workflow
- **HLT, EventIndex,** ...

DEFT use case examples

“...In mc16, the datasets are mutually exclusive so each dataset must have campaign and sub-campaign attributes. For examples, if I have an EVNT container it can have many TID datasets from several sub-campaigns and several TID dataset for each sub-campaign. At any times we can add TID extension dataset for any of the sub-campaigns....”

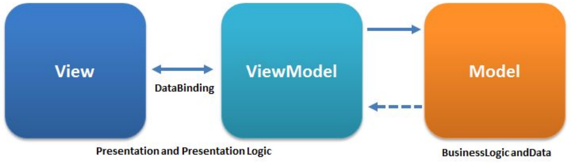
“...The original request is 1M events but due to some failures in the simulation step, the number of produced events is a little bit smaller than 1M events. In this case, the tasks of the digit+recon cannot be started due to "not enough input events". But if the request number is consistent with the existing events within 10% level, for example, in case of 1M events, if 900,000-1,000,000 events exists, tasks should be started w/o error. Is it possible?...”

“...This would mean that when the "Approve" button is used only the evgen task is created at first. Then, when the evgen task has reached some threshold of completed jobs, the rest of the chain from simulation onwards would be submitted...”

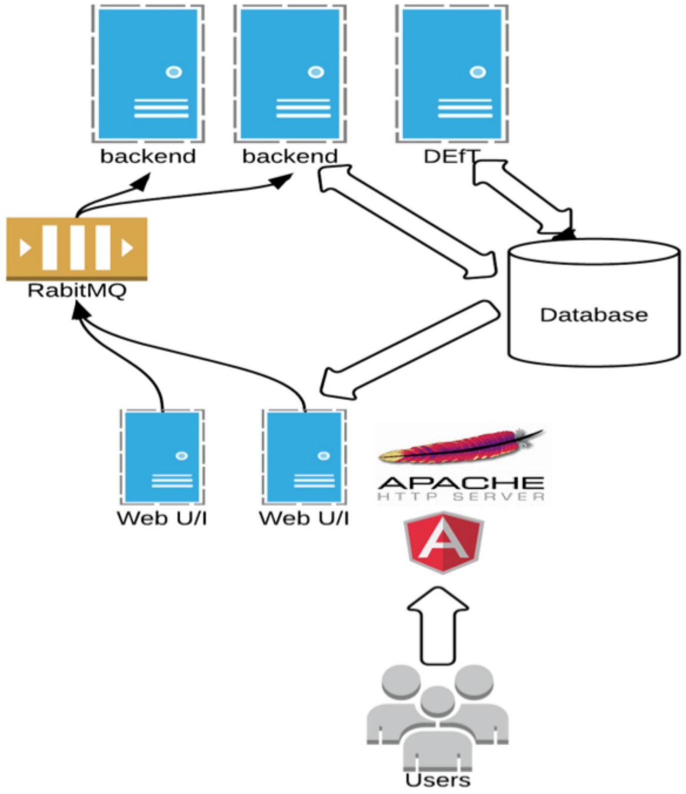
“...We would like to request a change to the way the final state is set for tasks with very low numbers of jobs, specifically those with <=10 jobs. At the moment these tasks are treated in the same way as scouts, i.e. they go to finished if any of the jobs succeed. This is problematic for us for two main reasons...”

DEFT and web UI development and deployment

- Key development points
  - Agile methodology: continuous meetings with the main users and often releases
  - Using open source
    - Django, Celery, AngularJS
  - «Model View ViewModel» approach

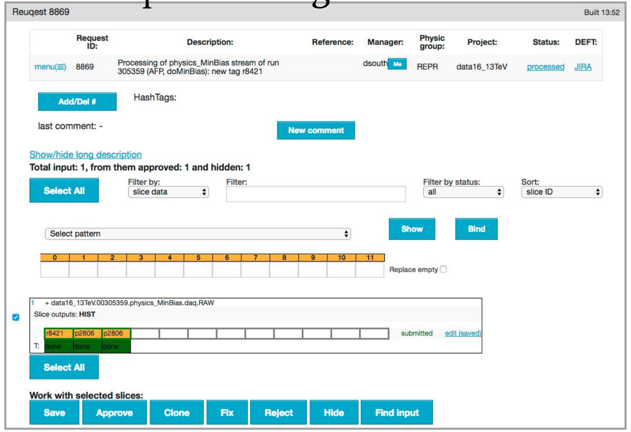


- Using CERN SSO(Shibboleth) for authentication and authorization



Web UI

Request management



Reuqest 8869 Built 13:52

	Request ID:	Description:	Reference:	Manager:	Physic group:	Project:	Status:	DEFT:
menu(☰)	8869	Processing of physics_MinBias stream of run 305359 (AFP, doMinBias): new tag r8421		dsouth Me	REPR	data16_13TeV	processed	JIRA

Add/Del # HashTags:

last comment: - New comment

Show/hide long description

Total input: 1, from them approved: 1 and hidden: 1

Select All Filter by: slice data Filter: Filter by status: all Sort: slice ID

Select pattern Show Bind

0	1	2	3	4	5	6	7	8	9	10	11

Replace empty ☐

☑ 1 + data16_13TeV.00305359.physics_MinBias.daq.RAW

Slice outputs: **HIST**

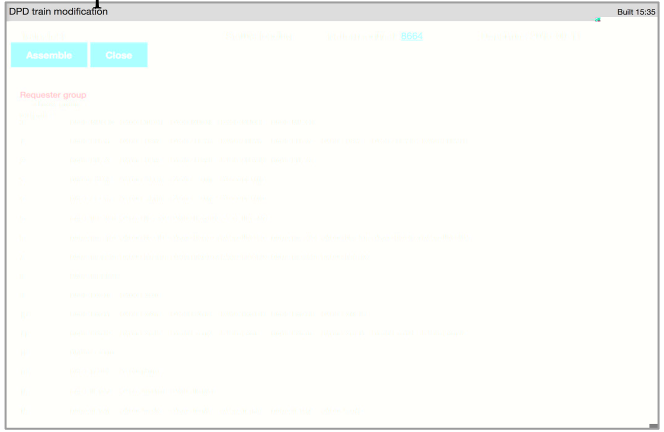
	r8421	p2806	p2806									submitted	edit (saved)
T:	done	done	done										

Select All

Work with selected slices:

Save Approve Clone Fix Reject Hide Find input

Request creation interface

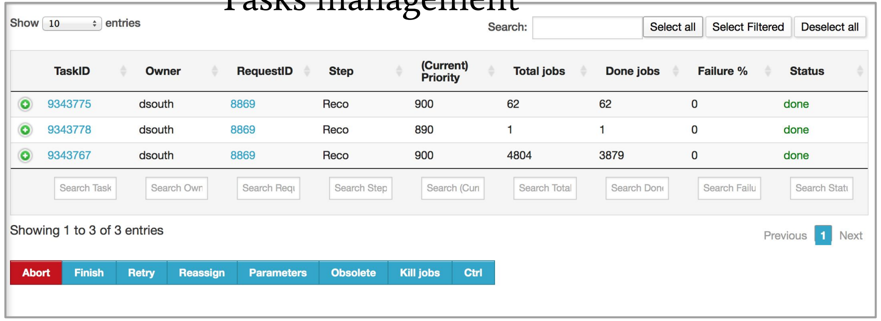


DPD train modification Built 15:35

Assemble Close

Requester group

Tasks management



Show 10 entries Search: Select all Select Filtered Deselect all

	TaskID	Owner	RequestID	Step	(Current) Priority	Total jobs	Done jobs	Failure %	Status
⊕	9343775	dsouth	8869	Reco	900	62	62	0	done
⊕	9343778	dsouth	8869	Reco	890	1	1	0	done
⊕	9343767	dsouth	8869	Reco	900	4804	3879	0	done
	Search Task	Search Own	Search Requ	Search Step	Search (Curr	Search Total	Search Done	Search Failu	Search Statu

Showing 1 to 3 of 3 entries Previous 1 Next

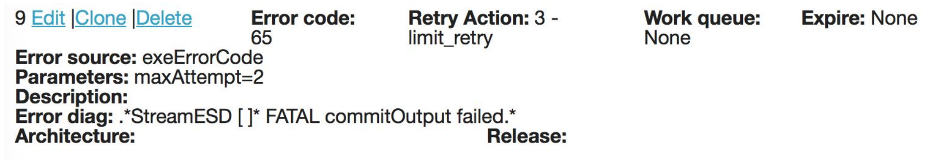
Abort Finish Retry Reassign Parameters Obsolete Kill jobs Ctrl

Production request processing

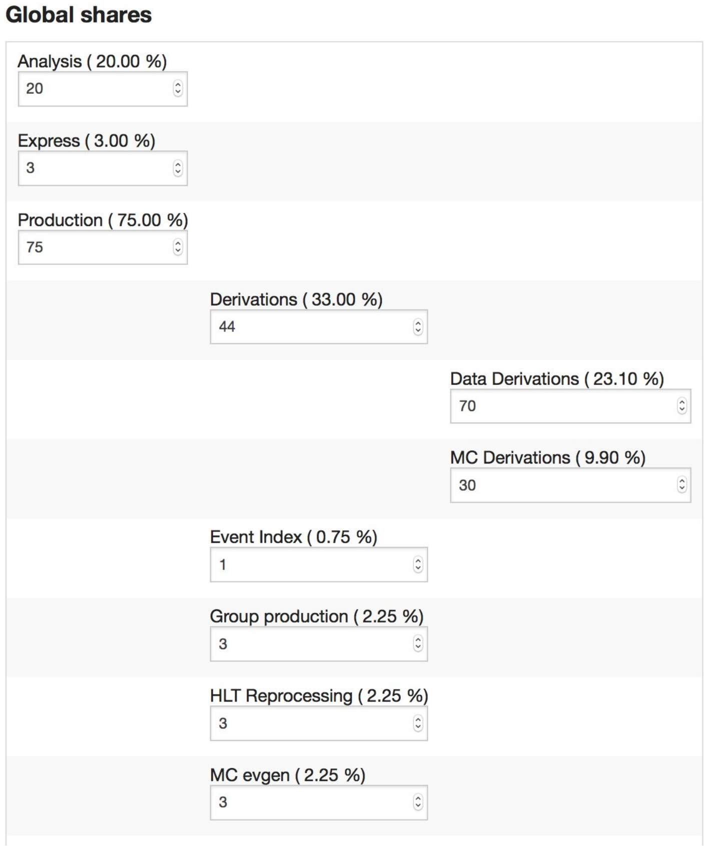
- Task request Web UI provides many general and experiment specific features:
  - **Bookkeeping**. Storing metadata, including arbitrary hashtags, allows to provide fine tuning statistics for running and historical tasks.
  - **Approval management.** E.g. MC production request required several levels of approval.
  - **Monitoring**. User can easily follow progress of a running tasks.
  - **Error Handling.** Task could fail because of many permanent (e.g. bug in software) and temporal (storage is down) reasons. To be able to quickly understand the root of the problem and fix it by redefining the task is one of the major features of the production system.
  - **Chaining** one production to the other. E.g. derivation production could be chained to MC or reprocessing task, that significantly speeds them up.
  - **Automation** of task submission. User can define a pattern and when new data appears tasks are started automatically.
  - ...

PanDA control

- Web UI is used to configure PanDA parameters
  - Limits, caps and weights
  - Share percentage
  - Retry module



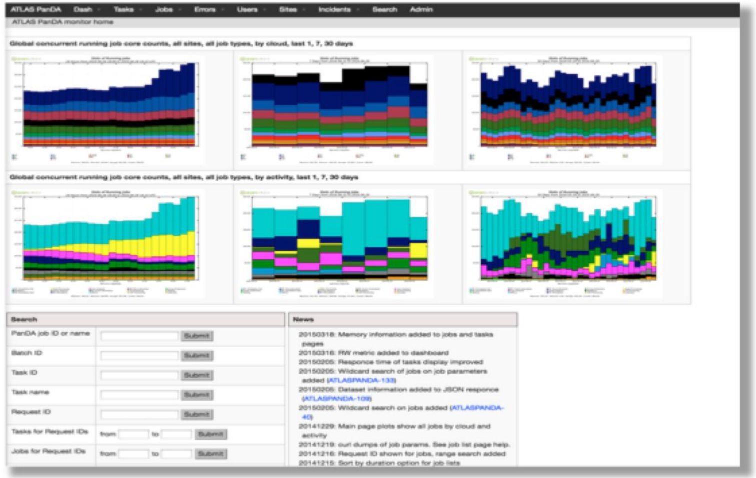
9 Edit |Clone |Delete **Error code:** 65 **Retry Action:** 3 - limit_retry **Work queue:** None **Expire:** None
Error source: exeErrorCode
Parameters: maxAttempt=2
Description:
Error diag: .*StreamESD [ ]* FATAL commitOutput failed.*
Architecture: **Release:**



Global shares

Analysis (20.00 %)	20
Express (3.00 %)	3
Production (75.00 %)	75
Derivations (33.00 %)	44
Data Derivations (23.10 %)	70
MC Derivations (9.90 %)	30
Event Index (0.75 %)	1
Group production (2.25 %)	3
HLT Reprocessing (2.25 %)	3
MC evgen (2.25 %)	3

BigPanDA Monitoring



Global concurrent running job core counts, all sites, all job types, by cloud, last 1, 7, 30 days

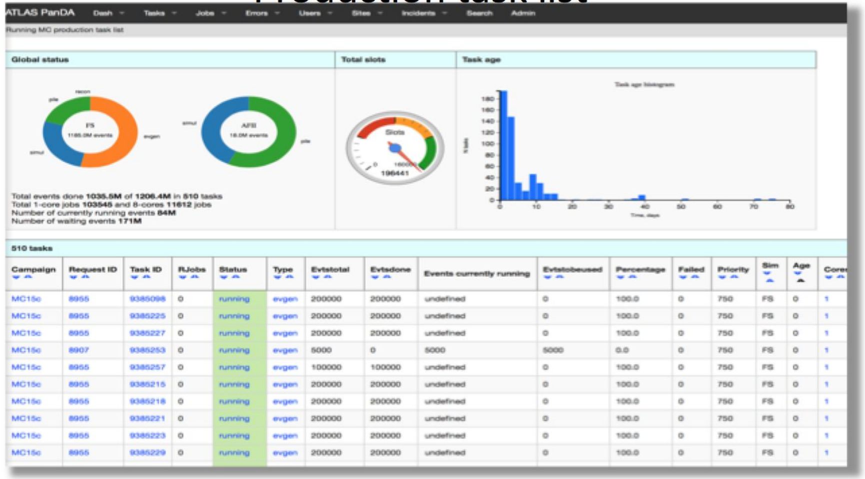
Global concurrent running job core counts, all sites, all job types, by activity, last 1, 7, 30 days

Search		
PanDA job ID or name		Submit
Batch ID		Submit
Task ID		Submit
Task name		Submit
Request ID		Submit
Tasks for Request IDs	from to	Submit
Jobs for Request IDs	from to	Submit

News

20150318: Memory information added to jobs and tasks pages
20150316: RW metric added to dashboard
20150205: Responce time of tasks display improved
20150205: Wildcard search of jobs on job parameters added (ATLASPANDA-133)
20150205: Dataset information added to JSON responce (ATLASPANDA-109)
20150205: Wildcard search on jobs added (ATLASPANDA-40)
20141229: Main page plots show all jobs by cloud and activity
20141219: curl dumps of job params. See job list page help.
20141216: Request ID shown for jobs, range search added
20141215: Sort by duration option for job lists

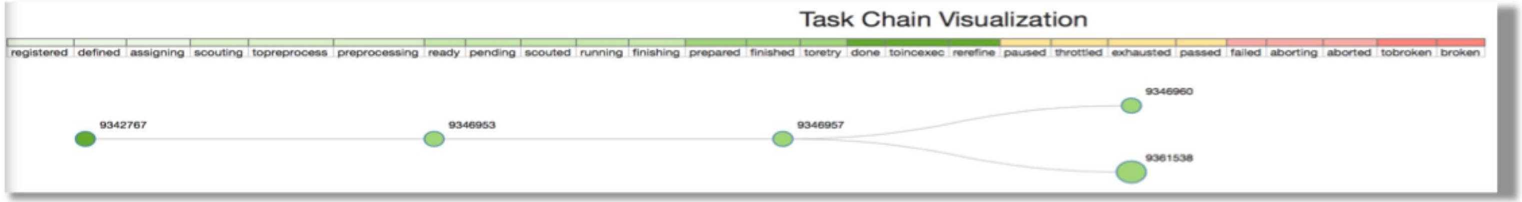
Production task list



Total events done **1035.5M** of **1206.4M** in **510** tasks
Total 1-core jobs **103548** and 8-cores **11612** jobs
Number of currently running events **84M**
Number of waiting events **171M**

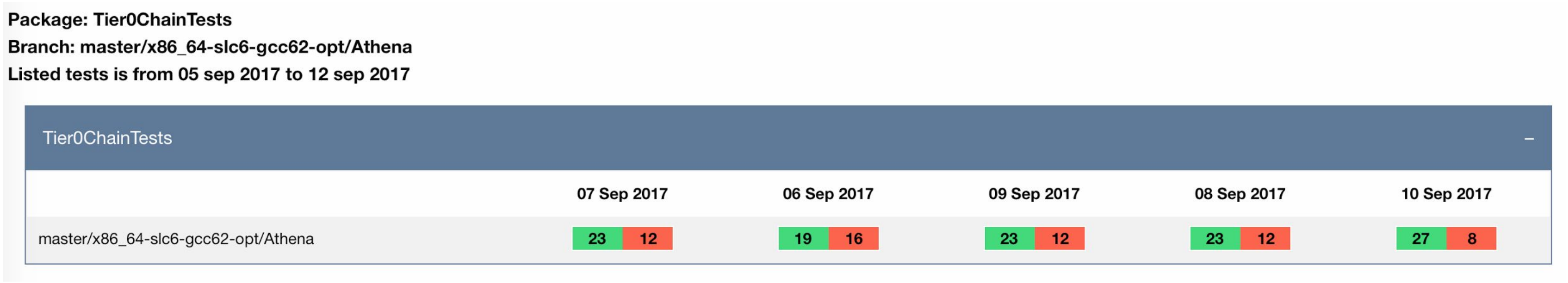
510 tasks

Campaign	Request ID	Task ID	RJobs	Status	Type	Evtstotal	Evtsdone	Events currently running	Evtstobeused	Percentage	Failed	Priority	Sim	Age	Cores
MC15c	8955	9385098	0	running	evgen	200000	200000	undefined	0	100.0	0	750	FS	0	1
MC15c	8955	9385225	0	running	evgen	200000	200000	undefined	0	100.0	0	750	FS	0	1
MC15c	8955	9385227	0	running	evgen	200000	200000	undefined	0	100.0	0	750	FS	0	1
MC15c	8907	9385253	0	running	evgen	5000	0	5000	5000	0.0	0	750	FS	0	1
MC15c	8955	9385257	0	running	evgen	100000	100000	undefined	0	100.0	0	750	FS	0	1
MC15c	8955	9385215	0	running	evgen	200000	200000	undefined	0	100.0	0	750	FS	0	1
MC15c	8955	9385218	0	running	evgen	200000	200000	undefined	0	100.0	0	750	FS	0	1
MC15c	8955	9385221	0	running	evgen	200000	200000	undefined	0	100.0	0	750	FS	0	1
MC15c	8955	9385223	0	running	evgen	200000	200000	undefined	0	100.0	0	750	FS	0	1
MC15c	8955	9385229	0	running	evgen	200000	200000	undefined	0	100.0	0	750	FS	0	1



ATLAS Nightly Test

- Some of the ATLAS functional software tests were moved to the grid
  - Too big to be run on dedicated test machines
  - It's important to test them in the production environment
- Special interface was developed for Nightly test submission and monitoring

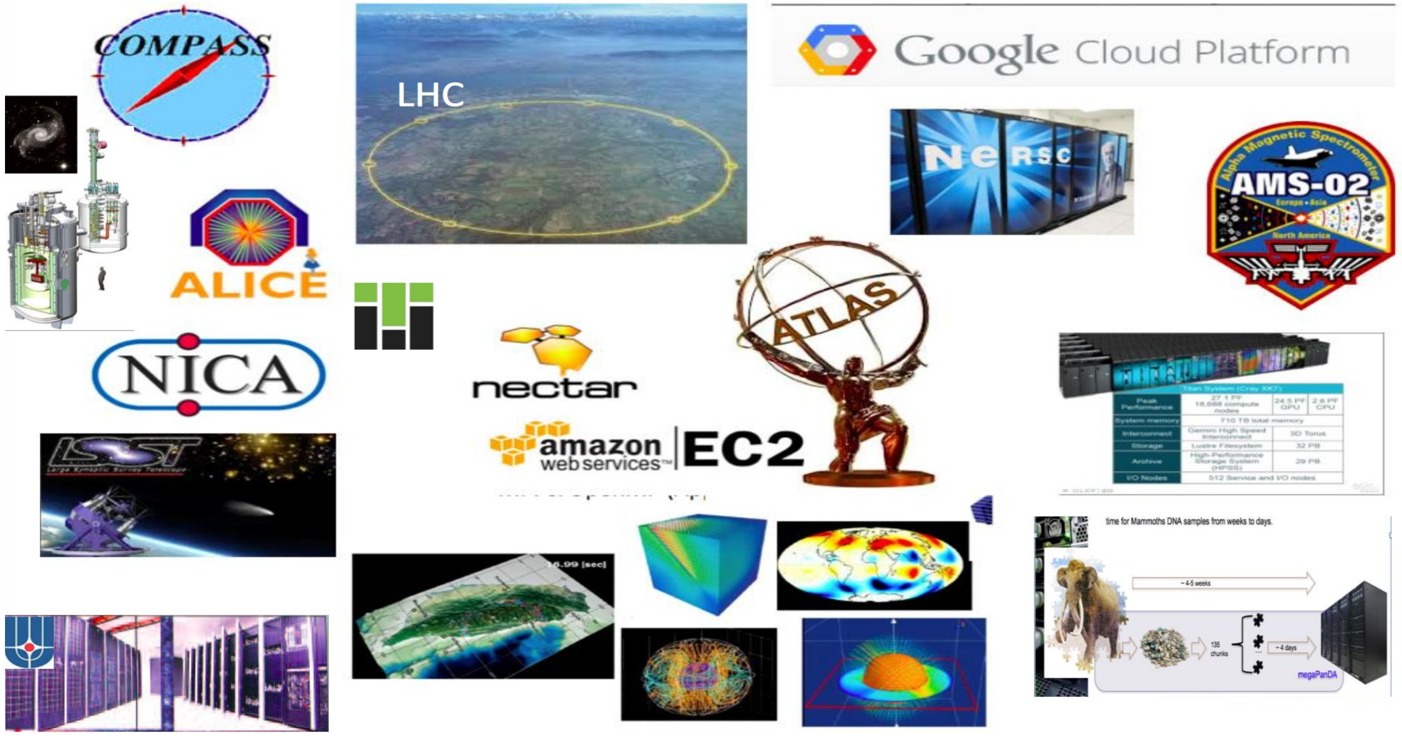


Package: Tier0ChainTests
Branch: master/x86_64-slc6-gcc62-opt/Athena
Listed tests is from 05 sep 2017 to 12 sep 2017

Tier0ChainTests

	07 Sep 2017	06 Sep 2017	09 Sep 2017	08 Sep 2017	10 Sep 2017
master/x86_64-slc6-gcc62-opt/Athena	🟩 23 🟥 12	🟩 19 🟥 16	🟩 23 🟥 12	🟩 23 🟥 12	🟩 27 🟥 8

The Growing PanDA Ecosystem



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		

Future

- Constantly increasing luminosity and always limited computing budget require to find ways for further efficient and economical use of traditional and new computing resources

- Automation of the system based on prediction for resource availability and the expected completion time for each task

- Interface evolution, such as automation of some operations, will improve system usability