The ATLAS Production System Evolution. New Data Processing and Analysis Paradigm for the LHC Run2 and High-Luminosity

M. Borodin, F. Barreiro, K. De, D. Golubkov, A. Klimentov, T. Korchuganova, T. Maeno, R. Mashinistov, S.Padolski, T. Wenaus

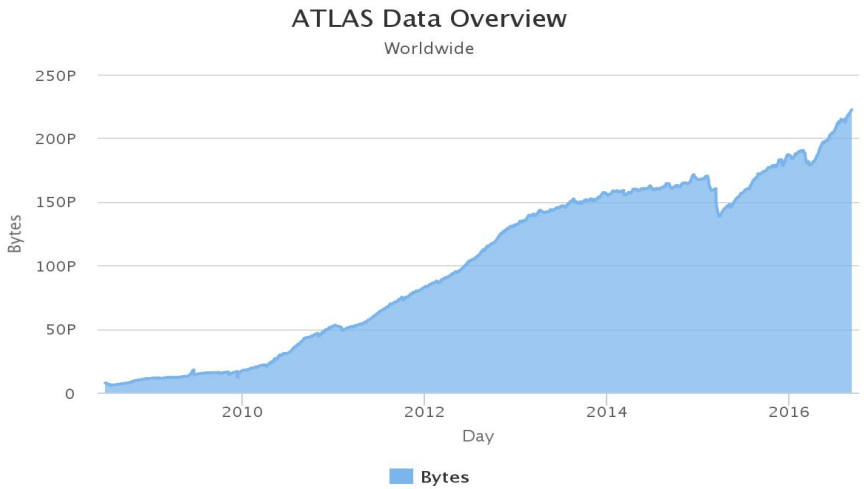
San Francisco, 10-14 Oct 2016
CHEP'2016

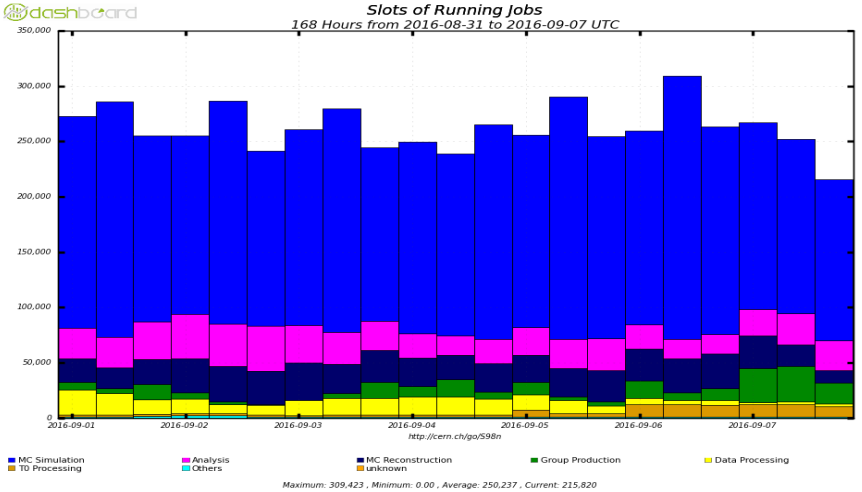
Introduction

- 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 – **ProdSys2**
  - Improved resource utilization
  - New types of computing resources: HPC, Clouds
  - Improved usability and robustness

Orders of magnitude



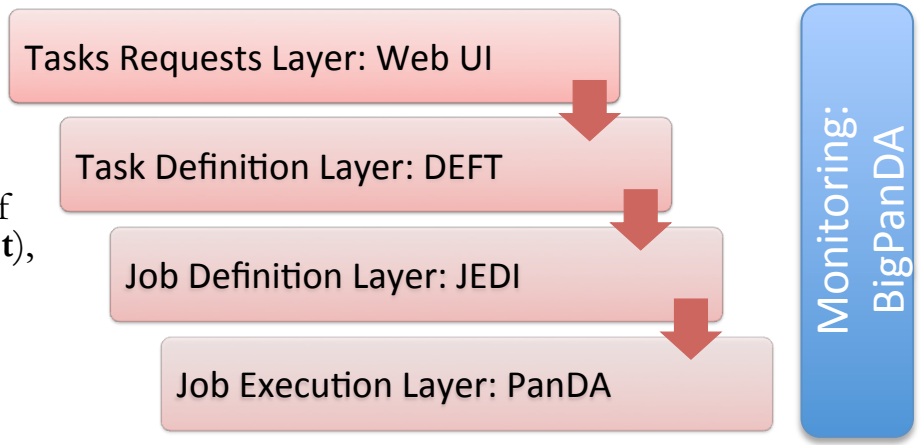


200 PB of data is managed by ATLAS DDM system (Rucio)

More than 250K cores used by simultaneously running jobs in the system

ATLAS production system components

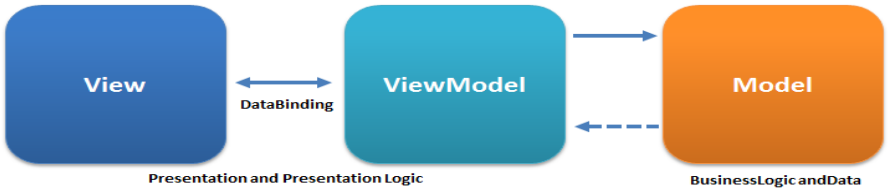
- **Web UI** for Managers and Users provides the interface for task and production request managing and monitoring at the higher level
- Database Engine for Tasks (**DEfT**): is responsible for formulating the tasks, **chains** of tasks and also task groups (**production request**), complete with all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

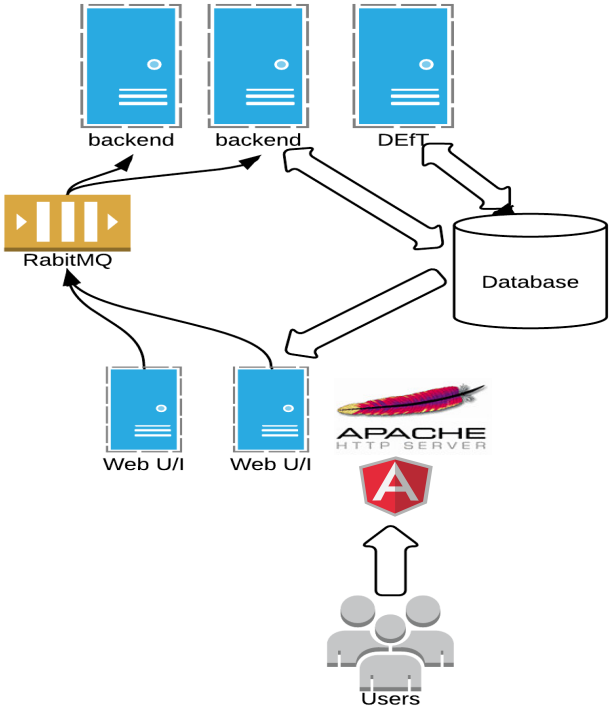


- Job Execution and Definition Interface (**JEDI**): is an intelligent component in the **PanDA** server to have capability for **task-level** workload management.
  - Key part of it is '**Dynamic**' job definition, which highly optimizes resources usage compare to 'Static' model used in ProdSys1.
    - 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.

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

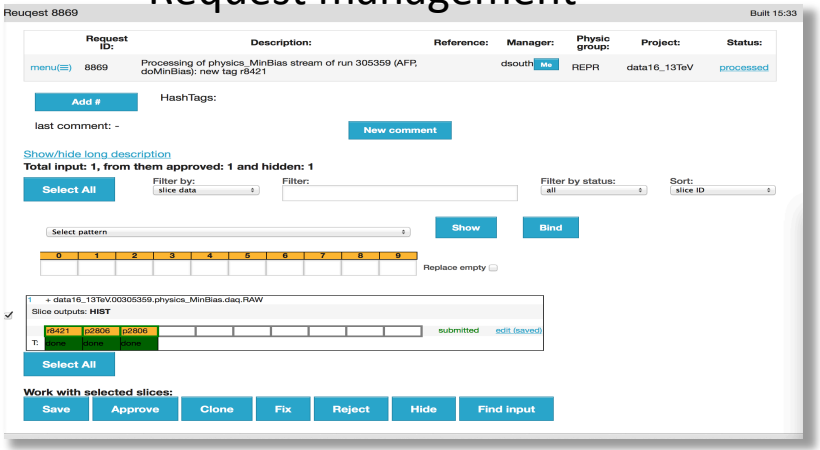
Production system data model and workflows

- 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

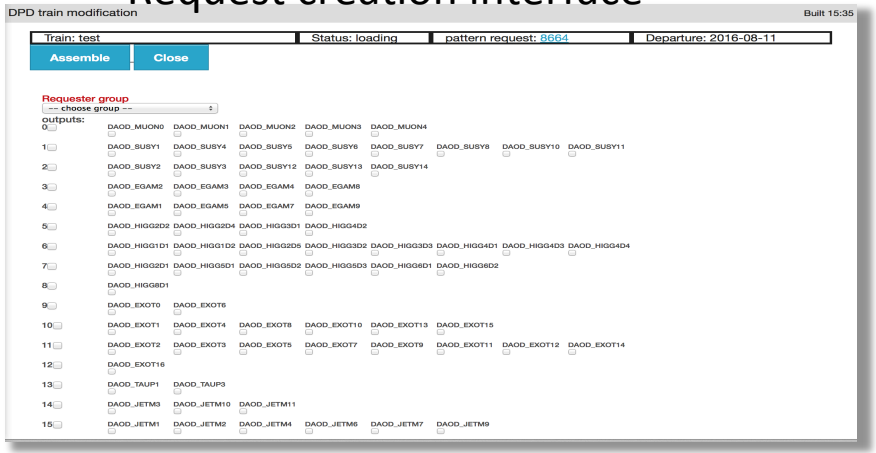
  - **Data Reprocessing** workflow has a tree structure, where output of one task can be an input for several more tasks
  - **Derivation** is using so called "train" model, there each input runs on some of many predefined outputs.
  - **Tier-0** workflow
  - **HLT, EventIndex, …**

Web UI

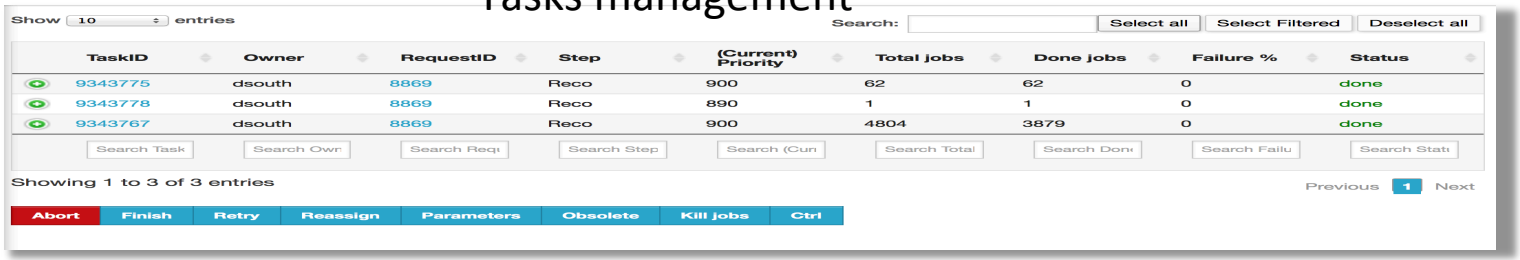
Request management



Request creation interface



Tasks management

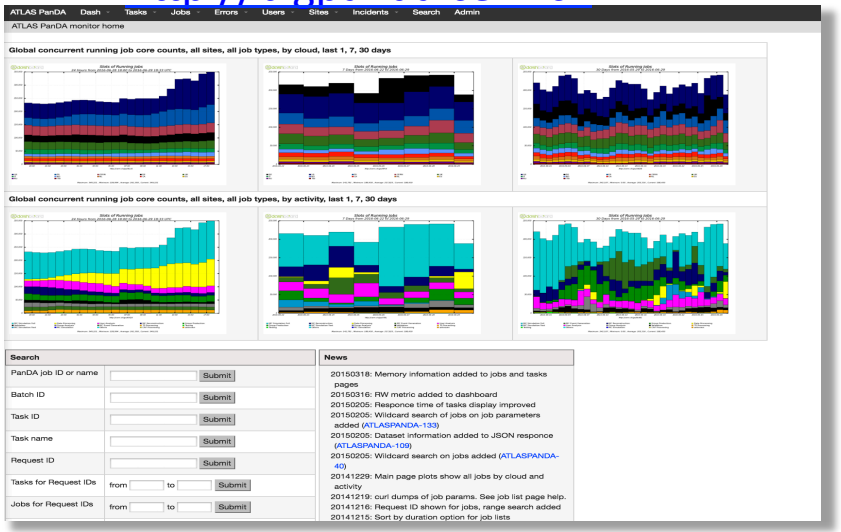


Work with production request

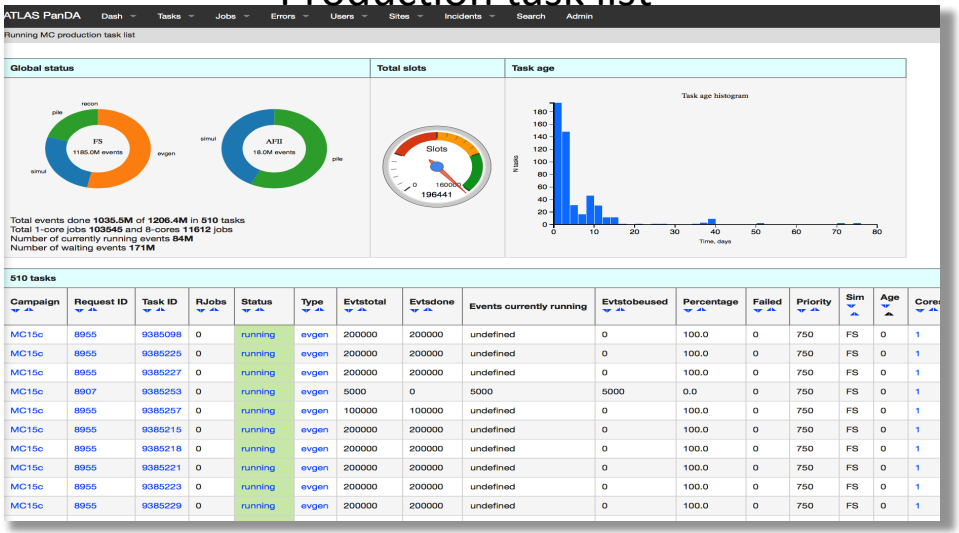
- Task request Web UI provides many general and experiment specific features:
  - **Bookkeeping**. Storing metadata, including arbitrary hashtags, allows to provide fine tuning statistics for running and historical tasks.
  - **Approval management**. E.g. MC production request required several levels 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.
  - …

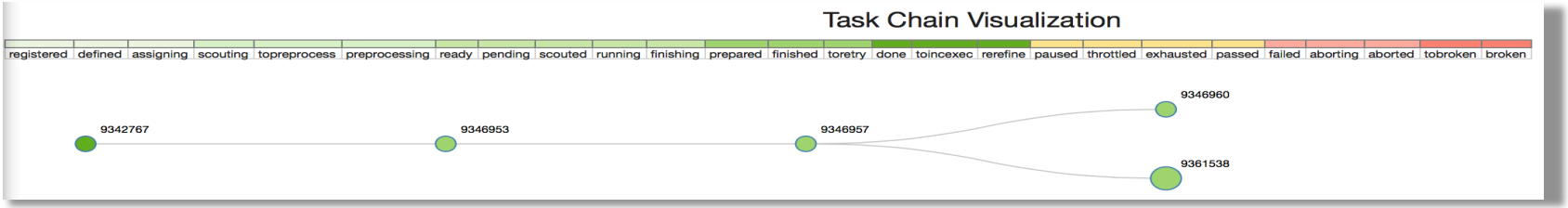
BigPanDA Monitoring

http://bigpanda.cern.ch



Production task list





BigPanDA Monitoring

- Rapid identification of failures and monitoring of progress of distributed physics analysis and production
- **4** distinct user behavior patterns: distributed computing systems operators, shifters, physicist end-users and computing managers
- **13k** queries, **1500** unique sessions/day
- Aggregation **2M** jobs/day with 6 months retrospective
- Drilling-down from high level summaries to detailed diagnostics data

BigPanDA Monitoring

- Feedback driven development (131 user requests implemented in 2016)
- Actively adding visualization (rendered both on web-client and server side)
- New forecasting features (Time To Complete)
- Continuously evaluating new approaches to raise up performance
  - Redistributing processing logic between Oracle storage and Django backend
  - Optimization data partitioning and inventing preprocessing
  - R&D tests show up to factor 150 improvement in response time

Conclusion

- New generation of ATLAS production system has performed well for Run 2 data
- New improvements and optimizations are on going
  - Using ML technology for task time to complete prediction and anomaly detection
  - New interface with richer functionality
  - Full automation of using different computing resources, such as HPCs and Clouds
- Developed system is being evaluated by other scientific experiments.