The Next Generation ATLAS Production System.

M. Borodin, K. De, D. Golubkov,
A. Klimentov, T. Maeno, R. Mashinistov,
A. Vaniachine

nëc’2015

Outline

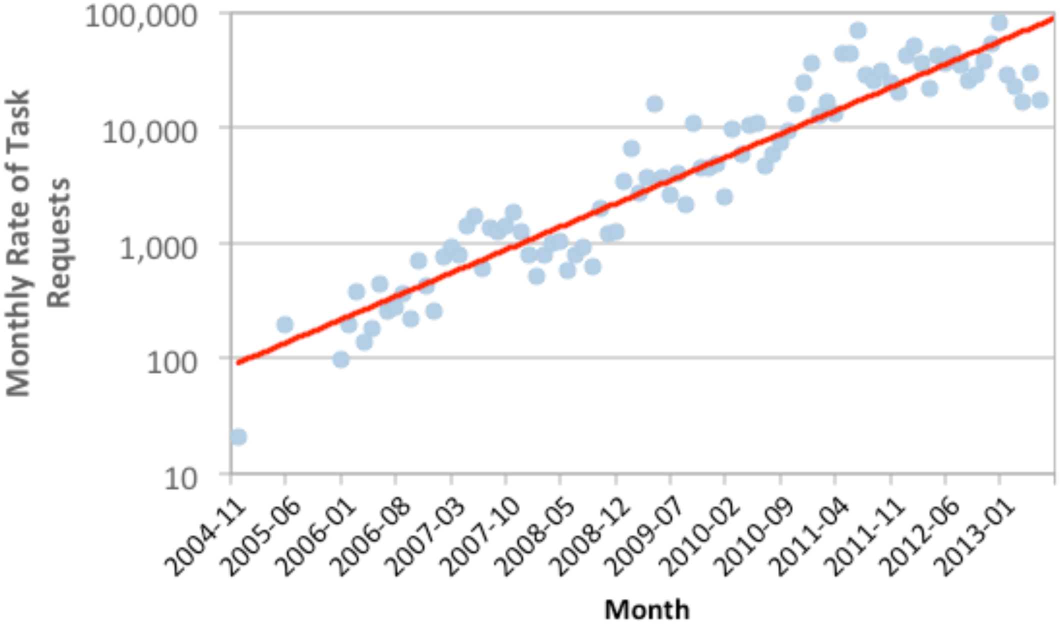
- Production system – what it is?
- Current state of the ATLAS production system
- Tasks requests system development overview

Production system

- During this week you've heard about many technologies and concepts which are used in HEP: Big Data, GRID, HPC, virtualization …
- It raises many question from end user point of view, when it's used for big experiment, such an ATLAS:
  - How to start your task?
  - How to monitor the progress?
  - How to manage tasks which are running?
  - How to recover failed tasks?
  - How to connect your task with already produced results?
  - …
- To answer all these questions we are developing "production system".

Motivation

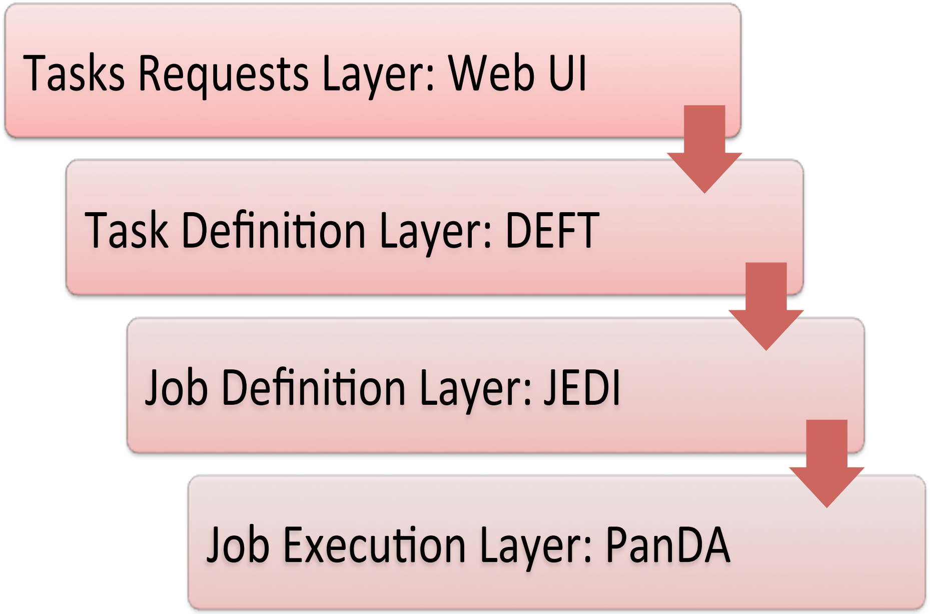
- In the ATLAS experiment, the Big Data processing generates a multiplicity of requirements as more data and use cases emerge
  - For Big Data processing, we adopted the data driven approach, where software applications transform the input data into the output data
    - In the ATLAS production system, each data transformation is represented by a task – a collection of many jobs submitted by the ATLAS workload management system (PanDA) and executed on the Grid



- Over the years, the success of our data transformation approach resulted in the exponential growth both in the number of data transformation applications and in the rate of task submissions
- The LHC shutdown presented an opportunity to reengineer the ATLAS Big Data processing infrastructure, adding extra layers further improving the system scalability and flexibility –ProdSys2

ProdSys2 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

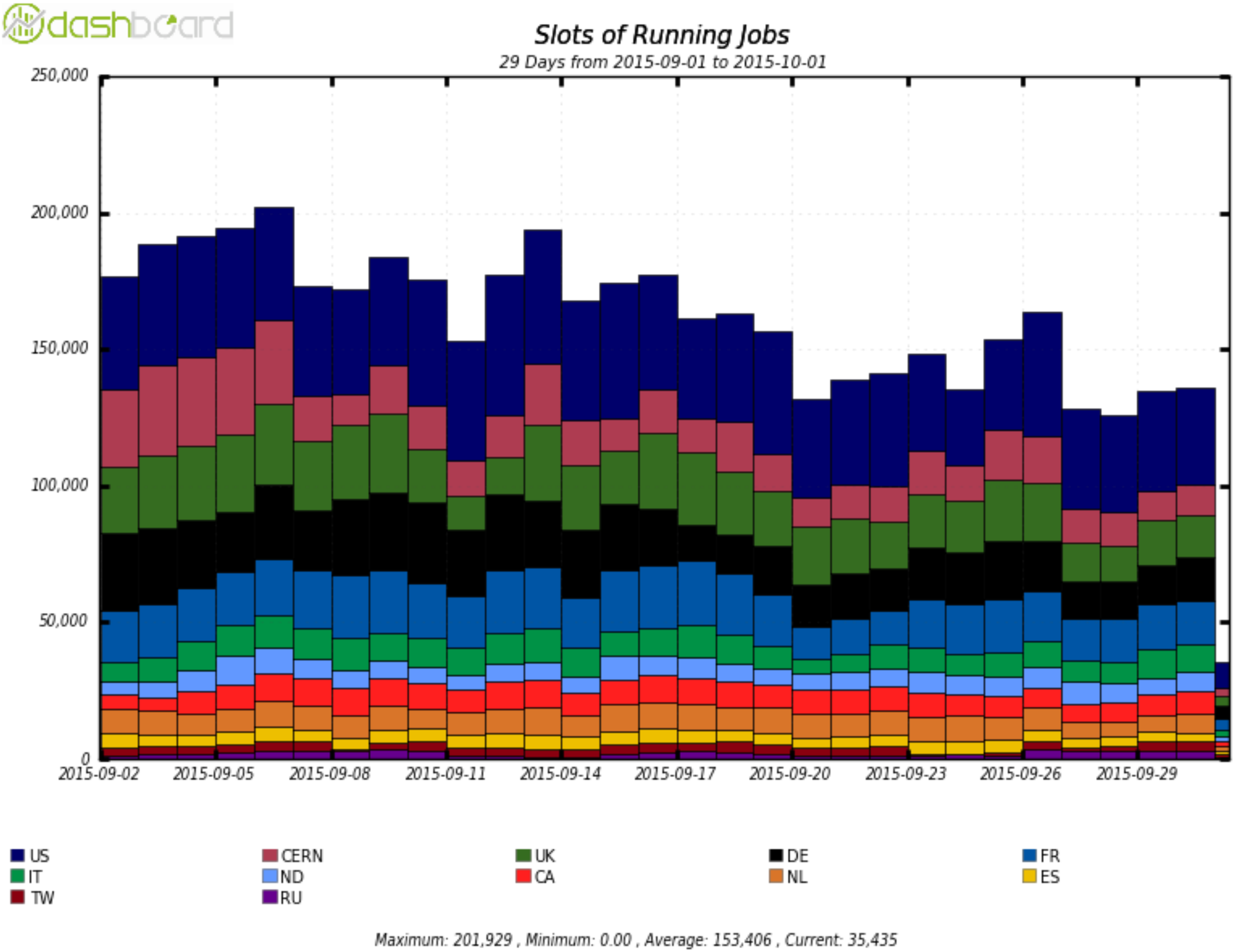
Job Definition Layer: JEDI

Job Execution Layer: PanDA

- 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, HPC's and other new requirements

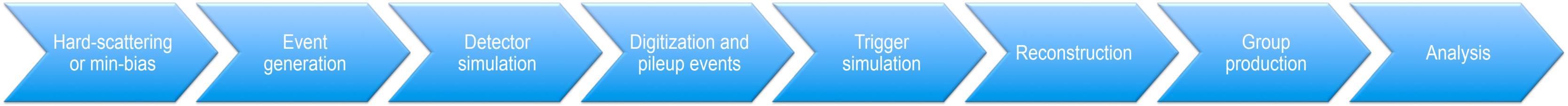
Statistics

- ProdSys2 is in production for ATLAS since December of 2014
  - More than 400 000 done tasks
  - More than 4000 production requests
  - Dozens of implemented workflows
  - Hundred papers used results, which are based on data produced with the production system
- Development of ProdSys2 is always on-going:
  - New workflows appear as users become familiar with the system;
  - Future automation allows to achieve faster request completion time.


dashboard
Slots of Running Jobs
29 Days from 2015-09-01 to 2015-10-01
250,000
200,000
150,000
100,000
50,000
0
2015-09-02
2015-09-05
2015-09-08
2015-09-11
2015-09-14
2015-09-17
2015-09-20
2015-09-23
2015-09-26
2015-09-29
US
CERN
UK
DE
FR
IT
ND
CA
NL
ES
TW
RU
Maximum: 201,929 , Minimum: 0.00 , Average: 153,406 , Current: 35,435

ProdSys2 use-cases

- Production system is a workflow driven system and it's used for dealing with the all variety of ATLAS activities:
  - Simulation (MC production)
  - Data processing ("Tier0" processing and reprocessing)
  - High Level Trigger (HLT) reprocessing
  - Derivation and train production (slimming, skimming…)
  - Event Index
- Typical ATLAS workflow composed of many data transformation steps, e.g. the Monte Carlo simulations workflow 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



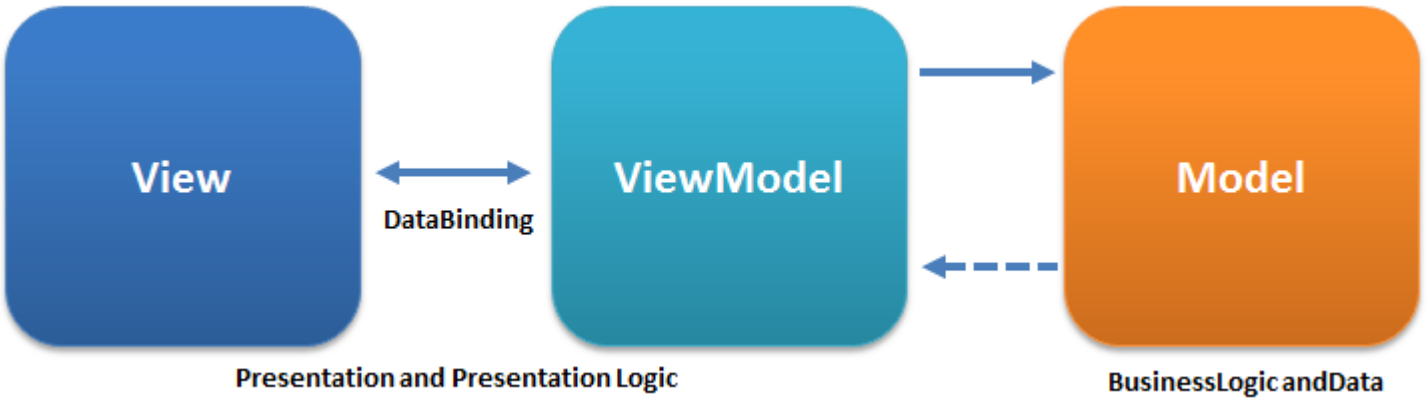
DEFT and WEB U/I Development

- Key development points –
  - Agile methodology: continuous meetings with the main users and often releases;
  - Using open source;
  - «Model View ViewModel» approach;
- Django is used on server side (Django ORM as Model)
- JavaScript(jQuery, AngularJS) is used on a client side

ORACLE django jQuery

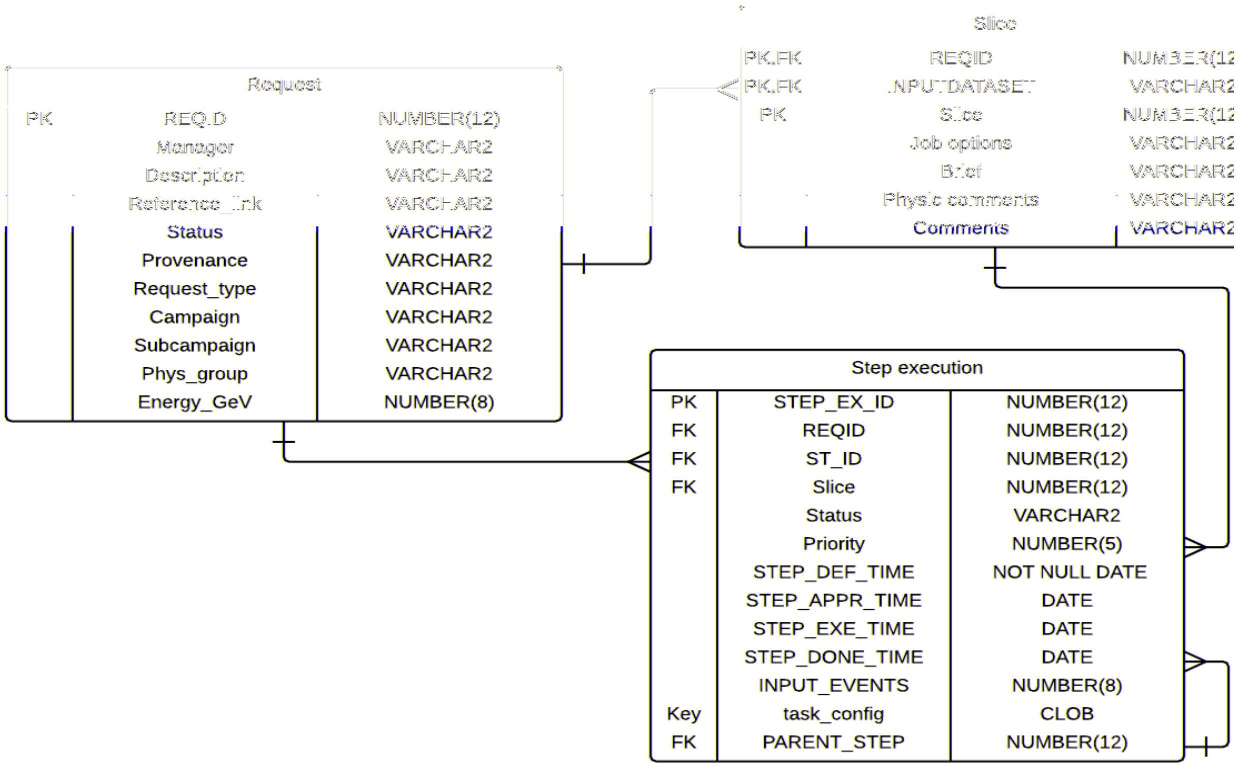
Server and client logic (View and ViewModel)

- “ViewModel” maintains access to data in a easy or an “used to” for concrete group of users way:
  - Most of functionality is shared, so writing code in reusable way is extremely important.
  - Using third open source library where it’s possible.
- “View” mostly repeats Model on client side
  - It includes checks which could be done on client side;
  - Optimized to be used for quite big amount of manageable data (thousands tasks on one page).



Production system data model

- Model is represented by multilevel relational instances:
  - Request -> Slice(chain of steps) -> Step -> Task
  - Depends on workflow each instance could play a role of a template;
  - Tasks are created by initiating step instance.



Request

PK	REQID	NUMBER(12)
	Manager	VARCHAR2
	Description	VARCHAR2
	Reference_link	VARCHAR2
	Status	VARCHAR2
	Provenance	VARCHAR2
	Request_type	VARCHAR2
	Campaign	VARCHAR2
	Subcampaign	VARCHAR2
	Phys_group	VARCHAR2
	Energy_GeV	NUMBER(8)

Slice

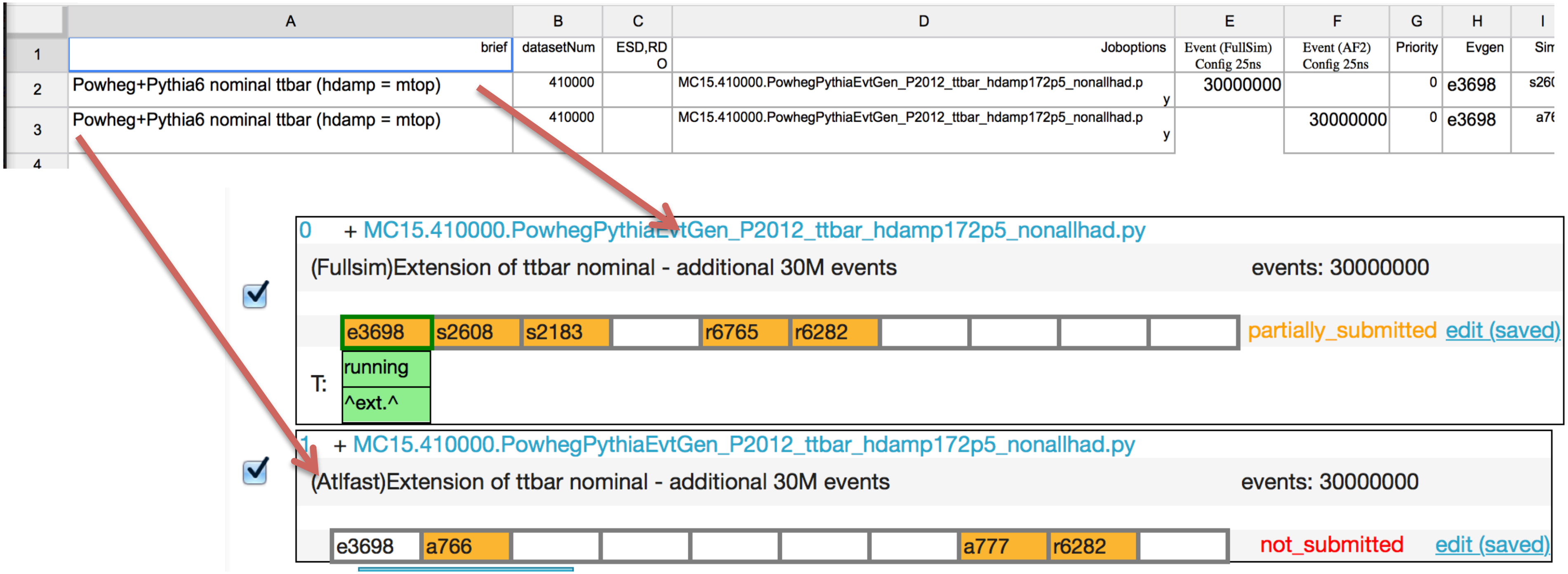
PK,FK	REQID	NUMBER(12
PK,FK	INPUTDATASET	VARCHAR2
PK	Slice	NUMBER(12
	Job options	VARCHAR2
	Brief	VARCHAR2
	Physic comments	VARCHAR2
	Comments	VARCHAR2

Step execution

PK	STEP_EX_ID	NUMBER(12)
FK	REQID	NUMBER(12)
FK	ST_ID	NUMBER(12)
FK	Slice	NUMBER(12)
	Status	VARCHAR2
	Priority	NUMBER(5)
	STEP_DEF_TIME	NOT NULL DATE
	STEP_APPR_TIME	DATE
	STEP_EXE_TIME	DATE
	STEP_DONE_TIME	DATE
	INPUT_EVENTS	NUMBER(8)
Key	task_config	CLOB
FK	PARENT_STEP	NUMBER(12)

MC production request creation

- Creating of production request is one of the examples how different workflows can be integrated to the system
- For MC Google spreadsheet were used by user to provide a data for MC tasks. In ProdSys2 it was adopted so user can submit input data in the same format as they use before.



	A	B	C	D	E	F	G	H	I
1	brief	datasetNum	ESD,RDO	Joboptions	Event (FullSim) Config 25ns	Event (AF2) Config 25ns	Priority	Evgen	Sim
2	Powheg+Pythia6 nominal ttbar (hdamp = mtop)	410000		MC15.410000.PowhegPythiaEvtGen_P2012_ttbar_hdamp172p5_nonallhad.py	30000000		0	e3698	s260
3	Powheg+Pythia6 nominal ttbar (hdamp = mtop)	410000		MC15.410000.PowhegPythiaEvtGen_P2012_ttbar_hdamp172p5_nonallhad.py		30000000	0	e3698	a76
4									

☑ 0 + MC15.410000.PowhegPythiaEvtGen_P2012_ttbar_hdamp172p5_nonallhad.py

(Fullsim)Extension of ttbar nominal - additional 30M events events: 30000000

e3698	s2608	s2183		r6765	r6282				

partially_submitted edit (saved)

T: running
^ext.^

☑ 1 + MC15.410000.PowhegPythiaEvtGen_P2012_ttbar_hdamp172p5_nonallhad.py

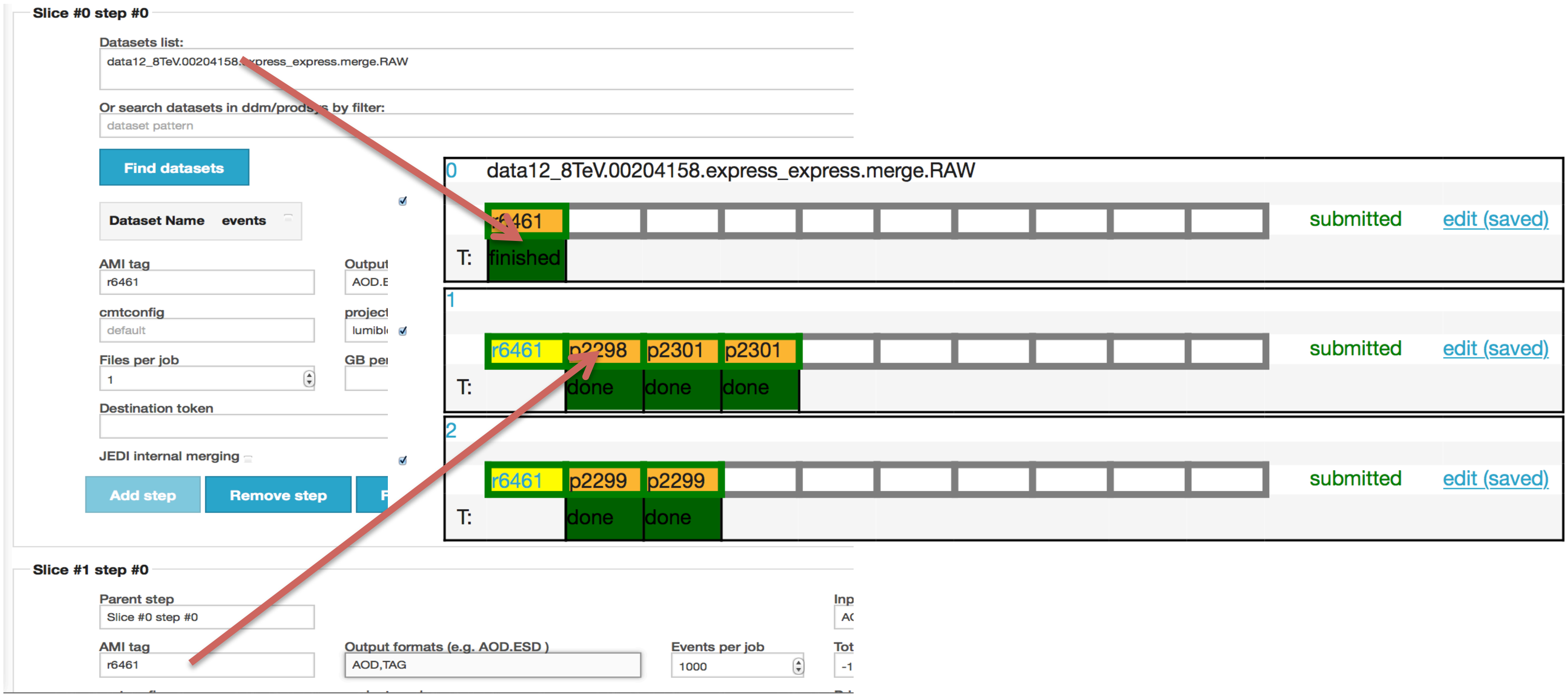
(Atlfast)Extension of ttbar nominal - additional 30M events events: 30000000

e3698	a766						a777	r6282	

not_submitted edit (saved)

Reprocessing production request creation

- Reprocessing workflow has a tree structure, where output of one task can be an input for several more tasks – interface for creating such structure was developed for it.



Slice #0 step #0

Datasets list: data12_8TeV.00204158.express_express.merge.RAW

Or search datasets in ddm/prodsys by filter: dataset pattern

Dataset Name events

AMI tag: r6461

Output: AOD.E

cmtconfig: default

project: lumibl ☑

Files per job: 1

GB per

Destination token

JEDI internal merging ☑

Slice #1 step #0

Parent step: Slice #0 step #0

AMI tag: r6461

Output formats (e.g. AOD.ESD): AOD,TAG

Events per job: 1000

Inp: AC

Tot: -1

0 data12_8TeV.00204158.express_express.merge.RAW

r6461 | submitted | edit (saved)
T: finished

1

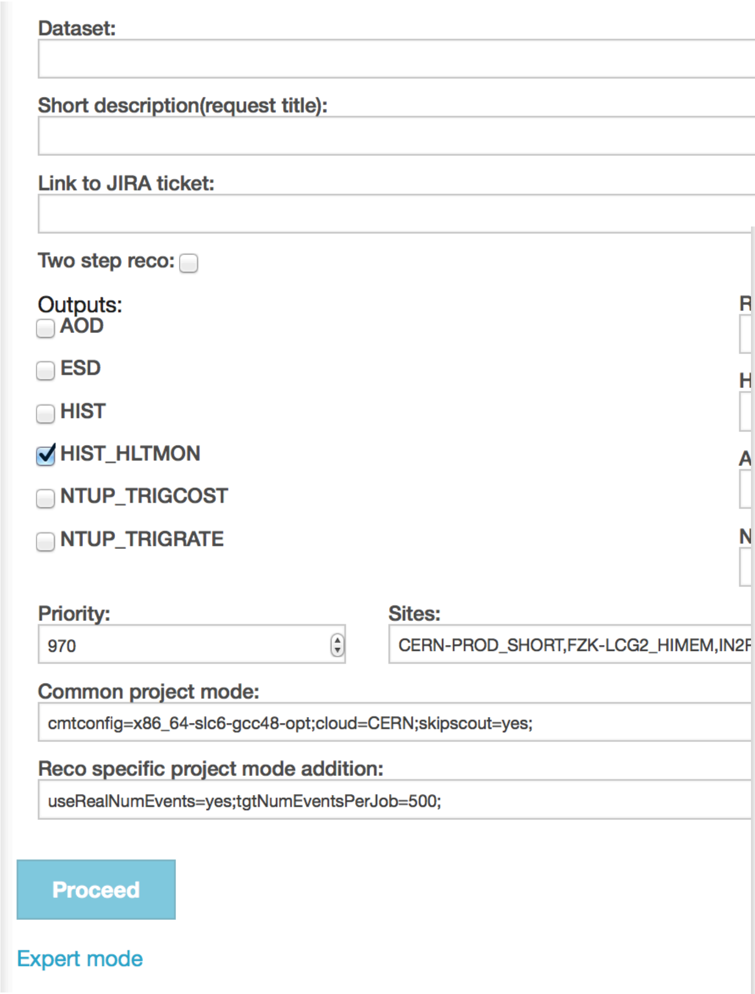
r6461 | p2298 | p2301 | p2301 | submitted | edit (saved)
T: done | done | done

2

r6461 | p2299 | p2299 | submitted | edit (saved)
T: done | done

HLT production request creation

- HLT reprocessing has a well defined workflow, so interface for HLT production request creation includes only a few fields.



Dataset:

Short description(request title):

Link to JIRA ticket:

Two step reco: □

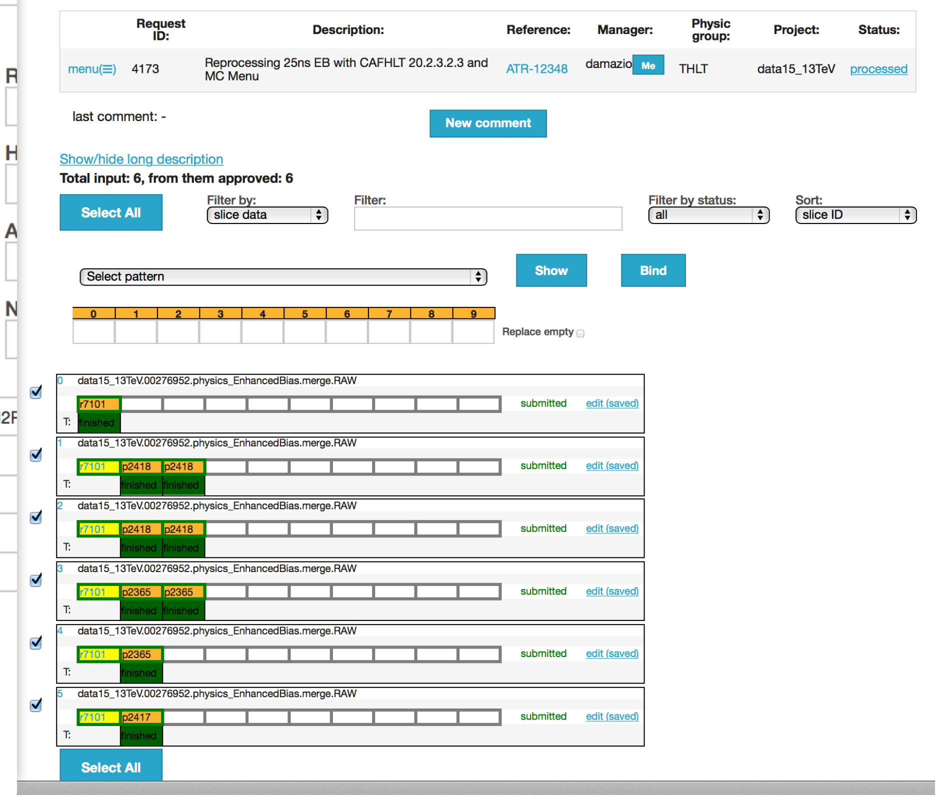
Outputs:
□ AOD
□ ESD
□ HIST
☑ HIST_HLTMON
□ NTUP_TRIGCOST
□ NTUP_TRIGRATE

Priority: 970

Sites: CERN-PROD_SHORT,FZK-LCG2_HIMEM,IN2P

Common project mode: cmtconfig=x86_64-slc6-gcc48-opt;cloud=CERN;skipscout=yes;

Reco specific project mode addition: useRealNumEvents=yes;tgtNumEventsPerJob=500;



	Request ID:	Description:	Reference:	Manager:	Physic group:	Project:	Status:
menu(≡)	4173	Reprocessing 25ns EB with CAFHLT 20.2.3.2.3 and MC Menu	ATR-12348	damazio	THLT	data15_13TeV	processed

last comment: -

Show/hide long description

Total input: 6, from them approved: 6

Filter by: slice data

Filter:

Filter by status: all

Sort: slice ID

Select pattern

Replace empty □

☑ 0 data15_13TeV.00276952.physics_EnhancedBias.merge.RAW
r7101 — submitted — edit (saved)
T: finished

☑ 1 data15_13TeV.00276952.physics_EnhancedBias.merge.RAW
r7101 p2418 p2418 — submitted — edit (saved)
T: finished finished

☑ 2 data15_13TeV.00276952.physics_EnhancedBias.merge.RAW
r7101 p2418 p2418 — submitted — edit (saved)
T: finished finished

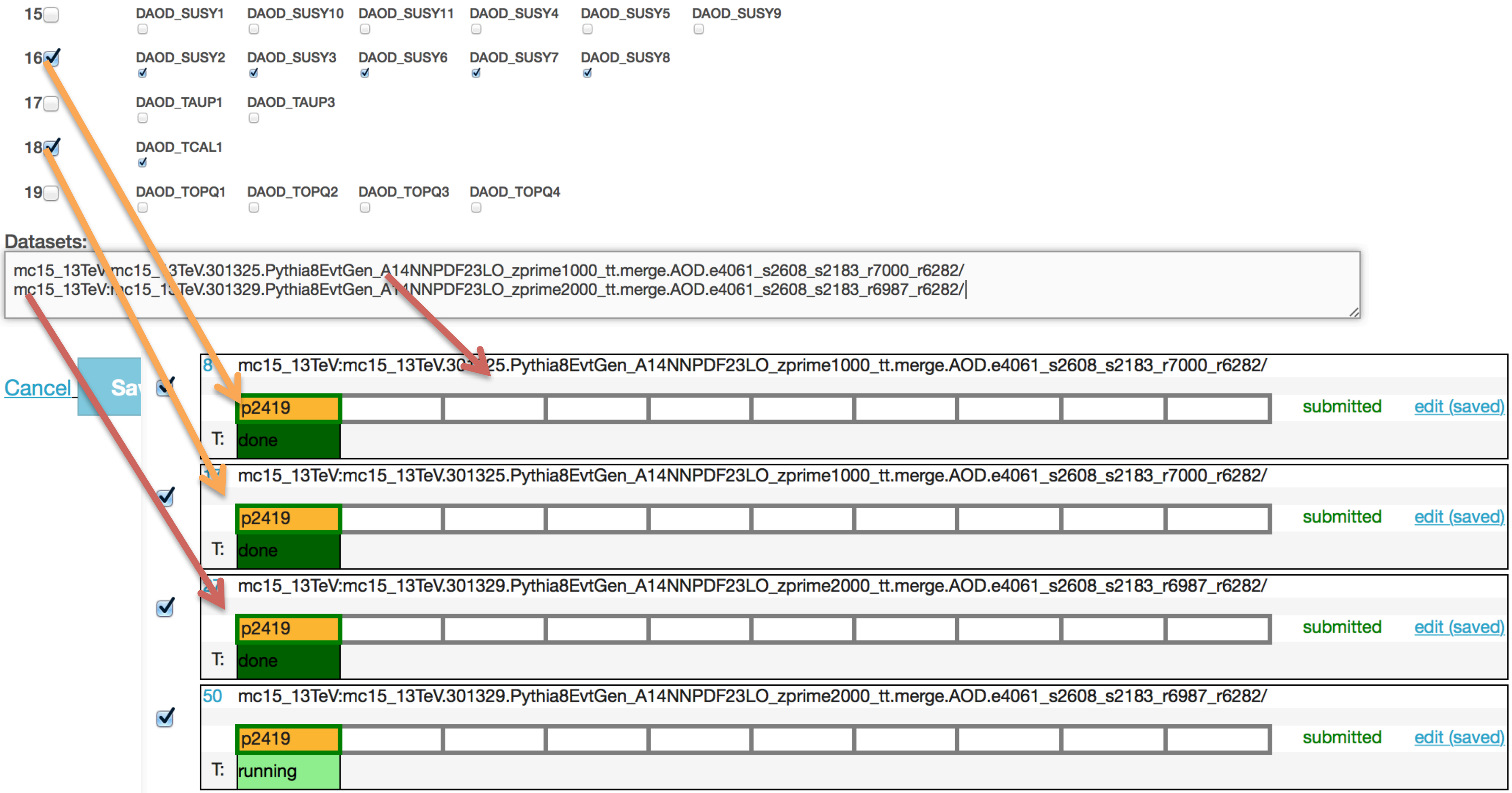
☑ 3 data15_13TeV.00276952.physics_EnhancedBias.merge.RAW
r7101 p2365 p2365 — submitted — edit (saved)
T: finished finished

☑ 4 data15_13TeV.00276952.physics_EnhancedBias.merge.RAW
r7101 p2365 — submitted — edit (saved)
T: finished

☑ 5 data15_13TeV.00276952.physics_EnhancedBias.merge.RAW
r7101 p2417 — submitted — edit (saved)
T: finished

Derivation production request creation

- Derivation is using so called “train” model, there each input runs on some of many predefined outputs. To manage with complexity, pattern request is created first, and system is using this pattern to generate possible options for the specific derivation production request creation interface.



15 □ DAOD_SUSY1 □ DAOD_SUSY10 □ DAOD_SUSY11 □ DAOD_SUSY4 □ DAOD_SUSY5 □ DAOD_SUSY9 □

16 ☑ DAOD_SUSY2 ☑ DAOD_SUSY3 ☑ DAOD_SUSY6 ☑ DAOD_SUSY7 ☑ DAOD_SUSY8 ☑

17 □ DAOD_TAUP1 □ DAOD_TAUP3 □

18 ☑ DAOD_TCAL1 ☑

19 □ DAOD_TOPQ1 □ DAOD_TOPQ2 □ DAOD_TOPQ3 □ DAOD_TOPQ4 □

Datasets:

mc15_13TeV:mc15_13TeV.301325.Pythia8EvtGen_A14NNPDF23LO_zprime1000_tt.merge.AOD.e4061_s2608_s2183_r7000_r6282/
mc15_13TeV:mc15_13TeV.301329.Pythia8EvtGen_A14NNPDF23LO_zprime2000_tt.merge.AOD.e4061_s2608_s2183_r6987_r6282/

☑ 8 mc15_13TeV:mc15_13TeV.301325.Pythia8EvtGen_A14NNPDF23LO_zprime1000_tt.merge.AOD.e4061_s2608_s2183_r7000_r6282/
p2419 submitted edit (saved)
T: done

☑ 17 mc15_13TeV:mc15_13TeV.301325.Pythia8EvtGen_A14NNPDF23LO_zprime1000_tt.merge.AOD.e4061_s2608_s2183_r7000_r6282/
p2419 submitted edit (saved)
T: done

☑ 27 mc15_13TeV:mc15_13TeV.301329.Pythia8EvtGen_A14NNPDF23LO_zprime2000_tt.merge.AOD.e4061_s2608_s2183_r6987_r6282/
p2419 submitted edit (saved)
T: done

☑ 50 mc15_13TeV:mc15_13TeV.301329.Pythia8EvtGen_A14NNPDF23LO_zprime2000_tt.merge.AOD.e4061_s2608_s2183_r6987_r6282/
p2419 submitted edit (saved)
T: running

Work with production request

- Tasks requests Web U/I provides many general and experiment specific features:
  - **Bookkeeping**.
  - **Approve 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 failed because of many permanent (e.g. bug in software) and temporal (storage is down) reasons. To be able quickly understand root of the problem and fix it by redefining the task is one of the major feature of the production system.
  - **Chaining** one production to the other. E.g. derivation production could be chained to MC or reprocessing task, that significantly speed up them.
  - **Automation** task submission. User can defined a pattern and when new data appears tasks are started automatically.
  - …

Summary

- Production system is a layer which connects distributed computing and physicists in a user-friendly way.

- Development of the ProdSys2 would be impossible without experience gained from Run1 operation and input provided by experts, who run production for many years.