PanDA@LCF. Experience and possible evolution

Danila Oleynik (UTA), Sergey Panitkin (BNL), Taylor Childers (ANL)

ATLAS TIM 2014

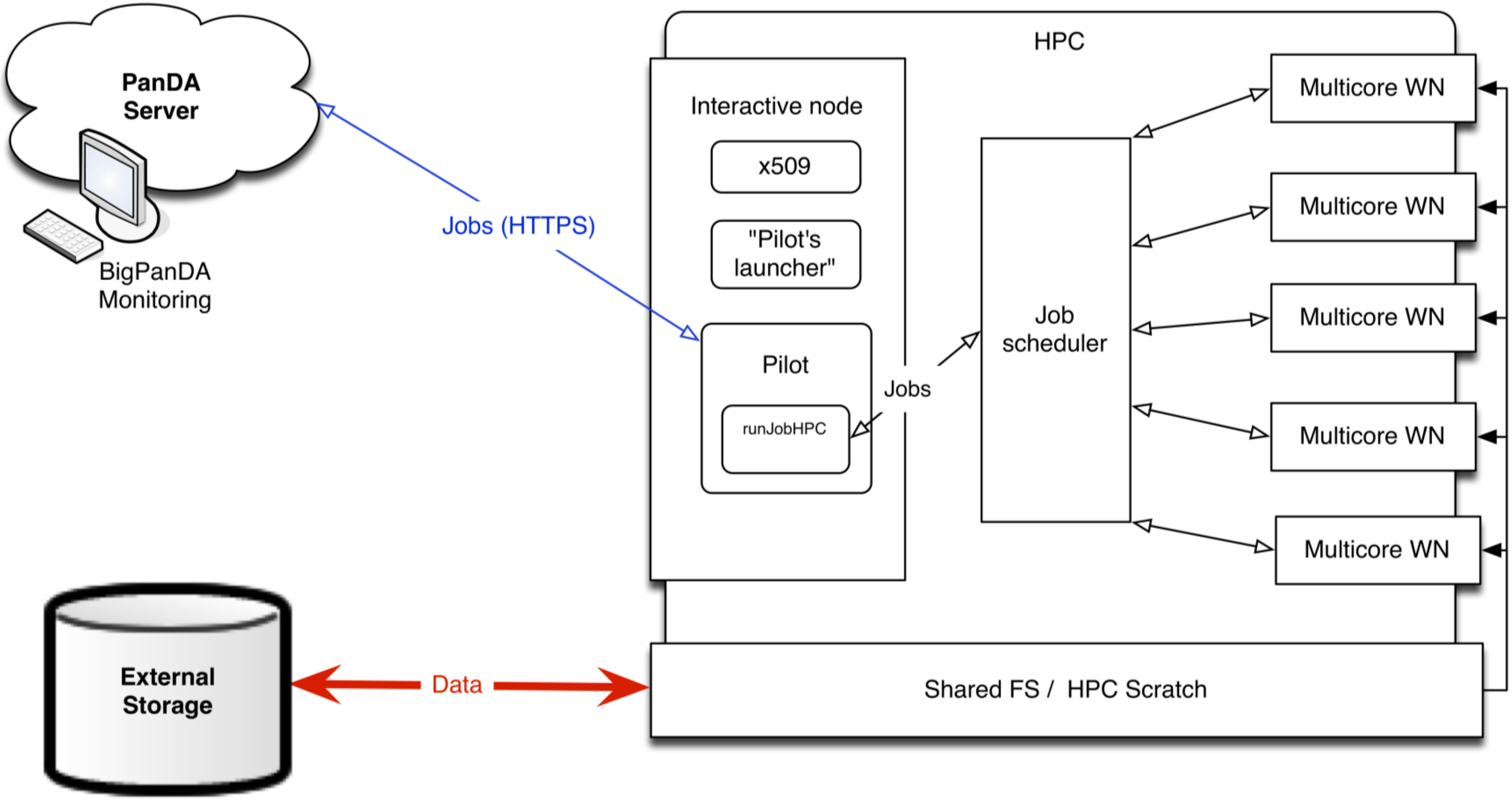
Outline

- Overview of current status
  - PanDA@LCF(HPC) architecture
  - PanDA@OLCF/NERSC (Titan, Hopper, Edison)
  - EventService on HPC
  - PanDA@ALCF (Mira)
- New trends
  - Data management
  - Software installation
  - Flexible tactic on HPC
- Workloads @ OLCF(Sergey)

Common features of HPC’s

- Restricted/no external access to computing nodes
- Payload execution management only through a local batch system
- Special treatment of shared file systems
- No pledged resources (even if you have allocation, that doesn’t correlate with priority of your tasks)
- Restricted configuration, and highly restricted system management.

PanDA@HPC (1/2)

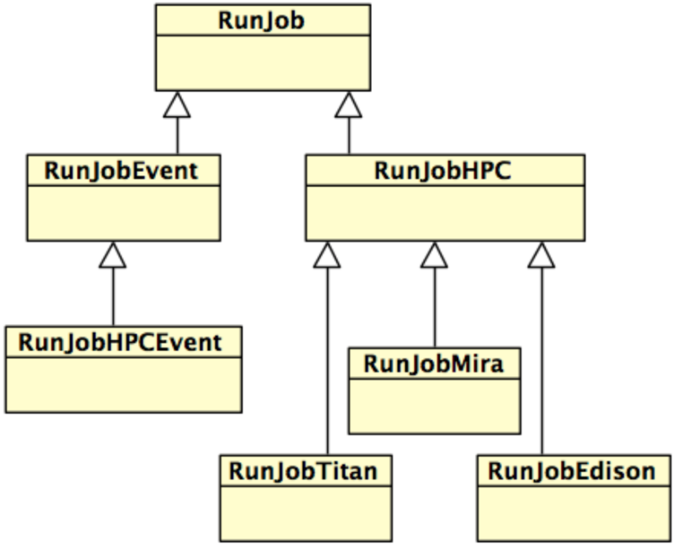

PanDA Server
BigPanDA Monitoring
Jobs (HTTPS)
HPC
Interactive node
x509
"Pilot's launcher"
Pilot
runJobHPC
Jobs
Job scheduler
Multicore WN
Multicore WN
Multicore WN
Multicore WN
Multicore WN
External Storage
Data
Shared FS / HPC Scratch

PanDA@HPC (2/2)

- Pilot(s) executes on interactive node (or some edge node)
  - Allow all necessary connections with PanDA server
- Pilot interacts with local job scheduler to manage job
  - Build on high level abstraction layer (SAGA) for supporting wide range of batch system backends
  - MPI wrapper/overlay scripts that allow to run multiple “single node” workload instances in parallel on multiple HPC nodes. We do not modify workloads.
- Number of executing pilots should be equal or less than number of available slots in local scheduler
  - Increase efficiency of usage of HPC
- Stage in/out procedures goes through dedicated OLCF facility – Data Transfer Nodes (DTN)
  - Speedup transfers

PanDA Pilot plugins model

- ATLAS modular pilot augmented with HPC specific classes
- RunJob class hierarchy
  - RunJob: used for running “normal” jobs
  - RunJobEvent: used for event service jobs on a grid site
  - RunJobHPC: common HPC code
    - HPC specifics are implemented in RunJobTitan, RunJobMira, ..
  - RunJobHPCEvent: ES capable module running on HPC Front- End
    - Knows how to submit jobs into an HPC using method from relevant RunJobHPC* class;
- Proper class selection through shchedconfig parameter



Inheritance diagram

RunJobTitan/Hopper/Edison

- Dedicated classes for execution of MPI payloads through HPC batch system
  - SAGA API used as an interface with HPC batch manager (PBS)
    - Close work with SAGA developers, for proper encapsulation of variety of PBS dialects
  - Instrumented for use of ‘backfill’ resources:
    - Special function collects information about available resources (number of free nodes and availability time) from MOAB
      - In plans: moving this functionality to SAGA API
    - PBS job parameters are formed according to available resources, queue policies and minimal requirements of PanDA job (min. number of nodes and min. walltime)
    - Introduced PBS wait time limit and retry mechanism

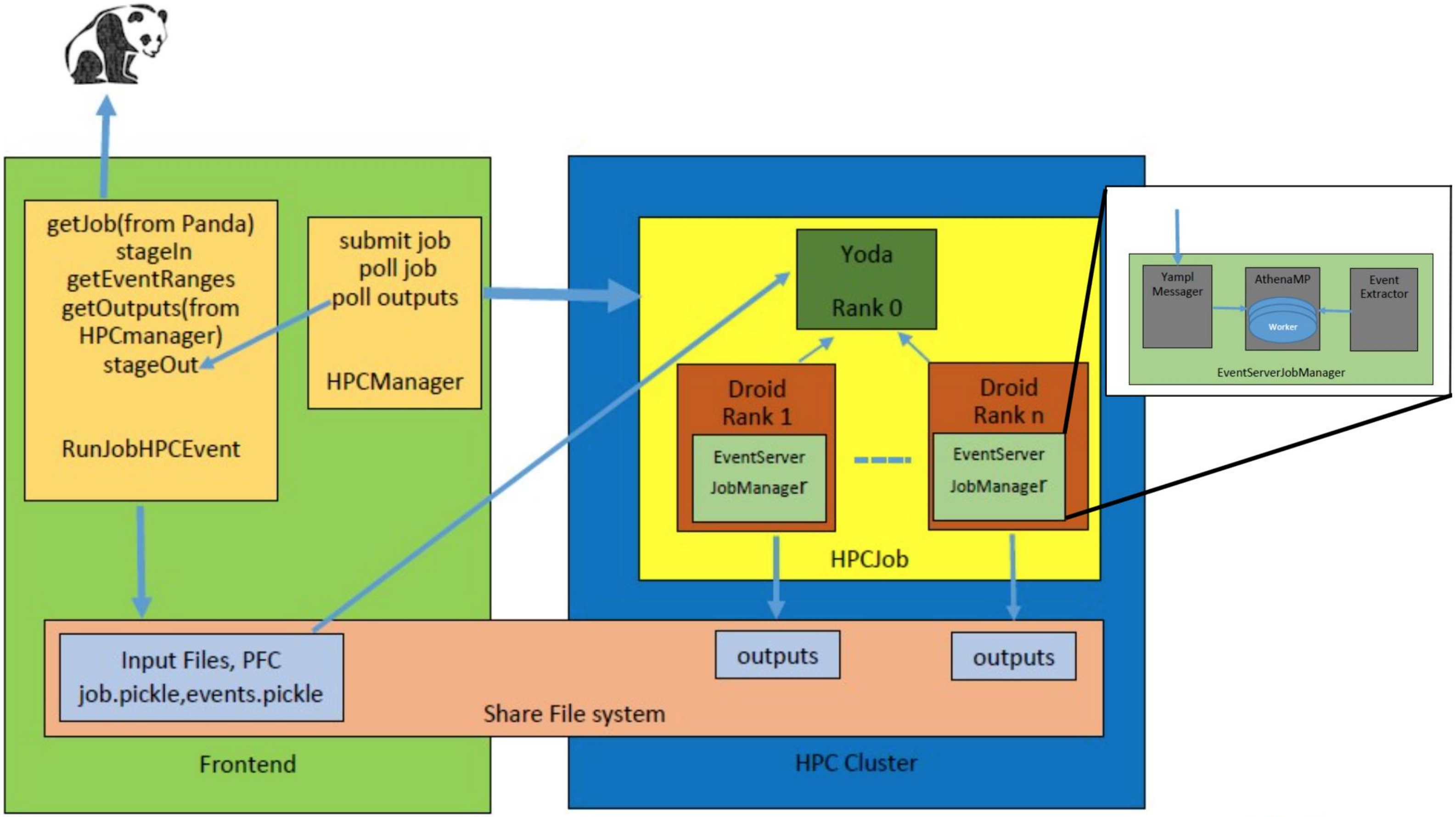
Alpha testing

Event Service on HPCs (1/2)

- NormalHPCjob
  - Pilot running on HPC Front-End node launches RunJobHPC subprocess (RunJobTitan/Hopper/Edison/Mira/..) which submits the actual jobs into the HPC
- EventServiceHPCjob
  - Pilot running on HPC Front-End node launches RunJobHPCEvent subprocess which in turn
    - Downloads event ranges from server using normal https calls
      - Job definition and event ranges are placed in pickle files in standard location
  - Creates TAG files from EVGEN files, the PFC and the file containing EVGEN GUID to TAG file mappings to be used by Token Extractors and places them in standard location
  - Submits the Yoda suite into the HPC
    - Script launched as Rank 0: Yoda = Light weight JEDI
    - Script launched as Rank N: Droids/Droid launcher (which in turn runs AthenaMP)
  - Periodically looks for updated event range file from Yoda and updates server
    - Final heartbeat sent when all event ranges are processed

Alpha testing

Event Service on HPCs (2/2)



W. Guan

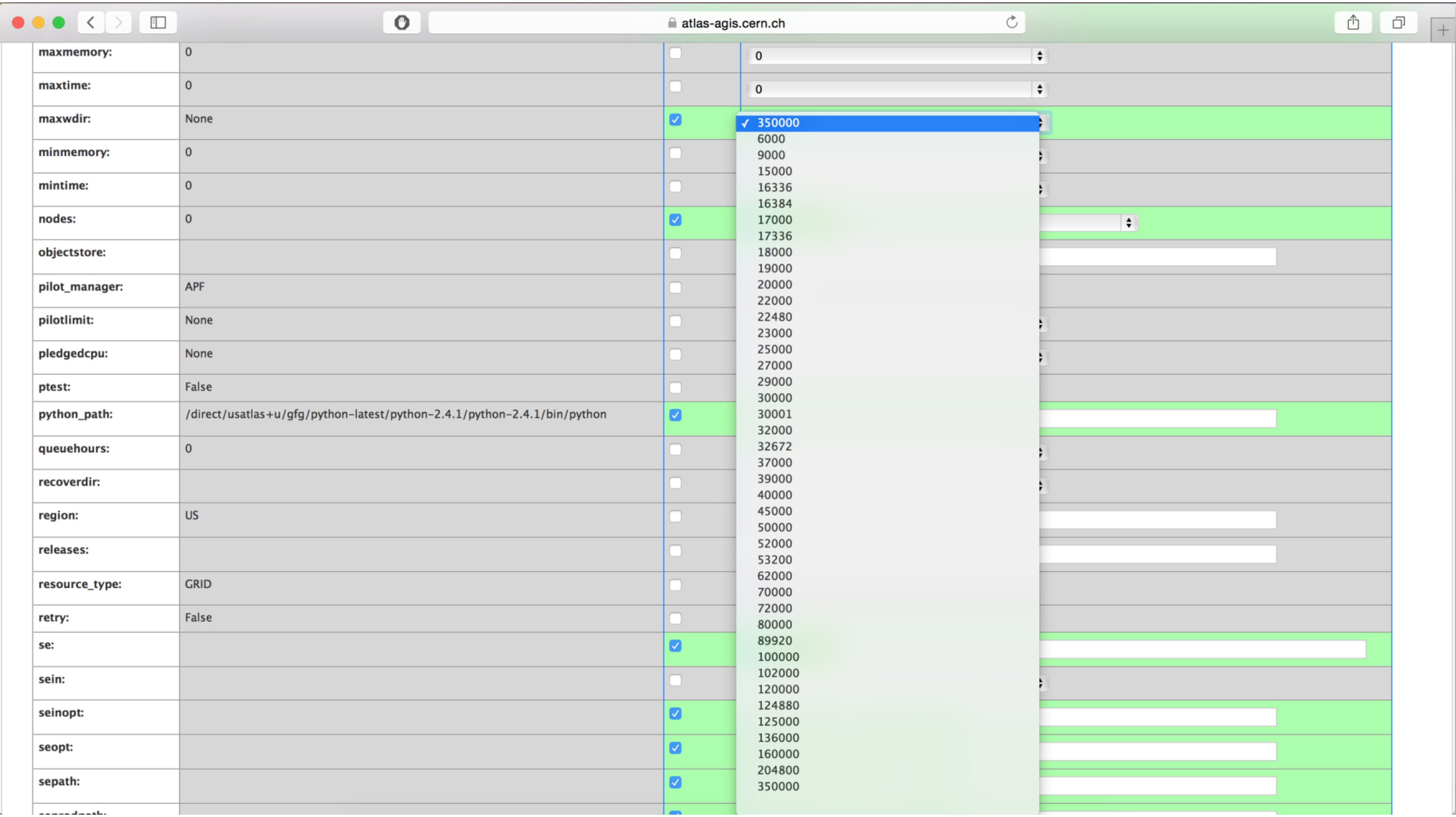
PanDA@ALCF

- Mira is not x86 compatible machine:
  - Focus on standalone applications: Alpgen, Sherpa
- Integration with PanDA
  - RunJobMira class development in progress
  - Pilot will submit jobs to ARGO/Balsam and receive status info, then finally the output dataset
- Backfill studies are progressing, have filled 2/3 of Mira with Alpgen in backfill.

New trends. Data management

- Real BigData: special policy for data organization required, starting from understanding of new limitations ;-)
- Single HPC job can use
  - Dozens of Tb of transient disk space
  - Up to hundreds of Gb of transient data during execution
  - Possibility of huge output
- Special treatments for IO, starting from Pilot itself
  - Change POSIX ‘ls *’ to platform oriented ‘lfs find’ etc.
  - Optimization of cleanup functions

Current AGIS UI limit on pilot's sandbox size



atlas-agis.cern.ch

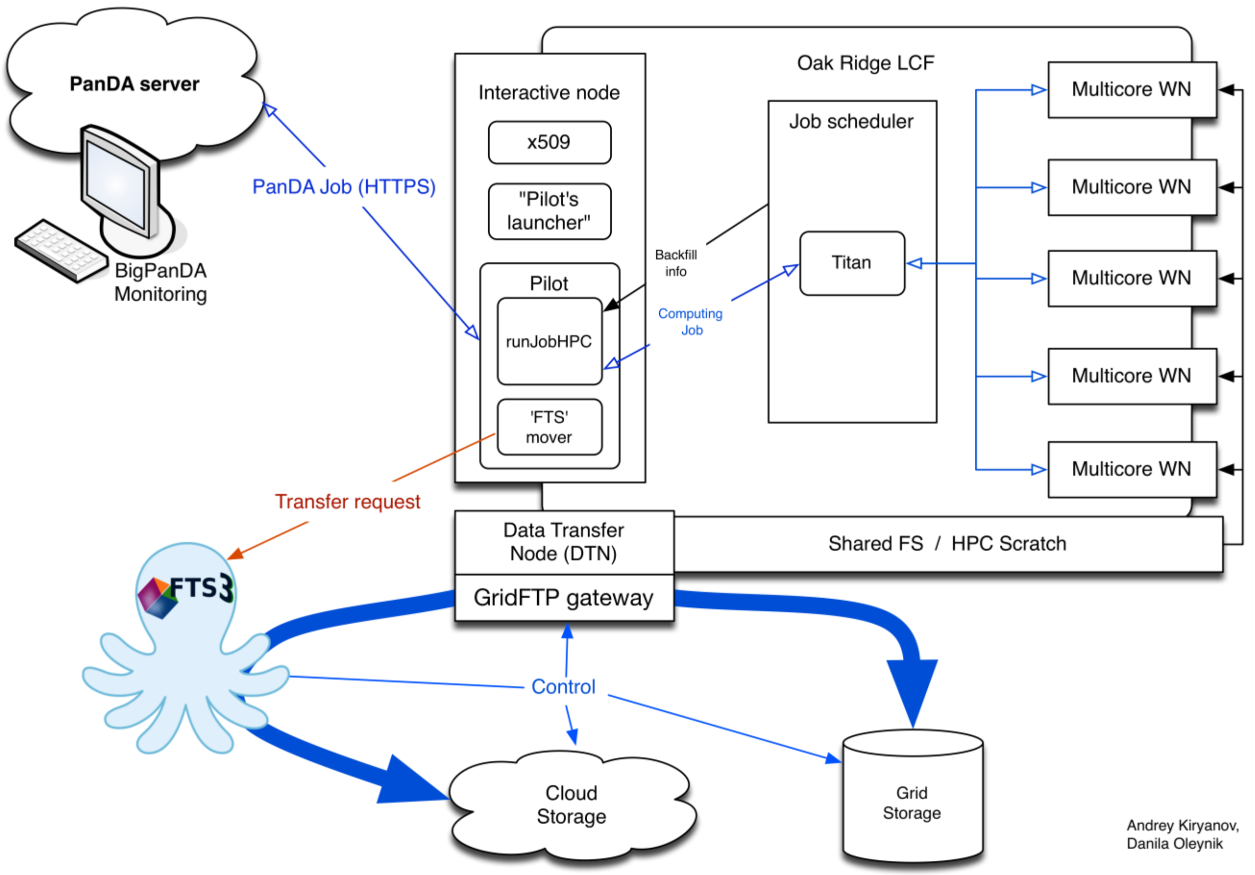
maxmemory:	0	☐	0
maxtime:	0	☐	0
maxwdir:	None	☑	350000
minmemory:	0	☐	
mintime:	0	☐	
nodes:	0	☑	
objectstore:		☐	
pilot_manager:	APF	☐	
pilotlimit:	None	☐	
pledgedcpu:	None	☐	
ptest:	False	☐	
python_path:	/direct/usatlas+u/gfg/python-latest/python-2.4.1/python-2.4.1/bin/python	☑	
queuehours:	0	☐	
recoverdir:		☐	
region:	US	☐	
releases:		☐	
resource_type:	GRID	☐	
retry:	False	☐	
se:		☑	
sein:		☐	
seinopt:		☑	
seopt:		☑	
sepath:		☑	

✓ 350000
6000
9000
15000
16336
16384
17000
17336
18000
19000
20000
22000
22480
23000
25000
27000
29000
30000
30001
32000
32672
37000
39000
40000
45000
50000
52000
53200
62000
70000
72000
80000
89920
100000
102000
120000
124880
125000
136000
160000
204800
350000

Hey, why only 350G?! I would like to have 2-5Tb limit for working directory on Titan! ;-)

New trends. Data management

- Proposed use of FTS as data transfer tool at OLCF



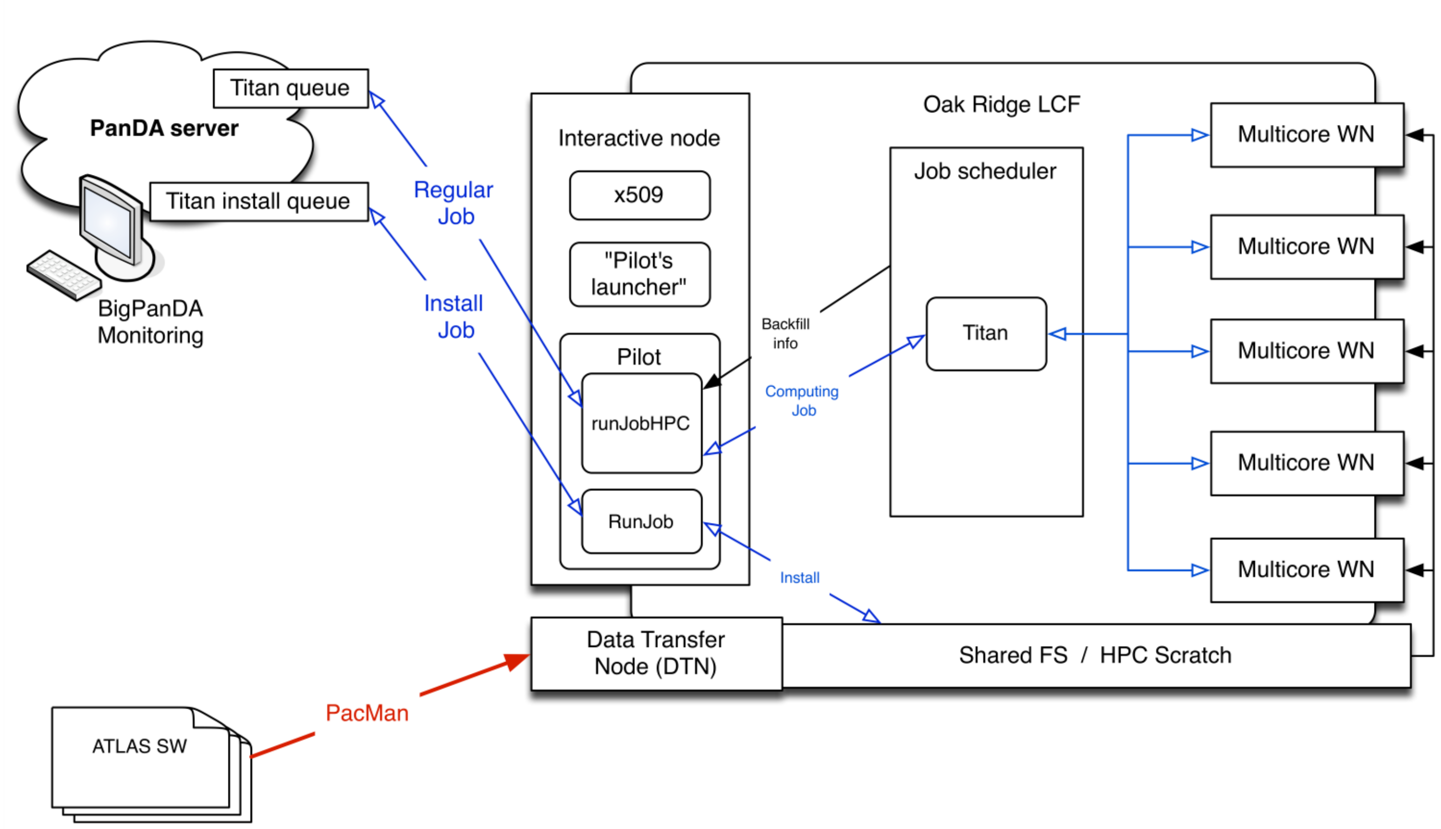
Andrey Kiryanov,
Danila Oleynik

New trends. Data management

- Why (Web)FTS?
  - Reliable file transfers between Grid and Cloud endpoints
  - Modular multi-protocol support
    - SRM, GridFTP, HTTP/WebDAV and XROOT
    - Can relay file transfers between endpoints with different protocols
  - Powerful service to manage transfers with web interface

New trends. Software installation

CVMFS & rsync doesn't work well for Titan (at least for the moment), but nothing new under the sun.



New trends. Software installation

- Installation should be formed as special job in PanDA
- Dedicated PanDA queue (install queue) should be used for this type of jobs, to avoid execution on CN
- Processing of this type of jobs should be done on login nodes (like regular grid job), execution backend will be known from queue parameter

New trends. Flexible tactics.

- ‘Backfill’ is great opportunity… unfortunately it’s not always in place or job's requirements are not fitted for this type of treatment, and regular HPC submission can be more efficient.
  - Adaptive algorithm for covering this issue in development
- In ‘backfill mode’ is not possible to predict how many cores/nodes will be involved in processing. For some workflows it’s important to have this information for proper adjustment of parameters in next submission.
  - This information can be provided to PanDA as soon as job execution started.

Conclusion

- OLCF and NERSC were integrated with PanDA using common pilot architecture (differences only to account for different site policies)
  - ALCF will use the same architecture
- Many standalone workloads tested
  - Alpgen, MadGraph, Powheg, Pythia, Geant4, Root
  - Production scale EvGen runs at ALCF and OLCF
- Significant progress with ATLAS SW on x86 compatible HPC machines (Titan, Hopper, Edison).
  - ATLAS releases installation via pacman
  - AthenaMP, Reco.trf tested
- Next steps:
  - Tests with multiple simultaneously Pilots
  - Improving of algorithm for resource grabbing
  - Integration with production system

Acknowledgments

- Kaushik De
- Vakhtang Tsulaia
- Paul Nilsson & Wen Guan
- Andrey Kiryanov (FTS Team)

OLCF. Payloads (Sergey Panitkin (BNL))