Integration of PanDA workload management system with Titan supercomputer at OLCF

K. De (UTA), A. Klimentov (BNL), D. Oleynik (UTA), S. Panitkin (BNL),
A. Petrosyan (UTA), J. Schovancova (BNL), A. Vaniachine (ANL), T. Wenaus (BNL)
For the ATLAS Collaboration

CHEP 2015 Okinawa, Japan, April 13-17, 2015

Outline

- Introduction and motivation
- PanDA workload management system (WMS)
- PanDA architecture for Titan
- PanDA Pilot with backfill capability
- MPI wrapper
- Workloads
- Summary

ATLAS and Supercomputers

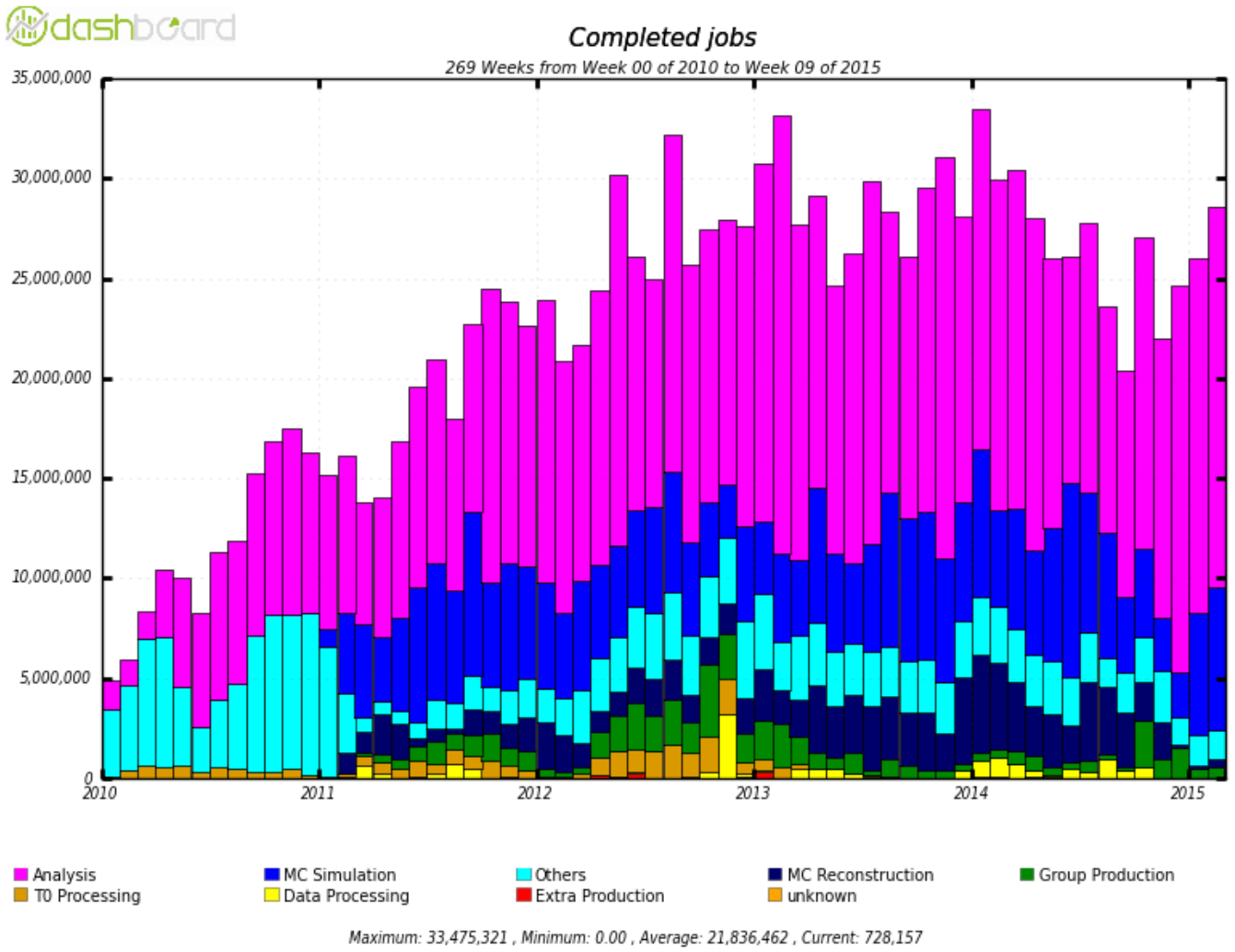
- Current pace of research and discovery is limited by ability of the ATLAS computing Grid to generate Monte-Carlo events - ”Grid luminosity limit”
  - Currently O(100k) cores available to ATLAS worldwide, ¾ dedicated to MC production.
  - Still not enough CPU power !
  - Many physics simulation requests have to wait for many months
- Supercomputers are rich source of CPUs
- ATLAS initiated R&D project aimed at integration of supercomputing and HPC resources into ATLAS distributed computing
- DOE ASCR supported project aimed at integration of PanDA WMS with Titan supercomputer at OLCF is part of this effort

PanDA in ATLAS

- ATLAS uses PanDA Workload Management System (WMS) to run jobs on WLCG
- PanDA - **P**roduction **an**d **D**ata **A**nalysis WMS

- Goal: An automated yet flexible WMS which can optimally make distributed resources accessible to all users
  - Adopted as the ATLAS wide WMS in 2008 (first LHC data in 2009) for all computing applications
  - Currently PanDA successfully manages O(10E2) sites, O(10E5) cores, O(10E8) jobs per year, serving O(10E3) users per year
  - PanDA is exascale now: 1.2 Exabytes of data processed by PanDA in 2013

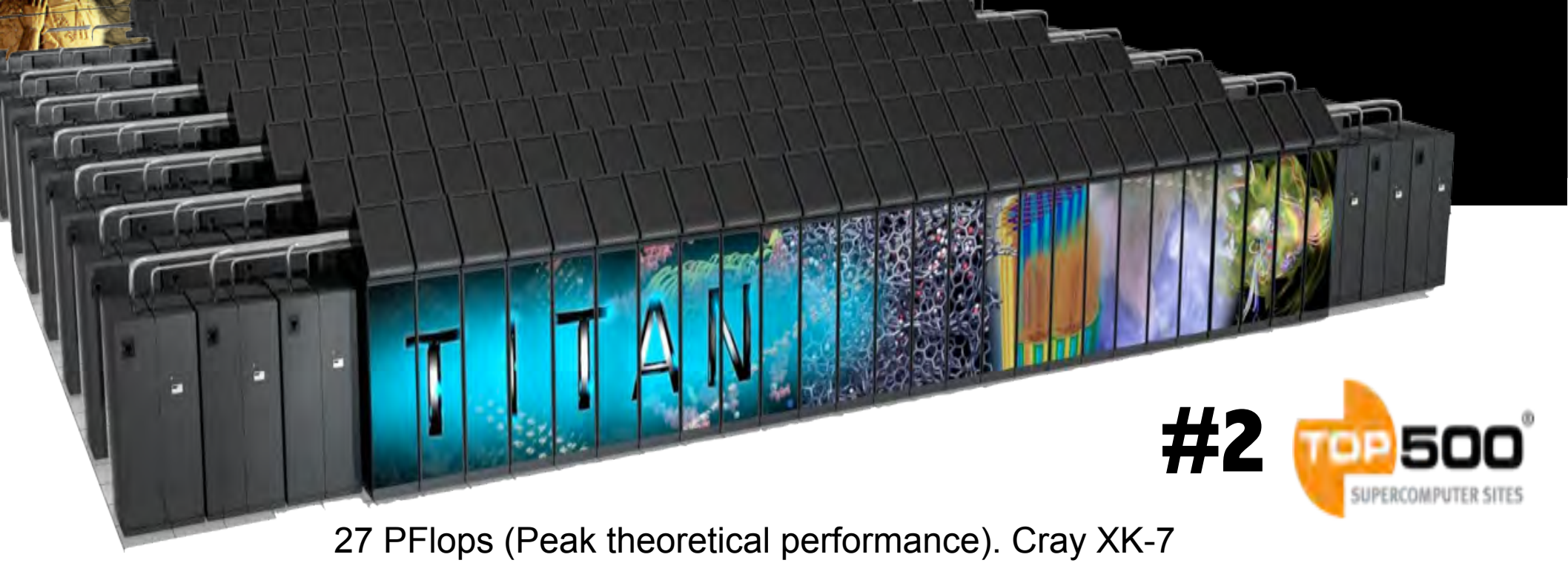
PanDA Performance



Current scale – 25M jobs completed every month at more than a hundred of sites

Key Features of PanDA

- Pilot based job execution system
  - Pilot manages job execution on local resources, as well as data movement for the job
  - Payload is sent only after pilot execution begins on CE
  - Minimize latency, reduce error rates
- Modular design
- Central job queue
  - Unified treatment of distributed resources
  - SQL DB keeps state - critical component
- Automatic error handling and recovery
- Extensive monitoring
- HTTP/S RESTful communications
- GSI authentication
- Use of Open Source components
- Workflow is maximally asynchronous



#2 TOP500 SUPERCOMPUTER SITES

27 PFlops (Peak theoretical performance). Cray XK-7
18,688 compute nodes with GPUs

299,008 CPU cores

AMD Opteron 6200 @2.2 GHz (16 cores per node)

32 GB RAM per node

NVidia TESLA K20x GPU per node

32 PB disk storage (center-wide Luster file system)

>1TB/s aggregate FS throughput

29 PB HPSS tape archive

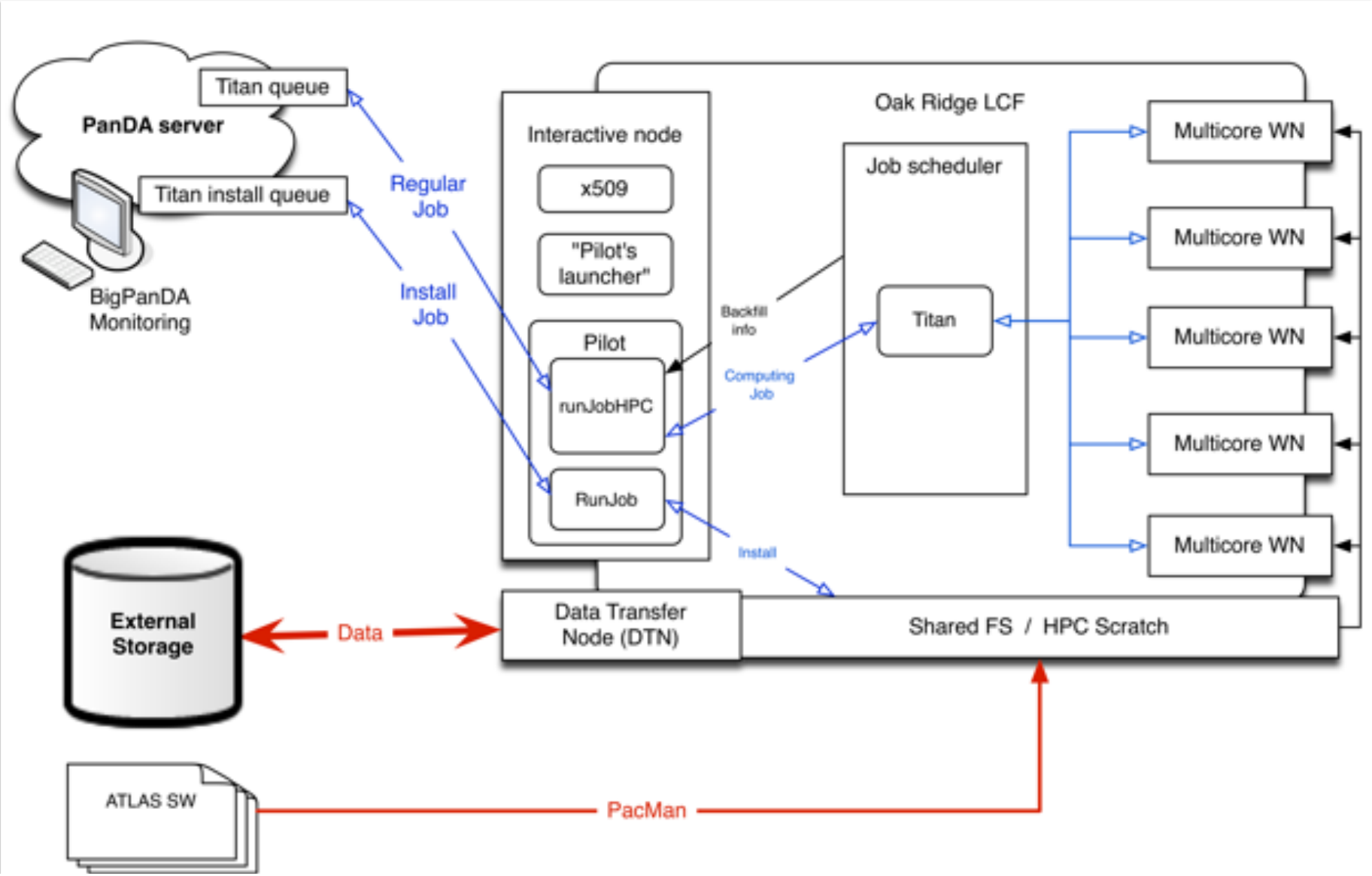
Some Titan features that affect integration with PanDA

- Highly restricted access. One-time password interactive authentication
  - No portals, gatekeepers, VO boxes. Pilot needs to run on Titan's login nodes
- No network connectivity from worker nodes to the outside world
  - Pilot can not run on worker nodes, needs a new mechanism for batch workload management
- Limit on number of submitted jobs in batch queue per user and limit on number of running jobs per user
  - Sequential submissions of single node jobs is not an option
  - Have to use MPI in some form!
- Specialized OS (SUSE based CNL) and software stack
- Highly competitive time allocation. Geared toward leadership class projects and very big jobs
  - Creates opportunity for backfill

PanDA setup on Titan

- Main idea - try to reuse existing PanDA components and workflow logic as much as possible
- Modified PanDA pilot runs on Titan's front end nodes, in user space
- All connections to PanDA servers at CERN or EC2 are initiated from the front end nodes by PanDA Pilot over HTTPS
- For local HPC batch interface use SAGA-Python (Simple API for Grid Applications) framework by Rutgers U. group
  - http://saga-project.github.io/saga-python/
  - http://www.ogf.org/documents/GFD.90.pdf
- Custom light-weight Python MPI wrapper scripts for running (single node) workloads in parallel on multiple multi-core WN
- Pilot instrumented to utilize information about free nodes on Titan
- Software is installed/ported in advance on Titan shared file system

PanDA setup on Titan



MPI wrapper for workloads

- In order to use Titan efficiently we have to use MPI
- We utilize light-weight Python MPI wrapper, specific to each workload type
- Uses mpi4py Python module
- The wrapper is launched on Titan by PanDA Pilot as MPI job of arbitrary size
- Then each wrapper instance knows its MPI rank and serves as “mini-Pilot”
  - Sets up Titan specific environment – like loading appropriate modules, environment, etc
  - Sets up workload specific environment
  - Creates working directory, copies necessary files to $PWD, creates symlinks, etc
  - Manipulates necessary input files for each rank to ensure uniqueness of every job output (random seeds, input file lists, etc)
  - Launches actual workload as sub-process and waits until it finishes
  - Performs necessary clean up of working directory or post-processing, if needed
- The wrapper allows to run simultaneously, arbitrary single-threaded or multi-threaded, non-MPI workloads on multiple multi-core worker nodes on Titan

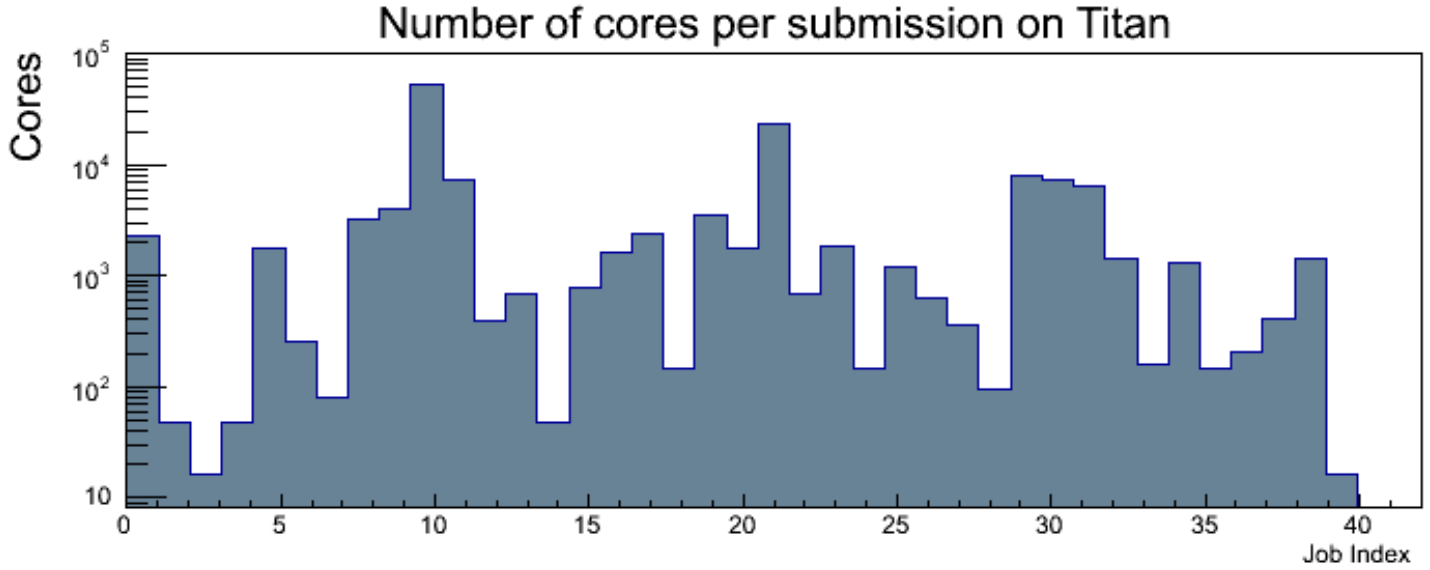
Backfill Enabled Pilot

- Typical LCF facility is ran on average at ~90% occupancy
  - On a machine of the scale of Titan that translates into ~300M unused core hours per year
- Anything that helps to improve this number is very useful
- We added to PanDA Pilot a capability to collect, in near real time, information about current free resources on Titan
  - Both number of free worker nodes and time of their availability
- Based on that information Pilot can define job submission parameters when forming PBS script for Titan, thus tailoring the submission to the available resource.
  - Takes into account Titan's scheduling policies
  - Can also take into account other limitations, such as workload output size, etc
  - Modular architecture, adaptable to other HPC facilities

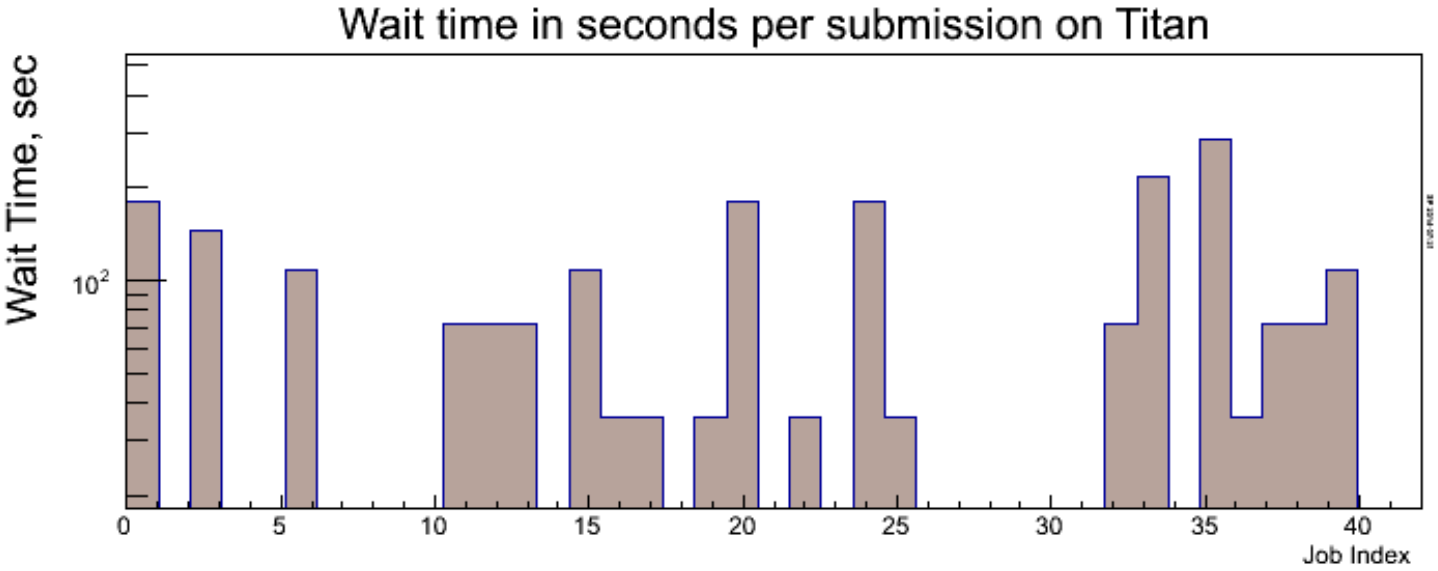
Titan Backfill tests

- We ran multiple continuous job submission tests with PanDA on Titan
- Goals for the tests
  - Test job submission chain and all system components
  - See what works, what breaks, what can be improved
  - Demonstrate that current approach provides tangible improvement in Titan's utilization.
- Through improvements in Pilot's job submission algorithm we have shown that we are able to capture significant CPU resources on Titan even with a single stream of pilots
  - In one of the tests we were able to collect ~200K core hours in 10 hours
  - Max number of nodes per job was 5835 (93360 cores)
    - Close to the entire ATLAS Grid in size!
  - Consistently short wait times for PanDA jobs. (~1min vs several hours)
  - **Used ~2.3% of all Titan core hours or ~14.4% of free core hours**

Example of PanDA test on Titan



- Backfill enabled pilot
- MPI wrapper



Average wait time
70 seconds !

Workloads on Titan

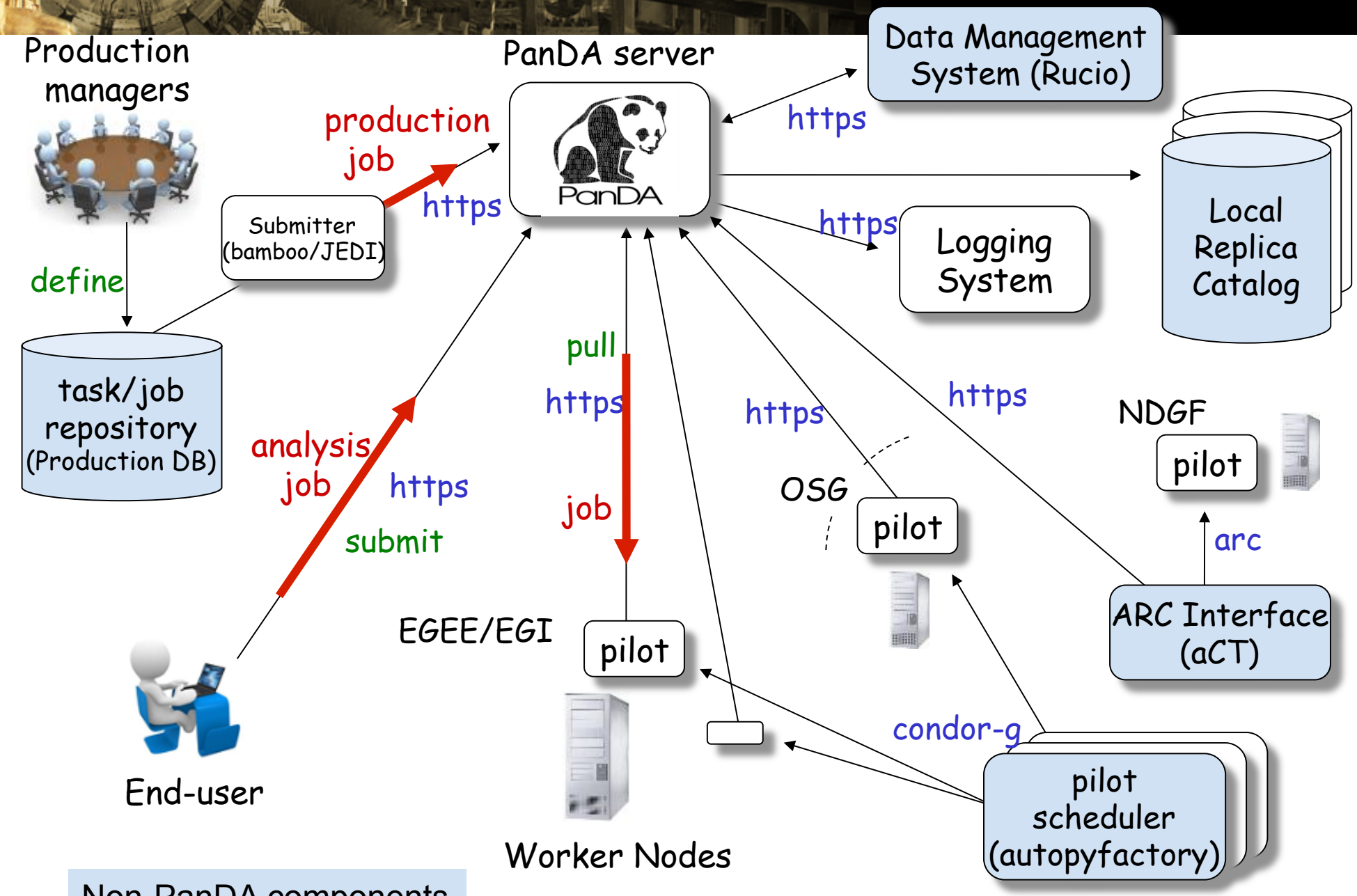
- Many physics packages were ported to Titan
  - Event Generators – SHERPA, MADGRAPH, ALPGEN, POWHEG, PYTHIA
  - Root
  - FairRoot and EICRoot frameworks
  - Geant4, including multithreaded v10
- ATLAS workloads
  - Several ATLAS software releases installed on Titan
  - Event Generation, Geant4 Simulation and Reconstruction chains were ran for several ATLAS physics scenarios
  - Already delivered several million simulated events to ATLAS physicists
  - Integration with the new ATLAS production system (ProdSys II) is in progress

Summary

- Work on integration of Titan with PanDA is well advanced
- PanDA pilot now uses information about free worker nodes on Cray machines for job submission.
- MPI wrappers developed to run unmodified single node workflows as multi-node MPI ensembles
- Ran continuous PanDA job submission tests in backfill mode on Titan
  - Stable operations, Short wait times
  - Demonstrated significant resource collection capability, and improvement in Titan utilization
- Work on ATLAS workloads is in progress
  - Already delivered several million simulated events to ATLAS physicists
  - Integration with ATLAS production system (ProdSys II) is in progress
- Collaboration with multiple groups and experiments
  - ALICE, nEDM, LSST, EIC,…
- Successful functional tests of the setup developed for Titan at NERSC and at IT4I in Ostrava, CZ
- Our project was showcased at SC14 Conference in November, 2014 as part of the US DOE Science Data Pilot Projects

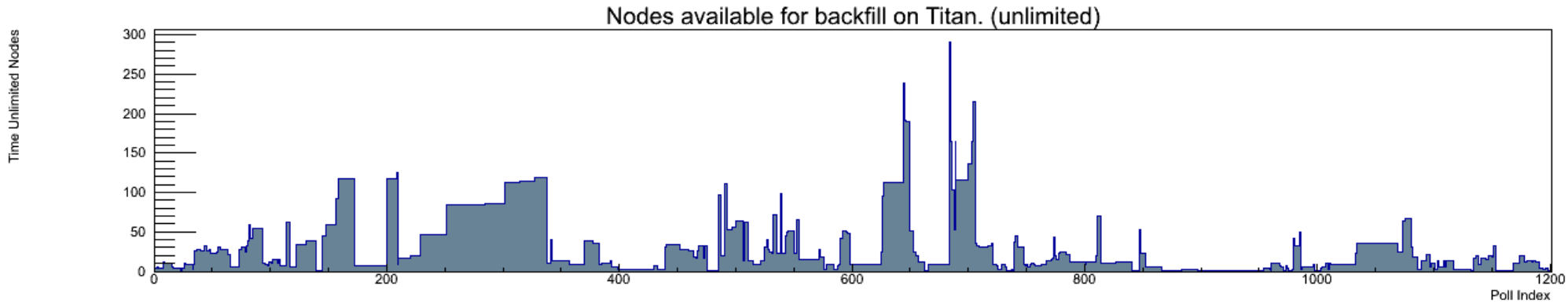
Backup Slides

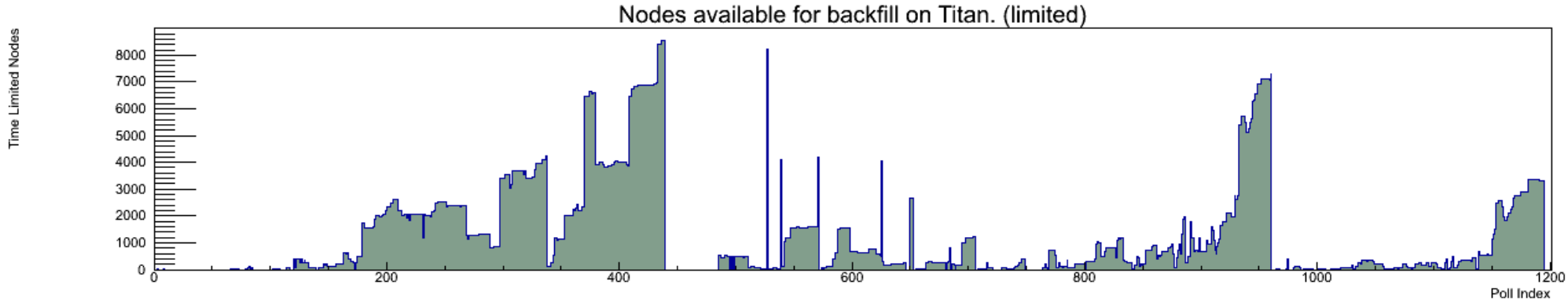
PanDA Workload Management System

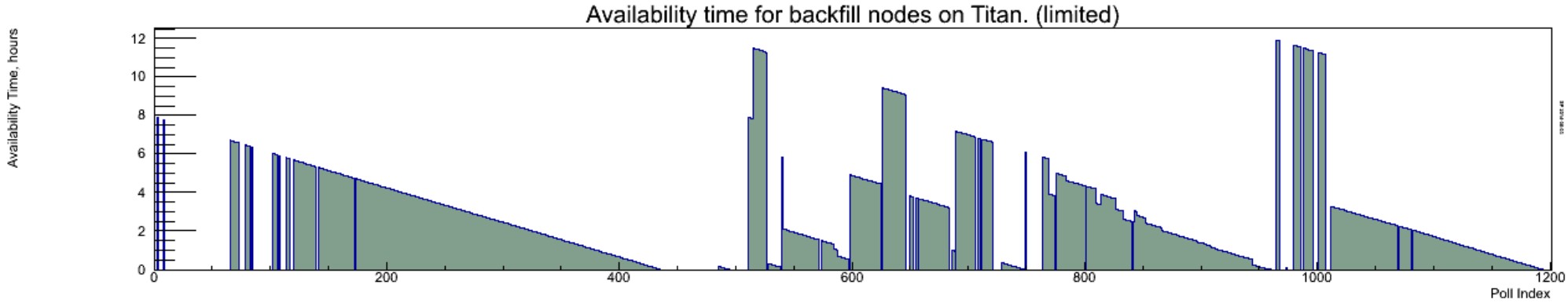


Non-PanDA components

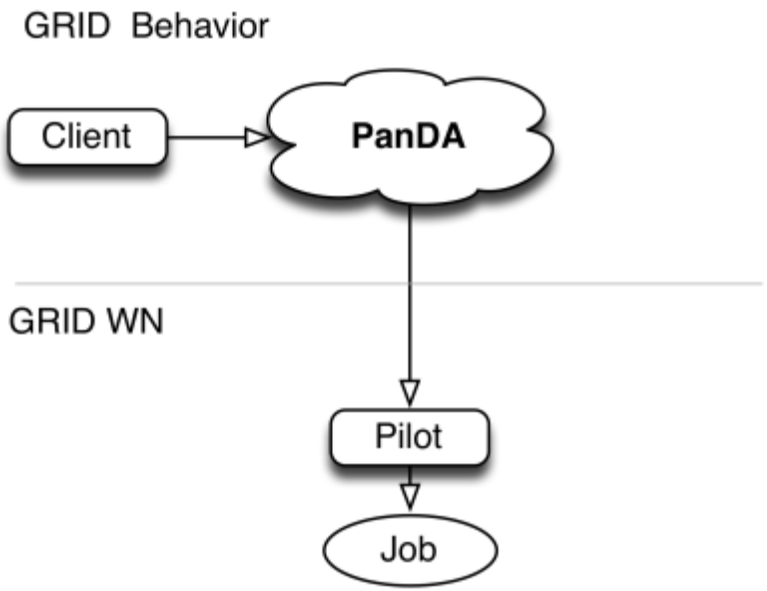
Typical Titan free resource availability pattern

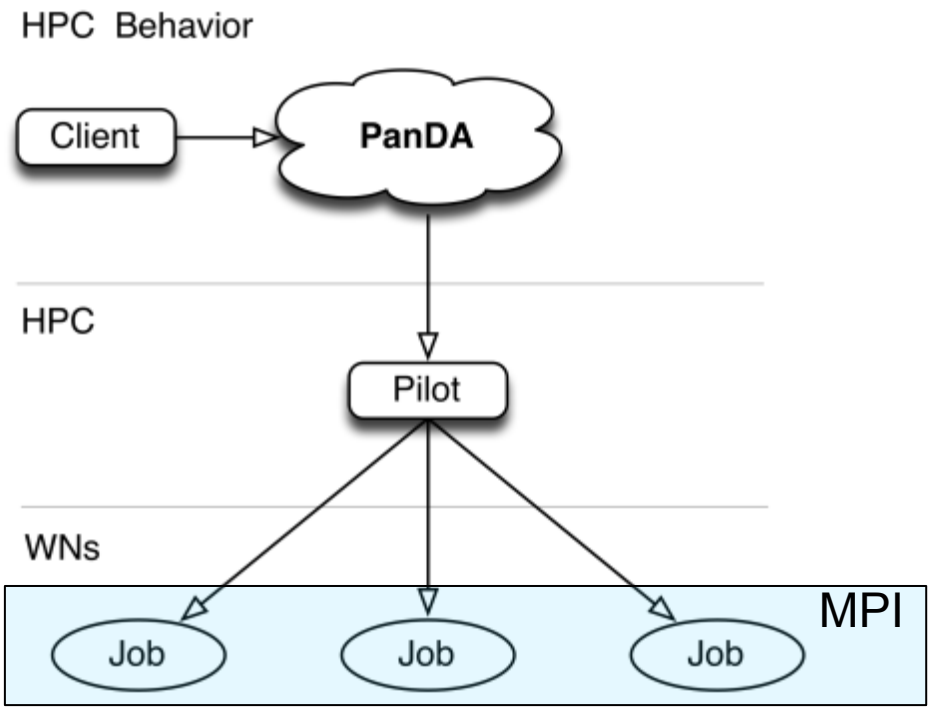






Pilot on HPC with MPI wrapper





"One to One"

"One to Many"