Panda:
Production and Distributed Analysis System for the ATLAS Experiment

Kaushik De

University of Texas at Arlington

The ATLAS Experiment

CHEP06, TIFR

February 15, 2006

Outline

- ATLAS production systems
- Panda overview
- Panda design
- Recent performance
- Current status
- Plan of work
- Conclusion
- More information

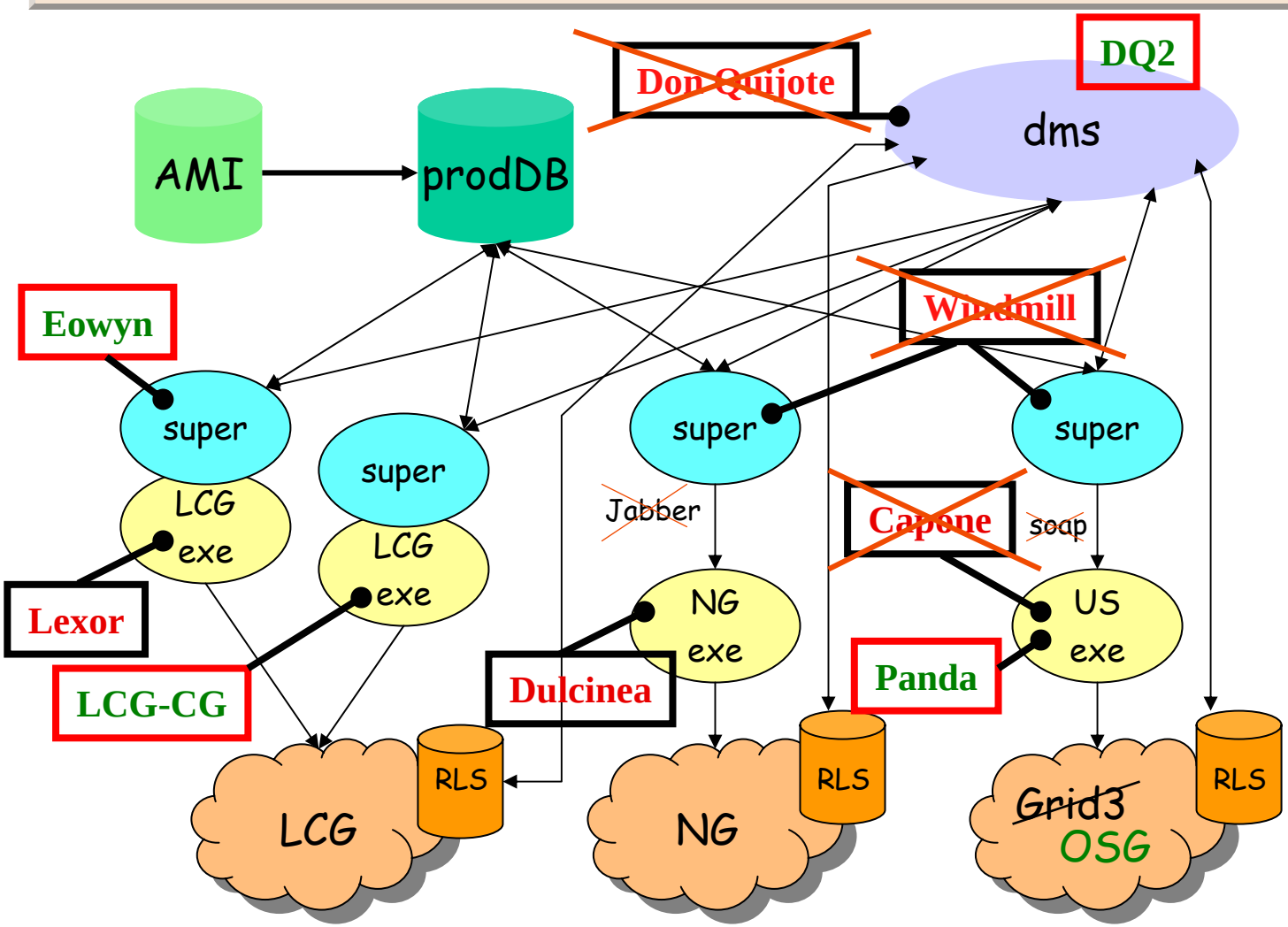
What is Panda

- PanDA – Production and Distributed Analysis system
- Project started Aug 17, 2005
- Baby Panda emerging!
- New system developed by U.S. ATLAS team
  - Rapid development from scratch
  - Leverages DC2/Rome experience
  - Inspired by Dirac & other systems
  - Already in use for CSC pre-production in the U.S.
  - Better scalability/usability compared to DC2 system
  - Will be available for distributed analysis users in few months
- One-stop shopping for all ATLAS users in the U.S.

Why Panda?

- ATLAS used supervisor/executor system for Data Challenge 2 (DC2) and Rome production in 2004-2005
  - Windmill supervisor common for all grids – developed by KD
  - U.S. executor (Capone) developed by UC/ANL team
  - Four other executors were available ATLAS-wide
  - See talks by G. Poulard #111, J. Shank #349, M. Mambelli #35
- Large scale production was very successful on the grid
  - Dozens of different workflows (evgen, G4, digi, pile-up, reco…)
  - Hundreds of large MC samples produced for physics analysis
- DC2/Rome experience led to development of Panda
  - Too labor-intensive (manual fixes for software/grid failures)
  - Could not use all available resources (scaling problems)
  - No distributed analysis system, no data management

ATLAS View of Production Systems

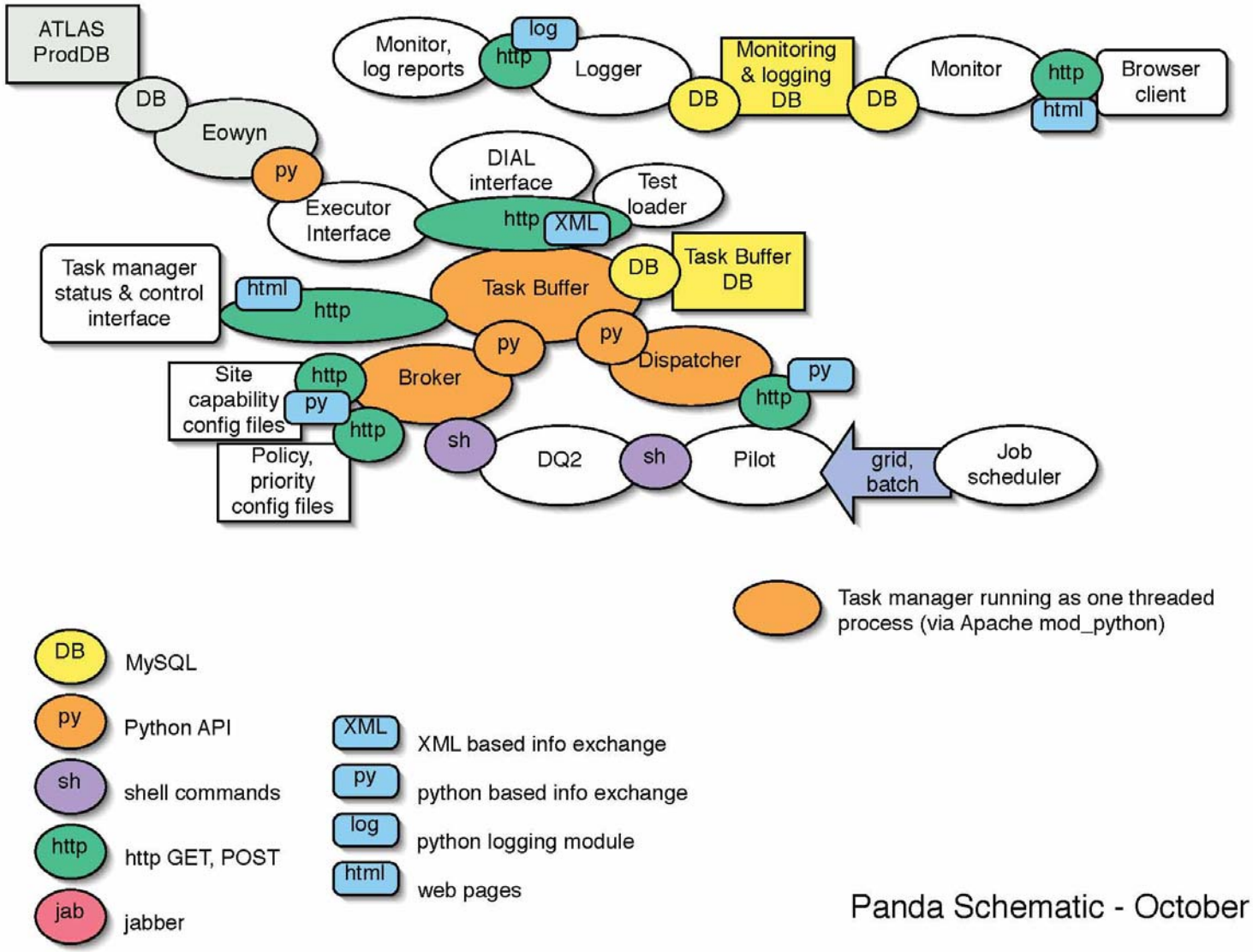


Changes

- DMS: DQ => DQ2
- Supervisor: Windmill (US) => Eowyn (CERN)
- US Executor: Capone => Panda
- Now 2 LCG executors
- Batch exe never used

Panda is the new U.S. executor for ATLAS

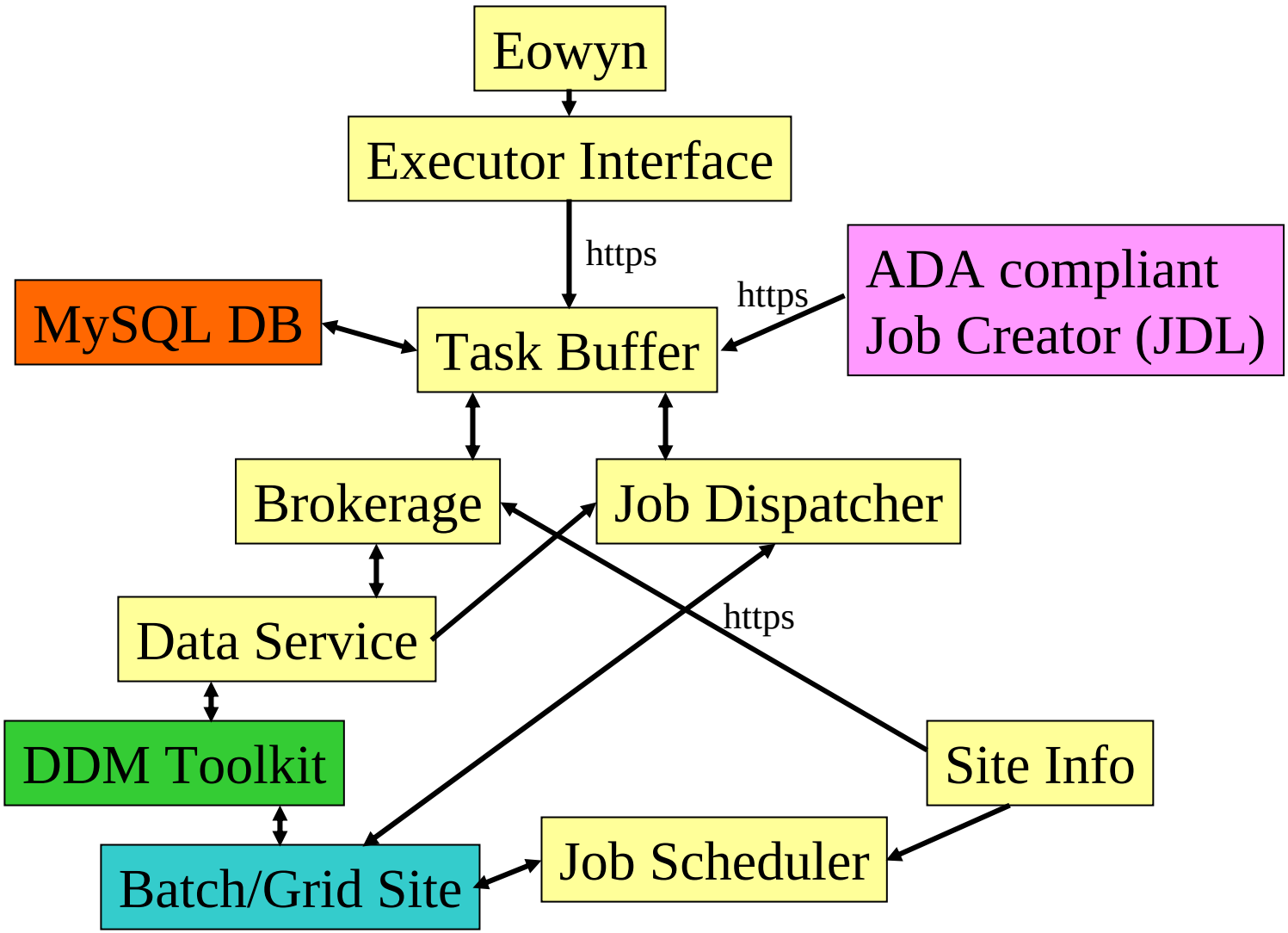
Panda Design



Key Panda Features

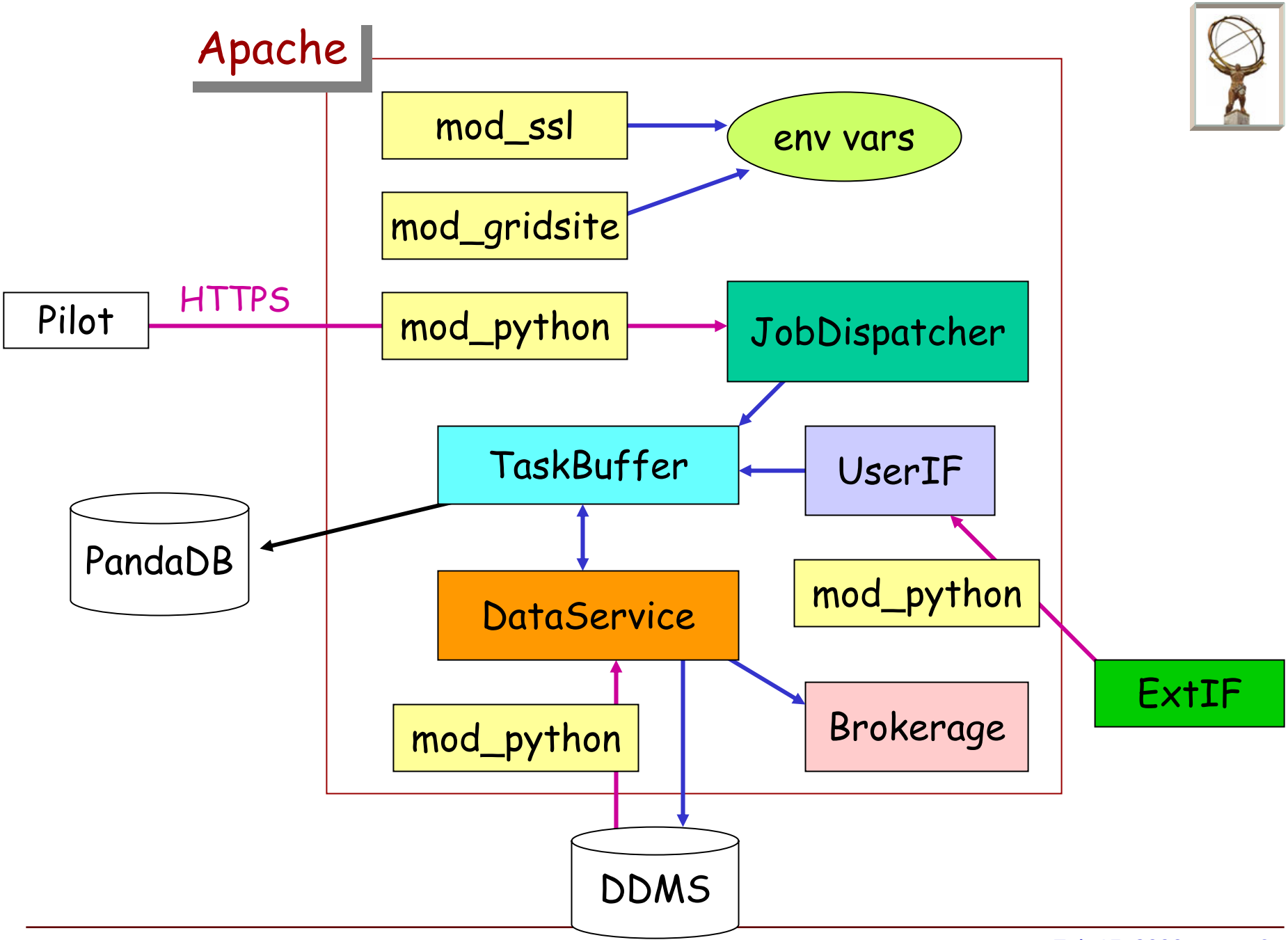
- **Service model** – Panda runs as an integrated service for all ATLAS sites (currently U.S.) handling all grid jobs (production and analysis)
- **Task Queue** – provides batch-like queue for distributed grid resources (unified monitoring interface for production managers and all grid users)
- **Strong data management** (lesson from DC2) – pre-stage, track and manage every file on grid asynchronously, consistent with DQ2 design
- **Block data movement** – pre-staging of output files is done by optimized DQ2 service based on datasets (see talk by D. Cameron #75), reducing latency for distributed analysis (jobs follow the data)
- **Pilot jobs** – are prescheduled to batch systems and grid sites; actual ATLAS job (payload) is scheduled when CPU becomes available, leading to low latency for analysis tasks
- **Support all job sources** – managed or regional production (ATLAS ProdSys), user production (tasks, DIAL, Root, pAthena, scripts or transformations, GANGA…) (see talks by D. Adams #39, D. Liko #263)
- **Support any site** – minimal site requirement: pilot jobs (locally or through grid), outbound http, and integration with DQ2 services

Panda Workflow

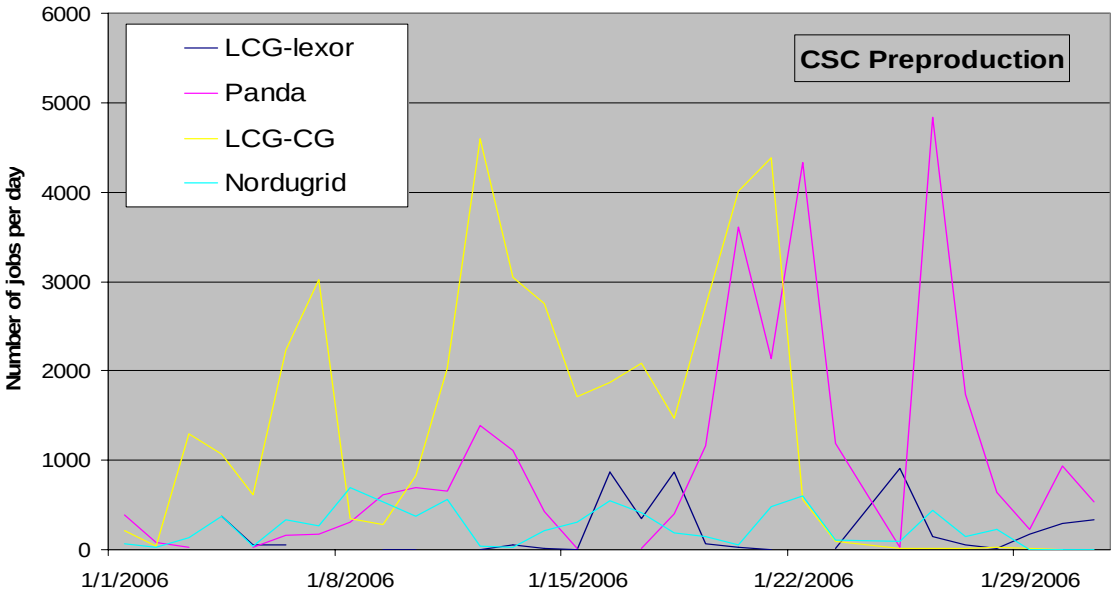


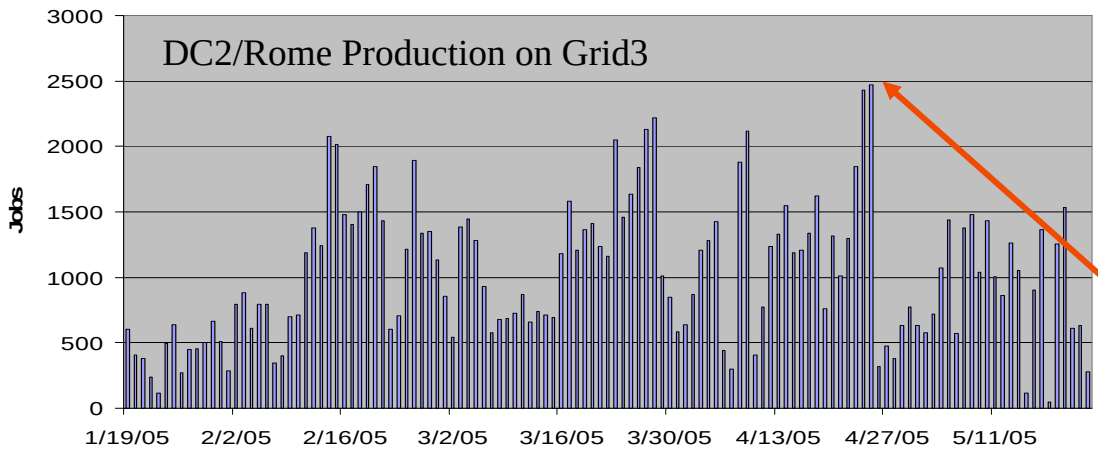
Panda Core Components

- Job Interface – allows injection of jobs into the system
- Executor Interface – translation layer for ATLAS prodsys/prodDB
- Task Buffer – keeps track of all active jobs (job state is kept in MySQL)
- Brokerage – initiates subscriptions for a block of input files required by jobs (preferentially choose sites where data is already available)
- Dispatcher – sends actual job payload to a site, on demand, if all conditions (input data, space and other requirements are met)
- Data Service – interface to DQ2 Data Management system
- Job Scheduler – send pilot jobs to remote sites
- Pilot Jobs – lightweight execution environment to prepare CE, request actual payload, execute payload, and clean up
- Logging and Monitoring systems – http and web based
- All communications through REST style HTTPS services (via mod_python and Apache servers)


Apache
mod_ssl
env vars
mod_gridsite
Pilot
HTTPS
mod_python
JobDispatcher
TaskBuffer
UserIF
PandaDB
DataService
mod_python
ExtIF
mod_python
Brokerage
DDMS

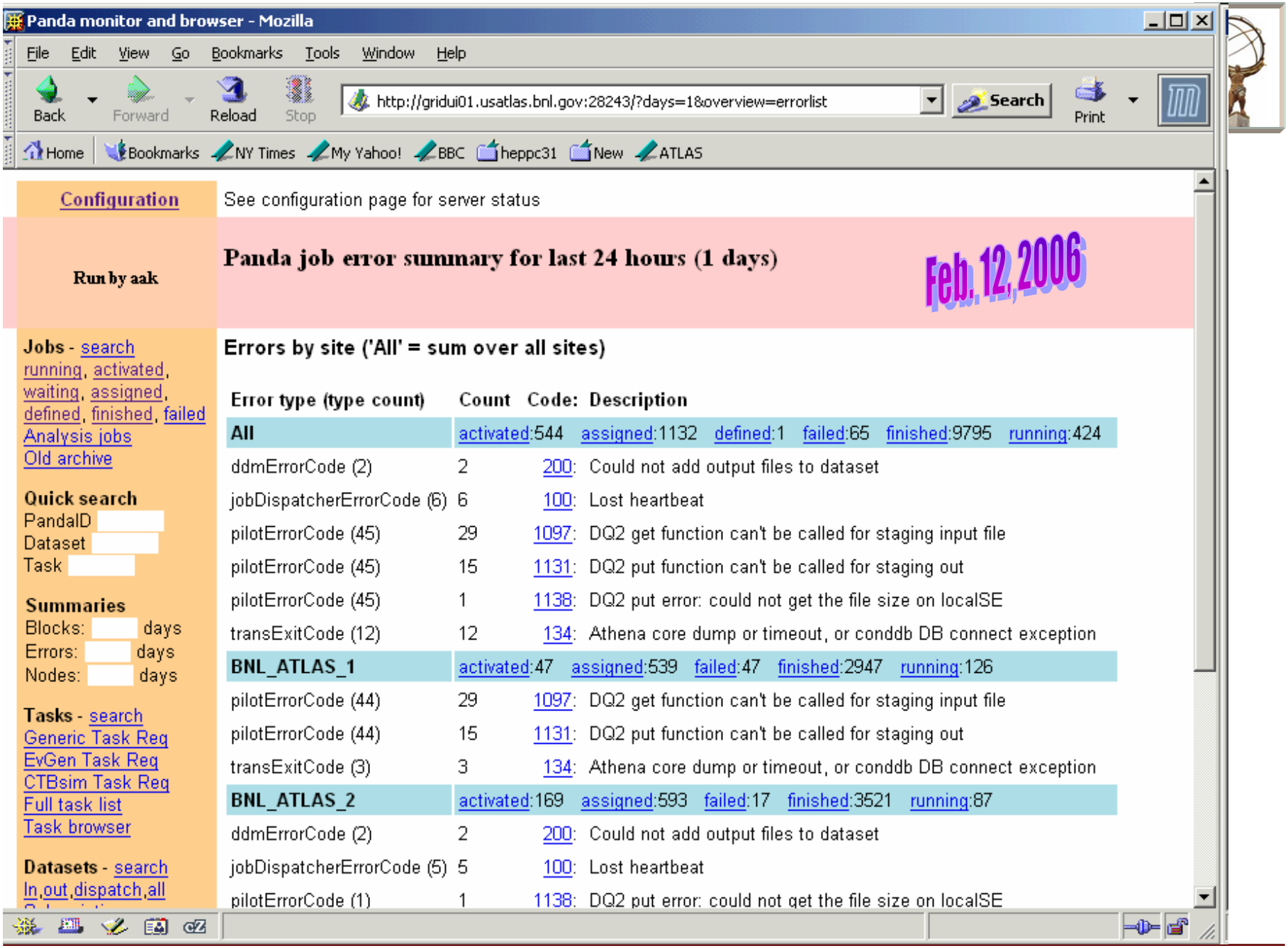
Panda in CSC Pre-production





- Panda – no scaling limits seen so far (target is factor of 10-20 higher)
- Update: 11k jobs finished Feb. 12th, 2006
- 30% of 113k ATLAS jobs (CSC) done by Panda
- Most efficient executor – lowest failure rate (target of <10% Panda failures already achieved)
- Fewer shifters required compared to DC2

Scaling issues seen during Rome Production on Grid3


Panda monitor and browser - Mozilla
File Edit View Go Bookmarks Tools Window Help
Back Forward Reload Stop
http://gridui01.usatlas.bnl.gov:28243/?days=1&overview=errorlist
Search Print
Home Bookmarks NY Times My Yahoo! BBC heppc31 New ATLAS
Configuration
See configuration page for server status
Run by aak
Panda job error summary for last 24 hours (1 days)
Feb. 12, 2006
Jobs - search
running, activated,
waiting, assigned,
defined, finished, failed
Analysis jobs
Old archive
Quick search
PandaID
Dataset
Task
Summaries
Blocks: days
Errors: days
Nodes: days
Tasks - search
Generic Task Req
EvGen Task Req
CTBsim Task Req
Full task list
Task browser
Datasets - search
In,out,dispatch,all
Errors by site ('All' = sum over all sites)
Error type (type count) Count Code: Description
All activated:544 assigned:1132 defined:1 failed:65 finished:9795 running:424
ddmErrorCode (2) 2 200: Could not add output files to dataset
jobDispatcherErrorCode (6) 6 100: Lost heartbeat
pilotErrorCode (45) 29 1097: DQ2 get function can't be called for staging input file
pilotErrorCode (45) 15 1131: DQ2 put function can't be called for staging out
pilotErrorCode (45) 1 1138: DQ2 put error: could not get the file size on localSE
transExitCode (12) 12 134: Athena core dump or timeout, or conddb DB connect exception
BNL_ATLAS_1 activated:47 assigned:539 failed:47 finished:2947 running:126
pilotErrorCode (44) 29 1097: DQ2 get function can't be called for staging input file
pilotErrorCode (44) 15 1131: DQ2 put function can't be called for staging out
transExitCode (3) 3 134: Athena core dump or timeout, or conddb DB connect exception
BNL_ATLAS_2 activated:169 assigned:593 failed:17 finished:3521 running:87
ddmErrorCode (2) 2 200: Could not add output files to dataset
jobDispatcherErrorCode (5) 5 100: Lost heartbeat
pilotErrorCode (1) 1 1138: DQ2 put error: could not get the file size on localSE

Panda Efficiency

- Automated error analysis is critical for scalability
  - Currently 5k jobs/day – if 2% unknown failures, requires debugging of 100 log files/day, which is the most time consuming part of operations
  - Scaling to 100k+ jobs per day will be a challenge
- Panda emphasized logging, monitoring, and error reporting from the start
  - Independent https based logging service
  - Integrated web based monitoring view
  - Panda reports 30+ different errors (and growing with experience)
- Many lessons from pre-production experience in January
  - Largest source of errors - if Panda cannot verify Athena/job completion (job definition errors, transformation errors, core software, test jobs …)
  - Discussion underway in ATLAS ProdSys group for better classifications
- Panda system errors – so far <<10% (majority of errors are site problems, or from other software systems)

Panda Status Today

- In full scale CSC production mode
  - But need to deploy more OSG sites (only ATLAS T1/T2's now)
  - Plan to test some LCG and NorduGrid sites
  - Learning operation of new system – debugging, fine-tuning…
- Many Panda components have alternate implementations for robustness
  - Pilot jobs can be submitted through CondorG or locally
  - Multiple job submission tools – official task request mechanism (processed through Eowyn/ExtIF), commandline (jobIF), pathena (user modified Athena jobs), Dial chained jobs (root, commandline), Ganga UI (soon)
- Web-based monitor in place with many views into system operations
  - Mainly production operations oriented to date; expanding into end-user oriented views, including production tracking and data discovery

Panda Program of Work

- Two main fronts:
  - Production - exercise Panda in production at increasing scales, and debug/refine/harden based on performance & feedback
  - Analysis - finish delivering an effective analysis capability in Panda
- The program, in outline:
  - Feb-Apr: evaluate how to scale Panda up for full analysis workloads, while meeting CSC production targets
  - May+: Validate Panda as a hardened production system with >90% overall efficiency and no scaling limits for production
  - Apr-Aug: implement Panda scaling for analysis
  - Fall: Integrate DQ2 support for personal datasets, data management and validate Panda for scalable analysis
  - Meet SC4 objectives – as discussed at SC4 workshop here last week: goals of increasing scale throughout 2nd half of 2006

Panda Contributors

- Project Cordinators: Torre Wenaus – BNL, Kaushik De – UTA
- Lead Developer: Tadashi Maeno – BNL
- Panda team
  - Brookhaven National Laboratory (BNL): Wensheng Deng, Alexei Klimentov, Pavel Nevski, Yuri Smirnov, Tomasz Wlodek, Xin Zhao; University of Texas at Arlington (UTA): Nurcan Ozturk, Mark Sosebee; Oklahoma University (OU): Karthik Arunachalam, Horst Severini; University of Chicago (UC): Marco Mambelli; Argonne National Laboratory (ANL): Jerry Gerialtowski; Lawrence Berkeley Lab (LBL): Martin Woudstra
  - Distributed Analysis team (from Dial): David Adams – BNL, Hyunwoo Kim – UTA

More Information

- Panda
  - https://uimon.cern.ch/twiki/bin/view/Atlas/PanDA
- Panda monitor
  - http://gridui01.usatlas.bnl.gov:28243/
- DDM
  - https://uimon.cern.ch/twiki/bin/view/Atlas/DistributedDataManagement
- Access to Panda data (with DQ2)
  - https://uimon.cern.ch/twiki/bin/view/Atlas/AccessPandaData
- Distributed analysis with Panda
  - https://uimon.cern.ch/twiki/bin/view/Atlas/DAonPanda
- DIAL
  - http://www.usatlas.bnl.gov/~dladams/dial/