The ATLAS PanDA Pilot in Operation

The ATLAS PanDA Pilot in Operation

P. Nilsson [1], J. Caballero [2], K. De [1], T. Maeno [2], A. Stradling [1], T. Wenaus [2] for the ATLAS Collaboration
[1] University of Texas at Arlington, Science Hall, P O Box 19059, Arlington, TX 76019, United States
[2] Brookhaven National Laboratory, Upton, NY 11973, United States
E-mail: Paul.Nilsson@cern.ch

Abstract. The **P**roduction **an**d **D**istributed **A**nalysis system (PanDA) [1-2] was designed to meet ATLAS [3] requirements for a data-driven workload management system capable of operating at LHC data processing scale. Submitted jobs are executed on worker nodes by pilot jobs sent to the grid sites by pilot factories. This paper provides an overview of the PanDA pilot [4] system and presents major features added in light of recent operational experience, including multi-job processing, advanced job recovery for jobs with output storage failures, gLExec [5-6] based identity switching from the generic pilot to the actual user, and other security measures. The PanDA system serves all ATLAS distributed processing and is the primary system for distributed analysis; it is currently used at over 100 sites world-wide. We analyze the performance of the pilot system in processing real LHC data on the OSG [7], EGI [8] and Nordugrid [9-10] infrastructures used by ATLAS, and describe plans for its evolution.

1. Introduction

Data analysis using grid resources was one of the fundamental challenges to address before the start of data taking at the Large Hadron Collider (LHC) [11]. The ATLAS experiment alone will produce petabytes of data per year that needs to be distributed to sites worldwide [12]. More than a thousand users need to run their analyses on this data. The analysis tools, grid infrastructure and its middleware have been mature since several years and the users are now benefitting from a well tested and established system with the start of LHC operations.

The PanDA system is a workload management system developed to meet the needs of ATLAS distributed computing. The pilot-based system has proven to be very successful in managing ATLAS distributed production requirements, and has been extended to manage distributed analysis on all three grids used by the experiment.

The following sections present the PanDA pilot system, its workflow and several of its advanced features, as well as performance in processing real LHC data on the OSG, EGI and Nordugrid infrastructures used by the ATLAS experiment.

2. The PanDA pilot workflow

Jobs are submitted to the PanDA server manually by users and automatically by the production system. Pilot factories send jobs containing a thin wrapper to the grid computing sites through Condor-G [13]. The local batch system sends the wrapper to a worker node (WN) where it downloads and executes the PanDA pilot code. The pilot begins with verifying the WN environment, local disk space, available memory, etc, and then proceeds with asking the PanDA job dispatcher for a job suitable to the WN and the site. After downloading a matching job, it attempts to setup the runtime environment before staging in any input files from the local Storage Element (SE). The user or production job is then executed and monitored for progress. When the job has finished successfully, the output files are staged out to the SE, and the final job status is communicated to the PanDA server. Before exiting, it cleans up after itself. The PanDA architecture and job flow are shown in Figure 1 while the pilot execution details are shown in Figure 2.

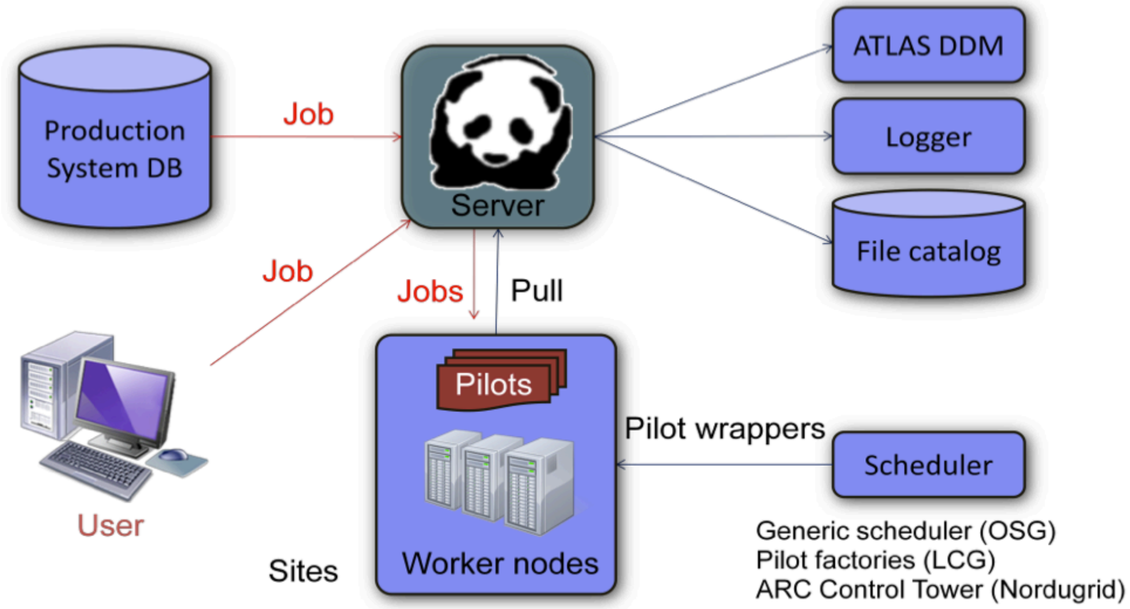

Figure 1. PanDA architecture and job flow. Users and the production system submit jobs to PanDA. Pilots are asynchronously asking the job dispatcher for suitable jobs.

The pilot gets configuration information (*queuedata*) about the site it is running on from the PanDA Schedconfig DB. The queuedata contains all the information that the pilot needs to know about the site, such as software directory, setup paths, which special features to use, which copy tools to use for stage-in/out, etc. The pilot supports 21 different copy tools for use on different storage systems. In case the storage system allows for remote access of the input files (e.g. xrootd), the pilot will skip the stage-in and leave the file access to the payload.

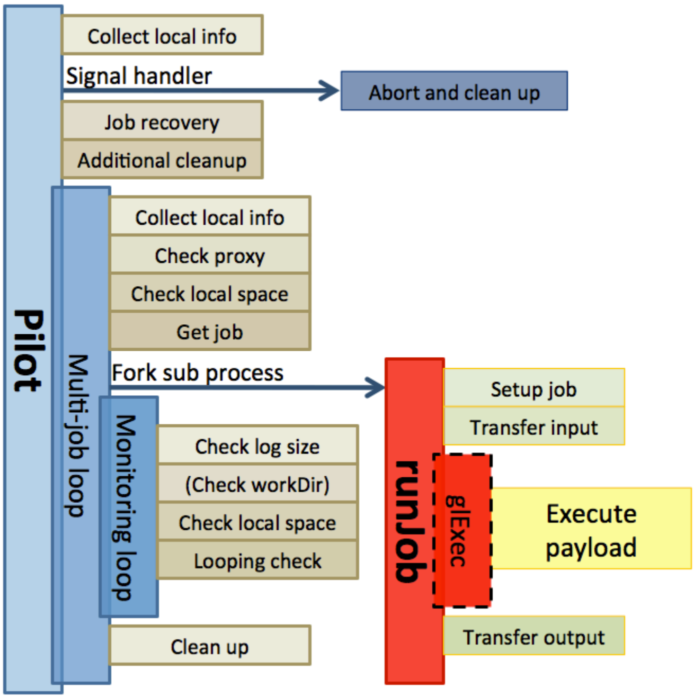

Figure 2. Pilot execution. The job payload is run by the runJob process, which in turn is a forked process monitored by the pilot.

3. Special pilot features

The PanDA pilot is equipped with a number of special features to improve success rate, job processing, security and error handling. These include the following:

- **Job and disk space monitoring**. The work directory of a user analysis job has to stay within a certain size (currently 7 GB) and will be aborted by the pilot if it breaks the limit. The sizes of output and payload log files (stdout) are also monitored, although currently there is only a size limit set for the log files (2 GB). In case the job hangs, the pilot will notice that it has stopped writing to its output files (known as a *looping job*). After a configurable time limit (per job), the pilot will abort the job. Furthermore, the pilot keeps track of the remaining disk space to ensure that there is enough space to finish the job.
- **Job recovery**. Occasionally sites experience SE issues. If a job has run for many hours and only fails during stage-out, we do not necessarily abandon the entire job and waste all the CPU time used. If a pilot fails to upload the output files for an otherwise completed job, it can optionally leave the output on the local disk for a later pilot to re-attempt the transfer. The feature is used on some production sites, but not on user analysis sites since the job turn-around should be quick.
- **Multi-job processing**. A single pilot can run several short jobs sequentially until it runs out of time (pre-determined and varies between different sites). The feature is particularly useful for reducing the pilot rate when there are many short jobs in the system. It is used both with production and user analysis jobs.
- **General security measures**. All communications with the PanDA server use certificates. During job download, a pilot can be considered legitimate if it presents a

special time limited token to the job dispatcher that was previously generated by the server, thus only allowing legitimate jobs already known to the server**.**

- **gLExec based security**. To prevent a user job from attempting to use the pilot credentials (which has many privileges), gLExec can switch the identity to the user prior to job execution. The user job is thus executed with the users' normal limited credentials. The integration of gLExec with the pilot is complete and is currently being tested [14].

4. Performance

PanDA is serving 40-50k+ concurrent production jobs world wide. At the same time it is also serving user analysis jobs. These jobs, which are chaotic by nature, have recently reached peaks of 27k concurrent running jobs.

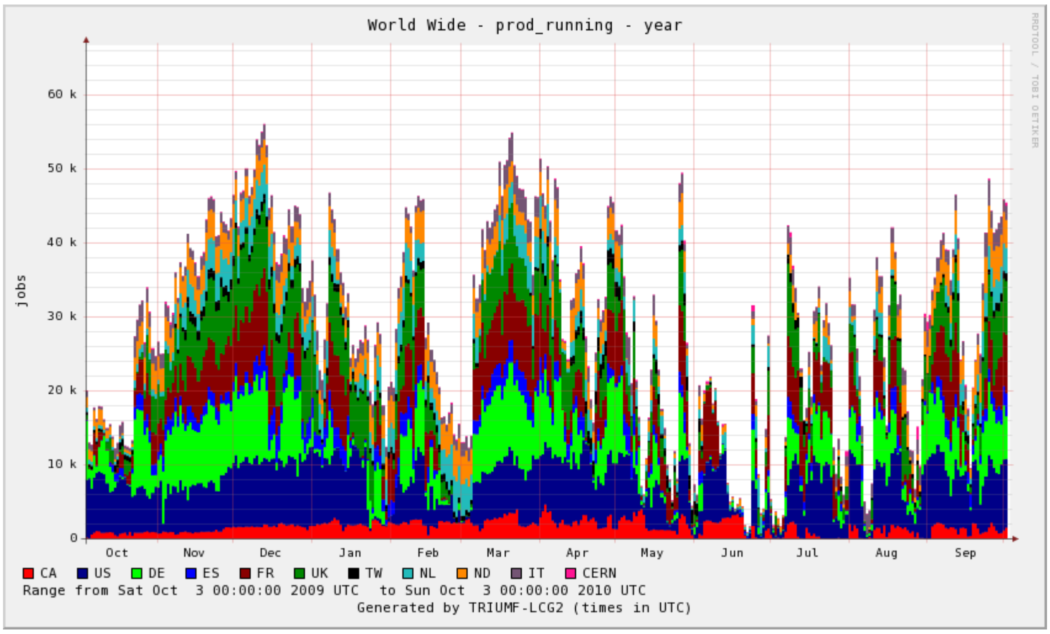

Figure 3. Concurrent production jobs in the system in 2009-10.

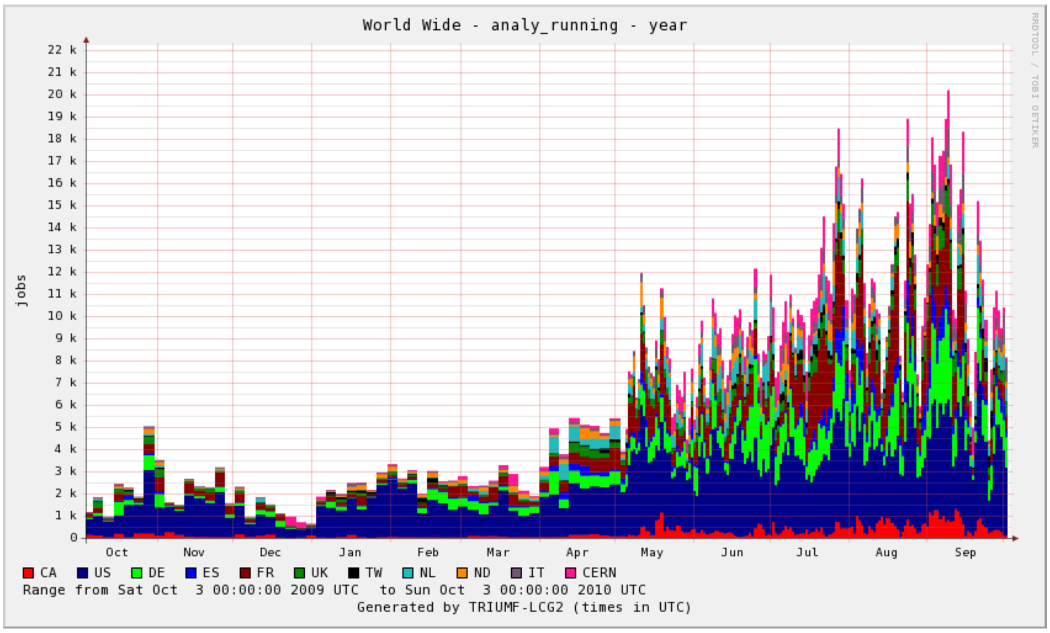

Figure 4. Concurrent user analysis jobs in the system in 2009-10. The sharp rise of activity beginning at the end of March marks the arrival of LHC data.

The error rate in the entire system is at the level of 10 percent. The majority of these errors are site or system related, while the rest are problems with ATLAS software. The PanDA pilot can currently identify about 100 different error types that are reported back to the server as they happen. The web based PanDA monitor [15] can be used to find individual jobs and their log files (entire workarea), which is highly useful, especially for debugging problematic jobs.

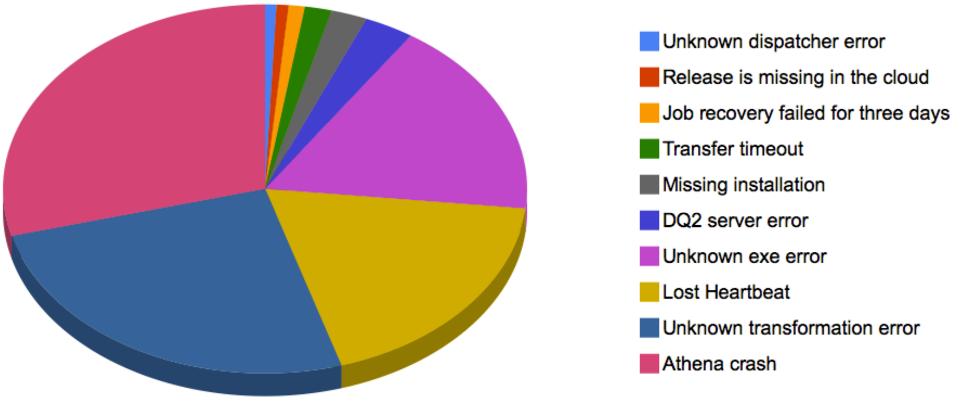

Figure 5. Reported top ten errors during September 2010.

5. Evolution of the pilot

Since the beginning of the PanDA project in 2005, a steady stream of new features and modifications has been added to the PanDA pilot. Partial refactoring of the code has occasionally been necessary, e.g. to allow easy integration of gLExec, as well as an ongoing cleanup and general improvement of older code. Overall the PanDA pilot has a robust and stable object oriented structure which will continue to smoothly allow for future improvements and add-ons. Recent or upcoming new features include the following:

- **Plug-ins**. The pilots' modular design makes it easy to add new features. E.g. a plug-in for WN resource monitoring using the STOMP framework [16] is currently being developed by people outside the PanDA team.

- **CernVM integration**. This project aims to use CernVM [17] package to run ATLAS jobs on cloud computing resources. The work has been done in collaboration with the CernVM team [18].

- **Next generation job recovery**. The job recovery algorithm is currently being rewritten from scratch for greater efficiency.

- **Extension of the pilot release candidate testing framework**. When a new pilot version is ready for release it is first tested with special test jobs on all sites using the pilot release candidate framework. In this framework the pilot wrappers send the final development version of the pilot which downloads and runs special test jobs. Work has recently been done to automatize the testing of a new pilot version using special analysis jobs with the HammerCloud system [19].

- **Retry of stage-in using alternative replica**. Occasionally the primary replica choice for a given job is unavailable due to various reasons. In this case the pilot will attempt to stage-in alternative replicas that are available at the same SE.

6. Conclusions

The PanDA pilot has been in operation in the ATLAS experiment since 2005. It is equipped with several advanced features, including job monitoring, recovery and multi-job processing. The code is continuously being adapted to changing conditions in the runtime and grid site environments. New features and improvements are added frequently to ensure efficient production and to aid and benefit the large group of analysis users.

7. References

[1] *Overview of ATLAS PanDA Workload Management*. **T. Maeno et al.**, These proceedings. Taipei : Journal of Physics: Conference Series (JPCS), 2011

[2] *The PanDA System in the ATLAS Experiment* . **P. Nilsson et al.**, PoS(ACAT08)027 , 2008.

[3] *ATLAS Technical Proposal*. **ATLAS Collaboration.** CERN/LHCC/94-43, 1994.

[4] *Experience from a Pilot based system for ATLAS*. **P. Nilsson et al.**, J.Phys.Conf.Ser.119:062038, 2008.

[5] *gLExec*. https://www.nikhef.nl/pub/projects/grid/gridwiki/index.php/GLExec.

[6] *gLExec: gluing grid computing to the Unix world*. **D. Groep, O. Koeroo, G. Venekamp.**, J.Phys.:Conf.Series 119 (2008) 062032.

[7] *Open Science Grid*. http://www.opensciencegrid.org.

[8] *European Grid Initiative*. http://www.egi.eu.

[9] *The Nordugrid Project*. **M. Ellert et al.**, NIM A, 2003, Vol. 502.

[10] *Nordugrid*. http://www.nordugrid.org.

[11] *The LHC Conceptual Design Report*. **LHC Study Group**. CERN/AC/95-05, 1995.

[12] *Managing ATLAS data on a petabyte-scale with DQ2*. **M. Lassnig et al.**, J.Phys.Conf.Ser.119 (2008) 062017.

[13] *The Condor Project*. http://www.cs.wisc.edu/condor.

[14] *Improving Security in the ATLAS PanDA System*. **J. Caballero et al.**, These proceedings. Taipei : Journal of Physics: Conference Series (JPCS), 2011.

[15] *PanDA monitor*. http://panda.cern.ch.

[16] *STOMP*. http://stomp.codehaus.org.

[17] *CernVM*. http://cernvm.cern.ch/cernvm.

[18] *CernVM CoPilot: a Framework for Orchestrating Virtual Machines Running Applications of LHC Experiments on the Cloud*. **Artem Harutyunyan et al.**, These proceedings. Taipei : Journal of Physics: Conference Series (JPCS), 2011.

[19] *HammerCloud overview page*. http://gangarobot.cern.ch/hc