Distributed Data Analysis in ATLAS

Paul Nilsson for the ATLAS Collaboration

University of Texas at Arlington, Science Hall, P O Box 19059, Arlington, TX 76019, United States

Abstract. Data analysis using grid resources is one of the fundamental challenges to be addressed before the start of LHC data taking. The ATLAS detector will produce petabytes of data per year, and roughly one thousand users will need to run physics analyses on this data. Appropriate user interfaces and helper applications have been made available to ensure that the grid resources can be used without requiring expertise in grid technology. These tools enlarge the number of grid users from a few production administrators to potentially all participating physicists. ATLAS makes use of three grid infrastructures for the distributed analysis: the EGEE sites, the Open Science Grid, and NorduGrid. These grids are managed by the gLite workload management system, the PanDA workload management system, and ARC middleware; many sites can be accessed via both the gLite WMS and PanDA. Users can choose between two front-end tools to access the distributed resources. Ganga is a tool co-developed with LHCb to provide a common interface to the multitude of execution backends (local, batch, and grid). The PanDA workload management system provides a set of utilities called PanDA Client; with these tools users can easily submit Athena analysis jobs to the PanDA-managed resources. Distributed data is managed by Don Quixote 2, a system developed by ATLAS; DQ2 is used to replicate datasets according to the data distribution policies and maintains a central catalog of file locations. The operation of the grid resources is continually monitored by the GangaRobot functional testing system, and infrequent site stress tests are performed using the HammerCloud system. In addition, the DAST shift team is a group of power users who take shifts to provide distributed analysis user support; this team has effectively relieved the burden of support from the developers.

Keywords: Distributed data analysis, ATLAS, LHC, CERN, OSG, PanDA, EGEE, NorduGrid, ARC, gLite, pAthena, Ganga, GangaRobot, HammerCloud, Don Quixote 2, Condor, glExec.
PACS: 89.20.Ff

INTRODUCTION

The ATLAS experiment (1) places challenging requirements on data analysis. The detector will produce several petabytes of data per year that need to be processed and distributed around the world before the users can run their physics analyses on it. The ATLAS distributed analysis model, which in turn is based on the ATLAS computing model (2), prescribes that data is to be transferred from the top Tier 0 to the Tier 1 and Tier 2 facilities by the ATLAS Distributed Data Management system, Don Quixote 2 (DQ2) (3). Data should always be available, and file management, distribution and archiving should be automated throughout the whole grid using a central catalogue, File Transfer Service (FTS) and LCG (4) File Catalogues (LFC) (5). Since user analysis jobs can potentially use very large input datasets (100 GB up to several TB), user jobs are sent to where the data is rather than sending the data to where the job will run. Furthermore, the model requires that the job results must be made available to the user, with partial results available already during processing.

DISTRIBUTED ANALYSIS IN ATLAS

ATLAS has a heterogeneous grid infrastructure which is based on three worldwide LCG Computing Grids (WLCG): the Open Science Grid (OSG) (6) in the US, NorduGrid (7) mostly in Scandinavia and the Enabling Grids for E-sciencE (EGEE) (8) in the rest of the world. The infrastructure is a tiered system adapted to optimize resource usage. The Tier 0, located at CERN, performs the primary event processing and has a mass storage facility. Ten facilities worldwide are hosting Tier 1 nodes which archive the data from Tier 0, have reprocessing capacities, and provide access to reprocessed data and scheduled analysis by the physics groups. More than 100 worldwide sites act

as Tier 2 nodes which in turn have capacity for Monte Carlo simulation, chaotic analysis and calibration work. Local resources can be used as Tier 3 nodes to supplement the Tier 2 nodes for work off the grid.

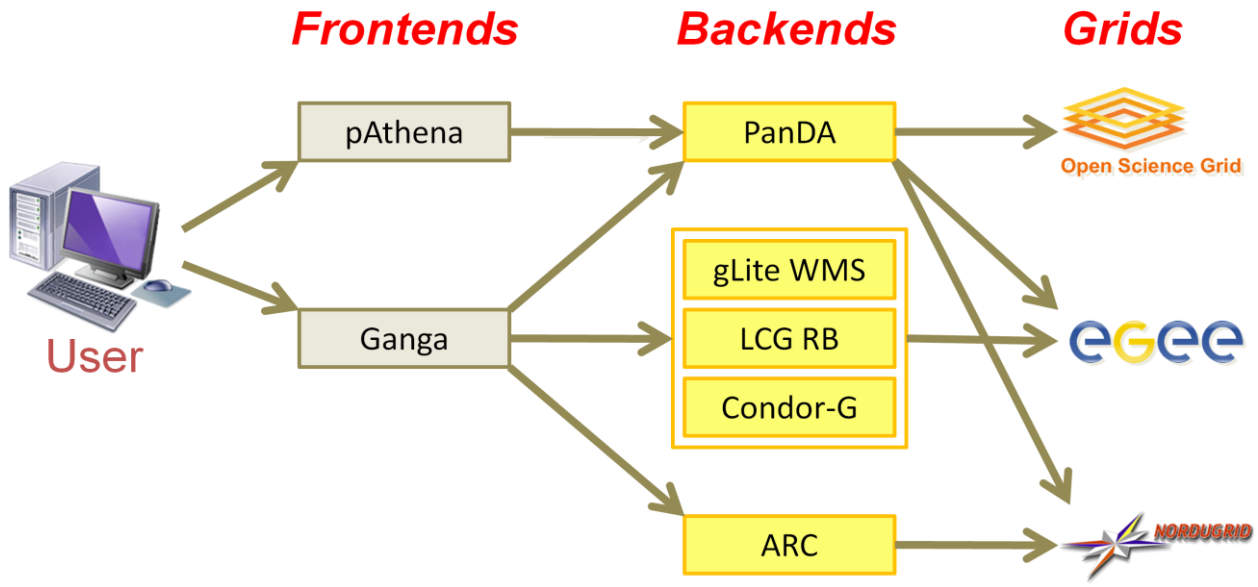

FIGURE 1. Distributed Analysis in ATLAS. A user can submit jobs to the grid by using any of the two frontend clients.

The ATLAS Production and Distributed Analysis system, PanDA (9; 10), works on all WLCG middleware and resources. PanDA was designed to meet the ATLAS requirements for a data-driven workload management system capable of operating at the LHC (11) data processing scale. It has a single task queue and uses pilots (12) for the job execution. The pilots are submitted to the batch queues through Condor-G (13) by the PanDA autopilot system on OSG, pilot factories on EGEE and by the ARC Control Tower on NorduGrid (14). The pilots retrieve the actual jobs from an Apache based central server as soon as CPUs are available, leading to low latencies that is especially useful for user analysis which has a chaotic behavior.

The users can submit jobs to the WLCG grids with two complementing frontend clients, the PanDA client (15) and Ganga (16), described in more detail in the next section (Figure 1).

Analysis jobs run under a production proxy unless glExec (17) is employed in identity switching mode. glExec is integrated and under test for switching job identity from the generic pilot to the actual user.

FRONTEND CLIENTS

The basic job cycle for an ATLAS user job is shown in Figure 2. The frontend client starts by packing the source code in the user working directory. The source is sent to a site holding the input data, followed by the creation of a build and run job. The build job compiles the source code on the site and creates a library file. The library file is then used as input for the run job(s), which in turn runs the actual job(s). If the input dataset is large, the run job will automatically be split into several sub jobs (user controllable). When a run job has finished, the output files are transferred to the site storage element where they can be picked up by the user. The ATLAS frontend clients have a large set of options that allow the users to modify many of the job parameters.

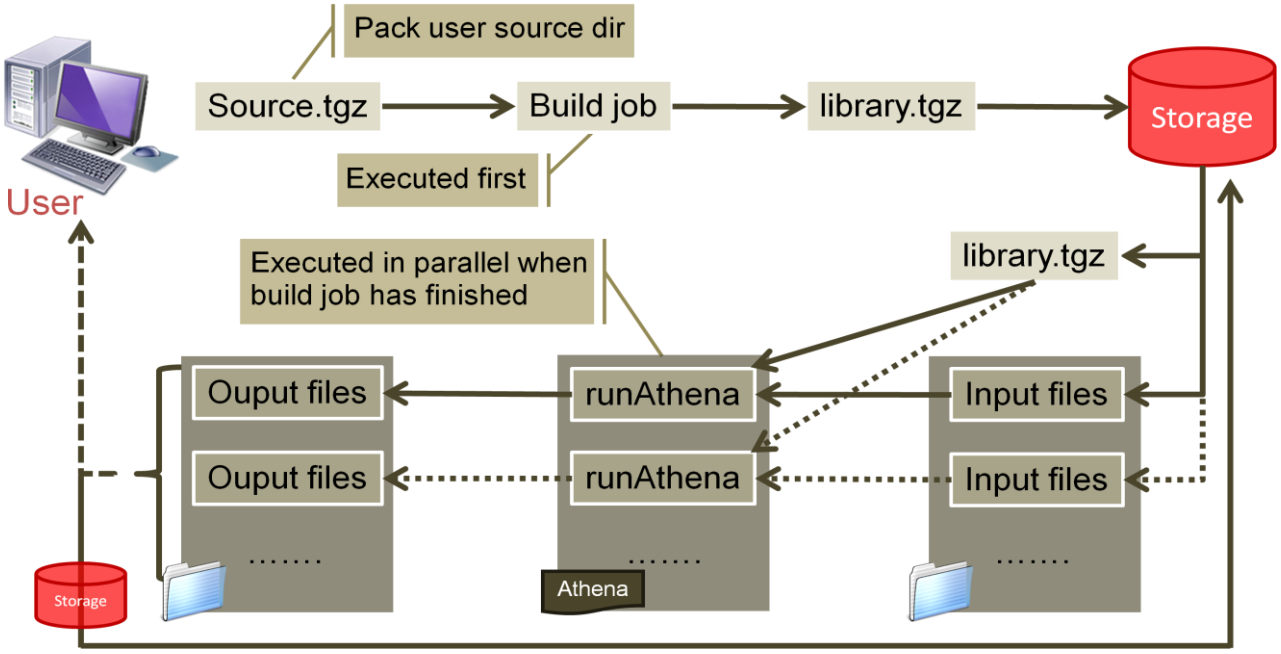

FIGURE 2. Job cycle for an ATLAS user analysis job.

PanDA Client

The PanDA client consists of several tools for job execution on the grid and for bookkeeping; ***pathena*** is the native PanDA tool for job submission to OSG resources, and can also be used to send user jobs to NorduGrid as well as to some EGEE resources; ***prun*** can be used to submit general jobs, such as ROOT and Python scripts; ***psequencer*** allows for a sequence of different tasks to be submitted (e.g. an analysis job followed by the transfer of the output back to the local machine); ***pbook*** is a bookkeeping tool for PanDA analysis jobs; and finally ***puserinfo*** can be used to control access for PanDA analysis jobs. Although any user can submit jobs to any site by default, each site can allow only a group of users to access all CPUs or a portion of the CPUs. If users want to use restricted CPUs, they first need to request access rights using the puserinfo command.

pathena is the main client tool for submitting user defined jobs from the command line to PanDA. It works on the Athena (the ATLAS offline software framework) (18) runtime environment for which it has a consistent user interface.

Users can follow the progress of their jobs by using the PanDA web based monitor (19), which among other things provides fast access to user jobs, output and log files. A particularly useful feature, especially for debugging problematic jobs, is that the full log for both the user job itself and the pilot running the job is available online. The monitor has multiple service instances for load balancing and reliable service.

Ganga

Ganga is a job management tool for local, batch systems and the grid. It was developed to meet the needs of the ATLAS and LHCb (20) experiments for a grid user interface, and has built in support for running and configuring applications based the Athena/Gaudi (21) framework that is common to these experiments. It allows for trivial switching between local testing and large-scale processing on the grid, and supports job submission from both the command line, graphical user interface, and via IPython (22) scripting. Ganga maps an arbitrary “application” to an arbitrary “backend” and can thus be used to send user jobs to any of the OSG, NorduGrid and EGEE grids.

GangaRobot and HammerCloud

Ganga is not only useful for end users, but also has a Python API that allows for grid application development. Two such examples come from the ATLAS Distributed Analysis Testing tools. The ***GangaRobot*** (23; 24) is used to run many short functional tests daily to continuously validate the DA workflows. Results from these tests are fed back into Ganga so that broken sites can be avoided. ***HammerCloud*** (25) is used to run large site stress tests to measure the behavior of the storage, networks, databases, etc, under heavy load. A recent study, STEP'09 (a coordinated effort of all four LHC experiments for running as much activity as possible simultaneously) (26), identified several site issues that are currently being improved upon.

DISTRIBUTED ANALYSIS USER SUPPORT

The ATLAS users get grid support from three places; Online documentation (starting with the ATLAS Computing Workbook (27)), Offline Software Tutorials (28) held at CERN every 6-8 weeks, and from an eGroups forum for Distributed Analysis Help. The latter is sustained by support from the Distributed Analysis Support Team (DAST) as well as from user-to-user. The DAST has been in operation since 2008 and currently has a team of ten shifters. Having expert knowledge on call relieves the developers of the user support burden. User problems are solved quickly, and organized distributed analysis shifter trainings help disseminate expert knowledge which was previously held only by developers.

REFERENCES

1. **ATLAS Collaboration.** *ATLAS Technical Proposal.* s.l. : CERN/LHCC/94-43, 1994.
2. **ATLAS Computing Group.** *Computing Technical Design Report.* s.l. : CERN-LHCC-2005-022, 2005.
3. *Managing ATLAS data on a petabyte-scale with DQ2.* **M. Lassnig et al.** s.l. : J.Phys.Conf.Ser.119 (2008) 062017.
4. *The LHC computing grid project.* **M. Lamanna et al.** s.l. : NIM A, 2004, Vol. 534.
5. *LCG File Catalog administrators' guide* . [Online] https://twiki.cern.ch/twiki/bin/view/LCG/LfcAdminGuide.
6. *Open Science Grid.* [Online] http://www.opensciencegrid.org.
7. *The NorduGrid Project.* **M. Ellert et al.** s.l. : NIM A, 2003, Vol. 502.
8. *Enabling Grids for E-sciencE.* [Online] http://www.eu-egee.org.
9. *The PanDA System in the ATLAS Experiment* . **P. Nilsson et al.** s.l. : PoS(ACAT08)027 , 2008.
10. *PanDA: Distributed production and distributed analysis system for ATLAS.* **T. Maeno et al.** s.l. : J.Phys.Conf.Ser.119:062036, 2008.
11. **LHC Study Group.** *The LHC Conceptual Design Report.* s.l. : CERN/AC/95-05, 1995.
12. *Experience from a Pilot based system for ATLAS.* **P. Nilsson et al.** s.l. : J.Phys.Conf.Ser.119:062038, 2008.
13. *The Condor Project.* [Online] http://www.cs.wisc.edu/condor.
14. *Advanced Resource Connector middleware for lightweight computational Grids.* **M. Ellert, et al.** 23, s.l. : Future Generation Computer Systems, 2007.
15. *PanDA client.* [Online] https://twiki.cern.ch/twiki/bin/view/Atlas/PandaTools.
16. *Ganga: A tool for computational-task management and easy access to Grid resources.* **J.T. Mościcki et al.** 11, s.l. : Computer Physics Communications, 2009, Vol. 180.
17. *gLExec: gluing grid computing to the Unix world.* **D. Groep, O. Koeroo, G. Venekamp.** s.l. : J.Phys.:Conf.Series 119 (2008) 062032.
18. *Athena Core Software.* [Online] http://atlas-computing.web.cern.ch/atlas-computing/packages/athenaCore/athenaCore.php.
19. *PanDA monitor.* [Online] http://panda.cern.ch.
20. **LHCb Collaboration.** *LHCb - Technical Proposal.* s.l. : CERN/LHCC98-4, 1998.
21. *GAUDI - A software architecture and framework for building LHCb data processing applications.* **G. Barrand et al.** Padova : CHEP 2000 proceedings, 2000.
22. *IPython.* [Online] http://ipython.scipy.org/moin.
23. *GangaRobot overview page.* [Online] http://gangarobot.cern.ch.
24. *Functional and Large-Scale Testing of the ATLAS Distributed Analysis Facilities with Ganga.* **D. van der Ster et al.** s.l. : CHEP 2009 proceedings, 2009.
25. *HammerCloud overview page.* [Online] http://gangarobot.cern.ch/hc.
26. **ATLAS Collaboration.** *STEP'09.* [Online] https://twiki.cern.ch/twiki/bin/view/Atlas/Step09.
27. *The ATLAS Computing Workbook.* [Online] CERN/ATLAS. https://twiki.cern.ch/twiki/bin/view/Atlas/WorkBook.
28. *ATLAS Regular Computing Tutorial.* [Online] https://twiki.cern.ch/twiki/bin/view/Atlas/RegularComputingTutorial.