ATLAS tier-3 data processing with GridPilot, NorduGrid and WLCG

Frederik Orellana, Morten Badensø, Jacob Debel, Simon Heisterkamp, Ask Emil Jensen

Niels Bohr Institute, University of Copenhagen

***Abstract*— We present a novel tool for users to manage data processing on grid infrastructures. The tool provides a graphical user interface that allows getting new people quickly up an running by using a library of applications built up by previous users.**

Data analysis, Data management, Computer interfaces, Computer data analysis, Computer simulation, Grid computing, Computer usability

I. Introduction

A challenge facing all new ATLAS master's and PhD students is getting access to and using grid computing facilities to access data or carry out their own large-scale data processing or simulation.

The grid facilities used by ATLAS are large in scale: the aggregated number of CPU cores used by the ATLAS production system is in the order of 42'000, the total storage available is in the order of 46 PB. These resources are distributed worldwide and used by hundreds of ATLAS users. Each of these grid facilities belongs to one or several of the grids used by ATLAS: WLCG, NorduGrid and Open Science Grid (OSG) and access thus proceeds via 3 different flavors of grid protocols: gLite [1], ARC [2] and Globus [3] respectively. On top of the grid facilities and protocols, ATLAS has built more layers of abstraction: the ATLAS production system and PanDA [4] and the DQ2 [5] dataset catalog and management system.

In order to allow users to use these resources for data processing and analysis, over the years, several client tool projects have emerged within ATLAS, notably Ganga [6] and Panda/Pathena [7]. Both of these present the user with a command-line interface[1] and handle data and job management via a mixture of the DQ2 Python API and plugins for the various grid protocols. Currently, both support OSG and WLCG, Panda/Pathena supports NorduGrid via a third-party service and Ganga supports NorduGrid via a plugin.

The e-science group at the Niels Bohr Institute (NBI) has for some time worked with the local ATLAS group on:

1) Contributing CPU and storage resources to ATLAS production through NDGF/NorduGrid.
2) Putting tools in place for local users to access ATLAS data from all over the world.
3) Putting tools in place for local users to use NDGF/NorduGrid resources to carry out data processing and analysis.
4) Creating a user-friendly local tier-3 analysis facility.

As the amount of data is increasing and getting more interesting, in the coming years, more and more users are expected to need 2 – 4. This makes it crucial to have automatized tools and procedures in place, such that a physicist spends his time doing physics, not computing, and such that the computing staff is not overloaded by manual support work. Notice that although the scientific content of the data analyses is expected to vary tremendously, the purely computational tasks are expected to follow a few well-known patterns: Athena simulation, Athena dataset processing and Root ntuple analysis.

Therefore we have put in place a library of applications, where a new student can simply find an application that matches his needs and only modify e.g. the name of the input dataset and/or a text file of code (e.g. Athena jobOptions or a Root macro). Currently, our library contains the following ATLAS applications:

- CSC simulation with a standard event generation transformation and a custom jobOptions file
- ESD to D3PD conversion with standard ATLAS reconstruction transformation
- ESD to ntuple conversion with standard ATLAS reconstruction transformation
- RDO to ESD conversion with a custom jobOptions file and extra Athena tags
- Boildown of ntuple files using a Root macro

Since importing and exporting applications is next to trivial, the library is expected to grow if and when students need different templates than those available.

II. Implementation

To manage user datasets and computing jobs, we use a GUI application called GridPilot. GridPilot was originally conceived for ATLAS testbeam data production [8] and developed into a general-purpose grid GUI in the context of

1 Ganga actually has a QT GUI, but it is not much used.

the NBI e-science group. It has a plugin architecture and supports various computing, file transfer and database back-ends – notably the NorduGrid and WLCG grids, the GridFTP and SRM file transfer protocols and the ATLAS DQ2 dataset/file catalog.

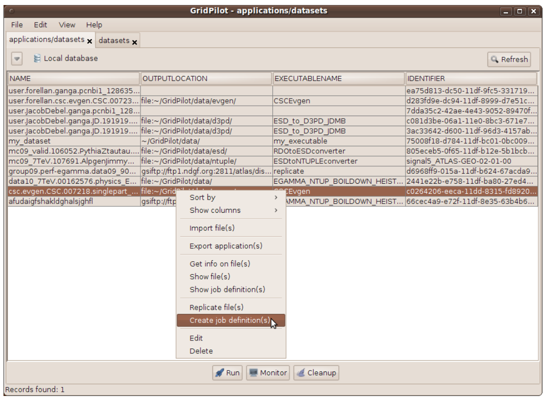

Illustration 1: Main GridPilot window with user applications/datasets.

GridPilot is implemented in Java and is available for on all popular operating systems (Windows *, Mac OS X, Linux). It has no external dependencies, meaning that direct communication with a number of web services were implemented, using available libraries from Globus, gLite and ARC – or, in the case of the ATLAS services, from scratch, without much documentation.

III. Work flow

The idea is that the user keeps his data and data productions organized in so-called datasets. To GridPilot, a dataset is simply a database record with a number of fields, notably name, input dataset, number of files, output location and executable.

The executable is another database record with a number of fields, notably name, version, executable file and runtime environment. The executable file is the script or binary that will actually be executed a number of times in order to produce this dataset. Typically it will be run with each of the files of the input dataset as input.

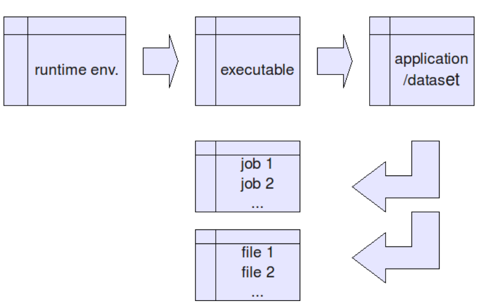

Illustration 2: Relationships of the GridPilot database records.

The runtime environment is yet another database record with a number of fields, notably name and computing system. Returning to the datasets, each dataset is the parent of a number of file records, and if the dataset was produced with GridPilot, it also is the parent of a number of job definition records. Each of these job definition records contain enough information to create a job that can run on any of the supported computing back-ends.

All the user has to worry about are the dataset and executable records. Once he has created a dataset record, job definitions can be created and run automatically. If the jobs finish successfully, output files are registered as file records in the local database. If he deems that his dataset is of interest to other ATLAS users, he can publish it to the DQ2 catalog by simply copy-pasting the record to the ATLAS datasets tab. If he deems that his work is a good example for others to use as starting point or template for their data processing, he can simply choose “Export selected application(s)” from the “File” menu. By default this will export the application to the central GridPilot “app store”, but the user can export to any directory - local or remote.

IV. Data management

GridPilot has some support for data management: Files can be downloaded from, uploaded to and replicated between grid file servers and the ATLAS DQ2 dataset catalog and LFC file catalog are specifically supported.

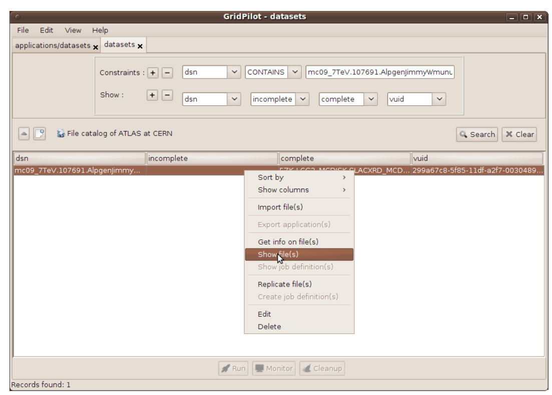

Illustration 3: Main GridPilot window with ATLAS datasets.

Notice that all this is simply meant to allow e.g. to easily to find and download a few files on which to try out an analysis before submitting a large-scale production to a grid back-end. For large-scale transfers, the PanDA web interface for replication requests or the DQ2 suite of command-line tools should be used.

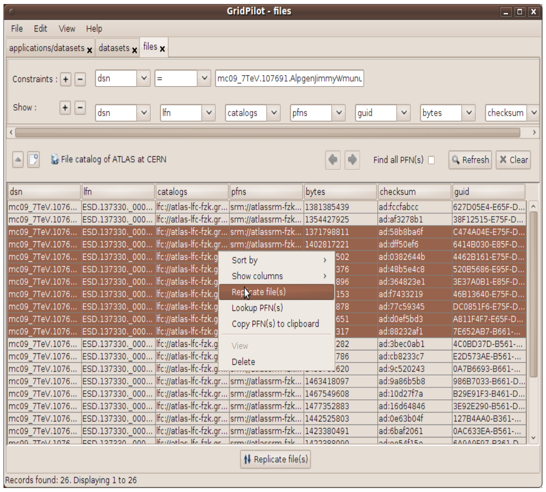

Illustration 4: Main GridPilot window with ATLAS file records.

V. Two examples

A. ESD to D3PD

This application converts ESD data files to D3PD format using a standard transformation from the ATLAS Athena software package (release 15.8.0.2), discarding information that is not relevant for the foreseen analysis. We ran the application over all 762 input files, generating 762 output files, first on a single NorduGrid (NG) cluster only, then on NorduGrid at large and finally on WLCG (GLite) at large. Each run is described below.

1. Local NorduGrid tier-3 cluster only

Total submission time:
Average submission time per job:
Average CPU time per job:
Total submission, processing and data transfer time:

2. All NorduGrid

Total submission time:
Average submission time per job:
Average CPU time per job:
Total submission, processing and data transfer time:

3. All WLCG

Total submission time:
Average submission time per job:
Average CPU time per job:
Total submission, processing and data transfer time:

B. Ntuple boildown

This application has the same purpose of filtering out information not considered relevant, but has Root ntuple data files as both input and output, reducing their size by about a factory of 10. We followed the same procedure as in the previous example, but with XXX input files and YYY output files.

1. Local NorduGrid tier-3 cluster only

Total submission time:
Average submission time per job:
Average CPU time per job:
Total submission, processing and data transfer time:

2. All NorduGrid

Total submission time:
Average submission time per job:
Average CPU time per job:
Total submission, processing and data transfer time:

3. All WLCG

Total submission time:
Average submission time per job:
Average CPU time per job:
Total submission, processing and data transfer time:

VI. Conclusion

The main motivation for the current work was a desire to get new ATLAS master's and PhD students faster on track with their real work, which is currently data analysis, and avoid having them spend too much time on the computing setup and infrastructure – in particular on the intricacies of the grid technology used by ATLAS.

For this, we have chosen to provide a working model we believe is easy and compelling and at the same time forces users to work in a systematic and reproducible manner - allowing others to reuse their work.

A related benefit is long-term traceability or data provenance: Data, produced by data simulation or processing on grid resources can be reproduced years after the final analysis of the data was done and the data probably discarded.

GridPilot is available for public download at www.gridpilot.dk.

References

[1] E. Laure et al., “Programming the Grid with gLite”, In Computational Methods in Science and Technology, pages 33–46, Scientific Publishers OWN, 2006

[2] M. Ellert et al., “Advanced Resource Connector middleware for lightweight computational Grids”, Future Generation Computer Systems (2007) 23, http://www.nordugrid.org/

[3] I. Foster and C. Kesselman, “Globus: A metacomputing infrastructure toolkit”, The International Journal of Supercomputer Applications and High Performance Computing, 11(2):115–128, 1997

[4] T.Maeno (ATLASCollaboration), "PanDA: Distributed Production and Distributed Analysis System for ATLAS", J.Phys.Conf.Ser. 119 (2008) 062036, http://iopscience.iop.org/1742-6596/119/6/062036

[5] M Branco et al (2008), "Managing ATLAS data on a petabyte-scale with DQ2", J. Phys.: Conf. Ser. 119 062017, doi: 10.1088/1742-6596/119/6/062017, http://iopscience.iop.org/1742-6596/119/6/062017

[6] J.T.Mościcki, F.Brochu, J.Ebke, U.Egede, J.Elmsheuser, K.Harrison, R.W.L.Jones, H.C.Lee, D.Liko, A.Maier, A.Muraru, G.N.Patrick, K.Pajchel, W.Reece, B.H.Samset, M.W.Slater, A.Soroko, C.L.Tan, D.C.Vanderster, M.Williams, "GANGA: A tool for computational-task management and easy access to Grid resources ", Computer Physics Communications, Volume 180, Issue 11, p. 2303-2316, http://arxiv.org/abs/arxiv:0902.2685

[7] G. Negri, D. Barberis, K. Bos, A. Klimentov and M. Lamanna, "Distributed Computing in ATLAS ", Proceedings of Science, PoS(ACAT08)035, http://pos.sissa.it/cgi-bin/reader/conf.cgi?confid=70

[8] M. Dosil, A. Farilla, M. Gallas, V. Giangiobbe and F. Orellana., "Massive data processing for the atlas combined test beam". February 2006. 4pp. Paper presented (talk) at CHEP '06 and published in IEEE Transaction on Nuclear Science, Volume 53, Issue 5, Oct. 2006 Page(s): 2887 - 2891. ISSN 0018-9499