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

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

Niels Bohr Institute, University of Copenhagen

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

Grid computing, usability, data management

I. Introduction

A challenge facing all new ATLAS diploma 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 thousands 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 (OSF) and access thus proceeds via 3 different flavors of grid protocols: gLite [1] (WLCG), ARC [2] (NorduGrid) and Globus [3] (OSF). On top of the grid facilities and protocols, ATLAS has built more layers of abstraction: the ATLAS production system and the DDM 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 and Panda/Pathena. Both of these handle data and job management via a mixture of the DDM Python API and plugins for the various grid protocols. Currently, both support OSF and WLCG, Panda supports NorduGrid via a third-party service and Ganga in principle supports NorduGrid via a plugin (that needs to be updated). Also, both of these are command-line oriented, Linux-only tools – although Ganga sports a QT GUI, which is not much used.

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

1) Contributing CPU and storage resources to ATLAS production through NDGF/NorduGrid.
2) Making ATLAS data from all over the world available to local users.
3) Making NDGF/NorduGrid resources available for local ATLAS users to carry out data processing and analysis.

As the amount of data is increasing and getting more interesting, in the coming years, more and more people are expected to need points 2 and 3. Moreover, although the scientific content of the computational work is expected to vary tremendously, the purely computational tasks are expected to follow a few well-known patterns.

Therefore is seems very desirable to 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).

This is exactly what we have done. Initially, 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

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

We use an application called GridPilot. It was originally conceived for ATLAS testbeam data production [4] and developed into a general-purpose grid GUI in the context of the NBI e-science group. GridPilot has a plugin architecture, supporting various computing, file transfer and database back-ends – notably the NorduGrid and WLCG grids, the GridFTP and SRM file transfer protocols and the ATLAS DDM dataset/file catalog.

GridPilot is implemented in Java are available for on all popular operating systems (Windows *, Mac OS X, Linux) from the GridPilot web site, www.gridpilot.dk.

From the beginning we considered it important that GridPilot should be pure Java and have no external dependencies. Achieving this meant that direct communication with a number of web services had to be 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.

The runtime environment is yet another database record with a number of fields, notably name and computing system.

To return 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.

IV. Data management

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

All this is not meant to be used for large-scale data transfers, but simply 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.

V. Two examples

A. ESD to D3PD

B. Ntuple boildown

VI. conclusion and outlook

Reproducibility and provenance.

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] 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.

[5] F. Orellana, C. U. Søttrup, A. Wäänänen, D. Kalici, M. Grønager, “The case for a simpler security model in grid computing”, March 2009. IEEE Proceedings of the 2009 International Conference on Availability, Reliability and Security, Fukuoka Institute of Technology, Fukuoka, Japan. ISBN 978-0-7695-3564-7.