Journal of Physics: Conference Series **331** (2011) 072037 doi:10.1088/1742-6596/331/7/072037

Distributed analysis at LHC*b*

Mike Williams, Ulrik Egede and Stuart Paterson on behalf of the LHC*b* Collaboration

Department of Physics, Imperial College London, London SW7 2AZ, UK

E-mail: michael.williams@imperial.ac.uk

Abstract. The distributed analysis experience to date at LHC*b* has been positive: job success rates are high and wait times for high-priority jobs are low. LHC*b* users access the grid using the GANGA job-management package, while the LHC*b* virtual organization manages its resources using the DIRAC package. This clear division of labor has benefitted LHC*b* and its users greatly; it is a major reason why distributed analysis at LHC*b* has been so successful. The newly formed LHC*b* distributed analysis support team has also proved to be a success.

1. Introduction

The LHC*b* experiment collects data at $\mathcal{O}(100)$ MB/s and expects to collect approximately 500 TB of raw data in 2011. In September 2010 alone, LHC*b* used over 300 CPU years of processing power (see Figure 1) and over 700 TB of disk space. Clearly, distributed computing resources are required to carry out LHC*b*'s physics program.

There is a clear division of labor regarding distributed analysis development work in LHC*b*. The GANGA package [1] is the frontend used by members of the LHC*b* virtual organization (VO) to access distributed resources. The main goal of GANGA is to ensure that users are able to efficiently access all available resources. The DIRAC package [2] is the workload management system (WMS) and data management system (DMS) for LHC*b*. The main goal of DIRAC is to ensure that the LHC*b* VO uses its resources efficiently and to enforce job prioritization. These packages have worked well together to provide a positive distributed analysis experience for LHC*b*.

2. GANGA

The GANGA package was initially developed to meet the needs of the ATLAS and LHC*b* collaborations for a grid user interface; however, it is now used by many other groups as well. In September 2010, the usage was approximately 45% ATLAS, 45% LHC*b* and 10% other groups. There were over 550 unique users of GANGA running over 40,000 sessions at more than 70 sites in September 2010.

GANGA is a job-management system; it handles the complete life cycle of a job. It can build and configure applications to run on various types of resources (*e.g.*, local machines, batch systems, the grid, *etc.*). It can submit jobs to different resources in a transparent way; *i.e.*, the user does not need to know the details about how to submit jobs to different backends because GANGA handles this for them. GANGA monitors the users' jobs and downloads the output for them automatically when the jobs are completed. It can also split jobs (according to a number of

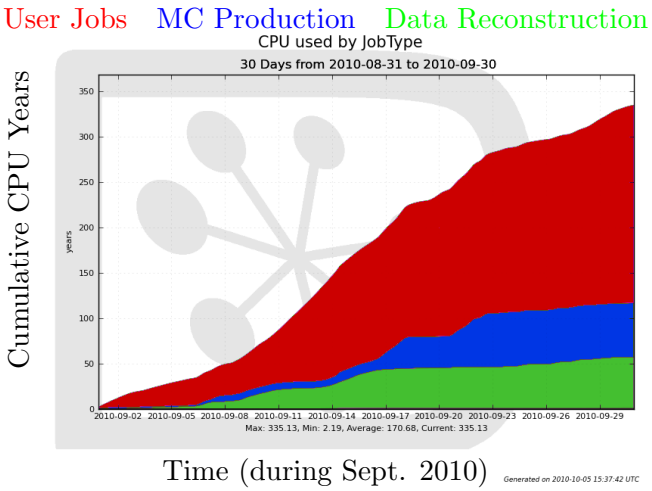

Figure 1. LHC*b* used over 300 CPU years (shared resources only) in September 2010. Approximately 65% of this was consumed by user jobs (top red band), 18% by Monte Carlo production jobs (middle blue band) and 17% by data reconstruction jobs (bottom green band). It is worth noting that LHC*b* was not taking data for most of this month; for this reason the data reconstruction fraction is atypically low.

criteria) and merge the output from lists of jobs. In this way, `GANGA` lives up to its mantra: *configure once, run anywhere.*

`GANGA` is not just a job submission program, it is a job-management system. `GANGA` uses an `XML` repository to persist information on the users' jobs. *E.g.*, when the job was created, when it was submitted, information obtained from the backend about the job, *etc.* are stored in the repository. It allows users to copy jobs, resubmit jobs, remove jobs and kill running jobs. In short: it provides users with an easy and efficient way to manage large numbers of jobs. It is also worth noting that `GANGA` can persist any `GANGA` object in the repository for the user; this is another very useful feature (that is discussed in more detail below).

The LHC*b* plug-in for `GANGA` is designed to seamlessly incorporate both the LHC*b* analysis software, based on the `GAUDI` framework [3], and the `DIRAC` WMS/DMS software into the core `GANGA` framework. Some of the job-related features provided by the `GANGA` LHC*b* plug-in are:

- automatic collection of user-modified LHC*b* software for inclusion in the input sandbox;
- the ability to check out (from `SVN` [4]) and build LHC*b* software packages;
- automatic discovery of output files (from LHC*b* software applications) for inclusion in the output sandbox;
- a backend to handle job submission to `DIRAC`;
- many built-in methods that make `DIRAC` features easily accessible from within `GANGA`;
- the ability to split jobs according to which sites have replicas of the input data;
- the ability to merge LHC*b* data files.

This list is not exhaustive; the LHC*b* plug-in provides a large number of useful features to LHC*b* users.

`GANGA` does not just handle jobs, it also handles data files and data sets. Full support for

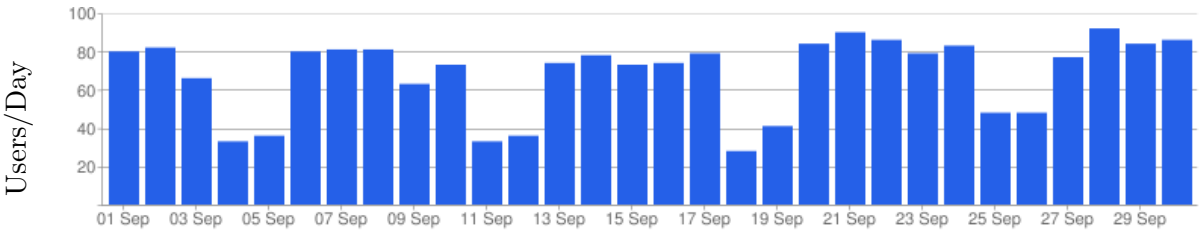

Figure 2. Unique GANGA users per day in September 2010. Almost 100% of LHC*b* virtual organization members who access the grid use GANGA.

logical[1] and physical[2] files is provided by the GANGA LHC*b* plug-in. Some of the data-related features provided are the abilities to:

- download logical files locally;
- upload physical files to grid storage elements;
- replicate logical files to other storage elements;
- obtain metadata for logical files;
- obtain the full list of replicas for logical files;
- remove logical files entirely from grid storage.

This list is also not exhaustive. LHC*b* users can also run the bookkeeping GUI from within GANGA and the files that they select are automatically stored as a data set in GANGA. There is also a GANGA object that can make bookkeeping queries. Like all GANGA objects, it can be persisted in the GANGA repository; thus, LHC*b* users can save their bookkeeping queries in GANGA and at any time have GANGA perform the query to obtain an up-to-date list of data files. Given all of the features that GANGA provides LHC*b* users, it is not hard to see why nearly 100% of all LHC*b* VO members access the grid via GANGA (see Figure 2).

3. DIRAC

As stated above, DIRAC is the WMS and DMS for LHC*b*. DIRAC is made up of both central services and distributed agents. DIRAC uses WLCG [5] resources and middleware to carry out distributed computing tasks for the LHC*b* virtual organization. It was the first system to use the (now commonly used) *pilot agent* paradigm on the grid. Prior to submitting jobs to a grid worker node, DIRAC first submits a so called pilot job. The pilot first checks that the worker node is suitable for running LHC*b* jobs (*e.g.*, it checks the local software environment, available disk space, *etc.*) prior to requesting jobs from the central WMS. The main benefits afforded to LHC*b* users by DIRAC are:

- the ability to monitor jobs via a web portal;
- central management of site, storage element and catalog masks;
- increased job success rates due to DIRAC's many failover mechanisms;
- fairshare at the WMS level.

[1] A logical file is a file that is stored on a grid storage element or elements. The logical file name does not refer to a block of memory on a device, but rather to the handle for the file in a catalog that contains the list of physical paths required to access the file on each of the grid storage elements on which it is replicated.
[2] A physical file is a file that is stored on a specific storage device. The physical file name can be used to access a block of memory on the device.

Of course, users also benefit from all the behind-the-scenes work done by the production team and the DIRAC developers investigating issues with grid sites, production jobs, *etc.*

DIRAC uses many failover mechanisms to increase the success rates of user jobs. *E.g.*, all output data are stored on a temporary storage element until they reach their final destination. Operations such as catalog registration are delayed until the files are where they are supposed to be. All output data are also automatically replicated to alleviate problems with storage elements at a later date. If TURLs cannot be obtained or if software downloads fail, then jobs are automatically rescheduled. Job status transitions and metadata are cached during the job; thus, they can be recovered if there are network interruptions and/or service instabilities. DIRAC also automatically uploads oversized output sandboxes to grid storage and seamlessly retrieves them for LHC*b* users. In short: DIRAC has a workaround for everything that a job does in case of failure(s).

Not only do LHC*b* users benefit greatly by using DIRAC (via GANGA), but the LHC*b* virtual organization also benefits in the following ways:

- unification of workload and data management;
- mutually beneficial coexistence of user and production jobs;
- having one central task queue means that the VO's highest priority jobs always run first;
- all grid activities are accounted for centrally.

DIRAC also uses a *filling* mode that makes it possible to run more jobs without increasing the load on the grid. This works by scheduling a second job to run on a worker node if an LHC*b* job has finished running on that node but enough time is left for the second job to run; thus, a second job can be run without the need for a second pilot.

4. DAST

In the spring of 2010, following what was done in ATLAS, LHC*b* formed the Distributed Analysis Support Team (DAST). Each week, one member of the team is on shift. During the week it is the shift-taker's duty to either answer or redirect all emails sent to the LHC*b* distributed analysis mailing list. Simple user errors are often answered very quickly by other users; this helps reduce the load on the DAST shifter. Obvious grid-site issues are redirected to the production team and the DIRAC developers, while obvious LHC*b* software issues are redirected to the maintainers of the specific package that the user is having problems with. All other issues are handled by the shifter directly. A new feature recently added to GANGA (the `report` function) allows users to upload their jobs (including the sandbox, local environment, recent GANGA history, *etc.*) to a server where it can be downloaded by the experts. This feature often greatly speeds up the debugging process.

5. Conclusions

Overall, the distributed analysis experience in LHC*b* to date has been positive. The clear division of labor between GANGA and DIRAC has benefitted LHC*b* and its users greatly. Both of these packages have performed well during the LHC data taking in 2010 and are expected to continue to perform well in the future.

References

[1] J.T. Moscicki *et al.* [The GANGA Developers], *Ganga: a tool for computational-task management and easy access to Grid resources*, Comp. Phys. Comm. **180**, Issue 11 (2009).
[2] `https://lhcbweb.pic.es/DIRAC`
[3] `http://proj-gaudi.web.cern.ch/proj-gaudi/`
[4] `http://subversion.apache.org/`
[5] E. Laure *et. al.*, *Enabling Grids for e-Science: The EGEE Project*, EGEE-PUB-2009-001.