

APEL: An implementation of Grid accounting using R-GMA

Rob Byrom^a, Roney Cordenonsi^b, Linda Cornwall^a, Martin Craig^a,
Abdeslem Djaoui^a, Alastair Duncan^a, Steve Fisher^a, John Gordon^a, Steve Hicks^a, Dave Kant^a,
Jason Leake^c, Robin Middleton^a, Matt Thorpe^a, John Walk^a, Antony Wilson^a

^a CCLRC, Rutherford Appleton Laboratory

^b Queen Mary, University of London

^c Objective Engineering Limited

Abstract

This article describes the implementation of an accounting tool in the LHC Computing Grid (LCG): a distributed computing grid project consisting of over 100 resource centres and more than 10,000 CPUs.

APEL (Accounting Processor for Event Logs) parses batch, system and gatekeeper logs generated by a site and builds accounting records, which provide a summary of the resources consumed based on attributes such as CPU time, Wall Clock Time, Memory and grid user DN. The accounting data is published into the R-GMA information and monitoring system, and archived for processing by a graphical front-end utilised by the accounting web tool.

1. LCG Accounting Overview

In the LCG [1] grid environment, the computing resources, the application data and the grid users belonging to Virtual Organisations (VO) are distributed. Jobs submitted by these users may be sent to computing resources where the data is stored locally, or may go to remote resources where there are available job slots to reduce queue times. Jobs that run on LCG resources must be properly accounted for, so that the resources consumed by VOs and the resources provided by sites (resource centres), as a function of time can be determined.

In this paper we distinguish between two flavours of grid accounting; “Grid Job accounting” in which a job usage record provides a complete description of resource consumption and, “Real-time Grid accounting”, which is based on an incremental determination of resource value while the job is being executed. In the former, the cost of computing is determined on the basis of what was done (after job execution), whilst in the latter, data feeds into a pricing model on which a cost for consumption is determined in real time.

An implementation of real-time accounting (DGAS [2]) is under development within the gLite [3] middleware framework of the EGEE [4] grid-computing project.

Below we summarise the implementation of a web-based job accounting tool that collects usage records from distributed resources using R-GMA [5] to a central point, and consolidates the information for presentation on the web.

2. Data Collection Mechanism and R-GMA

The collection of accounting usage records is done through R-GMA, an implementation of the Grid Monitoring Architecture (GMA) proposed by the Global Grid Forum (GGF). GMA models the information and monitoring system infrastructure of a grid as a set of Consumers (which request information), Producers (which provide information) and a registry, which mediates the communication between the producers and consumers.

In LCG accounting, each site publishes its own accounting data using an R-GMA primary producer using its locally assigned R-GMA server. To collect the data from all participating sites, data is streamed to a centralised database via a secondary producer.

The central database is located on the Grid Operations Centre (GOC) and is used to provide a web front end, generating a summary of resource usage across the LCG grid network.

3. APEL

Apel is a log processing application which is used to interpret gatekeeper and batch system logs to produce accounting records. It currently supports PBS and LSF batch systems but can easily be extended to support other variants. This is possible because of a newly developed plug-in architecture which separates the core functionality from the actual log parsing (Figure 1).

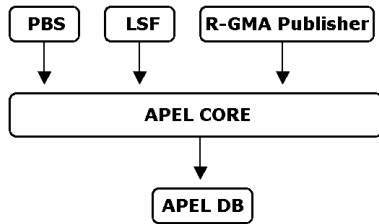


Figure 1: Apel provides a plug-in which parses PBS and LSF batch systems logs. A plug-in exists to publish accounting records into R-GMA. Each plug-in connects to the underlying database via the Apel core.

A complete accounting record is composed of (among others) the grid user, the job id of the submitted job and the resources used when executing the job. This information is typically dispersed between several different log file types such as those produced by the gatekeeper or batch system. For resource usage, a query is issued to the site's GIIS to lookup the CPU performance for the computing nodes where the job was executed. Apel attempts to collect all this piecemeal information together and manages it within a database. A further process carried out by Apel then attempts to join the data together to produce a list of final accounting records with all necessary details filled-in.

Apel is then used to publish the generated accounting records into R-GMA whereby they are collated at the GOC using an R-GMA secondary producer, as shown in Figure 2.

Apel provides support for republishing the complete local copy of accounting records to R-GMA (in cases when the GOC was offline). It also provides a mechanism for reliable delivery using a basic integrity check to compute the number of records that were last published compared with the actual count stored on the GOC.

Each accounting record is unique and there is only one record per grid job. The records may be consolidated in different ways to provide high-level views [6] of accounting data, such as the total CPU time consumed for each LHC VO at the tier-1 computing centres, as shown in Figure 3.

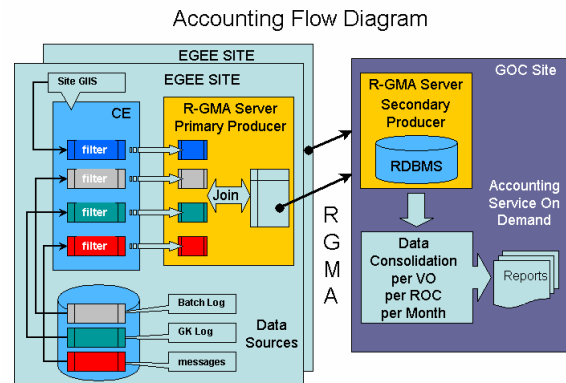


Figure 2: Accounting flow diagram providing a global overview of the data collection process (APEL), and the web reporting service.

4. Normalisation

Normalisation is an important issue when dealing with VO applications that run over heterogeneous resources. Within LCG, most sites deploy a PBS (OpenPBS, Torque, PBSPro) batch system which supports auto-scaling features. On a machine-by-machine basis, multiplicative factors, determined from published SI2K benchmark tests based on CPU/FSB information, are used to scale the CPU and wall clock times in the usage record. As a consequence, a heterogeneous farm is internally normalised into a homogeneous farm, and may be described in terms of a single (weighted averaged) SI2K benchmark that is taken into consideration when consolidating VO data at the GOC.

Note, that the GLUE Schema [7] does attempt to address the heterogeneous cluster problem but is not integrated at the batch log level where there is no differentiation between sub-clusters.

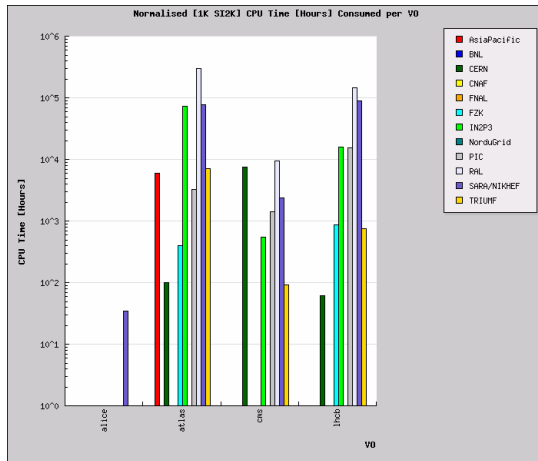


Figure 3: The CPU time (normalised to 1K.SI2K.Hours) provided by Tier-1 computing centres supporting LHC VOs during the first quarter of 2005. Data from some LCG Tier-1 sites (BNL, FNAL and NorduGrid) are not presented here.

It is recognised that most sites are not homogeneous environments and therefore describing a batch farm in terms of an average SI2K value is not entirely sufficient. The actual performance of a processor can differ from the published SI2K benchmarks due to compiler dependencies. Sites that do not apply the auto-scaling features use an SI2K for the slowest processor in the batch farm.

Within LCG, the majority of sites follow the same recipe (weighted averages based on official SI2K benchmarks), so the normalisation method described here is only an approximation but may be regarded as a reasonable treatment of the data.

5. Future Work

As LCG matures towards production quality middleware (gLite) and more and more resources are made accessible through the grid, it will be necessary to support a wider range of batch systems. The accounting infrastructure (APEL/DGAS) will support the Condor batch system, in addition to PBS and LSF. Additionally, hooks into other gLite components such as the Job Repository can be used to furnish accounting records with user credential information.

Accounting VO usage across grids – as is the case for LCG whose peers are EGEE, OSG [8] and NorduGrid [9], each with their own accounting infrastructure - presents an entirely different challenge involving data sharing. A possible way forward may involve adopting a centralized approach for reporting - as is done in TeraGrid [10] - using web service technologies such as OGSA-DAI [11].

References

- 
- [1] <http://lcg.web.cern.ch/LCG/>
 - [2] <http://egee-jra1-wm.mi.infn.it/egee-jra1-wm/org.glite.dgas.shtml>
 - [3] <http://glite.web.cern.ch/glite/>
 - [4] <http://public.eu-egee.org/>
 - [5] <http://www.r-gma.org/>
 - [6] <http://goc.grid-support.ac.uk/gridsite/accounting/>
 - [7] <http://infnforge.cnaf.infn.it/projects/glueinfnodel/>
 - [8] <http://www.opensciencegrid.org/>
 - [9] <http://www.nordugrid.org/>
 - [10] <http://www.teragrid.org/>
 - [11] <http://www.ogsadai.org.uk/>