DIRAC optimized workload management

To cite this article: S K Paterson and A Tsaregorodtsev 2008 J. Phys.: Conf. Ser. 119 062040

View the article online for updates and enhancements.

Related content
- Workload analyse of assembling process
L D Ghenghea
- DIRAC pilot framework and the DIRAC Workload Management System
Adrian Casajus, Ricardo Graciani, Stuart Paterson et al.
- User analysis of LHCb data with Ganga
Andrew Maier, Frederic Brochu, Greg Cowan et al.

Recent citations
- Distributed Late-binding Scheduling and Cooperative Data Caching
Antonio Delgado Peris et al
- Antonio Delgado Peris et al
DIRAC Optimized Workload Management

Stuart K. Paterson\textsuperscript{1} and Andrei Tsaregorodtsev\textsuperscript{2} on behalf of the LHC\textit{b} DIRAC Team

\textsuperscript{1} CERN, CH-1211, Geneva, Switzerland
\textsuperscript{2} CPPM, Marseille, France

E-mail: stuart.paterson@cern.ch, atsareg@in2p3.fr

Abstract. The LHC\textit{b} DIRAC Workload and Data Management System employs advanced optimization techniques in order to dynamically allocate resources. The paradigms realized by DIRAC, such as late binding through the Pilot Agent approach, have proven to be highly successful. For example, this has allowed the principles of workload management to be applied not only at the time of user job submission to the Grid but also to optimize the use of computing resources once jobs have been acquired. Along with the central application of job priorities, DIRAC minimizes the system response time for high priority tasks. This paper will describe the recent developments to support Monte Carlo simulation, data processing and distributed user analysis in a consistent way across disparate compute resources including individual PCs, local batch systems, and the Worldwide LHC Computing Grid. The Grid environment is inherently unpredictable and whilst short-term studies have proven to deliver high job efficiencies, the system performance over an extended period of time will be considered here in order to convey the experience gained so far.

1. Introduction

The DIRAC (Distributed Infrastructure with Remote Agent Control) Workload and Data Management System (WMS) \cite{1} is made up of central services and distributed agents. The main aim of the project is to integrate all of the heterogeneous compute resources available to LHC\textit{b} such as individual PCs, site clusters and Grids while reducing the human intervention required to utilize them. DIRAC has now matured into a complete community solution allowing coordination and management of the computing activity of a Virtual Organization across disparate resources and Grids. The DIRAC WMS employs several advanced optimization techniques in order to increase the efficiency and responsiveness of the system. The use of a central \textit{Task Queue} enables VO-centric management and accounting of workloads as well as application of job prioritization and VO policies. During the recent DC06 activity, 1.5 Million jobs were executed at over 120 distinct Grid sites with up to 10K concurrently running jobs \cite{2}.

This paper describes the paradigms extensively used by the DIRAC WMS and the workload management optimizations performed to support the LHC\textit{b} VO requirements. DIRAC realizes the \textit{PULL} scheduling approach and was the first system to introduce the now widely accepted concept of Pilot Agents on the Grid. The Pilot Agent paradigm is described in Section 3.

2. Overview of DIRAC

The DIRAC software architecture is based on a set of distributed, collaborating services. Designed to have a light implementation, DIRAC is easy to deploy, configure and maintain.
on a variety of platforms. Figure 1 outlines the relationship between resources, services, agents and clients which form the main components of DIRAC. These will be briefly discussed in turn below.

Figure 1. Overview of the main components of DIRAC: Resources; Services; Agents and Clients and how these components interact.

2.1. Clients
Clients in DIRAC can simply be considered as submitters of jobs or requests. Clients include the Bookkeeping Query Webpage [3], which requests information about datasets and their replicas on the Grid. For distributed analysis and user production jobs, clients interact with the central services via the DIRAC Application Programming Interface (API) [4] [5].

For LHCb production tasks, the Production Manager Console is used. This provides a general framework for the construction and management of production tasks and provides a GUI for users [6]. There is also a File Catalogue Browser which makes use of the Data Management components of DIRAC [7].

2.1.1. Services
The Services highlighted in Figure 1 accept requests from Clients and Agents. The DIRAC Job Management Services will be described individually in Section 6. They perform vital operations for production and distributed analysis jobs, such as uploading any necessary files for application steering and checking any requested input data is available.

The Configuration Service provides necessary site dependent information for Agents. The Job Monitoring Service keeps track of changes in job status. Similarly, the Bookkeeping Service
will log selected results to provide a history about jobs in case of failure. The role of the Job Accounting Service is to provide statistics on job efficiency and resource usage at the site and user level.

2.1.2. Agents Agents are deployed close to resources and form an Overlay Network. On LCG, Pilot Agents are deployed to Worker Nodes via the Resource Broker. These Pilot Agents in turn start a DIRAC Agent. On individual PCs and site clusters this can be done ‘by hand’.

2.1.3. Resources DIRAC can integrate resources such as Individual PC’s, site clusters and Grids. This is reflected in Figure 1, with the only difference from the perspective of Services being how the Agents are deployed in each context.

3. Pilot Agent Paradigm
The presence of jobs in the WMS central Task Queue triggers the submission of Pilot Agents to the Grid. LCG jobs are Pilot Agents for the WMS, the actual Workload Management is performed by DIRAC. Pilot Agents run on Grid WNs and are submitted by the DIRAC WMS using the credentials of the user. They reserve the resource for the immediate use, requesting jobs from the WMS. Pilot Agents steer job execution as well as operations needing to be performed after the job has finished such as uploading of data to a Grid SE.

After a Pilot Agent arrives at a WN, the local environment is checked prior to job scheduling. If any problems are discovered, such as a lack of local disk space or inconsistencies in the software environment, the Pilot can terminate gracefully and free the resource. Site problems can be reported to central WMS services such that human intervention can be initiated as necessary. In this way, Pilots submitted on behalf of the user, the door is opened to further optimizations on the level of that user. In the past, the main purpose of submission systems such as DIRAC was purely to deploy jobs to the Grid in as quick a manner as possible. Now, however, it becomes important to optimise the use of the resources once they have been obtained by Pilot Agents. The possibility of running further jobs for users on obtained resources is further discussed in Section 4.1. The use of Pilot Agents also means that the DIRAC Task Queue is the only waiting queue in the system. This allows the LHCb VO to impose prioritisation policies in one place, something that the presently available LCG tools cannot provide.

3.1. Job Scheduling in DIRAC
Matching time is defined as the time between a Pilot Agent requesting a job from the WMS and the job being delivered to the computing resource. LHCb Monte Carlo production jobs have fairly uniform requirements, normally including a particular amount of CPU time.

With uniform production jobs, using Task Queues to enable job scheduling is natural. However, distributed analysis tasks have chaotic requirements and present the most demanding task for the Matcher service. Figure 2 shows the matching times for 18,000 real user distributed analysis jobs.

The matching time can also be viewed as the ‘overhead’ associated with the PULL scheduling paradigm. 92% of jobs were matched in under 1 second for this sample and therefore the matching time is a negligible factor in the overall lifetime of DIRAC jobs. Furthermore, this demonstrates that the PULL scheduling paradigm originally employed for LHCb production tasks scales well to the chaotic requirements of distributed data analysis jobs.
Figure 2. User jobs with many varied requirements present the biggest challenge for the PULL paradigm. DIRAC matcher times for 18K real user jobs submitted between January and August 2007. 92% of jobs are scheduled in under 1 second.

Job Matching in DIRAC involves checking the job requirements against the resource requirements presented by the Agent in a ‘double matching’ mechanism. This is a ClassAd-based [8] approach that allows both resources and jobs to define criteria of eligibility. In addition to hardware specific requirements, Agents can easily be configured to be ‘specialised’ or ‘fully generic’. Specialised Agents could request specific jobs by their JobID or could request only jobs from a particular user. Fully generic Agents would place no further requirements on jobs and offer interesting optimizations in VO Workload Management. Generic Agents are further discussed in the next Section.

4. Generic Pilot Agents
Currently Pilot Agents are submitted on demand using the credential of the job owner. Submission of Generic Pilots would similarly be triggered by waiting jobs in the central Task Queue however, a single trusted credential would be used for all submitted Pilots. As mentioned in Section 3, Pilot Agents create an Overlay Network that masks the underlying diversity of compute resources. This can be considered as a pool of resources available to the DIRAC WMS. With non-generic Pilot Agents, not all pending tasks are eligible to be run on all resources. Generic Pilots would ensure that the VO can decide, for all submitted pilots, which task should be delivered to the available resources.

Furthermore, optimizations are currently possible at the level of the individual user whereby additional jobs can be run by a deployed Agent in the same computing slot. This optimized ‘filling mode’ serves to reduce the load on LCG whilst simultaneously offering a reduced starting time for tasks [4]. Currently this is restricted to the owner of the Pilot Agent deployed to Grid resources and their workload. Generic Pilot Agents would allow Agents sent on behalf of the
VO to immediately run the highest priority task for the VO. All Agents in the dynamic pool of resources could run one or more eligible tasks for all users that are members of the VO. Since LHCb distributed analysis tasks generally have a higher priority than production tasks, this is an important way to ensure a minimised start-up time for these jobs and offers an elegant solution to the job prioritization problem. The requirements from Grid sites are also simplified because the VO can request long queues (e.g. 24hr) everywhere and the local batch queue waiting times can be masked. Section 5 describes how job priorities can be assigned in the central Task Queue.

4.1. Proposed Usage of gLexec

![Diagram](image)

**Figure 3.** Overview of the DIRAC Filling Mode highlighting proposed gLexec usage. An Agent occupying a compute resource can make additional job requests when there is sufficient CPU / wall-clock time available.

DIRAC has been designed to accommodate Generic Pilot Agents and gLexec in this context means a tool to allow the switching of user identity on Grid WNs. Depending on the site configuration, gLexec could simply log the DN and time of the beginning of user task execution or could perform a switching of the local user identification. As shown in Figure 3, the DIRAC Agent can first make a request for tasks and then call gLexec to identify the user with the site. Assuming sufficient CPU and wall-clock time is available, subsequent requests for tasks can be made in order to ‘fill’ the available computing slot.

5. Job Prioritization and Policy

The use of a central Task Queue for a community allows the DIRAC WMS to apply policies for the prioritizations of tasks. Through the use of Pilot Agents, DIRAC ensures that the highest priority eligible jobs in the Task Queue can be sent to the first available Agent. This reduces the problem of applying job priorities on the Grid to correctly assigning the priority of jobs in the VO central Task Queue in a similar manner to standard batch systems.

The possibility of reusing standard batch system components also exists. A DIRAC Job Priority Agent running in the WMS can work on the central Task Queue assigning and updating priorities. Information from already running jobs and the accounting system can be used to
evaluate user quotas. The Job Priority Agent can make requests to standalone schedulers, for example, Maui was recently demonstrated to integrate with the DIRAC WMS [9].

6. DIRAC Workload Management System
An overview of the key WMS components is given in Figure 4. The function of each component will be briefly described in the context of the progression of jobs through the WMS.

![Figure 4. Overview of the DIRAC WMS.](image)

After jobs arrive in the WMS the Job Receiver assigns a JobID and enters the job into the Job Database. After this, jobs with an input data requirement pass through the Data Optimiser before being added to the central Task Queue. Jobs without an input data requirement are immediately entered into the Task Queue. At this point the Agent Director submits Pilot Agents to the Grid as necessary. The Agent Monitor tracks the status of submitted Pilots and can trigger submission of further Pilots as required. For the production activity, currently run under a single credential, all Pilot jobs are available to run the highest priority task assuming that the site and job requirements match. Submission of Pilots on-demand means that a continuous flow of Agents are requesting jobs in the saturated regime. This greatly improves system responsiveness for high priority tasks. The use of Generic Pilot Agents would allow all VO jobs to run in this manner.

After a Pilot Agent starts on the WN it makes a request to the Matcher to receive an eligible job. Subsequent to the job being delivered, the Job Wrapper is started on the WN along with a Watchdog process to monitor the health of the job. The Job Wrapper is responsible for providing access to input data if required [5], and then starts the desired application. After successful execution, the Job Wrapper will ensure any specified output data or large sandbox files are uploaded to a Grid SE. Small output sandbox files are currently uploaded to the central WMS.
One of the recent WMS developments is to perform preemptive checks on jobs that could be impossible to run successfully. The **Job Sanity Agent** has been used to identify and meaningfully fail jobs before wasting CPU resources unnecessarily. Some of the typical causes of failure include incorrectly constructed jobs, and job output data already existing in the file catalogue. Such features are useful to ensure consistency in both large, centrally managed production activities as well as for user tasks.

Workload management was the first major challenge for DIRAC but several years of development have resulted in a stable platform for the LHCb VO workload. Experience from the DC06 data processing phase has introduced a tighter coupling between Workload and Data Management in DIRAC. Data produced at LHCb Tier-1 sites was stored on tape and the data processing phase occurred afterwards. Initially files were attempted to be staged from the WNs by the DIRAC **Job Wrapper** but this resulted in significant wall-clock time being consumed without any CPU usage across the Tier-1’s. In order to cope with this, data had to be recalled from tape in advance of job execution. The DIRAC Stager performs this task and is described in Section 6.1.

### 6.1. Central Stager Service

Workload and data management on the Grid also extends to storage element file access. This is essential for reliable file transfers as well as job execution on the WN. For files stored on tape, it is essential to ensure that replicas are staged prior to the submission of Pilot Agents. Otherwise jobs can be running on site WNs without consuming any CPU. The LHCb DC06 reconstruction and stripping activity was performed using the Stager service.

![Figure 5. Overview of the DIRAC Stager service highlighting interactions with the WMS and DMS.](image)

The DIRAC Stager is shown in Figure 5. Jobs arriving in the WMS pass through the **Job Sanity Agent** and subsequently to the **Data Optimiser** where replicas are checked in the LFC [10]. At this point the **Data Optimiser** has all the necessary replica information and passes the job to the StagerDB. For centrally managed data processing activities, the DIRAC production management infrastructure has a database that ensures a fair distribution of tasks across the
Tier-1 sites. In this case the destination site is chosen from consultation with the database. For user distributed analysis tasks, preference is given to any site with larger numbers of replicas on disk storage before deciding upon a single destination site. Since jobs can be ‘rescheduled’ the chosen destination sites are retained such that the data is not staged simultaneously across eligible sites.

The Stager Agent subsequently makes remote staging requests and monitors the file until it is brought online. Stage requests are repeated several times a day as necessary and jobs can be meaningfully failed if not all files are brought online. The frequency of repeating stage requests and the maximum allowed time is configurable, e.g. requests could be repeated 6 times per 24hr period. After the maximum allowed time, files not yet staged are flagged as ‘problematic’ such that jobs can be recreated without these replicas. Problematic files can then be resolved by the Data Management integrity checking infrastructure [7]. This allows a swift discovery of tape or data consistency problems at Tier-1 sites and eliminates the chance that jobs arrive at a site and consume wall-clock time whilst waiting for files to become staged.

Statistics on staging performance for the LHCb Tier-1 sites were gathered during the DC06 activity. Figure 6 is a plot of the SRM metadata response time against the number of files in each query. Two dCache and two Castor sites were chosen to highlight the characteristic response specific to the backend storage system implementation. These plots show that the Castor sites were more stable during DC06 and seem to be less affected by the size of each query. Clustering of results around 50 files is due to the behaviour of the Stager in the saturated regime, 50 is the maximum number of files for any single metadata query to the sites.

![Figure 6](image_url)

**Figure 6.** DC06 staging performance at CERN, CNAF, PIC and GridKa. These plots show the SRM metadata response time against the number of files in each query.
The DIRAC Stager could be extended to maintain a centrally manage a list of stage requests and pin lifetimes which allows full control over the underlying storage system implementation. This would allow the LHCb VO to automatically manage disk caches at the Tier-1 sites. Stage requests can be generated by jobs, as described above, by data transfer requests, and by VO staging requests (e.g. for particular datasets of interest). As shown in Figure 5, the Stager is used both by the DIRAC WMS and Data Management System in a consistent way.

7. Conclusions
The advanced optimization techniques employed by the DIRAC Workload and Data Management System result in a high degree of job efficiency and system responsiveness. Late binding through the Pilot Agent approach, has proven to be highly successful and there is a negligible overhead in using the PULL scheduling paradigm. The DIRAC optimized Filling Mode allows workload optimization at the level of the individual user, however, the performance gains associated with the use of Generic Pilot Agents are eagerly anticipated. Workload optimizations in the central Task Queue allow VO-policy to be applied in one central location and also provide an elegant solution to the job prioritization problem. Recent developments such as the DIRAC Stager were essential in ensuring an efficient DC06 data processing phase. DIRAC meets the requirements of the LHCb VO and has now evolved into a complete community Grid solution.

References