DIRAC3 – the new generation of the LHCb grid software

2010 J. Phys.: Conf. Ser. 219 062029

(http://iopscience.iop.org/1742-6596/219/6/062029)

Journal of Physics: Conference Series **219** (2010) 062029 doi:10.1088/1742-6596/219/6/062029

DIRAC3 - the new generation of the LHCb grid software

A Tsaregorodtsev [1],N Brook [2], A Casajus Ramo [3], Ph Charpentier [4], J Closier [4], G Cowan [5], R Graciani Diaz [3], E Lanciotti [4], Z Mathe [6,4], R Nandakumar [7], S Paterson [4], V Romanovsky [8], R Santinelli [4], M Sapunov [1], A C Smith [4], M Seco Miguelez [9], A Zhelezov [10]

[1] Centre de Physique des Particules de Marseille, 163 Avenue de Luminy Case 902 13288 Marseille, France
[2] H. H. Wills Physics Laboratory, Royal Fort, Tyndal Avenue, Bristol BS8 1TL, UK
[3] University of Barcelona, Diagonal 647, ES-08028 Barcelona, Spain
[4] CERN CH-1211 Genève 23, Switzerland
[5] School of Physics and Astronomy, University of Edinburgh, Edinburgh EH9 3JY, UK
[6] School of Physics, University College Dublin, Belfield, Dublin 4, Ireland
[7] Rutherford Appleton Laboratory, Chilton, Didcot Oxon. OX11 0QX, UK
[8] Institute of High Energy Physics, Protvino, Moscow Region, RU-142280, Russia
[9] University of Santiago de Compostela, Campus Universitario Sur, ES-15706 Santiago de Compostela, Spain
[10] University of Heidelberg, Physics Institute, Philosophenweg 12 69120 Heidelberg, Germany

E-mail: atsareg@in2p3.fr

Abstract. DIRAC, the LHCb community Grid solution, was considerably reengineered in order to meet all the requirements for processing the data coming from the LHCb experiment. It is covering all the tasks starting with raw data transportation from the experiment area to the grid storage, data processing up to the final user analysis. The reengineered DIRAC3 version of the system includes a fully grid security compliant framework for building service oriented distributed systems; complete Pilot Job framework for creating efficient workload management systems; several subsystems to manage high level operations like data production and distribution management. The user interfaces of the DIRAC3 system providing rich command line and scripting tools are complemented by a full-featured Web portal providing users with a secure access to all the details of the system status and ongoing activities. We will present an overview of the DIRAC3 architecture, new innovative features and the achieved performance. Extending DIRAC3 to manage computing resources beyond the WLCG grid will be discussed. Experience with using DIRAC3 by other user communities than LHCb and in other application domains than High Energy Physics will be shown to demonstrate the general-purpose nature of the system.

1. Introduction

The LHCb Collaboration is constructing and operating one of the four experiments running on the LHC proton-proton collider at CERN, Geneva Switzerland. The DIRAC project was originally started in 2002 as a tool to support a large-scale production of the Monte-Carlo experiment modelling data for the LHCb Collaboration. The main design goals were defined as the following:

- Transparent access to heterogeneous computing resources available to LHCb at that time. These included various computing clusters in the LHCb collaborating institutes, as well as computing facilities at CERN but also individual workstations of the LHCb users.
- An integrated production system with minimal manpower requirements for its maintenance and operation. In particular, no special support by the site managers was considered to be a mandatory feature.

After a quick prototyping phase, the DIRAC project was successfully used in production in 2003 and in subsequent productions with resources provided both by the LCG computing grid and independent computing clusters (called DIRAC sites). We would like to note that the LHCb data production run in summer 2004 was the first successful demonstration of the massive usage of the LCG grid resources. This was largely due to the use of the DIRAC Workload Management System, which boosted significantly the efficiency of the production jobs. Based on the successful experience with the DIRAC usage for the Monte-Carlo data production, it was decided to extend the system to support also Data Processing and User Analysis activities. This resulted in the development of the second generation of the DIRAC project software that was now addressing most of the LHCb distributed computing tasks. The successful demonstration of the DIRAC functionality and performance allowed adopting it as the basis of the LHCb distributed computing system in the Computing TDR [1].

In order to deliver the highest quality product by the time of the LHC operation, it was decided that the project should undergo a profound review of its architecture and design solutions. The review took place in the end 2005 – beginning 2006 with multiple recommendations taken on board by the project development team. As a result, the DIRAC software was completely reengineered in order to meet the overview conclusions. A new, third generation of DIRAC was rolled out in summer 2008 and progressively put in production.

Now the DIRAC project provides a complete set of tools to support the whole LHCb data processing chain starting from Monte-Carlo Data Production and real Data Distribution to the final end User Analysis tasks. It is important to note that all the DIRAC facilities are implemented within the same base Framework by a single concerted team of developers sharing basic development tools and design solutions. This allowed a relatively small group of developers to deliver quickly a comprehensive suit of tools covering all the needs in the distributed computing management of such a large community as the LHCb Collaboration.

In this paper we will discuss the details and the outcome of the DIRAC Review in Section 2. The new features of the latest DIRAC project generation are covered in the subsequent sections. In Section 3 we present the new refurbished Framework. The new features of the Workload Management System are described in Section 4. In DIRAC much attention is paid to failure resilience, these aspects are discussed in Section 5. Since the DIRAC system is used with an increasingly high number of non-expert users, the importance of the user-friendly interfaces became evident. The new Web Portal allowing to monitor and control all the DIRAC activities is presented in Section 6. A higher-level Production System and other services based on the DIRAC Framework are outlined in Section 7. A summary of the operational experience can be found in Section 8. In Section 9 we present the conclusions and outlook for the future developments.

2. DIRAC Project Review

The review was organized in a serious of meetings of the DIRAC developers and the Review Board, which was composed of 7 experts coming from outside the project. Detailed presentations of all the architectural and technical aspects of the project software were presented and carefully examined by the reviewers. The outcome was summarized in a 40 pages Review Report and contained a list of

strong points of the project as well as identification of areas where improvements were needed. Recommendations to the developers on the possible solutions to identified problems was the most valuable part of the Review Report, which was taken on board as a task list by the DIRAC team.
The following main architecture and design solutions were considered as proven advantages of the DIRAC project:

- The choice of Python as the main programming language of the project was very much appreciated. Indeed in most cases the distributed computing middleware does not require ultimate computational efficiency offered by compiled languages. However, the interpreted languages, like Python, provide for rapid prototyping and development which allows to fix quickly encountered problems and add new features. It allows also easy deployment in various computing environments; for example, it made it easy to port DIRAC to the Windows platform.
- Using MySQL database to store the DIRAC services state has proven to be sufficient for the project purposes. It was noted that the actual database is isolated by a special software layer which allows to change easily the backend database engine and use ORACLE, for example.
- The client/service communication is performed with a light-weight protocol (XML-RPC) with additional standards based authentication mechanisms. This was considered to be adequate as it allowed very high rates of client/service queries without special high-end hardware to be installed.
- Services oriented architecture allowed to build a flexible and scalable system which can be easily extended and adapted to ever increasing needs in the system throughput. It allows deploying the DIRAC services either on a single server or on a set of servers even geographically distributed to achieve the required capacity.
- Modular design applied in various system components allowed to quickly introduce new functionality or make component modifications without affecting other components of the system. Plug-in mechanisms used in different places make it possible having varying functionality depending on particular environment, which can be conveniently described in the system configuration. The system algorithms can be easily changed “on the fly” to reflect changing operational conditions.
- The Workload Management System based on the central Task Queue and Pilot Jobs has proven to be the most adequate solution for the usage of unstable heterogeneous distributed computing resources.

The Review Report recommended to preserve the identified strong points of the project and drew the attention of the developers to a number of problems to be solved and possible enhancements of the software. It was suggested to pay special attention to the following items:

- The DIRAC security model must be fully compliant with the grid standards including a secure Proxy Management system. The Authentication mechanisms must be complemented by a versatile Authorization system to control access to the DIRAC services.
- The use of the computing resources of LHCb must allow for application of the Collaboration policies to manage the priorities attributed to various groups of users and to different types of activities.
- The grid computing resources and grid services being far from stable necessitates building a redundant system where all the vulnerable operations can be repeated using different instances of the services or deferred to a later execution when failing services are stabilizing.
- The more and more intensive use of the system needs a detailed and reactive Monitoring System for both user and data production activities.
- The DIRAC project software must be better structured. In particular, a clear separation between the general purpose services and LHCb specific functionality is required to allow extensions of the project to other user communities.

The DIRAC team in a series of workshops considered the recommendations of the DIRAC Review Board and it was decided that instead of an incremental upgrade of the existing software, a new

development branch should be rather started. In the meanwhile, the ongoing production activities could be supported by the current software with a frozen feature list. In reality, this decision led to a number of difficulties because the same DIRAC team members were involved in both the support of the LHCb Production System and in the development of the new system. Since the two systems were not compatible, the new required features were sometimes developed in both of them causing a certain duplication of the effort. Also splitting the developer attention between the two systems had consequences for the overall work efficiency. In the end of the long development cycle of the new system, the integration and testing phases were very long as all the components were essentially new. For all these reasons, the new generation of the DIRAC software was ready for production only in summer 2008, more than 2 years after the project review was carried out. This experience suggests in particular that, where possible, another development approach with short cycles incrementally adding new features and fixing problems should be privileged as much as possible.

As a result, the new generation of the DIRAC project implements all the elements of the whole LHCb data processing chain. All the new DIRAC software components are built homogeneously in the same solid framework, which is shared by all the developers providing for an efficient and concerted work. In the following we present the highlights of the new DIRAC software generation, which makes it different with respect to the previous one, which was described earlier [2].

3. DIRAC Framework

The DIRAC Secure client/service framework called DISET (standing for DIRAC SEcure Transport) is full-featured to implement efficient distributed systems in the grid environment [3].

3.1. Authentication

First of all, it provides a secure connection layer between clients and services in the system. For that, it uses Python wrappers around the standard OpenSSL libraries and complements them with the grid compliant authentication mechanism based on the GSI standards using X509 certificates. Mutual client and service authentication is performed. Note that the authentication mechanism does not use the standard grid middleware libraries, which makes it much lighter and easier to deploy on various platforms. However, it is supporting the necessary mechanisms for certificate proxy generation and delegation including limited proxies necessary for Workload Management Systems with Pilot Jobs.

3.2. Authorization

Once the client authentication is done its rights to access the service functions must be evaluated. DISET framework provides a mechanism to describe fine-grained service access rules. All the users are performing grid operations as members of certain groups; each group in turn has certain capabilities described as group properties. These properties define which functions on a given service members of the group can execute. Access rights by default can also be described.

Additional rules based on the job ownership can be provided. Jobs ownership can be strictly individual or group-wide. In the latter case all the group members share access to all the job information and controls.

All the authorization rules are described in the DIRAC Configuration Service and can be changed quickly as necessary such that they become immediately applicable to all the distributed DIRAC services.

3.3. Proxy Management

DIRAC provides a full set of tools for managing certificate proxies. The user proxies are generated with an embedded group membership information implemented as proxy extensions. These extensions can coexist with VOMS proxy extensions where the environment requires that.

The DIRAC Proxy Management Service provides the functionality of the Proxy Repository where user proxies can be stored and retrieved as necessary by the DIRAC components. Strict access rules for these sensitive operations are defined. In particular, user proxies can be retrieved by the Multiuser

Pilot Jobs, which run with special credentials to allow a late binding between the user payload ownership and a grid computing slot attribution.

The Proxy Management Service also provides the proxy renewal functions similar to the ones of the MyProxy grid service. It is used by the DIRAC Workload Management System dealing with user jobs that can be waiting for long times in the Central Task Queue; it is also used by the Request Management System for execution of delayed job operations needing special user access rights (see below).

3.4. Security Logging Service

Information about all the operations in the secure distributed environment must be logged with all the details about the client identities and requested services. The Security Logging data must be kept for a long period of time (typically 90 days) in order to help resolution of eventual security incidents. In DIRAC all the client/service communication details from all the distributed services are collected in a special Security Logging Service. It provides a secure backed-up and easy-to-search repository for this data.

4. Workload Management

The DIRAC Workload Management System (WMS) was based from the very beginning on the "Pull" paradigm coupled with the idea of the Pilot Jobs. The advantages of this type of job scheduling are now well demonstrated and all the four LHC experiments are applying this schema with somewhat varying details. This job scheduling mechanism was discussed several times [2]. In the following we present some details about recent developments in this domain [4].

4.1. WMS with MultiUser Pilot Jobs

In the DIRAC WMS the user jobs are submitted to the central Task Queue where they are waiting until they will be picked up by one of the Pilot Jobs as shown in Figure 1. The Pilot Jobs are submitted by Pilot Director components specialized for each type of computing resources available to the LHCb User Community.

The most evident advantage of this approach that the user jobs are only picked up after the execution environment on the Worker Node is checked and guaranteed. However, the real advantage of the Pilot Job approach becomes evident when it comes to the necessity to apply Community wide policies on how to share the common resources. In this case, it is necessary to provide a mechanism to define job priorities for different user groups based on various assumptions, quotas, accounting, etc.

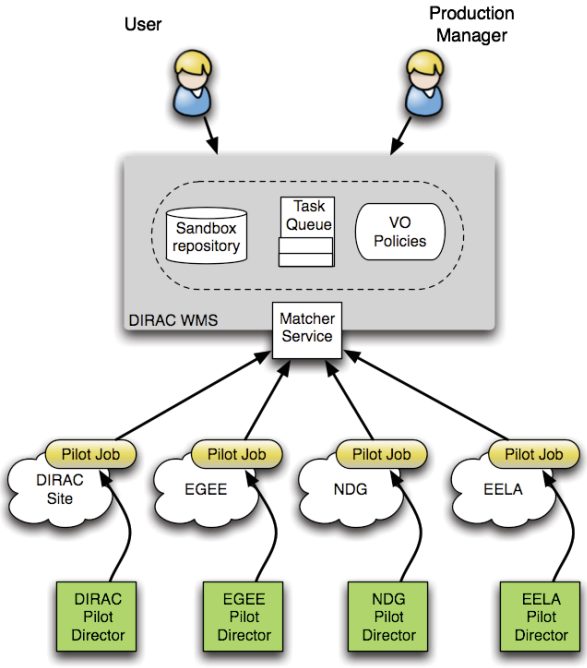

Figure 1 Workload Management System with Pilot Jobs

There are two basic ways how these policies can be imposed. One way is to propagate the Community policies to all the sites and implement them in the local batch systems. This way is quite heavy and needs a lot of effort of the site managers especially for the sites serving multiple user communities. It is also very difficult to keep the policy definitions on the sites up to date in case of changes. Attempts to develop tools synchronizing Virtual Organization (VO) policies on all the participating sites were not successful so far.

In the other approach all the VO policies can be applied in a single place – the central Task Queue. Of course, in this case it is much easier to maintain the policies by the VO managers taking away this

load from the site managers. Due to a late job to resource binding the highest priority job from the Task Queue is given to the Pilot Job request for the immediate execution. This removes uncertainties due to unpredictable times that a job can spent waiting in the local batch system and makes the policies application precise.

However, this way of VO policies management necessitates the use of MultiUser Pilot Jobs. This means that the Pilot Jobs are submitted with special credentials allowing them to pick up the payload of any user in the community. Indeed, if it has to take the highest priority job, this job can belong to any user not necessarily the one who had the highest priority at the moment of the Pilot Job submission. Running MultiUser Pilot Jobs needs a special attention to the treatment of the security issues. In particular, the actual user payload ownership evaluation and a consequent authorization is required by some sites. This needs the use of the glExec facility coupled with the SCAS site Authorization Service [5]. DIRAC is fully capable to use this facility [6]. The glExec tests are progressively done now involving more and more sites.

4.2. Spanning various grids

Another advantage of the WMS systems with Pilot Jobs is the easiness of incorporating various computing resources different in their nature and belonging to different administrative domains and available to a given user community.

In the case of the LHCb Collaboration, most of the available computing resources were clusters in the participating institutes. Geographically distributed these clusters were operating under a number of different Batch Systems (PBS/Torque, LSF, SGE, BQS). With more and more sites joining the WLCG grid, there are still few institutes outside any grid system still willing to contribute to the LHCb computing. There are also examples of user communities interested in building up combined distributed systems on the level of a region or university campus. In this case, there is a need to incorporate already existing clusters, which were not set up from the very beginning with a future grid participation in mind. For such cases, building local grids or joining larger grids using standard grid middleware poses significant restrictions on the choice of computing cluster software and can present a significant threshold for local site managers. The solution based on WMS with Pilot Jobs can be a lighter alternative easier to deploy and manage.

Most of the LHCb resources are available through the WLCG grid. However, due to the participation in the EELA grid project [7], the LHCb Collaboration gained access to some sites in Latin America. Incorporating those sites into the DIRAC WMS was very easy and was reduced to setting one more Pilot Director sending Pilot Job to the EELA sites. Since the EELA grid is constructed using the same middleware as the EGEE/WLCG grid, it was only necessary to configure the new Pilot Director to access EELA facilities without any new software development.

Another example is incorporation of NDG sites into the DIRAC WMS. The NDG grid is using different middleware and the EGEE Pilot Director could not be reused [8]. Therefore, a new Pilot Director was developed. However, it was as simple as submitting all the same Pilot Jobs to the available NDG sites. The Pilot Job scripts almost did not need to be modified. The LHCb Collaboration does not have officially resources in the NDG grid. Therefore, the exercise was only limited to few demonstration jobs executing the LHCb standard application workflows.

In all the cases, the new resources were incorporated seamlessly into the DIRAC WMS. The production managers just see new sites appearing in the job monitoring systems. No special support from the DIRAC site, EELA or NDG managers was necessary.

5. Failover mechanisms

The widely distributed computing systems like grids with a large number of interdependent services, many people with varying level of skills responsible for the system operation and evolving non-stable middleware are intrinsically not stable and should be treated as such. Therefore, a lot of effort was spent in the DIRAC project to cope with various kinds of failures that can occur while using the grid resources. There are several ways how the issue can be addressed.

First, the crucial DIRAC services are implemented in a highly redundant manner with multiple mirrors that can be accessed by the clients in a random round-robin order. This provides also a certain level of load balancing. The DIRAC Configuration Service and LHCb instances of the LCG File Catalog (LFC) Service are organized in this way [2].

Second, there are many operations that can fail and the execution environment does not allow repeating them until success. For example, file uploads to a misbehaving Storage Element can be retried in a running job because this will be blocking a CPU slot leading to a lower efficiency. However, many of such operations can be retried later asynchronously if they are properly registered with all the needed information for the retrial. In DIRAC this is done by means of the Request Management System.

5.1. Request Management System

The Request Management System (RMS) is a specialized database with a service interface, which allows to collect and serve requests for various operations. The RMS itself is organized as a distributed redundant set of services. An instance of the RMS system is running at each DIRAC VO-box host at all the LHCb Tier-1 centres as shown in Figure 2. This guarantees that there is a functioning instance of the RMS service at any moment. The RMS service instances are receiving requests from the clients, mostly from the jobs running in the grid. There are two main kinds of the requests:

- Requests for Data Management Operations, for example, data replication or registration in a file catalog;
- DISET requests which can encapsulate any client/service communication in the DIRAC framework.

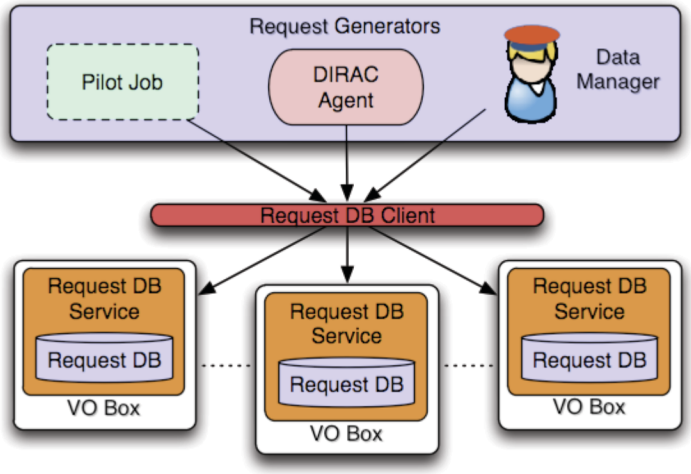

Figure 2 DIRAC Request Management System

The requests are executed by dedicated agents running as part of the RMS service at a Tier-1 site VO-box or can be forwarded to the Central RMS running at a VO-box at CERN Tier-0 site. Accumulating requests in the central service allows better monitoring of the pending operations. It also helps to increase the requests execution efficiency as they can be grouped for some bulk operations, for example, moving sets of files using the File Transfer Service (FTS) .

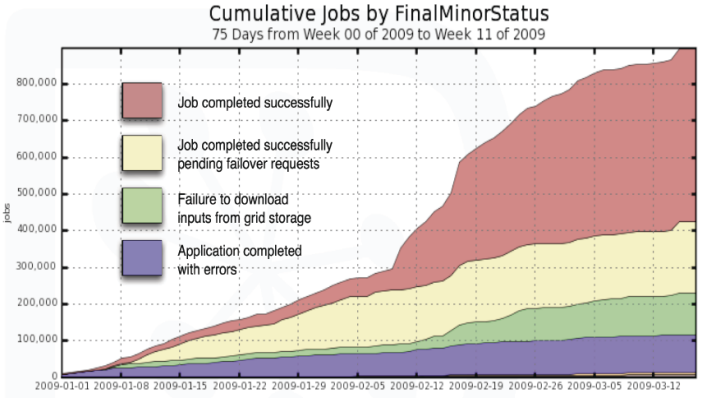

Figure 3 Job completion efficiency. About 20% of jobs finish with failover requests

Figure 3 demonstrates the effect of using the RMS service for the data production jobs. About 20% of them finish with some pending operations. Typically, the destination Storage Element was unavailable at the moment when a job was uploading the results of its work. In this case, the files are stored in some intermediate Storage Element with a request set to move data to the final destination. Without the RMS system these 20% of jobs would be lost making the overall efficiency of the LHCb Data Production System much lower.

5.2. Data Management System

It should be noted that the RMS system can be used as a generic Task Queue mechanism. In particular, in the DIRAC project the Data Management System is built on top of an RMS service instance as shown in Figure 4. The Data Transfer requests coming from various sources are collected in a RMS Task Queue. The requests may come from running jobs defined to output results to several destinations, from jobs failed to store data to their destination and requesting to move data from temporary storage, or from LHCb Data Managers defining large data distribution operations. The requests in the RMS Task Queue are examined by dedicated agents, grouped according to their transfer channels and passed to the WLCG FTS service for the execution. The data transfer request execution status is properly updated. If requests are coming from jobs in the DIRAC WMS system, the status of the jobs is also updated accordingly.

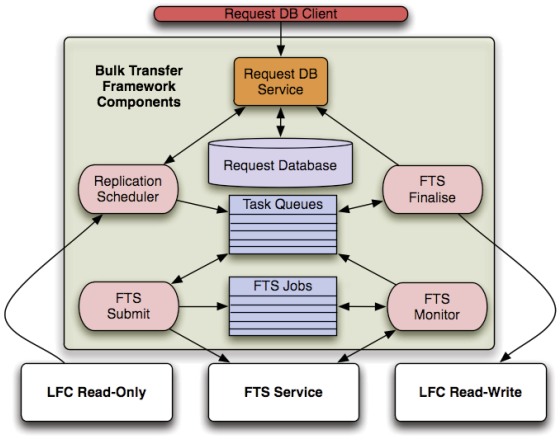

Figure 4 Data Management System based on the Request Database

6. Web Portal

With more and more users using DIRAC tools to run their analysis or data production jobs, it became clear the necessity to provide adequate User Interfaces to facilitate all the activities. DIRAC project provides many command line tools as well as a comprehensive Python API to be used in a scripting environment. However, in DIRAC3 emphasis was put on providing a full-featured graphical Web Portal [9].

The DIRAC Web Portal is a web application, which allows steering of all the LHCb grid activities. It meets the needs of all the LHCb user groups, from DIRAC administrators to the physicist users providing multiple views to monitor the ongoing work. Note that it also allows interaction with the system like editing its configuration and controlling its components. A non-comprehensive list of features includes:

- Browsing and editing the DIRAC Configuration, monitoring the state of the system components;
- Monitoring the LHCb computing resources: sites, data storages, etc;
- Detailed monitoring and control of the user and pilot jobs;
- Steering data production and data distribution activities;
- Accessing various Accounting and Monitoring data in a graphical form.

The Web Portal interface is constructed using common paradigms found in the desktop applications. This provides rich controls together with a familiar “look and feel” to the DIRAC users, which is important to make it easy to learn and use. The interfaces are implemented with modern web programming techniques including Ajax technologies and advanced Javascript libraries.

Since the Web Portal provides access to some sensitive information and allows changing the system state, much attention is paid to the security aspects. The users accessing the portal pages are authenticated using their grid certificates loaded in their browsers. Once authenticated, the user rights to access various services are managed by the general DIRAC authorization rules.

The DIRAC Web Portal is being actively developed to access more information and to provide more complete functionality. We are experimenting with more advanced interfaces like “Site Map” based on a geographical presentation of the state of the resources and ongoing activities. It is foreseen to allow customization of the portal functionality based on individual user preferences.

7. Other DIRAC services

DIRAC project includes a number of other services to support various activities. It is important that all the services are built based on the same Framework, which allows sharing the experience and development patterns across the whole team of developers. In particular, the high-level Production Management and Data Management Systems are constructed using common tools whereas in other LHC experiments these components are often developed by distinct development teams. Some of the services new or considerably updated in the DIRAC3 software are outlined in the subsequent subsections.

7.1. System Logging Service

The System Logging Service (SLS) receives important error messages from all the DIRAC components, classifies and stores them in the internal catalogue. A series of specialized agents are analysing the errors and take appropriate actions, for example, sending alarms to responsible system managers. The SLS system is an essential part of the monitoring tools allowing spotting failures of any DIRAC or third party services. It is especially important in highly distributed systems as the one of the LHCb experiment providing production managers and shifters a single entry point to the system status information [reference to monitoring paper].

7.2. Bookkeeping Database Service

The Bookkeeping Database Service (BDS) collects and serves the provenance information about all the data files produced by the LHCb experiment [10]. It contains all the necessary metadata to allow users selection of their preferred data sets. The service was developed originally outside the DIRAC project and there were a lot of problems with the service maintenance and operations as the expertise was going with the departure of the developers. Once the service was ported into the DIRAC Framework, the operational problems disappeared because the service controls became similar to the other DIRAC services.

The BDS is using Oracle database backend unlike other DIRAC services using the MySQL databases. However, since there is a software layer isolating the particular database technology from the service logic, it was easy to introduce a new one. This also opens an opportunity to use Oracle databases for other DIRAC services if it will be preferable for certain new installations.

7.3. Production Management System

The Production Management System (PMS) functionality consists in a definition and steering of large numbers of jobs grouped in so-called productions as described in [2,11]. It automates job submission as soon as the corresponding input data to be processed are available from the LHCb detector Data Acquisition System or from the previous stages of the data processing cycle. The PMS was considerably reworked in the new DIRAC3 Project:

- A completely new implementation of the Workflow package, which allows description of workflows of arbitrary complexity, built from basic Modules. The modules can exchange data of all the Python allowed types. The Module interface description allows for an easy construction of complex workflows with multiple connections between various Steps. A new Web based Workflow Editor facilitates definition and storing of the LHCb production workflows.
- The Web based Production Request Editor and Production Monitor allow to perform all the production related operations using a comprehensive graphical user interface.
- The Production Database and associated services and agents form the core of the PMS as shown in Figure 6. It interacts with many other DIRAC services (File Catalogues, Bookkeeping Database, WMS, etc) organizing their collective work to accomplish data production tasks:
    - Bookkeeping Agent discovers data files to be processed and adds them to the internal production queue;

- Transformation Agent forms production jobs by associating predefined chains of applications to the data to be processed and assigns the jobs to particular Sites according to the LHCb Computing Model;
- Job Submission Agent submits jobs to the DIRAC WMS;
- Production State Agent monitors the production progress and updates production data to be available through command line and Web interfaces.

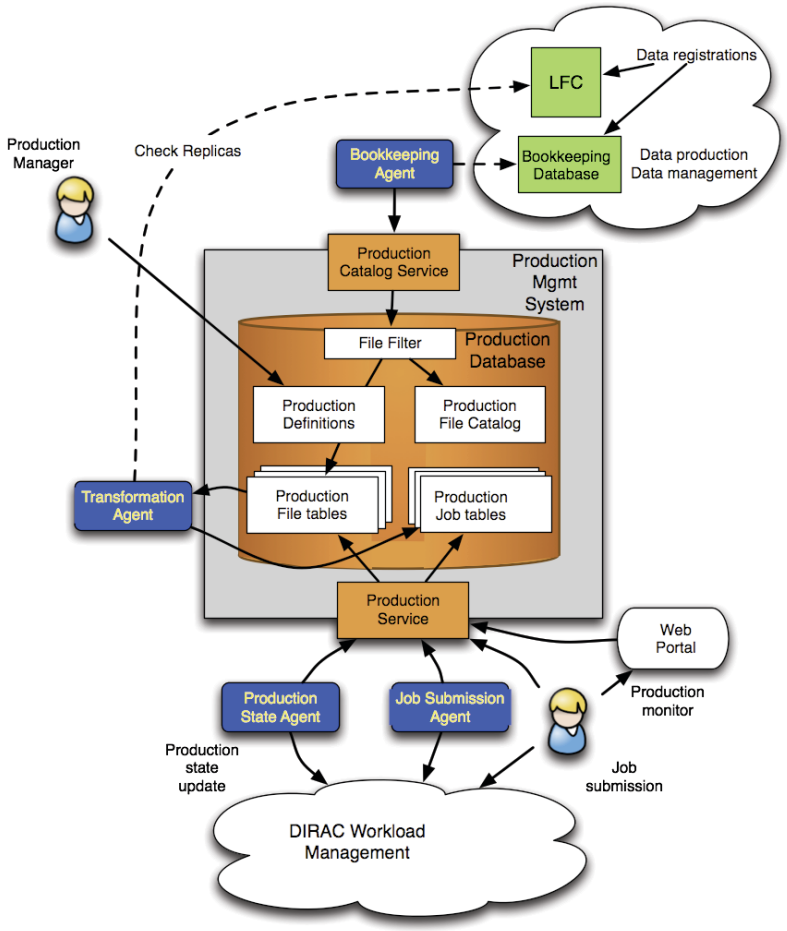

Figure 5 Outline of the LHCb Production Management System based on the DIRAC Framework

8. Running experience

The LHCb Data Production System based on the DIRAC software is now in a constant use to generate the detector simulation data, which is analyzed by the LHCb users. It was also used recently in the CCRC and FEST'09 test runs in parallel with the other LHC experiments. Detailed evaluation of the results of these runs can be found in [6].

In general, the system demonstrated a very stable behaviour and excellent scalability properties. The DIRAC WMS was demonstrated to scale up to 15K simultaneously running jobs with several tens of thousands of jobs waiting in the Task Queue and more than 100K jobs in a final state ready for the results retrieval as demonstrated in Figure 6. These levels were achieved with most of the DIRAC WMS services running on a single mid-range host and were limited by the hardware capacity of the machine. The future migration to a more powerful hardware together with wider distribution of the DIRAC services over a number of servers will help to further increase the overall capacity of the system.

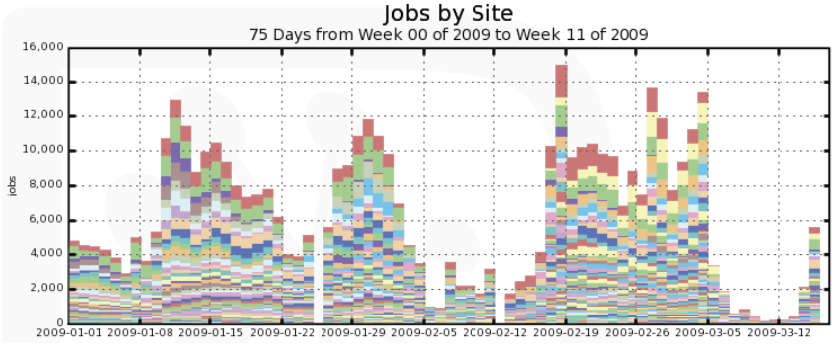

Figure 6 Snapshot of the number of Running Jobs controlled by the DIRAC WMS

The main encountered problems were related to access to data stored in various Storage Elements across the Grid. The most complicated activity is the Data Processing with large number of files brought on-line from the tape storage. The capability of the DIRAC WMS to prestage those files before actually sending the jobs for execution turned out to be essential to increase the job efficiency by lowering the number of failures during the input data resolution and by improving the CPU to Wall Clock time ratio of the jobs.

9. Conclusions and outlook

After several years of development and operation, the DIRAC Project proved to be a reliable, efficient and flexible tool to build distributed computing systems. It was extensively tested in a long series of Data Challenges to achieve the maturity level necessary for the start of the real data taking by the LHCb experiment. The new generation of the software is leveraging the strong points of the DIRAC architecture and is now built using solid programming techniques, which allowed to achieve high scalability levels. It has evolved from an LHCb specific product to a general-purpose project suitable for other user communities.

The further DIRAC developments are aiming at various optimizations to increase the system capacity and reliability as well as to improve the system usability by perfecting user interfaces including the Web Portal. The latter will become the main user entry point to all the DIRAC systems. Since the system is attracting interest of other user communities than LHCb, more work will be necessary to adapt their application for running on the grid using DIRAC community solutions. For example, many application domains outside the High Energy Physics are exploiting massively parallel computations using MPI interprocess communication protocols. WMS with Pilot Jobs provided by the DIRAC project offers very interesting opportunities for this class of applications since it can provide advanced resource reservation procedures necessary to run multiple parallel tasks. The MPI job support is now under the development.

Another challenging goal is to include the LHCb Online Computing Farm into the pool of available off-line computing resources. This cluster consists of 16 thousands computing cores used for the High Level Trigger (HLT) selecting in real time interesting physics events registered by the LHCb detector. However, outside the LHC accelerator runs this computing facility is idle. The work is now in progress to include this resource seamlessly into the DIRAC Data Production System with a possibility to switch from on-line to off-line mode of operation and back in a matter of minutes. This will potentially increase the computing resources available to LHCb by 40-50% [12].

10. Acknowledgements

Members of the DIRAC Project team would like to thank the LHCb site managers for their patience and excellent cooperative work that allows executing the ever-growing LHCb user payloads on the Grid.

References

[1] LHCb Computing TDR, CERN-LHCC-2005-019
[2] A.Tsaregorodtsev et al, DIRAC, The LHCb Data Production and Distributed Analysis System, Proceedings of the CHEP 2006 Conference
A.Tsaregorodtsev et al, DIRAC: Community Grid Solution, Proceedings of the CHEP 2007 Conference
[3] R.Graciani Diaz, A.Casajus Ramo, DIRAC Secure Distributed Platform, Proceedings of the CHEP 2009 Conference
[4] R.Graciani Diaz, A.Casajus Ramo, A.Tsaregorodtsev, Pilot Framework and the DIRAC WMS, Proceedings of the CHEP 2009 Conference
[5] D.Groep et al, glExec - gluing grid computing jobs to the Unix world, Proceedings of the CHEP 2007 Conference
[6] J.Closier, S.K.Paterson, Performance of Combined Production And Analysis WMS in DIRAC, Proceedings of the CHEP 2009 Conference
[7] EELA-2 Grid, http://www.eu-eela.eu
[8] NorduGrid, http://www.nordugrid.org
[9] A.Casajus Ramo, M.Sapunov, DIRAC Secure Web User Interface, Proceedings of the CHEP 2009 Conference
[10] Z.Mathe, E.Lanciotti, The LHCb Data Bookkeeping System, Proceedings of the CHEP 2009 Conference
[11] S.Paterson, A.Tsaregorodtsev, A.Zhelezov, Managing Large Data Productions in LHCb, Proceedings of the CHEP 2009 Conference
[12] M.Frank, A. Puig Navarro, Event reconstruction in the LHCb Online cluster, Proceedings of the CHEP 2009 Conference