DIRAC Data Management Framework

A. Tsaregorodtsev[1], on behalf of the DIRAC Project
Aix Marseille Université, CNRS/IN2P3, CPPM UMR 7346
163, avenue de Luminy, 13288 Marseille, France
Plekhanov Russian University of Economics
36, Stremyanny per., 117997 Moscow, Russia
E-mail: atsareg@in2p3.fr

DIRAC Project is developing software for building distributed computing systems for the needs of research communities. It provides a complete solution covering both Workload Management and Data Management tasks of accessing computing and storage resources. The Data Management subsystem of DIRAC includes all the necessary components to organize data in distributed storage systems. The DIRAC File Catalog (DFC) service keeps track of data file physical replicas. This service is a central component of a logical File System of DIRAC presenting all the distributed storage elements as a single entity for the users with transparent access to the data. The DFC service provides also a Metadata Catalog functionality to classify data with user defined tags. This can be used for an efficient search of the data necessary for a particular analysis. The Data Management system provides also support for usual data management tasks of uploading/downloading, replication, removal of data with a special emphasis on the bulk data operations involving large numbers of files. Automation of data operations driven by new data registrations is also possible. In this article we will make an overview of the DIRAC Data Management System and will give examples of its usage by several research communities.

International Symposium on Grids and Clouds 2016
13-18 March 2016
Academia Sinica, Taipei, Taiwan

[1]Speaker

1. Introduction

The number of scientific domains with very large volume datasets collected and analyzed is rapidly increasing. The High Energy Physics (HEP) experiments at the LHC collider at CERN were the first to enter the new era of highly data intensive studies; their cumulative data volume is approaching now the Exabyte scale. However, other disciplines are quickly increasing their data volume requirements. New scientific communities need urgently tools to work with large datasets adapted to their specific tasks and suitable to the computing expertise level of their scientists. The scientific collaborations nowadays are often international with many groups coming from different laboratories and universities. As a result, the available computing and storage resources of a given collaboration are usually distributed as each group is coming up with its own contribution. Therefore, there is a strong necessity in building computing systems that cope with large volumes of distributed data and distributed computing resources that can be used for these data analysis.

The DIRAC Project was started to solve the data intensive analysis problem for one of the LHC experiments, LHCb, in 2003 [1,2]. It was started as a Workload Management System (WMS) in order to operate multiple computing centers in Europe to produce modeling data for the experiment optimization [3]. However, the need in an efficient Data Management System coping with many millions of files with distributed replicas and having a close coupling with the WMS was quickly understood. Now the DMS component is an important part of the DIRAC software stack and provides not only multiple tools to store and access data in various storage systems but also support for reliable massive data operations. As a result, the DMS allows to perform all the data management tasks of LHCb and other HEP experiments [4,5,6] basing their computing systems on DIRAC.

After multiple years of successful usage in the HEP domain, the DIRAC software was generalized to be suitable for other applications requiring large data volumes and computing power. It provides a development framework and many ready-to-use services to build distributed computing systems adapted to particular scientific communities. These tools serve to interconnect technologically heterogeneous computing and storage resources into a coherent system seen by the users as a single large computer with friendly interface and consistent computational and storage subsystems. Therefore, we speak about DIRAC interware – technology to aggregate multiple computing resources and services.

In this article we will describe the DMS framework of the DIRAC interware. We will start with the formulation of the problem that the DMS is solving, followed by the description of the software architecture. We will then present the main APIs of the system and services to perform basic operations. High-level services for reliable massive data operations will be outlined afterwards. Several examples of usage for different applications will be also given followed by conclusions.

2. The problem of the Data Management System

The scientific data can be collected in a variety of forms: data files stored in conventional file systems, records in databases of different types, data in special file systems supported by some particular analysis mechanism, e.g. HDFS/Hadoop, and others. In the HEP applications, the physical events data is stored in a form of files that can be downloaded and stored in local Unix file systems. DIRAC supports data models based on files. So, we are speaking about geographically distributed data partitioned in files with possibly multiple physical replicas for each of them. The main basic operations with this kind of data are:

- Initial file upload from some local storage where the data is created to a remotely accessible Storage Element (SE) service.
- Registration of the file in a File Catalog service together with its metadata and replica information.
- File replication to bring it closer to the analysis facilities, to make an archive copy or for a general redundancy.
- File access and/or download for processing.
- Checking the integrity of information stored in the File Catalog and the actual status of files in SEs.
- File removal including all the replicas and registration information.

In addition to the basic operations the DMS should ensure transparent access to data by the users while fully acknowledging their access rights. Since the volume of scientific data is quite high, the system must provide means to perform operations on large numbers of files in one go ensuring successful execution of all the elementary operations. Many operations with data in practice are recurrent and must be automated as much as possible to lessen the need of the human intervention.

3. DMS Software Architecture

The DIRAC DMS software stack is schematically presented in Figure 1. At the bottom of the stack there is a number of plug-ins for each external service technology. Those plug-ins are implementing an abstract interface defined by DIRAC for various kinds of services, currently Storage Element and Catalog types of services. New service types are likely to appear in the future, for example a File Transfer service, for which the abstract interface is not yet defined.

The next level is plug-in aggregators. DIRAC defines for the moment two kinds of aggregators: FileCatalog and StorageElement. Aggregators allow to work with multiple services as if with a single one from the client perspective. For example, an SE service can have different access endpoints with different access protocols. In this case the SE aggregator will contain instantiated plug-ins for all the configured protocols and will choose the most suitable one depending on the execution environment. Examples of aggregation of different File Catalog services will be given below (Section 5.2, Section 7).

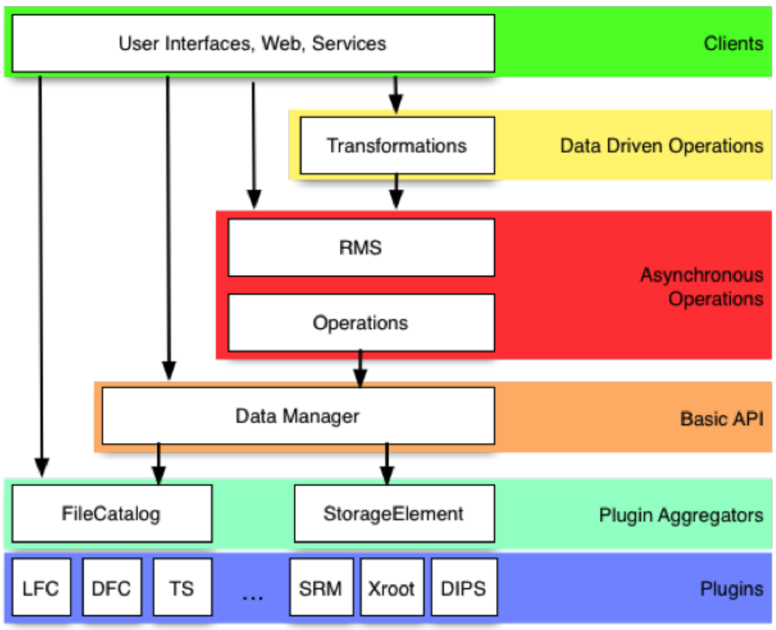


Figure 1: DIRAC Data Management software architecture

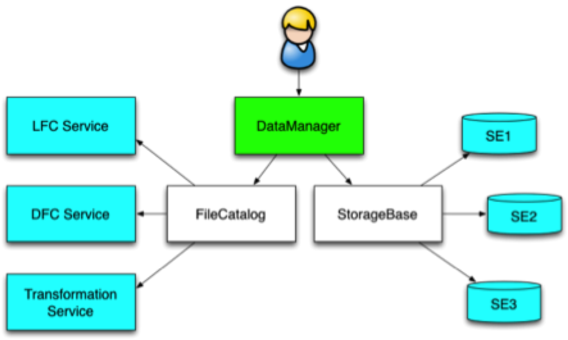


Figure 2: DataManager API using aggregated Storage and Catalog plug-ins

All the plug-ins and aggregators are hidden behind the DataManager API which have methods to perform all the basic operations that need access to both Storage and Catalog services. This API internally instantiates the necessary plug-ins given the logical names of the SEs and Catalogs as defined in the DIRAC Configuration System. Therefore, the users are only seeing logical entities without the need to know the exact type and technology of the external services (Figure 2).

The DataManager API is used to execute basic operations on data. However, bulk data operations need special support so that they can be performed asynchronously with no need for a user to wait for the operation completion at the interactive prompt. The support for asynchronous operations is provided by the Request Management System (RMS), which allows to define persistent Operations objects containing instructions on the requested file manipulations.

Finally, the Transformation System (TS) provides means to automate recurrent massive data operations driven by the data registration or status change events.

In addition to the main DMS software stack, DIRAC provides several more services helping to perform particular data management tasks:

- Staging service to manage bringing data on-line into a disk cache in the SEs with tertiary storage architecture. These operations are usually triggered automatically by the WMS before the jobs using these data as input can be submitted for execution to the worker nodes.
- FTS Manager service to submit and manage data transfer requests to an external File Transfer Service.
- Data Logging service to log all the operations on a predefined subset of data mostly for debugging purposes.
- Data Integrity service to record failures of the data management operations in order to spot malfunctioning components and resolve issues.
- The general DIRAC Accounting service is used to store historical data of all the data transfers, success rates of the transfer operations, etc.

4. Basic Data Management components

In this section we describe basic plug-ins used to access services external to the DIRAC software and provided by third parties, such as Storage Elements and Catalogs. Note that DIRAC has its own implementation of a Storage Element and Catalog services. These services are also accessed through the DataManager API with corresponding plug-ins implementing the common abstract interfaces.

4.1 Storage Resources

DIRAC provides plug-ins for a number of storage access protocols most commonly used in the distributed storage services:

- SRM;
- XRootD;
- RFIO;
- gfal2 library based access protocols (DCAP, HTTP-based protocols, S3, WebDAV, and others) [7].

If some DIRAC user community would need access to a storage system not yet supported by the DIRAC interware, it will be easily incorporated by providing a new plug-in to the system.

DIRAC provides its own implementation of a Storage Element service and the corresponding plug-in using the custom DIPS protocol. This is the protocol used to exchange data between the DIRAC components. The DIRAC StorageElement service allows exposing data stored on file servers with POSIX compliant file systems. This service helps to quickly incorporate data accumulated by scientific communities in any *ad hoc* way into a distributed system under the DIRAC interware control.

Another important service to work with distributed data is Storage Element Proxy. This service allows access to any Storage Element from machines not having the software needed for corresponding plug-ins. In this case, the client is accessing the Storage Element Proxy service with the DIPS protocol and the service transmits the access request to the destination SE with the suitable protocol. The client credentials are checked and used to access the destination service by delegation. An interesting example of the use of the Storage Element Proxy service is the case of running user jobs in computing centers where worker nodes do not have access to the WAN and therefore can not upload the resulting data directly. In this case, running the Storage Element Proxy service in the Gateway host of the computing center can help to export data from the worker nodes without a need to use some intermediate buffer storage and transfer data asynchronously by some additional agent or cron job (Figure 3).

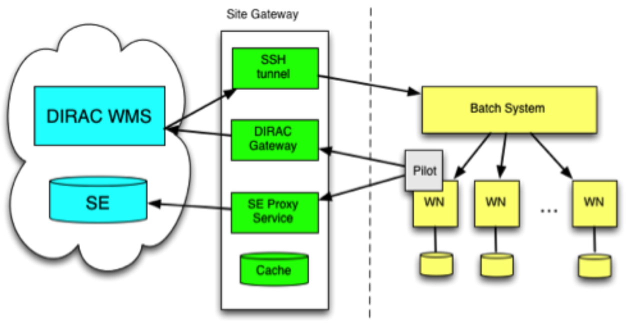


Figure 3: Storage Element Proxy service used to export output data in an HPC center without WAN access from the worker nodes

4.2 Catalogs

Similarly to Storage Elements, DIRAC provides access to file catalogs via client plug-ins (Figure 2). The only plug-in to an external catalog service is for the LCG File Catalog (LFC), which used to be a *de facto* standard catalog in the WLCG and other grid infrastructures. Other available catalog plug-ins are used to access the DIRAC File Catalog (DFC) service and other services all written within the DIRAC framework with a catalog-like behavior or "pseudo catalogs". Those plug-ins are implementing an abstract DIRAC File Catalog interface and can be aggregated together so that all the connected catalogs are receiving the same messages. The usefulness of aggregating several catalogs can be illustrated by an example of a user community that wants to migrate the contents of their LFC catalog to the DFC catalog. In this case, the catalog data can be present in both catalogs for the time of migration or for redundancy purpose. Another example is the use of pseudo catalogs like the Transformation Manager service (see Section 5.2 below).

DIRAC interware provides a complete File Catalog service, DFC, which is described in details in [8]. Here we present a list of the main DFC features:

- Standard file catalog functionality allowing to store file metadata, ACL information, check sums, etc.
- Complete Replica Catalog functionality to keep track of physical copies of the files.

- Additional file metadata to define ancestor-descendent relations between files often needed for applications with complex workflows.
- Efficient storage usage reports to allow implementation of community policies, user quotas, etc.
- Metadata Catalog functionality to define arbitrary user metadata on directory and file levels with efficient search engine.
- Support for dataset definition and operations.

The DFC implementation is optimized for efficient bulk queries where the information for large numbers of files is requested in case of massive data management operations. Altogether, the DFC provides logical name space for the data and, together with storage access plug-ins, makes data access as simple as in a distributed file system.

5. Massive Data Management Operations

The massive data management operations are performed by a collaborative work of several DIRAC subsystems. These subsystems as well as their interactions are described in the following.

5.1 Request Management System

The reliable massive data operations require special systems for asynchronous execution of each basic task. The reliability is achieved by creating a bunch of requests for elementary operations. These requests are stored persistently in a special database. Once requests are created, they are executed one by one by dedicated agents with possible retrial of operations failed due to temporary unavailability of the target service. In DIRAC this mechanism is realized in the Request Management System (RMS). The RMS was introduced in order to cope with any failed operation in an unstable grid environment providing a reliable failover mechanism. The operations can be any failed RPC call to any of the DIRAC services, but it can be also any data management operation like data upload and registration in the catalog. A typical use case of the RMS system is a failure of data upload in otherwise successful job due to a temporary unavailability of the destination SE. In this case the data will be temporarily stored in some dedicated buffer SE and a request to move the files to the final destination SE will be created. Once the destination SE service will come online, the data will be moved automatically to where it is defined by the logic of the job.

Clients are creating requests by communicating with the Request Manager service. Since the RMS requests are an important element of the failover mechanism, creating requests must be extremely reliable itself. Therefore, the central Request Manager service is complemented by an arbitrary number of geographically distributed Request Proxy services that can accept and temporarily store requests if the master Request Manager service is not responding or is overloaded.

The RMS requests are complex objects themselves (Figure 4). Each request contains an ordered list of operations. Each operation can have a list of associated files. While processing one request, its operations are executed one by one in the specified order. The next operation is selected for execution only if the previous operation was executed completely and successfully;

if there are associated files, the operation must be completed for all of them. The order in which operations are executed is important. For example, moving a file from Storage A to Storage B can be accomplished by two subsequent operations - replication from A to B and removal of the replica at A – but only in this order. The request is completed when all the contained operations are successfully executed.

Figure 4: RMS Request structure

Each request contains identity of the Owner – the user who has created the request or on whose behalf the request was created by the system. Special RMS agents fetch requests from the database and execute them with the credentials of the request's Owner. The Owner credentials are retrieved from the DIRAC Proxy Manager service, which has similar functionality as the MyProxy service in the standard grid middleware. For the massive data operations these are usually credentials of a Data Manager of the user community that owns the data.

Multiple request execution agents can run in parallel to increase the operational capacity of the RMS system. A special executor software module is defined for each Operation type. Agents are invoking a relevant executor for each Operation according to its type. The system can be easily extended to incorporate new types of Operations by adding new executor modules. This is yet another example of the plug-in mechanism widely used in the DIRAC software.

5.2 Transformation System

Each operation applied to a large dataset can be considered as a transformation of the initial dataset into some new data. The Transformation System (TS) is implementing this concept. It allows to create, store and launch Transformations. Each Transformation object definition contains the following elements:

- template for actions to be applied to each file of a given dataset;
- dataset to which the Transformation will be applied;

The Transformation dataset can be defined as files selected by a given query to the File Catalog. When a Transformation is created the corresponding files are selected and their references are stored in the TS database.

It is important to note that newly registered files corresponding to the Transformation definition will be automatically added to the Transformation. This is achieved by using the catalog aggregation feature of the DIRAC FileCatalog component. The Transformation Manager service is implementing the File Catalog interface and can be added to the FileCatalog aggregator object. In this case the Transformation Manager service will receive all the same messages as any other File Catalog in the system. This mechanism allows automatic addition of new data to existing Transformations making it a full-fledged data driven workflow engine (Figure 5).

Figure 5: Transformations combined together to form complex workflows

Actions performed by Transformations can be one of the following:

- jobs submitted to the DIRAC WMS with the input data as subsets or individual files from the Transformation dataset;
- RMS requests to perform operation on the Transformation dataset, for example, file replication or removal of temporary data.

Transformations can define how to group files together for each individual job or RMS request, define destination SE for the data replication or particular catalog for the new data registration. Standard recipes to create jobs and RMS requests can be enhanced by plug-ins specific for a particular user community. For example, the plug-ins can realize complex data placement scenarios taking into account different storage capacities and community policies. Transformation's output data can be used as input in a definition of yet another Transformation. Altogether this allows automatic chaining of the Transformations thus creating workflows of arbitrary complexity.

5.3 Putting this all together

The massive data operations in DIRAC are performed by a combined use of the RMS, TS as well as other auxiliary DIRAC systems (Figure 6). A typical scenario for a one-time massive data operation is the following. Each large user community nominates a Data Manager, the person responsible for operations with the data belonging to the community as a whole. The Data Manager is creating one or more RMS requests for bulk data operations containing multiple files. The files are identified by their LFNs as registered in the File Catalog. Typical request operation types are replication to a predefined SE, removal from one or multiple SEs, complete removal, etc.

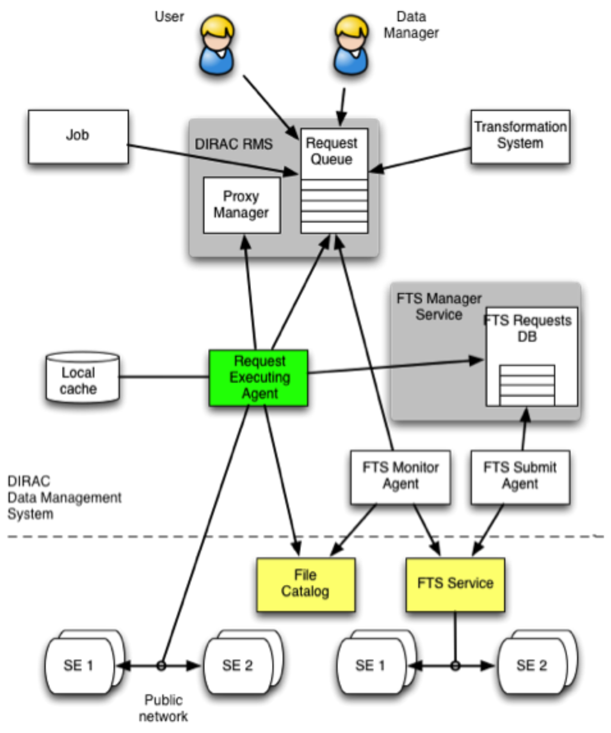


Figure 6: Outline of the DIRAC services for massive data management operations

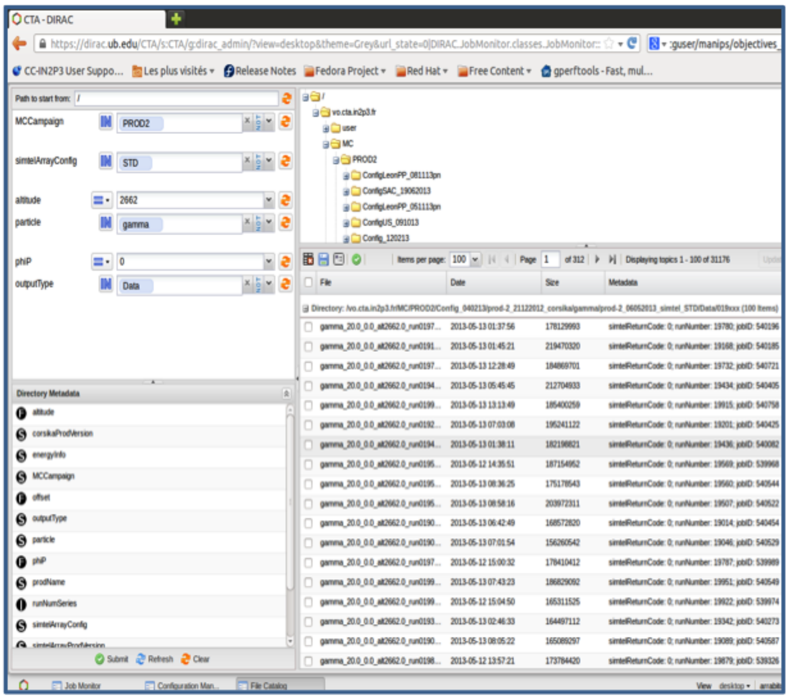
The Request Executing Agent picks up the so-defined RMS requests. The Agent can be configured to perform replication or removal operations in two different ways. First, it can access the Storage Elements directly using the DataManager API described above. In this case the third party transfers between the source and destination SEs are performed where possible, otherwise the transfers are done via a local disk cache. Second, an external file transfer service can be used. Currently, the FTS service [9] can be configured to do the transfers. For that DIRAC provides a special FTS Manager service to keep track of the corresponding FTS requests. Two agents complement the set of the FTS components, FTS Submit Agent and FTS Monitor Agent, to correspondingly submit and monitor requests in the external FTS service. The FTS Monitor Agent also registers the new replicas to the File Catalog once the request is successfully executed. As soon as the FTS request is finished, the status of the corresponding RMS Operation is set to Done and the RMS Request can proceed further.

Another scenario concerns recurrent data-driven massive operations with the data. In this case, a special Transformation is defined to apply certain operations on the registered data corresponding to certain criteria, for example, user defined metadata. Then as soon as suitable data is registered to the File Catalog, the Transformation System will trigger creation of the corresponding RMS request, for example to replicate the new files to one or more destination SEs. Once the RMS request is created, it is executed exactly like in the previous scenario. This mechanism allows to completely automate all the recurrent bulk data operations of a given user community.

6. Interfaces

The DIRAC DMS functionality is available to users via a number of interfaces. First of all, a number of command line tools were provided following the same style as the standard grid middleware. Recently, a new DIRAC extension, COMDIRAC, was added, which provides a set of easy-to-use intuitive commands resembling usual Unix command syntax to manipulate data with similar semantics with a letter *d* prepended to the command names, for example: *dls, dcd, dpwd, dfind, ddu,* etc. There are few non-Unix-like commands to perform data upload, download and replication: *dput, dget, drepl.* Altogether, a user working with the COMDIRAC command line interface has an impression of using two Unix shells simultaneously – one is the standard local Unix shell with the local file system, and the other one is a similar shell but in the logical name space provided by the DIRAC File Catalog.

Another interface, which is being rapidly developed now, is the DFC Web interface as part of the general DIRAC Web Portal. This interface provides a complete graphical file browser with some additional functionality. In particular, it provides forms to create queries using the DFC user metadata, to store the results of the queries for subsequent use, to look up file user metadata, etc. An example of the DFC Web interface as used by the CTA experiment is presented in Figure 7 [10].



https://dirac.ub.edu/CTA/s:CTA/g:dirac_admin/?view=desktop&theme=Grey&url_state=0|DIRAC.JobMonitor.classes.JobMonitor::

Path to start from:	/
MCCampaign	PROD2
simtelArrayConfig	STD
altitude	2662
particle	gamma
phiP	0
outputType	Data

```
/
  vo.cta.in2p3.fr
    user
    MC
      PROD2
        ConfigLeonPP_081113pn
        ConfigSAC_19062013
        ConfigLeonPP_051113pn
        ConfigUS_091013
        Config_120213
```

Items per page: 100 Page 1 of 312 Displaying topics 1 - 100 of 31176

Directory: /vo.cta.in2p3.fr/MC/PROD2/Config_040213/prod-2_21122012_corsika/gamma/prod-2_06052013_simtel_STD/Data/019xxx (100 Items)

File	Date	Size	Metadata
gamma_20.0_0.0_alt2662.0_run0197...	2013-05-13 01:37:56	178129993	simtelReturnCode: 0; runNumber: 19780; jobID: 540196
gamma_20.0_0.0_alt2662.0_run0191...	2013-05-13 01:45:21	219470320	simtelReturnCode: 0; runNumber: 19168; jobID: 540185
gamma_20.0_0.0_alt2662.0_run0197...	2013-05-13 12:28:49	184869701	simtelReturnCode: 0; runNumber: 19732; jobID: 540721
gamma_20.0_0.0_alt2662.0_run0194...	2013-05-13 05:45:45	212704933	simtelReturnCode: 0; runNumber: 19434; jobID: 540405
gamma_20.0_0.0_alt2662.0_run0199...	2013-05-13 13:13:49	185400259	simtelReturnCode: 0; runNumber: 19915; jobID: 540758
gamma_20.0_0.0_alt2662.0_run0192...	2013-05-13 07:03:08	195241122	simtelReturnCode: 0; runNumber: 19201; jobID: 540425
gamma_20.0_0.0_alt2662.0_run0194...	2013-05-13 01:38:11	182198821	simtelReturnCode: 0; runNumber: 19436; jobID: 540082
gamma_20.0_0.0_alt2662.0_run0195...	2013-05-12 14:35:51	187154952	simtelReturnCode: 0; runNumber: 19569; jobID: 539968
gamma_20.0_0.0_alt2662.0_run0195...	2013-05-13 08:36:25	175178543	simtelReturnCode: 0; runNumber: 19560; jobID: 540544
gamma_20.0_0.0_alt2662.0_run0195...	2013-05-13 08:58:16	203972311	simtelReturnCode: 0; runNumber: 19507; jobID: 540522
gamma_20.0_0.0_alt2662.0_run0190...	2013-05-13 06:42:49	168572820	simtelReturnCode: 0; runNumber: 19014; jobID: 540454
gamma_20.0_0.0_alt2662.0_run0190...	2013-05-13 07:01:54	156260542	simtelReturnCode: 0; runNumber: 19046; jobID: 540529
gamma_20.0_0.0_alt2662.0_run0197...	2013-05-12 15:00:32	178410412	simtelReturnCode: 0; runNumber: 19787; jobID: 539989
gamma_20.0_0.0_alt2662.0_run0199...	2013-05-13 07:43:23	186829092	simtelReturnCode: 0; runNumber: 19951; jobID: 540549
gamma_20.0_0.0_alt2662.0_run0199...	2013-05-12 15:04:50	165311525	simtelReturnCode: 0; runNumber: 19922; jobID: 539974
gamma_20.0_0.0_alt2662.0_run0193...	2013-05-13 02:46:33	164497112	simtelReturnCode: 0; runNumber: 19342; jobID: 540273
gamma_20.0_0.0_alt2662.0_run0190...	2013-05-13 08:05:22	165089297	simtelReturnCode: 0; runNumber: 19089; jobID: 540587
gamma_20.0_0.0_alt2662.0_run0198...	2013-05-12 13:57:21	173784420	simtelReturnCode: 0; runNumber: 19879; jobID: 539326

Figure 7: DIRAC File Catalog Web interface

7. Examples of Usage

Multiple scientific collaborations and user communities use the DIRAC interware either with dedicated DIRAC installations or with multi-community installations provided by several grid infrastructures.

The LHCb Collaboration is the most intensive user of DIRAC and its Data Management System in particular. Recently, LHCb undertook migration from the use of the LFC catalog service to DFC. The migration was quite smooth as both catalogs are supported by DIRAC with the same user interface. However, several modifications of the DFC functionality were necessary. For example, LHCb has provided a specific plug-in module to manage ACL rules for its users where several user groups can have similar rights to access certain directories. Now the LHCb DFC service counts about 20 million files in approximately 7 million directories, which constitutes about 40 PB data volume. As a result of migration from LFC to DFC, the response time to queries to the DFC service, especially for bulk queries, has been considerably reduced.

LHCb has a custom Metadata service, called Bookkeeping service, which is written in the DIRAC framework but is not using the Metadata capabilities of the DFC. However, the Bookkeeping service is implementing the File Catalog interface and is aggregated with the DFC as one of the pseudo catalogs as described above. LHCb also uses intensively the Transformation System for bulk data operations and has provided several custom plug-ins to generate data management requests.

Several other large collaborations in HEP and Astrophysics domains (ILC [11], BES III [6], CTA [10]) use DFC intensively as both Replica and Metadata catalogs. The ILC Collaboration was the first to adopt DFC for their data management system. An advantage with respect to the LFC service, the main alternative at this time, was the possibility to define arbitrary user metadata, which can be used to classify and tag the data for efficient searches. The simple provenance information that is supported by DFC where one can define ancestor-descendant relations is also used heavily in the ILC data management. The DFC capacity to generate efficiently storage usage reports is exploited by ILC and other communities for monitoring the occupancy of the storage systems.

One of the requirements of the BES III experiment was the possibility to define and manipulate datasets as collections of specially selected files. The dataset support following the BES III specification was added to the DFC functionality. The goal is to be able to manipulate large collections of files as easily as a single file with respect to common operations like data moving, access and removal. Datasets can be registered in the same name space as ordinary files with associated ownership and access rights information.

The CTA Collaboration data production system is based on the DIRAC Transformation System (TS) described above. In particular, the complex workflows of CTA defined as chains of data Transformations include also bulk automatically triggered data operations like data replication or moving from one location to another. Ones defined, these workflows save a lot of time of data managers freeing them from multiple routine data operations.

The DFC and other DMS services are a standard part of multi-community DIRAC installations in France, UK, and other countries as well as the DIRAC4EGI service operated by the EGI infrastructure in Europe [12]. New scientific communities evaluating functionality of distributed computing infrastructures pay a special attention to the availability of tools to manage their data. For example, the EISCAT-3D Collaboration [13] is constructing its data management system based on the EGI computing infrastructure using the DIRAC File Catalog and other data manipulation tools. In particular, the DFC Web interface was complemented by new utilities, e.g data downloading or job submission for selected data. Some of these new developments will be incorporated into the core DIRAC libraries and will be available for all the users.

8. Conclusions

The DIRAC interware provides a framework and a rich set of services to build distributed computing systems. The Data Management part of the interware provides a complete functionality to work with file-based distributed scientific data. It provides tools to perform basic data management operations with intuitive easy-to–use interfaces. The support for massive data management operations and automation of recurrent tasks makes the DIRAC DMS a powerful and versatile tool for data managers of large scientific collaborations. Many large experiments use the DMS to manage their data and help to further develop the software by providing new plug-ins and formulating specifications for the new desired functionality.

It is important to stress that the DIRAC DMS is built in the same framework as other DIRAC components, such as WMS, which allows easy communications between different systems with common protocols and data formats. This has many advantages compared to other frameworks which are dealing with data or workload management tasks independently: it allows WMS to use efficiently the information about files and dataset locations in taking job scheduling decisions; some jobs may require special data preparation, e.g. bringing online from a tape storage or extra data placement, that can be triggered automatically by the WMS components; other usages of the DMS components can be done by community specific plug-ins in the WMS job scheduler.

Since access to physical storage is done using plug-ins specific for each storage type and data transfer service, the DIRAC DMS can be easily adapted to the new currently available and future technologies. Finally, having within the same framework solutions for both data and workload management tasks of a large user community makes it much easier to deploy and maintain all the necessary services, which can be done by a small team of managers. Altogether, this makes DIRAC an attractive choice for large scientific communities that need to build and operate distributed computing systems for managing very large volumes of data.

References

[1] Tsaregorodtsev A et al 2008 DIRAC: a community grid solution *Journal of Physics: Conference Series* **119** 062048

[2] DIRAC Project - http://diracgrid.org

[3] Casajus Ramo A, Graciani Dias R, Paterson S, Tsaregorodtsev A 2010 DIRAC pilot framework and the DIRAC Workload Management System *Journal of Physics: Conference Series* **219** 062049

[4] Stagni F, Charpentier P 2012 The LHCb DIRAC-based production and data management operations systems *Journal of Physics: Conference Series* **368** 012010

[5] Grzymkowski R, Hara T 2015 Belle II public and private cloud management in VMDIRAC system *Journal of Physics: Conference Series* **664** 022021;

[6] Lin T, Zhang X M , Li W D and Deng Z Y 2014 The High-Level Dataset-based Data Transfer System in BESDIRAC *Journal of Physics: Conference Series* **513** 032059

[7] Gfal2 Project - https://dmc.web.cern.ch/projects-tags/gfal-2

[8] Poss S and Tsaregorodtsev A 2012 DIRAC File Replica and Metadata Catalog *Journal of Physics: Conference Series* **396** 032108

[9] FTS project - http://egee-jra1-dm.web.cern.ch/egee-jra1-dm/FTS

[10] Arrabito L et al 2015 Prototype of a production system for Cherenkov Telescope Array with DIRAC *Journal of Physics: Conference Series* **664** 032001

[11] Grefe C, Poss S, Sailer A, Tsaregorodtsev A 2014 ILCDIRAC, a DIRAC extension for the Linear Collider community *Journal of Physics: Conference Series* **513** 032077

[12] Tsaregorodtsev A 2014 DIRAC Distributed Computing Services *Journal of Physics: Conference Series **513** 032096*

[13] EISCAT-3D EGI Competence center: https://wiki.egi.eu/wiki/Competence_centre_EISCAT_3D