DIRAC Framework for Distributed Computing

A Casajús Ramo and R Graciani Díaz

Departament d'Estructura i Constituents de la Matèria. Facultat de Física i Química. Universitat de Barcelona, Barcelona, Spain

E-mail: adria@ecm.ub.es

Abstract. The DIRAC system is made of a number of cooperating Services and Agents that interact between them with a Client-Server architecture. All DIRAC components rely on a low level framework that provides the necessary basic functionality. In the current version of DIRAC these components have been identified as: DISET, the secure communication protocol for remote procedure call and file transfer; Configuration System, providing redundant distributed mechanism for configuration and service discovery; Logging and Monitoring System, a uniform way for all components to report their status and activities, and present that information to the users. The current functionality is the result of the experience collected during the last years of running with DIRAC for LHC*b* .

1. Introduction

DIRAC (*Distributed Infrastructure with Remote Agent Control*) is the grid environment designed for the LHCb experiment. DIRAC's logic is built on top of basic services that provide transversal functionality to DIRAC Systems. The set of basic services that form the core framework for DIRAC are:

- Configuration Service
- Monitoring Service
- Logging Service
- DISET
- Web framework

The Configuration Service is responsible for propagating all the necessary configuration parameters for all DIRAC components to run. Before doing anything, DIRAC components must contact a Configuration Server to retrieve the information they need. For instance, where other components are or what is the valid user list.

Once the component has completed a task, reporting can be done through Monitoring servers. The Monitoring system is repository of finished tasks status. For more detailed information, DIRAC components can send their log messages to the Logging Service. Where they will be stored for a specified period of time allowing DIRAC administrators to see what's happening almost in realtime.

Displaying Monitoring and Logging Services can be done though the DIRAC web site. The web site can identify users based on the certificate that can be loaded on the web browser. It can react to that certificate granting the user permissions based on definitions in the Configuration

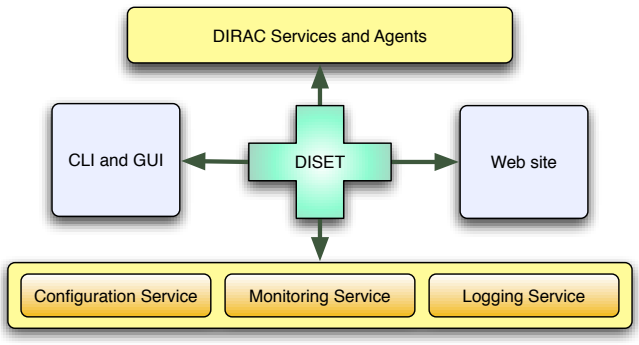

Figure 1. DIRAC Framework

Service. All that communication is done using DISET (*DIRAC SEcure Transport*), that provides RPC and file transfer mechanisms through TCP/IP allowing grid authentication.

2. Configuration Service

The Configuration Service (CS) is a data repository used to share configuration information over the Internet to external applications. Data is organized in sections containing sub-sections, and options with their values, and distributed using DISET. Being able to distribute the configuration is critical to DIRAC. If the configuration data is not available, DIRAC systems can't work. Because of that, the Configuration Service has been designed with resilience and scalability as the top priorities.

DIRAC Configuration Service is divided in:

- Configuration Master server
- Configuration Slave servers
- Configuration clients

All Configuration servers hold the configuration data in memory and on disc, although it is served from memory. Configuration servers are organized in a hierarchical way. There's one master server and multiple slaves. Only the master server allows modifications to the configuration data. Slave servers are read-only. To synchronize servers, slaves periodically poll the master server for modifications. If the configuration data has changed, slaves download the whole new configuration data.

DIRAC Components query the Configuration library to access the configuration data. Configuration library tries to answer the query from the memory cache it has. The memory cache is synchronized to any Configuration server as slave servers are synchronized to the master server. But instead of contacting any server periodically, the memory cache has a expiration time. Whenever a query to the cache is done, the expiration date is checked. If the cache has expired, a new copy of the configuration is downloaded from a Configuration Server overwriting the previous cache. The new configuration data in the cache will be used to answer the query.

A part from having the global configuration data, Configuration clients need a local one. For instance, in order to access the global configuration, a Configuration server has to be defined. A local configuration can be defined using one, or multiple files, or command line switches. Configuration Client will merge all the configuration sources (including the remote one if a Configuration server has been specified) into one. To do so, sources have to be ordered by precedence. Command line switches take precedence over the rest, then local files are included.

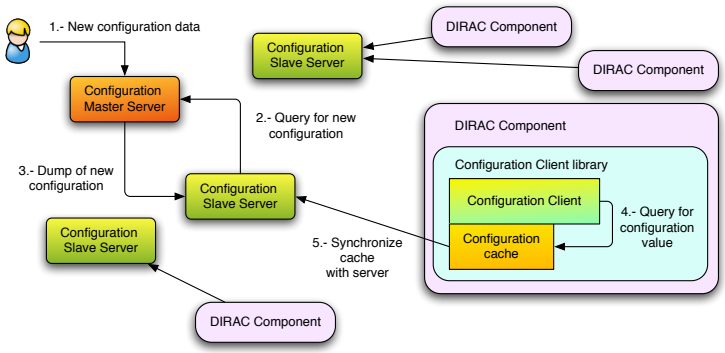

Figure 2. DIRAC Configuration Service schema

Remote configuration data is the source with less precedence. So, when a query is received, if it is specified in a command line switch that value is used as answer. Then, if it's specified in a local file, that value is taken. If all of the previous fail, then the remote configuration is take into account.

3. Monitoring and Logging Services

Monitoring and Logging systems are the representations of the status of a DIRAC setup. DIRAC Services and Agents send their activity reports to the Monitoring System, and important text messages to the Logging System respectively. Those systems process the received input from services and store the result for a specified period of time.

In order for DIRAC components to connect to the Monitoring Service, they use the Monitoring client. DIRAC components must define data sources before reporting anything. To define a data source, a name, units, granularity, life time, group and type must be defined. Once data sources have been defined, the Monitoring client is ready to receive reports. DIRAC components do not need to explicitly send the data. They only have to declare when or how many activities have finished. Periodically, the Monitoring client will check if the component has declared any activity. If any activity has been declared, it will be sent to the Monitoring server. For instance, if *files sent* was to be reported, every time a file is sent the DIRAC component must tell the Monitoring client. The Monitoring client will wake up every five minutes (that number can be changed in the configuration) and send any report. Using the granularity specified in the data source, a Monitoring Agent will process the activity received and store it.

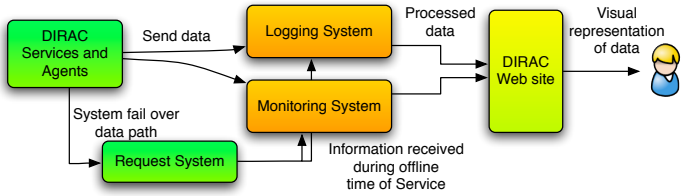

Figure 3. Monitoring and Logging Services data path

The Logging Service has a similar way of sending data to the server. DIRAC components also have a Logging client that will periodically try to send the messages to the server. The

only difference is that the Logging client can have more than one data destinations. The usual behavior for a Logging client is to show the messages using the standard output a part from sending them to the server. Each data destination can have a level of relevance defined. Any message below that relevance level won't be sent to that data destination. Standard output destination can have *INFO* relevance level, whereas the remote destination can have a *ERROR* relevance.

Monitoring and Logging Agents process the received data as well as generate reports and statistics for DIRAC users. These reports and statistics are made available dynamically through the DIRAC web site. In case the servers are offline Monitoring and Logging clients will send the data to the Request Service which will forward the data to the final destination once the services are back online.

4. DISET

DISET (DIRAC SEcure Transport) is the DIRAC secure transport layer. It takes care of all the communication needs between DIRAC components. Communication is done using plain TCP/IP when unsecure connections are needed. When a secure connection is requested, DISET uses OpenSSL through a modified python binding. This provides grid authentication and encryption, using X509 certificates and grid proxies.

DISET provides RPC (*Remote Procedure Call*) and file transfer capabilities to DIRAC Services, allowing multiple threads. Only the servers functionality has to be coded. When a RPC is queried, DISET serializes the *python* objects involved and sends them though the connection. Once received, the query is expanded and the type of the parameters received is checked. If the types received are not expected, an error is returned to the client. If the call is correct, the proper user coded method is called. After the execution of the user code, the response is serialized and sent back to the origin.

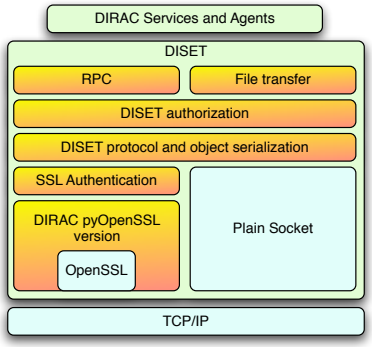

Figure 4. DISET structure

File Transfer is also provided by DISET. DIRAC servers can receive and send files and directories. When a directory is transferred, it is packed on the fly and sent through the connection. On the receiving side, a directory can be unpacked into the filesystem, recovering the original directory structure, or saved as a standard UNIX *.tar* file.

User and group authorization is also provided for RPC and file transfer. Each method can have its own authorization rule defined. Authorization rules are defined using the Configuration System. They can be defined in the global configuration, so any change will propagate to all running instances of affected Services.

5. Web site

The DIRAC web site is the standard way to present information to visitors. Unauthenticated visitors can access unrestricted information such as project information, downloads, documentation, DIRAC status... For more detailed information visitors must authenticate themselves loading a valid grid certificate into the web browser. The web server *ssl* module will authenticate the certificate using grid approved Certification Authorities (CAs). Once the user certificate has been authorized, DIRAC web site will check if the user's Distinguished Name (DN) is defined in the global configuration. That information will be used to grant access to different parts of the web site.

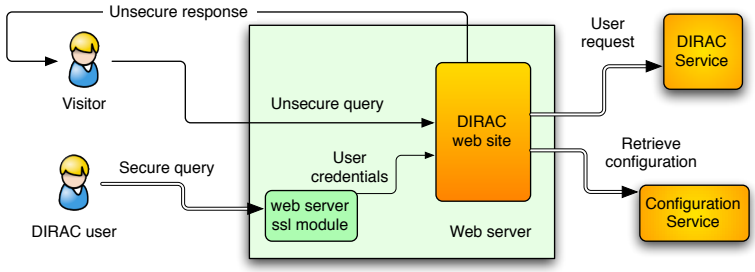

Figure 5. DIRAC Web site functionality

DIRAC web site can provide different functionality to authenticated users based on their groups. For instance, DIRAC users can see their jobs and have minimal interaction with them. DIRAC administrators can change the global configuration in a visual way using web technologies like *AJAX*. And DIRAC production managers can manage productions easily. All that interaction is done using DISET to communicate with DIRAC servers to retrieve required information and forward the user's request.

6. Conclusions

DIRAC Framework is a complete suite that eases building complex Systems on top. DISET allows to communicate easily any service by issuing secure RPC calls, and remotely sending and receiving files and directories. Using DISET, Monitoring and Logging services provide administrators almost realtime status of Services so they can react in time. That information can be displayed through the web site. The Configuration System is the glue that ties all together by sharing global configuration required by DIRAC Services.

References

[1] Tsaregorodtsev A et al. 2007 *DIRAC: A community grid solution* (Computing in High Energy and Nuclear Physics, Victoria, Canada)

[2] Graciani Díaz R and Casajús Ramo A 2007 *DIRAC Agents and Services* (Computing in High Energy and Nuclear Physics, Victoria, Canada)

[3] Paterson S 2007 *DIRAC Optimized Workload Management* (Computing in High Energy and Nuclear Physics, Victoria, Canada)

[4] Smith A 2007 *DIRAC: Reliable Data Management for LHC*b (Computing in High Energy and Nuclear Physics, Victoria, Canada)

[5] Casajús Ramo A and Graciani Díaz R 2006 *Proc. Computing in High Energy and Nuclear Physics (Mumbai)* vol 2, ed S Banerjee (India: Macmillan) p 746

[6] Paterson S and Tsaregorodtsev A 2006 *Proc. Computing in High Energy and Nuclear Physics (Mumbai)* vol 2, ed S Banerjee (India: Macmillan) p 1096