

Introduction

*IHEP, Beijing,
13-15 November 2013*



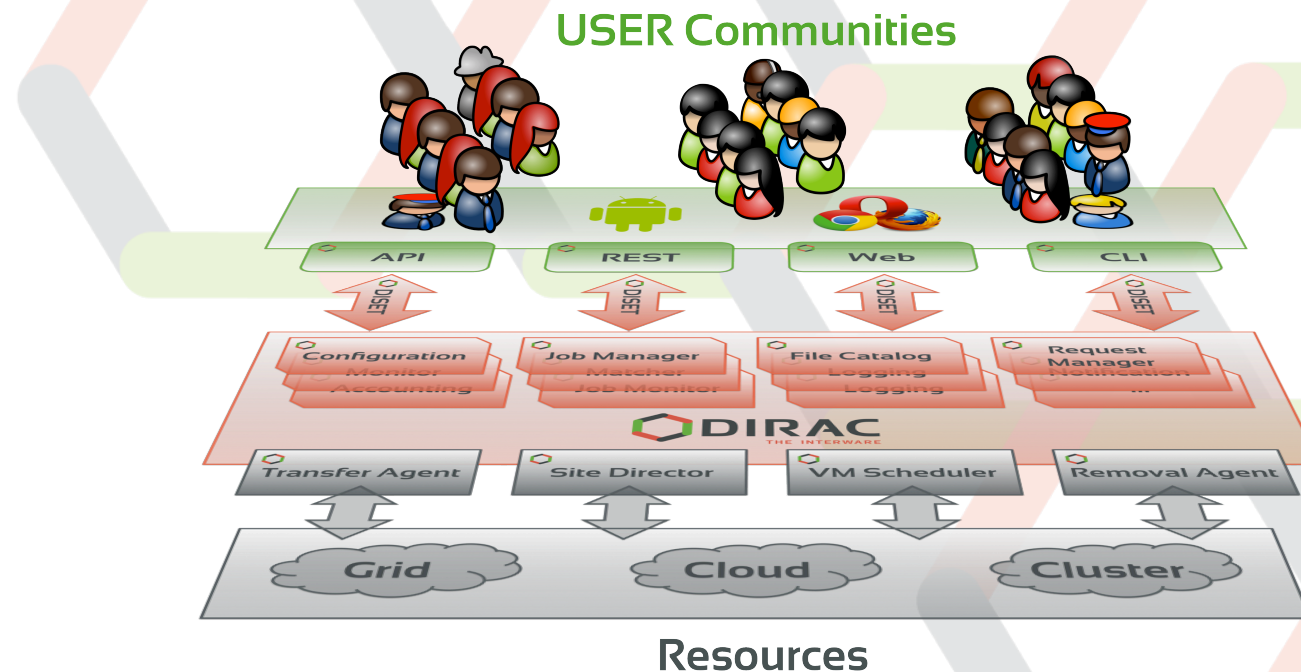
- ▶ DIRAC Project
- ▶ DIRAC grid middleware
- ▶ DIRAC as a Service
- ▶ Tutorial plan

- ▶ LHC experiments pioneered the massive use of computational grids
 - ▶ 10s of PBytes of data per year
 - ▶ 100s of thousands CPUs in 100s of centers
 - ▶ 10s GB/sec network transfers
 - ▶ 100s of users from 100s of institutions
- ▶ CERN Director General Rolf Heuer about the Higgs discovery:

"It was a global effort and it is a global success. The results today are only possible because of the extraordinary performance of the accelerators, including the infrastructure, the experiments, and the *Grid computing*."
- ▶ Other domains are catching up quickly with the HEP experiments
 - ▶ Life sciences, earth sciences, astrophysics, social sciences, etc

- ▶ The computing expertise level in non-HEP scientific domains is relatively lower
 - ▶ Grouped around well known applications and scientific portals
 - ▶ Moving existing applications to run in distributed environments is still difficult
- ▶ Convenient tools for small research groups with no local gurus are clearly needed
- ▶ All LHC experiments developed their own middleware
 - ▶ PanDA, AliEn, glideIn WMS, PhEDEx, DIRAC, ...
 - ▶ WMS with pilot jobs, intelligent data management, software distribution, ...
- ▶ Experience of the LHC experiments in using distributed computing infrastructures should now be made available for non-LHC user communities

- ▶ DIRAC provides all the necessary components to build ad-hoc distributed computing infrastructures interconnecting resources of different types, allowing interoperability and simplifying interfaces. This allows to speak about the DIRAC *interware*.

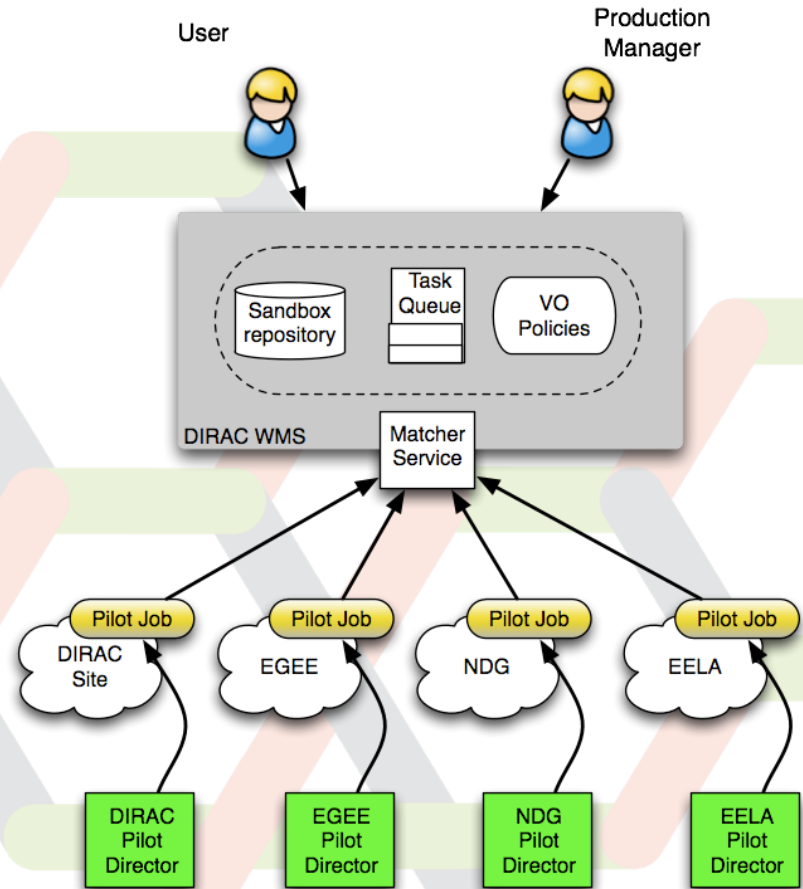


- ▶ Several new experiments expressed interest in using this software relying on its proven functionality
- ▶ In 2009 the core DIRAC development team decided to generalize the software to make it suitable for any user community.
 - ▶ Separate LHCb specific functionality into a set of extensions
 - ▶ Introduce new services to make it a complete solution
 - ▶ Support for multiple small groups by a single DIRAC installation
 - ▶ General refurbishing of the code, code management, deployment, documentation, etc
- ▶ This work made it possible to offer general-purpose DIRAC services to any scientific community



Workload Management

- ◆ Jobs are submitted to the DIRAC Central Task Queue with credentials of their owner (VOMS proxy)
- ◆ Pilot Jobs are submitted by specific Directors to a Grid WMS with credentials of a user with a special Pilot role
- ◆ The Pilot Job fetches the user job and the job owner's proxy
- ◆ The User Job is executed with its owner's proxy used to access SE, catalogs, etc

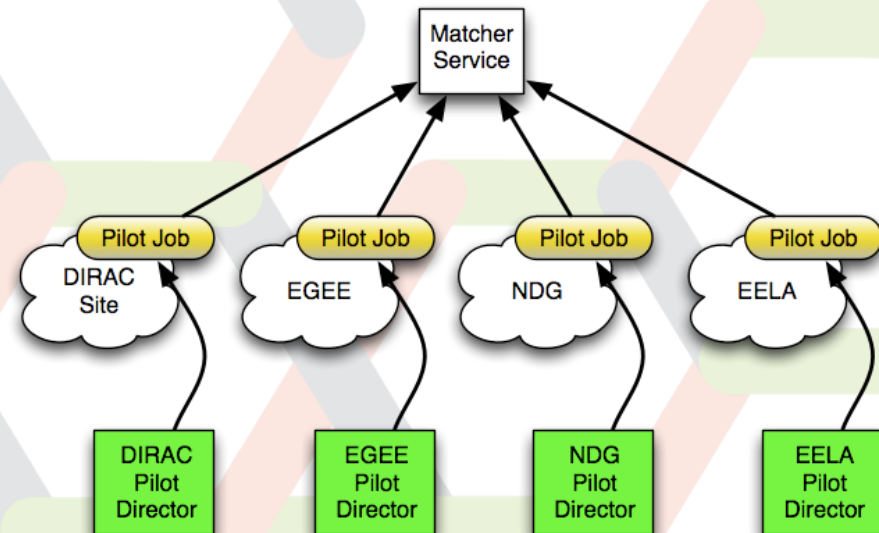


- ▶ Including resources in different grids and standalone clusters is simple with Pilot Jobs

Needs a specialized Pilot Director per resource type

Demonstrated with various grid sites, clouds, etc

Users just see new sites appearing in the job monitoring



- ▶ DIRAC middleware facilitates access to various types of resources
 - ▶ gLite and ARC middleware based grids (EGI, NDGF, etc)
 - ▶ Standalone clusters
 - ▶ Simple SSH accessible account is sufficient to include the site
 - ▶ Clouds (Amazon, OpenStack, OpenNebula, OCCl compliant)
 - ▶ Automatic virtual machine scheduling
 - ▶ Desktop Grid
 - ▶ Based on BOINC technology
 - ▶ Support for multiple platforms with virtualization

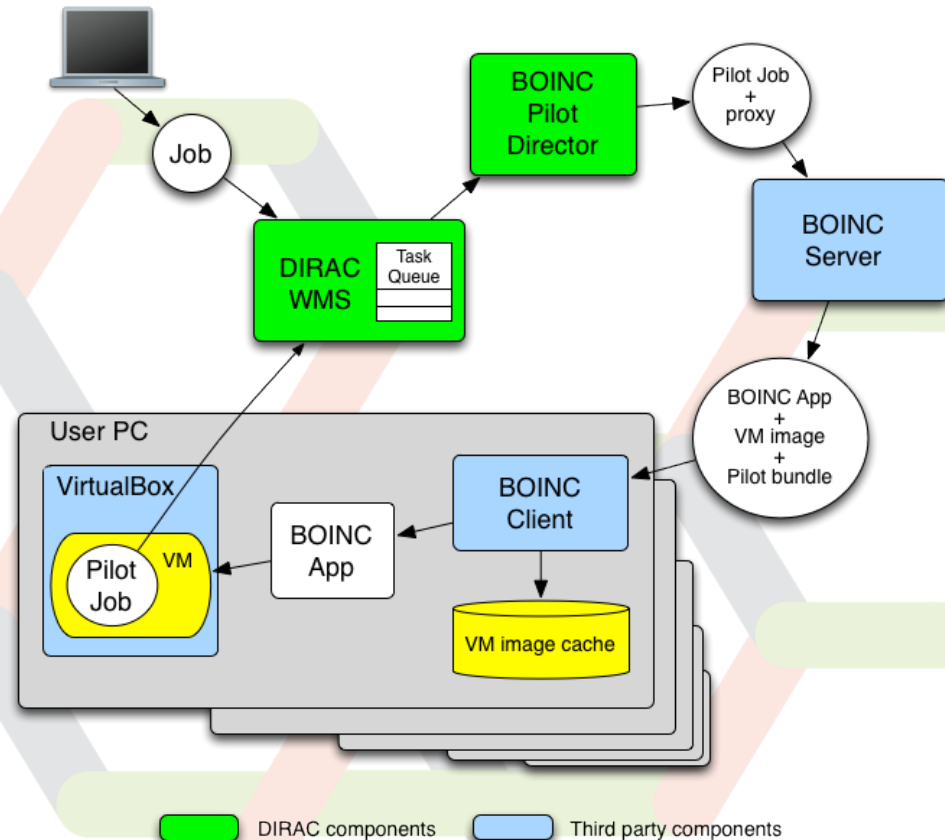
- ▶ On the client PC the third party components are installed:
 - ▶ VirtualBox hypervisor
 - ▶ Standard BOINC client

- ▶ A special BOINC application
 - ▶ Starts a requested VM within the VirtualBox
 - ▶ Passes the Pilot Job to the VM and starts it

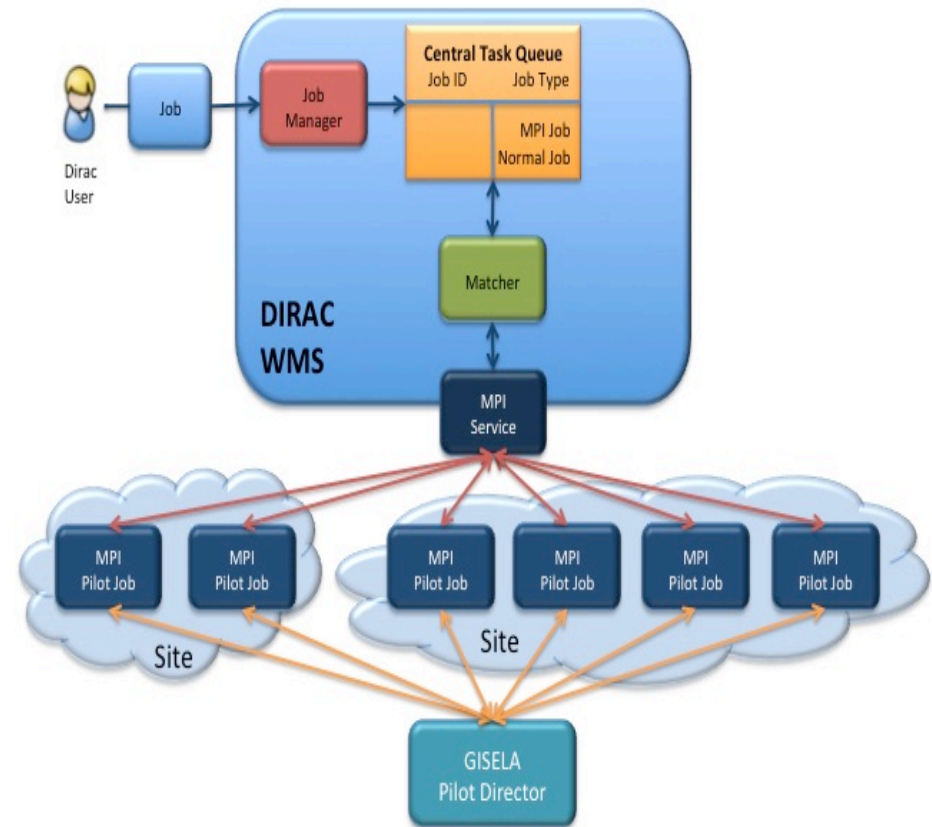
- ▶ Once the Pilot Job starts in the VM, the user PC becomes a normal DIRAC Worker Node

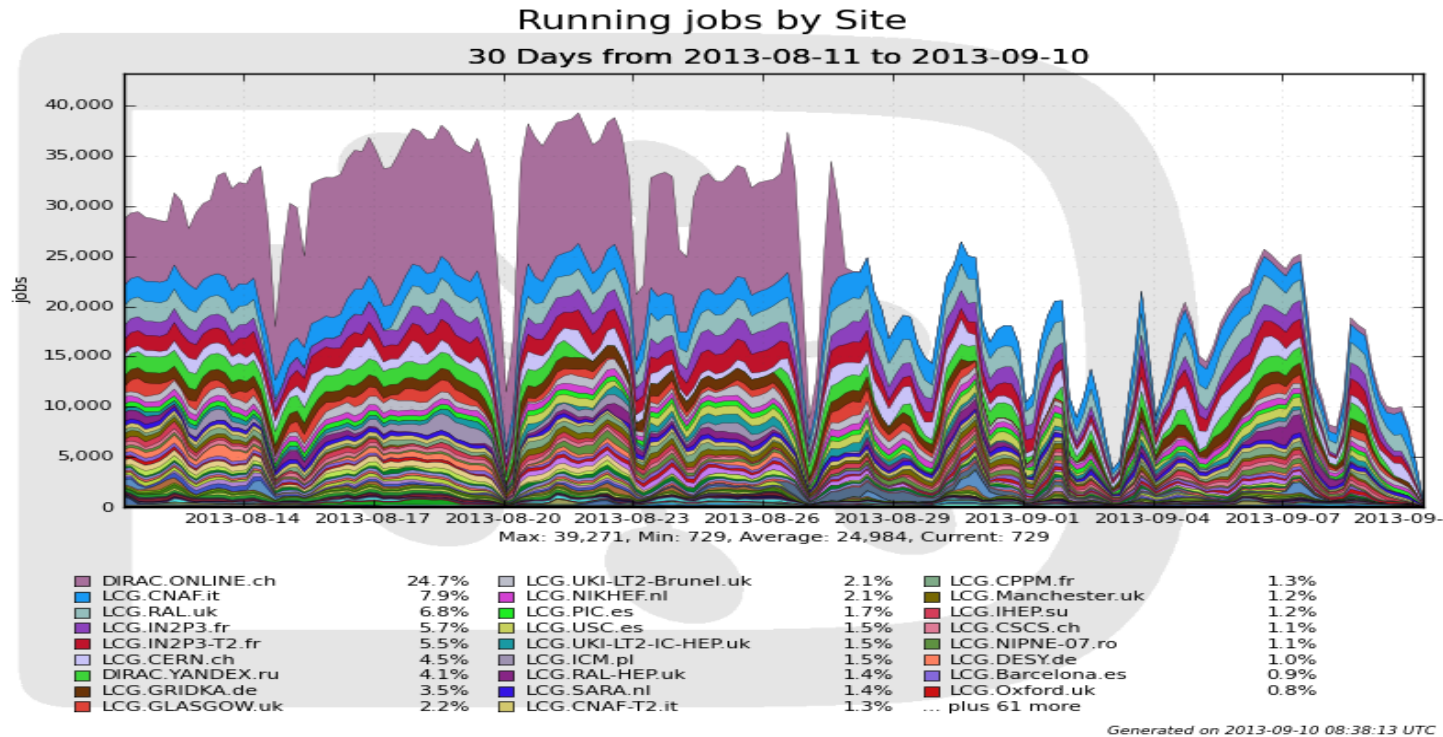
- ▶ Possibility to use the MarketPlace repository of VM images

- ▶ Interfacing DIRAC to EDGI resources
 - ▶ Using EDGI provided special CREAM CE service



- ▶ MPI Service developed for applications in the EELA/GISELA Grid
 - ▶ Astrophysics, BioMed, Seismology applications
 - ▶ No special MPI support on sites is required
 - ▶ MPI software is installed by Pilot Jobs
 - Possibility to use distributed file systems, e.g. *Parrot*
 - ▶ MPI ring usage optimization
 - ▶ Ring reuse for multiple jobs
 - Lower load on the gLite WMS
 - ▶ Variable ring sizes for different jobs





- ▶ DIRAC performance in production
 - ▶ Up to 50K concurrent jobs in ~120 distinct sites
 - ▶ 10 mid-range central servers hosting DIRAC services
 - ▶ Further optimizations to increase capacity are possible



Data Management

▶ Storage Elements

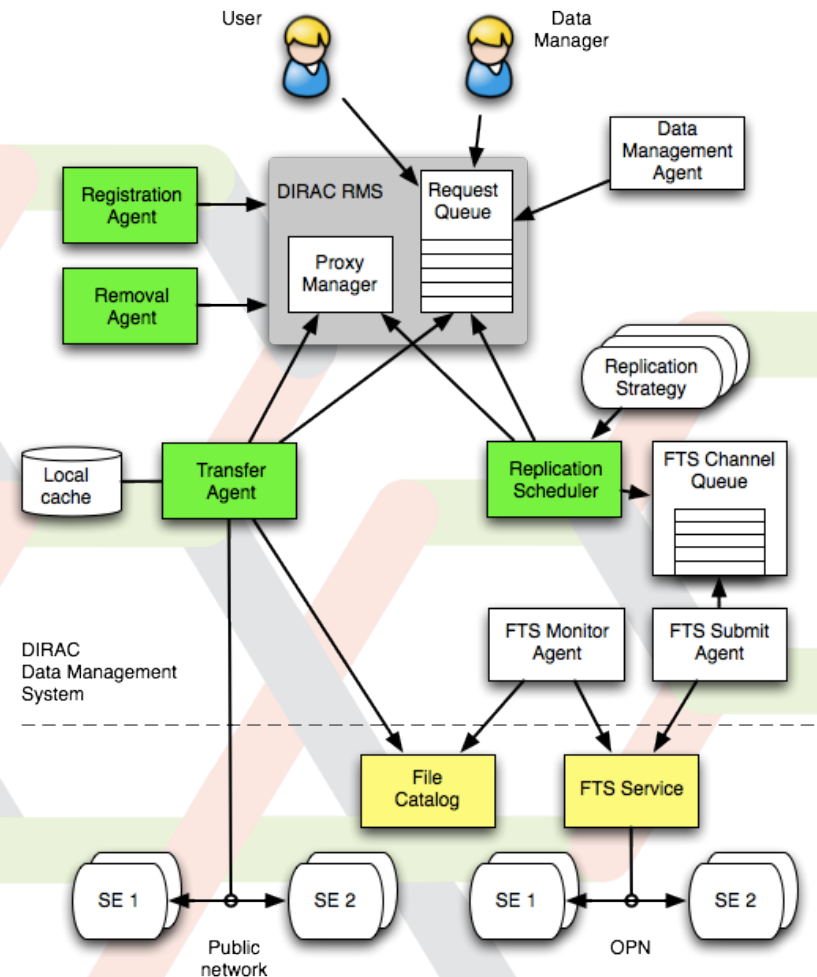
- ▶ gLite/EGI Storage Elements
- ▶ DIRAC Storage Elements
- ▶ More Storage Elements can be included
 - ▶ (F,SF,HT,BBF)TP servers
 - ▶ iRods, S3

▶ File Catalogs

- ▶ LCG File Catalog (LFC)
- ▶ DIRAC File Catalog
 - ▶ Support for the User Metadata (similar to the AMGA gLite service)
 - ▶ Support for data provenance
- ▶ More Catalogs can be included
 - ▶ LHCb has developed several specific catalogs in the same framework

- ▶ For DIRAC users the use of any Storage Element or File Catalog is transparent
 - ▶ Community choice which components to use
 - ▶ Different SE types can be mixed together
 - ▶ Several File Catalogs can be used in parallel
 - ▶ Complementary functionality
 - ▶ Redundancy
- ▶ Users see depending on the DIRAC Configuration
 - ▶ Logical Storage Elements
 - ▶ e.g. DIRAC-USER, M3PEC-disk
 - ▶ Logical File Catalog

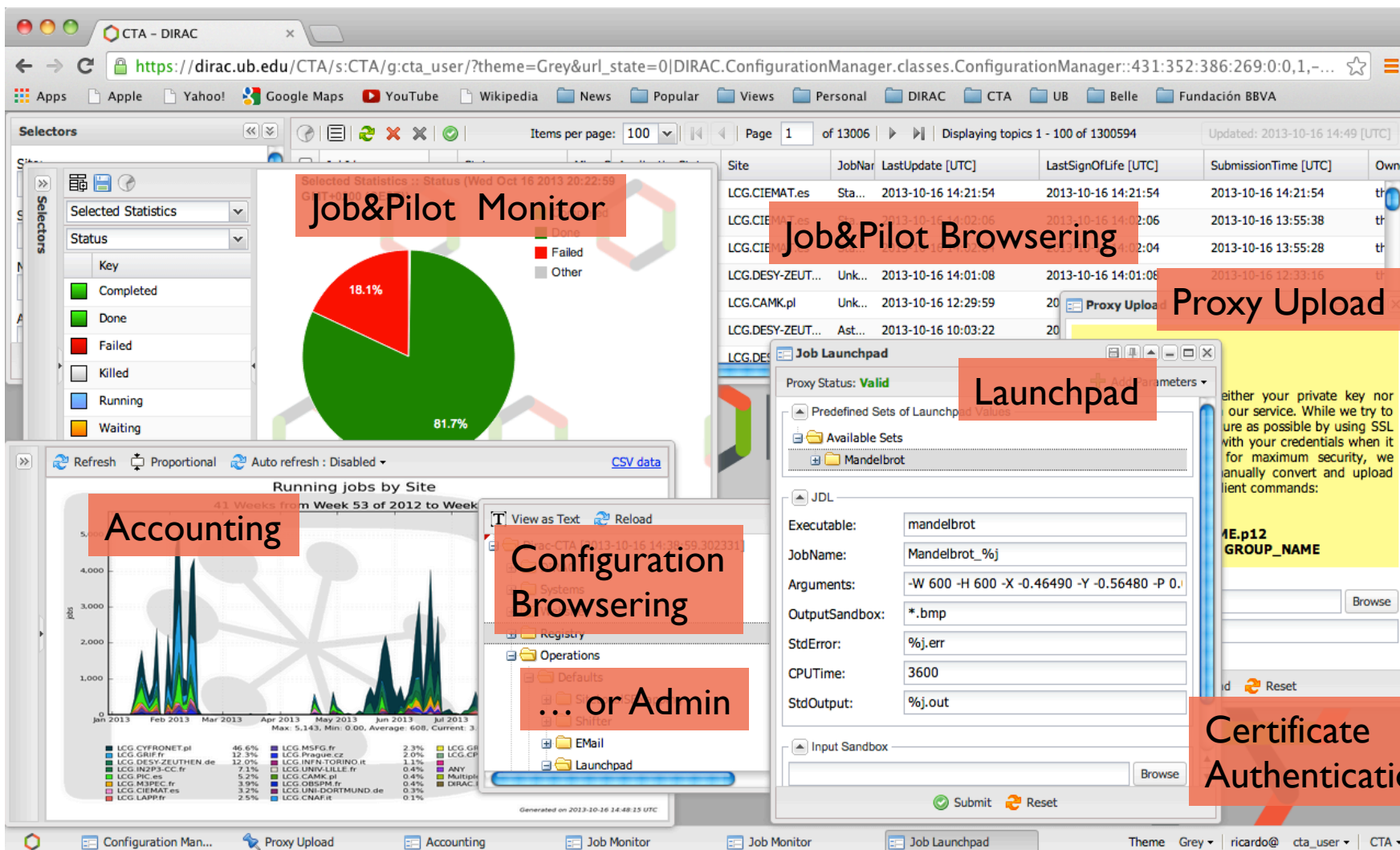
- ▶ Based on the Request Management System
- ▶ Asynchronous data operations
 - ▶ transfers, registration, removal
- ▶ Two complementary replication mechanisms
 - ▶ Transfer Agent
 - ▶ user data
 - ▶ public network
 - ▶ FTS service
 - ▶ Production data
 - ▶ Private FTS OPN network
 - ▶ Smart pluggable replication strategies





User Interfaces

- ▶ Focus on the Web Portal as the main user tool for interactions with the grid
- ▶ Intuitive desktop application like interface
 - ▶ Ajax, Pylons, ExtJS Javascript library
- ▶ Monitoring and control of all activities
 - ▶ User job monitoring and manipulation
 - ▶ Data manipulation and downloads
 - ▶ DIRAC Systems configuration and management
- ▶ Secure access
 - ▶ Standard grid certificates
 - ▶ Fine grained authorization rules



The screenshot shows the DIRAC web portal interface with several key components highlighted:

- Job&Pilot Monitor:** A pie chart showing job status distribution: 81.7% Done (green), 18.1% Failed (red), and 0.2% Other (grey).
- Job&Pilot Browsing:** A table listing jobs with columns for Site, JobName, LastUpdate [UTC], LastSignOfLife [UTC], SubmissionTime [UTC], and Owner.
- Proxy Upload:** A section for uploading proxy certificates, including a text area for instructions and a 'Browse' button.
- Launchpad:** A configuration window for a job launchpad, showing fields for Executable (mandelbrot), JobName (Mandelbrot_%j), Arguments (-W 600 -H 600 -X -0.46490 -Y -0.56480 -P 0.), OutputSandbox (*.bmp), StdError (%j.err), CPUTime (3600), and StdOutput (%j.out).
- Accounting:** A line graph showing running jobs by site over time, with a legend listing various sites and their percentages.
- Configuration Browsing ... or Admin:** A sidebar menu for navigating through system configurations and administrative tools.



DIRAC Framework

- ◆ Services oriented architecture (SOA)
- ◆ DIRAC has a well defined architecture

Services

passive components reacting to client request

Keep their state in a database

Light distributed agents

▶ permanently running components, animating the whole system

Clients

User interfaces

Agent-service, service-service communications

▶ Technologies

▶ Python, MySQL, OpenSSL

- ▶ All the communications between the distributed components are secure

 - DISET custom client/service protocol

 - Focus on efficiency

 - Control and data communications

 - X509, GSI security standards

 - Fine grained authorization rules

- ▶ Framework allows to easily build these components concentrating on the business logic of the applications

 - Making use of rich base services

▶ **Redundant Configuration Service**

Provides service discovery and setup parameters for all the DIRAC components

Full featured proxy management system

Proxy storage and renewal mechanism

Support for multiuser pilot jobs

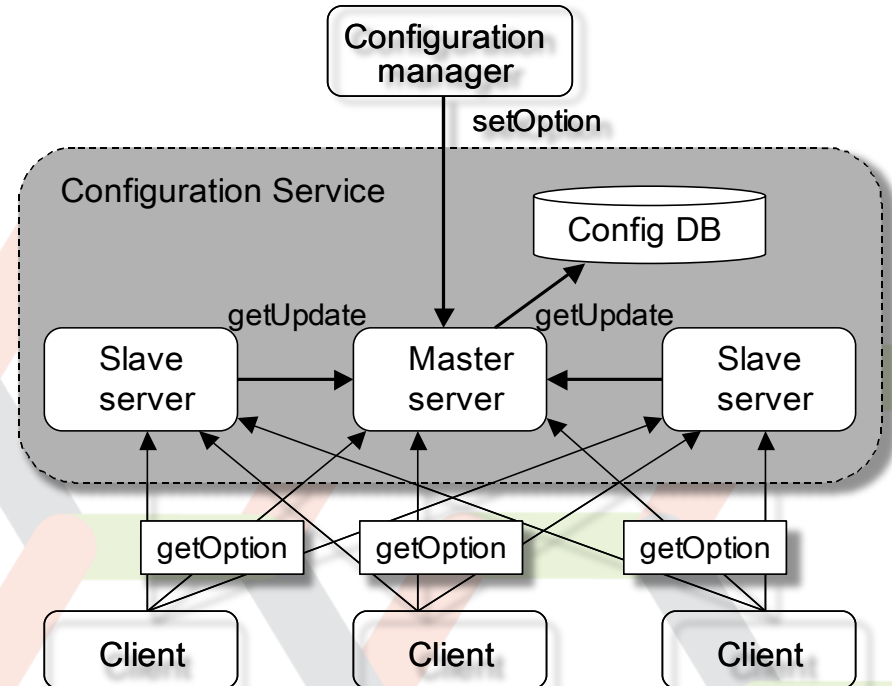
System Logging service

Collect essential error messages from all the components

Monitoring service

Monitor the service and agents behavior

▶ **Accounting service**

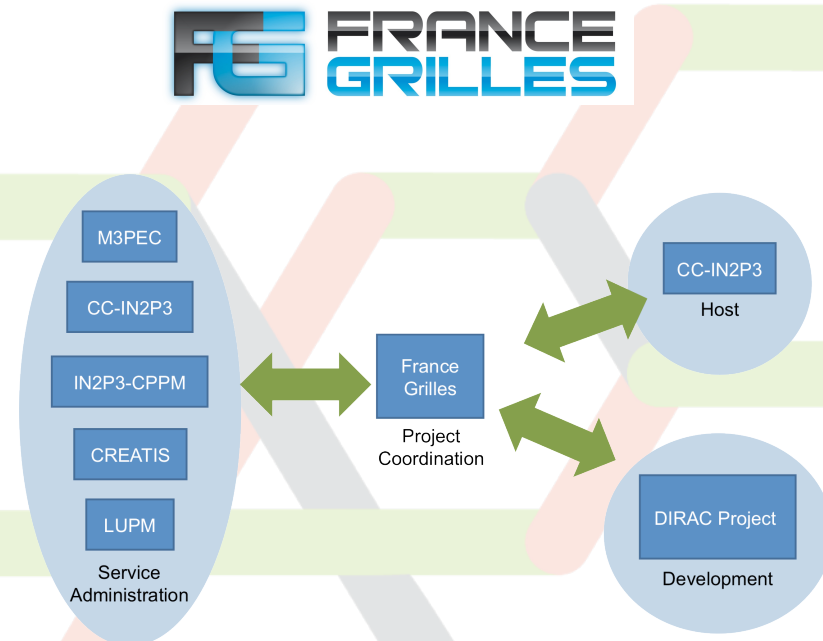




DIRAC as a Service

- ▶ DIRAC client is easy to install
 - ▶ Part of a usual tutorial
- ▶ DIRAC services are easy to install but
 - ▶ Needs dedicated hardware for hosting
 - ▶ Configuration, maintenance needs expert manpower
 - ▶ Monitoring computing resources is a tedious every-day task
- ▶ Small user communities can not afford maintaining dedicated DIRAC services
 - ▶ Still need easy access to computing resources
- ▶ Large grid infrastructures can provide DIRAC services for their users.

- ▶ **DIRAC** services are provided by several National Grid Initiatives: France, Spain, Italy, UK ...
- ▶ Example: France-Grilles DIRAC service
 - ▶ Hosted by the CC/IN2P3
 - ▶ Distributed administrator team
 - ▶ 5 participating universities
 - ▶ 15 VOs, ~100 registered users
 - ▶ In production since May 2012
 - ▶ 7 millions jobs



- ▶ Heavily used for the grid tutorials
 - ▶ Using resources of the VO france-formation
- ▶ Support for users, applications
 - ▶ Forum for experience dissemination
 - ▶ Help in porting applications to the grid
 - ▶ Help new communities to try out DIRAC for their production systems

- Fermi-LAT, Glast
- LSST
- CTA
- ...



DIRAC has most of the features of a “standard” Grid middleware stack

Power users will see extra support:

- Massive job execution

- Data operations

Developers can easily add new functionalities specific for their applications

Community administrators get tools to apply community policies

- User and group priorities, quotas

Site administrators can easily include their resources

- Easy addition of new resources without bulky installation

- Easy user management with only one “VO user”

The DIRAC project is in full development

- More new exciting features to come – stay tuned !

- Your contributions are welcome

Getting Started

- DIRAC client installation

- Getting ready user credentials

Job execution mechanics

- Basic job operations with Web Portal explained

Job manipulation tools

- Submission, monitoring, getting results

Basic data management operations

- Data upload, download, replication

Advanced job operations

- Jobs with input and output data

- Bulk job submission

Emphasis on exercises

Agenda <https://indico.in2p3.fr/conferenceDisplay.py?confId=9051>

- ▶ DIRAC service installation at *dirac.ihep.ac.cn*

The service used for the tutorial is permanent, will stay in place afterwards

Resources

5 EGI sites

IHEP Computing Centre as a standalone cluster

Several BOINC nodes

4 SE's

One DIRAC SE (DIRAC-USER)

3 SRM EGI SE's

Tutorial materials are available here

<https://github.com/DIRACGrid/DIRAC/wiki/DIRAC-Tutorials>

- ▶ The course will be given by the members of the DIRAC Project team (<http://diracgrid.org>)
- ▶ Tutors:
 - ▶ Andrei Tsaregorodtsev – DIRAC Project Coordinator, CPPM
 - ▶ Vanessa Hamar – Responsible for the DIRAC production infrastructure DIRAC France-Grilles NGI, CC/IN2P3