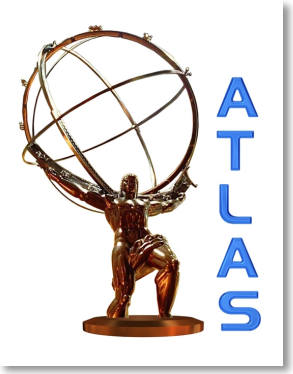
Introduction

DIRAC Project

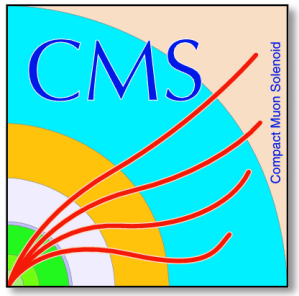
Plan

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

Grid applications



- HEP experiments collect unprecedented volumes of data to be processed on large amount of geographically distributed computing resources
  - 10s of PBytes of data per year
  - 10s of thousands CPUs in 100s of centers
  - 100s of users from 100s of institutions



However, other application domains are quickly approaching these scales

User problems on the grid

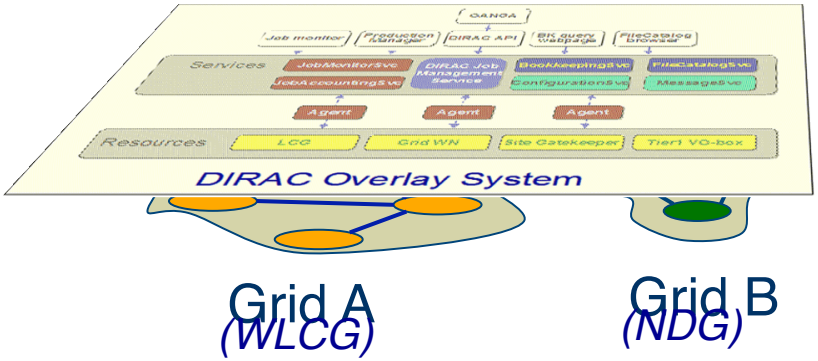
- Complicated interfaces
  - Especially for non-computing experts
- Confusing security infrastructure
  - Not easy to get and properly set up grid certificates
- Frustration with failing resources and middleware
  - Why my jobs worked yesterday and not today ?
- For small communities difficult to organize collective work
  - Lack of expertise in high level computing tasks
    - Massive jobs, massive data movement, etc
- Small communities tend to become larger with time

Large community issues

- Large user communities (Virtual Organizations) have specific problems
  - Dealing with heterogeneous resources
    - Various computing clusters, grids, etc
  - Dealing with the intracommunity workload management
    - User group quotas and priorities
    - Priorities of different activities
  - Dealing with a variety of applications
    - Massive data productions
    - Individual user applications, etc

DIRAC Grid Solution

- LHC experiments developed their own middleware to address the above problems
  - DIRAC is developed originally for the LHCb experiment
- DIRAC is providing a complete grid middleware stack with the goal:
  - Integrate all the heterogeneous computing resources available
  - Minimize human intervention at sites
  - Make the grid convenient for the users:
    - Fault tolerance, quicker turnaround of user jobs
    - Enabling Community policies



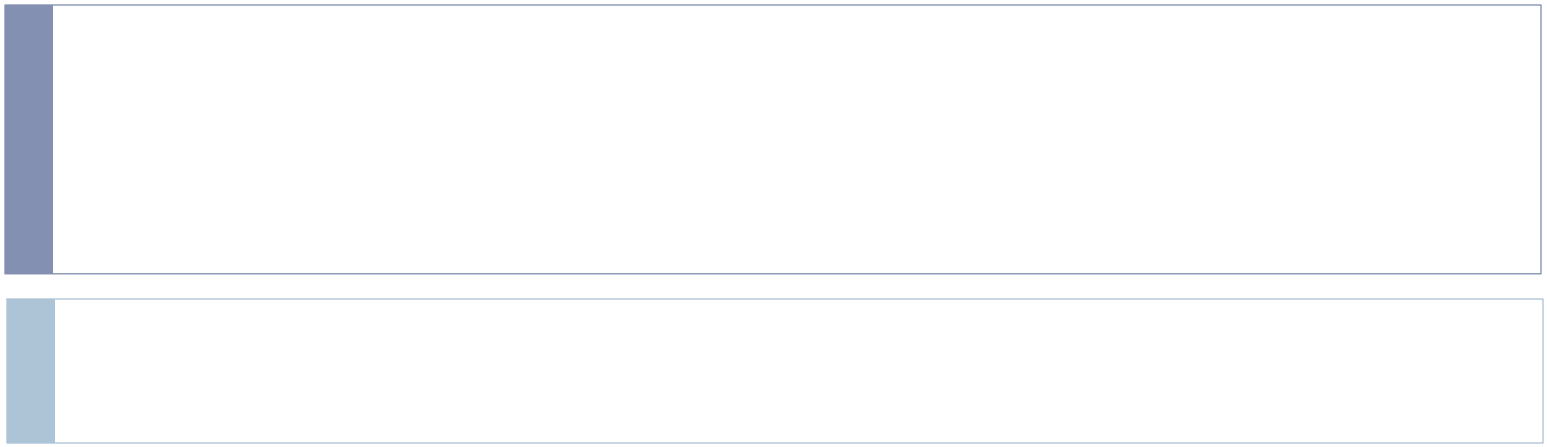
Origins

- DIRAC project was started as the LHCb distributed computing project
  - First, as a MC production engine
  - Then extended for all the other LHCb distributed computing tasks
- DIRAC was reorganized to separate generic and LHCb specific functionality in 2008-2010
  - Since 2010 DIRAC became an independent project
    - With LHCb staying the main client of the project
  - Main DIRAC developers are also LHCb experiment members
    - Guarantees of the project sustainability

DIRAC Consortium

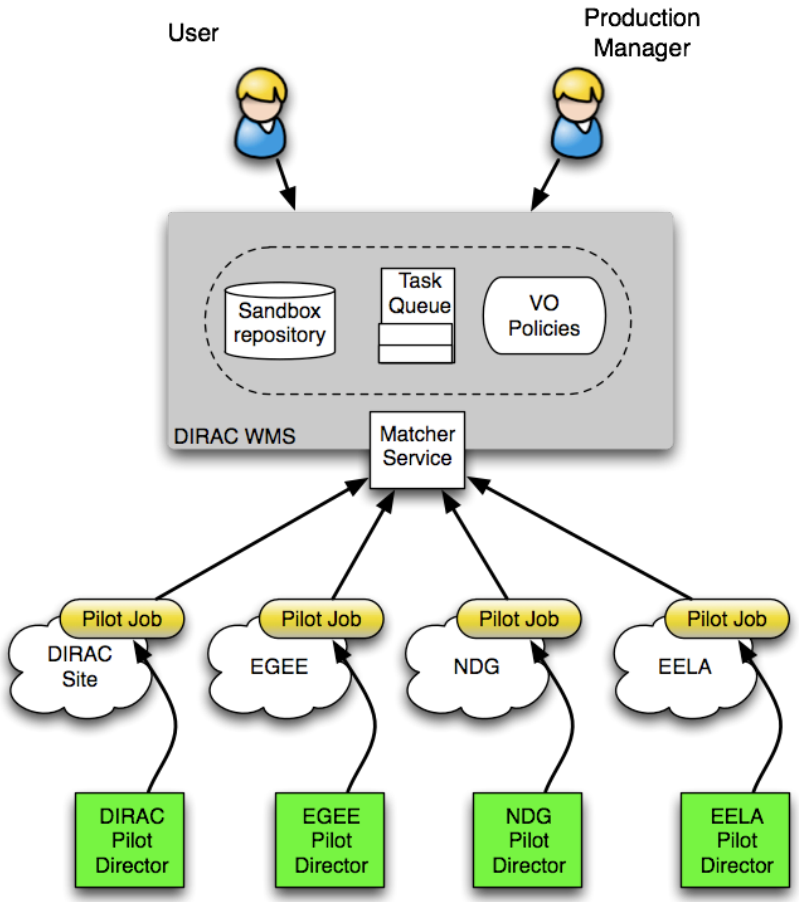
- Other projects are starting to use or evaluating DIRAC
  - CTA, SuperB, BES, VIP(medical imaging), …
    - Contributing to DIRAC development
    - Increasing the number of experts
  - Need for user support infrastructure
- Turning DIRAC into an Open Source project
  - DIRAC Consortium agreement in preparation
    - IN2P3, Barcelona University, CERN, …
  - http://diracgrid.org
    - News, docs, forum

Workload Management



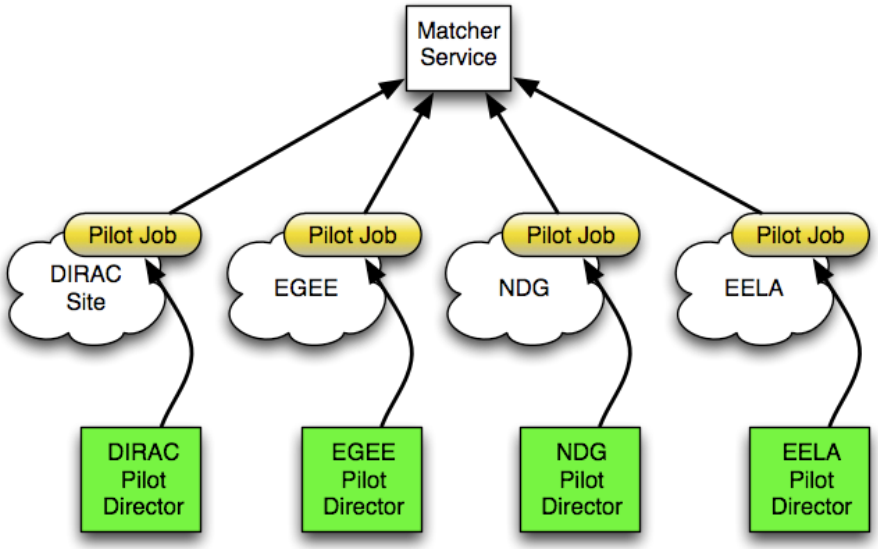
DIRAC WMS

- 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



WMS: using heterogeneous resources

- Including resources in different grids and standalone clusters is simple with Pilot Jobs
  - Needs a specialized Pilot Director per resource type
  - Demonstrated with NDG and GISELA grid sites, Amazon cloud, etc
  - Users just see new sites appearing in the job monitoring

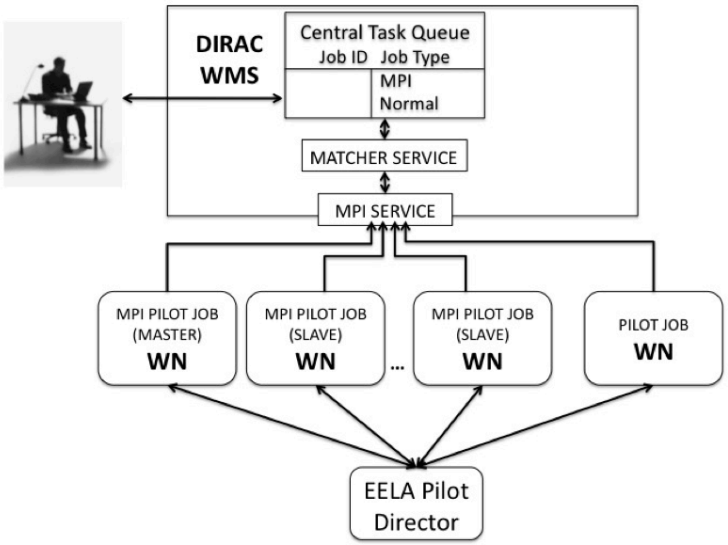


Resources provisioning

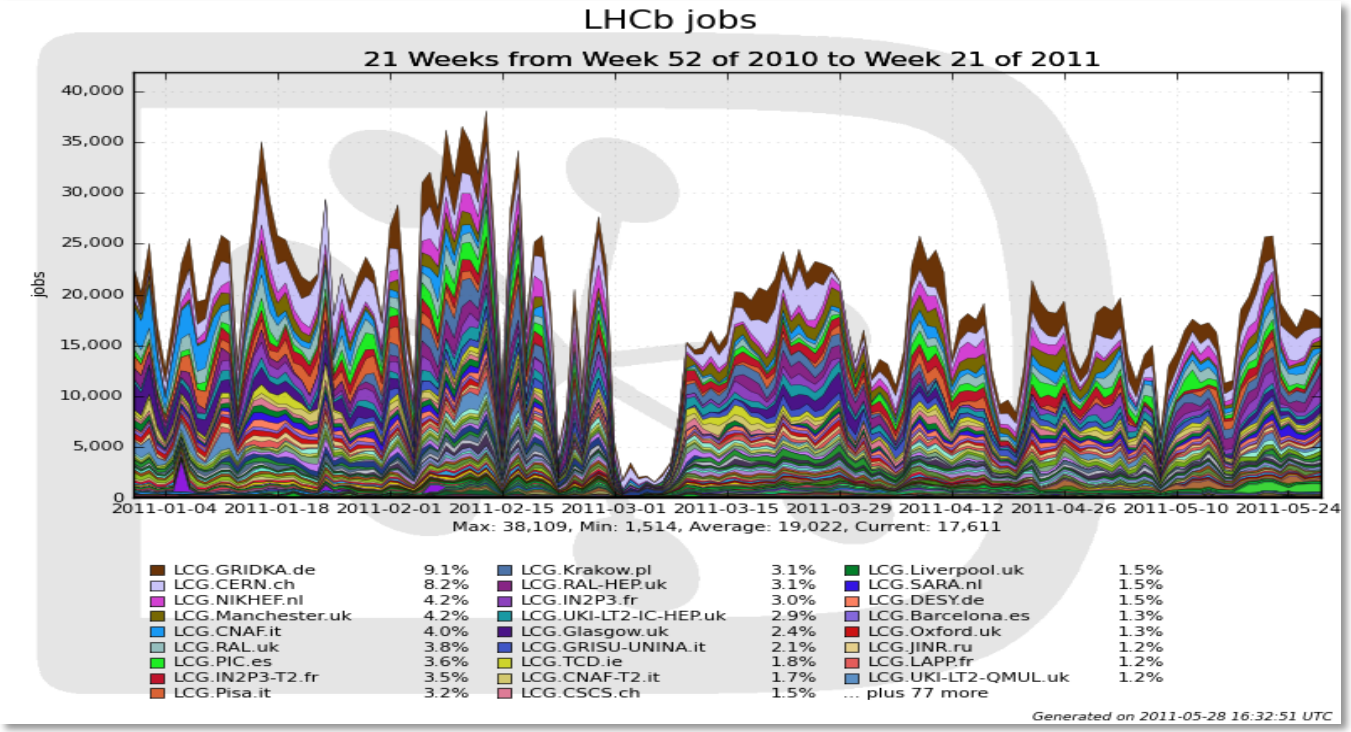
- DIRAC middleware facilitates access to various types of resources
  - gLite based grids
  - Standalone clusters
    - Simple SSH accessible account is sufficient to include the site
  - Clouds (Amazon, OpenNebula, OCCI compliant)
    - Automatic virtual machine scheduling
  - Desktop Grid
    - Based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

Support for MPI Jobs

- MPI Service developed for applications in the EELA Grid
  - Astrophysics, BioMed, Seismology applications
  - No special MPI support on sites
    - MPI software installed by Pilot Jobs
  - MPI ring usage optimization
    - Ring reuse for multiple jobs
      - Lower load on the gLite WMS
    - Variable ring sizes for different jobs

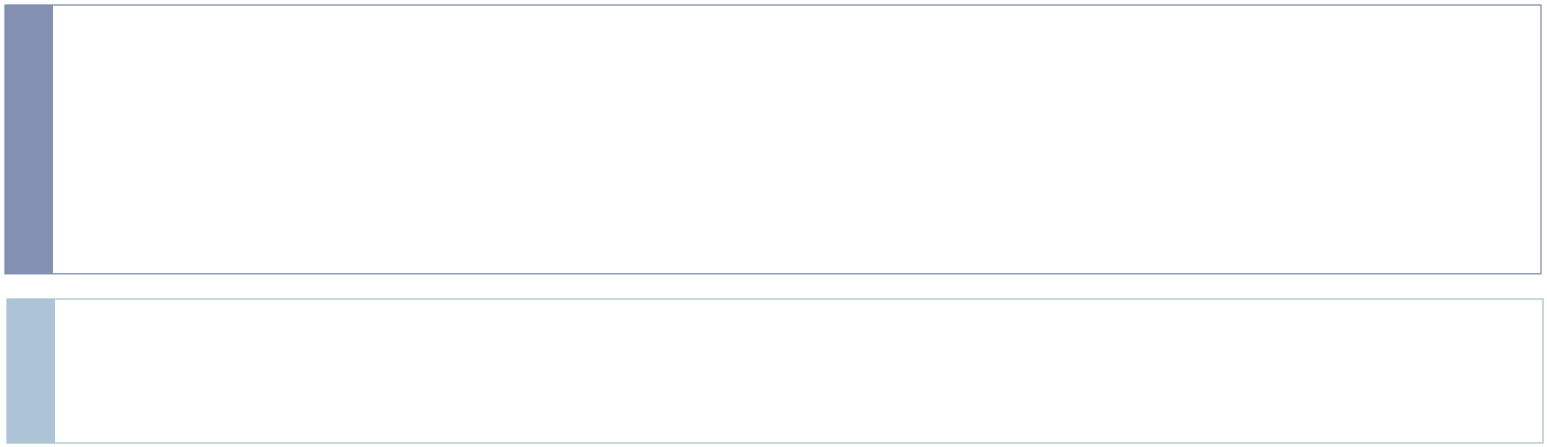


LHCb DIRAC performance



- DIRAC performance in production
  - Up to 35K concurrent jobs in ~120 distinct sites
  - 5 mid-range central servers hosting DIRAC services
  - Further optimizations to increase capacity are possible
    - Hardware, database optimizations, service load balancing, etc

Data Management



Data Management components

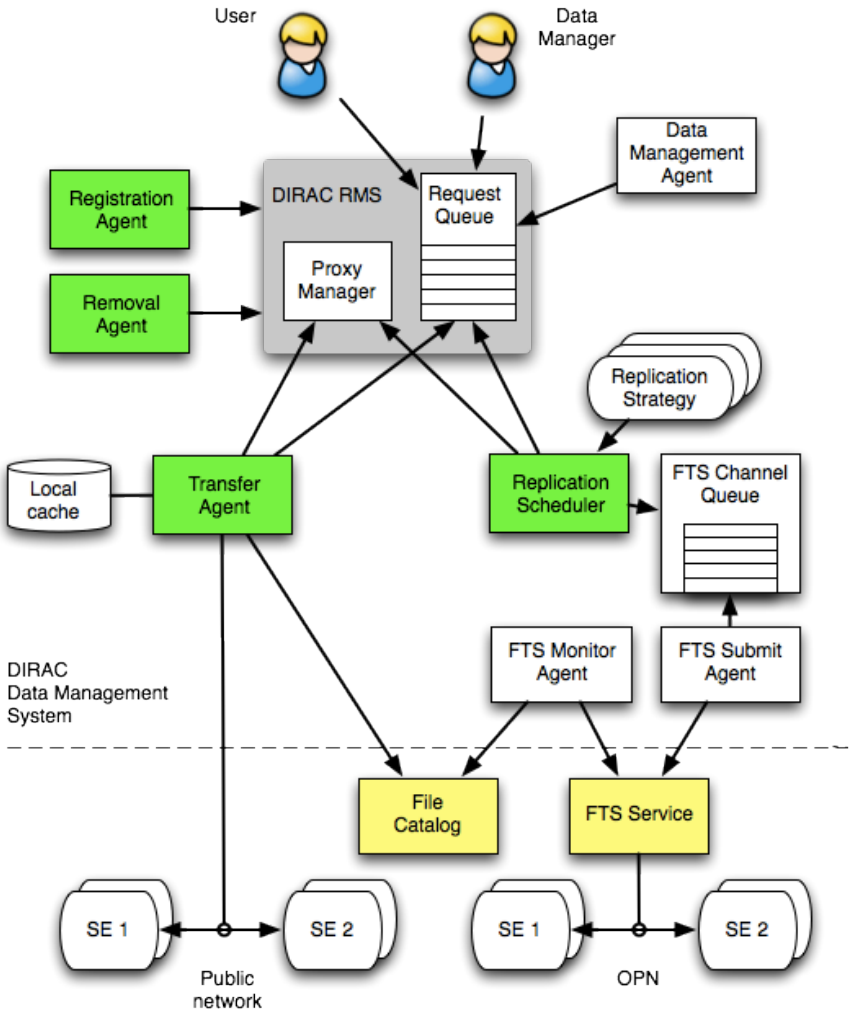
- Storage Elements
  - gLite/EGI Storage Elements
  - DIRAC Storage Elements
  - More Storage Elements can be included
    - (F,SF,HT,BBF)TP servers
- File Catalogs
  - LCG File Catalog (LFC)
  - DIRAC File Catalog
    - Support for the User Metadata (similar to the AMGA gLite service)
    - Support for data provenance
  - More Catalogs can be included
    - LHCb has developed several specific catalogs in the same framework

Data Management components

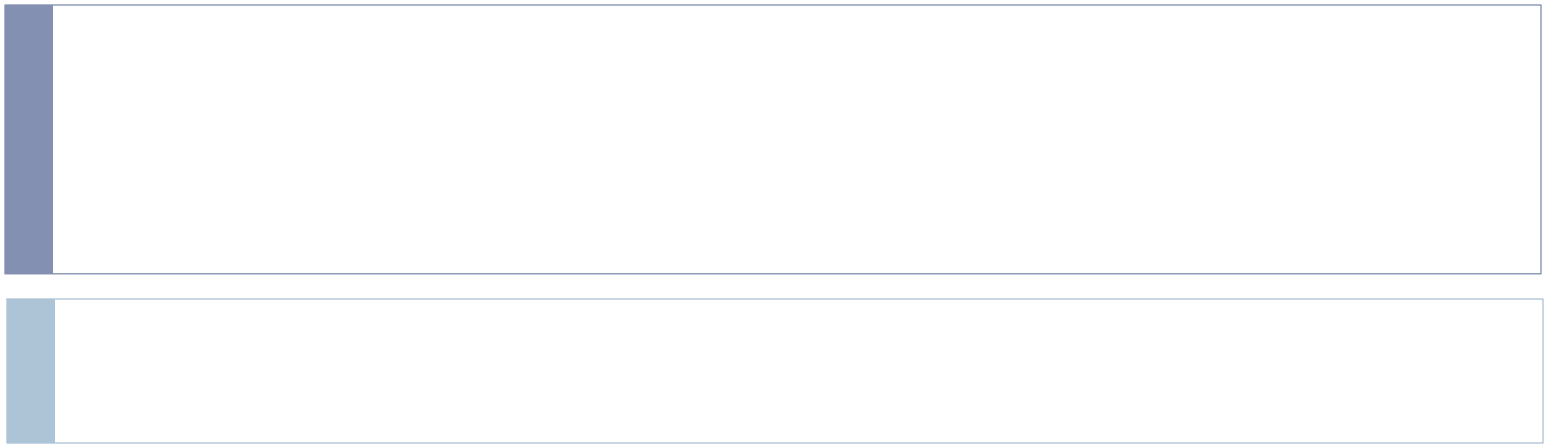
- For DIRAC users the use of any Storage Element or File Catalog is transparent
  - Community choice which components to use
  - Different SE types can be mixed together
  - Several File Catalogs can be used in parallel
    - Complementary functionality
    - Redundancy
- Users see depending on the DIRAC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

Data Management services

- 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



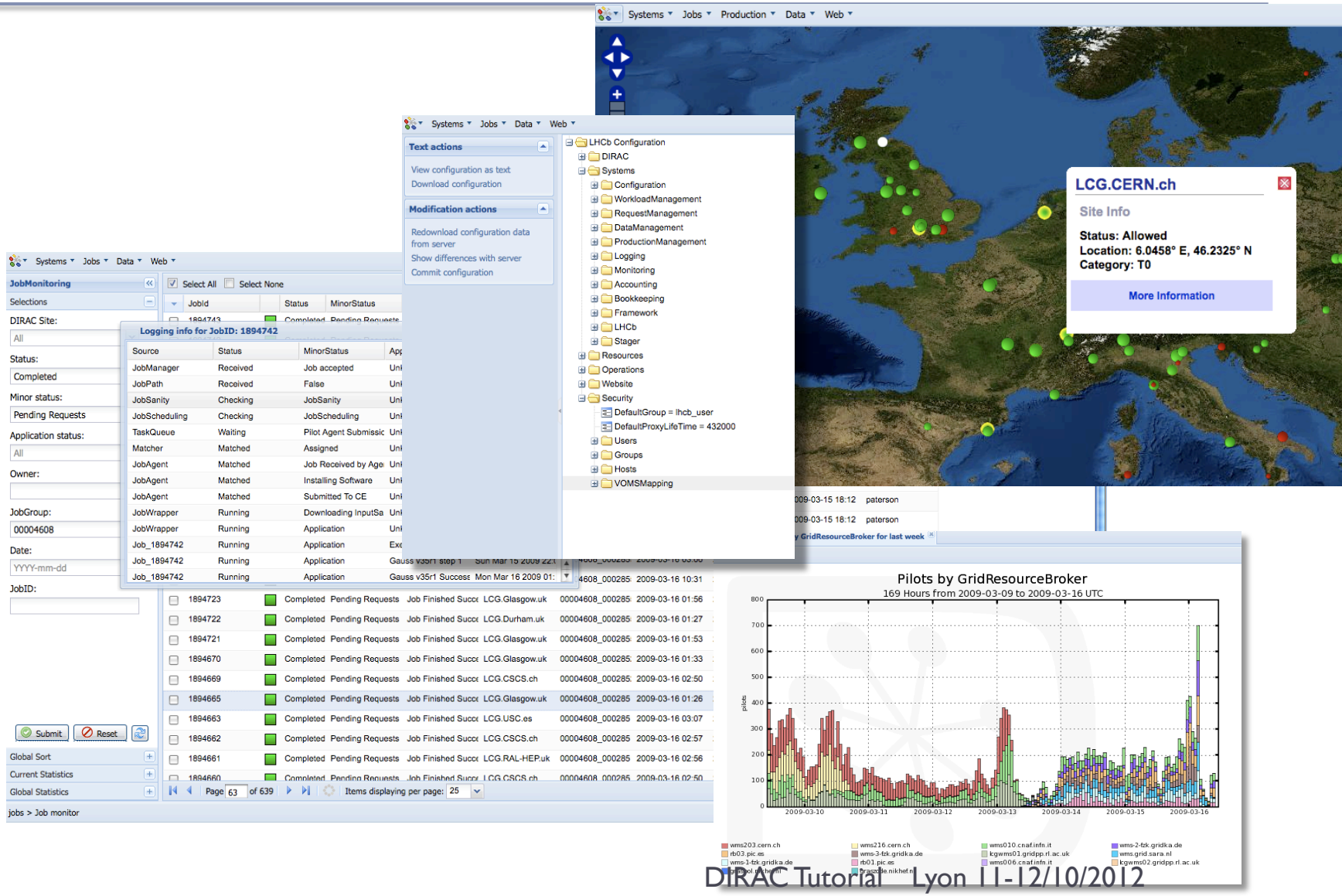
User Interfaces



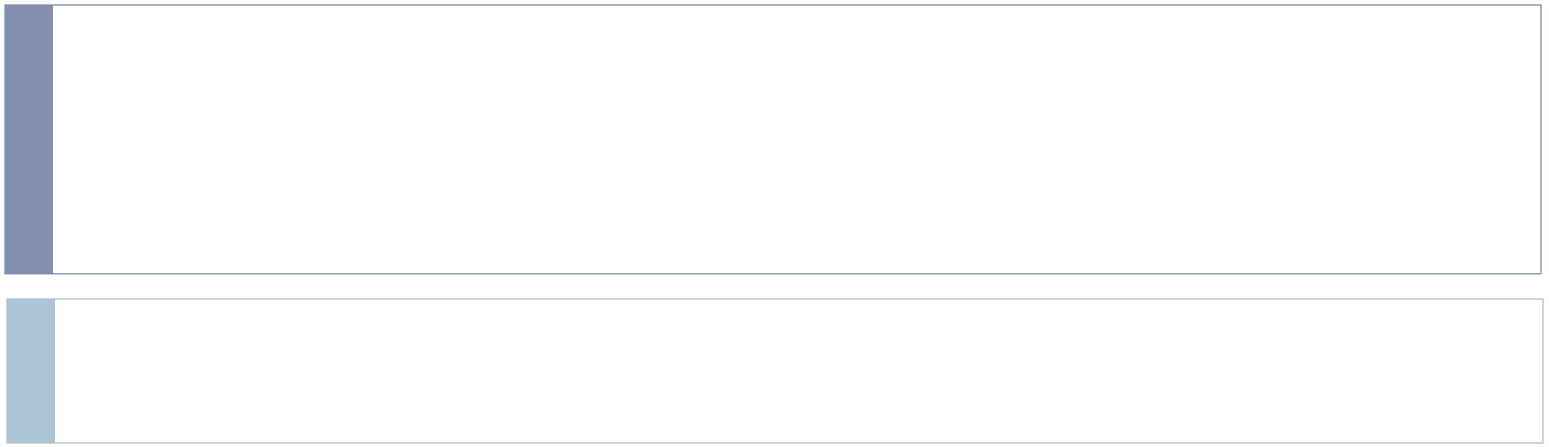
DIRAC: Secure Web Portal

- 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

Web Portal: example interfaces



DIRAC Framework



DIRAC middleware

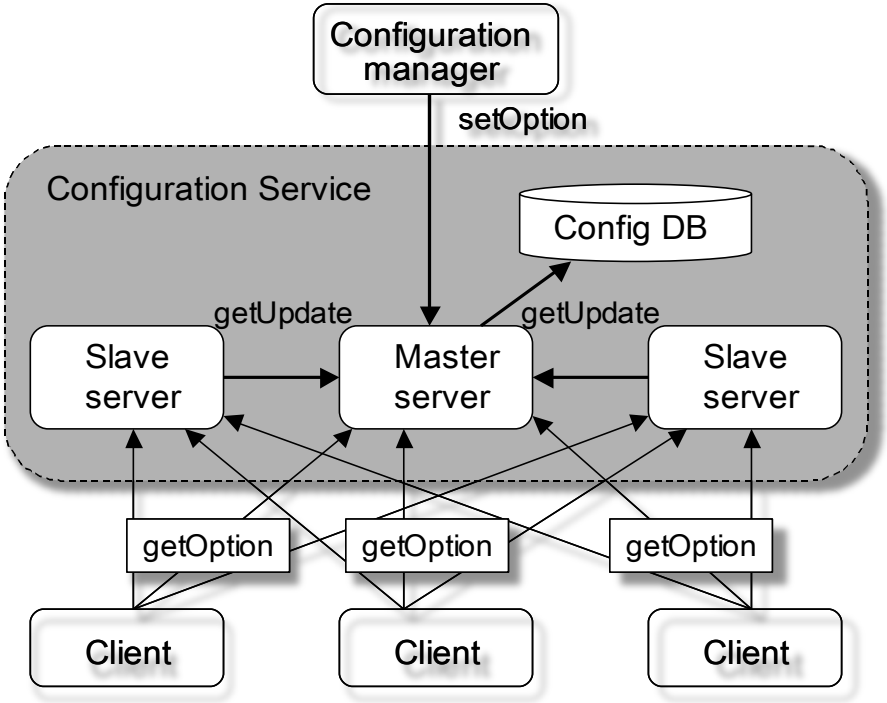
- Services oriented architecture (SOA)
- DIRAC has a well defined architecture
  - Services
    - passive components reacting to client request
    - Keep their state in a database
  - Light distributed agents
    - permanently running components, animating the whole system
  - Clients
    - User interfaces
    - Agent-service, service-service communications
- Technologies
  - Python, MySQL, OpenSSL

DIRAC Framework

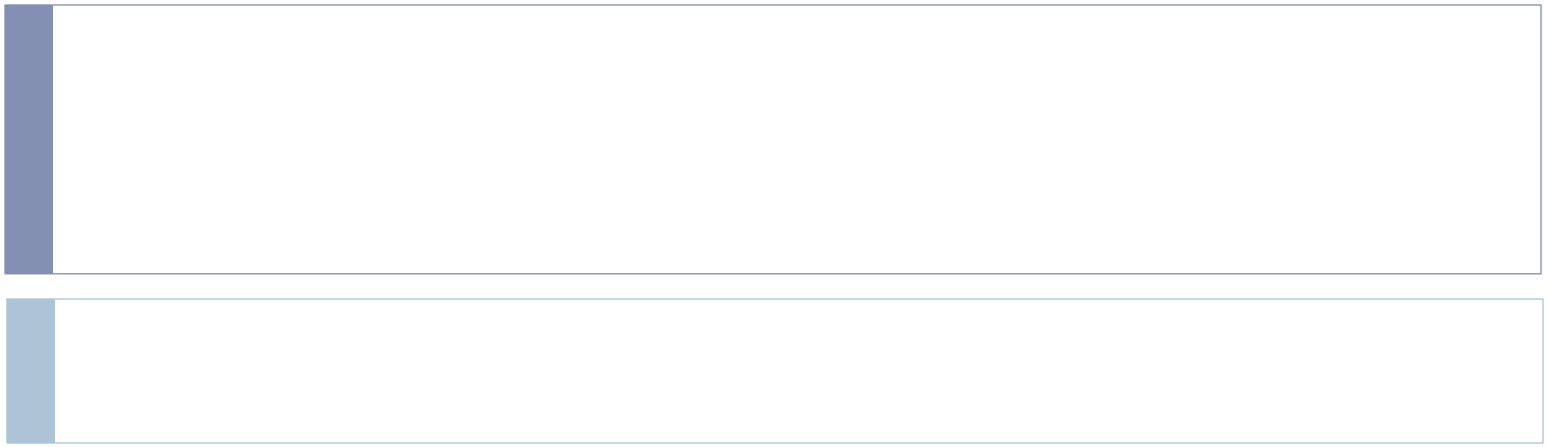
- All the communications between the distributed components are secure
  - DISET custom client/service protocol
    - Focus on efficiency
    - Control and data communications
  - X509, GSI security standards
  - Fine grained authorization rules
- Framework allows to easily build these components concentrating on the business logic of the applications
  - Making use of rich base services

DIRAC 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 as a Service

- DIRAC middleware is providing a complete grid functionality and is rather complex to install and manage
  - especially for small user communities with low grid expertise level.
- Several grid projects are providing now the DIRAC functionality as a ready to use service for their users
  - GISELA Latin America Grid,
  - France-Grilles, IberGrid NGIs
- Other grid infrastructure projects are considering provisioning DIRAC services

Summary

- DIRAC has most of the features of a "standard" Grid middleware stack

- Occasional users will not see much difference in functionality compared to other middlewares
  - Better efficiency and turnaround for intensive work
- 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

DIRAC Tutorial plan

- Getting Started
  - Getting ready user credentials
- Job execution mechanics
  - Basic job operations with Web Portal explained
- Job manipulation tools
  - Submission, monitoring, getting results
- Advanced job operations
  - Jobs with input and output data
  - Bulk job submission

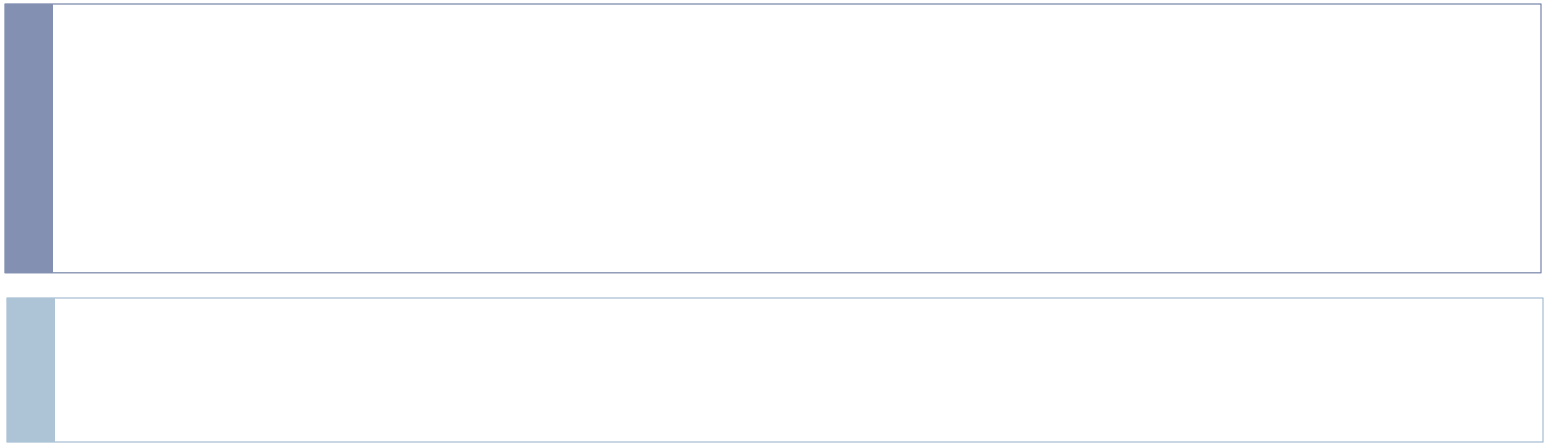
DIRAC Tutorial setup

- DIRAC installation at dirac.france-grilles.fr
  - The service used for the tutorial is permanent, will stay in place afterwards
- Resources
  - 21 EGI sites
  - 4 Ses
    - One DIRAC SE (DIRAC-USER)
    - 3 SRM EGI SEs
- Tutorial materials are available here
  - http://diracgrid.org/files/docs/UserGuide/Tutorials/

The course tutors team

- The course will be given by the members of the DIRAC Project team (http://diracgrid.org)

Backup slides



WMS: applying VO policies

- In DIRAC both User and Production jobs are treated by the same WMS
  - Same Task Queue
- This allows to apply efficiently policies for the whole VO
  - Assigning Job Priorities for different groups and activities
  - Static group priorities are used currently
  - More powerful scheduler can be plugged in
    - demonstrated with MAUI scheduler