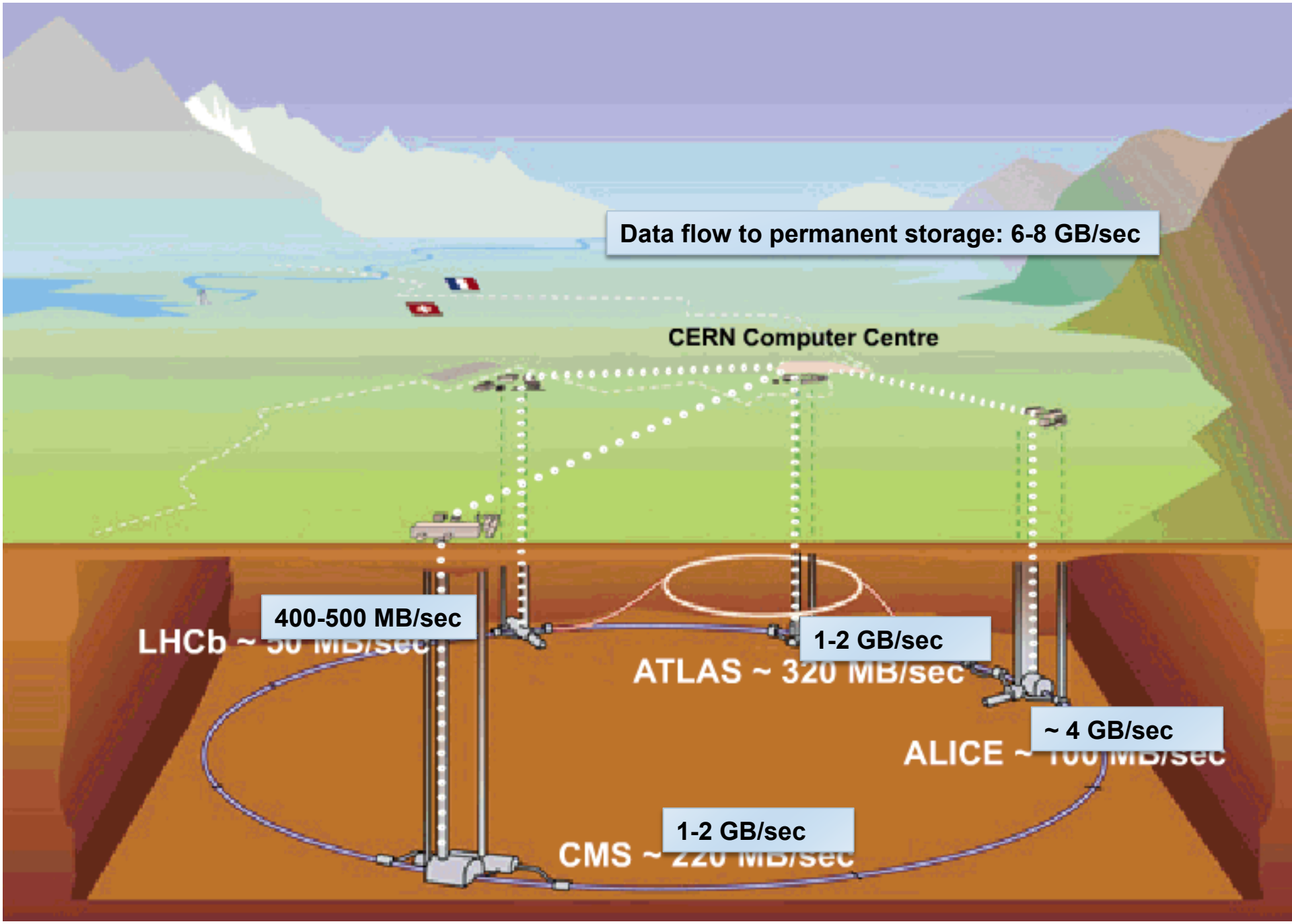
Distributed Infrastructure for Scientific Applications

A.Tsaregorodtsev,
Aix Marseille Univ, CNRS/IN2P3, CPPM
Réunion Big Data AMU, 3 Oct 2018

DIRAC
THE INTERWARE

Outline

- Large scale computations problem
- DIRAC Interware
  - WMS and computing resources
  - DMS and storage resources
  - Interfaces
  - Framework for distributed computing systems
- Conclusions


Data flow to permanent storage: 6-8 GB/sec
CERN Computer Centre
400-500 MB/sec
LHCb ~
1-2 GB/sec
ATLAS ~ 320 MB/sec
~ 4 GB/sec
ALICE ~
1-2 GB/sec
CMS ~

Massive computations: HEP

- LHC experiments pioneered the massive use of computational grids as a solution to the High Energy Physics Big Data problem
  - Many 10s of Petabytes of data per year
    - Total volume is approaching 1 Exabyte
  - Many 100s of thousands CPUs in 100s of centers
  - 10s of Gbyte/s data 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

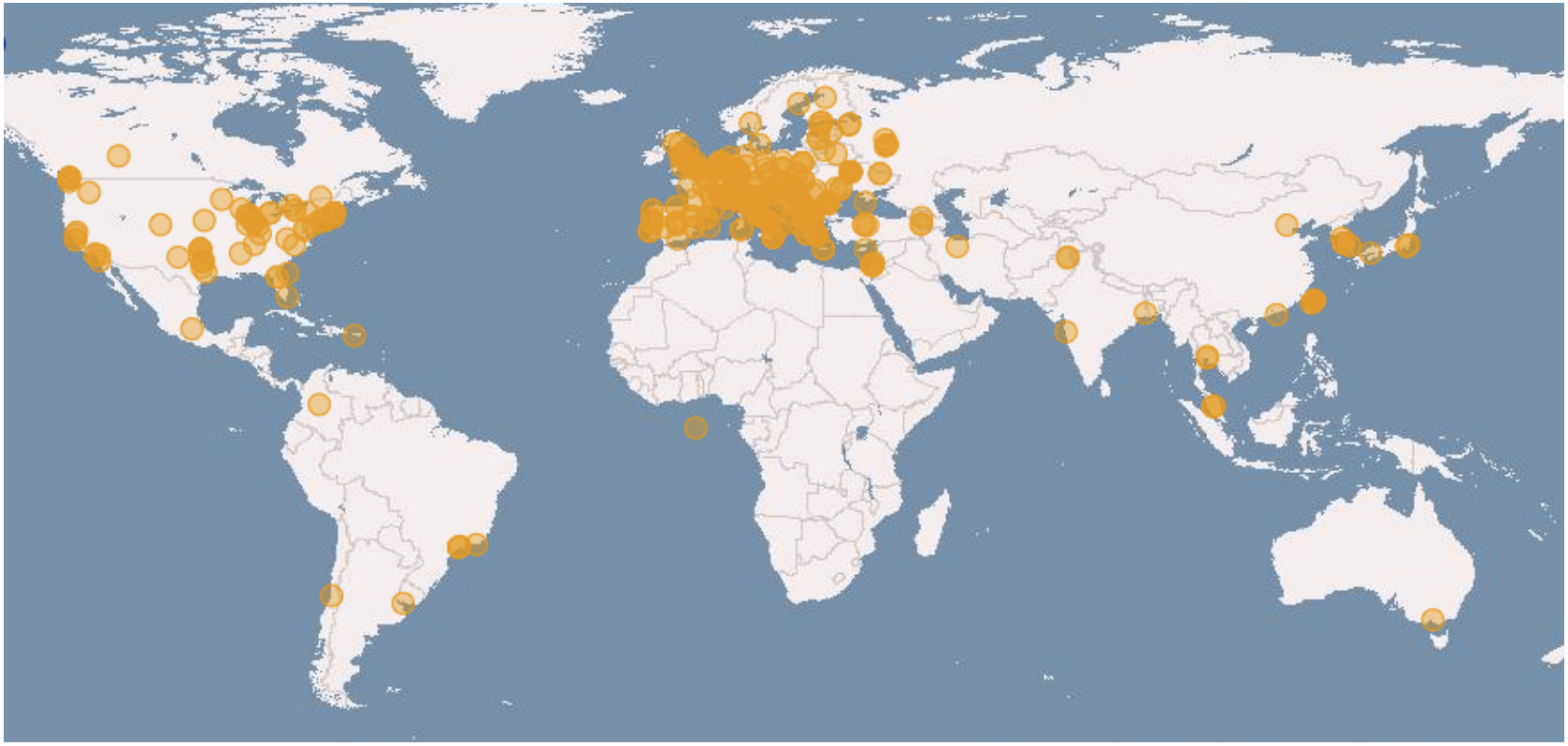
DIRAC
THE INTERWARE

Computing grids

- Grids are providing common infrastructure and rules to integrate multiple computing centers
  - Common computing element access protocols
  - Common user task scheduling system
  - Common policies and security rules
    - Users are signing up once and have access to multiple computing centers
  - Common monitoring of the user jobs
  - Common planning and accounting of computing resources
  - Common user training and support
  - Common problem tracking system
- Grids are making all the constituent computing centers to be seen as one entity for the users
- Now most of the computing power for LHC is provided by grid systems

DIRAC
THE INTERWARE

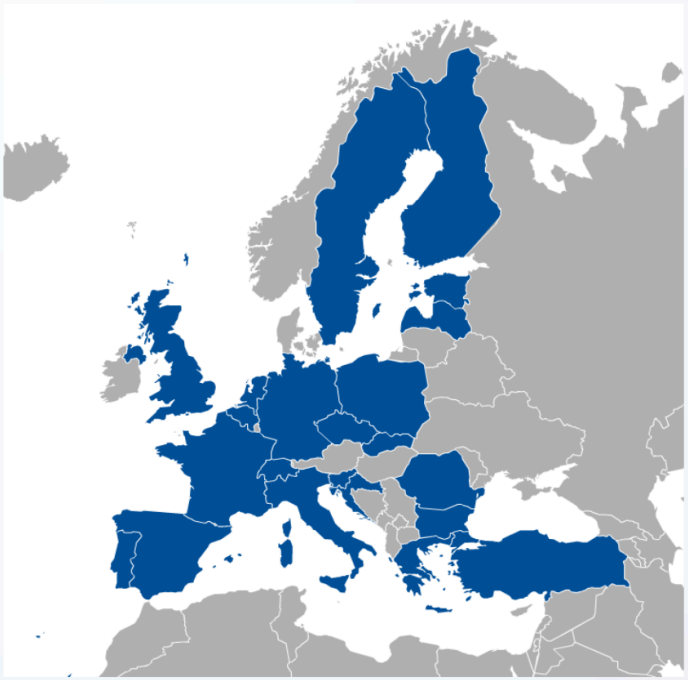
Worldwide LHC Computing Grid Collaboration (WLCG)



- >800 PB of data at CERN and major computing centers
- Distributed infrastructure of 150 computing centers in 40 countries
- 700+ k CPU cores (~ 5M HEP-SPEC-06 or 5 PetaFlops)
- The biggest site with ~50k CPU cores, 12 T1 centers with 2-30k CPU cores
- Distributed data, services and operation infrastructure

European Grid Infrastructure EGI

- 22 national infrastructures
- 2 international scientific organizations: CERN и EMBL



www.egi.eu/about/egi-foundation/

EGI Resources

~750.000 CPU cores

~565PB of storage

1.7 Million jobs/day

2.6 billion CPU hours/year (+26%)

~50K users (+25%)

Virtualized Computing Resources

- Cloud technology allows to offer computing resources on demand according to the user specification and budget
  - Pay for what is consumed
  - Widely used as a platform for user defined services (PaaS)
  - Application portals (SaaS)
  - Amazon EC2 Cloud pioneered the field
- Since recently computing centers started to provide their resources using cloud technologies
  - Virtualized computing infrastructures, IaaS
  - Very flexible – provides resources adapted to the user needs
    - Operating systems, memory, CPU power, etc.
- However, large scientific collaborations can have access to multiple computational clouds
  - Dealing with independent clouds separately requires a huge management effort
  - Federating multiple clouds into a single coherent system is necessary to provide a transparent access for users.
    - Analogous to the grid infrastructures

DIRAC
THE INTERWARE

Other resources

- Standalone computing clusters not included in any grid infrastructure
  - Resources available at Universities and scientific laboratories
- High Performance Computing (HPC) Centers, or Supercomputers
  - Computing centers oriented towards massively parallel applications using specialized hardware
- Volunteer Computing
  - Mostly based on BOINC technology
  - SETI@Home, LHC@Home, etc

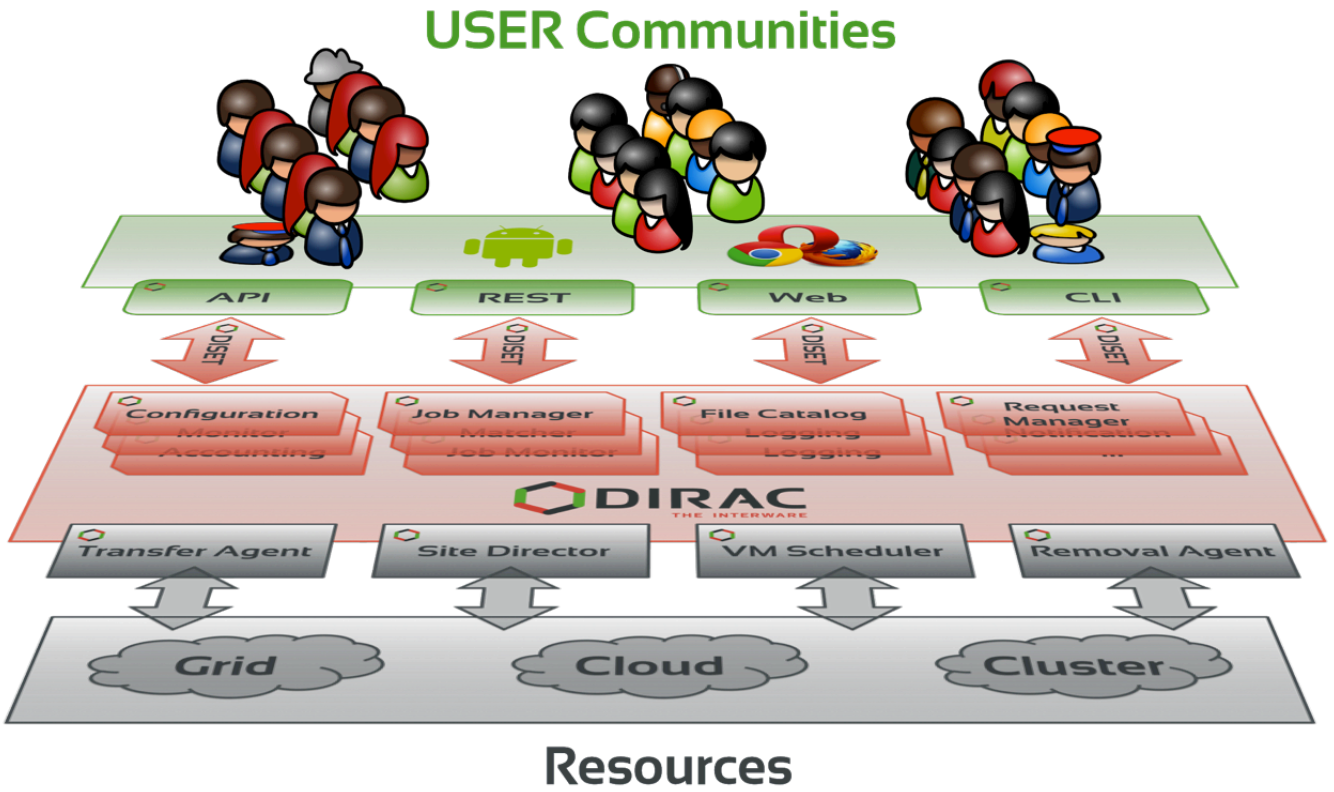
DIRAC
THE INTERWARE

Large scientific communities problems

- Large international scientific communities have divers contributions to their common computing resources
  - Different technologies, geographical and administrative domains
- Large communities have a complex internal structure with different user groups and activities having different priorities
- For resources providers it should be easy to incorporate their centers
  - There should be minimal requirements for installed software, configuration, maintenance
- Standard grid middleware based scheduling systems do not scale to the necessary amount of resources
  - More scalable solution is needed

DIRAC: the interware

- A software framework for distributed computing
- A **complete** solution to one (or more) user community
- Builds a layer between users and resources



… a few examples of what DIRAC can be used for

- sending jobs to "the Grid"
  - the obvious one
- interfacing with different sites
  - with different computing elements
    - and batch systems
  - with different storage elements
- interfacing with different information systems

- managing productions
- managing dataset transfers
  - and removals…
- providing a failover system
  - your jobs won't fail because a certain SE is down, nor because of central service are down
- transfer data from the experiment to a Grid SE
- … and more

An open source project

- Started as an LHCb project, became experiment-agnostic in 2009
  - First users (after LHCb) end of 2009
- Developed by communities, for communities
  - Open source (GPL3+), GitHub hosted, python 2.7
  - No dedicated funding for the development of the "Vanilla" project
  - Publicly documented, active assistance forum, yearly users workshops, open developers meetings
  - 4 FTE as core developers, a dozen contributing developers
- The DIRAC consortium as representing body
  - CNRS, CERN, University of Barcelona
  - IHEP, KEK, PNNL, University of Montpellier

Users/communities/VOs

A *framework* shared by multiple experiments/ projects, both inside HEP, astronomy, and life science

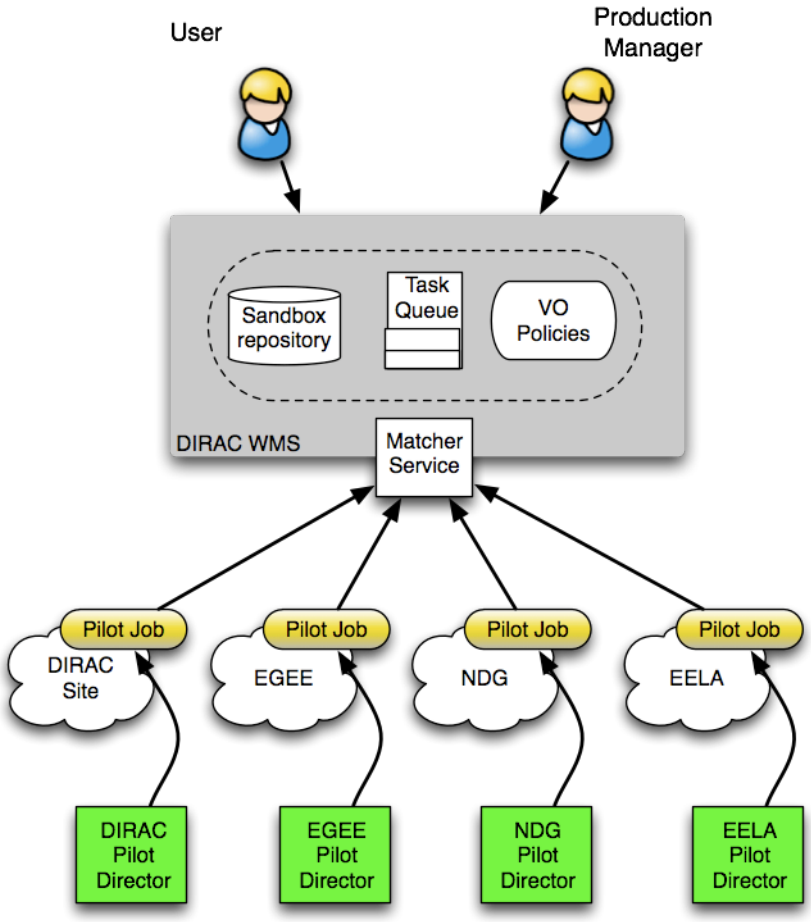
<u>Experiment agnostic</u>
<u>Extensible</u>
<u>Flexible</u>

DIRAC
THE INTERWARE

Workload Management

Job scheduling

- Pilot jobs are submitted to computing resources by specialized Pilot Directors
- After the start, Pilots check the execution environment and form the resource description
  - OS, capacity, disk space, software, etc
- The resources description is presented to the Matcher service, which chooses the most appropriate user job from the Task Queue
- The user job description is delivered to the pilot, which prepares its execution environment and executes the user application
- In the end, the pilot is uploading the results and output data to a predefined destination

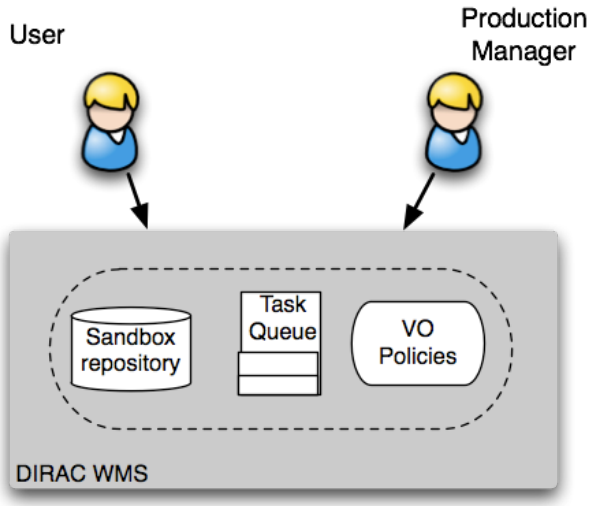


Advantages of pilot based scheduling

- User jobs only start running after the execution environment is validated
  - The risk of user job failures is considerably reduced compared to direct user job submission
- In case of a constant inflow of user jobs, computing resources are guaranteed to be fully exploited
  - Computing centers actively looking for payload to execute
- Resource providers have to deal with less number of users
  - Pilot identity represents a whole user community
  - Community management is moved to the DIRAC level

WMS: applying VO policies

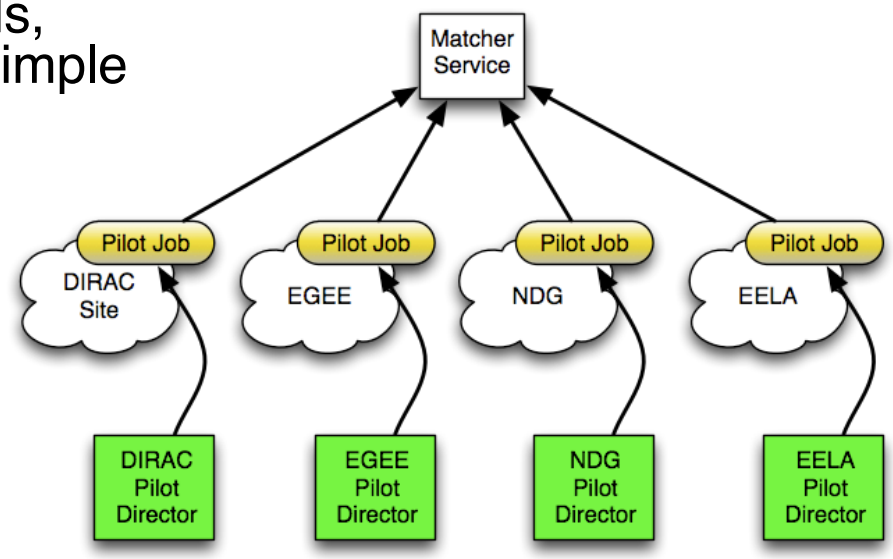
- In DIRAC jobs from all the users are treated by the same WMS
  - Same Task Queue
  - All the policies are applied in one place
- This allows to apply efficiently policies for the whole VO
  - Assigning Job Priorities for different groups and activities



- After a user job is received by the pilot it passes to execution immediately
  - Policies application is precise
- No need to apply special rules by the resources providers
  - Although they still can do

WMS: using heterogeneous resources

- Pilot based Workload Management provides abstraction of Computing Resources
  - Allows to combine heterogeneous resources in a transparent way
- Including resources in different grids, clouds and standalone clusters is simple with Pilot Jobs
  - Needs a specialized Pilot Director per resource type
  - Users just see new logical sites appearing
- Similar patterns are applied also for the Data Management System of DIRAC (*see below*)



DIRAC
THE INTERWARE

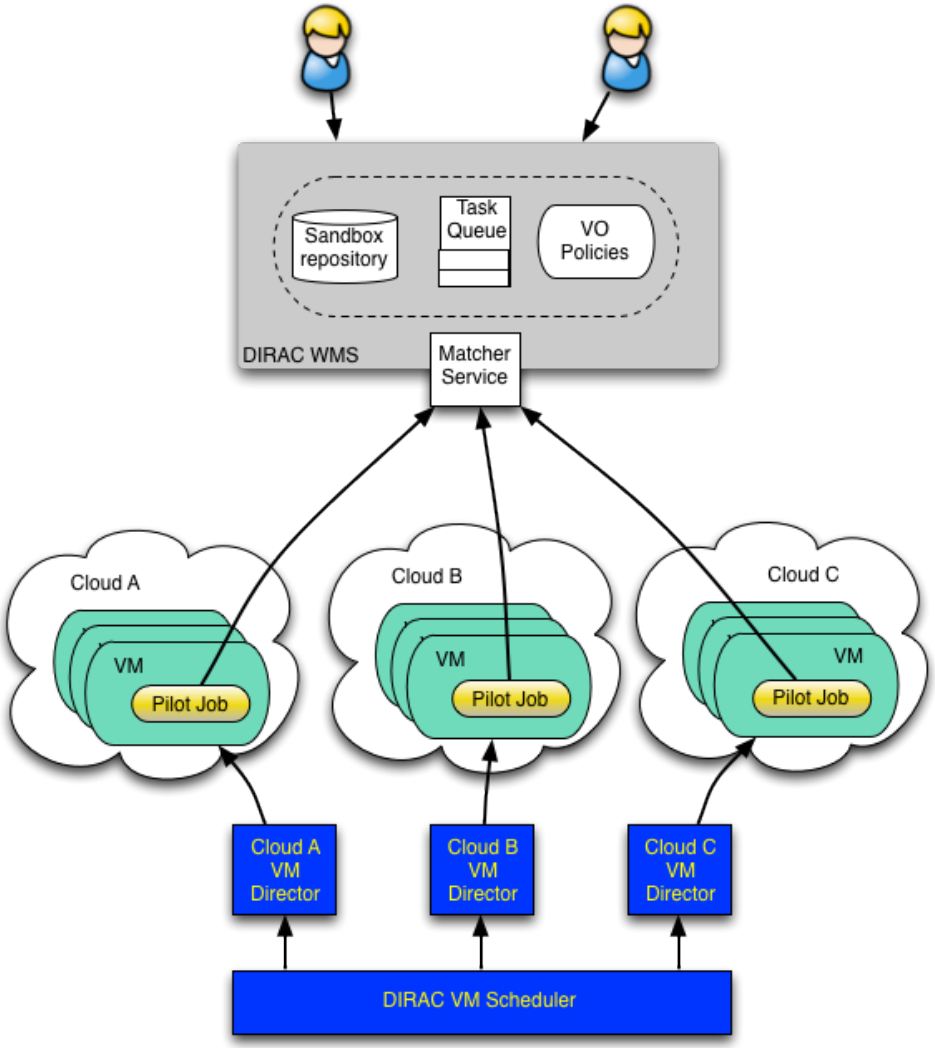
DIRAC computing resources

Computing Grids

- DIRAC was initially developed with the focus on accessing conventional Grid computing resources
  - WLCG grid resources for the LHCb Collaboration
- It fully supports different middleware based grids
  - European Grid Infrastructure (EGI), WLCG, etc
    - DIRAC is an officially supported WMS service for the EGI infrastructure
  - Northern American Open Science Grid (OSG)
    - Using VDT middleware
      - Direct submission to computing elements
  - Northern European Grid (NDGF)
    - Using ARC middleware
      - Direct submission to computing elements
- Other types of grids can be supported
  - As long we have customers needing that

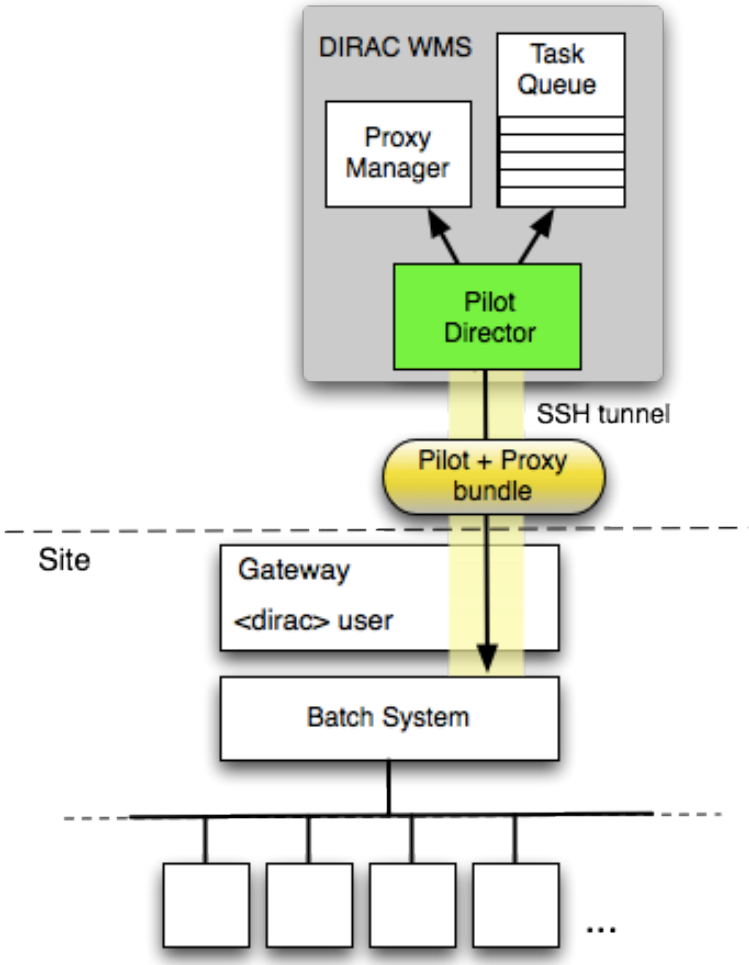
Clouds

- VM scheduler
  - Dynamic VM spawning taking Task Queue state into account
  - Discarding VMs automatically when no more needed
- The DIRAC VM scheduler by means of dedicated VM Directors is interfaced to
  - OCCI compliant clouds:
    - OpenStack, OpenNebula
  - Amazon EC2
  - …



Standalone computing clusters

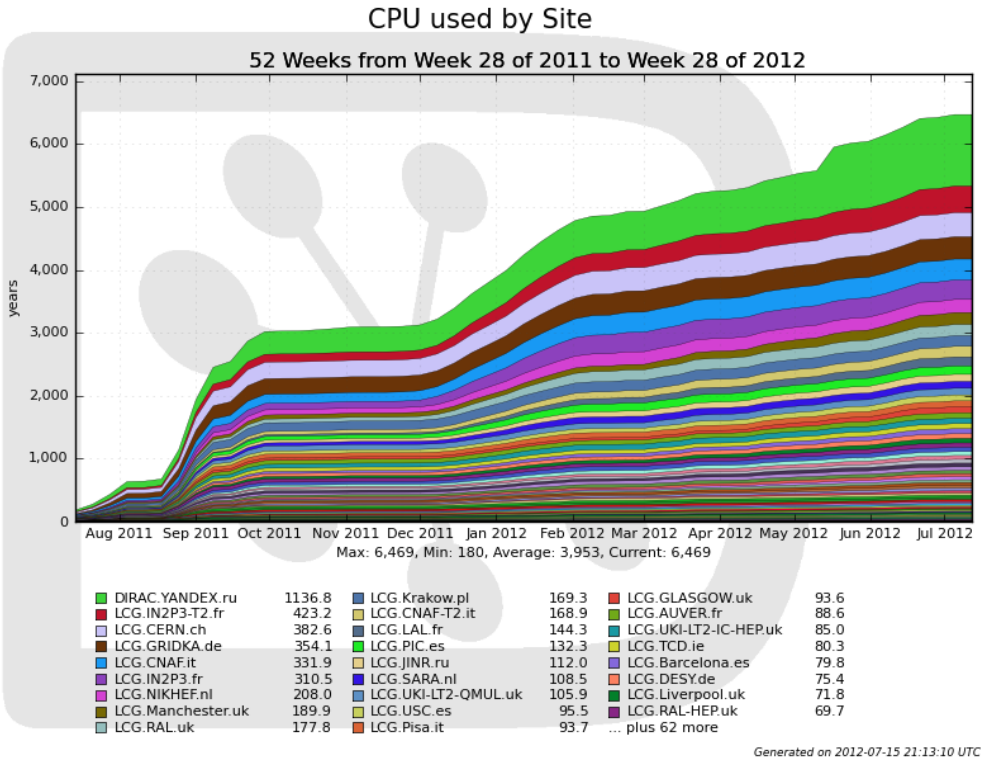
- Off-site Pilot Director
  - Site delegates control to the central service
  - Site must only define a dedicated local user account
  - The payload submission through an SSH/GSISSH tunnel
- The site can be:
  - a single computer or several computers without any batch system
  - a computing cluster with a batch system
    - LSF, BQS, SGE, PBS/Torque, Condor
      - Commodity computer farms
    - OAR, SLURM
      - HPC centers
- The user payload is executed with the owner credentials
  - No security compromises with respect to external services



Standalone computing clusters

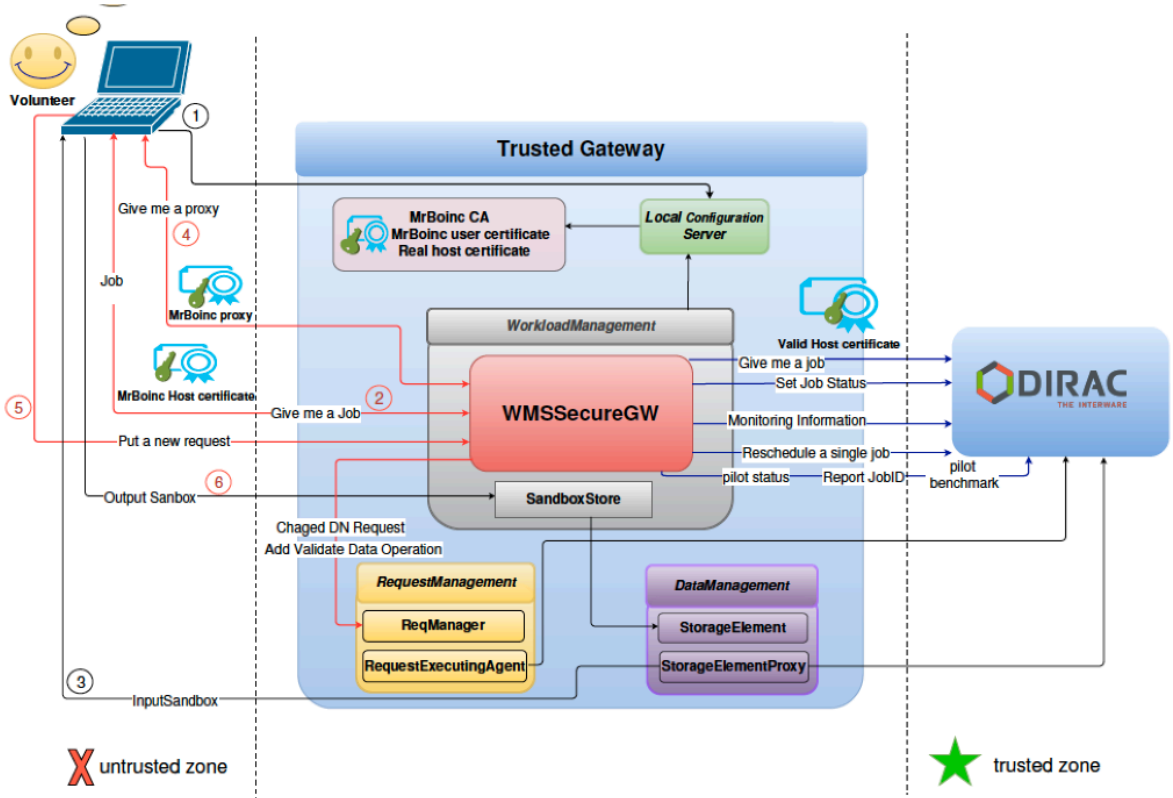
Examples:

- DIRAC.Yandex.ru
  - >2000 cores
  - Torque batch system, no grid middleware, access by SSH
  - Second largest LHCb MC production site
- LRZ Computing Center, Munich
  - SLURM batch system, GRAM5 CE service
  - Gateway access by GSISSH
  - Considerable resources for biomed community (work in progress)
- Mesocentre Aix-Marseille University
  - OAR batch system, no grid middleware, access by SSH
  - Open to multiple communities (work in progress)



Volunteer grids

- Solution to the trust problem: separate trusted and untrusted worlds
  - Put a gateway in between to ensure communication
    - Use temporary certificates in the untrusted machines, communicate with a real host certificate to DIRAC service
    - Validate any output of the jobs before uploading to the final destination



DIRAC
THE INTERWARE

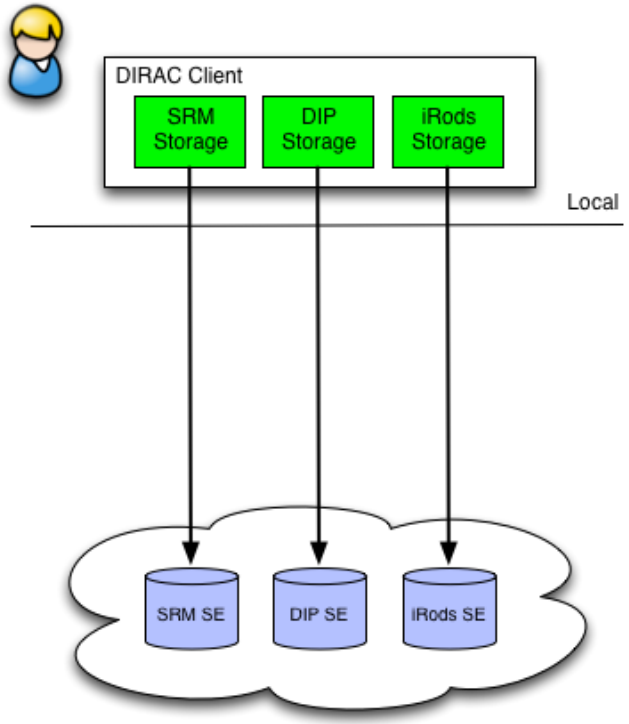
Data Management

DM Problem to solve

- There are many different formats in which data can be stored
  - Structured and non-structured databases, objects stores, file systems, etc
- DM problem addressed By DIRAC
  - Data is partitioned in files
  - File replicas are distributed over a number of Storage Elements world wide
  - Data Management tasks
    - Initial File upload
    - Catalog registration
    - File replication
    - File access/download
    - Integrity checking
    - File removal
  - Need for transparent file access for users
  - Often working with multiple (tens of thousands) files at a time
    - Make sure that ALL the elementary operations are accomplished
    - Automate recurrent operations

Storage plugins

- Storage element abstraction with a client implementation for each access protocol
  - DIPS – DIRAC data transfer protocol
  - FTP, HTTP, WebDAV
  - SRM, XROOTD, RFIO, DCAP, etc
    - HEP centers specific protocols
    - Using gfal2 library developed at CERN
  - S3, Swift, CDMI: cloud specific data access protocols
- Like with CE's, each SE is seen by the clients as a logical entity
  - With some specific operational properties
    - Archive, limited access, etc
  - SE's can be configured with multiple protocols
- Including new data access technologies requires creating new specific plug-in



File Catalog Service

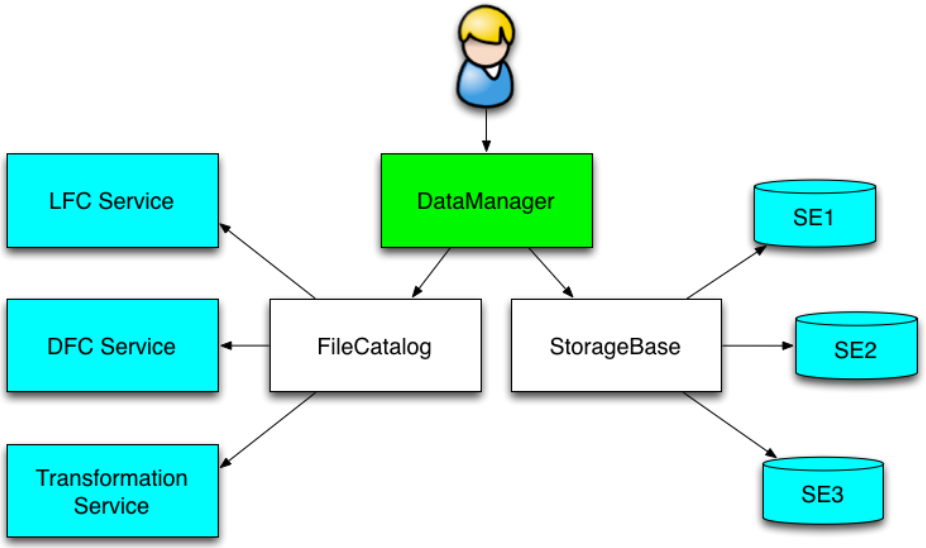
- File Catalog is a service to keep track of all the physical file replicas in all the SE's
  - Stores also file properties:
    - Size, creation/modification time stamps, ownership, checksums
    - User ACLs
- DIRAC relies on a *central* File Catalog
  - Defines a single logical name space for all the managed data
  - Organizes files hierarchically like in common file systems
  - Other projects, e.g. distributed file systems, keep file data in multiple distributed databases
    - More scalable
    - Maintaining data integrity is very difficult

File Catalog Interware

- DIRAC, as for other components, defines an abstraction of a File Catalog service with several implementations
  - LCG File Catalog (LFC) – de facto standard grid catalog (obsoleted)
  - File Catalog implementation by the DIRAC Project itself
  - Specialized catalogs, e.g. LHCb Bookkeeping service
- Several catalogs can be used together
  - The mechanism is used to send messages to "pseudocatalog" services:
    - Transformation service (see later)
    - Community specific services
  - A user sees it as a single catalog with additional features

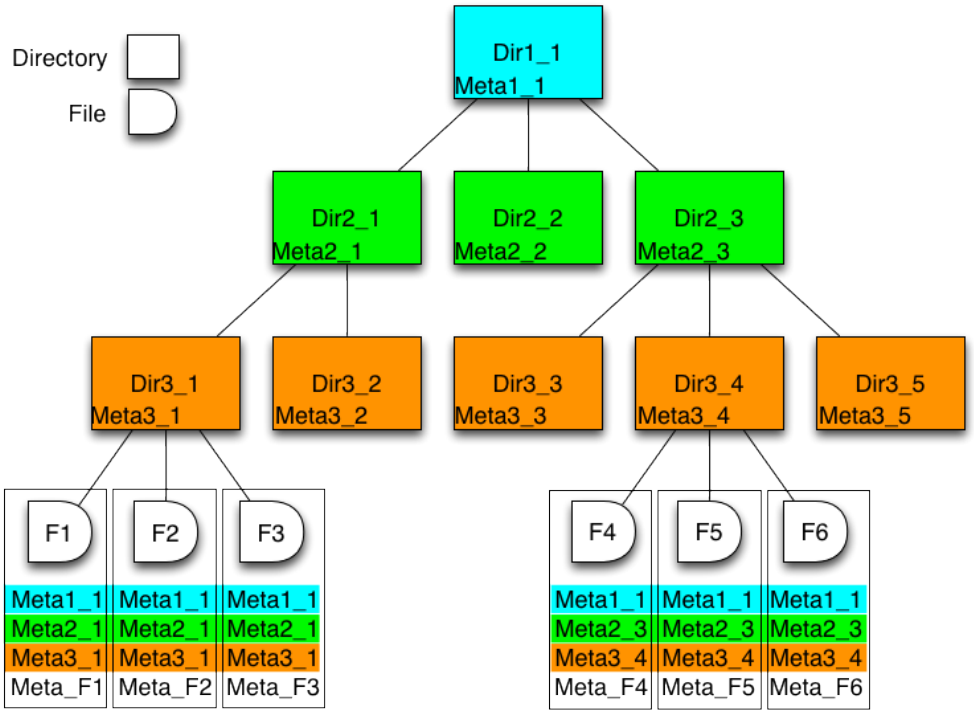
Combined data API

- Together with the data access components DFC allows to present data to users as a single global file system
  - Can be even mounted as a file system partition on a user computer (FSDIRAC project)
- DataManager API is a single client interface for logical data operations



File Catalog: Metadata

- DFC is Replica and Metadata Catalog
  - User defined metadata
  - The same hierarchy for metadata as for the logical name space
    - Metadata associated with files and directories
    - Allow for efficient searches
  - Efficient Storage Usage reports
    - Suitable for user quotas



- Example query:
  - `find /lhcb/mcdata LastAccess < 01-01-2012 GaussVersion=v1,v2 SE=IN2P3,CERN Name=*.raw`
- Result of file search is a precise list of corresponding files
  - Unlike Google index

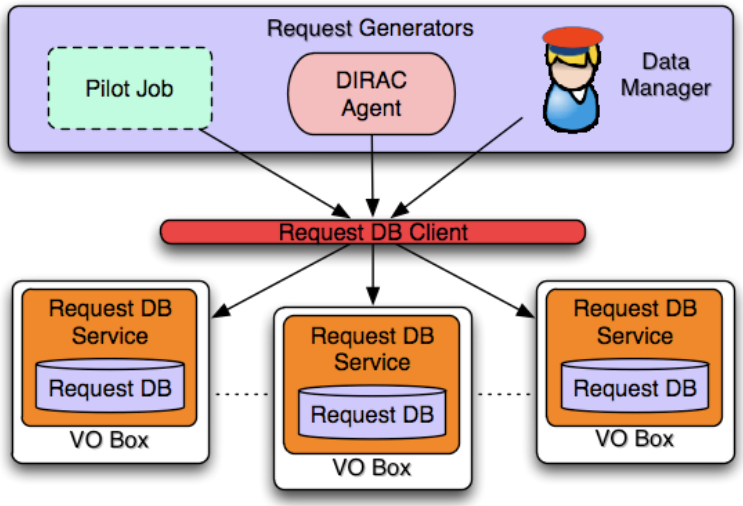
DIRAC
THE INTERWARE

Massive data operations

- DIRAC is dealing with large volumes of scientific data
  - 10's of Petabytes
  - 10^7-10^8 of files and directories
- There is a need for massive (bulk) operations
  - Examples:
    - Replicate 10^5 files from SE A to SE B
    - Remove 10^5 files and all their replicas in all the storages
- Massive data operations require
  - Asynchronous execution
  - Automatic failure recovery
  - Data integrity checking

Request Management system for asynchronous operations

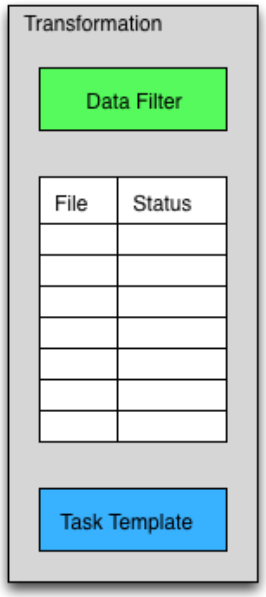
- Request Management System (RMS) receives and executes asynchronously requests for any kind of operation
  - Data upload and registration
  - Job status and parameter reports
- RMS is used heavily as part of the failure recovery procedure
  - Any operation that can fail can be deferred to the RMS system

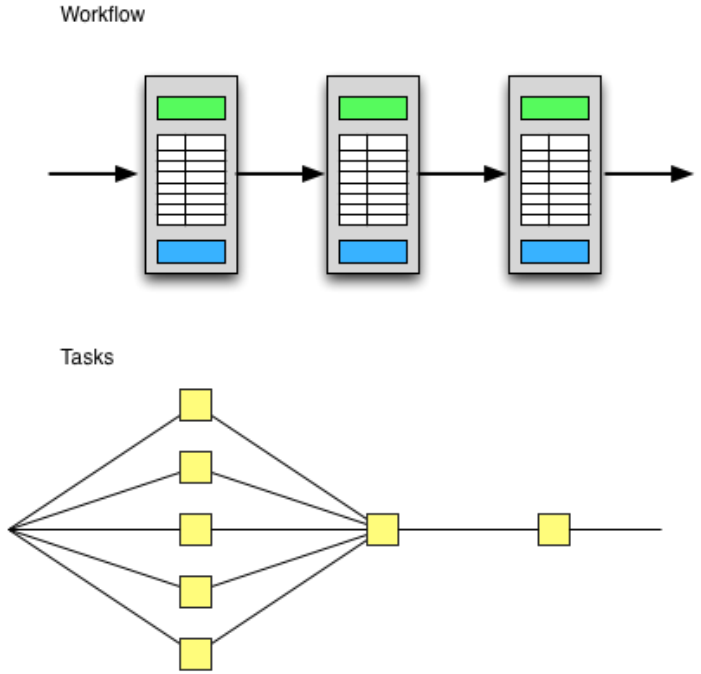


- Requests are collected by RMS instances at geographically distributed sites
  - Extra redundancy in RMS service availability
- Requests are forwarded to the central Request Database
  - For keeping track of the pending requests
  - For efficient bulk request execution
- RequestExecution agents execute the stored requests
  - With multiple retries if necessary until the operation is successful

Transformation System for data driven workflows

- Data driven workflows as chains of data transformations
  - Transformation: input data filter + recipe to create tasks
  - Tasks are created as soon as data with required properties is registered into the system
  - Tasks: jobs, data replication, etc
- Transformations can be used for automatic data driven bulk data operations
  - Scheduling RMS tasks
  - Often as part of a more general workflow





Other data services

- Data logging
  - Keeping a history of all operation for a given file
- Data provenance
  - Keeping ancestor-descendant relations for each file
- Data integrity
  - Collecting reports on all the data access failures
  - Automated data recovery and validation
- Storage usage reports
  - Storage resources consumption at any moment
    - Help Data Managers
    - Allow to impose user quotas
- Accounting
  - Storage consumption
  - Data transfer traffic and error rates

DIRAC
THE INTERWARE

Interfaces

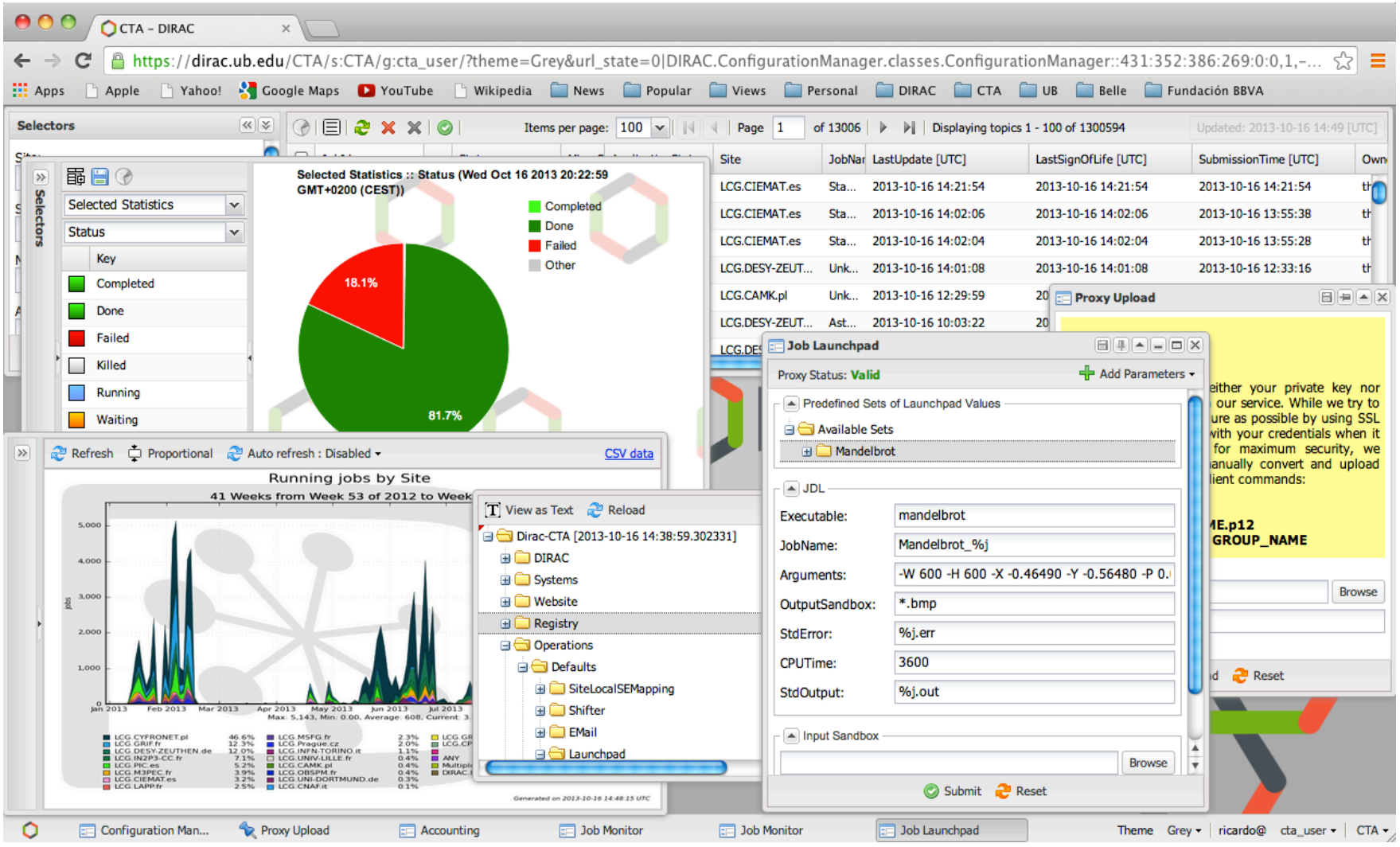
DM interfaces

- Command line tools
  - Multiple dirac-dms-… commands
- COMDIRAC
  - Representing the logical DIRAC file namespace as a parallel shell
  - **dls, dcd, dpwd, dfind, ddu** etc commands
  - **dput, dget, drepl** for file upload/download/replication
- REST interface
  - Suitable for use with application portals
  - Multiple application portals are interfaced to DIRAC this way

Web Portal

- Desktop paradigm for the DIRAC Web interface
  - Intuitive for most of the users

Web Portal applications



DIRAC
THE INTERWARE

Distributed Computer

- DIRAC is aiming at providing an abstraction of a single computer for massive computational and data operations from the user perspective
  - Logical Computing and Storage elements (Hardware)
  - Global logical name space (File System)
  - Desktop-like GUI
- Applications can be interfaced with graphical user interfaces in the DIRAC Web Portal framework

DIRAC
THE INTERWARE

DIRAC Framework

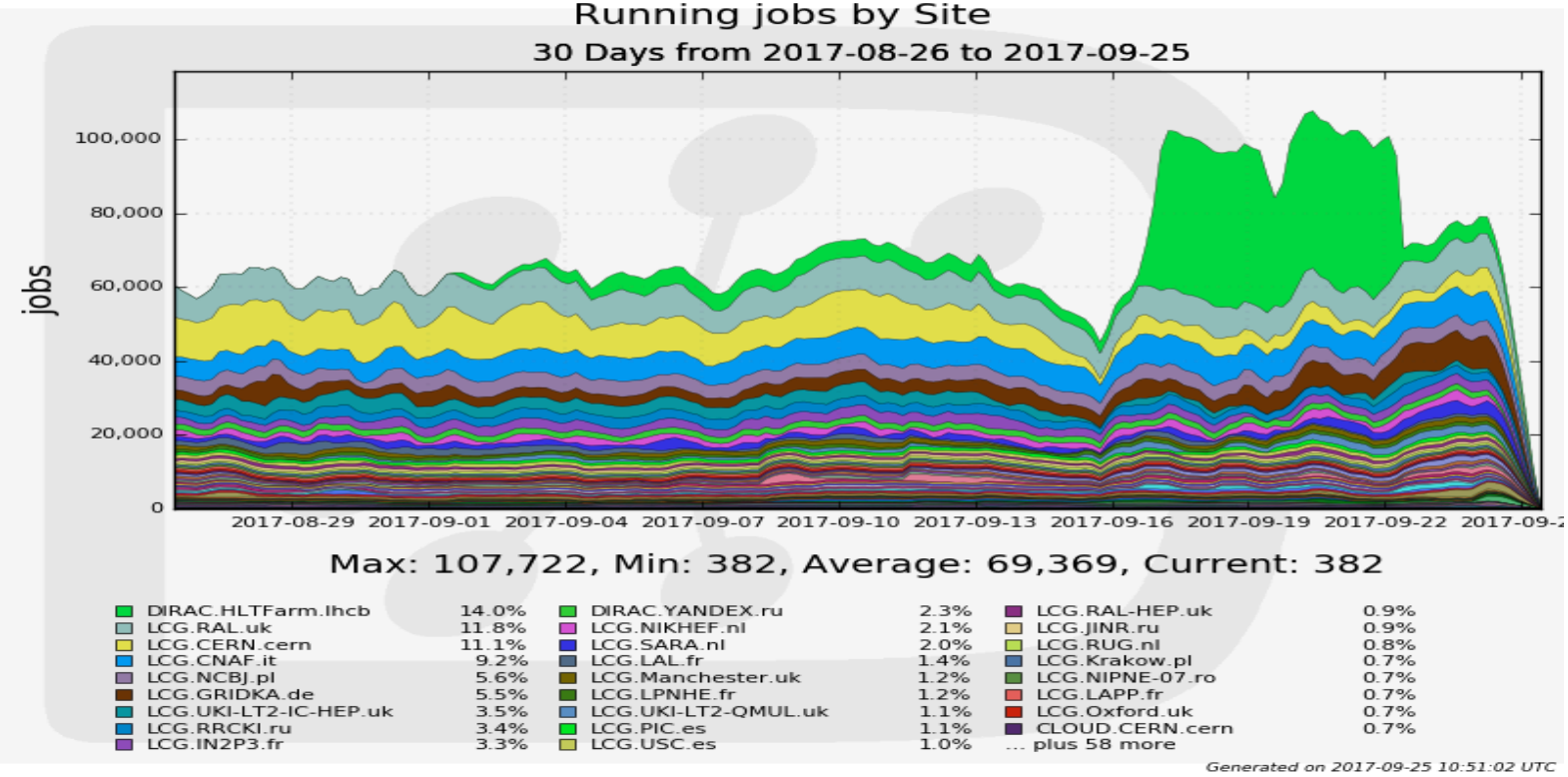
DIRAC Framework

- All the communications between the distributed components are secure
  - DISET custom client/service protocol
    - Focus on efficiency
    - Control and data transfer communications
    - HTTP based protocol is studied for further DIRAC evolution
  - X509, GSI security standards
    - Users and services are provided with digital certificates
    - User certificate proxies (passwordless, time limited temporary certificate copies) are used for distributed operations on the users's behalf
  - Fine grained service access authorization rules
- SSO authentication is in the works
  - EGI Check-In solution

DIRAC
THE INTERWARE

DIRAC users

LHCb Collaboration

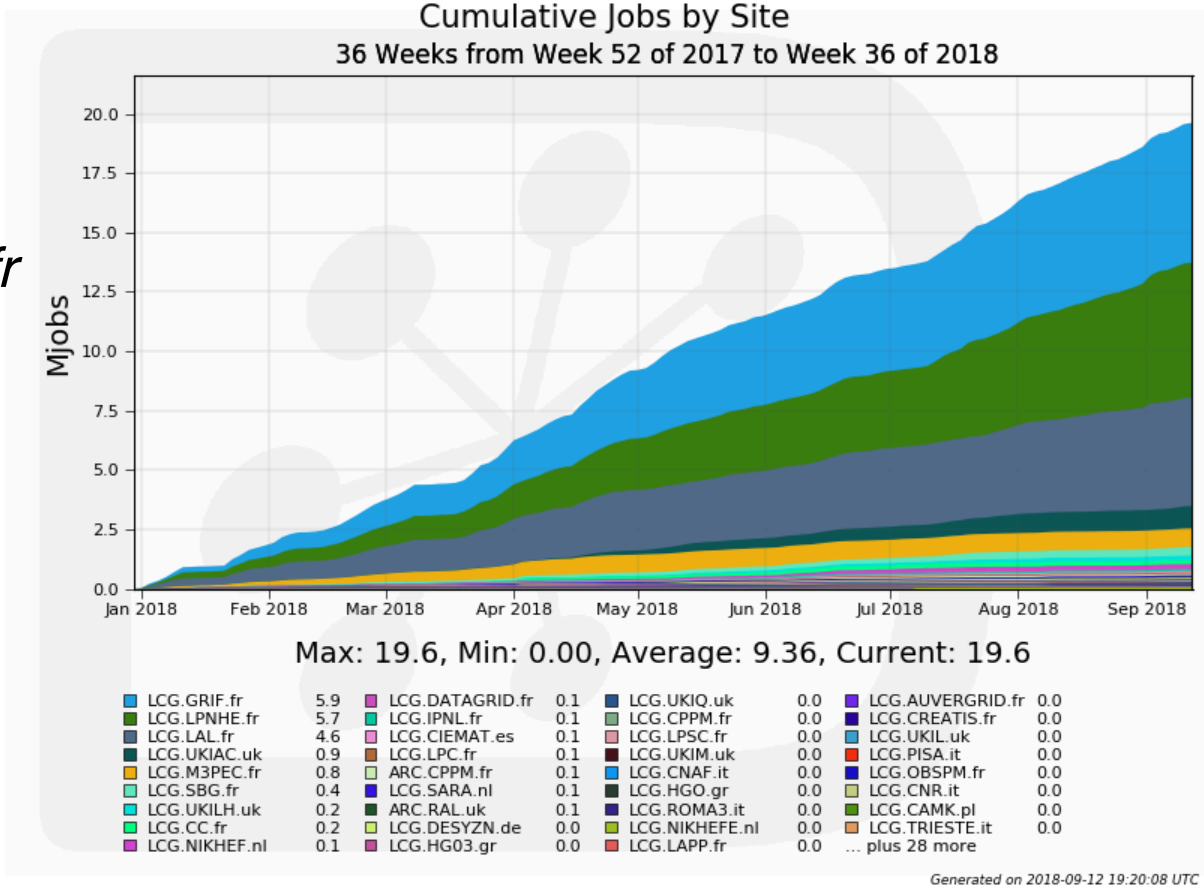


- More than 100K concurrent jobs in ~120 distinct sites
  - Limited by available resources, not by the system capacity
- Further optimizations to increase the capacity are possible
  - Hardware, database optimizations, service load balancing, etc

National services

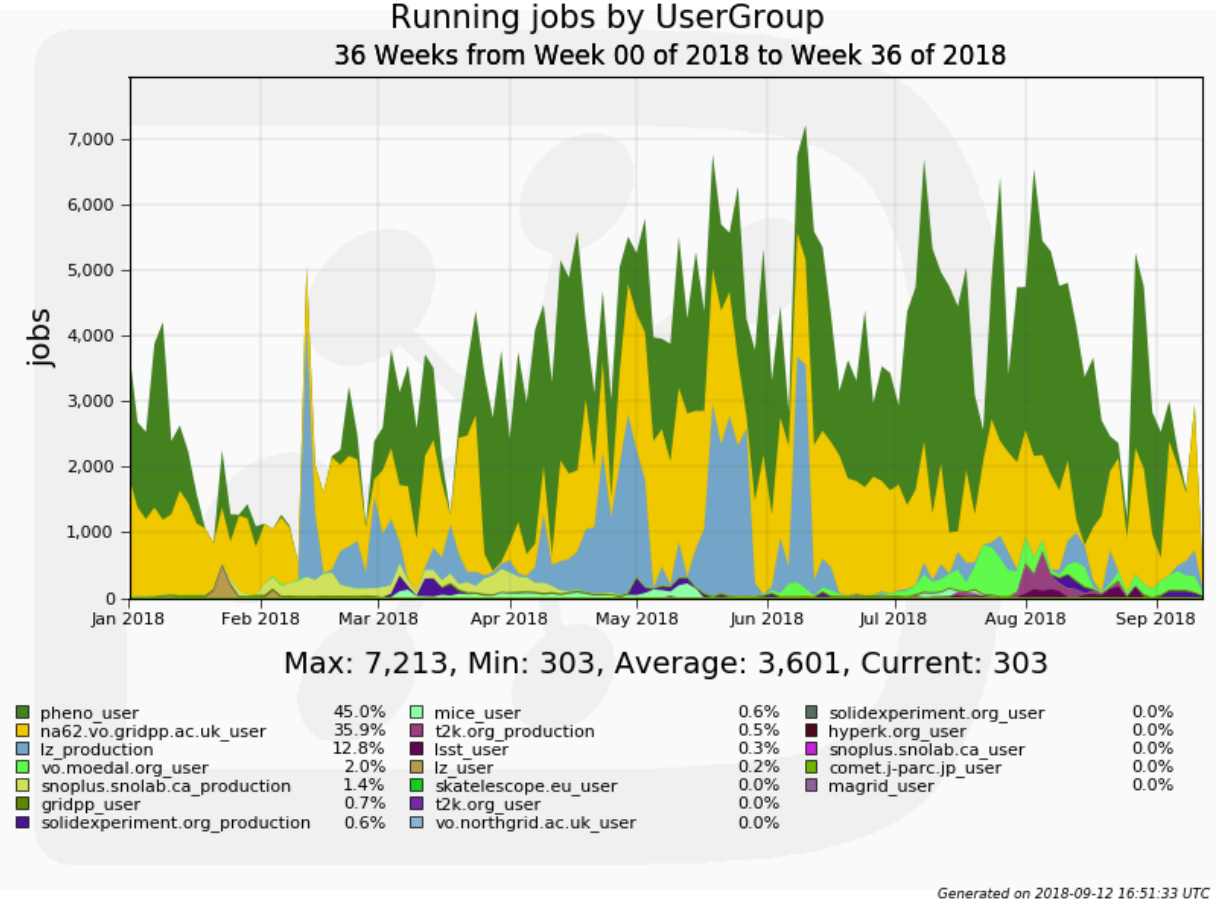
- Maintaining dedicated DIRAC services for small communities is not affordable
  - Requires expertise and effort to setup and run the system
- There was a clear need for services like DIRAC for an increasing number of communities with a low expertise in (distributed) computing and with high demands for computing resources
  - DIRAC framework was updated to support this kind of installations
- **DIRAC** services were provided by several National Grid Initiatives: France, Spain, Italy, UK, China, Romania, …
  - Some of them did not survive (Spain, Romania)
  - Some are still in active production

- Hosted by the CC/IN2P3, Lyon
  - *dirac.france-grilles.fr*
- Distributed administrator team
  - 5 participating universities
- In production since May 2012



- About 5 active communities – complex-systems, biomed, vo.france-grilles.fr, …
- > 20M jobs executed this year at 90 sites

- DIRAC service provided for the GridPP NGI
  - *dirac.gridpp.ac.uk*
- Hosted and operated by the Imperial College team

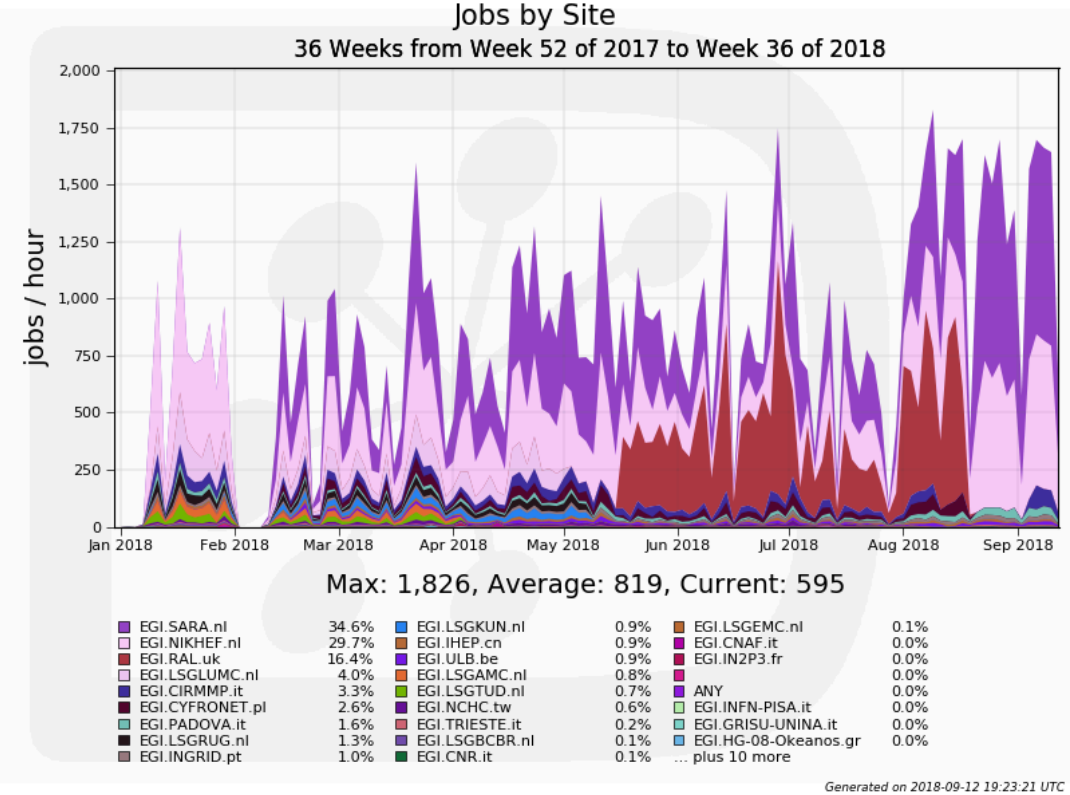


- About 10 active communities – LZ, NA62, Pheno, Snoplus...
- > 5M jobs executed this year

DIRAC4EGI service

- Partners
  - Operated by EGI
  - Hosted by CYFRONET, Krakow
  - DIRAC Project providing software, consultancy
  - Supported via the EOSC-Hub H'2020 grant
  - *dirac.egi.eu*
- 20 Virtual Organizations
  - enmr.eu
  - virgo
  - eli-beams
  - eiscat.se
  - fedcloud.egi.eu
  - …
- Usage
  - > 5 million jobs processed this year

DIRAC4EGI activity snapshot



EOSC-Hub services

- The main goal is to provide a Workload Management service to access EGI computing resources
  - Replacement of the gLite/EMI WMS
  - Providing access also to cloud resources (VMDIRAC extension)
    - EGI FedCloud sites
    - France-Grilles Federated Cloud sites
    - GridPP relies on Vac/Vcycle project to access national cloud resources
- Getting requests for higher level services
  - Bulk job operations
  - Workflow management (DIRAC Transformations)
- Possibility to attach private computing resources
  - Without the need to install grid middleware, using SSH tunnels

Services

- Data Management functionality is not in the formal mandate of the DIRAC4EGI service
  - However, there is a clear need for that
    - LFC decommissioning, access to data transfer services (FTS), etc
    - Managing user metadata
- DM components are provided by DIRAC
  - Configuration of the grid Storage Elements
    - Possibility to add private storages by installing DIRAC SE service
  - General purpose File Catalog
  - Dedicated community File Catalogs
    - Help in importing replica data (from LFC, from physical storage dumps)
  - Specific developments are done for several Competence Centers
    - E.g. custom File Catalog service with specific data access rules for the Eiscat 3D community

DIRAC
THE INTERWARE

User support

- Providing training for the EGI users
  - General courses
  - Webinars
  - Training for specific communities (Virgo, Auger)
- Helping in porting the applications
- Developing specific Web Portal features
  - E.g. custom File Browser and Job Launcher for the Eiscat 3D Collaboration

AMU Infrastructure

- The same functionality can be provided for the AMU users
  - M3AMU project
  - Integration of various computing and storage resources available for AMU users
  - Ensure sharing with uniform access rules
  - Increase usage efficiency and create new opportunities for the AMU users

Conclusions

- Large scientific communities have to employ divers geographically distributed computing and storage resources
- Several frameworks exist for building distributed computing systems aggregating multiple types of resources within large distributed infrastructures
- DIRAC provides an integrated solution with a reach set of ready to use services for managing computing resources, application workloads and data
- DIRAC can be easily extended to build high level services specific for particular user communities and architectures

http://diracgrid.org