WMS and Computing Resources

A.Tsaregorodtsev,
CPPM-IN2P3-CNRS, Marseille,
9th DIRAC User Workshop,
14 May 2019 London

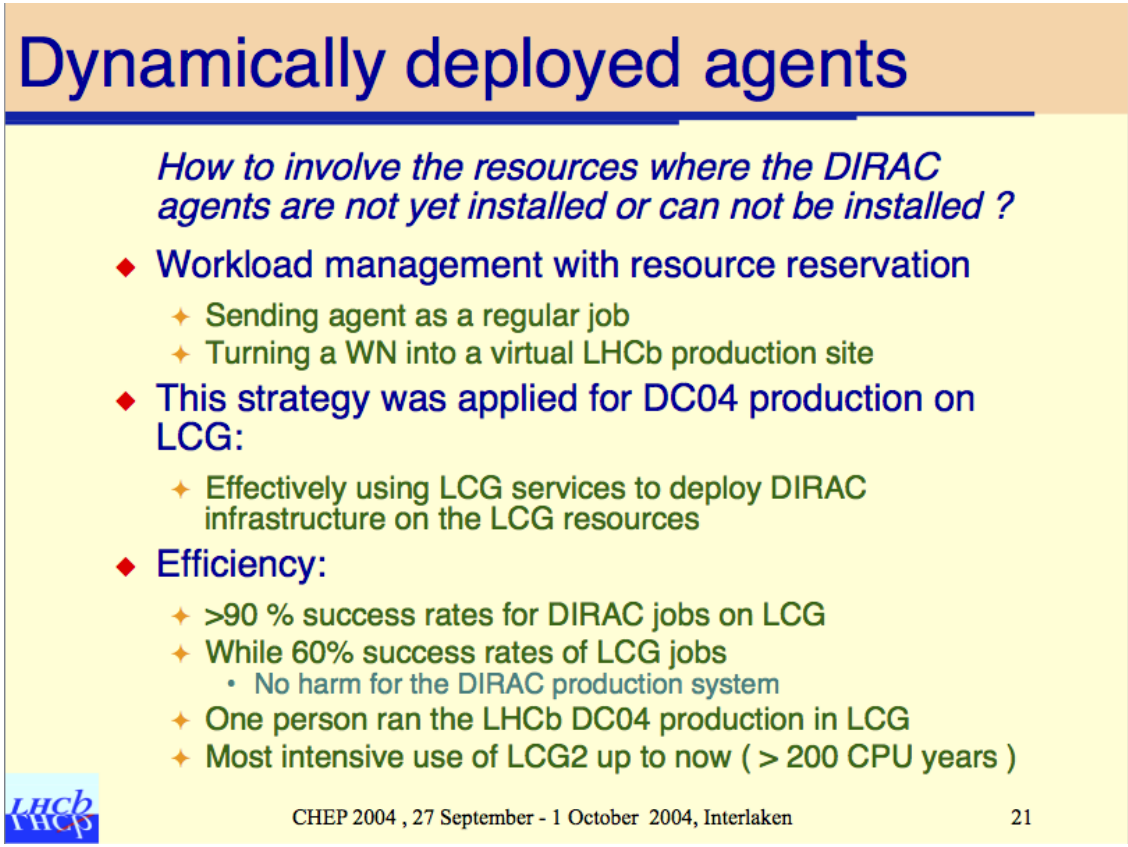
DIRAC
THE INTERWARE

Outline

- General WMS architecture
- Computing resources
- Interfaces
- Conclusions

WMS with pilot jobs

- First demonstration in 2004 in the LHCb Data Challenge
- Natural evolution of the ***PULL*** scheduling paradigm (as opposed to ***PUSH*** scheduling)

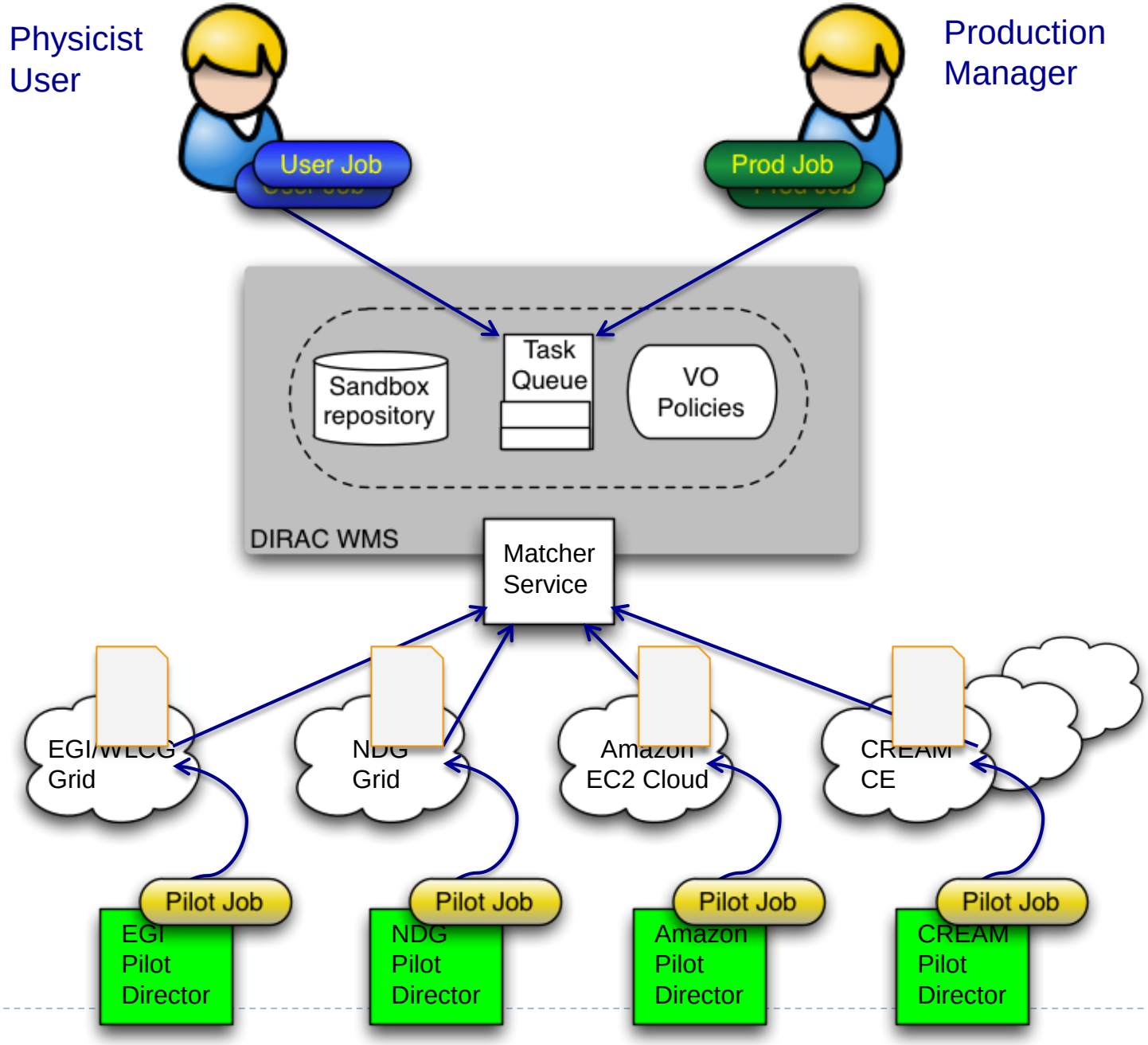


Dynamically deployed agents

How to involve the resources where the DIRAC agents are not yet installed or can not be installed ?

- Workload management with resource reservation
  - Sending agent as a regular job
  - Turning a WN into a virtual LHCb production site
- This strategy was applied for DC04 production on LCG:
  - Effectively using LCG services to deploy DIRAC infrastructure on the LCG resources
- Efficiency:
  - >90 % success rates for DIRAC jobs on LCG
  - While 60% success rates of LCG jobs
    - No harm for the DIRAC production system
  - One person ran the LHCb DC04 production in LCG
  - Most intensive use of LCG2 up to now (> 200 CPU years)

CHEP 2004 , 27 September - 1 October 2004, Interlaken 21



DIRAC
THE INTERWARE

DIRAC
THE INTERWARE

Pilot based WMS

- One evident advantage is that the users' payload is starting in an already verified environment
  - In early days of the grid and even now users saw an important decreasing of their jobs failure rate
- The environment checks can be tailored for specific needs of a particular community by customizing the pilot operations

Pilot based WMS

- Site resources providers does not need to distinguish individual users
  - One user identity represents the whole community to the sites
  - Simplifies site management but needs special trust relation between the site and the community
- Sites does not need to organize local resources to meet the community requirements
  - E.g. special queues per community groups with special fair sharing
- Adding new sites to the pool of DIRAC managed resources is considerably simpler
  - DIRAC does not require special services to be deployed on sites
    - There are exceptions (see the HPC case below)

Pilot based WMS

- User jobs submitted to the system are not passed immediately to a selected site but wait in the central repository – Task Queue
- Possibility to apply community policies by dynamically adjusting the job priorities
  - Similar mechanism to batch systems fair sharing mechanism
  - Job priorities can be adjusted using community specific plugins
  - Standard plugins include static group shares
  - Job priorities of users in the same group are dynamically adjusted based on the recent history of the consumed resources as provided by the Accounting service

Computing resources: grid

- DIRAC was trying to exploit all the possibilities to reserve computing resources
  - Using ***both*** grid job orchestration services (Resource Brokers, gLite WMS) and direct submission to Computing Elements
    - Wise decision: all the middleware brokers/WMS are now the story of the past
  - Need for dealing with different types of Computing Elements:
    - Computing Element abstraction with multiple implementations for different CE types
  - Currently supported Computing Elements types
    - Globus GRAM (is it used ?)
    - CREAM
    - HTCondorCE
    - ARC
    - SSH

CREAM CE

- CREAM will be retired soon
- Good service with a complete functionality:
  - CE status query
  - Job submission, status monitoring, logging, output retrieval
  - User proxy renewal
- No need to keep traces of the jobs locally
  - Different DIRAC services and agents can interact with the same CREAM CE service

ARC

- ARC
  - No a real service interface
    - Rumors are that a REST interface is in the works (ARC6)
  - Can not interrogate detailed CE status
    - Ldap service, not always up to date (not directly from the batch system)
  - Problems retrieving pilot output on demand to WMSAdministrator
  - Not using ARC data management

HTCondorCE

- HTCondorCE
  - Not very convenient to use
    - using command line
    - relying on locally stored information
  - Can not interrogate CE status
    - relying on PilotAgentsDB
  - Problem retrieving pilot output on demand to WMSAdministrator

How to cope with ?

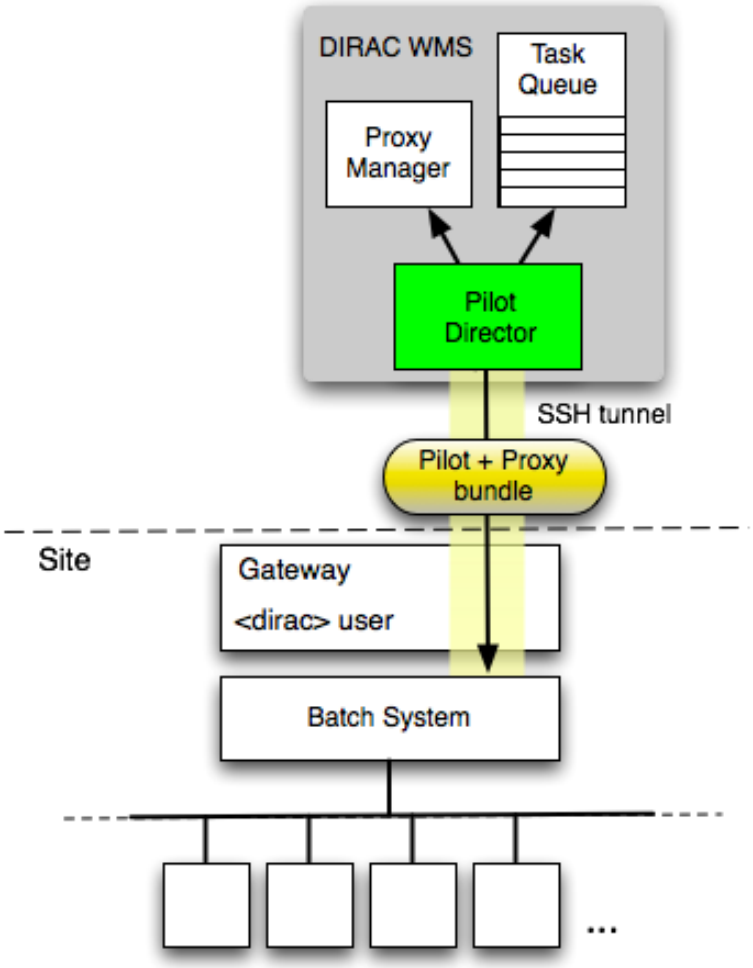
- CE status evaluation by monitoring submitted pilots
  - Pilots after the start are proactively reporting their status to the DIRAC central service
  - No need to interrogate Ces
  - Aborted pilots can be detected with a long delay
- Pilot output retrieval is rather complicated
  - Need for a generic mechanism to upload pilot output to the DIRAC central service (dedicated SE)
  - Pilot logging using MQ protocol is a step in this direction

No CE resources

- Computing resources with no special service for remote access
- Abstraction of a Batch System with implementations for:
  - GE, Condor, Torque, LSF, SLURM, OAR
  - Host
- The BatchSystem objects are used by special ComputingElements
  - LocalComputingElement
    - Submits and manage jobs in the local batch system
    - Can be used by on-site Pilot Directors
  - SSHComputingElement
    - See below

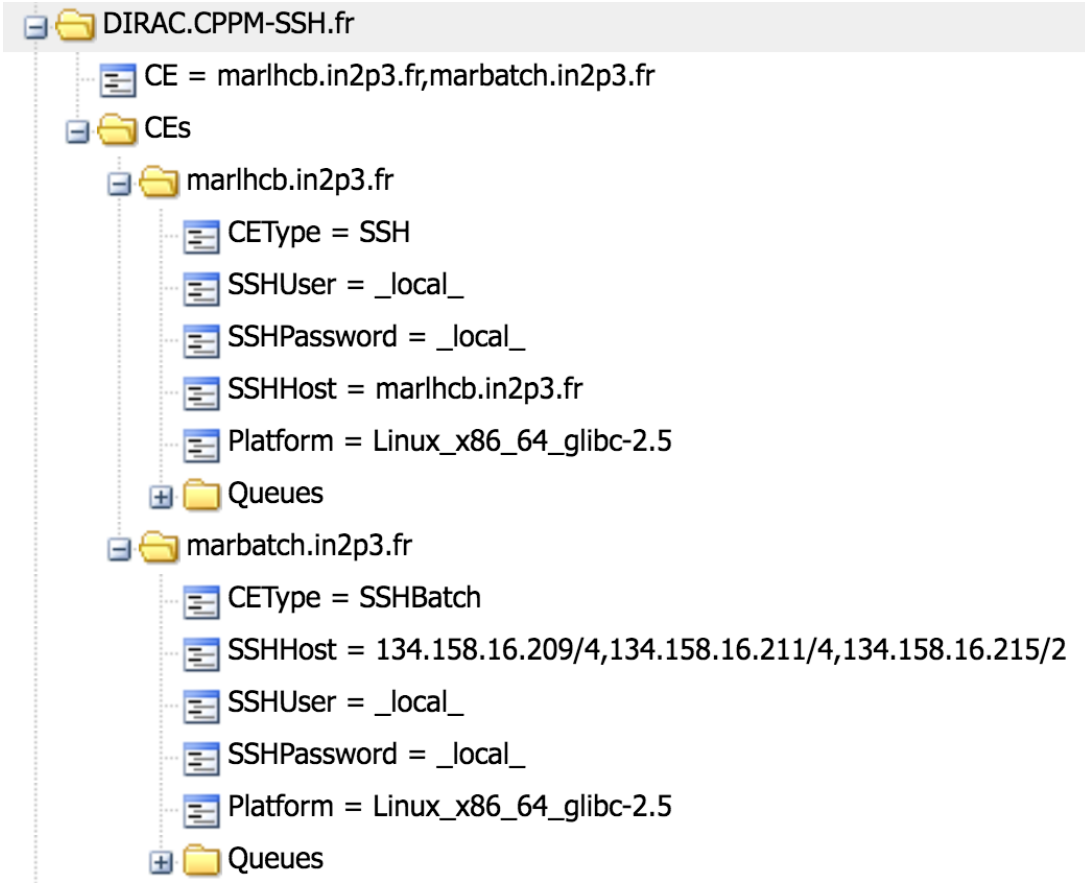
Standalone computing clusters

- 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



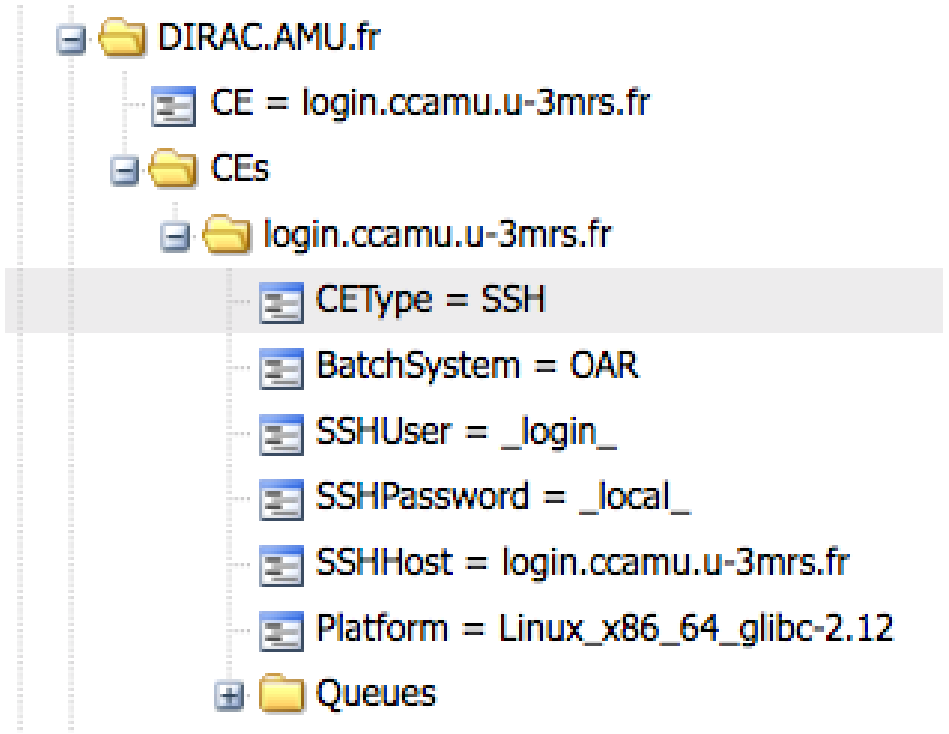
SSH CE: simplest case

- SSH CE simplest case:
  - One host CE with one job slot
- SSHBatch CE
  - Several hosts form a CE
    - Same SSH login details
    - Number of job slots per host can be specified
- Pilots are sent as an executable self-extracting archive with the pilot proxy bundled in



SSH CE with a batch system

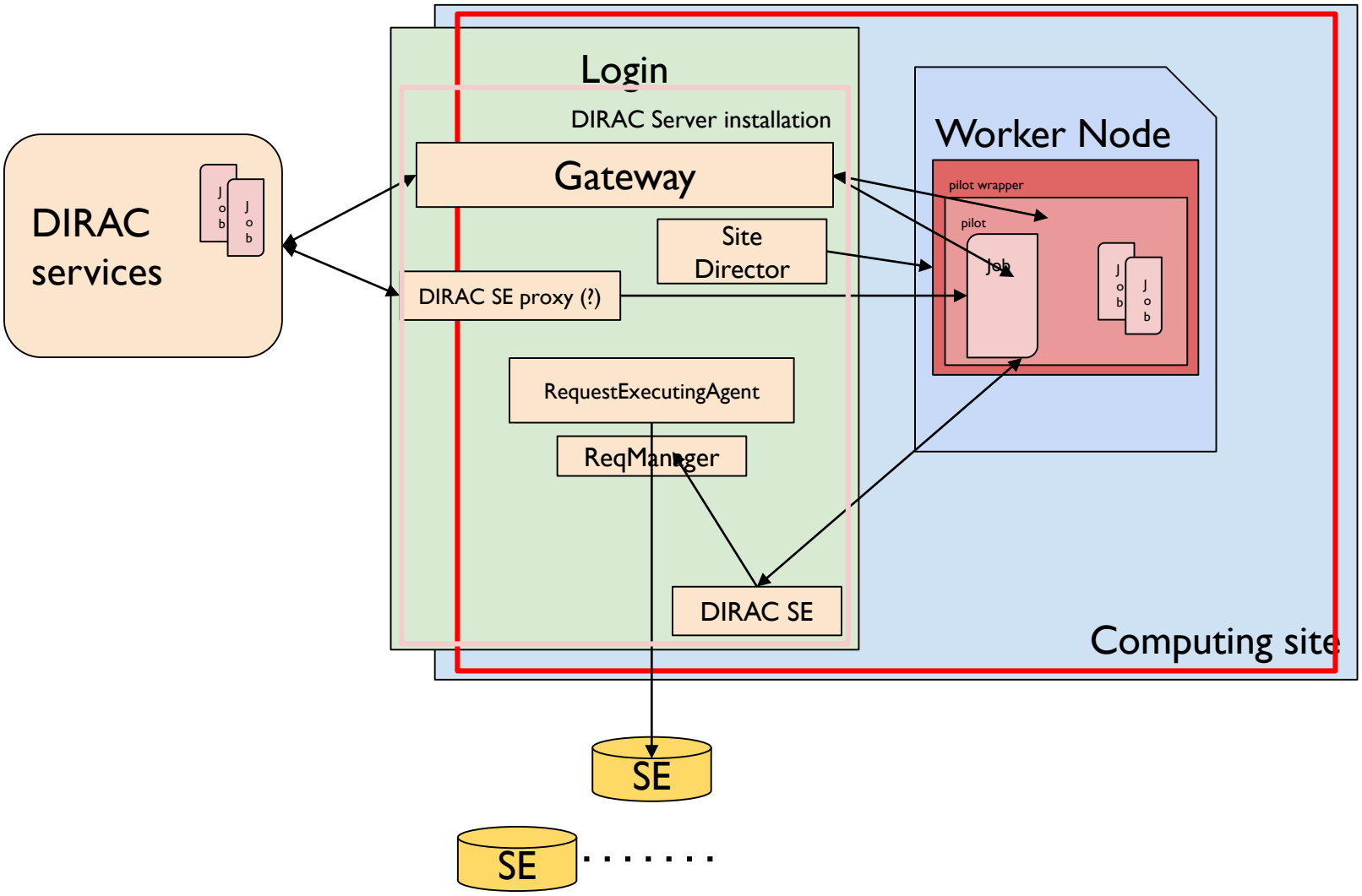
- SSH login to the cluster interactive host
  - Copy several tools, e.g. BatchSystem plugin at the first time
- Submit pilots to the local cluster using a relevant BatchSystem plugin



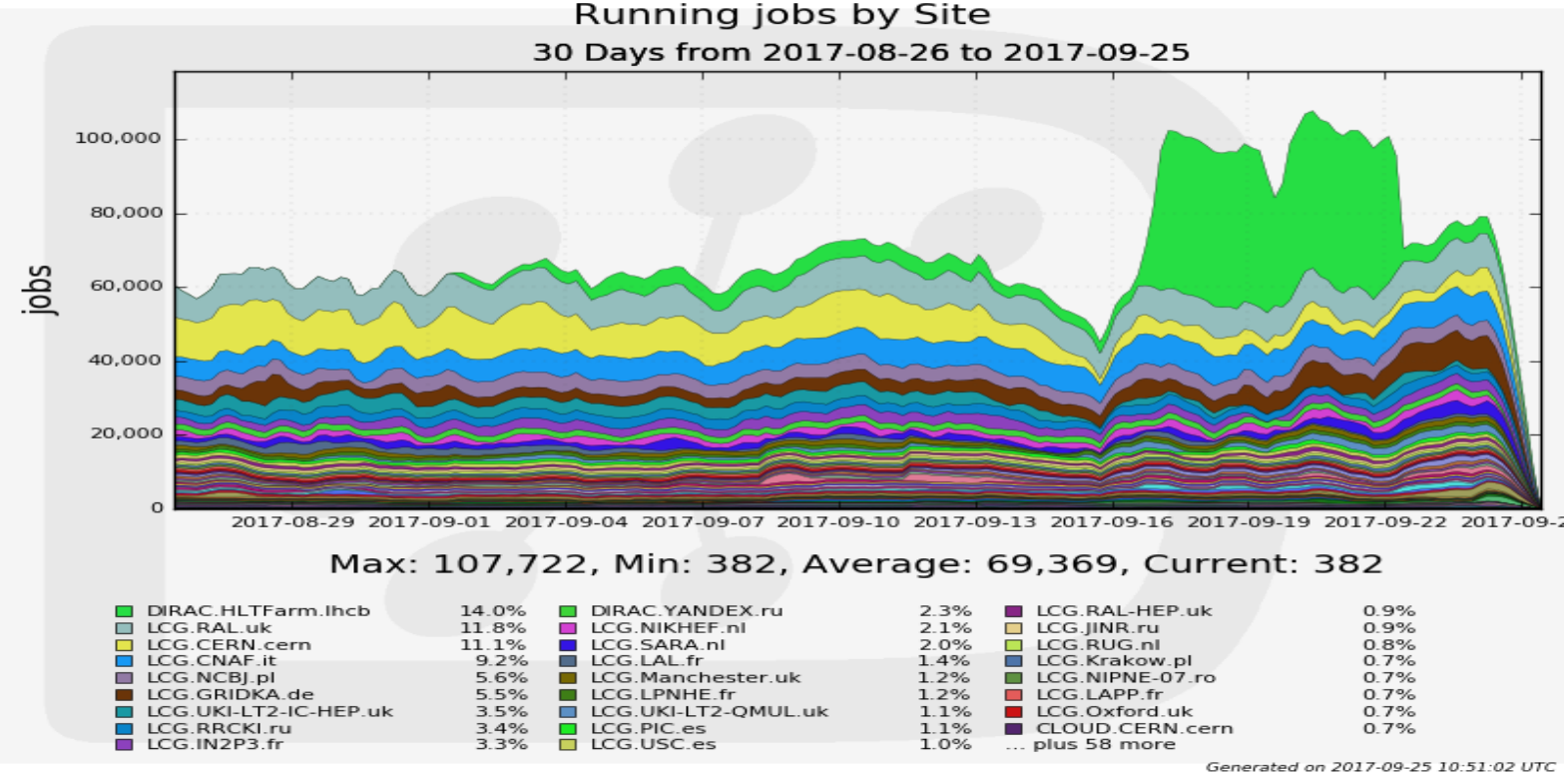
HPC Centers

- The number of accessible HPC centers is growing
- They are typically not part of any grid infrastructure
  - Define own rules for accessing, outbound connectivity, software distribution, etc
- Under certain circumstances the use is trivial
  - SSH access allowed
  - Outbound connectivity from WNs
  - CVMFS available
- Otherwise special setup is needed for each center
  - See example below

HPC access possible setup



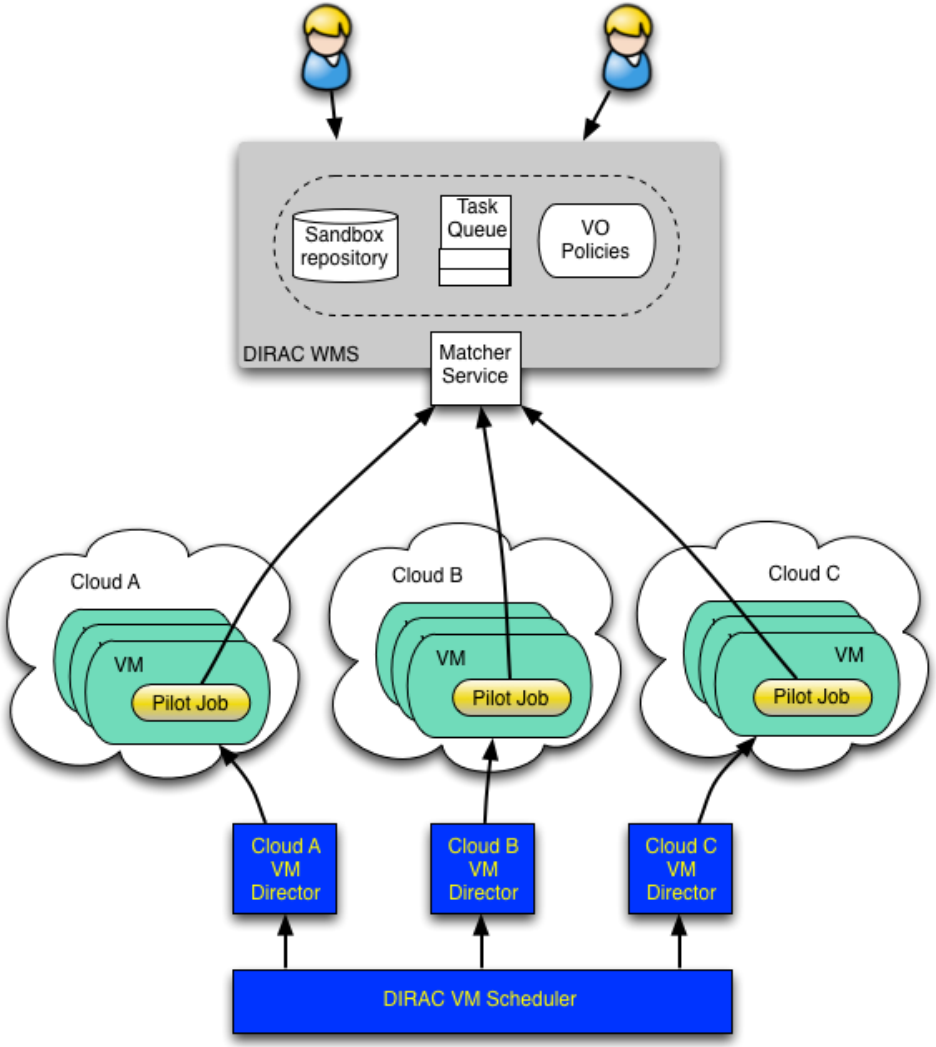
LHCb Collaboration



- 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

Clouds

- Clouds can provide flexibly computing resources
  - The amount of HTC suitable resources remains small
- 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 different cloud provider services



VMDIRAC

- Currently supported cloud connectors
  - Legacy cloud endpoint connectors
    - Apache-libcloud
    - Rocci command line
    - OCCI REST
  - Amazon EC2
    - Using boto2 python binding
  - Using directly cloud services
    - Openstack REST interface
    - OpenNebula XML-RPC interface

Openstack cloud

- Openstack is the most popular cloud endpoint
  - Most of the EGI FedCloud sites
- Keystone v2 & v3 is supported
  - Login/password
  - VOMS proxies
- Openstack dashboards support SSO login
  - Not yet used in the service APIs

VMs

- VM contextualization
  - Starting from standard images (SL6 or CC7 or …)
    - No need to maintain special DIRAC images
    - Specific images can be needed to access special resources, e.g. GPUs
  - The necessary software environment is prepared during the VM bootstrapping
    - CVMFS, Docker, Singularity, etc
    - More elements can be added

VM Pilots

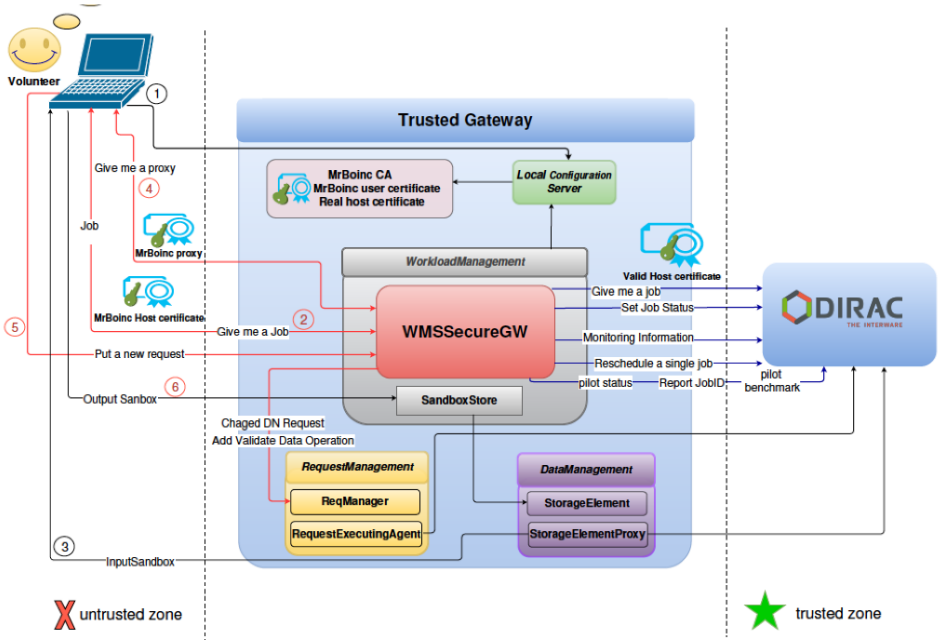
- Pilots are started as the last step of the VM contextualization process
  - Still use VMDIRAC specific pilot deployment
  - Pilot 3 integration is necessary
- Single pilot is started on one VM
  - Exploiting all the VM CPU cores
  - No one pilot per CPU core

VM Pilots

- VM pilots use PoolComputingElement to manage local resources
  - On-WN batch system
  - Flexible strategy with prioritized job requests to the Matcher, e.g.:
    - First, ask for jobs requiring WholeNode tag
    - If none, ask for jobs requesting as many cores as available
    - If none, ask for jobs with MultiProcessor requirement
    - If none, ask for single-core jobs
  - The goal is to fill the VMs with payloads fully exploiting there multi-core capacity
    - Other job request strategies can be added

Volunteer grids

- LHCb solution
- 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



- BoincDIRAC extension created
  - Waiting for an interested developer to make it a generic solution

User interfaces

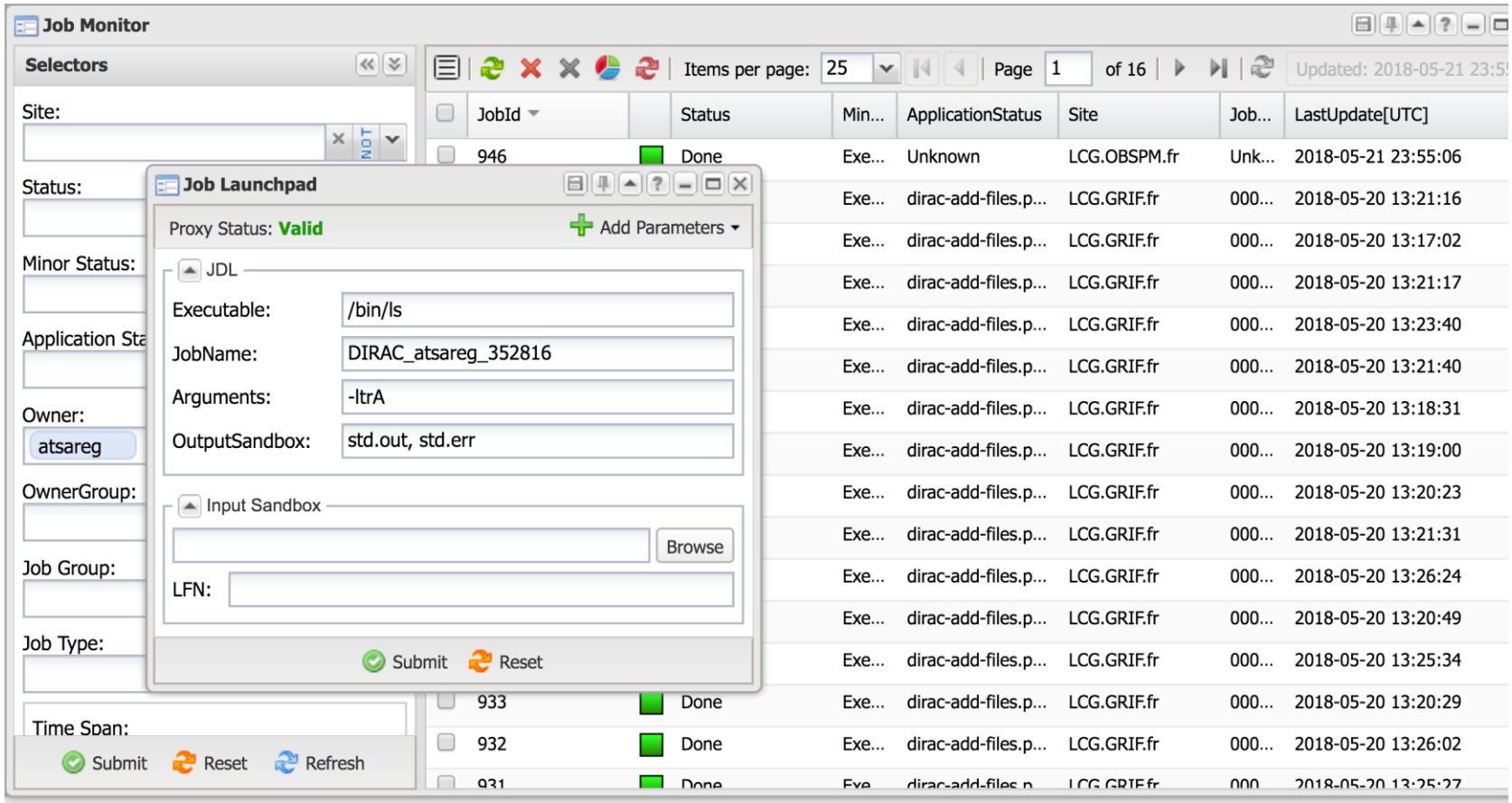
- For the users all the internal WMS/pilots machinery is completely hidden. They see all the DIRAC operated computing resources as single large batch system
  - Presented in the Basic DIRAC Tutorial tomorrow
  - Command line

```
[atsareg:~/work/test/DiracTest] $ dsub /bin/echo "Hello world"
946
[atsareg:~/work/test/DiracTest] $ dstat
JobID  Owner    JobName  OwnerGroup      JobGroup  Site          Status   MinorStatus            SubmissionTime
================================================================================================================
  946  atsareg  Unknown  dirac_tutorial  00000000  LCG.OBSPM.fr  Matched  Job Received by Agent  2018-05-21 23:54:48

[atsareg:~/work/test/DiracTest] $ doutput 946
[atsareg:~/work/test/DiracTest] $ ls -l 946
total 8
-rw-r--r--  1 atsareg  staff  12 May 22 01:54 std.out
```

  - DIRAC API
  - Web Portal

Jobs in the Web Portal



Conclusions

- Pilot based WMS proven to be efficient in the HEP experiments is now available for the users of dedicated and multi-VO DIRAC services
- A large variety of heterogeneous computing resources can be federated due to the pilot job mechanism
- Ongoing effort to make new non-grid resources conveniently available (HPC, Cloud)
- Keeping uniform resource access interfaces for the users – single DIRAC computer