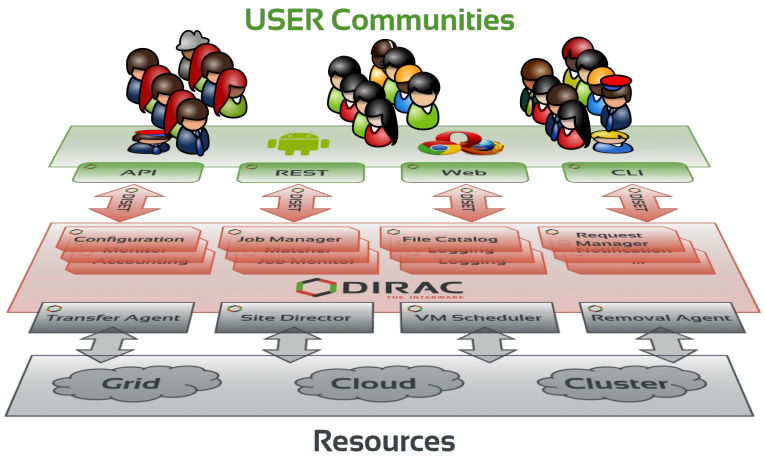
The DIRAC interware

current, upcoming and planned capabilities and technologies

Federico Stagni
DIRAC technical coordinator
on behalf of the DIRAC consortium
federico.stagni@cern.ch

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



- Started as an LHCb project, experiment-agnostic in 2009
- Developed by communities, for communities
  - Open source (GPL3+), GitHub hosted
  - Python 2.7 (python 3 in development)
  - No dedicated funding for the development of the “Vanilla” project
  - Publicly documented, active assistance forum, yearly users workshops, open developers meetings and hackathons
- The DIRAC consortium as representing body

Users/communities/VOs

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

Experiment agnostic

Extensible

Flexible

DIRAC
THE INTERWARE

Jobs and files

WMS and DMS

[WMS] resources federation

Pilots are the "federators"

Send them

as "pilot jobs" (via a CE)

Or just **Run them!**

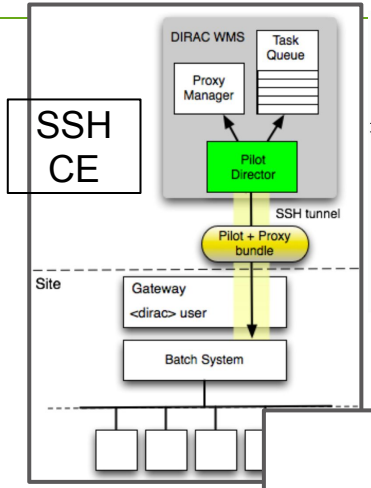
e.g. as part of the contextualization of a (V)M

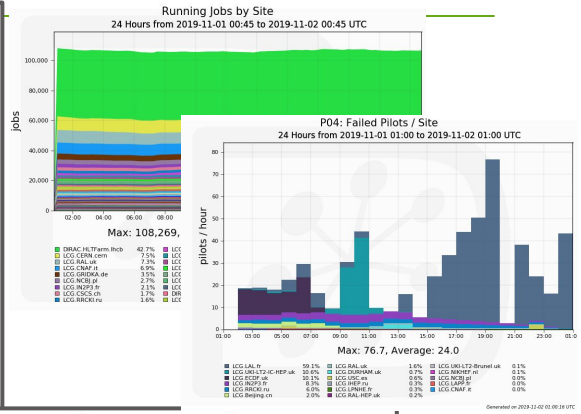
OR

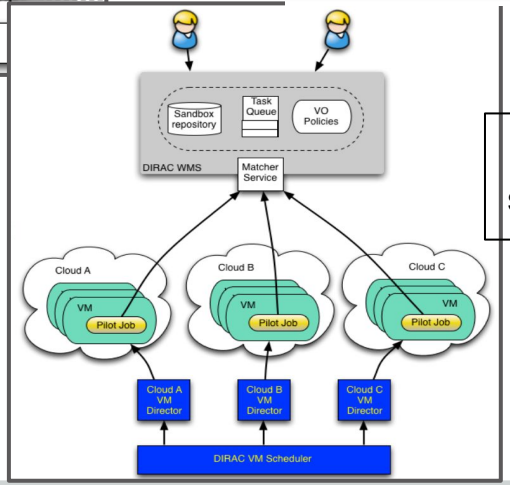
"Make a machine a pilot machine, and you are done"

[WMS] Computing resources

- **Grids** (EGI, OSG, NorduGRID)
  - CREAM, HTCondor, ARC
- **Clusters** behind a BS
  - access through SSH/GSISSH tunnel
    - a really thin layer that we call “SSH CE”
- **Vacuum**:
  - VAC/vcycle resources
  - BOINC Volunteer resources
  - HLT farm (LHCb)
- **VMs scheduler**:
  - Openstack, Keystone v2 & v3, OpenNebula XML-RPC, Amazon EC2 (boto2), Apache libcloud, rocci cli, OCCI REST
  - Contextualization from standard images
    - with, at least, the DIRAC pilot
- **HPC sites**
  - see later





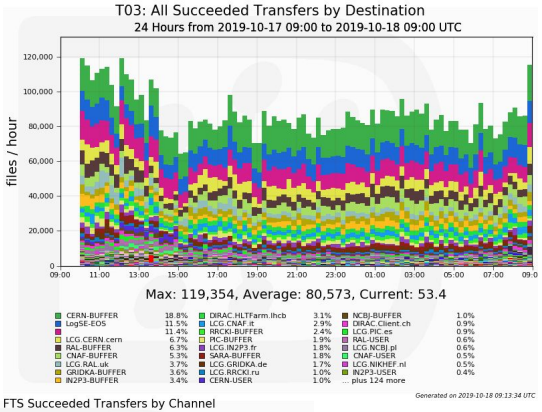


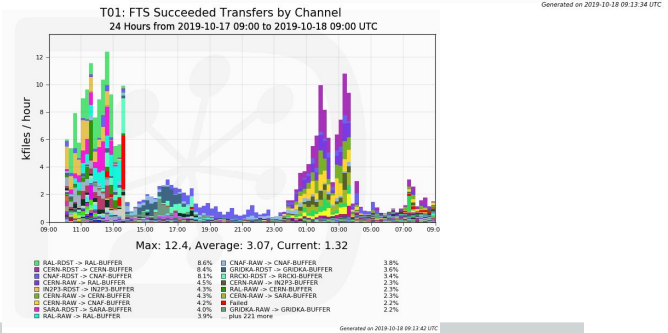
VM scheduler

[DMS] Data Management

Basics of DMS:

- **LFN**s: unique identifier within DIRAC of a file
  Logical File Name
  (described as paths)
- LFNs are registered in **catalog**(s).
  and there are implementations like the DFC
  → and you can connect as many catalogs as you want
  (including the LFC or Rucio catalog)
- LFNs *may* have **PFN**s, stored in **SE**s.
  Physical File Name on **Storage Elements**
  (and SEs are monitored, within the DIRAC Resource Status System)
- You can access those PFNs with several protocols.
  e.g. root, gsiftp, srm, http, file, dip
  (and can also be brought online - i.e. staged)





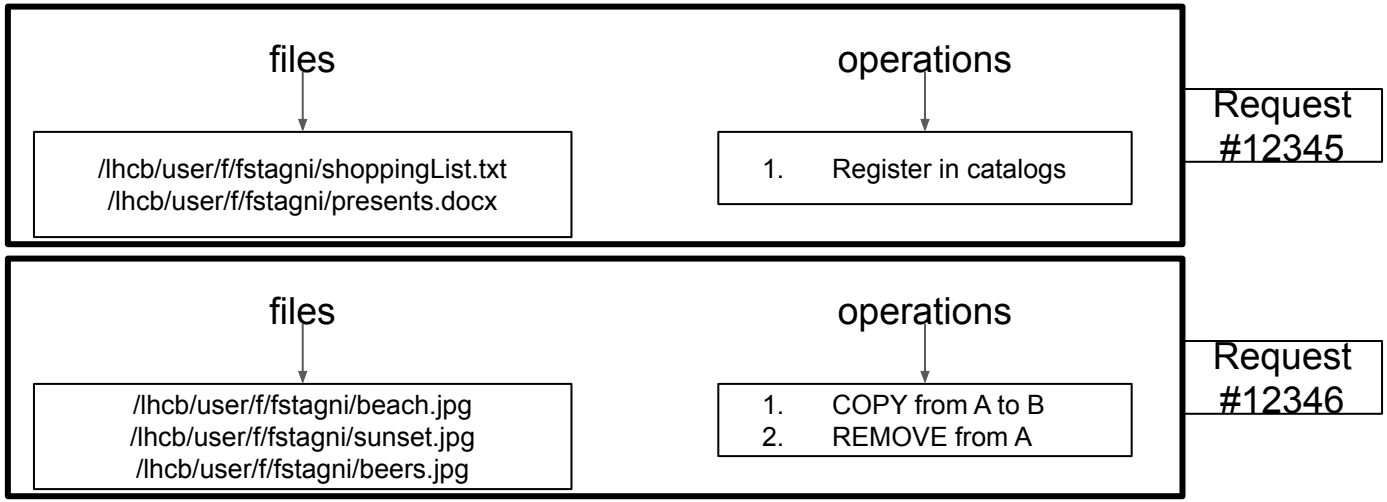
DIRAC
THE INTERWARE

Productions and datasets

DIRAC RMS

Request management system

A generic, flexible system, which can be used for queueing *operations* (on files, but not only)
like a to-do list



A key system for **job failovers**, and for **DMS operations**

Operation types:	• ReplicateAndRegister (e.g. using FTS) • RemoveFile/RemoveReplica • Issue an RPC call to a DIRAC service • ...add your own

DIRAC TS

Transformation system

A generic system for queueing similar *operation types* on certain *datasets* and forward them to the appropriate *systems*

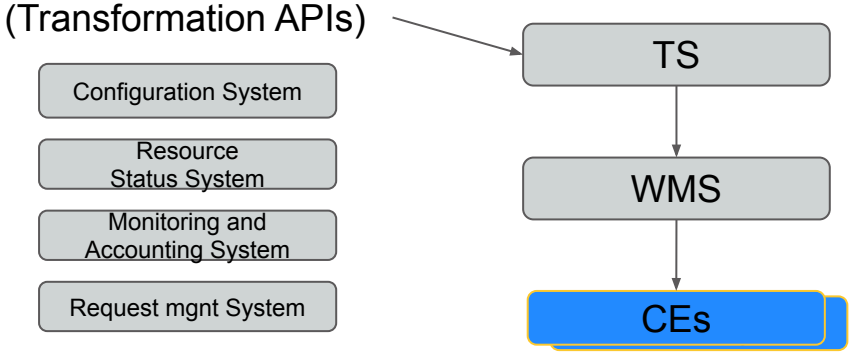
An *operation type* can be, e.g.: • a simulation workflow • a reconstruction workflow • a replication • a removal • ...	A *dataset* is split into groups, based on criterias defined by *plugins*, e.g.: • split by size • by destination • by metadata • … [code it]	A *system* is either (today) the DIRAC WMS (for productions) or the DIRAC RMS (for dataset management operation types)

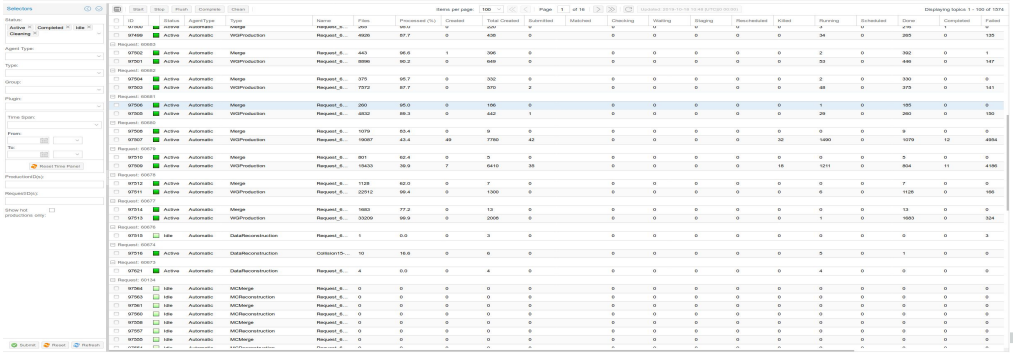
[DMS] example (for dataset management): `Take all my holidays pictures from 2018 with tag='sunset', make sure that there is one copy on tape and one on disk, distributed on all the sites according to free space, and group the operations by group of at most 100 files.`

[WMS] example (for jobs productions): `Take all my holidays pictures from 2018 with tag='sunset', make sure to run (only once) the 'red-enhancer' workflow on each one of them, using only Tier2 sites.`

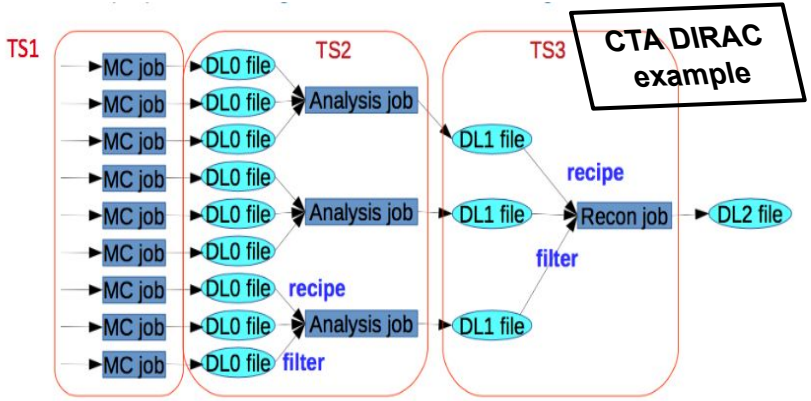
[PMS] Productions management

DIRAC Transformation System + WMS:
for single productions



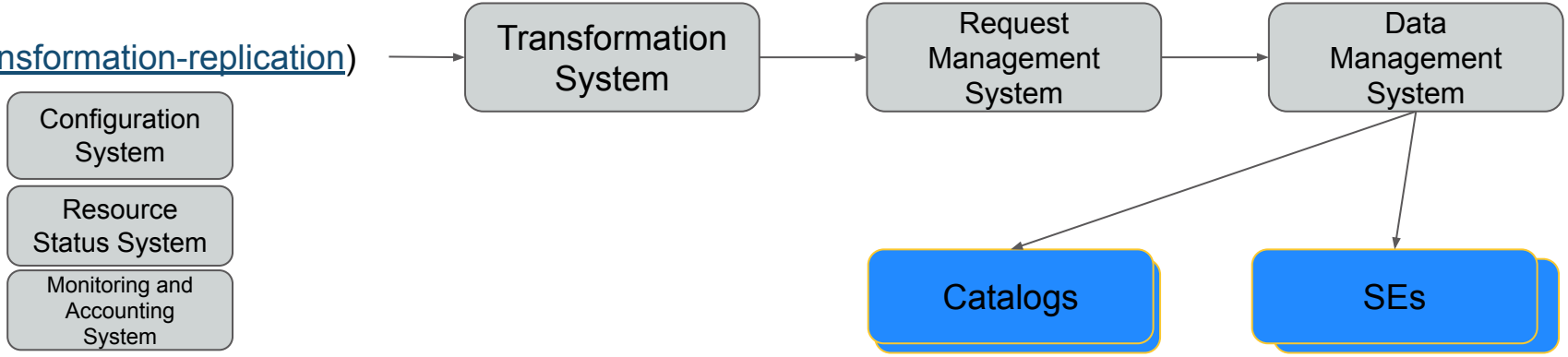


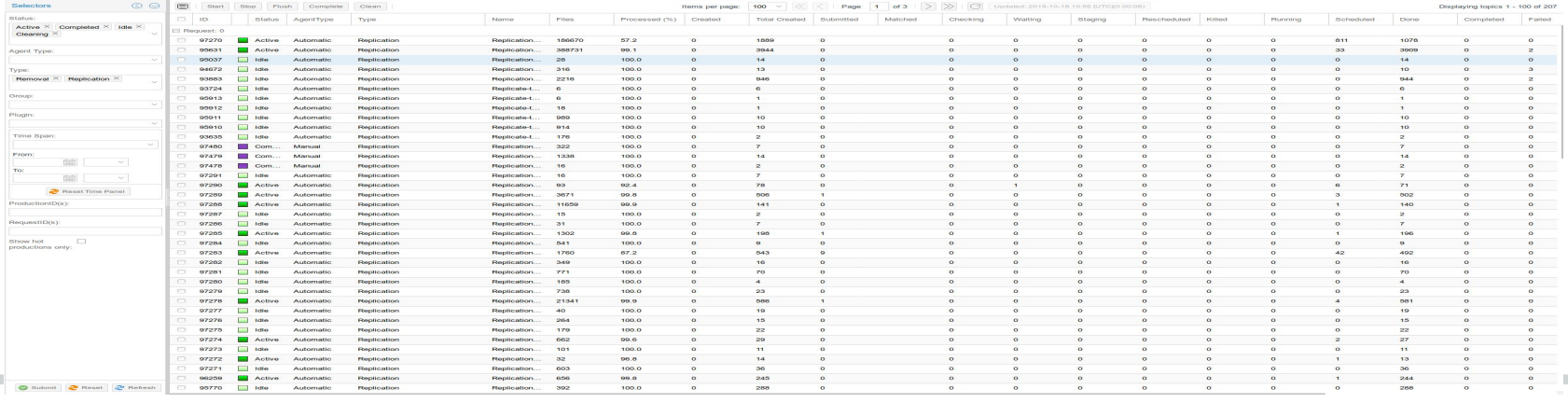
DIRAC Productions System:
chaining job productions together



[DMS] Dataset management

(dirac-transformation-replication)

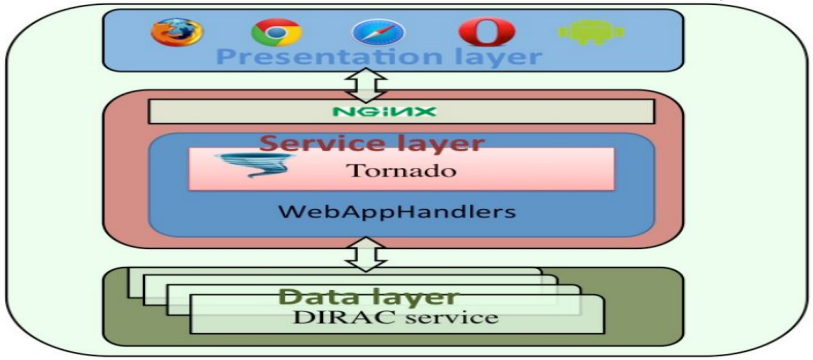


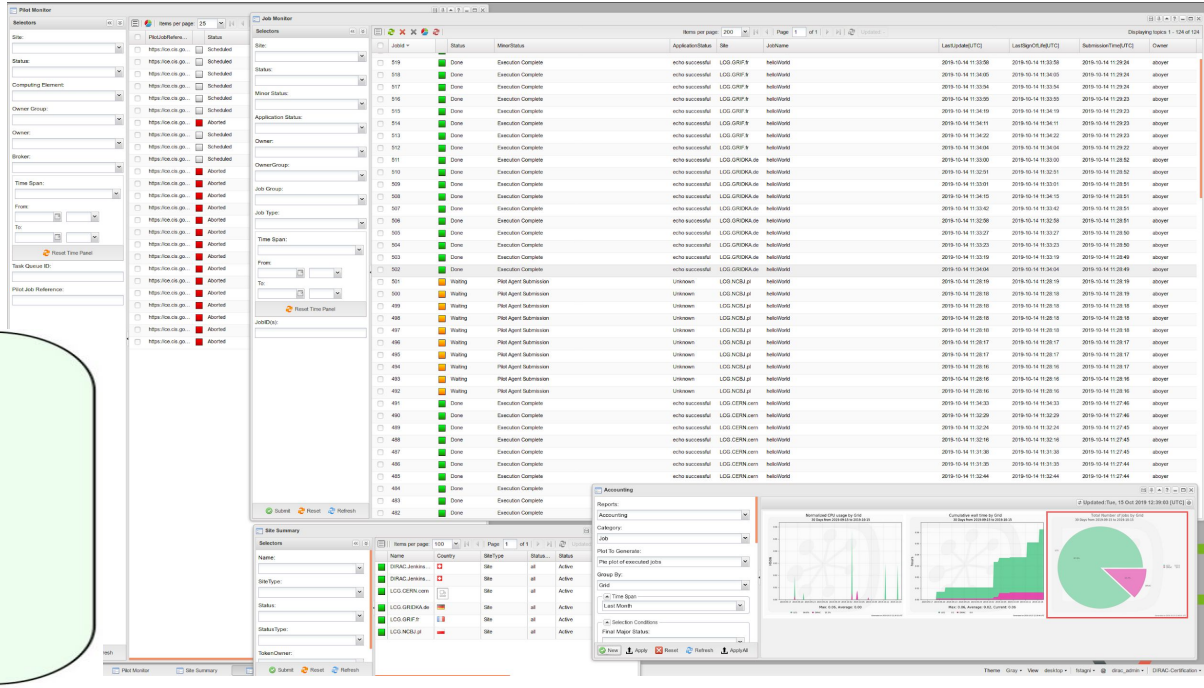


WebApp

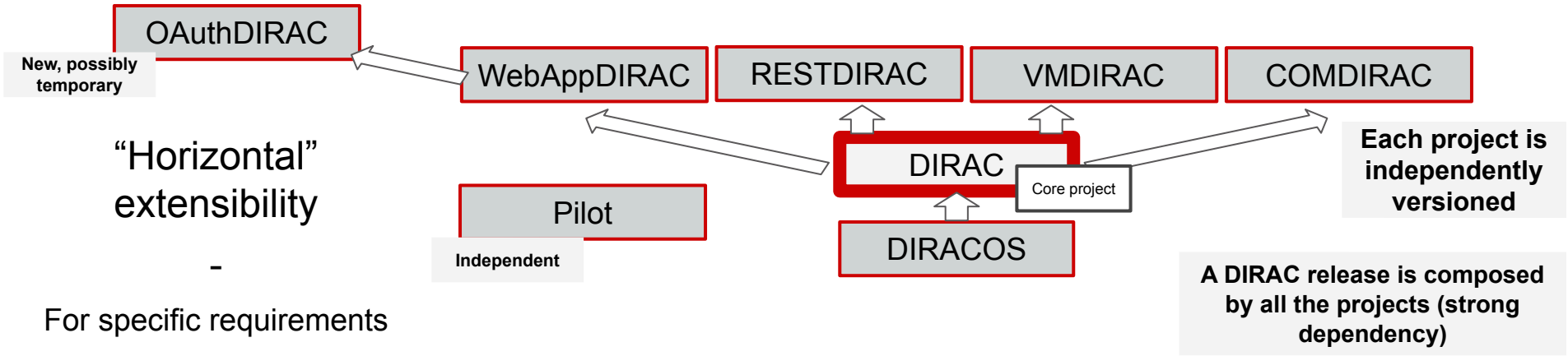
- Web users' interface
- Frontend: ExtJS6
- Backend: tornado, NGINX

- Each system has its own Web application





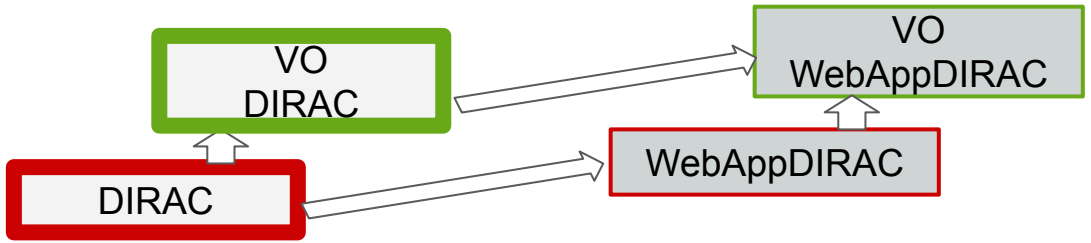
Experiment agnostic, and extensibility



"Horizontal" extensibility

-

For specific requirements



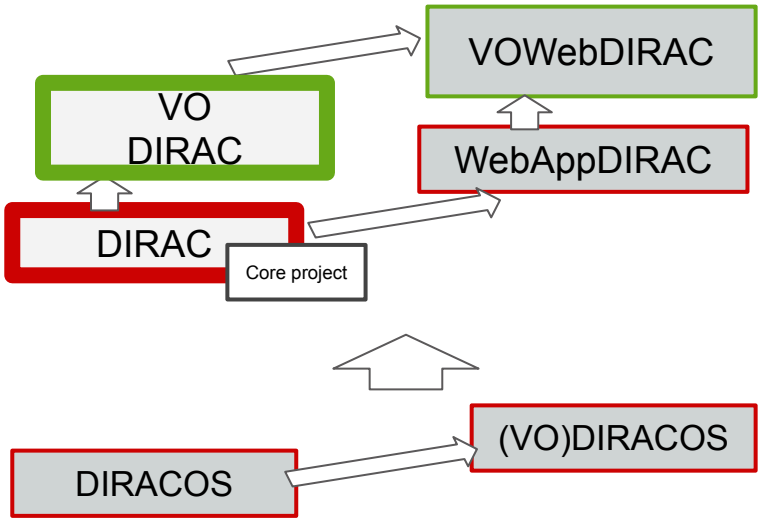
"Vertical" extensibility

-

Community driven

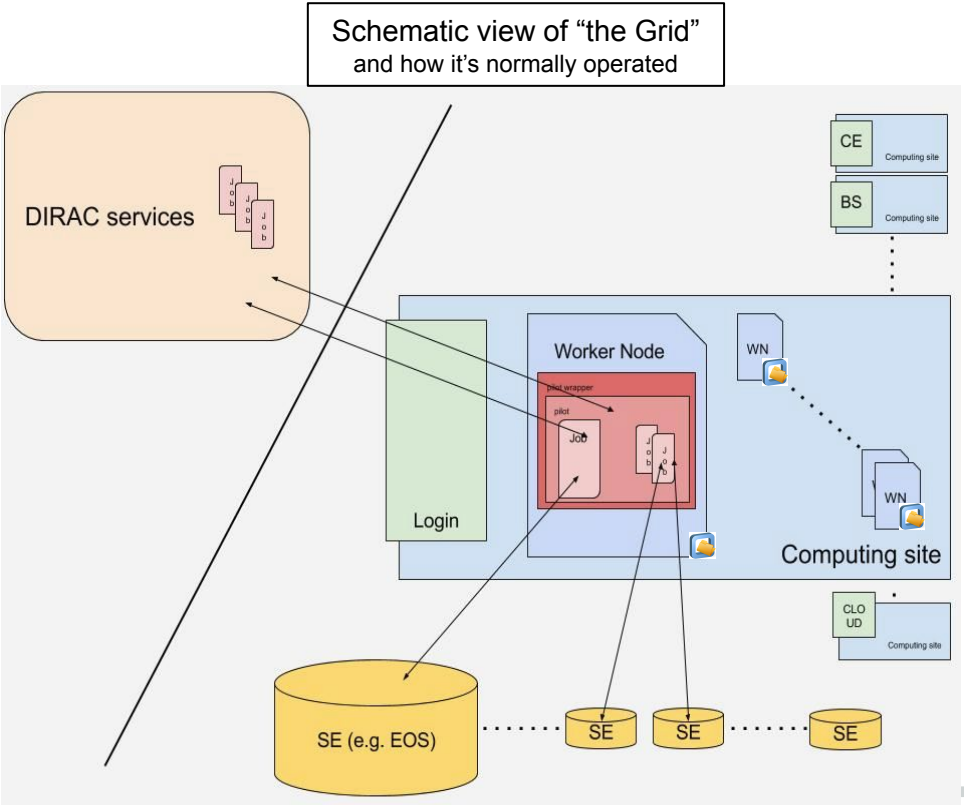
Recent and ongoing developments

DIRACOS



- New way of packaging (VO)DIRAC dependencies
- Based on Fedora Mock and Yum repo
- Simple grammar
- Extensible
- Testable
- Automated build
- ...already thinking at v2 with Conda
- On github
- More in this pres

General HPC challenges:
distributed computing



~easy integration when:

1. WNs have outbound connectivity
2. CVMFS endpoint(s) mounted on the WNs

   (this is pretty common req)

3. SLC6 or CC7 “compatible”, or Singularity. x86.

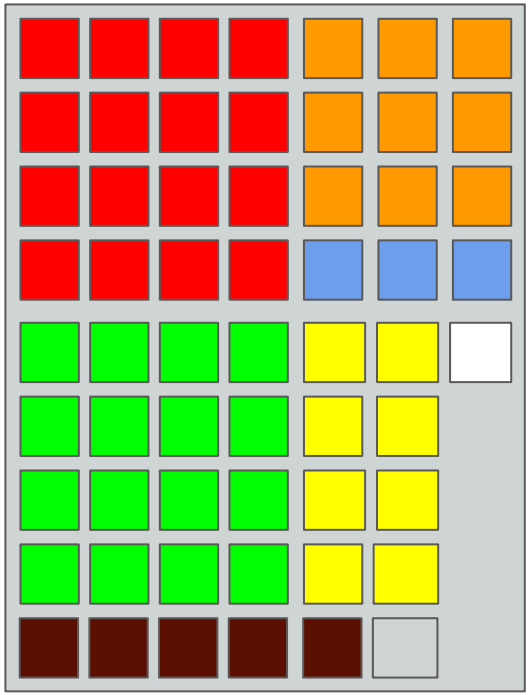
   (this might or might not apply to you)

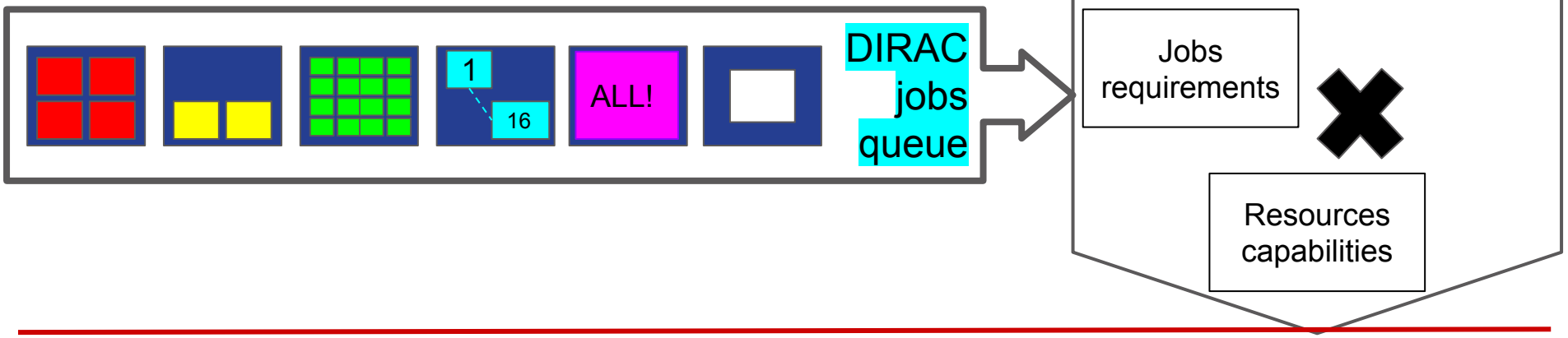
If some of the above is missing, work need to be done to address it. DIRAC has partial solutions for case 1.

The case of fat nodes

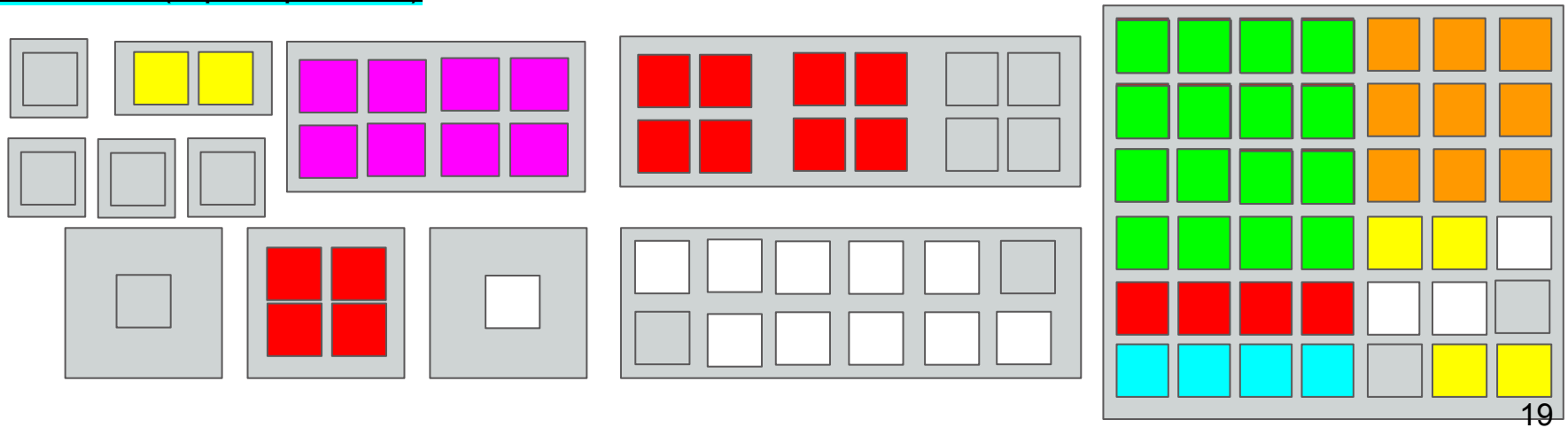
- Exploiting many-core architectures

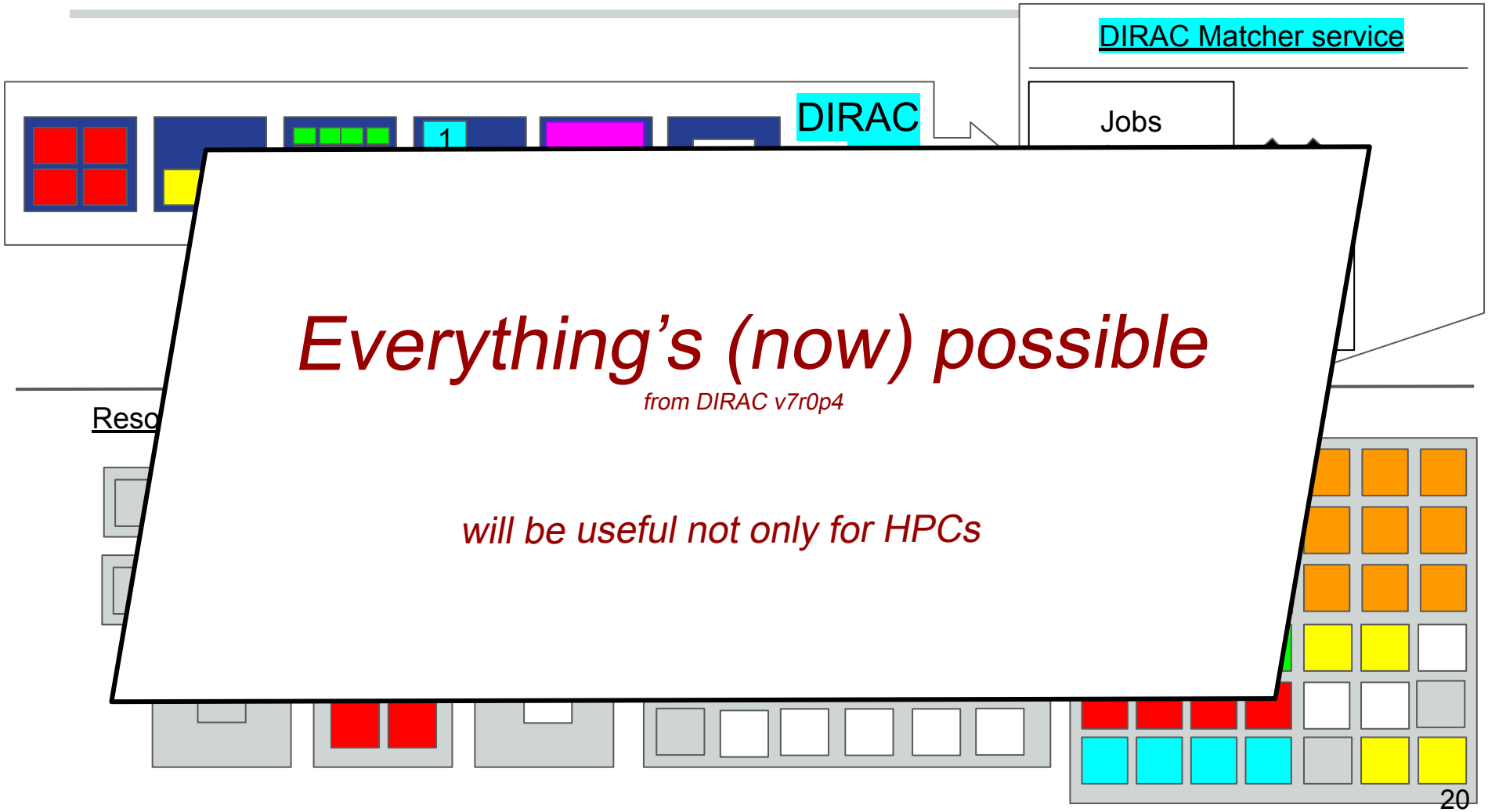
  [LHCb has a case for running on nodes with 272 logical processors (CINECA)]
- DIRAC needs to “partition” the node for optimal memory and throughput (and maybe only use a subset of the logical processors)
  - Use DIRAC “Pool”, an “inner Computing Element”
  - Parallel jobs matching





Resources (1 pilot per box)

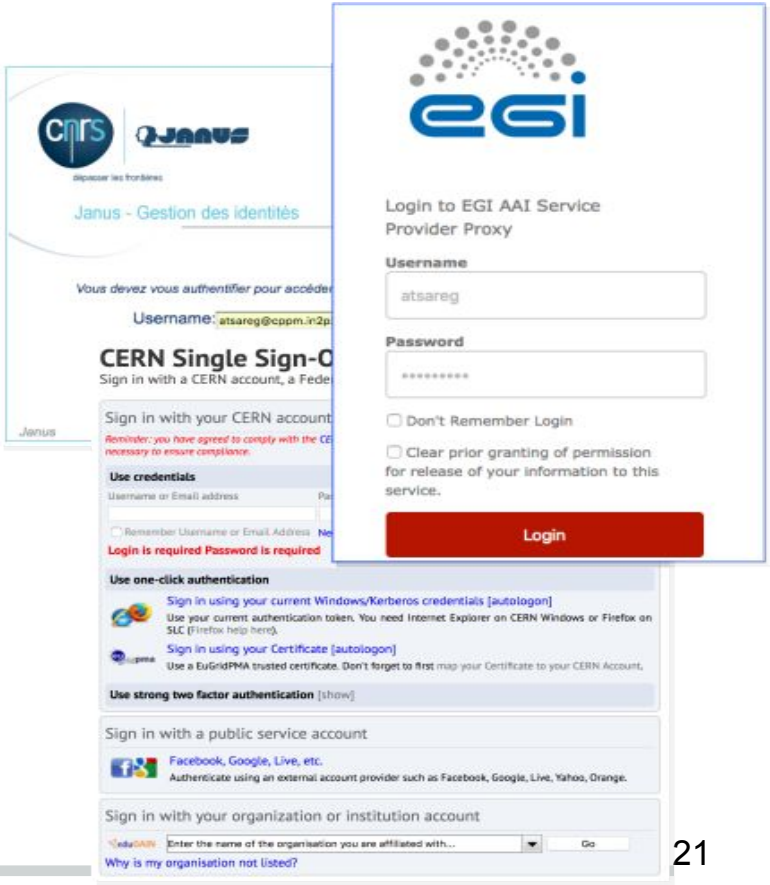



DIRAC Matcher service
DIRAC
Jobs
Reso
Everything's (now) possible
from DIRAC v7r0p4
will be useful not only for HPCs

Oauth based authN/Z

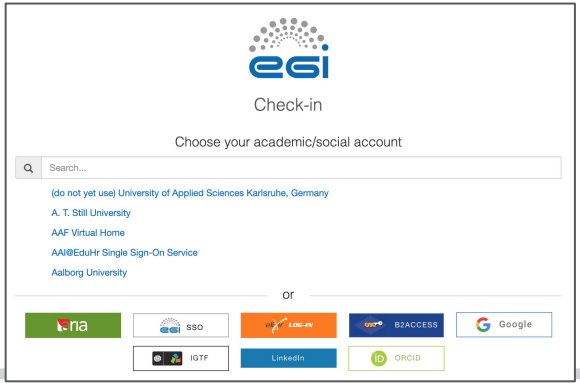
Until “yesterday”: X509 certificates, DIRAC groups, proxies, VOMS

- DIRAC can delegate AuthN to an external server
  - ensure provisioning of X509 certificate proxies
- Focus: OAuth/OIDC as “industry standards”
  - Use case: EGI Check-in SSO hub



Web portal authentication

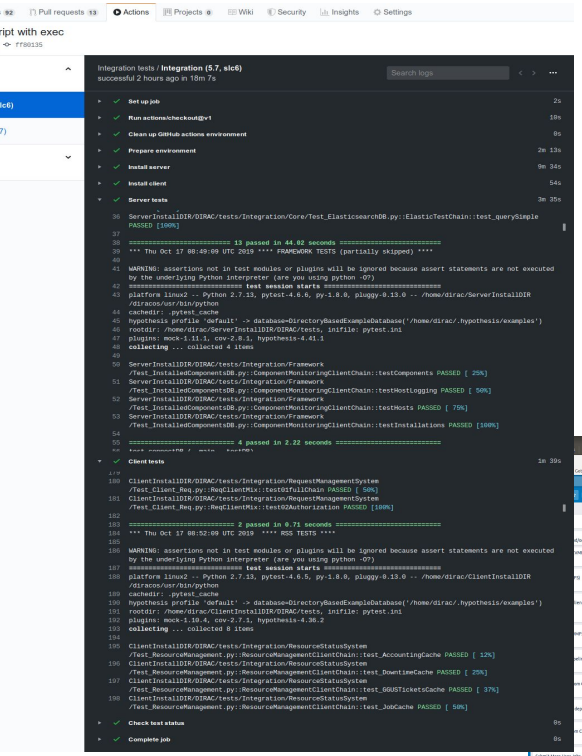
CLI authentication



Other developments

- `dips://` → `https://`
  - `dips`: proprietary protocol for RPC calls
  - `http`: frameworks already exists in python 2&3 for server-side (tornado) and client side (requests)
- Python 3
  - Migration started, first production release next year
  - DIRAC Pilot will move first (also b/c of CentOS8)
- Interfacing with INDIGO IAM (“after VOMS”)
- DIRAC ←→ Rucio bridge
- Containerizations (several)

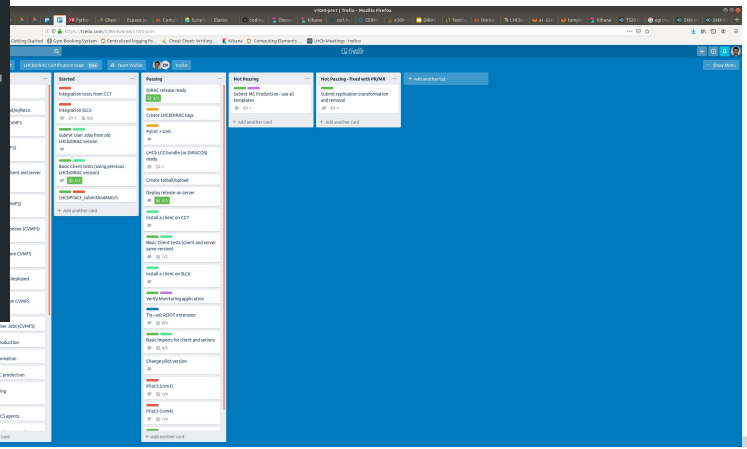
Development and testing

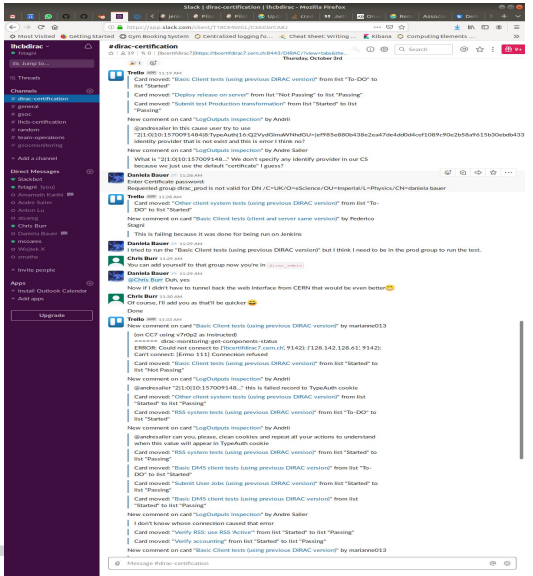


~5 FTE as core developers, a dozen contributing developers

Tests, certification, integration process is a daily work.

- We use GitHub Actions, GitLab CI/CD (Travis, Jenkins…)
- We run certification hackathons





Outreach

- diracgrid.org
- dirac.readthedocs.io
  - including code documentation
- Ops and general questions: Google forum
- Dev and DevOps issues: on github
- Bi-weekly developers meetings (and/or hackathons): BILD

10th DIRAC
Users' Workshop

indico.cern.ch/e/DUW10

25-29 May 2020
KEK, Japan

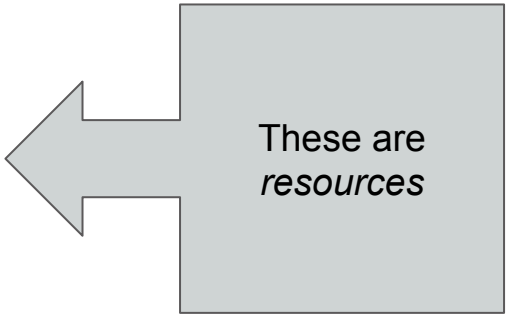
DIRAC
THE INTERWARE

Questions/comments

?

… a few examples of what DIRAC can be used for

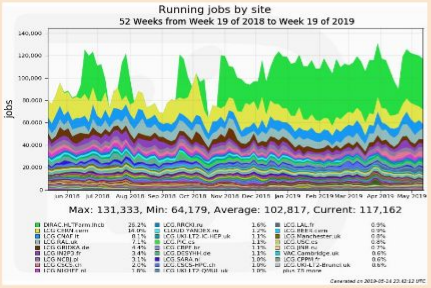
- sending jobs to “the Grid”
  - the obvious one...
- interfacing with different *sites*
  - with different *computing elements*
    - and *batch systems*
  - with different *storage elements*
- interfacing with different *information systems*
- interfacing with different *catalogs*
- interfacing with different *MQs, DBs*
- authenticate through different *providers*



- managing “productions” (e.g. reconstruction, simulation…)
- 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
- monitor your resources with a policy-based system
- … and more

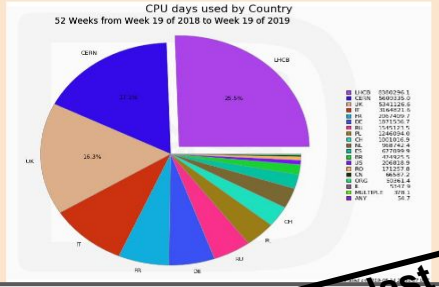
Managing *resources*

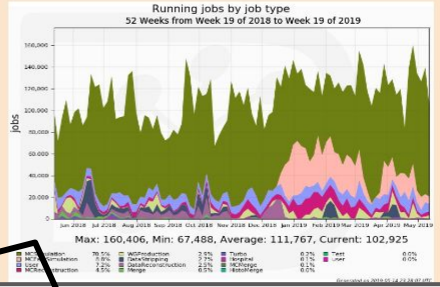
- Computing
  - **CEs**: ARC, CREAM, HTCondor, “SSH” for standalone BS, ...
  - **Batch**: LSF, BQS, SGE, PBS/Torque, SLURM, Condor,...
  - Clouds, BOINC, HPC, “desktops”
- Storage
  - SRM2, GSIFTP, XRoot, http, DIPs, …
    - EOS, Castor, DPM, dCache, StoRM, ECHO, CTA, …
- Catalog
  - DIRAC FC, LFC, (Rucio), [LHCb Bookkeeping], ...
- Information services
  - BDII, GOCDB, CRIC…
- IdProviders
- ProxyProviders
  - VOMS, OAuth2, PUSP...
- DBs, MQs, LogBackends
  - MySQL, Oracle, ElasticSearch
  - stomp → ActiveMQ, RabbitMQ
  - file, MQ, ES
    - and logs centralization is easy to set up



LHCb activities in the last year

- HLTfarm works even during data taking
- LHCb 25%
- CERN
- UK, IT, FR, DE
- RU, PL, CH, ……

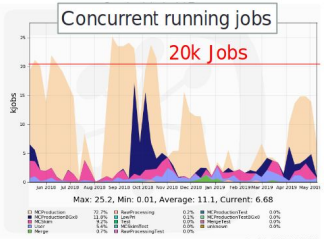


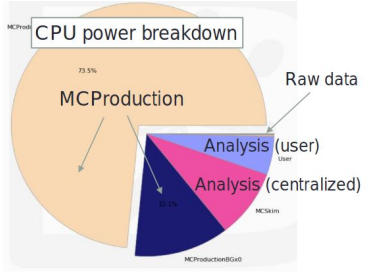


- MCSimulation
- MCFastSimulation
- User
- …..

From last DIRAC users' workshop

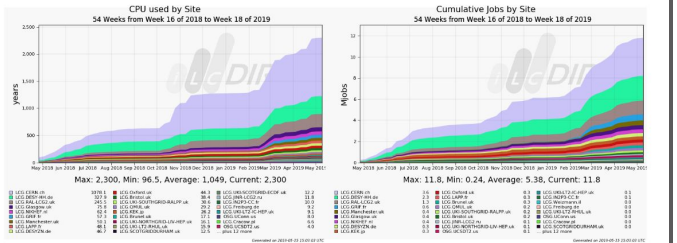
Belle II computing performance in a year





- Consumed CPU power is not so different from last year
- Resource usage is getting practical
  - Increasing analysis jobs
  - Increasing raw (beam and cosmic ray) data processing

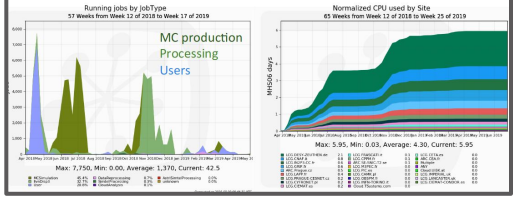
Usage



2.3k CPU Years, 11.8M Jobs

DIRAC usage since last year

- MC production and analysis running in parallel
- 144 M HS06 hours
- 1.9 M executed jobs



Business as usual - just more of it (Success!)