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

Experiment agnostic
Extensible
Flexible
Jobs and files

WMS and DMS
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/GSI/SSH 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**

  "Integrating LHCb workflows on HPC resources"

  14:45, R4
[DMS] Data Management

**Basics** of DMS:

- **LFNs**: 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 **SEs**.
    - *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)
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

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

files
/lhcb/user/f/fstagni/shoppingList.txt
/lhcb/user/f/fstagni/presents.docx

operations
1. Register in catalogs

files
/lhcb/user/f/fstagni/beach.jpg
/lhcb/user/f/fstagni/sunset.jpg
/lhcb/user/f/fstagni/beers.jpg

operations
1. COPY from A to B
2. REMOVE from A

Request #12345
Request #12346

A key system for job failovers, and for DMS operations
**DIRAC TS**  
**Transformation system**

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

<table>
<thead>
<tr>
<th>An <em>operation type</em> can be, e.g.:</th>
<th>A <em>dataset</em> is split into groups, based on criterias defined by <em>plugins</em>, e.g.:</th>
<th>A <em>system</em> is either (today) the DIRAC WMS (for productions) or the DIRAC RMS (for dataset management operation types)</th>
</tr>
</thead>
</table>
| ● a simulation workflow           | ● split by size  
● by destination  
● by metadata  
● ... [code it]                                                                 |                                                                                                                                 |
| ● a reconstruction workflow      |                                                                                                                                 |                                                                                                                                 |
| ● a replication                   |                                                                                                                                 |                                                                                                                                 |
| ● a removal                        |                                                                                                                                 |                                                                                                                                 |
| ● ...                              |                                                                                                                                 |                                                                                                                                 |

**[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

(Transformation APIs)

Configuration System
Resource Status System
Monitoring and Accounting System
Request mgnt System

TS

WMS

CEs

DIRAC Productions System:
chaining job productions together

TS1

CTA DIRAC example

DIRAC Transformation System + WMS:
for single productions

Configuration System
Resource Status System
Monitoring and Accounting System
Request mgnt System

TS

WMS

CEs

DIRAC Productions System:
chaining job productions together

TS1

CTA DIRAC example
[DMS] Dataset management

(dirac-transformation-replication)

Transformation System

Request Management System

Data Management System

Catalogs

SEs

Configuration System

Resource Status System

Monitoring and Accounting System

Catalogs
- **Web users’ interface**
- Frontend: ExtJS6
- Backend: tornado, NGINX

- Each system has its own Web application
Recent and ongoing developments
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
Current 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)


- "**DIRACOS: a cross platform solution for Grid tools**"
  - 7/11, 11:00, R3

- "**A gateway between Gitlab CI and DIRAC**"
  - Poster
~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
● 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

Outreach

10th DIRAC Users' Workshop
indico.cern.ch/e/DUW10
25-29 May 2020
KEK, Japan
… 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
- ......

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

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

DIRAC usage since last year

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

From last DIRAC users’ workshop

2.3k CPU Years, 11.8M Jobs
Experiment agnostic, and extensibility

"Horizontal" extensibility
- For specific requirements

"Vertical" extensibility
- Community driven