Using OSG Computing Resources with iLCDIRAC

Computing in High Energy Physics, Oct.10–14, 2016
San Francisco, California, USA

André Sailer, Marko Petric (CERN)
On Behalf of the CLICdp Collaboration

Motivation

The ILC VO is a merger of OSG and WLCG VOs for linear collider studies. Initial computing demands could be satisfied with the resources available in the WLCG via glite-WMS, CREAM, and ARC computing elements. However, the adoption of the iLCDirac tool by more and more members of the linear collider community, and the untapped but accessible resources in the OSG encouraged the development of the interfaces between DIRAC and the HTCondor-CE and Globus computing element middlewares.

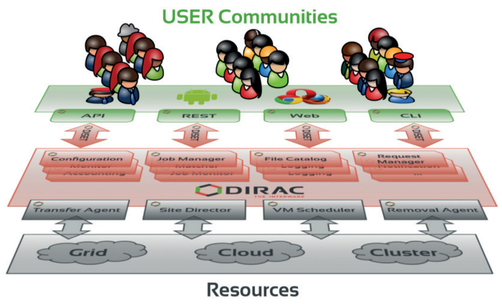
DIRAC

The DIRAC *interware* gives homogeneous access to heterogeneous resources

- Job environment provided through *Pilots*
- Common issues solved by DIRAC: Workload Management; central Productions; File Catalog; asynchronous Requests; access to different computing elements, batch systems, or clouds
- Can be extended to meet needs specific to VOs

Further information:

- DIRAC in Large Particle Physics Experiments; Oct 13, 14:00 Track 7
- diracgrid.org, http://github.com/DIRACGrid



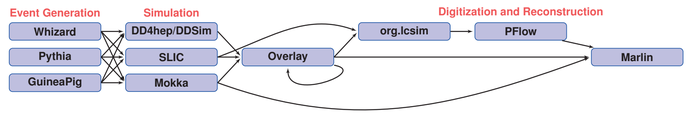
iLCDirac

iLCDirac is the extension for the Linear Collider community

- Python interface to run linear collider software
- Allow one to create chains of the linear collider applications
- Overlay system to choose random background samples for physics events

Further information:

- ILCDIRAC, a DIRAC extension for the Linear Collider community (CHEP 2013)




```
from DIRAC.Core.Base import Script
Script.parseCommandLine()
import UserJob
import Marlin
import DiracILC
d = DiracILC()
j = UserJob()
j.setOutputSandbox("recEvents.slcio")
m = Marlin()
m.setVersion("ILCSoft-01-17-09")
m.setSteeringFile("Steering.xml")
m.setInputFile("SimEvents.slcio")
j.append(m)
j.submit(d)
```

Using HTCondor-CEs inside DIRAC

DIRAC Computing Element API

Computing Element classes need to fulfil the following interface

- `submitJob`: Submit one or many pilots to given CE
- `getJobStatus`: Status of individual job on CE
- `getJobOutput`: Get pilot output and error file
- `getCEStatus`: Get running and pending jobs at CE
- `killJob`: kill the (pilot) job
- `getPilotLoggingInfo`: Get log for pilot

Submission and Configuration

- `condor_submit -terse subFile.jdl`

```
executable = DIRAC_XXXXXPilot.py
universe = grid
use_x509userproxy = true
output = $(Cluster).$(Process).out
error = $(Cluster).$(Process).err
log = $(Cluster).$(Process).log
environment = "HTCONDOR_JOBID=$(Cluster).$(Process)"
initialdir = %(initialdirOnSubmitServer)s
grid_resource = condor %(ceName)s %(ceName)s:9619
ShouldTransferFiles = YES
WhenToTransferOutput = ON_EXIT_OR_EVICT
kill_sig=SIGTERM
%(extraLine)s
Queue %(nJobs)s
```

- Pilot-executable bundled with proxy, created dynamically
- Further configuration – `extraLines` – via DIRAC ConfigSystem: Global, per Site, per CE
  - `periodic_remove=NumSystemHolds > 0`
  - `request_cpus = 8`
  - ...

Requirements

- DIRAC instance
- Running condor daemons (`condor_master`) on one of the DIRAC servers

Implementation

- The HTCondor-CE computing element class is implemented by calling plain condor commands
- Using raw commands for easier debugging; keep DIRAC and condor issues separate

Job Control

DIRAC Workload Management system takes care of matching payloads to jobs, pilot status is monitored automatically

- Environment variable `HTCONDOR_JOBID`, unique for each pilot, is used to identify pilots
- `condor_q` and `condor_history` to obtain pilot status
- `condor` to kill and remove *held* pilots to prevent restart
- Output log files automatically downloaded by HTCondor and passed into DIRAC on demand (see Monitoring to the right)

Problems and Open Issues

Problems:

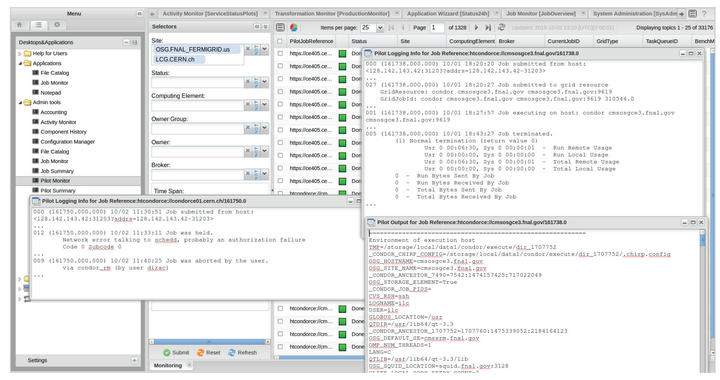
- Pilots and payloads do not support check-pointing, if a job is held it has to be killed
- `periodic_remove` needs to be tweaked for specific CEs

Open issues:

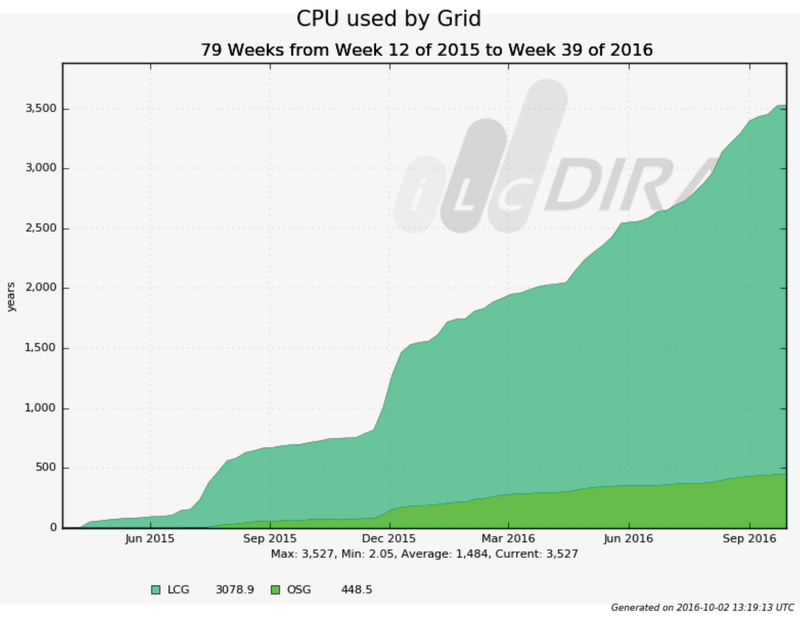
- Monitoring CE status: Query the CE how many jobs from given VO are running, currently using DIRAC's own count of pilots at CE

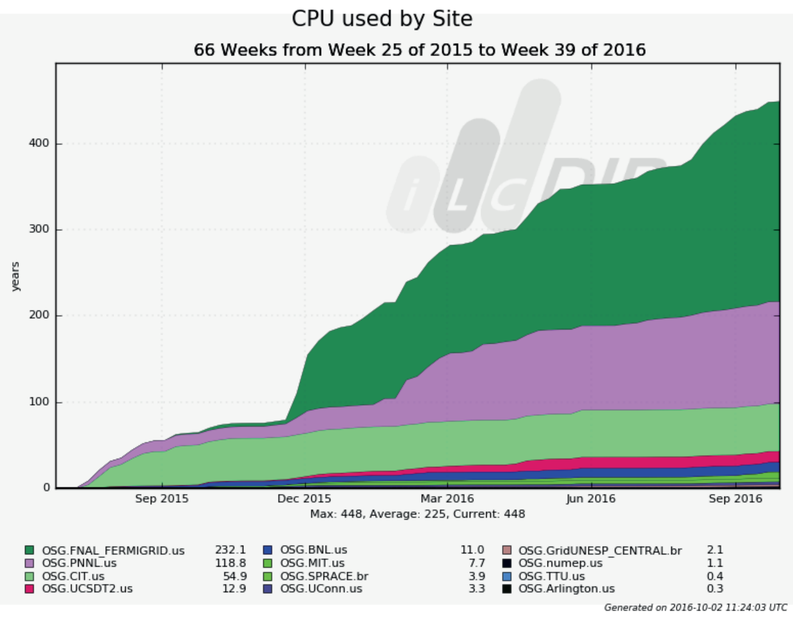
Monitoring

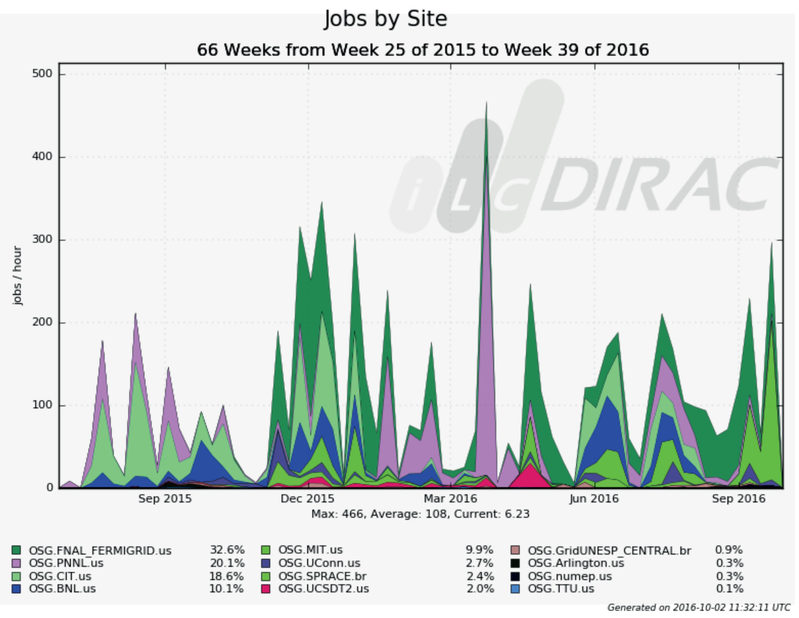
Monitor pilot status, read log files and output via the DIRAC WebApp



OSG Resource Usage by the ILC VO







Globus CE

- Link from DIRAC to Globus CEs implement via `globus-job-*` commands
  - globus-job-submit, globus-job-clean, globus-job-status, globus-job-get-output
- Number of Globus CEs in decline since HTCondor-CE available
  - Only one (1) left for the ILC VO

Summary

- OSG Computing Elements (HTCondor-CE, Globus) fully integrated in DIRAC
- Minimal effort for Dirac instance administrators to use HTCondor-CEs or Globus CEs
- Completely transparent to end users

Acknowledgement

Thanks to Iain Steers (CERN), the HTCondor team, and the DIRAC developers for their advice and support.

Mail: andre.sailer@cern.ch, marko.petric@cern.ch
WWW: http://ilcdirac.cern.ch