Using GridPP DIRAC via Ganga

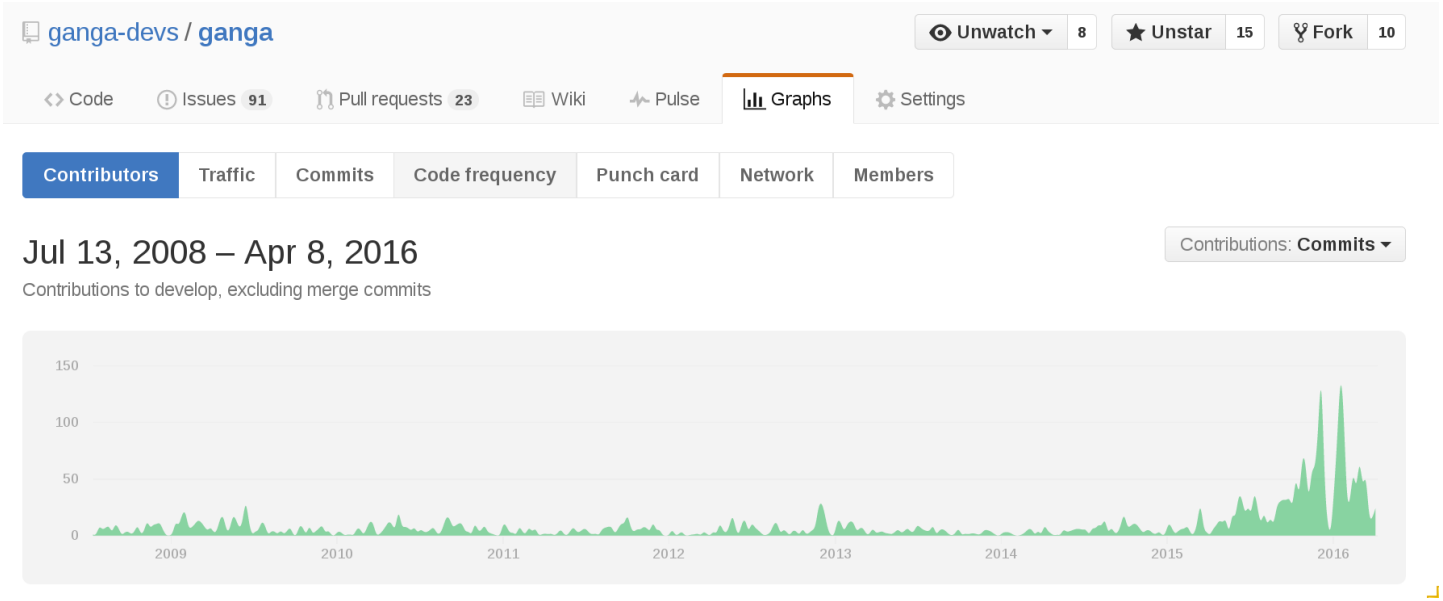
on behalf of:
Alex Richards, Mark Slater, Matthew Williams, **Robert Currie**, Ulrik Egede

Contents

1. Latest news from Ganga
2. Working with new users/communities
3. Short how-to on submitting a job

Recent news from Ganga

- Now migrated to github:
  `http://www.github.com/ganga-devs/ganga`



- documentation at:
  `https://github.com/ganga-devs/ganga/wiki/Full-Tutorial`
- More documentation as of 6.1.19 at:
  `http://ganga.readthedocs.org`

Recent news, cont.

- Ganga now available via "`pip install`"
- "`import ganga`" has been improved
  Jobs can now be controlled/submitted in a standard python environmant/script.
- Continuing to improve performance and scalability
- Migrate to Python 2.7 as minimal requirement in near feature.
  Allow to use new features and for releases that work for both Python 2.X and Python 3.X

These changes have been focussed on making Ganga a tool for supporting a broad community rather than just the LHC experiments.

Outreach, LSST a new-user use case

LSST (Large Synoptic Survey Telescope) is a project which aims to map the night sky and to take a census.

Over the past couple of months we have begun working with LSST Physicists within the UK to allow them to perform their analyses using the GridPP resources.

LSST project will generate a large dataset which needs to be processed.

This work is strongly limited by the amount of CPU time available.

Current workflow makes use of a single batch system in the US. Want to take advantage of UK resources.

LSST new user problems

Common problems being faced by LSTT developers moving to the grid:

- Determining if jobs have successfully completed can be difficult.
- Test runs are having to manage $\sim 4,000$ files on grid storage. This will escalate $\times 1,000$ during data taking.
- Final output ($>$ 10Gb in $> 1,000$files) needs to be collected and merged for further processing.

Reasons for LSST users to prefer Ganga

Some of the advantages Ganga offers LSST developers:

- More advanced job management, grouping and accounting
- Low barrier to entry for submitting jobs to the Grid, Batch, Localhost, etc.
- Automated tool for collecting/managing data across different storage solutions
- Automatic credential management

Using GridPP with Ganga (Requirements)

To use Ganga with GridPP Dirac the following is recommended:

1. Valid grid credentials & Python (preferably version 2.7)
2. CernVM instance or an install of a SLC6 Machine
3. Use of a cvmfs installation for both Dirac and Ganga just run:
   `/cvmfs/ganga.cern.ch/runGanga-dirac.sh`

Now you can submit jobs to the GridPP Dirac.

Using GridPP with Ganga, cont.

- Alternatively, you can install ganga and Dirac by hand
- For non *gridpp_user* accounts using DIRAC:
  See the "`.gangarc`" file in the appendix of this talk
- Ganga is highly customisable:
  Single instance can submit and manage Jobs to Batch cluster, DIRAC, Localhost, etc.
- One central configuration can be used to configure Ganga for all users at a given site

GridPP Dirac “hello world”

```
*** Welcome to Ganga ***
Version: 6.1.18
Documentation and support: http://cern.ch/ganga
Type help() or help('index') for online help.

This is free software (GPL), and you are welcome to redistribute it
under certain conditions; type license() for details.


Ganga.Utility.Config                : INFO     reading config file /home/rcurrie/.gangarc

[07:07:40]
Ganga In [1]: j=Job()

[07:07:46]
Ganga In [2]: j.backend=Dirac()

[07:07:50]
Ganga In [3]: j.submit()
07:07:54,972 Ganga.GPIDev.Lib.Job                : INFO    : submitting job 0
...

07:08:23,186 Ganga.GPIDev.Lib.Job                : INFO    : job 0 status changed to "completed"

[07:08:23]
Ganga In [4]: jobs(0).peek('Ganga_Executable.log')
<<<<<<<<<< exe-script.py Standard Output >>>>>>>>>>

Hello World
```

Benefits for for LSST users

With some further integration Ganga LSST users are able to configure their jobs as follows:

```
[07:07:40]
Ganga In [1]: j=Job()

[07:07:46]
Ganga In [2]: j.backend=Dirac()

[07:07:50]
Ganga In [3]: j.application = Im3Shape(location=DiracFile(lfn=‘‘/some/LFN’’))

[07:07:54]
Ganga In [4]: j.splitter = Im3ShapeSplitter(size=5)   # Split each file by 5

[07:07:58]
Ganga In [4]: j.dataset = GangaDataset(lfnList)   # 4,000 LFN on Grid Storage

[07:08:02]
Ganga In [4]: j.submit()   # Will submit 20,000 DIRAC jobs managed as 1 Ganga job
```

This allows the LSST users to submit $\sim 20,000$ jobs in a way which is managed by Ganga's Job repository.

Conclusions

- Ganga is now hosted on github
- Ganga is the recommended route for new users to the Grid
- Ganga offers advanced methods for job management for users
- Ganga can manage job configurations/recipes for posterity
- We encourage people to try out Ganga 6.1.18, soon 6.1.19
- Large scale shared MC productions (and similar) with uniform workflow may prefer to use the Dirac tools directly.

Thanks for listening!

Appendix

Configuring Ganga for use with GridPP Dirac

Copy this config to $HOME/.gangarc and change <dirac_user_group>

```
[Configuration]
RUNTIME_PATH = GangaDirac

[LCG]
GLITE_ENABLE = True
GLITE_SETUP = /cvmfs/ganga.cern.ch/dirac_ui/bashrc

[DIRAC]
DiracEnvFile = /cvmfs/ganga.cern.ch/dirac_ui/envfile

[defaults_GridCommand]
info = dirac-proxy-info
init = dirac-proxy-init -g <dirac_user_group> -M
```