Tutorial for LHCb

Will Reece

36th Software Week

Imperial College London

What is the Grid?

- A way to access resources across the world in a seamless way.
- CPU resources (Computing Elements) at sites across the world
- Disk pools across the globe (Storage Elements)
- Only a single “login” required to give you access. After identification security is taken care of by the system.
- Several flavors of the Grid available; however, in LHCb we (at the moment) only have to deal with the LCG (LHC computing Grid).

List of LCG Grid Sites:

CERN.ch, CNAF.it, GRIDKA.de, IN2P3.fr, NIKHEF.nl, PIC.es, RAL.uk, ACAD.bg, APC.fr, Barcelona.es, Bari.it, BHAM-HEP.uk, BMEGrid.hu, Bologna.it, Bristol-HPC.uk, Bristol.uk, Brunel.uk, Cagliari.it, Cambridge.uk, Catania.it, CBPF.br, CESGA.es, CGG.fr, CNAF-GRIDIT.it, CNAF-T2.it, CNR-ILC-PISA.it, CPPM.fr, CSCS.ch, DESY.de, Dortmund.de, EFDA.uk, ESA-ESRIN.it, Ferrara.it, FESB.hr, GR-04.gr, GR-05.gr, IFH.de, IFJ-PAN.pl, IHEP.su, Il-BGU.il, Imperial.uk, IN2P3-T2.fr, INR.ru, IPP.bg, IPSL-IPGP.fr, IRB.hr, ITEP.ru, ITEPnew.ru, ITPA.lt, ITWM.de, JINR.ru, KIAE.ru, KIAM.ru, Krakow.pl, LAL.fr, Lancashire.uk, LAPP.fr, Legnaro.it, Liverpool.uk, LNS.it, LPC.fr, LPN.fr, LPNHE.fr, LT2-IC-HEP.uk, Manchester.uk, Milano.it, MPI-K.de, MPI-RZG.de, NAPOLI-ARGO.it, NAPOLI-ATLAS.it, NAPOLI-CMS.it, NAPOLI-PAMELA.it, Napoli.it, NCP.pk, NGCC.bg, NIPNE-07.ro, NIPNE-11.ro, Oxford.uk, Padova.it, PAKGRID.pk, Pisa.it, PNPI.ru, Poznan.pl, QMUL.uk, RAL-HEP.uk, RHUL.uk, SARA.nl, Sheffield.uk, SINP.ru, SNS-PISA.it, Sofia.bg, SPACI-LECCE.it, SPBU.ru, SRCE.hr, TCD.ie, Torino.it, Trieste.it, UCL.uk, UKI-LT2-Brunel.uk, UKI-LT2-IC-HEP.uk, UKI-LT2-IC-LeSC.uk, UKI-LT2-QMUL.uk, UKI-LT2-RHUL.uk, UKI-SCOTGRID-DURHAM.uk, UKI-SCOTGRID-ECDF.uk, UKI-SCOTGRID-GLASGOW.uk, UNIZAR.es, USC.es, WCSS.pl, WEIZMANN.il

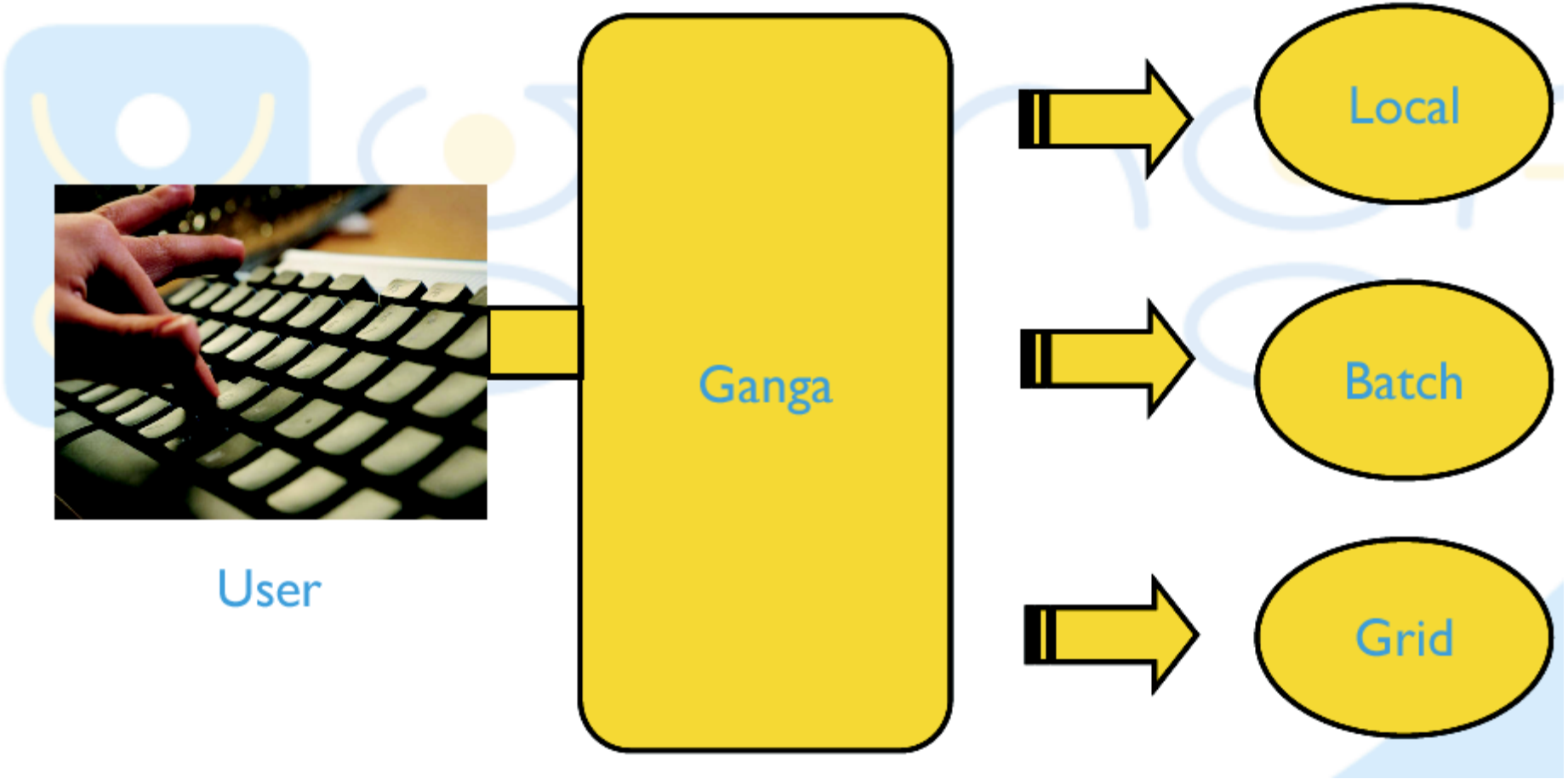
Grid Resources

- Your “grid certificate” is what gives you a unique identification on the Grid.
- At the basic level just 2 files in your ~/.globus directory.
- By sending a “grid proxy” along with a job you allow computers for a limited time to act on your behalf. This lets you run jobs at LCG sites and to read(write) files from(to) remote storage.
- You obtain a proxy for a finite time. Jobs running on the grid will keep running beyond the end of the proxy validity, but you need a new one to obtain the results.

What is Ganga?

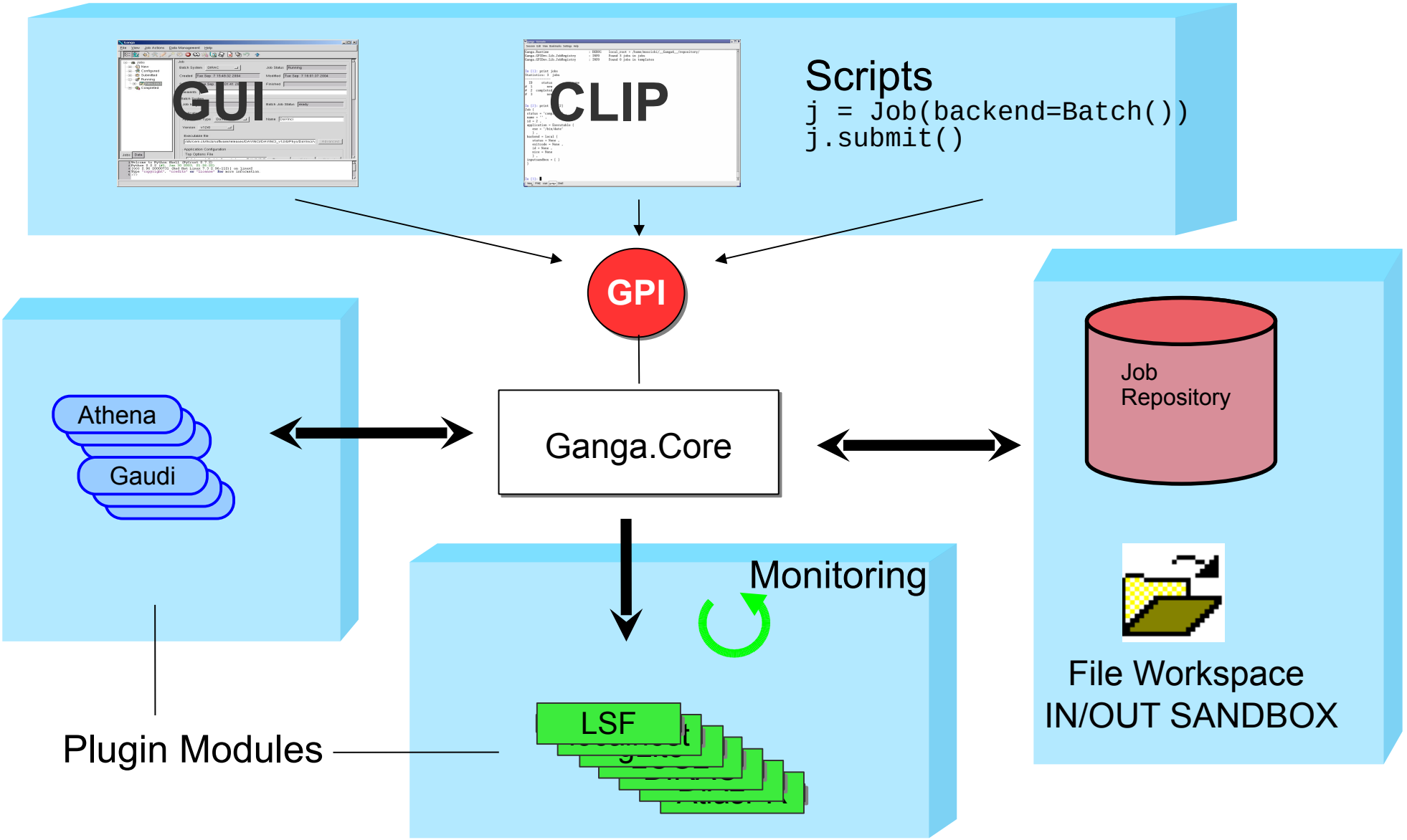
- Started off as a Atlas/LHCb project
- Ganga is an application to enable a user to perform the complete life cycle of a job
- Build – Configure – Prepare – Monitor – Submit – Merge – Plot
- Run on the local machine (interactive or in background), Batch systems (LSF, PBS, SGE, Condor) or Grid systems (LCG, gLite, NorduGrid)
- Interfaces with workload management systems (Dirac, Panda)
- Jobs look the same whether the run locally or on the Grid!

The Ganga Mantra

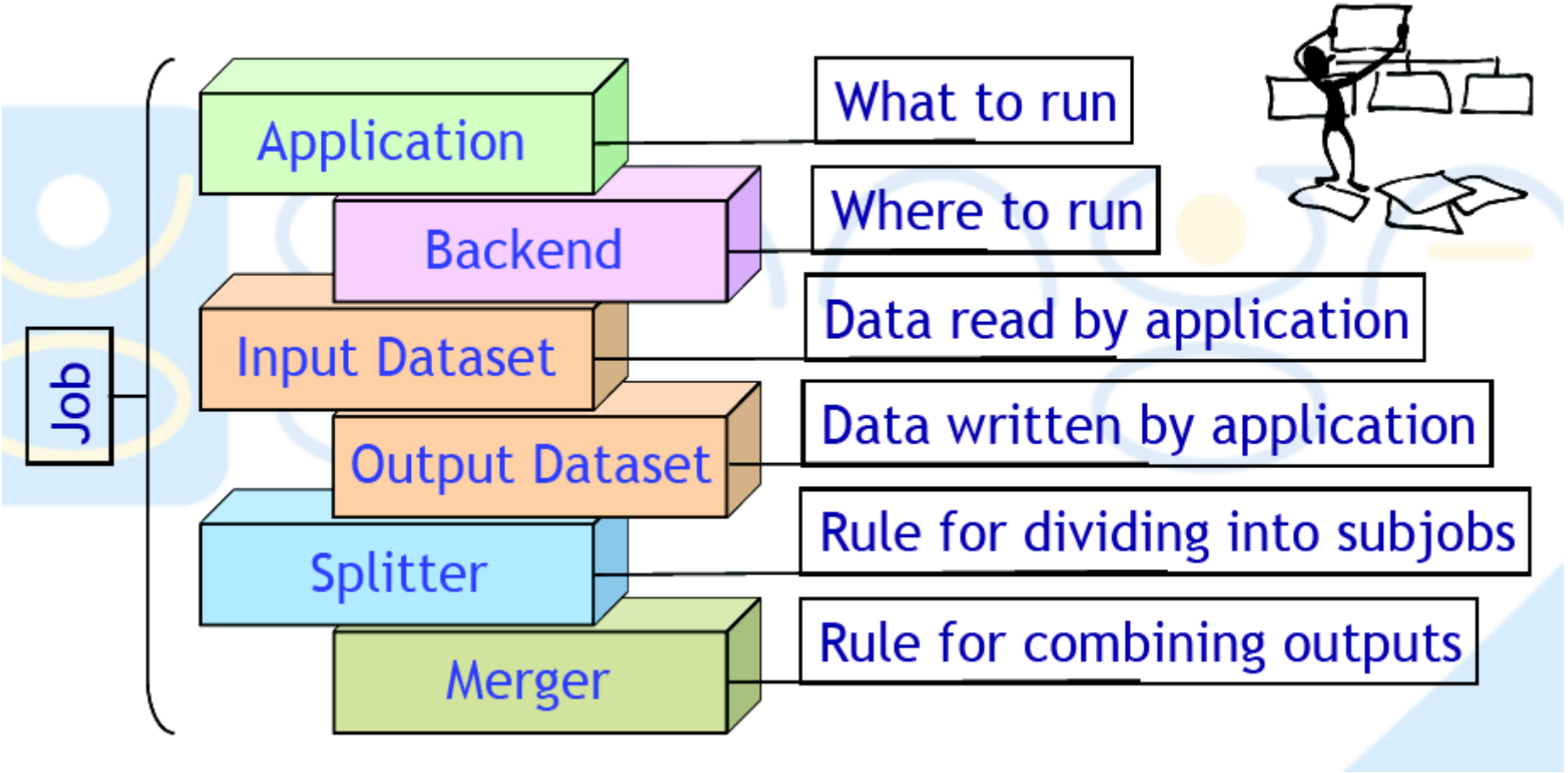


Configure once, run anywhere!

Ganga Architecture



The Ganga Job Object



Running Ganga

Set up the Ganga environment, on LHCb account on lxplus

```
% GangaEnv
```

Pick the default version (unless specifically instructed otherwise)

```
% ganga
    *** Welcome to Ganga ***
    Version: Ganga-5-1-3
    Documentation and support: http:/cern.ch/ganga
    Type help() or help('index') for online help.
```

Type your grid password to obtain a grid certificate (or hit Ctrl-D several times to bypass this).

To quit, type Ctrl-D.

Ganga Prompt

Ganga is based on Python and has an enhanced Python prompt (ipython)

Python programming/scripting:

```
myvar = 5
print myvar*10
```

Easy access to shell commands in Ganga:

```
!less ~/.gangarc
!pwd
```

History (<arrow-up>), TAB completion on keywords, variables, objects, try:

```
my<TAB>
```

Plus many more features!

Python Syntax

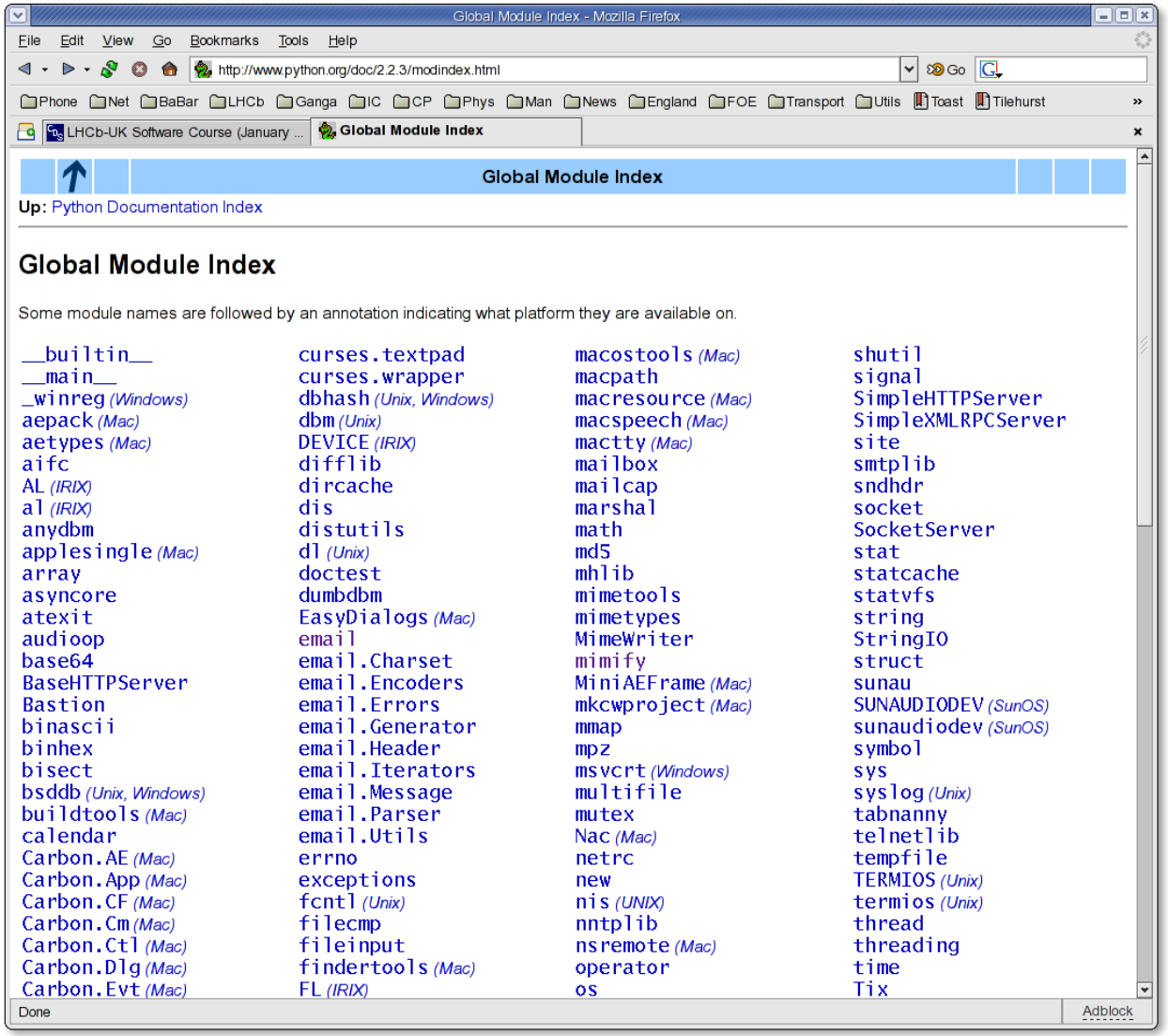
```
# variables
x = 2
print x*3 # This is a comment!
alist = [1,2,3]
for y in alist:
  print y          # NOTE INITIAL SPACES!
# functions
def square(v):
  return v*v
print square(2)
# modules
import os
print os.environ['HOME']
```

For more information on python see the excellent tutorial

http://docs.python.org/tut/tut.html (python version specific)

Python: Modules

- Very large number of Python modules available.
- Check if somebody else has written it before you go ahead.
- Python documentation is excellent.



Job Objects and Job Registry

```
j = Job()
print j
print j.application
print j.backend
# job registry: a list of all your jobs
jobs
# Get reference to job with id=2
j = jobs(2)
# Get reference to the last job
j = jobs[-1]
# Get more information - can be lots of output
full_print(j)
```

Job Create/Submit

```
[In:] j = Job()
[In:] j.application = Executable(exe='/bin/hostname')
[In:] j.name = "MyTest"
[In:] print j
[In:] j.submit()

# wait until job is completed and look
# at the output directory

[In:] j.status
[In:] j.peek()
[In:] j.peek('stdout')

# The syntax for peek is very flexible see
[In:] help(j.peek)
```

More on Job Submit

```
# once a job is submitted you cannot modify
# it. if you want to submit a new job you
# should create a new job object

[In:] j2 = j.copy()
[In:] j2.backend = Batch()#Default is LSF at CERN
[In:] j2.submit()


# if you have GRID certificate you can try

[In:] j3 = j.copy()
[In:] j3.backend = Dirac()
[In:] j3.submit()

# print jobs to see all your jobs
```

More with Ganga

```
# see predefined plugins
# (only some combinations supported)
plugins('applications')
plugins('backends')
```

```
# or using a splitter to create subjobs
list = [["hello", str(i)] for i in range(5)]
j = Job()
j.application.exe='/bin/echo'
j.splitter=ArgSplitter(args=list)
j.submit()
```

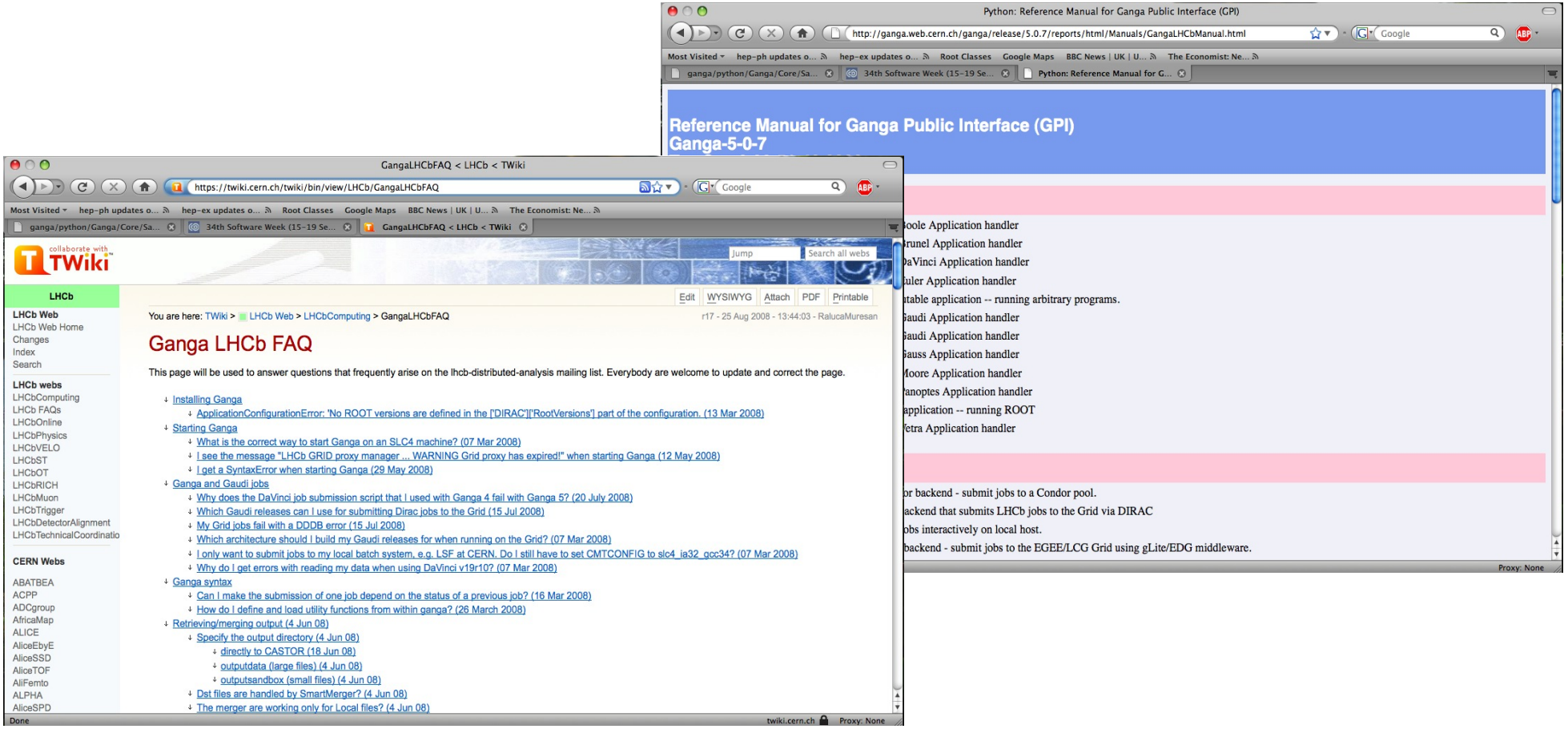
Job Templates

```
# For frequently used jobs - use templates!
t = JobTemplate(j3)
t.name = 'MyTemplate'
# Persisted across sessions
print templates
j = Job(templates(1)) # By ID
j2 = Job(templates['MyTemplate']) # By name

# Other useful methods
j.kill()
j.remove() # Removes the output as well!
slice = jobs.select(status='failed')
slice.remove() # delete all failed jobs
```

Ganga: Help+Debug

```
# ganga has online help
help('index') # See all the ganga objects

j = Job(backend=Dirac(), application=Root())

# How do I set the CPU time for my job?
help(j.backend)# I use CPUTime!

# ganga (unlike python) is typesafe
j.application.version = 51800 # WILL FAIL!!!
j.application.version = '5.18.00' # That's better!

# What methods?
dir(j.backend)
help(j.backend.getOutput)# That's how I use it!
```

Ganga Docs



- Lots of LHCb documentation at http://cern.ch/ganga
- FAQ: https://twiki.cern.ch/twiki/bin/view/LHCb/GangaLHCbFAQ
- Mailing list: lhcb-distributed-analysis@cern.ch

LHCb Jobs

For LHCb the main use of Ganga is for running Gaudi jobs.

This Includes:

- Configuring Gaudi jobs
- Specify the datasets
- Run the jobs locally, on batch systems and on the Grid via Dirac
- Managing the output data, Ntuples and histogram files.

Gaudi App Handlers

There is a specific application handler for each Gaudi app:
['Brunel', 'Moore', 'DaVinci', 'Panoptes', 'Gauss', 'Boole', 'Vetra', 'Euler']

```
# Define a DaVinci application object
d = DaVinci()
d.optsfile = d.user_release_area + \
  'Phys/DaVinci/options/myopts.py'
d.extraopts = '''
ApplicationMgr().EvtMax = 1000
HistogramPersistencySvc().OutputFile = "DVHistos_1.root"
'''
# extraopts must use Python options.
# Use triple quotes ('''') - allows multi-line
# and use of un-escaped quotes.
```

GaudiPython

```
# The use of GaudiPython is also supported directly
g = GaudiPython()
print g
GaudiPython (
 project = 'DaVinci' ,
 platform = 'slc4_ia32_gcc34' ,
 version = 'v21r0' ,
 setupProjectOptions = '' , # passed to SetupProject
 script = [] # see below for details
 )
j = Job(application=g, backend=Dirac())
# main script controls event loop - other files
# added to inputsandbox - must import manually
```

Backends

There are 4 backends of interest for running LHCb jobs:

- Interactive – in the foreground on the client
- Local – in the background on the client
- LSF – on the LSF batch system (SGE/PBS/Condor systems supported as well)
- Dirac – on the Grid via DIRAC

```
# Define a Dirac backend object
d = Dirac()
print d
Out[34]: Dirac (
 status = None ,
 destination = None ,
 id = None
 )
```

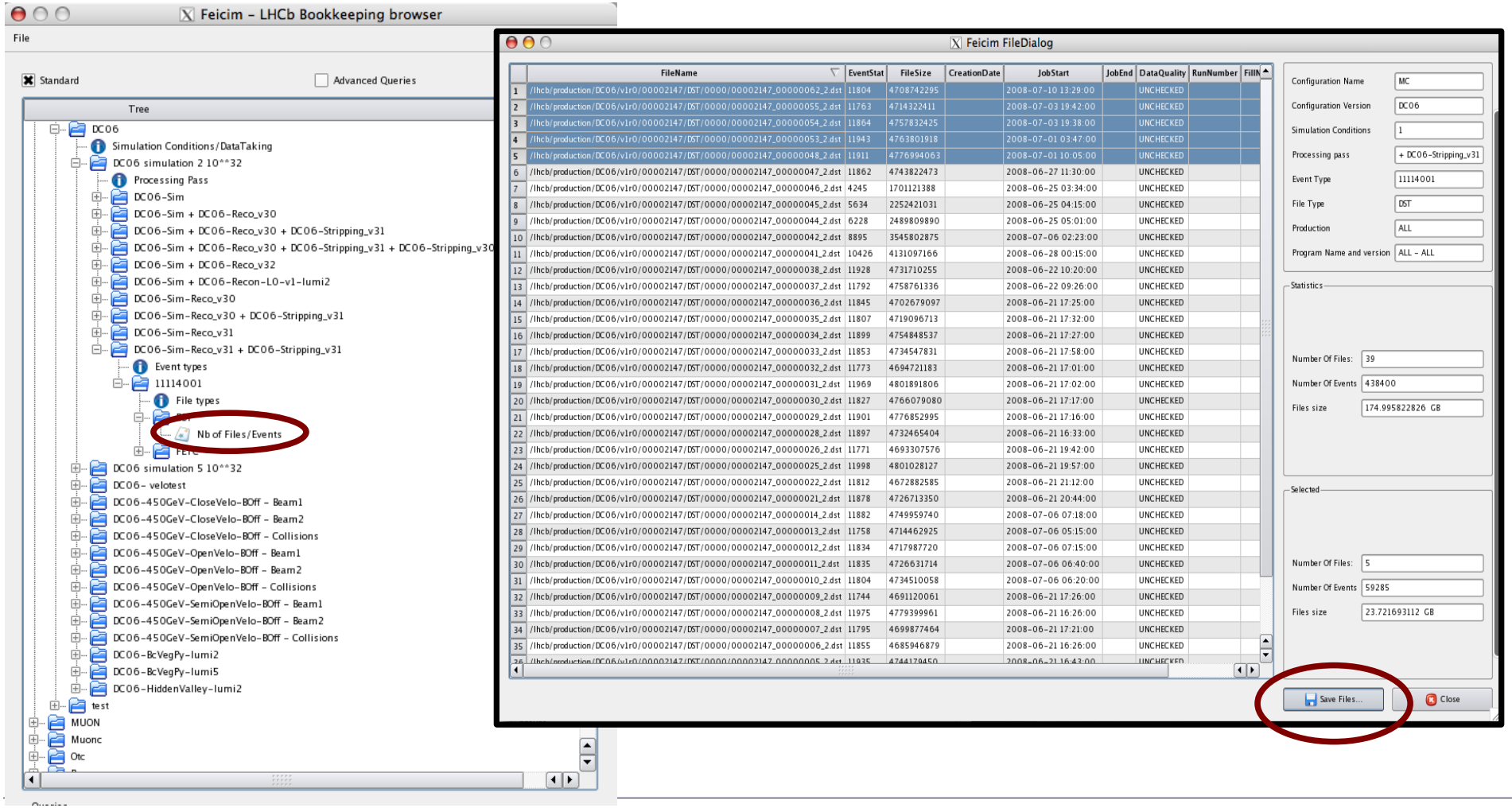
Split/Merge

- Run many similar jobs (subjobs) with different input
- Running over data files on the Grid? Use the DiracSplitter – Uses the LFC to group files smartly
- Running lots of ToyMC with RooFit? Use the Root application with the ArgSplitter
- Produces lots of files that need merging together
- See help for RootMerger, TextMerger and DSTMerger, SmartMerger and MultipleMerger also useful
- Want something really special? See the CustomMerger.

Inputdata

Can use the LHCb bookkeeping to get a list of files to run over

```
j.inputdata = browseBK() # opens BK browser
```



LHCb Jobs

Putting together an LHCb job just involves combining the parts

```
# Put together an LHCb job and submit
dv = DaVinci(...)
d = Dirac(...)

j = Job(name='MyJob', application=dv, backend=d)
print j
Out[38]: Job (
 status = 'new' ,
 name = 'MyJob' ,
 application = DaVinci (...)
 backend = Dirac (...)
 ...
)
j.submit()
```

Execution/Finalization

- The progress of a job is monitored by Ganga
- Looking at the overview (jobs) or the status field for the backend will show the current status.
- When a job is finished the outputdir will contain the stdout and stderr of the job and your output sandbox files. Output data files are stored in a storage element on the Grid.
- Large files are uploaded to a storage element - Download with j.backend.getOutputData (Dirac only)
- You can build a list of LFNs of these files – j.backend.getOutputDataLFNs (Dirac only)

*** Use job.outputsandbox and job.outputdata to determine which files go where!**

```
e.g. job.outputsandbox = ['histos.root']  # to send histos.root to output dir
     job.outputdata = ['histos.root']     # to send histos.root to mass storage
```

Dirac Monitoring

Jobs monitoring as lhcb@LHCb-Production

https://lhcbweb.pic.es/DIRAC/LHCb-Production/lhcb/jobs/JobMonitor/display

Systems Jobs Data Web — Selected setup: LHCb-Production

DIRAC SideBar

Selections

DIRAC Site: All

Status: All

Minor status: All

Owner: wreece

JobGroup: All

Date: YYYY-mm-dd

JobID:

Submit Reset

Global Sort

Selected Statistics

Global Statistics

Job Monitoring

Select All Select None — Reset Kill Delete

JobId	Status	MinorStatus	ApplicationStatus	Site	JobName	LastUpdate [UTC]	LastSignOfLife [UT	SubmissionTime [U	Owner
922979	Failed	Maximum of resche	Unknown	LCG.GRIDKA.de	Ganga_ROOT_5.1	2008-12-05 23:19	2008-12-05 23:19	2008-12-05 10:23	wreece
920882	Failed	Uploading Job Outp	/afs/cern.ch/user/w/	LCG.UKI-SCOTGR	Ganga_ROOT_5.1	2008-12-05 21:09	2008-12-05 21:09	2008-12-05 09:19	wreece
919804	Killed	Marked for terminat	unknown	ANY	Ganga_ROOT_5.1	2008-12-05 09:18	2008-12-05 09:18	2008-12-05 08:46	wreece
910243	Done	Execution Complete	/scratch/z5/wreece/	LCG.RAL.uk	Ganga_Gauss_v35	2008-12-05 07:23	2008-12-05 07:23	2008-12-04 13:10	wreece
910232	Done	Execution Complete	/scratch/z5/wreece/	LCG.RAL.uk	Ganga_Gauss_v35	2008-12-05 05:25	2008-12-05 05:25	2008-12-04 13:10	wreece
910221	Done	Execution Complete	/scratch/z5/wreece/	LCG.PIC.es	Ganga_Gauss_v35	2008-12-05 08:26	2008-12-05 08:26	2008-12-04 13:09	wreece
910209	Done	Execution Complete	/scratch/z5/wreece/	LCG.PIC.es	Ganga_Gauss_v35	2008-12-05 06:23	2008-12-05 06:23	2008-12-04 13:09	wreece
910199	Done	Execution Complete	/scratch/z5/wreece/	LCG.CNAF-T2.it	Ganga_Gauss_v35	2008-12-05 09:09	2008-12-05 09:09	2008-12-04 13:09	wreece
910188	Done	Execution Complete	/scratch/z5/wreece/	LCG.RAL.uk	Ganga_Gauss_v35	2008-12-05 04:49	2008-12-05 04:49	2008-12-04 13:08	wreece
910177	Done	Execution Complete	/scratch/z5/wreece/	LCG.GRIDKA.de	Ganga_Gauss_v35	2008-12-05 05:58	2008-12-05 05:58	2008-12-04 13:08	wreece
910166	Done	Execution Complete	/scratch/z5/wreece/	LCG.GRIDKA.de	Ganga_Gauss_v35	2008-12-04 23:24	2008-12-04 23:24	2008-12-04 13:08	wreece
910154	Done	Execution Complete	/scratch/z5/wreece/	LCG.GRIDKA.de	Ganga_Gauss_v35	2008-12-04 21:24	2008-12-04 21:24	2008-12-04 13:07	wreece
910145	Done	Execution Complete	/scratch/z5/wreece/	LCG.RAL.uk	Ganga_Gauss_v35	2008-12-05 05:31	2008-12-05 05:31	2008-12-04 13:07	wreece
910134	Done	Execution Complete	/scratch/z5/wreece/	LCG.PIC.es	Ganga_Gauss_v35	2008-12-05 08:16	2008-12-05 08:16	2008-12-04 13:07	wreece
910122	Done	Execution Complete	/scratch/z5/wreece/	LCG.PIC.es	Ganga_Gauss_v35	2008-12-05 08:19	2008-12-05 08:19	2008-12-04 13:07	wreece
910112	Done	Execution Complete	/scratch/z5/wreece/	LCG.GRIDKA.de	Ganga_Gauss_v35	2008-12-04 21:36	2008-12-04 21:36	2008-12-04 13:06	wreece
910099	Done	Execution Complete	/scratch/z5/wreece/	LCG.GRIDKA.de	Ganga_Gauss_v35	2008-12-04 21:36	2008-12-04 21:36	2008-12-04 13:06	wreece
910087	Done	Execution Complete	/scratch/z5/wreece/	LCG.GRIDKA.de	Ganga_Gauss_v35	2008-12-04 21:04	2008-12-04 21:04	2008-12-04 13:06	wreece
910076	Done	Execution Complete	/scratch/z5/wreece/	LCG.GRIDKA.de	Ganga_Gauss_v35	2008-12-05 07:42	2008-12-05 07:42	2008-12-04 13:05	wreece

Page 1 of 13 — Items displaying per page: 25 — Displaying 1 - 25 of 318

jobs > Job monitor — wreece@ lhcb (/DC=ch/DC=cern/OU=Organic Units/OU=Users/CN=wreece/CN=667821/CN=Will Reece)

Done — lhcbweb.pic.es — Proxy: None

Getting Organized

- Ganga has logical folders organized in the jobtree
- Very similar to symbolic links in a file system.
- A job can be in several places e.g.: folder ‘FailedJobs’, folder ‘GaussJobs’, folder ‘TutorialJobs’

```
# Available methods for
dir(jobtree)
Out[85]: ['add', 'cd', 'cleanlinks', 'copy',
'exists', 'find', 'getjobs', 'isdir', 'listdirs',
'listjobs', 'ls', 'mkdir', 'name', 'printtree',
'pwd', 'rm']
```

Hints

Have a look also in the config file: ~/.gangarc. Many default settings can be changed if required.

See also the config object in ganga:

```
config.DIRAC
DIRAC : Parameters for DIRAC
    AllowedPlatforms = ['slc4_ia32_gcc34']
        Allowed platforms for submission to DIRAC
        Type: <type 'list'>
   DIRACsite = ''
        Used for testing only
        Type: <type 'str'>
    DiracLoggerLevel = 'ERROR'
        The logging level of DIRAC, not the logging level of the Dirac
        plugin in Ganga
        Type: <type 'str'>
```

Get used to using scripts and templates - saves lots of typing and errors!

Summary

The LCG provides LHCb users with a wealth of CPU power and storage space.

Ganga allows users to seamlessly run jobs locally, on batch systems or on “The Grid”.

Ganga is written in Python making its syntax is easy to understand.

Ganga seeks to provide as much help to its users as possible via manuals, online FAQ's, etc.

Ganga provides a number of specific tools for running LHCb jobs.

Try getting started with the first (more to come) “hands on” Ganga tutorial:
https://twiki.cern.ch/twiki/bin/view/LHCb/GangaTutorial1