



Ganga user interface for job definition and management

U.Egede¹, K.Harrison², R.W.L.Jones³, D.Liko⁴, A.Maier⁴, J.T.Mościcki⁴, G.N.Patrick⁵, A.Soroko⁶, C.L.Tan⁷

¹Imperial College London, ²University of Cambridge, ³University of Lancaster, ⁴CERN, ⁵RAL, ⁶University of Oxford, ⁷University of Birmingham

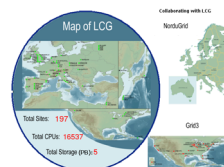
HEP Computing



Uniform User Interface

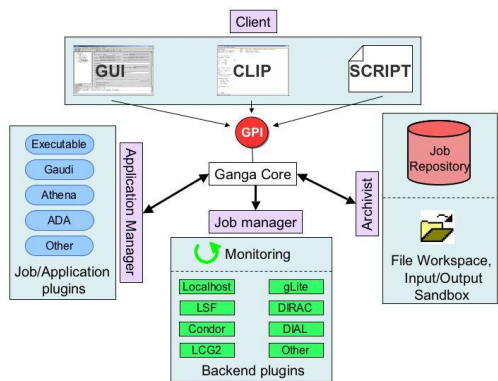
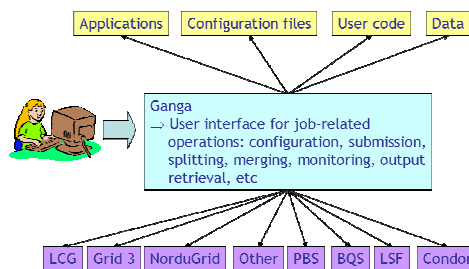


Worldwide Resources



Scope

Ganga is an easy-to use frontend for job definition and management. It allows the user to specify the software to run, which may include user code, to set values for any configurable parameters, to select data for processing, and to submit jobs to a variety of local batch queues and Grid-based systems, hiding Grid technicalities. Ganga is being developed in the context of two High-Energy Physics (HEP) experiments, ATLAS and LHCb, whose specific needs it addresses, but offers possibilities for extension and customisation that make it potentially interesting for a wide range of user communities.



Design

Ganga is implemented in Python, and has an extensible architecture. The main functionality is divided between three components:

- the Application Manager deals with defining the task to be performed within a job;
- the Job Manager creates a wrapper script for running the specified application, performs job submission to a chosen processing system (backend), monitors the job progress, and retrieves output files when the job completes.
- the Archivist stores job information for subsequent retrieval, and manages the Ganga workspace.

Components communicate with one another through the Ganga Core, and their functionality is made available through the Ganga Public Interface (GPI). Users access the GPI through a fourth Ganga component, the Client, which provides a Graphical User Interface (GUI), a shell – the Command Line In Python (CLIP) – and the possibility to run GPI scripts

User view

Different types of application and backend are supported in Ganga through plug-in classes. These define each application and backend in terms of its own schema, placing in evidence the configurable properties and their meanings. Users can easily add new plug-in classes, or suppress the loading of unwanted classes. With CLIP or GPI, a user needs only a few commands to define a job and run it on a Grid backend, with the output retrieved automatically. Job definitions in Ganga can readily be copied for re-use, stored as templates, or exported for sharing with other users. A GUI, to further simplify user interaction with Ganga, is under development.

```

Ganga
File Edit View Terminal Go Help

In [1]: status = os.system( "/bin/echo '/bin/hostname -f' - /bin/date" )
surv04.hep.phy.cam.ac.uk - Fri Sep 9 12:19:35 BST 2005

In [2]: j = Job( application = Executable(), backend = LCG(), name = "HelloFromRemoteHost" )

In [3]: j.application.exe = "/bin/echo"

In [4]: j.application.args = [ "Hello from '/bin/hostname -f' - /bin/date" ]

In [5]: status = j.submit()
Ganga.GPISubmitLib.Job      : INFO   submitting job 1

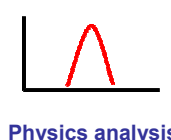
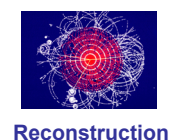
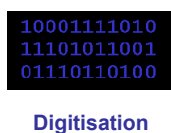
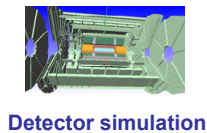
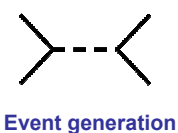
In [6]: Ganga.Lib.LCG      : INFO   Job 1 Ready at mu6.matrix.sara.nl:2119/jobmanager-pbs-atlas - Fri Sep 9 12:20:13 2005
Ganga.Lib.LCG              : INFO   Job 1 Scheduled at mu6.matrix.sara.nl:2119/jobmanager-pbs-atlas - Fri Sep 9 12:20:50 2005
Ganga.Lib.LCG              : INFO   Job 1 Running at mu6.matrix.sara.nl:2119/jobmanager-pbs-atlas - Fri Sep 9 12:22:00 2005
Ganga.Lib.LCG              : INFO   Job 1 Done (Success) at mu6.matrix.sara.nl:2119/jobmanager-pbs-atlas - Fri Sep 9 12:24:26 2005

In [6]: print jobs
Statistics: 1 jobs
-----
ID      status  name                                     mu6.matrix.sara.nl:2119/jobmanager-pbs-atlas
-----
1      completed HelloFromRemoteHost

In [7]: print file( j.outputdir + "stdout" ).read()
Hello from mu12.matrix.sara.nl - Fri Sep 9 13:20:27 CEST 2005

In [8]:

```



High-Energy Physics applications

ATLAS and LHCb will investigate various aspects of particle production and decay in high-energy proton-proton interactions at the Large Hadron Collider (LHC), due to start operation at the European Laboratory for Particle Physics (CERN), Geneva, in 2007. Both experiments will require processing of data volumes of the order of petabytes per year, and will rely on computing resources distributed across multiple locations. The experiments' data-processing applications, from event modelling to physics analysis, are based on the Gaudi/Athena C++ framework. Ganga (Gaudi/Athena and Grid Alliance), dramatically simplifies configuring these applications, running them on the Grid, and keeping track of results.