

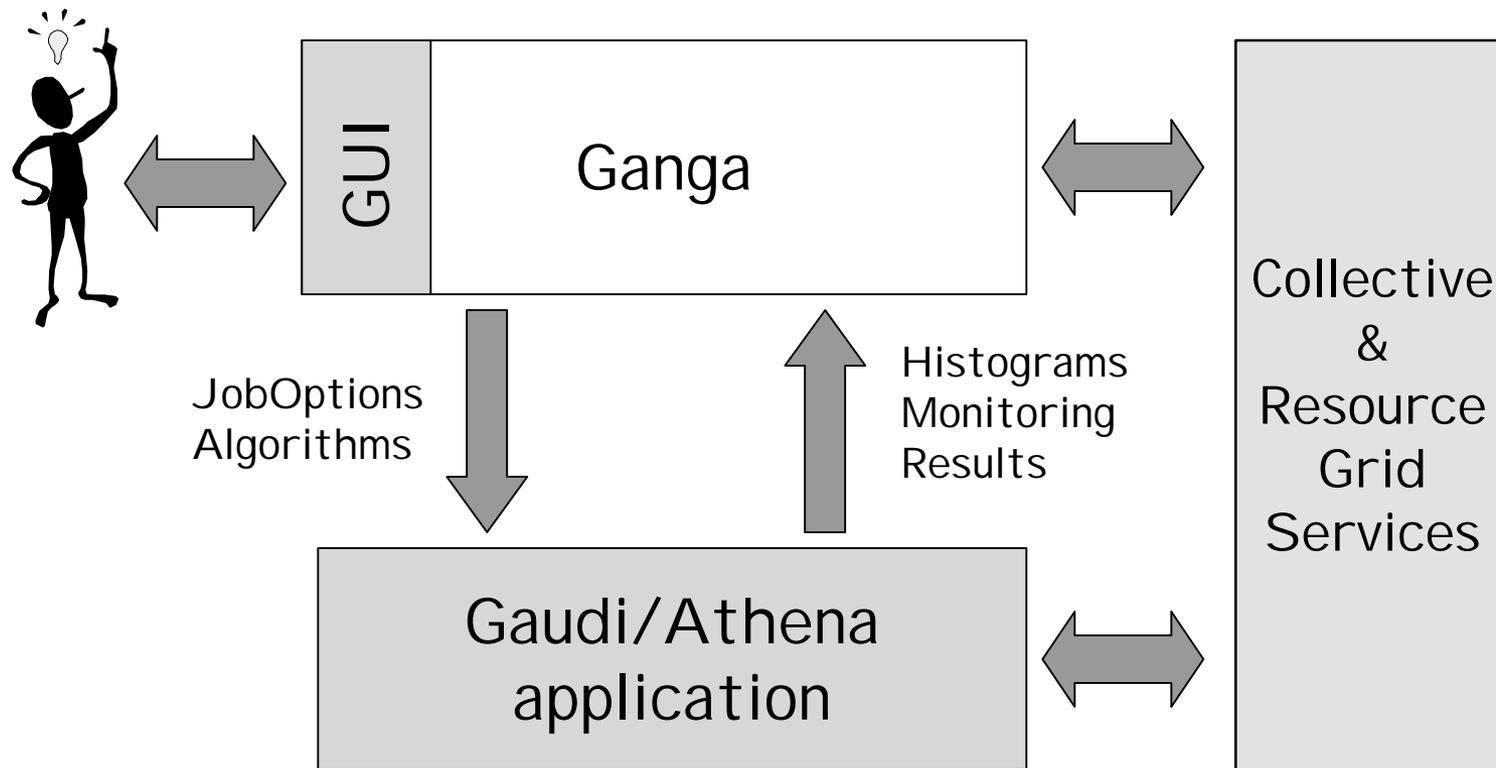
THE GANGA PROJECT

- Project objectives and organisation
 - Ganga design
 - Current status of software
 - Conclusions
-

Main objectives

- Aim to develop a tool to help physicists in ATLAS and LHCb carry out their studies in the computing environment of LHC: distributed datasets in the petabyte range; distributed resources; complex, highly configurable software
- Help configure applications developed within the Gaudi/Athena framework shared by ATLAS and LHCb
- Deal with submitting and monitoring jobs to distributed (Grid) and local batch systems
- Help users to keep track of what they've done
- Hide all technicalities: allow user to concentrate on the physics
- Provide a single user-friendly desktop environment for all tasks
- ▶ Ganga: Gaudi/Athena and Grid Alliance

Ganga: Gaudi/Athena and Grid Alliance



Project organisation

Ganga is being developed as an ATLAS/LHCb common project, with support in UK from GridPP

Current main contributors are:

- ▶ Developers: K.Harrison, W.Lavrijsen, A.Soroko, C.L.Tan
- ▶ Technical direction: P.Mato, C.E.Tull
- ▶ GridPP coordination: N.Brook, R.W.L.Jones

Ganga-related information regularly updated on web site:
<http://ganga.web.cern.ch/ganga>

A mailing list has been set up: project-ganga@cern.ch

Usually have telephone meeting at least once every two weeks

- ▶ Details of times placed on web site and circulated to mailing list

Detailed objectives (1)

Short-term plans: to end of September 2003

Longer-term plans: October 2003 onwards

- 1) Simplify users' lives by providing a single interface for working with all Athena-based (offline) applications
 - Short term: graphical user interface (GUI) and command-line interface (CLI) for working with analysis jobs; incorporate features from Athena Startup Kit, ASK (W.Lavrijsen)
 - Longer term: allow for running of production-type jobs, integrating with existing production tools: of ATLAS and LHCb: AtCom (L.Goossens et al.) and DIRAC (A.Tsaregorodtsev et al)

Detailed objectives (2)

- 2) Make workflows/data transformations easy to define, store and instantiate, and supply templates for common use cases
 - Short term: Ganga will provide a simple workflow-definition mechanism of its own
 - Longer term: Adopt more sophisticated workflow-definition procedure, for example based on Chimera (R. Gardner et al.)

- 3) Help with configuration by providing a job-options editor
 - Short term: allow access to, and modification of, all job options, with possibilities for choosing options for a particular algorithm or user favourites
 - Longer term: give guidance on meaningful values (with input from algorithm developers)

Detailed objectives (3)

Provide simple, flexible procedure for splitting and cloning jobs

- Short term: introduce splitting/cloning procedure, deal with common use cases, take care of merging of outputs where appropriate/possible
- Longer term: dependent on user feedback

Help users keep track of what they've done

- Short term: provide catalogue of jobs and their status, and allow access to settings for each
- Longer term: dependent on user feedback

Perform job monitoring tasks on local and distributed systems

- Short term: pull information from jobs, allowing automatic updates of status and user-initiated queries
- Longer term: move to system where jobs push information to a user-specified location; integrate with NetLogger for Grid

Detailed objectives (4)

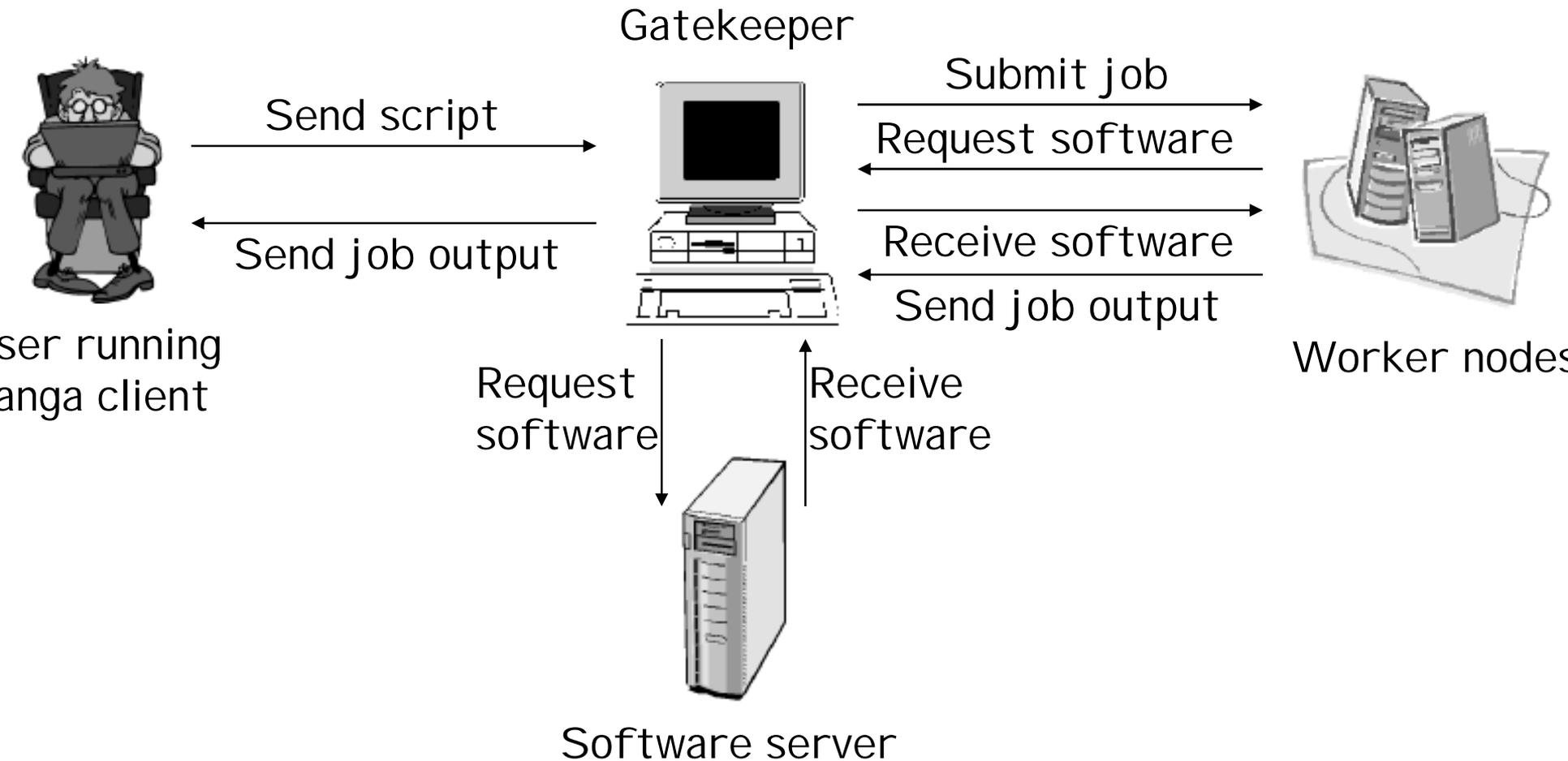
1) Allow for user mobility

- Short term: provide a single procedure for submitting jobs to different types of batch systems (EDG, LSF, PBS, Condor, other), with the batch system accessible from the machine where Ganga is run
- Longer term: allow user to submit jobs from any machine with Ganga running, to batch queues on any machine (Gatekeeper) where user has an account or is in the Grid mapfile; take care of software installation at remote nodes

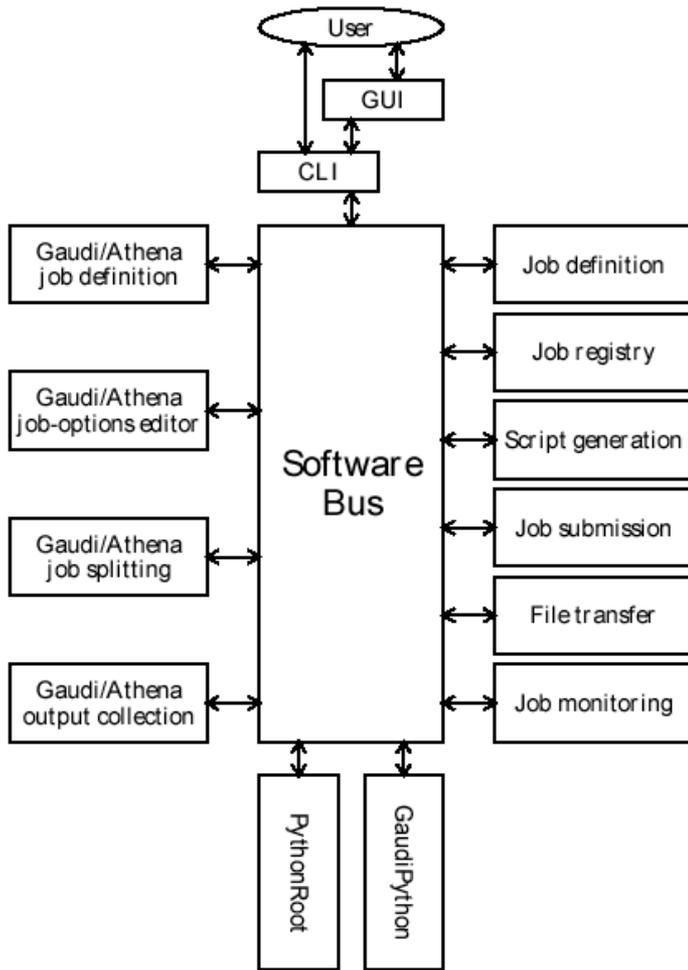
2) Other things, to be determined by user requests, but should consider possibilities for:

- web-portal interface
- interactive analysis (based on ROOT)
- data-management services

Pre-execution of release installation



Design



- User has access to functionality of GANGA components both through GUI , and through CLI , layered one over the other above a software bus
- Components used by GANGA can be divided into three categories:
 - ▶ Ganga components of general applicability (to right in diagram)
 - ▶ Ganga components providing specialised functionality (to left in diagram)
 - ▶ External components (at bottom in diagram)

Components of General Applicability (1)

- Components potentially have uses outside ATLAS and LHCb
 - ▶ Could be of interest for LCG PI project and other eScience applications
- Core component provides classes for job definition, where a job is characterised in terms of: name, workflow, required resources, status
 - ▶ Workflow is represented as a sequence of elements (executables, parameters, input/output files, etc) for which associated actions are implicitly defined
 - ▶ Required resources are specified using a generic syntax

Components of General Applicability (2)

- Other components perform operations on, for, or using job objects
 - ▶ Job-registry component allows for storage and recovery of job information, and allows for job objects to be serialized
 - ▶ Script-generation component translates a job's work flow into the set of instructions to be executed when the job is run
 - ▶ Job-submission component submits work flow script to target batch system, creating JDL (job-description language) file if necessary and translating resource requests as required
 - ▶ File-transfer component handles transfer between sites of input and output files, adding appropriate commands to work flow script at submission time
 - ▶ Job-monitoring component performs queries of job status

Specialised components for ATLAS and LHC

- Components incorporate knowledge of the Gaudi/Athena framework
 - ▶ Component for Gaudi job definition adds classes for workflow elements not dealt with by general-purpose job-definition component, for example applications packaged using CMT; component also provides workflow templates covering common tasks
 - ▶ Other components provide for job-option editing, job splitting, and output collection

External Components

- Additional functionality obtained using components developed outside of Ganga:
 - ▶ Modules of python standard library
 - ▶ Non-python components for which appropriate interface has been written, for example Gaudi framework itself (GaudiPython) and ROOT (PyROOT)

Current status

Current release of Ganga (Ganga-01-03-00) can be obtained from Gaudi CVS repository, and includes:

- ▶ GUI
- ▶ Command-line access to underlying tools (but not user oriented)
- ▶ Job-options editor (so far set up only for AtIfast)
- ▶ Submission of various types of jobs to different batch systems (LSF, PBS, EDG)
- ▶ Mechanism for splitting/cloning jobs
- ▶ Job catalogue
- ▶ Automatic monitoring

Also have, not yet added to release:

- ▶ Prototype software bus, to improve component management
- ▶ Python bindings for Magda and AMI

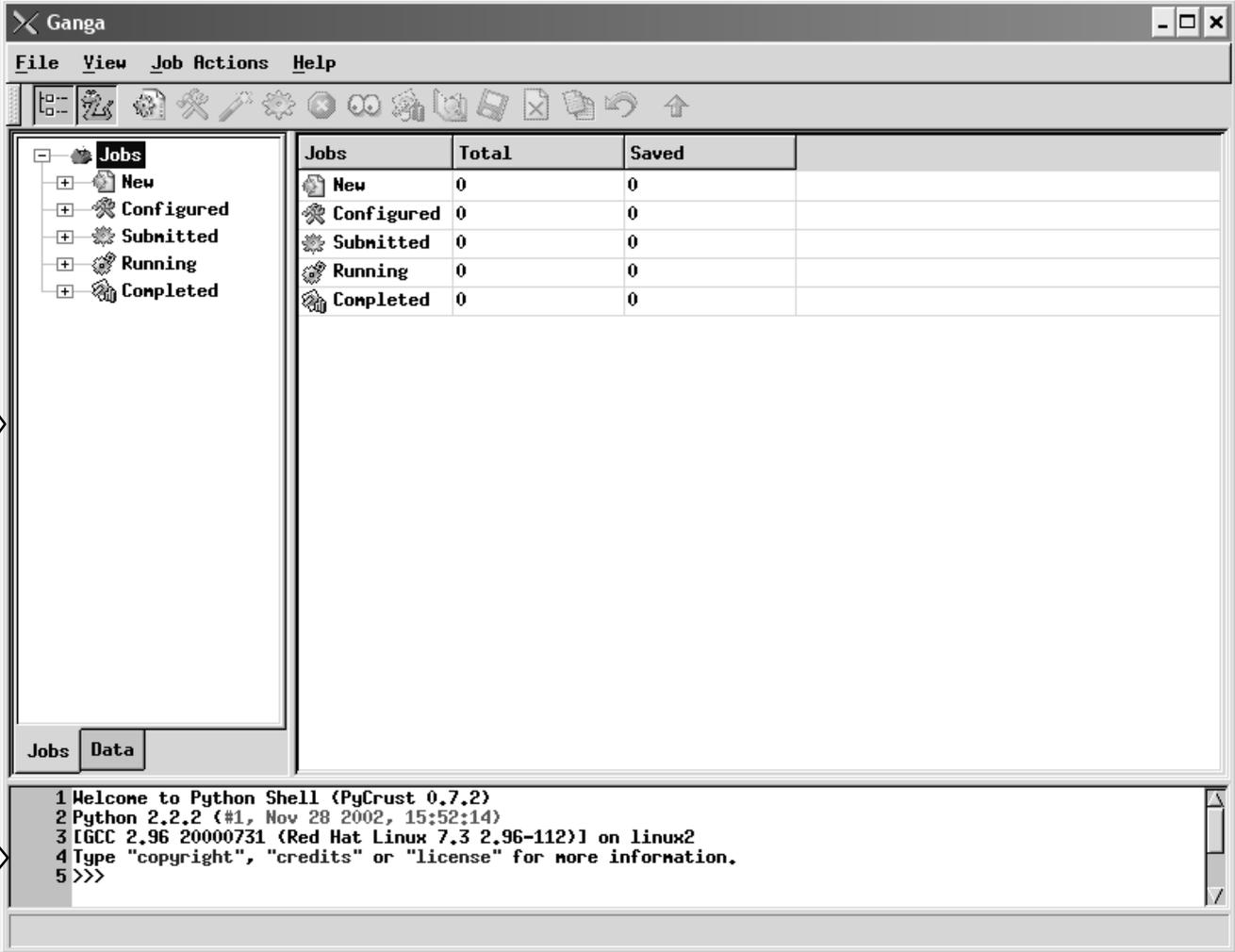
First uses of Ganga

Ganga can already be used to run ATLAS simulation and reconstruction (both treated as generic applications) and Atlfast (with customisations that help with configuration)

Ganga session included in BNL ATLAS software tutorial (27th August 2003), attended by about 50 physicists

- ▶ Clear that Ganga needs improvement, but everyone able to launch GUI and run some jobs
- ▶ This is the first large group of people to try out Ganga
- ▶ Thanks to S.Rajagopalan for scheduling session, and to tutorial participants who (perhaps unknowingly) acted as guinea pigs

Basic GUI



Toolbar

Job tree

Main panel

Python interpreter

Job creation

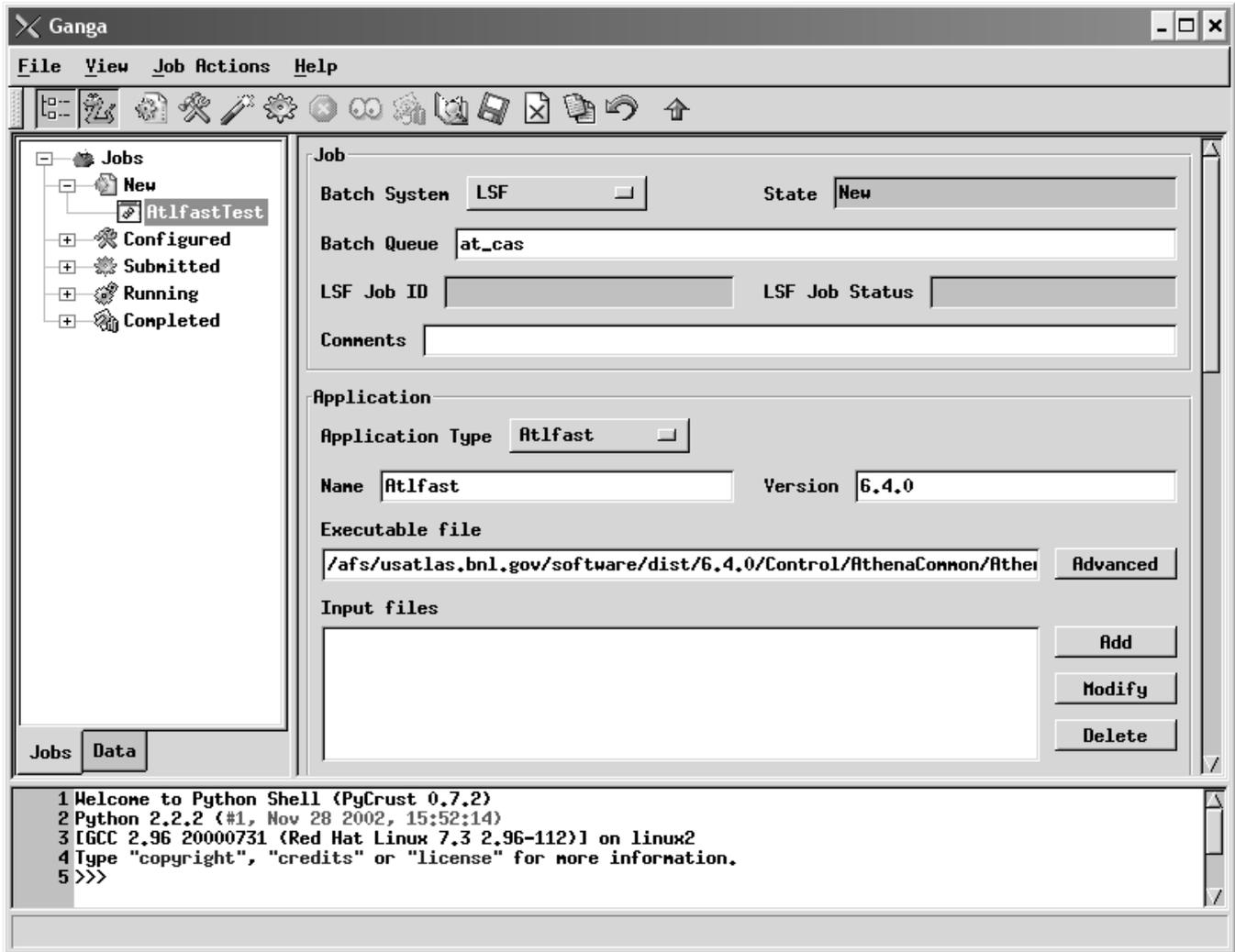
The screenshot displays the Ganga GUI interface. On the left, a sidebar shows job status categories: New, Configured, Submitted, Running, and Completed. The main area features a table with columns for 'Jobs', 'Total', and 'Saved', showing zero counts for all categories. A 'New Job' dialog box is open, containing the following fields:

- Job name: At1fastTest
- Batch system: LSF
- Type of application: At1fast

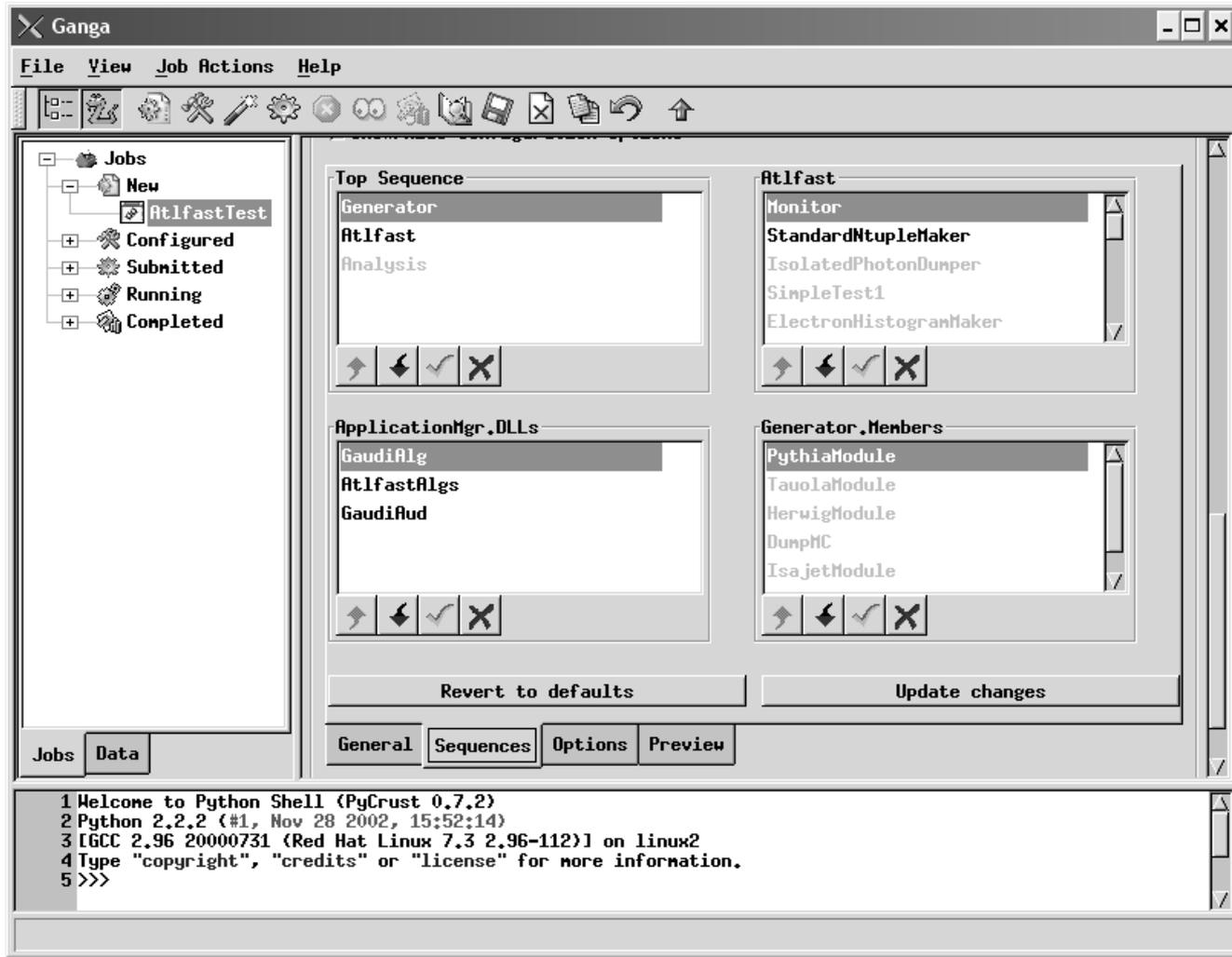
Buttons for 'OK' and 'Cancel' are visible at the bottom of the dialog. Below the table, a terminal window shows the following output:

```
1 Welcome to Python Shell (PyCrust 0.7.2)
2 Python 2.2.2 (#1, Nov 28 2002, 15:52:14)
3 [GCC 2.96 20000731 (Red Hat Linux 7.3 2.96-112)] on linux2
4 Type "copyright", "credits" or "license" for more information.
5 >>>
```

Job-parameters panel



Job-options editor: sequences



Job-options editor: options

	Option Name	Option Value
1	ApplicationMgr.EvtMax	10
2	ApplicationMgr.ExtSvc	['AtRndnGenSvc']
3	ApplicationMgr.HistogramPersistency	HBOOK
4	AtRndnGenSvc.Seeds	[]
5	AtlfastB.AtlfBNSet	1
6	AtlfastB.AtlfBjetSwitch	true
7	AtlfastB.AtlfCalSwitch	true
8	AtlfastB.AtlfTauSwitch	true
9	AtlfastB.AtlfTauVetoSwitch	false
10	AtlfastB.AtlfTrigMuSwitch	false
11	AtlfastB.CJetCorrFile	./AtlfastBcjet.dat
12	AtlfastB.JetCorrFile	./AtlfastBjet.dat
13	AtlfastB.OutputLevel	5
14	AtlfastB.TauEff	0.05
15	AtlfastB.TauVetoOption	1
16	AtlfastProtoJetMaker.ClusterPJConstruct	true

```
1 Welcome to Python Shell (PyCrust 0.7.2)
2 Python 2.2.2 (#1, Nov 28 2002, 15:52:14)
3 [GCC 2.96 20000731 (Red Hat Linux 7.3 2.96-112)] on linux2
4 Type "copyright", "credits" or "license" for more information.
5 >>>
```

Job submission

The screenshot shows the Ganga GUI with the following components:

- Menu Bar:** File, View, Job Actions, Help
- Toolbar:** Contains various icons for job management.
- Tree View (Left):** Shows a hierarchy of jobs: Jobs (expanded) -> New, Configured, Submitted, Running (selected), Completed. Under 'Running', 'AtlfastTest' is listed.
- Main Table:** A table with columns: Job Name, Job State, Batch System, Application Name. It contains one row: AtlfastTest, Running, LSF, Atlfast.
- Log Window (Bottom):** Shows the following text:

```
6 The _AUTO_INSTALL variable is set to TRUE.  
7 Job <22437> is submitted to queue <at_cas>.  
8 >>>  
9 JOBID  USER  STAT  QUEUE  FROM_HOST  EXEC_HOST  JOB_NAME  SUBMIT_TIME  
10 22437  karl  RUN   at_cas  acas038   acas060   Job200382823254701024  Aug 28 19:02  
11 >>>
```

Job submission

Jobs

- New
- Configured
- Submitted
- Running**
 - AtlfastTest
- Completed

Job Name	Job State	Batch System	Application Name
AtlfastTest	Running	LSF	Atlfast

```
6 The _AUTO_INSTALL variable is set to TRUE.
7 Job <22437> is submitted to queue <at_cas>.
8 >>>
9 JOBID USER STAT QUEUE FROM_HOST EXEC_HOST JOB_NAME SUBMIT_TIME
10 22437 karl RUN at_cas acas038 acas060 Job200382823254701024 Aug 28 19:02
11 >>>
```

Examination of job output

The screenshot shows the Ganga GUI interface. The main window displays a table of jobs:

Job Name	Job State	Batch System	Application Name
AtlfastTest	Completed	LSF	Atlfast

A dialog box titled "Choose a file to view" is open, showing the current directory: `/usatlas/u/karl/myGangaJobs/Job200382823254701024/output`. The dialog displays a file listing:

Name	Size	Date	Time	Permissions
..	<DIR>	28.08.2003	19:03	rwx
atlfast.hbook	12288	28.08.2003	19:03	rw-
atlfast.ntup	262144	28.08.2003	19:03	rw-
std.err	11616	28.08.2003	19:03	rw-
std.out	87116	28.08.2003	19:03	rw-

The main window also shows a terminal output at the bottom:

```
6 The _AUTO_INSTALL variable is set to TRUE.
7 Job <22437> is submitted to queue <at_cas>.
8 >>>
9 JOBID  USER  STAT  QUEUE  FROM_HOST  EXEC_HOST  JOB_NAME  SUBMIT_TIME
10 22437  karl  RUN   at_cas  acas038   acas060   Job200382823254701024  Aug 28 19:02
11 >>>
```

Conclusions

- Much work has been done on Ganga since project launch in May 2002
- Ganga source code is in Gaudi CVS repository
- Current version of Ganga (Ganga-01-03-00) is not production quality, but is useful for giving a feel of how things will work
- Ganga team has a well-defined plan of additions and improvements, but welcome user feedback on what is already implemented, and on priorities for the things that are missing
- Need to evaluate draft report from LCG RTAG 11:
Architectural Roadmap towards Distributed Analysis (ARDA)
 - ▶ understand where Ganga can contribute