Grid Computing
From a User's Point of View

A Thousand CPUs at Your Command

Adrian Vogel
DESY FLC

What Is the Grid?

World-wide network of computing resources

- motivated by LHC's computing needs:
  LHC Computing Grid (LCG)
- now shared with other HEP projects
  (HERA, ILC, IceCube, theory, . . .)

The Grid takes the idea of the WWW to the next level

- WWW: world-wide information out of the socket
- Grid: world-wide computing out of the socket

Current examples for usage of the Grid:
Monte Carlo production for CMS, ATLAS, ZEUS, H1

Grid Resources

Abstract concepts for hardware facilities

- Computing Elements (CEs): computer clusters
- Storage Elements (SEs): data mass storage

Actual implementation is transparent to the user

- all CEs and SEs should behave the same

Various services manage information on different levels

- the picture can become arbitrarily complicated (BDII, GIIS, GRAM, LRC, PBS, RB, RMC, SRM, . . .)
- but you don’t have to worry as an end user

The Grid Middleware

Set of commands to interact with Grid resources

- turns a computer into a Grid User Interface (UI)

Probably preinstalled somewhere at your site

- supposed to run under Scientific Linux 3
- source a shell script to set up your environment (`PATH`, `MANPATH`, and some other variables)
- ask your local computing support for details!

Comes in various flavours, gLite 3.0 is the latest release

- bound to funding projects (e. g. EGEE)
- newer packages still contain legacy commands

Certificates and VOs

Grid certificates

- authentication (proves who you are)
- one certificate per human being
- X.509-style with public and private key (RSA)
- signed by your regional Grid Certification Authority
- used to create “proxy certificates” as necessary (unprotected working copies with limited lifetime)

Virtual Organisations (VOs)

- authorisation (grants access to resources)
- world-wide “user groups” in the Grid
- everybody should be member of (at least) one VO

Getting a Certificate

Creating a certificate request

- log in to any Grid UI
- `grid-cert-request` creates a key pair

Your local Registration Authority (RA) handles requests

- send your request file via e-mail
- hand in a signed paper request form
- attach a copy of your passport / ID card

Your Certification Authority (CA) issues the certificate

- allow a few hours for processing
- you will receive the signed certificate via e-mail

Becoming Member of a VO

VO Membership Service (VOMS) knows all members

- web-based user interface for management

Submit a user registration request

- import the Grid certificate into your browser (use `openssl` to create a PKCS#12 file)
- choose your VO (probably `ilc` or `calice`)
- go to the VOMS web page
- request the VO membership

An administrator of the VO will accept or decline

Getting Started (e. g. at DESY)

Log in to an SL3 machine, get a UI

- `ssh slref.desy.de` (for example)
- `source /afs/desy.de/group/it/grid/\`
  `UI/GLITE/etc/profile.d/grid_env.sh`

Create a proxy (default lifetime 12 hours)

- `glite-voms-proxy-init --voms ilc`
- `glite-voms-proxy-info --all`
- `glite-voms-proxy-destroy` (after you're finished)

You should now have access to all Grid resources which are available to the VO `ilc`

Computing Elements

CEs resemble traditional batch clusters

- one or more job queues
  (sometimes split up depending on execution time)
- a cluster of invisible worker nodes (WNs)
  executes the submitted jobs
- a certain amount of CPUs is available to each VO

More information on available CEs

- `lcg-infosites --vo ilc ce`
- shows also the current queue status

Job Description Language

Job Description Language (JDL) files
tell the Computing Element what to do

- VO membership
- the executable file to run
- additional command line arguments
- input / output files (“sandboxes”, limited in size)
- stdin / stdout / stderr redirection
- required properties of the CE (via GLUE expressions)
- additional job attributes (e. g. number of retries)

Human-readable files in plain text with JDL syntax

JDL – An Example

```
VirtualOrganisation = “ilc”;
Executable    = “test.sh”;
Arguments     = “foo 5”;
InputSandbox  = { “test.sh” };
OutputSandbox = { “std.out”, “std.err” };
StdInput      = “”;
StdOutput     = “std.out”;
StdError      = “std.err”;
Requirements  = RegExp(“desy.de”,
                  other.GlueCEUniqueId);
RetryCount    = 2;
```

Job Submission

Check the JDL file (and list suitable CEs)

- `glite-job-list-match test.jdl`

Submit the job, store the job ID

- `glite-job-submit -o test.jid test.jdl`

Check the job status (e. g. “scheduled”, “running”, “done”)

- `glite-job-status -i test.jid`

Retrieve the job output (→ status “cleared”) – only once!

- `glite-job-output -i test.jid --dir .`

Cancel a job (→ status “cancelled”)

- `glite-job-cancel -i test.jid`

Storage Elements

SEs are mass storage devices

- simple disks
- RAID disk arrays
- huge tape robots
- dCache systems

More information on available SEs

- `lcg-infosites --vo ilc se`
- shows also the current usage of space

The File Catalogue

The LCG File Catalogue (LFC) maps logical filenames (LFNs) to actual files residing on some Storage Element

- one global LFC host for each VO
- hierarchical directory structure
- file ownership, access control

Logical filenames are automatically resolved

- possibility to use human-readable, structured names
- one file can have multiple replicas on different SEs (to improve safety and efficiency)

Every file also has a unique low-level identifier (GUID)

Data Management – Examples

Examine catalogue contents

- `lfc-ls -l /grid/ilc/flc/vogel`

Create a new subdirectory

- `lfc-mkdir /grid/ilc/flc/vogel/test`

Upload a local file (copy and register)

- `lcg-cr -d <SE> -l lfn:<LFN> file:<FILE>`

Download a file from an SE

- `lcg-cp lfn:<LFN> file:<FILE>`

Delete a file (on all SEs and in the LFC)

- `lcg-del -a lfn:<LFN>`

CEs – An Unfriendly Place

Don’t expect too much out there!

- a running operating system (SL3)
- the Grid Middleware
- a temporary working directory only for you
- one executable which will be run

Everything else has to be supplied by you

- some shell script (typically) takes control
- ship large executables, shared libraries you need, and data files via an SE to the worker node
- you may want to use static linking (or `ldd`)

Some Tips for Grid Jobs

Always produce plenty of diagnostic output

- `date`, `hostname`, `uname`, `whoami`, `env`, `...`
- use verbose mode of commands whenever possible
- this will be the only way to solve problems!

Don’t rely on SEs

- SEs can be temporarily unavaliable
- try different SEs, create replicas of files
- wait – services may come back after some time
- failures at the end of the job are the most annoying!

Remember that you may not be alone

Problems With the Grid

Grid Middleware is not perfect yet

- still under development, sometimes major changes
- sometimes clumsy usage, not intuitive, not consistent
- usefulness of documentation may vary

Grid sites do not always run smoothly

- jobs are aborted for no obvious reason
- little possibility of intervention by the user
- large VOs (LHC experiments) get massive support
- but ILC is only a “parasitic” user – up to now

You’ll have to get used to something new!

Why Use the Grid, Then?

Most problems can be solved

- ask somebody who has done it before (mailing lists)
- contact the administrators (friendly human beings)
- every user is still a tester, too – feedback helps!

Grid computing is the future

- local computer clusters will die out
- future money will go into Grid resources
- LHC needs the Grid, and so will we
- demonstrate the need for computing power!

Remember when e-mail and WWW were something new?

Further Information

Read The Fine Manual (LCG-2 User Guide)

- Google knows where to find it
- contains a lot of technical mumbo-jumbo, but:
- chapters 5 and 6 get straight to the point

Report problems to the Global Grid User Support

- `www.ggus.org` (needs a certificate)

Browse the Grid Operations Centre Wiki

- `goc.grid.sinica.edu.tw/gocwiki`

Ask your local Grid administrators for help!

Conclusion and Summary

Use the Grid. Now.