NorduGrid Tutorial

First steps

Aleksandr Konstantinov
Vilnius University/Lithuania
University of Oslo/Norway

University of Iceland, Reykjavik, November 17, 2004

Install software

Following information is available in "The NorduGrid user guide" at http://www.nordugrid.org/papers.html#man-user

- NorduGrid provides "standalone client" which contains all necessary software, public keys and configuration files to start User Interface. Procedure to install:
  - Download from http://ftp.nordugrid.org/download/index.php suitable package.
  - Unpack tarball:

tar -zxvf nordugrid-standalone-<version>.<architecture>.tgz

  - Make shell process execute setup script

cd nordugrid-standalone-0.4.4

source setup.sh

  - Now environment is ready for basic ARC and Globus commands

Become recognised

Start from obtaining recognised certificate

- Most sites which are members of NorduGrid infrastructure recognise CAs accepted by the European Policy Management Authority for Grid Authentication in e-Science http://marianne.in2p3.fr/datagrid/ca/ca-table-ca.html
- Always use Your national or organizational CA.
- After choosing Your CA follow procedure:
  - Run grid-cert-request -int -ca
  - Read and follow instructions on screen carefully.
- After You received certificate store it at proper place
- Now You will be recognised at remote sites
- Run grid-cert-info -subject -enddate
  - Subject of certificate is Your name on the Grid
  - Enddate tells You when to ask for new certificate

Become welcomer

- To be allowed user must join VO
  - Normally that shouldn't be a problem unless You are first Grid user in a neighborhood
  - NorduGrid Guest VO is good start for those who want to try
- Send letter to nordugrid-support@nordugrid.org
  - Provide information about Your affilation, subcject of the certificate and anything to persuade VO maintainer that You need Grid (not a problem at all)

Available commands

- User Interface consists of command line utilities

ngsub	submit a task
ngstat	obtain the status of jobs and clusters
ngcat	display the stdout or stderr of a running job
ngget	retrieve the result from a finished job
ngkill	cancel a job request
ngclean	delete a job from a remote cluster
ngrenew	renew user's proxy
ngsync	synchronize the local job info with the MDS
ngcopy	transfer files to, from and between clusters
ngremove	remove files

- Before doing anything "login into Grid"

  grid-proxy-init

  once per 12 hours. Do not forget clock of You computer must be set properly.

How to write description of a job

- For complete information read "*XRSL (Extended Resource Specification Language)*" at http://www.nordugrid.org/papers.html
- In short, RSL consists of attributes
  - (name=values)
  - glued together with *logical and*
- Example of simple *Hello World* job

```
&
(executable=/bin/echo)                  main executable
(arguments="Hello World")               arguments for main executable
(cputime=1m)                            request 1 minute of CPU time
(stdout=out.txt)                        store output into out.txt
(jobname=hello)                         name OF a job for convenience
```

Run Your first job

- **Try to submit simplest *Hello World* job**

  $ ngsub '&(executable=/bin/echo)(arguments="Hello World")(cputime=1m)
  (stdout=out.txt)(jobname=hello)'

  > Job submitted with jobid gsiftp://fire.ii.uib.no:2812/jobs/
  39311005309408777145 81

- **Check for status of the job time to time till it turns into finished**

  $ ngstat hello

  > Job gsiftp://fire.ii.uib.no:2812/jobs/
  39311005309408777145 81

  Jobname: hello

  Status: FINISHED 2004-11-15 16:20:01

- **Retrive produced results**

  $ ngget hello

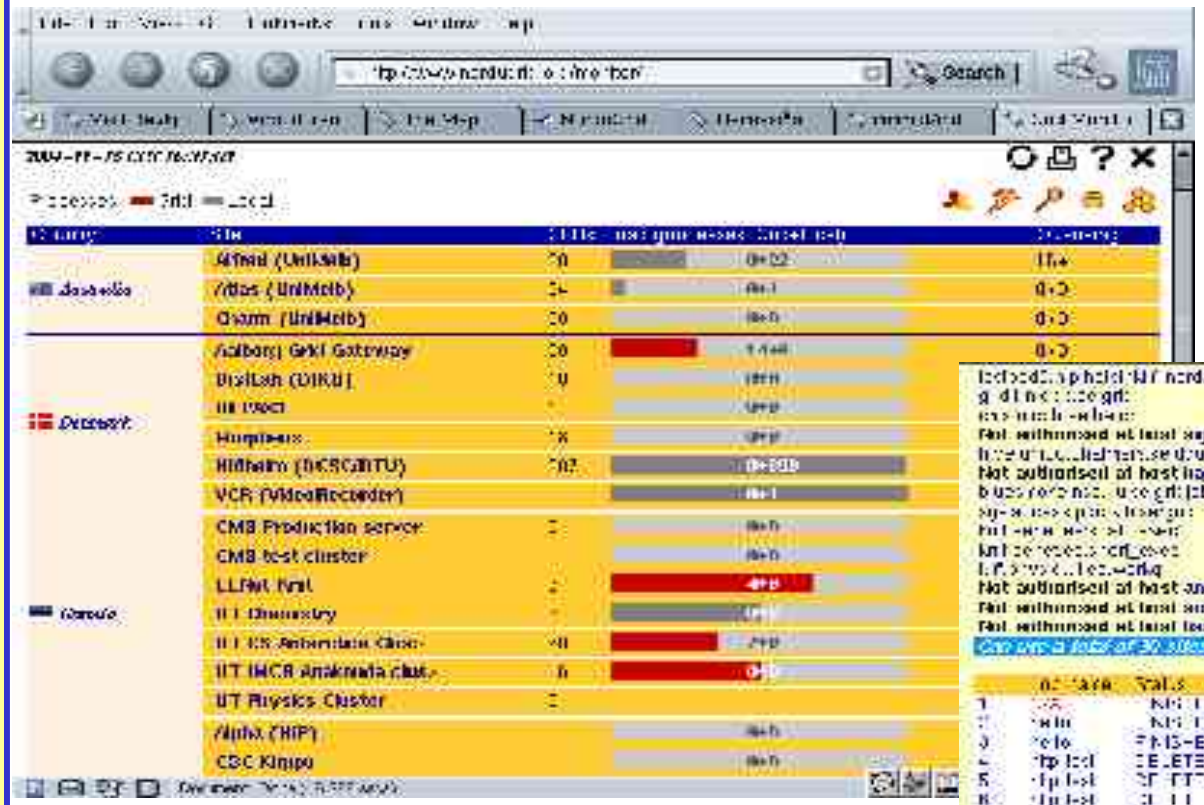
  > ngget: downloading files to /home/SanjaK/39311005309408777145 81
  ngget: download successful - deleting job from gatekeeper.

  $ cat 39311005309408777145 81/out.txt

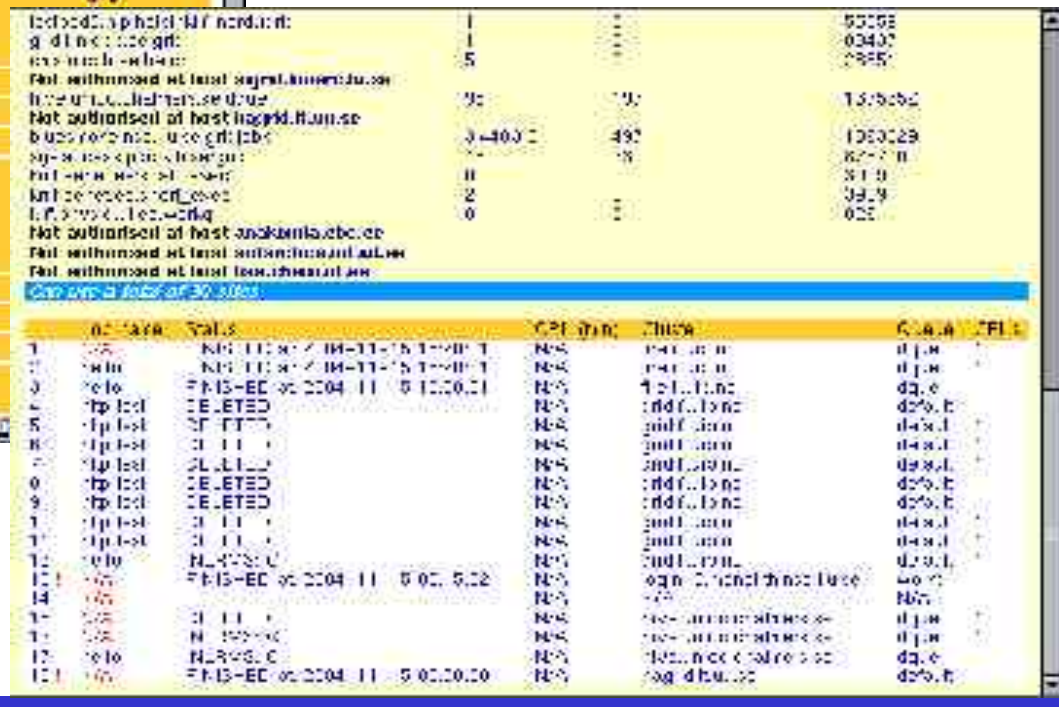
  > Hello World

Monitoring of job

- **ngstat**
- **Grid Monitor -** web based job monitoring tool. It's Your best friend in a netcafe.



- Summary per cluster
- Jobs per cluster
- Jobs per user
- etc.



Sophisticated job

- **Let's run job with input and output data, Runtime Environements, etc.**
  - Wrapper script - runpov.sh

```
#! /bin/sh
povray +H600 +W800 +omonolith.png monolith.pov
```

  - RSL - povray.rsl

```
&(executable=runpov.sh)
(runtimeenvironment=POVRAY)
(inputfiles=
 ("monolith.pov"
    "http://www.nordugrid.org/tutorial/dapsys_tutorial/povraydemo/monolith.pov")
 ("front.png" "http://www.nordugrid.org/tutorial/dapsys_tutorial/povraydemo/front.png"))
 (outputfiles=("monolith.png" ""))
(stdout=out.txt)(stderr=err.txt)
(gmlog=logs)
(cputime=1h)
(jobname=povray)
```

Sophisticated job (continued)

- Job uses
  - Preinstalled software package POW-Ray
  - Input files monolith.pov and front.png from external HTTP server www.nordugrid.org.
  - Executable file from user's computer runpov.sh
- Job produces
  - Printed output out.txt
  - Errors err.txt
  - Diagnostics information in subdirectory log
  - Rendered picture monolith.png

This example was derived from one by Leif Nixon which also shows how to split task into multiple jobs by rendering image in slices.
http://www.nsc.liu.se/~nixon/ng-povray/