CENTRO DE SUPERCOMPUTACIÓN DE GALICIA

CESGA Experience with the Grid Engine batch system

E. Freire, J. López, C. Fernández, A. Simon, P. Rey, R. Diez, S. Diaz

HEPiX Spring 2010 Workshop - Lisbon, Portugal, 21/04/2010

CESGA
XUNTA DE GALICIA
CONSELLERÍA DE INNOVACIÓN, E INDUSTRIA
CSIC
CONSEJO SUPERIOR DE INVESTIGACIONES CIENTÍFICAS
MINISTERIO DE CIENCIA E INNOVACIÓN
FEDER
FONDO EUROPEO DE DESENVOLVEMENTO REXIONAL

OUTLINE

1. Introducing GE
2. Grid Engine experience in a large cluster
3. Utilities
4. Grid Engine and gLite Middleware

CESGA Experience with the Grid Engine batch system

INTRODUCING GE

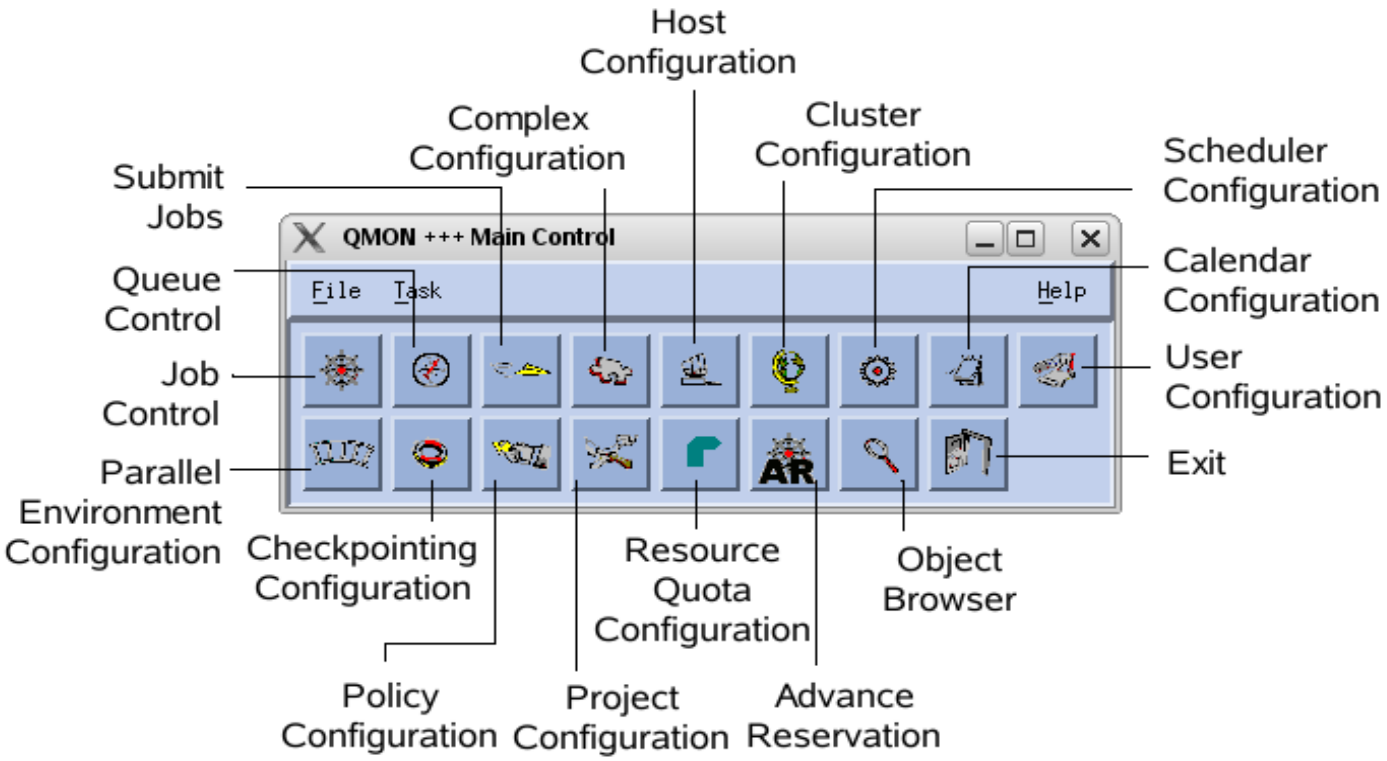
CESGA
XUNTA DE GALICIA
CONSELLERÍA DE INNOVACIÓN, E INDUSTRIA
CSIC CONSEJO SUPERIOR DE INVESTIGACIONES CIENTÍFICAS
MINISTERIO DE CIENCIA E INNOVACIÓN
FEDER
FONDO EUROPEO DE DESENVOLVEMENTO REXIONAL

Introducing GE

- Grid Engine, an open source job management system developed by Sun
  - http://gridengine.sunsource.net/
- GE Features:
  - Flexible Scheduling Policies: Priority; Urgency; Ticket-based: Share-Based, Functional, Override
  - Supports Array Jobs
  - Complex Resource Atributes
  - Shadow Master Hosts (high availability)
  - Tight integration of parallel libraries
  - Supports Check-pointing and Migration
  - Supports DRMAA 1.0
  - Good Documentation: Administrator's Guide, User's Guide, mailing lists, wiki, blogs
  - Enterprise-grade scalability:10,000 nodes per one master

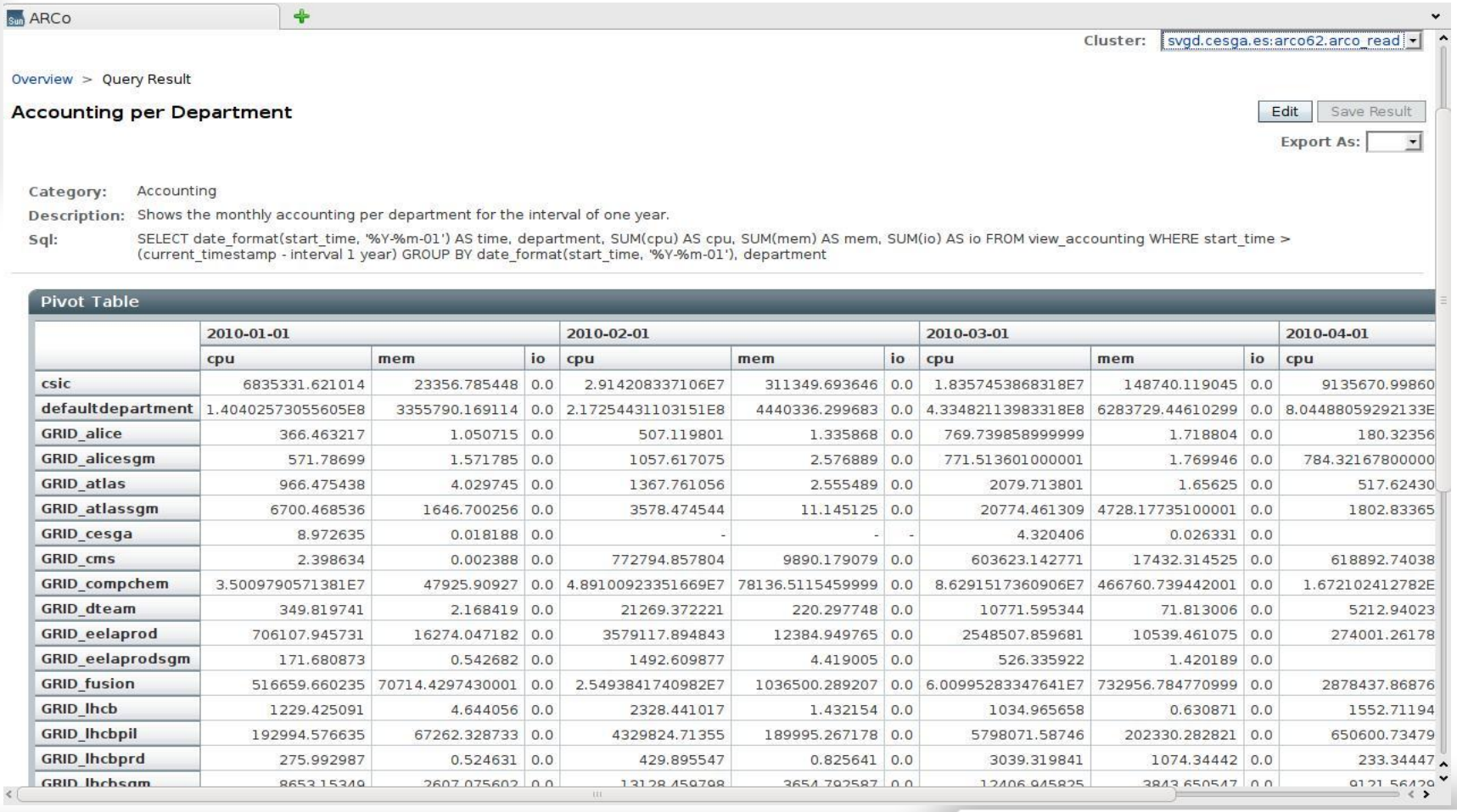
Some important GE Administration tools

- Qmon: A Graphical User Interface (GUI)



Some important GE Administration tools

Accounting and Reporting Console (ARCo)



	2010-01-01			2010-02-01			2010-03-01			2010-04-01
	cpu	mem	io	cpu	mem	io	cpu	mem	io	cpu
csic	6835331.621014	23356.785448	0.0	2.91420837106E7	311349.693646	0.0	1.8357453868318E7	148740.119045	0.0	9135670.99860
defaultdepartment	1.40402573055605E8	3355790.169114	0.0	2.17254431103151E8	4440336.299683	0.0	4.33482113983318E8	6283729.44610299	0.0	8.04488059292133E
GRID_alice	366.463217	1.050715	0.0	507.119801	1.335868	0.0	769.739858999999	1.718804	0.0	180.32356
GRID_alicesgm	571.78699	1.571785	0.0	1057.617075	2.576889	0.0	771.513601000001	1.769946	0.0	784.32167800000
GRID_atlas	966.475438	4.029745	0.0	1367.761056	2.555489	0.0	2079.713801	1.65625	0.0	517.62430
GRID_atlassgm	6700.468536	1646.700256	0.0	3578.474544	11.145125	0.0	20774.461309	4728.17735100001	0.0	1802.83365
GRID_cesga	8.972635	0.018188	0.0	-	-	-	4.320406	0.026331	0.0	
GRID_cms	2.398634	0.002388	0.0	772794.857804	9890.179079	0.0	603623.142771	17432.314525	0.0	618892.74038
GRID_compchem	3.5009790571381E7	47925.90927	0.0	4.89100923351669E7	78136.5115459999	0.0	8.6291517360906E7	466760.739442001	0.0	1.672102412782E
GRID_dteam	349.819741	2.168419	0.0	21269.372221	220.297748	0.0	10771.595344	71.813006	0.0	5212.94023
GRID_eelaprod	706107.945731	16274.047182	0.0	3579117.894843	12384.949765	0.0	2548507.859681	10539.461075	0.0	274001.26178
GRID_eelaprodsgm	171.680873	0.542682	0.0	1492.609877	4.419005	0.0	526.335922	1.420189	0.0	
GRID_fusion	516659.660235	70714.4297430001	0.0	2.5493841740982E7	1036500.289207	0.0	6.00995283347641E7	732956.784770999	0.0	2878437.86876
GRID_lhcb	1229.425091	4.644056	0.0	2328.441017	1.432154	0.0	1034.965658	0.630871	0.0	1552.71194
GRID_lhcbpil	192994.576635	67262.328733	0.0	4329824.71355	189995.267178	0.0	5798071.58746	202330.282821	0.0	650600.73479
GRID_lhcbprd	275.992987	0.524631	0.0	429.895547	0.825641	0.0	3039.319841	1074.34442	0.0	233.34447
GRID_lhcbsgm	8653.15349	2607.075602	0.0	13128.459798	3654.792587	0.0	12406.945825	3843.650547	0.0	9121.56429

CESGA Experience with the Grid Engine batch system

Grid Engine experience in a large cluster (Finis Terrae)

CESGA
XUNTA DE GALICIA
CONSELLERÍA DE INNOVACIÓN, E INDUSTRIA
CSIC
CONSEJO SUPERIOR DE INVESTIGACIONES CIENTÍFICAS
MINISTERIO DE CIENCIA E INNOVACIÓN
FEDER
FONDO EUROPEO DE DESENVOLVEMENTO REXIONAL

Grid Engine experience in a large cluster (Finis Terrae)

- The Finis Terrae cluster is composed by:
  - 142 Itanium nodes
    - 16 CPUs
    - 128 GB of Memory
  - 1 node with 128 cores and 1 Tb of memory.
  - 1 node with 128 cores and 384 Gb of memory.

- GE integration with MPI:
  - We decided to implement the tight implementation for HP-MPI because we want that all the task of MPI jobs are under full control of GE and also to obtain the full accounting for all the task.

- Tigh HP-MPI Integration notes can be found on link:
  - http://wiki.gridengine.info/wiki/index.php/Tight-HP-MPI-Integration-Notes

Tight HP-MPI Integration

- Set up the Grid Engine Parallel Environment (PE):
  - To add a new parallel environment:

```
qconf -ap <pe_name>
pe_name mpi
slots 9999
user_lists NONE
xuser_lists NONE
start_proc_args <YOUR_SGE_ROOT>/mpi/startmpi.sh \
                -catch_rsh  $pe_hostfile
stop_proc_args <YOUR_SGE_ROOT>/mpi/stopmpi.sh
allocation_rule $fill_up
control_slaves TRUE
job_is_first_task FALSE
urgency_slots min
accounting_summary FALSE
```

- Modify the starter_method and pe_list options of the queue you want to use:

```
starter_method <YOUR_PATH>/job_starter.sh
pe_list <pe_name>
```

- Set up the hp-mpi environment:

```
export MPI_REMSH=$TMPDIR/rsh
```

Tight Intel-MPI Integration: Extra steps

Apart from the necessary steps to implement a Tight integration we detected some extra actions to complete the integration of Intel MPI.

- Create independent rings by adding the variable to avoid processes interfering in the all-exit step:

```
export I_MPI_JOB_CONTEXT="$JOB_ID-$SGE_TASK_ID"
```

Note: Suppose that we have two different jobs running in the same node. If one of the job is killed or deleted, then both jobs will be killed or deleted except that we set up the I_MPI_JOB_CONTEXT variable in the global/user environment.

- Modify mpirun to use by default RSH instead of SSH:

```
other_mpdboot_opt="$other_mpdboot_opt --rsh=$TMPDIR/rsh"
```

Disabling direct SSH connection to the nodes

To accomplish this the following steps are required:

- MPI Tight Integration.
- Qlogin: configure it to run over qrsh following also the ***tight integration* way.**

  qconf -mconf
  [.....]
  qlogin_command builtin
  qlogin_daemon builtin
  rlogin_daemon builtin
  rsh_daemon builtin
  rsh_command builtin
  rlogin_command builtin
  [.....]
- Re-configuration of ssh to allow only certain administrative users to connect (option *AllowUsers*).

  -In this way all processes are correctly accounted in the accounting file.
- To facilitate interactive use of the nodes a wrapper was created.

- More info in:
  - http://wiki.gridengine.info/wiki/index.php/Disabling_\direct_ssh_connection_to_the_nodes

Checkpointing in GE

- Berkeley Lab Checkpoint/Restart (BLCR)
- Integrated with GridEngine (GE)
  - Configure a checkpoint environment:

```
qconf -sckpt BLCR
ckpt_name          BLCR
interface          application-level
ckpt_command $BLCR_PATH/bin/cr_checkpoint -f $ckpt_dir/\
             $ckptfile --stop -T $process2
migr_command $BLCR_PATH/bin/cr_restart $ckptfile
restart_command    none
clean_command      $BLCR_PAT/bin/clean.sh $job_id $ckpt_dir
ckpt_dir           $SGE_O_WORKDIR
signal             none
when               xsmr
```

Note: -f $ckptfile is where it saves the ckpt file. -stop stops the processes after doing the ckptointing and -T $process2 is the root process from the process tree, if instead of -T option, we use -t, it would only do chkpointing to the PID indicated.

  - Set up a ckpt_list in the queue configuration
- Checkpointing with gaussian.

CESGA Experience with the Grid Engine batch system

Utilities

CESGA
XUNTA DE GALICIA
CONSELLERÍA DE INNOVACIÓN, E INDUSTRIA
CSIC CONSEJO SUPERIOR DE INVESTIGACIONES CIENTÍFICAS
MINISTERIO DE CIENCIA E INNOVACIÓN
FEDER
FONDO EUROPEO DE DESENVOLVEMENTO REXIONAL

qsub wrapper

- Instead of use the original qsub, a wrapper let us check jobs fulfil some basic requirements before queued:

  - Check user has given an estimation of all the required resources (*num_proc*, *s_rt*, *s_vmem* and *h_fsize*) and that them fulfil the default limits.
  - Distribute jobs to the corresponding queues according the resources requested
  - Verify if the job can run in the queues before submission.

Let us to define environment variables necessary to some application (e.g. *OMP_NUM_THREADS.* The OMP_NUM_THREADS environment variable sets the default number of threads to use during execution).

- Support special resources requests (per user limit).

qsub wrapper: default limits

- Maximum number of processors: 192
  Maximum value of num_proc: 16
  Maximum number of slots:192

- **Execution time (s_rt):**
  Jobs without checkpointing: 300 hours
  Sequential jobs: Until 1000 hours
  Parallel jobs:

From 2 to 16 processors: 300 hours
From 17 to 32 processors: 200 hours
From 33 to 64 processors: 100 hours
From 65 to 128 processors: 50 hours
From 129 to 160 processors: 40 hours
From 160 to 192 processors: 20 hours
More than 192 processors: 10 hours

- **Memory:**
  Per core: 8GB
  Per node: 120 GB
- **H_fsize**: 700GB (scratch local)

qsub wrapper: queue distribution

Number of cores	Small queue	Medium queue	Large queue

queue
nodes

Management of special user requirements

- Default limits are not enough.
  - In this case, users can request access to extra resources (memory, CPU time, number of processors, space on scratch disk, …).

- The special resources requests are stored in a database and then exported to a XML file.

- The XML file is processed by the qsub wrapper.

```
<users>
  <user login="esfreire">
    <application id_sol="71">
      <start>2010-04-01</start>
      <end>2011-05-31</end>
      <priority>1</priority>
      <s_rt>1440</s_rt>
      <s_vmem_total>0</s_vmem_total>
      <h_fsize>0</h_fsize>
      <n_proc_total>256</n_proc_total>
      <num_proc>0</num_proc>
      <slots_mpi>0</slots_mpi>
      <s_vmem_core>0</s_vmem_core>
      <exclusivity>0</exclusivity>
    </application>
  </user>
......
<users>
```

Job prioritization

There are Some kind of jobs (special user requirements, challenges, agreements, …) that need to be prioritized.

When the jobs are submitted cannot be prioritized by the *qsub* wrapper so we have to do it after that they are enqueued:

- Change the number of override tickets (***qalter –ot***) and the priority (***qalter –p***) for the specified users.
- In very special cases a reservation (***qalter –R y***) is also done.
- Change the hard requested queues.

Job prioritization: special case

Jobs of the ***regional weather forecast service*** cannot wait for free slots as a normal job because the forecast for the weather has to be published on time. => So we have to move into execution its pending jobs as soon as possible.

- Process:
  - Check if there are jobs in error state and clears this state if it is necessary.
  - Hold all the pending jobs except its jobs.
  - Change the priority for this user.
  - Restrict access to the nodes while we are increasing complex_values like num_proc or memory to avoid other jobs entering in the selected nodes.
  - Restore the complex_values of the selected nodes.
  - Remove the hold state of the pending jobs

Should be prioritized some pending job?

- Daily email with information about pending jobs (resource request list, time since it was submitted, predecessor job, …).

- A job will be prioritized if:
  - Time since it was submitted is greater than the requested time.
  - Time since it was submitted is greater than a maximum waiting time limit (100 hours).

- The script can submit this information by email, and/or save it in a file in tab or CSV format. It can also save a summary by user and queue (historical information).

The script is based in the XML output of the *qstat* command (*qstat -u "*" -s p -r -xml*).

Are nodes overloaded?

- Nodes are shared between different jobs (big SMP nodes, specially superdome).
- Users not always use properly the number of slots required by the jobs. So we can have nodes **overloaded** or **underloaded**.
- For each node it is compared the load with the number of available processors and with the number of processors required by all the jobs running on it.
- The script is based in the XML output of the *qstat* command (*qstat -u "*" -s r -r -f -ne -xml -q *@NODE*) for each host (*qconf -sel*).

Application integration with GE: Gaussian

Gaussian is one of the most common application used incorrectly =>Created a wrapper to avoid it.
The wrapper let us:

- Control how it is used:
  - Gaussian is used only under the queue system.
  - Users don't try to use MPI environment
  - The queue system requirements are accordingly to Gaussian input
- Facilitate the use:
  - Not necessary to set all the requirements in the Gaussian input

When will be free the node X?

- Sometimes we need to know when a node will be free (maintenance, tests, …).

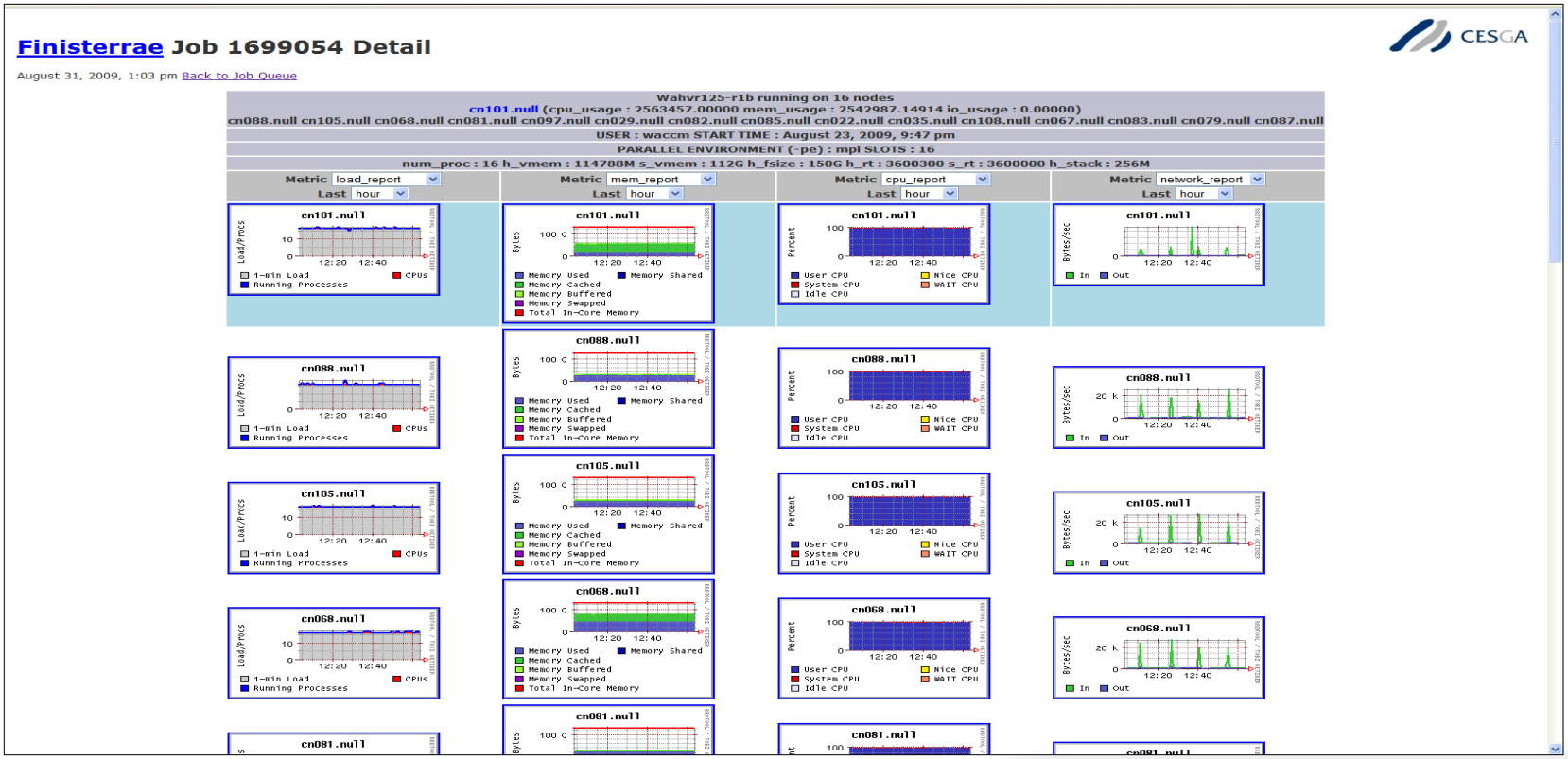
- Using the list of jobs running in the node and the required s_rt limit for each job we can obtain the expected end time of all the jobs.

```
JOBS RUNNING IN NODES: sdd001 sdd002 sdd003

| node    |  jobid  |   user    |     start time      |  s_rt   |      end time       |
|---------|---------|-----------|---------------------|---------|---------------------|
| sdd001  | 1702180 | uscqojvc  | 03/27/2010 09:02:01 |  716400 | 2010-04-04 16:02:01 |
| sdd001  | 1704961 | uviqoarl  | 03/31/2010 11:48:16 |  684000 | 2010-04-08 09:48:16 |
|---------|---------|-----------|---------------------|---------|---------------------|
| sdd002  | 1694756 | uviqfjhr  | 03/26/2010 21:08:39 |  720000 | 2010-04-04 05:08:39 |
|---------|---------|-----------|---------------------|---------|---------------------|
| sdd003  | 1704957 | uviqoarl  | 03/31/2010 11:46:07 |  684000 | 2010-04-08 09:46:07 |
```

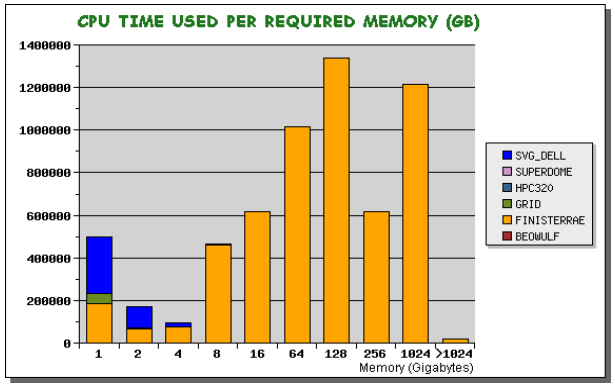
GE integration in Ganglia

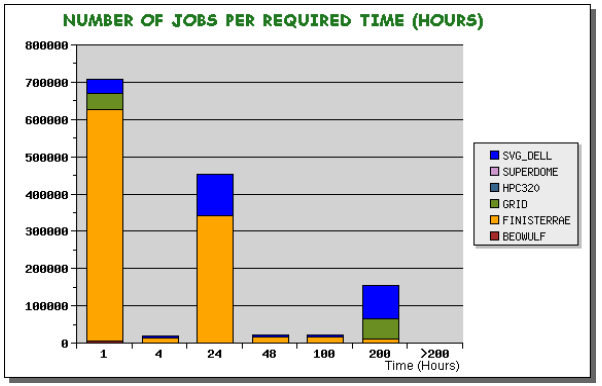
We modified the job detailed view to show the information of all nodes where the job is running. The information is obtained from the XML output of the qstat command (qstat -r -ext -urg -g t -xml -u '*').

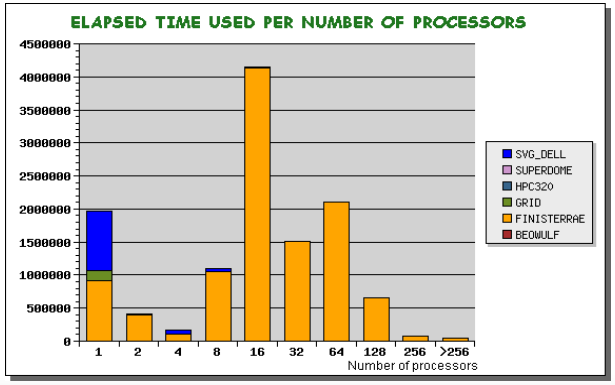


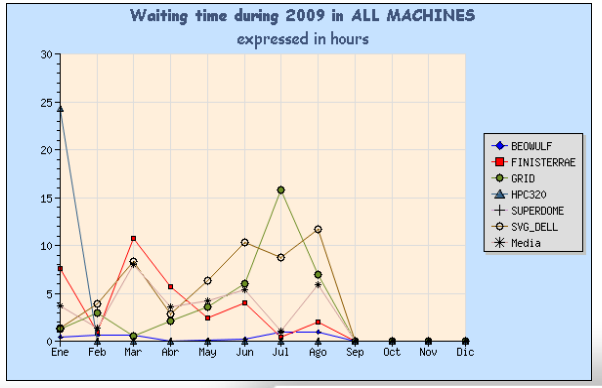
Management of accounting information

- We have created some scripts to process the GE accounting file for each machine and insert this information into a database.
- This information is used to get statics about nodes usage.









CESGA Experience with the Grid Engine batch system

Integration of GE in the gLite middleware (EGEE project)

CESGA
XUNTA DE GALICIA
CONSELLERÍA DE INNOVACIÓN, E INDUSTRIA
CSIC CONSEJO SUPERIOR DE INVESTIGACIONES CIENTÍFICAS
MINISTERIO DE CIENCIA E INNOVACIÓN
FEDER
FONDO EUROPEO DE DESENVOLVEMENTO REXIONAL

Grid Engine in the gLite middleware

- GE JobManager maintainers and developers.

- GE certification testbed.

CREAM-CE supporting GE batch system will be soon in production

- GE stress tests:

https://twiki.cern.ch/twiki/bin/view/LCG/SGE_Stress

We also performed stress tests in Finis Terrae supercomputer to check GE behaviour in a big cluster.

- Documentation:

SGE Cookbook, https://edms.cern.ch/document/858737/1.3

- End-user support for GE to the EGEE community.

project-eu-egee-batchsystem-sge@cern.ch

More info in:

https://twiki.cern.ch/twiki/bin/view/LCG/GenericInstallGuide310

CESGA Experience with the Grid Engine batch system

Esteban Freire García
esfreire@cesga.es **www.cesga.es**

CESGA
XUNTA DE GALICIA
CONSELLERÍA DE INNOVACIÓN, E INDUSTRIA
CSIC CONSEJO SUPERIOR DE INVESTIGACIONES CIENTÍFICAS
MINISTERIO DE CIENCIA E INNOVACIÓN
FEDER
FONDO EUROPEO DE DESENVOLVEMENTO REXIONAL