Grid Engine experience in ***Finis Terrae***, large Itanium cluster supercomputer

Pablo Rey Mayo

Systems Technician, Galicia Supercomputing Centre (CESGA)

Agenda

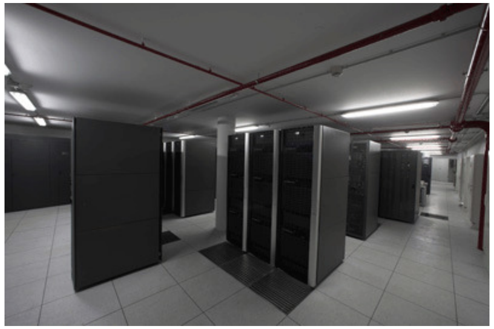
Introducing CESGA

Finis Terrae Architecture

Grid Engine Experience in Finis Terrae

Grid Engine Utilities

Grid Engine and gLite Middleware



ESTABLISHED IN 1993 IN SANTIAGO DE COMPOSTELA (SPAIN)

UNESCO World Heritage 1985
End of St. James Way
100,000 pilgrims in 2007

Past, Present, Future

Users: Three main Galician Universities and Spanish Research Council, Regional weather forecast service

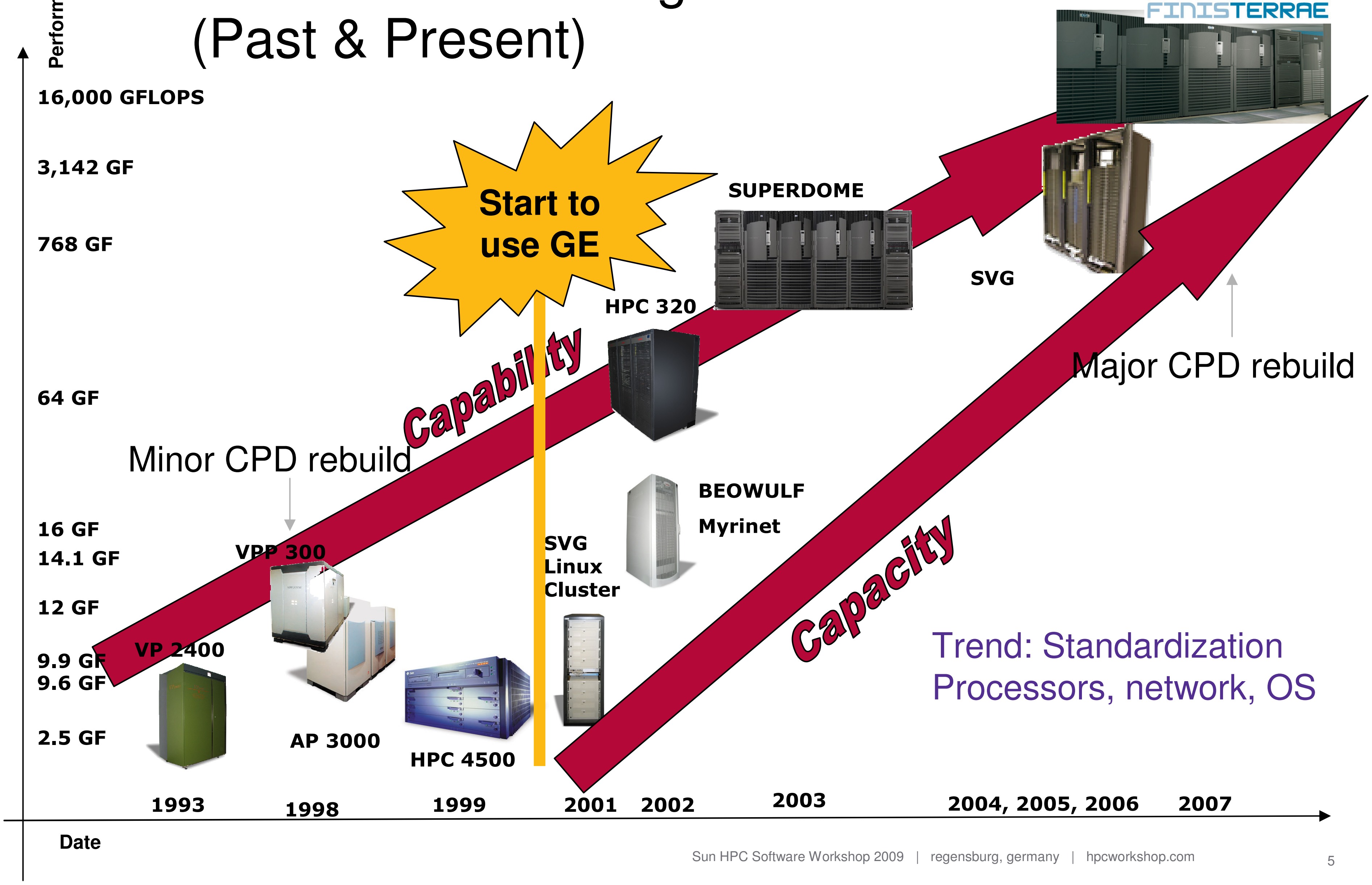
Services: High performance computing, storage and communication resources (RedIris PoP)

Promote new information and communication technologies (HPC & Grid projects)

Future: Centre of Excellence in Computational Science – C^{2}SRC

- 141 research staff
- 75 MM€ (31% building, 23% HPC)

CESGA’s Technological Evolution (Past & Present)



Start to use GE

Minor CPD rebuild

Major CPD rebuild

Trend: Standardization
Processors, network, OS

FINISTERRAE
EXPANDING
THE
FRONTIERS OF KNOWLEDGE

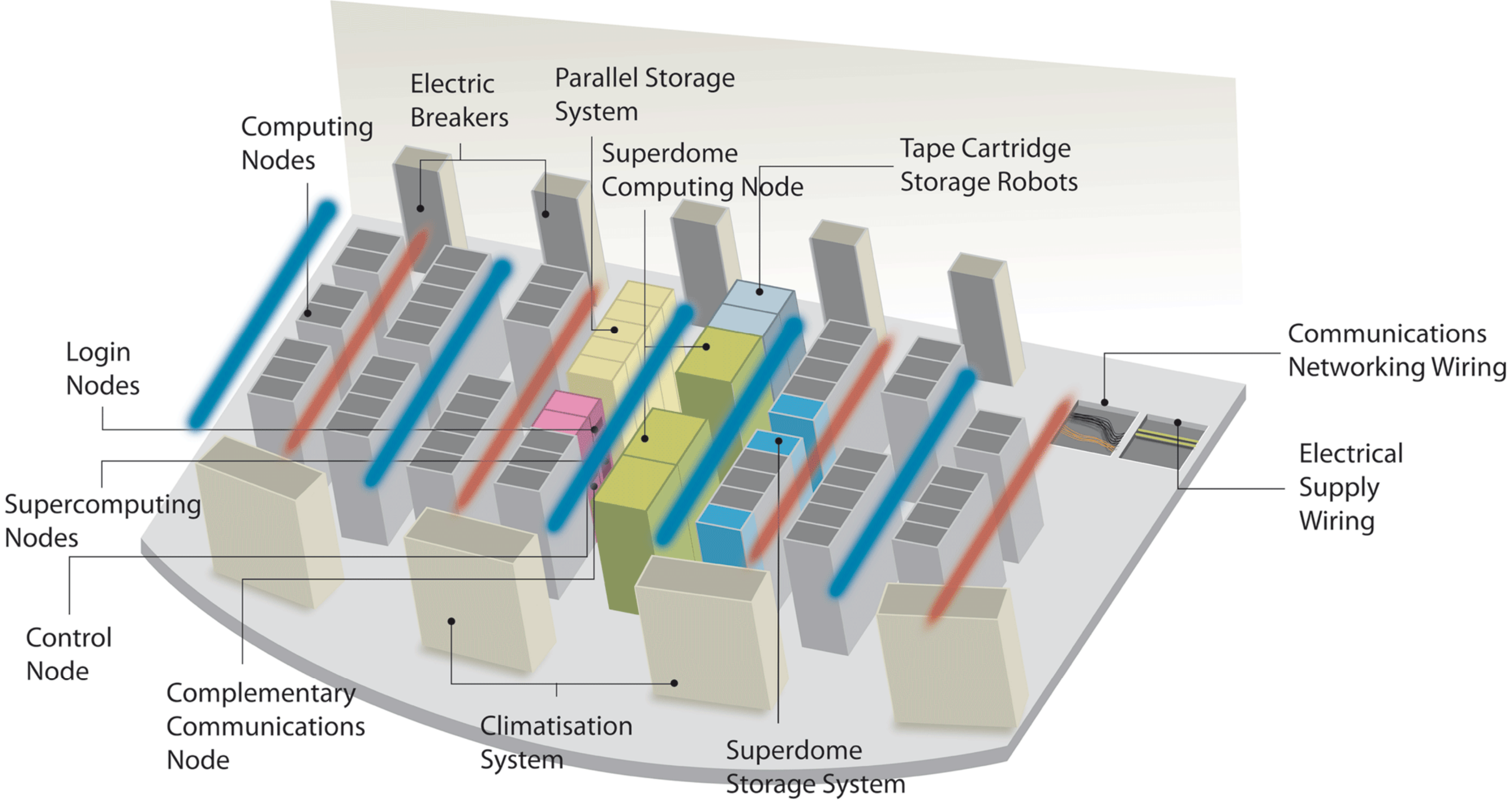
Finis Terrae Supercomputer

Spanish National Unique Scientific & Technological Infrastructure

More than: **16,000 GFLOPS** **2,580 Itanium2 CPUs** **19,640 GB Memory**

OPEN SOURCE

Finis Terrae SMP NUMA Cluster



Technical Specs.

Surface Area: 140 m^2

Weight: 35.000 Kg

Storage: 2.200.000 GB on tape; 390.000 GB on disk

Memory: 19.670 GB

2.528 Processing Cores

142 nodes, each with 16 cores & 128 GB memory

1 node with 128 cores & 1.024 GB memory

1 node with 128 cores & 384 GB memory

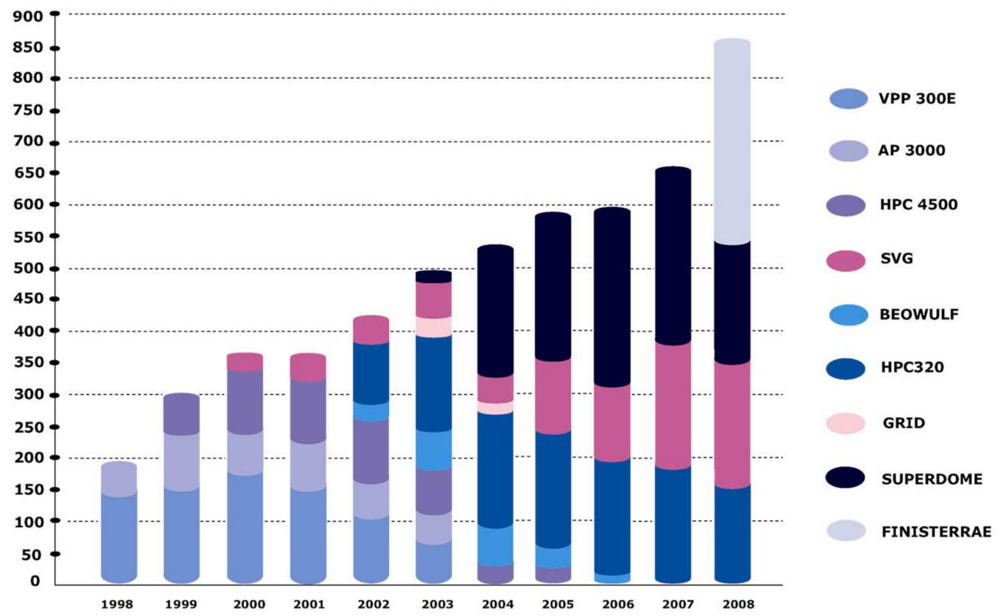
Node Interconnect INFINIBAND 4x DDR at 20 Gbps

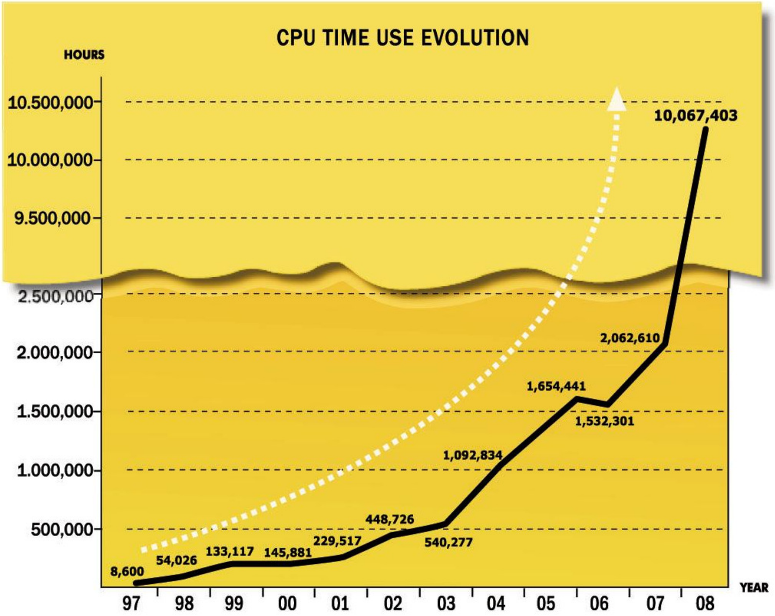
85 Km of interconnect cable

Open Software: Linux, Lustre, Globus...

Demand of computing resources at CESGA

USER ACCOUNT EVOLUTION PER SYSTEM PER YEAR





Grid Engine experience in *Finis Terrae*

Tight HP-MPI Integration

The integration has been tested both with GE 6.1 and 6.2

- GE 6.1 uses rsh. Limitations:
  - 256 connections limit, but one connection against each node/job.
  - No prompt in the session: not aware of any problem in batch jobs.
- GE 6.2 uses an internal communication protocol. Limitations:
  - No prompt.
  - No X11 forwarding.

More info in:

http://wiki.gridengine.info/wiki/index.php/Tight-HP-MPI-Integration-Notes

Tight HP-MPI Integration

Set up the Grid Engine Parallel Environment (PE):

```
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 mpi mpi_rr mpi_1p mpi_2p mpi_4p mpi_8p mpi_16p
```

Set up the hp-mpi environment:

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

qrsh and memory consumption in Itanium

qrsh in GE6.2 adds builtin communication (IJS) with threads.

Memory consumption is about ***2 times the stack limit***.

IF no stack limit, uses 32MB

Number of threads expanded by each qrsh depend of the shell. If shell NOT recognised as “sh”, extra thread consume 30MB.

if stack limit is 256MB each qrsh consumes >500MB of virtual memory ⇒ jobs killed if exceed memory limit.

Optimal memory consumption in Itanium a stack of 1MB.

Set this limit inside the rsh script before qrsh is launched.

Tight Intel-MPI Integration: Extra steps

Create independent rings by adding the variable to avoid processes interfering in the allexit step:

```
export I_MPI_JOB_CONTEXT=“$JOB_ID-$SGE_TASK_ID”
```

If Intel-MPI version is previous to 3.2, rsh script is not able to detect the python version.

```
if [ x$just_wrap = x ]; then
   if [ $minus_n -eq 1 ]; then
      if [ "$cmd" = 'python -c "from sys import version; print version[:3]"' ]; then
         exec $SGE_ROOT/bin/$ARC/qrsh $CESGA_VARS -inherit -nostdin $rhost
   /opt/cesga/sistemas/sge/tools/util/version_python.sh
      else
         exec $SGE_ROOT/bin/$ARC/qrsh $CESGA_VARS -inherit -nostdin $rhost $cmd
      fi
   else
......
```

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.

```
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.

Solved the X11 forwarding limitation in GE 6.2

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)

- Set a ckpt_list in the queue configuration
- Configure a checkpoint environment:

```
[sdiaz@svgd ~]$ 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
```

Checkpointing with gaussian.

Future: Checkpointing with parallel jobs

Utilities

qsub wrapper

Instead of 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 (cluster partitioning)
- 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*).

Support special resources requests (per user limit).

Similar to *Job Submission Verifiers (JSVs)*?.

qsub wrapper: default limits

- Maximum number of processors (*num_proc*slots*): 160
  - Maximum value of num_proc: 16
  - Maximum number of slots:160
- Execution time (s_rt):
  - Sequential jobs: Until 1000 hours
  - Parallel jobs:
    - From 2 to 16 processors: 200 hours
    - From 17 to 32 processors: 100 hours
    - From 33 to 64 processors: 50 hours
    - From 65 to 128 processors: 30 hours
    - From 129 to 160 processors: 20 hours
    - More than 161 processors: 10 hours
- Memory:
  - Per core: 8GB
  - Per node: 112 GB
- H_fsize: 500GB

qsub wrapper: queue distribution

Number of cores	Small queue	Medium queue	Large queue
≤ 4	✓	≤ 1 hour	≤ 1 hour
< 16	≤ 1 hour	✓	≤ 1 hour
≥ 16	≤ 1 hour	✓	✓

queue	Interactive	Small	Medium	Large	meteo
# nodes	2	14	62	62	2

Management of special user requirements

Default limits are not enough.

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=“prey">
    <application id_sol="71">
      <start>2009-06-01</start>
      <end>2010-05-31</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

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

The *qsub* wrapper don’t prioritize jobs so we have to do it after they are submitted:

- 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 ⇒ Move into execution its pending jobs.

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 send 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 (*qhost*).

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?

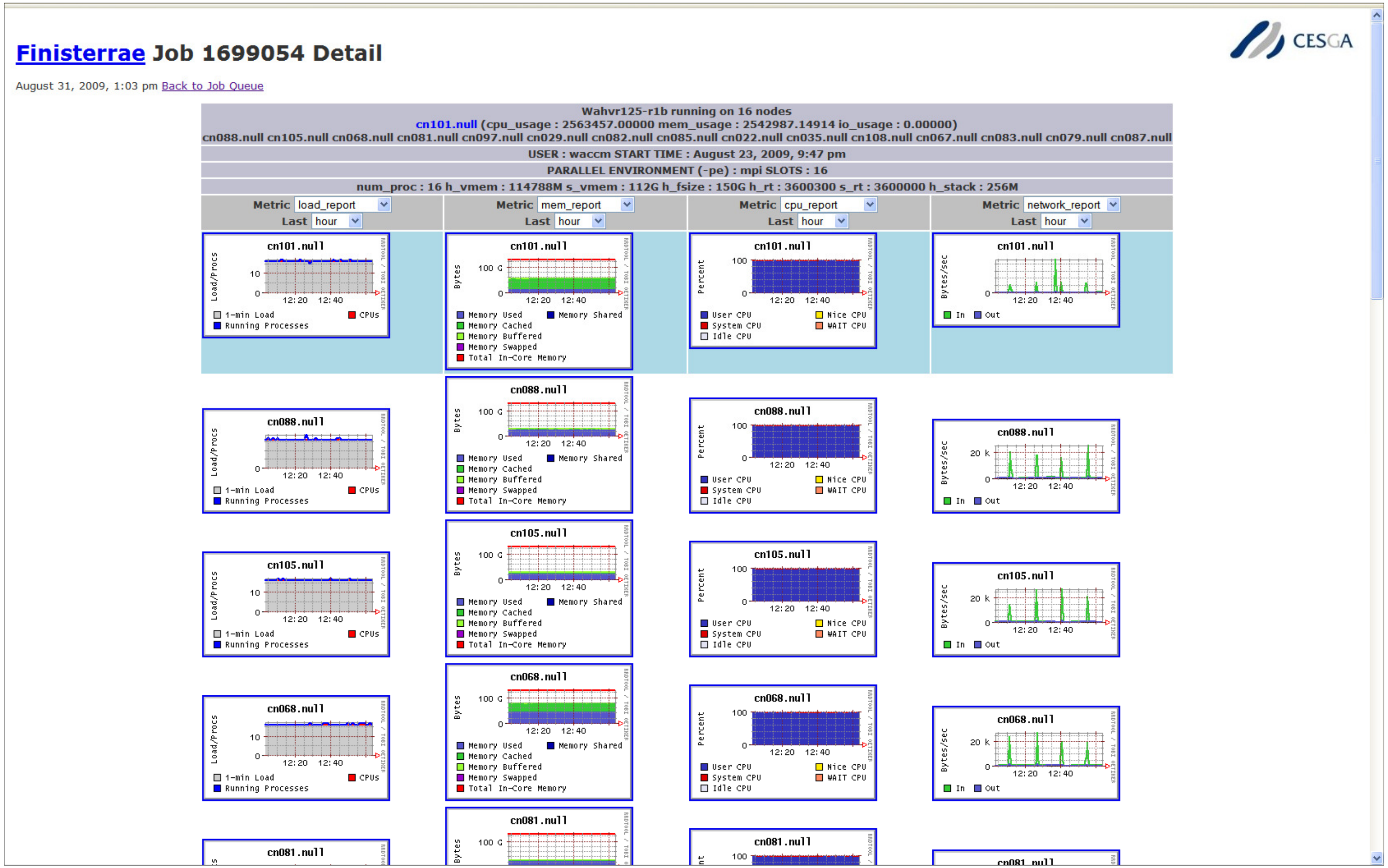
Sometimes we need to know when a node will be free (maintenance, tests, …).

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 | 08/27/2009 09:02:01 |   716400 | 2009-09-04 16:02:01 |
| sdd001 | 1704961 | uviqoarl | 08/31/2009 11:48:16 |   684000 | 2009-09-08 09:48:16 |
|--------|---------|----------|---------------------|----------|---------------------|
| sdd002 | 1694756 | uviqfjhr | 08/26/2009 21:08:39 |   720000 | 2009-09-04 05:08:39 |
|--------|---------|----------|---------------------|----------|---------------------|
| sdd003 | 1704957 | uviqoarl | 08/31/2009 11:46:07 |   684000 | 2009-09-08 09:46:07 |
-----------------------------------------------------------------------------------------------
```

GE integration in Ganglia



Finisterrae Job 1699054 Detail

August 31, 2009, 1:03 pm Back to Job Queue

Wahvr125-r1b running on 16 nodes
cn101.null (cpu_usage : 2563457.00000 mem_usage : 2542987.14914 io_usage : 0.00000)
cn088.null cn105.null cn068.null cn081.null cn097.null cn029.null cn082.null cn085.null cn022.null cn035.null cn108.null cn067.null cn083.null cn079.null cn087.null

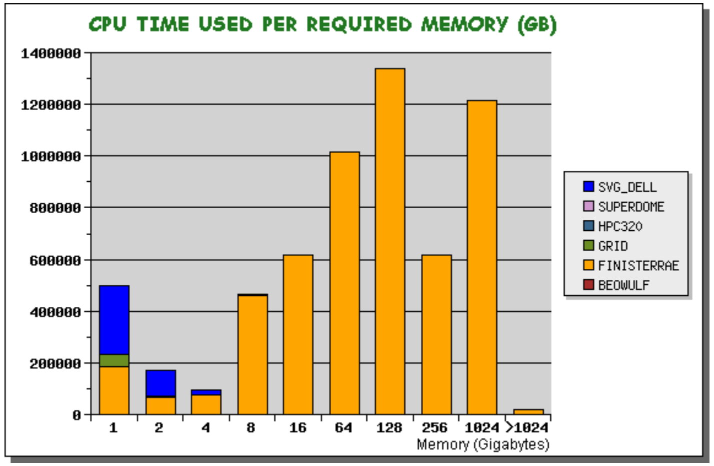
USER : waccm START TIME : August 23, 2009, 9:47 pm

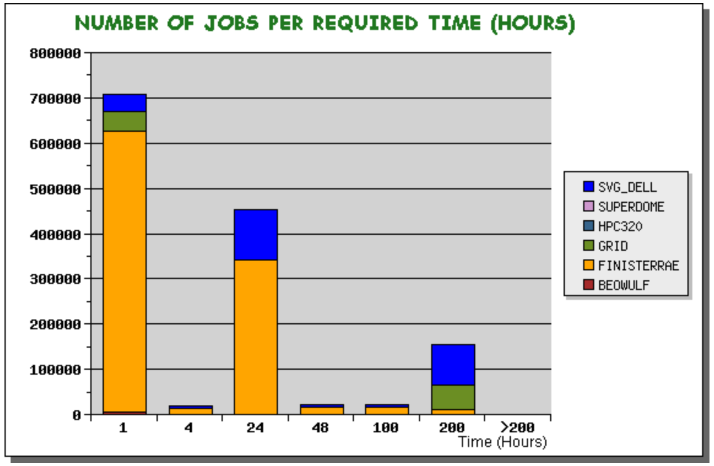
PARALLEL ENVIRONMENT (-pe) : mpi SLOTS : 16

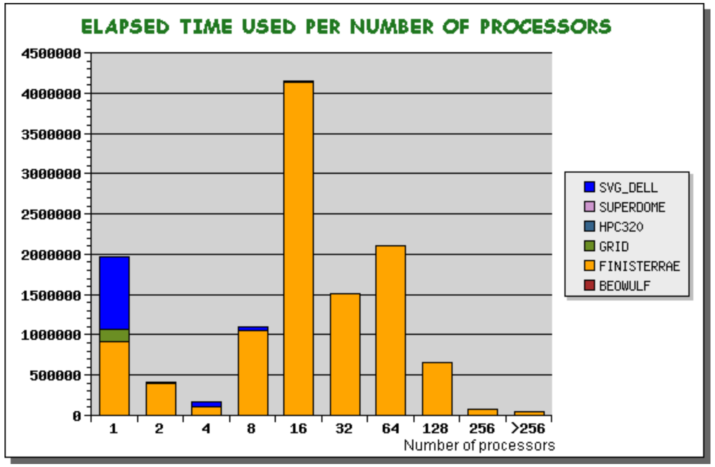
num_proc : 16 h_vmem : 114788M s_vmem : 112G h_fsize : 150G h_rt : 3600300 s_rt : 3600000 h_stack : 256M

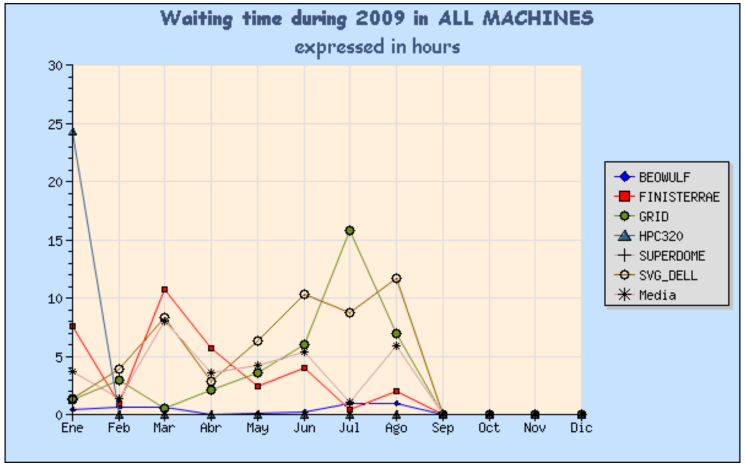
Metric load_report Last hour	Metric mem_report Last hour	Metric cpu_report Last hour	Metric network_report Last hour

Management of accounting information









Integration of GE in the gLite middleware (EGEE project)

Grid Engine in the gLite middleware

GE JobManager maintainers and developers.

GE certification testbed.

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:

- GE Cookbook (https://edms.cern.ch/document/858737/1.3)

End-user support for GE to the EGEE community.

More info in:

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

Grid Engine experience in Finis Terrae, large Itanium cluster supercomputer.

Pablo Rey Mayo
prey@cesga.es. www.cesga.es