UNIVERSIDAD
REY JUAN CARLOS

Máster Universitario en Software Libre

Curso Académico 2012/2013
Campus Fuenlabrada, Madrid, España

MSWL-THESIS - Proyecto Fin de Master

Distributed Batch Processing

Autor: Antonio Carmona Álvarez
Tutor: Gregorio Robles

Contents

List of Figures

Chapter 1

Introduction

The main target of every information or computer system is the processing of information , there are differents ways of do this depending of the type of information , the dependencies of the processing of this information etc, but in big terms we can make a big division in the processing of the information between “online” processing , that groups all type of processing that need a comunication or direct syncronous comunication with a person (with a thecnical profile or not) and “batch” processing that refers to a type of processing of information that is asyncronous and doesn’t need maintain a direct comunication and doesn’t need human intervention in any moment of the processing. Each of them are needed for differents purposes depending of the nature of the data to process or depending of the form of manipulation of the information that is needed.

In most cases a project needs both kinds of processing of information, a online processing is used when the intervention of a human person is needed, due that there are data that must be introduce into the system by human person or due to a intervention of a human is needed during the processing of the information for control and the take of some decisions that are very dificult to automatize or predefine because are product of a “fuzzy logic”.

In the other hand we have a specific information processing that can be well defined making posible the definition of a script or a tasks list made in whatever programming language and that contains all the needs that the program would has and, therefore, the processing of the information is a very well sorted list of steps that has a final generation or transformation of data. The user only has to check the finalization of the process and if the data has been well generated or not.

The nature of the batch processing makes that the needs will be different that the case of the online processing of data , firstable is a more flexible form of processing because there are not a person waiting in a screen for the visualization of the resultant data , and has a more flexible range of execution reason due to wich the most batch process are executing during the night when the servers has more resources due the lack of online users.

1.1 Batch environment elements

A well organized batch environment must dispose:

- A set of servers that acts as execution nodes that can execute every process and share the information with the rest of the nodes of the cluster/grid.

- Each node must has a set of enough physical resources (hardware) for the execution of the process in terms of memory, cpu, disk space etc.
- Each node must has a set of logical resources (software) as compilers, virtual machines, determined libraries etc.
- A system to launch process, for user requested at the moment, and time scheduled.
- A system that organize in queues the execution of the process with the objetive of control the use of the resources in manner that there aren't risk of overload or underutilization of the resources of the system.

1.2 Implementations of a batch environment

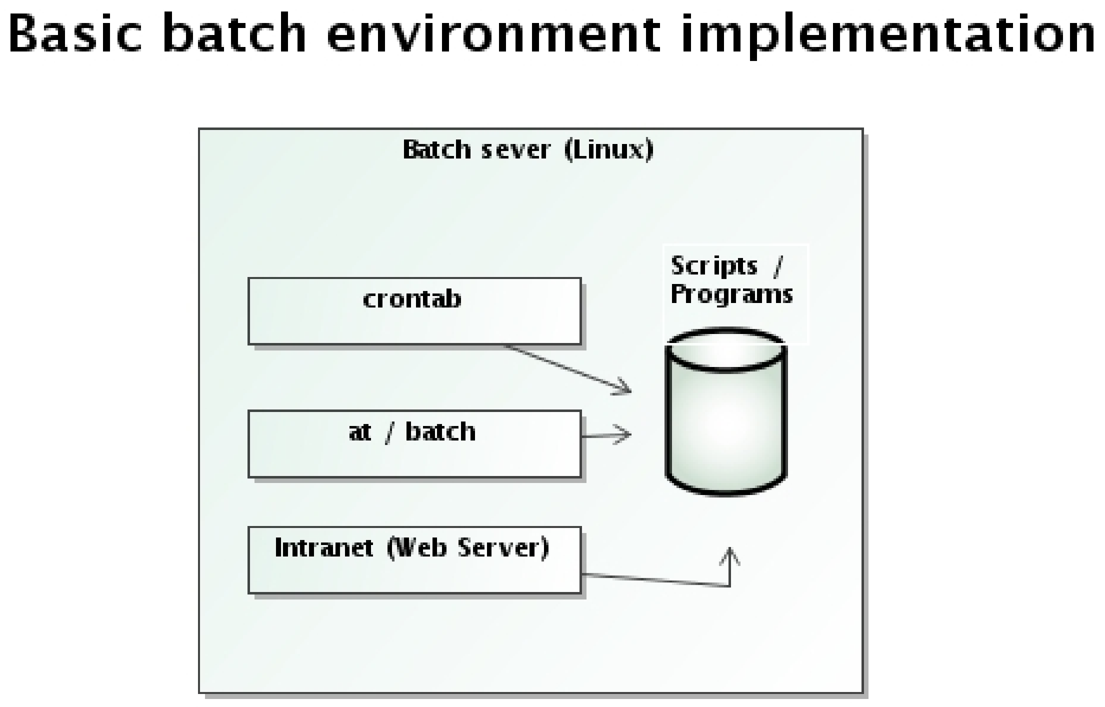
1.2.1 Basic implementation

A basic implementation of a batch environment in a Linux server would be formed with:

- Only one server or node that control, monitorization, launching and execution of user process.
- The at command : for execute process in a determined date and time
- The crontab daemon for execute process periodically
- The batch command: for launch process only when the server has enough free cpu.
- A Intranet, a set of cgis for launch process into the system.

This is the most easier implementation of a batch environment but has some problems like that there are no High Availability , there are no control of the use of the resources of the system and we could have lots of problems if we need to do some maintenance tasks over this server.

Figure 1.1: Batch environment basic implementation

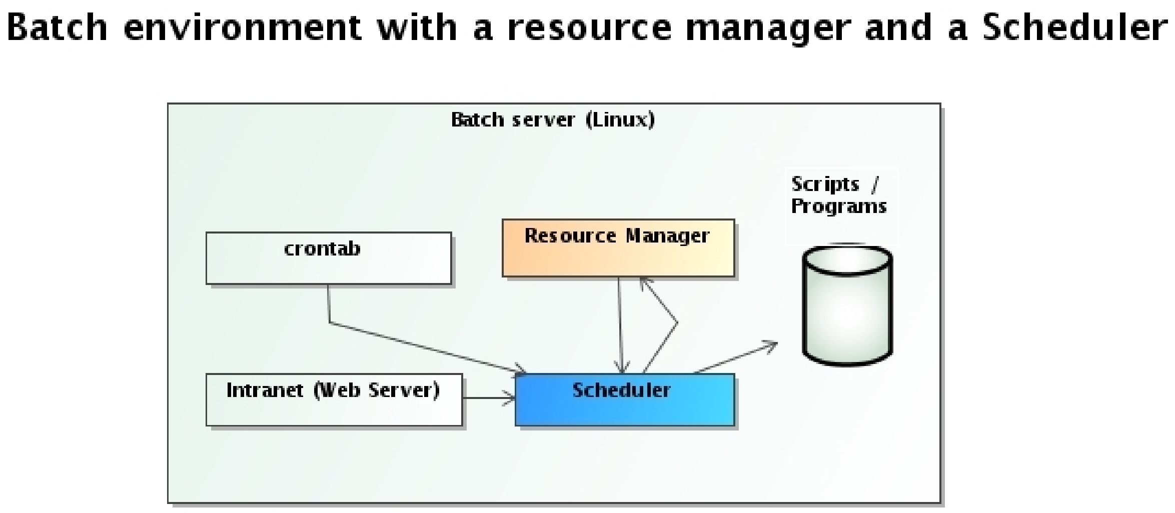


1.2.2 Scheduler and a Resource Manager in one node.

A more advance implementation of a batch environment would be the formed by:

- Only one server or node that controls, monitorize, launch and execute the user process.
- A Resource Manager that controls the availability of the resources of the systems.
- A Scheduler with a system of queues, ACLs, QoS , that launches process according the availability of the resources of the system and that be able to execute process in a determined date and time.
- A crontab daemon for launch process periodically
- A Intranet , a set of cgis for launch process into the system

Figure 1.2: Batch environment implementation with a Resource manager an Scheduler in one node



1.2.3 Cluster or Grid

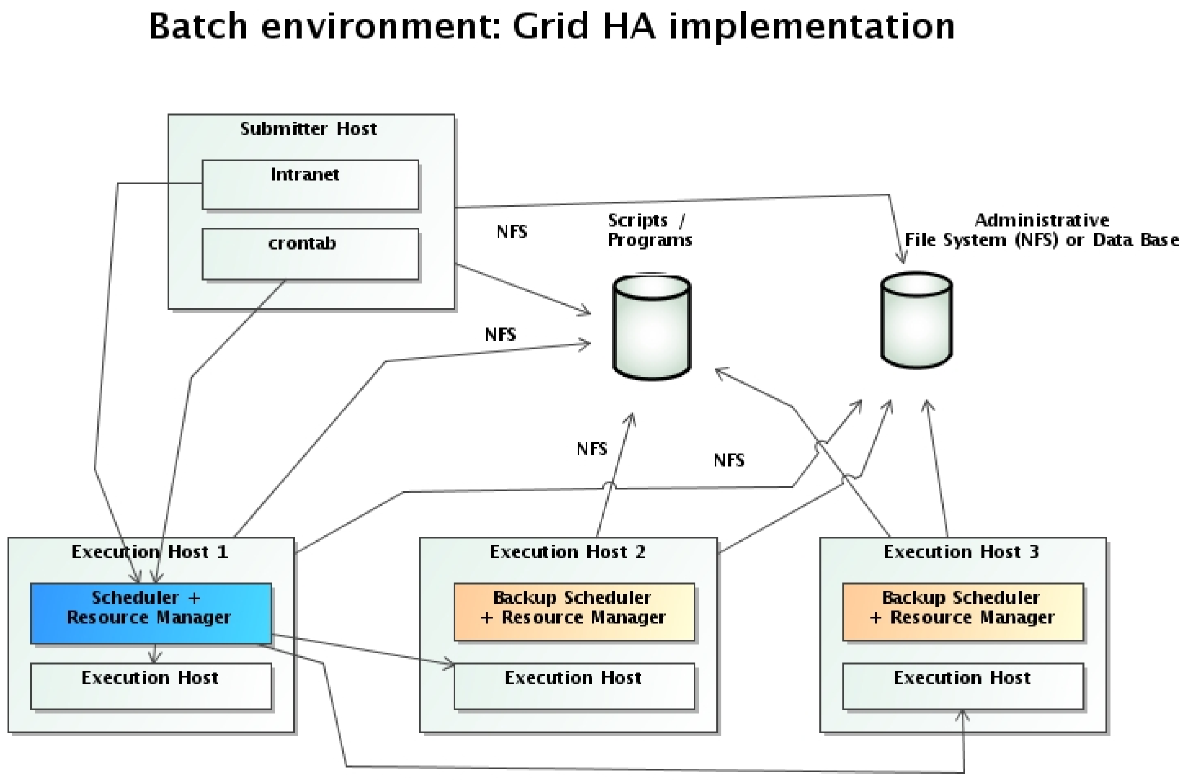
The use of a grid of server in a bacth system provide us:

- Scalability : is a highly scalable DRM system
- Flexibility: is possible to customize the system to exactly fit your needs.
- Advanced scheduler: supports a large variety policies for scheduling process.
- Reliability: its very easy to mantaing and keep it working.

The most recommended implementation for a batch environment is mounting a Grid or a Cluster, a set of batch servers that acts as only one. with:

- A specific monitor / control server
- Various execution servers
- A Resource Manager that controls the availability of the resources of the system.
- A Scheduler with a system of queues, ACLs, QoS , that launches process according the availability of the resources of the system and that be able to execute process in a determined date and time.
- A crontab daemon for launch process periodically
- A Intranet , a set of cgis for launch process into the cluster / grid

Figure 1.3: Batch environment Grid HA implementation


Batch environment: Grid HA implementation
Submitter Host
Intranet
crontab
NFS
Scripts / Programs
Administrative File System (NFS) or Data Base
NFS
NFS
NFS
Execution Host 1
Scheduler + Resource Manager
Execution Host
Execution Host 2
Backup Scheduler + Resource Manager
Execution Host
Execution Host 3
Backup Scheduler + Resource Manager
Execution Host

Thus, according to the explained before, the type of implementation that give us more flexibility , security and a best exploitation of the resources of the system is the implementation of a grid of servers with differents roles. There are two main FLOSS projects that allow us have a batch environment using a grid of Servers: “Torque/Maui” and “Sun Grid Engine” , both will be studied in the next chapters.

Chapter 2

Torque / Maui

2.1 Torque

TORQUE (Terascale Open-source Resource and QUEue manger) is a FLOSS resource manager (based in the PBS(Portable Batch System) manager developed by NASA) that allow us through a server process (pbs_server) and a client process ,that execute the process batch (pbs_mom) in every node host, access to the resources of the differents servers and can execute in them the batch process which we need.

Torque include a basic process scheduler called pbs_sched
Elements in Torque implementation:

- Server Host: process:
  - PBS Server : pbs_server : Manager that queue the process.
  - Scheduler : pbs_sched : Basic Scheduler that execute the client process.
  - Mom Client : Torque client that must be installed and running in all the execution hosts.
- Node Host 1: Mom Client : Torque client that must be installed and running in all the execution hosts.
- Node Host 2: Mom Client : Torque client that must be installed and running in all the execution hosts.
- Node Host N. . .

2.1.1 Installation and configuration

The default install include the next components:

- PBS Server : pbs_server : Manager that queue the process
- Scheduler : pbs_sched : Scheduler that execute the client process

Figure 2.1: Cluster Resources Torque / Maui Logo [6]

CLUSTER RESOURCES INC.

- Mom Client : Torque client that must be installed and running in all the execution hosts.

The server name must exists in an DNS server or in the /etc/hosts file.
For the first start of the server:

```
[root@oscar torque-2.2.1]# ./torque.setup root
initializing TORQUE (admin: root@oscar)
Max open servers: 4
Max open servers: 4
[root@oscar torque-2.2.1]#
```

For check that is running correctly:

```
[root@oscar torque-2.2.1]# qstat -q
server: oscar
Queue            Memory CPU Time Walltime Node  Run Que Lm  State
---------------- ------ -------- -------- ----  --- --- --  -----
batch              --      --       --      --    0   0 --   E R
                                               ----- -----
                                                   0     0
```

If we want to erase a previous configuration of a pbs server and start a new once:

```
#pbs_server -t create
```

2.1.2 Configuration of the PBS Server Host

The configuration files are in /var/spool/torque/
File with all the nodes of the system: /var/spool/torque/server_priv/nodes :

```
 oscar np=1000: Number of maximum concurrent process that are allowed in this
 node, if i don't write anything here, only one process will be allowed at the
 same time.
 charlie np=100
 ...
```

For the configuration of the PBS server we must use the command qmgr, before is need give the correct permissions to the file /usr/local/sbin/pbs_iff:

```
chmod u+s /usr/local/sbin/pbs_iff
```

For view the current configuration of the PBS server we must execute:

```
[root@oscar torque-2.2.1]# qmgr -c 'p s'
#
# Create queues and set their attributes.
#
#
# Create and define queue batch
#
create queue batch
set queue batch queue_type = Execution ---> Type of queue, can be Execution
(normal) or Route (redirect to other queue)
set queue batch resources_default.nodes = 1
set queue batch resources_default.walltime = 01:00:00
set queue batch resources_available.nodect = 99999
set queue batch enabled = True  ----> Enable the queue of process
set queue batch started = True  ----> Enable the execution of the queued process
#
# Set server attributes.
#
set server scheduling = True   --------> Needed for use pbs_sched
set server managers = root@oscar
set server operators = root@oscar
set server default_queue = batch
set server log_events = 511
set server mail_from = adm
set server resources_available.nodect = 99999
set server scheduler_iteration = 600
set server node_check_rate = 150  ----> specifies the minimum duration (in seconds)
that a node can be unresponsive to server queries before being marked down by
the pbs_server daemon
set server tcp_timeout = 6     ----> specifies the pbs_server to pbs_mom
TCP socket timeout in seconds.
set server mom_job_sync = True  ---> specifies that the pbs_server will
synchronize its view of the job queue and resource allocation with
compute nodes as they come online.  If a job exists on a computer node
in a pre-execution or corrupt state, it will be automatically cleaned
up and purged. (enabled by default in TORQUE 2.2.0 and higher)
set server pbs_version = 2.2.1
set server keep_completed = 300  --> Time that Maui retains information about a
process currently terminated (C state)
```

The Walltime attribute is the maximum execution time that a determined node allow for each process, if a process last more time that the indicate in this attribute, the process is eliminated, by default this value is the indicated in the configuration of each node, in most cases the best option is disable the walltime with the next command:

```
qmgr -c 'unset server resources_default.walltime'
```

The walltime indicated in the execution of a process (either indicate in the qsub command or in a PBS directive) is used also as a priorization factor that can help to known to the Scheduler if is a long or a short process and priorizate them according this circunstance.
Another important attribute is resources_default.nodect that specifies the cumulative resources available to all jobs running in the server. Is important setup this attribute to a high value:

```
qmgr -c 'set server resources_default.nodect=999999'
```

Configure also the tcp_timeout attribute:

```
qmgr -c 'set server tcp_timeout = 60'
```

Configuration of the queues:

```
Qmgr: create queue a
Qmgr: set queue a queue_type = Execution
Qmgr: set queue a Priority = 1
Qmgr: set queue a max_running = 4
Qmgr: set queue a enabled = True
Qmgr: set queue a started = True

Qmgr: create queue b
Qmgr: set queue b queue_type = Execution
Qmgr: set queue b Priority = 1
Qmgr: set queue b max_running = 2
Qmgr: set queue b enabled = True
Qmgr: set queue b started = True

Qmgr: create queue f
Qmgr: set queue f queue_type = Execution
Qmgr: set queue f Priority = 1
Qmgr: set queue f max_running = 3
Qmgr: set queue f enabled = True
Qmgr: set queue f started = True

Qmgr: create queue k
Qmgr: set queue k queue_type = Execution
Qmgr: set queue k Priority = 1
Qmgr: set queue k max_running = 5
Qmgr: set queue k enabled = True
Qmgr: set queue k started = True

Qmgr: create queue o
Qmgr: set queue o queue_type = Execution
Qmgr: set queue o Priority = 1
Qmgr: set queue o max_running = 12
Qmgr: set queue o enabled = True
```

```
Qmgr: set queue o started = True

Qmgr: create queue i
Qmgr: set queue i queue_type = Execution
Qmgr: set queue i Priority = 1
Qmgr: set queue i max_running = 5
Qmgr: set queue i enabled = True
Qmgr: set queue i started = True
```

The queues exists physically in the disk in the path: /var/spool/torque/server_priv/queues
If we want to erase a queue:

```
qmgr -c 'delete queue <Nombre_de_cola>'
```

Show view about a server, queues , nodes:

```
qmgr -c 'list server'
qmgr -c 'list queue a'
qmgr -c 'list node oscar'
```

Start / Stop of the process of the server host:

- Start / Stop PBS Server

  ```
  #pbs_server   : Start PBS server
  #qterm        : Stop PBS server
  ```

- Start / Stop Basic PBS Scheduler

  ```
  #pbs_sched       : Start basic pbs Scheduler
  #kill <proceso> : Stop basic pbs Scheduler
  ```

2.1.3 Configuration of the PBS Nodes

Cliente MOM Installation
There are two forms of install the MOM client in the nodes:

- With the same .tar that install the PBS server but instead execute make, execute: make install_clients
- Executing make packages and using the generated shell: torque-package-clients-linux-i686.sh

Client MOM configuration:
The configuration files of the MOM client are in the path: /var/spool/torque/mom_priv
File: config:

```
$pbsserver oscar   ---> Indicate where is the PBS server
$job_output_file_umask  022   --> Mask of the output files generated.
```

If the node is remote and we have mounted NFS file systems for the comunication between nodes, we must put this into the config file:

```
$usecp <full hostname of head node>:<home directory path on head node> <home
directory path on worker node>
```

Start / Stop MOM client y nodes:

```
#pbs_mom        :  Start the client MOM
#kill <proceso> :  Stop the client MOM
```

2.1.4 Operation

Launch a process with the command : Qsub

```
qsub [-a date_time] [-A account_string] [-c interval] [-C directive_prefix] [-e path]
[-h] [-I] [-j join] [-k keep] [-l resource_list] [-m mail_options] [-M
user_list] [-N name] [-o path] [-p priority] [-q destination] [-r c] [-S
path_list] [-u user_list] [-v variable_list] [-V] [-W additional_attributes] [-z] [script]

  -a date_time  Execute the job only after date_time.
  -A Account for the launch of the process
  -l List of resources required by the process
  -p Priority of the process
  -q Queue and optionally server where the process will be launched.
  -r  Job is rerunnable
  -V  Export environment variables in your current environment to the job.
  -I  Run interactively, usually for testing purposes
  -t Execute the process consecutively n times.
  -u <user> : User that will execute the script.
  -w For indicate determined FLAGS.
         depend=
                =after:jobid:jobid..
                =afterok:jobid:jobid..
                =afternook:jobid:jobid..
                =afterany:jobid:jobid..
                =before:jobid:jobid..
                =beforeok:jobid:jobid..
                =beforenotok:jobid:jobid..
                =beforany:jobid:jobid..
```

Execute a script with parameters with qsub:

```
qsub -q <COLA> -v param1=val,param2=val  .....  script.sh      (Really are
environment variables used inside the script
$param1 and $param2)
```

Specify execution time:

```
qsub -l nodes=1,walltime=00:05:00,cput=00:01:00 script.sh

This process will execute with a maximum time of 5 minutes , 1 of them as a
maximum cpu time.
```

PBS directives indicates in the users shells:

```
#!/bin/sh
#PBS -q batch                           Desirable queue batch
#PBS -l nodes=1:ppn=1                   Requeriments of the execution: 1 node
and one cpy by node
#PBS -l walltime=00:15:00               Estimated duration of the
process , if the process takes more time that here specified a warnning
is sent to the user.
#PBS -N NombrelDelScript                Name of the process (its not a id of
the process)
#PBS -m bea                             Treatment of the input / output of the
script
#PBS -M email@email.com  Email for the notifications
```

Others PBS directives:

```
#PBS -N myJob  Assigns a job name. The default is the name of PBS job script.
#PBS -l nodes=4:ppn=2  The number of nodes and processors per node.
#PBS -q queuename  Assigns the queue your job will use.
#PBS -l walltime=01:00:00  The maximum wall-clock time during which this job can run.
#PBS -o mypath/my.out  The path and file name for standard output.
#PBS -e mypath/my.err  The path and file name for standard error.
#PBS -j oe  Join option that merges the standard error stream with the standard output stream of the job.
#PBS -W stagein=file_list  Copies the file onto the execution host before the job starts. (*)
#PBS -W stageout=file_list  Copies the file from the execution host after the job completes. (*)
#PBS -m b  Sends mail to the user when the job begins.
#PBS -m e  Sends mail to the user when the job ends.
#PBS -m a  Sends mail to the user when job aborts (with an error).
#PBS -m ba  Allows a user to have more than one command with the same flag by grouping the messages
 together on one line, else only the last command gets executed.
#PBS -r n  Indicates that a job should not rerun if it fails.
#PBS -V  Exports all environment variables to the job.
```

Therefore, there are two forms of indicate directives to the PBS server for configure the execution of a process, from the qsub command in the moment of the launching or with PBS directives in the beginning of the script (or both forms) . The parameters specified in the command qsub prevails over the specified in the script with PBS directives.
Environment variables:

```
Variable   Description
PBS_ARRAYID  value of job array index for this job. (Version 2.2.0 and later)
PBS_TASKNUM  number of tasks requested
PBS_MOMPORT  active port for mom daemon
PBS_JOBNAME  user specified jobname
PBS_JOBID  The identifier that PBS assigns to the job.
PBS_JOBCOOKIE  job cookie
PBS_NODENUM  node offset number
PBS_NODEFILE  file containing line delimited list on nodes allocated to the job
PBS_ENVIRONMENT This is set to PBS_BATCH for batch jobs and to PBS_INTERACTIVE for interactive jobs.
PBS_QUEUE  job queue
PBS_O_WORKDIR  jobs submission directory
PBS_O_HOME  home directory of submitting user
PBS_O_LOGNAME  name of submitting user
PBS_O_LANG  language variable for job
PBS_O_SHELL  script shell
PBS_O_JOBID  unique pbs job id
PBS_O_HOST  host on which job script is currently running
PBS_O_LOGNAME  The login name on the machine on which the qsub was run.
PBS_O_QUEUE  The original queue to which the job was submitted.
PBS_O_PATH  path variable used to locate executables within job script
```

2.1.5 Examples

Example of execution of a batch using Torque in the same host where the PBS server is installed. Check of the state of the nodes before start the execution:

```
[root@oscar bin]# pbsnodes -a
oscar
    state = free   (Is OK,  state = down is that node is unavalaible)
    np = 1
    ntype = cluster
    status = opsys=linux,uname=Linux oscar 2.6.18-8.1.15.el5xen
```

Check the status of the scheduler:

```
[root@oscar ~]# qstat -q
server: oscar
Queue            Memory CPU Time Walltime Node  Run Que Lm  State
---------------- ------ -------- -------- ----  --- --- --  -----
batch              --      --       --      --    0   0 --   E R     There are
no process in the system
----- -----
                                                  0     0
```

Launch a process:

```
qsub /usr/local/test/shell1.sh
10.oscar   ------------> Id returned
```

Display of the state of the process:

```
[root@oscar ~]# qstat
Job id                    Name             User            Time Use S Queue
------------------------- ---------------- --------------- -------- - -----
10.oscar                  shell1.sh        test              0      R batch
State R: Process executing
```

```
[root@oscar ~]# qstat -q
server: oscar
Queue            Memory CPU Time Walltime Node  Run Que Lm  State
---------------- ------ -------- -------- ----  --- --- --  -----
batch              --      --       --      --    1   0 --   E R        -- 1
Process Executing
----- -----
                                                  1     0
```

Display of the status of a process at the end of its execution:

```
[root@oscar ~]# qstat
Job id                    Name             User            Time Use S Queue
------------------------- ---------------- --------------- -------- - -----
10.oscar                  shell1.sh        test       00:00:00   C     batch
Sate C: Process finished

[root@oscar ~]# qstat -q
server: oscar
Queue            Memory CPU Time Walltime Node  Run Que Lm  State
---------------- ------ -------- -------- ----  --- --- --  -----
batch              --      --       --      --    0   0 --   E R         -- None
process in execution
----- -----
                                                  0     0
```

Display of the logs: Standard an Error output:

```
The files with the content of the standard and error output are created in the
home directory of the user that launch the process, in two files with the next
format:

  $HOME/<name-shell>.o<IdReturnedBy-qsub>
  $HOME/<name-shell>.e<IdReturnedBy-qsub>
```

Cancel a job:

```
$ qstat
Job id           Name             User             Time Use S Queue
---------------- ---------------- ---------------- -------- - -----
4807             scatter          user01           12:56:34 R batch
...
$ qdel -m "Message for the owner of the process.." 4807
```

2.2 Maui: Advance Scheduler for Torque

The scheduler included with Torque (PBS scheduler) is very simple, exists another FLOSS scheduler that we can use instead , called Maui.
Maui is therefore a scheduler that will launch the process in the correct moment in base of some queues defined and some quality of service parameters defined by the administrator.

Elements:

- Tasks: Indivisible set of resources of the same node.
- The resources are grouped in Tasks
- A Task can consist in for example a one processor, 2GB of memory and 10 GB of disk .
- A Reservation consist in one or more Tasks
- Job: Is a user process or script to execute.

2.2.1 Installation and configuration

Define the next environment variables:

```
export TORQUESPOOL=/var/spool/torque
export TORQUEPREFIX=/usr/local/torque
export MAUISPOOL=/var/spool/maui
export MAUIPREFIX=/usr/local/maui

tar -xvf <fichero.tar>
./configure --with-pbs   : --with-pbs: This makes Maui compatible with Torque.
./make
./make install
```

Recommended links:

```
PATHLINKS=/usr/local/bin
ln -s /usr/local/maui/bin/showres $PATHLINKS/showres
ln -s /usr/local/maui/bin/setres $PATHLINKS/setres
ln -s /usr/local/maui/sbin/maui $PATHLINKS/maui
ln -s /usr/local/maui/bin/showq $PATHLINKS/showq
ln -s /usr/local/maui/bin/diagnose $PATHLINKS/diagnose
ln -s /usr/local/maui/bin/releaseres $PATHLINKS/releaseres
ln -s /usr/local/maui/bin/runjob $PATHLINKS/runjob
ln -s /usr/local/maui/bin/checkjob $PATHLINKS/checkjob
ln -s /usr/local/maui/bin/canceljob $PATHLINKS/canceljob
ln -s /usr/local/maui/bin/checknode $PATHLINKS/checknode
ln -s /usr/local/maui/bin/showgrid $PATHLINKS/showgrid
ln -s /usr/local/maui/bin/showstats $PATHLINKS/showstats
ln -s /usr/local/maui/bin/showstart $PATHLINKS/showstart
ln -s /usr/local/maui/bin/showstate $PATHLINKS/showstate
```

Maui config file: /usr/local/maui/maui.cfg

```
#RMCFG[CHARLIE] TYPE=PBS@RMNMHOST@    : Comment this line
RMTYPE[0]                       PBS      :  Put this line
RMSERVER[0]                     NombreNodo  : Put this line
```

Full configuration file as example:

```
# maui.cfg 3.2.6p19
SERVERHOST            oscar
# primary admin must be first in list
ADMIN1                root
ADMINHOST             oscar
# Resource Manager Definition
RMTYPE[0]                       PBS
RMSERVER[0]                     oscar
#RMPORT[0]                      <port>
RMPOLLINTERVAL                  0:00:30
DEFAULTCLASSLIST                default
# Allocation Manager Definition
AMCFG[bank]  TYPE=NONE
# full parameter docs at http://supercluster.org/mauidocs/a.fparameters.html
# use the 'schedctl -l' command to display current configuration
RMPOLLINTERVAL        00:00:30
#SERVERPORT            42559
SERVERMODE            NORMAL
# Admin: http://supercluster.org/mauidocs/a.esecurity.html
LOGFILE               maui.log
LOGFILEMAXSIZE        10000000
LOGLEVEL              3
# Job Priority: http://supercluster.org/mauidocs/5.1jobprioritization.html
QUEUETIMEWEIGHT       1
# FairShare: http://supercluster.org/mauidocs/6.3fairshare.html
#FSPOLICY              PSDEDICATED
#FSDEPTH               7
#FSINTERVAL            86400
```

```
#FSDECAY              0.80
# Throttling Policies: http://supercluster.org/mauidocs/6.2throttlingpolicies.html
NODECFG[oscar] PARTITION partition1
# NONE SPECIFIED
# Backfill: http://supercluster.org/mauidocs/8.2backfill.html
BACKFILLPOLICY        FIRSTFIT
RESERVATIONPOLICY     CURRENTHIGHEST
# Node Allocation: http://supercluster.org/mauidocs/5.2nodeallocation.html
NODEALLOCATIONPOLICY  MINRESOURCE
# QOS: http://supercluster.org/mauidocs/7.3qos.html
# QOSCFG[hi]  PRIORITY=100 XFTARGET=100 FLAGS=PREEMPTOR:IGNMAXJOB
# QOSCFG[low] PRIORITY=-1000 FLAGS=PREEMPTEE
# Standing Reservations: http://supercluster.org/mauidocs/7.1.3standingreservations.html
# SRSTARTTIME[test] 8:00:00
# SRENDTIME[test]   17:00:00
# SRDAYS[test]      MON TUE WED THU FRI
# SRTASKCOUNT[test] 20
# SRMAXTIME[test]   0:30:00
# Creds: http://supercluster.org/mauidocs/6.1fairnessoverview.html
# USERCFG[DEFAULT]      FSTARGET=25.0
# USERCFG[john]         PRIORITY=100  FSTARGET=10.0-
# GROUPCFG[staff]       PRIORITY=1000 QLIST=hi:low QDEF=hi
# CLASSCFG[batch]       FLAGS=PREEMPTEE
# CLASSCFG[interactive] FLAGS=PREEMPTOR
RESCTLPOLICY ANY
```

2.2.2 Operation

Start of the Scheduler MAUI daemon:

```
[root@oscar /]# /usr/local/maui/sbin/maui
ps -ef | grep maui      : For check if is running.
```

Administration commands:
showq

```
[root@oscar /]# /usr/local/maui/bin/showq
ACTIVE JOBS--------------------
JOBNAME            USERNAME      STATE  PROC   REMAINING            STARTTIME
     0 Active Jobs       0 of    0 Processors Active (0.00%)
IDLE JOBS----------------------
JOBNAME            USERNAME      STATE  PROC     WCLIMIT            QUEUETIME
0 Idle Jobs
BLOCKED JOBS----------------
JOBNAME            USERNAME      STATE  PROC     WCLIMIT            QUEUETIME
Total Jobs: 0   Active Jobs: 0   Idle Jobs: 0   Blocked Jobs: 0
```

setres: Make a reservation of resources

```
user> setres -u nwmcjmm -s 13:40:00 -e 14:00:00 -n my-res TASKS==8
   reservation 'my-res.0' created on 2 nodes (8 tasks)
   comp008.nw-grid.ac.uk:1
   comp007.nw-grid.ac.uk:1
user> qsub -W x=FLAGS:ADVRES:my-res.0 test
   9.man2.nw-grid.ac.uk
user> checkjob 9
user> showq
Ejemplo1:
/usr/local/maui/bin/setres -u artichoke -s +14:02:23 -n ReservaCron1 TASKS==1
Ejemplo2:
/usr/local/maui/bin/setres -u artichoke -s 14:15:23 -n ReservaCron1 TASKS==1
```

2.3 License

Torque is released with the OpenPBS v2.3 Software License is a open source license but is not recognised either the FSF, or OSI as FLOSS software license.

2.4 Community

Torque / Maui doesn't has a big community, inherits mailing lists and documentation from the old PBS project, and ads its own users lists but there isn't a well formed community around Torque / Maui, there is a company behind the project called Adaptative Consulting (before called Cluster Resources) that mantains the project as Open Source but only for offers theirs added software Moab (a more advance scheduler than Maui) [8]

Chapter 3

Sun Grid Engine

A more advanced resource manager than Torque / Maui is Sun Grid Engine. Sun Grid Engine is a DRM (Distribute Resource Management), is a FLOSS distributed process execution system developed by Sun Microsystem , its function is manage, organize and finally execute user process in a grid of servers interconnected between them.

SGE is in charge of the aceptation, scheduling, send and manage of the distribution and remote execution of user process. For the correct realization of these tasks a administration and monitorization of all the resources that conform the grid is needed, checking their availability and taking decisions about what system or server member of the grid will be in charge of the execution of a task in each moment according to the availables resources in the grid.

3.1 Concepts

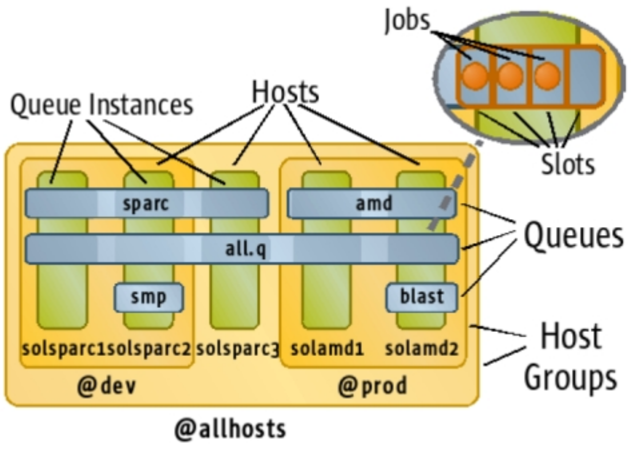
- **Grid**: Set of heterogeneous hosts spread for several facilities and that works like a one only server.
- **Cluster**: Set of homogeneous hosts installed in the same place and that works like a one only server.
- **Complex**: A complex is a resource of the cluster , of a host or of a queue in particular.
- **Cells**: Set of clusters, the cell is referenced by the variable SGE_CELL (only if we have more than a cluster), each cell has a directory registered by the variable SGE_CELL (for example: /opt/sungrid/sge6_2u5/default).

```
$SGE_CELL/
        /common
        /spool
```

  - **Common Zone**: /opt/sungrid/sge6_2u5/default/common: The common zone has information about the configuration of the cluster , accounting ,etc.., where are some files like act_qmaster that contains the hostname of the machine that acts as master host in a determined moment. All nodes use the information of this file for connecting

Figure 3.1: Grid Engine Logo [1]

Figure 3.2: Elements of a Grid [1]



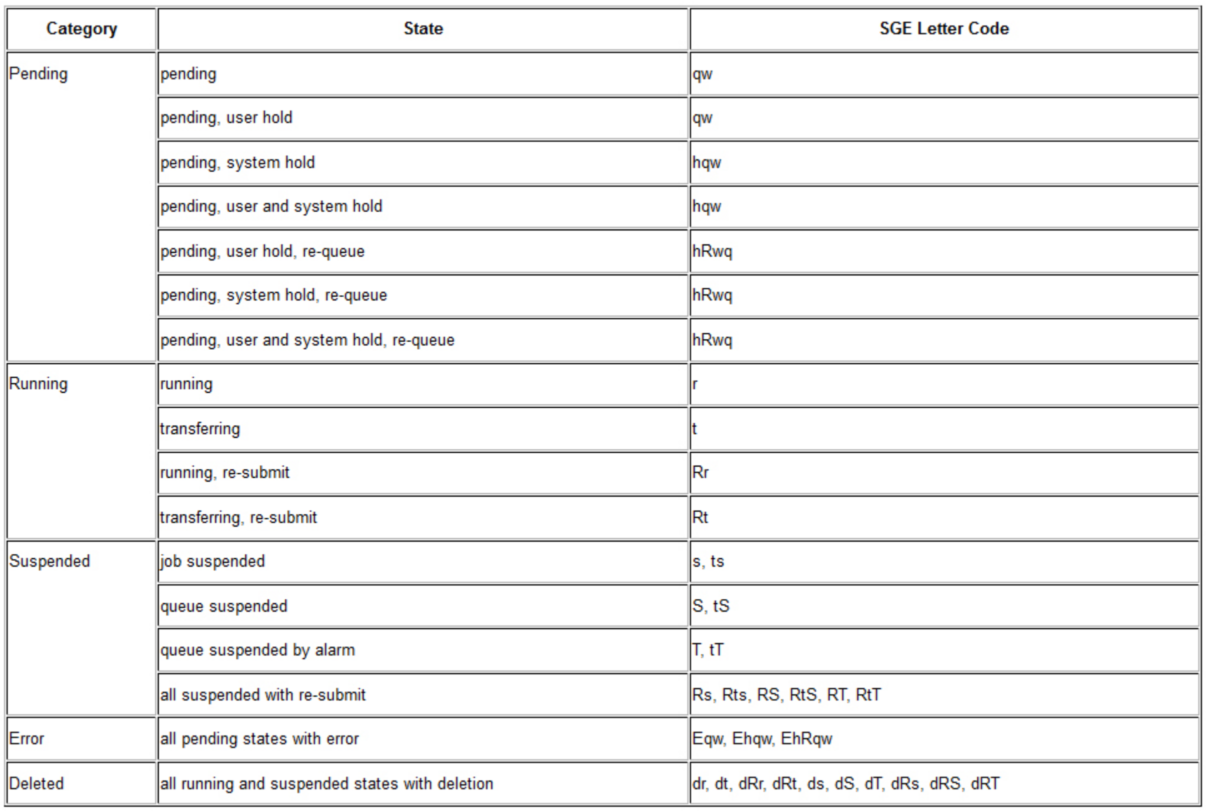
to the master host. In our configuration if the node that acts as a master host fails , there is another node ready to take control . For this, is necesary that previous execute the sge_qmaster daemon modify the act_qmaster writting his hostname. Only after that he could execute the sge_qmaster daemon. We have commented that all nodes of the cluster known who is the master host thanks to the file act_qmaster, if we make changes in this file , this changes must be visibles for all the nodes of the cluster inmediatly , for this reason we must stored the common directory in a shared disk and must be accesible for all the nodes of the cluster.

  - **Spool Zone**: /opt/sungrid/sge6_2u5/default/spool: The spool directory contains all the logs and contains all the data managed by the master node.

- **Queue**: By default all the system Sun Grid has a queue called all.q , that includes all the execution nodes . A queue can be present on one or more hosts. States of a queue:
  - **Suspend**: stop execution of all jobs running in the queue and close the queue
  - **Unsuspend**: resume execution in the queue and open the queue
  - **Disable**: close the queue, but do not affect running jobs
  - **Enable**: open the queue
- **Job**: Task or job segment.
- **Project**: Set of jobs with the same finality or that form a set of something.
- **Job slot**: Is the ability of a host to execute a job, in most cases , the number of job slots is determined by the number of cpu cores available in this host.
- **Job states**: A job can be in different states , see next section.

3.2 Job states definition

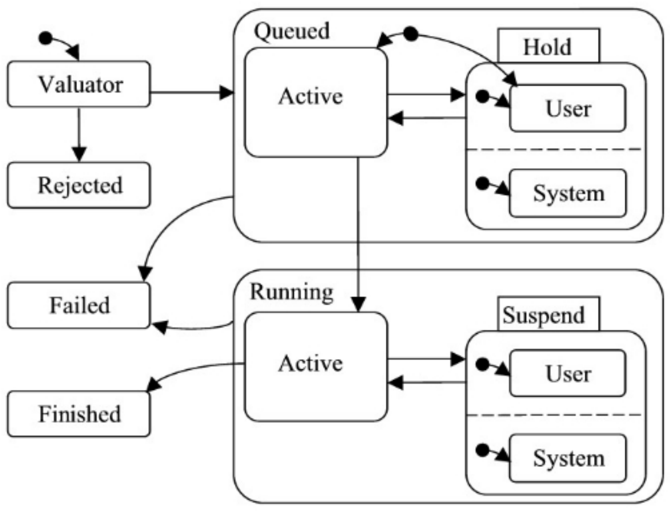
Figure 3.3: Job states definition [1]



Category	State	SGE Letter Code
Pending	pending	qw
	pending, user hold	qw
	pending, system hold	hqw
	pending, user and system hold	hqw
	pending, user hold, re-queue	hRwq
	pending, system hold, re-queue	hRwq
	pending, user and system hold, re-queue	hRwq
Running	running	r
	transferring	t
	running, re-submit	Rr
	transferring, re-submit	Rt
Suspended	job suspended	s, ts
	queue suspended	S, tS
	queue suspended by alarm	T, tT
	all suspended with re-submit	Rs, Rts, RS, RtS, RT, RtT
Error	all pending states with error	Eqw, Ehqw, EhRqw
Deleted	all running and suspended states with deletion	dr, dt, dRr, dRt, ds, dS, dT, dRs, dRS, dRT

3.3 Transitions between states

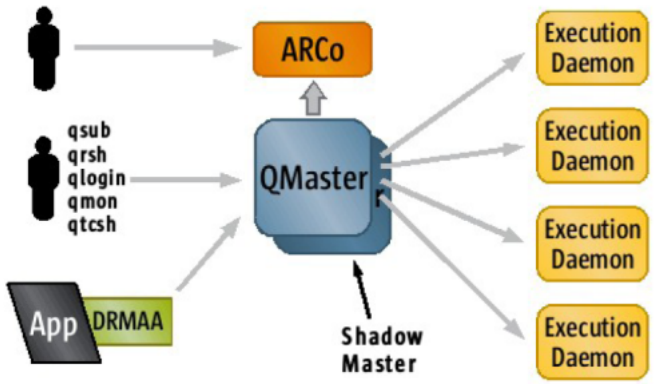
Figure 3.4: Transitions between states [1]



3.4 Server roles

- **Master Host** (Qmaster): Is the administration host, must be unique. The host that acts as a master node must be stable, doesn't be very loaded with other process and has at least 1 GB of free mem.
  - master daemon: sge_qmaster
  - Spool: /opt/sungrid/sge6_2u5/default/spool/qmaster
- **Shadow master host** : This host is a backup of the master host, and will acts as a master node if the principal is down. We can have some shadow hosts. The sge_shadowd is in charge of the monitorizing of the master host and share the state and the configuration of it, due this the directory sge-root/default/common must be accesible by all the shadow master hosts.
  - Daemon: sge_shadowd
  - Spool: /opt/sungrid/sge6_2u5/default/spool/qmaster
- **Execution host**: Host that can execute user process launched by the master host , we can have some of them , a execution host is also a administrative host.
  - Daemon: sge_execd
  - Spool: /opt/sungrid/sge6_2u5/default/spool/nombreHostEXE
- **Submit host**: Is a host that can submit process to the grid , by default master host is a submit host , we can have some of them.
- **Administration host**: Is a host that can do administrative tasks like configure queues, add users, by default a master host is an administration host also, we can have some of them.

Figure 3.5: Grid roles and components [1]



Between the servers that can be master host , only one could be , the rest remain waiting (they take the rol of shadow master) until detect the down of the master , in this moment the order in which the shadow hosts would take the rol of master host is defined in the file $SGE_CELL/common/shadow_masters

```
# cat /opt/sungrid/sge6_2u5/default/common/shadow_masters
 node1.domain.com
 node2.domain.com
 node3.domain.com
```

We can force the change of the rol of the master host between the shadows hosts (according to the order defined in the file $SGE_CELL/common/shadow_masters) with the next command:

```
/opt/sungrid/sge6_2u5/default/common/sgemaster -migrate
```

The above command must be executed in the shadow master host that we want take the role of master host.

3.5 Comunication between servers

There are two forms of comunication between servers:

- **Classic mode**: The spooling is on text files over a NFS systems (For small clusters)
- **BerkeleyDB**: The spooling is made on a Berkeley data base. This solution is for very big clusters of 100 or 200 nodes.

3.6 Complex: Resources of the cluster

We can see the resources (complex) of a cluster with the QMON tool or with the next command:

```
[jcl@test-sge1 lx24-amd64]$ ./qconf -sc
#name               shortcut    type        relop requestable consumable default  urgency
#----------------------------------------------------------------------------------------
arch                a           RESTRING    ==    YES         NO         NONE     0
calendar            c           RESTRING    ==    YES         NO         NONE     0
cpu                 cpu         DOUBLE      >=    YES         NO         0        0
display_win_gui     dwg         BOOL        ==    YES         NO         0        0
h_core              h_core      MEMORY      <=    YES         NO         0        0
h_cpu               h_cpu       TIME        <=    YES         NO         0:0:0    0
h_data              h_data      MEMORY      <=    YES         NO         0        0
```

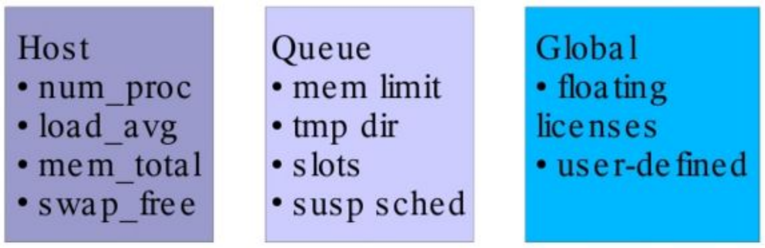
```
h_fsize            h_fsize     MEMORY      <=    YES         NO         0         0
h_rss              h_rss       MEMORY      <=    YES         NO         0         0
h_rt               h_rt        TIME        <=    YES         NO         0:0:0     0
h_stack            h_stack     MEMORY      <=    YES         NO         0         0
h_vmem             h_vmem      MEMORY      <=    YES         NO         0         0
hostname           h           HOST        ==    YES         NO         NONE      0
load_avg           la          DOUBLE      >=    NO          NO         0         0
load_long          ll          DOUBLE      >=    NO          NO         0         0
load_medium        lm          DOUBLE      >=    NO          NO         0         0
load_short         ls          DOUBLE      >=    NO          NO         0         0
m_core             core        INT         <=    YES         NO         0         0
m_socket           socket      INT         <=    YES         NO         0         0
m_topology         topo        RESTRING    ==    YES         NO         NONE      0
m_topology_inuse   utopo       RESTRING    ==    YES         NO         NONE      0
mem_free           mf          MEMORY      <=    YES         NO         0         0
```

- The sge_execd daemon sends periodically some load parameters (defined in en $SGE_ROOT /doc/load_parameters.asc) to the sge_master daemon. These values are stored in the sge_execd daemon itself and are showed in Qmon in the section : "Complex Configuration"
- The resources can be asigned, increased, decreased dynamically, every change assimilated in the momment by the SGE and the Scheduler.
- The administrator can define a consum by default of a determined resource. That value can see and configure since the window "Complex Configuration" in the Qmon in the column "Default" of each defined resource. From start the only one resource that has defined a default consum is SLOTS that has a value of 1 slots consumed for each process or job that is in execution.
- A script indicate the amount of a consumable resource that need with the argument "-l" and in base to this request and the available amount of that resource the process is executed or not , if a script doesn't determine what amount of a resource is needed , the Grid Scheduler takes the value of "Default consum" specified in the configuration of the resources in the section "Complex Configuration"
- The type of data that we can use for define the consum of a resource can be:
  - INT : Numerical
  - DOUBLE : Numerical
  - MEMORY : Numerical
  - TIME : Numerical
  - STRING
  - CSTRING
  - BOOL
  - RESTRING

3.6.1 Scope of the resources

- **Queue level**: example: h_vmem: maxim memory that a process can use.
- **Host level**
- **Cluster or global level**: example: disk space in a shared NFS.

Figure 3.6: Scope of the resources [1]



We can create resources defined by the user and assign to a queue, host or globally to all server , we must create since the screen “Complex Configuration” in the Qmon tool and then assign the resource create to a Queue, Host or all Cluster.

If we want to assign the resource to a queue , in qmon we must access to “Cluster Queue Control” select a queue, do click in “Modify” and in the tab “Complexes” add this resource and its value.

If we want to assign the resource to a host, in Qmon access to “Host Configuration” select tab “Execution Host” , select a host, do click in the windows “Consumables / Fixed Attributes” and then in “Modify” , and another time in “Consumables / Fixed Attributes” where we must indicate the new resource and its value.

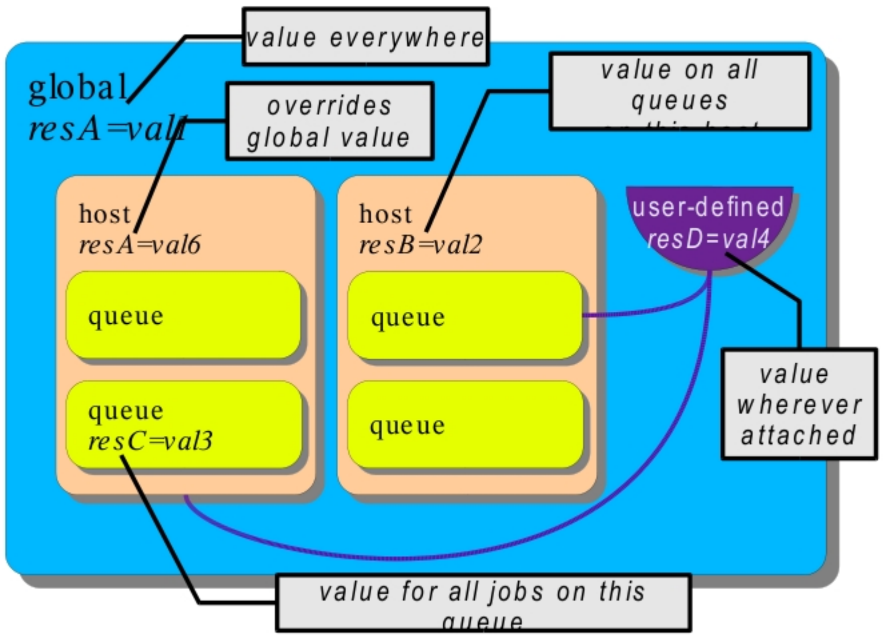
The process of checking the avalability of the resources is , firstable the resource cluster level is checked , after the host level , and lastly the queue level, Only if all the requirements are accomplished the process will be launched to the grid.

The cluster level resources are the requirements more restricted, for this reason if a requirement is not accomplished is not needed see the other levels of requirements. The cluster level resources are common between all the hosts of the same cluster and are shared between them.

This means:

- The cluster level resources are shared between all the system
- The queue level resources only exists for each host a determined number of resources.
- The host level resources only are applicable to this host.

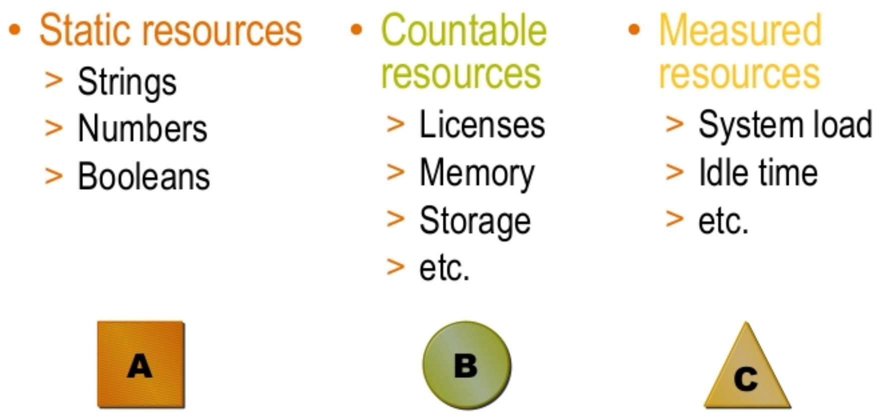
Figure 3.7: Scope of the resources II. [1]



3.6.2 Type of the resources

- **Static resources** : like installed programs, example: java
- **Dynamic resources**: resources that are modified with the time and are defined by the user. LoadSensor is a program included in SGE that measure a determined resource and update its state, example cpu load.
- **Consumable resources**: Are resources that can be locked by a job that used it, when the job is finished the resource is released , for this reasons is needed define the total amount avalaible of this resource in the cluster, host or queue.
  - We can see what resources are consumables in the column "Consumable" in the " Complex Configuration" window in the Qmon tool.
  - The consumbale resource are defined manually by the user forever, the only one defined by the system by default is SLOTS
  - A process consum resources similarly that a job consum "job slots" of a queue.
  - Only numerical attributes can be configured as consumables: INT, DOUBLE, MEMORY, or TIME.
  - The administrator can force a process to consum determined resources in a virtual manner for, for example, manage the memory of the server in a better way.
- **Predefined resources**: are dynamic resources predefined by SGE , and measure by the execution daemon itself, like cpu, memory.

Figure 3.8: Some types of resources [1]



3.6.3 Resource Quotas

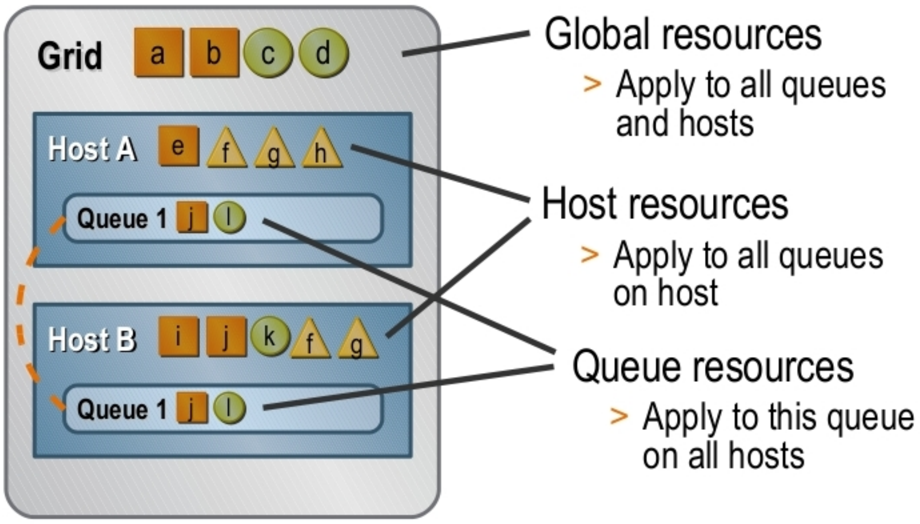
The “Resource Quotas” allow us apply limits of use of the systemś resources to the jobs or tasks . This limitation provide a indirect method of prioritize users, departments or projects.

The resources can be: slots, arch, mem_total, num_proc, swap_total, built-in resources, or any other resource defined by the user.

The resources consumer can be: users, queues, hosts, projects or “parallel enviroments”

For define the priority for wich a consumer should get a resource the “urgency” and “shared-based” policies are used.

Figure 3.9: Example of resource distribution [1]


Grid
a
b
c
d
Host A
e
f
g
h
Queue 1
j
l
Host B
i
j
k
f
g
Queue 1
j
l
Global resources
> Apply to all queues and hosts
Host resources
> Apply to all queues on host
Queue resources
> Apply to this queue on all hosts

3.6.4 Examples of the more used resources and its manage

- **virtual_free (vf)**: Determine the amount of virtual memory available , is a combination of swap space and physical memory available . In a ideal model the virtual memory used usually must be equal or lower than the physical memory installed for avoid the constant swapping that degrade the speed of the execution of the process.
  - If we want that SGE, manage correctly the system , virtual_free must be defined as consumable resource and its value must be equal or lower than the physical memory installed.
  - Take into account that if we work with SGE memory parameters very fit , we must don't launch process outside the system SGE (manually in the server, outside the Grid) because doesn't be controlled by the Grid and could down the server due the lack of memory.
  - Is usual have process with differents needs of memory and group them for be executed in differents queues according to their consum of memory. For this manner we assign a amount of memory to each queue:

    ```
    qsub -l vf=100M shell
    ```

  - Space Sharing: Consists in group jobs or tasks (for example in queues of process) according to consum of memory and assign quotas (a determined memory amount to each queue)
- **h_fsize**: (The total number of disk blocks that this job can create.) . Determine the maxim size that a file could has in execution time , at the same manner than the virtual memory is usual defined as consumable and group the process in different queues according the size of his files.
- **np_load_avg**: Cpu load normalized to the number of cpus , this value is used to measure the load of cpu of a host because is indepent of the number of cpus that a host can contain and then we can compare the cpu load between servers with differents process capacities.
- **load_avg**: Classic cpu load without normalize according the number of cpus of a server.

- **s_cpu** The per-process CPU time limit in seconds
- **s_core** The per-process maximum core file size in bytes.
- **s_data** The per-process maximum memory limit in bytes.
- **s_vmem** The same as s_data (if both are set the minimum is used).
- **h_cpu** The per-job CPU time limit in seconds.
- **h_data** The per-job maximum memory limit in bytes.
- **h_vmem** The same as h_data (if both are set the minimum is used).

3.7 Scheduler

The process scheduler is inside the qmaster daemon (MasterHost), the scheduler is executed by one of these reasons:

- Executed by determined intervals.
- Due to the queue of a new process or the finalization of one.
- Force with the command: qconf -tsm

Plan a process has two phases:

3.7.1 Phase-1: Job selection

According to the scheduler policies, a job receive more priority depending of: who has launched the process, how many resources need, when the job must be started , how many time has waiting to be executed.

The policies that generate the total priority of a process are:

- **Ticket Policies**: Tickets are distributed between jobs according the "Scheduler Policies" . The possesion of a tickets for a job represents its priority according a policy . There are three tickets policies:
  - Share Tree o Fair Share Policy
  - Functional ticket Policy
  - Overrride ticket policy
- **Urgency Policies**: Determine the urgency of a process
- **Custom Policies**: Modify policies of jobs just launched

3.7.2 Phase-2: Job scheduling

Is the assignment to a job of a set of free resources, a job is assigned to a job slot depending of the resources that the job has requested and the state of the queues and hosts with available job slots.

In the SGE installation by default a job slot is assigned to a job in a process of four successive steps and filters.

When a job is launched , a copy of the shell is made in /opt/sungrid/sge6_2u5/default/spool/qmaster/job_scripts but with a identificator number give by SGE in the queue moment by name.

```
[root@test-sge1 job_scripts]# pwd
/opt/sungrid/sge6_2u5/default/spool/qmaster/job_scripts
[root@test-sge1 job_scripts]# ll
total 8
-rw-r--r-- 1 root root 610 Apr 15 18:49 66
-rw-r--r-- 1 root root 610 Apr 15 18:49 68
```

SGE gives incremental identificators to the jobs starting the value contained in this file: /opt/sungrid/sge6_2u5/default/spool/qmaster/jobseqnum

3.7.3 Scheduler strategies

The SGE administrator can modify the following parameters that modify the normal scheduler strategies:

- Dynamic resource management : SGE controls and fits dynamically the resources that are assigned to the jobs.
- Queue sorting: SGE clasify the queues according to the order in wich they must be filled.
- Job sorting: Sort of jobs that determine the order in wich SGE attend to the jobs.
- Resource reservation and backfilling: Reserve resources for a job.

Default scheduling

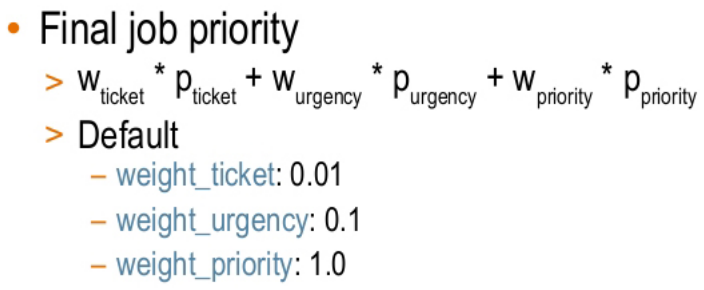
The default scheduling is a first-in, first-out policy. In other words, the first job that is submitted is the first job the scheduler examines to dispatch it to a queue. If the first job in the list of pending jobs finds a queue that is suitable and available, that job is started first. A job ranked behind the first job can be started first only if the first job fails to find a suitable free resource.

The default strategy is to select queue instances on the least-loaded host, provided that the queues deliver suitable service for the job's resource requirements. If several suitable queues share the same load, the queue to be selected is unpredictable.

Modify the default scheduling

- Changing the scheduling algorithm
- Scaling system load
- Selecting queue by sequence number
- Selecting queue by share
- Restricting the number of jobs per user or per group

Figure 3.10: Final job priority [1]



- Final job priority
  - > $w_{ticket} * p_{ticket} + w_{urgency} * p_{urgency} + w_{priority} * p_{priority}$
  - > Default
    - – weight_ticket: 0.01
    - – weight_urgency: 0.1
    - – weight_priority: 1.0

3.8 Policies

Without any intervention of the SGE adminsitrator the default execution policy is FIFO , the first job that enter is the first that is executed.

Priority of a process:

```
Priority of a process=(Urgency*Normalized Urgency Value)+(Ticket*Normalized
Ticket Value)+(Priority*Normalized Priority Value)
```

3.8.1 Tickets Policies

The policies based on tickets use a distribution of them for determine the priority of a job , from start the system count with a determined number of tickets that distribute between the process according the next policies.

Note: The total number of tickets doesn't has a special meaning , the real important are the relationship between the policies , though, a high number of tickets allow us establish the relations with more accuracy.

Shared Tree Policy

The policies based in quotas are implemented using a hierarchical tree of quotas that specify for a determined period of time , how the system's resources are distributed between the different levels of the tree whose nodes can be users, departments or projects.

The quotas (share) of a resource of a node tree depends of the available quotas of his father's nodes and the nodes of the same level.

Concepts:

- Half-Life: Temporary penalty against a user due to the excesive use of resources . Indicate how many time of the resources use SGE is going to take into account for assign more to a user or penalty for this.
- Compensation Factor (CF): (Between 2 and 10 is a normal value) Allow us limit the use of resources , prevent that a user or a project can monopolize totally a resource just because be the first one in use it. Example, a compensation factor of two limit the current resource quota of a project or of a user to the double of its quota initially established. The CF allow a administrator limit the massive use in short term.

Figure 3.11: Example I of Share Tree [1]

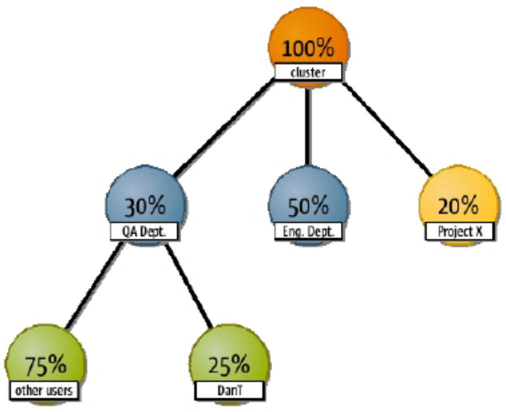
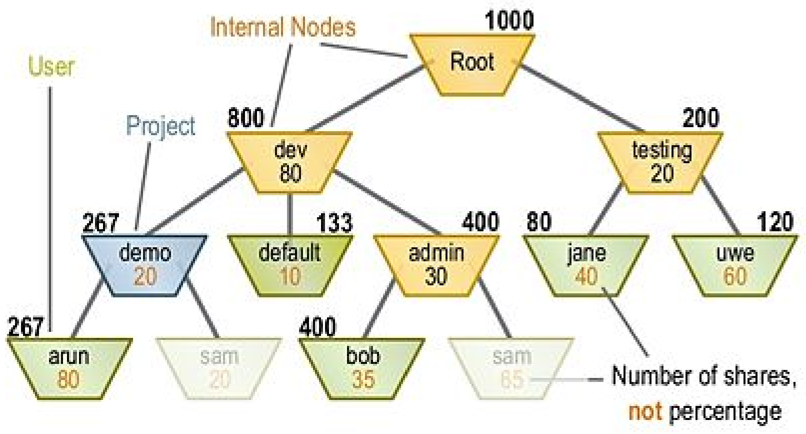


Figure 3.12: Example II of Share Tree [1]



- Leaf Nodes: Must be users or projects .
- This policy guarantee that there aren't underutilized resources and there aren't resource quotas that remain without use.
- A node can be a user, a project or a department
- A project can't be referenced more than once in the tree.
- A user only can appear once in the subtree of a project.
- A user only can appear once outside the subtree of a project.
- A user only can appear as a Leaf Node.
- All the lead nodes of a project must reference to a known user or to the generic user "Default"
- The projects can't has subprojects.
- All the users (leaf nodes) of a project (subtree) has access to this project.

- Default User: We can use in the nodes than identify users a Default user with the purpose of simplify the maintenance of the tree when we have a lot of users.

  example:

```
ROOT
 |
 ------ Project A
 |          |
 |          --------- Default
 ------ Project B
          |
          |---------- Default (shares=20)
          |---------- User A (shares=10)
          ----------- User B (shares=40)
```

  In the above example all the users of the Project A has the same quota (share) of use of the resources , in the Project B all has 20 quotas except the user A and B that has 10 and 40 respectively

- Available resource quotas: Exists three types of available resource quotas: Cpu, Memory and I/O and the sum of them result the 100% of the avaliable resources of the system.

Functional Ticket Policy

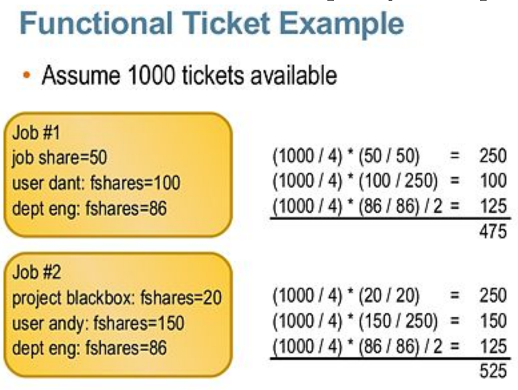
Functional policies allow us asociate a job with a user, a department or a project , a set of quotas or privileges are assigned to these jobs. In this type of policies is garantized in every moment the 100% of use of all the resources , because the quotas of non used resources are distribuited between the users, departments, or projects that need resources in each moment.

During the installation the administrator split the total number of Function Tickets between the Functional Categories of users , departments or projects.

The Functional Tickets of a category can be distributed between all the jobs of this category or not, depending on the selection or not of the "Share Override Tickets" option.

Functional Ticket Policy is like Share Tree policy of two levels with the difference that a job can be associated with some categories at the same time . A job can belongs to a determined user , but also can belongs to a project or to a department.

Figure 3.13: Functional ticket policy example I [1]

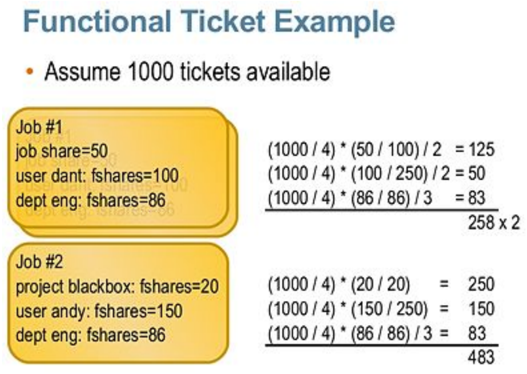


Functional Ticket Example

- Assume 1000 tickets available

Job #1 job share=50 user dant: fshares=100 dept eng: fshares=86	(1000 / 4) * (50 / 50) = 250 (1000 / 4) * (100 / 250) = 100 (1000 / 4) * (86 / 86) / 2 = 125 475
Job #2 project blackbox: fshares=20 user andy: fshares=150 dept eng: fshares=86	(1000 / 4) * (20 / 20) = 250 (1000 / 4) * (150 / 250) = 150 (1000 / 4) * (86 / 86) / 2 = 125 525

Figure 3.14: Functional ticket policy example II [1]



Functional Ticket Example

- Assume 1000 tickets available

Job #1
job share=50
user dant: fshares=100
dept eng: fshares=86

```
(1000 / 4) * (50 / 100) / 2  = 125
(1000 / 4) * (100 / 250) / 2 = 50
(1000 / 4) * (86 / 86) / 3   = 83
                               258 x 2
```

Job #2
project blackbox: fshares=20
user andy: fshares=150
dept eng: fshares=86

```
(1000 / 4) * (20 / 20)      =  250
(1000 / 4) * (150 / 250)    =  150
(1000 / 4) * (86 / 86) / 3  =  83
                               483
```

Override Ticket Policy

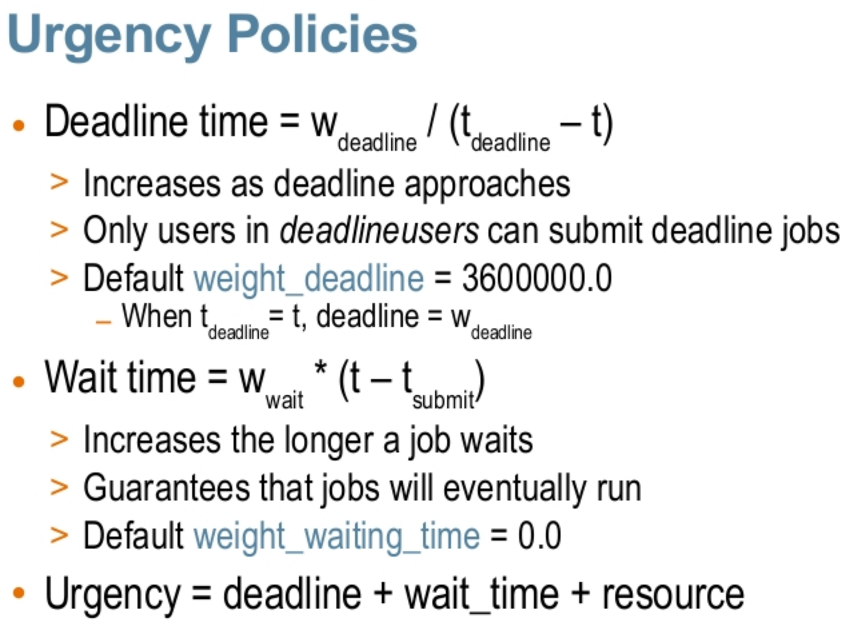
The Override Ticket Policy allows to the system administrator dynamically adjust the number of tickets of a job or some jobs that belongs to a user, project or department , these tickets are called “override tickets” , these tickets allow us modify shared-based and functional policies for a very determined moments without have to change the configuration of those policies.

3.8.2 Urgency Policies

The Urgency Policies determine the urgency of a process, this means that this policies mark the need of increase the priority of a job in base of three main aspects:

- **Wait Time Policy**: Determine that the more time a process be waiting the more priority has it.
- **DeadLine Policy**: Determine that the more closer will be a process of his deadline the more priority will has.
- **Resource Urgency Policy**: Determine that the more urgent will be the use of a resource by a job according to the definition in “Complex Configuration” the more priority must has this job.

Figure 3.15: Urgency Policies [1]



Urgency Policies

- Deadline time = $w_{deadline}$ / ($t_{deadline}$ – t)
  - > Increases as deadline approaches
  - > Only users in *deadlineusers* can submit deadline jobs
  - > Default weight_deadline = 3600000.0
    - – When $t_{deadline}$ = t, deadline = $w_{deadline}$
- Wait time = w_{wait} * (t – t_{submit})
  - > Increases the longer a job waits
  - > Guarantees that jobs will eventually run
  - > Default weight_waiting_time = 0.0
- Urgency = deadline + wait_time + resource

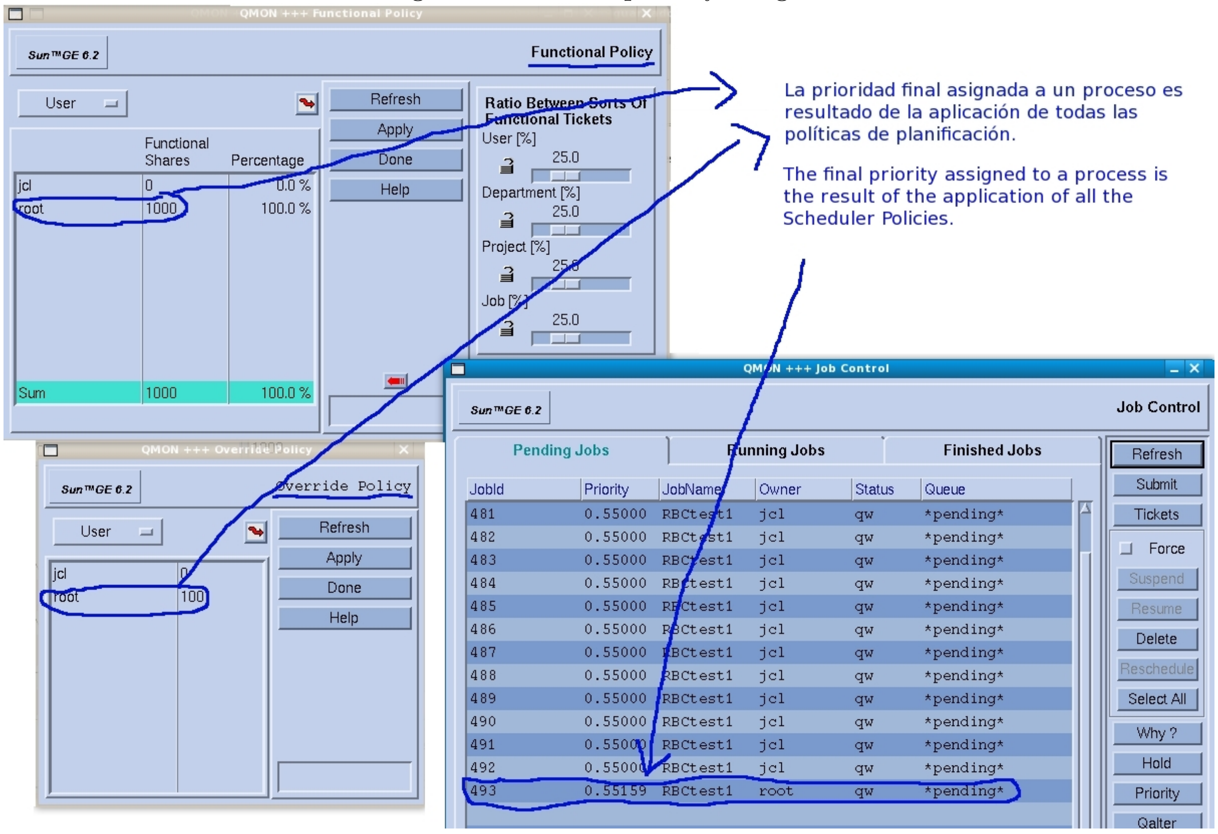
3.8.3 Custom Policy o POSIX Policy: Modify policies of jobs just launched

This policy allow to the administrator asign a priority directly to a job wich value can be in the same range that the used by the POSIX standard : -1023 (less priority) to 1024 (more priority)

This priority assignment has a very important impact in the total priority resultant of apply all the policies, this policy is used for assign more or less priority to a job that has already launched to the SGE system

Only the administrator can assign priorities greater than 0 to a job.

Figure 3.16: Final priority assigned



	Functional Shares	Percentage
jcl	0	0.0 %
root	1000	100.0 %
Sum	1000	100.0 %

jcl	0
root	100

JobId	Priority	JobName	Owner	Status	Queue
481	0.55000	RBCtest1	jcl	qw	*pending*
482	0.55000	RBCtest1	jcl	qw	*pending*
483	0.55000	RBCtest1	jcl	qw	*pending*
484	0.55000	RBCtest1	jcl	qw	*pending*
485	0.55000	RBCtest1	jcl	qw	*pending*
486	0.55000	RBCtest1	jcl	qw	*pending*
487	0.55000	RBCtest1	jcl	qw	*pending*
488	0.55000	RBCtest1	jcl	qw	*pending*
489	0.55000	RBCtest1	jcl	qw	*pending*
490	0.55000	RBCtest1	jcl	qw	*pending*
491	0.55000	RBCtest1	jcl	qw	*pending*
492	0.55000	RBCtest1	jcl	qw	*pending*
493	0.55159	RBCtest1	root	qw	*pending*

La prioridad final asignada a un proceso es resultado de la aplicación de todas las políticas de planificación.

The final priority assigned to a process is the result of the application of all the Scheduler Policies.

3.9 Resource Reservations

The scheduler will reserve and adquire as soon as the resources needed by the highest priority job be free. But while Backfilling is allowed, Backfilling is allow use this reserved resources to other process if they finish soon.

Despite a job has the highest priority , if is impossible to satisfy its requirements it never will be executed : In this case the reserve of resource is needed.

RQS: Resource Quota Sets

The target is limit the use of resources with access to resource quotas

We can add rules like this:

```
limits users @QA projects ProjectX hosts (@production) to slots=1
```

Figure 3.17: Resource reservation example 1 [1]

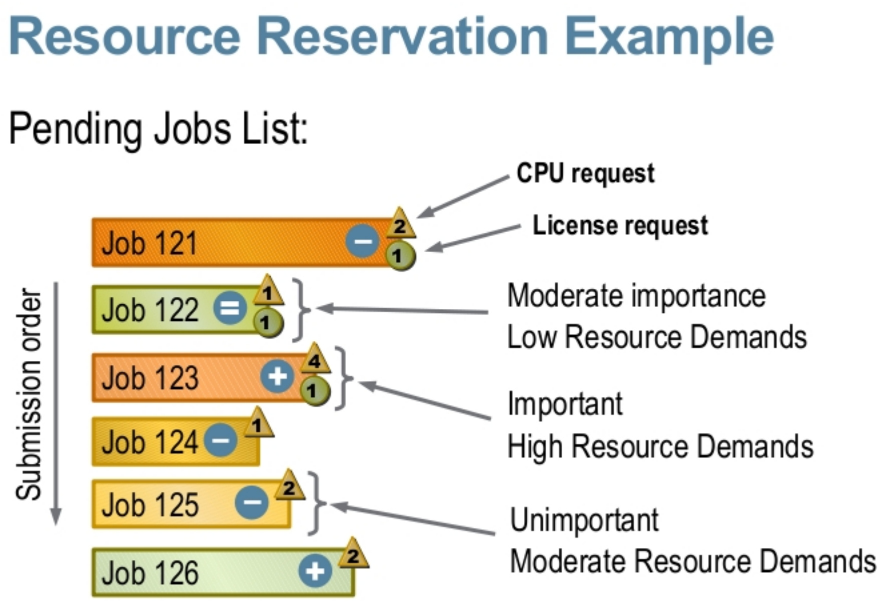


Figure 3.18: Resource reservation example 2 [1]

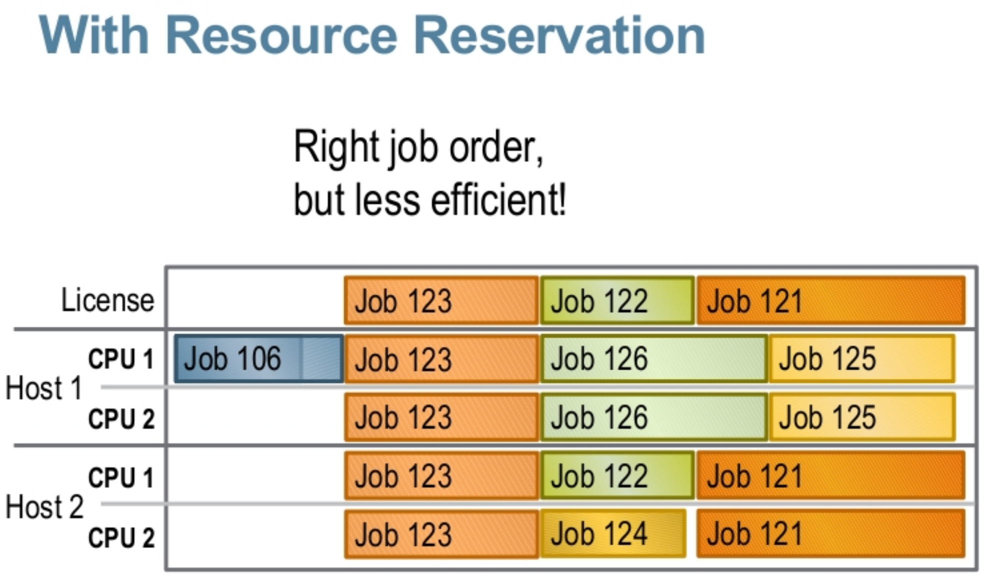


Figure 3.19: Resource reservation example 3 [1]

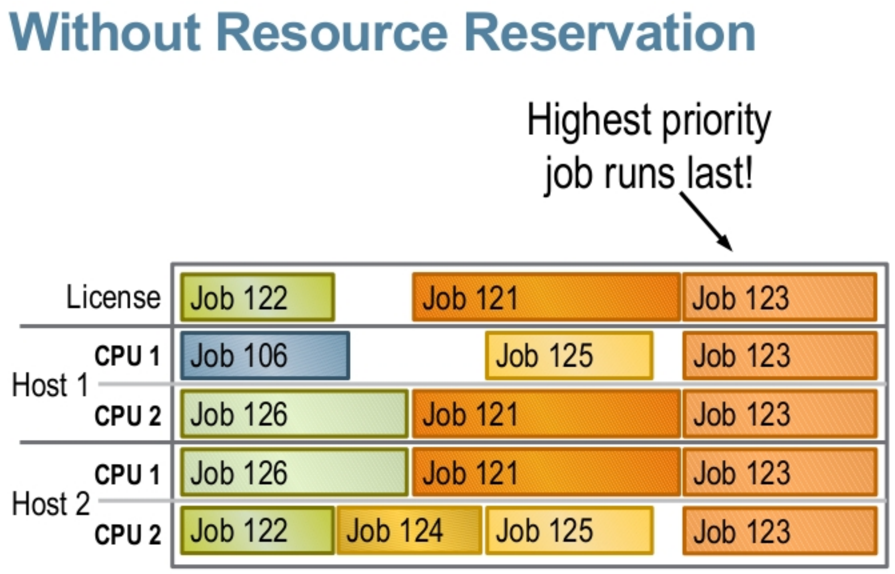
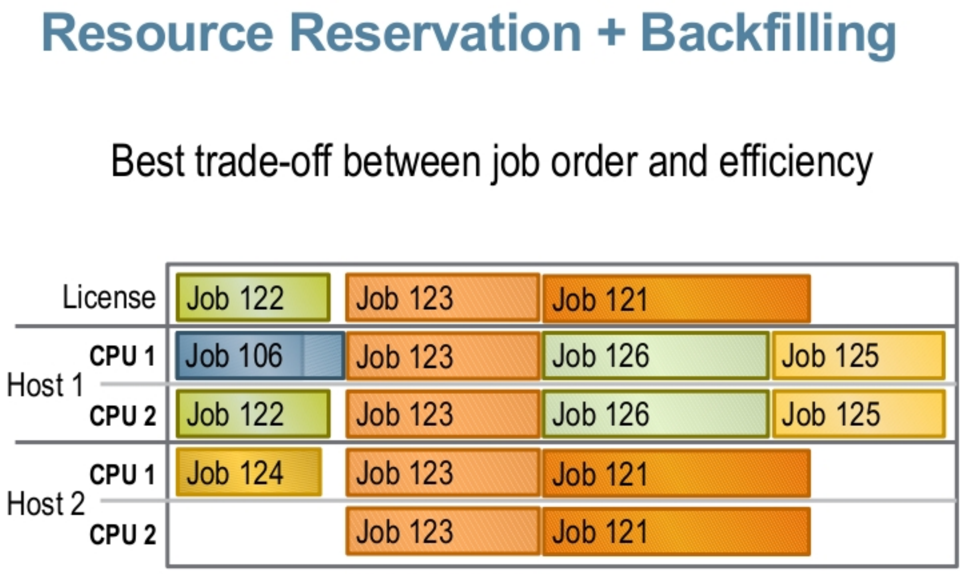


Figure 3.20: Resource reservation example 4 [1]



3.9.1 Advance Reservations: Reserve resources for a determined time

The advance reservations allow a user to reserve determined resources for a determined interval of time , the users can request the execution job in a “Advance Reservation” determined.

3.10 Exclusive Host Access

A host can be used exclusively to the execution of a job.

3.11 JSV: Job Submission Verifiers: Filter the launching of a job

Is a filter process defined by the administrator of the Grid by wich can accept, refuse or modify job launch requests.

3.12 Installation of a Master Host

For a correct installation is needed erase in the file /etc/hosts the line that define the IP 127.0.0.1 the name of the server and left only the definition of localhost

```
127.0.0.1          moscar0 localhost.localdomain localhost
192.168.1.98       moscar0   ---> ERASE IT!
```

This is the final correct /etc/hosts:

```
# Do not remove the following line, or various programs
# that require network functionality will fail.
127.0.0.1       localhost.localdomain localhost
::1             localhost6.localdomain6 localhost6
```

The qmaster's data spool can be based on flat files (known as classic spooling because it was once the only option), a local Berkeley Database data store, or a remote Berkeley Database server. The computing environment, the size of the cluster, and the expected cluster throughput are key factors used in determining which spooling option is best.

Make the data spool available via a shared file system, usually NFS. With local Berkeley spooling, this approach presents a problem as the Berkeley Database software does not work with NFSv2 or NFSv3, and NFSv4 is not yet widely adopted.

If shadow masters will not be used in the cluster, the qmaster's data spool need not be shared.

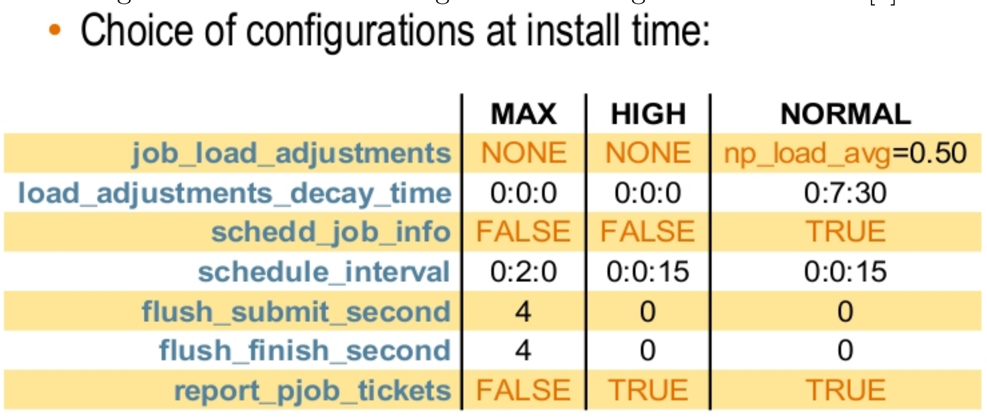
The choice of data spooling options also affects where the data spool is located.For classic spooling, the data spool is located in the qmaster's spool directory,which is located at $SGE_ROOT/$SGE_CELL/spool/qmaster by default. As explained above, however, a local Berkeley Database data store often cannot be located on an NFS-shared file system. As mentioned in the previous section the cell directory is usually NFS-shared. To resolve this conflict, the qmaster allows the local Berkeley data store to be located in a directory that is different from the qmaster's spool directory.

/opt/sungrid/sge6_2u5/default/spool /opt/sungrid/sge6_2u5/default/spooldb

```
    ./install_qmaster -jmx
sge_qmaster Please enter an unused port number >> 6444
sge_execd   Please enter an unused port number >> 6445
Cell Name : Default
Cluster Name: <<< cluster >>>

Setup spooling
--------------
Your SGE binaries are compiled to link the spooling libraries
during runtime (dynamically). So you can choose between Berkeley DB
spooling and Classic spooling method.
Please choose a spooling method (berkeleydb|classic) [berkeleydb] >> <<< classic >>>

Grid Engine JMX MBean server
----------------------------

In order to use the SGE Inspect or the Service Domain Manager (SDM)
SGE adapter you need to configure a JMX server in qmaster. Qmaster
will then load a Java Virtual Machine through a shared library.
```

```
NOTE: Java 1.5 or later is required for the JMX MBean server.

Do you want to enable the JMX MBean server (y/n) [y] >>  <<< y >>>

Please give some basic parameters for JMX MBean server
We will ask for
   - JAVA_HOME
   - additional JVM arguments (optional)
   - JMX MBean server port
   - JMX ssl authentication
   - JMX ssl client authentication
   - JMX ssl server keystore path
   - JMX ssl server keystore password

Detecting suitable JAVA ...

Enter JAVA_HOME (use "none" when none available)Please enter additional JVM arguments (optional, default is [-Xmx256m]) >>
Please enter an unused port number for the JMX MBean server [6446] >>
Enable JMX SSL server authentication (y/n) [y] >>   <<< y >>>
Enter JMX SSL server keystore path [/var/sgeCA/port6444/default/private/keystore] >>
Enter JMX SSL server keystore pw (at least 6 characters) >>
Using the following JMX MBean server settings.
   libjvm_path             >/usr/lib/jvm/java-1.6.0-openjdk-1.6.0.0.x86_64/jre/lib/amd64/server/libjvm.so<
   Additional JVM arguments >-Xmx256m<
   JMX port                >6446<
   JMX ssl                 >true<
   JMX client ssl          >true<
   JMX server keystore     >/var/sgeCA/port6444/default/private/keystore<
   JMX server keystore pw  >********<

Do you want to use these data (y/n) [y] >>

Setup spooling
--------------
Your SGE binaries are compiled to link the spooling libraries
during runtime (dynamically). So you can choose between Berkeley DB
spooling and Classic spooling method.
Please choose a spooling method (berkeleydb|classic) [berkeleydb] >> classic

Initializing Certificate Authority (CA) for OpenSSL security framework
-----------------------------------------------------------------------
There are already directories of the CA infrastructure in
   /opt/sungrid/sge6_2u5/default/common/sgeCA
   or
   /var/sgeCA/port6444/default

Do you want to recreate your SGE CA infrastructure (y/n) [y] >>   <<< y >>>
Grid Engine group id range
--------------------------
    You can change at any time the group id range in your cluster configuration.
    Please enter a range [20000-20100] >>  <ENTER>

Grid Engine cluster configuration
---------------------------------
    Please give the basic configuration parameters of your Grid Engine
    installation:
    <execd_spool_dir>
    The pathname of the spool directory of the execution hosts. User >root<
    must have the right to create this directory and to write into it.
    Default: [/opt/sungrid/sge6_2u5/default/spool] >>

Creating CA certificate and private key
---------------------------------------
 Please give some basic parameters to create the distinguished name (DN)
 for the certificates.
 We will ask for
   - the two letter country code
   - the state
   - the location, e.g city or your buildingcode
   - the organization (e.g. your company name)
   - the organizational unit, e.g. your department
   - the email address of the CA administrator (you!)
     Please enter your two letter country code, e.g. 'US' >> SP
     Please enter your state >> MADRID
     Please enter your location, e.g city or buildingcode >> 28010
     Please enter the name of your organization >> company
     Please enter your organizational unit, e.g. your department >> sistemas
     Please enter the email address of the CA administrator >>
     mymail@company.com

   You selected the following basic data for the distinguished name of
   your certificates:
   Country code:         C=SP
   State:                ST=MADRID
   Location:             L=28010
   Organization:         O=company
   Organizational unit:  OU=sistemas
   CA email address:     emailAddress=mymail@company.com

  Do you want to use these data (y/n) [y] >>
  Creating CA certificate and private key
  Generating a 1024 bit RSA private key
  .........++++++
  ........++++++
  writing new private key to '/var/sgeCA/port6444/default/private/cakey.pem'
  -----
 Hit <RETURN> to continue >>   RETURN

qmaster startup script
---------------------
 We can install the startup script that will
 start qmaster at machine boot (y/n) [y] >>  y
 cp /opt/sungrid/sge6_2u5/default/common/sgemaster /etc/init.d/sgemaster-cluster
 /usr/lib/lsb/install_initd /etc/init.d/sgemaster-cluster

Grid Engine qmaster startup
---------------------------
  Starting qmaster daemon. Please wait ...
     starting sge_qmaster
  Hit <RETURN> to continue >>   RETURN


Adding Grid Engine hosts
------------------------
 Please now add the list of hosts, where you will later install your execution
 daemons. These hosts will be also added as valid submit hosts.
 Please enter a blank separated list of your execution hosts. You may
 press <RETURN> if the line is getting too long. Once you are finished
 simply press <RETURN> without entering a name.
 You also may prepare a file with the hostnames of the machines where you plan
 to install Grid Engine. This may be convenient if you are installing Grid
 Engine on many hosts.
 Do you want to use a file which contains the list of hosts (y/n) [n] >> n

Adding admin and submit hosts
-----------------------------
 Please enter a blank seperated list of hosts.
 Stop by entering <RETURN>. You may repeat this step until you are
 entering an empty list. You will see messages from Grid Engine
 when the hosts are added.
 Host(s): test-sge0 test-sge1

      adminhost "test-sge0" already exists
      test-sge0 added to submit host list
      test-sge1 added to administrative host list
      test-sge1 added to submit host list
       Hit <RETURN> to continue >>

Adding admin and submit hosts
-----------------------------

  Please enter a blank seperated list of hosts.
  Stop by entering <RETURN>. You may repeat this step until you are
  entering an empty list. You will see messages from Grid Engine
  when the hosts are added.
  Host(s): RETURN
--
  If you want to use a shadow host, it is recommended to add this host
  to the list of administrative hosts.
  If you are not sure, it is also possible to add or remove hosts after the
  installation with <qconf -ah hostname> for adding and <qconf -dh hostname>
  for removing this host

  Attention: This is not the shadow host installation
  procedure.
  You still have to install the shadow host separately
  Do you want to add your shadow host(s) now? (y/n) [y] >>   <<< y >>>

Adding Grid Engine shadow hosts
-------------------------------
  Please now add the list of hosts, where you will later install your shadow
  daemon.
  Please enter a blank separated list of your execution hosts. You may
  press <RETURN> if the line is getting too long. Once you are finished
  simply press <RETURN> without entering a name.

  You also may prepare a file with the hostnames of the machines where you plan
  to install Grid Engine. This may be convenient if you are installing Grid
  Engine on many hosts.

  Do you want to use a file which contains the list of hosts (y/n) [n] >> n

Adding admin hosts
------------------
  Please enter a blank seperated list of hosts.
  Stop by entering <RETURN>. You may repeat this step until you are
  entering an empty list. You will see messages from Grid Engine
  when the hosts are added.
  Host(s): test-sge1

       adminhost "test-sge1" already exists
       Hit <RETURN> to continue >>

Creating the default <all.q> queue and <allhosts> hostgroup
-----------------------------------------------------------
     root@test-sge0 added "@allhosts" to host group list
     root@test-sge0 added "all.q" to cluster queue list
     Hit <RETURN> to continue >>    RETURN

Choosing password for the administrative user of SGE daemons
```

Figure 3.21: Different configurations during the installation [1]



- Choice of configurations at install time:

	MAX	HIGH	NORMAL
job_load_adjustments	NONE	NONE	np_load_avg=0.50
load_adjustments_decay_time	0:0:0	0:0:0	0:7:30
schedd_job_info	FALSE	FALSE	TRUE
schedule_interval	0:2:0	0:0:15	0:0:15
flush_submit_second	4	0	0
flush_finish_second	4	0	0
report_pjob_tickets	FALSE	TRUE	TRUE

```
------------------------------------------------------------
    Enter pw for default's keystore (at least 6 characters) >> <PASSWORD>

Scheduler Tuning
----------------
    The details on the different options are described in the manual.


Configurations
--------------
   1) Normal
          Fixed interval scheduling, report limited scheduling information,
          actual + assumed load
   2) High
          Fixed interval scheduling, report limited scheduling information,
          actual load
   3) Max
          Immediate Scheduling, report no scheduling information,
          actual load
    Enter the number of your preferred configuration and hit <RETURN>!
    Default configuration is [1] >> 1



Using Grid Engine
-----------------
   You should now enter the command:
       source /opt/sungrid/sge6_2u5/default/common/settings.csh
   if you are a csh/tcsh user or
       # . /opt/sungrid/sge6_2u5/default/common/settings.sh
   if you are a sh/ksh user.

   This will set or expand the following environment variables:

   - $SGE_ROOT         (always necessary)
   - $SGE_CELL         (if you are using a cell other than >default<)
   - $SGE_CLUSTER_NAME (always necessary)
   - $SGE_QMASTER_PORT (if you haven't added the service >sge_qmaster<)
   - $SGE_EXECD_PORT   (if you haven't added the service >sge_execd<)
   - $PATH/$path       (to find the Grid Engine binaries)
   - $MANPATH          (to access the manual pages)

    Hit <RETURN> to see where Grid Engine logs messages >>   RETURN

Grid Engine messages
--------------------
    Grid Engine messages can be found at:
       /tmp/qmaster_messages (during qmaster startup)
       /tmp/execd_messages   (during execution daemon startup)
   After startup the daemons log their messages in their spool directories.
   Qmaster:     /opt/sungrid/sge6_2u5/default/spool/qmaster/messages
   Exec daemon: <execd_spool_dir>/<hostname>/messages


Grid Engine startup scripts
---------------------------
   Grid Engine startup scripts can be found at:
      /opt/sungrid/sge6_2u5/default/common/sgemaster (qmaster)
      /opt/sungrid/sge6_2u5/default/common/sgeexecd (execd)
   Do you want to see previous screen about using Grid Engine again (y/n) [n] >>

Your Grid Engine qmaster installation is now completed
------------------------------------------------------
   Please now login to all hosts where you want to run an execution daemon
   and start the execution host installation procedure.
   If you want to run an execution daemon on this host, please do not forget
   to make the execution host installation in this host as well.
   All execution hosts must be administrative hosts during the installation.
   All hosts which you added to the list of administrative hosts during this
   installation procedure can now be installed.
   You may verify your administrative hosts with the command

   # qconf -sh
   and you may add new administrative hosts with the command
   # qconf -ah <hostname>

    Please hit <RETURN> >>
```

For load the environment variables:

```
. /opt/grid/sge6_2u5/default/common/settings.sh
```

3.13 Installation of a Execution Host

- Previous requeriment: A master server with qmaster and qsched process is needed
- We must indicate in wich Cluster will belong this execution host, in this case Default cluster
- By default all Sun Grid System has a default queue called all.q that includes all the execution hosts.
- Each execution daemon has its own spool directory. The default location for the spool directory offered by the installation process is *SGE_ROOT*/SGE_CELL/spool/hostname

The execution daemon's spool directory contains information about the jobs currently being executed by that execution daemon, so that in the event of a failure, the execution daemon can try to reconstruct its previous state when it restarts. Among the information stored in the execution daemon's spool directory is a local copy of every job script being executed by that execution daemon. When a user submits a binary job, only the path to the binary is passed on to the execution daemon. When a user submits a script job, however, the script file is copied and sent along with the job. When the execution daemon executes the job, it runs a local copy of the script saved into its spool directory, rather than the original script file submitted by the user.

```
. /opt/sungrid/sge6_2u5/default/common/settings.sh
./install_execd

Welcome to the Grid Engine execution host installation
------------------------------------------------------
If you haven't installed the Grid Engine qmaster host yet, you must execute
this step (with >install_qmaster<) prior the execution host installation.
```

```
For a sucessfull installation you need a running Grid Engine qmaster. It is
also neccesary that this host is an administrative host.

You can verify your current list of administrative hosts with
the command:
   # qconf -sh
You can add an administrative host with the command:
   # qconf -ah <hostname>
The execution host installation will take approximately 5 minutes.
Hit <RETURN> to continue >>   RETURN


Checking $SGE_ROOT directory
----------------------------
The Grid Engine root directory is:
   $SGE_ROOT = /opt/sungrid/sge6_2u5
If this directory is not correct (e.g. it may contain an automounter
prefix) enter the correct path to this directory or hit <RETURN>
to use default [/opt/sungrid/sge6_2u5] >>   RETURN

Grid Engine cells
-----------------
Please enter cell name which you used for the qmaster
installation or press <RETURN> to use [default] >>  RETURN

Grid Engine TCP/IP communication service
----------------------------------------
The port for sge_execd is currently set by the shell environment.
   SGE_EXECD_PORT = 6445
Hit <RETURN> to continue >>  RETURN

Execd spool directory configuration
-----------------------------------
You defined a global spool directory when you installed the master host.
You can use that directory for spooling jobs from this execution host
or you can define a different spool directory for this execution host.

ATTENTION: For most operating systems, the spool directory does not have to
be located on a local disk. The spool directory can be located on a
network-accessible drive. However, using a local spool directory provides
better performance.

FOR WINDOWS USERS: On Windows systems, the spool directory MUST be located
on a local disk. If you install an execution daemon on a Windows system
without a local spool directory, the execution host is unusable.

The spool directory is currently set to:
<</opt/sungrid/sge6_2u5/default/spool/test-sge1>>

Do you want to configure a different spool directory
for this host (y/n) [n] >> n

Creating local configuration
----------------------------
root@test-sge1 added "test-sge1" to configuration list
Local configuration for host >test-sge1< created.

Hit <RETURN> to continue >>   RETURN

execd startup script
--------------------
We can install the startup script that will
start execd at machine boot (y/n) [y] >>
cp /opt/sungrid/sge6_2u5/default/common/sgeexecd /etc/init.d/sgeexecd.-cluster
/usr/lib/lsb/install_initd /etc/init.d/sgeexecd.-cluster

Hit <RETURN> to continue >>   RETURN

Adding a queue for this host
----------------------------
We can now add a queue instance for this host:
   - it is added to the >allhosts< hostgroup
   - the queue provides 1 slot(s) for jobs in all queues
     referencing the >allhosts< hostgroup
You do not need to add this host now, but before running jobs on this host
it must be added to at least one queue.
Do you want to add a default queue instance for this host (y/n) [y] >>

root@test-sge1 modified "@allhosts" in host group list
root@test-sge1 modified "all.q" in cluster queue list

Using Grid Engine
-----------------
You should now enter the command:
   source /opt/sungrid/sge6_2u5/default/common/settings.csh
if you are a csh/tcsh user or
   # . /opt/sungrid/sge6_2u5/default/common/settings.sh
if you are a sh/ksh user.

This will set or expand the following environment variables:
   - $SGE_ROOT         (always necessary)
   - $SGE_CELL         (if you are using a cell other than >default<)
   - $SGE_CLUSTER_NAME (always necessary)
   - $SGE_QMASTER_PORT (if you haven't added the service >sge_qmaster<)
   - $SGE_EXECD_PORT   (if you haven't added the service >sge_execd<)
   - $PATH/$path       (to find the Grid Engine binaries)
   - $MANPATH          (to access the manual pages)

Hit <RETURN> to see where Grid Engine logs messages >>
Grid Engine messages
--------------------
Grid Engine messages can be found at:
   /tmp/qmaster_messages (during qmaster startup)
   /tmp/execd_messages   (during execution daemon startup)
After startup the daemons log their messages in their spool directories.
   Qmaster:     /opt/sungrid/sge6_2u5/default/spool/qmaster/messages
   Exec daemon: <execd_spool_dir>/<hostname>/messages

Grid Engine startup scripts
---------------------------
Grid Engine startup scripts can be found at:

   /opt/sungrid/sge6_2u5/default/common/sgemaster (qmaster)
   /opt/sungrid/sge6_2u5/default/common/sgeexecd (execd)

Do you want to see previous screen about using Grid Engine again (y/n) [n] >>
Your execution daemon installation is now completed.
```

3.14 Installation of a Shadow Host

The host where the shadow master is installed must has access to these directories of the master host:

```
Spool: /opt/sungrid/sge6_2u5/default/common --> Shared Disk with the MASTER-HOST
    /common/shadow_masters  --> File that represents the order of
    intervention of the shadow master hosts
Spool: /opt/sungrid/sge6_2u5/default/spool/qmaster --> Shared Disk with the
Master Host
```

Load the environment variables:

```
. /opt/sungrid/sge6_2u5/default/common/settings.sh
./inst_sge -sm

 Shadow Master Host Setup
 ------------------------
  Make sure, that the host, you wish to configure as a shadow host,
  has read/write permissions to the '''qmaster spool and SGE_ROOT/<cell>/common
  directory!'''
  '''For using a shadow master it is recommended to set up a Berkeley DB Spooling Server'''
  Hit <RETURN> to continue >>

Choosing Grid Engine admin user account
---------------------------------------

You may install Grid Engine that all files are created with the user id of an
unprivileged user.

This will make it possible to install and run Grid Engine in directories
where user >root< has no permissions to create and write files and directories.

   - Grid Engine still has to be started by user >root<

   - this directory should be owned by the Grid Engine administrator

Do you want to install Grid Engine
under an user id other than >root< (y/n) [y] >>  n

Checking $SGE_ROOT directory
----------------------------

The Grid Engine root directory is:

   $SGE_ROOT = /opt/sungrid/sge6_2u5

If this directory is not correct (e.g. it may contain an automounter
prefix) enter the correct path to this directory or hit <RETURN>
to use default [/opt/sungrid/sge6_2u5] >> <ENTER>

Checking hostname resolving
---------------------------

Cannot contact qmaster. The command failed:

   ./bin/lx24-amd64/qconf -sh

The error message was:

   error: commlib error: got select error (Connection refused)
ERROR: unable to send message to qmaster using port 6444 on host "test-sge0":
 got send error

You can fix the problem now or abort the installation  procedure.
The problem can be:

   - the qmaster is not running
   - the qmaster host is down
   - an active firewall blocks your request

Contact qmaster again (y/n) ('n' will abort) [y] >>


Grid Engine JMX MBean server
----------------------------

In order to use the SGE Inspect or the Service Domain Manager (SDM)
SGE adapter you need to configure a JMX server in qmaster. Qmaster
will then load a Java Virtual Machine through a shared library.
NOTE: Java 1.5 or later is required for the JMX MBean server.

Please give some basic parameters for JMX MBean server
We may ask for
   - JAVA_HOME
   - additional JVM arguments (optional)

Detecting suitable JAVA ...

Enter JAVA_HOME (use "none" when none available) [/usr] >> /usr/lib/jvm/java-1.6.0-openjdk-1.6.0.0.x86_64/jre
Please enter additional JVM arguments (optional, default is [-Xmx256m]) >>

Using the following JMX MBean server settings.
   libjvm_path              >/usr/lib/jvm/java-1.6.0-openjdk-1.6.0.0.x86_64/jre/lib/amd64/server/libjvm.so<
   Additional JVM arguments >-Xmx256m<

Do you want to use these data (y/n) [y] >> y

Si utilizamos la opción Berkelye DB es posible que nos salga este error si no utilizamos una version
de NFS correcta:
 ERROR:
 Spooling directory exported as nfs is not supported!
 Please install the database directory on a NFSv4 fileserver or use the RPC Client/Server mechanism
Pero en principio nosotros No utilizaremos la BD berkeley lo haremos a través de ficheros en un NFS compartido.

Creating local configuration
----------------------------
root@test-sge1 added "test-sge1" to configuration list
Local configuration for host >test-sge1< created.

Hit <RETURN> to continue >>
Creating shadow_masters file for host test-sge1
Starting sge_shadowd on host test-sge1
Copying certificates to host test-sge1
Connecting as uid=0(root) to host test-sge0 ...
Warning: Permanently added 'test-sge0' (RSA) to the list of known hosts.
root@test-sge0's password:

The certificate copy failed for host test-sge1!
Shadowhost installation completed!
[root@test-sge1 sge6_2u5]#
```

Start / Stop the shadowd daemon:

```
$SGE_ROOT/$SGE_CELL/common/sgemaster -shadowd stop
$SGE_ROOT/$SGE_CELL/common/sgemaster -shadowd start
```

3.15 Configuration of the grid

3.15.1 Qmon: Graphical configuration

Qmon is a configuration and monitorization tool included with Sun Grid Engine

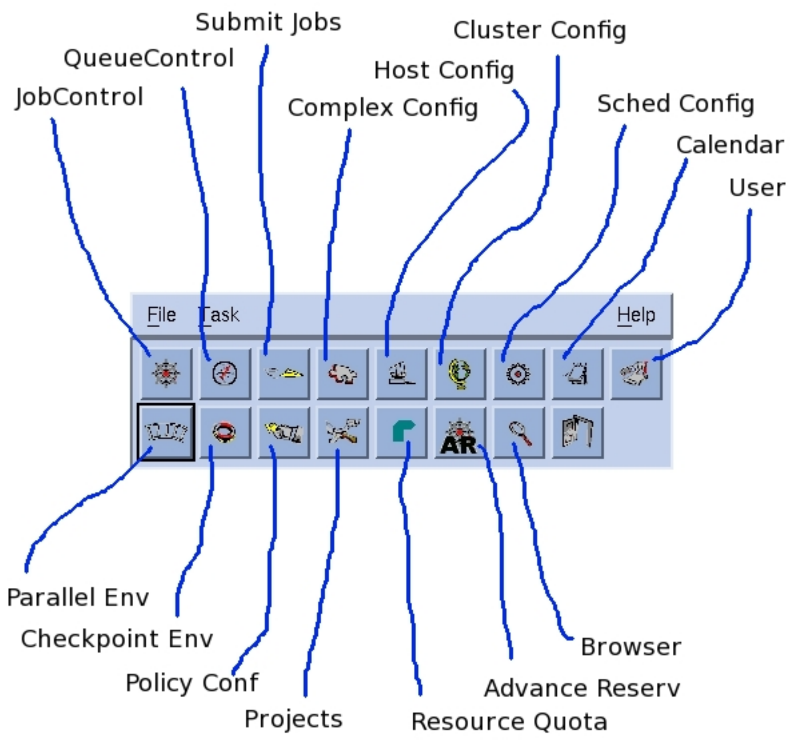
These are the requeriments of the tool:

```
 xorg-x11-xbitmaps.x86_64
 xorg-x11-fonts-75dpi.noarch
 xorg-x11-fonts-ISO8859-1-75dpi.noarch
 xorg-x11-xinit.x86_64
 openmotif-devel-2.3.2-1.fc8.x86_64
 openmotif-2.3.2-1.fc8.x86_64
 libXp.x86_64
 lesstif-0.95.0-26.el5.x86_64.rpm
 automake.noarch
 libXext-devel.x86_64
 libXp-devel.x86_64
 libXt-devel.x86_64
 lesstif-devel-0.95.0-26.el5.x86_64.rpm

In Path:  /opt/sungrid/sge6_2u5/lib/lx24-amd64 create the following link:
  lrwxrwxrwx 1 root     root          23 feb 25 15:48 libXm.so.3 ->
/usr/lib/libXm.so.4.0.2
```

If i execute qmon from a remote server, don't forget execute in the local host xhost + and in the remote a export of the variable DISPLAY to the local host IP:

```
xhost +
ssh user@server
export DISPLAY="192.168.X.XX:0.0"
. /opt/sungrid/sge6_2u5/default/common/settings.sh
qmon &
```

Figure 3.22: Qmon Main screen



3.15.2 Command line configuration

We can see and modify the current configuration of the grid also from the command line with the command qconf:

qconf: -s: show, -m:modify:

examples:

```
./qconf -sh             : Show the name of master host
./qconf -sql            : Show all the queues
./qconf -ss             : Show all the hosts
./qconf -sconf          : Show the config of the cluster
./qconf -suserl         : Show the user list
./qconf -auser          : Add a user
./qconf -sq <cola>      : Show the config of a queue
./qconf -mq <cola>      : Modify the config of a queue
./qconf -ssconf         : Show the config of the Scheduler
./qconf -msconf         : Modify the config of the Scheduler
./qconf -so             : Show the users with operator role
./qconf -ae             : Add a host
./qconf -Me             : Modify a host
./qconf -de             : Erase a host
./qconf -sel            : Show list of execution hosts
./qconf -sm             : Show the users with manager role.
```

```
[root@deltaweb bin]# qconf -sconf
#global:
execd_spool_dir              /opt/sungrid/sge6_2u5/default/spool
mailer                       /bin/mail
xterm                        /usr/bin/X11/xterm
load_sensor                  none
```

```
prolog                       none
epilog                       none
shell_start_mode             posix_compliant
login_shells                 sh,ksh,csh,tcsh
min_uid                      0
min_gid                      0
user_lists                   none
xuser_lists                  none
projects                     none
xprojects                    none
enforce_project              false
enforce_user                 auto
load_report_time             00:00:40
max_unheard                  00:01:00
reschedule_unknown           00:01:00
loglevel                     log_warning
administrator_mail           antonio.carmona@-is.es
set_token_cmd                none
pag_cmd                      none
token_extend_time            none
shepherd_cmd                 none
qmaster_params               ENABLE_RESCHEDULE_KILL=true
execd_params                 none
reporting_params             accounting=true reporting=false \
                             flush_time=00:00:15 joblog=false sharelog=00:00:00
finished_jobs                100
gid_range                    20000-20100
qlogin_command               builtin
qlogin_daemon                builtin
rlogin_command               builtin
rlogin_daemon                builtin
rsh_command                  builtin
rsh_daemon                   builtin
max_aj_instances             2000
max_aj_tasks                 75000
max_u_jobs                   0
max_jobs                     0
max_advance_reservations     0
auto_user_oticket            0
auto_user_fshare             0
auto_user_default_project    none
auto_user_delete_time        86400
delegated_file_staging       false
reprioritize                 false
jsv_url                      none
libjvm_path                  /usr/lib/jvm/java-1.6.0-openjdk-1.6.0.0.x86_64/jre/lib/amd64/server/libjvm.so
additional_jvm_args          -Xmx256m
jsv_allowed_mod              ac,h,i,e,o,j,M,N,p,w
```

3.15.3 SGE Environment Variables

The main environment variables are:

```
SGE_ROOT=/opt/sungrid/sge6_2u5
SGE_CELL=default
SGE_CLUSTER_NAME=-cluster
SGE_QMASTER_PORT=6444
SGE_EXECD_PORT=6445
```

Also a job launched by SGE has access to the next environment variables:

- ARCH: Architecture : example: ARCH=$SGE_ROOT/util/arch
- COMMD_PORT: Specifies the TCP port on which sge_commd(8) is expected to listen for communication requests
- SGE_ROOT: Directory of installation SGE_ROOT=/opt/sungrid/sge6_2u5

- SGE_CELL: Cell where the job is executed SGE_CELL=default
- SGE_JOB_SPOOL_DIR: The directory used by sge_shepherd(8) to store jobrelated data during job execution
- SGE_O_HOME: - The home directory path of the job owner on the host from which the job was submitted
- SGE_O_HOST: - The host from which the job was submitted
- SGE_O_LOGNAME: - The login name of the job owner on the host from which the job was submitted
- SGE_O_MAIL: - The content of the MAIL environment variable in the context of the job submission command
- SGE_O_PATH: - The content of the PATH environment variable in the context of the job submission command
- SGE_O_SHELL: - The content of the SHELL environment variable in the context of the job submission command
- SGE_O_TZ: - The content of the TZ environment variable in the context of the job submission command
- SGE_O_WORKDIR: - The working directory of the job submission command
- SGE_CKPT_ENV: - Specifies the checkpointing environment (as selected with the qsub -ckpt option) under which a checkpointing job executes
- SGE_CKPT_DIR: - Only set for checkpointing jobs; contains path ckpt_dir (see the checkpoint manual page) of the checkpoint interface
- SGE_STDERR_PATH: - The path name of the file to which the standard error stream of the job is diverted; commonly used for enhancing the output with error messages from prolog, epilog, parallel environment start/stop or checkpointing scripts
- SGE_STDOUT_PATH: - The path name of the file to which the standard output stream of the job is diverted; commonly used for enhancing the output with messages from prolog, epilog, parallel environment start/stop or checkpointing scripts
- SGE_TASK_ID: - The task identifier in the array job represented by this task ENVIRONMENT - Always set to BATCH; this variable indicates that the script is run in batch mode
- HOME - The users home directory path from the passwd file
- HOSTNAME: - The host name of the node on which the job is running
- JOB_ID: - A unique identifier assigned by the sge_qmaster when the job was submitted; the job ID is a decimal integer in the range to 99999
- JOB_NAME: - The job name, built from the qsub script filename, a period, and the digits of the job ID; this default may be overwritten by qsub -N
- LOGNAME: - The user's login name from the passwd file

- NHOSTS: - The number of hosts in use by a parallel job
- NQUEUES: - The number of queues allocated for the job (always 1 for serial jobs)
- NSLOTS: - The number of queue slots in use by a parallel job

Environment variables of a Shadow Master Host

These environment variables are essential for a correct and faster transtition of the master host to another node.

```
export SGE_DELAY_TIME=60
export SGE_CHECK_INTERVAL=40
export SGE_GET_ACTIVE_INTERVAL=20
```

- SGE_DELAY_TIME : This variable controls the interval in which sge_shadowd pauses if a takeover bid fails. This value is used only when there are multiple sge_shadowd instances that are contending to be the master (the default is 600 seconds).
- SGE_CHECK_INTERVAL :This variable controls the interval in which the sge_shadowd checks the heartbeat file (the default is 60 seconds).
- SGE_GET_ACTIVE_INTERVAL :This variable controls the interval when a sge_shadowd instance tries to take over when the heartbeat file has not changed.

Job Script example

```
#!/bin/sh  This script is interpreted by the Bourne shell, sh.
#$ -N JobName
#$ -S /bin/sh Sun Grid Engine always uses sh to interpret this script.
#$ -a 12241200         // The job can be launched since 12:00 of 24 of December
#$ -dl 12241800        // DEADLINE, after this date the process must not be
executed
#$ -r y     // The job can be relaunched in case of error
#$ -j y
#$ -l recurso1=5
#$ -pe pvm 6- -l mem=128 // I need 6 or more parallel process and 128 MB or
memory
#$ -e moscar1:/tmp,moscar2:/var/tmp  // If the process is executed in node moscar1,
 the path of the error files is /tmp, instead if is moscar2 the path is /var/tmp
#$ -M antonio.carmona@-is.es  // Send mails to this users
#$ -m bes     // Send mails under this circustances: beginning/end/on suspension
# Specify that these environment variables are to be sent to SGE with the job:
#$ -v DISPLAY
#$ -v VGL_CLIENT
#$ -v VGL_GAMMA
#$ -v VGL_GLLIB
#$ -v VGL_SPOIL
#$ -v VGL_X11LIB
#$ -v SSH_CLIENT
# If these variables are not set before qsub/qrsh is invoked,
# then the job will find these variables set, but with a null string value ("").
#
```

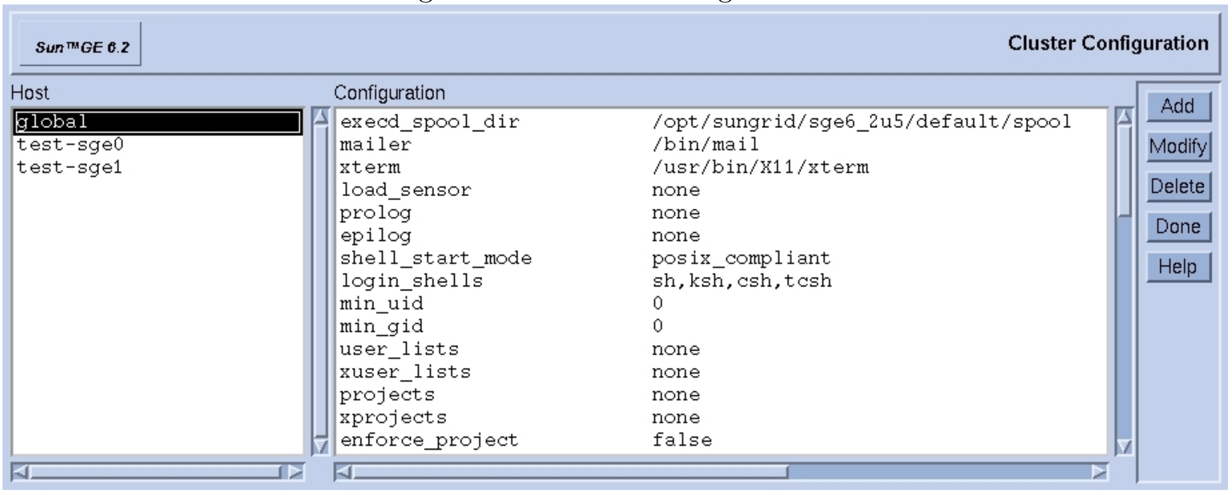
3.15.4 Configuration of the cluster

For configure the cluster we must pulse **"Cluster Configuration"** in the main screen of Qmon:

Selecting a host on **"global"** and after in modify we can modify some global parameters of the cluster.

The next configuration parameters are needed for the right operation of the grid (See figure 3.22 and 3.23)

Figure 3.23: Cluster configuration



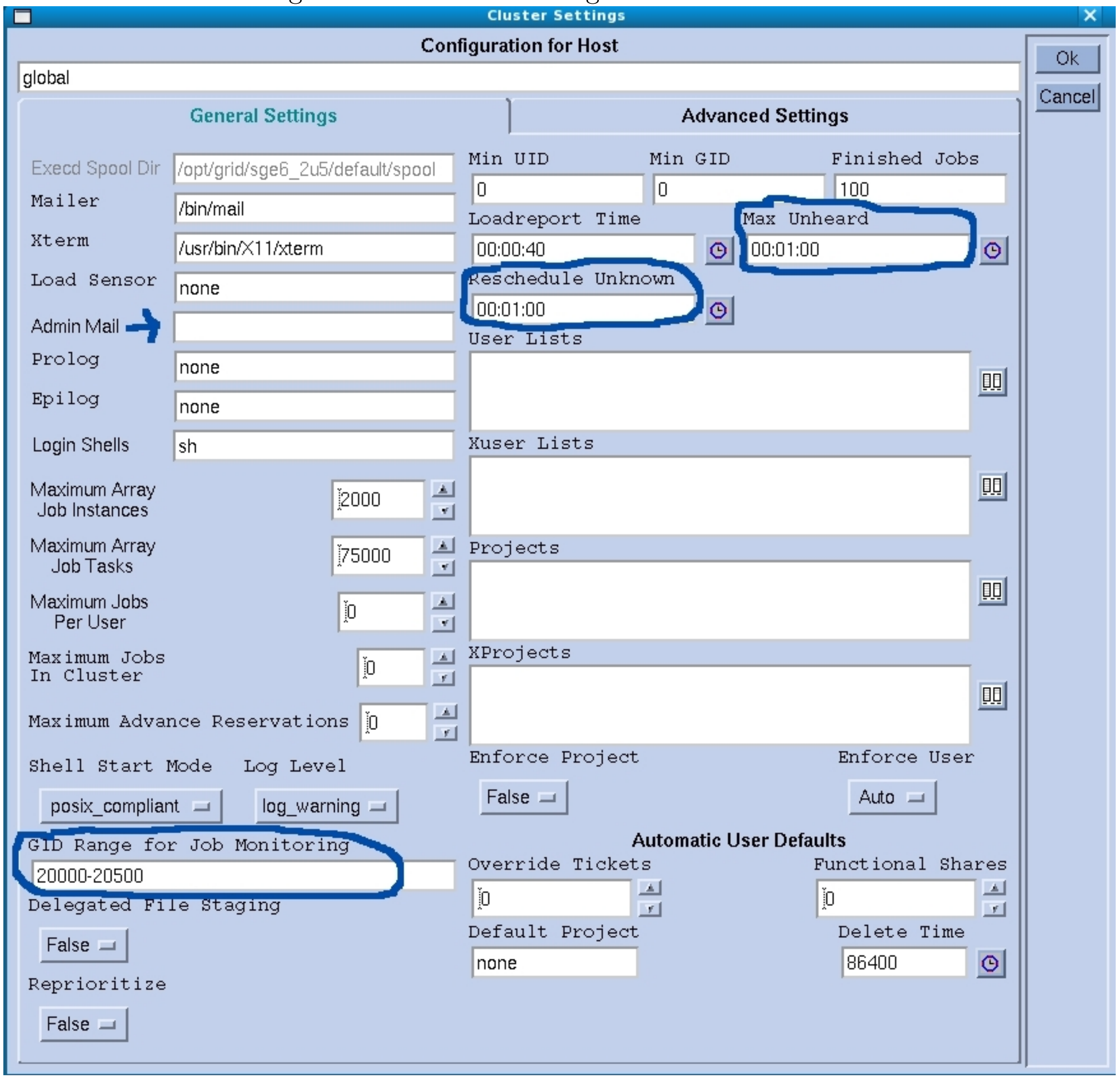
```
global
test-sge0
test-sge1
```

```
execd_spool_dir             /opt/sungrid/sge6_2u5/default/spool
mailer                      /bin/mail
xterm                       /usr/bin/X11/xterm
load_sensor                 none
prolog                      none
epilog                      none
shell_start_mode            posix_compliant
login_shells                sh,ksh,csh,tcsh
min_uid                     0
min_gid                     0
user_lists                  none
xuser_lists                 none
projects                    none
xprojects                   none
enforce_project             false
```

- Is necesary change the value of **max_unheard** for SGE be more sensible!!! of a down of a host, otherwise the sever takes much time awake of the down of a server.
- Modify **reschedule_unknown**
- Add the to the parameter : qmaster_params the attribute: **ENABLE_RESCHEDULE_KILL=true** : for free the hangup process and doesn't be in running state permanetly due to doesn't receive response for example after the down of the execution hosts where the jobs were executing. (See figure 3.25)

In the figure 3.24 we can see two very important parameters: Log Level and Max unheard that indicate the maximum time once terminated a master host is considered down.

Figure 3.24: Cluster settings: Max time unheard



Cluster Settings

Configuration for Host

global

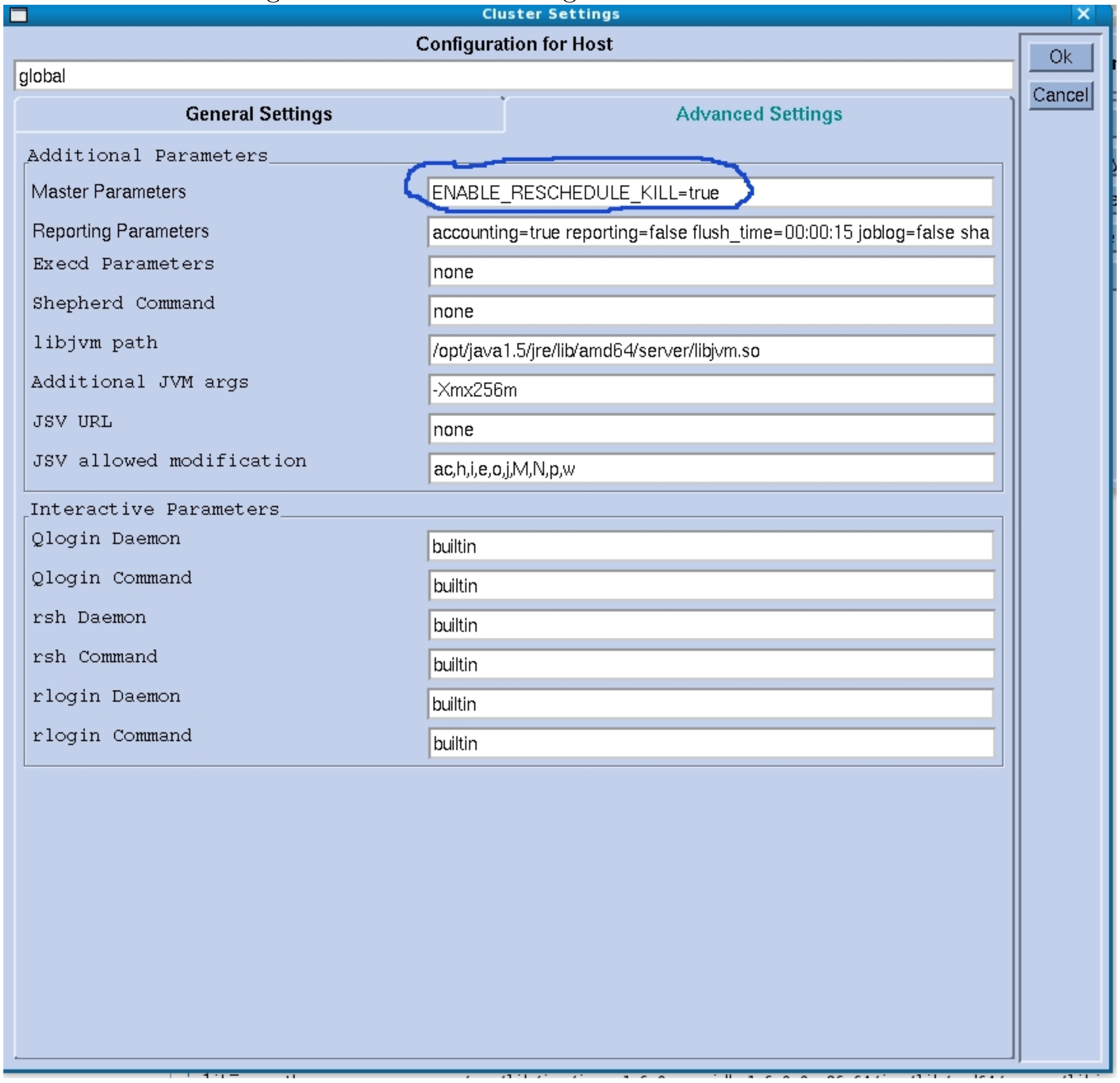
General Settings | Advanced Settings

Execd Spool Dir	/opt/grid/sge6_2u5/default/spool
Mailer	/bin/mail
Xterm	/usr/bin/X11/xterm
Load Sensor	none
Admin Mail	
Prolog	none
Epilog	none
Login Shells	sh
Maximum Array Job Instances	2000
Maximum Array Job Tasks	75000
Maximum Jobs Per User	0
Maximum Jobs In Cluster	0
Maximum Advance Reservations	0
Shell Start Mode	posix_compliant
Log Level	log_warning
GID Range for Job Monitoring	20000-20500
Delegated File Staging	False
Reprioritize	False
Min UID	0
Min GID	0
Finished Jobs	100
Loadreport Time	00:00:40
Max Unheard	00:01:00
Reschedule Unknown	00:01:00
User Lists	
Xuser Lists	
Projects	
XProjects	
Enforce Project	False
Enforce User	Auto

Automatic User Defaults

Override Tickets	0
Functional Shares	0
Default Project	none
Delete Time	86400

Figure 3.25: Cluster settings: Enable reschedule kill.



Configuration for Host	global

Additional Parameters

Master Parameters	ENABLE_RESCHEDULE_KILL=true
Reporting Parameters	accounting=true reporting=false flush_time=00:00:15 joblog=false sha
Execd Parameters	none
Shepherd Command	none
libjvm path	/opt/java1.5/jre/lib/amd64/server/libjvm.so
Additional JVM args	-Xmx256m
JSV URL	none
JSV allowed modification	ac,h,i,e,o,j,M,N,p,w

Interactive Parameters

Qlogin Daemon	builtin
Qlogin Command	builtin
rsh Daemon	builtin
rsh Command	builtin
rlogin Daemon	builtin
rlogin Command	builtin

3.15.5 Configuration of queues

The configuration of queues can be made since the **“Queue Control”** screen , selecting a queue and after pulsing **“Modify”**

General configuration

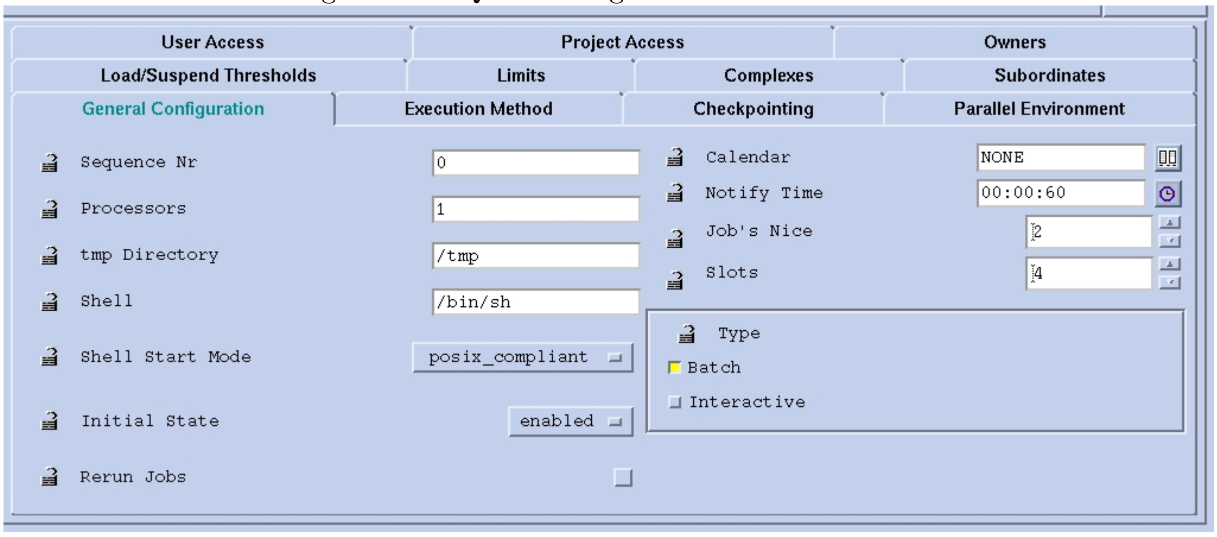
See figure 3.26

- Sequence Nr: Allow us sort the queues for determine in what order are filled (lower number, more priority)
- A specifier for the processor set to be used by the jobs running in that queue. For some operating system architectures, this specifier can be a range, such as 14,8,10, or just an integer identifier of the processor set. See the arc_depend_.asc files in the doc directory of your Grid Engine software distribution for more information. Caution Do not change this value unless you are certain that you need to change it.
- Temporary directory path.
- Shell Default command interpreter to use for running the job scripts.
- Shell Start Mode The mode in which to start the job script.

```
1. posix_compliant
2.NONE
3.Script_from_stdin
4.unix_behavior
```

- Initial State The state in which a newly added queue comes up. Also, the state in which a queue instance is restored if the sge_execd running on the queue instance host gets restarted.
- Rerun Jobs The queue’s default rerun policy to be enforced on jobs that were aborted, for example, due to system crashes. The user can override this policy using the qsub -r command or the Submit Job dialog box.
- Calendar A calendar attached to the queue. This calendar defines on-duty and off-duty times for the queue.
- Notify Time The time to wait between delivery of SIGUSR1,SIGUSR2 notification signals and suspend or kill signals. After this time a process is considered like hangup and will be killed.
- Job’s Nice The nice value with which to start the jobs in this queue. 0 means use the system default.
- Slots: The number of jobs that are allowed to run concurrently in the queue. Slots are also referred to as job slots.
- Type The type of the queue and of the jobs that are allowed to run in this queue. Type can be Batch, Interactive, or both.

Figure 3.26: Queue configuration: General screen



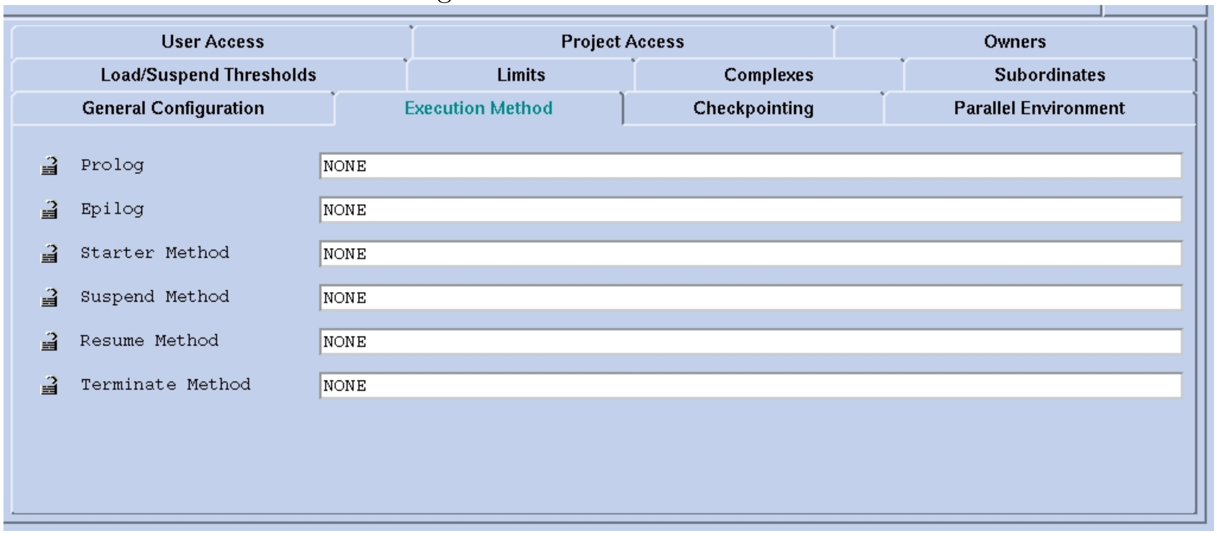
Sequence Nr 0
Processors 1
tmp Directory /tmp
Shell /bin/sh
Shell Start Mode posix_compliant
Initial State enabled
Rerun Jobs
Calendar NONE
Notify Time 00:00:60
Job's Nice 2
Slots 4
Type
Batch
Interactive

Execution method

See figure 3.27

- **Prolog** A queue-specific prolog script. The prolog script is run with the same environment as the job before the job script is started.
- **Epilog** A queue-specific epilog script. The epilog script is run with the same environment as the job after the job is finished.
- **Starter Method, Suspend Method, Resume Method, Terminate Method** Use these fields to override the default methods for applying these actions to jobs.

Figure 3.27: Execution method



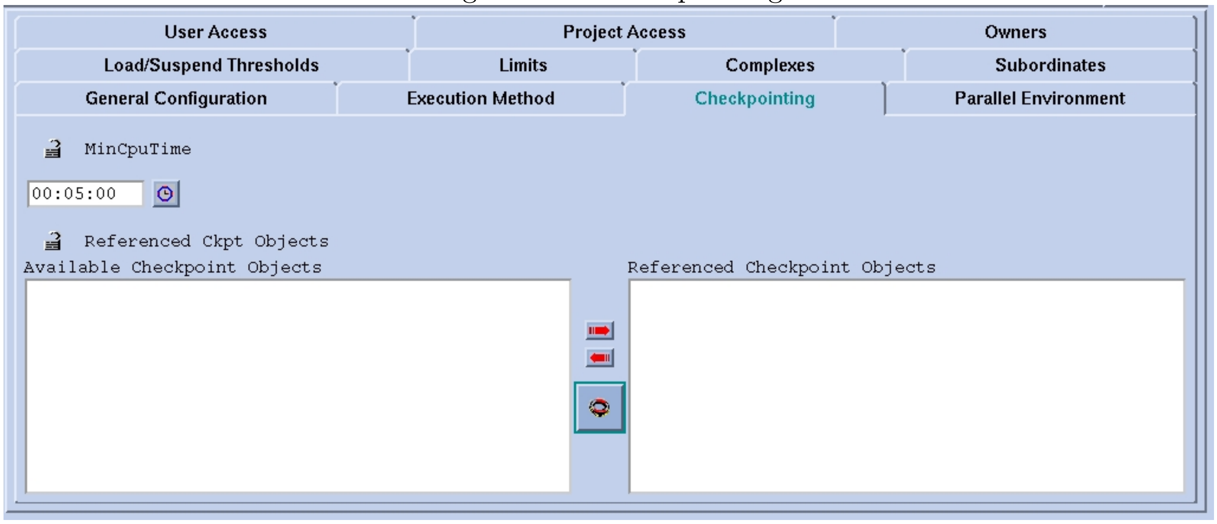
Prolog	NONE
Epilog	NONE
Starter Method	NONE
Suspend Method	NONE
Resume Method	NONE
Terminate Method	NONE

Checkpointing

See figure 3.28

- To modify the MinCpuTime, click on the clock icon and then adjust the time.
- To reference a checkpointing environment from the queue, select the name of a checkpointing environment from the Available list, and then click the right arrow to add it to the Referenced list.
- To remove a checkpointing environment from the Referenced list, select it, and then click the left arrow.
- To add or modify checkpointing environments, click the button below the red arrows to open the Checkpointing Configuration dialog box.

Figure 3.28: Checkpointing



MinCpuTime

00:05:00

Referenced Ckpt Objects

Available Checkpoint Objects

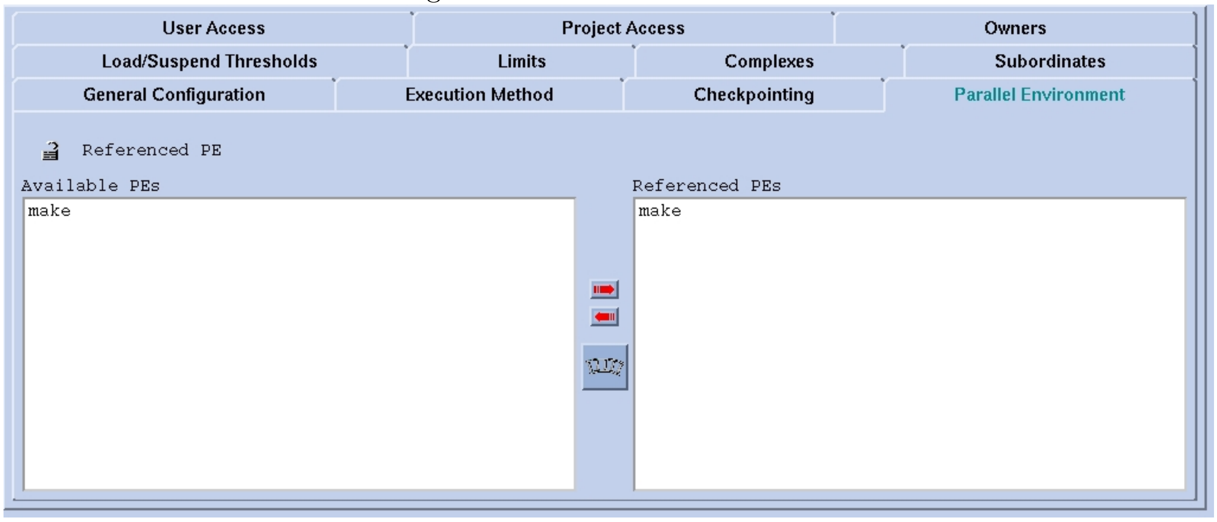
Referenced Checkpoint Objects

Parallel Enviroment

See figure 3.29

- To reference a parallel environment from the queue, select the name of a parallel environment from the Available PEs list, and then click the right arrow to add it to the Referenced PEs list.
- To remove a checkpointing environment from the Referenced PEs list, select it, and then click the left arrow.
- To add or modify parallel environments, click the button below the red arrows to open the Parallel Environment Configuration dialog box.

Figure 3.29: Parallel Enviroment



```
Available PEs
make

Referenced PEs
make
```

Load/Suspend Thresholds

See figure 3.30

Types of Thresholds:

- **Load Thresholds**: Don't allow that more process are queued.
- **Suspend Thresholds**: Suspend jobs in the queue for reduce the load.

The Load Thresholds and the Suspend Thresholds tables display the defined overload thresholds for load parameters and consumable resource attributes. In the case of load thresholds, overload prevents the queue from receiving further jobs. In the case of suspend thresholds, overload suspends jobs in the queue to reduce the load.

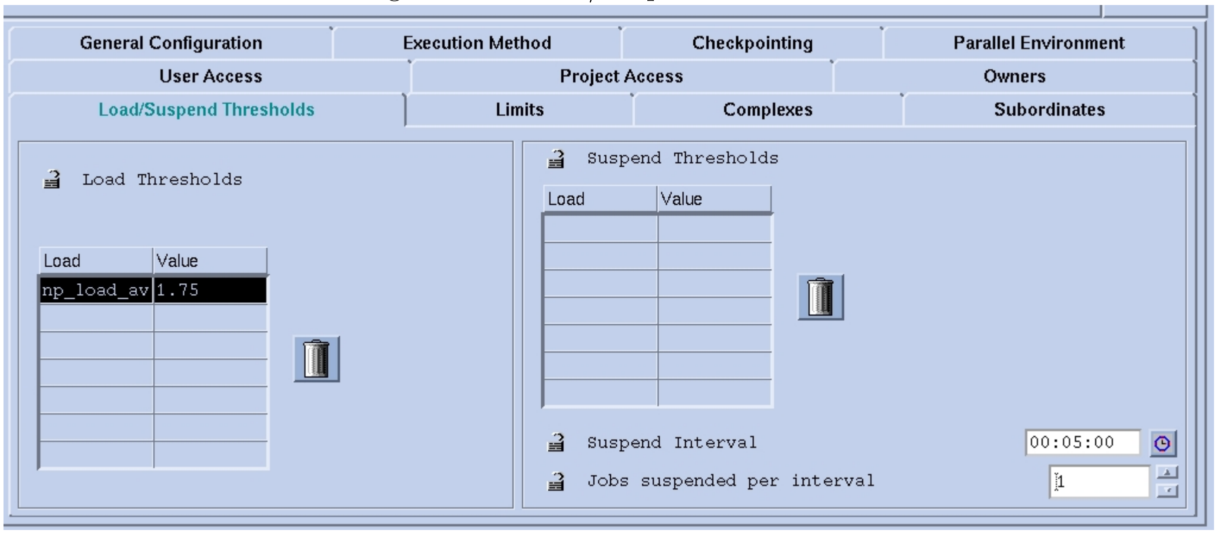
- Suspend interval. The time interval between suspension of other jobs in case the suspend thresholds are still exceeded.
- Jobs suspended per interval. The number of jobs to suspend per time interval in order to reduce the load on the system that is hosting the configured queue.

  From the Load/Suspend thresholds tab, you can perform the following tasks:
- To change an existing threshold, select it, and then double-click the corresponding Value field.
- To add new thresholds, do the following:

```
1.Click Load or Value.
    A selection list appears with all valid attributes that are attached to the queue.
2.To add an attribute to the Load column of the corresponding threshold table,
 select an attribute, and then click OK.
```

- To delete an existing threshold, select it, and then type Ctrl-D.

Figure 3.30: Load/Suspend Thresholds



Load Thresholds

Load	Value
np_load_av	1.75

Suspend Thresholds

Load	Value

Suspend Interval 00:05:00

Jobs suspended per interval 1

Limits

See figure 3.31

A dialog box appears where you can type either Memory or Time limit values. When a hard limit is exceeded, the running job in the queue is stopped immediately. When a soft limit is exceeded, a signal is sent that the job can intercept before the job is stopped.

Complexes : Resources

See figure 3.32

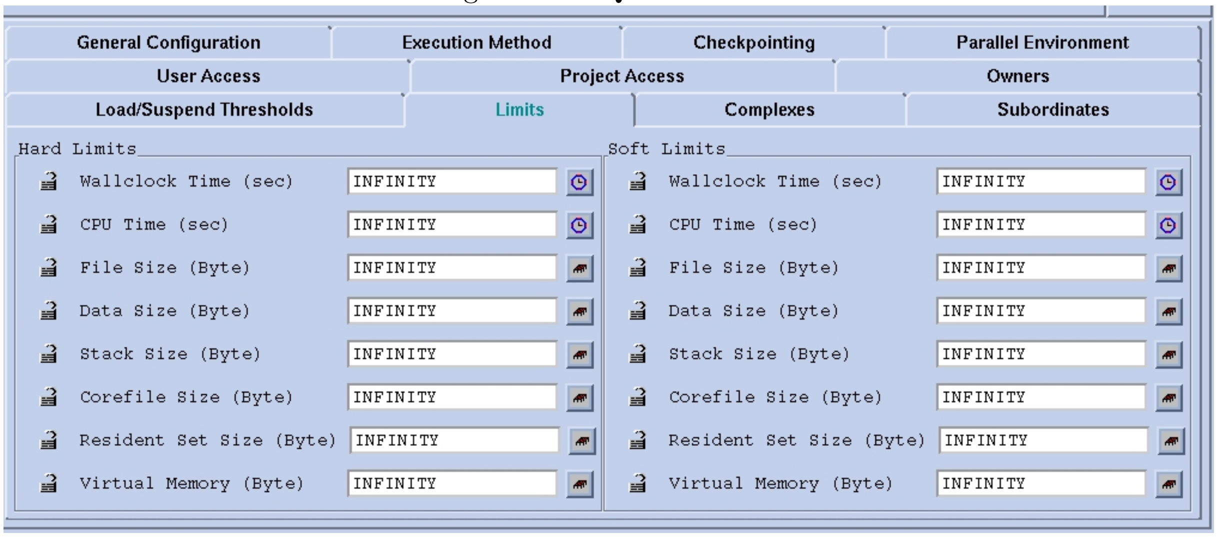
Resource attributes are either consumable or fixed. The definition of a consumable value defines a capacity managed by the queue. The definition of a fixed value defines a queue-specific value.

- To modify an attribute, select it, and then double-click the corresponding Value field.
- To add new attribute definitions, click Load or Value.

  The Attribute Selection dialog box appears with a list of all valid attributes that are attached to the queue.
- To add an attribute to the Load column of the attribute table, select it, and then click OK.
- To delete an attribute, select it, and then press Ctrl-D.

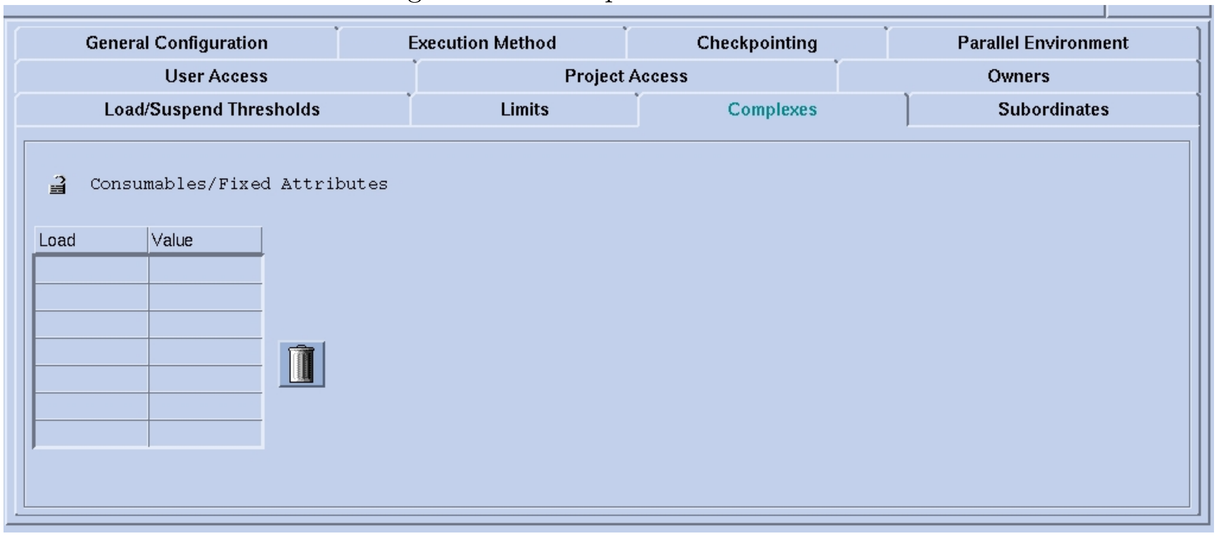
Use the Complex Configuration dialog box to check or modify the current complex configuration before you attach user-defined resource attributes to a queue or before you detach them from a queue. To access the Complex Configuration dialog box, click the Complex Configuration button on the QMON Main Control window.

Figure 3.31: Qmon: Limits



Hard Limits		Soft Limits	
Wallclock Time (sec)	INFINITY	Wallclock Time (sec)	INFINITY
CPU Time (sec)	INFINITY	CPU Time (sec)	INFINITY
File Size (Byte)	INFINITY	File Size (Byte)	INFINITY
Data Size (Byte)	INFINITY	Data Size (Byte)	INFINITY
Stack Size (Byte)	INFINITY	Stack Size (Byte)	INFINITY
Corefile Size (Byte)	INFINITY	Corefile Size (Byte)	INFINITY
Resident Set Size (Byte)	INFINITY	Resident Set Size (Byte)	INFINITY
Virtual Memory (Byte)	INFINITY	Virtual Memory (Byte)	INFINITY

Figure 3.32: Complexes : Resources



Consumables/Fixed Attributes

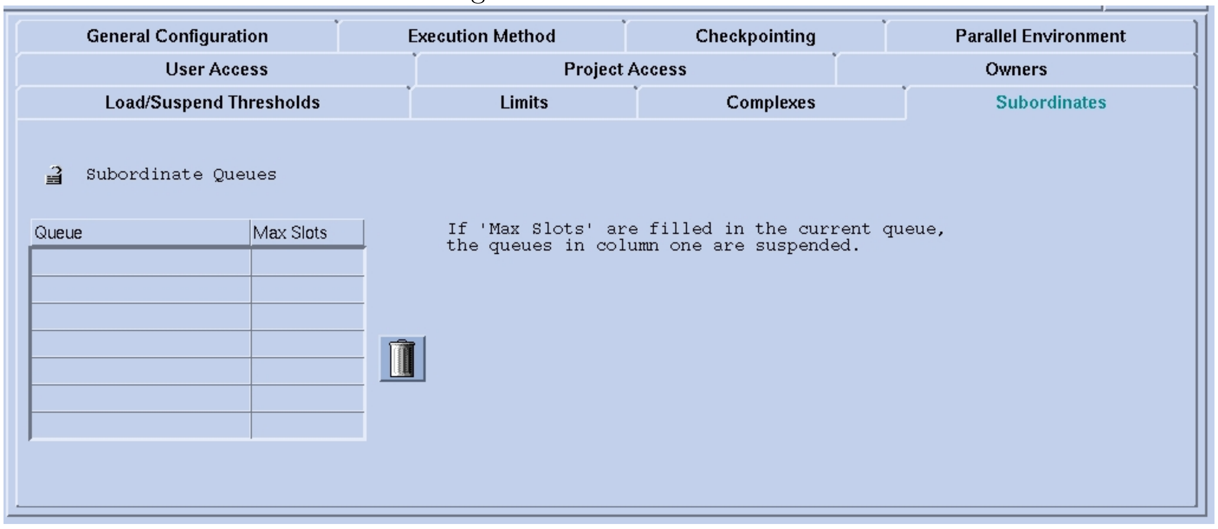
Load	Value

Subordinates

See figure 3.33

- Queue A list of the queues that are subordinated to the configured queue.
- Subordinated queue instances on a host are suspended if the configured queue instance on the same host becomes busy. Subordinated queue instances on that host are resumed when the configured queue instance is no longer busy.
- Max Slots For any subordinated queue, you can configure the number of job slots that must be filled in the configured queue to trigger a suspension. If no maximum slot value is specified, all job slots must be filled to trigger suspension of the corresponding queue.

Figure 3.33: Subordinates



If 'Max Slots' are filled in the current queue,
the queues in column one are suspended.

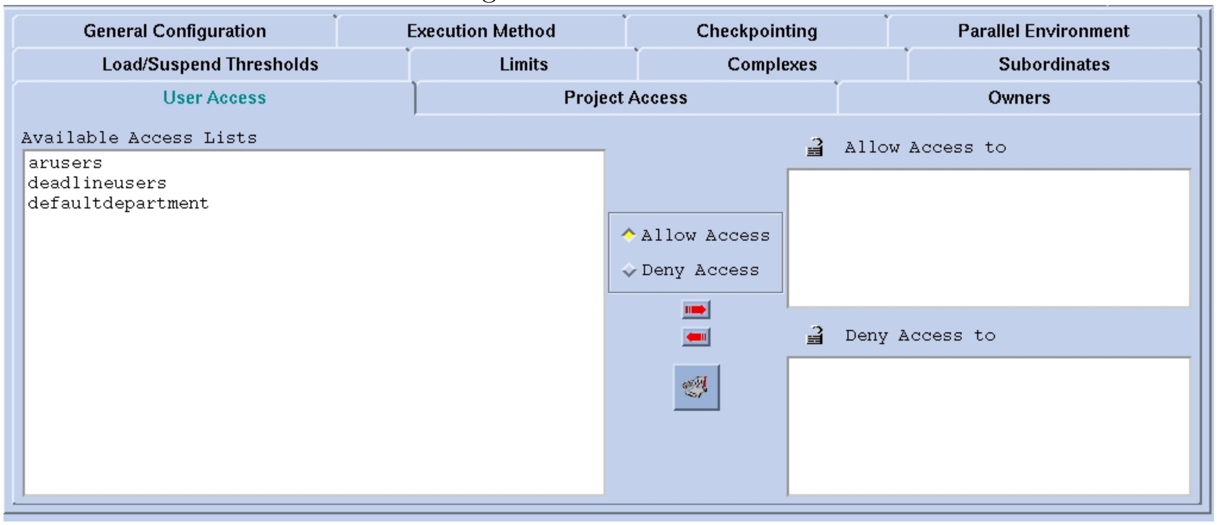
User access

See figure 3.34

Users or user groups belonging to access lists that are included in the Allow Access list have access to the queue. Users who are included in the Deny Access list cannot access the queue. If the Allow Access list is empty, access is unrestricted unless explicitly stated otherwise in the Deny Access list.

- To add user access lists, do the following: 1.Click the button below the red arrows. The User Configuration dialog box appears. 2.Click Add. 3.Click Ok to save your changes. Click Cancel to leave the dialog box without saving your changes.
- To modify user access lists, do the following 1.Click the button below the red arrows. The User Configuration dialog box appears. 2.Select a user access list and click Modify. 3.Click Ok to save your changes. Click Cancel to leave the dialog box without saving your changes.

Figure 3.34: User access



```
arusers
deadlineusers
defaultdepartment
```

Projects access securization

See figure 3.35

Jobs that are submitted to a project that belongs to the list of allowed projects have access to the queue. Jobs that are submitted to denied projects are not dispatched to the queue.

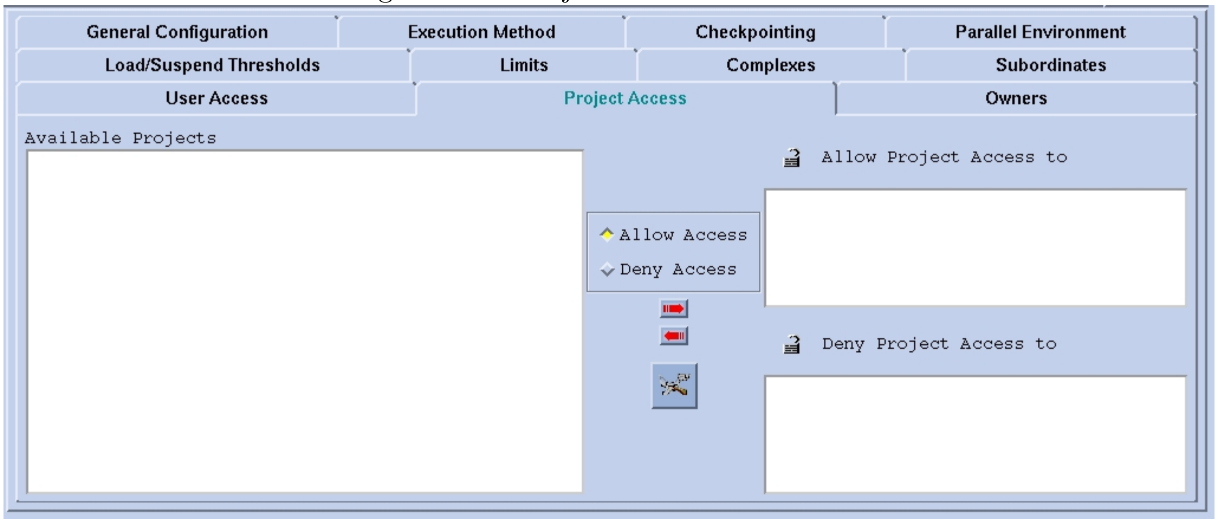
- To add project access, do the following:

```
1.Click the button below the red arrows.
The Project Configuration dialog box appears.
2.Click Add.
3.Click Ok to save any changes.
Click Cancel to leave the dialog box without saving your changes.
```

- To modify project access, do the following:

```
1.Click the button below the red arrows.
The Project Configuration dialog box appears.
2.Select a queue and click modify.
3.Click Ok to save your changes.
Click Cancel to leave the dialog box without saving your changes.
```

Figure 3.35: Projects access securization



3.15.6 Configuration of resources

From the main screen of Qmon, pulsing “Complex Configuration” i have access to all the different resources available at queue , host or cluster level.

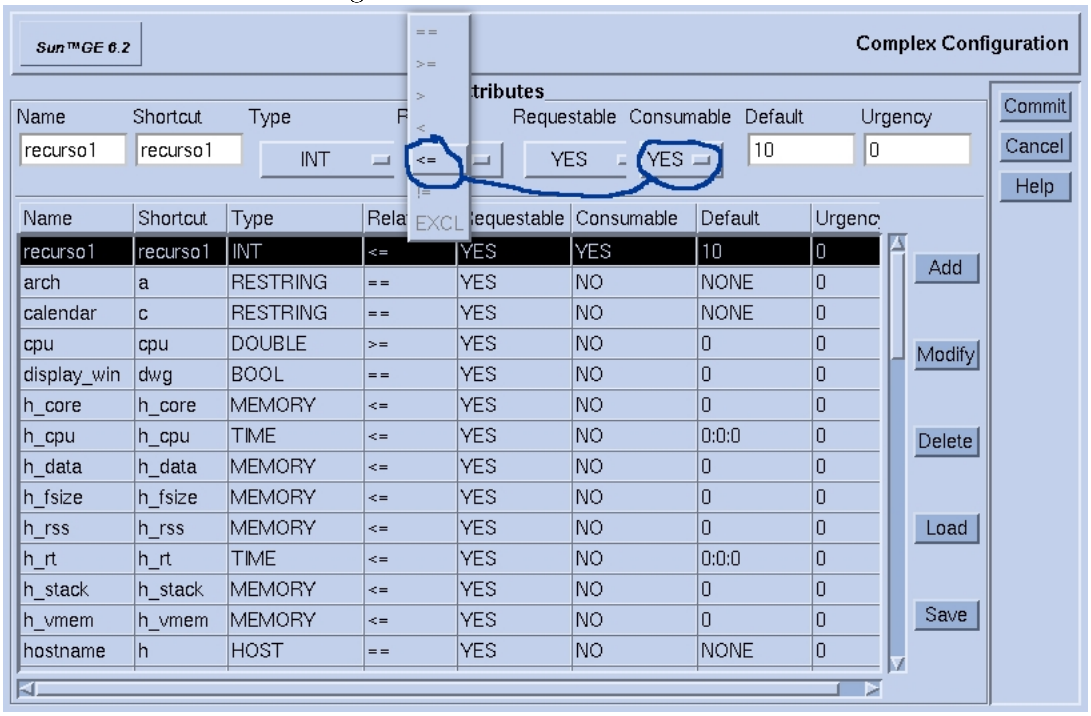
Create a new resource

In the figure 3.36 we can see the creation of a consumable resource,with this kind of resource only one type of relation is available “Minor or Equal”

For stop a queue , we will create a fictitious resource (called rbc-STOP in the example) wich will allow us enable / disable queues according the existence / need of this fictitious resource.

We will create it in the tab Complex of the Qmon tool (figure 3.37)

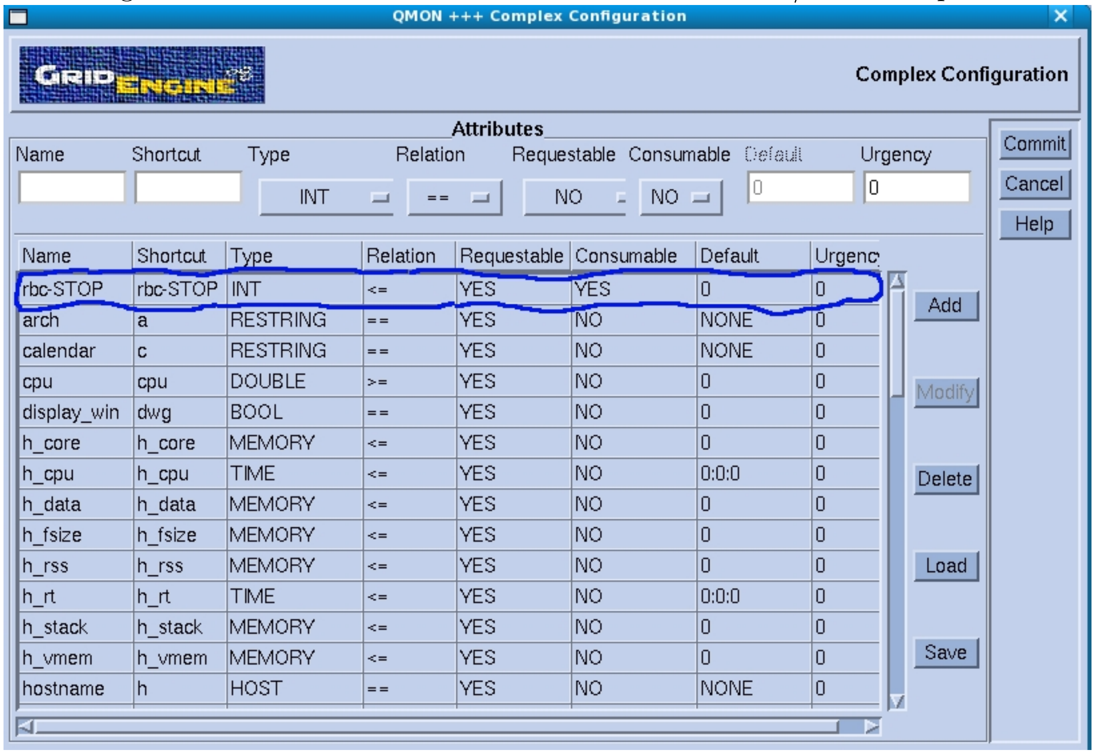
Figure 3.36: Create a new resource



Name	Shortcut	Type	Relation	Requestable	Consumable	Default	Urgency
recurso1	recurso1	INT	<=	YES	YES	10	0

Name	Shortcut	Type	Relation	Requestable	Consumable	Default	Urgency
recurso1	recurso1	INT	<=	YES	YES	10	0
arch	a	RESTRING	==	YES	NO	NONE	0
calendar	c	RESTRING	==	YES	NO	NONE	0
cpu	cpu	DOUBLE	>=	YES	NO	0	0
display_win	dwg	BOOL	==	YES	NO	0	0
h_core	h_core	MEMORY	<=	YES	NO	0	0
h_cpu	h_cpu	TIME	<=	YES	NO	0:0:0	0
h_data	h_data	MEMORY	<=	YES	NO	0	0
h_fsize	h_fsize	MEMORY	<=	YES	NO	0	0
h_rss	h_rss	MEMORY	<=	YES	NO	0	0
h_rt	h_rt	TIME	<=	YES	NO	0:0:0	0
h_stack	h_stack	MEMORY	<=	YES	NO	0	0
h_vmem	h_vmem	MEMORY	<=	YES	NO	0	0
hostname	h	HOST	==	YES	NO	NONE	0

Figure 3.37: Creation of a fictitious resource for enable/disable a queue



Name	Shortcut	Type	Relation	Requestable	Consumable	Default	Urgency
		INT	==	NO	NO	0	0

Name	Shortcut	Type	Relation	Requestable	Consumable	Default	Urgency
rbc-STOP	rbc-STOP	INT	<=	YES	YES	0	0
arch	a	RESTRING	==	YES	NO	NONE	0
calendar	c	RESTRING	==	YES	NO	NONE	0
cpu	cpu	DOUBLE	>=	YES	NO	0	0
display_win	dwg	BOOL	==	YES	NO	0	0
h_core	h_core	MEMORY	<=	YES	NO	0	0
h_cpu	h_cpu	TIME	<=	YES	NO	0:0:0	0
h_data	h_data	MEMORY	<=	YES	NO	0	0
h_fsize	h_fsize	MEMORY	<=	YES	NO	0	0
h_rss	h_rss	MEMORY	<=	YES	NO	0	0
h_rt	h_rt	TIME	<=	YES	NO	0:0:0	0
h_stack	h_stack	MEMORY	<=	YES	NO	0	0
h_vmem	h_vmem	MEMORY	<=	YES	NO	0	0
hostname	h	HOST	==	YES	NO	NONE	0

Assign a resource to a queue

If we want to assign a resource to a queue, from qmon we access to “Cluster Queue Control” select a queue, pulse in “Modify” and in the tab “Complexes” add this resource and its value. (See figure 3.38)

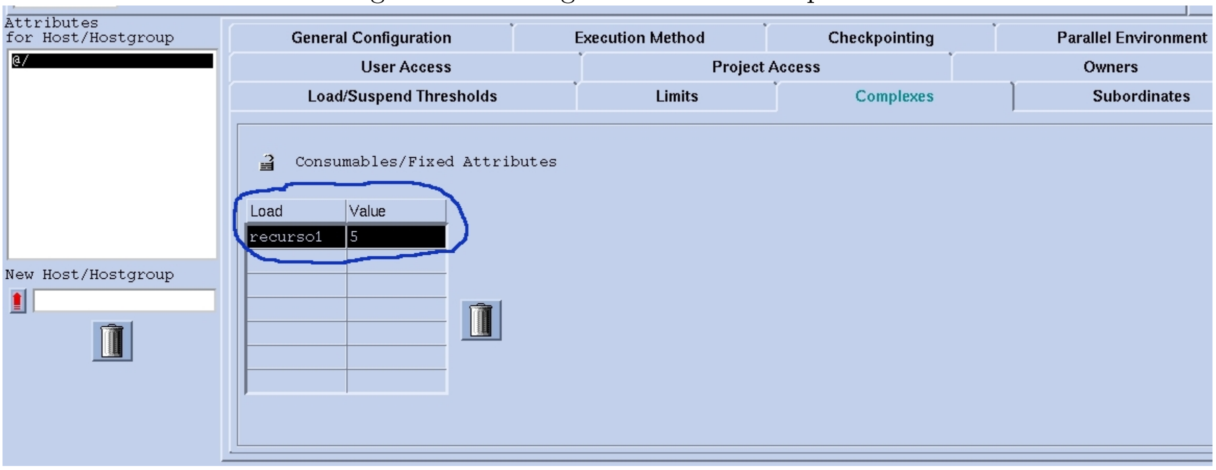
Assign a resource to a host

If we want to assign the resource to a host, we must access to “Host Configuration” select tab “Execution Host”, select a host, pulse in the window “Consumables / Fixed Attributes” and after in “Modify” and another time in the tab “Consumables / Fixed Attributes” where we must indicate the new resource and its value. (See figure 3.39 and 3.40)

Assign a resource to a cluster

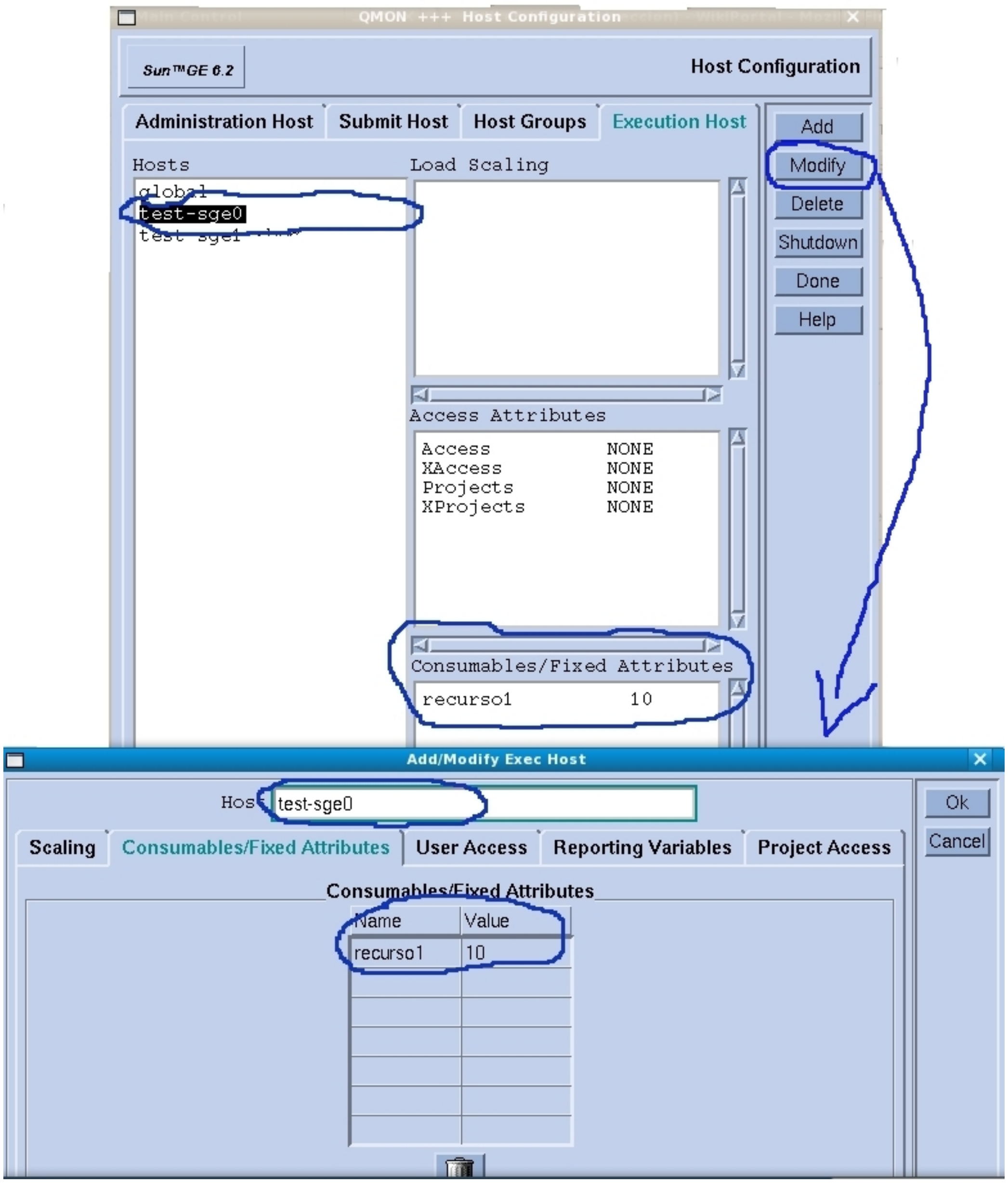
If we want to assign resources globally to the cluster and that can affect to all the servers that compound it, we must enter in “Host Configuration” in Qmon, after pulse in tab “Execution Hosts” , select in form Hosts global and after pulse in “Modify” for can have access to the global resources of the cluster. (See figure 3.41)

Figure 3.38: Assign a resource to a queue



Load	Value
recurso1	5

Figure 3.39: Assign a resource to a host



```
Access          NONE
XAccess         NONE
Projects        NONE
XProjects       NONE
```

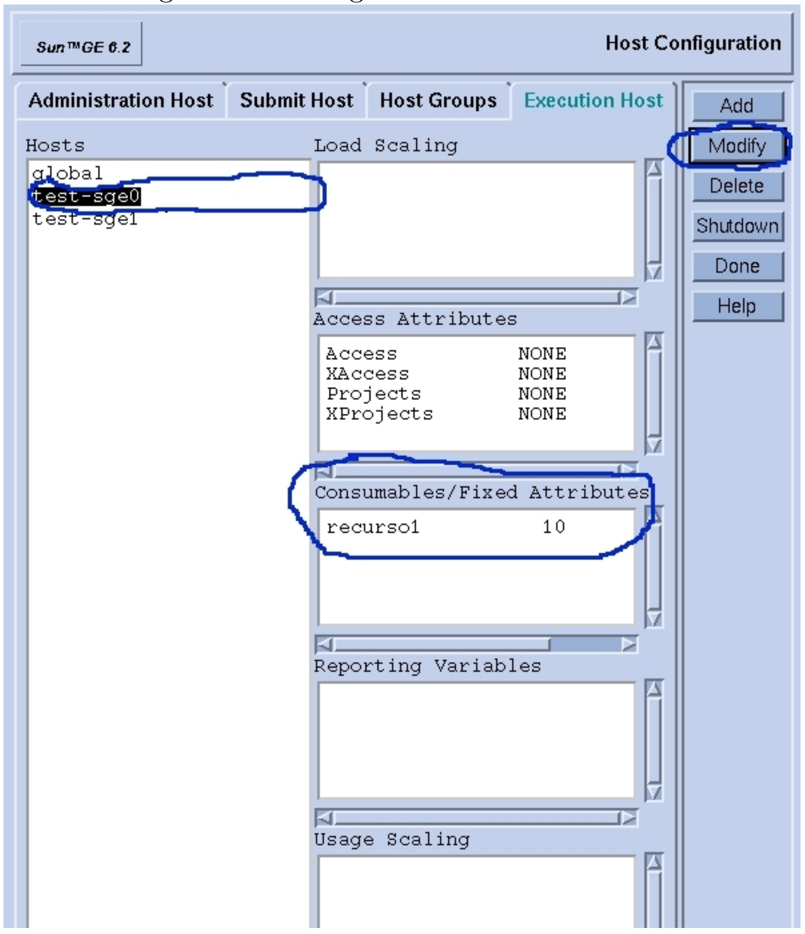
```
recurso1          10
```

Host: test-sge0

Consumables/Fixed Attributes

Name	Value
recurso1	10

Figure 3.40: Assign a resource to a host II

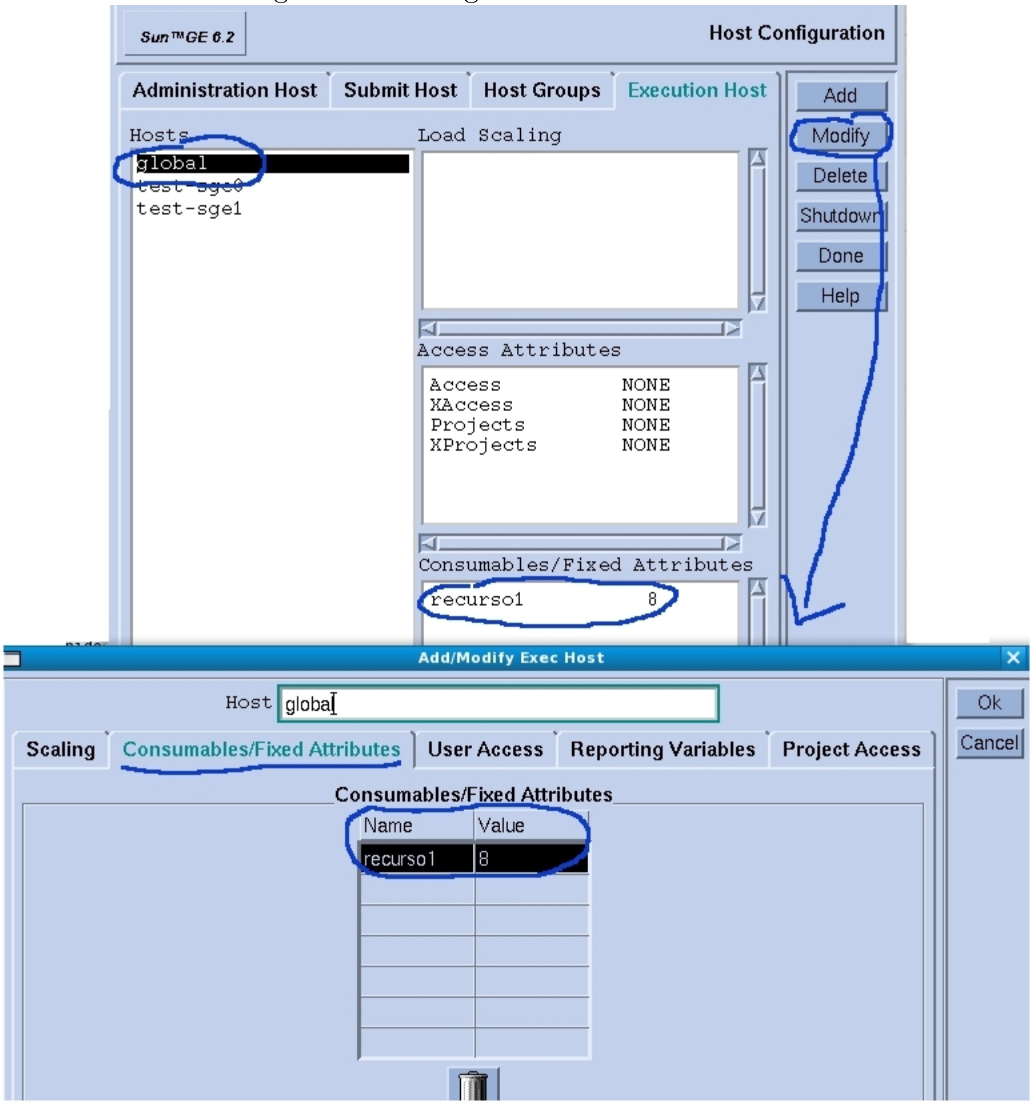


```
Hosts
global
test-sge0
test-sge1
```

```
Access      NONE
XAccess     NONE
Projects    NONE
XProjects   NONE
```

```
recurso1         10
```

Figure 3.41: Assign a resource to a cluster



Hosts
```
global
test-sge0
test-sge1
```

Access Attributes
```
Access          NONE
XAccess         NONE
Projects        NONE
XProjects       NONE
```

Consumables/Fixed Attributes
```
recurso1              8
```

Host global

Consumables/Fixed Attributes

Name	Value
recurso1	8

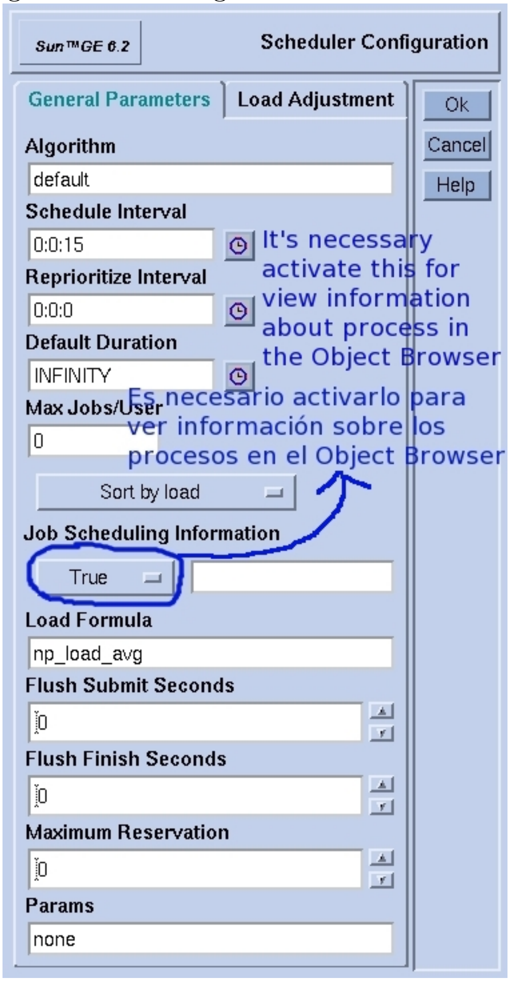
- Firstable the satisfy of the requeriments at cluster level are checked, after requeriments at host level, and finally requeriments at queue level, only if acomplish all the levels of requeriments the process will be launched.
- The resources at cluster level are therefore the more restricted arguments , this means if this requeriments are not acomplished , the next levels are not checked. The resources at cluster level are common for all the hosts that belongs to the cluster and are distributed between them
- Some of the more used predefined resources are:

```
h_fsize  : Disk free space
     ej:   h_fsize 1G

virtual_free : Virtual Memory free
      ej:   virtual_free  500M   o 2G ....
```

3.15.7 Configuration of the Scheduler

With the target of can get aditional information about the state of the process , is needed active an option in the configuration of the scheduler, in the button "Scheduler Configuration" of the window "Main control" as is explained in the figure 3.42.

Figure 3.42: Configuration of the scheduler



Sun™GE 6.2 Scheduler Configuration

General Parameters | Load Adjustment

Ok | Cancel | Help

Algorithm: default

Schedule Interval: 0:0:15

Reprioritize Interval: 0:0:0

Default Duration: INFINITY

Max Jobs/User: 0

Sort by load

Job Scheduling Information: True

Load Formula: np_load_avg

Flush Submit Seconds: 0

Flush Finish Seconds: 0

Maximum Reservation: 0

Params: none

It's necessary activate this for view information about process in the Object Browser

Es necesario activarlo para ver información sobre los procesos en el Object Browser

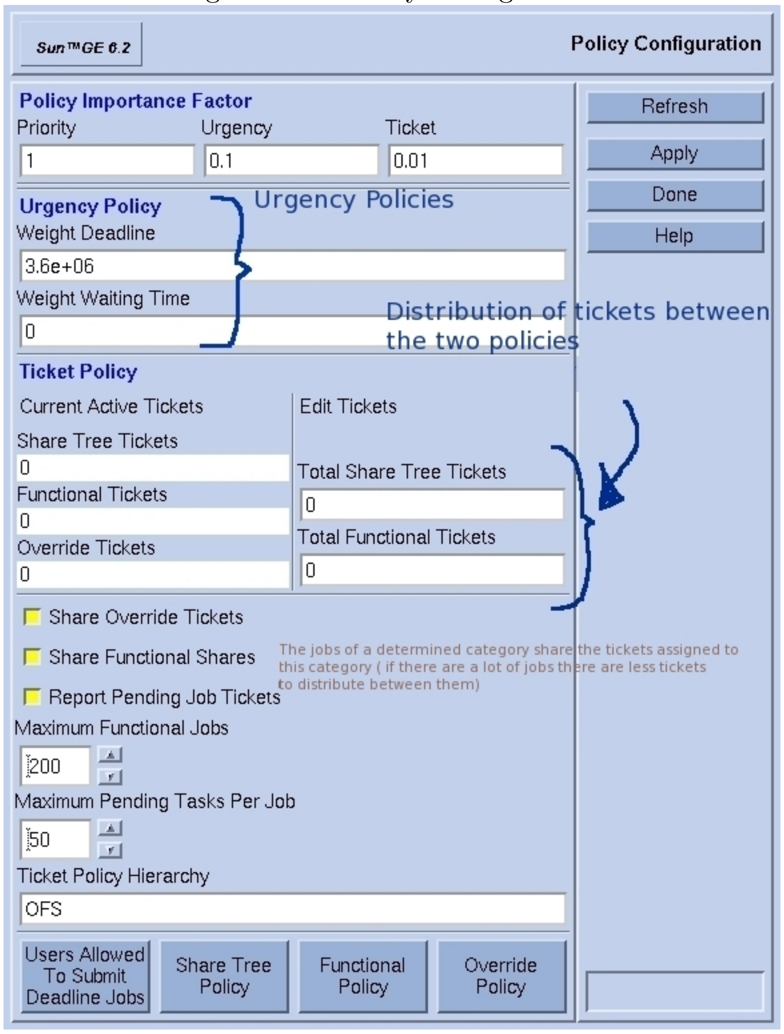
3.15.8 Configuration of policies

Should be assigned the double of tickets to the Shared-based policy than the Functional Policiy, this makes that the system seems more a store of quotas of access to resources.

The total number of tickets doesn't has a special meaning, the important are the relations between the policies . A High number of tickets allow us establish the relations with more accuracy.

You can see the configuration of the Share Tree Policy in teh figure 3.44 and the configuration of the Functional Policy in the figure 3.45

Figure 3.43: Policy Configuration



Sun™GE 6.2

Policy Configuration

Policy Importance Factor

Priority	Urgency	Ticket
1	0.1	0.01

Urgency Policy

Weight Deadline: 3.6e+06

Weight Waiting Time: 0

Urgency Policies

Distribution of tickets between the two policies

Ticket Policy

Current Active Tickets

Share Tree Tickets: 0

Functional Tickets: 0

Override Tickets: 0

Edit Tickets

Total Share Tree Tickets: 0

Total Functional Tickets: 0

Share Override Tickets

Share Functional Shares

Report Pending Job Tickets

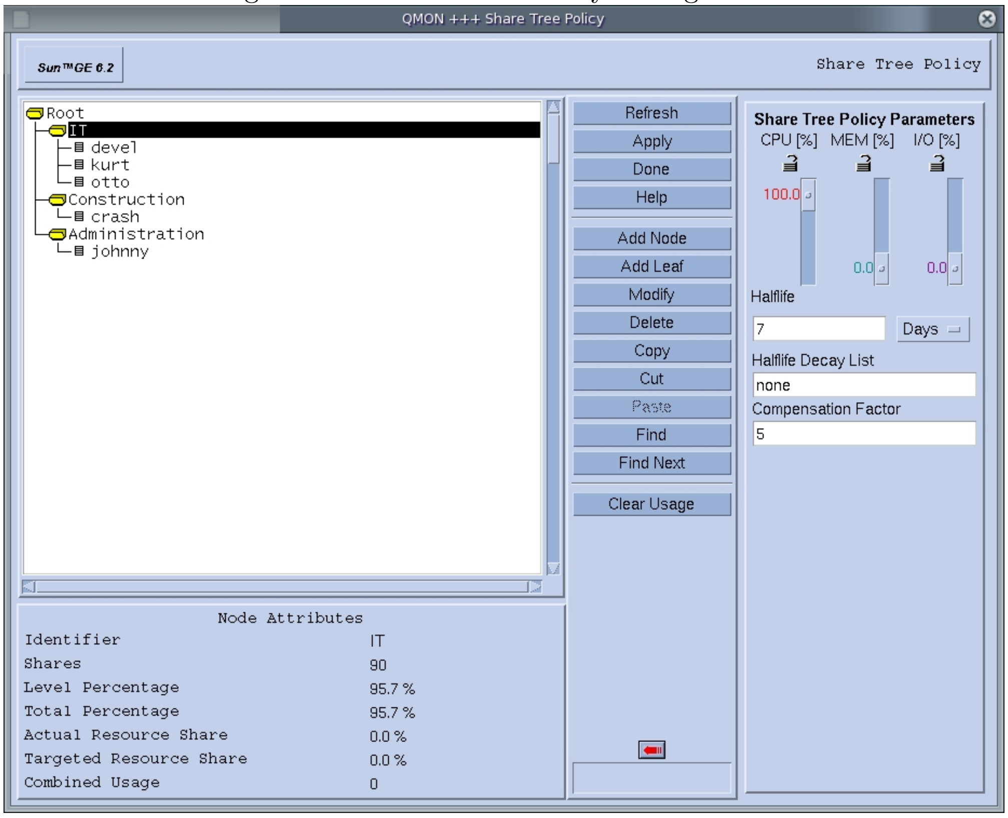
The jobs of a determined category share the tickets assigned to this category (if there are a lot of jobs there are less tickets to distribute between them)

Maximum Functional Jobs: 200

Maximum Pending Tasks Per Job: 50

Ticket Policy Hierarchy: OFS

Figure 3.44: Share-Tree Policy Configuration



```
Root
├IT
│├devel
│├kurt
│└otto
├Construction
│└crash
└Administration
 └johnny
```

Node Attributes

Identifier	IT
Shares	90
Level Percentage	95.7 %
Total Percentage	95.7 %
Actual Resource Share	0.0 %
Targeted Resource Share	0.0 %
Combined Usage	0

Share Tree Policy Parameters

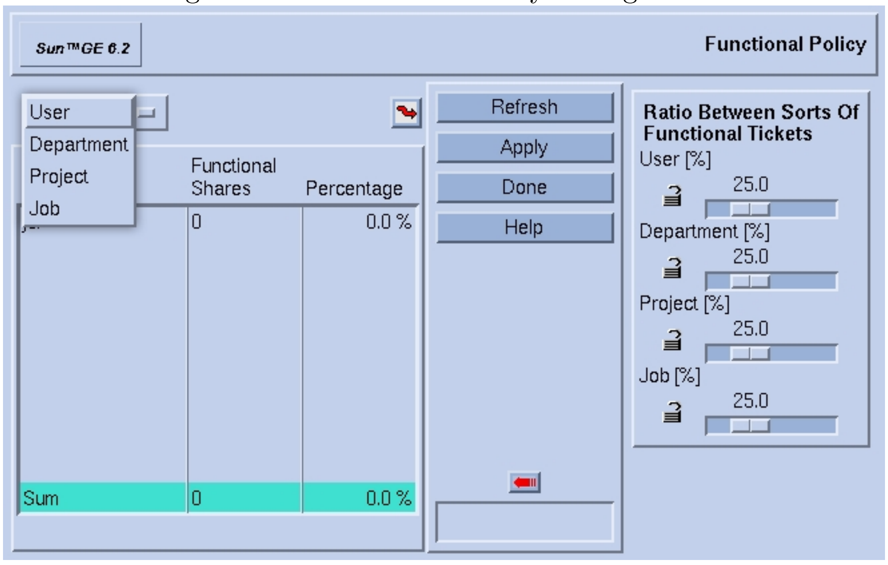
CPU [%]	MEM [%]	I/O [%]
100.0	0.0	0.0

Halflife: 7 Days

Halflife Decay List: none

Compensation Factor: 5

Figure 3.45: Functional Policy Configuration



	Functional Shares	Percentage
	0	0.0 %
Sum	0	0.0 %

Ratio Between Sorts Of Functional Tickets

User [%] 25.0

Department [%] 25.0

Project [%] 25.0

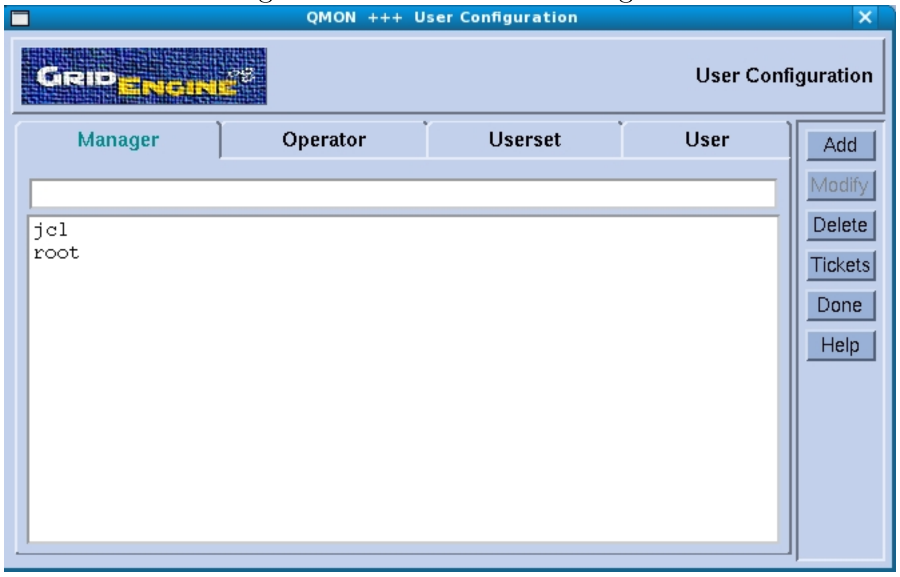
Job [%] 25.0

3.15.9 Configuration of projects and users

In the next figures 3.46 and 3.47 you can see how to add users for administration issues o simply for can use the grid:

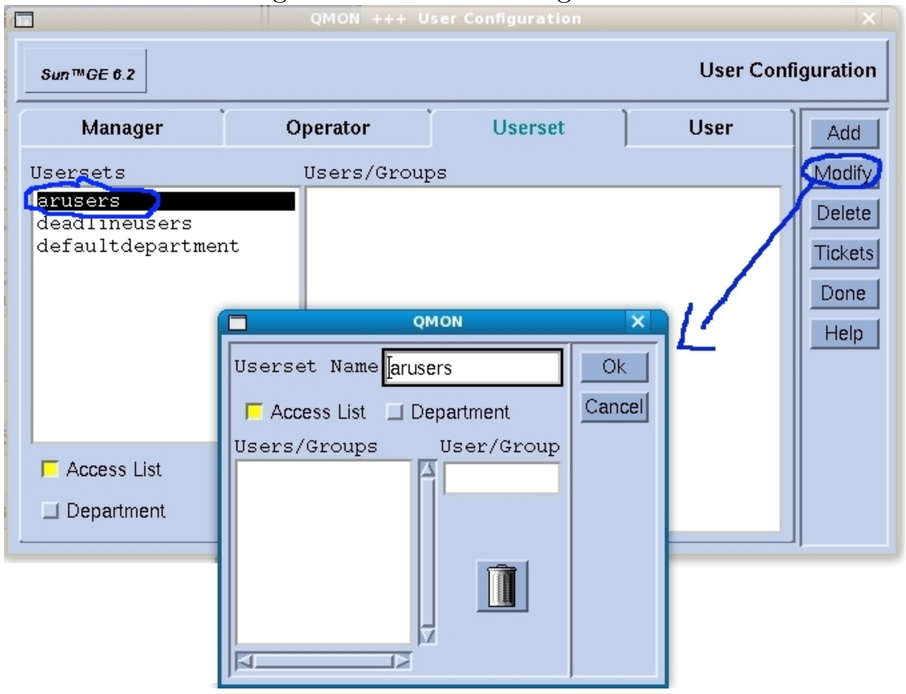
In the figure 3.48 yo can see how to add / modify / delete or in general manage projects in the grid.

Figure 3.46: Add users to the grid



```
jcl
root
```

Figure 3.47: Users Configuration

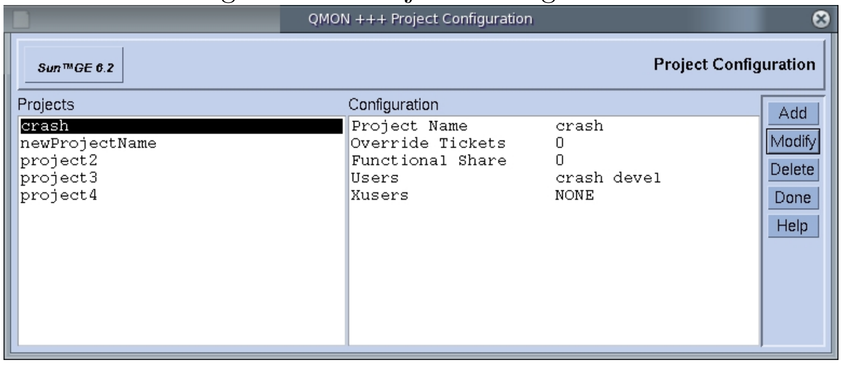


```
arusers
deadlineusers
defaultdepartment
```

Userset Name arusers

[✓ Access List] Department

Figure 3.48: Projects Configuration



Projects

```
crash
newProjectName
project2
project3
project4
```

Configuration

```
Project Name         crash
Override Tickets     0
Functional Share     0
Users                crash devel
Xusers               NONE
```

3.16 Grid operation and administration

3.16.1 Hosts control

We can monitorize the hosts with the commands qhost and qacct:

qhost: shows the list of all hosts:

```
HOSTNAME                ARCH         NCPU  LOAD  MEMTOT  MEMUSE  SWAPTO  SWAPUS
-------------------------------------------------------------------------------
global                  -               -     -       -       -       -       -
test-sge                lx24-amd64      1  0.00    1.0G  166.2M    2.0G     0.0
```

qhost -q shows the list of all hosts and the queues of each one:

```
[root@test-sge lx24-amd64]# ./qhost -q
HOSTNAME                ARCH         NCPU  LOAD  MEMTOT  MEMUSE  SWAPTO  SWAPUS
-------------------------------------------------------------------------------
global                  -               -     -       -       -       -       -
test-sge                lx24-amd64      1  0.00    1.0G  166.2M    2.0G     0.0
    all.q                BIP   0/0/1
```

qacct -h host : Use of a host:

```
[jcl@deltasub log]$ qacct -h delta0
HOST                   WALLCLOCK    UTIME      STIME       CPU         MEMORY        IO       IOW
====================================================================================================
delta0  1246303    191578.926   6380.799   197983.303   181289.673    0.000    0.000
```

3.16.2 Jobs control

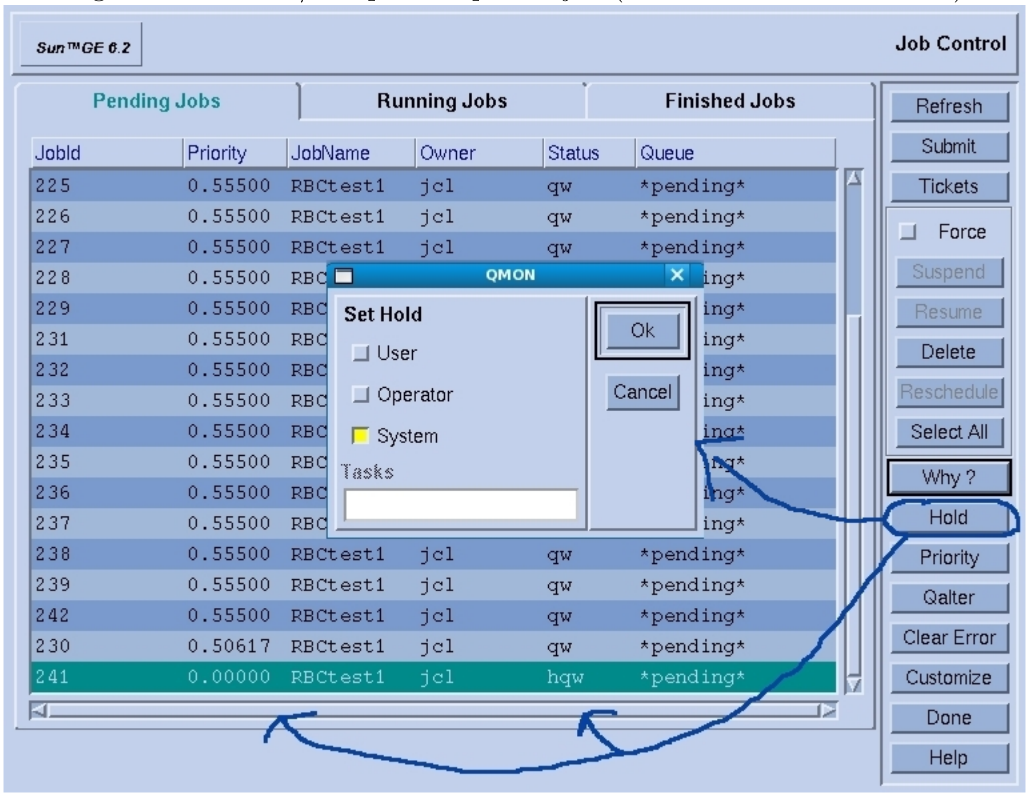
In the tab **"Pending Jobs"** appears a list of jobs that are queued (waiting to enter in execution) sorted by priority, left up the process with more priority and therefore those that will be executed first.

For block or suspend (left in Pause) a queued process for that doesn't enter in execution , we must select it and pulse the button **"Hold"** (in Pendings Jobs tab) , in this moment the process will be at the end of the jobs list in "hqw" state and with a priority of 0.00000 (see figure 3.49)

We can see it also in command line:

```
qstat -j <jobid> : Show information of a process NOT FINISHED YET
qacct -j <jobid> : Show information of a process ALREADY FINISHED
```

Figure 3.49: Hold / Suspend a queued job (doesn't enter in execution)



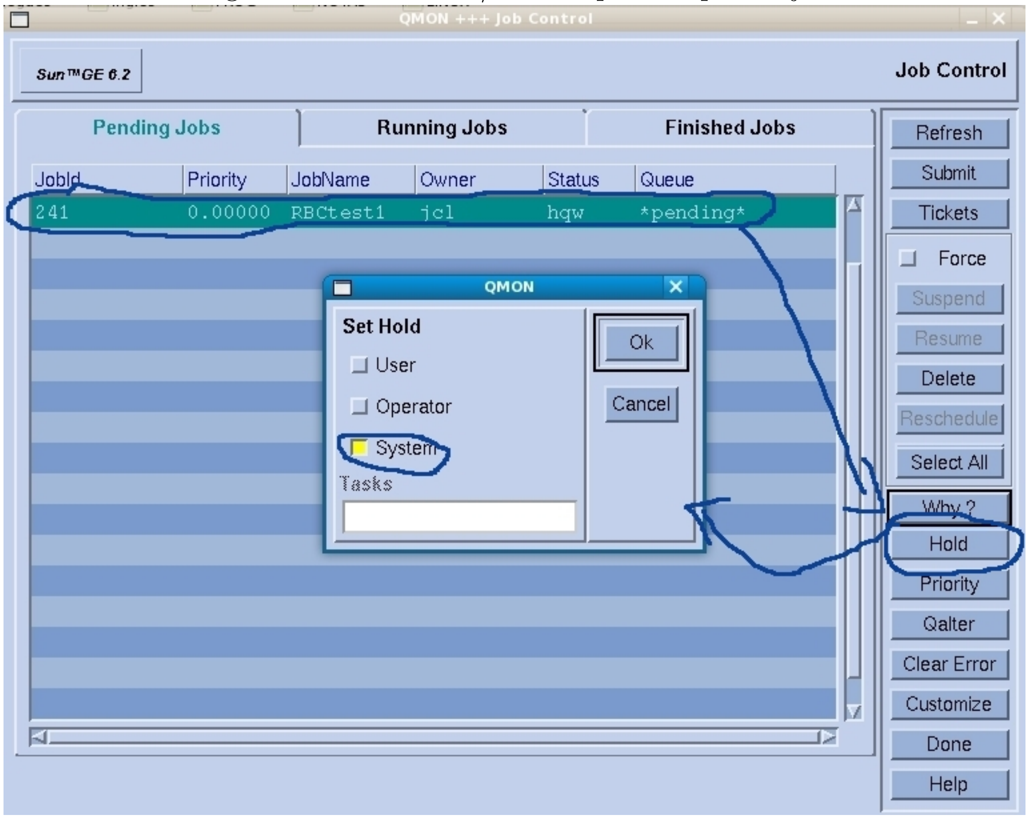
JobId	Priority	JobName	Owner	Status	Queue
225	0.55500	RBCtest1	jcl	qw	*pending*
226	0.55500	RBCtest1	jcl	qw	*pending*
227	0.55500	RBCtest1	jcl	qw	*pending*
228	0.55500	RBC			ing*
229	0.55500	RBC			ing*
231	0.55500	RBC			ing*
232	0.55500	RBC			ing*
233	0.55500	RBC			ing*
234	0.55500	RBC			ing*
235	0.55500	RBC			ng*
236	0.55500	RBC			ing*
237	0.55500	RBC			ing*
238	0.55500	RBCtest1	jcl	qw	*pending*
239	0.55500	RBCtest1	jcl	qw	*pending*
242	0.55500	RBCtest1	jcl	qw	*pending*
230	0.50617	RBCtest1	jcl	qw	*pending*
241	0.00000	RBCtest1	jcl	hqw	*pending*

```
[jcl@test-sge1 lx24-amd64]$ ./qstat
job-ID  prior   name       user         state submit/start at     queue                          slots ja-task-ID
-----------------------------------------------------------------------------------------------------------------
    227 0.55500 RBCtest1   jcl          r     04/20/2010 17:07:57 b@test-sge0         1
    228 0.55500 RBCtest1   jcl          r     04/20/2010 17:07:57 b@test-sge1         1
    229 0.55500 RBCtest1   jcl          qw    04/20/2010 16:36:21                                    1
    231 0.55500 RBCtest1   jcl          qw    04/20/2010 16:36:24                                    1
    232 0.55500 RBCtest1   jcl          qw    04/20/2010 16:36:24                                    1
    233 0.55500 RBCtest1   jcl          qw    04/20/2010 16:36:24                                    1
    234 0.55500 RBCtest1   jcl          qw    04/20/2010 16:36:24                                    1
    235 0.55500 RBCtest1   jcl          qw    04/20/2010 16:36:24                                    1
    236 0.55500 RBCtest1   jcl          qw    04/20/2010 16:36:25                                    1
    237 0.55500 RBCtest1   jcl          qw    04/20/2010 16:36:25                                    1
    238 0.55500 RBCtest1   jcl          qw    04/20/2010 16:36:25                                    1
    239 0.55500 RBCtest1   jcl          qw    04/20/2010 16:37:14                                    1
    242 0.55500 RBCtest1   jcl          qw    04/20/2010 16:37:17                                    1
    230 0.50617 RBCtest1   jcl          qw    04/20/2010 16:36:24                                    1
    241 0.00000 RBCtest1   jcl          hqw   04/20/2010 16:37:17                                    1
```

I can get information of this hold job executing the next command (In the last line we can see the current status of the job)

```
[jcl@test-sge1 lx24-amd64]$ ./qstat -j  241
==============================================================
job_number:                 241
exec_file:                  job_scripts/241
submission_time:            Tue Apr 20 16:37:17 2010
owner:                      jcl
uid:                        500
group:                      sag
gid:                        500
sge_o_home:                 /home/jcl
sge_o_log_name:             jcl
sge_o_path:                 /opt/sungrid/sge6_2u5/bin/lx24-amd64:/usr/kerberos/bin:/usr/local/bin:/bin:/usr/bin:/home/jcl/bi
```

Figure 3.50: Un-Hold / Un-Suspend a queued job



```
JobId   Priority  JobName   Owner  Status  Queue
241     0.00000   RBCtest1  jcl    hqw     *pending*
```

```
sge_o_shell:                /bin/bash
sge_o_workdir:              /produccion/SCRIPTS
sge_o_host:                 test-sge0
account:                    sge
merge:                      y
hard resource_list:         recurso1=4
mail_options:               aes
mail_list:                  antonio.carmona@-is.es
notify:                     FALSE
job_name:                   RBCtest1
stdout_path_list:           NONE:NONE:/produccion/SCRIPTS/log
jobshare:                   0
hard_queue_list:            b
env_list:
script_file:                /produccion/SCRIPTS/test1-sungrid.sh
version:                    1
scheduling info:            Job is in hold state
```

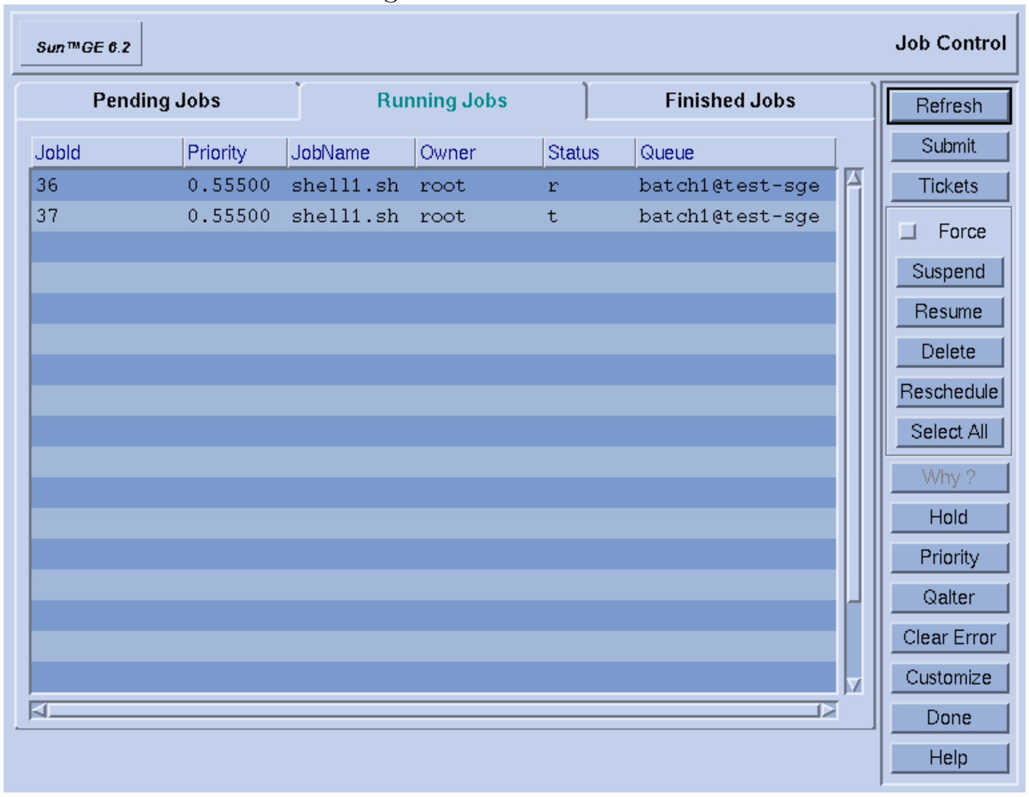
For left the job un-hold for be considered for enter in execution we must pulse another time in the Hold button and unselect all the options as explained in the figure 3.50

In the tab **"Running Jobs"** appears the jobs in execution in the current moment (see figure 3.51)

The button **"Suspend/Resume"** allow us suspend and resume a process that is currently in execution (is not in queued stated) (see figure 3.51)

In figure 3.52 (Job Control: How to get explanation of current status of a job), if we pulse the button **"Why"** we will get information about the current status of a job and the reasons of this state.

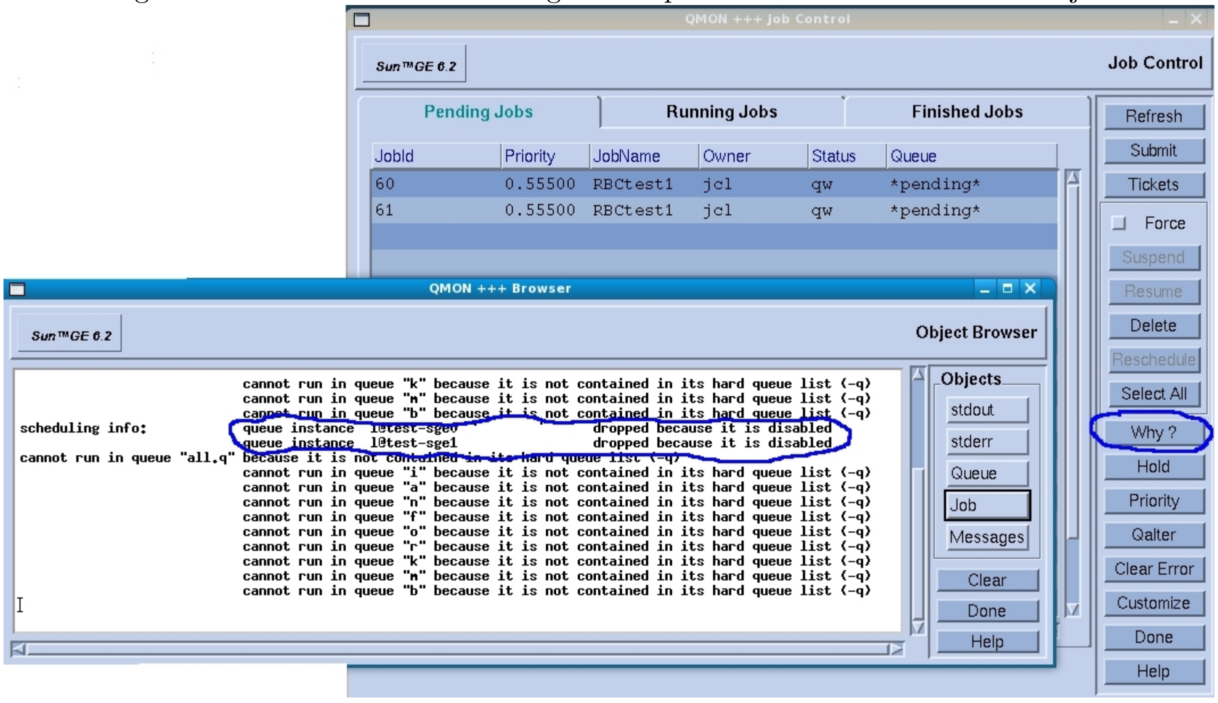
Figure 3.51: Job Control



JobId	Priority	JobName	Owner	Status	Queue
36	0.55500	shell1.sh	root	r	batch1@test-sge
37	0.55500	shell1.sh	root	t	batch1@test-sge

For can visualize this information in the **"Object Browser"** window, we must configure properly the Scheduler (see figure 3.42 Configuration of the Scheduler)

Figure 3.52: Job Control: How to get an explanation of current status of a job



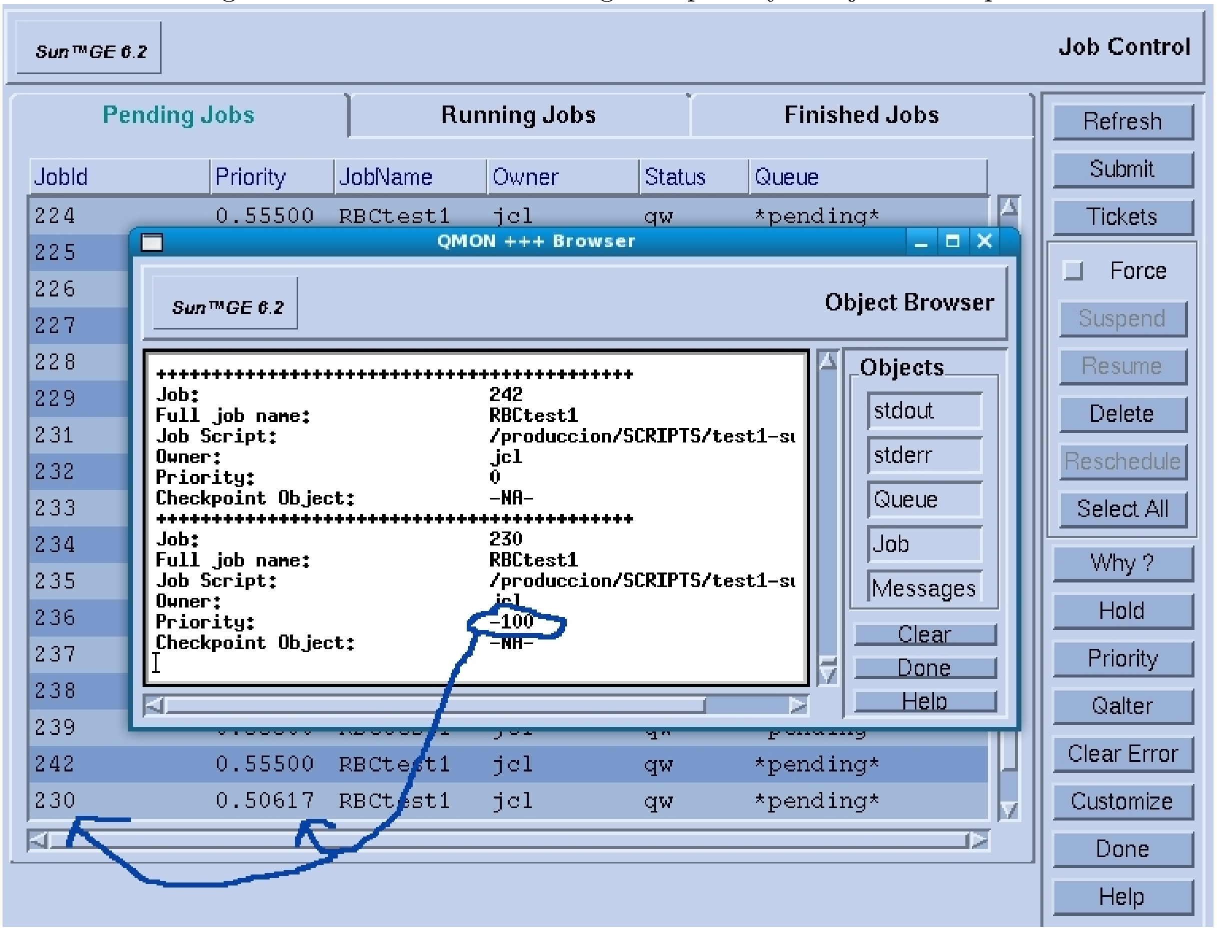
JobId	Priority	JobName	Owner	Status	Queue
60	0.55500	RBCtest1	jcl	qw	*pending*
61	0.55500	RBCtest1	jcl	qw	*pending*

```
                          cannot run in queue "k" because it is not contained in its hard queue list (-q)
                          cannot run in queue "m" because it is not contained in its hard queue list (-q)
                          cannot run in queue "b" because it is not contained in its hard queue list (-q)
scheduling info:          queue instance  l@test-sge0                  dropped because it is disabled
                          queue instance  l@test-sge1                  dropped because it is disabled
cannot run in queue "all.q" because it is not contained in its hard queue list (-q)
                          cannot run in queue "i" because it is not contained in its hard queue list (-q)
                          cannot run in queue "a" because it is not contained in its hard queue list (-q)
                          cannot run in queue "n" because it is not contained in its hard queue list (-q)
                          cannot run in queue "f" because it is not contained in its hard queue list (-q)
                          cannot run in queue "o" because it is not contained in its hard queue list (-q)
                          cannot run in queue "r" because it is not contained in its hard queue list (-q)
                          cannot run in queue "k" because it is not contained in its hard queue list (-q)
                          cannot run in queue "m" because it is not contained in its hard queue list (-q)
                          cannot run in queue "b" because it is not contained in its hard queue list (-q)
```

We can change the priority of a queued process for advance/delay its entrance in execution, we must select a process and pulse the button **"Priority"** and put a new value for the process. (see figure 3.53)

After change the priority of a job we will see how SGE realign the job list adjusting to the news priorities, in the next example we put a priority lower than the rest (-100) to the process number 230 and for this reason its appears at the end of the jobs list . In the **"Object browser"** window we can see also the priority of the job. (figure 3.53)

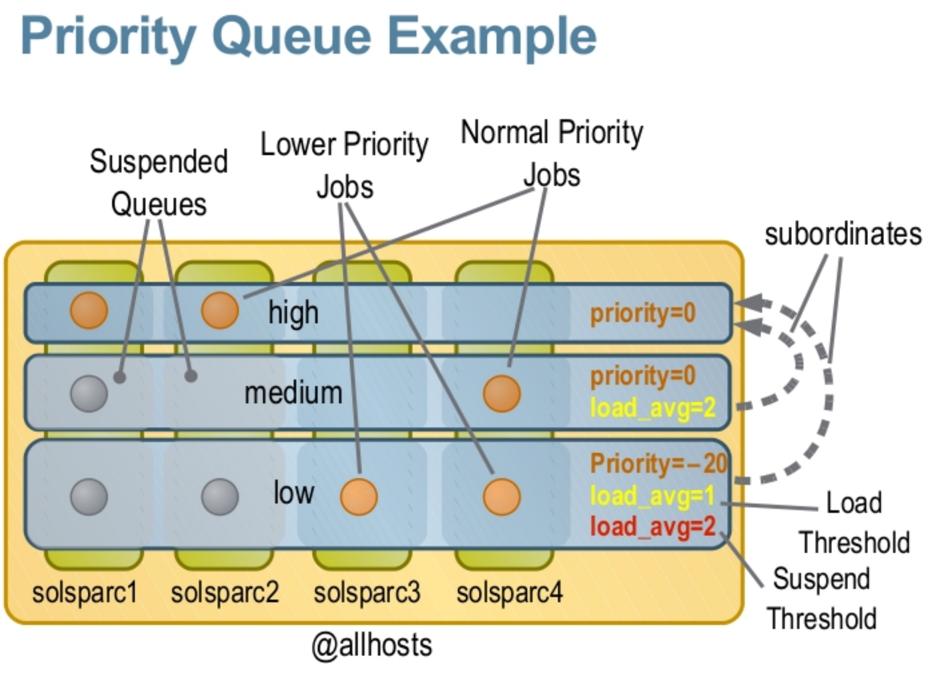
Figure 3.53: Job Control: Change the priority of a job. Example



```
JobId      Priority  JobName   Owner   Status  Queue
224        0.55500   RBCtest1  jcl     qw      *pending*
225
226
227
228
229
231
232
233
234
235
236
237
238
239
242        0.55500   RBCtest1  jcl     qw      *pending*
230        0.50617   RBCtest1  jcl     qw      *pending*
```

```
++++++++++++++++++++++++++++++++++++++++++
Job:                          242
Full job name:                RBCtest1
Job Script:                   /produccion/SCRIPTS/test1-su
Owner:                        jcl
Priority:                     0
Checkpoint Object:            -NA-
++++++++++++++++++++++++++++++++++++++++++
Job:                          230
Full job name:                RBCtest1
Job Script:                   /produccion/SCRIPTS/test1-su
Owner:                        jcl
Priority:                     -100
Checkpoint Object:            -NA-
```

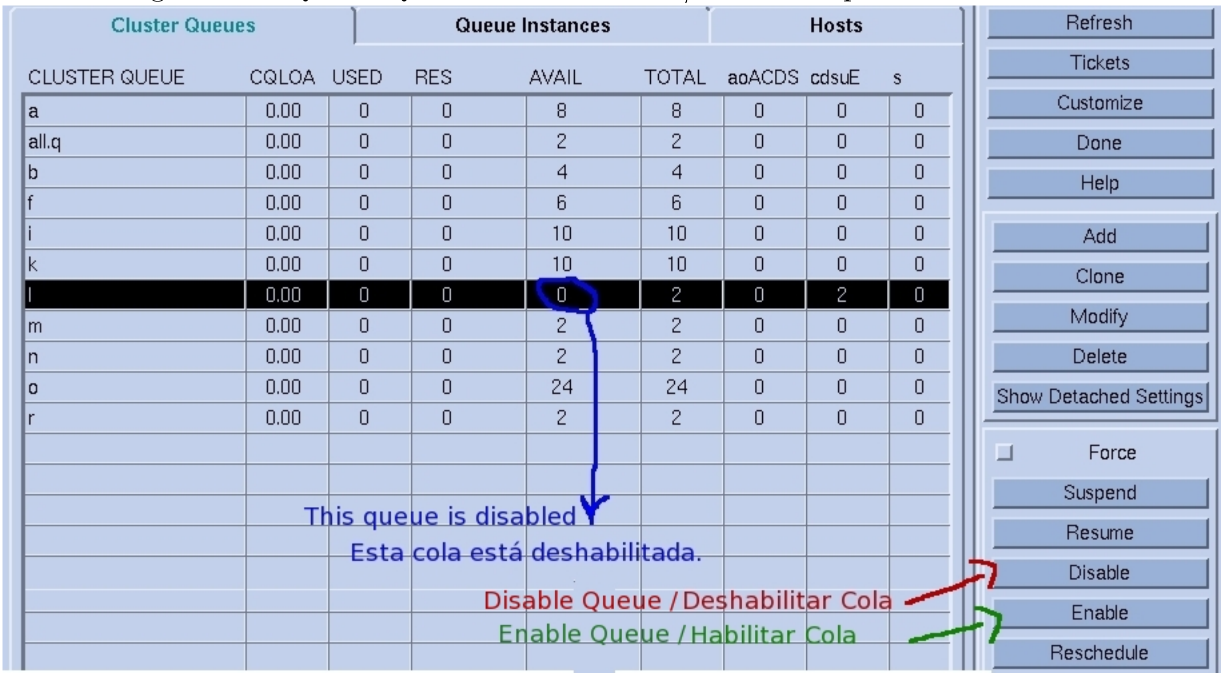
Figure 3.54: Jobs priority [1]



3.16.3 Queue control

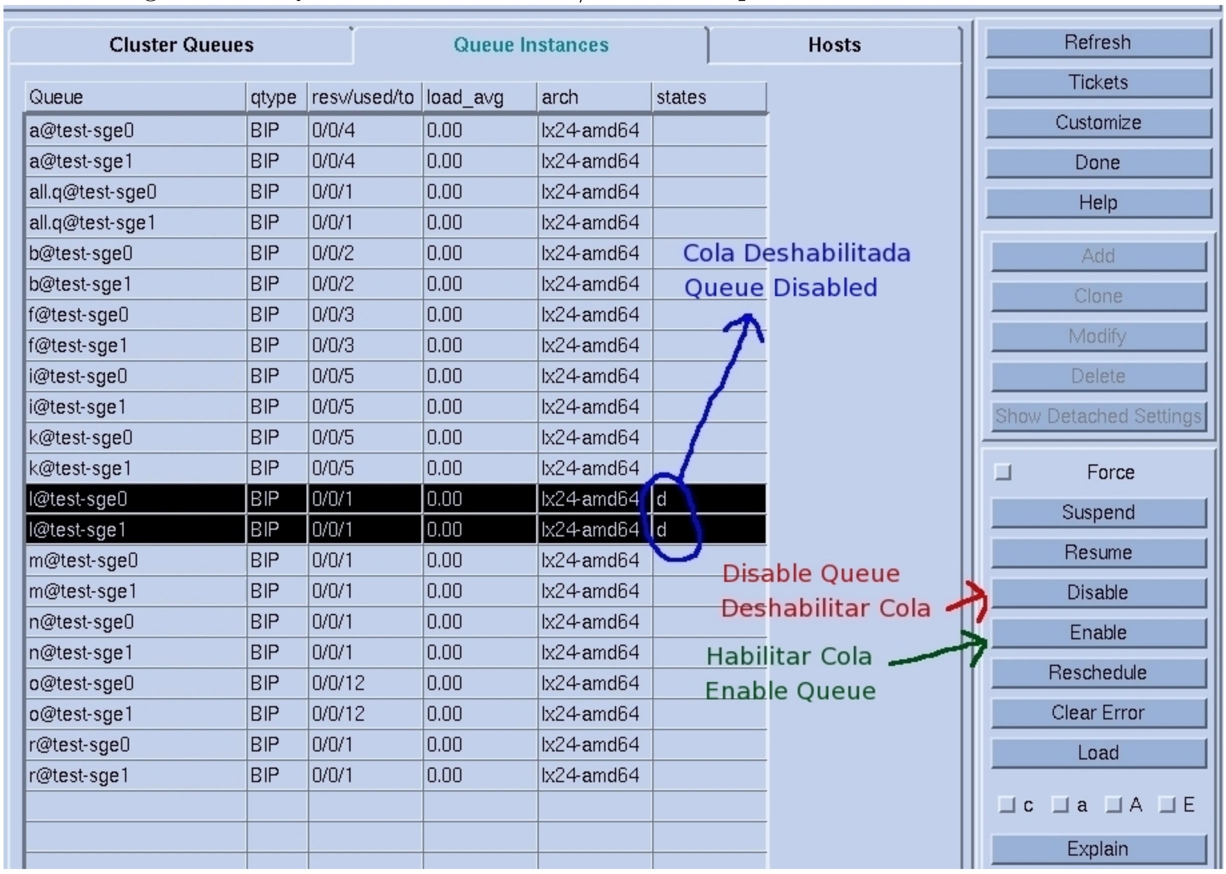
In the figure 3.55 we can see how to Enable / Disable a queue in ALL the nodes, in the figure 3.56 we see who to Enable / Disable a queue in ONE or more nodes

Figure 3.55: Qmon: Queue control: Enable / Disable a queue in ALL nodes



CLUSTER QUEUE	CQLOA	USED	RES	AVAIL	TOTAL	aoACDS	cdsuE	s
a	0.00	0	0	8	8	0	0	0
all.q	0.00	0	0	2	2	0	0	0
b	0.00	0	0	4	4	0	0	0
f	0.00	0	0	6	6	0	0	0
i	0.00	0	0	10	10	0	0	0
k	0.00	0	0	10	10	0	0	0
l	0.00	0	0	0	2	0	2	0
m	0.00	0	0	2	2	0	0	0
n	0.00	0	0	2	2	0	0	0
o	0.00	0	0	24	24	0	0	0
r	0.00	0	0	2	2	0	0	0

Figure 3.56: Queue control: Enable / Disable a queue in ONE or more nodes



Queue	qtype	resv/used/to	load_avg	arch	states
a@test-sge0	BIP	0/0/4	0.00	lx24-amd64	
a@test-sge1	BIP	0/0/4	0.00	lx24-amd64	
all.q@test-sge0	BIP	0/0/1	0.00	lx24-amd64	
all.q@test-sge1	BIP	0/0/1	0.00	lx24-amd64	
b@test-sge0	BIP	0/0/2	0.00	lx24-amd64	
b@test-sge1	BIP	0/0/2	0.00	lx24-amd64	
f@test-sge0	BIP	0/0/3	0.00	lx24-amd64	
f@test-sge1	BIP	0/0/3	0.00	lx24-amd64	
i@test-sge0	BIP	0/0/5	0.00	lx24-amd64	
i@test-sge1	BIP	0/0/5	0.00	lx24-amd64	
k@test-sge0	BIP	0/0/5	0.00	lx24-amd64	
k@test-sge1	BIP	0/0/5	0.00	lx24-amd64	
l@test-sge0	BIP	0/0/1	0.00	lx24-amd64	d
l@test-sge1	BIP	0/0/1	0.00	lx24-amd64	d
m@test-sge0	BIP	0/0/1	0.00	lx24-amd64	
m@test-sge1	BIP	0/0/1	0.00	lx24-amd64	
n@test-sge0	BIP	0/0/1	0.00	lx24-amd64	
n@test-sge1	BIP	0/0/1	0.00	lx24-amd64	
o@test-sge0	BIP	0/0/12	0.00	lx24-amd64	
o@test-sge1	BIP	0/0/12	0.00	lx24-amd64	
r@test-sge0	BIP	0/0/1	0.00	lx24-amd64	
r@test-sge1	BIP	0/0/1	0.00	lx24-amd64	

3.17 Additional graphical tools for manage and monitoring the Grid

There are some added (outside of the main project of Sun Grid Engine) graphical tools for managing, monitoring, and get statistics about the Grid, these are: Arco, Inspect and xml-qstat.

3.17.1 Arco

ARCO: Accounting and Reporting Console is a web tool for manage and store historic information about the process executions in the grid engine for a later analisys and investigation , is not a monitoring system in real time.

A process called sgedbwriter is in charge of get data and store them into a mysql dabatase.

For use Arco is needed:

- Sun Grid Engine
- Java Runtime Environment (JRE) version 1.5 or higher
- Java Web Console
- MySql
- Dbwriter
- Reporting

3.17.2 Inspect

Is Monitoring and Configuration Console, allows to monitor Sun Grid Engine clusters using the data stored by the JMX MBean server resident in que Master daemon of Sun Grid Engine and provide a flexible interface for display current and historic data of the execution of the process.

Inspect need:

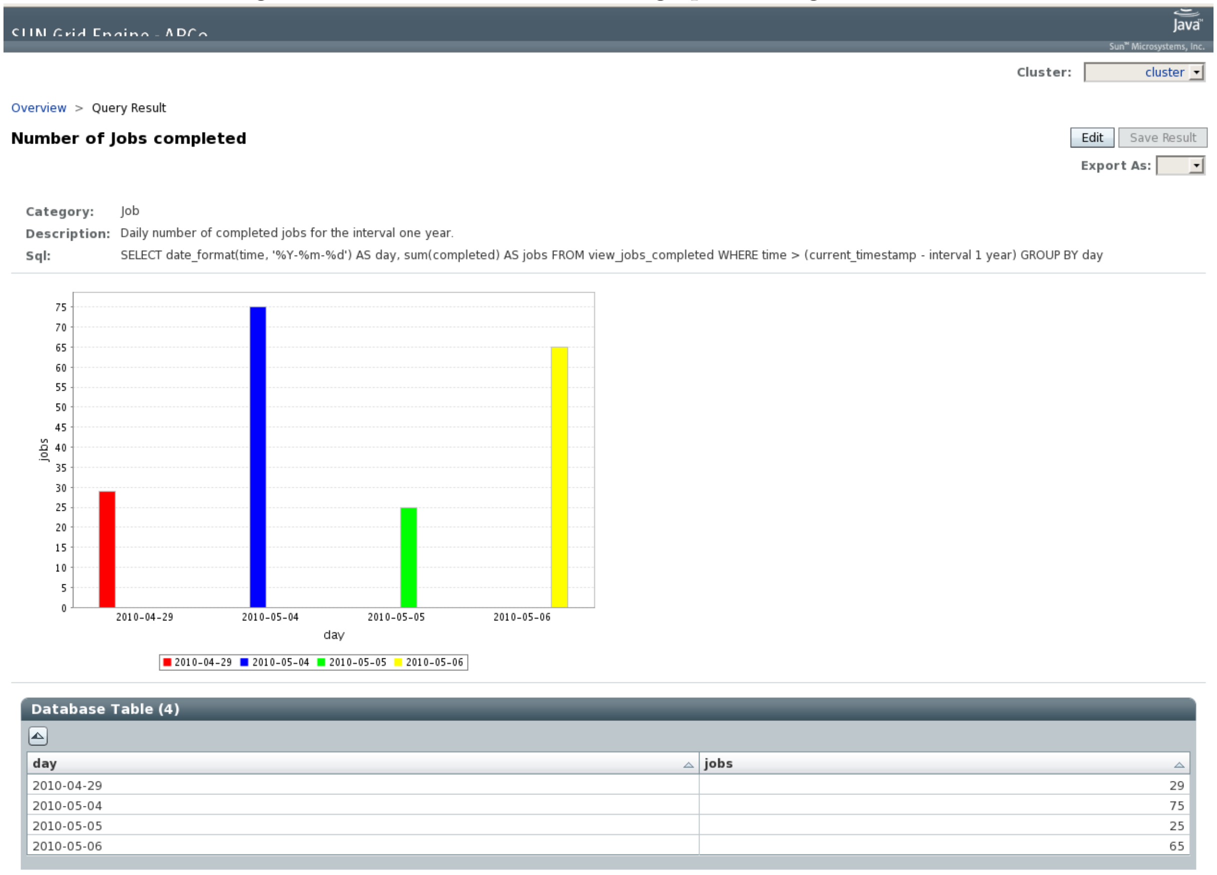
- Java JDK
- JMX MBean installed in the server Sun Grid Engine

3.17.3 xml-qstat

xml-qstat is a tool for monitoring the grid based in the framwork Apache Cocoon, for use xml-qstat we need

- Sun JDK 1.6
- Apache Tomcat
- FrameWork Apache Cocoon 2.1.11
- xml-qstat

Figure 3.57: Arco: Tool for manage processing historic data



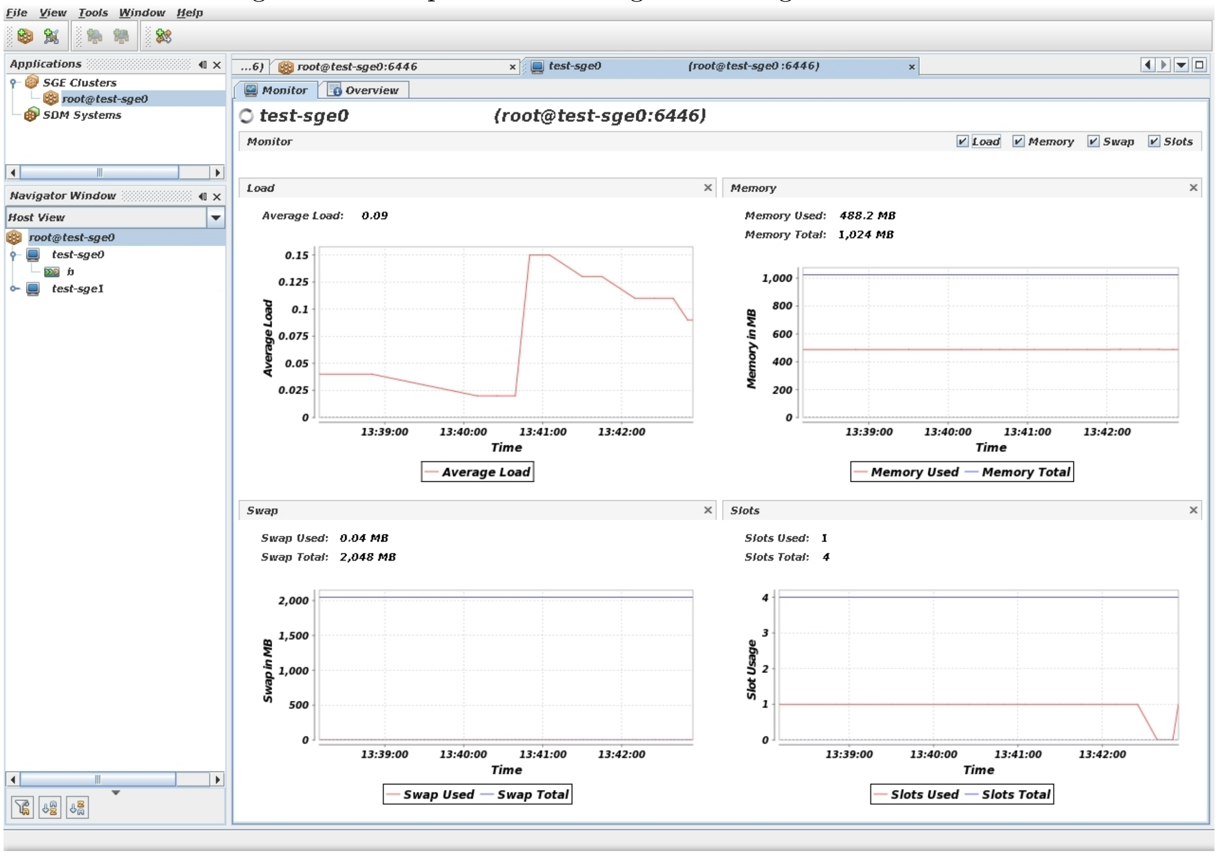
Number of Jobs completed

Category:	Job
Description:	Daily number of completed jobs for the interval one year.
Sql:	SELECT date_format(time, '%Y-%m-%d') AS day, sum(completed) AS jobs FROM view_jobs_completed WHERE time > (current_timestamp - interval 1 year) GROUP BY day

Database Table (4)

day	jobs
2010-04-29	29
2010-05-04	75
2010-05-05	25
2010-05-06	65

Figure 3.58: Inspect: Monitoring and Configuration Console



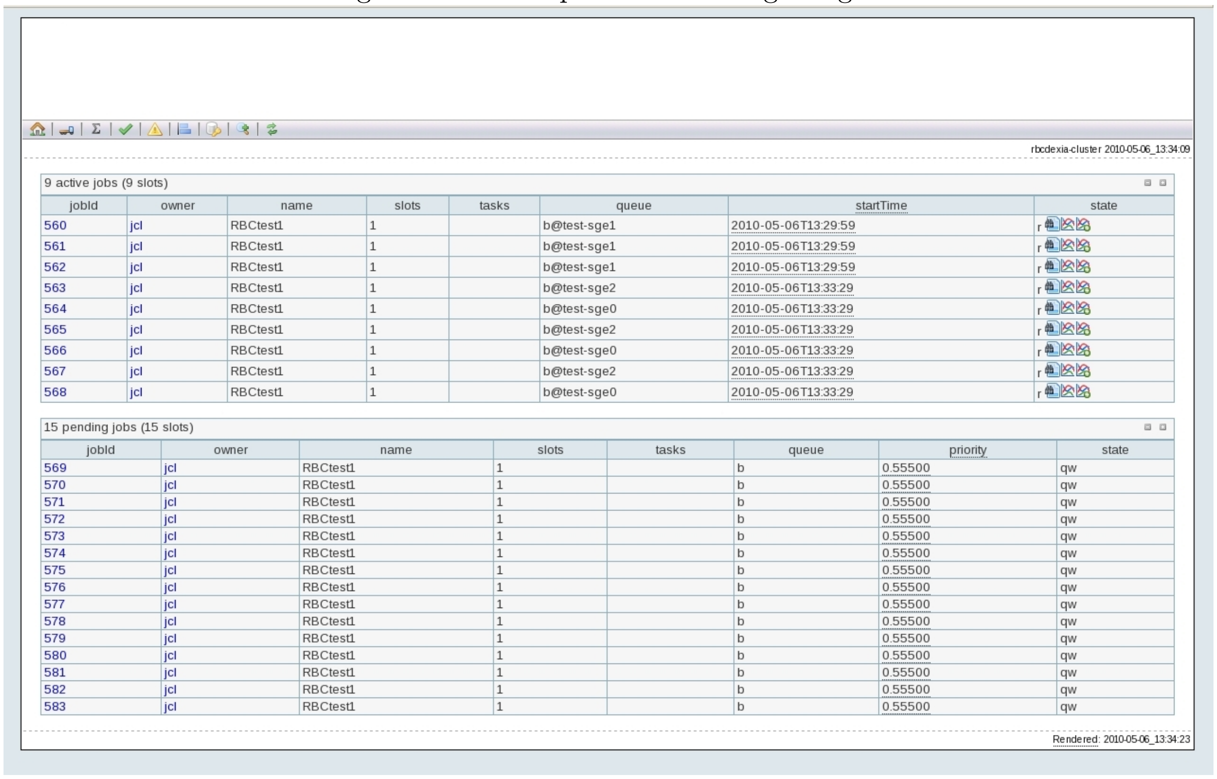
Average Load: 0.09

Memory Used: 488.2 MB
Memory Total: 1,024 MB

Swap Used: 0.04 MB
Swap Total: 2,048 MB

Slots Used: 1
Slots Total: 4

Figure 3.59: xml-qstat: Monitoring the grid



rbcdexia-cluster 2010-05-06_13:34:09

9 active jobs (9 slots)

jobId	owner	name	slots	tasks	queue	startTime	state
560	jcl	RBCtest1	1		b@test-sge1	2010-05-06T13:29:59	r
561	jcl	RBCtest1	1		b@test-sge1	2010-05-06T13:29:59	r
562	jcl	RBCtest1	1		b@test-sge1	2010-05-06T13:29:59	r
563	jcl	RBCtest1	1		b@test-sge2	2010-05-06T13:33:29	r
564	jcl	RBCtest1	1		b@test-sge0	2010-05-06T13:33:29	r
565	jcl	RBCtest1	1		b@test-sge2	2010-05-06T13:33:29	r
566	jcl	RBCtest1	1		b@test-sge0	2010-05-06T13:33:29	r
567	jcl	RBCtest1	1		b@test-sge2	2010-05-06T13:33:29	r
568	jcl	RBCtest1	1		b@test-sge0	2010-05-06T13:33:29	r

15 pending jobs (15 slots)

jobId	owner	name	slots	tasks	queue	priority	state
569	jcl	RBCtest1	1		b	0.55500	qw
570	jcl	RBCtest1	1		b	0.55500	qw
571	jcl	RBCtest1	1		b	0.55500	qw
572	jcl	RBCtest1	1		b	0.55500	qw
573	jcl	RBCtest1	1		b	0.55500	qw
574	jcl	RBCtest1	1		b	0.55500	qw
575	jcl	RBCtest1	1		b	0.55500	qw
576	jcl	RBCtest1	1		b	0.55500	qw
577	jcl	RBCtest1	1		b	0.55500	qw
578	jcl	RBCtest1	1		b	0.55500	qw
579	jcl	RBCtest1	1		b	0.55500	qw
580	jcl	RBCtest1	1		b	0.55500	qw
581	jcl	RBCtest1	1		b	0.55500	qw
582	jcl	RBCtest1	1		b	0.55500	qw
583	jcl	RBCtest1	1		b	0.55500	qw

Rendered: 2010-05-06_13:34:23

3.18 API DRMAA

DRMAA (Distributed Resource Management Application API) is an programming API for differents languages like : C, Java, or Ruby. For Ruby, the library is a simple file called drmaa.rb, before use it, is need modify the library with the right path of the installation of Sun Grid Engine:

```
# do dlopen() and initialize symbols

         def DRMAA.dopen
      ------>   'pathLib="/opt/sungrid/sge6_2u5/lib/lx24-amd64"'
                 if @libdrmaa.nil?
                         osname = `uname`.strip!
                         case osname
                         when "Darwin"
                                 libext = ".dylib"
                         else
                                 libext = ".so"
                         end
      ------->          '@libdrmaa = DL.dlopen(pathLib+'/libdrmaa' + libext)'

                         @drmaa_version = @libdrmaa['drmaa_version', 'IiisI']
                         @drmaa_get_drm_system = @libdrmaa['drmaa_get_DRM_system', 'IsIsI']
                         @drmaa_get_contact = @libdrmaa['drmaa_get_contact', 'IsIsI']
```

3.18.1 Launch a process

Example of how to launch a process with the API DRMAA:

```
require '/lib/drmaa'
begin
  session = DRMAA::Session.new
  jobSge = Job.new
  jobSge.command=command
  jobSge.arg=arguments
  jobSge.join=true
  jobSge.mail=MailList
  jobSge.stdout=":#{gridOutputFile}"
  gqsubLaunchOptions="-m as -q #{inputQueue} -l #{inputResourceRequeriments}"
  jobSge.set("drmaa_native_specification","#{gqsubLaunchOptions}")
  jobSge.set("drmaa_job_name",inputNameId)
rescue Exception => e
  ## Error comunicationg with Master Host
  exit 1
end

### Launching the process.
begin
  jobid = session.run(jobSge)
rescue Exception => e
  ### ERROR trying to queue the process
  exit 1
end
```

3.18.2 Get the status of a process

Example of how to get the status of a process with the API DRMAA:

```
require '/lib/drmaa'
session = DRMAA::Session.new
begin
   status=session.job_ps(jobid)
rescue DRMAA::DRMAACommunicationError  => errorSGE
   return -1
rescue DRMAA::DRMAAException => errorSGE
   return -1
end
```

Available states of a job returned by the drmaa_job_ps function:

```
# drmaa_job_ps() constants
STATE_UNDETERMINED          = 0x00
STATE_QUEUED_ACTIVE         = 0x10
STATE_SYSTEM_ON_HOLD        = 0x11
STATE_USER_ON_HOLD          = 0x12
STATE_USER_SYSTEM_ON_HOLD   = 0x13
STATE_RUNNING               = 0x20
STATE_SYSTEM_SUSPENDED      = 0x21
STATE_USER_SUSPENDED        = 0x22
STATE_USER_SYSTEM_SUSPENDED = 0x23
STATE_DONE                  = 0x30
STATE_FAILED                = 0x40
```

3.19 License

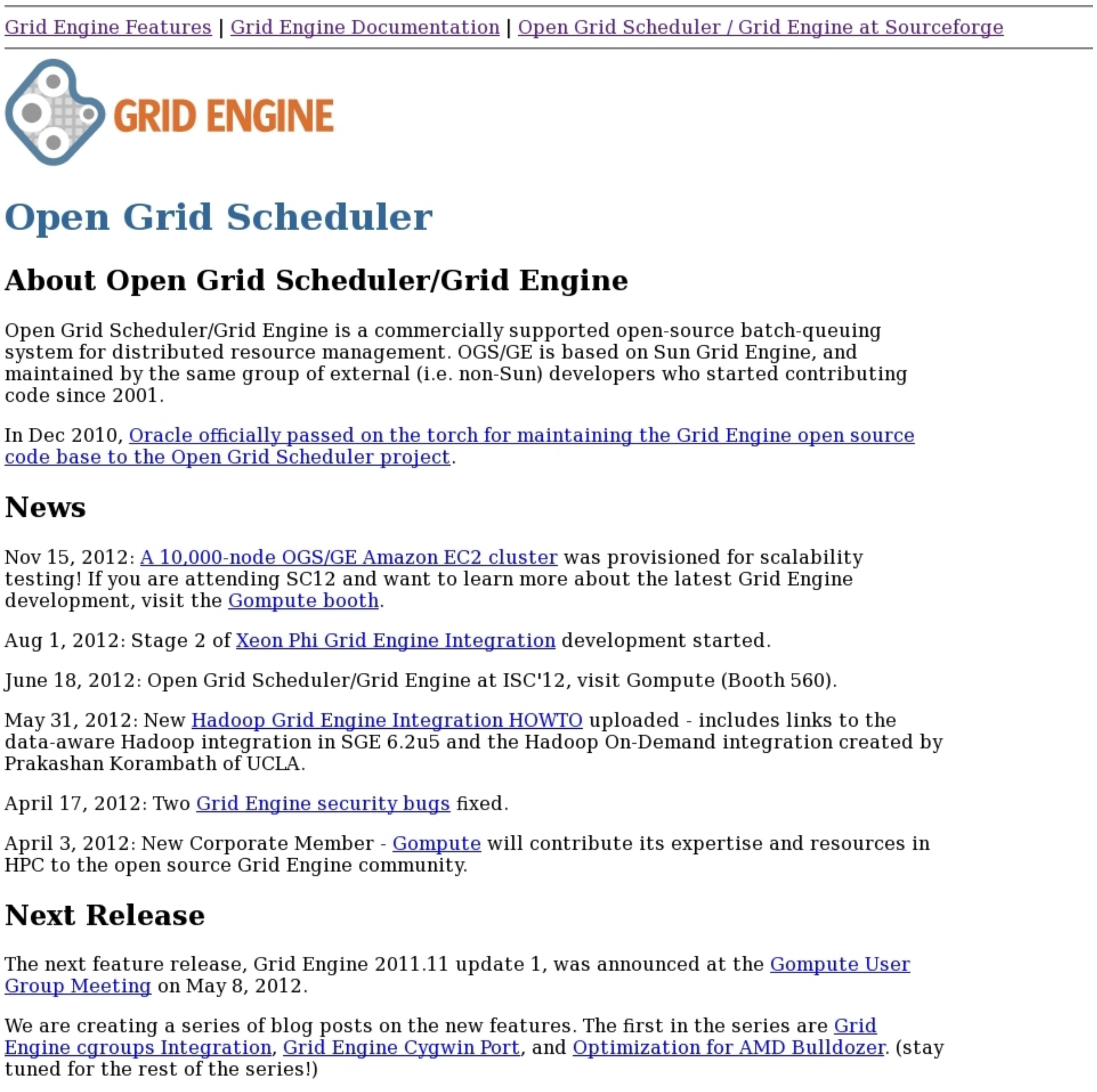
Sun Grid Engine is released with the SISSL (Sun Industry Standards Source License) v 1.2 license, is a FLOSS license recognized by the FSF and the OSI but is not compatible with the GPL.

3.20 Community

After the adquisition of Sun Microsystem by Oracle the project was in doubt, Oracle took the project , made a restyling and relaunched as Oracle Grid Engine, for the support of the FLOSS version a community version rise with the name of "open grid scheduler" and is maintained by the same group of developers who started contributing code since 2001 in Sun Grid Engine project.

The FLOSS community that supports "open grid scheduler" is hosted in the url `http://gridscheduler.sourceforge.net` and provide extend documentation about the project, access to the source code in Sourceforge (`http://sourceforge.net/projects/gridscheduler`) and access to the mailing lists of the project.

Figure 3.60: Open Grid Community



Grid Engine Features | Grid Engine Documentation | Open Grid Scheduler / Grid Engine at Sourceforge

Open Grid Scheduler

About Open Grid Scheduler/Grid Engine

Open Grid Scheduler/Grid Engine is a commercially supported open-source batch-queuing system for distributed resource management. OGS/GE is based on Sun Grid Engine, and maintained by the same group of external (i.e. non-Sun) developers who started contributing code since 2001.

In Dec 2010, Oracle officially passed on the torch for maintaining the Grid Engine open source code base to the Open Grid Scheduler project.

News

Nov 15, 2012: A 10,000-node OGS/GE Amazon EC2 cluster was provisioned for scalability testing! If you are attending SC12 and want to learn more about the latest Grid Engine development, visit the Gompute booth.

Aug 1, 2012: Stage 2 of Xeon Phi Grid Engine Integration development started.

June 18, 2012: Open Grid Scheduler/Grid Engine at ISC'12, visit Gompute (Booth 560).

May 31, 2012: New Hadoop Grid Engine Integration HOWTO uploaded - includes links to the data-aware Hadoop integration in SGE 6.2u5 and the Hadoop On-Demand integration created by Prakashan Korambath of UCLA.

April 17, 2012: Two Grid Engine security bugs fixed.

April 3, 2012: New Corporate Member - Gompute will contribute its expertise and resources in HPC to the open source Grid Engine community.

Next Release

The next feature release, Grid Engine 2011.11 update 1, was announced at the Gompute User Group Meeting on May 8, 2012.

We are creating a series of blog posts on the new features. The first in the series are Grid Engine cgroups Integration, Grid Engine Cygwin Port, and Optimization for AMD Bulldozer. (stay tuned for the rest of the series!)

Chapter 4

Real Implementation

There are some important reasons that make very useful the implementation of a H.A. grid of servers for batch processing. A grid open us a great variety of posibilities, we can add or quit execution host with the grid continuously running, for example if we want to do determined maintenance tasks in a server, we can quit it of the grid , do the administration tasks on it and after add it again to the grid maintaining a batch environment ever available in all moment passing the batch load to other available execution host during the maintenance tasks.

Also another situacion like an accidental fall of a execution host of the grid shows the advantages of have this kind of batch system implementation. In this case the fallen host is separated and ignored and the user process are continuously launched normally to the rest of the execution hosts until the fallen host be again operating , moment in wich the master host will add again to the grid as before.

The use of a resource manager that monitorize all the resources of the grid prevents the collapse of a host in terms of cpu, memory or whatever parameter configured, this manager in combination with a scheduler don't allow the launching of any process if for example a determined host has high values of cpu load or has low values of free memory preventing its fall.

All this combination of features allow us have a lot of posibilities referring batch processing and give us the security of have continued availability in our batch processing system.

4.1 Decide the roles of the servers

There are several roles that you can assign to a group of hosts, but the most important roles are "Submit role" that gives us the capacity of launch process to the grid, and the "Execution role" that allow to a server execute user process , in the example of implementation showed we difference two kind of servers according to its roles, the "Submit host" is a host that is in charge of the launching of the process, has a configured cron that plans the automatic executions of the process and has an intranet in a web server for allow to users to launch process to the grid, the other servers, the "Execution Hosts" must have installed all the requirements of the users process in order to execute them.

The figure 4.1 represents a real implementation of a grid for batch processing, in it we have different machines with different roles:

- moscarsub: Server that only monitorize and launch process to the grid, has a cron and a intranet (Apache web server) for do this tasks.

- moscar0,moscar1,moscar2: Are execution batch , execute the process launched by the server moscarsub.

4.2 Decide the number of queues and its size

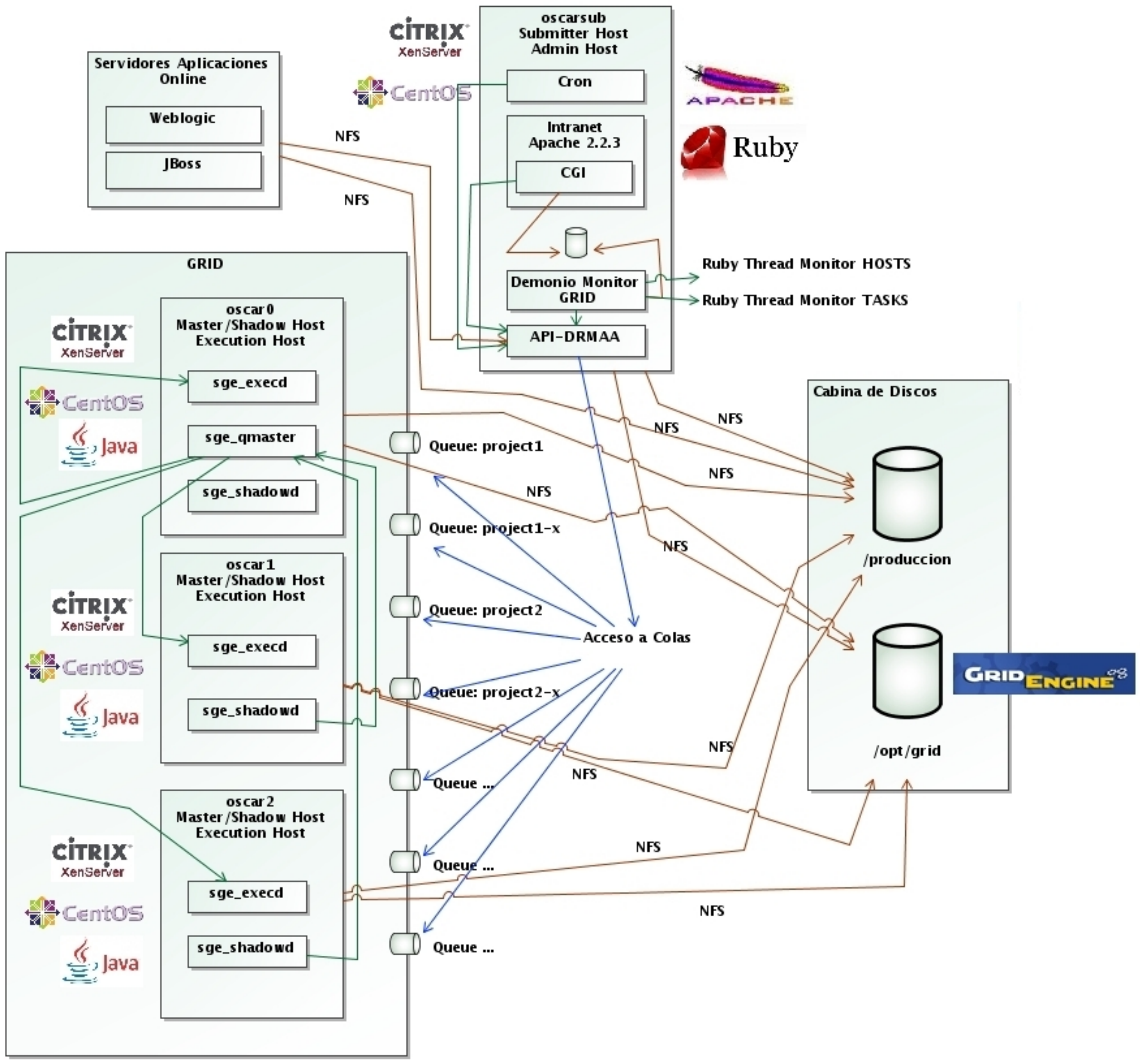
The number of queues and its size depends of our needs, all the process that access for example to a critical section and wich must control its concurrent execution must be queued in the same queue giving special attention to the size of the queue that represents the number of concurrent process that can be in execution state by each host that has defines this queue.

4.3 Shared Disks

For the correct implantation of a grid of server we will need that almost two portions of disk will be shared (usually by NFS) between all the servers of the grid:

- Section of disk for the spool directory of the grid.
- Section of disk with the users scripts / programs that will be executed by the execution hosts.

Figure 4.1: Real implementation of a grid



4.4 Developing our monitor of the grid: grid-monitor

The monitorization of the grid is a very important aspect when we think in how to integrate the grid into the current infraestructure of a Data Center, with a grid you are adding an abstraction layer that could difficult to know what is really happennig in our batch environment, applications like Sun Grid Engine provide tools in command line and with a GUI like Qmon that allow us to configure the grid and have a control of what happen in the grid, but for many cases these tools aren't enough for at glance know the current state of the Grid, also theses tools aren't easily integrate with other systems , reasons for wich is very useful use the DRMAA API included with SGE and program our monitor and control system that fits perfectly to our Data Center.

For develop a monitor for the grid we only need known some basic objects of the API DRMAA (See the section API DRMAA) and with a simple system of cgis over a web server (like Apache web server) we can design a pretty good html page that contains all that we need to know at glance the state of our grid batch system. In our real implementation of a grid, we have a monitor system developed in Ruby using the API DRMAA called **grid-monitor** and composed by these elements:

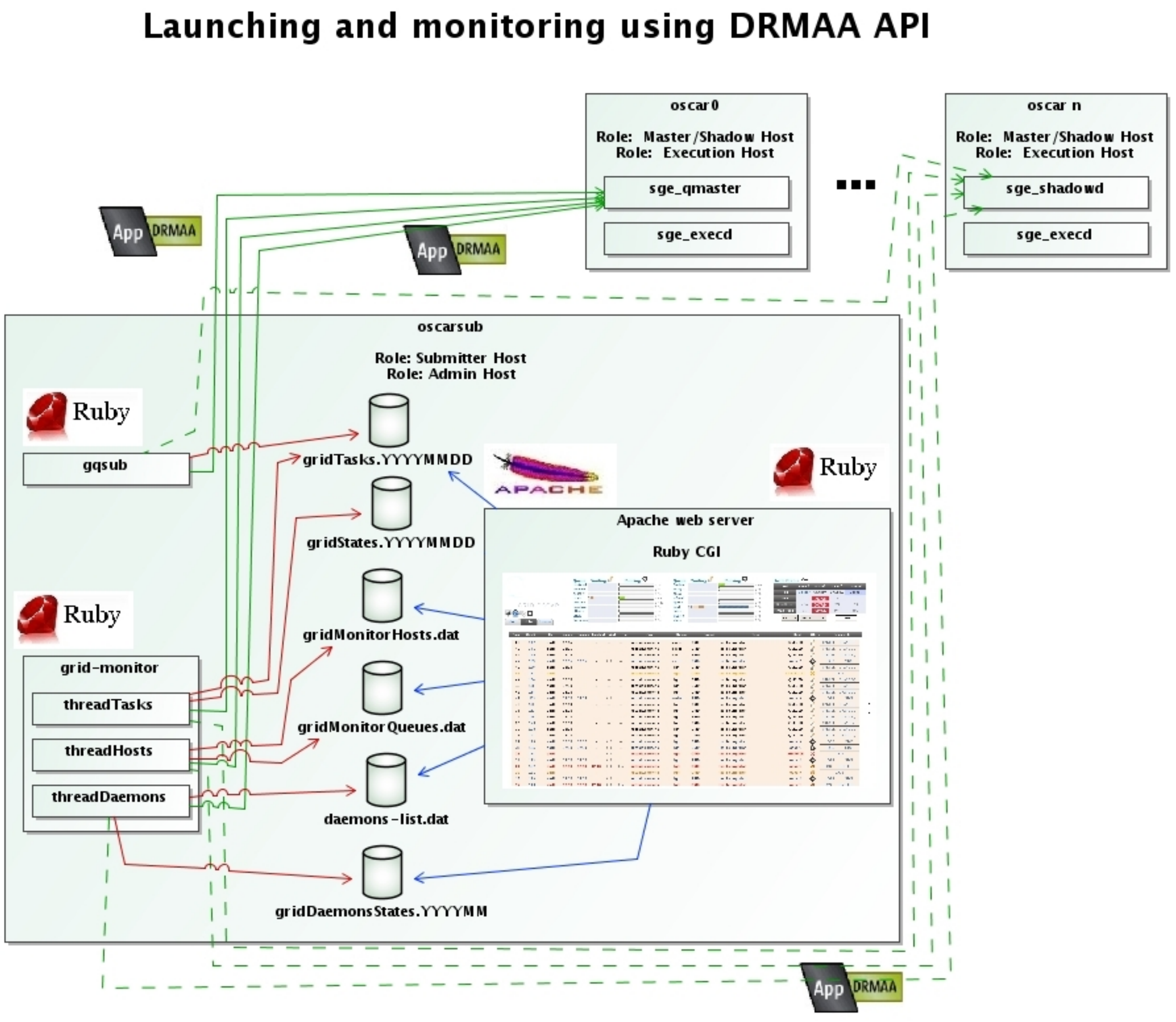
- A Daemon grid-monitor that using the API DRMAA (Ruby) is continuously asking to the master host of the grid (if the master host goes down will ask to a shadow master of other hosts) about the status of the user tasks and about the status of the hosts, this information is written to a plain files wich will be accessed by the cgis opened by the users.
- A serie of cgis, these cgis doesn't realize any monitorization , only access to the files filled by the daemon grid-monitor leaving to this daemon the heavy monitor tasks and having therefore thin cgis allowing us open some of them without increase the load of the grid server.
- A script ruby called qsub that using the API DRMAA allow us to launch process in to the grid and controlling the status of its queuing process.

The daemon grid-monitor (See figure 4.2) is composed by three threads:

- **threadTasks**: In charge of monitoring the status of the tasks and writting the status in the file gridTasks.YYYYMMDD
- **threadHosts**: In charge of monitoring the status of the hosts and queues and writting the status in the files gridMonitorHosts.dat and gridMonitorQueues.dat
- **threadDaemons**: In charge of monitoring the status of the daemons and writting the status in the file gridDaemonsStates.dat

There is a logical separation between the normal user tasks and the user daemons because we understanding of daemon the user tasks wich must be repeted every period of time a lot of times along the day (usually launched periodically by the cron daemon) and is better control them separately in other plain files and be displayed in others html screens that represent better its current status.

Figure 4.2: Developing a grid monitor



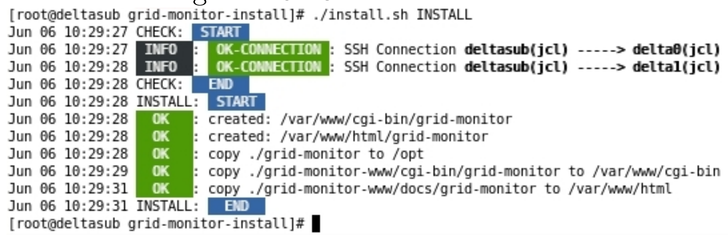
4.4.1 Monitor installation

Requirements

Before install grid-monitor you need:

- A Linux server.
- Sun Grid Engine installed (usually in /opt/grid or /opt/sungrid)
- Ruby
- Ruby Gems: YAML
- Apache Web server
- All the servers of the grid must be able to connect between them automatically with a user specified in the configuration file (userSSHMonitor) (must has well configured the file .ssh/authorized_keys)

Figure 4.3: Grid monitor installation



```
[root@deltasub grid-monitor-install]# ./install.sh INSTALL
Jun 06 10:29:27 CHECK:  START
Jun 06 10:29:27  INFO  :  OK-CONNECTION : SSH Connection deltasub(jcl) -----> delta0(jcl)
Jun 06 10:29:28  INFO  :  OK-CONNECTION : SSH Connection deltasub(jcl) -----> delta1(jcl)
Jun 06 10:29:28 CHECK:   END
Jun 06 10:29:28 INSTALL:  START
Jun 06 10:29:28   OK   : created: /var/www/cgi-bin/grid-monitor
Jun 06 10:29:28   OK   : created: /var/www/html/grid-monitor
Jun 06 10:29:28   OK   : copy ./grid-monitor to /opt
Jun 06 10:29:29   OK   : copy ./grid-monitor-www/cgi-bin/grid-monitor to /var/www/cgi-bin
Jun 06 10:29:31   OK   : copy ./grid-monitor-www/docs/grid-monitor to /var/www/html
Jun 06 10:29:31 INSTALL:  END
[root@deltasub grid-monitor-install]# █
```

- All the server must share by NFS the path of the installation of the grid (usually /opt/grid) and the path where are the scripts or user programs that will be executed in the Grid.

Installation

For install the monitor added to this thesis you must follow the next steps:

- Download the installation files from : `https://github.com/antcargit/MSWL-Thesis/blob/master/mswl-thesis-acarmona-grid-monitor-install.tar`
- Decompress the tar file: mswl-thesis-acarmona-grid-monitor-install.tar
- Configure the installation checking the file: grid-monitor-install.conf, in this file is very important set correctly the situation of the cgi-bin and html directories (pathcgiBinApache and pathDocsApache properties) and the path base of the installation of the monitor daemon (option pathBaseInstallation usually set to /opt)
- ./install CHECK : For check the environment
- ./install INSTALL : For install grid-monitor
- ./install UNINSTALL : If you want to uninstall grid-monitor.

For configure the grid-monitor daemon as a service, copy the file grid-monitor-wrapper.sh in /etc/init.d (or path equivalent in your distribution).

4.4.2 Licenses

License of grid-monitor:

Third-Party Licenses

The licenses of the third-party software added to grid-monitor:

Dynamic Drive SDMenu license

Prototype JavaScript framework license

4.4.3 Operation

Once the installation is completed we can access to the web monitor:

http://servername.domain.com/cgi-bin/grid-monitor/jobs

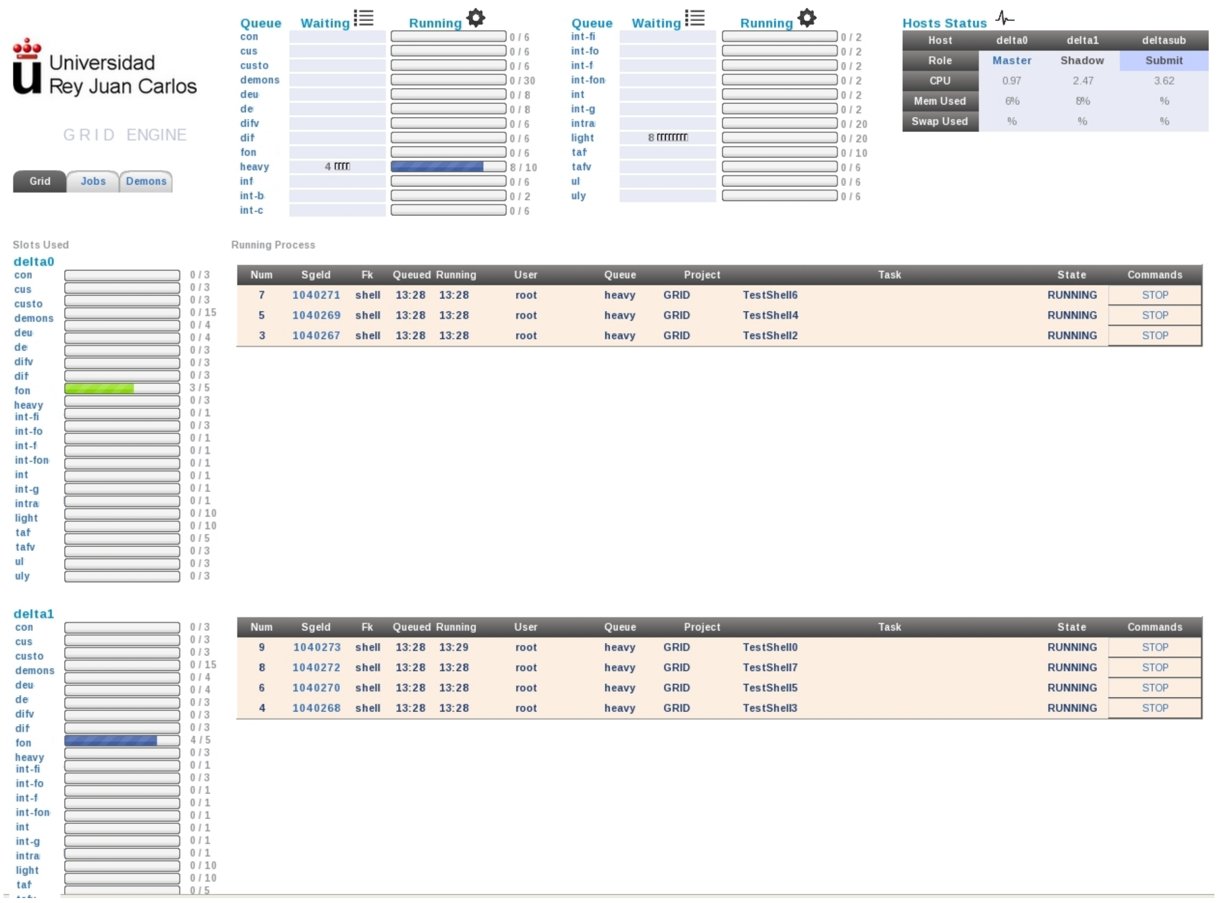
We can see the list of jobs and its state (figure 4.5), the current running process in the Grid (figure 4.4), and information about the status of a determined process (figure 4.6).

In the figure 4.7 we can see in detail how the waiting and running process are showed in each queue, in the figure 4.8 is showed a “Suspend” queue that doesn’t admit tasks with the label “CLOSED” and doesn’t execute process (“OFF” label), and a “Disabled” queue that allows enter tasks but doesn’t execute any of them and has only the “OFF” label.

In the figure 4.9 we can see how is showed the status of the hosts and specifically when a host is down.

Finally the figure 4.10 shows the different status of a jobs and how grid-monitor shows them.

Figure 4.4: Grid monitor current running process



Queue	Waiting	Running
con		0 / 6
cus		0 / 6
custo		0 / 6
demons		0 / 30
deu		0 / 8
de		0 / 8
difv		0 / 6
dif		0 / 6
fon		0 / 6
heavy	4	8 / 10
inf		0 / 6
int-b		0 / 2
int-c		0 / 6

Queue	Waiting	Running
int-fi		0 / 2
int-fo		0 / 2
int-f		0 / 2
int-fon		0 / 2
int		0 / 2
int-g		0 / 2
intra		0 / 20
light	8	0 / 20
taf		0 / 10
tafv		0 / 6
ul		0 / 6
uly		0 / 6

Hosts Status

Host	delta0	delta1	deltasub
Role	Master	Shadow	Submit
CPU	0.97	2.47	3.62
Mem Used	6%	8%	%
Swap Used	%	%	%

Slots Used

delta0

con	0 / 3
cus	0 / 3
custo	0 / 3
demons	0 / 15
deu	0 / 4
de	0 / 4
difv	0 / 3
dif	0 / 3
fon	0 / 3
heavy	3 / 5
int-fi	0 / 3
int-fo	0 / 1
int-f	0 / 3
int-fon	0 / 1
int	0 / 1
int-g	0 / 1
intra	0 / 1
light	0 / 1
taf	0 / 1
tafv	0 / 10
ul	0 / 10
uly	0 / 5
	0 / 3
	0 / 3
	0 / 3

Running Process

Num	SgeId	Fk	Queued	Running	User	Queue	Project	Task	State	Commands
7	1040271	shell	13:28	13:28	root	heavy	GRID	TestShell6	RUNNING	STOP
5	1040269	shell	13:28	13:28	root	heavy	GRID	TestShell4	RUNNING	STOP
3	1040267	shell	13:28	13:28	root	heavy	GRID	TestShell2	RUNNING	STOP

delta1

con	0 / 3
cus	0 / 3
custo	0 / 3
demons	0 / 15
deu	0 / 4
de	0 / 4
difv	0 / 3
dif	0 / 3
fon	0 / 3
heavy	4 / 5
int-fi	0 / 3
int-fo	0 / 1
int-f	0 / 3
int-fon	0 / 1
int	0 / 1
int-g	0 / 1
intra	0 / 1
light	0 / 1
taf	0 / 1
	0 / 10
	0 / 10
	0 / 5

Num	SgeId	Fk	Queued	Running	User	Queue	Project	Task	State	Commands
9	1040273	shell	13:28	13:29	root	heavy	GRID	TestShell0	RUNNING	STOP
8	1040272	shell	13:28	13:28	root	heavy	GRID	TestShell7	RUNNING	STOP
6	1040270	shell	13:28	13:28	root	heavy	GRID	TestShell5	RUNNING	STOP
4	1040268	shell	13:28	13:28	root	heavy	GRID	TestShell3	RUNNING	STOP

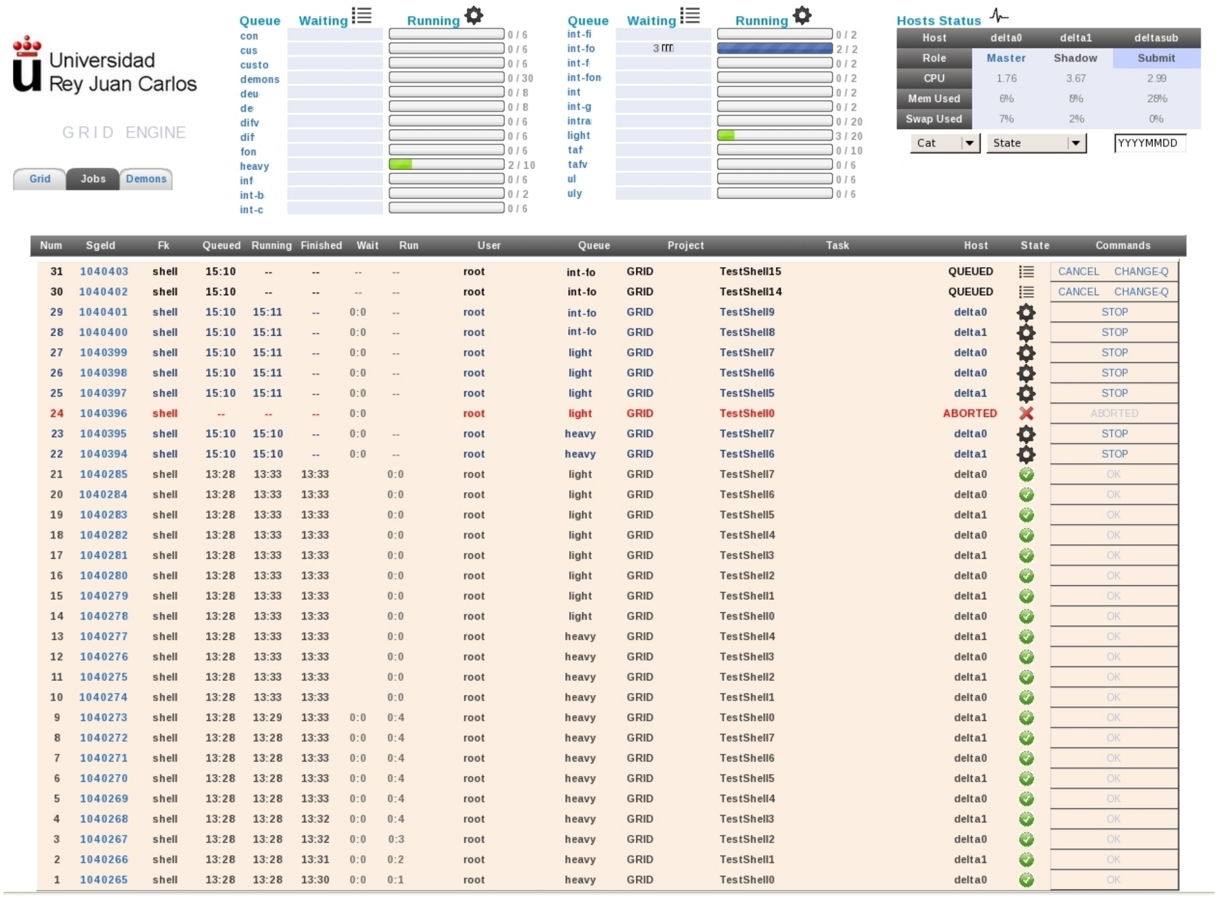


Figure 4.5: Grid monitor job list

Queue	Waiting	Running
con		0 / 6
cus		0 / 6
custo		0 / 6
demons		0 / 30
deu		0 / 8
de		0 / 8
difv		0 / 6
dif		0 / 6
fon		0 / 6
heavy		2 / 10
inf		0 / 6
int-b		0 / 2
int-c		0 / 6

Queue	Waiting	Running
int-fi		0 / 2
int-fo	3	2 / 2
int-f		0 / 2
int-fon		0 / 2
int		0 / 2
int-g		0 / 2
intra		0 / 20
light		3 / 20
taf		0 / 10
tafv		0 / 6
ul		0 / 6
uly		0 / 6

Hosts Status

Host	delta0	delta1	deltasub
Role	Master	Shadow	Submit
CPU	1.76	3.67	2.99
Mem Used	6%	8%	28%
Swap Used	7%	2%	0%

Num	SgeId	Fk	Queued	Running	Finished	Wait	Run	User	Queue	Project	Task	Host	State	Commands
31	1040403	shell	15:10	--	--	--	--	root	int-fo	GRID	TestShell15	QUEUED	☰	CANCEL CHANGE-Q
30	1040402	shell	15:10	--	--	--	--	root	int-fo	GRID	TestShell14	QUEUED	☰	CANCEL CHANGE-Q
29	1040401	shell	15:10	15:11	--	0:0	--	root	int-fo	GRID	TestShell9	delta0	⚙	STOP
28	1040400	shell	15:10	15:11	--	0:0	--	root	int-fo	GRID	TestShell8	delta1	⚙	STOP
27	1040399	shell	15:10	15:11	--	0:0	--	root	light	GRID	TestShell7	delta0	⚙	STOP
26	1040398	shell	15:10	15:11	--	0:0	--	root	light	GRID	TestShell6	delta0	⚙	STOP
25	1040397	shell	15:10	15:11	--	0:0	--	root	light	GRID	TestShell5	delta1	⚙	STOP
24	1040396	shell	--	--	--	0:0		root	light	GRID	TestShell0	ABORTED	✗	ABORTED
23	1040395	shell	15:10	15:10	--	0:0	--	root	heavy	GRID	TestShell7	delta0	⚙	STOP
22	1040394	shell	15:10	15:10	--	0:0	--	root	heavy	GRID	TestShell6	delta1	⚙	STOP
21	1040285	shell	13:28	13:33	13:33		0:0	root	light	GRID	TestShell7	delta0	✓	OK
20	1040284	shell	13:28	13:33	13:33		0:0	root	light	GRID	TestShell6	delta0	✓	OK
19	1040283	shell	13:28	13:33	13:33		0:0	root	light	GRID	TestShell5	delta1	✓	OK
18	1040282	shell	13:28	13:33	13:33		0:0	root	light	GRID	TestShell4	delta0	✓	OK
17	1040281	shell	13:28	13:33	13:33		0:0	root	light	GRID	TestShell3	delta1	✓	OK
16	1040280	shell	13:28	13:33	13:33		0:0	root	light	GRID	TestShell2	delta0	✓	OK
15	1040279	shell	13:28	13:33	13:33		0:0	root	light	GRID	TestShell1	delta1	✓	OK
14	1040278	shell	13:28	13:33	13:33		0:0	root	light	GRID	TestShell0	delta0	✓	OK
13	1040277	shell	13:28	13:33	13:33		0:0	root	heavy	GRID	TestShell4	delta1	✓	OK
12	1040276	shell	13:28	13:33	13:33		0:0	root	heavy	GRID	TestShell3	delta0	✓	OK
11	1040275	shell	13:28	13:33	13:33		0:0	root	heavy	GRID	TestShell2	delta1	✓	OK
10	1040274	shell	13:28	13:33	13:33		0:0	root	heavy	GRID	TestShell1	delta0	✓	OK
9	1040273	shell	13:28	13:29	13:33	0:0	0:4	root	heavy	GRID	TestShell0	delta1	✓	OK
8	1040272	shell	13:28	13:28	13:33	0:0	0:4	root	heavy	GRID	TestShell7	delta1	✓	OK
7	1040271	shell	13:28	13:28	13:33	0:0	0:4	root	heavy	GRID	TestShell6	delta0	✓	OK
6	1040270	shell	13:28	13:28	13:33	0:0	0:4	root	heavy	GRID	TestShell5	delta1	✓	OK
5	1040269	shell	13:28	13:28	13:33	0:0	0:4	root	heavy	GRID	TestShell4	delta0	✓	OK
4	1040268	shell	13:28	13:28	13:32	0:0	0:4	root	heavy	GRID	TestShell3	delta1	✓	OK
3	1040267	shell	13:28	13:28	13:32	0:0	0:3	root	heavy	GRID	TestShell2	delta0	✓	OK
2	1040266	shell	13:28	13:28	13:31	0:0	0:2	root	heavy	GRID	TestShell1	delta1	✓	OK
1	1040265	shell	13:28	13:28	13:30	0:0	0:1	root	heavy	GRID	TestShell0	delta0	✓	OK

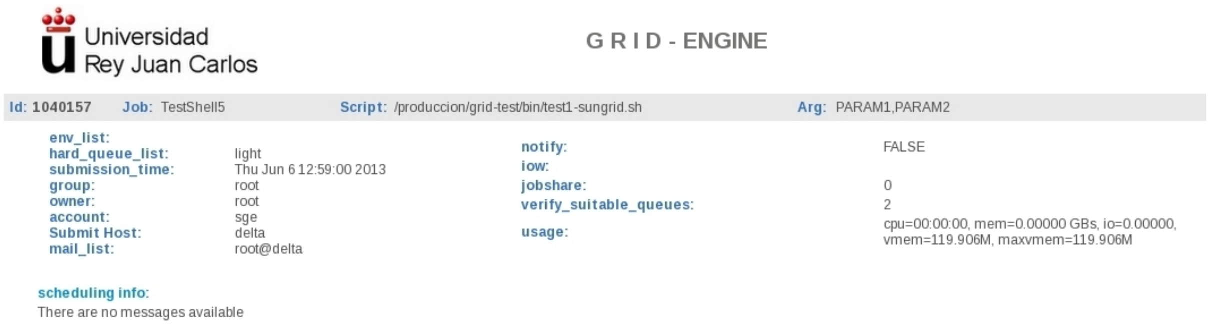


Figure 4.6: Grid monitor job info

Id: 1040157 **Job:** TestShell5 **Script:** /produccion/grid-test/bin/test1-sungrid.sh **Arg:** PARAM1,PARAM2

env_list:	
hard_queue_list:	light
submission_time:	Thu Jun 6 12:59:00 2013
group:	root
owner:	root
account:	sge
Submit Host:	delta
mail_list:	root@delta

notify:	FALSE
iow:	
jobshare:	0
verify_suitable_queues:	2
usage:	cpu=00:00:00, mem=0.00000 GBs, io=0.00000, vmem=119.906M, maxvmem=119.906M

scheduling info:
There are no messages available

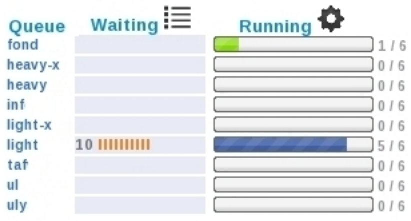
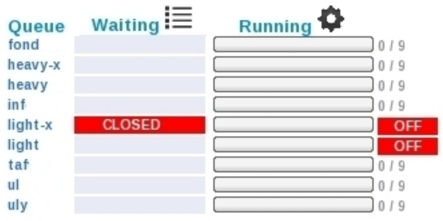


Figure 4.7: Grid monitor queue status

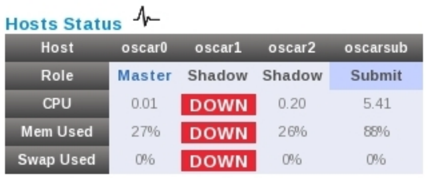
Queue	Waiting	Running
fond		1 / 6
heavy-x		0 / 6
heavy		0 / 6
inf		0 / 6
light-x		0 / 6
light	10	5 / 6
taf		0 / 6
ul		0 / 6
uly		0 / 6

Figure 4.8: Grid monitor Suspend / Disable queues



Queue	Waiting	Running	
fond			0 / 9
heavy-x			0 / 9
heavy			0 / 9
inf			0 / 9
light-x	CLOSED		OFF
light			OFF
taf			0 / 9
ul			0 / 9
uly			0 / 9

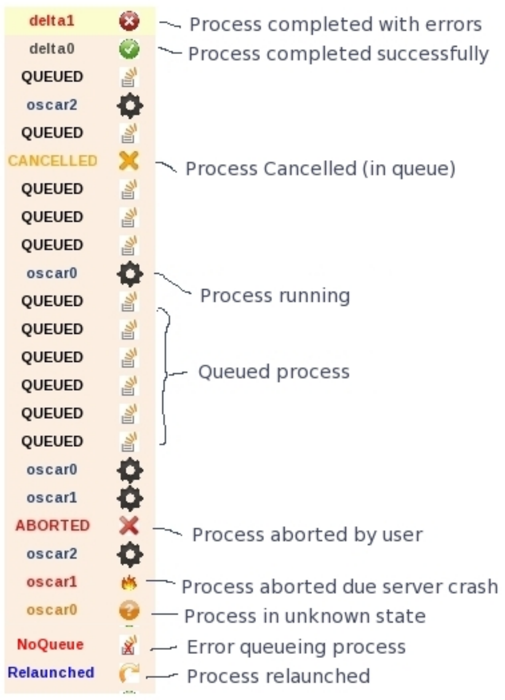
Figure 4.9: Grid monitor hosts status



Hosts Status

Host	oscar0	oscar1	oscar2	oscarsub
Role	Master	Shadow	Shadow	Submit
CPU	0.01	DOWN	0.20	5.41
Mem Used	27%	DOWN	26%	88%
Swap Used	0%	DOWN	0%	0%

Figure 4.10: Grid monitor list of job states



delta1	Process completed with errors
delta0	Process completed successfully
QUEUED	
oscar2	
QUEUED	
CANCELLED	Process Cancelled (in queue)
QUEUED	
QUEUED	
QUEUED	
oscar0	Process running
QUEUED	Queued process
QUEUED	
QUEUED	
QUEUED	
QUEUED	
QUEUED	
oscar0	
oscar1	
ABORTED	Process aborted by user
oscar2	
oscar1	Process aborted due server crash
oscar0	Process in unknown state
NoQueue	Error queueing process
Relaunched	Process relaunched

4.5 Conclusions

The implantation of a Grid for manage the batch environment (specifically Sun Grid Engine / Open Grid) of our Data Center is highly recommended, it is easy to install , only the requirement of mount NFS file systems between the servers is worth mentioning, give us a high flexibility can add or quit batch servers in the grid dynamically, exploiting the most of the resources availables in the servers, allowing save situations of overload of cpu or memory use, avoiding the collapse of a server and allowing us to perform maintenance tasks in the servers faster and easily can maintain a full batch environment ever available.

Sun Grid Engine / Open Grid specifically is a very complete project , that allows us have small and big grids , easily manageable thanks to its command line interface and its graphical tool Qmon. Finally is important highlight the very complete API (DRMAA) that gives us a lot of posibilities allowing us programming our tools that fits perfectly with the functionality of our Data Center.

Bibliography

[1] http://gridscheduler.sourceforge.net/

[2] http://gridscheduler.sourceforge.net/Gridengine_SISSL_license.html

[3] http://www.drmaa.org/

[4] http://www.ggf.org/

[5] http://www.adaptivecomputing.com/products/open-source/torque/

[6] http://www.clusterresources.com/products.php

[7] http://www.supercluster.org/mailman/listinfo

[8] http://www.supercluster.org/pipermail/torqueusers/