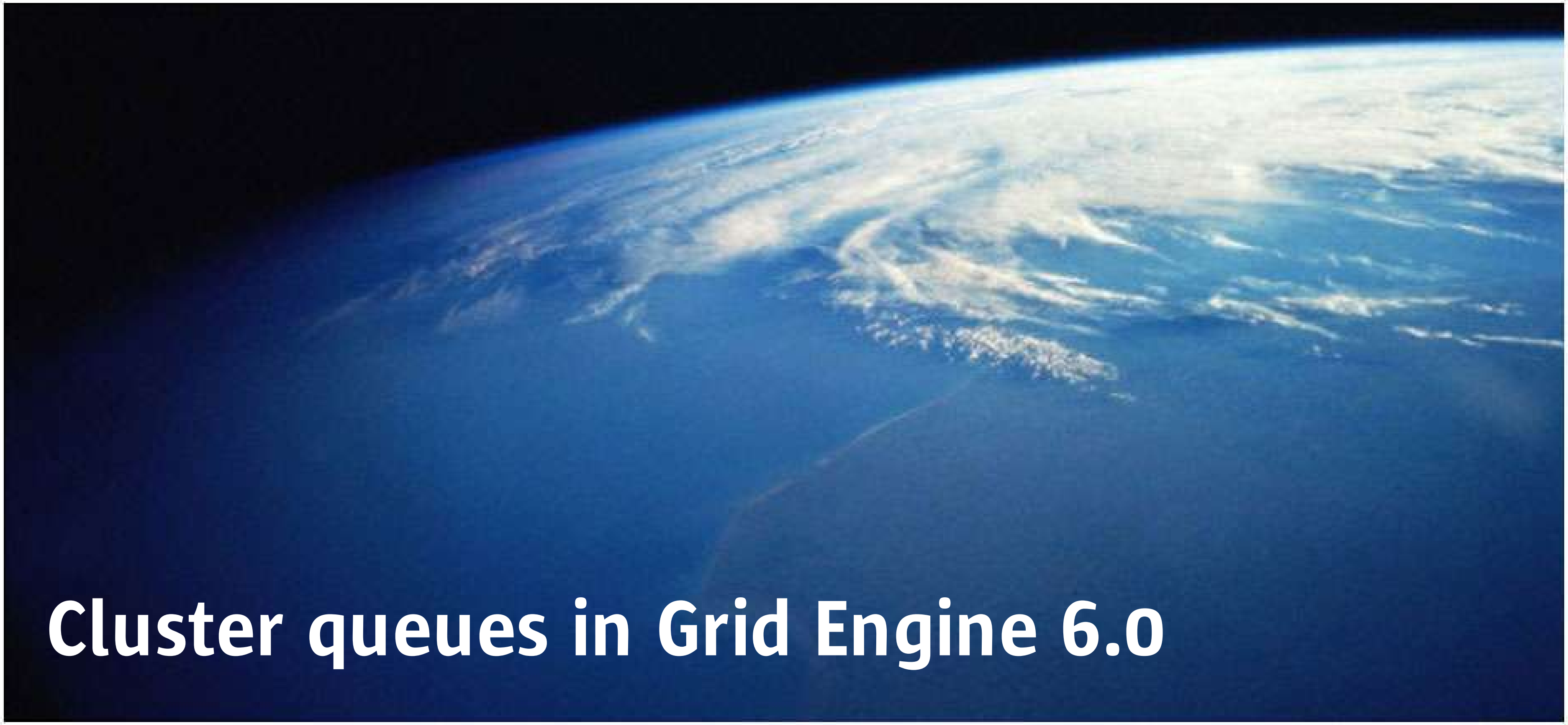


Cluster queues in Grid Engine 6.0

Andreas Haas

Software Engineering

Sun Grid Engine

Sun microsystems
We make the net work.

Introduction

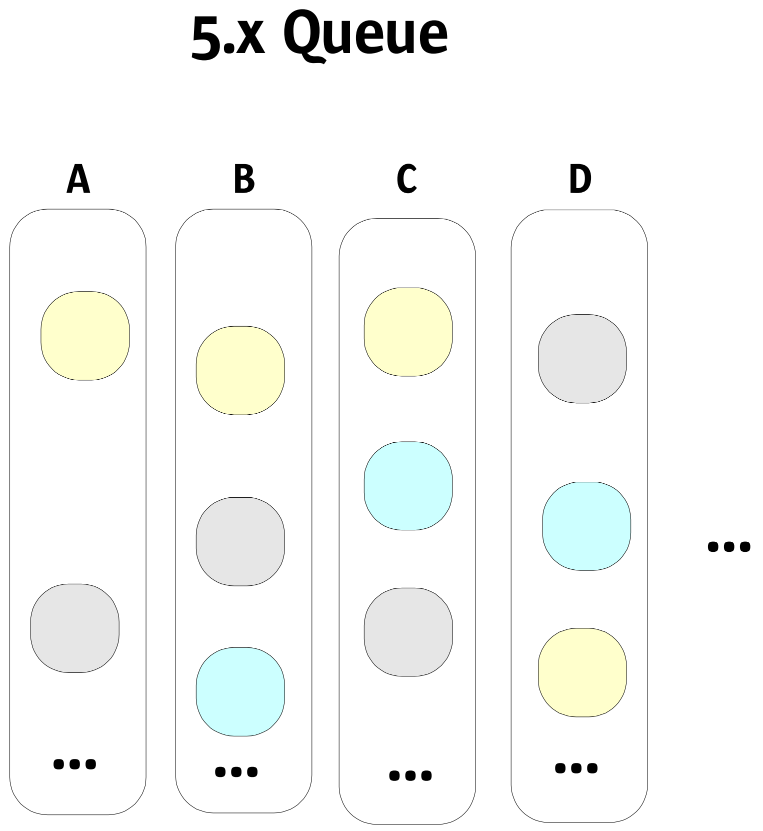
- What is a Grid Engine 5.x queue?
  - Partitions a host
  - Describes the profile of requirements a job must have to be started
  - Describes the runtime environment of a executed job

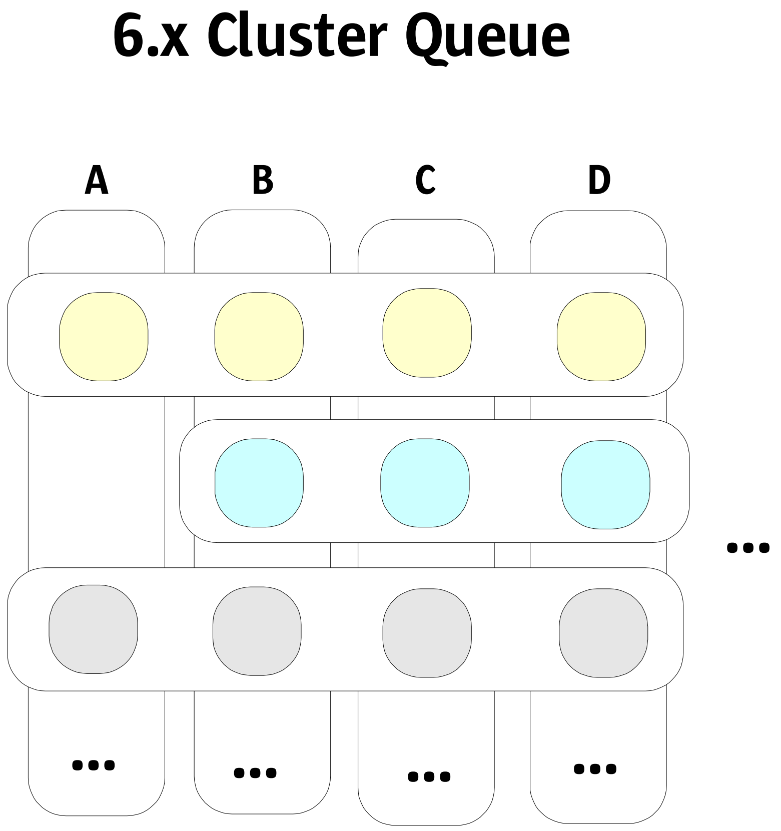
Why so many Queues?

- Many different types of jobs
- Different policies
- Many hosts

- Reduction via grouping by hosts/job-types/etc.
- Greatly will simplify administration

Overall Changes





Three steps

- Support of multiple hosts per queue configuration
- Stand-by with different queue attribute settings per execution host as used
- New hostgroups can be used in queue configuration

Multiple Hosts

```
qname              big
hostlist           balrog sauron fangorn durin frodo eomer
seq_no             0
load_thresholds    NONE
suspend_thresholds NONE
...
```

The first step is to support in Grid Engines queue configuration not only a single hostname but also a list of hostnames. This makes the queue a cluster queue, since it allows managing a cluster of execution hosts by means of a single queue configuration.

Different attribute settings

```
Qname              big
hostlist           balrog eomer ori fangorn durin frodo
seq_no             0,[balrog=1],[eomer=1],[durin=2],[fangorn=2],[frodo=2]
load_thresholds    NONE
suspend_thresholds NONE
...
```

The next step is to allow for a differentiation of each queue attribute separately for each execution host. This significantly broadens the applicability of cluster queues as it allows for managing also fairly heterogeneous clusters by means of a single queue configuration.

New Object: Hostgroup

```
group_name   @solaris64                group_name   @linux
hostlist     balrog eomer ori          hostlist     fangorn durin frodo
```

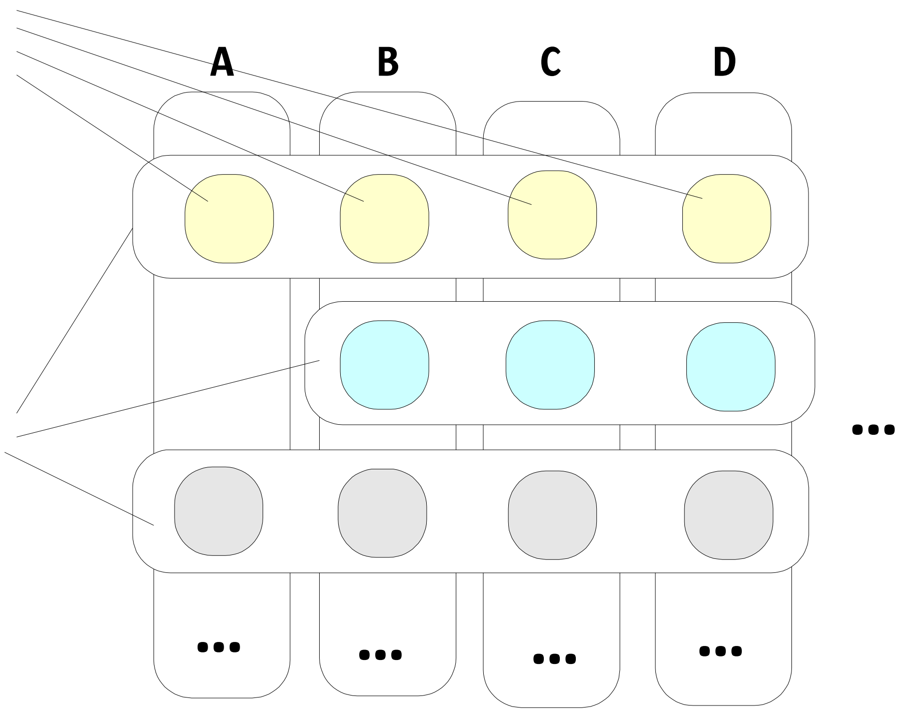
```
qname              big
hostlist           @solaris @linux
seq_no             0,[@solaris64=1],[@linux=2],[ori=0]
load_thresholds    NONE
suspend_thresholds NONE
```

The next step is to introduce host groups into the standard build of Grid Engine and allow host groups to be used for expressing differentiation of queue attributes as with execution hosts in the step before.

Cluster Queue Glossar

Queue Instance:
identical to 5.x Queue

Hostgroup:
Group of hosts
defined by
administrator



Cluster Queue:
describes a set of
Queue Instances

Queue Domain:
all Queue Instances of
a Cluster Queue whose
hosts belong to a
particular Hostgroup

Compreshensive cluster overview

```
queuename       load_avg  used  E     u     A     a     s     d     tot.
-----------------------------------------------------------------------------
   ---
cluster_q1      0.79      00438 00000 00050 00000 00050 00012 00005 00500
cluster_q2      0.78      00302 00000 00048 00000 00048 00000 00100 00450
cluster_q3      0.91      00448 00002 00000 00000 00000 00000 00000 00450
...
```

load_avg := sum(np_load_avg * slots_at_host) / all_available_slots

Number of job slots:

- used
- E: queue instance error
- u: unknown state
- A: suspend alarm
- a: load alarm
- s: suspended
- d: disabled
- tot.: total available

Natural Queue grouping

Per definition Cluster Queues form a set of Queue Domains and Queue Instances. To realize this in 5.x it was necessary to define various complex attributes.

Some examples for Queue specifications in 6.x:

```
qsub -q medium

qsub -q fast@@solaris job.sh

qmod -e big

qmod -c big@@linux big@balrog
```

Diagnosis queue instances

The configuration of Cluster Queues and Queue Instances might bee seen with the qconf -sq command.

```
> qconf -sq cluster_queue
qname                cluster_queue
hostlist             @solaris64 ori
seq_no               0,[balrog=1]
load_thresholds      np_load_avg=1.75
suspend_thresholds   NONE
nsuspend             1
suspend_interval     00:00:60
priority             0,[balrog=3]
min_cpu_interval     00:05:00
processors           UNDEFINED
qtype                BATCH INTERACTIVE,
   [ori=BATCH]
ckpt_list            NONE,[@solaris64=pe1]
pe_list              NONE
rerun                FALSE
slots                1,[@solaris64=2]
...
```

```
> qconf -sq cluster_queue@ori
qname                cluster_queue@ori
hostname             ori
seq_no               0
load_thresholds      np_load_avg=1.75
suspend_thresholds   NONE
nsuspend             1
suspend_interval     00:00:60
priority             0
min_cpu_interval     00:05:00
processors           UNDEFINED
qtype                BATCH
ckpt_list            NONE
pe_list              NONE
rerun                FALSE
slots                1
...
```

Additional conceptual cleanup

- New queue states
- Parallel Environment changes
- Checkpoint Interface changes
- User defined complexes cleanup

Complex Attributes

- The value attribute is removed from the complex configuration
- Host, queue and user-defined complexes are obsolete. All complex attribues are part of a global container.
- Forced attributes are configured differently in 6.x: non-consumable fixed attributes have to be specified in the complex_values field of the Cluster Queue.
- The complex_list attribute in the Cluster Queue is obsolete.

5.x Scenario:

```
qname             queue_name
complex_list
    user_defined1
complex_values    xyz
...
```

6.x Scenario:

```
qname             queue_name
complex_values    xyz=5
...
```

Parallel Object

5.x Scenario:

```
pe_name     pe1
queue_list  fast
...

qname       fast
qtype       BATCH PARALLEL
...
```

6.x Scenario:

```
pe_name     pe1
...

qname       fast
qtype       BATCH
pe_list     pe1
...
```

- The relation between Cluster Queue and Parallel Environment is defined by the pe_list attribute in the Cluster Queue configuration.
- A Cluster Queue is automatically of type PARALLEL if the pe_list attribute containes at least one reference.
- In 5.x it was possible to use keyword 'all' for queue_list. This is not necessary anymore. A Parallel environment attached to a Cluster Queue is automatically attached to each Queue Instance.

Checkpointing Interface

5.x Scenario:

```
ckpt_name   ckpt1
queue_list  fast
...

qname       fast
qtype       BATCH CHECKPOINTING
...
```

6.x Scenario:

```
ckpt_name   ckpt1
...

qname       fast
qtype       BATCH
ckpt_list   ckpt1
...
```

- The relation between Cluster Queue and Checkpointing Interface is defined by the ckpt_list attribute in the Cluster Queue configuration.
- A Cluster Queue is automatically of type CHECKPOINTING if the ckpt_list attribute containes at least one reference.
- In 5.x it was possible to use keyword 'all' for queue_list. This is not necessary anymore. A Checkpointing Interface attached to a Cluster Queue is automatically attached to each Queue Instance.

Further information

Specification and implementation details can be found on following page:

http://gridengine.sunsource.net/unbranded-source/browse/~checkout~/gridengine/doc/devel/rfe/cluster_queue.txt?content-type=text/plain