Implementation and Exploitation of Fair Shares in LCG*

Sergio Andreozzi, Gian Luca Rubini, Jeff Templon

May 31, 2006

Contents

1 Introduction **2**

2 Requirements From Experiments **2**
2.1 ATLAS 2
2.2 CMS 3

3 Phase 1 **3**

4 Phase 1 Work at NIKHEF **4**
4.1 Summary of Progress 4
4.2 Group Mapping 5
4.2.1 Concrete Actions 5
4.2.2 VO Subgroups: Careful with Naming 6
4.3 Maui Shares 6
4.3.1 Current Maui Setup 7
4.3.2 Actual Fair-Share Operation 8
4.4 Publishing in the Information Service 10

5 Phase 1 work at CNAF **12**
5.1 Definition of Shares based on a Service Class Model 12
5.2 Configuring LSF Shares 12
5.3 Publishing in the Information Service 13
5.4 Group Mapping 14
5.5 Test Results 15
5.6 Workload Manager System 15

6 Editing Catch-All **19**

7 Appendix: NIKHEF maui.cfg **19**

8 Appendix: NIKHEF Local Notes **21**

*$Id: prior2.tex,v 1.3 2006/05/30 13:18:54 templon Exp templon $

Contributors

This document contains the description of the activity performed by the Job Priorities WG. Besides the authors that effectively edited the present document, the following persons have contributed to the activity: Marco Cecchi (WMS-GPBox interaction), Vincenzo Ciaschini (GPBox/VOMS), Andrea Ferraro (GPBox), Antonia Ghiselli, David Groep (GID mapping strategies), Francesco Giacomini (WMS), Alessandro Italiano (LSF), Oscar Koeroo (LCAS/LCMAPS/VOMS), Dietrich Liko (coordinator), Salvatore Monforte (WMS), Felice Rosso (monitoring), Davide Salomoni, Steve Traylen (Maui fair shares).

1 Introduction

Several experiments have indicated that they would like to define different classes of users within their experiment virtual organization, and define different shares of computing capacity for these groups. For example, ATLAS might assign 70% of its CPU allocation to Monte Carlo production, 20% to official analyses of these data, and 10% to 'any other' work.

In order to realize this, tests need to be done on

- mapping of 'grid stuff' like VOMS FQANs to 'site stuff' like unix GIDs
- how to convince schedulers like Maui to do the right thing with the defined shares
- how to handle publishing of the share information to the experiments, how do they take advantage of it, and how to handle dynamics (change of allocation) in a sensible fashion.

This document is essentially a log of what we (the people in the author list) have figured out.

2 Requirements From Experiments

In this section, we describe the requirements from two LHC experiments: ATLAS and CMS.

2.1 ATLAS

The ATLAS experiment requires the ability to define priorities and shares for different activities. The main activities are 'production' and 'analysis'. For each activity, the VO should be able to allocate dedicated shares of resources. Furthermore, the production jobs are characterized by long running jobs and it is typically a batch activity, while the analysis jobs are characterized by short/medium jobs that should possibly not wait for days in a queue because of the production activity. In Table 1, we describe the VOMS groups/roles defined by ATLAS.

The jobs should have different priorities and access to different shares depending on the submitters credentials VOMS groups and roles. Users with the same group/role should be considered equivalent. The resource allocation should be as dynamic as possible and should be under the control of the VO manager.

Table 1: VOMS Groups/Roles defined by ATLAS

VOMS Group or Role	Description
/atlas	the main group containing all the VO users
/atlas/Role=production	role to be used for production computing activity
/atlas/Role=software	role to be used for VO management operations like installing software
/atlas/Role=lcgadmin	the same as previous

2.2 CMS

In the CMS experiments, users can belong to groups and have special roles. Different groups and roles can be attached to each job user and the following situation should be also considered: a user can submit a production job and after a Susy analysis job; each job should use the dedicated resources to each type of activity. In the near future, O(10) relevant group/role combinations are expected. If resources allocated to a certain type of activity are not used, then it should be possible for other activities to use such resources. Changes to resource allocation must be effective in less than one day. All users are equal if they do the same work. User A cannot block access to site X for other users just because he submitted 10000 jobs. Users can have different priorities depending on the kind of their jobs. In Table 2, we describe the VOMS groups/roles defined by CMS.

Table 2: VOMS Groups/Roles defined by CMS

VOMS Group or Role	Description
/cms	the main group containing all the VO users
/cms/StandardModel	
/cms/HeavyIons	
/cms/Susy	
/cms/Higgs	
/cms/Role=production	
/cms/Role=lcgadmin	

3 Phase 1

In order to get up and running quickly, and not start designing too much stuff before we have clearly defined the relevant issues, the following sequence of tests has been defined in a discussion of the EGEE Job Priorities Working Group.

1. come up with a prototype scheme for mapping VOMS groups onto a set of pool accounts with the right set of primary (and secondary?) unix gids.

2. come up with a prototype mapping of the relevant GIDs to the right set of Maui groups / accounts / shares

3. implement two queues for the relevant test VOs (ATLAS and CMS), one long and one short, and enable access to these queues by the proper GIDs defined above.

At this point, the experiments can submit jobs to a hard-coded list of such queues and we get feedback about the prototype fair share system and 'training material' on how to improve it.

4. publishing in the information system: use the "vomap" configuration section of the lcg-info-dynamic-scheduler to map VOMS FQANs to unix groups (or vice versa) for the VOView stuff. Requires some coding to the dynamic scheduler plugin, may also require other changes to software that assumes that all GlueCEAccessControlBaseRules look like `vo: voname`.

   This may also be a good point to consider switching over to new-style VO names like /atlas.ch ....

5. WMS matching of jobs by VOView information. Requires implementation of the scheme on the PPS plus backporting of Glue 1.2 support to the WMS on gLite 3.0.

at this point, a new class of tests is enabled, to see how accurate WMS scheduling might be with the new ERT stuff. Never really tested before.

4 Phase 1 Work at NIKHEF

We describe what has been done and what the plans are, following the structure of the steps defined above. By definition, we can't get further than step 4) until the Glue 1.2 patches, plus Salvatore's VOView mechanism, has been incorporated into the WMS. However we can try all the stuff listed in italics!

4.1 Summary of Progress

For those who don't want to wade through the technical information: as of 30 May 2006, at NIKHEF we have

1. created the prototype scheme for mapping groups onto VOs (see below)

2. created a prototype mapping of GIDs onto shares. At the moment this is only amongst VOs; it makes no sense to do the subgroups since none of the production users are submitting under VOMS proxies.

3. not implemented separate short and long queues yet

4. implemented the changes to `lcg-info-dynamic-scheduler` that are needed to publish VOViews corresponding to VOMS FQANs. This information provider has been run in standalone mode, but has not yet been connected to the production information system.

These correspond directly to steps 1-4 of Sec. 3 above. Jobs have been submitted under a VOMS proxy to NIKHEF and they ran correctly as a user with the "special" GID corresponding to the VO subgroup. Step 3 has not been done but is irrelevant until jobs start to appear, and it is also not quite clear how "long" the "short" queue should be. The last step, using the RB to match on this information, is not yet possible. All these steps have been done for the ATLAS VO, and using the NIKHEF production cluster.

4.2 Group Mapping

Start with ATLAS. Define three new account pools with corresponding primary GIDs; leave old 'atlas' account pool around, will be used for the moment whenever a job is sent without VOMS, as well as when a jobs VOMS proxy does not match any of the three defined groups.

NOTE Pool ***groups*** are being used ... this means that **each user in the relevant VOMS class will be mapped into an individual account, each of which has the same specific GID**. This must be standard practice (also in YAIM!) in dealing with VOMS groups.

For the moment assign `atlas` as secondary group. This might be needed in order to get access to ATLAS `VO_SW_DIR` in case any files are not world-readable. If the software is world readable, then we could probably omit the secondary group. This has been tested; a secondary group membership is sufficient for the read permissions to work right.

It may be necessary to add the proper magic to the LCMAPS groupmapfile and gridmapfile; LCAS GACLs are also needed in some cases, these can be generated using the command `edg-lcas-voms2gacl`. It's not clear whether we will need them (depends on being in a transition stage between LDAP and VOMS based VOs). So far they have not been needed.

4.2.1 Concrete Actions

Originally the following was tried:

- mapped `/VO=atlas/GROUP=/atlas/ROLE=production` to unix group `atlb` via the gridmapfile-local mechanism.
- added `atlb` accounts to `atlas` group (so this is now a secondary group for those accounts.
- added group `atlb` to ACL for `atlas` queue in torque server.

Given the way LCMAPS is configured by default, the gridmapfile mechanism did not work. So instead of the steps above, the mapping information needed to be entered directly into the LCMAPS configuration, as follows: add the line

```
"/VO=atlas/GROUP=/atlas/ROLE=production"        atlb
```

to the file `/opt/edg/etc/lcmaps/groupmapfile`, and the line

```
"/VO=atlas/GROUP=/atlas/ROLE=production"        .atlb
```

to the file `/opt/edg/etc/lcmaps/gridmapfile` .

The VOMS proxy was generated using the following command:

```
voms-proxy-init -voms atlas:/atlas/Role=production
```

which generated a proxy with the following attributes (displayed via the command `voms-proxy-info -all`:

```
bosui:~> voms-proxy-info -all
subject   : /O=dutchgrid/O=users/O=nikhef/CN=Jeffrey Templon/CN=proxy
issuer    : /O=dutchgrid/O=users/O=nikhef/CN=Jeffrey Templon
identity  : /O=dutchgrid/O=users/O=nikhef/CN=Jeffrey Templon
type      : proxy
strength  : 512 bits
path      : /tmp/x509up_u500
timeleft  : 11:59:46
VO        : atlas
subject   : /O=dutchgrid/O=users/O=nikhef/CN=Jeffrey Templon
issuer    : /C=CH/O=CERN/OU=GRID/CN=host/lcg-voms.cern.ch
attribute : /atlas/Role=production/Capability=NULL
attribute : /atlas/Role=NULL/Capability=NULL
attribute : 22/Role=22/Capability=22
attribute : 45/Role=45/Capability=45
timeleft  : 11:59:45
```

This combination results in my proxy being mapped to the new secondary ATLAS group (and associated pool accounts), meaning that this group could now be used to arrange for `ROLE=production` users to receive a different fair share within ATLAS.

4.2.2 VO Subgroups: Careful with Naming

Originally three subgroups for ATLAS were created at NIKHEF: `atla`, `atlb`, and `atlc`. JT was briefly very proud of hisself until one of his collegues started to laugh, at which point he spontaneously remembered how pool accounts work: `atla001` is a wonderful valid pool account for pool `.atla`. `atlas001` is every bit as valid, but considerably less wonderful as it belongs to a different GID. Bottom line: be careful with how you name the VO subgroup pools.

4.3 Maui Shares

Map the primary groups to Maui `GROUP`s; bundle these groups together into `QOS`es that model VOs. Need to play with the fair-share weighting; should be that

$$\text{QOSWEIGHT} > \text{GROUPWEIGHT} > \text{USERWEIGHT}$$

Reasoning is that it is much more important that VO shares (relative usage of LHCb vs ATLAS) are balanced than it is that the three ATLAS subgroups are in the proper proportion. So the `QOS` (or VO) weighs heavier. Otherwise we could have the situation that the scheduler allows ATLAS production jobs to run in an attempt to get the FS ratio production/analysis correct within ATLAS, even though this causes ATLAS to take more than their fair share relative to LHCb. A similar argument applies for the relationship `GROUPWEIGHT` vs `USERWEIGHT`. The factors by which these should differ are not yet determined, experience is needed.

There are similar issues with `QUEUETIMEWEIGHT` and `XFACTOR` that have to do with how long a job is sitting in the queue and how long the job is expected to take once it's running. Again we need experience to set these correctly.

We used QOS since there is a standard LCG component that assigns an ACCOUNT to each job; even though this ACCOUNT has no function and could be easily removed, we preferred a solution that required as few changes as possible, so QOS was used instead.

4.3.1 Current Maui Setup

At the moment, Maui shares operate in "absolute" mode, meaning that the priority component associated with fair share is proportional to (FS target) - (usage). There are good reasons to try to switch to "relative" mode, meaning that the priority component should be proportional to

$$\frac{(\text{FStarget}) - \text{usage}}{\text{FStarget}}$$

The information below assumes that Maui is being used in absolute mode. If relative mode becomes available we will update the document.

Here are the relevant parts of the Maui config, interspersed with comments.

```
# Priority Weights
QUEUETIMEWEIGHT          2
XFACTORWEIGHT           10
XFACTORCAP          100000
RESWEIGHT               10

CREDWEIGHT              29
USERWEIGHT              10
GROUPWEIGHT             10

FSWEIGHT                70
FSUSERWEIGHT             1
FSGROUPWEIGHT            2
FSQOSWEIGHT             32
```

These specify the weights of the various rank components. The main idea behind the choices here is that the PRIORITY component establishes a general layering amongst the various job types; other factors like fair share usage, time spent in queue, expected job run time, influence the rank with respect to the base level.

At NIKHEF, dteam has a PRIORITY of 5000, meaning that a base overall rank level of 1450000 (CREDWEIGHT $\times$ GROUPWEIGHT $\times$ PRIORITY, or $29 \times 10 \times 5000$) is set. All the LHC VOs (as well as D0) have a PRIORITY of 100, meaning an overall rank of 29000 (so dteam always wins!). The biomed VO has a PRIORITY of 10, resulting in a base rank of 2900. It is also possible to assign PRIORITY at the user level but we do not do this.

The overall scale for the "service" component (queue times, *etc.*) is left at the default of 1; the PRIORITY component is multiplied by 29 (CREDWEIGHT); and the fair share component is multiplied by 70 (FSWEIGHT).

Note as well that the relative FS weights (the last three lines) are not yet optimal, the GROUP weight needs to be larger.

```
FSPOLICY              DEDICATEDPES
```

The DEDICATEDPES above means that what is counted in the fair share is the wall time during which a job sits on a WN. Sites that allow more than one job per CPU may need to do something else here.

```
FSDEPTH                 24
FSINTERVAL              24:00:00
FSDECAY                 0.99
FSCAP                   100000
```

These lines mean that fair shares are computed over a 24 day period, where each day in the past is weighted at 0.99 relative to the previous one.

```
USERCFG[DEFAULT]        FSTARGET=7          MAXJOBQUEUED=350
GROUPCFG[DEFAULT]       FSTARGET=1          PRIORITY=1      MAXPROC=132
```

Basic defaults that probably still need tuning, but are needed in order to get e.g. per-user FS to work.

```
GROUPCFG[dteam]   FSTARGET=2      PRIORITY=5000   MAXPROC=32

USERCFG[atlas081] FSTARGET=1-     PRIORITY=1      MAXPROC=1     QDEF=lhcatlas
GROUPCFG[atlas]   FSTARGET=54     PRIORITY=100    MAXPROC=200   QDEF=lhcatlas
GROUPCFG[atlsgm]  FSTARGET=5      PRIORITY=100    MAXPROC=200   QDEF=lhcatlas
QOSCFG[lhcatlas]  FSTARGET=54                     MAXPROC=200
```

Here you can compare the setup for ATLAS to that of dteam. Note that while dteam has very high priority, the fair share is low and more than 32 simultaneous jobs is forbidden. We've given the general ATLAS group the full ATLAS share, while the SGM group gets only a tenth of that. Both are assigned to the QOS lhcatlas which also has a 54% share. Additional VOMS groups would be supported by simply creating more GROUPCFG lines with the relevant shares. We also have a USERCFG setting an extremely low priority for one notorious ATLAS user who was doing unauthorized work.

```
GROUPCFG[lhcb]    FSTARGET=36     PRIORITY=100    MAXPROC=200   QDEF=lhclhcb
GROUPCFG[lhcbsgm] FSTARGET=4      PRIORITY=100    MAXPROC=200   QDEF=lhclhcb
QOSCFG[lhclhcb]   FSTARGET=36                     MAXPROC=200

GROUPCFG[dzero]   FSTARGET=37     PRIORITY=100    MAXPROC=200
GROUPCFG[biome]   FSTARGET=1-     PRIORITY=10     MAXPROC=20
```

These last two lines illustrate how to handle VOs that are allowed to soak up excess cycles: don't give them a QoS. Dzero has the same base rank as the other VOs; the FS component for the VO as a whole (QoS) is zero. Hence as long as the LHC VOs have not yet reached their FS targets, their QoS rank component will cause them to rank higher than Dzero. Once the LHC VOs reach their fair share (or stop submitting), Dzero's rank becomes competitive.

4.3.2 Actual Fair-Share Operation

Here is an example from actual operation, of the production cluster, on 11 May 2006 at NIKHEF. When this snapshot was taken, the following population of running and queued jobs were present:

```
 account    running  queued
 atlas-1     88     681
 atlas-2      4       0
 atlas-3     11       0
 biome-1     14      16
 biome-2      0       3
   cms-1      6       4
 dteam-1      0       1
 dzero-1      0       8
  lhcb-1    112       0
 lhcbsm-1     9       0
    total    244     713
```

Each line corresponds to a different grid user, and the names correspond to unix groups (lhcb and lhcbsm are different GIDs but in the same VO, see above).

```
group  PRIORITY*   Cred( User:Group)    FS( User:Group:  QOS)  Serv(QTime:XFctr)
Weights  ------      29(   10:   10)    70(    1:    2:   32)     1(    2:   10)

dteam   1450812    99.9(  0.0:5000.)   0.1(490.0:279.5:  0.0)   0.0( 28.1: 14.0)
atlas     67933    42.7(  0.0:100.0)  56.6(-633.:2791.:36290)   0.7(473.6: 10.5)
dzero     30613    94.7(  0.0:100.0)   3.7(487.7:640.3:  0.0)   1.6(474.4: 10.5)
cms        5149    74.0(  0.0: 20.0)  17.1(369.7:-100.:-1611)   8.8(681.2: 10.8)
biomed     1804    58.5(  0.0: 10.0)  31.8(423.6:-2001:  0.0)   9.7(471.8: 10.5)
```

Some relevant information: the dteam user had used 0% of the farm during the averaging period, and dteam as a group had also used 0% (to the printing accuracy of the stats programs) of the farm. The fair-share target for this user is 7% and the dteam group target is 2% ; there is no QoS or "VO" share for dzero. The resulting fair-share priority components are $70 \times 7 = 490$ (user component) and $70 \times 2 \times 2 = 280$ (group part).

To compare, the ATLAS user above had used 16.05% of the farm over the measured period, and the group ATLAS had used 34.06%, and the VO as a whole (normal ATLAS plus sgm users) had used 37.80%. The associated fair-share targets were 7%, 54%, and 54%. This results in priority components of $70 \times (7 - 16.05) = -633$ (user part), $70 \times 2 \times (54 - 34.06) = 2791$ (group part), and $70 \times 32 \times (54 - 37.80) = 36288$ (VO or QoS part).

Dzero has the same "Cred" (called `PRIORITY` in the Maui config file) as does ATLAS but has no guaranteed VO fair share – Dzero sucks up spare cycles. Due to the missing VO component, the overall rank of Dzero is about half that of ATLAS. Of course once ATLAS meets its VO fair-share target, the ATLAS VO component becomes zero as well, and Dzero can compete effectively for cycles.

CMS is not a NIKHEF supported experiment (Dzero is), so it gets lower "cred" and hence comes in lower than dzero, even though the total value of their fair-share components are comparable.

The biomed people have even lower "cred" since HEP gets top priority. This ranks them at the bottom.

A final note: in each line, the number that directly precedes an open-parenthesis is the percentage for that component, for that job. To be clear, of the ATLAS total rank of 67933, 42.7% of that comes from the "Cred" component, 56.6% comes from the fair-share component, and the rest from the "Serv" component. The main point here is that for

VOs like ATLAS, about half of the ranking is determined by the fair-share, given the setup we have, whereas for the other VOs the rank is mostly determined by the value of `PRIORITY`.

4.4 Publishing in the Information Service

The main idea is to define as few queues (or Computing Elements) as possible, and leave the task of VO (or VOMS) snapshots to the VOViews. Indeed this is the entire rationale behind the development of the VOView concept. To first order we don't define any new queues until a need for them has been concretely demonstrated.

```
dn:
GlueCEUniqueID=hostname:2119/jobmanager-lcglsf-short,mds-vo-name=local,o=grid
...
GlueCEAccessControlBaseRule: VO:atlas
GlueCEAccessControlBaseRule: VO:cms
GlueCEAccessControlBaseRule: VOMS: /cms/Higgs
GlueCEAccessControlBaseRule: VOMS: /atlas/Role=production

dn:
GlueVOViewLocalID=atlas,GlueCEUniqueID=hostname:2119/jobmanager-lcglsf-short,m
 ds-vo-name=local,o=grid
...
GlueCEAccessControlBaseRule: VO:atlas
GlueCEStateEstimatedResponseTime: 21534

dn:
GlueVOViewLocalID=atlas_role_production,GlueCEUniqueID=hostname:2119/jobmanage
 r-lcglsf-short,mds-vo-name=local,o=grid
...
GlueCEAccessControlBaseRule: VOMS:/atlas/Role=production
GlueCEStateEstimatedResponseTime: 723

dn:
GlueVOViewLocalID=cms,GlueCEUniqueID=hostname:2119/jobmanager-lcglsf-short,mds
 -vo-name=local,o=grid
...
GlueCEAccessControlBaseRule: VO:cms
GlueCEStateEstimatedResponseTime: 31504

dn:
GlueVOViewLocalID=cms_Higgs,GlueCEUniqueID=hostname:2119/jobmanager-lcglsf-sho
 rt,mds-vo-name=local,o=grid
...
GlueCEAccessControlBaseRule: VOMS:/cms/Higgs
GlueCEStateEstimatedResponseTime: 31504
```

The mechanism for communicating information about the shares is demonstrated here: for ATLAS, the Estimated Response Time (ERT) is lower for the production group than

it is for the VO as a whole, presumably because the production group has yet to use its assigned share, so jobs submitted under this group are running more quickly than those for the entire VO.

In contrast we have the two CMS views reporting the same number, presumably because the Higgs group has already used their share so their jobs are waiting longer. We have the same number being reported twice, due to the current (pessimistic) assumptions made by the ERT info providers: if one says VO: cms, this includes *all* jobs in that VO, including those of the Higgs group. The estimate is generated by finding the longest waiting time of all jobs, so it will not be possible for the VO:cms ERT to be lower than that of one of the subgroups. This is the way the information provider works at the moment, if it is demonstrated that this is not the best approach, the algorithm can be changed. We prefer to leave it as it is until a need to change has been demonstrated in practice.

From a practical viewpoint, the info provider can be configured by using the "vo-map" configuration section of the lcg-info-dynamic-scheduler to map VOMS FQANs to unix groups (or vice versa) for the VOView stuff. It requires some coding to the dynamic scheduler plugin, may also require other changes to software that assumes that all GlueCEAccessControlBaseRules look like `vo: voname`.

Table 3: Computing Differentiated Service Model

Service Class	Share	Description
EXPRESS	1%	the express job has high priority, but small share
GOLD	69%	the gold share is the biggest share of VO resources in a site
SILVER	20%	the silver share is the intermediate share of VO resources in a site
BRONZE	10%	the bronze share is the smallest share of VO resources in a site

5 Phase 1 work at CNAF

In this section, we report on the activity performed at INFN-CNAF. The information service has been configured following the strategy of a CE per VO. This is not a constraint of our proposal, but just an approach to implement a service classes model.

5.1 Definition of Shares based on a Service Class Model

Since the combination of VOMS groups and roles will be in the order of tens in the near future, we propose to define a base service class model to be configured for each VO. This should be suggested to the various sites. The VO will have tools to dynamically assign VOMS credentials to the predefined shares. By coupling this mechanism with policies on Grid accounting, we can achieve a great level of flexibility in the allocation of resources to different users within each VO. The initial service class model that we propose is described in Table 3. It should be noted that the shares defined in the second column are just a proposal. The important characteristic of the model is that there is an order relationship among the shares: $GOLD >> SILVER >> BRONZE$. One more class to be added is the $EXPRESS$ class, with a very limited number of resources (e.g., max 2 running jobs in a site), but high priority (i.e., if the limit of 2 running jobs is not reached and there is one job in this class to be scheduled, this would jump ahead all VO jobs that are queued).

The $EXPRESS$ service is designed to be assigned to the VO software manager. For instance, let us suppose that all ATLAS resources are busy and there are 1000 jobs in the ATLAS queue, if the ATLAS VO manager submits a job, this will be scheduled as soon as there is one free slot. **In the following part, we only consider the Gold/Silver/Bronze service classes. We plan to add later the Express service class**.

5.2 Configuring LSF Shares

For each share of each VO, we configure a pool account. See as follows:

```
Use Case ATLAS:
GID: atlasgold, atlassilver, atlasbronze
UID: atlasgold001, ..., atlasgold050
     atlassilver001, ..., atlassilver050
     atlasbronze001, ..., atlasbronze050

Use Case CMS:
```

```
GID: cmsgold, cmssilver, cmsbronze
UID: cmsgold001, ..., cmsgold050
     cmssilver001, ..., cmssilver050
     cmsbronze001, ..., cmsbronze050
```

The shares have been implemented using a hierarchical fair share configuration. First, it is necessary to define an hosts group, the HostPartition, associated to a VOMS group/role sharing access. The group is also composed by three subgroups where everyone is associated to a second share level. In the following part, we present the configuration file:

lsb.host file

```
cds_nodes     (wn-03-05-01-a)
Begin           HostPartition
HPART_NAME   =  CDS
HOSTS        =  cds_nodes # Specify hosts for the host partition
USER_SHARES  = ([group_cmscds, 40] [group_atlascds, 40])
End HostPartition
```

lsb.users file

```
group_cmsgold     (!) ([default, 1])
group_cmssilver   (!) ([default, 1])
group_cmsbronze   (!) ([default, 1])
group_cmscds (group_cmsgold group_cmssilver group_cmsbronze)
([group_cmsgold, 60] [group_cmssilver, 30] [group_cmsbronze, 10])
```

lsb.queues file

```
Begin Queue

QUEUE_NAME           = cmscds
JOB_ACCEPT_INTERVAL  = 0
PRIORITY     = 40
CPULIMIT     = 3300      # 55 hours
RUNLIMIT     = 3360      # 56 hours walltime
JOB_STARTER = /usr/share/lsf/conf/scripts/jobstarter-lsf-lcg-test.sh
USERS        = group_cmscds
HOSTS        = CDS
DESCRIPTION  = Test queue for debugging jobs. On this nodes login by users is permitted End Queue
```

5.3 Publishing in the Information Service

In our testbed, we have configured a Computing Element per each VO. In each Computing Element, one or more VOViews will publish the several configured shares (a.k.a. service classes) within a specific VO (gold, silver, bronze in our service classes model).

In the scenario where G-PBox is available in production system, it can be avoided to publish the VOMS authorized groups/roles in the information service, since the association between a certain service class and a certain set of VOMS credential is a policy present in the G-PBox. **In this initial phase, since G-PBox is not available in the production system, we propose to publish both service class and VOMS credentials**. The publication of the service class is not essential in this phase to let the system work. However, it is recommended to publish the information of labelled shares following the VO service class definitions. For the time being, we decide to use the attribute `GlueCEAccessControlBaseRule`. We can later decide to use a dedicated experimental attribute (e.g., `LCGCEServiceClass`) to be later proposed for addition in the GLUE Schema.

```
dn: GlueCEUniqueID=hostname:2119/jobmanager-lcglsf-cmscds,mds-vo-name=local,o=grid
...
GlueCEStateEstimatedResponseTime: 500
GlueCEAccessControlBaseRule: VO:cms
GlueCEAccessControlBaseRule: VOMS:/cms/StandardModel
GlueCEAccessControlBaseRule: VOMS:/cms/HeavyIons
GlueCEAccessControlBaseRule: VOMS:/cms/Susy

dn: GlueVOViewLocalID=cmsgold,GlueCEUniqueID=hostname:2119/jobmanager-lcglsf-cms,mds-vo-name=local,o=grid
...
GlueCEStateEstimatedResponseTime: 300
GlueCEAccessControlBaseRule: VOMS:/cms/StandardModel
GlueCEAccessControlBaseRule: SC:gold

dn: GlueVOViewLocalID=cmssilver,GlueCEUniqueID=hostname:2119/jobmanager-lcglsf-cms,mds-vo-name=local,o=grid
...
GlueCEStateEstimatedResponseTime: 200
GlueCEAccessControlBaseRule: VOMS:/cms/HeavyIons
GlueCEAccessControlBaseRule: SC:silver

dn: GlueVOViewLocalID=cmsbronze,GlueCEUniqueID=hostname:2119/jobmanager-lcglsf-cms,mds-vo-name=local,o=grid
...
GlueCEStateEstimatedResponseTime: 200
GlueCEAccessControlBaseRule: VOMS:/cms/Susy
GlueCEAccessControlBaseRule: SC:bronze
```

In the initial phase, we also propose to advertise the authorized FQAN in each VOView by using the `GlueCEAccessControlBaseRule` attribute. We define the prefix `VOMS:` to be used to advertise this information.

5.4 Group Mapping

In this first phase, we use static VOMS credential mapping via LCMAPS as described in the following part:

```
/atlas/Role=production -> atlasgold
/atlas                 -> atlassilver
/atlas/Role=software, /atlas/Role=lcgadmin -> atlasbronze
/cms/StandardModel -> cmsgold
/cms/HeavyIons     -> cmssilver
/cms/Susy, /cms    -> cmsbronze
```

In the following part, we list two sections of the groupmapfile and the gridmapfile respectively.

```
"/VO=atlas/GROUP=/atlas/ROLE=lcgadmin/Capability=NULL" atlasbronze
"/VO=atlas/GROUP=/atlas/ROLE=lcgadmin" atlasbronze
"/VO=atlas/GROUP=/atlas/ROLE=production/Capability=NULL" atlasgold
"/VO=atlas/GROUP=/atlas/ROLE=production" atlasgold
"/VO=atlas/GROUP=/atlas/Role=NULL/Capability=NULL" atlassilver
"/VO=atlas/GROUP=/atlas" atlassilver
"/VO=cms/GROUP=/cms/HeavyIons/Role=NULL/Capability=NULL" cmssilver
"/VO=cms/GROUP=/cms/HeavyIons" cmssilver
"/VO=cms/GROUP=/cms/Higgs/Role=NULL/Capability=NULL" cms02
"/VO=cms/GROUP=/cms/Higgs" cms02
"/VO=cms/GROUP=/cms/StandardModel/Role=NULL/Capability=NULL" cmsgold
"/VO=cms/GROUP=/cms/StandardModel" cmsgold
"/VO=cms/GROUP=/cms/Susy/Role=NULL/Capability=NULL" cmsbronze
"/VO=cms/GROUP=/cms/Susy" cmsbronze
"/VO=cms/GROUP=/cms/Role=NULL/Capability=NULL" cmsbronze
"/VO=cms/GROUP=/cms" cmsbronze

"/VO=atlas/GROUP=/atlas/ROLE=lcgadmin/Capability=NULL" .atlasbronze
"/VO=atlas/GROUP=/atlas/ROLE=lcgadmin" .atlasbronze
"/VO=atlas/GROUP=/atlas/ROLE=production/Capability=NULL" .atlasgold
```

```
"/VO=atlas/GROUP=/atlas/ROLE=production" .atlasgold
"/VO=atlas/GROUP=/atlas/Role=NULL/Capability=NULL" .atlassilver
"/VO=atlas/GROUP=/atlas" .atlassilver
"/VO=cms/GROUP=/cms/StandardModel/Role=NULL/Capability=NULL"
.cmsgold "/VO=cms/GROUP=/cms/StandardModel" .cmsgold
"/VO=cms/GROUP=/cms/Susy/Role=NULL/Capability=NULL" .cmsbronze
"/VO=cms/GROUP=/cms/Susy" .cmsbronze
"/VO=cms/GROUP=/cms/Role=NULL/Capability=NULL" .cmsbronze
"/VO=cms/GROUP=/cms" .cmsbronze
```

5.5 Test Results

In this section, we present a graph showing the initial results of tests conducted at CNAF on LSF batch system. The results were conduct on a test farm where 187 job slots are configured. The two VO's CMS and ATLAS compete for an utilization target of 50% each. The intra-VO sharing is configured using the service class model defined in Table 3. Moreover, VOMS groups are mapped as defined in Section 5.4. Finally, jobs are submitted using different proxies from users of the two VO's. In Figure 5.5, we can see the effect of the overall settings on the resource sharing. A key graph to be added is the behavior of the attribute `GlueCEStateEstimatedResponseTime` for each service class. It is important as it provides the way for the WMS to rank resources available to a certain VOMS group during a matchmaking phase.

5.6 Workload Manager System

The Workload Manager System (WMS) system queries the root information service (BDII) in order to gather a representation of Grid resources to be cached in the Information Supermarket (ISM). Information purchasers acquire information about both Computing and Storage Elements (CE's and SE's). With respect to the Computing Element information, the following objectclasses are involved: `GlueCE`, `GlueCESEBind`, `GlueCluster`, `GlueSubCluster`.

The ISM purchaser has been modified in order to also query the `GlueVOView` objectclass and handling the information about VOView according to the GLUE Schema 1.2 specification. The information gathered is processed and a ClassAd representation of the Computing Element inserted in the ISM. For a given computing Element, all the VOView-related information is processed. For each defined VOView, a ClassAd representation of the Computing Element is generated, merged with the VOView attributes and finally inserted in the ISM. In order to maintain the same space of authorization rules, the splitting between the ClassAd representation of a CE and the relevant views is performed by computing the set intersection of the GlueCEAccessControlBaseRules information.

```
<CE>.GlueCEACBR ? <View i>.GlueCEACBR
```

where <CE> is the ClassAd representing the CE and <View i> is the i-th view defined for such a CE. If some entry in <CE>.ACBR has not been mapped to any VOView defined for such a CE, then a CE Ad with ACBR value equal to the list of such entries is also inserted in the ISM.

As an example let's consider the following scenario where a computing element providing access to tree different VOs has only two of these VOs bound to voviews

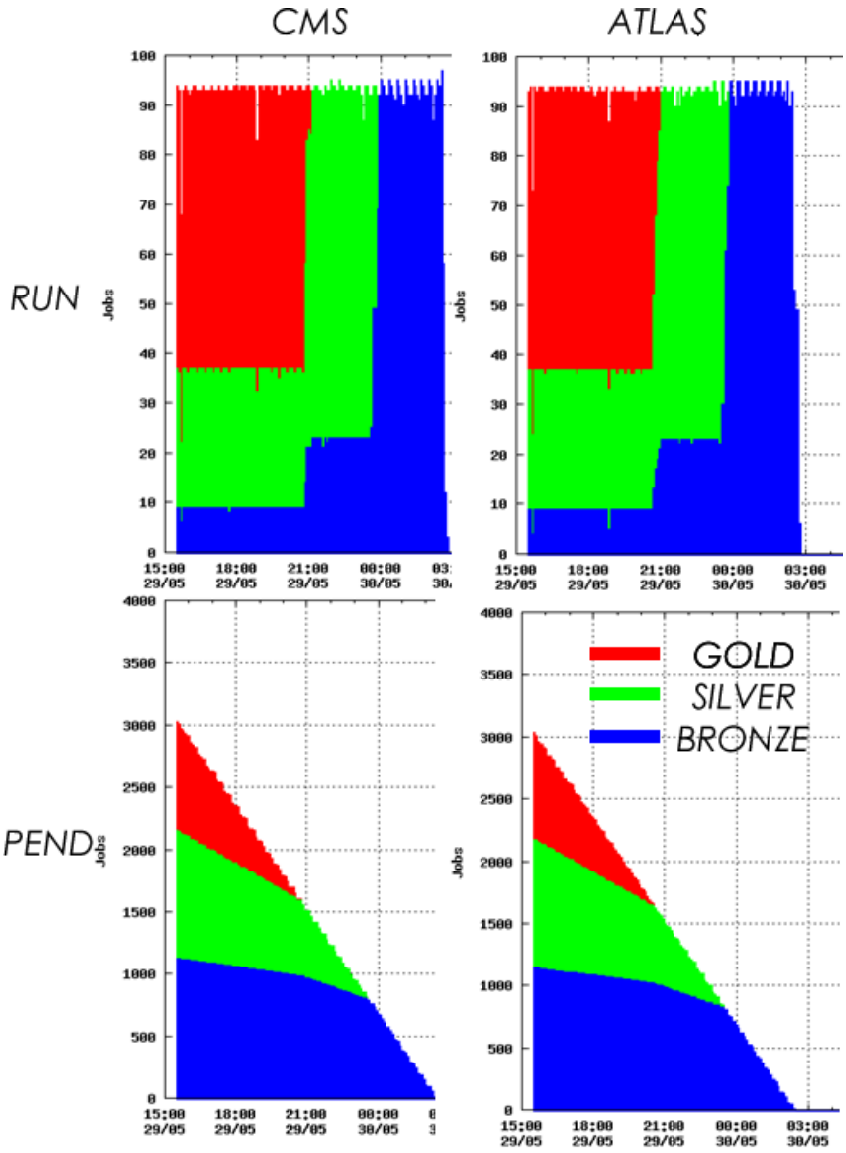

Figure 1: Inter/Intra-VO resource sharing with the Olympic Model

```
dn: GlueCEUniqueID=wn-04-01-03-a.cr.cnaf.infn.it:2119/jobmanager-lcglsf-cms,
    mds-vo-name=local,o=grid
...
GlueCEAccessControlBaseRule: VO:cms
GlueCEAccessControlBaseRule: VO:atlas
GlueCEAccessControlBaseRule: VO:gilda ...

dn: GlueVOViewLocalId=cms-view,GlueCEUniqueID=wn-04-01-03-a.cr.cnaf.infn.it:2119/jobmanager-lcglsf-cms,
    mds-vo-name=local,o=grid
...
GlueCEAccessControlBaseRule: VO:cms
...

dn: GlueVOViewLocalId=atlas,GlueCEUniqueID=wn-04-01-03-a.cr.cnaf.infn.it:2119/jobmanager-lcglsf-atlas,
    mds-vo-name=local,o=grid
...
GlueCEAccessControlBaseRule: VO:atlas
...
```

Clearly, the space of authorization rules the computing element provides, is not enti-

rely covered by the rules the Views supplies with. In such a case, in order to authorize users matching the third access control base rule, also the initial computing element classad specification should be inserted in the ISM and the GlueCEAccessControlBaseRule accordingly modified.

VOMS extension support As agreed, the short term solution is based on a static site configuration and the use of VOViews to publish information of shares or priorities associated to a VOMS group or role. Therefore, we are going to publish ACL rules within the GlueCEAccessControlBaseRule of either the GlueCE or GlueVOView. To perform the actual matchmaking, the gLite Resource Broker relies on the information the ISM supplies with. The matchmaking is performed by generating a symmetric ClassAd match context where the requirement expressions of the submitted requestAd (JDL) and the ClassAd representation of a computing element are evaluated.

The requirement expression defined in the CE ClassAd is the following:

```
... CloseOutputSECheck = IsUndefined(other.OutputSE) || member(other.OutputSE,GlueCESEBindGroupSEUniqueID);

AuthorizationCheck = member(other.CertificateSubject, GlueCEAccessControlBaseRule) ||
                     member(strcat("VO:",other.VirtualOrganisation), GlueCEAccessControlBaseRule);

requirements = AuthorizationCheck && CloseOutputSECheck; ...
```

As can be clearly seen the AuthorizationCheck is performed by checking either whether the certificate subject of the user who submitted the request or the virtual organization he/she belongs to is included in the authorization space defined by the access control base rule attribute in the computing element.

In order to handle ACL/FQAN definition at matchmaking level an ad hoc comparator has been developed and added to the ClassAd functions plugin library. This function is used at ClassAd match level in order to perform filtering of the candidate CEs based on the default FQAN VOMS_FQAN attribute which is included in the JDL.

The authorizationCheck expression has been modified according to use such an extension:

```
...
AuthorizationCheck = member(other.CertificateSubject, GlueCEAccessControlBaseRule) ||
  member(strcat("VO:",other.VirtualOrganisation), GlueCEAccessControlBaseRule) ||
  FQANmember(strcat("VOMS:",other.VOMS_FQAN), GlueCEAccessControlBaseRule);

requirements = AuthorizationCheck && CloseOutputSECheck; ...
```

As an example:

```
dn:GlueCEUniqueID=wn-04-01-03-a.cr.cnaf.infn.it:2119/jobmanager-lcglsf-cms,
   mds-vo-name=local,o=grid
...
 GlueCEAccessControlBaseRule: VOMS:/cms/Higgs/gold/Role=*
 GlueCEAccessControlBaseRule: VOMS:/cms/Higgs/silver/Role=*
 GlueCEAccessControlBaseRule: VO:atlas
...

dn: GlueVOViewLocalId=cms-view-gold,GlueCEUniqueID=wn-04-01-03-a.cr.cnaf.infn.it:2119/jobmanager-lcglsf-cms,
    mds-vo-name=local,o=grid
...
 GlueCEAccessControlBaseRule: VOMS:/cms/Higgs/gold/Role=*
...

dn: GlueVOViewLocalId=cms-view-silver,GlueCEUniqueID=wn-04-01-03-a.cr.cnaf.infn.it:2119/jobmanager-lcglsf-atlas,
```

```
   mds-vo-name=local,o=grid
...
 GlueCEAccessControlBaseRule: VOMS:/cms/Higgs/silver/Role=*
...
```

6 Editing Catch-All

Moved edited stuff here, so as not to forget it later.

6. Dynamics: experiments try changing fair shares and see whether this is 'easy', first via email and later via the G-PBOX.

7 Appendix: NIKHEF maui.cfg

There have been some refinements (based on experience on the production system) of the priority weighting in `maui.cfg` since mid-may, here is the most recent version.

```
# Weights of various components in scheduling ranking calc

QUEUETIMEWEIGHT         2
XFACTORWEIGHT          10
XFACTORCAP         100000
RESWEIGHT              10

CREDWEIGHT             30
USERWEIGHT             10
GROUPWEIGHT            10

FSWEIGHT               20
FSUSERWEIGHT            1
FSGROUPWEIGHT          10
FSQOSWEIGHT           100

# FairShare
# use dedicated CPU ("wallclocktime used") metering
# decays over 24 "days"
FSPOLICY              DEDICATEDPES
FSDEPTH               24
FSINTERVAL            24:00:00
FSDECAY               0.99
FSCAP                 100000

############################################################################
#
# use PRIORITY to define various levels.
# test groups have highest priority, e.g. dteam PRIORITY=5000
# Tier-1 HEP VOs have next PRIORITY, all = 100
# other VOs have less, e.g biomed PRIORITY 10, esr PRIORITY 50,
# geant PRIORITY 80

# USERs in Maui map to real users
# GROUPs in Maui map to unix GIDs which map to VOs or VO subgroups
# QoS in Maui map to VOs (bundle together VO subgroups)
```

```
# other considerations: in 2007, we promised (in kSI2K)
# 159:908:610  to ALICE:ATLAS:LHCb.   This is then
# about 54 : 36 : 9  in percent (leaving one percent floating).
#
# installed capacities
#
# subnet 15:   52.16 kSI2K
# subnet 16:   79.86 kSI2K
# subnet 17:   82.10 kSI2K
# total: 214

# note that fair-share competition between users is absolutely disabled
# unless users have a fair share.  A first guess: use 1%.  This didn't
# work since it meant that any user actually using cycles had a
# rather large discrepancy from the target.  Retry: right now there
# are 14 distinct users with jobs in the queue: set to 1/14.  This means
# that everything else being equal, each user should get an equal share.

USERCFG[DEFAULT]    FSTARGET=7          MAXJOBQUEUED=350
GROUPCFG[DEFAULT]   FSTARGET=1    PRIORITY=1      MAXPROC=132

# the limits applied appear to be a MIN() of all applicable limits

GROUPCFG[users]     FSTARGET=1    PRIORITY=10     MAXPROC=50
GROUPCFG[dteam]     FSTARGET=2    PRIORITY=5000   MAXPROC=32

GROUPCFG[nikalice]  FSTARGET=1    PRIORITY=100                  QDEF=lhcalice
GROUPCFG[alice]     FSTARGET=9    PRIORITY=100    MAXPROC=220   QDEF=lhcalice
GROUPCFG[alicesgm]  FSTARGET=9    PRIORITY=100    MAXPROC=220   QDEF=lhcalice

GROUPCFG[atlas]     FSTARGET=54   PRIORITY=100    MAXPROC=220   QDEF=lhcatlas
GROUPCFG[atlsgm]                  PRIORITY=200    MAXPROC=1     QDEF=lhcatlas
USERCFG[atlas081]   FSTARGET=1-   PRIORITY=1      MAXPROC=1     QDEF=lhcatlas

GROUPCFG[lhcb]      FSTARGET=36   PRIORITY=100    MAXPROC=220   QDEF=lhclhcb
GROUPCFG[lhcbsgm]   FSTARGET=4    PRIORITY=200    MAXPROC=1     QDEF=lhclhcb

GROUPCFG[geant]     FSTARGET=1    PRIORITY=80     MAXPROC=100   QDEF=lhcgeant
GROUPCFG[cms]       FSTARGET=1-   PRIORITY=20     MAXPROC=10    QDEF=lhccms
GROUPCFG[cmssgm]                  PRIORITY=40     MAXPROC=1     QDEF=lhccms

GROUPCFG[dzero]     FSTARGET=37   PRIORITY=100    MAXPROC=220

GROUPCFG[biome]     FSTARGET=1-   PRIORITY=10     MAXPROC=20
GROUPCFG[esr]       FSTARGET=10   PRIORITY=50     MAXPROC=32    QDEF=nlgrid
GROUPCFG[ncf]       FSTARGET=10   PRIORITY=50     MAXPROC=132   QDEF=nlgrid
GROUPCFG[asci]      FSTARGET=10   PRIORITY=50     MAXPROC=132   QDEF=nlgrid
GROUPCFG[pvier]     FSTARGET=5    PRIORITY=50     MAXPROC=12    QDEF=nlgrid
GROUPCFG[vlemed]    FSTARGET=10   PRIORITY=100    MAXPROC=132   QDEF=nlgrid
```

```
# versto: maxproc=132 because of size of NCF farm
USERCFG[versto]     FSTARGET=1-   PRIORITY=1      MAXPROC=132

USERCFG[davidg]                   PRIORITY=500000
USERCFG[templon]                  PRIORITY=500000
USERCFG[ronalds]                  PRIORITY=500000

QOSCFG[lhcalice]    FSTARGET=9                        MAXPROC=220
QOSCFG[lhcatlas]    FSTARGET=54                       MAXPROC=220
QOSCFG[lhclhcb]     FSTARGET=36                       MAXPROC=220
QOSCFG[lhccms]      FSTARGET=1-                       MAXPROC=10
QOSCFG[nlgrid]      FSTARGET=37                       MAXPROC=200

CLASSCFG[qinfinite]               PRIORITY=1
```

8 Appendix: NIKHEF Local Notes

Not really relevant for this document, but relevant for remembering exactly what was done.

VOMSES

The VOMS proxy init did not work intially due to the absence of the VOMS server information. I fixed this by making a file `.edg/vomses` in my home directory and adding the line

```
"atlas" "tbed0152.cern.ch" "15001" \
   "/C=CH/O=CERN/OU=GRID/CN=host/lcg-voms.cern.ch" "atlas"
```

to it. This line has been split at the \ character for formatting convenience, but it is a single unbroken line in the actual file.

Interaction With LDAP

The command to do the secondary group modifications in the LDAP directory is quite tricky. From my laptop using ssh tunneling to the real farmnet server, the command is

```
ldapmodify -H ldaps://localhost:1636/ -W -Z -x -D \
   "cn=Jeff Templon,ou=Managers,dc=farmnet,dc=nikhef,dc=nl" \
   -f tmp.ldif
```

Pushing Profiles

Procedure:

1. edit profiles in private copy of CVS
2. checkin
3. login to `ndpfmgr` account on quattor server
4. `cvs upd` in appropriate directory
5. `pushxprof -f prd -p tbn20`