Abstract. On Grid sites, the requirements of the computing tasks (jobs) to computing, storage, and network resources differ widely. For instance Monte Carlo production jobs are almost purely CPU-bound, whereas physics analysis jobs demand high data rates. In order to optimize the utilization of the compute node resources, jobs must be distributed intelligently over the nodes. Although the job resource requirements cannot be deduced directly, jobs are mapped to POSIX UID/GID according to the VO, VOMS group and role information contained in the VOMS proxy. The UID/GID then allows to distinguish jobs, if users are using VOMS proxies as planned by the VO management, e.g. ‘role=production’ for Monte Carlo jobs. It is possible to setup and configure batch systems (queuing system and scheduler) at Grid sites based on these considerations although scaling limits were observed with the scheduler MAUI. In tests these limitations could be overcome with a home-made scheduler.

1. Introduction
Grid computing has become the key technology for providing computing resources in e-Science. The LHC experiments depend on the world-wide LHC Computing Grid (WLCG)[1] which was set up and is being operated in the context of the projects EGEE and its successor EGI, NorduGrid, and OSG [2]. The same Grid infrastructure is being used very successfully by groups in other fields of e-Science such as non-LHC high energy physics (HEP) experiments, astroparticle physics, life sciences, etc. These collaborations and groups founded Virtual Organizations (VO) – a key concept in Grid Computing [3] – in which the provision and usage of resources is organized based on common sharing rules. VO members use X509 certificates which are issued by Certification Authorities (CA) for authentication. Authorization is managed by the VO Membership Services (VOMS).

The EGI Grid site DESY-HH, which is located at DESY in Hamburg, Germany, operates a multi-VO Grid infrastructure and serves as a Tier-2 center for the LHC experiments ATLAS and CMS within WLCG. The Grid resources are federated among around 20 VOs. They are used according to the agreements and contracts, such as the WLCG Tier-2 Memorandum of Understanding (MoU), in which certain pledges per VO are defined. Additional and temporarily unused resources are distributed opportunistically among all VOs. DESY-HH, as one of the largest Tier-2 centers, provides as of summer 2012 a total of 4800 job slots. In fall 2012 new hardware was added (partly as a replacement for old hardware) to reach a total of 6000 job slots. In order to use the computing resources most efficiently with respect to the usage of CPU cycles, physical and virtual memory, local disk space, and network bandwidth, the computing tasks jobs must be distributed intelligently on compute nodes by the batch system.
We describe here the concept and the technical means to achieve stable operations and guaranteed shares at maximal occupancy of the computing resources.

### 2. Jobs

In HEP the computing requests (jobs) are parallelized at the job level because applications do not (yet) make use of multi-threading technologies and/or use multi-core architectures. Hence, jobs are independent and self-contained and can be treated individually.

In the computing models of the LHC experiments, Tier-2 centers are responsible for Monte Carlo production and for user specific analysis. Those *job classes* have fundamentally different characteristics with respect to their resource requirements as listed in table 1.

**Table 1. Job classification.**

| Purpose          | Class     | Submitter | Characteristics                                      |
|------------------|-----------|-----------|-----------------------------------------------------|
| Monte Carlo      | Production| central   | CPU-dominated, little input, moderate output         |
| Analysis         | Analysis  | users     | I/O-dominated, massive scratch disk usage           |

Jobs are submitted to the Computing Elements (CE) which act as gateways to the batch system. The jobs contain the credentials of the job submitter which are encoded in the X509-formatted VOMS proxy (*job identification*). Individual users as well as centrally managed submitters make use of VOMS groups and roles which are managed by the VOs and are contained in the VOMS proxy. If such VOMS groups and roles are consequently used, job classes can be distinguished.

The CEs map the VOMS proxy to configurable site-specific local accounts (POSIX: UID/GID) by using the VO, VOMS group, and role information of the proxy (*job mapping*). The accounts are chosen uniquely from pools to ensure individuality per user. A typical example for the VO CMS is listed in table 2. These unique users and groups are used throughout all subsequent processes and allow to distinguish classes of jobs. Caution is required with so-called pilot jobs. Such jobs are submitted from a central agent using a standard VOMS proxy with the VOMS ‘role=pilot’. The real ownership of the job payload, which is started by the pilot job, is not revealed before run-time and precludes the batch system from identifying the job class.

**Table 2. Example which shows the association of VOMS groups and roles to accounts.**

| purpose           | VOMS           | VO   | UID     | GID     |
|-------------------|----------------|------|---------|---------|
| user analysis     | /cms           | cms  | cmsusr007 | cmsusr  |
| German user       | /cms/de        | cms  | cmsger032 | cmsger  |
| priority analysis | /cms/role=analysis | cms | cmsana010 | cmsana  |
| software manager  | /cms/role=lcgadmin | cms | sgmcms   | cmssgm  |
| MC production     | /cms/role=production | cms | cmsprd099 | cmsprd  |
| pilot jobs        | /cms/role=pilot | cms  | cmsplt029 | cmsplt  |
| unknown to site   | /cms/any/role=any | cms | cmsusr199 | cmsusr  |
3. Resources

The computing resources are provided by Worker Nodes (WN) which are operated as batch clients of a batch system. Most Grid sites purchase hardware in stages over many years, which leads to rather heterogeneous setups. DESY-HH operates more than 300 WNs including 8 to 24-core Intel Xeon machines with and without hyper-threading (HT) as well as 64-core AMD Opteron hosts. In total 6000 job slots delivering 50 kHS. For each job slot 2 GB of virtual memory and 20 GB of local scratch space is provided. Each WN is equipped with a 1 Gbit/s network link. The optimization of the utilization of the computing resources incorporates three aspects. The top goal of any Grid site is to guarantee stable operations. If farm nodes crash due to exhausted resources, usually many more jobs than the responsible one are effected; at least all jobs on the particular node. In order to minimize costs for procurement, maintenance, and operations, resources should be utilized efficiently. A good measure is the ratio of CPU time and wall time which is preferably close to 1. Even if individual jobs have good efficiency, running many of them on one WN might lead to performance losses:

- Too many I/O intensive jobs per WN might exhaust the network and leave the CPU idle,
- many CPU dominated jobs per WN may lead to a waste of network bandwidth capacities,
- jobs which extensively exploit the scratch space on the local disk might cause high CPU wait times for all jobs on that node.

In particular large VOs such as the LHC VOs have MoUs with the sites to specify resource requirements and to define pledges per VO. This includes the number of guaranteed job slots normalized to specs as well as memory and scratch space per slot. Central accounting information is checked against the pledges monthly and sites are notified of imbalances.

4. Batch Systems

Jobs are submitted to a batch system by a CE using the POSIX UID/GID of the mapping. The batch system queues and schedules jobs according to configurable rules which are based on the jobs’ UID/GID. DESY-HH as well as many WLCG Grid sites deploy TORQUE [4] as the queuing system and the scheduler MAUI [5]. Both these open source products are supported by the gLite/EMI [6] middleware.

At DESY-HH a configuration with four queues was chosen to allow for easy operations and maintenance; it contains one for each LHC VO (ATLAS, CMS), one for all other VOs, and one for monitoring purposes. Table 3 lists the queue properties. This setup allows to manage the two big VOs independently.

The simplest scheduling ansatz is a first-in-first-out (FIFO) algorithm with only one queue. The scheduler would simply try run the jobs in order of appearance. Obviously it is not possible to give priorities in the order or number of jobs of the various VOs and groups in this scenario.

### Table 3. Job queues with the limits.

| Queue   | VO       | CPU time | wall time | max running | max queueable |
|---------|----------|----------|-----------|-------------|---------------|
| atlas   | atlas, ops | 60h      | 90h       | 4000        | 10000         |
| cms     | atlas, ops | 60h      | 90h       | 4000        | 15000         |
| desy    | non-Tier-2 | 60h      | 90h       | 4000        | 10000         |
| operations | ops     | 15min    | 1h        | 50          | 1000          |
Moreover, a congestion of jobs of the same class on a WN could not be avoided. Hence, schedulers have to act more intelligently to obey sharing rules and achieve efficient distribution of jobs. The scheduler MAUI was configured to meet the VO pledges of the used wall time. MAUI also took care of the job class mix per WN by limiting the number of jobs of the same job class in terms of GID per WN. This was achieved by explicitly defining groups and limits in the MAUI configuration. This setup was able to handle up to 4800 running jobs plus up to 10000 queued jobs waiting for execution. Typically an occupancy of above 90% of the job slots was achieved and resource bottlenecks at the WNs could be avoided.

5. The MySched project

Long term experiences have shown that the scheduler MAUI can be configured to apply sharing rules for VOs and groups and concurrently distribute jobs among the WNs for good resource utilization. It turned out though that the complexity of the scheduling algorithm of MAUI as well as the many configuration options are hard to control. At DESY-HH the performance of MAUI did not scale beyond 10000 queued jobs which led to blocked queues and unused job slots; hence low occupancy. In view of the increasing demand for computing resources by the LHC experiments, which will lead to a significant increase of the number of WNs and job slots, an alternative to MAUI was investigated. The goal was to show that optimal resource utilization can be achieved by the ideas discussed in this paper.

5.1. Considerations

DESY-HH has been studying a home-made scheduler implementation which is based on the C-API of TORQUE and replaces MAUI (working title: MySched). It is tailored to the rather simple use-case of HEP which currently does not make use of parallelization on application level and multi-core jobs. This ansatz avoids the need for node reservations which complicates the scheduling process considerably. The following requirements were made to MySched:

- Achieve hundred percent occupancy,
- be light-weight (e.g. short processing times),
- guarantee VO and user/group specific target shares of the wall time,
- fairly (opportunistically) distribute free job slots,
- optimize resource utilization by intelligently distributing jobs to the WNs,
- allow to configure limits, shares, and distribution algorithms.

The main idea is to carry out the scheduling process in two subsequent steps. First, MySched retrieves a list of all jobs from the queuing system. For each job the job number (the smaller the older), the status (running (R) or queuing (Q)), and the VO, user/group (UID/GID) are stored. For the currently running (R) jobs as well as for finished jobs (wall time) in a past period – typically 2 days – usage statistics is generated. The queued (Q) jobs are then reordered such that all users/groups get the target share of running jobs. Therefore jobs (oldest first) of users/groups with the smallest relative number of running jobs are on top of the list. Jobs above the target share reside later in the job list.

In the second step a list of all Worker Nodes is retrieved. For each worker node the number of job slots as well as the user/group names of the running jobs is stored. Jobs are then processed job-by-job from the top of the list down considering their user/group names. Each worker node is tested for the number of free slots and the number of already running jobs of the user/group of the job. If more than one worker node fulfills configured limits, such as the maximal number of jobs of a user/group per node, the one with the smallest number is picked. The job is submitted to this worker node and the list is updated. If no worker node meets the criteria, the job is
rejected. All jobs in the list are processed like this. This scenario allows to fill the job slots such that the user/group shares are obeyed. If there are free slots, also jobs above the shares are submitted (opportunistic usage).

A first implementation was written in C++ and contains roughly 4000 lines of program code. Limits and shares are read at run-time from a configuration file in which the limits per VOs and users/groups are explicitly defined. In addition Worker Nodes can be dedicated to certain queues. The scheduler process is automatically started; typically every minute. The scheduler logs comprehensive information for monitoring purposes.

5.2. Results
The test implementation met the requirements denoted above. Full occupancy at reasonable processing times (O(40s)) could be achieved. The system was successfully operated with up to 6000 running plus 20000 queued jobs without the scaling problems observed with MAUI. Figure 1 shows the job slot usage per VO with MAUI (until February) and MySched (from March) on. Since March almost all job slots are used.

The figures 2 and 3 show running and queuing jobs at a later time (Sep) after adding more Worker Nodes (6000). The fraction of jobs per VO (e.g. ATLAS in 'orange' and CMS in 'red') meet the target shares; while smaller VOs still get resources. Even at times of many queued jobs of one VO all other VOs get their target shares (week 36). At times of almost no queued jobs (week 36/37), a limit on the maximal number of jobs does not allow use all free slots. Such limits are introduced to avoid overloading of storage elements or VO software directory servers. Also periods of maintenance could be handled (week 38/39).

6. Summary
The classification of jobs due to their (assumed) resource requirements on the WNs is essential for an optimal utilization of computing resources in Grid batch systems. The key to job classification are the user credentials (VOMS proxy) of the jobs which can be mapped to distinguishable local accounts (user/group). On the basis of these accounts it is possible for batch systems to intelligently distribute jobs to the compute nodes in order to avoid bottlenecks and exhaustion of local resources. The widely used scheduler MAUI has capabilities to optimize resource utilization. It turned out though that MAUI does not scale under the conditions at DESY-HH. With a home-grown implementation of a light-weight scheduler, which is tailored to the needs at DESY-HH and which is designed to take the considerations presented in this paper into account, it could be shown that high occupancy and optimal resource utilization is possible. It is obvious though that the usage of pilot jobs may spoil the described mechanisms if the job class remains hidden to the batch system.

Acknowledgments
The DESY Grid team would like to thank the DESY Computer Center for their support in installing and operating the Grid infrastructure.

References
[1] WLCG: http://wlcg.web.cern.ch
[2] EGEE: http://www.eu-egee.eu, EGI: http://www.egi.eu, NorduGrid: http://www.nordugrid.org OSG: http://www.opensciencegrid.org
[3] Foster I and Kesselman C 1999 The Grid Blueprint for a New Computing Infrastructure (San Francisco: Morgan Kaufmann Publisher, Inc.)
[4] TORQUE: http://www.adaptivecomputing.com/products/open-source/torque
[5] MAUI: http://www.clusterresources.com/products/maui-cluster-scheduler.php
[6] gLite: http://glite.web.cern.ch and EMI: http://www.emi.eu
**Figure 1.** Job slot usage per VO with MAUI (until Feb) and MySched (from Mar on).

**Figure 2.** Running jobs per VO.

**Figure 3.** Queuing jobs per VO.