Nine months in the life of EGEE: a look from the South

Georges Da Costa
dacosta@irit.fr
IRIT
Paul Sabatier University, Toulouse III
France

Marios D. Dikaiakos
mdd@cs.ucy.ac.cy
Department of Computer Science
University of Cyprus
Nicosia, Cyprus

Salvatore Orlando
orlando@dsi.unive.it
HPC Laboratory. ISTI-CNR of Pisa
via G. Moruzzi 1, 56124 PISA, Italy

***Abstract*— Grids have emerged as wide-scale, distributed infrastructures providing enough resources for always more demanding scientific experiments. EGEE is one of the largest scientific Grids in production operation today, with over 220 sites and more than 30,000 CPU all over the world. A further evolution of EGEE needs to be based on knowledge of deficiencies and bottleneck of the current infrastructure and software. To provide this knowledge we analyzed nine months of job submissions on the South-East federation of EGEE. We provide information on how users submit their jobs:throughput, bursts, requirements, VO. We study the current behavior of EGEE middleware too, by evaluating its performance and the retry policy. We finally show that even if the middleware provides advanced functionality, most submissions are still embarrassingly parallel jobs.**

I. Introduction

Grids are large, distributed computing infrastructures that seek to support resource sharing and coordinated problem solving in dynamic, multi-institutional Virtual Organisations (VOs) [6]. Typical Grid infrastructures comprise large numbers of geographically distributed and heterogeneous resources (hardware and software), belonging to different administrative domains and interconnected through an open network. Grids are quickly gaining popularity, especially in the scientific sector, where projects like *EGEE (Enabling Grids for E-sciencE)* [1], provide and operate the resources required to accommodate large computational experiments with thousands of scientists, tens of thousands of computers, trillions of instructions per second, and petabytes of storage [1]. At the time of this writing, the infrastructure operated by EGEE assembles over 220 sites around the world, thousands of job queues, more than 30,000 CPUs, about 5PB of storage, and supports over 200 Virtual Organizations.

So far, however, the configuration of EGEE resources and services has evolved empirically. Due to the complexity of EGEE's middleware, which consists of numerous cooperating software components operated and maintained by different institutions (middleware services, logical file systems, databases, end-user portals and gateways), it is difficult to have an *a priori* understanding of how the infrastructure is used by different user communities and what are its constraints and bottlenecks. However, as more users are attracted to EGEE, information about submitted jobs, resource usage, etc. can be collected and analyzed in order to provide valuable insights about emerging patterns of Grid use, and to guide future improvements in EGEE's middleware design and configuration.

To derive such insights, we investigate the usage of EGEE as represented by the characteristics of Grid jobs submitted for execution. We retrieve and analyze logs from the *Resource Brokers* (RB) operated by the South-Eastern Europe (SEE) federation of EGEE. This federation manages jobs submitted by EGEE participants from Greece, Switzerland, Serbia, Romania, and others South-Eastern Europe countries. In particular, we examine logs retrieved from the RB located in Cyprus[1] and operated by the University of Cyprus (RB-CY). These logs capture the RB activity during a nine-month period, extending from February 14, 2006 to November 24, 2006. During this period, the broker handled 41,124 jobs and gathered 1.3 GB of log-data. To precise further some of our observations, we analyzed a log-file retrieved from the remaining Resource Brokers[2] of the SEE federation, which are hosted in Athens, Greece, and are operated by GRNET, the Greek Research and Education Network (RB-ATH). Although, the RB-ATH log corresponds to a time-frame shorter than that of the RB-CY (the RB-ATH log extends from November 7, 2006 to January 30, 2007) and maintains fewer details about the life-cycle of Grid jobs, it captures a significantly larger number of submitted jobs (103,265). RB-ATH contains all the available data as no data before the last middleware installation were available. At this time, logs from the other EGEE federations are not yet available.

The remainder of this paper is organized as follows: Section 2 describes briefly the process of job submission in EGEE. Section 3 presents our findings regarding the patterns of jobs submitted to EGEE through the RB-CY and RB-ATH brokers. Section 4 describes the error information captured in our logs and Section 5 examines the performance of EGEE's middleware. We conclude in Section 6.

To the best of our knowledge, this work is one of the

[0]This work was funded by Ercim at CNR-ISTI (Pisa, Italy) and at UCY (Nicosia, Cyprus). This research work is carried out under the FP6 Network of Excellence CoreGRID funded by the European Commission (Contract IST-2002-004265).

[1] **rb101.grid.ucy.ac.cy**

[2] **rb01.egee-see.org**, **rb.isabella.grnet.gr** and **wms01.egee-see.org**

first studies that characterize the workload of a large-scale Grid infrastructure at an extended time-frame. In contrast to previous studies[7], [9], which characterize the use of Grids in generic terms, our work focuses on the complete chain of one specific system, from job requirements to the middleware behavior and performance. Some work is currently in progress to retrieve logs from the whole EGEE infrastructure[3], but this work focuses in the retrieving methods and resulting logs still have to be analyzed to provide insights on the middleware.

In complement to our high-level approach, [11] provides an in-depth study of the job inter-arrival time. This study uses three months worth of data from one EGEE broker and exhibits the self-similarity nature of job inter-arrival time. Authors provide and evaluate models to generate such jobs. Thus [11], in complement to this article, could lead to a complete workload generator for grids.

II. Job submission in EGEE

Grid computing infrastructures are usually large-scale services that enable the sharing of heterogeneous resources (hardware and software) over the Internet. A Grid is organised in Virtual Organisations (VOs) [6], collections of computational and storage resources, application software, as well as individuals (end-users) that usually have a common research area. Access to Grid resources is provided to VO members through the Grid middleware, which exposes high-level programming and communication functionalities to application programmers and end-users, enforcing some level of resource virtualisation. VO membership and service brokerage is regulated by *access and usage policies* agreed among the infrastructure operators, the resource providers, and the resourse consumers.

The European project Enabling Grids for E-sciencE (EGEE) currently supports the largest grid infrastructure in the world, with more than 200 participating sites. EGEE uses the [8] middleware.

Within EGEE there exist several Virtual Organisations (VOs), as for example the Computational Chemistry VO. Users registered within a specific VO obtain credentials for single grid sign-on [3] that enables them to have access to the entire set of resources within (belonging to) that particular VO, despite the fact that such resources span different grid sites across different countries.

Users have access to a User Interface (UI) node for submitting jobs to the Grid, for requesting job status and resources information, and for obtaining the output from completed jobs. In brief, a grid job is usually a set of input files (the *input sandbox*) and an executable that processes the given input on a set of grid resources, according to the user requirements set forth in the Job Description Language (JDL) file that accompanies every grid job submission. The Job Description Language (JDL) is a user-oriented language for describing jobs [4] and the information obtained from a JDL file is taken into account by the Grid Workload Management System (WMS) components in order to schedule and submit a job [12]. A job can have particular user-defined requirements for the resources it needs, such as computational capacity, physical memory capacity, the proximity (network latency-wise) of certain files that will be used as input, and the availability of specific application software. Grid jobs can be classified as CPU-intensive and data-intensive, depending on the type of work performed.

Jobs are submitted from the UI to a Resource Broker (RB), a central (global) grid service. The RB is a component of the distributed Workload Management System (WMS) of a grid infrastructure. The RB performs *matchmaking* by identifying a set of resources that satisfy the job requirements. The matchmaking is done based on data received by querying an Information Index, another central middleware service that provides up-to-date information about the state of grid resources, usually spanning several sites.

If the matchmaking is successful, the job is sent from the RB to the the matching Computing Element (CE) for execution. A Computing Element is at site level and it is comprised of the Grid Gate node and several Worker Nodes (WNs). The services running on the Grid Gate node are primarily responsible for authenticating users, accepting jobs, and performing resource management and job scheduling (the last two services comprise the *batch system*). The Worker Nodes are usually powerful machines in terms of processing power and memory capacity, and are responsible for executing jobs arriving at the site, as dictated by the batch system on the Grid Gate. If a job successfully completes execution, the result is then sent back to the Resource Broker and the user is able to access it from there using the User Interface. A UML diagram depicting the life cycle of a typical Grid job can be seen in Figure 1. The RB manages the whole life-cycle of the job and can keep track of all events that occur during its life-cycle. Therefore, the analysis of RB traces can give valuable insights about the characteristics of Grid jobs, their requirements, their performance, the overhead introduced to job execution by the middleware, job failure rates, etc.

During job scheduling and execution, if any input files are necessary, they are either sent by the user during submission (included in the *input sandbox*), or they are already resident on a Storage Element (SE) and the user needs only to specify their location. This brings us to the central Data Management services: the Replica Catalog holds information about the location of various replicas of a file held at the Storage Elements of various sites, and the File Transfer Service is responsible for replicating files across different Storage Elements that are close to Computing Elements, as needed by various jobs.

In general, the *output sandbox* contains the result of a job after it has run on a CE, and contains a set of files that were specified by the user (e.g. a file that contains what would be the output of the console if a job was running on the user's computer). The entire set of output files from a completed job can either be transferred onto the RB (as part of the *output sandbox*) that the user will collect using the User Interface. Alternatively, the output files can be saved onto a Storage Element and registered with the Replica Catalog so the user

[3] **http://gridportal.hep.ph.ic.ac.uk/rtm/reports.html**

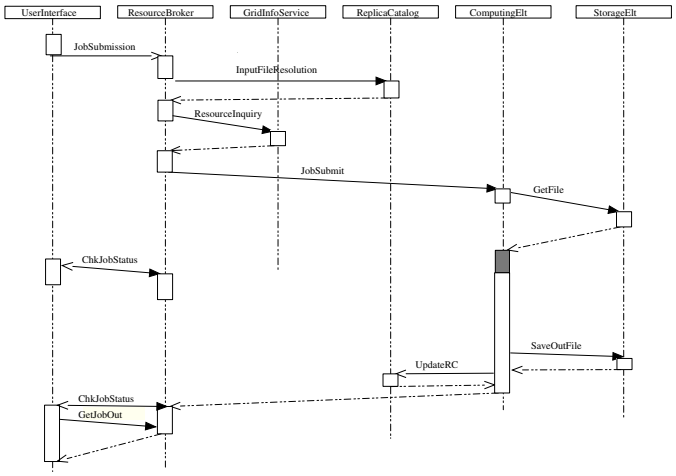

Fig. 1. Life-cycle of a typical Grid job in EGEE.

can access them in the future (most probably these will be very large files of intermediate results that will serve as input to another job).

For more efficient project management, EGEE is divided into different federations/regions, and into each such federation resides a Regional Operations Centre (ROC) that is responsible for supporting and monitoring a set of EGEE-participating grid sites, the Resource Centres (RCs). These divisions into federations are usually organised geographically. As an example, the South East Europe (SEE) federation has the Regional Operations Centre based in Greece and comprises production sites in Bulgaria, Cyprus, Israel, Romania, Serbia, and Turkey. Apart from ROCs and RCs, there is also the EGEE-wide Grid Operations Centre (GOC), responsible for coordinating and monitoring the operation of the Grid infrastructure, and a total of four Core Infrastructure Centres (CICs) which provide monitoring and operational troubleshooting services, acting as second-level support to ROCs.

III. Job characteristics

The two logs of our study (RB-CY and RB-ATH) capture a total of 144,389 jobs. Nearly all the jobs found in these logs are of type *"normal"*. Besides the normal jobs, in our data-set we identified 24 interactive jobs. The next version of the middleware will also support the execution of *"interactive," "collection-type," "parametric"*, and *"DAG-type" (directed acyclic graph)* jobs. These new job types serve two purposes. The first one is to move inside the middleware some work which is repeatedly done by users. The second goal is to provide more information to the middleware about the jobs. For instance, an interactive job should be run with a low latency, parametric jobs can be run in parallel whereas to run a "DAG-type" job, care must be taken to run jobs in the right order. Improving job types thus lead to reduce issues for users, and a the same time improves performances.

Requirement type	Number of jobs	Percentage
BlackList	83,102	58%
MaxCpuTime	80,629	56%
Site	32,299	22%
None	18,287	13%
Library	11,833	8%
WallClockTime	4,457	3%
CpuNumber	1,500	1%
MemorySize	1,334	.9%
other	536	.4%

TABLE I
TYPE OF RESOURCES REQUIREMENTS FOUND IN THE QUERIES.

In our data-set, 121,518 out of 144,389 jobs (84%) are composed of an executable and of input files. There are two ways to provide input files: the first one is to "stage-in" the input files together with the executable; the second, is to upload input files to a logical file system on the Grid, and have the running job accessing these files at runtime through a logical file catalog. The second option was rarely taken in our data-set, with only 295 jobs making use of the logical file catalog.

A. Jobs requirements

As mentioned earlier, the JDL description of a submitted job represents the requirements of this job in terms of Grid resources. For example, the JDL specification for a job may require that $WallClockTime \geq 72h$, i.e., the job should be able to execute without interruption for at least 72 hours, or $Mpich \in Library$, i.e., the job should be scheduled on a site with an installation of the MPICH library. The names of requirement-attributes found in JDL files are taken from the GLUE schema specification, which is adopted by the Information indexes of EGEE [2]. Below, we present an example of a composite resource requirement found in a JDL job specification:

```
GlueCEInfoHostName == "ce101.grid.ucy.ac.cy"
 && Member("MPICH", SoftwareRunTimeEnvironment)
 && GlueCEInfoTotalCPUs >= 4;
```

It is worth noting that in our data-set we found a very small set of distinct requirements: out of the 144,389 jobs, there were only 1,056 jobs with distinct requirements specified; identical requirements share exactly the same resource requirements and associated values. Table I shows the distribution of requirements found in our data-set. A JDL specification defines a *site* in order for the job to run on a particular site, and *blacklist* in order to prevent the job from running on a particular site. *Library* is used when a particular library is needed. This table shows that a large amount of jobs (22%) does not really use the requirements system as they already specify a site name. Some of those (8.5% of the jobs requiring a particular site) still use requirements in order to run on a subset of the site by requesting other characteristics.

The majority of jobs (77%) define requirements that focus on four particular requests: i) The blacklisting of certain sites.

Virtual organization	Number of jobs
biomed	83,065
see	29,180
dteam+ops	19,642
eumed	9,281
others	3,221

TABLE II
NUMBER OF JOBS FOR EACH VIRTUAL ORGANIZATION

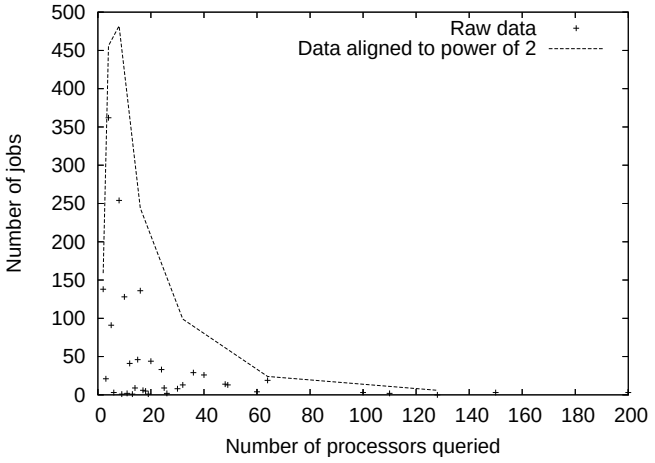

Fig. 2. Repartition of jobs according to the number of processors queried. The curve shows raw data reduced to power of two.

This is used primarily by jobs submitted by the *biomed* Virtual Organization, with the number of blacklisted sites growing over time. During the large-scale computational experiments of the *biomed* VO, failing sites were identified and blacklisted in order to enhance the efficiency of the experiments. (ii) To find sites that accept long-running jobs (specified through the *MaxCpuTime* or *WallClockTime* attribute). (iii) To run on a specific site, and (iv) to guarantee the availability of a particular software library.

It is worth noting that the requirements defined correspond primarily to static characteristics of resource configuration, such as the *WallClockTime* policy or the installation of specific libraries. Also, that the requirements that request a certain performance capacity from the Grid resources, mainly specify the number of requested processors. Figure 2 presents a diagram of the number of jobs that request a particular number of processors. This characteristic seems to follow a long tailed distribution. 61% of these request are for a number of processor that is equal to a power of two. The curve on Figure 2 shows that when the raw values are reduced to powers of two only, they follow the *lognormal* distribution.

B. Virtual organizations

As mentioned earlier, EGEE support over 200 Virtual Organizations. Table II presents the VOs that are actually using EGEE through the two Resource Brokers of our study. The active Virtual Organizations are:

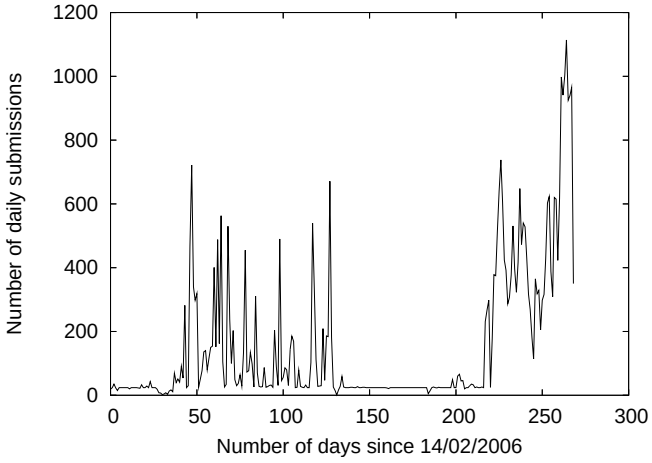

Fig. 3. Number of daily submitted jobs during the 9 months for RB-CY

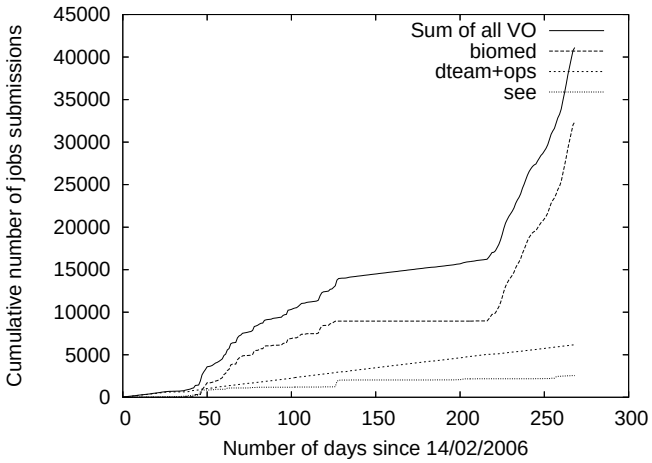

Fig. 4. Cumulative number of submitted jobs during the 9 months for the most used VO for RB-CY

- *Biomed* : Focuses on biomedical applications
- *See* : A generic VO introduced to support scientists from South-east Europe that do not yet belong to another established VO
- *Eumed* : A VO established to support euro-mediterranean collaboration in Grid infrastructures
- *dteam, ops* : VOs established to support administrators who are responsible for deploying and testing middleware, and for operating the EGEE infrastructure
- *others* : Other scientific projects using EGEE, such as ATLAS, Lhcb, ALICE

As we can see from Table II, the south-east federation of EGEE mainly supported production jobs coming from *biomed* and *see* Virtual Organizations which are production VO. Thus this confirms that RB-ATH and RB-CY mainly contain production jobs.

C. Job throughput

Figure 3 shows the number of submitted jobs per day in the 9-month long RB-CY data-set. Two types of behaviors

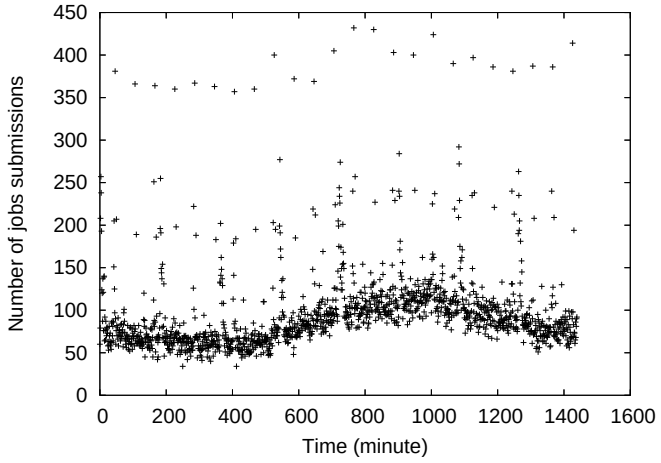

Fig. 5. Number of submission according to the minute of the day using RB-CY and RB-ATH

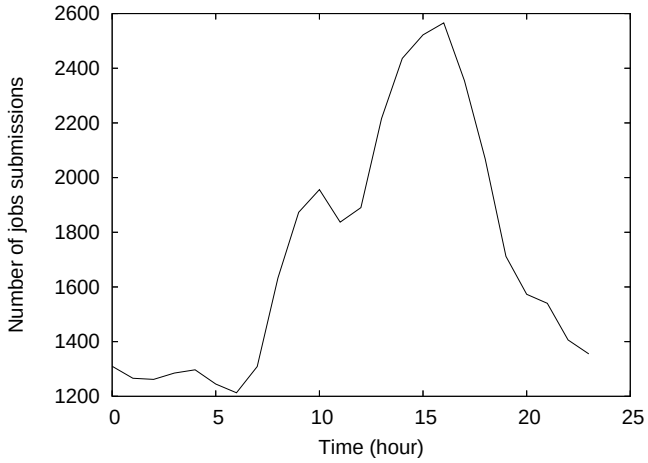

Fig. 6. Number of submission according to the hour of the day using RB-CY

alternate: chaotic with activity bursts, and calm with constant throughput. Grid middleware has to be able to handle large number of jobs submissions in a small amount of time. During peaks, 10 jobs were submitted during one second on one RB, and at other scales, 20 jobs were submitted during ten seconds. Those peaks are of low duration as the maximum number of jobs submitted during one minute is 24, 95 for one hour, and 1,114 during one day.

Figure 4 shows the cumulative throughput per Virtual Organization for the same data-set. The two previous behavior are dependent of the Virtual Organization type. Production jobs, like the ones coming from the *biomed* VO, are more localized in time. Operational jobs coming from the *ops* and *dteam* VOs have a stable throughput. More precisely, *biomed* used EGEE to undertake two "data challenges" during the period captured inside our data-set: the first one, on the Bird Flu during April and May, and the second one finishing at the end of January 2007 on the malaria began in October. As stated in [7], large Grid like EGEE are project driven.

Final state	Number of jobs	Percentage
OK	19334	48%
CANCEL	16658	42%
FAIL	3858	10%

TABLE III
FINAL STATE OF THE JOBS IN RB-CY. 1274 JOBS MARKED AS NONE (I.E. NOT ENOUGH INFORMATION IN THE DATABASE TO CONCLUDE) ARE EXCLUDED.

D. Daily load repartition

We were not able to identify a submission pattern on day-to-day basis. Yet, user behavior during the day are structured. Figure 5 shows the number of jobs according to the minute of its submission for the two set of data. Every hour, several points go out of the mean curve. These points are linked to the use of WMProxy[4] like services: to reduce peak of workload on the RB, the WMProxy can be used as an buffer when a large number of jobs are to be sent to EGEE. Instead of sending jobs directly to the RB, users can choose to send them to this service. It wakes up every hour and, if enough jobs are finished, submits new ones. This leads to the hourly aggregation shown in Figure 5. Currently WMProxy is not deployed on SEE and this technique is applied by hand.

Figure 6 shows an aggregated view of RB-CY with a broader granularity of one hour. This data-set is used to reduce the dispersion due to timezones. It appears that some job submissions are done during the morning, and most of them are done in the afternoon at around 3pm. During the workload peak, submissions are more than doubled compared to laid-back times. This curve follows a distribution similar to the one observed in the context of Web and Peer-to-Peer usage [5]. It shows that even if some proxies can automate submission, a large part of submission are still done by hand.

IV. FAULTS

Complex systems as Grids are prone to errors and failures of various kinds [10]. Table III summarizes the possible final states of jobs in the RB-CY data-set. A job is denoted as *OK* when it finishes successfully and sends back its results. It is denoted as *CANCEL* when the user decided to stop the job at some point. The state *FAIL* means that the job failed for some reason. Finally the state *NONE* describes jobs for which our data-set does not contain enough information. *CANCEL* is mostly used when a job waits during a long time in a queue in order to run on another site. Results in the following sections are extracted from RB-CY as corresponding information are not available for RB-ATH.

As Grids are large, there is a high probability that at some point a problem arises and prevents the job from achieving success. To reduce the number of failing jobs, Grid middleware retries to schedule failing jobs. Figures 7 and 8 show the number of retries used for jobs that respectively succeed and fail.

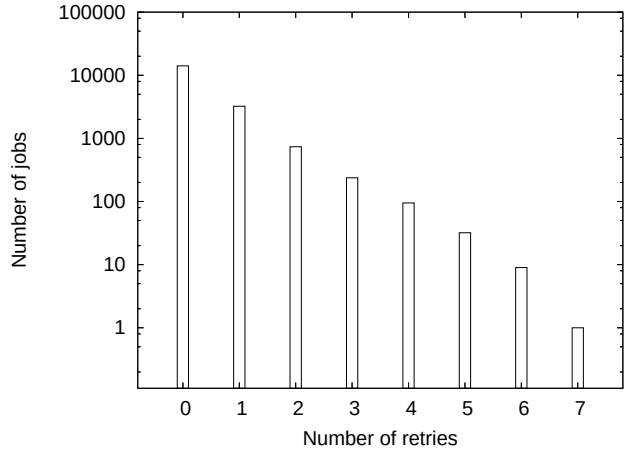

Fig. 7. Number of retries needed for jobs to eventually success in RB-UCY

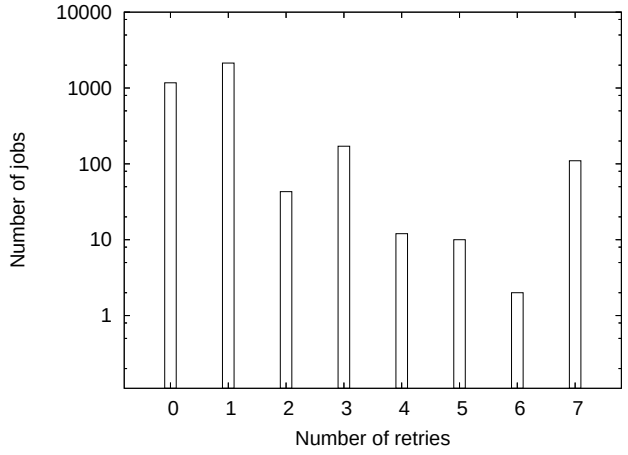

Fig. 8. Number of time the middleware unsuccessfully retries to run jobs which eventually fail in RB-UCY

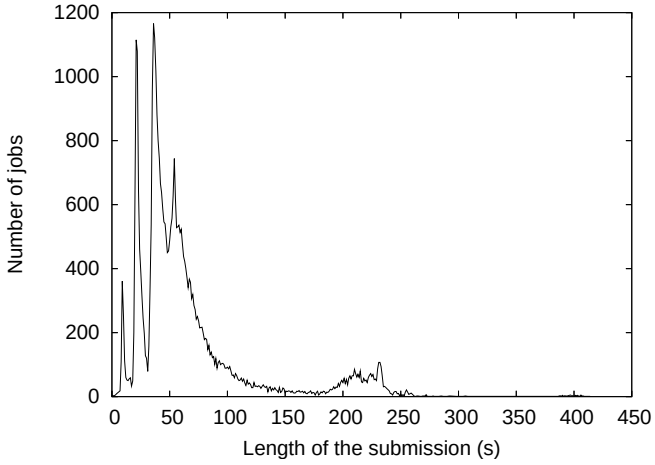

Fig. 9. Time between the job submission and when the job is sent to the local queue

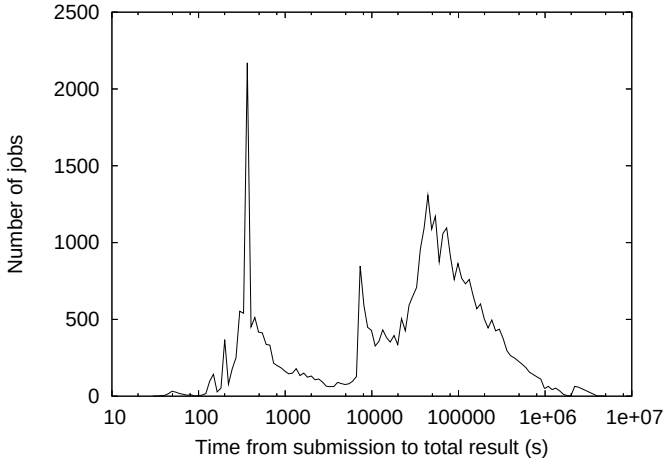

Fig. 10. Time between the job submission and the final completion

Figure 7 represents on a logarithmic scale the number of jobs that need n retries to success. The number of successful jobs decreases exponentially with the number of retries. But as there are usually several retries, 83% of jobs that are executed (i.e. not in *CANCEL* or *NONE* state) succeed.

Figure 8 shows the number of retries used by the jobs that eventually fail. A value of *1 retry* means the job was retried once: there was two failed attempts and the maximum authorized number of retries specified in the JDL was two. The maximum hard-wired value is seven. Most failing jobs are retried at most once.

V. Middleware performance

A preliminary study of EGEE middleware was done in [13], using a simulated workload. In the following, the performance evaluation is based on the real workload of production jobs captured in the RB-CY trace.

Several metrics are usable to evaluate middelware performance, depending on the intended use. For interactive use, latency is primordial, but for normal use, a good site choice is more important. In the following, the metric used will be the latency, in order to provide insight on why there are so few interactive jobs in the data-set.

The complete execution of a job is split in three phases. First the job is sent by the user to the RB which chooses the site where it will be run. The second phase starts when the site is given the job. The site puts it in one of its queues until the necessary resources are available. The last phase is the actual running of the job in the site.

The data processed in this article are obtained in the RB. They provide only timing information of the first phase, plus the ending time of the job. It is sufficient to evaluate the cost induced by the middleware.

Figure 9 shows the time needed by the middleware to choose a queue and to send the job to this queue. There are four high peaks: *9s 21s 36s 54s*. These peaks correspond to the exact RB which treat the request. Each server has different performances which show up on this figure as their hardware and load are different (they provide other services as well). The last peak is farther and smaller, around *230s* and seems to correspond to a timeout. At least, this figure shows how efficient the current

Ratio Managing time/Total time	$\leq 1\%$	$\leq 5\%$	$\leq 10\%$	$\leq 20\%$	$\leq 50\%$
Percentage of jobs	74%	84%	95%	98%	99.9%

TABLE IV

RATIO BETWEEN THE MANAGING TIME AND THE TOTAL TIME UNTIL COMPLETION. THE MANAGING TIME IS BETWEEN THE JOB SUBMISSION AND WHEN THE JOB IS SENT TO A QUEUE.

middleware is. Most jobs are processed under half a minute, and a large majority are under one minute.

Figure 10 shows the total time the jobs are in the system, ie. the middleware time, the waiting time in the queue and the running time. There are several order of magnitude between the times. Peaks are: *6min, 2h, 12h, 72h*. It shows that most jobs run at least one order of magnitude over the middleware time. Moreover, in EGEE *12h* and *72h* are quite common *WallClockTime*s. Each site has a policy and kills jobs which are not finished when the running time attains the *WallClockTime*. Dispersion around those values should be caused by the waiting time in queues before jobs effectively run.

Table IV shows the ratio for each job between the time passed in the middleware and the total time. For most jobs, the middleware time is negligible compared to the total time. Jobs for which this ratio is relatively high are jobs which finish quickly. EGEE seems to be more efficient to manage long jobs compared to really fast ones as the overhead becomes negligible in the first case.

A final remark. There is no correlation between the submission date and the length of the job or between the submission date and the middleware time. Those time are not project-dependant.

VI. CONCLUSION

In this paper we analyze nine months worth traces of a subset of South-East of EGEE. Our goals were to provide insight on how users of EGEE are using the Grid, thus leading to being able to produce realistic workload and showing the lack of local pattern of job submission. The second goal was to evaluate the performance of the current EGEE middleware. We propose a characterization of real requirements and of job throughput. We evaluate the current retry mechanism used by EGEE middleware. We showed the efficiency of this middleware and identified bottlenecks: First many jobs are manually canceled due to bad site choice done by the middleware; Second current expressiveness is not sufficient as most jobs only rely on trivial requirements and use of grid services for retrieving files; Third automatic test of site characteristics would reduce the necessity of blacklisting and of human interaction. Finally we evaluate the middleware performance and compared it to the jobs running time. We demonstrated that submission of jobs are fast, but not enough for interactive submission. Even for non-interactive jobs, users currently use proxies as they found that submitting too many jobs at the same time saturates the middleware.

Acknowledgement: This work was funded in part by the European Commission under the the Sixth Framework Programme through the Network of Excellence CoreGRID (Contract IST-2002-004265) and the Enabling Grids for E-sciencE project (Contract number INFSO-RI-031688). The authors wish to acknowledge Kyriacos Neokleous and Kostas Koumantaros for providing access to the RB-CY and RB-ATH databases respectively, and Nicolas Jacq for providing information about *biomed*.

REFERENCES

[1] Enabling Grids for E-SciencE project. http://www.eu-egee.org/ (last accessed June 2006).

[2] GLUE Schema. http://glueschema.forge.cnaf.infn.it/ (last accessed Jan. 2007).

[3] Internet X.509 Public Key Infrastructure – Certificate and Certificate Revocation List (CRL) Profile. http://www.ietf.org/rfc/rfc3280.txt (last accessed March 2006).

[4] Job Description Language: Attributes Specification. http://edms.cern.ch/document/590869/, May 2006 (last accessed Jul. 2005).

[5] Georges Da Costa, Corine Marchand, Olivier Richard, and Jean Marc Vincent. Resources availability for peer to peer systems. In *The IEEE 20th International Conference on Advanced Information Networking and Applications (AINA)*, 2006.

[6] I. Foster, C. Kesselman, and S. Tuecke. The Anatomy of the Grid: Enabling Scalable Virtual Organizations. *International Journal of High Performance Computing Applications*, 15(3):200–222, 2001.

[7] Alexandru Iosup, C. Dumitrescu, Dick Epema, Hui Li, and Lex Wolters. How are real grids used? the analysis of four grid traces and its implications. In *The 7th IEEE/ACM International Conference on Grid Computing (Grid2006)*, pages 262–269, 2006.

[8] E. Laure, F. Hemmer, et al. Middleware for the Next Generation Grid Infrastructure. In *Computing in High Energy and Nuclear Physics (CHEP) 2004*, Interlaken, Switzerland, September 2004.

[9] Emmanuel Medernach. Workload analysis of a cluster in a grid environment. In *11th International Workshop on Job Scheduling Strategies for Parallel Processing, JSSPP 2005*, pages 36–61, 2005.

[10] Kyriacos Neocleous, Marios Dikaiakos, Paraskevi Fragopoulou, and Evangelos Markatos. Failure management in grids: The case of the egee infrastructure. Technical Report TR-0055, Institute on System Architecture, CoreGRID - Network of Excellence, July 2006.

[11] Michael Oikonomakos, Kostas Christodoulopoulos, and Emmanouel (Manos) Varvarigos. Profiling computation jobs in grid systems. In *CCGRID '07: Proceedings of the Seventh IEEE International Symposium on Cluster Computing and the Grid*, pages 197–204, Washington, DC, USA, 2007. IEEE Computer Society.

[12] F. Pacini. gLite Workload Management System service. https://edms.cern.ch/document/572489/, May 2006 (last accessed June 2007).

[13] N Svraka, A Balzs, A Belic, and A Bogojevic. glite workload management system performance measurements. In *IV INDEL*, pages 294–297, Banjaluka, 2006.