Estimation of the latency of a production grid over several weeks

Diane Lingrand[1], Johan Montagnat[1] and Tristan Glatard[1,2]
[1]RAINBOW team - I3S UMR 6070 - Univ. Nice - Sophia Antipolis / CNRS
B.P. 145 - F 06903 Sophia Antipolis Cedex - France
[2]University of Amsterdam - Institute of Informatics/Academic Medical Center
Diane.Lingrand@unice.fr

ABSTRACT

In this paper, we study grid job submission latencies. Together with outliers, the latency highly impacts performances on production grids, due to its high values and variations. It is particularly prejudicial for determining the expected duration of applications handling a high number of jobs.

In previous work, a probabilistic model of the latency has been used to estimate an optimal timeout value considering a given distribution of jobs latencies. This timeout value is then used in a job resubmission strategy.

The purpose of this paper is to evaluate to what extent updating this model with relevant contextual parameters can help to refine the latency estimation. In the first part of the paper, we study the validity of parameters along several weeks. Experiments on the EGEE grid show that performance can be improved by the update of model parameters. In the second part, we study the influence of the resource broker or the computing site and the day of the week. We experimentally show that some of them have a statistically significant influence on the jobs latency. We exploit this contextual information in the perspective of proposing a reliable job submission strategy.

Keywords

grid computing, model, context

1. INTRODUCTION

Grids are powerful tools for large scale medical studies and specifically for analyzing medical images [1]. These tools allow compute intensive processings such as bio-modeling (cardiac modeling, MRI simulation [2]) and simulation for surgery planning. With grids, large database can be processed e.g. to build personalized atlases or to conduct epidemiological and statistical studies [12, 5]. Image indexing and retrieval similarly benefit from federated databases [13].

However, the behavior of production grids is highly variable and they are subject to faults. In particular, the latency, measured as the time between the submission of a job and its running start, is very high (mean of 5 minutes in our experiments) and highly variable (standard deviation in the order of 5 minutes). Compared to other application domains, the biomedical field applications involves mainly relatively short jobs (execution time in the order of few minutes) thus leading the latency to have a high impact on performances. The aim of this work is to study the latency of the grid in order to improve job submissions and estimation of performance.

We adopt a probabilistic approach for capturing the variability of production grids. Similar probabilistic models have been proposed to estimate timeouts in other complex systems [18, 10]. Previous works [8] have shown the validity of such a model to capture the overall behavior of a large scale grid infrastructure. However these studies have been made on a short period of time thus hiding the temporal variability of workload conditions. In this paper, we conduct our study during several weeks in order to study the model temporal validity.

We also aim at refining our grid model with more local and dynamic parameters from the execution context. Each job can be characterized by its execution context that depends on the grid status and may evolve during the job life-cycle. The context of a job depends both on parameters internal and external to the grid infrastructure. The internal context corresponds to parameters such as the computers involved in the Workload Management System (WMS) of a specific job. It may not be completely known at the job submission time. The external context is related to parameters such as the day of the week and may have an impact on the load imposed to the grid. Preliminary works [6, 11] have shown the validity of this approach by extracting parameters that have an influence on the latency.

Our final goal is to improve jobs execution performance on grids. This requires taking into account contextual information and its frequent update.

2. RELATED WORKS

Several initiatives aim at modeling grid infrastructure Workload Management Systems (WMS). In [9], correlations between job execution properties (job size or number of processors requested, job run time and memory used) are studied on a multi-cluster supercomputer in order to build models of workloads, enabling comparative study on system design and scheduling strategies. In [15], authors make predictions of batch queues waiting time which improve the total execution time.

Taking into account contextual information has been reported to help in estimating single jobs and workflows execution time by rescheduling. Feitelson [4] has observed correlations between run time and job size, number of clusters and time of the day. In [14], the influence of changes in

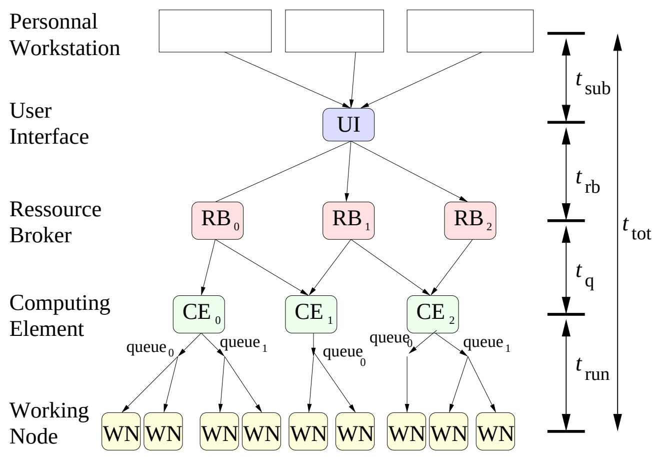

Figure 1: EGEE job life cycle

transmission speed, in both executable code and data size, and in failure likelihood are analyzed for a better estimation of end time of sub-workflows. This information is used for re-scheduling jobs after fault or overrun.

Authors of [16] analyze job inter-arrival times, waiting times in the queues, execution times and data exchange sizes. They conducted experiments on the EGEE[1] grid on several VOs (Virtual Organizations) and studied the influence of the day of the week and the time of the day. Their conclusion is that there is an increase of the load at the end of the day but that it is difficult to extract a precise model of this behavior with respect of the day or the time.

To refine grid monitoring, [17] presents a model of the influence between the grid components and their execution context (system and network levels), experimented on the Grid'5000 French national platform.

3. EXPERIMENTAL PLATFORM

Our experiments are conducted on the biomed VO of the EGEE production grid infrastructure. With 40,000 CPUs dispatched world-wide in more than 240 computing centers, EGEE represents an interesting case study as it exhibits highly variable and quickly evolving load patterns that depend on the concurrent activity of thousands of potential users. Even if the infrastructure is relatively homogeneous from the OS point of view (all host are running RedHat based Linux), important architecture and performance variations are expected among the worker nodes (64/32 bit machines, single/multiple processors, different speeds, single/multiple cores).

On the EGEE grid, when a user wants to submit a job from her workstation, she connects to an EGEE client known as a User Interface (see figure 1). A Resource Broker (RB) queues the user requests and dispatches them to the available computing centers. The gateway to each computing center is one or more Computing Element (CE). A CE hosts a batch manager that will distribute the workload over the Worker Nodes of the center. Different queues handle jobs with different forecast wall clock times. The policies for deciding of the number of queues and the maximal time assigned to each of them are site-specific.

[1] `http://www.eu-egee.org`

During its life-cycle, a job is characterized by its evolving status. If everything happened as expected, the job is then *completed*. Otherwise, it might be *aborted*, *timed-out* or in an *error* status depending on the type of failure.

The latency is measured as the time between the submission time of a computation job and the beginning of its execution, denoted as $(t_{tot} - t_{run})$ in figure 1. As said in the introduction, the mean latency, its variations and the execution time of most jobs from the biomed VO present comparable values. Thus, the latency and its variations have an important impact on the performance. A precise model of the latency is needed to handle failed jobs and refine submission strategies.

4. USING LATENCY MODELS FOR OPTIMIZATION

Models of the grid latency enable the optimization of job submission parameters such as jobs granularity or the timeout value needed to make the WMS robust against system faults and outliers.

Properly modeling a large scale infrastructure is a challenging problem given its heterogeneity and its dynamic behavior. In a previous work, we adopted a probabilistic approach [7] which proved to improve application performances while decreasing the load applied on the grid middleware by optimizing jobs granularity.

In [8], we have shown how the distribution of the grid latency impacts the choice of a timeout value for the jobs. We model the grid latency as a random variable R with probability density function (pdf) f_R and cumulative density function (cdf) F_R. The optimal timeout value is obtained by minimizing the expectation of the job execution time J which can be expressed as a function of R, the timeout t_∞ and the proportion of outliers ρ:

$$E_J(t_\infty) = \frac{1}{F_R(t_\infty)} \int_0^{t_\infty} u f_R(u) du + \frac{t_\infty}{(1-\rho)F_R(t_\infty)} - t_\infty \tag{1}$$

Figure 2 shows the cumulative density of the latency (F_R) computed from a set of 5800 measurements. This cdf and the computed ratio of outliers are introduced in equation 1 in order to determine the expected execution time with respect to the timeout value E_J (see figure 3). The minimum of the curve gives the optimal timeout value (here, $t_\infty = 528$ s leading to $E_J = 494s$).

5. EXPERIMENTAL DATA

To study the grid latency, measures were collected by submitting a very large number of probe jobs. These jobs, consisting in the execution of a negligible duration `/bin/hostname` command, are only impacted by the grid latency. In the remainder we make the hypothesis that the users job execution time is known and that therefore only the grid latency varies significantly between different runs of the same computation task. To avoid variations of the system load coming from our

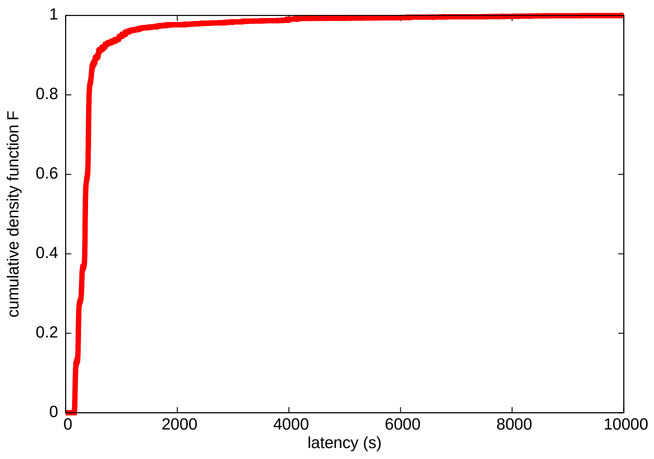

Figure 2: Cumulative density function (F_R) of the latency

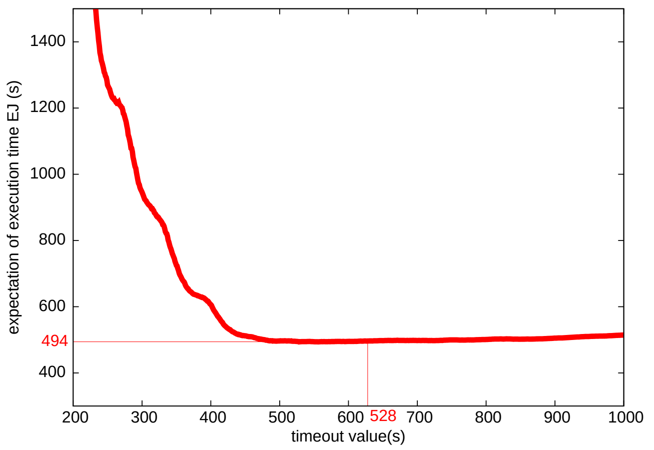

Figure 3: Expected execution time with respect to the timeout value.

monitoring, a constant number of probes was maintained inside the system at any time of the data collection: a new probe was submitted each time another one completed. For each probe job, we logged the job submission date, the job status and the total duration. The probe jobs were assigned a fixed 10000 seconds maximal duration beyond which they were considered as outliers and canceled. This value is far greater than the average latency observed. We also considered as outliers the failed jobs, assuming that they were resubmitted at t_∞, as the other outliers.

In a previous work [6], we collected a log gathering 5800 job traces in September 2006 (denoted further as 2006-IX). In this paper, we added 5093 job traces (with a ratio of outliers 21%) acquired from:

- September 5th 2007 until September 27th 2007 (weeks 2007-36 to 2007-39)
- December 12th 2007 until December 17th 2007 (week 2007-50)
- December 21th 2007 until January 22th 2008 (weeks 2007-51 to 2008-03)

The discontinuity of the periods in the new data set is due to unscheduled failures in the acquisition system and does not have any relations with authors choices.

6. ALONG THE WEEKS

Figure 4 shows the cumulative density function of the latency for the different weeks and for the whole period of 2007-2008. The curves covering the period 2007-2008 present a similar profile with steps coming from the waiting time of the jobs in the resource brokers (RB). One of the hypothesis is that they could be due to the internal scheduling algorithm of resource brokers. Another possible cause might be implementation flaws in the RB code. Those steps have also been observed in the vlemed VO of the EGEE grid.

An interesting way to compare those curves is to consider the differences between the optimal timeout values that they lead to (computed using equation 1). Figure 5 shows the expectation of the execution time for the different weeks. Despite the fact that the curves have different profiles, the optimal timeout values are visually in the same interval around 400s.

These values are detailed in table 1: the optimal value for 2006 is 528s while values for 2007-2008 range between 422s and 491s. The table also presents, for each period of time, the mean value and the standard deviation of the latency R. In most cases, a reduction of the mean latency corresponds to a decrease of the standard deviation. Finally the optimal expected execution time is shown. Assuming that the optimal timeout value has been computed in September 2006 (528s), we compute, in table 2, the resulting expectation of execution time and the relative difference with the optimal value computed week by week in order to measure the impact of parameters chosen earlier instead of the optimal one. The relative differences are up to 8%. It happens that this timeout value is greater than all optimal values for

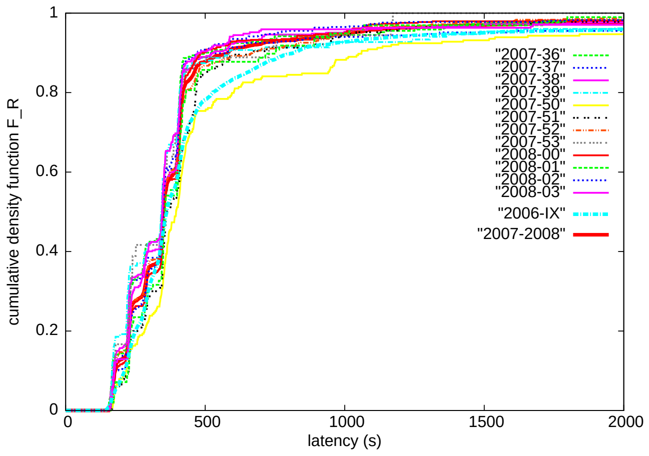

Figure 4: Cumulative density function of the latency for each week, computed on completed jobs.

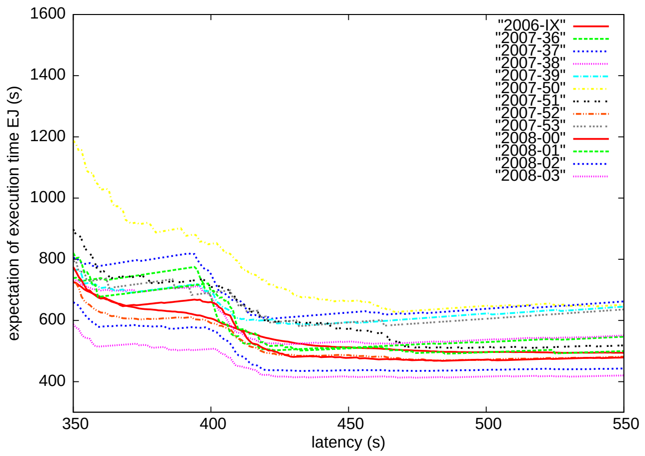

Figure 6: Close-up around the optimal values from figure 5

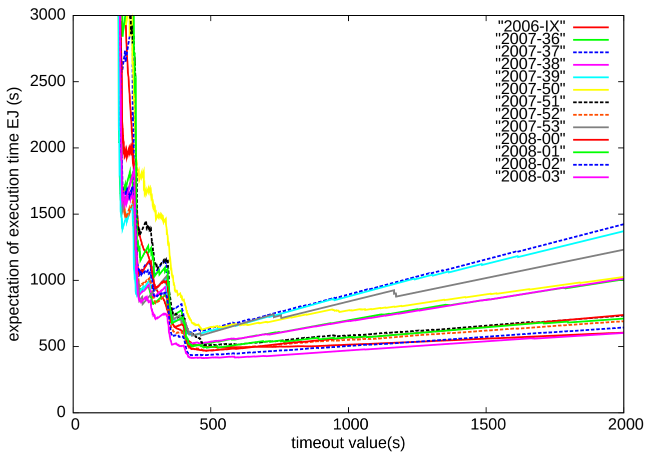

Figure 5: Expectation of job execution time with respect to the timeout value (t_∞). The minimum of each curve gives the best timeout value for the considered data set.

date	R	$\sigma(R)$	outliers	best t_∞	$E_J(t_\infty)$
2006-IX	570s	886s	5%	$\boxed{\text{528s}}$	494s
2007/08	469s	723s	17%	474s	500s
2007-36	446s	748s	24%	423s	502s
2007-37	506s	848s	33%	$\boxed{\text{422s}}$	606s
2007-38	447s	682s	24%	428s	522s
2007-39	489s	741s	32%	436s	585s
2007-50	660s	1046s	18%	467s	628s
2007-51	478s	510s	13%	$\boxed{\text{491s}}$	510s
2007-52	443s	582s	13%	482s	469s
2007-53	375s	238s	31%	432s	581s
2008-00	454s	699s	14%	484s	468s
2008-01	434s	317s	13%	485s	491s
2008-02	418s	547s	12%	433s	435s
2008-03	538s	1196s	10%	474s	413s

Table 1: Mean and standard variation of the latency, fraction of outliers, best timeout value and minimal expectation of execution time. These quantities are computed for the 2006 period, for the 2007-2008 period and for all weeks in the 2007-2008 period. The minimal optimal timeout value is 422 s while the maximal one is 491 s.

date	E_J(528s)	ΔE_J	date	E_J(528s)	ΔE_J
2007-36	528s	5.2 %	2007-52	477s	1.7 %
2007-37	648s	7.0 %	2007-53	623s	7.1 %
2007-38	544s	4.2 %	2008-00	475s	1.5 %
2007-39	631s	7.9 %	2008-01	493s	0.4 %
2007-50	652s	3.9 %	2008-02	441s	1.4 %
2007-51	514s	0.9 %	2008-03	418s	1.2 %

Table 2: In this experiment, the timeout value from the period of September 2006 has been used (528s). For each week of the 2007-2008 period, we present the expectation of the execution time and the relative difference with the optimal one.

date	E_J(422s)	ΔE_J%	E_J(491s)	ΔE_J%
2007-36	505.5	0.7%	527.1	5.0%
2007-37	605.9	0%	632.2	4.3%
2007-38	524.8	0.5%	530.5	1.6%
2007-39	602.9	3.1%	616.8	5.5%
2007-50	718.7	14.5%	642.3	2.3%
2007-51	594.9	16.7%	509.6	0%
2007-52	491.2	4.8%	470.9	0.4%
2007-53	593.7	2.1%	600.0	3.2%
2008-00	501.9	7.2%	470.0	0.4%
2008-01	516.7	5.2%	493.1	0.4%
2008-02	437.0	0.6%	437.2	0.6%
2008-03	419.1	1.5%	414.8	0.5%

Table 3: In this experiment, we focus on data from the period 2007-2008. As determined in table 1, the minimum timeout value is 422s and the maximum is 491s. For these extreme values, the new expectation of execution time and the relative difference with the optimal value are presented.

the period 2007-2008. The highest differences are obtained when the ascending slope of figure 5 are the highest, which is directly related to the fraction of outliers.

Furthermore, we took the minimal and the maximal of timeout values among the different weeks : 422 s and 491 s. We present the expected execution time for each of these values and the relative differences in table 3. In the case of the maximal timeout value, relative errors are below 6% while in the case of the minimal timeout value, relative errors are up to 17%. This is clearly explained by the shape of the curves on figure 5: the slope of the decreasing part is higher than the slope of the increasing part of each curve. Thus, an overestimation of the timeout value is better than an underestimation, if this overestimation is not too high, which must be quantified. As a conclusion of this part of the study, actualization of the timeout value may improve the total execution time, up to 17%.

7. LATENCY CONTEXT PARAMETERS

In order to refine our latency model, we consider the parameters of the execution context that could explain its high variability. Two different jobs submitted on the EGEE grid may differ by the path they follow from their submission site (UI) to the execution site (WN). Different hardware characteristics and software configuration/version may influence the performance. The load of the infrastructure is also an important factor and may depend on the behavior of the different users and the nature of the experiments they are conducting. Not only users are to be considered but also administration operations that influence the grid status. Similarly, time context (working or not-working periods) is to be considered. Failures may have different explanations such as hardware breakdowns, load and other external factors such as extremely hot weather conditions leading to air conditioning breaking down.

Going into the details of all the local parameters of the problem is intractable due to the size and the availability of such information. The level of detail that can be exploited depends both on the availability of the contextual information and the needs of the model.

We thus restrict our study to context parameters that could realistically be used for estimating job latencies and optimizing job resubmission strategies.

In this paper, we consider high level information such as Resource Broker, Computing Element and queues used. We also study the time context using the day of the week. Similarly, in a future work, cities temperature may be a sufficient and accessible information to further optimize it.

7.1 Resource Broker parameter (RB)

The probe measures acquired during three weeks in September 2006 (log 2006-IX) were submitted to 3 different Resource Brokers (RBs):

- a French one (`grid09.lal.in2p3.fr`),
- a Spanish one (`egeerb.ifca.org.es`) and
- a Russian one (`lcg16.sinp.msu.ru`).

Figure 7 displays the cumulative density function of the latency of the probe jobs sent to each of the RBs as well as the one of the submission time considering all RBs.

The 3 RBs exhibit quite different behaviors that need to be quantified. Table 4 displays:

- the optimal estimated timeout value;
- the difference between this value and the global reference value obtained using all measurements without distinction;
- the minimal expected execution time;
- the expected execution time if the timeout is set to the global reference value;
- and the difference with the optimum.

The optimal timeout values significantly differ and the most distinct is the one associated to the French RB (variation of 31%). However, the expected execution time varies by

	all RBs	RB fr	RB es	RB ru
optimal t_∞ Δt_∞	$t_{\infty\text{ref}} = 556$ s 0%	729 s 31%	546 s 2%	506 s 9%
best E_J	479.125 s	483.7 s	445.2 s	476.2 s
$E_J(t_{\infty\text{ref}})$ ΔE_J	479.125 s 0%	488.8 s 1%	445.9 s 0.2%	477.9 s 0.4%

Table 4: Study of the influence of the Resource Brokers.

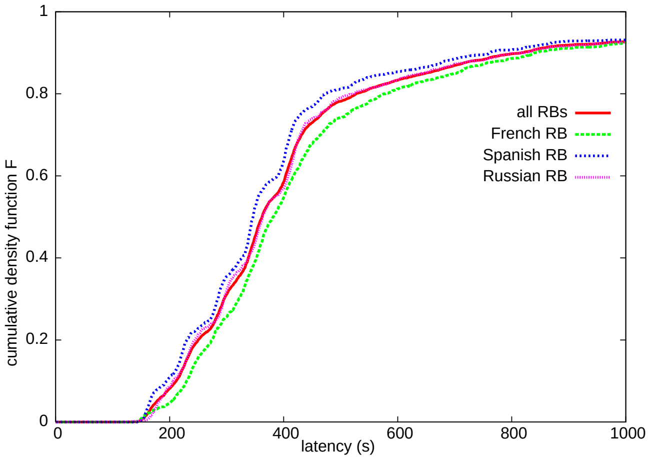

Figure 7: Cumulative density function for the different Resource Brokers: France (fr), Spain (es) and Russia (ru).

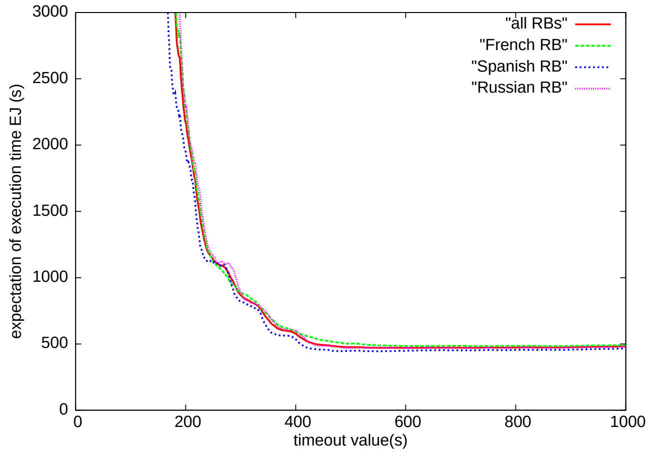

Figure 8: Expectation of execution time for the different Resource Brokers.

a much smaller amount (1% maximum). This is related to the fact that in this case (relatively low outliers ratio and rather homogeneous infrastructure), slightly overestimating the timeout has little impact on the execution time. It should be noted that an underestimation is impacting the execution time much more though as can be seen on figure 8.

The period of this experiment presented a very low level of outliers, in the order of 3%. This implies low increasing slope of expectation of execution time after the minimum value (see figure 8). To simulate a more variable infrastructure, we applied the model considering a variable level of outliers between the different RBs (ρ = 20%, 3% and 0% respectively). These errors are realistic as error conditions regularly lead to similar values (ρ = 21% was observed in 2007/08). The results are summarized in the following table:

	all RBs	RB fr	RB es	RB ru
ρ optimal t_∞	7.7% $t_{\infty\text{ref}} = 868$ s	20% 551 s	3% 546 s	0% 865 s
best E_J $E_J(t_{\infty\text{ref}})$ ΔE_J	452.3 s 452.3 s 0%	639.8 s 691.7 s 8%	445.2 s 456.2 s 2.5%	451.7 s 451.7 s 0%

In this case, the model consistently reports growing execution time disruptions with the increase of the number of outliers. The resubmission strategy still rather efficiently cope with the errors as the execution time variation does not exceed 8%. Taking into account the submission RB can help in adapting the optimal timeout choice. The more variable the infrastructure, the more valuable the optimization.

7.2 Computing Element (CE)

In a computing center, the batch submission system is usually configured with several queues. The influence of the Computing Element (CE) and the associated queues, later abbreviated as CE-queue, is considered in this section. The same methodology than with RBs in section 7.1 could be envisaged but a significant difference is that the number of CE-queues is much larger than the number of RBs in the same set of data: we had 92 CEs and queues and only 3 RBs. It might thus be relevant to group similar CE-queues to obtain fewer classes. As can be seen in figure 9 many of the 92 CE-queues have similar cdfs while others are more singular. The idea we promote here is to group CEs and queues that have similar properties into different classes. To ensure statistical significance, CE-queues with less than 30 probe measures were removed from the study. Thus, 60 CE-queues out of the 92 were remaining.

7.2.1 Classification of the CE and queues

Figure 9 suggests that 3 classes can be identified among the CEs. A k-means classification was thus done on the cumulative density functions of the CEs and the obtained classes are identified with distinct colors on the figure. Centroids of the classes are plotted in black.

The first class of CEs, pictured in blue, has the highest performance in average. The median of its centroid is 237 seconds. It is composed of 15 CEs. The second class of CEs, pictured in green, is composed of 35 CEs. The median of

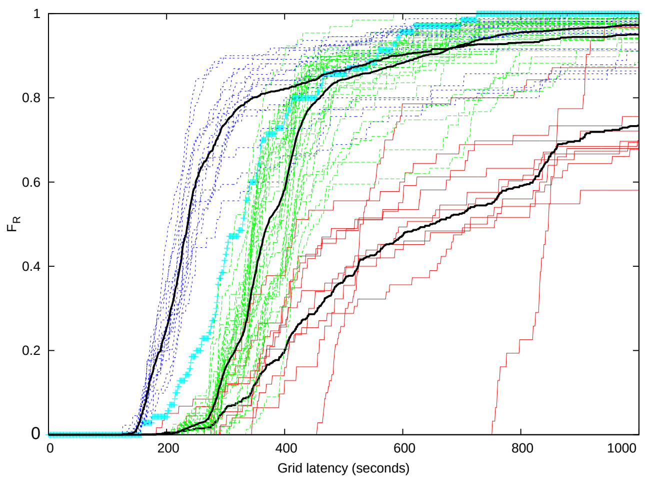

Figure 9: Classification in 3 classes of the cumulative density functions of the grid latencies by CE. Centroids of the k-means classes are plotted in black.

CE group	Median (s)	Expect. (s)	Stdev (s)
not lcgpbs (blue)	237	436	880
lcgpbs (green)	373	461	493
other (red)	652	1132	1396
Whole data	363	559	850

Table 5: First moments and median of the grid latency w.r.t the execution CE class

its centroid is 373 seconds, which corresponds to a 1.6 ratio with respect to the fastest class. Finally, the slowest class, pictured in red, is composed of 10 CEs and the median of its centroid is 652 seconds. Table 5 compares the median, expectation and standard-deviation of the grid latency for each CE class. It reveals that even if the first (blue) class of CEs has the highest performance in average, it is also more variable than the second (green) class. The third (red) class is the most variable. The impact of variability on the performances of an application depends on the number of submitted jobs and on the performance metric. In some cases (high number of jobs), it would be better to submit jobs on a less variable CE class, even if it has the lowest performance in average.

A noticeable feature of the green class is that almost all of its CEs contain the `lcgpbs` string in their names. In this class, the only CE whose name does not contain this string is plotted in cyan on figure 9 and is close to the border of this class. In the blue class, no CE contains this string in its name and in the slowest class, 7 CEs have this string in their name. This shows that the `lcgpbs` string name is informative in itself although the reasons are not necessarily known (it may correspond to a specific middleware version deployed on some of the CEs in this heterogeneous infrastructure).

7.2.2 Testing different number of classes for CE-queues

Different aggregations of CE-queues were tested based on their cdf using the k-means classification algorithm with $k = 2$ to 10 classes. For each CE-queue entity, the cdf has been computed. From this cdf the optimal timeout value is computed , by minimizing equation 1. Figure 10 shows the repartition of the timeout values in the classes. The width of each box is proportional to the number of CE-queues in the class.

In order to measure if the classes are statistically discriminant, we have tested the hypothesis H_0 "*all sets have equal mean and equal variance*" using ANOVA (ANalysis Of VAriance). The results are reported in the following table where:

Df: degree of freedom

F: ratio statistic of the between groups variance to the within groups variance

p value: the significance level of H_0

symbol *:** means rejection of hypothesis H_0 with high confidence (level 1% of p < 0.01)

nb. of classes	Df	F	p value	H_0 rej.
2	1	13.9	4.10^{-04}	***
3	2	10.0	1.10^{-04}	***
4	3	14.1	2.10^{-07}	***
5	4	10.3	1.10^{-06}	***
6	5	8.4	2.10^{-06}	***
7	6	10.9	1.10^{-08}	***
8	7	9.6	2.10^{-08}	***
9	8	9.5	8.10^{-09}	***
10	9	8.3	3.10^{-08}	***

The result of the ANOVA test shows that the H_0 hypothesis is strongly rejected in all cases at a level less than 1% (between 8.10^{-7}% and 0.04%). The best result is obtained for 9 classes but the gain is not so high. Note that the ANOVA test only shows that the hypothesis H_0 is rejected: this does not necessary imply that all classes differ from each other.

In the case of 2 classes, these classes are statistically discriminant. But for all other cases, further tests must be done in order to determine how many classes are independent.

7.2.3 Refining the ANOVA analysis

Let us look, for example, at the case of classification into 3 different classes (classes 0, 1 and 2). Using ANOVA, if we test classes 1 and 2, we observe that they do not differ significantly: $F = 0.2334$ ($p < 0.6338$). Building a new class, class 1+2, from classes 1 and 2, we now test class 0 against class 1+2 and obtain that they differ significantly: $F = 19.651$ ($p < 3.003e{-}05$). We observe that grouping two classes after the classification $k = 3$ gives a similar although slightly better result than the classification $k = 2$.

The optimal timeout for the 2 populations (classes 0 and 3) are $t_{\infty 0} = 779s$ and $t_{\infty 3} = 881s$ respectively (see figure 12).

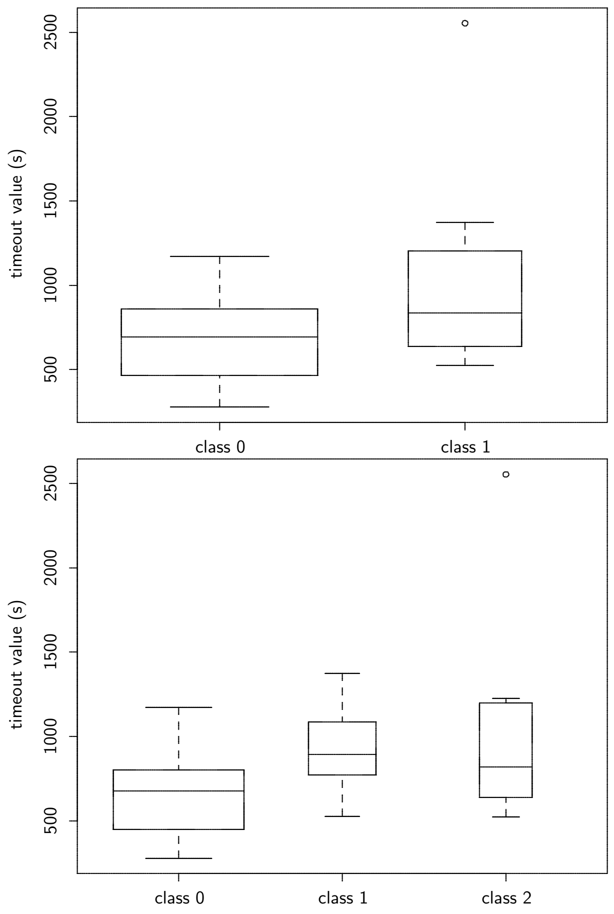

Figure 10: Timeout values repartition after k-mean classification into 2 classes (on top) and 3 classes (on bottom) of CEs and queues.

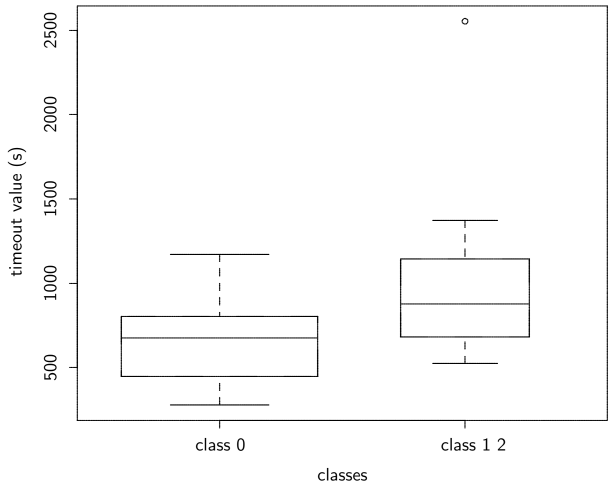

Figure 11: k-means classification into 3 classes after grouping classes 1 and 2.

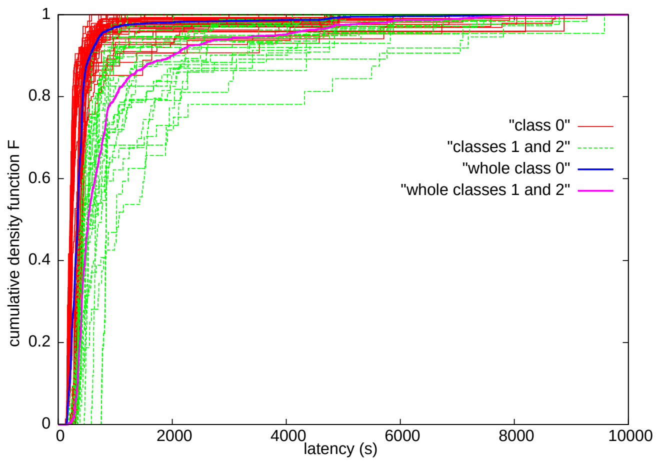

Figure 12: Cumulative density functions of latency with respect to time (in seconds). This figure is obtained from the k-means classification into 3 classes. We grouped the last 2 classes into a single one so that we have 2 classes: the initial class 0 (in red) and new class 3 (in green) resulting of the merging of classes 1 and 2. Each curve corresponds to the cdf of one CE-queue. The curves in blue and magenta correspond to the cdf of the centroid of class 0 (in blue) and class 3 (in magenta).

Figure 13 shows the errors computed between the best E_J and $E_J(t_\infty)$, where t_∞ if computed from the whole class.

7.2.4 Discussion

The order of magnitude of the grid latency thus appears to be correlated to the execution CE. It is relevant because the CE is directly related to the job queuing time as a CE exactly corresponds to a batch queue. Variations of middleware, system versions and of the availability of the site to VOs may explain the differences observed among the 3 different classes while variations inside a given class may be coming from the load imposed by the users and the performance of CEs host hardware. However, in general, the execution CE is only known after the job submission, during the scheduling procedure. Thus, this information could only be exploited for parameters that can be updated once the job has been submitted, as for instance the timeout value or the application completion prediction.

7.3 Day of the week

Considering that the load of the grid may depend on the human activity (users are often starting to submit jobs during their work time), we have, as a first step, considered the influence of the day of the week on the latency. We are expecting to have different behaviors during the week than during the week-end, considering that in most western countries, Saturdays and Sundays are non working days. The considered data is presented in section 5 and concerned several weeks.

Figure 14 shows the different expectations of latency for each day of the week. The different days of week present different

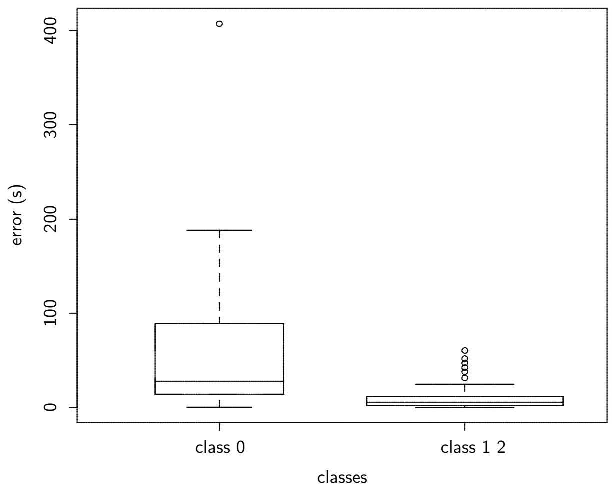

Figure 13: Errors measured between best E_J and $E_J(t_\infty)$, where t_∞ if computed from the whole class.

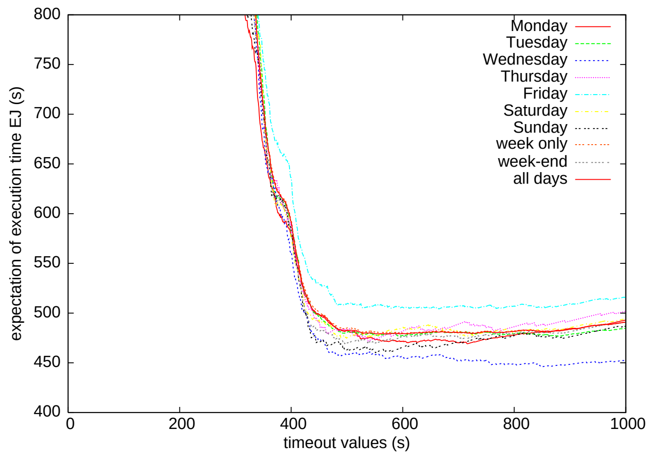

Figure 14: Expectation of job execution time for each day of the week.

behaviors. However, there is no clear distinction between week and week-end, even if we observe that the two extrema curve correspond to Saturday and Sunday. An explanation is that we can submit an experiment on Friday that will automatically submit the jobs for several days and that there is probably a stress effect on Friday's submissions.

To quantify the influence of the day of the week, we compute, for each week and each day of the week, the optimal timeout value, according to equation 1. These values are plotted on figure 15 with respect to the day of the week. As confirmed by ANOVA analysis, there is no significant difference between the days of the week.

We also present on figure 16 the evolution of the optimal timeout value with respect to the week of the experiment, for each day of the week. There is no evidence of absolute pertinence of the day of the week in this figure. Even the

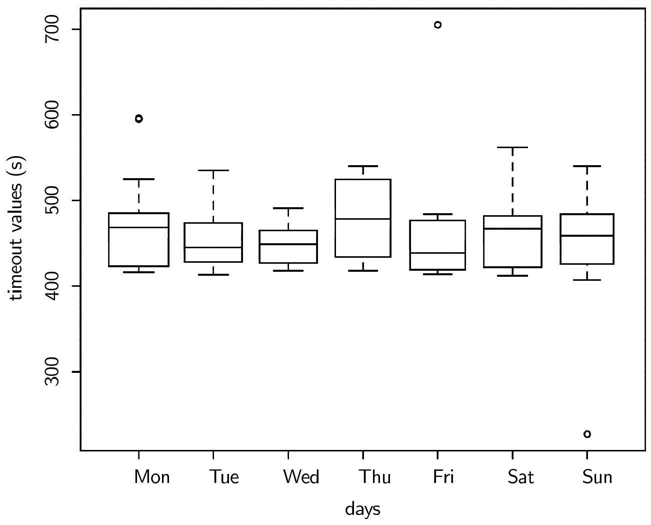

Figure 15: For each day of the week and for each week, the best timeout value is computed. We have plotted boxes for each day of the week. According to ANOVA, there is no significant difference between the days of the week.

period between Christmas and New Year's eve is not really separated from the others. However, in figure 17, we observe that, in most weeks, there is a decrease of best timeout value between Tuesday or Wednesday and Thursday followed by an increase until Friday or Saturday. This profile information needs further investigation to be exploited.

This time context study may be completed by the analysis of the hours in the day.

8. CONCLUSION

We have shown that some context parameters related to the EGEE production grid such as Resource Brokers and CE-queues have an influence on the expected job execution time. Moreover, we have shown that we can group CE-queues into classes that are statistically different, thus reducing the number of data to be analyzed. The methodology used could be applied to other grids by replacing CEs and RBs by the equivalent workload management services. In the DIET middleware [3] for instance, it could correspond to Master Agents (MA) and Local Agents (LA).

The experiment on the influence of the day of the week shows that it has a hardly relevant impact. The hour of the day could be considered alternatively.

The study along several weeks shows that variations of the load conditions over long periods of time make it necessary to update the model parameters along time. Future work will focus on strategies to perform this update.

Moreover, parameters of the execution context such as the Resource Broker, class of Computing Center and queues and time need to be studied over long period of time to determine (i) if there are global trends observable during several weeks and (ii) how frequently the experimental models need to be

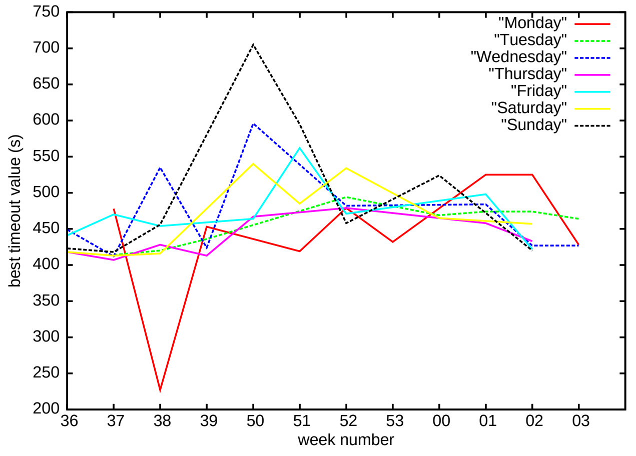

Figure 16: Each curve corresponds to a day of the week. We plotted optimal timeout values with respect to the weeks of the experiment.

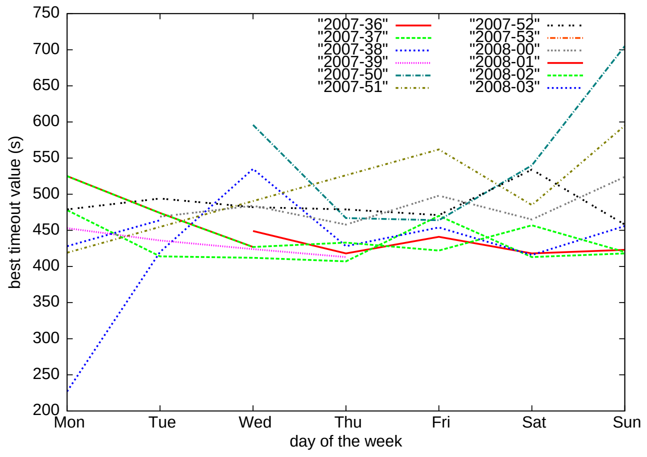

Figure 17: Each curve corresponds to a week of the experiment. The optimal timeout value is plotted with respect to the day of the week.

updated.

9. ACKNOWLEDGMENTS

This work is partially funded by the French national research agency (ANR), NeuroLog project[2] under contract number ANR-06-TLOG-024. This project is in the scope of scientific topics of the STIC-ASIA OnCoMedia project[3]. We are grateful to the EGEE European project for providing the grid infrastructure and user assistance.

10. REFERENCES

[1] G. Aloiso, S. Benkner, H. Bilofsky, I. Blanquer Espert, V. Breton, M. Cannataro, I. Chouvarda, B. Claerhout, K. Dean, S. Fiore, K. Hassan, G. Heeren, V. Hernández García, J. Herveg, M. Hofmann, J. Herveg, V. Koutkias, C. Jones, G. Lonsdale, V. López, N. Maglaveras, L. Maigne, A. Malousi, F. Martin-Sanchez, R. McLatchey, E. Medico, S. Miguet, M. Mirto, J. Montagnat, G. De Moor, K. Nozaki, I. Oliviera, X. Pennec, J. Sanchez, T. Solomonides, M. Taillet, P. Veltri, C. De Wagster, and R. Ziegler. HealthGrid White Paper, Sept. 2005.

[2] H. Benoit-Cattin, G. Collewet, B. Belaroussi, H. Saint-Jalmes, and C. Odet. The SIMRI project : a versatile and interactive MRI simulator. *Journal of Magnetic Resonance Imaging (JMRI)*, (173):97–115, 2005.

[3] E. Caron and F. Desprez. DIET: A Scalable Toolbox to Build Network Enabled Servers on the Grid. *International Journal of High Performance Computing Applications*, 20(3):335–352, 2006.

[4] D. Feitelson. *Workload modeling for performance evaluation*, pages 114–141. Springer-Verlag - LNCS vol 2459, Sept. 2002.

[5] J. Geddes, C. Mackay, S. Lloyd, A. Simpson, D. Power, D. Russell, M. Katzarova, M. Rossor, N. Fox, J. Fletcher, D. Hill, K. McLeish, J. Hajnal, S. Lawrie, D. Job, A. McIntosh, J. Wardlaw, P. Sandercock, J. Palmer, D. Perry, R. Procter, J. Ure, P. Bath, and G. Watson. The Challenges of Developing a Collaborative Data and Compute Grid for Neurosciences. In *19th IEEE International Symposium on Computer-Based Medical Systems, 2006. CBMS 2006*, pages 81–86, Salt Lake City, Utah, June 2006. IEEE Computer Society.

[6] T. Glatard, D. Lingrand, J. Montagnat, and M. Riveill. Impact of the execution context on Grid job performances. In *International Workshop on Context-Awareness and Mobility in Grid Computing (WCAMG'07)*, pages 713–718, Rio de Janeiro, May 2007. IEEE.

[7] T. Glatard, J. Montagnat, and X. Pennec. Probabilistic and dynamic optimization of job partitioning on a grid infrastructure. In *14th euromicro conference on Parallel, Distributed and network-based Processing (PDP06)*, pages 231–238, Montbéliard-Sochaux, France, Feb. 2006.

[8] T. Glatard, J. Montagnat, and X. Pennec. Optimizing jobs timeouts on clusters and production grids. In

[2]Neurolog: `http://neurolog.polytech.unice.fr`
[3]OnCoMedia: `http://www.onco-media.com`

International Symposium on Cluster Computing and the Grid (CCGrid'07), pages 100–107, Rio de Janeiro, May 2007. IEEE.

[9] H. Li, D. Groep, and L. Walters. Workload Characteristics of a Multi-cluster Supercomputer. In *Job Scheduling Strategies for Parallel Processing*, pages 176–193. Springer Verlag, 2004.

[10] L. Libman and A. Orda. Optimal Retrial and Timeout Strategies for Accessing Network Resources. *IEEE/ACM Transactions on Networking (TN)*, 10(4):551–564, Aug. 2002.

[11] D. Lingrand, J. Montagnat, and T. Glatard. Estimating the execution context for refining submission strategies on production grids. In *Assessing Models of Networks and Distributed Computing Platforms (ASSESS) (CCgrid'08)*, Lyon, May 2008. IEEE.

[12] J. Montagnat, F. Bellet, H. Benoit-Cattin, V. Breton, L. Brunie, H. Duque, Y. Legré, I. Magnin, L. Maigne, S. Miguet, J.-M. Pierson, L. Seitz, and T. Tweed. Medical images simulation, storage, and processing on the european datagrid testbed. *Journal of Grid Computing (JGC)*, 2(4):387–400, Dec. 2004.

[13] J. Montagnat, T. Glatard, and D. Lingrand. Texture-based Medical Image Indexing and Retrieval on Grids. *Medical Imaging Technology (MIT)*, 25(5), Nov. 2007.

[14] J. Nichols, H. Demirkan, and M. Goul. Autonomic Workflow Execution in the Grid. *IEEE Transactions on Systems, Man, and Cybernetics*, 36(3), May 2006.

[15] D. Nurmi, A. Mandal, J. Brevik, C. Koelbel, R. Wolski, and K. Kennedy. Evaluation of a Workflow Scheduler Using Integrated Performance Modelling and Batch Queue Wait Time Prediction. In *Conference on High Performance Networking and Computing*, Tampa, Florida, Nov. 2006. ACM Press.

[16] M. Oikonomakos, K. Christodoulopoulos, and E. Varvarigos. Profiling Computation Jobs in Grid Systems. In *IEEE International Symposium on Cluster Computing and the Grid (CCGrid07)*, pages 197–204, Rio de Janeiro, Brasil, May 2007.

[17] S. Ravelomanana, S. C. S. Bianchi, C. Joumaa, and M. Sibilla. A Contextual GRID Monitoring by a Model Driven Approach. In *Proceedings of the Advanced International Conference on Telecommuni cations and International Conference on Internet and Web Applications and Servic (AICT/ICIW)*, pages 37–43, Gosier, Guadeloupe, Feb. 2006. IEEE.

[18] A. van Moorsel and K. Wolter. Analysis of Restart Mechanisms in Software Systems. *IEEE Transactions on Software Engineering (TSE)*, 32(8):547–558, Aug. 2006.