Database characterisation of HEP applications

2012 J. Phys.: Conf. Ser. 396 052060

(http://iopscience.iop.org/1742-6596/396/5/052060)

Journal of Physics: Conference Series **396** (2012) 052060 doi:10.1088/1742-6596/396/5/052060

Database characterisation of HEP applications

Mariusz Piorkowski, Eric Grancher, Anton Topurov
CERN IT-DB, Geneva, Switzerland

E-mail: mariusz.piorkowski@cern.ch,eric.grancher@cern.ch,anton.topurov@cern.ch

Abstract. Oracle-based database applications underpin many key aspects of operations for both the LHC accelerator and the LHC experiments. In addition to the overall performance, the predictability of the response is a key requirement to ensure smooth operations and delivering predictability requires understanding the applications from the ground up. Fortunately, database management systems provide several tools to check, measure, analyse and gather useful information. We present our experiences characterising the performance of several typical HEP database applications performance characterisations that were used to deliver improved predictability and scalability as well as for optimising the hardware platform choice as we migrated to new hardware and Oracle 11g.

1. Introduction

The LHC experiments and the accelerator are using databases for many of their operations, from control to analysis. Every year most of the applications grow in activity, complexity and data volume. These are the main reasons why the workload of the database servers is increasing. On the other side, expectations from applications users, especially regarding database performance and query response time, are getting higher or at best stay on the same level. Those factors affect future planning decisions, particularly upgrades to new database systems. In order to cope with the growing demand, the standard approach is to get more powerful hardware. However, given the limited operational budget, one cannot buy top-notch hardware in provision for having enough capacity for all possible increases of workloads. To estimate properly the hardware needs and architect the best configuration as well as to iterate on performance tuning exercises, a trustable workload replay methodology is needed.

2. HEP database applications

Almost all database applications at some point during their lifetime will go through a phase requiring special attention to performance issues. It could be performance problem like increase of response time, never finishing SQL statements or even complete application hang causing database slowdown. The root cause can be different in every particular case; however most cases are closely connected to resource starvation or not optimally written code, namely SQL statements. According to database experts, up to 80% of application or database performance problems can be resolved by optimizing

SQL statements [1]. In order to forego a SQL statement or database tuning exercise, one needs to quantify and save performance indicators.

In case of physics applications, which are tightly linked with high activity and the criticality of the databases deployed, the performance problems are often challenging in terms of activity and data size, even if the bulk of the data is not stored in database. In the database environment for LHC experiments, most applications are running perfectly well, however some of them require important computing resources and special attention is needed to avoid severe performance with even minimal application change. In this paper, the focus is set on a few applications which are especially interesting from the database point of view. Of course there are many more applications but the ones chosen for this study have different design purpose and goals to achieve, representing different types of applications among the physics applications used in the Worldwide LHC Computing Grid project.

The Dashboard [2] is a project which provides a single entry point to the monitoring data collected from the LHC experiments. It collects information from multiple sources, covering different activities like job processing, data management, transfer tests, monitoring of the reconstruction at TIER-0 and monitoring of site efficiency.

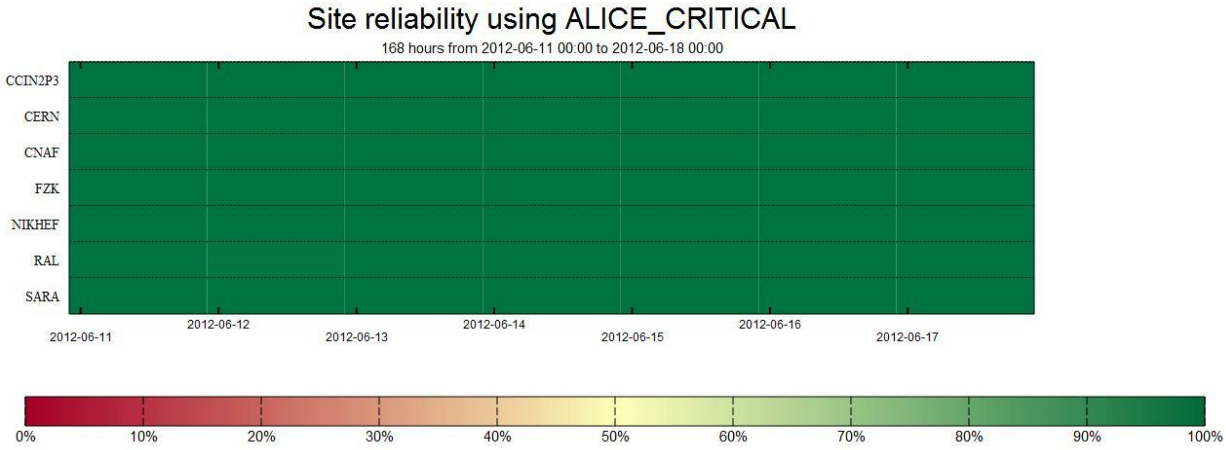

Figure 1. Dashboard – sum visualization Alice.

The CASTOR Stager [3] application manages files on pools of disks and organises transfers for those files to and from the tape system. The Stager has the primary role of a disk pool manager whose function is to allocate space on disk to store a file, to maintain a catalogue of all the files in its disk pools and to clear out old or least recently used files in these pools when more free space is required.

The PVSS [4] application is a composite SCADA [5] system with a highly distributed architecture. It is used to connect to hardware or software devices, acquire the data they produce and use it for their supervision i.e. to monitor their behaviour and to initialize, configure and operate them.

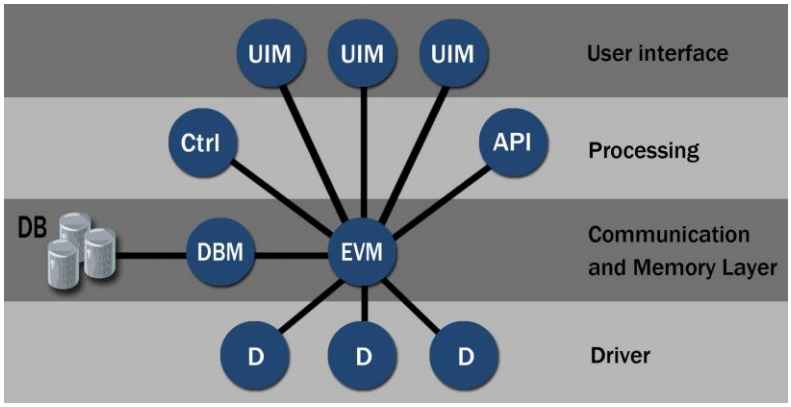

Figure 2. The PVSS architecture.

In case of such a complex systems, it is not simple to measure the total application workload at the database level. It is essential to simplify the problem by focusing on measurements on major application components instead of doing it for the whole application.

Fortunately not all applications at CERN have such complex architecture and demanding requirements, this is why some do not require such a complex relational database management system as Oracle. MySQL has been chosen by a number of projects because of its simplicity and the fact that it is open source and widely used. This has lead the IT database group to also offer the MySQL solution through the database OnDemand automated service. Through the database OnDemand interface, MySQL users can check and analyze applications performance using the provided monitoring views.

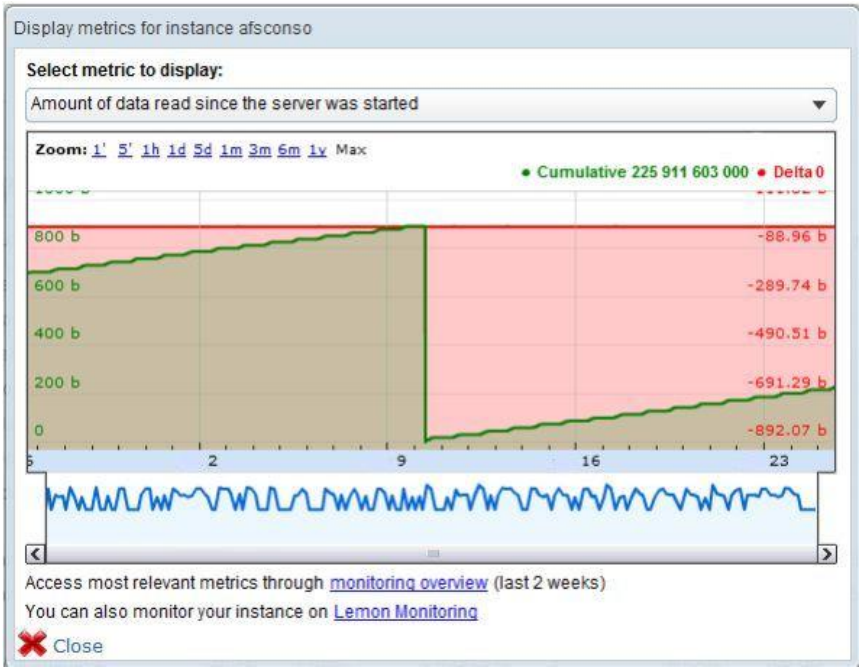

Figure 3. AFS console application metric.

One of those applications running on the MySQL database is the AFS console [6] used for monitoring the CERN AFS system. AFS (Andrew File System) is a service which provides networked file storage

for CERN users, in particular home directories, work spaces and project spaces. The AFS Service is based on OpenAFS, an open-source distributed filesystem which provides a client-server architecture for location-independent, scalable, and secure file sharing.

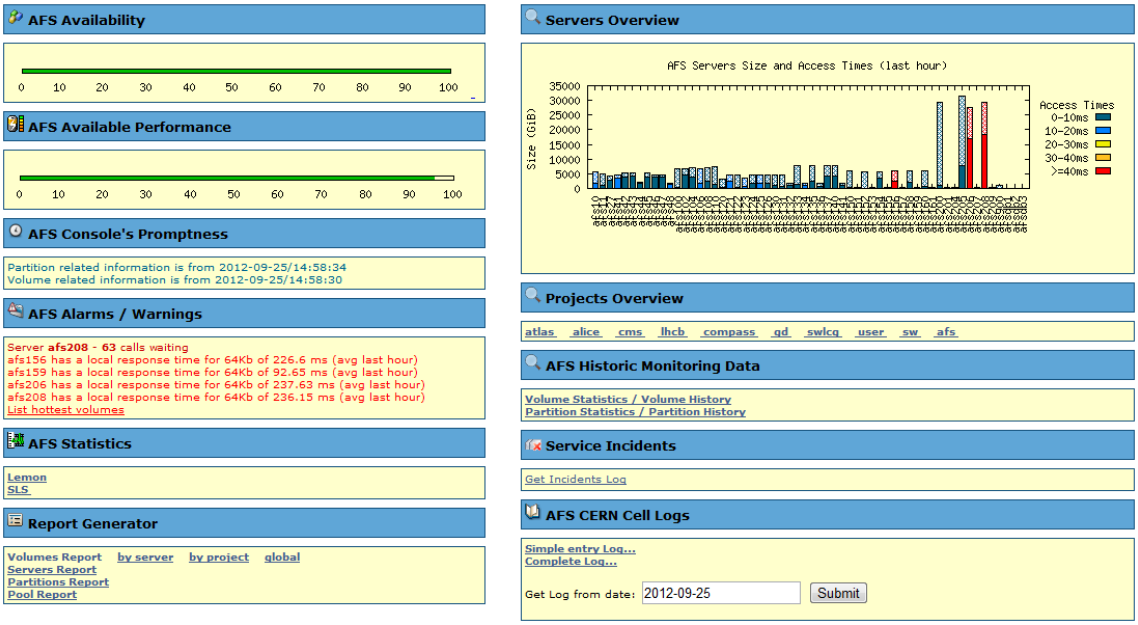

Figure 4. AFS console application metric.

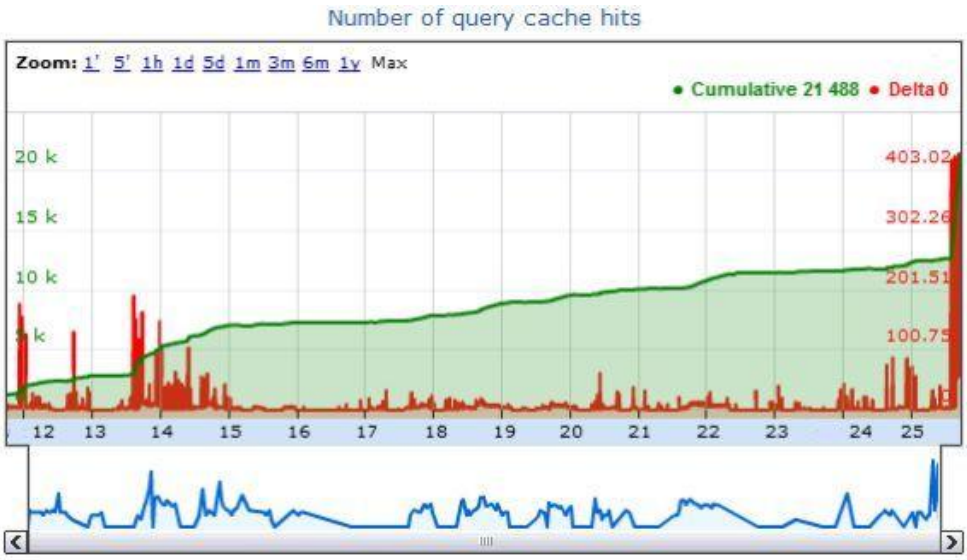

Figure 5. AFS console - number of query cache hits.

Another application which is also relying on the MySQL database is the Drupal [7] management platform. At CERN lots of projects have recently started to use Drupal for websites and applications. This platform provides a user friendly interface for users and also what is even more important, any changes or updates of the content are easy to implement. The "amount of data written view" is available on the MySQL service portal as an historical performance view. For the Drupal MySQL database, the graph (see below) clearly shows the regular steady usage in writing of the application.

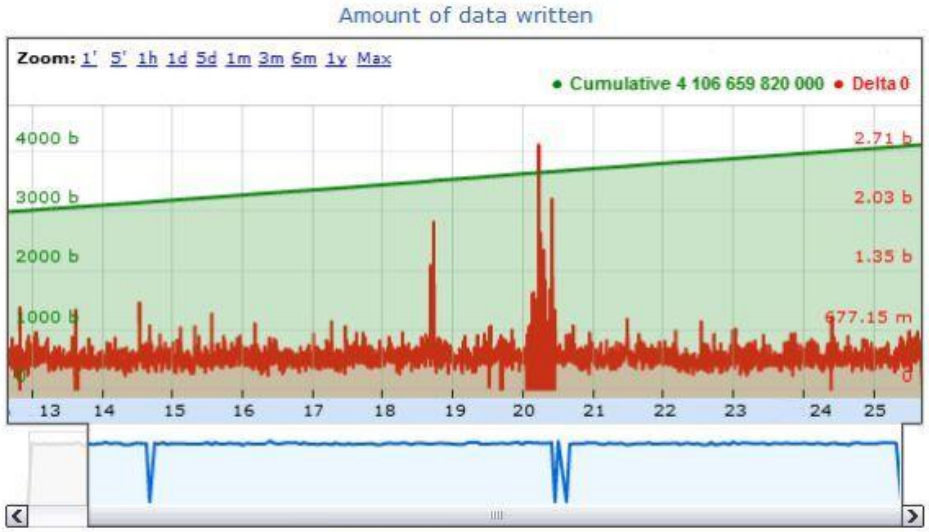

Figure 6. Amount of data written - MySQL Drupal database.

The most intensive physics application at the database level running on the MySQL database is currently HammerCloud [8] for the ATLAS experiment. This application is a Distributed Analysis testing system that can test the site(s) and report the results obtained. The following metric shows how intensive the user CPU usage is.

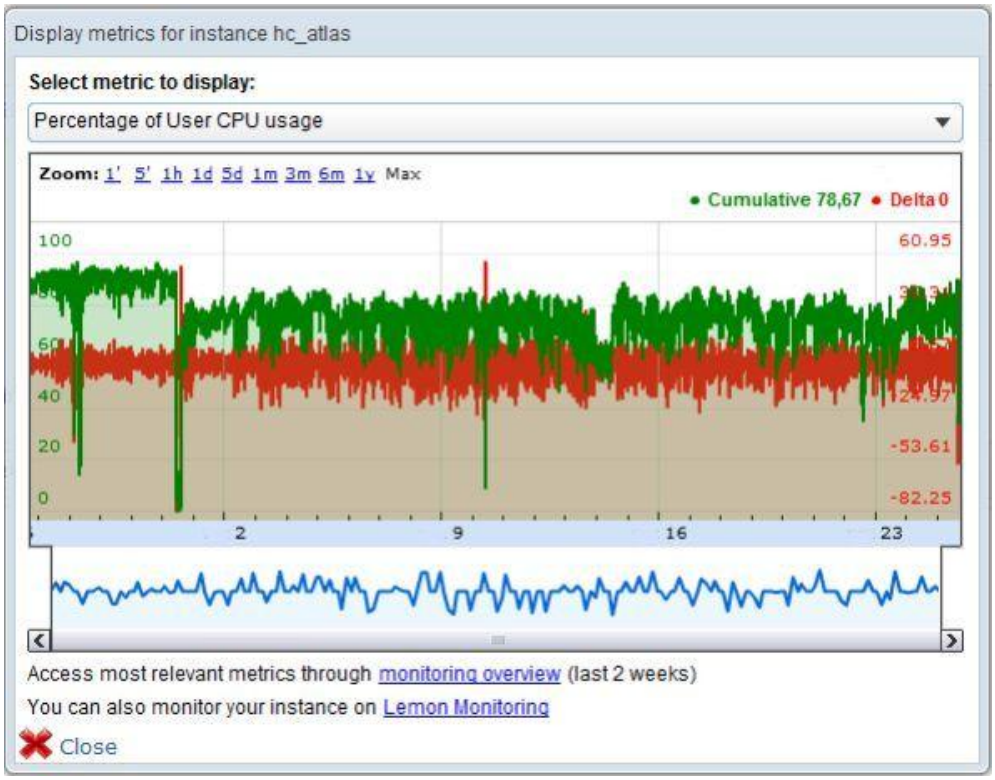

Figure 7. Percentage of user CPU usage.

All of the above characteristics are really valuable not only for application developers and service providers but also for the users who can better understand key characteristics of the application Thanks to them it much easier to check performance problems, follow users behaviour and understand the trend of usage for specific application components. However, it should be noted that tuning of applications, improving the scalability or performance is a task that can require a very detailed understanding of both the application and the database management system.

3. Methodology and tools

The database management system provides several tools which can be successfully used to measure application attributes. Since the tools alone are useless without a proper testing methodology, the following section describes how we should achieve our goals before we start any performance analysis.

The methodology used to gather HEP applications characteristics at the database level is simple and not related only to database management tools. It can be used in different scenarios, with various solutions. The diagram below (Figure 8) represents a circle scheme, where all steps can be repeated until the predefined performance goals are reached.

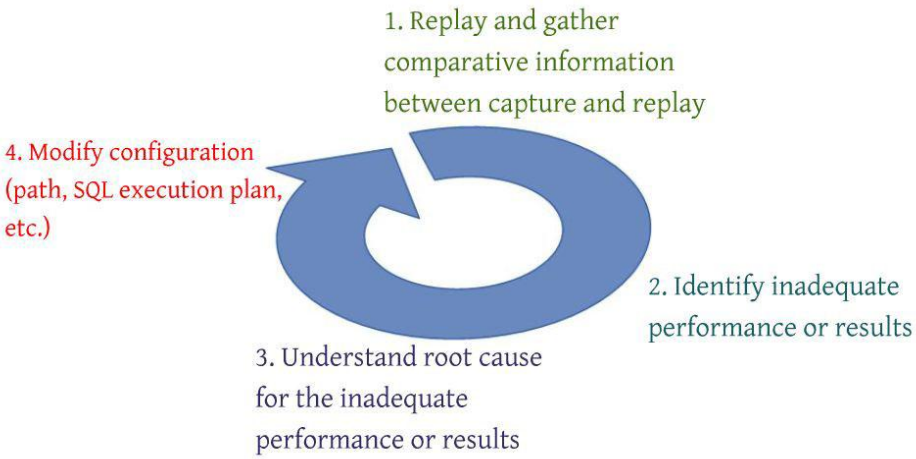

Figure 8. Methodology scheme.

In the beginning, the first step is to gather comparative information about the system or application in question; this can be done with Oracle Real Application Testing (RAT) tool [9]. Having the ability to capture systems workload is highly critical as it enables later changes to the system and compares the performance with base performance figures. This tool allows capturing real life database workloads, process and replays it on the modified or new database system. At CERN, RAT has proven to be particularly useful during database migration process to new 11.2 database version. (Figure 9)

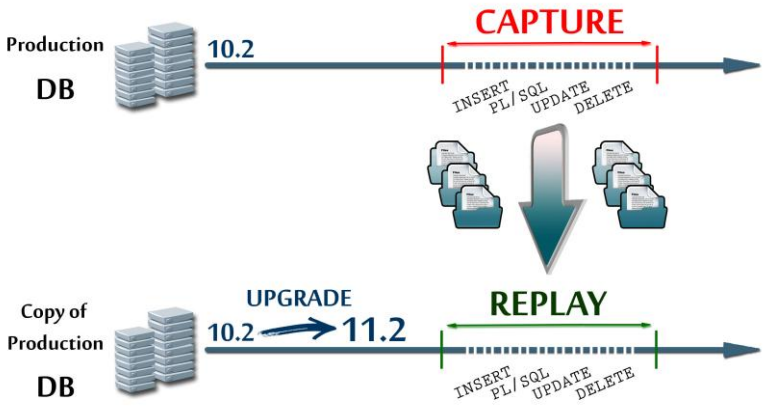

Figure 9. Migration process with Real Application Testing feature.

The most efficient way is to capture the workload during the most active period of the production database and then replay it on a test database. Thanks to that, users are able to identify severe performance problems and eliminate them without affecting the production environment. To facilitate the analysis of potential problems, in addition to Real Application Testing it is recommended to use Automatic Workload Repository (AWR) [10] snapshots, which gather all the necessary information about the current system and Active Session History (ASH) [11], which keeping records information about current actively waiting sessions. Both tools are really handy, and due to automatically scheduled AWR snapshots, the user can review the database parameters before, during or after workload capture/replay (Figure 10).

Top 5 Timed Foreground Events

Event	Waits	Time(s)	Avg wait (ms)	% DB time	Wait Class
DB CPU		424		76.62	
log file sync	47,070	47	1	8.41	Commit
db file sequential read	997	44	44	7.87	User I/O
PX Deq: Slave Session Stats	7,733	14	2	2.53	Other
reliable message	14,405	13	1	2.42	Other

Figure 10. AWR statistics gathered during workload capture.

Thanks to the reference figures in AWR, it is much easier to understand the root cause of performance problems and to implement changes which address the top issues. If the user has already identified the cause of the problem, he can start applying the necessary changes. Those changes should be done in the fourth step. After that, the user needs to check the actual improvement by measuring the performance of the examined system. In order to observe and isolate the differences related to the changes made, it is recommended to perform several replays and then do comparison between them (Figure 11) rather than a comparison with the base workload.

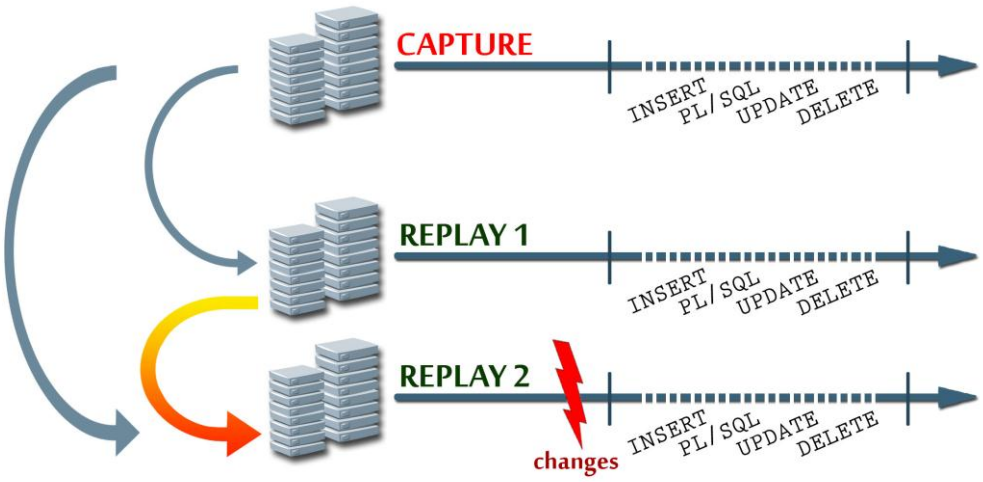

Figure 11. Comparison between several replay processes.

That approach is much more resilient than comparing with the production capture: indeed, instead of comparing every replay configuration with the original one, it is better to compare "good" replays between them each other, because changes already done by user are much more visible. When all method steps are completed and the tuning goal is not yet reached, an extra iteration can be made to further identify performance problems and, if possible, address them.

There are other tools that provide information about the usage of the database management systems, some of them can be compared with the tools provided by Oracle. They can be classified in logging tool and testing tools. For example log4J is a widely deployed tool that can gather lot of different logging information from applications, similar to the Oracle AWR snapshots. Analysing specific logging information can help identify the problematic component of applications and fix it. There is also a large number of testing tools exist. The Unit Testing tools are being increasingly widely deployed, the choice of the right tool depends on the technology used to develop the application is developed so that a tight integration can be achieved. The tools are necessary but the most important is to have a correct methodology.

4. Results and conclusions

The methodology developed and described in this article is simple and versatile. It was followed at CERN during the databases migration process to the Oracle Database 11.2 release, and proved to be reliable and problem free. In those few easy steps, tested environment can be checked with all possible configurations changes with minimal amount of time. Additional advantage of this methodology is that it can be applied not only to database management tools but also to other solutions.
However, in this study the methodology was used with Oracle database management tools, giving the opportunity to check characteristics of HEP applications on fully virtualized systems. Production workload was replayed on RAC database created on top of two virtual machines with different hardware configurations. The results have shown virtual servers not being as efficient as physical machines for database workloads, especially serving HEP applications. The time spent in database operations, called "database time", for the examined applications workload is more than 2 times bigger (in average) on virtual machines than on physical ones (Figure 12).

CASTOR Stager	PVSS	Dashboard
DB time:	**DB time:**	**DB time:**
00:09:00	00:11:47	01:01:19
00:29:38	00:30:52	01:41:52
00:25:16	00:30:47	01:31:48
00:23:41	00:24:11	01:27:12
00:24:14	00:26:17	01:28:21

Figure 12. DB time difference between physical and virtual systems.

By changing the hardware configuration settings, it was identified that increasing CPU count and memory size above certain threshold is not providing sufficient benefits. Tests have shown that the optimal configuration had 4 CPUs and 4 GB of memory for each RAC node for all of the tested applications. Increasing the number of CPUs and memory above these thresholds provided negligible performance improvements, while the cost of that change was inadequate in relation to the that gain (Figure 13).

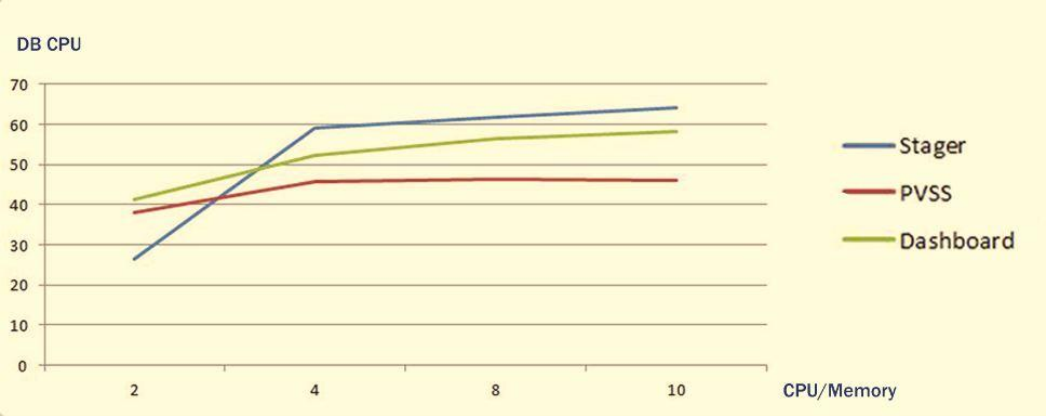

Figure 13. DB CPU workload replay characteristic.

The workload replay methodology is an help for new system configurations, databases and applications or for optimizing existing ones. There is still room for improvements in the capture and replay method described in this paper. For example, in order to have the replay systems be as close as possible to the production system states, one could warm up the database buffer cache before starting the replay process. Creating virtual machines with the newest virtualisation technologies can be done in future research with other test cases using the methodology described here. One should note that the characterisation of HEP applications as shown in this article was measured only from the database point of view, not from the application server or the developer side. More studies can be done by combining those two or more points of view in order to have an even more precise picture of the application usage.

References

[1] Craig S. Mullins, "Performance Tuning Requirements for Database Applications", http://www.craigsmullins.com/dbta_041.htm

[2] Dashboard, http://dashboard.cern.ch/

[3] CASTOR Stager, http://castor.web.cern.ch/

[4] PVSS, http://itcofe.web.cern.ch/itcofe/Services/OPC/RecommendedTools/Clients/SCADA/PVSS/welcome.html

[5] SCADA (Supervisory Control And Data Acquisition), http://www.automation.siemens.com/mcms/human-machine-interface/en/visualization-software/scada/Pages/Default.aspx

[6] AFS Console, http://afs-console.web.cern.ch/afs-console/cgi-scripts/index.cgi

[7] Drupal, http://drupal.org/

[8] HammerCloud, http://gangarobot.cern.ch/hc

[9] Oracle Real Application Testing, http://www.vldb.org/pvldb/2/vldb09-588.pdf

[10] AWR – Automatic Workload Repository, http://www.oracle.com/webfolder/technetwork/de/community/dbadmin/tipps/AWR_transport/index.html

[11] ASH – Active Session History, http://psoug.org/reference/ash.html