The Use of Service Desk System to Keep Track of Computational Tasks on Supercomputers

A.V. Bogdanov[1], V.Yu. Gaiduchok[2], I.G. Gankevich[1], Yu.A. Tipikin[1], and N.V. Yuzhanin[1(✉)]

[1] Saint Petersburg State University, Saint Petersburg 199034, Russia
igankevich@cc.spbu.ru, afk.apmath@gmail.com
[2] Saint Petersburg Electrotechnical University "LETI", Saint Petersburg 197376, Russia
gvladimiru@gmail.com

Abstract. This paper discusses the unusual use of service desk system–tracking of computational tasks on supercomputers as well as the collection of statistical information on the calculations performed. Particular attention is paid to the possibilities of using such statistics by a supercomputer user as well as the data center staff. The analysis of system requirements for tracking computational tasks and capabilities of service desk and job scheduler systems led to design and implementation of a way to integrate these systems to improve computational tasks management.

1 Introduction

As a result of rapid growth in HPC industry many powerful supercomputers appeared in the last years. They could be very efficient in terms of GFLOPS * hour, but one should always try to keep average resources load at an acceptable level in order to achieve good resource utilization. This implies monitoring, analysis and assessment, and it could be a really challenging task in modern hybrid systems consisting of heterogeneous resources (different CPU architectures, specialized accelerators like NVIDIA Tesla or Intel MIC). In turn multi-user environment imposes load balancing and security problems, logging and accounting tasks.

Problems mentioned above could be partly solved using Portable Batch System (PBS). There are many implementations of PBS that differ from each other in detail, but the main idea is the same. Such systems start jobs in accordance with a scheduler's plan that is based on user preferences, resources load and availability. The scheduler can be a part of entire PBS system or can be a separate software package. It makes a decision about job start time depending on system data collected by PBS. This data updates constantly which is reflected in log files. PBS also store information about jobs: requested resources, resources used, timeframes, etc., and such data is written in the log file too. So, in order to get accounting data one needs to parse large log files and retrieve necessary data.

There is a possibility to send e-mails to users when their jobs are started or stopped (with or without errors). But this is inconvenient too: there are usually too many e-mails

M.L. Gavrilova and C.J. Kenneth Tan (Eds.): Trans. on Comput. Sci. XXVII, LNCS 9570, pp. 1–9, 2016.
DOI: 10.1007/978-3-662-50412-3_1

with such information (users usually run many jobs), so this method will simply trash the box. Console PBS commands usually require some Linux skills, so they are inconvenient for inexperienced users too. In this article some approaches for controlling and managing shared HPC resources, which are quite convenient but not traditional, are reviewed.

Similar problems also considered by many authors [1, 2], but they often discuss the HPC in application to the tasks of one research team, so those tasks are analogical and often have a template solution. In our case the information from workload management systems is characterized by its variety and requires more complicated processing methods and tools.

2 Conflict of Interests

HPC center as any social institute involves working with social groups having different goals. Question of retrieving and processing monitoring information is answered differently by the two main groups– users and personnel of HPC center which have different capabilities and different goals. As a result, they need different representations of the monitoring information about the computational jobs. But these groups can also be divided into smaller ones. Users can be divided into executors and leaders while personnel consist of administrators, support engineers and managers. These subgroups have different requirements for the monitoring information too.

For example, user who executes scientific task and directly submits computational jobs to a PBS cluster needs to know which resources are available right now, view job queue, and investigate job failures. He needs to know execution time of finished jobs and amount of consumed resources. Important factor is user-friendly, clear representation of statistics. At the same time such user does not need to know details.

In contrast to executor, research work principal is not interested in particular job. He controls executors, plan the use of the resources which are necessary for work. Last but not the least, he must report to grantor. That is why he needs information about each executor: he needs statistics with the same level of detail as executor and at the same time he needs general reports reflecting the overall status of the research work of his scientific group. Tables are more suitable than graphs and diagrams for principal. He usually needs more parameters than executor as he ought to assess the overall process. He can also find automatic reports generation and prediction of resources usage very useful because he has to estimate the efficiency of the work and submit plans for oncoming quarter. He also has to submit requests for additional resources in case of lack of computational power.

On the other side of computational process there are system administrators and managers of the computer center. Administrators should have the possibility to get any information about any job. At the same time they need to get samples from overall statistics about cluster and system usage. For example, it can be necessary for capacity planning and load balancing. Also administrators have to track erroneous jobs and solve any problems related to jobs, system and hardware. So, they need a monitoring system that meets these requirements and also a service desk system which can be used to facilitate work with users and improve work flow. Such system can speed up overall work process and clear interaction with users by creating tickets.

Manager of computational center, in turn, wants to get information about overall resources usage. At the same time, he wants the possibility to assess the resource usage of specific user groups and he does not need detailed reports but an overview of total resource usage during the different time frames. Also, he needs an opportunity to sort reports by each resource and each scientific group.

As one can see, the requirements of mentioned groups have both intersections and differences. Universal system corresponding to the needs of all groups is difficult to implement and the problem is intended to be solved with a system based on a set of services attached to service desk.

3 Computational Infrastructure

Let's take a look at a computational infrastructure of a typical computational center before going deeply into the questions of service desk and monitoring system. Scientific computational clusters are usually equipped with one of Portable Batch System implementations. This system consists of three main parts: PBS Server, PBS Scheduler and PBS MOM with PBS Server being a key component of PBS system. Its task is to receive, modify, create and run jobs. PBS Scheduler decides when and where jobs should be started based on different policies, which administrator can set, and a current cluster load. It interacts with PBS Server in order to get new jobs and with PBS MOMs to retrieve current information about cluster nodes. PBS MOM (or Job Executor) actually starts jobs (creates new processes for jobs on cluster nodes). An instance of PBS MOM runs on each cluster node (moreover, several PBS MOM processes can be started within one node). It is called in this way because such process becomes a parent for all jobs. PBS MOM starts a job when PBS Server requests it. It starts a new session (such a session is identical to a certain degree to a common Linux user session). User needs to install tiny package containing basic PBS client commands. After that jobs can be submitted to a PBS server. All in all, the PBS work can be illustrated in the picture.

Queue is the key component of PBS and all cluster nodes are assigned to a particular queue. Usually there is more than one queue within a PBS system. Nodes of the cluster can be assigned to several queues at the same time. Queues can vary in parameters: priority, time restrictions for jobs, hardware restrictions (e.g. max number of CPUs for a job), etc. Cluster administrator should declare some policy to create queues and assign necessary parameters to them. For example, several queues that have different priority and different time restrictions can be created, i.e. some queues with high priority and severe time restrictions (job can be executed only several hours or minutes), and some queues with low priority and execution time restricted by days or months. So, when a job is submitted, a user can specify the queue based on his preferences. If he wants to compute his task right now, he should choose a queue of the first type.

Server management, submission of jobs and job control are performed using console and all PBS commands can be divided into three groups. The first one is user commands which can be performed by an authorized user. For example, 'qsub' command is used to submit a job to the queue, 'qstat' command is used for retrieving job status. User can also monitor the cluster status using 'pbsnodes' command and some parameters of PBS system.

The main information for customer is his jobs status which shows whether his job was started or not and how much time was consumed. Another set of commands is operator commands. Operator has permissions to monitor and control (hold, alter or terminate) every job. Last but not the least the commands that administrator can execute include all the commands of operator and customer: administrator can control all jobs, perform security tasks and manage cluster nodes.

At this point one can think that PBS is quite enough for performing monitoring of supercomputer systems, however, as with most Linux daemons PBS monitoring capabilities are limited to writing important information to logs. Moreover, in order to get working system that satisfies all user requirements, complies with center policies and is convenient for everyone one should solve some more problems.

One can face many problems while providing users with secure and comfortable access to the cluster. When there are several clusters in the center the problems only grows and expands. How to organize work with clusters, provide scientific data storage and comply with security policies? All these tasks can be done using virtualization. This technology becomes one of the most promising nowadays.

Users of the center are provided with personal virtual machines from which they can submit jobs to a cluster [3]. Also virtual machines can be used not only for job submission, but for data storage, visualization and even for simple pre- and postprocessing of data. Virtual machine can be viewed as an HPC resource when it is migrated to a specialized high-performance SMP node and its resource capacity (RAM, number of CPUs) is expanded accordingly. Such migration neither change configuration of user environment nor it implies explicit movement of data because it is saved in the home directory. So, virtual machine can be seen as a building block of computer center infrastructure.

Described approach has many advantages both for computer center staff and users. Computer center staff benefits from the following capabilities.

- Dynamic resource reallocation.
- Enhanced security.
- VM migration.
- Fault tolerance.
- Flexible configuring.

From the user perspective one can note next advantages.

- Enhanced security.
- High availability.
- Flexible VM configuring.
- Access to customized environment from anywhere.
- Ability to create own virtual clusters.

Data consolidation is one of the today's challenges. Imagine a typical situation for HPC center when many customers have access to many virtual machines and clusters, each user has different home directories on different servers. The other big problem is access to user data from cluster nodes. Both these problems can be solved using single mounted point for user home directory. Separate storage system can be used for this goal. All user's virtual machines mount the home directory from that storage. When user job is started on

cluster nodes his home directory is mounted automatically. So, virtual machines and cluster nodes mounts home directory from single place, and user has universal access to his data from all the resources. That is how data consolidation is achieved in our case.

4 Implementation

In its usual setting the service desk provides the single point of ticket reception, ticket workflow and archiving after the solution was found and the ticket was closed [4]. Also the service desk provides the permissions on actions with tickets to the helpdesk staff according their roles, accounts the incident solution time, provides reminders, notifications, ticket escalation mechanism, etc. Service desk system also usually has the knowledge base which contains most frequent incidents and its solutions.

The basis of our research is a model of supercomputer center which provides scientists with computational resources. Scientific work requires detailed report to the grantors, that is why such supercomputer center must provide detailed report and statistics to its users. In such center the service desk system can be the central point of the interaction between supercomputer center staff and users. All correspondence about the incidents also can be processed by service desk.

On the assumption of such position of the service desk it is reasonable to make the website based on the open source CMS as a frontend and the service desk as a backend of the user support system. On our virtual stand OTRS service desk and the website based on the Drupal CMS were integrated using the OTRS GenericInterface and SOAP protocol [5]. To track the computational tasks on the supercomputers the idea of receiving the email reports of PBS as tickets by the service desk was used.

It seems to be simpler to solve the problem without service desk and use direct transfer of the PBS email reports to the CMS database instead. In this way the processing can be done by a custom website module. However, the usage of service desk in this case has strong advantages. On the one hand the database schema development is not needed. On the other hand the service desk system has a convenient “ticket” object with automatic state machine. This object allows us to automatically open a ticket by the keyword in the email header and to close it the same way. And finally the service desk allows the supercomputer center staff to watch the overall picture of the batch system and to operate with the PBS error as with a trouble-ticket.

Also usage of the service desk system as the core of the information system of supercomputer center corresponds the good practices of the ITIL [6]. The integration of service desk with PBS and the website realizes a part of the capacity management.

Basic level of integration is very simple: the PBS server has an installed MTA (mail transfer agent) that sends reports to the service desk from the task executor account. The OTRS service desk has an additional queue and the mail-account connecting it with the email-address. User account on the website has a frame that displays the tickets based on computational tasks in PBS in turn. Basic element of integration between the website and the OTRS is a set of REST web services that get the data from the OTRS database and represents it to the user (Fig. 1).

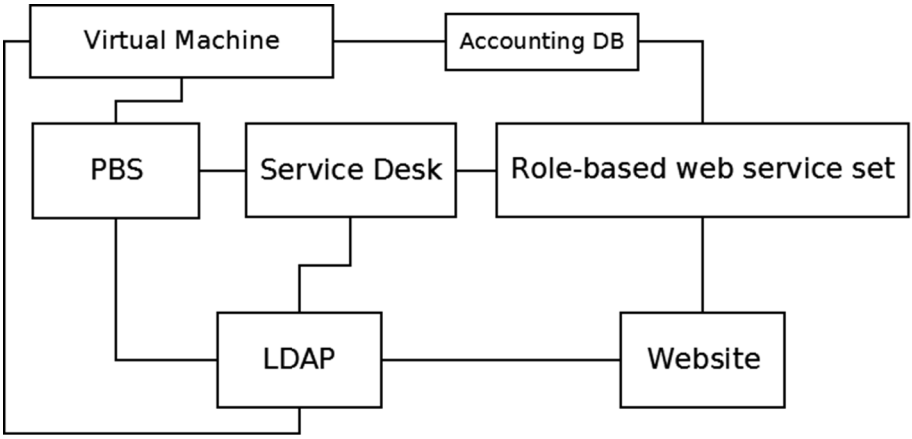

Fig. 1. Integration scheme.

Such configuration works and provides the information, but there are two significant disadvantages: there is no automatic change of ticket states and there are two tickets for every computational task. Large amount of tickets in the dashboard seems uncomfortable for the user. The second disadvantage is inconvenience of the data representation for the user and the lack of statistics and metrics. The first disadvantage can be eliminated by the usage of service desk module for the ticket state automation.

In particular the OTRS SystemMonitoring module can automate ticket state changing. This module gives a possibility to close tickets automatically by the keyword in the header or body of the ticket. The state changing rules are set by the regular expressions and are easy to configure. The purpose of the SystemMonitoring module is the integration of OTRS with different monitoring systems such as Zabbix or Nagios, but it can be used for nonstandard tasks. But this module cannot work with more than one source of information (email account). So it can be used either for the real monitoring or our unusual application, but not for both. That is why usage of this module on the production OTRS server is impossible without the loss of monitoring possibility. Such problem has two appropriate solutions: first is to set up alternative virtual OTRS server only for task tracking using common LDAP server and website interface; the second way to solve the problem is to use the set of postmaster filters based on the regular expressions instead of the SystemMonitoring module. However, the latter approach is complicated with sophisticated regular expression patterns and the resulting system may become unstable or slow due to inefficient processing of the filters.

The first solution allows not only to automate the ticket state changing process but also automatically set the special incident state of the ticket in case of program error keyword in the follow-up or by the timeout. Such incidents can be redirected and then processed by the technical support staff of the computing center. Such possibility is highly important for the system administrators and allows saving the technical support time from the manual batch system monitoring.

Table 1 displays the growth of the efficiency of error processing after the application of our solution. Earlier the period of manual batch system monitoring was

approximately 8 h, so the average time of the error detection was 4 h respectively. After deployment the system based on the service desk the manual monitoring is not actual so the average error detection time became 10 min and depends of service desk dashboard refresh time. The average time of the incident processing by the supercomputer center staff did not change and still 2 h. So the average error processing time decreased from nearly 6 h to slightly more than 2 h and the helpdesk staff spends their time 3 times more efficiently.

Table 1. Time costs of an error processing.

	Error detection	Incident processing	Total
Before	4 h	2 h	6 h
After	10 min	2 h	2 h 10 min

So we solved the problem of ticket processing automation but the representation of the information is uncomfortable for the user. The simplest way to represent the data is to put it into the tables. But the most comfortable way is to combine tables with graphs and diagrams.

The next problem is the statistics and metrics generation. Relevant information for the user depends on this user role: executor, research work principal, system administrator, computer center manager. Such dependency can be realized by the LDAP technology. LDAP account can contain role information and information about user's priority for the batch system. The OTRS, PBS and either the website uses the common LDAP server to associate the data relative with one user account. For the statistic report creation OTRS Stats module can be used. This tool can be controlled via SOAP, so user can choose reported parameters on the website and generate the report at any time interacting with simple and comfortable GUI.

The question of ordering and recording the activities of various groups of users on the physical computing resources is not a simple question, especially when organizing access to the resource through a heterogeneous system of services. Stats module can be used to gather statistics, but it is not finely tuned and does not allow changing the configuration of the output data "on the fly". For fast and fine-tuning data output, a web service connected to OTRS via the corresponding API can be used.

First, the possible information structure of datacenter can be considered. On the upper level (frontend) there is a portal (website) of center with a functional personal account. The instruments providing control of computational resource consumption will be placed in that account. On the lower (middleware) level there are virtual machines. Through them users execute their intensive computational tasks on the third level of infrastructure– clusters. Such infrastructure allows information about virtual machine resource utilization and information about running and completed tasks on clusters to be obtained and processed. To solve tasks it makes sense to use the ready-made solutions namely PBS for clusters and monitoring modules of VMWare vSphere for VM's.

Personal account is implemented as a separate module of Drupal CMS interacting with databases (CouchDB, ActiveDirectory) through REST web services and direct

requests using Sag library. The summary report of work made using datacenter resources contains resource utilization figures, however, datacenter systems that log activity of user virtual machines and applications running on cluster are not linked directly to this account. To display on-demand computing resource usage statistics, a service that would collect all metrics on the user virtual machines and represent this data in the user account in accessible graphical form.

To understand the structure of such implementation data sources for charts should be defined. Suppose that virtual machines accounting is stored separately in the SQL database. It is quite common solution, but this is not the most efficient way of storing such information because of large number of records. Consequently, the request gathering statistics of several years will be executed for an unacceptably long time. A more appropriate way of storing account data is to store the data in a distributed database using staging tables that already contain aggregated data up to current day. Information about tasks executed on a supercomputer is stored as chains of tickets: one for the beginning of the computational task and one for the end. This information is attached to personal account through PHP extensions for CMS and different criteria can be chosen to be displayed. PHP extension then submits request to the web service that contains the data and then pre-generated chart is received as a static picture.

A web service receives a list of user virtual machines from personal account and accounting information from OTRS and a combined chart is created for both cluster and virtual machine utilization. Based on received data web service creates a chart and returns it as a bitmap. The main difficulty lies in the fact that the request has to be completed in a short period of time. That is why the distributed database is a requirement. For chart creation free Java library JfreeChart is used. Such approach allows meeting the needs of executors and research work principal for the visual representation of data in the form of graphs and charts.

Graphical representation gives users more options for an adequate estimation of resources utilization, which further helps to avoid excessive (or lack of) resource allocation, increases efficiency of computational services and prevents from conflicts between customers and computer center staff. As a part of the development process resource quotas can be introduced for each research work group so that virtual machines can be created independently by customers from these groups. In this case, a fine-grained control of resources is transferred to customers' hands.

5 Conclusions

The task of integrating multiple information systems with comfortable user-friendly environment for computational task tracking allows an executor to make necessary quantitative evaluations of his work on the supercomputer. For example such system can be useful to make the report about numerical simulation performed within the confines of a research work or to track the computational resource quotas utilization and make predictions.

Our system will be especially useful for research work principals. Using the system they can evaluate the progress of all computational tasks within the confines of the group, the productivity of every subordinate, the measure and the uniformity of quotas

consumption. Also the automated report generator eliminates the need for manual generation of report after the research work ends. Metrics of total resource consumption is included into the report automatically.

Supercomputer center staff also will not be forgotten. Technical support can easily receive information about the total computational resource consumption and make the prognosis or a plan of further services. System keeps the data in the service desk database and allows archiving and making backups to protect the information. Also thanks to the database the reports can cover any time period. Such opportunity is very useful for the computing center managers.

Another advantage of solution based on the service desk system became a possibility to increase the efficiency of solving the program errors on the supercomputers. The time of error recognizing decreases thanks to automatic incident state changing. That is why overall time costs also decreases and the helpdesk staff works more efficient.

Acknowledgements. Research was carried out using computational resources provided by Resource Center "Computer Center of SPbU"(Official web site: http://cc.spbu.ru/.) and supported by Russian Foundation for Basic Research (project N 13-07-00747) and Saint Petersburg State University (projects N 9.38.674.2013, 0.37.155.2014).

References

1. Krompass, S., et al.: Dynamic workload management for very large data warehouses: Juggling feathers and bowling balls. In: Proceedings of the 33rd International Conference on Very Large Data Bases, pp. 1105–1115 (2007)
2. Davison, D., Graefe, G.: Dynamic resource brokering for multi-user query execution. In: Proceedings of the 1995 ACM SIGMOD International Conference on Management of Data, pp. 281–292 (1995)
3. Gayduchok, V.Yu., Bogdanov, A.V., Degtyarev, A.B., Gankevich, I.G., Gayduchok, V.Yu., Zolotarev, V.I.: Virtual workspace as a basis of supercomputer center. In: Proceedings of the 5th Internship Conference "Distributed Computing and Grid-Technologies in Science and Education" (Dubna, 16-21 July, 2012)/ Joint Institute for Nuclear Research (Dubna), pp. 60–66 (2012)
4. Kácha, P.: OTRS: CSIRT WorkFlow Improvements. – CESNET, Technical Report, 10 (2010)
5. Bakker, R., et al.: OTRS 3.3 - Admin Manual (2013). http://ftp.otrs.org/pub/otrs/doc/doc-admin/3.3/en/pdf/otrs_admin_book.pdf. Accessed 7 April 2014
6. Potgieter, B.C., Botha, J.H., Lew, C.: Evidence that use of the ITIL framework is effective. In: 18th Annual Conference of the National Advisory Committee on Computing Qualifications, Tauranga, NZ. C, pp. 160–167 (2005)