A Design of Application-Aware Profiling and Monitoring Architecture for Cloud Computing

Sheng-Hao Liu Wei-Jen Wang

Department of Computer Science and Information Engineering
National Central University
{93502025, wjwang}@csie.ncu.edu.tw

***Abstract*—A cloud computing system should put more attentions on system monitoring and application profiling because it demands better performance guarantee. For example, the cloud computing resource providers may charge users according to how the resources are used. Thus service quality implies business income. Previous grid computing approaches only focus on monitoring hardware components, failing to accurately depict the triangle relationship among the users, applications, and the resource providers. In this paper, we present an application-aware profiling and monitoring architecture which consists of three major components: the application-aware profiling agents, a profiling database, and filters. The application-aware profiling agents are installed on every computing node to keep track of the execution status of the applications. The observed information is then sent to the filters for preliminary processing and then to the profiling database, or directly to the profiling database.**

Observed data sets are formatted in the XML standard. Details of application activities thus can be found on the profiling database though user interfaces. The proposed architecture is able to integrate the information of users, applications, jobs, processes, and resources. When problems arise, applications with high performance guarantee can be identified to get timely service. This approach provides better performance guarantee and resource utilization. Based on our design, cloud computing service providers can develop different billing strategies, service-level agreements, and user privileges according to user payment and the observed application activity information.

***Index Terms* — Clouding Computing, Profiling, Monitoring.**

I. Introduction

A cloud computing system usually consists of a large amount of distributed, heterogeneous computing resources. It may concurrently host many different types of applications of different users. A cloud computing system involves resource providers (including system administrators), users, and applications (or application developers). The resource providers offer users a variety of fundamental computing services, such as storage systems and computing platforms. Applications can be built from resource providers' cloud services and other cloud applications. Resource providers may charge users according to how the resources and the services are used. As a result, it is important to provide monitoring and profiling services on a cloud computing system. Compared to a grid system, a cloud computing system must emphasize on system monitoring and application profiling because it should provide some degree of performance guarantee for users' payment — System monitoring helps users know what is happening in the system; application profiling abstracts historic records for administrators to fine-tune the application configurations.

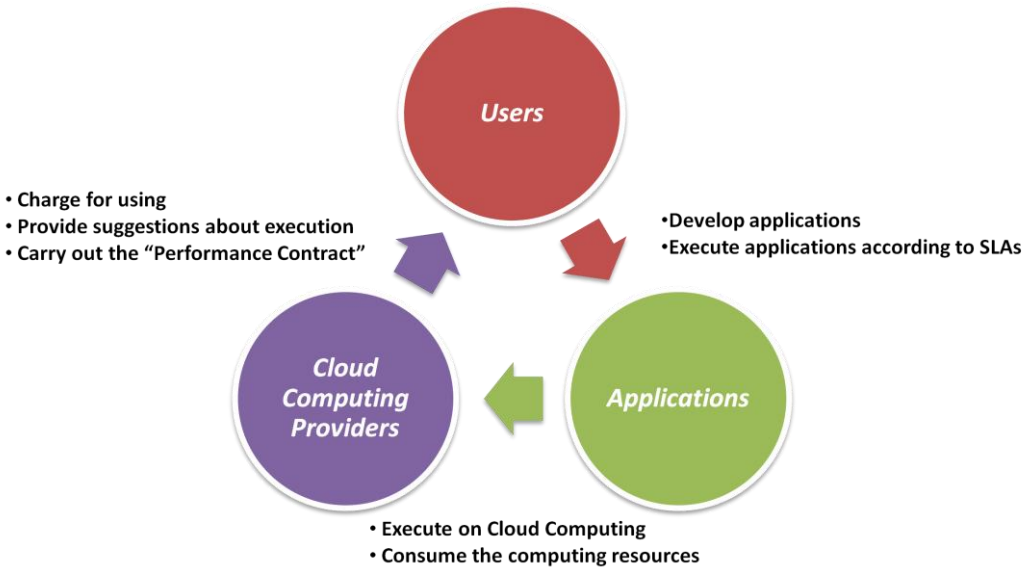

Figure 1. Relationship among users, applications, and cloud computing providers.

It is desirable for users/ administrators to be able to obtain the configurations of applications, but it is not easy because a cloud computing system is usually large and complex, involving considerable amount of computing resources and applications.

The ability of obtaining accurate system configuration is critical in cloud computing because wrong resource/application deployment can seriously degrade system performance. For example, a private business cloud [14] without using any suitable monitoring and managing service can suffer low efficiency of resource usage. As a result, applications on the cloud may require more resources and time than expected. Some applications with hard time restrictions may even cause financial loss. For example, some timely sales analysis applications may not be able to complete within the expected timeline and thus results in huge loss for the enterprise. In order to avoid the situation mentioned above, monitoring, profiling, and managing a cloud computing system becomes important issues.

In a computing system, any parts of the system can become the performance bottleneck of the system, such as hardware, networking, and system loading status [8]. Most existing grid computing approaches only focus on monitoring hardware components, such as CPU utilization, memory usage, I/O rate, etc. By using the monitoring data, a grid system can be dynamically reconfigured to improve efficiency. This approach is not enough in a cloud system, because cloud computing emphasizes the triangle relationship among users, applications, and cloud service providers, as shown in Figure 1. In the triangle relationship model, users develop their applications and execute them on a cloud system with some degree of quality guarantee (i.e. service-level agreements, SLAs [16]). Cloud computing providers must provide a way to ensure the quality of service (i.e. providing

a performance contract), and charge users according to the consumption and the quality of computing resources of applications. As a result, cloud computing demands not only resource-level monitoring but also application-level profiling and monitoring. The improvement of monitoring helps users have more control on application execution status. Users also acquire better performance guarantee as a "performance contract" between service providers and users [11].

A. Challenges

It is hard to monitor the behavior of applications on a cloud computing system because many types of cloud applications are distributed/parallel tasks. Compared to traditional sequential applications, distributed and parallel applications can potentially run in a complex way, such as MapReduce applications [13]. Furthermore, distributed/parallel applications usually rely on many communication and synchronization operations among computing nodes. Another challenge to monitor a cloud computing system is that it usually consists of dozens or even tens of thousands of computing nodes. Therefore, monitoring different types of applications on a cloud computing system is a fairly complex and difficult task.

B. Design Overview

In order to make our design of application-aware profiling and monitoring architecture relatively simple, we use Ganglia [2] [12] [15] as the basis of our research. The proposed system architecture consists of three parts: application-aware profiling agents, a profiling database, and filters (as mediate information collectors/senders). The application-aware profiling agents are installed on every computing node to keep track of the execution status of the applications. The observed information is then sent to the filters for preliminary processing and then to the profiling database, or directly to the profiling database. Details of application activities thus can be found on the profiling database.

C. Contributions

Most existing systems only monitor computing resources to get targeted information, such as CPU usage, memory usage, and network speed. The major difference between our system and existing systems is that the proposed architecture is an application-oriented monitoring architecture. Furthermore, it is able to integrate the information of users, applications, jobs, processes, and resources. When problems arise, applications with high performance guarantee can be identified to get timely service. Thus users are able to get better performance guarantee and service providers can have better resource utilization. Most important of all, cloud computing providers can take advantages of our system to develop a set of billing strategies, to create different service-level agreements, and to protect the rights of different customers who pay different money.

D. Structure of the Paper

The rest of the paper is organized as follows: Section 2 describes the related work, including several existing monitoring systems and the study of performance contracts. Section 3 describes the proposed architecture which shows how the system works, and how the components of the system communicate. Section 4 provides details of each system component. Section 5 presents implementation challenges and considerations.

Section 6 concludes the paper and discusses the future work of the paper. It reveals our plan of implementing and extending the proposed monitoring architecture.

II. Related Work

A. System Monitoring

Many existing monitoring systems provide the ability to monitor computing resources. Ganglia [2] [12] [15], developed by UC Berkeley, is a free, open source software. Ganglia uses a hierarchical design targeted at federations of clusters. Ganglia can monitor each individual computing node of a grid/cloud computing system with a Ganglia monitoring agent. The observed data from the monitoring agents of the entire computing system is delivered layer by layer, and eventually processed by a main node with the installed Gmeta [12] software. Ganglia is proven to be scalable and can monitor a large amount of computing nodes. Autopilot [4], developed by UIUC, is a system of effectiveness measurement. It helps dynamic performance tuning of computational grids. A variety of different monitoring systems targeting at different computing environments are still in development. More related software and systems can be found in [18].

There are also many programming-language-level monitoring tools in literature. For example, NetLogger [8] is an event-driven, visualization tools for distributed applications. NetLogger can obtain a variety of events which are generated by instrumented application components to understand the actual situation of an application, especially for multithreaded and parallel/distributed programs. NetLogger can give programmers and system administrators a clearer grasp of how computation is performed by each individual thread/process of an application. Therefore it is very useful for debugging. In addition, NetLogger supports various types of programming languages, including C, C++, Java, Perl, and Python by providing APIs for the above programming languages. OverView [19] is another programming-language-level monitoring tool which supports Java and SALSA [10]. It can replay computing events by animations.

B. Performance Contract

Most existing grid computing systems are designed as batch systems. Grid applications are mostly executed as batch-system jobs which are uploaded to some non-specific host. Thus a grid system emphasizes the total throughput and the average elapsed time of programs. Compared to the existing grid applications, many types of existing applications have a wider variety and diversity. For example, one may execute a long-term service, such as a web service; one can also execute a task which demands completion in a specific time range. Thus applications can be roughly classified by two user-defined restrictions, as shown in Figure 2: the time restriction (efficiency), and the stability

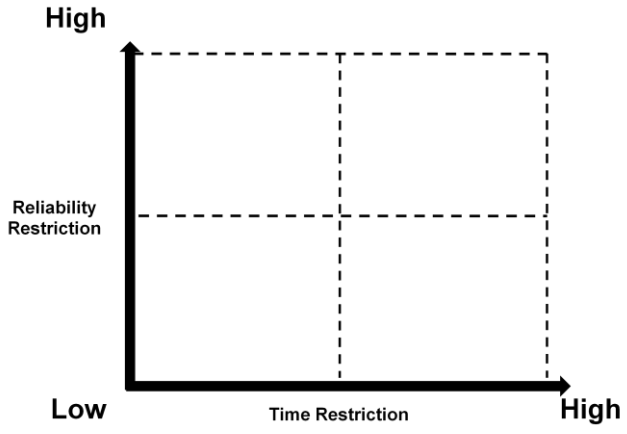

Figure 2. Applications types

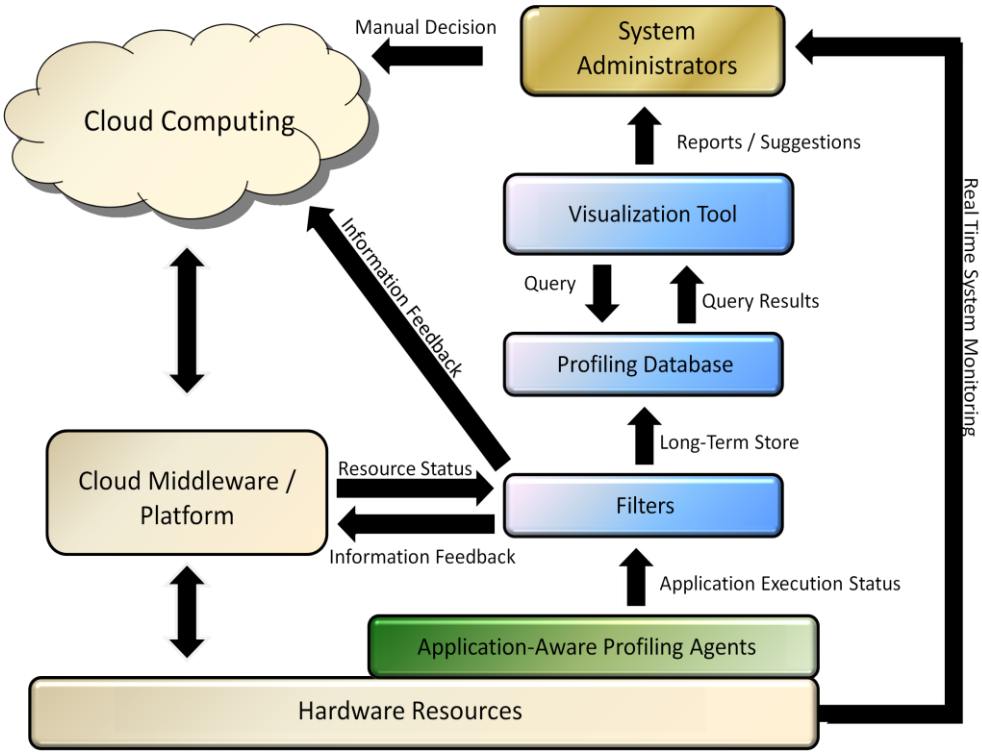

Figure 3. The proposed architecture.

restriction (reliability). Of course, we must consider more related criteria in order to achieve better service quality guarantee. This kind of service quality guarantee is called a performance contract [11].

A performance contract is a form of service level agreement (SLA). It is a kind of agreement between the applications, the cloud systems, and the users. The content should include requirements for the cloud system, which may cover the execution environment, computing resources, and user-defined restrictions. In other words, the contract asks the application to run in a certain way on the system. It simply is what users want their applications to be executed. From the perspective of users, this guarantee definition must be very straightforward. These user-defined descriptions can be regarded as the agreements signed by the users and the cloud system. From a resource-level view, the definition can be treated as how to select a computing task and how to allocate the underlying computing resources appropriately. From a system-level view, the definition can be treated as the coordination agreements of related services [16]. To negotiate the service level agreement and to carry out the performance contract , a cloud computing system requires more advanced monitoring services, in particular for application-level management and monitoring.

III. Proposed Architecture

In the proposed architecture as shown in Figure 3, application-aware profiling agents are installed on every computing node to monitor the behavior of applications and the usage of computing resources. When an application is executed on a cloud system, each involved application-aware profiling agent should be notified by the job/application execution

service. Then it can monitor the computing activity on each host and collect the information of each involved application on the node. Notice that a profiling agent may only have a partial view of the behavior of an application if the application is running across several computing nodes. The observed data, such as the life time of each application component (process/thread/object), the CPU usage, and the memory usage, could be sent to filters for preliminary processing. Filters are middleware which can collect selected data items and deliver them as feedbacks to resource management services. Filters can also aggregate the information which needs to be stored for further analysis, and then deliver the information to a profiling database. The information in the profiling database can then be used as a data source for analysis of long-term application behaviors. In addition, the information stored in the profiling database can also be sent to a visualization tool to provide human-readable data for system administrators. The observed data thus can help system administrators to manually reconfigure the system, and it can help identify the performance bottlenecks in the current system as well. System administrators can also take an initiative query to the profiling database. Since there are all kinds of application data sets in the profiling database, system administrators can choose a specific application as the targeted monitoring object and can also observe the computing activity of the targeted application at some specific time period. Furthermore, the system administrators can get real-time system information if they can receive information directly from the cloud system or the hardware. Notice that the information does not have to go through the application-aware profiling agents or filters.

Through the proposed application-aware profiling and monitoring architecture, users can have a grasp of the status of each cloud computing application, as well as the system performance in soft real time. The feedback information which is generated by the system can help fine-tune the system configuration, and it can also be used for performance bottleneck analysis. In other words, the proposed architecture improves the monitoring scope. Not only the proposed architecture monitors cloud computing resources like a grid computing system, but it also takes applications into consideration.

IV. System Components

The proposed architecture consists of three major components, two of which are derived from Ganglia and the other one is the profiling database.

A. Application-Aware Profiling Agents

Agents of a monitoring system are generally interested in collecting the usage data of computing resources. Some may provide limited job-aware or process-aware monitoring abilities. The proposed application-aware profiling agents are very different from existing works. The proposed application-aware profiling agents are developed based on Ganglia software. Thus they are able to monitor computing resources. A new developed module will be plugged into each of the original Ganglia monitoring agent. The new module should be able to communicate with job/application execution service to know which application components are sent to the host it resides in. Then the agent collects information for each different application. The concept of application-aware

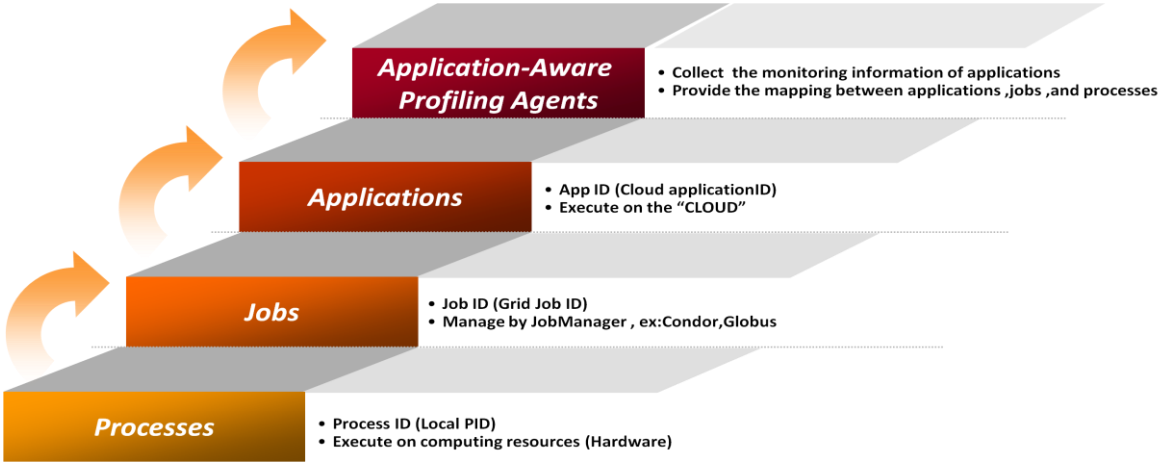

Figure 4. The relationship of applications, jobs, and processes in the proposed architecture.

profiling and monitoring simplifies a lot of management issues in cloud computing. For instance, the system administrators only need to set up the destination host of the observed data. In most cases the destination host is a web server which can illustrate the status of each application on each computing hosts. The system administrator may use the processed data to get a high-level overview of the cloud system because resource usage and applications are linked together. Since applications and users are linked together by default, the information of per-user-resource-usage can be identified easily. If users have to pay for resource usage, the proposed architecture can provide useful information for the resource providers to charge users.

One advantage of using application-aware profiling agents is that they hide the details of the complicated mapping relationship between applications and processes. The application-aware profiling agents should integrate the information of the users and the applications. A business strategy can be used here to assign different levels (priorities) to different user accounts while executing the applications. Figure 4 explains how the application-aware profiling agents handle processes, jobs, applications. First, an application may contain several jobs, while a job may also constituted by several processes. For example, a data mining application may generate a bunch of independent jobs with different parameters, and each job may generate several processes in a multi-processor system to accelerate data mining. Then the behaviors of the processes of a specific application are observed by the agents. Through the management interface, system administrators only need to know the application's ID, and all related observed data can be acquired easily.

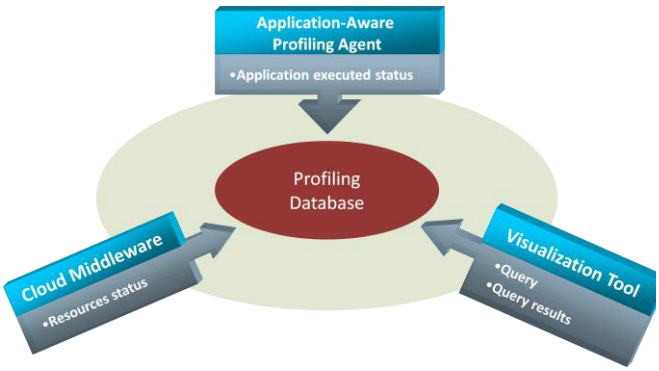

Figure 5. The role of the profiling database.

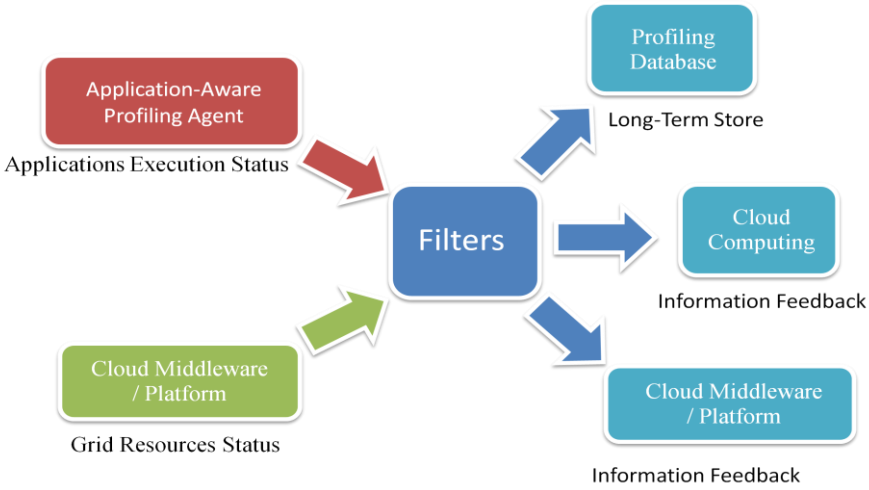

Figure 6. Data path via filters.

B. Profiling Database

The role of the profiling database is shown in Figure 5. The information generated by the profiling agents will be eventually transferred to a profiling database through the filters if the information needs to be stored. The data in the profiling database can provide system administrators a long-term or short-term view of applications. System administrators can use a visualization tool or a graphical web interface to connect to the profiling database to have a general view of the system. Users can also obtain the information of how their applications are executed via the visualization tool. A query command interface which supports advanced queries is provided as well. Other middleware services can also acquire information through the standard query interface. For example, a job scheduling service can use the historical data of a specific application to have a better arrangement of computing resources.

C. Filters

Filters in our system are mediate information transferring services, as shown in Figure 6. The observed data sets to be delivered in the data path use the XML standard. Information such as application behaviors is sent to the filters for preliminary processing. A middleware service can also provide information to the filters for any purpose. According to how the filters are designed, different types of data sets are delivered to different types of filters, and then re-transmitted to their final destinations. Filters are responsible for choosing the right data path. However, they are not responsible for complex data analysis or processing. This strategy can improve scalability and ensure data consistency as well.

V. Implementation Issues

A. Fault tolerance

The observed data is transmitted through the Internet in the proposed architecture. However, one cannot expect that the network environment is always stable. One cannot expect that the information is always transmitted normally either. When the data transmission problems occur, the

system temporarily loses contact with the application-aware profiling agents. Because the contact loss may last for a long time, the failure must be recovered properly. Problems may occur at any component of the proposed architecture. When a component (such as the profiling database) encounters a problem, a data backup and recovery mechanism or even a system recovery mechanism should be applied if needed.

B. Scalability

Since there are a lot of applications running on the cloud computing systems, we cannot just simply direct the data to one single point. Thus filters can form a hierarchical structure to process data, and they can also serve as a preliminary gatekeeper to remove unnecessary data sets. The profiling database is able to deal with huge among of data from all directions, such as the data from the application-aware profiling agents and the state of computing resources from the computing system. The role of the filters plays a very critical part in the scalability issues. With proper configuration of the filters, a cloud system can prevent delays due to the increase of data from the profiling and monitoring services.

C. Security Issues

Many security issues are needed to be considered in the design of the proposed architecture. At the beginning stage, we only focus on the certification mechanism, which is one of the most important issues of the proposed architecture to be discussed. Data in the profiling database contains the most detailed information, including the status of applications, the location of computing nodes, and the system's overall status. Since the system involves different users, different hosts, different applications, the system demands a certification mechanism to confirm identity of each roles. When components/users/applications are authorized, they can query the database according to their privileges.

VI. Conclusions and Future Work

The major contribution of the proposed application-aware profiling architecture is that it emphasizes the relationship among applications, users, and computing resource providers. By monitoring applications, system administrators no longer need to track the processes and jobs. The behaviors of applications can be understood better as well because the application-aware profiling agents handle the relationship among applications, processes, jobs, and users. As a result, all the complicated relationship is hidden from the users. System administrators need only to control high-end applications. Thus it simplifies system management. In addition, we also consider tuning application performance with the long-term observed application data, as well as the observed computing resource data. Filters transfer the observed data sets in the XML format. The data through filters can be sent to other middleware services or the computing resources as the feedback information to refine the system.

To conclude, we proposed a profiling and monitoring architecture for different types of applications on a cloud system. Users and system administrators can have a high-level scope to monitor and to operate the system, which makes monitoring and management more convenient on a cloud system.

In the future, we plan to implement the proposed architecture and measure the efficiency and the overhead of the proposed architecture. We may use some application/VM migration policies and the real-time monitoring feedback information to dynamically tune the performance of applications. An adaptive control mechanism will be established and the system can be adjusted automatically to provide better quality of services.

VII. Acknowledgements

This work was supported in part by the Taiwan National Science Council under Grant NSC-98-2221-E-008-080.

VIII. Reference

[1] The GridLab Project http://www.gridlab.org

[2] Ganglia http://ganglia.info/

[3] A.Iosup, N.Tapu, S.Vialle, "A monitoring architecture for control grids," Proceedings of the European Grid Conference (Lecture Notes in Computer Science),Vol. 3470,pp.922–931,Springer:Berlin,2005.

[4] A.Litke, K.Konstanteli, V.Andronikou, S.Chatzis,T.Varvarigou, "Managing service level agreement contracts in OGSA-based Grids," Future Generation Computer Systems,Vol. 24, Issue 4, pp.245-258,April 2008

[5] A. Waheed, W. Smith, J. George, and J. Yan. “An Infrastructure for Monitoring and Management in Computational Grid," Proceedings of the 2000 Conference on Languages, Compilers, and Runtime Systems(Lecture Notes in Computer Science), Vol. 1915,pp.922–931,Springer:Berlin,2000.

[6] B. Balis, M. Bubak, W. Funika, T. Szepieniec, R. Wismüller, M. Radecki, “Monitoring Grid Applications with Grid-enabled OMIS Monitor”, Proceedings of the First European Across Grids Conference, Vol.2970 ofLNCS, Santiago de Com-postela, Spain, pp.230–239,2003.

[7] B. Balis, M. Bubak, W. Funika, T. Szepieniec, R. Wismüller, "An Infrastructure for Grid Application Monitoring." Proceedings of the 9th European PVM/MPI Users’GroupMeeting, Linz,Austria, September/October 2002.

[8] B. Tierney, D. Gunter, "NetLogger: a toolkit for distributed system performance tuning and debugging," Proceedings of the IFIP/IEEE Eighth International Symposium on Integrated Network Management ,Vol. 246 of IFIP Conference Proceedings, Kluwer,2003, pp.97–100.

[9] B. Tierney, R.Aydt, D. Gunter,W. Smith,M. Swany,V. Taylor, R.Wolski,"A Grid Monitoring Architecture.", GGF Technical Report GFD-I.7, January 2002.

[10] C. A. Varela and G. Agha. "Programming dynamically reconfigurable open systems with SALSA", ACM SIGPLAN Notices. OOPSLA '2001 ACM Conference on Object-Oriented Systems, Languages and Applications, 36(12):20–34, Dec. 2001.

[11] D.A.Reed, C.L.Mendes, C.da Lu, I.Foster, C.Kesselman. "The Grid 2: Blueprint for a New Computing Infrastructure - Application Tuning and Adaptation.", Morgan Kaufman, 2003.

[12] F.D. Sacerdoti, M.J. Katz, M.L. Massie, D.E. Culler, "Wide Area Cluster Monitoring with Ganglia",Proceedings of the IEEE Cluster 2003 Conference, December 2003

[13] J.Dean ,S.Ghemawat, ”MapReduce: simplified data processing on large clusters.” Proceedings of the Sixth Symposium on Operating Systems Design and Implementation, San Francisco, pp.137-150,2004

[14] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph,

R. Katz, A. Kon-winski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Za-haria. Above the Clouds: A Berkeley View of Cloud Computing.Technical Report 2009-28, UC Berkeley, 2009.

[15] M.L. Massie, B.N. Chun, D.E. Culler, "Ganglia Distributed Monitoring System: Design, Implementation, and Experience",Parallel Computing Vol.30,pp.817–840,2004

[16] P.Hasselmeyer, H.Mersch, B.Koller, H.-N. Quyen, L.Schubert, P.Wieder. "Implementing an SLA Negotiation Framework". Proceedings of e-Challenges 2007

[17] R.L. Ribler, J.S. Vetter, H. Simitci, D.A. Reed, "Autopilot: adaptive control of distributed applications," Proceedings of the 7th IEEE Symposium on High-Performance Distributed Computing, pp.172–179,1998

[18] S. Zanikolas, R. Sakellariou, "A taxonomy of grid monitoring systems",Future Generation Computer Systems Vol.21,pp.163–188,2005

[19] Travis Desell, H. Iyer, Abe Stephens, and Carlos A. Varela. "OverView: A Framework for Generic Online Visualization of Distributed Systems". In Proceedings of the European Joint Conferences on Theory and Practice of Software (ETAPS 2004), eclipse Technology eXchange (eTX) Workshop, Barcelona, Spain, March 2004.

[20] Z. Balaton and G. Gombás, "Resource and Job Monitoring in the Grid", Proceedings of the 9th International Euro-Par Conference,Vol. 2790 of LNCS, Klagenfurt, Austria, pp.404