Accounting Information Gathering System for Grid Environment

Haeng Jin Jang**, Gil Su Doo***, Jeong Jin Lee*,
Beob Kyun Kim*, Ho Jeon Hwang*, Dong Un An*, Seung Jong Chung*
* Korea Institute of Science and Technology Information
** Dept. of Electrical & Electronic Engineering, Seonam University
*** Dept. of Computer Engineering, Chonbuk National University
Tel : +82-063-270-2412 Fax : +82-063-270-2394 E-mail: hjjang@kisti.re.kr,
doogilsu@naver.com, jeongjin2@hanmail.net, {kyun, hjhwang}@duan.chonbuk.ac.kr,
{duan, sjchung}@chonbuk.ac.kr

Abstract: Grid computing represents the fundamental computing shift from a localized resource computing model to a fully-distributed virtual organization with shared resources. Accounting is one of the main obstacles to widespread adoption of the grid. Accounting has until recently, been a sparsely-addressed problem, particularly in practice. In this paper, we design and implement the accounting information gathering system. Implemented system is based on OGSA, following GSAX framework of RUS-WG in GGF. And the schema of gathered and serviced accounting information is following Usage Record Fields of UR-WG in GGF. Also, the accounting information integrating and monitoring tool for system management in the grid environment are implemented.
Grid, Accounting, Process, Job Manager

1. INTRODUCTION

Grid computing represents the fundamental computing shift from a localized resource computing model to a fully-distributed virtual organization with shared resources [1]. Fueling the emergence of grid computing is the ever-increasing cost of local information technology resources. With the Grid, companies achieve a cost efficient and effective mechanism for building and deploying applications across a wide spectrum of devices.

Several commercial obstacles, most notably security and accounting, have impeded the widespread adoption of the Grid, Several projects around security and authentication have begun both within and outside the Grid community, enabling companies to confidently use Grid services. Accounting for these services has until recently, been a sparsely-addressed problem, particularly in practice. The grid community has yet to produce either framework or, better still, an implementation of grid accounting [2].

We design and implement the accounting information gathering system. Implemented system is based on OGSA (Open Grid Service Architecture) [4], following GSAX (Grid Service Accounting Extension) framework [2] of RUS-WG (Resource Usage Service) in GGF. And the schema of gathered and serviced accounting information is following Usage Record Fields [7] of UR-WG (Usage Record) in GGF. The system comprises of several modules which work independently from each other. In addition, the accounting information integrating and monitoring tool for the system management in the grid environment are implemented.

2. RELATED WORKS

2.1 GSAX

GSAX [2] is an extensible OGSA accounting and logging framework. It is designed to provide a functionally modular accounting framework which can be expanded by adding or changing components, to allow use of accounting at many levels of application and user understanding, to provide information at different levels of granularity (from real-time information to data on a per-job basis), to integrate QoS and service-level agreements into the accounting framework, and at different levels, to be independent of any economic model, and to allow dynamic pricing stages.

2.2 DGAS

DGAS (Datagrid Accounting System) model, developed by DataGrid Project [9], envisions a whole new economic Grid market, where supply and demand of Grid resources work in unison to strive towards equilibrium where all resources are fully utilized to the lowest possible price. The Home Location Register (HLR) acts as a local bank branch managing the fund status of a subset of Grid users and resources.

But, such centralized solutions are not in agreement with the decentralized nature of the Grid. So, the system, designed in this paper, is follows GSAX framework.

2.3 Minimum set of Usage Record Fields

UR-WG in GGF provides information to the Grid community in the area of usage records and accounting. It summarizes the usage record fields used at a sampling of different sites and gets an

overview of the types of data that are being used for accounting in existing systems and determines a minimum set of usage record fields anticipated to represent the accounting needs for contemporary systems requesting or supplying resources to a grid.

3. DESIGN OF ACCOUNTING INFORMATION GATHERING SYSTEM

3.1 Schema of Usage Record

Accounting in the grid environment is very different from that in the traditional computing environment, because the concept of the user is different from the traditional local user and the format of accounting data of each system is different from each other. Accounting information in the grid environment is not produced by the local user but by the grid user. And the format of accounting data on linux is different from that on other operating system.

We chose Usage Record format [7], proposed by UR-WG in GGF, as a common usage record format in the grid environment.

Table 1 Minimum Set of Usage Record (UR-WG)

Num	Field Name	Num	Field Name
1	Username	11	NumNodes
2	ProjectName	12	CpuTime
3	JobId	13	WallTime
4	Queue	14	Memory
5	GridId	15	Disk
6	FromHost	16	Network
7	execHost	17	jobName
8	StartTime	18	Status
9	EndTime	19	Charge
10	Processors		

3.2 Accounting Information Service

Because of characteristics of the grid environment, most of grid programmers try to keep the autonomy of each site with minimum intrusion. Thus, the use of the output of the local accounting system is preferred.

We design the accounting information service system to be independent from any other services or resources and to follow the GSAX framework. Each resource sends its accounting data to the accounting information service.

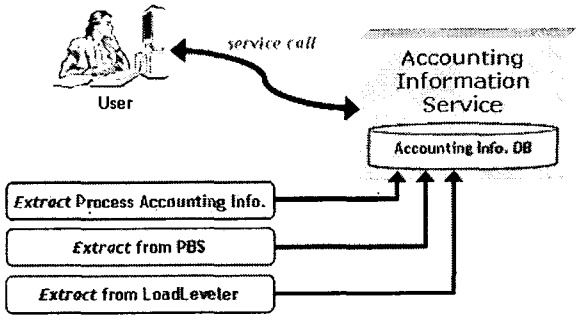

Fig. 1 Accounting Information Service

To obtain his or her accounting information, user uses a service call following OGSA.

3.3 Gathering Process Accounting Information

Most of Unix operating systems provide utilities for monitoring process activities on each system. For the Linux, psacct package contains several utilities for monitoring process activities. The result of the process monitoring is saved into the file "pacct". The location of this file is different from each operating system and site. We use this file to extract process accounting information. This file contains information sufficient for Minimum Set of Usage Record (Table 1). The extracted process accounting information from this file is sent to accounting information database in the Accounting Information Service. We tested it for IBM AIX 4.3.2, IBM AIX 5.1L, Linux 8.0, Linux 9.0.

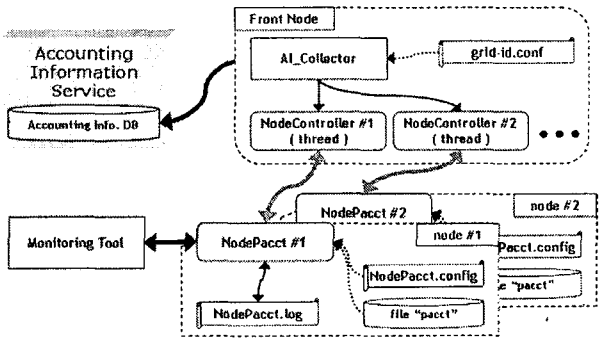

Fig. 2 Architecture for Gathering Process Accounting Information

Figure 2 shows the architecture for gathering process accounting information, designed in this paper. If a machine is structured as a cluster, AI_Collector is located in the front node only and creates NodeController for each slave node. For each slave node, NodePacct is created by the Monitoring Tool which may locate outside of this machine. NodePacct, located in each slave node, collects process accounting information from the file "pacct" and interacts with NodeController, located in the front node and created by AI_Collector. Process accounting information in each NodeController is gathered by AI_Collector and sent to accounting information database in the Accounting Information Service. Process accounting information is formatted so that it follows Minimum Set of Usage Record (Table 1).

Configuration file "grid-id.conf" contains the list of local usernames that allocated to the grid user. Each NodePacct collects process accounting information which is produced by these users. Environment Variables for extracting process accounting information is contained in "NodePacct.config" and Check points for reading the file "pact" is contained in "NoePacct.log"

We designed this architecture to be scalable. NodeController in the front node is created against each NodePacct in the slave node. So, if new slave node is added, NodeController is created automatically. That is, adding and controlling of slave node is very easy.

3.4 Gathering Accounting Information from Job manager

The system, designed in this paper, can collect accounting information produced by job manager. We had applied to LoadLeveler, the job manager for IBM AIX machine. Most of job managers can produce accounting information for the machine. So, we use this accounting information of the machine to gather grid accounting information. Figure 3 shows the architecture for gathering accounting information produced by LoadLeveler.

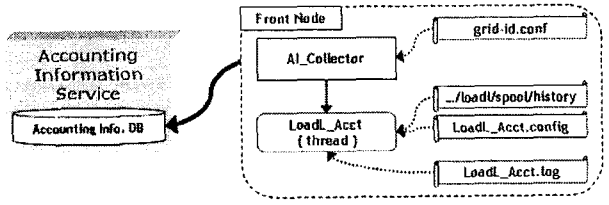

Fig. 3 Architecture for Gathering Accounting Information produced by LoadLeveler

AI_Collector in Figure 3 is the same module in Figure 2. AI_Collector initiates LoadL_Acct to collect accounting information. LoadL_Acct extract accounting information from the history file of LoadLeveler. This file contains accounting information of the job which is executed via LoadLeveler. Check points information for reading the history file is contained in "LoadL_Acct.log." Accounting information is formatted so that it follows Minimum Set of Usage Record (Table 1).

Application to PBS (Portable Batch System) or other Job Manager is very easy to achieve by developing only the module, which plays the role for LoadLeveler.

4. IMPLEMENTATION

4.1 Implementation of Accounting Information System

We implemented this system in the following environments. For the portability of this system, Java is selected. We use MySQL for DBMS. But, for the service following OGSA, we would like to recommend the native XML database [11].

Table 2 Experimental Environments

	Environments
Language	Java 1.4.2
DBMS	MySQL
Operating System for process accounting	Redhat 8.0, 9.0, AIX 4.3.2, AIX 5.1L
Job Manager	LoadLeveler on AIX 5.1L

Figure 4 shows an example of gathered process accounting information. In the figure 4, we can see all the required information without the grid user identity. To find the grid user who creates these processes, utility for access control in the grid environment is needed [10].

Figure 5 shows an example of the operation of AI_Collector. The accounting information from the slave nodes is transferred to accounting information database in the Accounting Information Service. Figure 6 shows an example of the operation of NodePacct, sending extracted accounting information to the NodeController in the front node.

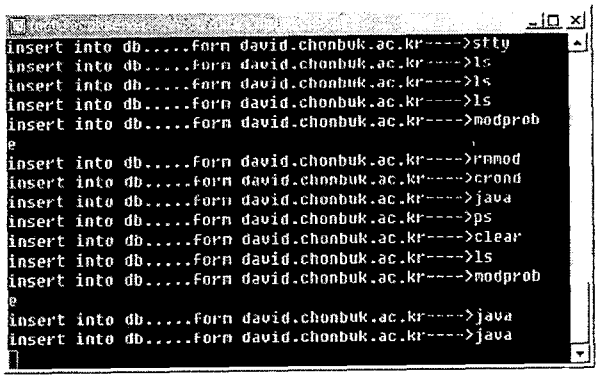

Fig. 5 AI_Collector

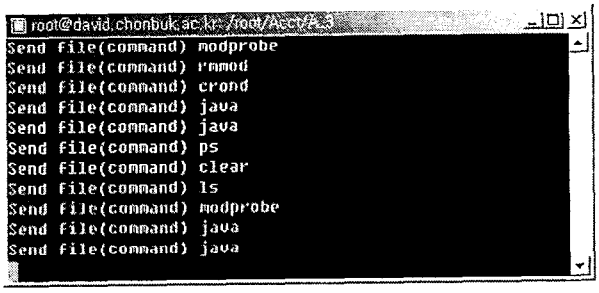

Fig. 6 NodePacct

	HostName	CommandName	UserId	Tty	BeginTime	EndTime	RealTime	SystemTim	UserTim	CharsIO	BlockRW	CpuFacto	HogFactc	MeanMemSiz	KcoreMir	ActFlag	ExitStat
▶	node20.chonbuk.ac.kr	#globus-g	hjhwang	?	20021117200041	20021117200044	3.22	0.02	0.94	43544	0	0.984	0.296	1136	18.05	2	0
	node20.chonbuk.ac.kr	date	hjhwang	?	20031117200049	20031117200049	0	0	0	31	0	0	0	0	0	0	0
	node20.chonbuk.ac.kr	globus-j	hjhwang	?	20021117200044	20021117200049	5.61	1.06	0.27	192384	0	0.2	0.237	410	9.08	0	0
	node20.chonbuk.ac.kr	#globus-g	hjhwang	?	20021117202245	20021117202246	1.58	0	0.62	43544	0	1	0.396	1121	11.68	2	0
	node20.chonbuk.ac.kr	date	hjhwang	?	20031117202251	20031117202251	0	0	0	31	0	0	0	0	0	0	0
	node20.chonbuk.ac.kr	globus-j	hjhwang	?	20021117202247	20021117202252	5.52	1.03	0.33	192320	0	0.241	0.246	519	11.76	0	0
	node20.chonbuk.ac.kr	#globus-g	hjhwang	?	20021117202358	20021117202400	2.69	0.02	0.89	43536	0	0.983	0.337	1140	17.22	2	0
	node20.chonbuk.ac.kr	ls	hjhwang	?	20031117202405	20031117202405	0	0	0	5	0	0	0	76	0	0	0
	grid.chonbuk.ac.kr	#globus-g	griduser01	?	20031118162118	20031118162120	1.2	0.8	0.35	28040	0	0.53	0.18	2230	16.02	2	0
	grid.chonbuk.ac.kr	df	griduser01	?	20031118162120	20031118162120	0	0	0	21	0	0	0	0	0	0	0
	grid.chnbuk.ac.kr	globus-j	griduser01	?	20031118162118	20031118162120	5.5	1.02	0.33	4218	0	0.1	0.212	84	7.02	0	0
	tea01.chonbuk.ac.kr	#globus-g	user01	?	20031118163847	20031118163852	2.82	0.02	0.92	43532	0	0.887	0.282	1124	16.02	2	0
	tea01.chonbuk.ac.kr	cpi	user01	?	20031118163849	20031118163849	0	0	0	20	0	0	0	0	0	0	0
	tea01.chonbuk.ac.kr	globus-j	user01	?	20031118163848	20031118163852	4.81	0.8	0.25	185020	0	0.228	0.326	1138	8.06	0	0

Fig. 4 An Example of Gathered Process Accounting Information

4.2 Monitoring Tool for Process Accounting

Also, we implement the monitoring tool for process accounting (Figure 7). Through this monitoring tool, the site administrator can initiate or sleep NodePacct. This tool can be applied to job managers.

5. CONCLUSION AND FUTURE WORKS

In this paper, we design and implement the accounting information service. This service can integrate the accounting information of process and from job manager. This service follows the GSAX framework and the accounting information is formatted so that it follows Minimum Set of Usage Record. It is very easy to add or delete slave node. Because the architecture of the gathering system is layered, extension to other job manager or other kind of platform is very easy. Application to other Job Manager is very easy to achieve by developing only the module, which plays the role for LoadLeveler.

But, for the future application, the native XML database would be used as DBMS. And more various kind of Job Manager would be considered.

References

[1] I. Foster, C. Kesselman(eds), S. Tuecke Q.25, "The Anatomy of the Grid:Enabling Scable Virtual Organizations", Intl. J. Supercomputer.

[2] A. Beardsmore et al, "GSAX(Grid Service Accounting Extensions)", (draft), GGF6, 2002

[3] S. Mullen et al, "Grid Authentication, Authorization and Accounting Requirements Research Document", (draf), GGF8, 2003

[4] S. Tuecke et al, "Open Grid Services Infrastructure (OGSI)", (draft), GGF, 2003

[5] http://www.ggf.org

[6] http://www.globus.org

[7] http://forge.gridforum.org/projects/ur-wg

[8] http://www.gridforumkorea.org

[9] http://eu-datagrid.web.cern.ch/eu-datagrid

[10] Beob Kyun Kim et al, "Grid Access Control System for Site Autonomy", ICEIC, 2004

[11] http://www.xmldb.org/xapi

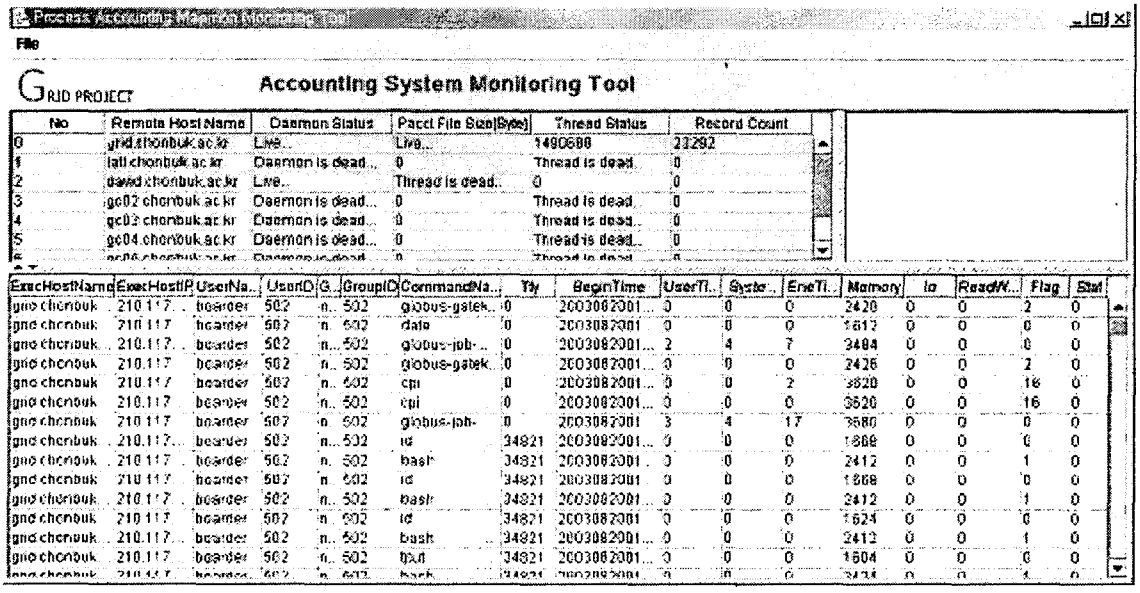

Fig. 7 Monitoring Tool for Process Accounting