

A Grid Portal for Grid Resource Information Service

Phil Doo Hong* **, Yong Woo Lee (Corresponding Author)*
Seoul Grid Center*, Meritz Financing Group**
iamhpd@nate.com, ywlee@uos.ac.kr

Abstract — RT-GRIS, our grid portal, has two purposes. One is effective CPU scavenging and other is supporting a great number of grid node. CPU-scavenging, cycle-scavenging, cycle stealing or shared computing creates a “grid” from the unused resources in a network of participants. It is an important technique for a distribution of grid job and node resource. And, if the grid portal uses fewer resources, those resources are more available to the grid nodes. Eventually the grid portal is supporting a great number of grid nodes. For this reason, the following four kinds of techniques are proposed, first is HDA(hierarchical data access), second is SMF(Standard Message Format), third is GRISP(GRIS Protocol), last is RRM(Rollback Recording Mechanism) / RCM(Reconnecting Mechanism).

RT-GRIS has been built and operated at real-fields which are M-Grid and U-Grid using these proposed conception. In this paper, architecture design and implementation of GRIS system are explained. Particularly, performance of RT-GRIS system is evaluated clearly.

Keywords — Grid, Grid Portal, CPU Scavenging, Real-time Push Service, Ubiquitous Computing, distributed system.

1. INTRODUCTION

The Grid Computing means the sharing between many computing resources. And the Grid has three points. First is that coordinate resources which are not subject to centralized control. Second are that using standard, open, general-purpose protocols and interfaces. Last is that deliver nontrivial qualities of service.

A grid portal is a high-level tool that helps application users use the grid infrastructure more efficiently. However, the current technology of grid portals has limitations. The cpu scavenging, the main purpose of grid computing should be run more efficiently. Such as interactive-method or simple push-method, current node managing methods of grid node are limitations to perform CPU Scavenging. For this reason, the following four kinds of techniques are proposed, HDA, SMF, GRISP and RRM / RCM.

In this paper, we will present our grid portal, called RT-GRIS, from the viewpoints of advanced job-scheduling and fast information delivery system. In chapter 2, we will explain related work in RT-GRIS, the world-wide grid portal, cpu scavenging and job scheduling, polling and event method and MDS (The Monitoring and Discovery System). In chapter 3, we will provide our motivation and contribution. In chapter 4, we will discuss our system design conception. In chapter 5, we will explain about our system implementation. In chapter 6, we prove our proposition. Finally, we will present our concluding proposal.

2. RELATED WORKS

A. The study of the world-wide grid portal.

Nowadays, many labs are concentrating on research of Grid portals.[1][2][3][8][9][10][11][12][13] G-Monitor was developed in response to the need for Web-based portals that hide low level details of accessing Grid services for deployment and execution management of applications. It was developed by the University of Melbourne, Australia [6]. A Science Portal is a problem solving environment that allows scientists the ability to program, access and execute distributed "Grid" applications from a conventional Web Browser and other desktop tools. It was developed by Indiana University, USA [4]. The UCLA Grid Portal (UGP) is a web application that provides web services in support of high performance computing [7]. In order to construct Grid portals easily, many Development Kits were released. Some examples of these cases are the GPDK (Grid Portal Development Kit) [4] and the Gridsphere [5]. The Gridsphere project is focusing on the study of support for Globus Toolkit 4 Project and Web Service. It is not necessary to use the development tool-kits to construct Grid Portals, In fact, in many cases, Grid portals are developed directly without using the tool-kits. But when the tool-kits are utilized, they offer some advantages such as Plug & Play, reduced development time, etc. However, in order to use SNMP in MDS (Monitoring and Discovery System) of Globus, this grid portal are not able to execute fast grid resource information service. GRIS (Grid Resource Information Service) is our new grid portal for advanced job scheduling using fast grid resource information service.

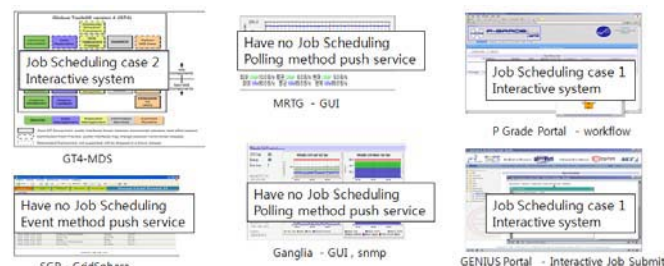


Figure 1. World-wide grid portals

B. CPU Scavenging and Job Scheduling

CPU-scavenging, cycle-scavenging, cycle stealing or shared computing creates a “grid” from the unused resources in a network of participants whether worldwide or internal to an organization. Volunteer computing projects use the CPU scavenging model almost exclusively. In practice, participating computers also donate some supporting amount of disk storage space, RAM, and network bandwidth, in

addition to raw CPU power. Since nodes are likely to go "offline" from time to time, as their owners use their resources for their primary purpose, this model must be designed to handle such contingencies.

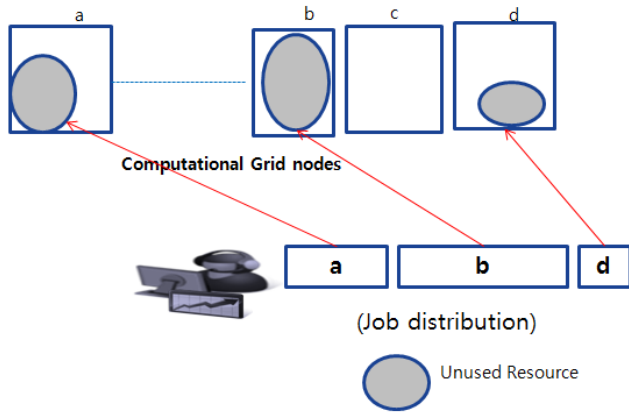


Figure 2. CPU Scavenging

Job scheduling is traditional issue in grid field. Job scheduling is a process of job distribution by node resource status. In this process, distribution and relocation method of job are important. Most of the grid portal was passively job scheduling. One job is distributed and placed according to grid nodes status at the time of execution of the job. But, the relocation of the job does not happen during execution of the job. When one node fault or node hanging is happened during the job execution, then the job will be stop in all nodes.

The job scheduling has these 3 cases. Case1 is that the job scheduling is not happening from the point of job starting to the point of job ending. Case 2 is that job scheduling is happening at the time-slice periods from the point of job starting to the point of job ending. Last case is that Job Scheduling is happening with real-time – node's status information such as node-fault, node-crash or network-fault, etc.

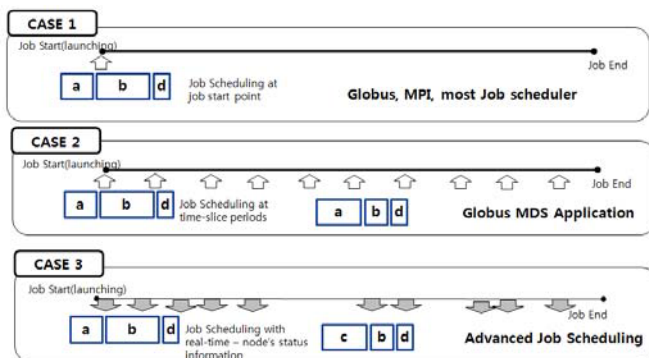


Figure 3. Job Scheduling

If not expensive job-node switch cost such as a context switch, then the case3 scheduling is nice method. The ideal grid system shall have the following characteristics. Also, the following is our grid system's goal.

- Real-time scheduling based on external, un-predictable events
- Automatic restart and recovery in event of failures

- Alerting and notification to operations personnel
- Generation of incident reports
- Audit trails for regulatory compliance purposes

C. Polling and Event method

In interactive system, if the system do not request, do not respond. Therefore, the grid-system cannot know whether any grid node's problem is happened before the inquiry the node-information. But the push system doesn't look for the information the users want but actively send users the information they want. The push system is divided into two different ways. One is the polling method. Another is event method.

TABLE 1. PUSH SYSTEM

Polling method	Retry request by system with time-slice period.
Event method	When event occurred, send information automatically.

D. MDS (The Monitoring and Discovery System)

The Monitoring and Discovery System (MDS) is the information services component of the Globus Toolkit and provides information about the available resources on the Grid and their status. MDS uses the SNMP (Simple Network Management Protocol) internally. SNMP uses the polling method in gathering node-information. But, our MDS of RT-GRIS uses the event method.

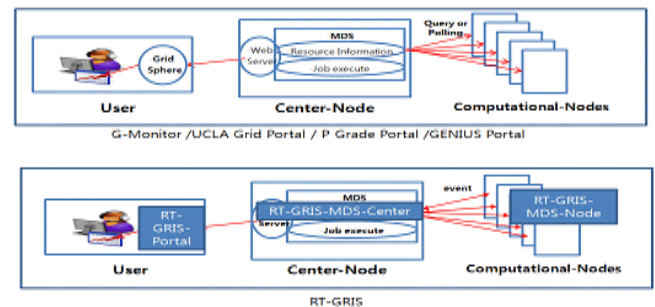


Figure 4. MDS vs. RT-MDS

3. MOTIVATION & CONTRIBUTION

RT-GRIS (Real-time Grid Resource Information Service) is our new grid portal for advanced job scheduling using fast grid resource information service.

RT-GRIS has two purposes. One is effective CPU scavenging and other is supporting a great number of grid node. CPU-scavenging, cycle-scavenging, cycle stealing or shared computing creates a "grid" from the unused resources in a network of participants (whether worldwide or internal to an organization). It is an important technique for a distribution of grid job and node resource. And, if the grid portal uses fewer resources, those resources are more available to the grid nodes. Eventually the grid portal is supporting a great number of grid nodes. For this reason, the following four kinds of techniques are proposed, first is HDA(hierarchical data access), second is SMF(Standard Message Format), third is

GRISP(GRIS Protocol), last is RRM(Rollback Recording Mechanism) / RCM(Reconnecting Mechanism).

4. SYSTEM DESIGN

A. HDA (Hierarchical Data Access)

Our Grid Portal has three hierarchical layers and is operated with some special method we called HDA. This HDA method has some useful factors for our data control method. A conceptual configuration in a common grid is shown in this figure.

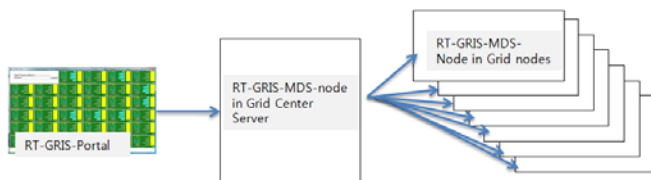


Figure 5. RT-GRIS conceptual configuration

The data store has three locations in our grid system. They are place at main memory of connecting node, internal disk of connecting node and internal disk of central database server. We define the three layers base on this topic. We call these as MMDB layer, Node DB Layer and Center DB Layer. Data is located in one of three layers and decided placement by character of the data and data location is flowing. The three layers have hierarchical classes. First class is center DB layer. Second class is Node DB Layer. Third class is MMDB Layer. But, the data is stored in two places; one is the RT-GRIS-MDS-node in Grid Center Server. Another is the RT-GRIS-MDS-Node in Grid nodes.

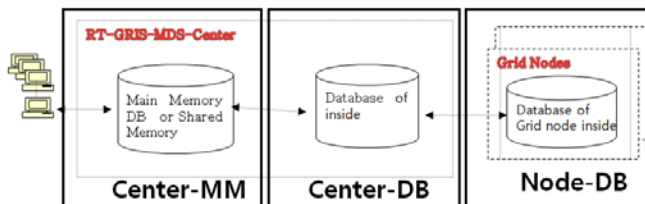


Figure 6. HDA (Hierarchical Data Access)

B. SMF (Standard Message Format)

In our grid portal, for efficient communication, we were all unified communications protocol. we simply called SMF (Standard Message Format). The following are examples of packet format.

```
ARP|/home/stock/alpha-run/load/NeoWEBTr_mng|71986|
stock|0.2|0.0|120|132|-|A|05:03:47|1:23|test|1|103642|0
```

- ARP : Process status type
- /home/stock/alpha~ : processing name and running directory name.
- 71986 : PID (process ID)
- stock : user
- 0.2 : %CPU
- 0.0 : %MEM

- 120 : SZ(an approximate amount of swap space that would be required if the process were to dirty all writable pages and then be swapped out. This number is very rough!)
- 132 : RSS (resident set size, the non-swapped physical memory that a task has used)
- : TTY
- A : STAT (with Process State Code)
- 05:03:47 : STIME (sleep - time)
- 1:23 : TIME (using-cpu-time)
- test : node name
- 1 : node number
- 103642 : send-time
- 0 : packet-sequence number

Figure 7. SMF(Standard Message Format)

C. GRISP(GRIS Protocol)

GRISP enables communication between nodes in real time. It has 4 parts. First is the session binding mechanism, second is the session breaking mechanism, third is the protocol for the historical monitoring and last is the protocol for the real time monitoring.

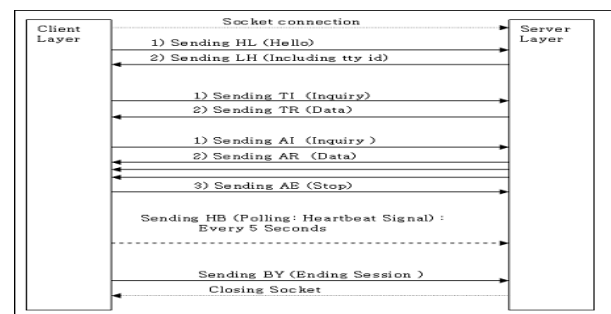


Figure 8. GRISP(GRIS Protocol)

The session binding mechanism :

Client send HL(Hello) to the server, Server reply LH to client. There are established session binding.

At this time, Server gives a sequence number (ttyid) to the client. And Client connection information is managed using ttyid.

The session breaking mechanism:

Client send a BY(bye) to the server. at this time, Client and Server close socket.

The protocol for the historical monitoring:

Client send TI(TR inquiry) to the server, Server reply TR(TR Return) to client. This protocol is get to historical monitoring information data.

The protocol for the real time monitoring:

Delivering information which the node status fluctuated at real-time uses AI(Inquiry), AE(Stop) and AR(Data). In order to request data automatically, Client send AI to the Server. Server replies AR to client continuously. AR is a real-time monitoring data packet. When Client sends AE to client, It is stopped.

In order to know destination window for data received, TR and AR packets have a destination identity like a windows handle.

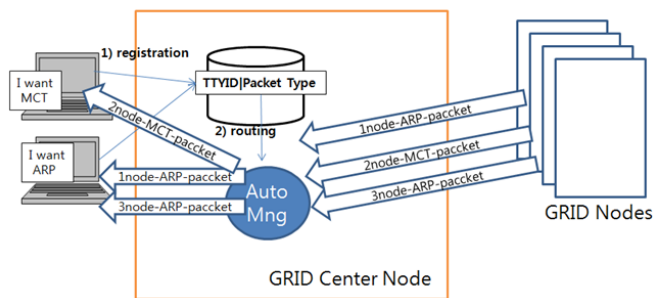


Figure 9. GRISP (GRIS Protocol) flow

The sequence number (ttyid) which is attached server is used for searching destination client, and server send packet to the client. The next step, In client, sends the received packet to a last destination window. in this case, the HWND is used. The right figure is the protocol specification of AI, AE and AR.

The real time monitoring packet has some type. The each type is divided with the auto id number. For examples, The auto id 1 is the CPU utilization value, the auto id 2 is a paging space value, and the auto id 99 is a resource idle value.

In addition, such as Polling, GRISP has a healthy check mechanism. When Client is alive, then send HB (Heart beat) packet to server every 5 seconds. Receiving packet, Server writes an arrival-time in the terminal management table at this server. Session manager in the server checks a time interval between a received-time and a current-time. If a time interval more than the decided position values, then Server closes the session. It is a management that the zombie clients are removed.

D. RRM (Rollback Recording Mechanism) / RCM (Reconnecting Mechanism)

When it occur a network error status, we solve the fault using the RCM (Reconnection Mechanism). Additionally, system is rollback in previously condition. We use the RRM (Rollback Recording Mechanism) for this.

In Our System, communication protocol (or handshaking) is figure . First step is session start. Client send a start packet to server, then server send an OK packet to client.

After when it processes a session starting, it ready condition for client to request. If client send a request packet to server, then server send a response packet for request. When stock trading program is ended, client and server send and receive a close packet. Particularly, server sends a heart-beat packet to client at every 1 second. That is because it maintains a session between server and client. A heart-beat packet has a time-stamp which is wrote send-time by server. Client records a time-stamp with the received packet. The record value is useful data for checking session maintenance. If the time

interval - between current time and arrival time when receive a last heart-beat packet- is bigger than 2 seconds, then we suppose to error status of network. Else then we suppose to normal status. Anyway, the heart-beat time stamp is very useful for checking the network.

When it occur a network error status, we solve the fault using

the RCM (Reconnection Mechanism). First step of RCM is a sensing network error. When it uses a heart-beat time stamp, we know this situation easily. Second, we have to re-connect session between client and server. Third, we send a data for login such as ID, PASSWORD to server. 4th, because we have

to use public certification processing in Korea stock market, send a data for certification processing to server. Last, in order to restore in previously situation, we send a data of input-fields which it requests previously. Finally, reconnection processing is OK.

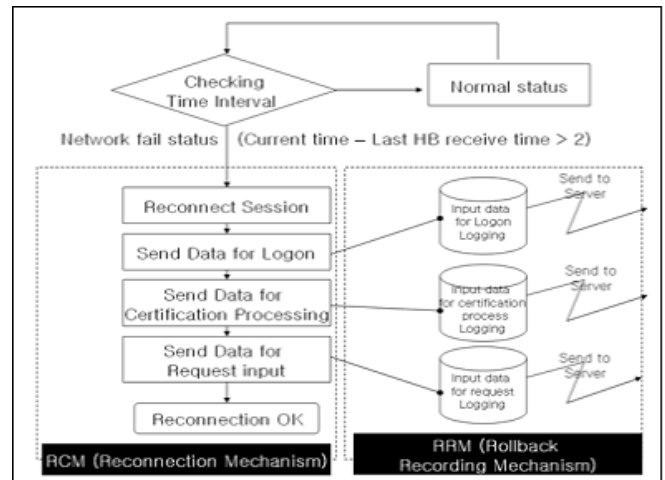


Figure 10. RRM/RCM

5. SYSTEM IMPLEMENTATION

A. System Architecture

Our grid portal solution is a user friendly approach to grid portal. And the design concept for Grid Friend includes availability, generalization, extension, stabilization, supporting of open system and tracing of technical trends.

The system architecture of our grid portal has installed in 3-parts which are grid-nodes, a grid-center node and a user's PC. RT-GRIS is composed of four-parts, RT-GRIS-MDS-node, RT-GRIS-MDS-MSU, RT-GRIS-MDS-Center and RT-GRIS-Portal.

RT-GRIS-MDS-Center is located in the center-server of the grid system which collects, processes and views information distributed over grid nodes. The Center Server is composed of operating system UNIX and programmed by Ansi-C. RT-GRIS-MDS-node exists in each of computational grid nodes. The node agent sends status information to the control tower and performs commands received from the control tower. Most of computational nodes are composed of operating system UNIX and programmed by Ansi-C. RT-GRIS-Portal is installed in user's PC. With this program, a user is able to monitor the status of node, control some tasks and equipment device as well. It is a type of Windows UI programs. RT-GRIS-MDS-MSU is a special equipment unit. This unit gets installed in some moving automobiles. RT-GRIS-MDS-MSU takes charge of sensing GPS as well as temperature and humidity data. It is a type of Windows agent

programs. RT-GRIS-MDS-MSU is a test module that is available to use ubiquitous computing node.

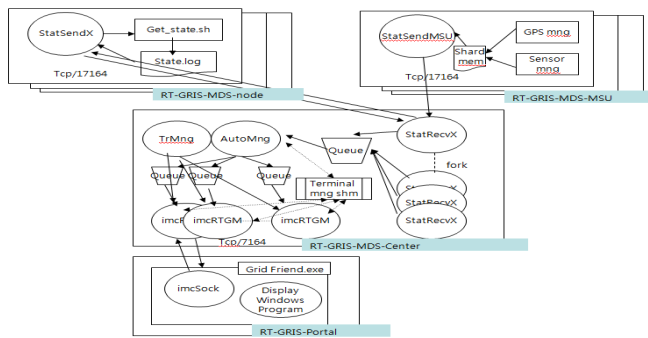


Figure 11. System Architecture

B. Experimental Conditions

We set the RT-GRIS to M-GRID node and U-GRID node. M-Grid node is the distributed computing nodes in Meritz securities company. We operate this system on a real field in order to show a really useful grid portal. Meritz securities has 34 branches nation-wide and one branch has one branch server. Then we have used these servers as grid nodes. U-Grid node composed of 10 Grid nodes in the ubiquitous grid lab, University of Seoul. Because these nodes are professional dedicated grid nodes, we were here important tests.

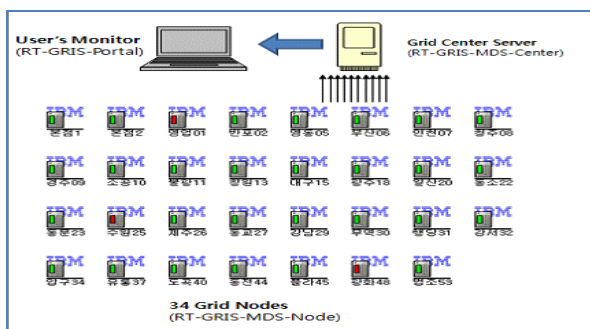


Figure 12. Experimental Conditions

6. SYSTEM EVALUATION

Many world-wide labs are concentrating on research of grid portals and grid solutions nowadays. But, they have some weak points. Because they not have advanced job-scheduling and fast information delivery system, they are very hard to perform efficient CPU Scavenging and handling large number of grid nodes. To solving this reasoning, a grid system must have two features. One is that the job-scheduling should be real-time. Another is that the grid handling process should use fewer resources. For this reason, we propose the following. first is HDA(hierarchical data access), second is SMF(Standard Message Format), third is GRISP(GRIS Protocol), and last is RRM (Rollback Recording Mechanism) / RCM (Reconnecting Mechanism). To prove our proposition, the following were evaluated.

- Interrupt latency time measurements are short.
- CPU usage measurements are more efficient.

TABLE 2. Strong-point of RT-GRIS

Effect	Interrupt latency time measurements are short.	CPU usage measurements are more efficient
HDA	Frequently used data stored on the fastest device available, so data access is fast.	IO-resource will be reduced
SMF	Tag - parsing overhead reduces more than using XML	Tag - parsing overhead reduces more than using XML
GRISP	Event method push service uses, the response speed is fast	Using event-method, unlike polling method of SNMP, use fewer resources
RRM /RCM		auto-reconnection functionality, so that reliable service is possible

The Interrupt latency time, or interrupt response time, is the sum of the cycle-time and the response-time in this system. The cycle-time(C-Time) is a process turnaround time in the grid node. And, the response-time(R-time) is the response time after completed processing between RT-GRIS-MDS-Center and RT-GRIS-Portal.

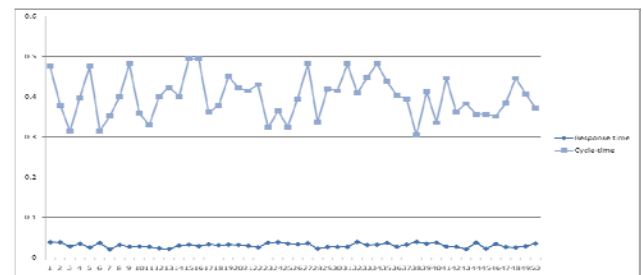


Figure 13. Evaluation Result

TABLE 3. Evaluation Result

Num	R-time	C-time	Num	R-time	C-time
1	0.038817	0.4769	26	0.033812	0.3954
2	0.038537	0.3782	27	0.036272	0.4824
3	0.028121	0.3143	28	0.02253	0.3361
4	0.034829	0.3683	29	0.02665	0.4208
5	0.025564	0.4757	30	0.026726	0.4146
6	0.036781	0.3148	31	0.028819	0.4818
7	0.020491	0.3516	32	0.03553	0.4034
8	0.032009	0.3595	33	0.032054	0.4174
9	0.027473	0.4624	34	0.032608	0.4834
10	0.02807	0.356	35	0.03716	0.4374
11	0.027225	0.3298	36	0.027397	0.4035
12	0.023464	0.401	37	0.032658	0.3944
13	0.021745	0.4242	38	0.036966	0.3035
14	0.030183	0.4014	39	0.035077	0.4132
15	0.032646	0.4953	40	0.037926	0.3353
16	0.026764	0.4947	41	0.027751	0.4466
17	0.033545	0.361	42	0.027394	0.3626
18	0.031456	0.3788	43	0.021274	0.3831
19	0.032697	0.4512	44	0.037962	0.3572
20	0.032056	0.4228	45	0.022321	0.3555
21	0.028778	0.4188	46	0.034672	0.3512
22	0.025804	0.4308	47	0.026724	0.3836
23	0.037884	0.3282	48	0.025778	0.4468
24	0.038806	0.3647	49	0.026548	0.4077
25	0.035306	0.3251	50	0.035505	0.3723

This system shows that the average time within 500 milliseconds. This value is shorter then time-slice value in the context-switch of Linux or Unix kernel.

We measured the cpu-usage of RT-GRIS-MDS-node, RT-GRIS-MDS-Center and RT-GRIS-Portal. As a result,

RT-GRIS was very efficient more than any other world-wide portal. The following figure shows the measure of RT-GRIS (RT-GRIS-MDS-node, RT-GRIS-MDS-Center and RT-GRIS-Portal)

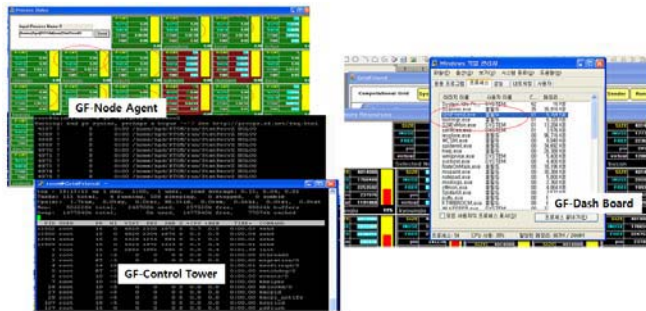


Figure 14. CPU-Usage

The following table details other grid solutions are compared.

TABLE 4. Comparison diagram

	Information Delivery Speed / Job Submit	Supporting a Great Number of Node	Graphic al User Interface	UI-Tool	MDS	Informa tion Col lection
RT-GRIS	Job Scheduling case 3 (Available) event method push service	Thread based light processes, Be able to Parallel extension	Chart/Animation	Self-made	RT-MDS(Self-made)	RT-MDS(Self-made)
MRTG / Ganglia	Have no Job Scheduling Polling method push service	No features	Text-based	C/S (Self-made)	-	snmp-polling
P Grid Portal	Job Scheduling case 1 Interactive system	No features	Text-based	web-sphere	WEB-MDS	snmp-query
G-Monitor	Job Scheduling case 1 Interactive system	No features	Text-based	WEB	WEB-MDS	snmp-polling
Science Portal	Job Scheduling case 1 Interactive system	No features	Text-based	WEB	MDS	snmp-polling
UCLA Grid Portal	Job Scheduling case 1 Interactive system	No features	Text-based	Web sphere	WEB-MDS	snmp-query

This positive result is due to RT_MDS of RT-GRIS. RT-MDS uses a different mechanism by an index to collect information to the MDS of existing grid tool kits. Automatically, RT-MDS send grid resource information of nodes to the center-grid server by the RT-GRIS-MDS-Node agents. The service method of RT-GRIS is an event method push service. But, GIIS's of GT2-MDS send LDAP queries to remote serves and WS MDS Index servers of GT4-MDS use a plug-in-based architecture to support several mechanisms to collect information. The Globus Toolkit supplies plug-in that

support collecting information via polling (resource property queries), subscription/notification, and by program execution.

7. CONCLUSION

In this paper, we have discussed our grid portal, RT-GRIS. Specially, RT-MDS is a very important part of RT-GRIS, an advanced grid portal. RT-GRIS will enable us to solve the problem of the cpu scavenging and the job Scheduling. In addition, we have verified our system. RT-GRIS grid portal is designed specifically for the Seoul R&BD Program. However, it has been proven to be a useful architecture for common grid and ubiquitous environments as well. Of course, there is a need for some changes to be made in this system, but the basic concept is still useful. Finally, the improvement and standardization of the system are currently under progress.

8. ACKNOWLEDGMENT

This study was supported by the Seoul Research and Business Development Program (10561), Smart (Ubiquitous) City Consortium and Seoul Grid Center.

REFERENCES

- [1] <http://www.csse.monash.edu.au/~davida/nimrod/>
- [2] <http://www.cs.wisc.edu/condor/>
- [3] <http://ganglia.info/>
- [4] <http://doesciencegrid.org/projects/GPDK/>
- [5] <http://www.gridisphere.org/>
- [6] Martin Placek and Rajkumar Buyya, "G-Monitor: A Web Portal for Monitoring and Steering Application Execution on Global Grids", 2004
- [7] Kejian Jin, UCLA Grid Portal Implementation ? December 18, 2004.
- [8] Yong Woo Lee, Seoul Grid Portal: A Grid Resource Management System for Seoul Grid Testbed, Lecture Notes in Computer Science , Volume 3251 / 2004.
- [9] Phil Doo Hong, Yong Woo Lee: Web Service for Seoul Grid Testbed. IEEE-CIT 2006: 63
- [10] <http://www.accessgrid.org/>
- [11] E. Elmroth, P. Johansson, B. Kagstrom, and D. Kreßner. A Web Computing Environment for the SLICOT Library. In Proc. The Third NICONET Workshop, pp. 53-61, 2001.
- [12] D. Gannon, G. Fox, M. Pierce, B. Plale, G. von Laszewski, C. Severance, J. Hardin, J. Alameda, M. Thomas, J. Boisseau. Grid Portals: A Scientist's Access Point for Grid Services (DRAFT 1). Sept. 19 2003, GGF working draft.
- [13] Dennis Gannon, Middleware Technology to Support Science Portals: a Gateway to the [18] J. Novotny. The Grid Portal Development Kit. Concurrency Computat.: Pract. Exper., 14, pp. 1129-1144, 2002.
- [14] J. Novotny, M. Russell, and O. Wehrens. GridSphere: a portal framework for building collaborations. Concurrency Computat.: Pract. Exper., 16, pp. 503-513, 2004.