PhoenixG: a Unified Management Framework for Industrial Information Grid

Jianfeng Zhan, Gengpu Liu, Lei Wang, Bibo Tu, Yi Jin, Yang Li, Yan Hao, Xuehai Hong, Dan Meng, Ninghui Sun
Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
jfzhan@ncic.ac.cn

Abstract

Industrial Information Grid (IIG) *is a special kind of Grid system, the users of which exclusively own geographically distributed resources for Web service applications and try to decrease the lowest total cost of ownership while guaranteeing quality of service of applications. In this paper, we classify IIG as an extension to the general Grid problem. Our contributions are three-fold: first, we develop a unified management framework PhoenixG, which supports the collaboration of system administrators geographically distributed at different locations. Second, we propose a self-organizing algorithm that supports the initial establishment, daily management and exception processing of IIG. Finally, we evaluate the performance of system management and analyze the management overhead of PhoenixG.*

1. Introduction

Since the Grid technology matches the need of Grid Problem of "resource sharing and cooperation among virtual organization" [1], it is prevailing. Researchers are trying to specify the common requirements for Grid systems, and develop the unified frameworks. OGSI [2], Globus [3, 4] and ConderG [5] are these typical research efforts. Since more and more organizations prepare to construct and deploy production Grid systems, e.g. Grid 2003 [9], UT Grid [10], the diversity of user requirements appears to be a prominent problem. In this paper, we focus on the requirements of the special industrial users, like Web search engine service providers or federated digital library providers. This kind of users often exclusively own geographically distributed resources for Web service, and try to decrease the total cost of ownership (TCO) while they need guarantees quality of service of applications. This kind of system is obviously different from the existing systems that have been extensively studied, e.g. computational Grid [3, 4, and 5], data Grid [6, 7, and 8] and interaction Grid [11]. So in this paper, we coin a new term *Industrial Information Grid (IIG)* to describe it.

Why we need to put an import emphasis on the requirements of IIG and propose a new unified management framework for it? Firstly, the number of deployed IIG is growing. More and more enterprises have adopted information technology to enhance their business, and their IT information infrastructures have the trend to be geographically distributed. Secondly, the diversity, heterogeneity and geographic distribution of IT infrastructure are also challenging to the system administration, which results in a tremendous budget of IT specialists for management and maintenance. The industrial users need a unified management framework that supports the collaboration of system administrators geographically distributed at different locations, through which they can control the number of IT staffs to the lowest degree. Third, the new management paradigm needs a self-organizing and robust management platform.

Most research literatures focus on the issues of resource sharing and collaboration between virtual organizations [1, 2]. For example, the computational Grid focuses on job management on multi-institutional sites [5]. Data Grid focuses on the architecture for the distributed management and analysis of large scientific datasets [6]. Interaction grid extends the application domain to include interactive graphical sessions [11]. Besides, some basic management facilities have been studied and developed for cluster systems and Grid systems, such as system monitoring and system administration. Though these facilities are important, they are only parts of the whole system. To the best of our knowledge, this is the first paper and the first effort to propose a unified management framework for IIG.

The research contributions of our paper are four-fold: first, for the first time, we classify a special kind of grid system: IIG as an extension to a general Grid problem; second, we propose and develop a unified management framework that supports the collaboration

of system administrator geographically distributed at different locations for IIG, which facilitate the shift of management paradigm; third, we propose a self-organizing algorithm that supports the initial establishment , daily management and exception processing for IIG.
The structure of this paper is as follows: in Section 2, we describe the system scenario. In Section 3, we provide the background of Fire Phoenix Cluster Operating System, on which we build PhoenixG. In Section 4, we introduce the system design issues in detail. Section 5 introduces the experiment and evaluation. Section 6 gives an overview of the related work and Section 7 draws the conclusion and discusses the future work.

2. Scenario of IIG

Let us take an account of a Web search Engine provider like Google [12] or a federated digital library provider like China National Digital Library [13]. As shown in Figure 1, some search engine companies own several sites composed of Cluster systems [14] or high-speed PC LANs which are geographically distributed. Deploying distributed resources on different locations is natural for them, since crawling data from the whole Internet to a single place will incur huge bandwidth consumption. As for federated digital libraries, since each digital library provider has the unique digital library resource and they would like to prefer to promote the resources utilization by cooperation and sharing.
The characterisitics of this kind of system can be concluded as follows:
First, it is composed of geographically distributed resources connected to Internet, including Cluster systems or ordinary PC LANs, and the scales of resources are very large. For example, Google system includes more than ten thousands computer nodes [14].
Second, the ownership of IIG is often exclusive. The sharing is usually on the knowledge level, and users from different organizations can access these knowledge resources freely. In terms of knowledge sharing, IIG is an extension to the general Grid problem.
Third, IIG is often used for running Web service applications, instead of running scientific computing applications. Maintaining the lowest TCO and guaranteeing the quality of service of application are the main targets for industrial users. The system management and maintenance cost is a big part of TCO of IIG, among which the budget of IT specialists is tremendous.
Fourth, to maintain the lowest TOC, the industrial user will choose to fully exploit all their different kind systems, so heterogeneity is the nature of IIG.
Let's further describe the present situation of managing this kind of system. In Section 4 we'll propose new system design.
As shown in Figure 1, the different groups of system administrators are distributed at different location, responsible for the management and maintenance of the sites in their charges. Since all resources are geographically distributed, the management overhead is heavy without an unified management framework that supportes the collaboration of administrators geographically distributed at different locations. This situation needs to be improved.

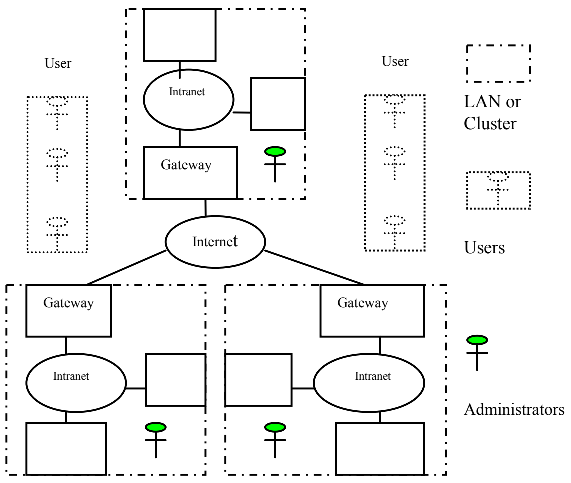


Fig.1. system scenario.

3. Introduction of Phoenix Cluster OS [15]

Phoenix Cluster OS is a complete cluster system software stack that supports both scientific computing and Web application. In Phoenix, the cluster system software stack is divided into three layers: heterogeneous resources, cluster operating system kernel (*Phoenix kernel*) and user environments. Heterogeneous resources are hidden in the bottom layer and invisible to user environments, while the Phoenix kernel provides a stable minimum set of core functions with scalability and fault-tolerant support. Based on the Phoenix kernel, user environments can be easily constructed according to the requirements of users. Phoenix provides different user environments including a system construction tool, a system management tool, a system monitoring tool, a job management system and a business application runtime environment.

4. System design issues

4.1 Grid Units and Grid Center

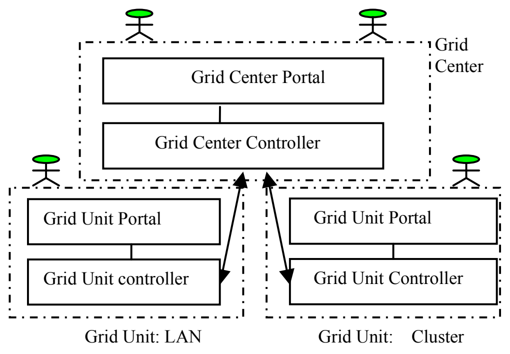


Fig.2. the architecture of IIG.

Figure 2 shows the architecture of IIG. In IIG, we treat each cluster or LAN as a basic management unit, of which we call Grid Unit. If a Grid Unit doesn't join IIG, it is an independent site with autonomic management ability.
As a basic building block, Grid Units negotiate with each other and gradually evolve into IIG. The differences between Grid Unit and Grid Center are distinguished by their different roles. In the self-organizing process of IIG, one Grid Unit will be chosen as Grid Center, and then its controller will raise itself as Grid Center Controller. Though the modules within the Grid Center Controller are similar to the ones within Grid Unit Controller, the data structure will be filled with different information because of different playing roles. The Grid Unit Controller only care about the information from its own Grid Unit, while the Grid Center Controller will care about the information from all Grid Units.
In the following three cases, one Grid Unit can raise itself as a Grid Center: first, in the initial establishment phase of IIG, one of Grid Units will be selected as the Grid Center; second, in the case of the failure of the Grid Center, one of Grid Units will raise itself as the Grid Center; third, in the case of network splitting, some Grid Units can't contact the Grid Center, among which one Grid Unit will be selected to be the new Grid Center;
In the following case, the Grid Center may degrade to a Grid Unit:In the case of the recovery of network splitting, all other Grid Centers will degrade to Grid Units except that one Grid center keeps its role.
As an intelligent agent of a Grid unit, Grid Unit Controller is responsible for Grid Unit's joining and leaving of IIG. Besides, Grid Unit Controller includes several modules: user management, overview information and integrated management. The user management module of Grid Unit Controller is responsible for creating and deleting of accounts of Grid Unit administrators and Grid Center administrators. The overview information module collects the overview information of the Grid Units, and the integrated management module provides the exported management tool of user environments.
The Grid Unit Web portal is the GUI of the Grid Unit system, through which system administrators can manage the resources. The Grid Center Web Portal is the GUI of Grid Center system, through which the system administrators can manage the whole resources of IIG.

4.2 The self-organizing algorithm of IIG

In this section, we introduce the self-organizing algorithm of IIG. The algorithm is responsible for the initial establishment and daily management of IIG, dealing with the issue of new Grid Unit's joining, the failure of Grid Center and merging of several Grid Centers in case of the recovery of network splitting.
For each Grid Unit, the algorithm is as follows:
(1) The system administrator finishes the routine tasks such as configuration, deployment and booting of Phoenix OS [15]. The Grid Unit Controller (GUC) sets its state as *booted*, and subscribes a *well-known multicast address C*.
(2) The GUC sends the *message of joining Grid Center* that contains its own *unique ID* to the *multicast address C*.
(3) The GUC waits for a period of *T1*. Before the timeout, if the GUC receives the *message of Grid Center's address information* from the *multicast address C* ， and then continues; else if the GUC receives *other message of joining Grid Center* or *reselecting Grid Center* that contains the unique *ID* value greater than its own one, then loops (3); else it will promote itself as the Grid Center after the timeout, and subscribes a *multicast address L that indicates its new role of Grid Center*. Finally, it set its state as *center,* and then goes to (8).
(4) The GUC establish the reliable connection to Grid Center Controller (GCC) the SOAP interface. If OK, the GUC sets its state as *joined*, sends its overview information to GCC, then goes to (6); else continues.
(5) The GUC sends the *message of reselecting Grid Center* containing its own *unique ID* to the *multicast address C*, and then goes to (3).
(6) The GUC sets a timer *T2*[1]. If the timer timeouts, it sends the message of *querying grid center* that

[1] This step takes an account of the possible failure of Grid Center.

contains its own ID to the *multicast address C*, and then continues.

(7) The GUC waits for a period of *T3*. If the GUC receives the *message of grid center's address information* from the *multicast address C* before the timeout, the GUC goes to (6); else it goes to (5).

For Grid Center, it can be described as follows:

(8) If the Grid Center Controller (GCC) receives the *message of joining Grid Center, reselecting Grid Center or querying Grid Center*, it responds with the *message of Grid Center's address information.*

(9) The GCC sets a timer *T4*. After the timeout, it sends *the messages of merging grid center* to the *multicast address L.*

(10) The GCC waits for a period of *T5*. If the GCC receives other *message of merging grid center that* contains the ID value greater than its own one after the timeout*,* it unsubscribes the *multicast addresses L,* and goes to (2)*;* else it goes to (9).

Figure 3 shows the transition diagram of self-organizing algorithm described. The number in boxes indicates each step of the algorithm. For a robust IIG, we have considered the algorithm from several aspects: the initial establishment, the joining of Grid Unit, system running, failure of Grid Center, and merging of Grid Centers.

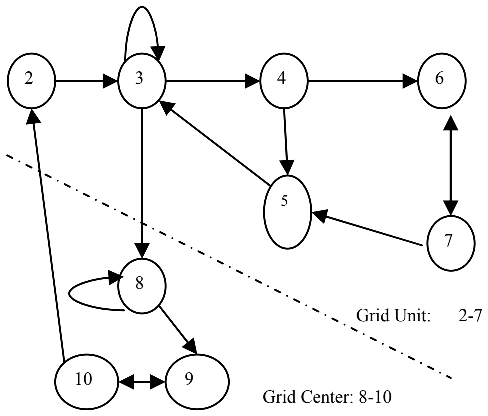


Fig.3. the state transition diagram of self-organization process.

4.3 Unified component Framework with the Support of New Management Paradigm

As shown in Figure 4, we propose a unified component framework that supports the new management paradigm.

In our implementation, for system monitoring, each detector on each node will submit its monitoring information to the data bulletin service (partition service), which summarizes the overview information of each partition. The detail is as follows:

(1) If the scale is large, a Grid Unit will be divided into several partitions, and each one has its own partition services. For system monitoring, there is one Data bulletin Service as the partition service that provides the interfaces for detectors exporting their information and provides in-memory formatted data for consumers' querying under some constraints.

(2) The partition service directly calls the API of each node services on each node, which is the detector that monitors the resource usage and application-specific status. In each partition, all the detectors will export their information to its affiliated data bulletin service. GUC (Grid Unit Controller) won't directly interact with each node service.

(3) Through calling the interfaces of all data bulletin services, GUC can get all interested information of the cluster system.

(4) Once the state of a Grid Unit is set as *joined*, the GUC will submit its overview information to GCC (Grid Center Controller). The GUC also accepts the registration of events that GCC feels interested in. Once these events are produced, GUC will report them to GCC immediately.

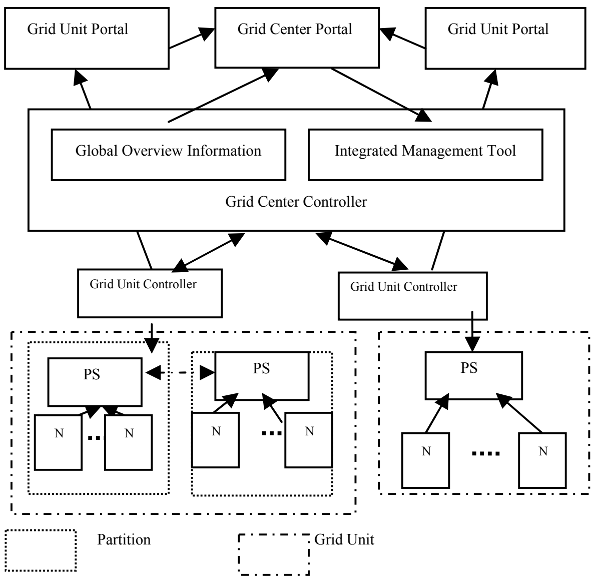


Fig.4. the component frameworks of IIG. PS stands for Partition Service; N stands for Node Service.

(5) The Grid Center Web portal provides the overview information for administrators of Grid Center. If he

wants to delve into the details, he can click the link to jump to the Grid Unit Web Portal. If the administrators of Grid Center find some abnormal things, for example many zombie processes in some nodes, he can use the integrated management tool to clean them.
With this new management paradigm, the distribution of labor is very easy. The simplest situation is that the most of administrators can gather at one place, e.g. Grid Center, and the rest are distributed at different locations of Grid Units. In Section 5.3, we'll evaluate the advantage of this management paradigm over other management paradigms.

5. Experiments and evaluation

5.1 The emulated testbed

As shown in Figure 5, the test bed is as follows:

- Cluster A: 8 X86-64 nodes (AMD Opteron 2G Memory), and the operating system is Turbo Linux 8 (kernel 2.4.21).
- Cluster B: a heterogeneous cluster, with 6 linux64 nodes, and 2 solaris9 nodes.
- LAN A: Heterogeneous nodes, with 4 WinXP workstation and 3 Linux Nodes (Fedora Core 4).
- LAN B: Heterogeneous nodes, with 4 WinXP workstation and 3 Linux Nodes (Fedora Core 4).
- There are two switches: WS-C3750G-24T-S 100M switch and Foundry SuperX GB switch

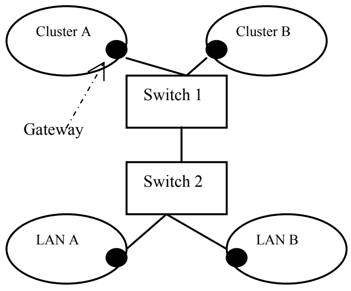
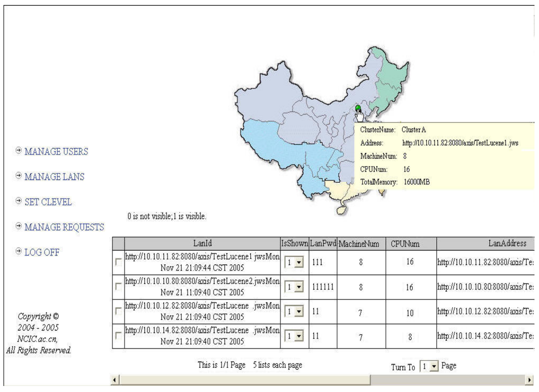


Fig.5. the overview of testbed.

In our experiments, Cluster A, Cluster B, LAN A and LAN B are deployed as an independent Grid Unit with each Grid Unit Controller deployed on the gateways. Figure 6 shows the snapshot of the Grid Center Web portal that has three joined Grid Units.



ClusterName: Cluster A
Address: http://10.10.11.82:8080/axis/TestLucene1.jws
MachineNum: 8
CPUNum: 16
TotalMemory: 16000MB

0 is not visible;1 is visible.

	LanId	IsShown	LanPwd	MachineNum	CPUNum	LanAddress
☐	http://10.10.11.82:8080/axis/TestLucene1.jwsMon Nov 21 21:09:44 CST 2005	1	111	8	16	http://10.10.11.82:8080/axis/Te
☐	http://10.10.10.80:8080/axis/TestLucene2.jwsMon Nov 21 11:09:40 CST 2005	1	111111	8	16	http://10.10.10.80:8080/axis/Te
☐	http://10.10.12.82:8080/axis/TestLucene .jwsMon Nov 21 21:09:40 CST 2005	1	11	7	10	http://10.10.12.82:8080/axis/Te
☐	http://10.10.14.82:8080/axis/TestLucene .jwsMon Nov 21 21:09:40 CST 2005	1	11	7	8	http://10.10.14.82:8080/axis/Te

This is 1/1 Page 5 lists each page Turn To 1 Page

Copyright © 2004 - 2005 NCIC.ac.cn, All Rights Reserved.

Fig.6. the snapshot of Grid Center Web Portal.

5.2 The performance Evaluation

In this section, we evaluate the performance of the unified component framework.

Firstly, we describe how to shutdown nodes with the new management paradigm. The process can be described as follows:

(1) On the Web portal, system administrator chooses the nodes which he wants to shut down, and calls the interfaces of the integrated management tool module of Grid Center Controller.
(2) According to the location of the chosen nodes, the Grid Center Controller concurrently calls the interfaces of the specific Grid Unit Controllers.
(3) Once each Grid Unit Controller receives the call from the Grid Center Controller, it will decide the partitions that contained the targeted nodes, and concurrently call the process manage services in each the specific partitions, which is shown in Figure 4.
(4) The process manage service requests each command execution agent on each node to execute the "shut down" command.
(5) The command execution agent provides the service of executing a single command on a specific node. It can obtain the exit code from the requested command, and redirect the standard input, output and error. If needed, these execution statuses of the commands will be returned to the PMS.

With the new management paradigm, the system management tasks can be done independently on two different levels. The first level is Grid Unit, and the second level is partitions in each Grid Unit. On two different levels, all management tasks can be done in parallel with each other, so we only need evaluate the execution of a parallel command within one partition.

We choose Cluster A in Figure 5 as the testbed, all the test nodes belong to the same partition of the same Grid Unit, and the standard input, output and error are redirected to the PMS. The command to be executed is '*ps –ef*'. For the implementation of PMS, We introduce multi-threads techniques, and nodes in the same partition are divided into several groups, each of which is in the charge of one thread.
The performance data is shown in Figure 7.

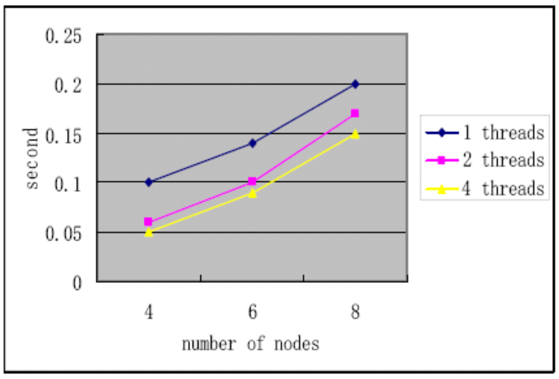


Fig.7. the performance data of concurrently executing 'ps-ef' on several nodes within the same partition.

5.3 The cost of new management paradigm

With the previous management paradigm, as shown in Figure 1, different groups of system administrators are distributed at different location, responsible for management and maintenance of each independent sites. With the new management paradigm supported by PhoenixG, most of the system administrators could gather at the Grid Center, while only small part are distributed at different locations of Grid Units, as shown in Figure 2, which can conveniently controls the numbers of system administrators.
Figure 8 compares the cost of different management paradigms, presuming each independent site is composed of about 100 nodes.
With traditional management paradigm, presuming that each site needs 3 administrators, the OTA shows the number of total administrators varies with the increase of node scale.
With new management paradigm, presuming that each Grid Unit need 1 administrator, and the Grid Center needs 2 administrators per 500 nodes, the NTA1 shows the number of total administrators changes with the nodes scale increases. Compared with OTA, the NTA1 save the half of administrators.
With new management paradigm, presuming that each Grid Unit need 1 administrator, and the Grid Center needs 3 administrators per 500 nodes, the NTA2 shows the number of total administrators changes with the scale increases. Compared with the OTA, the NTA2 almost save the half of administrators.
These analyses show the new management paradigm has an advantage of decreasing total cost of ownership over the traditional management paradigm.

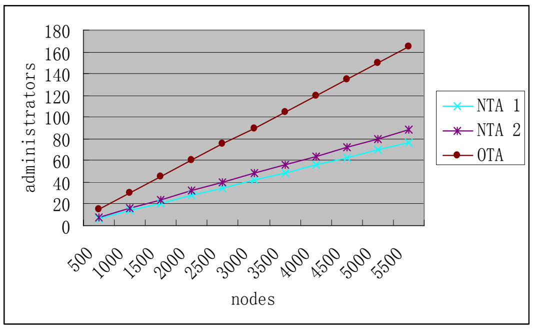


Fig.8. the cost of administration staffs for different management paradigms.

6. Related Work

In the area of computational grid, previous efforts have undergone many infrastructure projects like Globus [3, 4] and Sun Grid Engine [16], which enable users to combine a set of distributed resources into one integrated computing Grid. Condor flocking [17] supports multi-domain computation management by using multiple Condor flocks to exchange load. The major difference between Condor-G [5] and Condor flocking is that Condor-G allows inter-domain operation on remote resources that require authentication, rather than the special- purpose sharing mechanisms of Condor. With the design objective toward an open, extensible architecture that can be expanded, TeraGrid [7] uses a suite of grid and middleware software anchored by the Globus Toolkit. Data Grid [6] focuses on the architecture for the distributed management and analysis of large scientific datasets. The Grid Datafarm [8] architecture is designed for global petascale data-intensive computing that provides a global parallel file system with online petascale storage, scalable I/O bandwidth, and scalable parallel processing. Interactive grids [11] extend the application domain to include interactive graphical sessions. All these work have different focuses from our paper. They focus on computational grid, data grid and interactive grids, while we focus on the requirements of special industrial users, who exclusively own geographically distributed resources for Web service and try to maintain the lowest TCO while guaranteeing quality of service.

There are many peering cycle harvesting system. The goal of resource discovery in CCOF (Cluster Computing on the Fly) [18] is to find idle hosts in a highly dynamic peer-to-peer environment. The project Flock of Condors at Purdue [19] proposes to connect Condor pools using a Pastry overlay. Flock of Condors presents a technique for resource discovery in distributed Condor pools using peer-to-peer mechanisms that are self-organizing, fault-tolerant, scalable, and locality-aware. These projects have some similar aspects to our work. In the self-organizing process of IIG, each Grid Unit maintains a peer-to-peer relationship, but once the whole system is established, the relationship between Grid Center and Grid Unit is master/slave, so IIG does not use the overlay network.
On production grid system, UT Grid [10] supports integrated, high throughput computing at the first stage, and will evaluate and deploy new data Grid technologies at second stage; The Grid2003 Project [9] has deployed a multi-virtual organization, application -driven grid laboratory. Our work has different research angle from the related work, and focuses on the unified management framework that supports the new management paradigm. Our analysis shows it can decrease the cost of the TCO of managing Information Industrial Grid.
Leandor et al [20] investigate the requirements of collaborative applications for groups and discuss how to expand the current Grid and P2P architectures to meet with this need and suggest possible extensions described in "function dispersion architecture". Our work has different research angle and stresses the collaboration work among administrators graphically distributed at different locations.

7. Conclusions and Future Work

To the best of our knowledge, this is the first paper to classify Industrial Information Grid as an extension to the general Grid problem. The users of IIG exclusively own geographically distributed resources for Web service applications, and try to maintain the lowest total cost of ownership while guaranteeing quality of service. For this kind of system, the sharing is usually on the knowledge level, and the common users from different organizations access these knowledge resources freely.
We propose a self-organizing algorithm that takes Grid Unit as a basic building block, supports the initial establishment and daily management of IIG, and deals with several issues, such as the joining of new Grid Unit, the failure of Grid Center and merging of several Grid Centers in case of network splitting.
For IIG, we propose and develop a unified management framework that supports the distribution of administration labor and collaboration of system administrator at different locations, which results in the new management paradigm. Besides, we also compare the management overhead of different management themes. We also propose a unified component framework for system management and system monitoring.
In the near future, we'll deploy this system in the real WAN environment; and evaluate the management overhead on the production systems.

8. Acknowledgments

The work is supported by the National High Technology Research and Development Program of China (863 Program) (No. 2004AA616010) and the 10th Five-year Plan of Key Technologies R&D Program of China (No.2004BA811B09-1).

9. References

[1] Ian Foster etc, The Anatomy of the Grid Enabling Scalable Virtual Organizations, International J. Supercomputer Applications, 15(3), 2001.
[2] Ian Foster etc, The Physiology of the Grid An Open Grid Services Architecture for Distributed Systems Integration, , Open Grid Service Infrastructure WG, Global Grid Forum, June 22, 2002.
[3] I. Foster, C. Kesselman, Globus: A Metacomputing Infrastructure Toolkit. Intl J. Supercomputer Applications, 11(2):115-128, 1997.
[4] Ian Foster, Globus Toolkit Version 4: Software for Service-Oriented Systems, Math & Computer Science Division, Argonne
[5]Frey J, Tannenbaum T etc, Condor-G: a computation management agent for multi-institutional grids. IEEE High Performance Distributed Computing, 2001.
[6] Ann Chervenak etc, The Data Grid: Towards an Architecture for the Distributed Management and Analysis of Large Scientific Datasets, Journal of Network and Computer Applications, 23:187-200, 2001
[7] Charlie Catlett etc, The Philosophy of TeraGrid: Building an Open, Extensible, Distributed TeraScale Facility, Second IEEE International Symposium on Cluster Computing and the Grid. Berlin, Germany:
[8] Osamu Tatebe etc, Grid Datafarm Architecture for Petascale Data Intensive Computing, CCGrid 2002, Berlin, Germany:
[9] Ian Foster etc, The Grid2003 production grid: principles and practice, Proceedings of 13th IEEE

International Symposium on High performance Distributed Computing, 2004. Page(s): 236- 245
[10] Boisseau, J.R etc, UT Grid: a comprehensive campus cyberinfrastructure, 274- 275, Proceedings of 13th IEEE International Symposium on High performance Distributed Computing, 2004.
[11] Vanish Talwar etc, An Environment for Enabling Interactive Grids, Proceedings of the 12th IEEE International Symposium on High Performance Distributed Computing (HPDC'03)
[12] Monika Henzinger, Indexing the Web - A Challenge for Supercomputers, International supercomputer conference, June 19, 2002 in Heidelberg.
[13] China National Digital Library, http://www.nlc.gov.cn
[14] Luiz André Barroso etc, "Web Search for a Planet: The Google Cluster Architecture," IEEE Micro, Vol. 23, No.2, March 2003, pp.22-28.
[15] Jianfeng Zhan etc, Fire Phoenix Cluster Operating System Kernel and its Evaluation, In: IEEE cluster 2005, Boston, USA.
[16] Wolfgang Gentzsch etc, Sun Grid Engine: Towards Creating a Compute Power Grid, CCGrid 2001
[17] Epema, D.H.J., Livny, M., Dantzig, R.v., Evers, X., and Pruyne, J., "A Worldwide Flock of Condors: Load Sharing among Workstation Clusters", Future Generation Computer Systems, 12, 1996.
[18] Dayi Zhou etc, Cluster Computing on the Fly: Resource Discovery in a Cycle Sharing Peer-to-Peer System , 2004 IEEE International Symposium on Cluster Computing and the Grid
[19] A. Butt, R. Zhang, and Y.Hu. A self-organizing flock of Condors. In Proc. SC2003, 2003.
[20] Leandro etc, On Distributed Systems and CSCL, CCGrid 2004