HIERARCHICAL DATA REPLICATION AND SERVICE MONITORING METHODS IN A SCIENTIFIC DATA GRID

***Weizhong Lu*[1,2*]*, Yuanchun Zhou*[1]*, Kaichao Wu*[1]*, Baoping Yan*[1]**

[1]Computer Network Information Center, Chinese Academy of Sciences, Beijing, China
Email: yczhou@sdb.cnic.cn, wkc@sdb.cnic.cn, ybp@cnic.cn
[2]Graduate University of Chinese Academy of Sciences, Beijing, China
**Email: lvweizhong@sdb.cnic.cn*

ABSTRACT

In a grid and distributed computing environment, data replication is an effective way to improve data accessibility and data accessing efficiency. It is also significant in developing a real-time service monitoring system for a Chinese Scientific Data Grid to guarantee the system stability and data availability. Hierarchical data replication and service monitoring methods are proposed in this paper. The hierarchical data replication method divides the network into different domains and replicates data in local domains. The nodes in a local domain are classified into hierarchies to improve data accessibility according to bandwidth and storage memory space. An extensible agent-based prototype of a hierarchical service monitoring system is presented. The status information of services in the Chinese Scientific Data Grid is collected from the grid nodes based on agent technology and then is transformed into real-time operational pictures for management needs. This paper presents frameworks of the hierarchical data replication and service monitoring methods and gives detailed resolutions. Simulation analyses have demonstrated improved data accessing efficiency and verified the effectiveness of the methods at the same time.

Keywords: Data grid, Hierarchical, Data Replication, Service monitoring, Agent technology

1 INTRODUCTION

The Scientific Data Grid (SDG) of the Chinese Academy of Sciences, which is based on scientific databases hosted and supported by different institutes, is a fundamental infrastructure for many data-intensive natural scientific research projects. The scientific databases are major scientific and technological information resources with a total data volume of hundreds of terabytes and consist of geographically distributed and heterogeneous multidisciplinary data resources. The data grid system is growing more and more complex because of heterogeneity and diversity. A large number of computational elements, storage elements, and service providing elements are expected to be connected and integrated into one transparent powerful computing and storage system in the data grid system.

In obtaining better performance and higher reliability of the data grid system, data replication is an effective way to improve data accessibility and data accessing efficiency. At same time, it is increasingly more important to develop an integrated management system, based on a monitoring system to detect the real-time status of resources, devices, and services across grid nodes, to guarantee availability and performance of the overall grid system. Accordingly, hierarchical data replication and service monitoring methods are proposed in this paper. The data replication method selects grid nodes as data replica holders by considering information about data access frequencies, network topology, and link bandwidth. The method groups the entire grid system into different domains and replicates data in local domains. The nodes in local domains are classified into hierarchies to improve data accessibility according to bandwidth and storage memory space. As for the hierarchical service monitoring method, the Simple Network Management Protocol (SNMP) and agent technology are two popular key methodologies to address monitoring issues in large scale distributed environments. Agents distributed in grid elements collect management information and interact with the monitoring server remotely through a SNMP protocol to perform monitoring tasks. If the services or devices being monitored seem abnormal, messages will be sent to the system manager, and the abnormal status will be prominently displayed. This paper presents the framework of a hierarchical data replication and service monitoring methodology and gives detailed procedures.

The remainder of this paper is organized as follows. In section 2, we discuss related works and the motivation for our methods. We then present the hierarchical data replication method and the hierarchical service monitoring method respectively in two separate descriptions because that the two components in our Scientific Data Grid are implemented on different physical machines and have different logical topologies. Accordingly, in section 3, we present a detailed description and simulation analysis of the hierarchical data replication method. A detailed resolution of the hierarchical service monitoring method in the scientific data grid is proposed in section 4. Finally, we conclude this paper in section 5.

2 RELATED WORKS

In a data grid environment, data replication is a very important method of improving data access reliability and performance, and has been widely studied recently. The Globus Toolkit 4.0 (http://www.globus.org/) introduced and implemented a Data Replication Service (DRS) that provides a pull-based replication capability for grid files. The DRS is a higher-level data management service based on the Reliable File Transfer (RFT) Service and the Replica Location Service (RLS). RLS is a simple registry that keeps track of which nodes the replicas exist on. Users or services register files in RLS when the files are created. The Storage Resource Broker (SRB) (Baru, Moore, Rajasekar, & Wan, 1998) is a storage management system designed for data grid environments. SRB is client-server middleware that provides a uniform interface and mechanism to access heterogeneous data resources distributed on multiple hosts and multiple platforms by using logical to physical name mapping. In Lamehamedi, Shentu, Szymanski, and Deelman (2003), a method is proposed for dynamically creating replicas based on a cost estimation model. According to the model, replication decisions are made based on gains of creating a replica against costs of creation and maintenance. Carmen (2002) aimed to achieve global optimization and proposed a technique to maximize the profit and minimize the cost of data resource management. Similarly, Bell, Cameron, Carvajal-Schiaffino, Millar, Stockinger, and Zini (2003) proposed another economy based approach for file replication by means of reverse Vickery auction to predict the cost and benefits. At the same time, the consistency issues in data replication have been studied for several years. Dullmann, Hosckek, Jaen-Martinez, Segal, Samar, Stockinger, and Stockinger (2001) proposed a Grid Consistency Service (GSC), which implements replica synchronization across the grid. Lin and Buyya (2005) introduced a novel approach to data resource broker policies for a replication scheduler under the tiered MONARC data grid structure from a market economy resource management approach. This new approach extended the commodity market model of the market economy to take into account other factors such as server reliability, transfer speeds, link reliability, and service costs. Compared with all the methods mentioned above, this paper presents a novel *simplified* mechanism to implement data replication in a data grid to improve data accessibility and efficiency.

In the area of service monitoring, network management research to address system and network problems in distributed network environments has been done for many years. There are many traditional approaches and products to perform the network management, such as HP OpenView, IBM NetView, and many other tools. SNMP and agents have been implemented by many researchers in different contexts, each one with its own strengths and weaknesses. One approach is GridRM (Baker & Smith, 2002), which consists of a generic monitoring architecture that has been specifically designed for the Grid. It was developed by integrating several technologies and standards such as Java and SQL databases and follows several Open Grid Forum (OGF) (http://www.ogf.org) recommendations for resource management. Other alternatives are MonAlisa (Legrand, Newman, et al., 2004) and Ganglia (Massie, Chun, & Culler, 2004), which are well known systems for monitoring computational resources and clusters, respectively. On the one hand, MonAlisa requires complementary systems to be helpful in the Grid management area. On the other hand, Ganglia is not oriented to large-scale Grids, being oriented to clusters and high capacity resources. Also worth mentioning is JAMM (Java Agents for Monitoring and Management), a distributed set of sensors that collect and publish monitoring information of computational resources. The hierarchical service monitoring method proposed in this paper, however, can be more competitive due to its lower resource consumption and its greater availability.

3 THE HIERARCHICAL DATA REPLICATION METHOD

In this section, we introduce the detailed framework, system model, and basic algorithm of the hierarchical data replication method and then present simulation results and analysis of the method.

3.1 The Framework

The idea of this new hierarchical data replication mechanism is motivated by the assumptions that group of nodes that can be functionally divided into logical domains are almost located in same network regions. In many cases, a logical domain may usually be correspondent to a network region in the data grid environment. A logical domain is a group of nodes in grid environment that specifically have a common application background. Each logical domain has a unique logical domain name, and all the nodes in a logical domain have the same logical domain names. The replicas accessions are confined to the local domain to get high performance. If the required replica is in the same network domain, the job is able to fetch the replica quickly because of the broader bandwidth that can be provided within a local network domain. If the required replica is located at a node in another network domain, then a replica will be created in a node of the local domain. The framework of the hierarchical data replication model is shown in Figure 1.

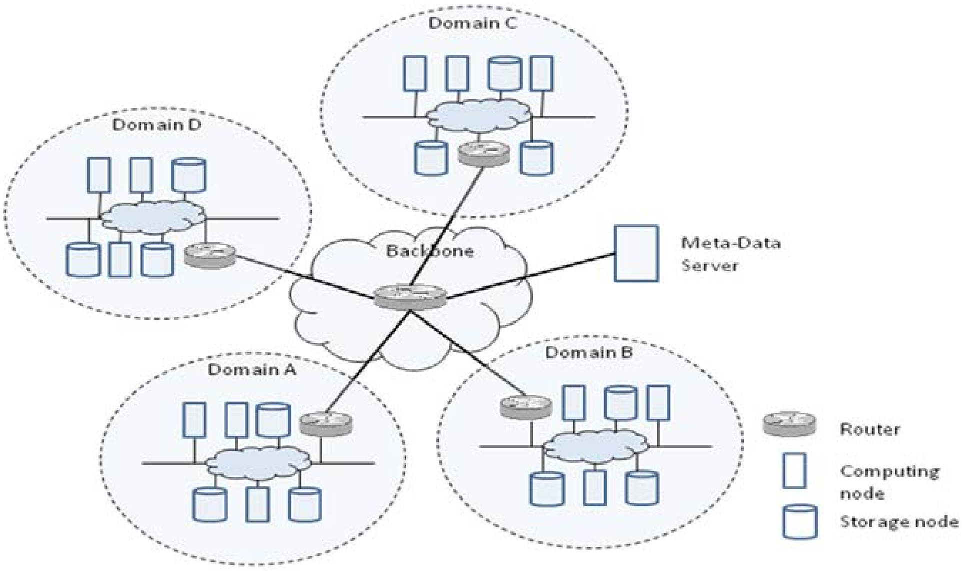

Figure 1. The hierarchical data replication system in scientific data grid

In the hierarchical data replication method, the grid environment is divided into several logical domains, and each domain is logically composed of computing nodes, storage nodes, and router nodes. The files shared in the grid are replicated and stored in the storage nodes, and jobs are submitted to the computing nodes by applications. The data replication is performed in case of high access frequency to a file in another node located in a different domain in the grid. A metadata server is adopted to maintain information of data mapping from logical name to physical storage location. In this method, the nodes with high performance in data transportation and data storage are fully utilized.

3.2 The system model

In our hierarchical data replication method, we make the following assumptions:

1. The set of all nodes in this method is denoted by $N = \{N_1, N_2, \cdots, N_n\}$, where n is the total number of nodes in grid network and N_j $(1 \leqslant j \leqslant n)$ denotes the unique node identifier.
2. The set of all logical domains are denoted by $D = \{D_1, D_2, \cdots, D_k\}$, where k is the total number of domains in the grid system and D_j $(1 \leqslant j \leqslant k)$ denotes the unique logical domain identifier.
3. The set of all data files are denoted by $F = \{F_1, F_2, \cdots, F_m\}$, where m is the total number of data files in the system and F_j $(1 \leqslant j \leqslant m)$ denotes the unique data file identifier. Additionally, for simplicity, all data files are of the same size, and each holds by a particular node initially.

4. Each node has different memory space for storing replicas, excluding the space for original data files.
5. In this method, we assume that the data files are not updated and cannot be overwritten by nodes.
6. Each node can measure the links bandwidth and stability connected to it and registers messages to the metadata server.

In the hierarchical data replication method, the nodes of a domain are classified into three groups A, B, and C according to three properties of the nodes: (1) availability, which indicates the availability of a node to others, (2) bandwidth and stability of links connecting to other nodes, which signify speed and efficiency of data transportation, and (3) memory space, which indicates the number of replicas a node can hold. From the classification, we know that A nodes denote those with higher link bandwidth and stability than B and C nodes and with sufficient memory space to store replicas. In this method, we select A nodes to be the backbone nodes and to store more replicas than B nodes and C nodes.

3.3 The basic algorithm

The hierarchical data replication method tries to replicate files that are likely to be used frequently within the network domain in the near future. The logical metadata server allows the storage nodes to register file replication messages, which can be used by computing nodes to find a file replica. It also provides mappings from logical names of files to the storage system locations of one or more replicas of these objects. The other key function of the metadata server is to evaluate the access frequency of files and to determine when to create a replica and where to place it.

The specific description of the method is as follows:

1. Initially, files are distributed across the nodes in the data grid, and the metadata server keeps track of mappings from the logical names of the files to their physical locations.
2. The storage nodes of a domain in the data grid are classified into hierarchies according to their link bandwidth and storage memory space. Also the classification information of every storage node within a domain is kept and maintained by the metadata server.
3. The access frequency of each file within a domain is calculated periodically by the metadata server, which can monitor and collect the file access information. Thus, a file with higher access frequency will have relative priority to be replicated in A class storage nodes.
4. When a computing node accesses a file in the data grid, it tries to make certain where the file and its replicas are located. If the required file or the replica is located in any storage node in the local domain, the requiring node fetches the data file from the storage node.
5. Otherwise, the requiring node fetches the data file from its original location according to mapping information on the metadata server. Next, the metadata server makes certain where to replicate the required file in the local domain.
6. In the next step, the metadata server recalculates and sorts the file access frequency in the local domain and reclassifies storage nodes according to their current bandwidth and memory space statement. Consequently, the data file is replicated in a storage node with A, B, or C class accordingly. As a result, the files with high access frequency reside in the storage nodes with high performance.
7. If there is not enough space to hold a new replica, files with low access frequency in the nodes with lower classification are deleted to make room for new replicas.

From the previous discussion about the hierarchical data replication method, we know that data replicas are shared among nodes within a domain and make full use of class A nodes with high performance and stability. Consequently, the data accessibility in this method is expected to be higher.

3.4 Simulation and Analysis

3.4.1 The methodology

We evaluate the hierarchical data replication method by using simulation tool OptorSim developed in JAVA, which assumes that a data grid consists of several sites, each of which may provide computational and data storage resources for submitted jobs. In OptorSim, computing elements (CEs) run jobs, which use data in files stored on storage elements (SEs), and a single resource broker controls the scheduling of jobs to CEs. Sites without SEs or CEs act as network nodes or routers. The movement of data associated with jobs between sites is performed by a component called the Replica Manager. Within the Replica Manager, the decision to create or delete replicas is controlled by the replication algorithm. In the data grid environment, each site can contain several storage or computing elements. OptorSim takes a grid configuration and a replication algorithm as input and then runs a number of grid jobs using the given configuration. It also allows a user to visualize the performance of the algorithm.

In this simulation, we define two evaluation metrics to be examined. First, we define the mean execution time of jobs, which is the metric of performance. It is equal to the value of the execution time of total jobs submitted to the system divided by the number of jobs. Next, the effective network usage is defined to describe the efficiency of file replication. This is effectively the ratio of files transferred to the files requested. Therefore, a low value indicates that the replication mechanism is successful at putting files in the right place. The specifics of these definitions are expressed as the following equations:

$$Mean\_Job\_Time = \frac{Total\_job\_time}{number\_of\_jobs} \tag{1}$$

$$Effective\_Network\_Usage = \frac{N_{remote\_file\_accesses} + N_{file\_replications}}{N_{remote\_file\_accesses} + N_{local\_file\_accesses}} \tag{2}$$

Table 1. The parameters of the experiment

Parameters	Values
Number of jobs	100
Number of files	56
Size of file	1GB
Memory space in SE	50GB
Memory space in CE	10GB
Initial file distribution	Random
Bandwidth between domains	100Mbps
Bandwidth to A class nodes	1000Mbps
Bandwidth to B class nodes	500 Mbps
Bandwidth to C class nodes	100Mbps

We perform the simulation with the data grid topology shown in Figure 1. We assume there are four domains in the grid, and each domain has six nodes, which in turn have 3 CEs and 3 SEs respectively. In the simulation, 100 jobs are submitted to the grid, each of which accesses 12 to 16 data files. Initially, all 56 files are distributed randomly across the grid. The memory space in SE and CE are 50GB and 10GB respectively. The bandwidths between domains are set to be 1000Mbps. Detailed parameters of the simulation are presented in Table 1.

3.4.2 Results and analysis

In the simulation, we compared the result of four replication algorithms with each other: Simple algorithm, Lfu algorithm, EcoModel algorithm, and Hierarchical Replication Mechanism (HRM) algorithm. The Simple algorithm means that no replication occurs during the course of the simulation. The Lfu algorithm indicates that replicas are always created if the node does not hold the required file, deleting the least frequently accessed file if there is not enough space for replication. In the EcoModel algorithm, replicas are created when the eco-model of the algorithm says yes, possibly deleting the least valuable file according to the binominal prediction function. In contrast, in our HRM method, a replica is created when the required file is not located in the node issuing queries or in any A class nodes of local domain that is close to the node issuing queries with broad bandwidth.

Table 2. The results of the experiment

	Simple	Lfu	EcoModel	Hrm
Mean job time of all jobs (s)	3230	1050	577	728
Effective network usage	90.8%	27.5%	22.7%	23.8%

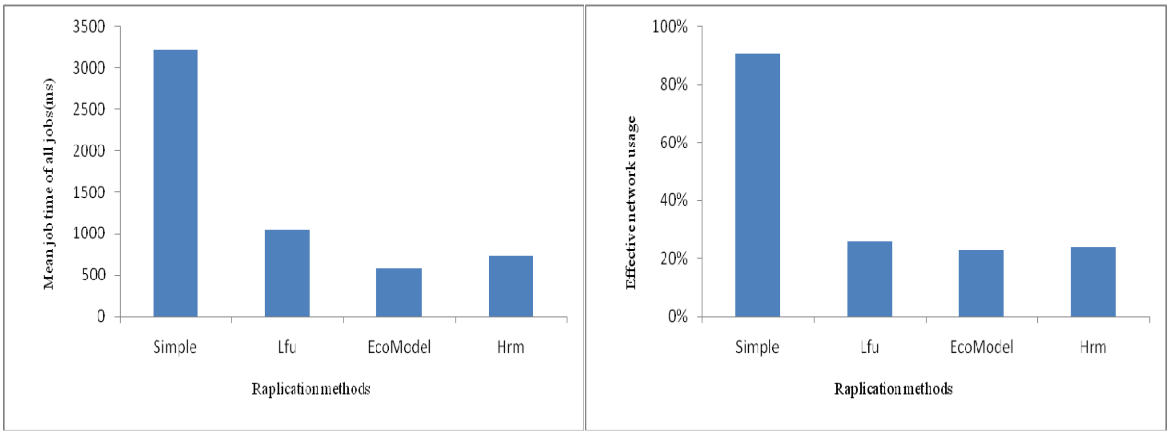

Figure 2. The mean time of job and effective network usage

The simulation results are shown in Figure 2 and Table 2. In the simulation, it is appeared that the EcoModel algorithm and the hierarchical data replication method show almost the same mean job execution time and effective network usage. The hierarchical data replication method has a little lower performance than the EcoModel algorithm because in it the replicas are shared in a domain. Therefore, the hierarchical data replication method may promote increased traffic in the domain to transfer replicas between nodes. It also may increase the mean job execution time at the same time. The Simple algorithm requires the greatest mean job execution time and effective network usage among the four algorithms because no replication occurs. The Lfu algorithm has a medium performance score as the replicas are always created by the requiring node even if the neighboring node holds one. From the above, the hierarchical data replication gets an acceptable performance score for replication in a data grid simulation. To some extent, it can improve data accessibility and performance in a data grid environment.

4 THE HIERARCHICAL SERVICE MONITORING METHOD

In this section, we present the framework and detailed implementation of the hierarchical service monitoring method in the Chinese Scientific Data Grid.

4.1 The Framework

As shown in Figure 3, the Scientific Data Grid of the Chinese Academy of Sciences is a large scale data sharing system geographically spanning across all the institutes of CAS. Each institute has its own huge data storage environment to support the specific research work, and the storage environments of all institutes are connected by a backbone network to be integrated into one huge scientific data grid. The data service of each institute is provided through a service interface (SI) at each institute to interact with other institutes. The service interface plays an important role in data access and communications among institutes. Behind the service interface are some specific databases storing relevant discipline-specific research data that are the core of the data grid. All research data can be accessed both locally in the institute and remotely from other institutes through the SI. Furthermore, a portal to all the CAS users of the data grid has been developed to provide a web based approach.

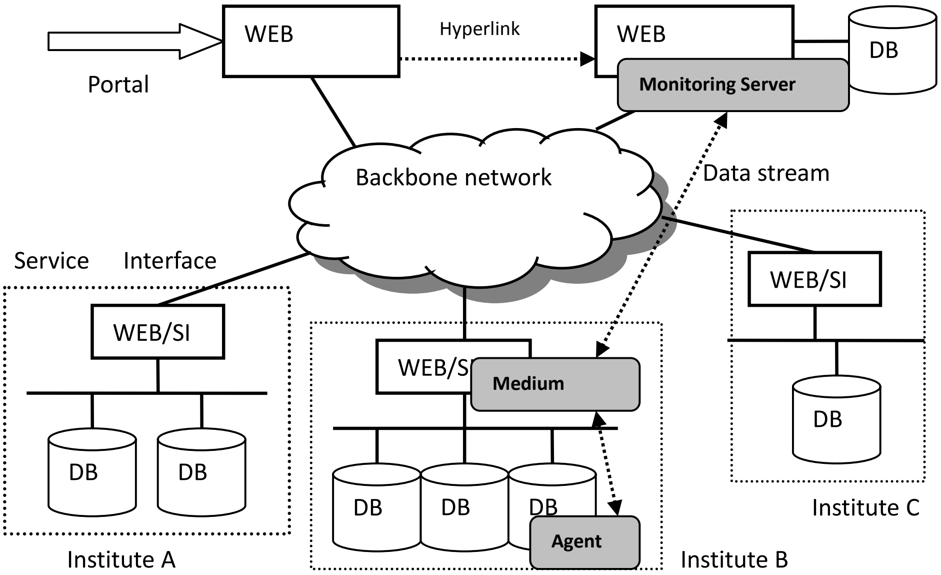

Figure 3. The hierarchical service monitoring system in scientific data grid

The monitoring system to be deployed in the Scientific Data Grid logically consists of three components: monitoring server, medium agents, and institute agents. The architecture of the hierarchical service monitoring is described in Figure 3. The monitoring server is in charge of monitoring information collection, storage, display, and other specific processions. The monitoring information collected by distributed agents and medium agents is stored in a specific database called the monitoring information database and can be accessed or displayed by means of a web service provided by the monitoring server from the grid portal. The agents distributed in the database servers of institutes are key components to monitor the running status and statistics of database servers. The monitoring information obtained by the agents is then used by the medium agent of the same institute to monitor information about the SI and all the databases behind itself and responding to requirements issued by the monitoring server. The medium agents just act as proxies to transmit the information requirements from the monitoring server forward to institute agents residing on the database servers behind the SI. Collecting monitoring information about database servers in the Scientific Data Grid is actually performed by the institute agents.

4.2 The Implementation

In this section, we propose a detailed implementation of a prototype of the monitoring system to be deployed in our Scientific Data Grid. The key aspects and technologies adopted in developing the monitoring system are described.

4.2.1 The monitoring protocol

The Simple Network Management Protocol was introduced by the Internet Engineering Task Force (IETF) to meet the growing need for a standard for managing Internet Protocol (IP) devices, including routers, switches, servers, printers, and uninterruptable power supplies. The protocol defines the framework of network management and the structure of packets. The SNMP management system contains four components: manager, agents, management protocol, and management information bases (MIBs). The core of SNMP is a simple set of operations (get, set, response, and trap) that gives an administrator the ability to monitor and control the state of SNMP-based devices. It is fairly simple to use the SNMP to monitor the health of routers, servers, and other pieces of network hardware, and we can also use it to monitor and control network devices if problems arise. Any device that runs agent software allowing the retrieval of the SNMP information can be managed, including not only physical devices but also software, such as web servers and databases. Accordingly, the SNMP-based network monitoring is another key aspect of our network management.

4.2.2 The platform

When we began to develop a monitoring system, the most important step of all was to implement the SNMP protocol in the specific environment. Fortunately, a software project called Net-SNMP, which was historically developed by the U.S. Carnegie Mellon University, was applicable. Net-SNMP is a suite of applications used to implement SNMP v1, SNMP v2c and SNMP v3 using both IPv4 and IPv6. The suite includes command-line applications, graphical MIB browser, a SNMP daemon application for receiving SNMP instructions, and a library for developing new SNMP applications using both C and PERL APIs. As to applicable system environment, Net-SNMP is available for many Unix and Unix-like operating systems as well as Microsoft Windows. Based on Net-SNMP, agents conforming to the SNMP protocol can be deployed in the nodes to be monitored in a distributed environment. At the same time, the agents also can be extended by adding MIBs and related procedures to provide more information to managers. Thus, coordinated with other network management tools, both open source or commercial products, Net-SNMP is a popular, well developed platform for implementing a self-defined SNMP management system.

4.2.3 The MIBs

Managed objects are accessed via a virtual information store, termed the Management Information Base (MIB). Objects in the MIB are defined using Abstract Syntax Notation One (ASN.1). For ease of description, in particular, each object has a name, syntax, and an encoding. As described in Bell, Cameron, Carvajal-Schiaffino, Millar, Stockinger, and Zini (2003), the name is an object identifier, an administratively assigned name that specifies an object type. The object type together with an object instance serves to identify uniquely a specific instantiation of the object. For human convenience, we often use a textual string, termed the OBJECT DESCRIPTOR, to refer to the object type. The syntax of an object type defines the abstract data structure corresponding to that object type. The encoding of an object type is simply how the object type is represented using the object type's syntax. In the management system, an MIB is a database managed by a SNMP agent controlling the managed objects according to the Structure of Management Information (SMI) rules. It has a tree structure similar to that used in the Domain Name System (DNS), as shown in Figure 4. Each node has a readable string reference and each object could be identified by a readable string or by its OBJECT IDENTIFIER (OID).

Many applications and tools have already implemented standard MIBs that contain a number of standard objects. If an SNMP agent must manage its own objects, these objects belong to a private MIB. According to the mechanism of an MIB, the objects must be located in the branch *iso.org.dod.internet.private.enterprises* shown in Figure 4. One who wants to extend the MIB of an SNMP agent must apply for a private enterprise code to the IANA authority (Internet Assigned Numbers Authority). For example, the number assigned to the Computer Network Information Center of Chinese Academy of Sciences is 8728.

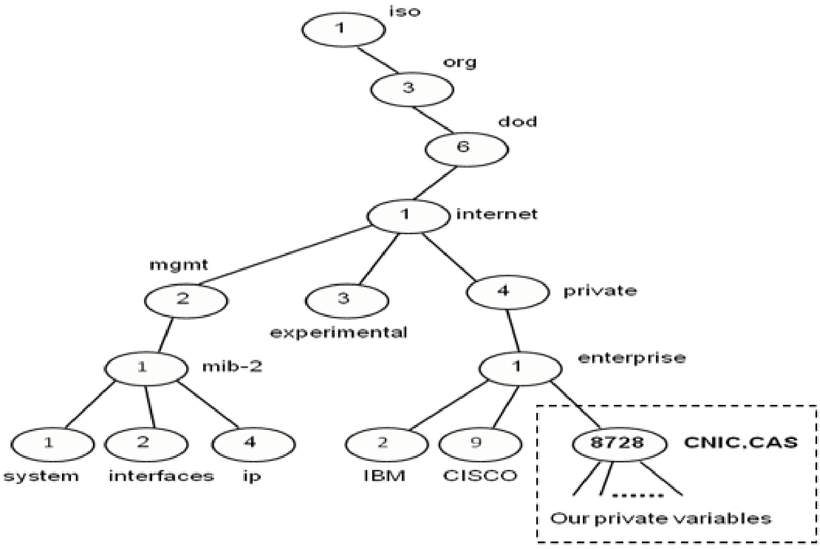

Figure 4. The MIB tree in our service monitoring system

In our hierarchical service monitoring system, the extended MIBs containing self-defined monitoring objects are constructed. As shown in Figure 4, the locations of our self-defined objects in the MIB tree are confined to the subtree 1.3.6.1.4.1.8728 (iso.org.dod.internet.private.enterprises.CNIC). To implement our self-defined MIBs, we followed three steps based on the Net-SNMP platform. The first step is constructing MIB files according to the syntax of rules described in Bell et al. (2003) and other RFCs (Request for Comments). Then, by using tools such as mb2c, we can translate a self-made MIB file, which is presented in a text file, into C procedures. Finally, the last and the key step is coding in C programming language to realize the function and to obtain from the system the values of the monitored objects to respond to the query issued by the manager. Consequently, extended MIB can be compiled into the agent application to be loaded into the system to respond to requirements of the self-defined objects.

4.2.4 The agents

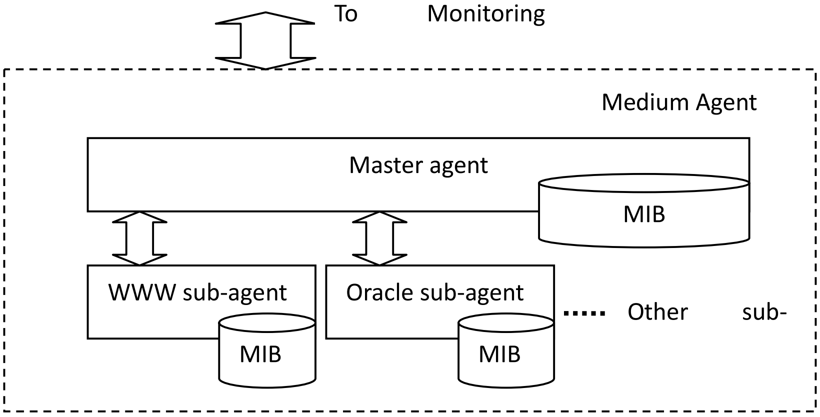

Figure 5. The relation between the master agent and sub-agents

In the hierarchical service monitoring system, agents are divided into two hierarchies, named medium agents and institute agents respectively, as shown in Figure 3. The agents, which reside in the data server of institutes, are single agents responsible for collecting monitoring information of a data server. These agents are database specific agents provided by the database server system. If a SNMP based agent is not integrated into the data server product by the provider, a self-defined MIB and agent can be developed by means of extending MIBs of the Net-SNMP platform.

The key task includes constructing MIB files that define the objects to be monitored and implementing the procedures to determine the value of the requested object. Obtaining the value of the monitored object of the data server system can be fulfilled by way of operating system's API or other database system state information.
The medium agents located in the service interface of institutes are proxy agents to some extent. In every institute there is only one medium agent responsible for responding to the query of the monitoring server. As shown in Figure 5, the medium agent consists of one master agent, one web sub-agent, one database sub-agent, and other sub-agents if required. The master agent responds to the status query of the service interface and forwards the status query of the web and database to relative sub-agents. The database sub-agent can be a proxy forwarding to the actual database server running in another node. By integrating the master agent and the sub-agents, the medium agent of the monitoring service system can respond to requests from the monitoring server to provide the running status information of the service interface, the web service, and the database service.

4.2.5 The monitoring server

The top hierarchy of the service monitoring system is the monitoring server, which is in charge of managing all information of monitored objects and of providing a query interface to administrators at the same time. The monitoring server applications are developed in C based on the Net-SNMP programming API that implements the SNMP operations and supports the SNMP based server and agent applications. The monitoring server has its own database to store the monitoring information, which is required by the "get" operation and receives "trap" information periodically. To facilitate administration, the monitoring system implements a web based interface by which the monitoring system provides service information querying services. Administrators can access the monitoring system information either by logging onto the monitoring sever or through the portal of the Scientific Data Grid. The results will be displayed graphically as needed to provide a more a friendly interface.

5 CONCLUSIONS AND FUTURE WORKS

The data replication and performance monitoring in the Chinese Scientific Data Grid is an important and necessary portion, as our data grid system progressively becomes more complicated and powerful. In this paper, we present a novel data replication method, which takes the characteristics of nodes and network topology into consideration. The hierarchical data replication method makes full use of the bandwidth and memory space of storage nodes in local domains. It also has the merits of self-organization and load balancing. At the same time, we also present a novel service monitoring system based on a multi-agent technology in a data grid environment to realize real-time service status collection and monitoring, by adopting popular techniques of multi-agent and SNMP. Finally, we present a detailed implementation of an extensible agent-based prototype of the monitoring system and simulation analyses to verify the feasibility and the performance of the proposed data replication method.

In the future, we plan to develop approaches to avoid single point failure of the metadata server and to improve its performance. Furthermore, we also plan to develop a prototype system to verify its effectiveness. Due to the vulnerability of the SNMP protocol, which is based on IP protocol adopted in the monitoring system, we are planning to address security issues involved in the communication between agents and server. We are also preparing to deploy the prototype of the monitoring system in the SDG of Chinese Academy of Sciences.

6 ACKNOWLEDGEMENT

This research is partially supported by the Informationzation Constructing Projects of the Chinese Academy of Sciences (CAS) (INF105-SDB).

7 REFERENCES

Apache Web Server 2.0. Retrieved from the WWW, February 19, 2009: http://www.apache.org

Baker, M. & Smith, G. (2002) GridRM: A Resource Monitoring Architecture for the Grid. In: *3rd International*

Workshop on Grid Computing. Baltimore, Maryland, USA.

Baru, C., Moore, R., Rajasekar, A., & Wan, M. (1998) The SDSC Storage Resource Broker. *Proceedings of the 1998 Conference of the Centre for Advanced Studies on Collaborative Research.*

Bell, W., Cameron, D., Carvajal-Schiaffino, R., Millar, A., Stockinger, K., & Zini, F. (2003) Evaluation of an Economy-Based File Replication Strategy for a Data Grid. *Proceedings of 3rd IEEE International Symposium on Cluster Computing and the Grid, IEEE Computer Society Press.*

Carman, M., Zini, F., Serafini, L., & Stockinger, K. (2002)Towards an Economy-Based Optimisation of File Access and Replication on a Data Grid. *Proceedings of the 1st IEEE/ACM International Conference on Cluster Computing and the Grid, IEEE Computer Society Press.*

Case, J., Harrington, D., Presuhn, R., & Wijnen, B. (2002*) Message Processing and Dispatching for the Simple Network Management Protocol (SNMP).* RFC 3412, SNMP Research, Enterasys Networks, BMC Software, Lucent Technologies.

Dullmann, D., Hosckek, W., Jaen-Martinez, J., Segal, B., Samar, A., Stockinger, H., & Stockinger, K. (2001) Models for Replica Synchronisation and Consistency in a Data Grid. *Proceeding of 10th IEEE International Symposium on High Performance and Distributed Computing.*

GT4: Globus Toolkit 4. Retrieved from the WWW, February 19, 2009: http://www.globus.org/

Harrington, D., Presuhn, R., & Wijnen, B. (2002) *An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks.* RFC 3411, Enterasys Networks, BMC Software, Lucent Technologies.

IANA : The Internet Assigned Numbers Authority. Retrieved from the WWW, February 23, 2009: http://www.iana.org/

JAMM Project (2000) Java Agents for Monitoring and Management. Lawrence Berkeley National Laboratory. Retrieved from the WWW, February 23, 2009: http://acs.lbl.gov/DIDC/JAMM/

Lamehamedi, H., Shentu, Z., Szymanski, B., & Deelman, E. (2003) Simulation of Dynamic Data Replication Strategies in Data Grids. *Proceedings of the 17th International Parallel and Distributed Processing Symposium, , IEEE Computer Society*. Nice, France.

Legrand, I., Newman, H., et al. (2004) MonALISA: An Agent based, Dynamic Service System to Monitor, Control and Optimize Grid based Applications. In: *CHEP 2004*. Interlaken, Switzerland.

Lin, H. & Buyya, R. (2005) *Economy-based Data Replication Broker Policies in Data Grids*. Unpublished thesis, Department of Computer Systems and Software Engineering, The University of Melbourne.

Massie, M., Chun, B., and Culler, D. (2004) The Ganglia Distributed Monitoring System: Design, Implementation and Experience. *Parallel Computing 30(7).*

Net-SNMP 5.1. Retrieved from the WWW, February 19, 2009: http://www.net-snmp.org

Open Grid Forum. Retrieved from the WWW, February 19, 2009: http://www.ogf.org

OptorSim – A replica optimizer simulation. Retrieved from the WWW, February 19, 2009: http://edg-wp2.web.cern.ch/edg-wp2/optimization/optorsim.html.

Rose, M. (1990) *Management Information Base for Network Management of TCP/IP-based internets: MIB-II.* RFC 1158, Performance Systems International.

SMI Network Management Private Enterprise Codes. Retrieved from the WWW, February 19, 2009: http://www.iana.org/assignments/enterprise-numbers

(Article history: Received 25 November 2008, Accepted 19 February 2009, Available online 24 April 2009)