I.J.E.M.S., VOL.2(1) 2011: 38 - 42 ISSN 2229 -600X

INTERNATIONAL JOURNAL OF ENGINEERING AND MANAGEMENT SCIENCES

© 2004-2011 Society for Science and Nature (SFSN) All rights reserved

www.scienceandnature.org

CLUSTER-BASED DBMS MANAGEMENT TOOL

Sharad Nigam & Sunil Kumar Singh
Research Scholar, Singhania University, Pacheri bari, Jhunjhunu Rajasthan, India

ABSTRACT
A management tool which is needed for monitoring and managing cluster-based DBMSs has been little studied. So, we design and implement a cluster-based DBMS management tool with high-availability that monitors the status of nodes in a cluster system as well as the status of DBMS instances in a node. The tool enables users to recognize a single virtual system image and provides them with the status of all the nodes and resources in the system by using a graphic user interface (GUI). By using a load balancer, our management tool can increase the performance of a cluster-based DBMS as well as can overcome the limitation of the existing parallel DBMSs.

Key words: DBMS, Graphic user interface, cluster.

1. INTRODUCTION
Cluster systems are developed by connecting PCs and workstations using high-speed network first requirement is we need to support 24-hours nonstop service for the Internet. That is why; there are a wide range of researches on cluster-based DBMSs that offer a mechanism to support high performance, high availability, and high scalability. They include Informix Extended Parallel Server, Oracle 9i Real Application Server and IBM DB2 Universal Database EEE. To manage the cluster- based DBMS more efficiently, a management tool for the cluster- based DBMS is needed. First of all the tool enables users to recognize a cluster system consisting of multiple nodes as a single virtual system image. Secondly, by using a graphic user interface, the tool provides users the status of all the nodes in a system and all the resources (i.e., CPU) in a node. Thirdly, a load balance function is needed to make all the nodes perform efficiently by evenly distributing user's requests. Finally, a fail-over technique is needed to support high availability when the node failure is occurred. In this paper, we have designed and implemented a cluster-based DBMS management tool which monitors the status of all the nodes in a cluster system as well as the status of DBMS instances in a node. This tool will enable users to recognize a single virtual system image and provides them with the status of all the nodes and resources in the system with the use of graphic user interface (GUI). In addition, the cluster-based DBMS management tool with a load balancer can increase the performance of a cluster- based DBMS as well as can it overcome the limitation of the existing parallel DBMSs. The next section discusses related work on existing cluster management tools. In section 3, we designed a cluster-based DBMS management tool and it's GUI. In section 4, we have described the implementation and the performance analysis of our cluster-based DBMS management tool.

2. RELATED WORK
In this section, we introduce the existing management tools; the OCMS(Oracle Cluster Management System)which is a well known as a cluster-based DBMS management tool and the SCMS(SMILE Cluster Management System) which is a cluster system management tool for Linux Beowulf. OCMS is included as a part of the Oracle8i Parallel Server product on Linux and it provides cluster membership services, a global view of clusters, node monitoring, and also cluster reconfiguration. It consists of watchdog daemon, node monitor, and a cluster manager. First, the watchdog daemon offers it services to the cluster manager and to the node monitor. It makes use of the standard Linux watchdog timer to monitor selected system resources for preventing the corruption of database. Secondly, node monitor passes node-level cluster information to the cluster manager. It maintains a consistent view of the cluster and also informs the status of each local node of the cluster manager. The node monitors also cooperates with the watchdog daemon for stopping the node with the abnormally heavy load. Finally, the cluster manager passes the instance-level of cluster information to the Oracle instance. It maintains the process-level status of a cluster system. The cluster manager then accepts the registration of Oracle instances to the cluster system and then provides a consistent view of the Oracle instances. Figure A shows the overall architecture of OCMS

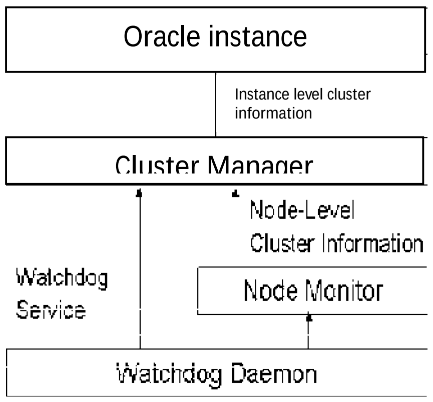
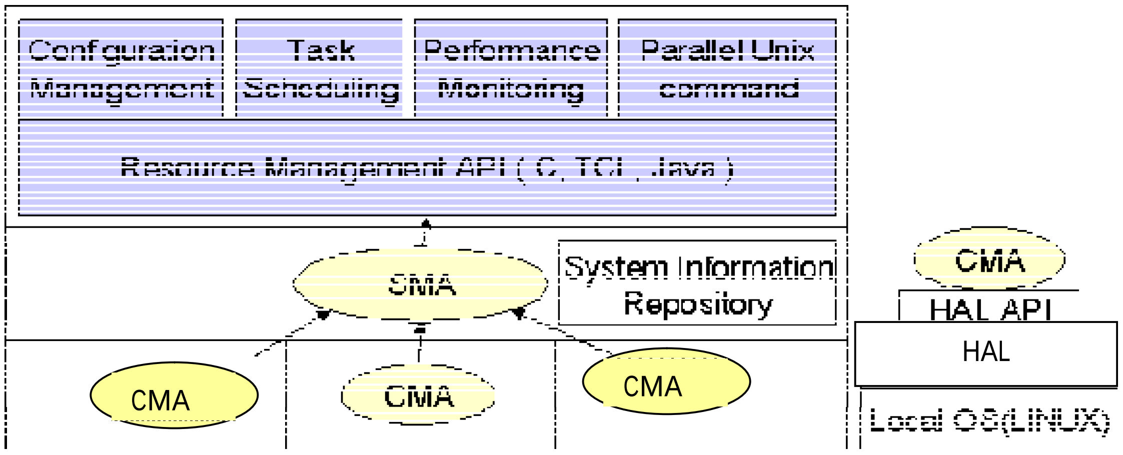


Figure A Overall architecture of OCMS

The SCMS (SMILE Cluster Management System) is developed by the Kasetsart University in Thailand as a cluster system management tool for Beowulf cluster. It consists of SMA (Systems Management Agent), CMA (Control and Monitoring Agent) and RMI (Resource Management Interface). Fist, CMA runs on each node and collects system statistics continuously. It reads system information through a layer called HAL (Hardware Abstraction Layer). Secondly, system statistics are collected by a centralize resource management server known as SMA. The SMA responses a user query on a system status and sends some commands to CMA. Finally, RMI is provided as a set of APIs for system monitoring and logging applications. By using the RMI, SCMS provides monitoring software's, configuration utilities and parallel UNIX commands. Figure B Shows the overall architecture of SCMS.



3. DESIGN OF CLUSTER-BASED DBMS MANAGEMENT TOOL

The cluster-based DBMS consists of multiple server nodes and uses a shared disk. Each server node is connected by every other by using high-speed gigabit Ethernet. A master node performs both the roles of a gateway of the cluster system and as well as a server node. It also manages all the information gathered from all the server nodes and uses a Linux virtual server for its scheduling algorithms. First, a user service request is been transmitted to the master node by using virtual IP. Then the Master node send it to a server node which is selected by the scheduling algorithm of the Linux virtual server. The selected server node then processes the user request and after that returns a result to the user. Figure C shows the overall architecture of cluster-based DBMS using high-speed gigabit Ethernet.

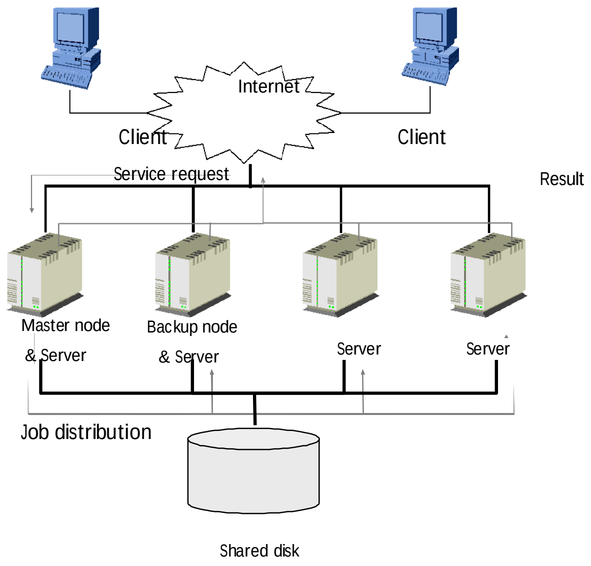


Figure C. Overall architecture of Cluster-based DBMS

3.1 Cluster-based DBMS Management Tool

This tool monitors both the status of system resources and database instance in each and every node. It also perceives the error of each node and then performs its recovery procedure to make a cluster-based DBMS run in normal situation. For designing a good monitoring tool, we first need to minimize the objects to be monitored so that we may avoid the additional load of monitoring itself. Secondly, we also have to change the frequency of monitoring dynamically so that we can control the amount of network traffic where a node transmits its status information to the master node. The cluster-based DBMS management tool has four components; probe, handler, CM(Cluster Manager), and also NM(Node Manager). In addition, the probe (or handler) can also be classified into system, DB, and Load Balancer (LB) probes (or handler). Figure D shows the components of the cluster-based DBMS management tool.

3.1.1 System Probe & System Handler:

The work of system probe is to monitor the status of CPU, memory, disk, and network of each node. For this, it uses /proc virtual file system to gather the status information of main system resources, such as CPU, memory, and disk, also for the transmission and reception status of packets through the network. It also generates events according to the status of the system. When the monitoring is performed without errors the system probe sends the events and also the monitored information to the system handler. The system handler then stores the events and the monitored information in the service status table, and then it performs a procedure according to the events. When an error occurs in the network, the system probe then updates the service status table. If the system probe or NM has a failure, the system handler makes it restart

3.1.2 DB Probe & DB Handler:

On the DBMS side, the DB Probe monitors the usage rate of CPU as well as of memory when DBMS is running and also generates events as a status of DBMS. When the cluster-based DBMS runs without any error, the Bribe generates 'DB_ALIVE' event. The DB Handler maintains a service status table and performs a procedure according to the event. In case the cluster-based DBMS having a failure, the DB probe performs a procedure to recover previous transaction, removes the failed server node from the available server list, and also makes the cluster-based DBMS restart. It performs a recovery procedure according to the network error perceived by CM.

3.1.3 LB Probe & LB Handler:

The LB probe monitors the load balancer and also generates event as a status of load balancer. For this, we tend to make use of a Linux virtual server as a load balancer. That is, we adopt a direct routing technique among the system structures supported by the Linux virtual server and also use a round robin scheduling algorithm. When the Linux virtual server runs without any error, the LB probe generates 'LB_ALIVE' event. The LB handler maintains a service status table and it performs a procedure accordingly to the event. In case the Linux virtual server has a failure, the LB probe then removes the failed server perceived by CM from the available server list and also then makes the Linux virtual server restart.

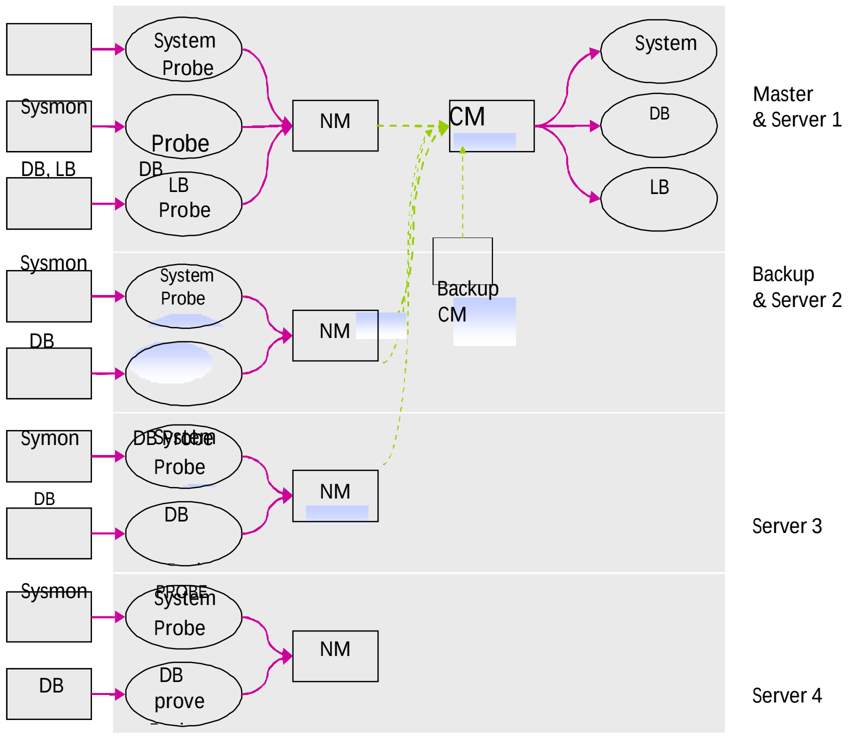


Figure D. Components of cluster-based DBMS man

3.1.4 Node Manager (NM):

The NM is a component which manages network communication with CM in the master node. It also transmits both the events generated by each probe and the monitored information to the CM in the master node. It perceives the status of CM according to the response of CM. If NM perceives the error of master node during the communication with the CM, then NM makes a connection with the backup node and also transmits all the information to it. The NM generates an event and then transmits it to the CM when it perceives an error of each probe. If the NM does not receive any response from the CM during a defined interval, then the NM considers that the CM has failed and makes a connection with a backup node.

3.1.5 CM (Cluster Manager)

The Cluster manager running on the master node manages the service status table that describes the process status of each probe, service and NM. It also perceives the status of networks by sending a ping message to each of the server node through both the cluster network and also the service network. Based on the status of networks and service status table, the CM then manages all system resources and services. It also then analyzes the events received from the each server node and transmits them to a handler to perform the appropriate procedure. It also makes the NM restart, if it perceives the error of the NM. If the CM perceives an error of the network by using a ping message, it then generates an event and transmits it to the corresponding service handler which has to perform its recovery procedure. Now since the CM is running on the master node, the failure of the master node then causes the failure of whole cluster- based DBMS. And now to solve the problem, the CM selects a backup node that also plays role of the master node when the master node has been failed. The backup CM running on the backup node communicates with the CM of the master node and stores in its local disk all information which the master CM manages. If the CM perceives the failure the backup node, it also selects one of the available server nodes as a backup node

4.0 Recovery procedures for failures

The Server nodes consisting of a cluster system can be classified into a backup node, a master node and a database server node. The status of nodes can be classified into 4 types according to the status of both service and cluster networks. First, if there is no failure in both the networks, the cluster system run with a normal situation. Secondly, if service network fails, a server node can then communicate with the other nodes, but it will not receive a user request and return the result to the user. Thirdly, if the cluster network fails, the server node cannot communicate with the others and so the master node cannot distribute an user request to a server node. And finally, if both networks fail, the situation is considered as a node failure since a node cannot work anymore. Status of nodes can be classified according to network failures as shown in Table 1.

Table 1. Classification of network failures

Network / Status of node	Cluster network	Service network
Normal situation	O	O
Service network failure	O	X
Cluster network failure	X	O
Node failure	X	X

4.1 Master node failure

In cluster system, a master node manages cluster-based DBMS by distributing a user request to the each of the server node. The master node sends a ping message to each server node in cluster system and perceives the failure of a node by analyzing the response of each of the server node. When the master node never receives the responses from any of the server nodes, it regards the situation as a cluster network failure. Meanwhile, when the backup node doesn't communicate with the master node using a ping message, it regards this situation as the master node failure. Now to prevent this situation, a backup node checks the failure of master node by sending it a ping message periodically and becomes a new master node when the master node has failed.

4.2 Backup node failure

In cluster system, the backup node stores each and every information received from the CM and monitors the master node. When the master node is failed, the backup node becomes the new master node and also selects one of available server nodes as a backup node. Now if the backup node has failed, the cluster-based DBMS management tool perceives the failure and then terminates the backup node. After then the backup node terminates active DB, Simon, NM, and also its backup CM. In this time, the master node performs its recovery procedure for removing the backup node from available server nodes and then to select a new backup node from them.

5.0 Performance analysis

We do the performance analysis of our tool using abase/Cluster. We estimated both a sensing time for three types of node failures and also a time to perform a recovery procedure for them. Table 2 shows the sensing time and the recovering time for three types of node failures.

Table 2. Sensing time and recovering time for three types of node failures.

	Sensing time	Recovering time
Master node failure	0.90	0.78
Backup node failure	0.88	0.51
Database server node failure	0.80	0.71

First of all , when the master node failure is occurred, the backup node becomes aware of the master node failure by using the result of ping message sent to the master node. We set the limit of response time for the ping message to 2 second. The time for sensing the master node failure is 0.91 second and the time for doing its recovery procedure is 0.78 second. If the backup node plays a role of the master node, it sets its virtual IP and creates its thread monitoring the network status of nodes in the cluster system. Secondly, when the backup node failure is occurred, the master node becomes aware of the failure and selects one of available server nodes as a backup node. The time for sensing the backup node failure is 0.89 second and the time for doing its recovery procedure is 0.51 second. A new backup node creates a thread to monitor the master node and receives the information of the service status table of the master node periodically. Finally, when the database server node failure is occurred, the master node perceives the failure and performs its recovery procedure. The time for sensing the database server node failure is 0.87 second and the time for doing its recovery procedure is 0.71 second. If a database server node failure has occurred, the master node removes the server node from the available server list in order that a user request is not transmitted into the sever node anymore.

6.0 CONCLUSTION

In this paper, we have designed a cluster-based DBMS management tool which is managing a cluster-based DBMS efficiently. Our tool monitored the system resources of all the server nodes and became aware of the failures of nodes. When any failure has occurred, our cluster-based DBMS management tool has performed its recovery procedure in order to perform a normal service, regardless of the failure. Our tool enables users to recognize a single virtual system image and can also provide them the status of all the nodes and resources by using the convenient graphic user interface (GUI). DBMS management tool supports nonstop service by performing its recovery procedure even though a node has failed.

7. REFERENCES

[1] Arkoma Bunya, High Performance Cluster Computing 1, 2, Prentice Hall PTR, 1999.

[2] High Performance Communication, http://wwwcsag.cs.uiuc.edu/projects/communication.html

[3] C. S. You, “Linux Clustering”, Communication of the Korea Information Science Society, Volt 18, No 2, pp33~39, 2000.

[4] J. Y. Choy, S. C. Wang, “Software Tool for Cluster”, Communication of the Korea Information Science Society, Volt 18, No 3. pp40~47, 2000

[5] Gregory, F.Pfister, In Search of Clusters 2nd Edition, Prentices-Hall, 1998.

[6] Linux Clustering, http://dpnm.postech.ac.kr/cluster/index.htm.

[7] Oracle Corporation, “Oracle 8i Administrator’s Reference Release3 (8.1.7) for Linux Intel", chapter 7, Oracle Cluster Management Software, 2000.

[8] Pitching Uthayopas, Jullawadee Maneesilp, ParichaIngongnam,"SCMS: AnIntegratedCluster Management Tool for Beowulf Cluster System", Proceedings of the International Conference on Parallel and Distributed Proceeding Techniques and Applications 2000 , Las Vegas, Nevada , USA , 26～28 June 2000.

[9] Linux Virtual Server, http://www.linuxvirtualserver.org.

[10] Hong-Yean Kim, Kid-Sung Jin, June Kim, and Ming-Jon Kim, "abase/Cluster: Extending the BADA-IV for a Cluster Environment", Proceeding of the 18th Korea Information Processing Society Conference, Vol. 9, No. 2, pp. 1769-1772, Nov. 2002.