INTERNATIONAL JOURNAL OF SCIENTIFIC PROGRESS AND RESEARCH (IJSPR) ISSN: 2349-4689

Fault Tolerance Using Grid Computing

Sahana S N[1] , Shashank A R[2,] Swetha K R[3,]

[12]*UG Student* [3]*Asst. Professor, Dept. Of CSE,*

BGS Institute of Technology,BG Nagara, Karnataka, India

Abstract - In this technology, lots of clients may be connected to a single server in order to share information and transferring data. But let's consider the problems of some existing system. Google Talk is used to make and receive calls and messages. But while transferring the files, GTalk would act as file is sending but actually it is not transfer at the destination. And at the time of receiving, when file is downloaded, it cannot be opened and copied to other location due to crashing of files in between transmission. In case of Apple Ipad, when we try to transfer file, it is just crashed and cannot transfer to at receiver. So such problems must be overcome by monitoring the transmission of files any by perfectly sending the crashed file at the particular location.

Key words:GTalk,Grid Computing.

I. INTRODUCTION

In recent history there is no recovery of work that is lost because of crash or failure. To reduce the failure in large application, in this paper we deal with the fault tolerance executing on grid or large cluster. Grids are referred to be more closely coupled, heterogeneous and geographically dispersed. And also, Grid computing [1] combines computer resources from multiple administrative domains to reach a common goal.

The main strategy of Grid Computing is to divide pieces of a program among several computers [2], it uses a middleware. A middleware [3] runs separately in each node which is used to allow access to grid members and to manage the clients. There are a number of clients acting as members of the grid system and there is a central administrator, who controls and monitors the actions of a client. There is time based roll back of data in the client machines. There is communication via message passing among the grid members. Files can also be transferred with limited access. These actions are managed by the networking and file transfer modules. There is an assignment of work to the clients by controller and stealing of work is objected by work stealing module. The client will log off from the grid automatically if it tries to do an illegal work. The roll back module deals with time based roll backing and recovery of files.

II. PROBLEM DEFINITION

The Grid computing combines computer resources from multiple administrative domains to tackle a single big problem that requires a great number of computer processing cycles or access to large number of data. There is no mechanism to recover any work that is lost due to a crash or failure. This paper deals with fault tolerance mechanisms to avoid failure in large applications executing on a Grid.

Grid computing can be thought of as a distributed system with non-interactive workloads that involve a large number of files. What distinguishes grid computing from conventional high performance computing systems such as cluster computing is that grids tend to be more loosely coupled, heterogeneous, and geographically dispersed. Although a grid can be dedicated to a specialized application, it is more common that a single grid will be used for a variety of different purposes. Grids are often constructed with the aid of general-purpose grid software libraries known as middleware.

The main strategy is the use of a middleware to divide pieces of a program among several computers. This project deals with mechanisms for roll-back in a crashed system. The grid controller monitors various clients and communicates with the clients through message and file transfer. The clients work independently and can pass messages and files among them. The controller can view any attempt of a client to do an illegal work. There is automatic log off mechanism provided in case of illegal work access.

III. EXISTING SYSTEM

This section gives information about the current grid system and its requirements. Grid computing system uses a pooled concept and it shares the computer resources like CPU, Memory and Storage [2].

In general, a grid computing system requires:

- At least one computer, usually a server, which handles all the administrative duties for the system.
- A network of computers running special grid computing network software.
- A collection of computer software called middleware.

The existing grid system handles the solution of a very large problem that is very costly when done using a single high performance system. Grid computing is cost effective as well as space consuming. But the drawback is that there is no mechanism to recover any work that is lost due to a crash or failure.

Grid size can vary by a considerable amount. Furthermore, "distributed" or "grid" computing, in general, is a special type of parallel computing that relies on complete computers (with onboard CPUs, storage, power supplies, network interfaces, etc.) connected to a network (private, public or the Internet) by a conventional network interface, such as Ethernet. This is in contrast to the traditional notion of a supercomputer, which has many processors connected by a local high-speed computer bus.

If middleware is the workhorse of the grid computing system, the control node is the dispatcher. The control node must prioritize and schedule tasks across the network. It's the control node's job to determine what resources each task will be able to access. The control node must also monitor the system to make sure that it doesn't become overloaded. It's also important that each user connected to the network doesn't experience a drop in his or her computer's performance. A grid computing system should tap into unused computer resources without impacting everything else.

IV. PROPOSED SYSTEM

In this section a new method of design is proposed after considering the limitations of the existing system. The complex infrastructure, mass storage devices and inter-connection networks in the grid systems cause problems like challenges to system reliability. The absence of fault tolerance mechanisms will cause a very high probability of failure. But the failure of a single node will lead to entire execution failure. Thus there is a need for roll back in a crashed system.

The various modules [4] implemented are:

1. Grid module: The main strategy of Grid Computing is to use a middleware to divide pieces of a program among several computers. In the grid module, a middleware runs separately in each node which is used to allow access to grid members and to manage the clients. If a grid member is working in the grid,no other member is allowed to play the same role a that time.

2. Network monitoring module: There are a number of clients acting as members of the grid system and there is a central administrator, who controls and monitors the actions of a client. Client-server computing or networking is a distributed application architecture that partitions tasks or workloads between service providers (servers) and service requesters, called clients. Often clients and servers operate over a computer network on separate hardware. A server machine is a high-performance host that is running one or more server programs which share its resources with clients. A client also shares any of its resources; Clients therefore initiate communication sessions with servers which await (listen to) incoming requests.

3. File transfer and message sending: There is communication via message passing among the grid members. Files can also be transferred with limited access. These actions are managed by the message and file transfer modules.

4.Work stealing: The stealing of work is objected by work stealing module. No client is allowed to take control over the work of other clients. The client will log off from the grid automatically if it tries to do an illegal work.

5. Work assigning: There is an assignment of work to the clients by controller. The portions of a large work are already assigned by the grid controller. Each client can access only the works assigned to him/her. Whenever a client tries to access an illegal work, it will automatically log off from the system. Also if a client attempts a new work it will be visible to the controller.

6. Check pointing and Roll backing: The roll back module deals with time based roll backing and recovery of files. Check-pointing relies on periodically saving the state of the computation to stable storage. If a fault occurs, the computation is restarted from one of the previously saved states.

A crashed process can be recovered by

1) Restoring it to the initial state and

2) Replaying the logged events to it in the same order they appeared in the execution before the crash.

V. SYSTEM DESIGN

A Work flow diagram

There is a central grid controller who can monitor all the various actions of the grid members working in the grid system. There are a number of clients who have separate work areas and are working on a single big problem.

The client can do a work that is already assigned to him by the grid controller or he can attempt a new work. Another action of the client is that he can update his works and can perform communication via message passing and file transfer. While selecting the files there are two options. The client can either select a legal work that he is supposed

to do or he can try an illegal work of some other member. The grid system doesn't allow such a bribing of work. Such an attempt is named as 'Work Stealing'. The client who tries to access an illegal work is called a thief. Our project has a fine mechanism to control and prevent an illegal process. The 'Flexible Rollback in grid Computing' responds to illegal process access by an automatic logging off mechanism.

On trying to log off, a log-based recovery mechanism is applied so that the current work of the client need not be lost. This is done with respect to time. The current states of the files are saved automatically just before logging off. Also there is a chance for the system to crash or a power failure can occur. In this case also there is an automatic time based check pointing applied in-order to save the lost data. The controller can monitor all these events in time. The figure 5.1.7 shows the work flow diagram which shows the overall working of the time based roll-backing.

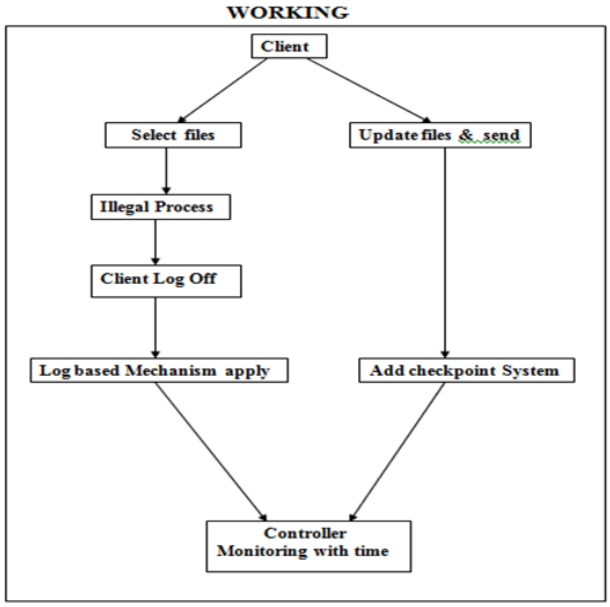

Fig 1 Work flow diagram

B. Data Flow Diagrams

The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of the input data to the system, various processing carried out on these data, and the output data is generated by the system.

1. Level 0 DFD

The figure 2 shows the level 0 DFD. It shows the members that are grid controller and clients interacting with the grid system. They share various resources through the database, as shown in fig 2.In general a grid system requires a grid controller and a network of computers called clients. Grid controller and client can communicate with each other by sending and receiving messages through ports. Also clients can transfer files. The grid controller monitors all the works done by the clients. The monitoring and roll backing are done by inserting to and retrieving data from corresponding databases.

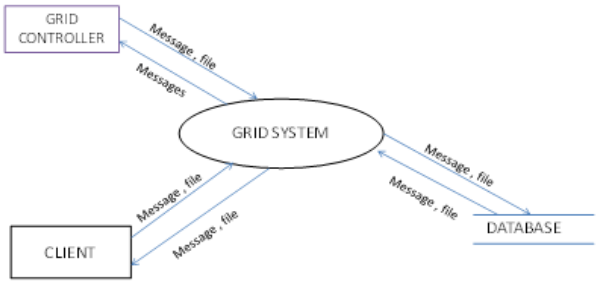

Fig 2 DFD of Level 0

2. Level 1 DFD

Figure 3 shows the role of middle ware in a grid system. Middle ware is used to control who can access system and use its resources. One user can't play the role of another user who has already logged in to the grid system. This can be done by setting a variable to either 1 or 0. To check whether the grid controller is active the value of the variable is retrieved from the database table grid. If the value is 0ne, then it means the grid is already running in the grid, and the user can't be logged in as the grid controller, so user exits the grid system. If the value of the variable is zero, then the user will undergoes a validation process by checking the username and password. If the validation process is successful, then he can enter the controller area. On entering the controller area, the value of the variable is set to one and that entry is inserted in to the table grid.

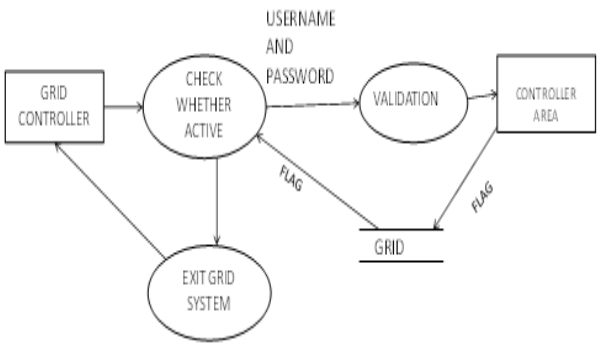

Fig 3 Level 1 DFD of Grid controller

Similar to the grid controller the user also enter to the client work area after checking whether client is active or

not. It is also using the table grid. This is as shown in figure 4.

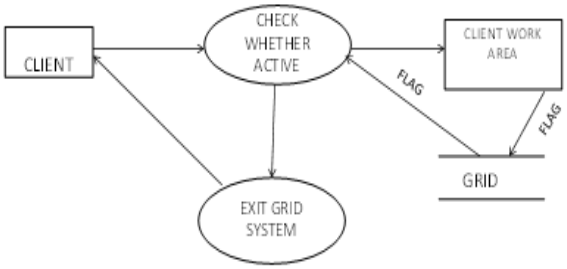

Fig 4 Level 1 DFD of Client

3. Level 2 DFD

Figure 5 shows the client work area. The client can perform various tasks i.e shown in this figure. He can select and do an existing work or he can create a new work. If he is selecting an existing file, then after a specific time period, the work will be automatically saved. This is called time-induced check pointing. The file system is not a database; it is actually the file system of the computer. On the other hand, if he is creating a new work, he has to do the work in client workspace and it will be automatically saved to the file system. In order to make the new work accessible by the grid controller, it is inserted to the table file. In any case if a client is doing a work, then a message showing his activity is inserted to the table client. In our project there are facilities for sending and receiving messages and files through ports.

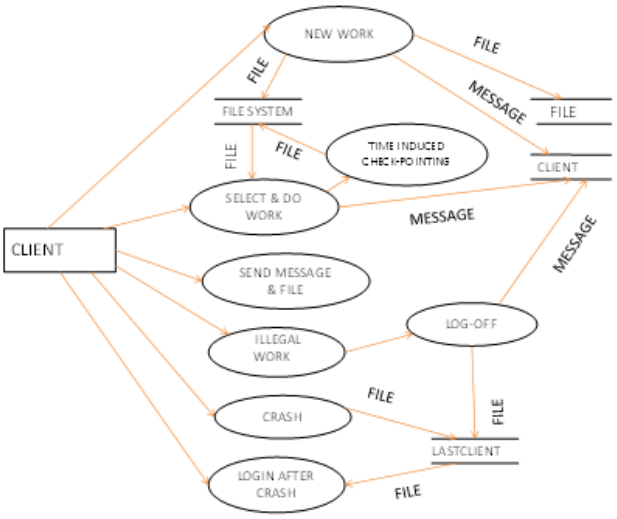

Fig 5 level 2 DFD of client

A client can only access the work assigned to him. If he is trying to do an illegal work, then he will be automatically logged off. And a message is inserted to the table client. Also the work is saved and inserted to the table last client in order to avoid the loss of work. If the system crashes, the work that we have done yet is saved to the table last client. Then after the crash when the client is logged in, the work is retrieved from the table last client.

Fig 6 shows the grid controller area. The grid controller can view and monitor the all the activities ie performed by all the clients by retrieving values from the table client. Similar to the clients, grid controller can also send and accept messages and files through ports. Another function of grid controller is to save clients new work, that is retrieved from the table file in to the computer's file system.

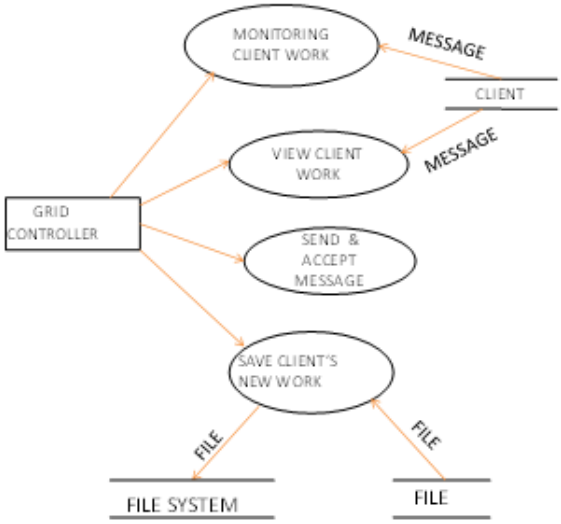

Fig 6 Level 2 DFD of grid controller

VI. CONCLUSION

It overcomes the problems of applications executing in large systems. According to this, the work performed by various nodes in a grid system gets automatically saved, thus providing a fault tolerant mechanism. In case of any system crash or power loss the individual nodes can resume their work, using this check-pointing system. A grid middleware had been implemented to control the access to grids. Also the grid controller monitors the client work and communicates with them through file and message transfer.

We have implemented a time based approach for automatic file updating. After the specified time interval the files get automatically saved. A client can do new work or select an existing work or can work on file send by the server. The check pointing applies to all these work and there will not be any loss of work. The clients are given permission only to select a work that is assigned to him. The main grid controller monitors all the client works.

REFERENCES

[1] https://www.techopedia.com/definition/87/grid-computing

[2] IEEE paper on "Flexible Rollback Recovery in Dynamic Heterogeneous Grid Computing " by Samir Jafar, Axel Krings, Senior Member, IEEE, and Thierry Gautier in the year 2009.

[3] "Middleware Architecture with patterns and framework" by Sacha Krakowiak in February 27, 2009.

[4] "Grid Infrastructure Architecture – A modular approach from core grid" by Augusto Ciuffoletti, Antonio Congiusta, Gracjan Jankowski, Michal Jankowski, Norbert Meyer, Ondrej Krajicek in 2006.