Design and Implementation of Dual Database Server and Clients

Asia Ali Salman * Intisar Shaded Chleib** Khalid Tourky Atah***

Received on: 17 / 9 / 2007
Accepted on: 7 / 7 / 2008

Abstract

The aim of this article is to find an efficient method to keep the database of critical tasks correct and update for performing the work of real time processing (such as power or water distributed environment), design and implement dual database server to control access to the database so the client read from one of dual server and update, write to both of them, if one of them fail then the other one must be capable of control and access all the client to its database. When the other sever return to work the worked server would be responsible for updating this server database. The result shows that, using duality system resolves those problems existing in single server system and eliminates the dependency on the clients in accessing the database; the existence of a second active server gives continuity for the work and never stopped the clients.

الخلاصة

تهدف المقالة الى ايجاد طريقة كفوءة للحفاظ علىالبيانات المهمة صحيحة ومحدثة من اجل اتجاز العمل لانظمة الوقت الحقيقي(real time system ,متل انظمة توزيع الطاقة اوالمياه). تصميم وتنفيذ خادمين لقاعدة بيانات ثنائية يمتلكون السيطرة على وصول الزبائن الى كل قاعدة من قواعد البيانات الموجودة لكل خادم, الفكرة بان نكون خادم على كل حاسبة شخصية كل من خادم الاول والخادم الثاني يمتلك نسخة من قاعدة البيانات المراد الوصول اليها من قبل الزبائن الذين يرتبطون مع كل خادم ,القراءة تتم من خادم واحد بينما التحديث والكتابة تتم على كلا الخادمين لضمان تطابق البيانات. في حالة فشل احدهم, الخادم الاخر هو الذي يكون مسئول بحيث كل الزبائن يتجهون اليه قي القراءة والكتابة والتحديث للبيانات عندما يعود الخادم للعمل ,الخادم الذي كان مسؤول عن الربط والعمل مع بقية الزبائن سوف يقوم بتحديث قاعده بياناته وعودته للعمل مع الزبائن . النتائج اثبتت انه استخدامنا للثنائية في العمل حل مشكلة توقف الخادم الوحيد وايضاالغاء الاعتماد على الزبائن في وصولهم وتعاملهم مع قاعدة البيانات, فوجود خادم ثاني يكون فعال مع الخادم الاول اعطى استمرارية في العمل وعدم توقف الزبائن .

University of TechnologyComputer Science & information Technology
Computer ScienceGeneral Systems Company
Computer EngineeringGeneral Systems Company

1-Introduction

Dual database server means that there are two servers both of them have the same Database files and each one is on a private computer. Dual server are connected through the network using a specific network communication technique, any client attend to deal with the database files it must notify the dual server that it's exist so its request go through the network and gets the information it needs. Design and implement dual database server give a more powerful management (read from files and write to files); it allows making use of the functionality of networking since more than one path would exist between dual server and clients. Since dual servers both have the same files any error occurred in one of them there will be a backup database files updated and saved on the other server, dual server reduce the load on the primary server [1, 2].Reliability is gained, since if one of the servers is fault all connected clients could deal with the still working server [2], while if there is only one server and stopped all clients will stay waiting for any one of the dual server. This type of duality is called active dual server redundancy because at starting both of dual servers are working and watching each other and waiting to deal with any new clients. someone may be ask why we use this type of redundancy and performing loading process on network the answer has remained the same: the network increase the efficiency and reduce the costs[3].This type of system can be widely used in many real-time processing fields such as power supply scheduling, technical production control and so on[4]. many of SCADA (system control and data Acquisition) systems which is exist today build in there own SCADA package dual redundancy Server (primary and secondary server), redundancy used in many fields such as plant control and monitoring system, citect company use redundancy server ,a second server added so that if the primary server fails, the client's requests data from the second server [5], also Zhuang, et al [4] have ,design and implement a dual-module sun workstation redundant system in which a fault-tolerant computer is one such that if problems arise then the system has the capacity to fiend and correct them and ensure the system runs normally, while Cisco use in scaling Cisco RPMS deployments system, redundant high availability dual server, if one server fails, the other server uses its stored cache to keep tracking calls. Each message received by a high available server is forwarded to its peer ensuring that each server has an active, or hot, call state in the memory cache of managed calls [6].

2-Dual Database Server and Client

In dual server each server has almost three tasks:

1-When dual server starts they establish connection path between them by sending a handshaking message to each other, that it exists and starts successfully (if the communication path is established then statues information send from one server to another server, a living message is continuously send between the dual server to make each server notes quickly if one of them is disconnected from the other one) suppose they have names (primary server, secondary server).

If at starting both of them starting successfully then they will stay waiting for any incoming client that want to connect to them, if any one came then communication path is established between the dual server and this new client(s). If this client wants only to perform a read operation from the database files it will logically deals with the first server that it is connect to it. But if the operation is to update (write) database it will deal with dual server, as shown in figure (1).

2-Suppose that if secondary server (server2) is stopped the primary server (server1) could deals with this event since there is a communication path between them, therefore the primary server will blocked all connected clients to make them notice that it is the only existing server so it is became a master server and they must access only to it in reading and writing to database files, as shown in figure (2).

Note, the same operation is done if the primary is stopped and the secondary is still in service.

3- If the second server is returned to work, the master server would notice this event and must block (notify) the clients and synchronize its database files with the secondary server database. Then make the clients return to work a gain

While from the client side, each client has three tasks, note that the following description is about only one client and it is the same for all other clients.

1-When client start and find the dual server is working the client(s) will connect to them and perform its tasks on the dual server database files.

2-When client(s) is working with dual server normally, then one of the dual server, suppose the secondary server, is fault the client(s) will disconnected from the secondary server depends on the message came from primary server, client(s) will perform all its work (dealing with database files)on the primary server dealing with it as a master server.

3- Suppose the secondary server returned to work, the master server would notice this event and must block (notify) the client(s). The client(s) stay waiting until master server synchronize database of the secondary server and send message to the client(s) telling them that the secondary server is working. Client(s) could work with dual server; means update database files performed in the primary server as well as in the database files in the secondary server, while the read operation will stay with the primary server (master).

3- Descriptions for Dual Database Server Operations

In this section adscription of the dual database server operations will be presented:
Note: the dual server would be called as server1 and server2 only for distinguish between them and we will explain the dual operations between these two servers in the following subsections.

3.1- Startup

This is the first operation which is used to read the Startup information consist of the names and paths of connection link between server and clients, also it responsible on check the existing of the database files folder if it is exist.

3.2- Initialization

This operation creates a connection path which is used to receive the server2 path. Create function that is used for receiving any incoming clients or performing the connection with the server2 and create operation functions that are responsible on watching an availability of server2 and notify clients about it as described below.

3.2.1- DualDB Moniter Client Function

This function creates a connection path that is used to receive the path of the new client. It doesn't start receiving any new client until the connection between the dual server is established or the number of server1 attempts to connect with server2 is reach a predefined constant value specified by the designer this means that the server2 is not working if we reach to this value, then server1 inform each new client that the server2 is not working. When DualDB_Moniter_Client function received new client first would check if new client is already connected to server1 by calling the function named (Check_Exist_Client ()) if not or if it is an old client and disconnected then deal with this new client by calling the function named (DualDB_WithClients).

3.2.1.1- Check_Exist_Client Function

This function check the existing of the new client if its doesn't exist then deal with it as new client and return the Connect_Clients variable incremented by one, else check its status if it's disconnect then deal with it as old and disconnect client, return it's old index, else return negative value.

3.2.1.2- DualDB_WithClients Function

This function is prepared the required structured information that both of servers need, in order to find each other. They keep trying to connect to the client, after the connection success each server will send specified path used to connect to the new client, if connect operation is success then set the ClientStatus flag to 1. sending a start message such as **"server1"** to client, since server1 can know the existing of server2 or not, so if server2 is disconnect then send **Disconnect** message to the new client to ends the checking process performed by this client to server2. After that it is create Client_Thread which is used to watch the connected new client activities.

3.2.2- Watchdog Timer Function

At first watchdog timer technique means that if a pulse (signal in our work) does not arrive within time period, it causes an interrupt to indicate that the pulse is late or missing [7]. This is the idea of watchdog timer as hardware added to the work, but we use the idea and build it as software (subroutine named Watch_DualDB function ()) used to check the following:

Unexpected network error (such as, an accident cause cutting the connection cable) when this error occurred, notifying the connected clients would be faster. At practice, detecting this failure takes a lot of time (approximately one minute); to detect it faster (a few seconds) we use this technique. A constant variable is set to a predefined value and decremented or incremented depends on living messages, if it's equal to zero then this mean there is no living message between dual server so unexpected network error is occurred, therefore call for an operation to notify the connected clients (Notify_Client_Disconnect function) that server2 is stopped, then if the living message between the dual servers return sending between them, the timer variable returns to its initial state.

3.2.3- Keep watching other Server Function (DualDB_Moniter_Server)

it is responsible on keep receiving from server1 communication path that is established in the initialize function described above until server2 send its path location then another function would be created named (DualDB_Server) and change flags in order to prevent the new clients from connect to server1 until the connection is established between server1 and server2. If the server2 doesn't send it's path location increment number of attempts variable until it reach the predefined value in this case server2 not working so this would give the permission to the DualDB_Moniter_Client function to start reading new clients.

3.2.4- Monitoring other Server Function (DualDB_Client)

This function depends on the (DualDB_Moniter_Server) function as shown in (Figure-13-DualDB_Client function). When it is started it's open communication path of server2 and preparing server1 path, send its path to server2, server1 receive response of server2. Depends on information each server receives, another communication path is open and the connection is established (Server1will send information to server2 and vise verse). If the communication path between them is broken and server1 has connected clients then notify the client using Notify_Client_Disconnect function. If the communication path is broken or a network error is occurred terminate the function DualDB_Server return to the beginning of process, if no error then compare the received data with "*Restart Server*" message if they are equal then set the RestartEvent event variable which is used in Notify_MyClients_Reconnect() function (this event means server2 was finished the update operation to server1 Database files), else keep watching server2.

3.2.5- DualDB_Server Function

Prepare the server1 path of interprocess communication technique and connect to server2, then keep watching it if any received data send a living message each 250 millisecond (if any problem is occurred in communication path the detection will be faster the value specified by the designer depends on trial and error process) if path is broken or network error occurred, then set all required variables to it's initial state and terminate, if server2 information is received then compare the received server2 information with server1 information and take the proper operation depends on the result of compression as shown in the following algorithm(**algorithm(1)**)

3.2.5.1- Notify_Client_Reconnect Function
Server1 calls this function when it's responsible on updating operation. This function creates an event used to know which client is response to a Hold message. Send a Hold message for the connected clients (depends on the ClientStatus to know which client is connect) and wait until each client is response to the hold message, hold message will prevent the connected clients from access server1 Database files, so updating the server2 Database files will complete without interruption, the updating operation is performed by calling the SynchOther_ServerDB () function. Send a Restart message to server2 to make it know that the SynchOther_ServerDB () function is finished its work, so it ready now to receive and connect to new client(s). Send a Restart message for the connecting clients (depends on the ClientStatus to know which client is connect) to give them a permission to access server1 Database files and return to there work with server2.

3.2.5.1-A- SynchOther_ServerDB Function
This is the most important operation for dual server; it is responsible on updating the server2 Database files and vice versa depends on the conditions described in DualDB_Server Operation above, as described in section (3-3 Dual Database Client).

3.2.5.2- Notify_MyClients_Reconnect Function
The work of this function is the same as the work of Notify_Client_Reconnect () function, but it instead of calling SynchOther_ServerDB () function it will wait the RestartEvent event which is signaled from DualDB_Client. In which the signal of RestartEvent event after receiving the restart message from server2 when the updating operation is finished.

3.2.5.3- Notify_Client_Disconnect Function
This function sends a Stop message to the connected client(s) (depends on the ClientStatus) when server2 is disconnected means either connection is broken or unexpected network error.

3.2.6- Client_Thread Function

This operation is created by DualDB_WithClients() function for each new connected client, used by the server to keep watching the clients, if the connection path is broken or a network error is occurred, then set the flag variable to zero, disconnect and close the connection path of this client, and terminate. If received data throw the connection path is greater than zero, and then set the ClientEvent event to know that this client is response to the server send messages. The client response means that this client is entered a critical section (which is means that when the process (in our case client) accessing shard modifiable data, they need to be enforced in mutual exclusion, so when a process is accessing shared modifiable data, the process is said to be in a critical section, all other processes (at least those that access the same data) are excluded from there own critical sections[8]) to prevent itself from access Database files while updating operation. ClientEvent event was used in Notify_Client_Reconnect and Notify_MyClient_Reconnect operation. The following figures (3, 4) shown the execution of dual server and how both of them connected together.

Note:
Sending or receiving information between dual server and clients are done using the named pipe technique, mailsloat also used to establish the connection and to do some preparing situation at the beginning then we use named pipes technique which are a simple interprocess communication (IPC) mechanism included in Microsoft Windows NT, Windows 2000, Windows XP (in our work), Windows 95, and Windows 98 (but not Windows CE). Named pipes provide reliable one-way and two-way data communications among processes on the same computer or among processes on different computers across a network. One of the best reasons for using named pipes as a networking communication solution is that they take advantage of security features built into Windows NT and Windows 2000, Windows XP(in our work)[9].

4- Description for Dual Clients Operations

Each client would call DBInitialize (Client Name) function the client would read the startup information and prepare its names used with server1 and server2, then call Initialize function which is the first operation in the Dual Client class. The Dual Client class consists of the following function:

4.1- Dual Client1 Function

Creates Client_Wait_Server1 function in which the client would receive connection path name from server1. Open server1 connection path and sending the client connection path name to sever1 until the Client_Wait_Server1 is reading the client connection path name from server1. Use connection path which is created by dual server1, keep watch server1 messages as described in the following algorithm 2:

4.2- Dual Client2 Function

The work of this function is the same as Dual_Client1, but dealing would be with server2.

4.3- Client_Wait_Server1 Function

This function wait until server1 send client connection path name then set Start1 flag to TRUE to make Dual Client1 function exit from writing to the Mailslot of server1.

4.4- Client_Wait_Server2 Function

The work of this function is the same as Client_Wait_Server1 operation, but dealing with server2.
Note that when the dual server started any new client connect to them simultaneously as shown in the following figure (figure 5). If the clients disconnected from the dual server or the operator (manually) terminate the clients then the dual server terminates clients as shown in figure 6.

5- How Dual Server Start Running

Dual Server start work (run) when the computer is starting up. Because the dual server is exist at the following path (the start menu -> All programs-> startup ->DualServer.exe) it could be keep at any location, but to make it worked automatically put it in start up. When dual server start a tray icon in the notification area of taskbar of windows operating system will appeared, as shown in figure(6), also a window is appeared at the desktop with a message tell the operator that the dual server started successfully, this window display the names and paths of the connected clients and the other dual server, if the other dual server started first then, the first server would perform file synchronization to the second server, and vise verse if the second server started first. If the operator click at the close command at the right corner of the window would be disappeared in order to show the window a gain double left click at the tray icon. Right click at the tray icon would give you a popup menu contain number of commands could be done by the operator and its looks as shown in figure 7.

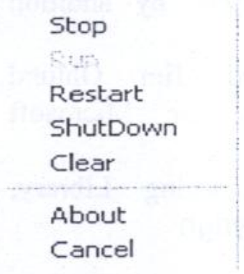


Fig (8)

These commands could be described as follows:

- Stop: this command will Stop the server and disconnect the clients from the server. So the stop and the restart commands then would be hidden and all other commands would be active. As shown in this figure (8).

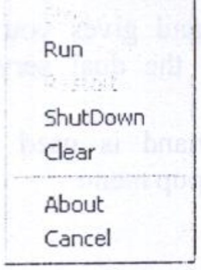


Fig (8)

- Run: this command is hidden if the Stop or Restart commands is active otherwise it is an active command. When Run Command is chosen the server would be running as shown in this figure (9).

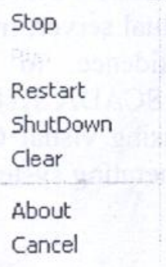


Fig (9)

- Restart: this command is doing as the Stop and Run commands doing sequentially all the commands would be hidden as in the figure (10).

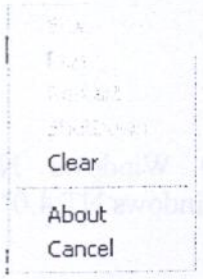


Fig (10)

- ShutDown: this command is ShutDown dual server it means disconnect the clients and remove the tray icon from the notification area of the task bar.

Note: if ShutDown command is used then to return the server in service the operator must follow the steps which is

(From the start menu -> All programs-> startup ->DualServer.exe) to run the server.

- Clear: this command would clear the window of the dual server.
- About: this command gives you a description a bout the dual server version.
- Cancel: this command is used to disappearing the popup menu

Results:
The result shows that dual sever system resolves those problems existing in old SCADA systems which depend on the clients in dealing with database with no server control access to the database. Using dual server faster and more reliable access to the database controlled by dual server, make the database more confidence to the operators and clients of the SCADA system. The system implemented using visual C++ ver 6.0 under win NT/XP operating system.

Conclusion:
The implementation shows that:-
Using duality in real time environments keeps the productivity of the work and leasing the probability of errors in database with controlling access to it by the redundant dual server.

References

[1]-MSDN Library 2000 Windows NT 4.0,"DNS and Microsoft Windows NT 4.0".

[2]- CitectSCADA version 6.0,"CitectSCADA_Tech_Overvie.pdf" ©copyright 2004, Citect Corporation.

[3]- Dr.Nihad M. Al-Rawi,Firas M.To'amia,Deya J.Kadhim,"Anew Technique of Remote Management for Power Plant", conference of Iraq Commission for Computer and informatic,2003

[4]- Yi Zhuang; Yang Lu; Bo Zhu; Min Zhu, "Design and implementation of a dual-server fault-tolerant real-time processing system", Intelligent Control and Automation, 2000. Proceedings of the 3rd World Congress on volume 5, Page(s):3618 - 3622.

[5]- http://www.controsys.hu/anyagok/Redundancy.pdf.

[6]- Cisco Resource Policy Management System (RPMS) Solutions Guide Release 2.0.2, chapter 3 ©copyright 1992-2008 Cisco Systems, Inc.

[7]-Thomes W. Schult, " C and the 8051 Hardware, modular, programming, and Multitasking", © copyright Prentice Hall, 1998

[8]-Harvery M. deitel, "An Introduction to Operating Systems", second Edition, critical section © copyrights 1990 by addition Company. Inc

[9]-Anthony Jones and Jim Ohlund "Network Programming for Microsoft Windows"
Microsoft Enterprise Learning Library, Developer Edition © Copyrigh

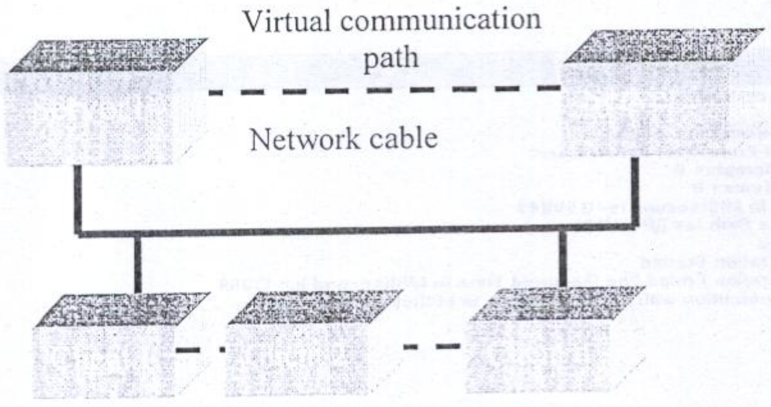


Fig. (1) Dual Server and Clients

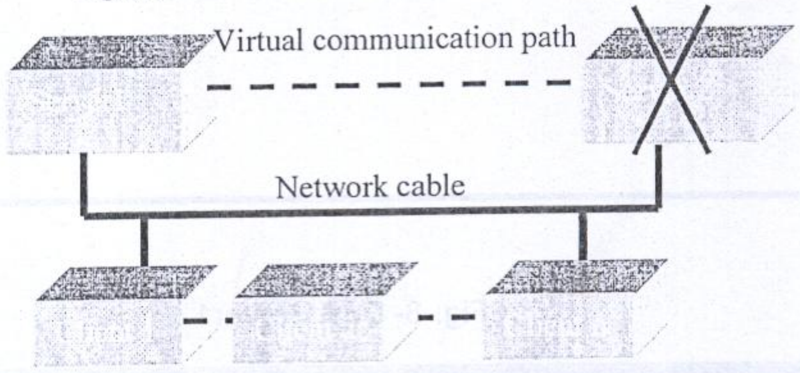
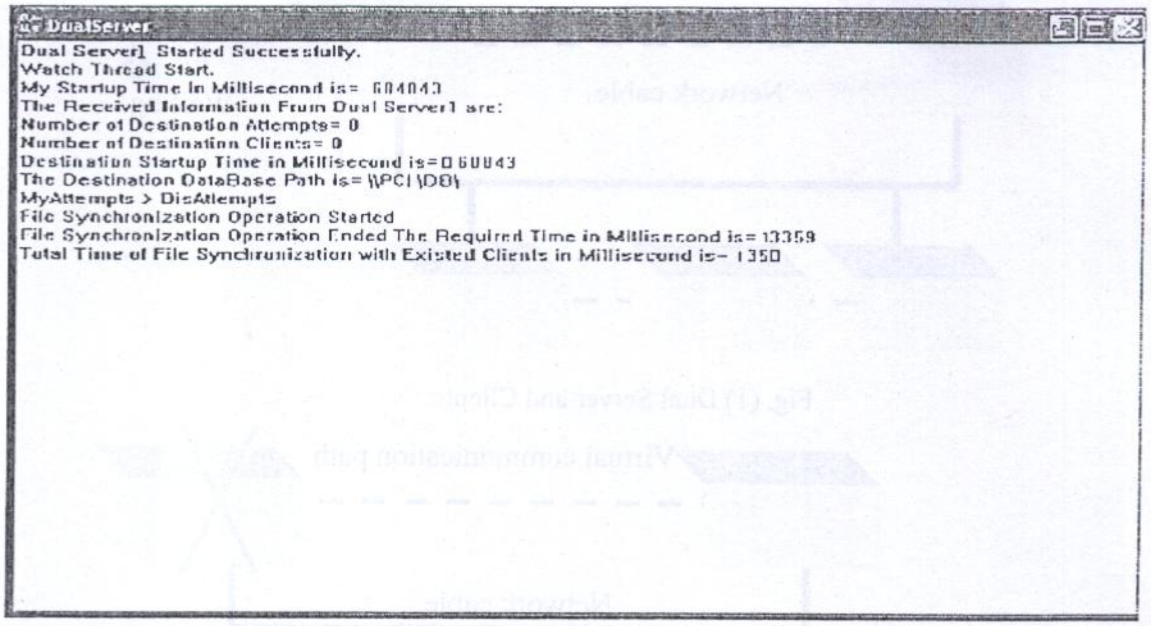


Fig. (2)One Dual Server and Clients

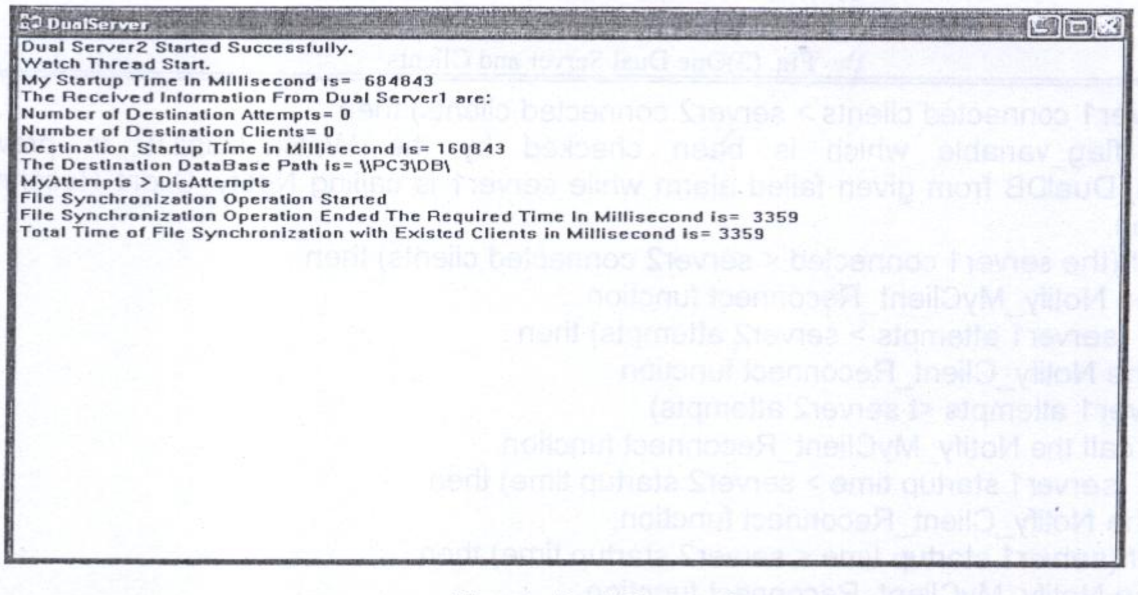
```
If (server1 connected clients > server2 connected clients) then
  Set flag_variable which is been checked by the Watch_DualDB to prevent
Watch_DualDB from given failed alarm while server1 is calling Notify_Client_Reconnect
function.
 Else if (the server1 connected < server2 connected clients) then
Call the Notify_MyClient_Reconnect function
Else if (server1 attempts > server2 attempts) then
 Call the Notify_Client_Reconnect function
If (server1 attempts <t server2 attempts)
 Then call the Notify_MyClient_Reconnect function
Else if (server1 startup time > server2 startup time) then
 Call the Notify_Client_Reconnect function;
 Else if (server1 startup time < server2 startup time) then
Call the Notify_MyClient_Reconnect function
 the above operations are done without allowing any interruption from new clients then do
some setting to flags checked by the clients note that server2 is connected to srever1
then allow new clients to connect to server1 and continue watching server2.
```

Algorithm (1) DualDB_Server Function



```
DualServer
Dual Server1 Started Successfully.
Watch Thread Start.
My Startup Time In Millisecond is= 684843
The Received Information From Dual Server1 are:
Number of Destination Attempts= 0
Number of Destination Clients= 0
Destination Startup Time in Millisecond is=0 60843
The Destination DataBase Path is= \\PC1\DB\
MyAttempts > DisAttempts
File Synchronization Operation Started
File Synchronization Operation Ended The Required Time in Millisecond is= 13359
Total Time of File Synchronization with Existed Clients in Millisecond is= 1350
```

Fig.-3- Dual Server1

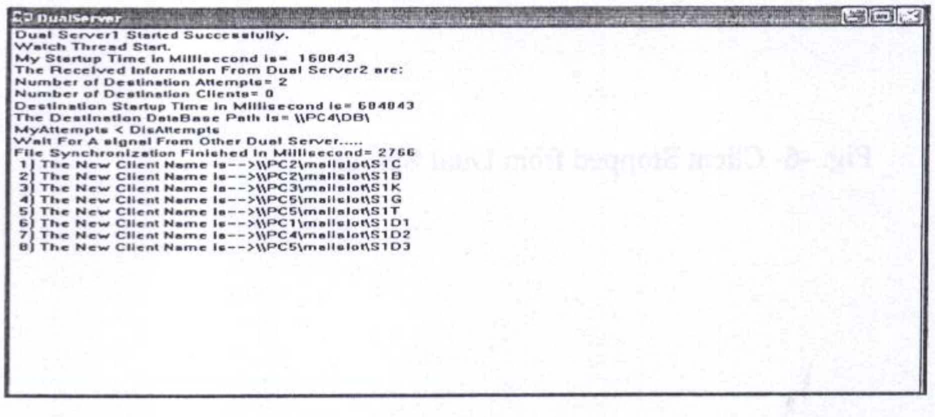


```
DualServer
Dual Server2 Started Successfully.
Watch Thread Start.
My Startup Time In Millisecond is= 684843
The Received Information From Dual Server1 are:
Number of Destination Attempts= 0
Number of Destination Clients= 0
Destination Startup Time in Millisecond is= 160843
The Destination DataBase Path is= \\PC3\DB\
MyAttempts > DisAttempts
File Synchronization Operation Started
File Synchronization Operation Ended The Required Time in Millisecond is= 3359
Total Time of File Synchronization with Existed Clients in Millisecond is= 3359
```

Fig.-4- Dual Server2

```
If (received data> zero) then
 Switch statement (on received data)
Case: received data=**Server1** then
Call DBServer1Initialize () function
 Set Status1
If (Prime Server = zero)
Prime server=1
TerminatDB1= 1
If (Critical2 = 1)
Means the Dual_Client2 is Disconnect (Entered Critical Section)
 Free it here by using Leave Critical Section function (to prevent client from access
database files while server2 is update database files of server1)
Case: **Disconnect**
 Means server2 not working and must terminate Dual_Client2 and Client_Wait_Server2,
 Case: **Hold**
 then prevent the client from accessing database files while server1 update server2 database
Files by using Enter Critical Section function and send a message to server1 that this client
was entered critical section
 Set Critical1 which is used by Dual_Client2 to leave critical section
Case: **Restart**
 Means server1 is finishing the updating server2 database files so the client could return to
work with server2 by creating Dual_Client2 function
Case: **Stop**
Means server2 is stopped
Terminate the Dual_Client2 function
End switch
 If connection path is broken or network error occurred then check the TerminatDB1
variable if equal to zero set it to one then call DBTerminate1 () function return to work
from the beginning else means Dual_Client2 function is received stop message first and
called DBTerminate1() and terminate the Dual_Client1 function.
```

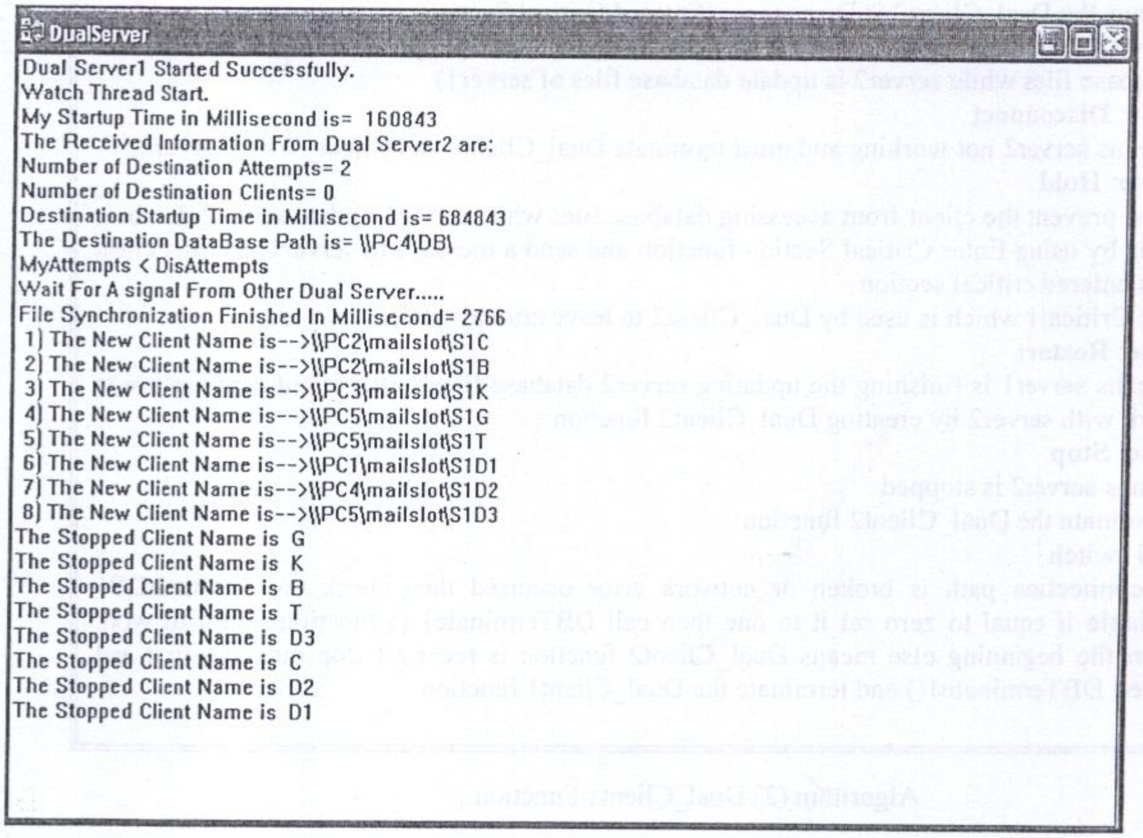
Algorithm (2) Dual_Client1 Function



```
Dual Server1 Started Successfully.
Watch Thread Start.
My Startup Time in Millisecond is=  160843
The Received Information From Dual Server2 are:
Number of Destination Attempts= 2
Number of Destination Clients= 0
Destination Startup Time in Millisecond is= 684843
The Destination DataBase Path Is= \\PC4\DB\
MyAttempts < DisAttempts
Wait For A signal From Other Dual Server.....
File Synchronization Finished In Millisecond= 2766
 1] The New Client Name Is-->\\PC2\mailslot\S1C
 2] The New Client Name Is-->\\PC2\mailslot\S1B
 3] The New Client Name Is-->\\PC3\mailslot\S1K
 4] The New Client Name Is-->\\PC5\mailslot\S1G
 5] The New Client Name Is-->\\PC5\mailslot\S1T
 6] The New Client Name Is-->\\PC1\mailslot\S1D1
 7] The New Client Name Is-->\\PC4\mailslot\S1D2
 8] The New Client Name Is-->\\PC5\mailslot\S1D3
```

Fig. -5- Client Connect to Dual Server

If the clients disconnected from the dual server or the operator (manually) terminate the clients then the dual server terminates clients as shown in figure 6.



```
Dual Server1 Started Successfully.
Watch Thread Start.
My Startup Time in Millisecond is=  160843
The Received Information From Dual Server2 are:
Number of Destination Attempts= 2
Number of Destination Clients= 0
Destination Startup Time in Millisecond is= 684843
The Destination DataBase Path is= \\PC4\DB\
MyAttempts < DisAttempts
Wait For A signal From Other Dual Server.....
File Synchronization Finished In Millisecond= 2766
 1) The New Client Name is-->\\PC2\mailslot\S1C
 2) The New Client Name is-->\\PC2\mailslot\S1B
 3) The New Client Name is-->\\PC3\mailslot\S1K
 4) The New Client Name is-->\\PC5\mailslot\S1G
 5) The New Client Name is-->\\PC5\mailslot\S1T
 6) The New Client Name is-->\\PC1\mailslot\S1D1
 7) The New Client Name is-->\\PC4\mailslot\S1D2
 8) The New Client Name is-->\\PC5\mailslot\S1D3
The Stopped Client Name is  G
The Stopped Client Name is  K
The Stopped Client Name is  B
The Stopped Client Name is  T
The Stopped Client Name is  D3
The Stopped Client Name is  C
The Stopped Client Name is  D2
The Stopped Client Name is  D1
```

Fig. -6- Client Stopped from Dual Server

The following flowcharts describe Dual Server work

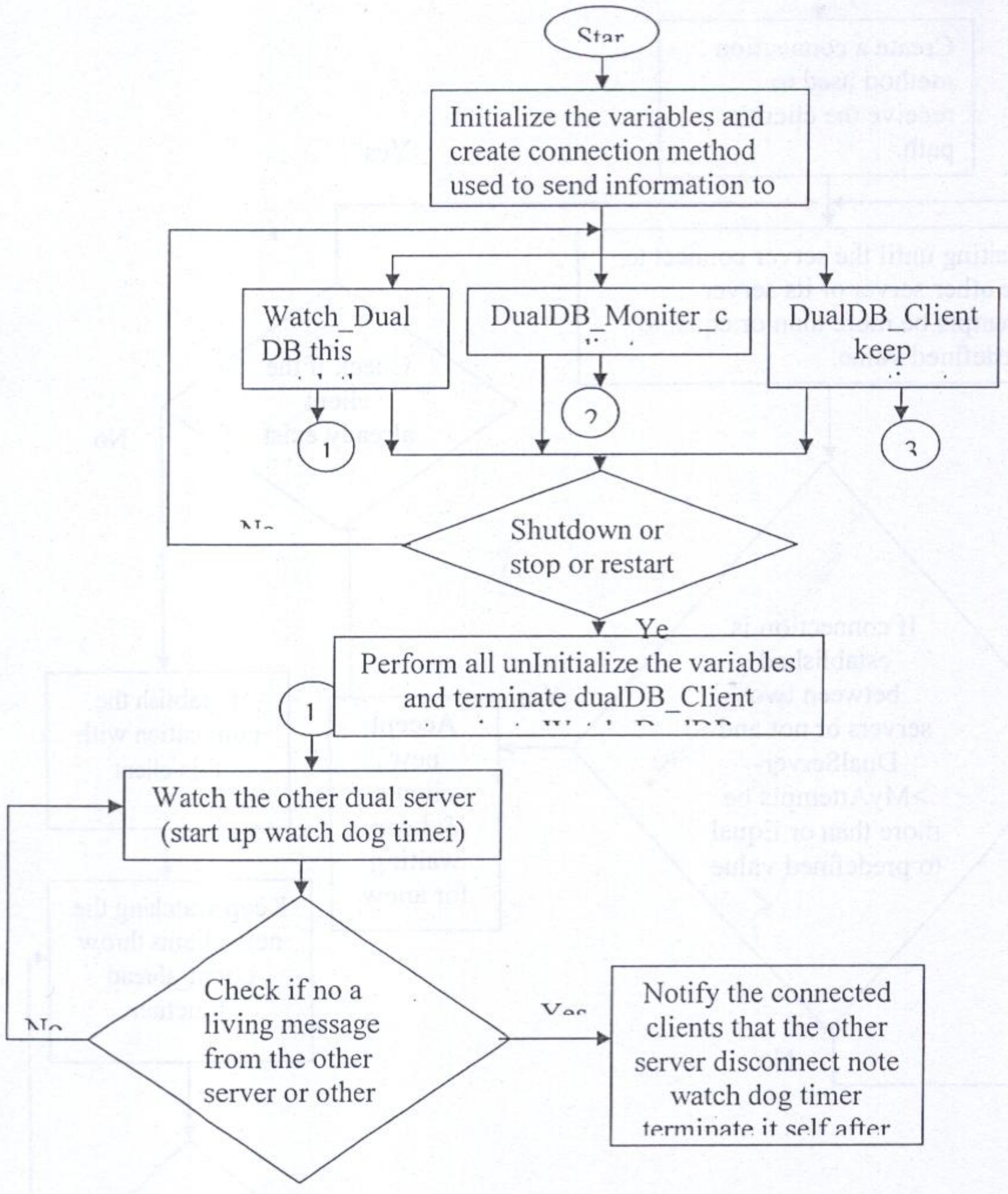

Star
Initialize the variables and create connection method used to send information to
Watch_Dual DB this
DualDB_Moniter_c
DualDB_Client keep
1
2
3
Shutdown or stop or restart
No
Ye
Perform all unInitialize the variables and terminate dualDB_Client
1
Watch the other dual server (start up watch dog timer)
Check if no a living message from the other server or other
No
Yes
Notify the connected clients that the other server disconnect note watch dog timer terminate it self after

Fig. (11)Watch_DualDB function

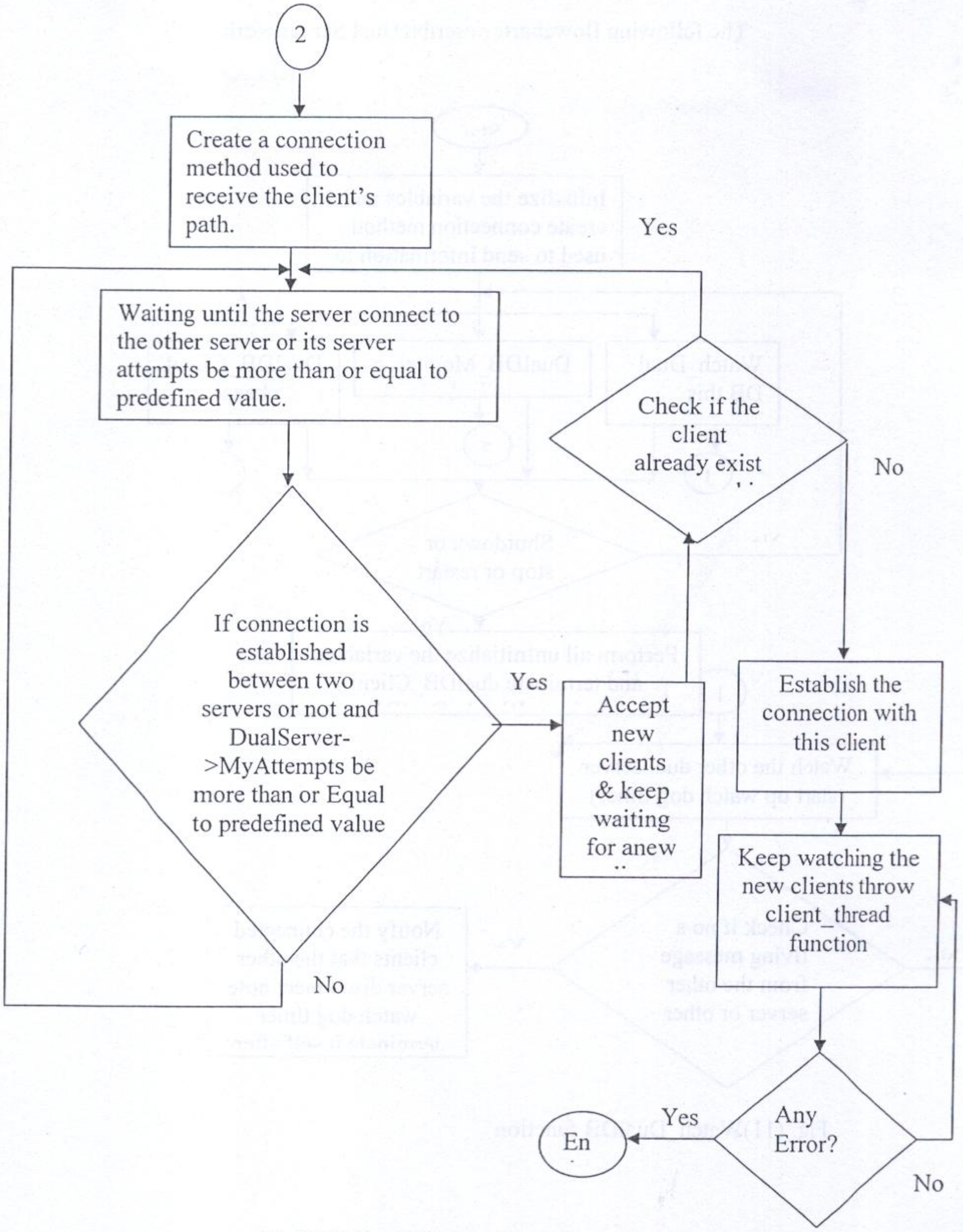

2
Create a connection method used to receive the client's path.
Waiting until the server connect to the other server or its server attempts be more than or equal to predefined value.
If connection is established between two servers or not and DualServer->MyAttempts be more than or Equal to predefined value
Yes
No
Accept new clients & keep waiting for anew
Check if the client already exist
Yes
No
Establish the connection with this client
Keep watching the new clients throw client_thread function
Any Error?
Yes
No
En

Fig (12) DualDB_Moniter_client
function

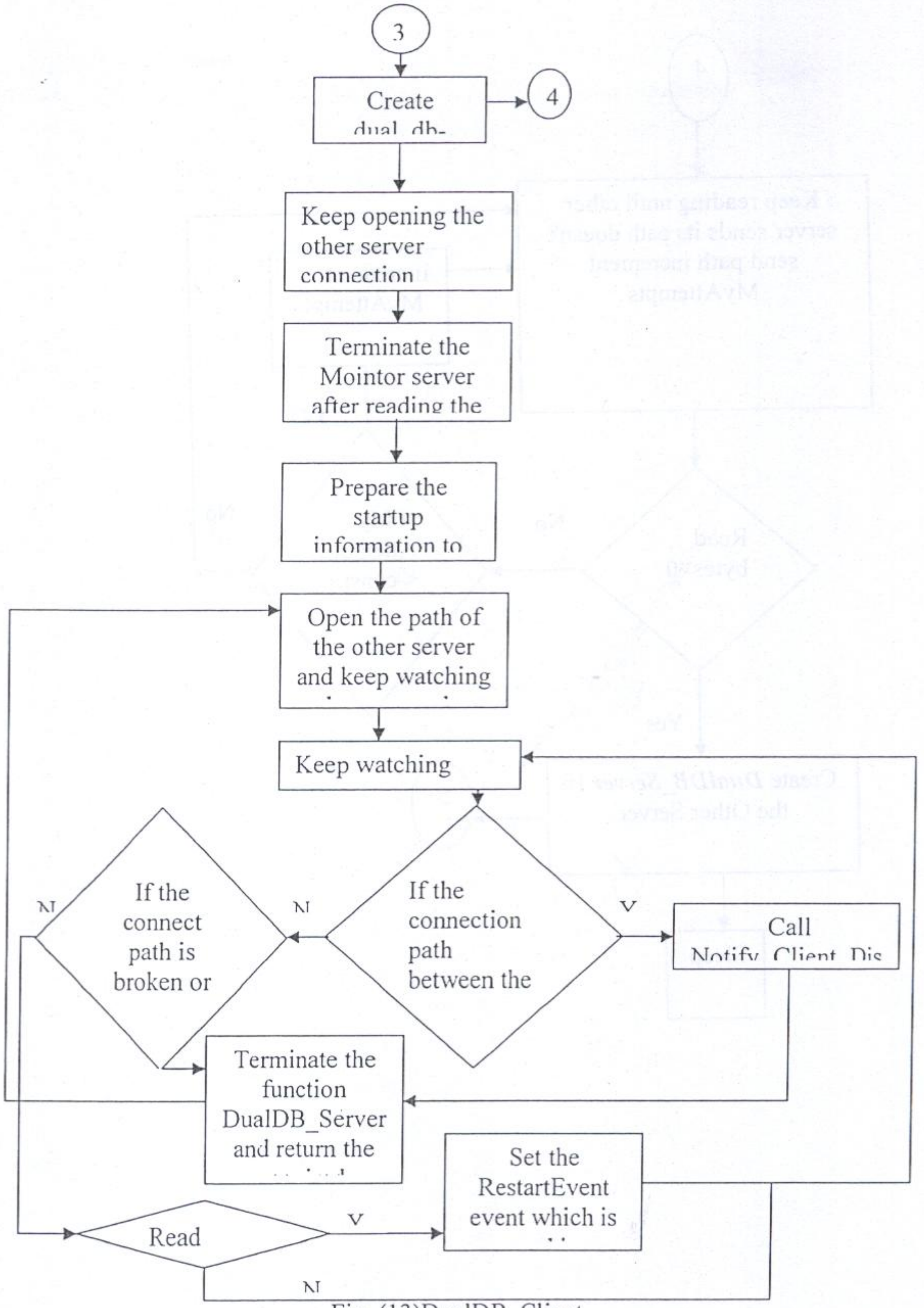
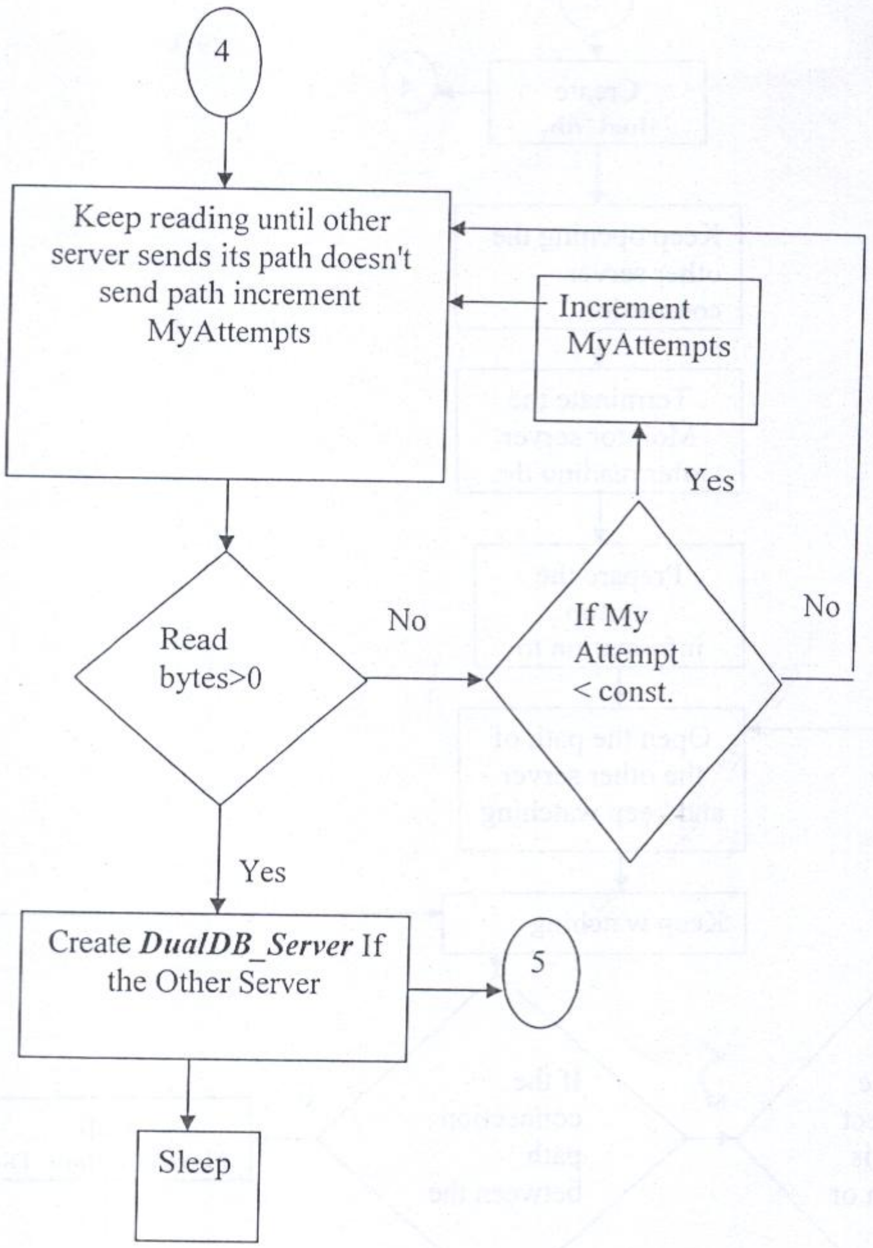

3
Create dual_db-
4
Keep opening the other server connection
Terminate the Mointor server after reading the
Prepare the startup information to
Open the path of the other server and keep watching
Keep watching
If the connect path is broken or
N
If the connection path between the
N
Y
Call Notify_Client_Dis
Terminate the function DualDB_Server and return the
Set the RestartEvent event which is
Read
Y
N

Fig. (13)DualDB_Client


4
Keep reading until other server sends its path doesn't send path increment MyAttempts
Increment MyAttempts
Yes
Read bytes>0
No
If My Attempt < const.
No
Yes
Create *DualDB_Server* If the Other Server
5
Sleep

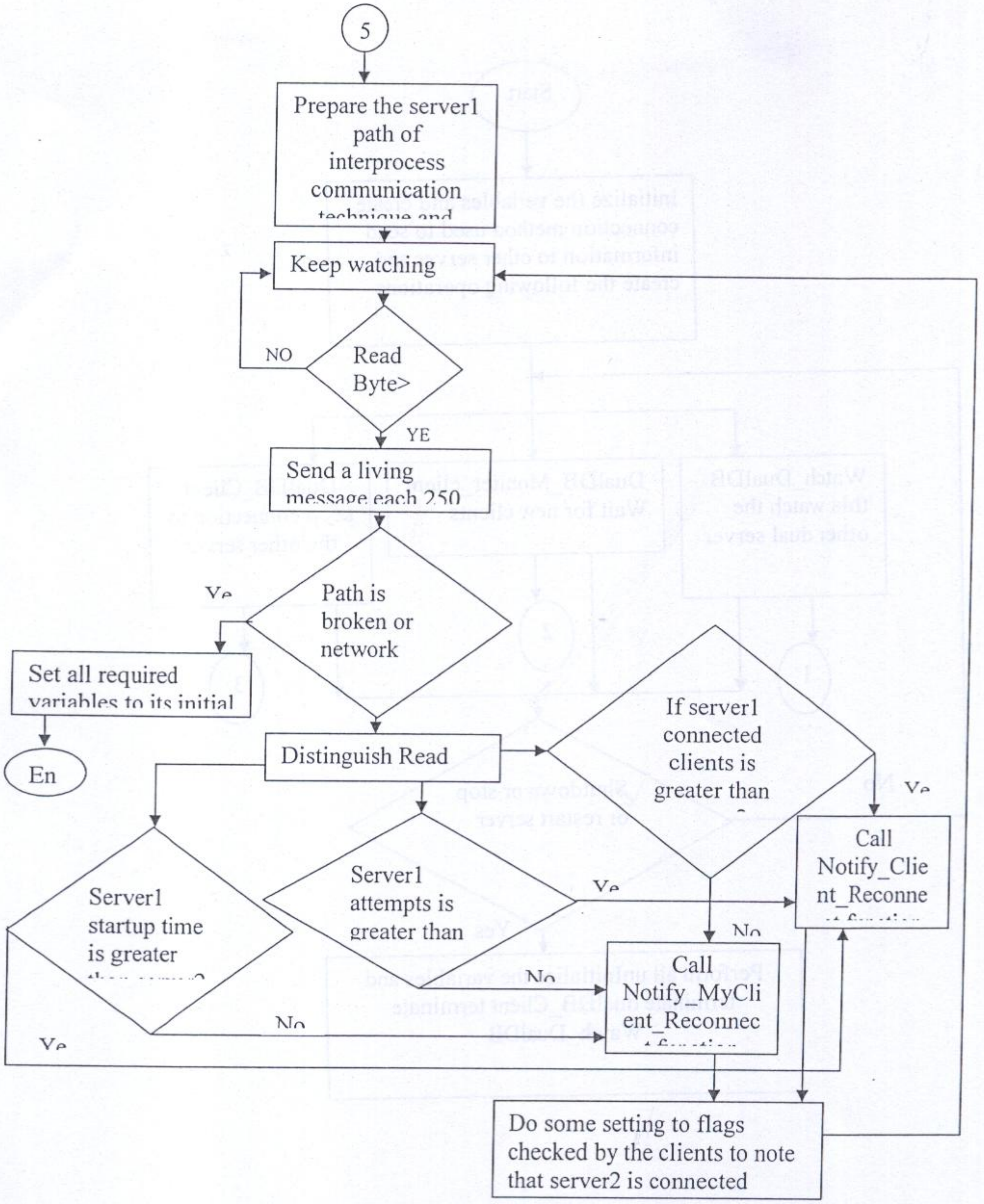
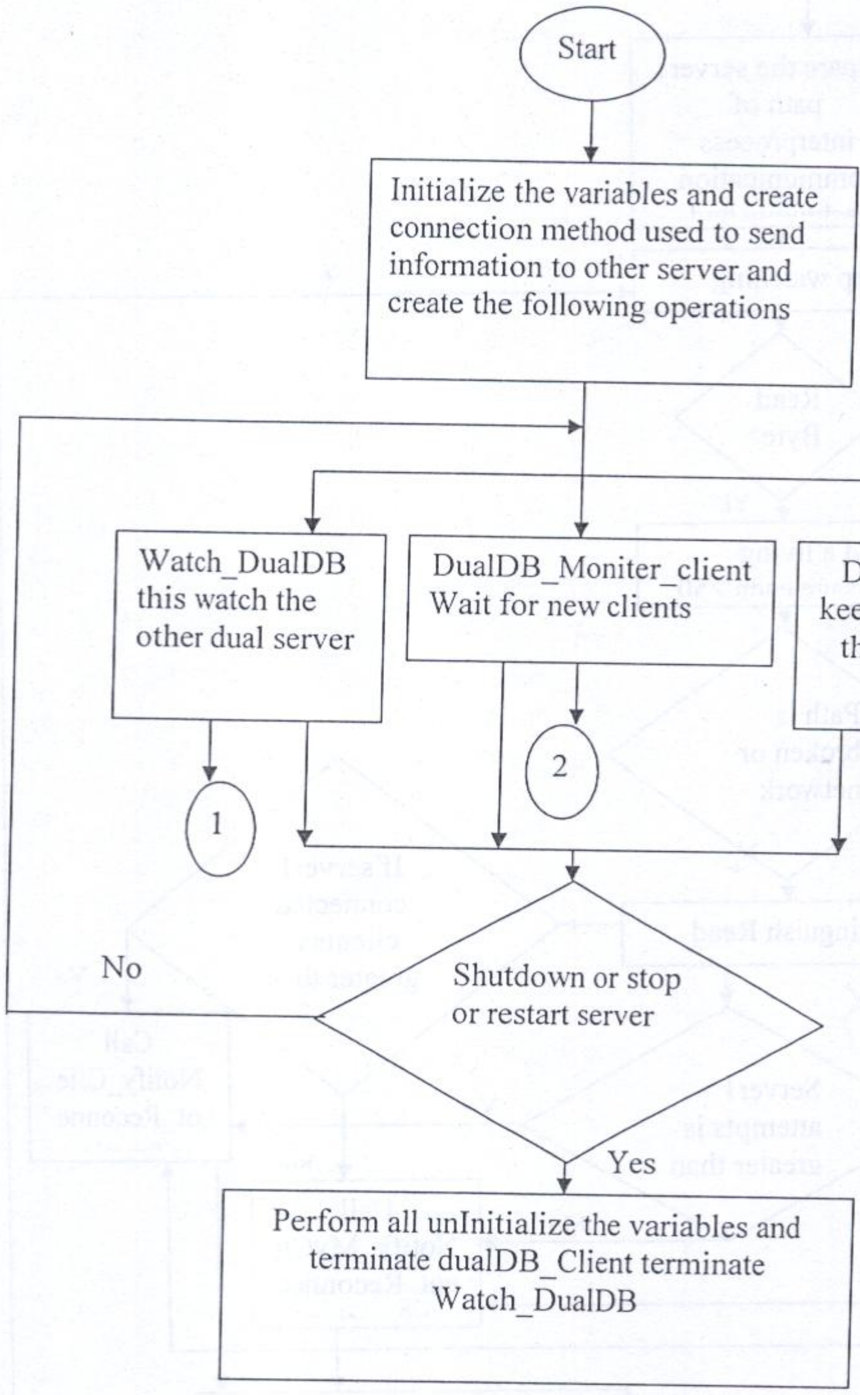
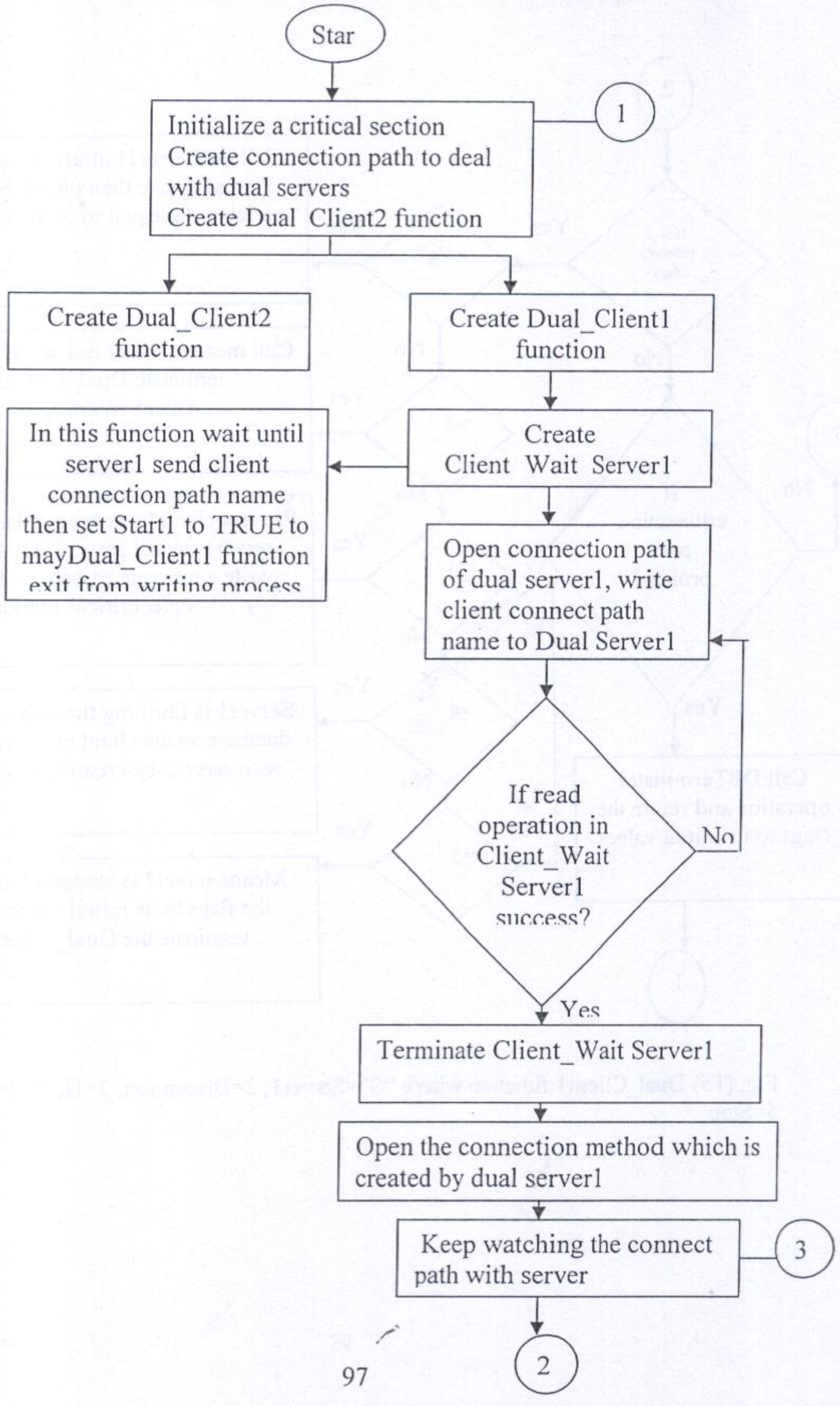

5
Prepare the server1 path of interprocess communication technique and
Keep watching
Read Byte>
NO
YE
Send a living message each 250
Path is broken or network
Ye
Set all required variables to its initial
En
Distinguish Read
If server1 connected clients is greater than
Server1 startup time is greater
Server1 attempts is greater than
Ye
No
Ye
Call Notify_Client_Reconne
No
Call Notify_MyClient_Reconnec
No
Ye
Do some setting to flags checked by the clients to note that server2 is connected

Fig. (14)DualDB_Server function

The following flowcharts describe Dual Server work


Start
Initialize the variables and create connection method used to send information to other server and create the following operations
Watch_DualDB this watch the other dual server
DualDB_Moniter_client Wait for new clients
DualDB_Client keep connection to the other server
1
2
3
No
Shutdown or stop or restart server
Yes
Perform all unInitialize the variables and terminate dualDB_Client terminate Watch_DualDB

The following flowchart describes the work of the dual client module:


Star
Initialize a critical section
Create connection path to deal with dual servers
Create Dual Client2 function
1
Create Dual_Client2 function
Create Dual_Client1 function
In this function wait until server1 send client connection path name then set Start1 to TRUE to mayDual_Client1 function exit from writing process
Create Client Wait Server1
Open connection path of dual server1, write client connect path name to Dual Server1
If read operation in Client_Wait Server1 success?
No
Yes
Terminate Client_Wait Server1
Open the connection method which is created by dual server1
Keep watching the connect path with server
3
2

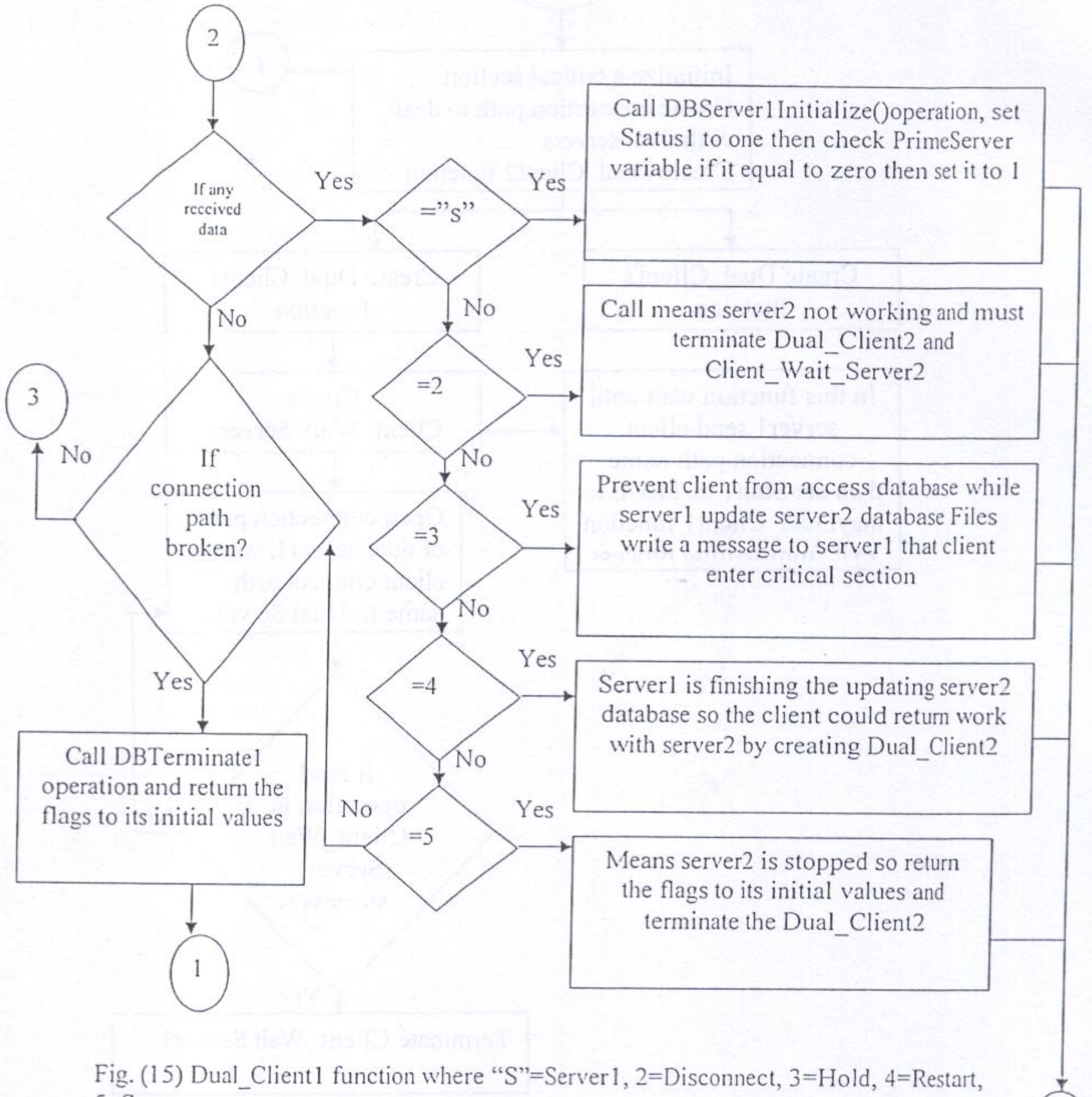

2
If any received data
Yes
No
="S"
Yes
Call DBServer1Initialize()operation, set Status1 to one then check PrimeServer variable if it equal to zero then set it to 1
No
=2
Yes
Call means server2 not working and must terminate Dual_Client2 and Client_Wait_Server2
No
=3
Yes
Prevent client from access database while server1 update server2 database Files write a message to server1 that client enter critical section
No
=4
Yes
Server1 is finishing the updating server2 database so the client could return work with server2 by creating Dual_Client2
No
=5
Yes
Means server2 is stopped so return the flags to its initial values and terminate the Dual_Client2
No
If connection path broken?
No
3
Yes
Call DBTerminate1 operation and return the flags to its initial values
1
3

Fig. (15) Dual_Client1 function where "S"=Server1, 2=Disconnect, 3=Hold, 4=Restart, 5=Stop

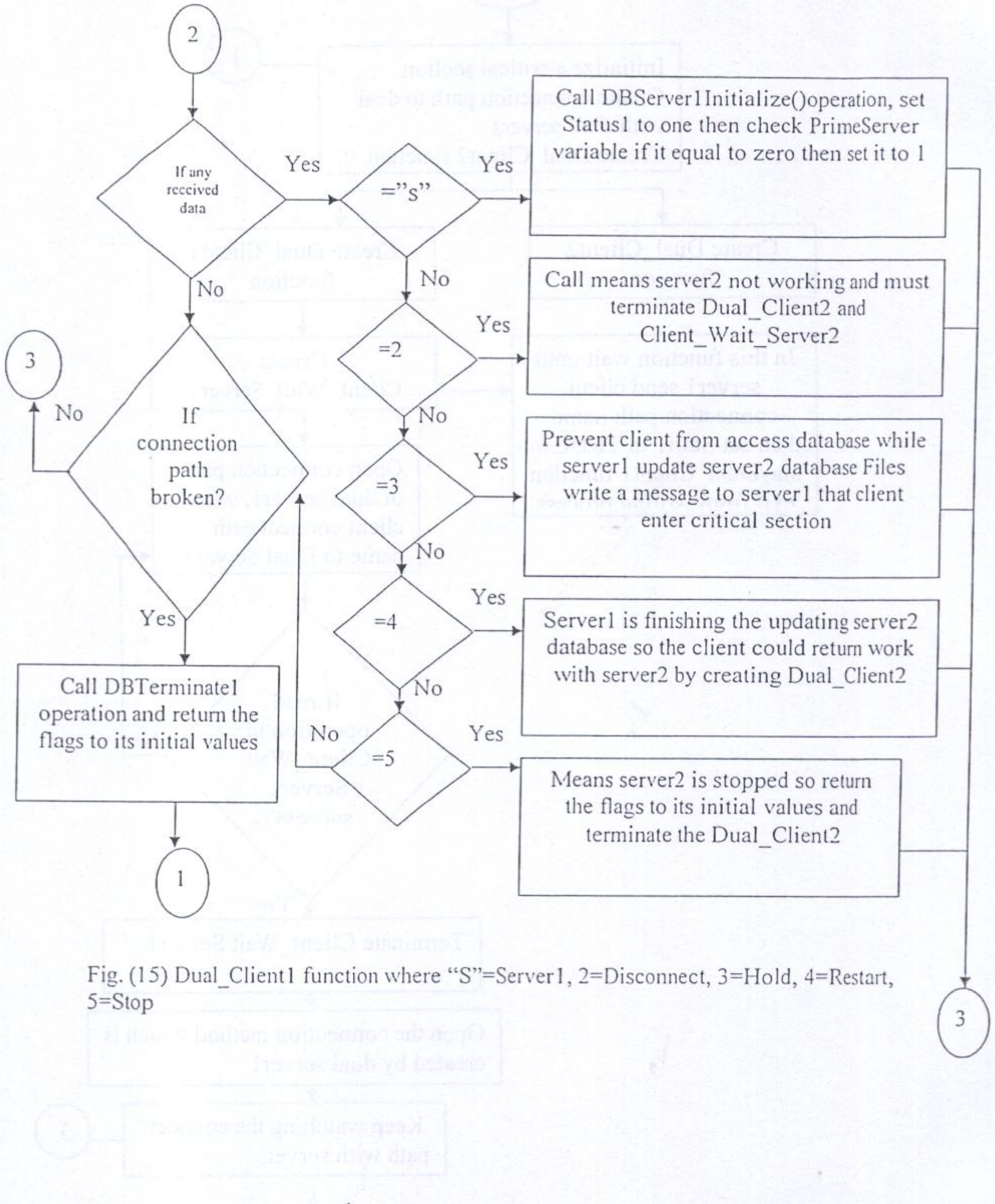

2
If any received data
Yes
No
="s"
Yes
Call DBServer1Initialize()operation, set Status1 to one then check PrimeServer variable if it equal to zero then set it to 1
No
=2
Yes
Call means server2 not working and must terminate Dual_Client2 and Client_Wait_Server2
No
=3
Yes
Prevent client from access database while server1 update server2 database Files write a message to server1 that client enter critical section
No
=4
Yes
Server1 is finishing the updating server2 database so the client could return work with server2 by creating Dual_Client2
No
=5
Yes
Means server2 is stopped so return the flags to its initial values and terminate the Dual_Client2
No
3
No
If connection path broken?
Yes
Call DBTerminate1 operation and return the flags to its initial values
1
3

Fig. (15) Dual_Client1 function where "S"=Server1, 2=Disconnect, 3=Hold, 4=Restart, 5=Stop