Fault-Tolerant distributed discussion board

- Santosh Bahir

- Mahendra Konda

The goal of the project is to develop Fault-Tolerant Distributed Discussion Board. General Requirements of the system are:-

1) Efficient use of messages – Don't send messages unnecessarily
2) System that produces correct results for all inputs
3) Handling of system crashes
4) Handling of network partition

Overview:

The discussion board is the place where information is shared among users/clients which are present over network of computer. The discussion board is server application running on the replicated server and clients can connect to any of the servers over the network. The key requirement is whichever server the client is connecting it should get consistent view. Maintaining consistency becomes more difficult when there are crashes of server s or network could partition away.

How the resiliency of the replicated server is maintained in face of partition and system failure is described in this design document.

This design document is organized as below:

A. Message Format:
B. Data Structure:
C. Client Program:
D. Server Program:

Message Format:

1) Exchange/Log Message Format:

TimeStamp (Leader)	Discussion Board	Object Number	Version Number	Object

2) Database Format: per discussion board, there will be one file. The record in the file will be as per the format given below.

Discussion Board	Object Number	Update Version	Object

3) Message from Client:

Operation	Discussion Board	Object Number	Object

4) Message from Server:

Discussion Board	

Data Structure:

Data Structure at server:

DB_clients: This is linked list of the clients currently connected to this server. The below information is kept about client

Client private group	Discussion board of client

Data structure at client:

discussion_board: This is the linked list of the object in the discussion board currently client joined.

Each node will contain following information

Discussion Board Index	Name of discussion Board	Object Number	Object

Client Program:

Client program is running at the user machine. It provides simple interface to users to provide below functionality. Data communication is done using Spread via the private group of the client and a group with only that server in it. The group created with Server when client connects to particular server is used to know if there is disconnection with server.

a) Spread Multicast Group:

Essentially, each client is member of below two types of group

1) *Client_private_group:*
   This is private group of client created when the client connects to Spread.
2) *Client_server_group:*
   This is a group with the server to which it is connected. This is only to realize the availability/failure of the server. No Data will be sent on this group.

1. Connecting to a discussion server. "c 3"

   Action: Client creates group client_group on Spread and sends the request message to the server group which known to client. In this request message, client will put the group name client_group. So the server will join this group. This group is just for membership messages so whenever the server crashes, client will come to know about this.

   Response to user: Once above action is completed, the client shows to user message
   "Connected to server 3"

2. Join a discussion. "j board1"

   Action: Client will send the message to server containing the 'discussion board name' and the operation as 'join group' (see 'client message format above')

   Response to user: Once above action is completed client wait until it get the discussion board with its object. Once the discussion board is received, it shows it to the user on screen.

3. Append to a discussion. "a" and then type a line

   Action: Client will send the message to server containing operation as 'Append 'and required details as per client message format.

   Response to user: Once the completion of append took place at server, client will receive response from server with updated discussion board. Client will show this updated discussion board to user (client will refresh the screen)

4. Edit an object in a discussion. "e 5" and then type a line

   Action: Client will send the message to server containing operation as 'Edit'and required details as per message format

   Response to User: Once the completion of edit took place at server, client will receive response from server with updated discussion board. Client will show this updated discussion board to user (client will refresh the screen)

5. Delete an object in a discussion. "d 5"

Action: Client will send the message to server containing operation as ‘Delete ‘and required details as per message format

Response to User: Once the completion of ‘Delete’ took place at server, client will receive response from server with updated discussion board. Client will show this updated discussion board to user (client will refresh the screen)

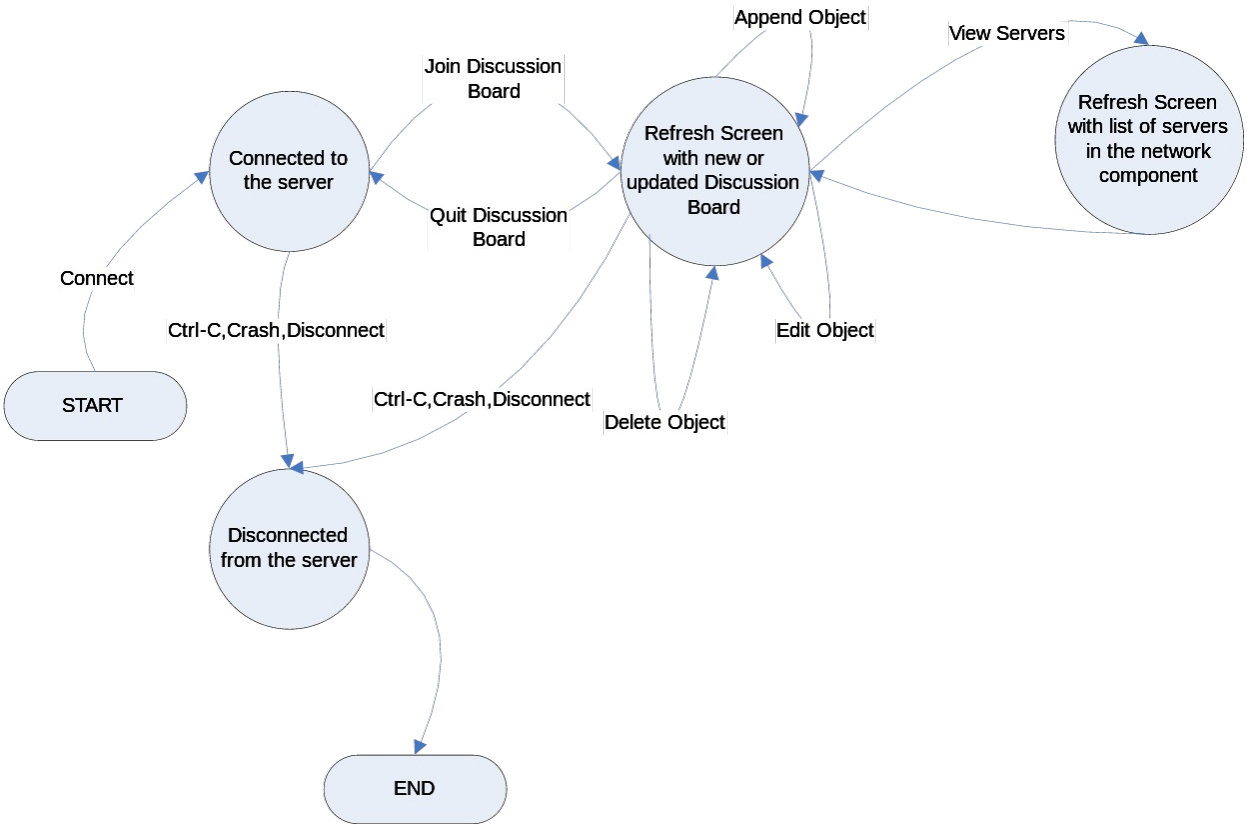
6. Print the view of the servers in the current server's network component
   Action: Client will send the message to server containing operation as ‘Print view ‘and required details as per message format

   Response to User: Once server responds with the list of the server in its current partition, the client will show the list of server in its partition to user.

b) Algorithm:

- Received Message on private group:
  - Refresh the screen
- Received Membership message:
  - If Server is disconnected
  - Join to another server

c) Client State Transition Diagram:



Server Program:

Server will serve all the above requests of clients connected to it. At the same time each server ensures that every replicated server is consistent and enough information is logged after every update so that recovery is possible in case of server crash/failure.

To ensure consistency each server has to be member of different Spread multicast groups.

a) Spread Multicast Groups:

Each server is member of below four types of Spread Multicast Group

1) *Servers_replication_group:*
   This is the group of all the replicated servers in the system.
2) *Client_server_group:*
   This is the group of each client and the server to which this particular client is connected.
3) *Server_public_group:*
   This is the group in which there is only one server. This is well-known public group so that ,to connect to server, Client can easily send message to this group.
4) *Server_private_group:*
   This is the each servers own private group. This is created when server joins Spread

Each serve can get update from its clients or other server which in turn has got the update from their clients. When there is update to server it should be ensured that it is applied to all the replicated servers in the same order. This is state machine replication and hence ensures consistency.

If there is partition, the server in each partition will act as independent set of replicated server and will keep on serving the clients who are joined to any of the server in that partition. When the partition is healed all the servers will exchange the missing updates and will come to sync once message exchange is done.

For each update at server, we maintain one log entry on disc in log file. This log entry format is given above. When we get an update from client or other server we first enter the update in log file. This log file helps to recover the system to consistent state in the case of node crash or node failure. Also when the partition is healed we are exchanging update messages from log files only to bring all the replicated servers in the consistent state in healed partition.

The abstract algorithm of the server program is given below.

b) Algorithm

- Received Update from client:
  - Write it to log file
  - Multicast to server group
- Received Update message from Server group:
  - Write it to log file
  - Commit it to database file
  - Send the discussion board corresponding to the update received to all clients who have joined this particular discussion board
  - If all servers are present in current partition; delete the entry from log file
- Received Membership message:
  - If partition took place
    - Change the leader
  - If partition healed
    - Exchange messages so that joining partition will be in sync
    - If all the servers are present
      - Delete the log file
    - Else
      - Merge the log file from both the leader
    - Select new leader

c) Server State Transition Diagram