CMSC-621, PROJECT 2(DISTRIBUTED CLOCK)
SHREYA DATE (PQ56297)

CODE DESIGN

The project Distributed lock contains the following files:

1. GringottsServer: contains the server class having methods for handling server logic and accepting multiple client requests on separate threads

2. GringottsClient : contains the Client class having logic for client to create N client threads. N which is the number of threads is accepted as a command line argument.

3. Socket: contains wrappers over the socket system calls

4. SocketDef: contains structure for socket data

5. ServerMain.cpp : This is the process which kicks off server creation by using methods exposed by the GringottsServer class

6. ClientMain.cpp : This is the process which kicks off client creation by using methods exposed by the GringottsClient class

**Note that because of heavy modularization, code workflow is pretty clear and hence not commented much.

CODE WORKFLOW

1. Server implements the socket client APIs.
2. For each incoming client request, it spawns a thread and handles communication with that particular client on a certain thread.
It queues the client id sent by the client in a transactionQueue data structure.
The client id is nothing but the socket file desc of the client sent by the client.
3. In the thread handler, the server checks if the file has access.
4. if iAccess variable is set to 1, then the thread does not have access to the file and waits for it.
5. Once access is given to the file, it sends a message to the client and client updates the file.
6. After client has updated the file, it sends a message to the server.
7. The server then removes the client from the transactionQueue and releases the lock so that other threads can access it.

LEARNINGS

Synchronization between threads is achieved.

ISSUES

1. Sometimes, client threads got locked out because of incorrect handling of access variables.
