Technical Impression

Weiran Guo

This project is built upon last one. As we need multiple servers running, and clients can access to the server at the same time, the data synchronization is important for our system to work properly. As multiple updates can occur, the order is important to make sure they are logically correct. As a result, we need a coordinator to make sure things happen in order. It takes me long time to design a coordinator. First, a coordinator need to know all replica servers’ip address and port number to perform RMI. Also, each single server can be a coordinator as the code should be run on different machine. So I design the server to know whether it is a coordinator by passing the command line arguments. The coordinator server will keep a Hashmap of replica server, it will be empty when the coordinator starts. When each new replica server launches, it will call the register RMI of coordinator, so coordinator server will know a new server is added to the cluster.

When client connect to one of server, it can call the get/put/delete RMI of that server. For get, it will simply access data from that server, but for get and put operation, things will happen differently. If the server is coordinator server, then it will commit those changes to all the replica servers, and after each commit is finished, it will return to the client and client knows update has been performed to all servers. If the server is a replica server, then it will release the lock, and commit changes to the coordinator, and let coordinator to decide when to call the RMI functionalities of itself. After coordinator receives the commit, and perform operations in order, it will call the RMI functionalities of all replica servers to perform updates. Then the coordinator will return to this replica server, so it knows commit is finished.