UNIVERSITY OF TEXAS-ARLINGTON

PROJECT-3 REPORT

Submitted in partial fulfilment of the requirement for the course-work
CSE-5306-003 DISTRIBUTED SYSTEMS

SUBMITTED BY

JAYACHANDRA JARAJAPU (1001964536)

PRUDHVI GURRAM (1001955767)

DECLARATION

I PRUDHVI GURRAM bearing UTA ID: 1001955767 and JAYACHANDRA JARAJAPU bearing UTA ID: 1001964536, of the Department of Computer Science, University of Texas at Arlington Confirm that we have neither given nor received unauthorized assistance on this work.

Prudhvi Gurram

Jayachandra Jarajapu

11/01/2022

INTRODUCTION:

The Requirement of the project is to implement a 2-PC distributed commit protocal for a distributed network. In this project we have implemented using python sockets and threading methods. There will be multiple threads connecting to the server using sockets, the server will act as a coordinator for the threads in the distributed network. We are using file logger system where each transaction is saved in the server and client log files and the files are saved on the disc. The number of nodes we used in our network is 3 and are sequenced using threading and sockets

IMPLEMENTATION OF 2-PC DISTRIBUTED SYSTEM:

In this project, we were told to implement a 2-PC distributed protocol where there exists a coordinator in the network and communicates with all other nodes in the network. We have targeted three nodes in our distributed system. In python we have a package called sockets which will be able to create a server using localhost and port number of that host. Once the host is created, we connect several nodes using threading where all the threads are binded into a socket using hostname and port address. The threads are present in the nodes of the distributed system and each node is connected to the same Ip address and port.

The program has been implemented using socket programming in python and we have used threading in python as well.

WORKFLOW:

- In this project, we are defining two python scripts where one will act as coordinator and the other file will be the nodes in the network
- Initially we define a constructor class where we define the host, port and bind the connection using sockets.
- We have defined functions for starting and accepting the connections to this socket using threading in python
- If the receiving node is true then we send a query to the node using send function and run that particular transaction using socket and save the transaction into a log file
- To save the transactions to a file we using logging package in python to store into a separate log file of server and clients

- To inject a server failure, we have implemented a condition that if the server sends a text as down to the nodes, then the server sends the nodes to all nodes expect a particular node and it crashes
- In such event the transaction is globally aborted and even if the other clients send the commit message, it is still globally aborted.
- Once the server sends a query message, the clients will have an option to commit or abort the query.
- If at least one client aborts the transaction, then all the transactions are globally aborted.
- Once after receiving the message the server will initiate global commit or abort and waits for the acknowledgement
- The clients later receive the acknowledgement and then closes that particular transaction.

ISSUES ENCOUNTERED:

- Implementing a 2PC coordinator and nodes distributed network requires that transaction coordinator should fail to show how nodes are reacting to it, making it fail held us back for a while until we came up with a solution for coordinator to fail for certain type of query we enter, a counter messaging approach is used
- To create a communication between the nodes we tried to implement in different approaches as xml rpc server, json rpc server and sockets but finally we chose sockets as it is easy to bind it with single ip and host in our pc
- Coordinators should wait for responses from all the nodes making it wait and decide maintaining the flow with the function run_transaction took ample amount of time.

Team Collaboration:

Prudhvi Gurram:

- Learnt about 2-phase protocal and how it can be implemented in python and initialize the nodes for each transaction performed
- Worked on implementing a failure for server timeout and initializing threading for performing operations between the nodes
- Learnt about global commit and abort to all nodes in cases of failure in the distributed network.
- Prepared the ReadMe file and part of report for the project

Jayachandra Jarajapu:

- Worked on implementing sockets and threads for sending the messages and listening to the communication port for all devices.
- Learnt about socket library which will help us to learn host a server and later can be used by threading to create multiple threads
- Learnt on injecting a failure where if one nodes fails to sends commit message then whole transaction should be aborted.
- Prepared the readme file and part of report for the project.

KNOWLEDGE WE GAINED:

- After implementing the project, we gained knowledge about 2pc and its drawbacks over a distributed network, if in case coordinator fails in between while running a transaction - it results node to never complete their transactions. Common expectation is that the coordinator saves or stores the node transactions somewhere where it can complete those transactions once it is again up and running
- We learnt we could only wait for one response type instead of both, where receiving the vote we found out we can wait for abort alone in a set and search for it, instead of search for both responses from node as we can have n nodes
- We have learnt about python binding of server and client (coordinator and node) can be done with one ip address and port in sockets.
- Logging appears to be easy, but we learnt we can also configure it in python as in formats we need. we also learnt about how locking works in multiprocessing, but didn't find need to use it

References:

Dubois, Julien. "RedbeanGit/2PC-Example." *GitHub*, Sept. 2022, https://github.com/RedbeanGit/2pc-example.

DB, Apsara. "Tech Insights - Two-Phase Commit Protocol for Distributed Transactions." *Alibaba Cloud Community*, Feb. 2021, https://www.alibabacloud.com/blog/tech-insights---two-phase-commit-protocol-for-distributed-transactions_597326.

uppal, shaurya. "Socket Programming with Multi-Threading in Python." *GeeksforGeeks*, 14 July 2022, https://www.geeksforgeeks.org/socket-programming-multi-threading-python/.