PROJECT 2 REPORT

Preetham Karanth Kota – 1002076418

pxk6418@mavs.uta.edu

I have neither given nor received unauthorized assistance on this work

Signed: Preetham Karanth Kota Date: 10/07/2022

1 Technologies Used:

- Designed and developed the application for the said requirement using Python
- Used XMLRPC library provided by python for remote procedure call between client and server
- Implemented the said application in Linux environment. Any Linux machine would suffice to run this application
- Used Multithreading concept

2 Things Learnt:

- Got good know how about server client model and vector clock and logical clock in general
- Got to know basics of RPC and its application and usage
- Hands on python programming language and multi-threading

3 Project

Implemented a multi-threaded file server that supports UPLOAD, DOWNLOAD, DELETE, and RENAME file operations. Use different folders to hold files downloaded to the client or uploaded to the server.

3.1 Implementation:

- Used Linux i.e. ubuntu to develop and design this particular software.
- There is a Master node which takes input like “from node” and “to node” to which message is to be sent and takes the action accordingly.
- The nodes node_A,node_B,node_C are 3 individual servers which are listening in a loop and doesn’t join or stop. Has threads to send and receive messages to each other based on the indication from the master node.
- The said node updates its vector clock and prints the vector clock before and after an event occurs at its end
- Open any Linux terminal and run the command "python3 master_node.py" to get the Master node running
- Open any Linux terminal and run the command "python3 node_A.py" to get the node A running
- Open any Linux terminal and run the command "python3 node_B.py" to get the node B running
- Open any Linux terminal and run the command "python3 node_C.py" to get the node C running
- Enter 1 to continue at Master node terminal. Enter the Sending node,recieving node and type the message to be sent when promted on the terminal
- Once done you can check the respective nodes about their vector clock printed before and after the said event happens along with the message

```
pkkota@LAPTOP-4UD2F03U:UTA_CSE_5306_PROJECT_02$ python3 master_node.py

----- MASTER NODE ------


[*] Enter 1 to RUN, 0 EXIT: 1
[*] Enter the Sending Node (A/B/C): A
[*] Enter the Recieveing Node (A/B/C): B
[*] Enter the Message: Hello A -> B
[*] [PORT NUM] Proc A: 9000  Proc b: 8000  Proc C: 7000
[*] { A }-->{ B } MSG:' Hello A -> B '

[*] Enter 1 to RUN, 0 EXIT: 1
[*] Enter the Sending Node (A/B/C): B
[*] Enter the Recieveing Node (A/B/C): C
[*] Enter the Message: Hello B -> C
[*] [PORT NUM] Proc A: 9000  Proc b: 8000  Proc C: 7000
[*] { B }-->{ C } MSG:' Hello B -> C '

[*] Enter 1 to RUN, 0 EXIT: 1
[*] Enter the Sending Node (A/B/C): C
[*] Enter the Recieveing Node (A/B/C): A
[*] Enter the Message: Hello C->A
[*] [PORT NUM] Proc A: 9000  Proc b: 8000  Proc C: 7000
[*] { C }-->{ A } MSG:' Hello C->A '

[*] Enter 1 to RUN, 0 EXIT:
```

Fig 1 : Prompt of master node taking input to direct the messages from one node to another

```
pkkota@LAPTOP-4UD2F03U:UTA_CSE_5306_PROJECT_02$ python3 node_A.py

*** Process Node A started

[*] Process Listening at PORT 9000 ....

[*] {A}-->{ B } MSG:' Hello A -> B ' Time Vector Before:  [0, 0, 0]
127.0.0.1 - - [06/Oct/2022 20:30:03] "POST /RPC2 HTTP/1.1" 200 -
[*] {A}-->{ B } MSG:' Hello A -> B ' Time Vector After:  [1, 0, 0]

[*] { C } ---> {A} MSG:  Hello C->A BEFORE Vector Node A:  [1, 0, 0]
127.0.0.1 - - [06/Oct/2022 20:30:56] "POST /RPC2 HTTP/1.1" 200 -
[*] { C } ---> {A} MSG:  Hello C->A AFTER Vector Node A:  [2, 2, 2]
```

Fig 2: Vector clock printed at node A before and after sending/receiving message from another node.

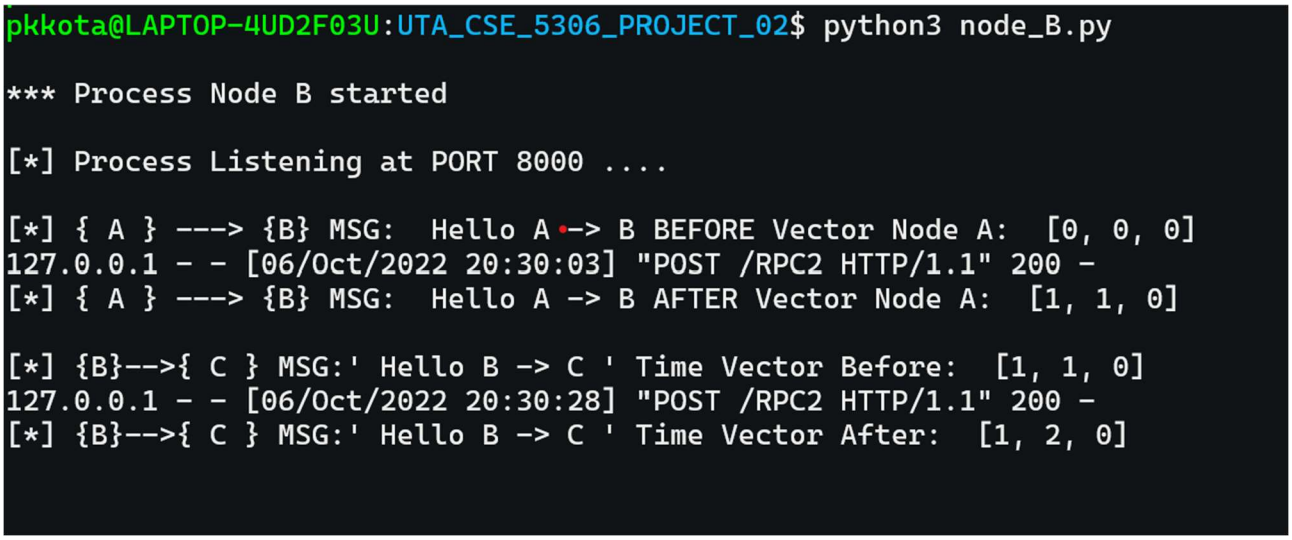

Fig 3: Vector clock printed at node B before and after sending/receiving message from another node.

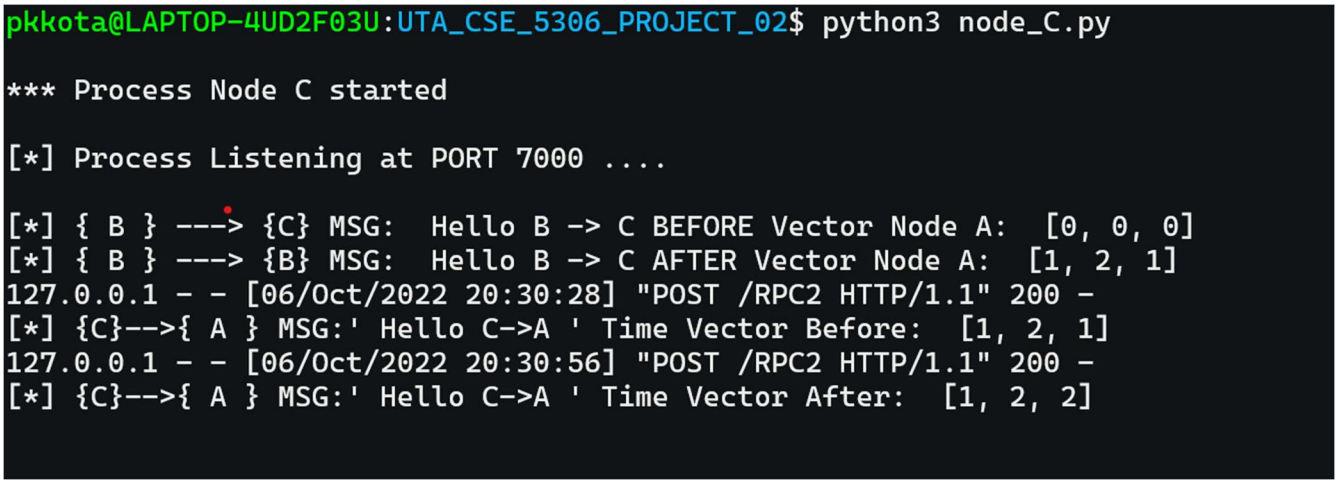

Fig 4: Vector clock printed at node C before and after sending/receiving message from another node.