COMP 7005 – Computer Systems Technology

Assignment #1 Report

Due Date: October 2, 0900 hrs.

Name(s): Kuanysh Boranbayev, Parm Dhaliwal

Objective: To implement a simple client-server using the TCP/IP protocol suite.

Overview:

A program is designed so that the Server is initially on listen mode on default port 7005. Given that a Client knows the IP or domain name of the Server with default or specified port, a Client will be able to send connect request to the Server. The Server is simply designed so that it can accept any Client with no restrictions. Server will respond to a Client's commands: GET and SEND.

1. GET command is used for to get a file from the Server and send it to Client machine.
2. SEND command is used for to send a file from the Client machine to the Server.

Server-Side Functions:

Server program is designed so that it will respond only to a Client commands. There is no direct commands implemented in Server terminal.

Once the program is executed, it will always listen on default port 7005 unless specified otherwise initially.

After a Client machine is connected to the Server, Server will wait for a command from the Client. Once, a command is received, the Server will prompt a Client for a File Name.

File Name Request is needed for both commands. In case of GET, Server needs a file name to read the contents of the file. In case of SEND, Server needs to create the same file name as in Client's.

After a File Name is received, Server will initiate new connection from port 7006 to port X specifically for file transfer mode. Once the port is chosen by Server, it will send the port no. through original connection on port 7005. Afterwards, Server will behave accordingly to the Client command.

GET: Server will try to open the file with the obtained file name. If the file exists, Server will read the contents of the file and stored them as a single string so it can be sent through buffer to Client.

SEND: Server will be on listen mode on port 7006. Once, the connection is established, Server will receive file contents as a buffer. Server will write the buffer contents into the file and store it locally.

Once anyone of these commands purpose is achieved, Server will close the file transfer connection port 7006 or X with Client. And then it responds to the next Client in queue or goes back to its initial state which is listen mode on port 7005.

Client-Side Functions:

Client program is designed so that it can request only one of these commands (GET or SEND) per execution of the program. Once the connection is established with the Server, Client will be prompted for a command to send to Server.

After command is sent, Server will prompt the Client for the file name.

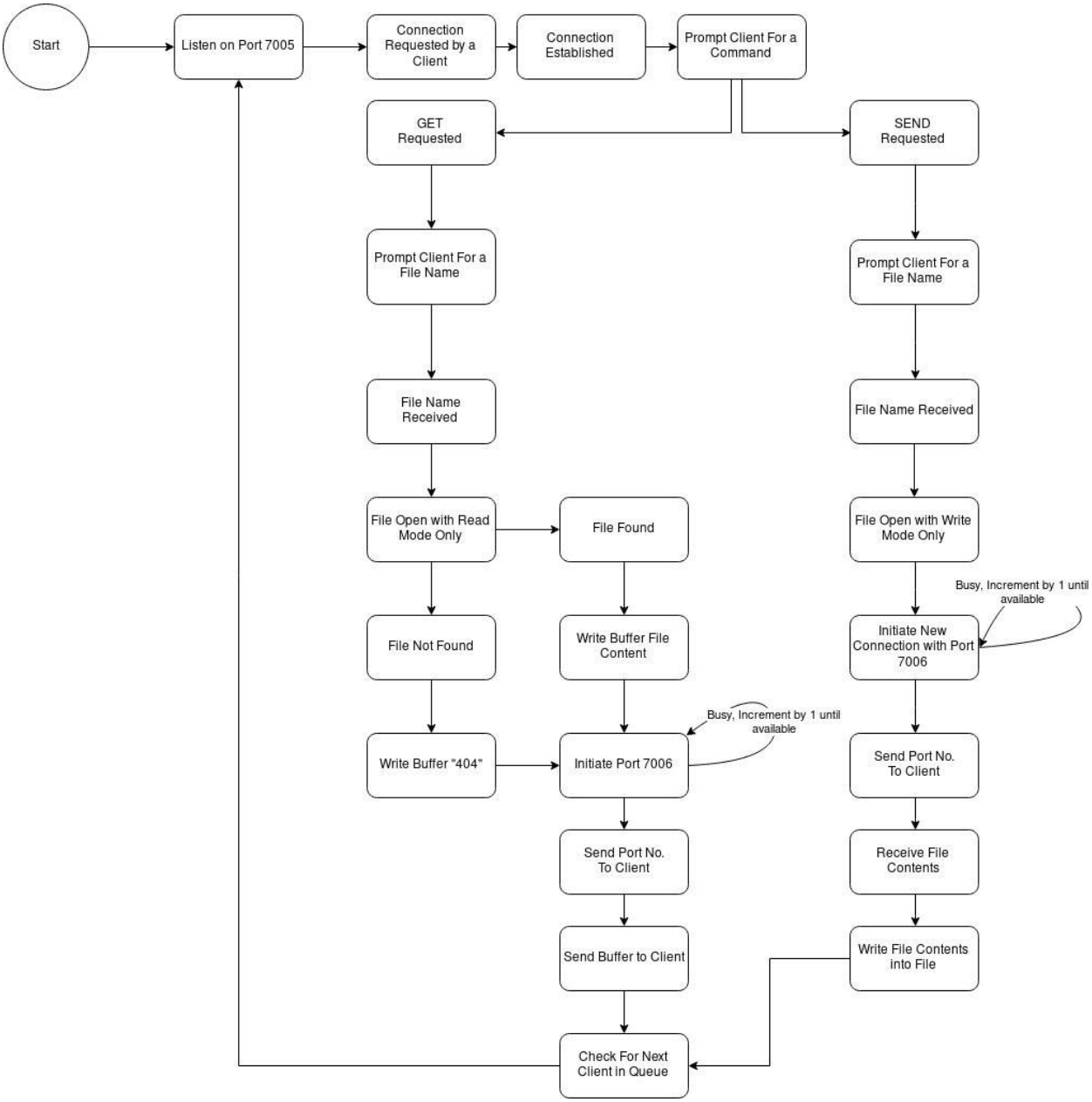
After a command is sent, Client will wait for port no. from the Server. Once the port no is received by Client, Client will try to connect to that port no. After the connection is established, Client will behave according to its initial command.

GET: Client will receive file contents as a buffer. Client will write the buffer contents into the file and store it locally.

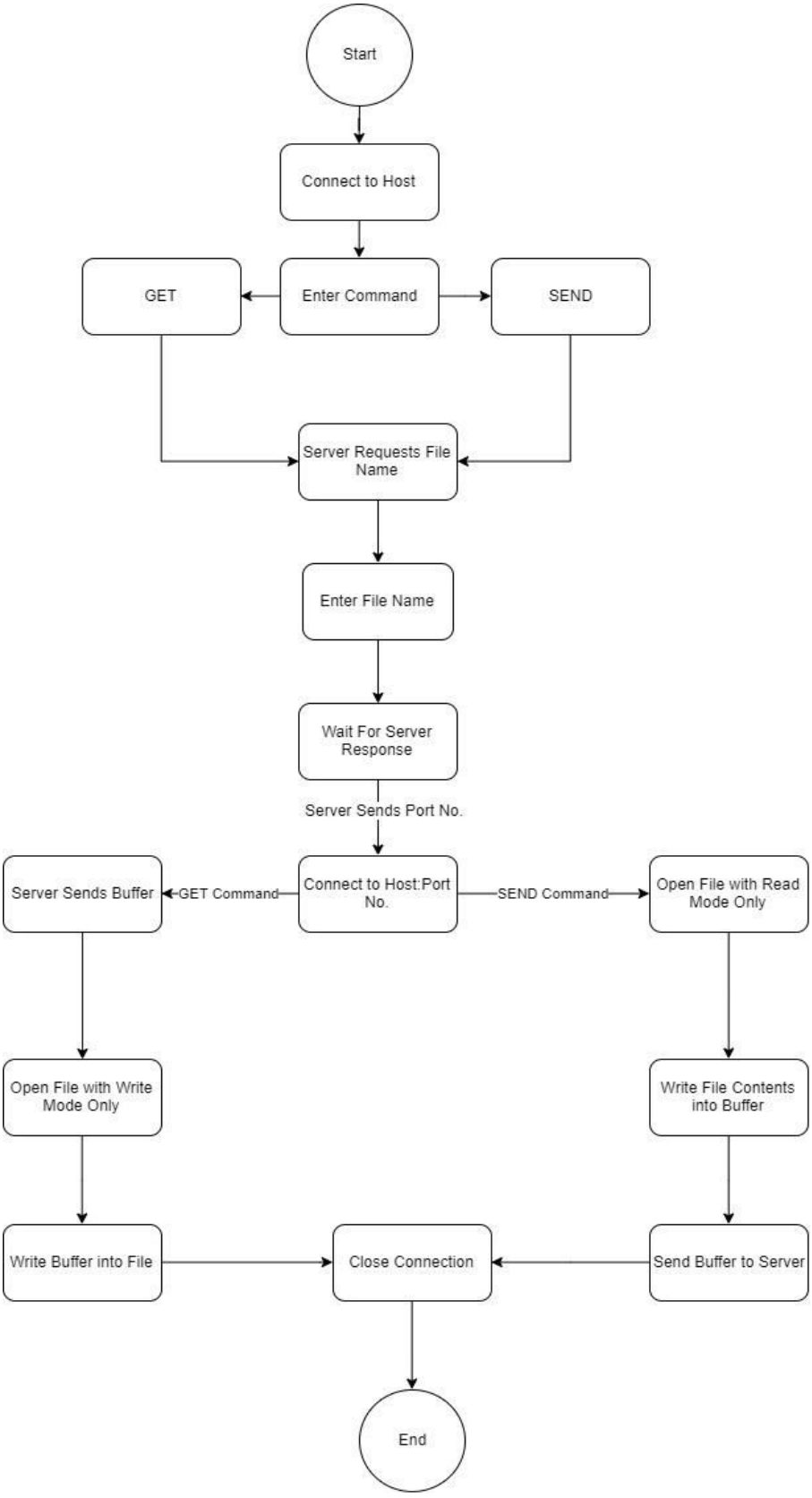
SEND: Client reads the file contents and store it as a single string so that it can be sent through buffer to Server.

After a command accomplishes its purpose, Client will close file transfer connection and then initial command connection, and then, the program exits.

Picture 1. State Diagram for Server-Side



Picture 2. State Diagram for Client-Side



Pseudocode for Server:

```
Create_socket (port=7005)
While(true)
        Listen (on port 7005)
        Accept (client)
        Save client->info
        Receive client->command
        Request file_name->client
        If (command = GET)
                Create socket(7006)
                While (port is busy)
                        Port + 1
                Send port->client
                File_open(file_name->read)
                Save text
                Close (File)
                Send text->client
        If (command = SEND)
                Create socket
                Port = 7006
                While (port is busy)
                        Port + 1
                Send port->client
                Wait(client->connect)
                Accept (client)
                Save client->info
                Wait for buffer
                File_open(file_nam->write)
                Save buffer
                Close (file)
        Close socket(7006)
Close socket(7005)
```

Pseudocode for Client

```
ConnectHost(7005)
Enter command
send (command->server)
Enter file_name
Send(file_name->server)
Receive(portNo)
ConnectHost(7006)
If (command=GET)
        Receive(buffer)
        File_Open(file_name->write)
        Store(buffer->file)
        Close(file)
        Close(socket->7006)
If (command = SEND)
        File_Open(file_name->read)
        Store(text->buffer)
        Send(buffer->server)
        Close(file)
        Close(socket->7006)
If (command = unknown)
        Enter command again
Close(socket->7005)
Exit
```