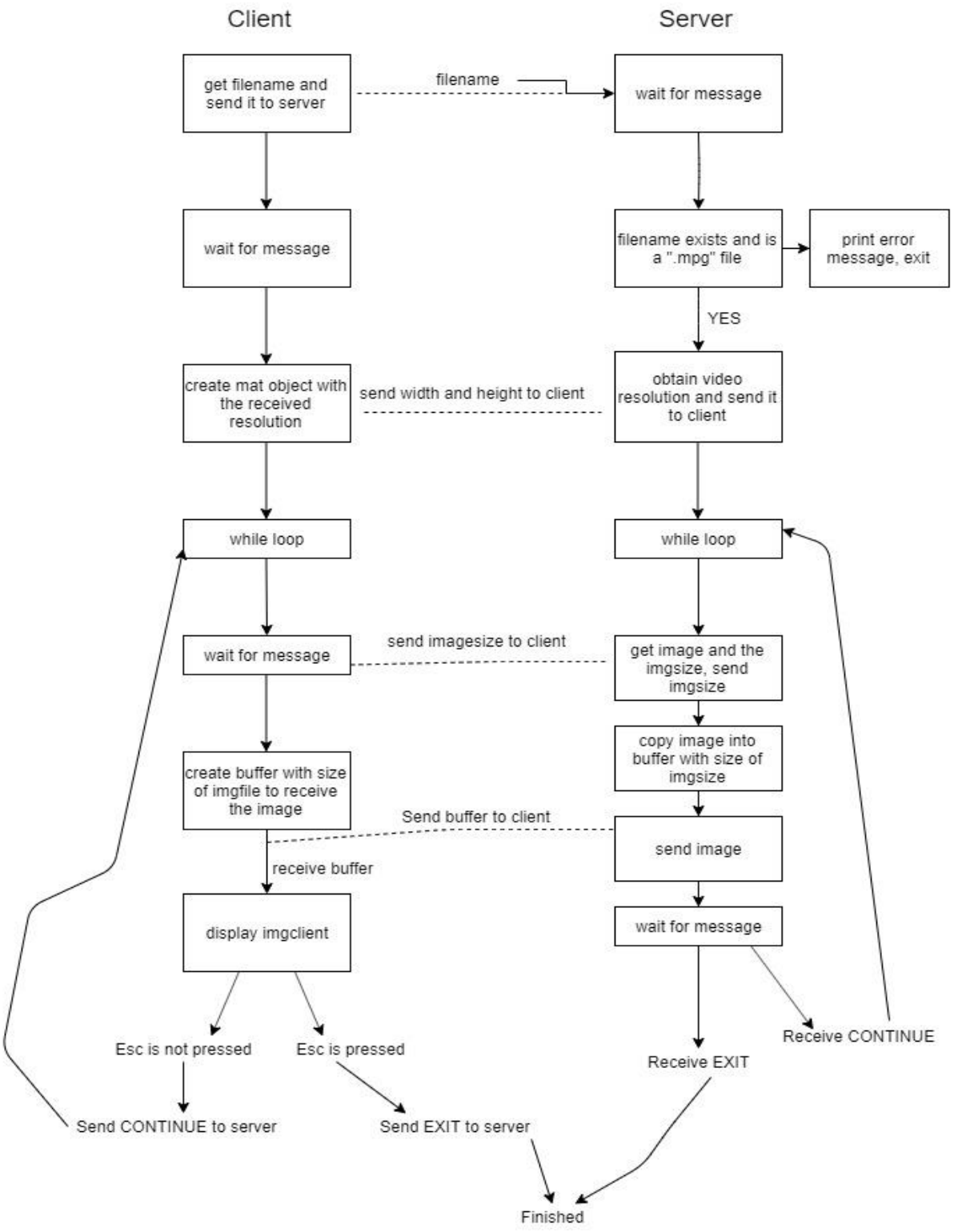
Name: Calvin Liu (劉益瑋)
Student ID: B06902100

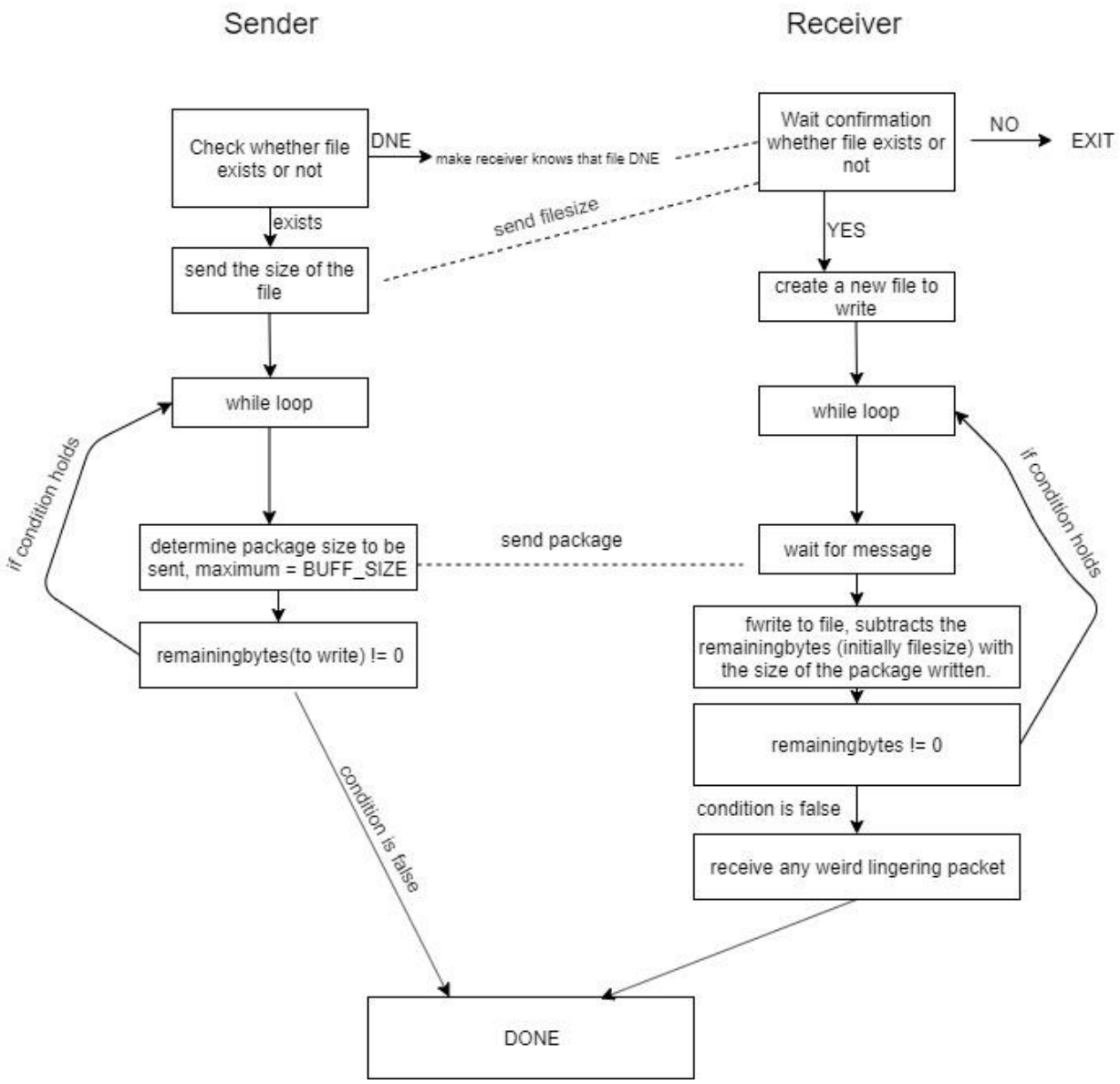
Computer Network Homework 2 Report

0. Some notes regarding the code:
   All commands are supported and should work properly. However, there might occur some bugs that I do not know of since I have not tested the code thoroughly. Server can accept multiple connection of clients. When a client is connecting or disconnecting to/from the server, information regarding the client will be printed in the server. There are also some other additional output in both client and server for the sake of clarity. When the server is terminated, the connected clients will idle in the terminal and should be disconnected manually.

1. Flowchart of Video Streaming Protocol (in next page).


Client
Server
get filename and send it to server
filename
wait for message
wait for message
filename exists and is a ".mpg" file
print error message, exit
YES
create mat object with the received resolution
send width and height to client
obtain video resolution and send it to client
while loop
while loop
wait for message
send imagesize to client
get image and the imgsize, send imgsize
copy image into buffer with size of imgsize
create buffer with size of imgfile to receive the image
Send buffer to client
send image
receive buffer
display imgclient
wait for message
Esc is not pressed
Esc is pressed
Receive CONTINUE
Receive EXIT
Send CONTINUE to server
Send EXIT to server
Finished

2. Flowchart of File Transferring Protocol:



3. SIGPIPE is a signal which is sent to a process when it attempts to write to a socket that is no longer open for reading, and the default action is to terminate the process. To ignore the SIGPIPE signal, add “signal(SIGPIPE, SIG_IGN);” in the beginning of the main function for both client and server, and this will ensure that any socket write will not be causing a SIGPIPE signal.

4. Despite the similarity, blocking I/O does not equal to synchronized I/O. Blocking means that the operations have to wait for some event to get complete. That Is, the thread is placed in wait state and will not be scheduled for execution until some event occurs. Synchronized means an activity that must wait for a reply before the thread can move forward. A blocking I/O will always be synchronous , but synchronous I/O coulde be a non-blocking, if it repeatedly check for some event to occur before proceeding for next instruction, and caller method does not enter wait stte on some event to complete.