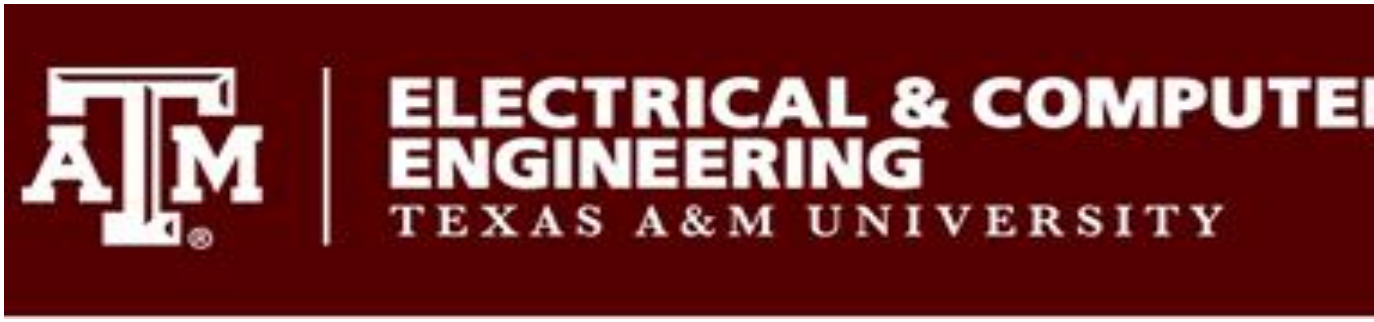



# PA1: A Client Process Speaking to a Server

Maria Dmitrievskaia

UIN: 927009911

CSCE 313 – 512

Due Date: September 6, 2020

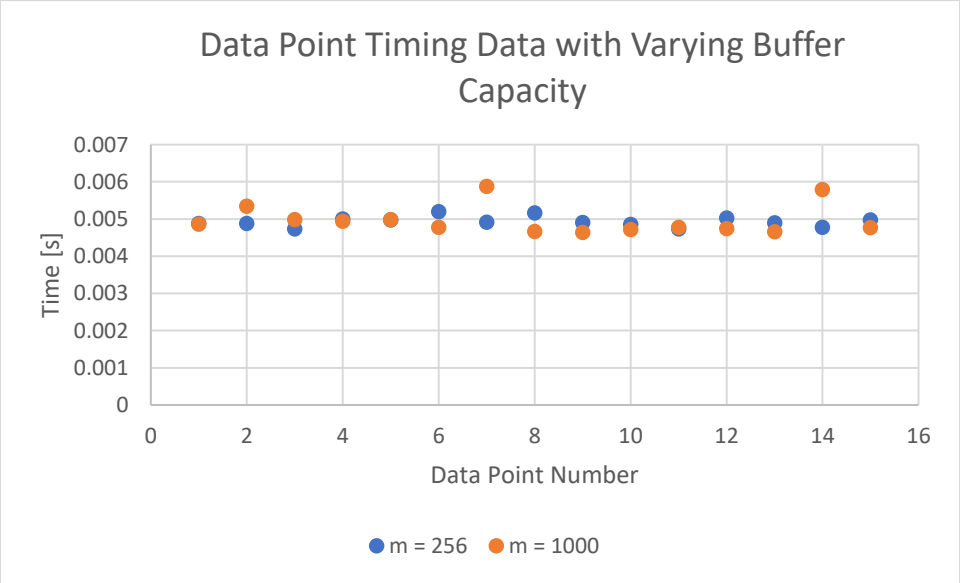
**Design:**

The written code implements client-server communication. The Inter Process Communication mechanism used to enable communication between the server and the client is FIFO pipes, which are implemented in FIFORequestChannel.h/cpp. Sending a message in either dirrection between server and client is done through the cwrite() funciton of FIFORequestChannel. Receiving in either direction is done by the cread() function of FIFORequestChannel. The server contains the heart-rate data for 15 patients over a 60 second time period. The server is able to serve these data points one at a time in response to data message requests. The server also supposts file messages, new channel messages, and quit messages.

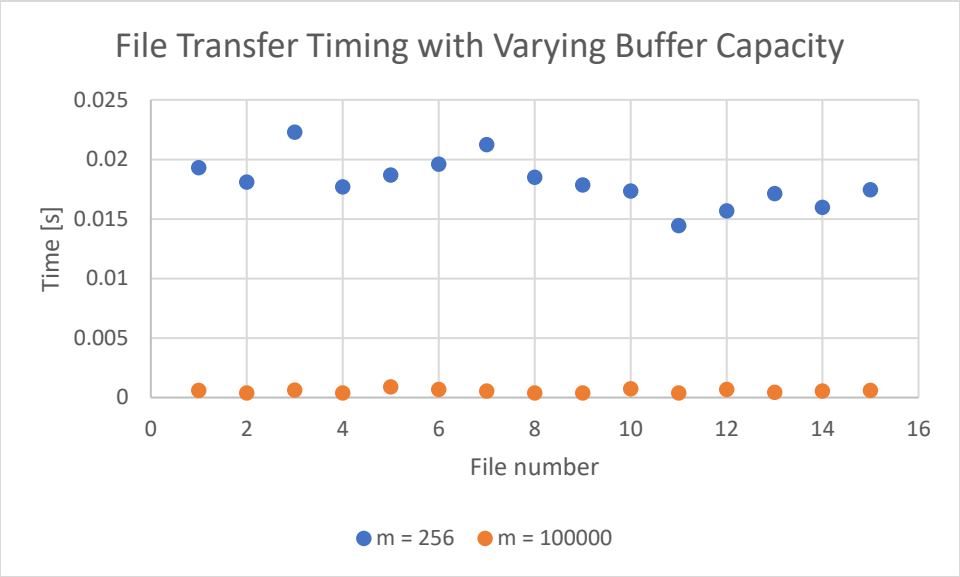
A data message is sent to te server to obtain a single data point at a specific time stamp. If no time is specified in the data message, then 1000 data points will be received from the specified person and ecg number. In order to transfer a file, the desired file must be requested. The client requests to know the full length of the file, then calculates the number of requests it will need, and then proceeds to make those requests. The new channel message creates another simultaneous IPC channel between the client and the server. Here, the server desides on an agreed-upn name for the new channel and sents it back to the client. Then, the server and client join the new channel using the new name. The quit message is the only message that the server does not reply to. Both the server and the client terminate after this message.

This code also implements running the server as a child process though fork() and exec(), so that two terminals( one for client and one for server) are not needed.

**Data Points Timing Data:**

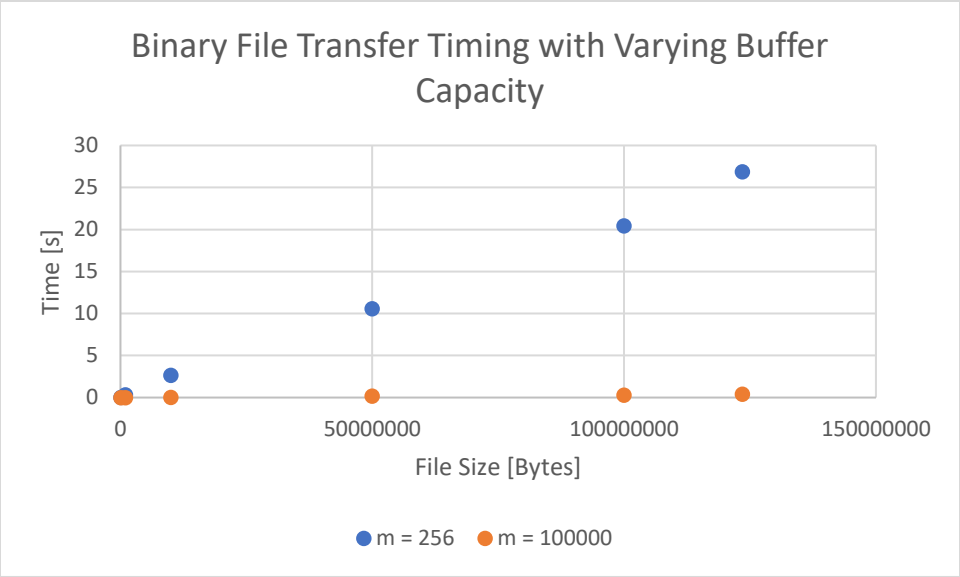


**Data Points Timing Data:**




| file number | Time [s] for buffer size m = 256 | Time [s] for buffer size m = 100000 |
|---|---|---|
| 1 | 0.019304 | 0.000625 |
| 2 | 0.018097 | 0.000393 |
| 3 | 0.022302 | 0.000635 |
| 4 | 0.017698 | 0.000404 |
| 5 | 0.018697 | 0.000899 |
| 6 | 0.019622 | 0.000698 |
| 7 | 0.021241 | 0.000549 |
| 8 | 0.01852 | 0.000393 |
| 9 | 0.017871 | 0.000389 |
| 10 | 0.017352 | 0.000749 |
| 11 | 0.014453 | 0.000403 |
| 12 | 0.015699 | 0.000683 |
| 13 | 0.017137 | 0.000449 |
| 14 | 0.015996 | 0.00056 |
| 15 | 0.017449 | 0.000606 |

**Binary Files Timing Data:**




| Binary File size | Time [s] for buffer size m = 256 | Time [s] for buffer size m = 100000 |
|---|---|---|
| 100000000 | 20.425 | 0.314028 |
| 10000000 | 2.6434 | 0.0327 |
| 1000000 | 0.33089 | 0.008088 |
| 100000 | 0.034033 | 0 |
| 123456789 | 26.85597 | 0.424125 |
| 50000000 | 10.56086 | 0.196181 |

The bottleneck for transferring files is the buffer capacity. With increasing the buffer capacity, we can decrease the amount of time it takes to transfer a file because there are less messages needed to send back and forth between the server and client and larger amounts are data are grabbed at a time.

YouTube Video Link:

https://youtu.be/88mhX5ziFmM

Note: This was my first time recording and I could not get my sound to work, even after trying to record the video several times. I will try to have my sound working for the next PA.

## Resources

The following resource was used to implement the gettimeofday function:

https://www.geeksforgeeks.org/measure-execution-time-with-high-precision-in-c-c/

The following resource was used to implement the wait() function:

http://www.cplusplus.com/forum/general/123529/