Ryan Russell

Dr. Ahmed

CSCE 313.503

4 October 2020

Programming Assignment Three Report

Description:

When came to the design of this programming assignment, the base code that was used was the code from programming assignment one that Dr. Tanzir provided. Then, the client and server.cpp files were changed to be able to implement two additional inter process communication methods (Message Queue and Shared Memory) besides FIFO. This was done by creating an abstract class called RequestChannel which the three inter process communication methods were sub-classes to. This abstract class would contain a constructor, destructor, the Side and Mode enums, and the cread, cwrite, and open_ipc methods which all three of the sub-classes could provide their own implementation. Adding these two additional methods into the client and server.cpp files was done by updating the given FIFOreqchannel header and cpp file so that it could work with the abstract RequestChannel and creating header and cpp files for the MQreqchannel and SHMreqchannel classes.

The MQreqchannel's constructor focused on using the open_ipc for two sides with both being given the ability to create, read, and write. The open_ipc method for this class focused on used the struct mq_attr and then using that to open a file descriptor which would be the return value for the method. The destructor closes these two sides and then unlinks the names they were given. As for the SHMreqchannel, the constructor creates two SHMQ objects (which is defined in the SHMreqchannel header file) and then swaps the two if it is on the Client side of the process. The destructor is simple as it just destroys these two objects and there is no open_ipc method used in this method.

Data Collection and Comparison:

IPC Method	Requesting 1K Data Points	Requesting 1MB file split into 50 Channels
FIFO	1.40138*10^7 microseconds	3.75645*10^6 microseconds
MQ	1.41585*10^7 microseconds	2.48167*10^6 microseconds (could only do 48 on this one)
SHM	1.38982*10^7 microseconds	2.11862*10^6 microseconds

The FIFO method consists of data being stored in buffers in the memory instead of never being written to the disk. This is done so that the overhead of disk I/O is evaded. FIFOs also do not go away even if you do a system reboot (must be manually cleaned up) and data must be collected in a first in first out manner. However, FIFO requires that a name for a channel is agreed upon ahead of time so that communication can be executed, and multiple threads can cause problems. The Message Queue method is similar to FIFO as multiple number of processes are also allowed to send and receive from a single pipe of communication. In this case, it is a single message queue and the message is passed in as blocks of a certain size. This is different from FIFO as it passes its messages through streams. Also, message queues can collect data in any order, is bidirectional, and the writer/reader processes do not have to wait for the other end to do its job. This is why Message Queues are around the same speed or faster than FIFOs because there is no wait time for the other process to execute. The Shared Memory method is the fastest form of communication out of the three as it shares data through a segment. This segment can be mapped or unmapped by the processes can be read or written to by more than one process. However, synchronization is now up to the responsibility of the programmer to make sure that one process is not messing with another one's data. There is also no write or read functions and semaphores objects are used. These semaphores help the synchronization since they are visible to all processes, but they do not share the address space that other methods of communication might.

Questions:

- Are there any difference in runtime? Can you explain the reason?

  Data collection runtime was pretty similar with only minute differences in the amount of time each method took to execute. However, the amount of time that it took for file transfer to execute differed by each method due to the fact that some of the methods are meant for short forms of data and not for long forms of data.
- How many request channels can we have when we are using MQ and SHM? Are these number any different from that in FIFO?

  We can have ten request channels using MQ and four using shared memory. These numbers are different from that in FIFO because the number of request channels using FIFO is based solely on the amount of memory available.

Demo Video:

https://drive.google.com/file/d/1Ldrf-DyCnFrJRRaXyv2DZFGj54afR3mq/view?usp=sharing