TDQC

Relay

Jack Spence

18 Jan 2019

1. Write-Up

1.1 Requirements

Requirements were to create two programs that will communicate across processes. While not restricted to a particular method, it became apparent that network sockets were the easiest way to accomplish this. Another requirement be that the programs utilize data specified in the environmental variables for the shell the programs are launched from.

1.2 Suggested Features

Suggested feature attempted in my submission is to implement the “-l” flag to limit the amount of concurrent listener sessions as well as to allow multiple concurrent listener sessions.

1) make || make profile || make debug || make listener || make dispatcher

2) ./bin/dispatcher [-l <Client Max #>]

2. Project Design Plans

2.1 Initial Design Plans

My initial design plan was to use a character array in shared memory as my inter-process communication medium. While in theory this would work, it became a bit more than I was willing to chew rather quickly. My idea was to have this array in shared memory that all the processes could attach to with the RELAY environment variable be used to generate the key required for shared memory access. As well as use named semaphores to synchronize the processes.

2.2 What didn't work

Hardly anything from the initial design plan worked. The furthest I had got, was having the dispatcher successfully send a message and the listener print it. After the print both processes went into infinite loops. The synchronization fell apart as fast as I could understand what was happening. It became apparent that my plan was far less than ideal.

2.3 What went well

This project is my first ever implementation of network sockets in any language. I feel as though using them for this project was kind of a no-brain er. Network sockets were significantly easier to communicate between processes than the other methods recommended in the rubric. Also with this implementation, adapting the code to work across an actual network wouldn't be that hard.

2.4 Conclusion

This project was pretty daunting at first, I felt this class was over a lot of subjects that I couldn't quite wrap my head around as far as coding the ideas being taught. So when it came time to write out some C code to implement a few of the ideas, I felt pretty lost. This caused me to dig into a lot of man pages about semaphores, shared memory, and later sockets. This was also the first project where I found myself getting more answer from man pages than stackoverflow, which I feel is a plus.