Operating Systems

Program 3

Distributed Interprocess Communication tool

by Zach Owen

Program Description: This program emulates a shared memory communication system. Uses a mailbox system where users can store strings of data and retrieve them at will. Is able to dynamically bind and create mailboxes at runtime. Program is written in C using POSIX for threading and Semaphores. MUST RUN ON A 64 BIT MACHINE

Libraries Used:
stdio.h: Standard input and output

stdlib.h: Used for atof()

string.h: Used for strcpy, strcat, and strlen
sys/socket.h: Contains headers for sockets
arpa/inet: Ease of address translation
unistd.h : Needed for write() for socket communication

sys/types.h: Contains Shared memory functions
sys/shm.h: Contains Shared memory functions
sys/ipc.h: Contains Shared memory functions

Program flow: Program Begins by parsing and creating mailboxes in shared memory as well as all of the tools to be used to manipulate them. It also connects itself to a socket for communication. After setup is complete and the process forks to the background. At this point it creates a file containing its PID so that the process can be easily killed by dipcrm. And then waits for connections. When a connection is established the program creates a thread to handle the connection and the main thread returns to a waiting state to accept more connections. The thread then waits for the user to command it to read, write, or quit and will exit when finished.

Functions:

Main(): Sets up Shared memory and mutexes and starts accept loop

count(): Functions that allow to keep track of quantity of Readers in a mailbox

AcceptClient(): Readies a socket to thread and waits for a connection. Calles CommThread() when connection becomes established as a POSIX thread. Then exits back to main()

CommThread(): Thread Control function, Gets input from user across socket and parses and executes commands.

ClientRead(): Read Command, reads data in mailbox. Protected with doubly locked Mutexes

ClientWrite(): Write Command, writes data to mailbox. Protected with doubly locked Mutexes

CreateM(): Creates array's of mutexes for mailbox protection

CreateStable(): Creates Mailbox shared memory segments. Starts at keyvalue 1337

Make_Token(): Function copied from dsh program, returns copy of a section of the input string

Parse(): Takes a command given to CommThread() and determines what command it is and calls the correct function to deal with that command.

Setargs(): Function to place command line parameters from argv into their proper places as well as typecasts them to a useful format

SetupBind(): Function to bind the server to a socket

Tokenize(): Copied from dsh program, takes a string of input and returns the tokens in a reasonable useful format

Testing:

Program was altered to facilitate testing. Sleep() statements were added to critical sections in both read and write sections of shared memory. Program showed staunch defence against Deadlocking, and resistance to Starvation, although if enough read threads are placed in a single box, followed by many writes, while the box does not corrupt some starvation can occur. Though it is mitigated by the fact that new read threads are not allowed into boxes with Writers waiting.