CIS 415 Operating Systems

Project 2 Report Collection

Submitted to:

Prof. Allen Malony

Author:

*<Karma Woeser>*

*<UO ID - kwoeser>*

*<Duck ID- 951872986>*

**Project 2 Report**

**Introduction**

*This project is a Master Control Program (MCP). It is a process management program. The program reads from an input file and executes the commands from the file. The Master Control Program manages the process execution flow. There are four parts to the MCP.* *As of now, it only allows the user to read from a file. The project is for the CS415 – Operating Systems class.*

**Project Details**

*For this project, I had to learn more about processes and how they work. Including process scheduling algorithms like Round-Robin. I had to learn about system calls like fork() and execvp() for process creation and execution. Most of the research done was for system calls, process management and scheduling. Code from labs and the previous project 1 helped in finishing this project. I ran into many issues with part 4 as it was difficult getting the PIDs from /proc.*

**Implementation**

*The code snippet below show my implementation of my execute function. Execute() is a helper function that reads commands from the file then uses the system calls fork() and execvp() for process creation and execution.*

void execute(pid\_t \*processes, FILE \*fp, sigset\_t sigs) {

    char line[ARG\_SIZE];

    int current\_process = 0;

    // Read lines in file, fork, and then execute.

    while (fgets(line, sizeof(line), fp)) {

        char \*args[256];

        token\_line(line, args);

        // Fork a new process

        processes[current\_process] = fork();

        if (processes[current\_process] < 0) {

            perror("Error during forking process");

            exit(EXIT\_FAILURE);

        }

        // Child process

        if (processes[current\_process] == 0) {

            // if command fails execvp returns -1

            int sig\_i;

            sigwait(&sigs, &sig\_i);

            if (execvp(args[0], args) == -1) {

                perror("Error during execvp process");

                exit(EXIT\_FAILURE);

            }

        }

        // Next process

        current\_process++;

    }

}

*Figure SEQ Figure \\* ARABIC 1: Some Algorithm implementation*

**Performance Results and Discussion**

*The Master Control Program (MCP) dealt with some issues with memory and part 4. The system seems to efficiently handle running the program for most of the parts. This project used system calls, scheduling algorithms, and signals. Thankfully I never ran into serious issues during the implementation of the project. I ran into issues with SIGALRM since I couldn’t register the signal correctly. The program would run but the alarm wouldn’t go off. After some research online I figured out that I had another issue with SIGWAIT that I had to resolve which also fixed my SIGARLM. Part 4 also had issues about not being able to grab certain PIDs and print them. I couldn’t fully fix this issue but most of the PIDs will print.*

**Conclusion**

*In conclusion, The Master Control Program (MCP) project is a process management program split into 4 parts. Implemented using scheduling algorithms, system calls, and signals. I ran into many problems during implementation due to how new a lot of the topics were. Overall, this project helped me see the importance of scheduling processes and process management.*