Assignment 2, Part 3: Shuttle Terminal

COP4610

Taylor Brockhoeft

Justin Block

Design

This program simulates an airport terminal. It supports dropping off and loading passengers at each terminal. This is done within a LKM (Loadable Kernel Module) via system calls. This allows us to manipulate the Kernel Module remotely from anywhere in the system, just by calling any of our pre-defined system calls:

start\_shuttle: Starts the shuttle sequence. Passengers can be queued or not queued.

stop\_shuttle: Drops remaining passengers off, without picking up any more, and then finally stopping the system thread.

issue\_request: Issues a single request for a passenger with a type, initial terminal, and destination terminal.

We store all the data for handling the shuttle and terminals within the files Shuttle.c and Shuttle.h. We also have Passenger.c and Passeneger.h which are built to handle the passengers. Terminal.c is the module file.

We use a linked list data structure to store all of our passenger data, we also keep corresponding counts for the passengers at each location that are updated as passengers are loaded and unloaded between the shuttle and terminals.

Our scheduling algorithm is more or less the C-SCAN algorithm. We dubbed it the typewriter algorithm before we realized this. It is not the most efficient, however, it gets the job done and works much like a realistic terminal, which stops at each terminal.

Schedule Algorithm:

For our project, we decided to use the typewrite algorithm, or, a c-scan. Basically, Our shuttle starts in the middle of the terminals and progressively moves in one direction. Once we are at the end of the terminals, we then move to the beginning and start again. This continues until the shuttle stops. ( The Shuttle moves in this continuous order: 3,4,5,1,2,3,4,5,1,2,3,4,5..... )

Development Journal

09/21: First day of development, hashing out project, allocating responsibilities.

10/05: Compiling the Kernel, had issues with the lab computers. Ended up moving to our personal computers due to lost time/Pre-development for Shuttle/terminal classes.s

10/12: Got STUB System call links working. Pre-development planning for Passenger classes.

10/19: Continual Independent Development of Shuttle/Terminal and Passenger code

10/26: Wrote Scheduler as well as a lot of testing to ensure operations we're being performed and passengers we're being moved between terminals appropriately.

10/29: Gathering materials for final submission, fixing last few minor issues.

Division of Labor

Taylor: Building Module, Stub system calls, Linking system calls, Cleanup, Shuttle/terminal class, Scheduler, Userspace Interaction, Readme, Report.

Justin: Thread Manipulation, Mutex handling, Passenger class, Issue Requests, Load/Unload passengers, Readme, Report.

Missing Functionality

None!

Bugs/Issues

* Issue[FIXED as of 10-29-14]: Currently, the count for seats/available has a minor counting issue. We believe this stems from the conversion of seat values of each passenger type. The face that the kernel doesn't handle float output made this difficult.

Suggestions for Assignment Write-up

Justin: I think the assignment was a good though long one. I learned a lot from it though I wish it wasn't as extensive and all the information was given up front.

As we started early we actually got locked up because a lot of the information we needed was not all given from the start. Being trickled information was unfortunately a big time consumer because we had to ask and find ways that were later given to everyone who waited towards the end. I also think the weights should be fixed in the future. The locking information was well explained in the slides but it would have been nice to have access to the lab computers and able to compile the kernel from the getco.

Taylor: I think that this assignment was, in the end a good one. I learned a lot on how to implement system calls to talk to other processes, which is awesome.

We started fairly early, and I feel like that handicapped us. There was a lot of information released in slides 4 and slides 5 of the project that laid things out pretty clearly that we figured out the hard way in the beginning. Students who started later we're kind just given this information, and didn't have to do as much searching. Also, I think the values for the passenger weights should be 1,2, and 4 (Rather than .5, 1, and 2). I found out the hard way, after a few hours, that the kernel proc files don't like to print floats. Initially, the lab computers had an issue that prevented us and many other teams from working on those computers. In the future, this should be squared away before the project starts. We lost almost 4 days of work because of that.