15-395 Lab 5
Process Tools

Objective

This project is designed to give you experience developing system tools using the */proc* file system. It might also be a good opportunity for you to show off you Perl or shell scripting, or ability to use make.

Times of Interest

Assigned: Tuesday, November 27th, 2007
Due: Tuesday, December 4th, 2007

Part 1: Process Status

This part of the assignment asks you to implement a tool similar to the usual *ps*. Before beginning this part of the assignment, please "*man ps"* and give it a try.

As with *ps*, the output should be nicely formatted. And include a header, unless otherwise specificed on the command-line. Given a *pid* it should produce a single line summary of the process's vital statistics:

- command name, e.g., argv[0]
- state, e.g., S, R, T, Z, W, etc
- pid
- ppid
- uid
- pgrp

The following flags should be supported:

- no flag, reports on all processes using the same terminal as the *ps*, itself
- -a, reports on all processed on the system
- -u *uid*, reports on all processes owned by user with *uid*
- -u *username*, reports on all processes owned by *userid*
- -p *pid*, report on only the process with pid *pid*
- -n, omit the header

SPECIAL REQUIREMENT: Please do this twice, once in shell script and the other in the 1st class programming language of your choice.

Part 2: top

This part of the assignment asks you to implement a *top*-like to for summarizing system activity. Before beginning this part of the assignment, please "*man top"* and give it a try. Unlike the standard *top*, your version need not be interactive. For those who would like to make it interactive, "*man curses"*.

The report produced by your *top*, should have two sections. The top section should report system summary information. The next section should list the ps-style information, presented in sortd order, for the top *n* resource consuming processes. All information should be well formatted with labels and/or headers, as appropriate

The system-wide information should include the following:

- loadaverages, for 1, 5 and 15 minutes
- uptime, since the last system boot
- memory information:
    - total size
    - total used
    - total free
    - swap size
    - swap free
- processor load: jiffies of user time, system time, and idle time
- accesses per active ide disk or SCSI controller

The per-process information should include the following:

- process name (argv[0])
- userid of owner
- total available memory size
- process's time in user mode
- process's time in system mode
- process's total time

The following flags should be accepted:

- -n, number of processes to list, if unspecified, 25 is assumed
- -m, sort by memory size
- -t sort by processes total time (assumed, if neither of –t, -s, or –u are specified, or if more than one is specified)
- -s, sort by processes system time
- -u, sort by processes user time
- -r reverse the order of the sort

Part 3: Common Sense

If your project contains code which requires assembly (compiling, generation, 7c), it must include a *Makefile* that can build either of, or all of, the tools in your suite. This file should be well-written, represent all of the dependencies, and should include a *clean-rule*.

This project contains significant common functionality among the parts. As appropriate. This functionality should be factored out and placed into some type of utility library. Significant common code should not be re-written.