INTRODUCTION (CS422)

Ken Wong
Washington University, St. Louis

kenw@arl.wustl.edu
www.arl.wustl.edu/~kenw

CS422

- What It IS NOT About
  - Programming languages
  - Kernel programming
  - OS survey
- Assumptions
  - You can write a good program in C/C++
  - CS422 is a software systems course
- Some Principles
  - "It's impossible to learn much by simply sitting in lectures ..." (Richard Feynman)
  - We learn by exploring the boundaries as well as the core ...
  - Real systems don't always operate as expected
  - Programming is trivial; good programming is much, much harder.

MISCELLANEOUS

- See Web Page for general information.
- Follow CS422 link: http://www.arl.wustl.edu/~kenw
- Books
  - Stallings, Operating Systems (Required)
  - Stevens, Advanced Programming in the UNIX Environment (Optional)
- Grading
  - 40% Exams (2)
  - 40% Labs/Project(s)
  - 20% Homework/Quizzes (5-7)
- Prerequisite: CS 342S (Object-Oriented Software Laboratory); C/C++
- Fillout Survey and return to instructor.
- Computer Accounts: CEC Unix systems, Sever 214

ASSIGNMENT

- Read Chapters 1-2
- Send me email in the following format:
  YOUR Name (lower case, last name first, no spaces)
  |  
  |  
  Tab | Tab  
  |  
  |  
  |  
  |  
  |  
  --- YOUR Email Address
  
  CS422  wong, ken  kenw@arl.wustl.edu
- Put "cs422" in the Subject part
- Put the information in the body of the email as plain text
  - Do NOT mime-encode the line
  - Do NOT send an attachment
NAD (Not Another Device) (1)

- A Special Device
  * Periodically delivers \( N \) longs from a set of remote sensors

**Software Architectures**

<table>
<thead>
<tr>
<th>Special Program</th>
<th>Single-User</th>
<th>Multi-User</th>
</tr>
</thead>
<tbody>
<tr>
<td>Device Driver</td>
<td>Library</td>
<td>System Calls</td>
</tr>
<tr>
<td>Device Driver</td>
<td></td>
<td>Device Driver</td>
</tr>
</tbody>
</table>

- Layers of Abstraction
- Control and Manage Resources

**An Operating System**

- OS: A collection of system programs which allow the user to run applications
  * Provides users with an abstract machine (hides some details)
  * Logical resources and well-defined operations on those objects

- Supporting Layers of Abstraction means
  * Simplified Programming
  * Isolation/Protection
  * Efficiency
  * Reliability

- Resource Management and Control
  * Provides mechanisms and policies for the control of objects/resources
  * Controls how different users and programs interact

**Physical Resources**
AN ADVANCED APPLICATION

OPERATING SYSTEM FACILITIES
(An Overview)

- Processes
  - Process Control: exec, fork, and waitpid
- Filesystem
- Input/Output
  - Buffered I/O and Unbuffered I/O (Streams and File Descriptors)
  - Unix: Standard Input, Standard Output, Standard Error
  - Access through system calls
- Communication
- Signals
- Timer
- Initialization

PROGRAMS AND PROCESSES

```c
#include <unistd.h>
#include <stdio.h>
int main (int argc, char *argv[]) {
    printf ("Hello, I’m process %d with %d parameters\n",
            getpid(), argc);
    exit (0);
}
```

- **Program**: An executable file residing on disk
  - A partial machine image
- **Process**: An executing instance of a program
  - Has a run-time state

```
solaris> gcc -o foo hello.c
solaris> foo
Hello, I’m process 7164
```

UNIX PROCESS CONTROL

- Putting a program to sleep
  ```
  > vi hello.c
  > z
  # control-z
  > jobs
  [1] + Suspended vi hello.c
  ```
- Background Execution and Signals
  ```
  > foo > foo.out &
  > jobs
  [1] + Suspended vi hello.c
  [2] + Suspended foo > foo.out
  > kill %2
  Job 2 terminated.
  ```
- Concurrent, Communicating Processes
  ```
  ps -el | more
  ```
INTERPROCESS COMMUNICATION

Client → Well Known FIFO → Client → Server → Client

UNIX FILE SYSTEM

Superblocks → Inodes

Cylinder Track Block

Data Blocks → Free list

Inodes

Superblocks

 usr

vmunix

bin

tmp

home

ls

kenw

LIFE OF A UNIX fread CALL

- A Unix file is a byte stream

fread ( strptr, sizeof(char), nitems=40, stdout )