William Paterson University of New Jersey
Department of Computer Science
College of Science and Health
Course Outline

I. Course Title: CS-405 Systems Programming 3 credits

II. Course Description:
The course familiarizes the student with the organization, system libraries, and tools for software development in the Unix system. The student should leave this course with the ability to use system level facilities provided by Unix.

III. Prerequisites: CS 345 with a grade of C- or better.

IV. Course Objectives:
To introduce the key techniques of processes and interprocess communications that are essential to distributed client/server computing. This implies the following:

1. Understanding of program execution, system calls and library functions in a Unix environment.
2. Understanding of a Unix executable file format, system memory and process memory.
3. Introduction to the traditional Unix I/O functions.
4. Understanding of properties of a Unix file and all the functions that operate on files.
5. Understanding of Unix processing environment and Unix process control.
6. Understanding of interprocess communication techniques for process coordination, for information sharing, and for client/server applications.
7. Introduction to threads and multithreaded programming.

V. Student Learning Outcomes:
After the completion of this course, a successful student will be able to do the following:

1. Explain the difference between system call and library functions
2. Explain the use of the Unix manual pages to get information about system calls and library functions.
3. Create and execute programs with command-line arguments
4. Describe Unix executable file format, system memory, and process memory.
5. Use Unix I/O functions to perform file input/output in programs
6. Describe the properties of a Unix file system and be able to use functions that operate on files in a program.
7. Write programs that generate processes and that also access the environment of those processes.
8. Write programs that use lock files, and locking files for process cooperation.
9. Write programs that use pipes, message queues, semaphores, and shared memory for process cooperation and communications.
10. Write programs that use RPC and sockets for process communications.
11. Write programs with threads.

The course will also reinforce the following students learning outcomes of the university:

a) Effectively express themselves in written and oral form. Measure: exams & projects.

b) Demonstrate ability to think critically. Measure: exams and projects.

c) Locate and use information. Measure: projects.

d) Demonstrate ability to integrate knowledge and idea in a coherent and meaningful manner. Measure: exams, surveys, and projects.

VI. Course Contents:

1. Programs and Processes
   - Introduction: multiprogramming, source program, and executable program
   - library functions, system calls, program linkage, executable file format, and managing failures.
   - process, Unix system memory, process memory, and the U area.
2. Unix File Input/Output (Chap 3 in Advanced...)
   - File descriptors, and open, creat, close, lseek, read and write functions.
   - I/O efficiency, and file sharing.
   - atomic operations, dup, dup2, fcntl,and ioctl functions, and the /dev/fd directory.
3. Unix Files and Directories (chap 4 in Advanced . . .)
   - stat, fstat, and lstat functions
   - file types, set-userID, set-group-ID, file access permissions, and ownership of new files and directories.
   - access, umask, chmod, fchmod, chown, fchown, and lchown functions.
   - file size, file truncation, filesystems, and the link, unlink, remove, and rename functions.
   - mkdir and rmdir functions, reading directories, and chdir, fchdir, and getcwd functions.
4. The Environment of a Unix Process, and Process Control (Advanced 7 & 8)
   - process ID, parent process ID, process group ID, real and effective user and group IDs
   - process resource limits, process termination, command-line arguments, and environment variables.
   - process control with fork, exit, wait, waitid, and exec functions.
5. Primitive Communications
   - lock files
   - locking files
   - signals and signal management calls.
6. Pipes
   - unnamed pipes
   - named pipes
7. Message Queues
   - creating a message queue

- message queue control
- message queue operations
- client-server message queue examples

8. Semaphores
   - creating and accessing semaphores sets
   - semaphore control
   - semaphore operations
9. Shared Memory
   - creating a shared memory segment
   - shared memory control
   - shared memory operations
   - using a file as shared memory
10. Remote Procedure Calls
   - Executing remote commmands at a system level
   - executing remote commands in a C program
   - transforming a local function call into a remote procedure
11. Sockets
   - network addresses, domains-network, and communication, protocol families, and socket types.
   - IPC using socketpair
   - IPC using connection-oriented sockets (Unix domain and internet domain)
   - connectionless client-server communication (Unix domain and internet domain)
12. Threads
   - creating and exiting threads
   - basic threads management
   - thread attributes
   - scheduling threads

VII. Teaching Methods:

1. Classroom lectures
2. Exercise/homework/lab assignments discussions
2. Open and close lab sessions

VIII. Evaluation:

1. Software projects
2. Tests and final exam

IX. Textbook(s):

UNIX System Programming, 2nd Ed., by K. Haviland, D. Gray, and B. Salama, 1998, Addison- Wesley.

UNIX for Programmers and Users, 2nd Ed., by Graham Glass and King Ables, 1999, Prentice Hall

X. Bibliography:

Inter process Communications in Unix, The Nooks & Crannies, 2nd edition, by John Shapley Gray, Prentice Hall, 1998.

Unix Systems Programming: Communication, Concurrency and Threads, 2nd Edition, by Kay Robbins and Steve Robbins, Prentice-Hall, 2004.

Solaris Systems Programming, by by Rich Teer, Prentice-Hall, 2004.

Advanced Programming in the Unix Environment, 2nd edition, by W. Richard Stevens and Stephen A. Rago, Addison-Wesley, 2005.

Advanced UNIX Programming, 2nd edition, by Marc J. Rochkind, Addison-Wesley, 2004.

The Art of UNIX Programming, by Eric S. Raymond, Addison-Wesley, 2004.

UNIX Internals: The New Frontiers, by Uresh Vahalia, Prentice-Hall, 1996.

Unix Internals: A Systems Operations Handbook, by Clement Shaw and Susan Soltis Shaw, Tab, 1987,

XI. Prepared by: Dr Gilbert Ndjatou

XII. Original Department Approval Date: Spring 1997.

XIII. Revised by: Dr. Gilbert Ndjatou, Spring 2005 and previously on April 1, 2000.

XIV. Department Revision Approval Date: Spring 2005.