CMP2204 Operating System Technologies

Period per Week			Contact Hour per Semester	Weighted Total Mark	Weighted Exam Mark	Weighted Continuous Assessment Mark	Credit Units
LH	PH	TH	CH	WTM	WEM	WCM	CU
45	30	00	60	100	40	100	4

Rationale

An operating system defines a software interface of the computer hardware and the architecture with which computer engineers can control and exploit the hardware to provide maximum benefit to the user. It also manages sharing of resources (hardware and software) among the computer's users (user programs and systems programs). Many of the ideas involved in operating system use have wider applicability across the

field of computer engineering such as concurrent programming.

Objectives

- Student should understand the basic principles and the purposes of an operating system prior to a study of digital instrumentation and embedded systems.
- To addresses both the use of operating systems (externals) and their design and implementation (internals).

Course Content

1. ***History and Overview***
   - Indicate some reasons for studying operating systems
   - Highlight some people that influenced or contributed to the area of operating systems ; Indicate some important topic areas such as function and design, concurrency, scheduling, dispatch, memory management, device management, file systems, security, and protection ; Describe the purpose of an operating system
   - Indicate the meaning of an interrupt
   - Describe the meaning of concurrency and the reasons for its importance
   - Illustrate the manner in which scheduling and dispatch take place in a computer through its operating system
   - Describe the manner and importance of memory management
   - Describe the manner and importance of device management
   - Explore some additional resources associated with operating systems
   - Explain the purpose and role of operating systems in computer engineering
2. ***Design Principles***
   - Functionality of a typical operating system
   - Mechanisms to support client-server models, hand-held devices
   - Design issues (efficiency, robustness, flexibility, portability, security, compatibility)
   - Influences of security, networking, multimedia, windows
   - Structuring methods (monolithic, layered, modular, micro-kernel models)
   - Abstractions, processes, and resources ;
   - Concepts of application program interfaces (APIs) specific to operating systems
   - Applications needs and the evolution of hardware/software techniques; Device organization; Interrupts: methods and implementations
   - Concept of user/system state and protection, transition to kernel mode
3. ***Concurrency***
   - States and state diagrams
   - Structures (ready list, process control blocks, and so forth)
   - Dispatching and context switching ; The role of interrupts
   - Concurrent execution: advantages and disadvantages
   - The "mutual exclusion" problem and some solutions
   - Deadlock: causes, conditions, prevention
   - Models and mechanisms (semaphores, monitors, condition variables, rendezvous)
   - Producer-consumer problems and synchronization
   - Multiprocessor issues (spin-locks, reentrancy)
4. ***Scheduling And Dispatch***
   - Preemptive and non-preemptive scheduling

- Schedulers and policies
- Processes and threads
- Deadlines and real-time issues

5. ***Memory Management***
   - Review of physical memory and memory management hardware
   - Overlays, swapping, and partitions
   - Paging and segmentation
   - Placement and replacement policies
   - Working sets and thrashing
   - Caching
6. ***Device Management***
   - Characteristics of serial and parallel devices
   - Abstracting device differences
   - Buffering strategies
   - Direct memory access
   - Recovery from failures
7. ***Security and Protection***
   - Overview of system security
   - Policy/mechanism separation
   - Security methods and devices
   - Protection, access, and authentication
   - Models of protection
   - Memory protection
   - Encryption Recovery management
8. ***File Systems***
   - Files: data, metadata, operations, organization, buffering, sequential, non-sequential
   - Directories: contents and structure
   - File systems: partitioning, mount/unmount, and virtual file systems
   - Standard implementation techniques
   - Memory-mapped files
   - Special-purpose file systems
   - Naming, searching, access, backups
9. ***System Performance Evaluation***
   - Why and what system performance needs to be evaluated
   - Policies for caching, paging, scheduling, memory management, security, and so forth
   - Evaluation models: deterministic, analytic, simulation, or implementation-specific
   - How to collect evaluation data (profiling and tracing mechanisms)

Learning Outcomes

- Identify some contributors to operating systems and relate their achievements to the knowledge area; provide some reasons for a computer to have an operating system; describe concurrency and reasons for its importance; describe scheduling and illustrate how it works to improve computer performance; sketch an example of how and why a compute would need to manage memory; identify some devices an operating system would manage; and describe how computer engineering uses or benefits from operating systems.

- Demonstrate understanding of Operating Systems as an interface between user programs and the computer hardware; demonstrate understanding of the logical layers and the benefits of building these layers in a hierarchical fashion; relate system state to user protection; explain the range of requirements that a modern operating system has to address; define the functionality that a modern operating system must deliver to meet a particular need; and articulate design tradeoffs inherent in operating system design.
- Justify the presence of concurrency within the framework of an operating system; demonstrate the potential run-time problems arising from the concurrent operation of many (possibly a dynamic number of) tasks; summarize the range of mechanisms (at an operating system level) that are useful to realize concurrent systems and be able to describe the benefits of each; explain the different states that a task may pass through and the data structures needed to support the management of many tasks.
- Compare and contrast the common algorithms used for both preemptive and non-preemptive scheduling of tasks in operating systems; describe relationships between scheduling algorithms and application domains; and investigate the wider applicability of scheduling in such contexts as disk I/O, networking scheduling, and project scheduling.
- Introduce memory hierarchy and cost-performance tradeoffs; explain virtual memory and its realization in hardware and software; examine the wider applicability and relevance of the concepts of virtual entity and of caching; evaluate the trade-offs in terms of memory size (main memory, cache memory, auxiliary memory) and processor speed; and defend the different ways of allocating memory to tasks based on the relative merits of each.
- Identify the relationship between the physical hardware and the virtual devices maintained by the operating system; differentiate the mechanisms used in interfacing a range of devices (including hand-held devices, networks, multimedia) to a computer and explain the implications of these for the design of an operating system; and implement a simple device driver for a range of possible devices.
- Defend the need for protection and security, and the role of ethical considerations in computer use; summarize the features of an operating system used to provide protection and security, and describe the limitations of each of these; and compare and contrast current methods for implementing security.
- Summarize the full range of considerations that support file systems; compare and contrast different approaches to file organization, recognizing the strengths and weaknesses of each.
- Describe the performance metrics used to determine how a system performs; explain the main evaluation models used to evaluate a system.

Recommended and Reference Books

[1] David Groth, Dan Newland, 2001. *A+ Operating System Technologies Study Guide.* Published by John Wiley & Sons Australia, Limited, 2001. ISBN 0782128076, 9780782128079

[2] Charles J. Brooks, 2004. *A+ Certification Operating System Technologies.* Prentice Hall, ISBN 0131147730, 9780131147737

[3] Andrew S. Tanenbaum, 2007. *Modern Operating Systems (3rd Edition) (GOAL Series).* Prentice Hall; 3 Edition ISBN-10: 0136006639, ISBN-13: 978-0136006633