CSE 2431 (Approved): Systems II: Introduction to Operating Systems

Course Description

Introduction to operating system concepts: process, CPU scheduling, memory management, file system and storage, and multi-threaded programming.

Prior Course Number: CSE 660
Transcript Abbreviation: Sys II: Oper Sys
Grading Plan: Letter Grade
Course Deliveries: Classroom
Course Levels: Undergrad
Student Ranks: Sophomore, Junior
Course Offerings: Autumn, Spring, Summer
Flex Scheduled Course: Never
Course Frequency: Every Year
Course Length: 14 Week
Credits: 3.0
Repeatable: No
Time Distribution: 3.0 hr Lec
Expected out-of-class hours per week: 6.0
Graded Component: Lecture
Credit by Examination: No
Admission Condition: No
Off Campus: Never
Campus Locations: Columbus
Prerequisites and Co-requisites: CSE 2421 or ((CSE 360 or ECE 2560 or ECE 265) and (CSE 2451 or CSE 459.21 or CSE 459.22))
Exclusions: Not open to students with credit for CSE 660
Cross-Listings:

The course is required for this unit's degrees, majors, and/or minors: Yes
The course is a GEC: No
The course is an elective (for this or other units) or is a service course for other units: Yes

Subject/CIP Code: 14.0901
Subsidy Level: Baccalaureate Course

Programs

Abbreviation	Description
BS CSE	BS Computer Science and Engineering

Course Goals

Be competent with process concepts and CPU scheduling.
Be competent with memory hierarchy and memory management.
Be familiar with process control blocks, system calls, context switching, interrupts, and exception control flows.
Be familiar with process synchronization, inter-process communication, and threads.
Be familiar with multi-threaded programming.
Be familiar with file systems and disk scheduling algorithms.

Course Topics

Topic	Lec	Rec	Lab	Cli	IS	Sem	FE	Wor
Introduction to operating systems, overview of related computer architecture concepts (CPU modes of operation, exceptions/interrupts, clock).	3.0							
Process concepts, process control block, memory and CPU protection, process hierarchy, shell, process (Unix-like) related system calls, interactions between systems calls, context switching and underlying interrupt, timer mechanisms.	6.0							
Process interactions, exception control flow (classes of exceptions, exception handling, private address space, user and kernel modes, process control, loading and running programs, Unix fork and exec system calls, signals).	3.0							
Process synchronization (e.g., critical section problem, synchronization problems), deadlock and inter-process communication, threads.	6.0							
Process (CPU) scheduling (various CPU scheduling algorithms).	3.0							
Multi-thread programming.	3.0							
Memory hierarchy	6.0							
Memory management (contiguous allocation, paging, segmentation, virtual memory).	6.0							
File systems (file system hierarchy, i-node, files, directories, file system management and optimization).	3.0							
Disk allocation and disk arm scheduling.	3.0							

Representative Assignments

Building a simple shell step by step, including system call invocation, signal handling, and file operations.
Implementing algorithms for a classic synchronization problem.
Designing and implementing multi-threaded programs.

Grades

Aspect	Percent
Programming assignments (4-5)	35%
Written assignments (3-4)	10%
Mid-term	20%
Final exam	35%

Representative Textbooks and Other Course Materials

Title	Author
Operating System Concepts	Silberschatz, Galvin, and Gagne
Computer Systems: A Programmer's Perspective	Bryant and O'Hallaron

ABET-EAC Criterion 3 Outcomes

Course Contribution		College Outcome
**	a	An ability to apply knowledge of mathematics, science, and engineering.
**	b	An ability to design and conduct experiments, as well as to analyze and interpret data.
**	c	An ability to design a system, component, or process to meet desired needs.
	d	An ability to function on multi-disciplinary teams.
***	e	An ability to identify, formulate, and solve engineering problems.
	f	An understanding of professional and ethical responsibility.
	g	An ability to communicate effectively.
	h	The broad education necessary to understand the impact of engineering solutions in a global and societal context.
*	i	A recognition of the need for, and an ability to engage in life-long learning.
	j	A knowledge of contemporary issues.
***	k	An ability to use the techniques, skills, and modern engineering tools necessary for engineering practice.

BS CSE Program Outcomes

Course Contribution		Program Outcome
**	a	an ability to apply knowledge of computing, mathematics including discrete mathematics as well as probability and statistics, science, and engineering;
**	b	an ability to design and conduct experiments, as well as to analyze and interpret data;
**	c	an ability to design, implement, and evaluate a software or a software/hardware system, component, or process to meet desired needs within realistic constraints such as memory, runtime efficiency, as well as appropriate constraints related to economic, environmental, social, political, ethical, health and safety, manufacturability, and sustainability considerations;
	d	an ability to function on multi-disciplinary teams;
***	e	an ability to identify, formulate, and solve engineering problems;
	f	an understanding of professional, ethical, legal, security and social issues and responsibilities;
	g	an ability to communicate effectively with a range of audiences;
	h	an ability to analyze the local and global impact of computing on individuals, organizations, and society;
*	i	a recognition of the need for, and an ability to engage in life-long learning and continuing professional development;
	j	a knowledge of contemporary issues;
***	k	an ability to use the techniques, skills, and modern engineering tools necessary for practice as a CSE professional;
**	l	an ability to analyze a problem, and identify and define the computing requirements appropriate to its solution;
***	m	an ability to apply mathematical foundations, algorithmic principles, and computer science theory in the modeling and design of computer-based systems in a way that demonstrates comprehension of the tradeoffs involved in design choices;
**	n	an ability to apply design and development principles in the construction of software systems of varying complexity.

Prepared by: Bruce Weide