CSE2005	OPERATING SYSTEMS	L	T	P	J	C
		2	0	2	4	4
Pre-requisite	**NIL**	**Syllabus version**				
		v1.0				

Course Objectives:

1. To introduce the concept of Operating system concepts and designs and provide the skills required to implement the services.
2. To describe the trade-offs between conflicting objectives in large scale system design.
3. To develop the knowledge for application of the various design issues and services.

Expected Course Outcome:

1. Interpret the evolution of OS functionality, structures and layers.
2. Apply various types of system calls and to find the stages of various process states.
3. Design a model scheduling algorithm to compute various scheduling criteria.
4. Apply and analyze communication between inter process and synchronization techniques.
5. Implement page replacement algorithms, memory management problems and segmentation.
6. Differentiate the file systems for applying different allocation and access techniques.
7. Representing virtualization and Demonstrating the various Operating system tasks and the principle algorithms for enumerating those tasks.

Student Learning Outcomes (SLO):	**2, 14, 17**

Module:1	**Introduction**	**2 hours**

Introduction to OS: - Functionality of OS - OS Design issues - Structuring methods (monolithic, layered, modular, micro-kernel models) - Abstractions, processes, and resources - influence of security, networking, multimedia.

Module:2	**OS Principles**	**3 hours**

System Calls System/Application Call Interface - Protection User/Kernel modes - Interrupts Processes and Threads - Structures (Process Control Block, Ready List etc).

Module:3	**Scheduling**	**5 hours**

Processes Scheduling - CPU Scheduling - Pre-emptive non-pre-emptive - Resource allocation and management - Deadlocks Deadlock Handling Mechanisms.

Module:4	**Concurrency**	**4 hours**

Inter-process communication Synchronization - Implementing Synchronization Primitives Semaphores - Monitors - Multiprocessors and Locking - Scalable Locks - Lock-free Coordination.

Module:5	**Memory management**	**5 hours**

Main Memory management Memory allocation strategies Caching -Virtual Memory Hardware TLB - Virtual Memory OS techniques Paging Segmentation Page Faults Page Replacement Thrashing Working Set.

Module:6	**Virtualization**	**4 hours**

Virtual Machines Virtualization (Hardware/Software, Server, Service, Network) Hypervisors -OS - Container Virtualization - Cost of virtualization.

Module:7	**File systems**	**3 hours**

File system interface - file system implementation File system recovery Journaling - Soft updates LFS - Distributed file system.			
Module:8	**Security Protection and trends**		**4 hours**
Security and Protection - Mechanism Vs Policies Access and authentication - models of protection Memory Protection Disk Scheduling - OS performance, Scaling OS - Mobile OS: Recent Trends: - Future directions in Mobile OS / Multi-core Optimization /Power efficient Scheduling			
	Total Lecture hours:	**30 hours**	
Text Book(s)			
1.	Abraham Silberschatz, Peter B. Galvin, Greg Gagne-Operating System Concepts, Wiley (2012).		
Reference Books			
1.	Ramez Elmasri, A Carrick, David Levine, Operating Systems, A Spiral Approach - McGrawHill Science Engineering Math (2009).		
2.	Remzi H. Arpaci-Dusseau, Andrea C. Arpaci-Dusseau, Operating Systems, Three Easy Pieces,Arpaci-Dusseau Books, Inc (2015).		
Mode of Evaluation: CAT / Assignment / Quiz / FAT / Project / Seminar			
List of Challenging Experiments (Indicative)			
1.	Write a boot loader - to load a particular OS say TinyOS/ KolibriOS image - code to access from BIOS to loading the OS - involves little assembly code may use QEMU/virtual machines for emulation of hardware.		3 hours
2.	Allocate/free memory to processes in whole pages, find max allocatable pages, incorporate address translation into the program.		3 hours
3.	Create an interrupt to handle a system call and continue the previously running process after servicing the interrupt.		3 hours
4.	Write a Disk driver for the SATA interface. Take care to check readiness of the controller, locked buffer cache, accept interrupts from OS during the period, interrupting the OS again once done and clearing buffers.		3 hours
5.	Demonstrate the use of locks in conjunction with the IDE driver.		3 hours
6.	Run an experiment to determine the context switch time from one process to another and one kernel thread to another. Compare the findings.		3 hours
7.	Determine the latency of individual integer access times in main memory, L1 Cache and L2 Cache. Plot the results in log of memory accessed vs average latency.		3 hours
8.	Compare the overhead of a system call with a procedure call. What is the cost of a minimal system call?		3 hours
9.	Compare the task creation times. Execute a process and kernel thread, determine the time taken to create and run the threads.		3 hours
10.	Determine the file read time for sequential and random access based of varying sizes of the files. Take care not to read from cached data - used the raw device interface. Draw a graph log/log plot of size of file vs average per-block time.		3 hours
		Total Laboratory Hours	30 hours
Mode of assessment: Project/Activity			
Recommended by Board of Studies	04-04-2014		
Approved by Academic Council	No. 37	Date	16-06-2015