San Pablo Catholic University (UCSP)
Undergraduate Program in
Computer Science
SILABO

CS2S1. Operating systems (Mandatory)

2023-II

1. General information		
1.1 School	:	Ciencia de la Computación
1.2 Course	:	CS2S1. Operating systems
1.3 Semester	:	6^{to} Semestre.
1.4 Prerrequisites	:	CS221. Computer Architecture. (3^{rd} Sem)
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 4 HP;
1.8 Credits	:	4
1.9 Plan	:	Plan Curricular 2016

2. Professors

Lecturer

- Julio Omar Santisteban Pablo <jsantisteban@ucsp.edu.pe>
  - PhD in Ciencias de la Computación, Universidad Nacional de San Agustin, Perú, 2021.
  - MSc in Internetworking, University of Technology, Australia, 2008.

3. Course foundation

An Operating System (OS) manages the computing resources to complete the execution of multiple applications and their associated processes. This course teaches the design of modern operating systems; and introduces their fundamental concepts covering multiple-program execution, scheduling, memory management, file systems, and security. Also, the course includes programming activities on a minimal operating system to solve problems and extend its functionality. Notice that these activities require much time to complete. However, working on them provides valuable insight into operating systems.

4. Summary

1. Overview of Operating Systems 2. Operating System Principles 3. Concurrency 4. Scheduling and Dispatch 5. Memory Management 6. Security and Protection 7. Virtual Machines 8. Device Management 9. File Systems 10. Real Time and Embedded Systems 11. Fault Tolerance 12. System Performance Evaluation

5. Generales Goals

- Study the design of modern operating systems.
- Provide a practical experience by designing and implementing a minimal operating system.

6. Contribution to Outcomes

This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Assessment**)

4) Recognize professional responsabilities and make informed judgments in computing practice based on legal and ethical principles. (**Familiarity**)

5) Function effectively as a member or leader of a team engaged in activities appropriate to the program's discipline. (**Usage**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Usage**)

7) Develop computational technology for the well-being of all, contributing with human formation, scientific, technological and professional skills to solve social problems of our community. (**Assessment**)

7. Content

UNIT 1: Overview of Operating Systems (3)

Competences:

Content	**Generales Goals**
• Role and purpose of the operating system • Functionality of a typical operating system • Mechanisms to support client-server models. • Design issues (efficiency, robustness, flexibility, portability, security, compatibility) • Influences of security, networking, multimedia, windowing systems	• Explain the objectives and functions of modern operating systems [Familiarity] • Analyze the tradeoffs inherent in operating system design [Assessment] • Describe the functions of a contemporary operating system with respect to convenience, efficiency, and the ability to evolve [Familiarity] • Discuss networked, client-server, distributed operating systems and how they differ from single user operating systems [Familiarity] • Identify potential threats to operating systems and the security features design to guard against them [Familiarity]

Readings: Avi Silberschatz (2012), Stallings (2005), Tanenbaum (2006), Tanenbaum (2001), Anderson and Dahlin (2014)

UNIT 2: Operating System Principles (6)

Competences:

Content	Generales Goals
• Operating Sistems Structure (monolithic, layered, modular, micro-kernel models) • Abstractions, processes, and resources • Concepts of application program interfaces (APIs) • The evolution of hardware/software techniques and application needs • Device organization • Interrupts: methods and implementations • Concept of user/system state and protection, transition to kernel mode	• Explain the concept of a logical layer [Familiarity] • Explain the benefits of building abstract layers in hierarchical fashion [Familiarity] • Describe the value of APIs and middleware [Familiarity] • Describe how computing resources are used by application software and managed by system software [Familiarity] • Contrast kernel and user mode in an operating system [Assessment] • Discuss the advantages and disadvantages of using interrupt processing [Familiarity] • Explain the use of a device list and driver I/O queue [Familiarity]

Readings: Avi Silberschatz (2012), Stallings (2005), Tanenbaum (2006), Tanenbaum (2001), Anderson and Dahlin (2014)

UNIT 3: Concurrency (9)

Competences:

Content	Generales Goals
• States diagrams • Structures (ready list, process control blocks, and so forth) • Dispatching and context switching • The role of interrupts • Managing atomic access to OS objects • Implementing synchronization primitives • Multiprocessor issues (spin-locks, reentrancy)	• Describe the need for concurrency within the framework of an operating system [Familiarity] • Demonstrate the potential run-time problems arising from the concurrent operation of many separate tasks [Usage] • Summarize the range of mechanisms that can be employed at the operating system level to realize concurrent systems and describe the benefits of each [Familiarity] • Explain the different states that a task may pass through and the data structures needed to support the management of many tasks [Familiarity] • Summarize techniques for achieving synchronization in an operating system (eg, describe how to implement a semaphore using OS primitives) [Familiarity] • Describe reasons for using interrupts, dispatching, and context switching to support concurrency in an operating system [Familiarity] • Create state and transition diagrams for simple problem domains [Usage]

Readings: Avi Silberschatz (2012), Stallings (2005), Tanenbaum (2006), Tanenbaum (2001), Anderson and Dahlin (2014)

UNIT 4: Scheduling and Dispatch (6)	
Competences:	
Content	**Generales Goals**
• Preemptive and non-preemptive scheduling • Schedulers and policies • Processes and threads • Deadlines and real-time issues	• Compare and contrast the common algorithms used for both preemptive and non-preemptive scheduling of tasks in operating systems, such as priority, performance comparison, and fair-share schemes [Assessment] • Describe relationships between scheduling algorithms and application domains [Familiarity] • Discuss the types of processor scheduling such as short-term, medium-term, long-term, and I/O [Familiarity] • Describe the difference between processes and threads [Familiarity] • Compare and contrast static and dynamic approaches to real-time scheduling [Assessment] • Discuss the need for preemption and deadline scheduling [Familiarity] • Identify ways that the logic embodied in scheduling algorithms are applicable to other domains, such as disk I/O, network scheduling, project scheduling, and problems beyond computing [Familiarity]
Readings: Avi Silberschatz (2012), Stallings (2005), Tanenbaum (2006), Tanenbaum (2001), Anderson and Dahlin (2014)	

UNIT 5: Memory Management (6)	
Competences:	
Content	**Generales Goals**
• Review of physical memory and memory management hardware • Working sets and thrashing • Caching	• Explain memory hierarchy and cost-performance trade-offs [Familiarity] • Summarize the principles of virtual memory as applied to caching and paging [Familiarity] • Evaluate the trade-offs in terms of memory size (main memory, cache memory, auxiliary memory) and processor speed [Assessment] • Defend the different ways of allocating memory to tasks, citing the relative merits of each [Familiarity] • Describe the reason for and use of cache memory (performance and proximity, different dimension of how caches complicate isolation and VM abstraction) [Familiarity] • Discuss the concept of thrashing, both in terms of the reasons it occurs and the techniques used to recognize and manage the problem [Familiarity]
Readings: Avi Silberschatz (2012), Stallings (2005), Tanenbaum (2006), Tanenbaum (2001), Anderson and Dahlin (2014)	

UNIT 6: Security and Protection (6)	
Competences:	
Content	**Generales Goals**
• Overview of system security • Policy/mechanism separation • Security methods and devices • Protection, access control, and authentication • Backups	• Articulate the need for protection and security in an OS [Familiarity] • Summarize the features and limitations of an operating system used to provide protection and security [Familiarity] • Explain the mechanisms available in an OS to control access to resources (cross reference IAS/Security Architecture and Systems Administration/Access Control/Configuring systems to operate securely as an IT system) [Familiarity] • Carry out simple system administration tasks according to a security policy, for example creating accounts, setting permissions, applying patches, and arranging for regular backups (cross reference IAS/Security Architecture and Systems Administration) [Familiarity]
Readings: Avi Silberschatz (2012), Stallings (2005), Tanenbaum (2006), Tanenbaum (2001), Anderson and Dahlin (2014)	

UNIT 7: Virtual Machines (6)	
Competences:	
Content	**Generales Goals**
• Types of virtualization (including Hardware/Software, OS, Server, Service, Network) • Paging and virtual memory • Virtual file systems • Hypervisors • Portable virtualization; emulation vs. isolation • Cost of virtualization	• Explain the concept of virtual memory and how it is realized in hardware and software [Familiarity] • Differentiate emulation and isolation [Familiarity] • Evaluate virtualization trade-offs [Assessment] • Discuss hypervisors and the need for them in conjunction with different types of hypervisors [Familiarity]
Readings: Avi Silberschatz (2012), Stallings (2005), Tanenbaum (2006), Tanenbaum (2001), Anderson and Dahlin (2014)	

UNIT 8: Device Management (6)	
Competences:	
Content	**Generales Goals**
• Characteristics of serial and parallel devices • Abstracting device differences • Buffering strategies • Direct memory access • Recovery from failures	• Explain the key difference between serial and parallel devices and identify the conditions in which each is appropriate [Familiarity] • Identify the relationship between the physical hardware and the virtual devices maintained by the operating system [Familiarity] • Explain buffering and describe strategies for implementing it [Familiarity] • 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 [Familiarity] • Describe the advantages and disadvantages of direct memory access and discuss the circumstances in which its use is warranted [Familiarity] • Identify the requirements for failure recovery [Familiarity] • Implement a simple device driver for a range of possible devices [Usage]
Readings: Avi Silberschatz (2012), Stallings (2005), Tanenbaum (2006), Tanenbaum (2001), Anderson and Dahlin (2014)	

UNIT 9: File Systems (6)	
Competences:	
Content	**Generales Goals**
• Files: data, metadata, operations, organization, buffering, sequential, nonsequential. • Directories: contents and structure. • File systems: partitioning, mount/unmount, virtual file systems. • Standard implementation techniques • Memory-mapped files • Special-purpose file systems. • Naming, searching, access, backups • Journaling and log-structured file systems	• Describe the choices to be made in designing file systems [Familiarity] • Compare and contrast different approaches to file organization, recognizing the strengths and weaknesses of each [Assessment] • Summarize how hardware developments have led to changes in the priorities for the design and the management of file systems [Familiarity] • Summarize the use of journaling and how log-structured file systems enhance fault tolerance [Familiarity]
Readings: Avi Silberschatz (2012), Stallings (2005), Tanenbaum (2006), Tanenbaum (2001), Anderson and Dahlin (2014)	

UNIT 10: Real Time and Embedded Systems (6)	
Competences:	
Content	**Generales Goals**
• Process and task scheduling • Memory/disk management requirements in a real-time environment • Failures, risks, and recovery. • Special concerns in real-time systems	• Describe what makes a system a real-time system [Familiarity] • Explain the presence of and describe the characteristics of latency in real-time systems [Familiarity] • Summarize special concerns that real-time systems present, including risk, and how these concerns are addressed [Familiarity]
Readings: Avi Silberschatz (2012), Stallings (2005), Tanenbaum (2006), Tanenbaum (2001), Anderson and Dahlin (2014)	

UNIT 11: Fault Tolerance (3)	
Competences:	
Content	**Generales Goals**
• Fundamental concepts: reliable and available systems • Spatial and temporal redundancy • Methods used to implement fault tolerance • Examples of OS mechanisms for detection, recovery, restart to implement fault tolerance, use of these techniques for the OS's own services.	• Explain the relevance of the terms fault tolerance, reliability, and availability [Familiarity] • Outline the range of methods for implementing fault tolerance in an operating system [Familiarity] • Explain how an operating system can continue functioning after a fault occurs [Familiarity]
Readings: Avi Silberschatz (2012), Stallings (2005), Tanenbaum (2006), Tanenbaum (2001), Anderson and Dahlin (2014)	

UNIT 12: System Performance Evaluation (3)	
Competences:	
Content	**Generales Goals**
• Why system performance needs to be evaluated? • What is to be evaluated? • Systems performance policies, e.g., caching, paging, scheduling, memory management, and security • Evaluation models: deterministic, analytic, simulation, or implementation-specific • How to collect evaluation data (profiling and tracing mechanisms)	• Describe the performance measurements used to determine how a system performs [Familiarity] • Explain the main evaluation models used to evaluate a system [Familiarity]
Readings: Avi Silberschatz (2012), Stallings (2005), Tanenbaum (2006), Tanenbaum (2001), Anderson and Dahlin (2014)	

8. Methodology

1. El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.
2. El profesor del curso presentará demostraciones para fundamentar clases teóricas.
3. El profesor y los alumnos realizarán prácticas
4. Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.

9. Assessment

Continuous Assessment 1 : 20 %

Partial Exam : 30 %

Continuous Assessment 2 : 20 %

Final exam : 30 %

References

Anderson, Thomas and Michael Dahlin (2014). *Operating Systems: Principles and Practice*. 2nd. Recursive Books. ISBN: 978-0985673529.

Avi Silberschatz Peter Baer Galvin, Greg Gagne (2012). *Operating System Concepts, 9/E*. John Wiley & Sons, Inc. ISBN: 978-1-118-06333-0.

Stallings, William (2005). *Operating Systems: Internals and Design Principles, 5/E*. Prentice Hall. ISBN: 0-13-147954-7.

Tanenbaum, Andrew S. (2001). *Modern Operating Systems, 4/E*. Prentice Hall. ISBN: 0-13-031358-0.

Tanenbaum, Andrew S. (2006). *Operating Systems Design and Implementation, 3/E*. Prentice Hall. ISBN: 0-13-142938-8.