

**Proyecto/Guía docente de la asignatura Estructura de Sistemas Operativos**

Asignatura	ESTRUCTURA DE SISTEMAS OPERATIVOS		
Materia	ARQUITECTURA DE COMPUTADORES, SISTEMAS OPERATIVOS Y SISTEMAS DISTRIBUIDOS		
Módulo	COMUNES A LA INFORMÁTICA		
Titulación	GRADUADO EN INGENIERÍA INFORMÁTICA		
Plan	545	Código	46915
Periodo de impartición	Segundo Cuatrimestre	Tipo/Carácter	Complementos de Informática
Nivel/Ciclo	Grado	Curso	2
Créditos ECTS	6		
Lengua en que se imparte	Español		
Profesor/es responsable/s	Benjamín Sahelices Fernández		
Datos de contacto (E-mail, teléfono...)	benja@infor.uva.es Tfno. 983185643, 983423713		
Horario de tutorías	Lunes (11:00-13:00), jueves (11:00-13:00) y viernes (11:00-13:00)		
Departamento	Informática		

1. Situación / Sentido de la Asignatura

1.1 Contextualización

La asignatura *Estructura de Sistemas Operativos* se encuentra dentro del conjunto de materias comunes a la informática. En concreto está incluida en la materia *Arquitectura de Computadores, Sistemas Operativos y Sistemas Distribuidos* siendo su carácter obligatorio. Se imparte en el segundo curso, segundo cuatrimestre, de la titulación de *Graduado en Ingeniería Informática*. Se trata de una continuación de la asignatura *Fundamentos de Sistemas Operativos*, que se imparte en el primer cuatrimestre del segundo curso.

1.2 Relación con otras materias

La asignatura *Estructura de Sistemas Operativos* está muy relacionada con la asignatura *Fundamentos de Sistemas Operativos*. Se trata de una continuación en la que se explican desde un punto de vista de estructura y diseño los conceptos básicos de sistemas operativos.

Adicionalmente esta asignatura está fundamentada sobre un conjunto de conocimientos conseguidos en las asignaturas *Fundamentos de Programación*, *Fundamentos de Computadoras* y *Arquitectura y Organización de Computadoras*.

1.3 Prerrequisitos

Recomendación: haber cursado previamente las asignaturas *Fundamentos de Programación* (primer curso, primer cuatrimestre), *Fundamentos de Computadoras* (primer curso, segundo cuatrimestre) y *Fundamentos de Sistemas Operativos* (segundo curso, primer cuatrimestre).

2. Competencias

2.1 Generales

G2	Conocimientos básicos de la profesión
G3	Capacidad de análisis y síntesis
G4	Capacidad de organizar y planificar
G5	Comunicación oral y escrita en la lengua propia
G8	Habilidades de gestión de la información
G9	Resolución de problemas
G10	Toma de decisiones
G11	Capacidad crítica y autocrítica
G12	Trabajo en equipo
G14	Responsabilidad y compromiso ético
G15	Liderazgo
G16	Capacidad de aplicar los conocimientos en la práctica
G18	Capacidad de aprender
G19	Capacidad de adaptarse a nuevas situaciones
G20	Capacidad de generar nuevas ideas
G21	Habilidad para trabajar de forma autónoma
G22	Diseño y gestión de proyectos

2.2 Específicas

CI1	Capacidad para diseñar, desarrollar, seleccionar y evaluar aplicaciones y sistemas informáticos, asegurando su fiabilidad, seguridad y calidad, conforme a principios éticos y a la legislación y normativa vigente.
CI4	Capacidad para elaborar el pliego de condiciones técnicas de una instalación informática que cumpla los estándares y normativas vigentes
CI5	Conocimiento, administración y mantenimiento de sistemas, servicios y aplicaciones informáticas
CI10	Conocimiento de las características, funcionalidades y estructura de los Sistemas Operativos y diseñar e implementar aplicaciones basadas en sus servicios
CI12	Conocimiento y aplicación de las características, funcionalidades y estructura de las bases de datos, que permitan su adecuado uso, y el diseño y el análisis e implementación de aplicaciones basadas en ellos.

3. Objetivos

- Conocer los servicios que ofrece un SO, específicos desde el punto de vista de un usuario, de un programador y de un administrador y utilizar diferentes sistemas de interacción con el SO.
- Diseñar e implementar aplicaciones basadas en los servicios del SO, seleccionando los más adecuados a cada caso.
- Comprender y saber gestionar los sistemas de control de acceso a usuarios y recursos para garantizar un nivel de seguridad adecuado a nivel sistema operativo.
- Conocer los fundamentos de las técnicas de virtualización y, en concreto, de la virtualización de sistemas.

4. Bloques temáticos¹

Bloque 1: Gestión de Memoria

Carga de trabajo en créditos ECTS:

3,6

a. Contextualización y justificación

La gestión de memoria es un aspecto básico en las prestaciones de las computadoras. Se trata de un recurso crítico en el que las decisiones de administración tienen un elevado impacto en las prestaciones. En este bloque temático se estudian los mecanismos de gestión de memoria de los sistemas operativos. Estos conocimientos son imprescindibles para poder identificar y resolver problemas de prestaciones en la administración de sistemas operativos, para diseñar aplicaciones con buenas prestaciones y para utilizar correctamente los servicios de los sistemas operativos.

b. Objetivos de aprendizaje

- Conocer los servicios que ofrece un SO específicas desde el punto de vista de un usuario, de un programador y de un administrador y utilizar diferentes sistemas de interacción con el SO.
- Diseñar e implementar aplicaciones basadas en los servicios del SO, seleccionando los más adecuados a cada caso.
- Conocer la influencia de los sistemas de gestión de memoria en el diseño de aplicaciones y en la utilización de los servicios que proporciona un SO.

c. Contenidos

- Capítulo 1: Espacios de Direcciones.

Multiprogramming and time sharing. Ready queue. Protection. Address Space. Multiple threads. Abstraction. Hardware+software implementation Transparency, efficiency, protection. Homework: free, pmap, memory-user.c

- Capítulo 2: API Memoria.

Types of memory in AS. Code, stack, heap. malloc(). sizeof. free() Common errors: good compilation vs correction (not the same!). Allocation errors, initialization errors, free-type errors. Tools: valgrind (sudo apt install valgrind) Underlying OS support: brk, sbrk, mmap. Homework: debugging with gdb and valgrind (null.c, vectorInt.c)

- Capítulo 3: Traducción de direcciones.

Memory virtualization. Illusion. Hardware-based. Virtual Address (VA) to Physical address (PA). AS of a simple example (Suma3). Dynamic relocation. Hardware based. Hardware support. OS issues: mem. management, base/bound management, exception handling, OS boot. Homework: relocation.py

- Capítulo 4: Segmentación.

Segmentation. Spare ASs. Base/bound generalization. Segment identification. Segment Table (ST) storage. Stack. Sharing. OS support: context switch, grow/shrink, free space. Homework: segmentation.py

- Capítulo 5: Gestión del Espacio Libre.

External/internal fragmentation. malloc-free. Split and coalescing. Free list. Tracking size (free(pointer)). Embedding free list in free space. Growing the heap. Assignment strategies: best, worst, first, next (fit) Slab allocator. Buddy allocator. Homework: malloc.py

- Capítulo 6: Introducción a la Paginación

Fixed-size pieces: pages/frames. Page table (PT). Virtual Page Number (VPN) to Physical Frame Number (PFN) conversión PT in physical or virtual memory. Page Table Base Register (PTBR). Page Table Entry (PTE): PFN, v, prot, present, dirty, ref Speed problem. Memory trace: array.c Homework: paging-linear-translate.py

- Capítulo 7: TLBs

Hardware to speed-up translation. VA to PA. Hardware managed Translation Lookaside Buffer (TLB). Example. Hardware managed TLB (CISC processors). Software managed TLB (RISC processors). TLB_MISS. TLB contents. Fully associative. VPN/ASIDS, PFN, valid, protection, dirty bit. Context switches. Replacement policies. LRU. Random. Real TLB entry (R4000) Homework: tlb.c

- Capítulo 8: Tablas de Páginas Avanzadas

PT requires a lot of storage. Bigger pages. Problem Paging+segments. VPN to PFN conversion. External fragmentation for PTs. Multi-level PT. Page Directory (PD) + PT (1 or more levels; each portion in one frame). Spare ASs. Detailed examples. Three levels. Inverted PT. Swapping. Virtual spaces. Homework: paging-multilevel-translate.py

- Capítulo 9: Intercambio – Mecanismos

AS greater than physical memory. Backup devices. Abstraction. Very large ASs for each process. Swap space. Disk address on PTE. Present bit. Trap to OS: page fault. OS handler. Page fault. I/O. Block process. Memory full. Victim selection. Replacement. Run at disk-speed instead of memory-speed. Page fault control flow. Page fault software description. Replacements. High Watermark (HW) / Low Watermark (LW). Page daemon. Page clusters to optimize disk operations. Homework: vmstat, mem.

- Capítulo 10: Intercambio – Políticas

Cache management. Minimize cache misses. Minimize page faults. Average Memory Access Time (AMAT). Miss probability (Pmiss). Between memory (T_m - nsecs) and disk (T_d - msecs). Optimal replacement policy. Comparing. Difficult to know the future. FIFO. Belady's anomaly. Random. LRU. MRU. MFU. Workload: fully random, 80-20, looping sequential. Algorithm implementation. Approximating LRU. Dirty pages. Page selection: demand paging, prefetching. Thrashing. Homework: paging-policy.py

- Capítulo 11: Sistema Completo de Memoria Virtual

VAX/VMS. Memory management hardware. Segment-paging. PT of user segments (P0, P1) in kernel virtual memory (can be swapped). PT of kernel segment (S) in physical memory. VAX AS description. User - OS. Sharing between user space and OS. Context switch. Protection. Page replacement. PTE (Page Table Entry). Resident Set Size (RS). FIFO list of pages, dirty-page list, clean-page list. Clustering. Zeroing pages - lazy policy. COW.

Linux (x86). User AS. Code, heap, stack. Kernel AS. KLAS. KVAS. Page table structure. Large page support. TLB reach. Internal fragmentation. Page cache. From ASs (anonymous) and from memory-mapped file. Page cache hash table. Dirty pages, clean pages. pdflush. Replacement algorithms. Version of 2Q. Inactive list. Active list. Memory-mapped files. mmap. Security and buffer overflow. PTE entry to forbid execution. Return-oriented programming (ROP). Address Space Layout Randomization (ASLR). Meltdown, spectre.

Parte práctica

- Programación de sistemas: C / UNIX
- El núcleo de MINIX
- Llamadas al sistema en MINIX

d. Métodos docentes

- Sesiones de aula: clases magistrales participativas y estudios de caso.
- Prácticas supervisadas en laboratorio con seguimiento en cada sesión de cada estudiante para poder realizar una evaluación continua del apartado práctico.

e. Plan de trabajo

Semana 1:

Aula: capítulos 1, 2 y 3

Laboratorio: entorno de desarrollo C en UNIX. Compilación, control de versiones, depuración, documentación. Práctica 1

Semana 2:

Aula: capítulos 4 y 5

Laboratorio: programación de utilidades de UNIX. Práctica 1.

Semana 3:

Aula: capítulo 6

Laboratorio: programación de utilidades de UNIX. Prácticas 1 y 2.

Semana 4:

Aula: capítulos 6 y 7

Laboratorio: programación de utilidades de UNIX. Prácticas 1 y 2.

Semana 5:

Aula: capítulos 7 y 8

Laboratorio: programación de utilidades de UNIX. Prácticas 1 y 2.

Semana 6:

Aula: capítulos 9 y 10

Laboratorio: programación de utilidades de UNIX. Prácticas 1 y 2.

Semana 7:

Aula: capítulo 11

Laboratorio: instalación y programación Minix. Práctica 3.

Semana 8:

Aula: repaso capítulos 1 a 11

Laboratorio: núcleo Minix I. Práctica 4.

Semana 9:

Aula: examen escrito semi-objetivo y de solución de problemas para el bloque 1

Laboratorio: núcleo Minix II. Práctica 5.

f. Evaluación

- Examen escrito semi-objetivo y de solución de problemas para el bloque 1.
- Evaluación continua para la parte práctica

g. Bibliografía básica

- **Remzi H. Arpaci-Dusseau and Andrea C. Arpaci-Dusseau, “Operating Systems: Three Easy Pieces”, Arpaci-Dusseau Books. Versión on-line: <http://ostep.org>**
- A.Silberschatz, P.Galvin, G.Gagne, "Operating System Concepts. 8th Edition", Wiley, 2010. ISBN 978-0-470-23399-3.



- A.S.Tanenbaum, A.S.Woodhull, "Sistemas Operativos. Diseño e Implementación", Segunda Edición, Prentice-Hall, 1997

h. Bibliografía complementaria

- A.S.Tanenbaum, "Sistemas Operativos Modernos. Tercera Edición.", Pearson/Prentice-Hall, 2009
- A.S.Tanenbaum, A.S.Woodhull, "Sistemas Operativos. Diseño e Implementación. El Libro de MINIX", Tercera Edición, Prentice-Hall, 2006

i. Recursos necesarios

Aulas Virtuales Escuela de Informática (www.inf.uva.es)

<http://www.minix3.org>

Bloque 2: Gestión del Almacenamiento

Carga de trabajo en créditos ECTS:

a. Contextualización y justificación

En los dispositivos de almacenamiento secundario se almacena la configuración de los sistemas operativos, el software de sistema y todos los datos de los usuarios. Por diseño estos dispositivos tienen mayor latencia y menor ancho de banda que el conjunto formado por el procesador y la memoria. En este bloque temático se estudia todo lo necesario para que el estudiante comprenda los problemas asociados al almacenamiento secundario.

b. Objetivos de aprendizaje

- Conocer los servicios que ofrece un SO específicas desde el punto de vista de un usuario, de un programador y de un administrador y utilizar diferentes sistemas de interacción con el SO.
- Diseñar e implementar aplicaciones basadas en los servicios del SO, seleccionando los más adecuados a cada caso.
- Conocer la influencia de los sistemas de gestión del almacenamiento y de la entrada/salida en el diseño de aplicaciones y en la utilización de los servicios que proporciona un SO.

c. Contenidos

- Capítulo 12: Dispositivos I/O

System architecture. Bus hierarchy. Canonical device. Canonical protocol. Programmed I/O. Polling. Interrupts. Hardware interrupts. ISR. Direct Memory Access. OS-device interaction. in/out. Memory mapped I/O. Device driver. Example. Simple disk driver

- Capítulo 13: HDDs

Interface. Blocks. Logical numbers. Clusters. Basic geometry. Platter, surface, track, head, arm. Simple disk drive example. I/O time. Performance vs capacity. Workloads (sequential vs random). Disk scheduling. SSTF, SCAN, SPTF

- Capítulo 14: RAID

Interface. Linear array of blocks. Logical requests. Fault model. RAID evaluation. Capacity. Reliability. Performance. RAID 0: striping. Chunk size. Capacity, reliability, performance. RAID 1: mirroring. Capacity, reliability, performance. RAID 4: parity. Reconstruction. Capacity, reliability, performance. RAID5: rotating parity. Capacity, reliability, performance. Other considerations: other RAID levels, hot spare disks, data integrity, software RAID. Homework: raid.py

- Capítulo 15: Ficheros y Directorios

File. Directory. FS interface. Creating files. Reading and writing files. Non-sequential R/W. Open File Table. Shared entries. Fsync. Renaming files. Getting information about files. stat. Removing files. strace. unlink. Making, reading and deleting directories. Hard links. Symbolic links. ACLs. Permission bits. Making and mounting FS

- Capítulo 16: Implementación del Sistema de Ficheros

Model of a FS. Overall organization. Blocks, inodes, superblock, inode bitmap, block bitmap. Directory organization. Linear list, b-tree, hash. Free space management. Reading and writing files. Caching and buffering. Homework: vsfs.py

- Capítulo 17: Sistemas de Ficheros Rápidos

Poor performance. Seeks and latency. Fragmentation. Cylinder group. Allocation of files and directories. File locality in name space. Large files. Optimizations. Sub-blocks. libc optimizations. Homework: ffs.py

- Capítulo 18: Crash Consistency

Survive to crashes. Example. Crash scenarios. Atomicity. Solution #1: FS checker. Lazy approach: fsck. fsck phases. Solution #2: journaling. Logging. Transactions. Atomicity of transactions. Circular logs. Metadata journaling. Other approaches. Homework: fsck.py

- Capítulo 19: SSDs

SLC. MLC. Basic flash operations: read-erase-program. States: invalid, erased, valid. Performance. Reliability. Wear-out. SSD=flash chips + FTL + SRAM. log-structured FTL. Mapping table. Logical blocks, physical blocks, physical pages. Garbage collection. Mapping table. Page-based. Block-based. Hybrid. SSD performance and cost. Homework: ssd.py

Parte práctica

- Planificación de procesos en MINIX
- El gestor de memoria de MINIX
- El sistema de ficheros de MINIX
- Controladores de entrada/salida en MINIX: terminal, reloj y disco

d. Métodos docentes

- Sesiones de aula: clases magistrales participativas y estudios de caso.
- Prácticas supervisadas en laboratorio con seguimiento en cada sesión de cada estudiante para poder realizar una evaluación continua del apartado práctico.

e. Plan de trabajo

Semana 10:

Aula: capítulos 12 y 13

Laboratorio: planificación de procesos en Minix. Práctica 6

Semana 11:

Aula: capítulos 14 y 15

Laboratorio: gestión de memoria en Minix. Práctica 7

Semana 12:

Aula: capítulos 16 y 17

Laboratorio: realización prácticas 3 a 7.

Semana 13:

Aula: capítulos 18 y 19

Laboratorio: realización prácticas 3 a 7. Práctica abierta opcional.

Semana 15:

Aula: revisión capítulos 12 a 19

Laboratorio: realización prácticas 3 a 7. Práctica abierta opcional.

f. Evaluación

- Examen escrito semi-objetivo y de solución de problemas para los bloques 2 y 3.
- Evaluación continua para la parte práctica.

g. Bibliografía básica

- **Remzi H. Arpaci-Dusseau and Andrea C. Arpaci-Dusseau, “Operating Systems: Three Easy Pieces”, Arpaci-Dusseau Books. Versión on-line: <http://ostep.org>**
- A.Silberschatz, P.Galvin, G.Gagne, "Operating System Concepts. 8th Edition", Wiley, 2010. ISBN 978-0-470-23399-3.
- A.S.Tanenbaum, A.S.Woodhull, "Sistemas Operativos. Diseño e Implementación”, Segunda Edición, Prentice-Hall, 1997

h. Bibliografía complementaria

- A.S.Tanenbaum, "Sistemas Operativos Modernos. Tercera Edición.", Pearson/Prentice-Hall, 2009
- A.S.Tanenbaum, A.S.Woodhull, "Sistemas Operativos. Diseño e Implementación. El Libro de MINIX”, Tercera Edición, Prentice-Hall, 2006

i. Recursos necesarios

<http://www.minix3.org>

Aulas Virtuales Escuela de Informática (www.inf.uva.es)

Bloque 3: SeguridadCarga de trabajo en créditos ECTS: **a. Contextualización y justificación**

La garantía de integridad de la información en los sistemas informáticos se basa en dos aspectos básicos, el control interno del acceso a los recursos, protección, y el control de la comunicación entre el sistema informático y el exterior, seguridad. En este bloque se estudia una introducción a los conceptos fundamentales de protección y seguridad, incluyendo la protección de acceso a recursos, el software maligno, los diferentes métodos de ataque a un sistema informático y la seguridad de claves, entre otros conceptos.

b. Objetivos de aprendizaje

- Conocer los servicios que ofrece un SO específicas desde el punto de vista de un usuario, de un programador y de un administrador y utilizar diferentes sistemas de interacción con el SO.
- Diseñar e implementar aplicaciones basadas en los servicios del SO, seleccionando los más adecuados a cada caso.
- Comprender y saber gestionar los sistemas de control de acceso a usuarios y recursos para garantizar un nivel de seguridad adecuado a nivel sistema operativo.

c. Contenidos

- Capítulo 20: seguridad

El problema de la seguridad. Métodos de ataque. Niveles de medida de seguridad. Amenazas basadas en programas. Caballos de Troya. Puertas traseras. Bombas lógicas. Stack overflow. Virus. Amenazas al sistema y a la red. Gusanos. Escaneo de puertos. Denegación de servicio. Criptografía. Cifrado simétrico. Cifrado asimétrico. RSA. SSL. Autenticación de usuarios. Implementación de defensas de seguridad.

Parte práctica

- Prácticas 3 a 7
- Práctica abierta opcional

d. Métodos docentes

- Sesiones de aula: clases magistrales participativas y estudios de caso.
- Prácticas supervisadas en laboratorio con seguimiento en cada sesión de cada estudiante para poder realizar una evaluación continua del apartado práctico.

e. Plan de trabajo

Semana 14:

Aula: capítulo 20

Laboratorio: prácticas 3 a 7. Práctica abierta opcional

f. Evaluación

- Examen escrito semi-objetivo y de solución de problemas para los bloques 2 y 3.
- Evaluación continua para la parte práctica

g. Bibliografía básica

- A.Silberschatz, P.Galvin, G.Gagne, "Operating System Concepts. 8th Edition", Wiley, 2010. ISBN 978-0-470-23399-3.
- A.S.Tanenbaum, A.S.Woodhull, "Sistemas Operativos. Diseño e Implementación", Segunda Edición, Prentice-Hall, 1997

h. Bibliografía complementaria

- A.S.Tanenbaum, "Sistemas Operativos Modernos. Tercera Edición.", Pearson/Prentice-Hall, 2009
- A.S.Tanenbaum, A.S.Woodhull, "Sistemas Operativos. Diseño e Implementación. El Libro de MINIX. Tercera Edición.", Prentice-Hall, 2006

i. Recursos necesarios

Aulas Virtuales Escuela de Informática (www.inf.uva.es)

5. Métodos docentes y principios metodológicos

Se utilizan diferente metodología docente para la teoría y los laboratorios:

- **Teoría.** La metodología docente principal se basa en las clases teóricas. El estudiante deberá acudir a estas clases con una actitud activa plasmada en las siguientes cuestiones:
 - Estudio previo de la materia de cada clase para que le resulte más fácil su comprensión.
 - Toma de notas sobre los aspectos más relevantes.
 - Participación en las preguntas planteadas por el profesor.
 - Resolución de las principales dudas sobre la materia.

Adicionalmente, y de forma optativa, se plantea una metodología basada en el desarrollo de un trabajo teórico/práctico. Se propone al estudiante realizar un trabajo de forma paralela al desarrollo de la docencia y con mucha relación con ésta, de forma que se complementen mutuamente y se mejore el aprendizaje. Este trabajo es valorado en la nota final de la asignatura.

- **Laboratorios.** La metodología está orientada al desarrollo de proyectos y la evaluación es continua. Se plantean dos proyectos sucesivos. El primero, llamado *práctica cerrada*, va guiando al estudiante sobre los diferentes apartados del sistema operativo didáctico Minix, y le va planteando retos a desarrollar, de forma que aprende dicho sistema operativo y mejora sus capacidades de trabajo autónomo y resolución de problemas. El segundo, llamado *práctica abierta*, consiste en un pequeño proyecto a realizar completamente por el estudiante. Se le permite elegir el tema (dentro del contexto Minix/Linux) y su orientación. Se pretende que el estudiante continúe el aprendizaje de forma autónoma (supervisada por el profesor) con los objetivos de que aprenda los detalles de un aspecto concreto de las plataformas de prácticas y aprenda a resolver problemas complejos en dicho contexto.

6. Tabla de dedicación del estudiante a la asignatura

ACTIVIDADES PRESENCIALES	HORAS	ACTIVIDADES NO PRESENCIALES	HORAS
Clases teórico-prácticas (T/M)	28	Estudio y trabajo autónomo individual	65
Clases prácticas de aula (A)		Estudio y trabajo autónomo grupal	25
Laboratorios (L)	30		
Prácticas externas, clínicas o de campo			
Seminarios (S)			
Tutorías grupales (TG)			
Evaluación	2		
Total presencial	60	Total no presencial	90

7. Sistema y Características de la Evaluación

INSTRUMENTO/PROCEDIMIENTO	PESO EN LA NOTA FINAL	OBSERVACIONES
Examen escrito	30%	Teórico semiobjetivo y de resolución de problemas. Correspondiente al bloque 1. Se realizará en la semana 7.

		Este examen elimina materia pero únicamente si el estudiante lo aprueba, es decir, obtiene 5 puntos sobre 10 como mínimo. En caso de aprobar, la nota contabilizará un 30% de la nota final. En caso de no aprobar, este examen se recuperará en el examen final.
Proyecto práctico colaborativo	30%	Correspondiente a los bloques 0, 1, 2, 3 y 4. Evaluación continua.
Examen final ordinario	40% ó 70%	Teórico semiobjetivo y de resolución de problemas. Realizado en periodo ordinario de exámenes. Incluirá todos los bloques temáticos para aquellos alumnos que no hayan superado el examen parcial (70% de peso) o únicamente los bloques 2 y 3 para aquellos alumnos que lo hayan superado (40% de peso). En cualquiera de los dos casos el estudiante debe obtener al menos 5 puntos sobre 10 para que le contabilice el resultado.
Examen final extraordinario	40% ó 70%	Igual que el examen ordinario.

Criterios de calificación:

- **Proyecto práctico:**
 - **Prácticas 1 a 3 (programación de sistemas):** evaluación continua en las sesiones de laboratorio. Entregas del código fuente bien documentado. Sólo en determinados casos concretos se realizará un examen oral (no asistencia al laboratorio, progreso insuficiente).
 - **Prácticas 3 a 7 (Minix):** se realizará evaluación continua a lo largo de todas las sesiones de laboratorio. El estudiante únicamente entregará el cuaderno de bitácora. Sólo en determinados casos concretos se realizará un examen oral (no asistencia al laboratorio, progreso insuficiente).
 - **Práctica abierta:** la evaluación se realizará de la misma forma que la práctica cerrada. Se tendrá en cuenta el diseño y la implementación de la práctica así como su dificultad, corrección funcional, la documentación realizada y el cuaderno de bitácora. En caso necesario también se tendrá en cuenta la defensa que realice de su trabajo.
- **Exámenes escritos:** respuesta limitada a cuestiones teóricas y resolución de problemas. En la resolución de problemas se tendrá en cuenta la línea de razonamiento, los fundamentos teóricos aplicados y la metodología usada para resolverlo. La corrección o incorrección del resultado final del problema tendrá una relevancia menor.

8. Consideraciones Finales

Planificación temporal por bloques temáticos:

BLOQUE TEMÁTICO	CARGA ECTS	PERIODO PREVISTO DE DESARROLLO
Blq. 1: gestión de memoria	3,6	Semanas 1 a 9
Blq. 2: gestión del almacenamiento	2	Semanas 10, 11, 12, 13, 15
Blq. 3: protección y seguridad	0.4	Semana 14



Otras consideraciones:

- En la parte teórica de la asignatura la asistencia a clase no será tomada en cuenta para la evaluación final de la asignatura.
- En la parte práctica se controlará la asistencia al laboratorio y los progresos en la realización de las prácticas. En caso de asistencia y progreso regular, el profesor recogerá y evaluará el trabajo práctico atendiendo a los criterios descritos en el apartado 7. En caso de no asistencia al laboratorio, el profesor, adicionalmente, convocará al estudiante de forma individual para que defienda el trabajo práctico realizado.