Planificación de

Sistemas Operativos I

Código: *R-322*

FACULTAD DE CIENCIAS EXACTAS, INGENIERIA Y AGRIMENSURA

Identificación y características del Espacio Curricular

Carrera/s:	Licenciatura en Ciencias de la Computación		
Plan de Estudios:	2010	Carácter:	Obligatoria
Bloque / Campo:		Área:	Algoritmos y Lenguajes
Régimen de cursado:	Cuatrimestral		
Cuatrimestre / Año:	6to cuatrimestre / 3er año		
Carga horaria (total y semanales):	105 hs. / 7 hs. semanales	Formato curricular:	Asignatura
Escuela:	Ciencias Exactas y Naturales	Departamento:	Ciencias de la Computación
Docente/s responsable/s:	PIRE, Taihú		

Programa Sintético

Programación y procesos. Regiones de memoria. Creación y destrucción de procesos. Sincronizaciones y comunicación. Condiciones de concurso y regiones críticas. Exclusión mútua. Problemas relacionados. Deadlock y livelock. Programación concurrente. Interbloqueos. Formalismos. Bibliotecas de programación paralela y distribuida.

Espacios Curriculares Relacionados

Previas Aprobados:	R-223 Lógica, R-312 Estructura de Datos y Algoritmos II, R313 Análisis de Lenguajes de Programación.
Simultáneas Recomendados:	–.
Posteriores:	R-412 Sistemas Operativos II.

Vigencia desde: 2015

Fundamentación

En la actualidad las arquitecturas de hardware son por defecto paralelas, conocer técnicas adecuadas para programarlas garantiza un mejor aprovechamiento de las mismas. Independientemente de esto, muchas veces es necesario diseñar o interactuar con sistemas de software que son inherentemente concurrentes.
La materia brinda nociones sólidas de los sistemas concurrentes, junto con las técnicas actuales para diseñarlos e implementarlos.
Esto permite cubrir una cuota importante en el acervo que debe tener un egresado de LCC, para afrontar en forma autoasistida su actualización. El enfoque seleccionado es teórico-práctico, enfrentando al alumno a situaciones que demandan el uso de las técnicas presentadas y un proyecto final integrador que será desarrollado de manera colaborativa y servirá a su vez de instancia de evaluación.

Se dicta en el tercer año de la carrera. Los alumnos deben haber superado y adquirido las competencias de las materias de formación en estructura de datos, lógica, algoritmos y lenguajes de programación.

Resultados de aprendizaje

Al terminar con éxito esta asignatura, los estudiantes serán capaces de:

- RA1: Comprender los conceptos básicos de la programación concurrente tales como hilos, procesos y sincronización.
- RA2: Comprender la importancia de la programación concurrente en las aplicaciones de la actualidad.
- RA3: Conocer y entender los problemas que plantea el desarrollo de programas concurrentes y que no aparecen en la programación secuencial.
- RA4: Conocer los principales modelos de programación concurrente, paralela y distribuida.
- RA5: Aplicar correctamente mecanismos de exclusión mutua, sincronización y comunicación entre procesos para evitar condiciones de carrera (race conditions) y otros errores de concurrencia.
- RA6: Medir y comparar el rendimiento de diferentes soluciones a un mismo problema de concurrencia.
- RA7: Concebir y desarrollar sistemas informáticos centralizados o distribuidos.
- RA8: Conocer y ser capaz de usar bibliotecas y plataformas estandarizadas para la implementación de programas concurrentes basados en memoria compartida y para sistemas distribuidos.
- RA9: Diseñar e implementar sistemas concurrentes que sean seguros y eficientes.
- RA10: Identificar las propiedades de safety y liveness que un sistema concurrente debe cumplir y razonar si se cumplen o no.
- RA11: Adquirir experiencia para el trabajo colaborativo e intercambio de ideas.
- RA12: Utilizar adecuadamente herramientas colaborativas para coordinar el trabajo en equipo de forma autónoma.

Competencias / Ejes transversales

La siguiente tabla consigna la relación entre los resultados de aprendizaje y las competencias a las que tributa el espacio curricular.

Competencia/Eje Transversal al que tributa	Nivel	Resultados de Aprendizaje
CGT2. Concepción, diseño y desarrollo de proyectos de informática	2	RA7, RA8, RA9, RA10
CGT4. Utilización de técnicas y herramientas de aplicación en la informática	3	RA5, RA6, RA12
CGT5. Generación de desarrollos tecnológicos y/o innovaciones tecnológicas	2	RA2, RA3, RA7, RA8, RA9, RA10
CGS1. Fundamentos para el desempeño en equipos de trabajo	2	RA11, RA12

Programa analítico

Unidad 1:
1.1. Definición de Sistema operativo.
1.2. Nociones básicas de las funciones de un Sistema Operativo.
1.3. Programación paralela y concurrente. Definiciones. Características. Hitos históricos.

1.4. Ley de Moore.

Unidad 2:
2.1. Definición de proceso. Etapas y regiones de un proceso.
2.2. Garantías del Sistema Operativo. El rol del scheduler de un Sistema Operativo.
2.3. El Sistema Operativo GNU/Linux.
2.3. Llamadas de Sistema. Primitivas de creación y terminación de procesos.
2.4. Información de un proceso monitoreado por el SO. Descriptores de Archivo.
2.5. Comunicación Interprocesos: Señales. Pipes. Shared Memory. Socket locales.

Unidad 3:
3.1 Posix Threads.
3.2. Interliving. Recursos compartidos y Región crítica.
3.3. Condiciones de carrera. Problema del jardín Ornamental.
3.4. El problema de Exclusión mutua.
3.5. Algoritmo de Peterson y Lamport como solución al problema de exclusión mutua. Problemas en arquitecturas modernas. Barreras de memoria. Instrucciones por hardware para exclusión mutua.

Unidad 4:
4.1. Locks y Spinlocks.
4.2. Variables de Condición. Necesidad y uso. Problema de Productores y Consumidores con buffer acotado.
4.3. Monitores. Semántica de Hoare y Mesa.
4.4. Semáforos de Dijkstra. Problema de los escritores y lectores.
4.5. Rendezvous. Problema de la Barbería.
4.6. Barreras de sincronización.
4.7. Deadlocks. Condiciones necesarias para su aparición. Problema de los filósofos comensales. Livelocks.

Unidad 5:
5.1. Mensajes síncronos y asíncronos.
5.2. Modelo CSP. Canales.
5.3. Modelo de Actores. Erlang. Canales en Erlang. Guardas.
5.4. Socket sobre red. Modelo Cliente/Servidor. RPC.
5.5. Modelos de I/O. Kernel queues. Uso de epoll.

Unidad 6:
6.1. Especificación de ejecución concurrente. Condiciones de Bernstein.
6.2. Propiedades de Safety y livenes.
6.3. API OpenMP. Modelo de ejecución Fork-Join. Directivas de sincronización.
6.4. Modelo de programación para memoría distribuida.
6.5. Bibliotecas de paso de mensajes. MPI. Comunicaciones P2P y colectivas.
6.6. Entornos de ejecución distribuidos.
6.7. Medidas de performance. SpeedUp. Eficiencia
6.8. Erlang Distribuido. Servicio de Distribución de Datos (DDS).

Unidad 7:
7.1 Lineamientos para trabajo en equipo
7.2 Herramientas para gestión de tareas y proyectos. Trello.
7.3 Herramientas para trabajo colaborativo. Github. Vscode live share.
7.4 Guías de estilo al programar
7.5 Herramientas de Versionado de código, git. Guías de estilo de versionado.
7.6 Documentación técnica.
7.7 Herramientas para generación automática de documentación. Doxygen. Edoc

Modalidades de enseñanza

El dictado de la asignatura constará de clases teórico-prácticas y de laboratorio. En las clases teóricas se tratan los distintos aspectos conceptuales de las diferentes unidades. Estas clases se complementan con prácticas acordes en laboratorio de informática. La resolución de los problemas se realiza en grupos de 2 integrantes. Se utiliza la entrega de ejercicios seleccionados de forma regular para medir la incorporación adecuada de los conceptos con devolución del trabajo a cada grupo y devolución general de errores comunes.
Realización de trabajo práctico final que engloba los principales temas con seguimiento y guía del cuerpo docente.

En resumen:
- Clases teóricas interactivas (teórico expositivas / teórico prácticas).
- Clases prácticas en laboratorio de informática.
- Resolución de problemas en grupos de 2 integrantes. Devolución del trabajo a cada grupo y devolución general de errores comunes.
- Realización de trabajo práctico final (proyecto) con seguimiento por parte del cuerpo docente.

Recursos

La asignatura se dicta en aula y laboratorios de informática. Se utilizan computadoras de escritorios o laptops, a razón de 1 máquina cada 2 alumnos, con sistema operativo tipo Linux, entorno de programación C (editor de texto, compiladores, debugger, herramientas de profiling, git), Erlang y acceso a internet. También contamos con el acceso a un cluster de computadoras para probar implementaciones distribuidas.
Durante el dictado de clase se utiliza proyector multimedia. Todo el material de estudio se sube al campus virtual (comunidades) provisto por la Universidad para que todos los alumnos tengan acceso al material. Para la comunicación se utiliza un canal de zulip hosteado por el departamento de computación.
Se cuenta además con herramientas de dictado virtual en caso de ser necesario por fuerza mayor (paro de transporte).

Link de la materia: https://dcc.fceia.unr.edu.ar/es/lcc/r322

Actividades de Formación Práctica

N° Actividad	Título	Descripción
1	Procesos y Programación de Sistemas en Unix	Introducción a la programación en entornos linux mediante problemas sencillos. Uso del man. Uso de mecanismos de comunicación entre procesos en programas breves.
2	Multiprocesador y exclusión mutua.	Resolución de problemas básicos de concurrencia con memoria compartida usando Posix Threads. Uso de técnicas de exclusión mutua.
3	Sincronización	Resolución de problemas de sincronización con mutex, semáforos, monitores, canales, variables de condición.
4	Programación Paralela y Distribuida	Programación de aplicaciones paralelas usando OpenMP. Programación de aplicaciones distribuidas con MPI. Se exhiben técnicas para evaluar el impacto en performance de diferentes soluciones. Programación de aplicaciones distribuidas con Erlang.
5	Práctico Final (Proyecto final)	Se requiere el desarrollo de un sistema concurrente donde deban aplicarse las técnicas estudiadas durante el cursado. Se pide una implementación robusta y eficiente. La implementación debe estar correctamente documentada e ir acompañada con un informe donde se justifiquen las decisiones de diseño. Al desarrollo del proyecto se agrega el uso de una herramienta para gestión de proyectos colaborativa donde el cuerpo docente puede observar las tareas que va desarrollando el grupo de trabajo así como la auto organización de sus integrantes.

Cronograma de actividades

SEMANA	UNIDAD	CONTENIDO	ACTIVIDADES TEÓRICAS - PRÁCTICA
01	1 y 7	Unidad 1: Introducción a Sistemas Operativos Unidad 7: Lineamientos para trabajo en equipo	Definiciones. Funciones de un Sistema Operativo. Programación Concurrente y paralela. Hitos históricos. Ley de Moore. **Armado de grupos. actividad asíncrona.**
02	2	Procesos: Ciclo de Vida, Creación de Procesos	Definiciones. Etapas de un proceso. Garantías de sistema operativo. GNU/Linux. Llamadas al sistema. Primitivas Fork y Exec. Información de un proceso monitoreada por el SO. Descriptores de archivo. **Práctica 1 en laboratorio.**
03	2 y 7	Unidad 2: Comunicación entre Procesos. Unidad 7: Herramientas de versionado.	IPC, Signals, Pipes. Shared Memory. Socket locales. Git. Guías de estilo al programar y versionar el código. **Práctica 1 en laboratorio.**
04	3	Multiprogramación y el problema de exclusión mutua.	Posix Threads. Interliving. Recursos compartidos y región crítica. Problema de Exclusión Mutua: Condición de Carrera, Operación Atómica,Problema del Jardín Ornamental. Sección Crítica. **Práctica 2 en laboratorio.** **Entrega Práctica 1.**
05	3 y 7	Unidad 3: Sincronización entre Procesos. Unidad 7: Documentación técnica.	Algoritmo de Peterson, Algoritmo de Lamport, Consistencia Secuencial, Ejecución Fuera de Orden, Barrera de Memoria **Práctica 2 en laboratorio.**
06	4	Mecanismos de sincronización	Locks y Spinlocks. **Práctica 2 en laboratorio.**
07	4	Mecanismos de sincronización	Variables de Condición. Necesidad y uso. Problemas de Productores y Consumidores con buffer acotado. Monitores. Semántica de Hoare y Mesa. **Práctica 3 en laboratorio.** **Entrega Práctica 2**
08	4	Mecanismos de sincronización	Semáforos de Dijkstra. Problemas de los escritores y lectores. Rendezvous. Problema de la barbería. **Práctica 3 en laboratorio.**
09	4	Mecanismos de sincronización	Barreras de sincronización. Deadlocks. Condiciones para su aparición. Problemas de los filósofos comensales. Livelocks. **Práctica 3 en laboratorio.**
10	5	Comunicación mediante mensajes	Mensajes síncronos y asíncronos. Socket sobre red. Modelos Cliente servidor. RPC. Modelos de I/O Kernel queues. Epoll. **Práctica 3 en laboratorio.**
11	5	Comunicación mediante mensajes	Modelo de CSP. Canales.Modelo de Actores. Erlang. Canales en Erlang. Guardas. **Práctica 3 en laboratorio.** **Entrega Práctica 3.**
12	6	Computación Paralela y Distribuida	Especificación de ejecución concurrente.Condiciones de Bernstein. Propiedades de Safety y Livenes. **Parcial**
13	6	Computación Paralela y Distribuida	Modelo de ejecución Fork-Join. Api OpenMP. Directivas de sincronización. Modelo de programación para memoria distribuida. MPI. Comunicación P2P y colectivas. Medidas de performance y eficiencia. **Práctica 4 en laboratorio.**
14	6	Programación Distribuida y tolerante a fallas Erlang	Entornos de ejecución distribuídos. Clusters. Erlang distribuído. Programación tolerante a fallos. Servicio de Distribución de datos (DDS). **Práctica 4 en laboratorio.**
15	7	Herramientas para Trabajo Colaborativo	Herramientas para gestión de tareas y proyectos. Trello. Github. Presentación del práctico final. Herramientas de generación automática de documentación. Doxygen. Edoc **Entrega Práctica 4.**

Evaluación

La evaluación de los alumnos se realizará en forma continua durante el desarrollo de las clases, siendo evaluado su desempeño durante la implementación de diferentes prácticos y una instancia de práctico final donde integran y profundizan los temas aprendidos durante el cursado.

En la tabla se muestra la relación que existe entre resultados del aprendizaje esperados y las distintas actividades e instancias de evaluación.

.

Resultado de Aprendizaje	Actividades/Modalidad de Enseñanza	Modalidad de Evaluación
RA1	Actividad 1 y 2. Clases Teóricas Interactivas y Prácticas en Laboratorio.	Entrega de ejercicios seleccionados forma grupal de ejercicios seleccionados de la Actividad 1 y 2. Parcial escrito individual.
RA2	Actividad 2 y 3. Clases Teóricas Interactivas y Prácticas en Laboratorio.	Entrega de ejercicios seleccionados forma grupal de ejercicios seleccionados de la Actividad 2 y 3. Coloquio oral
RA3	Actividad 2, 3, 4, y 5. Clases Teóricas Interactivas y Prácticas en Laboratorio.	Entrega de ejercicios seleccionados forma grupal de ejercicios seleccionados de la Actividad 2, 3 y 4. Coloquio oral.
RA4	Actividad 4. Clases Teóricas Interactivas y Prácticas en Laboratorio.	Entrega de trabajos prácticos de forma grupal de ejercicios teórico-prácticos seleccionados de la Actividad 4. Coloquio oral
RA5	Actividad 2, 3, 4, y 5. Clases Teóricas Interactivas y Prácticas en Laboratorio.	Entrega de ejercicios seleccionados forma grupal de ejercicios seleccionados de la Actividad 2, 3 y 4. Examen Parcial escrito individual de la Unidad 1 a 4.
RA6	Actividad 3 y 4. Clases prácticas en el Laboratorio.	Entrega de trabajos prácticos de forma grupal de ejercicios seleccionados de la Actividad 3 y 4.
RA7	Actividad 5.	Entrega de trabajo práctico final grupal. Defensa del trabajo práctico final (proyecto) individual.
RA8	Actividad 4. Clases Teóricas Interactivas y Prácticas en Laboratorio.	Entrega de trabajos prácticos de forma grupal de ejercicios seleccionados de la Actividad 4.
RA9	Actividad 5.	Entrega de trabajo práctico final grupal. Se prueba la robustez y eficiencia a través de tests automáticos. Defensa de decisiones de diseño del trabajo práctico final (proyecto) individual.
RA10	Actividad 3. Clases Teóricas Interactivas y Prácticas en Laboratorio. Parcial	Entrega de trabajos prácticos de forma grupal de ejercicios seleccionados de la Actividad 3. Parcial escrito individual. Entrega de trabajo práctico final grupal. Se prueba la robustez y eficiencia a través de tests automáticos.
RA11	Entrega de trabajos prácticos de forma grupal de ejercicios seleccionados de la Actividad 1, 2, 3, 4 y 5.	Entrevista oral.

RA12	Entrega de trabajos prácticos de forma grupal de ejercicios seleccionados de la Actividad Actividad 5.	Observación a través de la bitácora de las herramientas colaborativas utilizadas. (git, trello, etc).

Condiciones de Regularización:
- Aprobar las entregas de las 4 prácticas grupales
- Aprobar con nota 6 o más el parcial.

El requerimiento de la aprobación de las 4 prácticas grupales se debe a que cada práctica evalúa conceptos distintos.

Condiciones de Aprobación:

- **Para Regulares**
  - Aprobar práctico final (proyecto)
  - Coloquio
- **Para Libres:**
  - Con prácticas aprobadas:
    - Aprobar un examen escrito teórico práctico escrito
    - Aprobar el práctico final grupal
    - Coloquio
  - Sin prácticas aprobadas:
    - Aprobar un examen escrito teórico práctico escrito
    - Aprobar ejercicios prácticos en laboratorio
    - Aprobar el práctico final grupal
    - Coloquio I

Distribución de la carga horaria

Presenciales		**Horas**
Teóricas		42
Prácticas	Formación Experimental	0
	Resolución de problemas y Ejercicios	0
	Resolución de Problemas vinculados a la Profesión	20
	Actividades de Proyecto y Diseño de Sistemas Informáticos	40
	Práctica Profesional	0
Evaluaciones		3
	Total	105

Dedicadas por el alumno fuera de clase	**Horas**
Preparación Teórica	20
Preparación Práctica	20
Elaboración y redacción de informes, trabajos, presentaciones, etc.	35
Total	75

Bibliografía básica

Autores (Apellido, Inicial Nombre.)	Año de edición	Título de la obra	Editorial o Revista	Ejemplares disponibles o sitio web de acceso
Silberchatz et al	2006	Fundamentos de Sistemas Operativos	Mc Graw Hill	1
Andrews, G. R.	2000	Foundations of Multithreaded, Parallel, and Distributed Programming. ISBN 0-201-35752-6	Adisson-Wesley	1
Downey, A. B	2016	The Little Book of Semaphores		https://greenteapress.com/wp/semaphores/[1]
Galli, R.	2015	Principios y algoritmos de concurrencia	Createspace	https://gallir.wordpress.com/principios-de-concurrencia/[2]
Armstrong, J et all	2006	Concurrent Programing in Erlang (2nd Ed.)	Prentice-Hall	1
Raynal, M.	2013	Concurrent Programming: Algorithms, Principles, and Foundations. ISBN 978-3-642-32027-9	Springer	1

Bibliografía complementaria

Autores (Apellido, Inicial Nombre.)	Año de edición	Título de la obra	Editorial o Revista	Ejemplares disponibles o sitio web de acceso
Herlihy, M. ; Shavit, N.	2012	The Art of Multiprocessor Programming. ISBN 978-0123973375	Elsevier	1
McKenney, P. E	2022	Is Parallel Programming Hard, And, If So, What Can You Do About It?		https://mirrors.edge.kernel.org/pub/linux/kernel/people/paulmck/perfbook/perfbook.html[3]
Raymond, E.	2003	The Art of Unix Programming ISBN0-13-142901-9	Addison-wesley	http://www.catb.org/~esr/writings/taoup/html/[4]
Wolf, Ruiz, Bergero, Meza	2015	Fundamentos de Sistemas Operativos. ISBN 978-607-02-6544-0	Universidad Nacional Autónoma de México	1 http://ru.iiec.unam.mx/2718/[5]

[1] Accedido por última vez septiembre 2023. Con licencia Creative Commons (CC BY-NC-SA 4.0).
[2] Accedido septiembre 2023. Con licencia Creative Commons (CC BY-SA 3.0 ES).
[3] Accedido septiembre 2023. Con licencia Creative Commons (CC BY-SA 3.0 US).
[4] Accedido septiembre 2023. Con licencia Creative Commons (Attribution-NoDerivs 1.0).
[5] Accedido septiembre 2023. Con licencia Creative Commons (CC BY-SA versión 4.0).