PROGRAMA de Programación Concurrente

Carreras: Tecnicatura Universitaria en Programación Informática - Licenciatura en Informática - Licenciatura en Bioinformática

Asignatura: Programación Concurrente

Núcleo al que pertenece: Básico

Profesores: Daniel Ciolek y Pablo Terlisky

Asignaturas Correlativas: Estructuras de Datos

Objetivos:

- Presentar los conceptos básicos y los modelos más utilizados asociados a la programación concurrente.
- Estudiar las dificultades asociadas a la interacción de componentes en un sistema concurrente y los recursos de los que se disponen para mitigarlos.
- Analizar programas concurrentes adecuadamente.
- Desarrollar aplicaciones concurrentes sencillas.

Contenidos mínimos:

- Los porqués de la concurrencia. Concurrencia vs Paralelismo.
- Modelo de memoria compartida, atomicidad e independencia.
- Secciones críticas, locks y barreras, semáforos, monitores y variables de condición, rendezvous.
- Problemas de concurrencia: Starvation, Deadlocks, Liveness y Progress, Safety, Race conditions, Fairness.
- Modelo de pasaje de mensajes: Comunicación sincrónica vs comunicación asincrónica, Modelo de transacciones.
- Modelos de interacción: Cliente/Servidor, Productor/Consumidor.
- Aplicación de los conceptos estudiados en lenguajes de programación concretos, mecanismos de sincronización.

Carga horaria semanal: 4 hs

Programa analítico:

Unidad 1: Los porqués de la concurrencia.

Concurrencia vs Paralelismo. Trazas. Scheduling cooperativo vs expropiativo. Hilos, procesos y subprocesos.

Unidad 2: Modelo de memorias

Memoria compartida, atomicidad e independencia. Fenómeno de interferencia. Secciones críticas y exclusión mutua. Algoritmos de Peterson, Dekker y Lamport.

Unidad 3: Problemas de concurrencia

Starvation, Deadlocks, Liveness y Progress, Safety, Race Conditions, Fairness. Soluciones mediante rupturas de simetrías. Invariantes de programas concurrentes.

Unidad 4: Secciones críticas con semáforos.

Mutexes. Ciclo de vida y estados de un proceso. Problemas de sincronización. Productor-consumidor y lector-escritor con semáforos.

Unidad 5: Monitores y variables de condición.

Locks. Deadlock por anidamiento de monitores. Estructuras de datos concurrentes. Thread pools, execution services, pipes and filters.

Unidad 6: Modelo de pasaje de mensajes

Comunicación sincrónica vs comunicación asincrónica, rendezvous. Movilidad. Datos inmutables. Estrategias ante fallas en entornos distribuidos.

Unidad 7: Modelos de interacción

Cliente-Servidor, Productor-Consumidor, Lector-Escritor. Modelo de actores. Modelo de transacciones. Tuplas de Linda. Join calculus. Vectorización.

Bibliografía obligatoria:

- M. Ben-Ari, Principles of Concurrent and Distributed Programming, Pearson. 2nd. Edition. 2006.
- Michel Raynal. Concurrent Programming: Algorithms, Principles and Foundations. Springer-Verlag Berlin Heidelberg, 1st Edition, 2013 (Estaba como bibliografía de consulta).
- Doug Lea. Concurrent Programming in Java: Design Principles and Patterns. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA. 2nd. Edition 1999.

Bibliografía de consulta:

- No posee

Organización de las clases:

Las clases se organizan como teórico-prácticas. Durante las clases se presenta el contenido formal y se ahonda en las características de la programación concurrente.

Las clases son acompañadas por un apunte en forma de diapositivas que sirve como material de referencia más allá de la bibliografía sugerida. Luego de la presentación de cada tema se resuelven ejercicios de programación haciendo hincapié en las características de la programación concurrente. Los ejercicios son discutidos y resueltos en el pizarrón, luego puede probarse su funcionamiento en las computadoras del laboratorio. Además, se responden consultas, principalmente las generadas por los trabajos prácticos de la materia.

Respecto las actividades extra-áulicas obligatorias, se deberán realizar:

- Resolución de ejercicios (consolidación de conceptos y preparación para los exámenes, no participa en la nota final).
- Trabajos prácticos (aplicación práctica de los conceptos estudiados en un caso de mayor envergadura, no participa en la nota final)

Trabajos Prácticos

Se cuenta con guías de Trabajos Prácticos (TP), que requieren la resolución de problemas de análisis de trazas y programación concurrente con distintas herramientas de sincronización. Previo a la realización de cada TP se proporciona una explicación. El trabajo se realiza en grupos de dos, tanto en las tareas de programación como en la elaboración de informes. Dependiendo del TP se realizan evaluaciones cualitativas y/o cuantitativas de los resultados obtenidos.

Los trabajos prácticos a desarrollar son:

1. **Análisis de trazas**: Se analizan trazas generadas por distintos programas concurrentes simples donde se evidencia el fenómeno de interferencia en el acceso concurrente a memoria compartida.
2. **Exclusión mutua**: Se analizan propuestas de solución al problema de la exclusión mutua donde las únicas operaciones atómicas son las de lectura y escritura de memoria global.
3. **Instrucciones atómicas**: Se analizan propuestas de solución al problema de la exclusión mutua introduciendo distintas operaciones atómicas.
4. **Sincronización mediante semáforos**: Se implementan soluciones a problemas simples de programación concurrente utilizando semáforos como herramienta de sincronización.
5. **Esquemas con semáforos**: Se implementan soluciones a problemas recurrentes en el área de programación concurrente (a saber esquema de Productor-Consumidor y Lectores-Escritores) utilizando semáforos como herramienta de sincronización.
6. **Sincronización mediante monitores**: Se implementan estructuras de datos mediante el uso de monitores, de forma tal que permitan la sincronización de múltiples procesos cooperativos.
7. **Esquemas con monitores**: Se implementan soluciones a problemas recurrentes en el área de programación concurrente (a saber Thread-Pool,

Worker-Threads y Promises-Futures) aplicando monitores como herramientas de sincronización.

8. **Intercambio de mensajes**: Se implementan soluciones a problemas de programación concurrente mediante la comunicación de procesos sin memoria compartida utilizando canales de comunicación y soluciones esquema con orientación a conexión.

Modalidad de evaluación:

Los mecanismos de evaluación en modalidades libre y presencial de esta asignatura están reglamentados según los siguientes artículos del Régimen de estudios de la UNQ (Res. CS 201/18). En la modalidad de libre, se evalúan los contenidos de la asignatura con un examen escrito, un examen oral e instancias de evaluación similares a las realizadas en la modalidad presencial.

CRONOGRAMA TENTATIVO

Semana	Tema/unidad	Actividad*				Evaluación
		Teórico	Práctico			
			Res Prob.	Lab.	Otros Especificar	
1	Introducción a la programación concurrente	X				
2	Exclusión mutua	X	X			
3	Instrucciones atómicas	X	X			
4	Semáforos	X	X			
5	Esquemas de resolución con semáforos	X	X			
6	Consultas y ejercitación			X	X	
7	Primer parcial				X	X
8	Monitores	X	X			
9	Ejercicios avanzados con monitores		X	X		
10	Resolución de trabajo práctico en clase			X		
11	Mensajes	X	X			
12	Entrega y defensa del trabajo práctico					X
13	Temas avanzados de concurrencia	X				
14	Consultas y recuperatorio del trabajo práctico				X	X
15	Segundo Parcial					X
16	Consultas y repaso general sobre las correcciones				X	
17	Recuperatorio del primer parcial					X
18	Recuperatorio del segundo parcial					X
19	Integrador					X

***INDIQUE CON UNA CRUZ LA MODALIDAD**

Firmado digitalmente por: SUAREZ Mariana Alejandra
Directora del Departamento de Ciencia y Tecnologla
UNIVERSIDAD NACIONAL DE QUILMES