Universidad Nacional Autónoma de México

Facultad de Ingeniería

Sistemas Operativos

Proyecto 1:Ejercicio de Sincronización de Nuestro día a día.

Acosta Jacinto Alan, "320101179"

Rubio Carmona Jose Angel, "320118937"

Sexto semestre.

Fecha de entrega : 8 de abril de 2025

Profesora: Gunnar Eyal Wolf Iszaevich

Semestre 2025-2

Ciudad de México, 8 de abril de 2025.

Planteamiento del proyecto

Un problema del día a día:Recarga de tarjetas en el metro

El objetivo del proyecto, es visualizar una problemática casual o no tan casual del mundo real.Es costumbre que el metro de la ciudad de México tenga fallas, pero sus equipos igual, uno de ellos son las máquinas de recargas, el cual actualmente son la única forma de recarga físicamente, para ello trabajaremos con python, debido a que ese lenguaje se no es más entendible para el uso de sincronización.

Normalmente al recargar debes formarte, pero al hacerlo puede que falle la máquina, o que simplemente tarde el usuario(muy típico), por ello se planteó este ejercicio , ya que es un problema del día a día, más hablando de los estudiantes, los cuales se trasladan por este medio de transporte para llegar al escuela diariamente.

Donde para este proyecto se buscará emular la experiencia de recarga de tarjetas en un escenario concurrido como el metro, utilizando concurrencia y sincronización para modelar el comportamiento de múltiples usuarios que esperan su turno, recargar su tarjeta, y enfrentan posibles fallos en el proceso.

Problemática

En esta tarea modelamos el problema de una estación de recarga de tarjetas de transporte, en la cual llegan múltiples usuarios con la intención de recargar su tarjeta. En esta estación únicamente hay 2 máquinas disponibles, y cada máquina solo puede atender a una persona a la vez.

Para mantener el orden, los usuarios deben hacer una fila única, y se deben respetar estrictamente los turnos. Además, existe un riesgo: la máquina podría fallar al leer la tarjeta del usuario (con una probabilidad del 15%). En caso de fallo, el usuario deberá volver a formarse desde el final de la fila para intentar nuevamente.

El objetivo de esta simulación es garantizar que todos los usuarios recarguen eventualmente, con un control justo, concurrente y sin que el sistema se quede colgado.

Funcionamiento del código

El código fue implementado en Python, haciendo uso de la librería threading y estructuras de sincronización como semáforos (Semaphore), condiciones (Condition) y colas (Queue) para lograr una ejecución concurrente justa y controlada.

Control de concurrencia y sincronización

- Semáforos de máquinas: Se crean 2 semáforos, uno por máquina, con valor inicial 1, lo cual garantiza exclusividad de acceso por parte de un usuario a la vez.
- Condición y cola de espera: La estructura Queue mantiene el orden FIFO de llegada de los usuarios. La Condition asegura que solo el primer usuario en la fila pueda intentar recargar si una máquina está disponible. Los demás deben esperar su turno.
- Control de reintentos y finalización: Si un usuario falla en su intento de recarga, se reinserta al final de la fila, y su hilo continúa activo hasta que finalmente logra una recarga exitosa.

Creación y comportamiento de hilos

Cada usuario es representado por un hilo independiente que:

1. Se forma en la fila.
2. Espera su turno.
3. Toma una máquina si está disponible.
4. Recarga (con posibilidad de fallo).
5. Termina sólo cuando se recarga exitosamente.

Este comportamiento se logra a través de un ciclo interno que mantiene al hilo activo hasta lograr la recarga.

Simulación de tiempos

La función time.sleep(random.uniform(1.5, 3.5)) simula el tiempo que tarda un usuario en recargar. También se usa time.sleep(0.5) para simular la llegada escalonada de los usuarios a la fila.

¿Qué ejecuta el código?

El programa lanza 10 hilos de usuarios, numerados del 1 al 10, que llegan con un intervalo breve entre sí. Cada uno:

- Se forma en la fila.
- Espera hasta que sea su turno y haya una máquina libre.
- Realiza la recarga con una posibilidad de fallo.
- Vuelve a formarse si falla, o termina si tiene éxito.

Durante la ejecución se imprime el estado de cada usuario y máquina, y al finalizar, el sistema muestra un resumen con el número de intentos fallidos por usuario.