LABORATORIO 1: GESTIÓN DE PROCESOS

Estados de Procesos

Se desarrolló un script en Python para simular el ciclo de vida de un proceso: Nuevo, Listo, Ejecutando, Bloqueado y Terminado. Cada estado fue acompañado por capturas del Administrador de tareas en la pestaña Detalles para observar el proceso 'python.exe'. El uso de CPU y memoria se monitoreó en cada etapa.

Para observar los estados del ciclo de vida de un proceso, se desarrolló un script Python que simula cada estado:

- Nuevo: cuando el proceso es creado, pero aún no se ejecuta.
- Listo: cuando está preparado para ejecutarse.
- Ejecutando: cuando está usando la CPU.
- Bloqueado: cuando espera un recurso (simulado con sleep).
- Terminado: al finalizar su ejecución.

Se utilizaron capturas del Administrador de tareas (pestaña "Detalles") para evidenciar la existencia del proceso python.exe en cada etapa. El uso de CPU y memoria fue monitoreado.

Este ejercicio permite visualizar cómo el sistema operativo gestiona los distintos estados internos de un proceso a lo largo de su ciclo de vida.

Este experimento permitió visualizar cómo el sistema operativo administra los distintos estados de un proceso a lo largo de su ejecución, mostrando los cambios en el comportamiento del sistema en términos de consumo de recursos.

Scheduling de un Sistema Operativo

Se ejecutaron 5 procesos intensivos en CPU de forma simultánea. Cada uno fue iniciado desde el símbolo del sistema, provocando una carga elevada sobre el procesador. El Administrador de tareas evidenció que el uso de CPU alcanzó el 100% y que los 5 procesos 'python.exe' recibían una distribución más o menos equitativa del procesador.

El uso de CPU está distribuido entre los procesos, con cada uno utilizando un porcentaje similar del CPU total. Esto indica que el sistema operativo está alternando el tiempo entre ellos de forma balanceada.

La pestaña Rendimiento-CPU muestra que la CPU trabaja constantemente al 100% y la frecuencia del procesador es de 3.20 GHz (plena capacidad).

Comparando los algoritmos teóricos, en FIFO, un proceso se ejecuta hasta terminar y luego pasa al siguiente. Esto no ocurre aquí, ya que todos los procesos activos comparten el CPU al mismo tiempo. Mientras que en Round Robin El sistema operativo asigna a cada proceso un pequeño intervalo de tiempo (quantum) y luego rota al siguiente. Lo que seria nuestro caso en Windows 10

Creación de Deadlock

Se simuló un escenario de deadlock con dos procesos que acceden a recursos compartidos en orden inverso. Ambos procesos quedaron esperando indefinidamente, sin liberar los recursos, lo que generó un bloqueo mutuo. El Administrador de tareas mostró los procesos activos, sin uso de CPU, confirmando el estado de espera infinita.

El sistema operativo no resolvió el deadlock por sí solo, evidenciando que este tipo de situaciones deben ser prevenidas por el programador. Se propusieron soluciones como el acceso ordenado a recursos y cerrar el programa desde el administrador de tareas.

Capturas y documentación en la Carpeta Capturas.