# Simulador de un Kernel

## Partes 1 y 2

Sistemas Operativos

11/11/2022

Ian Fernandez Hermida

# Índice

# Introducción

Esta práctica se realiza para la asignatura de Sistemas Operativos de tercer curso de Ingeniería Informática. En esta se intenta simular de forma muy sencilla el comportamiento de un Kernel Monolítico. El objetivo de la práctica, es conseguir realizar el procesamiento paralelo de múltiples tareas simultáneamente.

# Objetivos

Para el correcto desarrollo de esta , se han repartido las tareas a realizar en distintas partes. En esta etapa del proyecto se tratarán las partes uno y dos.

En la primera parte se deben implementar un clock (reloj) y varios temporizadores (timers) y conseguir que se sincronicen entre ellos.

En la segunda parte se hace uso del reloj y los temporizadores previamente implementados. En esta fase hay que simular un scheduler y un dispatcher

# Desarrollo

## 1. Primera parte

Para poder implementar el reloj y el temporizador en está fase primero hay que comprender el correcto funcionamiento de cada una de estas piezas. El reloj es un componente que emite pulsos a una cierta frecuencia y el temporizador es el encargado de recibir esos pulsos y contarlos.

Para sincronizar estos dos componentes es necesario implementar un mutex condicional con el que se garantiza la exclusión mutua y se consigue tener los datos perfectamente distribuidos. Con este método se logra sincronizar no uno, sino múltiples temporizadores junto a él mismo reloj.

Son tres funciones las realizadas para simular reloj y los dos timers. La función del clock produce y la de los timers consumen. Se hace uso de la librería pthreads y serán hilos simulados los que ejecuten cada una de estas funciones.

Una vez sincronizados estos dos elementos, son necesarios dos temporizadores, uno contabiliza cada cuanto se genera un proceso y el otro lo usa el scheduler para identificar el tiempo que un proceso ha de estar en ejecución.

Por último, crear las estructuras necesarias para crear los procesos, en este caso, una PCB (Process Control Block) que para esta parte solo almacenará el id del proceso. Este id obtiene como identificador el número de proceso.

## 2. Segunda parte

No he tenido tiempo de implementar la segunda parte del proyecto, pero esta es una aproximación de lo que habría que realizar. Primero crear una cola de procesos donde se almacenen las PCBs de cada proceso generado. A continuación se necesita una función que realice la tarea del scheduler es decir que saque estos procesos de la cola y los introduzca en el procesador. Por cada proceso, crear un hilo que será el que se ocupe de la ejecución del proceso en cuestión.

El dispatcher será el encargado de sacar los procesos de memoria por lo que estos deberán tener un tiempo establecido que indique cuánto tiempo han de estar en ejecución (quantum).

Para finalizar, arbitrariamente unos procesos finalizarán mientras que otros volverán a entrar en la cola del scheduler.

# Conclusión

Pese a no haber terminado la segunda parte por falta de conocimientos frente al lenguaje en el que se presenta el proyecto y carencias en la estructura, creo que he logrado el objetivo de la primera parte en cuanto a la parte de sincronización y espero poder realizar de la misma forma la segunda.