PROYECTO FINAL SISTEMAS OPERATIVOS

Maximiliano Soto Jiménez

Universidad Veracruzana

08 de enero de 2024

Introducción

En la materia de sistemas operativos, se vio el administrador del procesador, el cual se encarga de administrar y planificar la CPU de la computadora, en el cual se ejecutan los procesos y los hilos de las aplicaciones y programas de la computadora.

Los hilos en Java son una característica que permite la ejecución simultánea de dos o más partes de un programa para la máxima utilización de la CPU. Cada hilo representa un subproceso y es un código en ejecución dentro de un proceso. La máquina virtual Java (JVM) es capaz de manejar multihilos y puede crear varios flujos de ejecución de manera simultánea, administrando los detalles como asignación de tiempos de ejecución o prioridades de los hilos, de forma similar a como administra un sistema operativo múltiples procesos.

En este documento se utilizo el esquema de planificación de Round Robin usando hilos en Java, para realizar un pequeño código en el que se asigna la CPU entre 3 hilos utilizando dicho esquema de planificación.

Código completo

```java
import java.util.LinkedList;
import java.util.Queue;

public class ProyectoSOIndividual implements Runnable{
    Queue<String> colaHilos;
    int quantum;

    public ProyectoSOIndividual(int quantum){
        this.colaHilos= new LinkedList<>();
        this.quantum=quantum;
    }
    public void agregarHilo(String hilo) {
        colaHilos.add(hilo);
    }
    public void run(){
        while(!colaHilos.isEmpty()){
            String hiloActual= colaHilos.poll();
            int tiempoDeEjecución= Math.min(quantum, hiloActual.length());
            System.out.println("Ejecutando el Hilo: " + Thread.currentThread().getName()+ " por "+ tiempoDeEjecución+" unidades de tiempo
            hiloActual=hiloActual.substring(tiempoDeEjecución);

            if (!hiloActual.isEmpty()) {
                colaHilos.add(hiloActual);
            }
        }
    }
    Run | Debug
    public static void main(String[] args) {
        ProyectoSOIndividual hilos= new ProyectoSOIndividual(quantum:4);
        Thread h1 = new Thread(hilos,name:"Hilo 1");
        Thread h2 = new Thread(hilos,name:"Hilo 2");
        Thread h3= new Thread(hilos,name:"Hilo 3");
        hilos.agregarHilo(hilo:"Hilo 1");
        hilos.agregarHilo(hilo:"Hilo 2");
        hilos.agregarHilo(hilo:"Hilo 3");
        h1.start();
        h2.start();
        h3.start();

    }
}
```

Análisis del código

1. Importes de las clases utilizadas

```java
import java.util.LinkedList;
import java.util.Queue;
```

En este caso, se utilizo LinkedList para simular el agregar hilos la cola de los hilos listos, mientras que se usó Queue para simular dicha cola.

2.Creacion de la clase y sus atributos y constructor

```java
public class ProyectoSOIndividual implements Runnable{
    Queue<String> colaHilos;
    int quantum;

    public ProyectoSOIndividual(int quantum){
        this.colaHilos= new LinkedList<>();
        this.quantum=quantum;
    }
```

La clase ProyectoSOIndividual implementa la interfaz Runnable para crear los hilos. Tiene dos atributos, un entero que será el quantum de tiempo que los hilos tendrán disponible la CPU y una cola tipo Queue llamada cola Hilos, simulando la cola en la que los hilos estarán esperando la CPU. El constructor inicializa dichas variables.

3.Métodos de hilos

```java
    public void agregarHilo(String hilo) {
        colaHilos.add(hilo);
    }
    public void run(){
        while(!colaHilos.isEmpty()){
            String hiloActual= colaHilos.poll();
            int tiempoDeEjecución= Math.min(quantum, hiloActual.length());
            System.out.println("Ejecutando el Hilo: " + Thread.currentThread().getName()+ " por "+ tiempoDeEjecución+" unidades de tiempo
            hiloActual=hiloActual.substring(tiempoDeEjecución);

            if (!hiloActual.isEmpty()) {
                colaHilos.add(hiloActual);
            }
        }
    }
```

En el primer método, el programa agrega un hilo a la cola de los hilos en espera para el Round Robin, el método run es de la interfaz implementada Runnable y es la que se especifica el Round Robin, verificando si la cola tiene algún hilo de ser así, el sistema empieza el esquema usando el tiempo entero introducido como quantum y el cómo se procesa el hilo durante dicho tiempo, después analiza si el hilo termino su ejecución, de no hacerlo, lo volverá a colocar en la cola. Una vez que se proceso el hilo, se envía un mensaje que muestra qué hilo se ejecutó por cuanto tiempo.

4. Clase Main

```
    Run | Debug
    public static void main(String[] args) {
        ProyectoSOIndividual hilos= new ProyectoSOIndividual(quantum:4);
        Thread h1 = new Thread(hilos,name:"Hilo 1");
        Thread h2 = new Thread(hilos,name:"Hilo 2");
        Thread h3= new Thread(hilos,name:"Hilo 3");
        hilos.agregarHilo(hilo:"Hilo 1");
        hilos.agregarHilo(hilo:"Hilo 2");
        hilos.agregarHilo(hilo:"Hilo 3");
        h1.start();
        h2.start();
        h3.start();

    }
}
```

En la clase main, se crea un objeto clase ProyectoSOIndividual llamada hilos, el cual solicita un quantum de tiempo fijo, después crea 3 hilos, usando el esquema hilos, y con nombre “Hilo N” con N siendo un numero entero, después agrega estos hilos usando el método, para finalizar, corre los hilos, los cuales se rigen por el esquema Round Robin.

Resultados

```
Ejecutando el Hilo: Hilo 3 por 4 unidades de tiempo.
Ejecutando el Hilo: Hilo 2 por 4 unidades de tiempo.
Ejecutando el Hilo: Hilo 1 por 4 unidades de tiempo.
Ejecutando el Hilo: Hilo 3 por 2 unidades de tiempo.
Ejecutando el Hilo: Hilo 2 por 2 unidades de tiempo.
Ejecutando el Hilo: Hilo 1 por 2 unidades de tiempo.
PS C:\Users\Maxim\OneDrive\Documents\LIS Ago 2023-Ene 2024\Repaso SO>
```

Al ejecutar el programa con un tiempo quantum de 4, los hilos 3,2,1 se ejecutan en ese orden, procesándose durante 4 unidades de tiempo, al no finalizar ninguno, se vuelven a ingresar a la cola, por lo que se vuelven a procesar, solo que esta vez, finalizan su ejecución en menos de lo que dura el quantum, por lo que os 3 finalizan antes de acabar su segundo quantum de tiempo.