Instituto Tecnológico Superior De Pátzcuaro

Sistemas Operativos I

Practica 2: multitarea

Equipo 3:

Jaime Ariel Ávila García

Juan Eduardo Calderón Servín

Rafael González Isasaga

Profesor: MTI Lenin López Fernández De Lara

Introducción

Todas las personas que utilizan con frecuencia un equipo de cómputo habrán experimentado la situación incómoda en que algún programa no responde y no hay otra manera de solucionarlo que cerrando dicha aplicación o bien reiniciando su ordenador. Pues la causa de estos fenómenos en informática a nivel de sistema operativo se conoce como interbloqueos.

Lo que sucede detrás de nuestras coloridas ventanas es que al momento en que dos procesos requieren del mismo recurso, en algunos casos es imposible asignarlo a los dos procesos, asi que uno de los dos procesos dependiendo de el algoritmo de resolución del problema en cuestión se quedará esperando el recurso, es entonces cuando el programa eja de responder a las peticiones del usuario.

En el siguiente documento analizaremos de manera un poco mas profunda los interbloqueos, sus causas, sus posibles soluciones y las formas de protegernos de dichos eventos desafortunados.

Índice

Desarrollo

Multitarea

El término multitarea se refiere a la capacidad del Sistema Operativo para correr mas de un programa al mismo tiempo. Existen dos esquemas que los programas de sistemas operativos utilizan para desarrollar Sistema Operativo multitarea, el primero requiere de la cooperación entre el Sistema Operativo y los programas de aplicación.

Multitarea basada en hilos

Con la multitarea basada en hilos, el hilo es la unidad de código más pequeña. Un programa puede realizar dos o más tareas de manera simultánea, como guardar un archivo mientras seguimos editándolo.

La multitarea basada en hilos genera una sobrecarga menor que la basada en procesos. Los procesos son tareas que requieren una mayor cantidad de recursos del CPU, además que la comunicación entre ellos suele ser limitada.

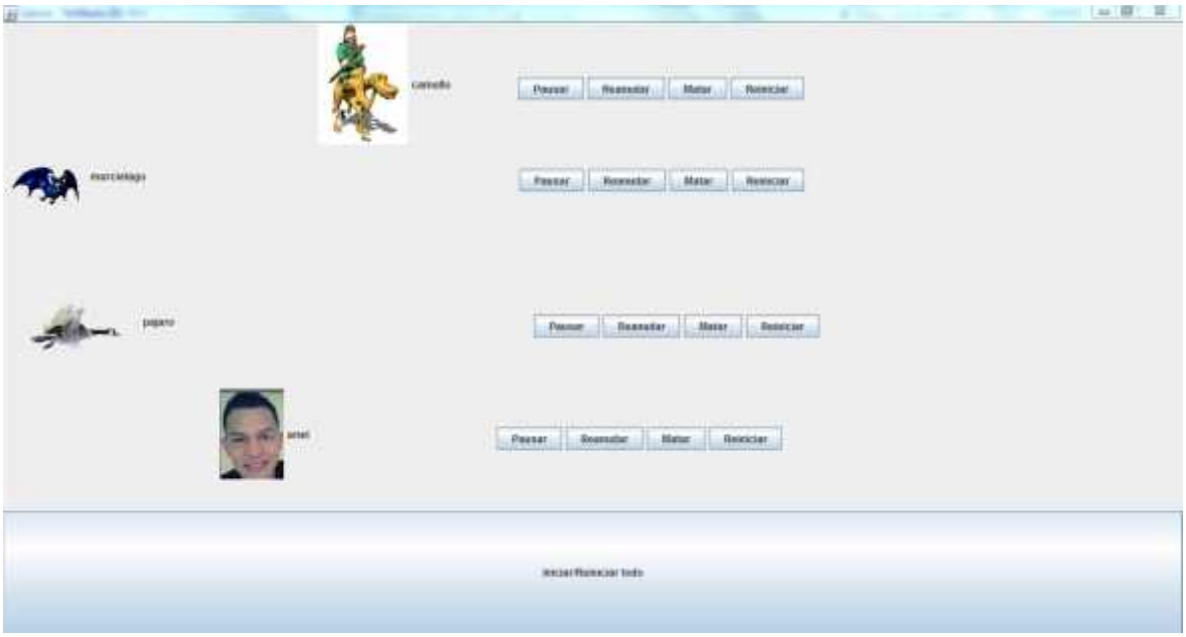
La multitarea basada en hilos permite escribir programas muy eficientes que hacen una utilización óptima del procesador, minimizando el tiempo libre que éste tiene.

Ejemplo desarrollado en Java

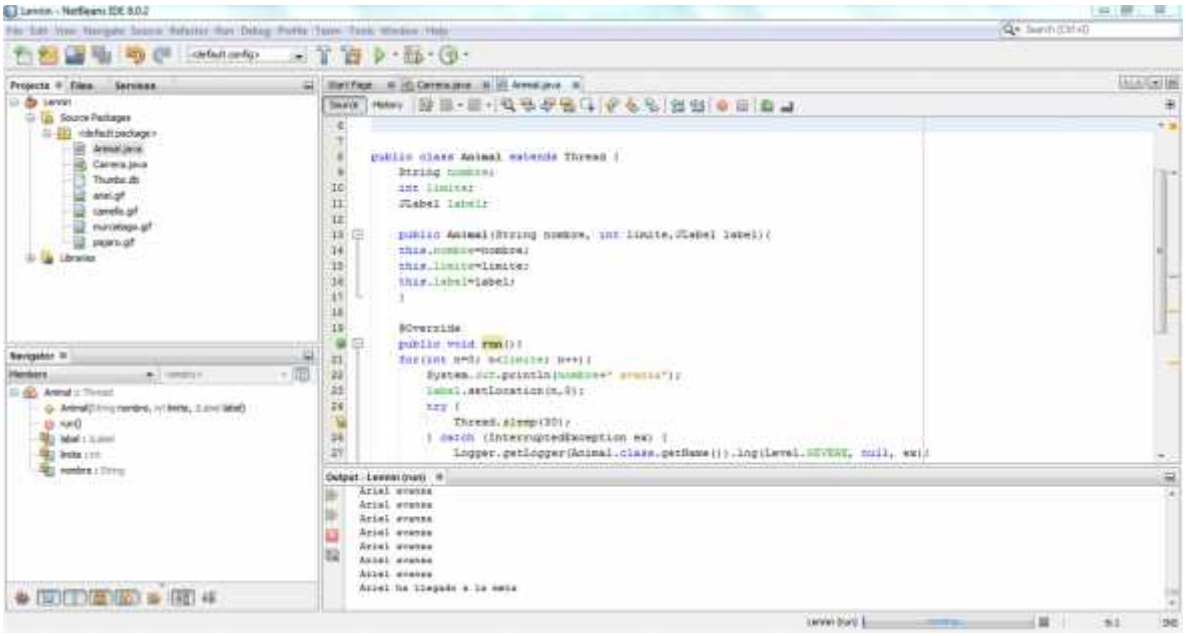
Aquí se desarrolló un pequeño programa en el lenguaje de programación Java, en el cual se muestra 4 procesos usando multitarea.

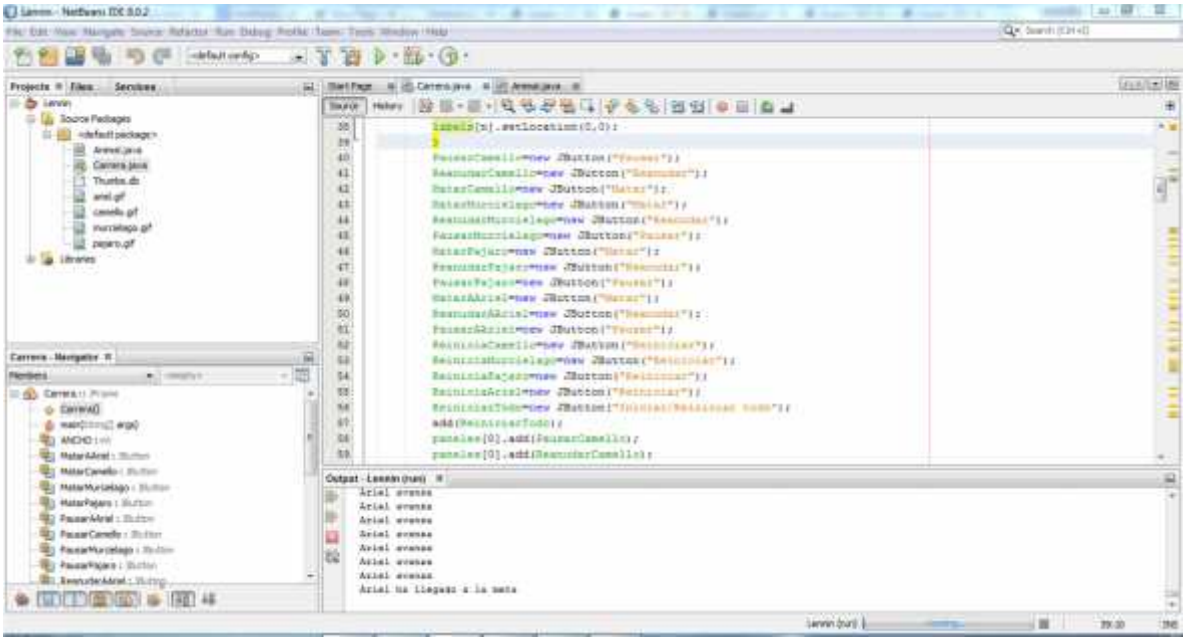
El ejemplo muestra un programa sobre una carrera de 4 animales, el cual cada animal es un proceso los cuales se ejecutan aleatoriamente por determinado tiempo.

En el programa se pueden iniciar la carrera, la cual inicia todos los procesos para ver cual es el que termina primero, pero también podemos detener a cada uno o a todos los procesos, podemos reanudar alguno o todos, podemos pausarlos y podemos matar el proceso para que ya no se esté ejecutando.



Todo esto se hace mediante hilos en la interface de Java la cual ya cuenta con la esta clase llamada Thread lo único que se hace es heredarla y así se podrán utilizar todos sus métodos, los cuales son detener, empezar, reanudar y muchos más, los cuales hicieron mucho más fácil el desarrollo del programa.



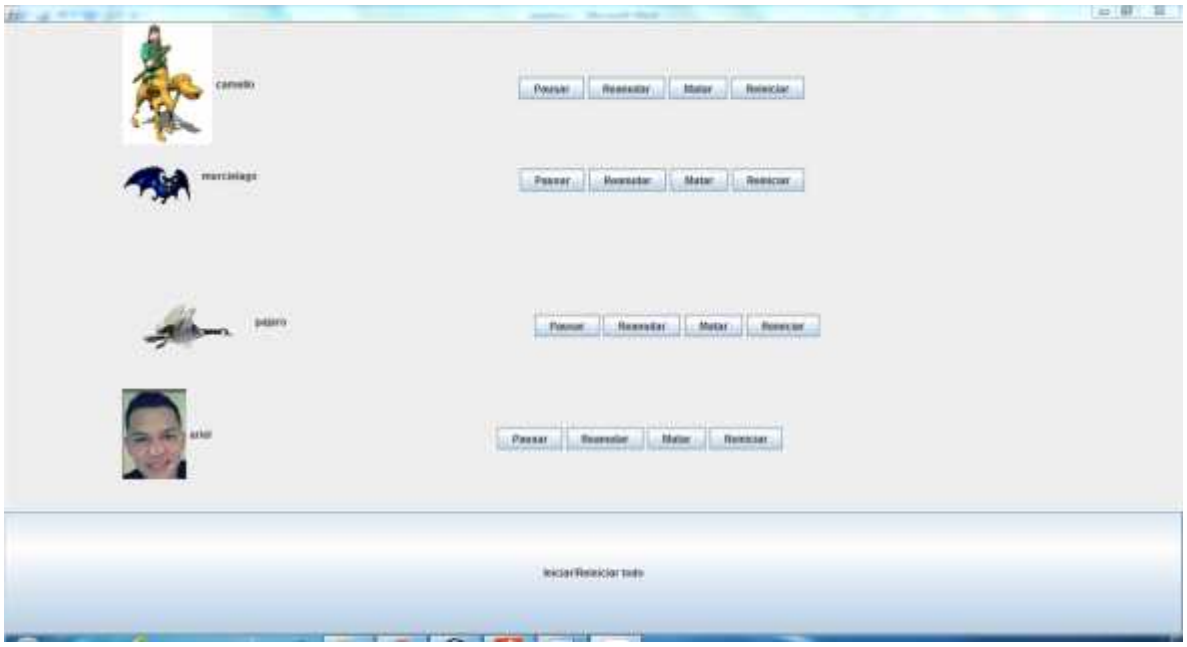


Como vemos al heredar la clase hilos el programa se hace mucho mas sencillo.

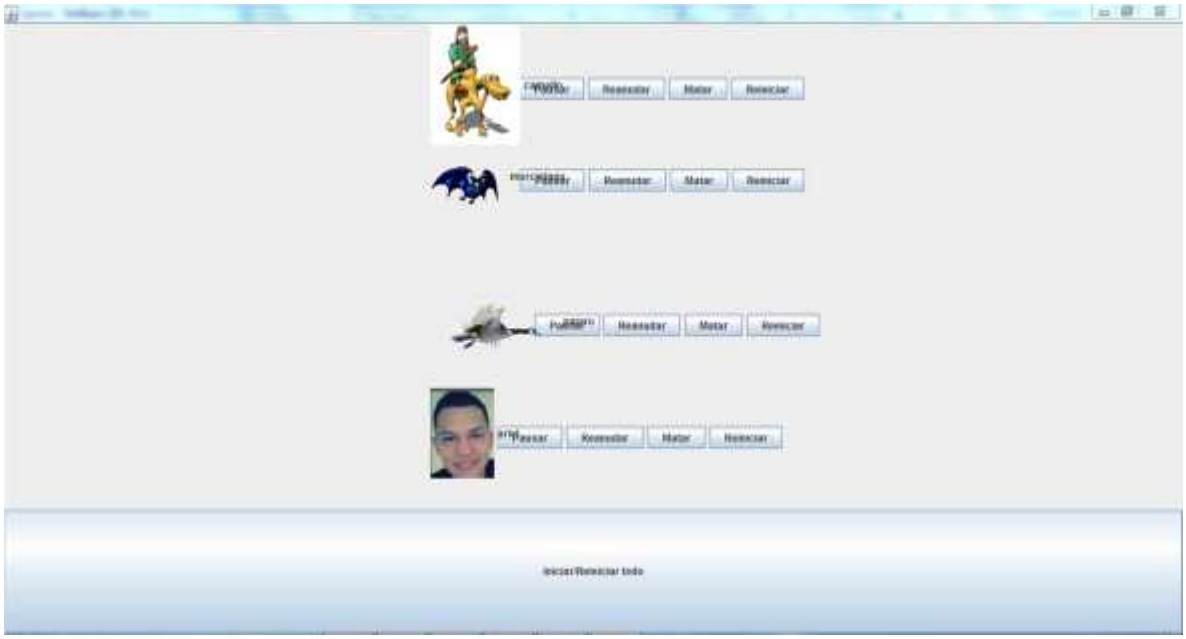
```
public class Animal extends Thread {
    String nombre;
    int limite;
    JLabel label;
```

```
    PausarCamello.addActionListener(new ActionListener() {

        @Override
        public void actionPerformed(ActionEvent e) {
            camello.suspend();
        }
    });

    ReanudarCamello.addActionListener(new ActionListener() {

        @Override
        public void actionPerformed(ActionEvent e) {
            camello.resume();
        }
    });
     MatarCamello.addActionListener(new ActionListener() {

        @Override
        public void actionPerformed(ActionEvent e) {
            camello.stop();
        }
```

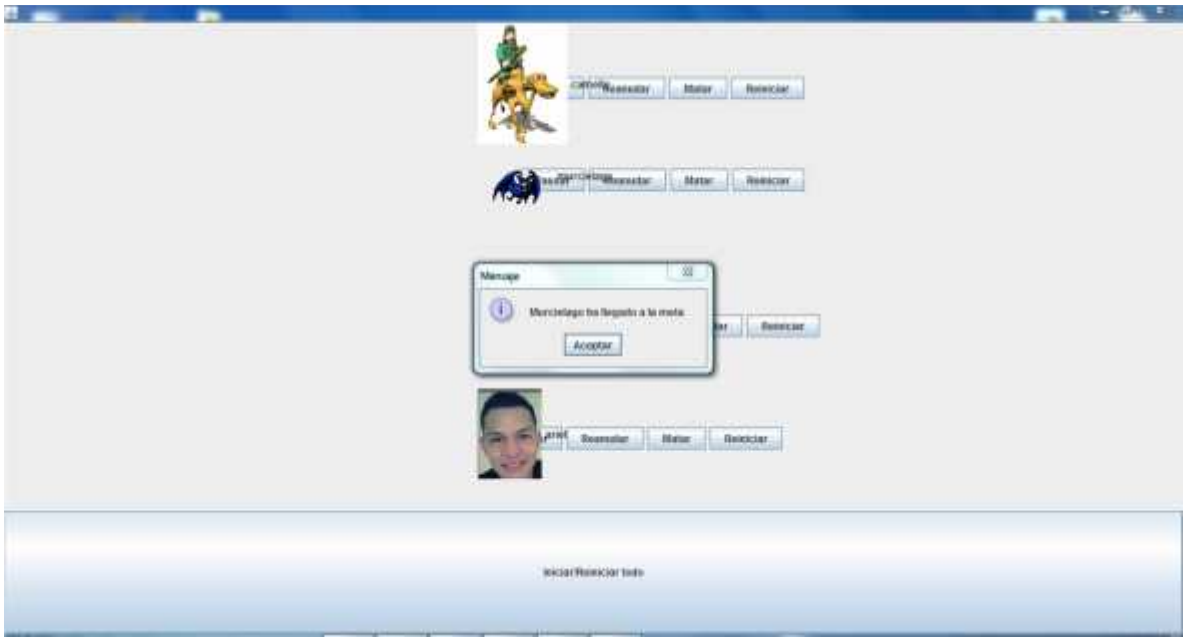
Bueno aquí se muestra como es que funciona el programa:



Al iniciar cada proceso avanza determinado tiempo aleatoriamente, al terminar espera unos milisegundos antes de que el siguiente avance, y así sucesivamente hasta que los 4 terminen, imprimiendo un mensaje de cuál fue el ganador.



Y como es aleatoriamente cada que se ejecute puede ganar cualquiera de los 4 animales:



Y pues cada uno de los botones que se muestran en la pantalla son para cada proceso el cual es independiente de los demás, es decir si matamos a solo ese proceso se detendrá los demás seguirán ejecutándose:

Conclusiones

Jaime Ariel:

La multitarea es una cuestión importante, ya que, con el paso del tiempo los equipos son más rápidos y cuentan con mas recursos, de la misma manera los programas día con día con más pesados y útiles pero requieren un mayor consumo de recursos, la ejecución de procesos multitarea nos ayuda a solventar dichos recursos de por parte de todas las aplicaciones que requiere el usuario, es decir podrás abrir los programas que necesites a la hora que los necesites y sin tener problemas.

Durante el programa que desarrollamos en Java, justamente ejemplificamos la forma en que varios procesos compiten por acceder a un recurso, a través del uso de hilos, lo cual nos ayudó a entender con mayor profundidad como es que se trabaja esta parte del procesamiento.

Juan Eduardo:

Los hilos son de vital importancia en un sistema operativo ya que sin ellos no se podría realizar tareas al mismo tiempo (al menos de manera visible) y se tendría que ejecutar cada proceso determinado tiempo y cerrarlo para poder abrir alguno otro. Gracias a los hilos se pueden realizar varias tareas a la vez, no al mismo tiempo, más bien simulando la multitarea concurrente.

Simulando la multitarea con hilos se tiene una mayor apreciación de cómo es que funciona debido a que se tiene el control de lo que se quiere realizar debido a que para ejecutar varios hilos, a cada uno se le asigna un tiempo para que esté en funcionamiento y con un tiempo mínimo de unos 5, 10, 15 segundos a más para su ejecución se puede apreciar mejor cómo es su funcionamiento, además de que se pueden pausar, reanudar, matar o reiniciar ya sean todos juntos o de uno en uno. Por lo tanto, aunque en un sistema operativo se aprecie multitarea real, puede ser que en realidad no sea así y se encuentre ejecutando hilos para cada tarea solicitada durante un tiempo o hasta que el usuario decida qué hacer con cada proceso en ejecución.

Rafael:

Un hilo en un sistema operativo es muy importante ya que puede permitir a una aplicación realizar varias tareas, esto es muy importante para el usuario por que por lo general está ejecutando más de un programa a la vez en su sistema operativo.

En el desarrollo del programa pude observar más claramente cómo es que funcionaban los hilos y porque eran muy importantes, ya que pusimos múltiples tareas asignándoles tiempos de ejecución a cada una, deteniéndolas y ejecutando inmediatamente

las siguientes, así se optimiza los programas más eficientemente, porque hacen una utilización más óptima del uso del procesador, haciendo menor el tiempo libre que él tiene.

General:

En conclusión, la practica como tal nos ayuda a darnos cuenta de cómo es que se ejecutan los procesos multitarea, es demasiado importante conocer cómo es que se desenvuelve ya que la ejecución multitarea está presente en casi todas las aplicaciones informáticas que utilizamos a diario.

En el programa que se desarrolló como ejemplificación de la ejecución de procesos multitarea utilizando los hilos o threads, pudimos darnos cuenta de cómo es que el procesador asigna tiempos definidos para cada proceso, de ese modo es que podemos ejecutar distintos programas al mismo tiempo.

La distribución del tiempo se realiza en base a distintos criterios, se puede utilizar el principio básico de colas de llegada FIFO, así como asignar prioridades dependiendo de qué tipo de proceso esté esperando a ejecutarse.

Bibliografía

Santamarina, R. (s.f.). *Java Threads .* Recuperado el 25 de Febrero de 2015, de http://vis.usal.es/rodrigo/documentos/aso/seminarios/javaThread.pdf

sistemas operativos y conceptos basicos. (2013). Recuperado el 2015, de http://correo.uan.edu.mx/~iavalos/OS.htm

.