Salas Garcia Mauricio

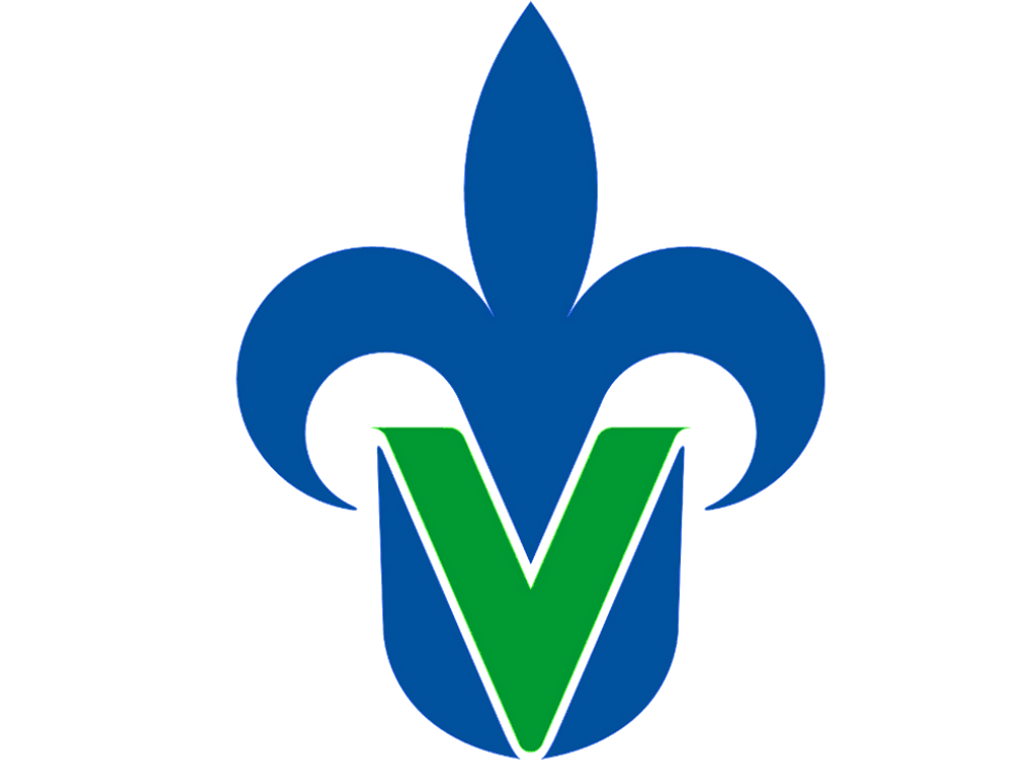
Fecha de entrega: 15/12/2023

EE: Sistemas operativos

Ingeniería de software  
Universidad Veracruzana  
Profesor: López Herrera Juan Luis

Documentación de proyecto

Simulador de planificador de procesos por prioridad



Contenido

[Documentación de proyecto: Simulador de planificador de procesos mediante el método de mayor prioridad 2](#_Toc153452181)

[Objetivo: 2](#_Toc153452182)

[Contexto: 2](#_Toc153452183)

[Procedimiento: 2](#_Toc153452184)

[Diagrama: 7](#_Toc153452185)

[Video: 7](#_Toc153452186)

[Conclusión: 7](#_Toc153452187)

# Documentación de proyecto: Simulador de planificador de procesos mediante el método de mayor prioridad

## Objetivo:

Lograr simular el proceso de asignar tiempo de la CPU a un proceso mediante el método de el de mayor prioridad, mostrándolo de manera gráfica y entendible.

## Contexto:

Se ha propuesto la creación que una aplicación que simule la llegada de múltiples procesos al mismo tiempo y se encargue de manejarlos de manera correcta dependiendo de un algoritmo de planificación establecido, para este caso, se solicitó utilizar el algoritmo de la mayor prioridad. Internamente la aplicación deberá de usar múltiples hilos para simular la llegada de procesos y a su vez asignarlos a la CPU. Todo esto debe de suceder mostrando los cambios a el usuario y como se maneja la planificación de procesos, mediante una interfaz gráfica que sea entendible y clara para que el usuario non se pierda en ella.

## Procedimiento:

Debido a que el proyecto se centra en tener una interfaz gráfica entendible, se ha priorizado esa parte, para que los usuarios tengan una buena experiencia de uso, iniciando con un dibujo simple que planeaba la forma de esta interfaz:

Diagrama, Esquemático

Descripción generada automáticamente

Donde se cuenta con 3 columnas, la de procesos mostrará cada proceso que se generará y se mostrará con sus datos importante, luego la segunda esta dividida en 2, la primera parte “En ejecución” muestra el proceso que pudo entrar a la CPU, mientras que la parte de abajo “Exclusiones”, muestra los procesos que no pudieron entrar a la CPU porque la memoria RAM es insuficiente para poderlos cargar. La tercera columna “Hechos”, deberá mostrar los procesos que lograron terminar la ejecución de la CPU y salieron exitosamente sin errores.

Finalmente se contará con un apartado que muestre la memoria RAM actual, el valor en milisegundos que le tomará a cada tiempo del proceso transcurrir (Velocidad del procesador) y por ultimo el tiempo en el que se generan los procesos, cada uno de estos parámetros deberá de ser modificable por el usuario durante la ejecución para que pueda jugar con los valores.

Contando con la interfaz gráfica que se realizará, se ha decidido separar el proyecto en 2 partes, tal como lo haría un sistema operativo real, se generaron 2 clases adicionales, una llamada “Gráficos” que actúa como la controladora de la interfaz gráfica, se encarga de mostrar todos los datos necesarios a el usuario y de recibir sus solicitudes para enviarlas a la segunda parte, la cual se llama “Sistema”, esta actúa como el kernel del sistema, cuenta con todas las operaciones necesarias para lograr completar la simulación de la planificación, recibe las solicitudes del usuario a través de lo que le envía la clase “Gráficos” y a su vez le envía lo que necesita a esa clase para que le pueda mostrar lo que el usuario desea conocer de la planificación de los procesos. Ambas clases están relacionadas fuertemente, dado que una necesita de otra y viceversa para poder funcionar y que la aplicación inicie normalmente.

Como parte del funcionamiento, se ha visto necesaria la utilización de hilos, ya que estos permitirán generar procesos y procesarlos al mismo tiempo, para ellos, se ha utilizado un hilo que acceda a la parte del sistema que genera hilos, esto lo hará de manera perpetua hasta que la aplicación finalice. A su vez, existe otro hilo que escogerá uno de los procesos con menor prioridad para ejecutarlo, después, se duerme el tiempo solicitado por los tiempos de ejecución del programa, finalmente, lo envía a la lista de los terminados, el hilo principal se encarga de inicializar estos 2 hilos y supervisar su ejecución para que nunca se interrumpan el uno a el otro.

Para que el segundo hilo pueda elegir el proceso con la prioridad más baja, se ha propuesto separar los procesos por prioridad, y dado que puede llegar más de uno, se ha propuesto que los identifique de manera de que elija a el de menor prioridad que primero llegó, para lograr esta meta, se generaron colas específicas que albergan solo el mismo tipo de prioridad, y dado que se tienen prioridades de -19 a 20, se generaron 40 colas con todas las prioridades, estas colas se manejan con una lista que las alberga, en donde el hilo que elige el proceso que se le asignara la CPU revisa, buscando la cola con la prioridad más baja que contiene algún proceso.

Para poder mantener el orden de los procesos como llegaron y quitar el proceso que está en ejecución, se creó una lista adicional que lleva esa cuenta de procesos en orden como entraron, por cada movimiento de procesos la interfaz gráfica se actualiza con los cambios que se le realizaron a la lista dada creada, por lo que cada vez que se quiera borrar, mover, o agregar un proceso, solo se deberá modificar la lista de procesos y mandar una llamada para actualizar la interfaz gráfica.

Para poder llegar a estas conclusiones de funcionamiento se realizó un pequeño sketch que lo explica:

Diagrama

Descripción generada automáticamente

Finalmente, cuando se completa la ejecución el proceso se saca de las colas y listas, y solo se muestra su información en la tabla de procesos terminados de la interfaz gráfica, logrando completar con la simulación completa y realista de como funciona una asignación de procesos por prioridad a el procesador.

Como métodos adicionales, tenemos un puente entre la interfaz gráfica y los atributos del sistema, tales como la memoria, el tiempo de procesamiento, o el tiempo de generación, al ingresar un nuevo dato y pulsar enter, el manejador de la interfaz gráfica le envía la información a el sistema, para que este se encargue de cambiar los datos de simulación y por lo tanto el comportamiento.

Para completar, el manejador de la interfaz gráfica tiene todos los métodos necesarios para poder borrar, actualizar, crecer, disminuir y ajustar cada una de las columnas según los procesos que contengan, además de crear la ventana principal y todo lo que lleva, colores, textos, etc.

Como era de esperar, existe una clase “Proceso” que contiene toda la información necesaria para manejarlo, simulando así de forma completa el procesamiento de estos.

Se muestra la aplicación terminada:

Captura de pantalla con letras y números

Descripción generada automáticamente con confianza media

La interfaz gráfica funciona exactamente igual a como se propuesto inicialmente, por lo que no se considera relevante explicar su funcionamiento.

Al iniciar la aplicación, se inicia la generación y procesamiento de procesos, los valores por defecto de los atributos del sistema son: 800 MB de memoria RAM, cada tiempo del procesador equivale a 15 milisegundos, y se generarán procesos cada 600 milisegundos. Estos valores fueron escogidos cuidadosamente para intentar lograr un equilibrio y que no se acumulen demasiados procesos de un solo lado. La memoria RAM se ha decidido que sea de 800 MB para que de paso a que algunos procesos que requieran más memoria sean excluidos y se muestre de manera clara lo que pasa.

Los valores de los procesos se han decidido de esta manera:

* El nombre del proceso es “Proceso” más el número de creación.
* La memoria que requieren es de 1 MB a 1000 MB.
* Su tiempo de procesamiento es de 1 a 100, esto para que no se quede demasiado tiempo en el procesador.
* Su prioridad puede generarse de -19 a 20, dado que se deseó seguir las prioridades más realistas de un sistema operativo real.

Todos los datos de los procesos son generados de manera aleatoria siguiendo los valores propuestos anteriormente.

## Diagrama:

Se incluye el diagrama de clases solicitado:

Diagrama

Descripción generada automáticamente

## Video:

Adicionalmente a todo esto, se ha incluido un video donde se explica más a detalle el funcionamiento de la aplicación y el código, se ha compartido mediante un link de un video para que sea más fácil de visualizar.

Link: https://youtu.be/KDHU76i\_kAQ

Link acortado: [Video explicativo proyecto simulador de planificador (youtube.com)](https://www.youtube.com/watch?v=KDHU76i_kAQ)

## Conclusión:

Se ha logrado completa el objetivo principal de la aplicación, con una interfaz gráfica comprensible y que cumple el algoritmo de planificación de prioridad e procesos, dando por concluido este proyecto y su debida documentación de su elaboración y las partes que contiene.

Autor: Mauricio Salas García.

Con fecha del 14/12/2023