TEC | Tecnológico de Costa Rica

Instituto Tecnológico de Costa Rica
Área Académica de Ingeniería en Computadores
CE 4303 — Principios de Sistemas Operativos

Proyecto 1
Simulación de procesos en tiempo real

Fecha de asignación:	6 de Octubre, 2020	Fecha de entrega:	29 de Octubre, 2020
Grupos:	Individual	Profesores:	Jason Leitón, Luis Barboza

1. Objetivo

Implementar un simulador de un calendarizador de tiempo real, con el fin de evaluar el rendimiento de los algoritmos de planificación para los RTOS.

2. Atributos a evaluar

- Aprendizaje continuo. Se requiere que el estudiante valore las estrategias y el conocimiento adquirido para alcanzar el objetivo.
- Herramientas de Ingeniería. Se requiere que el estudiante sea capaz de adaptar técnicas, recursos y herramientas modernas para la solución de problemas.

3. Motivación

En el mundo informático, cada vez más se necesitan aplicaciones con restricciones de tiempo, por lo que los RTOS se vuelven esenciales en este campo de la ingeniería. Uno de los campos de acción de un Ingeniero en Computadores es el desarrollo de hardware y software para aplicaciones críticas, por esto es importante que se familiarice con la calendarización de tareas cuyo objetivo es ejecutarse en un determinado tiempo.

4. Descripción

El proyecto consiste en crear un laberinto, el cual debe tener un punto de entrada y un punto de salida. La idea es que un marciano inicie en el punto de entrada y llegue al punto de salida. Cada marciano será modelado por un proceso, el cual debe compartir los recursos (el procesador) para avanzar dentro del laberinto.

Los marcianos seguirán rutas totalmente aleatorias para encontrar la salida. Cabe destacar que cada marciano cuenta con nivel de energía, el cual se debe de gastar antes de que se vuelva a generar la energía para dicho marciano, en caso contrario se generaría un error, ya que significa que no se puede calendarizar. Una vez que no tenga energía (tiempo transcurrido) se pausará y

otro marciano podrá avanzar. Cabe destacar que solo un marciano puede estar moviéndose en el laberinto. Cada marciano tendrá un tiempo de energía (tiempo de ejecución) y un tiempo que debe esperar para obtener de nuevo el máximo nivel de la misma (periodo).

La idea fundamental es que usted pueda calendarizar a los marcianos para que todos se puedan ejecutar de manera planificada con los algoritmos de RM y EDF (el usuario elige cual de los dos).

En caso de que no haya forma de calendarizar los marcianos, la simulación debe de ejecutarse hasta el momento en que ocurre el error, el cual debe indicárselo al usuario y terminar la simulación (de manera elegante).

4.1. Modo de operación

Existirá dos modos de operación, automático y manual.

- Manual: En este modo la simulación inicia y el usuario puede crear un marciano en cualquier instante, se le debe de pasar el tiempo de ejecución (tiempo que dura la energía) y el periodo (Tiempo en el que se regenera nuevamente) para cada marciano que se desea crear.
- Automático: La simulación inicia con una carga de trabajo, la cual debe ser ingresada por medio de interfaz gráfica por el usuario. De igual manera se le debe pasar el tiempo de ejecución y el periodo por cada marciano que se desea crear.

Cabe destacar que el usuario puede terminar la ejecución en cualquier momento presionando la tecla x. Una vez que termine se mostrará el reporte.

4.2. Interfaz gráfica

Toda la simulación debe ser con entorno gráfico, así como el ingreso de los datos y el reporte (se detalla más adelante).

4.3. Detalles de simulación

Con el fin de que el reporte se pueda visualizar de la mejor manera se tomará el ciclo del reloj como una unidad de tiempo lineal. Y una arquitectura con un único CPU.

4.4. Reporte

Cuando la simulación termine se deberá desplegar un reporte de como fueron calendarizados los procesos según los ciclos del procesador. El reporte debe ser similar al mostrado en la figura 1.

4.5. Laberinto

El laberinto será diseñado por el estudiante, sin embargo, debe de proporcionar las validaciones necesarias para que los marcianos se puedan mover con fluidez y no haya ningún tipo de choques. La figura 2 muestra un ejemplo de laberinto.

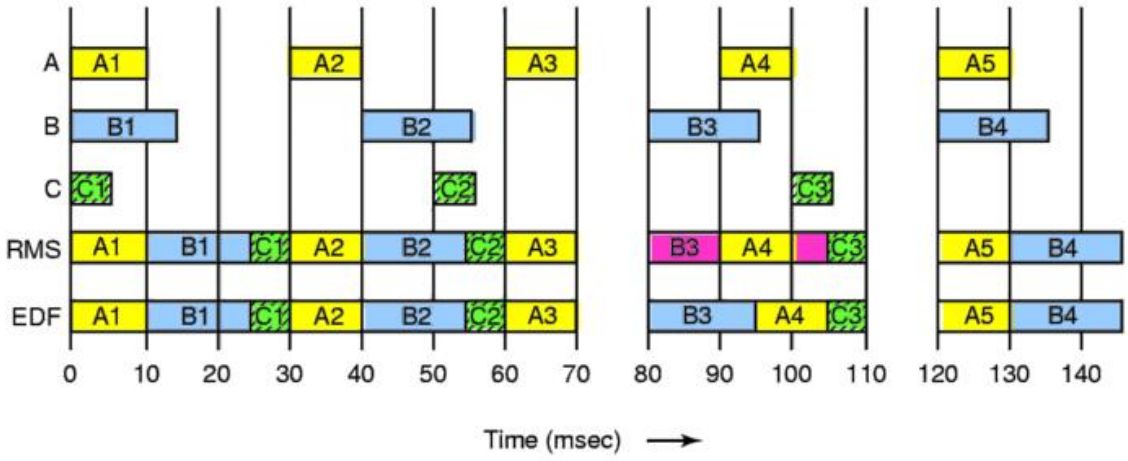

Figura 1: Ejemplo de reporte

Figura 2: Ejemplo de laberinto

4.6. Niveles de energía

En la parte superior derecha se deberá mostrar el nivel de energía que va consumiendo cada marciano que esté en el laberinto, ya que es el indicador de cuanto tiempo se ha ejecutado el proceso. Una sugerencia es hacerlo por medio de barras con divisiones discretas con tamaño variable.

4.7. Analogía con RTOS

Se desea modelar un RTOS duro, donde los marcianos son los procesos, el tiempo de la energía (cuando se mueve por el laboratorio) corresponde al tiempo de ejecución de un proceso, mientras que el tiempo de regeneración corresponde al periodo de cada proceso.

4.8. Requerimientos técnicos

- Este proyecto se debe realizar en el lenguaje de programación C. Pueden utilizar las bibliotecas que sean necesarias.
- Debe ser implementado en Linux y se debe proporcionar un makefile.
- No se permite soluciones .ªlambradas".
- Se debe prestar especial atención a los errores de acceso a memoria o utilización de recursos. Es inaceptable el error *segmentation fault*.

5. Documentación- Estilo IEEE-Trans (máximo 5 páginas)

- Introducción: Teoría necesaria, breve descripción del proyecto y qué es lo que se espera en el escrito.
- Ambiente de desarrollo: Todos los detalles de implementación y herramienta durante el desarrollo del proyecto.
- Atributos: Esta sección deben de describirse cuales atributos fueron reforzados durante el desarrollo del proyecto (documento aparte que explique cómo se reforzó).
- Detalles del diseño del programa desarrollo, tanto del software como del hardware (en caso de que aplique): Diagramas UML (obligatorio), diagrama de arquitectura (obligatorio), diagrama de funcionalidades (obligatorio), imágenes, descripciones entre otros, todo lo que sea necesario para entender de una mejor manera el diseño y funcionamiento del proyecto.

- Instrucciones de cómo se utiliza el proyecto.
- Tabla de actividades por cada estudiante: bitácora con el total de horas trabajadas.
- Conclusiones
- Sugerencias y recomendaciones.
- Referencias

6. Entregables

- Código fuente con documentación interna.
- Documentación.
- Archivos necesarios para ejecutar el programa.

7. Evaluación

- Algoritmo EDF 15 %
- Algoritmo RM 15 %
- Interfaz 15 %
- Reporte 15 %
- Integración 10 %
- Interacción con usuario 5 %
- Archivos de configuración y makefiles 5 %
- Documentación 20 %

8. Fecha de entrega

- 30 de octubre. 23:55 por tecdigital.

9. Otros aspectos administrativos

- Para la revisión del proyecto se debe de entregar tanto la documentación como la implementación del software.
- No se reciben trabajos después de la hora indicada.
- En la revisión del proyecto pueden estar presentes el coordinador y asistente.
- Es responsabilidad del estudiante proveer los medios para poder revisar la funcionalidad del software, por ejemplo, si no se realiza la interfaz, se debe de proporcionar otro medio para la verificación, de lo contrario la nota será cero en los rubros correspondientes a la funcionalidad faltante.