Proyecto 2 sistemas operativos: Scheduling ships

1st Juan Pablo Carrillo Salazar
Ingeniería en computadores
Instituto Tecnológico de Costa Rica
Cartago, Costa Rica
juanpcarrillo@estudiantec.cr

2nd Josue Cubero Montero
Ingeniería en computadores
Instituto Tecnoógico de Costa Rica)
San José, Costa Rica
jdcubero@estudiantec.cr

3rd Jose Pablo Fuentes
Ingeniería en computadores
Instituto Tecnoógico de Costa Rica)
Cartago, Costa Rica
jp.1398@estudiantec.cr

4th Dagoberto Rojas
Ingeniería en computadores
Instituto Tecnoógico de Costa Rica)
Cartago, Costa Rica
krojas96@estudiantec.cr

***Abstract*—La idea fundamental del presente proyecto es realizar una implementación de los hilos en espacio de usuario de tal manera que sea lo más eficiente posible de acuerdo con la funcionalidad solicitada. Donde la abstracción de todo el ambiente que se debe de tener para controlar cualquier evento se vuelve esencial para un correcto funcionamiento del programa. Para esto se realizó el diseño e implementación de un proyecto que simule el proceso de creación de procesos en un sistema operativo y que se encargue de organizarlos y manejarlos mediante el uso de algoritmos de calendarización. A partir de la realización del proyecto se profundizó en el área de paralelismos en los sistemas operativos, ayudando este a comprender y poner en práctica los conceptos básicos relacionados con este tema.**

***Index Terms*—Linux, Proceses, Schedulers, CPU, Threads**

I. Introducción

Para que un sistema operativo tenga un manejo eficiente de paralelismo, se debe de tener especial cuidado en la creacion y manipulacion de procesos. En este proyecto se busca implementar los hilos en espacio de usuario definiendo la prioridad en su ejecucion en base a una serie de algoritmos de tal manera que sea lo mas eficiente posible. Para lograr un buen funcionamiento del programa, debemos de controlar cualquier evento considerando cual seria el modo mas optimo para hacerlo. Para hacer esto posible debemos de hacer uso de hilos y definir su optima ejecucion en base a un calendarizador, quien realiza este proceso en base a una serie de algoritmos tale comos **RR**, **Prioridad**, **SJF**, **FCFS** y **Tiempo real**. Estos algoritmos son quienes definen -dentro del calendarizador- el orden de la cola de listos y el orden en el que como estos iran siendo ejecutados.
En el caso del manejo de hilos, se tuvo que reimplementar la biblioteca Pthreads llamandola CEthreads y haciendo uso de directivas de C. Dicha reimplementacion incluye las funciones:

- CEthread_create
- CEthread_end
- CEthread_join
- CEmutex_init
- CEmutex_destroy
- CEmutex_unlock

En el contexto del problema a resolver dentro de este proyecto, se utilizaron las siguientes abstracciones:

- Cada hilo seria considerado como un **barco**.
- El procesador seria considerado como un **canal** sobre el cual viajan los barcos.

Si consideramos la idea que hay detras de la ejecucion de un proceso, sabemos que este siempre tendra un tiempo de ejecucion en base a lo que sea que este haga. Para lograr que los distintos hilos tuvieran tiempos de ejecucion varibles dentro de la abstaccion utilizada, cada barco debia de tener una seria de parametros que sirviesen para definir dicho tiempo de ejecucion. En este caso el tiempo de ejecucion de cada hilo/barco seria definido en base a su velocidad, la cual estaria definida segun el tipo de barco:

- Normales: Su velocidad es baja, lo cual implica alto tiempo de ejecucion.
- Pesqueros: Su velocidad es media, lo cual implica un tiempo de ejecucion entre el tiempo de ejecucion de los barcos normales y los patrulla.
- Patrulla: Su velocidad es alta, lo cual implica que su tiempo de ejecucion seria el mas rapido respecto al de los demas barcos.

Para identificar hasta que punto cada barco/hilo se logra ejecutar o avanzar a travez dek canal antes de que este sea detenido por el calendarizador, se dividio el procesador/canal en etapas. Cada una de estas etapas funcionarian para conocer cuan avanzado esta la ejecucion de cada hilo/barco. Finalmente, para lograr que estas dos abtracciones, tanto para los hilos como para el procesador, pudieran ser mostradas ante un usuario, se hizo uso de dos elementos:

- Hardware: Este debe de servir para darle al usuario una nocion de que es lo que esta pasando con la ejecucion de cada hilo/barco.
- Interfaz grafica: Este recurso serviria para lo mismo que el anterior pero haciendo uso unicamente de software.

debiamos de ”dividir” en etapas a el procesador para que asi, pudiesemos llevar un control conocido de cuanto tiempo iba

poder ejecutarse cada hilo en base a su duracion y cuanto de su trabajo iba a poder lograrse en dicho dicho tiempo.

II. Ambiente de Desarrollo

Para el desarrollo de cada módulo, algoritmo o proceso se uso el lenguaje de programación C,los cuales eran compilados y ejecutados en la última versión estable de Ubuntu, la 21.10 (Impish Indri). El compilador utilizado fue GCC, una serie de compiladores de software libre distribuido por Free Software Foundation.

Para la ejecución de este proyecto se requiere descargar e instalar una biblioteca gráfica, necesaria para visualizar el comportamiento de la aplicación. El proyecto usa "raylib" y puede ser descarga e instalada siguiendo los pasos desde su repositorio oficial https://github.com/raysan5/raylib.

Los requisitos mínimos para la ejecución del programa son:

- Procesador de doble núcleo de 2 GHz o superior
- 2 GB de RAM
- 25 GB de espacio libre en el disco duro (Para poder instalar Ubuntu sin problemas)
- Tarjeta gráfica VGA

III. Atributos

Esta sección describe los atributos del Instituto Tecnológico de Costa Rica y cómo estos se emplearon durante el desarrollo del proyecto. [1]

A. Aprendizaje continuo

De acuerdo con la definición del Instituto Tecnológico de Costa Rica, el aprendizaje continuo es: [1]

> *Capacidad para reconocer las necesidades propias de aprendizaje y la habilidad de vincularse en un proceso de aprendizaje independiente durante toda la vida, en un contexto de amplio cambio tecnológico.*

Durante este proyecto se trabajó en el aprendizaje continuo de las diferentes maneras:

- Comprensión de problemas complejos sobre analogías de calendarización de procesos en el espacio de usuario.
- Desarrollo de algoritmos de calendarización en bajo nivel, basados en descripciones de alto nivel.
- Aprendizaje sobre el manejo y creación de bibliotecas estáticas en lenguaje de programación C.

Se encontraron, además, varios retos importantes en esta categoría, tales como la búsqueda de una biblioteca mínima para el manejo de hilos en C, de manera que se pudiera implementar solo los métodos necesarios para obtener los resultados deseados.

B. Herramientas de Ingeniería

Nuevamente, la definición del Instituto Tecnológico de Costa Rica en este atributo se define como:

> *Capacidad para crear, seleccionar, aplicar, y adaptar apropiadamente técnicas, recursos y herramientas modernas de ingeniería y de tecnologías de información, incluyendo predicción y modelado de problemas de complejos de ingeniería, con la comprensión de las limitaciones asociadas.*

Durante este proyecto, se implementaron las siguientes herramientas de ingeniería:

- Diagramas UML (secuencia, arquitectura)
- Desarrollo ágil, priorizando entregables mínimos funcionales de inicio a fin.
- Diseño de esquemáticos electrónicos, para una propuesta de hardware
- Desarrollo itereativo, añadiendo elementos en cada entregable
- Desarrollo usando branching en git
- Uso de git como control de versiones.

En este atributo, se aprendió y hubo familiaridad con nuevas estrategias de desarrollo presentadas por los diferentes compañeros, llevando a una mejor comprensión y manejo de las mismas, por ejemplo git.

IV. Diseño

La parte central de este proyecto se basa en la ejecucion de una serie de hilos. Esta ejecucion es controlada por el calendarizador el cual hace uso de una serie de algoritmos para definir como dicha ejecucion se llevará acabo. El diagrama de Arquitectura del programa se muestra en la Figura 1

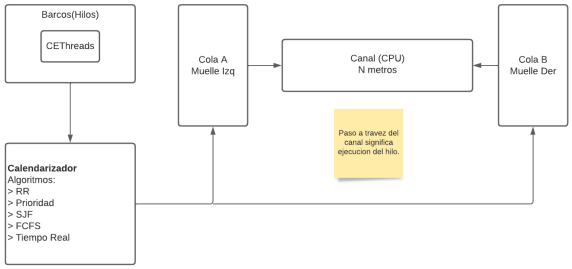

Fig. 1. Diagrama de arquitectura de la aplicación

Notese del Diagrama de Arquitectura de la Figura 1, como el calendarizador interacciona tanto con los hilos como con las listas desde las cuales se ejecutaran dichos procesos.
Tal y como se a mencionado en la introduccion, para este proyecto, los hilos fueron abstraidos como un barco. Ademas, el procesador fue abstraido como un canal dividido en una cantidad de etapas en funcion del tamaño del canal.
De la Figura 2 puede observarse que para un canal de tamaño n, este tendrá n etapas.

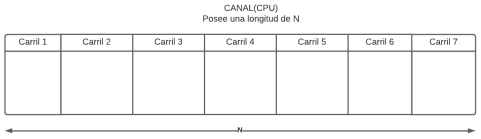

Fig. 2. Abstracion del Procesador.

Para el caso de los barcos, de sus parametros mas relevantes se tiene la velocidad, posicion, su identificador y la direccion en la que este se está desplazando a travez del canal. En la Figura 3 se puede observar como estos estarian siendo ejecutados en el procesador y cuales serian los valores de sus parametros.

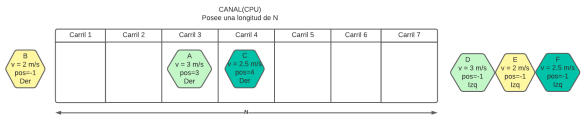

Fig. 3. Barcos en ejecución.

Nótese como el valor de la posición de cada barco es consistente a la posición dentro del canal en la que este se encuentra. Además, es importante resaltar que la velocidad de cada barco indica que tan rápido se moverá este a través del canal mientras dicho barco está siendo ejecutado. También, puede observarse que cada barco tiene un parámetro que identifica el sentido en el que este cruza el canal.

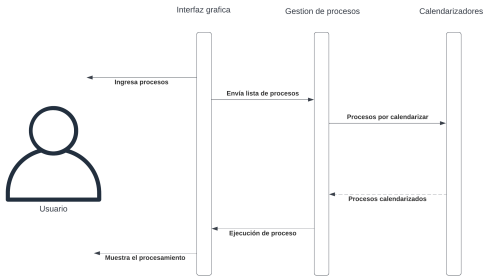

Fig. 4. Diagrama de secuencia de la aplicación.

En la figura 4 se muestra el comportamiento principal de la aplicación mediante un diagrama de secuencia. En esta figura se muestra como el usuario interactúa primeramente con el componente de interfaz gráfica para agregar procesos (barcos) a cada uno de los oceanos. Posteriormente, la lista de procesos es controlada por el componente de gestión de procesos, quien haciendo uso del componente calendarizador controla y administra el tiempo de ejecución de cada uno de los procesos.

Finalmente el usuario podrá visualizar como cada uno de los procesos se ejecuta, haciendo uso de la interfaz gráfica y apoyándose en la información mostrada por medio del hardware desarrollado.

Por último, en la figura 5 se muestra el diagrama de alto nivel del hardware utilizado. Se usan 3 leds para indicar el el sentido del canal, mientras que los otros dos leds restantes se iluminan cuando hay barcos de cada uno de los océanos presentes en el canal. Estos leds utilizan una resistencia de 330 ohmios como protección y cada uno de ellos está conectado a un pin GPIO del microcontrolador ATMEGA 328 P.

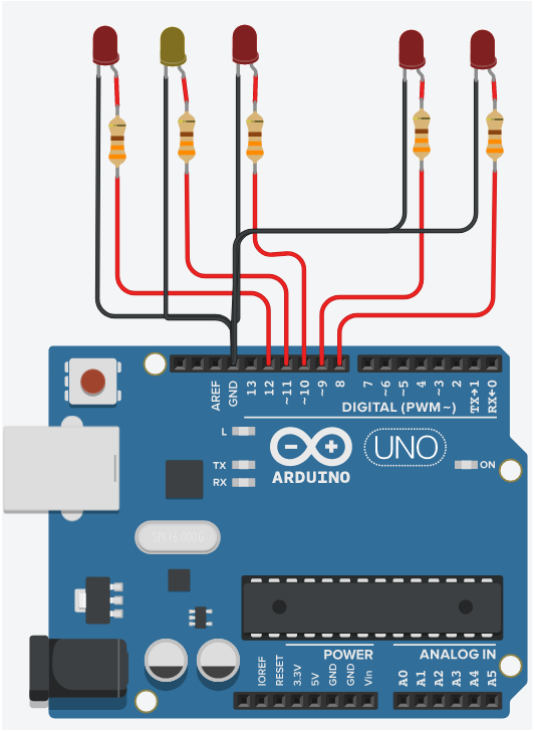

Fig. 5. Diagrama de alto nivel del hardware utilizado.

V. Instrucciones de uso

1) Ejecutar el archivo makefile haciendo uso del comando make.
2) Conectar la placa de desarrollo Arduino mediante un cable USB.
3) Indicar cada uno de los parámetros en el archivo de configuración inicial (Cantidad de barcos, velocidad, tamaño del canal, etc).
4) Ejecutar el archivo creado en el paso 1.
5) Ingresar cada uno de los parámetros solicitados antes de comenzar la ejecución.
6) La ejecución comenzará automáticamente en cuanto todos los parámetros estén debidamente configurados.
7) Una vez inicie la ejecución, podrá pulsar la tecla 1 para agregar barcos en el océano 1, la tecla 2 para el segundo océano.
8) Para finalizar la ejecución pulse la tecla ESC.

VI. Conclusiones

1)

VII. Sugerencias y recomendaciones

1) Se recomienda usar la biblioteca para interfaces gráficas *raylib*, ya que está especialmente optimizada para la creación de videojuegos, por lo que la programación de objetos moviéndose en la pantalla se facilita.
2) Usar el puerto serial de un microcontrolador facilita la comunicación entre un programa ejecutandose en una maquina y un microcontrolador conectado mediante un cable USB.

REFERENCES

[1] Instituto Tecnológico de Costa Rica 2022. Atributos TEC. [online]. Disponible en:https://www.tec.ac.cr/atributos-tec [Accesado 31 Octubre 2022].

VIII. TABLA DE ACTIVIDADES

TABLE I
TABLA GENERAL DE ACTIVIDADES POR ESTUDIANTE

Actividad	**Responsable**	**Horas**
Diseño de la solución	Todos	3
Desarrollo de CEThreads	Dagoberto	3
Implementación algoritmo de prioridad	Jose Pablo	3
Implementación algoritmo fcfs	Josue	3
Implementación algoritmo sjf	Dagoberto	3
Implementación algoritmo RR	Juan Pablo	3
Creación de Barcos	Josue y Jose Pablo	4
Manejo de archivos	Josue	2
Algoritmos para el tránsito	todos	3
Conexión C-Arduino	todos	4
Documentación	todos	4
Prueba de funcionalidades	todos	3
Total de tiempo invertido		38

IX. ANEXOS