Scheduling Ships

Dylan Gerardo Garbanzo Fallas
Tecnológico de Costa Rica
Escuela de Ingeniería en Computadores
dgarbanzof@estudiantec.cr

Alejandra Rodríguez Castro
Tecnológico de Costa Rica
Escuela de Ingeniería en Computadores
2021131070@estudiantec.cr

Isabel Andrea Cordoba Quesada
Tecnológico de Costa Rica
Escuela de Ingeniería en Computadores
isacordobaq@estudiantec.cr

Diego Vega Mora
Tecnológico de Costa Rica
Escuela de Ingeniería en Computadores
diegovm02@estudiantec.cr

Cristhofer Azofeifa Ureña
Tecnológico de Costa Rica
Escuela de Ingeniería en Computadores
azofeifacris4@estudiantec.cr

***Abstract*—This project aims to implement a thread library in the C programming language to execute multiple tasks simultaneously and schedule them using various scheduling algorithms. The developed system simulates the crossing of ships through a canal that allows one-way traffic at a time. The scheduling algorithms implemented include Round Robin, Priority, Shortest Job First (SJF), First Come First Serve (FCFS), and Real-Time scheduling. The project integrates a graphical user interface (GUI) built with GTK to display the ship animations and an Arduino-based hardware prototype that reflects the events occurring in the canal. The results demonstrate the efficiency of the created thread library and its correct integration with the scheduling algorithms, ensuring no collisions and maintaining ship traffic flow within the canal.**

***Index Terms*—Threads, scheduling, canal, scheduling algorithms, C, GTK, Arduino.**

I. Introducción

En sistemas operativos, la planificación de procesos es esencial para la administración eficiente de recursos, especialmente en entornos donde múltiples tareas deben ejecutarse en paralelo. Este proyecto se enfoca en la creación de una biblioteca de hilos llamada CEthreads, con el objetivo de calendarizar barcos que deben cruzar un canal mediante diversos algoritmos de planificación.

Los algoritmos implementados incluyen Round Robin (RR), Prioridad, Shortest Job First (SJF), First Come First Serve (FCFS) y Tiempo Real. Cada algoritmo gestiona una cola de barcos y garantiza que no haya colisiones durante el cruce del canal, que opera en un solo sentido a la vez. Además, se desarrolló una interfaz gráfica en GTK para visualizar la simulación y un sistema de hardware basado en Arduino para representar el estado de los barcos en el canal.

Este trabajo abarca tanto el diseño de software como de hardware, ofreciendo una solución completa para la simulación y planificación de múltiples procesos.

II. Ambiente de desarrollo

El proyecto fue desarrollado completamente en el lenguaje de programación C, utilizando la librería estándar del sistema y una implementación propia de hilos. Para la interfaz gráfica, se utilizó la librería GTK, la cual fue la única instalación adicional requerida.

El hardware utilizado consistió en una placa Arduino, la cual se usó para simular el comportamiento del canal y de los barcos, permitiendo la visualización física de los eventos, como el cambio de sentido del canal y el movimiento de los barcos. El entorno de desarrollo fue Linux, y se proporcionó un *Makefile* para la compilación y ejecución del proyecto.

III. Atributos

Durante el desarrollo de este proyecto, se reforzaron varios atributos clave, los cuales son descritos a continuación:

A. Trabajo individual y en equipo

- Estrategias inclusivas: Se planificó de forma equitativa e inclusiva, dividiendo el trabajo de manera clara para aprovechar las fortalezas de cada miembro del equipo.
- Planificación clara: Cada etapa del proyecto fue asignada a un responsable, con metas y roles definidos, asegurando que todos los aspectos fueran cubiertos.
- Colaboración: Se promovió una comunicación constante entre los miembros del equipo mediante reuniones semanales, garantizando la colaboración efectiva en cada etapa del desarrollo.
- Ejecución de estrategias: Las estrategias de colaboración y planificación se llevaron a cabo de manera efectiva, lo que permitió cumplir los objetivos dentro de los plazos establecidos.
- Evaluación del desempeño: Cada miembro fue evaluado por su capacidad de cumplir con sus responsabilidades individuales y por su contribución al éxito del equipo.
- Equidad e inclusión: Durante todo el proceso, se fomentó la participación equitativa de todos los miembros, asegurando que cada uno tuviera la oportunidad de contribuir y aprender.
- Colaboración en equipo: Las acciones de colaboración se vieron reflejadas en la integración de los distintos módulos (CEthreads, GUI, y hardware), permitiendo que el sistema funcione de manera fluida.

IV. Diseño

A. Visión General del Sistema

El sistema implementado simula la calendarización de barcos a través de un canal utilizando la biblioteca de hilos `CEthreads`. Los componentes principales incluyen la gestión de hilos, calendarización de barcos, gestión del canal y la interfaz gráfica. La arquitectura se organiza en los siguientes módulos:

- **Módulo de Gestión de Hilos**: Proporciona las funciones para crear, ejecutar y sincronizar los hilos que representan los barcos.
- **Módulo de Calendarización**: Implementa los algoritmos de calendarización como Round Robin, Prioridad, SJF, FCFS y Tiempo Real para ordenar los barcos que cruzan el canal.
- **Módulo de Gestión del Canal**: Controla el flujo de barcos a través del canal, asegurando que no se produzcan colisiones y manejando la dirección de tránsito.
- **Módulo de Entrada de Configuración**: Lee los parámetros de configuración como el ancho del canal, el tipo de control de flujo y los barcos definidos por el usuario.
- **Módulo de Interfaz Gráfica**: Proporciona una representación visual de los barcos, el canal y el estado de la simulación, incluyendo indicadores de dirección y el progreso de los barcos.

B. Descripción de los Módulos

1) Módulo de Gestión de Hilos: El módulo de hilos gestiona la creación y finalización de hilos que representan a cada barco. Se utilizan funciones como `cethread_create` y `cethread_join` para controlar el ciclo de vida de los hilos. Cada barco es un hilo que cruza el canal según el algoritmo seleccionado.

2) Módulo de Calendarización: Este módulo se encarga de la calendarización de los barcos en la cola de ”listos”. Los algoritmos implementados son:

- **Round Robin (RR)**: Los barcos se calendarizan de manera circular.
- **Prioridad**: Los barcos se calendarizan según su prioridad asignada.
- **SJF (Shortest Job First)**: El barco con el menor tiempo estimado para cruzar el canal tiene prioridad.
- **FCFS (First Come, First Serve)**: Los barcos se calendarizan según el orden en que llegan.
- **Tiempo Real**: El barco con una restricción de tiempo más estricta cruza primero.

3) Módulo de Gestión del Canal: Este módulo maneja el flujo de los barcos a través del canal y asegura que no ocurran colisiones. El usuario puede elegir entre diferentes métodos de control de flujo, como *Equidad*, *Letrero* o *Tico*, para regular el sentido del tráfico en el canal. También controla la longitud y la velocidad de los barcos en el canal.

4) Módulo de Entrada de Configuración: Se procesan los parámetros proporcionados por el usuario, como el ancho del canal, el método de control de flujo y la cantidad de barcos. Estos parámetros son esenciales para la simulación y se cargan desde un archivo de configuración.

5) Módulo de Interfaz Gráfica: Este módulo se encarga de visualizar el estado de la simulación, incluyendo la representación gráfica del canal, los barcos en espera y los que cruzan, además de los indicadores de flujo de tráfico. La interacción con el usuario se gestiona a través de teclas para generar nuevos barcos o cambiar configuraciones en tiempo real.

C. Diagrama de Arquitectura del Sistema

El siguiente diagrama de arquitectura ilustra la interacción entre los módulos principales del sistema: Gestión de Hilos, Calendarización, Gestión del Canal, Entrada de Configuración e Interfaz Gráfica. Muestra cómo los barcos (hilos) son gestionados y calendarizados para cruzar el canal, y cómo los diferentes módulos interactúan entre sí.

D. Diagrama de Secuencia

A continuación, se presenta un diagrama de secuencia que describe el ciclo de vida de un hilo (barco) en el sistema. El diagrama comienza con la creación del hilo, sigue con su calendarización según el algoritmo seleccionado, y muestra cómo el hilo intenta cruzar el canal, interactúa con mutexes para evitar colisiones y termina su ejecución.

V. Instrucciones

Para poder ejecutar el programa de Scheduling Ships, de primero uno debe de compilar el programa de arduino y conectar el arduino, Después se debe de ejecutar el programa y revisar en cual puerto esta conectado el arduino; en este caso estaba conectado en ”ACM0” al utilizar un ambiente en linux; y revisar que en la lectura de ”calendar.c” concuerden estos puertos. Después de subir el programa al arduino uno debe de asegurarse que el monitor serial del arduino este cerrado para poder compilar y ejecutar ”calendar.c”. AL ejecutar ”calendar.c” de primero se le va a pedir que ingrese ene la terminal el valor del ancho del canal, después se debe de elegir el tipo de control de flujo; donde si se elige el control de equidad se debe de indicar un parámetro W; si no fuera equidad no se debe de indicar el valor. Después de elegir el control de flujo, se indica cual tipo de calendarización se desea usar, y al final se indica el valor del tiempo de cambio del letrero, el cual se usa principalmente para el control de flujo de letrero. En la consola se va a ver la dirección del letrero y el barco que este avanzando y su estado. En el arduino de primero se va a notar en el led de 7 segmentos el numero de barcos que hay con un máximo de 9 para el arduino, no para el codigo; después en el led RGB se va a ver algún cambio de color, azul si es un barco pesquero, verde si es normal y rojo si es patrullero. Después hay dos leds rojo y verde de ambos lados si el led esta en verde de un lado significa que ese lado puede pasar y en el otro lado se vera el led rojo encendido

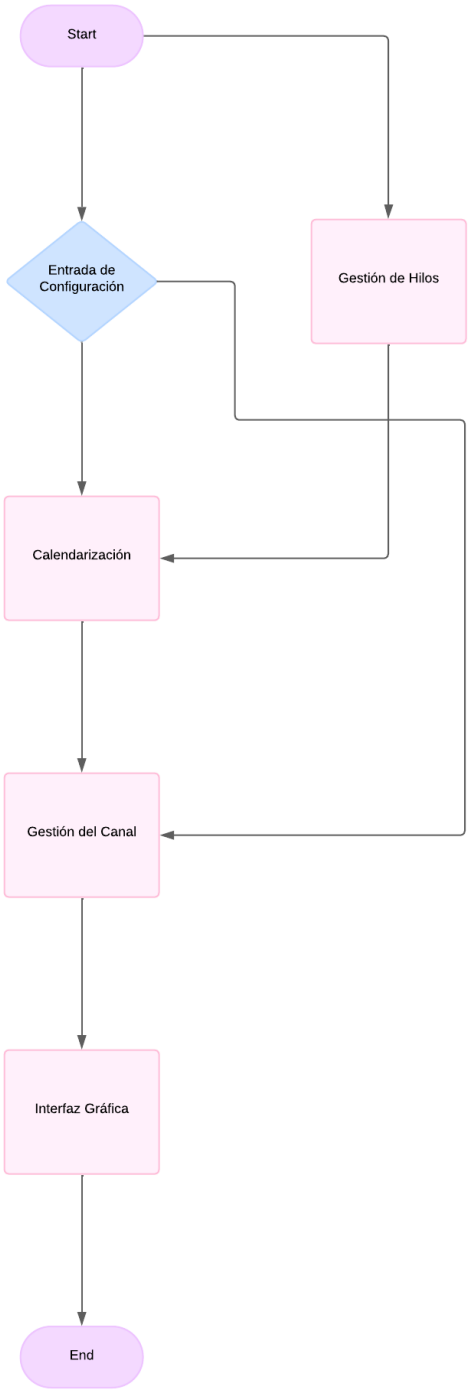

Fig. 1. Diagrama de Arquitectura del Sistema

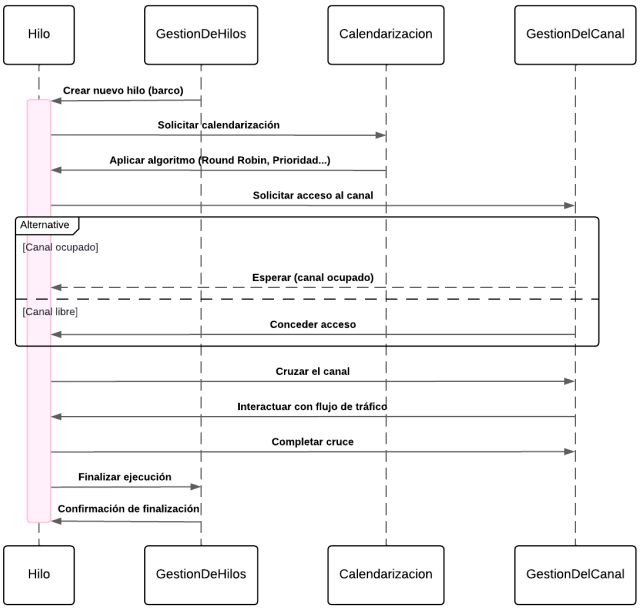

Fig. 2. Diagrama de Secuencia del Ciclo de Vida de un Hilo

y viceversa; después durante el movimiento del barco se van a encender los 3 leds amarillos de manera intermitente; si el barco llega a completar su camino el led blanco del centro se va a encender. Esto se debe al caso de los calendarizadores no apropiativos, como el Round Robin y el de Tiempo Real, donde un barco se puede pausar en su camino, de tal manera al tener el eld blanco uno va a poder notar si ya termino su ejecución.

VI. Conclusiones

El desarrollo de una biblioteca de hilos personalizada en el lenguaje C fue facilitado por la simplicidad y el control que ofrece este lenguaje de bajo nivel. La implementación de la biblioteca CEthreads permitió calendarizar múltiples barcos que cruzaban un canal, utilizando algoritmos de planificación bien conocidos como Round Robin, Prioridad, SJF, FCFS, y Tiempo Real.

Aunque C es un lenguaje complejo, la amplia documentación existente para la biblioteca de hilos Pthreads ayudó significativamente en el diseño de nuestra propia solución. La naturaleza eficiente y detallada de Pthreads sirvió como una excelente referencia para este proyecto, permitiendo un desarrollo robusto y flexible de nuestra biblioteca de hilos.

El desarrollo de los algoritmos de calendarización para el proyecto ha permitido una comprensión más profunda de las técnicas utilizadas en sistemas operativos para gestionar procesos, aquí representados como barcos que atraviesan un canal. Cada algoritmo implementado —Round Robin, Prioridad, SJF (Shortest Job First), FCFS (First Come First Serve) y Tiempo Real— aborda el problema de la calendarización desde una perspectiva diferente, adaptándose a las necesidades específicas de la simulación.

Cada uno de los algoritmos presenta ventajas y desafíos únicos, y su implementación nos ha permitido evaluar su comportamiento en situaciones reales simuladas. La flexibilidad del sistema para adaptarse a distintos escenarios mediante la selección del algoritmo más adecuado resalta la importancia de un diseño modular y robusto. En general, la integración exitosa de estos algoritmos en el sistema no solo mejora la eficiencia de la simulación, sino que también demuestra cómo los conceptos fundamentales de la calendarización en sistemas operativos pueden ser aplicados en una variedad de contextos.

VII. Recomendaciones

- Se recomienda a futuros desarrolladores estudiar detalladamente la documentación de Pthreads para entender la implementación de hilos y sincronización en C, antes de desarrollar una biblioteca propia.
- Se recomienda modularizar cada algoritmo de forma independiente y clara, asegurando que se puedan añadir o modificar fácilmente sin afectar otras partes del sistema. Esto no solo facilitará la comprensión del código, sino que permitirá la extensión del sistema con nuevos algoritmos de calendarización o mejoras a los existentes. La implementación de interfaces o plantillas para los algoritmos garantizaría que todos sigan un formato estándar, lo que reducirá errores y aumentará la mantenibilidad del código.

VIII. Sugerencias

- Para mejorar la interfaz gráfica, se sugiere implementar visualizaciones en tiempo real que muestren no solo el avance de los barcos, sino también los tiempos restantes y el uso de recursos en la simulación.
- Que se incluya un sistema de pruebas automatizadas para validar el comportamiento de los algoritmos de calendarización bajo distintos escenarios, como alta carga de barcos o situaciones de tráfico mixto. Este sistema debería generar casos de prueba con diferentes combinaciones de barcos y condiciones de tráfico, permitiendo a los desarrolladores identificar rápidamente ineficiencias o posibles errores en la implementación de los algoritmos.

References

[1] Carnegie Mellon University, “Pthreads Programming,” [Online]. Available: https://www.cs.cmu.edu/afs/cs/academic/class/15492-f07/www/pthreads.html.

[2] M. J. Quinn, *Parallel Programming in C with MPI and OpenMP*, 1st ed. McGraw-Hill, 2003, ch. 5. “Process Scheduling Algorithms”.