Materia: Programación Paralela.
Título: Simulación de restaurante.

Integrantes Y Matrículas:
Víctor Raúl Alcántara Sánchez - 20231146
Cristopher Santana - 20231193
Raymond Isaac Moreno Guridis - 20230950
Elier Moreta Encarnación - 20231168

Lider: Pamela Blanco Vincent - 20231668

Carrera: Desarrollo de software.

Docente: Erick Leonardo Pérez Veloz.

Fecha:
23/04/2025

1. Introducción

Presentación general del proyecto

El proyecto en sí consiste en el desarrollo de una simulación que digamos modela el funcionamiento de un restaurante como tal, abarcando desde la preparación de platillos hasta la entrega de dichos pedidos. La simulación integra diversas etapas y tareas que se ejecutan en paralelo, permitiendo observar de manera dinámica y realista cómo se coordinan las operaciones internas (como la preparación en cocina) y el servicio de delivery.

Entre sus características se incluyen la aleatoriedad en la disposición del menú, la ejecución simultánea de múltiples pedidos y la implementación de mecanismos de sincronización (como locks) para el manejo de datos compartidos (por ejemplo, el total de pedidos y las estadísticas de entregas).

Asimismo, la simulación incorpora elementos de incertidumbre, como la posibilidad de entregas tardías y la política de ofrecer la comida de forma gratuita si el pedido se demora más de 30 minutos en situaciones específicas. Esta estructura permite evaluar el comportamiento del sistema a través de indicadores como el speedup y la eficiencia, y estudiar estrategias de paralelización y escalabilidad en contextos de alta carga.

Justificación del tema elegido

La elección de este tema se fundamenta en diversos aspectos de relevancia tanto teórica como práctica:

- **Relevancia técnica y educativa:**
  El proyecto aborda conceptos fundamentales de la programación concurrente y paralela, permitiéndonos aplicar técnicas de gestión de hilos, sincronización y escalabilidad. Esto es crucial para entender cómo se diseñan sistemas robustos y eficientes en entornos que demandan alta.
- **Aplicación en entornos reales:**
  La simulación se basa en una simple dinámica prácticamente es un restaurante real, donde la coordinación entre diferentes áreas (cocina, servicio al cliente y delivery) es esencial para garantizar la satisfacción del cliente. La inclusión de elementos aleatorios y de probabilidad (como los tiempos de entrega) refleja los desafíos reales en la gestión logística y operacional, permitiendo explorar soluciones que pueden ser aplicables en escenarios del mundo real.
- **Optimización de procesos y evaluación del rendimiento:**
  Al incorporar métricas de rendimiento (speedup y eficiencia) y explorar distintos niveles de paralelización, el proyecto proporciona una plataforma para analizar y optimizar recursos en sistemas concurrentes.

Objetivos (general y específicos)

Objetivo General

Desarrollar una simulación de un restaurante que integre la preparación de platillos y el servicio de delivery, utilizando técnicas de programación concurrente y paralela para lograr una representación realista y escalable de las operaciones internas y logísticas.

Objetivos Específicos

- **Modelar y simular el flujo de trabajo de un restaurante:**
  Definir y programar el proceso para la preparación de los platillos, estableciendo etapas claras (por ejemplo, preparación de ingredientes, cocción, emplatado, etc.) que se ejecuten de forma aleatoria y en paralelo.
- **Implementar mecanismos de ejecución concurrente:**
  Utilizar hilos para simular la ejecución paralela de múltiples pedidos y platillos, permitiendo la coordinación simultánea entre la preparación en cocina y el envío del pedido.
- **Gestionar la sincronización de tareas compartidas:**
  Emplear herramientas de sincronización (como locks) para el manejo seguro y eficiente de datos críticos, tales como el número total de pedidos, tiempos de entrega y resultados de rendimiento.
- **Evaluar el rendimiento y la escalabilidad del sistema:**
  Aplicar métricas como speedup y eficiencia para analizar el impacto del paralelismo en la ejecución del sistema, y estudiar diferentes estrategias de escalamiento para optimizar el rendimiento bajo diversas cargas de trabajo.

2. Descripción del Problema

- **Contexto del problema seleccionado**

El problema se centra en simular el funcionamiento de un restaurante, donde se prepara la comida y se realizan entregas. La idea es imitar el proceso real de un restaurante como tal, desde que se cocina el platillo hasta que llega al cliente, incluyendo pasos de manera aleatoria y en función de distintos tiempos.

- **Aplicación del problema en un escenario real**

En un escenario real, este problema se puede ver reflejado en la operación de un restaurante que debe coordinar la cocina con el servicio de entrega. Cada pedido tiene varios platillos como tal los cuales se deben preparar al mismo tiempo, lo que permite simular el ajetreo del lugar con mucha demanda en si. Además, se añaden detalles como la posibilidad de que una entrega tarde y la política de dar la comida gratis si pasa un tiempo determinado, lo que ayuda a pensar en soluciones prácticas para mejorar el servicio.

- **Importancia del paralelismo en la solución**

El uso del paralelismo es muy importante en este proyecto, ya que nos permite que prácticamente varias tareas se realicen al mismo tiempo sin esperar a que termine la anterior. Esto significa que, mientras se prepara un platillo, otro puede estar en proceso de entrega. Esta forma de trabajar mejora el tiempo de respuesta del sistema y hace que la operación del restaurante sea mucho más eficiente.

3. Cumplimiento de los Requisitos del Proyecto

1. **Ejecución simultánea de múltiples tareas**

Se genera “x” cantidad de pedidos y se procesa cada platillo en paralelo. Mientras se prepara un platillo, el servicio de delivery puede comenzar a enviar la comida, lo cual simula múltiples tareas en paralelo.

2. **Necesidad de compartir datos entre tareas**

Las tareas deben acceder a estructuras comunes como el total de pedidos, entregas a tiempo y entregas tardías, utilizando mecanismos como locks.

3. **Exploración de diferentes estrategias de paralelización**

Descomposición de datos: Utilizamos descomposición de datos, ya que tomamos una lista de pedidos y la dividimos en partes independientes, y a cada pedido se le aplicó la misma lógica, de procesamiento y entrega.

Paralelismo de tareas: Utilizamos paralelismo de tareas ya que mientras se preparan los pedidos, otros hilos que simulan los repartidores van entregando los pedidos que están listos, simulando el funcionamiento real de un restaurante.

Sincronización de Recursos Compartidos: Utilizamos mecanismos de sincronización como lock, para asegurarnos de que solo un hilo acceda a una variable compartida a la vez y evitar errores.

Control de paralelismo: Utilizamos MaxDegreeOfParallelism para limitar el número máximo de procesadores.

4. **Escalabilidad con más recursos**

Se puede aumentar la cantidad de hilos utilizados para procesar los pedidos a medida que se disponga de más procesadores.

5. **Métricas de evaluación del rendimiento**

Tiempo: Medimos el tiempo total de cada ejecución, tanto de forma secuencial como de forma paralela con diferente número de procesadores.

Speedup: Calculamos el Speedup de cada ejecución paralela, para verificar qué tan rápido son en comparación con la ejecución secuencial.

Eficiencia: Calculamos la eficiencia para ver que tan bien se aprovechan los recursos de cada ejecución paralela.

Entregas a tiempo: Calculamos la cantidad de entregas a tiempo.

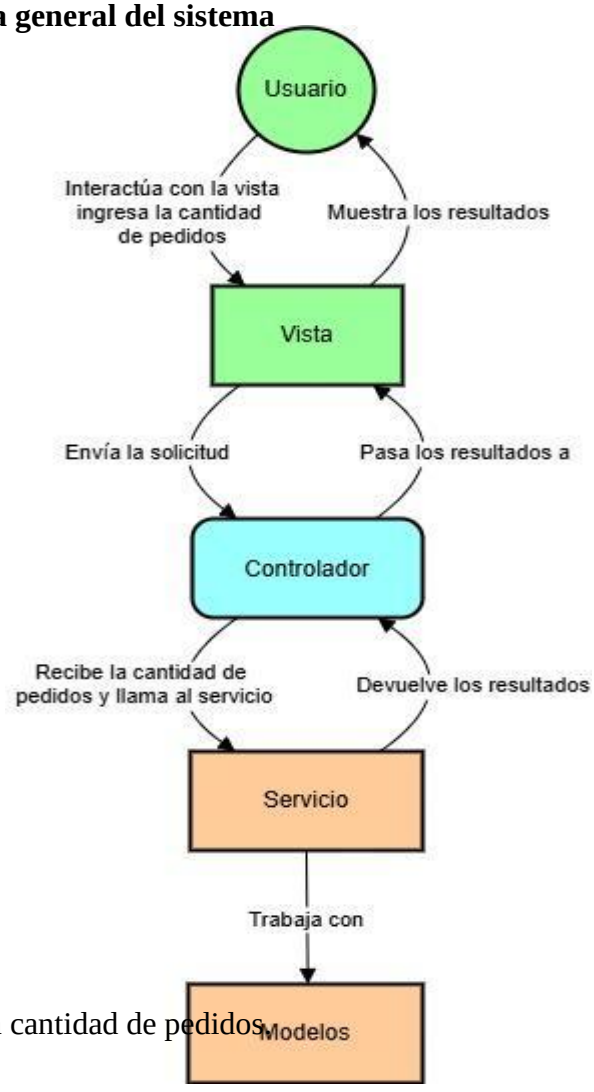
Entregas gratis: Calculamos la cantidad de entregas tardías, que salen gratis.

6. **Aplicación a un problema del mundo real**

Esta simulación refleja el funcionamiento de un restaurante real, con pedidos, platillos y tiempos de preparación, integrando aspectos como la posibilidad de retrasos y la probabilidad de entregas tardías.

4. Diseño de la Solución

- **Arquitectura general del sistema**



Vista ()

El usuario ingresa la cantidad de pedidos

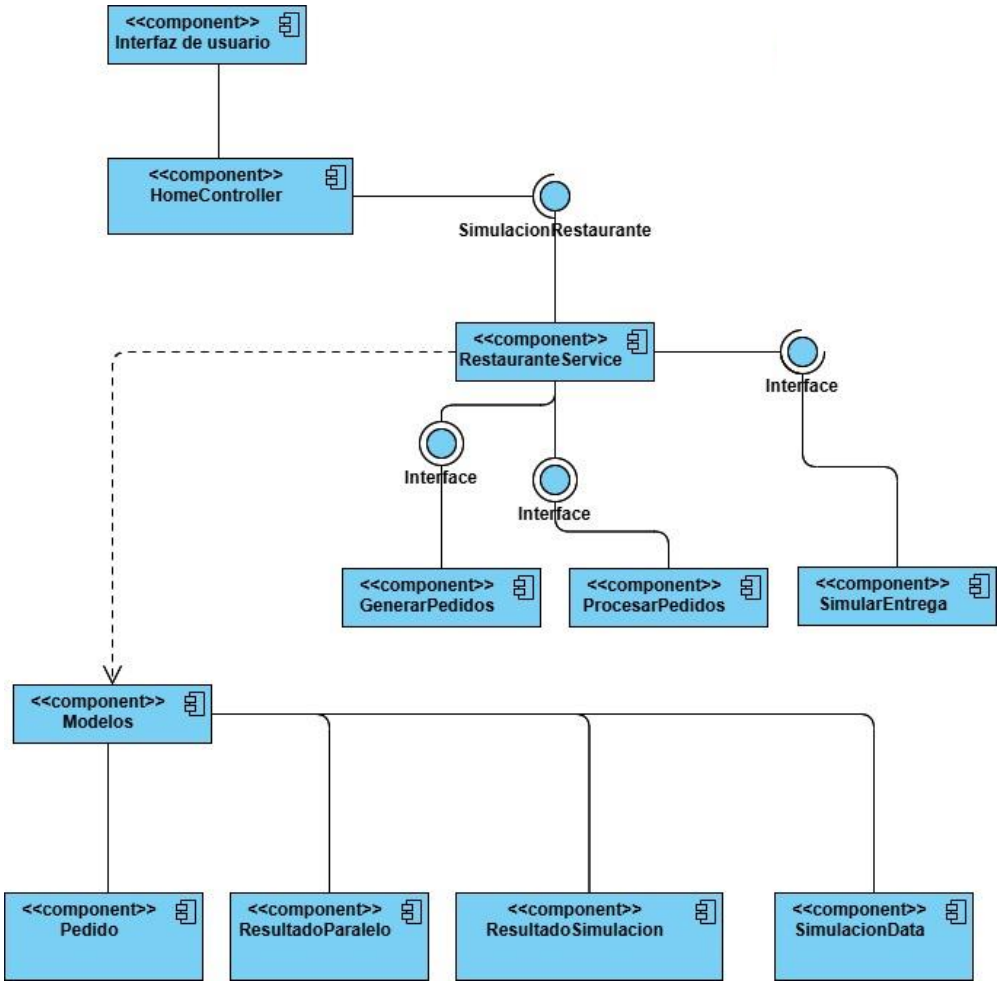
Controlador (RestauranteController)
Recibe la cantidad y llama al Servicio pasando ese valor.

Servicio (RestauranteService)
Genera los pedidos, ejecuta la simulación (secuencial y paralela), calcula métricas (como speedup y eficiencia), y retorna un el resultado.

Controlador (RestauranteController)
Recibe el resultado y se lo pasa a la vista.

Vista()
Muestra los resultados.

- **Diagrama de componentes/tareas paralelas**



- **Estrategia de paralelización utilizada**

Descomposición de datos: Utilizamos descomposición de datos, ya que tomamos una lista de pedidos y la dividimos en partes independientes, y a cada pedido se le aplicó la misma lógica, de procesamiento y entrega.

Paralelismo de tareas: Utilizamos paralelismo de tareas ya que mientras se preparan los pedidos, otros hilos que simulan los repartidores van entregando los pedidos que están listos, simulando el funcionamiento real de un restaurante.

Sincronización de Recursos Compartidos: Utilizamos mecanismos de sincronización como lock, para asegurarnos de que solo un hilo acceda a una variable compartida a la vez y evitar errores.

Control de paralelismo: Utilizamos MaxDegreeOfParallelism para limitar el número máximo de procesadores.

- **Herramientas y tecnologías empleadas**

C# ASP.NET CORE MVC, HTML, CSS, BOOSTRAP, JS