Universidad de Antioquia



Informe desarrollo del desafío 2

Escalante Fonseca Juan David

Murillo Hernández Marcos Jardel

Informática 2

Aníbal José Guerra Soler

Augusto Enrique Salazar Jiménez

Medellín 27/04/2024

**Índice**

**1-** Análisis del problema y consideraciones para la alternativa de solución propuesta. **2-** Diagrama de clases de la solución planteada. lógica de las tareas que usted definió para aquellos subprogramas. **3-** Algoritmos implementados debidamente intra-documentados. **4-** Problemas de desarrollo que afrontó. **5-** Evolución de la solución y consideraciones para tener en cuenta en la implementación.

1. **Análisis del problema y consideraciones para la alternativa de solución propuesta.**

El **objetivo principal** del simulador es modelar la red del metro, permitiendo realizar la gestión de las estaciones y líneas, además, debemos simular los tiempos de viaje entre estaciones de una misma línea.

El metro es uno de los sistemas de transporte mas usados en las ciudades grandes, permitiendo trasladar gran cantidad de personas de un punto A hacia un punto B en poco tiempo.

**Definiciones:**

* **Red:** Conjunto de líneas dentro de un área específica.
* **Línea:** Se identifica por su nombre y a cada línea se le especifica un tipo de transporte.

**Los tipos de transporte pueden ser tren o tranvía.**

**Los puntos clave a considerar son:**

1. **Modelado de la red, líneas y estaciones.**

* La red del metro consistirá en múltiples líneas sin bifurcaciones ni bucles y con tráfico bidireccional.
* Las estaciones pueden ser comunes o de transferencia, donde las últimas permiten el cambio entre distintas líneas.

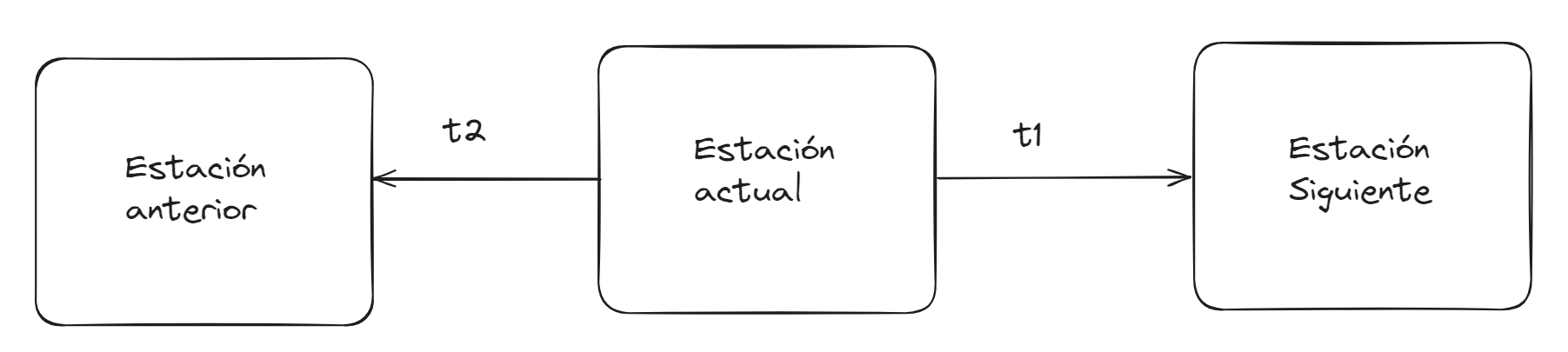
1. **Operaciones a realizar:**

* Agregar y eliminar estaciones y líneas.
* Consultar la cantidad de líneas en una red y estaciones en una línea.
* Simular tiempos de viaje entre estaciones de una misma línea.

1. **Restricciones:**

* No eliminar líneas que contienen estaciones de transferencia.
* Una estación de transferencia puede aparecer en varias líneas pero es única en una línea específica.

**Para cada estación:** Se conoce el tiempo en segundos que toma, llegar desde la estación actual hasta la estación siguiente o a la estación anterior.



t1: Tiempo desde la estación actual hasta la siguiente estación y viceversa.

t2: Tiempo desde la estación actual hasta la estación anterior y viceversa.

1. **Diagrama de clases de la solución planteada. lógica de las tareas que usted definió para aquellos subprogramas**.

Para el simulador del metro, el diagrama de clases se organiza de la siguiente forma para reflejar adecuadamente las relaciones entre las diferentes clases del sistema:

1. **Redmetro:**

* Atributos:
  + Líneas: Lista de objetos de la clase ***lineaMetro.***
* Métodos:
  + agregarLinea(LineaMetro): Añade una nueva línea a la red.
  + eliminarLinea(LineaMetro): Elimina una línea de la red, verificando que no contenga estaciones de transferencia.
  + obtenerNumeroDeLineas(): Retorna el número total de líneas en la red.
  + obtenerTotalEstaciones(): Retorna el número total de estaciones en la red, considerando las estaciones de transferencia.

1. **LineaMetro:**

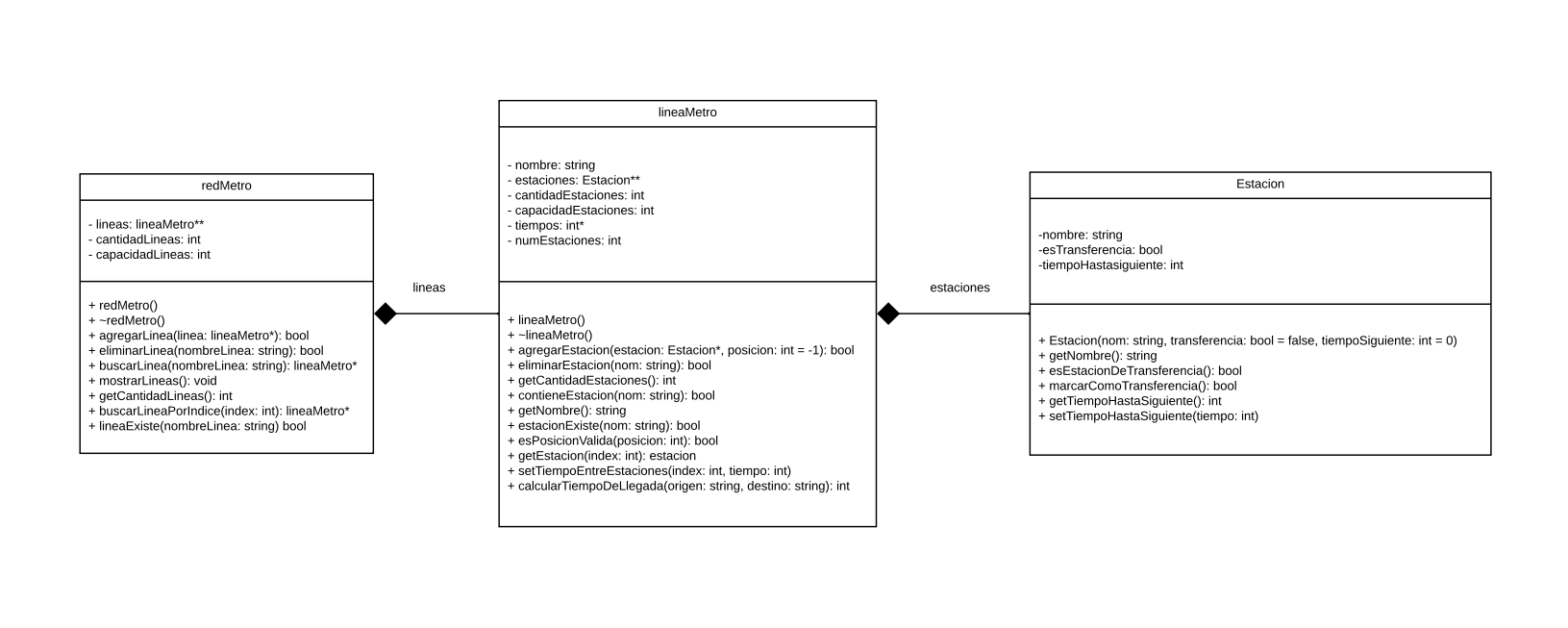
* Atributos:
  + nombre: Nombre de la línea.
  + estaciones: Lista de objetos de la clase Estacion.
  + esTranvia: Booleano que indica si la línea es de tranvía o no.
* Métodos:
  + agregarEstacion(Estacion, posicion): Agrega una estación en una posición específica dentro de la línea.
  + eliminarEstacion(Estacion): Elimina una estación de la línea, excepto estaciones de transferencia.
  + obtenerNumeroDeEstaciones(): Retorna el número de estaciones en la línea.
  + perteneceEstacion(Estacion): Verifica si una estación pertenece a la línea.

1. **Estacion:**

* Atributos:
  + nombre: Nombre de la estación.
  + tiempoHastaSiguiente: Tiempo en segundos hasta la siguiente estación.
  + tiempoHastaAnterior: Tiempo en segundos hasta la estación anterior.
  + lineasConectadas: Lista de nombres de líneas a las que esta estación está conectada (para estaciones de transferencia).
* Métodos:
  + actualizarTiempoHastaSiguiente(int): Actualiza el tiempo hasta la siguiente estación.
  + actualizarTiempoHastaAnterior(int): Actualiza el tiempo hasta la estación anterior.

**Relaciones:**

* Relación entre **RedMetro** y **LineaMetro**: Una **RedMetro** contiene múltiples **LineaMetro**.
* Relación entre **LineaMetro** y **Estacion**: Una **LineaMetro** tiene varias **Estacion**.
* Relación de Estaciones de Transferencia: Una **Estacion** puede estar conectada a múltiples **LineaMetro**.



**4-Problemas de desarrollo que afrontó:**

Un problema que se afronto es definir que iba a ir en cada clase sus atributos y métodos para tener una estructura para empezar a desarrollar el desafío.

Sin ser consientes estábamos usando el método cin.ignore(); en varias ocasiones sin darnos cuenta que este hacia que muchas veces se perdiera la información que el usuario ingresaba por lo tanto las funciones no podían realizar su trabajo.

En cada caso teníamos que manejar varias excepciones para los diferentes casos que se presentaran y que el sistema no se cayera o a su vez los otros casos no realizaran de manera correcta su funcionamiento simplemente porque había fallado una excepción y esto hacia que la red metro no fuera la correcta

Varias veces presentábamos problemas al no poner las cabeceras de las funciones en los headers y esto provocaba que el programa no funcionara.

Para calcular el tiempo entre estaciones de esta línea no estaba retornando un valor al usuario y tampoco si se borraba una estación estaba redimensionándolas, no mostraba el tiempo que debería ser entre estaciones el correcto.

se presentaron conflictos en GitHub con temas de compatibilidad

**5- Evolución de la solución y consideraciones para tener en cuenta en la implementación.**

Se empezaron definiendo las clases a su vez los métodos y atributos de cada clase por consiguiente pasamos a crear líneas y eliminarlas respectivamente también saber cuántas líneas había en la red metro por ende pasamos a crear estaciones que pertenecían a una línea a su vez eliminándolas también sabiendo cuantas estaciones habían en la red luego empezamos a unir las líneas por medio de una estación de transferencia, en este punto del desarrollo empezamos a arreglar y revisar el manejo de excepciones en diferentes casos que se podían presentar, luego saber si una estación dada pertenece a una línea en especifica y por ultimo el tiempo entre estaciones de la misma línea.

**-Consideraciones para tener en cuenta:**

Se aclara que solamente se esta considerando que solo es para una sola red metro. Solo se crea una línea sin estaciones. Para crear otra línea la línea a la que se va a conectar debe tener estaciones si no no es creada. Si hay dos o más líneas en la red metro ya ninguna línea se puede eliminar. El caso de las estaciones solo se crea de a una estación si se desea crear otra se debe volver a ingresar desde el menú. En el menú principal se incluyó el punto I para el calculo de los tiempos entre estaciones que este aparecía como un subprograma en el archivo del desafío 2.

