Informe Desarrollo Parcial II

Manuel Esteban Orjuela Montealegre

Universidad De Antioquia

Facultad de Ingeniería

Departamento de Electrónica y Telecomunicaciones

Informática II

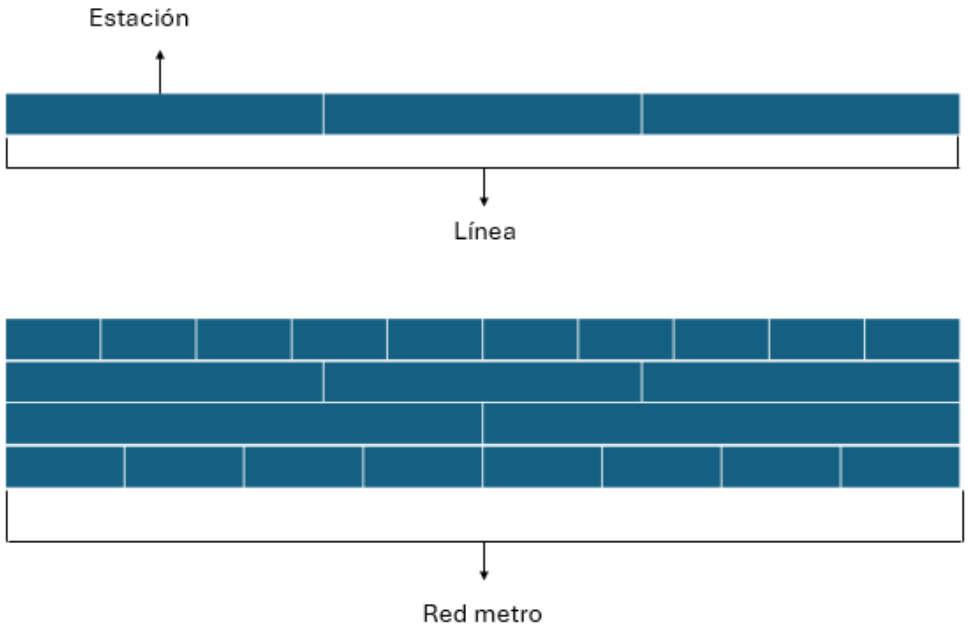
Semestre 2024 – 1

Análisis del Problema y Consideraciones para la Alternativa de la Solución Propuesta

Estructura de Datos definida.

Para la solución planteada se tomó en cuenta la siguiente premisa:

. Modelar una estructura dinámica doble con la particularidad de que las dimensiones de las componentes internas no son uniformes.



De acuerdo a la ilustración anterior, se establece una relación de 1 a muchos entre las clases definidas para la solución, siendo así:

. Un conjunto de estaciones es una línea.

. Un conjunto de líneas corresponde a una red metro.

Estaciones de transferencia.

Una característica destacada del desafío son las estaciones de transferencia, las cuales pueden estar conectadas con múltiples líneas del sistema, lo que puede llevar a la creación de bucles. Para abordar este desafío, se implementó una validación que nos permite determinar si una línea ya está conectada. Esto es importante porque, al buscar un modelo lineal, una vez que existe una conexión en la red, se establece un camino para llegar a cualquier punto sin necesidad de crear nuevas conexiones.

Por otro lado, al crear una nueva estación de transferencia y asociarla con dos líneas diferentes, existe la posibilidad de que esta estación recién creada se conecte con una estación

de transferencia existente en la red. Esto permite cumplir con el requisito de que una estación de transferencia puede estar asociada con más de dos líneas dentro del sistema.

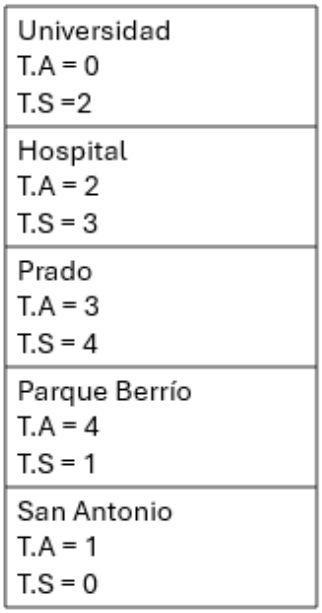
Cálculo de tiempo entre estaciones.

La decisión tomada para el calculo del tiempo fue tener dos atributos en cada estación que nos indicara cuánto tiempo tarda de ir de la estación actual a la anterior y siguiente. Teniendo en cuenta lo siguiente:

Si la estación se ubica en la posición inicial, el tiempo anterior es 0.

Si la estación se ubica en la posición final, el tiempo siguiente es 0.

De esta forma, para saber cuánto tardaremos en ir de A – B, será sumar el valor de los atributos.



Ejemplo Línea A

Por ejemplo. Tiempo de A – B
2+3+4+1 = 8 Minutos.
Sumando los tiempos siguientes.

Diagrama de Clases de la Solución Planteada y Tareas Definidas para el Desarrollo de Algoritmos

El diagrama clases se encuentra en un documento de Power Point en la carpeta 'Diagrama de Clases' del repositorio de desarrollo.

Para la clase Línea.

1. Módulo de Validaciones.
2. Módulo de Errores.
3. Módulo de Funciones de Manejo (Funciones auxiliares a las tareas que requiere el cliente).
4. Módulo de Operaciones de Estaciones.

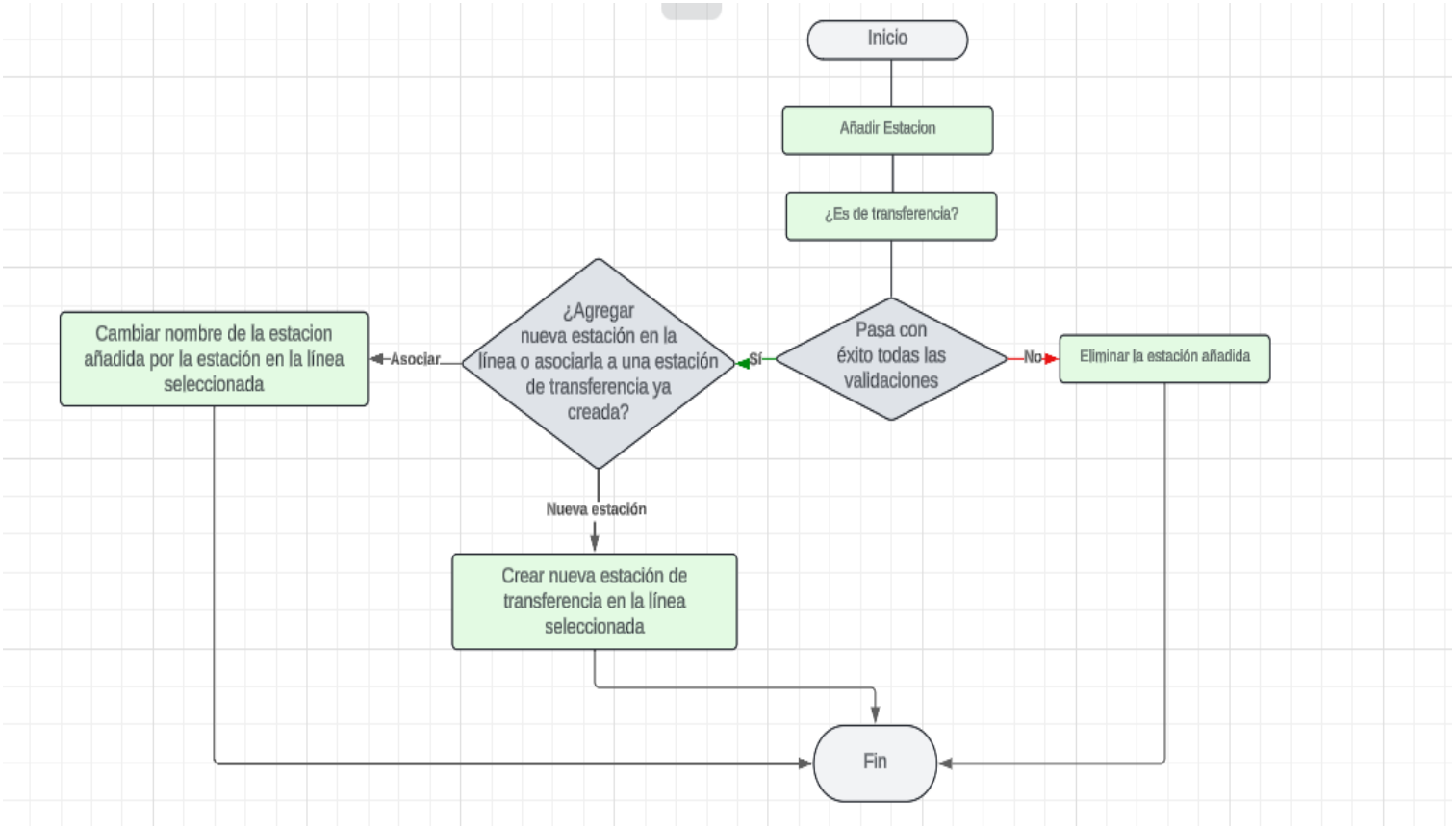
Para la clase Metro.

1. Módulo de Validaciones.
2. Módulo de Errores.
3. Módulo de Funciones de Visualización.
4. Módulo de Funciones de Manejo. (Funciones auxiliares al manejo de estaciones de transferencia).
5. Módulo de Operaciones de Líneas.
   5.1 Módulo de Operaciones de Estaciones.

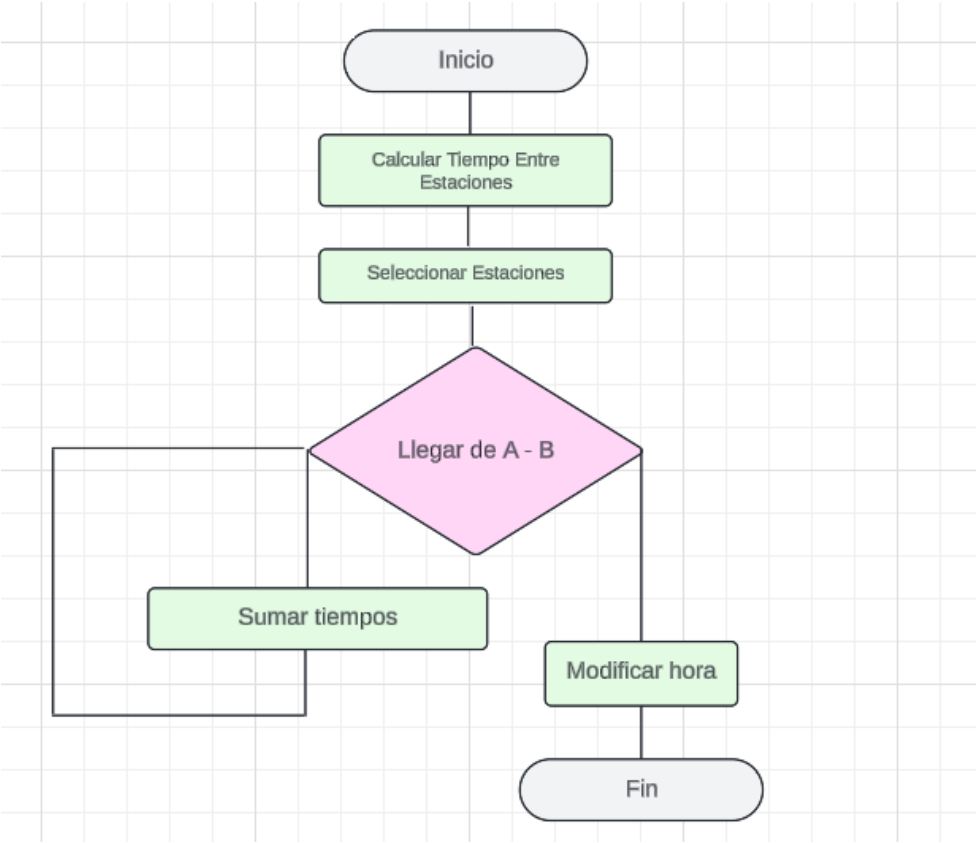
Algoritmos Implementados

En este apartado se describirán mediante un diagrama de flujo los algoritmos fundamentales de la implementación.

Añadir una estación de transferencia.



Cálculo de tiempo entre estaciones.



Problemas en el Desarrollo y Evolución de la Solución

Momento 1. Creación estructura de datos.

En la primera implementación del programa, codifiqué la definición de las utilizadas en la solución, establecí atributos y características de cada una junto con tareas sencillas como constructores, getters, y setters. Además, hice las primeras pruebas de utilizar arreglos dinámicos de clases (Estación y línea).

Momento 2. Implementación general de la clase Red.

Primera aproximación al funcionamiento del arreglo bidimensional dinámico Red_Metro junto con tareas básicas como añadir y eliminar línea o estación.

Problema Enfrentado: Debido a la falta de liberación de memoria con el operador delete[] en el destructor de la clase Línea, surgieron fugas de memoria al operar directamente con la red. Posteriormente, al implementar el uso correcto del operador delete[] en el destructor, se encontraron errores al inicializar los punteros de líneas y estaciones en nullptr. Este problema se manifestaba especialmente al añadir una línea y luego intentar eliminarla, ya que la condición de verificación previa no se cumplía, lo que provocaba una falla en la ejecución del programa. La solución adoptada fue introducir otro parámetro para las operaciones relacionadas con el contenido.

Momento 3. Implementación de Validaciones y Errores.

Problema Enfrentado: El programa experimentaba abortos durante la ejecución debido al manejo inadecuado de las entradas por consola. Como solución, se propuso implementar un conjunto de validaciones y manejo de errores tanto en la clase Red como en la clase Línea. Por otro lado, las entradas por consola también fueron monitoreadas debido a problemas de lectura en el buffer.

Momento 4. Implementación estaciones de transferencia.

Problema Enfrentado: Al existir una correlación entre las estaciones de transferencia y las líneas, como primera solución se propuso utilizar otro arreglo dinámico en la clase red que almacenara todas las líneas conectadas, pero al ser demasiado ineficiente debido al manejo dinámico de éste, se optó por añadir un atributo en la clase línea que indicara si ya estaba conectada a la red y al final utilizar el algoritmo descrito anteriormente.

Momento 5. Errores de lectura de datos.

Problema Enfrentado. En cierto caso específico, las entradas por consola pueden corromper los datos almacenados causando una ejecución del programa, para evitar esto, el funcionamiento de la red debe iniciar con mínimo dos líneas para así considerarse una red.

Momento 6. Implementación aplicación tiempo y demás requerimientos del cliente.

En esta fase hubo un problema de fuga de memoria al tener una red demasiado extensa, por lo que tuvo que realizarse un cambio con respecto al destructor de la clase línea ya que solo liberaba memoria si el tamaño de la línea era distinto de 0, lo cual en algunas ocasiones podía

crear problemas si no se liberaban en todos los casos. Igualmente, se implementó una función para calcular el tiempo entre estaciones utilizando parámetros como tiempo anterior y tiempo siguiente (Algoritmo explicado en el apartado anterior) junto con el calculo del total de estaciones en la red, la formula utilizada para esta tarea fue:

No. Total Estaciones = No.Estaciones cada línea – (Líneas conectadas – 1).