



UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE CIENCIAS

Modelado y programación.

Rosa Victoria Villa Padilla

[2025-1] (12/11/2024)

Proyecto 2:

Rutas óptimas para el Pumabús.

Equipo: Christian.

- Leon Navarrete Adam Edmundo
- Morales Martínez Edgar Jesús
- Rubio Resendiz Marco Antonio
- Valencia Pérez Guillermo Emanuel



Ciudad Universitaria, Cd. Mx. 2024

Problemática.

El Pumabus es el servicio de transporte más utilizado por estudiantes, profesores y trabajadores de Ciudad Universitaria para moverse dentro del campus. Pero a pesar de su rol clave en la movilidad, la complejidad de sus rutas, la afluencia de estas y el desconocimiento del funcionamiento del sistema ha resultado un problema para múltiples usuarios del área que complica el uso del servicio.

Actualmente, los usuarios carecen de herramientas para conocer una ruta óptima para llegar de un punto a otro de C.U. Por ello nuestro objetivo con el 2do proyecto de Modelado y Programación es el desarrollo de un software que será capaz de ofrecerle a los usuarios la ruta que deberían tomar para trasladarse por Ciudad Universitaria haciendo uso del Pumabus.

Elección de patrones de diseño.

MVC.

Nuestro sistema se divide en tres responsabilidades esenciales: Manejo de las rutas, visualización de las rutas y la interacción del usuario. Esta división es adecuada para segmentar el proyecto en sus tres componentes fundamentales.

- **Modelo:** Parte encargada de procesar la información referente a como se construyen las rutas que estarán a disposición del usuario, consiste en un paquete completo de nombre edd/ que cuenta con las estructuras de datos que se emplearán para la lógica del software. En este caso es fundamental este componente, pues se trata de una implementación de Gráficas y Gráficas dirigidas que nos permiten realizar el cálculo de trayectorias óptimas. Esta sección también abarca el paquete composite/ que se encarga de configurar las diferentes rutas del sistema de movilidad
- **Vista:** Parte encargada de convertir la información del modelo en algo visualizable y comprensible por el usuario. Este componente está comprendido por el paquete graficable/ que nos permite procesar cualquier gráfica dirigida cuyo tipo sea una implementación de la interfaz VerticeCoordenado para proporcionar una gráfica (en este caso codificada en svg) que constituye una representación visual de la información contenida en la gráfica. Esta sección también abarca el paquete menús/ que se encarga de proporcionar al usuario una interfaz gráfica con la que interactuar.
- **Controlador:** Sección encargada de gestionar las entradas del usuario para constituir el modelo (una gráfica y una trayectoria) y proporcionarle al usuario un objeto visible (la gráfica en svg). Este componente se trata de la clase ClienteRemoto que se encarga de procesar las solicitudes del usuario a través de un proxy que puede acceder a los datos cargados por el servidor. En general se encarga de pedir y proporcionar la información que necesita el usuario para mostrar su ruta.

Singleton.

Se decidió implementar singleton para la clase Servidor con el objetivo de garantizar que los usuarios accedan a la misma información de las rutas que haya sido leída por el servidor una vez este haya sido encendido. Esto permite evitar inconsistencias con la información, así como asegurar que todos los usuarios están actualizados.

Proxy.

El patrón proxy será el encargado de gestionar el acceso de los usuarios a la información del sistema de rutas. Este patrón nos permite centralizar las solicitudes de los usuarios, esta parte en conjunto con singleton serán particularmente útiles si es que se busca manejar un número potencialmente alto de solicitudes (como lo haría idealmente una plataforma de movilidad).

Nota: Se hace uso de hilos y sockets para las solicitudes de los usuarios, se crea un nuevo hilo por cada solicitud nueva. Así no se mezclan los accesos al servicio y dejamos al socket abierto a la espera de nuevos solicitantes.

Builder.

Se implementará el patrón de diseño builder (paquete graficable/GraficadorBuilder) para estandarizar el proceso de construcción de gráficas svg. Pues al disponer de múltiples opciones para la creación de las distintas estructuras en la clase GraficadorGrafo, es fundamental contar con una clase que sea capaz de utilizar el graficador para constituir un objeto inmutable en un orden específico. Pues se busca que al momento de graficar se genere un bloque de código estable a la prioridad en la que las figuras se muestran en el objeto construido. Es decir, el builder nos permite centralizar la lógica de graficado a una secuencia jerárquica de graficar objetos en un orden específico (primero aristas, después trayectorias y después vértices).

Composite.

Se decidió emplear composite (paquete composite) para este proyecto pues, al tratarse de un sistema de movilidad con múltiples rutas. Es fundamental que cada una de ellas forme parte del sistema pero que al mismo tiempo sea desacoplable del mismo, es decir, es importante que el sistema completo esté compuesto por sistemas más básicos y que su tarea tenga el mismo enfoque.

El enfoque dado a este patrón será constituir una gráfica de gráficas dirigidas donde cada gráfica dirigida es una ruta del pumabus y se conecta con las demás rutas a través de las estaciones que comparte con ellas. Esto es importante para permitirle al software incorporar una forma de componer una gráfica dirigida del sistema completo para la búsqueda de rutas, así como de permitirle a las rutas más básicas buscar las mismas rutas óptimas (dentro de la ruta concreta por supuesto). Esto es importante por cuestiones del desacoplamiento del objeto compuesto como de los que lo componen.

Factory.

Se emplea el patrón de diseño factory (paquete factory/) para la construcción de instancias de GraficaDirigida que representarán las rutas del sistema de movilidad. El objetivo es que cada una de las fábricas pueda fabricar las estaciones de una ruta correspondiente proporcionando información particular para las estaciones de la ruta a la que pertenece. Esto desacopla la lógica de fabricación de las estaciones de la creación de las gráficas dirigidas que las contienen, facilitando el encapsulamiento y manteniendo la consistencia de los datos contenidos en las gráficas.

Por lo anterior tendremos una fábrica de estaciones por cada una de las 12 rutas del sistema de movilidad.

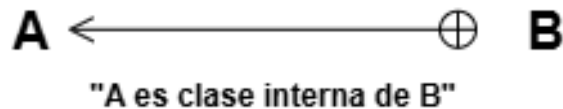
Strategy.

Se utiliza el patrón de diseño strategy (paquete strategy/) para poder dar la posibilidad al usuario de escoger un criterio de optimización para la trayectoria que desee solicitar. De esta forma podemos implementar distintos tipos de algoritmos para el cálculo de rutas. Por el momento, se han implementado dos estrategias de optimización de rutas:

- MenorAfluencia: Para considerar el dato de afluencia al momento de calcular la ruta, para obtener la trayectoria menos concurrida.
- MenorNumeroDeParadas: Para considerar que se recorra la menor cantidad de estaciones en el trayecto. De esta manera se da la trayectoria más fluida.

Respecto al diagrama de clases.

Es importante mencionar que el paquete edd/ de nuestro proyecto comprende implementaciones propias de estructuras de datos para las cuales java no proporciona una implementación de estas (las gráficas). Estas estructuras de datos emplean clases anidadas para su funcionamiento, por ello que se hace uso del operador de anidamiento para denotar esta parte.



Respecto a la compilación del proyecto.

El proyecto hace uso de un pom.xml para su instalación. Esto fue comentado en clase con la profesora y se nos dio permiso debido a dos aspectos particulares de nuestro proyecto:

- **Uso de JavaFX:** El proyecto emplea la versión javafx 21.0.5 así como un plugin que nos permite configurar el plugin javafx-maven-plugin con la versión 0.0.8 del repositorio de Maven para que el usuario no tenga que contar previamente con una versión de JavaFX instalada en su sistema. Además, esto nos permite usar cualquier versión de java para compilar el proyecto, relegando la ejecución del sistema a un comando de Maven.
- **Uso de Batik:** Es esencial para nuestro proyecto el uso de la librería Batik, pues el programa genera en última instancia un archivo .svg para los cuales JavaFX no cuenta con un soporte para mostrarlos en las interfaces gráficas. Por ello que se decidió emplear Batik para transcodificar archivos .svg a archivos .png que puedan ser mostrados por las interfaces gráficas.

Respecto al uso del proyecto.

El proyecto fue desarrollado en las versiones de java 11.0.22 y java 21.0.5 y se empleó Linux (Debian y Ubuntu) durante su desarrollo. Para ejecutarlo (en cualquier sistema que cuente con Maven y java) se deben usar los siguientes comandos en terminal:

Cabe mencionar que para usar el programa es necesario que se utilicen al menos dos terminales, una para encender el servidor del sistema y otra para hacer uso del programa como usuario. Es necesario que el servidor esté encendido para que se pueda usar el programa.

Para compilar todas las clases y generar los archivos .class:

```
$ mvn install
```

Para encender el servidor (Terminal 1):

```
$ java -cp target/classes/ mx.unam.ciencias.modeloado.proyecto2.proxy.ClienteRemoto
```

Para para usar el programa como usuario (Terminal 2):

```
$ mvn javafx:run
```

Nota 1: Una vez el servidor detecta que ya no hay usuarios usando el sistema se muestra el siguiente mensaje en la terminal 1:

```
$ No hay más clientes conectados. Cerrando el servidor.
```

Esto se hizo con el único propósito de que el programa terminará.

Nota 2: Una vez generada una gráfica con el servidor, el sistema puede mostrar la siguiente advertencia en la terminal 2:

```
$ (java:4016): Gdk-WARNING **: 21:58:36.974: XSetErrorHandler() called with a GDK error trap pushed. Don't do that.
```

Esto se debe a un conflicto menor entre JavaFX y el sistema que gestiona las ventanas emergentes GDK. Este conflicto solamente ocurre en sistemas Unix y no es un conflicto grave, solamente es una ligera discrepancia en como se muestran ciertos entornos gráficos y no afecta el funcionamiento del programa.

Ejemplo de uso:

Compilar el programa y encender el servidor:

Instalando el proyecto

Enciende servidor

```
toctny@debian-Marco: ~/Documentos/Code/Modelado/proyecto2/Proyecto02_Christian
toctny@debian-Marco: ~/Documentos/Code/Modelado/proyecto2/Proyecto02_Christian$ mvn install
[INFO] Scanning for projects...
[INFO]
[INFO] -----< mx.unam.ciencias.modeloado.proyecto2:proyecto >-----
[INFO] Building Proyecto 2 MVP
[INFO] -----[ jar ]-----
[INFO]
[INFO] --- maven-enforcer-plugin:3.0.0-M3:enforce (enforce-maven) @ proyecto ---
[INFO]
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ proyecto ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory /home/toctny/Documentos/Code/Modelado/proyecto2/Proyecto02_Christian/src/main/resources
[INFO]
[INFO] --- maven-compiler-plugin:3.8.1:compile (default-compile) @ proyecto ---
[INFO] Changes detected - recompiling the module!
[INFO] Compiling 42 source files to /home/toctny/Documentos/Code/Modelado/proyecto2/Proyecto02_Christian/target/classes
[INFO]
[INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ proyecto ---
[INFO] skip non existing resourceDirectory /home/toctny/Documentos/Code/Modelado/proyecto2/Proyecto02_Christian/src/test/resources
[INFO]
[INFO] --- maven-compiler-plugin:3.8.1:testCompile (default-testCompile) @ proyecto ---
[INFO] Changes detected - recompiling the module!
[INFO] Compiling 2 source files to /home/toctny/Documentos/Code/Modelado/proyecto2/Proyecto02_Christian/target/test-classes
[INFO]
[INFO] --- maven-surefire-plugin:2.22.4:test (default-test) @ proyecto ---
[INFO] Surefire report directory: /home/toctny/Documentos/Code/Modelado/proyecto2/Proyecto02_Christian/target/surefire-reports
[INFO]
[INFO] -----
[INFO] T E S T S
[INFO] -----
[INFO] Running mx.unam.ciencias.modeloado.proyecto2.test.TestGrafica
[INFO] Tests run: 27, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.989 sec
[INFO] Running mx.unam.ciencias.modeloado.proyecto2.test.TestGraficaOrlrigida
[INFO] Tests run: 27, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 1.34 sec
[INFO]
[INFO] Results :
[INFO]
[INFO] Tests run: 54, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO] --- maven-jar-plugin:3.2.0:jar (default-jar) @ proyecto ---
[INFO] Building jar: /home/toctny/Documentos/Code/Modelado/proyecto2/Proyecto02_Christian/target/projecto2.jar
[INFO]
[INFO] --- maven-install-plugin:2.4:install (default-install) @ proyecto ---
[INFO] Installing /home/toctny/Documentos/Code/Modelado/proyecto2/Proyecto02_Christian/target/projecto2.jar to /home/toctny/.m2/repository/mx/unam/ciencias/modelado/proyecto2/projecto2.jar
[INFO] Installing /home/toctny/Documentos/Code/Modelado/proyecto2/Proyecto02_Christian/target/projecto2.jar to /home/toctny/.m2/repository/mx/unam/ciencias/modelado/proyecto2/projecto2-MVP/projecto2-MVP.jar
[INFO]
[INFO] BUILD SUCCESS
[INFO]
[INFO] Total time: 7.236 s
[INFO] Finished at: 2024-11-10T20:50:50-06:00
[INFO]
[INFO] toctny@debian-Marco: ~/Documentos/Code/Modelado/proyecto2/Proyecto02_Christian$ java -cp target/classes/ mx.unam.ciencias.modeloado.proyecto2.proxy.ClienteRemoto
Servidor iniciado en el puerto 12345
```

Ejecutar el programa como usuario:

Inicia el programa

Conecta con el servidor

```
toctny@debian-Marco: ~/Documentos/Code/Modelado/proyecto2/Proyecto02_Christian
toctny@debian-Marco: ~/Documentos/Code/Modelado/proyecto2/Proyecto02_Christian$ mvn javafx:run
[INFO] Scanning for projects...
[INFO]
[INFO] -----< mx.unam.ciencias.modeloado.proyecto2:proyecto >-----
[INFO] Building Proyecto 2 MVP
[INFO] -----[ jar ]-----
[INFO]
[INFO] >>> javafx-maven-plugin:0.0.8:run (default-cli) > process-classes @ proyecto >>>
[INFO]
[INFO] --- maven-enforcer-plugin:3.0.0-M3:enforce (enforce-maven) @ proyecto ---
[INFO]
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ proyecto ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory /home/toctny/Documentos/Code/Modelado/proyecto2/Proyecto02_Christian/src/main/resources
[INFO]
[INFO] --- maven-compiler-plugin:3.8.1:compile (default-compile) @ proyecto ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] <<< javafx-maven-plugin:0.0.8:run (default-cli) < process-classes @ proyecto <<<
[INFO]
[INFO] --- javafx-maven-plugin:0.0.8:run (default-cli) @ proyecto ---
[INFO]
[INFO] Conectado al servidor en localhost:12345
```

Interfaz para seleccionar datos sobre la ruta deseada:

Una de las rutas del sistema:

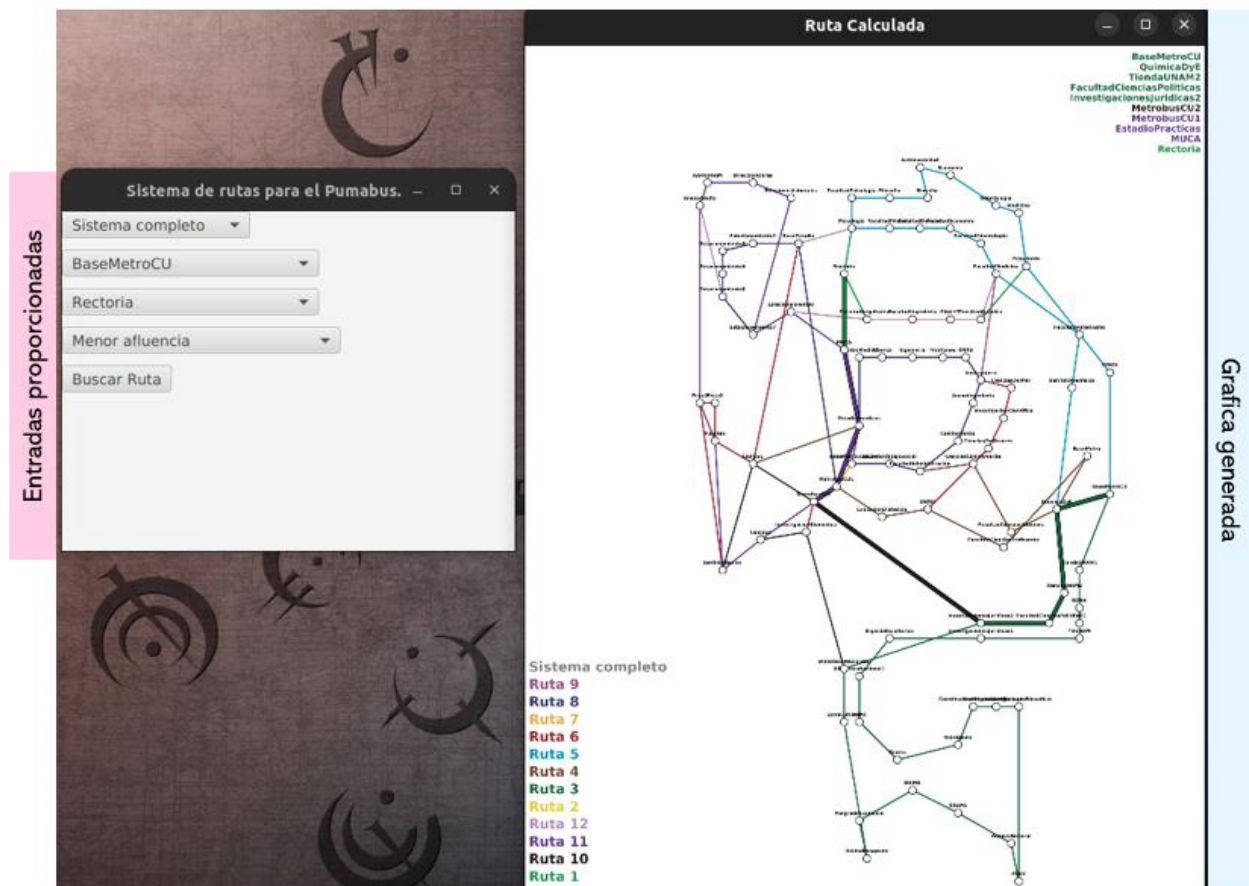
La estación de origen:

La estación destino:

Criterio de optimización de la ruta:

Para hacer la solicitud:

Ejemplo [Se busca la ruta de menor afluencia para ir desde la BaseMetroCU a Rectoría]:



Ejemplo [Se busca la ruta de menos paradas para ir desde la BaseMetrobusCU a CienciasCaminoVerde usando la ruta 2]:

Entradas proporcionadas

Sistema de rutas para el Pumabus.

Ruta 2

BaseMetrobusCU

CienciasCaminoVerde

Menor número de paradas

Buscar Ruta

Ruta Calculada

BaseMetrobusCU

MetrobusCU1

EducacionDistancia

DGTIC


FacultadCienciasProfesiones

BaseMetro

QuimicaByE

FacultadCienciasAlumnos

CienciasCaminoVerde



Gráfica generada

Una vez cerradas las ventanas, termina la ejecución:

Termina ejecución

```
[INFO] --- javafx-maven-plugin:0.0.8:run (default-cli) @ proyecto ---
Conectado al servidor en localhost:12345
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 6.081 s
[INFO] Finished at: 2024-11-10T21:04:20-06:00
[INFO] -----
tocny@debian-Marco: ~/Documentos/Code/Modelado/proyecto2/Proyecto02_Christian$
```

Terminal 2

Si el servidor ya no cuenta con usuarios usando el software, se cierra:

Apaga servidor

```
No hay más clientes conectados. Cerrando el servidor.
tocny@debian-Marco: ~/Documentos/Code/Modelado/proyecto2/Proyecto02_Christian$ java -cp target/classes/ mx.unam.ciencias.modeloado.proyecto2.pro
Servidor iniciado en el puerto 12345
cliente conectado.
cliente desconectado.
No hay más clientes conectados. Cerrando el servidor.
tocny@debian-Marco: ~/Documentos/Code/Modelado/proyecto2/Proyecto02_Christian$
```

Terminal 1