Universidad Latinoamericana de Ciencia y Tecnología.

Ingeniería informática

Programación concurrente
1CO24-

Proyecto Final
Proyecto Programado

Integrantes:
Dylan Rojas Gallo.
Daniel Segnini Arias
Daniel Alpizar Batista
Carlos Villanueva

Cuatrimestre I

2024.

Indice

Introducción

Para este proyecto, se nos ha solicitado realizar en el lenguaje java, un programa que simule el movimiento de 10 buses dentro de una ruta con 20 paradas. Se deberán de crear los autobuses como hilos, ya que el movimiento de los 10 buses durante la ruta será simultáneo. La ruta es un circuito, por lo que los buses seguirán haciendo la ruta hasta que el día finalice, sin importar cuantas veces repiten la ruta. Deberemos de crear un rango de tiempo, ya que obviamente no se va a correr el programa por las 19 horas, por lo que se definirá que 1min equivale a 1 segundo. Esto podrá variar durante el desarrollo del programa. En el caso de nosotros, deberemos de hacer la ruta San José - Cartago - San José. La idea de este proyecto, es simular una aplicación que podría hacerse pública, como usuario finales, la población costarricense que utiliza autobuses. Hoy en día no existe ninguna aplicación que siga las rutas de los autobuses y que nos avisen por donde van, por lo que esta aplicación sería de gran ayuda para todos los costarricenses.

Funcionalidad de la aplicación

Apenas el programa se ejecute, se abrirá la interfaz de usuario, donde habrá un botón para iniciar la ruta. Dentro de la interfaz de usuario, se encontrará un mapa, donde será donde se verá reflejado el movimiento de los buses alrededor de la ruta. Se observa cómo inicia saliendo el bus número 1, y consecutivamente el resto de buses hasta que el décimo sale. Los buses irán recorriendo la ruta establecida, esto sin pasarse uno al otro. Mientras se van moviendo los buses, a la derecha saldrá un historial, donde podremos seleccionar alguno de los 10 buses, y observar tanto por qué parada van, como a que hora pasaron por cada parada. Una vez finalizada la hora de ruta del bus, se procede a parar los buses y podremos seguir observando la información recolectada de cada uno de los buses.

Estrategia de solución del proyecto

Lo primero que hemos decidido hacer para realizar la solución y ejecución de este proyecto, ha sido revisar lo solicitado por el profesor, para así empezar a plantear posibles soluciones para la realización del programa. Una vez interpretado lo que se nos solicita, veremos qué herramientas necesitamos para la solución programada, ya que hay ciertas bibliotecas o nuevas herramientas de las que no tengamos tanto conocimiento, en este caso, serían los hilos, ya que es un tema nuevo para nosotros. Procederemos a realizar un diagrama de flujo, para así comprender de mejor manera cómo realizamos el proyecto y así a la hora de comenzar a programarlo, tener las ideas más claras. Como parte de lo que haremos previo a iniciar a programar, será establecer las variables, clases, métodos y paquete a utilizar. Esto nos ayudará no solo para poder trabajarlo de manera más ordenada, sino que al trabajar en equipo y de manera

separada, todos podrán crear las clases y métodos que les toquen, con los mismos nombres de variables ya establecidos. Una vez establecido el punto anterior, y que todos estemos de acuerdo, empezaremos realizando la interfaz de usuario, ya que en un proyecto anterior procedimos a realizar primero la interfaz de usuario y luego hacer la parte del backend, y nos pareció muy eficiente este orden a la hora de programar. También muy importante, realizaremos el proyecto utilizando GIT, ya que esto nos facilitará el trabajo en equipo, ya que todos´ podrán ir subiendo ahí sus avances de lo que vayan trabajando. Procederemos a dividir y distribuir las diferentes partes del código. Estableceremos fechas tanto de entrega como de revisión, donde cada integrante del grupo deberá de presentar sus avances al resto los integrantes, esto para estar al tanto de que todos están realizando sus partes como se esperaba y de igual manera para saber si alguien requiere ayuda en realizar el trabajo asignado.

Manual Tecnico

El “Manual Técnico” está enfocado en los desarrolladores y personas que sean parte del desarrollo de una aplicación. Este manual tiene como función brindar información sobre el diseño, arquitectura y funcionamiento interno del software. La idea de este manual, es beneficiar a los desarrolladores a entender como funciona lo que está siendo creado, así a un futuro, se agilizaran los procesos de mantenimiento y solución de problemas. A continuación el manual técnico de nuestro proyecto.Hablaremos de unas cuantas librerías que no se explicaron durante clase, pero que utilizamos para la creación de nuestro proyecto:

- La clase “Interfaz Cliente” lo que hace es crear la interfaz de cliente. Al inicio de la clase hay un constructor que sirve para mostrar los datos que se necesitan cuando inicia el programa. En la misma clase, se agrega la vista del panel, se actualiza la representación visual del panel y se revalida el panel para que se muestran los cambios. El método “borrar todo” borra todo lo que hay en el panel. En el método “btnAceptarActionPerform” se verifica cual bus está seleccionado, y cambia las paradas que están en el panel, por las del bus que esté en ese momento.

- En el “main cliente” se inicia el cliente y se abre la interfaz.

- En la clase “Paradas” lo primero que hay son unos arraylist que contiene todas las paradas que ha recorrido cada bus. Luego hay un método divide todos los mensajes que llegan con las paradas de cada bus y los distribuye al arraylist del bus correspondiente.

- En la clase “recibe” es donde ocurre la conexión con el servidor y recibe todos los mensajes de todas las clases de los buses para que salgan en la interfaz del cliente. Al final de la clase se imprimir fin del dia ya que todos los buses acaban su proceso

- En la clase “Cronometro_Frame” es donde se crea el cronómetro y sucede todo lo relacionado a este. De igual manera aquí ocurre el inicio y fin de los buses.

- La clase “interfaz” es la que contiene toda la parte de interfaz gráfica, relacionada al servidor.

- En la clase “main” se inicializa la interfaz gráfica y se inicia el servidor.

- En la clase “Servidor” se conecta el cliente y manda los mensajes con las paradas al cliente. También se inicia el cronómetro, se almacena el paquete del mensaje que se va a enviar. De igual forma esta el metodo para que inicien y terminen los buses

Tabla de Clases

Nombre	***Descripción***	***Accion que realiza***
bus1	Es la clase que mueve la imagen del autobús #1, es el hilo de este autobús	Se encarga de que la imagen siga la ruta del autobús.
bus2	Es la clase que mueve la imagen del autobús #2, es el hilo de este autobús	Se encarga de que la imagen siga la ruta del autobús.
bus3	Es la clase que mueve la imagen del autobús #3, es el hilo de este autobús	Se encarga de que la imagen siga la ruta del autobús.
bus4	Es la clase que mueve la imagen del autobús #4, es el hilo de este autobús	Se encarga de que la imagen siga la ruta del autobús.
bus5	Es la clase que mueve la imagen del autobús #5, es el hilo de este autobús	Se encarga de que la imagen siga la ruta del autobús.
bus6	Es la clase que mueve la imagen del autobús #6, es el hilo de este autobús	Se encarga de que la imagen siga la ruta del autobús.
bus7	Es la clase que mueve la imagen del autobús #7, es el hilo de este autobús	Se encarga de que la imagen siga la ruta del autobús.
bus8	Es la clase que mueve la imagen del autobús #8, es el hilo de este autobús	Se encarga de que la imagen siga la ruta del autobús.

bus9	Es la clase que mueve la imagen del autobús #9, es el hilo de este autobús	Se encarga de que la imagen siga la ruta del autobús.
bus10	Es la clase que mueve la imagen del autobús #10, es el hilo de este autobús	Se encarga de que la imagen siga la ruta del autobús.
Contrometro_Frame	Esta clase será la que contenga todas las fechas y los datos del reloj	Se encargará de llevar el control del reloj y de las fechas para saber cuando termina y empieza el día y saber qué día es.
InterfazServidor	Va a ser la clase que contenga al JFrame usado para la interfaz gráfica del servidor.	Genera el JFrame con todas las opciones que hay en el programa en la parte del servidor y también permite ver la ruta y los buses.
Main	Es la clase principal del programa.	Desde esta clase se inicia el programa.
InterfazCliente	Se encarga de mostrar los datos que se necesitan en el cliente.	Muestra los datos de los buses y permite elegir cual bus se va a poder ver.
Recibe	Recibe los datos que manda el servidor.	Convierte los datos para que se puedan leer y usar para mostrarlos en la interfaz del cliente.
Main_Cliente	Es el main del cliente	Desde esta clase se inicia el programa.
Paradas	Guarda las paradas a la que han llegado los buses	Añade las paradas al bus correspondiente

Tabla de Métodos

Nombre	***Descripción***	***Accion que realiza***
Run	Este método se repite por cada uno de los buses	Su función es que el bus se mueva a través de la ruta.
InterfazServidor	Constructor de la clase InterfazServidor	Se ejecuta al llamar a esta clase.
Métodos de Ayudante de JFrame	El ayudante usado para hacer la interfaz gráfica	Va a crear métodos por sí solo que van a tener las funciones usadas dentro de la parte gráfica.
Main	Método que se usa para ejecutar el programa	Método que se usa para ejecutar el programa
InterfazCliente	Constructor de la clase InterfazCliente	Se ejecuta al llamar a esta clase.
Métodos de Ayudante de JFrame	El ayudante usado para hacer la interfaz gráfica	Va a crear métodos por sí solo que van a tener las funciones usadas dentro de la parte gráfica.
Actualizar Tiempo	Hace que el reloj funcione	Cambia los valores del reloj
Run(clase recibe)	Es lo que corre el hilo, va a permitir recibir los datos del servidor	Se utilizará para mostrar en la parte gráfica los datos del bus elegido.
Actualizar Cronómetro	Muestra los valores del reloj en la interfaz	Muestra los valores del reloj en la interfaz
Terminal Buses	Termina la ejecución de lo buses	Termina la ejecución de lo buses
IniciarBuses	Inicia la ejecución de lo buses	Inicia la ejecución de lo buses

AñadirParada	Añade la parada a la lista de paradas del bus correspondiente	Añade la parada a la lista de paradas del bus correspondiente

Tabla de Librerías

NOTA: los imports de la parte gráfica no se usan porque el ayudante de JFrame genera el código de una manera en la que no se ocupen usar

Nombre	***Descripción***	***Accion que realiza***
import static java.lang.Thread.sleep	Permite pausar el hilo por un tiempo determinado	Se va a usar para tener una pausa entre los movimientos de los buses.
import javax.swing.ImageIcon	permite poner una imagen a un label	Se va a utilizar en los threads de los buses para asignarle una imagen a los labels de cada bus.
import java.io.DataOutputStream	A data output stream lets an application write primitive Java data types to an output stream in a portable way	Se usa en la conexión con el cliente.
import java.net.Socket	This class implements client sockets	A socket is an endpoint for communication between two machines. Se usa en la conexión con el cliente.
import java.io.DataInputStream	A data input stream lets an application read primitive Java data types from an underlying input stream in a machine-independent way	Se usa en la conexión con el servidor.

import java.net.ServerSocket	This class implements server sockets	Se usa en la conexión con el servidor.
import javax.swing.JLabel;	Es para crear los labels	Es para crear los labels
import javax.swing.JPanel;	Funciona para crear los paneles	Funciona para crear los paneles
import javax.swing.WindowConstants;	Funciona para crear el botón de minimizar y de salir	Funciona para crear el botón de minimizar y de salir
import java.text.ParseException;	Evitar errores en el programa	Evitar errores en el programa
import javax.swing.Timer;	Funciona para crear el timer	Funciona para crear el timer
import java.awt.event.ActionEvent;	Funciona para agregar un evento a objetos dentro de la interfaz	Funciona para agregar un evento a objetos dentro de la interfaz

Tabla De Variables

Nombre	***Descripción***	***Tipo de Dato***	***Accion que realiza***
activo	es la variable que permite saber si el bus está activo o no	boolean	permite llevar el control de la posición del bus
bus1	es el bus #1 del programa	JLabel	es el bus #1 del programa
bus2	es el bus #2 del programa	JLabel	es el bus #2 del programa

bus3	es el bus #3 del programa	JLabel	es el bus #3 del programa
bus4	es el bus #4 del programa	JLabel	es el bus #4 del programa
bus5	es el bus #5 del programa	JLabel	es el bus #5 del programa
bus6	es el bus #6 del programa	JLabel	es el bus #6 del programa
bus7	es el bus #7 del programa	JLabel	es el bus #7 del programa
bus8	es el bus #8 del programa	JLabel	es el bus #8 del programa
bus9	es el bus #9 del programa	JLabel	es el bus #9 del programa
bus10	es el bus #10 del programa	JLabel	es el bus #10 del programa
hilo1	es el objeto del hilo #1 que se va a usar para determinar si el bus está en movimiento o no	Autobús1	se usará para iniciar o pausar el bus
hilo2	es el objeto del hilo #2 que se va a usar para determinar si el bus está en movimiento o no	Autobús2	se usará para iniciar o pausar el bus
hilo3	es el objeto del hilo #3 que se va a usar para determinar si el bus está en movimiento o no	Autobús3	se usará para iniciar o pausar el bus

hilo4	es el objeto del hilo #4 que se va a usar para determinar si el bus está en movimiento o no	Autobús4	se usará para iniciar o pausar el bus
hilo5	es el objeto del hilo #5 que se va a usar para determinar si el bus está en movimiento o no	Autobús5	se usará para iniciar o pausar el bus
hilo6	es el objeto del hilo #6 que se va a usar para determinar si el bus está en movimiento o no	Autobús6	se usará para iniciar o pausar el bus
hilo7	es el objeto del hilo #7 que se va a usar para determinar si el bus está en movimiento o no	Autobús7	se usará para iniciar o pausar el bus
hilo8	es el objeto del hilo #8 que se va a usar para determinar si el bus está en movimiento o no	Autobús8	se usará para iniciar o pausar el bus
hilo9	es el objeto del hilo #9 que se va a usar para determinar si el bus está en movimiento o no	Autobús9	se usará para iniciar o pausar el bus
hilo10	es el objeto del hilo #10 que se va a usar para determinar si el bus	Autobús10	se usará para iniciar o pausar el bus

	está en movimiento o no		
ventana	es el objeto de la parte gráfica	InterfazServidor	permite llamar a la ventana en el main
dia	permite saber en qué día estamos cuando se está corriendo el programa	string	permite saber en qué día estamos cuando se está corriendo el programa
hora	permite saber qué hora es del dia	int	permite saber qué hora es del dia
minutos	permite saber cuántos minutos han transcurrido de la hora	int	permite saber cuántos minutos han transcurrido de la hora
segundos	permite saber cuántos segundos han transcurrido del minuto	int	permite saber cuántos segundos han transcurrido del minuto
socket	Se usa para almacenar el socket creado para la conexión con el cliente.	socket	Se usa para almacenar el socket creado para la conexión con el cliente.
mensaje	Se va a usar para almacenar los datos que se van a pasar al cliente.	string	Se va a usar para almacenar los datos que se van a pasar al cliente.

out	se usa para mandar el mensaje a través del socket	DataOutputStream	se usa para mandar el mensaje a través del socket
socket	Se usa para almacenar el socket creado para la conexión con el servidor.	ServerSocket	Se usa para almacenar el socket creado para la conexión con el servidor.
socket_servidor	Se usa para almacenar los datos del socket del servidor	Socket	Se usa para almacenar los datos del socket del servidor
In	se usa para recibir el mensaje que manda el servidor	DataInputStream	se usa para recibir el mensaje que manda el servidor
mensaje	se va a usar para almacenar el mensaje recibido	string	se va a usar para almacenar el mensaje recibido
Datos	Guarda los datos ordenadamente para poderlos imprimir luego correctamente.	ArrayList<String>	Guarda los datos ordenadamente para poderlos imprimir luego correctamente.
Autobús	Almacena el autobús que se quiere utilizar	String	Almacena el autobús que se quiere utilizar

Tabla	Tabla que se va a usar para mostrar los datos del bus elegido.	JTable	Tabla que se va a usar para mostrar los datos del bus elegido.

Pantallas del Sistema

En la primera imagen, se observa la interfaz del mapa de la aplicación, donde podemos ver como los buses van cada uno moviéndose dentro de la ruta por cada parada-

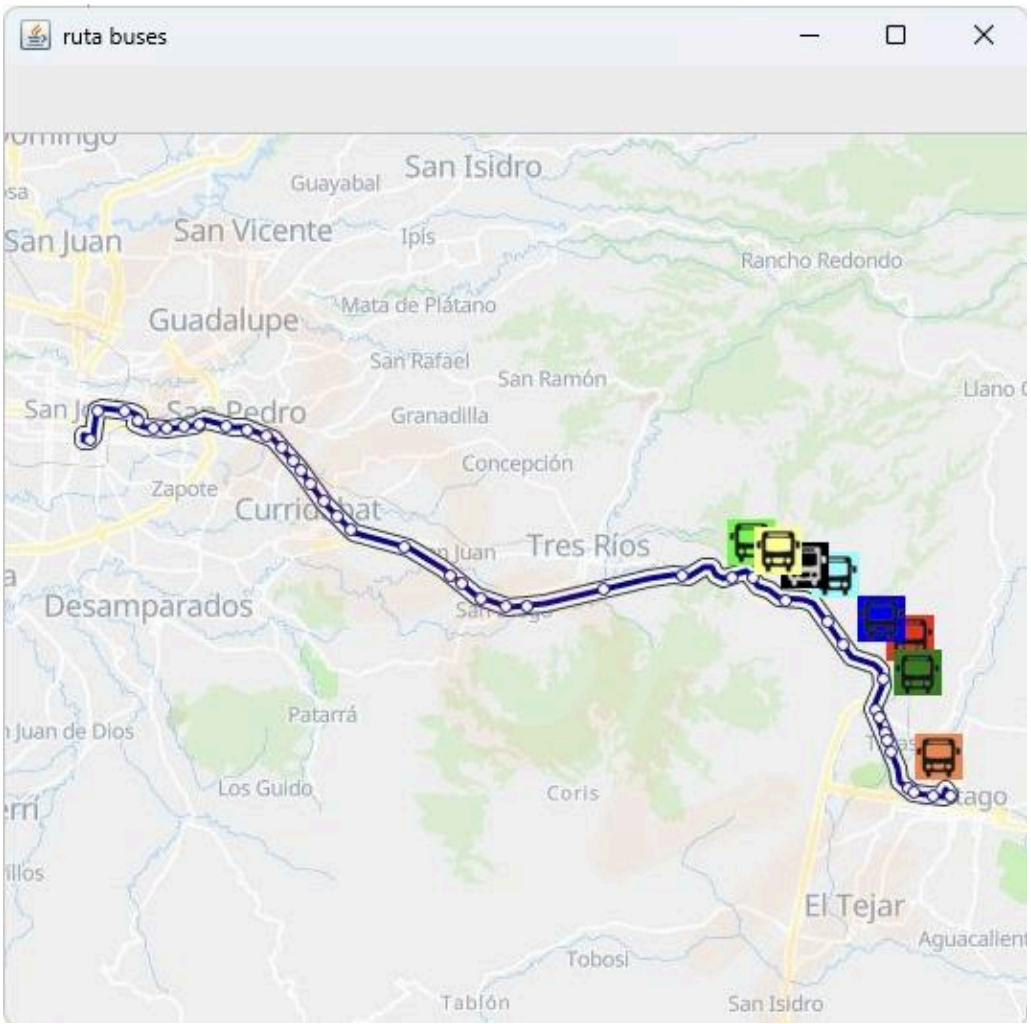

Imagen 1

En esta segunda imagen, se observa la interfaz del cliente, donde se van generando la hora por la que pasa por cada respectiva parada. Arriba a la derecha se puede cambiar con respecto al bus que el usuario desee observar.

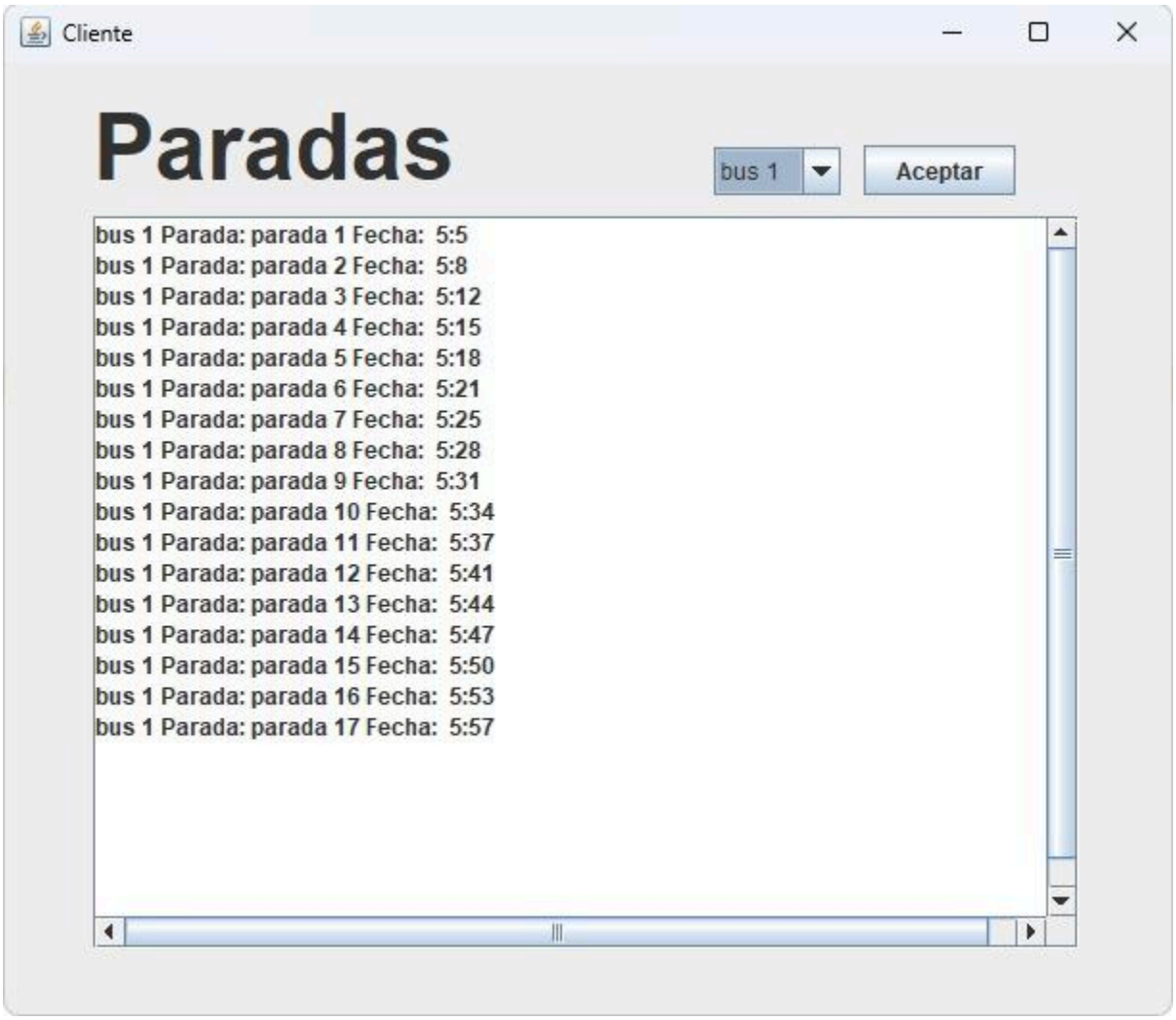

Imagen 2

En esta tercera imagen se encuentra la hora y fecha. De este reloj es de donde se van generando las distintas horas con respecto a los movimientos de los buses. Si se desea se puede adelantar la hora o adelantar un día.

Imagen 3

En la imagen número 4, se encuentra la misma interfaz del cliente que en la imagen número 2, solamente que aquí mostramos el mensaje de fin del día, ya que demuestra que el bus alcanzó su hora máxima de recorrido.

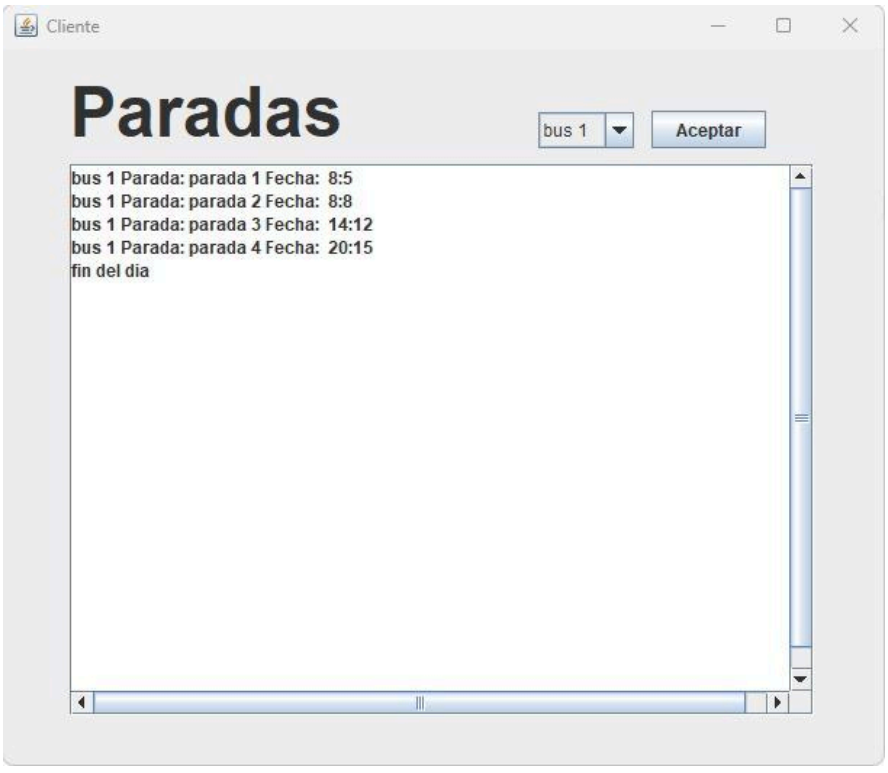

Imagen 4