Imagen en blanco y negro

Descripción generada automáticamente con confianza media

**Patrones Software – Sistema de peajes**

Integrante 1:

Víctor Sanavia Valdeolivas

03202543T

[victor.sanavia@edu.uah.es](mailto:victor.sanavia@edu.uah.es)

Integrante 2:

Adrián Rodríguez Hurtado

09064004A

[a.rodriguezh@edu.uah.es](mailto:a.rodriguezh@edu.uah.es)

ÍNDICE

[Enunciado y Requisitos 3](#_Toc185954333)

[Introducción 3](#_Toc185954334)

[Descripción de la aplicación 3](#_Toc185954335)

[Objetivos 3](#_Toc185954336)

[Características principales 4](#_Toc185954337)

[Requisitos funcionales y técnicos 4](#_Toc185954338)

[Innovaciones 5](#_Toc185954339)

[Manual de usuario 6](#_Toc185954340)

[Concurrencia 6](#_Toc185954341)

[Distribuida 8](#_Toc185954342)

[Diseño de la aplicación 10](#_Toc185954343)

[Patrones de creación 10](#_Toc185954344)

[Patrón Factory Method 10](#_Toc185954345)

[Patrón Singleton 10](#_Toc185954346)

[Patrones estructurales 11](#_Toc185954347)

[Patrón Adapter 11](#_Toc185954348)

[Patrón Facade 11](#_Toc185954349)

[Patrón Flyweight 11](#_Toc185954350)

[Patrones de comportamiento 13](#_Toc185954351)

[Patrón Iterator 13](#_Toc185954352)

[Patrón State 13](#_Toc185954353)

[Patrón Strategy 13](#_Toc185954354)

[Aplicación 15](#_Toc185954355)

[Clases 15](#_Toc185954356)

[Uso de Java 15](#_Toc185954357)

# Enunciado y Requisitos

## Introducción

Nuestro proyecto tiene como objetivo la implementación de un programa que simule el funcionamiento de un sistema de peaje de carreteras. A través de este software, se busca representar el flujo de vehículos que pasan por el peaje y la interacción con los empleados que trabajan en las cabinas. El sistema contará con diferentes tipos de vehículos y estaciones de peaje, permitiendo observar el proceso de cobro, la ocupación de cabinas y el comportamiento de los empleados. Además, tendrá una interfaz gráfica y la opción de un módulo de consulta remota.

## Descripción de la aplicación

La aplicación es una simulación en Java del sistema de peaje, en el que múltiples vehículos (coches, camiones y ambulancias) se dirigen hacia las cabinas para realizar el pago correspondiente. Los vehículos se distribuyen en cabinas específicas de acuerdo con su tipo y a las modalidades de pago, y son atendidos en orden de llegada. En particular, el sistema otorga prioridad a las ambulancias para que accedan de manera inmediata en situaciones de emergencia.

El comportamiento de los vehículos se registra en un archivo de log y se visualiza en la interfaz. Además, el sistema incluye un módulo de consulta remota que permite visualizar el estado del peaje en tiempo real desde una ubicación externa, así como habilitar o deshabilitar cabinas de pago con tarjeta.

## Objetivos

* **Desarrollar** una simulación funcional del sistema de peaje, que represente vehículos y empleados como hilos.
* **Implementar** una interfaz gráfica intuitiva que permita observar el estado de cada cabina y de cada vehículo en el sistema.
* **Crear** un módulo de consulta remota que permita supervisar el peaje desde un cliente externo.
* **Registrar** en un log el comportamiento de cada vehículo y empleado, facilitando el análisis del sistema.
* **Flexibilizar** el programa para la aplicación de distintos patrones software.

## Características principales

* **Simulación por hilos**: Cada vehículo y empleado se representa mediante un hilo, permitiendo una simulación realista de concurrencia.
* **Asignación de cabinas**: Los vehículos son dirigidos a cabinas específicas, con distinción de tipo de pago (manual o automático).
* **Registro de eventos**: Cada evento en el sistema se guarda en un log con el momento exacto de la acción, facilitando el análisis de comportamiento.
* **Interfaz gráfica**: Se cuenta con una interfaz que muestra el estado de cada cabina, la posición de los vehículos y el estado de cada empleado.
* **Módulo de consulta remota**: Permite la supervisión remota del sistema en tiempo real, actualizando cada segundo.

## Requisitos funcionales y técnicos

**Requisitos Funcionales:**

* **Simulación del flujo de vehículos** hacia el peaje, incluyendo la asignación y cobro de acuerdo con su tipo.
* **Prioridad para vehículos de emergencia**, como las ambulancias, que siempre avanzan en la fila.
* **Interfaz gráfica** para visualizar en tiempo real el estado del peaje, cabinas y vehículos en espera.
* **Log de eventos** para almacenar todos los movimientos y operaciones realizadas en el peaje.
* **Módulo de consulta remota** que permita observar el sistema desde otro dispositivo, con la capacidad de abrir o cerrar cabinas.

**Requisitos Técnicos:**

* **Programación en Java**, usando hilos para simular el comportamiento concurrente.
* **Uso de NetBeans** como entorno de desarrollo.
* **Sincronización de hilos** para evitar conflictos en el acceso a las cabinas y en la interacción con los empleados.
* **Mecanismos de comunicación distribuida** para el módulo de consulta remota.
* **Estructura de logs** en un archivo de texto, que registre cada evento con la fecha de la acción.

## Innovaciones

* **Generación dinámica de ambulancias**: Mediante un botón en la interfaz gráfica, el usuario puede simular emergencias, añadiendo ambulancias a la fila de vehículos de manera prioritaria.
* **Módulo de consulta en tiempo real**: La implementación de un sistema de consulta remota permite supervisar el sistema de manera externa, actualizando la información cada segundo.
* **Control de cabinas de manera remota**: A través del módulo de consulta, es posible abrir o cerrar cabinas de pago con tarjeta, ajustando el flujo de la simulación en tiempo real.

# Manual de usuario

El usuario tiene dos formas de ejecutar el programa. Una de ellas es mediante concurrencia, donde el usuario va a poder observar las distintas cabinas de coches y camiones.

## Concurrencia

Interfaz de usuario gráfica

Descripción generada automáticamente

Figura 1. Interfaz concurrente

Dentro de esta interfaz, el usuario puede parar el programa, congelando así a todos los vehículos que se encuentran dentro de las cabinas y los que están esperando a entrar. Además, existe la opción de poder generar una ambulancia, la cual se situará directamente la primera para entrar debido a la preferencia.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Figura 2. Cabina de coche

También existe la posibilidad de confirmar si un vehículo ha pasado el peaje o no, y comprobar la factura que ha pagado.

Interfaz de usuario gráfica

Descripción generada automáticamente con confianza baja

Figura 3. Coche que ha pasado el peaje

Forma

Descripción generada automáticamente con confianza media

Figura 4. Coche que todavía no ha pasado el peaje

## Distribuida

Interfaz de usuario gráfica

Descripción generada automáticamente

Figura 5. Interfaz distribuida

Dentro de esta interfaz, el usuario puede abrir o cerrar las cabinas automáticas. Si el usuario decide cerrar una cabina, el vehículo que se encuentre dentro de ella finalizará su ejecución y procederá a cerrar ésta, inutilizándola hasta que se vuelva a abrir.

Interfaz de usuario gráfica

Descripción generada automáticamente

Figura 6. Interfaz distribuida con las cabinas automáticas cerradas

Interfaz de usuario gráfica

Descripción generada automáticamente

Figura 7. Interfaz concurrente con las cabinas automáticas cerradas

# Diseño de la aplicación

## Patrones de creación

### Patrón Factory Method



### Patrón Singleton



## Patrones estructurales

### Patrón Adapter



### Patrón Facade



### Patrón Flyweight



## Patrones de comportamiento

### Patrón Iterator



### Patrón State



### Patrón Strategy



# Aplicación

## Clases

## Uso de Java