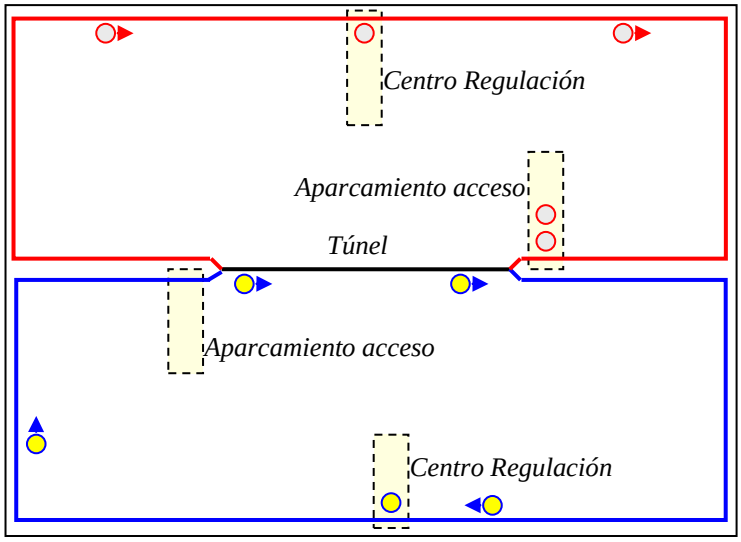
Proyecto RedFerroviaria

Esta aplicación consiste en el diseño del software de los trenes y de la infraestructura ferroviaria de una red de transporte tipo metro con las siguientes características.

La ***red ferroviaria*** se compone de un circuito con dos ***tramos ferroviarios*** independientes y cerrados que son recorridos por un conjunto de **trenes**. En cada tramo los trenes circulan en la misma dirección, y a diferentes velocidades. Los trenes rápidos deben reducir su velocidad si encuentran un tren más lento delante de él, ya que en una vía no se puede producir un adelantamiento. Sin embargo cuando varios trenes se encuentran aparcados en los andenes (de acceso al túnel o en el centro de regulación) siempre sale primero el que es más rápido.



Cada tramo ferroviario tiene un ***centro de regulación***, en el que los trenes se aparcan temporalmente a fin de regular sus tiempos de salida y con ello conseguir un tráfico uniforme. Para ello se permiten salir los trenes de los puntos de regulación con los intervalos de tiempo que corresponden a dividir el tiempo teórico de recorrido por el número de trenes. Denominamos tiempo teórico de recorrido al tiempo que tarda en recorrer el tren en recorrer su circuito sin paradas.

La red ferroviaria contiene un túnel que es compartido por ambos tramos ferroviarios. En el túnel sólo hay una vía que es utilizada por los trenes de ambos tramos ferroviarios. Los trenes de cada tramo ferroviario pasan por el túnel en sentido contrario, por lo que mientras que un tren de un tramo ferroviario esté atravesando el túnel, los trenes que lleguen del otro tramo ferroviario deben estacionarse en el acceso del túnel hasta que no haya trenes circulando en sentido contrario. Sin embargo dos trenes que circulen en el mismo sentido, si pueden circular simultáneamente por el túnel. A fin de hacer máximo uso del túnel (máximo throughput), se permite que los trenes que circulen en la misma dirección que el que están circulando por él, puedan acceder con independencia de que haya esperando trenes en el otro extremo del túnel.

La red ferroviaria representa la infraestructura de señalización, y dispone de un sistema de comunicación que mantiene los trenes comunicados en todo momento con ella. Los trenes tienen un sistema GPS y conocen en todo momento su posición. Asimismo, los trenes deben establecer periódicamente comunicación con la infraestructura,

comunicarle su posición, y recabar de ella su estado de marcha: esto es, si debe parar o a la velocidad a la que debe avanzar.

Objetivos

Los objetivos de la práctica son:

- La especificación, el análisis orientado a objetos de los patrones de interacción entre objetos, y del diseño de la aplicación para ser implementado en Java. La herramienta a utilizar para estas actividades es UML.
- El diseño e implementación de una interfaz grafica de usuario utilizando la familia de componentes SWT de Eclipse.
- La documentación que se genera debe ser la adecuada para que un programador que conozca el lenguaje de programación Java, pueda codificar cualquier clase sin que tenga que tomar decisiones de diseño.

Tareas:

Tarea 1: Realizar la especificación de usuario y detallada de la aplicación utilizando diagramas de caso de uso, y diagramas de contexto.

Tarea 2: Realizar el análisis de la aplicación, formulando el diagrama de clases de la aplicación:

- Identificar las clases que constituyen la aplicación, y de cada una de ellas identificar los atributos que constituyen su estado y los métodos que constituyen su interfaz. Establecer entre las clases las relaciones de herencia, agregación, dependencia y visibilidad que existen. Documentar todos los elementos que se incluyan (clases, atributos, métodos y relaciones).
- Describir mediante diagramas de actividad los métodos cuya lógica sea compleja.
- Describir mediante diagramas de estado las clases que tenga un comportamiento complejo.

Tarea 3: Mostrar los patrones de interacción básicos entre objetos. Documentar mediante diagramas de secuencias las interacciones que tengan una complejidad que lo requiera.

Tarea 4: Validar el análisis mediante la comprobación de que todos los casos de uso pueden ser implementados utilizando las clases propuestas.

Tarea 5: Realizar el diseño de la aplicación para que sea implementado utilizando Java:

- Mapear los tipos y los contenedores del análisis a tipos Java.
- Realizar el diseño de concurrencia de las aplicaciones y proponer el framework de de ejecución. Identificar las clases que son activas y los threads que requieren la ejecución.

Tarea 6: Diseñar la interfaz de usuario utilizando la familia de componentes SWT de Eclipse.

Tarea 7: Generar el esqueleto de código Java comprobando la completitud del diseño.

Tarea 8: Recopilar en un breve documento (20 páginas) la información útil para el programador que se ha generado en las anteriores tareas.