

---

# UML y red ferroviaria

**Programación concurrente y Distribuída**

**Curso 2011-12**



**Miguel Telleria, Laura Barros, J.M. Drake**

telleriam AT unican.es

**Computadores y Tiempo Real**

<http://www.ctr.unican.es>

---

# Objetivos

- Mostrar como aplicar UML a un problema concreto
  - Distinguiendo descripciones estáticas y dinámicas
  - Fijando el nivel de detalle
- Presentaros la aplicación que vamos a utilizar en muchas prácticas
  - Desde un enfoque funcional (lo **que** hace), no como lo hace
  - Dejar claro también lo que **no** hace.

# Contenido

- Requerimientos
- Diagramas estáticos
- Diagramas dinámicos
- Implementación secuencial

---

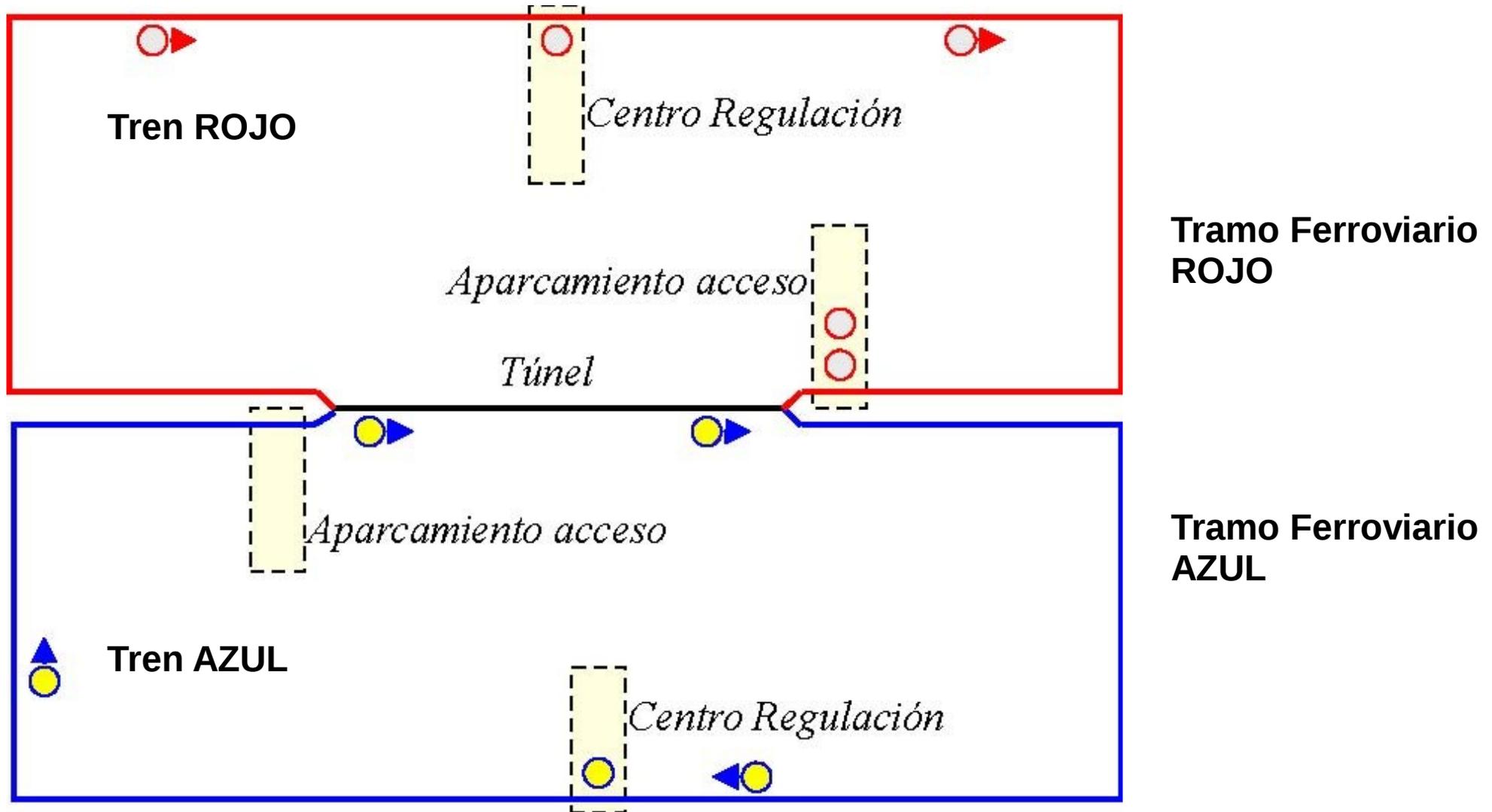
# Requerimientos

---

# Enunciado de la aplicación

- La red ferroviaria se compone de un circuito con 2 tramos ferroviarios independientes conectados por un túnel y que son recorridos por un conjunto de trenes a una velocidad propia.
- Los trenes y cada tramo van asociados a un color y sentido de circulación. En el túnel trenes de colores (y tramos) distintos circulan en sentidos opuestos.
- El túnel puede ser compartido por trenes del mismo color, pero no por trenes de color distinto (ya que habría colisiones).
- Los trenes cuando no circulan están aparcados en los centros de regulación (del tramo) o en los aparcamientos de acceso (del túnel).
- En las prácticas implementaremos diferentes políticas de salida de los centros de regulación y/o de los túneles.
- Los trenes obtienen de la red ferroviaria información de ocupación de la vía y siguen un recorrido en varias fases: centro de regulación, tramo, túnel y vuelta al centro.

# Una imagen vale más que 1000 palabras



# Vamos a verla funcionar

---

# Diagramas estáticos

---

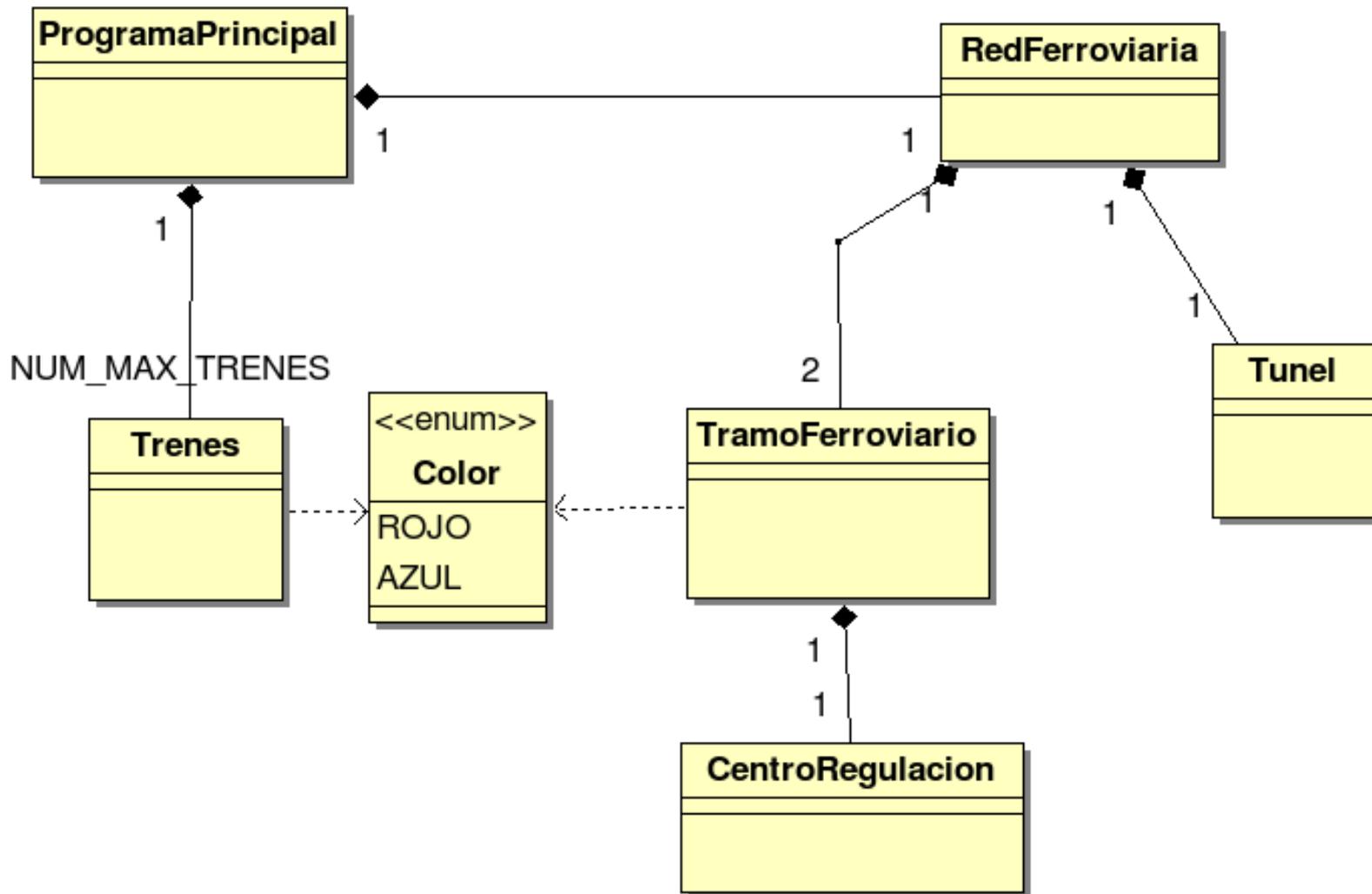
# Diagramas estáticos

- Describimos **la información que manejamos** y la relación entre los objetos.
- No describimos algoritmos o procedimientos.
  - Para esto hay otros diagramas
- Importante:
  - Donde va cada cosa
  - Asociaciones, enlaces entre clases
  - Cardinalides entre clases
  - Partes públicas y privadas
- Proceso sugerido:
  - Primero sacar las clases, sus composiciones y cardinalidades
  - Definir la “misión” de cada clase
  - Luego detallar atributos y operaciones necesarios para cada misión

# Identificación de las clases y atributos

- La **red ferroviaria** se compone de un circuito con 2 **tramos ferroviarios** independientes conectados por un **túnel** y que son recorridos por un conjunto de **trenes** a una velocidad propia.
- Los trenes y cada tramo van asociados a un **color** y sentido de circulación. En el túnel trenes de colores (y tramos) distintos circulan en sentidos opuestos.
- El túnel puede ser compartido por trenes del mismo color, pero no por trenes de color distinto (ya que habría colisiones).
- Los trenes cuando no circulan están aparcados en los centros de regulación (del tramo) o en los aparcamientos de acceso (del túnel).
- En las prácticas implementaremos diferentes políticas de salida de los **centros de regulación** y/o de los túneles.
- Los trenes obtienen de la red ferroviaria información de ocupación de la vía y siguen un recorrido en varias fases: centro de regulación, tramo, túnel y vuelta al centro.

# Composiciones



# Misiones

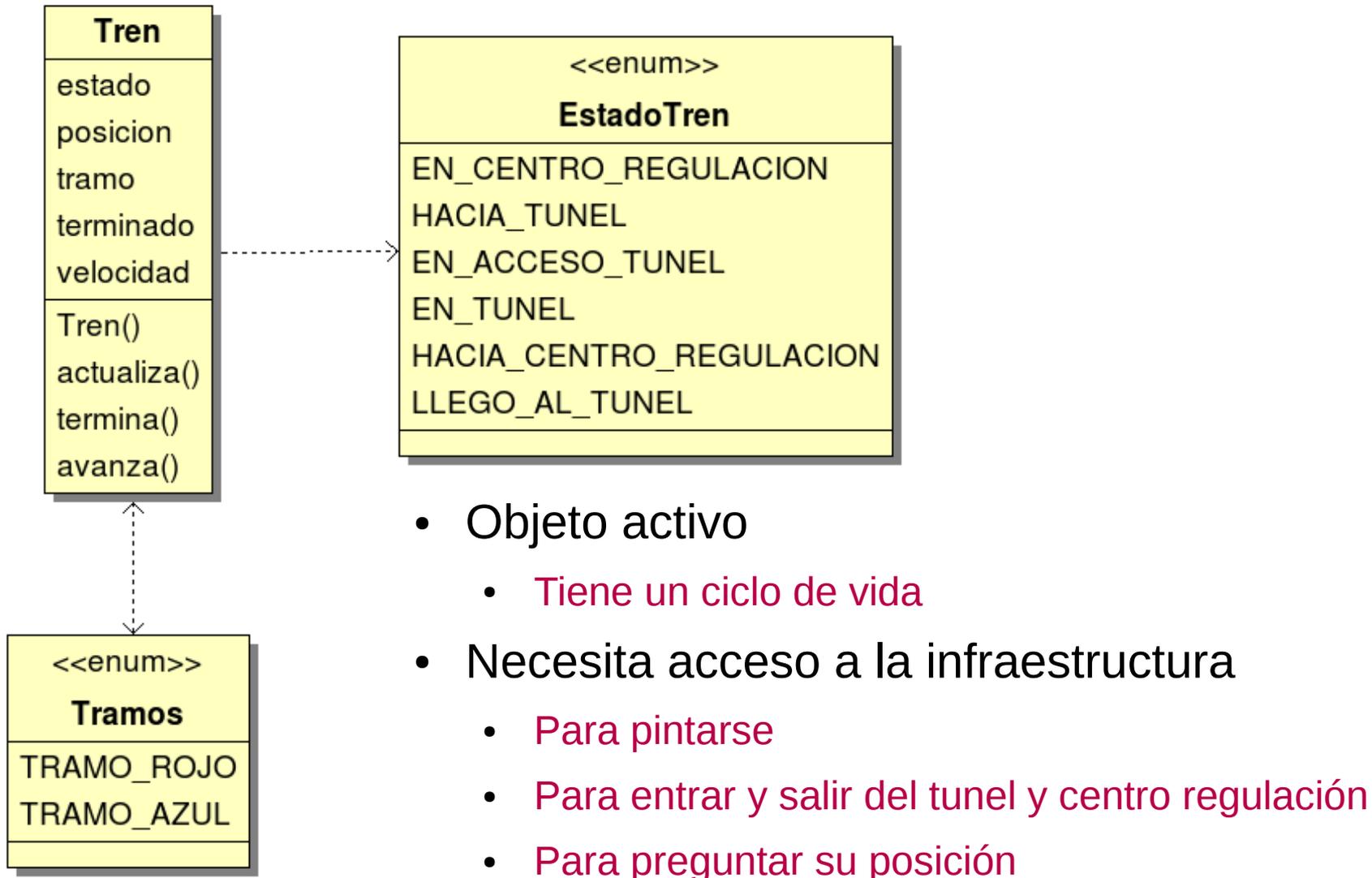
- Red ferroviaria
  - Datos globales
  - Punto de acceso para el resto de las clases
- Tren
  - Objetos activos
- TramoFerroviario
  - Ocupación de la vía
  - Contiene el centro de regulación
- Centro de regulación
  - Política de salidas
- Tunel
  - Exclusión mutua (respecto a tramos diferentes)
  - Política de compartición
- Programa principal
  - Instanciar el sistema

# Red Ferroviaria

- Maneja la GUI
- Da acceso al resto de elementos



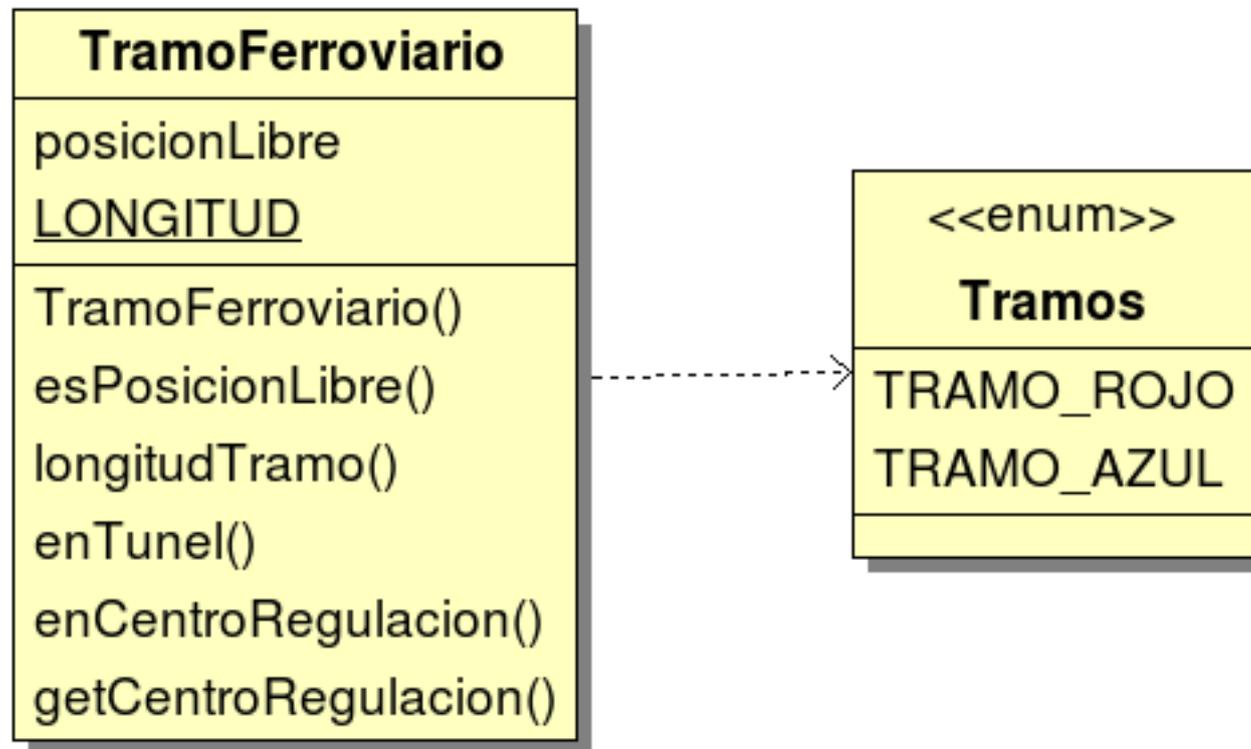
# Tren



- Objeto activo
  - Tiene un ciclo de vida
- Necesita acceso a la infraestructura
  - Para pintarse
  - Para entrar y salir del tunel y centro regulación
  - Para preguntar su posición

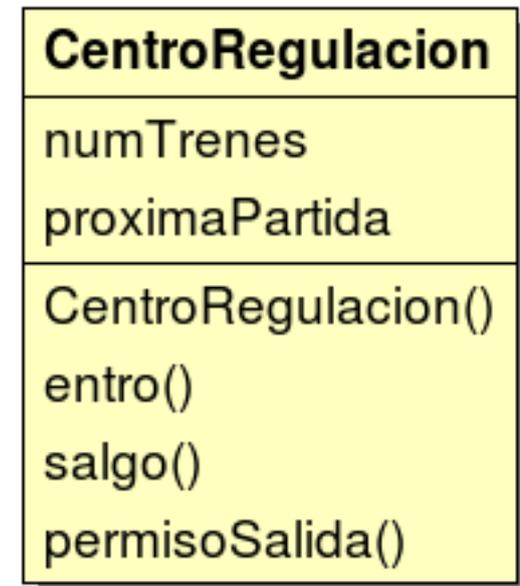
# TramoFerroviario

- Lleva la cuenta de las posiciones en la vía
- Da acceso al Centro de Regulación correspondiente



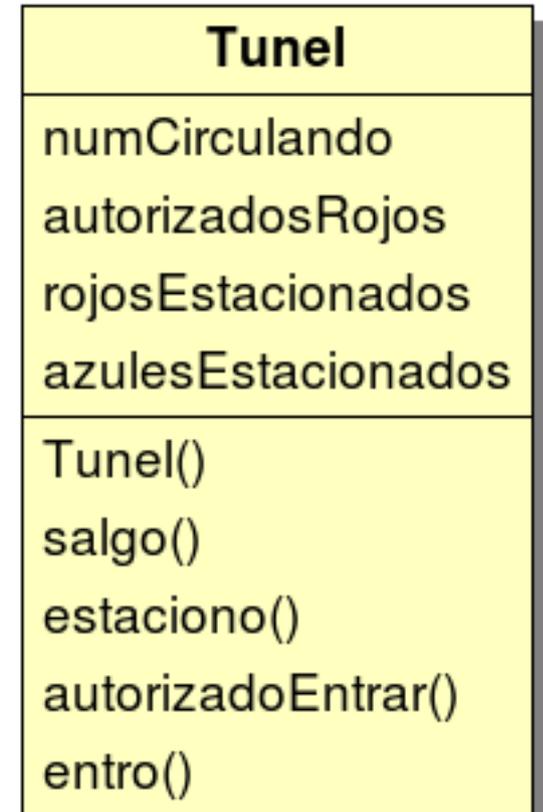
# CentroRegulacion

- Tiene la política de entrada y salida
  - Da el permiso de salida
- Lleva la cuenta de los trenes aparcados



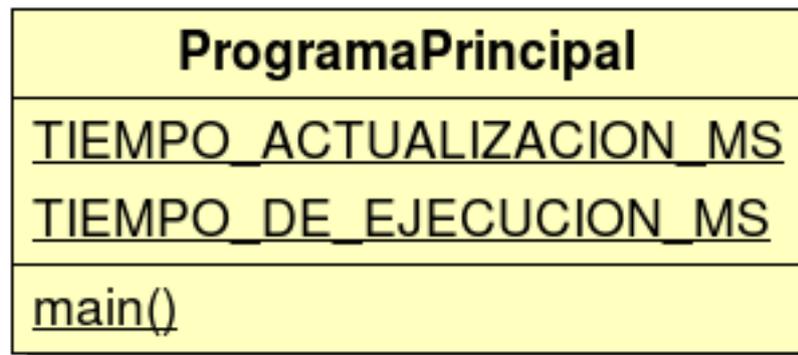
# Tunel

- Administra el acceso a un color de los trenes
- Lleva la cuenta de los trenes estacionados

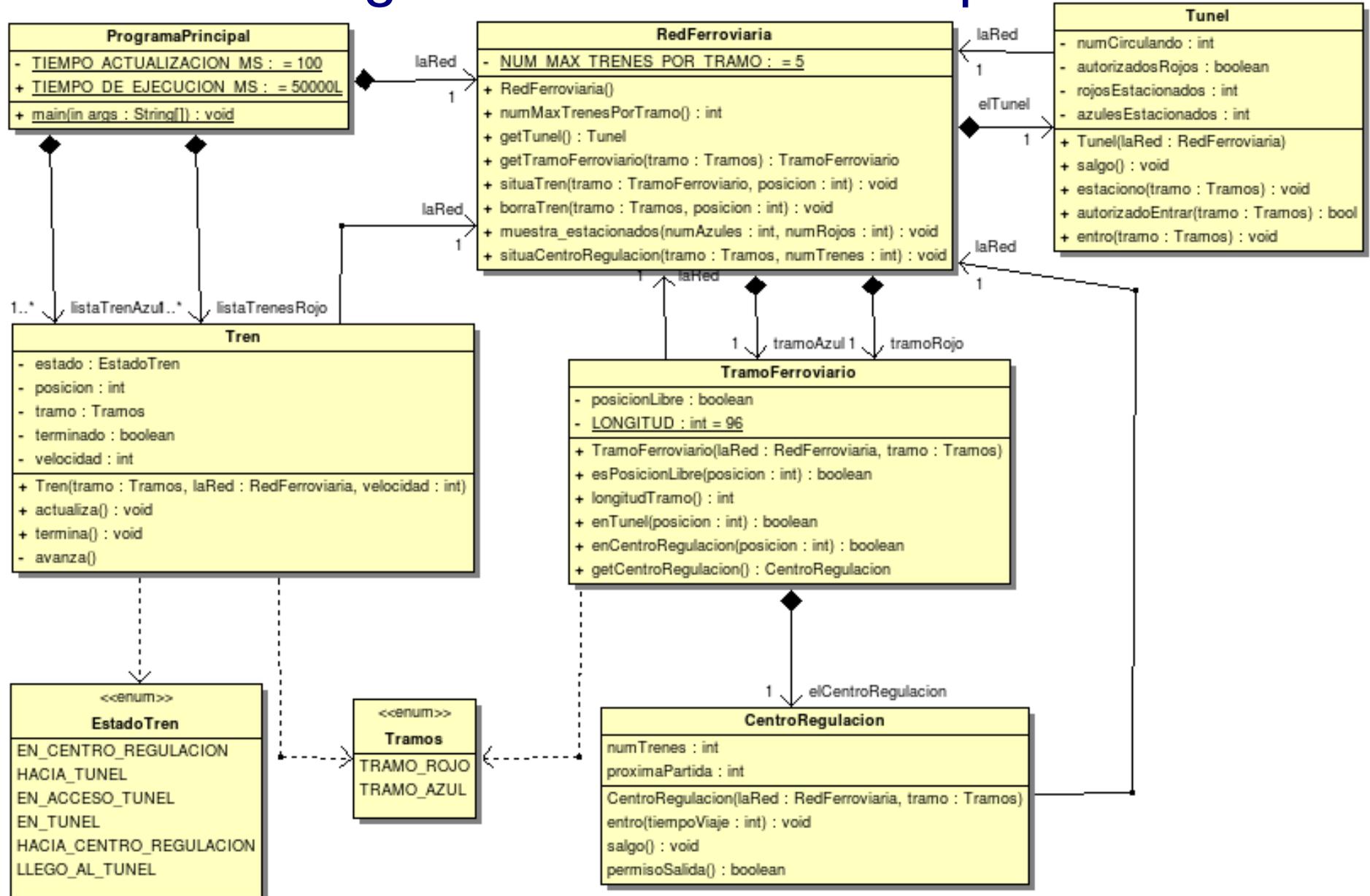


# Programa principal

- Instancia la RedFerroviaria
- Instancia los trenes
- Mantiene el programa vivo
- En la implementación secuencial actualiza periódicamente los trenes



# Diagrama de clases completo



---

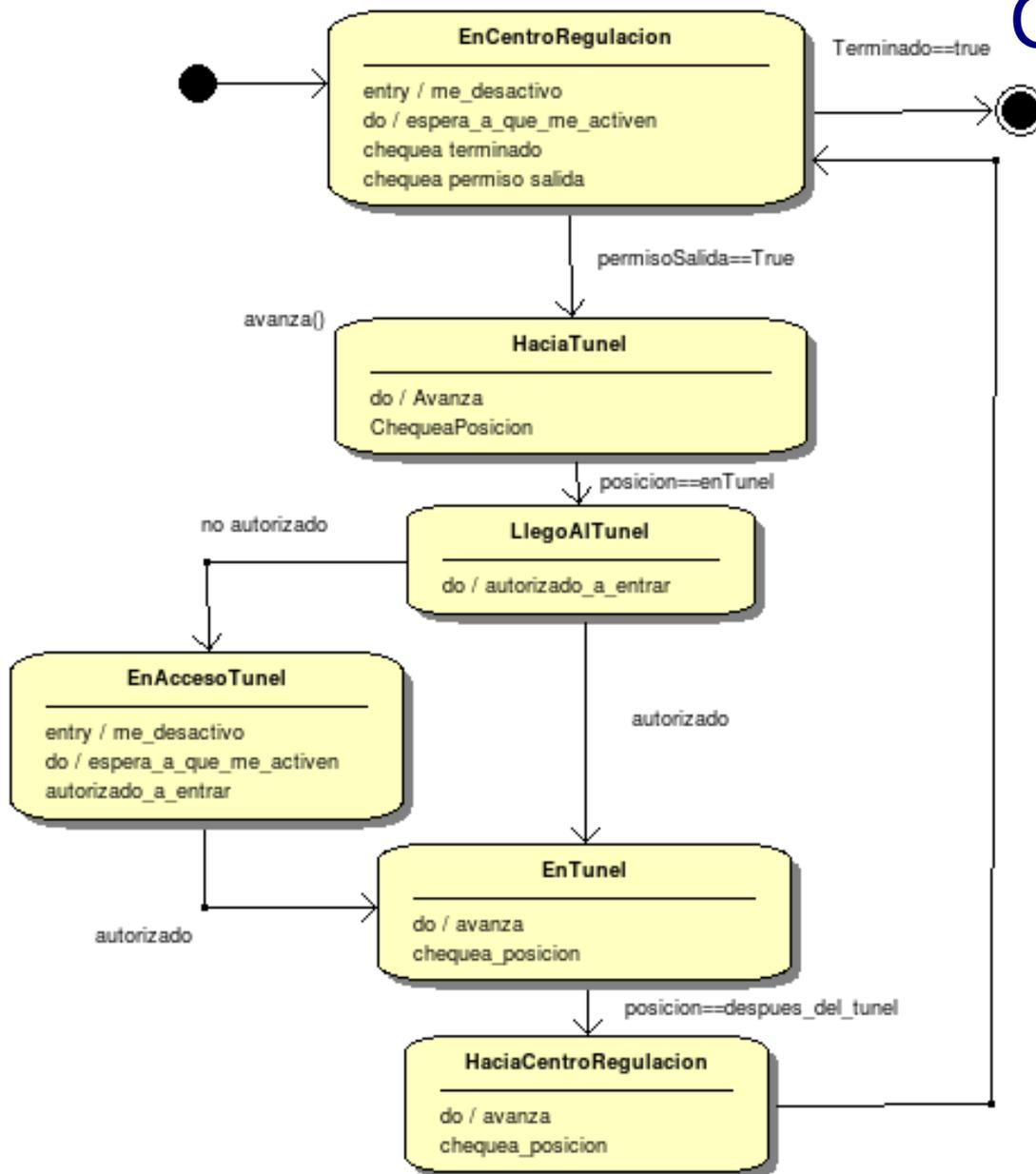
# Diagramas dinámicos

---

# Diagramas dinámicos

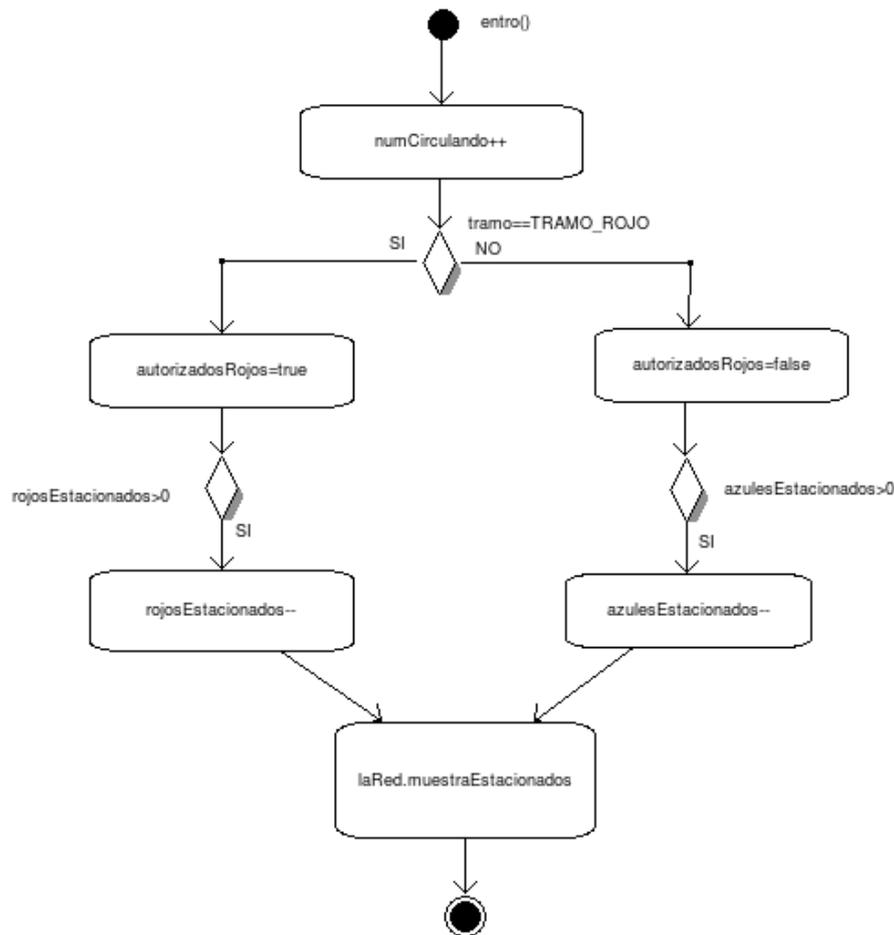
- Describen comportamientos, algoritmos, acciones.
- Se basan en la información reflejada en los diagramas estáticos.
- Importante
  - Orden de las acciones (secuencia, actividad)
  - Eventos y respuestas
  - Decisiones
- Ejemplos típicos
  - Ciclo de vida de un objeto activo
  - Colaboración de acceso concurrente a un objeto pasivo
  - Proceso de inicialización de una aplicación

# Ciclo de vida de un tren

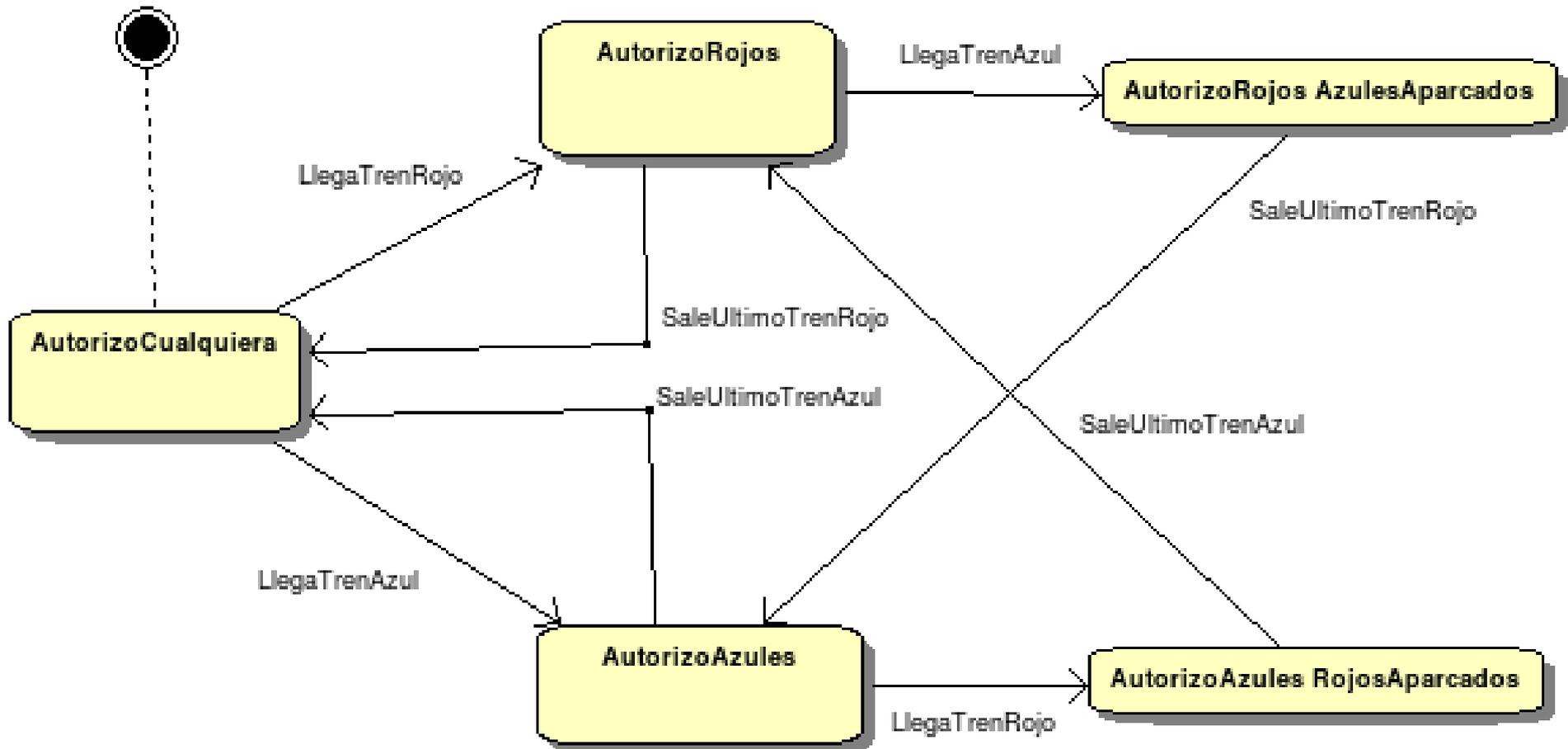


# Acceso concurrente entro() en el tunel

- En este diagrama no se especifica la autorización a entrar



# Proceso de decisión

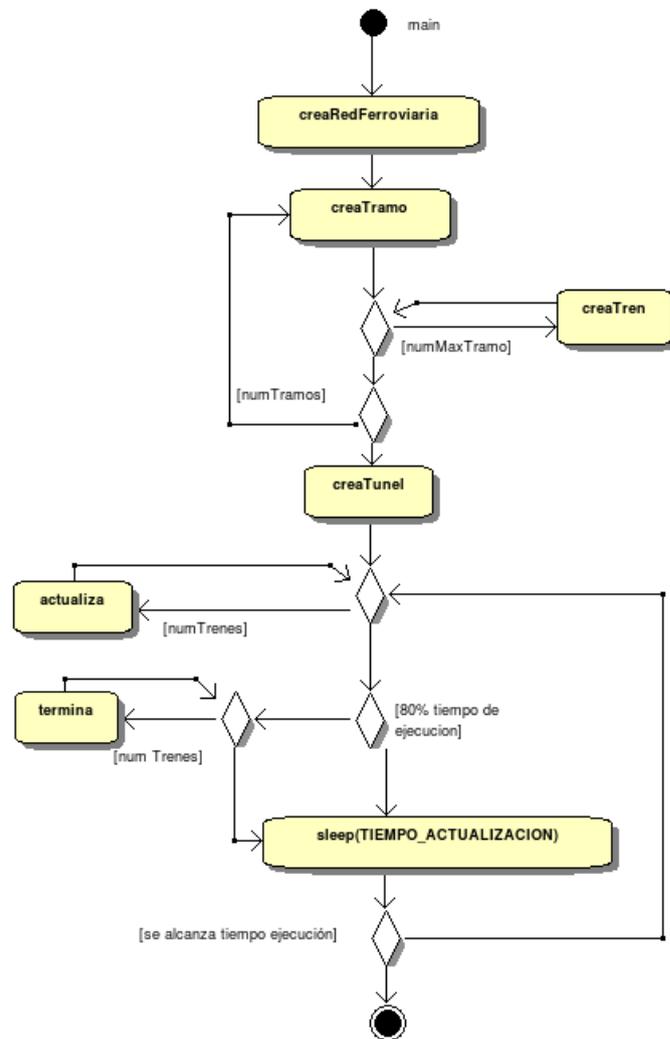


---

# Implementación secuencial

---

# Implementación secuencial



- Los trenes aquí son también objetos pasivos
- Responden al método **actualiza()** enviado por el programa principal