Universidad Nacional de Quilmes

Departamento de Ciencia y Tecnología

Informe

Sistema de estacionamiento medido

Alumnos:

Diaz, Ignacio

nacho.pecci23@gmail.com

Giamminelli, Nahuel

nahuelgiamminelli@gmail.com

Piana, Augusto

augustopiana3@gmail.com

Decisiones de Desarrollo

Modalidades de Estacionamiento

Para la modalidad **A** de inicio de estacionamiento decidimos crear una clase llamada “PuntoDeVenta” que es la encargada de registrar los estacionamiento de forma física, tomandole su patente y la cantidad de horas.Además es la encargada de cargarle saldos a los clientes.
En el momento que el punto de venta registra un estacionamiento y/o recarga saldos, también decidimos que sea el encargado de crear los “Ticket”, ya sean “TicketRecarga” o “TicketEstacionamiento”, asi se los envía al “SEM” con y este guarde registro de los mismos.

Para la modalidad **B** de inicio de estacionamiento, decidimos crear la clase “AppSEM” que es aquella con la que el usuario final interactúa. Sobre esta es donde se carga el saldo del cliente.

Mediante esta modalidad el usuario es quien inicia el estacionamiento, cuando la AppSEM reciba el mensaje de inicio esta la delega sobre su estado, quien es quien decide que hacer sobre esta situacion.
Si la AppSEM se encuentra en estado NoVigente, luego de chequear que tenga saldo mínimo, le envia un mensaje a su SEM iniciando el estacionamiento y le indica a la AppSEM que su nuevo estado sera Vigente.
Si la AppSEM se encuentra en estado Vigente, no realiza ninguna acción.

Para poder realizar el fin de estacionamiento mediante la AppSEM, decidimos también delegar este mensaje sobre su estados, quien es el que decide qué hacer.
Si la AppSEM se encuentra en estado NoVigente, no realiza ninguna accion.
Si la AppSEM se encuentra en estado Vigente, le envia un mensaje a su SEM finalizando el estacionamiento y le indica a la AppSEM que su nuevo estado sera NoVigente.

En este momento tomamos la decisión de crear un clase abstracta llamada EstadoEstacionamiento, superclase de ambos estados. En esta implementamos todos los mensajes, dejándolos vacíos para que sus subclases no deban implementar la totalidad de ellos.

Movement Sensor:

Decidimos que la AppSEM implemente esta interfaz, asumiendo que de alguna manera gracias a los sensores del teléfono reciba cada cierto tiempo los mensajes de driving/walking . De esta manera, mediante una propiedad de la clase, esta puede gestionar estos cambios de movimiento para así notificar los posibles inicios/fines de estacionamiento y así delegar a su estado actual, mediante el cual podemos confirmar si hay efectivamente un posible inicio/fin

Estacionamientos:

Para los estacionamientos utilizamos una clase abstracta Estacionado, con el objetivo de que el SEM pueda almacenar de manera indiferente todos los tipos de estacionamientos que se realizaron y gestionarlos, sin importar desde donde fueron iniciados. Luego para los estacionamientos iniciados por un punto de venta creamos la subclase EstacionadoPV y para los iniciados por app EstacionadoAPP. Esto lo desarrollamos de esta manera ya que la información almacenada y requerida para cada tipo de estacionamiento generado distinto.

Modo Manual y Automático :

En cuanto a los modos de estacionamiento, nos dimos cuenta que, de lo que realmente se deben encargar, es de gestionar la manera en la que se interactúa con las notificaciones recibidas gracias a movement sensor, ya que una vez que el estado verifica que efectivamente nos encontramos ante un inicio/fin de estacionamiento, este delega al modo la manera en la que actuamos, que pueden ser notificando al usuario o directamente iniciando el estacionamiento.
Algo a destacar respecto al modo automático es, que si bien este iniciará/finalizará los estacionamientos por sí solo, el usuario sigue teniendo la posibilidad de gestionar el estacionamiento por su cuenta.

Patrones de diseño

Los patrones utilizados para el desarrollo de este trabajo son:

State:

Para representar en qué estado se encuentra un estacionamiento de la app de usuario.
Esto con el objetivo de evitar que un usuario puede iniciar dos veces un mismo estacionamiento o que pueda iniciar a la vez dos estacionamientos distintos.
Además de que nos ayuda a gestionar las notificaciones de posible inicio/fin de los estacionamientos.

Los roles de este patrón son:

- Context: La AppSEM

- State: EsadoEstacionamiento
- Concrete State: NoVigente / Vigente

Strategy:

Para los modos de estacionamiento, los cuales los usamos principalmente para gestionar los posibles inicios/fin de estacionamientos, en el momento en que los state envian el mensaje de posible inicio/fin a los strategy, estos se encargan de gestionar. En el caso de manual, le indica a la app que envíe una notificación al teléfono y en el caso del modo automático le indica a la app que inicie el estacionamiento.

Los roles son:

- Context: AppSEM
- Strategy: ModoEstacionamiento
- Concrete Strategy: Manual/Automatico

Observer:

Lo implementamos para que el SEM puede notificar a otras entidades del inicio/fin de los estacionamientos.

Los roles son:
- Subject: SEM
- Observer: SistemaMonitoreo