Informe Trabajo Práctico Final: P002

Integrantes:
Giuffra, Mateo
(matteogiuffrah40@gmail.com)
Picchio, Bianca
(biancapicchio04@gmail.com)

Profesores:

- Butti, Matias
- Cano, Diego
- Torres, Diego

DECISIONES DE DISEÑO:

ENUM:
Ya que no vimos posibilidad de cambios en los siguientes estados y estrategias decidimos hacerlos de tipo ENUM aparte de aportar mas legibilidad al codigo. Los tipos ENUM creados a lo largo del tp son:
EstadoEstacionamiento, EstrategiaModo.
FranjaHoraria:
Discutimos modelar la Franja horaria pero sentiamos que no valia la pena porque sus mensajes invocados por el SEM terminaban delegando al mismo, entonces preferimos poner una condicion que verifica si la hora del SEM estaba dentro de la franja horaria o no.

PATRONES DE DISEÑO:

OBSERVER: Usamos este patrón en el SEM(ConcretSuject) y la Interface Entidad(AbstractObserver), ya que el SEM posee una lista de entidades a las cuales les avisa determinados eventos del SEM. Estas entidades pueden suscribirse o desuscribirse del aviso.
OBSERVER: La AppUsuario(ConcretSuject) implementa la interfaz externa MovementSensor(AbstractObserver), la cual le avisa cada cierta cantidad de segundos cuando está Caminando o Manejando.

ADAPTER: Entidad(Adapter) esta pensada como una interfaz ya que futuras entidades o sistemas que no conocemos pueden implementarla y ser alertados de los distintos eventos del SEM.
STRATEGY: La AppUsuario(Context) puede decidir que tipo de modo quiere utilizar, el Automatico o el Manual (ConcretStrategies) segun sus preferencias ya que cada modo actúa de forma diferente.
STATE: AppUsuario(Context) posee un estado llamado EstadoDesplazamiento, el cual es el encargado de avisarle al usuario de un posible inicio o fin de estacionamiento, dependiendo de si está o no está estacionado o si no está en alguna zona de estacionamiento. Este estado cambia constantemente a EstadoAPie o EstadoManejando(concretStates) gracias a un interfaz externa llamada SensorMovimiento.
STATE: AppUsuario(context) posee un EstadoEstacionamiento, que dependiendo si quiere inicializar o finalizar alguno o si está caminando o manejando, cada estado hace una cosa distinta. Los concret states en este patrón son: NoEstaEnZona, EstaEstacionado, NoEstaEstacionado. Estos se conocen y settean entre si dependiendo que mensaje reciben.

CORRECCIONES:

- Cambiar los enums por clases o interfaces.

Actualizacion:

- Decidimos poner el Strategy (EstrategiaModo) como Interface ya que posee methods abstractos y brinda mayor flexibilidad al codigo para futuros modos.
- El State ahora es una clase abstracta llamada EstadoEstacionamiento, ya que posee algunos hook methods que de ser necesario se pueden sobreescribir en las subclases.

- Cambiar el altaInfraccion() del inspector para delegarsela al SEM.

Actualizacion:

- Se creo en el Sem el method altaInfraccion(), donde se pregunta si el auto estacionado (tarea del Sem), se crea la infraccion y se agrega a su lista de infracciones (tarea del Sem).