Programación con Objetos 2

Trabajo Final

Sistema de Estacionamiento Medido

- Federico Hernan Sanchez - FedericoSanchezUNQ@gmail.com

- Juan Cruz Sanchez –juancruz.unq@gmail.com

- Matias Adrian Dominguez - dmzmatias32@gmail.com

Patrones de diseño

Observer -

El sem puede desuscribir y suscribir usuarios, para poder notificarle sobre los eventos que están sucediendo . Los usuarios reciben la actualización de un estacionamiento iniciado, finalizado o una recarga de crédito.

Se implementó la clase Reloj para que al momento de ser requerido, se puedan crear los registros que requieran la hora y/o la fecha. Además se le atribuye una lista de observers que son notificados del cambio de hora del Reloj, entre los posibles usos esta la finalización de la jornada del Sem a las 20:00 hs, y la finalización de los estacionamientos vigentes en ZonaDeEstacionamiento que cumplieron con su horario de fin.

State -

Se utilizó el patrón de State en la superclase ModoMovimiento, la cual es extendida por Driving y Walking, ya que estos estados responden a su manera

a los cambios de movimiento, tanto para iniciar como para finalizar un posible estacionamiento.

Strategy -

El patrón Strategy fue implementado en los modos de la AppCliente, pudiendo ser Manual o Automatico, ambos implementan la interfaz de ModoApp. De esta forma si se implementa un nuevo modo de app, solamente debe implementar la interfaz y responder a los mensajes como sea requerido.

Decisiones De Diseño

Como deben mantenerse diversos registros en el Sem, se crearon clases para que cada tipo de registro pueda guardar los datos requeridos, así como también tener subclases como REstacionamiento tiene de subclase a REstacionamientoPuntual y a REstacionamientoApp.

Para mantenerse al corriente de los estacionamientos vigentes, cada zona de estacionamiento tiene un atributo estacionamientosVigentes con una lista de REstacionamientos, para que de esta forma el

Inspector asignado a la zona pueda verificar si los vehículos se encuentran vigentes.

Del lado del usuario-cliente se implementó la clase Celular en donde deriva las acciones de mensajes respecto al estacionamiento a la clase AppCliente. Dentro de la clase Celular se instala la App correspondiente.

La clase Inspector posee una zonaDeEstacionamiento a la que se le asigna y un Celular con una App instalada a modo de admin. Tiene como variable un reloj para conocer la hora para realizar las infracciones . Esta clase realiza las tareas mandadas por el Inspector, por ejemplo consultar si un estacionamiento es vigente , o darle una infraccion al vehiculo indicado.

La clase conductor tiene asignado un vehiculo y un celular que es con el que va a poder realizar las todas las acciones para el estacionamiento o recarga de credito

Detalles De Implementación

Asumiendo que se actualiza de forma constante, se agregó un atributo de gps a Celular que guarda la ZonaDeEstacionamiento en la cual se encuentra, y si no se encontrara en ninguna zona, guardaría el valor null.

Los metodos driving y walking dados por el gobierno , alertan sobre el cambio de movimiento. Al detectar este cambio se dispara automáticamente los eventos de inico y fin estacionamiento. Para eso se agregaron los atributos necesarios para que puedan enviar los datos necesarios para completar este evento.