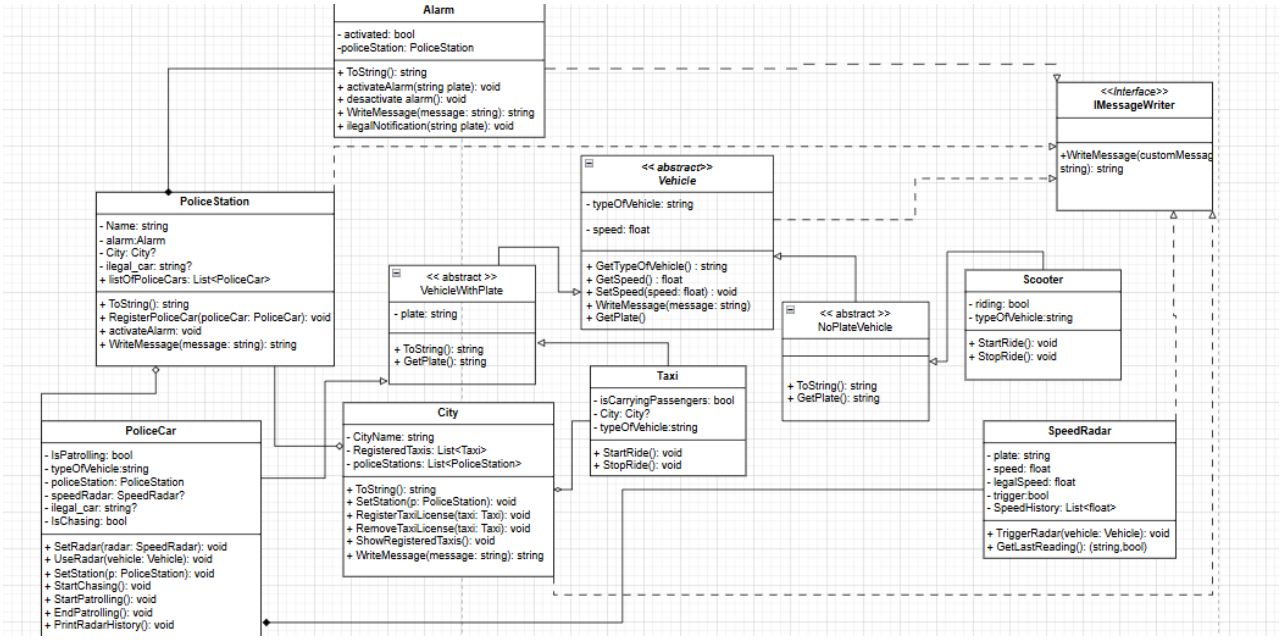
Práctica 2 Paradigmas de la Programación

Bernardo Gómez Carrasco



En la entrega de la práctica dejo adjunto el archivo donde realicé el UML para una mejor visión.

Pregunta 2.6

Explicar brevemente si se están aplicando los principios SOLID dentro del código y el porqué de esa implementación, hacer las modificaciones necesarias en el código y en la arquitectura para que se cumplan los principios.

- En mi opinión creo que mi código cumple con los principios SOLID. En primer lugar, he desglosado el código de la mejor manera posible con el fin de conseguir que cada clase tenga una única responsabilidad o propósito claro. De esta forma se evitar que una clase asuma múltiples tareas que no están relacionadas entre sí (cumpliendo el principio S/SRP).

- Las clases están abiertas para extensión, pero cerradas para modificación. Es decir, se pueden agregar nuevas funcionalidades a una clase sin modificar su código existente como por ejemplo se pueden aportar más vehículos sin tener que modificar el código de la clase padre (cumpliendo con O/OCP).

- Por otro lado, las clases derivadas como “PoliceCar” y “Taxi” respetan el comportamiento de su clase padre “Vehicle” y no realizan comportamientos inesperados (cumpliendo con L/Liskov),

- Además, Las clases no implementan interfaces que no utilizan. Disponemos de una única interfaz “IMessageWritte” dedicada exclusivamente a crear mensajes. Y siempre que las clases implementan la interfaz hacen uso de ella (cumpliendo I/ISP).

- El código también utiliza abstracciones como con la clase Vehicle para que de esta manera los módulos de alto nivel como son la clase padre no tengan que depender de los módulos de bajo nivel (cumpliendo la D/DIP).

Por lo tanto, considero que mi código cumple con las propiedades SOLID

Pregunta 2.7

Ahora queremos que el policía pueda tener diferentes aparatos de medida, que pueden ser un medidor de velocidad (Radar) o un medidos de alcohol (Alcoholímetro). Al coche de policía solo se le puede asignar un único medidor, y en el coche de policía únicamente va a haber un método para activar el aparato de medida. Con la arquitectura actual, ¿Qué principio SOLID incumpliríamos y cómo lo solucionarías?

Con esta implementación para añadir un posible alcoholímetro a los coches de policía deberíamos modificar la clase PoliceCar para adaptarnos al nuevo modelo. Pues en la situación actual el coche de policía solo puede recibir existir sin ningún dispositivo o con un dispositivo radar.

Por tanto, el principio SOLID que se incumpliría en este caso sería el de O (Open/Close Principle), el cual establece que cualquier clase debe estar abierta para su extensión, pero cerrada para su modificación.

Para resolver esto, es necesario permitir que los dispositivos (como el radar o el alcoholímetro) puedan extenderse sin necesidad de modificar la clase PoliceCar.

Una solución posible sería la de crear una interfaz para los dispositivos que los coches de policía pueden equipar (medidor). Posteriormente hacer que la clase Radar y Alcoholímetro utilicen esa interfaz para realizar sus respectivas mediciones. Es decir que la interfaz contenga un método llamado “useDevice” para que tanto el radar como el alcoholímetro lo usen a su manera.

De este modo se le permitiría a la clase PoliceCar usar dispositivos a través de la interfaz de dispositivos.