FACULTAD DE INGENIERIA
Universidad de Buenos Aires

Trabajo Práctico 2 — Java

[7507/9502] Algoritmos y Programación III

Curso 2

Primer Cuatrimestre de 2022

Nombre	Padrón	Mail
Lucas Waisten	101908	lwaisten@fi.uba.ar
Camila Fiorotto	108239	cfiorotto@fi.uba.ar
Juan Martin Munoz	106699	jmmunoz@fi.uba.ar
Ignacio Regazzoli	105167	iregazzoli@fi.uba.ar
Daniel Tomas Kim	107188	dakim@fi.ubar.ar

Tutor: JOSE PABLO SUAREZ

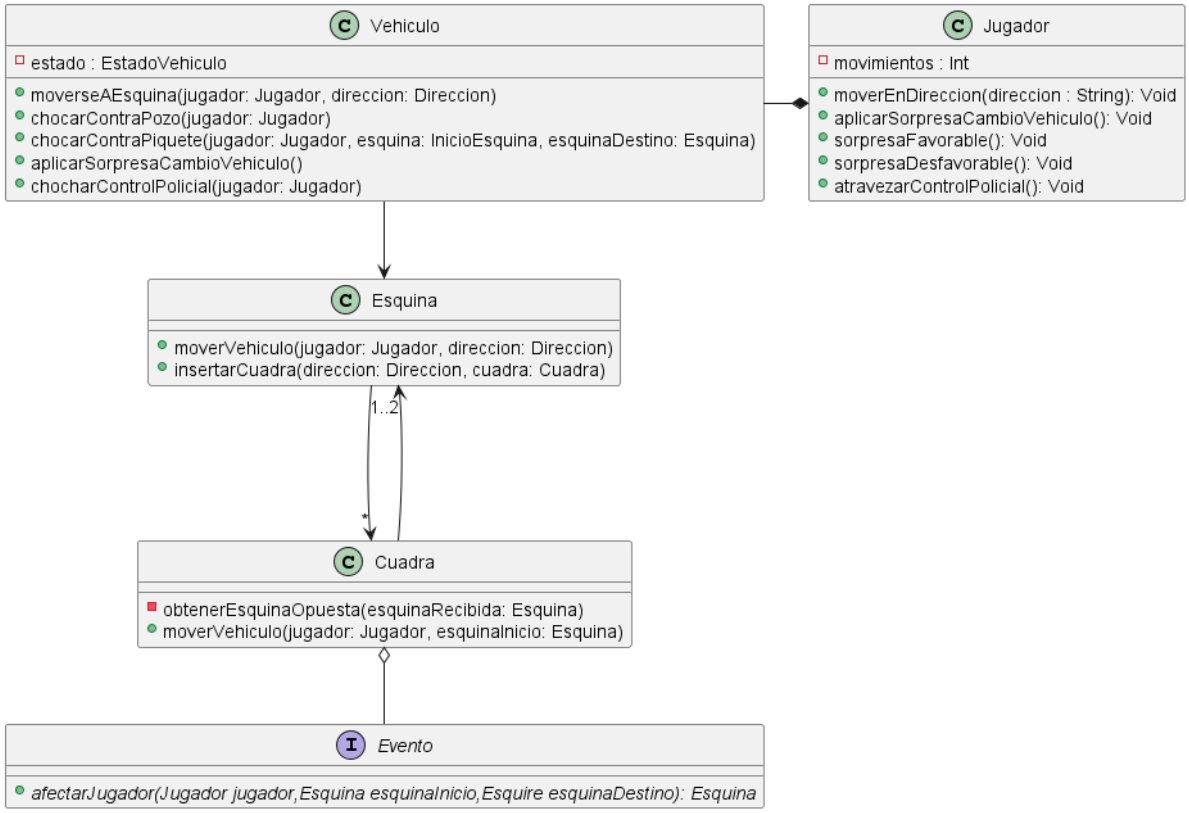
Índice

1. Supuestos

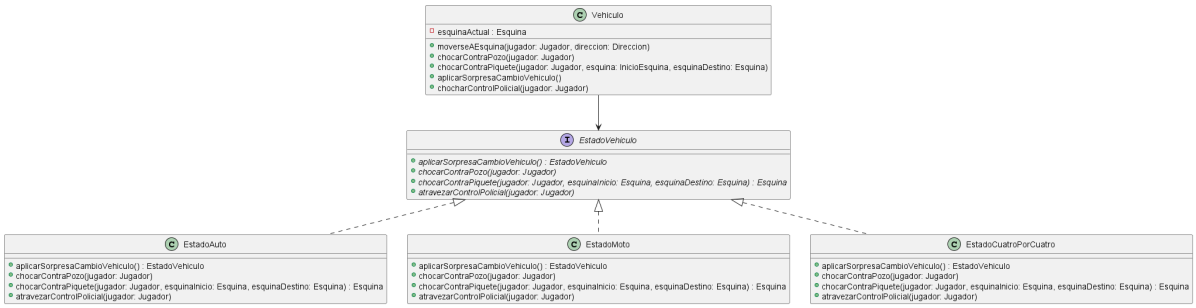
- El jugador tiene un vehículo.
- El tablero está realizado con esquinas y cuadras que se conectan.
- Todo desplazamiento del jugador tiene costo base 1 movimiento.
- El jugador arranca con 0 movimientos.
- Si una 4x4 pasa por un pozo 3 veces y es penalizada se reinicia el contador de pozos atravesados.
- Los movimientos realizados se redondean.
- La posición de la meta a la que el jugador tiene que llegar se encuentra en la esquina inferior izquierda del mapa.
- El jugador empieza a desplazarse por el mapa y a jugar con un Auto.
- La posición en la que el jugador aparece al iniciar el juego es en la esquina superior derecha del mapa.

2. Diagramas de clases

a. Diagrama del modelo general

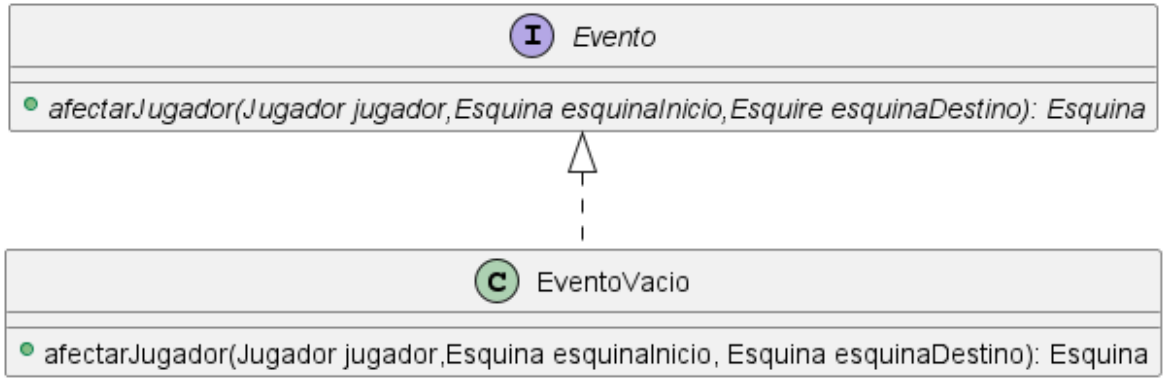


b. Diagrama en detalle de la clase vehículo y su estado

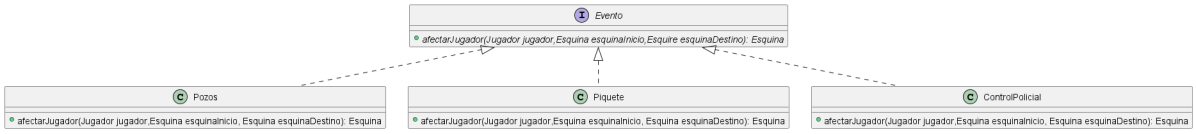


c. Diagramas en detalle de la clase evento

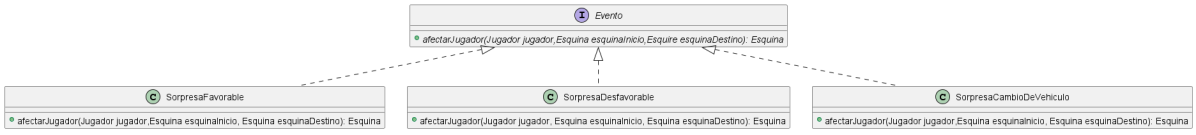
i. Diagrama de evento vacío



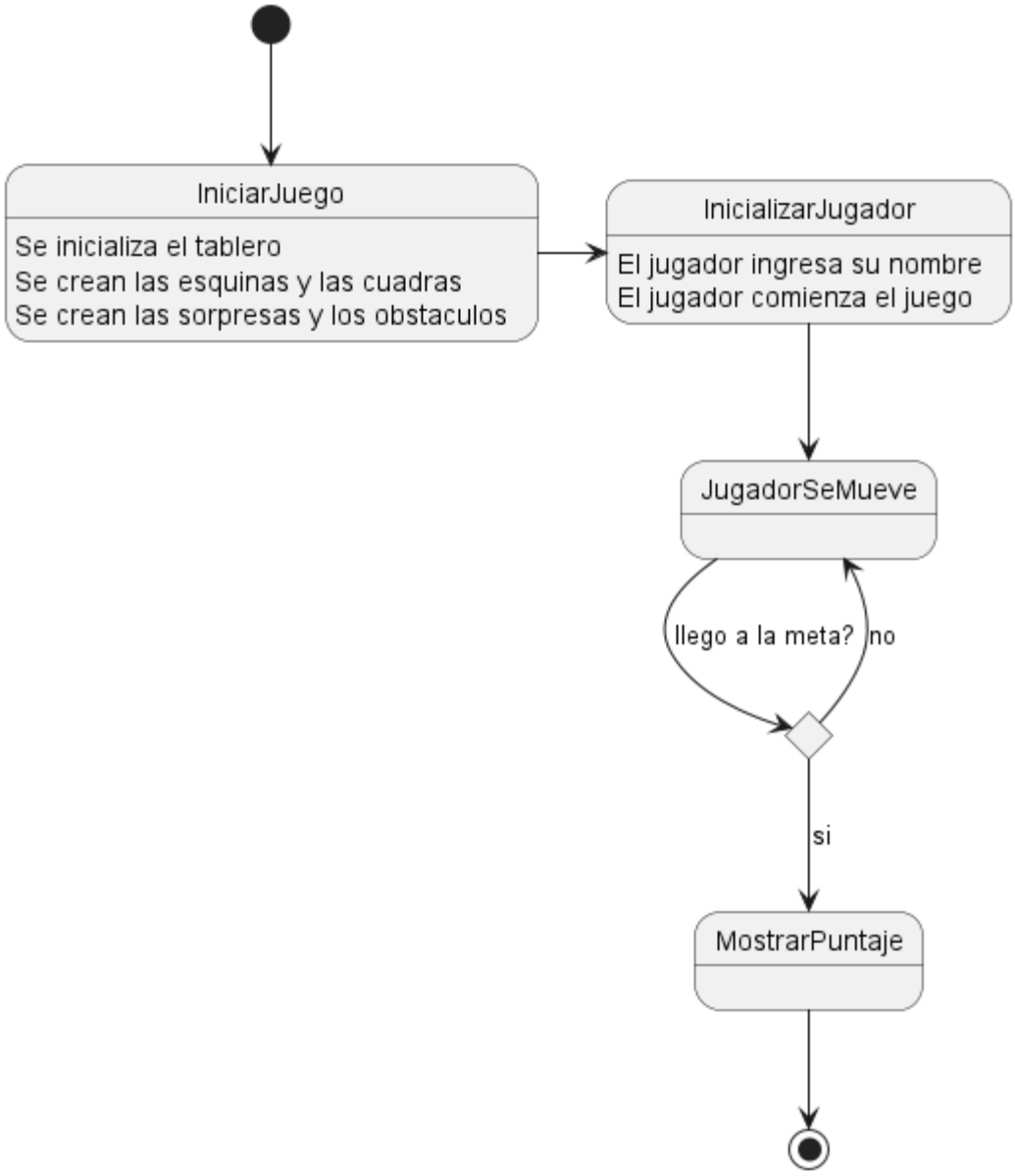
ii. Diagrama de las clases obstáculos



iii. Diagrama de las clases sorpresas



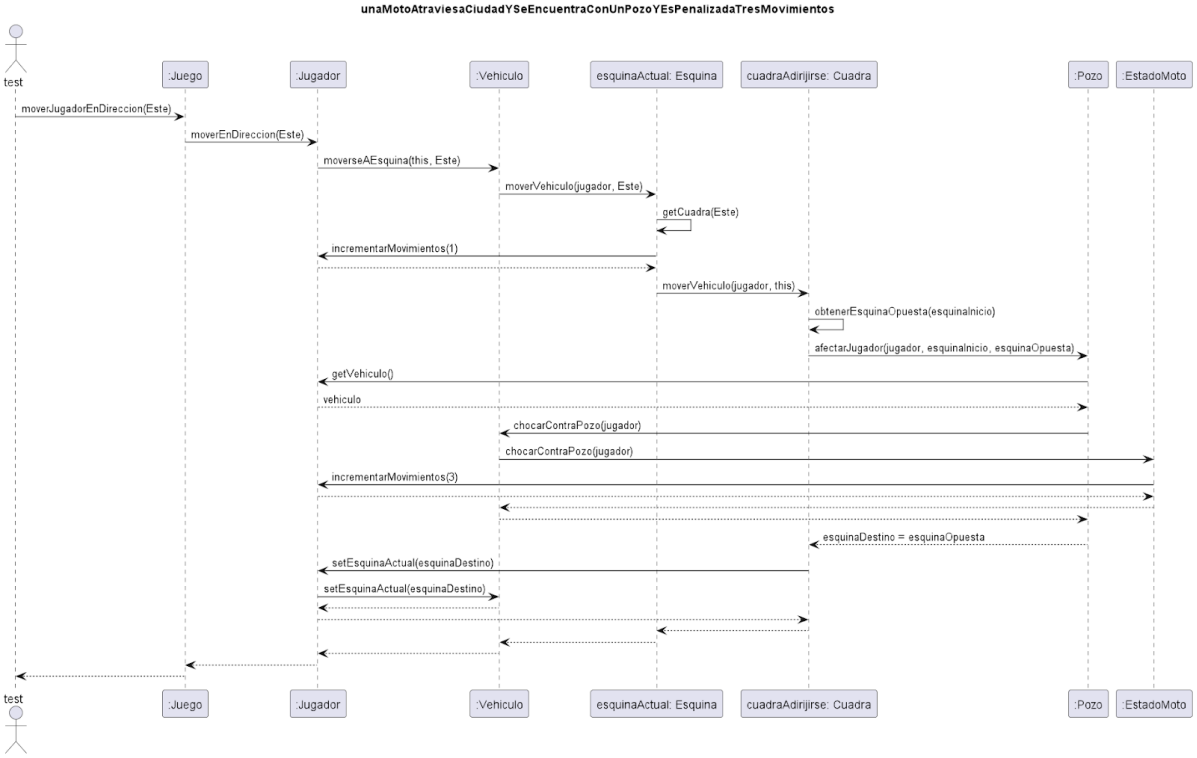
3. Diagrama de estado



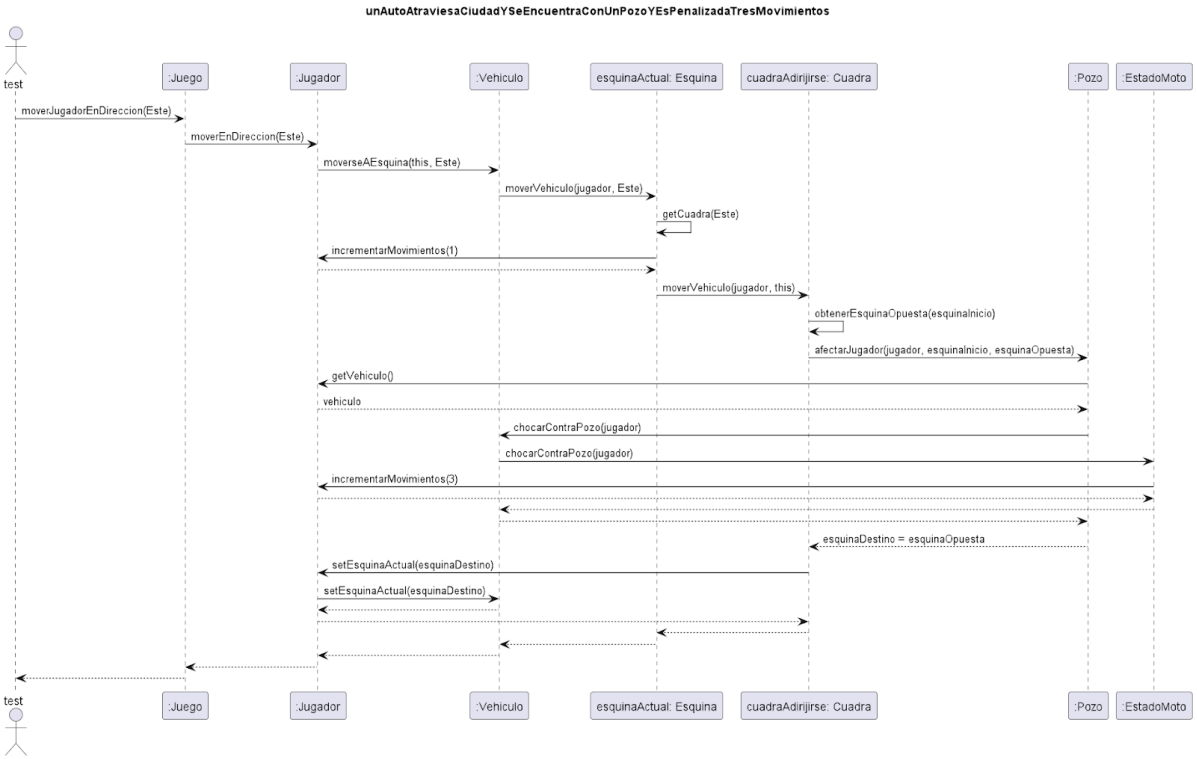
4. Diagramas de secuencia

a. Diagramas correspondientes a la entrega 1

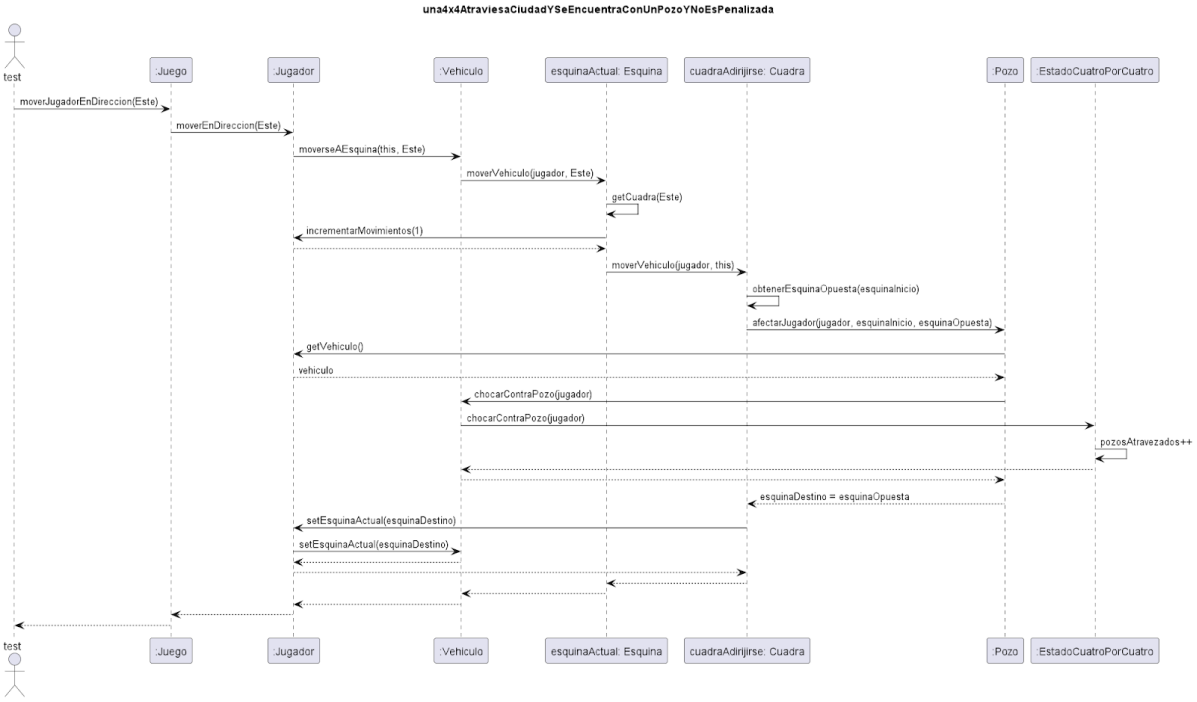
i. Caso de uso 1



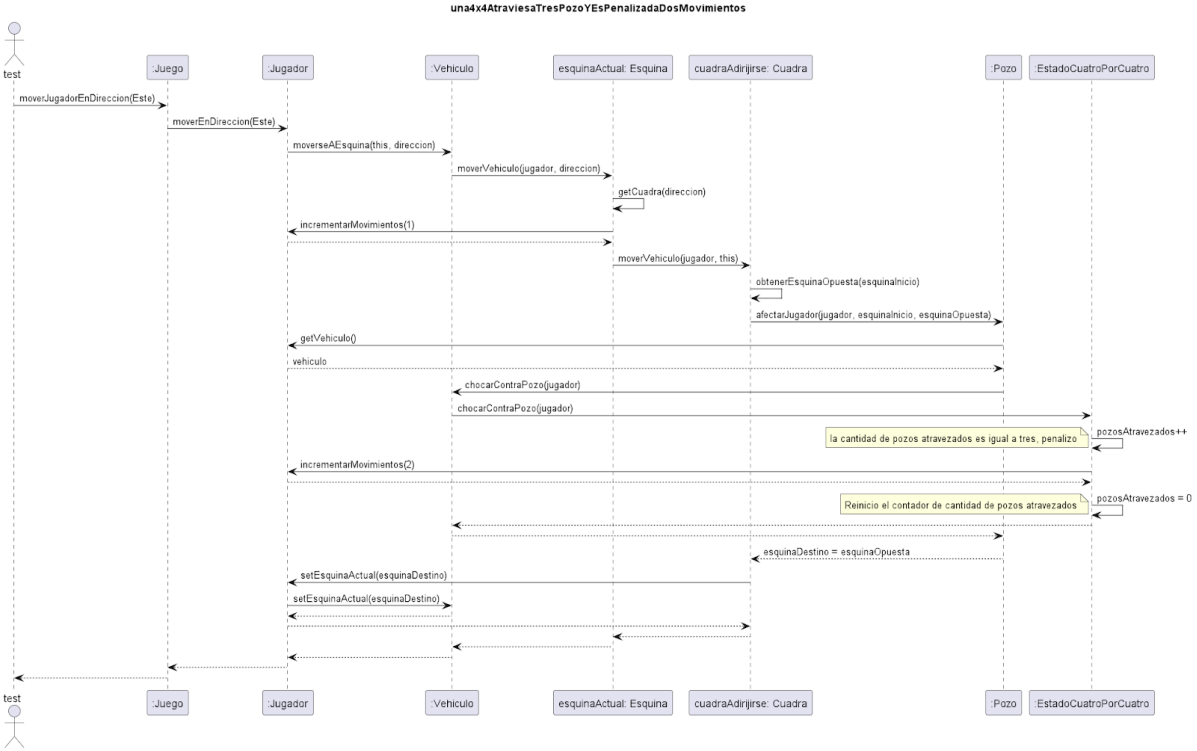
ii. Caso de uso 2



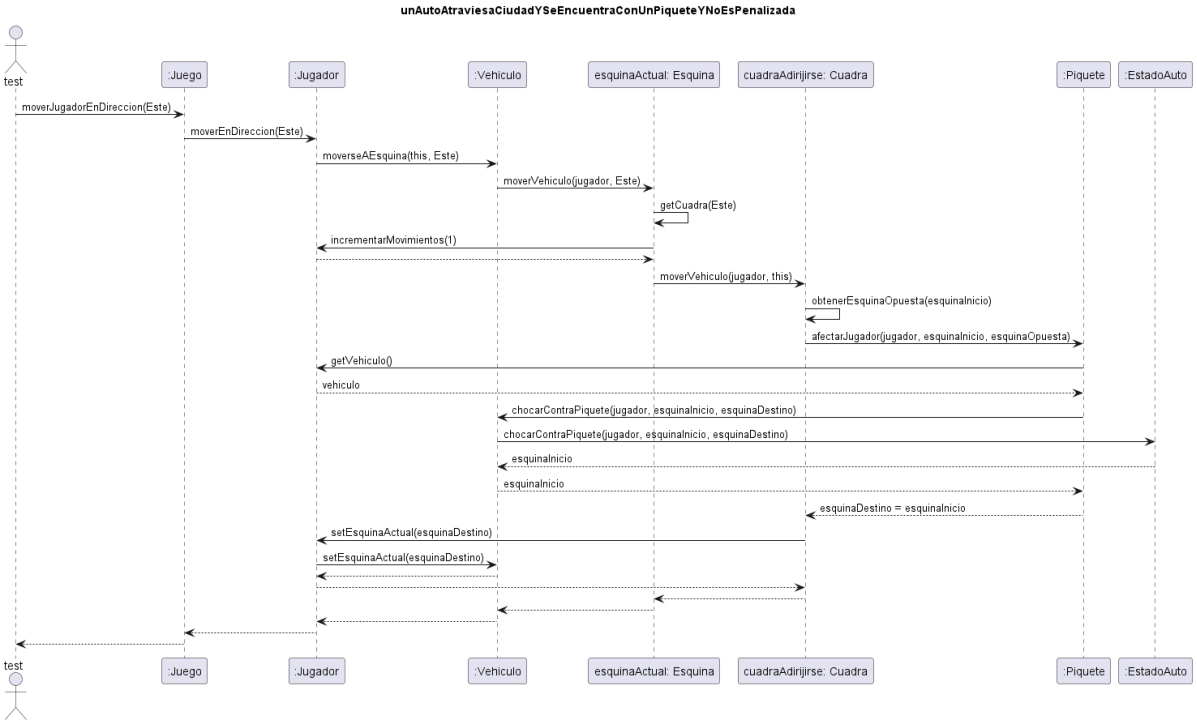
iii. Caso de uso 3



iv. Caso de uso 4

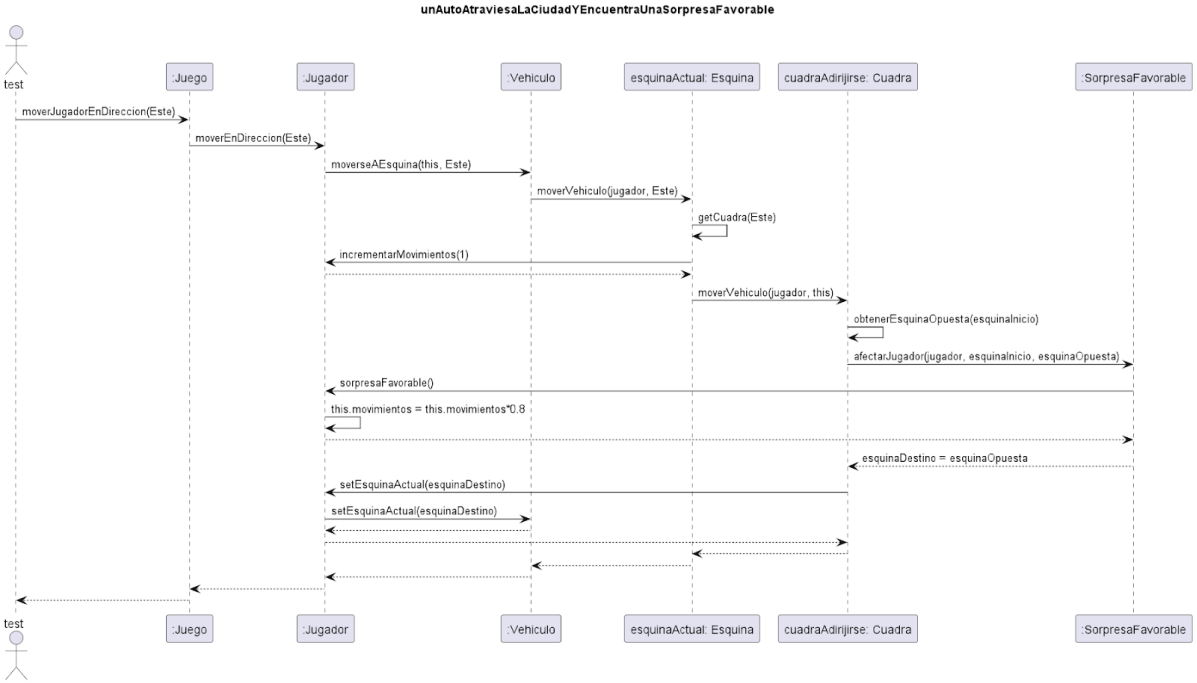


v. Caso de uso 5

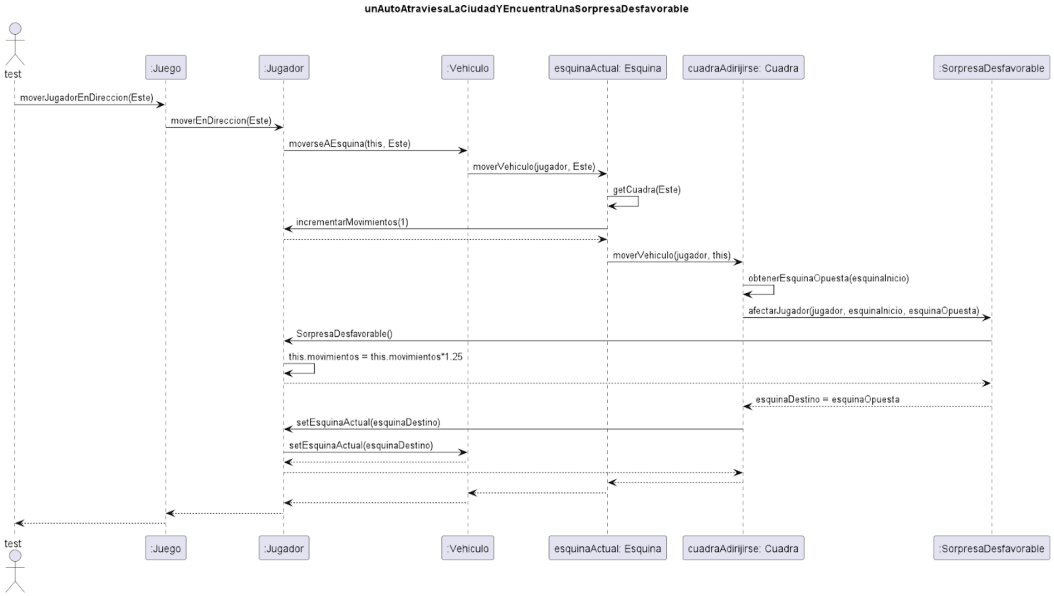


b. Diagramas correspondientes a la entrega 2

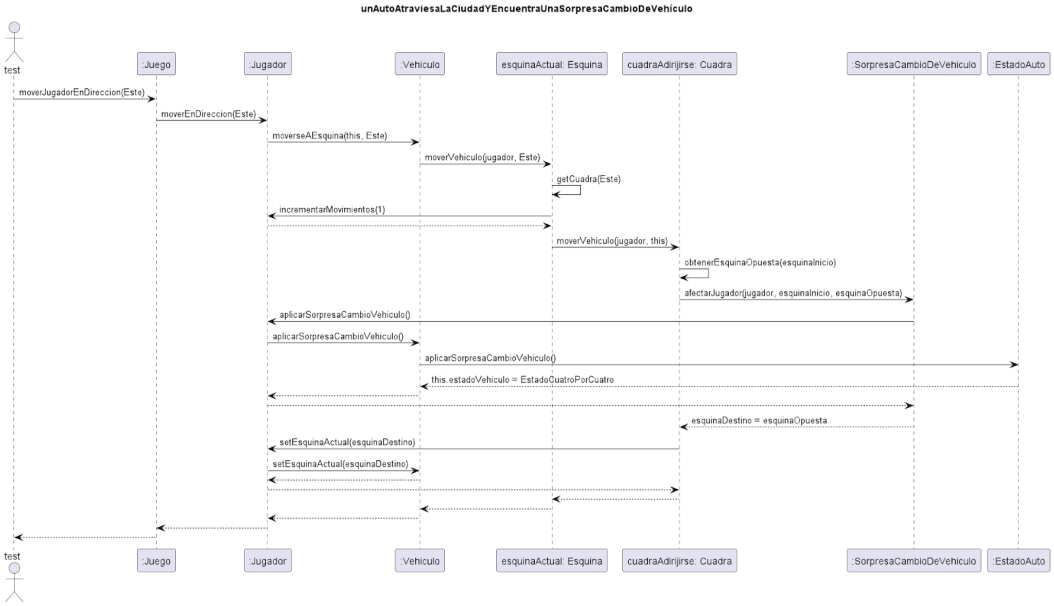
i. Caso de uso 1



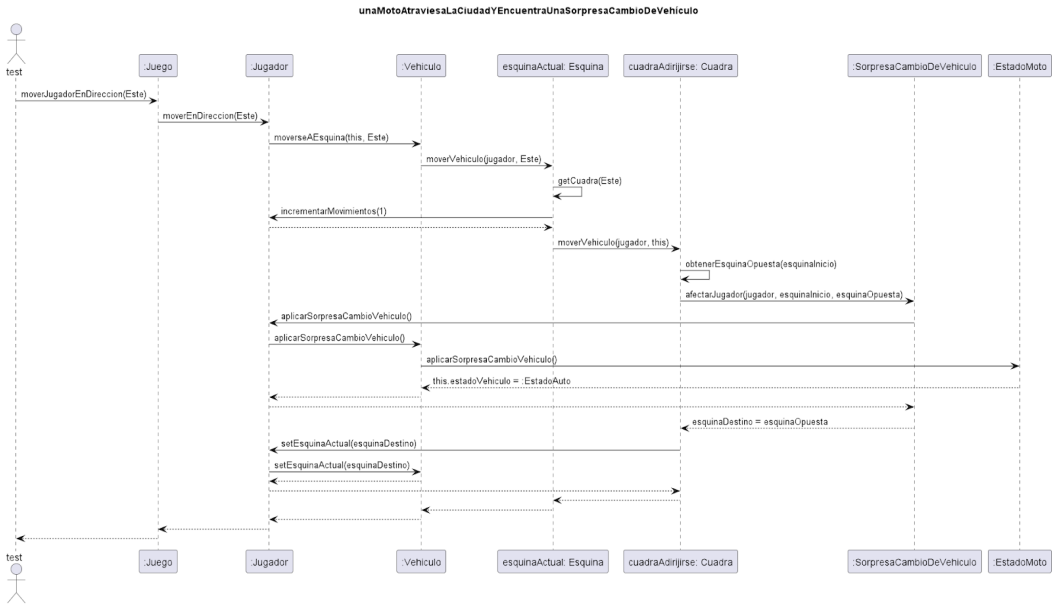
ii. Caso de uso 2



iii. Caso de uso 3



iv. Caso de uso 4



5. Detalles de implementación

- Eventos devuelvan la posición en la que se mueve: Idea que nació al tratar de solucionar una problemática que tuvimos con los piquetes, dado que si un Auto se choca con un piquete el mismo no puede pasar, entonces pensamos que lo mejor es que en lugar de devolver una excepción, devolvemos la posición en la que se encontraba el auto.
- Cómo modelamos el movimiento: Jugador contiene un método el cual recibe una dirección a la cual moverse, que a su vez es un objeto. Luego, el jugador le delega al vehículo a qué esquina moverse según la dirección recibida. Vehículo contiene la esquina actual sobre la cual está parado, es esta esquina quien recibe al jugador y a la dirección para obtener la cuadra correspondiente. Cuadra obtendrá la esquina opuesta de la cual partió inicialmente el jugador, finalmente se le incrementaron los movimientos al jugador y se le setea la nueva esquina.
- Uso de delegación: La delegación podemos verla en la clase Esquina y Cuadra. Ambas reciben un vehículo y acorde al comportamiento del mismo es necesario que a través de la Esquina el vehículo transite una Cuadra para encontrarse con un Obstáculo o Sorpresa.
- Uso de Herencia: El uso de la herencia se puede ver plasmado en la clase Evento que es de tipo Interfaz. La misma tiene por contrato un método el cual varias clases podrán responder de forma independiente según el propio comportamiento dado. Esto nos permite un desacoplamiento y claridad en el código.

a. Patrón de diseño utilizados

i. State

El atributo estado de la clase Vehiculo es una interfaz que son implementadas por las clases EstadoAuto, EstadoCuatroPorCuatro, y EstadoMoto.

Por su parte, el vehículo utiliza dicho atributo para adquirir distintos comportamientos dependiendo de qué comportamiento se le haya asignado. Por ejemplo, este patrón se ve especialmente útil cuando se quiere modelar la interacción entre el vehículo y una SorpresaDeCambio de vehículo, dado que mediante el uso del patrón State el vehículo puede alterar su comportamiento fácilmente.

ii. Double Dispach

Es un patrón que te permite manejar y resolver de forma sencilla la interacción entre clases por medio de una implementación abierta en la cual la clase que es interactuada recibe por parámetro que tipo de respuesta va a dar.
Este caso lo vemos claro en las clases Dirección implementada. El vehículo va a saber responder al mensaje *moverEnDireccion(unaDirección)* y con el mismo parámetro que va a variar en entre la clases Este,Oeste,Norte,Sur sabrá ya responder a su mensaje con un comportamiento correspondiente

iii. MVC

Utilizamos el patrón modelo vista controlador. Este nos sirve para la integración entre la interfaz gráfica, el modelo y los controladores que interactúan con entre el modelo y la vista

6. Excepciones

- ***Cuadra Inexistente Exception:***

En caso de que el jugador se quiera mover en una posición inválida saltará el error, por ejemplo: el jugador quiere moverse a una posición que está fuera del mapa.

- ***Esquina No Válida Exception:***

Cuando el jugador tenga que hacer un movimiento y reciba una esquina inexistente se lanza esta excepción

- ***Hay Un Piquete Exception :***

Esta excepción se lanza dependiendo del estado que el automóvil tenga. Esto nos permite utilizar desde el controlador el manejo de cambio de vehículo en la escena