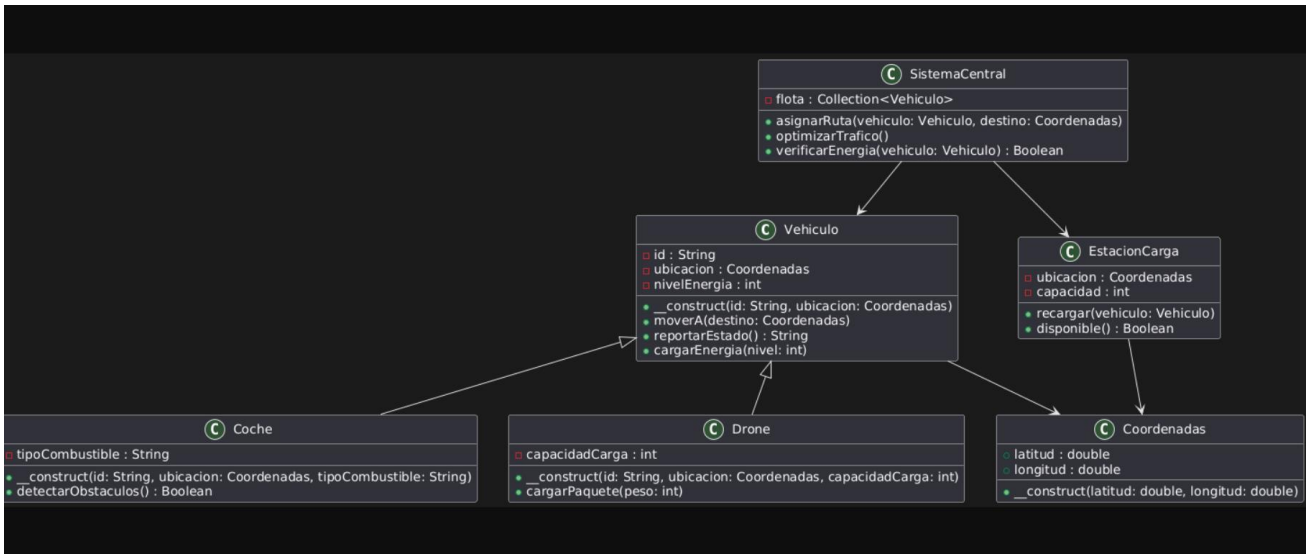
Actividad de análisis de clases

Fíjate en el siguiente diagrama de clases. Todos los aspectos de diseño, el porqué de estos símbolos y su significado lo has visto en cursos anteriores. En esta actividad vamos a repasar conceptos relacionados



Se pide:

1. Describe cómo están relacionadas cada una de las clases que figuran en el diagrama anterior
2. Herencia y Relaciones:

- **Identifica la relación de herencia** entre Vehiculo, Coche, y Drone. ¿Cómo implementa el sistema el polimorfismo en este contexto?

- ¿Qué ventajas tiene el hecho de que el SistemaCentral interactúe con la clase Vehiculo en lugar de con Coche o Drone directamente?

3. Encapsulación:

- ¿Por qué las propiedades como ubicacion y nivelEnergia están protegidas (representadas con -) y cómo se accede a ellas desde fuera de la clase?

4. Polimorfismo:

- Explica por qué el método moverA() sólo está en la clase vehículo. Es este método polimórfico

- ¿Cómo sería un ejemplo práctico de uso de polimorfismo en este sistema? Piensa en un algoritmo donde el SistemaCentral asigna rutas sin importar si es un coche o un dron.

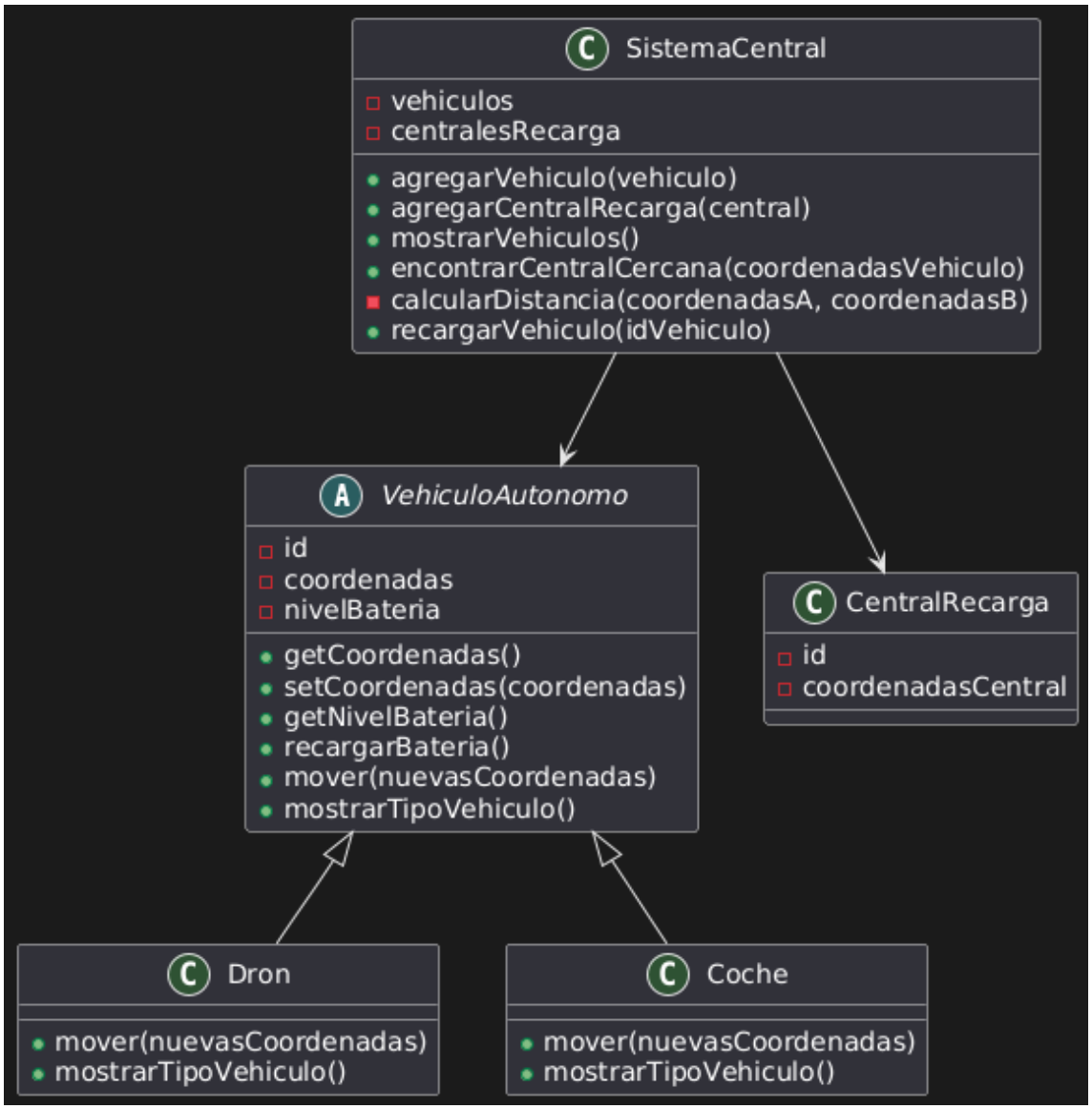
5. Relaciones y Dependencias:
    - ¿Qué tipo de relación existe entre Vehiculo y Coordenadas? ¿Qué responsabilidades tiene la clase Coordenadas dentro del sistema?
    - ¿Cómo interactúan el SistemaCentral y las Estaciones de Carga? Está esta relación bien representada ¿Qué falta?
6. Ampliación del Diagrama:
    - Si quisieras agregar un nuevo tipo de vehículo, como un camión autónomo, ¿qué modificaciones harías en el diagrama? ¿Qué atributos y métodos únicos podría tener este nuevo vehículo?
    - Imagina que deseas introducir un sistema de planificación de rutas basado en IA. ¿Cómo incorporarías este sistema al diagrama de clases?
7. Optimización de Rutas:
    - El SistemaCentral tiene un método optimizarTrafico(). Propón una implementación básica para este método que considere el número de vehículos y las rutas asignadas.
    - ¿Cómo podrías mejorar la eficiencia del sistema, en términos de consumo de energía, utilizando la clase EstacionCarga?

Actividad de análisis e implementación de clases

Fíjate en el siguiente modelo de clases. Representa el diseño de **un sistema de control de vehículos autónomos. Ten en cuenta que el modelo está incompleto**

El modelo representa un sistema central de control de vehículos autónomos, los cuales están geolocalizados (su estado refleja su posición espacial en todo momento). El sistema obtiene información diversa de cada vehículo . El sistema planifica el viaje a la estación de recarga si el vehículo llega a un porcentaje muy bajo de batería.

Modelo de clase del Sistema



Los drones tienen cuatro motores, cuyo estado es necesario supervisar. Su estado será 1 si funciona bien y 0 si no funciona.

Los coches autónomos necesitan tener siempre el sistema de supervisión de conductor en estado ok, si está en error no podrá circular.

Las centrales de recarga están todos georreferenciadas, para poder planificar una ruta hacia ellas, y tendrán un nombre. La central tiene un número de puestos de carga libres que deberá ser observado por el sistema de control a la hora de elegirla para que un vehículo autónomo.

Desarrollo de la actividad práctica

1.- Realiza los cambios más apropiados para que el modelo refleje las características que se indican

2.- Codifica un programa de prueba en php que cree varios vehículos autónomos y que los registre en el sistema de control con las siguientes restricciones

- Las clases van en un modulo que se llama SistemaCentral.php

- El modulo de prueba debe importar este módulo para trabajar con el

A continuación se muestra un esquema de las clases de implementación completa las partes

```
<?php

// Clase base para todos los vehículos autónomos
abstract class VehiculoAutonomo {
    protected $id;
    protected $coordenadas; // [x, y] array
    protected $nivelBateria;

    public function __construct($id, $coordenadas, $nivelBateria
= 100) {
        ‘ ----------------------'
    }

    public function getCoordenadas() {
        ‘ ----------------------'

    }

    public function setCoordenadas($coordenadas) {
        ‘ ----------------------'

    }
```

```
    public function getNivelBateria() {
        ‘ ----------------------'

    }

    public function recargarBateria() {
        ‘ ----------------------'
    }

    abstract public function mover($nuevasCoordenadas);

    abstract public function mostrarTipoVehiculo();
}

// Clase para Dron, que extiende VehiculoAutonomo
class Dron extends VehiculoAutonomo {

    public function mover($nuevasCoordenadas) {
        // Lógica de movimiento específica para el dron (por
aire)
        echo "El dron se mueve de " . implode(",", $this-
>coordenadas) . " a " . implode(",", $nuevasCoordenadas) . "\n";
        $this->coordenadas = $nuevasCoordenadas;
    }

    public function mostrarTipoVehiculo() {
        echo "Este es un dron.\n";
    }
}

// Clase para Coche, que extiende VehiculoAutonomo
```

```
class Coche extends VehiculoAutonomo {


    public function mover($nuevasCoordenadas) {

        // Lógica de movimiento específica para el coche (por
carretera)

        echo "El coche se mueve de " . implode(",", $this-
>coordenadas) . " a " . implode(",", $nuevasCoordenadas) . "\n";

        $this->coordenadas = $nuevasCoordenadas;

    }


    public function mostrarTipoVehiculo() {

        echo "Este es un coche autónomo.\n";

    }

}


// SistemaCentral.php

class SistemaCentral {

    private $vehiculos = []; //implementa un array asociativo
[‘id_vehiculo’=>objetovehiculo]

    private $centralesRecarga = [];


    public function agregarVehiculo(VehiculoAutonomo $vehiculo)
{

        $this->vehiculos[$vehiculo->id] = $vehiculo;

    }


    public function agregarCentralRecarga(CentralRecarga
$central) {

        $this->centralesRecarga[$central->id] = $central;

    }


    public function mostrarVehiculos() {
```

```
        foreach ($this->vehiculos as $vehiculo) {
                '----------------------------'
        }
    }

    public function
encontrarCentralCercana($coordenadasVehiculo) {
        $centralMasCercana = null;
        $distanciaMinima = PHP_INT_MAX;

        foreach ($this->centralesRecarga as $central) {
            $distancia = $this-
>calcularDistancia($coordenadasVehiculo, $central-
>coordenadasCentral);
            if ($distancia < $distanciaMinima) {
                $distanciaMinima = $distancia;
                $centralMasCercana = $coordenadasCentral;
            }
        }

        return $centralMasCercana;
    }

    private function calcularDistancia($coordenadasA,
$coordenadasB) {
        return sqrt(pow($coordenadasB[0] - $coordenadasA[0], 2)
+ pow($coordenadasB[1] - $coordenadasA[1], 2));
    }

    public function recargarVehiculo($idVehiculo) {
        if (isset($this->vehiculos[$idVehiculo])) {
            $vehiculo = $this->vehiculos[$idVehiculo];
```

```
            $centralCercana = $this-
>encontrarCentralCercana($vehiculo->getCoordenadas());


            if ($centralCercana) {
                echo "Recargando el vehículo en la central de
recarga más cercana en: " . implode(",", $centralCercana) .
"\n";
                $vehiculo->recargarBateria();
            } else {
                echo "No hay centrales de recarga
disponibles.\n";
            }
        } else {
            echo "Vehículo no encontrado.\n";
        }
    }
}



?>
```

Plantea un módulo PHP prueba.php

```
// prueba.php


// 1.-Instancia el SistemaCentral


// 2.- crea un Dron en coordenadas [10,20] y un Coche en
coordenadas [30,40]


//3.-Agrega los vehículos en el Sistema central
//Agrega una central de recarga en [15,25] y [35,45]
```

```
// Muestra los vehículos que tiene el sistema

$sistema->mostrarVehiculos();

// Recarga el dron

// Recarga el coche
```

Cuestiones

a) ¿ Por qué la clase Vehículo es abstracta?
b) ¿ Por qué las siguientes funciones son abstractas?

```
abstract public function mover($nuevasCoordenadas);

abstract public function mostrarTipoVehiculo();
```