<!--Diseño de Software-->

State {

<Por="Dylan Rodríguez/>

}

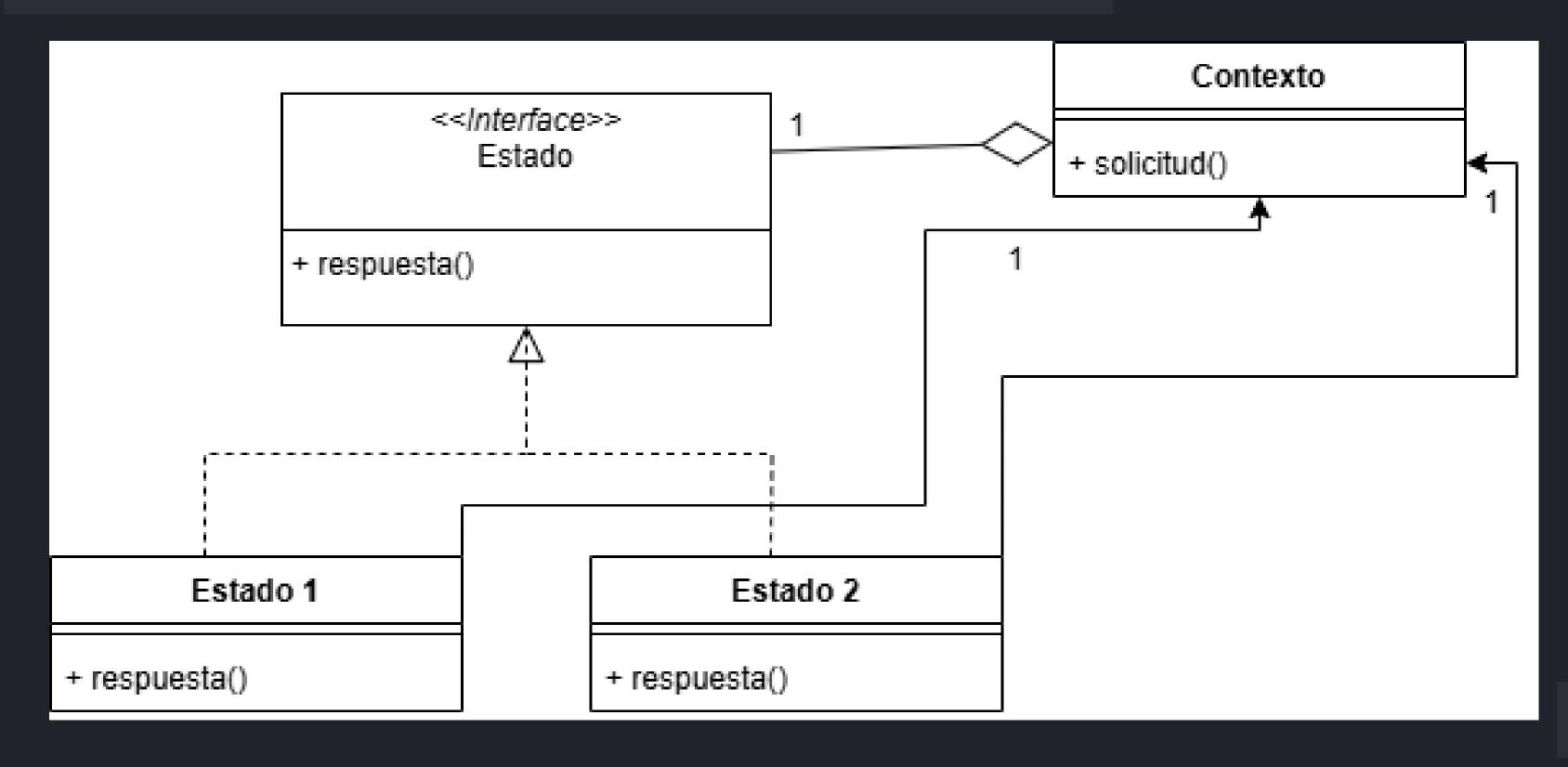
¿Qué es?

Es un patrón de diseño que permite a un objeto cambiar su comportamiento dependiendo de su estado.

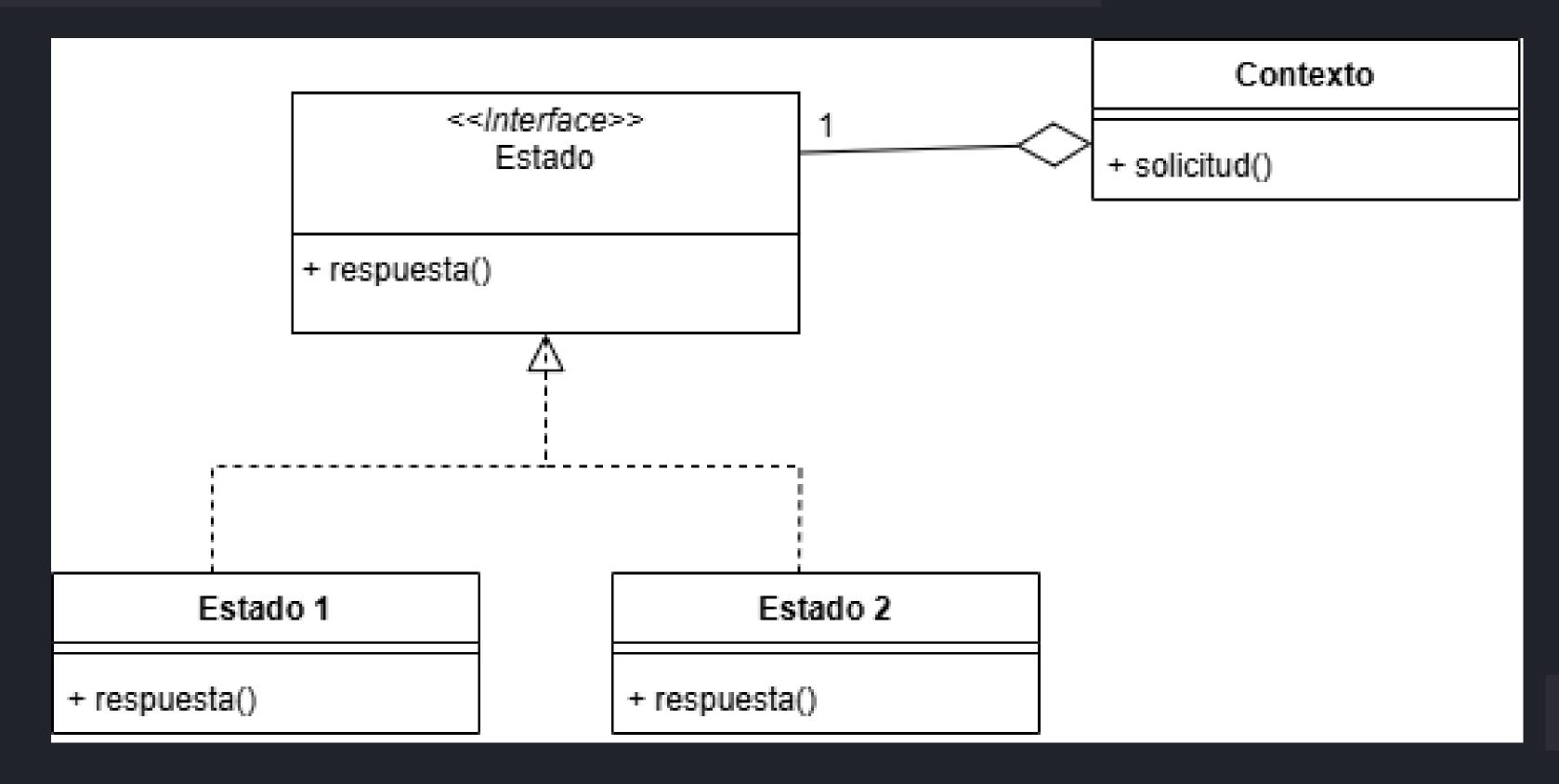




Implementación a nivel de diagrama {



Implementación a nivel de diagrama {



Implementación incorrecta a nivel de diagrama {

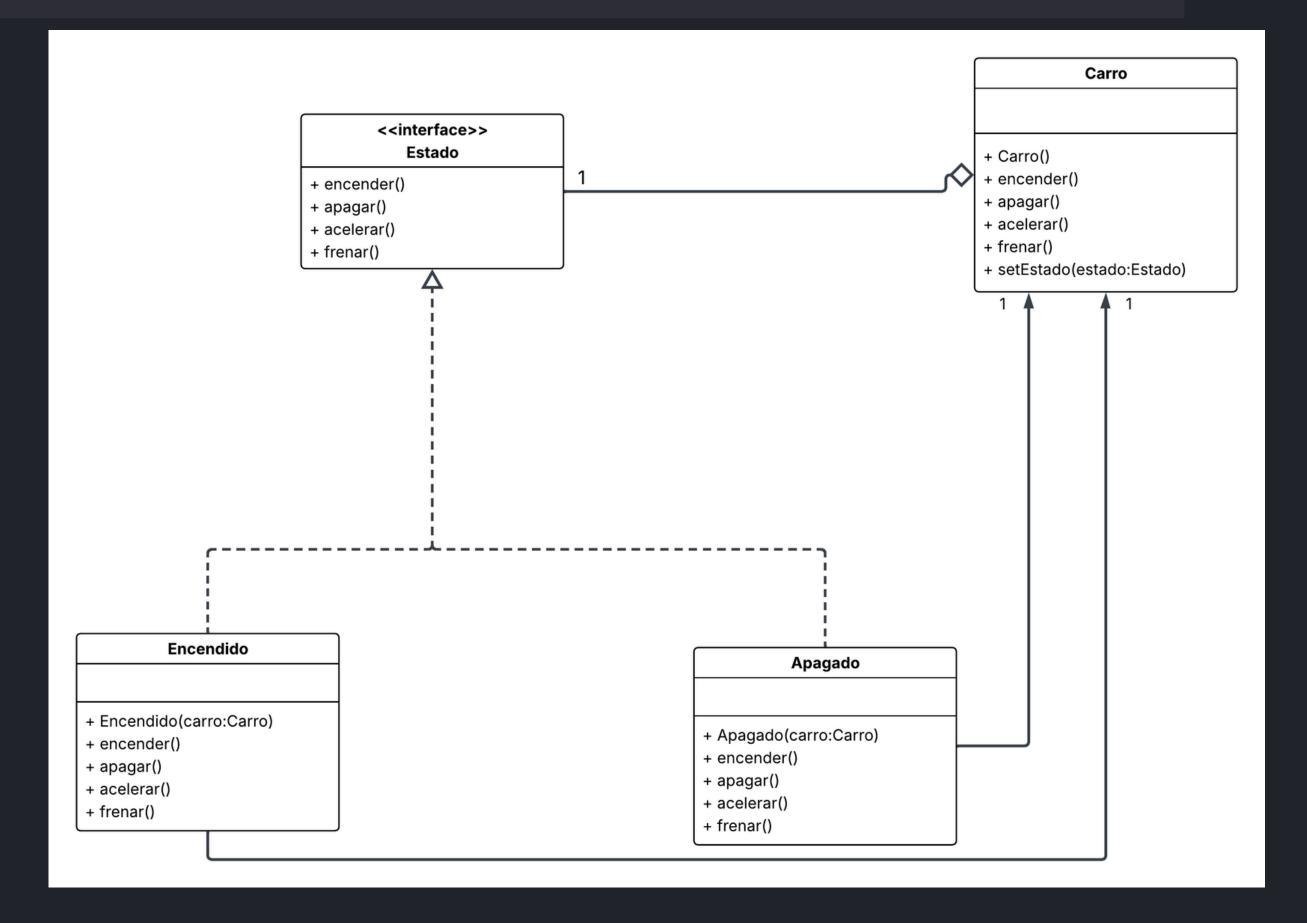
Contexto

- estado
- + accion1()
- + accion2()

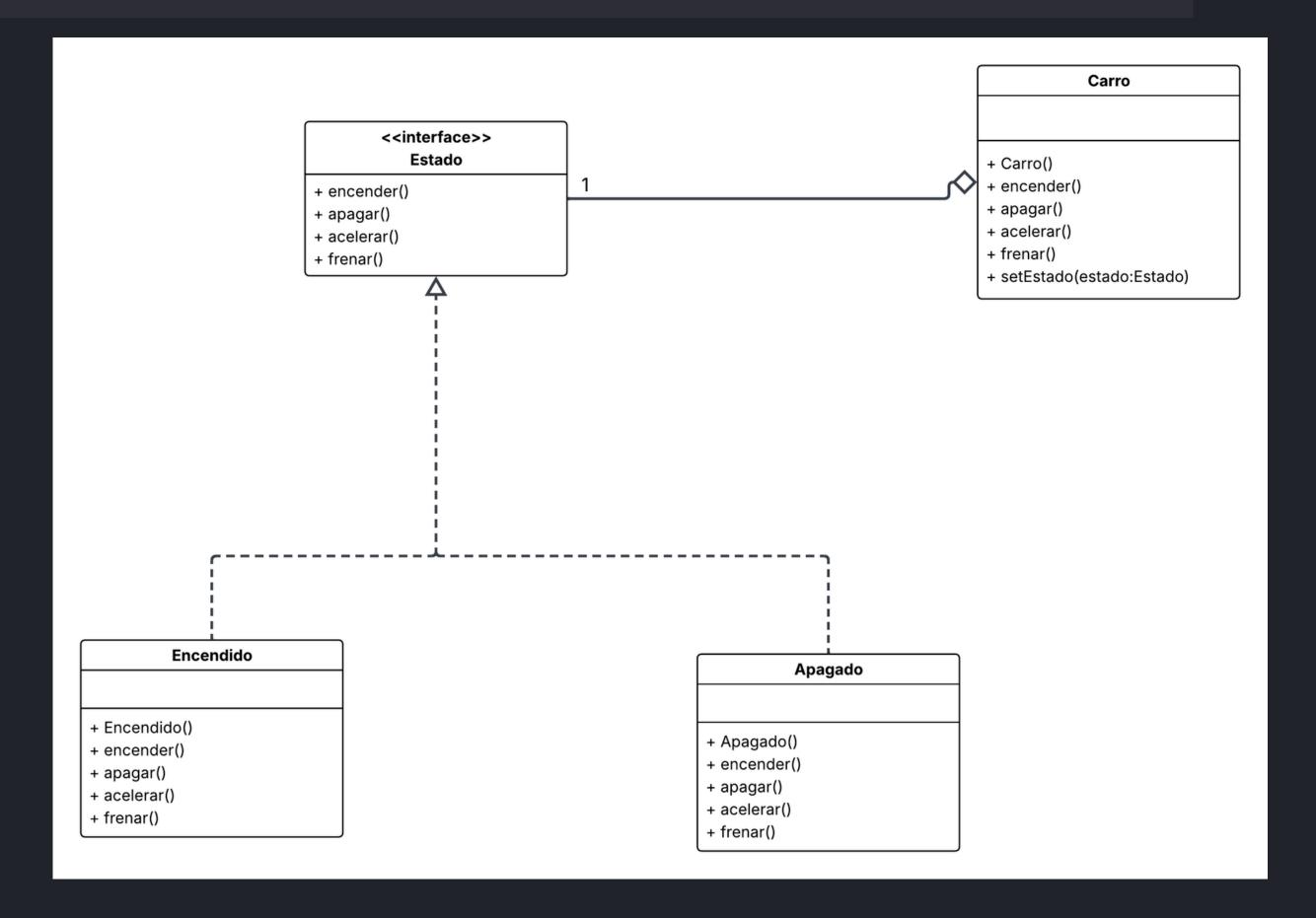
Ventajas

- Permite añadir fácilmente nuevos estados a la clase.
- Genera un código mejor estructurado ya que su funcionamiento no reside en el uso de clausulas if-then-else o switch.
- La transición entre estados es más sencilla.
- Cumple con el principio Open/Closed.

Ejemplo Carro - Cambio de estado interno {



Ejemplo Carro - Cambio de estado externo {

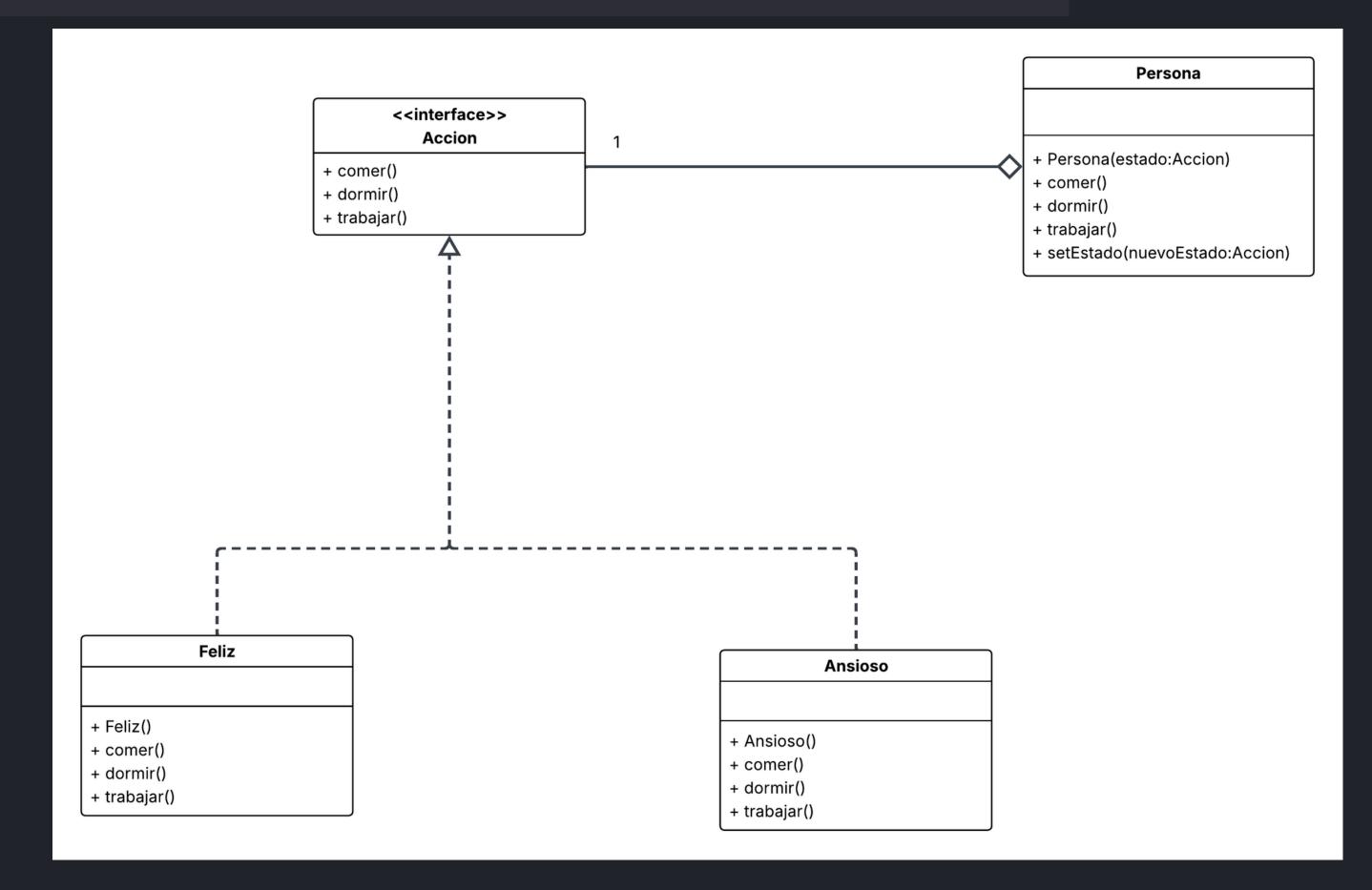


Ejemplo Carro - Sin aplicar el patrón {

Carro

- estado: boolean
- + Carro()
- + encender()
- + apagar()
- + acelerar()
- + frenar()

Ejemplo Persona - Aplicando el patrón {

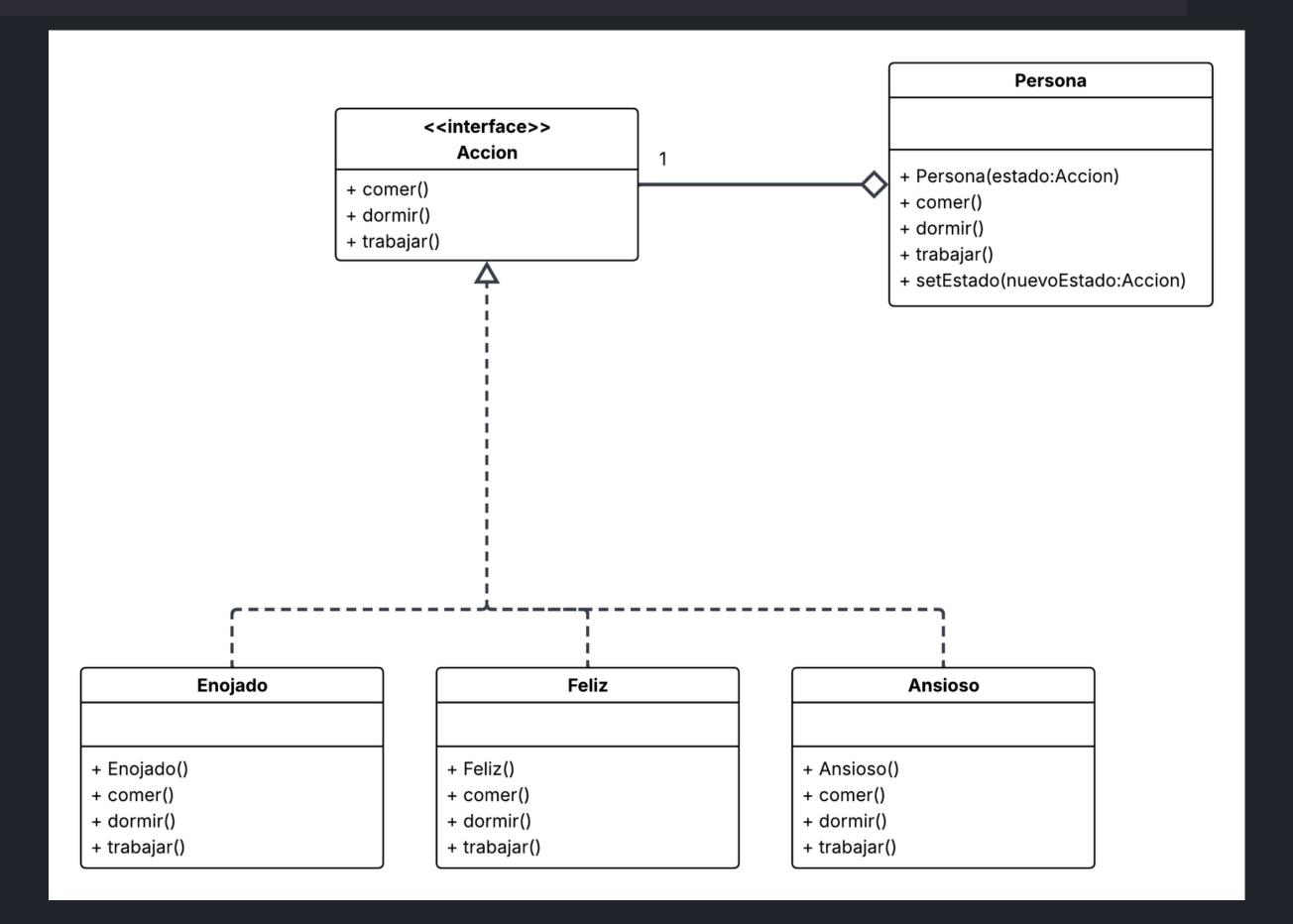


Ejemplo Persona - Sin aplicar el patrón {

Persona

- estado:String
- + Persona(estado:String)
- + trabajar()
- + dormir()
- + comer()
- + setEstado(estado:String)

Ejemplo Persona - Agregando más elementos {



```
Gracias {
```

}