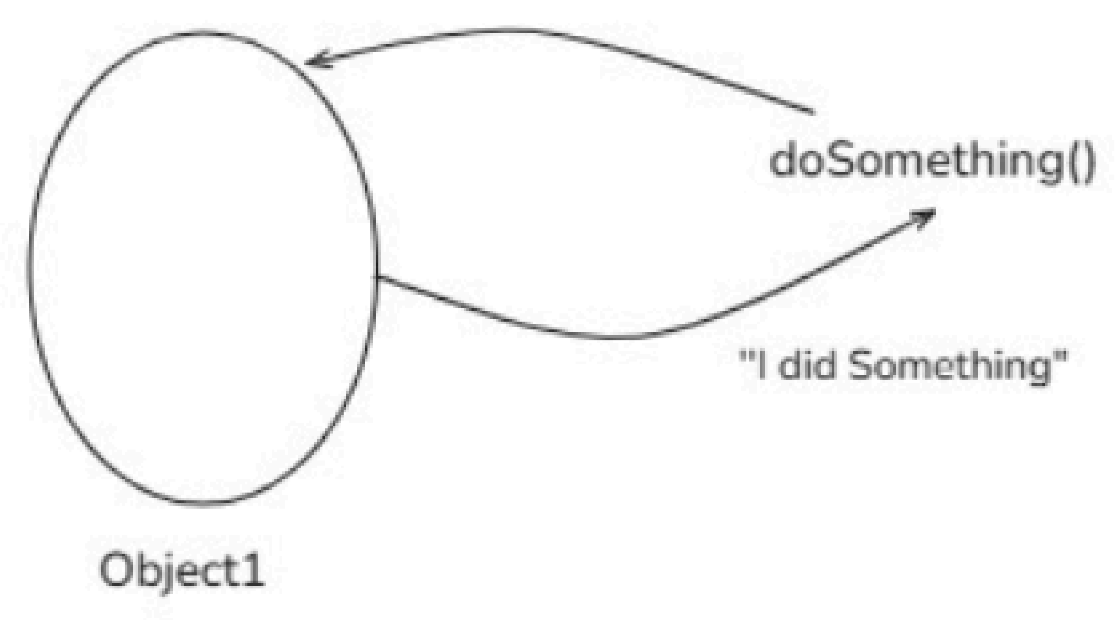
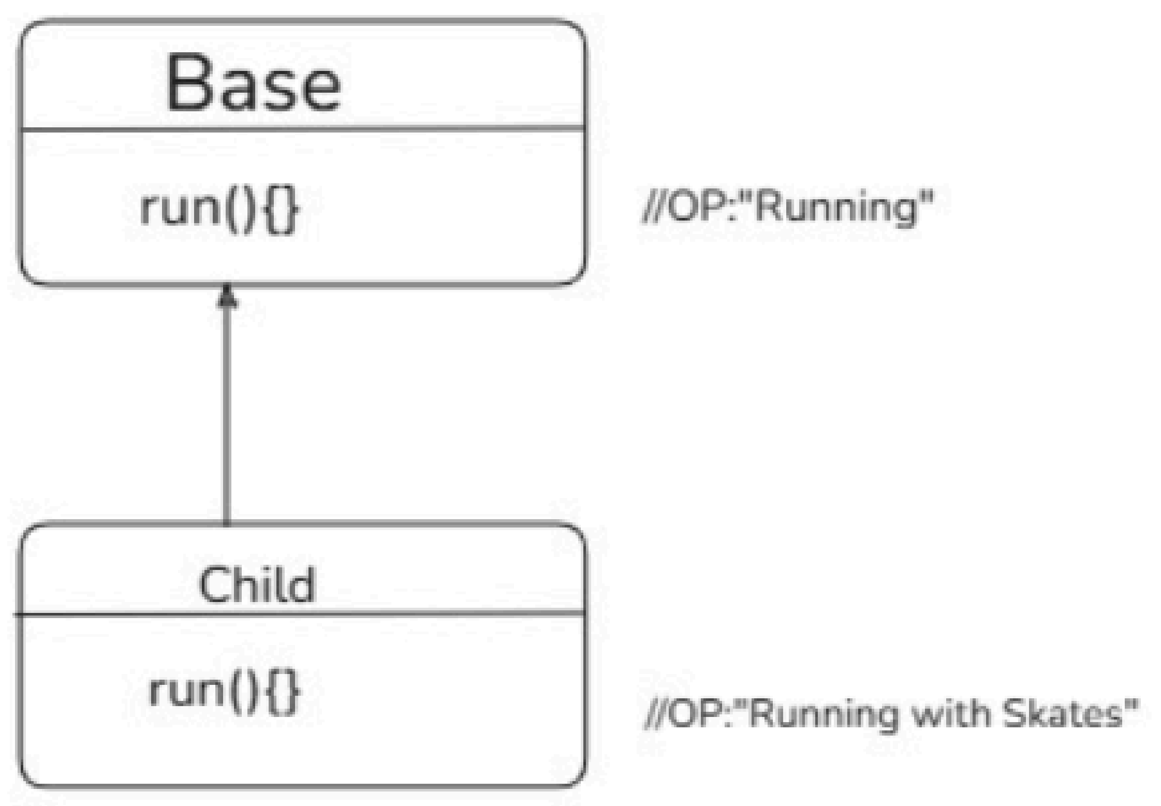
DECORATOR DESIGN PATTERN



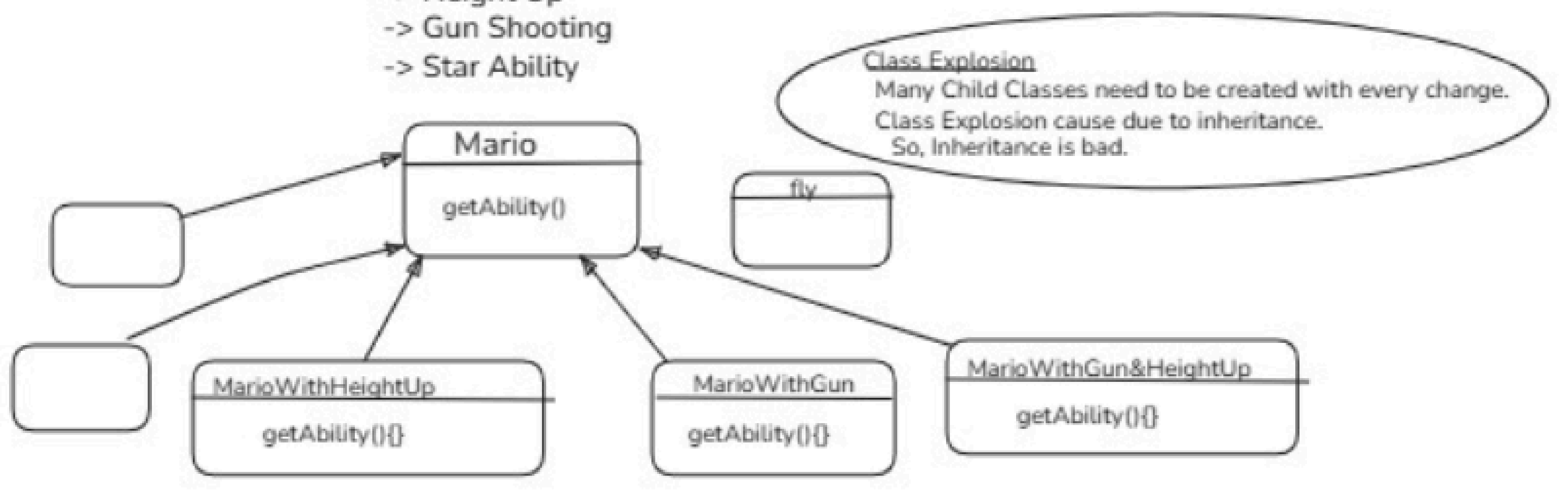
• provide additional responsibilites(functionalities) at runtime by Object1.



```
Base* b = new Child();
b->run();//OP:"Running with Skates"
```

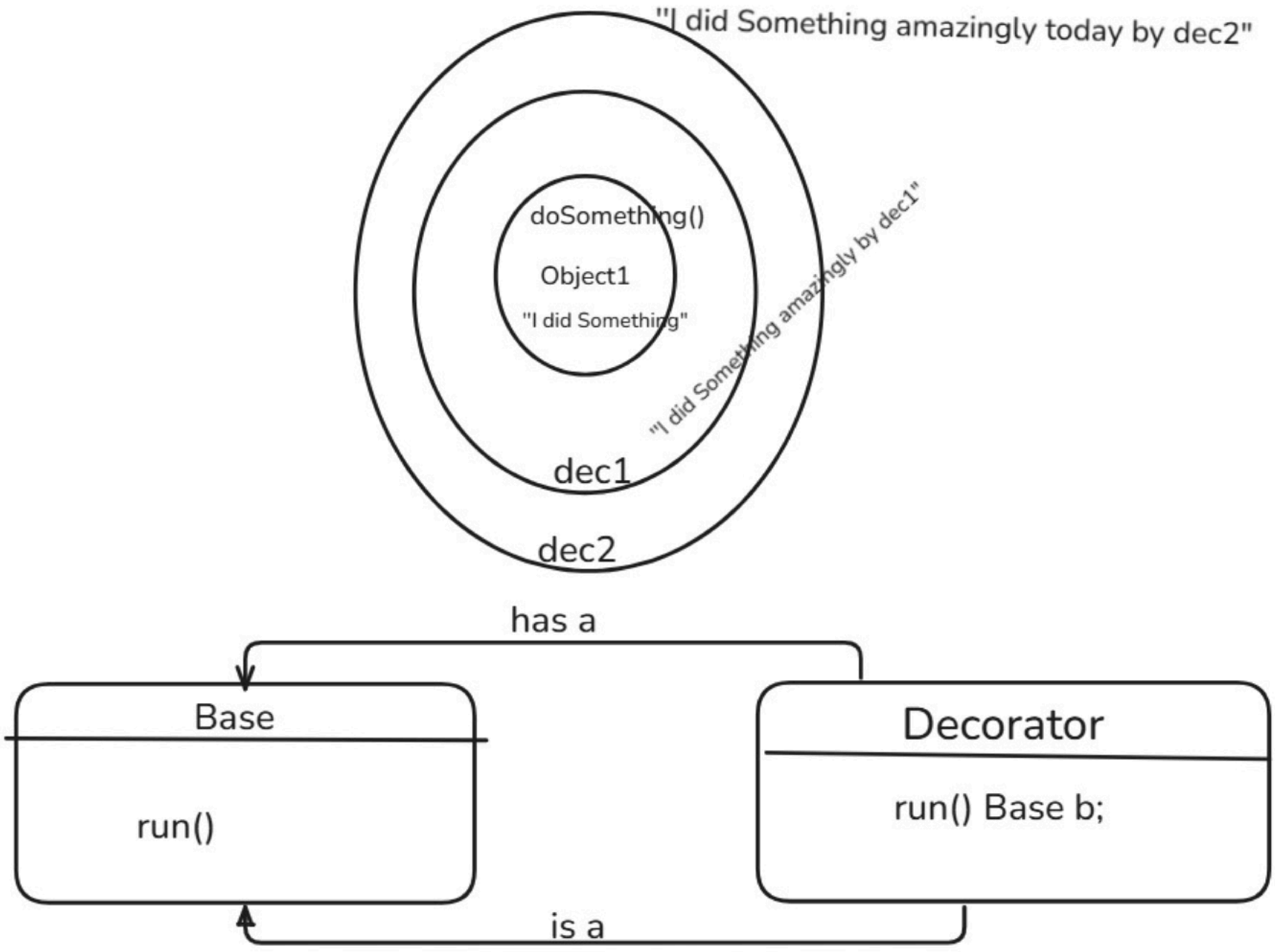
Mario Game Example using Inheritance

PowerUps
-> Height Up
-> Gun Shooting
-> Star Ability

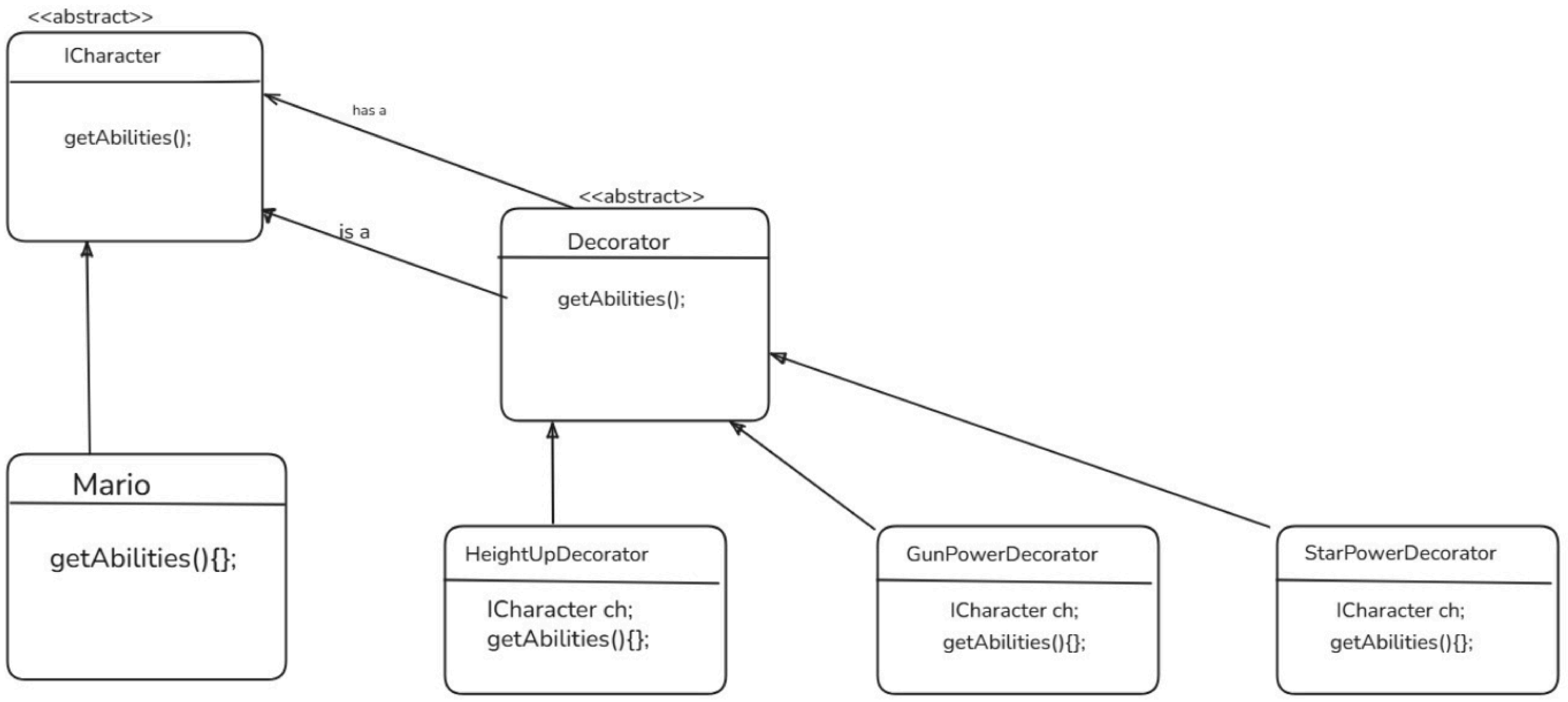


Class Explosion
Many Child Classes need to be created with every change.
Class Explosion cause due to inheritance.
So, Inheritance is bad.

Mario Game Example using Decorators



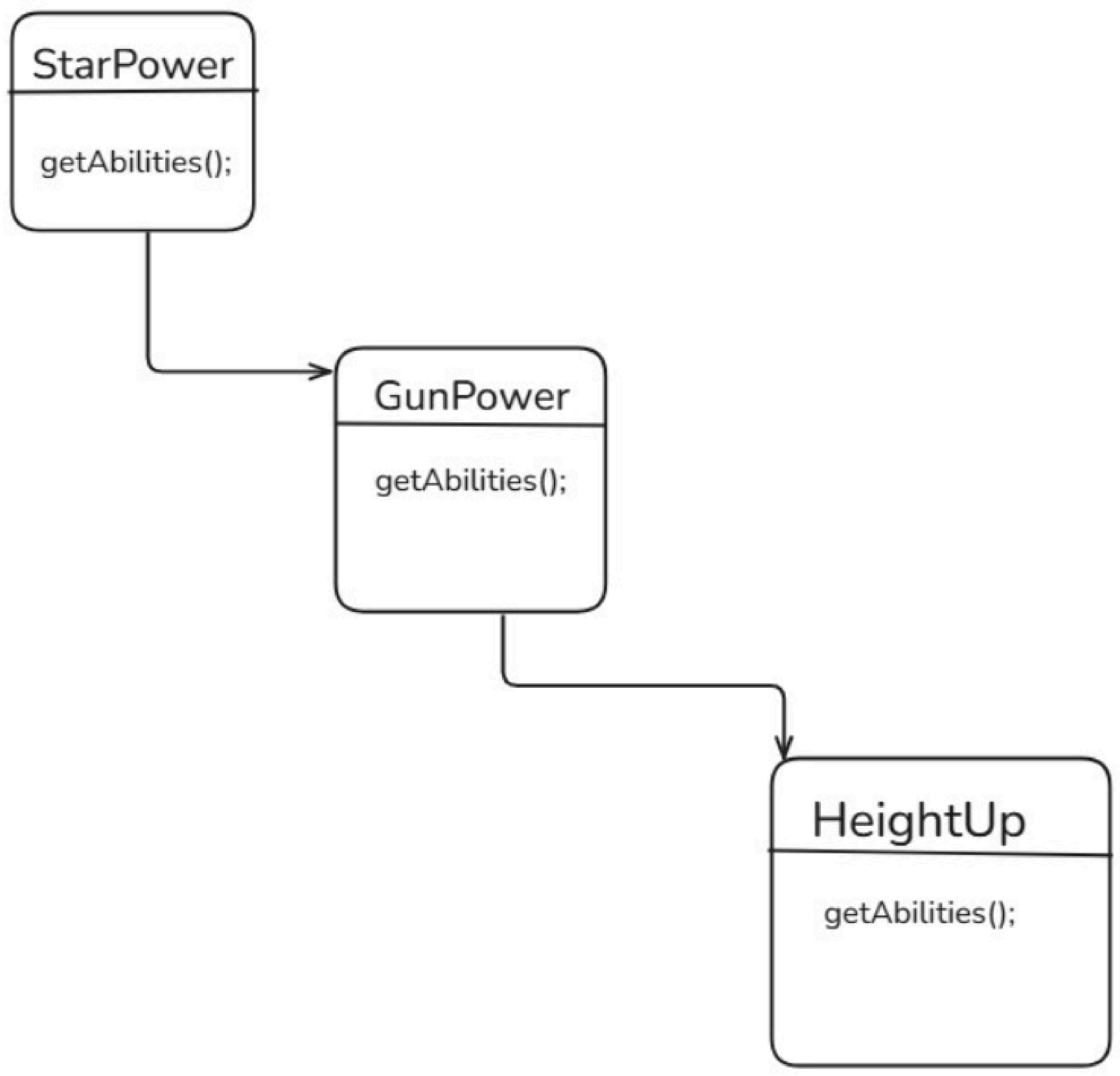
UML For Mario Example



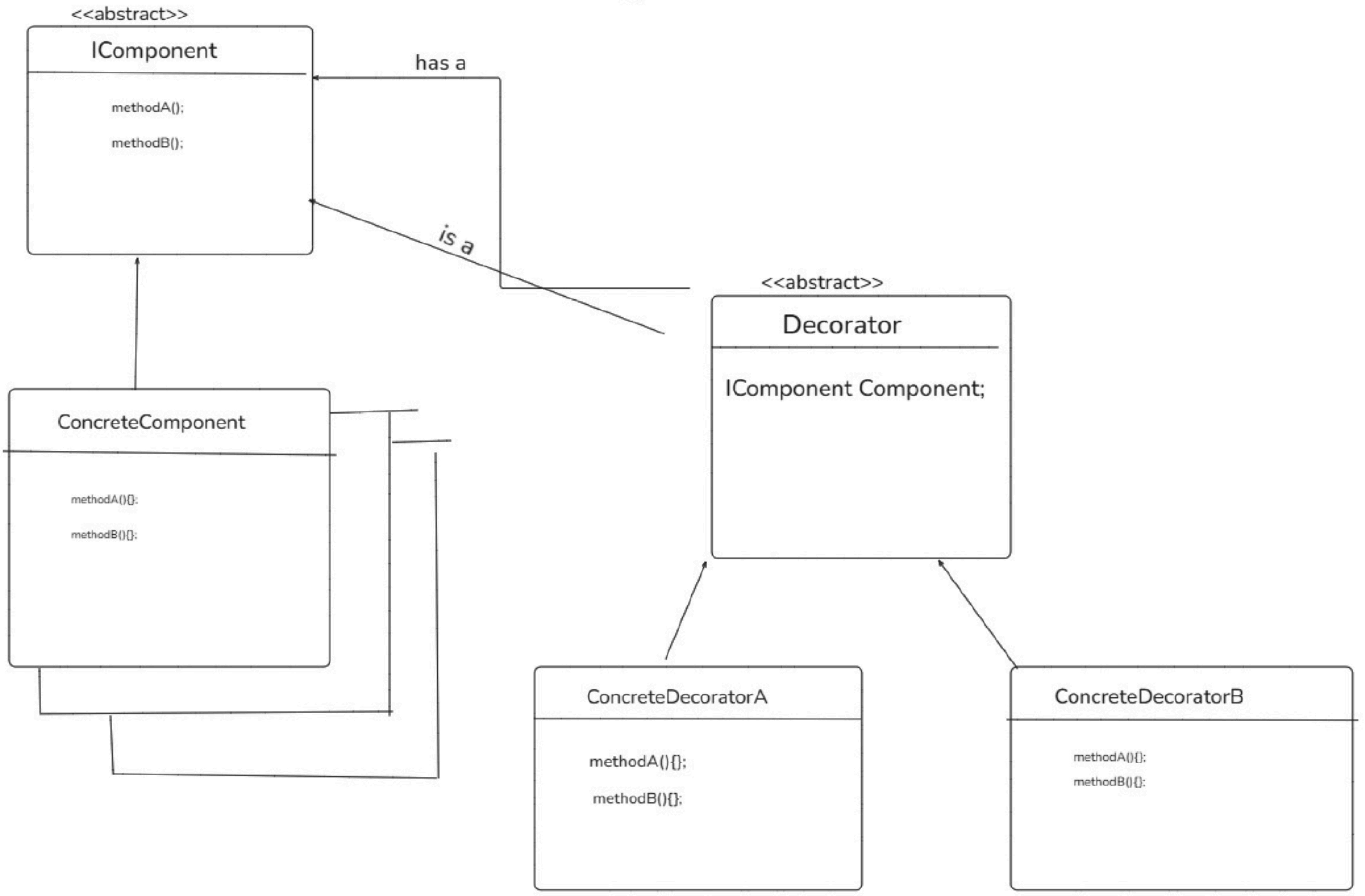
```
ICharacter mario = new HeightUpDecorator(new  Mario());
 mario->getAbilities()
```

Here we're using Recursion

```
ICharacter mario = new StarPower(new GunPower(new HeightUp( new Mario())));
```



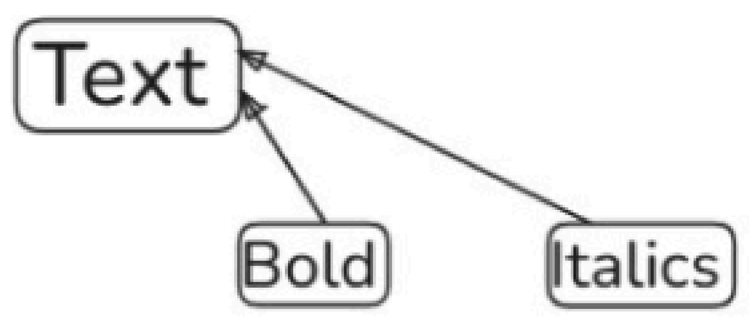
Standard UML Diagram



Definition

Decortaor pattern attaches additional responsibilities to an object dynamically .
Decorator provides a flexible alternative to subclassing for extending functionality.

RealWorld Use Case



Document Editor we can have text in varieties Bold, Italics. We can do with decorate pattern.

- Suppose we have a form in frontend and need to check valid at backend for email. EmailCheck Decorator pattern used.