Universidade Tecnológica Federal do Paraná

Engenharia de Software - Curso de Arquitetura de Software (AS27S)

INSTRUTOR: Prof. Dr. Gustavo Santos

Kamila Antunes de Souza Neves, 2204690

CCH – Padrão de Projeto Estrutural Bridge – Spider-Man

Problema

O padrão de projeto Bridge é um padrão de criação que tem como objetivo separar uma abstração de sua implementação, permitindo que ambas possam variar independentemente. Isso possibilita a criação de estruturas flexíveis, onde mudanças na implementação não afetam a interface e vice-versa. O padrão Bridge promove a composição ao invés da herança, proporcionando maior flexibilidade e extensibilidade ao sistema.

Descrição da Solução

O objetivo deste exemplo é representar o padrão de projeto estrutural do tipo Bridge usando o contexto do Spider-Man. O padrão de projeto Bridge permite separar a abstração (**SuperHeroi**) da implementação (**SuperPoderes**), permitindo que variem independentemente.

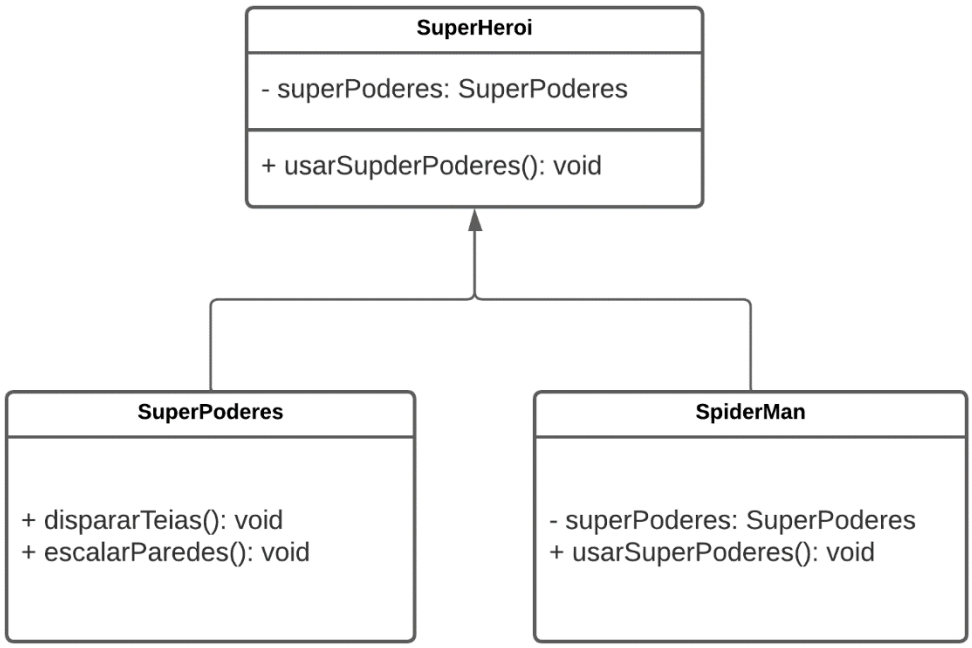
No código, a classe **SuperPoderes** representa a implementação concreta dos superpoderes do Spider-Man, como disparar teias e escalar paredes. A classe **SuperHeroi** representa a abstração do Spider-Man e possui uma referência aos superpoderes através do parâmetro **superPoderes** em seu construtor.

A classe **SpiderMan** é uma implementação concreta da abstração **SuperHeroi** e implementa o método **usarSuperPoderes()**. Nesse método, a implementação correta dos superpoderes é utilizada para disparar teias e escalar paredes.

Por fim, criamos uma instância de **SuperPoderes**, em seguida criamos uma instância de **SpiderMan**, passando a instância de **SuperPoderes** como argumento. Conseguinte, chamos o método **usarSuperPoderes()** na instância de **HomemAranha**, que delegará as chamadas aos métodos correspondente na implementação de **SuperPoderes**. Isso resultará nas mensagens “Spider-Man está disparando teias!” e “Spider-Man está escalando paredes!”.

Ao utilizar esse padrão, podemos adicionar novas implementação de superpoderes sem modificar a abstração ou outras implementações concretas existentes. Isso oferece maior flexibilidade e extensibilidade ao sistema, pois podemos adicionar diferentes tipos de superpoderes sem afetar a lógica do personagem.

Visão Geral [exemplo]



- **SuperHeroi** – classe abstrata que represente o super-herói genérico, contendo o atributo **superpoderes** que é do tipo **SuperPoderes**, fazendo referência à implementação. Possui o método abstrato **usarSuperPoderes()**, que será implementado pelas classes concretas.
- **SuperPoderes** – interface que define os métodos de superpoderes disponíveis. Nesse caso, os métodos **dispararTeias()** e **escalarParedes()**.
- **SpiderMan** – classe concrete que herda de **SuperHeroi** e implementa o método **usarSuperPoderes**. Faz referência ao objeto **superpoderes** do tipo **SuperPoderes**, que será passado no momento da criação do objeto.

Exemplo de Código JavaScript

O código pode ser acessado através do repositório disponibilizado através do link – CCH 2 Padrões de projetos estruturais.

Consequências [vantagens e desvantagens]

- Vantagens
    - Separação entre abstração e implementação: permitindo assim que possam variar independentemente. Oferecendo maior flexibilidade e extensibilidade ao sistema, facilitando a introdução de novas abstrações e implementações sem afetar as outras.
    - Reduz do acoplamento: o padrão de projeto bridge promove a composição em vez da herança, reduzindo o acoplamento entre a abstração e a implementação. Isso significa que mudanças em uma das partes não afetarão a outra, tornando o código mais modular e facilitando a manutenção.
    - Suporte a evolução e extensibilidade: permite adicionar novas implementações independentemente das abstrações existentes e vice-versa.
- Desvantagens
    - Complexidade adicional: este padrão de projeto pode adicionar complexidade ao código, uma vez que envolve a criação de classes adicionais para

representar as abstrações e implementações. Podendo tornar o código mais difícil de entender e manter.

- Sobrecarga de código: o padrão bridge pode levar uma maior quantidade de classes e interfaces, o que pode resultar em um código mais extenso. Aumentando a curva de aprendizado e dificultando a compreensão do sistema como um todo.