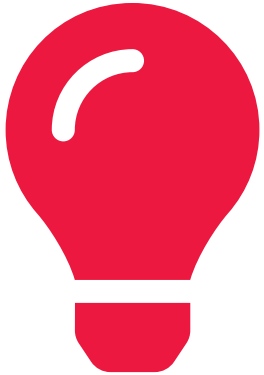
Proxy pattern

Certified Developer
The Ultimate Tech Degree

DigitalHouse >
Coding School

Vamos ver como podemos controlar o acesso a um objeto, permitindo que algo seja feito antes ou depois que a solicitação o atinge.



Propósito e solução

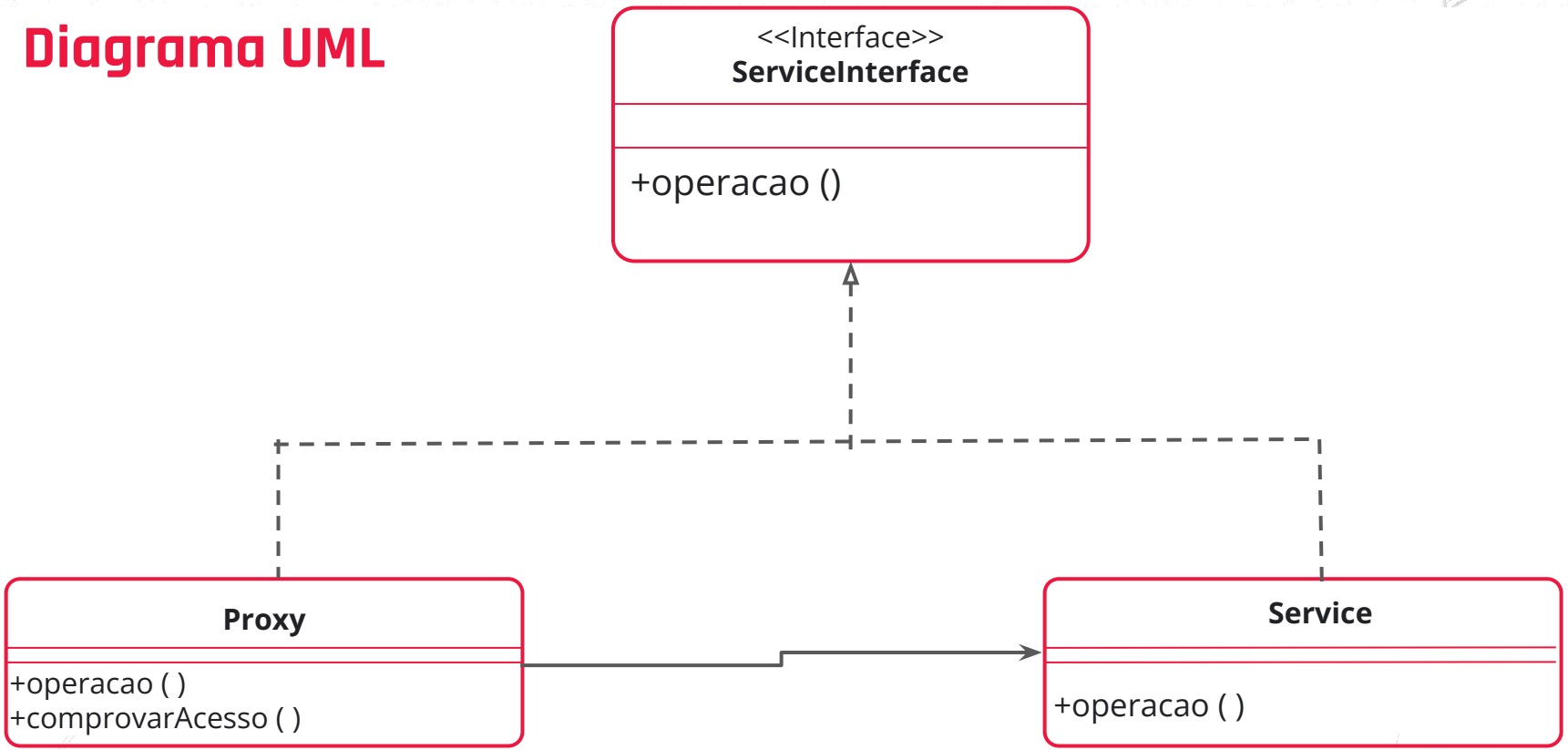
Propósito

Seu objetivo é desenvolver a função de ser um intermediário que agrega funcionalidade a uma classe, sem tocar nela.

Solução

Definir uma classe Proxy com a mesma interface do objeto de serviço original. Posteriormente, nosso aplicativo deve ser atualizado para que os clientes se comuniquem com o proxy e não com o serviço de destino. Ao receber um pedido de um cliente, o proxy irá encaminhá-lo para o serviço, mas como intermediário poderemos realizar operações antes ou depois de direcionar a solicitação.

Diagrama UML



Vantagens

O proxy funciona mesmo se o objeto de serviço não estiver pronto ou disponível.

Princípio de aberto/fechado: podemos introduzir novos proxies sem alterar o serviço ou clientes.

Desvantagens

Ao adicionar mais uma camada entre o cliente e o serviço real, a resposta pode ser atrasada.


DigitalHouse>
Coding School


Certified Tech
Developer
The Ultimate Degree