

WZORCE PROJEKTOWE

WZORCE
KREACYJNE

WZORCE
STRUKTURALNE

WZORCE
CZYNNOŚCIOWE

WZORCE KREACYJNE

Singleton Pattern

Zapewnia, że klasa posiada
pojedynczą instancję

ConfigManager

Get()

Set()

GetInstance()

ConfigManager

Get()

Set()

- ConfigManager()

ConfigManager

Get()

Set()

- ConfigManager()

- instance

ConfigManager

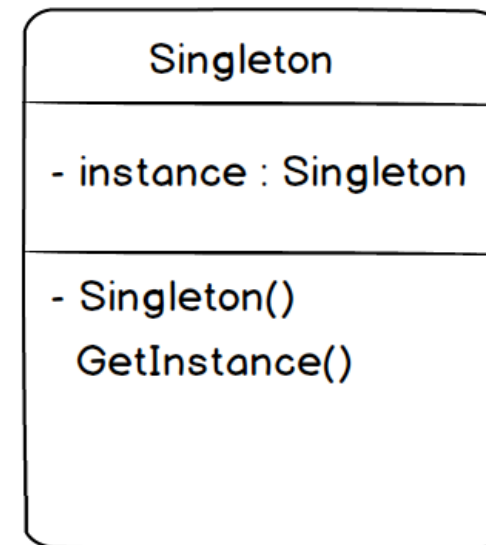
Get()

Set()

- ConfigManager()

- instance

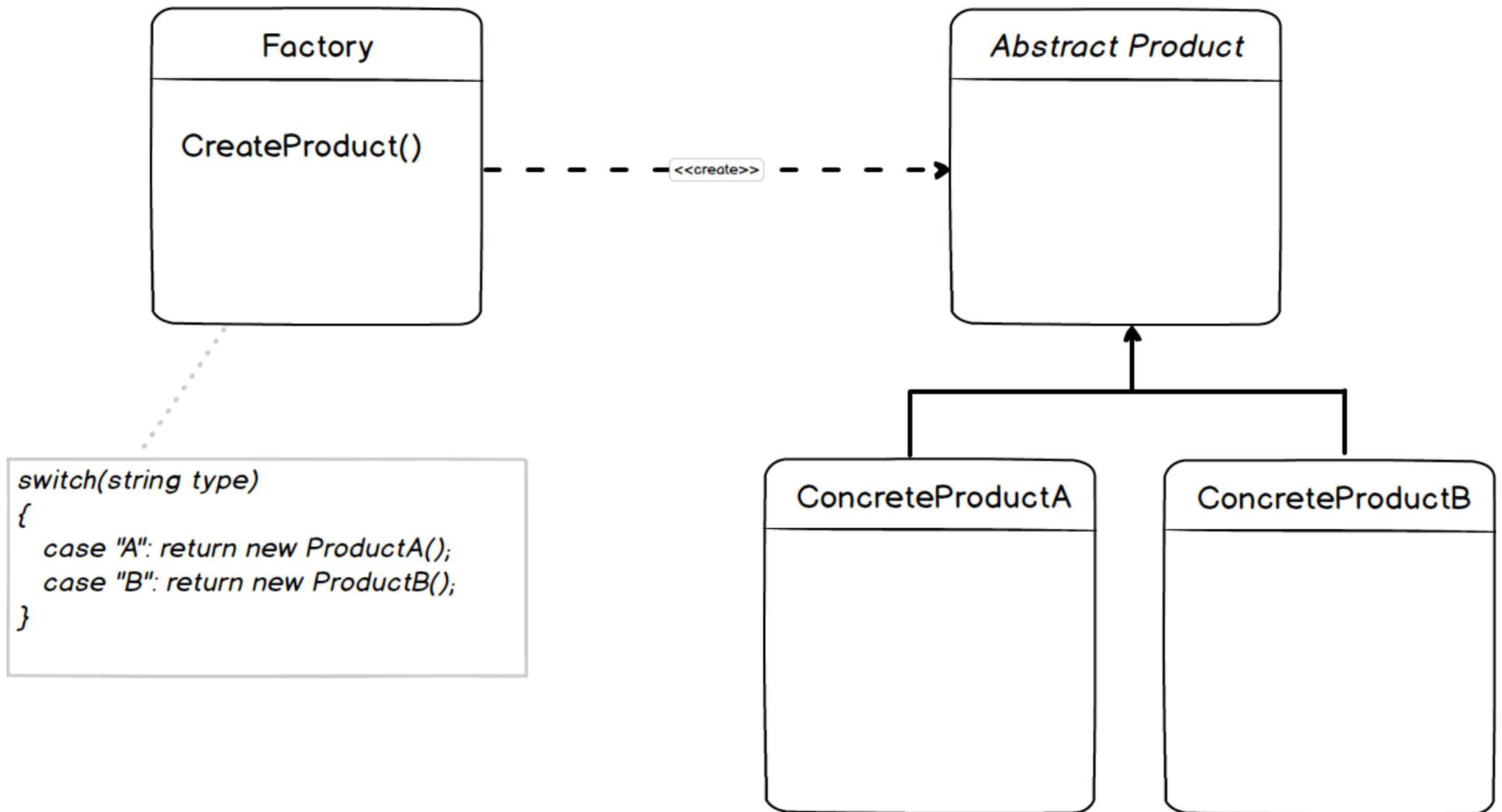
GetInstance()



Pozwala zachować pewność, że istnieje wyłącznie **jedna instancja** danej klasy.

Simple Factory Pattern

Zapewnia tworzenie obiektu na podstawie wielu warunków



Abstract Factory Pattern

Zapewnia interfejs do tworzenia
rodziny powiązanych obiektów

INTERFACE

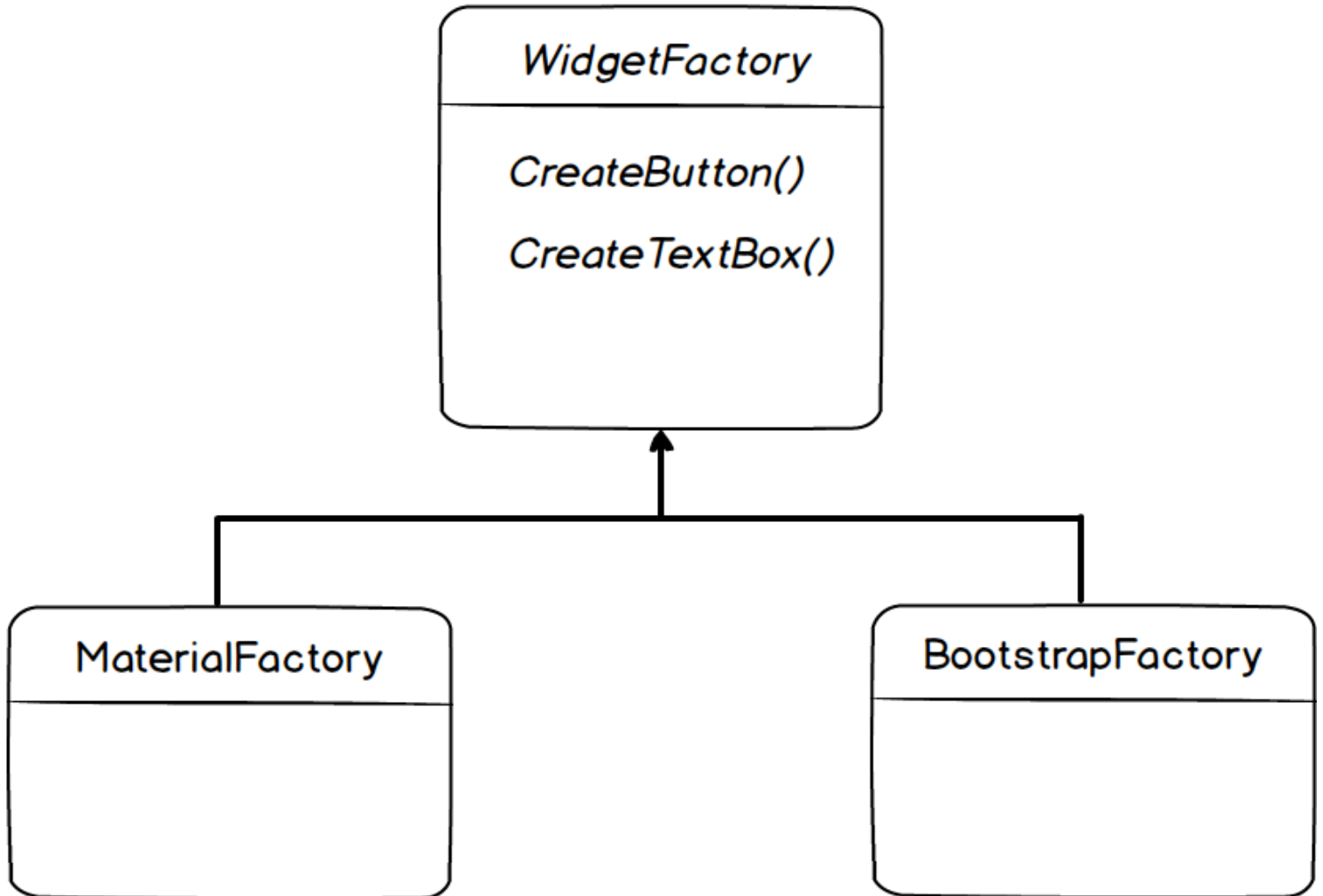
WidgetFactory

CreateButton()

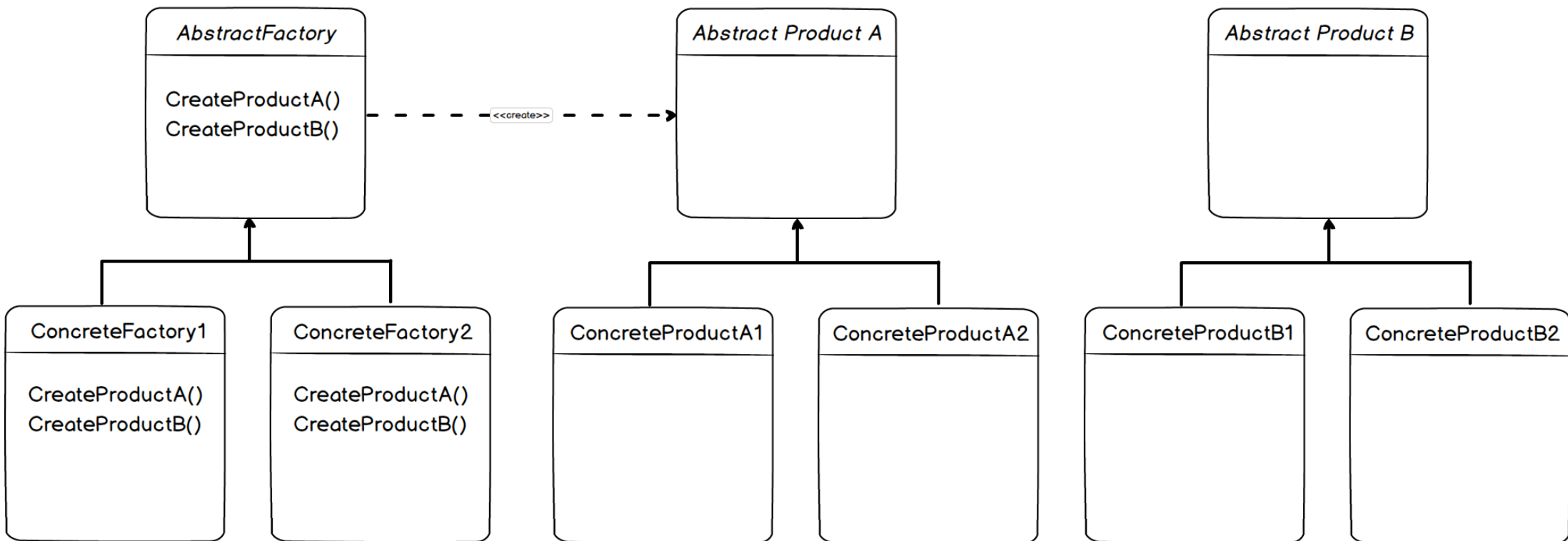
CreateTextBox()

MaterialFactory

BootstrapFactory



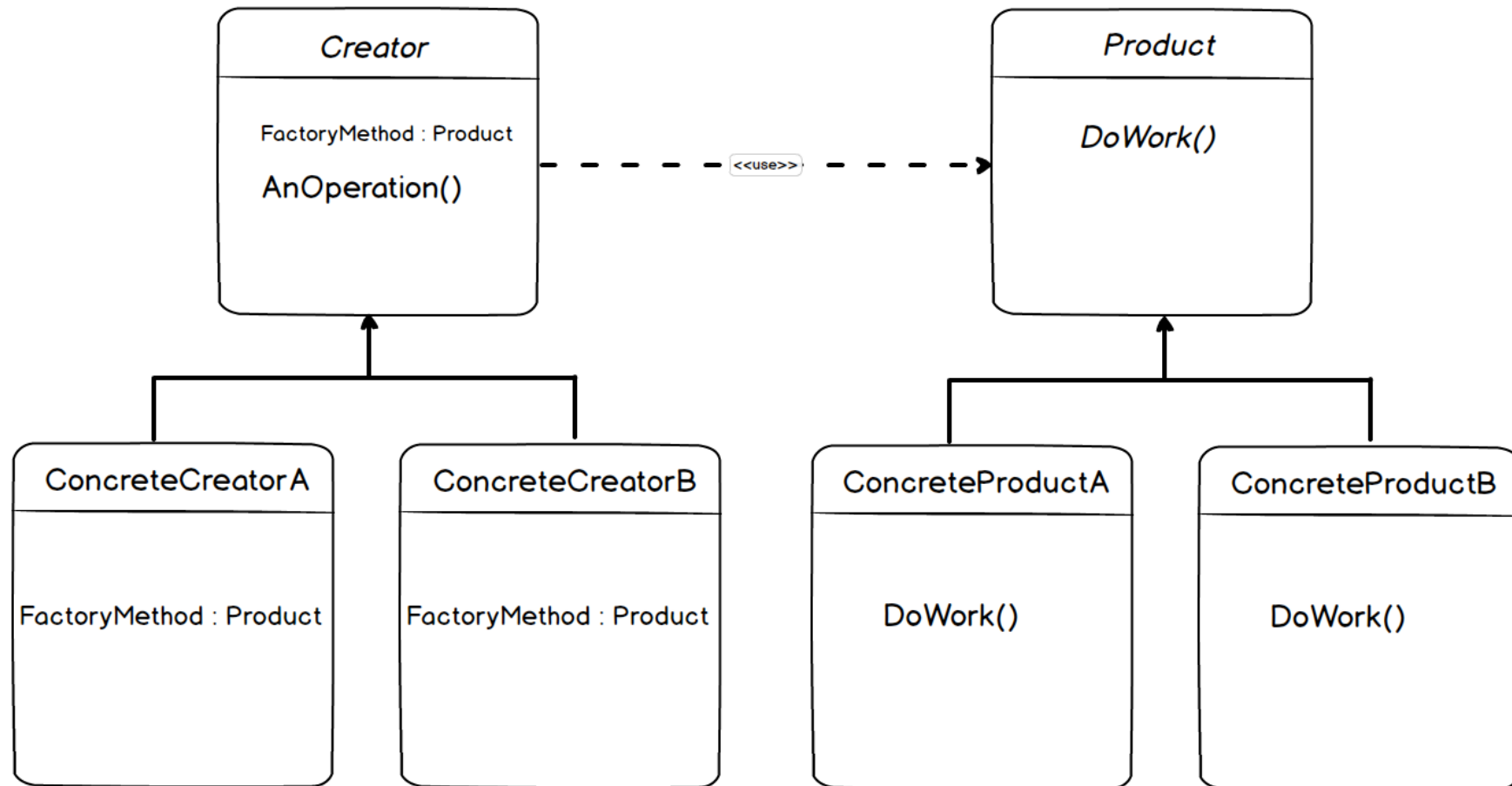
INTERFACE



Factory Method Pattern

Odkłada tworzenie obiektu
do **podklas**

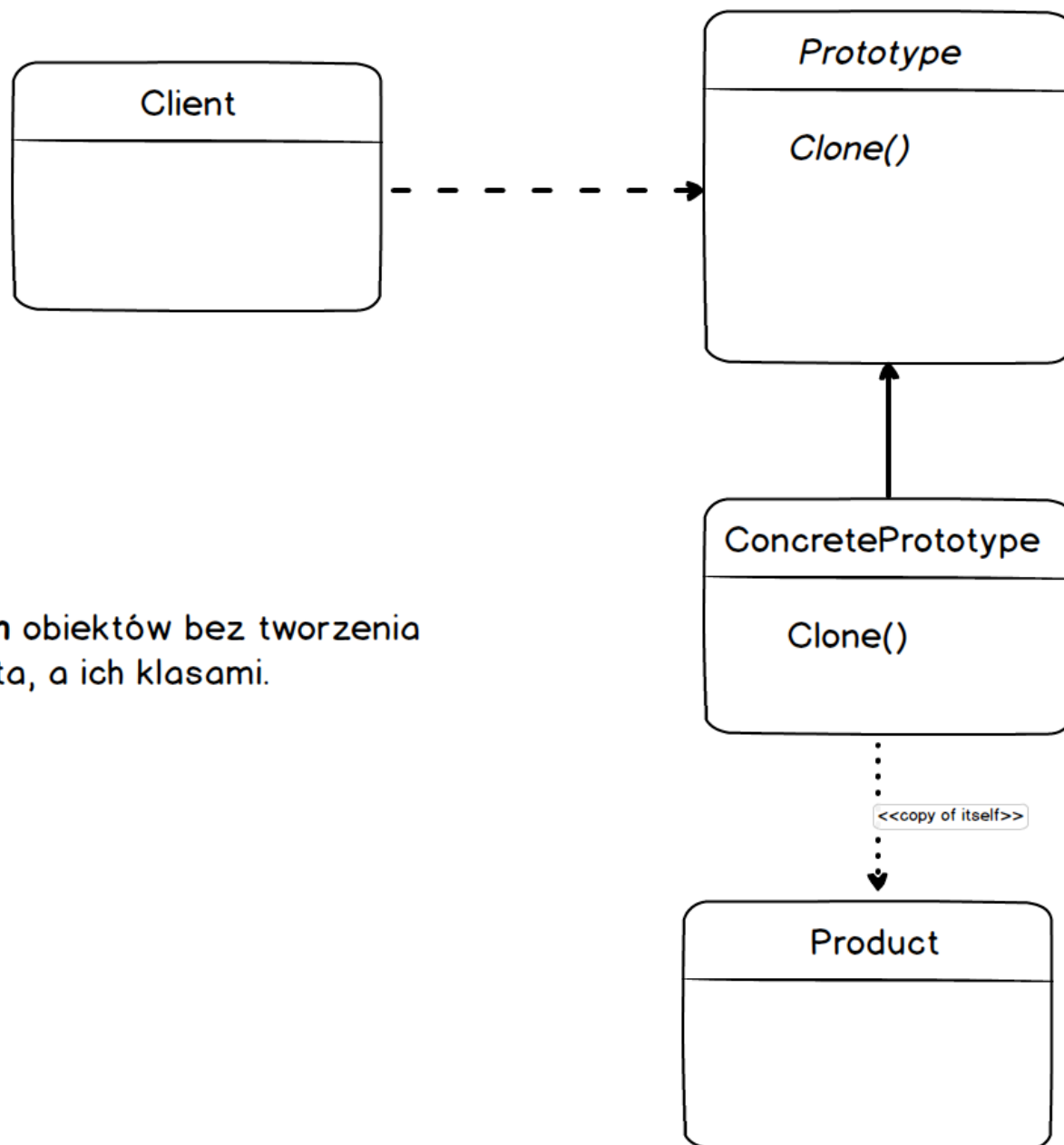
INTERFACE



Prototype Pattern

Tworzy nowe obiekty
kopiując istniejący obiekt

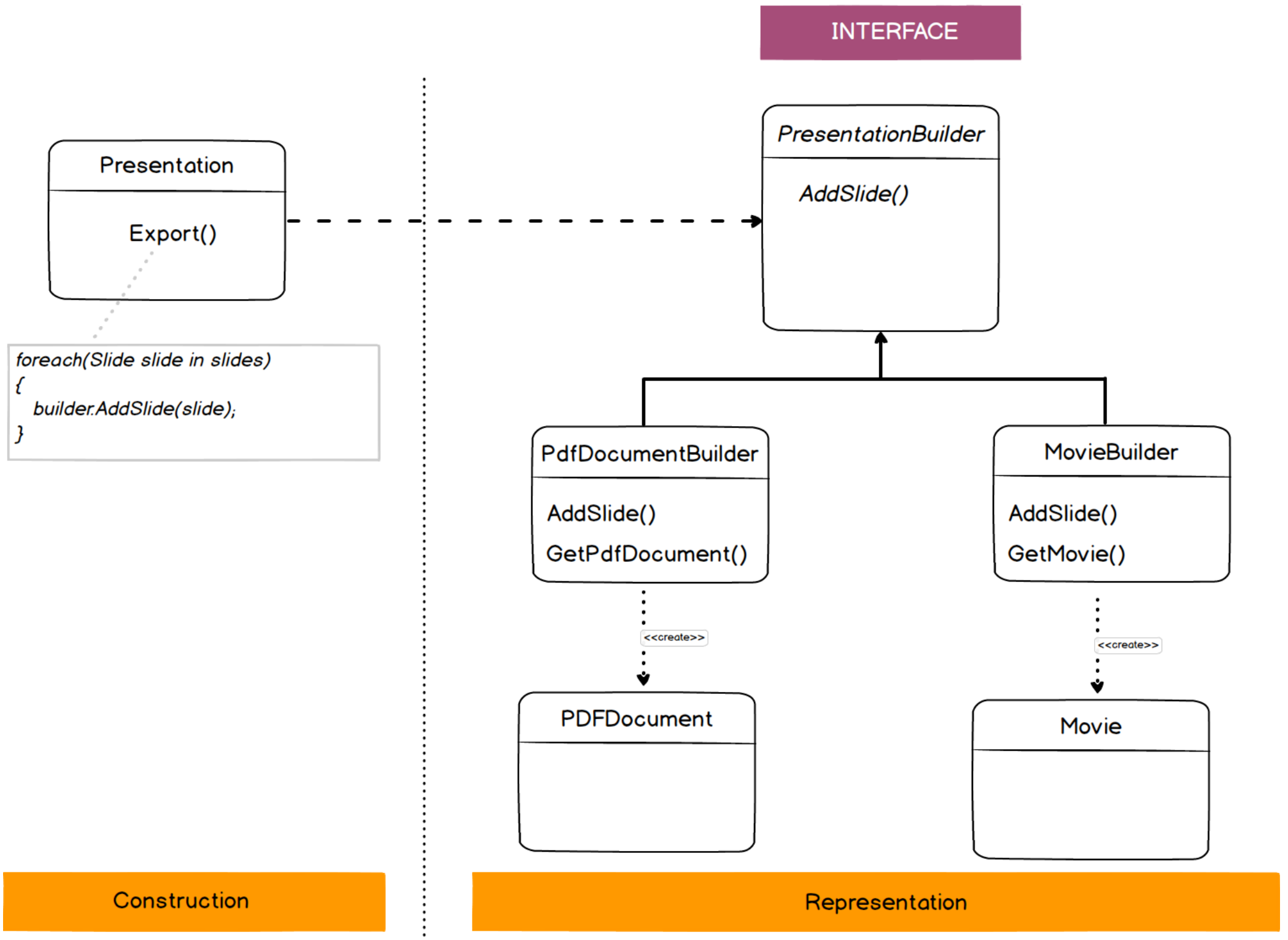
INTERFACE



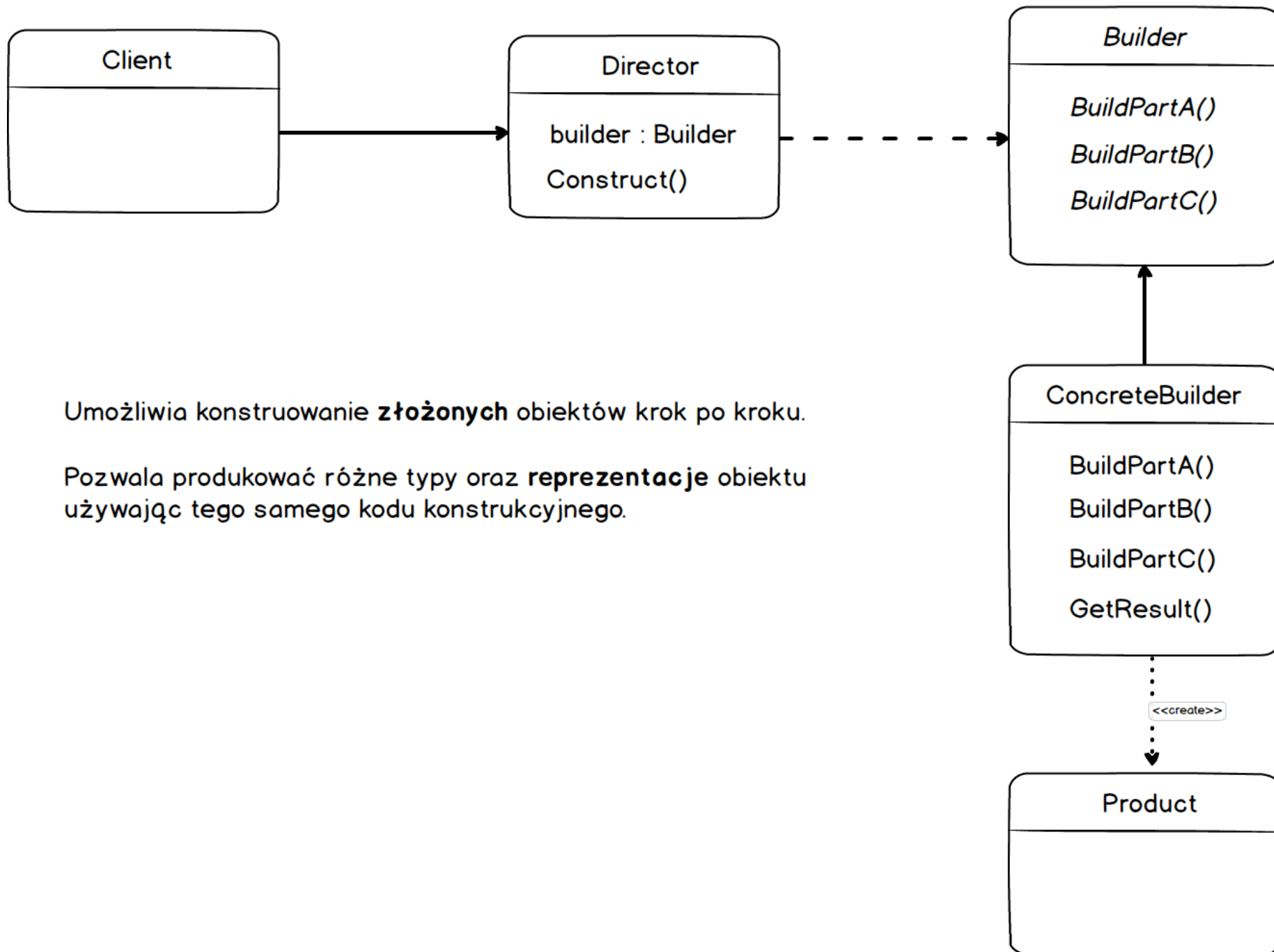
Umożliwia kopiowanie **istniejących** obiektów bez tworzenia zależności pomiędzy kodem klienta, a ich klasami.

Builder Pattern

Oddziela konstrukcję obiektu
od jego reprezentacji



INTERFACE



Umożliwia konstruowanie **złożonych** obiektów krok po kroku.

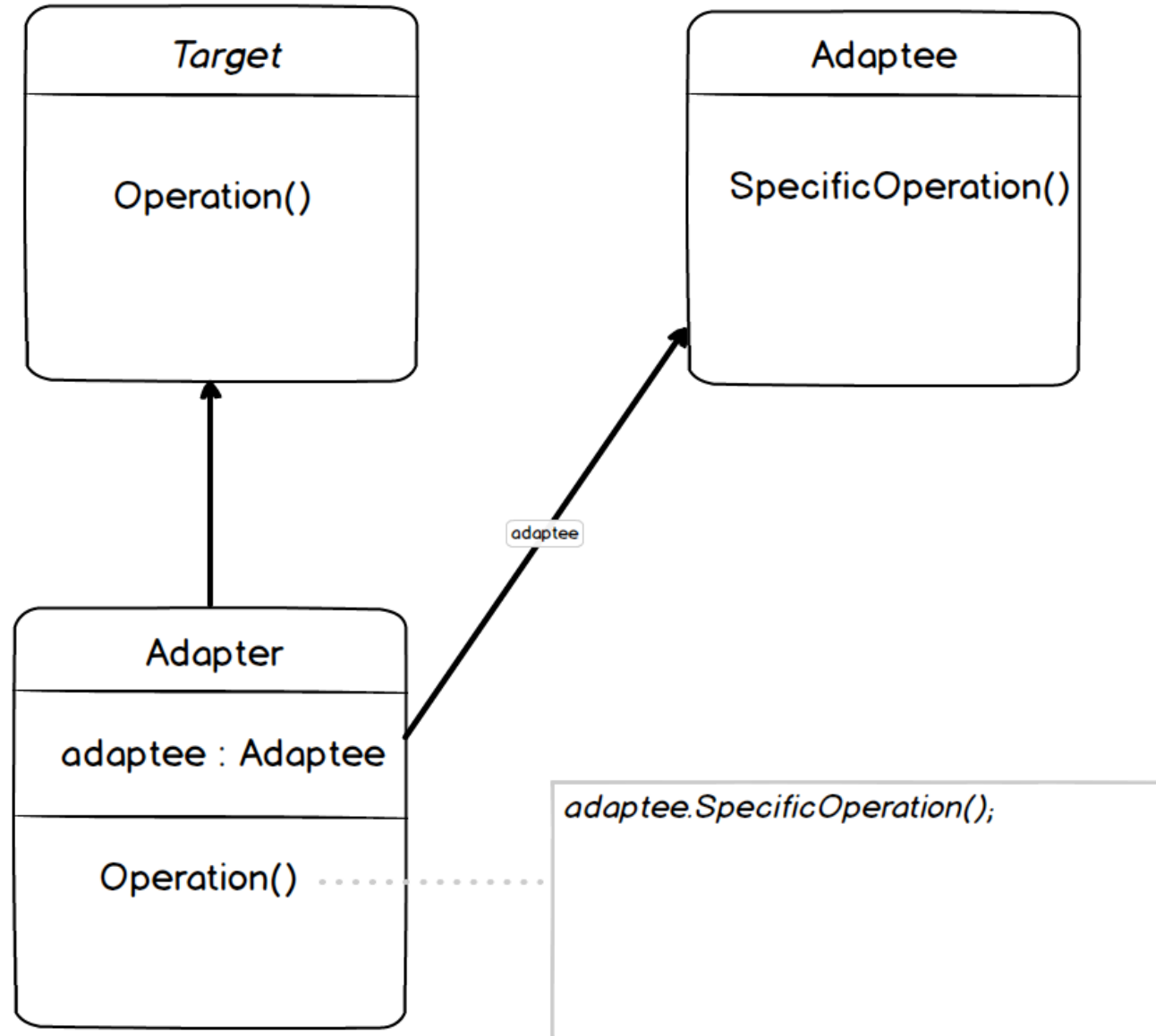
Pozwala produkować różne typy oraz **reprezentacje** obiektu używając tego samego kodu konstrukcyjnego.

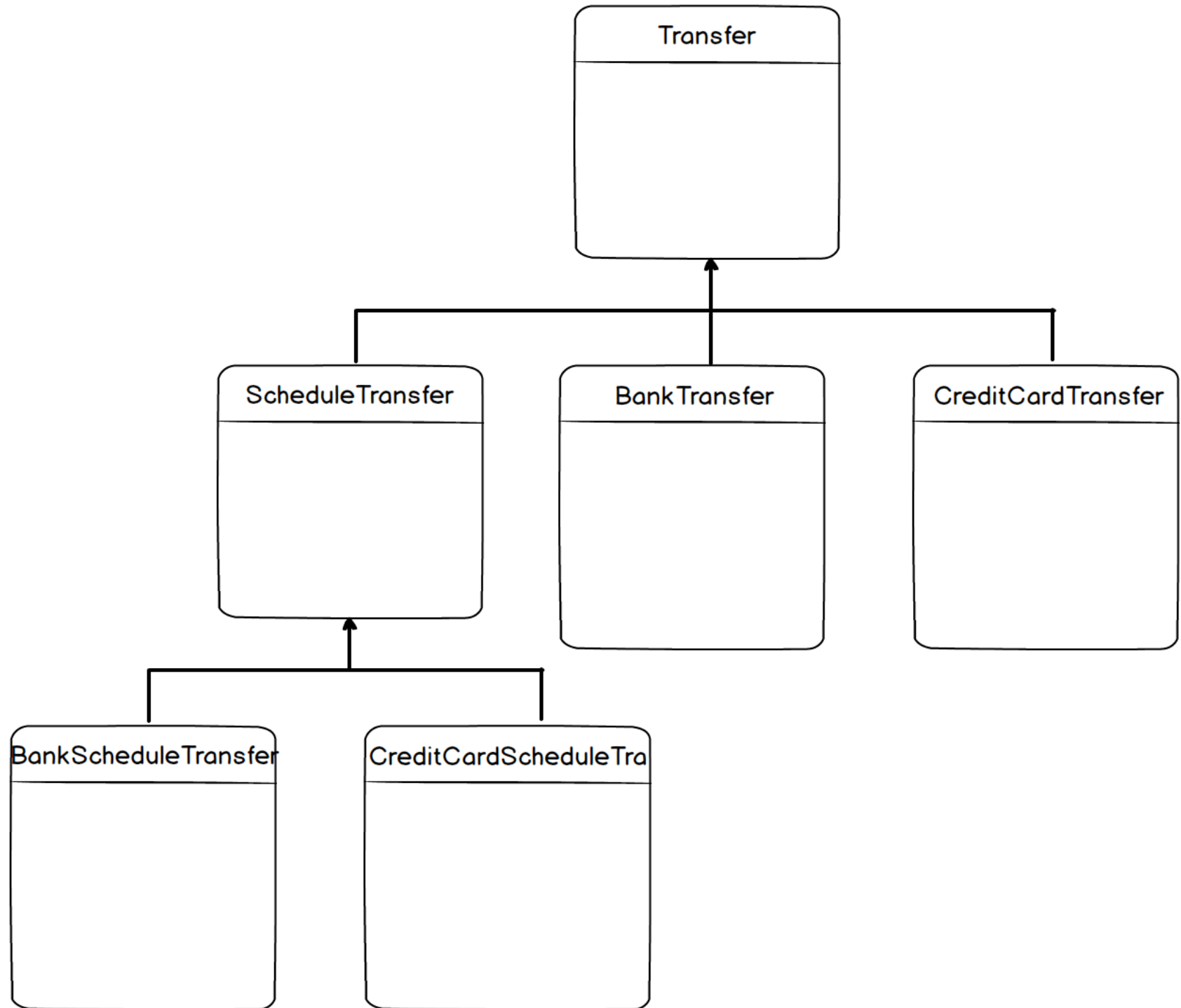
WZORCE STRUKTURALNE

Adapter Pattern

Pozwala na współdziałanie ze sobą
obiektów o niekompatybilnych interfejsach

Object Adapter





FEATURE

Transfer

ScheduleTransfer

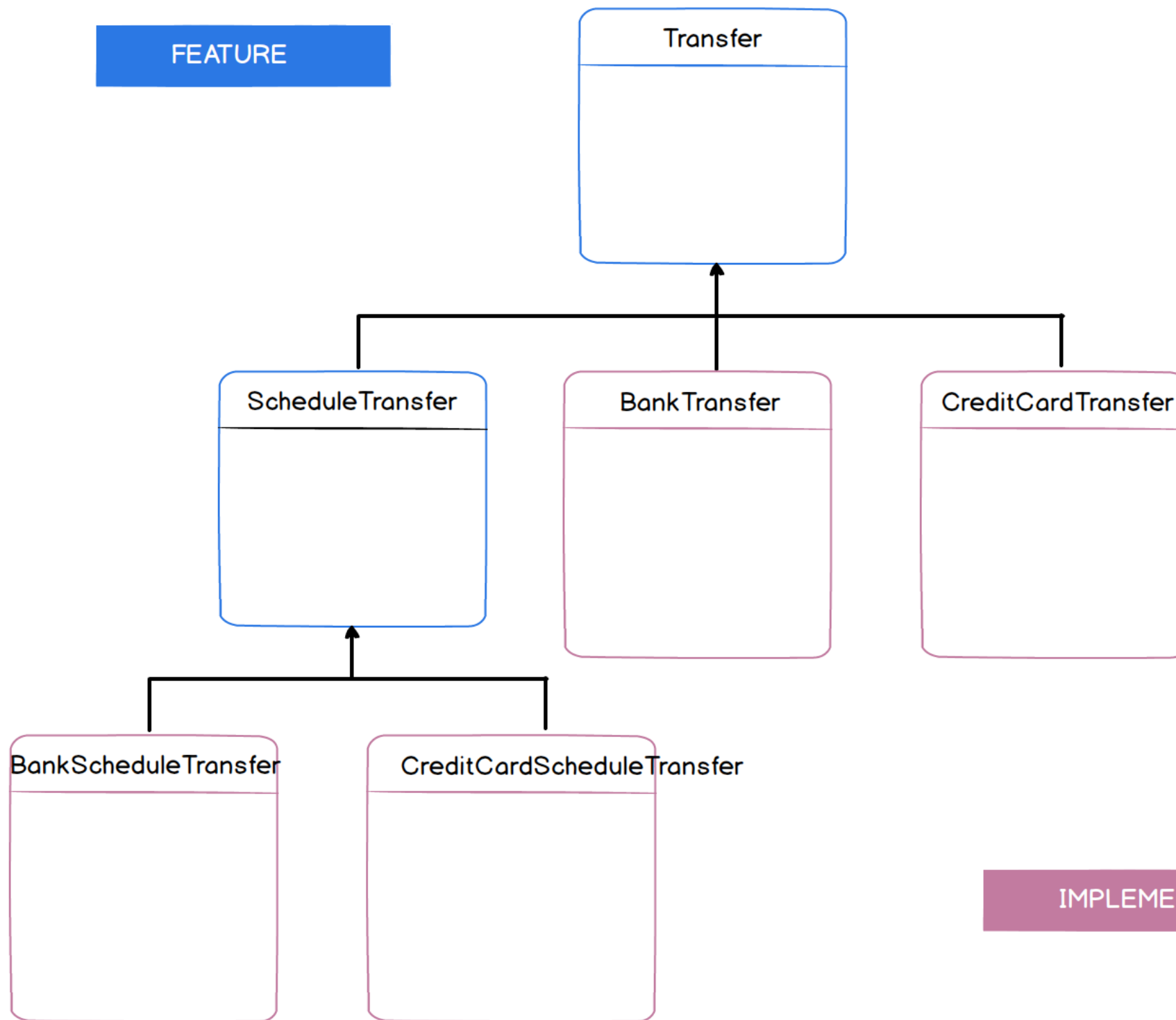
BankTransfer

CreditCardTransfer

BankScheduleTransfer

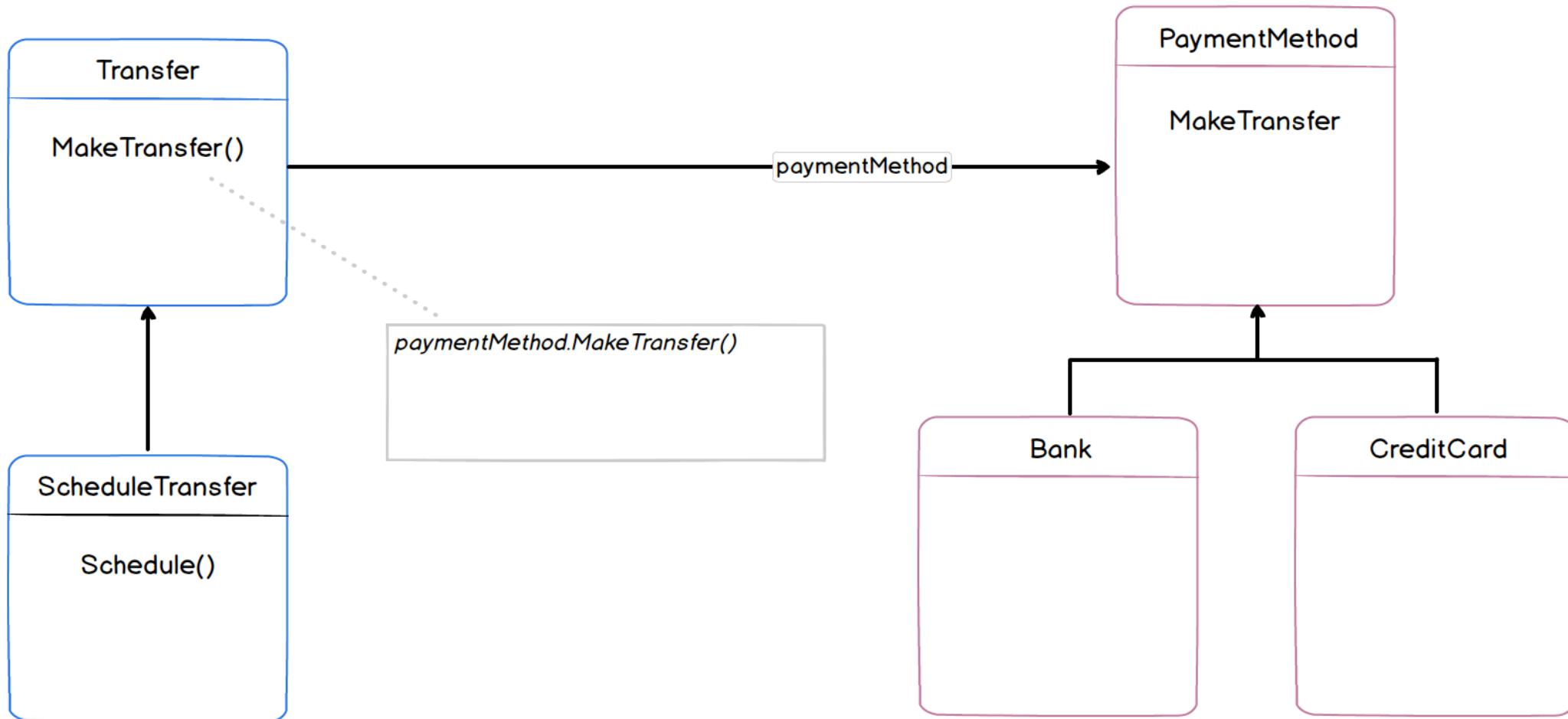
CreditCardScheduleTransfer

IMPLEMENTATION



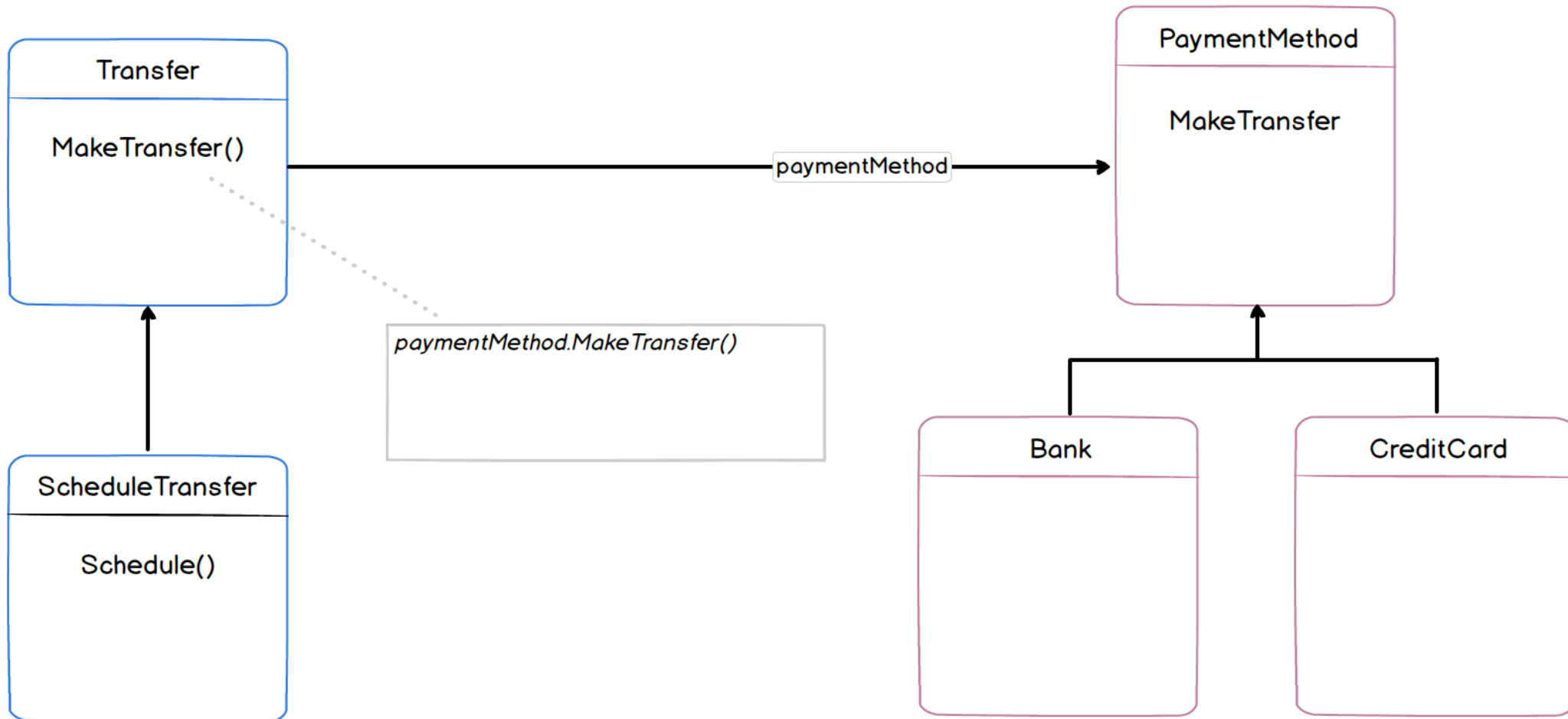
FEATURE

IMPLEMENTATION



FEATURE

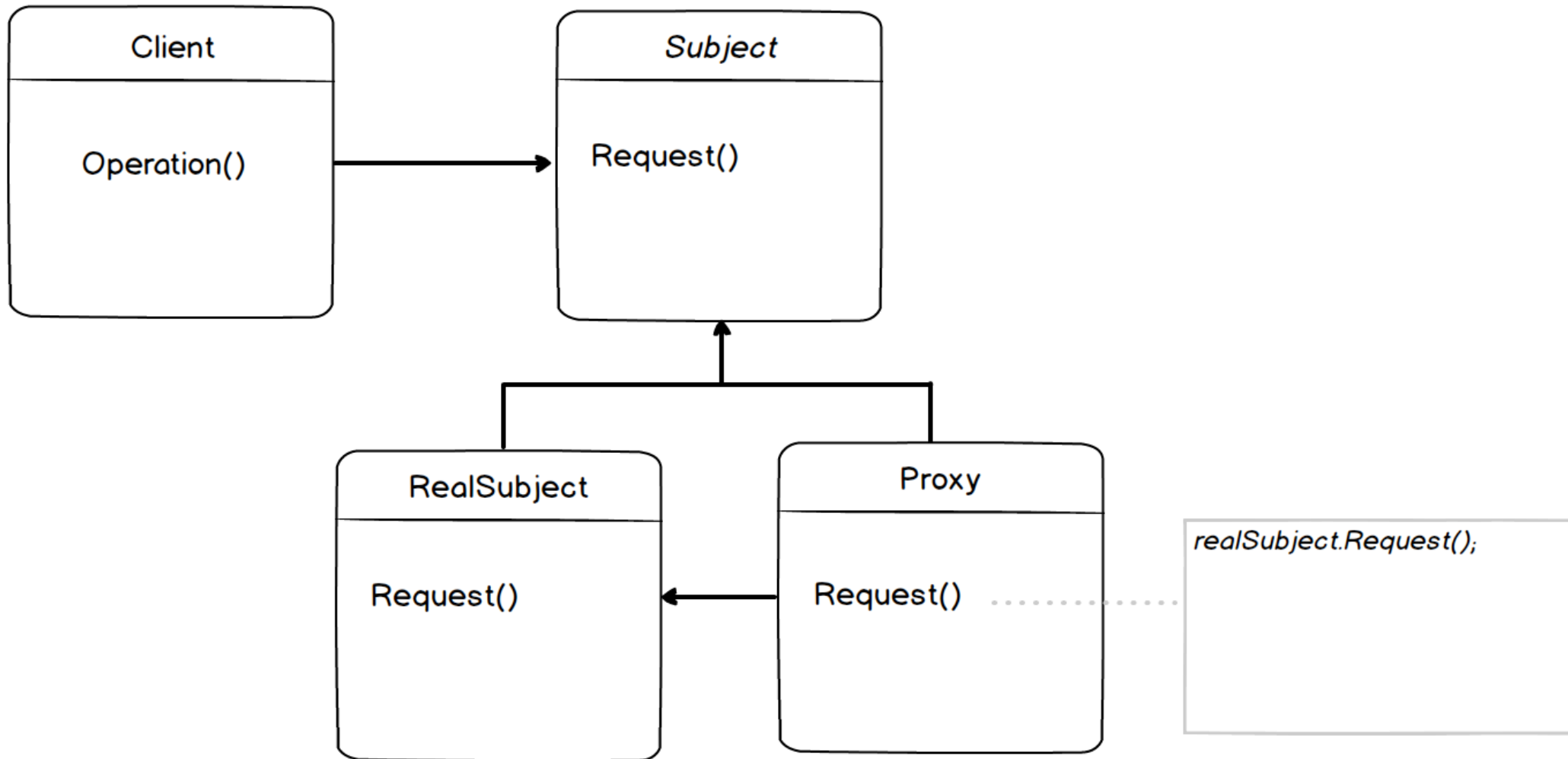
IMPLEMENTATION



Proxy Pattern

Pozwala na stworzenie obiektu zastępczego
w miejsce innego obiektu.

INTERFACE



Decorator Pattern

Przedkładaj kompozycję nad dziedziczenie

WZORCE CZYNNOŚCIOWE

Fasada Pattern

Ukrywa złożony zestaw klas
w uproszczony interfejs.

Chain of Responsibility Pattern

Pozwala przekazywać żądania wzdłuż
łańcucha obiektów obsługujących

Composite Pattern

Komponuje obiekty w
struktury drzewiaste

State Pattern

Pozwala obiektowi zmieniać swoje zachowanie
gdy zmieni się jego **stan wewnętrzny**

Flyweight Pattern

Pozwala zmieścić więcej obiektów w pamięci poprzez współdzielenie części opisu ich stanów

