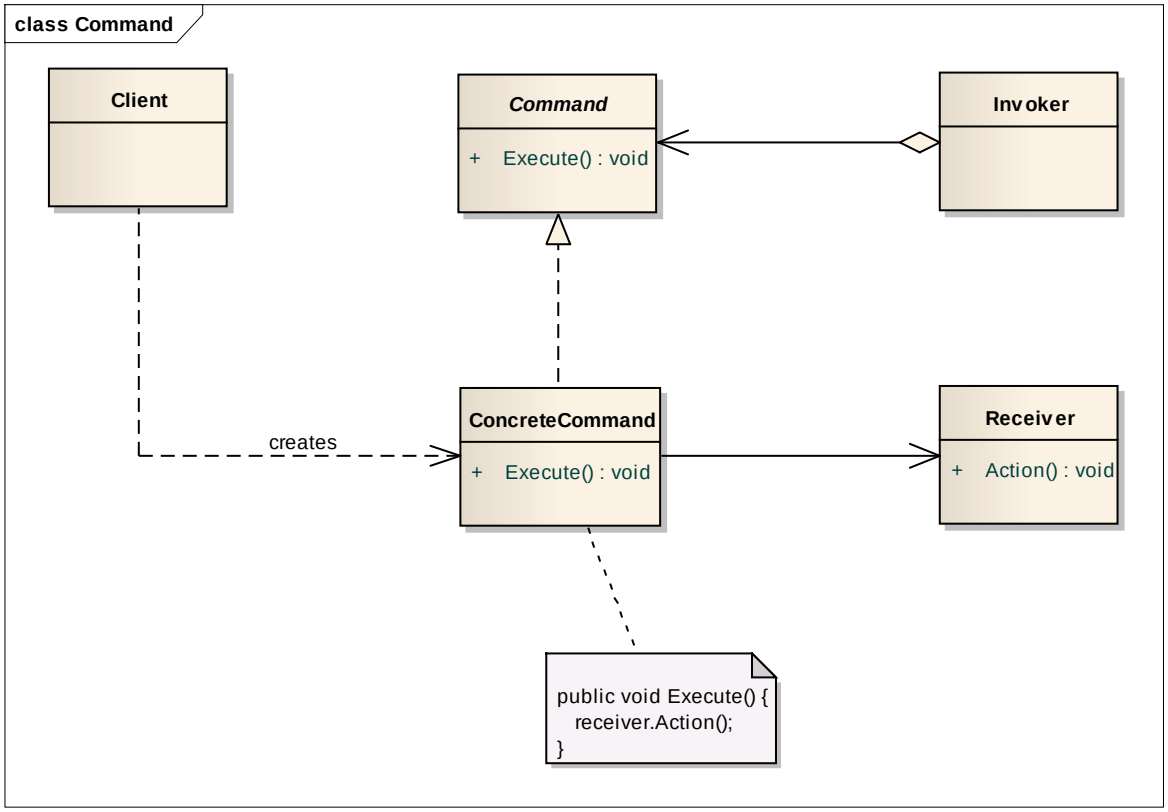
Projektowanie obiektowe oprogramowania
Wykład 9 – wzorce czynnościowe (3)
Wiktor Zychla 2012

1 Command

Motto: kapsułkowanie żądań w postaci obiektów o jednolitym interfejsie dostępu, oddziela wywołującego (Invoker) od odbiorcy (Receiver)
Uwaga: prostszą alternatywą byłoby zapamiętanie funkcji zwrotnej, ale komenda może mieć szerszy interfejs niż tylko Invoke() (np. Undo(), Log() itd.)

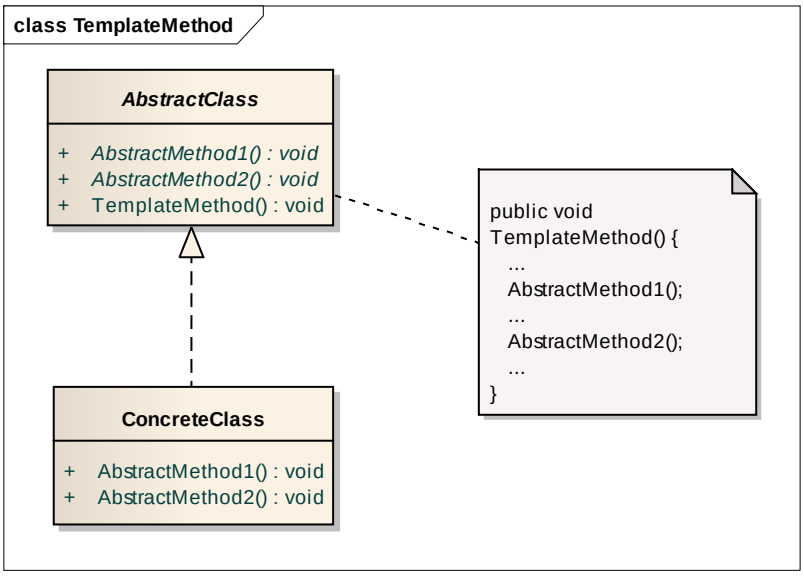


Typowe niedostatki implementacyjne:

- Konkretne polecenie nie deleguje wykonania do odbiorcy, tylko całą logikę implementuje samo
- Konkretne polecenie nie zawiera żadnego stanu, jest tylko delegatorem wywołania do odbiorcy (czy to jest problem ?)

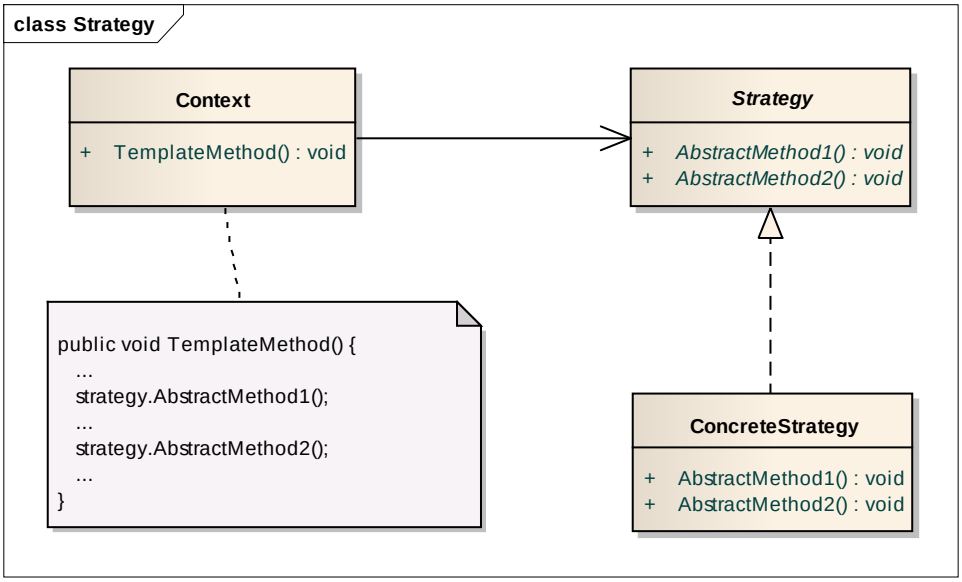
2 Template Method

Motto: określ szkielet algorytmu i zrzucając odpowiedzialność za implementację szczegółów do podklasy
Kojarzyć z: Template Method = Strategy przez dziedziczenie



3 Strategy

Motto: określ szkielet algorytmu i zrzucając odpowiedzialność za implementację szczegółów do klasy, do której delegujesz żądania
Kojarzyć z: Strategy = Template Method przez delegację



Template Method vs Strategy

- Template Method wiąże algorytm z konkretną hierarchią klas, Strategy jest implementowane przez interfejs + delegację – można powiedzieć że to jest „nowocześniej” („*prefer compostion over inheritance*”)
- Template Method może oznaczać mniej kodu, bo klasa przeciąży tylko metody które chce uszczegółowić

Przykład Strategy: Array.Sort, korzysta z IComparable/IComarer

Przykład Template Method: Microsoft.XNA.Framework.Game

4 State

Motto: maszyna stanowa
Uwaga: zbyt rzadko stosowana (można częściej niż się wydaje)
Alternatywne zastosowanie: zarządzanie złożonym GUI