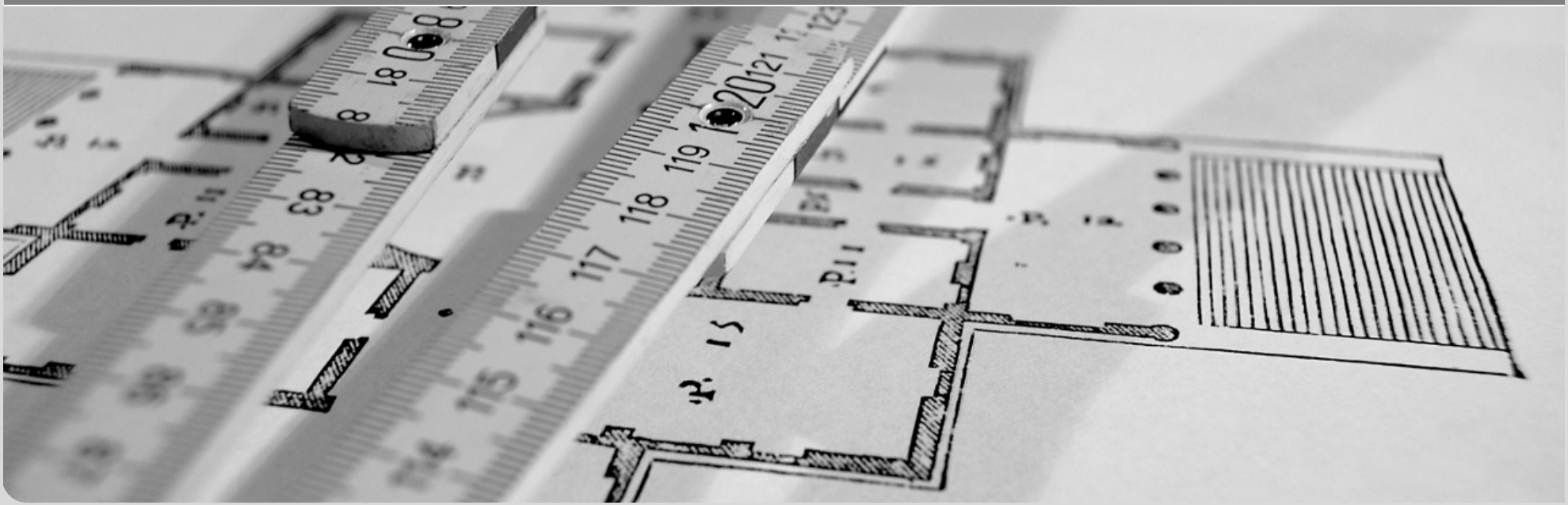
Karlsruher Institut für Technologie

Softwaretechnik 1 - 3. Tutorium

Tutorium 03

Felix Bachmann | 12.06.2017

KIT - INSTITUT FÜR PROGRAMMSTRUKTUREN UND DATENORGANISATION (IPD)



2. Übungsblatt Statistik

Häufige Fehler

Allgemein

Häufige Fehler

Aufgabe 1 (Plug-In-Architektur): ∅ von 4 (+ 1)

Häufige Fehler

Aufgabe 2 (Plug-In): ∅ von 4

Häufige Fehler

Aufgabe 3 (iMage-Bundle): ∅ von 2

Häufige Fehler

Aufgabe 4 (Aktivitätsdiagramme (Geometrify)): ∅ von 10

Häufige Fehler

Aufgabe 5 (Sequenzdiagramm (main-Methode)): ∅ von 5

Häufige Fehler

Aufgabe 6 (Substitutionsprinzip): ∅ von 3

Wo sind wir?

- die ersten 2 Phasen des Wasserfallmodells sind geschafft

Wo sind wir?

- die ersten 2 Phasen des Wasserfallmodells sind geschafft
  $\Longrightarrow$ Welche waren das nochmal?

Wo sind wir?

- die ersten 2 Phasen des Wasserfallmodells sind geschafft
  $\Longrightarrow$ Welche waren das nochmal? Planung, Definition!

Wo sind wir?

- die ersten 2 Phasen des Wasserfallmodells sind geschafft
  $\Longrightarrow$ Welche waren das nochmal? Planung, Definition!
  $\Longrightarrow$ Dokumente?

Wo sind wir?

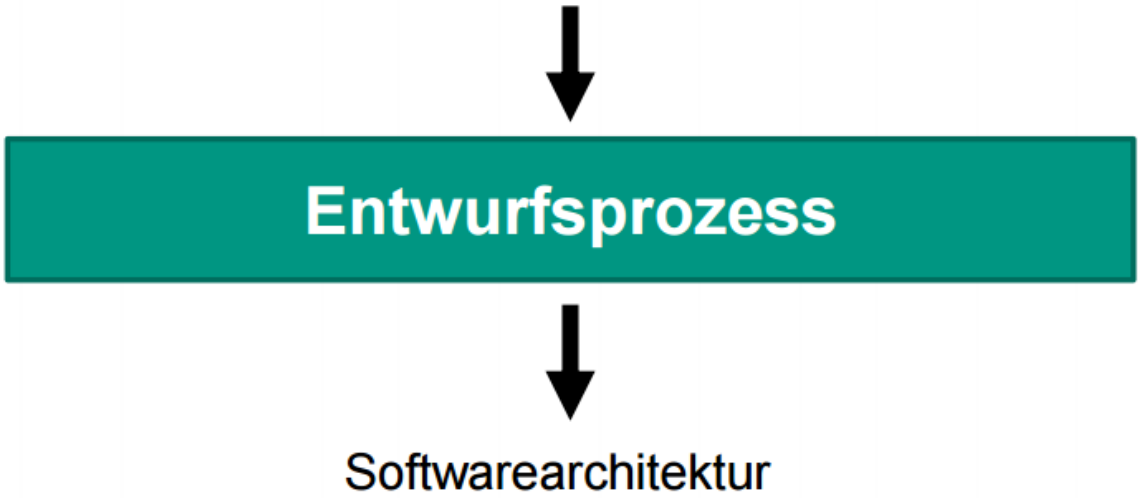
- die ersten 2 Phasen des Wasserfallmodells sind geschafft
  $\Longrightarrow$ Welche waren das nochmal? Planung, Definition!
  $\Longrightarrow$ Dokumente? Lastenheft, Pflichtenheft (+ andere...)

Wo sind wir?

- die ersten 2 Phasen des Wasserfallmodells sind geschafft
  $\Longrightarrow$ Welche waren das nochmal? Planung, Definition!
  $\Longrightarrow$ Dokumente? Lastenheft, Pflichtenheft (+ andere...)
- jetzt: Entwurf!

Wozu Entwurf?

- Pflichtenheft (einschl. Modelle)
- Konzept Benutzungsoberfläche
- Benutzerhandbuch + Hilfekonzept



Softwarearchitektur ist Grundlage für Implementierung!

Abgrenzung Definition vs. Entwurf

- Definition: **Was** ist zu implementieren?

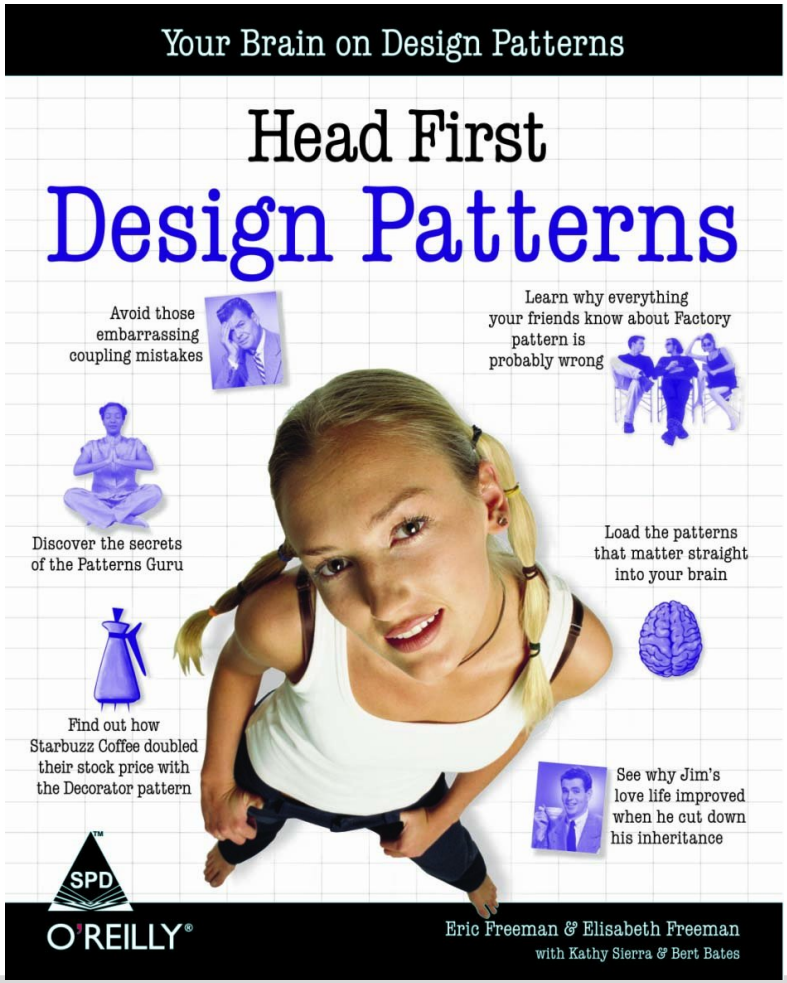
Abgrenzung Definition vs. Entwurf

- Definition: **Was** ist zu implementieren?
- Entwurf: **Wie** ist das System zu implementieren?

Empfehlenswerte Literatur (wirklich!)

knapp 700 Seiten

$\Longrightarrow$ als interaktives Nachschlagewerk, falls man bestimmte Muster nicht versteht



Wozu Entwurfsmuster?

Entwurfsmuster

Ein Software-Entwurfsmuster beschreibt eine Familie von Lösungen für ein Software-Entwurfsproblem.

Wozu Entwurfsmuster?

Entwurfsmuster

Ein Software-Entwurfsmuster beschreibt eine Familie von Lösungen für ein Software-Entwurfsproblem.

- schematische Klassendiagramme zur Lösung von häufig auftretenden Problemen

Wozu Entwurfsmuster?

Entwurfsmuster

Ein Software-Entwurfsmuster beschreibt eine Familie von Lösungen für ein Software-Entwurfsproblem.

- schematische Klassendiagramme zur Lösung von häufig auftretenden Problemen
- Wiederverwendung von Entwurfswissen $\Longrightarrow$ Rad nicht neu erfinden!

Wozu Entwurfsmuster?

Entwurfsmuster

Ein Software-Entwurfsmuster beschreibt eine Familie von Lösungen für ein Software-Entwurfsproblem.

- schematische Klassendiagramme zur Lösung von häufig auftretenden Problemen
- Wiederverwendung von Entwurfswissen $\Longrightarrow$ Rad nicht neu erfinden!

Geheimnisprinzip

Geheimnis- / Kapselungsprinzip

Jedes Modul verbirgt eine wichtige Entwurfsentscheidung hinter einer wohldefinierten Schnittstelle, die sich bei einer Änderung der Entscheidung nicht mit ändert.

Geheimnisprinzip

Geheimnis- / Kapselungsprinzip

Jedes Modul verbirgt eine wichtige Entwurfsentscheidung hinter einer wohldefinierten Schnittstelle, die sich bei einer Änderung der Entscheidung nicht mit ändert.

Sinn?

Geheimnisprinzip

Geheimnis- / Kapselungsprinzip

Jedes Modul verbirgt eine wichtige Entwurfsentscheidung hinter einer wohldefinierten Schnittstelle, die sich bei einer Änderung der Entscheidung nicht mit ändert.

Sinn? $\Longrightarrow$ Änderungen ohne Risiko durchführen

Geheimnisprinzip

Geheimnis- / Kapselungsprinzip

Jedes Modul verbirgt eine wichtige Entwurfsentscheidung hinter einer wohldefinierten Schnittstelle, die sich bei einer Änderung der Entscheidung nicht mit ändert.

Sinn? $\Longrightarrow$ Änderungen ohne Risiko durchführen

Beispiel?

Geheimnisprinzip

Geheimnis- / Kapselungsprinzip

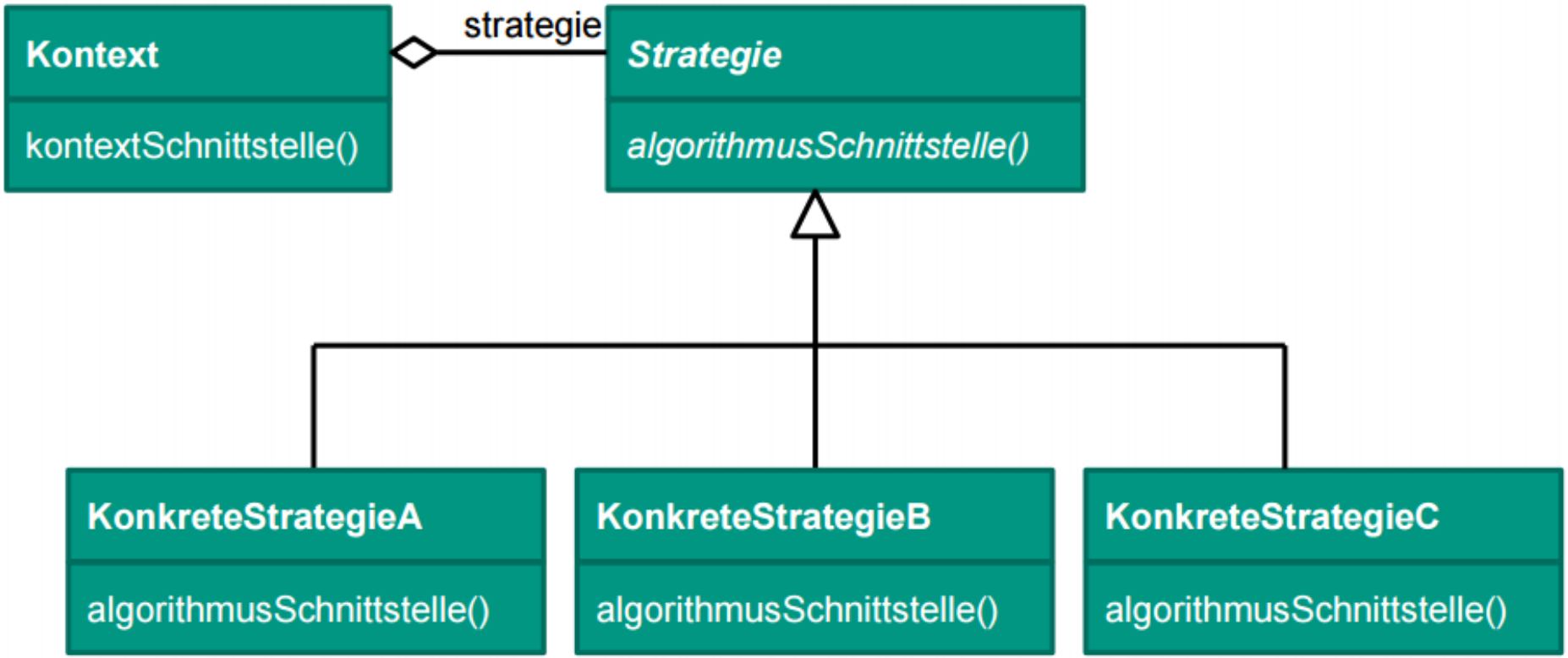
Jedes Modul verbirgt eine wichtige Entwurfsentscheidung hinter einer wohldefinierten Schnittstelle, die sich bei einer Änderung der Entscheidung nicht mit ändert.

Sinn? $\Longrightarrow$ Änderungen ohne Risiko durchführen

Beispiel? $\Longrightarrow$ private Attribute mit get()- und set()-Methoden

Vorgriff: Entwurfsmuster Strategie

- Ziel: Algorithmen kapseln, austauschbar machen
- wird in vielen Entwurfsmustern verwendet



Kategorien der Entwurfsmuster

- **Entkopplungs-Muster**
  - **Adapter**
  - **Beobachter**
  - **Iterator**
  - **Stellvertreter**
  - **Vermittler**
  - Brücke
- Varianten-Muster
- Zustandshandhabungs-Muster
- Steuerungs-Muster
- Bequemlichkeits-Muster

Entkopplungs-Muster

- übergeordnetes Ziel: System in Teile aufspalten, die unabhängig voneinander sind
  $\Longrightarrow$ Teile austauschbar bzw. veränderbar

Adapter

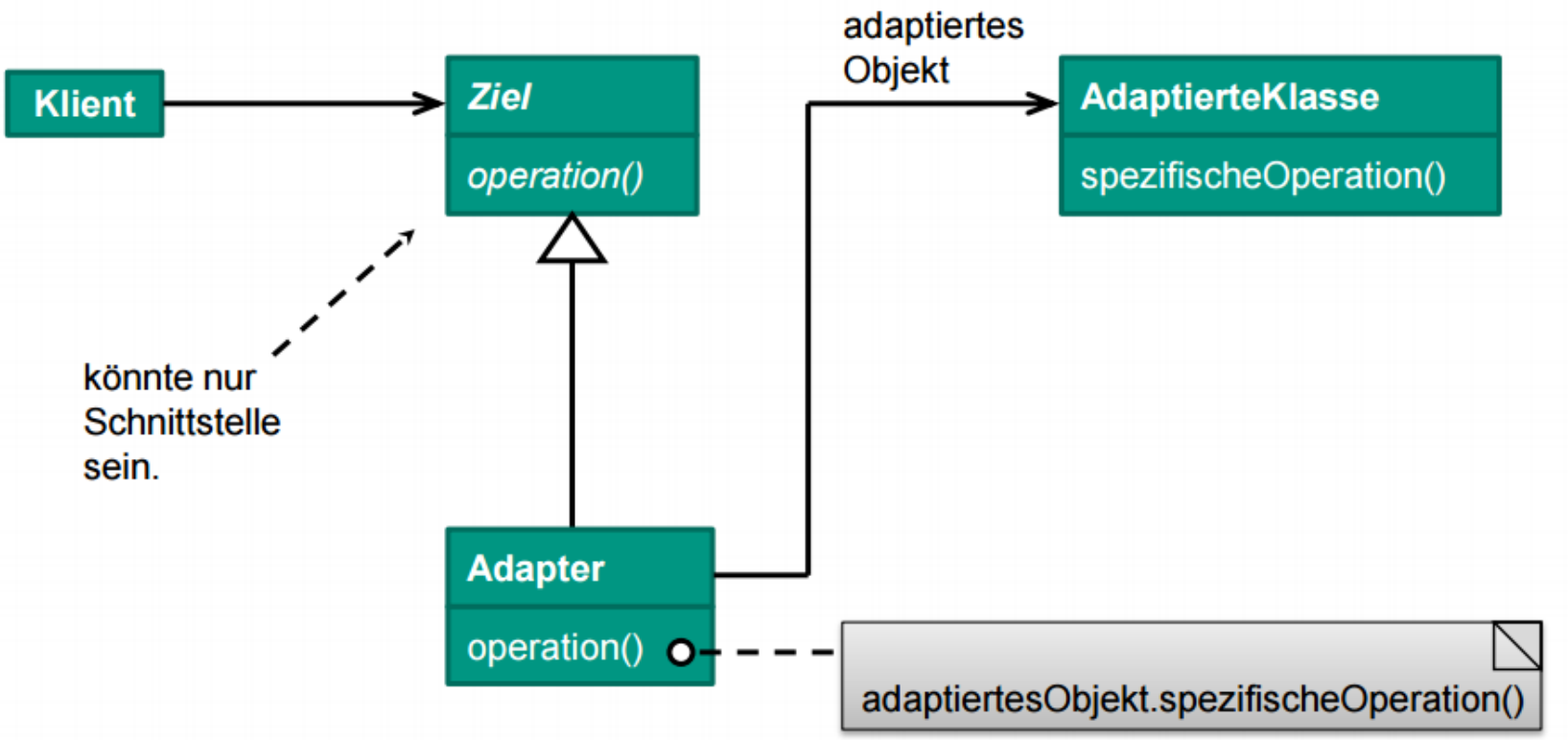
Problem

- Klassen mit inkompatiblen Schnittstellen, die wir aber zusammen benutzen wollen
- Schnittstellen nicht änderbar (z.B. externe Bibliotheken)

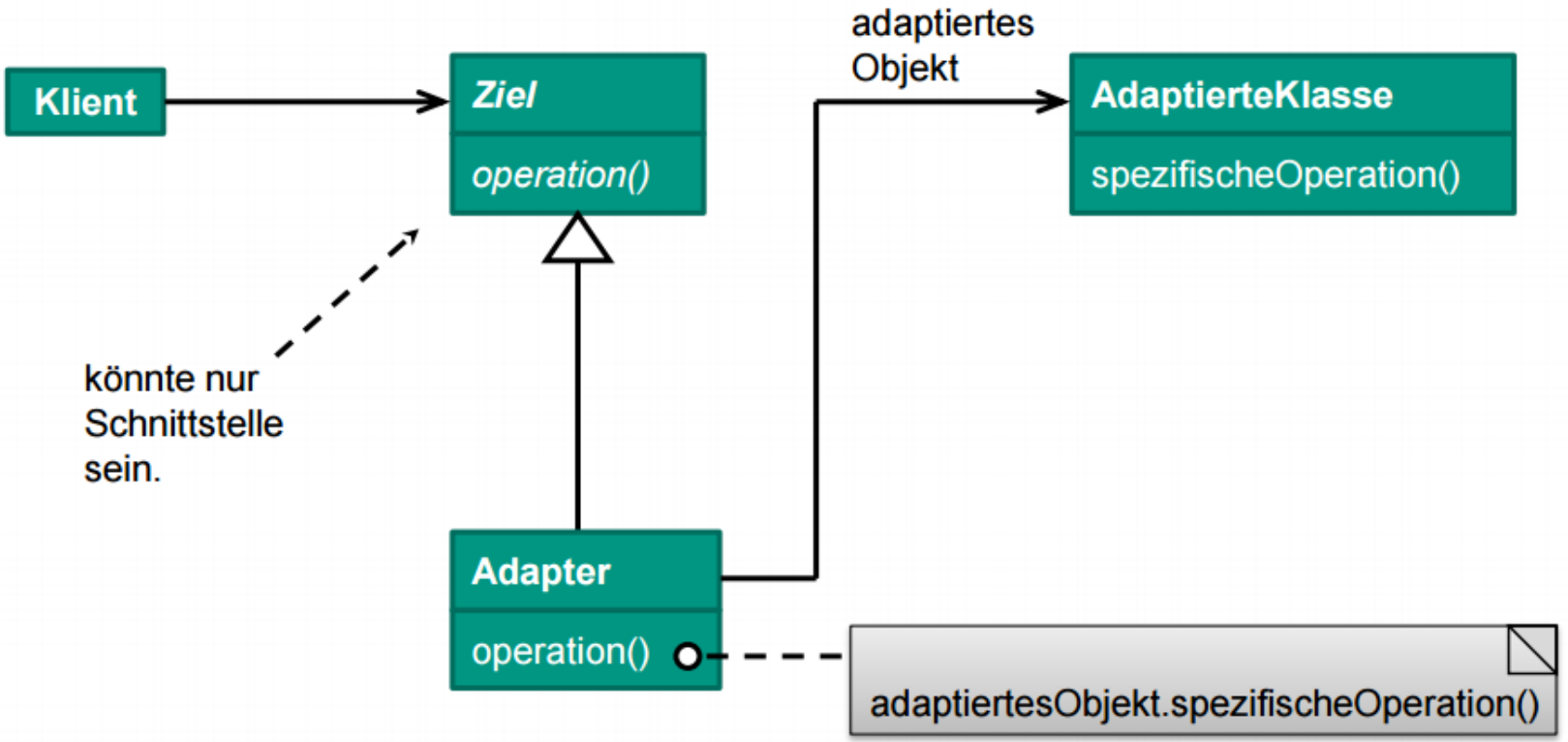
Adapter

Problem

- Klassen mit inkompatiblen Schnittstellen, die wir aber zusammen benutzen wollen
- Schnittstellen nicht änderbar (z.B. externe Bibliotheken)



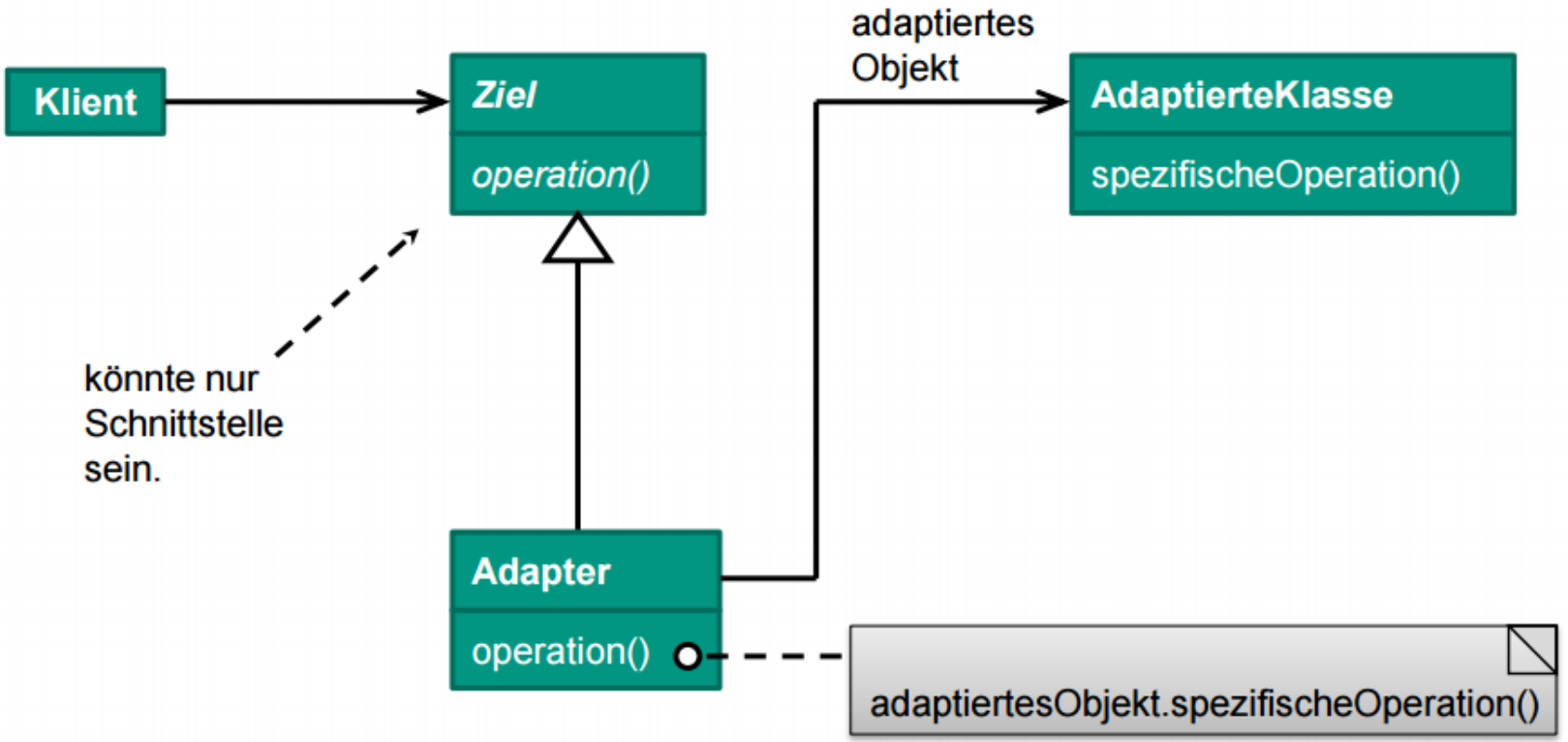
Adapter



Wir sind bei Entkopplung-Mustern, Preisfrage:

Wo ist hier die Entkopplung?

Adapter

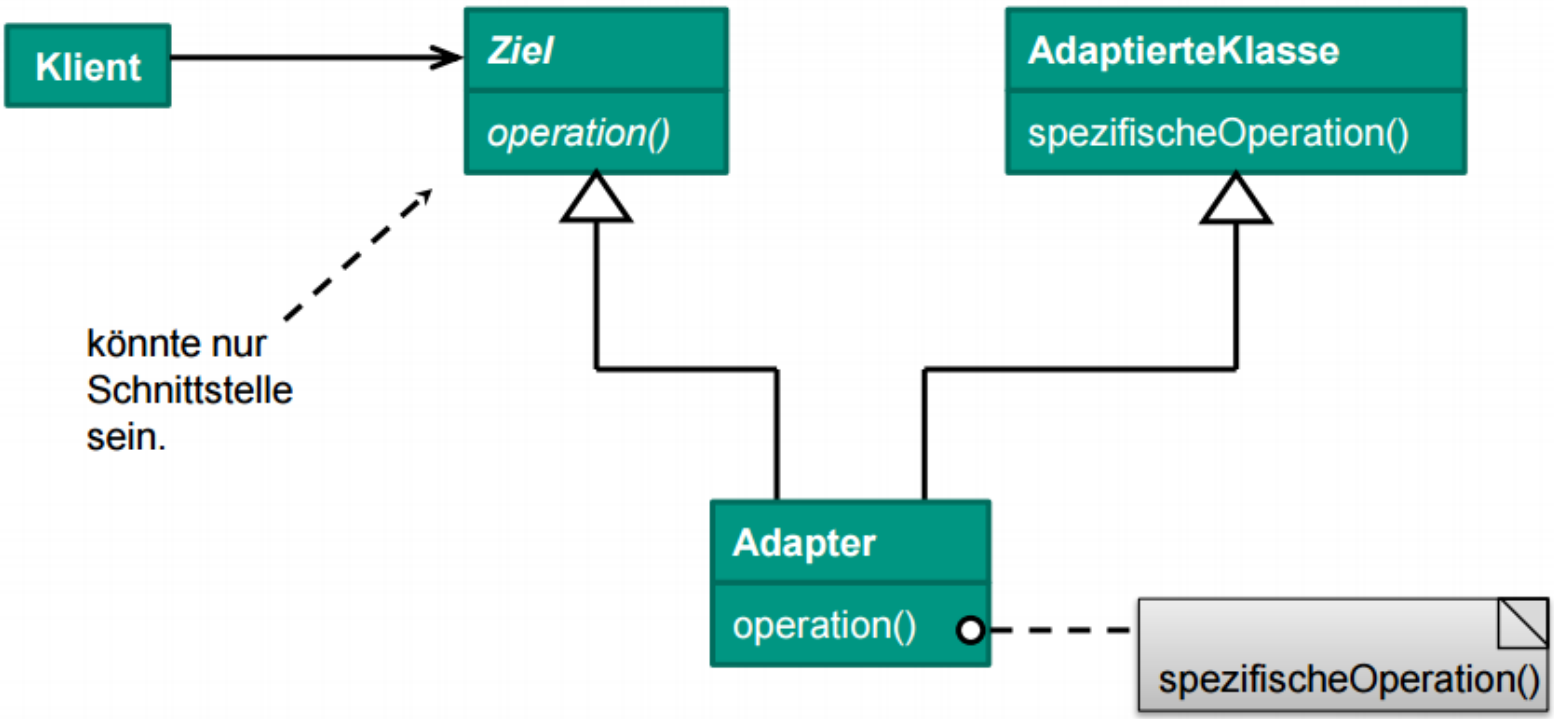


Wir sind bei Entkopplung-Mustern, Preisfrage:

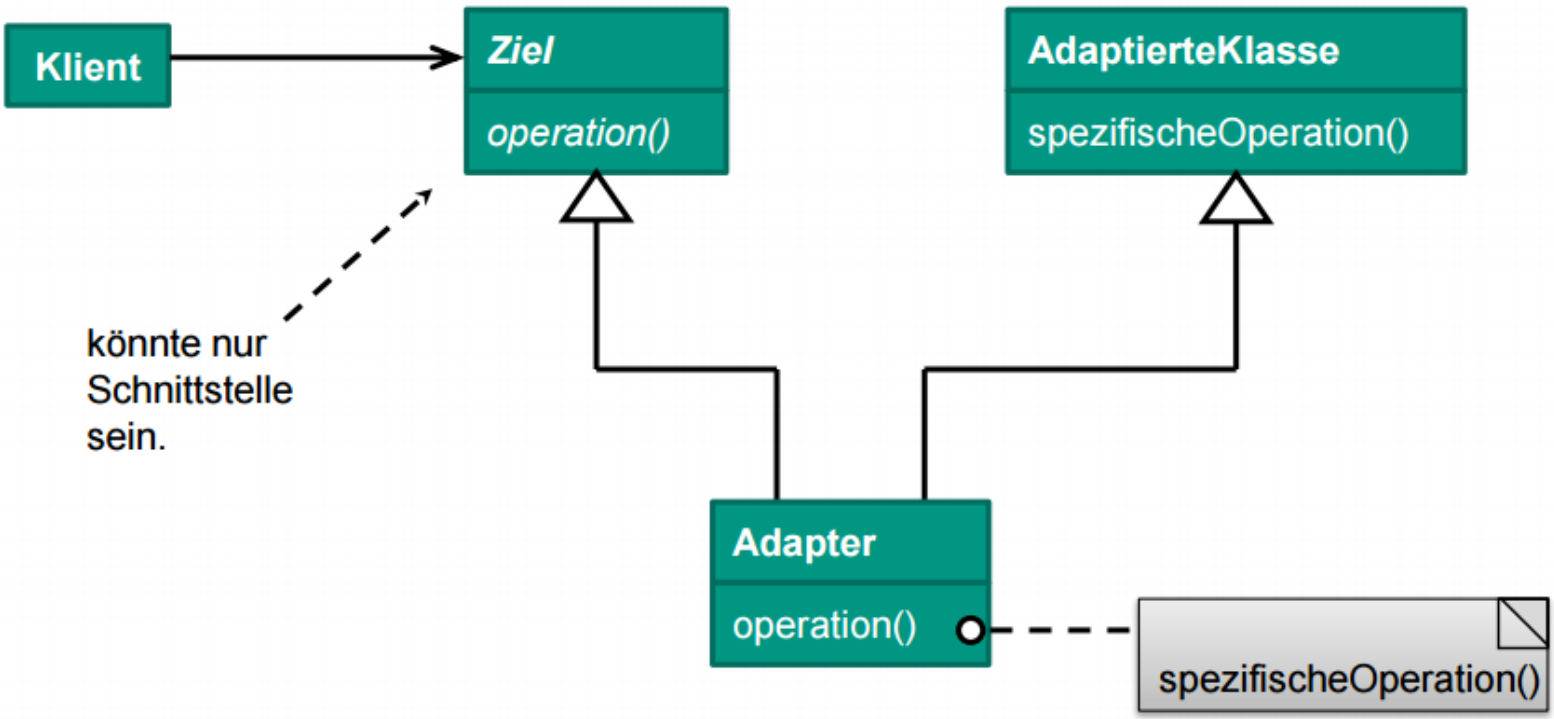
Wo ist hier die Entkopplung?

der Klient ist von der adaptierten Klasse entkoppelt $\Longrightarrow$ austauschbar

Adapter - Alternative

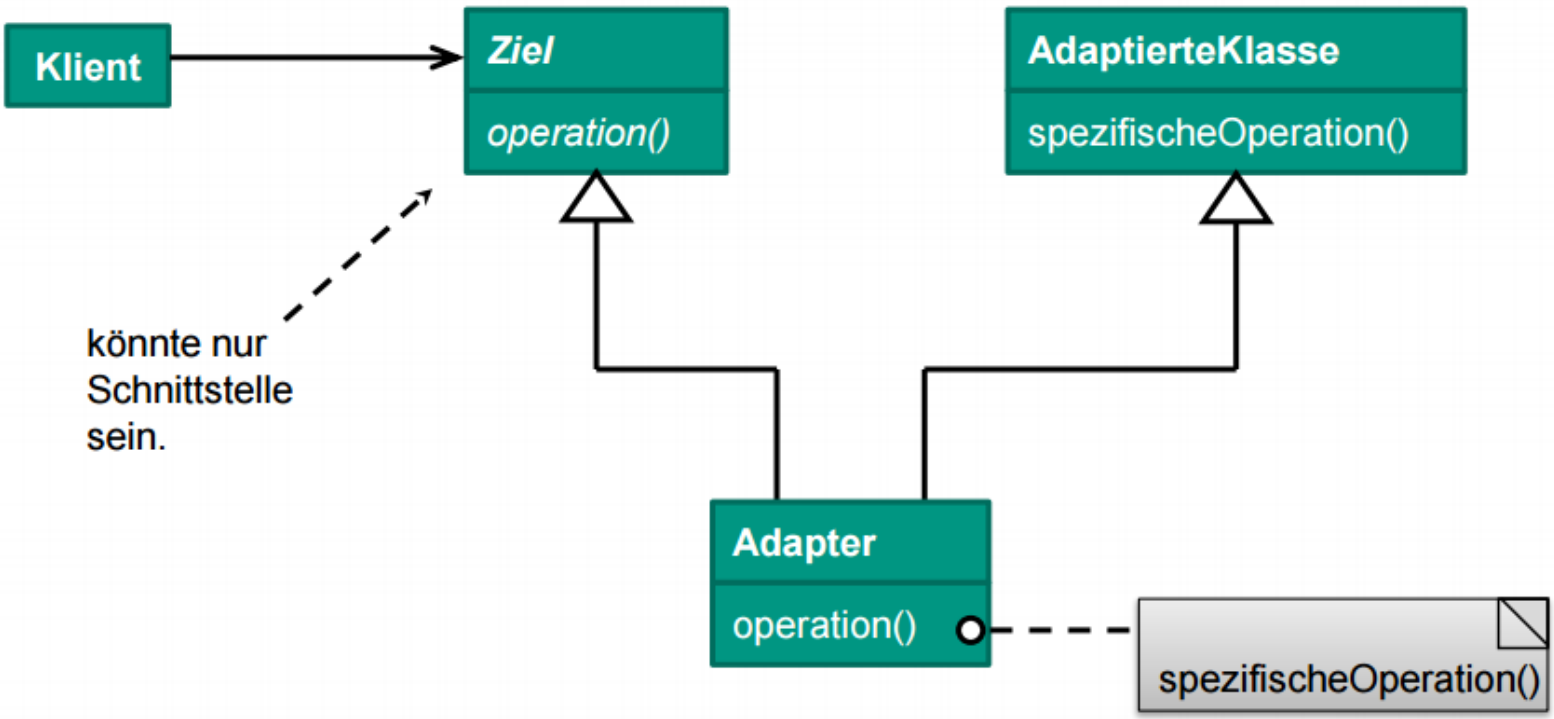


Adapter - Alternative



Was für ein Problem bekommt ihr, wenn ihr das auf einem ÜB implementieren müsst?

Adapter - Alternative



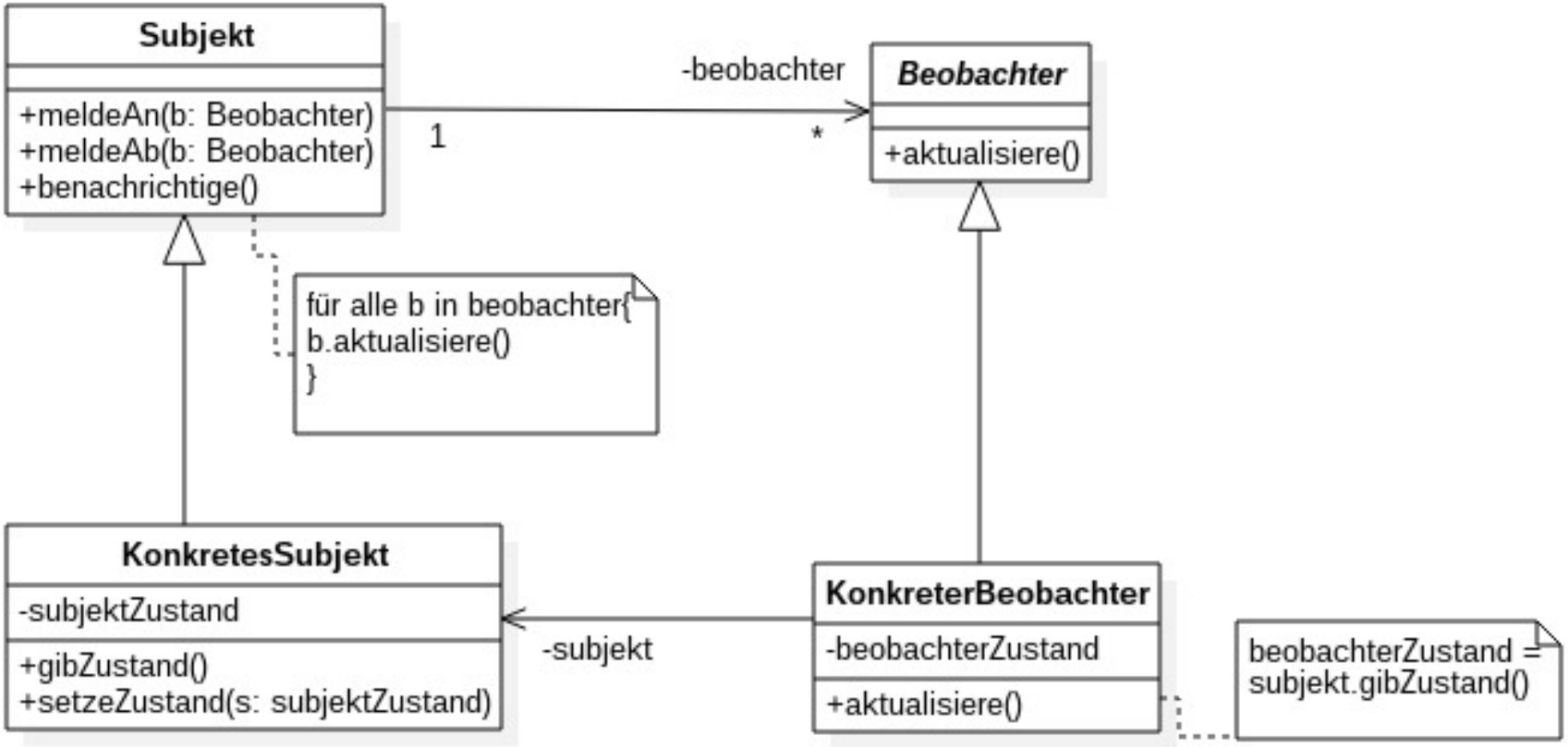
Was für ein Problem bekommt ihr, wenn ihr das auf einem ÜB implementieren müsst?

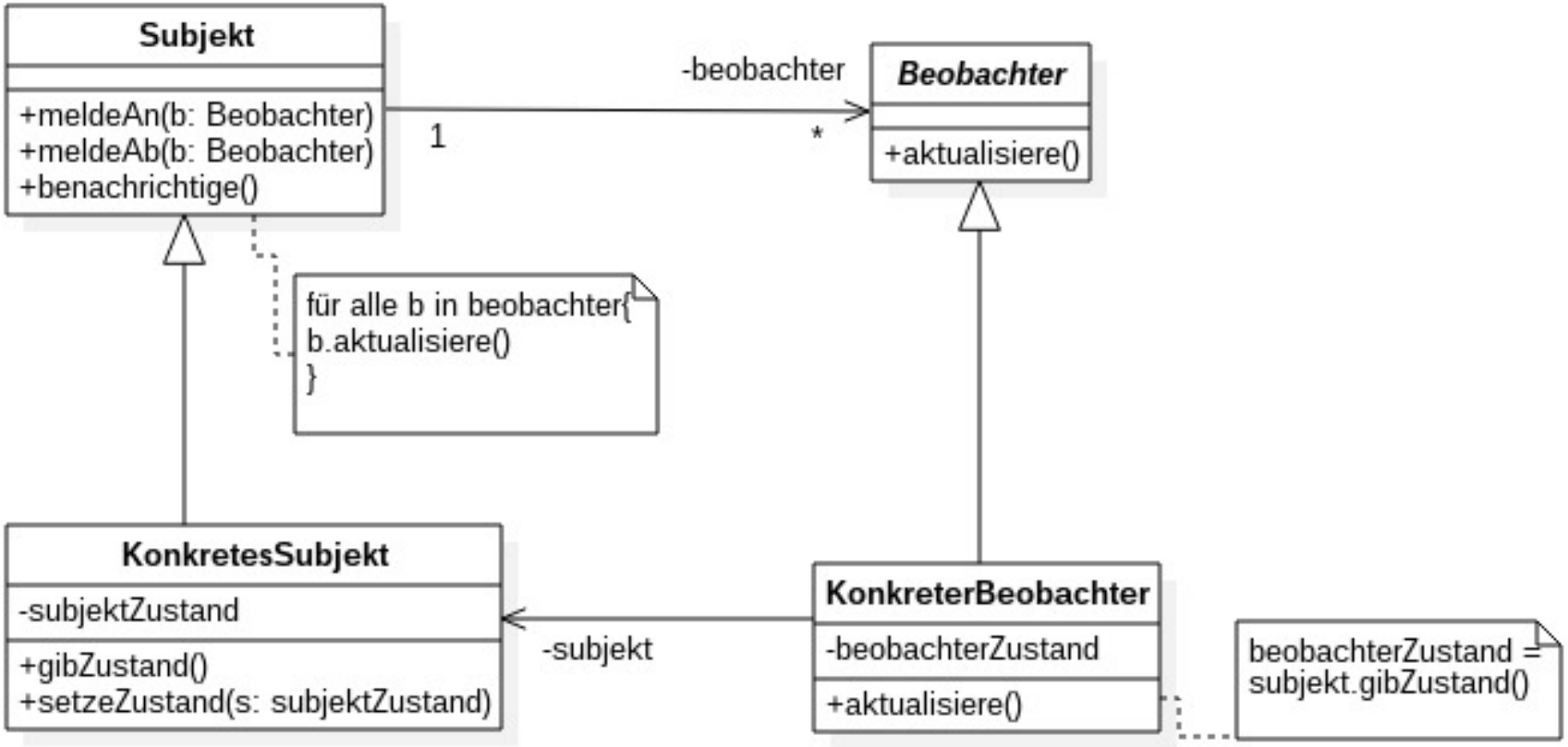
$\Longrightarrow$ keine Mehrfachvererbung in Java!

Beobachter/Observer: abstrakt

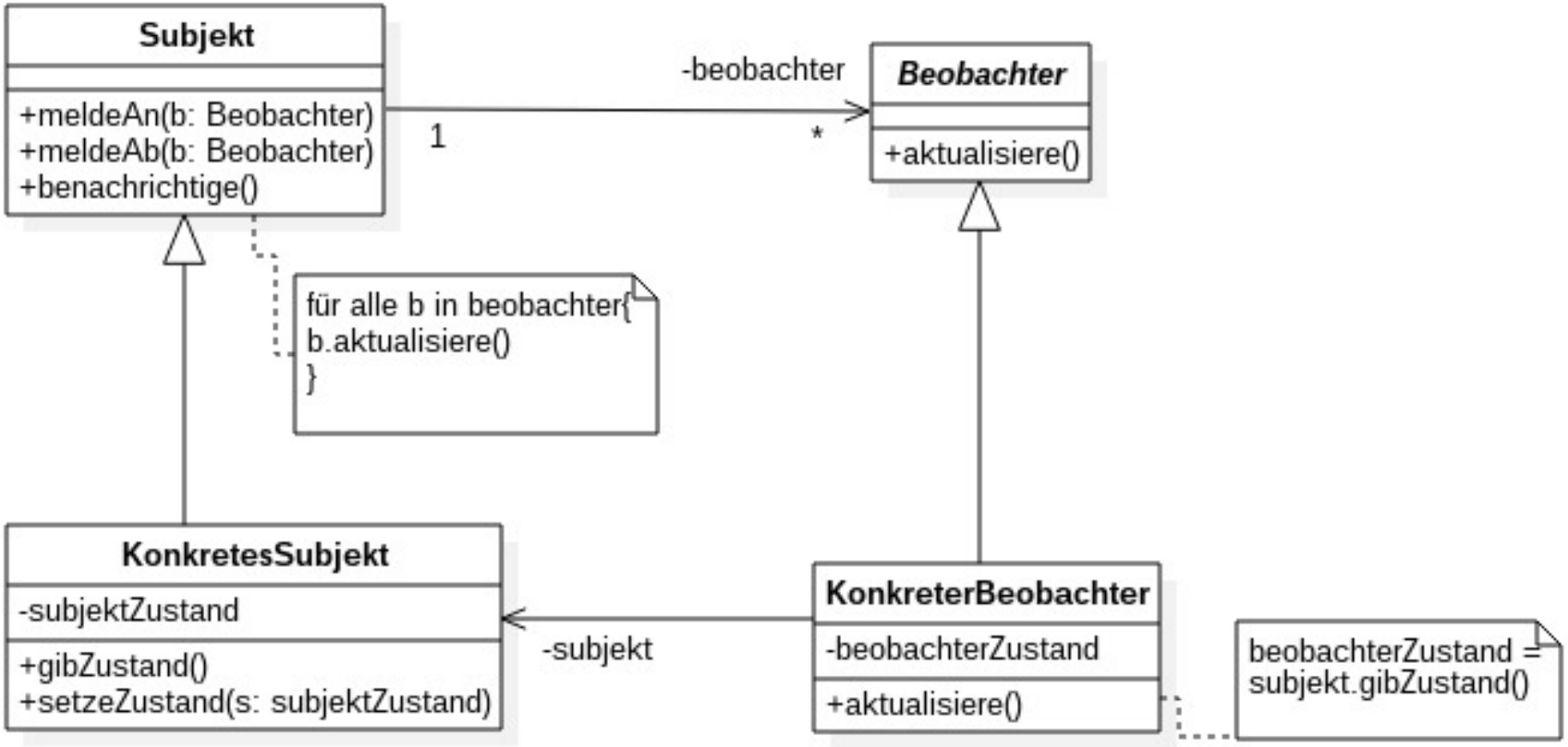
Problem

- ein Subjekt, viele Beobachter
- Subjekt ändert Zustand $\Longrightarrow$ Beobachter machen “irgendwas”


Subjekt
+meldeAn(b: Beobachter)
+meldeAb(b: Beobachter)
+benachrichtige()
-beobachter
1
*
Beobachter
+aktualisiere()
für alle b in beobachter{
b.aktualisiere()
}
KonkretesSubjekt
-subjektZustand
+gibZustand()
+setzeZustand(s: subjektZustand)
-subjekt
KonkreterBeobachter
-beobachterZustand
+aktualisiere()
beobachterZustand =
subjekt.gibZustand()

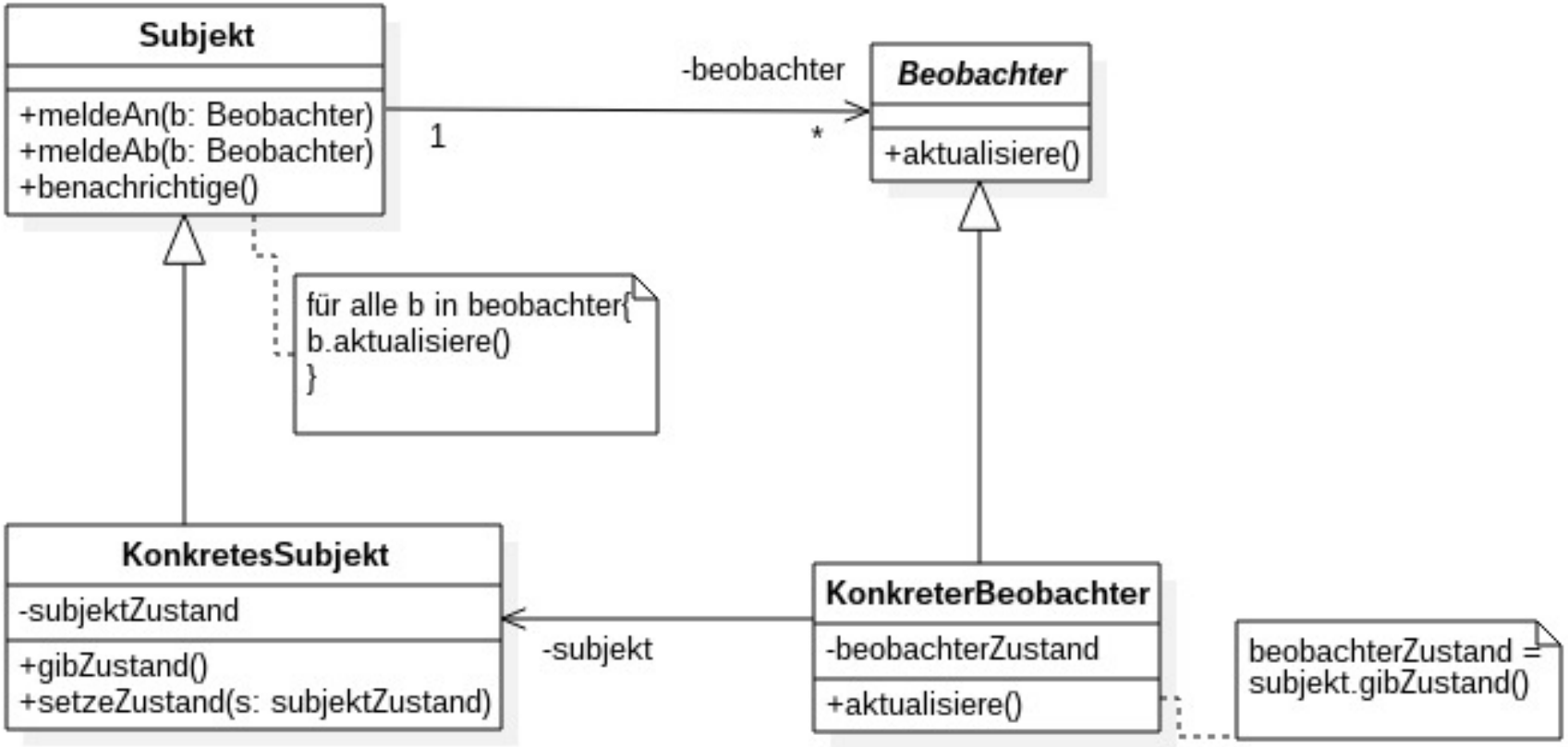


Entkopplung?



Entkopplung?

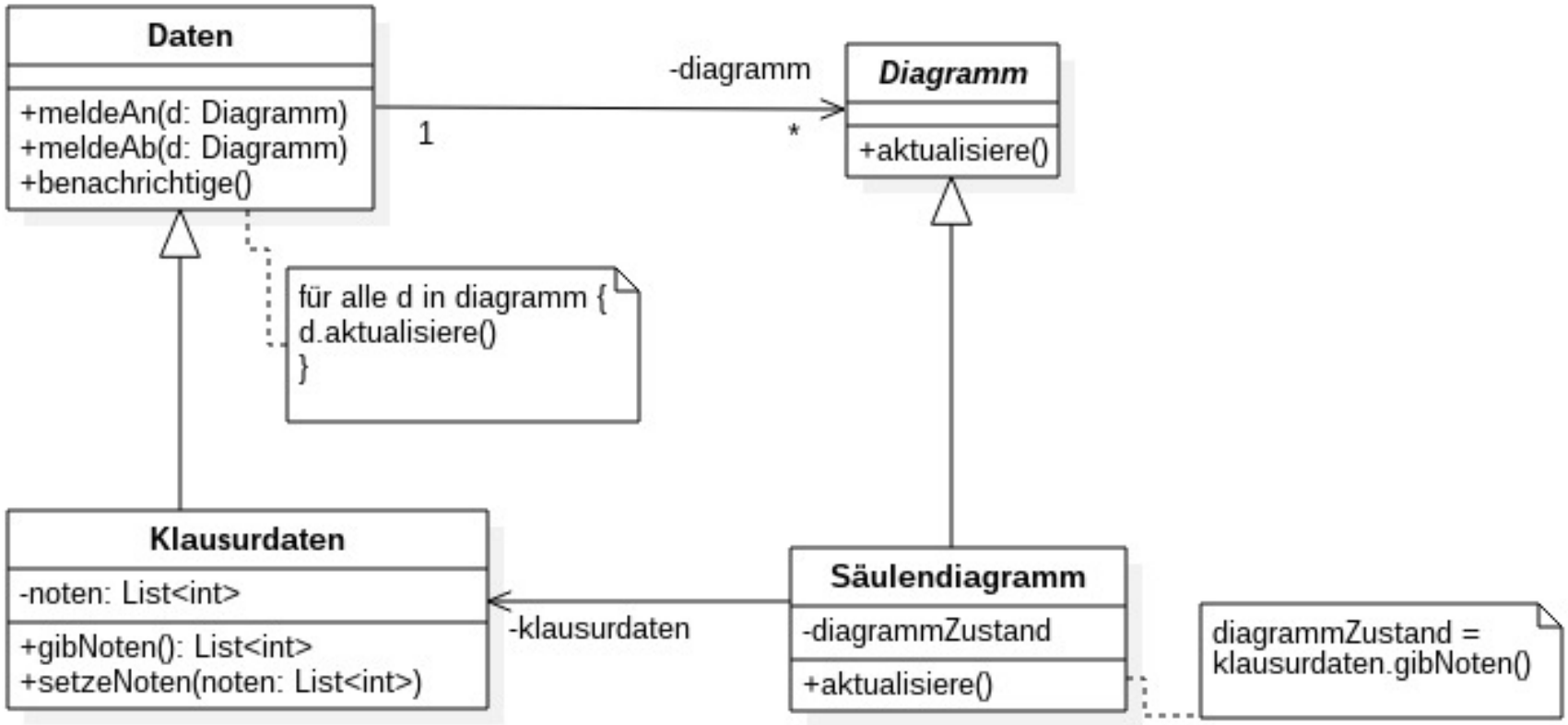
- jeder Beobachter definiert, was bei Benachrichtigung passiert, Subjekt kriegt davon nichts mit



Entkopplung?

- jeder Beobachter definiert, was bei Benachrichtigung passiert, Subjekt kriegt davon nichts mit
- zur Laufzeit änderbar: Anzahl der Beobachter

Beobachter/Observer: am Beispiel



Iterator

Problem

- wollen über Datenstruktur iterieren + Operationen ausführen
  $\Longrightarrow$ Hinzufügen, Löschen...

Iterator

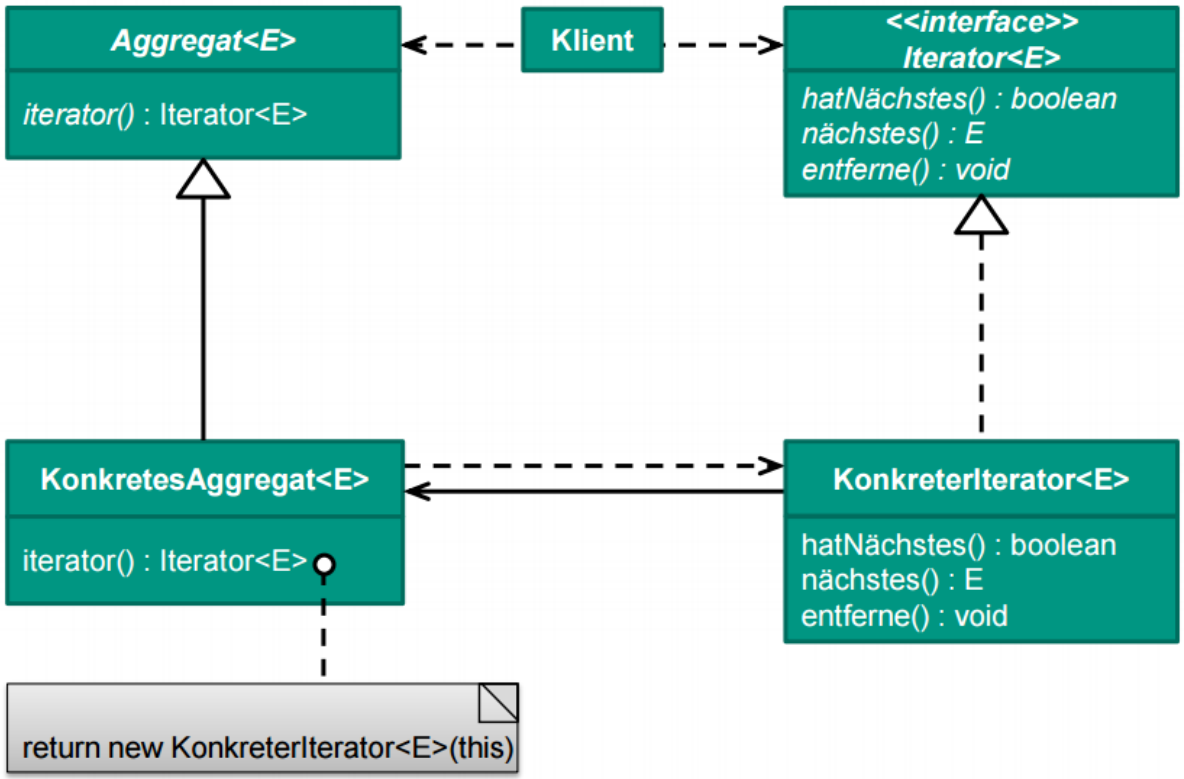
Problem

- wollen über Datenstruktur iterieren + Operationen ausführen
  $\Longrightarrow$ Hinzufügen, Löschen...
- das Ganze ohne Kentniss des internen Aufbaus der Datenstruktur

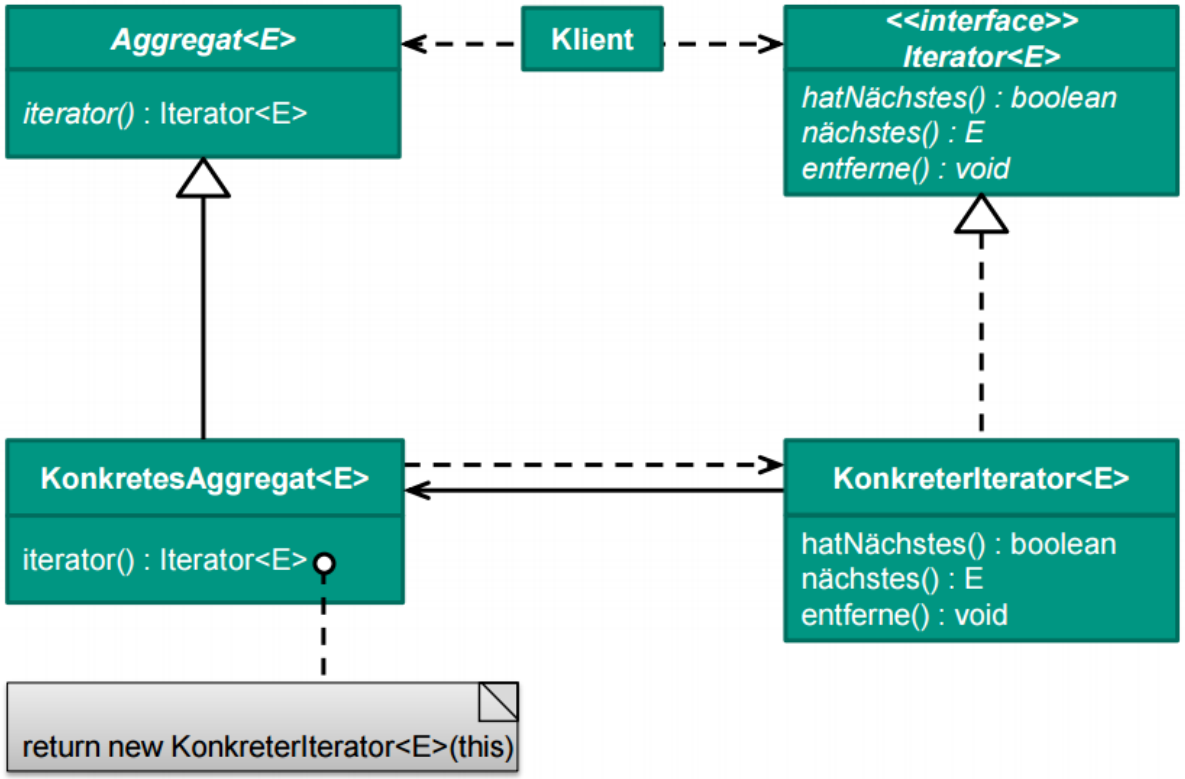
Iterator

Problem

- wollen über Datenstruktur iterieren + Operationen ausführen
  $\Longrightarrow$ Hinzufügen, Löschen...
- das Ganze ohne Kentniss des internen Aufbaus der Datenstruktur

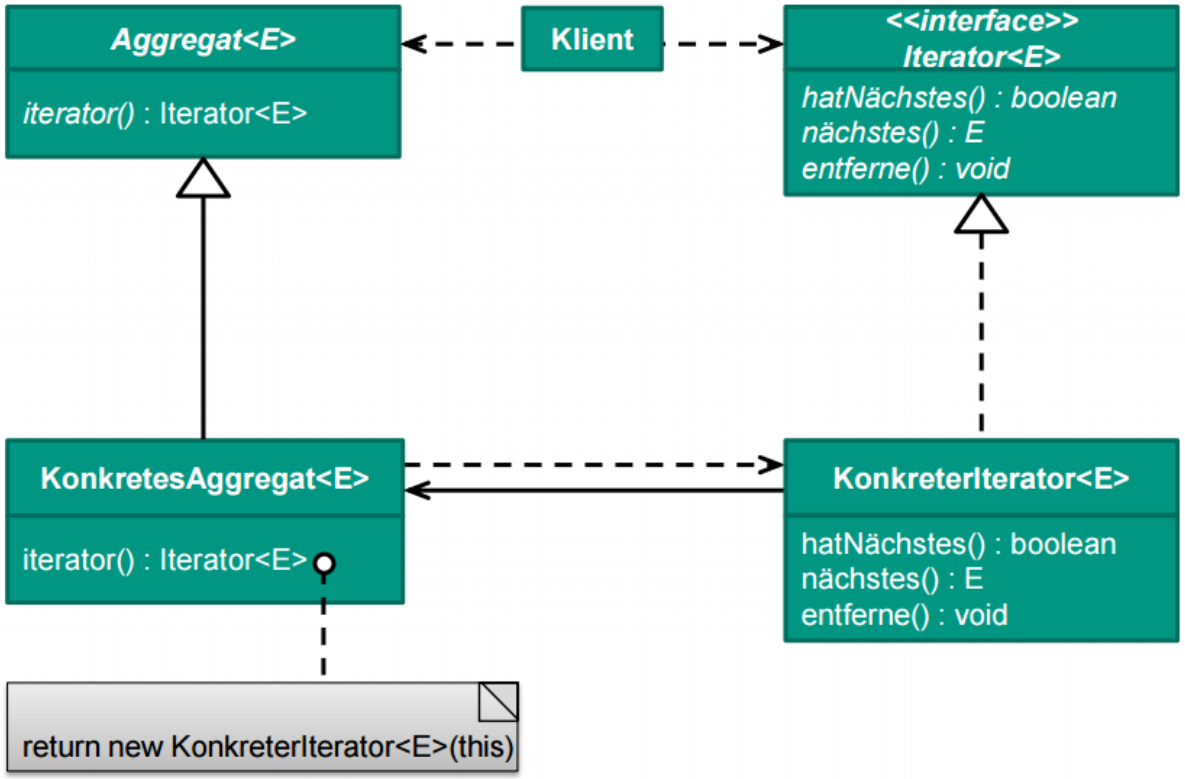


Iterator



Entkopplung?

Iterator



Entkopplung?

- Klient benutzt nur Methoden der Schnittstelle auf dem konkreten Iterator
  $\Longrightarrow$ Implementierung austauschbar

Stellvertreter

Problem

- wollen Zugriff auf ein Objekt kontrollieren, ohne seine Klasse zu ändern

Stellvertreter

Problem

- wollen Zugriff auf ein Objekt kontrollieren, ohne seine Klasse zu ändern

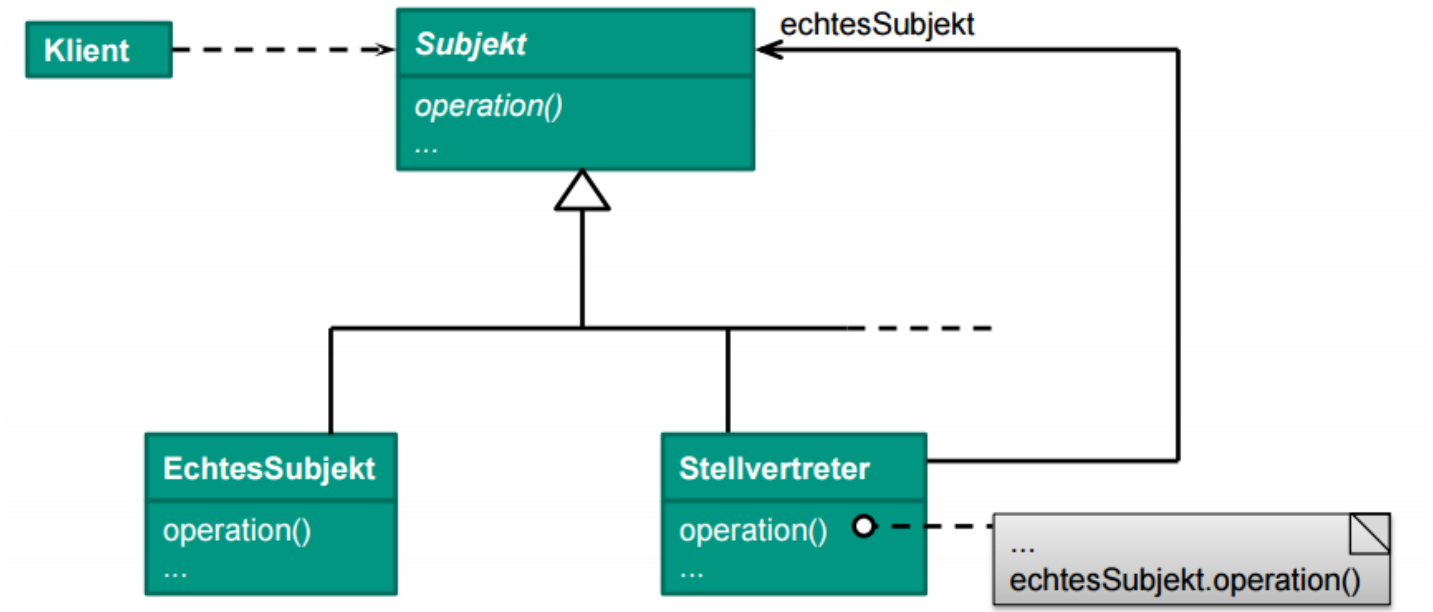
  $\Longrightarrow$ Stellvertreter macht Zugriffskontrolle

Stellvertreter

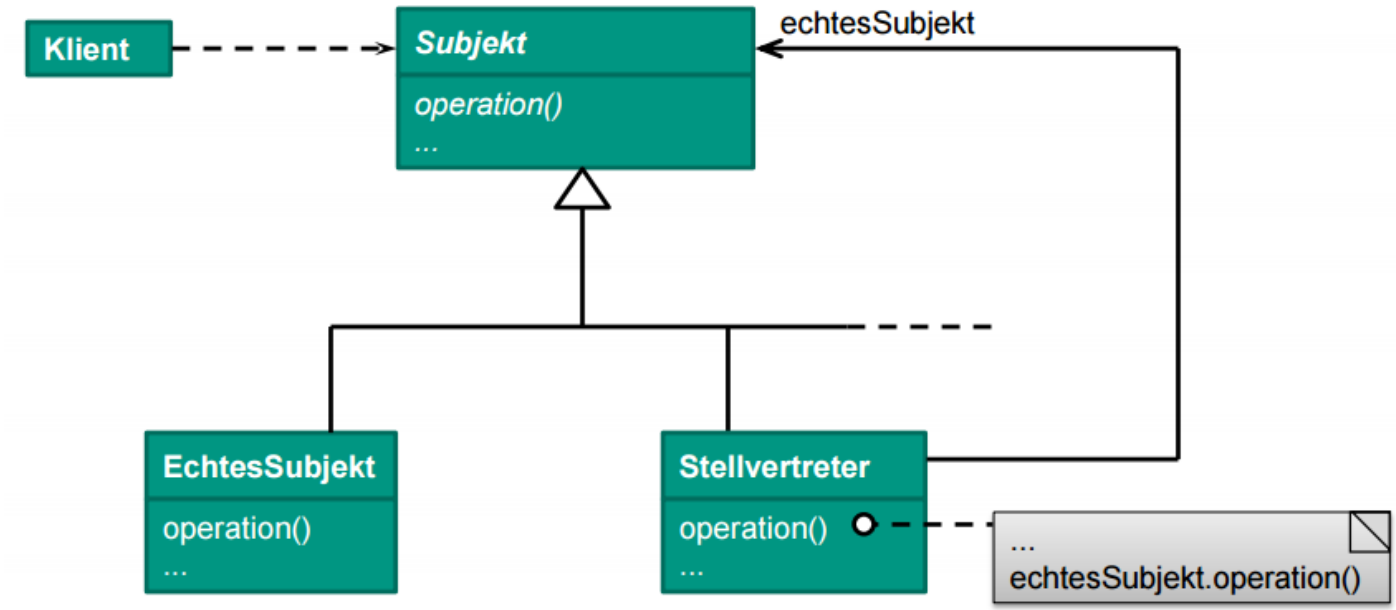
Problem

- wollen Zugriff auf ein Objekt kontrollieren, ohne seine Klasse zu ändern

  $\Longrightarrow$ Stellvertreter macht Zugriffskontrolle

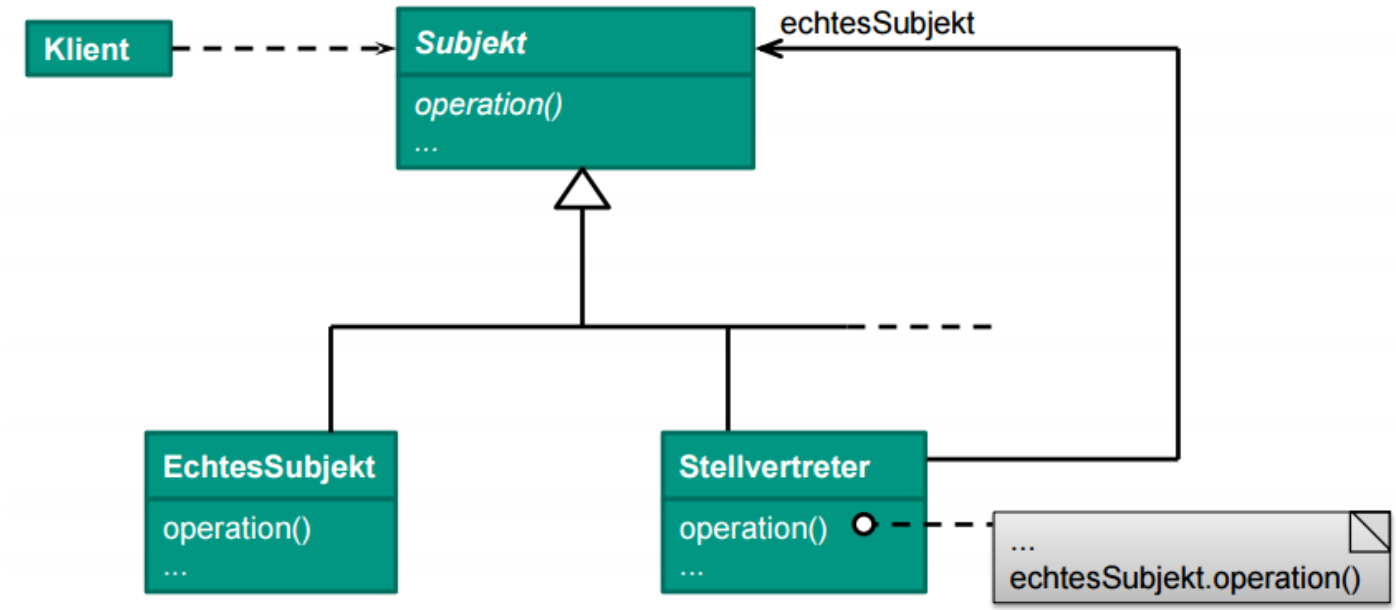


Stellvertreter



Entkopplung?

Stellvertreter



Entkopplung?

- Klient hat keinen direkten Zugriff auf das echte Subjekt

Vermittler

Problem

- mehrere abhängige Objekte

Vermittler

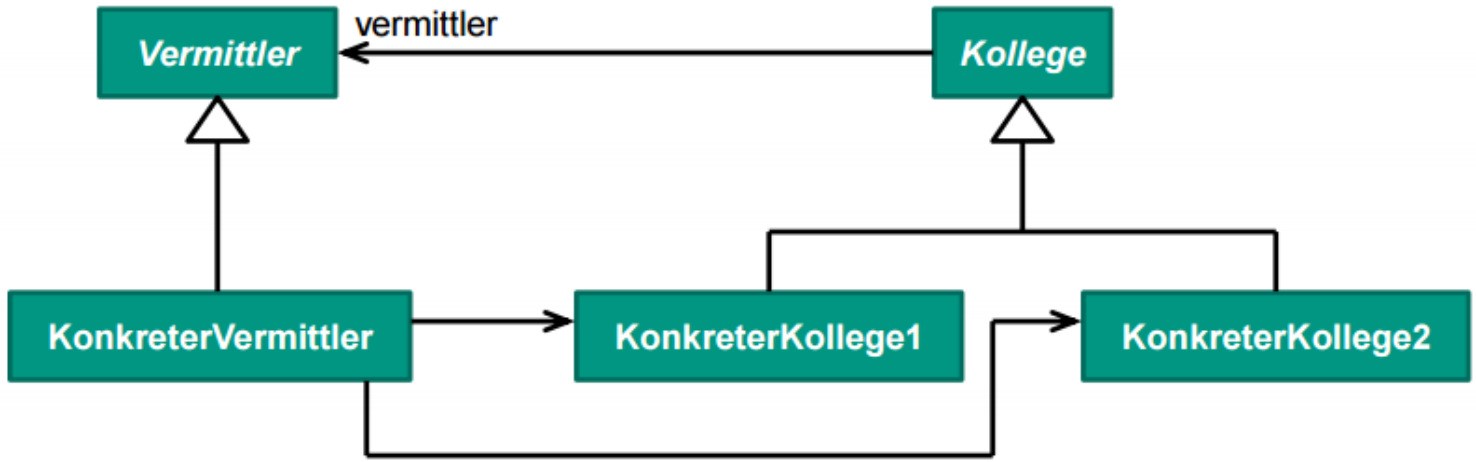
Problem

- mehrere abhängige Objekte
  $\Longrightarrow$ Zustände der Objekte von anderen Zuständen abhängig

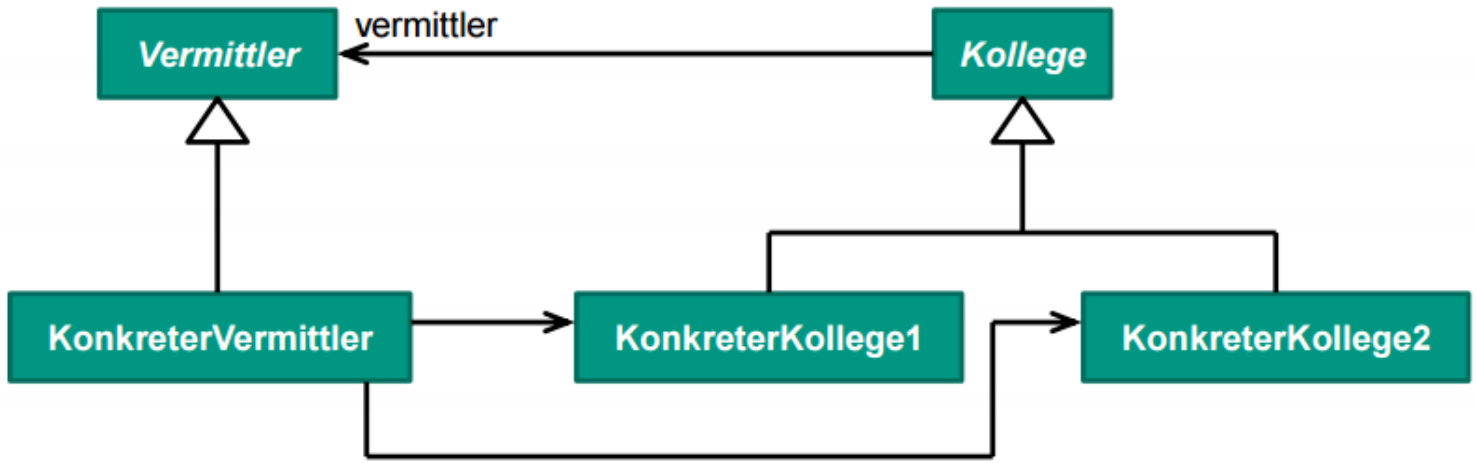
Vermittler

Problem

- mehrere abhängige Objekte
  $\Longrightarrow$ Zustände der Objekte von anderen Zuständen abhängig

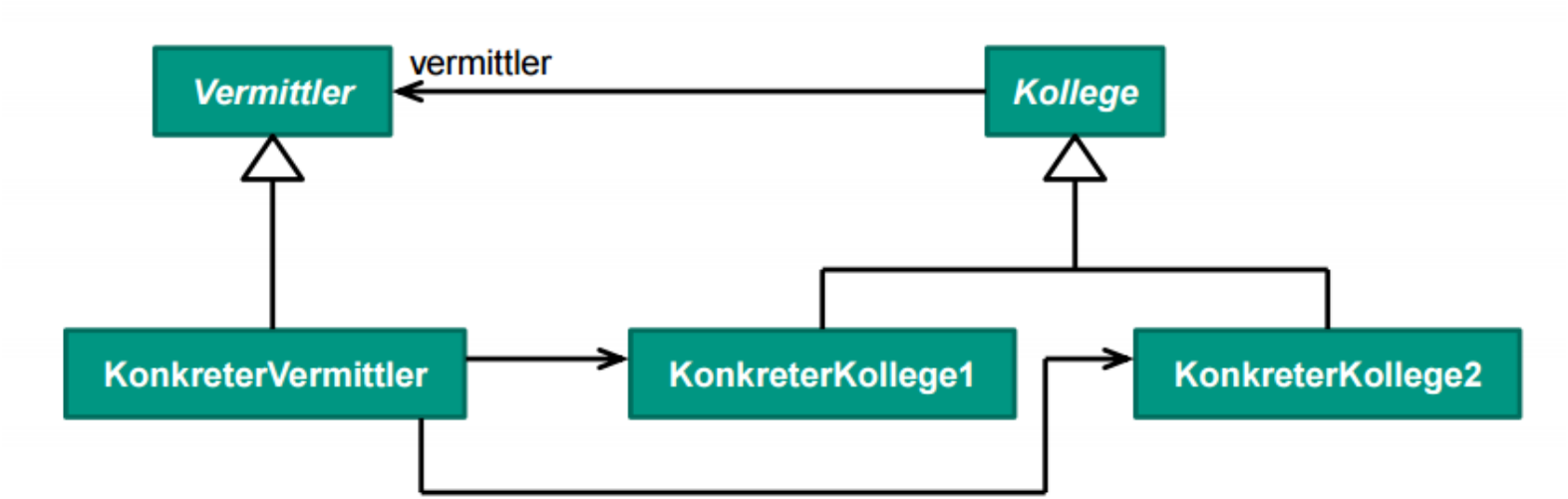


Vermittler



Entkopplung?

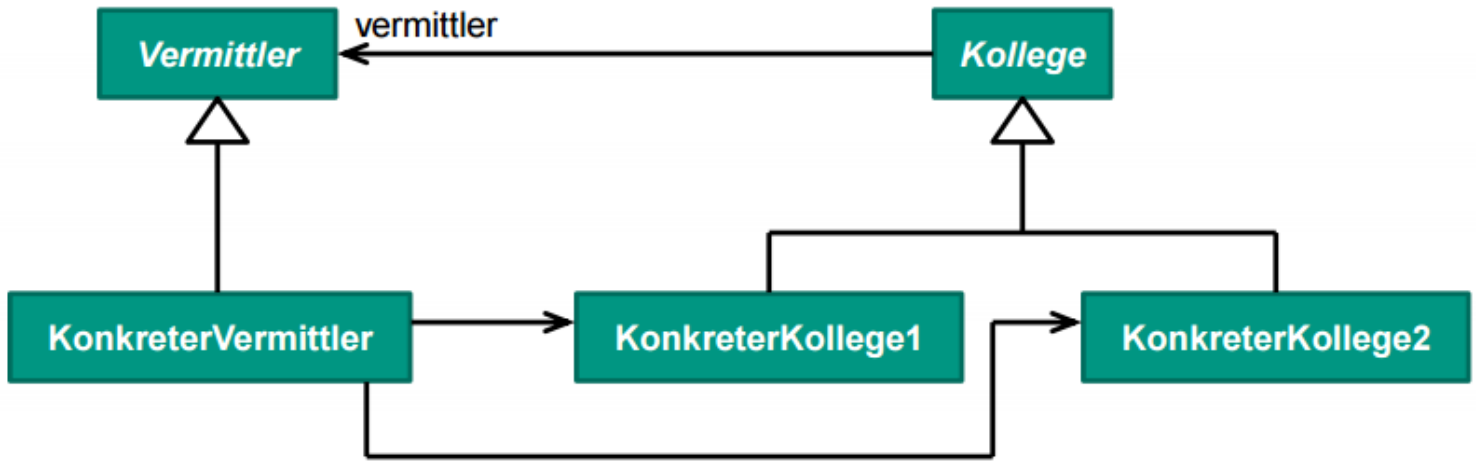
Vermittler



Entkopplung?

- Kollegen kennen sich nicht direkt

Vermittler



Entkopplung?

- Kollegen kennen sich nicht direkt
  $\Longrightarrow$ Hinzufügen eines Kollegen erfordert keine Änderung der alten Kollegen

Tipps - 4. Übungsblatt

Aufgabe

Tipps - 4. Übungsblatt

Aufgabe

Aufgabe

Tipps - 4. Übungsblatt

Aufgabe

Tipps - 4. Übungsblatt

Aufgabe

Aufgabe

Denkt dran!

Abgabe

- Deadline am 21.6 um 12:00

Bis dann! (dann := 26.06.17)