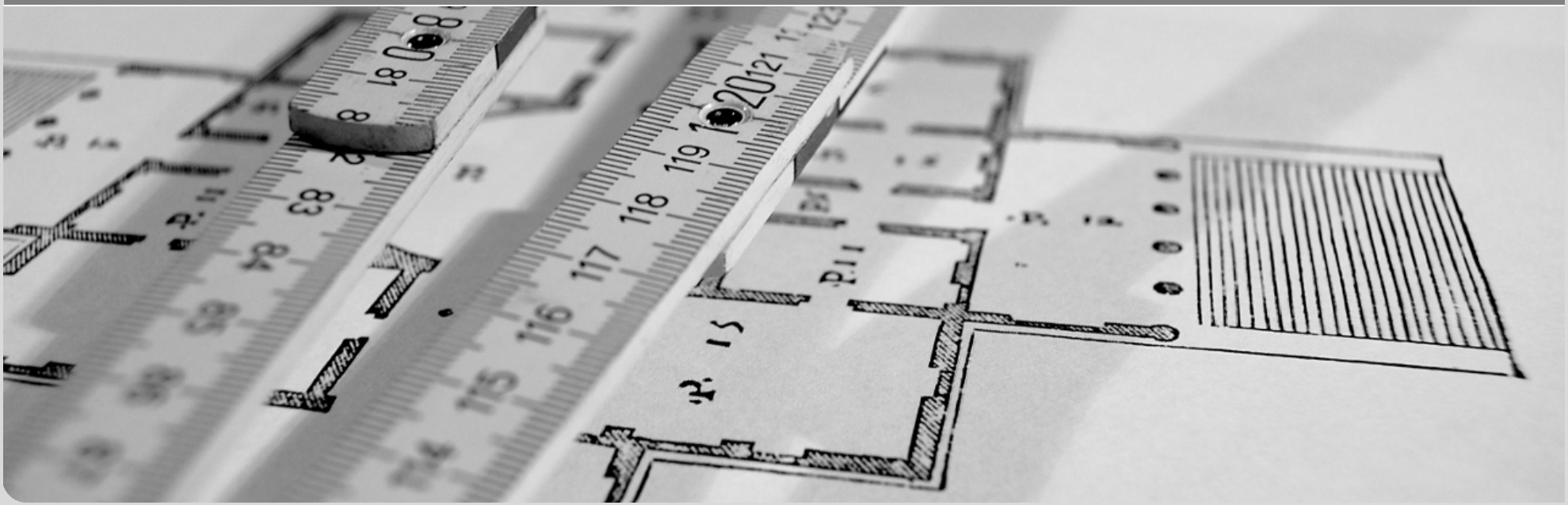
Karlsruher Institut für Technologie

Softwaretechnik 1 - 3. Tutorium

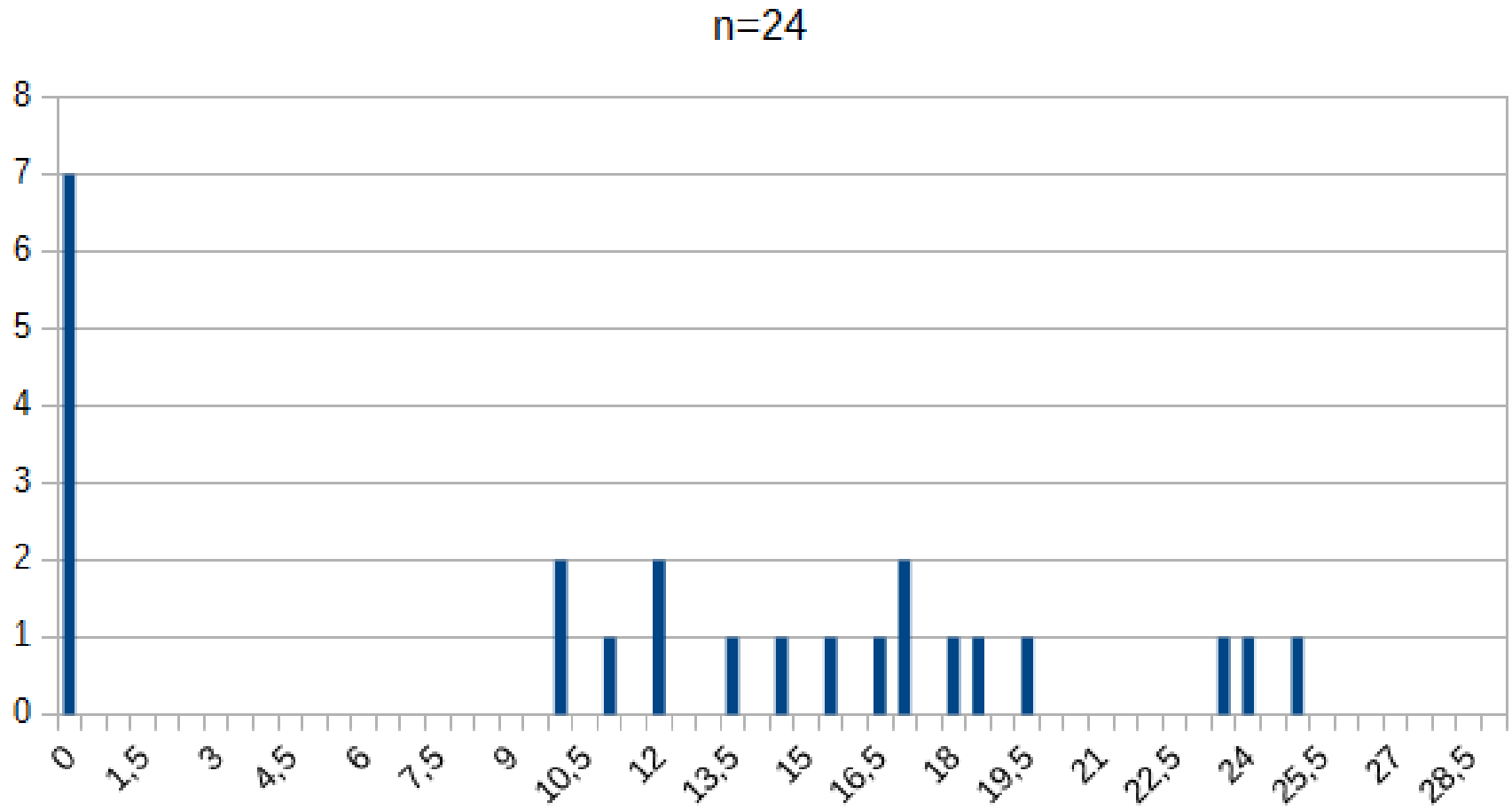
Tutorium 03

Felix Bachmann | 12.06.2017

KIT - INSTITUT FÜR PROGRAMMSTRUKTUREN UND DATENORGANISATION (IPD)



2. Übungsblatt Statistik



∅ 11,56 bzw. 16,32 von 28+1

Häufige Fehler

Allgemein

- verspätete Abgaben bekomme ich erst beim jeweils nächsten Tutorentreffen
  $\Longrightarrow$ Rückgabe dauert länger; gibt keine Punkte, nur grobe Korrektur

Häufige Fehler

Aufgabe 1 (Plug-In-Architektur): ∅ 2,25 bzw. 3,86 von 5

Häufige Fehler

Aufgabe 1 (Plug-In-Architektur): ∅ 2,25 bzw. 3,86 von 5

- keine riesigen switch-cases/if-else Anweisungen in compareTo() benutzen
  $\Longrightarrow$ Erweiterbarkeit wird dadurch eingeschränkt

Häufige Fehler

Aufgabe 1 (Plug-In-Architektur): ∅ 2,25 bzw. 3,86 von 5

- keine riesigen switch-cases/if-else Anweisungen in compareTo() benutzen
  $\Longrightarrow$ Erweiterbarkeit wird dadurch eingeschränkt
  $\Longrightarrow$ Java vergleicht Enum-Elemente anhand ihrer Position in dem Enum

Häufige Fehler

Aufgabe 1 (Plug-In-Architektur): ∅ 2,25 bzw. 3,86 von 5

- keine riesigen switch-cases/if-else Anweisungen in compareTo() benutzen
  $\Longrightarrow$ Erweiterbarkeit wird dadurch eingeschränkt
  $\Longrightarrow$ Java vergleicht Enum-Elemente anhand ihrer Position in dem Enum
- orientiert euch nicht am JMJRST-Stil

Häufige Fehler

Aufgabe 2 (Plug-In): ∅ 1,79 bzw. 3,58 von 4

- keine :D

Aufgabe 3 (iMage-Bundle): ∅ 0,44 bzw. 1,75 von 2

- 6 Abgaben . . .

Häufige Fehler

Aufgabe 4 (Aktivitätsdiagramme (Geometrify)): ∅ 4,83 bzw. 7,25 von 10

Häufige Fehler

Aufgabe 4 (Aktivitätsdiagramme (Geometrify)): ∅ 4,83 bzw. 7,25 von 10

- denkt an die Rauten!

Häufige Fehler

Aufgabe 4 (Aktivitätsdiagramme (Geometrify)): ∅ 4,83 bzw. 7,25 von 10

- denkt an die Rauten!
- Geometrify sollte eigener Kasten mit Objektfluss sein

Häufige Fehler

Aufgabe 4 (Aktivitätsdiagramme (Geometrify)): ∅ 4,83 bzw. 7,25 von 10

- denkt an die Rauten!
- Geometrify sollte eigener Kasten mit Objektfluss sein
- [Bedingung]

Häufige Fehler

Aufgabe 4 (Aktivitätsdiagramme (Geometrify)): ∅ 4,83 bzw. 7,25 von 10

- denkt an die Rauten!
- Geometrify sollte eigener Kasten mit Objektfluss sein
- [Bedingung]
- verschachtelte Aktivitäten $\Longrightarrow$ irgendwo passender Kasten dazu

Häufige Fehler

Aufgabe 5 (Sequenzdiagramm (main-Methode)): ∅ 1,58 bzw. 2,71 von 5

Häufige Fehler

Aufgabe 5 (Sequenzdiagramm (main-Methode)): ∅ 1,58 bzw. 2,71 von 5

- bzgl. Konstruktor sind VL-Folien etwas blöd

Häufige Fehler

Aufgabe 5 (Sequenzdiagramm (main-Methode)): ∅ 1,58 bzw. 2,71 von 5

- bzgl. Konstruktor sind VL-Folien etwas blöd
- asynchron vs. synchron (Pfeilspitzen sind wichtig!)

Häufige Fehler

Aufgabe 5 (Sequenzdiagramm (main-Methode)): ∅ 1,58 bzw. 2,71 von 5

- bzgl. Konstruktor sind VL-Folien etwas blöd
- asynchron vs. synchron (Pfeilspitzen sind wichtig!)
- nicht statische Instanzen unterstreichen

Häufige Fehler

Aufgabe 5 (Sequenzdiagramm (main-Methode)): ∅ 1,58 bzw. 2,71 von 5

- bzgl. Konstruktor sind VL-Folien etwas blöd
- asynchron vs. synchron (Pfeilspitzen sind wichtig!)
- nicht statische Instanzen unterstreichen
- Instanz-Kästen erst dann hinzeichnen, wenn Instanz auch existiert

Häufige Fehler

Aufgabe 6 (Substitutionsprinzip): ∅ 0,67 bzw. 1,23 von 3

Häufige Fehler

Aufgabe 6 (Substitutionsprinzip): ∅ 0,67 bzw. 1,23 von 3

- die Methode wurde überladen

  $\Longrightarrow$ Java schaut sich nur die Signatur an

Häufige Fehler

Aufgabe 6 (Substitutionsprinzip): ∅ 0,67 bzw. 1,23 von 3

- die Methode wurde überladen
  $\Longrightarrow$ Java schaut sich nur die Signatur an
- für Erfüllung des Substitutionsprinzips auch Verhalten wichtig

Häufige Fehler

Aufgabe 6 (Substitutionsprinzip): ∅ 0,67 bzw. 1,23 von 3

- die Methode wurde überladen
  $\Longrightarrow$ Java schaut sich nur die Signatur an
- für Erfüllung des Substitutionsprinzips auch Verhalten wichtig
- Methoden sind dynamisch gebunden, Attribute sind statisch gebunden
  $\Longrightarrow$ merke: getter und setter benutzen
  $\Longrightarrow$ vermeidet mindfucks

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?

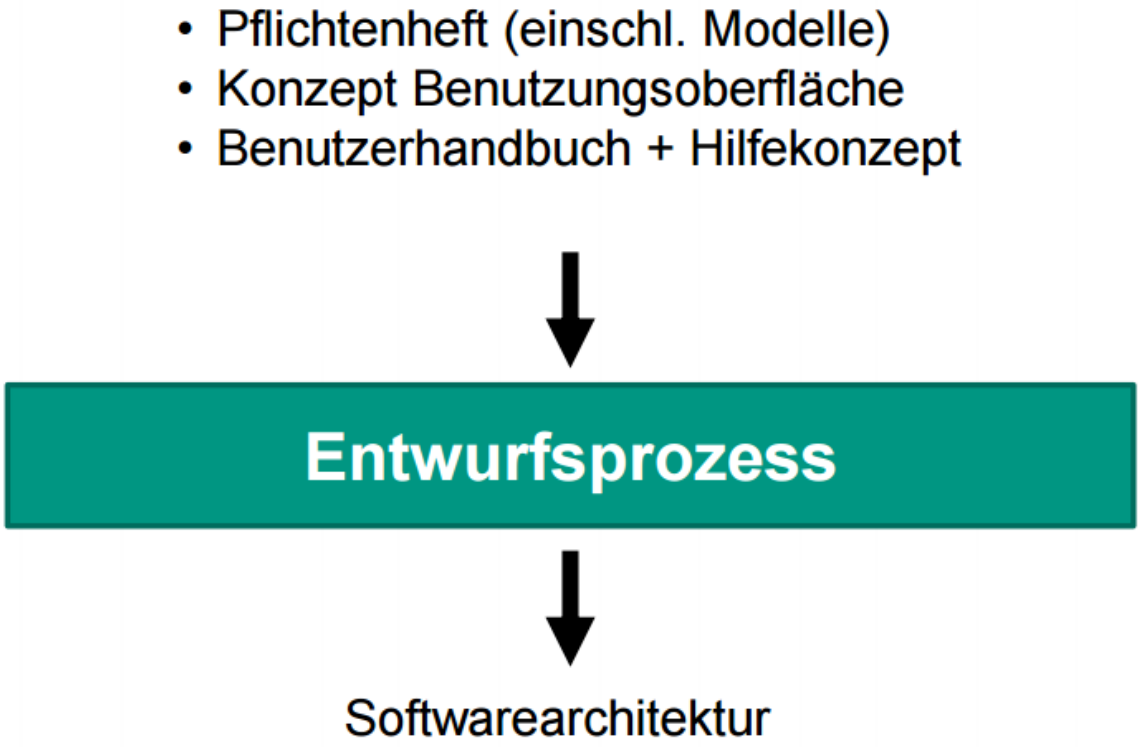
Wo sind wir?

- 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?



Softwarearchitektur ist Grundlage für Implementierung!

Abgrenzung Definition vs. Entwurf

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

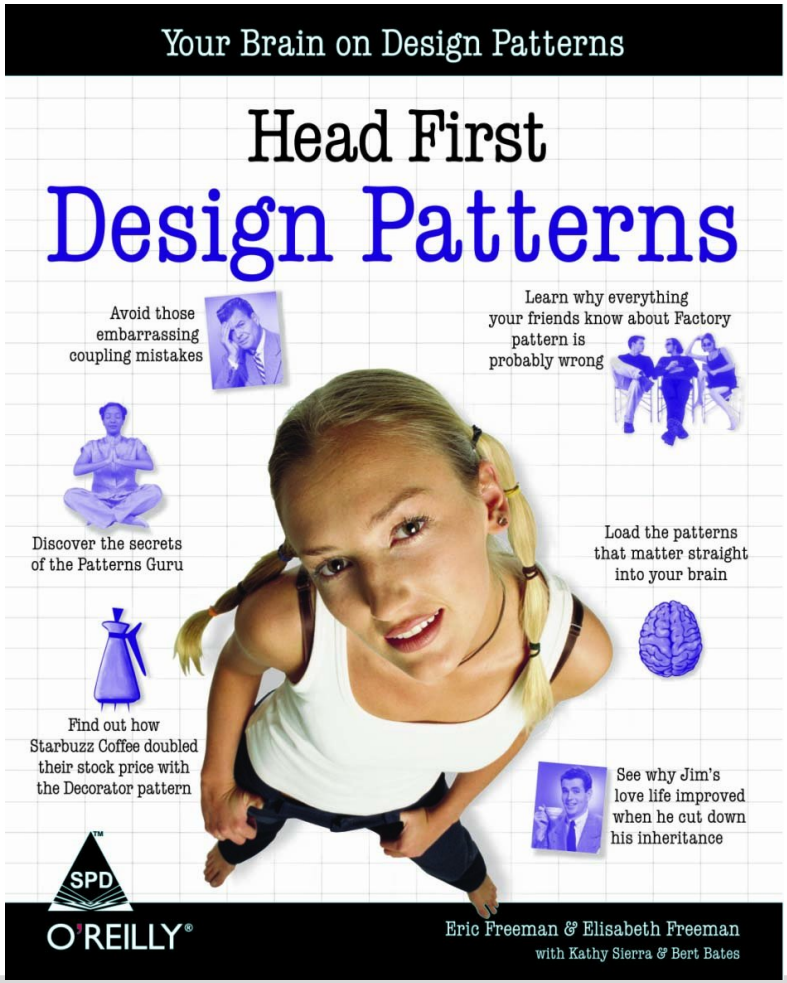
Abgrenzung Definition vs. Entwurf

- 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



Was sind Entwurfsmuster?

Entwurfsmuster

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

Was sind Entwurfsmuster?

Entwurfsmuster

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

- schematische Klassendiagramme zur Lösung von häufig auftretenden Problemen

Was sind 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!

Was sind 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?

- erleichtern Kommunikation

Wozu Entwurfsmuster?

- erleichtern Kommunikation
- erleichtern “gute“ Entwürfe und das Schreiben von wartbarem/erweiterbarem Code

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

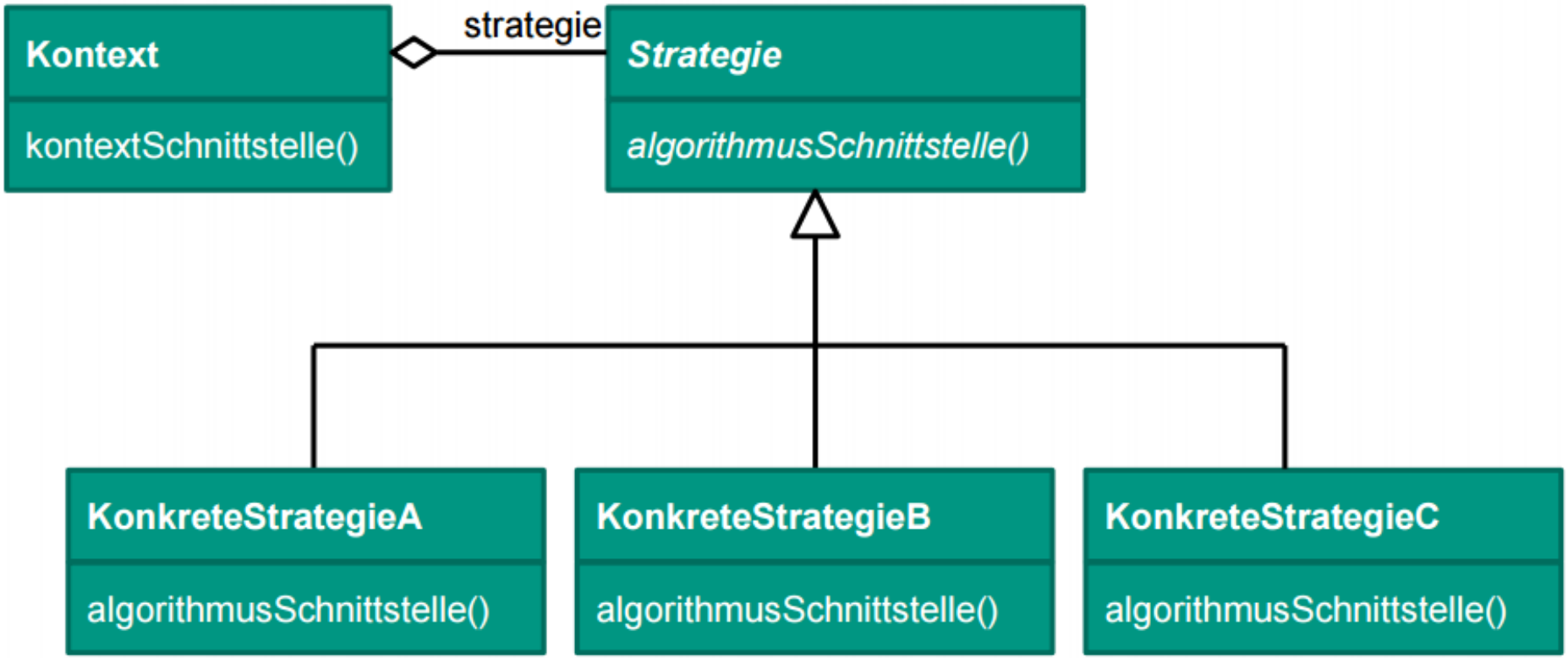
Jedes Modul verbirgt eine wichtige Entwurfsentscheidung hinter einer wohldefinierten Schnittstelle, die sich bei einer Änderung der Entscheidung nicht mit ändert.

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

Quiz (Ankreuzaufgaben aus Klausuren)

Wahr oder falsch?

- Das Entwurfsmuster Strategie bietet die Möglichkeit, eine Klasse mit einer von mehreren möglichen Verhaltensweisen zu konfigurieren.

Quiz (Ankreuzaufgaben aus Klausuren)

Wahr oder falsch?

- Das Entwurfsmuster Strategie bietet die Möglichkeit, eine Klasse mit einer von mehreren möglichen Verhaltensweisen zu konfigurieren.
  wahr

Quiz (Ankreuzaufgaben aus Klausuren)

Wahr oder falsch?

- Das Entwurfsmuster Strategie bietet die Möglichkeit, eine Klasse mit einer von mehreren möglichen Verhaltensweisen zu konfigurieren.
  wahr
- Das Strategiemuster erfüllt das Geheimnisprinzip, indem es Datenstrukturen, die in einer konkreten Strategie enthalten sind, vor dem Klienten verbirgt.

Quiz (Ankreuzaufgaben aus Klausuren)

Wahr oder falsch?

- Das Entwurfsmuster Strategie bietet die Möglichkeit, eine Klasse mit einer von mehreren möglichen Verhaltensweisen zu konfigurieren. wahr
- Das Strategiemuster erfüllt das Geheimnisprinzip, indem es Datenstrukturen, die in einer konkreten Strategie enthalten sind, vor dem Klienten verbirgt. wahr

Quiz (Ankreuzaufgaben aus Klausuren)

Wahr oder falsch?

- Das Entwurfsmuster Strategie bietet die Möglichkeit, eine Klasse mit einer von mehreren möglichen Verhaltensweisen zu konfigurieren. wahr
- Das Strategiemuster erfüllt das Geheimnisprinzip, indem es Datenstrukturen, die in einer konkreten Strategie enthalten sind, vor dem Klienten verbirgt. wahr
- Das Muster Strategie kapselt austauschbares Verhalten und verwendet Delegierung, um zu entscheiden, welches Verhalten verwendet wird.

Quiz (Ankreuzaufgaben aus Klausuren)

Wahr oder falsch?

- Das Entwurfsmuster Strategie bietet die Möglichkeit, eine Klasse mit einer von mehreren möglichen Verhaltensweisen zu konfigurieren. wahr
- Das Strategiemuster erfüllt das Geheimnisprinzip, indem es Datenstrukturen, die in einer konkreten Strategie enthalten sind, vor dem Klienten verbirgt. wahr
- Das Muster Strategie kapselt austauschbares Verhalten und verwendet Delegierung, um zu entscheiden, welches Verhalten verwendet wird. wahr

Quiz (Ankreuzaufgaben aus Klausuren)

Wahr oder falsch?

- Das Entwurfsmuster Strategie bietet die Möglichkeit, eine Klasse mit einer von mehreren möglichen Verhaltensweisen zu konfigurieren. wahr
- Das Strategiemuster erfüllt das Geheimnisprinzip, indem es Datenstrukturen, die in einer konkreten Strategie enthalten sind, vor dem Klienten verbirgt. wahr
- Das Muster Strategie kapselt austauschbares Verhalten und verwendet Delegierung, um zu entscheiden, welches Verhalten verwendet wird. wahr
- Das Hinzufügen einer neuen konkreten Strategie erfordert keine Änderung existierender konkreter Strategien.

Quiz (Ankreuzaufgaben aus Klausuren)

Wahr oder falsch?

- Das Entwurfsmuster Strategie bietet die Möglichkeit, eine Klasse mit einer von mehreren möglichen Verhaltensweisen zu konfigurieren. wahr
- Das Strategiemuster erfüllt das Geheimnisprinzip, indem es Datenstrukturen, die in einer konkreten Strategie enthalten sind, vor dem Klienten verbirgt. wahr
- Das Muster Strategie kapselt austauschbares Verhalten und verwendet Delegierung, um zu entscheiden, welches Verhalten verwendet wird. wahr
- Das Hinzufügen einer neuen konkreten Strategie erfordert keine Änderung existierender konkreter Strategien. wahr

Entkopplungs-Muster

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

Adapter

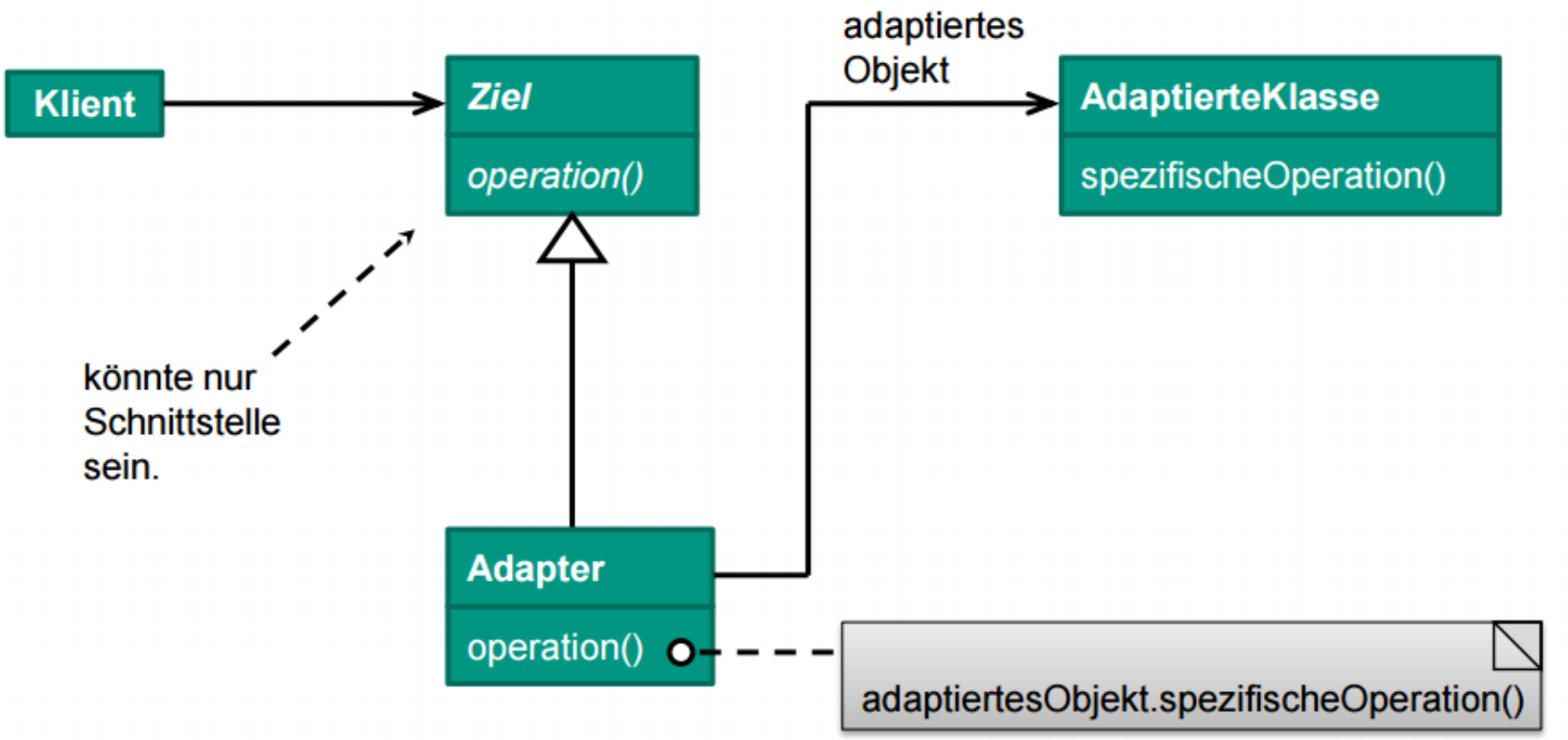
Problem

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

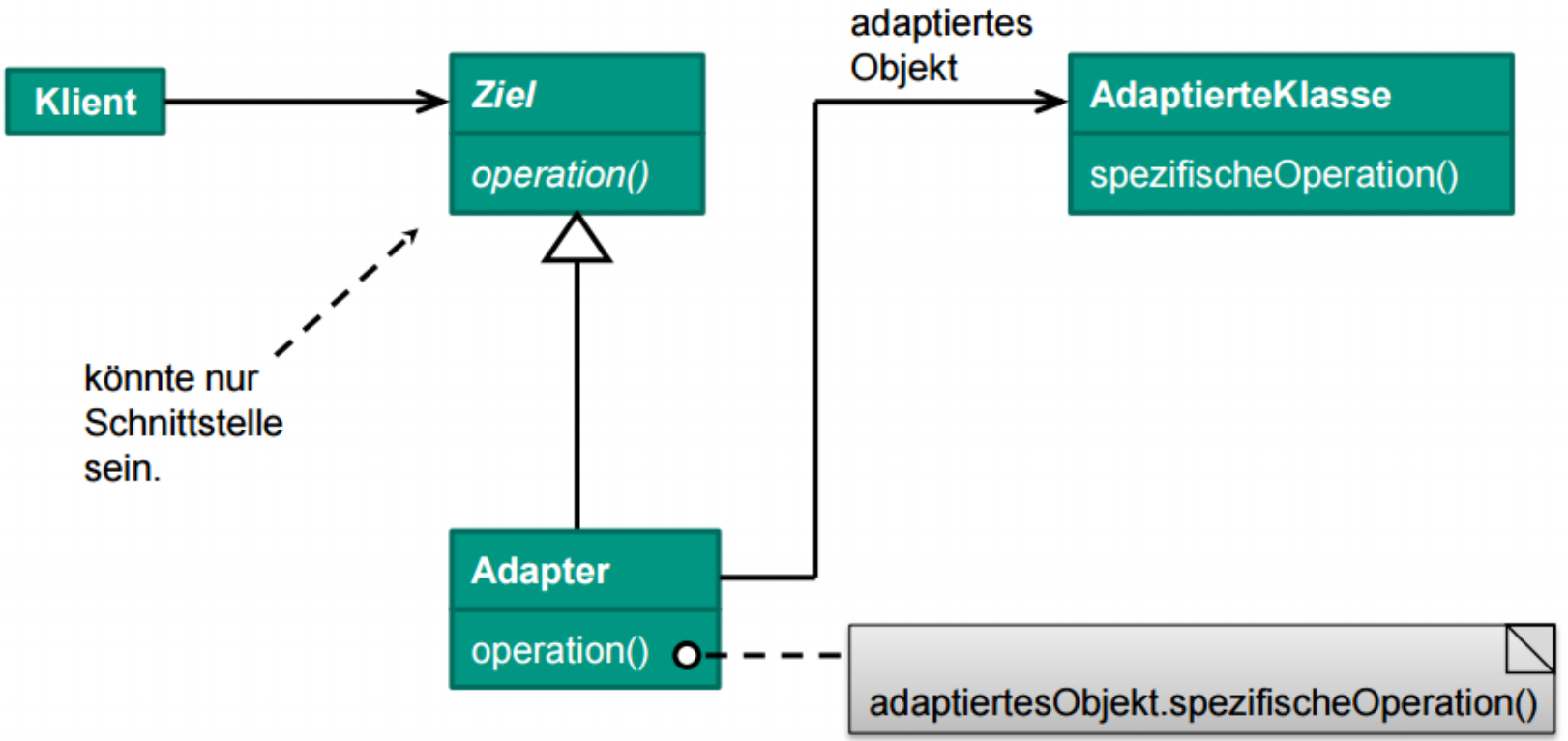
Adapter

Problem

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



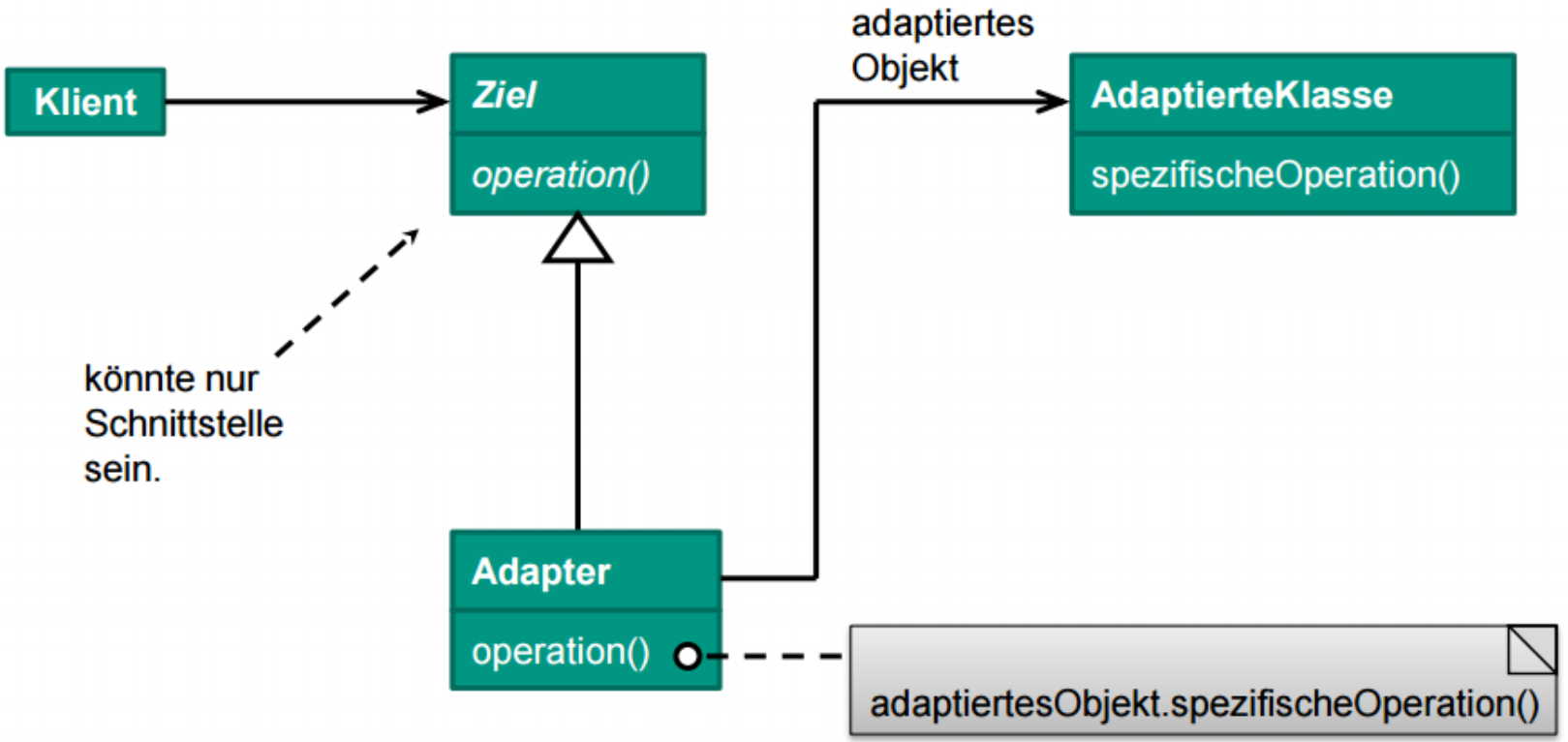
Adapter (Objektadapter)



Wir sind bei Entkopplung-Mustern, Preisfrage:

Wo ist hier die Entkopplung?

Adapter (Objektadapter)

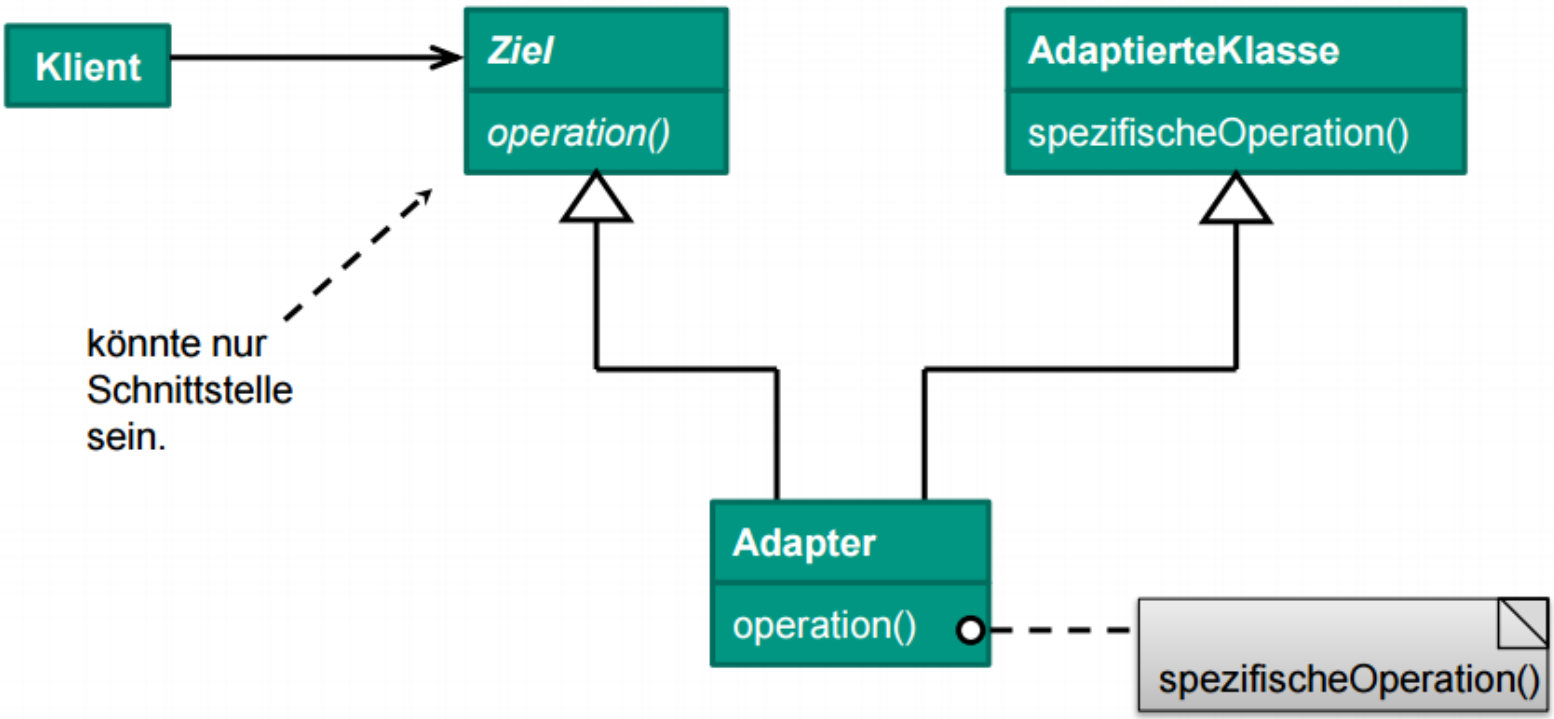


Wir sind bei Entkopplung-Mustern, Preisfrage:

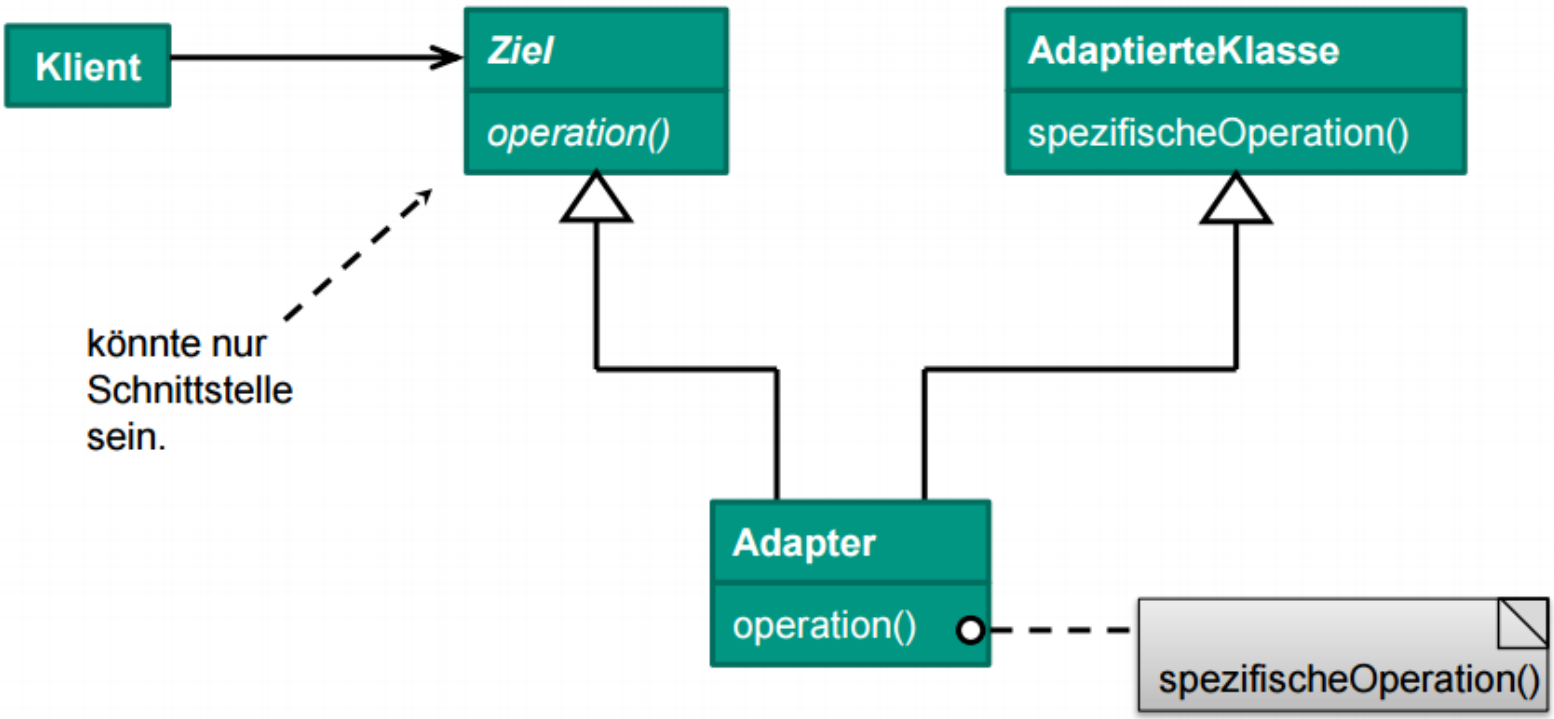
Wo ist hier die Entkopplung?

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

Adapter - Alternative (Klassenadapter)

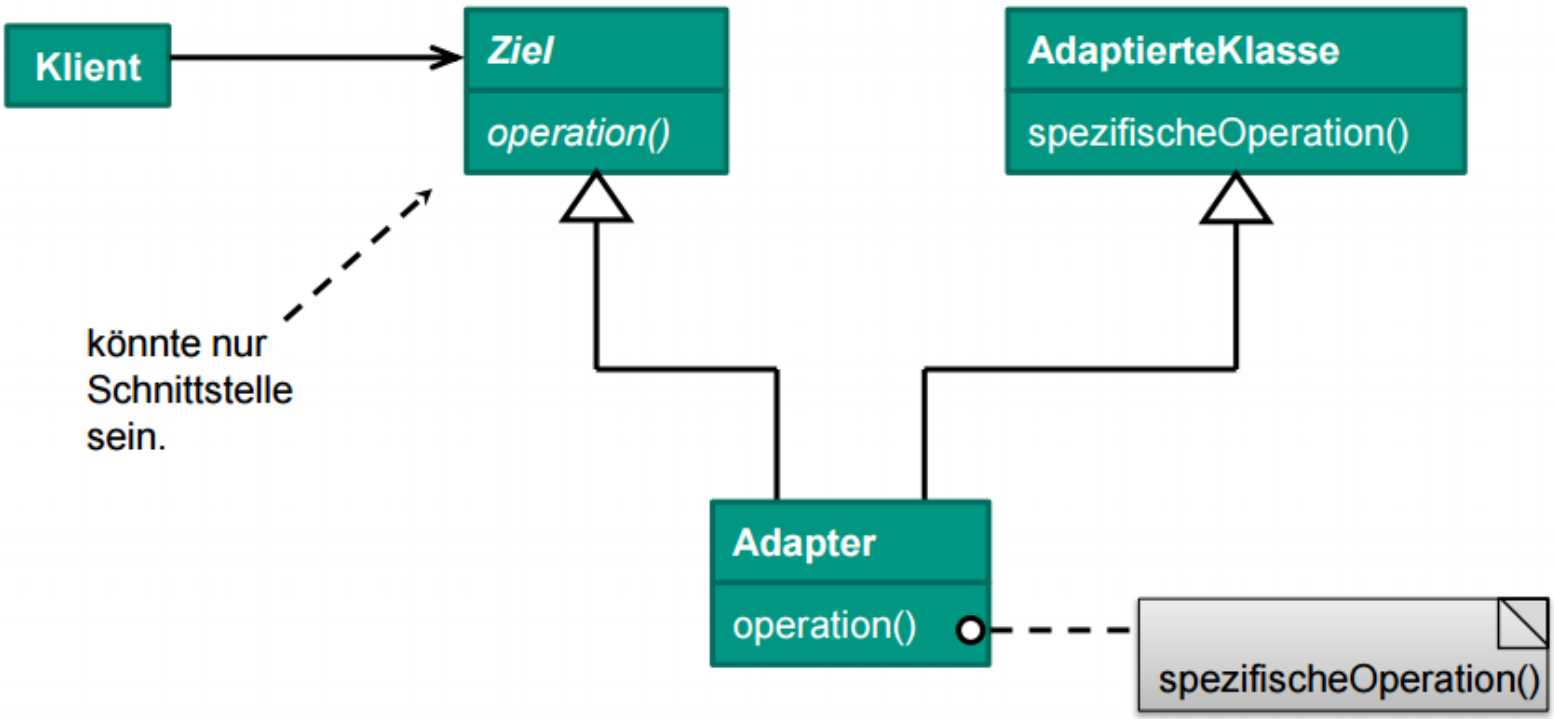


Adapter - Alternative (Klassenadapter)



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

Adapter - Alternative (Klassenadapter)



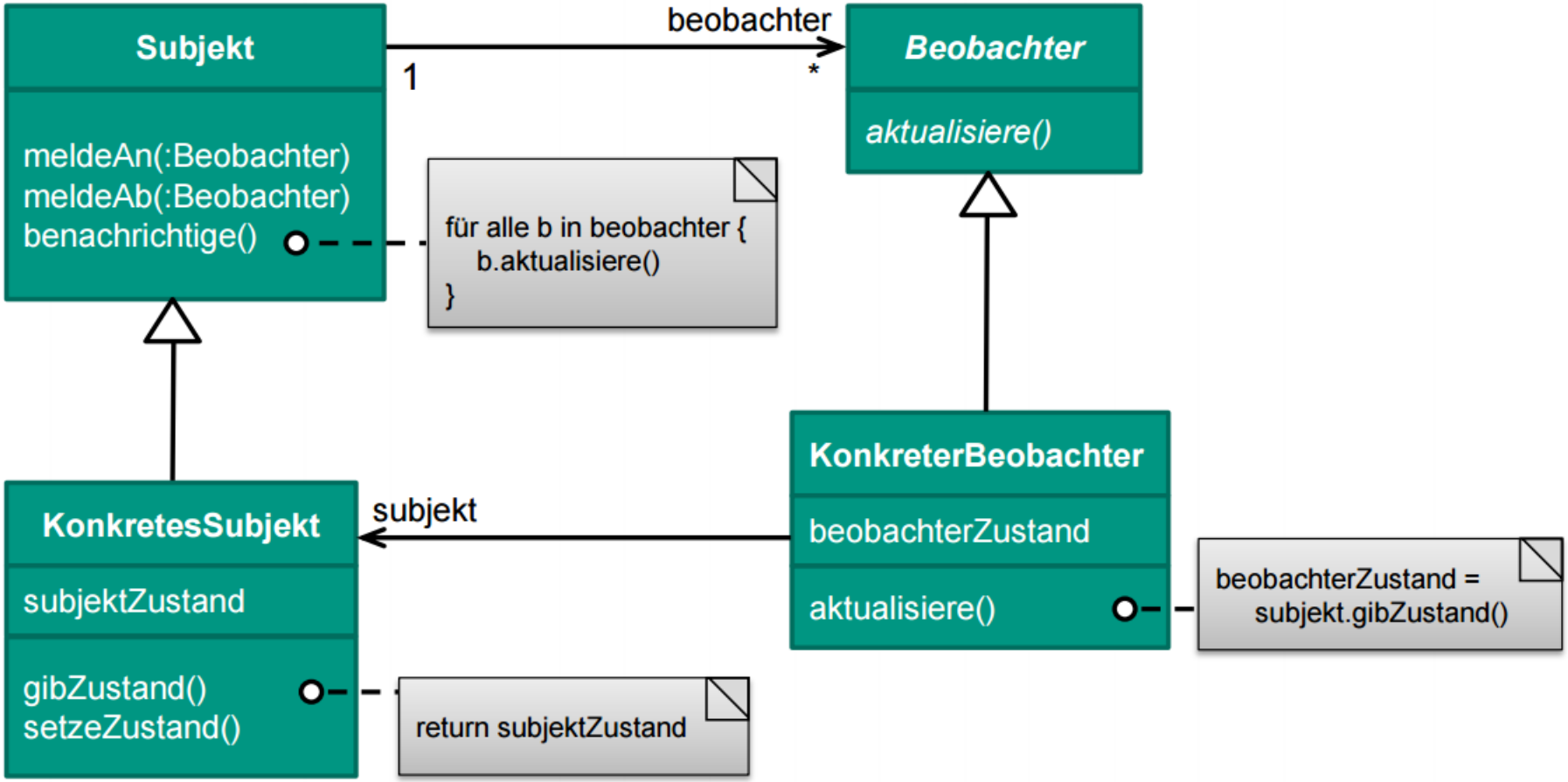
Was für ein Problem bekommt ihr, wenn ihr das auf einem ÜB implementieren müsst?

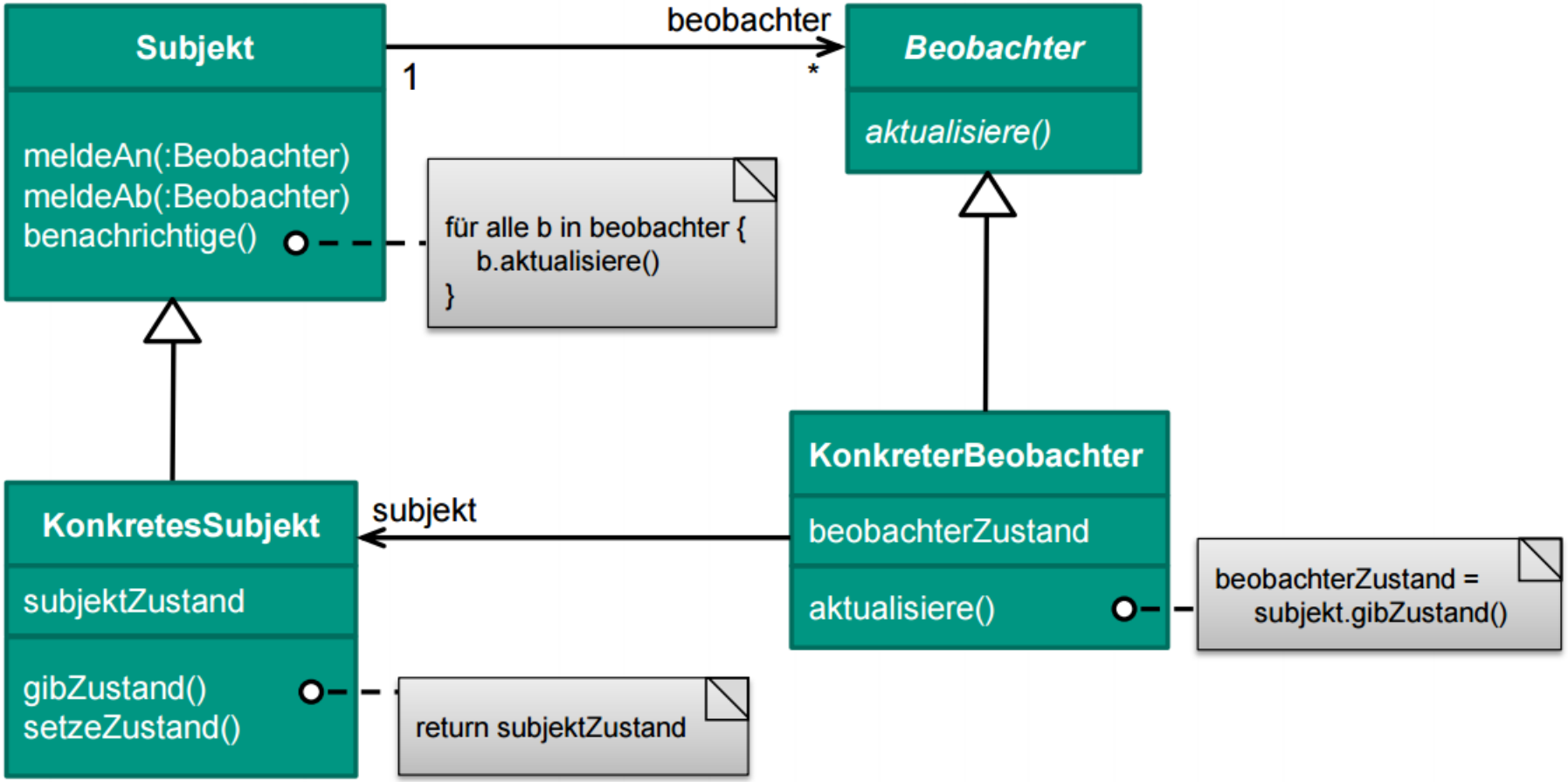
$\Longrightarrow$ keine Mehrfachvererbung in Java!

Beobachter/Observer: abstrakt

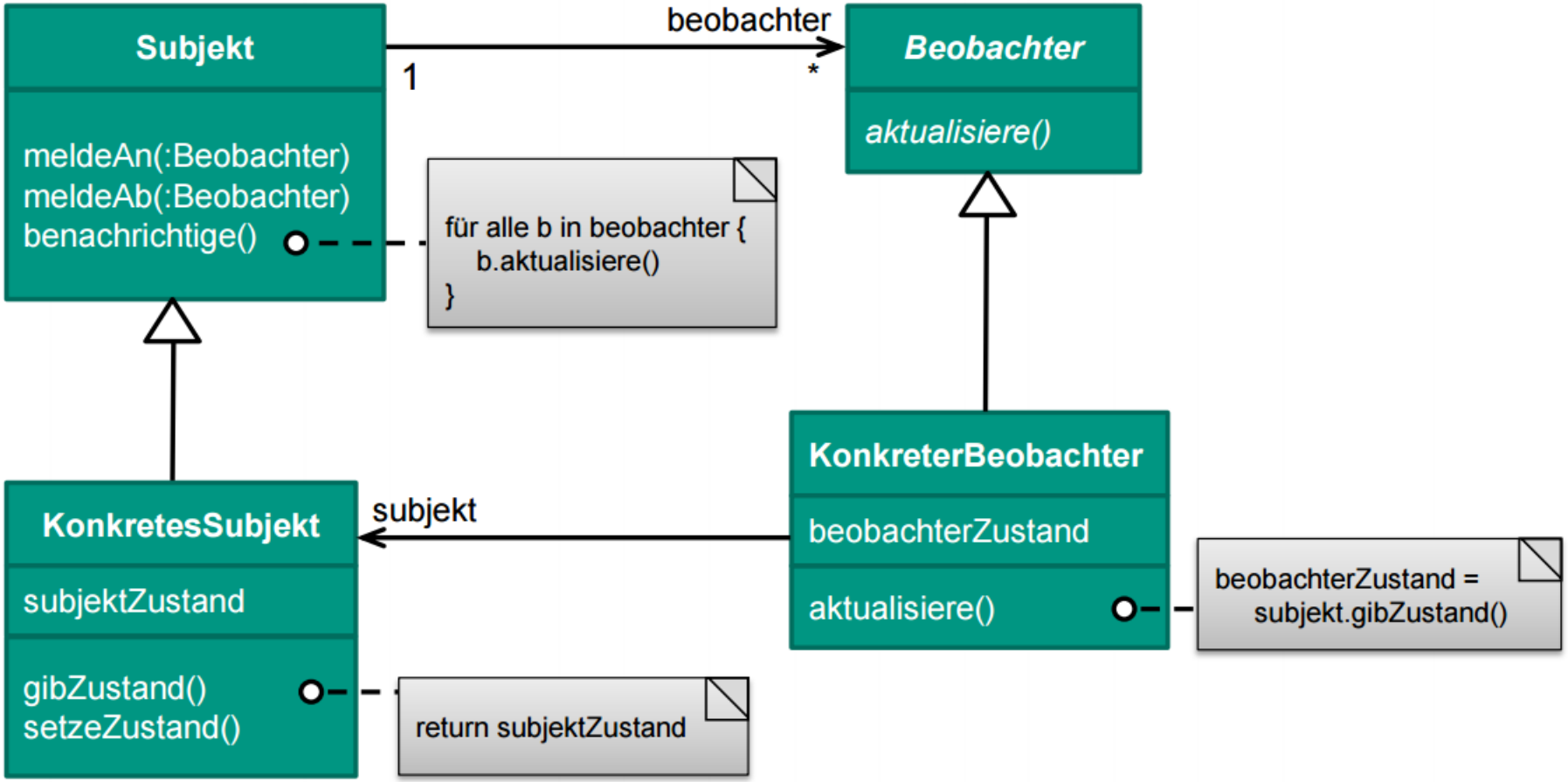
Problem

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


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

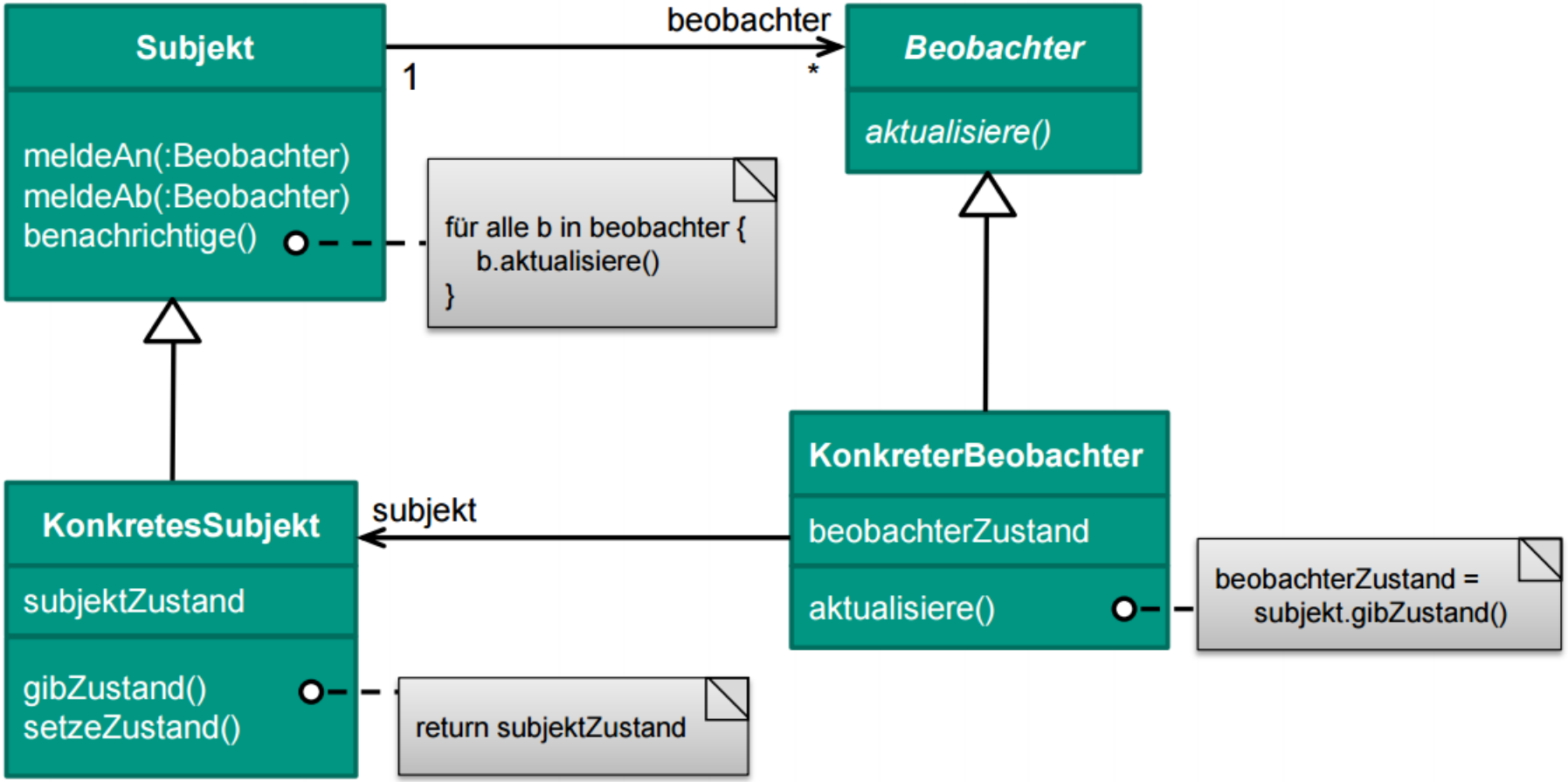


Entkopplung?



Entkopplung?

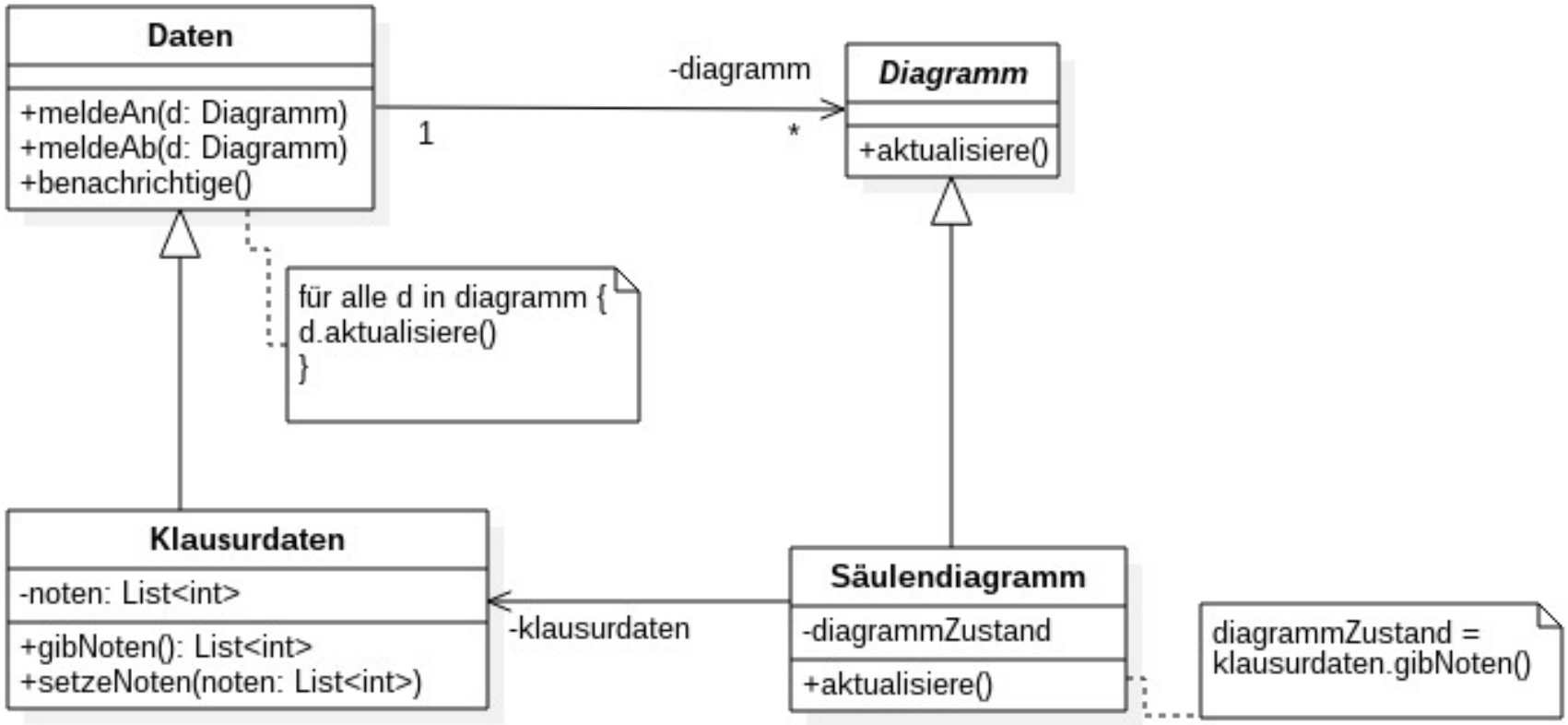
- jeder Beobachter definiert, was bei Benachrichtigung passiert, Subjekt kriegt davon nichts mit



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

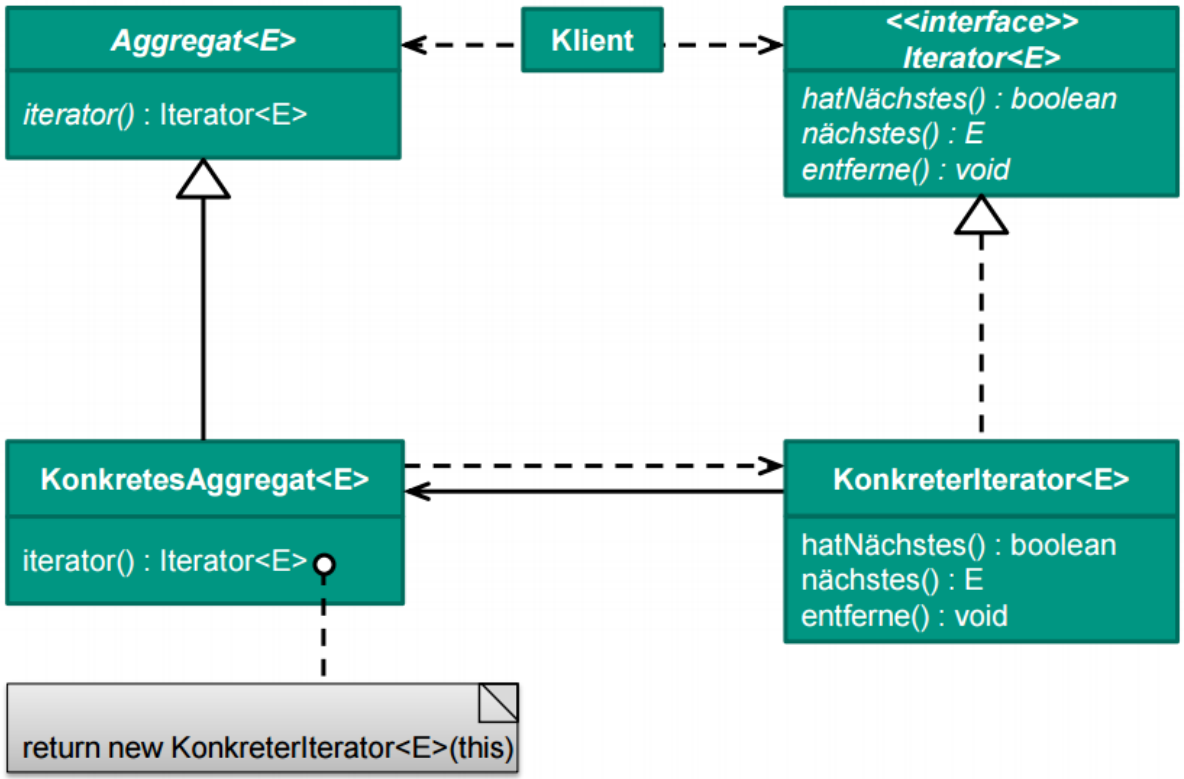
Problem

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

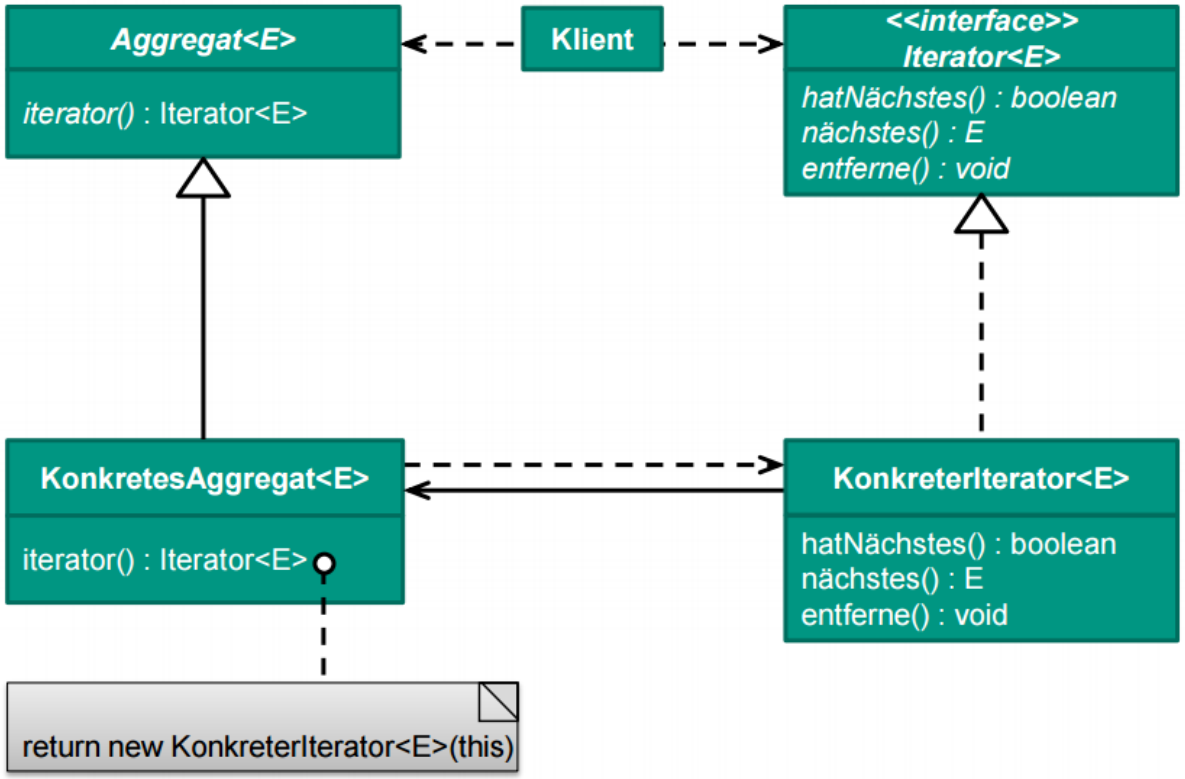
Iterator

Problem

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

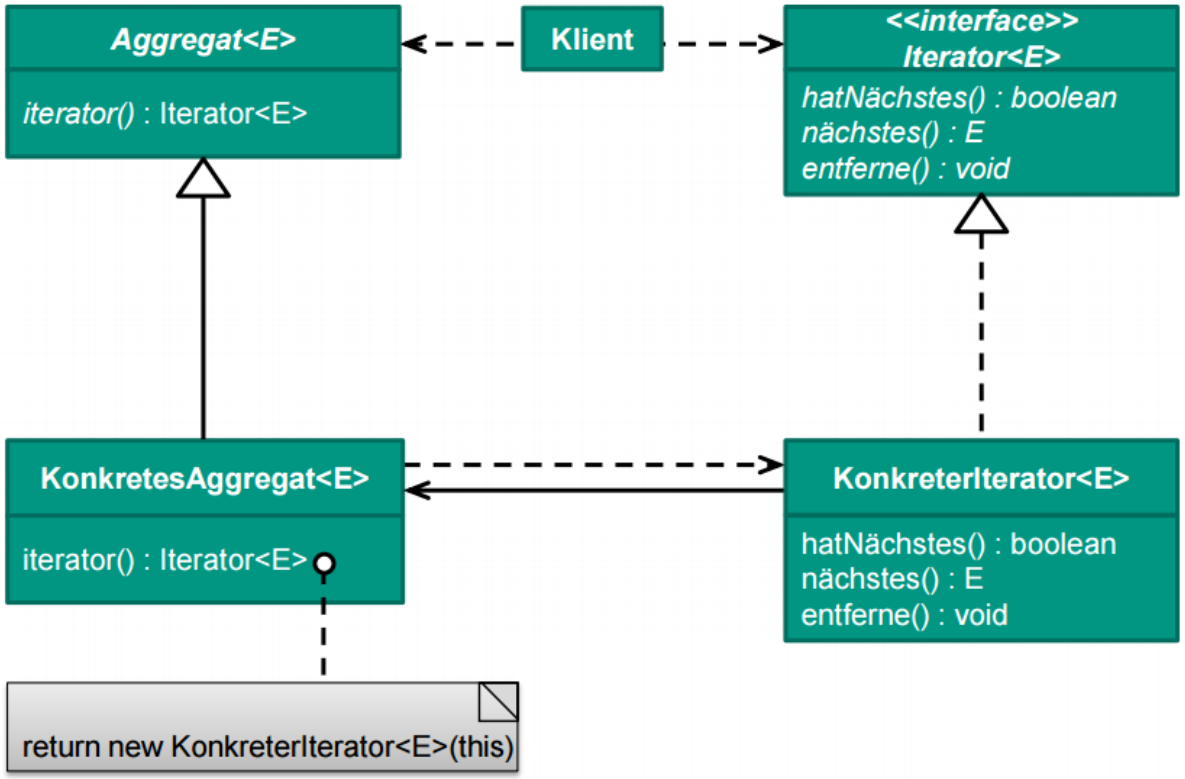


Iterator



Entkopplung?

Iterator



Entkopplung?

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

Quiz (Ankreuzaufgaben aus Klausuren)

Wahr oder falsch?

- Klienten können mithilfe des Iterator-Musters Sammlungen von Objekten und einzelne Objekte auf die gleiche Weise behandeln.

Quiz (Ankreuzaufgaben aus Klausuren)

Wahr oder falsch?

- Klienten können mithilfe des Iterator-Musters Sammlungen von Objekten und einzelne Objekte auf die gleiche Weise behandeln.
  falsch

Quiz (Ankreuzaufgaben aus Klausuren)

Wahr oder falsch?

- Klienten können mithilfe des Iterator-Musters Sammlungen von Objekten und einzelne Objekte auf die gleiche Weise behandeln.
  falsch
- Das Entwurfsmuster Iterator ist den Variantenmustern zuzuordnen.

Quiz (Ankreuzaufgaben aus Klausuren)

Wahr oder falsch?

- Klienten können mithilfe des Iterator-Musters Sammlungen von Objekten und einzelne Objekte auf die gleiche Weise behandeln.
  falsch
- Das Entwurfsmuster Iterator ist den Variantenmustern zuzuordnen.
  falsch

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

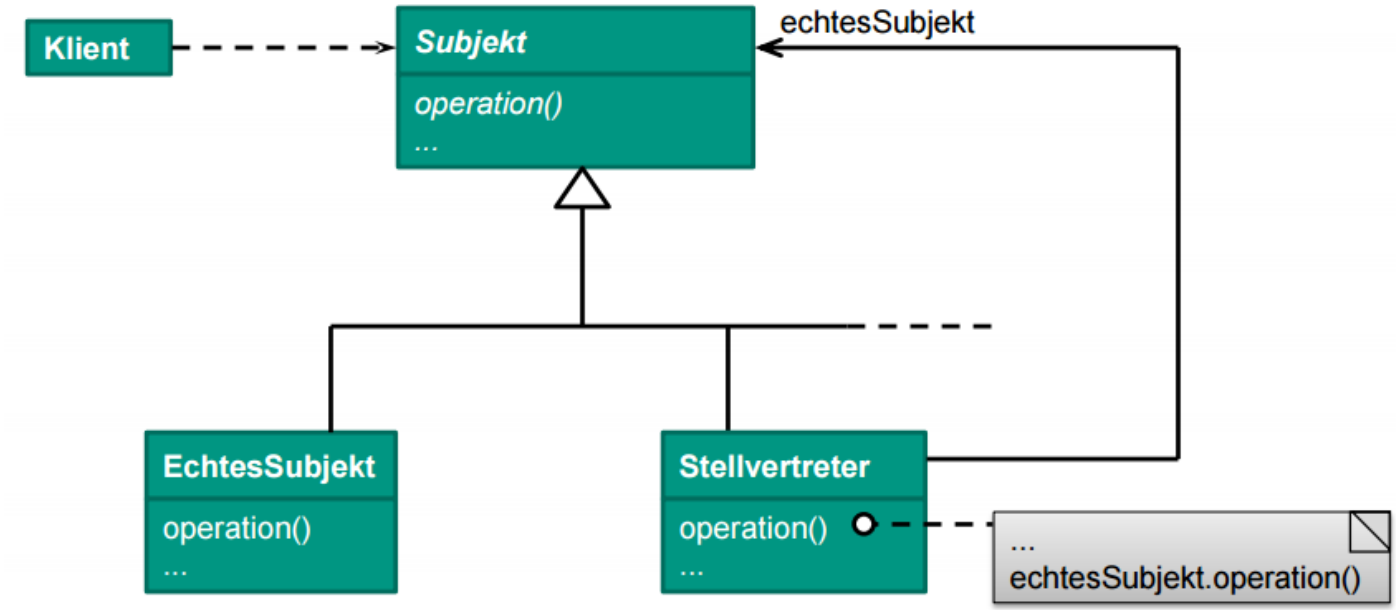
  $\Longrightarrow$ Stellvertreter macht Zugriffskontrolle

Stellvertreter

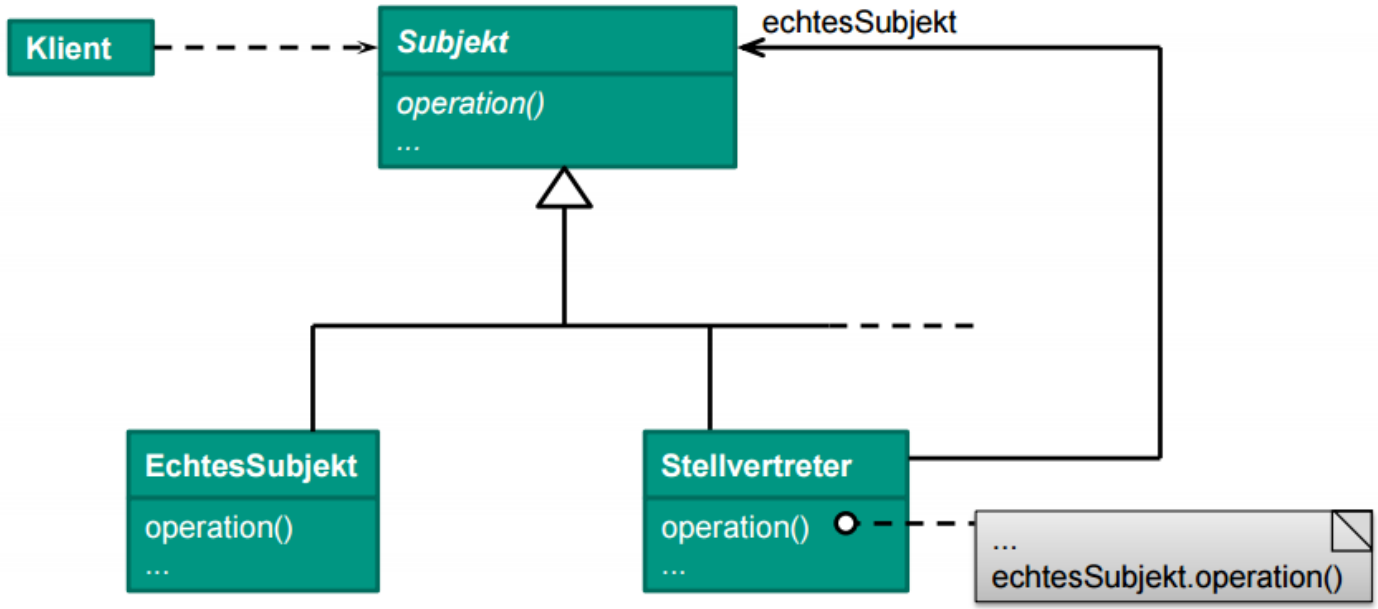
Problem

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

  $\Longrightarrow$ Stellvertreter macht Zugriffskontrolle

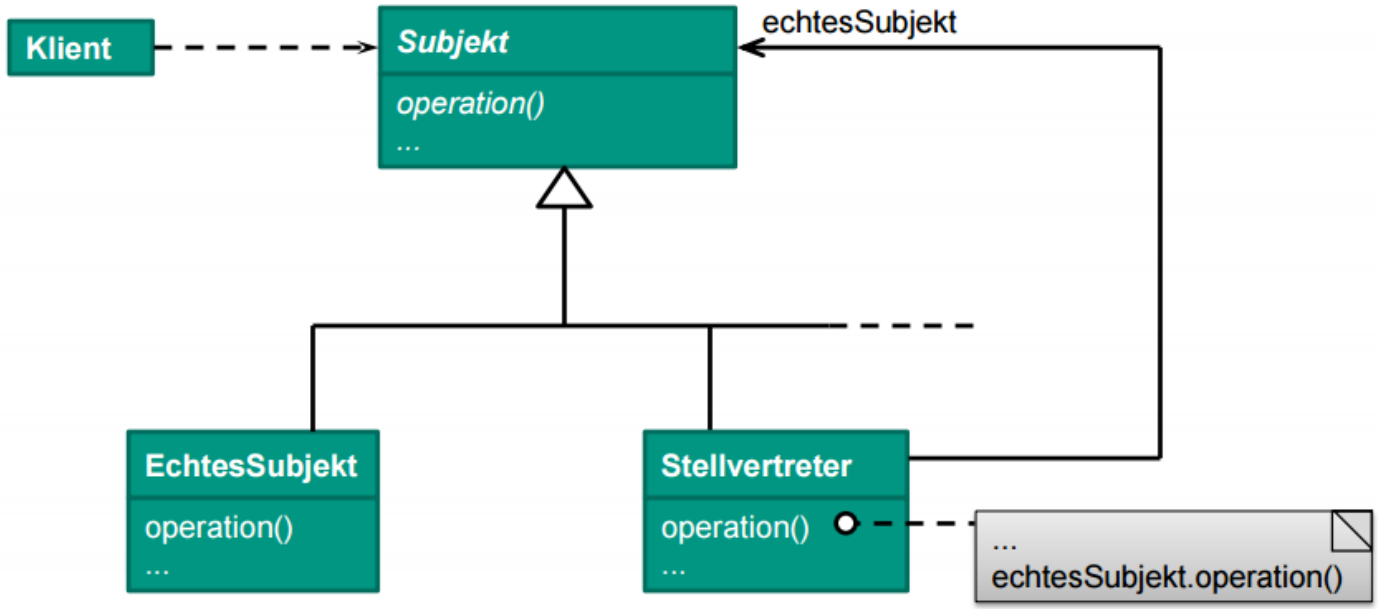


Stellvertreter



Entkopplung?

Stellvertreter



Entkopplung?

- Klient hat keinen direkten Zugriff auf das echte Subjekt

Vermittler

Problem

- mehrere voneinander abhängige Objekte

Vermittler

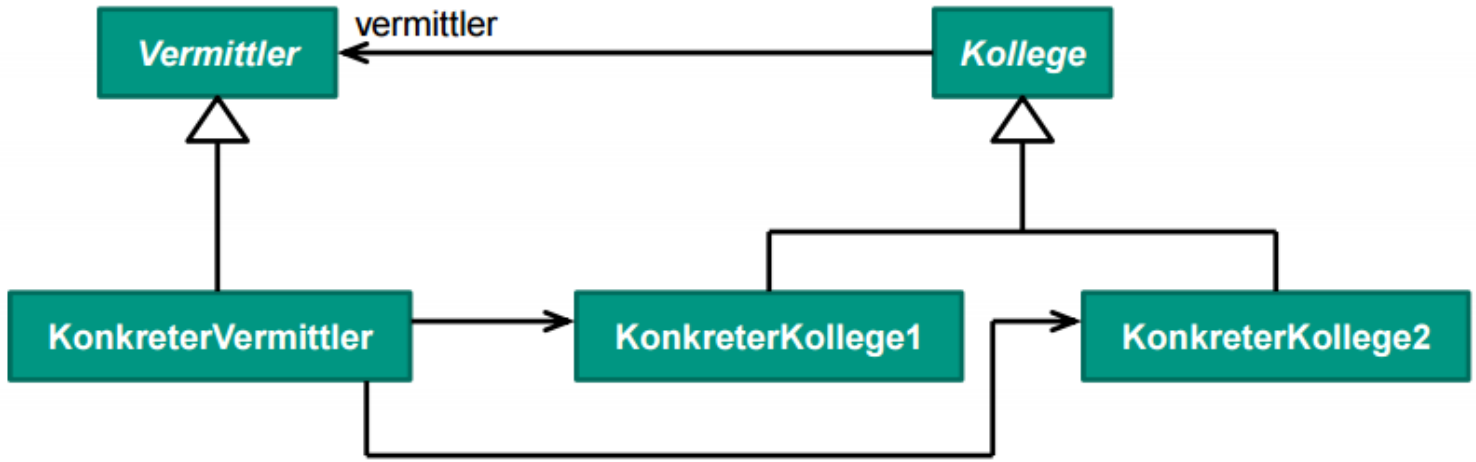
Problem

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

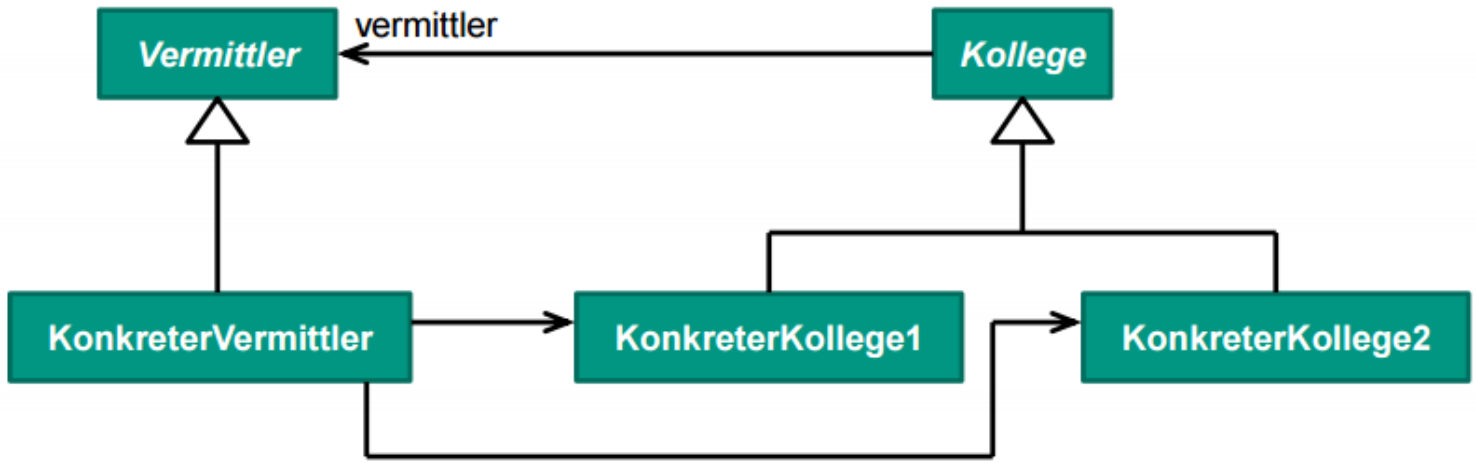
Vermittler

Problem

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

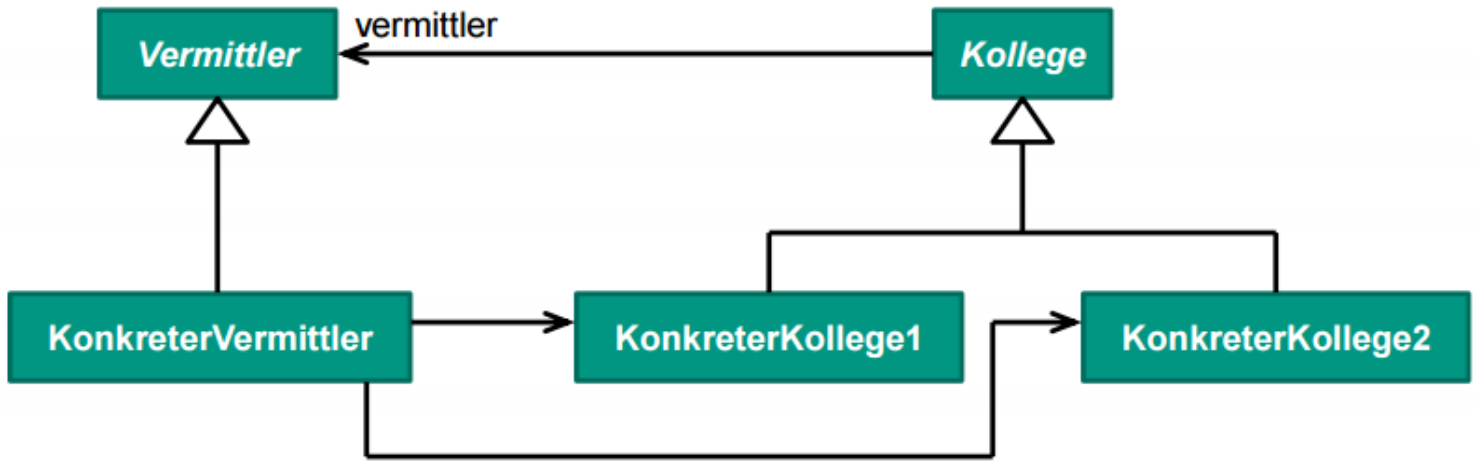


Vermittler



Entkopplung?

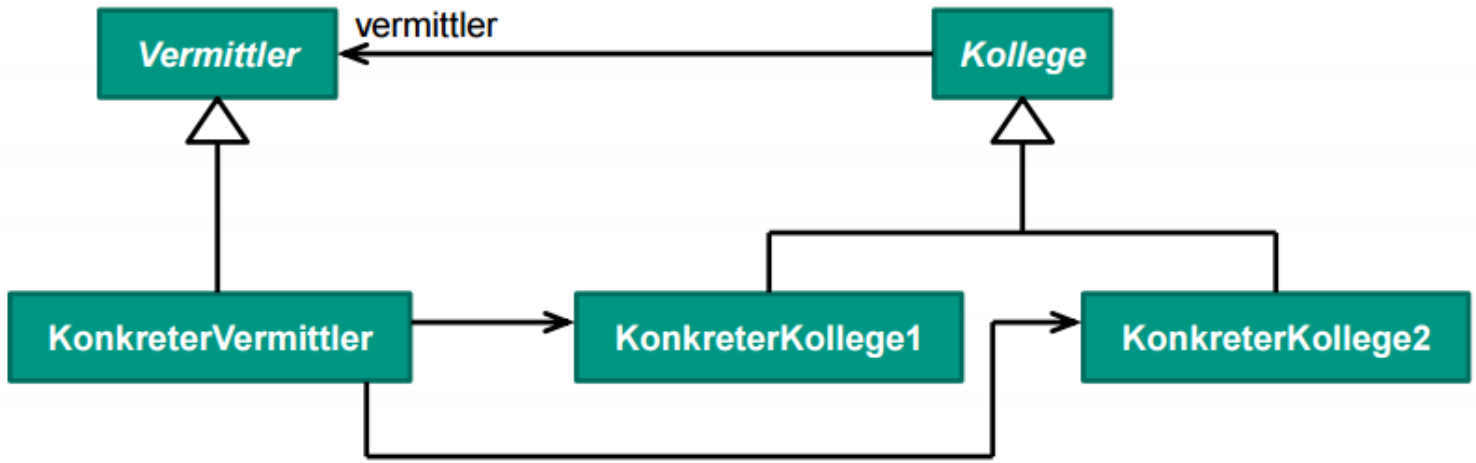
Vermittler



Entkopplung?

- Kollegen kennen sich nicht direkt

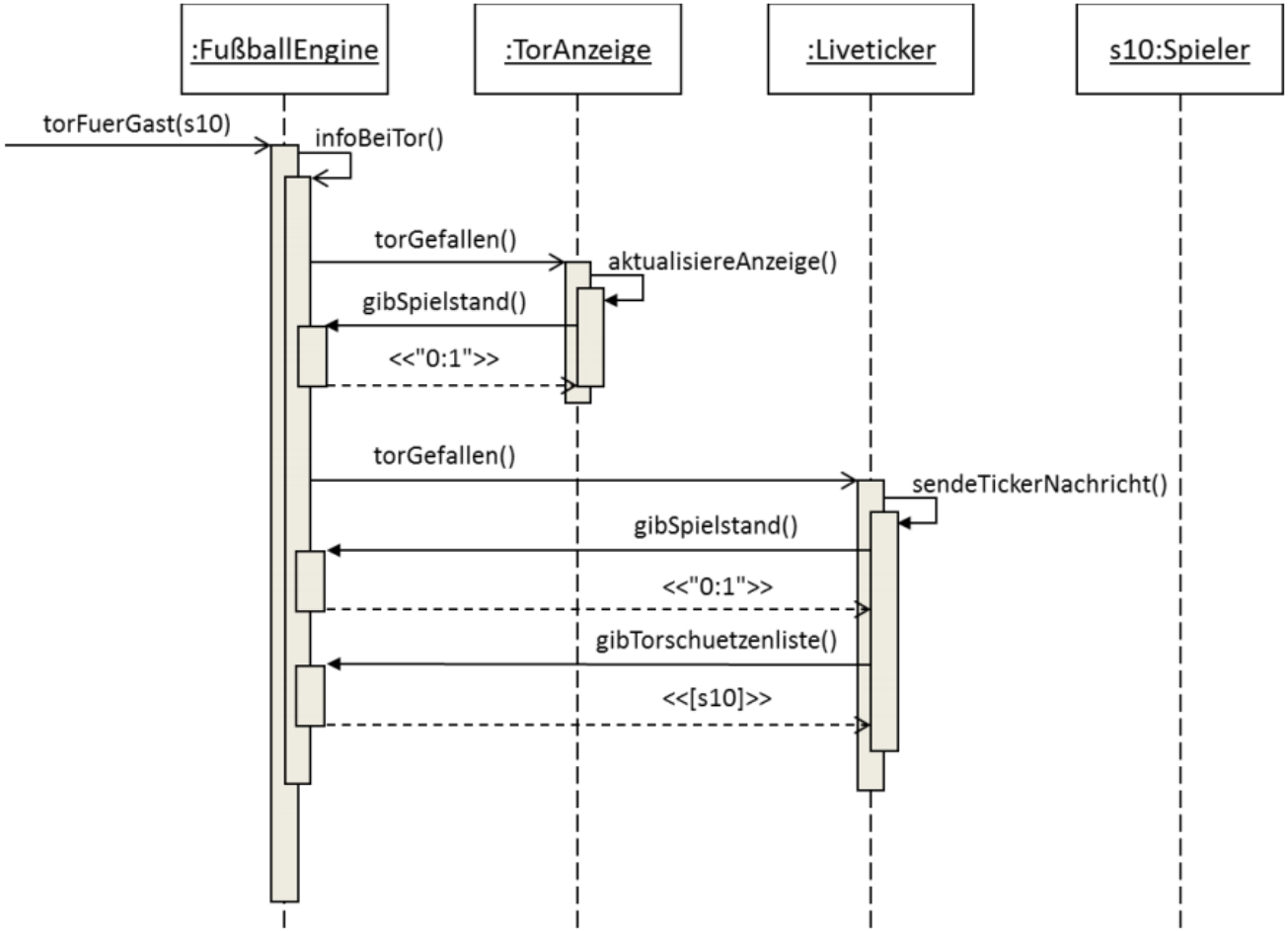
Vermittler



Entkopplung?

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

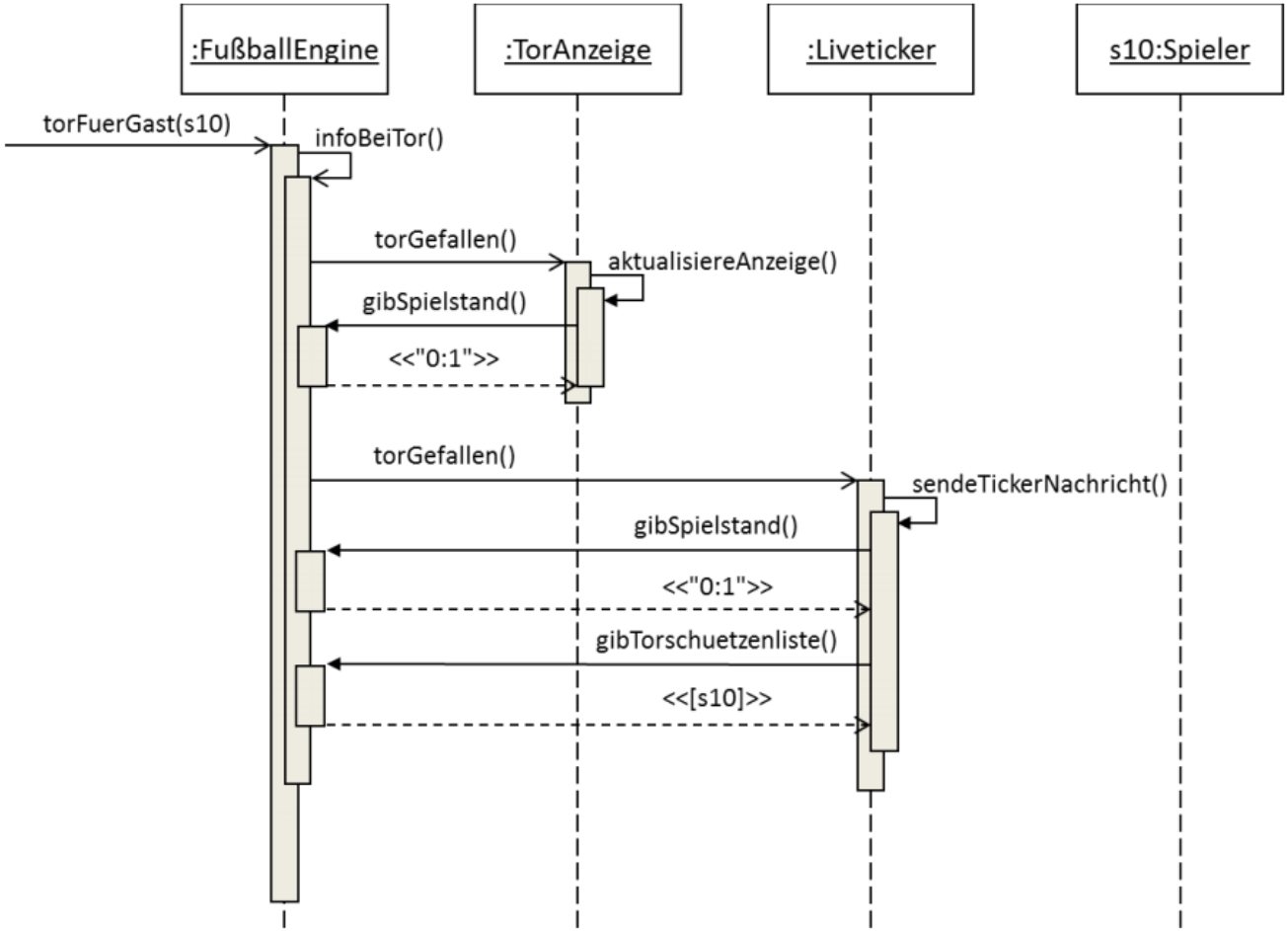
Klausuraufgabe (Hauptklausur SS 2012)



Aufgabe 1

Welches Entwurfsmuster erkennen Sie in diesem Diagramm?

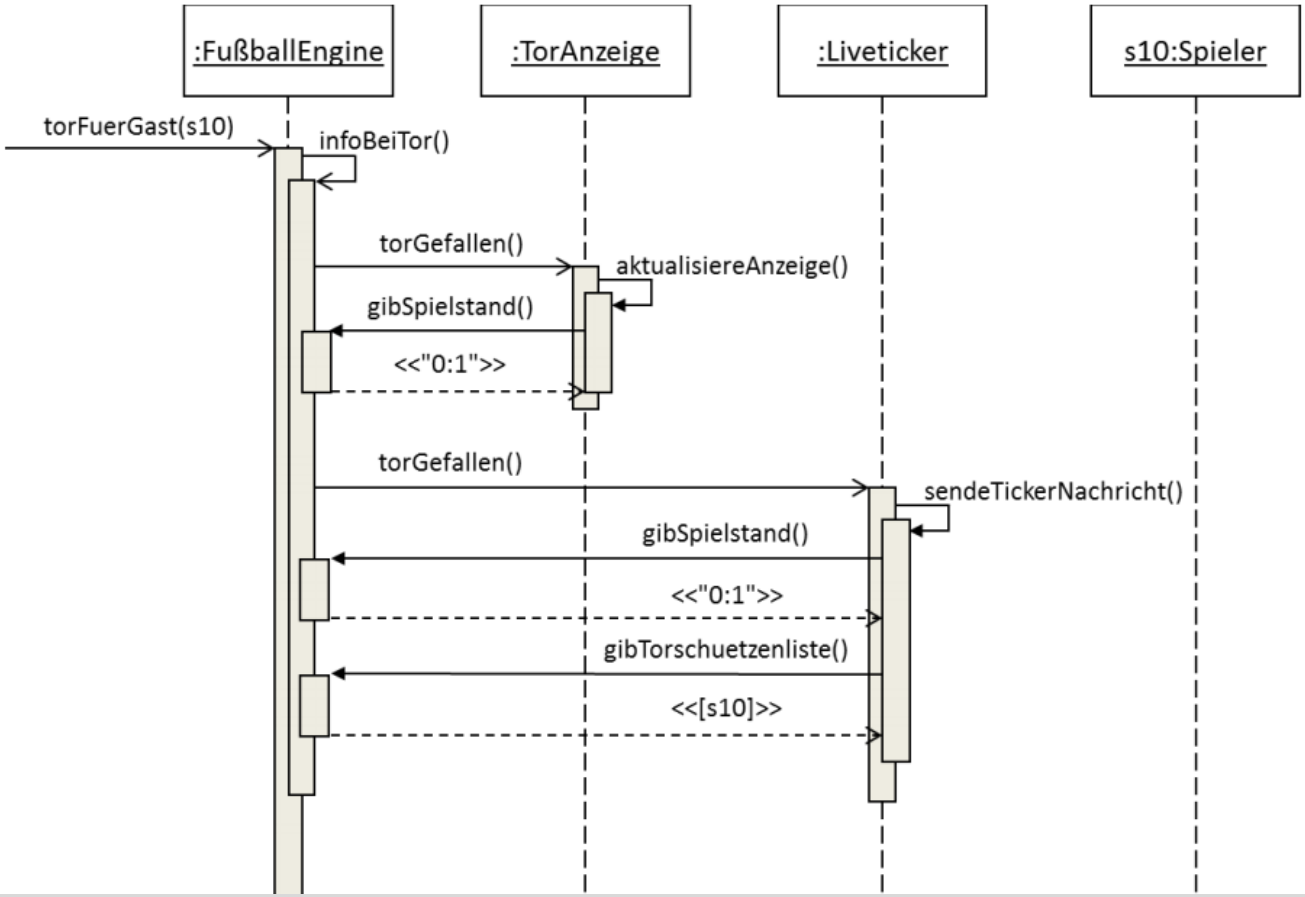
Klausuraufgabe (Hauptklausur SS 2012)



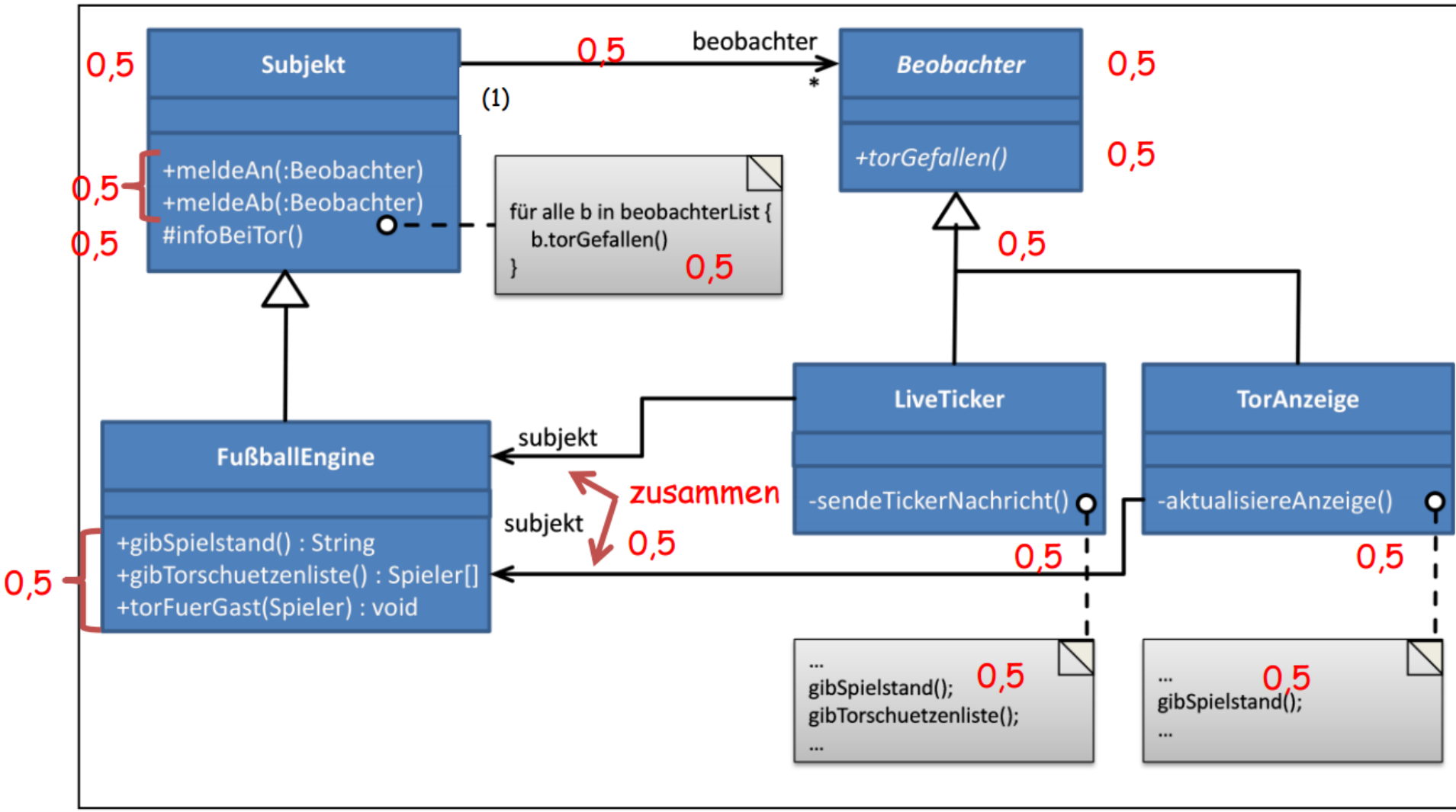
Aufgabe 1

Welches Entwurfsmuster erkennen Sie in diesem Diagramm? Beobachter.

Entwerfen Sie das folgende Klassendiagramm passend zu dem Sequenzdiagramm; es soll alle verwendeten Klassen und Methoden enthalten. Kennzeichnen Sie die Zugreifbarkeiten der Methoden mit den Symbolen +, -, #; seien Sie dabei möglichst restriktiv. Verzichten Sie auf die Modellierung von Attributen. Kennzeichnen Sie die Elemente des Entwurfsmusters und deren Funktion.



Musterlösung



Tipps - 4. Übungsblatt

Aufgabe 1: Zustandsdiagramm (LEZ)

- nochmal Syntax anschauen

  $\Longrightarrow$ Was darf in Zustandsdiagramm, was nicht? (laut VL)

Tipps - 4. Übungsblatt

Aufgabe 1: Zustandsdiagramm (LEZ)

- nochmal Syntax anschauen
  $\Longrightarrow$ Was darf in Zustandsdiagramm, was nicht? (laut VL)

Aufgabe 2: Die Abbottsche Methode

- back to Deutsch-Unterricht
  $\Longrightarrow$ prinzipiell nicht schwierig

Tipps - 4. Übungsblatt

Aufgabe 3: iMage-GUI

- macht die “kleinen“ Bonusaufgaben
  $\Longrightarrow$ relativ leichte Punkte

Tipps - 4. Übungsblatt

Aufgabe 3: iMage-GUI

- macht die “kleinen“ Bonusaufgaben
  $\Longrightarrow$ relativ leichte Punkte
- schaut euch die verschiedenen LayoutManager aus Java Swing an
  $\Longrightarrow$ verschiedene LayoutManager möglich (via mehrerer Container, z.B. JPanel)

Tipps - 4. Übungsblatt

Aufgabe 3: iMage-GUI

- macht die “kleinen“ Bonusaufgaben
  $\Longrightarrow$ relativ leichte Punkte
- schaut euch die verschiedenen LayoutManager aus Java Swing an
  $\Longrightarrow$ verschiedene LayoutManager möglich (via mehrerer Container, z.B. JPanel)

Aufgabe 4: Geheimnisprinzip

- leichte Punkte
- Attribute sollten ?? sein?
  $\Longrightarrow$ Und warum nochmal?

Denkt dran!

Abgabe

- Deadline am 21.6 um 12:00
- A{1,2,4} handschriftlich!

Bis dann! (dann := 26.06.17)