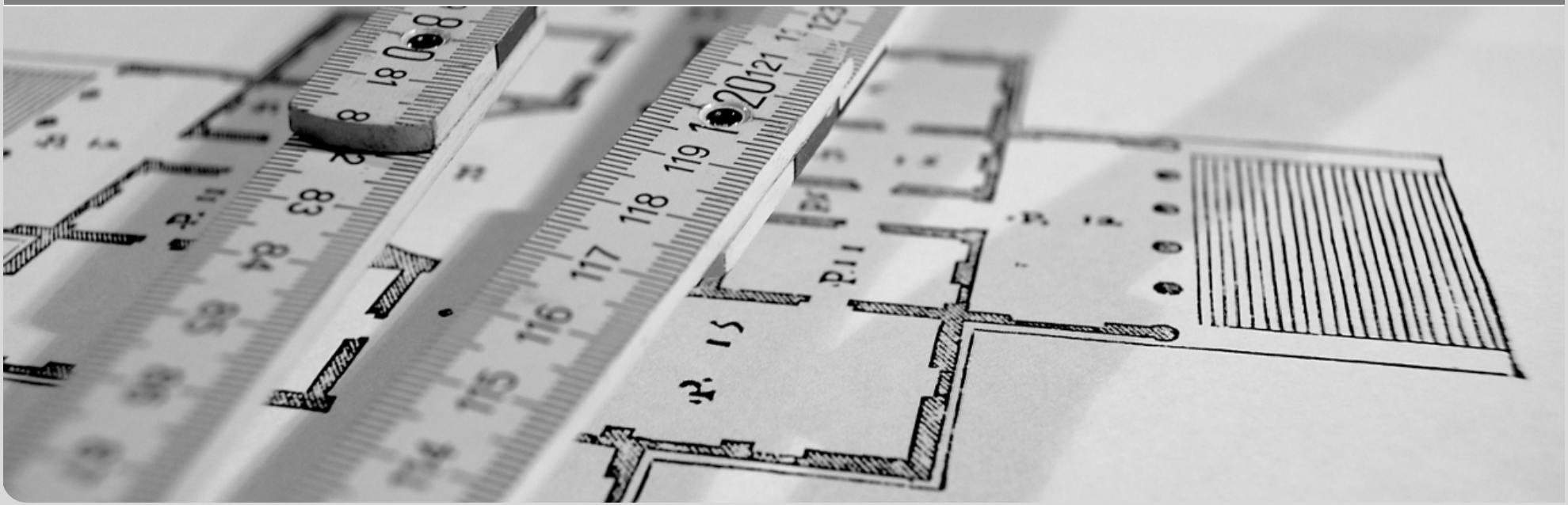
Karlsruher Institut für Technologie

Softwaretechnik 1 - 4. Tutorium

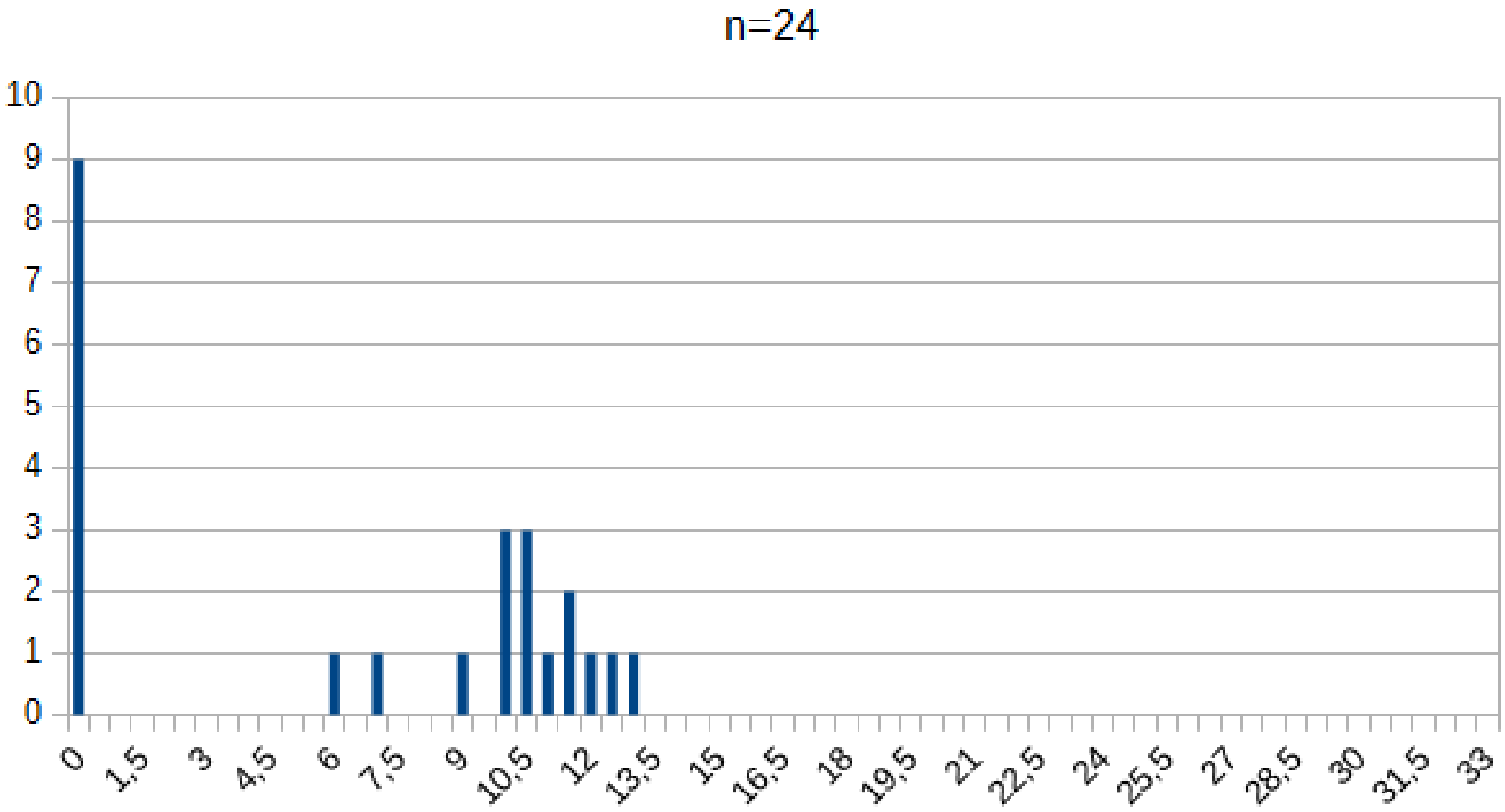
Tutorium 03

Felix Bachmann | 26.06.2017

KIT - INSTITUT FÜR PROGRAMMSTRUKTUREN UND DATENORGANISATION (IPD)



4. Übungsblatt Statistik



∅ 6,45 bzw. 10,33 von (bisher) 15+1

Häufige Fehler

Allgemein

- Form bei handschriftlichen Abgaben. . .

Häufige Fehler

Aufgabe 1 (Zustandsdiagramm - LEZ): ∅ 2,81 bzw. 4,5 von 5+1

Häufige Fehler

Aufgabe 1 (Zustandsdiagramm - LEZ): ∅ 2,81 bzw. 4,5 von 5+1

- Hierarchie sinnvoll, wenn aus mehreren Zuständen gleiche Übergänge in den gleichen Zustand gehen

Häufige Fehler

Aufgabe 1 (Zustandsdiagramm - LEZ): ∅ 2,81 bzw. 4,5 von 5+1

- Hierarchie sinnvoll, wenn aus mehreren Zuständen gleiche Übergänge in den gleichen Zustand gehen
- nach VL gibt es im Zustandsdiagramm kein “Karo“

Häufige Fehler

Aufgabe 1 (Zustandsdiagramm - LEZ): ∅ 2,81 bzw. 4,5 von 5+1

- Hierarchie sinnvoll, wenn aus mehreren Zuständen gleiche Übergänge in den gleichen Zustand gehen
- nach VL gibt es im Zustandsdiagramm kein “Karo“
- “Versehen Sie die Zustandsübergänge mit Ereignissen **und** Operationen.“

Häufige Fehler

Aufgabe 1 (Zustandsdiagramm - LEZ): ∅ 2,81 bzw. 4,5 von 5+1

- Hierarchie sinnvoll, wenn aus mehreren Zuständen gleiche Übergänge in den gleichen Zustand gehen
- nach VL gibt es im Zustandsdiagramm kein “Karo“
- “Versehen Sie die Zustandsübergänge mit Ereignissen **und** Operationen.“
  $\Longrightarrow$ kann in Klausur bei Nichtbeachtigung Punktabzug geben

Häufige Fehler

Aufgabe 2 (Abbottsche Methode): ∅ 1,73 bzw. 3,19 von 5

Häufige Fehler

Aufgabe 2 (Abbottsche Methode): ∅ 1,73 bzw. 3,19 von 5

- bei “auseinandergezogenen Verben“ alle Teile des Verbs markieren

Häufige Fehler

Aufgabe 2 (Abbottsche Methode): ∅ 1,73 bzw. 3,19 von 5

- bei “auseinandergezogenen Verben“ alle Teile des Verbs markieren
  z.B “teilnehmen“ $\Longrightarrow$ “Studenten nehmen an VL teil“

Häufige Fehler

Aufgabe 2 (Abbottsche Methode): ∅ 1,73 bzw. 3,19 von 5

- bei “auseinandergezogenen Verben“ alle Teile des Verbs markieren
  z.B “teilnehmen“ $\Longrightarrow$ “Studenten nehmen an VL teil“
- Worte kommen mehrfach vor $\Longrightarrow$ jedes Mal markieren!

Häufige Fehler

Aufgabe 2 (Abbottsche Methode): ∅ 1,73 bzw. 3,19 von 5

- bei “auseinandergezogenen Verben“ alle Teile des Verbs markieren
  z.B “teilnehmen“ $\Longrightarrow$ “Studenten nehmen an VL teil“
- Worte kommen mehrfach vor $\Longrightarrow$ jedes Mal markieren!
- bei jedem “ist“, “sind“, etc. Vererbung

Häufige Fehler

Aufgabe 2 (Abbottsche Methode): ∅ 1,73 bzw. 3,19 von 5

- bei “auseinandergezogenen Verben“ alle Teile des Verbs markieren
  z.B “teilnehmen“ $\Longrightarrow$ “Studenten nehmen an VL teil“
- Worte kommen mehrfach vor $\Longrightarrow$ jedes Mal markieren!
- bei jedem “ist“, “sind“, etc. Vererbung
- “wissenschaftlicher Mitarbeiter“ = Attribut und Klasse

Häufige Fehler

Aufgabe 2 (Abbottsche Methode): ∅ 1,73 bzw. 3,19 von 5

- bei “auseinandergezogenen Verben“ alle Teile des Verbs markieren
  z.B “teilnehmen“ $\Longrightarrow$ “Studenten nehmen an VL teil“
- Worte kommen mehrfach vor $\Longrightarrow$ jedes Mal markieren!
- bei jedem “ist“, “sind“, etc. Vererbung
- “wissenschaftlicher Mitarbeiter“ = Attribut und Klasse

Aufgabe 3 (iMage-GUI): ∅ (tbd)

- (nächstes Mal)

Häufige Fehler

Aufgabe 4 (Geheimnisprinzip): ∅ 1,92 bzw. 3,07 von 5

Häufige Fehler

Aufgabe 4 (Geheimnisprinzip): ∅ 1,92 bzw. 3,07 von 5

- nicht nur die öffentlichen Konstanten sind problematisch, sondern auch die getter und setter

Häufige Fehler

Aufgabe 4 (Geheimnisprinzip): ∅ 1,92 bzw. 3,07 von 5

- nicht nur die öffentlichen Konstanten sind problematisch, sondern auch die getter und setter
  $\Longrightarrow$ die Entscheidung den Zustand intern als int zu repräsentieren muss versteckt werden

Häufige Fehler

Aufgabe 4 (Geheimnisprinzip): ∅ 1,92 bzw. 3,07 von 5

- nicht nur die öffentlichen Konstanten sind problematisch, sondern auch die getter und setter
  $\Longrightarrow$ die Entscheidung den Zustand intern als int zu repräsentieren muss versteckt werden
  $\Longrightarrow$ nach außen immer boolean benutzen (wohldefiniert!)

Was bisher geschah..

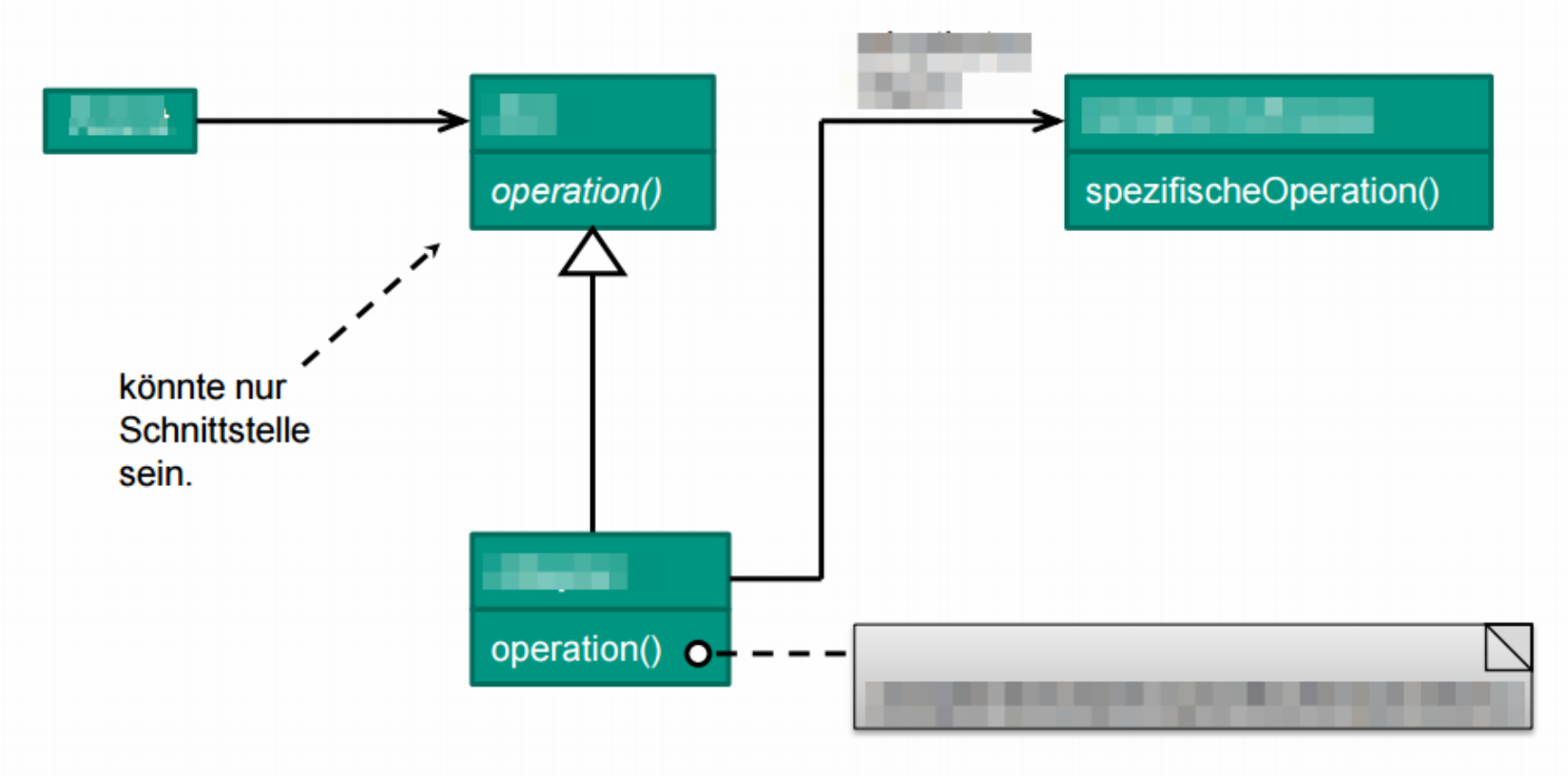
- haben uns erste Entkopplungmuster angeschaut

Was bisher geschah..

- haben uns erste Entkopplungmuster angeschaut
  $\Longrightarrow$ Beobachter, Iterator, Adapter, Stellvertreter, Vermittler

Was bisher geschah..

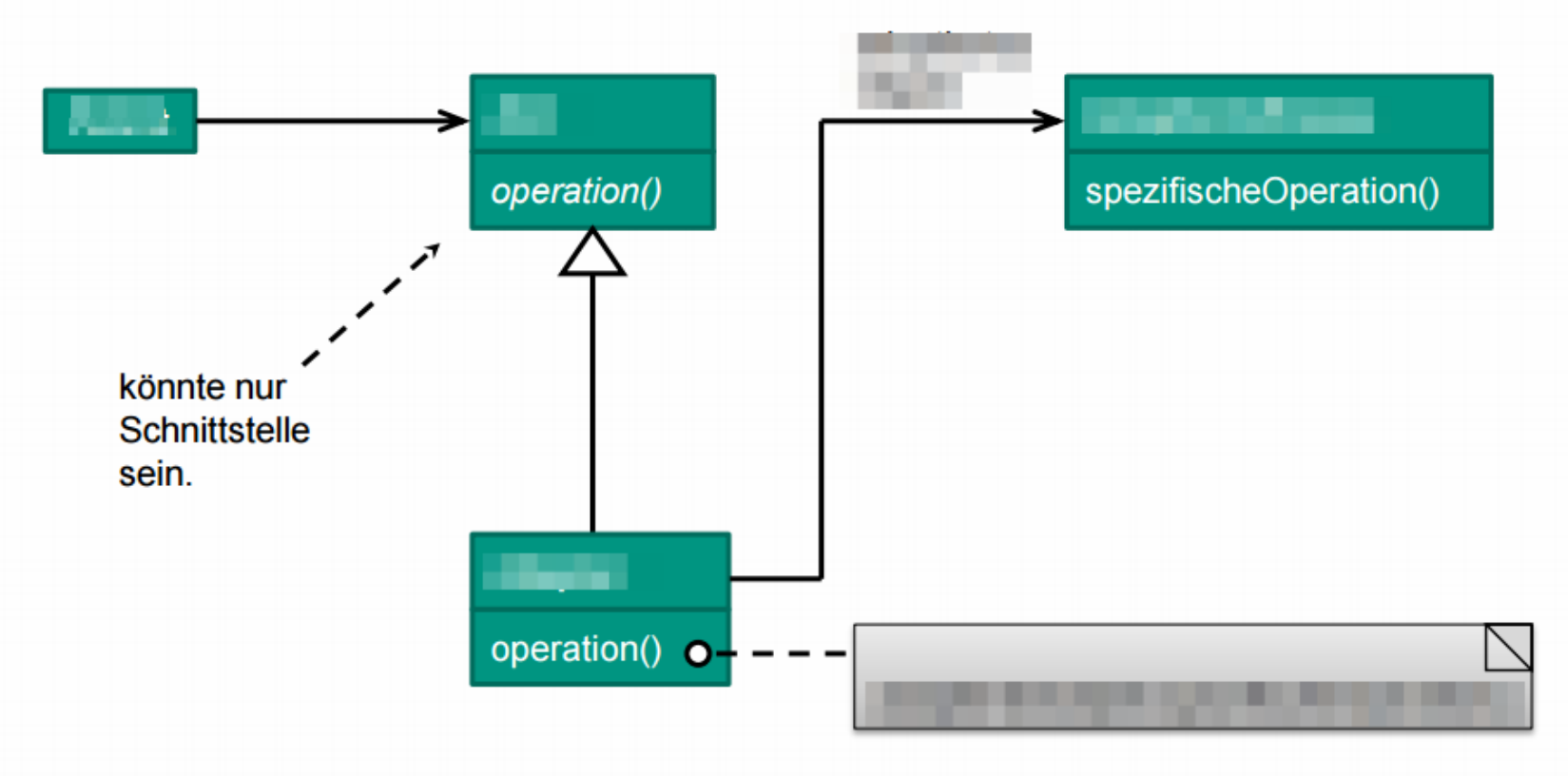
- haben uns erste Entkopplungmuster angeschaut
  $\Longrightarrow$ Beobachter, Iterator, Adapter, Stellvertreter, Vermittler



Welches Entwurfsmuster?

Was bisher geschah..

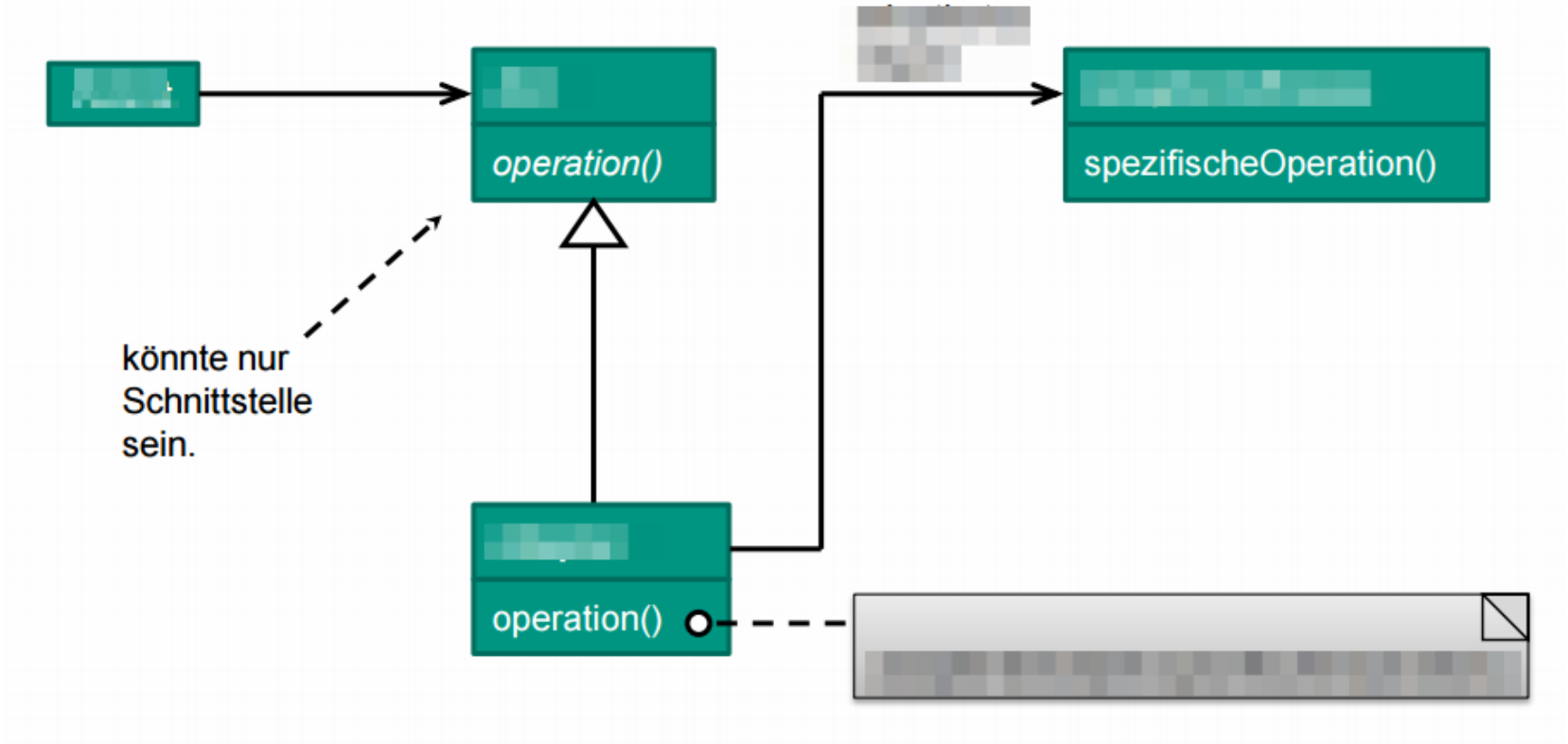
- haben uns erste Entkopplungmuster angeschaut
  $\Longrightarrow$ Beobachter, Iterator, Adapter, Stellvertreter, Vermittler



Welches Entwurfsmuster? (Objekt-)Adapter

Was bisher geschah..

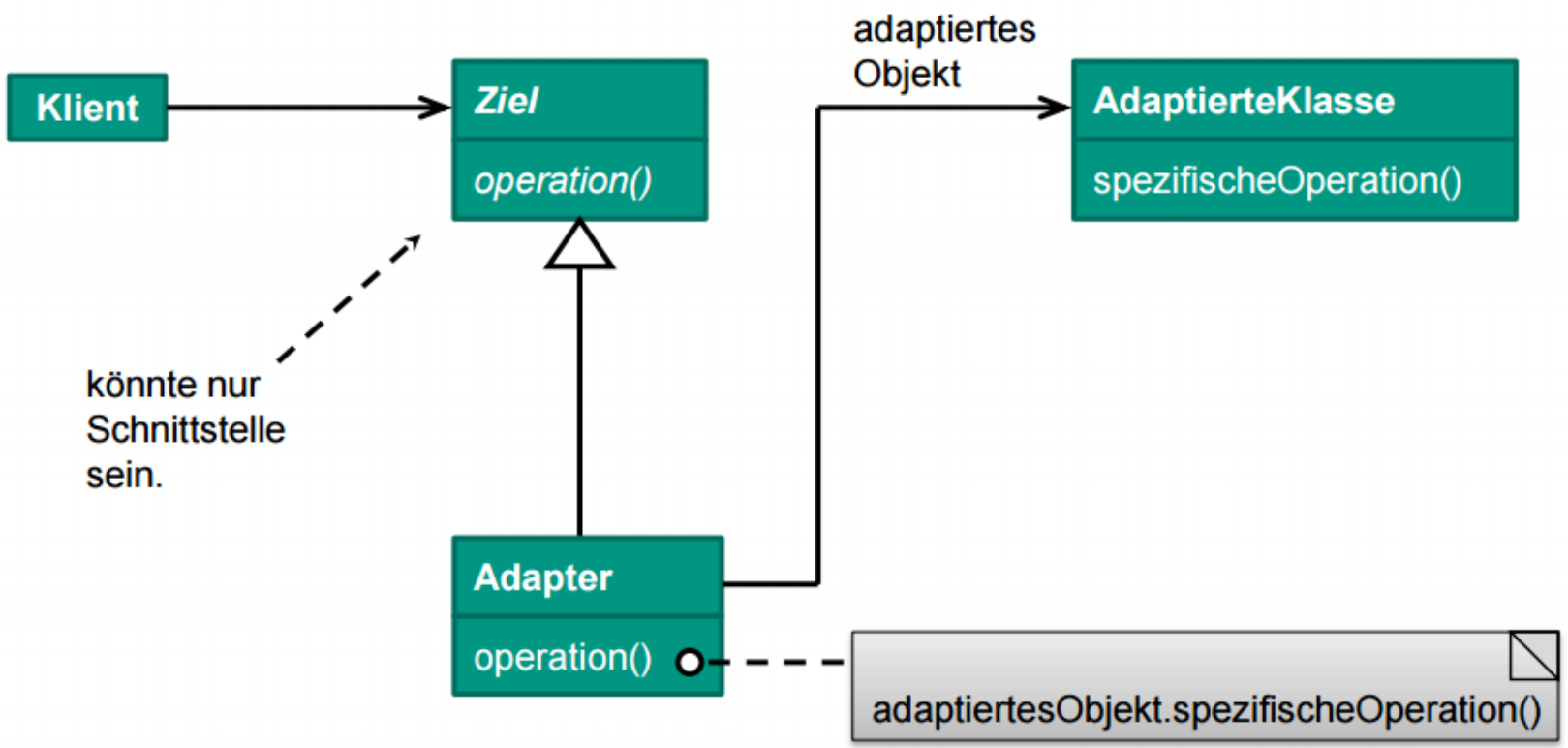
- haben uns erste Entkopplungmuster angeschaut
  $\Longrightarrow$ Beobachter, Iterator, Adapter, Stellvertreter, Vermittler



Welche Klassen?

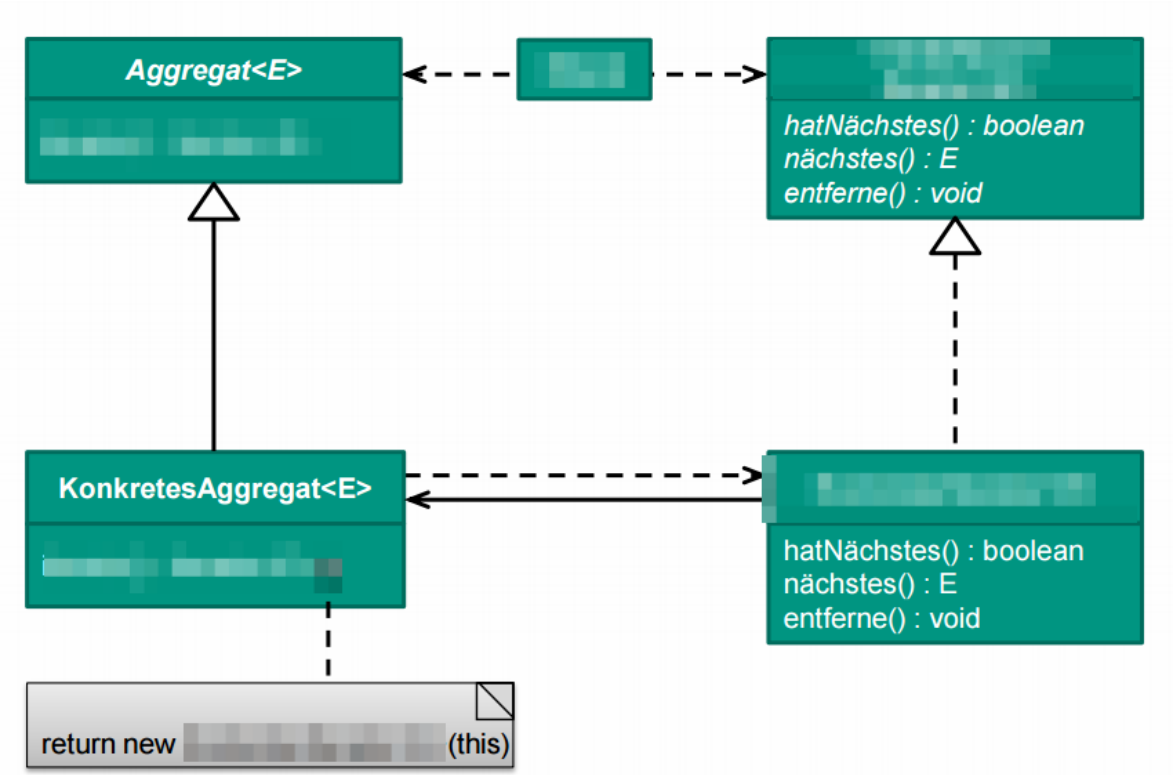
Was bisher geschah..

- haben uns erste Entkopplungmuster angeschaut
  $\Longrightarrow$ Beobachter, Iterator, Adapter, Stellvertreter, Vermittler



Was bisher geschah..

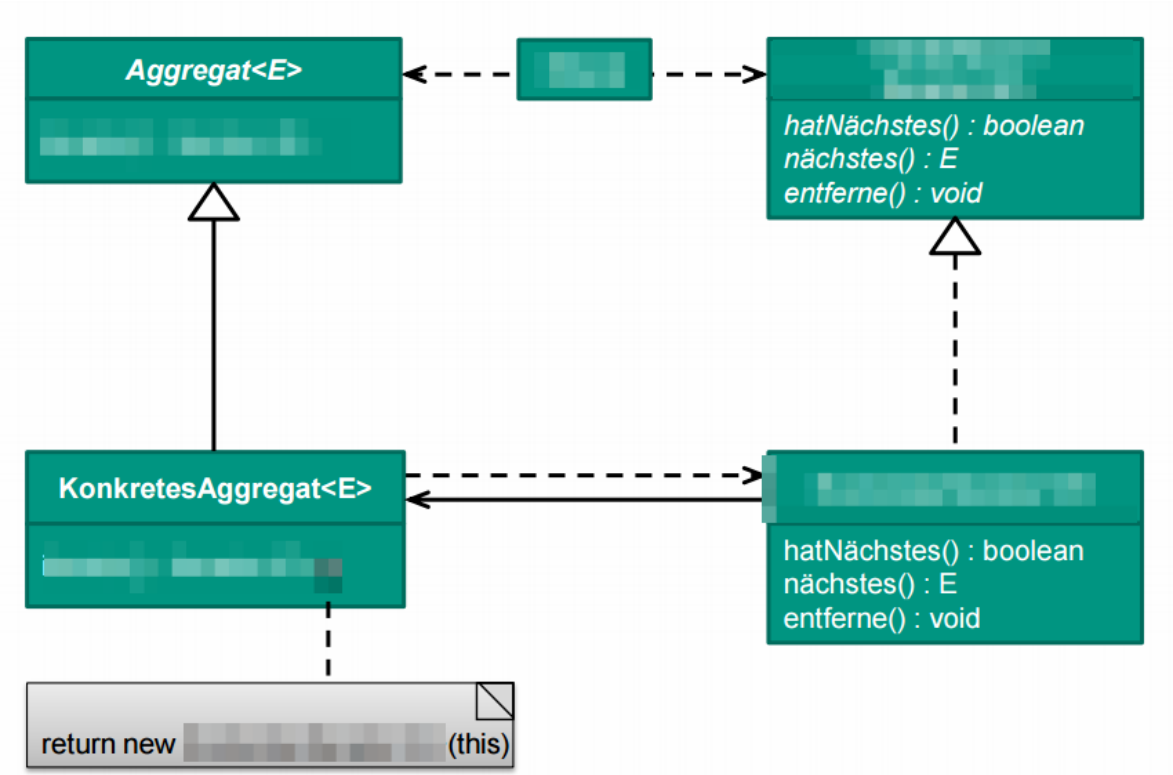
- haben uns erste Entkopplungmuster angeschaut
  $\Longrightarrow$ Beobachter, Iterator, Adapter, Stellvertreter, Vermittler



Welches Entwurfsmuster?

Was bisher geschah..

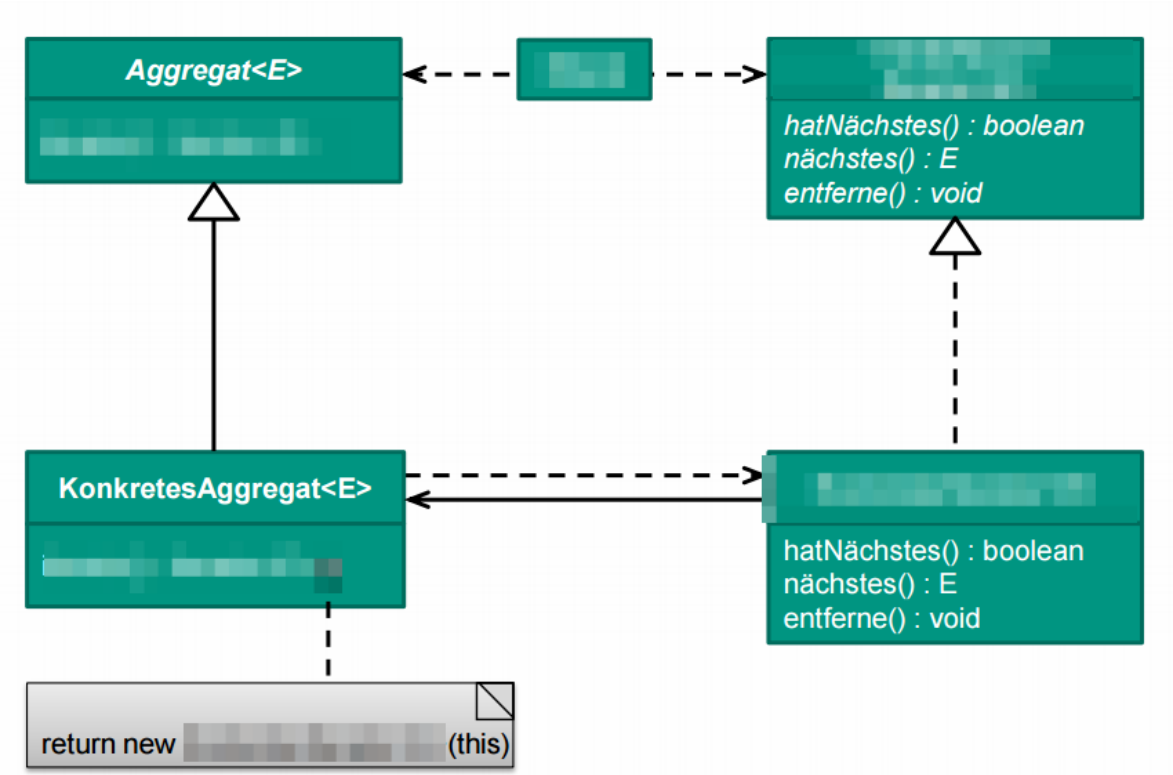
- haben uns erste Entkopplungmuster angeschaut
  $\Longrightarrow$ Beobachter, Iterator, Adapter, Stellvertreter, Vermittler



Welches Entwurfsmuster? Iterator

Was bisher geschah..

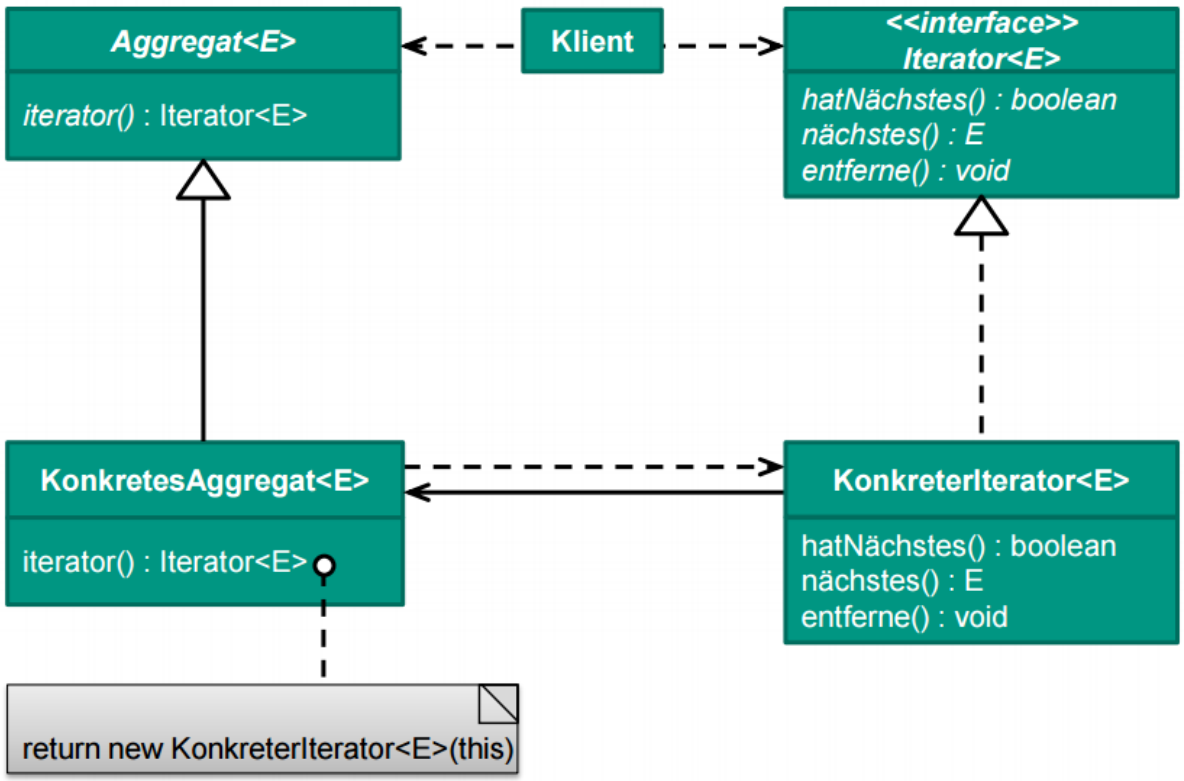
- haben uns erste Entkopplungmuster angeschaut
  $\Longrightarrow$ Beobachter, Iterator, Adapter, Stellvertreter, Vermittler



Welche Klassen und Methoden?

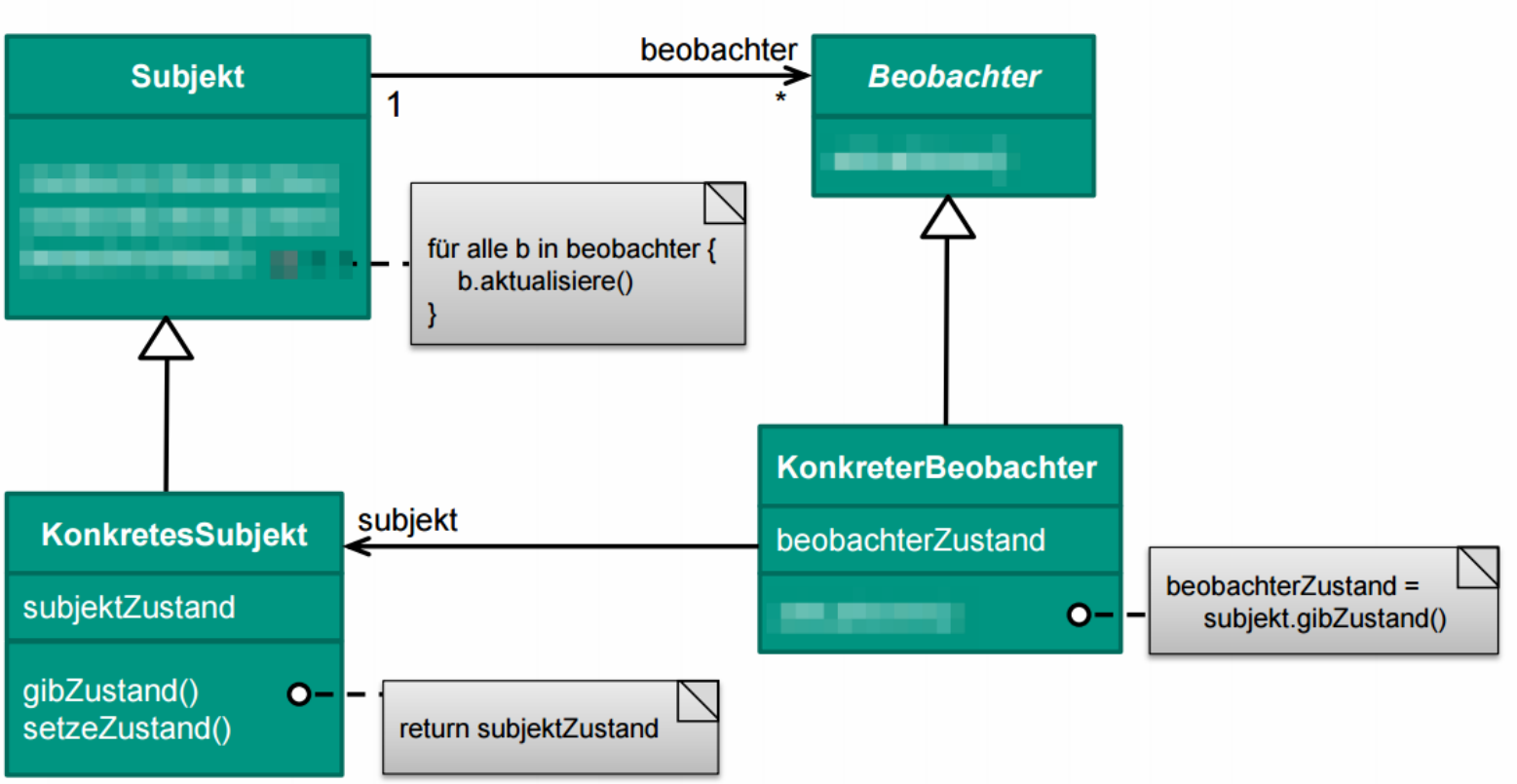
Was bisher geschah..

- haben uns erste Entkopplungmuster angeschaut
  $\Longrightarrow$ Beobachter, Iterator, Adapter, Stellvertreter, Vermittler



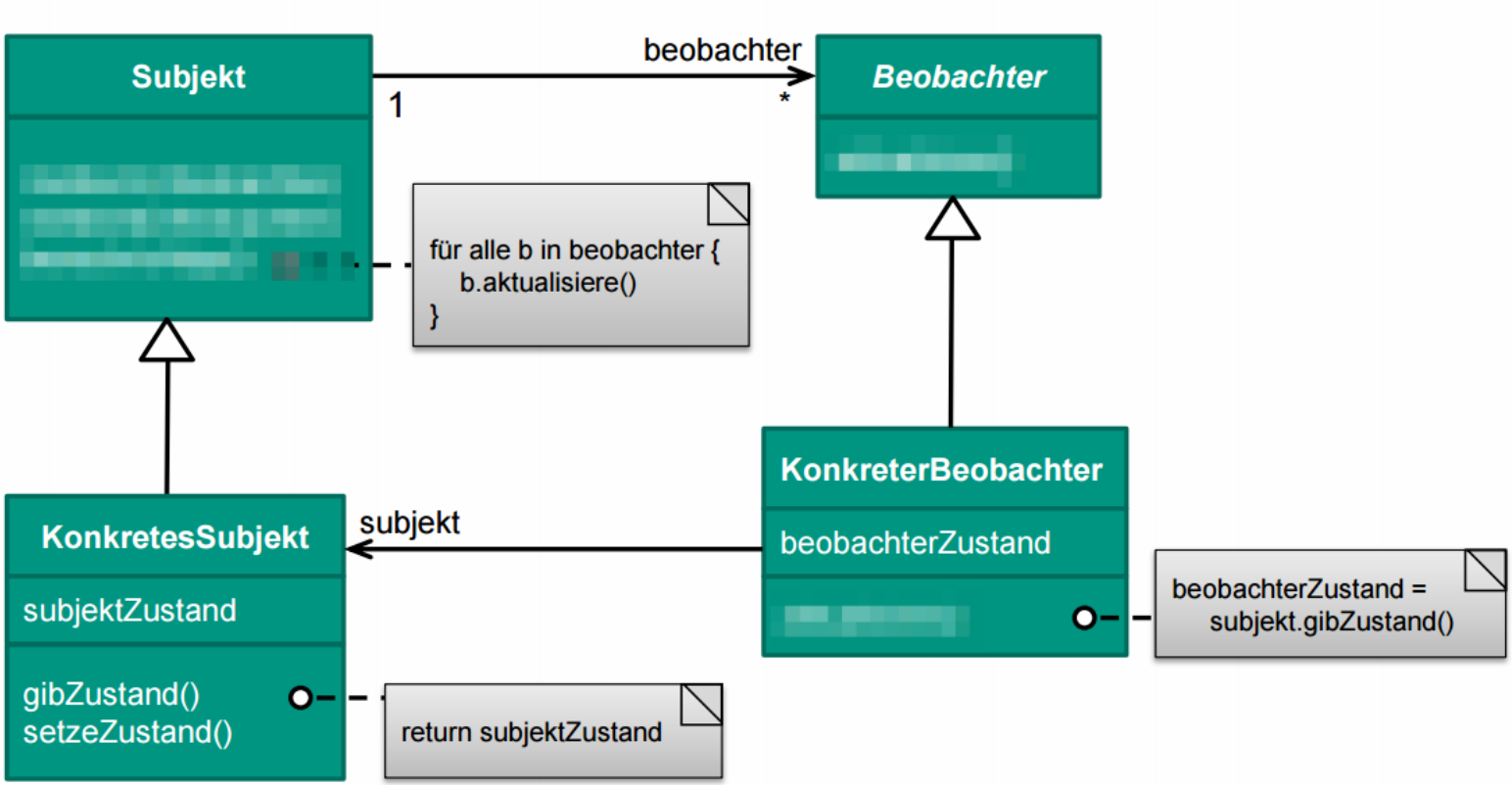
Was bisher geschah..

- haben uns erste Entkopplungmuster angeschaut
  $\Longrightarrow$ Beobachter, Iterator, Adapter, Stellvertreter, Vermittler



Was bisher geschah..

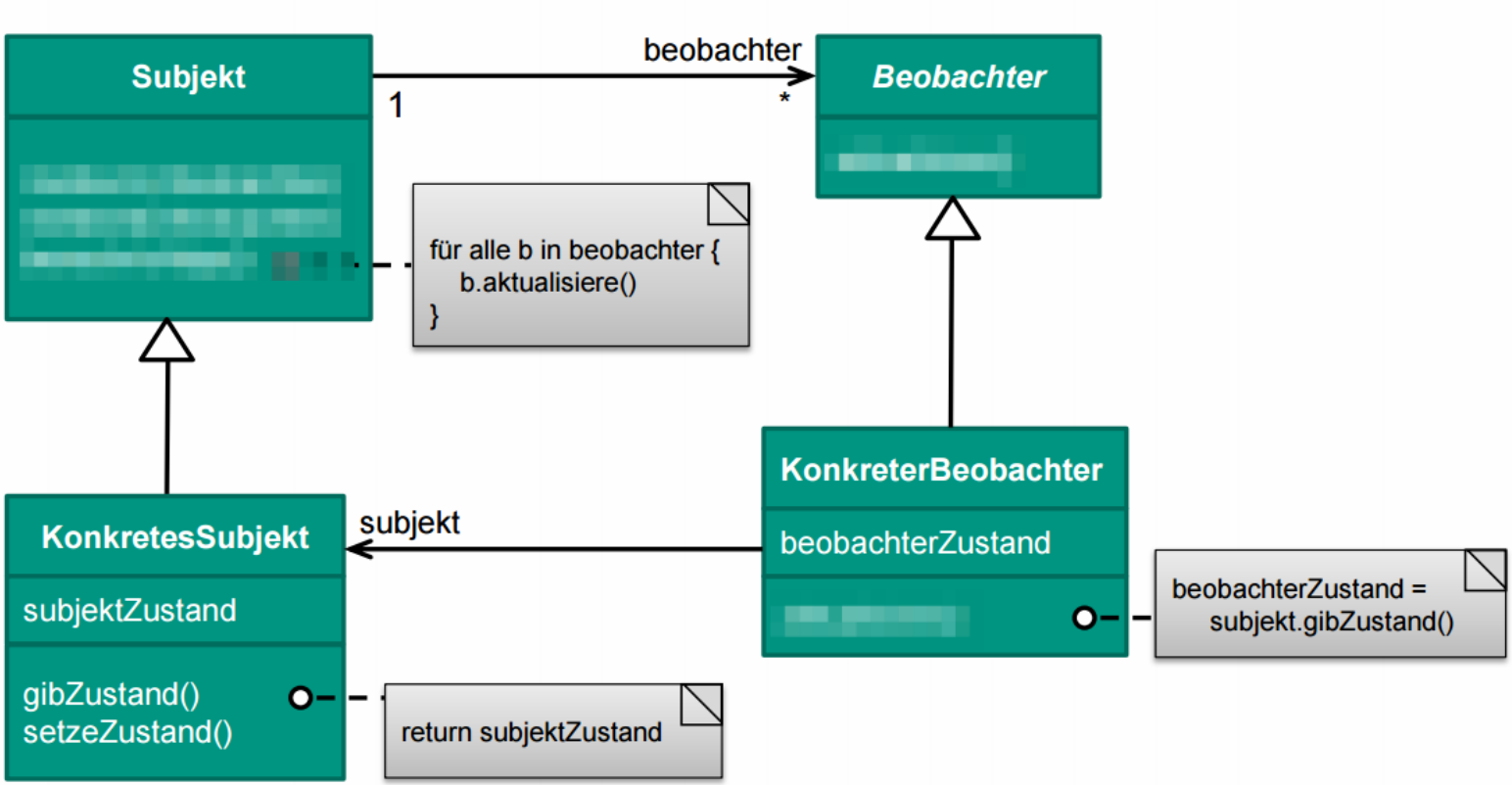
- haben uns erste Entkopplungmuster angeschaut
  $\Longrightarrow$ Beobachter, Iterator, Adapter, Stellvertreter, Vermittler



Ist wohl ein Beobachter :)

Was bisher geschah..

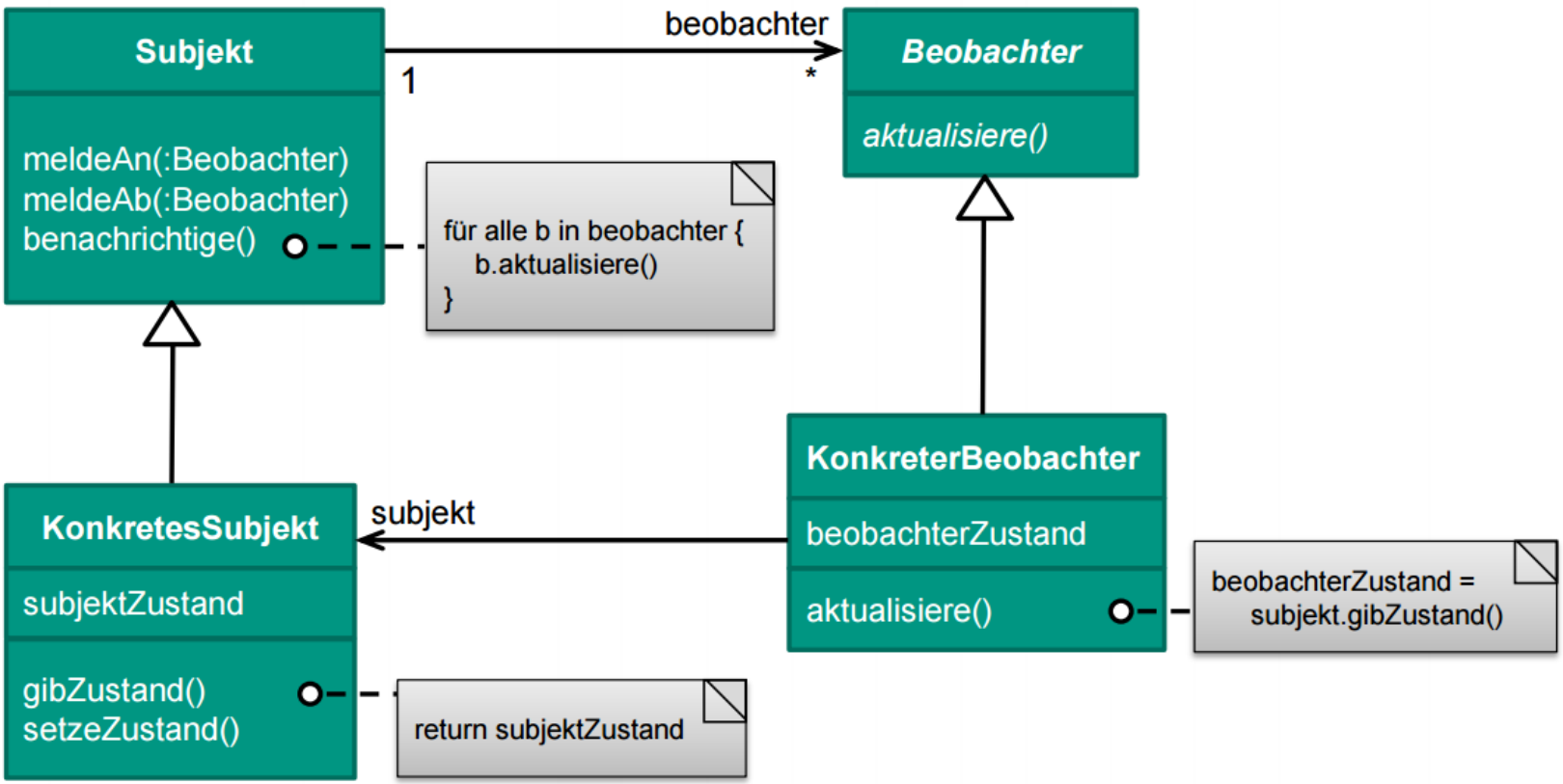
- haben uns erste Entkopplungmuster angeschaut
  $\Longrightarrow$ Beobachter, Iterator, Adapter, Stellvertreter, Vermittler



Ist wohl ein Beobachter :) Methoden?

Was bisher geschah..

- haben uns erste Entkopplungmuster angeschaut
  $\Longrightarrow$ Beobachter, Iterator, Adapter, Stellvertreter, Vermittler



Kategorien der Entwurfsmuster

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

Kategorien der Entwurfsmuster

- Entkopplungs-Muster fertig
- **Varianten-Muster**
  - (Abstrakte Fabrik)
  - (Besucher)
  - **Schablonenmethode**
  - **Fabrikmethode**
  - **Kompositum**
  - Strategie fertig
  - **Dekorierer**
- Zustandshandhabungs-Muster
- Steuerungs-Muster
- Bequemlichkeits-Muster

Varianten-Muster

Übergeordnetes Ziel

- Gemeinsamkeiten herausziehen und an einer Stelle beschreiben

Varianten-Muster

Übergeordnetes Ziel

- Gemeinsamkeiten herausziehen und an einer Stelle beschreiben
  $\Longrightarrow$ keine Wiederholung desselben Codes

Varianten-Muster

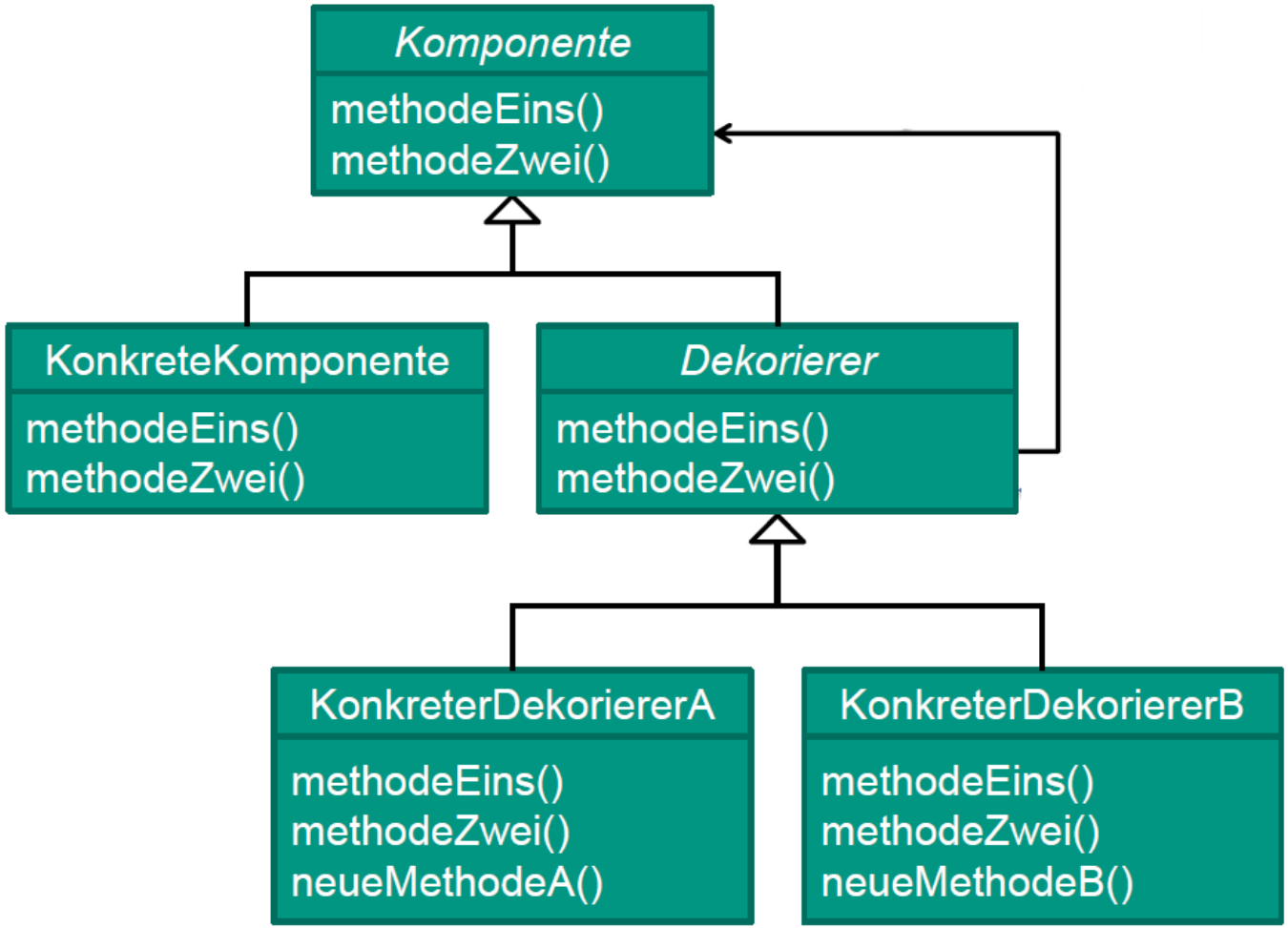
Übergeordnetes Ziel

- Gemeinsamkeiten herausziehen und an einer Stelle beschreiben
  - $\Longrightarrow$ keine Wiederholung desselben Codes
  - $\Longrightarrow$ bessere Wartbarkeit/Erweiterbarkeit

Jetzt: Gruppenarbeit

1. ihr kriegt pro Reihe eine Aufgabe
2. ihr habt Zeit zum Bearbeiten
3. Abgleichung mit Musterlösung
4. ihr stellt den anderen eure Lösung vor

Vorstellung Dekorierer



MuLö Dekorierer

Wo Gemeinsamkeiten?

Die beiden Methoden methodeEins() und methodeZwei().

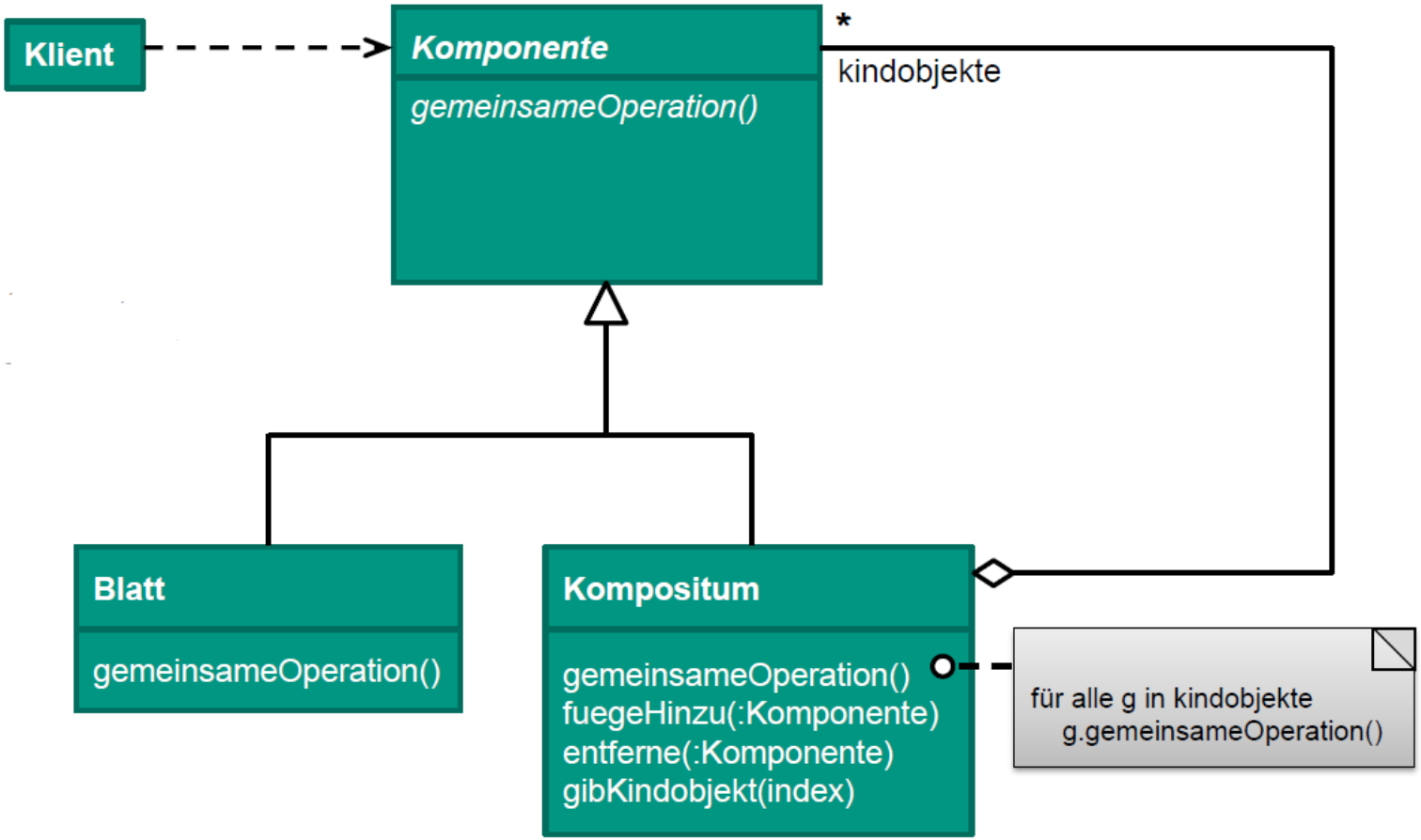
Wo Variation?

In den KonkretenDekorierern bzw. ihren Methoden. Hier: neueMethodeA(), neueMethodeB().

Wozu Instanzvariable?

Weiterleitung von Aufrufen der methodeEins() und methodeZwei() an die KonkreteKompenente.

Vorstellung Kompositum



MuLö Kompositum

Wo Gemeinsamkeiten?

gemeinsameOperation().

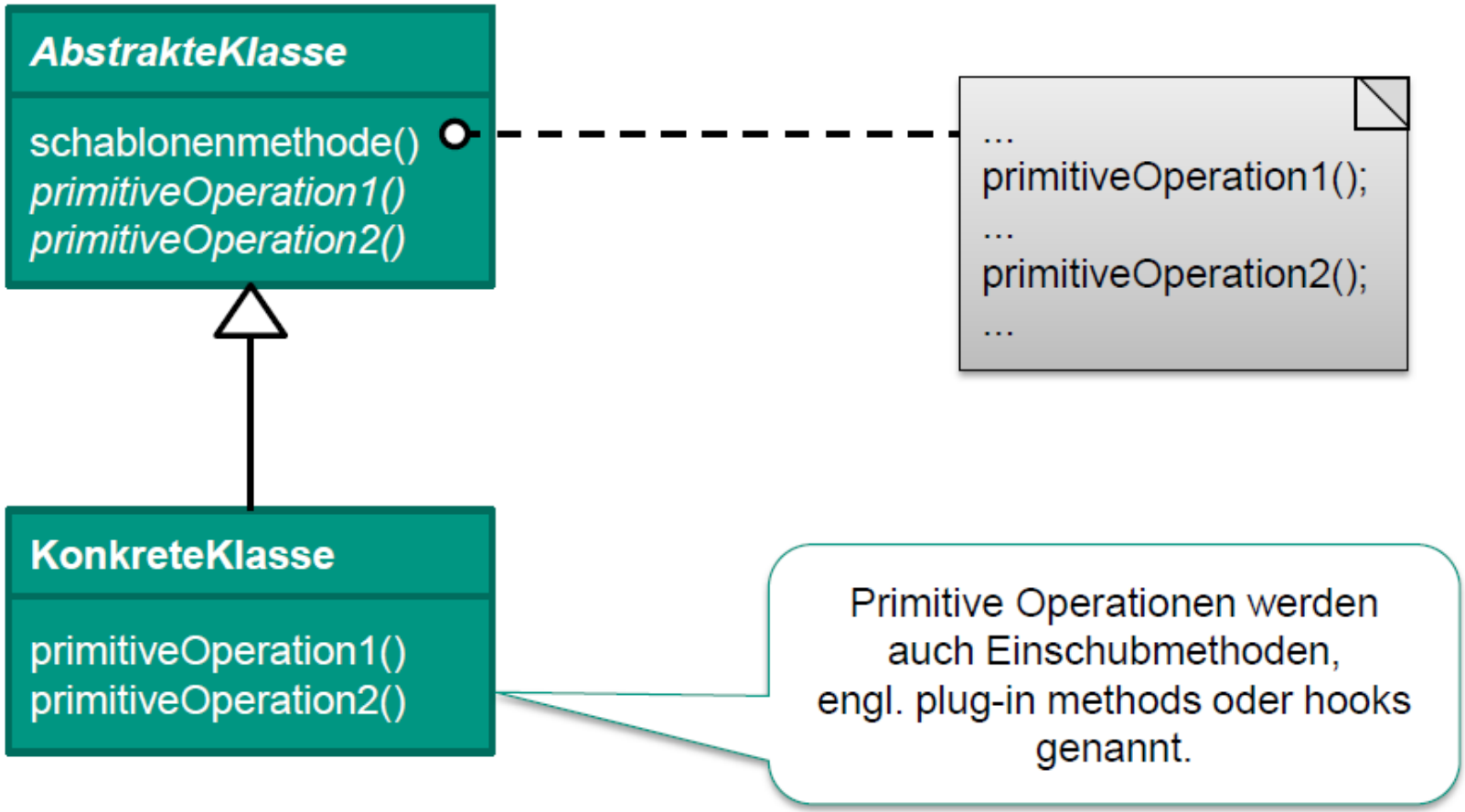
Wo Variation?

In Blatt/Kompositum-Klassen mit verschiedenen zusätzlichen Operationen.

Zusammengesetzt vs. nicht-zusammengesetzt

Kompositum = zusammengesetzt, Blatt = nicht-zusammengesetzt

Vorstellung Schablonenmethode



MuLö Schablonenmethode

Wo Gemeinsamkeiten?

Reihenfolge der Methodenaufrufe in der Schablonenmethode.

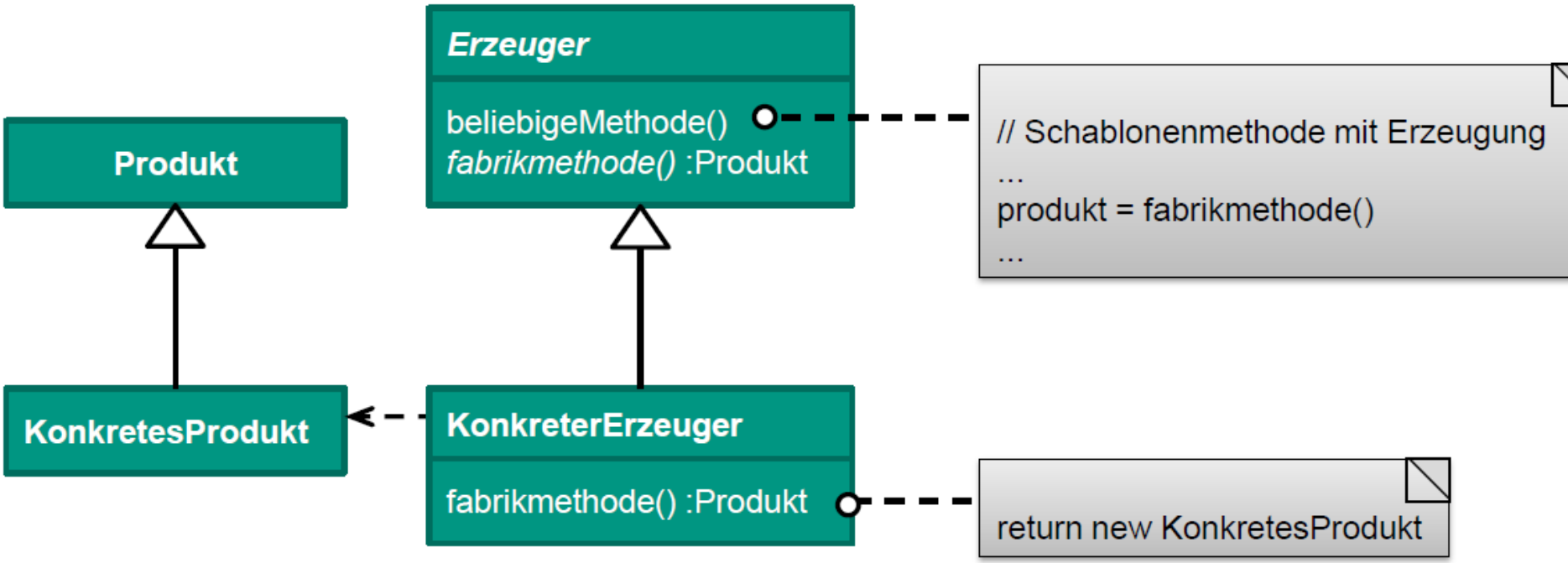
Wo Variation?

In den Einschubmethoden. (hier: primitiveOperation1() und primitiveOpoeration2())

Schablonenmethode vs. Einschubmethode

Einschubmethode ist eine der Methoden, die von der Schablonenmethode aufgerufen wird und deren Implementierung in den Unterklassen stattfindet.

Vorstellung Fabrikmethode



MuLö Fabrikmethode

Wo Gemeinsamkeiten?

Reihenfolge der Methodenaufrufe in der beliebigenMethode().

Wo Variation?

In der Fabrikmethode.

Klasse des Objekts, Oberklasse, Unterklasse

Klasse des Objekts = KonkretesProdukt, Oberklasse = Produkt, Unterklasse = KonkreterErzeuger

Unterschied zu Schablonenmethode?

Fabrikmethode benutzen, wenn ein Objekt erzeugt wird. Fabrikmethode ist Einschubmethode des Musters “Schablonenmethode“.

Wahr/falsch

Fabrikmethode ist eine Einschubmethode, keine Schablonenmethode.

Kategorien der Entwurfsmuster

- Entkopplungs-Muster fertig
- Varianten-Muster fertig
- **Zustandshandhabungs-Muster**
  - (Einzelstück)
  - (Fliegengewicht)
  - **Memento**
  - (Prototyp)
  - (Zustand)
- Steuerungs-Muster
- Bequemlichkeits-Muster

Zustandshandhabungs-Muster

Übergeordnetes Ziel

- den Zustand eines Objektes beschreiben (wer hätt's gedacht? :D)

Zustandshandhabungs-Muster

Übergeordnetes Ziel

- den Zustand eines Objektes beschreiben (wer hätt's gedacht? :D)
- aber unabhängig von dem Zweck des Objekts!

Memento

Problem

- internen Zustand eines Objekts “externalisieren“, um z.B. Zurücksetzen möglich zu machen

Memento

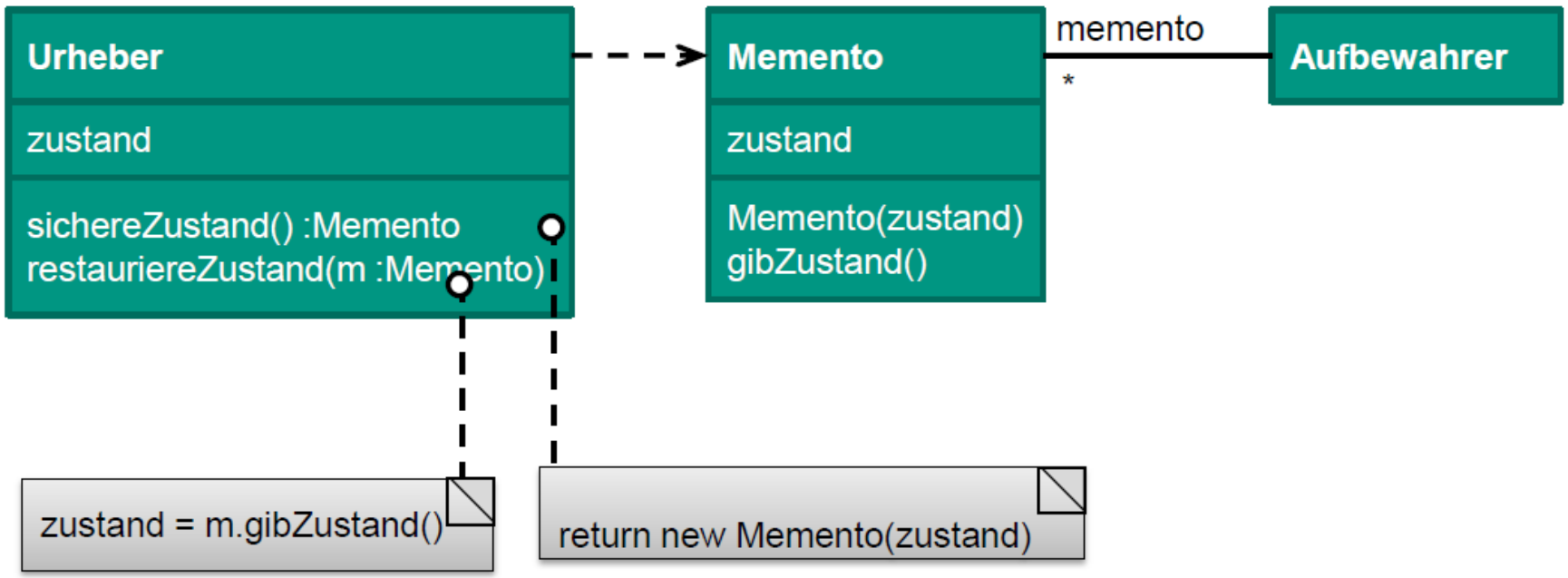
Problem

- internen Zustand eines Objekts “externalisieren“, um z.B. Zurücksetzen möglich zu machen
- ohne Kapselung zu verletzten!

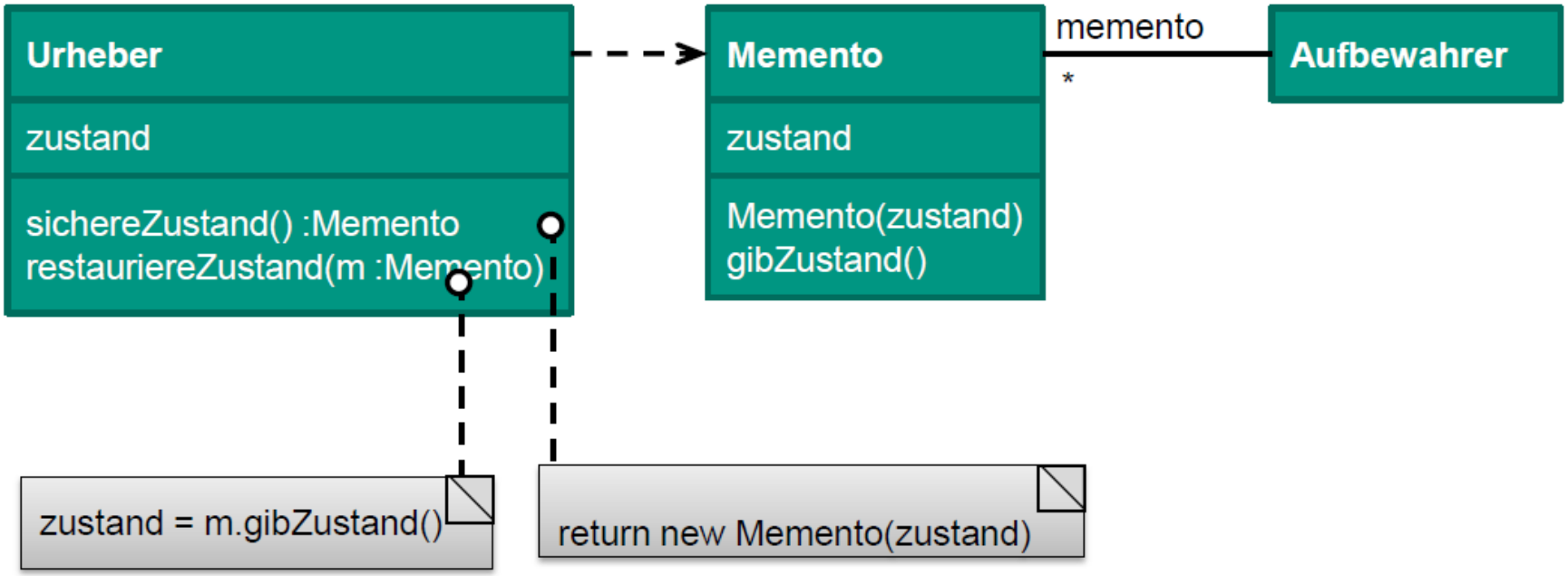
Memento

Problem

- internen Zustand eines Objekts “externalisieren“, um z.B. Zurücksetzen möglich zu machen
- ohne Kapselung zu verletzten!

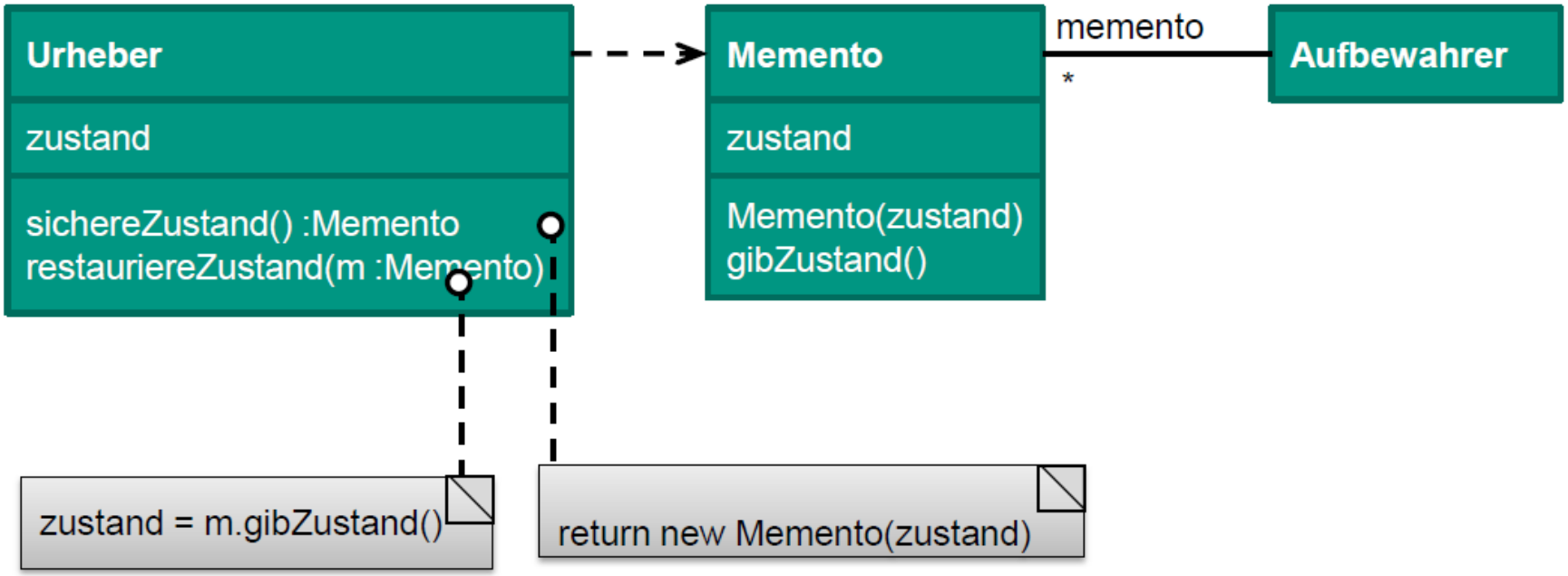


Memento



Problem gelöst?

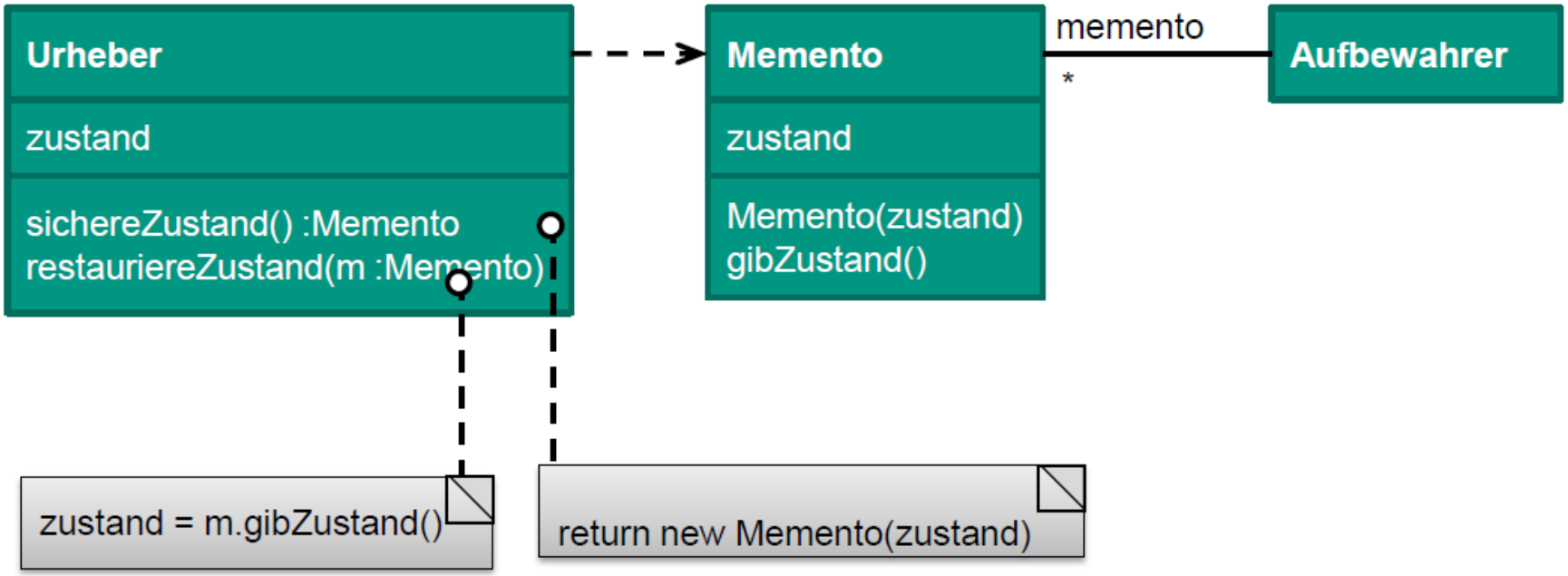
Memento



Problem gelöst?

- Ja

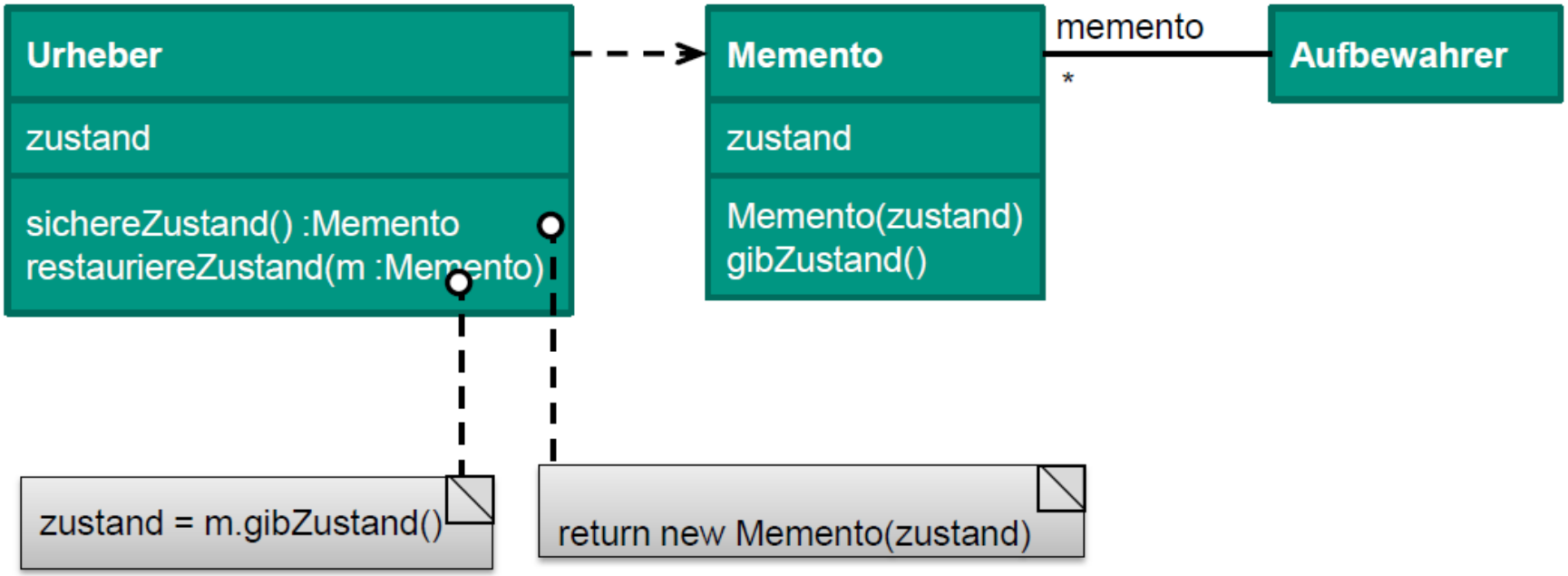
Memento



Problem gelöst?

- Ja
  - Zustand durch Memento externalisiert

Memento



Problem gelöst?

- Ja
  - Zustand durch Memento externalisiert
  - Kapselung nicht verletzt (Nutzer ruft nur sichereZustand() auf und kriegt neuen Memento)

Kategorien der Entwurfsmuster

- Entkopplungs-Muster fertig
- Varianten-Muster fertig
- Zustandshandhabungs-Muster fertig
- **Steuerungs-Muster**
  - **Befehl**
  - (master/worker)
- Bequemlichkeits-Muster

Steuerungs-Muster

Übergeordnetes Ziel

- steuern den Kontrollfluss

Steuerungs-Muster

Übergeordnetes Ziel

- steuern den Kontrollfluss
  $\Longrightarrow$ zur richtigen Zeit richtige Methoden aufrufen

Befehl

Problem

- Parametrisieren von Objekten mit einer auszuführenden Aktion

Befehl

Problem

- Parametrisieren von Objekten mit einer auszuführenden Aktion
- komplexe Operationen aus primitiven Operationen aufbauen

Befehl

Problem

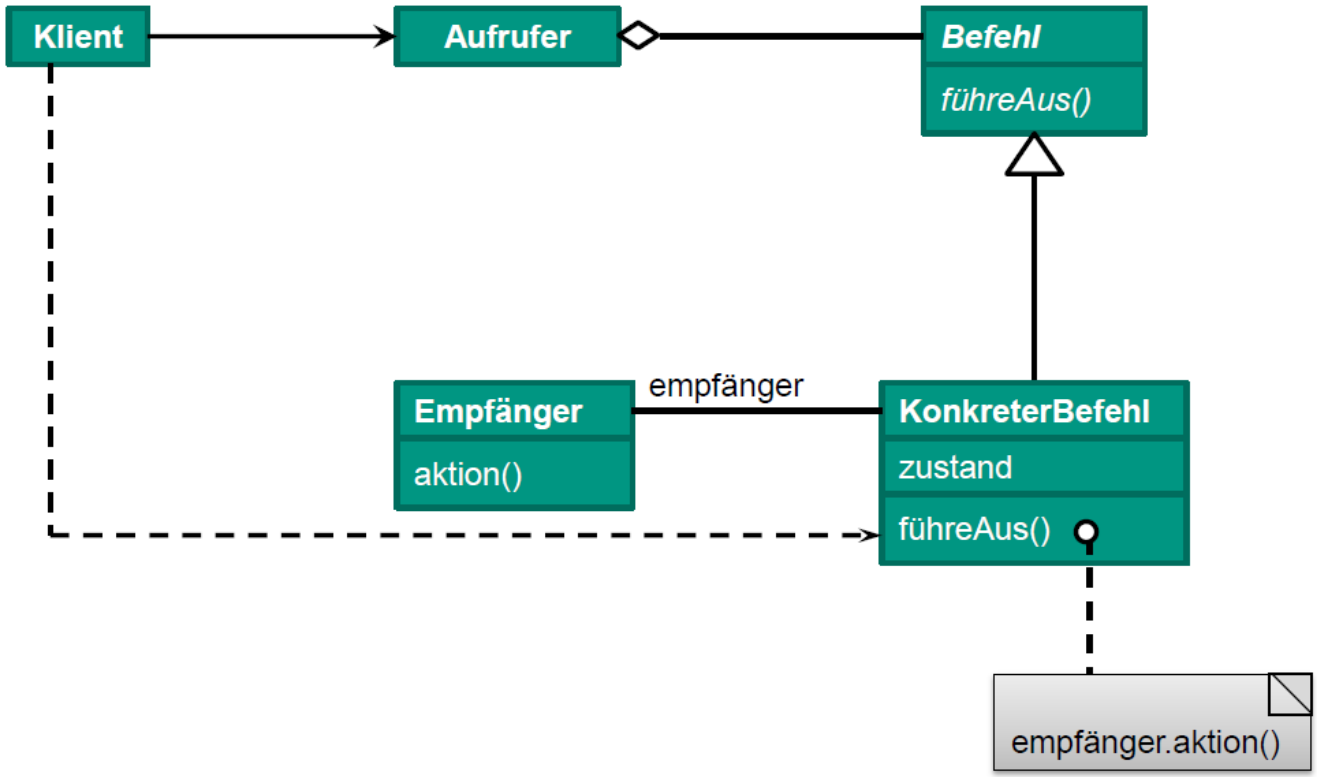
- Parametrisieren von Objekten mit einer auszuführenden Aktion
- komplexe Operationen aus primitiven Operationen aufbauen
  $\Longrightarrow$ Befehl nicht als Methode, sondern als Objekt modellieren

Befehl

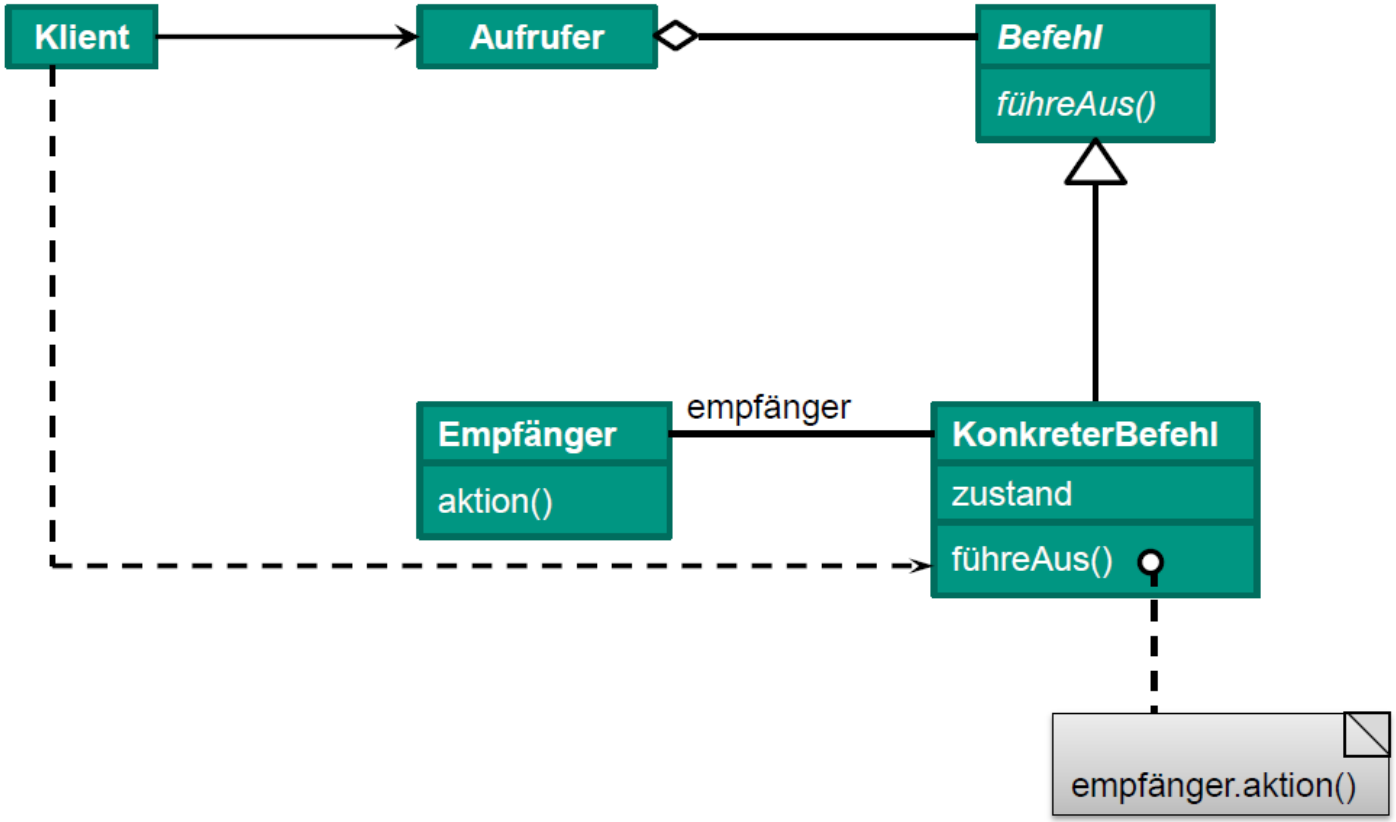
Problem

- Parametrisieren von Objekten mit einer auszuführenden Aktion
- komplexe Operationen aus primitiven Operationen aufbauen

  $\Longrightarrow$ Befehl nicht als Methode, sondern als Objekt modellieren

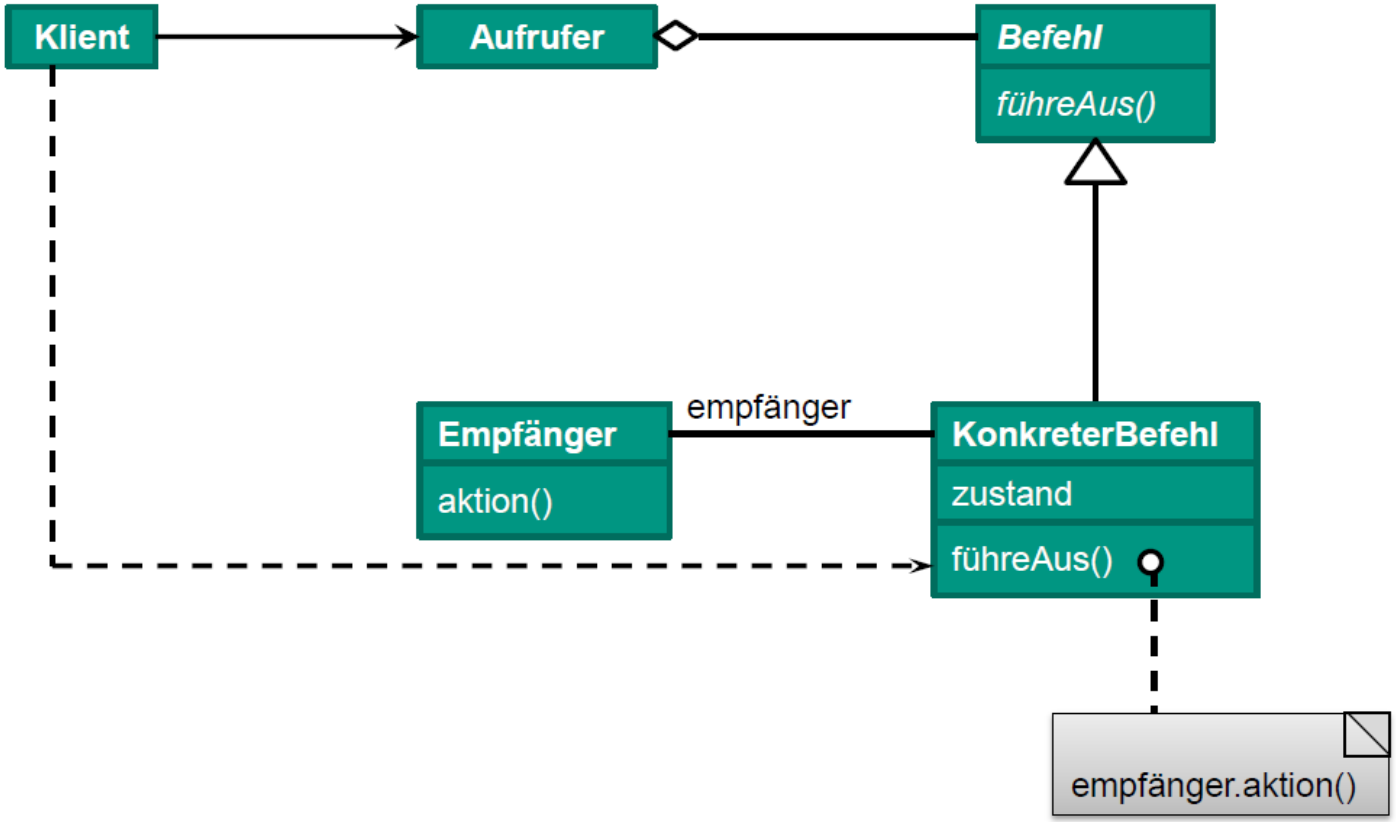


Befehl



Was haben wir erreicht?

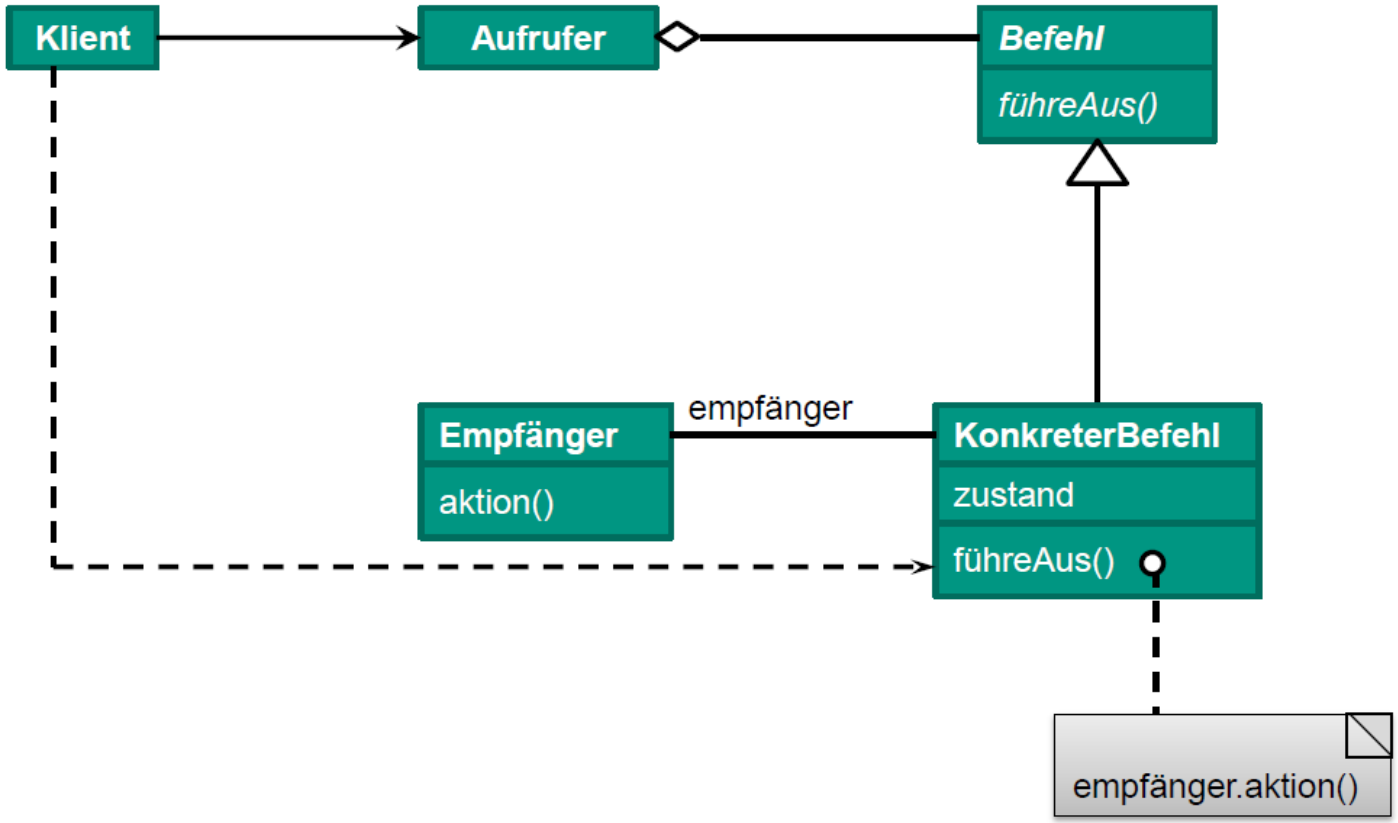
Befehl



Was haben wir erreicht?

- Austauschbarkeit: Befehle unabhängig vom Aufrufer, universell einsetzbar

Befehl



Beispiel!

Quiz (Ankreuzaufgaben aus Klausuren)

Wahr oder falsch?

- Bei dem Entwurfsmuster Befehl kennt der Empfänger den Befehl nicht, jedoch der Befehl den Empfänger.

Quiz (Ankreuzaufgaben aus Klausuren)

Wahr oder falsch?

- Bei dem Entwurfsmuster Befehl kennt der Empfänger den Befehl nicht, jedoch der Befehl den Empfänger. wahr

Quiz (Ankreuzaufgaben aus Klausuren)

Wahr oder falsch?

- Bei dem Entwurfsmuster Befehl kennt der Empfänger den Befehl nicht, jedoch der Befehl den Empfänger. wahr
- Ein Aufbewahrer im Entwurfsmuster Memento kann beliebig viele Mementos verwalten. Für die Restauration im Falle eines Reset ist er allerdings nicht verantwortlich.

Quiz (Ankreuzaufgaben aus Klausuren)

Wahr oder falsch?

- Bei dem Entwurfsmuster Befehl kennt der Empfänger den Befehl nicht, jedoch der Befehl den Empfänger. wahr
- Ein Aufbewahrer im Entwurfsmuster Memento kann beliebig viele Mementos verwalten. Für die Restauration im Falle eines Reset ist er allerdings nicht verantwortlich. wahr

Quiz (Ankreuzaufgaben aus Klausuren)

Wahr oder falsch?

- Bei dem Entwurfsmuster Befehl kennt der Empfänger den Befehl nicht, jedoch der Befehl den Empfänger. wahr
- Ein Aufbewahrer im Entwurfsmuster Memento kann beliebig viele Mementos verwalten. Für die Restauration im Falle eines Reset ist er allerdings nicht verantwortlich. wahr
- Die Fabrikmethode sorgt dafür, dass nur eine einzige Instanz einer Klasse fabriziert wird.

Quiz (Ankreuzaufgaben aus Klausuren)

Wahr oder falsch?

- Bei dem Entwurfsmuster Befehl kennt der Empfänger den Befehl nicht, jedoch der Befehl den Empfänger. wahr
- Ein Aufbewahrer im Entwurfsmuster Memento kann beliebig viele Mementos verwalten. Für die Restauration im Falle eines Reset ist er allerdings nicht verantwortlich. wahr
- Die Fabrikmethode sorgt dafür, dass nur eine einzige Instanz einer Klasse fabriziert wird. falsch

Quiz (Ankreuzaufgaben aus Klausuren)

Wahr oder falsch?

- Bei dem Entwurfsmuster Befehl kennt der Empfänger den Befehl nicht, jedoch der Befehl den Empfänger. wahr
- Ein Aufbewahrer im Entwurfsmuster Memento kann beliebig viele Mementos verwalten. Für die Restauration im Falle eines Reset ist er allerdings nicht verantwortlich. wahr
- Die Fabrikmethode sorgt dafür, dass nur eine einzige Instanz einer Klasse fabriziert wird. falsch
- Eine Schablonenmethode ist immer auch eine Fabrikmethode.

Quiz (Ankreuzaufgaben aus Klausuren)

Wahr oder falsch?

- Bei dem Entwurfsmuster Befehl kennt der Empfänger den Befehl nicht, jedoch der Befehl den Empfänger. wahr
- Ein Aufbewahrer im Entwurfsmuster Memento kann beliebig viele Mementos verwalten. Für die Restauration im Falle eines Reset ist er allerdings nicht verantwortlich. wahr
- Die Fabrikmethode sorgt dafür, dass nur eine einzige Instanz einer Klasse fabriziert wird. falsch
- Eine Schablonenmethode ist immer auch eine Fabrikmethode. falsch

Quiz (Ankreuzaufgaben aus Klausuren)

Wahr oder falsch?

- Bei dem Entwurfsmuster Befehl kennt der Empfänger den Befehl nicht, jedoch der Befehl den Empfänger. wahr
- Ein Aufbewahrer im Entwurfsmuster Memento kann beliebig viele Mementos verwalten. Für die Restauration im Falle eines Reset ist er allerdings nicht verantwortlich. wahr
- Die Fabrikmethode sorgt dafür, dass nur eine einzige Instanz einer Klasse fabriziert wird. falsch
- Eine Schablonenmethode ist immer auch eine Fabrikmethode. falsch
- Eine Komponente kann immer nur mit einem einzigen Dekorierer versehen werden.

Quiz (Ankreuzaufgaben aus Klausuren)

Karlsruher Institut für Technologie

Wahr oder falsch?

- Bei dem Entwurfsmuster Befehl kennt der Empfänger den Befehl nicht, jedoch der Befehl den Empfänger. wahr
- Ein Aufbewahrer im Entwurfsmuster Memento kann beliebig viele Mementos verwalten. Für die Restauration im Falle eines Reset ist er allerdings nicht verantwortlich. wahr
- Die Fabrikmethode sorgt dafür, dass nur eine einzige Instanz einer Klasse fabriziert wird. falsch
- Eine Schablonenmethode ist immer auch eine Fabrikmethode. falsch
- Eine Komponente kann immer nur mit einem einzigen Dekorierer versehen werden. falsch

Für die Klausur

- Entwurfsmuster kommen sehr sehr sehr wahscheinlich dran!

Für die Klausur

- Entwurfsmuster kommen sehr sehr sehr wahscheinlich dran!
- Kategorien helfen beim Lernen

Für die Klausur

- Entwurfsmuster kommen sehr sehr sehr wahscheinlich dran!
- Kategorien helfen beim Lernen
- jedes Entwurfsmuster erfüllt einen bestimmten Zweck
  $\Longrightarrow$ nicht nur die Klassen und Methoden auswendig lernen, sondern das Prinzip verstehen

Für die Klausur

- Entwurfsmuster kommen sehr sehr sehr wahscheinlich dran!
- Kategorien helfen beim Lernen
- jedes Entwurfsmuster erfüllt einen bestimmten Zweck
  $\Longrightarrow$ nicht nur die Klassen und Methoden auswendig lernen, sondern das Prinzip verstehen
- bei Unklarheiten in Head First Design Patterns nachlesen ;)

Feedback - Sagt mir eure Meinung

1. nehmt einen Zettel
2. schreibt (konstruktives!) Feedback darauf
   - am besten $\geqslant$ 4 Stichpunkte
3. legt euren Zettel beim Rausgehen nach vorne

Tipps - 5. Übungsblatt

Aufgabe 1: Manager-Deutsch und Architekturstile

- Architekturstile nochmal anschauen

Tipps - 5. Übungsblatt

Aufgabe 1: Manager-Deutsch und Architekturstile

- Architekturstile nochmal anschauen

Aufgabe 2: Iterator für Plug-Ins

- Iterator-Muster selbst benutzen

Tipps - 5. Übungsblatt

Aufgabe 3: Geometrify mit Entwurfsmustern

- überlegen, welches Entwurfsmuster **warum** Sinn macht

Tipps - 5. Übungsblatt

Aufgabe 3: Geometrify mit Entwurfsmustern

- überlegen, welches Entwurfsmuster **warum** Sinn macht

Aufgabe 4: Geometrify umstrukturieren

- Überlegungen aus Aufgabe 3 umsetzen

Tipps - 5. Übungsblatt

Aufgabe 5: GUI erweitern

- nochmal ServiceLoader $\Longrightarrow$ diesmal mit Primitiven

Denkt dran!

Abgabe

- Deadline am 27.6. um 12:00
- Aufgabe 1, 3 handschriftlich (wirklich handschriftlich!)

Bis dann! (dann := 03.07.18)