Vertraulich (II)

Willkommen

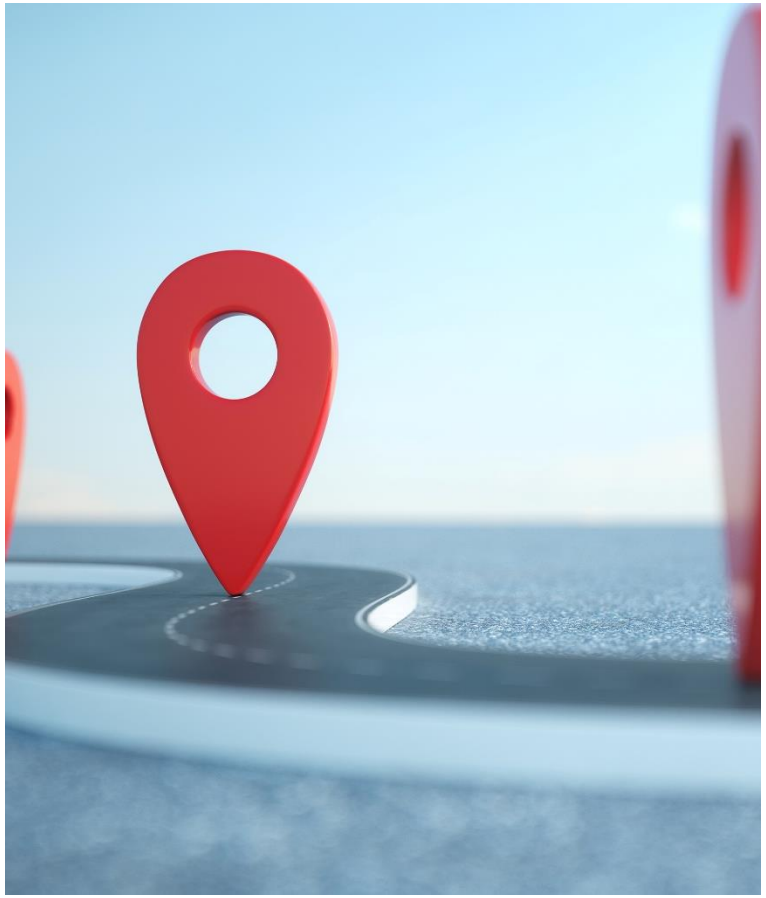
Diagramme und Darstellungen

PHOENIX CONTACT
INSPIRING INNOVATIONS

Erbringen der Leistungen und Auftragsabschluss (§ 4 Absatz 2 Nummer 7)	a) Leistungen nach betrieblichen und vertraglichen Vorgaben dokumentieren b) Leistungserbringung unter Berücksichtigung der orga-

Agenda

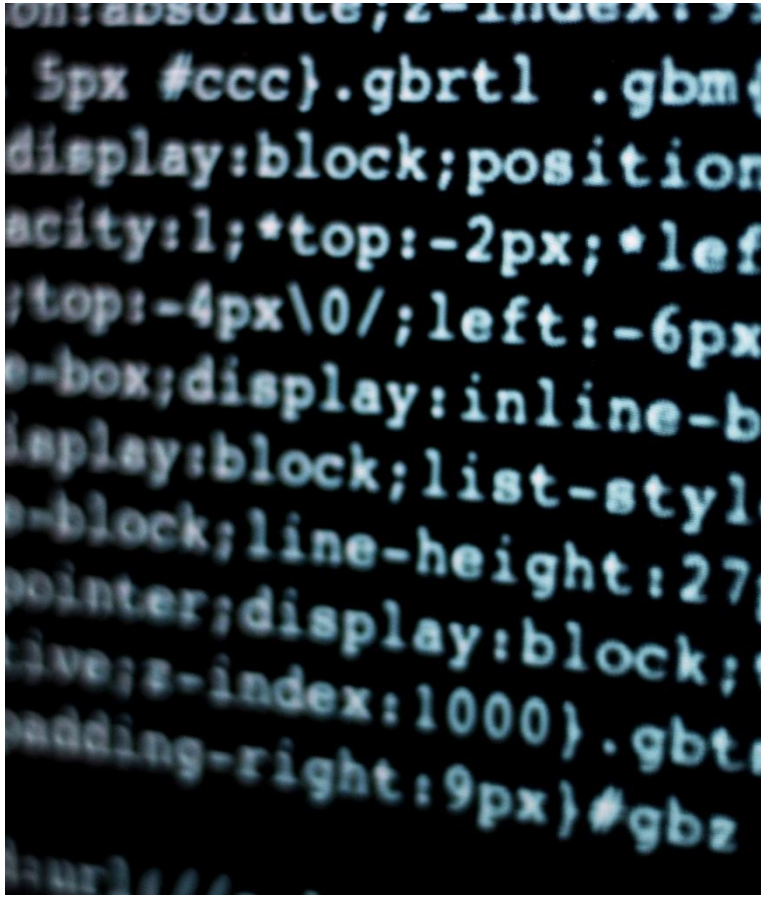
- Pseudocode
- Programmablaufplan (PAP)
- Struktogramme
- UML-Diagramme
  - Klassendiagramme
  - Sequenzdiagramme
  - Aktivitätsdiagramme
  - Zustandsdiagramme



Pseudocode

Was ist das?

- „Programmcode, der nicht zur maschinellen Interpretation, sondern lediglich zur Veranschaulichung eines Paradigmas oder Algorithmus dient“
- Unabhängige Beschreibung eines Programmablaufs
- Formaler und weniger missverständlich als Beschreibung in natürlicher Sprache
- Oftmals Zwischenschritt von Programmablaufplan/Struktogramm zu Quellcode



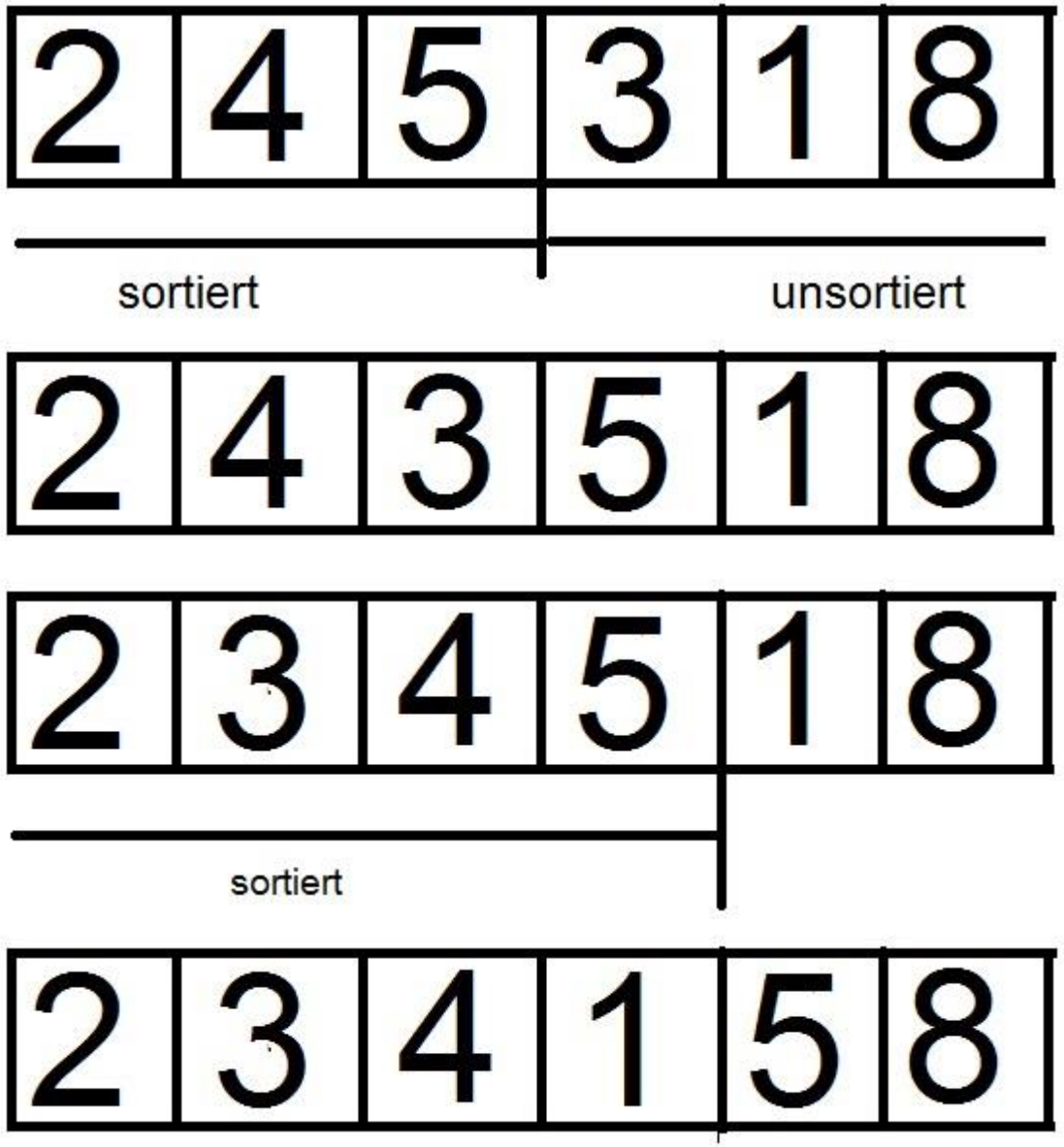
Häufige Schlüsselwörter

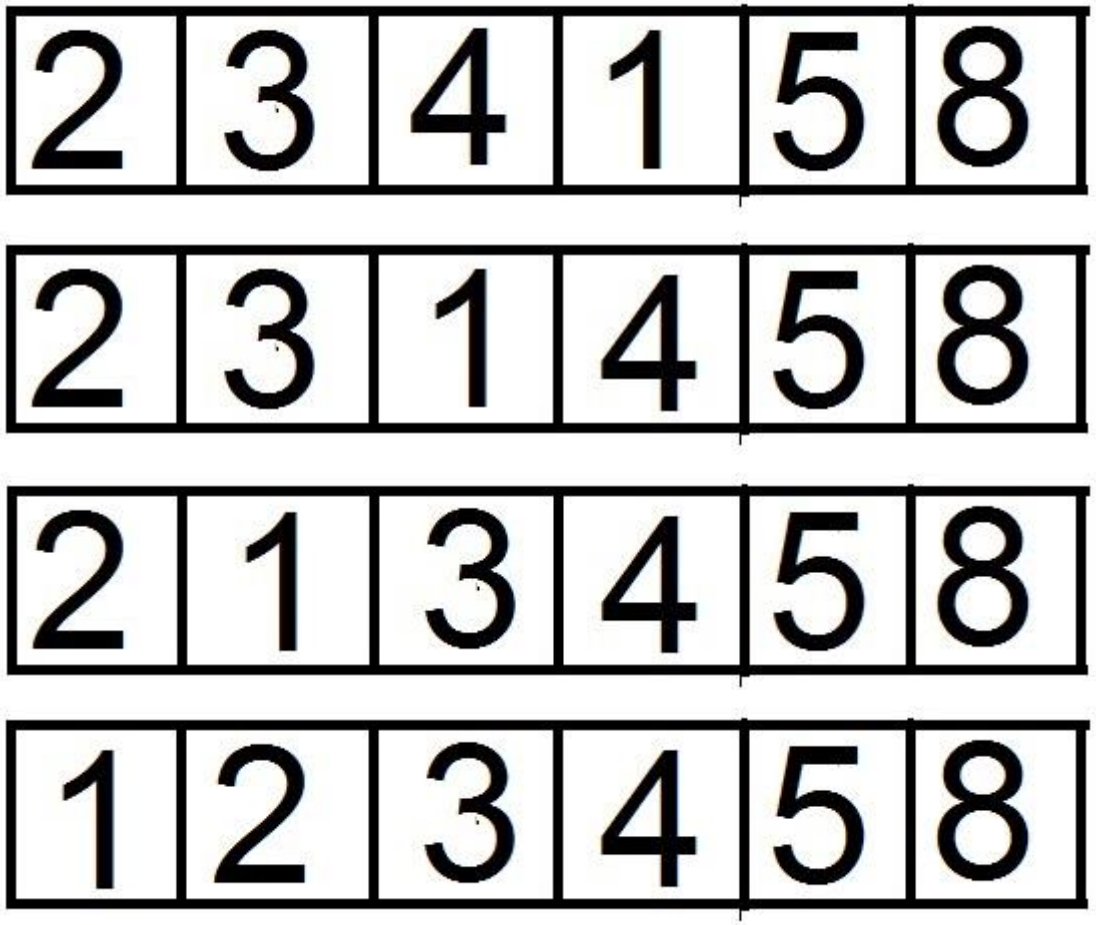
- **Module**
  - `program` *`Programmname`* `... end` *`Programmname`*
  - `klasse` *`Klassenname`* `{ ... }`
- **Fallunterscheidungen**
  - `if ... then ... else ... end if/exit`
  - `wenn ... dann ... sonst ... wenn_ende`
  - `falls ... dann ... falls_nicht ... falls_ende`
- **Schleifen**
  - `wiederhole ... solange/bis ... wiederhole_ende`
  - `while ... do ...`
  - `repeat ... until ...`
  - `for ... to ... step` *`Schrittweite`* `... next`
- **Kommentare**
  - *`// kommentar`*
  - *`# kommentar`*
  - *`/* kommentar */`*
- **Definition von Funktionen**
  - `function() ... begin ... end`
  - `funktion() ... start ... ende`
- **Zusicherungen**
  - `assert`
  - `jetzt gilt`

Pseudocode

Beispiel

```
1  INSERTION-SORT(A)
2     for j=2 to A.länge
3        schlüssel=A[j]
4        //füge A[j] in den sortierten Beginn des Arrays A[1..j-1] ein
5        i=j-1
6        while i>0 und A[i]>schlüssel
7           A[i+1]=A[i]
8           i=i-1
9        A[i+1]=schlüssel
```





Stellt den folgenden Algorithmus in Pseudocode dar.

Quelle: Bubblesort – Wikipedia

```
bubbleSort(Array A)
  for (n=A.size; n>1; --n){
    for (i=0; i<n-1; ++i){
      if (A[i] > A[i+1]){
        A.swap(i, i+1)
      } // Ende if
    } // Ende innere for-Schleife
  } // Ende äußere for-Schleife
```

Quelle: Bubblesort – Wikipedia

Übung

- Im Pseudocode soll die Funktionsweise (vereinfacht) einer Kaffeemaschine beschrieben werden.
- Die Kaffeemaschine hat einen Chipkartenleser für die Zahlung der Getränke.
- Zur Auswahl stehen eine Tasse Kaffee oder heißes Wasser für einen Tee.
- Die Tasse Kaffee kostet 0,50€ und heißes Wasser für eine Tasse 0,10€.
- Die Maschine wechselt am Ende wieder in den Initialzustand.

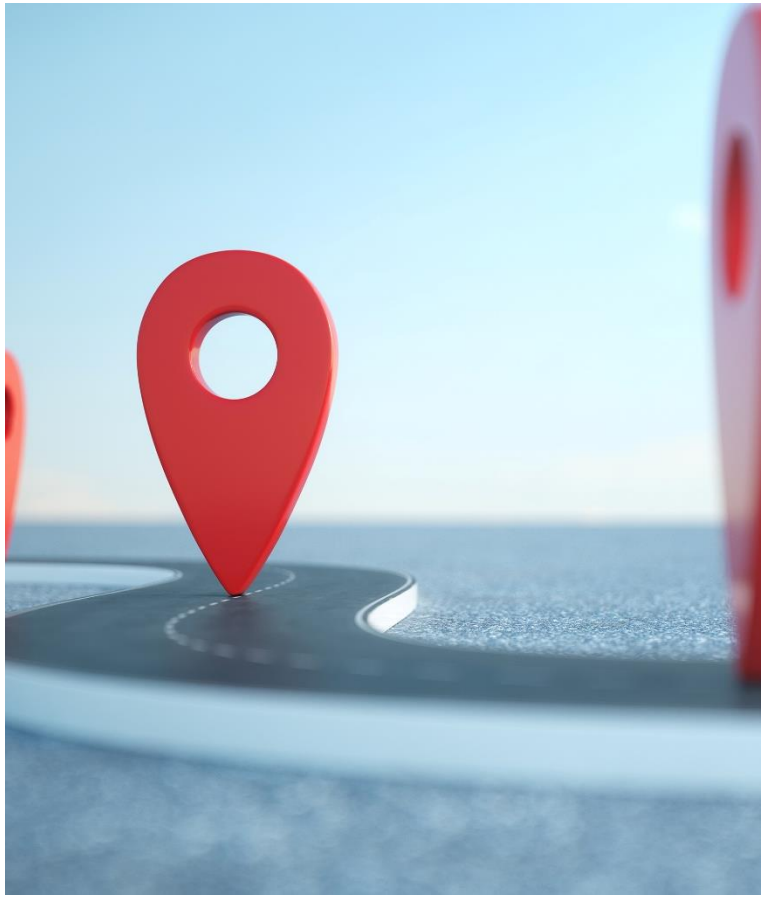
30 Minuten

```
program Kaffeemaschine
        # Deklaration und Initialisierung
        preis_kaffee = 0.5
        preis_wasser = 0.1

        while TRUE do
                if (bestellung_kaffee) {
                        # Kunde hat Kaffee ausgewählt
                        if (geldkarte.guthaben >= 0.5) {
                                geldkarte.guthaben = geldkarte.guthaben - 0.5
                                bereitezu_kaffee()
                        } else {
                                sende_nachricht("Nicht genug Guthaben")
                        } end_if
                } else if (bestellung_wasser) {
                        # Kunde hat Wasser ausgewählt
                        if (geldkarte.guthaben >= 0.1) {
                                geldkarte.guthaben = geldkarte.guthaben - 0.1
                                bereitezu_wasser()
                        } else {
                                sende_nachricht("Nicht genug Guthaben")
                        } end_if
                } end_if
        end_while
end Kaffeemaschine
```

Agenda

- Pseudocode
- Programmablaufplan (PAP)
- Struktogramme
- UML-Diagramme
  - Klassendiagramme
  - Sequenzdiagramme
  - Aktivitätsdiagramme
  - Zustandsdiagramme



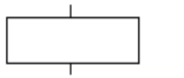
Programmablaufplan

Was ist das?

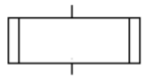
- „Ein Programmablaufplan (PAP) ist ein Ablaufdiagramm für ein Computerprogramm, das auch als Flussdiagramm (engl. flowchart) oder Programmstrukturplan bezeichnet wird. Es ist eine grafische Darstellung zur Umsetzung eines Algorithmus in einem Programm und beschreibt die Folge von Operationen zur Lösung einer Aufgabe.“
- Symbole nach DIN 66001 genormt

Programmablaufplan

Symbole (nach DIN 66 001)

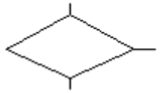


Operation/Verarbeitung allgemein (process)



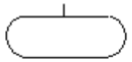
Unterprogrammaufruf (predefined process); Hinweis auf Dokumentation an anderer Stelle in Form von eindeutiger Innenbeschriftung

Ein-/Ausgabe; auch: Daten, allgemein (data)

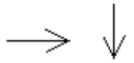


Verzweigung (decision)

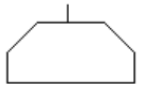
Übergang, Verbindungsstelle (connector)



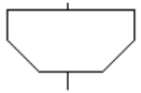
Grenzstelle (terminator)



Ablauflinien (line)

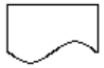


Schleifenbegrenzung für zählergesteuerte Wiederholungen (loop limit)
Anfang



Ende

Maschinell zu verarbeitende Daten (data to be processed by machine)

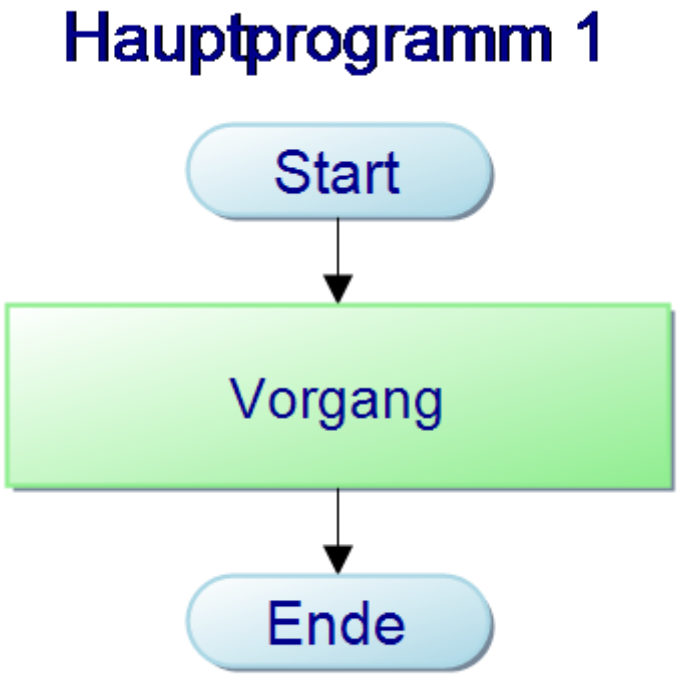


Daten auf Schriftstück (data on document)

Daten auf Speicher auch mit direktem Zugriff (data on direct access storage)

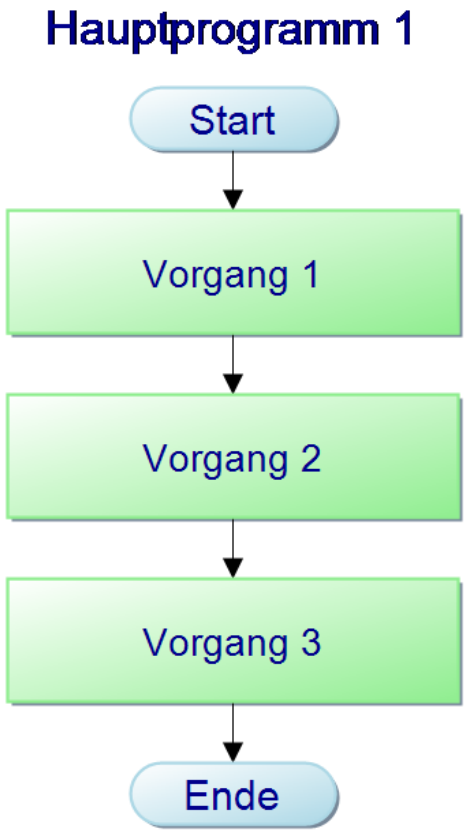
Programmablaufplan

Verarbeitung



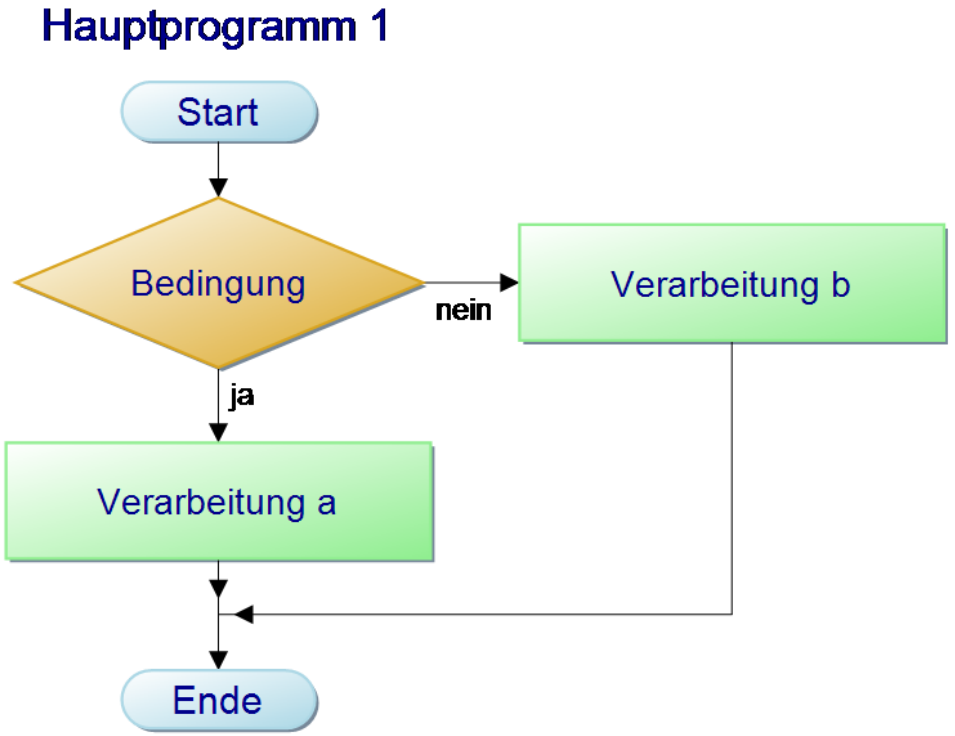
Programmablaufplan

Reihenfolge (Sequenz)



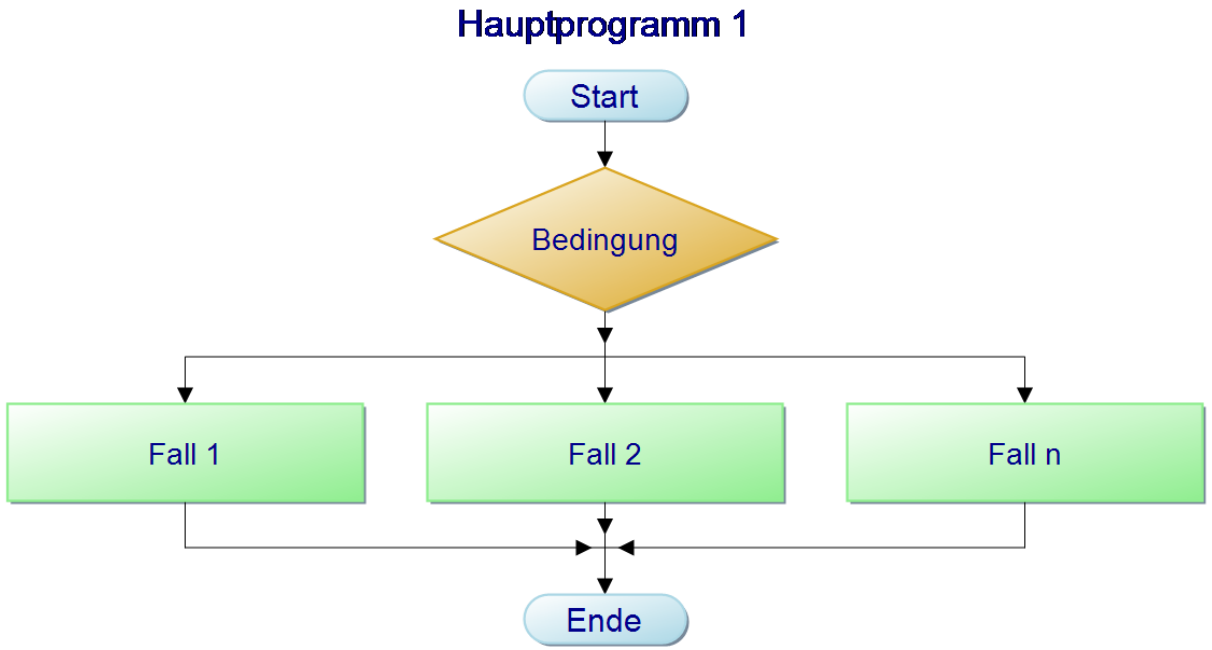
Programmablaufplan

Bedingte Verzweigung



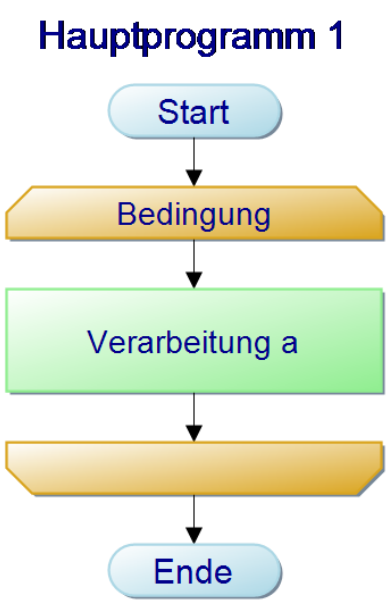
Programmablaufplan

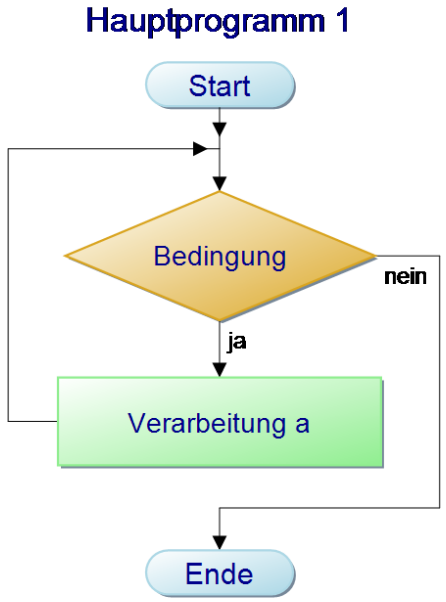
Fallabfrage, Fallunterscheidung



Programmablaufplan

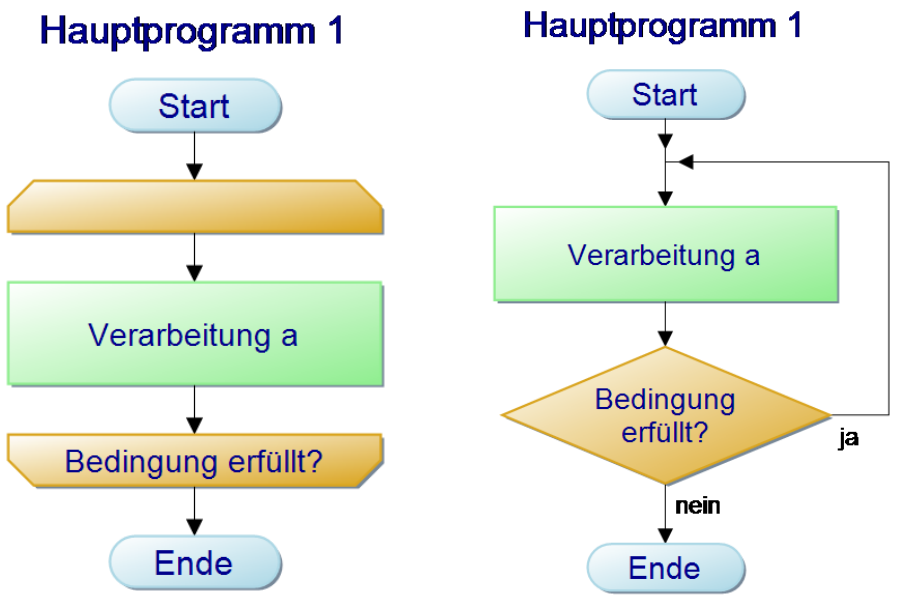
Wiederholung (kopfgesteuerte Schleife)





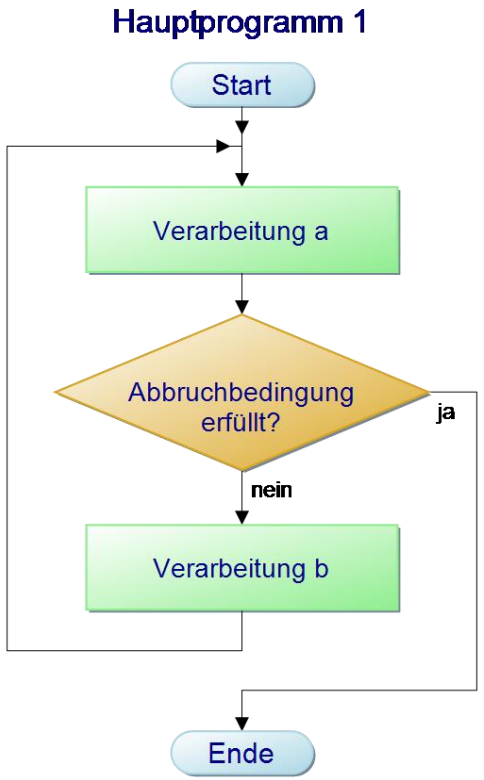
Programmablaufplan

Wiederholung (fußgesteuerte Schleife)



Programmablaufplan

Schleife mit Unterbrechung



Programmablaufplan

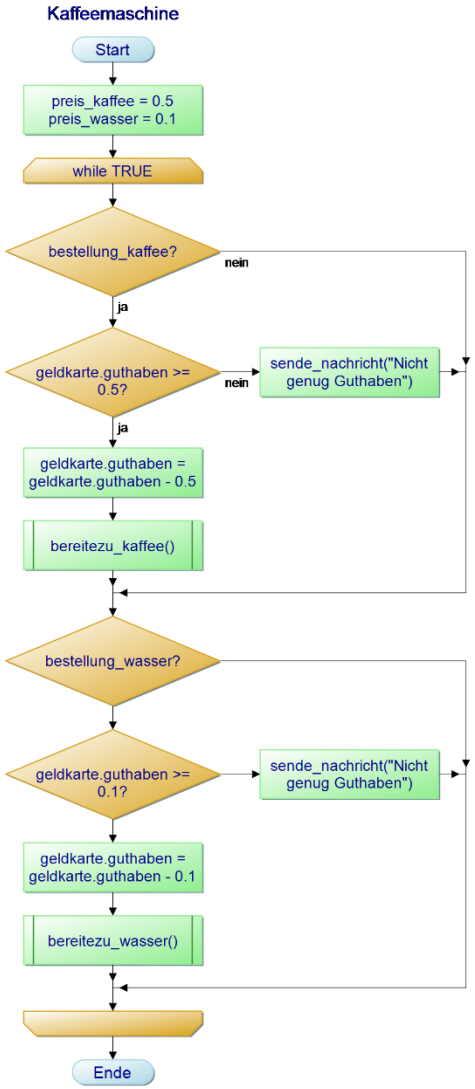
Software

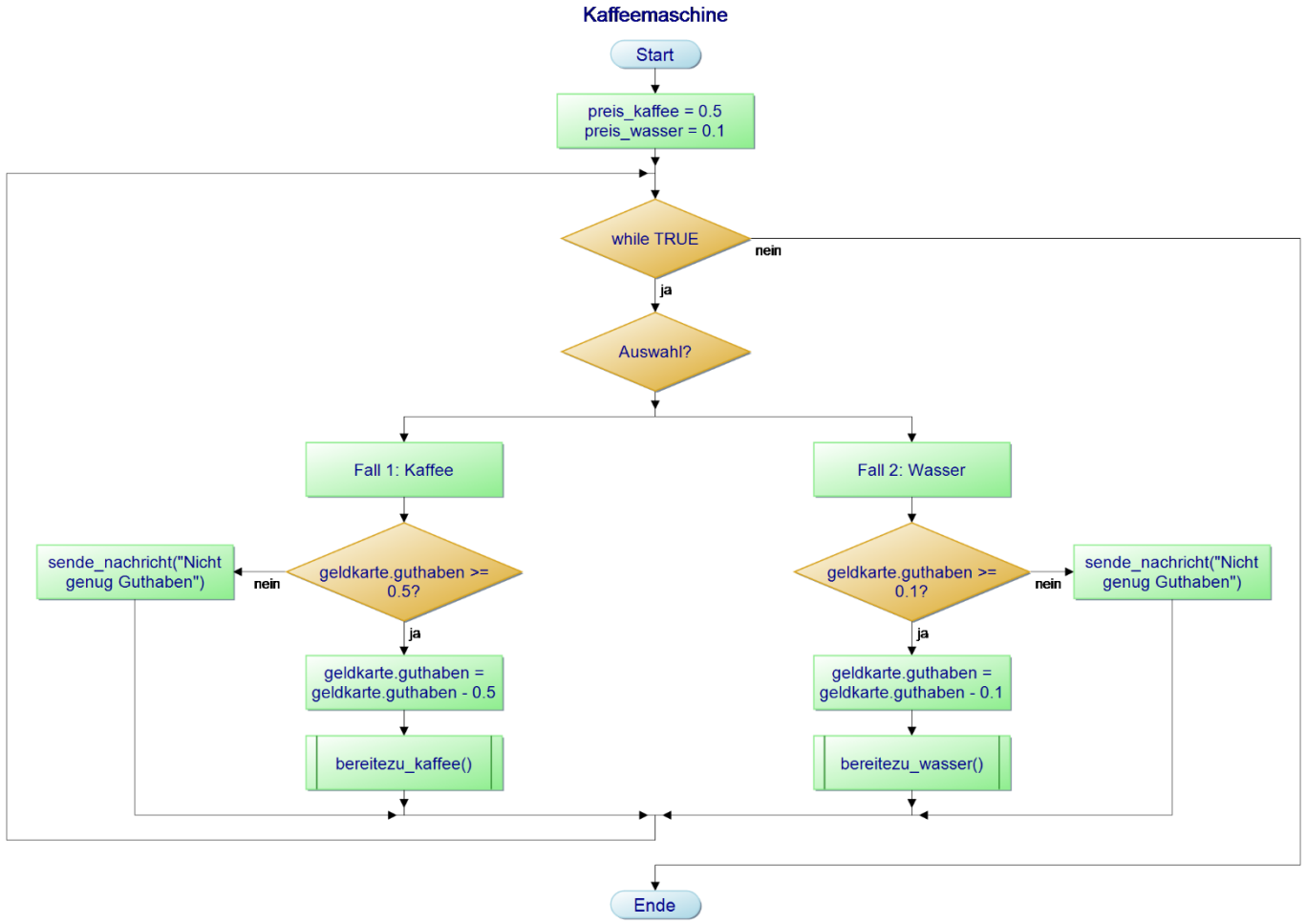
PapDesigner

Übung

- Die Funktionsweise (vereinfacht) der Kaffeemaschine soll nun als Programmablaufplan dargestellt werden.
- Die Kaffeemaschine hat einen Chipkartenleser für die Zahlung der Getränke.
- Zur Auswahl stehen eine Tasse Kaffee oder heißes Wasser für einen Tee.
- Die Tasse Kaffee kostet 0,50€ und heißes Wasser für eine Tasse 0,10€.
- Die Maschine wechselt am Ende wieder in den Initialzustand.

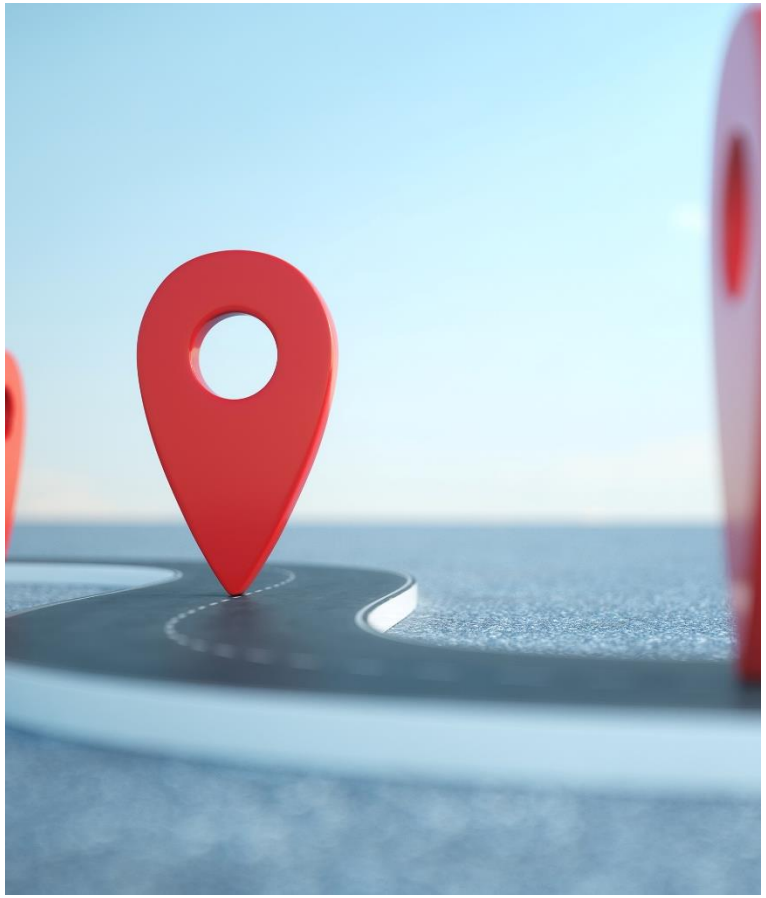
30 Minuten





Agenda

- Pseudocode
- Programmablaufplan (PAP)
- Struktogramme
- UML-Diagramme
  - Klassendiagramme
  - Sequenzdiagramme
  - Aktivitätsdiagramme
  - Zustandsdiagramme



Struktogramm

Was ist das?

- Auch bekannt als „*Nassi-Shneiderman-Diagramm*“
- In DIN 66 261 genormt
- Darstellung von Programmentwürfen im Rahmen der Methode der strukturierten Programmierung
  - Gesamtproblem in immer kleinere Teilprobleme zerlegt
  - … bis nur noch elementare Sequenzen und Kontrollstrukturen übrig bleiben

Aufgabe

- In den Anfängen der Programmierung herrschte der teilweise heute noch praktizierte lineare Programmierstil. Er ist für kleinere Aufgaben zwar sinnvoll, führt aber bei großen Programmieraufgaben zu einigen Nachteilen
  - Erläutere vier Nachteile des linearen Programmierstils, die zur Entwicklung der strukturierten Programmierung führten.
  - Erläutere die systematische Vorgehensweise bei der strukturierten Programmierung.

20 Minuten

Aufgabe - Lösungsvorschlag

- Erläutere vier Nachteile des linearen Programmierstils, die zur Entwicklung der strukturierten Programmierung führten.
  - Spaghetti-Technik: zahlreiche Verzweigungen mit Vor- und Rückwärtssprüngen
  - Segmentierung nicht möglich: Zerlegung in von verschiedenen Entwicklern zu erstellende Teile (Teamarbeit)
  - Unübersichtlichkeit
  - Schlechte Wart- und Änderbarkeit
- Erläutere die systematische Vorgehensweise bei der strukturierten Programmierung.

Aufgabe - Lösungsvorschlag

- Erläutere die systematische Vorgehensweise bei der strukturierten Programmierung.
  - Strukturierte Programmierung: systematisierter Prozess der Programmentwicklung mit dem Ziel, bessere Übersichtlichkeit und Wartbarkeit zu gewährleisten: Zerlegung des Programms in unabhängige Strukturblöcke (Top-Down)

Struktogramm

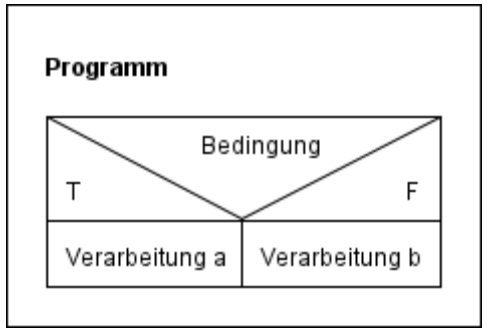
Prozess / Anweisung / Verarbeitung

Struktogramm

Reihenfolge (Sequenz)

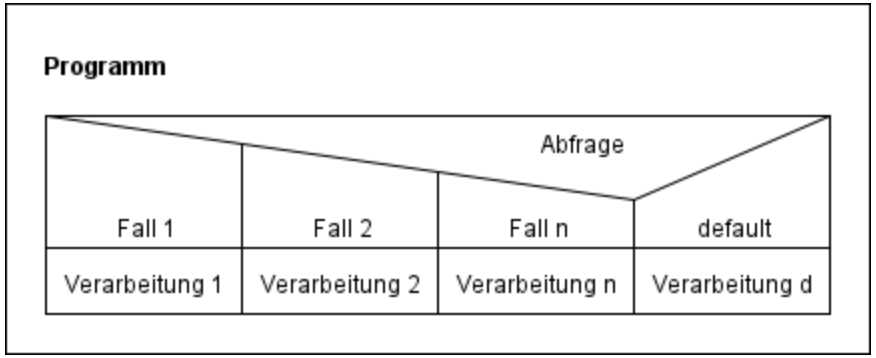
Struktogramm

Bedingte Verzweigung



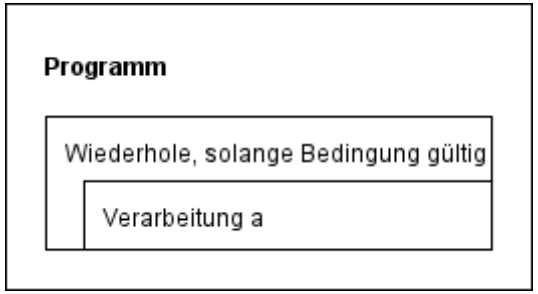
Struktogramm

Fallabfrage, Fallunterscheidung



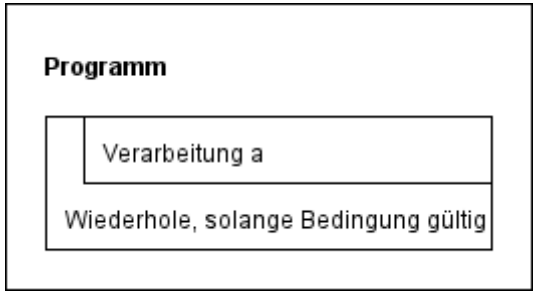
Struktogramm

Wiederholung (kopfgesteuerte Schleife)



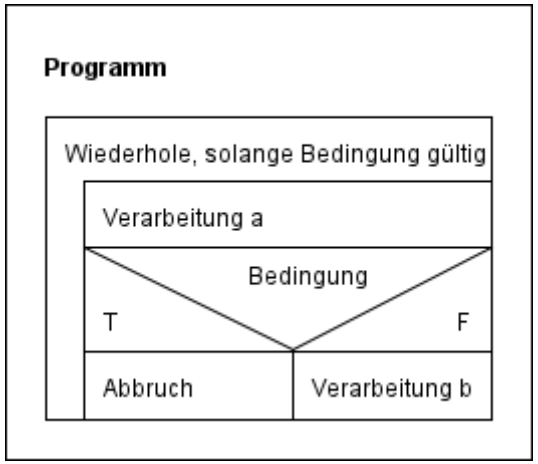
Struktogramm

Wiederholung (fußgesteuerte Schleife)



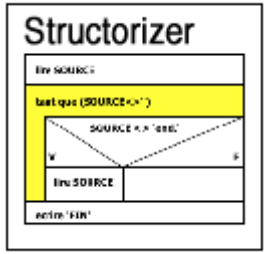
Struktogramm

Schleife mit Unterbrechung



Struktogramm

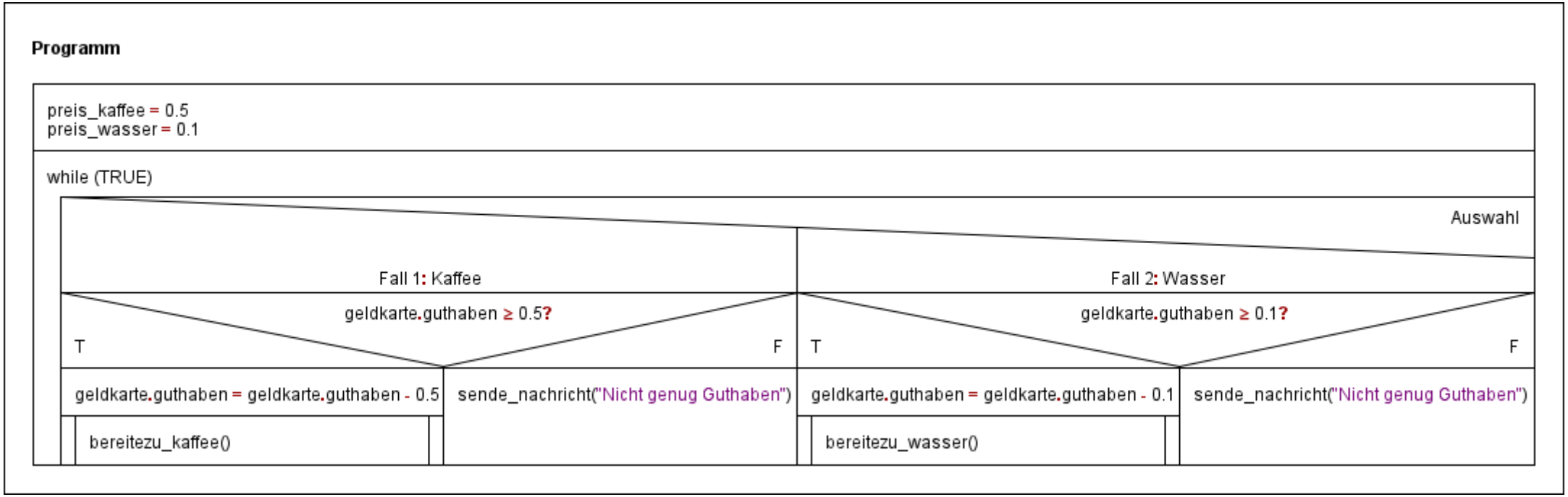
Software



Übung

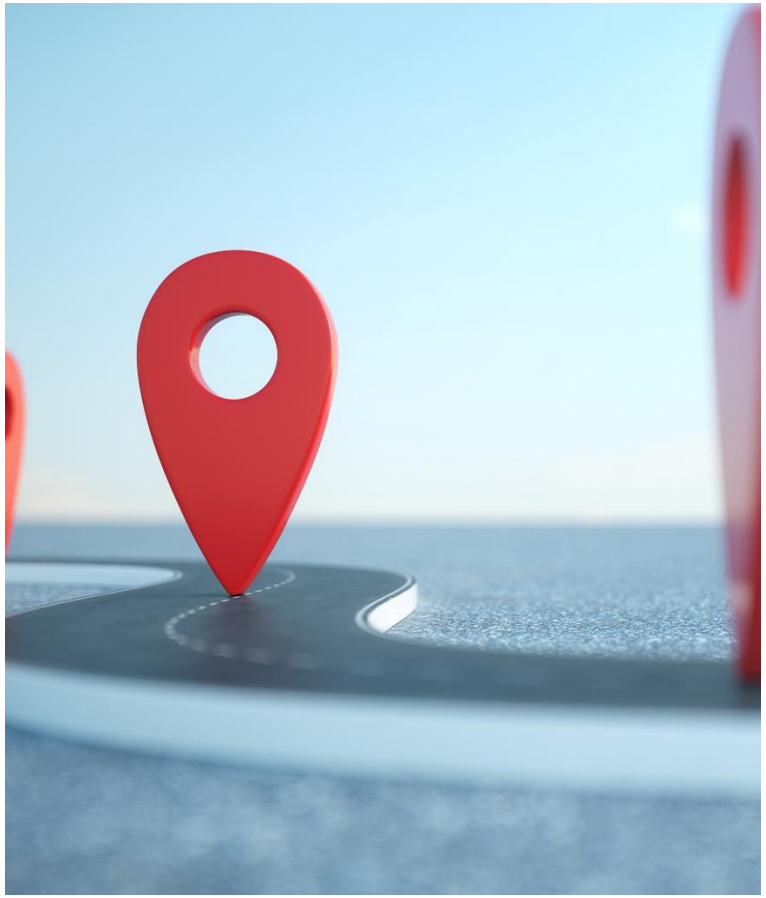
- Die Funktionsweise (vereinfacht) der Kaffeemaschine soll nun als Struktogramm dargestellt werden.
- Die Kaffeemaschine hat einen Chipkartenleser für die Zahlung der Getränke.
- Zur Auswahl stehen eine Tasse Kaffee oder heißes Wasser für einen Tee.
- Die Tasse Kaffee kostet 0,50€ und heißes Wasser für eine Tasse 0,10€.
- Die Maschine wechselt am Ende wieder in den Initialzustand.

30 Minuten



Agenda

- Pseudocode
- Programmablaufplan (PAP)
- Struktogramme
- UML-Diagramme
  - Klassendiagramme
  - Sequenzdiagramme
  - Aktivitätsdiagramme
  - Zustandsdiagramme



Was ist das?

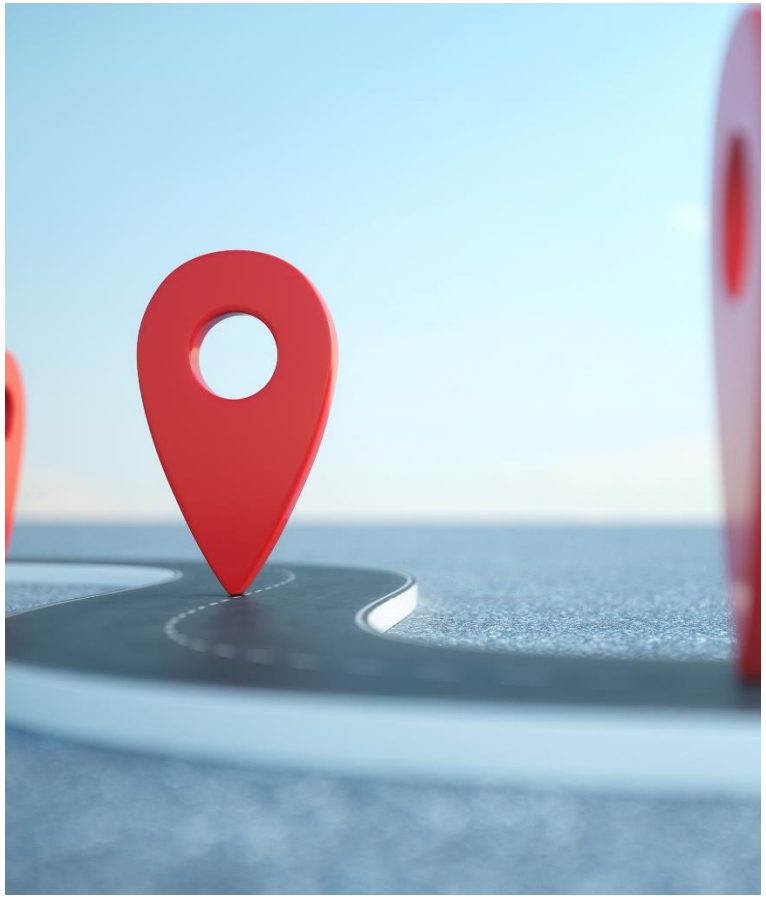
- Unified Modeling Language
- UML Klassendiagramm
  - grafischen Darstellung (Modellierung) von Klassen, Schnittstellen sowie deren Beziehungen
  - Eine Klasse ist in der Objektorientierung ein abstrakter Oberbegriff für die Beschreibung der gemeinsamen Struktur und des gemeinsamen Verhaltens von Objekten
- UML Sequenzdiagramm
  - Verhaltensdiagramm, welches eine Interaktion im Sinne der UML grafisch darstellt
  - beschreiben den Austausch von Nachrichten zwischen Objekten mittels Lebenslinien

Was ist das?

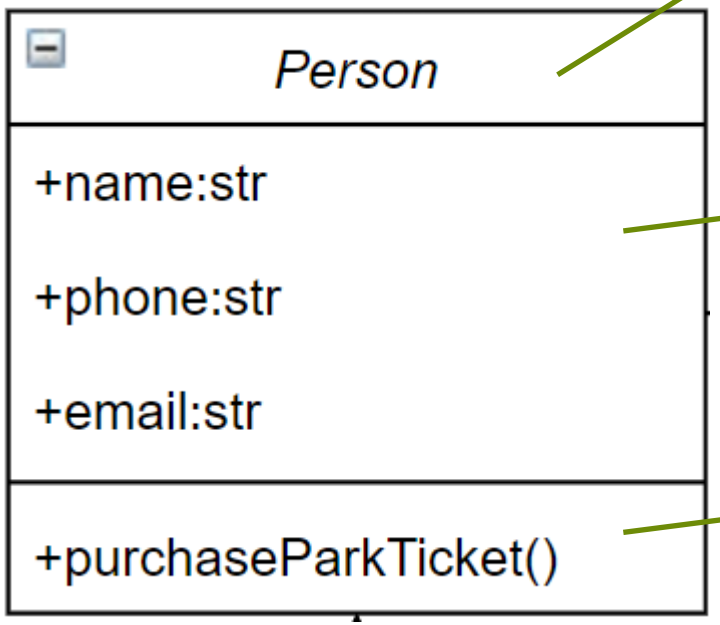
- UML Aktivitätsdiagramm
  - Vernetzung von elementaren Aktionen und deren Verbindungen mit Kontroll- und Datenflüssen
  - Beschreibt den Ablauf eines Anwendungsfalls
- UML Zustandsdiagramm
  - stellt einen endlichen Automaten in einer UML-Sonderform grafisch dar und wird benutzt, um entweder das Verhalten eines Systems oder die zulässige Nutzung der Schnittstelle eines Systems zu spezifizieren

Agenda

- Pseudocode
- Programmablaufplan (PAP)
- Struktogramme
- UML-Diagramme
  - Klassendiagramme
  - Sequenzdiagramme
  - Aktivitätsdiagramme
  - Zustandsdiagramme



Klassen



Klassenname

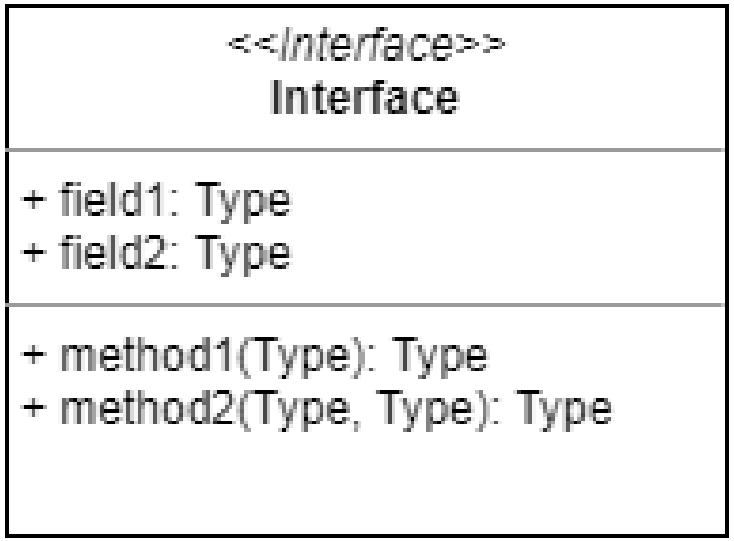
Attribute der Klasse

Methoden der Klasse

Was bedeutet das „+“?

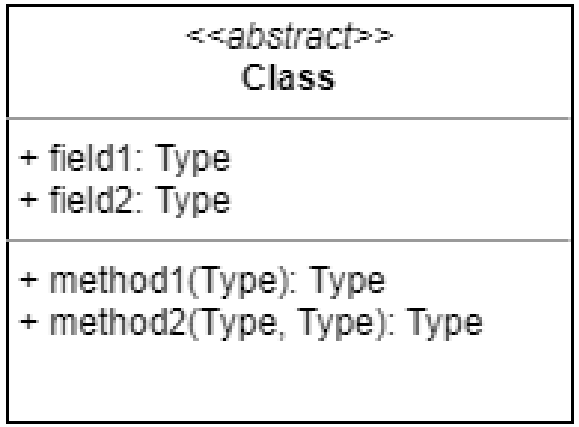
UML Klassendiagramm

Schnittstellen

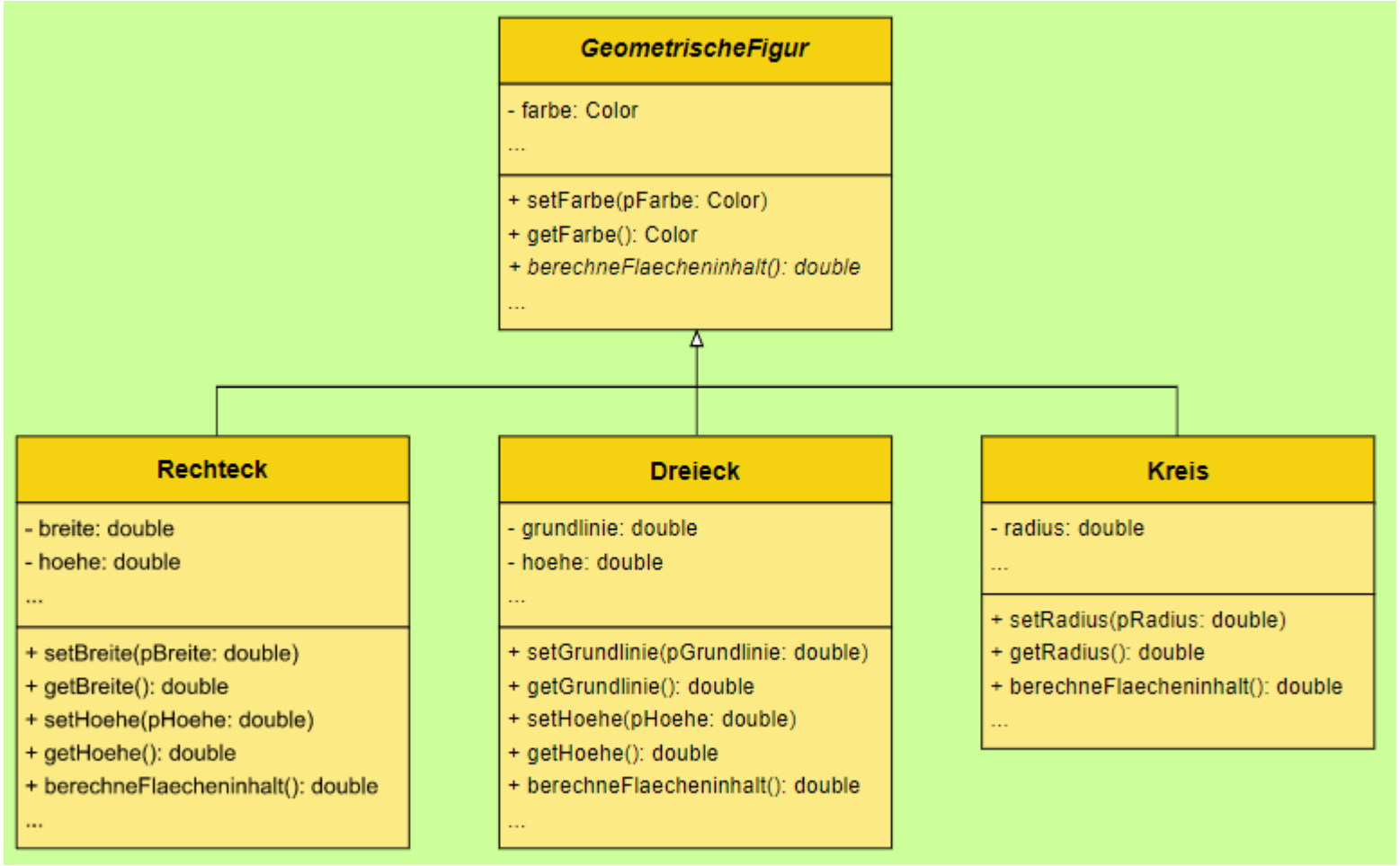


UML Klassendiagramm

Abstrakte Klassen

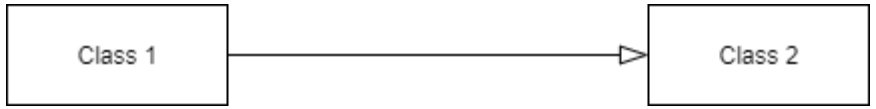


Abstrakte Klassen

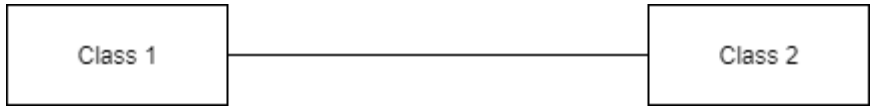


UML Klassendiagramm

Interaktionen - Vererbung

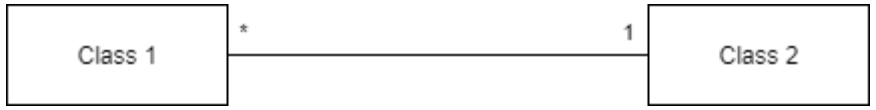


Interaktionen - Assoziation



UML Klassendiagramm

Kardinalitäten

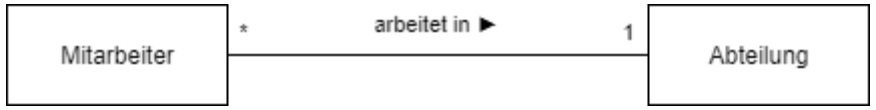


Kardinalitäten

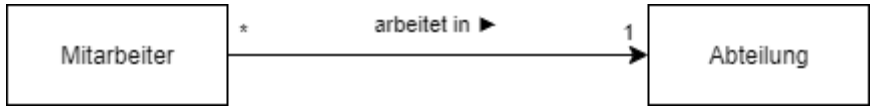
Multiplizität	Bedeutung
1	genau einer
0..1	keiner oder einer
1..5	einer bis fünf
*	keiner, einer oder mehrere
0..*	keiner, einer oder mehrere
1..*	mindestens einer

UML Klassendiagramm

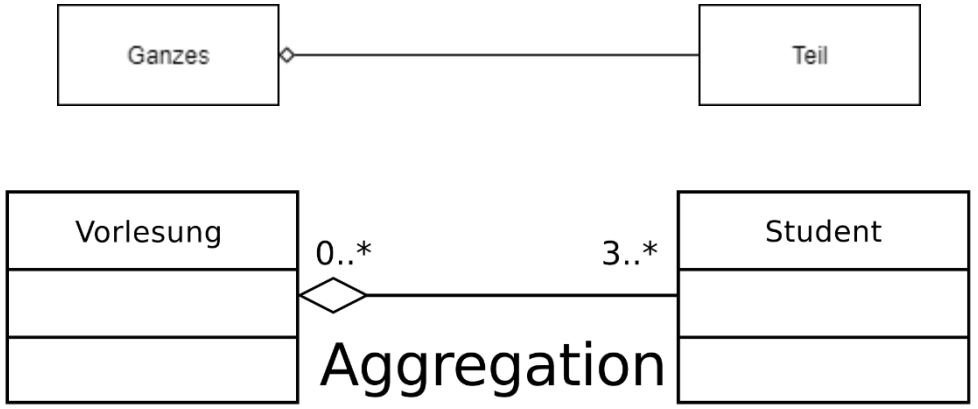
Kardinalitäten



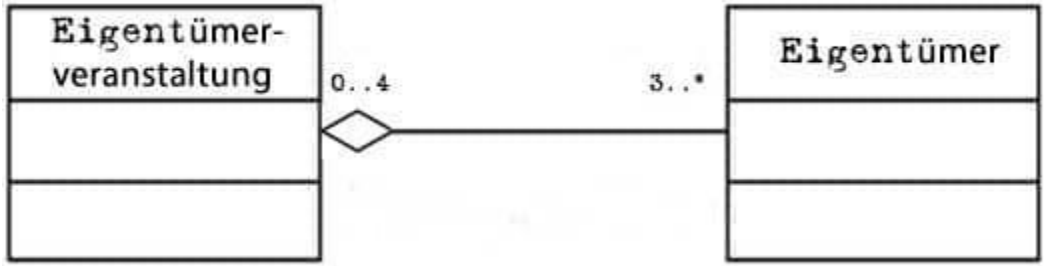
Interaktionen – Gerichtete Assoziation



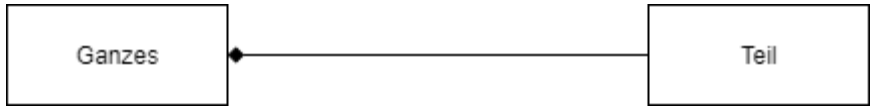
Interaktionen - Aggregation



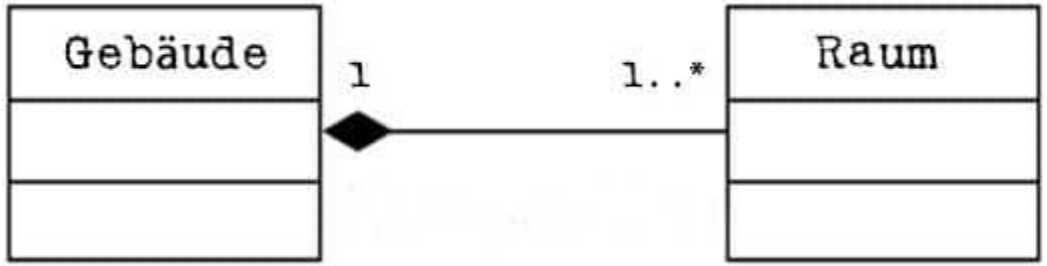
Interaktionen - Aggregation



Interaktionen - Komposition

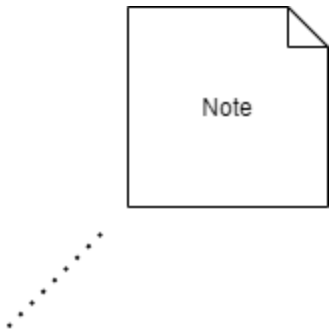


Interaktionen - Komposition



Interaktionen - Implementierung

Notiz



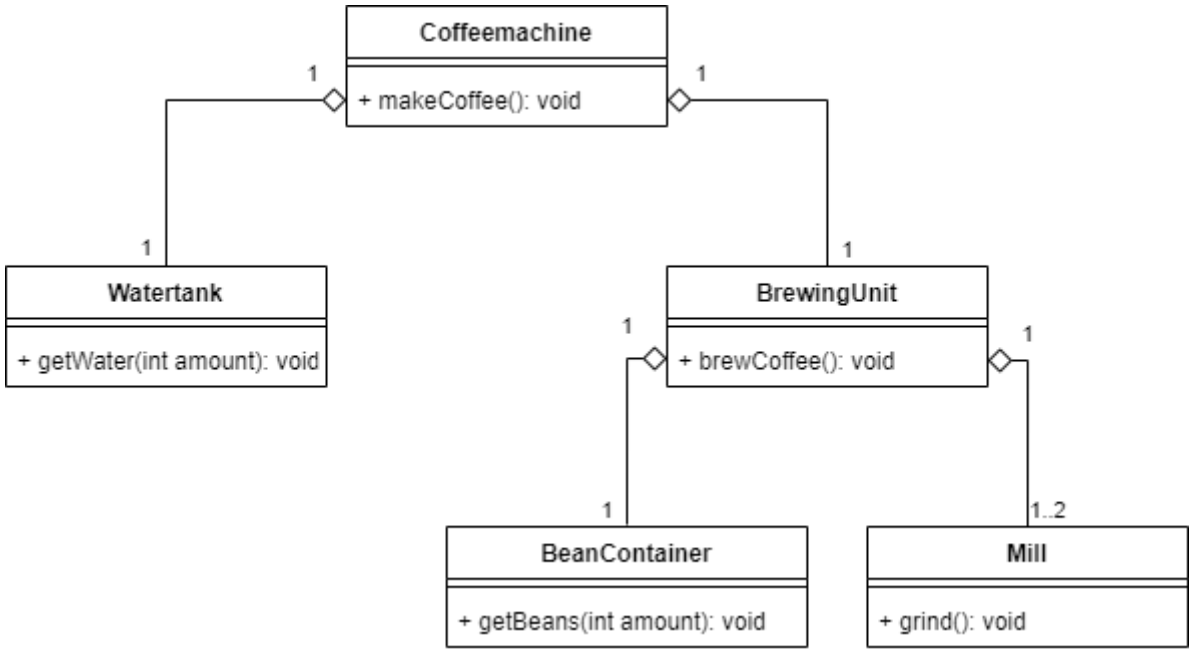
UML Klassendiagramm

Software

Übung

- Die Kaffeemaschine soll nun mit einem UML Klassendiagramm visualisiert werden.
- Die Maschine besteht (vereinfacht) aus den Komponenten Wassertank, Bohnenbehälter, Kaffeemühle und eine Brüheinheit.
- Erstelle sinnvolle Beziehungen zwischen den Komponenten und weise diesen Assoziationen Kardinalitäten zu.

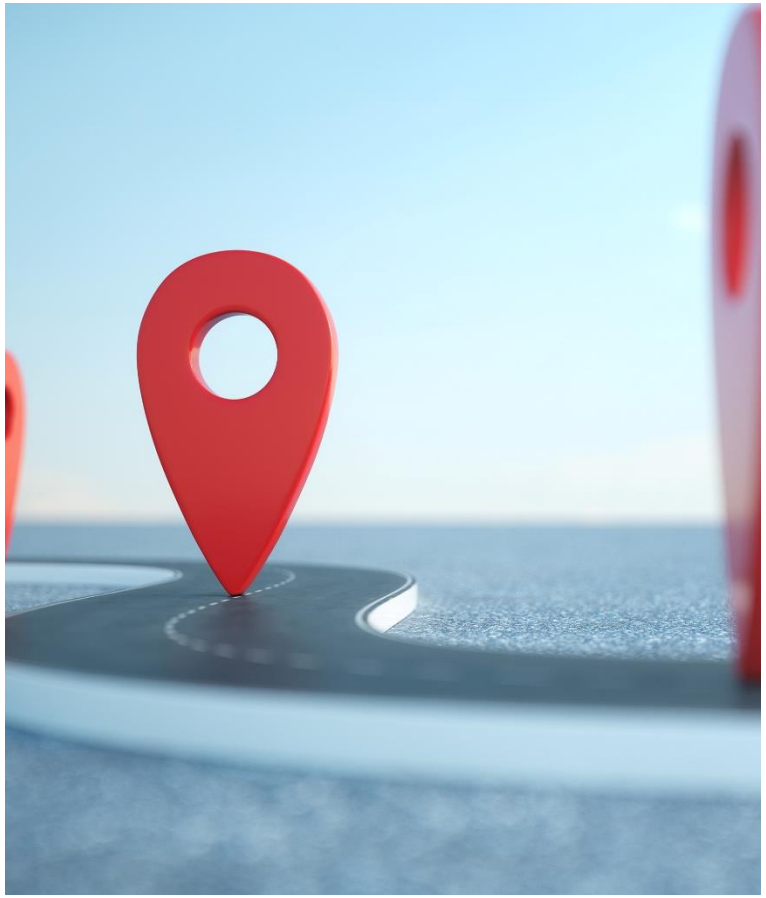
30 Minuten



Agenda

- Pseudocode
- Programmablaufplan (PAP)
- Struktogramme
- UML-Diagramme
  - Klassendiagramme
  - Sequenzdiagramme
  - Aktivitätsdiagramme
  - Zustandsdiagramme

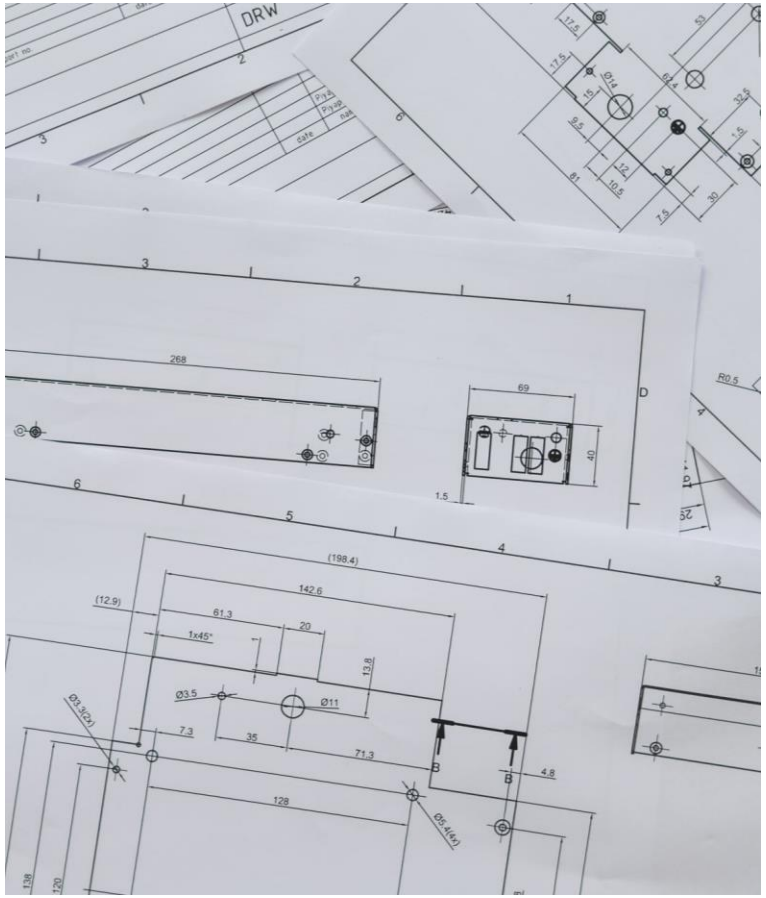
Flipped-Classroom

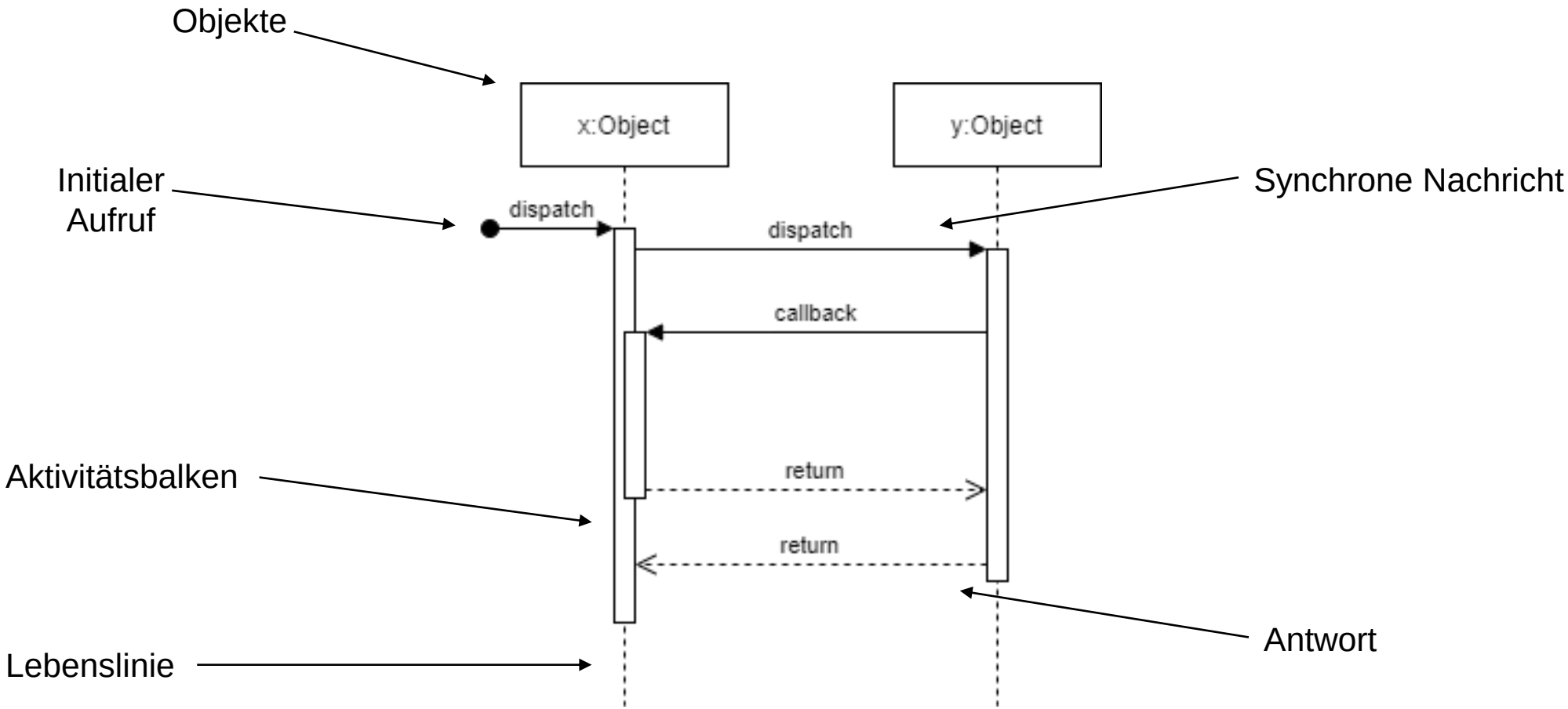


UML Sequenzdiagramm

Was ist das?

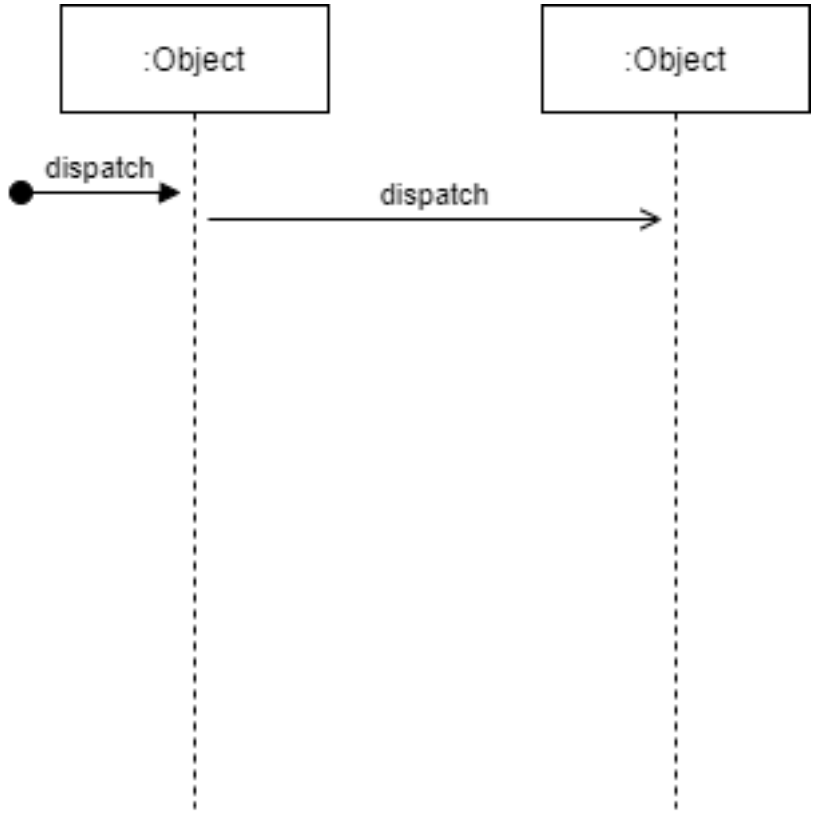
- Bildet die Interaktion zwischen einer Gruppe von Objekten sowie die Reihenfolge ab
- Praktisch, um Prozessabläufe grafisch festzuhalten





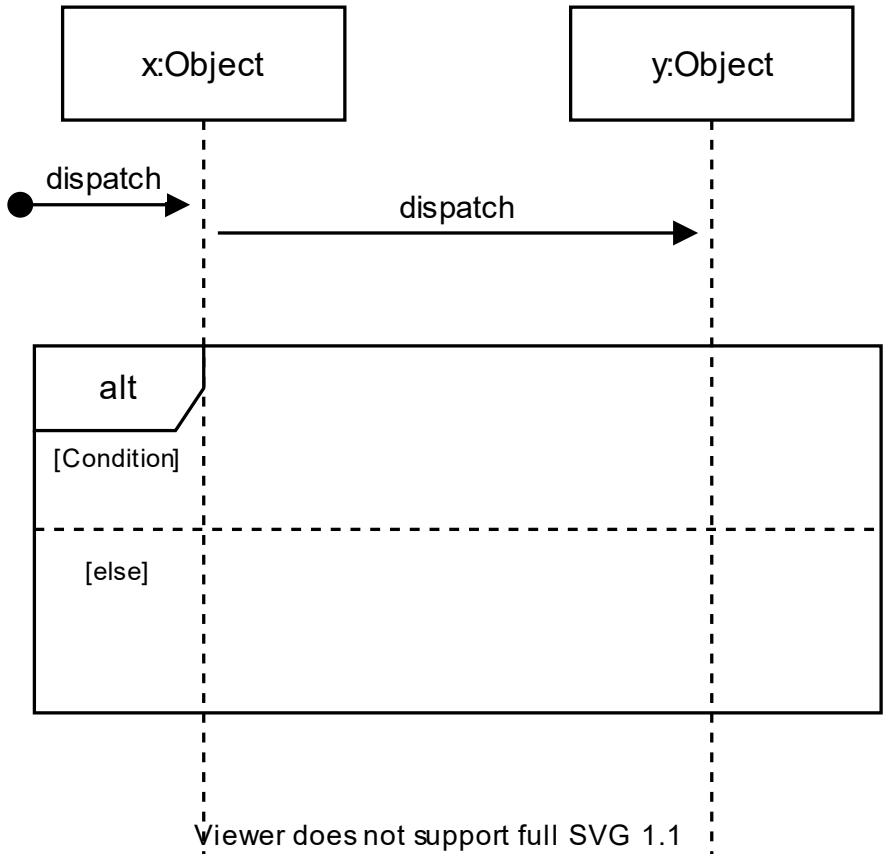
UML Sequenzdiagramm

Asynchrone Nachricht



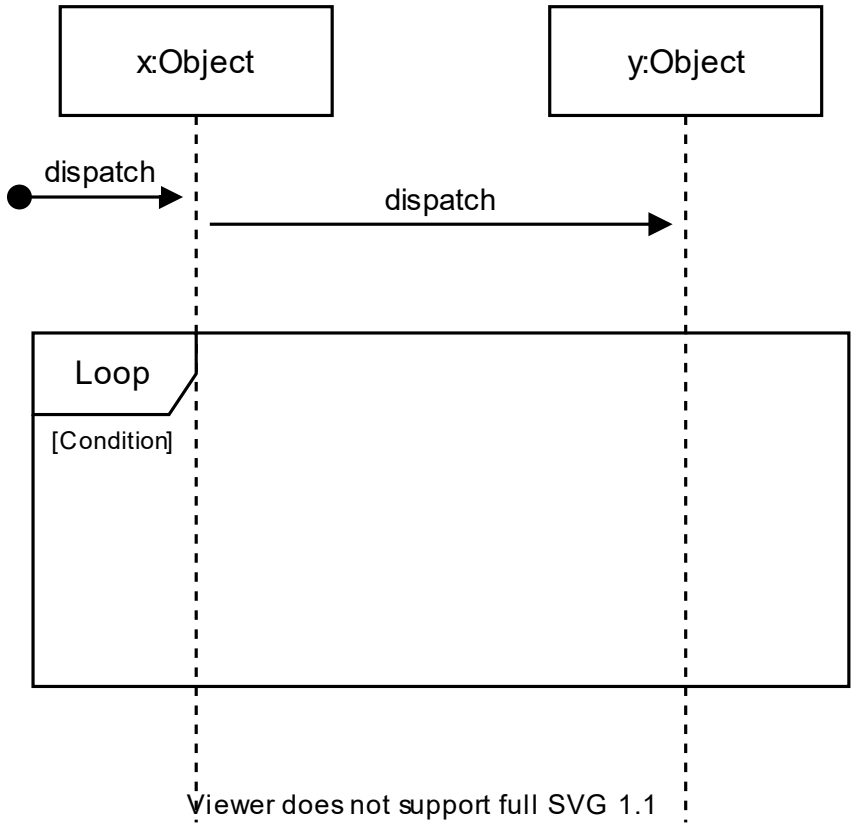
UML Sequenzdiagramm

If-Else



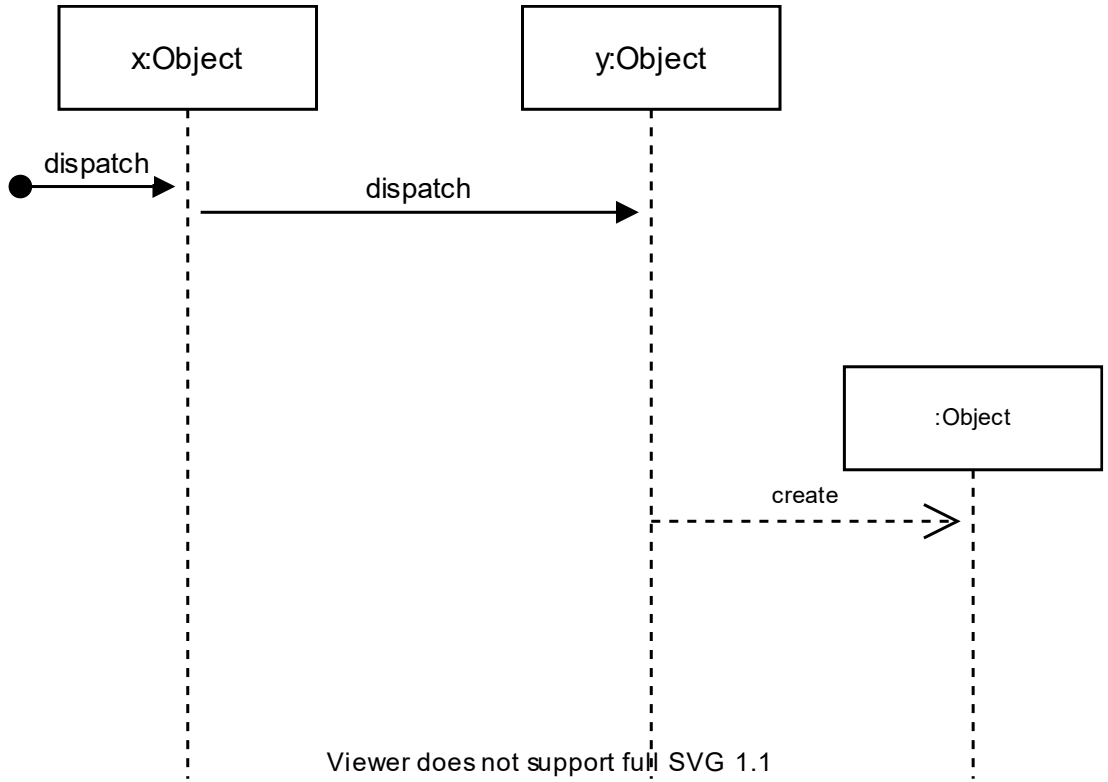
UML Sequenzdiagramm

Optionsschleife



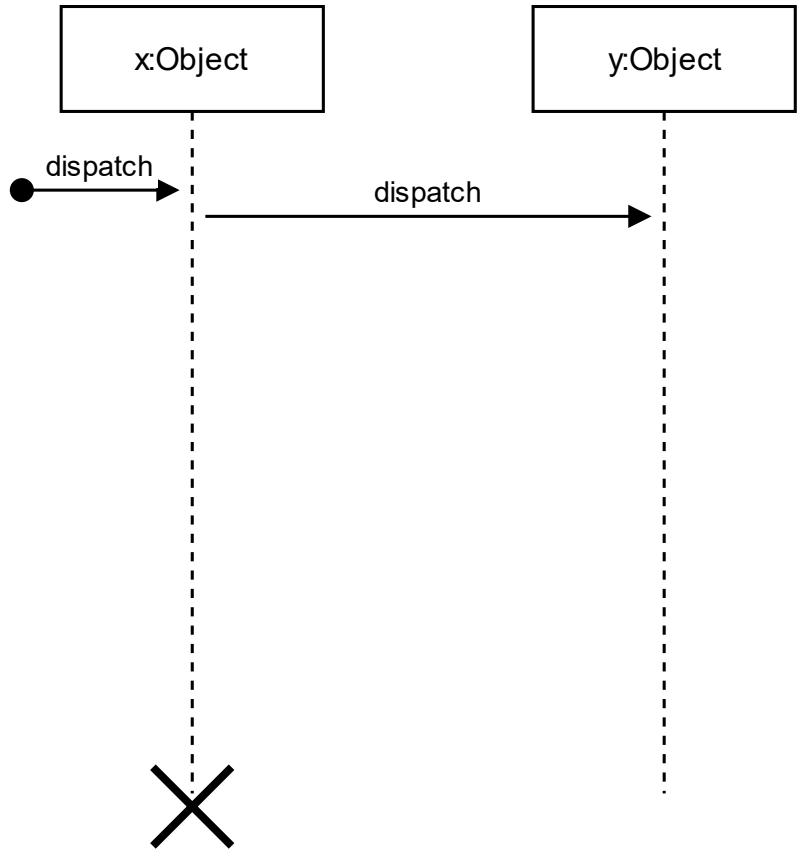
UML Sequenzdiagramm

Objektkonstruktion



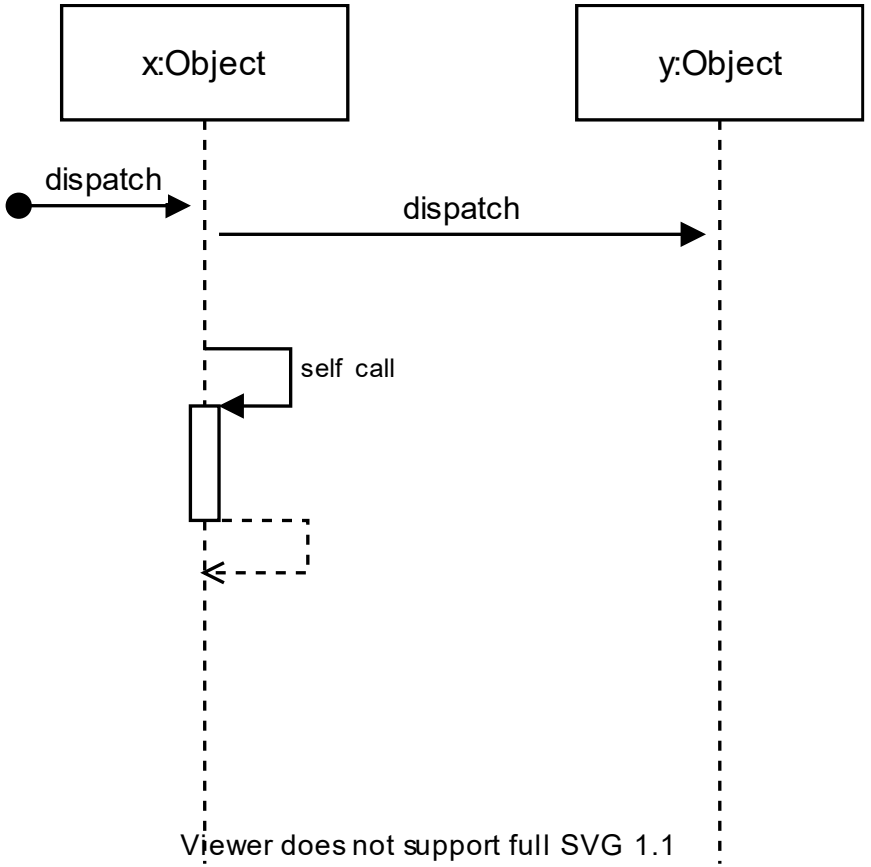
UML Sequenzdiagramm

Objektzerstörung



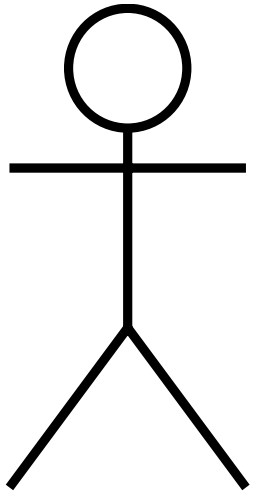
UML Sequenzdiagramm

Selbstaufruf



UML Sequenzdiagramm

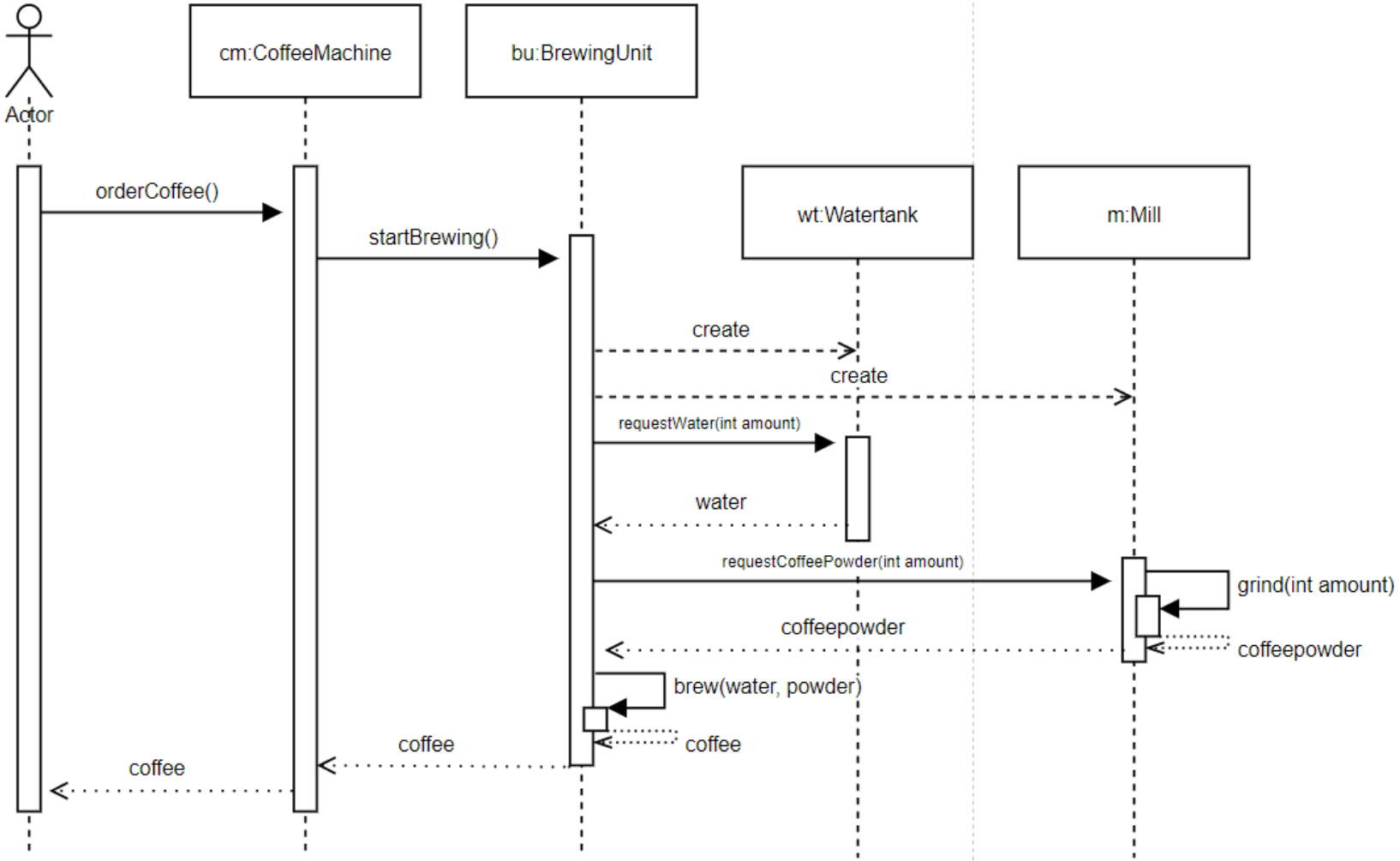
Akteur



Übung

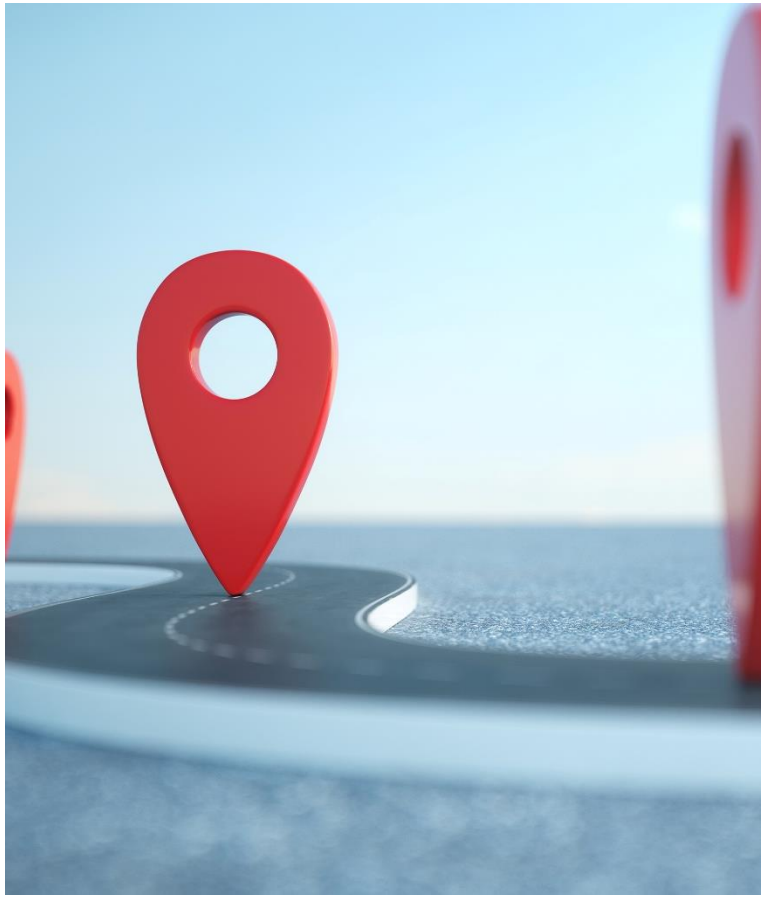
- Die Kaffeemaschine soll nun mit einem UML Sequenzdiagramm visualisiert werden.
- Die Maschine besteht (vereinfacht) aus den Komponenten Wassertank, Bohnenbehälter, Kaffeemühle und eine Brüheinheit.
- Erstelle sinnvolle Beziehungen zwischen den Komponenten und baue den Nutzer als Akteur ein.

30 Minuten



Agenda

- Pseudocode
- Programmablaufplan (PAP)
- Struktogramme
- UML-Diagramme
  - Klassendiagramme
  - Sequenzdiagramme
  - Aktivitätsdiagramme
  - Zustandsdiagramme



Was ist das?

- Flussdiagramm, das die von einem System ausgeführten Aktivitäten abbildet
- Hilfreich für Organisationsmitglieder auf Geschäfts- und Entwicklungsseite, eine gemeinsame Basis zu finden
- Eignet sich besonders für …
  - Demonstration der Logik von Algorithmen
  - Beschreibung der Schritte eines Use-Case
  - Illustration von Geschäftsprozessen Exkurs

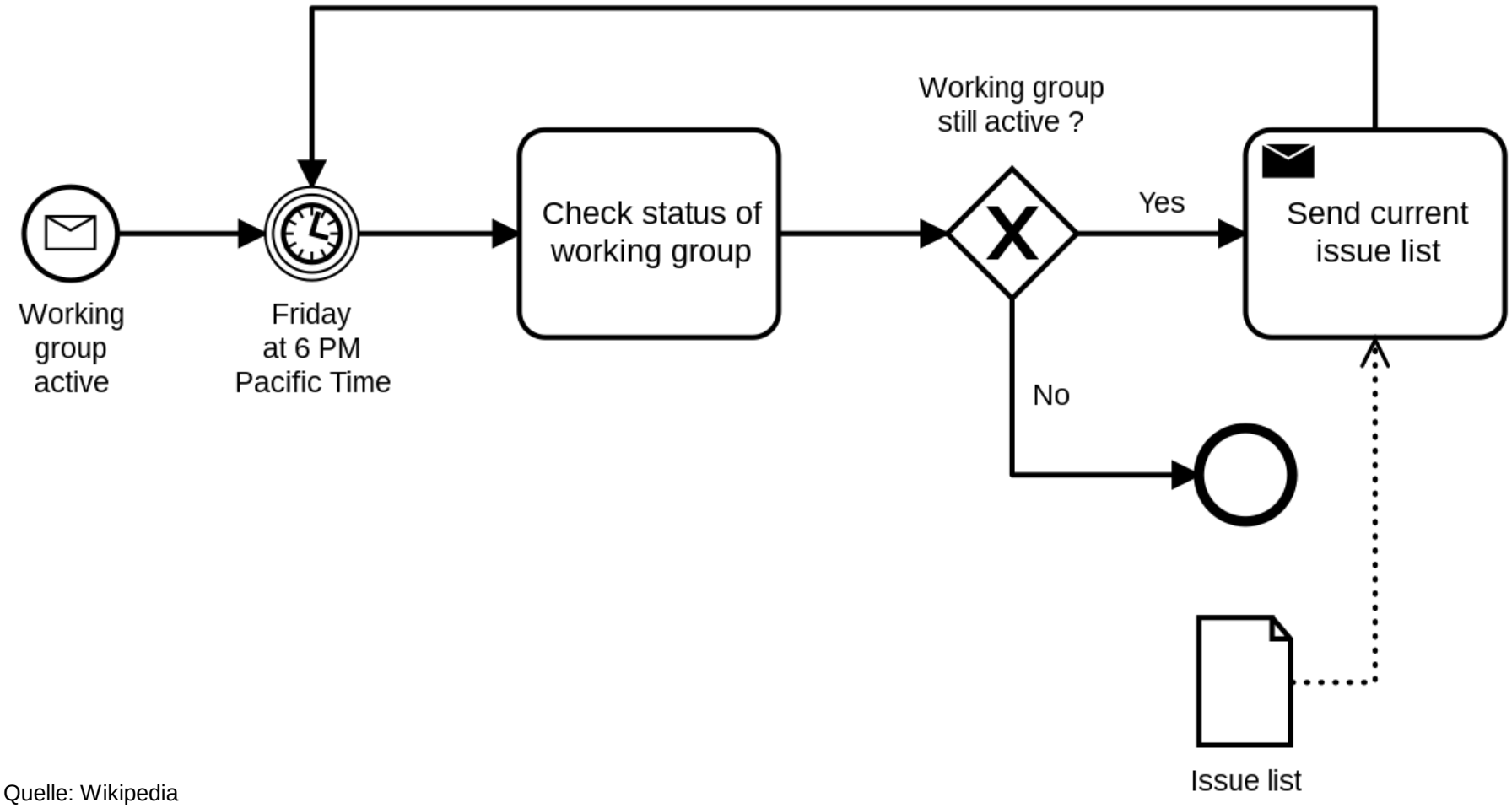
Exkurs

Business Process Model and Notation (BPMN)

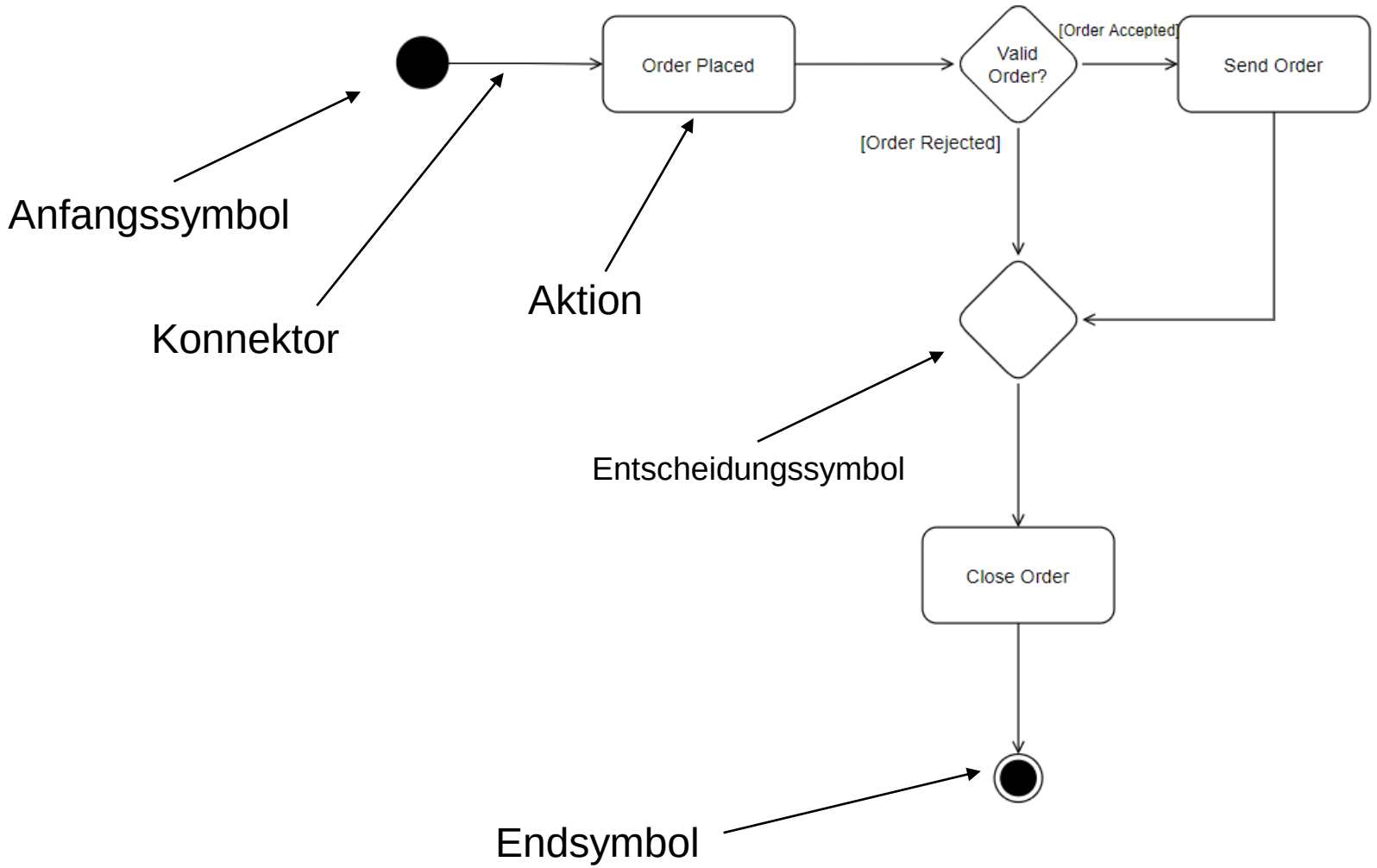
Geschäftsprozesse

- Spezielle Form für Geschäftsprozesse: Notationsstil BPMN
- Führender Standard für Geschäftsprozessmodelle
- Unterschied zu UML Aktivitätsdiagrammen?
  - UML ist objektorientiert
  - BPMN ist prozessorientiert
  - Verschiedene Sichtweisen

Exkurs

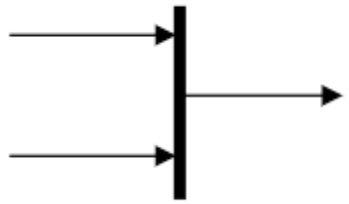


Quelle: Wikipedia

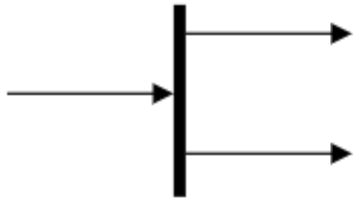


UML Aktivitätsdiagramm

Verbindungssymbol/Synchronisierung

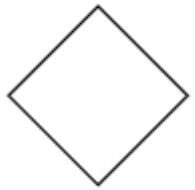


Verzweigungssymbol



UML Aktivitätsdiagramm

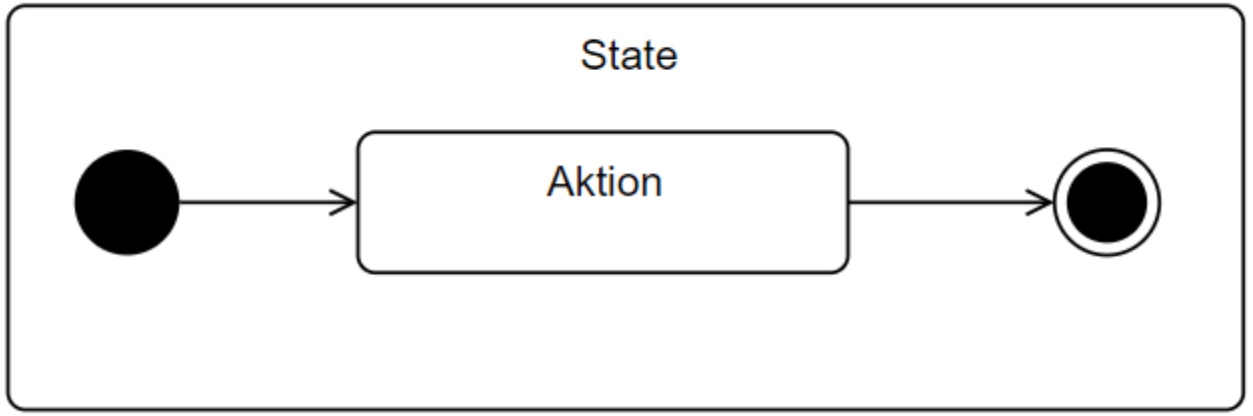
Entscheidungssymbol



UML Aktivitätsdiagramm

Hinweissymbol

Aktivität



Pseudostatus

UML Aktivitätsdiagramm

Activity Final Node

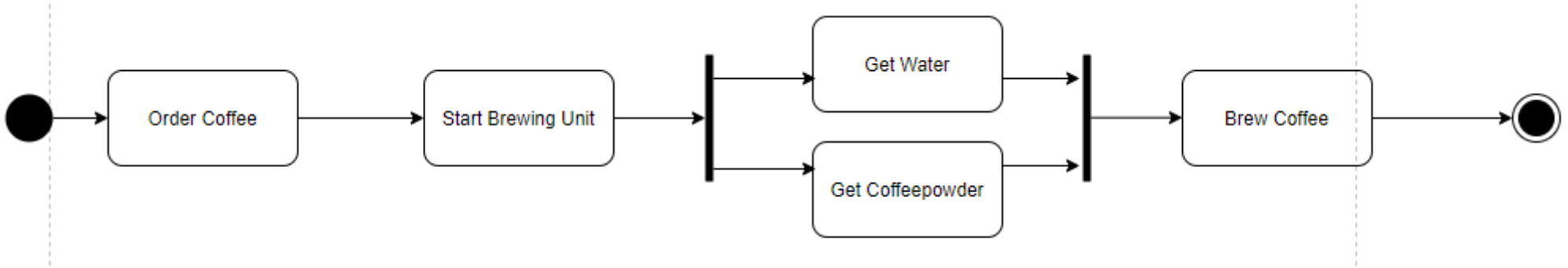
UML Aktivitätsdiagramm

Flow Final Node

Übung

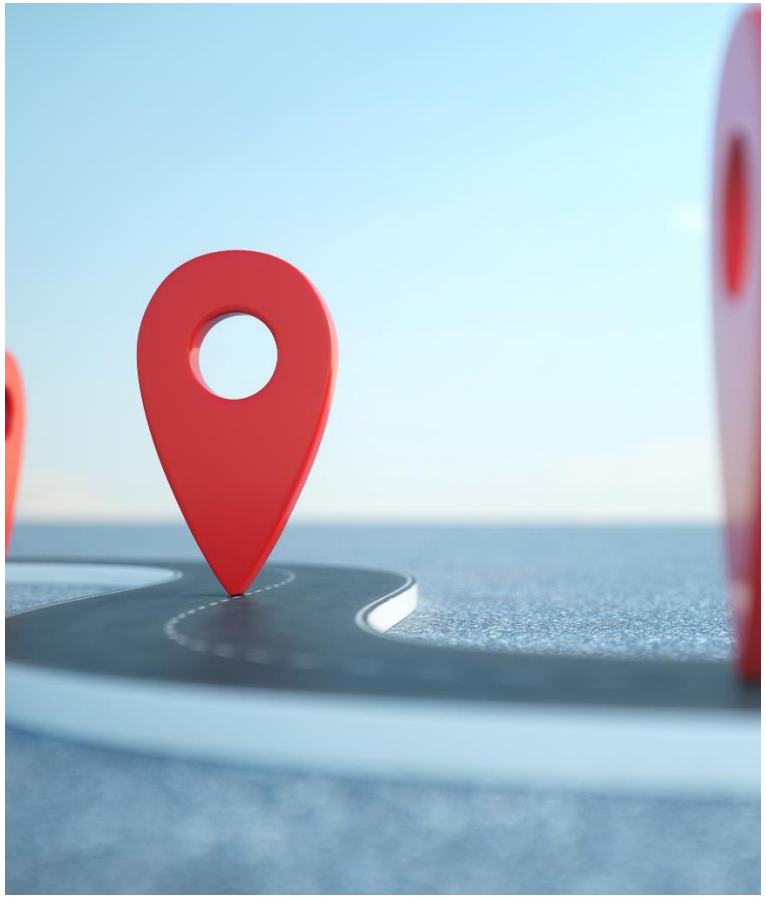
- Der Arbeitsablauf der Kaffeemaschine soll nun als UML Aktivitätsdiagramm dargestellt werden.
- Die Maschine besteht (vereinfacht) aus den Komponenten Wassertank, Bohnenbehälter, Kaffeemühle und eine Brüheinheit.
- Erstelle einen sinnvollen Ablauf für die Zubereitung einer Tasse Kaffee.

30 Minuten



Agenda

- Pseudocode
- Programmablaufplan (PAP)
- Struktogramme
- UML-Diagramme
  - Klassendiagramme
  - Sequenzdiagramme
  - Aktivitätsdiagramme
  - Zustandsdiagramme



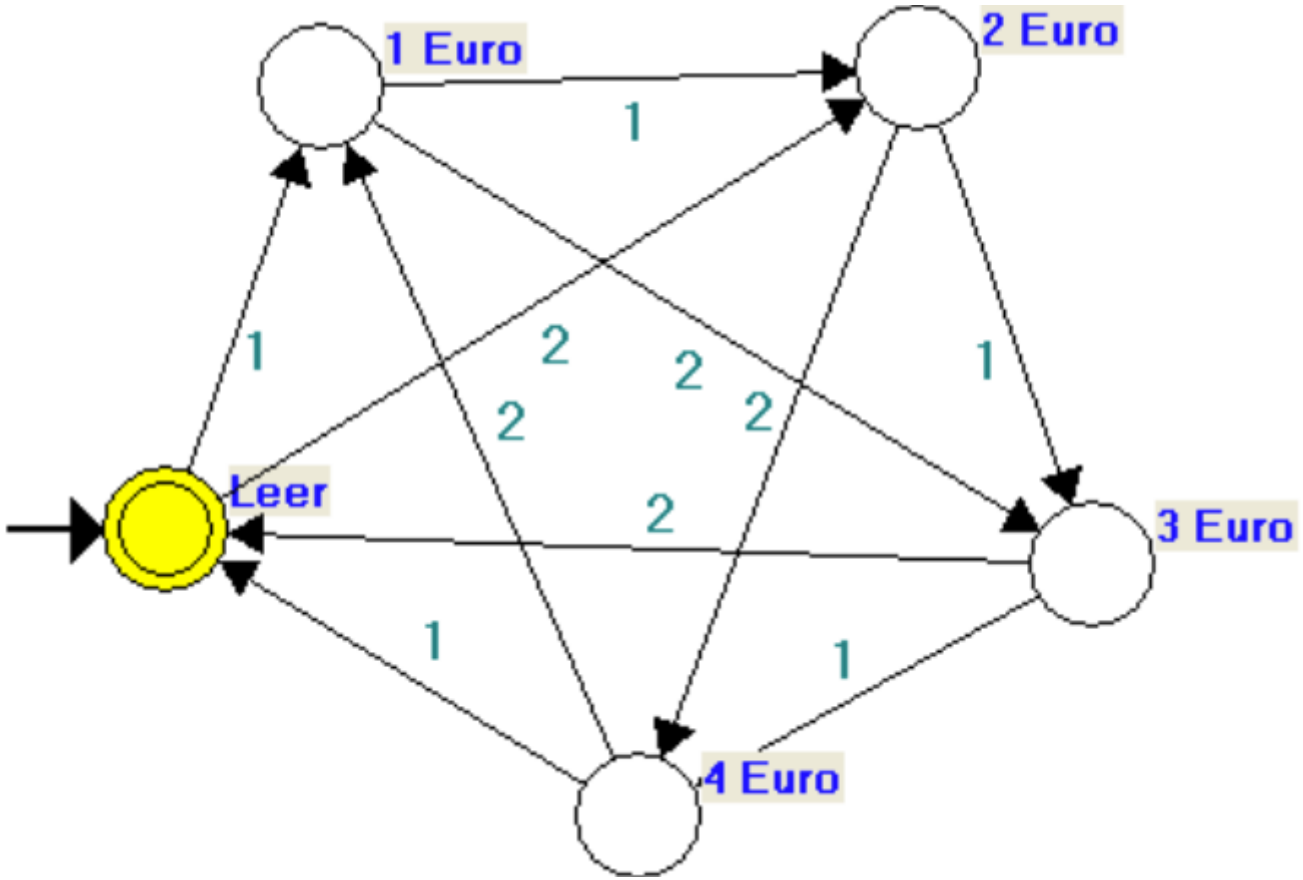
Was ist das?

- Visualisierung von Zuständen eines endlichen Automaten
- Ähnlichkeit zum Aktivitätsdiagramm
- Aktionen, Zustände, Zustandsübergänge

Endlicher Automat? Was ist denn das?

Was ist das?

- Stelle den nachfolgenden Parkautomaten formal als endlichen Automaten dar
  - Ein Parkautomat verkauft Parktickets in Höhe von 5€. Als Einwurf werden nur 1€- und 2€-Stücke akzeptiert. Bei Überbezahlung entspricht der Rest dem aktuellen Zahlstatus für das nächste Ticket.



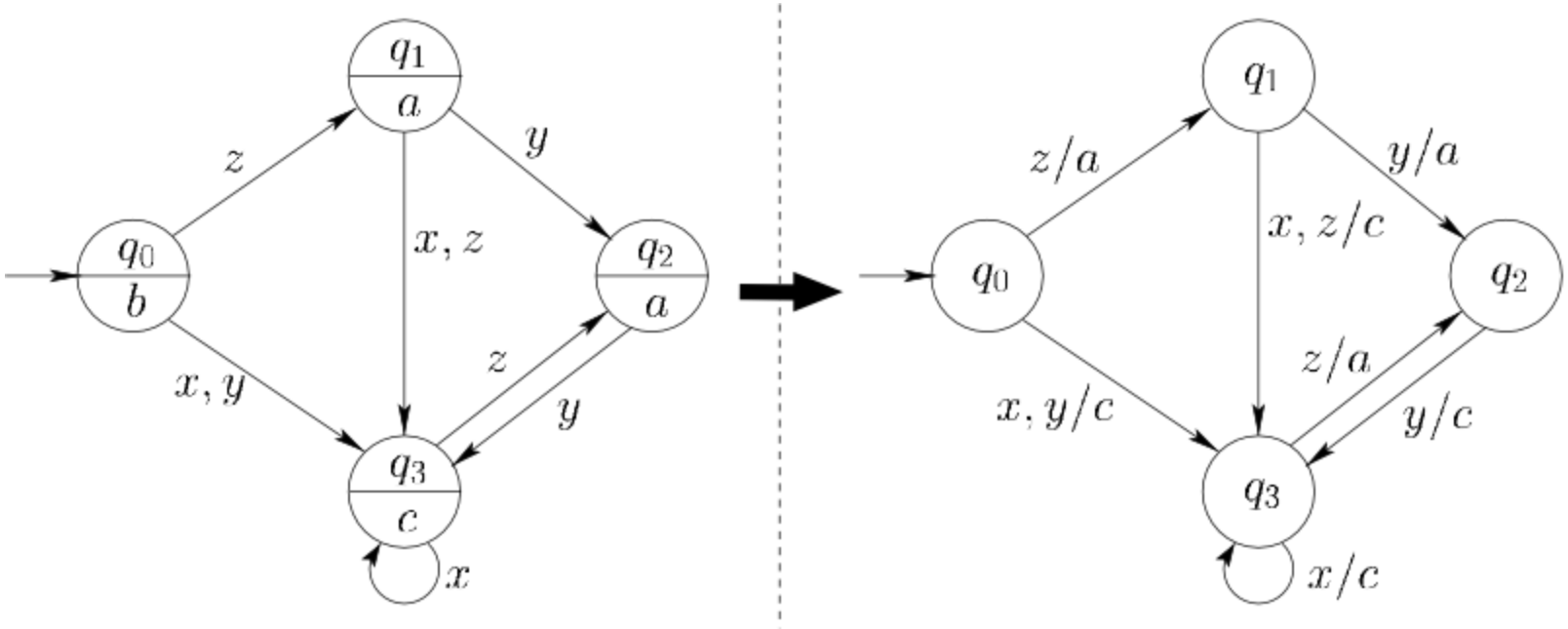
Exkurs

Endlicher Automat

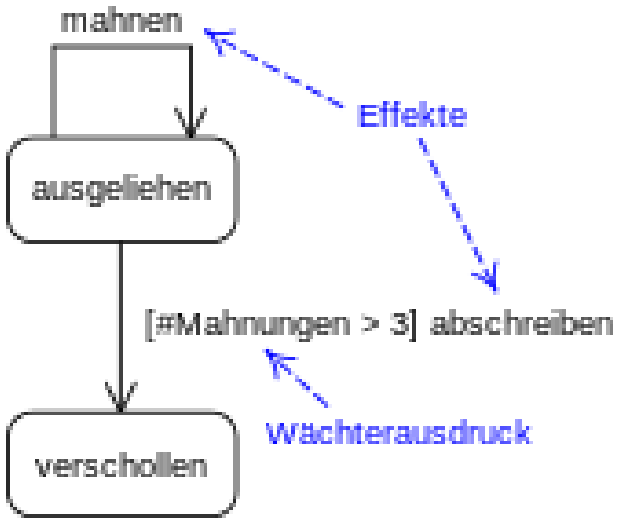
- Englisch: „Finite State Machine“
- Kombination aus Zuständen und Zustandsübergängen
- Je nach aktuellem Zustand und Eingabe können unterschiedliche Aktionen erfolgen
- Formale Präzisierung durch *5-Tupel*
  - Endliche Menge von Zuständen
  - Eingabealphabet
  - Überführungsfunktion, die für jeden Zustand mit jeder Eingabe einen Folgezustand definiert
  - Menge von Startzuständen
  - Menge von Endzuständen

Moore

Mealy



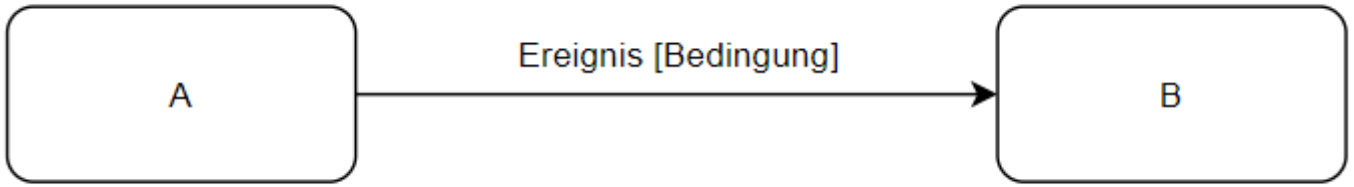
Quelle: dewiki



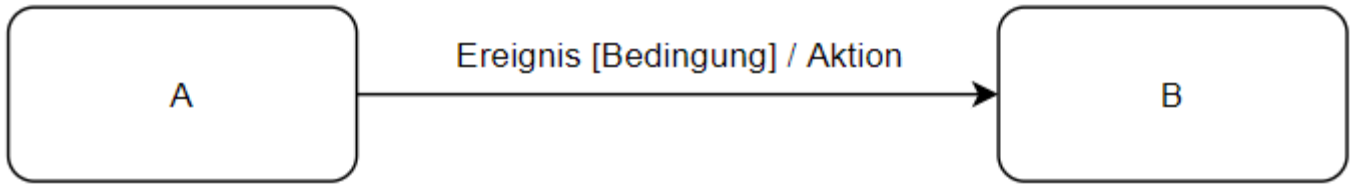
UML Zustandsdiagramm

Start- und Endzustand

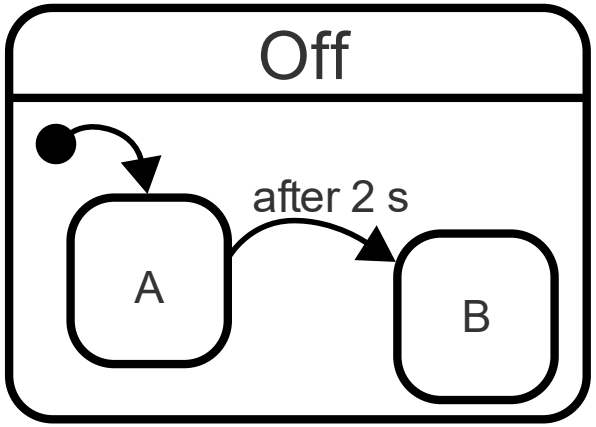
Bedingte Zustandsübergänge



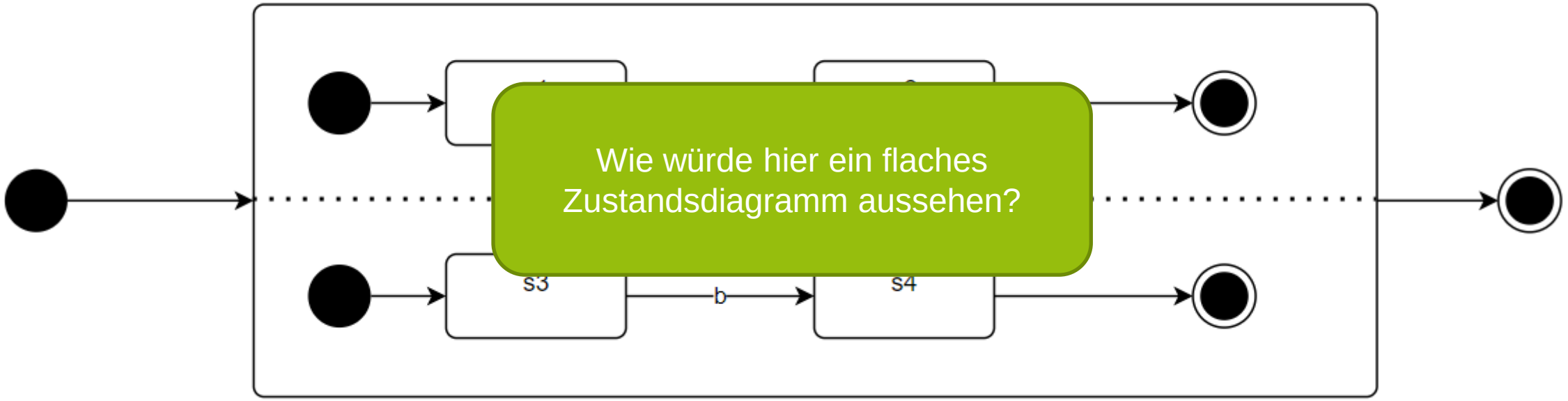
Aktionen bei Zustandsübergängen



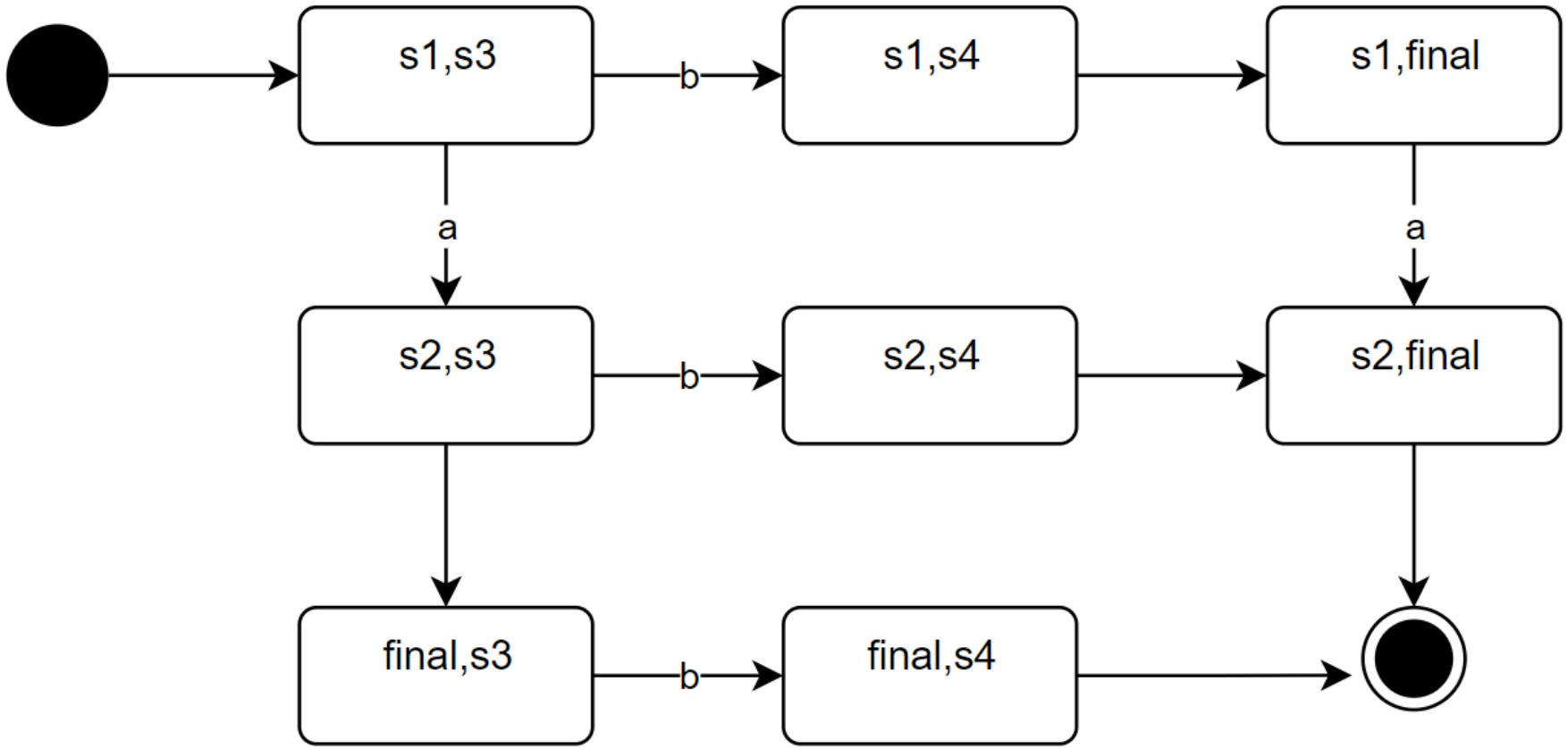
Zusammengesetzter Zustand



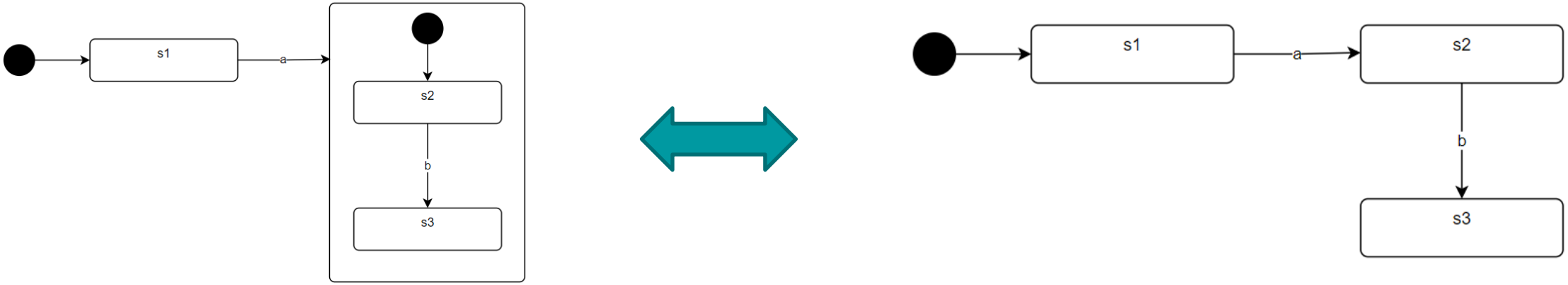
AND / Parallel state



AND / Parallel state (flaches Zustandsdiagramm)

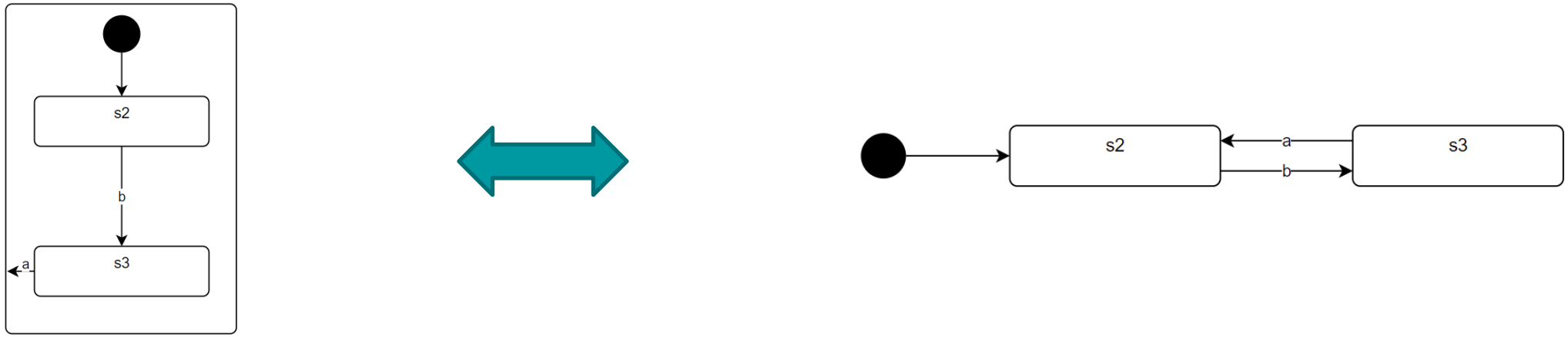


XOR-Zustände



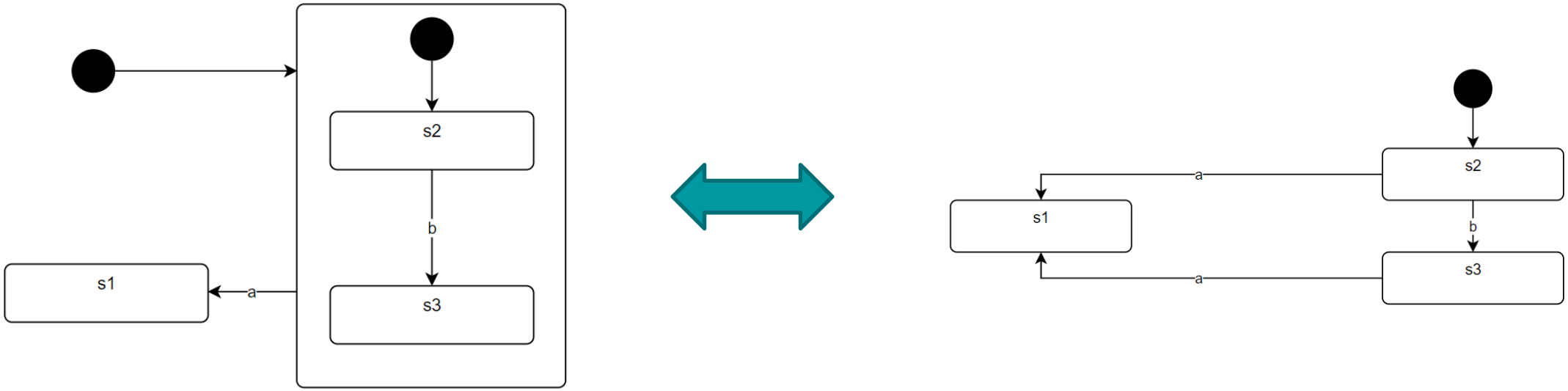
UML Zustandsdiagramm

XOR-Zustände



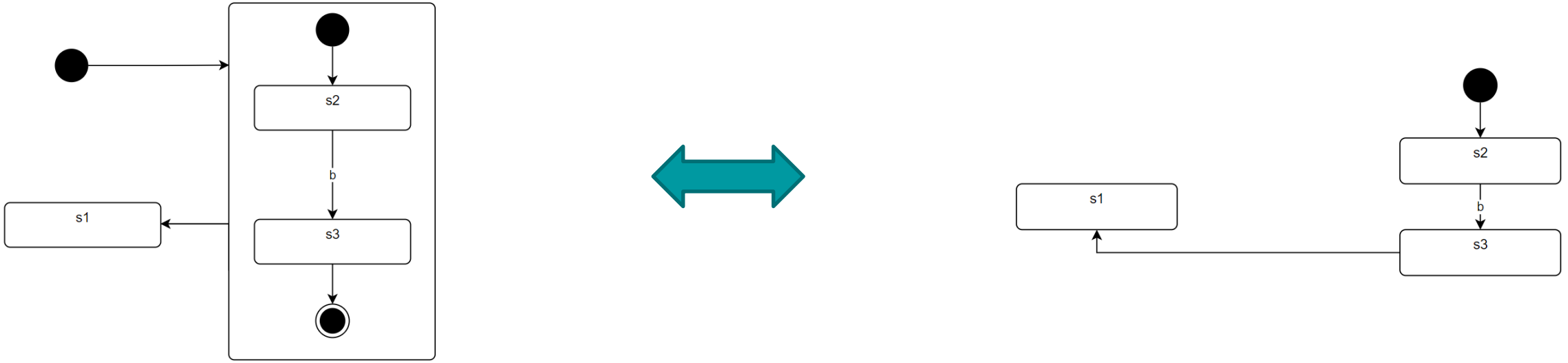
UML Zustandsdiagramm

XOR-Zustände

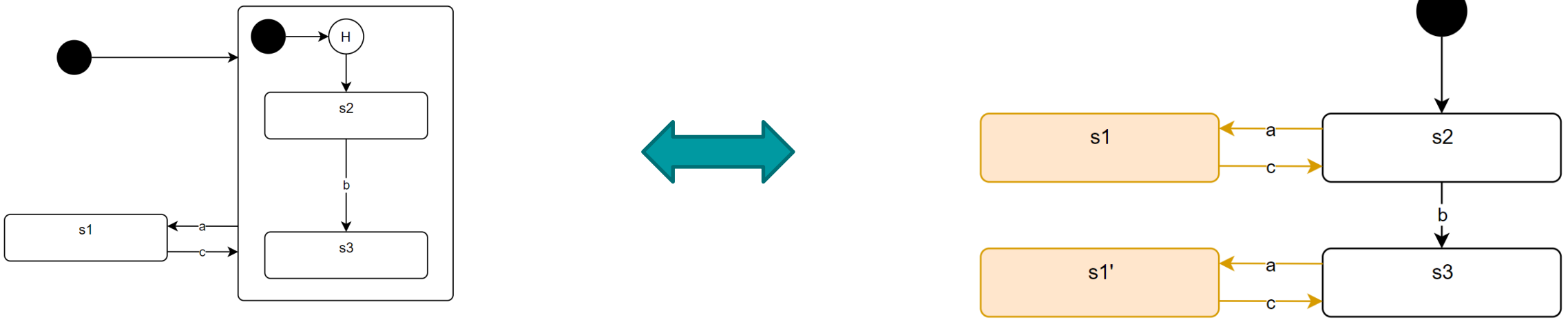


UML Zustandsdiagramm

XOR-Zustände

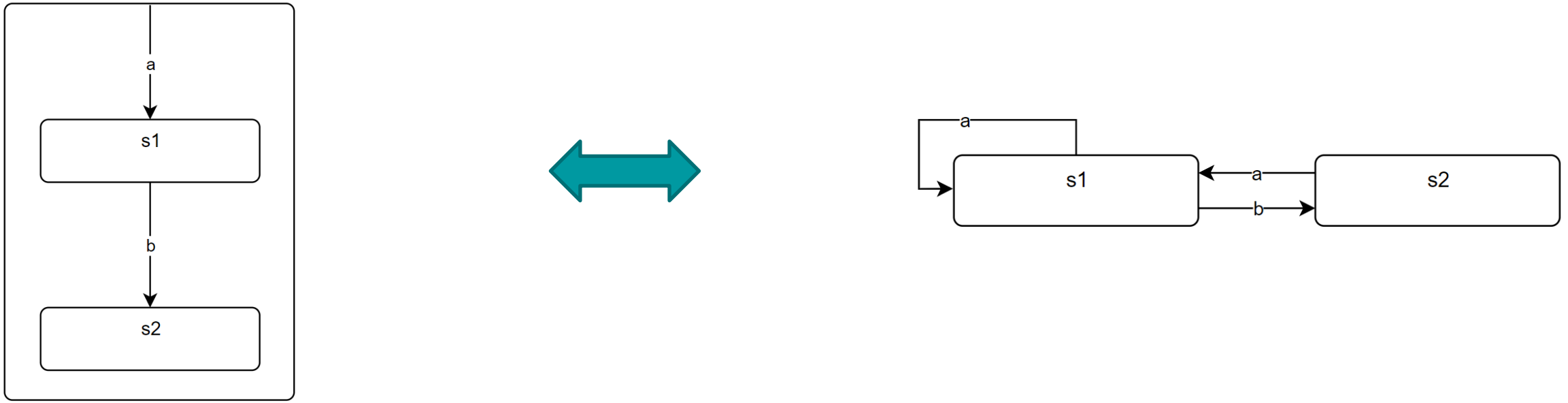


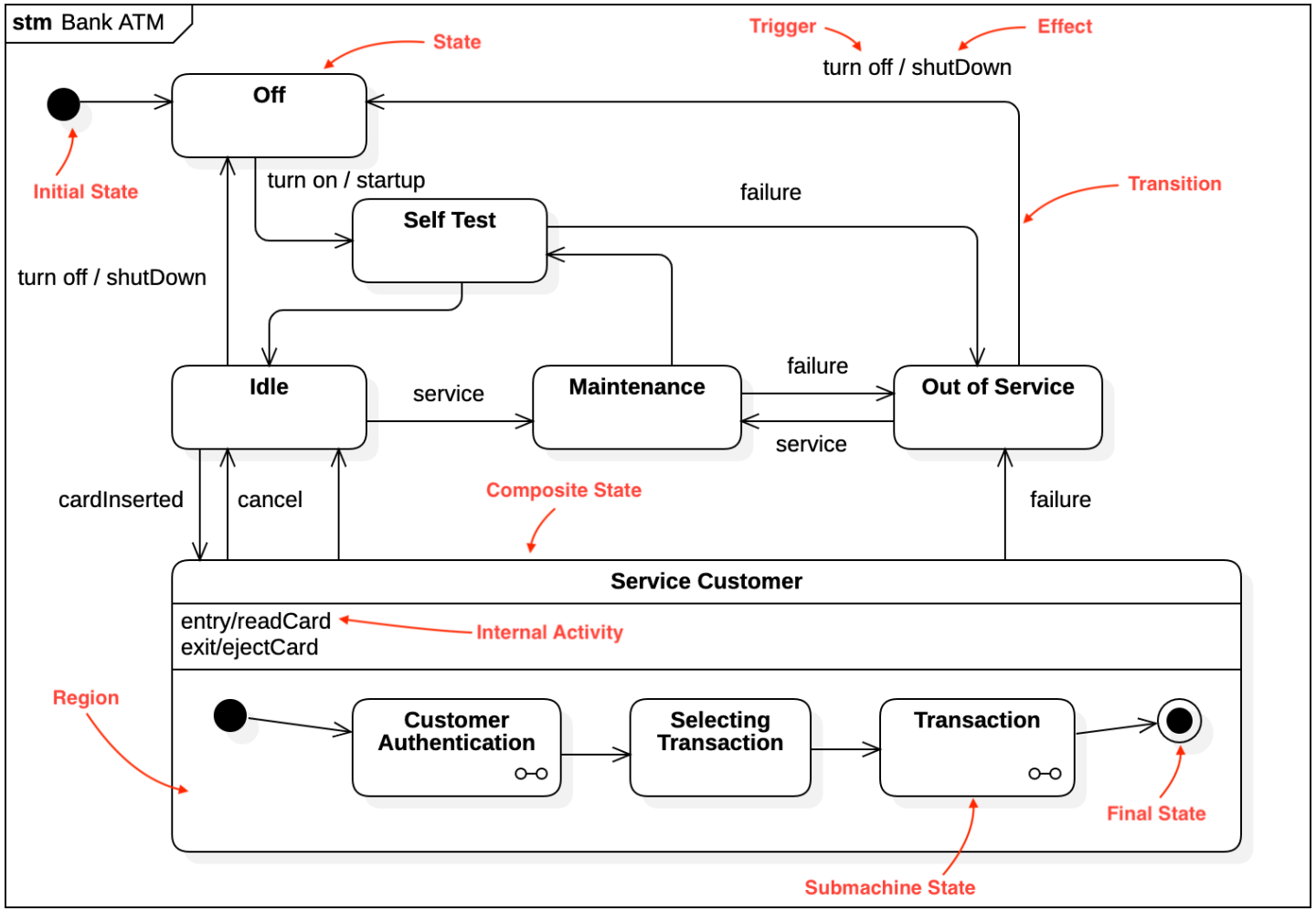
XOR-Zustände mit History-State



UML Zustandsdiagramm

XOR-Zustände



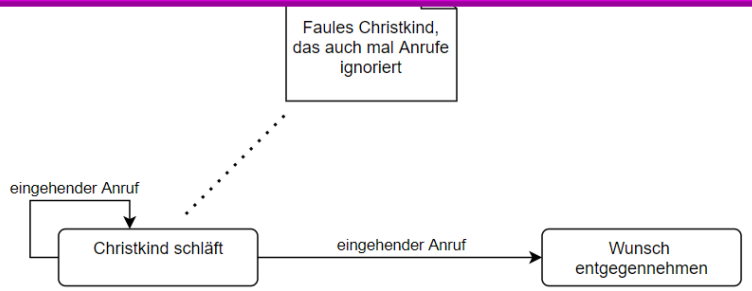


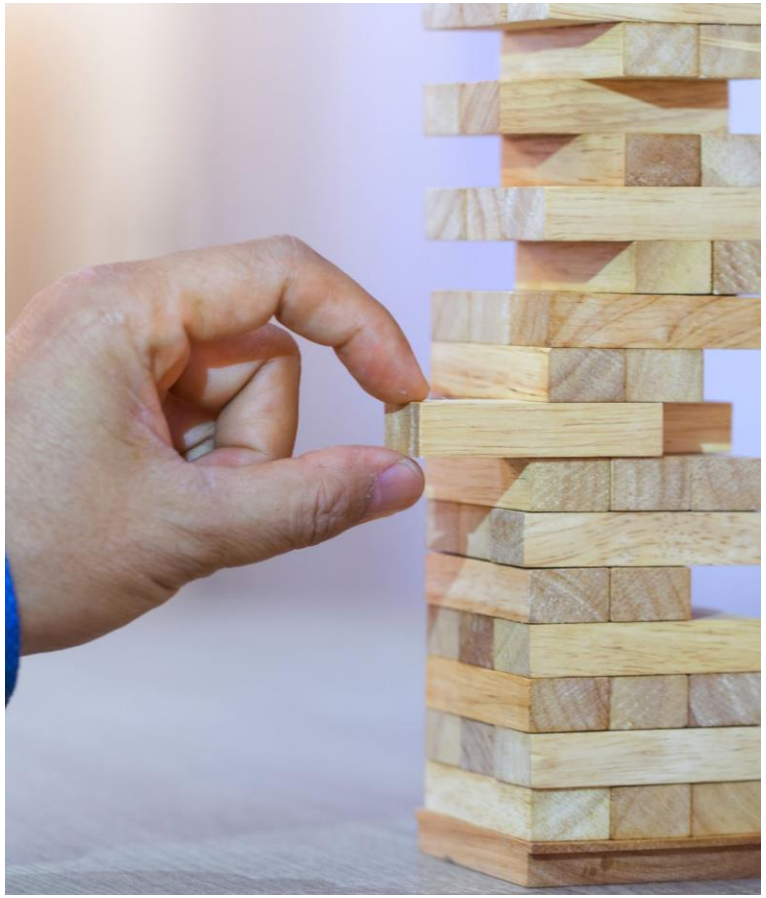
Quelle: StarUML

Konflikte

- Ein Konflikt zwischen Transitionen tritt auf, wenn mehrere Transitionen
  - denselben Quellzustand haben
  - dasselbe
  - nicht dur

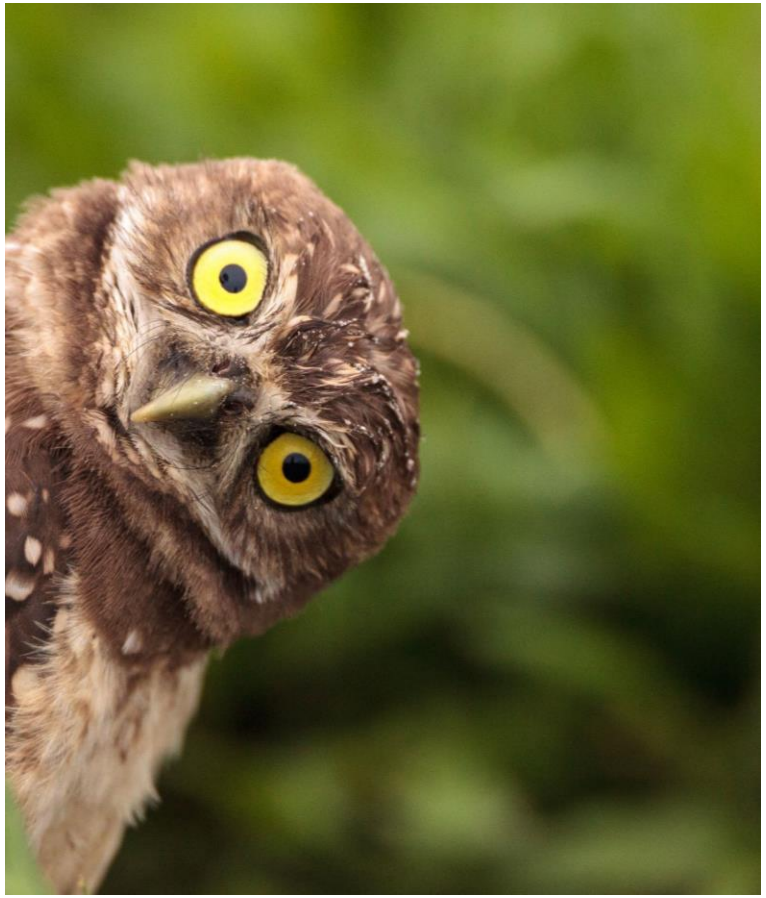
Konsequenz: **Nichtdeterminismus** (eine von beiden Transitionen kann zufällig ausgeführt werden)





Unterschiede Statecharts / Aktivitätsdiagramme

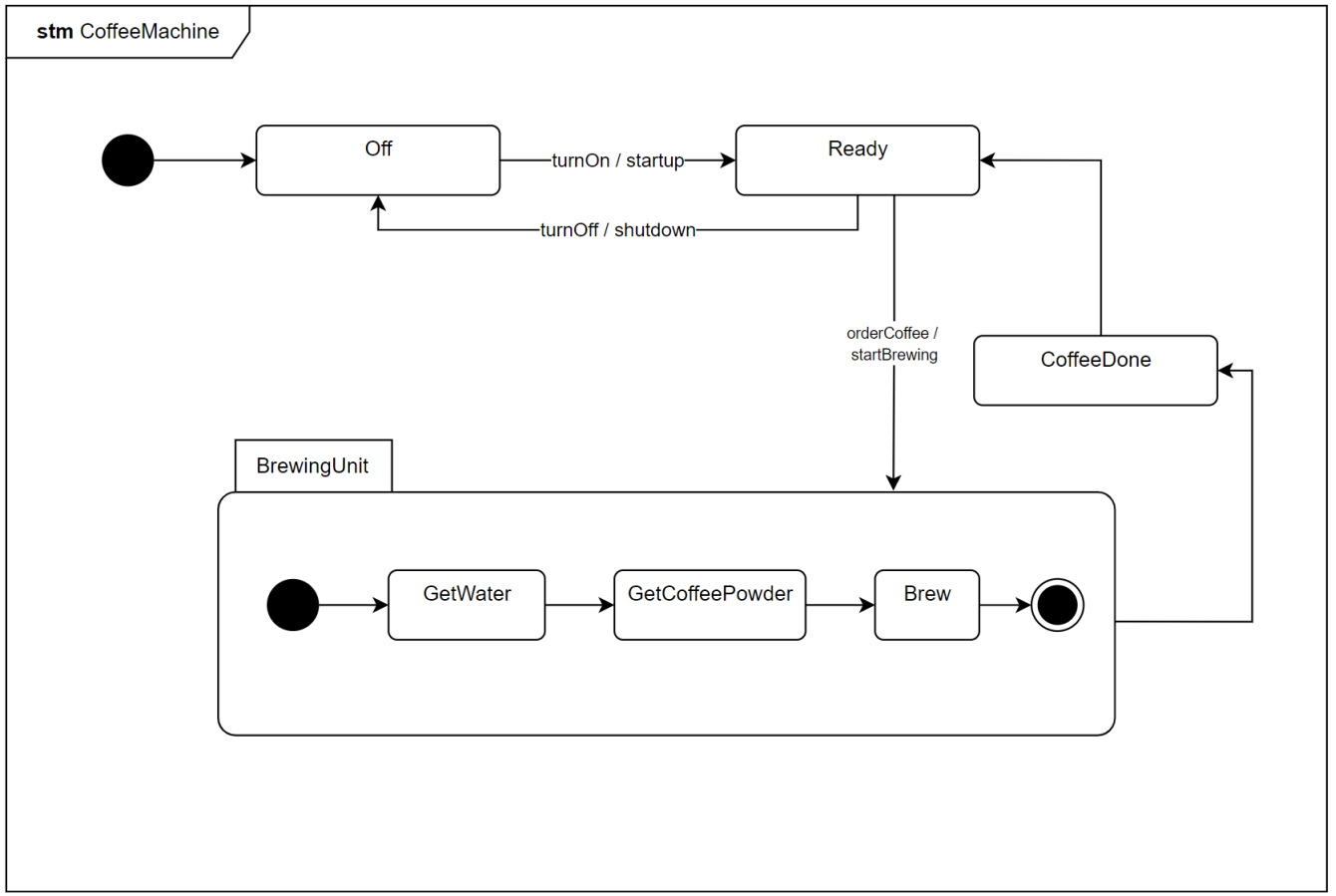
- Sind Statecharts und Aktivitätsdiagramme nicht das gleiche?
- **Aktivitätsdiagramme**
  - Beschreibung eines ganzen Ablaufs, unabhängig vom Initiator
  - Knoten beschreiben Tätigkeiten
  - Übergang, wenn vorherige Aktion abgeschlossen ist
- **Statecharts**
  - Einem bestimmten Interface zugeordnet (Klasse/Komponente)
  - Knoten beschreiben Zustände
  - Transitionen enthalten Ereignisse, die beschreiben, was passiert
    - Ursprung der Ereignisse meist außerhalb der Komponente



Übung

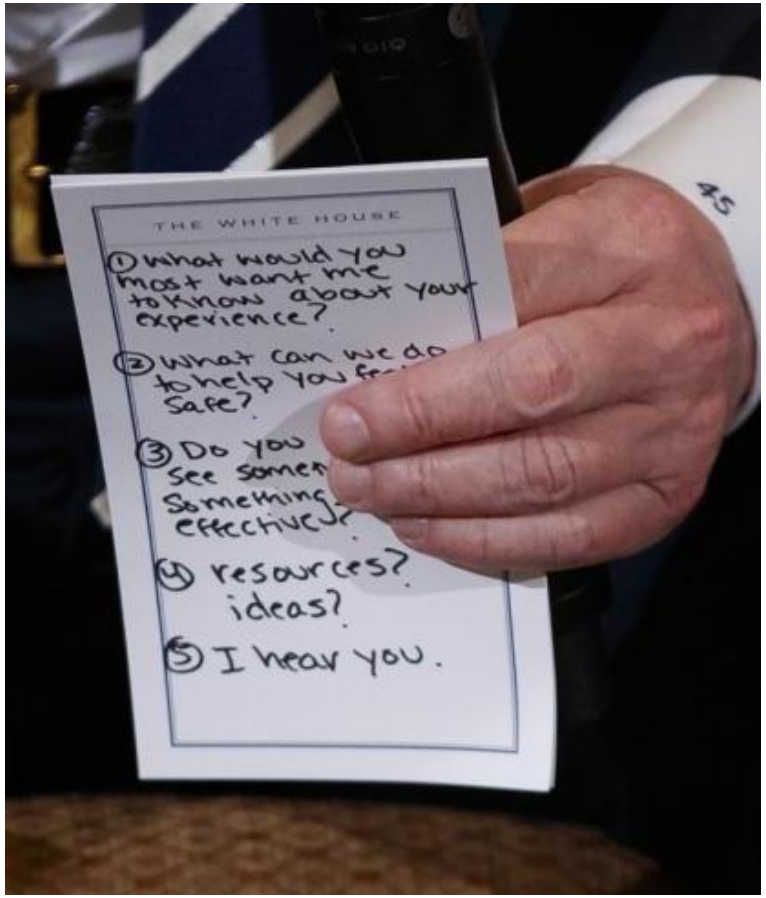
- Der Arbeitsablauf der Kaffeemaschine soll nun als UML Zustandsdiagramm dargestellt werden.
- Die Maschine hat die Zustände Off, Ready, Brewing, CoffeeDone
- Der Brühvorgang ist eine gesonderte Aktivität und beinhaltet die Aktionen GetWater, GetCoffeePowder und Brew.

30 Minuten



Cheat Sheet

- Cheat Sheets sind ein gängiges Mittel, viele und komplexe Inhalte zusammenzufassen
- Erstellt für die behandelten Themen ein Cheat Sheet (kann natürlich auch aus mehr als einer Seite bestehen)
- Am Ende schauen wir uns gemeinsam die Cheat Sheets von euch an



Quiz

- Zum Abschluss ein kleines Quiz für die Festigung des Wissens
- Es können *keine, eine oder mehrere* Antworten richtig sein

Quiz

Frage 1

Welcher Zustand ist hier zu sehen?

A Flow Final Node

B Activity Final Node

C Composition Final Node

Quiz

Frage 2

Welcher Zustand ist hier zu sehen?

A Initialstate

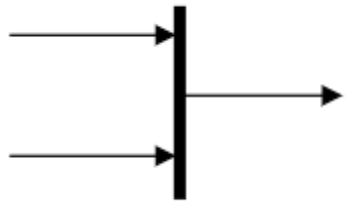
B Flow Final Node

C Pseudostatus

Quiz

Frage 3

Welche der folgenden Aussagen sind richtig?



A Keiner der Aktivitätsflüsse muss für die Fortsetzung verfügbar sein.

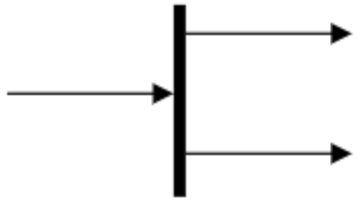
B Beide Aktivitätsflüsse müssen für die Fortsetzung verfügbar sein.

C Es handelt sich um das Verzweigungssymbol.

Quiz

Frage 4

Welche der folgenden Aussagen sind richtig?



A Beide Aktivitätsflüsse laufen nach der Spaltung gleichzeitig ab

B Beide Aktivitätsflüsse laufen nach der Spaltung nicht gleichzeitig ab

C Es handelt sich um das Verzweigungssymbol.

Quiz

Frage 5

Welche der folgenden Aussagen sind richtig?

A Eine synchrone Nachricht erzwingt eine Antwort des Empfängers.

B Eine synchrone Nachricht erzwingt keine Antwort des Empfängers.

C Der Pfeil der Antwort besteht aus gestrichelten Linien.

Quiz

Multiplizität	Bedeutung
1	genau einer
0..1	keiner oder einer
1..5	einer bis fünf
*	keiner, einer oder mehrere
0..*	keiner, einer oder mehrere
1..*	mindestens einer

A *; 1..*; 0..* sind Kardinalitäten mit der Bedeutung „keiner, einer oder mehrere“

B Die Kardinalität 1 bedeutet mindestens einer

C Die Kardinalität 1..3 bedeutet maximal 3 und minimal 1

Quiz

Frage 7

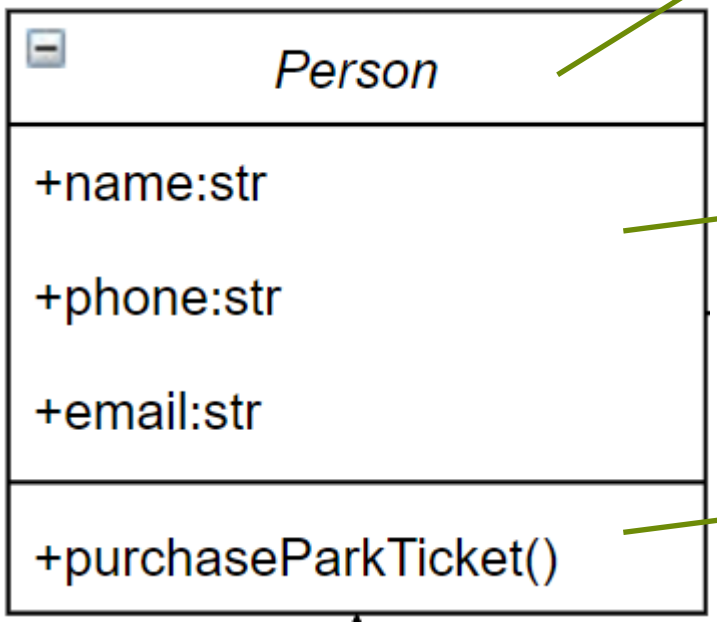
Welche der folgenden Aussagen sind richtig?

A Eine Klasse besteht aus Klassenname, Attribute und Methoden.

B Der Zugriffsmodifikator + bedeutet, dass ein unbeschränkter Zugriff existiert.

C Der Zugriffsmodifikator # bedeutet, dass nur die Klasse selbst auf Attribute/Methoden Zugriff hat.

Klassen



Klassenname

Attribute der Klasse

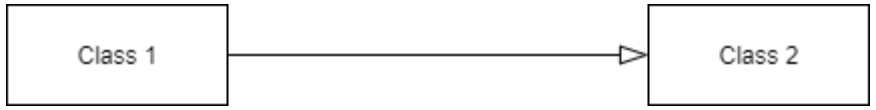
Methoden der Klasse

Was bedeutet das „+“?

Quiz

Frage 8

Welche der folgenden Aussagen sind richtig?

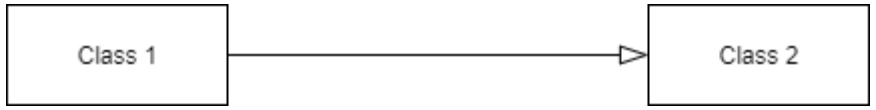


A Class 2 erbt die Attribute und Methoden von Class 1.

B Class 1 erbt die Attribute und Methoden von Class 2.

C Die abgeleitete Klasse kann um neue Attribute und Methoden erweitert werden.

Interaktionen - Vererbung



Quiz

Frage 9

Welche der folgenden Aussagen sind richtig?

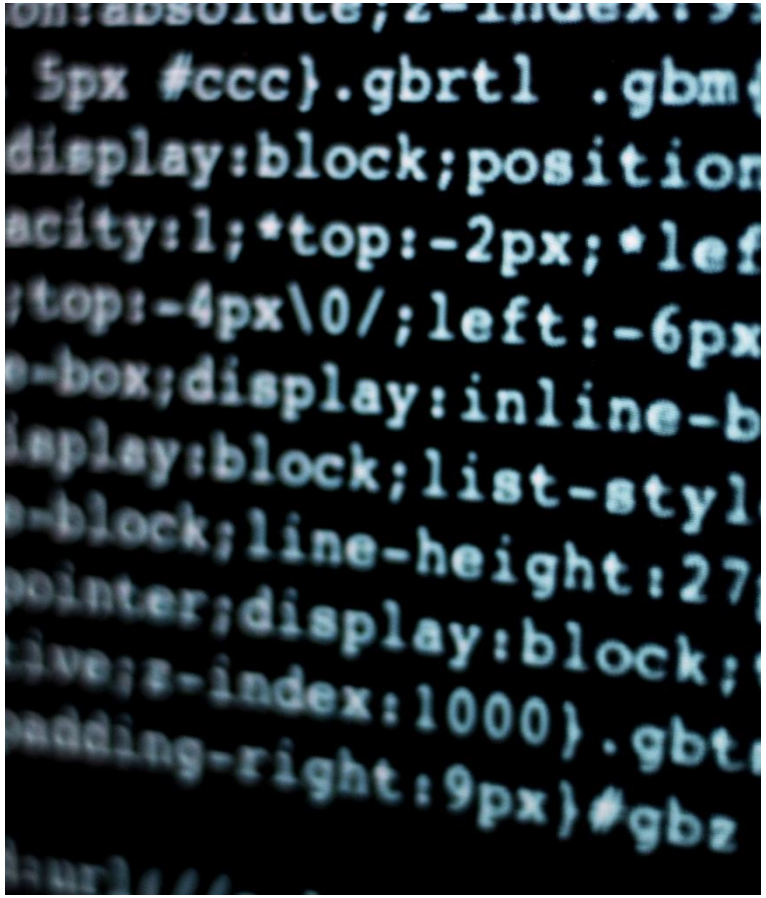
A Pseudocode ist von einer bestimmten Programmiersprache abhängig.

B Die Symbole eines Programmablaufplans sind nach DIN 66 001 genormt.

C Bei der unstrukturierten Programmierung wird das Gesamtproblem in immer kleinere Teilprobleme zerlegt.

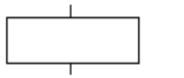
Was ist das?

- „Programmcode, der nicht zur maschinellen Interpretation, sondern lediglich zur Veranschaulichung eines Paradigmas oder Algorithmus dient“
- **Unabhängige Beschreibung eines Programmablaufs**
- Formaler und weniger missverständlich als Beschreibung in natürlicher Sprache
- Oftmals Zwischenschritt von Programmablaufplan/Struktogramm zu Quellcode

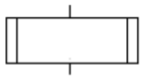


Programmablaufplan

Symbole (nach DIN 66 001)

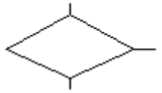


Operation/Verarbeitung allgemein (process)



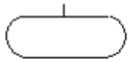
Unterprogrammaufruf (predefined process); Hinweis auf Dokumentation an anderer Stelle in Form von eindeutiger Innenbeschriftung

Ein-/Ausgabe; auch: Daten, allgemein (data)

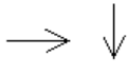


Verzweigung (decision)

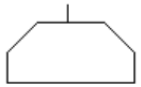
Übergang, Verbindungsstelle (connector)



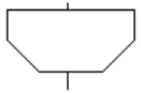
Grenzstelle (terminator)



Ablauflinien (line)

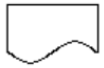


Schleifenbegrenzung für zählergesteuerte Wiederholungen (loop limit)
Anfang



Ende

Maschinell zu verarbeitende Daten (data to be processed by machine)



Daten auf Schriftstück (data on document)

Daten auf Speicher auch mit direktem Zugriff (data on direct access storage)

Was ist das?

- Auch bekannt als „*Nassi-Shneiderman-Diagramm*“
- In DIN 66 261 genormt
- Darstellung von Programmentwürfen im Rahmen der Methode der **strukturierten** Programmierung
  - Gesamtproblem in immer kleinere Teilprobleme zerlegt
  - … bis nur noch elementare Sequenzen und Kontrollstrukturen übrig bleiben

Quiz

Frage 10

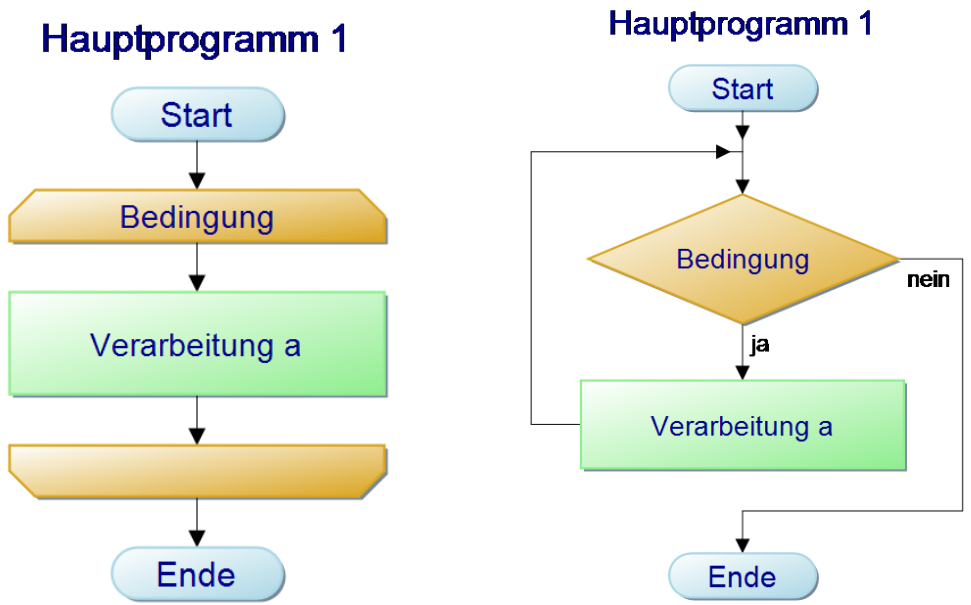
Welche der folgenden Aussagen sind richtig?

A Bei einer fußgesteuerten Schleife wird der Anweisungsblock mindestens einmal ausgeführt.

B Bei einer kopfgesteuerten Schleife wird der Anweisungsblock maximal einmal ausgeführt.

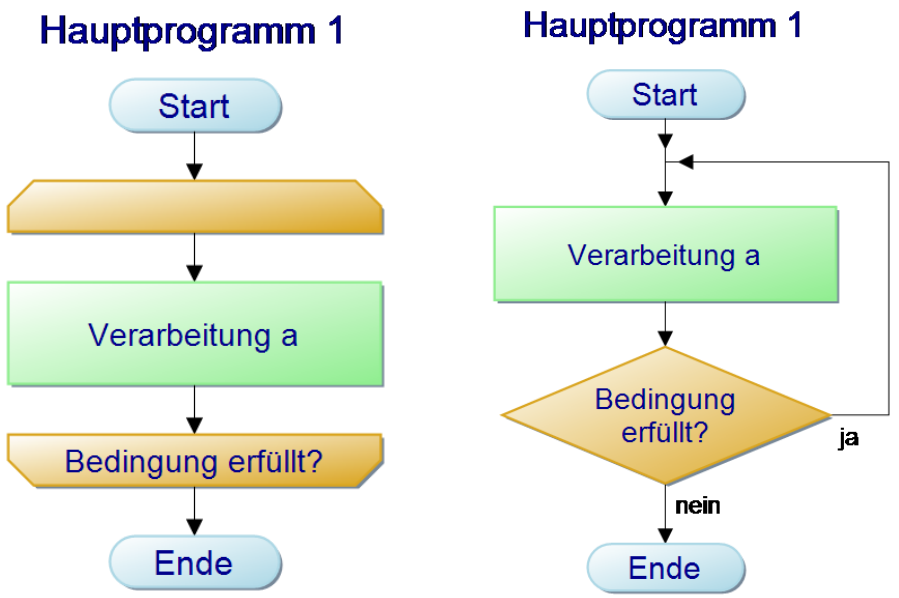
C Am Anfang eines Entwicklungsprozesses sind Anforderungen und Komplexitäten recht unklar.

Wiederholung (kopfgesteuerte Schleife)



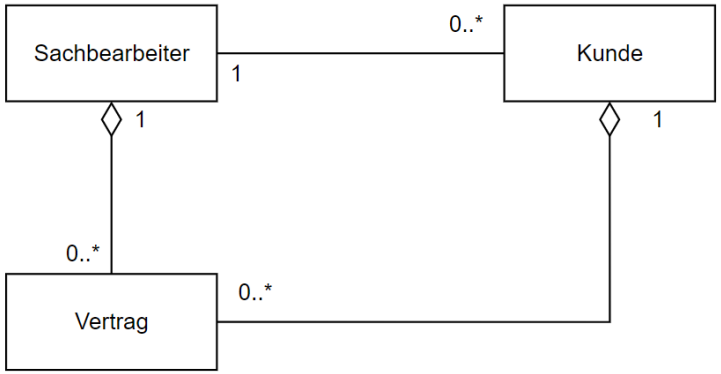
Programmablaufplan

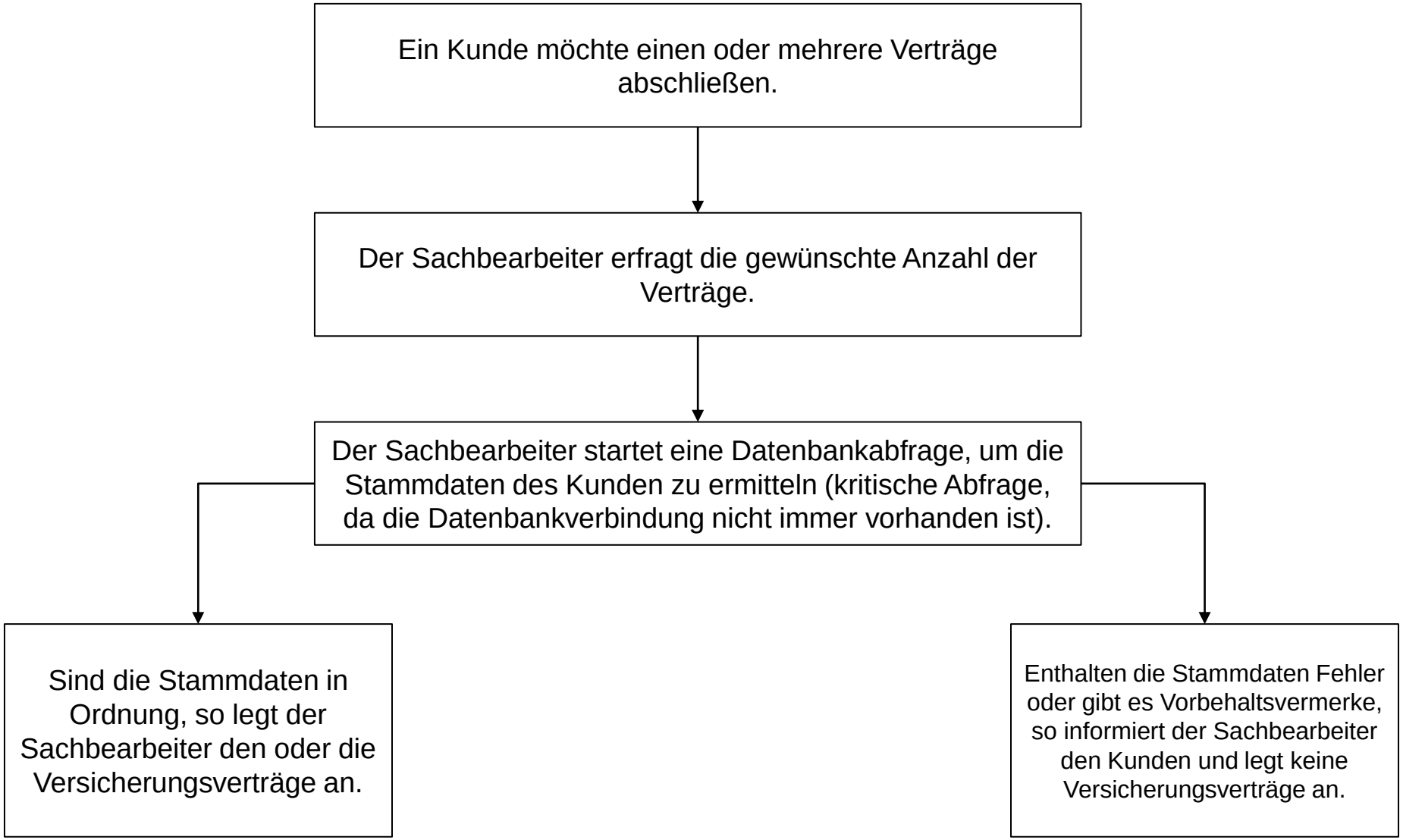
Wiederholung (fußgesteuerte Schleife)

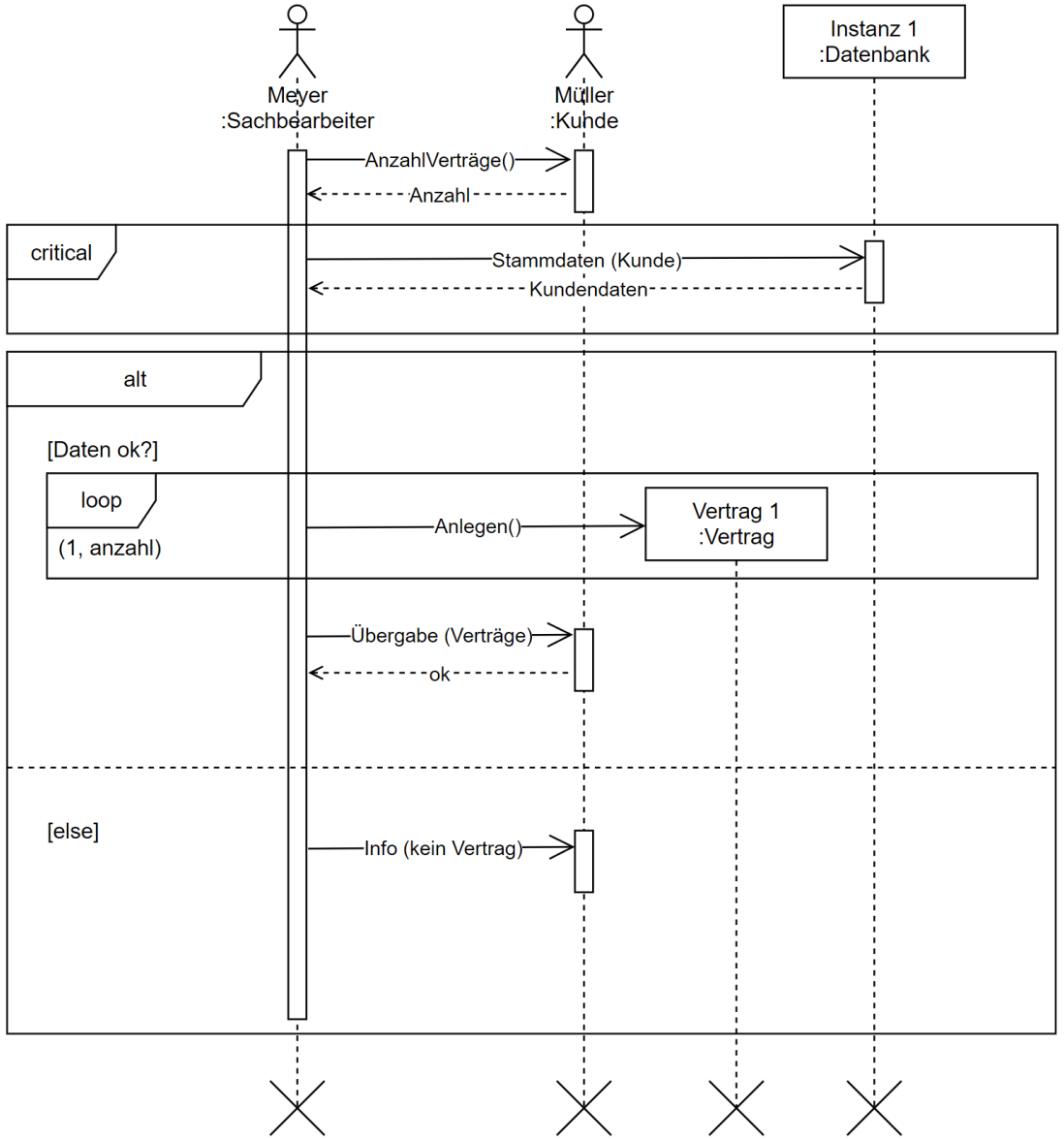


Aufgabe 1

- IT Solutions wird beauftragt, die Softwareentwicklung eines Versicherungsunternehmens zu begleiten. Dazu soll eine objektorientierte Analyse durchgeführt werden. Der Kunde hat bereits ein Klassendiagramm entwickelt und schildert die Ausgangssituation:
  - Für die Versicherung soll ein neues Softwaresystem entwickelt werden. Der Zusammenhang zwischen Kunden, Sachbearbeitern und Versicherungsverträgen ist bereits in einem Klassendiagramm erfasst worden. Entwickle ein Sequenzdiagramm.

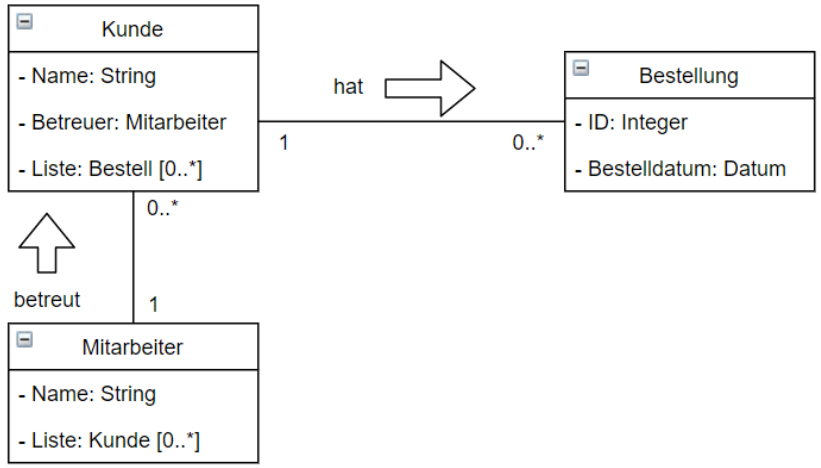






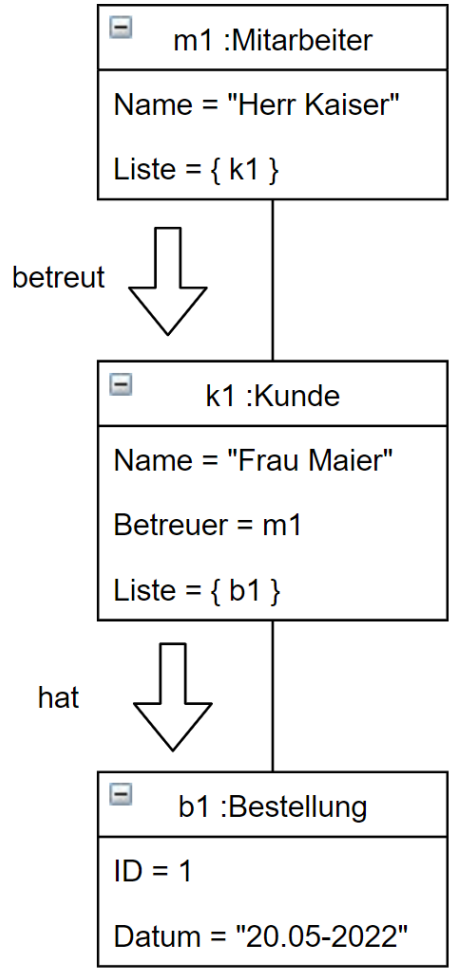
Aufgabe 2

- Informiere dich im ersten Schritt über UML-Objektdiagramme. Was unterscheidet sie von UML-Klassendiagrammen?
- Entwickle nun aus dem nebenstehenden Klassendiagramm ein Objektdiagramm.
  - Folgende Objekte sollen instanziiert werden:
    - Objekt Maier von Klasse Kunde
    - Objekt Kaiser von Klasse Mitarbeiter
    - Objekt B1 von Klasse Bestellung
  - Der Mitarbeiter Kaiser betreut den Kunden Maier und die Bestellung B1 (ID = 1 und Datum = „20.05.2022“) ist ebenfalls dem Kunden Maier zugeordnet.



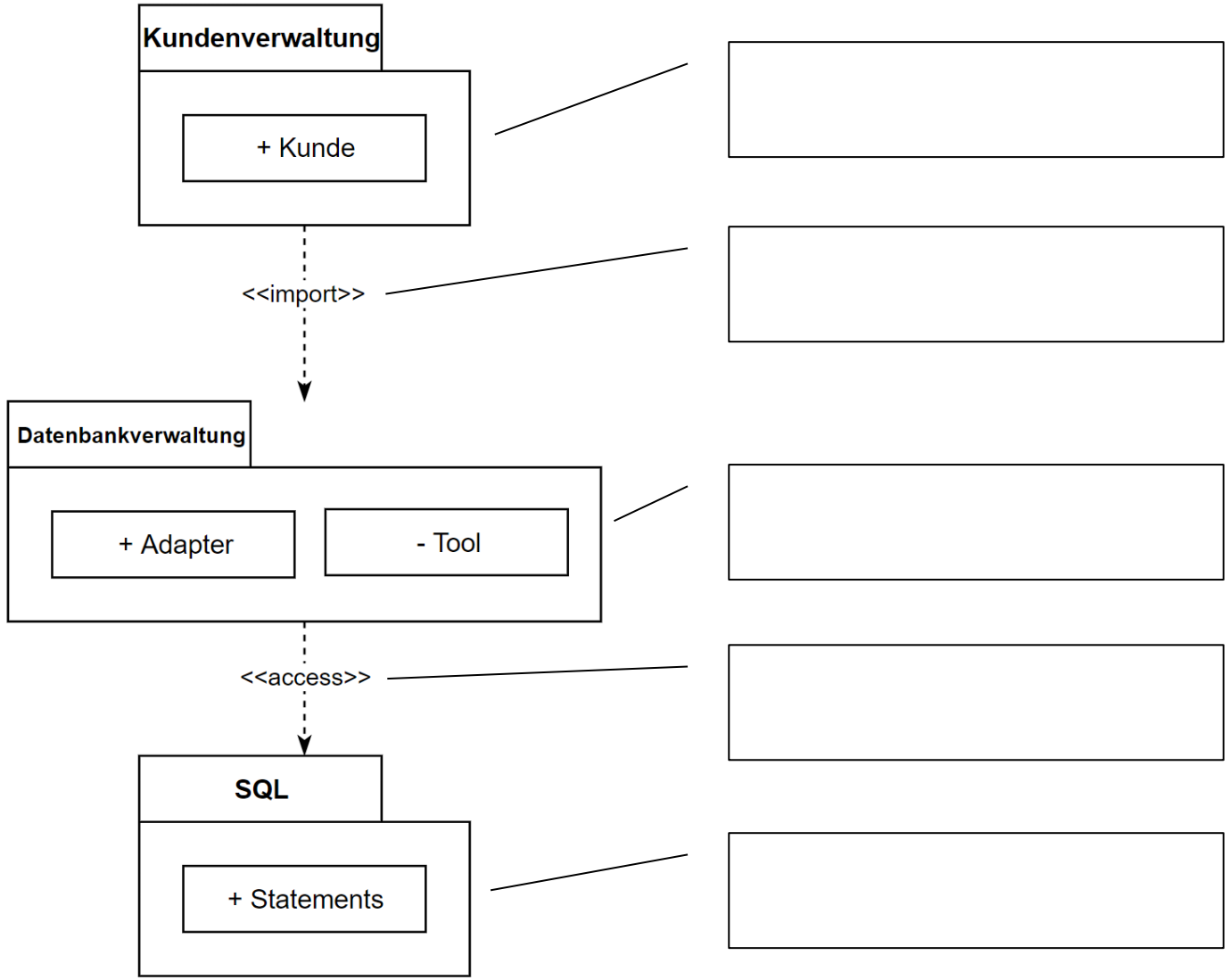
Aufgabe 2 - Lösungsvorschlag

- UML-Objektdiagramm steht für eine konkrete Instanz eines Klassendiagramms zu einem bestimmten Zeitpunkt



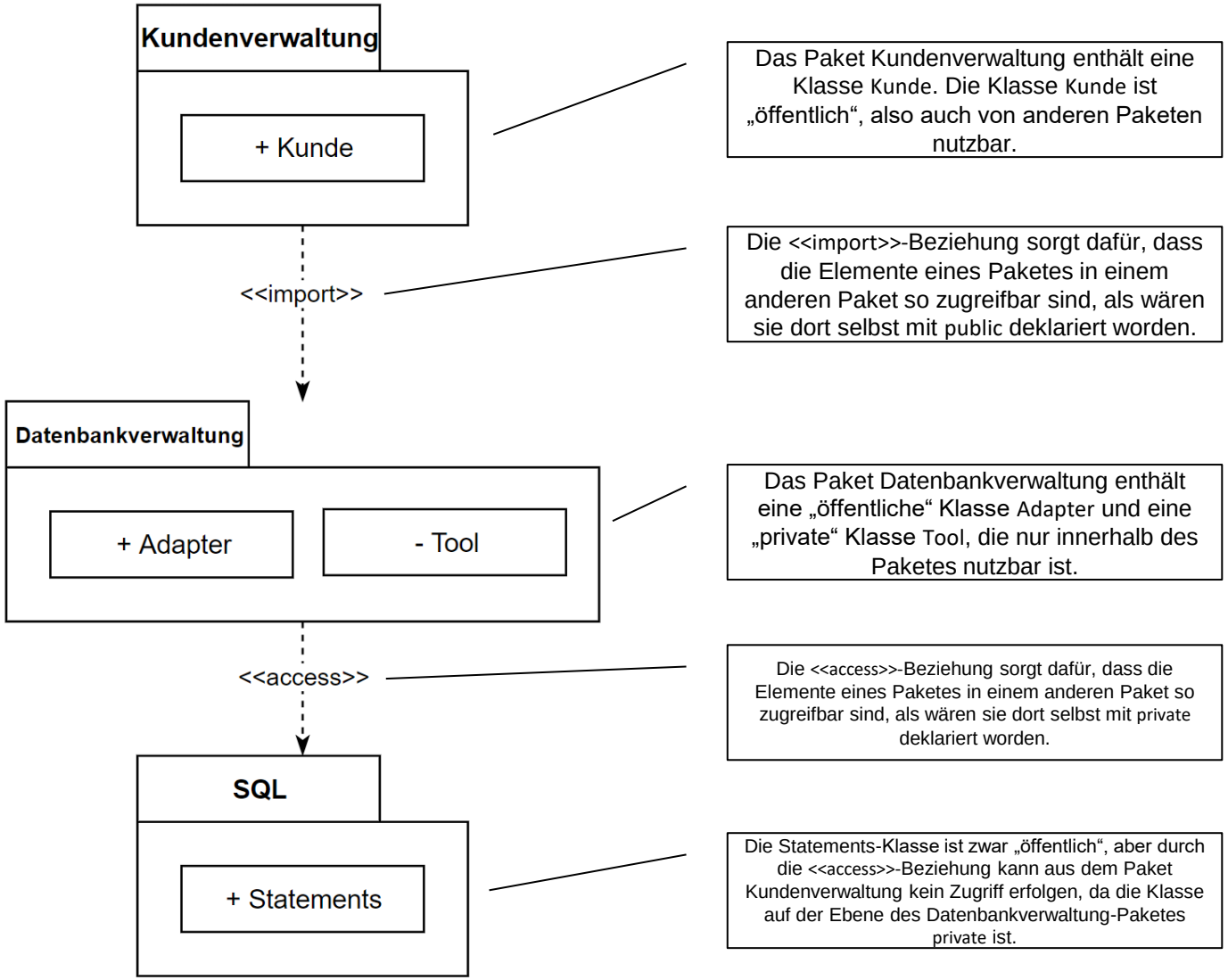
Aufgabe 3

- Für das erwähnte zu entwickelnde Softwaresystem wurde in der objektorientierten Analyse ein Paketdiagramm entwickelt. Beschreibe den Aufbau der Pakete und die Beziehungen zwischen den Pakten sowie deren Auswirkungen.



Aufgabe 3 - Lösungsvorschlag

- Für das erwähnte zu entwickelnde Softwaresystem wurde in der objektorientierten Analyse ein Paketdiagramm entwickelt. Beschreibe den Aufbau der Pakete und die Beziehungen zwischen den Pakten sowie deren Auswirkungen.

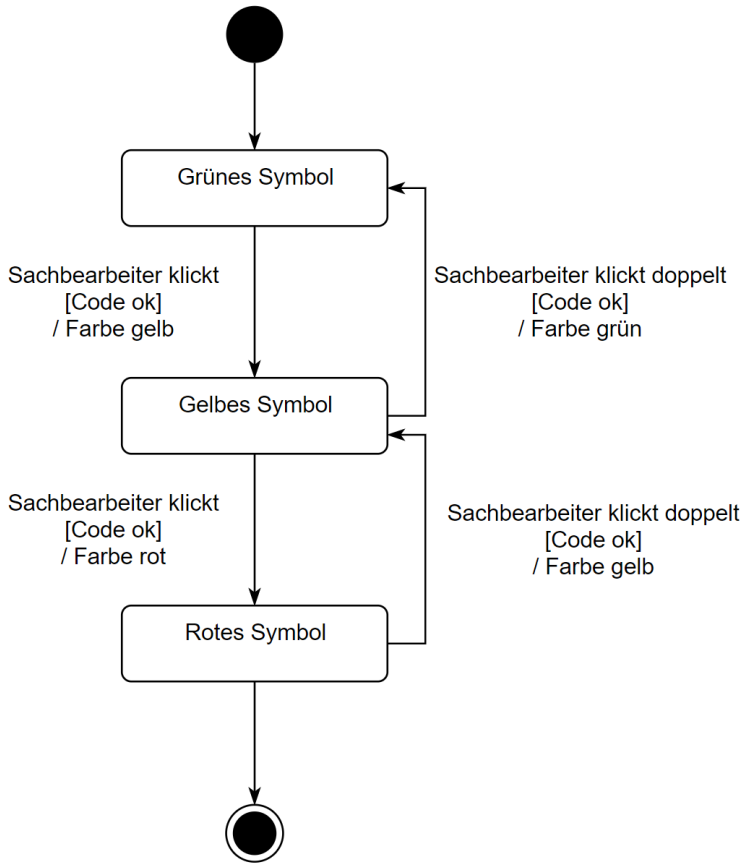


Aufgabe 4

- Laut Anforderungsanalyse soll das zu entwickelnde Softwaresystem eine Zustandsanzeige in Form einer Ampel enthalten. Jedem Kunden ist initial die Farbe grün zugeordnet. Stellt ein Sachbearbeiter fest, dass ein Kunde unzuverlässig ist, kann er auf dieses Symbol klicken muss einen Sicherheitscode eingeben und ändert damit die Farbe des Symbols auf gelb. Selbige Prozedur kann auch zu einem roten Symbol führen. Für alle Sachbearbeiter ist damit ersichtlich, ob ein Kunde problematisch ist. Wird ein Kunde wieder zuverlässiger, kann der Prozess analog rückgängig gemacht werden. Hierfür muss der Sachbearbeiter jedoch einen Doppelklick auf das Symbol tätigen.
- Ihr Auftrag ist es, zu dieser Problematik ein Zustandsdiagramm zu entwickeln.

Aufgabe 4 - Lösungsvorschlag

- Laut Anforderungsanalyse soll das zu entwickelnde Softwaresystem eine Zustandsanzeige in Form einer Ampel enthalten. Jedem Kunden ist initial die Farbe grün zugeordnet. Stellt ein Sachbearbeiter fest, dass ein Kunde unzuverlässig ist, kann er auf dieses Symbol klicken muss einen Sicherheitscode eingeben und ändert damit die Farbe des Symbols auf gelb. Selbige Prozedur kann auch zu einem roten Symbol führen. Für alle Sachbearbeiter ist damit ersichtlich, ob ein Kunde problematisch ist. Wird ein Kunde wieder zuverlässiger, kann der Prozess analog rückgängig gemacht werden. Hierfür muss der Sachbearbeiter jedoch einen Doppelklick auf das Symbol tätigen.
- Ihr Auftrag ist es, zu dieser Problematik ein Zustandsdiagramm zu entwickeln.



Aufgabe 5

- Verschaffe dir wiederholend einen Überblick über Struktogramm und PAP. Stelle die Vor- und Nachteile der beiden Diagramme dar.
- In Ihrer Firma ist ein erfahrener Mitarbeiter in den Ruhestand gegangen. Die Entwicklungsabteilung benötigt nun Unterstützung. Im Rahmen der strukturierten Programmierung sollen Diagramme für ein Softwareprojekt entwickelt werden.
- Nachfolgend findet sich die Ablaufbeschreibung für ein zu erstellendes Struktogramm.

Aufgabe 5

- Ein Teilbereich der Software soll die Berechnung von Kundenrabatten durchführen. Dazu muss der Sachbearbeiter einige Eingaben tätigen, welche anschließend den Rabatt bestimmen.
- Erster Teil der Eingabe sind Kundenname und Rechnungsbetrag.
- Danach fragt die Software nach der Bonität des Kunden.
  - Ohne Bonität wird der maximale Rabatt auf 4% begrenzt.
  - Handelt es sich um einen Stammkunden, so wird ein Grundrabatt von 2% gewährt.
  - Anschließend wird ein gestaffelter Rabatt berechnet:

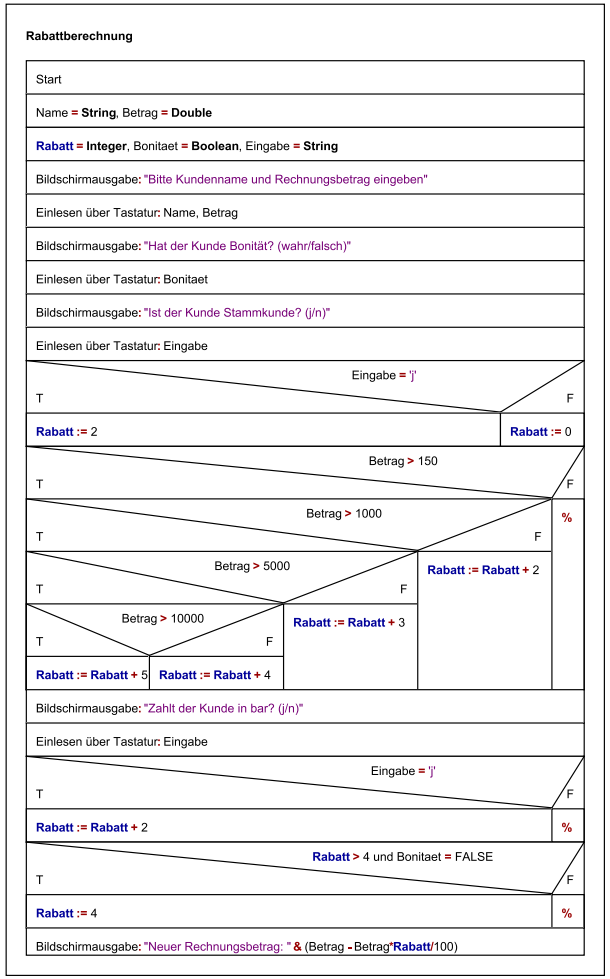
Aufgabe 5

0 < Rechnungsbetrag <= 150	Kein Rabatt
150 < Rechnungsbetrag <= 1000	2% Rabatt
1000 < Rechnungsbetrag <= 5000	3% Rabatt
5000 < Rechnungsbetrag <= 10000	4% Rabatt
10000 < Rechnungsbetrag	5% Rabatt

Aufgabe 5

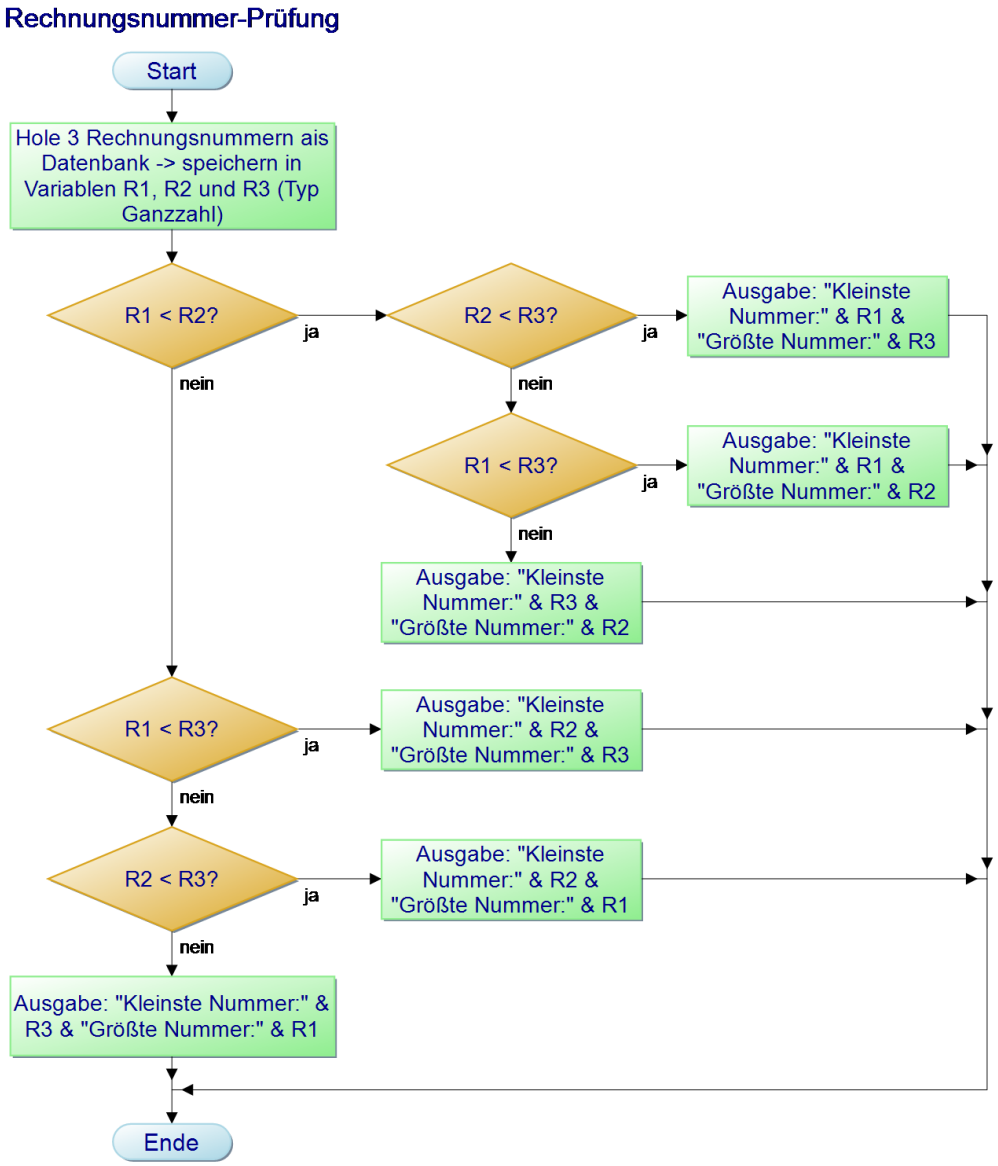
- Bezahlt der Kunde in bar, so wird zusätzlich ein Skonto von 2% gewährt.
- Alle Rabatte summieren sich auf und werden am Ende vom Rechnungsbetrag abgezogen. Das Programm kalkuliert dann den neuen Rechnungsbetrag.

Aufgabe 5 - Lösungsvorschlag

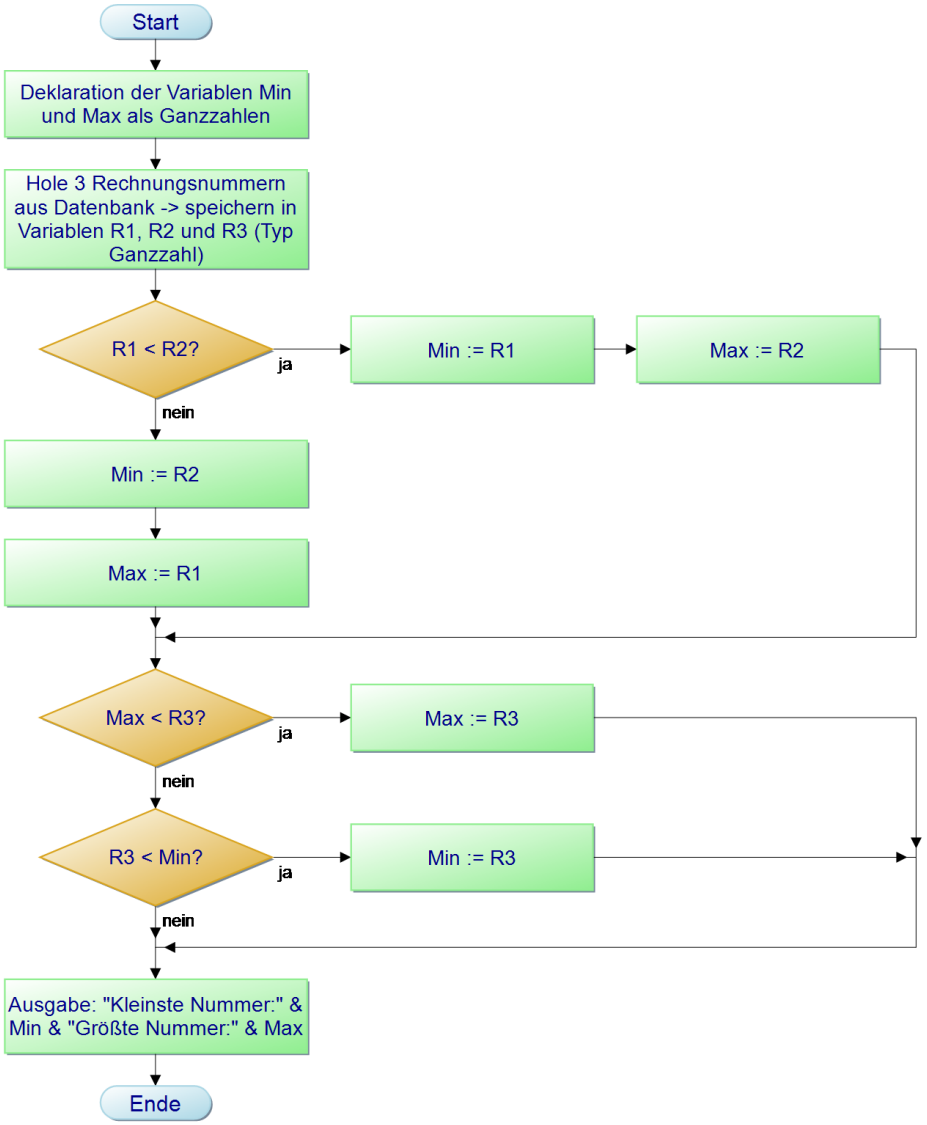


Aufgabe 6

- Ein Kollege von Ihnen hat einen Programmablaufplan zu einem anderen Teilproblem (Feststellung der Reihenfolge von drei Rechnungsnummern) entworfen. Analysieren Sie den PAP und prüfen Sie, ob dieser im Sinne des Refactorings vereinfacht werden kann. Für eine Performanceverbesserung hat der Entwicklungsleiter die Vorgabe gestellt, dass nur drei Verzweigungen benutzt werden dürfen. Weitere Variablen dürfen nach Belieben eingeführt werden.



Rechnungsnummer-Prüfung Refactoring



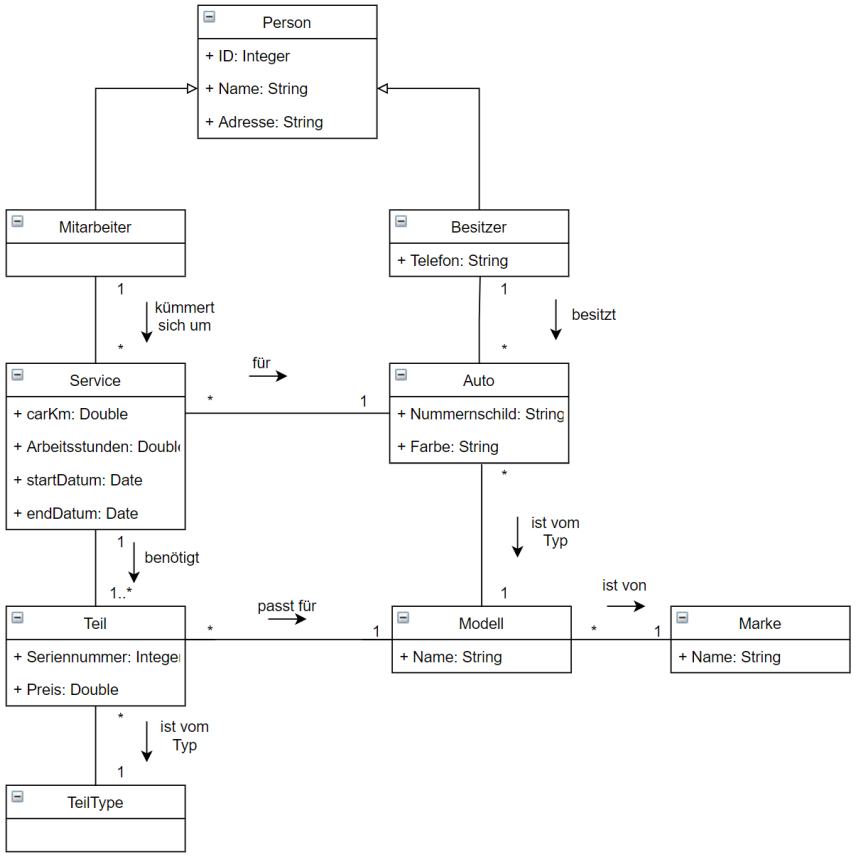
Aufgabe 7

- Eine Autowerkstatt hat Ihnen den Auftrag für die Entwicklung eines neuen Softwaresystems übergeben. Mit dem System sollen Kunden, Lagerbestände und Aufträge verwaltet werden.
  - Es gibt mehrere Mitarbeiter. Jeder von ihnen hat eine eindeutige Identifikationsnummer, einen Namen und eine Adresse.
  - In dieser Werkstatt werden Montagearbeiten durchgeführt, bei denen Teile und Zubehör in ein Fahrzeug eingebaut werden. Für jede dieser Dienstleistungen müssen die folgenden Daten gespeichert werden: An welchem Auto wurde der Service durchgeführt, wie viele Kilometer hatte das Auto zu dem Zeitpunkt, wer war der verantwortliche Mitarbeiter, welche Teile wurden eingebaut, wie viele Arbeitsstunden hat es gedauert und das Aufnahme- und Enddatum.
  - Teile und Zubehör werden nur zusammen mit einem Montageservice verkauft.
  - Jedes Zubehörteil passt nur in bestimmte Fahrzeugmodelle. Daher ist es wichtig, diese Informationen zu speichern.

Aufgabe 7

- Eine Autowerkstatt hat Ihnen den Auftrag für die Entwicklung eines neuen Softwaresystems übergeben. Mit dem System sollen Kunden, Lagerbestände und Aufträge verwaltet werden.
  - Jedes Zubehörteil hat eine Kategorie (Radio, Reifen, ...), eine Seriennummer und einen Preis.
  - Jedes Auto hat ein Nummernschild, eine Marke, ein Modell, eine Farbe und einen Besitzer.
  - Jeder Besitzer hat einen Namen, eine Identifikationsnummer, eine Adresse und eine Telefonnummer.
  - Eine Person kann mehr als ein Auto besitzen, aber ein Auto hat nur einen Besitzer.

Aufgabe 7 - Lösungsvorschlag



Lasten- und Pflichtenheft

- Lastenheft
  - Quasi ein „Anforderungskatalog“ des Auftraggebers (die „Last“)
  - Nicht nur für die Sammlung der technischen Anforderungen, sondern auch für die Auswahl eines geeigneten Auftragnehmers
  - Nach Erteilung des Auftrags ist es Grundlage für die weitere Orientierung im Projekt
  - Zieldefinition, Beschreibung des Ist- und Soll-Zustands, Schnittstellen und Zuständigkeiten, funktionale und technische Anforderungen, Glossar für Klärung von Fachbegriffen

Lasten- und Pflichtenheft

- Pflichtenheft
  - Projektplan vom Auftragnehmer zur Erfüllung des Lastenheft des Auftraggebers
  - Beschreibung der gestellten Anforderungen
  - Wer nimmt an der Entwicklung teil, wer ist für was verantwortlich?
  - Welche Voraussetzungen sind erfüllt?
  - Wie und bis wann soll die Umsetzung stattfinden?
  - Skizzierung in Form von Diagrammen, Use-Cases, UI-Skizzen, …
  - Auftraggeber übergibt bei Einverständnis den Auftrag (**rechtlich bindend**)

Aufgabe 8

- Nicht-funktionale Anforderungen
  - Qualitäten, die Produkt von vergleichbaren Produkten unterscheiden

Name	Schutz gegen Bruteforce
Typ	SICHER
Beschreibung	Die Software blockiert nach drei Fehlversuchen die Anmeldung für 30 Sekunden.
Zugeordnete(r) Use Case(s):	

Aufgabe 8

- Nicht-funktionale Anforderungen
  - Typen
    - *USE* (Benutzerfreundlichkeit) → Software muss … leisten, damit Zielgruppe gerne mit ihr arbeitet
    - *EFFIZIENZ* → Software muss … leisten, um mit Speicher und Laufzeit ökonomisch umzugehen.
    - *PFLEGE* → Software muss … leisten, damit vorhersehbare Änderungen und Erweiterungen leicht möglich sind.
    - *SICHER* → Software muss … leisten, damit die Verfügbarkeit sowie die Vertraulichkeit und Integrität der Daten gewährleistet ist.
    - *LEGAL* → Software muss … leisten, damit die relevanten Standards und Gesetzte berücksichtigt werden.

Aufgabe 8

Pflichtenheft - Aufgabe.pdf

PHOENIX CONTACT
INSPIRING INNOVATIONS