Einführung in die Informatik I
Schnittstellen, Klassen und Algebren

Prof. Bernd Brügge, Ph.D
Technische Universität München

Wintersemester 2000/2001
31. Oktober 2000

Überblick über die heutige Vorlesung

- Rest vom 24. Oktober
- Kummerkasten
  - Skript, …..
  - Wie nehme ich Notizen während der Vorlesung auf?
- Klasse
  - Attribute, Operationen
- Definition Rechenstruktur
- Instanz einer Klasse (Objekt vs. Klasse)
- Definition Instanzendiagram, Klassendiagram
  - Ein System ist eine Menge von kooperierenden Klassen oder Objekten
- Die Vererbungs-Beziehung
  - Beispiele von Vererbungen
- Kombination von Aggregation und Vererbung
- Entwurfsmuster: Kompositionsmuster

Überblick über die Dienstagsvolesung

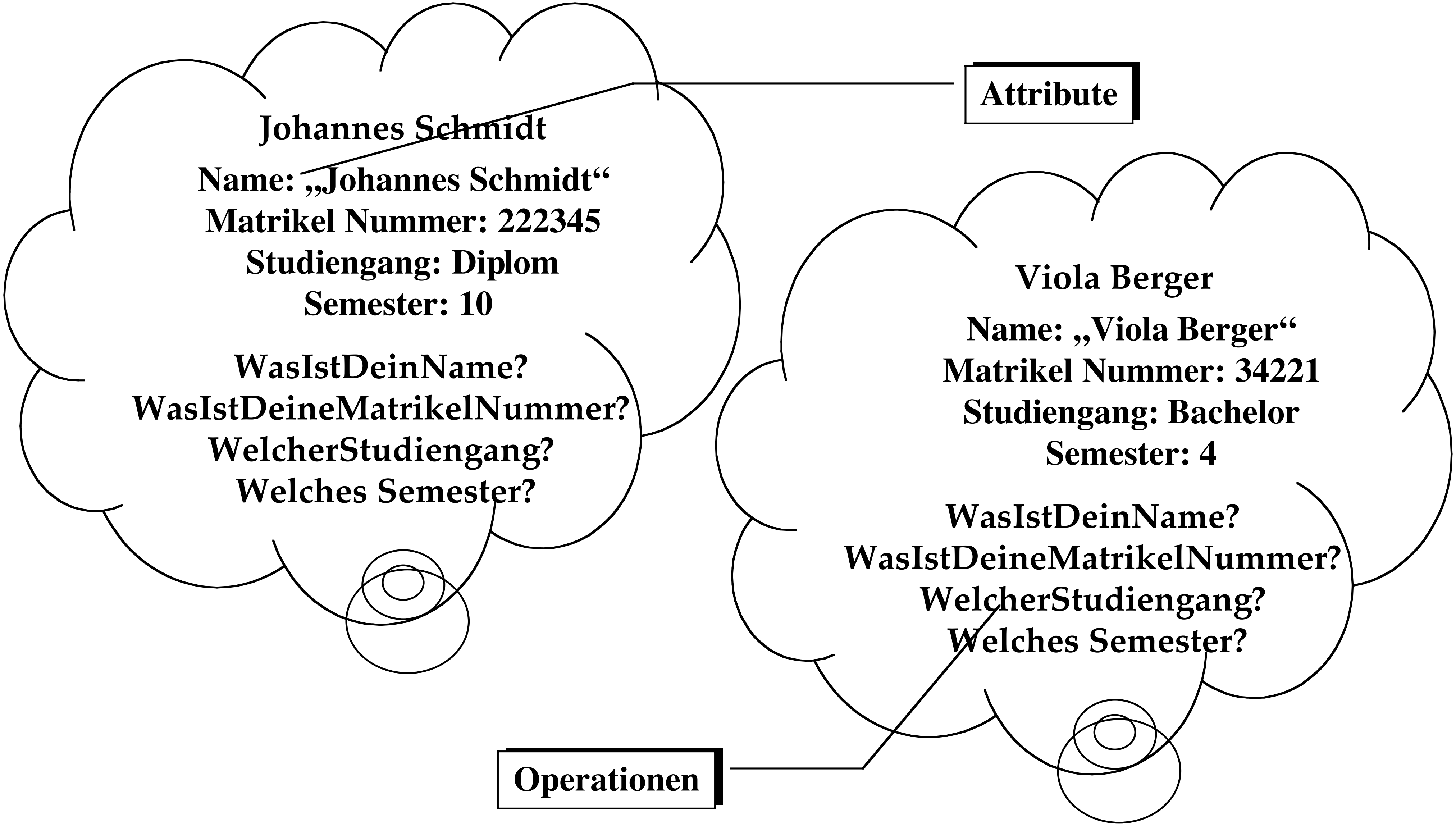
- Signatur
- Signaturgraph
- 3 Beispiele:
  - Keller, Sequenz, Studentenverwaltungssystem
- Abstrakte Algebra und Konkrete Algebra
- Terme
- Programmierschnittstelle (API)
- Vorbereitung:
  - Goos, Band 1, Kapitel über Algebren und Termalgebren
  - Broy. Band 1, Kapitel über Rechenstrukturen

Unser Grundmodell der Modellierung

- Ein System besteht aus Subsystemen, die wieder aus Subsystemen bestehen, und diese dann letztendlich aus Gegenständen.
  – Diese Gegenstände bezeichnen wir auch als Objekte.
- Alle zu verarbeitende Information in einem System ist auf diese Objekte verteilt.
  – Die Verarbeitung von Information geschieht entweder innerhalb der Objekte oder durch Kommunikation von Nachrichten zwischen zwei Objekten.

Objekt

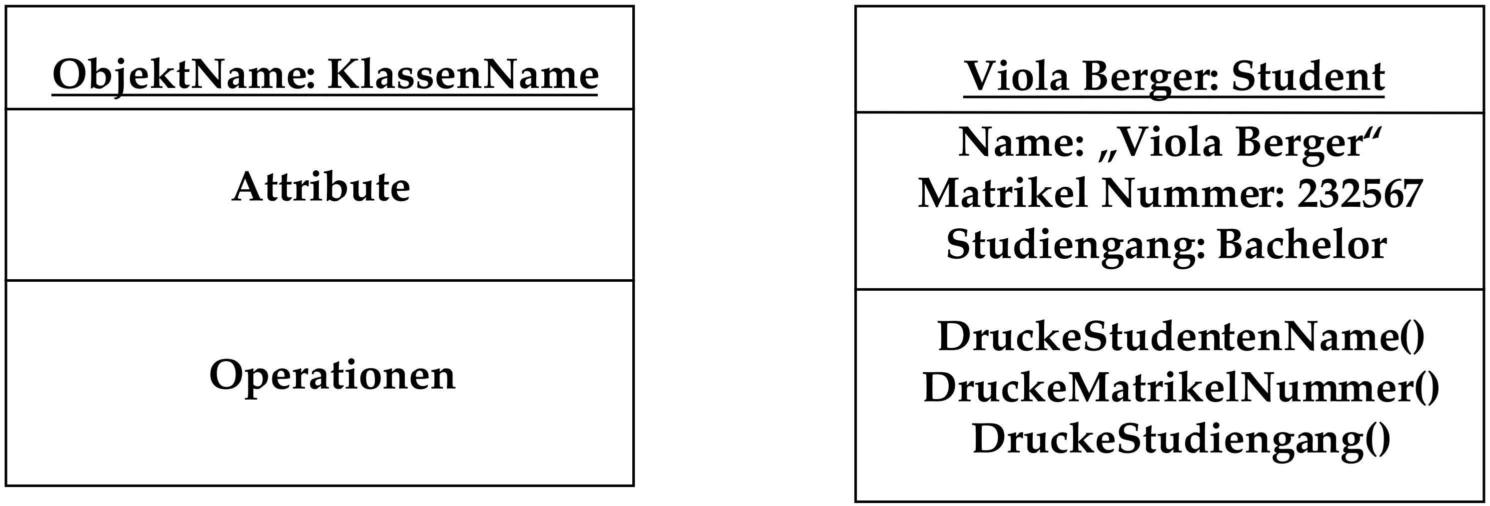
- **Definition Objekt:** Ein Objekt ist ein elementares Teilsystem. Es repräsentiert einen beliebigen Gegenstand in einem System.
- Ein Objekt ist durch seinen Zustand und seine Funktionalität gegeben.
- Zustand und Funktionalität setzen sich im Allgemeinen aus Teilzuständen und einzelnen Operationen zusammen.
  - Wir nennen die Teilzustände auch Attribute
  - Wir nennen die einzelnen Operationen Methoden des Objektes
- Wir nennen die Operationen eines Objektes, die von anderen Objekten aufgerufen werden können, die Schnittstelle des Objektes


Johannes Schmidt
Name: „Johannes Schmidt“
Matrikel Nummer: 222345
Studiengang: Diplom
Semester: 10
WasIstDeinName?
WasIstDeineMatrikelNummer?
WelcherStudiengang?
Welches Semester?
Attribute
Viola Berger
Name: „Viola Berger“
Matrikel Nummer: 34221
Studiengang: Bachelor
Semester: 4
WasIstDeinName?
WasIstDeineMatrikelNummer?
WelcherStudiengang?
Welches Semester?
Operationen

Attribute, Operationen, Merkmale

- Ein Objekt besitzt Attribute und Operationen
  - **Definition Attribut:** Messbare, durch Werte erfassbare Eigenschaften des Objektes.
  - **Definition Operation:** Die Tätigkeiten, die ein Objekt ausführen kann, um Berechnungen durchzuführen, Ereignisse auszulösen, sowie Botschaften zu übermitteln.
- **Definition Merkmal:** Die zu einem Objekt gehörigen Attribute und Operationen.
- **Definition Schnittstelle:** Die Menge der Operationen eines Objektes, die von anderen Objekten aufgerufen werden können.

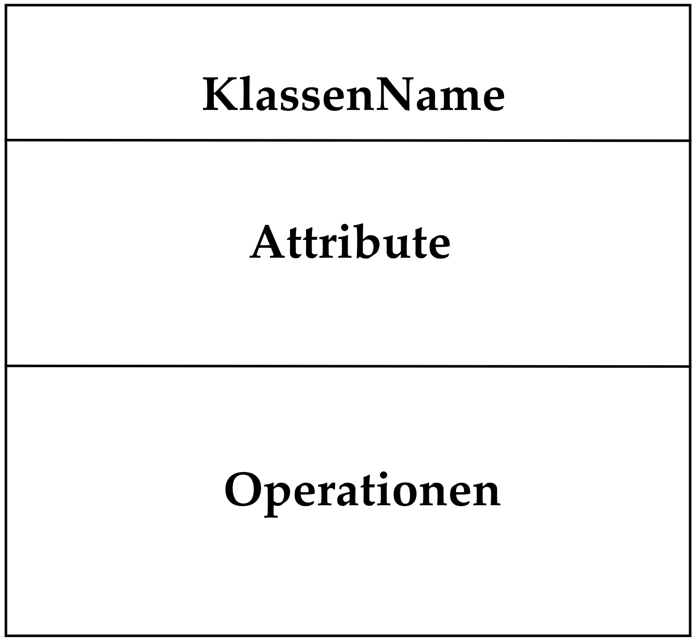
Graphische Darstellung: Objekt



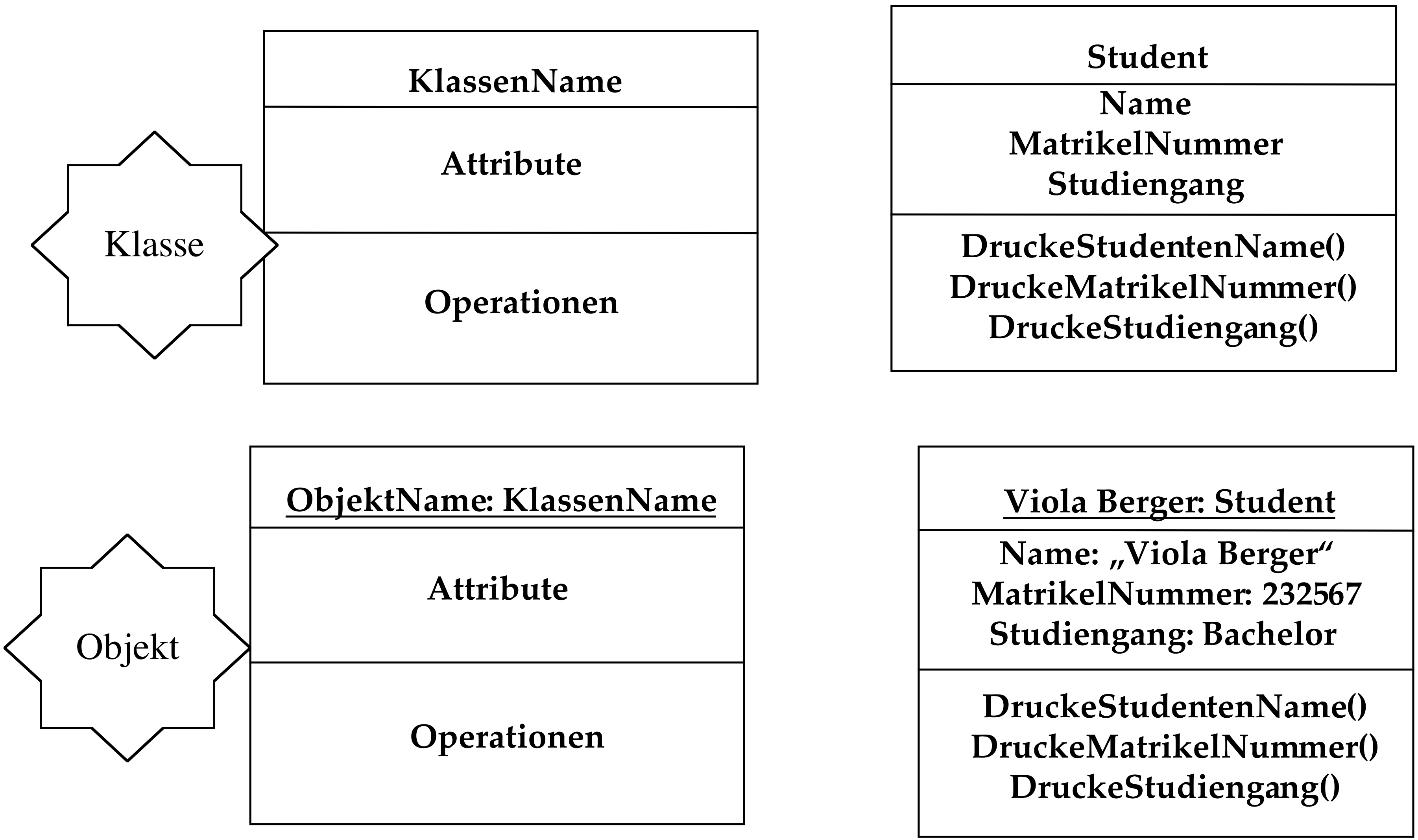
Klasse

- Jedes Objekt gehört zu einer Menge von Objekten mit gleichen Merkmalen. Wir nennen diese Menge auch Klasse.
- **Definition Klasse:**
  - Die Menge aller Objekte mit gleichen Merkmalen, d.h. mit gleichen Attributen und Operationen.
- **Definition Instanz:**
  - Ein Objekt ist eine Instanz einer Klasse K, wenn es Element der Menge aller Objekte der Klasse K ist.

Graphische Darstellung von Klassen



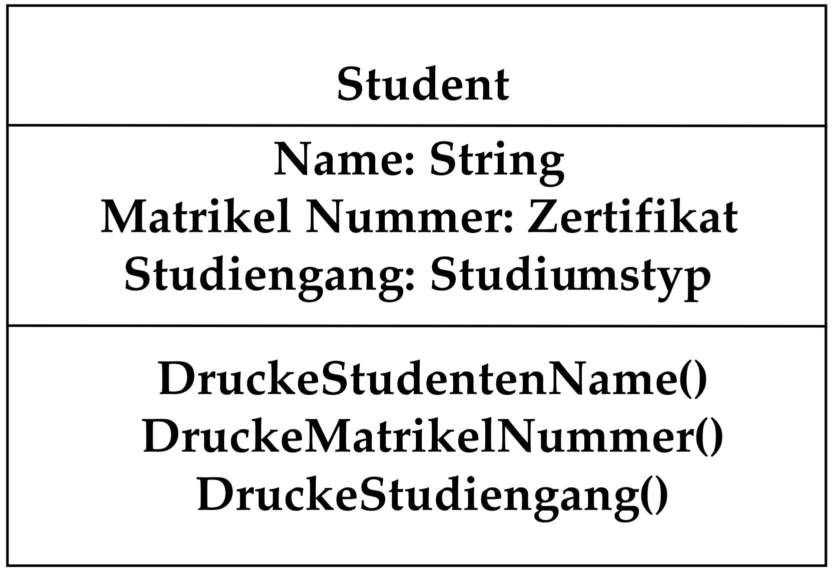
Graphische Darstellung: Objekt vs. Klasse

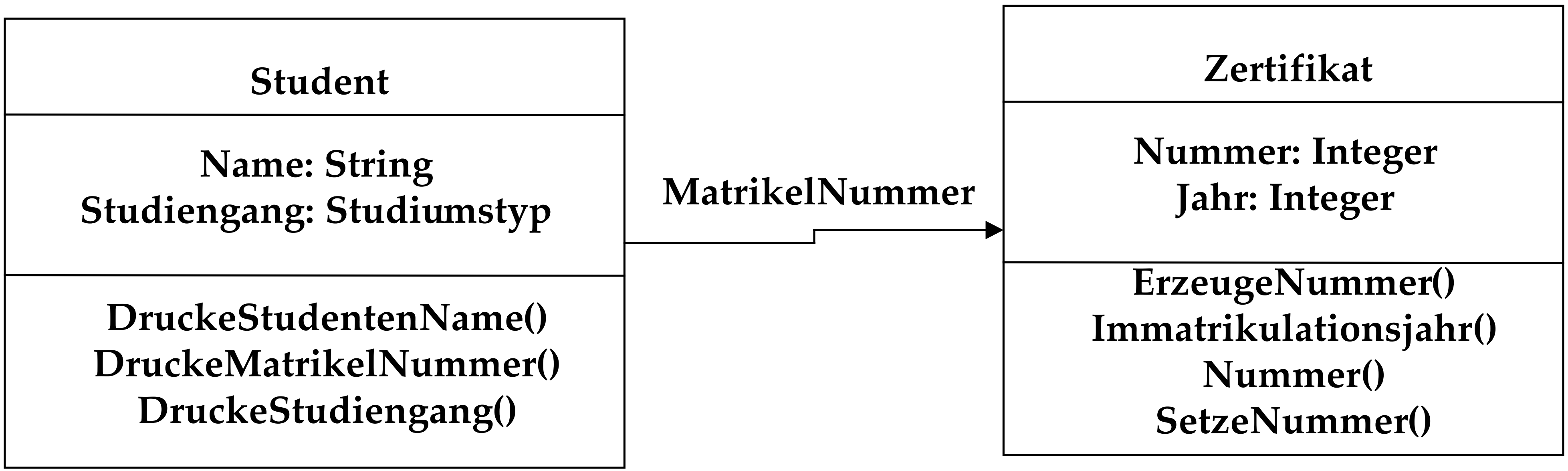


Attribute

- Ein Attribut einer Klasse wird in der Attributliste der Klasse als **Name** (während der Analyse) oder als **Name: Typ** (während des Detaillierten Entwurfs) aufgelistet. Beispiele von Typen sind
  - **String:** Die Menge aller Zeichenketten
  - **Integer:** Die Menge aller ganzen Zahlen
  - **Boolean:** Die Menge der Wahrheitswerte {Wahr, Falsch}
  - **Studiumstyp:** Die Menge der Studiengänge {Diplom, Bachelor}
- Ein Attribut kann auch als Beziehung zu einer Klasse gezeichnet werden, insbesondere wenn die Beziehung zwischen beiden Klassen klargemacht werden soll.
  - Der Name des Attributes ist dann der Name der Beziehung.

Attribut vs. Klasse





Operation

- Die Menge von Operationen, die eine Klasse oder eine Menge von Klassen (Subsystem) zur Verfügung stellt, bezeichnen wir als Schnittstelle der Klasse (des Subsystems).
- Diese Operationen arbeiten auf Attributen der Klasse und anderen Objekten, mit denen die Klasse eine Beziehung hat.

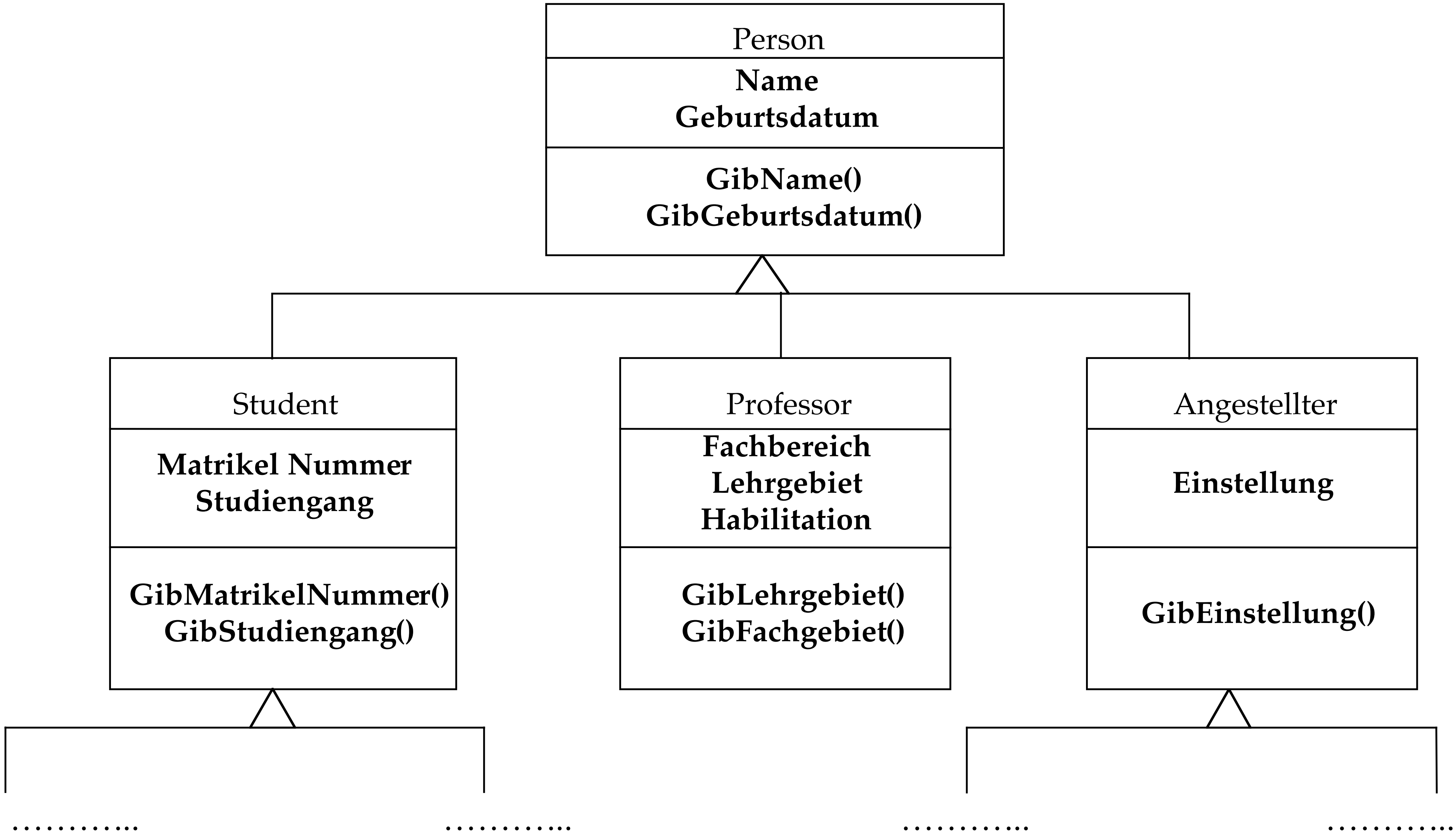
Zwei wichtige Prinzipien der Modellierung

- **Informationskapselung** (information hiding): Objekte können auf andere Objekte nur über deren Schnittstelle zugreifen.
  - Ein Objekt kann also nicht direkt auf die Attribute eines anderen Objektes zugreifen.
- **Klassifikation:** Komponenten können nach ihrer Schnittstelle klassifiziert werden.
  - Beispiel: Zwei Objekte Obj1 und Obj2 können zusammengefasst werden, wenn sie dieselbe Operation *Print()* verstehen.
- Klassifikationen kann man benutzen, um Mengen von Objekten hierarchisch zu strukturieren.
  - Beispiel: Die Personengruppen an einer Universität.

Klassifikation von Personengruppen an einer Universität

- Person
  Name
  Geburtsdatum
  - Student
    MatrikelNummer
    Studiengang
    - BachelorStudent
      ECTSPunkte
    - DiplomStudent
      DVP, DHP
  - Professor
    Fachgebiet
    Lehrgebiet
    Habilitation
  - Angestellte
    Einstellungsdatum
    - Wissenschaftliche Angestellte
      Promotionsthema
    - Verwaltungs Angestellte
      Dezernat

Klassifikation von Personengruppen an einer Universität

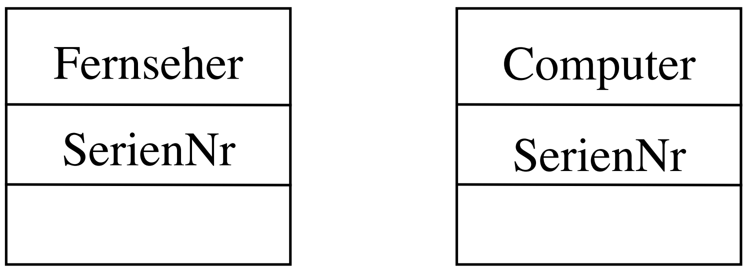


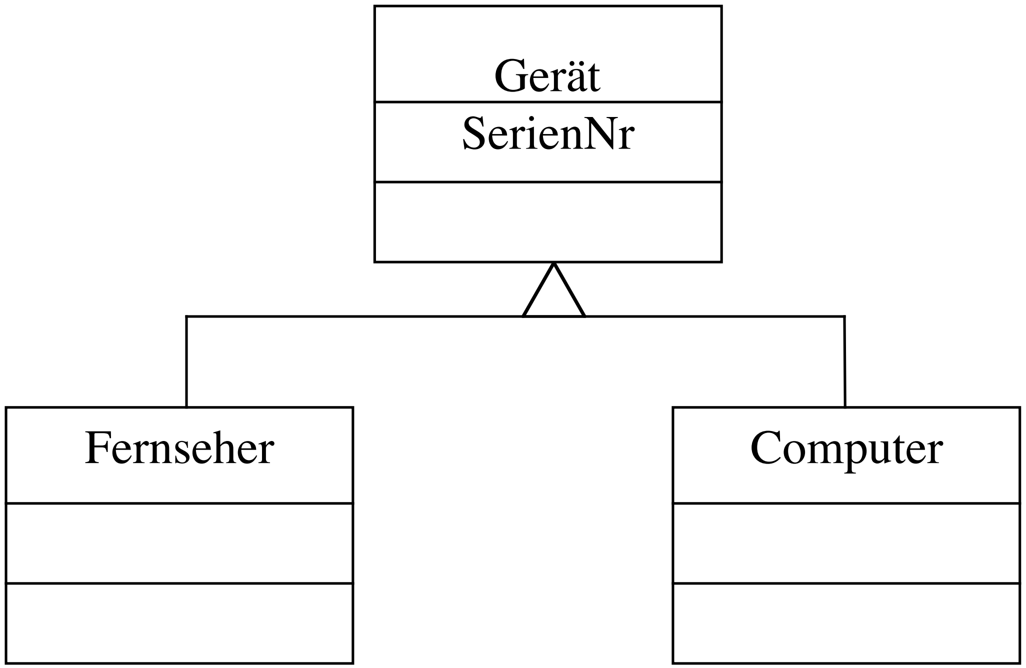
Die Vererbungsbeziehung

- Zwei Klassen stehen in einer **Vererbungsbeziehung** (inheritance relationship) zueinander, falls die eine Klasse, auch **Unterklasse** (Subklasse) genannt, alle Merkmale der anderen Klasse, auch **Oberklasse** genannt, besitzt, und darüber hinaus noch zusätzliche Merkmale.
- Es gilt somit für die Mengen A_S/A_O der Attribute und die Mengen O_S/O_O der Operationen der Unterklasse S und Oberklasse O:
  - $A_O \subset A_S$ und $O_O \subset O_S$
- Eine Unterklasse wird also durch Hinzufügen von Merkmalen spezialisiert.
- Umgekehrt verallgemeinert die Oberklasse die Unterklasse dadurch, dass sie spezialisierende Eigenschaften weglässt. Wir nennen das auch **Verallgemeinerungsbeziehung** (generalization relationship).

Vererbungsbeispiel

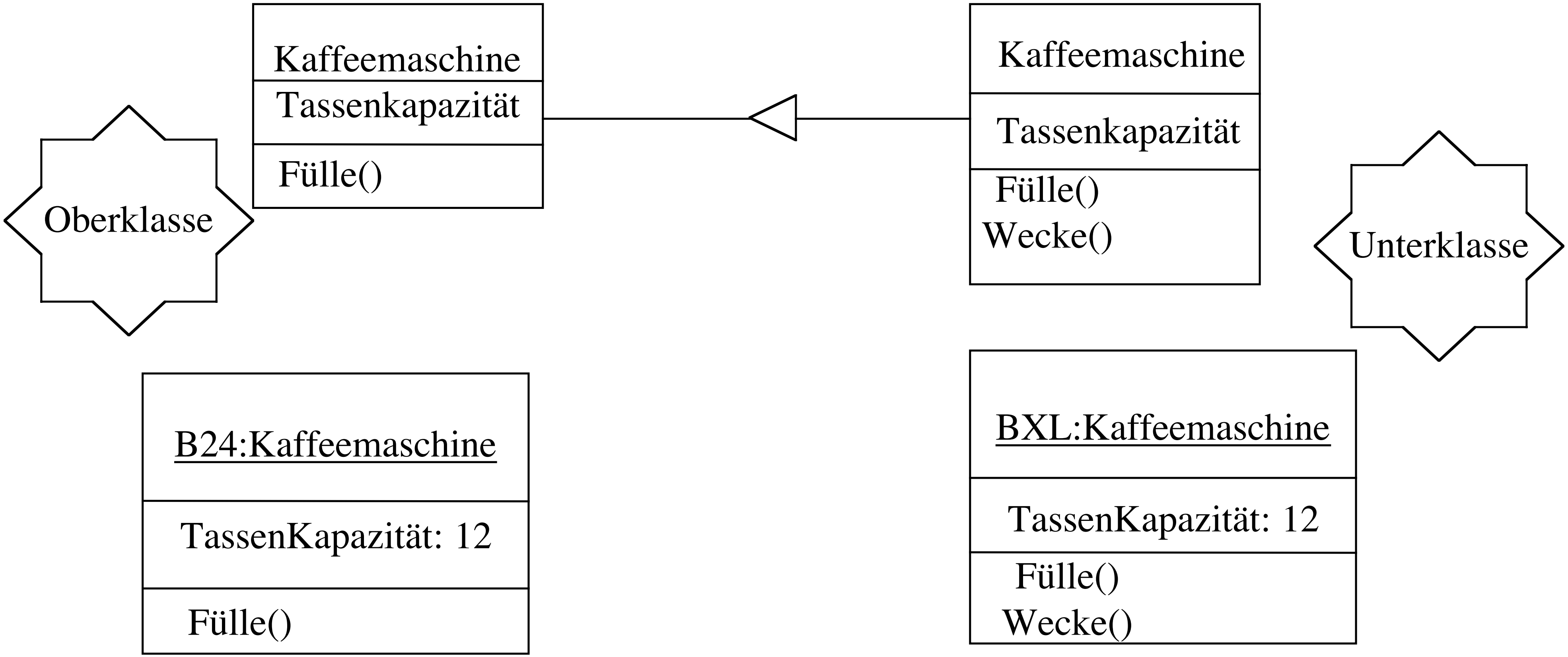
- Eine Firma stellt sowohl Fernsehgeräte als auch Computer her. Beide haben Seriennummern.
- Die Seriennumer wird in einer Oberklasse Gerät angeführt und von dort vererbt.





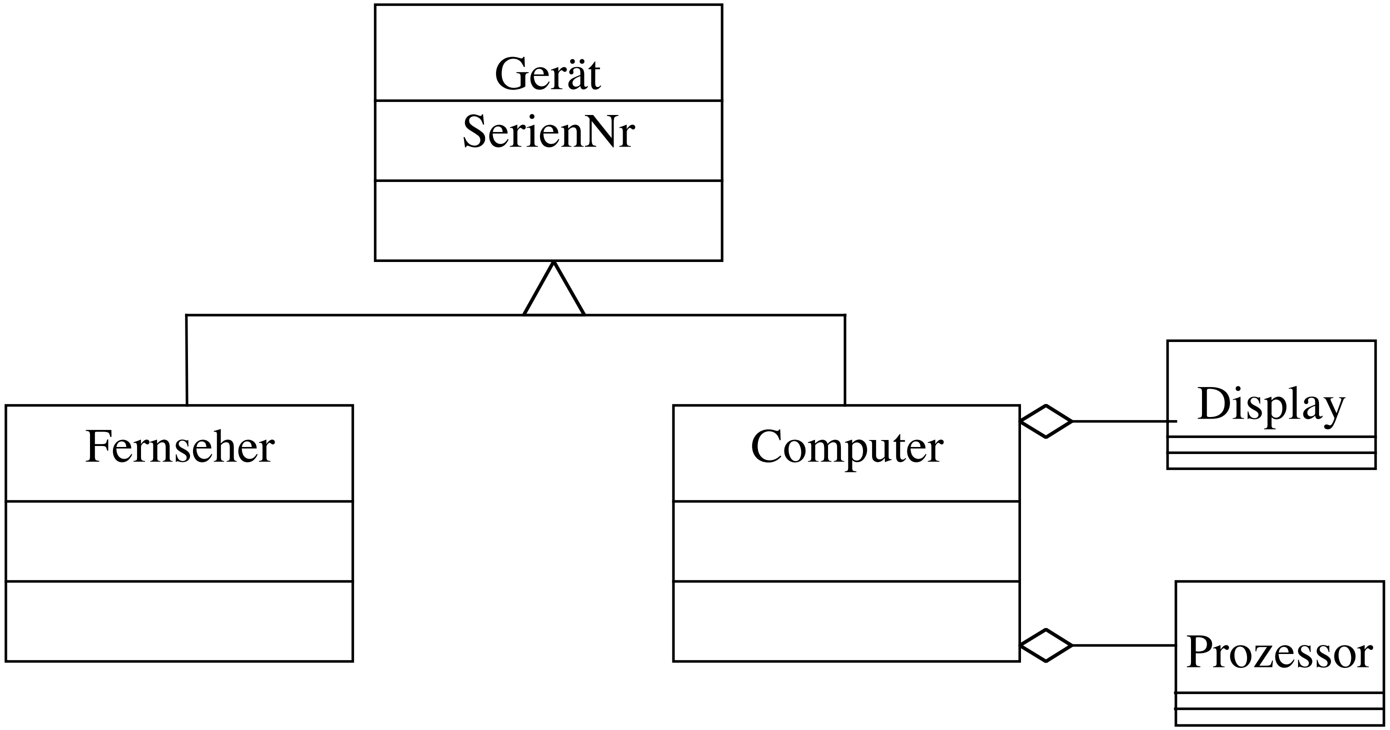
Noch ein Vererbungsbeispiel

- Eine Kaffeemaschine kann bis 12 Tassen halten.
- Das Luxusmodell hat noch eine Weckfunktion



Aggregation und Vererbung lassen sich kombinieren

- In der Modellierung tritt oft der Fall auf, dass wir Gegenstände klassifizieren müssen, aber gleichzeitig auch deren Struktur erkenntlich machen wollen.

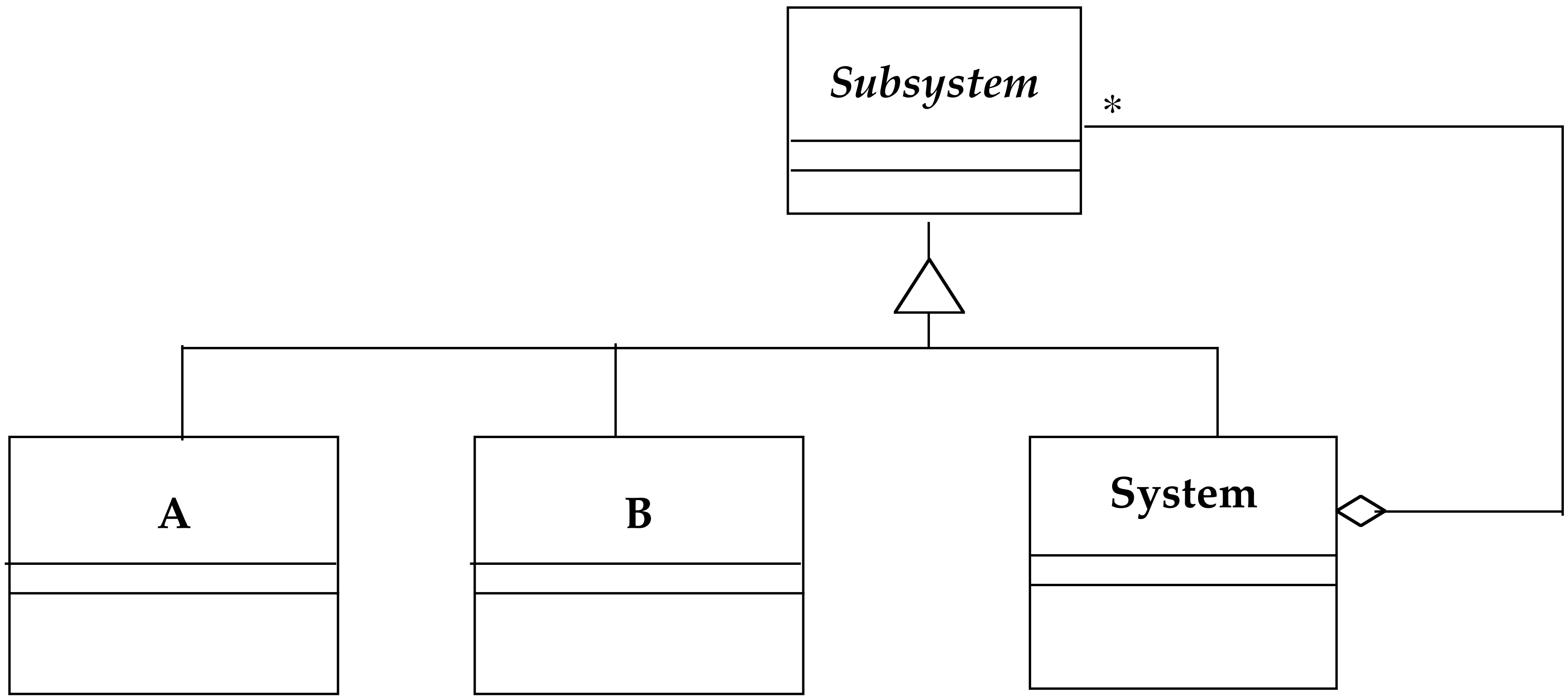


Modellierung unserer Systemdefinition

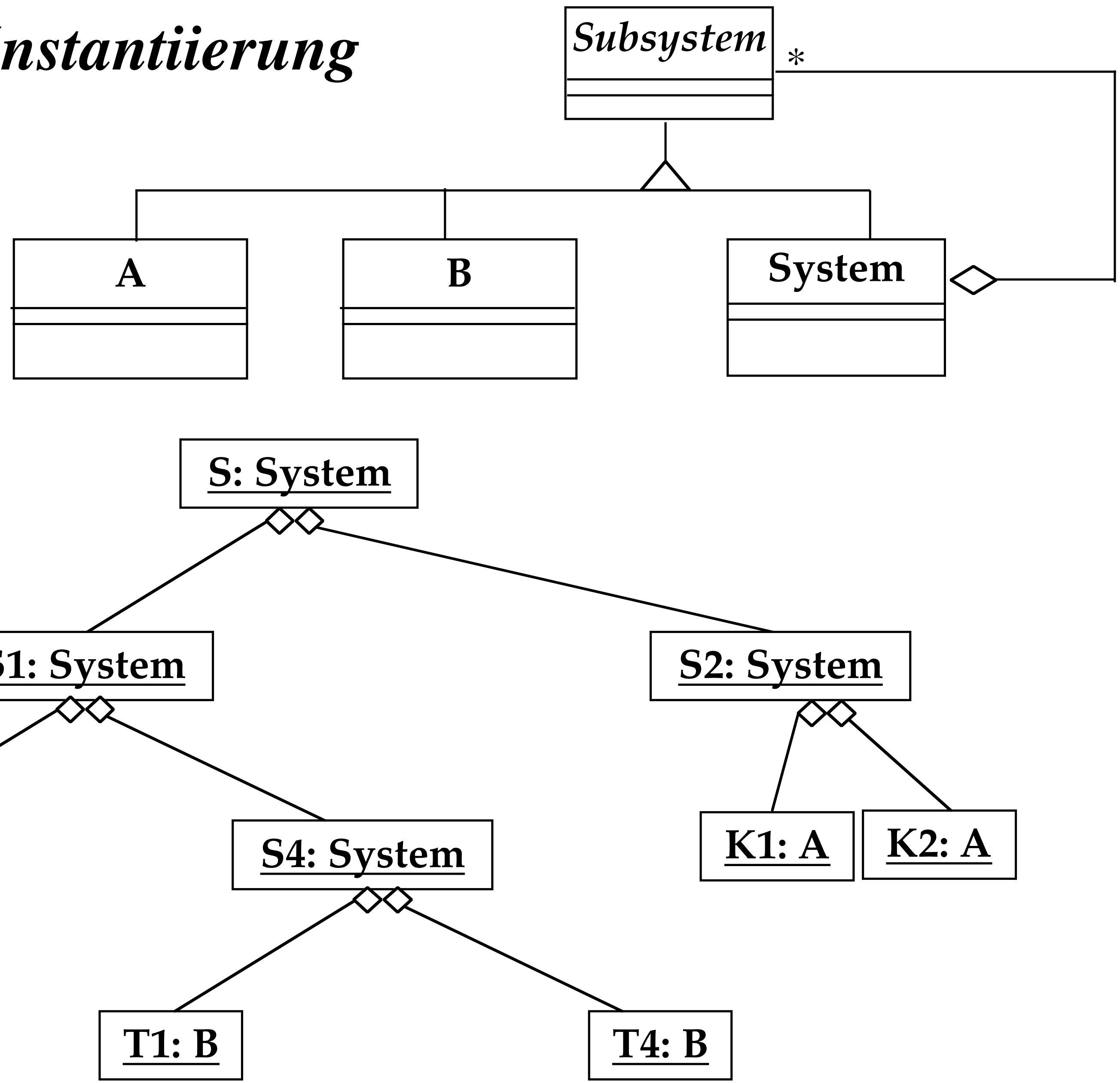
- **Definition eines Systems:** Unter einem System versteht man eine *Menge von Komponenten*, die in einem gegebenen Bezugssystem in einem Zusammenhang stehen, und die *Beziehungen zwischen diesen Komponenten*.
- Die Komponenten eines Systems können selbst wieder (Sub)Systeme sein.

Modellierung des Systembegriffs

- Beispiel: Ein System kann beliebig viele Subsysteme und 2 Arten von Komponenten A und B haben.

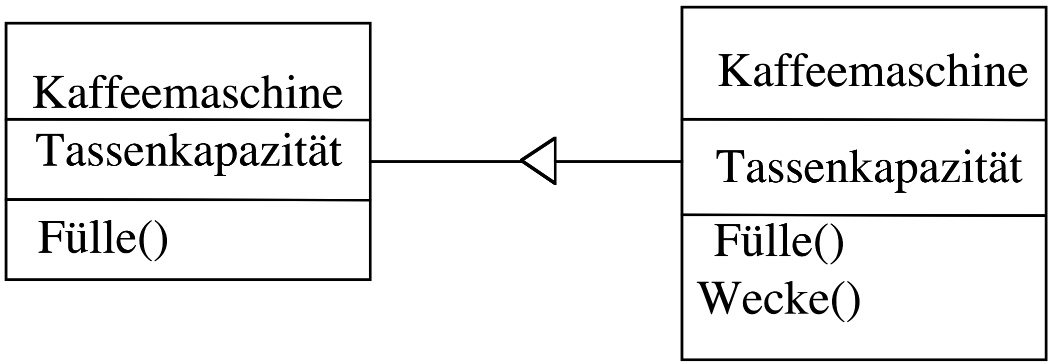


Beispiel einer Instantiierung

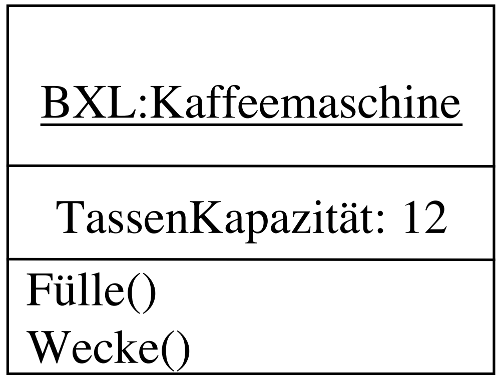


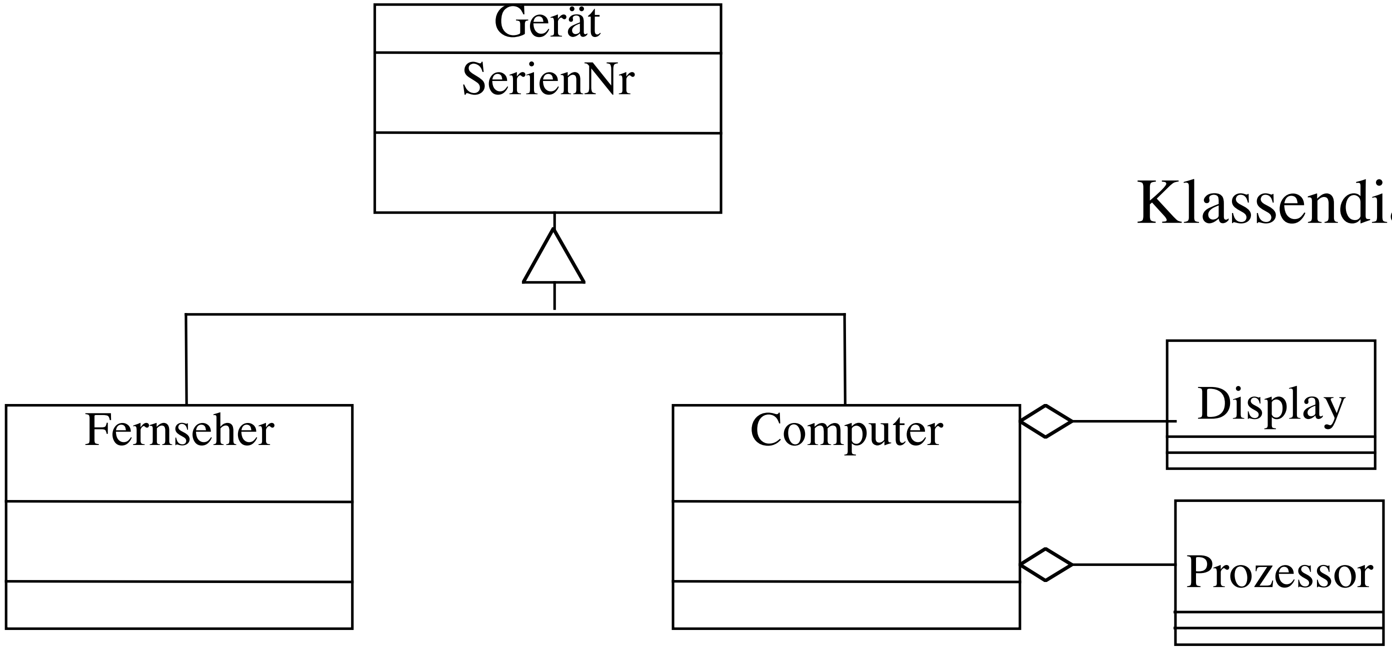
Was passiert mit der Vererbung bei Instanzdiagrammen?

Klassendiagramm

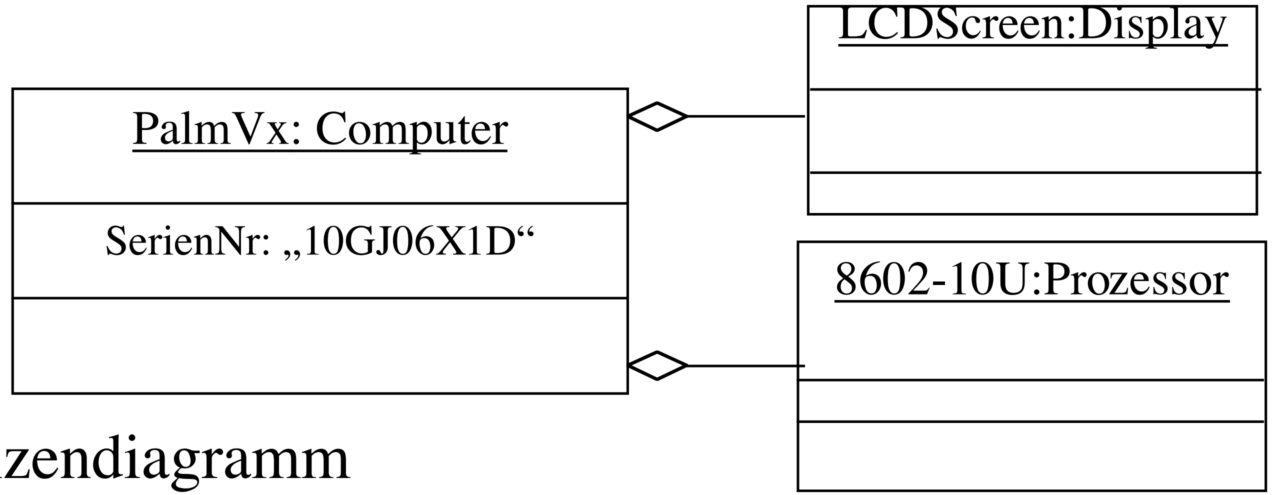


Objektdiagramm



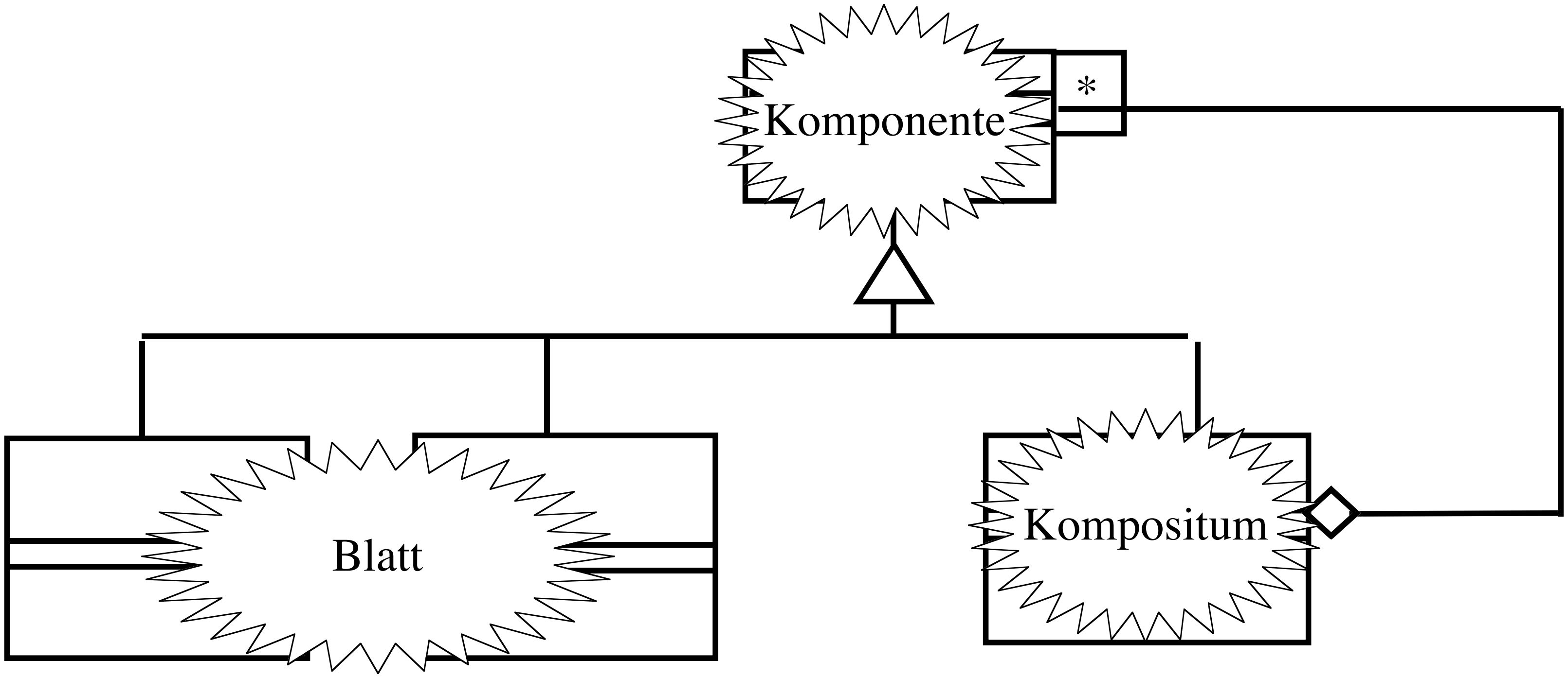


Klassendiagramm



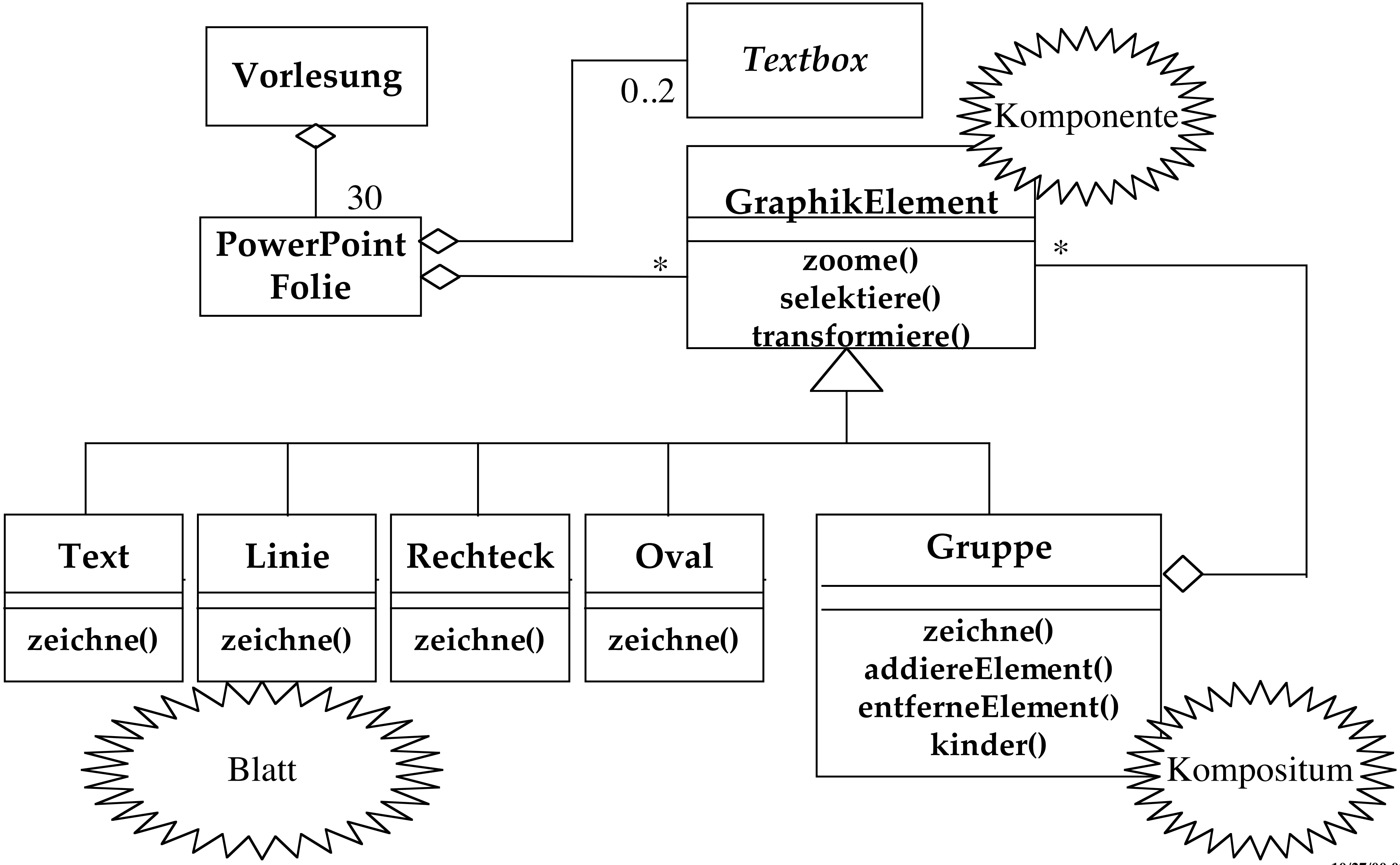
Instanzendiagramm

Kompositionsmuster 10/30/00



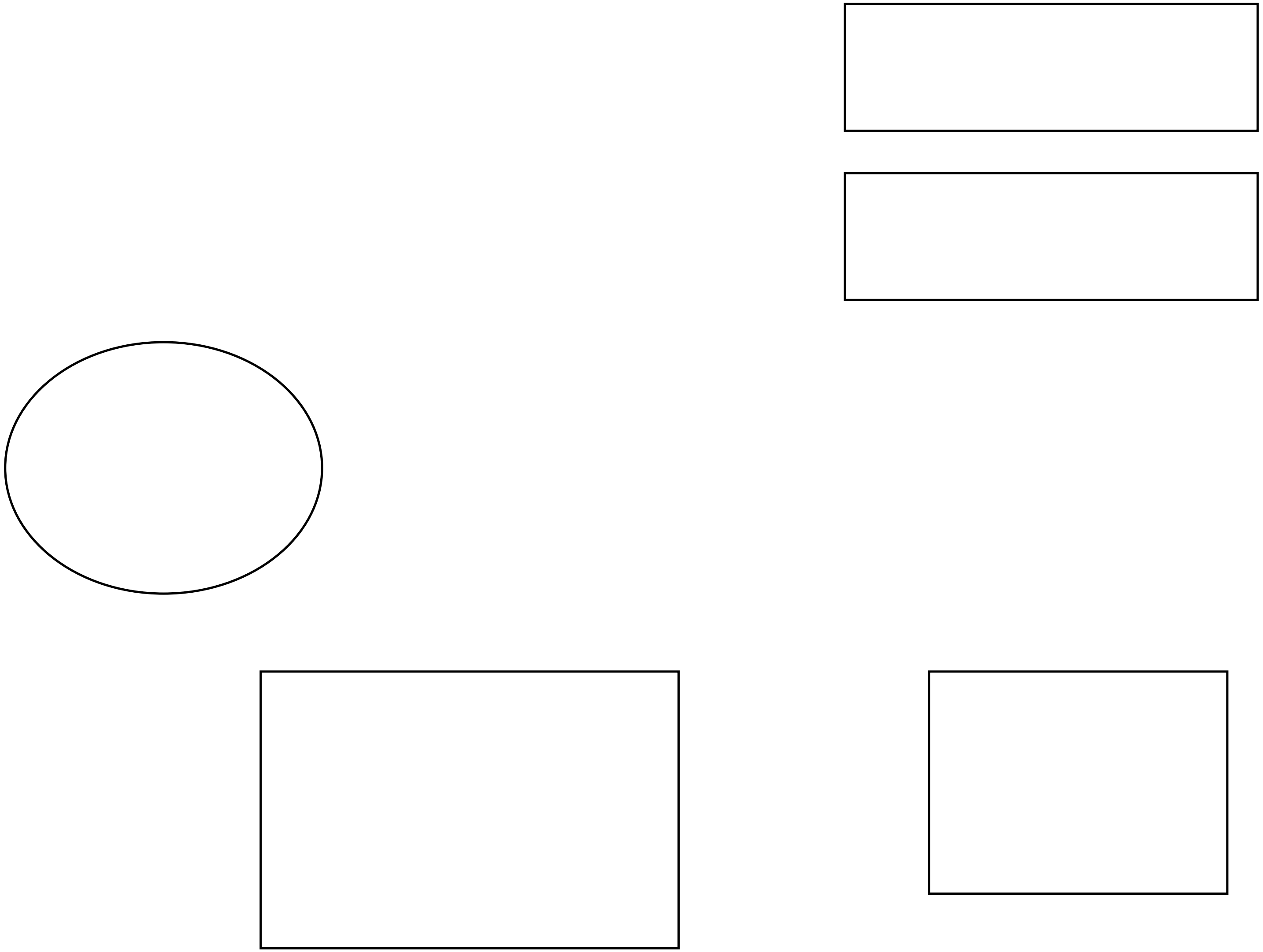
Gamma et. al: Composite Pattern,
Kompositum

Modellierung der Info I Vorlesung

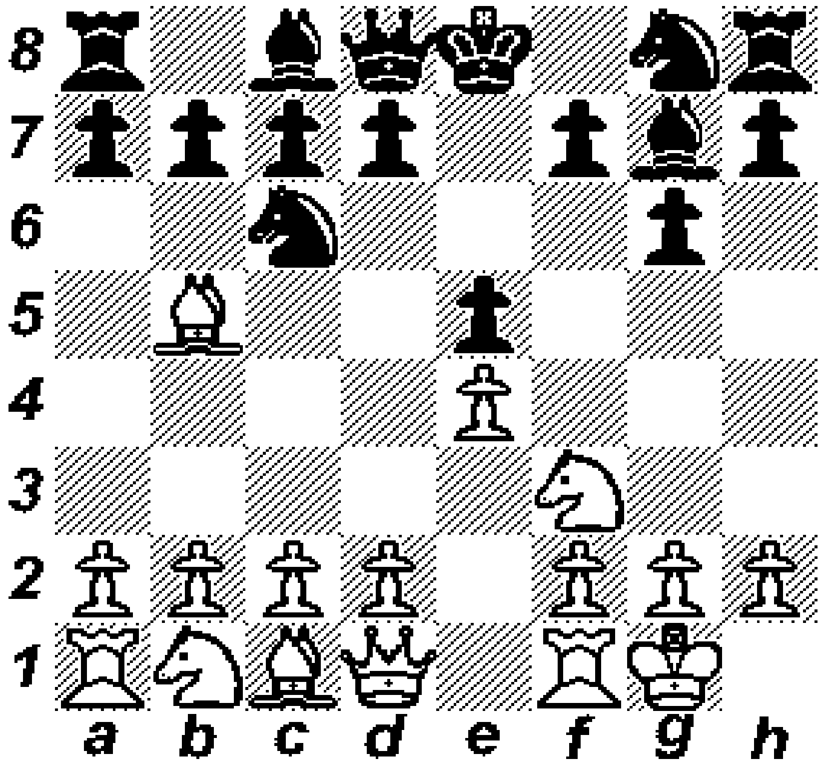


Dies ist ein Beispiel

- Eine äusserst interessante Folie



Muster im Schach



- Spanische Eröffnung
- Schwarz: Läufer Fiancetto
- Weiss: Kurze Rochade

Nützlichkeit von Entwurfsmustern

- Entwurfsmuster sind wiederverwendbares Wissen bei der Entwicklung von Informatik-Systemen, vor allem bei der Analyse und beim Systementwurf.
- Entwurfsmuster lassen sich zu einem Gesamtentwurf kombinieren, der dann als Grundlage für ein Informatik-System dienen kann.
- Wir werden noch weitere Entwurfsmuster kennenlernen:
  - Stellvertreter-Muster (Proxy Pattern): Beispiel Web Browser
  - Beobachter-Muster (Observer Pattern): Applets
  - Adapter-Muster (auch „Wrapper“ genannt): Zum Aufruf von alten, nicht mehr änderbaren Schnittstellen

Formalisierung von Schnittstellen: Algebra und Rechenstruktur

- Wir wollen jetzt Schnittstellen formalisieren:
- Eine **Algebra** ist eine Familie von Mengen und eine Menge von Abbildungen zwischen diesen Mengen.
  - Bei Broy heißen diese Mengen auch **Trägermengen**, und Bezeichner für diese Trägermengen heißen **Sorten**.
  - Beispiel für Trägermengen: {Wahr, Falsch}, {0,1,2…..}
  - Beispiel für Sorten: **bool, nat**
- Bei den Informatikern heißt die Algebra auch **Rechenstruktur.**
  - Wir werden beide Begriffe verwenden.

Operationen, Dienste und Programmierschnittstellen

- **Definition Schnittstelle:** Die Menge von Operationen, die eine Klasse oder ein Subsystem zur Verfügung stellt, bezeichnen wir als Schnittstelle der Klasse oder des Subsystems.
- Diese Operationen arbeiten im Allgemeinen mit Argumenten und ergeben Resultate.
  - Beim Systementwurf brauchen wir die Sorten der Argumente und Resultate noch nicht anzugeben. Die Operationsnamen genügen in dieser Phase. Wir nennen eine solche Schnittstelle einen **Dienst.**
  - Beim Detaillierten Entwurf geben wir die Sorten der Argumente und Resultate an. Wir nennen die Schnittstelle dann **Programmierschnittstelle** (API).

Definition von Algebren: Vorgehensweise

- Eine Menge von Operationen kann man durch eine Signatur $\sum$ bezeichnen. Wir brauchen also erst einmal die Definition einer **Signatur.**
    - Beispiel einer Signatur: $\sum$ = {ErzeugeVerzeichnis, FügeEin, Sortiere, Suche}
  - Ausdrücke über Signaturen heißen Terme und Formeln. Wir werden also auch **Terme** definieren.
    - a+b*5
    - Sortiere(FügeEin(StudentA, (FügeEin(StudentB, ErzeugeVerzeichnis())))
  - Für die Terme und Formeln gibt es Gesetze. Wir werden auch **Gesetze** definieren.
    - Beispiel von Gesetzen: a + (b+c) = (a+b) + c (Assoziativgesetz)
- Die Kombination einer Signatur, der Menge aller Terme und der darüber geltenden Gesetze heißt **Algebra.**
- Fangen wir jetzt mit der Definition einer Signatur an.

Definitionen

- Sei M eine Menge und $n \in \mathbb{N}$.
- Unter einer n-stelligen Operation von M auf M verstehen wir eine Abbildung $f: M^n \rightarrow M$.
  - M^n ist das kartesische Produkt M_1 x ….x M_n
  - Die m_i von $f(m_1, m_2, \ldots m_n)$ heißen die Argumente der Operation f

Noch eine Definition: $\sum^{(n)}$

- Gegeben sei eine Menge von Operationen auf einer Menge M. Dann sagen wir $\sum^{(n)}$ besteht aus allen Operationen *f* der Stelligkeit n. Z.B:
  - $\sum^{(1)}$**:** Besteht aus allen einstelligen (unären) Operationen:
    - das Komplement einer Menge
  - $\sum^{(2)}$**:** Besteht aus allen zweistelligen (binären) Operationen:
    - die Addition + zwischen zwei Zahlen: a+b
    - Einfügen von Studenten ins Verzeichnis: *fügeEin(Student, Verzeichnis)*
  - $\sum^{(0)}$**:** Besteht aus allen nullstelligen Operationen, d.h. Operationen ohne Argumente. Sie liefern immer dasselbe Resultat. Man nennt sie auch **Konstante**.
    - 5, 7 oder „Viola Berger“
    - das Erzeugen eines Verzeichnissses: e*rzeugeVerzeichnis*

Schreibweisen für Binäre Operationen

- Für Operationen gibt es verschiedene Schreibweisen für die Verknüpfung.
  - die Addition + zwischen zwei Zahlen: a+b
  - Einfügen von Studenten ins Verzeichnis: *fügeEin(Student, Studentenverzeichnis)*
- Seien a, b $\in M$ und op eine binäre Operation. Dann benutzen wir folgende Schreibweisen:
  - Infixform: a op b — Beispiel: a+b
  - Prefixform: op a b — Beispiel: +ab
  - Postfixform: a b op — Beispiel: ab+
  - Funktionsform: op(a,b) — Beispiel: add(a,b)
- In der Informatik verwenden wir oft die Funktionsform

Definition Signatur (Goos)

- Gegeben sei eine Menge M (die wir auch als Trägermenge bezeichnen).
- **Definition:**
  – Eine Menge $\sum = \sum^{(0)} \cup \sum^{(1)} \cup \ldots \cup \sum^{(n)}$ von Operationen $f\colon M^i \rightarrow M,\ i = 0\ldots n$, heißt eine **Signatur**.

Alternative Definition einer Signatur (Broy)

- **Definition Sorte:** Eine **Sorte** ist der Name einer Trägermenge
  - Beispiele:
    - **bool** ist der Name einer Menge *{False, True}*
    - **nat** ist der Name einer Menge *{0,1,2,….}*
- **Definition Signatur:** Eine Signatur $\sum$ ist ein Paar (S, F) von Mengen S und F, wobei S eine Menge von Sorten und F eine Menge von Funktionssymbolen ist:
  - Für jedes Funktionssymbol f $\in$ F ist eine Abbildung definiert, sodass
  - f: $(s_1 \times s_2, \times \ldots \times s_n) \rightarrow s_{n+1}$
- f heißt auch die Funktionalität und wird auch geschrieben als
  - **fct** f = $(s_1, s_2, \ldots, s_n)\ s_{n+1}$
  - Funktionalität der Addition ist **fct** add = (**nat**, **nat**) **nat**

Elementare Operanden

- Ist eine Signatur definiert, dann können wir darüber Ausdrücke formulieren, z.B. $5x+b$, oder $(M \cup N) \cap P$ oder *fügeEin(„Viola Berger“,ErzeugeVerzeichnis())*. Solche Ausdrücke enthalten elementare, nicht weiter zerlegbare Operanden.
- **Definition Menge der Elementaroperanden:** Die Menge X von Symbolen,
  - die nicht weiter zerlegbare Operanden in n-stelligen Operationen darstellen.
  - und die Menge aller Konstanten $\sum^{(0)}$.
- Beispiel: Der Ausdruck $5x + b$ enthält drei elementare Operanden 5, b und x
  - *b und x müssen in X definiert sein*
  - *5 ist eine Konstante aus $\sum^{(0)}$*

Termalgebra

- Für eine gegebene Signatur existiert immer eine spezielle Algebra, die Termalgebra, die sich aus der Menge der Terme, d.h. der Menge der über der Signatur formulierbaren Terme zusammensetzt.

Term

- **Definition:** Zu einer Signatur Í und einer Menge X von Elementaroperanden definieren wir die Menge T aller syntaktisch korrekten **Terme** wie folgt:
  - ein elementarer Operand a $\in$ X ist ein Term
  - die Anwendung einer Operation f(a, b,…) mit f $\in \sum$ ist ein Term, wenn jeder Operand a, b… ein Term ist.
- Beispiel: Sei X = {b, x} und $\sum$ = {5, +}.
  - Dann ist die Zeichenkette 5 + *b* ein Term, denn sie besteht aus
    - zwei elementaren Operanden 5 und b
      - *b ist in X definiert*
      - *5 ist eine Konstante aus* $\sum^{(0)}$
    - und der Anwendung der zweistelligen Operation + $\in \sum^{(2)}$ auf die zwei Terme 5 und b.

Definition Abstrakte Algebra

- Gegeben sei eine Signatur $\sum$, eine Menge von Elementaroperanden X und eine Menge von Gesetzen Q für die Anwendung von Operationen aus $\sum$.
- Dann heißt A = ($\sum$, *Q)* eine abstrakte Algebra.
- Wichtig
  - Durch die Definition der Signatur und der Menge der Elementaroperanden X ist die Menge *T* aller syntaktisch korrekten Terme für X und $\sum$ definiert.
- Jede abstrakte Algebra *A* besitzt also eine Menge *T* von syntaktisch korrekten Termen.
  - Ein syntaktisch korrekter Term kann als eine Folge von Operationen auf anderen Termen verstanden werden.

Beispiel für eine Algebra: Keller

- Problembeschreibung:
- Ein **Keller** (auch Stapel genannt) ist ein wichtiger Baustein in der Konstruktion von Informatik-Systemen. Folgende Operationen sind auf einem Keller definiert:
  – Ein Keller kann leer sein. Er ist z.B. leer, wenn er erzeugt wird. Die Erzeugungsoperation nennen wir *createStack*
  – Ein Keller kann Elemente aufnehmen. Die Operation zum Aufnehmen eines Elementes nennen wir *push*.
  – Aus einem Keller kann man das zuletzt aufgenommene Element wiederherausnehmen. Die Operation zum Herausnehmen nennen wir *pop*.
  – Wir können einen Keller fragen, ob er leer ist. Die Operation nennen wir *empty*.

Signatur eines Kellers

- Sei *T* die Menge der möglichen Elemente eines Kellers. Die Signatur $\sum$ des Kellers lautet:

  createStack: $\rightarrow$ Keller(T)

  push: Keller(T) x T $\rightarrow$ Keller(T)

  pop: Keller(T) $\rightarrow$ Keller(T)

  top: Keller(T) $\rightarrow$ T

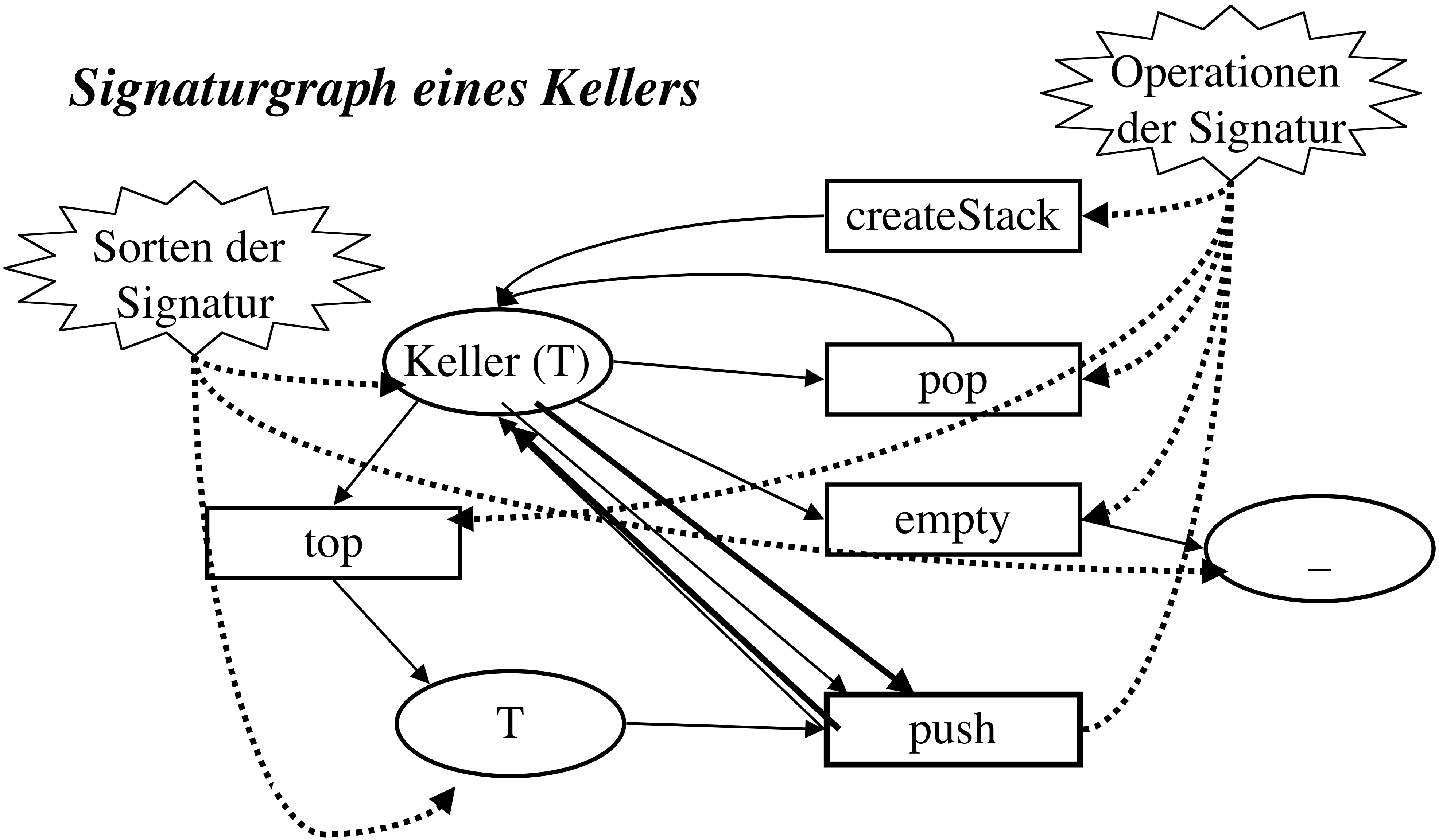
  empty: Keller(T) $\rightarrow$ $\mathbb{B}$

Über die Trägermenge von *T* ist noch nichts gesagt. Ein derartig definierter Keller kann also alles Mögliche aufnehmen.

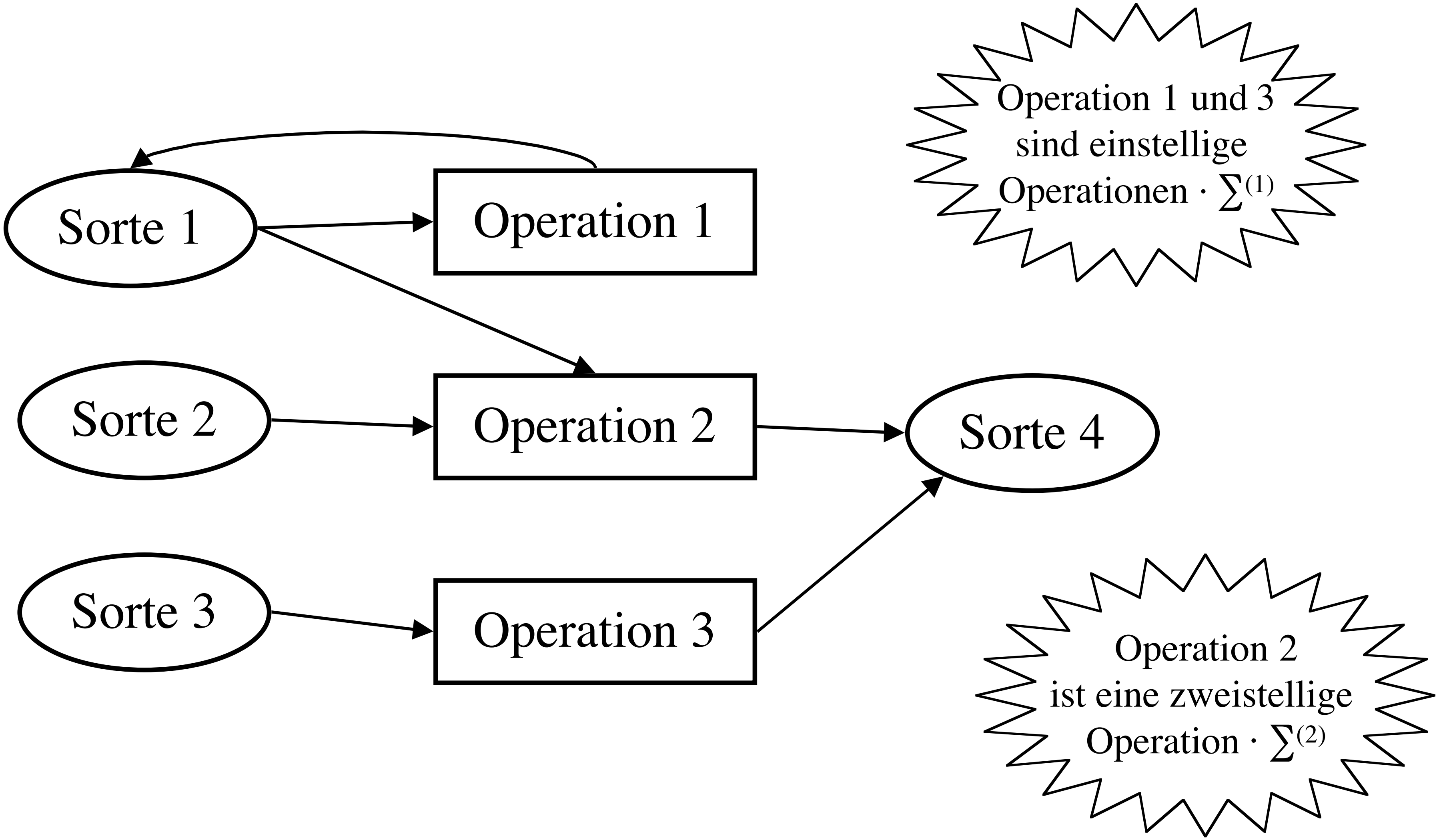
Signaturgraph

- Signaturen lassen sich graphisch übersichtlich durch Signaturgraphen (auch Signaturdiagramme genannt) darstellen.
- Ein Signaturgraph enthält zwei Arten von Knoten
  - Runde Knoten für die Sorten der Signatur
  - Eckige Knoten für die Operationen
- Eine Kante im Signaturgraphen führt
  - Von Sorte zu Operation, wenn die Sorte ein Argument der Operation ist
  - Von Operation zu Sorte, wenn die Sorte Ergebnistyp der Operation ist
- Solch einen Graphen bezeichnet man als **bipartit** (zweigeteilt)

Signaturgraph eines Kellers



Allgemeines Schema für Signaturgraphen



Keller der natürlichen Zahlen

- Sei $\mathbb{N}$ die Menge der natürlichen Zahlen und $\mathbb{B}$ die Menge der Wahrheitswerte {Wahr, Falsch}. Dann sind $\mathbb{N}$ und $\mathbb{B}$ die Trägermengen einer abstrakten Algebra Keller($\mathbb{N}$) mit der Signatur $\sum$:

createStack: $\rightarrow$ Keller($\mathbb{N}$)

push: Keller($\mathbb{N}$) x $\mathbb{N}$ $\rightarrow$ Keller($\mathbb{N}$)

pop: Keller($\mathbb{N}$) $\rightarrow$ Keller($\mathbb{N}$)

top: Keller($\mathbb{N}$) $\rightarrow$ $\mathbb{N}$

empty: Keller($\mathbb{N}$) $\rightarrow$ $\mathbb{B}$

- Ein derartig definierter Keller($\mathbb{N}$) kann nur natürliche Zahlen aufnehmen.
- Kellerinhalt: Die Menge aller Terme für Keller($\mathbb{N}$) enthält unter anderem:

  createStack

  push(createStack, 2)

  push (push(createStack, 2), 34)

  pop (push (push(createStack, 2), 34))

- *Was ist mit*
  - *push(2)?*
  - *push(pop(createStack()), 2))*
  - *top(createStack())?*

Partielle Abbildungen in Algebren

- *top(createStack())* können wir nicht berechnen.
  - Die Verknüpfung von *top* und *createStack* ist undefiniert.
  - Man sagt, *top* ist keine totale Abbildung, sondern nur eine partielle Abbildung. Dasselbe gilt für den syntaktisch korrekten Term *pop(createStack())*.
  - In der Praxis sollte ein Informatik-System eine Fehlermeldung „Undefinierter Wert“ generieren, wenn eine Operation oder ein Term undefiniert ist.
- Zur Bezeichnung von „Undefinierter Wert“ verwenden wir in der Algebra das Zeichen ⊥ (liest man als „Unten“ / „Bottom“)
- Wir können nun definieren:
  - *top*(*createStack()*) = ⊥ *pop*(*createStack()*) = ⊥
- Durch die Verwendung von ⊥ werden *top* und *pop* totale Abbildungen.

Gesetze eines Kellers

- Sei *t* ein beliebiger Wert der Menge *T*, und k ein Element der abstrakten Algebra *Keller(T)*. Dann gelten folgende Gesetze:
  - K1: empty(createStack()) = Wahr
  - K2: empty(push(k,t)) = Falsch
  - K3: pop(push(k,t)) = k
  - K4: top(push(k,t)) = t

Normalformen einer Algebra

Der Kellerwert

pop(push(push(createStack(), 2), 34))

ist gleich dem Kellerwert

push(createStack(), 2)

Wir können dies durch Anwendung von K3 zeigen:

- K3: pop(push(k,t)) = k

- Definition **Normalform x eines Kellers y:** Ein Wert x, der sich durch minimale Anzahl von Operationen auszeichnet, und durch y durch Anwendung von Gesetzen entsteht.
- Kellernormalformen sind Werte
  - *createStack()*
  - *push(….(push(createStack()), n_1),….), n_k)*

Definition Konkrete Algebra

- Gegeben sei eine abstrakte Algebra **A** = ($\sum$, Q).

- **Definition:** $A = (M, \sum, Q)$ heißt eine **konkrete Algebra,** wenn
  - eine Trägermenge M *mit* $X \subseteq M$ gegeben ist, und
  - es zu jeder Operation $f \cdot \sum^{(n)}$ in der Signatur eine konkrete Funktion $f_M : M^n \rightarrow M$ gibt, für welche die Gesetze von Q gelten.
- Beispiele für Trägermengen und Sorten:
  - Menge der natürlichen Zahlen {0,1,2,3,…}: Sorte **nat**
  - Menge der booleschen Werte {Wahr, Falsch}: Sorte **bool**
  - Alle Studenten, die Info I besuchen: Sorte **stud**
  - Die Zeichen im ASCII Code {a,...z,A,…!@#$,..}: Sorte **char**
- Konkrete Algebren sind Komponenten im Sinne der Definition aus der Vorlesung.

Zusammenfassung

- Klasse: Attribute, Operationen
- Objekt als Instanz einer Klasse
- Definition Instanzendiagramm, Klassendiagramm
- Kanonische Beziehungen in der Modellierung: Aggregation, Vererbung
- Kompositionsmuster
- Schnittstelle, Signatur
- Syntaktisch Korrekte Terme
- Definition Abstrakte Algebra, Konkrete Algebra (Rechenstruktur)
- Signaturgraph
- Die Struktur Keller
- Partielle und Totale Abbildungen
- Normalformen