Dr. Hanno Schauer
Mons-Tabor-Gymnasium Montabaur

UML-Klassendiagramme als Werkzeug im Unterricht

Blitzlicht

In welcher Programmiersprache(n) unterrichten Sie?

In welchem Umfang unterrichten Sie Objektorientierung?

Was sind Ihre Erwartungen an diese Fortbildung?

1

UML-Klassendiagramme: Sprachkonzepte

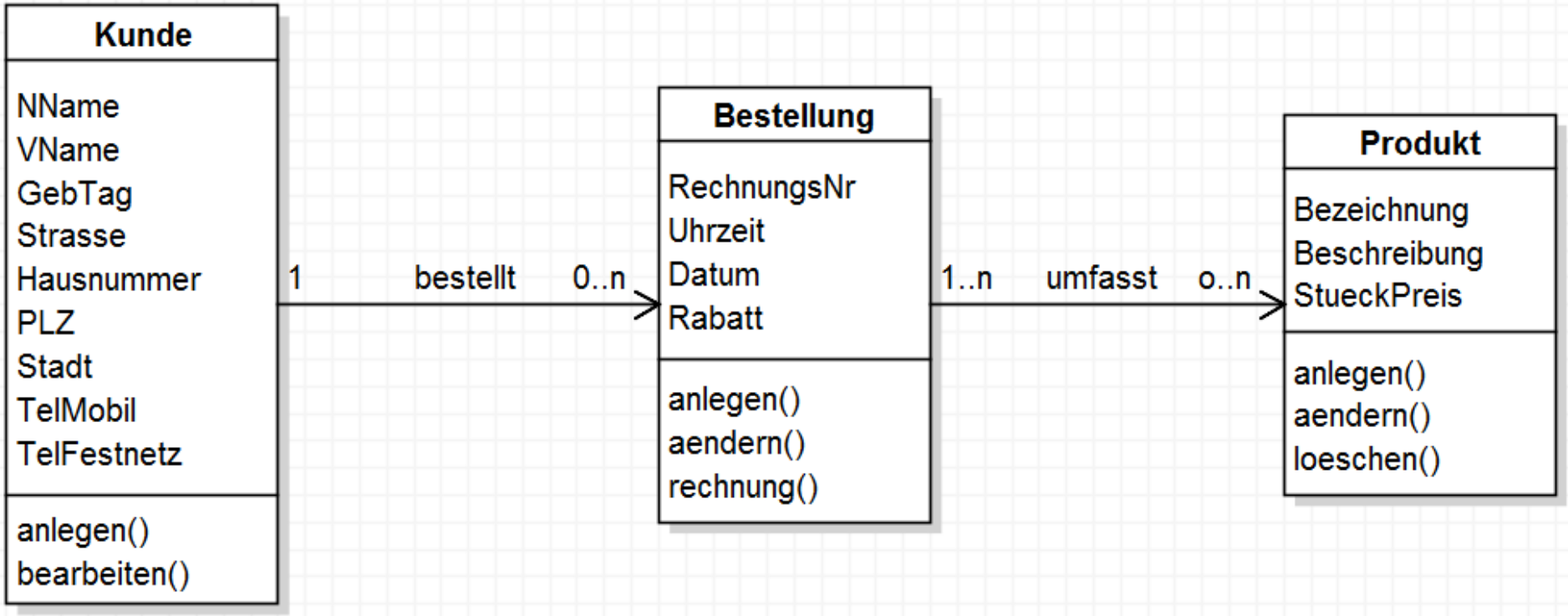
Übung zum Einstieg

Ein **Pizza-Lieferdienst** möchte seine Bestellungen elektronisch verwalten.

Welche **Informationen** muss das geplante Informationssystem erfassen bzw. vorhalten?

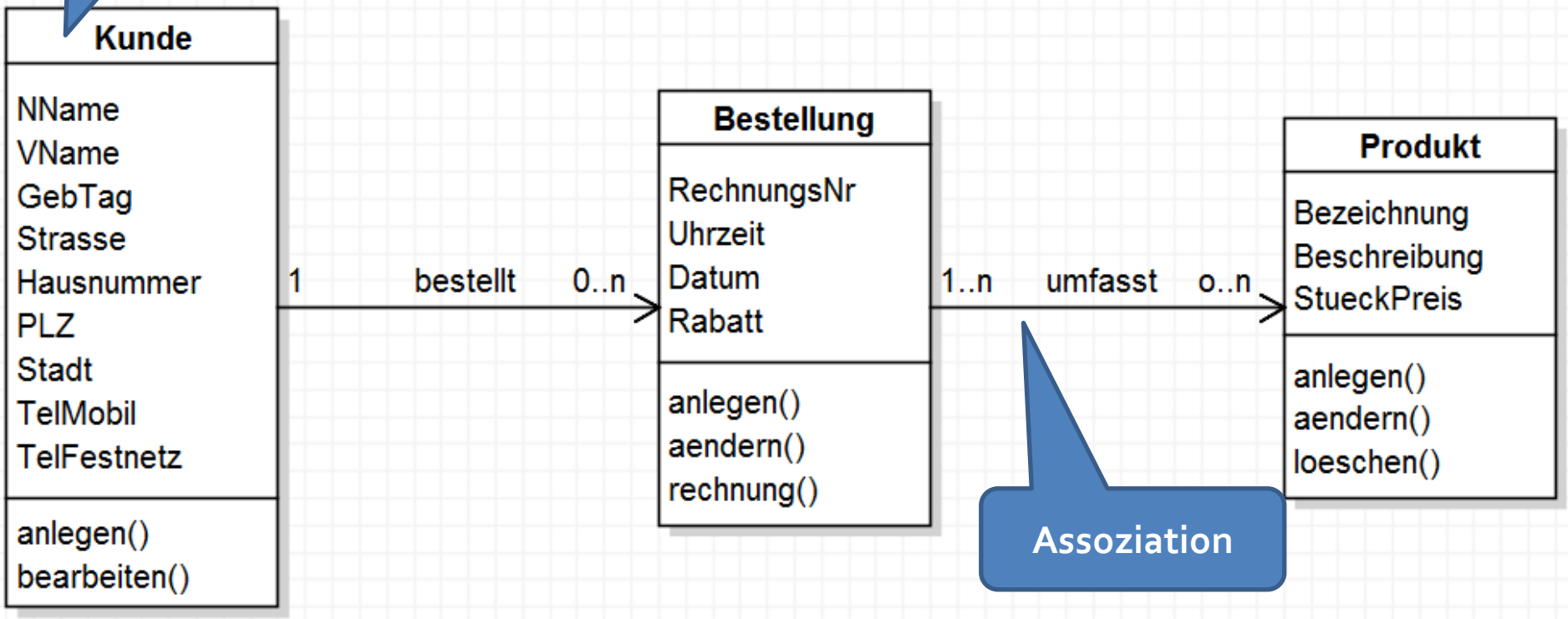
Modellieren Sie ein **UML-Klassendiagramm**.

Grundlegende Sprachkonzepte

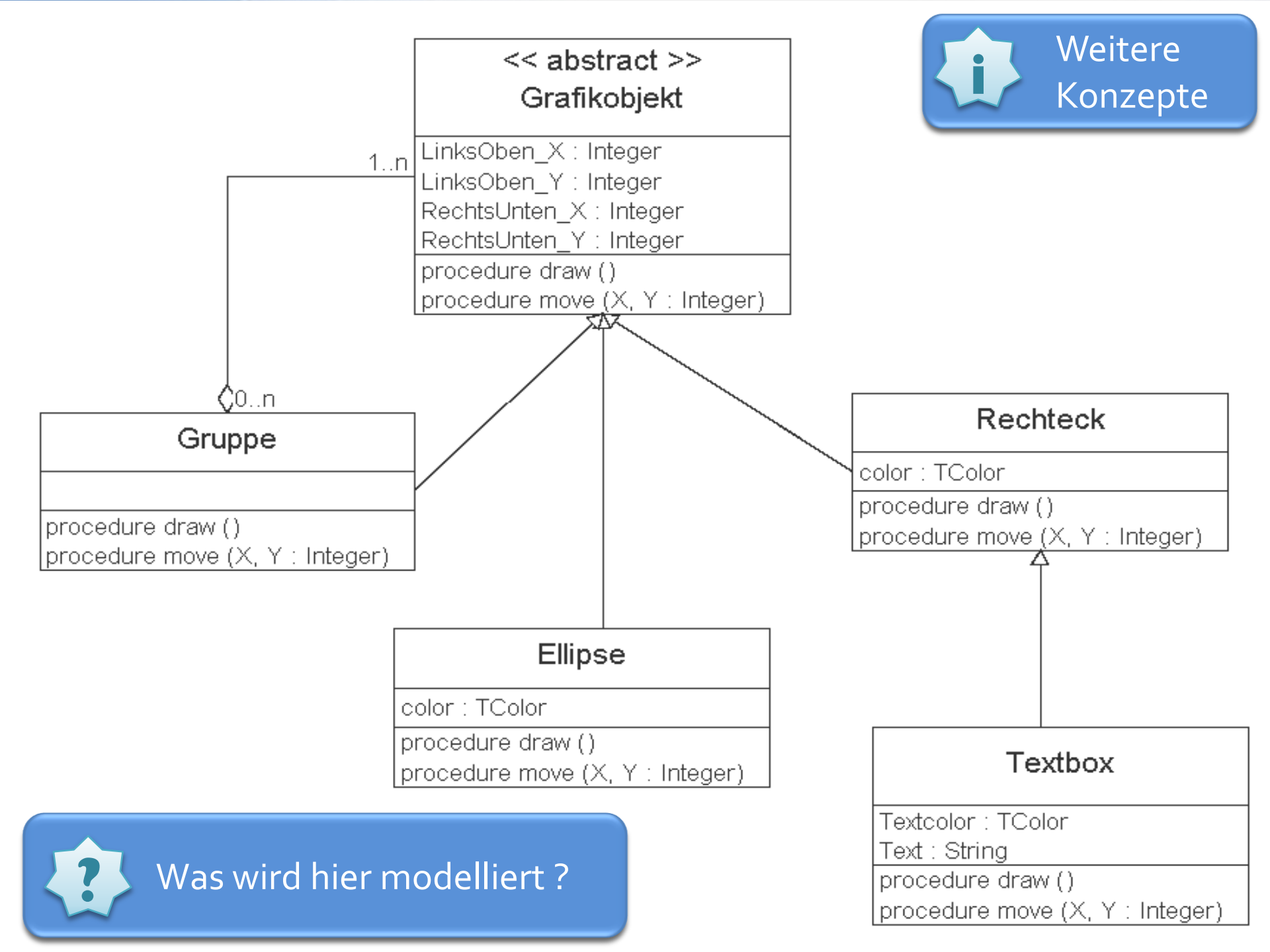


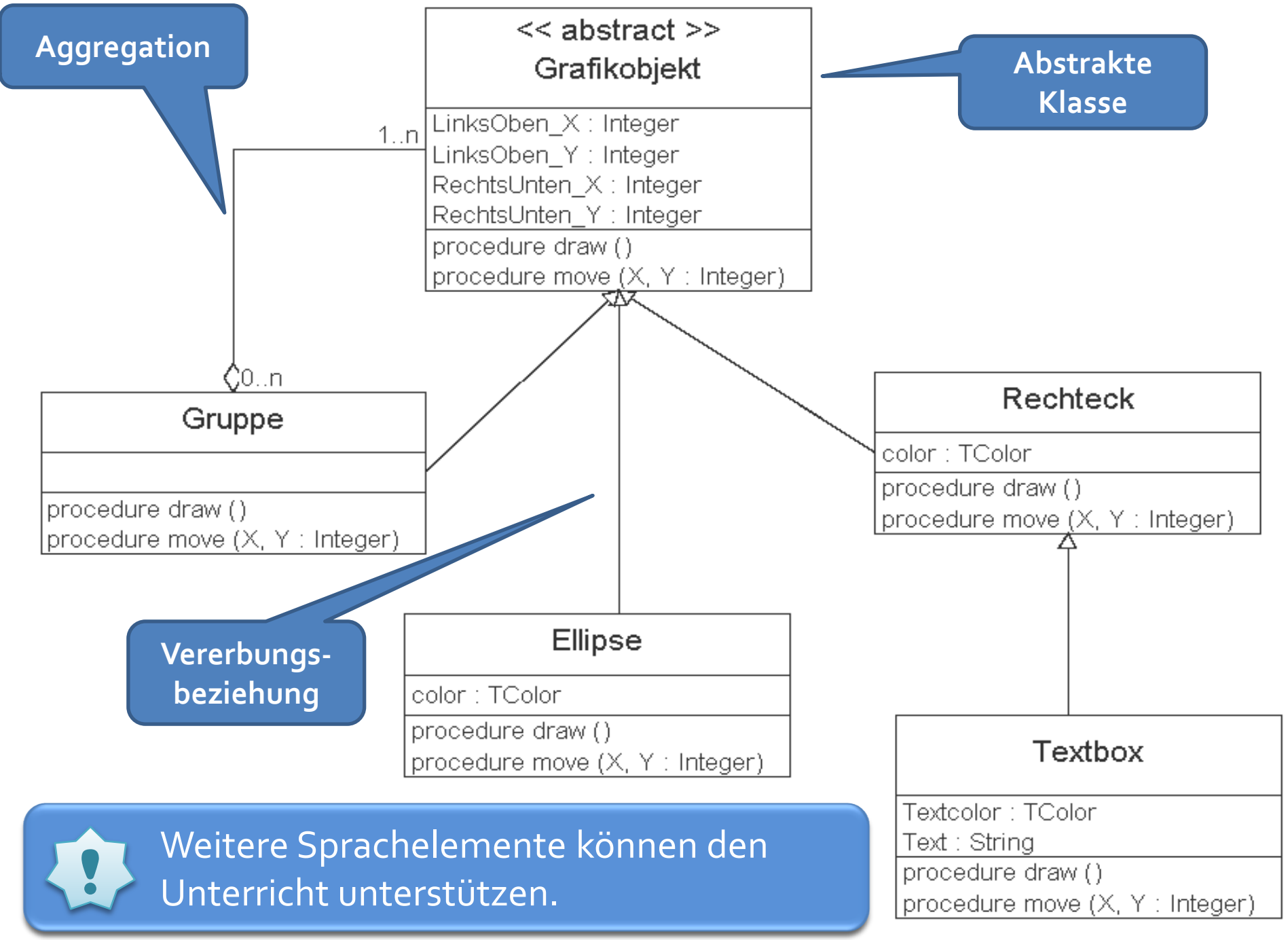
UML-Klassendiagramme: Grundlegende Sprachkonzepte (1)

Klasse:
- Klassenname
- Attribute
- Methoden



Achtung: Assoziationen müssen bei der Implementierung übersetzt werden (i. d. R. zu Attributen).


Weitere Konzepte
<< abstract >>
Grafikobjekt
LinksOben_X : Integer
LinksOben_Y : Integer
RechtsUnten_X : Integer
RechtsUnten_Y : Integer
procedure draw ( )
procedure move (X, Y : Integer)
1..n
0..n
Gruppe
procedure draw ( )
procedure move (X, Y : Integer)
Rechteck
color : TColor
procedure draw ( )
procedure move (X, Y : Integer)
Ellipse
color : TColor
procedure draw ( )
procedure move (X, Y : Integer)
Textbox
Textcolor : TColor
Text : String
procedure draw ( )
procedure move (X, Y : Integer)
Was wird hier modelliert ?


Aggregation
<< abstract >>
Grafikobjekt
LinksOben_X : Integer
LinksOben_Y : Integer
RechtsUnten_X : Integer
RechtsUnten_Y : Integer
procedure draw ( )
procedure move (X, Y : Integer)
Abstrakte Klasse
1..n
0..n
Gruppe
procedure draw ( )
procedure move (X, Y : Integer)
Rechteck
color : TColor
procedure draw ( )
procedure move (X, Y : Integer)
Vererbungs-beziehung
Ellipse
color : TColor
procedure draw ( )
procedure move (X, Y : Integer)
Textbox
Textcolor : TColor
Text : String
procedure draw ( )
procedure move (X, Y : Integer)
Weitere Sprachelemente können den Unterricht unterstützen.

Domänen vs. Implementierungsmodelle

Übersicht

Klassendiagramme werden in unterschiedlichen Phasen der Software-Entwicklung genutzt – insb.:

- **Domänenmodelle**
  (frühe Entwurfsphasen)
- **Implementierungsmodelle**
  (späte Entwurfsphasen)

Werden im Schulunterricht in unterschiedlichen Aufgabentypen genutzt.

2A

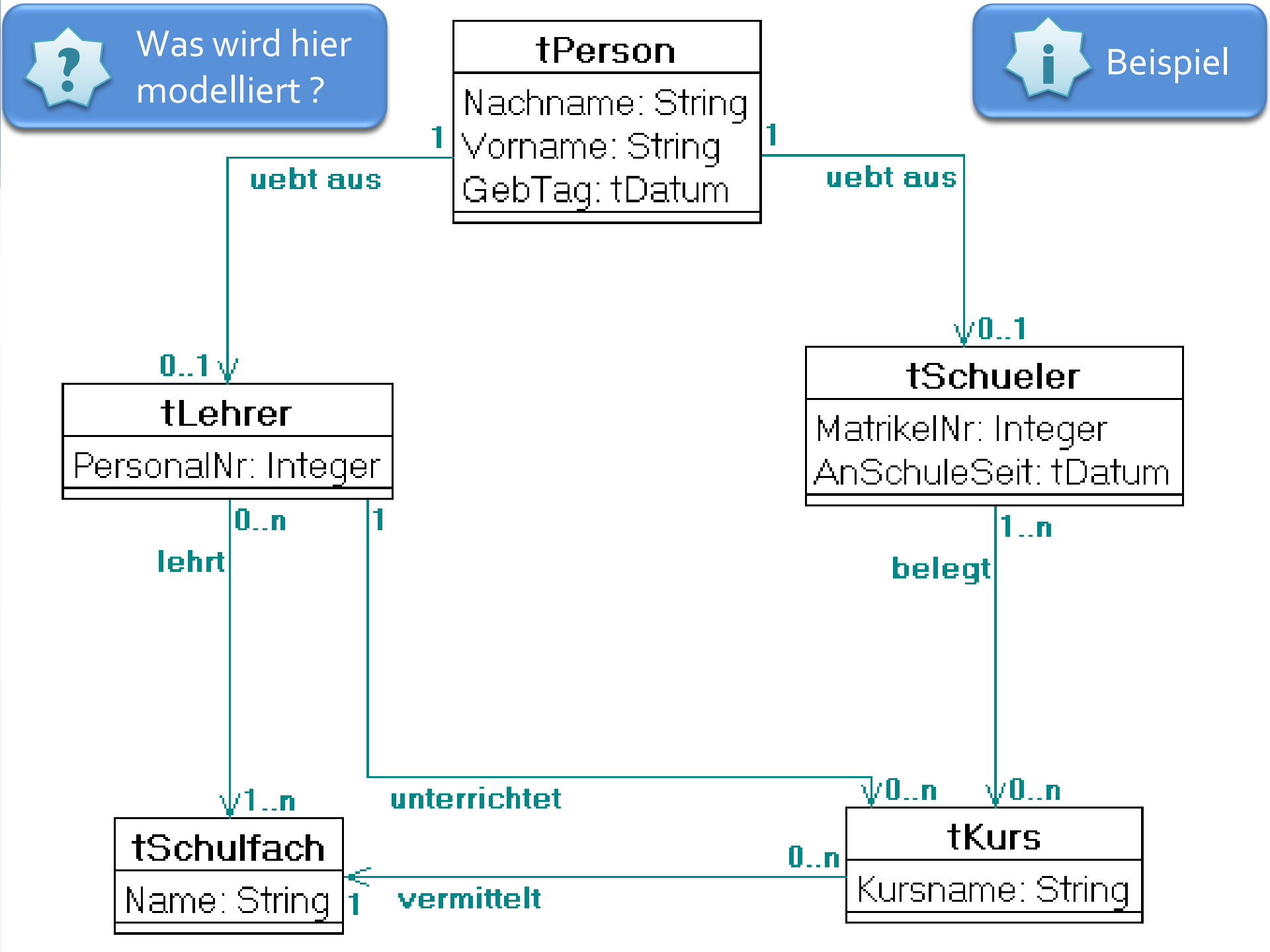
Implementierungsmodelle

Domänenmodelle

- Modellieren die „Domäne" einer Software aus **Anwendersicht** – insb.
  - Verwaltete Informationen
  - Zentrale Funktionalitäten

- **Kommunikationsinstrument** zwischen den Beteiligten
  - Programmierer
  - Anwender
  - Auftraggeber

- **Abstraktionsebene:**
  - Verzicht auf Besonderheiten einer Programmiersprache
  - Häufig unvollständige Modellierung


Was wird hier modelliert ?
Beispiel
tPerson
Nachname: String
Vorname: String
GebTag: tDatum
1
uebt aus
1
uebt aus
0..1
0..1
tLehrer
PersonalNr: Integer
tSchueler
MatrikelNr: Integer
AnSchuleSeit: tDatum
0..n
lehrt
1
1..n
belegt
1..n
tSchulfach
Name: String
unterrichtet
0..n
0..n
tKurs
Kursname: String
0..n
1
vermittelt

Anwendung im Unterricht

- Gemeinsam **modellieren**
  (Gruppenarbeit, Plenum)
- Bestehende **Modelle interpretieren**
  (siehe vorangegangene Beispiele)
- **Modelle vergleichen**
- **Entwurfsentscheidungen** diskutieren

Domänenmodellierung ist
Programmieren im Großen

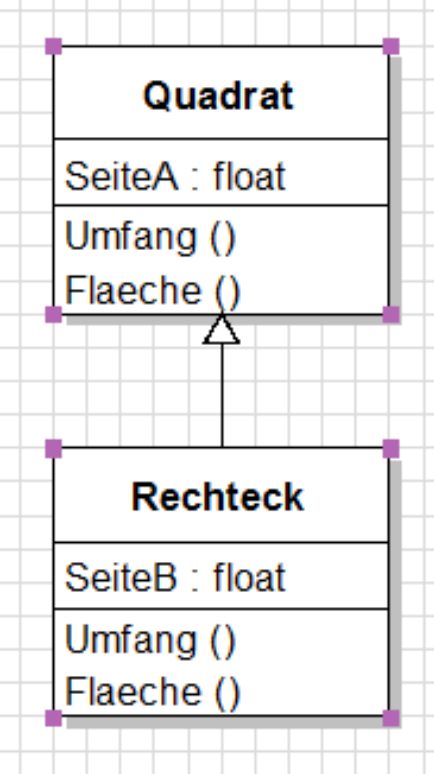
Entwurfsentscheidungen diskutieren
Beispiel: Rechteck – Quadrat

Gesucht:

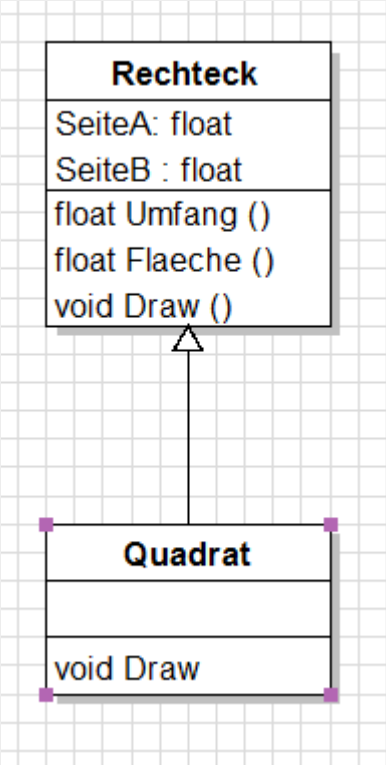
Repräsentation von Rechtecken und Quadraten für ein Geometrie-Programm.

Entwurf 1 + 2

1. Rechtecke sind spezielle Quadrate

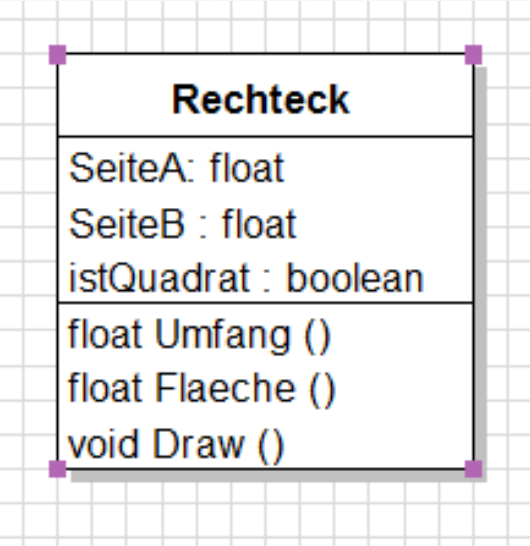


2. Quadrate sind spezielle Rechtecke

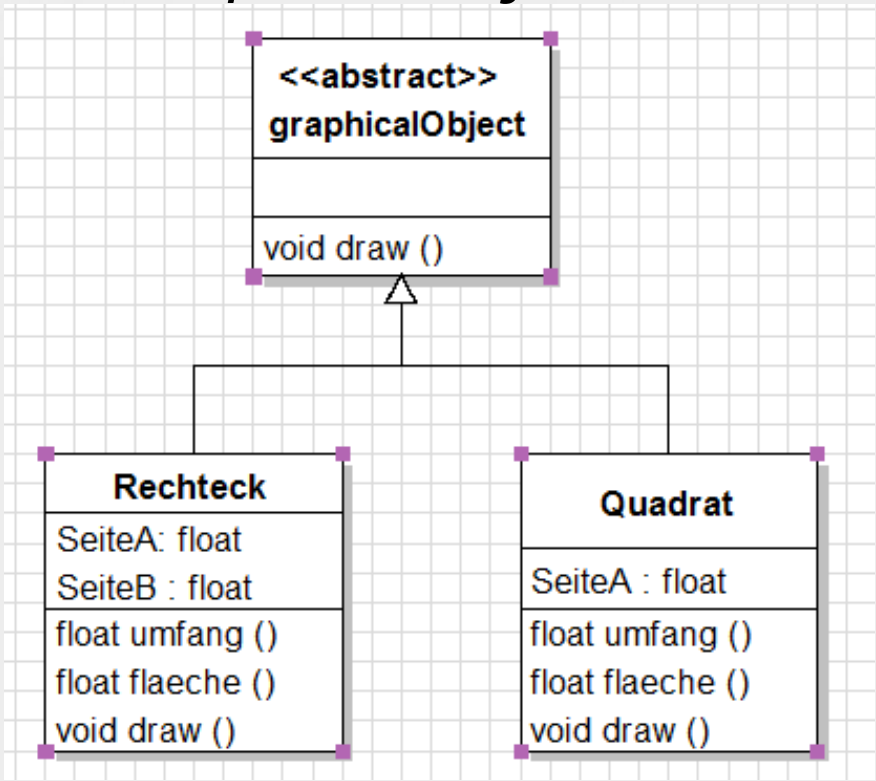


Entwürfe 3 + 4

3. Alles ist ein Rechteck

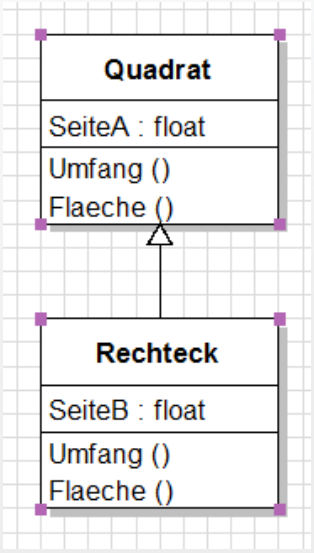


4. Alles ist ein *Graphical Object*

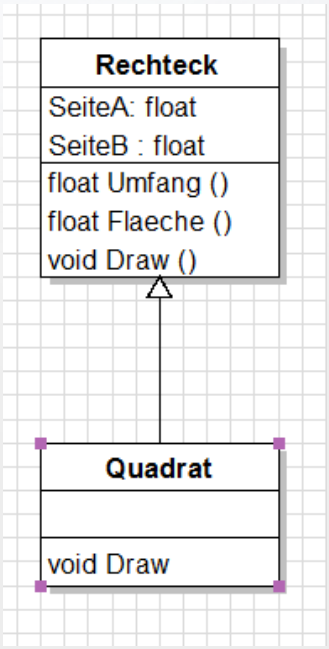


Welcher Entwurf ist geeignet?

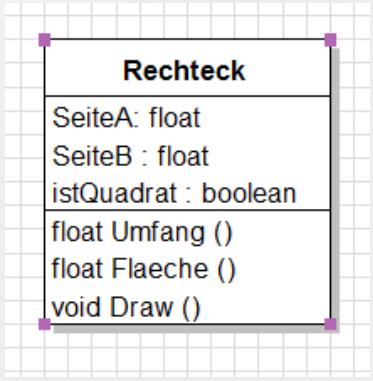
1.



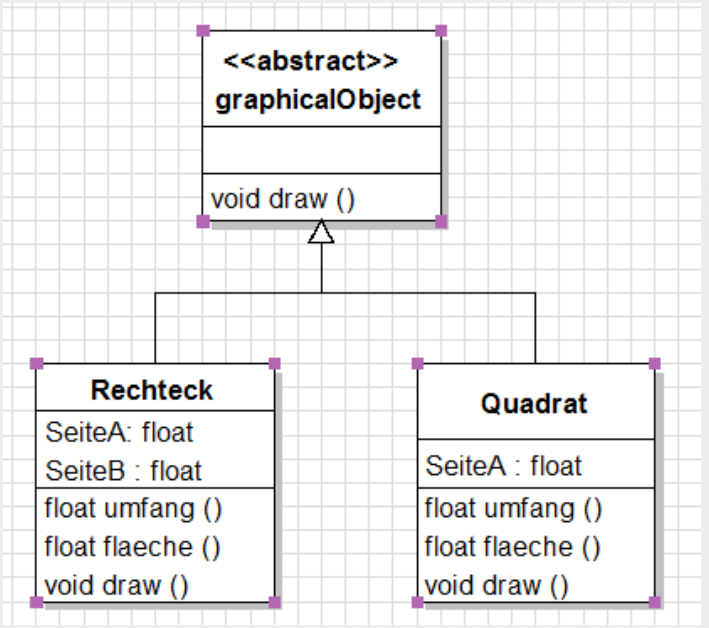
2.



3.



4.



Bewertung von Modellen

- Es gibt häufig **verschiedene Lösungen** für eine Problemstellung.
- Die Entscheidung für einen Entwurf wirkt sich auf **nachfolgende** Design- und Technologie-entscheidungen aus.

Diskussion von Entwürfen ist typische Aufgabe aller **Ingenieurs- und Technikfächer**

2B

Implementierungsmodelle

Implementierungsmodelle

- **Spezifikation** und **Dokumentation** von Softwarekomponenten.
- Zielgruppe: **Software-Entwickler**
- Modelliert auch **Implementierungsdetails** (z. B. View- und Control-Klassen)
- Verfasst in der Notation der **Programmiersprache**
  - Attribute (z. B. `protected int zahl1`)
  - Methodendeklaration (z. B. `public int setZahl1(int wert)`)

Besonderes Augenmerk gilt Softwareschnittstellen.

Anwendung im Unterricht

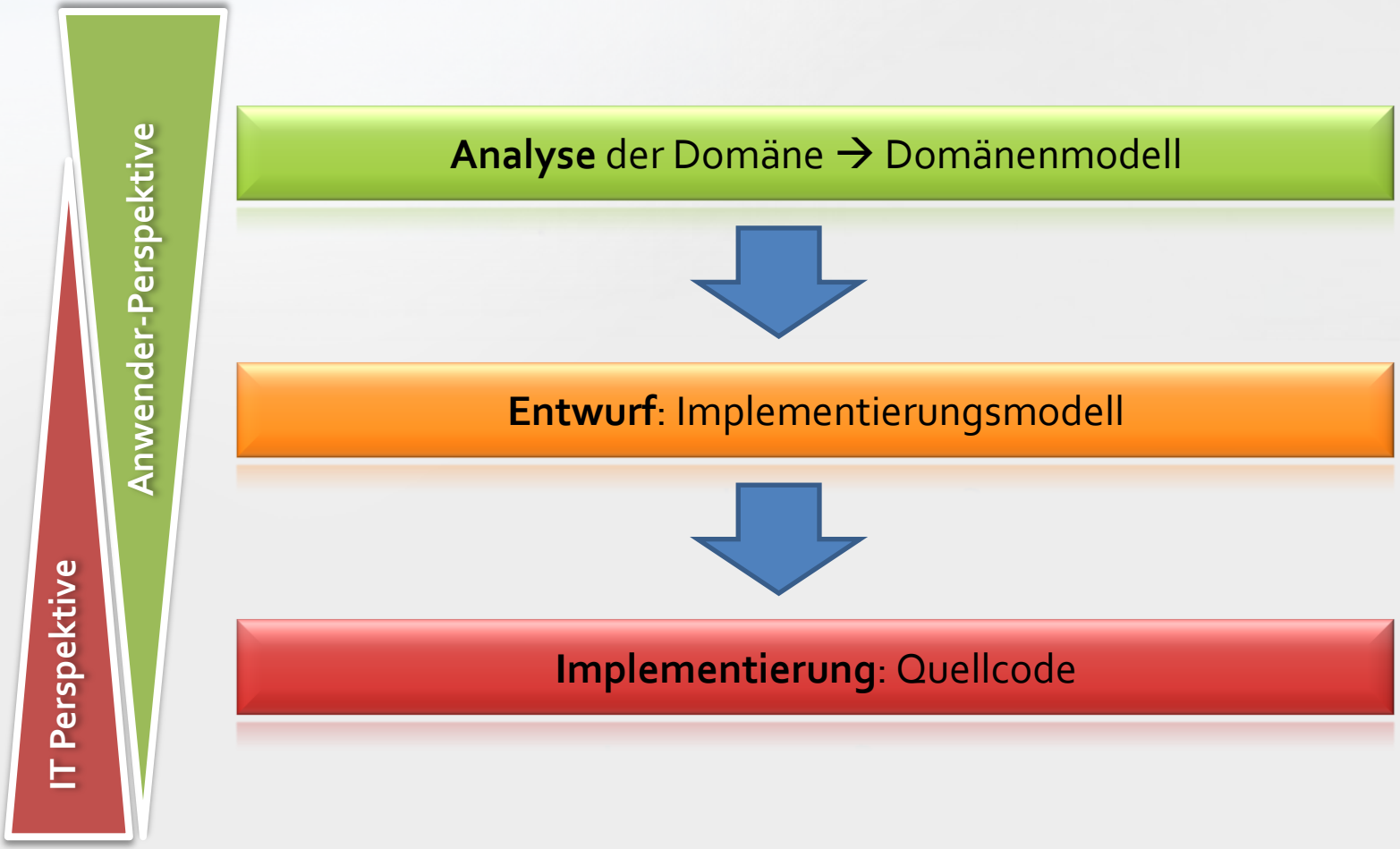
- **Übersetzungsaufgaben:**
  Klassendiagramm -> Code.
- **Ergänzungsaufgaben:**
  Klassendiagramme dokumentieren den bereitgestellten Code.
- **Verteilte Entwicklung:**
  Klassendiagramme spezifizieren die Software-schnittstellen. (z. B. elektronischer Spieler)

Herausforderung:
Assoziationen in Programmierkonzepte übersetzen.

2c

Software-Entwicklung

Software-Entwicklung: Durchgängige Nutzung der Diagramme



Software-Entwicklung im Unterricht

- Modellierung ermöglichen **schülerzentrierten** Zugang zu den Phasen der Softwareentwicklung
- Klassendiagramme sind **intuitiver** als (nur) Pflichten- und Lastenheft.
- **Durchgängige Beispiele** sind möglich – z. B.
  - Freundesnetzwerk („Class Book“)
  - Stundenplanverwaltung
  - Pizzalieferdienst
  - Allgemein: „Datengetriebene“ Anwendungen

Zusammenfassung, Tipps und Tools

UML-Klassendiagramme

- Dienen **statischem Entwurf** (Komponenten, Daten)
  (in Abgrenzung zu dynamischem Entwurf: Prozesse)
- Fokus auf **Klassen** (OO-Sprachen)
  bzw. **Module** (nicht oo Sprachen)
- Software-Entwicklung: **Durchgängige Verwendung** möglich (Domänenmodell, Implementierungsmodell, Code)

Tipps

- Klassendiagramme **als Werkzeug** im Unterricht nutzen (nicht nur Klassendiagramme unterrichten)
- Verdeutlichen, dass man – je nach Zweck des Modells – **unterschiedlich präzise** (und damit durchgängig) modellieren kann.
- Modellieren ist **Programmieren im Großen**: Man kann Software modellieren, die man im Unterricht sinnvoll nicht programmieren könnte.

Tools für Analyse und Design

Modellierungswerkzeuge

- Microsoft Visio
- yEd
- Violet UML Editor

Modellierungswerkzeuge +Entwicklungsumgebung

- BlueJ
- JavaEditor
- AmaterasUML/Eclipse

Vielen Dank!

Fragen?

Ergänzungen / Ausblick

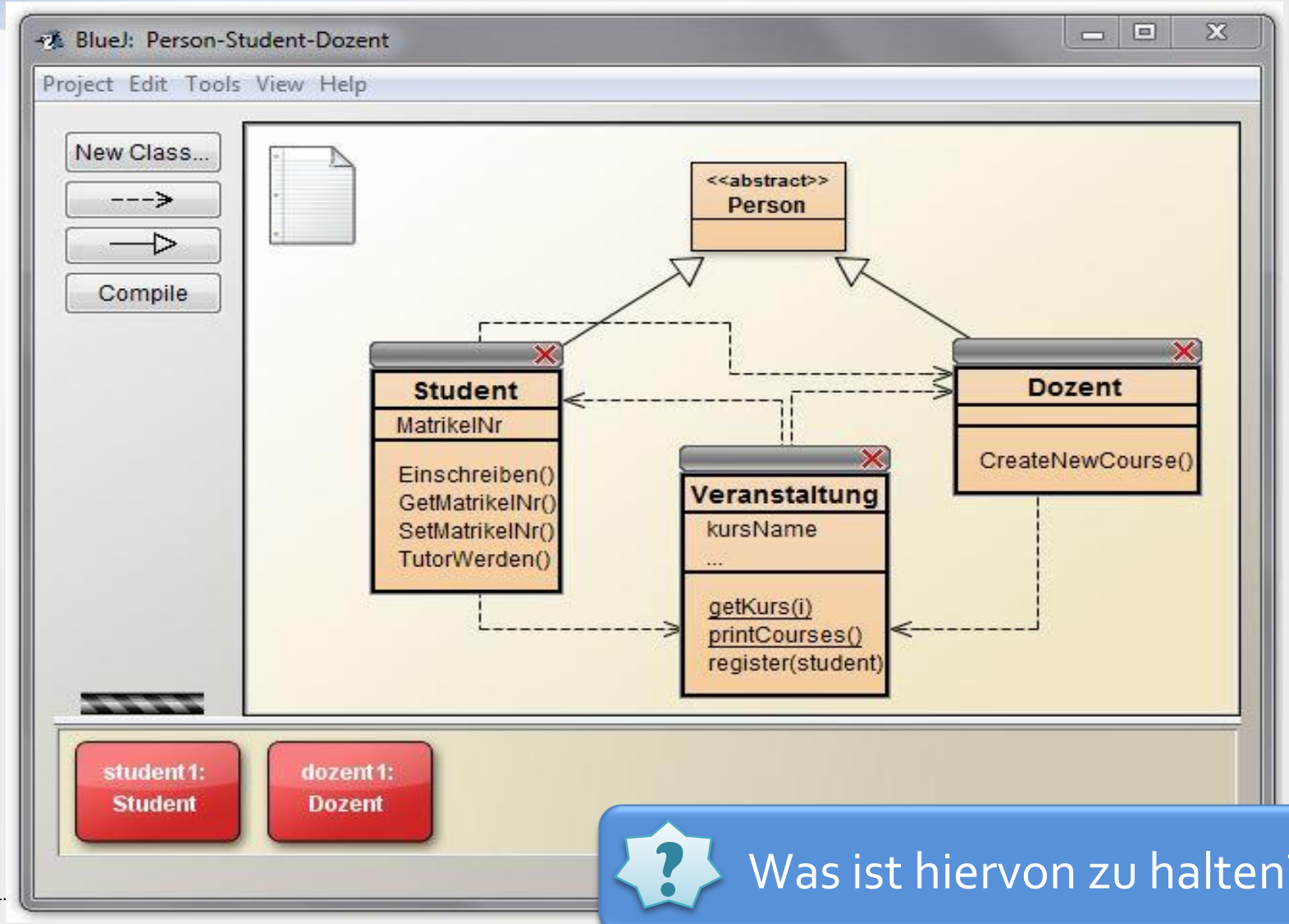
Beispiel / Übung

Modelle vergleichen:

Erstelle ein UML-Klassendiagramm für die Mitgliederverwaltung von Sportvereinen. Modelliere die Klassen `Verein`, `Person`, `Mitglied` und `Vorstandsmitglied` und verbinde sie sinnvoll mit Assoziationen. Gibt es unterschiedliche Wege, das Modell korrekt zu erstellen?

Aus Klassendiagrammen lernen

Beispiel „Delegationsproblematik“



Was ist hiervon zu halten?

… dann doch besser so