Hausaufgabe 6

Tim Wende

19. November 2021

Veranstaltungsverwaltungssystem

Gegeben sei folgende Beschreibung für ein Veranstaltungsverwaltungssystem:
Personen haben Zeichenketten als `Name`. Studierende sind Personen, erben also die Eigenschaften von Person, haben aber zusätzlich eine ganzzahlige `Matrikelnummer` und nehmen an beliebig vielen Veranstaltungen teil. Eine Veranstaltung hat potentiell beliebig viele Teilnehmende, wird aber von einem, zwei oder drei MitarbeiterInnen betreut. Eine Veranstaltung hat eine `Veranstaltungsnummer` und einen `Titel`. Seminare und Vorlesungen sind spezielle Veranstaltungen. Ein Seminar hat eine begrenzte `Anzahl an Plätzen`, für eine Vorlesung wird eine Klausur angeboten oder nicht. MitarbeiterInnen sind Personen und betreuen eine bis fünf Veranstaltungen und haben eine `Personalnummer`. ProfessorInnen und AssistentInnen sind Mitarbeitende. AssistentInnen sind bei genau einem/r ProfessorIn beschäftigt und haben eine bestimmt `Finanzierung` (Zeichenkette). Ein/e ProfessorIn hat ein `Lehrgebiet` (Zeichenkette), beschäftigt beliebig viele AssistentInnen und ist InhaberIn von genau einem Lehrstuhl. Ein Lehrstuhl hat eine `Bezeichnung` und genau einen/e ProfessorIn als InhaberIn.
Erstellen Sie anhand der obigen Beschreibung ein **Klassendiagramm**. Ihr Diagramm sollte folgende Punkte beinhalten:

- **Generalisierungsbeziehungen**,
- **Assoziationen** mit Assoziationsnamen und Leserichtung,
- **Multiplizitäten** sowie
- **Attributnamen** und (sinnvolle) –**typen**.

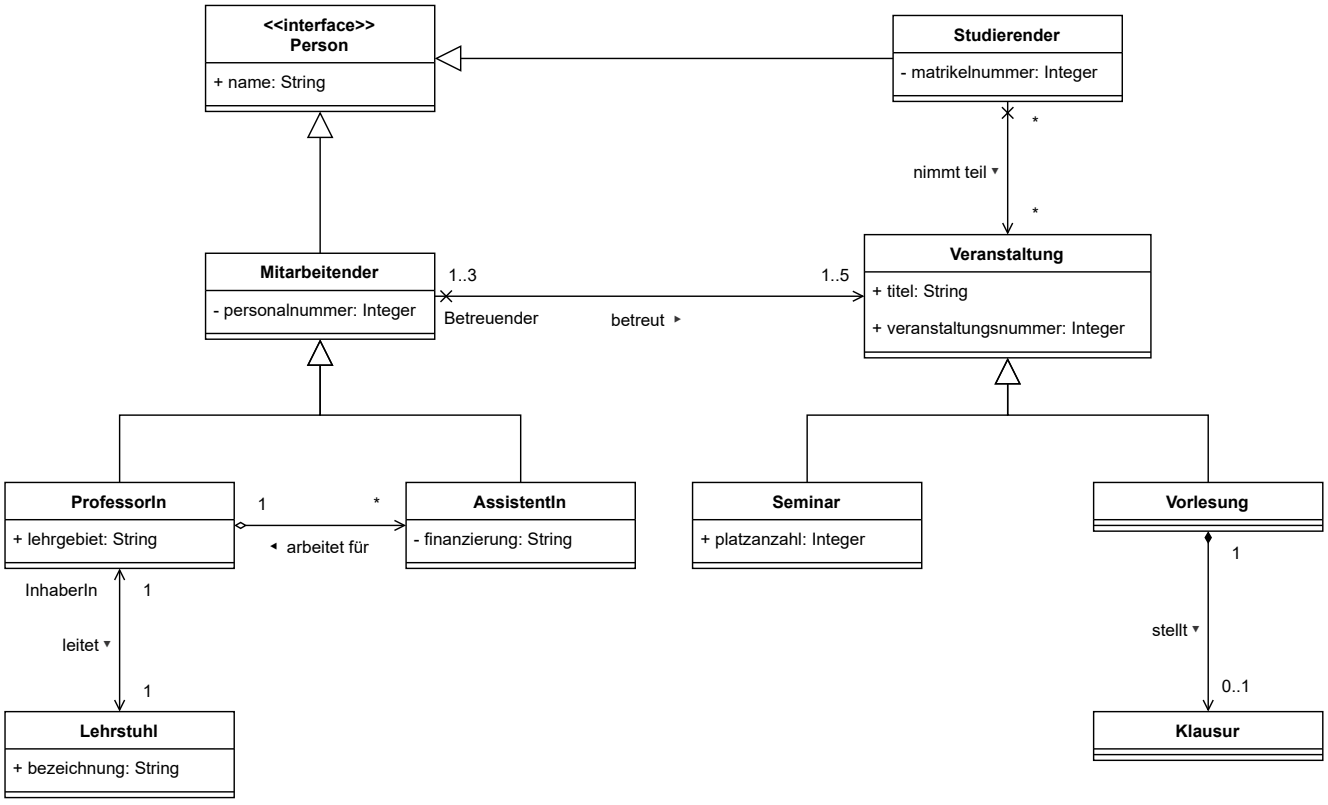

Abbildung 1: `class_diagram`

Finden Sie jeweils ein Beispiel, bei dem eine Aggregations- und eine Kompositionsbeziehung sinnvoll ist. Erläutern Sie kurz den Unterschied zwischen Aggretation und Komposition anhand des Bespiels.

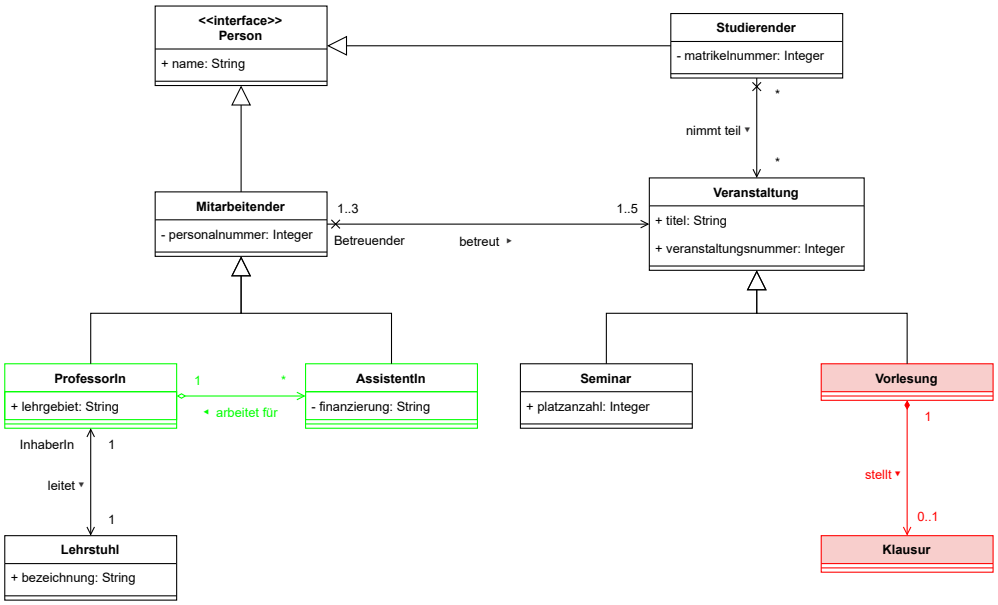

Abbildung 2: `class_diagram_diff`

Als Beispiel für die Aggregation habe ich die Beziehung zwischen ProfessorIn und AssistentIn gewählt. Im Gegensatz dazu steht die Komposition, welche durch die Beziehung zwischen Vorlesung und Klausur verkörpert wird.

Einleitend fass Wikipedia es sehr gut zusammen (stark gekürzt):
Aggregation:
„Eine exakte Definition wird in der UML2 nicht gegeben [...]. Ein konkreter Nutzen lässt sich z. B. ableiten, indem man einem Ende einer Assoziation eine besondere Betonung zukommen lässt“
Komposition:
Der Unterschied zur Aggregation ist, dass ein Objekt, das als Ganzes Teile enthält, für die Existenz der Teile verantwortlich ist.
Also anschaulich:

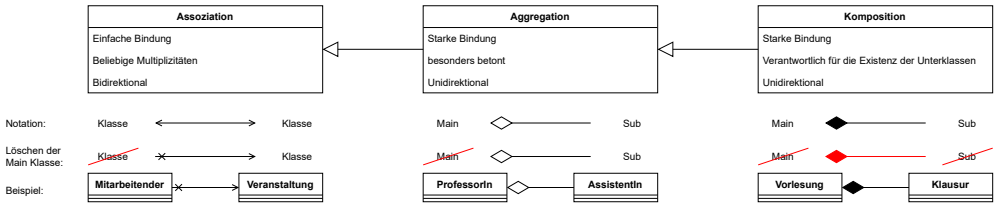

Abbildung 3: `class_diagram_chart`

Um diese Informationen auf unser Beispiel anzuwenden (bzg. der Klassen; nicht der Objekte):
Ein/e AssistentIn ist besonders *betont abhängig* von einem/einer ProfessorIn. Diese Bindung ist beispielsweise stärker, als die Bindung eines Studierenden zu einer Veranstaltung, jedoch ist der/die ProfessorIn nicht verantwortlich für die Existenz eines/einer AssistentIn.
Die Klausur existiert jedoch ausschließlich, wenn die Vorlesung existiert. Fällt diese Weg, wird das Objekt Klausur automatisch entfernt.

Anmerkungen zum Diagramm:

- Ich gehe davon aus, dass eine Klausur für nur eine Vorlesung genutzt werden kann. Da Dies nicht so genau aus der Aufgabenstellung erkennbar ist, sei dies hier erwähnt.
- Die Multiplizität am Pfeil `nimmt teil` bei Studierender könnte auch mit Hilfe des Attributs `platzanzahl` aus Seminar beschrieben werden. So könnte (um bei Java zu bleiben):
  `(Veranstaltung.isClass(Seminar)) ? ((Seminar) Veranstaltung).platzanzahl : *`
  die Genauigkeit der Obergrenze erhöhen.

Kohäsion und Kopplung

1. Beschreiben Sie in eigenen Worten das Prinzip der Kohäsion und Kopplung in der Implementierung von Software-Projekten.

   Die Bindung verschiedener Objekte (Klassen/ Methoden/ ganzen Modulen/ ...), welche durch gegenseitige Methodenaufrufe oder Referenzen entsteht, wird Kopplung genannt. Die Dichte bzw. Anzahl dieser Bindungen wird durch das Maß der Kohäsion beschrieben. Dieses Maß beschreibt den logischen Zusammenhang der jeweiligen Kopplung. So geht das eine nie ohne das andere einher. Man möchte die paketintern Kohäsion der Bindung also so hoch wie möglich halten, und die Bindung der verschiedenen Pakete so gering wie möglich.

2. Warum sind hohe Kohäsion und lose Kopplung der niedrigen Kohäsion und enger Kopplung vorzuziehen?

   Wenn man seinen Code in Pakete aufteilt, möchte man paketintern einen großen logischen Zusammenhang. Zusätzlich möchte man so wenige Querverbindungen zwischen den Paketen wie möglich erstellen, da man sonst in Gefahr läuft, dass jede Klasse in irgend einer Art und Weise abhänhig von einer Zweiten ist. Wenn man zusätzlich diese bereits reduzierten Querverbindungen auf eine Schnittstelle begrenzt, kann man „hinter“ dieser „Proxy“-Klasse den Code beliebig ändern, ohne, dass man andere Klassen ebenfalls ändern muss. Die einzige Klasse, welche beeinflusst wird, ist die „Proxy“-Klasse. So hat man nur eine (ver-)Bindung und in den Paketen eine hohe Kohäsion.

3. Geben Sie ein **eigenes** Beispiel in Form eines Klassendiagrammes für hohe Kohäsion und lose Kopplung an.

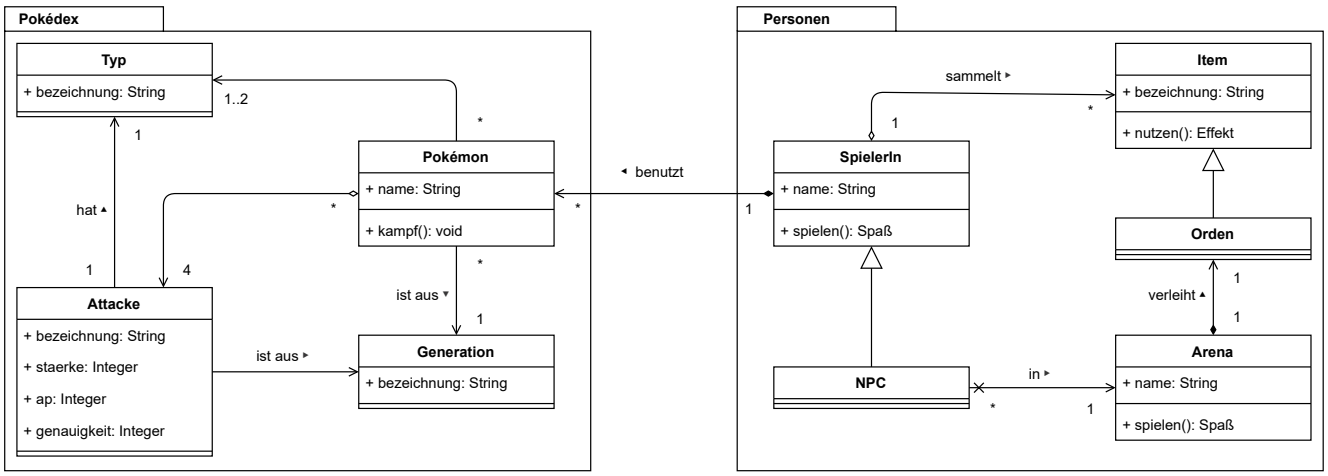

Abbildung 4: `class_diagram_example`