1. Komposition und Aggregation

Beziehungsarten

Komposition und Aggregation repräsentieren jeweils ein Ganzes und seine Teile. Sie ergänzen damit die Assoziationen resp. Beziehungsarten, welche Sie bereits aus der LU03 kennen:

Assoziation	**Beispiel (siehe Abb. 1)**
Allgemein	Mitarbeiter - Firma
Einseitig	Geldbeutel - Mensch
Zweiseitig	Kunde - Konto
Komposition	Blume - Blüte, Wurzel
Aggregation	Lampe - Fassung, Leuchtkörper

Abb. 6.1: Beispiele von verschiednenen Assoziationen

Teile-Ganzes Beziehung

1. Komposition: Die Einzelteile sind **existenziell abhängig** vom Ganzen. Wird eine Blume in ihre Einzelteile zerlegt, kann keine neue Blume daraus gebildet werden. Dasselbe gilt für einen Raum ohne Gebäude.
2. Aggregation: Die Einzelteile sind **auch alleine "lebensfähig"**. Ein Leuchtkörper kann der einen Lampe entnommen und in einer anderen Lampe verwendet werden. Ein Student existiert unabhängig von einer Vorlesung, kann mehrere Vorlesungen besuchen und ist dennoch jeweils Teil davon. Ein Rad oder Motor kann aus einem Auto ausgebaut und in einem anderen wieder verbaut werden.

Abb. 6.2: Beispiele für Komposition und Aggregation

Notation

- Die Raute gehört auf die Seite des Ganzen.
- Der Pfeil auf der Seite der Teile ist optional (siehe Blüte und Leuchtkörpen in Abb. 1).
- Er sollte jedoch entweder für alle Teile gezeichnet oder aber konsequent weggelassen werden (d.h. nicht so machen wie in Abb. 1)

Was ist eine Aggregation resp. Aggregat?

Aggregation ist der Sonderfall einer gerichteten Assoziation. Sie drückt ein starkes Verhältnis von zwei an sich **selbständigen Objekten** aus, von denen eines **Teil des** anderen ist.

Abb. 6.3: Umsetzung Aggregation (d.h. eigenständige Referenzen und zufügen der Komponenten über set Methode resp. Property)

Eigenschaften

- Komponente kann (zusätzlich) einem anderen Aggregat derselben Klasse zugeordnet oder einer anderen Klasse zugeordnet sein
- Komponente existiert unabhängig vom Aggregat

Was ist eine Komposition?

Die Spezialform eines Aggregats, welches wie bereits erwähnt aus **existenziell abhängigen** Komponenten besteht, nennt man **Komposition**. Sie wird auch als “echte Aggregation” bezeichnet und übernimmt die dynamische Weiterleitung von Botschaften an ihre Komponenten.

Abb. 6.4: Umsetzung Komposition (d.h. keine eigenständige Referenzen und Objekte im Konstruktor erzeugen)

Eigenschaften

- Erzeugung der Komposition erzeugt auch deren Komponenten
- Löschung der Komposition löscht auch deren Komponenten

Rückblick Delegation

In der LU05 haben Sie zum Thema *Delegation* gelernt, dass die Klasse `Eck` vier Instanzen der Klasse `Punkt` “aggregiert”. Ein Punkt ist Teil eines Ecks. Da der Punkt aber „sein“ Eck nicht kennt, handelt es sich hierbei um eine gerichtete, einseitige Beziehung mit einer entsprechenden Kardinalität. Weil hier ein `Punkt` ohne `Eck` wenig Sinn macht und darum existenziell von `Eck` abhängig ist, handelt es sich bei dieser Teile-Ganzes Beziehung um eine Komposition.

Abb. 6.5: Komposition eines Vierecks

Quellen:

- Assoziation UML
- Referenzbeziehungen
- OOP by René Probst

From:
https://wiki.bzz.ch/ - **BZZ - Modulwiki**

Permanent link:
https://wiki.bzz.ch/modul/archiv/m320/learningunits/lu98/theorie/lu6-kapitel_1?rev=1788091603

Last update: **2026/08/30 14:06**