LANDESBERUFSSCHULE 4 SALZBURG

Informatik

UML-Klassendiagramme

LBS 4

Dieses Skript dient als zusätzliche Lernunterlage für Informatik

Inhalt

UML

Die Unified Modeling Language (UML) ist heute die verbreitetste Notation, um Softwaresysteme zu analysieren und zu entwerfen. Allerdings ersetzt UML nicht eine Vorgehensweise bei Softwareprojekten. (V-Modell, Spiralmodell, ….)

UML gliedert sich in 3 Gruppen und 14 verschiedenen Diagrammtypen auf. Nachstehend sind alle Arten aufgeführt.

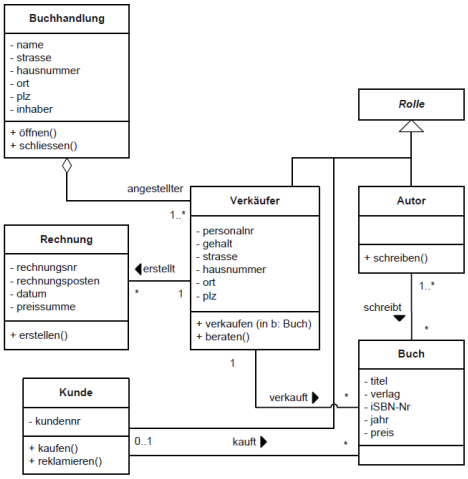
Strukturdiagramme	Verhaltensdiagramme	Interaktionsdiagramme
Klassendiagramm	Use-Case-Diagramm	Sequenzdiagramm
Objektdiagramm	Aktivitätsdiagramm	Kommunikationsdiagramm
Paketdiagramm	Zustandsautomat	Timingdiagramm
Kompositionsstrukturdiagramm		Interaktionsübersichtsdiagramm
Komponentendiagramm		
Verteilungsdiagramm		
Profildiagramm		

Mit diesen Diagrammen lassen sich alle mögliche Situationen in jeder Entwicklungsstufe und Komplexität darstellen.

Bei der Softwareentwicklung ist es wichtig das Konzept vom Kunden und Entwickler auf Papier zu bringen. Mit diesem Vorgang wird das Konzept für andere sichtbar und bringt Klarheit über die Idee und die Chancen des Projektes. Nicht jeder Diagrammtyp ist für ein Projekt notwendig oder möglich.

Zentrales Konzept objektorientierter Modellierung sind die Objekte bzw. deren Abstraktion zu Klassen. Letztere werden mit Klassendiagrammen modelliert. Dieser Diagrammtyp stellt das „Innenleben“ der Objekte, d. h. ihre Attribute und Operationen sowie ihre Beziehungen nach außen – Generalisierungs-, Assoziations- und Abhängigkeitsbeziehungen – dar.

Wir betrachten nur das Klassendiagramm mit den jeweiligen Eigenschaften und Beziehungen. Es beschreibt die Struktur des abzubildenden Systems.



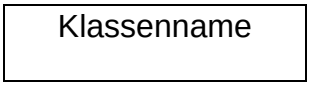
Klassendiagramme finden in allen Projektphasen Anwendung. Sie können Ihnen schon in den frühen Analyseschritten als erste Skizze von vorgefundenen Zusammenhängen dienen und bis zur Codierung eines Softwaresystems begleiten.

Zwangsläufig verändern sich die eingesetzten Formen des Klassendiagramms während des Projektverlaufs, so dass eigentlich nicht von „dem Klassendiagramm“ gesprochen werden kann. Im Grunde bietet die UML mit den Diagrammelementen des Klassendiagramms eine Auswahl an Konstrukten an, die in allen Modellierungsphasen genutzt werden können.

Notationselemente von Klassen:

Klasse:

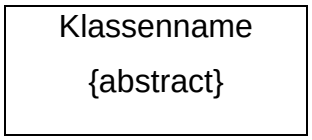
Eine Klasse wird durch ein Rechteck mit durchgezogener Linie dargestellt, der Name der Klasse steht, in Fettdruck und mittig, innerhalb des Rechtecks.



Klassen sind das zentrale Element in einem Klassendiagramm. Eine Klasse beschreibt eine Menge von Objekten mit gemeinsamer Semantik, gemeinsamen Eigenschaften und gemeinsamem Verhalten. Die Eigenschaften werden durch Attribute und das Verhalten durch Operationen abgebildet.

Abstrakte Klassen:

Abstrakte Klassen sind spezielle Klassen die mindestens eine abstrakte Methode beinhalten. Abstrakte Klassen können nicht Instanziiert werden. Sie dienen als Basis bei der Vererbung.



Attribute:

Attribute werden im zweiten Abschnitt des Klassensymbols, linksbündig, kleingeschrieben und untereinander angeordnet.

Attribute (Attributes) repräsentieren die strukturellen Eigenschaften von Klassen. Sie bilden somit das Datengerüst. Jedem Attribut wird zur Laufzeit, d. h. im Objekt der Klasse, ein entsprechender Wert zugewiesen – es wird befüllt.

Attributdeklaration:

Ein Attribut muss gemäß UML nur durch seinen Namen spezifiziert werden. Der Name muss innerhalb der Klasse eindeutig sein. Darüber hinaus können Sie die Attribute noch weiter spezifizieren, so dass die allgemeine Syntax der Attributdeklaration wie folgt lautet:

[sichtbarkeit] [/] name [: Typ] [[Multiplizität]] [= Vorgabewert] [{eigenschaftswert [, eigenschaftswert]*}]

[sichtbarkeit]: definiert die Sichtbarkeit der ein Attribut unterworfen ist. Mögliche Werte sind: private(-), public(+),protected(#)

[/]: bezeichnet ein abgeleitetes Attribut

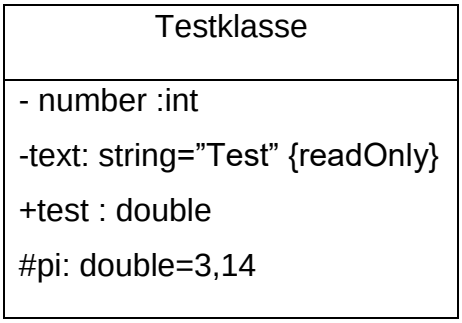
Name: Name des Attributs

[: Typ] : Datentyp des Attributs

[Multiplizität]: legt die Anzahl der ablegbaren Instanzen fest

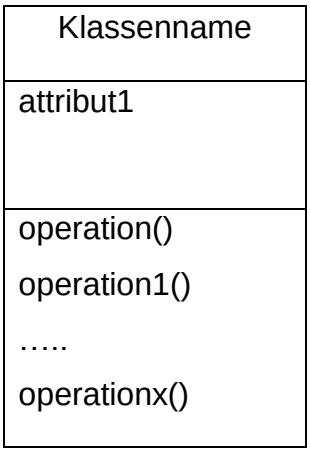
[= Vorgabewert] vergibt Standardwerte an Attribute

[{eigenschaftswert [, eigenschaftswert]*}] : in den geschweiften Klammern werden Charakteristika wie readOnly, ordered,… festgelegt



Operationen (Methoden)

Eine Operation wird immer durch ihren Namen sowie weitere Angaben spezifiziert. Mehrere Operationen werden dabei vertikal untereinander linksbündig, typischerweise im dritten Abschnitt/Rechteckelement des Klassensymbols angeschrieben. Abstrakte Operationen sind kursiv gesetzt;



Operationen werden eingesetzt, um das Verhalten von Objekten zu beschreiben. Alle Operationen einer Klasse zusammengenommen definieren die Möglichkeiten zur Interaktion mit Objekten dieser Klasse. Gleichzeitig stellen Operationen die einzige Möglichkeit dar, Zustandsänderungen eines Objekts herbeizuführen.

Deklaration:

Nachstehend sind die wichtigsten Parameter für Operationen angeführt.

[Sichtbarkeit] name ([Parameterliste]) [: [Rückgabetyp]

[sichtbarkeit]: definiert die Sichtbarkeit der ein Attribut unterworfen ist. Mögliche Werte sind: private(-), public(+),protected(#)

Name: Name des Attributs

([Parameterliste]) Parameter die beim Aufruf übergeben werden

[Rückgabetyp] Datentyp des Rückgabewertes

Klassenname

attribut1

+op1()
#op2(param : int=5)
-op3(param1 : int) :int

Beziehungen zwischen Klassen:

Multiplizität:

Die Multiplizität beschreibt ein Intervall, dass die obere und untere Grenze von Beziehungen zwischen Klassen beschreibt. Es gibt folgende Möglichkeiten:

3..5	zwischen drei und fünf Objekten
0..*, *	beliebig viele Objekte
1	genau ein Objekt
1..*	mindestens 1
1,4,7	eins, vier oder sieben Objekte

Assoziation

Im Klassendiagramm dient die Assoziation zur Darstellung von Beziehungen zwischen Klassen. Wie auch in den anderen UML-Diagrammen wird die binäre Assoziation durch eine durchgezogene Linie dargestellt, die die teilnehmenden Klassen verbindet.

Eine Assoziation beschreibt eine Menge gleichartiger Beziehungen zwischen Klassen. Dabei kann die Beziehung als Weg oder Kommunikationskanal verstanden werden. Die Assoziation definiert eine sehr enge Form der Beziehung zwischen Klassen, die das gegenseitige Zugreifen auf Elemente der Klasse (Attribute und Operationen) ermöglicht.

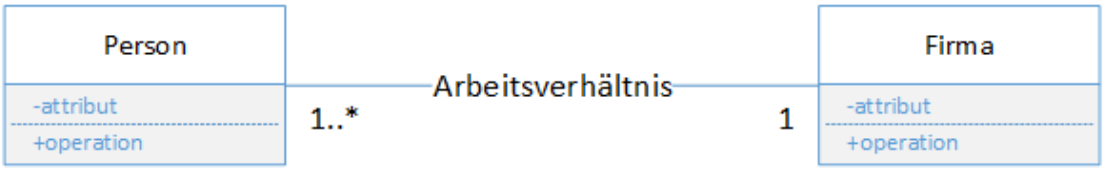


Abbildung 1: Assoziation

Abbildung 1 verbindet die beiden Klassen Person und Firma mit einem Arbeitsverhältnis. Die Angabe 1 am Assoziationsende (Firma) definiert, dass eine Person mit genau einer Firma im Arbeitsverhältnis steht.

Umgekehrt stehen mit der Firma mindestens eine, aber auch beliebige viele Personen in einem Arbeitsverhältnis. Dies wird durch die Angabe 1..* gefordert.

Im Beispiel muss ein Objekt der Klasse Person mit genau einem Objekt der Klasse Firma über einen Link verbunden werden.

Umgekehrt darf ein Objekt der Klasse Firma mit mindestens einem und maximal beliebig vielen (1..*) Objekten der Klasse Person in Beziehung stehen.

Aggregation

Eine Aggregation (im Beispiel die Personengruppe) drückt eine „Teile-Ganzes-Beziehung“ aus. Die aggregierten Instanzen einer Klasse sind dabei Teil eines Ganzen, das durch die Klasse am anderen Beziehungsende repräsentiert wird.

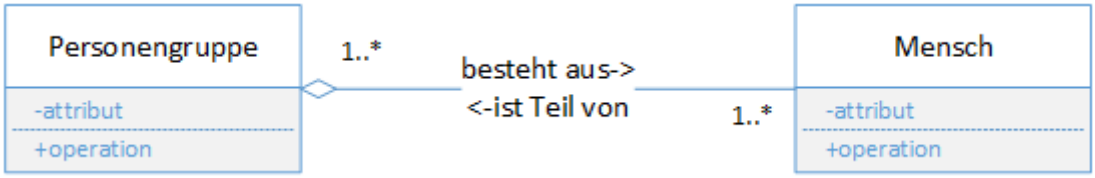


Abbildung 2: Aggregation

Demnach ist die Aggregation zu lesen als „Personengruppe besteht aus Menschen“. Dasjenige Assoziationsende, welches mit dem leeren Raute versehen ist, wird als „Ganzes“ aufgefasst, seine „Teile“ befinden sich als Klasse am anderen Assoziationsende. Umgekehrt bist der“ Mensch Teil von einer Personengruppe“.

Hinsichtlich der Multiplizitäten gilt für die Aggregation das Gleiche wie für die Assoziation allgemein: Sie können prinzipiell alle denkbaren Multiplizitäten angeben. Die Multiplizität sagt dann aus, ob das „Teil“ zwischen verschiedenen „Ganzen“ geteilt werden kann oder nicht. Ist die Obergrenze der Multiplizität größer 1, dann kann ein „Teil“ in mehreren „Ganzen“ auftreten.

Komposition

Eine strengere Form des Zusammenhanges wird durch die Komposition definiert. Sie drückt im Gegensatz zur Aggregation die physische Inklusion der Teile im Ganzen aus. Teile und Ganzes bilden eine Einheit, deren Auflösung durchaus die Zerstörung des Ganzen zur Folge haben kann.

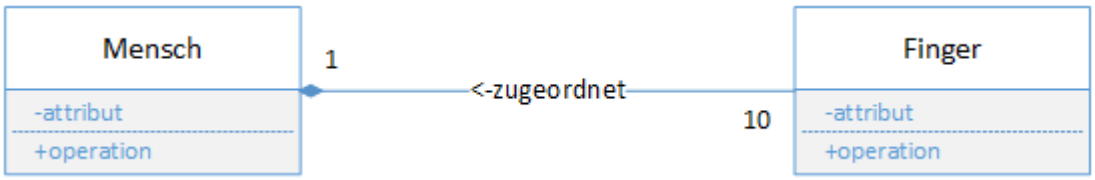
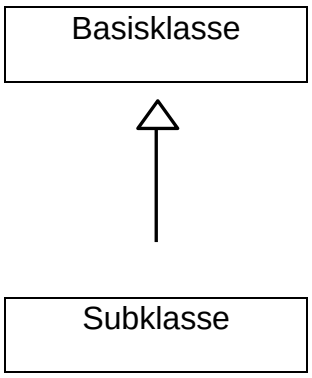


Abbildung 3:Komposition

Im Beispiel besitzt der Mensch 10 Finger, die zusammen eine Einheit bilden. Deshalb gilt hier die verschärfende Einschränkung, dass ein Teil zu einem Zeitpunkt höchstens genau einem Ganzen zugeordnet sein darf. Andernfalls würde es zu wechselseitigen Abhängigkeiten zwischen den Objekten kommen. (1 Finger ist genau einem Menschen zugeordnet.)

Generalisierung:

Die Generalisierungsbeziehung zwischen Klassen wird mit einem Pfeil mit nicht ausgefüllter dreieckiger Pfeilspitze dargestellt. Er zeigt von der spezialisierten Klasse (Pfeilende) hin zum allgemeinen oder generalisierten Klasse (Pfeilspitze).



Die Generalisierung stellt ein zentrales Konzept objektorientierter Modellierung dar. Sie setzt zwei Klassen in Beziehung, so dass eine Klasse eine Verallgemeinerung des anderen darstellt.

Die Umkehrung der Generalisierung ist die Spezialisierung.