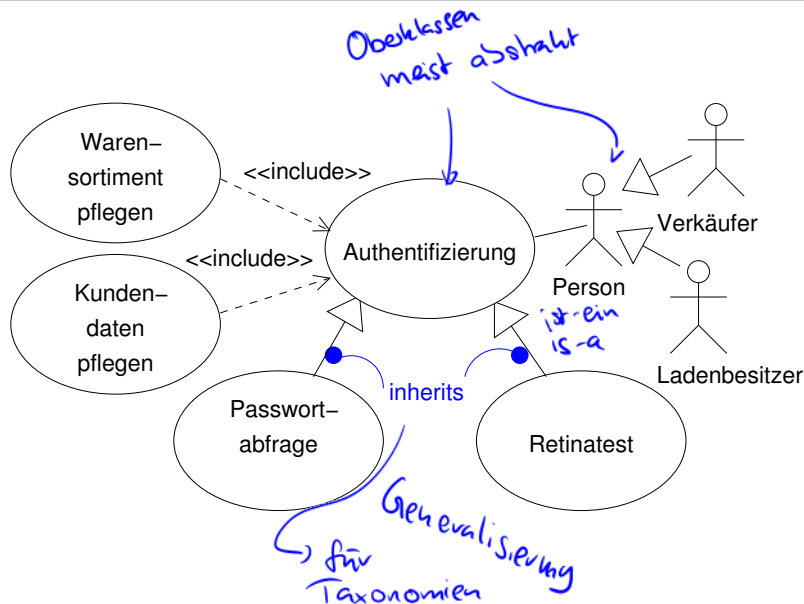
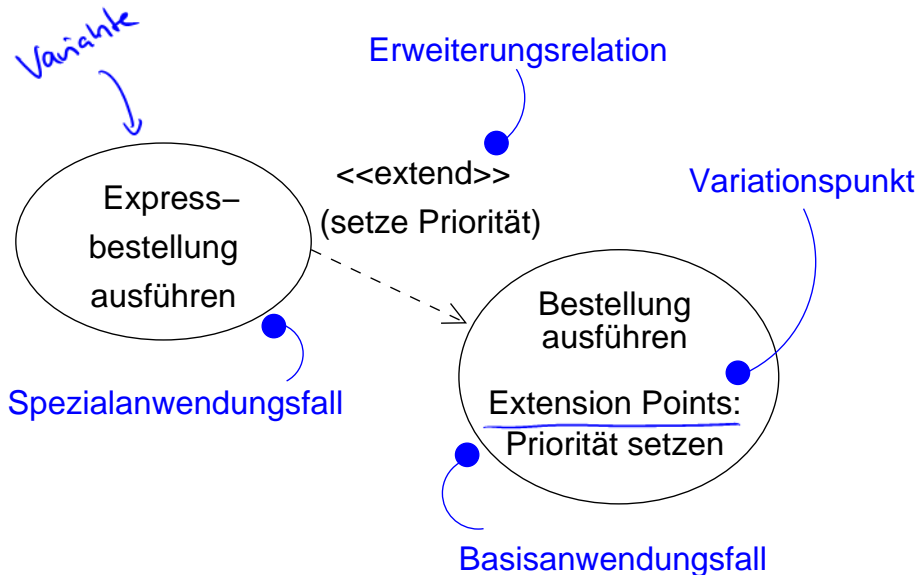


UML-Notation für Anwendungsfälle (OMG)



UML-Notation für Anwendungsfälle (OMG)



Anwendungsfalldiagramme:

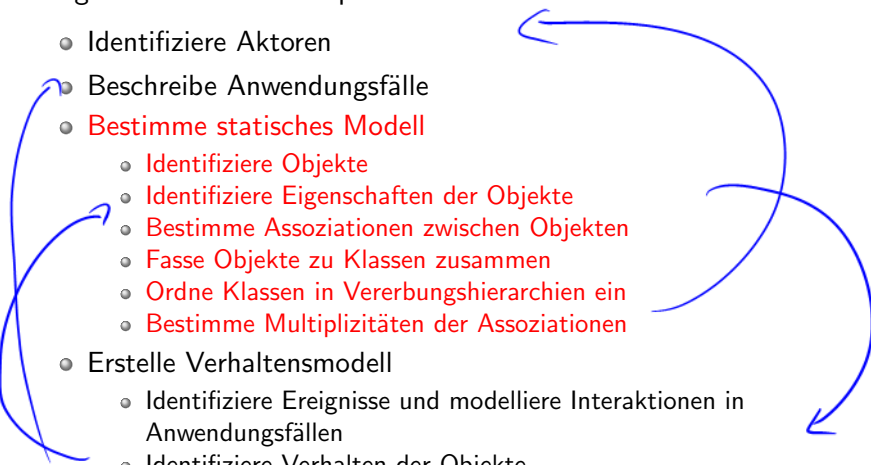
- deklarieren voneinander unabhängige Anwendungsfälle
- beschreiben die Einbettung der Anwendungsfälle in den Systemkontext (externe Akteure)
- beschreiben (konkretisieren) die Anwendungsfälle jedoch nicht
→ folgt später
- sind Ausgangspunkt für die objektorientierte Modellierung
 - statische Eigenschaften (Attribute)
 - dynamische Eigenschaften (Verhalten)

Ausgehend von Geschäftsprozessen...

- Identifiziere Aktoren
- Beschreibe Anwendungsfälle
- Bestimme statisches Modell
 - Identifiziere Objekte
 - Identifiziere Eigenschaften der Objekte
 - Bestimme Assoziationen zwischen Objekten
 - Fasse Objekte zu Klassen zusammen
 - Ordne Klassen in Vererbungshierarchien ein
 - Bestimme Multiplizitäten der Assoziationen
- Erstelle Verhaltensmodell
 - Identifiziere Ereignisse und modelliere Interaktionen in Anwendungsfällen
 - Identifiziere Verhalten der Objekte
 - Beschreibe das Verhalten (Vor- und Nachbedingungen)

Objektorientierte Modellierung

Ausgehend von Geschäftsprozessen...

- Identifiziere Aktoren
 - Beschreibe Anwendungsfälle
 - **Bestimme statisches Modell**
 - **Identifiziere Objekte**
 - **Identifiziere Eigenschaften der Objekte**
 - **Bestimme Assoziationen zwischen Objekten**
 - **Fasse Objekte zu Klassen zusammen**
 - **Ordne Klassen in Vererbungshierarchien ein**
 - **Bestimme Multiplizitäten der Assoziationen**
 - **Erstelle Verhaltensmodell**
 - Identifiziere Ereignisse und modelliere Interaktionen in Anwendungsfällen
 - Identifiziere Verhalten der Objekte
 - Beschreibe das Verhalten (Vor- und Nachbedingungen)
- 

Beispiel Fahrradladen

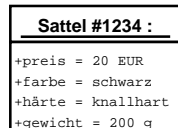
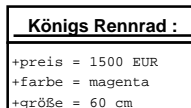
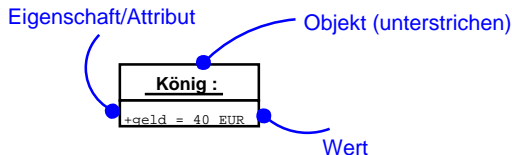
Identifiziere Objekte: Suche nach Substantiven in Anwendungsfällen.

Operation: Einkauf

- **Kunde König** möchte **Sattel** kaufen; besitzt mehrere **Fahrräder**
- ...
- König wählt sein **28"-Renncrad** mit aus; **Rahmenfarbe**: magenta
- ...
- **PDA** nimmt Verbindung mit **Laden-Host-Rechner** auf
- ...
- König lässt nach **Preis** sortieren
- ...
- **Verkäufer Volker** betrachtet **Auswahl** und berät

Anwendungsfall → Objekte

Objekt: „Ding“ (Gegenstand, Entität) mit eigener Identität

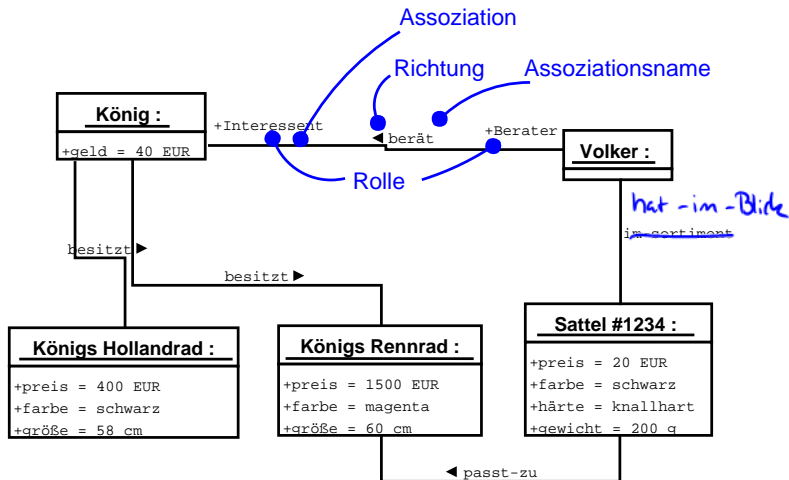


Ausgehend von Geschäftsprozessen...

- Identifiziere Aktoren
- Beschreibe Anwendungsfälle
- Bestimme statisches Modell
 - Identifiziere Objekte
 - Identifiziere Eigenschaften der Objekte
 - **Bestimme Assoziationen zwischen Objekten**
 - Fasse Objekte zu Klassen zusammen
 - Ordne Klassen in Vererbungshierarchien ein
 - Bestimme Multiplizitäten der Assoziationen
- Erstelle Verhaltensmodell
 - Identifiziere Ereignisse und modelliere Interaktionen in Anwendungsfällen
 - Identifiziere Verhalten der Objekte
 - Beschreibe das Verhalten (Vor- und Nachbedingungen)

Anwendungsfall → Objekte

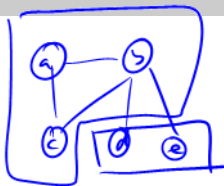
Assoziation: Beziehung zwischen Objekten



Ausgehend von Geschäftsprozessen...

- Identifiziere Aktoren
- Beschreibe Anwendungsfälle
- Bestimme statisches Modell
 - Identifiziere Objekte
 - Identifiziere Eigenschaften der Objekte
 - Bestimme Assoziationen zwischen Objekten
 - Fasse Objekte zu Klassen zusammen
 - Ordne Klassen in Vererbungshierarchien ein
 - Bestimme Multiplizitäten der Assoziationen
- Erstelle Verhaltensmodell
 - Identifiziere Ereignisse und modelliere Interaktionen in Anwendungsfällen
 - Identifiziere Verhalten der Objekte
 - Beschreibe das Verhalten (Vor- und Nachbedingungen)

Objekte versus Klassen



Instanz-Ebene

- **Objektdiagramme** beschreiben statische Zusammenhänge auf der Ebene einzelner, konkreter Dinge

Schema-Ebene

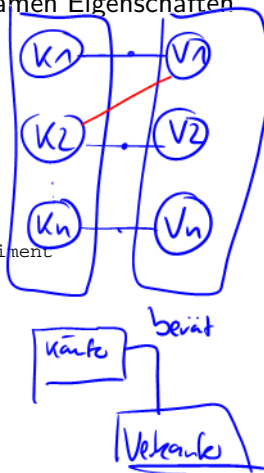
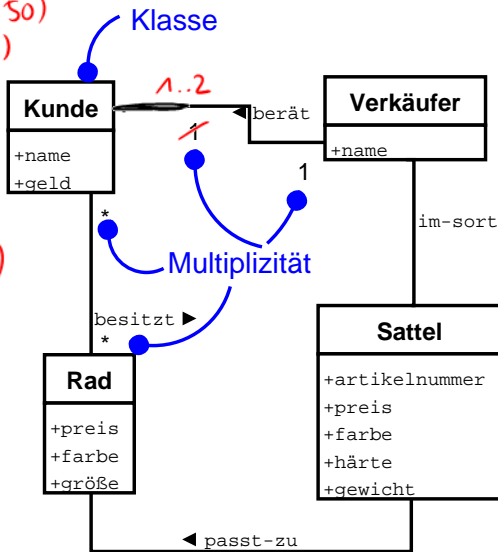
- **Klassendiagramme** beschreiben statische Zusammenhänge unabhängig von Details konkreter Objekte, auf der Ebene mehrerer gleichartiger Dinge

Objekte → Klassen

Klasse: Menge von gleichartigen Objekten mit gemeinsamen Eigenschaften

Kunde (K1, 'h', 50)
Kunde (K2, ...)
Verkäufer (V1, ...)

berät (V1, K1)
berät (V1, K2)



Ausgehend von Geschäftsprozessen...

- Identifiziere Aktoren
- Beschreibe Anwendungsfälle
- Bestimme statisches Modell
 - Identifiziere Objekte
 - Identifiziere Eigenschaften der Objekte
 - Bestimme Assoziationen zwischen Objekten
 - Fasse Objekte zu Klassen zusammen
 - Ordne Klassen in Vererbungshierarchien ein
 - Bestimme Multiplizitäten der Assoziationen
- Erstelle Verhaltensmodell
 - Identifiziere Ereignisse und modelliere Interaktionen in Anwendungsfällen
 - Identifiziere Verhalten der Objekte
 - Beschreibe das Verhalten (Vor- und Nachbedingungen)

- sind spezielle Beziehung auf Schema-Ebene
- beschreiben Beziehungen zwischen einer allgemeineren Klasse (Oberklasse, Superclass) und einer spezielleren Klasse (Unterklasse, Subclass)
- die Unterklasse „erbt“ die Eigenschaften der Oberklasse:
 - Attribute
 - Methoden und deren Aufrufchnittstellen
 - Assoziationen

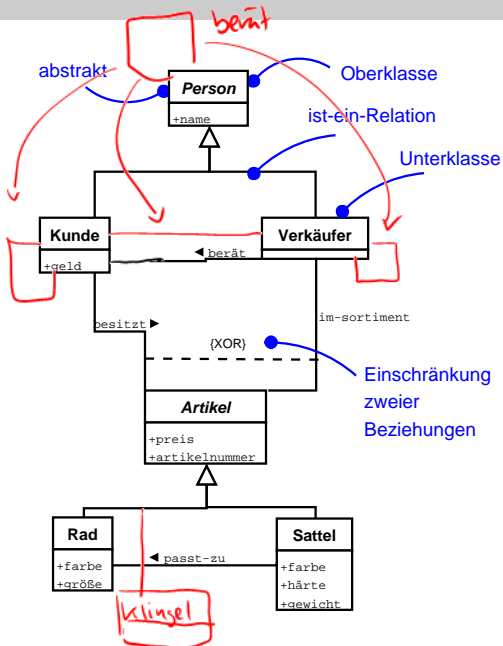
Generalisierungen

- sind spezielle Beziehung auf Schema-Ebene
- beschreiben Beziehungen zwischen einer allgemeineren Klasse (Oberklasse, Superclass) und einer spezielleren Klasse (Unterklasse, Subclass)
- die Unterklasse „erbt“ die Eigenschaften der Oberklasse:
 - Attribute
 - Methoden und deren Aufrufchnittstellen
 - Assoziationen

Unterklassen können

- neue Attribute, Methoden und Assoziationen definieren
- ererbte Attribute, Methoden und Assoziationen redefinieren (überschreiben)
- von mehreren Oberklassen erben (Mehrfachvererbung)

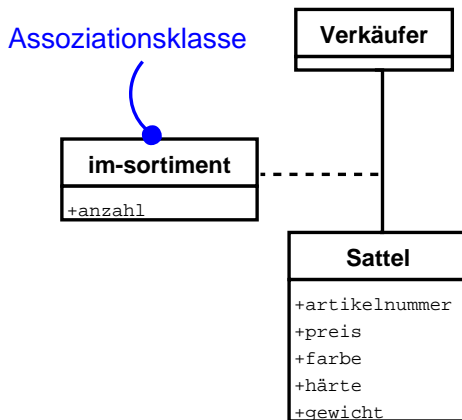
Klassen → Klassenhierarchien



Attributierte Assoziationen

Assoziationsklassen

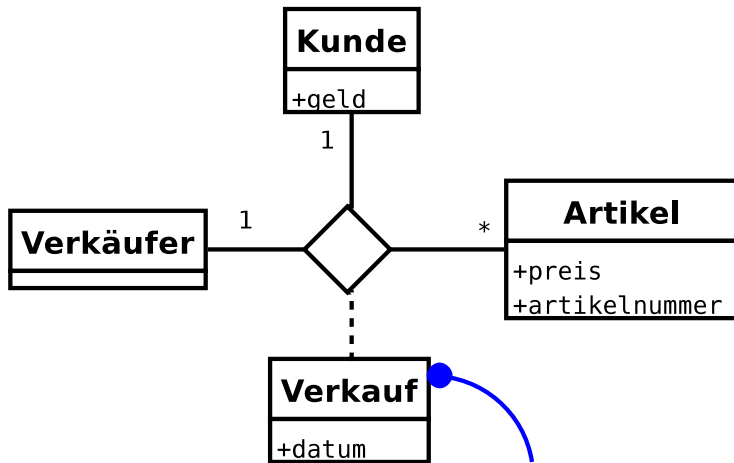
- beschreiben Beziehungen, die gleichzeitig Klassen sind
- werden genutzt, um Assoziationen Attribute zuzuordnen
- können eigene Beziehungen eingehen



Mehrstellige Assoziationen

Assoziationsklassen

- können auch mehrere Klassen in Beziehung setzen



Assoziationsklasse

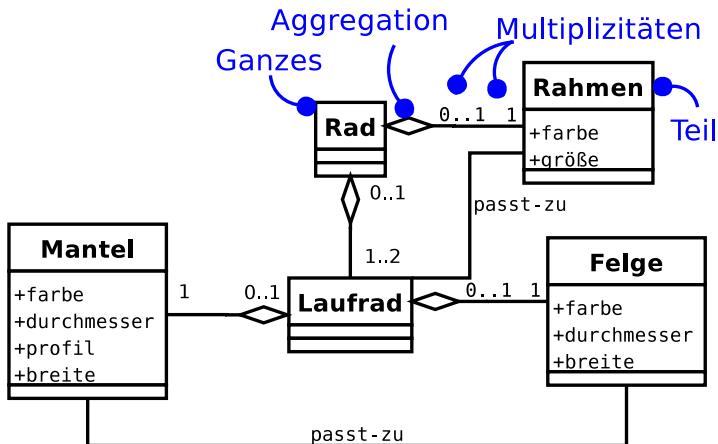
Aufbau von Objekten



Aufbau von Objekten

Aggregation

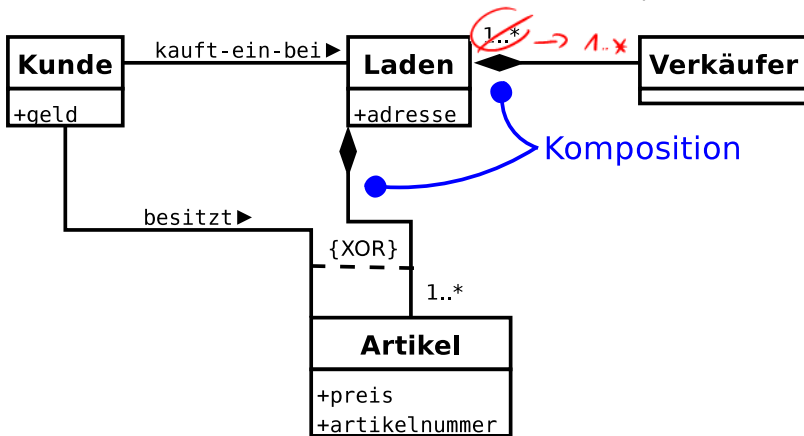
- ist spezielle Assoziation zur Verdeutlichung von „Teil-Ganzes-Beziehungen“
- beschreibt Zusammenfassung von Komponenten zu einem Aggregat



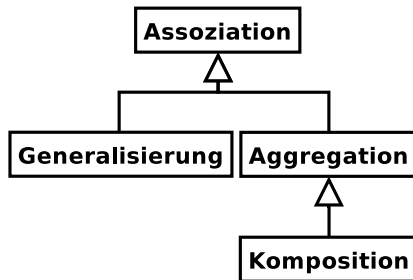
Aufbau von Objekten

Komposition ist spezielle Aggregation:

- Existenz der Komponenten ist an die Existenz des Aggregats gekoppelt,
- jede Komponente gehört zu genau einem Aggregat (strong ownership)



Vergleich der Assoziationstypen

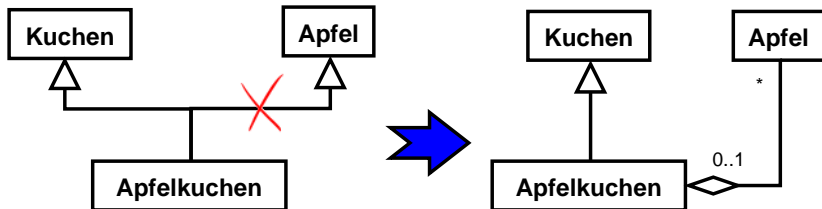


- Vererbung: ist-ein-Relation (Liskovs Substitutionsprinzip erfüllt)
- Aggregation: teil-von-Relation
- Komposition: teil-von-Relation
 - Teil gehört zu genau einem Ganzen
 - Teil existiert nur im Kontext des Ganzen
- allgemeine Assoziation: sonst

Liskovs Substitutionsprinzip (1988; 1994)

Definition

Liskovs Substitutionsprinzip: jede Instanz der Unterklasse kann immer und überall dort eingesetzt werden, wo Instanzen der Oberklasse auftreten.



Beschreibungsinhalt

- statische Systemaspekte
- Beschreibung der wesentlichen, unterscheidbaren Dinge eines Systems und ihrer Beziehungen

zentrale Modellierungskonstrukte:

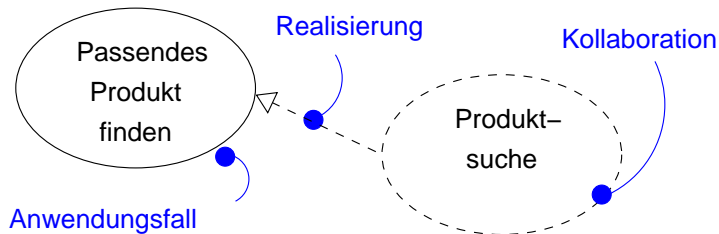
- Klassen
- Assoziationen (Beziehungsklassen)
 - spezielle Assoziationen: Generalisierung und Aggregation/Komposition

Ausgehend von Geschäftsprozessen...

- Identifiziere Aktoren
- Beschreibe Anwendungsfälle
- Bestimme statisches Modell
 - Identifiziere Objekte
 - Identifiziere Eigenschaften der Objekte
 - Bestimme Assoziationen zwischen Objekten
 - Fasse Objekte zu Klassen zusammen
 - Ordne Klassen in Vererbungshierarchien ein
 - Bestimme Multiplizitäten der Assoziationen
- **Erstelle Verhaltensmodell**
 - **Identifiziere Ereignisse und modelliere Interaktionen in Anwendungsfällen**
 - Identifiziere Verhalten der Objekte
 - Beschreibe das Verhalten (Vor- und Nachbedingungen)

- textuelle Szenario-Beschreibungen (siehe oben)
- Interaktionsdiagramme
- Aktivitätsdiagramme
- Zustandsdiagramme

UML-Notation für Anwendungsfälle (OMG)



Beschreibungsinhalt:

- dynamische Systemaspekte
- exemplarische Beschreibung von Interaktionsabfolgen zwischen kollaborierenden Objekten (Inter-Objektverhalten)

zentrale Modellierungskonstrukte:

- Objekte: „Dinge“ mit eigener Identität
- Nachrichten
 - beschreiben den Austausch von Informationen zwischen Objekten zum Auslösen von Aktivitäten
 - sind Signale/Ereignisse oder Methodenaufrufe


implementierung

Anwendung

- Präzisierung von Szenarien (exemplarische Folgen von Aktivitäten)
- Protokollierung des Nachrichtenaustausches
- Erhebung der von einzelnen Objekten bereit gestellten Funktionalität (Dienste, Methoden)

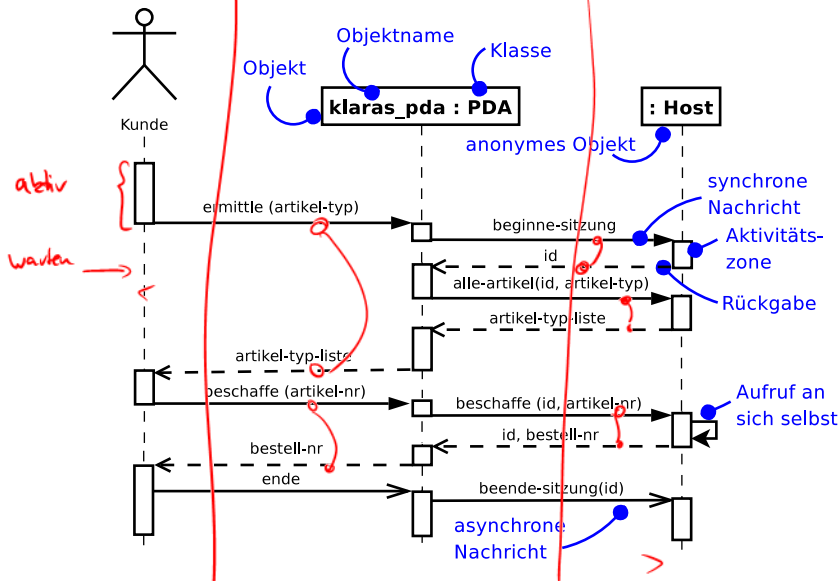
Varianten

- Sequenzdiagramme
 - betonen zeitlichen Ablauf
- Kollaborationsdiagramme
 - betonen Objektstruktur

Grenzen:

- beschreiben Verhalten nur beispielhaft und nicht vollständig

UML-Sequenzdiagramme (OMG)



UML-Kollaborationsdiagramme (OMG)

