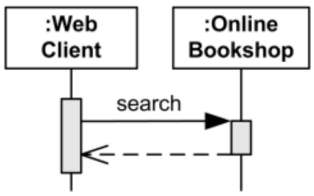
Sequenzdiagramme

- UML Sequenzdiagramme
- Beispiel
- Fortsetzung objektorientierten Analyse
- UML Kommunikationsdiagramme

UML Sequenzdiagramme

Darstellung von Nachrichten*, die zwischen Akteuren und Objekten in begrenzten Zeitrahmen ausgetauscht werden.

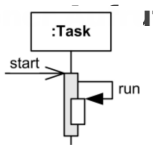
→ zur Beschreibung des Systemverhaltens in Anwendungsfällen

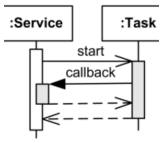


Lebenslinie – Lebenszyklus von (anonymen oder konkreten) Objekten und Akteuren

Nachricht – Darstellung einer einzelne Kommunikation zwischen zwei Lebenslinien

Ausführung – Darstellung Zeitraums, in der Verhalten oder *Aktion ausgeführt*, ein *Signal gesendet* oder *gewartet* wird

→ **Synchr** 

uf – Senden Nachricht und Unterbrechen der Ausführung während *Wartezeit* auf eine 

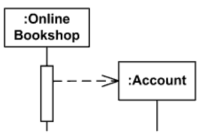
Antwort

→ **Asynchroner Aufruf** – Senden einer Nachricht und *sofortige Fortsetzung* der eigenen Ausführung
→ **Rückgabe** – Senden einer *Antwortnachricht* an das aufrufende Objekt
→ **Selbstdelegation** – Senden von Nachrichten eines Objekts an sich selbst
→ **Callback** – Ausführung, die synchron über eine Rückgabe für den originalen Aufrufer informiert

Benennung von Nachrichten: **Nummer** der Nachricht | **Name** der Methode | **Argumente** | **Rückgabetyp**

```
1: aMessage(num: int, value: double): String
```

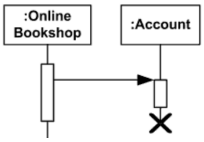
Bedingungen, ***wenn*** Nachricht gesendet:




```
1: (x<0): aMessage(num: int, value: double): String
```

Nachrichten können **wiederholt** gesendet werden: `1: *[while(result<25)]: result=operation()`

Instanziierung – Nachricht zur Erstellung einer Lebenslinie eines Objekts

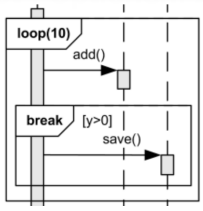


Terminierung – Nachricht zum Beenden einer Lebenslinie eines Objekts

Kombinierte Fragmente

alt : Alternativen: **Auswahl** des auszuführenden Verhaltens

opt : Optionen: **Auswahl** eines einzelnen Operanden



loop . Wiederholungen (Schleifen): **Iteration** im Ausführungsverhalten

break : **Abbrüche** oder Ausnahmen

par : **Parallele** Ausführungen

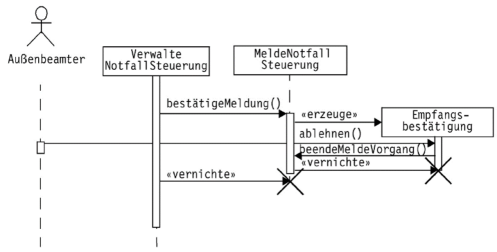
strict : Strikte Sequenzen: **Reihenfolge** einhalten

seq : Schwache Sequenzen: ausführen, **Reihenfolge** egal

Beispiel

Fortsetzung objektorientierten Analyse

Vorgehen: Abbildung der Anwendungsfälle auf Sequenzdiagramme



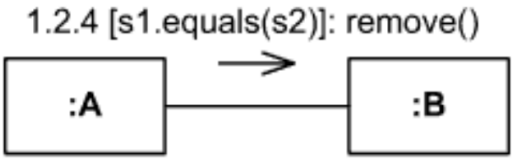
1. Erste Spalte ganz links: Akteure, die den Anwendungsfall veranlassen
2. Zweite Spalte von links: Systemgrenze (boundary object), die mit Akteur interagiert
3. Dritte Spalten von links: das relevante Kontrollobjekt (control object)
    1. Zugriff auf Fachentitäten (entity objects)
    2. Kann andere Kontrollobjekte und Fachentitäten erstellen (create -Nachricht)
    3. Kann mit anderen Kontrollobjekten interagieren

4. Folgende Spalten: Fachentitäten (entity objects)

UML Kommunikationsdiagramme

Darstellung der Interaktionen zwischen Objekten mithilfe sequenzierter Nachrichten

Lebenslinie – Darstellung individueller Teilnehmer der Interaktion




```
sequence-term ::= [integer [name]] [recurrence]
```

→ **sequence-term** repräsentiert *Reihenfolge der Nachricht* in der Kommunikation

→ **recurrence** nennt *Bedingung* oder Schleife

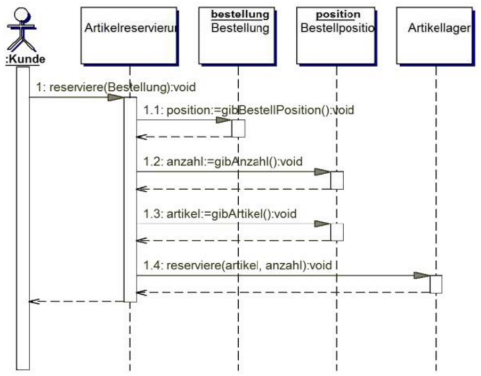
Beispiel: Instanz der Klasse A sendet remove() -Nachricht an Instanz von B, wenn s1 gleich s2 ist.

Beispiel

Reservierung von Artikeln eines Online-Shops als Sequenzdiagramm

Aufgabe:

Die Kommunikation soll übersichtlich zusammengefasst werden.



Mögliche Lösung: