Techniken der Projektentwicklungen

Dynamische Modellierung

Franz Kummert, Gerhard Sagerer

8. Termin

Statische Sicht auf Software

- Ausgangspunkt: Anforderungsanalyse
- Beschreibung der strukturellen Aspekte
- Identifikation von Klassen, Methoden und Attributen
- Konzeptionelle- und Implementierungssicht ergänzen einander
- Iterationen sind notwendig
- In UML: Objekt- und Klassendiagramm

Basiselemente des statischen UML-Modells

- Typen von Objekten (Klassen)
- Instanzen von Typen (Objekte)
- Attribute und Operationen
- Zugriffsrechte
- Beziehungen zwischen Elementen
- Einschränkungen bzgl. Verknüpfung von Elementen
- Schnittstellen und Schnittstellenklassen
- Gruppierungen (Pakete)

Was ist dynamische Modellierung

- Modellierung dynamischer Aspekte von Anwendungsfällen
- Spezifikation von Kontrollfluss und Interaktionen zwischen Objekten.
- Beschreibung dynamischer Strukturen einer Menge von Objekten

Was ist dynamische Modellierung

Zwei Typen von Interaktionsdiagrammen in der UML:

- Sequenzdiagramme: Zeitlicher Ablauf einer Interaktion
- Kollaborationsdiagramme: Strukturelle Zusammenarbeit

Basis für dynamische Modellierung:

- Ergebnisse der statischen Modellierung und der Anforderungsanalyse!
- **Aber: Trotzdem iterativer Prozess möglich und sinnvoll!**

UML Sequenzdiagramme

Grundlagen

- Betrachtung des Kontrollflusses objektorientierter Systeme
- Beschreibung von Szenarien, bei denen mehrere Objekte miteinander kommunizieren
- Schwerpunkt: Betrachtung zeitlicher Aspekte
- Spezifikation der chronologischen Abfolge der Nachrichten

Notation der beteiligten Instanzen

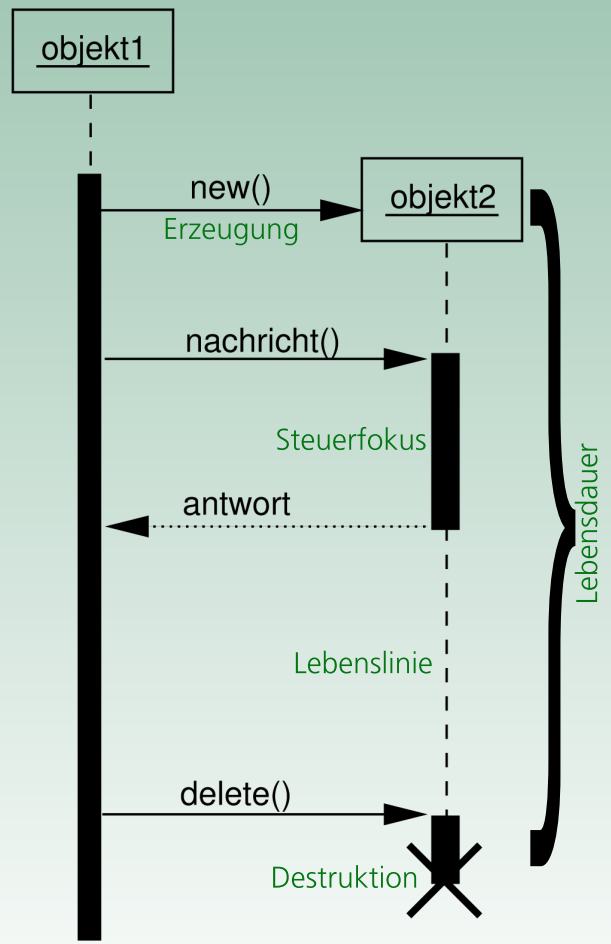
Basiskonzepte

Grundlegende Elemente

- Objekte, die an der Interaktion beteiligt sind
- Nachrichten zwischen Objekten

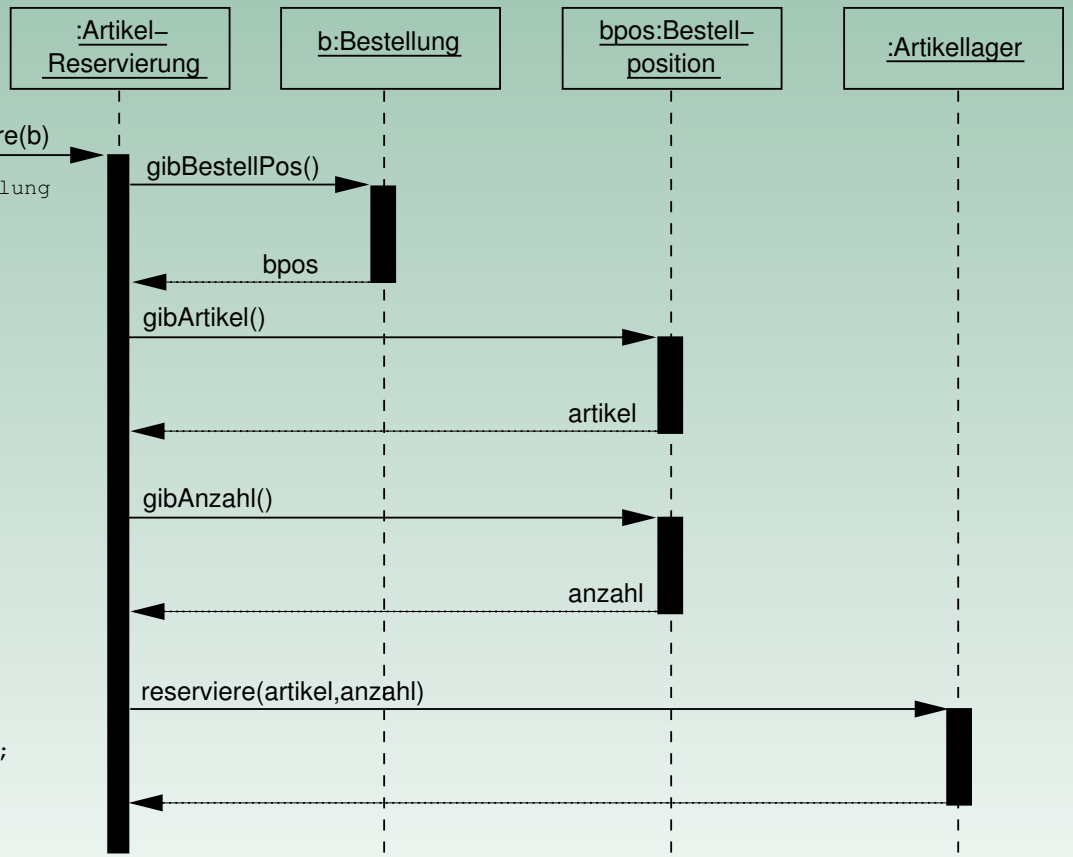
Zwei Dimensionen dargestellt

- vertikal: Zeitachse, zeitliche Abfolge „Top-Down"
- horizontal: betrachtete Objekte



Dargestellter Anwendungsfall: Artikel reservieren

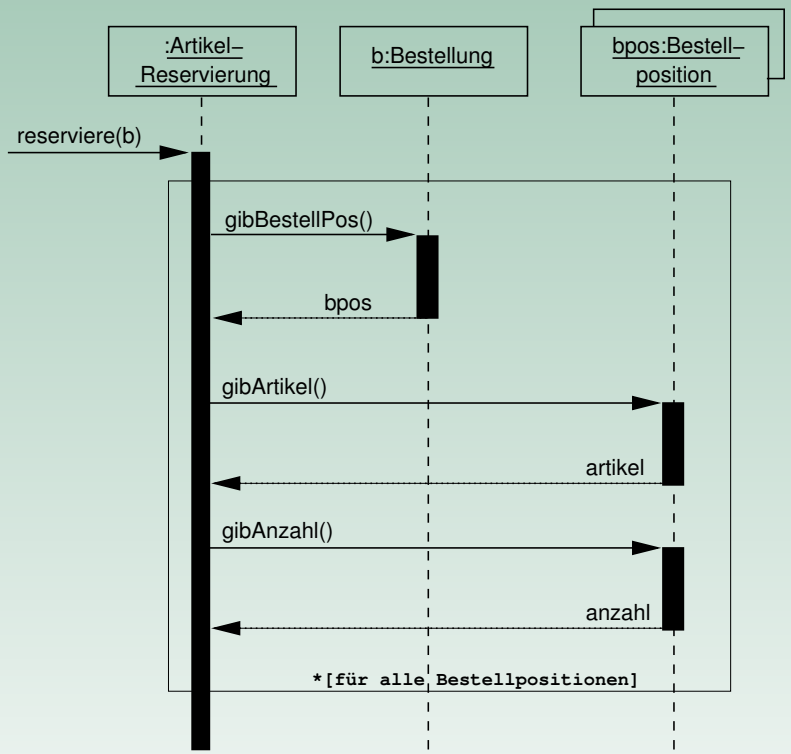
```
class ArtikelReservierung {
 ...
public void reserviere (Bestellung b) {
    ...
    // Bestellpositionen von der Bestellung
    // anfordern
    bpos = b.gibBestellPos();
    ...
    // Eine Bestellposition in Artikel
    // und Anzahl zerlegen
    artikel = bpos.gibArtikel();

    ...

    anzahl = bpos.gibAnzahl();

    ...



    // Artikel in entsprechender Anzahl
    // im Lager reservieren
    einLager.reserviere(artikel,anzahl);
 }
}
```



Nachrichten

Grundlegende Elemente

- Nachrichten sind Methodenaufrufe
- Typen von Nachrichten:
  - synchrone Methodenaufrufe
  - asynchrone Methodenaufrufe
  - Selbstaufruf eines Objekts
  - Iteration über Methoden



Nachrichten

Allgemeine Syntax von Nachrichten:

- Allgemeine Syntax von Nachrichten:

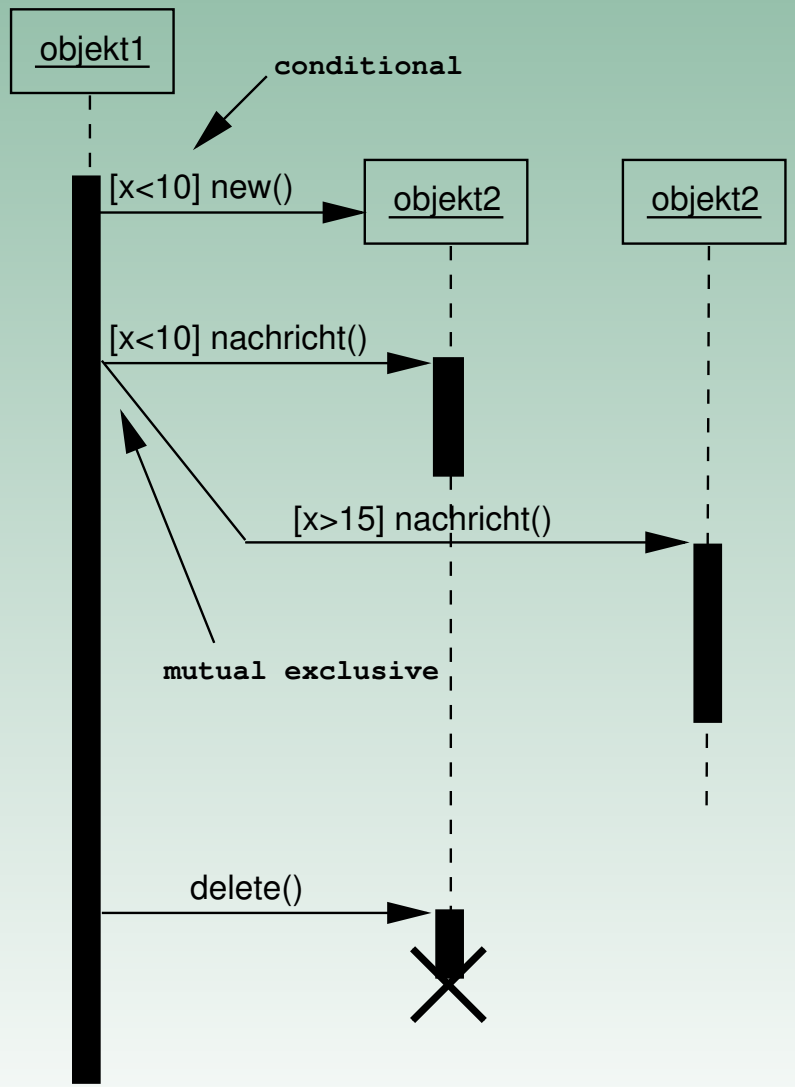
```
[<condition>] <return> := <message> (<parameter>:<parameterType>) : <returnType>
```

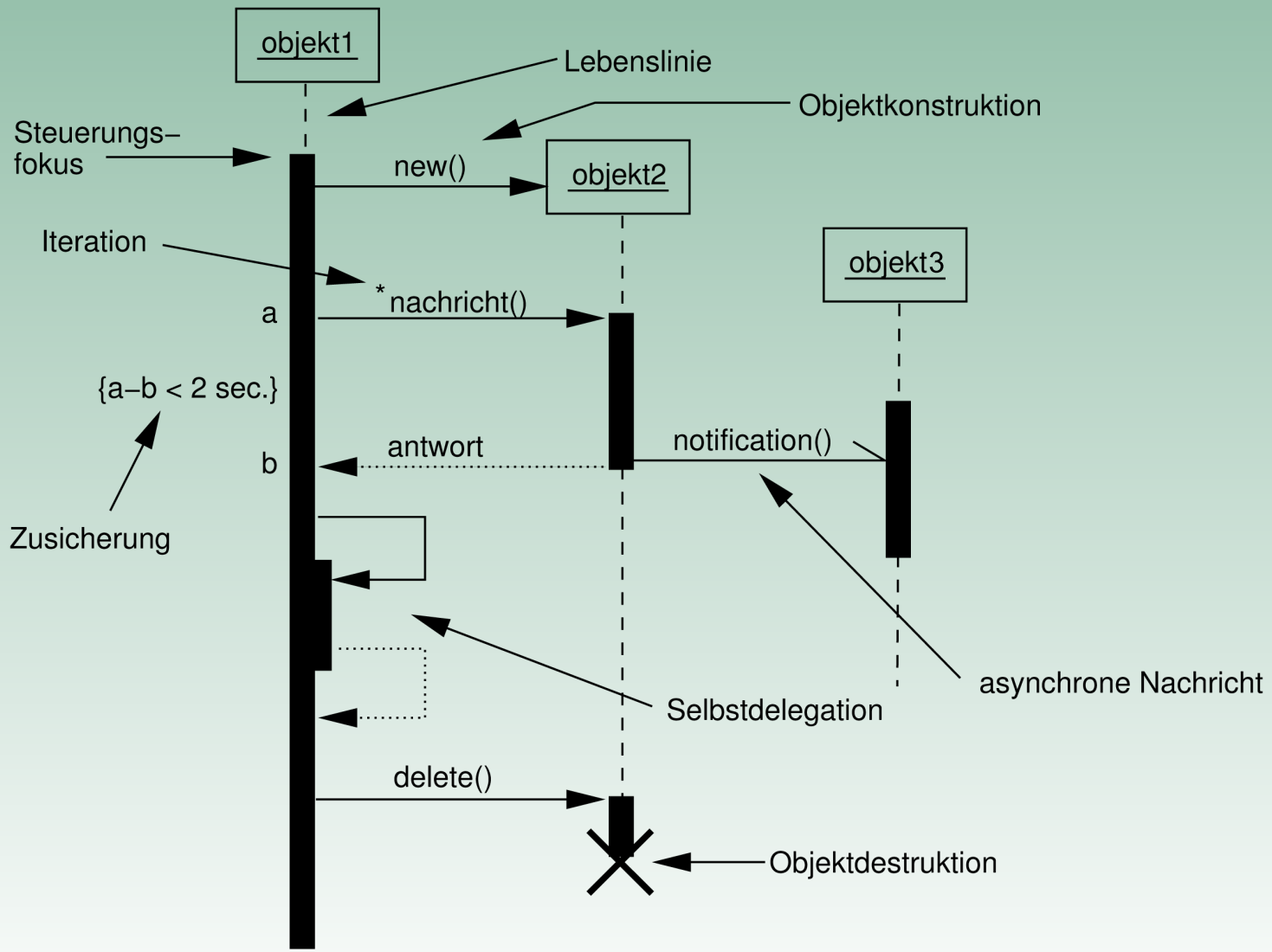
Beispiele (bis auf die Benennung einer Nachricht, alle Teile optional):

```
spec := getProductSpec(id)
spec := getProductSpec(id:ItemID)
spec := getProductSpec(id:ItemID) : ProductSpecification
```

Zwei Typen von bedingten Nachrichten:

- einfache bedingte Nachrichten (conditional messages)
- gegenseitig ausschließende bed. Nachrichten (mutually exclusive conditional messages)
- Zu viele Bedingungen: für jeden Fall eigenes Sequenzdiagramm
- Rückgabewerte verwendbar in Konditionen für weitere Nachrichten!




objekt1
Lebenslinie
Objektkonstruktion
Steuerungs–
fokus
new()
objekt2
Iteration
objekt3
* nachricht()
a
{a–b < 2 sec.}
antwort
notification()
b
Zusicherung
Selbstdelegation
asynchrone Nachricht
delete()
Objektdestruktion

Erweiterungen von Sequenzdiagrammen

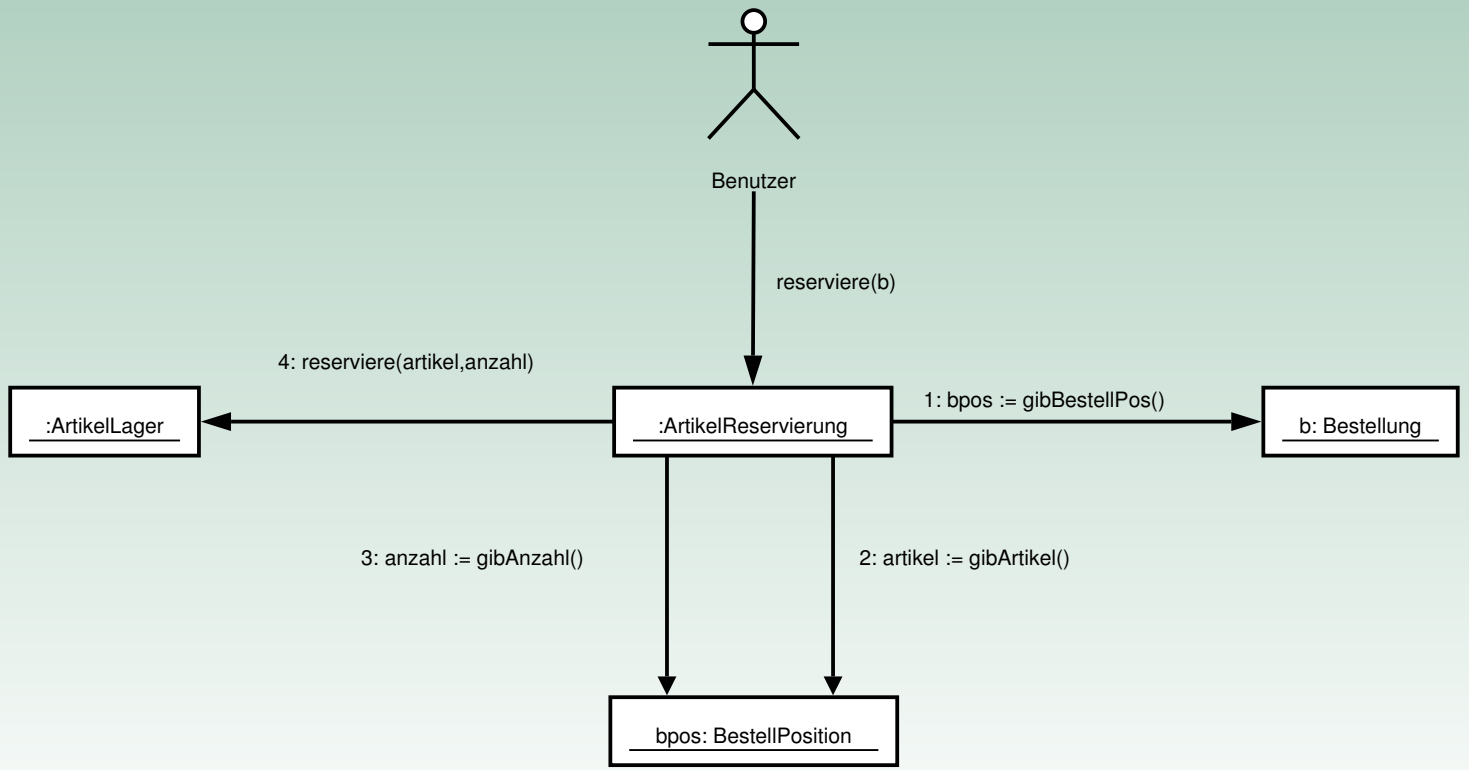
- alternative Lebenslinien für Objekte
- Parallelität

Regeln zur Verwendung

- Sequenzdiagramme primär für lineare Szenarien eines Use Case
- Geeignet für Darstellung des zeitlichen Ablaufs
- Keine Beschreibung des Nachrichtenflusses
- Verzweigung und Schleifen sparsam verwenden
- Jede Variante in eigenem Diagramm

UML Kollaborationsdiagramme

Kollaborationsdiagramme fassen Klassen, die zusammen ein kooperatives Verhalten bieten, dass mehr leistet als die Summe seiner Teile, zu Gruppen zusammen.



Eigenschaften

- Darstellung des Kontrollflusses entsprechend der Objektstruktur
- Kontext einer Interaktion wird sichtbar
- semantisch äquivalent zu Sequenzdiagrammen, Betonung anderer Aspekte
- unterschiedliche Interaktionspfade gleichzeitig darstellbar
- Nachteil: zeitlicher Ablauf nicht sofort erkennbar

Syntax

- zentrale Objekte immer nahe der Mitte des Diagramms
- Grundstruktur immer ein Graph
- Nachrichtennamen mit Symbol für Nachrichtentyp (synchron, asynchron) versehen
- Rücksprünge werden nicht explizit gezeigt
- zeitliche Abfolge über Nummerierungsschema definiert
- Notation von Nachrichten, Iterationen und Bedingungen wie in Sequenzdiagrammen

Kerngedanken von UML Interaktionsdiagrammen

- Modellierung dynamischer Aspekte von Anwendungsfällen
- Kontrollflüsse und Struktur einer Gruppe von Objekten
- Diskussionsbasis für Zuordnung von Anwendungsaufgaben
- zwei semantisch äquivalente Diagrammarten verfügbar:

Diagrammtyp	**Stärken**	**Schwächen**
Sequenz	Ablauf von Kontrollflüssen einfache Notation	horizontaler Platzbedarf Variationen leicht unübersichtlich
Kollaboration	Organisation von Kontrollflüssen komplexe Abläufe darstellbar verfügbarer Platz besser genutzt	Ablauf schwierig erkennbar wenig intuitive Notation

Zum nächsten mal

Klassendiagramm

In 5'er Gruppe, anhand der erstellten CRC Karten ein Klassendiagramm erstellen.

Sequenzdiagramm und Kollaborationsdiagramm

In 2'er und 3'er Gruppen aufteilen und zu den UseCases „NutzerInnen in P2P - Reichweite suchen" und „Passende ServicenutzerInnen in Umgebung lokalisieren" ein Sequenz- sowie ein Klassendiagramm erstellen.

Abgabe

- Bis zum Vortag des nächsten Tutoriums um 12 Uhr
- Klassendiagramm als PDF unter: /vol/tdpe/groupX/session8/teamY.pdf
- Sequenz- und Kollaborationsdiagramm als PDF unter: /vol/tdpe/groupX/session8/teamY[A|B].pdf
- Namen in die PDF Datei