Freie Universität Berlin, Institut für Informatik, Arbeitsgruppe Software Engineering
Lutz Prechelt, Linus Ververs, Nils Goldmann, Alexander Stage

Übungsblatt 6 **Statische und dynamische Analyse** zum 2025-05-26

---

## Aufgabe 6-1: Begrifflichkeiten & Statisches Objektmodell

*Lernziele: Kernbegriffe definieren können. Verstehen, wie ein UML-Diagrammtyp zielgruppen- und zweckgerecht verschieden eingesetzt werden kann.*

**a)** Grenzen Sie folgende Begriffe gegeneinander ab:

Problembereichsklassen *(application domain classes)* vs.
Lösungsklassen *(solution domain classes)*

**b)** Nennen Sie für jeden Begriff aus Teilaufgabe **a)** ein konkretes Beispiel aus dem Bereich Ihrer eigenen Softwareprojekt-Idee.

Das *statische Objektmodell* beschreibt die statischen Eigenschaften des Systems z.B. durch Klassen- und Objektdiagramme. Diese können in verschiedenen Phasen des Entwicklungsprozesses zum Einsatz kommen.

**c)** Recherchieren und beschreiben Sie den Unterschied zwischen den Klassendiagrammen, wie sie in den Phasen *Analyse*, *Entwurf* und *Implementierung* verwendet werden.

Differenzieren Sie nach folgenden Aspekten: Einsatz- oder Verwendungszweck, verwendete Terminologie, Semantik der modellierten Klassen und ihrer Eigenschaften, Semantik der Assoziationen, Detailgrad der Darstellung, Zielgruppe des Diagramms.

| | **Entwicklungsphase** | | |
|---|---|---|---|
| **Aspekt** | **Analyse** | **Entwurf** | **Implementierung** |
| *Einsatzzweck* | | | |
| *Terminologie* | | | |
| *Assoziationssemantik* | | | |
| *Detailgrad* | | | |
| *Zielgruppe* | | | |

Verwenden Sie bei der Beschreibung der jeweiligen Klassensemantik ausdrücklich die Begriffe aus Teilaufgabe **a)**.

## Aufgabe 6-2: Dynamisches Objektmodell (1): Aktivitätsdiagramm ★

*Lernziel: Einen natürlichsprachlichen Anwendungsfall als Aktivitätsdiagramm modellieren können.*

Die in Klammern angegeben Stichworte beziehen sich das “Dictionary of Terms” des Buchs *“The Unified Modelling Language Reference Manual. Second Edition”* von Rumbaugh, Jacobson und Booch (ab Seite 129). Lesen Sie bei Bedarf dort nach.

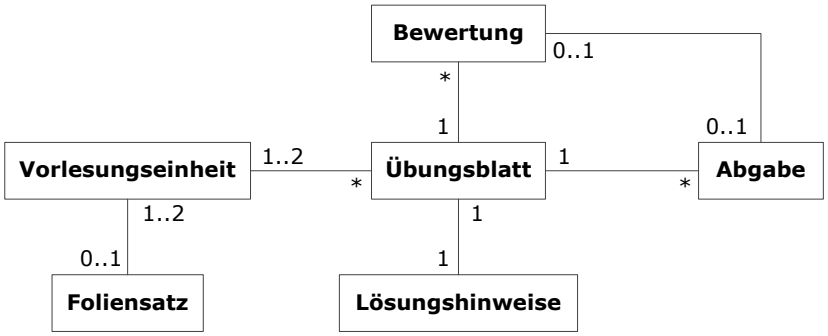
**Aufgabe:** Modellieren Sie den folgenden Anwendungsfall (nächste Seite) als UML-Aktivitätsdiagramm (Stichwort “activity”).

- Verwenden Sie Partitionen, um die Aktivitäten nach den jeweils ausführenden Akteuren zu trennen (“activity partition”).
- Modellieren Sie sowohl den Kontroll- als auch die Datenflüsse. Letztere können entweder als explizite “Pins” an den Aktivitätsknoten, oder als Objektknoten zwischen den Aktivitätsknoten notiert werden (“data flow”).

**Übersichts-Anwendungsfall**

1. Der Dozent der Veranstaltung nimmt sich eine unvorbereitete Vorlesungseinheit vor und plant diese Einheit, in dem er einen Foliensatz erstellt. Danach gilt die Vorlesungseinheit als vorbereitet.
2. Der Dozent nutzt den Foliensatz um später die Vorlesung zu halten. Außerdem lädt er den Foliensatz auf einen Webserver.
3. Wurde eine Vorlesungseinheit vorbereitet, geht diese an den Übungsleiter, der eine neue Übung plant, wenn für diese Vorlesungseinheit ein neuer Zettel nötig ist. (Übungszettel erscheinen wöchentlich, während es in der Regel zwei Vorlesungseinheiten in einer Woche gibt.)
4. Aus der Übungsplanung gehen einerseits ein neuer Übungszettel und andererseits die zugehörigen Lösungshinweise hervor.
5. Die Studierenden erhalten den Übungszettel, und bearbeiten diese, und geben sie bei ihren Tutoren ab.
6. Die Tutoren bewerten die Abgaben unter Verwendung der Lösungshinweise. Anschließend laden sie die Bewertungen in das KVV hoch.

Eine Kollegin hat bereits ein Klassendiagramm erstellt, das die Beziehungen zwischen den Domänenkonzepten spezifiziert. Achten Sie darauf, dass alle beteiligten Objekte (also Exemplare dieser Domänenklassen) als Objektflüsse im Aktivitätsdiagramm enthalten sind.



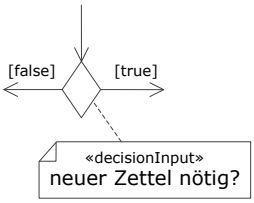

Ergänzende Erläuterungen zu den Domänenkonzepten:

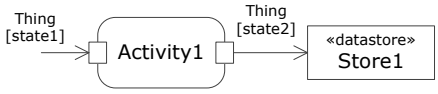
- Eine Bewertung bezieht sich auf einen konkreten Studierenden; auch wenn dieser keine Abgabe für ein bestimmtes Übungsblatt hat, soll es im KVV eine Bewertung geben.
- Vorlesungseinheiten haben zwei Zustände: `unvorbereitet` und `vorbereitet`.

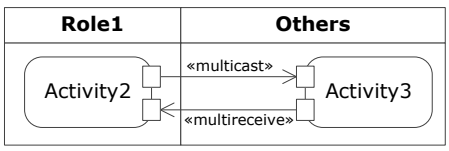
Folgende UML-Modellelemente könnten Ihnen dabei hilfreich sein (Notation: siehe unten):

- *Links:* Entscheidungen im Kontroll- und Datenfluss lassen sich durch einen sog. `decisionInput` fällen, die nicht ausmodelliert werden müssen. (“decision node”)
- *Mittig:* Zustände von Objekten können in eckigen Klammern angegeben werden. (“object node”)

  Webserver und KVV können Sie als `datastore` modellieren. (“data store node”)
- *Rechts:* Objektflüsse, die mehrere Adressaten gehen, können mit `multicast` aufgefächert, und mit `multireceive` zusammengefasst werden. (“object flow”)



## Aufgabe 6-3: Dynamisches Objektmodell (2): Zustandsdiagramm

*Lernziel: Einen einfachen Zustandsautomaten in UML modellieren können.*

*Zustandsdiagramme* dienen dazu, alle möglichen Verläufe der Zustände von Objekten einer Klasse darzustellen. In Softwaretechniksprache: Ein Zustandsautomat beschreibt den Lebenszyklus eines Objekts.

**a)** In Zustandsdiagrammen *(statechart diagrams)* können Aktionen/Aktivitäten zum einen an Zustandsübergängen *(transitions)* und zum anderen in den Zuständen *(states)* gebunden werden. Wie unterscheiden sich diese beiden Alternativen hinsichtlich ihrer Notation? Was ist der semantische Unterschied?

**b)** Erstellen Sie ein Zustandsdiagramm für eine elektrische Wäscheschleuder (wie abgebildet zum Trocknen von Wäsche).

Die Wäscheschleuder kennt drei Signale (*signal events*), die Sie in Ihrem Zustandsautomaten verwenden können:

- `zu machen`: Deckel schließen
- `auf machen`: Deckel öffnen
- `starten`: Schleuderstart-Taste wird gedrückt

sowie zwei Sensoren (Bedingungen):

- `beladen`: Trommel ist beladen
- `trocken`: Inhalt ist trocken

und drei Aktionen:

- `pieps()`: Fehleranzeige
- `an()`: Motor startet
- `aus()`: Motor stoppt

Sie soll sich folgendermaßen verhalten:

**V1** Der Deckel der Wäscheschleuder ist initial geschlossen.

**V2** Der Motor startet, wenn die Starttaste gedrückt wird, aber nur wenn der Deckel geschlossen, die Trommel beladen und die Wäsche feucht ist. Andernfalls wird die Taste mit Pieps quittiert.

**V3** Der Motor stoppt, wenn die Wäsche trocken ist oder der Deckel geöffnet wird.

**Erinnerung**: Es gibt verschiedene Ereignisse (*events*), die einen Zustandsübergang auslösen können (*trigger*). Sie werden sowohl von den o.g. Signalen Gebrauch machen müssen, als auch von Änderungsereignissen (*change event*), die auf einen booleschen Ausdruck lauschen und auslösen, wenn dieser wahr wird.

Signalereignisse werden einfach durch Angabe ihres Namens notiert, Änderungsereignisse als „`when(<boolean expression>)`“.