Freie Universität Berlin, Institut für Informatik, Arbeitsgruppe Software Engineering
Barry Linnert, Franz Zieris, Michaela Borzechowski, Tanja Linkermann, Marco Ziener

Übungsblatt 9 **OCL** zum 19.06.2016, 16:00 Uhr

Aufgabe 9-1: Begriffe

a) Was versteht man im Kontext der Softwareentwicklung unter dem Begriff *Information Hiding*? Erläutern Sie Prinzipien und Heuristiken.

b) In welchem Zusammenhang stehen die Begriffe Information Hiding und das *„Need to Know"-Prinzip*?

c) Warum ist das Prinzip des Information Hiding sinnvoll? Erläutern Sie dieses an einem Beispiel.

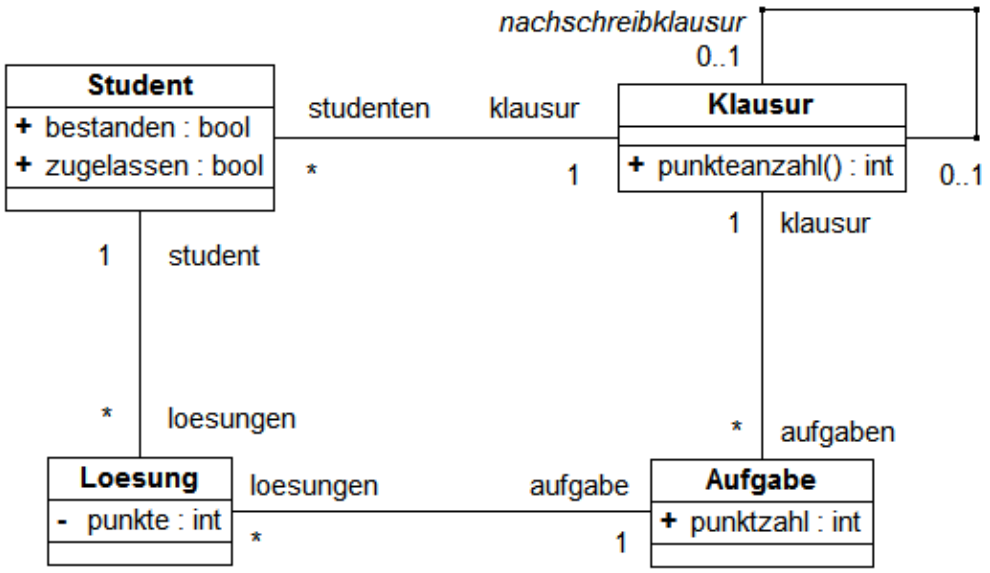
d) Was versteht man unter dem *„Design by Contract"-Prinzip*? In welchem Kontext wird es eingesetzt und warum?

e) Erläutern Sie in welchem Zusammenhang die Begriffe *OCL, Invariante, Design by Contract, precondition, constraints, postcondition* stehen.

Aufgabe 9-2: OCL lesen und schreiben

In einem Prüfungsverwaltungssystem sollen die Klausurergebnisse und die Punktzahlen der Studierenden bei den Lösungen der einzelnen Aufgaben verwaltet werden.

Das folgende UML-Klassendiagramm modelliert einen Teil der benötigten Daten.



a) Drücken Sie auf Basis des Diagramms folgende OCL-Constraints natürlichsprachlich aus, also in einer Form, die Sie einem Gespräch mit einer Person ohne Informatik-Hintergrund verwenden würden. (In Teilaufgabe **b)** finden Sie Formulierungen dieser Art.)

Mit `size()` wird die Elementanzahl einer Menge oder Liste berechnet. Mit `self` wird das Kontextexemplar bezeichnet.

1. `context Aufgabe inv: punktzahl > 0`
2. `context Student inv: loesungen->size() = klausur.aufgaben->size()`
3.
```
context Klausur inv:
    studenten->forAll (s |
       s.bestanden implies s.loesungen->exists (l |
          l.punkte > 0 and l.aufgabe.klausur = self
       )
    )
```

b) Geben Sie OCL-Constraints an, die folgende Sachverhalte formalisieren.

Schauen Sie auch in der OCL 2.2 Spezifikation (http://www.omg.org/spec/OCL/2.2/PDF) nach, falls Ihnen Ausdrucksmittel fehlen.

1. In jeder Klausur gibt es mindestens eine Aufgabe mit genau einem Punkt.
2. Eine Nachschreibklausur kann keine Nachschreibklausur haben.
3. Ist ein/e Student/in zugelassen, gibt es für jede Klausuraufgabe auch eine Lösung von ihm/ihr.

Aufgabe 9-3★: OCL-Modellerweiterungen

Diese Aufgabe baut auf dem gleichen UML-Klassendiagramm auf wie Aufgabe **9-2**. Nun soll zusätzlich auch die Klausurkorrektur modelliert werden: Bei der Korrektur geht der/die Dozent/in alle Lösungen einzeln durch, bewertet sie jeweils und errechnet die Gesamtpunktzahl für jede/n Student/in.

a) Erweitern Sie das Klassendiagramm um eine Operation `korrigieren()` mit passender Signatur sowie ein neues Attribut. Die Operation `korrigieren()` wird aufgerufen, sobald der/die Dozent/in eine Lösung korrigiert hat.

Es soll insbesondere damit möglich sein, dass

- das Attribut `punkte` in `Loesung` gefüllt wird (Achtung: Die Sichtbarkeit des Attributs darf nicht verändert werden!)
- die erreichte Gesamtpunktzahl des Studenten aktualisiert wird.

Beschreiben Sie zunächst verbal, was `korrigieren()` genau leisten soll. Finden Sie geeignete Stellen (und im Falle des neuen Attributs: auch einen geeigneten Namen) für die neuen Member im Klassendiagramm.

b) Gehen Sie zunächst in einer ersten Version davon aus, dass `korrigieren()` nur einmal pro Lösung aufgerufen werden darf; die einmal gesetzte Punktzahl ist danach unveränderlich.

Spezifizieren Sie sowohl möglichst strenge Vorbedingungen für Ihre Operation `korrigieren()`, als auch die alle Effekte *(postcondition)* der Operation komplett in OCL.

c) Gehen Sie nun von der realistischeren Anforderung aus, dass `korrigieren()` mehrfach aufgerufen werden kann, um z.B. die Punktzahl einer Lösung bei einer Klausureinsicht zu korrigieren.

Spezifizieren Sie wiederum möglichst strenge Vorbedingungen und alle Effekte der Operation in OCL.

d) Spezifizieren Sie den Effekt der Operation `Klausur.punkteanzahl()` unter Verwendung des OCL-Schlüsselwortes `result`.

Hinweis: Der resultierende Ausdruck passt bequem auf eine Zeile. Recherchieren Sie bei Bedarf nach OCL-Collections und ihren Operationen.