

Name: \_\_\_\_\_

Tutor: \_\_\_\_\_

Matrikelnummer: \_\_\_\_\_

Punkte: \_\_\_\_\_

Übungsleiter / Gruppe: \_\_\_\_\_

Abgabe: 2.5.2002, 8<sup>15</sup> Uhr**1. Aufgabe: UML-Werkzeuge****(8 Punkte)**

Beim Entwurf von Software mittels UML-Diagrammen ist es notwendig, die verschiedenen Sichten konsistent miteinander zu verknüpfen, sodass sich Änderungen in einem Diagramm sofort in allen abhängigen Diagrammen widerspiegeln. Diese Unterstützung bieten UML-Werkzeuge und UML-Modellierungsumgebungen an. Es gibt eine Vielzahl solcher Tools, die vieles gemeinsam haben, sich aber doch in mancher Hinsicht voneinander unterscheiden.

Ihre Aufgabe besteht nun darin, mindestens 2 solche UML-Werkzeuge zu finden und diese zu evaluieren. Dabei sollten Sie die folgenden Fragen kurz beantworten:

- Name des Werkzeuges, Hersteller/Vertreiber (mit URL), sonstige Informationsquellen
- Auf welchen Plattformen ist das Werkzeug verfügbar?
- Lizenzen:
  - Welche verschiedenen Arten der Lizenzierung gibt es? Was kosten diese?  
Welche Funktionalität bieten die verschiedenen Versionen an?
  - Gibt es Demo/Trial/Shareware/Freeware-Versionen?  
Was muss man tun, um diese zu bekommen?  
Was kann man mit diesen tun? Was nicht?
- Welche Diagramme werden unterstützt?  
Werden diese auch gegeneinander synchronisiert?
- Kann automatisch Programmcode erzeugt werden?  
Wenn ja, welche Programmiersprachen werden unterstützt?
- Ist Reverse Engineering (also die Generierung der Diagramme aus Quellcode) möglich?  
Wenn ja, welche Diagramme lassen sich so erzeugen, und  
welche Programmiersprachen werden unterstützt?

Erörtern Sie auch kurz die Bedienung der Werkzeuge. Ist sie einfach und intuitiv oder zu kompliziert? Würden Sie mit diesem Tool arbeiten wollen? Warum? Warum nicht?

**2. Aufgabe: Use Case Diagramm****(3 (+1) Punkte)**

Software für eine *Selbstbedienungstankstelle*

Kunden tanken ihre Fahrzeuge an den Zapfsäulen auf. Die Zapfsäulen ermitteln die ausgegebene Treibstoffmenge, die Treibstoffart (Super, Diesel, ...) und den zu bezahlenden Betrag, und übertragen diese Informationen an das System. Nach dem Tanken bezahlen die Kunden an der Kassa. Am Ende eines jeden Geschäftstages soll an der Kassa eine Gesamtrechnung der Einnahmen und der ausgegebenen Treibstoffmengen erstellt werden.

Erstellen Sie für die oben beschriebene Situation ein UML-Use-Case-Diagramm. Zeichnen Sie alle Aktoren, Use-Cases und deren Beziehungen sowie eventuelle Systemgrenzen (die EDV der Tankstelle soll das System sein) ein.

**Wenn Sie ein UML-Tool für diese Aufgabe verwenden, erhalten Sie 1 Extrapunkt.**

### 3. Aufgabe: Sequenzdiagramm

(6 Punkte)

```
public class GasStation {
    int gasSold;
    int moneyMade;

    public void daysTotal () {
        Iterator iter = getPumps();
        while (iter.hasNext()) {
            Pump p = (Pump) iter.next();
            gasSold += p.gasSold();
            moneyMade += p.moneyMade();
            p.resetDayCounters();
        }
    }

    private Iterator getPumps () { ... }

    ...
}
```

Stellen Sie die gegebene Java-Methode in einem UML-Sequenzdiagramm dar (`daysTotal()` wird, wie bei Aufgabe 2 beschrieben, von Kassa auf einem Objekt `myGS` aufgerufen). Das Diagramm soll alle Methodenaufrufe der Methode `daysTotal()` und auch die Zielobjekte der Aufrufe zeigen.

### 4. Aufgabe: Klassendiagramm

(7 (+2) Punkte)

Ein Grafikprogramm soll Kreise und Polygone darstellen können. Diese grafischen Objekte sollen gezeichnet und verschoben werden können. Jedes Grafikobjekt hat einen bestimmten Stil, der die Liniendicke und -farbe sowie die Füllfarbe bestimmt. Ein Kreis ist durch Mittelpunkt und Radius bestimmt, ein Polygon durch die Menge seiner Eckpunkte. Ein Punkt beschreibt eine Position auf der Zeichenebene durch seine x- und y-Koordinaten, die in Pixel vom linken, oberen Eckpunkte angegeben werden.

Entwerfen Sie aus dieser Spezifikation ein Klassenmodell (ev. nach der Methode von Abbott) und stellen Sie es als UML-Klassendiagramm dar. Das Klassendiagramm soll folgendes enthalten:

- notwendige Klassen und Interfaces
- Attribute mit ihrem Datentyp
- Methoden mit ihren Schnittstellen (= Typen von Rückgabewert und Parametern)
- Vererbungs- und sonstige Beziehungen zwischen den Klassen (Achten Sie auf die richtige Verwendung von Assoziation, Aggregation und Komposition!)

**Wenn Sie ein UML-Tool für diese Aufgabe verwenden, erhalten Sie 2 Extrapunkte.**