Kurs 1793 “Software Engineering I - Grundkonzepte der OOSE”

Klausur am 3.8.2002

Sommersemester 2002
Hinweise zur Bearbeitung der Klausur zum Kurs 1793
“Software Engineering I - Grundkonzepte der OOSE”

Wir begrüßen Sie zur Klausur "Software Engineering I". Bitte lesen Sie sich diese Hinweise vollständig und aufmerksam durch, bevor Sie mit der Bearbeitung der Aufgaben beginnen.

1. Prüfen Sie bitte die Vollständigkeit Ihrer Unterlagen. Ihr Klausurexemplar umfaßt:

   - 2 Deckblätter,
   - diese Hinweise zur Bearbeitung,
   - 1 Formblatt für eine Bescheinigung für das Finanzamt,
   - 4 Aufgaben (Seite 3 - Seite 7)

2. Füllen Sie, **bevor** Sie mit der Bearbeitung der Aufgaben beginnen, folgende Seiten des Klausurexemplars aus:

   a) **Beide** Deckblätter mit Name, Anschrift sowie Matrikelnummer. **Markieren Sie vor der Abgabe auf beiden Deckblättern die von Ihnen bearbeiteten Aufgaben**.
   b) Falls Sie eine Teilnahmebescheinigung für das Finanzamt wünschen, füllen Sie bitte das entsprechende Formblatt aus.

   Nur wenn Sie beide Deckblätter vollständig ausgefüllt haben, können wir Ihre Klausur korrigieren!

3. Streichen Sie ungültige Lösungen deutlich durch.

4. Schreiben Sie bitte auf jedes beschriebene Blatt oben links Ihren Namen und oben rechts Ihre Matrikelnummer. Wenn Sie weitere eigene Blätter benutzt haben, heften Sie auch diese, mit Namen und Matrikelnummer versehen, an Ihr Klausurexemplar.

   Lose Blätter, insbesondere ohne Name und Matrikelnummer, werden nicht bewertet!

5. Neben Schreibgerät und unbeschriebenen Konzeptpapier sind **Kurseinheiten, Einsendeaufgaben** sowie deren **Musterlösungen** als Unterlagen zugelassen.

6. Mit **Bleistift** geschriebene oder gezeichnete Lösungen werden **nicht korrigiert**. Verwenden Sie bitte nur Füller oder Kugelschreiber.

7. Es sind maximal 100 Punkte erreichbar.

Wir wünschen Ihnen bei der Bearbeitung der Klausur viel Erfolg!

Kurs 1793 “Software Engineering I - Grundkonzepte der OOSE”

Klausur am 3.8.2002

Kurs 1793 "Software Engineering I - Grundkonzepte der OOSE"

Klausur am 3.8.2002

Aufgabe 1 (*19 + 10 + 10 = 39 Punkte*)

In dieser Aufgabe sollen Sie ein Domänen-Klassendiagramm, zwei Objektdiagramme und ein Anwendungsfalldiagramm erstellen.

Ein Versandhandel möchte eine Warenbestellsystem erstellen, um Kunden das Bestellen von Waren (Produkten) über Internet zu ermöglichen. Der Auftraggeber gibt als Anforderungen u.a. Folgendes an:
"Jedes Produkt hat eine Bezeichnung, eine Beschreibung, einen aktuellen Preis und ist einer Produktgruppe eindeutig zugeordnet. Der Kunde kann ein Produkt auswählen, indem er sich alle Produkte einer Produktgruppe anzeigen lässt und aus dieser Liste ein Produkt auswählt. Alternativ kann er eine Menge von Suchbegriffen eingeben, anhand derer das System eine Liste aller Produkte anzeigt, die die Suchbegriffe in ihrer Bezeichnung enthalten. Vom Benutzer ausgewählte Produkte werden (zusammen mit einer Stückzahlangabe) in einem sog. "virtuellen Warenkorb" gespeichert, den der Kunde jederzeit einsehen kann. Wenn der Kunde die Produktauswahl beendet hat, kann er mit seinem virtuellen Warenkorb "zur Kasse gehen" und die darin enthaltenen Produkte bezahlen. Die Zahlung kann mit Kreditkarte, Rechnung, Lastschrift oder Nachnahme erfolgen. Das Sortiment und die Preise können sich ändern, dennoch muss auch im Nachhinein feststellbar sein, wieviel ein Kunde für ein bestimmtes Produkt bezahlt hat und um welches Produkt es sich handelte. Die Zuordnung von Waren zu Produktgruppen ändert sich nicht. Nach dem Bezahlvorgang ist der Warenkorb leer. Das System muss also in der Lage sein, folgende Anfragen zu beantworten:

1. Welche Produkte hat Kunde "Müller" zur Zeit in seinem Warenkorb?
2. Welche Produkte gibt es in der Produktgruppe "Buch"?
3. Welche Kunden haben schon einmal ein Produkt der Produktgruppe "Papier" bestellt?
4. Wieviel hat der Kunde "Meyer" am 25.09.2001 für das Produkt mit der Bezeichnung "Tintenpatrone schwarz" bezahlt?
5. Welche Produkte sind im Jahr 2001 bestellt worden, die heute nicht mehr im Sortiment sind?"

Hinweis: Zu dieser Aufgabe gibt es mehrere richtige Lösungen, Sie müssen nicht die Musterlösung "erraten".

Aufgabenstellung:

a) Erstellen Sie ein möglichst kleines und redundanzfreies Klassendiagramm für die beschriebene Anwendungsdomäne. Geben Sie zu jeder Beziehung die Multiplizitäten an.

b) Skizzieren Sie ein Beispiel-Objektdiagramm, bei dem ein Produkt in dem Warenkorb eines Kunden liegt, sowie ein zweites Objektdiagramm, dass den Zustand der Anwendung darstellt, nachdem dieser Warenkorb bestellt und mit Kreditkarte bezahlt wurde. (Sie dürfen annehmen, dass das Sortiment nur aus einer Ware und die Kundenkartei nur aus einem Kunden besteht.)

c) Erstellen Sie ein Anwendungsfalldiagramm (4-7 Anwendungsfälle) für den beschriebenen Ausschnitt des Warenbestellsystems (ohne die Anfragen).

Aufgabe 2 (*15 Punkte*)

In dieser Aufgabe sollen Sie den Bewegungsablauf eines mobilen Roboters mit Hilfe eines Zustandsdiagramms modellieren.

Die Aufgabe des mobilen Roboters ist es, einer auf den Boden gemalten gerade oder kurvenförmig verlaufenden Linie bis zu einem Zielpunkt zu folgen. Dabei hat der Roboter die Möglichkeit zur Geradeausfahrt, oder sich auf der Stelle links bzw. rechts herum zu drehen. Zur Abtastung der Linie besitzt der Roboter an seiner Unterseite einen optischen Sensor. Dieser meldet „Abweichung nach rechts“, wenn der Roboter ein bestimmtes Stück nach rechts von der Linie abgekommen ist. Der Roboter muss dann solange links herum drehen, bis der Sensor meldet: „Linie wieder erreicht“. Anschließend kann der Roboter weiter geradeaus fahren. Für die Abweichung nach links gilt Analoges.

Wird der Roboter eingeschaltet, fährt er zunächst geradeaus. Auf der Linie befinden sich in gewissen Abständen Kontakte, die vom Roboter bei seiner Überfahrt abgetastet werden und das Ereignis "Zwischenpunkt erreicht“ auslösen. Bei einem Zwischenpunkt soll die im Roboter eingebaute Signallampe während der Fahrt einmal kurz aufblitzen. Am Zielpunkt wird das Ereignis "Zielpunkt erreicht" ausgelöst. Dort soll der Roboter anhalten und die Lampe dauerhaft einschalten. Während der Geradeausfahrt wird, gesteuert durch einen Takt-Generator, in regelmäßigen Zeitabständen der Akku-Ladezustand des Roboters überprüft. Sinkt die Ladung unter einen vorgegebenen Mindestwert ab, hält der Roboter an und die Signallampe blinkt.

Der Roboter kann im Stillstand manuell wieder ausgeschaltet werden.

Vereinfachend können Sie von folgenden Annahmen ausgehen:

- Die Linie hat keinerlei Abzweigungen oder Kreuzungen, es gibt keine scharfen Kurven (d.h. Kurven mit engem Kurvenradius).
- Kontakte werden nur während einer Geradeausfahrt erreicht.
- Zum Start sitzt der Roboter genau auf der Linie.
- Die verschiedenen Funktionsarten der Signallampe werden durch entsprechende Operationen aktiviert, eigene Zustände sind hierfür **nicht** zu modellieren.

Aufgabenstellung:

Erstellen Sie zur Modellierung des Bewegungsablaufs des Roboters ein geeignetes Zustandsdiagramm.

Aufgabe 3 (*6 + 1 + 7 = 14 Punkte*)

Gegeben ist das Klassendiagramm in Abb. 1.

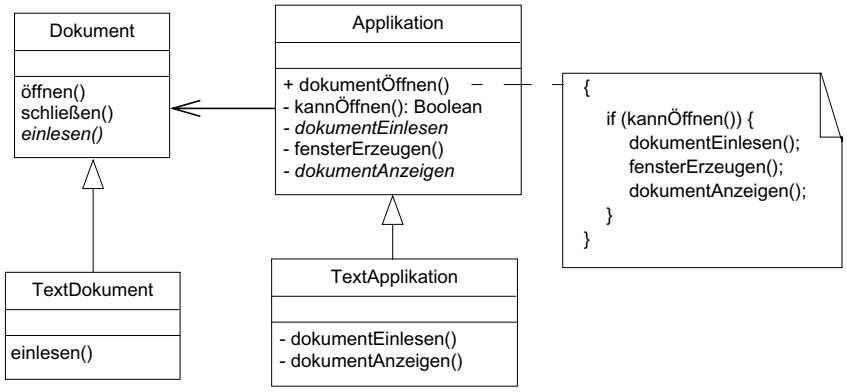

Abb. 1 Klassendiagramm

Aufgabenstellung:

a) Skizzieren Sie ein Sequenzdiagramm, das die Ausführung der Operation dokumentÖffnen() im Falle einer TextApplikation beschreibt.

b) Wie heißt das Entwurfsmuster, das hier zur Anwendung kommt? Geben Sie die deutsche und die englische Bezeichnung an.

c) Angenommen, die Operation kannÖffnen() prüft bisher nur, ob das zu öffnende Datei vorhanden ist und gelesen werden kann, jedoch nicht, ob sie den richtige Erweiterung hat. Beispielsweise sollte eine Textdatei die Dateinamenserweiterung ".txt" haben, während für Graphikdateien beispielsweise die Erweiterung ".png" vorgeschrieben sei (wir gehen vereinfachend davon aus, dass eine Applikation nur einen einzigen Dateityp öffnen kann).
Es stehen drei Entwurfsalternativen zur Diskussion, die den Test auf die korrekte Erweiterung realisieren:

1. Die Methode kannÖffnen() wird in den Unterklassen überschrieben. Für die Unterklasse TextApplikation könnte eine Realisierung von kannÖffnen() wie folgt lauten:
   return super.kannÖffnen(dateiname) and dateiname.endsWith(".txt");
2. In der Klasse Applikation wird ein Attribut erweiterung:String ergänzt. Im Konstruktor der Unterklassen wird dieses Attribut initialisiert, also im Fall der Textapplikation mit ".txt". Die Operation kannÖffnen() in der Klasse Applikation greift auf dieses Attribut zu, um die korrekte Erweiterung zu ermitteln.
3. In der Klasse Applikation wird eine abstrakte Operation gibErweiterung() definiert und in den Unterklassen überschrieben. Die Operation kannÖffnen() verwendet die Operation gibErweiterung() zur Ermittlung der korrekten Erweiterung.

Beschreiben Sie die Vor- und Nachteile der drei Alternativen und geben Sie an, für welche Alternative Sie sich entscheiden würden.

Aufgabe 4 (*20 + 6 + 6 = 32 Punkte*)

Für eine Schule soll ein Programm zur Verwaltung der Stunden- und Vertretungspläne erstellt werden. Sowohl Lehrer als auch Schüler haben einen Stundenplan, der eine Woche umfasst und jede Woche gleich ist. Zusätzlich gibt es einen Vertretungsplan, der für jede Unterrichtsstunde eines erkrankten Lehrers einen Vertretungslehrer aufweist (eine Vertretung ist nur bei Unter- und Mittelstufenklassen bzw. -Kursen erforderlich, d.h. bei den Jahrgangstufen 5-10). Dieser Plan ist von Woche zu Woche verschieden.

Abb. 2 zeigt einen Versuch, die Problemwelt dieser Anwendung durch ein Klassendiagramm zu modellieren. Allerdings widerspricht dieses Klassendiagramm einigen Heuristiken aus dem Kurs, enthält einige überflüssige Klassen und auch einige Redundanzen bzw. Modellierungsfehler (z.B. zum Beispiel wird der Lehrerstundenplan und der Vertretungsplan zusammen mit einer einzigen Assoziation (GibtUnterricht) modelliert, obwohl sie in der Anwendungsdomäne getrennt gehandhabt werden..

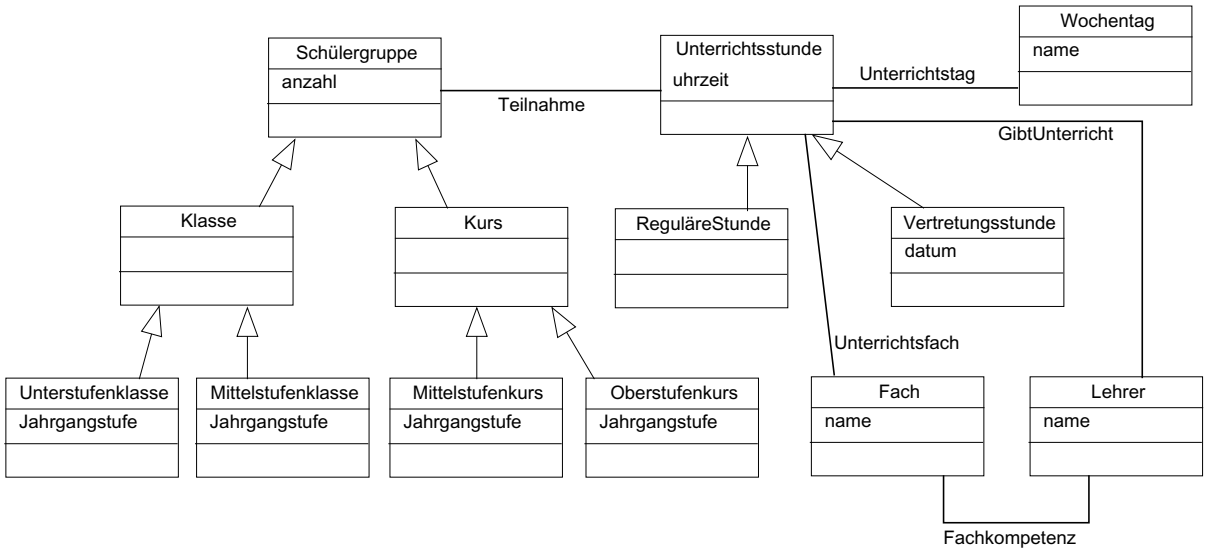

Abb. 2 Schlechtes Klassendiagramm zur Stundenplananwendung

Aufgabenstellung

a) Ändern Sie das Klassendiagramm in Abb. 2, so dass es

- den Heuristiken im Kurs entspricht
- möglichst wenige Klassen definiert
- keine Redundanzen enthält und
- der Lehrerstundenplan und der Vertretungsplan getrennt modelliert sind.

Hinweis: Es gibt eine Lösung, die ungefähr halb so viele Klassen enthält wie das Klassendiagramm in Abb. 2.

b) Begründen Sie jede Ihrer Änderungen in einem Satz.

c) Ergänzen Sie in ihrem Klassendiagramm die fehlenden Multiplizitäten.