Probeklausur zur Vorlesung „Softwaretechnologie“ 2010/2011

Dr. Günter Kniesel

Institut für Informatik III
Universität Bonn

– 23. Dezember 2010 –

Grundsätzliches

Die Probeklausur ist ein Auszug aus einer tatsächlich in dieser Form stattgefundenen Klausur. Der einzige Unterschied besteht darin, dass Aufgaben aus den in der laufenden Vorlesung noch nicht besprochenen Themenbereichen weggelassen worden sind. Dementsprechend führen wir die Probeklausur in 90 statt 120 Minuten durch.

Der Sinn der Probeklausur ist es, Ihnen eine Vorstellung von dem Ablauf der Klausur sowie der Art der Aufgaben zu vermitteln und Ihnen zu ermöglichen Ihren aktuellen Leistungsstand selbst einzuschätzen.

Versuchen Sie nicht anhand der Probeklausur „Kafffeesatzlesen“ zu betreiben! Sowohl der Schluss, dass etwas in der Klausur nicht vorkommen wird, weil es in der Probeklausur schon dran war, als auch der Umkehrschluss könnte sich als gefährlicher Trugschluss erweisen.

Das Einzige, wovon Sie ausgehen können ist, dass die Klausur

- mehr Aufgaben anbieten wird, als für das Erreichen der maximalen Punktzahl erforderlich wäre, so dass Sie die Möglichkeit haben, mindestens eine Aufgabe wegzulassen und trotzdem die Note 1 zu erreichen
- und in ungefähr die gleiche *Art* von Aufgaben (aber nicht unbedingt die gleichen Inhalte) enthalten wird.

Nachklausur zur Vorlesung „Softwaretechnologie“ 2009/2010

Dr. Günter Kniesel

Institut für Informatik III
Universität Bonn

– 26. März 2010 –

Aufgabe		Maximale Punkte	Erreichte Punkte	Prüfer / Beisitzer
1	Verschiedenes	8,0		
2	Klassen	15,5		
3	Aktivitäts-Diagramm	6,0		
4	Use Case	14,5		
5	Analyse	11,5		
6	Entwurfsmuster	9,0		
7	Entwurfsmuster	11,0		
8	CRC + DBC	9,0		
9	… gestrichen …	5,0		
10	… gestrichen ...	12,5		
Summe		**102,0**		
Note				

Wertung

Für die Note 1.0 reichen 76,5 von 102 Punkten. Diese Regelung ermöglicht Ihnen, Teilaufgaben im Umfang von ca. ¼ der Punkte wegzulassen und trotzdem eine 1.0 zu erzielen!

Setzen Sie also Prioritäten und teilen Sie sich die Zeit richtig ein! *Alle Aufgaben in der verfügbaren Zeit komplett zu bearbeiten dürfte schwierig sein.*

Zuordnung Punkte → Note			
.....– 76,5 → 1,0	68 – 64,5 → 2,0	56 – 52,5 → 3,0	44 – 40,5 → 4,0
76 – 72,5 → 1,3	64 – 60,5 → 2,3	52 – 48,5 → 3,3	40 – 0 → 5,0
72 – 68,5 → 1,7	60 – 56,5 → 2,7	48 – 44,5 → 3,7	

Weitere Hinweise

Folgende Aufgaben sind selbstständig, ohne Verwendung von Hilfsmitteln zu lösen.

Dinge die nicht in die Wertung einfließen sollen *(z.B. Randüberlegungen, als falsch erkannte Antworten, verworfene Versuche) bitte durchstreichen oder sonstwie deutlich markieren.*

Tragen sie Ihre Matrikelnummer oben rechts auf jedem Blatt ein. *Antworten, die auf nicht gekennzeichneten Blättern erfolgen, werden nicht gewertet. Sie dürfen auch die Rückseiten der Blätter mitbenutzen.* ***Wenn Sie für Ihre Antworten zusätzliche Blätter brauchen, erhalten Sie diese von den Betreuern.***

Bei Unklarheiten hinsichtlich der Aufgabenstellung*, melden Sie sich durch Handzeichen. Ein Betreuer kommt dann zu Ihrem Platz.*

Nicht mit Bleistift *sondern mit einem dokumentenechten Stift schreiben!*

Aufgabe 1. *Verschiedenes (8 Punkte)*

Bewerten Sie die Korrektheit der Aussagen mit **Ja** oder **Nein**. Es sind eventuell mehrere Aussagen richtig oder keine Einzige. Bewertungsschema:

- Eine *richtig* angekreuzte Aussage zählt +0,5 Punkte.
- Eine *nicht* angekreuzte Aussage zählt 0 Punkte.
- Eine *falsch* angekreuzte Aussage zählt –0,5 Punkte.

Man kann in der *gesamten* Aufgabe nicht weniger als 0 Punkte erreichen.

a) (2 Punkte)

Eine Referenzversion („Tag“) …		
… darf nur einmal pro Tag erstellt werden.	☐ Ja	☐ Nein
… sollte fehlerfrei sein.	☐ Ja	☐ Nein
… kann in SVN durch den Kopierbefehl erstellt werden.	☐ Ja	☐ Nein
… sollte aufgrund des hohen Speicherplatzbedarfs nicht allzu häufig erstellt werden.	☐ Ja	☐ Nein

b) (2 Punkte)

Refactoring …		
… ist die konsequente Änderung von Code wann immer es erforderlich ist	☐ Ja	☐ Nein
… setzt voraus, dass es automatisierte Tests der veränderten Funktionalität gibt	☐ Ja	☐ Nein
… sichert Verhaltenserhaltung durch Code Reviews	☐ Ja	☐ Nein
… setzt voraus, dass „Bad Smells“ automatisch erkannt werden können	☐ Ja	☐ Nein

c) (2 Punkte)

Zu den essentiellen Eigenschaften von Software-Komponenten gehört, dass sie …		
… auch „benutzte“ Interfaces spezifizieren	☐ Ja	☐ Nein
… alle Abhängigkeiten zu anderen Komponenten explizit machen	☐ Ja	☐ Nein
… Vererbung nutzen können	☐ Ja	☐ Nein
… objekt-orientiert implementiert sein müssen.	☐ Ja	☐ Nein

d) (2 Punkte)

Zu den essentiellen Eigenschaften von „agilen“ Softwareprozessen gehört, dass sie …		
… Kommunikation zwischen Entwicklern und Anwendern fördern	☐ Ja	☐ Nein
… extrem konsequent saubere / aktuelle Dokumentation erstellen	☐ Ja	☐ Nein
… extrem viel Refactoring und automatisierte Tests einsetzen	☐ Ja	☐ Nein
… im Entwurf zukünftige Anforderungsänderungen antizipieren und durch entsprechende Design Patterns vorbereiten	☐ Ja	☐ Nein

Aufgabe 2. *Klassen- und Objektdiagramme (15,5 Punkte)*

Zeichnen Sie Klassendiagramme, die folgende Sachverhalte modellieren. Notieren Sie auch die Multiplizitäten.

a) (**3 Punkte**) Zwischen einem Vermieter und einem Mieter besteht in der Regel eine Verbindung, die als „Vertrag“ bezeichnet wird. Mieter können mit einem oder auch mehreren Vermietern einen Vertrag haben, Vermieter können Mieter haben, müssen dies aber nicht.

b) (**2,5 Punkte**) Zeichnen Sie passend zu Ihrem Klassendiagram das Objektdiagramm welches beschreibt, dass Karl in seinem einzigen Haus je eine Wohnung an Maria und an Hans vermietet hat.

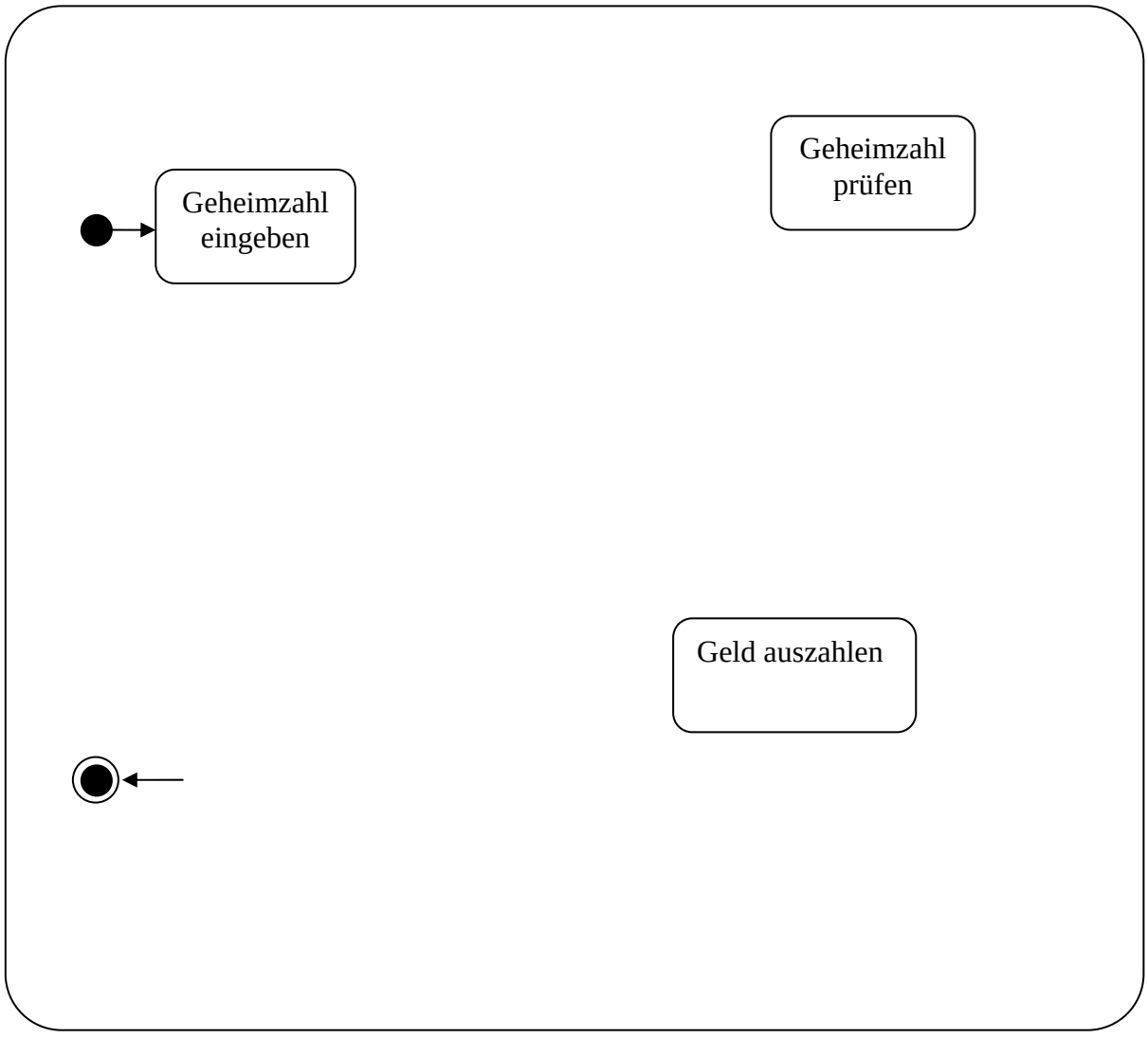
c) (**10 Punkte**) Ein Lied hat einen Komponisten und besteht aus zwei Nebenteilen und einem Hauptteil. Optional kann ein Mittelteil vorhanden sein. Alle Liedteile bestehen aus Taktteilen, die Noten oder Pausen sein können. Liedteile können nicht leer sein.

Aufgabe 3. *Aktivitätsdiagramme (6 Punkte)*

Vereinfacht stellt sich das Geldabheben an einem Automaten wie folgt dar:

- Der Kunde gibt dem Geldautomaten seine Geheimzahl ein.
- Der Kunde gibt an, wie viel Geld er abheben möchte.
- Der Geldautomat überprüft die Geheimzahl.
- Gleichzeitig überprüft der Automat die Liquidität des Kunden bei dessen Bank.
- Wenn der Kunde liquide ist und die korrekte Geheimzahl eingegeben hat, erhält er nacheinander das Bargeld und die Karte vom Automaten.
- Andernfalls erhält der Kunde eine Fehlermeldung angezeigt und danach die Karte zurück.

Vervollständigen Sie untenstehendes Aktivitätsdiagramm zu diesem Prozess.



Aufgabe 4. *Anwendungsfälle (14,5 Punkte)*

Sie sollen für den neu vorgestellten tragbaren Computer „iPad“ die Betriebsystem-Software weiterentwickeln. Dazu erhalten Sie folgende Funktionsbeschreibung:

> Der Benutzer entsichert seinen iPad, indem er mit dem Finger darüber streicht. Dabei wird sein Fingerabdruck analysiert. Wenn dabei festgestellt wird, dass der Fingerabdruck nicht dem Besitzer gehört, wird eine Sperrmeldung angezeigt.
>
> Nachdem der iPad entsichert wurde, kann der Benutzer Musik hören, Videos sehen oder elektronische Bücher lesen. In jedem Fall werden vor der Wiedergabe eines Mediums mit Hilfe eines externen Apple-Dienstes die digitalen Rechte geprüft.

a) (**10 Punkte**) Erstellen Sie für obige Notizen ein Use Case-Diagramm, das die relevanten Anwendungsfälle darstellt und untereinander in Beziehung setzt. Identifizieren Sie die relevanten Akteure. Wenn es sinnvoll ist, verwenden Sie jede der drei möglichen Beziehungen zwischen Anwendungsfällen. Bedingungen, für die es keine eigene Notation gibt, stellen Sie durch Notizen dar.

b) (**4,5 Punkte**) Fertigen Sie für den Anwendungsfall „iPad entsichern“ eine ausführliche textuelle Spezifikation an. Diese soll die folgenden Elemente beinhalten.

Name des Anwendungsfalls:

Akteure:

Anfangsbedingung:

Ereignisfluss:

Endbedingung:

Ausnahmen:

Aufgabe 5. *Analyse (11,5 Punkte)*

Im Rahmen der Anforderungserhebung für das Abspielen einer DVD in einem DVD-Player wurden folgende Objekte identifiziert:

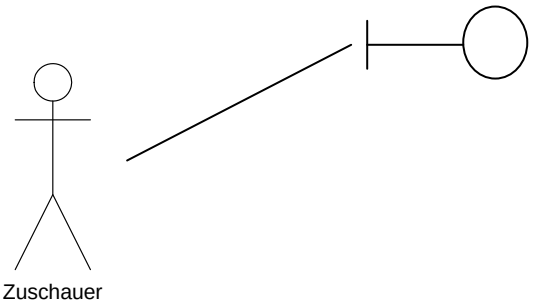
- DVD
- Kontrolltastenfeld
- Abspielsoftware
- Bildschirm

a) (**3,5 Punkte**) Klassifizieren Sie die Elemente in die Stereotypen des Analysemodells

b) (**8 Punkte**) Der Anwendungsfall „Film abspielen“ wird folgenderweise beschrieben:

Name:	Film abspielen
Akteure:	Zuschauer
Anfangsbedingung:	Die DVD befindet sich im Player.
Ereignisfluss:	1. Der Zuschauer drückt die Abspieltaste
	2. Der DVD-Player startet die Film-Wiedergabe
Endbedingung:	Der Film wird auf dem Bildschirm wiedergegeben.
Sonderfälle:	keine.

Vervollständigen Sie das folgende Kommunikationsdiagramm, so dass es die Interaktion des Akteurs mit den in der Aufgabenstellung genannten Objekten (DVD, ...) wiedergibt. Geben Sie dabei insbesondere auch die Nachrichten des Ereignisflusses an und verfeinern Sie den Ablauf von „DVD-Player startet die Film-Wiedergabe“ sinnvoll zu einer Interaktion die alle vier Objekte einbezieht.



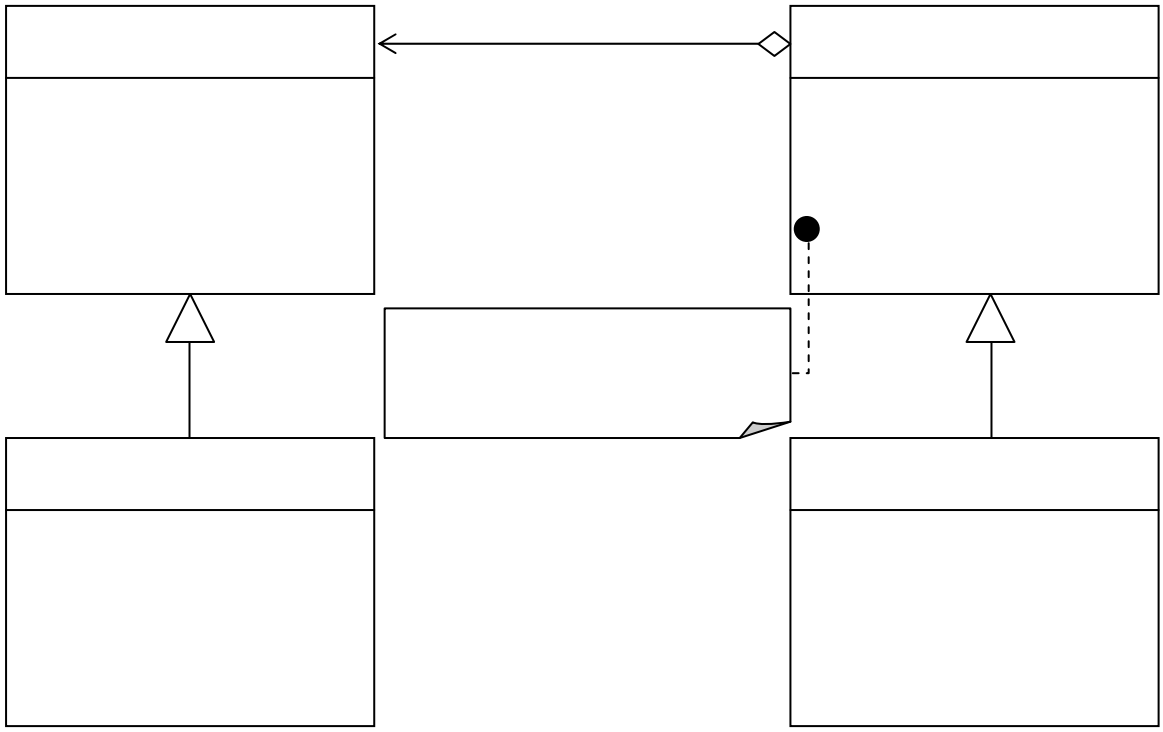
Aufgabe 6. *Entwurfsmuster theoretisch (9 Punkte)*

a) (**3 Punkte**) Verbinden Sie die folgenden Anwendungs-Beispiele jeweils mit dem Entwurfsmuster, das sich zur Implementierung anbietet:

- Eine *richtige* Verbindung zählt +0,5 Punkte.
- Eine *fehlende* Verbindung zählt 0 Punkte.
- Eine *falsche* Verbindung zählt –0,5 Punkte.

Steuerung von Anzeigen ○	○ Kommando (Command)
Zugriff auf entferntes Objekt ○	○ Stellvertreter (Proxy)
Anpassen von Schnittstellen ○	○ Beobachter (Observer)
Erzeugung von Objekten ○	○ Adapter
Kapselung von Undo-Aktionen ○	○ Besucher (Visitor)
Operation auf Elementen einer Objektstruktur ○	○ Fabrikmethode (Factory-Method)

b) (**6 Punkte**) Ergänzen Sie in folgender allgemeiner Darstellung des Bridge-Entwurfsmusters die Klassennamen, Methodensignaturen, Stereotypen und das als Notiz eingezeichnete Codefragment. Jeder Klassen- und Methodennamen soll dabei die Rolle des jeweiligen Elementes im Entwurfsmuster ausdrücken.



Aufgabe 7. *Entwurfsmuster angewandt (11 Punkte)*

Ein digitaler *Musikspieler* kann *Musik* abspielen. Dazu ruft der Musikspieler die *spiele()*-Methode der Musik auf. Musik ist entweder ein einzelnes *Musikstück* oder eine *Abspielliste* (Playlist), die wiederum Musik enthalten kann.

a) (**7 Punkte**) Modellieren Sie den angegebenen Sachverhalt mit Hilfe eines geeigneten Entwurfsmusters und zeichnen Sie das entsprechende Klassendiagramm. Geben Sie auch den Code der *spiele()*-Methode der Klasse Playlist an (als Notiz).

b) (**2 Punkte**) Geben Sie zu jeder Klasse aus (a) die Rollen an, die sie in dem Entwurfsmuster spielt (sie können sie direkt an die jeweilige Klasse als Notiz anfügen).

c) (**2 Punkte**) Erläutern Sie allgemein, in welchen Fällen das Entwurfsmuster angewandt werden kann.

Aufgabe 8. *CRC-Karten und Design by Contract (9 Punkte)*

In der Informatik-Fachschaft können Prüfungsprotokolle in Ordnern ausgeliehen werden. Dabei darf jede Studierende, die durch ihre Matrikelnummer identifizierbar ist, jederzeit nur einen Ordner ausleihen, und jeder Ordner kann jederzeit an maximal eine Studierende ausgeliehen werden.

a) (**7 Punkte**) Als Objekte wurden bereits *Fachschaft*, *Studierende,* und *Ordner* identifiziert. Unter Einsatz von CRC-Karten soll das Objektmodell um Verantwortlichkeiten und Kollaborationen erweitert werden, und zwar so, dass die Klassen anschließend alles beinhalten, was für folgende Szenarien erforderlich ist:

   a. Erfolgreiches Ausleihen eines Ordners
   b. Verweigertes Ausleihen eines Ordners

Klasse: Fachschaft **Beschreibung:**	
Verantwortlichkeiten (Responsibilities)	**Zusammenarbeit (Collaboration)**

Klasse: Studierende **Beschreibung:**	
Verantwortlichkeiten (Responsibilities)	**Zusammenarbeit (Collaboration)**

Klasse: Ordner **Beschreibung:**	
Verantwortlichkeiten (Responsibilities)	**Zusammenarbeit (Collaboration)**

b) (**2 Punkte**) Geben Sie die Vor- und die Nachbedingung der „Entleihe“-Operation an. Definieren Sie dazu – in Übereinstimmung mit Ihrer Lösung aus (a) – sinnvolle Methodensignaturen und bilden Sie daraus logische Ausdrücke in Java-Syntax.

context Fachschaft::entleihe(Studierende s, Ordner o)

a. **pre**:

b. **post:**