Examination
Software Engineering (SE)

Prof. Dr.-Ing. Christian Heller <christian.heller@ba-leipzig.de>

Personal Data	
First and Last Name	Ihre Daten müssen von der Klausuraufsichtsperson schriftlich auf der Anwesenheitsliste festgehalten werden!
Matriculation Number	
Subject and Year	CS 2014
Login	Das Login "klaus..." muss unbedingt schriftlich auf der Anwesenheitsliste festgehalten werden, da sonst keine Zuordnung des Logins zu Ihrem Namen und damit keine Korrektur der Klausur möglich ist!

Examination Data	
Date	2016-06-23
Duration [min]	120
Maximum Points [Credit Point]	100
Permitted Study Aids	Dokumentation im lokalen Netzwerkverzeichnis (Intranet); NICHT gestattet sind Kommunikationsmöglichkeiten (Internet) oder Anmeldung via SSH auf dem Rechner "fileserv", wo Ihr Homeverzeichnis liegt, oder eine Anmeldung mit Ihrem Klausur-Login nach Ende der Prüfung. Dies kann leicht geprüft werden (last cs12*, Server-Log-Dateien). Bitte unterlassen Sie also Täuschungsversuche in Ihrem eigenen Interesse.
Remarks	Hinterlegen Sie alle Programme und Antworten in elektronischer Form! Es wird kein Papier angenommen. Möchten Sie Lösungen erläutern, so nutzen Sie Quelltext-Kommentare oder legen eine Text-Datei an. Speichern Sie sämtliche Daten im HOME-Verzeichnis des Nutzers, d.h. unter Windows auf Laufwerk H:\ (NICHT auf C:\ oder "Eigene Dateien")! Idealerweise legen Sie dort ein Unterverzeichnis namens "klausur" an. Lesen Sie die Aufgaben komplett durch, bevor Sie sie lösen! Die Reihenfolge der Lösung ist Ihnen überlassen. Probieren Sie immer, eine Aufgabe zu lösen, da auch auf richtige Teile nicht vollständiger Lösungen Punkte vergeben werden! Falls vom Prinzip her richtig, so werden auch alternative Lösungen akzeptiert. Sie dürfen beliebig viele Bildschirmausgaben von Werten in den Quelltext einbauen, um ein Programm besser nachvollziehen zu können. Diese Aufgabenstellung in Papierform können Sie nach dem Ende der Klausur behalten.

Evaluation											
Task	1	2	3	4	5	6	7	8	9	10	Summe
Desired Value [Credit Point]	20	10	30	20	20	(5)	0	0	0	0	100
Actual Value [Credit Point]											

Im Rahmen der Klausur sind sechs Aufgaben verschiedener Themen, immer aber mit Bezug zur *Unified Modeling Language* (UML) zu lösen. Insofern modelliert werden soll, ist das *BOUML*-Werkzeug zu verwenden.

Task 1: Structure [20 Point]

Modellieren Sie den im Folgenden beschriebenen Sachverhalt in einem UML-Klassendiagramm und geben Sie dabei neben den Strukturelementen auch die beschriebenen Beziehungen einschließlich Kardinalitäten sowie erwähnte Methoden an! Verwenden Sie die in der Aufgabenstellung benutzten Begriffe für die Bezeichnungen/Namen!

a) Im Skilanglauf unterscheidet man als *Technik*: *Klassisch*, *Skating*, *Abfahrt*. [2]

b) Ein *Skilaeufer* kann auf eine dieser Techniken spezialisiert sein. [2]

c) Außerdem geht der *Skilaeufer* regelmäßig *trainieren*, wobei ihm eine *Strecke* in Form einer Kilometerzahl vorgegeben wird. Falls nicht, so absolviert er seine Lieblingsstrecke von 5 km. [2]

d) Passen Sie die Diagrammeinstellungen so an, dass alle Details (Parameter etc.) von Klassenmitgliedern angezeigt werden! [2]

e) Der *Skilaeufer* hat nur ein Paar *Schuhe* und ein Paar *Stöcke*. [2]

f) Bei den *Ski* (hiermit ist immer ein Paar gemeint, was nicht näher zu berücksichtigen ist) jedoch darf er wählerisch sein: Neben den standardmäßig immer vorhandenen *wettkampfSki* darf er optional noch maximal zwei *trainingsSki* besitzen. [2]

g) Ski können, aber müssen nicht, mit mehreren Arten *Wachs* präpariert sein. [2]

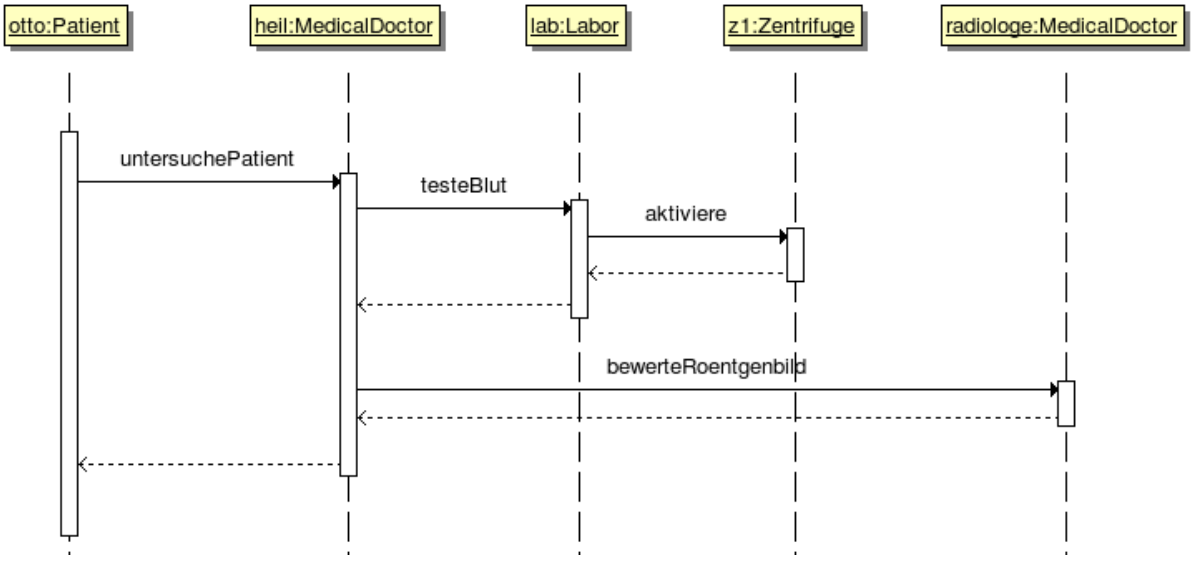
h) Im Allgemeinen unterscheidet man zwei Arten von Wachs: Gleitwachs und Steigwachs, wobei letzteres Hartwachs oder Klister sein kann. [2]

i) Während das Wachsen der Ski optional ist, müssen sie zwingend genau eine Bindung haben, da sie sonst nicht nutzbar wären. [2]

j) Spezielle Formen einer Bindung sind Riemen-, Bügel- und Schnabelbindung. [2]

Task 2: Behaviour [10 Point]

Gegeben sei folgendes Sequenzdiagramm, welches die Konsultation eines Arztes durch einen Patienten darstellt:



a) Überführen Sie das Sequenzdiagramm in ein äquivalentes Kommunikationsdiagramm! [8]

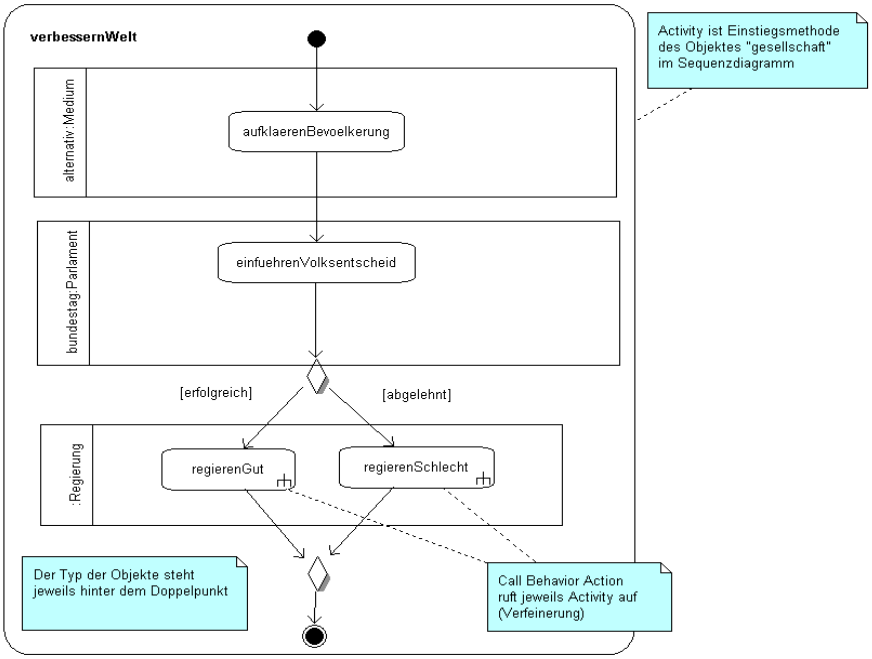
[4 auf Objekte bzw. Lebenslinien; 4 auf Links]

b) Passen Sie die Diagrammeinstellungen so an, dass bei der Darstellung der Zahlen die Aufrufhierarchie berücksichtigt wird! [2]

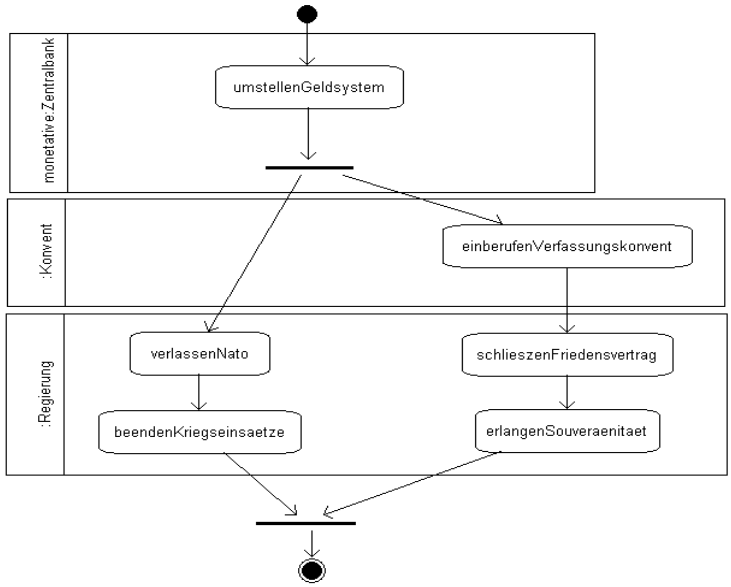
Task 3: Diagramme Transformation [30 Point]

Gegeben seien zwei Aktivitätsdiagramme, welche eine Weltverbesserungsvision beschreiben.

Aktivitätsdiagramm zur Aktivität "verbessernWelt":



Aktivitätsdiagramm zur Aktivität "regierenGut":



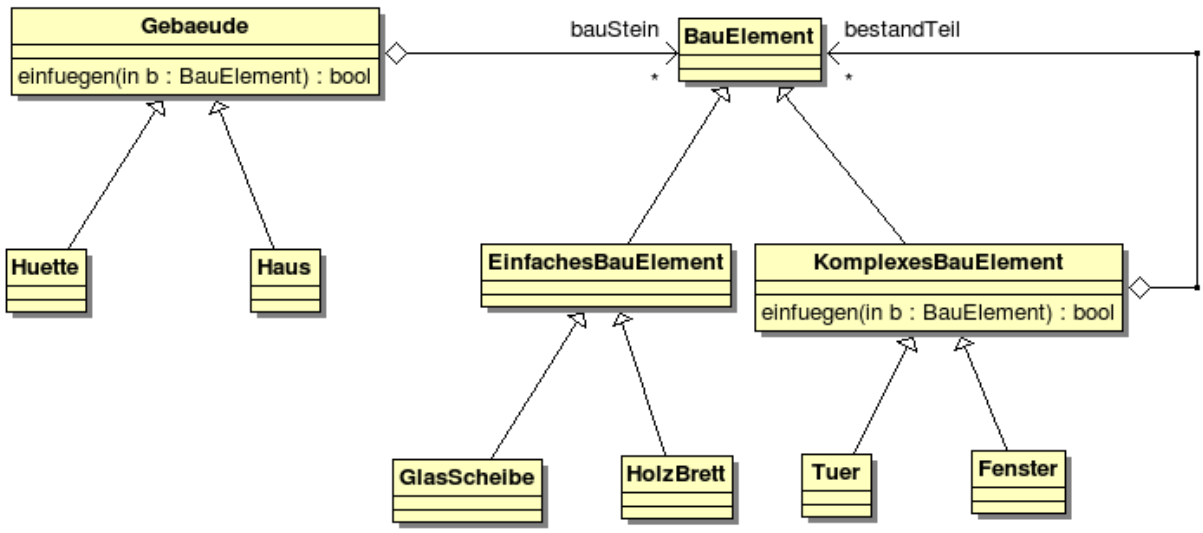
Überführen Sie die Aktivitätsdiagramme in EIN verhaltensgleiches Sequenzdiagramm! [30]

[10 auf Objekte (Name + Typ); 10 auf Botschaften; 10 für Elemente der Flusskontrolle]

Hinweise: Die Rückkehrpfeile von Methoden können weggelassen werden. Mögliche Details der Methode "regierenSchlecht" (etwa *abschaffenBargeld* oder *einfuehrenDiktatur*) sind ebenfalls wegzulassen.

Task 4: Roundtrip Engineering [20 Point]

Gegeben sei folgendes Klassendiagramm, welches die Struktur von Gebäuden wiedergibt:



a) Ordnen Sie die gezeigten Klassen wie folgt Quelltext-Artefakten zu [5]:

Artefakt	Klassen
Wohnstaette	Huette, Haus
Bauwerk	Gebaeude
Einzelteil	GlasScheibe, HolzBrett
Baugruppe	Tuer, Fenster
Baustein	BauElement, EinfachesBauElement, KomplexesBauElement

b) Stellen Sie die Artefakte in einem Verteilungsdiagramm dar und machen Sie die durch die Klassen verursachten Abhängigkeiten zwischen ihnen durch entsprechende Pfeile kenntlich! [5]

c) Versehen Sie die Artefakte mit einem Kommentar zu ihrer Beschreibung! Erzeugen Sie eine Dokumentation (nur zum Verteilungsdiagramm) im HTML-Format mit flacher Hierarchie und Vektorgrafiken! [5]

d) Generieren Sie durch *Forward Engineering* gemäß *Model Driven Architecture* (MDA) den zugehörigen Java-Quelltext mit den üblichen Standardfeldern! [5]

Task 5: Terminology [20 Point]

In dieser Aufgabe wird ergänzendes Grundlagenwissen abgeprüft.

a) Geben Sie sechs Aufgaben eines CASE-Werkzeuges an! [3]

b) Benennen Sie Einsatzschwerpunkte für die folgenden UML-Diagramme! [7]

Use Case Diagram (UCD), Class Diagram (CD), Component Diagram (CMD), Deployment Diagram (DD), Activity Diagram (AD), Sequence Diagram (SD), Communication Diagram (COD)

c) Benennen und erläutern Sie Gemeinsamkeiten von und Unterschiede zwischen: Assoziation, Rolle, Aggregation, Komposition! [2]

d) Wie nennt man die Erstellung eines UML-Diagrammes, welches ein Element eines bereits vorhandenen Diagrammes näher beschreibt? [2]

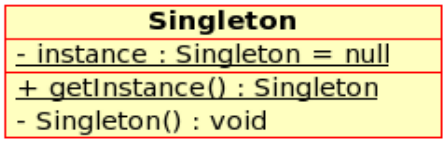
e) Was sind und wozu dienen Profile in der UML? [2]

f) Nennen Sie vier Teile der UML-Spezifikation! [2]

g) Wie nennt sich das zum *UML Meta Model* gehörige *Meta Meta Model*? [2]

Additional Task 6: Software Pattern [5]

Hier geht es um das Verstehen von Software-Mustern. Gegeben sei deren eines, genannt *Singleton*-Entwurfsmuster (siehe Abbildung):



Schreiben Sie eine mögliche Implementierung in der Programmiersprache Java nieder in eine Datei namens "Singleton.java"! [5]

Hinweise: Nebenläufigkeit (Multi-Threading) ist zu vernachlässigen. Alle korrekten Varianten werden als richtig gewertet (Lazy- oder Eager Initialisation etc.).

Viel Erfolg!