Furhan Hidayet Rafet Aydin 1638039
Stefan Stracke 1630343

Hochschule Hannover
University of Applied Sciences and Arts

Software Engineering 1
Übungsblatt 4
- Anforderungsanalyse -

Ausgabe: 01.04.2022 **Abgabe: 07.04.2022**

Ablauf der Übungen
- Erfolgreiches Bearbeiten von 2/3 der Pflicht-Übungsaufgaben und persönliche Vorstellung mindestens einer Aufgabenlösung (auf Nachfrage) berechtigen zur Klausurteilnahme.
- Eine Aufgabe gilt als erfolgreich bearbeitet, wenn eine korrekte Lösung bzw. ein nachvollziehbarer Lösungsversuch termingerecht abgegeben wurden.
- Die Aufgaben sollen in 2er Gruppen bearbeitet werden.
- Die Abgabe erfolgt per Moodle vor der Besprechung in der Übung (bitte als eine PDF-Datei).
- Das Ergebnis soll in der Übung vorgestellt werden.

Aufgabe 4.1: Unified Process
Im Unified Process wird ein Software-System inkrementell in aufeinander aufbauenden Iterationen (Stufen bzw. Versionen) entwickelt. Bei einem neuen Software-Entwicklungsprojekt stellt sich nun die Frage, in wie viele Iterationen es unterteilt werden soll. Beschreiben Sie die **Vor-** und **Nachteile** von sehr **großen**, d.h. eher wenige, Iterationen bzw. sehr **kleinen**, d.h. eher viele, Iterationen.

Aufgabe 4.2: Anforderungsanalyse – Pflichtaufgabe
Führen Sie für das im folgenden beschriebene System eine Anforderungsanalyse durch und fügen Sie alle erstellten Ergebnisse in einem Dokument als **Anforderungsmodell** (bzw. Pflichtenheft**)** zusammen.

a) Erstellen Sie ein **Use Case Diagramm**. Finden Sie dazu alle Akteure, welche das System benutzen und bestimmen Sie alle Use Cases (Anwendungsfälle).
Strukturieren Sie das Use Case Diagramm ggf. mit Hilfe von `<<include>>`- und `<<extend>>`-Beziehungen!
b) **Beschreiben** Sie die zwei wichtigsten Use Cases.
c) Erstellen Sie ein einfaches **Domänenmodell** !
d) Bestimmen Sie beispielhaft die wichtigsten **nichtfunktionalen Anforderungen** für das System.

Modulbelegungsmanagementsystem (ModMS)

Informationssysteme zum Management von Modulen, Modulbelegungen und Prüfungsleistungen befinden sich seit vielen Jahren an deutschen Hochschulen im Einsatz. Jedoch entsprechen viele dieser Systeme nicht mehr auf dem neuesten fachlichen und technischen Stand. Dies führt zu unnötigem manuellem Aufwand und wenig Begeisterung bei den unterschiedlichen Nutzergruppen.

Aus diesem Grund soll im Projekt ModMS (Modulbelegungsmanagementsystem) ein neues, integriertes Informationssystem für das Modul- und Prüfungsmanagement einer Hochschule entwickelt werden, das die kompletten Aufgaben der Modul- und Prüfungsverwaltung für Bachelor- und Masterstudiengänge fachlich flexibel und technisch auf dem neuesten Stand realisiert.

Ein Systemoperator bildet die Bedingungen der aktuell geltenden Prüfungsordnung studiengangsspezifisch im System ab. Wichtig ist hierbei die Vollständigkeit der Abbildung der Modulbeschreibungen sowie der Flexibilität bei der Formulierung von prüfungstechnischen Restriktionen (z.B. gewisse Module setzen andere Leistungen voraus oder das Kolloquium zur Bachelor-/Masterarbeit kann erst nach der letzten Prüfungsleistungen stattfinden). Das System soll natürlich für die gesamte

Hochschule gelten, also die unterschiedlichen Studiengänge für die verschiedenen Fakultäten, deren Prüfungsordnungen als auch die beiden Studienphasen Bachelor und Master verwalten.

Zum Start eines jeden Semesters legt das zuständige Studiendekanat die angebotenen Module für das anstehende Semester im System an. Studierende müssen sich zu Semesterbeginn für die Module definitiv anmelden, die sie in dem Semester belegen möchten. Die Anmeldung für ein Modul impliziert automatisch auch die Anmeldung zur Prüfung für dieses Modul. Das Studiendekanat legt den Anmeldezeitraum sowie den letztmöglichen Termin zur Abmeldung für ein Modul fest. Bei der Anmeldung für ein Modul prüft das System, ob der/die Studierende die in der Prüfungsordnung definierten Voraussetzungen erfüllt hat. D.h. das System stellt sicher, dass sich Studierende nur zu den Modulen anmelden können, zu denen sie auch formal zugelassen sind. Nach dem Abmeldetermin erlaubt das System keine Abmeldungen.

Um dem Studiendekanat und den Studierenden einen reibungslosen und wenig aufwändigen Anmeldeprozess zu ermöglichen, können sich die Studierenden selbst entweder über einen Internet-Zugang per Web-Browser oder über die ModMS-App von einem Smartphone aus komfortabel zu Modulen an- und abmelden. Hierbei werden ihnen vom System nur die Anmeldeoptionen angeboten, die dem aktuellen Studienstatus der Studierenden entsprechen. Die ModMS-App soll auf den beiden zurzeit am weitesten verbreiteten mobilen Betriebssystemen lauffähig sein.

Studienbescheinigungen mit einer Übersicht über den aktuellen Stand ihrer Studienleistungen können sich Studierende entweder direkt online ausdrücken und vom Studiendekanat in Papierform ausgedruckt bekommen.

Weiterhin werden vom Systemoperator alle Benutzer des Systems verwaltet, also angelegt, geändert oder gelöscht. Dabei weist der Systemoperator jeder Benutzergruppe bestimmte Rechte für das ModMS-System zu, z.B. ist die Eingabe von Noten nur der Gruppe der Prüfungsberechtigten oder dem Studiendekanat möglich.

Das System verwaltet auch die Krankmeldungen von Studierenden im Prüfungszeitraum. Die Krankmeldungen müssen im Prüfungsamt in Papierform abgegeben werden und werden dort im System erfasst.

Die Notenverbuchung für Modulprüfungen im System kann jede/r Prüfungsberechtigte selbst web-basiert vornehmen, natürlich nur für jeweils eigene Prüfungen. Prüfungsberechtigt sind Professoren, WiMis und Lehrbeauftragte. Es ist noch möglich, die eingegebenen Noten zu ändern, bis das Studiendekanat den Prüfungszeitraum offiziell abschließt. Das System generiert automatisch kumulierte Notenübersichten einer Prüfung, die die/der Prüfungsberechtigte online einsehen oder als PDF herunterladen kann. Über Authentifizierung und Verschlüsselung gewährleistet das System die notwendige Sicherheit. Das Studiendekanat ist berechtigt, alle Funktionalitäten der Prüfungsberechtigte auszuführen.

Aufgrund der Vertraulichkeit von Prüfungsergebnissen liegt ein besonderer Schwerpunkt des Projekts im Bereich Sicherheit und Datenschutz. Systemtechnisch ist sichergestellt, dass ein/e Studierende*r nur ihre/seine persönlichen Prüfungsergebnisse einsehen kann sowie ein/e Prüfungsberechtigte nur die Noten der ihr/ihm zugeordneten Modulen. Lediglich das Studiendekanat kann sämtliche Noten einsehen und ändern.

Aufgabe 4.1: Unified Process

Im Unified Process wird ein Software-System inkrementell in aufeinander aufbauenden Iterationen (Stufen bzw. Versionen) entwickelt. Bei einem neuen Software-Entwicklungsprojekt stellt sich nun die Frage, in wie viele Iterationen es unterteilt werden soll. Beschreiben Sie die **Vor-** und **Nachteile** von sehr **großen**, d.h. eher wenige, Iterationen bzw. sehr **kleinen**, d.h. eher viele, Iterationen.

große Iterationen

Vorteile:

- Beschäftigung hoch (Mitarbeiter haben zu tun)

Nachteile:

- Ergebnisse können nicht schnellstmöglich getestet werden
- zu viele Use Cases parallel

kleine Iterationen

Vorteile:

- Prototyp kann schnell erstellt werden

Nachteile:

- Nicht alle Mitarbeiter sind Beschäftigt

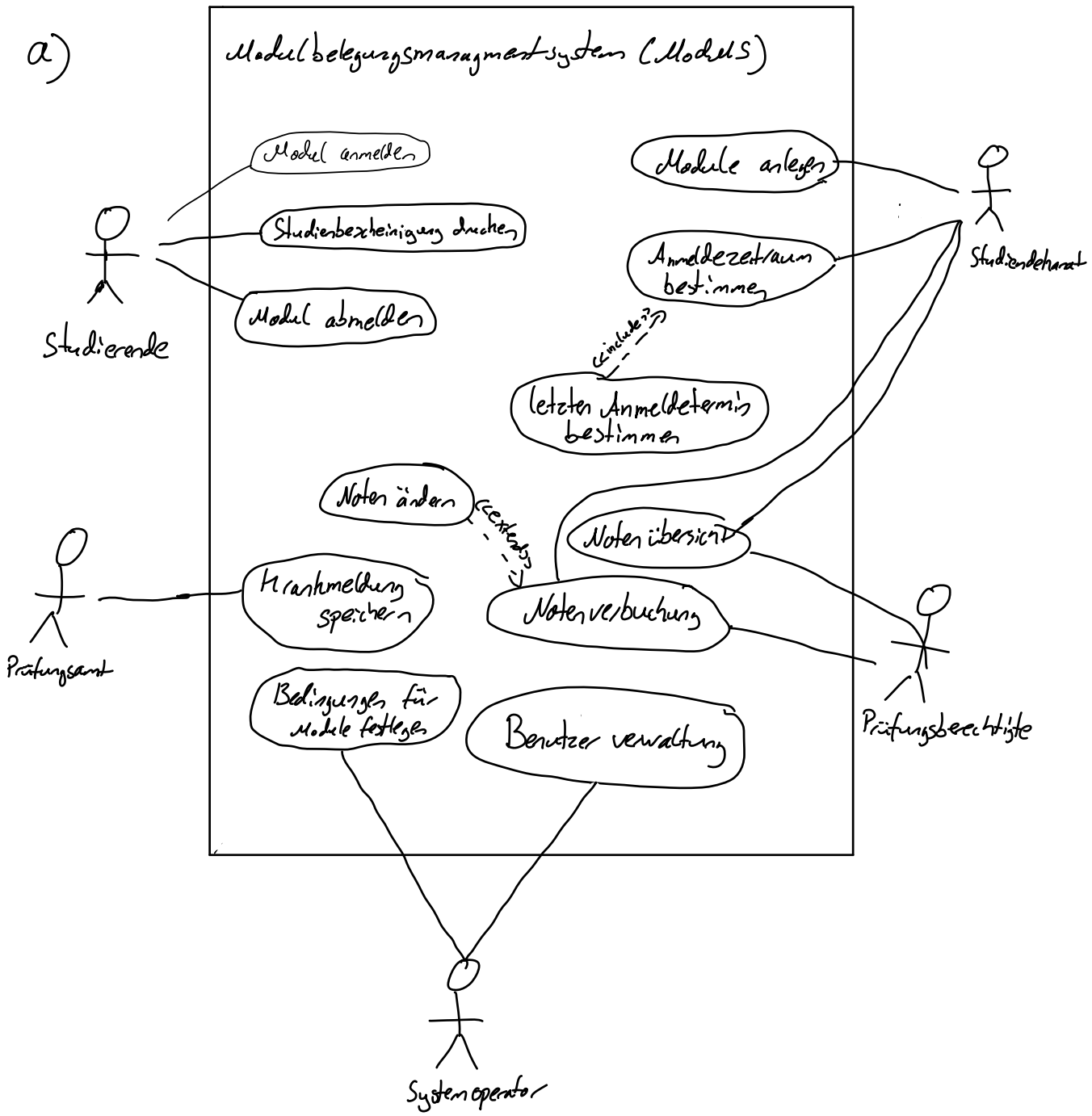
Aufgabe 4.2: Anforderungsanalyse – Pflichtaufgabe
Führen Sie für das im folgenden beschriebene System eine Anforderungsanalyse durch und fügen Sie alle erstellten Ergebnisse in einem Dokument als **Anforderungsmodell** (bzw. Pflichtenheft**)** zusammen.

a) Erstellen Sie ein **Use Case Diagramm**. Finden Sie dazu alle Akteure, welche das System benutzen und bestimmen Sie alle Use Cases (Anwendungsfälle).
Strukturieren Sie das Use Case Diagramm ggf. mit Hilfe von `<<include>>`- und `<<extend>>`-Beziehungen!

b) **Beschreiben** Sie die zwei wichtigsten Use Cases.

c) Erstellen Sie ein einfaches **Domänenmodell** !

d) Bestimmen Sie beispielhaft die wichtigsten **nichtfunktionalen Anforderungen** für das System.



5)

Use Case: Modul anmelden Akteur: Studierende

Vorbedingung: Studierender erfüllt Voraussetzung für Teilnahme

Ereignisfluss:

1. Studierende klickt auf Modul anmelden
2. System speichert gewähltes Modul
3. System prüft Anmeldefrist
4. Student erhält Textfeld
5. Anmeldung erfolgreich. Use Case beendet

Alternativen: zu 4: Anmeldefrist verpasst. Anmeldung nicht möglich

Nachbedingung: —

Use Case: Modul anlegen Akteur: Studiendekanat

Vorbedingung: —

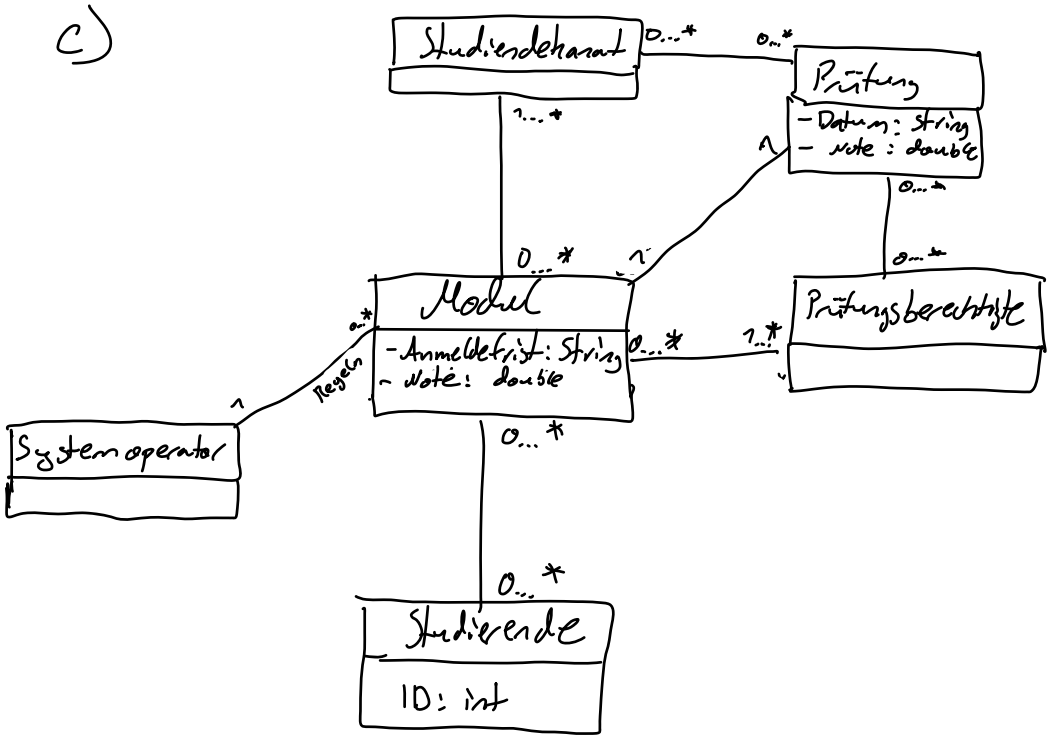
Ereignisfluss:

1. Modul anlegen klicken
2. Modulnamen eintragen
3. System speichert Modulnamen
4. Anmeldefristen festlegen
5. System speichert Fristen
6. Modul wird angelegt. Use Case beendet

Alternativen: —

Nachbedingung: —

c)



d) wichtigsten nichtfunktionalen Anforderungen:

- Sicherheitsanforderungen
- Datenschutzanforderungen
- Zuverlässigkeit