Rheinische Friedrich-Wilhelms-Universität Bonn
Institut für Informatik III

Übungen zur Vorlesung Softwaretechnologie

- Wintersemester 2019/20 -

Dr. Günter Kniesel

Übungsblatt 8

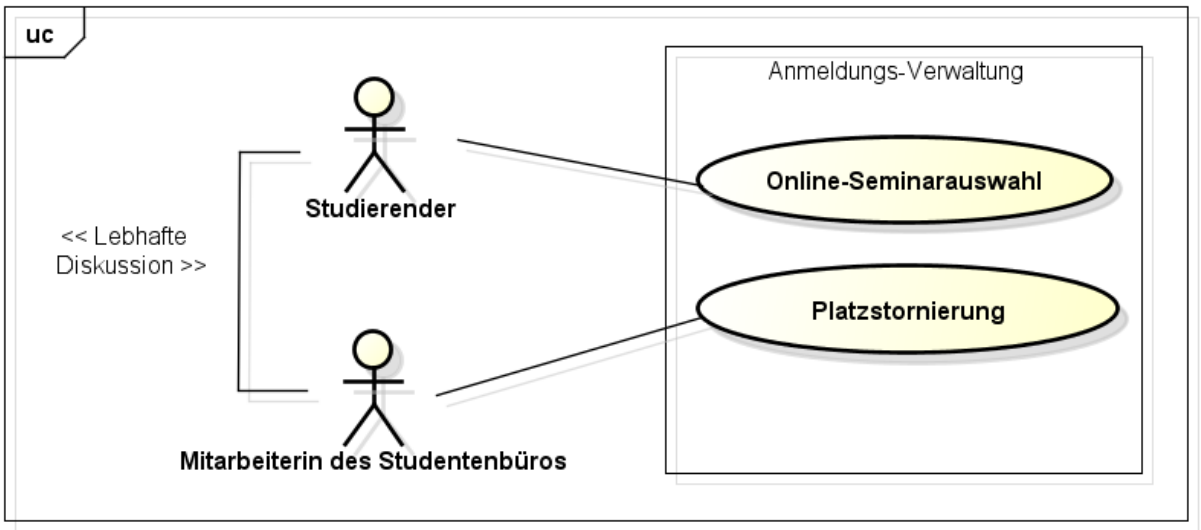
Zu bearbeiten bis: 29.11.2019, 16 Uhr

Bitte fangen Sie **frühzeitig** mit der Bearbeitung an, damit wir Ihnen bei Bedarf helfen können. Checken Sie die Lösungen zu den Aufgaben bitte in Ihr Repository ein, „Erklärungen" bitte als Textdatei. Fragen zu Übungsaufgaben respektive zur Vorlesung können Sie auf der Mailingliste swt-tutoren@lists.iai.uni-bonn.de, bzw. swt-vorlesung@lists.iai.uni-bonn.de stellen.

Praktische Aufgaben

Aufgabe 1. *Analysephase: Analyse-Objektmodell* **10 Punkte**

Sie sind mit dem Design eines Systems zur Seminarauswahl im Studienbüro beauftragt. Als Vorgabe erhalten Sie die zwei unten eingezeichneten Anwendungsfälle:



Die verbale und nonverbale Kommunikation zwischen Studierender und Mitarbeiterin ist hier nur durch die Beziehung `<<Lebhafte Diskussion>>` angedeutet. Sie wird nicht weiter modelliert, denn sie läuft nicht über das System.

Die Beschreibungen der Anwendungsfälle finden Sie auf der nächsten Seite.

Hinweis: Wenn Ihnen die Vorgehensweise nicht klar ist, sehen Sie sich die Folien zur Objektmodellierung im Analyseworkflow aus der Vorlesung an.
Beschreibungen der Anwendungsfälle:

Name des Anwendungsfalls: Online-Seminarplatzwahl

Akteure: Studierender

Anfangsbedingung:

Dieser Anwendungsfall beginnt, wenn der Studierende die URL der Online-Seminarwahl aufruft.

Ereignisfluss:

1. Studierender sucht Seminar im System.
2. Studierender wählt ein Seminar aus.
3. System zeigt Verfügbarkeit von Plätzen für das ausgewählte Seminar an und öffnet eine Eingabemaske für die Anmelde-Daten.
4. Studierender gibt Studiengang, Semesterzahl und Matrikelnummer an.
5. System reserviert Platz.
6. System druckt Bestätigung aus

Endbedingung:

Studierender ist für das Seminar registriert, die Anzahl der im Seminar freien Plätze ist um eins weniger als vorher und es wurde eine Reservierungs-Bestätigung gedruckt. (Wir hoffen auch sehr, dass sie vom Studenten mitgenommen wurde um sie bei Bedarf später vorzeigen zu können, aber das kann das System nicht selbst sicherstellen – vielleicht in der Ausbaustufe 2050, wo dann der KI-gesteuerte Drucker ihm den Ausdruck per Greifarm direkt in die Hand gibt – das dürfen aber dann Ihre Kinder modellieren).

Sonderfälle: keine.

Spezielle Anforderungen: Dem Studierenden steht ein Drucker zur Verfügung.

Name des Anwendungsfalls: Platzstornierung

Akteure: Studierender, Mitarbeiterin des Studienbüros

Anfangsbedingung:

Dieser Anwendungsfall beginnt, wenn der Studierende das Studienbüro betritt.
Der Studierende besitzt bereits eine Bestätigung für ein Seminar.

Ereignisfluss:

1. Mitarbeiterin nimmt die Reservierungs-Bestätigung des Studierenden entgegen.
2. Mitarbeiterin sucht Seminar im Stornierungs-System.
3. Mitarbeiterin sieht sich die Belegungsübersicht des Seminars an und wählt den Platz des Studierenden zur Freigabe aus
4. Mitarbeiterin bestätigt Platz-Freigabe
5. System gibt reservierten Platz wieder frei.
6. Mitarbeiterin vernichtet die Reservierungs-Bestätigung.

Endbedingung:

Der Platz ist wieder frei und der Studierende ist nicht mehr im Seminar registriert. Wir hoffen sehr, dass die Mitarbeiterin die Reservierungs-Bestätigung vernichtet hat, können das aber nicht garantieren.

Sonderfälle: keine.

Spezielle Anforderungen: keine.

a) **(4 Punkte)** Identifizieren Sie für jeden der beiden Anwendungsfälle jeweils Boundary-, Controller- und Entity-Objekte sowie die beteiligten Akteure. Es reicht, sie pro Anwendungsfall textuell aufzuzählen.

Online-Seminarplatzwahl

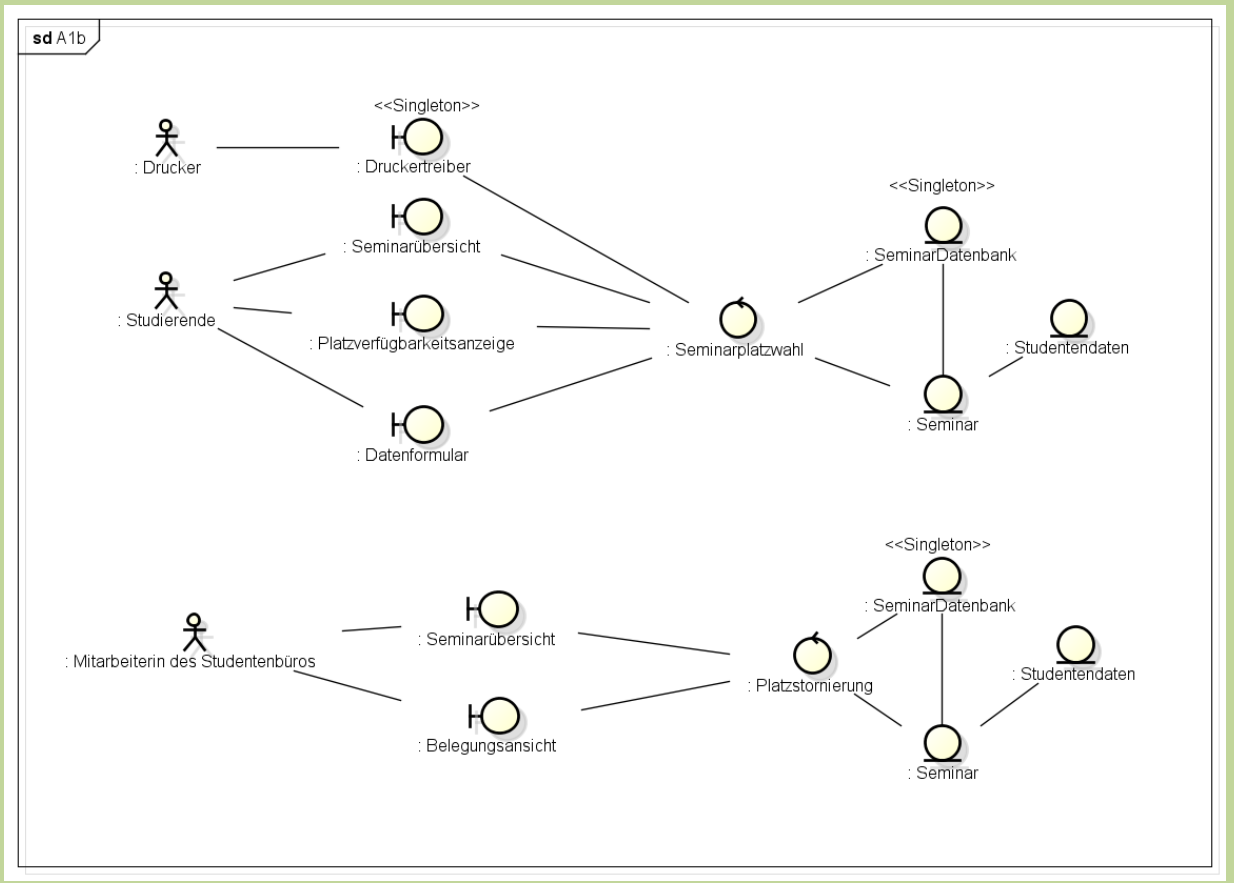
- Boundary
  - Seminarübersicht
  - Platzverfügbarkeits-Anzeige
  - Eingabemaske für Anmeldedaten
  - Druckertreiber
- Controller
  - Seminarplatzwahl
- Entity
  - Seminar
- Akteure
  - Studierender
  - Drucker

Platzstornierung

- Boundary
  - Seminarübersicht
  - Belegungsansicht
- Controller
  - Platzstornierung
- Entity
  - Seminar
- Akteure
  - Studierender
  - Mitarbeiterin

Bewertung: Pro korrekter Gruppe (Boundaries, Controller, Entities, Akteure) je 1P.

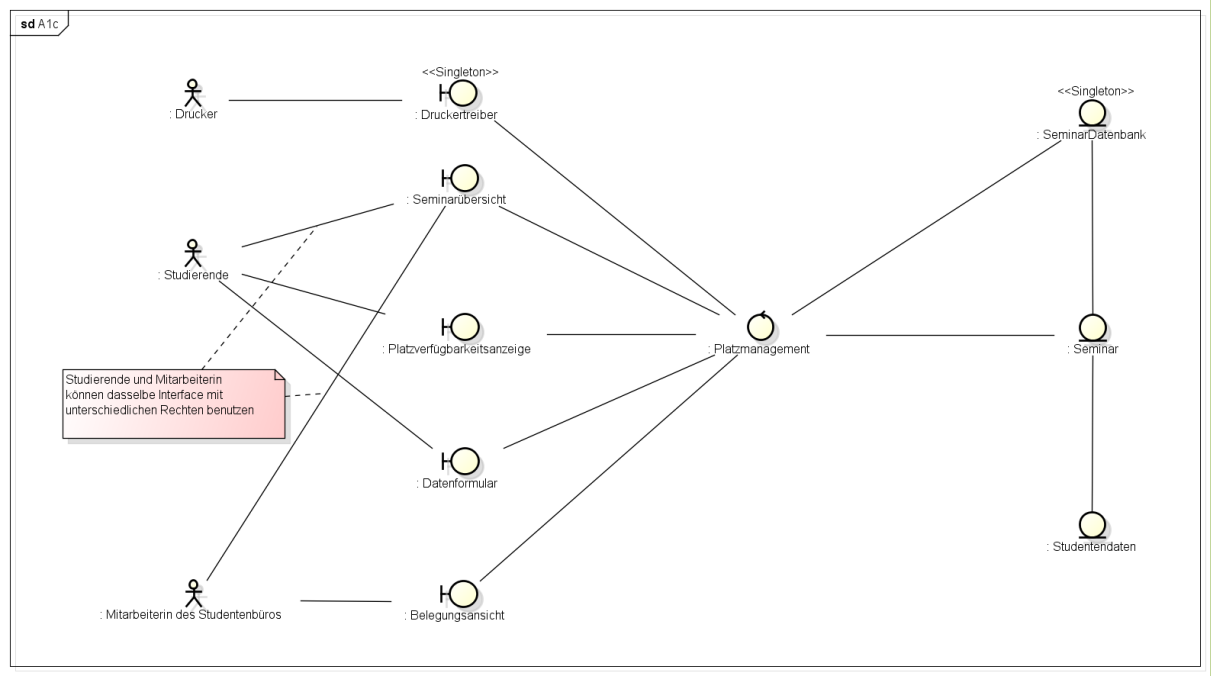
b) **(3,5 Punkte)** Setzen Sie die Ergebnisse aus Teilaufgabe a) in einem Kommunikationsdiagramm um, das lediglich die beteiligten Objekte und ihre Beziehungen erfasst. Nachrichten sind in diesem Schritt nicht erforderlich.



Pro Gruppe von Interaktionen (Akteur – Boundary, Boundary – Controller, Controller – Entity, Bonus: Entity – Entity) 1P + 0,5 Bezeichner = 3,5P + 1 Bonus

Im obigen Diagramm ist die Klasse *SeminarDatenbank* nicht im wörtlichen Sinne als Datenbank zu interpretieren, sondern einfach als eine Klasse, die die Menge aller Seminare verwaltet. Sie hätte auch *Veranstaltungsverwaltung* o.ä. heißen können. Wichtig ist hier, dass diese Klasse eine Methode *getSeminars()* bietet, mit der sich der Controller die Seminare holen kann, die in der Suchmaske angezeigt werden sollen (siehe Kommunikationsdiagramm zur nächsten Aufgabe, Schritt 1.2.2). Ob es eine Datenbank gibt, die alle oder einen Teil der Entitys persistent macht, ist ein Aspekt, den wir erst beim Systementwurf betrachten. Das Analysemodell soll unabhängig von solchen Entwurfsentscheidungen sein (→ plattformunabhängig / portabel).

c) **(2,5 Punkte)** Erstellen Sie ein neues Diagramm in dem Sie die Analyseobjekte aus Teilaufgabe b), die ähnliches tun bzw. konzeptuell identisch sind, zusammenfassen.

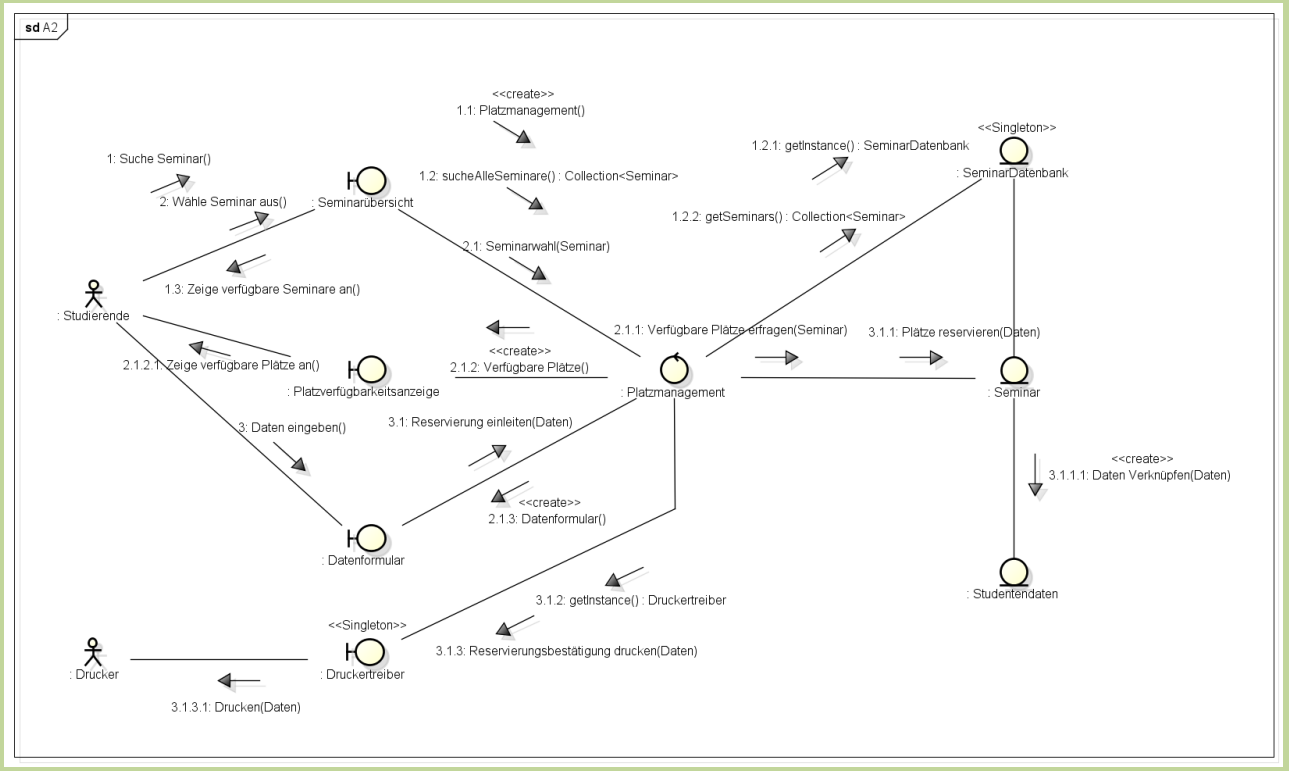


Gleiche Elemente zusammengefasst 1P + konzeptionell gleiche Controller zusammengefasst 1P + 0,5 Bezeichner = 2,5P

Hier können beide Controller zusammengefasst werden, da sie konzeptionell denselben Fall verarbeiten. Sie sind nur für unterschiedliche Aspekte bzw. Sichtweisen (einmal Studierender und einmal Mitarbeiterin) dieses Szenarios zuständig.

Aufgabe 2. *Analysephase: Kommunikationsdiagramm* **10 Punkte**

Extrahieren Sie für den Anwendungsfall **Online-Seminarauswahl** die Nachrichten des Ereignisflusses und notieren Sie sie in einem zugehörigen Kommunikationsdiagramm. Nutzen Sie als Grundlage das Ergebnis der *Aufgabe 1c*).



19x0,5 Korrekte Nachricht + 0,5 Bezeichner = 10P

Anmerkungen:

- Alle Interaktionen zwischen einem Akteur und einem Boundary finden außerhalb des Systems statt und sind insofern informelle Beschreibungen von Tätigkeiten, die der Akteur mit dem Boundary (z.B. *„Suche Seminar“*) bzw. das Boundary für den Akteur ausführt (z.B. *„Zeige verfügbare Seminare an“*). Es sind keine konkreten Nachrichten, die an ein Objekt im System geschickt werden.
- Die Aktion *„1: Suche Seminar“* ist zu interpretieren als *„Der Studierende wählt die ‚*Suche Seminar*‘- Aktion auf der Benutzeroberfläche aus“*. Wie das geschieht, wird hier nicht modelliert (Es kann z.B. durch klicken eines Links, Buttons, Menüeintrags, etc. oder durch Berühren eines bestimmten Elements auf einem Touchscreen realisiert sein). Ebenso wird der Rest der Benutzeroberfläche, die die Auswahl der *„Suche Seminar“*-Aktion zur Verfügung stellt, nicht modelliert.
- Wenn man das Boundary, in dem *„Suche Seminar“* angeklickt werden kann, explizit modellieren würde, sollte man die *„Suche Seminar“*-Interaktion daran richten und

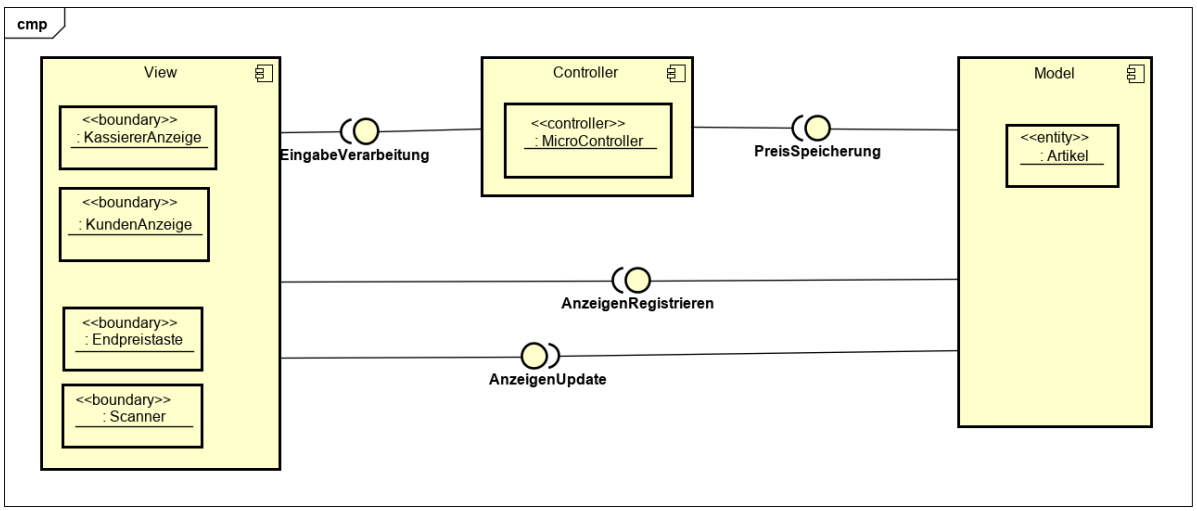
dann als erste Unteraktion eine <<create>>-Nachricht einfügen, die klar macht, dass als Reaktion das Boundary „Seminarübersicht“ erzeugt (und angezeigt) wird.

- Bezogen auf die Erzeugungsnachrichten fehlt im obigen Diagramm eine, die klar macht, dass der Controller „Platz-Management“ von dem Boundary „Seminarübersicht“ erzeugt wird (bevor es die *„verfügbare Seminare Suchen“*-Nachricht an den Controller schickt).

Aufgabe 3. *Komponentendiagramm* **7,5 Punkte**

Die Firma aus Aufgabe 3 von Blatt 7 entscheidet sich, eine Model-View-Controller Architektur für die Modellierung der Kasse zu verwenden. Die Anzeigen sollen sich während ihrer Initialisierung beim Modell anmelden, um Aktualisierungs-Benachrichtigungen zu erhalten.

Überlegen Sie sich sinnvolle Einteilungen der Analyseklasse in Komponenten (Subsysteme). Definieren Sie die Dienste dieser Subsysteme. Erstellen Sie dazu ein passendes Komponentendiagramm. Überlegen Sie sich, wie die Komponenten sonst noch miteinander interagieren könnten.



3x1 Einteilung in Komponenten + 4x0,5 „Nutzt“ + 4x0,5 „Bietet Dienst“ + 0,5 Bezeichner = 7,5P

Theoretische Aufgaben

Aufgabe 4. *Singleton und Façade Entwurfsmuster* **7 Punkte**

a) **(2 Punkte)** Façade Entwurfsmuster:

i. **(1 Punkt)** Welchen Zweck erfüllt das Façade Entwurfsmuster?

Reduzierung der Abhängigkeiten von einem Subsystem. Die Façade Klasse enthält die für externe Clients relevanten Methoden. Somit muss der Client nichts über die Interna wissen.

ii. **(0,5 Punkte)** Wofür nutzen wir ein Façade konkret bei der Implementierung einer Komponente (eines Subsystems)?

Schnittstelle nach außen.

iii. **(0,5 Punkte)** Welches Entwurfsmuster wird häufig zusammen mit dem Façade Entwurfsmuster verwendet? Begründen Sie Ihre Antwort.
Das Singleton Entwurfsmuster kann man nutzen, wenn man z.B. die Anzahl der Zugriffe auf das Subsystem kontrollieren möchte (oder andere Zustandsbeschränkungen).

b) **(5 Punkte)** Singleton Entwurfsmuster:

i. **(2 Punkte)** Welchen Zweck erfüllt das Singleton Entwurfsmuster und wie wird dies konzeptionell erreicht?

Durch das Singleton Entwurfsmuster kann man Klassen entwerfen, aus welchen man nur eine (oder begrenzte Menge an) Instanz(en) erzeugen kann.

- alle Konstruktoren privat
- ein privates Klassenfeld, das einzige Instanz speichert (Rollenname instancePool)
- Durch eine public Klassenmethode (Rollenname: getInstance()) erhält man die einzige Instanz und erzeugt diese, wenn sie noch nicht existiert.

ii. **(3 Punkte)** Geben Sie eine minimale Implementierung als Java-Code an.

```
public class Singleton{

        private static Singleton instance;

        private Singleton(){}

        public static Singleton getInstance(){
                if(instance == null) {
                        instance = new Singleton();
                }
```

```
            return instance;
        }
}
```

Aufgabe 5. *Subsystem und Architekturen* **9,5 Punkte**

Stoff wird in kommender Woche behandelt

a) **(1,5 Punkte)** Beschreiben Sie kurz die Funktionsweise von Pipe-and-Filter Systemen.
Filter sind grundsätzlich Funktionen, die einen Datenstrom in einen anderen überführen. Pipes sammeln die Daten der Filter und leiten sie an die nächsten Filter weiter (Synchronisierung parallel verarbeiteter Daten).

b) **(1,5 Punkte)** Beschreiben Sie die Funktionsweise der Repository-Architektur.

Subsysteme interagieren indirekt miteinander über das Repository, das registrierte Teilnehmer via Events über Änderungen informiert (lose Kopplung). Mit diesem schreiben und lesen sie Daten einer einzigen gemeinsamen Datenstruktur.

c) **(2 Punkte)** Geben Sie jeweils ein dazu passendes reales Beispiel für alle folgenden Architekturen an und benennen Sie die Rollen und wer sie spielt:

- 2-Schichten-Client-Server - Multiplayer Videospiel
  (Spieler = Client; Server = Server)
- 3-Schichten-Client-Server - Forum
  (Nutzer = Client; Server = Logische Schicht (PHP…); Datenbank für Einträge)
- 4-Schichten-Client-Server - Komplexes Computer Algebra System (CAS)
  (Nutzer = Client; Server zur Interaktion mit dem System; Applikationsserver; Datenbankserver)

  ➔ Konkretes Beispiel: Graph Algorithmen auf riesigen Graphen (z.B. Teilen des WWW)
- Peer-to-Peer - Beliebte Instant Messenger

d) **(1,5 Punkte)** Was sind Kohärenz und Kopplung? Wie sollte die Verteilung dieser in einem Subsystem aussehen?

Kohärenz: Abhängigkeitsgrad zwischen Klassen innerhalb eines Subsystems / einer Kapselungseinheit.

Kopplung: Abhängigkeitsgrad zwischen Subsystemen / Kapselungseinheiten der gleichen Art.

Die Kohärenz der Subsysteme sollte stark sein, ihre Kopplung gering.

e) **(0,5 Punkte)** Welche Faktoren unterscheiden Packages von Subsystemen/Komponenten?

Pakete haben keine eigene Kapselungsgrenze, sondern realisieren nur Namensräume.

f) **(1 Punkt)** Erklären Sie die Begriffe „Schicht“ und „Partition“.

Schicht: Subsystem, welches Dienste anbietet, ohne seine Nutzer (aus höheren Schickten) zu kennen.

Partition: Subsysteme, welche innerhalb der gleichen Schicht gegenseitig miteinander interagieren. Eine Schicht / Ebene *kann* in mehrere Partitionen aufgeteilt sein.

g) **(0,5 Punkte)** Was ist Middleware?

Folie 7b) – 10: Softwaresystem auf Basis standardisierter Schnittstellen und Protokolle, die Session- und Presentation-Ebene des ISO/OSI-Modells implementiert und somit den Aufbau verteilter Anwendungen erleichtert. Die Anwendungen müssen sich z.B. nicht selbst um den Verbindungsaufbau (Session-Ebene) kümmern, sondern sie interagieren (z.B. bei Java Remote Method Invocation) scheinbar direkt mit Objekten die in Wirklichkeit auf einem anderen Rechner / in einem anderen Prozess laufen.

h) **(0,5 Punkte)** Was sind Applikationsserver und wie unterscheiden sie sich von Middleware?

Folie 7b) – 11: Applikationsserver sind Softwaresysteme, die als Laufzeitumgebung für Anwendungen dient und folgende über Middleware-Funktionen hinausgehende Fähigkeiten bieten

- Transparenz der Datenquellen
- Objekt-Relationales Mapping
- Transaktionsverwaltung
- Lebenszyklusmanagement („Deployment“, Updates, Start)
- Verwaltung zur Laufzeit (Monitoring, Kalibrierung, Logging, ...)

i) **(0,5 Punkte)** Wie unterscheidet sich die Peer-to-Peer Architektur von der Client-Server Architektur?
Bei Peer-to-Peer sind beide Parteien gleichberechtigt, bei Client-Server übernimmt immer einer die Rolle des Servers.

∑ 44 Punkte