

## Zweitklausur

# Softwaretechnik II

Sommersemester 2022

Prof. Dr. Ralf H. Reussner

12.09.2022

**Name:** \_\_\_\_\_

**Matrikelnummer:** \_\_\_\_\_

Zur Klausur sind keine Hilfsmittel und kein eigenes Papier zugelassen. Die Bearbeitungszeit beträgt 90 Minuten. Die Klausur ist vollständig und geheftet abzugeben. Verwenden Sie ausschließlich dokumentenechte Stifte. Mit Bleistift oder roter Farbe geschriebene Angaben werden nicht bewertet. Lösen Sie die Aufgaben in leserlicher Schrift. Unlesbare Lösungen können naturgemäß nicht positiv bewertet werden.

<b>Aufgabe</b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>5</b>	<b>6</b>	<b><math>\Sigma</math></b>
Maximal	10	15	20	14	18	13	90
K1							
K2							
K3							

## Aufgabe 1: Wissensfragen (10 Punkte)

- a) Erklären Sie stichpunktartig das Prinzip der Offen- und Verschllossenheit (*Open Closed Principle*) aus der Clean-Code-Vorlesung. (1 Punkte)

- b) Für welchen Zweck werden Aggregate (*Aggregates*) im Domänen-getriebenen Entwurf eingesetzt. Nennen Sie einen Zweck und begründen Sie stichpunktartig wie Aggregate diesen erfüllen. (2 Punkte)

- c) Was ist ein Software Container? Schreiben Sie die Definition von Software Containern aus der Container-Vorlesung stichpunktartig auf. (1,5 Punkte)

- d) Handelt es sich bei dem im folgenden beschriebenen Echtzeitsystem um ein Monitoring- (*monitoring*) oder Kontrollsystem (*control system*)? Begründen Sie Ihre Antwort kurz. (1 Punkt)

*Das autonome Parkassistenzsystem einer Limousine überwacht, wenn aktiviert, die Fahrzeugumgebung mit Kameras kontinuierlich und steuert entsprechend den Einparkvorgang, indem es während des Einparkvorgangs Signale an die Lenkung, die Motorsteuerung und die Bremse sendet, bis das Fahrzeug seine Parkposition erreicht hat.*

- e) Nennen Sie fünf der sieben schädlichen Königreiche (Seven Pernicious Kingdoms), die potentielle Sicherheitsrisiken bergen, die in der Security-Vorlesung diskutiert wurden. (2,5 Punkte)

- f) Nennen Sie zwei der vier Werte, die im Manifest der agilen Softwareentwicklung (*agile manifesto*) proklamiert wurden. (2 Punkte)

## Aufgabe 2: Anforderungserhebung (15 Punkte)

Im Folgenden betrachten wir die Entwicklung eines Fertigungsassistenzsystems (*fabrication assistance system*), abgekürzt FAS, zur Unterstützung von manuellen Fertigungsschritten für die *Lynx Corporation*:

*Dieses Softwaresystem soll es Angestellten in der Fertigung (workers) erleichtern, kundenspezifische Produkte (products) anhand von individuellen Arbeitsplänen (work plans) zu fertigen und zu prüfen, indem der Angestellte sukzessive alle Schritte (steps) des Arbeitsplans abarbeitet. Die Arbeitspläne werden von Angestellten in der Planung (managers) erstellt und einem geeigneten Angestellten der Fertigung zugewiesen. Anhand der Arbeitspläne werden in der Lagerverwaltung (stock management) die für die Fertigung notwendigen Komponenten bestellt und bis zur Fertigung gelagert sowie die fertigen Produkte verpackt und versandt.*

Während der Anforderungserhebung wurden für das Fertigungsassistenzsystem unter anderem die folgenden Anforderungen identifiziert:

---

Nr.	Anforderung
-----	-------------

---

- |      |  |
|------|--|
| /R1/ | Das FAS muss die EG-Richtlinie 89/391 für „Mindestvorschriften für die Sicherheit und den Gesundheitsschutz von Arbeitnehmern“ für die Angestellten der Fertigung erfüllen.  |
| /R2/ | Das FAS muss es Angestellten der Fertigung erlauben, sich mit Nutzernamen und PIN anzumelden.  |
| /R3/ | Das FAS speichert für jeden Angestellten der Fertigung eine Liste von Fähigkeiten ab, wie z.B. Verschrauben, Löten oder Leitungsprüfung.   |
| /R4/ | Sobald ein Arbeitsplan einem Angestellten der Fertigung zugewiesen wird, prüft das FAS automatisch, ob diese Person die notwendigen Fähigkeiten für jeden Arbeitsschritt besitzt. Schlägt die Prüfung fehl, zeigt das FAS eine Warnmeldung an. |
| /R5/ | Das FAS muss die Sprachen Deutsch, Englisch sowie Chinesisch für Benutzeroberflächen und Arbeitspläne unterstützen.  |
| /R6/ | Das FAS soll sich intuitiv bedienen lassen.  |
| /R7/ | Die Benutzeroberfläche des FAS für den Angestellten der Fertigung muss auf einem Tablet mit einer 1.2 GHz CPU und 2 GB RAM laufen.   |
| /R8/ | Das FAS muss sich durch Plugins erweitern lassen.  |
- 

- a) Nennen Sie zwei weitere Gruppen/Rollen, die neben den Angestellten der Fertigung (*worker*) und Angestellten der Planung (*managers*) als Stakeholder für die Entwicklung des Fertigungsassistenzsystems relevant sind. (1 Punkte)

Name:

Matrikelnummer:

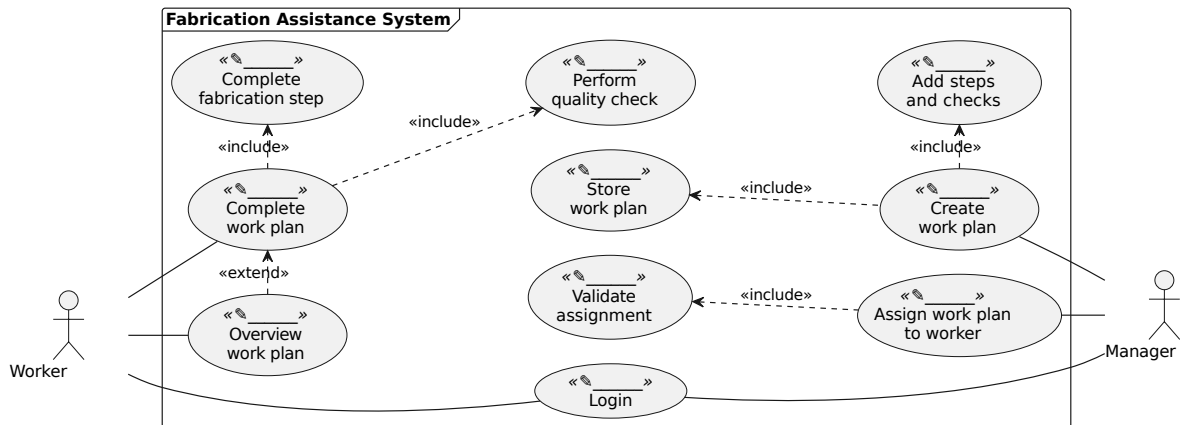
- b) Klassifizieren Sie in der nachfolgenden Tabelle die oben beschriebenen Anforderungen nach der in der Vorlesung behandelten Klassifikation von Glinz. Ordnen Sie hierzu die acht Anforderungen des Fertigungsassistenzsystem den in der Vorlesung genannten Anforderungsarten (*kind: functional, quality, constraint*), deren Unterkategorie (*subkind: functions, data, performance, security, availability, legal, ...*) sowie deren Repräsentation (*representation: operational, quantitative, qualitative, declarative*) zu. (12 Punkte)

Nr.	Anforderung	Art	Unterkategorie	Repräsentation
/R1/	EG-Richtlinie 89/391			
/R2/	Anmeldung in der Fertigung			
/R3/	Fertigungsfähigkeiten			
/R4/	Fähigkeitsprüfung			
/R5/	Sprachunterstützung			
/R6/	intuitiv Bedienung			
/R7/	Tablet mit 1,2 GHz und 2 GB			
/R8/	Erweiterbarkeit			


- c) Einige der oben stehenden Anforderungen verstoßen gegen die Empfehlungen für gut geschriebene Anforderungen (*basic writing recommendations*) aus der Vorlesung. Finden Sie **zwei** Anforderungen die gegen **unterschiedliche** Richtlinien verstoßen. Nennen Sie jeweils die Nummer der Anforderung und benennen Sie die Richtlinie, gegen die verstoßen wurde. (2 Punkte)

## Aufgabe 3: Anwendungsfallbeschreibung (20 Punkte)

Diese Aufgabe betrachtet die Anwendungsfälle der Angestellten in der Fertigung (*worker*) und Angestellten in der Planung (*manager*). Diese sind im folgenden Anwendungsfalldiagramm zusammengefasst und in der nachfolgenden Tabelle kurz erläutert.



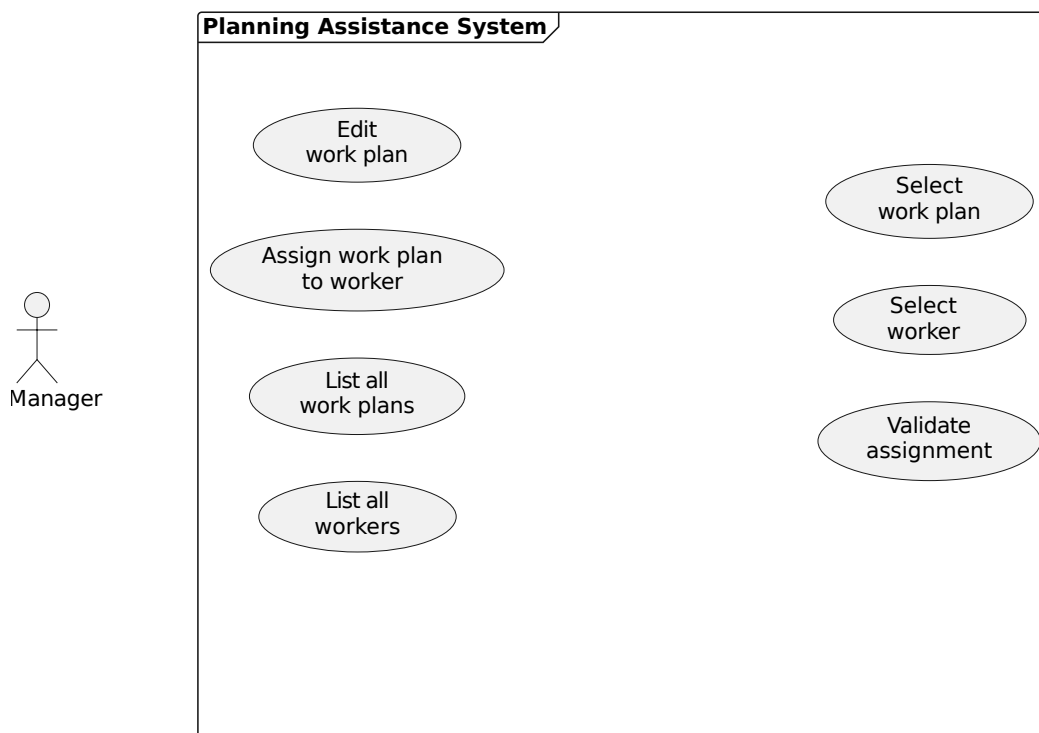
Anwendungsfall	Beschreibung
<i>Overview work plan</i>	Betrachten der Übersicht des zugewiesenen Arbeitsplans.
<i>Complete work plan</i>	Abarbeiten des zugewiesenen Arbeitsplans.
<i>Complete fabrication step</i>	Bearbeiten und abschließen eines Fertigungsschritts.
<i>Perform quality check</i>	Durchführen einer Qualitätsprüfung des zu fertigenden Produktes.
<i>Create work plan</i>	Anlegen eines Arbeitsplans zur Fertigung eines Produktes.
<i>Add steps and checks</i>	Hinzufügen von Montageschritten und Qualitätsprüfungen zum Arbeitsplan.
<i>Store work plan</i>	Abspeichern des Arbeitsplanobjekts in der Datenbank des FAS.
<i>Assign work plan to worker</i>	Zuweisen eines existierenden Arbeitsplans an einen Angestellten in der Fertigung.
<i>Validate assignment</i>	Überprüfen, dass zugewiesener Angestellter die notwendigen Fähigkeiten für jeden Arbeitsschritt des zugewiesenen Arbeitsplans besitzt.
<i>Login</i>	Anmelden des Angestellten am FAS.

- a) Klassifizieren Sie jeden Anwendungsfall im obigen Anwendungsfalldiagramm (use case diagram) entweder als Nutzerziel (*user goal*) mit U, als Teilfunktion (*subfunction*) mit S oder als Operation (*operation*) mit O. Tragen Sie hierzu die entsprechende Klassifikation in die mit „“ markierten Bereiche der Anwendungsfälle ein. (5 Punkte)

Da diese Anwendungsfälle nicht detailliert genug sind, ist es Ihre Aufgabe ein weiteres Anwendungsfalldiagramm zu erstellen. Hierzu charakterisiert die folgende Beschreibung die Aufgaben die in der Planung erfüllt werden:

*Im Folgenden wird angenommen, dass die planende Person (manager) im FAS angemeldet ist. Das FAS ermöglicht einem planenden Angestellten Arbeitspläne zu bearbeiten (edit work plan) und existierende Arbeitspläne einem fertigenden Angestellten (worker) zuzuweisen (assign work plan to worker). In beiden Fällen muss die planende Person zuerst den Arbeitsplan anhand seiner Kennung auswählen (select work plan). Falls die Kennung unbekannt ist, kann die Person sich über eine Schaltfläche die Liste aller Arbeitspläne anzeigen lassen (list all work plans). Bei der Zuweisung von Arbeitsplänen muss die planende Person zusätzlich einen geeigneten fertigenden Angestellten anhand seines Namens auswählen (select worker). Ist die Person unklar, kann sich die planende Person über einen Link die Liste aller fertigenden Angestellten und ihre Fähigkeiten anzeigen lassen (list all workers). Nach der Auswahl der fertigenden Person wird automatisch überprüft (validate assignment), ob sie die notwendigen Fähigkeiten für den zugewiesenen Arbeitsplan besitzt. Im Erfolgsfall wird die Zuweisung in der Datenbank des FAS gespeichert. Ansonsten zeigt das FAS eine Warnmeldung an und die planende Person muss einen anderen Angestellten auswählen.*

- b) Vervollständigen Sie nun das Anwendungsfalldiagramm (*use case diagram*) genau so, dass es die Anforderung im obigen Text beschreibt. Zeichnen Sie ausschließlich die Beziehungen von Aktoren (*actors*) zu Anwendungsfällen (*use cases*) und zwischen Anwendungsfällen (*use cases*) ein, die im Text beschrieben werden. Zeichnen Sie keine weiteren Anwendungsfälle (*use cases*) oder Aktoren (*actors*) ein. (7 Punkte)



Name:

Matrikelnummer:

---

- c) Verwenden Sie das textuelle Anwendungsfallbeschreibungsschema *Fully-Dressed* zur vollständigen Spezifikation des Erfolgsfalls der Zuweisung eines Arbeitsplans an einen Angestellten der Fertigung (*assign work plan to worker*). Nutzen Sie hierfür die unten gegebene Schablone. Nennen Sie zwei Stakeholder und geben Sie mindestens eine Erweiterung an. (8 Punkte)

---

**Name**

**Arbeitsplan zuweisen** (*assign work plan to worker*)

---

**Zielebene** (*Goal Level*)

**Primäraktor(en)** (*Primary Actor(s)*)

**Stakeholder(s)**

**Vorbedingungen**

(*Preconditions*)

**Nachbedingungen**

(*Postconditions*)

**Primäres Erfolgsszenario**

(*Main Success Scenario*)

**Erweiterung (Weiterer Ablauf)**

(*Extension; Additional Flow*)

---




Name:

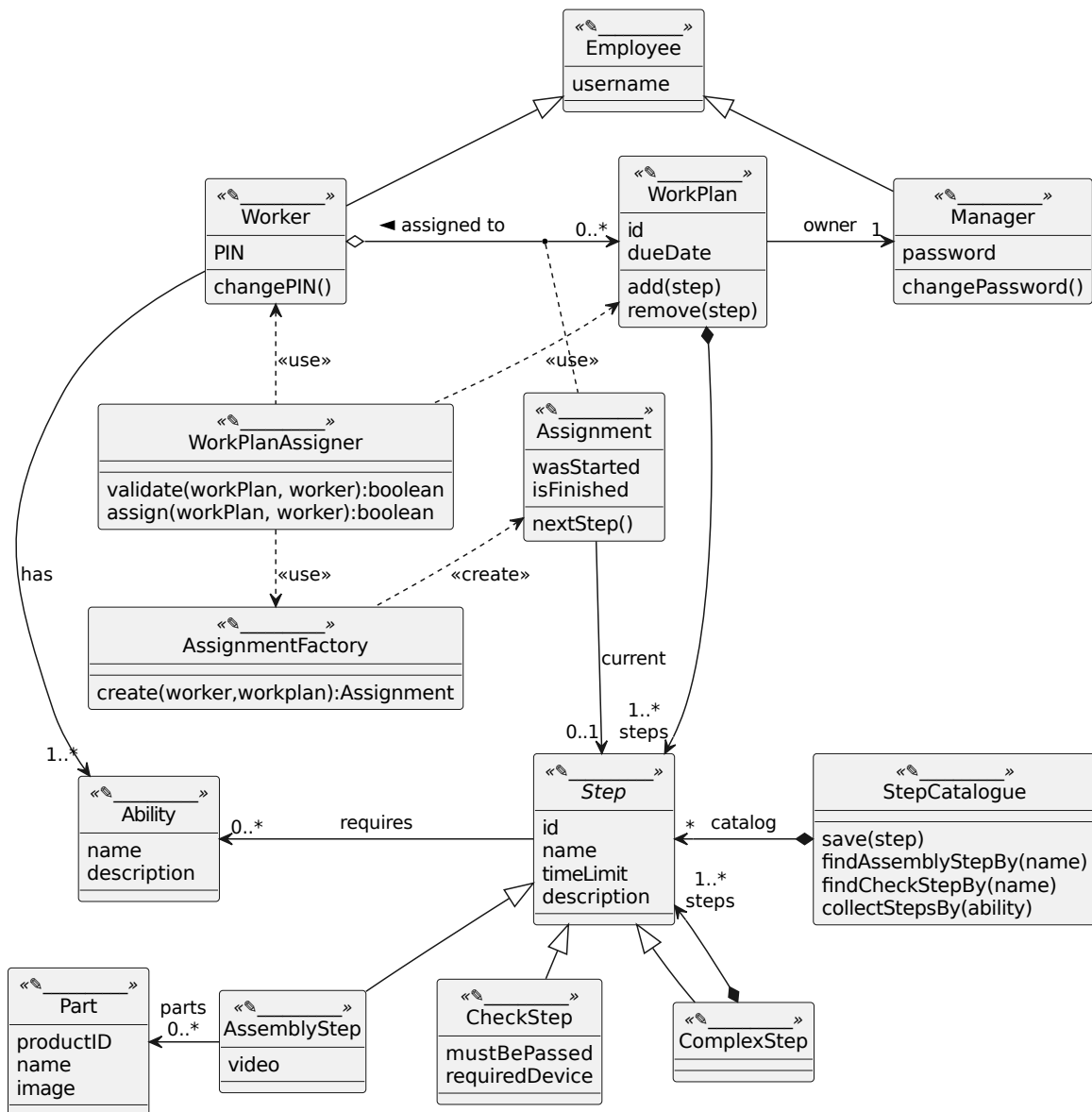
Matrikelnummer:

---

## Aufgabe 4: Domänen-getriebener Entwurf (14 Punkte)

Der folgende Aufgabenkomplex widmet sich dem Kern des Fertigungsassistenzsystems (FAS), in dem die Arbeitspläne (*work plans*) verwaltet und fertigenden Angestellten (*worker*) zugewiesen werden. Weiterhin wird hier zwischen Montageschritten (*assembly step*) und Prüfschritten (*check step*) unterschieden, die in einem eigenen Katalog (*step catalogue*) zusammen verwaltet werden.

- a) Durch die Domänenanalyse des FAS ist das folgende Klassendiagramm entstanden. Klassifizieren Sie jedes Domänenkonzept entsprechend der Bausteine (*building blocks*) des Domänen-getriebenen Entwurfs entweder als ValueObject mit V, Entity mit E, Service mit S, Factory mit F oder Repository mit R. Tragen Sie hierzu die Klassifikation in die mit „“ markierten Bereiche der Domänenklassen ein. (7 Punkte)



- b) Begründen Sie Ihre Entscheidung für die Klassifikation der Domänenklasse `WorkPlanAssigner` in Stichpunkten. (1 Punkt)

- c) Entwerfen Sie nun die Unterstützungsmuster *Repository* und *Factory* für das Domänenkonzept `WorkPlan`. Zeichnen Sie hierfür ein UML-Klassendiagramm, welches ein *Repository*, eine *Factory* sowie ihre Methoden mit Parametern und Rückgabetyt beinhaltet. Spezifizieren Sie für die *Factory* mindestens eine Methode und für das *Repository* mindestens drei Methoden. Zeichnen Sie zusätzlich die Domänenklasse `WorkPlan` sowie ihre Beziehungen zum *Repository* beziehungsweise zur *Factory* ein. (6 Punkte)

**Hinweis:** Datentypen (wie zum Beispiel `int`, `long`, `Date`, `String`, ...) müssen nicht angegeben werden. Beschriften Sie jede Assoziation mit mindestens einem Rollennamen oder Assoziationsnamen sowie mindestens einer Kardinalität.

## Aufgabe 5: Saubere Unternehmensarchitekturen (18 Punkte)

- a) Tragen Sie in die Tabelle die vier Schichten der sauberen Architektur (*clean architecture*) so ein, dass sie zur angegebenen Richtung der Abhängigkeiten passen (*dependency rule*). (2 Punkte)

Schicht	Abhängigkeitsregel	Schichtenname
1.	↓	
2.	↓	
3.	↓	
4.	↓	

- 
- The diagram illustrates the component architecture of a manufacturing system. It features several key components and their interdependencies:
- «subsystem» Supplier's System**: A subsystem component that depends on the **Part Order Gateway**.
  - Part Order Gateway**: A component that provides the **Part Order Gateway** interface to the **Production Planning** component.
  - Manager Client**: A component that depends on the **Manager Backend** via the **RPC** interface.
  - Manager Backend**: A component that provides the **RPC** interface to the **Manager Client** and the **Worker Backend**.
  - Worker Terminal**: A component that depends on the **Worker Backend** via the **REST** interface.
  - Worker Backend**: A component that provides the **REST** interface to the **Worker Terminal** and the **External Interface Adapter**.
  - External Interface Adapter**: A component that depends on the **Worker Backend** via the **USB** interface.
  - Cable Test Device**: A component that depends on the **External Interface Adapter** via the **USB** interface.
  - Person Register**: A component that provides the **IPersons** interface to the **Assignment Management** component.
  - Assignment Management**: A component that provides the **IPersons** interface to the **Production Planning** component.
  - Work Plan Repository**: A component that provides the **IWorkPlans** interface to the **Production Planning** component.
  - Step Catalog**: A component that provides the **IStepCatalog** interface to the **Production Execution** component.
  - Part Store**: A component that provides the **IParts** interface to the **Production Execution** component.
  - User Account Management**: A component that provides the **IAuthentication** interface to the **Production Planning** and **Production Execution** components.
  - Production Planning**: A component that provides the **IPersons**, **IAssignments**, **IModify**, **IQuery**, and **IParts** interfaces to the **Production Execution** component.
  - Production Execution**: A component that provides the **IPersons**, **IAssignments**, **IModify**, **IQuery**, and **IParts** interfaces to the **Production Planning** component.
  - JPA**: A component that provides the **JPA** interface to the **Assignment Management**, **Work Plan Repository**, and **Step Catalog** components.
  - Hibernate**: A component that provides the **JPA** interface to the **Relational Database**.
  - Relational Database**: A component that provides the **JPA** interface to the **Hibernate** component.

Der Quelltext in Listing 1 stellt einen Ausschnitt der Implementierung des Arbeitsplans (workPlan) im Fertigungsassistenzsystem dar. Genauer werden hier sowohl angemeldete Fertigungsschritte (AssemblyStep) wie auch Prüfschritte (CheckStep) modelliert. Die hierfür entwickelte Anwendung enthält jedoch einige Verstöße gegen Praktiken des sauberen Programmierens (Clean-Code-Prinzipien und Coding Conventions aus der Vorlesung).

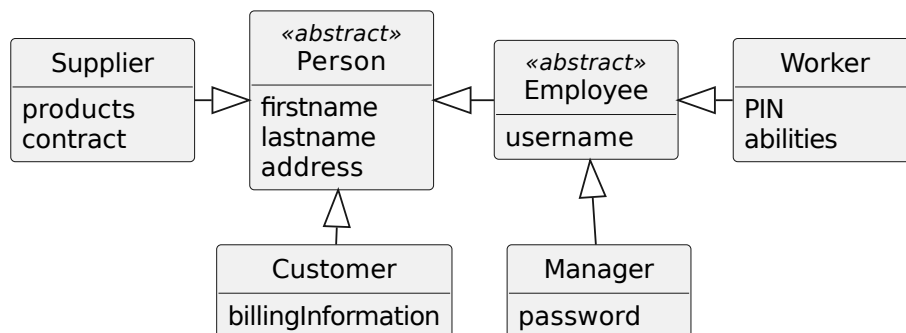
- c) Identifizieren Sie **sechs** Zeilen im Quelltextbeispiel (nächste Seite), die **unterschiedliche** Arten von Verstößen gegen die Praktiken des sauberen Programmierens (Clean-Code-Prinzipien und Coding Conventions aus der Vorlesung) enthalten. Benennen Sie jeweils die **Art** des Verstoßes mit der zugehörigen **Zeilennummer**. (6 Punkte)

```
1 public class Part{
2     public final long productId;
3     public final String name;
4     public final int quantity;
5     /* implements equals and hashCode */
6 }
7 public abstract class Step{
8     private final String name;
9     public Step(String name) {
10         //if (name==null) throw new IllegalArgumentException();
11         this.name = name;
12     }
13     public abstract List<Part> getParts();
14     public abstract boolean performCheck(boolean success);
15     /*...*/
16 }
17 public class AssemblyStep extends Step{
18     private final List<Part> parts=new ArrayList<>();
19     public AssemblyStep(String name, Part... parts) {
20         super(name);
21         //Add all parts to the list of parts
22         this.parts.addAll(Arrays.asList(parts));
23     }
24     public List<Part> getParts() { return parts; }
25     public boolean performCheck(boolean success) { return false; }
26 }
27 public class CheckStep extends Step{
28     public final boolean mustBePassed;
29     public CheckStep(String name, boolean mustBePassed) {
30         super(name);
31         this.mustBePassed = mustBePassed;
32     }
33     public List<Part> getParts() {
34         return Collections.emptyList();
35     }
36     public boolean performCheck(boolean success) {
37         return !mustBePassed || success;
38     }
39 }
40 public class WorkPlan {
41     private final ArrayList<Step> steps=new ArrayList<>();
42     /*...*/
43     public Map<Part,Integer> getOrder() {
44         Map<Part,Integer> result=new HashMap<>();
45         for (var step : steps) if (step instanceof AssemblyStep) for (var part : step.getParts())
46             result.compute(part, (p,v) -> (v==null ? p.quantity : v+p.quantity) );
47         return result;
48     }
49 }
```

Listing 1: WorkPlan-Klasse mit Bad Smells

## Aufgabe 6: Entwurfsmuster für Unternehmenssoftware (13 Punkte)

Das Domänenmodell des Fertigungsassistenzsystems enthält unter anderem die folgende Klassenhierarchie für die unterschiedlichen Personen, die im Personenregister (*person register*) verwaltet werden, genauer Kunden (*customers*), Zulieferern (*supplier*), Angestellte der Fertigung (*worker*) sowie Angestellte der Planung (*manager*). Um diese Klassenhierarchie in einer Datenbank abzubilden, sollen Sie in dieser Aufgabe die unterschiedlichen Objekt-Relationalen Strukturmuster (*object-relational structural patterns*), welche in der Vorlesung behandelt wurden, anwenden.



- a) Erstellen Sie ein Datenbankschema nach dem Strukturmuster *Concrete Table Inheritance*. Erstellen Sie hierzu ein UML-Klassendiagramm der nötigen Tabellen (dargestellt als UML-Klasse mit dem <<table>> Stereotyp) und ihrer Attribute. Führen Sie, falls nötig, neue Attribute ein, um *Primärschlüssel* und *Vererbung* darzustellen. (7 Punkte)

**Hinweis:** Datentypen (wie zum Beispiel int, long, Date, String, ...) müssen nicht angegeben werden.



Name:

Matrikelnummer:

---

- b) Erstellen Sie ein Datenbankschema nach dem Strukturmuster *Single Table Inheritance*. Erstellen Sie hierzu ein UML-Klassendiagramm der nötigen Tabellen (dargestellt als UML-Klasse mit dem <<table>> Stereotyp) und ihrer Attribute. Führen Sie, falls nötig, neue Attribute ein, um die *Primärschlüssel* und *Vererbung* darzustellen. (4 Punkte)

**Hinweis:** Datentypen (wie zum Beispiel int, long, Date, String, ...) müssen nicht angegeben werden.



- c) Geben Sie stichpunktartig einen Vorteil und einen Nachteil der *Concrete Table Inheritance* gegenüber der *Single Table Inheritance* an. (2 Punkt)



Name:

Matrikelnummer:

---

Name:

Matrikelnummer:

---

Name:

Matrikelnummer:

---