

PRÜFUNG SS 2019

Technologien und Methoden der Softwaresysteme I Softwaretechnik I

Name:	Vorname:	Matrikel-Nr:
-------	----------	--------------

- Prüfungstag:** 23.09.2019
- Prüfungsdauer:** 120 Minuten
- Maximalpunktzahl:** 120 Punkte
- Prüfungsumfang:** 5 Aufgaben (24 Seiten)
- Hilfsmittel:** Alle, außer Kommunikationsgeräte
- Hinweise:**
- 1.) Verlangt sind alle Aufgaben.
 - 2.) Verwenden Sie **keinen Rotstift / Grünstift / Bleistift**.
 - 3.) Tragen Sie auf dem **Deckblatt** Ihren **vollen Namen** und Ihre **Matrikelnummer** ein.
 - 4.) Tragen Sie auf **allen Blättern** Ihre **Initialen** (1. Buchstabe von Vor- und Nachnamen) ein.
 - 5.) Verwenden Sie für Ihre Lösung unbedingt die **jeweiligen Lösungsfelder**.
 - 6.) Geben Sie dieses **Deckblatt zusammen** mit Ihren in der **richtigen Reihenfolge sortierten Aufgabenblättern** ab.

----- Nicht von Studierenden auszufüllen -----

Aufgabe	1	2	3	4	5			
Maximal	20	25	20	25	30			
Erreicht								

Aufgabe 1: Analyse und Entwurf (20 Punkte)

Sie arbeiten als Softwareentwickler am IAS, dieses will einen automatisierten Smoothie-Mixer „SMOOTHIAS“ entwickeln. SMOOTHIAS soll aus verschiedenen Fruchtsäften Erfrischungsgetränke zubereiten können. Als einer der Entwickler sollen Sie zunächst die Analyse der neuen Steuerungssoftware durchführen. Da am IAS innovativ entwickelt wird, soll dies objektorientiert erfolgen.

SMOOTHIAS besteht aus sechs Tanks, die gekühlt werden können (fünf der Tanks sind mit verschiedenen Fruchtsäften gefüllt). Hierfür wurden in jeden Tank ein Kühlaggregat und ein Temperaturfühler eingebaut.

Je eine Pumpe kann den Inhalt aus den Tanks in den Reaktor pumpen. Die gepumpte Menge wird jeweils mit einem Durchfluss-Sensor erfasst, der die Durchflussmenge der Flüssigkeit misst. Die Durchfluss-Sensoren werden vor dem Start des Pumpvorgangs von der jeweiligen Pumpe zurückgesetzt. Daraufhin startet der Pumpvorgang, bis das geforderte Volumen vom Durchfluss-Sensor detektiert wurde und der Pumpvorgang wird gestoppt.

Im Reaktor befindet sich ein Rührer, um den Inhalt durch Rühren zu vermischen. Das fertige Produkt wird mit einem Ventil, am Boden des Reaktors, in einen Becher abgelassen. Der Reaktor verfügt außerdem über ein Heizelement, um die Flüssigkeit im Reaktor erhitzen zu können, sofern das gewählte Rezept dies vorsieht. SMOOTHIAS wird über einen Touchscreen bedient. Am Touchscreen wird das gewünschte Rezept ausgewählt.

Weiterhin steht ein Reinigungsprogramm zur Verfügung, das vom Putzpersonal gestartet werden kann. Im Reinigungsprogramm wird das Reinigungsmittel aus Tank 6 in den Reaktor gepumpt, anschließend erhitzt, dann gerührt und nach der Reinigung abgelassen.

Der Servicetechniker soll über den Touchscreen einzelne Tanks auswählen und anschließend eine individuelle Kühltemperatur einstellen können.

Hinweise:

- **Vereinfachung:** Die Tanks sind genügend groß dimensioniert und beinhalten daher immer genügend Flüssigkeit, um Erfrischungsgetränke herstellen bzw. das Reinigungsprogramm durchführen zu können.
- Der Motor für den Rührvorgang besitzt ein definiertes Programm, das unter Angabe der Rührdauern (in Sekunden) aufgerufen werden kann.
- Das Ventil zum Ablassen der Flüssigkeit aus dem Reaktor in den Becher besitzt genau zwei Zustände (offen und geschlossen).
- Das Heizelement erhitzt die Flüssigkeit im Reaktor auf die Soll-Temperatur und setzt den im Reaktor angebrachten Temperaturfühler ein, um die aktuelle Temperatur der Flüssigkeit im Reaktor zu ermitteln.
- Die Kühlaggregate kühlen die Flüssigkeit im jeweiligen Tank auf die Soll-Temperatur und setzen jeweils den entsprechenden Temperaturfühler ein, um die aktuelle Temperatur der Flüssigkeit im Tank zu ermitteln.
- Beim Ablassen der Flüssigkeit aus dem Reaktor soll angenommen werden, dass nach 5 Sekunden der Ablassvorgang abgeschlossen ist. Das Ventil soll daher nach 5 Sekunden geschlossen werden.

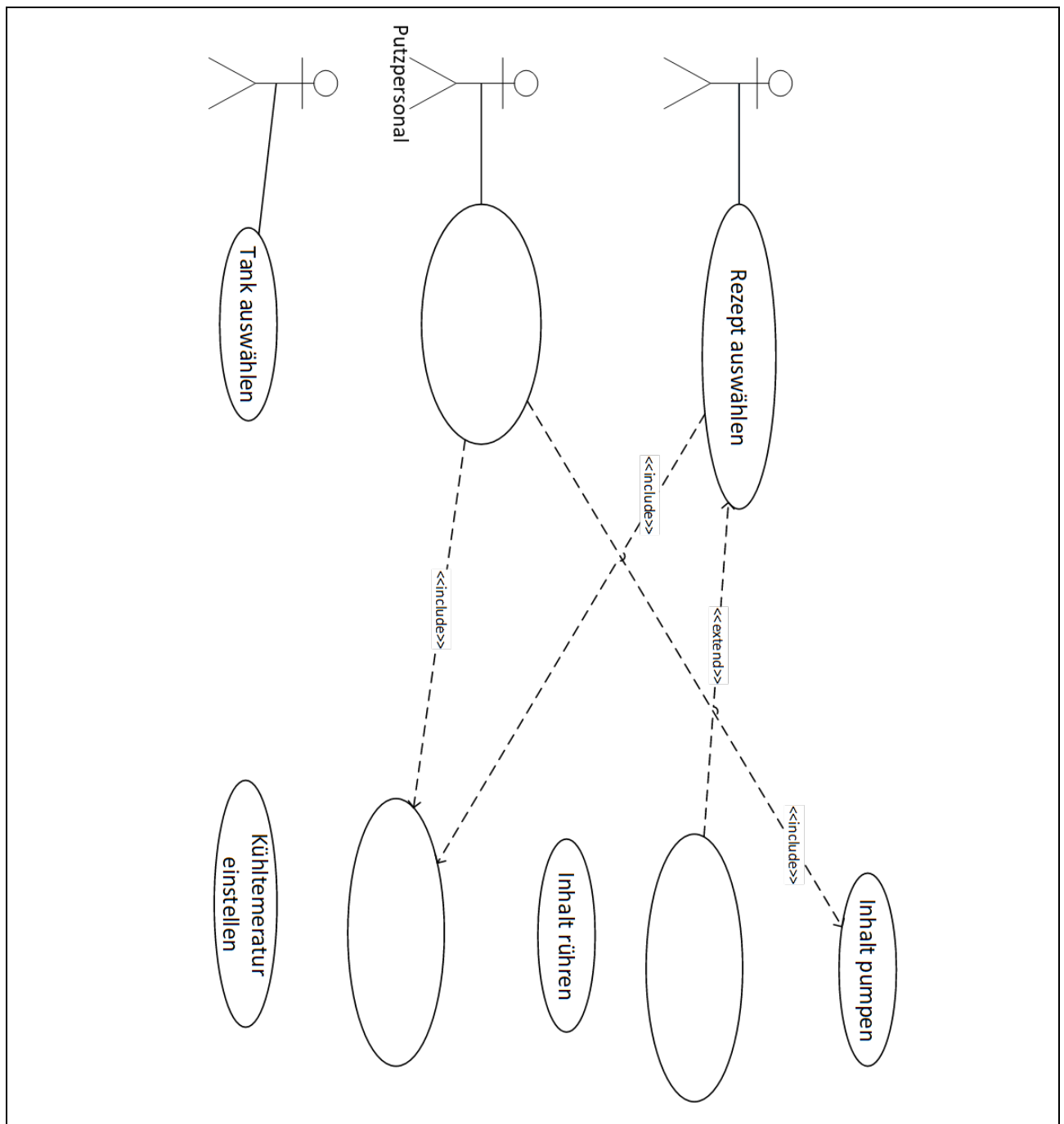
Aufgabe 1.1: Use Cases (6 Punkte)

Die Softwaresteuerung von SMOOTHIAS soll ausgehend von obiger verbaler Beschreibung mit Hilfe von Use Cases modelliert werden. Gehen Sie dabei wie folgt vor:

Vervollständigen Sie das angegebene Use Case Diagramm unter Beachtung der Hinweise um **alle fehlenden Informationen**. Verwenden Sie die, aus der Lehrveranstaltung bekannte, UML-Notation.

Hinweise:

- Ergänzen Sie **keine neuen Anwendungsfälle**.
- Ergänzen Sie **keine neuen Verbindungen** zwischen Akteuren und **Anwendungsfällen**.
- Ergänzen Sie **weitere Beziehungen** zwischen den **Anwendungsfällen**, sofern notwendig.

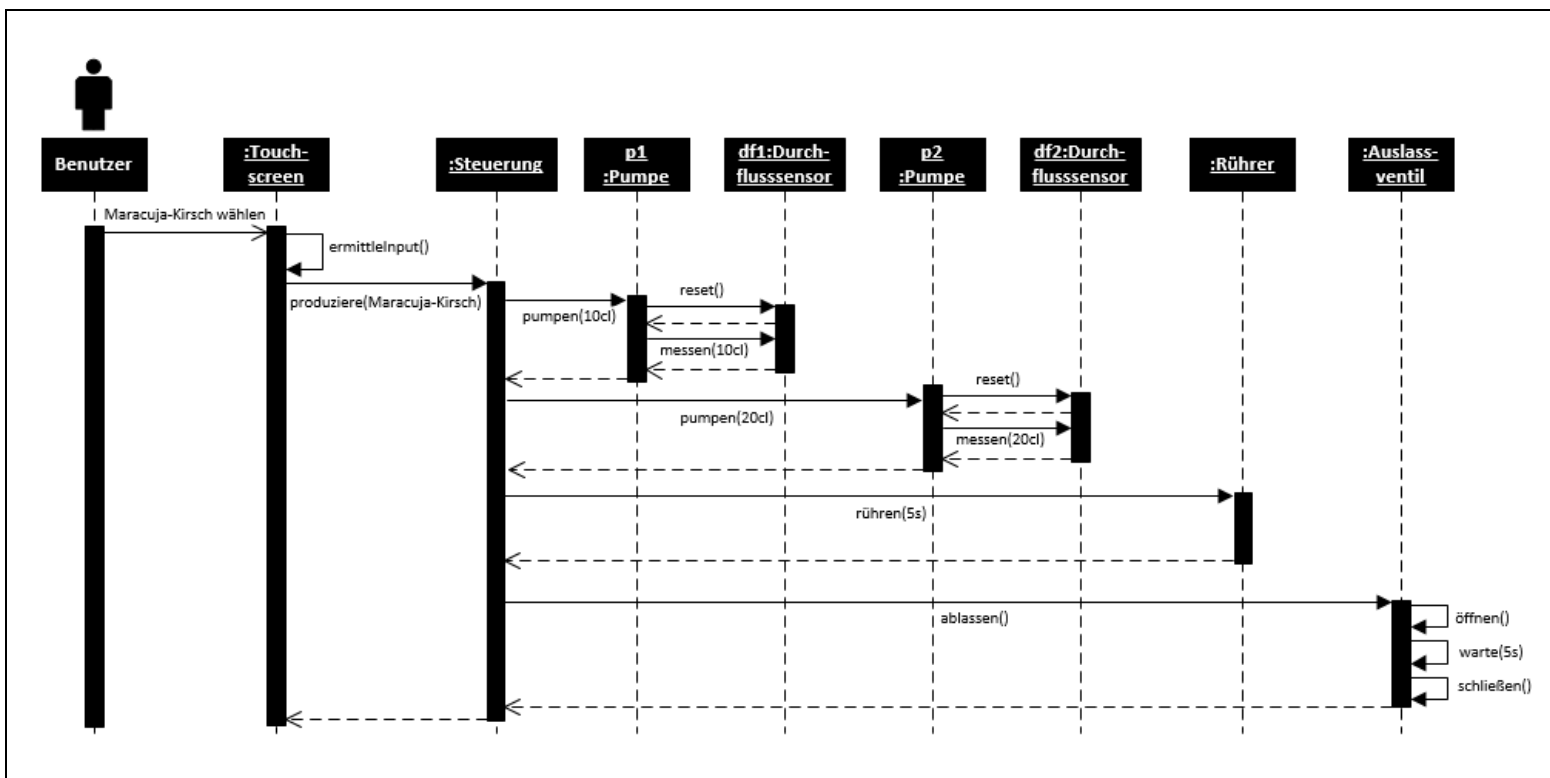


Aufgabe 1.2: Klassendiagramm (14 Punkte)

Für die Steuerungssoftware wurde bereits ein Sequenzdiagramm für das Rezept Maracuja-Kirsch-Smoothie erstellt. Dieses beschreibt den nachfolgenden Ablauf:

- Maracuja-Kirsch wird am Touch-Screen gewählt.
- 10cl Kirschsafft werden in den Reaktor gepumpt.
- 20cl Maracujasaft werden in den Reaktor gepumpt.
- Der Rührer rührt 5 Sekunden lang.
- Das Produkt wird in den Becher abgelassen.

Das zugehörige Sequenzdiagramm ist nachfolgend abgebildet:



Zeichnen Sie **ausgehend** von dem gegebenen Sequenzdiagramm **und** obiger Beschreibung (siehe Seite 2) ein Klassendiagramm der Softwaresteuerung von SMOOTHIAS nach UML-Notation. Identifizieren Sie alle relevanten Klassen und ihre Attribute und Methoden, sowie die Beziehungen untereinander mit den entsprechenden Multiplizitäten. Geben Sie die Sichtbarkeiten an.

Hinweise:

- Die Tanks und der Reaktor sollen nun über einen Status verfügen, der abgefragt werden kann, um zu überprüfen, ob die jeweilige Komponente in Ordnung ist. Dieser wird nach jedem Einschalten einmalig abgefragt.
- Die Datentypen müssen nicht berücksichtigt werden.

☐

Aufgabe 2: Implementierung und Test (25 Punkte)

Aufgabe 2.1: C0, C1-Abdeckung (11 Punkte)

Gegeben ist die folgende Funktion in der Programmiersprache C.

```
int calculate(int type, int param1, int param2) {  
    int result = 0;  
    if(type == 1) {  
        result = param1 + param2;  
    } else if(type == 2) {  
        if(param1 >= param2) {  
            result = param1 - param2;  
        } else {  
            result = param2 - param1;  
        }  
    } else if(type == 3) {  
        if(param1 < 0) param1 = param1 * (-1);  
        result = param1 * param2;  
    }  
    return result;  
}
```

- a) Geben Sie Testfälle an, um eine vollständige C0-Abdeckung zu erreichen. Verwenden Sie so wenig Testfälle wie möglich (5 Punkte).

☐

- b) Sind zusätzliche Testfälle notwendig, um eine vollständige C1-Abdeckung zu erreichen? Wenn ja, welche? (3 Punkte)

☐

- c) Beim Pfadüberdeckungstest (auch C2-Test bzw. englisch path coverage) werden im Kontrollflussgraphen

die möglichen Pfade vom Startknoten bis zum Endknoten betrachtet. In welchen Fällen ist bei Verwendung von Schleifen im Programmcode eine vollständige C2-Abdeckung möglich und in welchen nicht? Erklären Sie und geben jeweils ein Beispiel. Hinweis: diese Aufgabe kann unabhängig von den Teilaufgaben a) und b) gelöst werden. (3 Punkte)

Aufgabe 2.2: Äquivalenzklassen und Grenzwertanalyse (8 Punkte)

Gegeben ist eine Funktion, die überprüft, ob ein Datum gültig ist. Die Funktion besitzt drei Parameter (day, month, year) des Datentyps Integer. Gehen Sie zur Vereinfachung davon aus, dass alle Monate 31 Tage haben. Nur Jahreszahlen zwischen 2010 und 2019 sollen gültig sein.

- a) Stellen Sie Äquivalenzklassen zum Test dieser Funktion auf. (4 Punkte)

- b) Eine Funktion checkAge() mit den selben drei Parametern (day, month, year) überprüft, ob eine Person volljährig (d.h. mindestens 18 Jahre alt) ist. Mit welchen Grenzwerten sollte diese Funktion heute getestet werden, um die Korrektheit des Ergebnisses zu überprüfen? (2 Punkte)

- c) Gehören Äquivalenzklassenbildung und Grenzwertanalyse zu den Black-Box oder den White-Box-Verfahren? Begründen Sie. (2 Punkte)

☐

Aufgabe 2.3: Testtreiber (6 Punkte)

- a) Nennen Sie die Aufgaben eines Testtreibers. (2 Punkte)

- b) Schreiben Sie einen Testtreiber in C für die in Aufgabe 2.1 gezeigte Funktion, mit dem Sie eine vollständige C0-Abdeckung erreichen. (4 Punkte)

Aufgabe 3: Softwarearchitektur (20 Punkte)

Aufgabe 3.1: Bausteine und Schnittstellen (12 Punkte)

Gegeben ist eine generische Architektur bestehend aus System, Logic und 5 weiteren Bausteinen (A, B, C, D, E). Sowohl die Visualisierung der Architektur als auch die Beschreibung der Schnittstellen ist dabei nicht vollständig. Ergänzen Sie die Visualisierung der Architektur im nachfolgenden Lösungsfeld (4 Punkte), sowie die Beschreibung der Schnittstellen in den Tabellen 2-5 (8 Punkte). Die Architektur soll genau 9 Schnittstellen besitzen (interfaceX mit $1 \leq X \leq 9$), davon sind 2 Schnittstellen zu externen Systemen. Übergabeparameter und Rückgabetypp der Schnittstellen müssen nicht betrachtet werden.

Visualisierung der Architektur (muss ergänzt werden):

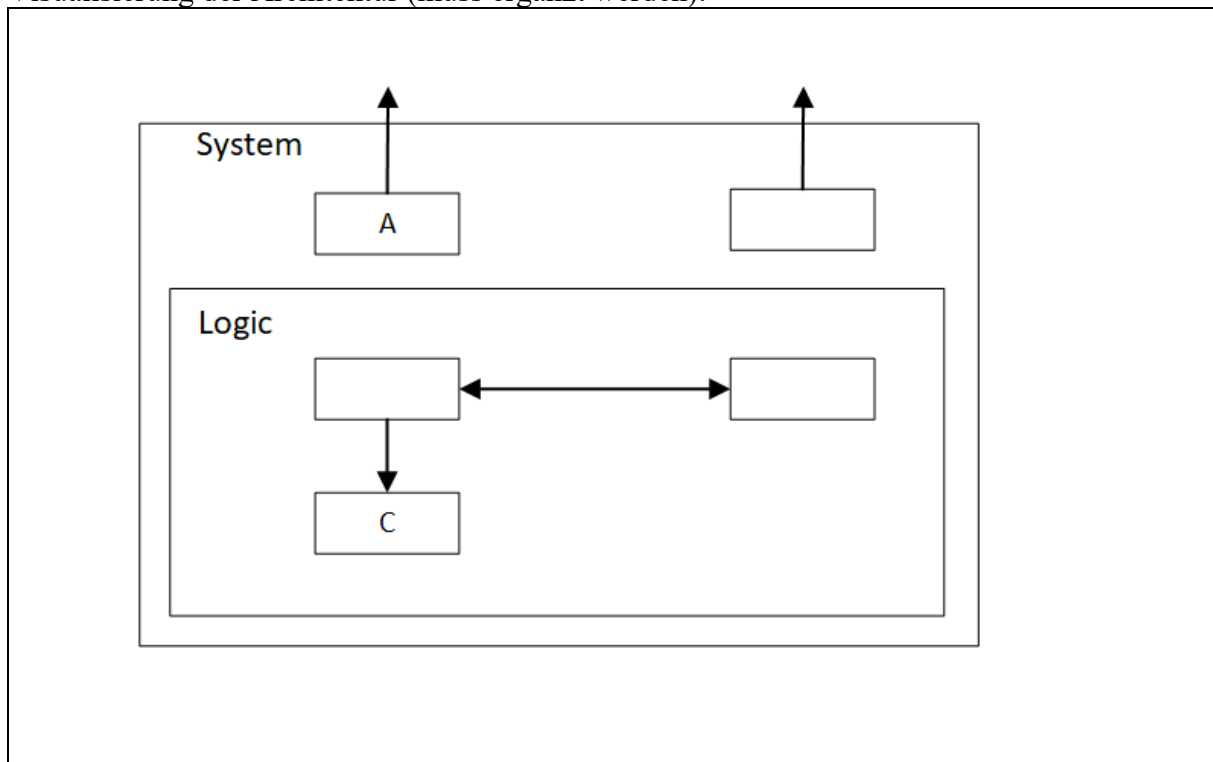


Tabelle 1: Schnittstellen von Baustein A (vollständig)

Funktion	Typ	Schnittstelle zu Baustein(en)
interface1	Imp.	extern
interface2	Imp.	E

Tabelle 2: Schnittstellen von Baustein B (muss ergänzt werden)

Funktion	Typ	Schnittstelle zu Baustein(en)
interface3	Imp.	extern
interface4	Imp.	D
interface5		D

Tabelle 3: Schnittstellen von Baustein C (muss ergänzt werden)

Funktion	Typ	Schnittstelle zu Baustein(en)
interface6	Exp.	
interface7	Imp.	

Tabelle 4: Schnittstellen von Baustein D (muss ergänzt werden)

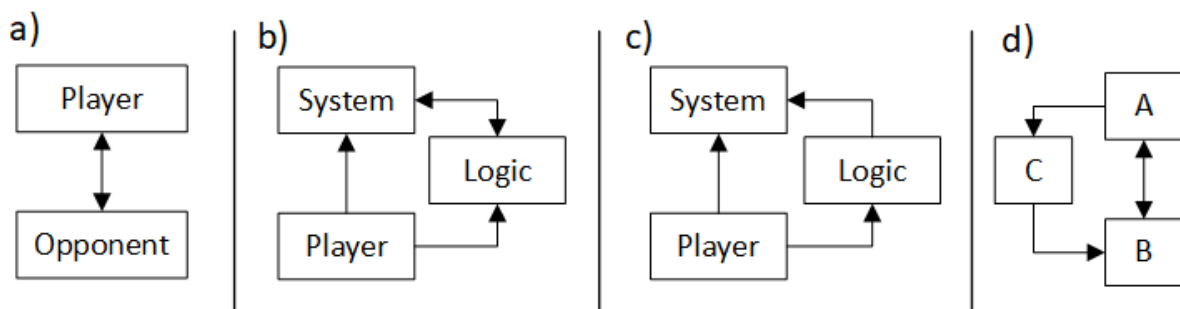
Funktion	Typ	Schnittstelle zu Baustein(en)
	Exp.	B
interface5	Imp.	
		C
interface8	Exp.	
interface9	Imp.	

Tabelle 5: Schnittstellen von Baustein E (muss ergänzt werden)

Funktion	Typ	Schnittstelle zu Baustein(en)
		A
interface6		C
interface8		
interface9		

Aufgabe 3.2: Identifikation von zyklischen Abhängigkeiten (4 Punkte)

Geben Sie jeweils die Anzahl der zyklischen Abhängigkeiten bei den folgenden vier Beispielen an. Nennen Sie die involvierten Komponenten für jede zyklische Abhängigkeit.



Anzahl der zykl. Abhängigkeiten und jeweils beteiligte Komponenten:

Beispiel a):

Beispiel b):

Beispiel c):

Beispiel d):

Aufgabe 3.3: SOLID-Prinzipien (4 Punkte)

„Das Befolgen der SOLID-Prinzipien führt zu komplexem Code / zu komplexen Klassenstrukturen“. Nehmen Sie in maximal fünf Sätzen zu dieser Aussage Stellung und gehen Sie dabei auch auf die Prinzipien ein, die zur angesprochenen Komplexität führen können.

☐

Aufgabe 4: Projektmanagement (25 Punkte)

Der Projektmanager verlässt kurz nach Projektbeginn das Unternehmen. Sie übernehmen die Aufgaben des Projektmanagements. Aus den Unterlagen Ihres Vorgängers entnehmen Sie die Aufteilung der Arbeitspakete (Tabelle 6). Für ein erfolgreiches Projektmanagement benötigen Sie eine Kosten- und Zeitplanung.

Tabelle 6: Aufteilung der Arbeitspakete

AP	Beschreibung	Dauer (M)	Vorgänger
A	Analyse der Anforderungen an die Brandmeldeanlage	3	-
B	Analyse der Anforderungen an die Komponenten	3	-
C	Entwurf der Komponenten	6	B
D	Entwurf der Gesamtarchitektur der Anlage	4	A, C
E	Implementierung der Sensoren-Software	3	C
F	Implementierung der BMZ-Software	2	D
G	Test der Sensoren	3	E
H	Test der BMZ	2	F, G
I	Systemtest	2	G, H
J	Abnahme durch Brandschutz-Beauftragten	1	I
K	Abnahme durch Behörde	1	J

Von der Geschäftsleitung erfahren Sie, dass ihr Vorgänger Kosten von 900.000€ für das Projekt abgeschätzt hat.

Tabelle 7: Aufwandsverteilungsmatrix

Phase	Analyse	Entwurf	Implementierung	Integration und Test
Tätigkeit (APs)	10%	20%	50%	
Anforderungen analysieren (A, B)	55%	20%		10%
Konzepte und Architektur entwerfen (C, D)	20%		10%	10%
Programmieren (E, F)		10%	45%	35%
Testplanung und -durchführung (G bis K)	10%	20%	30%	
Projektleitung	5%	5%	5%	5%

Die Kosten für einen Personenmonat verteilen sich wie folgt:

Tabelle 8: Kosten für Personenmonate

Mitarbeiter	Kosten	Tätigkeiten
Systemanalytiker	5.000 €	„Anforderungen analysieren“
Programmierer	3.000 €	„Konzept und Architektur entwerfen“, „Programmieren“
Tester	4.500 €	„Testplanung und -durchführung“
Manager	7.000 €	„Projektleitung“

Aufgabe 4.1 COCOMO (10 Punkte)

- a) Berechnen Sie den Gesamtaufwand für die Entwicklung in Personenmonaten nach dem COCOMO I-Verfahren. Sie schätzen, dass 61.000 Zeilen Code benötigt werden. Das System hat wenige Schnittstellen zu vorhandenen Systemen. (2 Punkte)

- b) Berechnen Sie mit den Angaben von Tabelle 7, wie sich der Aufwand auf die einzelnen Phasen und Tätigkeiten verteilt, verwenden Sie dabei ihr ganzzahlig gerundetes Ergebnis aus 4.1 a.). Füllen Sie hierzu die Tabelle im Lösungsfeld aus. (5 Punkte)

Tätigkeit (APs)	Phase	Analyse	Entwurf	Implementierung	Integration und Test
Anforderungen analysieren (A, B)					
Konzepte und Architektur entwerfen (C, D)					
Programmieren (E, F)					
Testplanung und -durchführung (G bis K)					
Projektleitung					

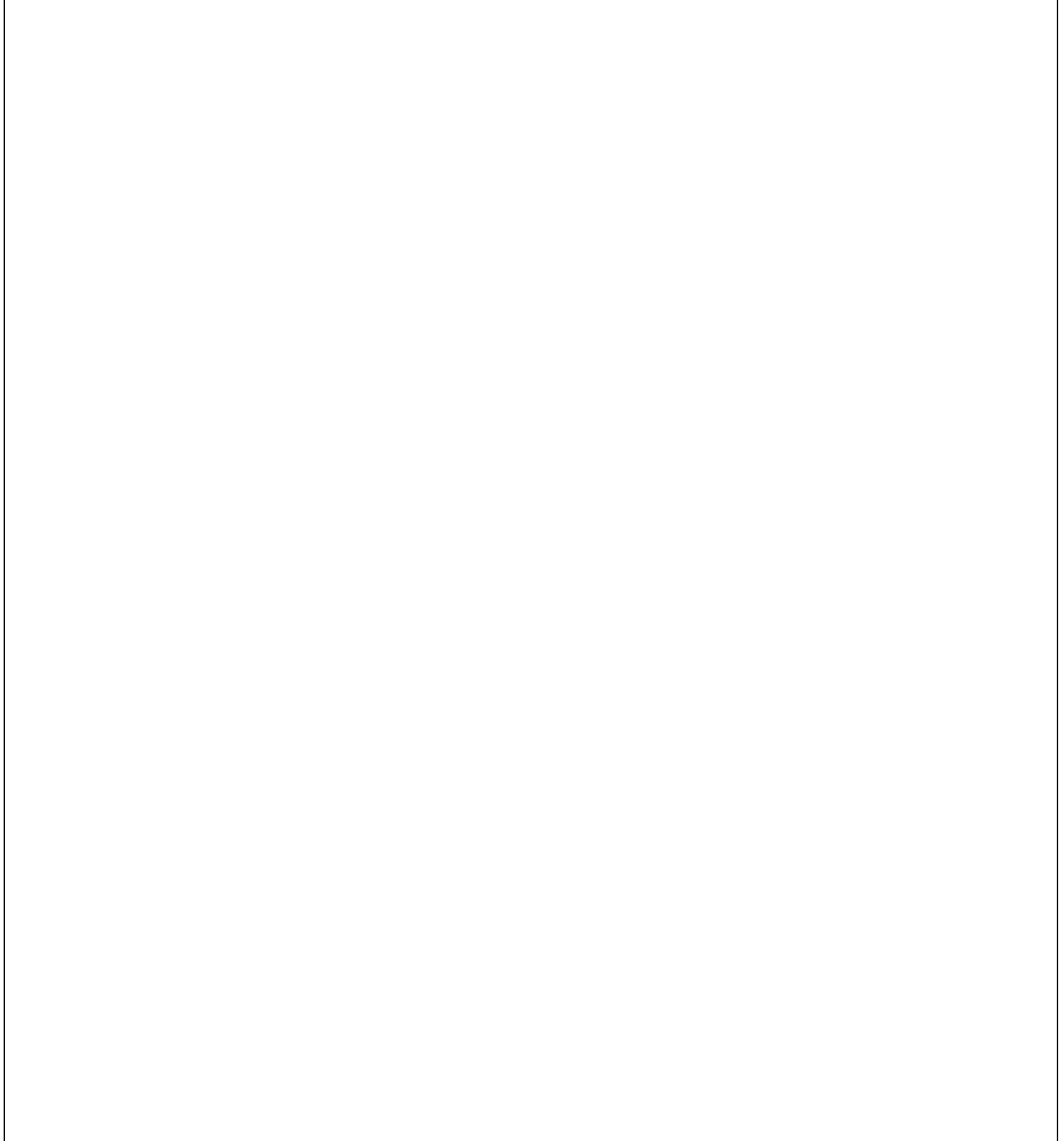
- c) Berechnen Sie die Gesamtkosten. Reichen die veranschlagten Kosten von 900.000 € aus? (3 Punkte)



Aufgabe 4.2 Netzplan (7 Punkte)

Zeichnen Sie einen Netzplan nach der Metra-Potenzial-Methode (MPM) für die in Tabelle 6 angegebenen Arbeitspakete, Dauern und Vorgängerbeziehungen und geben Sie den kritischen Pfad bzw. die kritischen Pfade an.

Berechnen Sie für jedes Arbeitspaket den frühesten und spätesten Anfangs- und End-Zeitpunkt.



Aufgabe 4.3 Balkenplan (8 Punkte)

Nach der Fertigstellung des Netzplans erkennen Sie, dass die Implementierung der BMZ-Software (AP F) von der Implementierung der Sensoren-Software (AP E) abhängig ist.

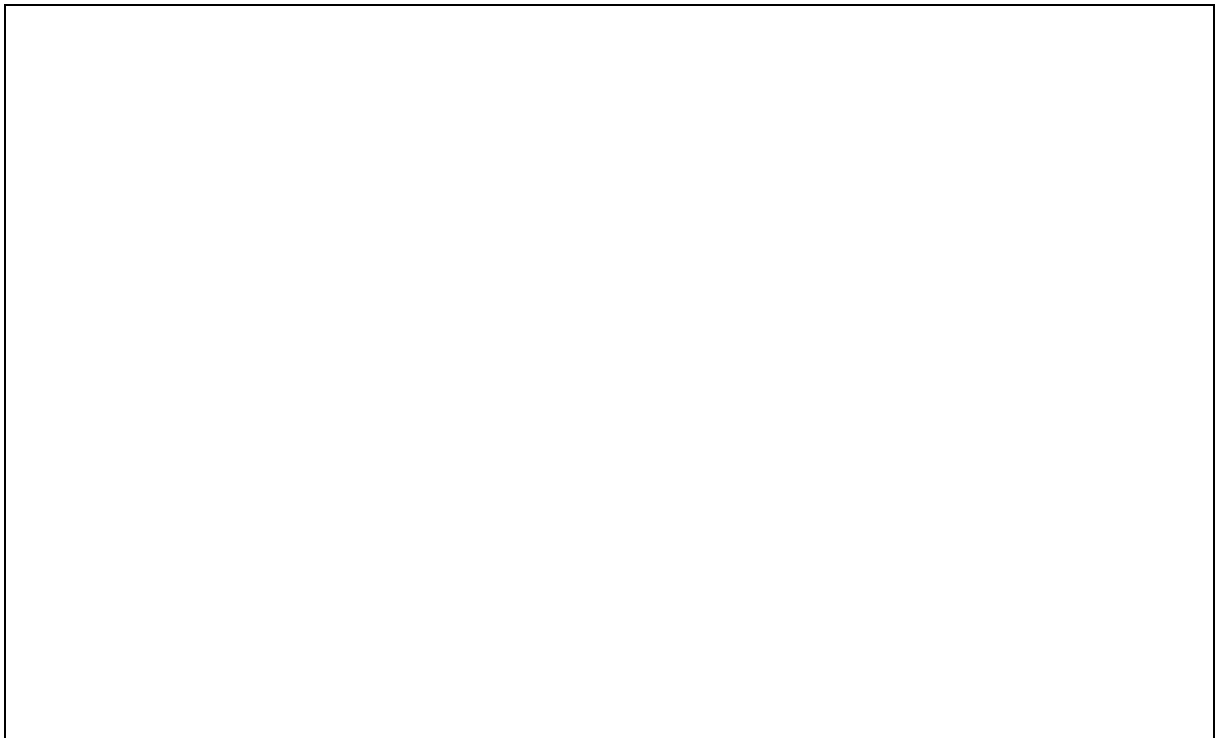
- a) Wie wirkt sich diese neue Abhängigkeit auf den Projektendtermin aus? (1 Punkt)

- b) Wie wirkt sich diese neue Abhängigkeit auf den kritischen Pfad bzw. die kritischen Pfade aus? (1 Punkt)

- c) Visualisieren Sie den geplanten Projektverlauf, unter Berücksichtigung der neuen Abhängigkeit, in einem Balkenplan. Zeichnen Sie alle Arbeitspakete und deren Abhängigkeiten ein. (4 Punkte)

AP	5	10	15	20	25	30	35	40	
A									
B									
C									
D									
E									
F									
G									
H									
I									
J									
K									

- d) Welche Vor- und Nachteile haben Netzpläne und Balkenpläne? Geben Sie jeweils einen, aus der Lehrveranstaltung bekannten, Vor- und Nachteil an. (2 Punkte)



Aufgabe 5: Kurzfragen (30 Punkte)

Wird nicht veröffentlicht!

$\Sigma =$