Technikerschule HF Zürich

Software Engineering (A&D)

Projektarbeit

Anforderungsanalyse

Design eines Kundenverwaltungssystems für die Firma PackZeugs AG

Datum: 23.12.2013

Autoren: Pascal Kern
David Marmy

Klasse: TSI1209I

Inhaltsverzeichnis

5. Aufgabe: Objektorientierte Analyse, Design & Programmieren (OOA/D/P) (Analyse- und Entwurfsmodell anpassen, Entwurfsmodell implementieren)

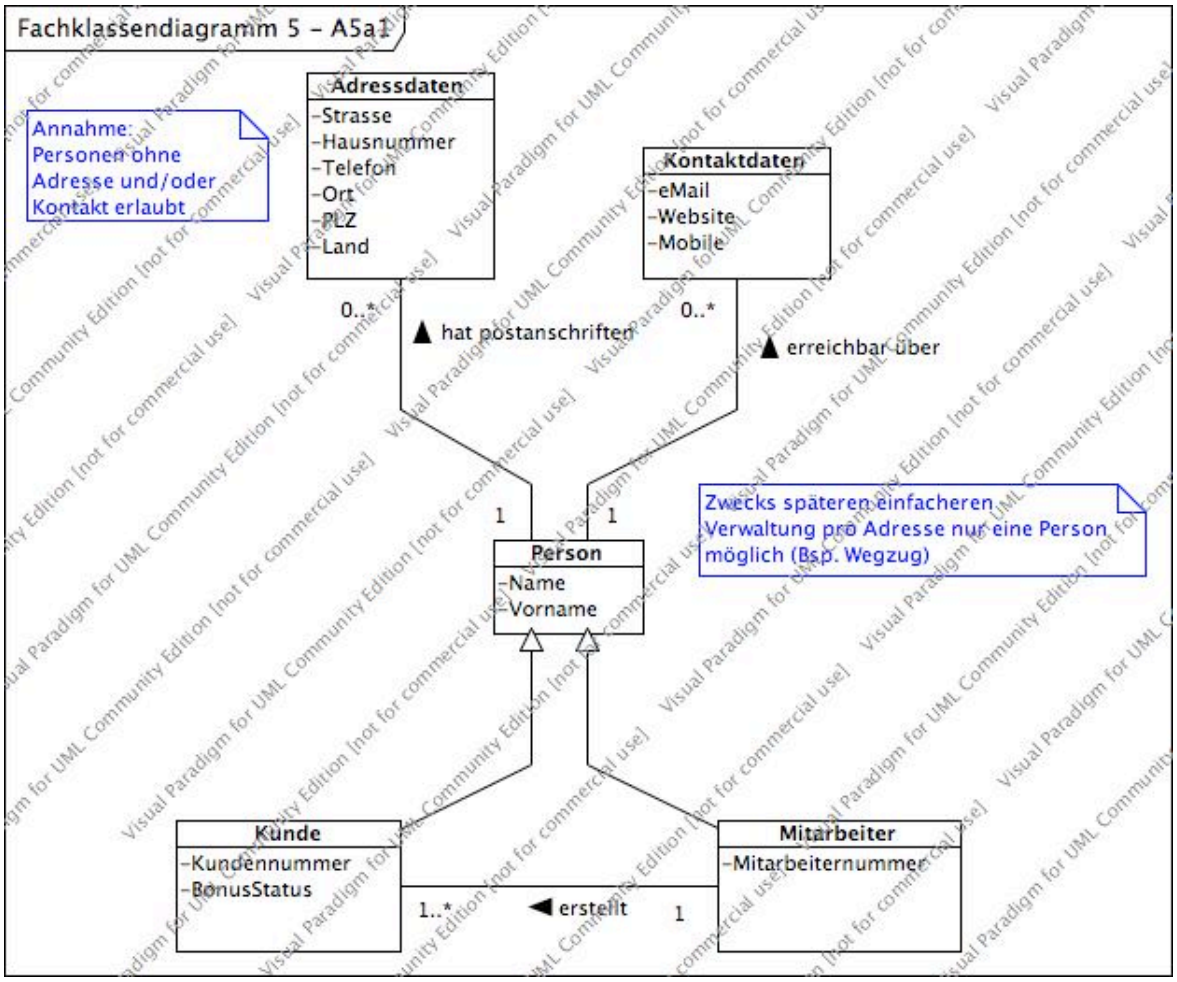
5.1. Anpassungen (Fach-) Klassendiagramm - Analysemodell 3
5.2. Angepasstes Klassendiagramm - (Entwurf-) Designmodell 4
5.3. Erweiterung Design- Entwurfsmodell in “Business” und “Persistence” Schicht um abstrakten Klassen und Schnittstellen 4
5.4. Klassen- und Komponentendiagramm mit “Singleton” und “Factory” (GoF) Design Pattern auf Business und Persistenc Schicht (3-Schichten Modell) 5
5.5. Sequenzdiagramm zur Überprüfung Normalfall “Kunde erstellen” - Korrekturen ableiten 6
5.6. Glossar ergänzen um Klassen und Schnittstellen aus Design- Entwurfsmodell [5.3] 6
Layer Presentation 6
Layer BusinessLogic 6
Layer Persistence 7
5.7. Angepasstes Design- Entwurfsmodell implementieren [5.3] 8
5.8. JUnit Test auf “Business” und “Persistenz” Schicht für Normalfall “Kunde erstellen” 8
5.9. Normalfall “Kunde erstellen” und “Kunde anzeigen” testen 8
5.10. JavaDoc generieren 8

6. Anhang

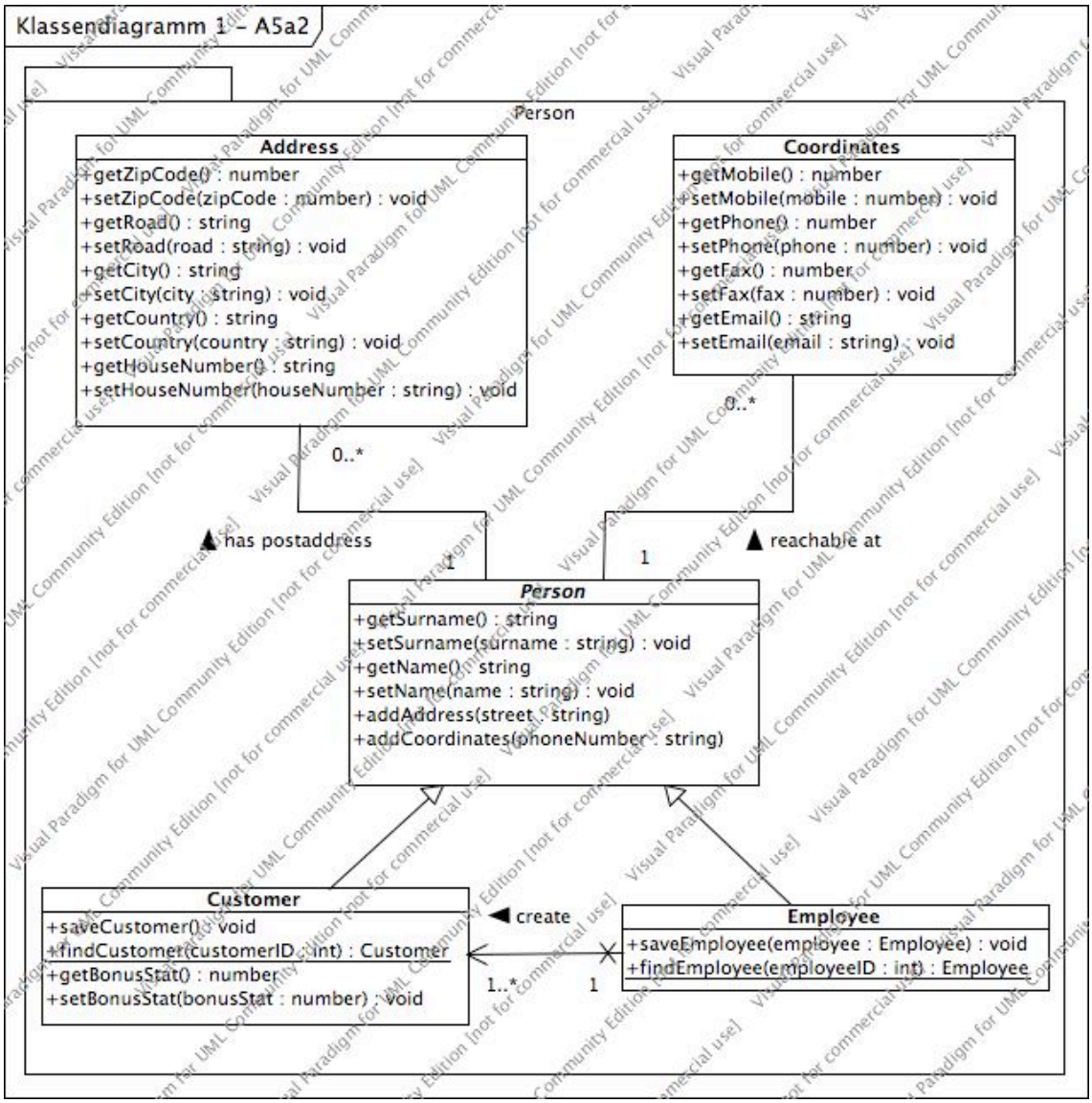
6.1. Aufgabenstellungen 9
Aufgabenstellung 5: 9
Bewertungskriterien Aufgabe 5: 10
6.2. Abgaberichtlinien 11
6.3. Zusatzinfos zu den Aufgaben 12
Aufgabe 4 12
Aufgabe 5 12

5. Aufgabe: Objektorientierte Analyse, Design & Programmieren (OOA/D/P) (Analyse- und Entwurfsmodell anpassen, Entwurfsmodell implementieren)

5.1. Anpassungen (Fach-) Klassendiagramm - Analysemodell



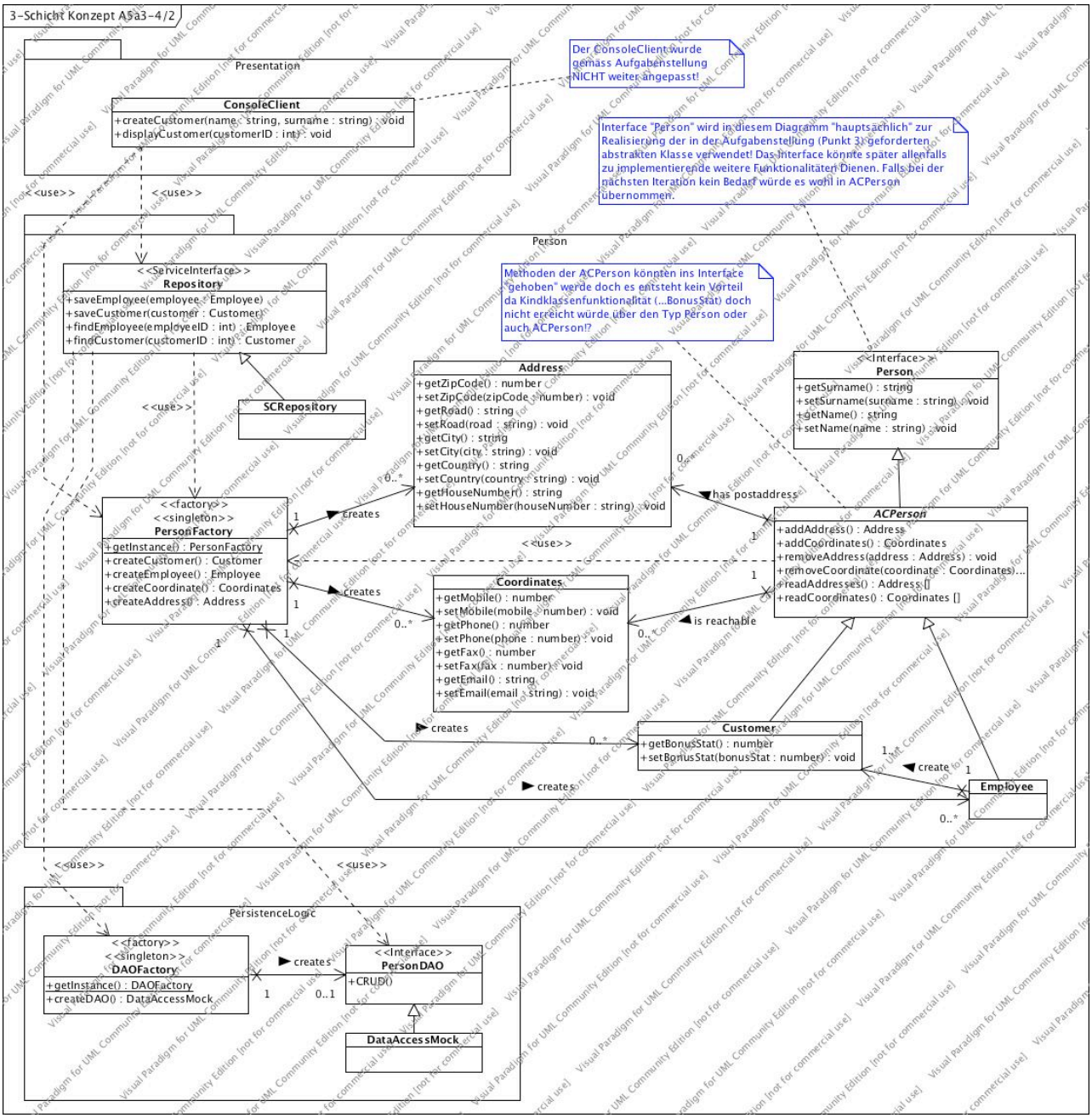
5.2. Angepasstes Klassendiagramm - (Entwurf-) Designmodell



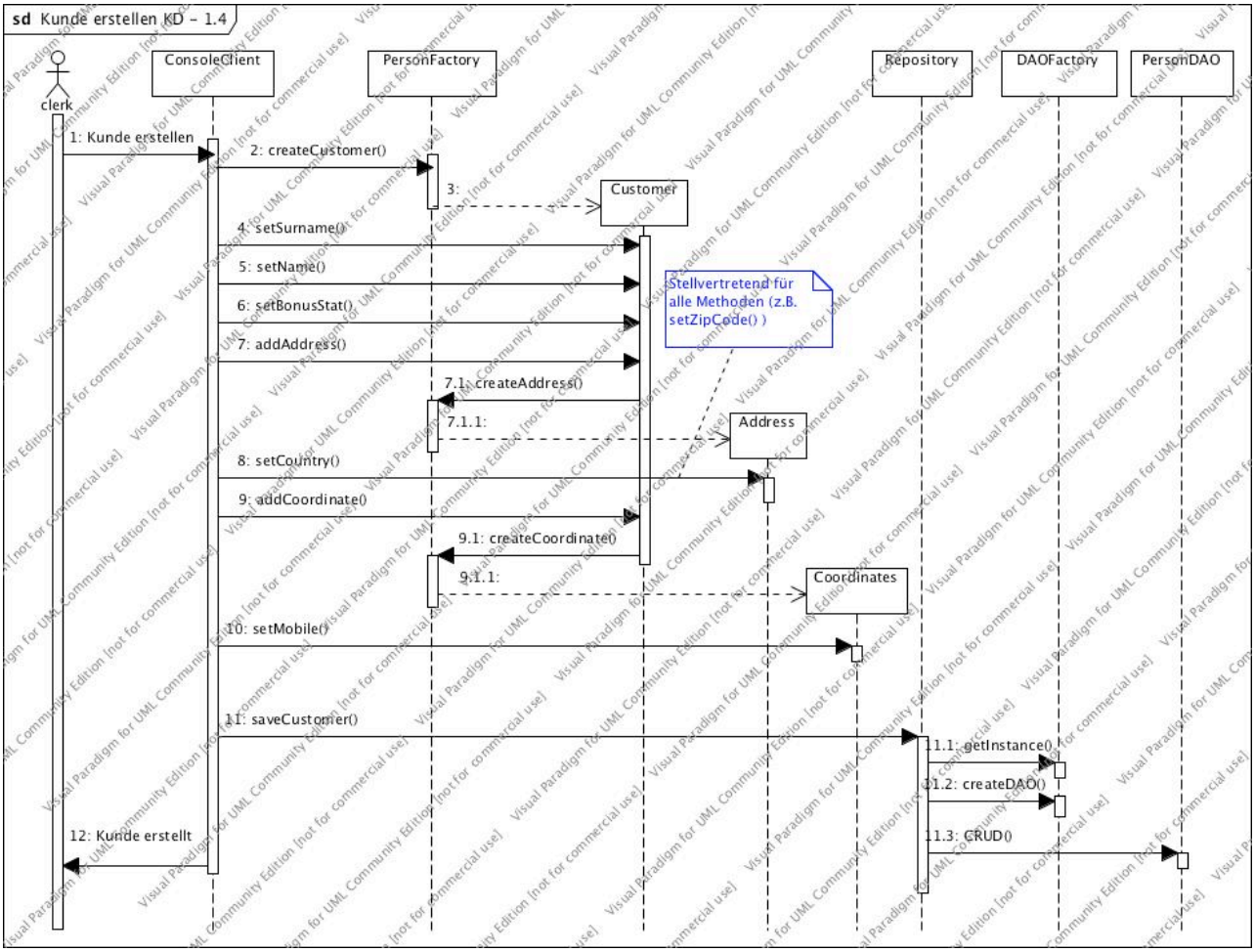
5.3. Erweiterung Design- Entwurfsmodell in “Business” und “Persistence” Schicht um abstrakten Klassen und Schnittstellen

Siehe nächste Aufgabe (5.4). Zusammengefast zu einer Aufgabe da für uns die Unterscheidung der Aufgabeninhalte nicht nachvollziehbar ist!

5.4. Klassen- und Komponentendiagramm mit “Singleton” und “Factory” (GoF) Design Pattern auf Business und Persistenc Schicht (3-Schichten Modell)



5.5. Sequenzdiagramm zur Überprüfung Normalfall “Kunde erstellen” - Korrekturen ableiten



5.6. Glossar ergänzen um Klassen und Schnittstellen aus Design-Entwurfsmodell [5.3]

Layer Presentation

Class/Interface (Technical term)	Synonyme	Beschreibung Description
ConsoleClient		Stellt einfachste Funktionalität für die Erstellung und Anzeige von Kunden zur Verfügung.
		Provides simple functionality for creating and displaying customer.

Layer BusinessLogic

Class/Interface (Technical term)	Synonyme	Beschreibung Description
Customer	Client, Consumer, --	Identifikation eines Bedarfsträgers beinhaltet Informationen zur Geschäftsbeziehung zwischen ihm (dem Kunden) und der Unternehmung. Bezieht gegen ein entsprechendes Entgelt eine Leistung vom (unserem) Unternehmen. Hält Informationen zu Adresse(n), Koordinate(n) und Bonusstufe.
		Identification of a carrier includes information required for business relationship between him (the client) and the company. Refers to rates of pay, a performance from (our) company. Holds information about address (es) coordinate (s) and bonus level.

Class/Interface (Technical term)	Synonyme	Beschreibung Description
Person <<interface>>	People, Human	Stellt den Zugang zu Vor- und Nachnahme eines (Geschäfts-) Partner zur Verfügung. ~~Beispiel: Gemeinsamkeiten von Kunden, Mitarbeitern aber kann auch für Lieferanten oder Geschäftspartner stehen.~~
		Provides access to first and last name of a (business partner. ~~For example, similarities of customers, employees but can also stand for suppliers or business partners.~~
ACPerson		Funktionalität zum einfügen, halten und entfernen von Adressen- und Koordinatendaten einer Person.
		To add functionality, hold and remove addresses and coordinate data of a person.
Employee	Personnel, Workforce, Employee(s)	Identifiziert jemand, der für die Unternehmung mit anderen zusammenarbeitet; Daten pflegt und Änderungen im Kundensystem vornimmt.
		Identifies someone who works for the company with others, maintains and modifies data in the customer system.
Address	--, Point of contact	Trägt Informationen zur Bezeichnung des Wohnorts, der Straße und der Hausnummer einer schriftlichen Kontaktmöglichkeit. Postalische Domizil-Daten einer Person oder Unternehmung.
		Carries information about the name of the place, the street and the house number written contact information. Postal domicile of a person or company data.
Coordinates	Coordinates, Contact-Personnel	Daten für die direkte Kommunikation mit einer Person. Ergänzend zu den Adressdaten wie allfällige Telefon, eMail, Website, Ansprechperson usw.
		Data for direct communication with a person. In addition to the address as any telephone, email, website, contact person, etc.
Repository <<interface>>		Zugriff auf Funktionalität der Klasse Repository hauptsächlich für Testzwecke der Repository Funktionalität. (einfacher Einsatz eines von Test-MOKs)
		Access to functionality of the class repository mainly for test purposes, the repository functionality. (simple use of a test-Mok)
SCRepository	Toolbox, Toolservice	Funktionen zur Verwaltung und Benutzung von Objekten der Klassen Kunde und Angestellter. Delegiert den Zugriff auf die Datenbank an die Schnittstelle PersonDAO und bildete Objekte des Layer Business über die PersonFactory.
		Functions for the management and use of objects of class customer and employee. Delegates to access the database at the interface personDao and formed objects of Layer Business about the person Factory.
PersonFactory		Kontrolliert die Erstellung von Objekten im Package Person.
		Controls the creation of objects in the package person.

Layer Persistence

Class/Interface (Technical term)	Synonyme	Beschreibung Description
DAOFactory		Kontrolliert den Lebenszyklus von Objekten mit der PersonDAO Schnittstelle.
		Controls the life cycle of objects with the personDAO interface.
PersonDAO <<interface>>		Database-Access-Object (Baustein) Ermöglicht das Schreiben, Lesen, Ändern und Löschen von Angestellten- oder Kundendaten. CRUD Funktionalität (Create-Read-Update-Delete)
		Database Access Object (block) Allows you to write, read, modify, and delete employee or customer data. CRUD functionality (Create-Read-Update-Delete)
DataAccessMock		Verarbeitet die Angestellten- und Kundendaten In-Memory.
		Processes the employee and customer data in-memory.

5.7. ~~Angepasstes Design- Entwurfsmodell implementieren [5.3]~~

5.8. ~~JUnit Test auf “Business” und “Persistenz” Schicht für Normalfall “Kunde erstellen”~~

5.9. ~~Normalfall “Kunde erstellen” und “Kunde anzeigen” testen~~

5.10. ~~JavaDoc generieren~~

A. Anhang

1. Aufgabenstellungen

Aufgabenstellung 5:

Aufgabe 5:
Objektorientierte Analyse, Design & Programmieren (OOA/D/P) (Analyse- und Entwurfsmodell anpassen, Entwurfsmodell implementieren)

Ausgabetermin: 30.11.2013 **Abgabetermin:** 10.12.2013, 02.01.2014

Ausgangslage:

Folgende neuen Anforderungen der Firma PackZeug AG liegen für die 2. Iteration vor:

- Kunden können mehr als eine Anschrift, ein Telefon, eine E-Mail etc. haben
- Je nach Anzahl Bestellungen im Jahr haben Kunden einen Bronze- ‚Silber- oder Goldstatus. Der Status bringt den Kunden Vorteile wie z.B. Preisrabatte

Die 2. Iteration umfasst folgende Aufgaben:

- Umsetzung der neuen Anforderungen
- Vollständige Behandlung der Kundendaten (Name, Vorname und Kontaktinformationen)
- Automatisiertes Testen der Module
- Umsetzung der nicht-funktionalen Anforderungen Wartbarkeit und Wiederverwendbarkeit

Im Rahmen dieser Aufgaben müssen Sie das vorliegende Analyse- und Entwurfsmodell anpassen und das angepasste Entwurfsmodell in der Programmiersprache Java implementieren.

Aufträge:

1. Passen Sie das UML Klassendiagramm (Fachklassenmodell) des Analysemodells an für die neuen Anforderungen.
2. Passen Sie das UML Klassendiagramm des Entwurfsmodells an das angepasste UML Klassendiagramm des Analysemodells an.
3. Erweitern Sie Ihr Entwurfsmodell auf den Schichten "Business" und "Persistence" um abstrakte Klassen und Schnittstellen. Stellen Sie das angepasste Entwurfsmodell mit einem UML Klassen- und Komponentendiagramm dar.
4. Wenden Sie auf den Schichten "Business" und "Persistence" die Entwurfsmuster "Singleton" und "Factory" an. Stellen Sie das angepasste Entwurfsmodell mit einem UML Klassen- und Komponentendiagramm dar.
5. Überprüfen Sie mit einem UML Sequenzdiagramm Ihr Entwurfsmodell für den Normalablauf des Anwendungsfalls „Kunde erstellen“ hinsichtlich Vollständigkeit und Verantwortlichkeiten der Klassen und Schnittstellen sowie Interaktionen der entsprechenden Objekte. Ergänzen Sie bei Bedarf das Entwurfsmodell nach dieser Überprüfung.
6. Ergänzen Sie das Glossar um die neuen abstrakten Klassen sowie Schnittstellen aus dem Entwurfsmodell.
7. Implementieren Sie Ihr angepasstes Entwurfsmodell.
8. Implementieren Sie für die Schichten "Business" und "Persistence" Unit-Tests mit JUnit für den Anwendungsfall „Kunde erstellen“.
9. Testen Sie Ihre Implementierung für die Normalabläufe der Anwendungsfälle "Kunde erstellen" und "Kunde anzeigen", indem Sie mehrere Kunden erstellen, speichern, lesen und auf der Konsole deren Daten ausgeben.
10. Dokumentieren Sie den Java-Quelltext (nur Dateiköpfe und Methoden) mittels Javadoc und erzeugen Sie die zugehörigen HTML-Dokumentationsseiten.

Bewertungskriterien Aufgabe 5:

Bewertung (mittels Kontrolle und/oder Gespräch, max. 45 Punkte):

#	Bewertungskriterien (Punkte: 0 => nicht erfüllt; 1 => >0% & <50% erfüllt; 2 => >=50% erfüllt; 3 => 100% erfüllt)	Punkte			
		0	1	2	3
1	Die Erweiterung des Fachklassenmodell (UML Klassendiagramm) des Analysemodells um die neuen Anforderungen ist korrekt und vollständig.				
2	Das Entwurfsmodell (UML Klassendiagramm) auf Basis des angepassten Analysemodells ist korrekt und vollständig.				
3	Die Erweiterung des Entwurfsmodells um abstrakte Klassen und Schnittstellen ist korrekt und vollständig.				
4	Die Anwendung der Entwurfsmuster "Singleton" und "Factory" auf das Entwurfsmodell korrekt und vollständig.				
5	Szenario (UML Sequenzdiagramm) für das Entwurfsmodell für den ausgewählten Anwendungsfall ist korrekt und vollständig.				
6	Abstrakte Klassen sowie Schnittstellen aus dem Entwurfsmodell sind korrekt und vollständig ins Glossar eingetragen.				
7	Implementation der Persistenzschicht (DataAccessMock) ist korrekt und vollständig.				
8	Implementation der Geschäftsschicht (fachliches Entwurfsmodell) ist korrekt und vollständig.				
9	Implementation der Präsentationsschicht (ConsoleClient) ist korrekt und vollständig.				
10	Die Implementation der Unit-Tests für die Persistence-Schicht ist korrekt und vollständig.				
11	Die Implementation der Unit-Tests für die Business-Schicht ist korrekt und vollständig.				
12	Die Implementation läuft fehlerfrei (Daten der Kunden werden gespeichert und gelesen).				
13	Quelltext-Dokumentation mit Javadoc ist korrekt und vollständig.				
14	Namen in den UML Diagrammen sind aussagekräftig und die bekannten UML Namenskonventionen werden beachtet.				
15	Namen im Java-Quelltext sind aussagekräftig und die bekannten Java Namenskonventionen werden beachtet.				

2. Abgaberichtlinien

Die Abgabe der bearbeiteten Projektteilaufgaben erfolgt per E-Mail an den Dozenten. Folgender Betreff ist für die E-Mail zu verwenden:

- `SE AD: Team <#>: Aufgabe <#>`
  Beispiel: `SE AD: Team 3: Aufgabe 3`

- `SE AD: <Vorname> <Nachname>: Aufgabe <#>`
  Beispiel: `SE AD: Beat Müller: Aufgabe 2`

Erlaubte Dateiformate

- PDF
- ODF-Formate (ODT, ODS)
- JPG
- ZIP
- Microsoft-Formate (DOCX, XLSX)

Regeln für Dateinamen

Dateien sind nach folgendem Schema zu benennen:

- `aufgabe_<#>_team_<#>.<erweiterung>`
  Beispiel: `aufgabe_1_team_3.odt`

- `aufgabe_<#>_<vorname>_<nachname>.<erweiterung>`
  Beispiel: `aufgabe_1_beat_mueller.odt`

Hinweis: Keine Leer- oder Sonderzeichen für Dateinamen verwenden

Dokumentenform

- Die Lösungsteile (Texte, Tabellen, Diagramme etc.) zu einer Aufgabe sind in einem einzelnen Dokument zusammenzufassen

- Ein Dokument für alle Aufgaben ist zu verwenden

- Das Titelblatt des Dokuments muss folgende Angaben beinhalten
  - Fach/Vorlesungstitel: `Technikerschule HF Zürich – Software Engineering (A&D)`
  - Haupttitel `Projektarbeit`, Untertitel gemäss Aufgabe
  - Datum (TT.MM.JJJJ)
  - Vorname(n) und Nachname(n) des Autors (der Autoren)
  - Klasse

- Je Aufgabe ein Hauptkapitel mit einem entsprechenden Namen

- UML-Diagramme sind als Grafiken in das Dokument zu integrieren. Falls dies nicht vernünftig darstellbar möglich ist, sind die betreffenden UML-Diagramme als JPG-Dateien beizulegen und im Dokument entsprechend zu referenzieren

- Texte, Tabellen, Diagramme etc. sind durch Überschriften bzw. Titeln den einzelnen Aufträgen einer Aufgabe zuzuordnen

3. Zusatzinfos zu den Aufgaben

Aufgabe 5

Hinweise:

- Allgemein:
  - Implementieren Sie nicht alles auf einmal, sondern gehen Sie nach dem Prinzip vor „Code a little, test a little“.
  - Implementieren Sie nur die Anwendungsfälle „Kunde anlegen“ und „Kunde anzeigen“.
  - Speichern und lesen Sie nur Vor- und Nachname der Kunden.
- Auftrag 7:
  - Optional können Sie Array durch ArrayList ersetzen, um die Implementierung zu vereinfachen
- Auftrag 8:
  - Legen Sie für die Unit-Tests folgende neuen Java-Packages an:
    - Business-Schicht
      ch.<WURZEL>.test.business
    - Persistence-Schicht
      ch.<WURZEL>.test.presentation
- Auftrag 10:
  - Nur Dateiköpfe und Methoden dokumentieren
  - getter- und setter-Methoden dürfen zusammengefasst dokumentiert werden