Technikerschule HF Zürich

Software Engineering (A&D)

Projektarbeit

Anforderungsanalyse

Design eines Kundenverwaltungssystems für die Firma PackZeugs AG

Datum: 09.12.2013

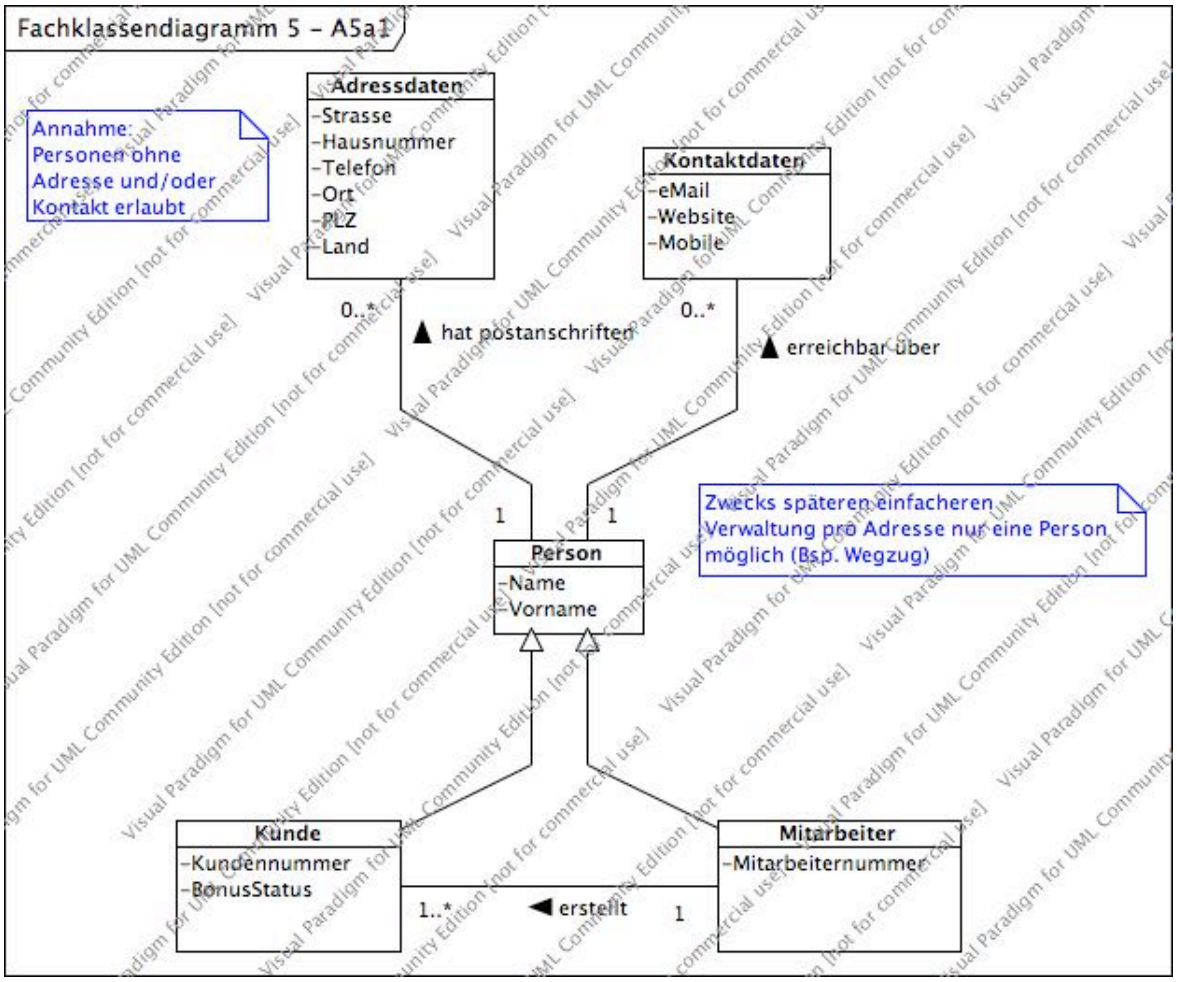
Autoren: Pascal Kern
David Marmy

Klasse: TSI1209I

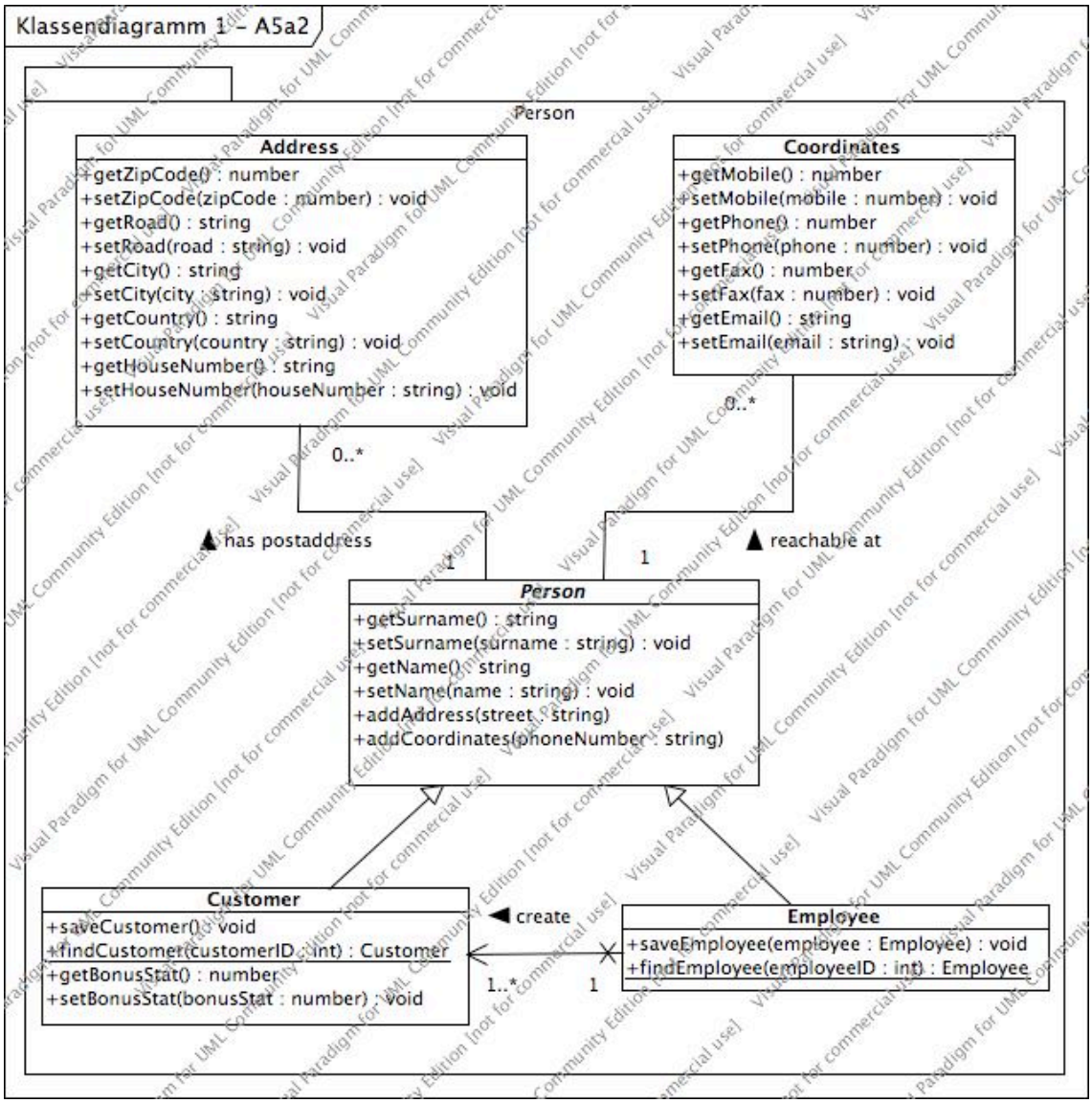
Inhaltsverzeichnis

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

5.1. Anpassungen (Fach-) Klassendiagramm - Analysemodell



5.2. Angepasstes Klassendiagramm - (Entwurf-) Designmodell



6. Anhang

6.1. 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:

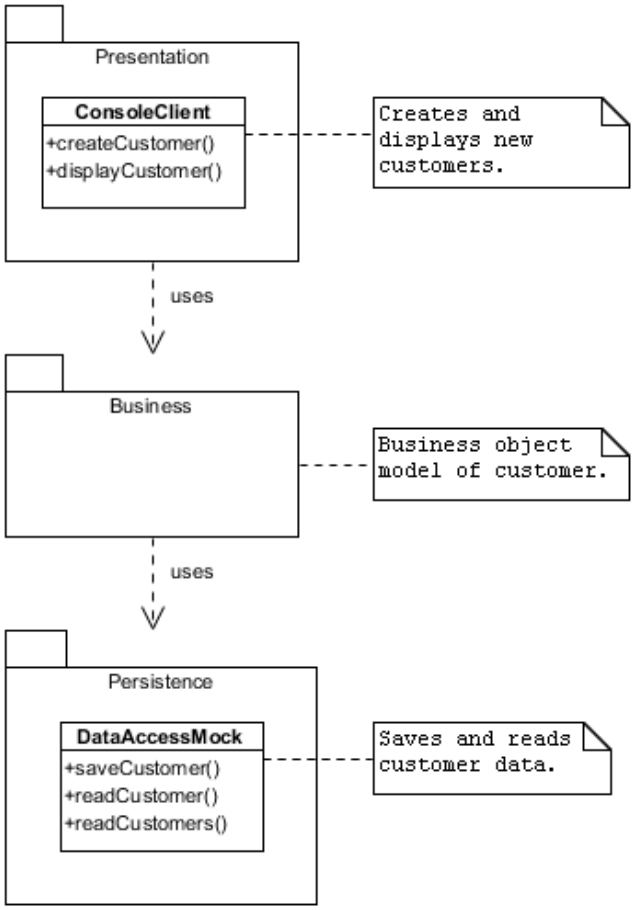
Bewertung (mittels Kontrolle und/oder Gespräch, max. 45 Punkte):

#	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.				

6.2. Zusatzinfos zu den Aufgaben

Aufgabe 4

Gegebene logische 3-Schichtenarchitektur:



Hinweise zu den Aufträgen 5 und 6:

- 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.

- Vorbereitung:
  - Erstellen Sie ein neues Eclipse-Projekt.
  - Legen Sie im neuen Eclipse-Projekt folgende Java-Packages an:
    - Presentation-Schicht
      ch.<WURZEL>.persistence
    - Business-Schicht
      ch.<WURZEL>.business
    - Persistence-Schicht
      ch.<WURZEL>.presentation

1. Persistence-Schicht:
   - Implementieren Sie die Klasse DataAccessMock.
   - Speichern Sie im DataAccessMock Kundendaten (Vor- und Nachname) nicht im Dateisystem, sondern in-memory (im Hauptspeicher mittels Arrays und Strings).

2. Business-Schicht
   - Implementieren Sie aus Ihrem Fachklassenmodells nur die Teile, die für die Anwendungsfälle "Kunde erstellen" und "Kunde anzeigen" benötigt werden.

3. Presentation-Schicht
   - Implementieren Sie die Klasse ConsoleClient.
     - Erstellen und speichern Sie mehrere Kunden
       - Verzichten Sie auf Benutzereingaben über die Tastatur und arbeiten Sie stattdessen mit "harten" Testdaten für die Daten der Kunden
     - Lesen Sie die Kunden und geben Sie diese wieder auf der Konsole aus.

Hinweise zum Auftrag 7:

- Nur Dateiköpfe und Methoden dokumentieren.
- getter- und setter-Methoden dürfen zusammengefasst dokumentiert werden.

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