

Projektarbeit

Anforderungsanalyse

**Design eines Kundenverwaltungssystems für die Firma
PackZeugs AG**

Datum: 24.11.2013

Autoren: Pascal Kern
David Marmy

Klasse: TSI1209I

Inhaltsverzeichnis

4. Aufgabe: Objektorientiertes Design & Programmieren (OOA/D/P) (Entwurfsmodell erstellen und implementieren)

4.1. Glossar ergänzen englische Namen der Klassen	3
Bemerkung zu den folgenden Diagrammen.....	4
4.2. Fachklassendiagramm 1. Iteration in Entwurfsmodell Fachklassenmodell überführen	4
4.3. Fachklassenmodell in 3-Schichtenarchitektur integrieren.....	5
4.4. Entwurfsmodell mit Sequenzdiagramm überprüfen	6
4.5. Entwurfsmodell in Prototyp implementieren.....	6
4.6. Prototyp testen	7
4.7. Quelltext per Javadoc als HTML-Dokument erstellen.....	7

5. Anhang

5.1. Aufgabenstellungen	8
Aufgabenstellung 4:	8
Bewertungskriterien Aufgabe 4	9
5.2. Abgaberichtlinien	10
5.3. Zusatzinfos zu den Aufgaben	11
Aufgabe 4.....	11

4. Aufgabe: Objektorientiertes Design & Programmieren (OOA/D/P) (Entwurfsmodell erstellen und implementieren)

4.1. Glossar ergänzen englische Namen der Klassen

Der Vollständigkeit halber haben wir neben den Namen und Synonymen auch die Bedeutung übersetzt. Dies muss nicht unbedingt bewertet werden.

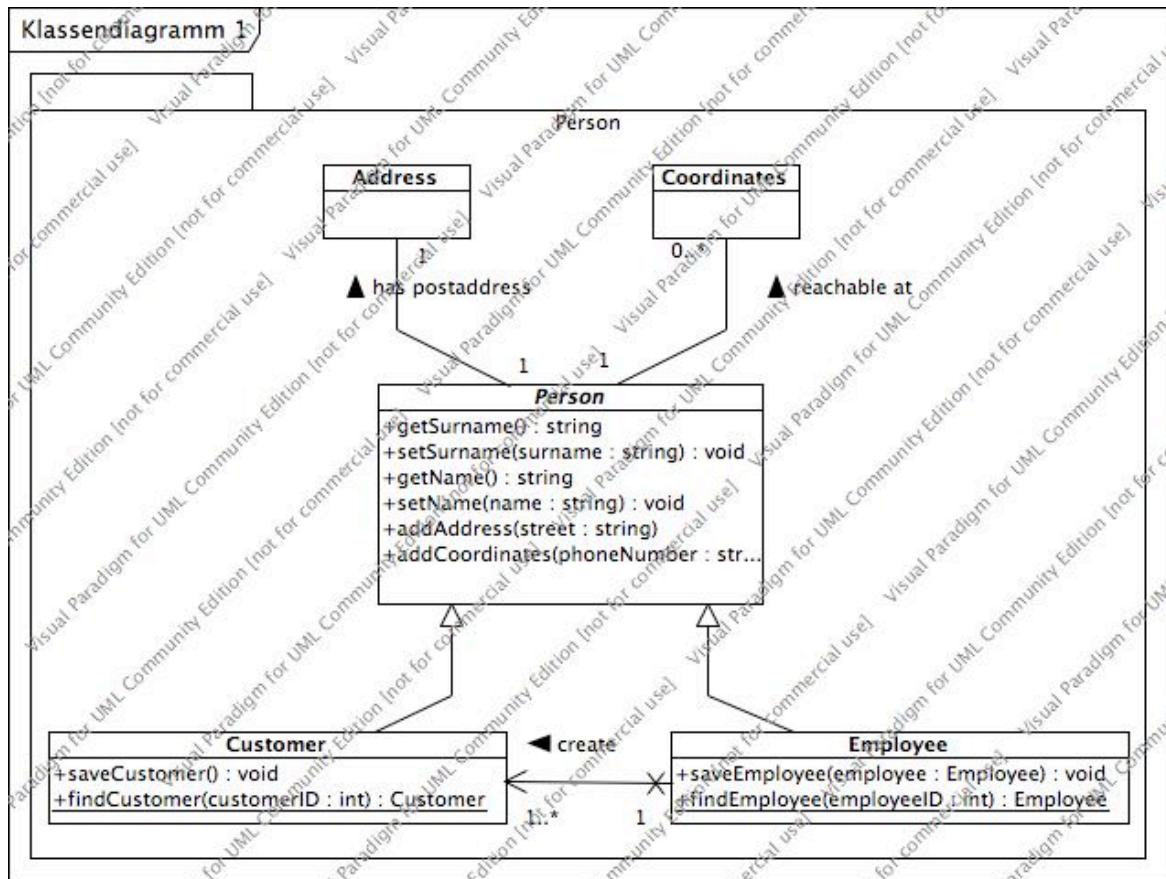
Bezeichnung	Übersetzung	Synonyme (nicht bewerten)	Übersetzung	Beschreibung Description
Kunde	Customer	Klient, Konsument, Verbraucher	Client, Consumer, --	Identifikation eines Bedarfsträgers beinhaltet Informationen zur Geschäftsbeziehung zwischen ihm (dem Kunden) und der Unternehmung. Bezieht gegen ein entsprechendes Entgelt eine Leistung vom (unserem) Unternehmen. 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.
Person	Person	Leute, Mensch	People, Human	<u>Abstrakte Darstellung!</u> Stellt den Zugang zu zusätzlichen Daten wie Anschrift, Kontaktinformationen eines (Geschäfts-) Partner zur Verfügung. Beispiel: Gemeinsamkeiten von Kunden, Mitarbeitern aber kann auch für Lieferanten oder Geschäftspartner stehen. <u>Abstract representation!</u> Provides access to additional information such as address, contact information of a (business) partner. For example, similarities of customers, employees but can also stand for suppliers or business partners.
Gender	Geschlecht			Angabe zum Kunden ob er männlich oder weiblich ist. Informations about the client if it's a male or female.
Mitarbeiter	Employee	Personal, Belegschaft, Angestellte(r)	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.
Adressdaten	Address	Anschrift, Kontaktstelle	--, 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.
Kontakt	Coordinates	Koordinaten, Kontakt-Personalien	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.

Bemerkung zu den folgenden Diagrammen

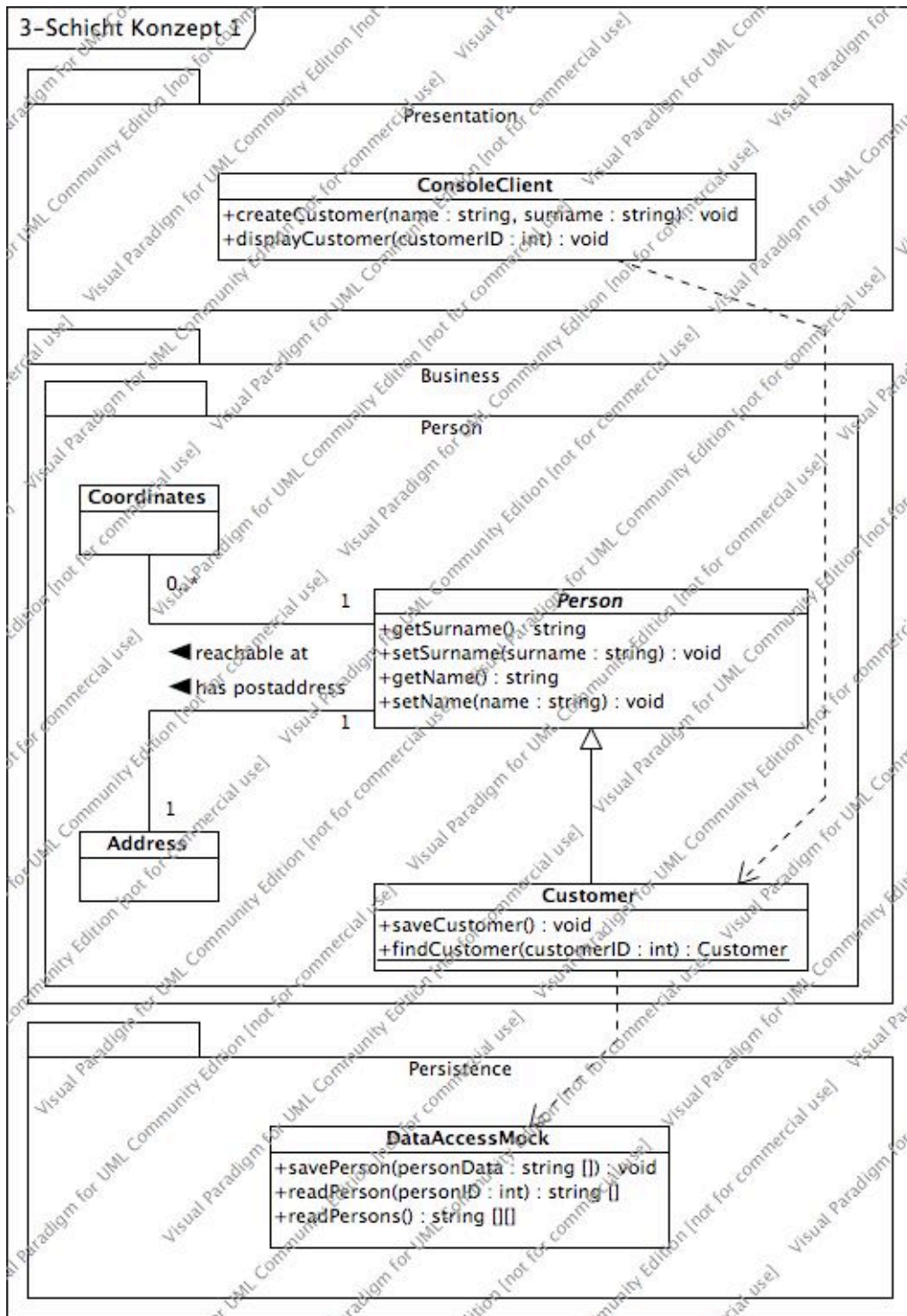
Gemäss Aufgabenstellung wurden nur die Eigenschaften Vor- und Nachname aufgeführt. Die im Analysemodell vorhandenen Eigenschaften Geschlecht (gender), Kundennummer (clientID) und Mitarbeiternummer (staffID) wurden also nicht "vergessen" sondern bewusst weg gelassen.

Bei den Methoden wurden die für die Klassen Adresse (Adress) und Kontakt (Coordinates) aber dennoch aufgenommen um die Logik einigermassen komplett dar zu stellen.

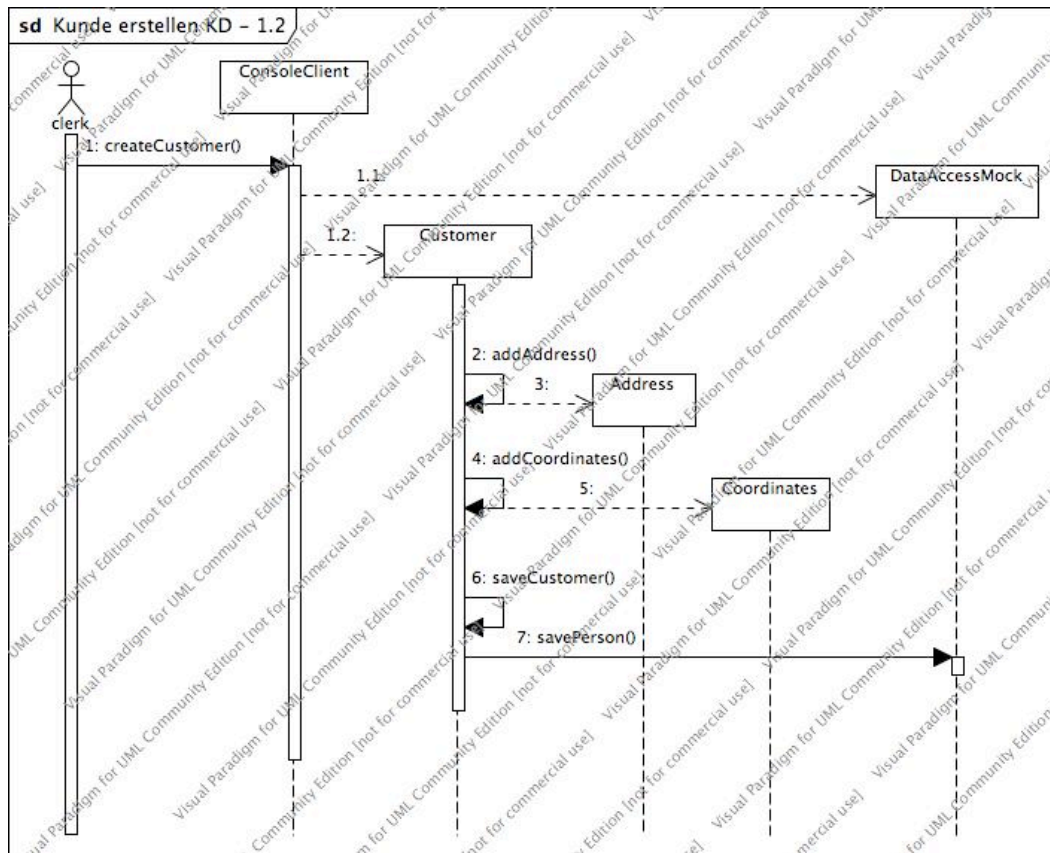
4.2. Fachklassendiagramm 1. Iteration in Entwurfsmodell Fachklassenmodell überführen



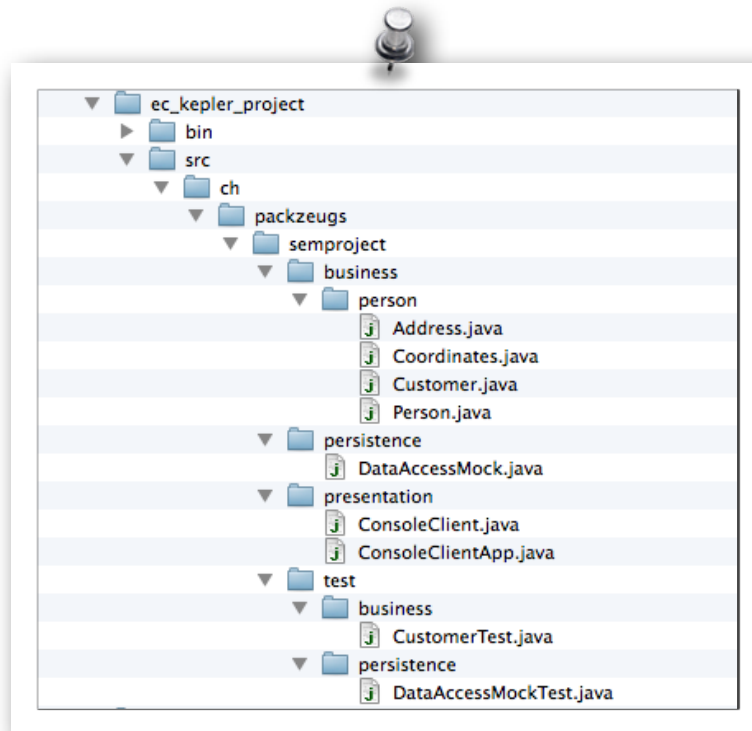
4.3. Fachklassenmodell in 3-Schichtenarchitektur integrieren



4.4. Entwurfsmodell mit Sequenzdiagramm überprüfen



4.5. Entwurfsmodell in Prototyp implementieren

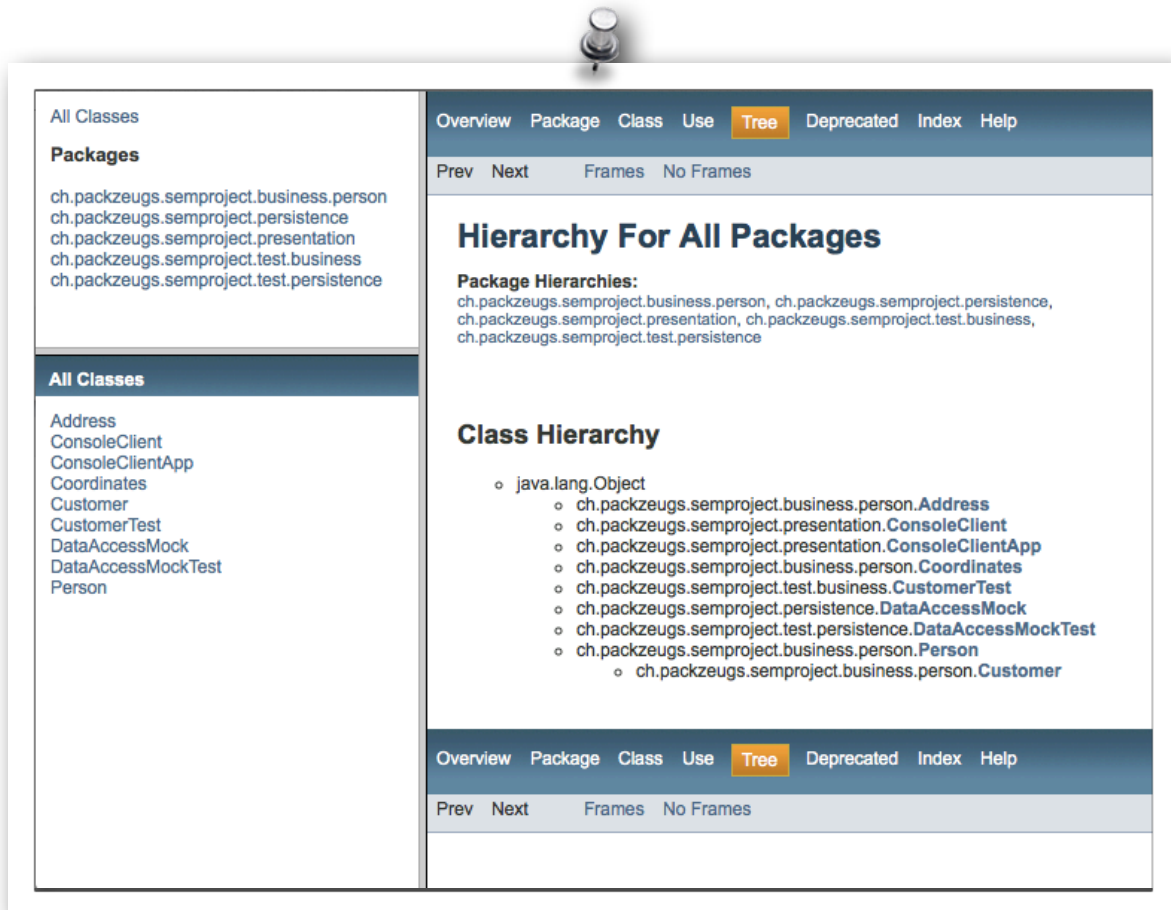


4.6. Prototyp testen

Einzelne Pakete (Schichten vom 3-Schicht Konzept) einzeln erfolgreich mit den Tests im Paket "ch.packzeugs.semproject.test" getestet.

Das Zusammenspiel der drei Schichten wurde mit der Klasse "ConsoleClientApp" erfolgreich durchgeführt und kann somit auch als erfolgreich getestet betrachtet werden.

4.7. Quelltext per Javadoc als HTML-Dokument erstellen



The screenshot displays a Javadoc-generated HTML document for a Java project. The document is titled "Hierarchy For All Packages" and shows a class hierarchy starting from `java.lang.Object`. The hierarchy includes the following classes:

- `java.lang.Object`
 - `ch.packzeugs.semproject.business.person.Address`
 - `ch.packzeugs.semproject.presentation.ConsoleClient`
 - `ch.packzeugs.semproject.presentation.ConsoleClientApp`
 - `ch.packzeugs.semproject.business.person.Coordinates`
 - `ch.packzeugs.semproject.test.business.CustomerTest`
 - `ch.packzeugs.semproject.persistence.DataAccessMock`
 - `ch.packzeugs.semproject.test.persistence.DataAccessMockTest`
 - `ch.packzeugs.semproject.business.person.Person`
 - `ch.packzeugs.semproject.business.person.Customer`

The document also includes a sidebar with "All Classes" and "Packages" sections. The "All Classes" section lists the following classes:

- `Address`
- `ConsoleClient`
- `ConsoleClientApp`
- `Coordinates`
- `Customer`
- `CustomerTest`
- `DataAccessMock`
- `DataAccessMockTest`
- `Person`

The "Packages" section lists the following packages:

- `ch.packzeugs.semproject.business.person`
- `ch.packzeugs.semproject.persistence`
- `ch.packzeugs.semproject.presentation`
- `ch.packzeugs.semproject.test.business`
- `ch.packzeugs.semproject.test.persistence`

5. Anhang

5.1. Aufgabenstellungen

Aufgabenstellung 4:

Aufgabe 4:**Objektorientiertes Design & Programmieren (OOA/D/P)
(Entwurfsmodell erstellen und implementieren)****Ausgabetermin:** 05.10.2013**Abgabetermin:** 28.10.2013**Ausgangslage:**

Auf Basis des vorliegenden Analysemodells müssen Sie unter besonderer Beachtung der nicht-funktionalen Anforderungen für die 1. Iteration ein Entwurfsmodell erstellen und dieses als Prototyp in der Programmiersprache Java implementieren. Die 1. Iteration umfasst die Klasse (bzw. Klassen) Ihres Analysemodells, die die Abstraktion "Kunde" modelliert. Für die 1. Iteration realisieren Sie die Normalabläufe der Anwendungsfälle "Kunde erstellen" und "Kunde anzeigen". Es genügt, wenn Sie dabei die Eigenschaften Vor- und Nachname von Kunden berücksichtigen.

Auftrag:

1. Ergänzen Sie das Glossar um die ins Englische übersetzten Namen der Klassen des Fachklassenmodells.
2. Überführen Sie für die 1. Iteration das Fachklassenmodell des Analysemodells in ein Fachklassenmodell für das Entwurfsmodell. Methoden sowie mögliche Assoziations- und Vererbungsbeziehungen sind anzugeben, Attribute nicht. Stellen Sie das angepasste Entwurfsmodell mit einem UML Klassendiagramm dar.
3. Integrieren Sie das Fachklassenmodell in die gegebene logische 3-Schichtenarchitektur (siehe Seite 3).
4. Überprüfen Sie mit einem UML Sequenzdiagramm Ihr Entwurfsmodell für den Normalablauf des Anwendungsfalls „Kunde erstellen“ hinsichtlich Vollständigkeit und Verantwortlichkeiten der Klassen sowie Interaktionen der entsprechenden Objekte. Ergänzen Sie bei Bedarf das Entwurfsmodell nach dieser Überprüfung.
5. Implementieren Sie für die 1. Iteration den Teil Ihres Entwurfsmodells, der es ermöglicht einen Kunden mit seinem Vor- und Nachnamen zu speichern. Implementieren Sie dazu einen Prototypen in Java.
6. Testen Sie Ihren Prototypen für die Normalabläufe der Anwendungsfälle "Kunde erstellen" und "Kunde anzeigen", indem Sie mehrere Kunden erstellen, speichern, lesen und auf der Konsole deren Vor- und Nachnamen ausgeben.
7. Dokumentieren Sie den Java-Quelltext (nur Dateiköpfe und Methoden) mittels Javadoc und erzeugen Sie die zugehörigen HTML-Dokumentationsseiten.

Bewertungskriterien Aufgabe 4**Bewertung (mittels Kontrolle und/oder Gespräch, max. 36 Punkte):**

#	Bewertungskriterien (Punkte: 0 => nicht erfüllt; 1 => <50% erfüllt; 2 => >=50% erfüllt; 3 => 100% erfüllt)	Punkte			
		0	1	2	3
1	Das UML Klassendiagramm zum Fachklassenmodell ist fachlich korrekt und vollständig.				
2	Die Integrations des Fachklassenmodells in die gegebene logische 3-Schichtenarchitektur ist fachlich korrekt und vollständig.				
3	Das UML Sequenzdiagramm ist fachlich korrekt und vollständig.				
4	Die ins Englische übersetzten Namen der Klassen des Fachklassenmodells sind fachlich korrekt und vollständig ins Glossar eingetragen.				
5	Die Implementation der Persistence-Schicht (DataAccessMock) ist korrekt und vollständig.				
6	Die Implementation der Business-Schicht (Fachklassenmodell) ist korrekt und vollständig.				
7	Die Implementation der Presentation-Schicht (ConsoleClient) ist korrekt und vollständig.				
8	Der Prototyp läuft fehlerfrei (Namen der Kunden werden gespeichert und gelesen).				
9	Die Quellcode-Dokumentation mit Javadoc ist korrekt und vollständig.				
11	Namen bzw. Bezeichnungen in den UML Diagrammen sind aussagekräftig und die bekannten UML Namenskonventionen werden beachtet.				
12	Namen bzw. Bezeichnungen im Java-Quelltext sind aussagekräftig und die bekannten Java Namenskonventionen werden beachtet.				

5.2. 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

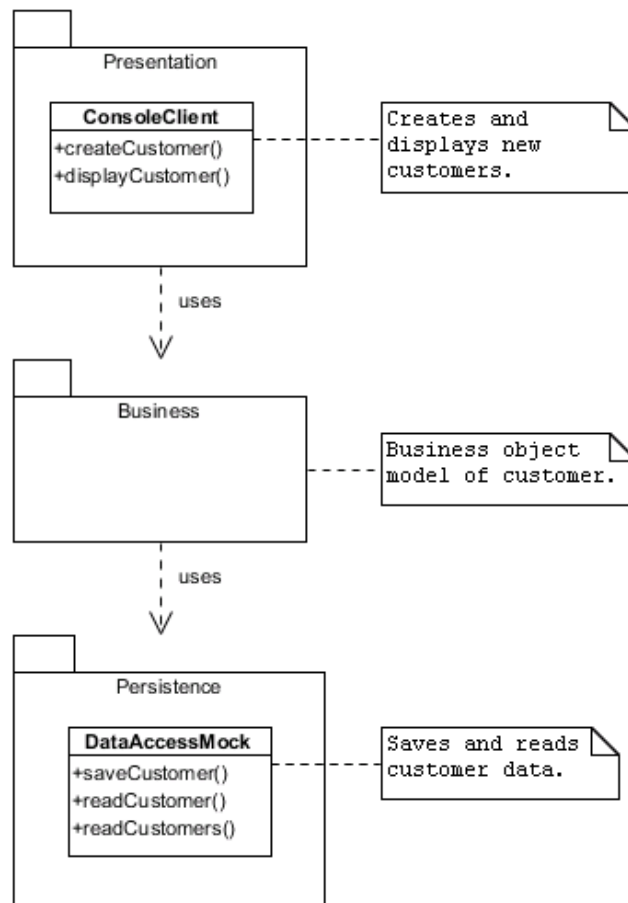
Dokumentenform

- 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

5.3.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.