*Dokumentation*

Java-Projekt

*2021*

Projektname: „Travel-Manager“

Kurs: INF-20ITA

Matrikelnummer: 7314712

Dozent: Dr. Werner Jost

Inhalt

[1 Grobe Spezifikation der gewünschten Leistung 2](#_Toc79283235)

[1.1 Ist-Zustand 2](#_Toc79283236)

[1.2 Soll-Zustand – Anforderungen an das System 3](#_Toc79283237)

[1.3 Entwurf 4](#_Toc79283238)

[1.4 Nutzerschnittstelle (Ein- und Ausgabe) 9](#_Toc79283239)

[1.4.1 Aktivitätsdiagramm 10](#_Toc79283240)

[1.5 Design Pattern 10](#_Toc79283241)

[1.5.1 Observer Pattern 10](#_Toc79283242)

[1.5.2 Komperator 11](#_Toc79283243)

[1.5.3 Delegatios-Pattern 11](#_Toc79283244)

[1.5.4 Factory Pattern 11](#_Toc79283245)

[2 UML Klassendiagramm 12](#_Toc79283246)

# Grobe Spezifikation der gewünschten Leistung

## Ist-Zustand

Ein Anwender möchte eine Reise in ein Land planen. Dieses Land ist eines der Länder, die in der Sammlung an CSV-Dateien des Travel Managers aufgelistet sind (Datei: dateien\kosten\“*Ländername*“.csv). Die verfügbaren Länder sind: Deutschland, norwegen, Spanien, England und Kroatien. Ein Eintrag (eine Zeile) in der Jeweiligen Liste enthält:

* Die zu erwartenden Kosten für eine Unterkunft
* Die zu erwartenden Kosten für Mahlzeiten
* Die zu erwartenden Kosten für den Transport vor Ort
* Die zu erwartenden Kosten für Aktivitäten und Besuche vor Ort im Reiseland
* Die so entstandenen Kosten Pro Person/Pro Tag
* Die so entstandenen Kosten Pro Woche



Abbildung 1: Deutschland.csv

In Abbildung eins sieht man diese Daten am Beispiel von Deutschland. Die Daten sind alle von [diesem Onlineportal](https://www.wohin-und-wann.de/budget/) mit der Einstellung "nur Hotelzimmer" und "eine Person" entnommen.

In der ersten Zeile stehen die Kosten für Reisende mit einem niedrigen Budget. Dies beinhaltet:

* Hotels mit 1 Stern oder Hostels, Günstige Restaurants und im Hotel zubereitete Speisen, Nur kostenlose Aktivitäten, Ausflüge und Besuche

Die zweite Zeile steht für ein mittleres Budget und beinhaltet:

* Nur Hotels mit 2 oder 3 Sternen, überwiegend günstige Restaurants und Fast-Food, ein gutes Restaurant alle 4 Tage, Aktivitäten, Ausgehen und kleine Ausflüge

Die dritte Zeile steht für ein hohes Budget und beinhaltet:

* Nur Hotels mit 3 oder 4 Sternen, Mittag und Abendessen im Restaurant, Aktivitäten, Ausflüge und Besuche jeden Tag

Zudem möchte der Anwender eine Packliste nutzen, die ihm verschiedene zusätzliche Kategorien bietet, sodass man seinen Koffer leichter packen kann.

Die vorgefertigten Packlisten liegen im Ordner dateien\packlisten vor. Ein Eintrag (eine Zeile) in der Jeweiligen Liste enthält

* Die Bezeichnung des Gegenstandes
* Die Anzahl
* Die Kategorie, in die der Anwender den Gegenstand dann selbst einordnen möchte, oder die Überkategorie, in die der Gegenstand passen würde
* Die Jahreszeit in die der Gegenstand passt, wenn es kein Jahreszeit-abhängiger Gegenstand ist steht hier „immer“
* Die Angabe, ob der Gegenstand eher für Männer oder für Frauen gemacht ist, wenn dies irrelevant ist steht hier „alle“

## Soll-Zustand – Anforderungen an das System

Dem Anwender ist die Reiseplanung erleichtert durch:

* Unterstützung bei der Planung des Gepäcks durch das Programm:
* Verwalten von Packlisten
* Bereitstellen von Packlisten
* Unterstützung bei der Planung der Kosten durch das Programm:
* Abschätzen der anfallenden Kosten im Reiseland
* Berechnen der Spritkosten
* Hinzufügen von sonstigen anfallenden Kosten, um einen Überblick über alle vorhersehbareren Ausgaben zu bekommen
* Unterstützung sowohl vor der Reise als auch bei der Reise-Durchführung:
* Umrechnen von Euro-Beträge in Fremdwährungen, wie Kuna, Norwegische Kronen, Pfund Sterling…

## Entwurf

***Wer wird das System benutzen?***

Ein Anwender, der vor hat eine Reise durchzuführen. Es kann nur ein Benutzer eine Reise planen, somit gibt es nur einen Akteur.

***Wer gibt Daten in das System ein?***

* Es gibt schon vorgefertigte Datensätze für die Reiseplanung. Diese sind aus [einem Onlineportal](https://www.wohin-und-wann.de/budget/) entnommen und in eine CSV Tabelle überführt wurden. Diese Datensätze kann der Anwender nicht bearbeiten. Für die Packlisten gibt es ebenfalls bereits vorgefertigte Liste, die Listen können jedoch beliebig erweitert und auch neu erstellt werden, vom Anwender.

***Wer erhält Daten aus dem System?***

* Der Anwender kann über das Menü auswählen, welche Aktion er gerne ausführen möchte und so die Daten aus auslesen und die anfallenden Kosten für seine/ ihre spezifische Reise berechnen. Somit ist er der einzige, der Daten aus dem System erhält.

***Welches Ziel hat der Anwender?***

Sein Ziel ist es, einerseits die Informationen, die er benötigt aus dem Manager zu entnehmen, und andererseits diese Informationen so aufbereitet dargestellt zu bekommen, dass es Ihm somit leichter fällt, einen Überblick über seine Reise und die anfallenden Kosten, bzw. einzupackenden Gegenstände zu erlangen.

***Was ist der Standard-Fall eines Anwendungsfalls?***

Der Reisende möchte eine Packliste einsehen und möglicherweise die anfallenden Kosten für seine Reise bezüglich Spritkosten, Hotelkosten, Transportkosten, Kosten für Aktivitäten vor Ort und Kosten für Mahlzeiten am Urlaubsort berechnen.

***Welche Erweiterungen und Sonderfälle sind bei einem Anwendungsfall***

***möglich?***

Der Reisende möchte die Packliste erweitern, Gegenstände löschen, eine neue Packliste anlegen oder in ein Land reisen, das nicht in der Datenbank zu finden ist

***Besitzen die Anwendungsfälle gemeinsame Funktionalitäten?***

Ja, die Funktionalitäten zum Einsehen und durchsuchen der Packliste werden haben mehrere Anwendungsfälle gemeinsam

|  |  |
| --- | --- |
| Anwendungsfall 1 | Packliste einsehen |
| Ziel | Der Anwender kann die gesuchte Packliste einsehen und nach seinen gewünschten Kriterien sortiert und gefiltert ausgeben lassen. |
| Vorbedingung | CSV-Dateien mit Packlisten befinden sich auf dem lokalen Rechner des Anwenders |
| Nachbedingung Erfolg | Die Packliste enthält alle gewünschten Informationen |
| Nachbedingung Fehler | Das Programm terminiert. |
| Akteure | Anwender, der eine Reise plant |
| Auslösendes Ereignis | Das Programm startet und der Anwender wählt den Punkt Packliste einsehen aus |
| Erweiterungen | Packliste bearbeiten, Packliste sortiert anzeigen, Packliste anlegen |

|  |  |
| --- | --- |
| Anwendungsfall 2 | Packliste bearbeiten |
| Ziel | Der Anwender kann die gesuchte Packliste einsehen und manche Gegenstände löschen oder hinzufügen |
| Vorbedingung | CSV-Dateien mit Packlisten befinden sich auf dem lokalen Rechner des Anwenders |
| Nachbedingung Erfolg | Die Packliste enthält alle gewünschten Informationen |
| Nachbedingung Fehler | Das Programm terminiert. |
| Akteure | Anwender, der eine Reise plant |
| Auslösendes Ereignis | Das Programm startet und der Anwender wählt den Punkt Packliste bearbeiten aus |
| Erweiterungen | - |

|  |  |
| --- | --- |
| Anwendungsfall 3 | Packliste sortiert anzeigen |
| Ziel | Der Anwender kann die gesuchte Packliste nach einem gewünschten Kriterium sortiert einsehen. |
| Vorbedingung | CSV-Dateien mit Packlisten befinden sich auf dem lokalen Rechner des Anwenders |
| Nachbedingung Erfolg | Die Packliste wird nach dem gewünschten Kriterium sortiert auf der Konsole ausgegeben. |
| Nachbedingung Fehler | Das Programm terminiert. |
| Akteure | Anwender, der eine Reise plant |
| Auslösendes Ereignis | Das Programm startet und der Anwender wählt den Punkt Packliste sortiert anzeigen aus |
| Erweiterungen | - |

|  |  |
| --- | --- |
| Anwendungsfall 4 | Packliste anlegen |
| Ziel | Der Anwender legt eine eigene Packliste an, die im CSV Format in das korrekte Verzeichnis gespeichert vorliegt |
| Vorbedingung | Das Verzeichnis ist bereits angelegt |
| Nachbedingung Erfolg | Die Packliste wurde erstellt und gespeichert |
| Nachbedingung Fehler | Das Programm terminiert oder die Packliste liegt nicht vor |
| Akteure | Anwender, der eine Reise plant |
| Auslösendes Ereignis | Das Programm startet und der Anwender wählt den Punkt Packliste erstellen aus |
| Erweiterungen | - |

|  |  |
| --- | --- |
| Anwendungsfall 5 | Die zu erwartenden anfallenden Kosten in einem Reiseland, je nach Budget berechnen. |
| Ziel | Der Anwender weiß, welche Kosten ihn im Reiseland erwarten. |
| Vorbedingung | CSV-Dateien mit Informationen über das Reiseland befinden sich auf dem lokalen Rechner des Anwenders. |
| Nachbedingung Erfolg | Das Programm gibt die Kosten, unter einbeziehung der Personenzahl und der |
| Nachbedingung Fehler | Das Programm terminiert |
| Auslösendes Ereignis | Das Programm startet und der Anwender wählt den Menüpunkt Lebenskosten am Reiseziel berechnen |
| Erweiterung | - |

|  |  |
| --- | --- |
| Anwendungsfall 6 | Sonstige Kosten hinzufügen |
| Ziel | Der Anwender kann einen beliebigen Betrag zu dem aktuellen Kostenstand hinzufügen |
| Vorbedingung | - |
| Nachbedingung Erfolg | Der Betrag wurde korrekt hinzugerechnet |
| Nachbedingung Fehler | Das Programm terminiert |
| Auslösendes Ereignis | Das Programm startet und der Anwender wählt den Menüpunkt Sonstige Kosten hinzufügen |
| Erweiterung | - |

|  |  |
| --- | --- |
| Anwendungsfall 7 | Euro-Beträge in Fremdwährungen umwechseln |
| Ziel | Der eingegebene Betrag soll in die gewünschte Fremdwährung umgerechnet vorliegen. |
| Vorbedingung | Die gewünschte Fremdwährung ist entweder norwegische Kronen, Pfund Sterling, Kroatische Kuna oder US-Dollar sein. |
| Nachbedingung Erfolg | Der korrekt umgerechnete Betrag liegt vor |
| Nachbedingung Fehler | Das Programm Terminiert |
| Akteure | Der Anwender |
| Auslösendes Ereignis | „Währungen umwandeln" wurde ausgewälzt |
| Erweiterung |  |

|  |  |
| --- | --- |
| Anwendungsfall 8 | Fahrtkosten berechnen |
| Ziel | Der Anwender weiß, wieviel er für Sprit ausgeben wird. |
| Vorbedingung | Dem Anwender ist bewusst, wieviel sein Auto verbraucht und wie viele Km er zurücklegen wird |
| Nachbedingung Erfolg | Die Fahrtkosten wurden korrekt ausgerechnet und zum aktuellen Kostenstand hinzugefügt |
| Nachbedingung Fehler | Das Programm terminiert |
| Auslösendes Ereignis | Das Programm startet und der Anwender wählt den Menüpunkt Fahrtkosten berechnen |
| Erweiterung | - |

|  |  |
| --- | --- |
| Anwendungsfall 9 | Reisekostenstand anzeigen |
| Ziel | Der Anwender weiß, wie sein aktueller Reisekostenstand ist |
| Vorbedingung | - |
| Nachbedingung Erfolg | Der Reisekostenstand wird korrekt auf der Konsole ausgegeben |
| Nachbedingung Fehler | Das Programm terminiert |
| Auslösendes Ereignis | Das Programm startet und der Anwender wählt den Menüpunkt Reisekostenstand anzeigen |
| Erweiterung | - |

## Nutzerschnittstelle (Ein- und Ausgabe)

Für das Nutzerkonzept wurde ein Use-Case- Diagramm erstellt.

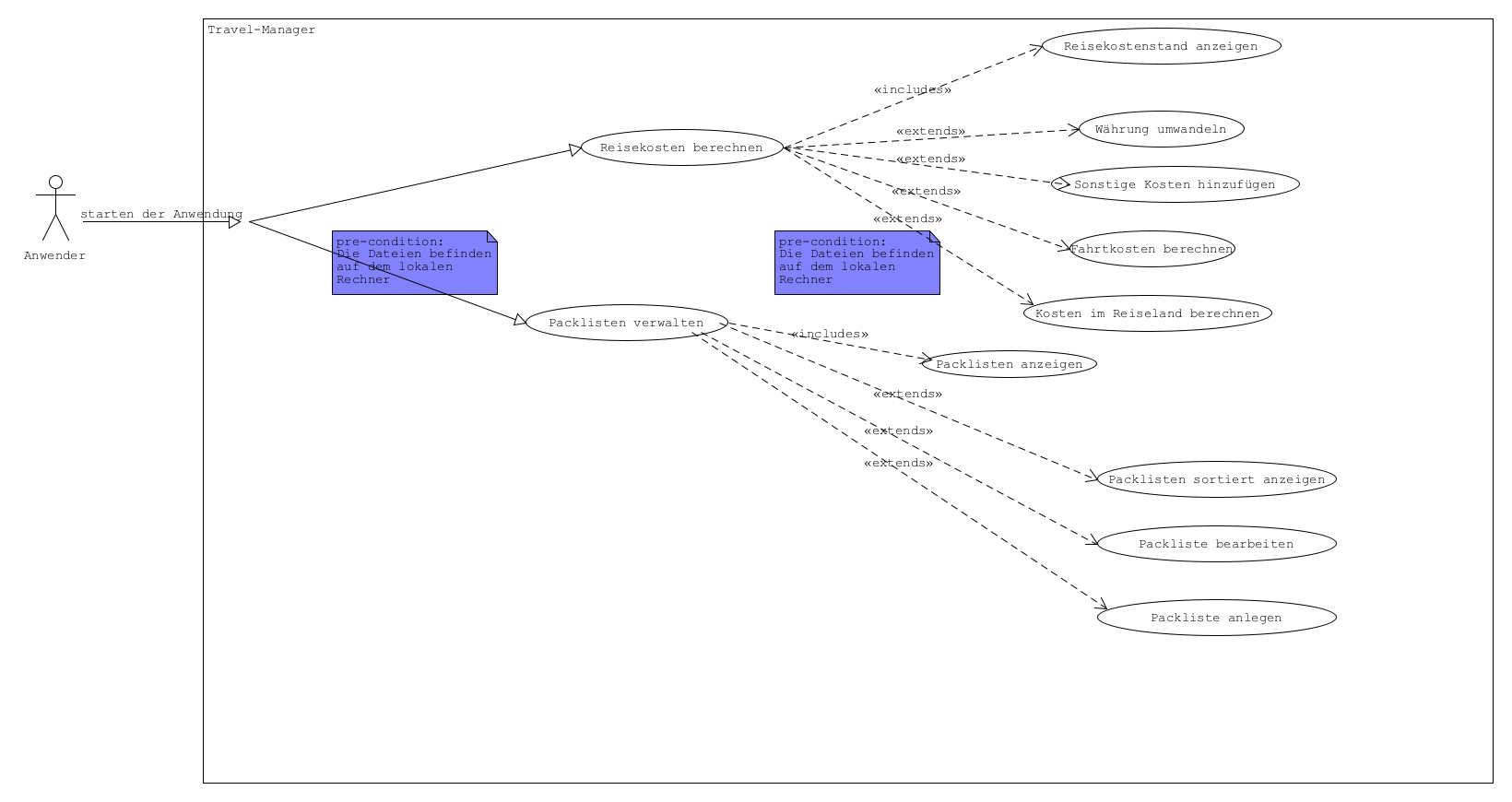


Abbildung 2: Use-Case Diagramm

### Aktivitätsdiagramm

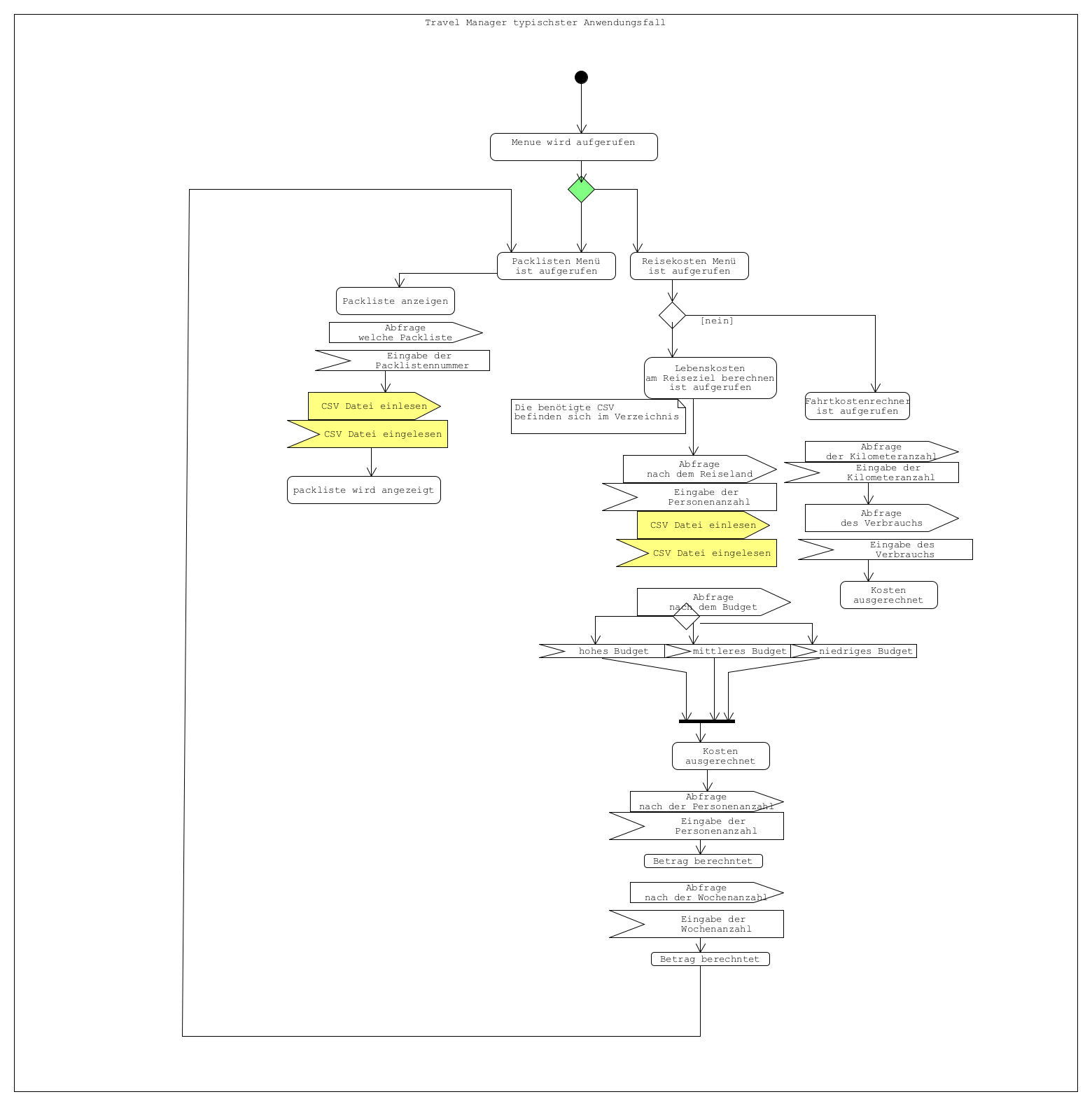


Abbildung 3: Aktivitätsdiagramm des typischsten Anwendungsfalles

## Design Pattern

Im Folgenden soll noch kurz auf einige Implementierungsdetails eingegangen werden. Insbesondere soll kurz die Verwendung von Design Pattern erläutert werden.

### Observer Pattern

Um die gesamtkosten im Auge zu Behalten wird das in der Vorlesung vorgestellte Observer -Pattern verwendet. Im Code wird die „Kostenanzeige“ als Beobachter des kostenstands eingesetzt, die den aktuellen Kostenstand bei einer Veränderung auf der Konsole ausgibt

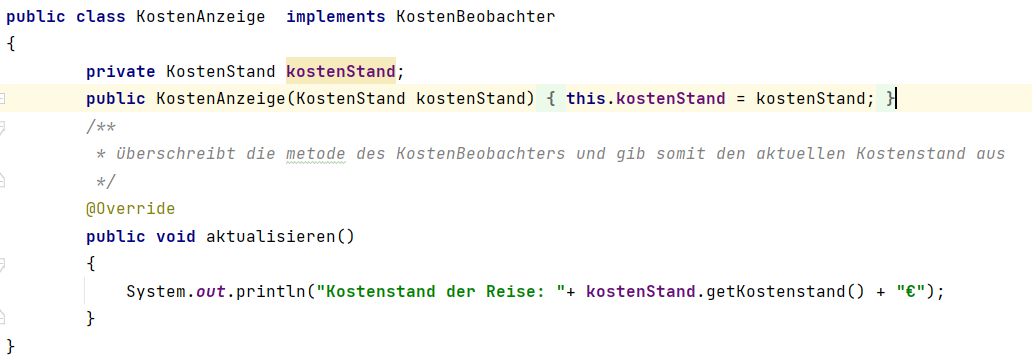


Abbildung 7: Konkrete Implementierung des KostenBeobachter Interfaces

Die Klasse Kosten Subjekt ist zusätzlich implementiert, damit das Beobachter-pattern auf beliebig viele Beobachter ausgeweitet werden kann, die in einem Set gespeichert werden.

### Komperator

Zum Sortieren der Packlisten-Elemente wurde der Komperator verwendet, und somit konten leicht neue, individuelle Komparatoren hinzugefügt werden.

### Delegatios-Pattern

Für den Währungsrechner wurde das Delegation Pattern verwendet. Dadurch kann leicht zwischen den unterschiedlichen Währungsrechner gewechselt werden.

Dies geschieht dann durch:

rechner.setAktuellerRechner(**new** NorwegischeKronenRechner());

**double** erg = rechner.recheneUm(23);

(beispielsweise)

### Factory Pattern

Die Berechnung des Kostenstandes wurde an das in der Vorlesung besprochene Factory-Pattern angelehnt.

# UML Klassendiagramm

Ein Bild, das Tisch enthält.

Automatisch generierte Beschreibung

Abbildung 8 UML Klassendiagramm

In Abbildung 8 wird das UML Klassendiagramm zum Travel-Manager vorgestellt. Die einzelnen Klassen werden mit ihrem Namen, ihre Klassen Attributen und ihren Methoden dargestellt. Die Klassen Attribute sind – wenn möglich – privat, um die Kapselung zu gewährleisten. Aus dem Klassendiagramm lassen sich die Assoziationen ablesen.