Entwicklung einer Carsharing-Anwendung

Programmentwurf 2022

im Rahmen der Prüfung zum

Bachelor of Science (B.Sc.)

des Studienganges Informatik

an der Dualen Hochschule Baden-Württemberg Karlsruhe

von

Clemens Richter, Johannes Peters

Abgabedatum:	04. April 2022
Bearbeitungszeitraum:	27.12.2021 - 01.04.2022
Matrikelnummer, Kurs:	5802185, TINF20B1
Ausbildungsfirma:	SAP SE Dietmar-Hopp-Allee 16 69190 Walldorf, Deutschland
Gutachter der Dualen Hochschule:	Prof. Dr. Richard Lutz

Inhaltsverzeichnis

Abkürzungsverzeichnis **II**

Abbildungsverzeichnis **III**

Quellcodeverzeichnis **IV**

1 Einleitung **1**

2 Lastenheft **2**
2.1 Zielsetzung 2
2.2 Anwendungsbereiche 2
2.3 Zielgruppen, Benutzerrollen und verantworklichkeiten 2
2.4 Zusammenspiel mit anderen Systemen 3
2.5 Produktfunktionen 4
2.6 Produktdaten 5
2.7 Produktleistungen 6
2.8 Qualitätsanforderungen 6

3 Aufgaben **7**
3.1 Analyse 7
3.2 Sequenzdiagramm und Aktivitätsdiagramm 7
3.3 Entwurf 8
3.4 Implementierung 9

4 Vereinfachung für den Programmentwurf **11**

5 Analyse **12**
5.1 Einleitung 12
5.2 Lastenheft 13

Abkürzungsverzeichnis

Abbildungsverzeichnis

Quellcodeverzeichnis

1 Einleitung

Für unsere Carsharing-Organisation *Citycar*BaDö* benötigen wir ein neues Buchungssystem, um dem wachsenden Bedarf an gemeinsam genutzten Fahrzeugen gerecht werden zu können.

*Citycar*BaDö* hat inzwischen fast 500 Mitglieder, denen eine Fahrzeugflotte von ca. 60 Fahrzeugen an ungefähr 30 Standorten in und um Bad Dödelhausen zur Verfügung steht.

Bisher werden die Mitgliedschaften und Vermietungen mit einem inzwischen in die Jahre gekommenen Buchungssystem verwaltet, das sich recht umständlich bedienen lässt. Zwar können Buchungen online erfolgen, aber wir würden gerne zusätzliche Informationen für die Online-Kunden zur Verfügung stellen und eine Erweiterung der alten Software lohnt sich nicht.

2 Lastenheft

2.1 Zielsetzung

Ziel des Entwicklungsauftrags ist eine Software für die Verwaltung aller Daten, die für die Verwaltung der Fahrzeuge, Kunden sowie Buchungen unserer Carsharing-Organisation anfallen und benötigt werden.

Alle Daten sollen zentral gespeichert werden, da mehrere Benutzer gleichzeitig auf die Daten und Termine zugreifen werden.

Ein selektiver Import und Export von Daten über lesbare Dateien muss für Backups und zum Datenaustausch möglich sein.

Eine intuitive, leicht bedienbare Benutzeroberfläche setzen wir als selbstverständlich voraus. Es sollen keine besonderen Computerkenntnisse zur Bedienung der Software erforderlich sein.

2.2 Anwendungsbereiche

Die Software soll ausschließlich für die Verwaltung von Fahrzeugen, Kunden, Ausrüstung, Fahrzeugstandorte und Angestellten und den damit direkt verbundenen Elementen verwendet werden. Sie soll im Alltag auf Desktop-Rechnern und Laptops eingesetzt werden.

2.3 Zielgruppen, Benutzerrollen und verantworklichkeiten

Es soll verschiedene Benutzerrollen geben:

- Organisatorinnen und Organisatoren pflegen die jeweiligen Buchungsdaten und Fahrzeuge.

- Personalmitarbeiter pflegen Mitarbeiterdaten im System
- Eine hauptverantwortliche Person (Administrator) hat Vollzugriff auf sämtliche Daten, vor allem für deren Import und Export sowie deren Backup.
- Es gibt keine Gruppen oder Abteilungen, die verwaltet werden müssen.

2.4 Zusammenspiel mit anderen Systemen

Die Daten über die Angestellten (Gehälter bzw. Löhne, Steuern, Kranken- und Rentenversicherung usw.) werden separat durch ein vorhandenes Personalbuchhaltungsprogramm verwaltet und müssen hier nicht berücksichtigt werden. Die finanztechnischen Daten werden über unser vorhandenes Finanzsystem erfasst und müssen hier ebenfalls nicht berücksichtigt werden.

Die Software soll aus zwei Teilen bestehen:

- Für die Mitarbeiter im Büro soll eine Desktop-Anwendung erstellt werden, mit denen die Datenbestände verwaltet werden können. Es sollen auch Buchungen erstellt werden können für Kunden, die keine Online-Buchungen machen wollen und persönlich in der Carsharing-Filiale erscheinen.
- Eine neue Web-Seite soll unseren Online-Kunden ermöglichen, nach einer Authentifizierung alle Standorte anzeigen zu lassen sowie natürlich die dort befindlichen Fahrzeuge, welche für einen anzugebenden Zeitbereich online gebucht werden können.

Die Web-Seite soll mit dem ersten Teilauftrag noch nicht programmiert werden, allerdings benötigen wir ein klares Konzept, wie diese Web-Seite realisiert werden soll (Schnittstellen usw.).

Möglichst alle Daten sollen vom alten in das neue System übertragen werden.

2.5 Produktfunktionen

/LF10/	Der jeweilige Benutzer muss die Möglichkeit haben, über eine grafische Benutzeroberfläche alle für ihn relevanten Daten einfach und übersichtlich zu verwalten. Es sollen zahlreiche Konfigurationsdaten gespeichert und beim nächsten Start des Programms verwendet werden (z.B. aktuelle Größe und Position des Fensters). Daneben sollen einige Elemente vor dem Start konfigurierbar sein (z.B. Überschriften, Schriftarten und -größen usw.).
/LF20/	Verwaltet werden sollen Mitarbeiter, Fahrzeuge, Standorte, Kunden, Buchungen, Rechnungen, Änderungen, Stornierungen, Mahnungen usw. Es muss möglich sein, jederzeit erkennen zu können, welche angestellte Person einen Datensatz angelegt, geändert oder gelöscht hat.
/LF30/	Buchungen haben eine Start- und einen Endtermin, Terminüberschneidungen müssen vermieden werden, um die Verfügbarkeit sicherzustellen.
/LF40/	Unsere Kunden haben neben ihren Kontaktdaten auch Vertragsunterlagen für die Teilnahme am Carsharing, die die Höhe des Eigenanteils für einen Schadensfall der Versicherung sowie die Höhe der Teilnahme-Kaution enthält. Diese Vertragsunterlagen werden von uns eingescannt und sollen als Dokument mit den Kundendaten gespeichert werden. Daneben wird jedem Kunden eine Karte zum Öffnen und Schließen der Fahrzeuge ausgehändigt. Ein kleiner Prozessor im Fahrzeug sendet nach Fahrtende (Terminende) die exakte Start- und Ende-Zeit sowie die gefahrenen Kilometer an einen Server. Diese Daten sollen von dem Server nach Buchungsende geholt und zur Berechnung der Kosten für die Buchung (und somit für die Rechnung) verwendet werden.

/LF50/	Die Fahrzeuge selbst gehören unterschiedlichen Kategorien an: Kleinfahrzeuge, Mittelklassefahrzeuge, gehobene Mittelklasse und Transportfahrzeuge. Allen Kategorien sind eine bestimmte Höhe der Stunden-Mietpauschale und die Kosten pro gefahrenem km zugeordnet. Alle Werte sollen konfigurierbar sein. Alle Fahrzeuge werden regelmäßig von Fremdfirmen gewartet. Die entsprechenden Dienstleistungen sollen den Fahrzeugen chronologisch zugeordnet werden.
/LF60/	Einem Standort können ein oder mehrere Fahrzeuge zugeordnet sein. Ein Fahrzeug ist immer nur einem Standort zugeordnet.
/LF70/	Nach jeder Fahrt werden sofort die Rechnungen erstellt und dem Kunden per E-Mail zugesandt. Die Bezahlung der Rechnungen erfolgt über Bankeinzug, was durch das Finanzbuchhaltungssystem (FBH) erledigt wird und hier nicht betrachtet werden muss. Allerdings muss über die vorhandene Schnittstelle des FBH der Stand der Rechnungsbegleichung abgefragt werden, damit über das neue System erkennbar ist, ob und wann eine Rechnung bezahlt wurde.
/LF80/	Buchungen können bis 10 Stunden vor Antritt der Fahrt storniert werden.
/LF90/	Zur einfacheren Eingabe der Daten soll es Auswahllisten für deren Eigenschaften geben, wo immer es möglich ist. Die Auswahllisten sollen auf einfache Weise erweiterbar sein.
/LF100/	Sämtlichen Elementen sollen mehrere Bilder mit Titel zugeordnet werden können, die zentral auf einem Verzeichnis liegen sollen

2.6 Produktdaten

/LD10/	Die Daten sollen zunächst in einer zentralen Datenbasis (lesbare Dateien) abgespeichert und später in eine Datenbank überführt werden.

2.7 Produktleistungen

/LL10/	Die Anzahl der zu verwaltenden Elemente wird auf ca. 100.000 geschätzt.
/LL20/	Um bei HW- und SW-Anschaffungen und -neuerungen flexibel zu bleiben, ist auf Plattformunabhängigkeit besonders zu achten.

2.8 Qualitätsanforderungen

Produktqualität	sehr gut	gut	normal	nicht relevant
Funktionalität	X			
Zuverlässigkeit		X		
Effizienz		X		
Benutzbarkeit (auch Gestaltung)	X			
Wartbarkeit			X	
Übertragbarkeit (Portabilität)			X	

3 Aufgaben

Es handelt sich hier um eine vereinfachte Verwaltungs-Software. Einzelne Lastenheftpunkte sind bewusst offengehalten. Denken Sie darüber nach, welche Informationen zusätzlich sinnvoll oder auch notwendig sind. Recherchieren Sie evtl. nach einzelnen Zusammenhängen im Internet.

3.1 Analyse

Für die Analyse sind zu erstellen:

- Analyse des Lastenhefts (Fragen und Antworten).
- Ein Use-Case-Diagramm der gesamten Anwendung incl. Beschreibung.
- Eine Verfeinerung des Use-Case-Diagramms incl. Beschreibung. (nach Absprache)
- Ein Analyse-Klassendiagramm incl. Beschreibung (Untersuchen Sie dabei den Einsatz geeigneter Analysemuster)
- Einfache GUI-Skizzen (Mockups) von mindestens zwei wesentlichen GUI-Komponenten (Hauptseite, Tabs, etc.). Die Skizzen können mit einem einfachen Grafikprogramm erstellt werden. Auch sorgfältige Handzeichnungen sind erlaubt. Keine Login-GUI skizzieren!

3.2 Sequenzdiagramm und Aktivitätsdiagramm

Erstellen Sie ein Sequenzdiagramm und ein Aktivitätsdiagramm (incl. Beschreibung) für folgende Szenarios (ein AD für das eine Szenario, ein SD für das andere Szenario):

- Die Aktion „Standort mit neuen Fahrzeugen anlegen“ durchführen. Ausgehend von einem neuen Standort und **leerer Datenbasis** werden dessen gesamte Daten

erfasst und in das System eingetragen. (dies wird als Gebrauchsanweisung für die Evaluation Ihrer Implementierung dienen)

- Die Aktionen „Buchung eines Fahrzeugs“ durchführen. Hierbei soll eine komplette Buchung inklusive Beendigung der Fahrt und bezahlen der Rechnung modelliert werden.

Die Bewertung Ihrer Diagramme erfolgt auf der Basis der Nutzung der UML-Elemente, auf Ihrer Kreativität sowie dem Detaillierungsgrad des jeweiligen Diagramms.

Fassen Sie bei beiden Diagrammen die Eingabe aller primitiven Attribute eines Elements (Float, String, Integer, ...) in einer einzigen Aktion zusammen (z.B. „Attribute eintragen“).

Für das Sequenzdiagramm ist das gewählte Szenario ausführlich zu entwickeln (idealerweise mit Pseudocode oder einer anderen Modellierungsmethode Ihrer Wahl). Es sind sämtliche referenzierten Elemente zu berücksichtigen, die zugeordnet werden können.

In allen Fällen wird eine (noch) leere Datenbasis angenommen. Denken Sie an geeignete Diagrammverfeinerungen.

3.3 Entwurf

Abzuliefern sind hier (alle Diagramme und GUIs jeweils mit Beschreibung):

- Entwurfsklassendiagramm (Untersuchen Sie dabei den Einsatz geeigneter Entwurfsmuster)
- GUI-Modellierung: Es ist das Kommunikationsschema eines Teils der während der Analyse skizzierten GUI mit **UML** zu modellieren. Die Anwendung selbst soll dabei nach dem einfachen Model-View-Control-Muster aufgebaut sein. Dazu sind mindestens ein Controller, die erforderlichen Modellklassen sowie eine unabhängige GUI (View) erforderlich.
- Die meisten GUI-Elemente werden über eine einfache kleine Java-Bibliothek zur Verfügung gestellt (swe-utils.jar), deren GUI-Komponenten in das Klassendiagramm zu integrieren sind, wenn sie verwendet werden.

- Die GUI-Modellierung kann in einem separaten Diagramm mit den relevanten (gewählten bzw. benötigten) Modellklassen erfolgen, falls das Entwurfsklassendiagramm sonst zu komplex werden würde.

3.4 Implementierung

Es ist eine einfache Java-Applikation zu implementieren, die es ermöglicht, Carsharing-Daten anzulegen, zu ändern und zu löschen.

Zur Realisierung wird die oben bei der Entwurfsaufgabe erwähnte Java-Bibliothek zur Verfügung gestellt (*swe-utils.jar*), die neben mehreren GUI-Komponenten einen *CSVReader*, einen *CSVWriter* sowie mehrere Interfaces bereitstellt (in den Packages *event* und *model*).

Daneben ist eine Mini-Test-Applikation gegeben, die die Funktionsfähigkeit der GUI-Komponenten demonstriert (Start mit *java -jar swe-utils.jar*). Details sind der Java-Dokumentation der Bibliothek zu entnehmen.

Zur leichteren und zukunftssicheren Evaluation Ihres Programmentwurfs soll die Java-Applikation als eine Desktop-Applikation mit CSV-Dateien (alternativ XML oder JSON) als zentrale Datenbasis realisiert werden, die von beliebigen Rechnern aus gestartet wird. Dabei sind mehrere Dateien analog zu Datenbanktabellen zu erzeugen.

Einzelne Aufgaben

- Hauptaufgabe ist die Realisierung einer MVC-Applikation mithilfe des Observer-Patterns entsprechend des vorgegebenen GUI-Entwurfs und der gegebenen Java-Bibliothek.
- Die Erzeugung der Instanzen soll in einer Entity-Factory erfolgen und zur Verwaltung der Instanzen ist ein Entity-Manager zu realisieren (beides siehe Vorlesung).
- Beim Anlegen einer Buchung muss für die Zuordnung von Hilfsmitteln sichergestellt sein, dass es keine zeitlichen Überschneidungen gibt (LF30+LF40).
- Es muss eine ausführbare JAR-Datei abgegeben werden, die mit

  „java -jar SWE-PE-2022_Carsharing_<name1>_<name2>.jar OPTIONEN“

gestartet werden kann. Hierfür ist ein BASH-Skript namens *startApp* zu erstellen

- Geprüft wird das Anlegen einer Buchung mit der Zuordnung aller zugehörigen Elemente. Nach dem Anlegen wird die Applikation erneut gestartet und geprüft, ob alle Daten korrekt abgespeichert und beim Laden wieder zugeordnet werden.

Verwendung von CSV-Dateien

- Die Daten sollen in CSV-Dateien vorliegen und können mittels den gegebenen Bibliotheksklassen *CSVReader* und *CSVWriter* gelesen bzw. beschrieben werden. Zur Vereinfachung können die Daten jeweils komplett geschrieben werden.
- Abgegeben werden soll ein ZIP-File (oder TAR-File) mit allen Java- und CSV-Dateien (letztere gesammelt in einem eigenen Verzeichnis):

  „SWE-PE-2022_Carsharing_<n1>_<n2>.zip (tar oder tar.z)
- Als OPTIONEN in der Startanweisung soll der Pfad zu den CSV-Dateien sowie zu einer Properties-Datei angegeben werden können:

  „java -jar SWE-PE-2022_Carsharing_<n1>_<n2>.jar **–d <csvpath> –p <prop-file>**"

4 Vereinfachung für den Programmentwurf

1. Es muss nicht dafür gesorgt werden, dass auf dieselben Daten bzw. CSV-Dateien nicht gleichzeitig zugegriffen werden kann, d.h. es ist kein *Locking*-Mechanismus erforderlich.
2. Eine Protokollierfunktion und ein Login-Vorgang sind für die Anwendung nicht erforderlich (in der Realität natürlich schon!).
3. Zeitliche Überschneidungen sind natürlich bei allen Buchungen möglich und müssten sowohl beim Anlegen als auch bei Änderungen von Terminen berücksichtigt werden. Im Programmentwurf sollte dies in der Modellierung berücksichtigt werden, bei der Implementierung ist jedoch nur eine Überprüfung bei der Auswahl des Starts und Endes der Buchung erforderlich.
4. Konfigurationsdaten (LF 10) sollen exemplarisch für wenige Elemente änderbar sein (Angabe der realisierten Elemente!)
5. Alle Elemente, die zu einer Buchung zugeordnet werden können, müssen nicht interaktiv erzeugbar, sondern können bereits in CSV-Dateien vorhanden sein. Verwenden Sie dabei realistische Attributwerte!

5 Analyse

Text in grün steht für die Fragen der Analyse.

Text in blau steht für die Antworten der Analyse.

5.1 Einleitung

Für unsere Carsharing-Organisation *Citycar*BaDö* benötigen wir ein neues Buchungssystem, um dem wachsenden Bedarf an gemeinsam genutzten Fahrzeugen gerecht werden zu können.

*Citycar*BaDö* hat inzwischen fast 500 Mitglieder, denen eine Fahrzeugflotte von ca. 60 Fahrzeugen an ungefähr 30 Standorten in und um Bad Dödelhausen zur Verfügung steht.

Was für ein Wachstum an Standorten, Mitgliedern und Fahrzeugen ist in den kommenden Jahren zu erwarten?

Was für Fahrzeugtypen? PKW, LKW, Elektroauto, Scooter?

Sollen Abrechnungen über das selbe System erfolgen? Abrechnung Kunde, Abrechnung Finanzamt?

Existiert ein Buchhaltungssystem?

Bisher werden die Mitgliedschaften und Vermietungen mit einem inzwischen in die Jahre gekommenen Buchungssystem verwaltet, das sich recht umständlich bedienen lässt.

Welches Buchungssystem haben Sie verwendet?

Was war daran so umständlich?

Was macht ein Mitglied aus, welche Eigenschaften hat es?

Zwar können Buchungen online erfolgen, aber wir würden gerne zusätzliche Informationen für die Online-Kunden zur Verfügung stellen und eine Erweiterung der alten Software lohnt sich nicht.

In welchem Kostenrahmen hätte sich diese Erweiterung bewegt, welcher Kostenrahmen soll dieses Projekt erhalten?

Was für zusätzliche funktionen konnte das alte Programm speziell nicht erfüllen?

5.2 Lastenheft

5.2.1 Zielsetzung