HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

INFORMATIK

cnlab

EcoHelper für das iPhone

[Bachelorarbeit]

Abteilung Informatik
Hochschule für Technik Rapperswil

Frühjahrssemester 2010,
22.02.2010 – 18.06.2010

Autoren:	Selim Akyol, Edon Berisha
Betreuer:	Prof. Dr. Peter Heinzmann
Projektpartner:	cnlab AG, Rapperswil
Experte:	Dr. Thomas Siegenthaler
Gegenleser:	Dr. Lothar Müller


Eco
Helper
V1.0

0. Inhalt

1. Administratives

1.1. Vereinbarung[1]

1.1.1. Gegenstand der Vereinbarung

Mit dieser Vereinbarung werden die Rechte über die Verwendung und die Weiterentwicklung der Ergebnisse der *Bachelorarbeit „EcoHelper für das iPhone*“ von *Selim Akyol* und *Edon Berisha* unter der Betreuung von *Prof. Dr. Peter Heinzmann* (für die Arbeit verantwortlicher Professor) geregelt.

1.1.2. Urheberrecht

Die Urheberrechte stehen der Studentin / dem Studenten zu.

1.1.3. Verwendung

Die Ergebnisse der Arbeit dürfen sowohl von allen an der Arbeit beteiligten Parteien, d.h. von den Studenten, welche die Arbeit verfasst haben, vom verantwortlichen Professor sowie vom Industriepartner verwendet und weiter entwickelt werden. Die Namensnennung der beteiligten Parteien ist bei der Weiterverwendung erwünscht, aber nicht Pflicht.

Rapperswil, den..... .....
Student

Rapperswil, den..... .....
Student

Rapperswil, den..... .....
Verantwortlicher Professor

Rapperswil, den..... .....
Industriepartner

[1] Diese Vereinbarung basiert auf den Muster-Vereinbarungen in den *Richtlinien der HSR zur Durchführung von Projekt-, Studien-, Diplom- oder Bachelorarbeiten* vom 16. Februar 2009.

1.2. Erklärung der Verfasser[2]

Die vorliegende Arbeit basiert auf Ideen, Arbeitsleistungen, Hilfestellungen und Beiträgen gemäss folgender Aufstellung:

Gegenstand, Leistung	Person	Funktion
-Anforderungsanalyse -GUI -Datenbank -Server-Kommunikation -OBD2	*Selim Akyol*	Autor der Arbeit
-Fahrdatenanalyse -Domainanalyse -Softwarearchitektur -OBD-Kommunikation (PLXDevice) -GUI	*Edon Berisha*	Autor der Arbeit
Korrekturlesen, Hinweise zur sprachlichen Überarbeitung	*Florence Akyol* *Laura Conoscenti*	Korrekturlesen
Idee, Aufgabenstellung, allgemeines Pflichtenheft, Betreuung während der Arbeit	*Prof. Dr. Peter Heinzmann*	Verantwortlicher Professor
Betreuung während der Arbeit, Spezifikation der Anwendung, Hilfestellung OBD	*Omid Afshari*	Assistent, HSR
Betreuung während der Arbeit, Hilfestellung bei der iPhone SW-Entwicklung; bestehende EcoHelper-Anwendungen, Web-Kopplung	*Raphael Juchli,* *René Vogt*	Industriepartner, cnlab AG

[2] Diese Erklärung basiert auf der Muster-Erklärung in den *Richtlinien der HSR zur Durchführung von Projekt-, Studien-, Diplom- oder Bachelorarbeiten* vom 16. Februar 2009.

Ich erkläre hiermit,

- dass ich die vorliegende Arbeit gemäss obiger Zusammenstellung selber und ohne weitere fremde Hilfe durchgeführt habe.
- dass ich sämtliche verwendeten Quellen erwähnt und gemäss gängigen wissenschaftlichen Zitierregeln korrekt angegeben habe.

Rapperswil, den..... .....
Student

Rapperswil, den..... .....
Student

1.3. Aufgabenstellung

1.3.1. EcoHelper – iPhone als persönlicher Benzinspar-Trainer

Studiengang: Informatik (I)

Semester: Frühjahrssemester 2010 (22.02.2010-18.06.2010)

Durchführung: Bachelorarbeit

Fachrichtung: Internet-Technologien und -Anwendungen

Institut: ITA: Internet-Technologien und Anwendungen

Studierende: Selim Akyol, Edon Berisha

Betreuer:	Dr. Prof. Peter Heinzmann
Gegenleser:	Dr. Lothar Müller
Experte:	Dr. Thomas Siegenthaler, CSI Consulting AG, Zürich
Industriepartner:	cnlab AG, Rapperswil

Ausschreibung:

EcoHelper hilft, eine umweltfreundliche und kostengünstige Fahrweise zu trainieren. Über den Onboard-Diagnose-Bus (OBD2) moderner Autos kann die EcoHelper-Handyapplikation Fahrdaten (z.B. Motordrehzahl, Gaspedalstellung und Benzinverbrauch) erfassen. Diese Daten werden mit GPS-Positionsdaten und Bilddaten des Handys ergänzt und auf eine Web-Plattform übermittelt. Es existieren bereits EcoHelper-Applikationen für Symbian- und für JavaME- fähige Mobiltelefone. Ziel dieser Arbeit ist es, eine EcoHelper-Applikation für das iPhone anbieten zu können. Die Arbeit umfasst folgende Teilaufgaben:

- Einarbeitung in Objective-C
- Einarbeitung in EcoHelper-Konzepte
- Ausarbeitung von Möglichkeiten zur Nutzung von Beschleunigungsdaten
- Realisierung der EcoHelper App gemäss Spezifikation
- Testfahrten

Die zu realisierende iPhone-EcoHelper-App muss folgende Funktionen unterstützen:

- Ansprechen des WLAN-OBD2-Adapters
- Auslesen der GPS-Koordinaten
- Auslesen von Beschleunigungsdaten
- Automatische Bildaufnahme
- Übermitteln der Daten zur Web-Plattform (Live-Mode)
- Übermitteln aufgenommener Daten, sobald ein WLAN verfügbar ist (Offline-Mode)

	Weitere Funktionen könnten sein: •Graphische Darstellung à la Rev [URL 12] •Einstellungsmöglichkeiten zu obigen Funktionen, wie Intervalle, Ziel-URL, etc. • Unterstützung für mehrere Fahrzeuge pro iPhone Audio-visuelle Hinweise zu den EcoDrive Regeln: o Schaltzeitpunkt o Pedalstellung o Beschleunigung Referenzen: • Foliensatz zu EcoHelper: [URL 13] • EcoHelper Web-Anwendung: [URL 8] • Mobile Datalogger Anwendung: [URL 14] • cnlab Speedtest iPhone App:[URL 15]
Voraussetzungen:	Flair für Physik, Vorkenntnisse Objective-C und Zugang zu eigenem Testfahrzeug (Auto mit OBD2-Interface) von Vorteil aber nicht Bedingung

1.3.2. EcoHelper – Grobspezifikation

1.3.2.1. Übersicht

Der EcoHelper besteht aus einer iPhone-Applikation und aus einer Webanwendung. Ziel der Bachelorarbeit ist es, die iPhone Anwendung zu realisieren und die erfassten Daten auch an die (bestehende bzw. erweiterte) Webanwendung zu liefern.
Die primäre Aufgabe der iPhone-Anwendung ist die Aufnahme der Fahrzeug-, GPS- und Beschleunigungsdaten und deren Übermittlung zur Webanwendung.
Des Weiteren soll EcoHelper als WAB2-Trainingsgerät zur Erlernung einer umweltfreundlichen Fahrweise verwendet werden. Dazu müssen diverse Fahrparameter für alle Fahrzeuginsassen sichtbar angezeigt werden. Anschliessend an die Lernfahrt soll der Instruktor anhand der statistischen Auswertung eine Aussage über das Fahrverhalten des Schülers machen können.

1.3.2.2. Datenerfassung

Es müssen Daten von verschiedenen Quellen erfasst werden:

- OBD2 Daten über WLAN (VIN, Geschwindigkeit, Drehzahl, MAF, etc.)
- Internes GPS (Breitengrad, Längengrad, Höhe, Richtung, Geschwindigkeit)

1.3.2.3. Datenspeicherung

- Offline-Mode:
  Die erfassten Daten müssen lokal gespeichert werden können. Auf Wunsch sollen die Daten bei der Verfügbarkeit eines WLAN zum zentralen Server hochgeladen werden.
- Online-Mode:
  Die erfassten Daten werden lokal gepuffert und über das Mobilfunknetz zum zentralen Server geschickt.

1.3.2.4. GUI-Livemodus

Es soll ein einfaches, gut lesbares GUI realisiert werden. Für die WAB2-Kurse ist die Anzeige des Durchschnittsverbrauchs und des Momentanverbrauchs vorgeschrieben.
Mögliche Parameter sind:

- Durchschnittsverbrauch (zwingend für WAB2-Gerätezulassung)
- Momentanverbrauch (zwingend für WAB2-Gerätezulassung)
- Eingelegter Gang
- Schubabschaltung
- Gaspedalstellung
- Geschwindigkeit
- Drehzahl

1.3.2.5. GUI-Auswertungsmodus

Zur Unterstützung des Instruktors bei Aussagen soll der EcoScore berechnet und verschiedene Grafiken (Zeitreihenanalysen, Summenhäufigkeiten) erstellt werden. Ausserdem sollten triviale Aussagen bereits gemacht werden. Zum Beispiel:

- Sie fahren zu hochtourig (20% über der Drehzahl 2300 RPM, 5% wäre gut)
- Sie nutzen die Schubabschaltung zu wenig (nur 1% nicht auf dem Gaspedal, 10% wäre gut)
- Sie fahren zu wenig vorausschauend und müssen daher zu häufig bremsen (Ihr Bremsfaktor x, y wäre gut) Bremsfaktor = Stärke und Häufigkeit der Bremsvorgänge
- Sie geben zu wenig zügig Gas, Sie beschleunigen zu langsam (beim Beschleunigen ist Gaspedal nur 50% gedrückt, 75% wäre richtig)
- Schalten Sie früher (90% Drehzahl zu hoch)

Es sollen folgende Parameter ausgewertet werden:

- aktueller Verbrauch
- Durchschnittsverbrauch
- Geschwindigkeit
- Schubabschaltung in % der Fahrzeit
- Distanz
- Schaltvorgänge
- Beschleunigung

1.3.2.6. Einstellungsmöglichkeiten

Sämtliche Parameter sollten in einem Einstellungsmenü konfigurierbar sein. Eventuell ist eine Aufteilung auf mehrere „Konfigurationsseiten“/Kategorien sinnvoll. Verschiedener Masseinheiten für Verbrauch, Geschwindigkeit, etc. sollen berücksichtigt werden.
Es sollen mehrere Fahrzeuge pro iPhone unterstützt werden. Das Verhältnis Gang/Drehzahl soll auch manuell konfigurierbar sein, um den Rückwärtsgang und Spezialfälle (z.B. Automatikgetriebe mit 8 Stufen) ausschliessen zu können.

1.3.2.7. Untersuchungen

- Reicht bereits die Auswertung der Beschleunigung, wie dies einige iPhone-Applikationen machen, um eine Aussage über die umweltfreundliche Fahrweise zu machen? Können die Beschleunigungen aus den GPS-Daten genügend genau bestimmt werden?
- Welche „Scores“ bzw. Bewertungsfaktoren gibt es, um die EcoDrive-Fahrweise zu beschreiben und die EcoDrive-Qualität quantitativ zu erfassen?
    - EcoDrive Kennzahl: e = Geschwindigkeit * Gewicht / Verbrauch
    - PLX Devices benutzt den Kiwi Score, um eine Bewertung der Fahrweise abzugeben: „The Kiwi score represents the efficiency of your driving style. A score of 100 represents the most efficient condition and a score of 0 represents wasteful driving conditions. The Kiwi score represents the sum average of 4 parameters, smoothness, drag, acceleration and deceleration. By maximizing your Kiwi score, you’ll maximize your fuel efficiency.”

2. Abstract

Abteilung	**Informatik**
Name der Studierenden	**Selim Akyol** **Edon Berisha**
Studienjahr (Dauer der Arbeit)	**vom 22.02.10 bis 18.06.10**
Titel der Studienarbeit	**EcoHelper** **EcoHelper für das iPhone**
Examinator	**Peter Heinzmann**
Themengebiet	**Software, iPhone, OBD2-Fahrzeugdaten**
Industriepartner	**cnlab AG, Peter Heinzmann**
Institut	**ITA**

Kurzfassung der Bachelorarbeit

EcoHelper hilft, eine umweltfreundliche und kostengünstige Fahrweise zu trainieren.
Die EcoHelper-Handyapplikation erfasst über den Onboard-Diagnose-Bus (OBD2) moderner Autos Fahrdaten (z.B. Motordrehzahl, Gaspedalstellung und Benzinverbrauch). Diese Daten werden mit GPS-Positionsdaten und Bilddaten des Handys ergänzt und auf eine Web-Plattform übermittelt. Es existieren bereits EcoHelper-Applikationen für Symbian- und für JavaME- fähige Mobiltelefone. Im Rahmen dieser Arbeit wurden Verfahren zur Bestimmung des Fahrverhaltens anhand der EcoHelper-Daten entwickelt und es wurde eine EcoHelper-App für das iPhone realisiert.

Die Arbeit umfasst folgende Teilaufgaben:
• Einarbeitung in Objective-C
• Einarbeitung in EcoHelper-Konzepte
• Ausarbeitung von Möglichkeiten zur Nutzung von Beschleunigungsdaten
• Realisierung der EcoHelper-App gemäss Spezifikation
• Testfahrten

Die realisierte iPhone EcoHelper-App bietet folgende Funktionen:
• Ansprechen des WLAN-OBD2-Adapters
• Auslesen der GPS-Daten
• Auslesen von Fahrdaten (Geschwindigkeit, Motordrehzahl, Drosselklappenstellung, etc.)
• Anzeige der aktuellen Werte während der Fahrt
• Anzeige der statistischen Auswertung der Fahrt am Ende jeder Fahrt
• Übermitteln der Daten zur Web-Plattform (Online-Mode)
• Übermitteln aufgenommener Daten, sobald ein WLAN verfügbar ist (Offline-Mode)

3. Management Summary

3.1. Ausgangslage

Die cnlab AG hat EcoHelper schon für verschiedene Mobile-Operating-Systeme (z.B. Symbian, Windows Mobile, JavaME) realisiert. Nun soll in dieser Arbeit auch eine Software für das iPhone entwickelt werden.
Die iPhone-EcoHelper-Applikation soll ähnlich wie die bereits existierenden EcoHelper-Applikationen mittels OBD2 und GPS Daten erfassen, visualisieren und auswerten können. Zusätzlich zu den auf den anderen Plattformen verfügbaren Funktionen sollen nun die speziellen Vorteile des iPhone wie z.B. das grosse Touchscreen ausgenutzt werden. Insbesondere ist eine lokale Visualisierung zur Fahrweise inklusive Beurteilung anhand charakteristischer Parameter zu realisieren.

Die beiden Studenten haben diese Arbeit gewählt, weil sie selber iPhone-Fans und passionierte Autofahrer sind.

3.2. Vorgehen

In diesem Projekt wurde nach RUP (Rational Unified Process) gearbeitet. Der Analyse-Teil in diesem Projekt wurde dabei ausführlich ausgearbeitet.

Die erste Phase des Projekts bestand aus Testfahrten und Analyse der Werte. Dabei wurden die Testfahrten mit der aktuellen Symbian-Lösung und einer OBD-Diagnose-Software durchgeführt. Mittels der ausgewerteten Fahrten wurde entschieden, welche Parameter während der Fahrt und welche Parameter zur Auswertung auf den TourLive-Server geschickt werden sollten.

Die zweite Phase des Projekts bestand daraus, die iPhone-Entwicklungsumgebung und die OBD2-Schnittstelle des Autos kennen zu lernen. In dieser Phase des Projekts musste herausgefunden werden, welche Werte man über die OBD2-Schnittstelle bekommt, welche Werte in allen Fahrzeugen abrufbar und welche modellspezifisch sind.

Zudem musste herausgefunden werden, was mit dem iPhone machbar ist, welche Schnittstellen unterstützt werden, was die Richtlinien von Apple sind und wie all diese Informationen iPhone-spezifisch implementiert werden können.

Die dritte Phase des Projekts bestand aus der Implementierung. Zuerst wurde ein GUI-Prototyp erstellt, um zu sehen, wie die Oberfläche aussehen soll und ob diese von den ausgewählten Testpersonen akzeptiert wird. Anschliessend wurde die Kommunikation mit dem Fahrzeug implementiert und zum Schluss noch die Kommunikation zum Webserver hergestellt.

Das ganze Projekt wurde durch Prof. Dr. Peter Heinzmann betreut. Herr Heinzmann und seine beiden Mitarbeiter Omid Afshari und Raphael Juchli haben uns mit Rat und Tat unterstützt und waren eine grosse Hilfe in diesem Projekt. In regelmässigen Sitzungen wurden das aktuelle Ergebnis und die nächsten Schritte ausführlich besprochen.

3.3. Ergebnisse

Die folgende Grafik zeigt die vereinfachte Systemübersicht des Projekts:

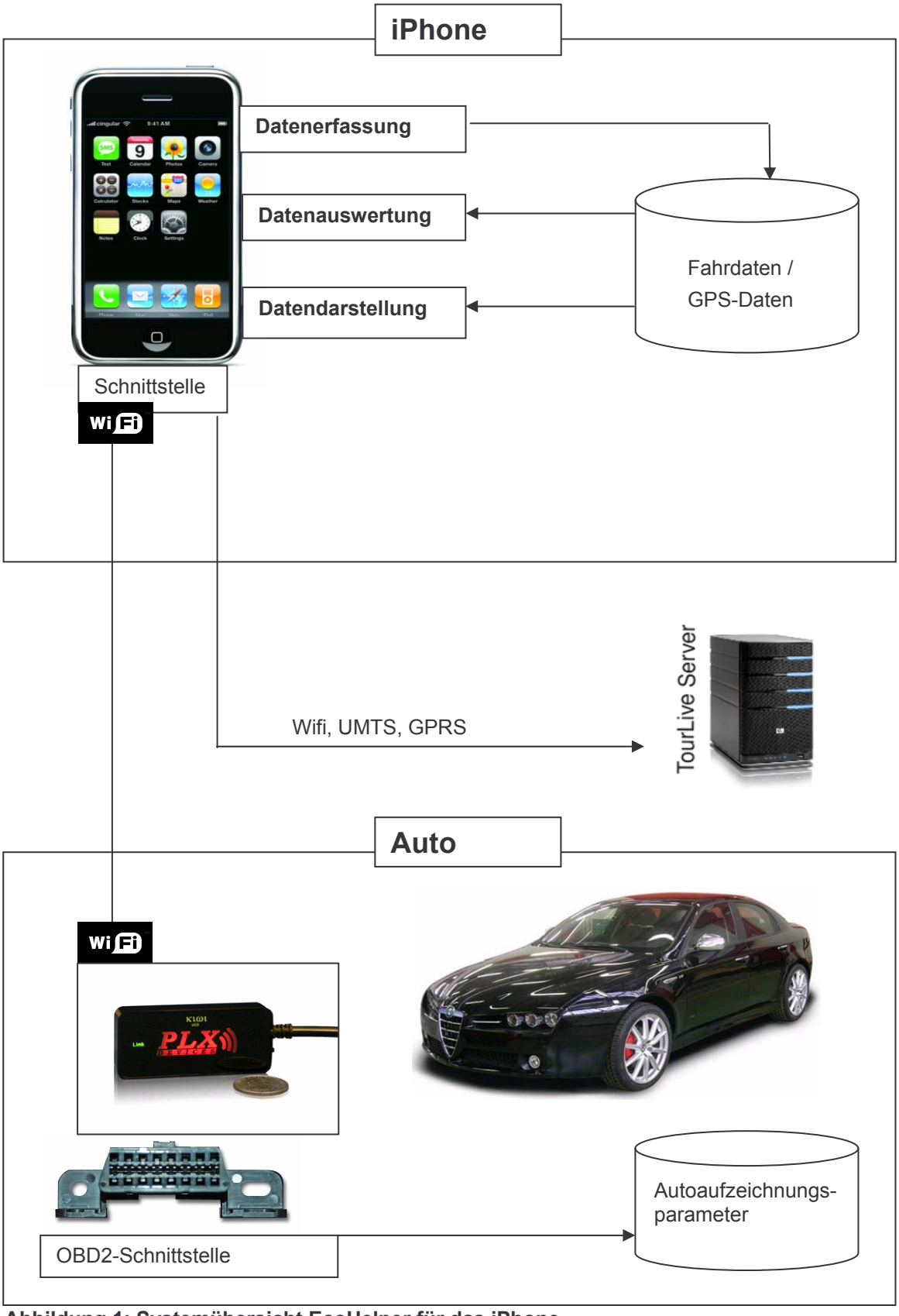


Abbildung 1: Systemübersicht EcoHelper für das iPhone

Da es schon vorherige EcoHelper-Lösungen gab, hatte man bereits einen Anhaltspunkt, wie die Kommunikation mit dem Auto hergestellt wird und wie die Berechnung der einzelnen Werte erstellt werden soll.

Doch haben wir in unserer Arbeit auch einige Berechnungen, wie z.B. die Schubabschaltung (Fuel Cut Off), verbessern können. Die alten Lösungen kommunizieren mit dem Auto über Bluetooth. Da das iPhone keine Bluetooth-Geräte ausser Audio-Bluetooth-Geräte unterstützt, kommuniziert die iPhone EcoHelper-Applikation über WLAN mit dem Wifi-Adapter (PLX Devices) des Autos.

Zu Beginn des Projekts wollte man ähnlich wie bei den bestehenden Lösungen den Kameramodus miteinbeziehen. Es sollten also während der Fahrt Fotos gemacht werden.

Da Apple bis und mit der OS Version 3.1.3 keine Parallelität unterstützt, musste auf diese Funktion verzichtet werden.

Auch wollte man zu Beginn eine eigene Zahl, ähnlich wie die EcoZahl, entwickeln. Diese Zahl sollte mittels bestimmten Fahrdatenparametern einen genaueren Wert (=HSR-Zahl) als die EcoZahl über das Fahrverhalten geben. Da der zeitliche Rahmen einer Bachelor-Arbeit für die Analyse nicht ausreichte, musste diese Zahl in der ersten Version ausgelassen werden.

Die iPhone-Lösung bietet aber auch Vorteile gegenüber den bisherigen Lösungen. Nicht nur lassen sich zum Beispiel textlich aktuelle Fahrdaten anzeigen, sondern es wird auch die ganze Bildschirmgrösse des iPhones ausgenutzt. Die Daten werden farbig und in Balken-Diagrammen zur besseren Visualisierung angezeigt.

Folgender Screenshot [Abbildung 1] zeigt wie z.B. die Detailansicht während der Fahrt aussieht:

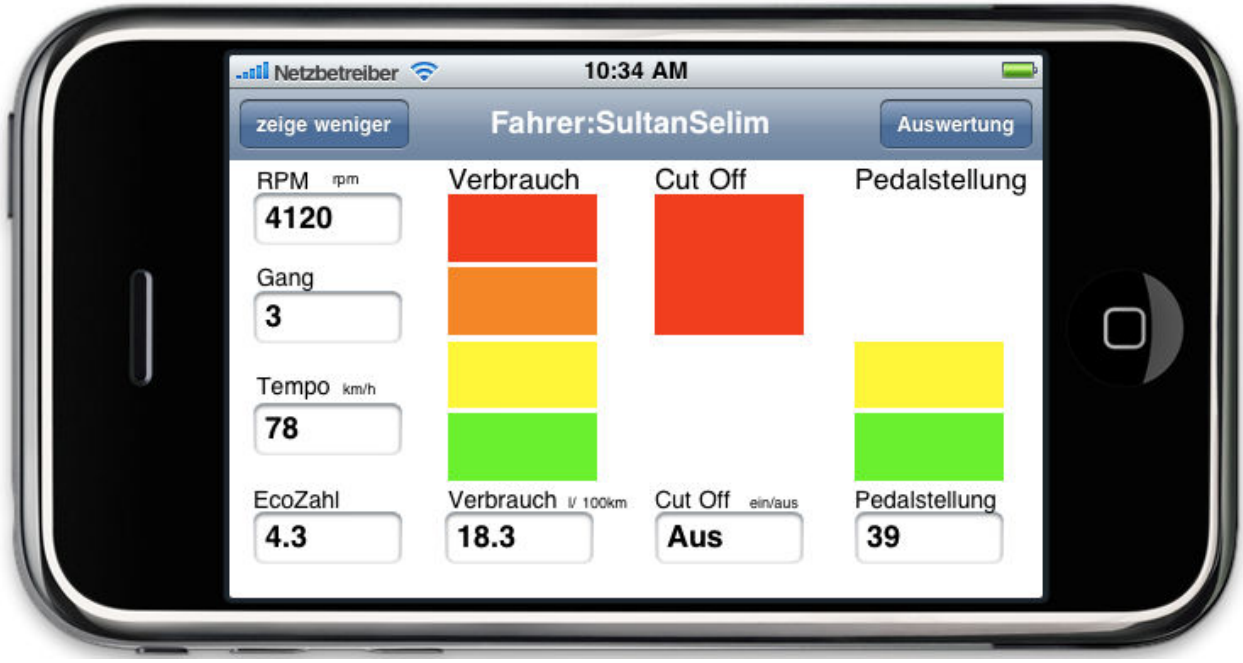


Abbildung 1: Aktuelle Anzeige während der Fahrt

Zusätzlich lassen sich die gespeicherten Fahrten statistisch und visuell mit Balken-Diagrammen auswerten. [Abbildung 2].

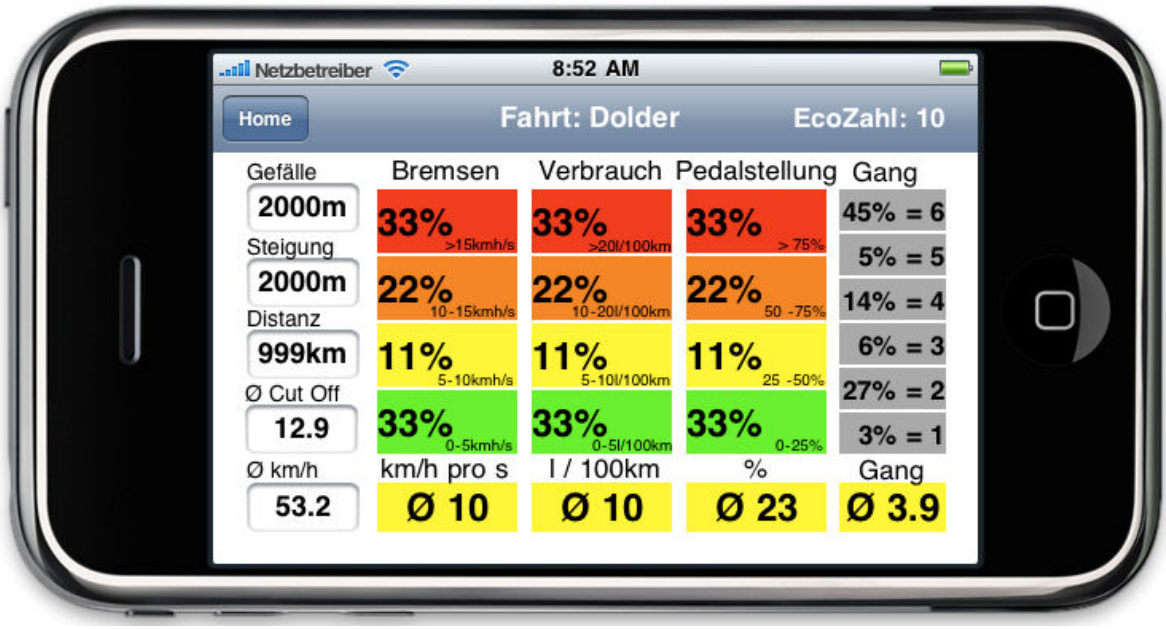


Abbildung 2: Auswertungsanzeige nach der Fahrt

Auch neu im Vergleich zur bisherigen Lösung ist, dass man mehrere Autos und Fahrer erfassen kann.

Diese Neuerungen brachten auch Änderungen auf dem TourLive Server. Neu werden auch der Fahrer und das Auto dem Server als Daten übermittelt und die EcoZahl wird nicht mehr auf dem Server, sondern auf dem Gerät berechnet und dem Server geschickt. Die Schubabschaltung wird ebenfalls auf dem iPhone berechnet und dem Server zugeschickt.

Da es von Begin an klar war, dass der Umfang des Projekts für ein ordentliches Arbeitspensum von je 360 Stunden zu gross ist, haben wir von Anfang an je 435 Stunden geplant.

Hauptgrund für diesen Arbeitsaufwand war, dass wir weder die OBD-Technologie noch die iPhone- bzw. Objective-C- Welt kannten. Nur schon das Einarbeiten in diese beiden Technologien war aufwendig. Andererseits waren wir als passionierte Autofahrer und iPhone-Fans überdurchschnittlich motiviert, das ganze Projekt zu realisieren.

3.4. Ausblick

Wie oben erwähnt wäre in der Version 2.0 die HSR-Zahl zu entwickeln. Zudem verspricht Apple, bei der neuen OS Version 4.0 Parallelität zu unterstützen. Dies würde bedeuten, dass man in Zukunft auch den Kameramodus miteinbeziehen könnte. Da mit der Version 1.0 noch keine grossen Feldtests gemacht werden konnten, werden sicher noch ein paar Änderungen anzubringen sein. Denkbar wäre es, die Bremsvorgänge genauer analysieren zu können und diese in der Version 2.0 einfliessen zu lassen.

Eine weitere Idee wäre, die EcoHelper-Applikation für das iPad zu erweitern. In diesem Fall könnte man die grössere Bildschirmfläche für noch mehr Auswertungsdaten nützen.

Schliesslich gäbe es auch die Option, dass man zusätzliche Autofunktionen anbietet. Denkbar wäre eine Art Fernbedienung, mit der man die Fenster, die Lichter oder sogar den Motor fernsteuern könnte.

4. Einleitung

4.1. Umweltproblem

Mit dem Aufkommen immer günstigerer und schnellerer mobiler Multifunktionsgeräte wie Handhelds und Pocket PCs eröffnen sich völlig neue Möglichkeiten zur Umsetzung von Ideen und Innovationen im Bereich Mobile-Computing. Heute wird oft über die Klimaveränderung und den damit verbundenen Folgen gesprochen. Einen Beitrag zur Verminderung des CO2-Ausstosses kann jeder leisten, vor allem beim Autofahren. Auch die hohen Benzinpreise regen viele Autofahrer zum Denken an. Es entsteht ein Bedürfnis, den Treibstoffverbrauch zu senken, um Kosten zu sparen und gleichzeitig die Schadstoff-Belastung zu verringern.
Es wird also ein Hilfsmittel benötigt, welches das Fahrverhalten eines Lenkers analysiert. Aus einer solchen Analyse geht hervor, in welchen Bereichen Verbesserungspotential vorhanden ist. Zudem soll die Analyse Hinweise geben, wie der Fahrstil optimiert werden kann.
Im Rahmen dieser Bachelorarbeit galt es einerseits, verschiedene Effekte zu untersuchen, welche den Treibstoffverbrauch beeinflussen. Andererseits sollte ein „EcoHelper-System“ realisiert werden, welches den Autofahrern Hinweise für eine ökonomischere Fahrweise gibt und sich einfach ins Auto einbauen lässt.

4.2. Realisierte iPhone EcoHelper-Applikation

EcoHelper ist eine Applikation für das iPhone zur Darstellung und Aufzeichnung von Fahrparametern sowie Positions- und Bewegungsdaten. Unter dem Begriff Fahrparameter sind Informationen wie Geschwindigkeit, Motordrehzahl, Gangposition und Treibstoffverbrauch zu verstehen. Die Fahrparameter werden über die weltweit standardisierte On-Board-Diagnose (OBD2[3]) Schnittstelle des Fahrzeugs ausgelesen und in einer Datenbank im iPhone abgelegt. Positionsdaten sind die mittels GPS erfassten geografischen Lokationsdaten. Bewegungsdaten werden aus den Geschwindigkeiten berechnet.

Die aufgezeichneten Fahrparameter und Bewegungsdaten werden auf der grafischen iPhone-Oberfläche angezeigt. Zudem erhält der Benutzer akustische und auch visuelle Informationen über sein aktuelles Fahrverhalten, um dies noch während der Fahrt verbessern zu können.

Alle aufgezeichneten Daten lassen sich automatisch oder mit der Exportfunktion zum EcoHelper-Web-Plattform exportieren. Dort können die Daten für weitere statistische Analysen und die Darstellung auf GoogleMap verwendet werden.

Die verschiedenen Komponenten des realisierten EcoHelper-Systems sind Abbildung 3 illustriert. Die wichtigste Systemkomponente bildet das iPhone mit der EcoHelper-Anwendung (1). Diese kommuniziert mittels WLAN mit der OBD2-Schnittstelle(2). Die gesammelten Daten werden dann zum TourLive Server gesendet(3).

[3] Der OBD2-Standard wird in Europa auch European On Board Diagnosebus (EOBD) genannt.

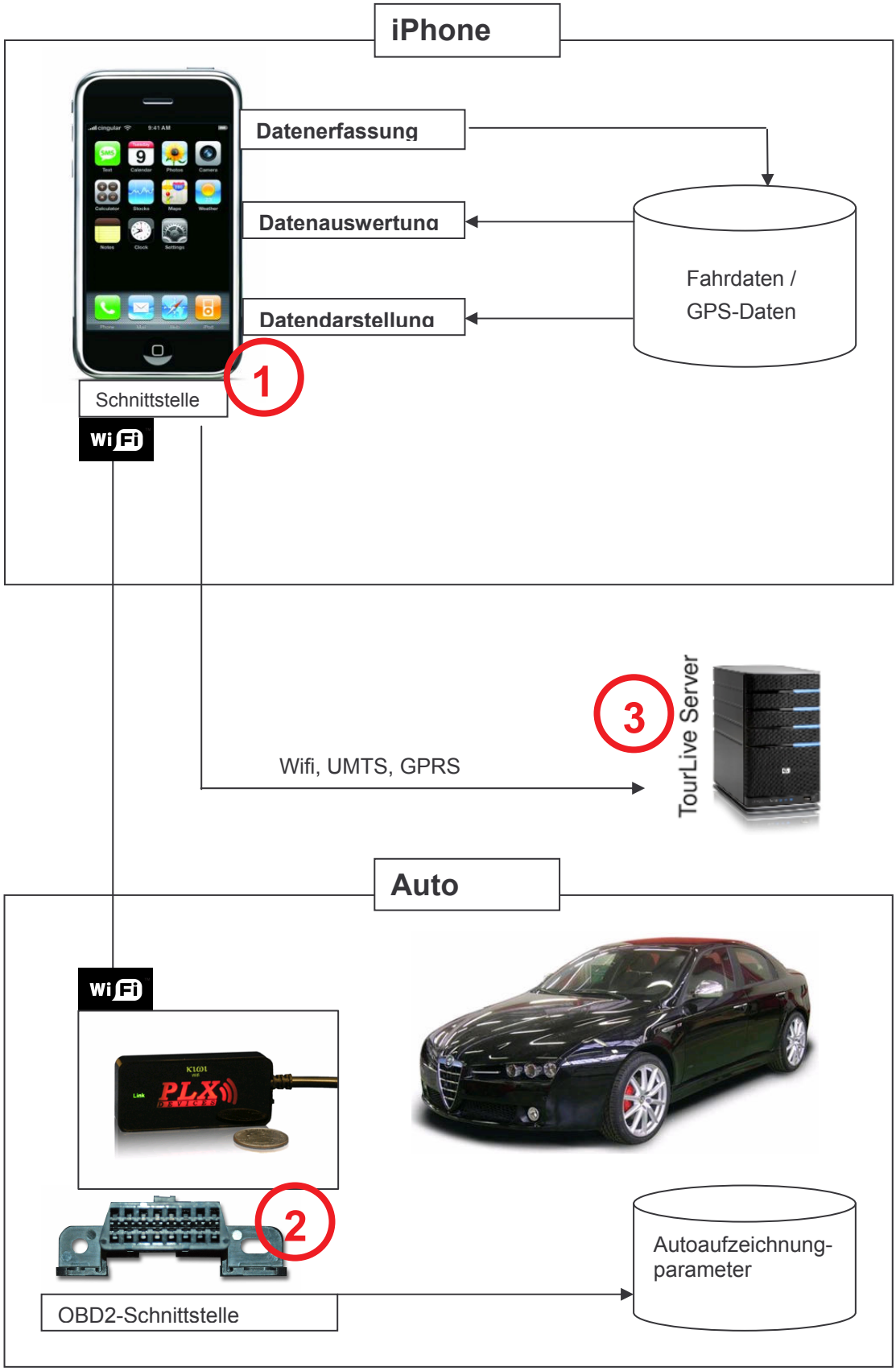


Abbildung 3 Systemübersicht EcoHelper für das iPhone

4.3. Hinweise zum Bericht

Kapitel 5 Analyse:

Dieses Kapitel beschreibt die Vorgehensweise und die Erkenntnisse der Analyse.

Es wird untersucht, welche Faktoren für den Verbrauch wichtig sind und welche Werte aus dem Auto auslesbar sind. Dieses Kapitel richtet sich an Interessierte des Treibstoffverbrauchs und an Autofanatiker, die erfahren möchten, auf welche Weise die verschiedenen Werte erstellt und berechnet werden.

Kapitel 6 Realisierung:

Dieses Kapitel richtet sich an Software-Ingenieure. In diesem Kapitel wird die Architektur der Applikation präsentiert. Die Architekturwahl und -beschreibung, der Aufbau sowie die benutzten Design-Patterns werden hier aufgezeigt. Auch werden die verschiedenen Software-Klassen und Ihre Beziehungen zueinander aufgezeigt.

Anhang:

Im Anhang finden alle wichtigen Dokumente wie Anforderungsspezifikationen, Projektplan, Protokolle, wichtige Kennzahlen und Erkenntnisse etc. ihren Platz. Die Personen, die genauere Details zu einem bestimmten Thema suchen, werden dort fündig.

Projektplan:

Der Projektplan zeigt auf, wie das ganze Projekt aufgegliedert und nach RUP umgesetzt wurde. Auch wird hier eine Übersicht der investierten Zeit pro Phase gegeben. Zudem werden wichtige Erkenntnisse betreffend der Projektplanung nochmals zusammengefasst.

Benutzeranleitung:

Die Software sollte selbsterklärend sein, doch um die Benutzung zu vereinfachen und den korrekten Umgang mit sämtlichen Funktionen von Anfang an zu gewährleisten, wurde eine Benutzeranleitung erstellt, die eine reibungslose Bedienung ermöglicht. Alle Funktionen, aber auch die richtige Vorgehensweise, werden Schritt für Schritt erklärt.

5. Analyse

5.1. Einflussumgebungen

Der Treibstoffverbrauch wird durch verschiedene Faktoren beeinflusst. Die Ursache des Treibstoffverbrauchs wird in die Hauptbereiche „Fahrstrecke“, „Verkehrssituation“, „Fahrzeugtechnik“ und „Fahrverhalten“ unterteilt.

Abbildung 4 Hauptbereiche des Treibstoffverbrauchs

Fahrstrecke

Ob auf einer Fahrt grosse Steigungen überwunden werden müssen oder ob man meistens abwärts fahren kann, beeinflusst den Treibstoffverbrauch. Auch Kurven und Fahrbelag wirken sich auf den Treibstoffverbrauch aus. Schliesslich spielt es eine Rolle, ob man nur über kurze oder über längere Strecken fährt.

Verkehrssituation

Wenn sich der Fahrer in einem Stau befindet, dann wird er wahrscheinlich ganz und gar nicht ökonomisch fahren können, da er kurzfristig immer wieder abbremsen und beschleunigen muss, um vorwärts zu kommen. Eine solche Situation wird bekanntlich „Stop-and-Go“ genannt.

Fahrzeugtechnik

Der hohe Kraftstoffverbrauch kann natürlich auch andere Ursachen haben, wie z.B. zu tiefer Reifendruck, ungeeignete Aerodynamik der Karosserie, schlechter Motorzustand, häufiger Einsatz der Klimaanlage, ungenügende Wartung und Pflege des Autos usw.

Fahrverhalten

Die Beurteilung des Fahrverhaltens wird durch die EcoHelper-Applikation aufgrund von verschiedenen Faktoren, wie z.B. Beschleunigung, Gaspedalstellung, Schubabschaltung, Motordrehzahl, usw., merklich vereinfacht und kann gezielt zum Antrainieren eines besseren Fahrverhaltens eingesetzt werden.

5.2. Analyse zum Treibstoffverbrauch

5.2.1. Einleitung

In der Einarbeitungsphase zur vorliegenden Bachelorarbeit sollte der Einfluss verschiedenster Faktoren auf den Treibstoffverbrauch untersucht werden. Es ging vor allem darum herauszufinden welche relevanten Parameter man aus einem Auto überhaupt auslesen kann und wie sich diese zueinander verhalten. So kann man dann die verschiedenen Werte in Relation zum Kraftstoffverbrauch evaluieren und in der zu entwickelnden Applikation entsprechend berücksichtigen.
Für das Auslesen der Daten wurde die Schnittsstelle im Auto namens OBD2 benutzt. „On-Board-Diagnose“ oder auch „OBD2“ ist ein standardisiertes Fahrzeugdiagnosesystem. Dieses wird in erster Linie zum Lesen von Fahrzeugdaten (Fehlercodes) verwendet. Es liefert je nach Fahrzeugtyp auch aktuelle Informationen, wie z.B. Geschwindigkeit, Motordrehzahl, Mass Air Flow (Luftdurchfluss = Verbrauchsindikator), Drosselklappen- bzw. Gaspedalstellung. Das ODB2-System brauchen typischerweise auch Autodiagnostiker, um den Zustand eines Fahrzeugs zu überprüfen. Über die OBD-Schnittstelle können Garagisten den Motorzustand abfragen und eventuelle Fehler herauslesen. Zudem werden die heutigen Abgaskontrollen auch über die OBD-Schnittstelle vorgenommen. Weitere Details findet man im Anhang unter dem Kapitel „OBD2“ oder auf den Internetseiten [URL 1] und [URL 2].

Nach genaueren Analysen, die im folgenden Kapitel 5.2.2 näher erläutert werden, hat man folgende wichtige Fahrzeugwerte definiert, die für die iPhone-Applikation relevant sind:

OBD2

- *Throttle Position* (Drosselklappenstellung) → Klappenänderung / Zeit (*delta Throttle Position*)
- *Speed* (Geschwindigkeit) → Geschwindigkeitsveränderung / Zeit (*Beschleunigung*)
- *RPM* (Motordrehzahl) → RPM-Veränderung / Zeit (delta RPM)
- *Mass Air Flow* (MAF, steht im Zusammenhang mit dem Verbrauch (Liter / 100km))

GPS (Global Positioning System = globales Navigationssystem)

- *Höhe* → Höhenveränderung / Zeiteinheit (*delta Höhe bzw. Steigung*)
- *Gefahrene Streckendistanz* (in km)

Die wichtigsten Fragen waren:

1. Welche Informationen bekommt man vom Auto tatsächlich?
2. Wie verhalten sich die verschiedenen Werte zueinander?
3. Um ein sparsames Fahrverhalten zu trainieren, muss der Kraftstoffverbrauch ständig überwacht und mit verschiedenen Werten in Verbindung gebracht werden, um Optimierungensgmöglichkeit aufzeigen zu können. Um welche Werte handelt es sich?

Auf den nächsten Seiten sind die Antworten auf diese Fragen in verständlicher Form präsentiert. Diese Informationen fliessen dann in die EcoHelper-Applikation ein.

5.2.2. Fahrdatenanalyse

Material

Um mit dem OBD2 kommunizieren zu können, braucht es entsprechende Geräte, welche von cnlab AG zur Verfügung gestellt wurden.

Materialliste:

- Nokia N92, Nokia N95 mit Symbian-Lösung von EcoHelper, genannt TourLive
- OBD2-Bluetooth-Adapter
    - OBD2-Stecker
    - OBD2-Kabel (Hersteller: Carcode Müller)
- Diagnosesoftware (ScanMaster)

Autofahrt

Für die Datenauswertung wurden vor allem 2 Testfahrten berücksichtigt. Die erste Strecke war die Fahrt von Dübendorf nach Dolder und zurück mit einem Alfa Romeo 156 2.5l V6 Sportwagon. Die Summenhäufigkeitsanalyse der vier Parameter Beschleunigung, Gang, Verbrauch und Gaspedalstellung bezieht sich auf die Neerach-Fahrt von Herrn Heinzmann mit einem Mazda 3 1,6l.

Evaluierung

In der TourLive-Applikation kann man in den Einstellungen vom Online- zum Offline-Modus umschalten. Im Online-Modus werden die aufgenommenen OBD2- und GPS-Daten fortlaufend an eine Webplattform gesendet und im Offline-Modus werden die Daten zwischengespeichert und erst bei verfügbarem WLAN mit Internetverbindung ebenfalls an die Webplattform geschickt.

Von dort kann man dann die entsprechenden Fahrdaten in Form einer CSV-Datei holen und die Daten so in Excel evaluieren.

Im Sekundenabstand werden die Daten von TourLive abgetastet und sind so jeweils auch in der CSV-Datei vorhanden.

Auf den nachfolgenden Seiten werden die wichtigsten Fahrparameter untereinander verglichen und die Resultate graphisch in Excel-Diagrammen dargestellt.

5.2.3. Throttle-Position vs. MAF

Beschreibung: Bei Benzinmotoren wird über die Drosselklappe (Throttle) die Luftmenge geregelt, welche in die Zylinder gelangt. Die Stellung der Drosselklappe steuert der Fahrer direkt (mechanisch oder elektronisch) über das Gaspedal.

Der Korrelationsfaktor zwischen der Throttle-Position (grüne Kurve) und dem MAF (rote Kurve) beträgt im vorliegenden Beispiel 0.59. Das heisst, die Kurven in der untenstehenden Grafik sind sehr ähnlich. (Korrelation 1.0 würde bedeuten, dass sich die beiden Kurven, abgesehen von den verschiedenen Wertebereichen, identisch verändern.) Daraus lässt sich schliessen, dass die Veränderung der Throttle-Position eine gleichartige Veränderung des MAF zur Folge hat (Musterähnlichkeit). Je stärker man auf das Gaspedal tritt, desto höher steigt der MAF. In den jeweiligen Streckensteigungen kann man beobachten, dass dort der Gang tiefer ist (höhere Zugkraft) und bei Gefällen der Gang höher ist.

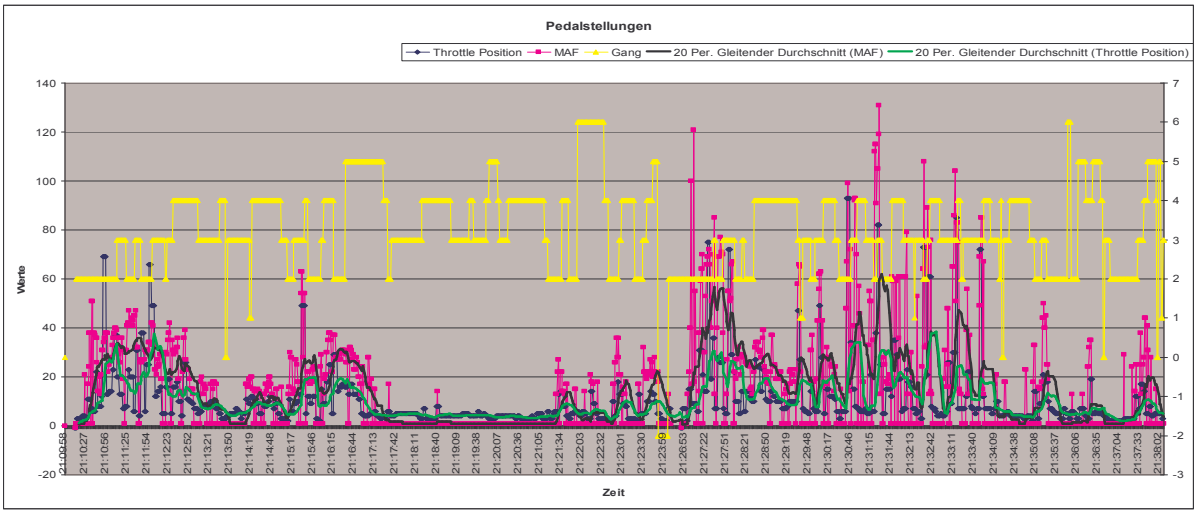


Abbildung 5 Throttle-Position vs. MAF {Ref: seak_Statistikauswertung (EcoHelper)_DOLDER.xls}

5.2.4. Speed (km/h) vs. MAF (inkl. Gang-Information)

Beschreibung: Bekanntlich werden kleine Gänge bei tiefen Geschwindigkeiten eingelegt, was tendenziell ein höheres MAF erzeugt. Demgegenüber eignen sich grosse Gänge besser für höhere Geschwindigkeiten, was sich in einem kleineren MAF widerspiegelt.

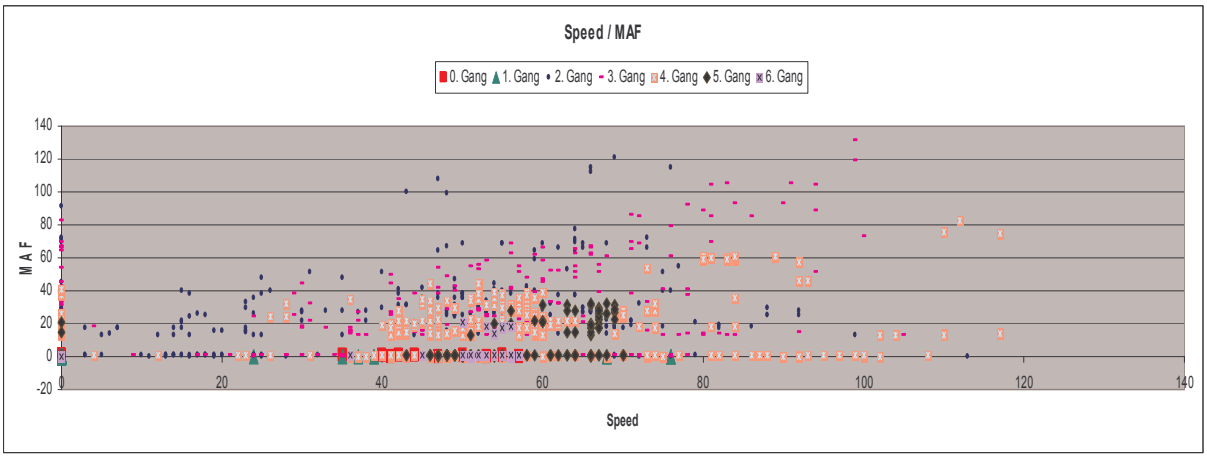


Abbildung 6 Speed (km/h) vs. MAF {Ref: seak_Statistikauswertung (EcoHelper)_DOLDER.xls}

5.2.5. Throttle-Position vs. Speed (km/h)

Beschreibung: Die Idee war bei diesem Diagramm, dass man nicht den Verbrauch analysiert, sondern anhand der Darstellung eine Fahrsituation erkennen kann. Bei den Geschwindigkeiten 50km/h und 80km/h wird z.B. eine Veränderung der Throttle-Position aufgezeichnet, weil beschleunigt wird, um die gewünschte Geschwindigkeit zu erreichen. Vereinzelt sieht man „Zick-Zack"-Formen, die auf eine unregelmässige Neigungsänderung der Drosselklappe und somit des Gaspedals schliessen lassen. Dies kommt vor in Situationen wie z.B. Stau, stockendem Verkehr oder – wie in unserem Fall auf der Dolder-Strecke – kurvige Strecken.

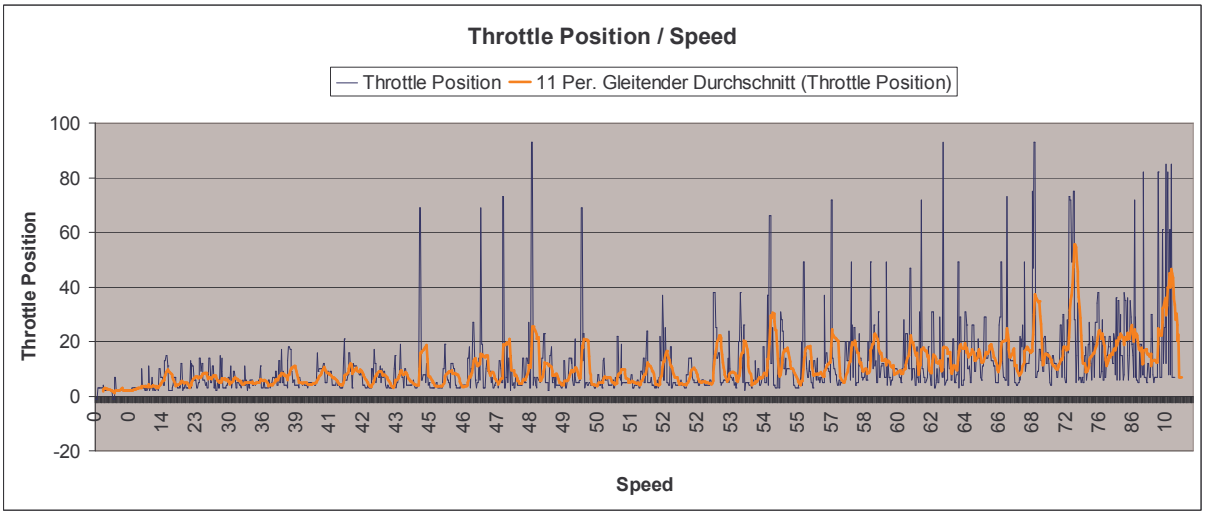


Abbildung 7 Throttle Position / Speed {Ref: seak_Statistikauswertung (EcoHelper)_DOLDER.xls}

5.2.6. RPM vs. MAF

Beschreibung: Die Analyse des Verhältnisses zwischen Drehzahl und MAF brachte eine wichtige Erkenntnis. Auffallend ist, dass sich die beiden Kurven sehr ähneln. Daraus lässt sich schliessen, dass die Höhe des MAF von der Drehzahl abhängig ist. Korreliert man die beiden Parameter in dieser Fahrsituation, erhält man einen Wert von 0.81, womit die Ähnlichkeit der beiden Parameter bewiesen ist.

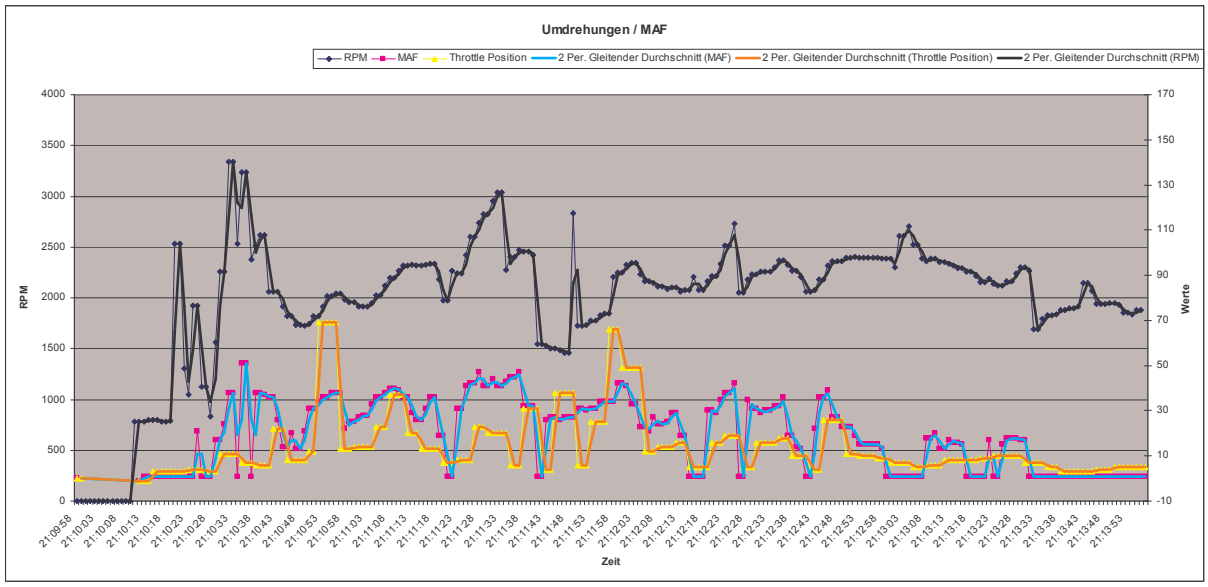


Abbildung 8 RPM vs. MAF {Ref: seak_Statistikauswertung (EcoHelper)_DOLDER.xls}

5.2.7. RPM vs. Speed

Beschreibung: Aus den über OBD erfassbaren Parametern Motordrehzahl (Rounds per Minute, RPM) und Geschwindigkeit kann berechnet werden, mit welchem Übersetzungsverhältnis bzw. Gang momentan gefahren wird.

Die Idee hinter dieser Darstellung ist, bestimmte Muster aufzuzeichnen, die Informationen darüber geben, in welchem Gang gefahren wird. Erstellt wird eine Formel zur Gangberechnung mit Angabe der Geschwindigkeit und der Drehzahl.
Bei genauer Betrachtung der Grafik sieht man, dass es diagonalähnliche Linien gibt.
Diese stellen in Wirklichkeit die Gänge dar. Dabei entspricht der 1. Gang der grössten Steigung und der 6. Gang der kleinsten Steigung. Die einzelnen Punkte, die nicht auf den Linien liegen, gehören zu Messwerten, bei welchen kein Gang eingelegt bzw. ausgekuppelt war (Leerlauf).

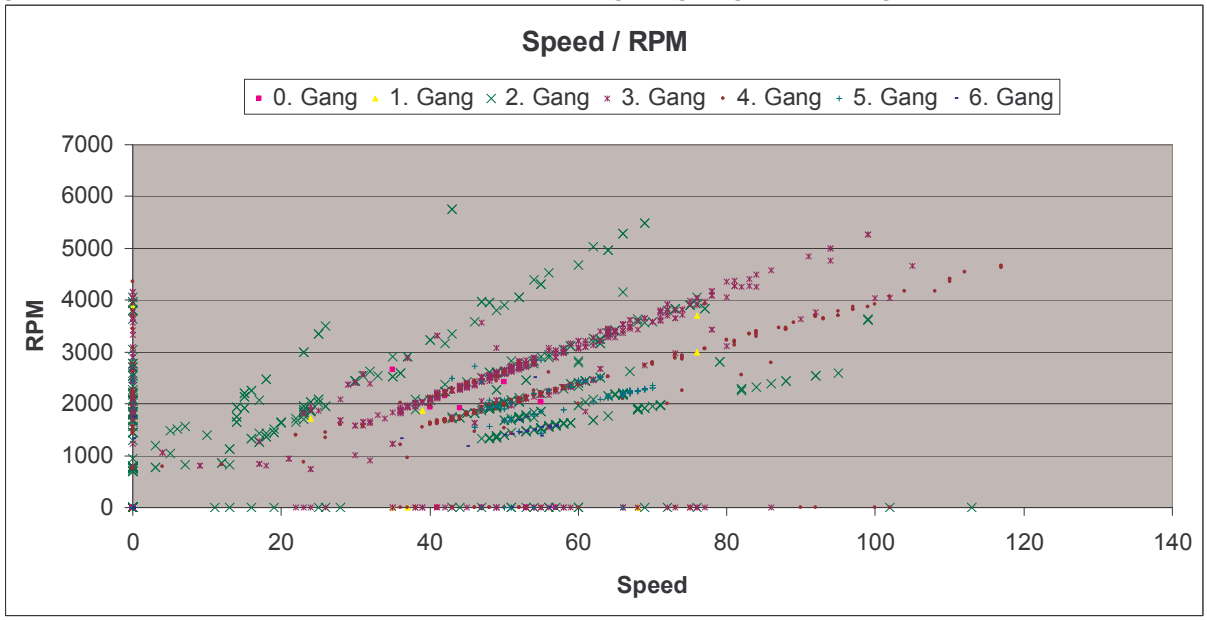


Abbildung 9 RPM vs. Speed {Ref: seak_Statistikauswertung (EcoHelper)_DOLDER.xls}

5.2.8. Beschleunigung vs. MAF

Beschreibung: Das folgende Scatterdiagramm zeigt, dass das MAF bei grossen Beschleunigungen wie erwartet stärker ansteigt und bei kleineren Beschleunigungen weniger stark. Das Runden der Zahlen erklärt wahrscheinlich den MAF-Unterschied bei gleicher Beschleunigung. Die MAF-Werte bei negativen Beschleunigungen sind ebenfalls über 0, da immer noch Benzin verbraucht und mit der Motorbremse gebremst wird.

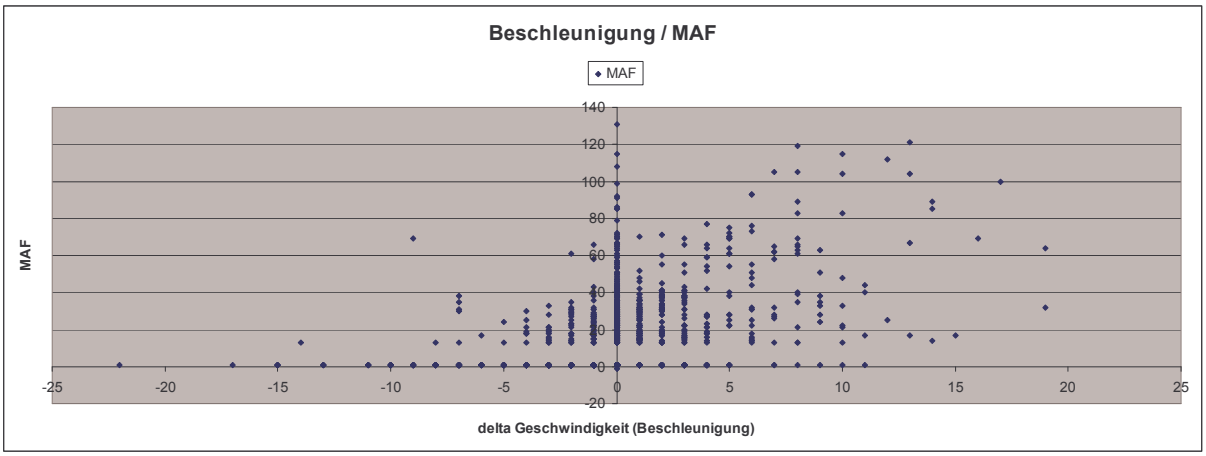


Abbildung 10 Beschleunigung vs. MAF {Ref: seak_Statistikauswertung (EcoHelper)_DOLDER.xls}

5.2.9. Korrelationen

Strecke	RPM / km/h	RPM / MAF	TP / MAF	RPM / TP
Dolder	0.70	0.81	0.59	0.46

Tabelle 1: Korrelationen

Informationen über das Thema Korrelation sind im Anhang „Wichtige Kennzahlen“ [E2] zu finden.

5.2.10. Fazit

Es lassen sich nicht nur verbrauchstechnische Informationen filtern, sondern auch Informationen zur Verkehrssituation. Wenn man diese beiden kombiniert, dann bekommt man authentische Schlussfolgerungen und kann den User für ein ökonomisches Fahrverhalten sensibilisieren.

Folgende Parameter stehen in direkter Verbindung zum MAF:

- Throttle-Position (Gaspedalstellung)
- Motordrehzahl (RPM)
- Beschleunigung (Geschwindigkeitsveränderung / Zeit)
- Streckensteigung und -gefälle (Höhenunterschied / Zeit)
- Standardabweichung
- Schubabschaltung

5.2.11.Schubabschaltung

[URL 3] Schubabschaltung ist eine gesteuerte Unterbrechung der Kraftstoffzufuhr zu einem Verbrennungsmotor, wenn dieser keine Leistung abgeben soll, sondern durch seine Schwungmasse weiter angetrieben wird, der Motor also angeschoben wird.
Eine Schubabschaltung wurde zuerst bei Dieselmotoren eingesetzt, wobei die Einspritzpumpe die Kraftstofförderung abschaltet, wenn der Drehzahlregler aktiv und die Motordrehzahl zu groß war. Das trat in der Regel dann ein, wenn man das Gaspedal nicht betätigt hatte und der Motor vom Fahrzeug geschoben wurde. Beim Ottomotor wird die Schubabschaltung seit 1980 in elektronischen Einspritzanlagen verwendet. Dabei wird die Kraftstoffzufuhr über die Einspritzventile abhängig von den Parametern Motortemperatur, Drehzahl, Drosselklappen- bzw. Gaspedalstellung abgeschaltet.
Im Wesentlichen dient die Schubabschaltung der Kraftstoffeinsparung. Sie kommt zum Tragen, wenn man die Geschwindigkeit verringert und den Motor dadurch als Bremse nutzen kann (Motorbremse). Die Entsprechung nennt man Schubbetrieb. Dabei darf man nicht die Kupplung betätigen, da dann der Motor vom Antriebsstrang getrennt wird und wieder Kraftstoff verbraucht. Es ist dann zwar relativ wenig, aber immer noch deutlich mehr als bei der Nutzung der Schubabschaltung. Eine Kraftstoffeinsparung erreicht man daher durch vorausschauende Fahrweise, indem das Gas rechtzeitig komplett zurückgenommen wird und man möglichst spät bremst. So nutzt man die Bewegungsenergie des Fahrzeugs optimal aus.

Bei modernen Fahrzeugen mit einer Anzeige des momentanen Kraftstoffverbrauches kann man den Unterschied zwischen Leerlauf und Schubabschaltung sehr gut beobachten. Bei der Schubabschaltung sollte die Anzeige 0,0 l / 100 km anzeigen. In einigen Ausnahmefällen (wie beispielsweise bei Automobilen der Hersteller Fiat und Daihatsu) ist die Anzeige trotz Schubabschaltung immer noch bei 1,0 bis 2,0 l / 100 km.
Um diese Schubabschaltung erkennen zu können wird abgefragt ob sich das Auto im Open oder Closed-Loop befindet. Die Schubabschaltung findet dann im Open-Loop Status statt.

Ein Fahrzeug befindet sich im Open-Loop[4]:

-wenn das Auto eingeschaltet wird und die Lambda-Sonde (auch O2-Messer genannt) auf Betriebstemperatur geheizt werden muss oder die Schubabschaltung aktiviert ist.

-wenn das Auto schnell beschleunigt wird. (z.B. Kick-Down)

Ein Fahrzeug befindet sich im Closed-Loop[5]:

-wenn die Lambda-Sonde ihre Betriebstemperatur erreicht hat.

-wenn das Auto normal gefahren wird.

[4] Im Open Loop werden die Feedback-Meldungen der Lambda-Sonde zur Erhaltung der Abgas-Werte ignoriert. Es findet also keine Regelung des Benzin/Luft-Gemisches statt. Sondern die Einspritzung wird anhand der Drosselklappenposition und des Luftdurchfluss bestimmt. Also ein fetthaltigeres Gemisch produziert.

[5] Im Closed Loop werden die Feedback-Meldungen zur Regelung der Abgaswerte entsprechend der Tabelle des ECU ausgewertet. Aus diesen Meldungen werden dann die neuen Benzin/Luft Gemische hergestellt um die Abgas-Werte einzuhalten.

5.3. Wichtige Kennzahlen

Wichtige Kennzahlen für die Auswertung des Fahrverhaltens bzw. des Fahrstils sind die Korrelation, die Standardabweichung, die Eco-Kennzahl und die Gaspedalstellung (Roheierfahrer oder Bleifüssler). Informationen dazu findet man im Kapitel 13 „Anhang“ unter „Wichtige Kennzahlen“.

5.3.1. Summenhäufigkeiten

Beschreibung: Man sieht hier gut, dass der Fahrer in fast 90% der Fahrt mit einer Gaspedalstellung von 5%-15% und den Rest der Fahrt mit einer Gaspedalstellung von 15%-35% gefahren ist.

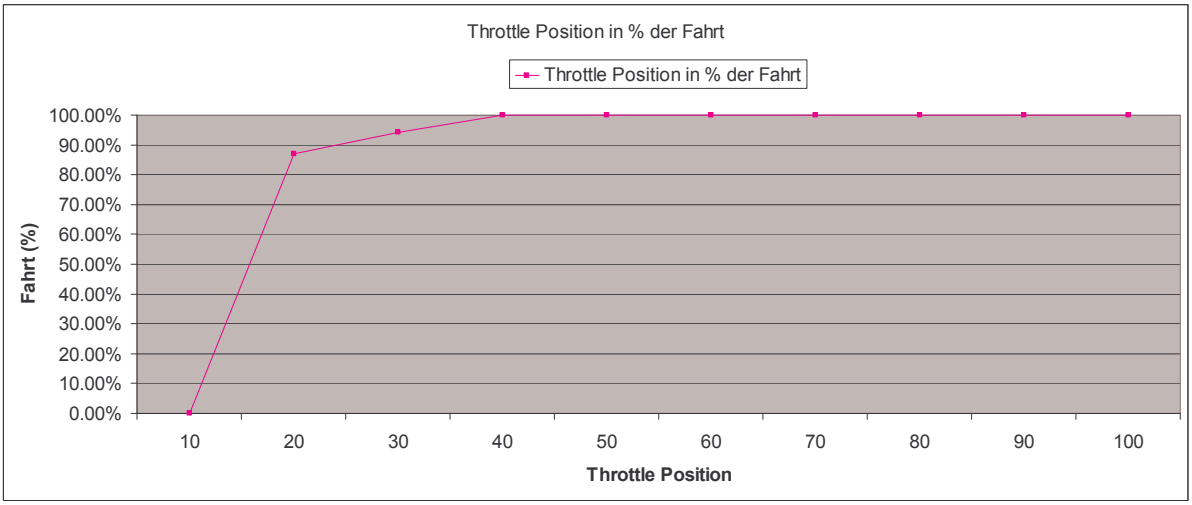


Abbildung 11 Summenhäufigkeit Gaspedalstellung {Ref: seak_Statistikauswertung (EcoHelper)_DOLDER.xls}

Beschreibung: Wenn man die Grafik genauer anschaut, dann erkennt man, dass sich der Fahrer meistens bzw. ca. 45% der Fahrt zwischen dem 4. und 5. Gang befindet.

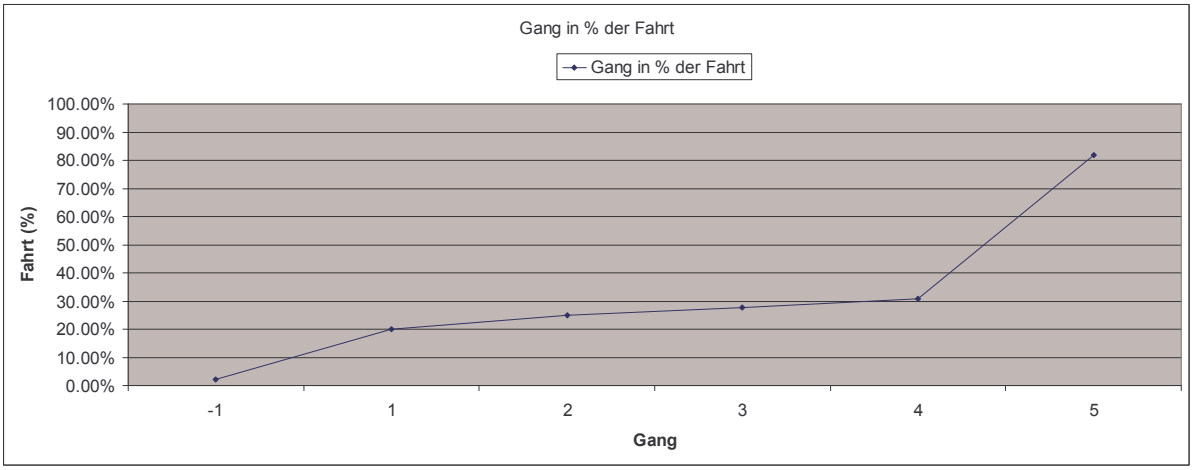


Abbildung 12 Summenhäufigkeit Gang {Ref: seak_Statistikauswertung (EcoHelper)_DOLDER.xls}

Beschreibung: Die negativen Beschleunigungen zwischen -40m/s^2 und -10 m/s^2 fanden in ca. 3% der Fahrt statt. Von -10 – 0 etwa in 53% der Fahrt und die positiven Beschleunigungen von 0 m/s^2 – 40 m/s^2 in etwa 38% der Fahrt.

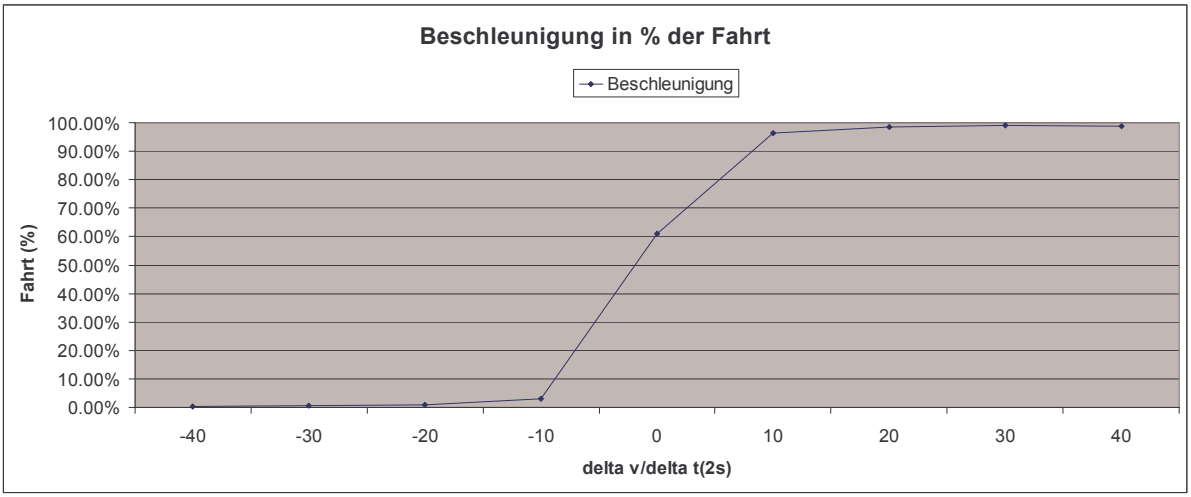


Abbildung 13 Summenhäufigkeit Beschleunigung {Ref: seak_Statistikauswertung (EcoHelper)_DOLDER.xls}

Beschreibung: In ca. 70% der Fahrt fährt der Fahrer mit einer MAF von 5-20g/s und den Rest der Fahrt mit ca. 20-40g/s.

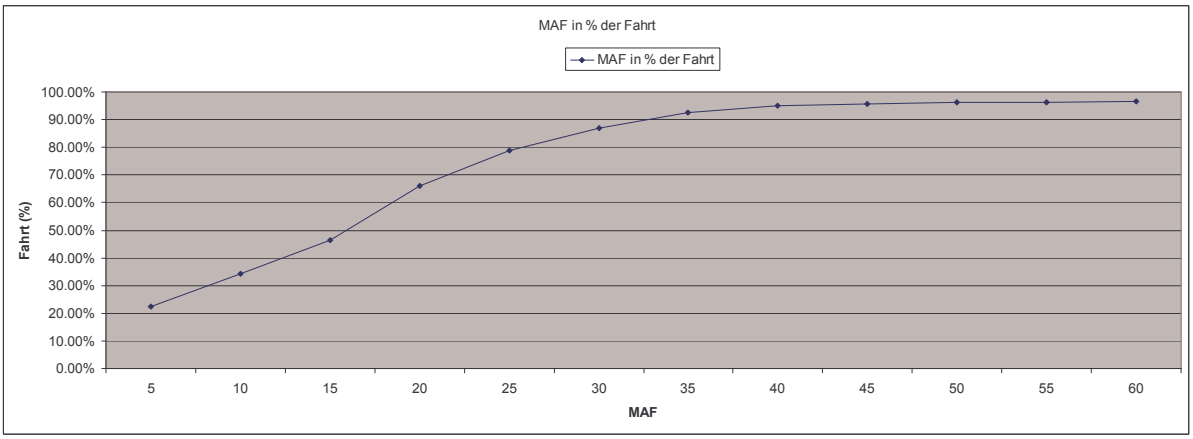


Abbildung 14 Summenhäufigkeit Beschleunigung {Ref: seak_Statistikauswertung (EcoHelper)_DOLDER.xls}

5.4. Berechnung der wichtigen Fahrparameter über OBD2-Werte

5.4.1. Geschwindigkeit

Die Geschwindigkeit wird von OBD als ein Byte-Wert zurückgegeben. Mit folgender Formel kann die Geschwindigkeit in [km/h] berechnet werden, wobei A für den erhaltenen Byte-Wert in Dezimal steht.

$Geschwindigkeit\,[km/h] = A$

5.4.2. Motordrehzahl

Die Motordrehzahl wird von OBD als zwei Byte-Wert zurückgegeben. Mit folgender Formel kann die Motordrehzahl in Umdrehungen pro Minute berechnet werden, wobei A für den ersten und B für den zweiten Byte-Wert in Dezimal steht.

$Motordrehzahl\,[\text{umdrehungen/min}] = ((A*256)B) / 4$

5.4.3. Mass Air Flow (MAF)

Der Mass Air Flow (MAF) gibt die aktuelle Luftmenge an, dieser wird von OBD als zwei Byte-Wert zurückgegeben. Mit folgender Formel kann der Mass Air Flow in [g/s] berechnet werden, wobei A für den ersten und B für den zweiten Byte-Wert in Dezimal stehen.

$MAF\,[g/s] = ((A * 256) B) / 100$

5.4.4. Gaspedalstellung

Die Gaspedalstellung wrid von OBD als ein Byte-Wert zurückgegeben. Mit folgender Formel kann die Gaspedalstellung in Prozent berechnet werden, wobei A für den erhaltenen Byte-Wert in Dezimal steht.

$Gaspedalstellung[\%] = A*100) / 255$

5.4.5. Gangposition

Theorie

Dieser Abschnitt befasst sich mit der Berechnung der Gangposition. Theoretisch ist es möglich, die einzelnen Übersetzungen zu ermitteln, um daraus die Gangposition zu bestimmen. Geht man von einem linearen Getriebe aus, kann aus der Geschwindigkeit und Drehzahl das Übersetzungsverhältnis für jeden Gang ausgerechnet werden. Dazu kann folgende einfache Formel angewendet werden.

Übersetzungsverhältnis = *Drehzahl / Gewschindigkeit*

Geschwindigkeit: v [km/h]
Drehzahl: [rpm]

Wendet man obige Formel an, kann daraus für jeden Gang des Autos das Übersetzungsverhältnis ausgerechnet werden. Dazu folgendes Beispiel:

Gangposition	Drehzahl [rpm]	Geschwindigkeit[km/h]	Übersetzungsverhältnis
1	5300	50	106
2	3000	50	60
3	3750	100	37.5
4	4000	150	26.667
5	4250	200	21.25

Tabelle 2 Übersetzungsverhältnisse Gang, RPM, km/h

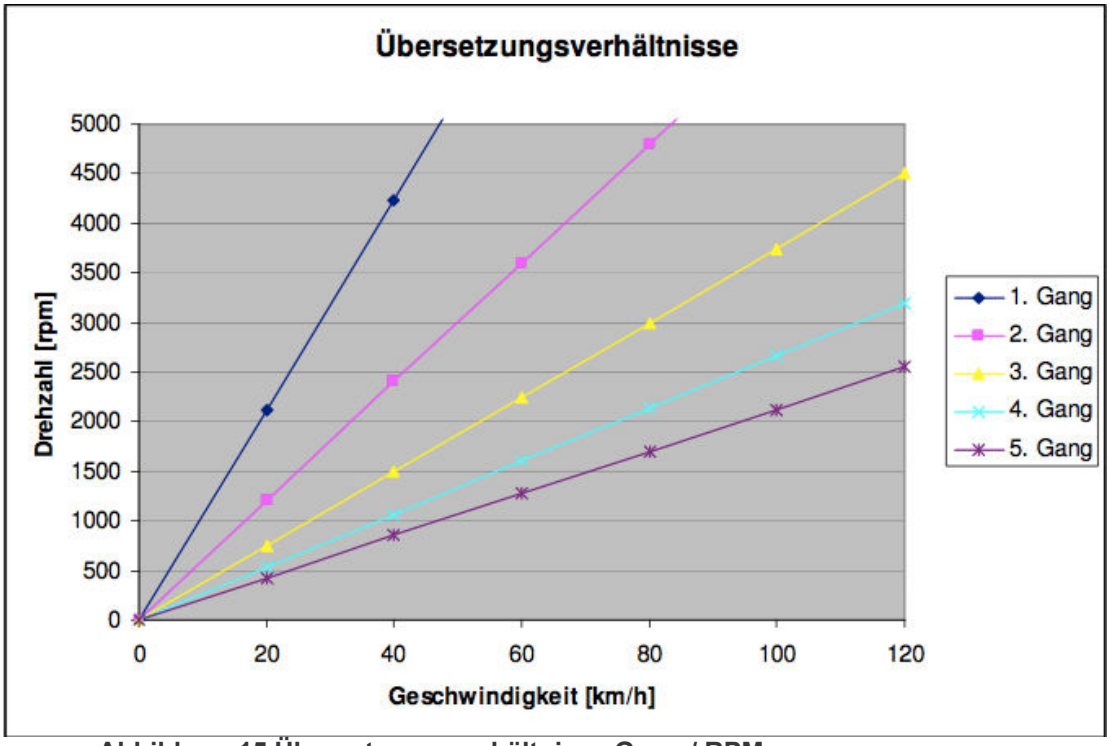


Abbildung 15 Übersetzungsverhältnisse Gang / RPM

Die Übersetzungsverhältnisse sind von Auto zu Auto unterschiedlich. Darum müssen diese Übersetzungsverhältnisse für jeden Typ bestimmt werden. Wie man aus Abbildung 16: Beispiel Gangberechnung (Übersetzungsverhältnisse) erkennen kann, liegen die Übersetzugsverhältnisse immer näher beieinander je höher der Gang ist.

Umsetzung
Im EcoHelper muss jeder Fahrt ein Auto zugeordnet werden. Beim Erstellen eines Autos muss das Übersetzungsverhältnis jedes einzelnen Gangs erfasst werden.
Da nun die Übersetzungen der Gänge bekannt sind, kann während der Datenaufnahme die aktuelle Übersetzung berechnet und mit den gespeicherten verglichen werden. So ist es möglich den aktuell eingelegten Gang zu ermitteln.

5.4.6. Verbrauch

Moderne Fahrzeuge benutzen die Daten von Luftsensoren um das Verhältnis zwischen Luft und Treibstoff zu berechnen. Dieses Verhältnis beträgt nach dem idealen chemischen Wert, 14.7 Gramm Luft auf jedes Gramm Treibstoff. Mass Air Flow (MAF) ist die Masse an Luft die pro Sekunde verbraucht wird.
Mit folgender Formel kann der Treibstoffverbrauch berechnet werden:
MassAirFlow [g/s] =MAF
Verbrauch [g/s] =x =MAF 14. 7
Dichte [g/l] =d =720 g dm³
Verbrauch [l/s] =y =x ds
*Verbrauch [l/h] =y * 3600*
*MAF →Verbrauch [l/h] =MAF / 14.7) / 720) *3600*

Der Verbrauch in [l/100Km] lässt sich über die Geschwindigkeit ebenfalls berechnen.
Verbrauch [l/h] =MAF / 14.7 * 0.72 *3.6

5.4.7. Gesamtverbrauch

Der Gesamtverbrauch wird über den aktuellen Verbrauch in Litern pro Stunde berechnet. Da jede Sekunde der Verbrauch berechnet wird, wird angenommen, dass dieser Verbrauchswert für genau eine Sekunde lang gültig ist. Dies ergibt eine Menge an Treibstoff der in dieser Sekunde verbraucht wird. Durch aufsummieren all dieser Verbrauchswerte, erhält man den Gesamtverbrauch der Fahrt.

5.4.8. Distanz

Die zurückgelegte Distanz des Trips wird ähnlich wie der Gesamtverbrauch berechnet. Es wird angenommen, dass die erfasste Geschwindigkeit für genau eine Sekunde lang gültig ist. In dieser Sekunde wird eine gewisse Distanz zurückgelegt. Durch Aufsummierung dieser einzelnen Distanzen erhält man die Gesamtdistanz der Fahrt.

6. Realisierung

6.1. Einleitung

Im Realisierungsteil sind die wichtigsten Artefakte und Software-Lösungen aufgezeigt. Beginnend mit der Domainanalyse, woraus die Datenhaltungslogik hervorgeht. Welche Informationen die EcoHelper-Applikation persistiert und verarbeitet. Die gesamte Applikationslogik wird anhand eines Klassendiagramms aufgezeigt. Ein Zustandsdiagramm zeigt den Zustand der Software im Datenaustauschzyklus mit dem Wifi-Adapter. Im Unterkapitel "Implementierung" sind die wichtigsten Code-Ausschnitte ersichtlich welche die Implementierung des Datenaustausches mit dem Wifi-Adapter aufzeigen.

6.2. Domainanalyse

Das Domain Modell [Abbildung 16] zeigt die Struktur der EcoHelper Applikation. Zudem ist es gleichzeitig die Vorlage des Entity Relationship Models (ERM) für die Datenbank. Die verschiedenen Informationsinstanzen und deren Beziehungen werden in der Konzeptbeschreibung unterhalb der Grafik genauer ausgeführt und erklärt.

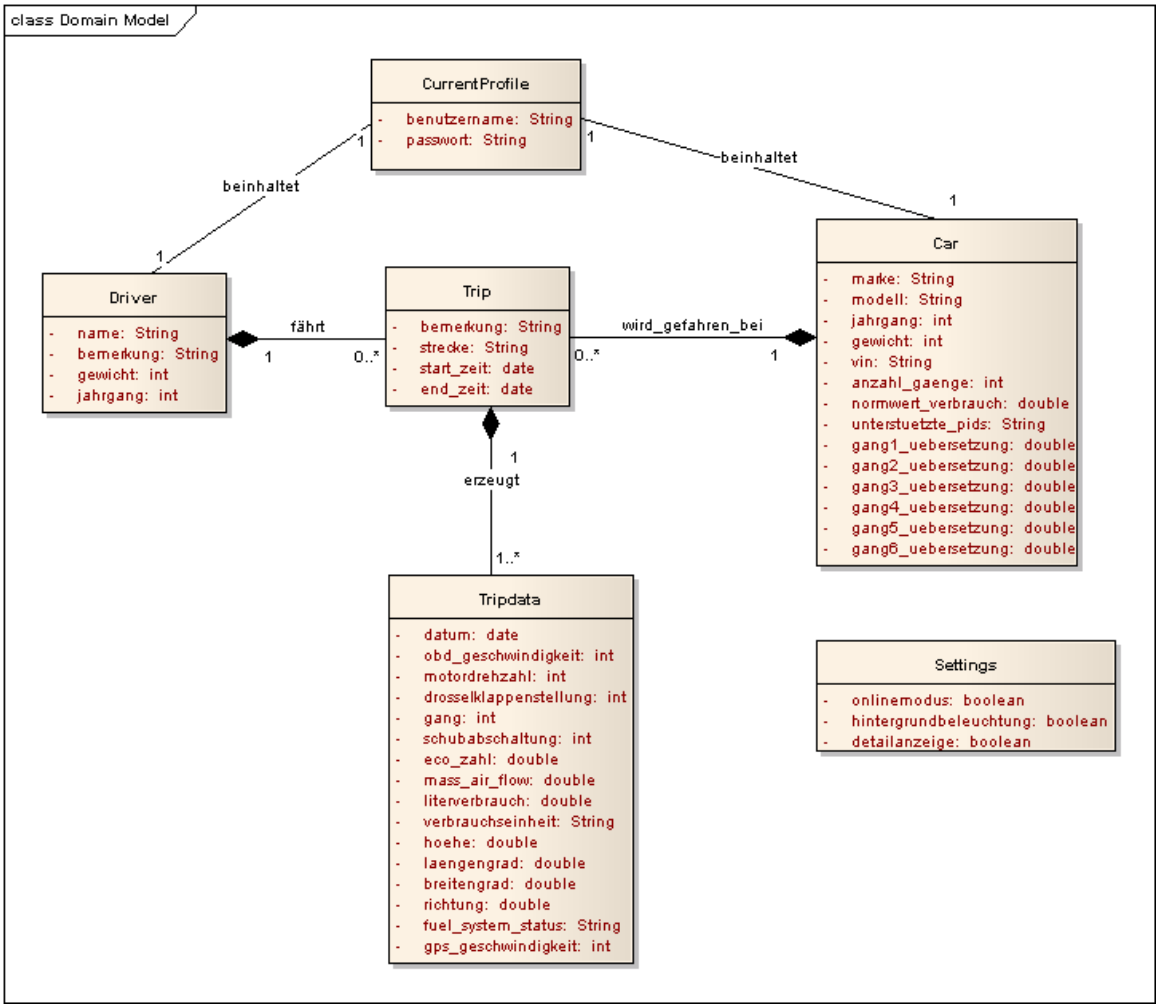


Abbildung 16 Domain Modell

6.2.1. Konzeptbeschreibung

1	Driver	
Beschreibung	Der zu testende Fahrer und evtl. auch User unserer Applikation.	
Attribute	name : String	Name des Fahrers.
	gewicht: int	Gewicht des Fahrers.
	jahrgang: int	Geburtsjahr des Fahrers.
	bemerkung: String	Zusätzliche Bemerkungen zum Fahrer.
Beziehungen	Er kann einen Trip fahren und im Profil als aktueller Fahrer eingestellt werden.	

2	Car	
Beschreibung	Das gefahrene Auto.	
Attribute	marke: String	Automarke.
	modell: String	Automodell.
	jahrgang: int	Jahr der ersten Inverkehrssetzung des Autos.
	gewicht: int	Gewicht des Autos.
	vin: String	Fahrzeugidentifikationsnummer (Vehicle Identification Number).
	anzahl_gaenge: int	Anzahl Gänge des Autos.
	normwert_verbrauch: double	Der durchschnittliche Normverbrauch des Autos (Information meist in der Betriebsanleitung des Autos.
	unterstuetzte_pids: String	Die unterstützten Fahrparameter (PID) in hexadezimaler Darstellung (Bit-Darstellung ergibt von links nach rechts die unterstützten PID's. Bit = 1 unterstützt und Bit = 0 nicht unterstützt.
	gang1_uebersetzung: double	Übersetzungsverhältniszahl für den 1. Gang.
	gang2_uebersetzung: double	Übersetzungsverhältniszahl für den 2. Gang.
	gang3_uebersetzung: double	Übersetzungsverhältniszahl für den 3. Gang.
	gang4_uebersetzung: double	Übersetzungsverhältniszahl für den 4. Gang.
	gang5_uebersetzung: double	Übersetzungsverhältniszahl für den 5. Gang.
	gang6_uebersetzung: double	Übersetzungsverhältniszahl für den 6. Gang.

Beziehungen	Das Auto wird bei einem Trip als Fahrzeug verwendet und kann im Profil als aktuelles Auto eingestellt werden.

3	**Trip**	
Beschreibung	Die ausgeführte Autofahrt.	
Attribute	bemerkung: String	Zusätzliche Bemerkungen zur Autofahrt.
	strecke: String	Bezeichnung der Fahrtstrecke.
	start_zeit: date	Startzeit der Autofahrt.
	end_zeit: date	Endzeit der Autofahrt.
Beziehungen	Die Autofahrt wird durch einen Fahrer in einem Auto ausgeführt. In der Tabelle TripData, wo Fahrparameter gespeichert werden, wird zusätzlich immer auf ein Trip referenzier.	

4	TripData	
Beschreibung	Aufgezeichnete Autofahrtzeilen mit verschiedenen direkt ausgelesenen und berechneten Fahrparametern.	
Attribute	datum: date	Startdatum & -zeit der aufgenommenen Autofahrtzeile.
	obd_geschwindigkeit:	Vom OBD ausgelesene Geschwindigkeit.
	motordrehzahl: int	Gemessene Motordrehzahl.
	drosselklappen-stellung: int	Gemessene Drosselklappenstellung. Im direkten Zusammenhang mit der Pedalstellung.
	gang: int	Ausgerechneter Gang.
	schubabschaltung: int	Schubabschaltung (Keine Benzineinspritzung, voraussichtliche Fahrweise).
	eco_zahl: int	Berechnete Eco-Zahl sagt aus wie effizient mit der Formel (Geschwindigkeit * Gewicht / Verbrauch) gefahren wird.
	mass_air_flow: double	Gemessener Luftdurchfluss. Zusammenhängend mit dem Literverbrauch.
	fuel_system_status: String	Dient als Zusatzinformation für die Berechnung der Schubabschaltung.
	literverbrauch: double	Berechneter Verbrauch in Liter/ 100km in fahrendem Zustand sonst in Liter/ h.
	verbrauchseinheit: String	Informiert, welche Verbrauchseinheit (Liter/ 100km oder Liter/ h) persistiert wurde.
	hoehe: double	Vom GPS ausgelesene Höhe.
	laengengrad: double	Vom GPS ausgelesenes Längengrad.
	breitengrad: double	Vom GPS ausgelesenes Breitengrad.
	richtung: double	Vom GPS ausgelesene Richtung.
	gps_geschwindigkeit: int	Vom GPS ausgelesene Geschwindigkeit.

Beziehungen	Die Autofahrtzeile enthält alle wichtigen Fahrparameter (direkt ausgelesen oder berechnet) und sie enthält jeweils eine Referenz auf eine bestimmte Autofahrt und besitzt somit auch eine Referenz auf das Auto und den Fahrer.

5	**CurrentProfile**	
Beschreibung	Aktuell eingestelltes Profil.	
Attribute	benutzername: String	Benutzername für das Senden der Daten an den TourLive-Server.
	passwort: String	Passwort für das Senden der Daten an den TourLive-Server.
Beziehungen	Das aktuelle Profil hat eine Referenz auf ein Auto und einen Fahrer. Wenn diese 4 Parameter (Benutzername, Passwort, Auto, Fahrer) gesetzt sind, dann ist das Profil komplett.	

6	**Settings**	
Beschreibung	Diverse Einstellungen.	
Attribute	onlinemodus: boolean	Modus, ob Fahrdaten während der Datenaufnahme zum Server geschickt werden sollen oder nachher.
	hintergrundbeleuchtung: boolean	Variable, ob Hintergrundbeleuchtung während der Applikationslaufzeit bleiben soll oder nicht.
	standardanzeige: date	Einstellung, ob standardmässig auf der Anzeigesicht nach Aufnahmestart sämtliche Werte oder nur einige angezeigt werden sollen.
Beziehungen	Keine Beziehungen zu den anderen Entitäten im Domain Modell. Wird jedoch in der Applikationsebene mehrmals referenziert, um die entsprechenden Einstellungen zu laden und zu setzen.	

6.3. Software Architektur

6.3.1. Overview Klassendiagramm

Das nachstehende Package Diagramm zeigt das Zusammenspiel zwischen den einzelnen Schichten. Es repräsentiert die 3-Schichten-Architektur, das aus GUI (Benutzeroberfläche), PD (Business Logik) und Data (Datenhaltung) besteht. Die Abhängigkeiten gehen immer vom oberen Package strikt auf das untere Package. Mittels dieser Applikationsstrukturierung ist gewährleistet, dass die einzelnen Software-Module besser voneinander entkoppelt sind.

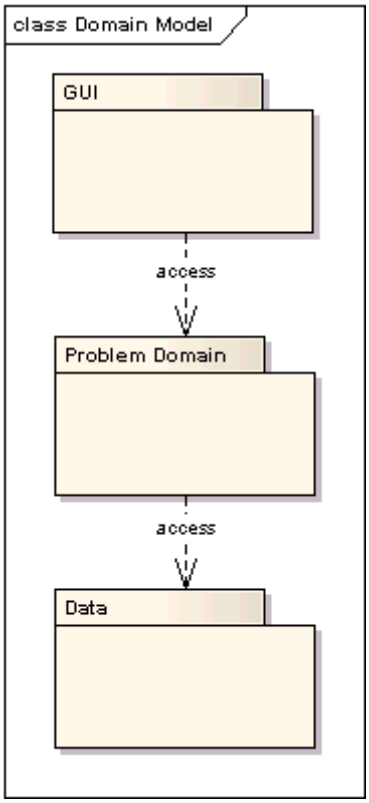


Abbildung 17 Vereinfachtes Domain Modell

6.3.2. Systemstruktur

6.3.2.1. Physische Schicht

Die Architektur der EcoHelper-Applikation ist eine 3-Schichten-Architektur. Hauptgrund zur Wahl dieser Architektur ist, dass mehrschichtige Systemarchitekturen wie die dreischichtige Architektur gut skalierbar sind, da die einzelnen Schichten logisch voneinander getrennt werden können.

Presentation-Tier Der Top-Most Level der Applikation ist das User Interface. Hauptfunktionalitäten des Interfaces sind Eingaben und Ausgaben so zu übersetzten, dass der Anwender sie ohne grosse Mühe versteht.	 
Logic-Tier Dieser Layer koordiniert die Applikationsprozesse, erstellt logische Entscheidungen und Evaluationen und berechnet diese. Es bewegt auch Daten bzw. Prozesse zwischen den umgebenden Layers.	 
Data-Tier Hier werden Informationen in der Datenbank gespeichert. Informationen werden auch von hier ausgelesen und den oberliegenden Tiers übergeben.	 

Tabelle 3 Physische Schicht

6.3.2.2. Logische Schicht (Klassendiagramm)

Auf ein Klassendiagramm des gesamten Designs mit allen Variabeln und Methoden wurde verzichtet. Mit ca. 50 Klassen wäre das Diagramm viel zu unübersichtlich. Hilfreich um die Zusammenhänge zu finden, ist die nachstehende Design-Beschreibung des Klassendiagramms.

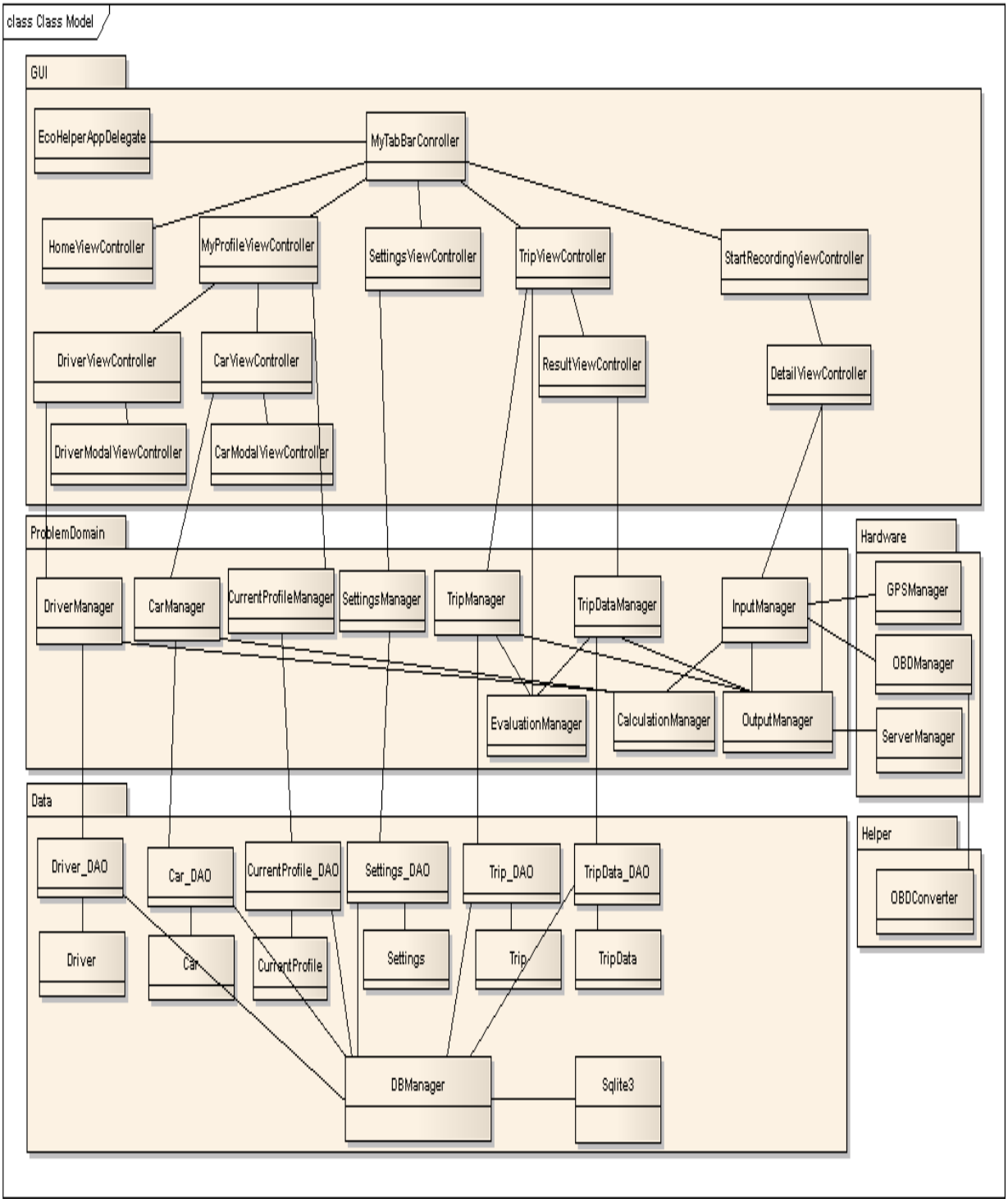


Abbildung 18 Logische Schicht

6.3.2.3. Design-Beschreibung

GUI

Das GUI-Package beinhaltet die wichtigsten Klassen der Benutzeroberfläche, die ViewController und die dahinterliegenden Views. Jeder ViewController ist nach dem Model-View-Controller Pattern jeweils für seine eigene View zuständig. Er nimmt etwa die Benutzeraktionen entgegen, delegiert diese an die Model (Probem-Domain- und Data-Schicht), nimmt deren Verarbeitungsresultate entgegen und zeigt diese wiederum in der View an.

ProblemDomain

Dieses Package bildet einen Teil des Models im Model-View-Controller Pattern. In dieser Schicht-Ebene sind die Managerklassen zuhause. Einige Managerklassen sind als Vermittler zwischen Data- und GUI-Schicht zuständig und bieten eine einfach anzusprechende Schnittstelle an, um auf die Datenbank zuzugreifen. Die anderen Managerklassen, welche keine direkte Verbindung zur Data-Schicht haben, sind grundsätzlich für die Datensammlung (InputManager, GPSManager, OBDManager), Datenberechnung (OBDConverter, CalculationManager, EvaluationManager) und für die Datenweiterverarbeitung (OutputManager) zuständig.

Der InputManager bekommt die Daten vom OBD- und GPSManager. Diese delegiert er an den OutputManager. Hier werden die Daten persistiert (TripManager, TripDataManager) wieder angezeigt (DetailViewController) und an einen Webserver gesendet (ServerManager).

Data

Diese Schicht enthält die Komponenten zur Persistierung der Daten die in der Applikation anfallen. Die DAO-Klassen bilden zusammen die Datenzugriffsobjekte (Data Access Object Pattern). Diese Klassen haben eine direkte Verbindung zur Datenbank (DBManager, Sqlite3) und bieten mit der implementierten Schnittstelle von aussen her einen komfortablen Zugriff auf die Datenbank.

Hardware

Die verantwortlichen Klassen für die Datenaufnahme von den Fahrparameter (OBD) und den Standortdaten (GPS) befinden sich im Package Hardware. In Hardware, weil die Datenquellen und -senken in der Realität wirkliche Geräte darstellen wie z.B. das iPhone für die GPS-Datenlieferung (GPSManager), der Wifi-Adapter für die OBD-Datenlieferung (OBDManager) oder der Web-Server für den Empfang der Daten (ServerManager).

Helper

Da die Fahrparameterwerte in hexadezimaler Darstellung geliefert werden und noch speziell umtransformiert werden müssen, befindet sich in dieser Schicht eine Klasse namens OBDConverter, die sich dieser Aufgabe widmet.

Nach dieser Implementierung ist gewährleistet, dass die Abhängigkeiten klar von den oberen Schichten zur jeweils unteren Schicht gehen.

Eine detailliertere Beschreibung der wichtigsten Klassen befinden sich im Softwarearchitektur-Dokument [E14].

6.3.2.4. Datenbankschicht (Entity Relationship Model)

SQLite ist eine Programmbibliothek, die ein relationales Datenbanksystem enthält. Die Datenbankschicht zeigt alle Entitäten inklusive Variabeln und dessen Typen auf. Auch sind die Verbindungen zwischen den einzelnen Entitäten und die Kardinalitäten ersichtlich. Mit dem SQLite-Framework für das IPhone ist folgende Datenbank entstanden:

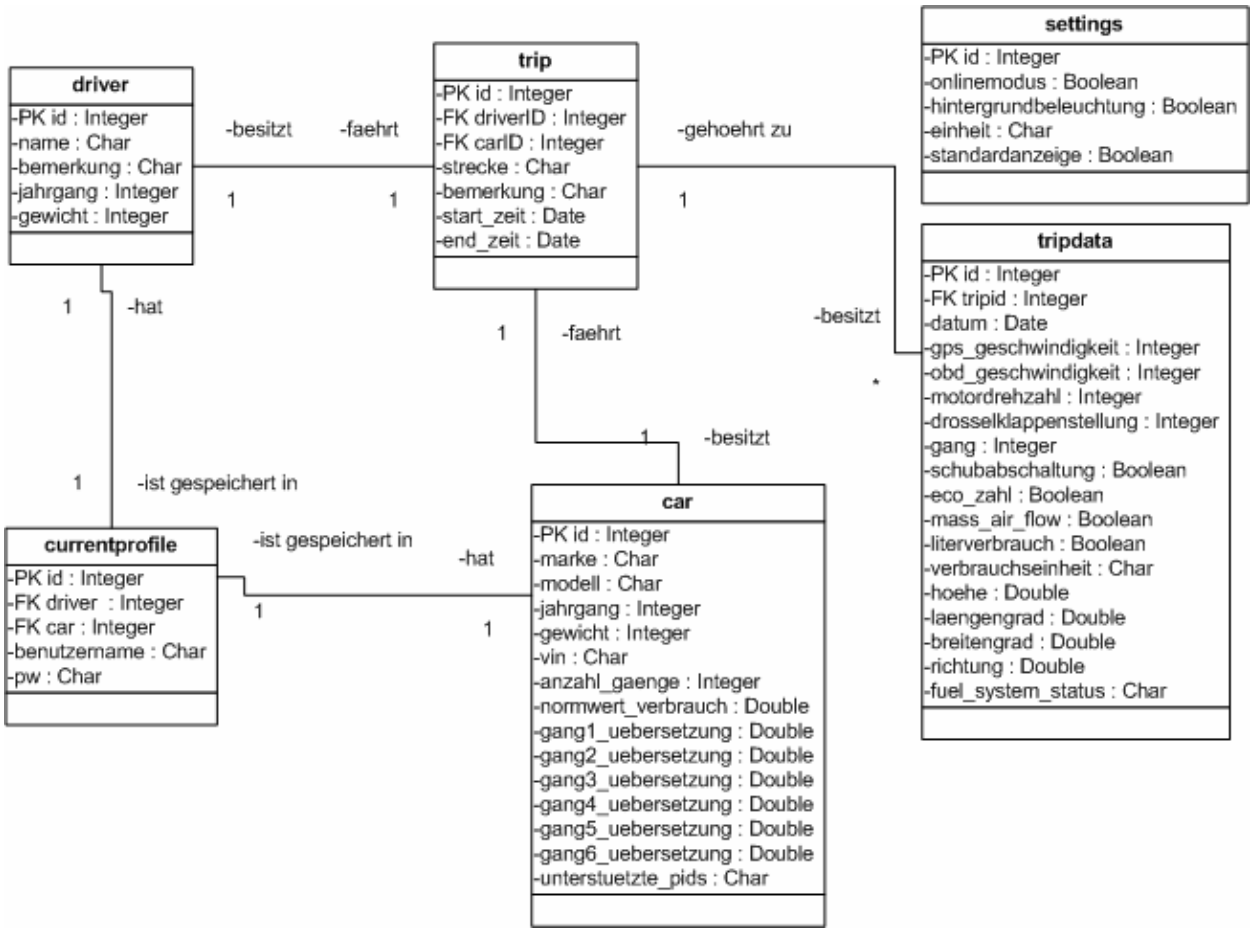


Abbildung 19 Datenbankschicht

Die Konzeptbeschreibung des Domain Modells beschreibt gleichzeitig auch den Aufbau des ERM [Abbildung 20] bis auf die zusätzlichen Schlüsselfelder für die Identifikationen und Relationen.

6.3.3. Sequenzdiagramm (Fahrtaufnahme starten)

Diese Sequenz wird erzeugt, sobald der User eine Fahrtaufnahme startet. Eine nähere Beschreibung befindet sich auf der nächsten Seite.

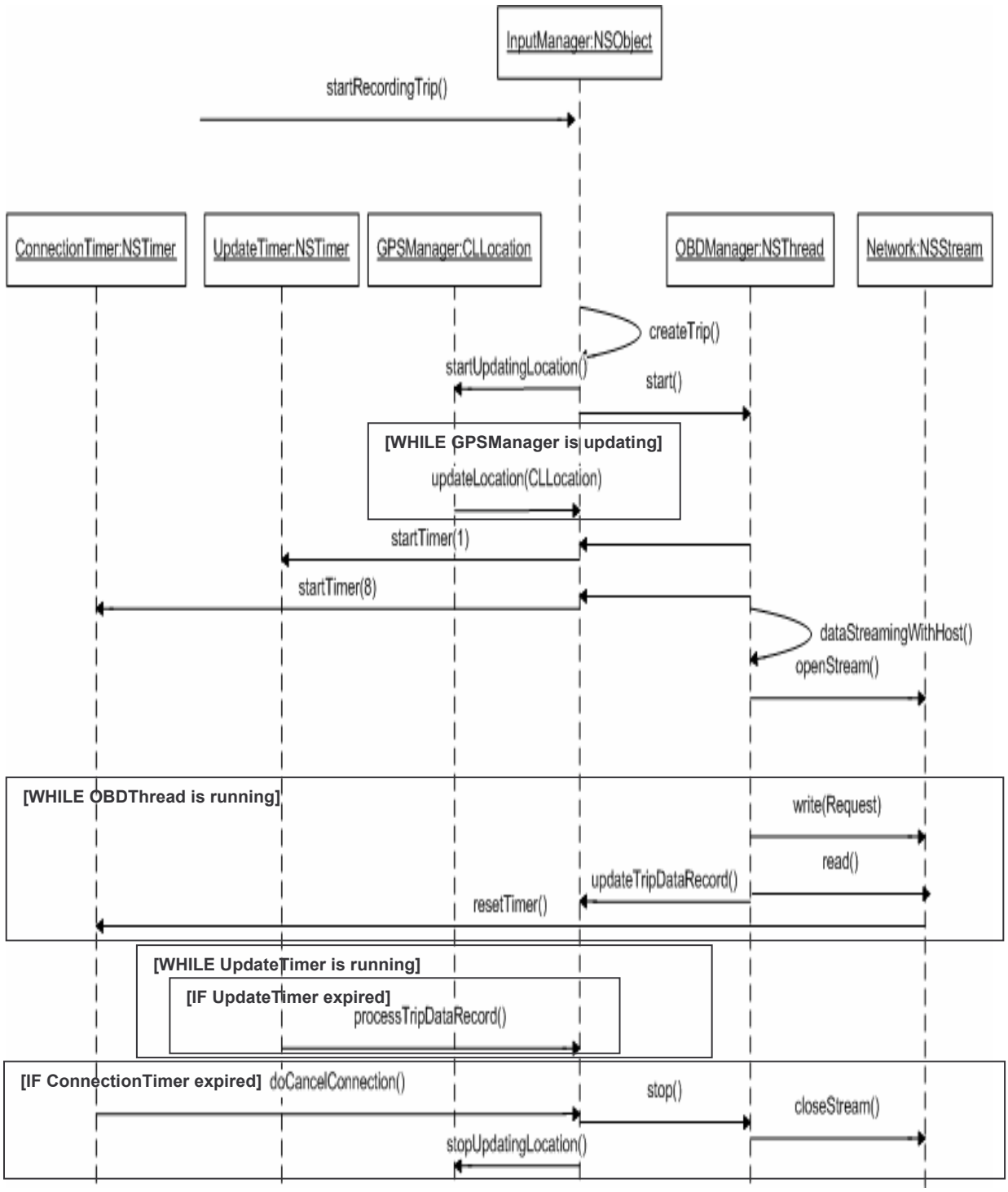


Abbildung 20: Sequenzdiagramm: Fahrtaufnahme starten

Nachdem der User eine Fahrtaufnahme gestartet hat, wird sich der InputManager darum kümmern einen neuen Trip zu erzeugen. Danach startet er die Ortaktualisierung des GPSManagers und den OBDManager-Thread.

Jedesmal wenn eine Ortaktualisierung stattfindet fängt der InputManager den Event "upateLocation(CLLocation)" ab und aktualisiert die Daten.

Der OBDManager-Thread startet eine Netzwerkverbindung mit dem WLAN-Adapter und gibt die Verantwortung für die Verarbeitung der empfangenen Fahrparameter dem InputManager. Die gemeinsam, kritische Datenzone ist "atomic" und somit thread-sicher. Ein Deadlock ist nicht möglich, da aufgrund der einzig, gemeinsamen Ressource die "hold and wait"-Bedingung nicht erfüllt ist.

Damit Netzwerkprobleme detektiert werden können, wird ein Timer installiert, der die Verbindung abbricht wenn 4 Sekunden lang keine Antwort vom WLAN-Adapter kommt. So wird verhindert, dass der OBDManager-Thread blockiert. Jedesmal, wenn eine Antwort zurückkommt wird der Verbindungstimer zurückgesetzt.

Ein zweiter Timer ist für die Weiterverarbeitung der aktualisierten Fahrtdaten verantwortlich. In der Weiterverarbeitung werden die aktualisierten Daten im Sekundentakt persistiert, angezeigt und an den Webserver gesendet.

Bei einem Verbindungsabbruch, ob durch einen Fehler oder gewollt durch den User, werden alle Timer deinstalliert, die Ortaktualisierung des GPSManagers und der OBDManager-Thread gestoppt.

6.3.4. Zustandsdiagramm (Zustandsmaschine)

Das folgende Zustandsdiagramm beschreibt die verschiedenen Zustände, welche die EcoHelper-Applikation einnehmen kann während dem Datenaustausch mit dem WLAN-Adapter.

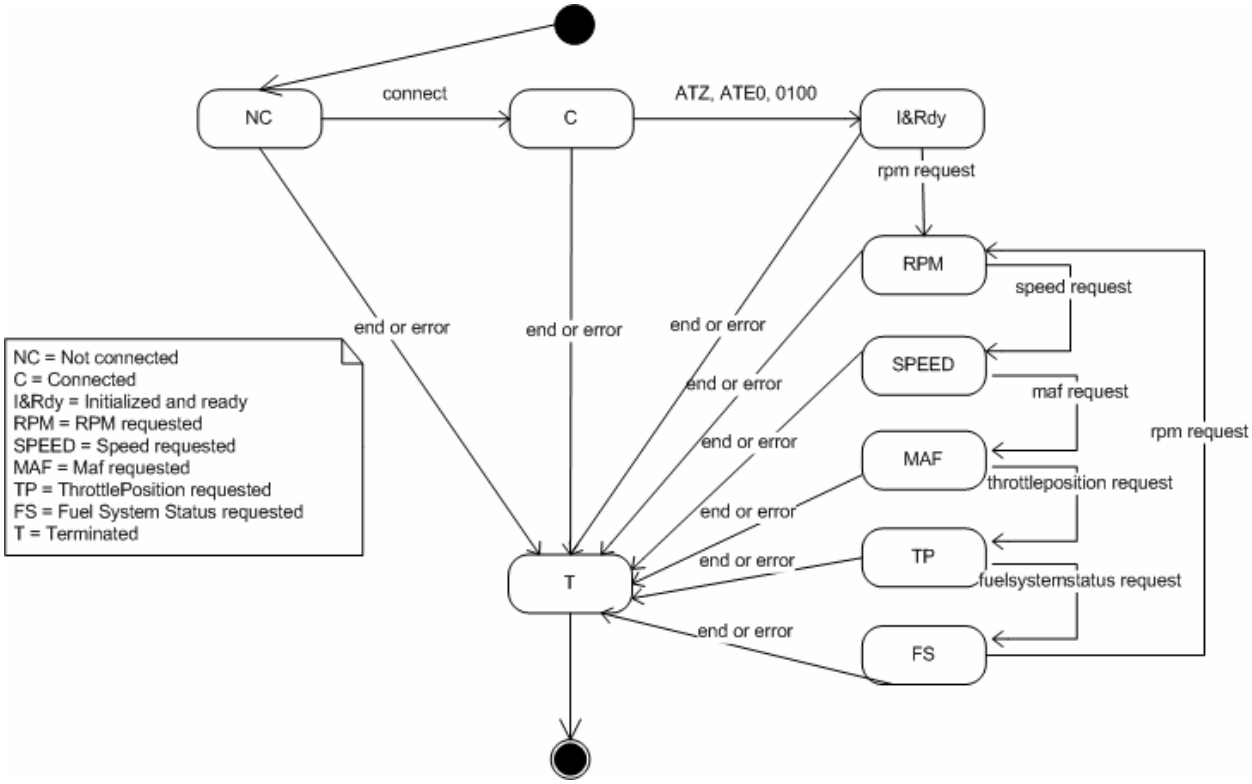


Abbildung 21: Zustandsdiagramm: Zustandsmaschine (WLAN-Kommunikation)

Zuerst ist die Applikation im Zustand "Not connected". Nach der erfolgreichen Verbindungsherstellung ist sie im Zustand "Connected". Jetzt schickt die Applikation dem WLAN-Adapter die Initialisierungsanfragen und erreicht somit den Zustand "Initialized and ready".

Ab hier werden die einzelnen Fahrparameter angefragt wie RPM (Motordrehzahl), SPEED (Geschwindigkeit), MAF (Mass Air Flow, Luftdurchfluss), TP (Throttle Position, Gaspedalstellung) und FS (Fuel System Status). Nachdem ein Fahrparameter angefragt wurde, wird der Zustand "Fahrparameter angefragt" erreicht und jedesmal der entsprechend nächste Fahrparameter angefragt.

Von jedem Zustand aus gibt es einen Zutandsübergang zu T (Terminated) bei Aufkommen von Verbindungsfehlern oder durch explizite Verbindungsbeendigung durch den User.

6.3.5. Architektur der Interaktion

6.3.5.1. MVC (Model-View-Controller) Pattern

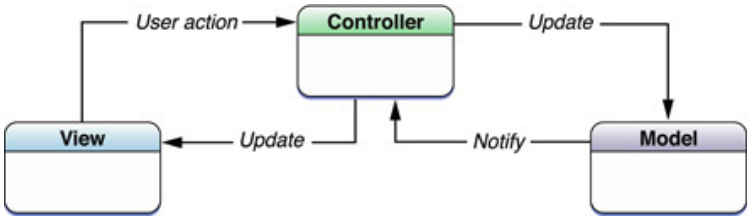


Abbildung 22 MVC Pattern

Das MVC Muster bestimmt in der heutigen IPhone-Entwicklung die Software-Architektur der IPhone-Applikationen. Die EcoHelper-Applikation basiert ebenfalls auf diesem Muster. Alle Views bilden zusammen die Benutzeroberfläche. Jede View hat ihren eigenen Controller (ViewController), der die entsprechenden Benutzeraktionen entgegennimmt, diese verarbeitet und die eigene View wieder mit den Resultaten aktualisiert. Die Controller kapseln das Model bzw. die eigentliche Logik und Datenhaltung. Das Model bekommt die entsprechenden Verarbeitungsbefehle durch den Controller führt diese aus und aktualisiert den Controller mit den ausgerechneten Ergebnissen.

6.3.5.2. Delegate Pattern

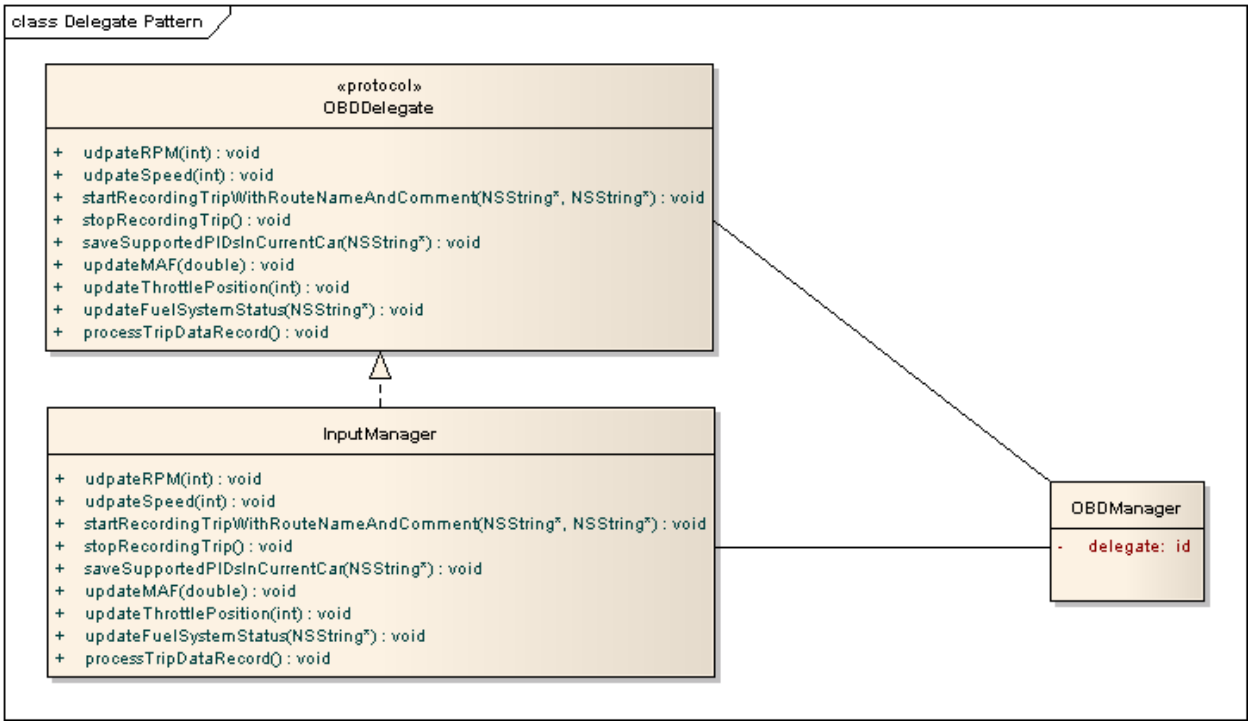


Abbildung 23 Delegate Pattern

Das Delegate Muster wird in der Entwicklungssprache des IPhone (Objective-C) unterstützt und in der EcoHelper-Applikation benutzt. Mit diesem mächtigen Muster kann man in einer Klasse eine Referenz auf ein Protokoll bzw. Interface deklarieren. Zur Laufzeit kann man dann bei der Referenzdefinition die Referenz auf ein bestimmtes Objekt, welches dieses Protokoll bzw. Interface implementiert, umzeigen lassen. Es wird ein Upcast gemacht, wodurch dann direkt

das eigentliche Objekt referenziert wird und somit den vollständigen Zugriff auf seine Methoden gewährleistet wird. Im Fall der EcoHelper-Applikation (wie im obigen Bild) hat der OBDManager eine Referenz auf das Interface „OBDDelegate“. Der OBDManager muss nur wissen, welche Methoden des Protokolls OBDDelegate er aufrufen kann, nicht aber wie diese implementiert sind. Mit dem Setzen des Delegates kann man die Verantwortung von Methoden-Abarbeitungen bestimmten Objekten zuteilen, wie in diesem Fall dem „InputManager“.

6.3.5.3. Singleton Pattern

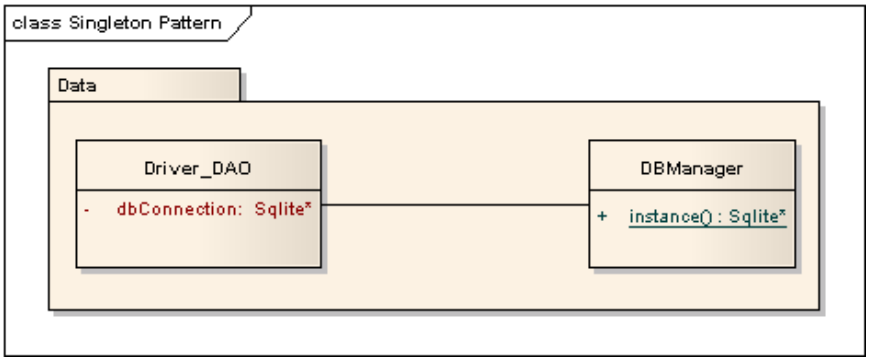


Abbildung 24 Singleton Pattern

Das Singleton Muster wird in der EcoHelper-Applikation in dem Sinn benutzt, dass nur eine Datenbank-Verbindung mittels der statischen „instance“-Methode instanziert und zurückgegeben wird. Die Data-Access-Object-Klassen haben von überall aus Zugriff auf die Datenbankverbindung.

6.3.5.4. Facade Pattern

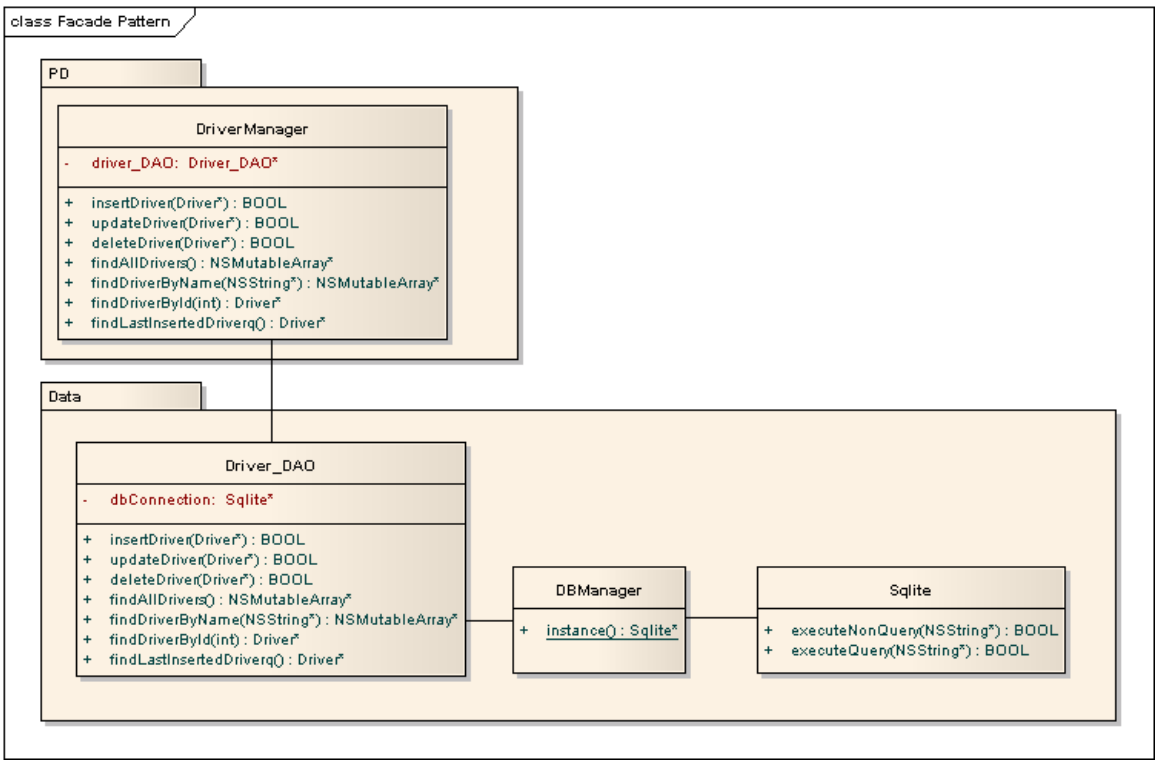


Abbildung 25 Facade Pattern

Das Facade Muster sagt aus, dass im Prinzip eine komplexe Klassenstruktur in einer oder wenigen Klassen gekapselt werden und entsprechende Methoden für den Zugriff und die Manipulation der Daten zur Verfügung gestellt werden. In der EcoHelper-Applikation entspricht die Wrapper-Klasse Sqlite3 diesem Muster. Während die Wrapper-Klasse eine komplexe Datenbankimplementierung beinhaltet, stellt sie nach aussen doch nur komfortable Methoden wie „executeQuery:(NSString *) selectStatement“ und „executeNonQuery:(NSString *) insertUpdateDeleteStatement“ zur Verfügung. Mithilfe der Data-Access-Objekt-Klassen ist diese Komplexität in eigenen Methoden gekapselt und bietet nach aussen einen komfortablen schreibenden sowie auch lesenden Zugriff auf die Datenbank.

6.4. Implementierung

Im Implementierungsabschnitt ist der wichtigste UseCase "Fahrtaufnahme starten" bzw. der Programmierteil mit dem Datenaustausch zwischen Applikation und Wifi-Adapter aufgezeigt.

Folgende Codeausschnitte werden ausgeführt, wenn der User eine Fahrtaufnahme startet.

6.4.1. Code-Ausschnitte

Sobald der User "Aufnahme starten" drückt, ist der StartRecordingViewController für die Reaktion auf das Knopfdrücken zuständig und implementiert dafür eine sogenannte IBAction-Methode namens "startRecording". Wenn die Vollständigkeitsprüfung des eingestellten Profils in Ordnung ist, wird die Ansicht "DetailViewController" initialisiert. Ist das Profil nicht vollständig, wird der User benachrichtigt und keine Aufnahme gestartet bis das Profil vollständig ist.

Implementierungsdatei StartRecordingViewController.m:

```
- (IBAction) startRecording {
    currentProfileManager = [[CurrentProfileManager alloc] init];
    CurrentProfile *currentProfile = [currentProfileManager getProfile];
    if(DriverManager.driver == nil || CarManager.car == nil || [currentProfile.username isEqualToString:@""] || [currentProfile.password isEqualToString:@""])
        NSString *message = [[NSString alloc] initWithFormat:@"Bitte vervollständigen Sie zuerst Ihr Profil!"];
        UIAlertView *emptyDataAlert = [[UIAlertView alloc]
                                       initWithTitle:@"Profil unvollständig"
                                       message:message
                                       delegate:self
                                       cancelButtonTitle:@"Ok"
                                       otherButtonTitles:nil];
        [emptyDataAlert show];
        [emptyDataAlert release];
    } else {
        detailViewController = [[DetailViewController alloc] initWithNibName:@"DetailView" bundle:nil];
        [[detailViewController inputManager] startRecordingTripWithRouteName:[routeTextField text] AndComment:[commentTextField text]];

        EcoHelperAppDelegate *ecoHelperAppDelegate = [[UIApplication sharedApplication] delegate];
        [ecoHelperAppDelegate.rootController presentModalViewController:detailViewController animated:YES];
    }
}
```

Abbildung 26: Methode "startRecording" aus StartRecordingViewController.m

In Objective-C gibt es keine Konstruktoren, sondern sogenannte init-Methoden, wo der Initial-Zustand der verschiedenen Objekte gesetzt werden kann. So auch in der init-Methode vom DetailViewController. Wichtig in dieser Methode ist die Initialisierung des Klassenobjektes InputManager, der später die Datensammlung ausführt.

Implementierungsdatei DetailViewController.m:

```
- (id)initWithNibName:(NSString *)nibNameOrNil bundle:(NSBundle *)nibBundleOrNil {

    [self defineBorderValues];

    inputManager = [[InputManager alloc] init];
    inputManager.outputManager.delegate = self;

    return [super initWithNibName:nibNameOrNil bundle:nibBundleOrNil];
}
```

Abbildung 27: Methode "initWithNibName" aus DetailViewController.m

In der Init-Methode vom InputManager werden wichtige Klassenobjekte initialisiert, welche die effektive Datensammlung über WLAN/ GPS und die Datenweiterleitung an die Ansicht, die Datenbank und an den Webserver vornehmen.

Implementierungsdatei InputManager.m

```
- (id) init {
    tripManager = [[TripManager alloc] init];
    trip = [[Trip alloc] init];

    tripDataManager = [[TripDataManager alloc] init];
    tripDataRecord = [[TripData alloc] init];

    carManager = [[CarManager alloc] init];

    gpsManager = [[GPSManager alloc] init];
    gpsManager.delegate = self;

    obdManager = [[OBDManager alloc] initWithTarget:self selector:@selector(obdThreadWorker) object:nil];
    obdManager.delegate = self;

    outputManager = [[OutputManager alloc] init];

    calculationManager = [[CalculationManager alloc] init];

    return [super init];
}
```

Abbildung 28: Methode "init" aus InputManager.m

Der Methodenaufruf „init“ des InputManagers in DetailViewController.m erzeugte die Ausführung des vorangegangenen Codes. Nach der Vorbereitung der Daten, startet der Methodenaufruf von startRecordingTripWithRouteName:(NSString*)AndComment:(NSString*) wiederum die effektive Fahrtaufnahme.

Der LocationManager des GPSManagers startet mit der Aktualisierung der Lokalisierungsdaten und triggert bei jedem Update einen wichtigen Event-Callback (locationUpdate:CLLocation*), wo die aktualisierten GPS-Daten gespeichert werden.

Der OBDManager ist ein Thread und wird hier vom MainThread abgespaltet, da die Netzwerkkommunikation über WLAN mehrere Fahrparameter pro Sekunde abfragt. Bei jedem Methodenaufruf wie z.B. "updateSpeed:(int)speedParam", der entsprechend mehrmals pro Sekunde ausgeführt wird, würde den MainThread blockieren und somit auch den User in seiner Flexibilität betreffend der Benutzeroberflächeninteraktion einschränken. Mit der Zuweisung des Delegates vom OBDManager, gewährleistet man die Referenzierung auf den InputManager und den Zugriff auf dessen Methoden für die Aktualisierung der ausgelesenen Werte.

Implementierungsdatei InputManager.m:

```
- (void) startRecordingTripWithRouteName:(NSString *) routeNameParam AndComment:(NSString *) commentParam {
    trip.routeName = routeNameParam;
    trip.comment = commentParam;
    [self prepareTripForRecording];

    [[gpsManager locationManager] startUpdatingLocation];

    [obdManager start];
}
```

Abbildung 29: Methode "startRecordingTripWithRouteName" aus InputManager.m

Der separat aufgerufene Thread-Kontext ist die Methode "threadWorker". Es wird ein Zähler ConnectionTimeoutTimer installiert, der nach 8 Sekunden abläuft, wenn bis dahin keine Antwort vom Netzwerk gekommen ist. Somit wird ein Verbindungsabbruch erzeugt. Der UpdateTripDataTimer wird installiert und ruft im Gegensatz zum ersten ConnectionTimeoutTimer wiederholend im 1 Sekundentakt die Methode "processTripDataRecord" auf, die gespeicherte Resultate weiterverarbeitet. Die Methode "dataStreamingWithHost:(NSString*)andPort:(int)" führt die effektive Netzwerkkommunikation aus.

Implementierungsdatei InputManager.m:

```
- (void) obdThreadWorker {
    NSAutoreleasePool* pool = [[NSAutoreleasePool alloc] init];

    [self installConnectionTimeoutTimer];
    [self installUpdateTripDataTimer];
    [obdManager dataStreamingWithHost:connectionIP andPort:connectionPort];

    [pool release];
}
```

Abbildung 30: Methode "obdThreadWorker" aus InputManager.m

Der folgende Code-Ausschnitt zeigt die Netzwerkkommunikation zwischen dem EcoHelper und dem WLAN-Adapter nach erfolgreichen Verbindungsaufbau. Mit der int-Variable "currentFlowState" ist die Realisierung einer Zustandsmaschine möglich. So wird immer der Zustand gespeichert, was als nächstes gesendet werden muss und so werden die Fahrparameter immer nacheinander abgefragt und per Referenz im InputManager auf dem MainThread verarbeitet. Im Falle eines Verbindungsabbruches wird die Bedingungsvariable "shouldKeepRunning" auf false gesetzt und die Verbindung ausserhalb der while-Schleife geschlossen.

Implementierungsdatei InputManager.m:

```
currentFlowState = STATE_SEND_REQUEST_RESETDEVICE;

while(shouldKeepRunning) {
    if(currentFlowState == STATE_SEND_REQUEST_RESETDEVICE) {
        ...}
    else if(currentFlowState = STATE_SEND_REQUEST_RPM) {
        while(![outputStream hasSpaceAvailable] && cancelConnection == false);
        if(cancelConnection) {
            break;
        } else {
            [self resetConnectionTimeoutTimer];
        }
        [outputStream write:requestRPM maxLength:5];
    } else if(currentFlowState ...) {} ...


    while(![inputStream hasBytesAvailable] && cancelConnection == false);
    if(cancelConnection) {
        break;
    } else {
        [self resetConnectionTimeoutTimer];
    }

    len = [inputStream read:buffer maxLength:sizeof(buffer)];
    response = [[NSString alloc] initWithBytes:buffer length:len encoding:NSASCIIStringEncoding];
    response = [response stringByReplacingOccurrencesOfString:@" " withString:@""];


    switch (currentFlowState) {
        case STATE_SEND_REQUEST_RESETDEVICE:
            currentFlowState = STATE_SEND_REQUEST_RPM;
            break;
        case ...
        case STATE_SEND_REQUEST_RPM:
            if([self isResponse: response ForRequestedPID : @"0C"]) {
                rpmInHex = [response substringWithRange:rpmRange];
                rpm = [obdConverter getRPM:rpmInHex];
                [delegate performSelectorOnMainThread:@selector(updateRPM:) withObject:[NSNumber numberWithInt:rpm] waitUntilDone:NO];
            }
            currentFlowState = STATE_SEND_REQUEST_SPEED;
            break;
        case ...
    }
}

[self closeStream];
```

Abbildung 31 Methondenausschnitt „dataStreamingWithHost“ aus OBDManager.m

7. Schlussfolgerungen

Was wurde erreicht?

In der Analysephase wurden die verschiedensten Kombinationen von Parametern, die man über das OBD oder das GPS erhält, untereinander verglichen. Die daraus gewonnen Erkenntnisse und Resultate wurden entsprechend in der Applikation berücksichtig. Auch musste ein fundiertes Wissen über die Technologie des OBD-Protokolls und dessen Verfahren erarbeitet werden. Die vielen Testfahrten führten zudem dazu, dass das Fahrverhalten dank der neu erworbenen Kenntnisse verbessert wurde.

Die Version 1.0 der iPhone-EcoHelper Applikation besitzt alle für das Fahrtraining wichtigen Funktionen der bereits existierenden Symbian-Lösung. Lediglich die Kamerafunktion konnte nicht implementiert werden, weil das aktuelle iPhone OS 3.1.3 leider kein Multitasking unterstützt.

Die Applikation kann während der Fahrt oder nach der Fahrt die Messwerte zum Server schicken. Sie kann die Fahrdaten mittels Balken und Anzeigefelder visualisieren und während der Fahrt interaktiv präsentieren.

Aufgrund der gründlichen Analysen und der Testfahrten wurden die wichtigsten Parameter ausgewählt. Diese werden nun dem Benutzer angezeigt. Zudem kann zwischen einer einfachen und einer detaillierteren Ansicht gewechselt werden.

Am Ende jeder Fahrt kann der Benutzer die Auswertung der Fahrt nochmals am Bildschirm ansehen. Auch lassen sich die verschiedensten Parameter online auf dem Webportal anschauen und miteinander vergleichen.

Was wurde nicht erreicht?

Da das iPhone-Betriebssystem keine Parallelität unterstützt, konnten wir die Kamerafunktion nicht implementieren. Mit dieser Funktion hätte während der Fahrt in Sekundenintervallen ein Foto von der Strecke gemacht werden sollen.

Auch wollte man zu Beginn eine eigene HSR-Zahl entwickeln, die ähnlich wie die Ecozahl Auskunft über die Fahrweise gibt. Diese Zahl hätte aber eine grössere Menge Parameter als die Ecozahl miteinbezogen. Dies konnte wegen mangelnder Zeit nicht ausführlich genug ausgearbeitet werden und musste für die Version 1.0 gestrichen werden.

Vergleich mit den vorherigen Lösungen?

Die EcoHelper-Version für das iPhone bietet folgenden Mehrnutzen gegenüber den anderen Versionen:

-Es können mehrere Autos erfasst und verwaltet werden.

-Es können mehrere Fahrer erfasst und verwaltet werden.

-Jede Fahrt kann mit einem Namen und einem Kommentar versehen werden.

-Die Darstellung mittels Balken während der Fahrt erhobenen Messwerte ist verfeinert und optisch verbessert worden.

-Die Fahrdaten können nach der Erfassung optional an das Webportal gesendet werden.

-Es können mehrere Fahrten gespeichert werden.

-Statistiken von abgeschlossenen Fahrten können graphisch ausgewertet werden.

Ausblick

Apple verspricht bei der neuen OS Version 4.0 Parallelität zu unterstützen. Dies würde bedeuten, dass man dann vielleicht auch den Kameramodus in die EcoHelper-Applikation miteinbeziehen könnte.

Da mit der Version 1.0 noch keine grossen Feldtests gemacht werden konnten, werden sicher noch ein paar Änderungen angebracht werden müssen.

Denkbar wäre auch, die Bremsvorgänge genauer zu analysieren und in die Version 2.0 einfliessen zu lassen.

Die Idee kam auf, die Messwerte während der Fahrt mit akustischen Meldungen zu unterstützen.

Zunächst werden nun ausgewählte Kunden die erste Version des EcoHelpers gründlich testen. Ihr Feedback wird ausschlaggebend dafür sein, was in der Version 2.0 miteinbezogen werden sollte.

8. Persönliche Erfahrungen

8.1. Persönliche Erfahrungen von Selim Akyol

Projektverlauf
Wir hatten den Vorteil, dass dieses Projekt ein realer Auftrag mit konkreter Anwendung war. Der Realitätsbezug hat die Motivation gesteigert. Das EcoHelper-iPhone-Projekt sprengt den Rahmen einer Bachelorarbeit und wird höchstwahrscheinlich weiterbearbeitet. Bereits wurden zusätzliche Wünsche und Ideen geäussert. Auch ist klar, dass Erweiterungsmöglichkeiten bestehen. Das Team, in welchem ich eingebunden bin, hat mir sehr gefallen. Sowohl Herr Heinzmann als auch seine Mitarbeiter Omid und Raphael haben uns immer mit Rat und Tat unterstützt. Das Arbeiten mit meinem Teamkollegen und unseren Projektbetreuern hat mir grösste Freude gemacht.

Was habe ich gelernt?
Die Programmiersprache Objective-C und die OBD-Diagnoseschnittstelle des Autos waren für mich Neuland. Die Einarbeitung in diese Gebiete war für mich kein Problem, da ich mich als eingefleischter Auto- und iPhone-Fan schon länger für diese Themen interessierte.
Leider mussten wir feststellen, dass Apple bezüglich der Entwicklung von Apps sehr strenge Richtlinien erlassen hat, sehr genaue Kontrollen durchführt und wenige Zugriffe auf einzelne Komponenten des iPhone zulässt. Auch war es anstrengend, bestimmte Entwicklungsschritte direkt im Auto vornehmen zu müssen. Das Programmieren mithilfe von Simulatoren vereinfachte zwar die Arbeit, doch einige Funktionen, wie Pedalstellung und die Schubabschaltung, mussten immer wieder am Auto selber getestet werden.
Diese kleinen Nachteile haben mir die Freude an der Arbeit nicht verdorben, im Gegenteil: Die Entwicklung auf dem iPhone macht viel Spass, da man seine Software immer gleich in der Tasche mitnehmen kann.

Fazit
Die Version 1.0 für das iPhone bietet eine solide Basis für erste Tests mit ausgewählten Kunden. Denkbar wäre, dass ein paar interessierte Fahrschulen mit der ersten Version Schüler unterrichten. Auch sind uns schon viele neue Ideen gekommen, welche zusätzlichen iPhone-Erweiterung wir für den EcoHelper noch entwickeln könnten.

8.2. Persönliche Erfahrungen Edon Berisha

Projektverlauf

Da ich wusste, dass dieses Projekt nach Abschluss effektiv bei Endkunden eingesetzt würde, war meine Motivation schon von Beginn an gross. Dank der Einrichtung eines zentralen SVN-Repository, wo wir unsere Code- und Dokumentationsdateien hinterlegten, lief die Projektkoordination reibungslos und das Abändern und Hinzufügen von Daten (Code, Dokumente, etc.) führte zu keinen Arbeitskollisionen.

In den wöchentlichen Teammeetings konnten wir immer den aktuellen Zwischenstand besprechen, ausgearbeitete Ergebnisse diskutieren und einen Plan für das weitere Vorgehen erstellen. Durch die gute Organisation und Koordination des Projekts war die aktuelle Aufgabenverteilung immer gesichert und wir konnten so die nächsten Schritte besprechen. Die Kommunikation zwischen den Projektbeteiligten (Projektmitglieder, Projektleiter und Auftraggeber) funktionierte wunderbar und auf die Unterstützung des Projektleiters und Auftraggebers konnten wir immer zählen.

Ein bisschen komplizierter war dafür, einen richtigen Workaround für die Entwicklungsphase zu finden. Es stand uns zwar ein OBD-Simulator zur Verfügung, der uns Fahrparameter wie z.B. Geschwindigkeit, Motordrehzahl, Mass Air Flow, etc. lieferte. Und auch der entsprechende WLAN-Adapter war vorhanden. Was wir aber schnell gemerkt haben war, dass die Reaktionszeiten des Simulators sich von den denjenigen der richtigen Autos unterscheiden. Die erste Version mit der Grobfunktionalität lief auf dem OBD-Simulator problemlos, jedoch beim Testen mit dem Fahrzeug zeigten sich immer andere Reaktionen und so kam es auch eimal vor, dass ich in der Garage im Auto auf das Durchhaltevermögen meiner Laptopbatteriezeit angewiesen war, um diverse Codeänderungen machen und direkt am Fahrzeug testen zu können.

Was habe ich gelernt?

Den Dokumentationsaufwand für dieses Projekt habe ich unterschätzt. Also empfehle ich dringend eine fortlaufende Pflege der jeweiligen Dokumentationen und ich werde es für das nächste Mal sicherlich beachten.

Ich hatte zuvor noch nie etwas mit der iPhone-Entwicklung und somit auch nicht mit der Programmiersprache Objective-C zu tun gehabt. Auch die Diagnoseschnittstelle (OBD2) der Autos war mir fremd. Das Einarbeiten in diese neuen Gebiete war für mich eine grosse Lehre.

Doch trotzdem muss ich sagen macht die Entwicklung auf ein iPhone viel Spass, da man seine Software immer gleich mit sich im Kleinformat mitnehmen kann.

Fazit

Das Projektziel der Bachelorarbeit war die Kernfunktionalität zu implementieren, was auch erreicht wurde. Es war grundsätzlich ein sehr lehrreiches Projekt mit vielen Hürden und Freuden.

Die Version 1.0 für das iPhone bietet eine solide Basis für erste Tests mit ausgewählten Kunden. Zur Erleichterung der Projektkoordination zwischen den Projektmitgliedern würde ich das nächste Mal evtl. geeignete open source Tools einsetzen.

9. Glossar

Begriff	*Beschreibung*
Closed Loop	Ein Fahrzeug befindet sich im Closed Loop: -wenn die Lambda-Sonde ihre Betriebstemperatur erreicht hat. -wenn das Auto normal gefahren wird. Was bedeutet Closed Loop: Im Closed Loop werden die Feedback-Meldungen zur Regelung der Abgaswerte entsprechend der Tabelle des ECU ausgewertet. Aus diesen Meldungen werden dann die neuen Benzin/Luft-Gemische so hergestellt, dass die Abgas-Werte eingehalten werden.
Drosselklappen	Um bei Ottomotoren die geforderte Abgabeleistung zu regulieren, wird die zugeführte Luft- und Treibstoffmenge reguliert. Die Luftmenge wird dazu durch eine Drosselklappe im Saugrohr gesteuert.
EcoZahl	EcoZahl = Geschwindigkeit * Gewicht / Verbrauch. Resultate von 1 bis10, wobei 10 gut ist und 1 schlecht.
GPS	Global Postitioning System: Satelitenbasiertes Positionierungs-System
JavaME	Java Micro Edition: Java-Applikationsplattform für Mobile-Geräte
MAF	Ein Luftmassenmesser oder kurz LMM (auch *mass air flow meter* (MAF), Luftmassensensor bzw. LMS) ist ein in der Regelungs- und Messtechnik eingesetzter Durchflusssensor, der die Masse der pro Zeiteinheit durchströmenden Luft (den Massenstrom) bestimmt.
OBD2	On Board Diacnostic Interface 2: Schnittstelle um diverse Parameter eines Fahrzeugs zu lesen/schreiben.
Open Loop	Ein Fahrzeug befindet sich im Open Loop: -wenn das Auto eingeschaltet wird und die Lambda-Sonde (auch O2-Messer genannt) auf Betriebstemperatur geheizt werden muss. -wenn das Auto schnell beschleunigt wird. (z.B. Kick-Down) Was bedeutet Open Loop: Im Open Loop werden die Feedback-Meldungen der Lambda-Sonde zur Erhaltung der Abgas-Werte ignoriert. Es findet also keine Regelung des Benzin/Luft-Gemisches statt. Die Einspritzung wird stattdessen anhand der Drosselklappenposition und des Luftdurchflusses bestimmt. Es wird also ein fetthaltigeres Gemisch produziert.
PID	OBD2 Parameter-ID: Designiert welche Nachricht an den OBD Controller gesendet werden soll und wie die Nachricht aussieht

Schubabschaltung	**Schubabschaltung** ist eine gesteuerte Unterbrechung der Kraftstoffzufuhr zu einem Verbrennungsmotor, wenn dieser keine Leistung abgeben soll, sondern durch seine Schwungmasse weiter angetrieben wird, der Motor also *angeschoben* wird. Eine Schubabschaltung wurde zuerst bei Dieselmotoren eingesetzt, wobei die Einspritzpumpe die Kraftstoffförderung abschaltet, wenn der Drehzahlregler aktiv und die Motordrehzahl zu groß war. Das trat in der Regel dann ein, wenn man das Gaspedal nicht betätigt hatte und der Motor vom Fahrzeug geschoben wurde. Beim Ottomotor wird die Schubabschaltung seit 1980 in elektronischen Einspritzanlagen verwendet. Dabei wird die Kraftstoffzufuhr über die Einspritzventile abhängig von den Parametern Motortemperatur, Drehzahl, Drosselklappen- bzw. Gaspedalstellung abgeschaltet. Im Wesentlichen dient die Schubabschaltung der Kraftstoffeinsparung. Sie kommt zum Tragen, wenn man die Geschwindigkeit verringert und den Motor dadurch als Bremse nutzen kann (Motorbremse). Die Entsprechung nennt man Schubbetrieb. Dabei darf man nicht die Kupplung betätigen, da dann der Motor vom Antriebsstrang getrennt wird und wieder Kraftstoff verbraucht. Es ist dann zwar relativ wenig, aber immer noch deutlich mehr als bei der Nutzung der Schubabschaltung. Eine Kraftstoffeinsparung erreicht man daher durch vorausschauende Fahrweise, indem das Gas rechtzeitig komplett zurückgenommen wird und man möglichst spät bremst. So nutzt man die Bewegungsenergie des Fahrzeugs optimal aus. Bei modernen Fahrzeugen mit einer Anzeige des momentanen Kraftstoffverbrauches kann man den Unterschied zwischen Leerlauf und Schubabschaltung sehr gut beobachten. Bei der Schubabschaltung sollte die Anzeige 0,0 l / 100 km anzeigen. In einigen Ausnahmefällen (wie beispielsweise bei Automobilen der Hersteller Fiat und Daihatsu) ist die Anzeige trotz Schubabschaltung immer noch bei 1,0 bis 2,0 l / 100 km.
SDK	Ein Software Development Kit (SDK) ist eine Sammlung von Werkzeugen und Anwendungen, um eine Software zu erstellen, meist inklusive Dokumentation.
Symbian	Ist ein Betriebssystem für Smartphones und PDAs.
WAB2	2. Phase der Weiterausbildungskurse für den Führerausweis in der Schweiz.
VIN	Die Fahrzeug-Identifizierungsnummer (FIN) (engl.: Vehicle identification number – VIN) ist die international genormte, 17-stellige Nummer, mit der ein Kraftfahrzeug eindeutig identifizierbar ist.

10. Literaturverzeichnis

10.1. Eigene Dokumente und Verzeichnisse auf der CD:

E 1 /EcoHelper/docs/

E 2 /EcoHelper/docs/Analyse/Wichtige Kennzahlen.doc

E 3 /EcoHelper/docs/Analyse/Fahrdatenanalyse.doc

E 4 /EcoHelper/docs/Analyse/Diesel-Engines.doc

E 5 /EcoHelper/docs/Analyse/ELM CHIP/

E 6 /EcoHelper/TourLive/

E 7 /EcoHelper/docs/Projekplan/Projekplan BA.xls

E 8 /EcoHelper/docs/Protokoll/

E 9 /EcoHelper/docs/Test/

E 10 /EcoHelper/docs/Email/

E 11 /EcoHelper/docs/Zeiterfassung/

E 12 /EcoHelper/docs/Präsentationen/

E 13 /EcoHelper/docs/Documentation/Benutzerdokumentation

E 14 /EcoHelper/docs/Design/Software Architektur Dokument.doc

10.2. Source Code Projekt auf der CD:

S 1 /EcoHelper/src/

10.3. Websites:

URL 1	Erläuterungen zu OBD:	http://www.obd-2.de/techn.html
URL 2	Wikipedia zu OBD:	http://de.wikipedia.org/wiki/On-Board-Diagnose
URL 3	Wikipedia zu Fuel Cut Off:	http://de.wikipedia.org/wiki/Schubabschaltung
URL 4	Apple Developer:	http://developer.apple.com/iphone/index.action
URL 5	Infos über VIN:	http://de.wikipedia.org/wiki/Fahrzeug-Identifizierungsnummer
URL 6	cnlab AG Vorlagen:	http://www.cnlab.ch/kurse/sada/
URL 7	PLX Homepage:	http://www.plxdevices.com
URL 8	Tourlive Homepage:	http://www.tourlive.ch
URL 9	Apple Coding Guidelines:	http://developer.apple.com/mac/library/documentation/cocoa/conceptual/CodingGuidelines/CodingGuidelines.html
URL 10	RUP	http://de.wikipedia.org/wiki/Rational_Unified_Process
URL 11	OBD Pin Belegung:	http://en.wikipedia.org/wiki/OBD-II_PIDs
URL 12	Rev	http://www.devtoaster.com/products/rev
URL 13	Foliensatz cnlab	http://www.cnlab.ch/referate/MobileDatalogger-Benzinspar-Fitness-Trainer.pdf
URL 14	Tourlive Datalogger	http://www.tourlive.ch/mobile
URL 15	Speedtest	http://www.appstorehq.com/cnlabspeedtest-iphone-73989/app

10.4. Bibliographie:

B 1 Werner Zimmermann | Ralf Schmidgall „Bussysteme in der Fahrzeugtechnik“
ISBN 978-3-8348-0447-1

B 2 Stephen G. Kochan „Objective-C 2.0“
ISBN 978-3-8273-2746-8

B 3 Michael Kain „iPhone Anwendungsentwicklung für Einsteiger“
ISBN 978-3-86802-031-1

B 4 Erica Sadun „Das iPhone Entwicklerbuch“
ISBN 978-3-8273-2816-8

11. Anhang

11.1. Anforderungsspezifikation

11.1.1. Einleitung

11.1.1.1. Produkt Perspektive

Das Produkt EcoHelper soll für die verschiedenen Weiterausbildungskurse (WAB), für den Fahrlehrer und für Privatpersonen als Treibstoffspartrainer genutzt werden.

Es soll dem Nutzer anzeigen können was für ein Typ Fahrer dieser ist und welche Massnahmen er unternehmen kann um den Treibstoffverbrauch zu senken.

11.1.1.2. Produkt-Funktion

Das Produkt EcoHelper für das iPhone muss Daten des Autos mittels der OBD2-Schnittstelle auslesen und auswerten können. Zudem werden GPS-Daten, Beschleunigungs-Daten und Fotos vom iPhone zur Auswertung miteinbezogen.

Die EcoHelper Applikation ermittelt aus diesen Daten den Fahrerstil und den Verbrauch anhand von verschiedenen Kennzahlen, die fortlaufend ausgerechnet werden. Dieser wird dann dem Fahrer aktiv während der Fahrt mitgeteilt (optisch und akustisch) und Vorschläge zu dessen Verbesserung angegeben (z.B. kleinere Beschleunigung, Gaspedal nur bis zu einer gewissen Stellung neigen, im Bereich kleinerer Motordrehzahlen fahren, etc.)

Zusätzlich kann der Benutzer nach der Fahrt über die Webseite [URL8| die gesammelten Informationen im Einzelnen genauer untersuchen.

11.1.1.3. Benutzer Charakteristik

Für folgende Zielgruppen ist die EcoHelper-Applikation gedacht:

WAB-Kurse: Fahrer, die einen provisorischen Führerausweis besitzen, müssen die gesetzlichen Wiederholungskurse absolvieren. In diesen Kursen soll die EcoHelper Applikation den Kursleiter unterstützen. Mittels der Applikation sollen den Schülern während und nach der Fahrt Hinweise zur Verbesserung des Treibstoffverbrauchs übermittelt werden.

Fahrlehrer: Die Fahrlehrer sollen mittels dem EcoHelper die Schüler von der 1. Fahrstunde an, auf eine ökonomische Fahrweise trainieren.

TCS: Der Touring Club Suisse (TCS) hat entsprechendes Interesse seinen Mitgliedern die iPhone-Applikation inkl. der Hardware Schnittstelle (OBD2) zum Auto als Gesamtpaket anbieten zu können.

Privatpersonen: Natürlich sollen Autobegeisterte die Applikation auch privat nutzen können. Diese brauchen dazu nur die Applikation und die entsprechende Hardware für die Kommunikation zwischen iPhone und dem Auto. Die Software kann dann über den Apple Store geladen werden und die entsprechende Hardware bei der cnlab AG bezogen werden.

11.1.1.4. Einschränkungen

Da jeder Fahrzeughersteller und jedes Fahrzeugmodell unterschiedliche Werte über die ODB2-Schnittstelle zur Verfügung stellt, können wir nicht garantieren dass die iPhone Applikation mit allen Fahrzeugen funktioniert. Wenn wichtige Daten wie z.B. die Masse, der in den Motor

einströmenden Luft (Mass Air Flow, MAF) nicht vom Fahrzeug ausgelesen werden können, fehlen der Applikationen wichtige Inputdaten um den Treibstoffverbrauch berechnen zu können.

Zudem können die wenigsten Benzinerautos, die älter als 2001 sind und Dieselautos, die älter als 2003 sind, die Applikation nutzen. Diese älteren Fahrzeuge besitzen meist noch keine OBD2-Schnittstelle. USA Importfahrzeuge besitzen die OBD2 Schnittstelle schon seit 1996.

11.1.2.Annahmen

Folgende Annahmen wurden gemacht:

Auslesen der OBD2-Daten:	Wir gehen davon aus, dass wir mittels der vom Hardwareherstellter zur Verfügung gestellten Library, das Auslesen und Senden von OBD2-Befehlen mittels iPhone mit einem kleineren Aufwand bewerkstelligen können.
iPhone Entwicklerumgebung:	Die iPhone Entwicklungsumgebung ist ein neues Gebiet, dass eingearbeitet werden muss. Es wird angenommen, dass diese Einarbeitungszeit nicht länger als 3 Woche geht.
Fahrzeugmodel:	Für die erste Entwicklung wird die Software nur für geschaltete Benziner erstellt und getestet.

11.1.3.Abhängigkeiten

Die Applikation hat folgende Abhängigkeiten:

Hardware:	Damit die iPhone Applikation mit dem Auto kommunizieren kann, ist ein OBD2 Stecker mit WIFI-Adapter von der Firma PLX-Devices notwendig. [URL7]
Fahrzeugmodell:	In der ersten Entwicklung werden nur geschaltete Benziner unterstützt.
GPS:	Damit auch die Daten zum richtigen Zeitpunkt und Ort erfasst werden können, braucht die Applikation einen GPS Empfang, um die Koordinaten auslesen zu können. Zudem muss auch die Fahrsituation wie Stau, Bergfahrt oder Autobahn für die korrekte Bewertung von Verbrauch mit einbezogen werden.
iPhone Modell:	Für die volle Nutzbarkeit der EcoHelper Applikation ist ein iPhone 3G(S) oder neuer notwendig.
OBD2 Parameter	Je nach Hersteller und Modell des Fahrzeuges werden unterschiedliche OBD-Werte zur Abfrage freigegeben. Für die Funktionsweise der Anwendung müssen folgende Werte ausgelesen werden können: RPM, MAF, Throttle > Anzeige des Bremsverhaltens, Gang, Verbrauch und Gaspedal RPM, MAF > Anzeige des Bremsverhaltens, Gang und Verbrauch RPM > Anzeige des Bremshaltens, Gang

11.1.4. Spezifische Anforderungen

11.1.4.1. Funktionale Anforderungen

Die EcoHelper Applikation gibt es schon für Mobiltelefone auf Symbian- und Android-Basis. Deshalb sind die Grundanforderungen für die iPhone Applikation ähnlich wie die der anderen Vorgänger. Zudem wurden die Anforderungen mittels Analyse von Fahrten und durch Vergleichen verschiedener Messwerte ermittelt. Wobei auch intensive Gespräche mit den Betreuern zu neuen und interessanten Anforderungen beigetragen haben.

11.1.4.2. Auto erfassen

Die iPhone EcoHelper Applikation kann mehrere Autos erfassen. Beim Auto muss der Benutzer die Gänge, das Gewicht und den durchschnittlichen Verbrauch laut Werk eingeben werden. Freiwillig kann der Benutzer noch die VIN Nummer des Autos angeben.

11.1.4.3. Fahrer erfassen

In der iPhone EcoHelper Applikation kann der Benutzer auch mehrere Fahrer erfassen. Dies ist vor allem für Fahrlehrer gedacht, die dann so ihre verschiednen Schüler erfassen können.

11.1.4.4. Schaltvorgänge erfassen

Die iPhone EcoHelper Applikation ermöglicht mittels Übersetzungsberechnungen (Geschwindigkeit / Umdrehungen) die Gänge zu erfassen. Dabei ist es wichtig, dass bei der ersten Erfassung, alle Gänge mindestens. 10 Sekunden lang nacheinander durchgeschaltet werden.

11.1.4.5. Fahrt erfassen

Mittels den Befehlen „Fahrt starten und Fahrt beenden“ kann eine neue Fahrt mit all den benötigten Daten erfasst werden.

Der Benutzer kann vor Beginn der Fahrt ein Auto auswählen.

Der Benutzer kann den Fahrer vor der Fahrtaufzeichnung auswählen.

11.1.4.6. Fahrdaten aus dem Auto auslesen und aufzeichnen

Die iPhone EcoHelper Applikation sendet und empfängt über WLAN-Adapter an die OBD2-Schnittstelle Befehle zum Auto, um Fahrparameter wie Motordrehzahl, Geschwindigkeit, Gangposition, Drosselklappenstellung, Luftmasse und Verbrauch auszulesen und aufzuzeichnen. Falls man die verschiedenen Gangpositionen nicht auslesen kann, muss man sie über die Übersetzungsverhältnisse berechnen.

Andere Daten wie Pedalstellung und Tankfüllung wären von Vorteil, werden aber leider nur von den wenigsten Autos zur Verfügung gestellt.

Falls die Verbindung unterbrochen wird, meldet sich das iPhone mit einer Meldung, sodass der Benutzer die Verbindung überprüfen kann und anschliessend weiterfahren kann.

11.1.4.7. GPS- und Beschleunigungsdaten aus dem iPhone auslesen und aufzeichnen

Die iPhone EcoHelper Applikation soll die vom iPhone zur Verfügung gestellten GPS und Beschleunigungsfunktionen nutzen und die ausgelesenen Werte zu speichern.

Das Auslesen der GPS-Geschwindigkeit kann zum Vergleichen der Auto-Geschwindigkeit gebraucht werden. Zudem sollen die GPS-Position und -Höhe aufgezeichnet werden. Auch sollen die Beschleunigungswerte zur Auswertung der Fahrt mit aufgezeichnet werden.

Falls die Verbindung unterbrochen wird, meldet sich das iPhone mit einem Piepston alle 10 Sekunden.

11.1.4.8. Fahrparameter darstellen

Die iPhone EcoHelper Applikation besitzt ein User Interface, das die aufgezeichneten Werte grafisch darstellt.

Das User Interface besteht aus folgenden Komponenten.

- Analyse und Auswertungsfunktionen
- Live Modus (Darstellung der wichtigsten Fahrdaten)
- Einstellungsseite

11.1.4.9. Kommunikation

Die Kommunikation zwischen der iPhone EcoHelper Applikation und dem Auto findet mittels einem OBD2-Stecker statt. Dieser Stecker verfügt über einen WLAN-Adapter sodass dieser mit dem iPhone kommunizieren kann. Die Daten werden dann entweder über Wifi oder über GSM auf das Webportal gesendet.

11.1.4.10. Länderspezifische Einstellungen

Der Benutzer kann in der Einstellungsseite zwischen CH, ENG und US als Ländereinstellung auswählen. Dies bewirkt dann, dass die Anzeige in die entsprechenden Einheiten gewechselt wird.

11.1.4.11. Datenablage

Die empfangenen Fahrdaten und GPS Daten sollen mittels einem festgelegten Intervall von 1 Sekunde in einer lokalen Datenbank, im iPhone, abgelegt werden

11.1.4.12. Log Files

Zu den Fahrten sollen entsprechende Logfiles gespeichert werden. Der Benutzer soll die Möglichkeit haben, ausgewählte Logfiles oder alle Logfiles per Mail zu verschicken.

11.1.4.13. Hintergrundbeleuchtung

Der Benutzer kann die Hintergrundbeleuchtung des iPhone unter den Einstellungen ein und ausschalten.

11.1.4.14. Daten exportieren

Die iPhone EcoHelper Applikation ermöglicht ein Export der gespeicherten Daten als eine CSV- und eine KML-Datei zu exportieren. Die CSV-Datei kann als Rohwerte für weitere Berechnungen berechnet werden oder auch um die verschiedenen Statistikfunktionen des Onlineportals des EcoHelpers zu benutzen. Dieser Export kann auf dem Onlineportal des EcoHelpers getätigt werden.

11.1.5. Nichtfunktionale Anforderungen

11.1.5.1. Hardwareanschliessung

Mit der Kurzanleitung [E13] sollte der Benutzer ohne Informatikkenntnisse innerhalb von 2 Minuten den WLAN-OBD-Adpater am Auto angeschlossen haben und diese dann mit dem iPhone verbinden können.

11.1.5.2. Fahrt aufnehmen

Das User Interface des EcoHelper sollte einfach und intuitiv bedienbar sein. Der Benutzer wird mittels visuellen und akustischen Meldungen auf sein Fahrverhalten hingewiesen. Zudem ist die Oberfläche so ausgewählt, dass die wichtigsten Informationen gut ablesbar sind, sodass der Lenker nicht zu stark von der Fahrt abgelenkt ist.

11.1.5.3. Online- / Offline Modus

Online-Modus:	Im Online-Modus werden die erhaltenen Werte von der Applikation direkt, während der Fahrt, zum Webportal geschickt.
Offline-Modus:	Im Offline-Modus werden die Daten in einer spezifischen iPhone-Datenbank gespeichert. Erreicht das iPhone ein WLAN mit Internetverbindung, so kann der Benutzer die Daten zur Webplattform schicken. So spart der Benutzer teure Verbindungskosten.

11.1.5.4. Korrektheit der Werte

Die iPhone EcoHelper Applikation soll die Werte, die sie vom Auto oder vom iPhone selber bekommt nicht verfälschen sondern nur allenfalls runden bzw. für den Gebrauch anpassen. Alle dem Benutzer angezeigten Daten müssen einen korrekten Wert haben und unverändert dem Benutzer angezeigt werden.

11.1.5.5. Bleibt nicht hängen

Die Applikation ist so aufgebaut, dass sie sich zu keinem Zeitpunkt aufhängt. Der Benutzer soll zu jeder Zeit vollen Zugriff auf die Funktionen haben. Auch wenn sich iPhone spezifische oder OBD2 spezifische Funktionen aufhängen wird die iPhone EcoHelper Applikation stets weiterarbeiten können.

11.1.5.6. Stabile Laufleistung

Die iPhone EcoHelper Applikation wird im eingeschalteten iPhone-Zustand zu 99% verfügbar sein. Die Applikation kann auch laufen ohne, dass die OBD2-Funktionen zur Verfügung stehen.

11.1.5.7. Startzeit

Die iPhone EcoHelper Applikation soll auf dem iPhone innerhalb von 8 Sekunden gestartet werden.

11.1.5.8. Verbindung mit dem Auto mittels OBD2

Die Verbindung zwischen dem Auto und der Applikation mus innerhalb von maximal 10 Sekunden stattfinden.

11.1.5.9. Fahrdaten

OBD kann Protokollbedingt 5 Werte pro Sekunde liefern. Falls man also nur einen Wert abfragt, liefert und die OBD Schnittstelle 5 Werte pro Sekunde. Fragt man 5 verschiedene Parameter ab, so bekommt man nur 1 Wert pro Sekunde und Parameter.

11.1.5.10. Wartbarkeit

Durch einhalten von modernen Software-Engineering-Methoden und einer guten Code-Dokumentation können Erweiterungen ohne grössere Einarbeitungsaufwände getätigt werden.

11.1.5.11. Updates

Änderungen die Bugfixes betreffen werden in der ersten Version über die Apple Add-Hoc Verteilung geupdatet. Zu einem späteren Zeitpunkt soll die App auch über das Apple Store geupdatet werden können.

11.1.5.12. Anpassbarkeit

Die iPhone EcoHelper Applikation kann im Breitbildschirm-Modus oder auch im Hochbildschirm-Modus angesehen werden. Zudem kann die erste Version nur an geschaltete Benziner mit einem OBD2-Anschluss angeschlossen werden.

11.1.5.13. Konfigurierbarkeit

Die iPhone EcoHelper-Applikation erlaubt dem User folgende Einstellungen vorzunehmen:

Kamera:	Der Benutzer kann den Kameramodus ein und ausschalten. Und zudem noch einen Intervall auswählen indem die Fotos gemacht werden sollen.
Auto:	Der Benutzer kann verschiedene Autos erfassen und diese vor der Fahrt auswählen.
Fahrer:	Der Benutzer kann verschiedene Fahrer erfassen und diese vor der Fahrt auswählen.
Logging:	Der Benutzer kann das letzte Log-File per Mail zusenden.
Onlinemodus:	Der Benutzer kann zwischen Online und Offline auswählen. Online bedeutet, dass die Daten sofort per GSM übertragen werden und im Offline Modus wird gewartet bis ein Wifi zur Verfügung steht um die Daten zu schicken.
Einheit:	Der Benutzer kann zwischen 3 Einheiten wählen, Schweiz (CH), England (GB) oder der USA (US).

11.1.6. Schnittstellen

11.1.6.1. Benutzerschnittstelle

Die Benutzerschnittselle stellt das grafische User Interface auf dem iPhone dar.

11.1.6.2. Hardwareschnittstelle

- OBD2 (Mittels Wifi-Adapter am OBD2 Stecker die Verbindung zum iPhone herstellen)
- Wifi und GSM Netz (für die Datenübertragung zum Webportal)

11.1.6.3. Softwareschnittstellen

- Kommunikationsschnittstelle mittels Hersteller Library zwischen OBD2-Empfänger und IPhone.
- Schnittstelle zwischen iPhone EcoHelper Applikation und IPhone Funktionen wird mittels IPhone SDK zur Verfügung gestellt.

11.1.6.4. Datenbankschnittstelle

IPhone spezifische Datenbankschnittstelle wird vom Apple IPhone SDK zur Verfügung gestellt.

11.1.6.5. Kommunikationsschnittstelle

Alle Daten werden zu dem EcoHelper Webserver geschickt. Dort kann dann der Benutzer zusätzlich noch weiter statistische Auswertungen machen und Fahrten untereinander vergleichen.

11.1.7. Lizenzanforderungen

Folgende Lizenzen werden benötigt:

- Apple XCode SDK (Entwicklungumgebung auf dem iPhone)
- PLX Devices Diagnostic Library (API für die Verbindung zwischen dem iPhone und dem ODB2 Adapter)
- Palmer Performance programmers API (OBD2 Abstraktion)

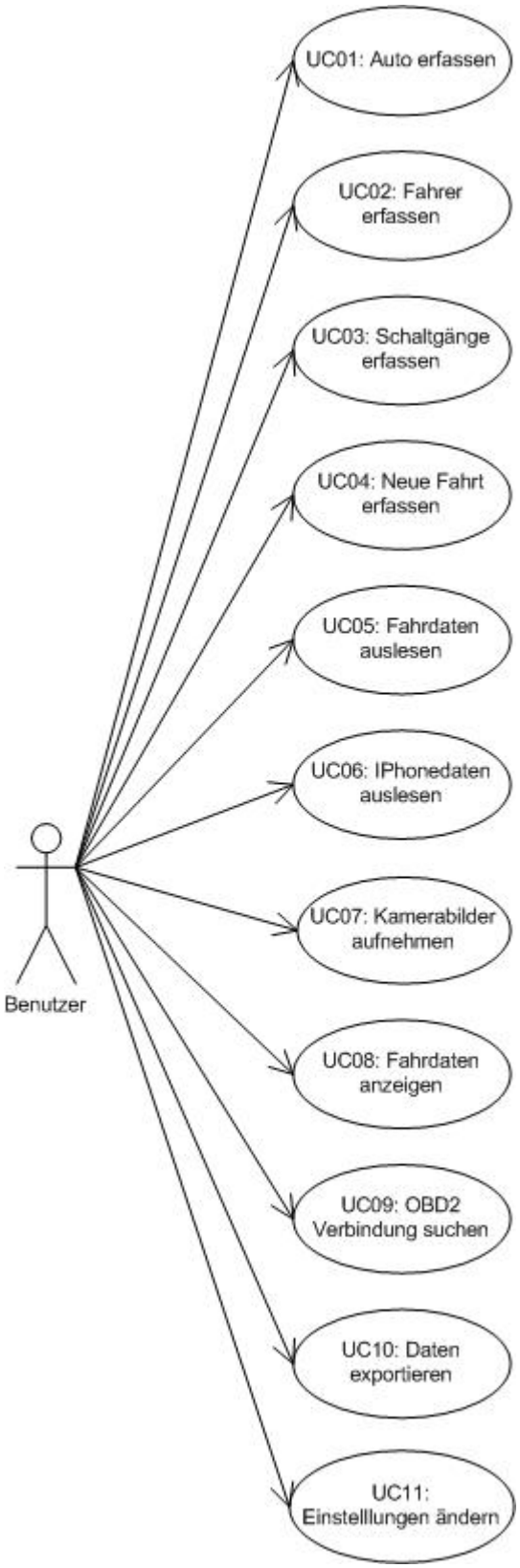
11.1.8. Verwendete Standards

Folgende Standards werden in diesem Projekt gebraucht:

- HSR und cnlab AG Dokumentationsstandard [URL6]
- Apple Programmierrichtlinien [URL 9]
- RUP als Vorgehensmodell für die Softwareentwicklung [URL10]

11.1.9.Use Cases

11.1.9.1. Use Case Diagramm



11.1.9.2. Aktoren & Stakeholders

Primary Actors ist der iPhone Benutzer und das System (definiert als EcoHelper Software inkl. iPhone Hardware).

Supporting Actors ist die Webapplikation TourLive und der OBD2-Wifi-Adapter.

Als Stakeholder treten Fahrlehrer, Fahrschüler, Autonarr und der Normal-Benutzer.

11.1.9.3. Use Case 01: Auto erfassen

Use Case Name	Auto erfassen (UC01)
Primary Actor	iPhone Benutzer
Stakeholder and Interests	- Normaler Benutzer: Möchte möglichst wenig Eingaben über das iPhone machen und keine Angaben aus dem Fahrzeughandbuch suchen müssen. - Fahrlehrer: Möchte seine verschiedenen Fahrzeuge erfassen können, nimmt sich auch Zeit die Detailangaben aus dem Fahrzeughandbuch einzugeben. - Autonarr: Möchte zu seinem Fahrzeug auch ein Foto ablegen.
Preconditions	- iPhone Eco Helper Applikation wurde gestartet
Postconditions	- Zwingende Fahrzeugdetails wurden in der internen Datenbank abgelegt.
Main Success Scenario	1. Benutzer drückt auf „Mein Profil“ 2. Benutzer drückt auf „Neues Auto“ 3. Benutzer füllt die Pflichtfelder (Marke, Modell, Gewicht und Anzahl Gänge aus) 4. Benutzer drückt auf „Speichern“ Benutzer kann Schritte 2 – 4 beliebig oft wiederholen.
Extensions (Alternative Flow)	3a. Benutzer möchte zusätzlich ein Foto verwenden. 1a. Benutzer möchte neues Bild aufnehmen 1. iPhone Kamera Modus starten 2. Foto aufnehmen 3. Foto prüfen und übernehmen 1b. Benutzer möchte bestehendes Bild nehmen 1. iPhone Fotogallerie anzeigen 2. Foto prüfen und übernehmen 4a. Plausibilitätsprüfung ergibt, dass das Gewicht in Tonnen anstatt Kg eingegeben wurde. 1. Dem Benutzer eine Fehlermeldung anzeigen 2. Zurück zu Schritt 3 4b. Plausibilitätsprüfung ergibt, dass Gänge keine Zahlen sind. 1. Dem Benutzer eine Fehlermeldung anzeigen 2. Zurück zu Schritt 3
Special Requirements	- Der relativ kleine Touchpad darf nicht überladen werden und muss auch von „nicht-iPhone-Besitzern“ bedient werden können. - Internationalisierung der Einheiten und der Textfelder.
Technology and Data Variations List	-
Frequency of Occurrence	Für Fahrlehrer öfters, für den normalen Benutzer meist nur bei der ersten Benützung
Open Issues	-

Table 1: Use Case 01 Auto erfassen

11.1.9.4. Use Case 02: Fahrer erfassen

Use Case Name	Fahrer erfassen (UC02)
Primary Actor	iPhone Benutzer
Stakeholder and Interests	- Normaler Benutzer: Möchte möglichst wenig Eingaben über das iPhone machen. - Fahrlehrer: Möchte seine verschiedene Schüler erfassen können, nimmt sich auch Zeit Bemerkung zum Schüler aufzuschreiben.
Preconditions	- iPhone Eco Helper Applikation wurde gestartet
Postconditions	- Zwingende Fahrerdetails wurden in der internen Datenbank abgelegt.
Main Success Scenario	1. Benutzer drückt auf „Mein Profil“ 2. Benutzer drückt auf „Neuer Fahrer“ 3. Benutzer füllt das Pflichtfeld „Fahrername“ aus 4. Benutzer drückt auf „Speichern“ Benutzer kann Schritte 2 – 4 beliebig oft wiederholen.
Extensions (Alternative Flow)	3a. Benutzer möchte zusätzlich eine Bemerkung dazu schreiben. 1a. Benutzer schreibt im Feld „Bemerkungen“ sein Text 4a. Prüfung ergibt, dass das Pflichtfeld „Fahrername“ nicht ausgefüllt wurde. 1. Dem Benutzer eine Fehlermeldung anzeigen 2. Zurück zu Schritt 3
Special Requirements	- Der relativ kleine Touchpad darf nicht überladen werden. Und muss auch von „nicht-iPhone-Besitzern“ bedient werden können. - Internationalisierung der Einheiten und der Textfelder.
Technology and Data Variations List	-
Frequency of Occurrence	Für Fahrlehrer öfters, für den normalen Benutzer meist nur bei der ersten Benützung
Open Issues	-

Table 2: Use Case 02 Fahrer erfassen

11.1.9.5. Use Case 03: Schaltvorgänge erfassen

Use Case Name	Schaltvorgänge erfassen (UC03)
Primary Actor	iPhone Benutzer
Stakeholder and Interests	- Normaler Benutzer: Möchte möglichst wenig Eingaben über das iPhone machen. - Fahrlehrer: Möchte die Gänge seines Autos schon vor den Schülerfahrten erfasst haben.
Preconditions	- UC 04
Postconditions	- Benutzer hat alle Gänge im 10 Sekundentakt nacheinander durchgeschaltet. iPhone hat die Schaltvorgänge zu diesem Auto intern gespeichert.
Main Success Scenario	1. Benutzer fährt alle 10 Sekunden die Gänge nacheinander ab. 2. Benutzer beendet die Fahrt indem er auf „Stop“ drückt Benutzer kann die Schritte 1-2 beliebig oft wiederholen.
Extensions (Alternative Flow)	1b. Fahrer überspringt Gänge 1. Gang wird zuerst falsch erfasst bis der übersprungene Gang gefahren wird.
Special Requirements	- Das Auto muss alle relevanten Parameter (Geschwindigkeit und RPM) über die OBD2-Schnittstelle liefern.
Technology and Data Variations List	-
Frequency of Occurrence	Für Fahrlehrer wie auch normale Benutzer jedesmal zu machen, wenn ein neues Auto hinzugefügt wurde.
Open Issues	-

Table 3: Use Case 03 Schaltvorgägne erfassen

11.1.9.6. Use Case 04: Neue Fahrt erfassen

Use Case Name	Neue Fahrt erfassen (UC04)
Primary Actor	iPhone Benutzer
Stakeholder and Interests	- Normaler Benutzer: Möchte möglichst wenig Eingaben über das iPhone machen. - Fahrlehrer: Wählt vor jeder Fahrt das entsprechende Auto und den entsprechenden Schüler aus.
Preconditions	- iPhone Eco Helper Applikation wurde gestartet - OBD2 Adapter mit dem iPhone verbunden - Fahrer und Auto wurden erfasst. - Profil ist komplett
Postconditions	- Benutzer hat die Fahrt mit dem „Fahrt beenden“ Button beendet und sieht die statistische Auswertung der Fahrt.
Main Success Scenario	1. Benutzer schliesst den OBD Adapter an das Auto an 2. Benutzer drückt auf „Mein Profil“ 3. Benutzer wählt einen Fahrer aus 4. Benutzer wählt ein Auto aus 5. Benutzer startet die Fahrt indem er auf „Aufnahme“ drückt 6. Daten werden entweder im Offline-Modus gespeichert oder kontinuierlich über GSM an den Webserver geschickt. 7. Benutzer beendet die Fahrt indem er auf „Fahrt beenden“ drückt Benutzer kann die Schritte 2 – 7 beliebig oft wiederholen.
Extensions (Alternative Flow)	1a. Verbindung kann nicht hergestellt werden. 1. Fehlermeldung ausgeben 2a. Zurück zu Schritt 1 5a. Prüfung ergibt kein Fahrer und/oder Auto ausgewählt. 3. Dem Benutzer eine Fehlermeldung anzeigen 4. Zurück zu Schritt 3 6a Im Offlinemodus werden die Daten intern gespeichert und sobald ein Wifi verfügbar ist an den WebServer geschickt. 6b Falls der Onlinemodus gewählt ist und kein GSM Empfang möglich ist werden die Werte intern gespeichert und sobald der GSM Empfang wieder da ist, gesendet.
Special Requirements	- Der OBD2-Adapter muss korrekt mit dem iPhone verbunden sein. - Das Auto muss die relevanten Parameter über die OBD2 Schnittstelle liefern können.
Technology and Data Variations List	-
Frequency of Occurrence	Dieser Use Case wird bei jeder Nutzung der Applikation gemacht.
Open Issues	-

Table 4: Use Case 04 Neue Fahrt erfassen

11.1.9.7. Use Case 05: Fahrdaten auslesen

Use Case Name	Fahrdaten auslesen (UC05)
Primary Actor	iPhone, OBD2 Adapter
Stakeholder and Interests	- Benutzer möchte während seiner Fahrt auch Parameter vom Auto zur Auswertung seiner Fahrt bekommen.
Preconditions	- iPhone EcoHelper Applikation wurde gestartet - OBD2 Adapter mit dem iPhone verbunden - UC 4 neue Fahrt erfassen
Postconditions	- Benutzer hat die Fahrt mit dem „Fahrt beenden“ Button beendet und sieht die statistische Auswertung der Fahrt.
Main Success Scenario	1. Daten werden während der Fahrtaufnahmen kontinuierlich vom Auto über die OBD2-Schnittstelle gelesen und über Wifi an das iPhone versendet. 2. Daten werden entweder im Online- oder Offlinemodus gespeichert und gesendet.
Extensions (Alternative Flow)	1a. Verbindung kann nicht hergestellt werden. 1. Fehlermeldung ausgeben 2a. Zurück zu Schritt 1 1b. Auto unterstützt nicht alle Pflichtparameter 1. Fehlermeldung ausgeben 2. iPhone EcoHelper Applikation ohne OBD nutzen 2a Im Offlinemodus werden die Daten intern gespeichert und sobald ein Wifi verfügbar ist an den WebServer geschickt. 2b Falls der Onlinemodus gewählt und kein GSM Empfang möglich ist werden die Werte intern gespeichert und sobald der GSM Empfang wieder da ist, gesendet.
Special Requirements	- Der OBD2-Adapter muss korrekt mit dem iPhone verbunden sein. - Das Auto muss die relevanten Parameter über die OBD2 Schnittstelle unterstützen
Technology and Data Variations List	-
Frequency of Occurrence	Dieser Use Case wird bei jeder Nutzung der Applikation, bzw. bei jeder Aufzeichnung der Fahrt gemacht.
Open Issues	-

Table 5: Use Case 05 Fahrdaten auslesen

11.1.9.8. Use Case 06: iPhonedaten auslesen

Use Case Name	iPhonedaten auslesen (UC06)
Primary Actor	iPhone
Stakeholder and Interests	- Benutzer möchte zur besseren Auswertung der Fahrt zusätzlich noch GPS Werte haben um eine Visualisierung seiner Fahrt auf eine Karte zu ermöglichen.
Preconditions	- iPhone EcoHelper Applikation wurde gestartet - UC 4 neue Fahrt erfassen
Postconditions	- Benutzer hat die Fahrt mit dem „Fahrt beenden“ Button beendet und sieht die statistische Auswertung der Fahrt.
Main Success Scenario	1. GPS Daten und Beschleunigungsdaten werden während der kontinuierlichen Fahrtaufnahme gespeichert. 2. Daten werden entweder im Online- oder Offlinemodus gespeichert und gesendet.
Extensions (Alternative Flow)	1a. GPS hat keinen Empfang. 2a. Datensatz ohne GPS Werte erfassen 1b. GPS Modul ist nicht erreichbar 1. Fehlermeldung ausgeben 2. Datensatz ohne GPS Werte erfassen
Special Requirements	- GPS Empfang muss möglich sein - Die Applikation muss mindestens auf einem iPhone 3G mit GPS installiert sein
Technology and Data Variations List	-
Frequency of Occurrence	Dieser Use Case wird bei jeder Nutzung der Applikation gemacht.
Open Issues	-

Table 6: Use Case 06 iPhonedaten auslesen

11.1.9.9. Use Case 07: Kamerabilder aufnehmen

Use Case Name	Kamerabilder aufnehmen (UC07)
Primary Actor	iPhone
Stakeholder and Interests	- Benutzer möchte für die bessere Auswertung der Fahrt zusätzlich noch Bilder während der Fahrt aufnehmen.
Preconditions	- iPhone EcoHelper Applikation wurde gestartet - UC 4 neue Fahrt erfassen - UC 11 Kameramodus eingeschaltet und Intervall ausgewählt
Postconditions	- Benutzer hat die Fahrt mit dem „Stop“ Button beendet und sieht die statistische Auswertung der Fahrt. - Benutzer kann seine Fahrt auf dem Webserver mit den Bildern auswerten.
Main Success Scenario	2. Kamerabilder werden während der kontinuierlichen Fahrtaufnahme gespeichert. 2. Daten werden entweder im Online- oder Offlinemodus gespeichert und gesendet.
Extensions (Alternative Flow)	1a. Kamera Modul ist nicht erreichbar 1. Fehlermeldung ausgeben 2. Datensatz ohne Kamerabilder erfassen
Special Requirements	- Kameramodus muss unter Einstellungen eingeschaltet sein - Die Applikation muss mindestens auf einem iPhone 3G mit Kamera installiert sein
Technology and Data Variations List	-
Frequency of Occurrence	Der Benutzer wird dieser UC in den meisten Fällen von Fahrtaufnahmen benutzen.
Open Issues	-

Table 7: Use Case 07 Kamerabilder aufnehmen

11.1.9.10. Use Case 08: Fahrdaten anzeigen

Use Case Name	Fahrdaten anzeigen (UC08)
Primary Actor	iPhone
Stakeholder and Interests	- Benutzer möchte für die bessere Auswertung der Fahrt zusätzlich interaktiv Fahrdaten ablesen können.
Preconditions	- iPhone EcoHelper Applikation wurde gestartet - UC 4 neue Fahrt erfassen - UC 9 OBD2 Verbindung aufgebaut
Postconditions	- In der Current View werden alle Parameter kontinuierlich angezeigt und ausgewertet. Die Daten werden permanent von der OBD2 Schnittstelle gesendet und vom iPhone ausgewertet und in der CurrentView angezeigt. - In der Evaluation View bekommte der Benutzer die Auswertung der ganzen Fahrt.
Main Success Scenario	1. In der CurrentView werden alle Parameter kontinuierlich angezeigt und ausgewertet. Die Daten werden permanent von der OBD2 Schnittstelle gesendet und von der iPhone EcoHelper Applikation ausgewertet und angezeigt. 2. Beendet der Benutzer die Fahrt durch drücken auf dem „Fahrt beenden“ Button, wechselt die Applikation auf die Evaluation View. In dieser Sicht wird die Fahrt statistisch ausgewertet und angezeigt.
Extensions (Alternative Flow)	2a. User beendet die Fahrt durch Verlassen des Programms. 1. Fahrt beenden und speichern.
Special Requirements	
Technology and Data Variations List	-
Frequency of Occurrence	Der Benutzer wird dieser UC in den meisten Fällen von Fahrtaufnahmen benutzen.
Open Issues	-

Table 8: Use Case 08 Fahrdaten anzeigen

11.1.9.11. Use Case 09: OBD2 Verbindung aufbauen

Use Case Name	OBD2 Verbindung aufbauen(UC09)
Primary Actor	iPhone, OBD2 Adapter, User
Stakeholder and Interests	- Benutzer möchte für die optimale Auswertung seiner Fahrt verschiedene Werte des Autos auf dem iPhone auslesen können.
Preconditions	- iPhone EcoHelper Applikation wurde gestartet - Das Auto ist ein EU Fahrzeug nach 2001 oder ein US Importfahrzeug nach 1994. - Das Auto ist ein Benziner und handgeschaltet - Das Auto besitzt einen OBD2 Anschluss - Das OBD2 Protokoll liefert zu diesem Auto die relevanten Parameter (RPM, Geschwindigkeit, MAF, Drosselklappenstellung)
Postconditions	- UC8 funktioniert einwandfrei
Main Success Scenario	1. Benutzer überprüft ob sein Auto OBD2 unterstützt 2. Benutzer schliesst den Adapter an das Auto an 3. Benutzer schaltet die Zündung des Auto an 4. Benutzer startet die EcoHelper Applikation 5. EcoHelper Applikation verbindet mit dem OBD2 Adapter 6. EcoHelper Applikation zeigt mit einer grünen Lampe das die Verbindung steht und alle wichtigen Daten unterstützt werden. Schritte 2 -6 können beliebig oft wiederholt werden
Extensions (Alternative Flow)	1a Auto unterstützt keine OBD2. 5a Verbindung kann nicht hergestellt werden. 1. Fehlermeldung ausgeben 2. Benutzer darauf aufmerksam machen die Wifi-Verbindung zu prüfen 3. Zurück zu Schritt 2 6a Verbindung hergestellt aber das Auto unterstützt nicht alle Werte 1. Meldung geben, dass nicht alle Werte unterstützt werden 2. User darauf aufmerksam machen, dass die Applikation nicht voll funktionsfähig sein wird
Special Requirements	-PLX WIFI OBD2 Adapter notwendig.
Technology and Data Variations List	-
Frequency of Occurrence	Dieser UseCase ist für jede Fahrtanalyse notwendig um die volle Funktionalität der EcoHelper Applikation nutzen zu können.
Open Issues	-

Table 9: Use Case 09 OBD2 Verbindung aufbauen

11.1.9.12. Use Case 10: Daten exportieren

Use Case Name	Daten exportieren (UC10)
Primary Actor	User
Stakeholder and Interests	- Benutzer möchte nicht nur auf dem iPhone, sondern auch auf dem Webportal seine Fahrten analysieren lassen. Auf dem Webportal kann er dann noch zusätzliche Funktionen wie Fahrten vergleichen und nutzen.
Preconditions	-UC 8 ist am laufen
Postconditions	- Daten sind auf dem Webportal zur Auswertung bereit
Main Success Scenario	1. Benutzer hat Applikation im Onlinemodus 2. Daten werden während der Fahrt live zum Webserver gesendet. 3. Daten werden intern gespeichert. 4. Beim Beenden des UC 8 wird der Export auch beendet. Schritte 2 -3 können beliebig oft wiederholt werden
Extensions (Alternative Flow)	1a Benutzer hat Applikation im Offlinemodus. 1. Daten werden in der internen Datenbank gespeichert. 2. Sobald der Benutzer zu einem Wifi kommt, kann er den Button “Fahrdaten exportieren” klicken und die letzte Fahrt wird dann zum Webserver geschickt.
Special Requirements	-GSM oder Wifi Verbindung notwendig
Technology and Data Variations List	-
Frequency of Occurrence	Dieser Use Case ist für jede Fahrtanalyse notwendig um die volle Funktionalität der EcoHelper Applikation nutzen zu können.
Open Issues	-

Table 10: Use Case 10 Daten exportieren

11.1.9.13. Use Case 11: Einstellungen ändern

Use Case Name	Einstellungen ändern (UC11)
Primary Actor	User
Stakeholder and Interests	- Benutzer möchte die iPhone EcoHelper Applikation nach seinen Wünschen einrichten.
Preconditions	- iPhone EcoHelper Applikation ist gestartet
Postconditions	- Benutzer hat die iPhone EcoHelper Applikation nach seinen Wünschen eingestellt.
Main Success Scenario	1. Der Benutzer folgende Einstellungen wählen: - Online oder Offlinemodus - Kamera ein und ausschalten - Kameraintervall wählen - Log Files senden - Hintergrundbeleuchtung ein und ausschalten - Ländereinstellung auswählen (MPG UK, MPG US, SI Einheiten) Schritt 1 kann beliebig oft wiederholt werden
Extensions (Alternative Flow)	
Special Requirements	
Technology and Data Variations List	-
Frequency of Occurrence	Dieser Use Case wird vom Benutzer meistens bei der Erstanwendung gebraucht.
Open Issues	-

Table 11: Use Case 11 Einstellungen ändern

11.2. Wichtige Kennzahlen

11.2.1.Standardabweichung

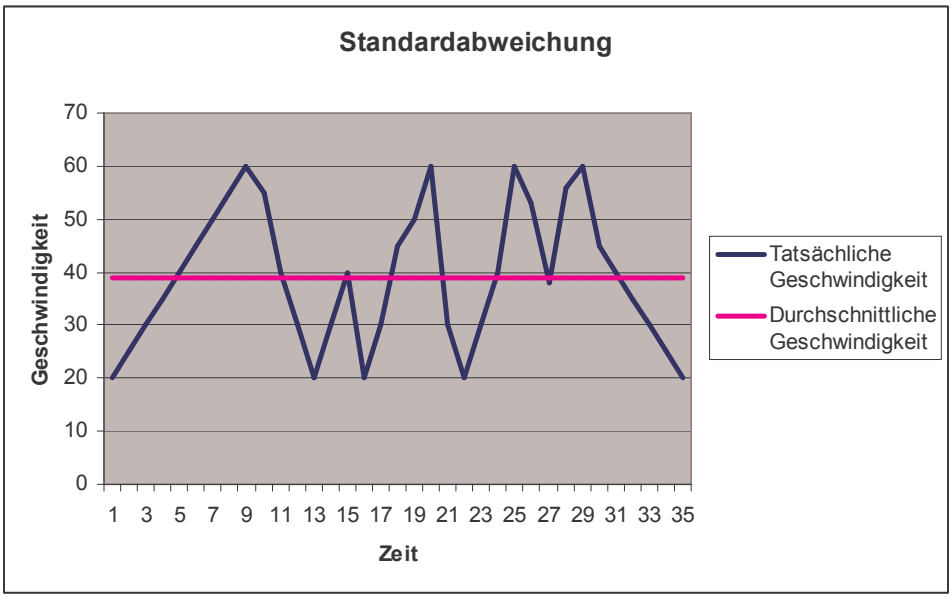


Abbildung 32 Standardabweichung (Geschwindigkeit) {Ref: seak_Statistikauswertung (EcoHelper)_DOLDER.xls}

Wenn man die Geschwindigkeitskurve über die Zeit beobachtet und gleichzeitig diesen an jedem Abtastzeitpunkt mit dem Mittelwert vergleicht, so kann man hier die Abweichung ausrechnen. Abweichung = Tatsächliche Geschwindigkeit – Mittelwert.
Bewegt sich die tatsächliche Kurve über der Mittelwertkurve bzw. ist die tatsächliche Geschwindigkeit grösser als der Mittelwert, so ist die Differenz positiv, sonst negativ. Da am Schluss aber eine positive Abweichung entstehen soll, kann man dies mit folgender Formel erreichen.
Gesamtabweichung = $\sqrt{}$ ((positive Abweichungen)^2 + (negative Abweichungen)^2)

→Mit dieser Information kann man herausfinden, ob der Fahrer eher aggressiv oder vorausschauend fährt. Denn je grösser die Gesamtabweichung von der Mittelwertkurve, desto kurviger ist die tatsächliche Geschwindigkeitskurve und desto aggressiver der Fahrer. Es besteht jedoch eine Abfälschungsszene. Denn es kann sein, dass die ständig steigende und sinkende Geschwindigkeitskurve mit der Verkehrssituation zusammenhängt (z.B. Stau). Darum müsste man diese noch mit einem Faktor erweitern, nämlich die Negativbeschleunigung. Naht sich die Gesamtabweichung dem Wert 0 an, so kann man daraus schliessen, dass der Fahrer nicht aggressiv fährt, sondern in einem Stau steht.

11.2.2. Gaspedalstellung über Zeitachse

Wenn man die Gaspedalstellungswerte verstreut auf der Zeitachse beobachtet, erkennt man anhand der Kurvensteigung, wie stark bzw. schnell der Fahrer das Gaspedal runterdrückt. Anhand von diesem Schema, kann man erkennen, ob der Fahrer eher ein Bleifuss- oder ein Roheierfahrer ist.

11.2.3. Korrelation

Um herauszufinden wie sich zwei Werte chronologisch zueinander verhalten, kann man mit der Korrelationen herausfinden, wie ähnlich die sich sind. Der Wertebereich liegt zwischen 0 und 1. 0 bedeutet gar keine Ähnlichkeit. 1 bedeutet identisch. Mehr dazu im Dokument „Wichtige Kennzahlen“ [E2].

11.3. Das Onboard Diagnose System (OBD2)

11.3.1. Übersicht

„On-Board-Diagnose“ oder auch „OBD2“ ist ein standardisiertes Fahrzeugdiagnosesystem. Dieses wird in erster Linie zum Auslesen von Fahrzeugzuständen (Fehlercodes) verwendet. Es liefert je nach Fahrzeugtyp auch aktuelle Informationen wie z.B. Geschwindigkeit, Motordrehzahl, Mass Air Flow (Verbrauchsindikator), Drosselklappen- bzw. Gaspedalstellung. Das ODB2-System brauchen typischerweise auch Autodiagnostiker, um den Zustand eines Fahrzeugs zu überprüfen. Über die OBD-Schnittstelle können Garagisten den Motorzustand abfragen und eventuelle Fehler auslesen. Zudem werden die heutigen Abgaskontrollen auch über die OBD-Schnittstelle erstellt. Weiter Informationen sind auf den Internetseiten [URL1] und [URL2] zu finden.

Der Anschluss von Geräten an das OBD2-System erfolgt über einen genormten Stecker (Abbildung 33), welche maximal 50cm vom Steuerrad entfernt sein soll. Der Stecker befindet sich bei den meisten Fahrzeugen im Bereich der Pedale (

Abbildung 34). Er kann aber auch hinter der Abdeckungen versteckt sein. [URL2] liefert eine Zusammenstellung zu OBD2-Stecker Positionen mit verfügbaren Parametern für verschiedenste Fahrzeuge. Im Rahmen des EcoHelper-Mietsystems soll eine solche Zusammenstellung erweitert und einfach zugänglich gemacht werden.

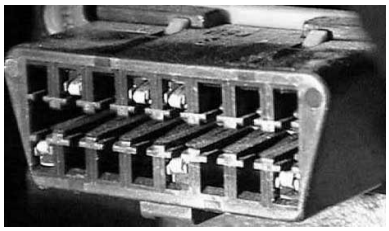


Abbildung 33 OBD-2 Steckbuchse

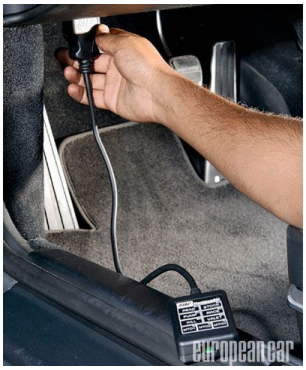


Abbildung 34 Häufige Position der Steckbuchse: Je nach Automarke ist die Steckleiste entweder in der Nähe der Pedale, weiter oben direkt unter dem Lenkrad oder in seltenen Fällen auch oberhalb des Lenkrads

Abbildung 35: Beim Alfa 156 befindet sich die OBD2 Steckdose hinter der Sicherungsabdeckung unter dem Lenkrad

Abbildung 36: Beim Citroen C1 befindet sich die OBD-Steckdose direkt unter dem Lenkrad

Als Alternative zu den teuren, herstellerspezifischen Fahrzeugdiagnosegeräten gibt es mittlerweile verschiedene Interfaces, über welche mittels PC auf OBD2-Daten zugegriffen werden kann. Diese verwenden in der Regel den ELM-Chip [E5]. Diese Interfaces werden direkt oder über ein OBD2-Kabel am OBD2-Stecker des Autos angeschlossen. Die Verbindung zum PC erfolgt via serielle Schnittstelle (RS232), Bluetooth oder WLAN. Abbildung 37 zeigt ein Beispiel einer Bauform eines solchen OBD2-Interfaces über Wifi.

Abbildung 37 Verbindungsmodul der Firma PLX

11.3.2. OBD2 Modi und Parameter

OBD2 ist so aufgebaut, dass über 9 Modi mit je bis zu 32 Parameter abgefragt werden können. [Tabelle 4] zeit die 9 verfügbaren Modi gemäss des OBD2-Standards.

Mode	Bedeutung
1	Zeigt die aktuellen Daten/ Sensorwerte (Echtzeitdaten).
2	Zeigt die zum Anfragezeitpunkt gefrorenen Daten an.
3	Zeigt gesetzte Fehlercodes.
4	Löscht den Fehlerspeicher.
5	Zeigt Selbsttest-Resultate der Lambdasonden.
6	Selbsttest-Resultate nicht kontinuierlich überwachter Systeme.
7	Fehlercode kontinuierlich überwachter Systeme.
8	Spezieller Kontrollmodus
9	Fahrzeuginformationen wie z.B. VIN (Fahrzeug SN.)

Tabelle 4: OBD2-Modes

Ein bestimmter Parameter eines Modes wird über die Parameter Identifikation (PID) adressiert.

Wird 0100 abgefragt, werden 4 Bytes zurückgegeben welche PID's, also welche Werte, vom Auto abgefragt werden können.

Die für uns wichtigsten PID's sind:

Mode (hex)	PID (hex)	Description
01	00	Unterstützte PID's 1-31
01	03	Offene/ Geschlossene Schleife
01	04	Motorlastwert
01	0C	Motordrehzahl
01	0D	Geschwindigkeit
01	10	Mass air flow (Luftdurchfluss)
01	11	Drosselklappenstellung
01	20	Unterstützte PID's
09	02	VIN (Fahrzeugidentifikationsnummer)

Tabelle 5: OBD2-PID's

Es existiert eine komplette Tabelle von allen PID's mit PID-Nummer und Bedeutung. Diese Tabelle finden Sie unter dem Link [URL11].

11.3.3. Abfrage von OBD2-Parametern

Mit der OBD2-Schnittstelle wird ein serieller Datenstrom auf hexadezimaler Basis ausgetauscht. Die Kommunikation zur Abfrage dieser PID's ist relativ einfach.

Der Aufbau eines Befehls, den man der OBD2-Schnittstelle sendet sieht so aus:

[Mode-Nummer][Abstand][PID-Nummer] wie z.B. der Befehl um die Geschwindigkeit (PID 0D = 13) auszulesen = „01 0D".

Einen Befehl abschicken und die Antwort empfangen dauert 200ms. Das bedeutet es können in einer Sekunden 5 Werte abgefragt werden. Die EcoHelper Applikation richtet sich nach dieser Regel und fragt 5 Werte ab. Diese Werte werden dann in der EcoHelper-Applikation im Sekundentakt angezeigt und persisiert.

Beim Versuch die Baudrate einzustellen mussten wir feststellen, dass der ELM-Chip sich nicht mehr ansprechen lies. Damit der Chip wieder benutzt werden konnte mussten wir den Chip kurzschliessen, sodass alle Register zurückgesetzt wurden.

11.3.4. Fahrzeugspezifische OBD-Daten und Authentisierung

Bei der Analyse von verschiedenen Fahrzeugen wurde festgestellt, dass jedes Modell unterschiedliche OBD Werte unterstützte. Alle hatten aber bis jetzt mindestens die Motordrehzahl und die Geschwindigkeit angezeigt.

Weiter wurde festgestellt, dass Herstellerspezifische Diagnosegeräte viel mehr OBD-Werte auslesen konnte als universale Softwarelösungen.

Dies liegt daran, dass die OBD-Schnittstelle eine Authentisierung verlangt. Je nach Authentisierung bekommt man verschieden viele OBD's.

Beim Nachfragen bei verschiedenen Hersteller konnte keiner genaue Auskunft über die Authentifizierung geben. Man hat festgestellt, dass sogar der Hersteller fast nichts darüber weiss.

11.3.5. Beeinflussung des Fahrzeugs über OBD

Über OBD kann man Werte abfragen(lesen), Werte löschen oder Werte überschreiben.

Während den Testfahrten haben wir festgestellt, solange nur Werte abgefragt werden wird das Fahrzeug nicht beeinflusst.

Werden aber während dem Fahren verschiedene Werte gelöscht oder überschrieben ist dies mit einem Ruckeln des Fahrzeuges spürbar.

Es wird empfohlen während der Fahrt nur Werte zu lesen und das Schreiben und Löschen von Werten nur im Stillstand zu machen.

11.4. TourLive Webportal Erweiterungen

11.4.1. Ablage

Die Dokumentation, die angepassten PHP-Skripte und die angepasste DB befinden sich im Projektordner [E6].

11.4.2. Anpassungen an der Datenbank

11.4.2.1. Einleitung

Der Datenbankteil welcher für den Upload zuständig ist besitzt 3 Tabellen (Mobileid, OBD und Positionen).

Die **Tabelle „Positionen“** musste **nicht** angepasst werden.

Die **Tabelle „OBD“** bekommt 2 neue Felder:

CutOff	Dieser wird im Sekundentakt mit den anderen Werten auf dem Server gesendet.
Ecozahl	Dieser Wert wird auch im Sekundentakt ausgerechnet und dem Server gesendet.

Die **Tabelle „Mobileid“** bekommt 3 neue Felder:

Passwort	Das Passwort wird nun auch als Authentifizierung genutzt und dem Server gesendet.
Auto	Da die iPhone Applikation mehrere Autos verwalten kann, werden diese zur Unterscheidung auch dem Webserver gesendet.
Fahrer	Gleich wie beim Auto können mehrere Fahrer verwaltet werden und werden zur Unterscheidung dem Webserver gesendet.

11.4.2.2. ERM

Damit diese Änderung auch schriftlich dokumentiert ist, wurde hier noch die ERM-Erweiterung zur Vollständigkeit hinzugefügt:

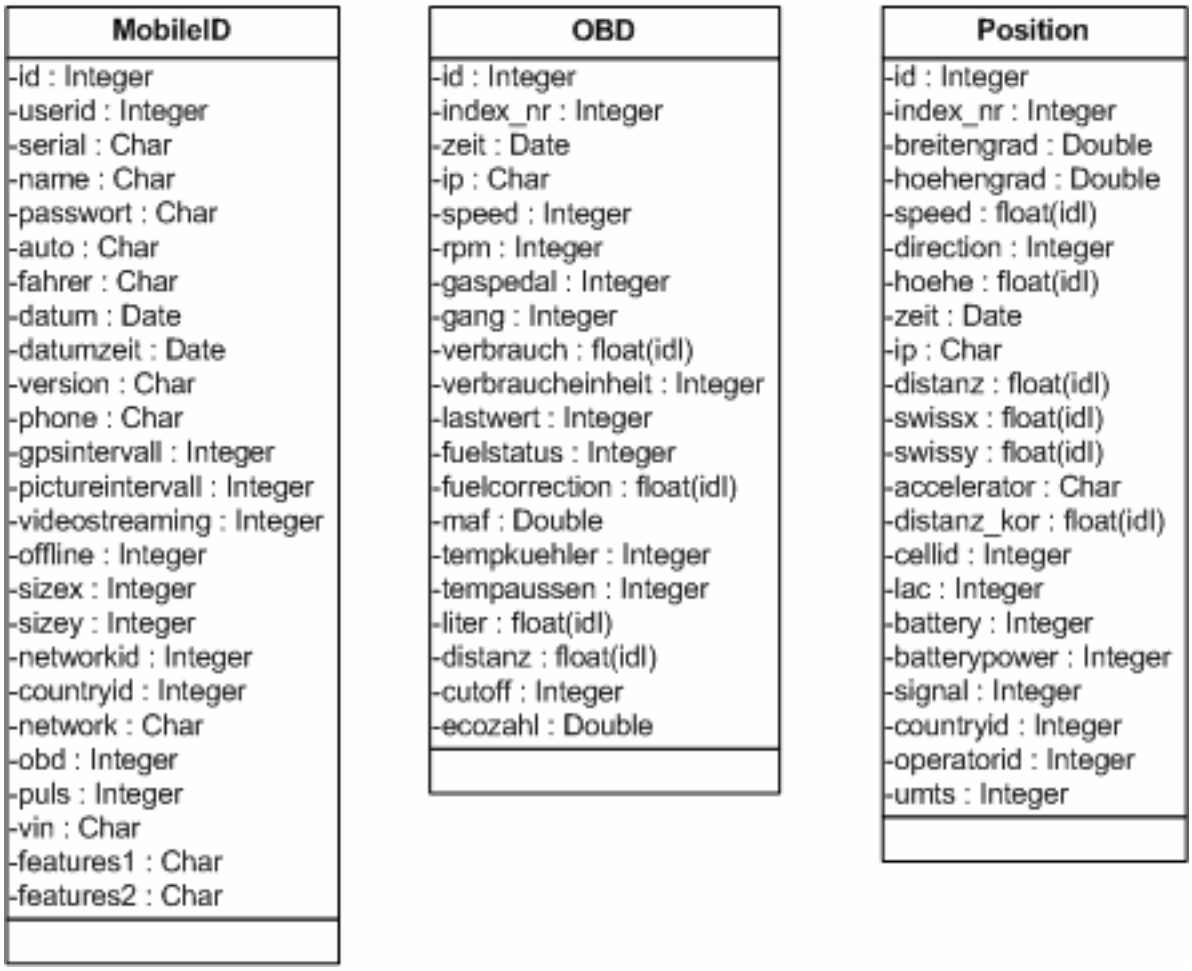


Abbildung 38 ERM des erweiterten TourLive Webportals

11.4.3.Anpassungen an den PHP-Scripts

11.4.3.1. Einleitung

Die Kommunikation zwischen dem iPhone und dem Tourlive Server wird über HTTP GET Request getätigt.

Zuerst wird der Benutzer inkl. seiner Auto- und Fahrer Einstellungen über das Script „getidsym.php“ authentifiziert. Falls die Authentifizierung erfolgreich war sendet der Server der iPhone Applikation eine ID zurück.

Folgende PHP-Scripte mussten angepasst werden:

-getidsym.php Die Felder Passwort, Fahrer und Auto als zusätzliche Parameter mitgeschickt.

-positionsym.php Die Felder Ecozahl und Cutoff als zusätzliche Parameter mitgeschickt.

11.5. Projektplan

11.5.1. Projekt Übersicht

EcoHelper hilft eine umweltfreundliche und kostengünstige Fahrweise zu trainieren. Über den Onboard Diagnose Bus (OBD2) moderner Autos kann die EcoHelper-Handyapplikation Fahrdaten (z.B. Benzinverbrauch, Motordrehzahl und Gaspedalstellung) erfassen. Diese Daten werden mit GPS Positionsdaten und Bilddaten des Handys ergänzt und auf die Web-Plattform [URL8] übermittelt. Es existieren bereits EcoHelper Applikationen für Symbian und für JavaME fähige Mobiltelefone.

11.5.2. Zweck und Ziel.

Ziel dieser Arbeit ist es:

- Das Fahrverhalten mit einfachen Parametern beschreiben und deren Einfluss auf den Benzinverbrauch quantitativ angeben zu können.
- Möglichst viele der für EcoHelper relevanten Fahrzeugdaten mittels iPhone über die OBD2 Schnittstelle auslesen und auswerten können
- Die Fahrzeugdaten sowie anderen relevanten Daten wie GPS, Zeit, Höhe etc. auf das iPhone übersichtlich darstellen.
- und schlussendlich eine EcoHelper Applikation für das iPhone anbieten zu können.

11.5.3. Annahmen und Einschränkungen (Assumptions and Constraints)

Wir gehen von folgenden Annahmen und Einschränkungen aus:

- Das Einarbeiten in die OBD2-Schnittstelle dauert nicht länger als 3 Wochen.
- Das Einarbeiten in die iPhone Entwicklungsumgebung dauert nicht länger als 4 Wochen.
- Die Schnittstellen zu den relevanten Daten sind verfügbar und dokumentiert.
- Die Richtzeit pro Mitglied wird auf 22 Stunden in der Woche angenommen.

11.5.4. Projektorganisation

11.5.4.1. Organisationsstruktur

Projektmitglied	Verantwortung	Aufgaben
Selim Akyol	Entwickler	- Analyse OBD2 - GUI - Datenhaltung - Server-Kommunikation - Evaluation der Fahrtdaten - Testen inkl. Probefahrten
Edon Berisha	Entwickler	- GUI - Datenhaltung - Software Architektur - Kommunikation mit OBD2-Schnittstelle - Umrechnung der Fahrtdaten - Testen inkl. Probefahrten
Peter Heinzmann	Betreuer	-Betreuung und Ansprechperson des Projekts
Omid Afshari	Assistent	-Betreuung und Unterstützung des Projekts
Raphael Juchli	Assistent	-Betreuung und Unterstützung des Projekts

Tabelle 6 Projektorganisation

11.5.4.2. Externe Schnittstellen

Ansprechpartner	Peter Heinzmann, cnlab AG Raphael Juchli, cnlab AG Omid Afshari, cnlab AG
Betreuer	Peter Heinzmann, HSR / cnlab AG
Koreferent	Prof. Dr. Lothar. Müller, HSR
Experte	Dr. Thomas. Siegenthaler, CSI

11.5.5. Management Abläufe

11.5.5.1. Projekt Kostenvoranschlag

Das Projekt startet am 22. Februar 2010 und die funktionsfähige Version muss am 18. Juni 2010 abgeliefert werden. Falls der Arbeitsaufwand grösser als geplant wäre, werden mehr als die vom Studienplan vorgesehenen 22-Stunden pro Woche am Projekt gearbeitet. Wir rechnen mit einer durchschnittlichen Zeit von 30 Stunden pro Woche.

11.5.6. Projektplan

11.5.6.1. Zeitplan

Siehe [E7]

11.5.6.2. Iterationsplanung / Meilensteine

Iterationsplanung		
Von	**Bis**	**Phase**
22.02.10	14.03.10	Inception: Analyse der Aufgabenstellung, Studium der Zusammenhänge verschiedener OBD2-Werte, Fahrverhalten und Verbrauch Anforderungsspezifikation der iPhone Anwendung
15.03.10	28.03.10	Elaboration Iteration 1: Anforderunkspezifikation und Domainanalyse
29.03.10	11.04.10	Elaboration Iteration 2: Internes und Externes Design erstellen
12.04.10	25.04.10	Construction Iteration 1: Proof of Concept aufstellen
26.04.10	09.05.10	Construction Iteration 2: Entwicklung am Code
10.05.10	23.05.10	Construction Iteration 3: Entwicklung des 1. Prototypen
24.05.10	13.06.10	Transition: Abschlussarbeiten für die Final Version & Testfahrten
13.06.10	18.06.10	Reserve

Tabelle 7 Iterationsplanung

Meilensteine			
Nr.	**Titel**	**Beschreibung**	**Datum**
1	Projektplan	Projektplan, Zeitplan, Meilensteine fertiggestellt und vom Betreuer gut geheissen	29.02.10
2	Anforderung und Analyse	Anforderungsspezifikation und Domainanalyse zum Review bereit	28.03.10
	Proof of Concept	Abfrage der Fahrdaten aus der OBD2-Schnittstelle. Demonstration für Betreuer	
3	Design	Internes Design und Designdokumentation sind vollständig und zum Review bereit	11.04.09
4	Construction	Entwicklung am Code abgeschlossen und bereit für Präsentation	23.05.09
5	Transition	Tests erfolgreich abgeschlossen. Dokumentation vollständig und zum Review bereit	13.06.09
6	Schlussabgabe	Abgabe des Projekts	18.06.09

Tabelle 8 Meilensteine

11.5.6.3. Besprechungen

Regelmässige Besprechungen finden in der Regel wöchentlich Anfangs der Woche statt und sonst gemäss Absprache mit den Dozenten. Diese Besprechungen sind gleichzeitig Teammeetings in denen die bisherigen Resultate besprochen werden.

11.5.6.4. Abgabe

Realease	Datum	Funktionalität
Prototyp	25.04.10	Grobe Funktionalität (Verbindung OBD2-iPhone und 1. Anzeige)
Alpha	09.05.10	Kernfunktionalität implementiert (Alle Daten auswertbar)
Beta	23.05.10	Code vollständig Implementiert mit Codeunschönheiten,
Final	13.06.10	Codeunschönheiten beseitigt, letzte Anpassungen vorgenommen, Applikation einsatzbereit

Tabelle 9 Abgabe Termine

11.5.7. Risiko Management

Risiko Bewertungen								
Risk ID	Risiko	Auswirkung	Massnahme	Kosten der Massnahmen in Stunden	Max. Schaden in Stunden	Wahrschein-lichkeit des Eintreffens	Gewichteter Schaden in Stunden	Priorität
R01	Projekt scheitert.	keine ETCS Punkte. BA muss wiederholt werden.	Kontinuierlicher Feedback mit Dozenten.	28	720	5%	36	1
R02	Fachbericht leidet an Qualität	schlechte Fachberichtnote	Fachbericht schon von Beginn an mitschreiben.	20	30	5%	2	1
R03	Zeitplan kann nicht eingehalten werden	Produkt kann nicht zum Abgabetermin fertiggestellt werden	Bei Verzögerungen eventuell Überstunden planen oder Arbeitspackete auslassen.	5	40	50%	20	1
R04	Modul kann Schnittstelle nicht benutzen	Keine Kommunikation zwischen Auto und Iphone möglich	Schnittstelle vor Modulimplementierung analysieren	10	10	30%	3	1
R05	Evaluierung braucht zu viel Zeit	Zeitplan kann nicht eingehalten werden	Testen der Schnittstellen (Proof of Concept)	15	20	20%	4	2
R06	Paket-Auswertung zu komplex	Zeitplan kann nicht eingehalten werden	Einschulung durch Bücher und Internet und durch Hilfeleistung vom Dozenten	5	10	30%	3	2
R07	Mitglied kann sein Arbeitspacket nicht erledigen	Zeitplan kann nicht eingehalten werden	Neuzuweisung des Arbeitspaketes. Betroffener erledigt andere Arbeitspaket	1	20	5%	1	2
R08	Komponenten des Testumfeldes funktionieren nicht	es können keine Tests durchgeführt werden	Komponenten schon in der Inceptionphase ausgiebig testen	10	40	20%	8	1
R09	Änderungswünsche des Betreuers	Projekt wird umfangreicher	Wenn Änderungen nicht zu gross, dann Änderungen	5	30	40%	12	2
	Total Kosten in Arbeitspaketen enthalten			99				
	Total Rückstellungen						89	

Tabelle 10 Risikomanagement

11.5.8. Arbeitspakete

Weitere Arbeitspakete werden am Ende jeder Iteration in den Teammeetings definiert, geschätzt und gemäss Milestones zugeteilt.

1. Inception

1.1	Projektplan	alle	10 h
Beschreibung	Projektplan gemäss Vorlage erstellen und einreichen. Projektorganisation festlegen und Iterationen / Meilensteine planen.		
Abhängigkeiten			
Risiken	-		
Artefakte	Projektplan.doc		

1.2	Projektplan erweitern	alle	5 h
Beschreibung	Projektplan erweitern, Arbeitspakete weiter unterteilen. Verantwortlichkeiten zuteilen.		
Abhängigkeiten	[1.2] Projektplan erstellt.		
Risiken	-		
Artefakte	Projektplan.doc		

1.3	Anforderungsanalyse	seak/edbe	16 h
Beschreibung	Anforderungsanalyse mit Betreuer durchführen und in Pflichtenheft festhalten.		
Abhängigkeiten	-		
Risiken	-		
Artefakte	Anforderungsspezifikation.doc		

1.4	Define Use Cases	seak/edbe	2 h
Beschreibung	Use Cases definieren.		
Abhängigkeiten	Anforderungsanalyse [1.4]		
Risiken	-		
Artefakte	Anforderungsspezifikation.doc		

1.5	Aufbau Testumgebung	seak/edbe	8 h
Beschreibung	Installation aller benötigten Soft- und Hardwares.		
Abhängigkeiten	Anforderungsanalyse [1.4]		
Risiken	-		
Artefakte	-		

1.6	Proof of Concept	seak/edbe	9 h
Beschreibung	Testprogramm schreiben: Auslesen der benötigten Daten und 1. Anzeige auf dem iPhone		
Abhängigkeiten	Aufbau Testumgebung [1.6]		
Risiken	-		
Artefakte	Object C Project mittels iPhone SDK auf dem SVN Server		

1.8	Glossary	edbe	1 h
Beschreibung	Glossar beginnen.		
Abhängigkeiten	-		
Risiken	-		
Artefakte	Glossar.doc		
1.9	**Analyse verschiedener Automodelle**	**seak & edbe**	**10**
Beschreibung	Mittels der Software ScanMaster verschiedene Automodelle untersuchen und schauen welche PID (Fahrparameter) unterstützt werden		
Abhängigkeiten	-		
Risiken	-		
Artefakte	Fahrdatenanalyse.doc und OBD2Fahrzeugtests.xls		

1.10	Analyse verschiedener Fahrten	Seak & edbe	10
Beschreibung	Mittels der Symbian EcoHelper Anwendung verschiedene Strecken mit verschiedenen Fahrtypen abfahren und analysieren. Die wichtigsten Werte in Zusammenhang betrachten und diese dokumentieren.		
Abhängigkeiten	-		
Risiken	-		
Artefakte	Fahrdatenanalyse.doc		

2. Elaboration

2.1	OBD2 und iPhone Daten auslesen	seak & edbe	10 h
Beschreibung	Modellierung der ersten auswertbaren Daten.		
Abhängigkeiten	-		
Risiken	-		
Artefakte	PD & Klassendiagramm in UML, Domainanalyse.doc		

2.2	Analyse des Frameworks (Klassenhierarchie, Kernmethoden)	seak & edbe	30 h
Beschreibung	Modellierung der wichtigsten Klassen und Methoden.		
Abhängigkeiten	-		
Risiken	-		
Artefakte	PD & Klassendiagramm in UML, Domainanalyse.doc		

2.3	Domain Analyse	seak & edbe	9 h
Beschreibung	PD und Klassendiagramm in UML.		
Abhängigkeiten	-		
Risiken	-		
Artefakte	PD & Klassendiagramm in UML, Domainanalyse.doc		

2.3.1	Problem Domain	seak & edbe	9 h
Beschreibung	PD: Mögliche Klassen und verantwortliche Methodenimplementierungen analysieren. Schnittstellen (Methoden, Formate, etc.) analysieren und weitere Designentscheidungen berücksichtigen. Analyse der PD.		
Abhängigkeiten	-		
Risiken	-		
Artefakte	PD & Klassendiagramm in UML, Domainanalyse.doc		

2.4	Speicherstrukturen & Design	seak & edbe	27 h
Beschreibung	Implementierungsentscheide & Klassendiagramm in UML.		
Abhängigkeiten	-		
Risiken	-		
Artefakte	PD & Klassendiagramm in UML, Domainanalyse.doc		

2.4.1	Implementierungsentscheidung & -modellierung (Speicher- und Zeitverbrauch)	seak & edbe	24 h
Beschreibung	**Analyse der Implementierungsmöglichkeiten der Auswertung der benötigten Daten und Anzeige auf das iPhone..**		
Abhängigkeiten	**-**		
Artefakte	PD & Klassendiagramm in UML, Domainanalyse.doc		

2.4.2	Klassendiagramm	edbe	3 h
Beschreibung	**Klassendiagramm in UML designen.**		
Abhängigkeiten	**-**		
Risiken	**-**		
Artefakte	Klassendiagramm in UML		

2.5	Refinement of Use Cases	seak & edbe	4 h
Beschreibung	UCs genau beschreiben.		
Abhängigkeiten	Define Use Caes [1.4]		
Risiken	-		
Artefakte	Anforderungsspezifikation.doc & Domainanalyse.doc		

2.5.1	All Ucs Fully dressed	seak	2 h
Beschreibung	Alle UCs im fully dessedformat beschreiben.		
Abhängigkeiten	Define Use Caes [1.4]		
Risiken	-		
Artefakte	Anforderungsspezifikation.doc		

2.5.2	SSD & Contracts	edbe	2 h
Beschreibung	SSD & Contracts zu allen UCs.		
Abhängigkeiten	Define Use Cases [1.4]		
Risiken	-		
Artefakte	Domainanalyse.doc		

2.6	Coding (1. Prototyp)	edbe & seak	64 h
Beschreibung	Ersten Prototypen mit Speicherung und Auswertung.		
Abhängigkeiten	-		
Risiken	-		
Artefakte	Erster Prototyp (Objective C auf SVN-Server)		

2.6.1	Problem Domain	edbe & seak	64 h
Beschreibung	Wichtigste Klassen und mindestens eine View (z.B. Benzinverbrauch) implementieren.		
Abhängigkeiten	-		
Risiken	-		
Artefakte	Erster Prototyp (Objective C auf SVN Server)		

2.7	Testen		6 h
Beschreibung	Schreiben von Modultests und Testen der Hauptfunktionalität.		
Abhängigkeiten	-		
Risiken	-		
Artefakte	Testprotokoll.doc		

2.8	Demonstration und Feedback		2 h
Beschreibung	Demonstration des Prototypen bei den Betreuern und Feedback einholen.		
Abhängigkeiten	-		
Risiken	-		
Artefakte	Sitzungsprotokoll.doc		

3. Construction Iteration 1

3.1	Domain Model	Seak Edbe	5
Beschreibung	PD und Klassendiagramm anpassen gemäss Betreuerfeedback.		
Abhängigkeiten	Demonstration und Feedback [2.6]		
Risiken	-		

3.2	Coding	Seak Edbe	50
Beschreibung	Entwicklung der Anpassungen.		
Abhängigkeiten	-		
Risiken	-		

3.2.1	Problem Domain	Seak Edbe	50
Beschreibung	Umsetzen der PD-Anpassungen.		
Abhängigkeiten	Domain Model [3.1]		
Risiken	-		

3.3	Testen	Seak Edbe	10
Beschreibung	Prototyptest.		
Abhängigkeiten	-		
Risiken	-		

3.4	Demonstration und Feedback	Seak Edbe	10
Beschreibung	Demonstration des Prototypen bei den Betreuern und Feedback einholen.		
Abhängigkeiten	-		
Risiken	-		

4. Construction Iteration 2

4.1	Domain Model		4
Beschreibung	PD und Klassendiagramm gemäss Betreuerfeedback anpassen.		
Abhängigkeiten	Demonstration und Feedback [3.4]		
Risiken	-		

4.1	Coding		30
Beschreibung	Entwicklung der kompletten Applikation.		
Abhängigkeiten	-		
Risiken	-		

4.1.1	Problem Domain		30
Beschreibung	Umsetzen der PD-Anpassungen.		
Abhängigkeiten	Domain Model [4.1]		
Risiken	-		

4.2	Demonstration und Feedback		10
Beschreibung	Demonstration des Prototypen bei den Betreuern und Feedback einholen.		
Abhängigkeiten	-		
Risiken	-		

5. Transition

5.1	Coding Final	Seak Edbe	30
Beschreibung	Fertigstellen der Applikation und Beseitigung von Codeunschönheiten.		
Abhängigkeiten	-		
Risiken	-		

5.2	Code Review	Seak Edbe	25
Beschreibung	Code Review und nächsten Release planen. (nach dem Projekt)		
Abhängigkeiten	-		
Risiken	-		

5.3	Abgabe	Seak Edbe	40
Beschreibung	Alle nötigen Artefakte für die Abgabe vorbereiten, zusammenstellen und abgeben.		
Abhängigkeiten	-		
Risiken	-		

11.5.9. Infrastruktur

11.5.9.1. Koordination

Zur Synchronisation der Projektentwicklung verwenden wir einen SVN-Server der HSR. Dieser dient zugleich für die Datenablage des Source-Codes sowie als Ablage für die Projektdokumentation.

11.5.9.2. Arbeitsumgebung

Folgende Hard- & Software brauchen wir während der Entwicklung unseres Projektes in unserer Testumgebung.

11.5.9.3. Hardwares

Gerät	Hersteller	Schnittstelle
Laptops	Apple	Wifi, GSM
iPhone	Apple	Wifi, GSM
PLX Wifi Adapter	PLX	Wifi
OBD-Simulator	PLX	Wifi, OBD

11.5.9.4. Softwares

	Name	Hersteller	Version
Betriebssysteme	Windows, MAC OS	Microsoft/ Apple	XP/ X, Leopard
	Windows Server	Microsoft	2003
Entwicklungsumgebung	Eclipse	-	3.2.2
	Subclipse	-	1.2
	xCode mit iPhone SDK	Apple	3.2.3
Office	Word	Microsoft	200x
	Excel	Microsoft	200x

11.5.10. Qualitätsmassnahmen

Um für unser Produkt sowie dem gesamten Projektverlauf eine möglichst hohe Qualität zu erreichen, werden folgende Massnahmen unternommen.

11.5.10.1. Richtlinien

Die verwendeten Coderichtlinien basieren auf den Apple Programmierrichtlinien [URL9]

11.5.10.2. Code

- Alle Namen, wie Variablen und Methoden sind auf Englisch.
- Kommentare sind in Deutsch gefasst.

11.5.10.3. Codereviews

- Wöchentlich bzw. bei jedem Teammeeting wird ein Codereview durchgeführt.
- Testprotokolle werden in den Teammeetings besprochen.

11.5.10.4. Usability-Tests

- Entwickler erstellen Usability-Tests und führen sie selbst durch
- Usability-Tests werden von Testpersonen ausgeführt und dokumentiert.

11.5.11. Prozessqualität

11.5.11.1. Diskussionen und Reviews

Die Projektmitglieder und Betreuer kommen einmal in der Woche zusammen um über erreichte Ziele und den nächsten nötigen Schritten zu diskutieren. Dabei schätzen wir den benötigten Aufwand für die nächsten Schritte ein und schlagen Realisierungs- und Vorgehensmöglichkeiten vor. Erreichte Ziele und unternommene Schritte werden kurz diskutiert. Wo nötig werden Verbesserungsvorschläge eingebracht und neue Termine festgelegt.

Entscheidungen werden schriftlich in Form von Sitzungsberichten festgehalten und, wo nötig, in die entsprechenden Stellen der Dokumentation aufgenommen. Die Sitzungsberichte werden nur für interne Zwecke verwendet, weshalb sie nicht zur Dokumentation gehören.

Um Missverständnisse unter den Projektmitgliedern zu vermeiden, finden unter der Woche, je nach bedarf, Kurzdiskussionen von ca. 5-10 Minuten statt.

11.5.11.2. Management und Verwaltung

Arbeitspakete und Teildokumentationen werden so verteilt, dass sich die Teammitglieder nicht überschneiden und/oder gegenseitig blockieren können. Sollte es dennoch zu Überschneidungen oder Blockaden kommen, wird die Arbeitsaufteilung neu diskutiert und aufgeteilt.

11.5.11.3. Dokumentation

Die Dokumentation wird je nach Aufgabenzuteilung aufgeteilt. Jeder erstellt die Teildokumentation für sich. Zeiterfassung und die persönliche Berichte werden nachgefasst, alle anderen Dokumente sind zu Beginn des jeweiligen Arbeitspakets zu erstellen und nach dem Beenden des Pakets nachzutragen.

Andere Teammitglieder können Verbesserungsvorschläge bringen oder auf Fehler hinweisen. Jedoch ist der Autor für die Überarbeitung zuständig. Zum Schluss werden die einzelnen Teile in ein Dokument von einem Mitglied zusammengenommen und von den übrigen gelesen. Überarbeitungen finden dann nur noch in der zusammengenommenen Dokumentation statt.

11.5.12. Produktqualität

11.5.12.1. Funktionalität

Die Funktionalität des Produktes richtet sich nach dem Dokument *Anforderungsspezifikation.doc* und ist auf das Dokument *Domainanalyse.doc im Verzeichnis [E1]*gestützt. Änderungen, Erweiterungen oder Einschränkungen der Funktionalität am Programmcode werden erst dann vorgenommen, wenn die Funktionsänderung säuberlich und eindeutig in den entsprechenden Dokumenten beschrieben wurde. Änderungen in den Dokumenten werden unverzüglich an alle Mitglieder kommuniziert.

Die entwickelten Funktionen werden, je nach Komplexitätsgrad, von mindestens einem weiteren Mitglied kurz auf Richtigkeit und Konformität überprüft. Fehler oder Unschönheiten werden gleich selber korrigiert oder dem zuständigen Entwickler kommuniziert.

11.5.12.2. Benutzbarkeit

Das Produkt wird als eigenständige Applikation auf dem iPhone installiert. Mittels Wifi-Verbindung und Adapter kann es die fahrzeugrelevanten Daten auslesen. Die restlichen benötigten Daten werden direkt vom iPhone zur Verfügung gestellt. Die Applikation wird so aufgebaut, dass Daten online sowie auch offline gespeichert werden können. D.h. die Daten werden entweder direkt auf der Webseite verschickt oder auf dem iPhone gespeichert und bei WiFi-Verbindung dann auf den Server geschickt.

Zudem können einzelne Komponenten der Applikation mittels Apple iPhone Virtualisierung direkt am PC getestet werden. Der Aufbau wird so gestaltet, dass sich Änderungen am Informationsgehalt leicht vornehmen lassen.

11.5.12.3. Ausfallsicherheit

Durch geeignete Unit- und Usertests wird die Fehlertoleranz des Systems erhöht. Fehleingaben durch den Benutzer sind zu testen. Das Verhalten bei Fehlerzuständen wird bei der Entwicklung getestet. Die Systemdaten sind in jeder erdenklichen und nichterdenklichen Situation wiederherzustellen. Serverseitig werden alle Änderungen protokolliert.

11.5.12.4. Effizienz / Leistung

Antwort- und Verarbeitungszeiten des Systems sind in der *Anforderungsspezifikation.pdf* zu finden. Das Auslesen und Anzeigen der relevanten Daten spielt für uns eine primäre Rolle, da diese kontinuierlich und möglichst zeitgleich angezeigt und ausgewertet werden müssen.

11.6. Beurteilung der Projektplanung

	SOLL	IST
vor Projektstart	**17.0**	**16.0**
Inception	**164.0**	**140.0**
Elaboration (1. Iteration)	**105.0**	**79.0**
Elaboration (2. Iteration)	**106.0**	**132.0**
Construction (1. Iteration)	**148.0**	**144.0**
Construction (2. Iteration)	**143.0**	**139.0**
Construction (3. Iteration)	**133.0**	**156.0**
Transition	**148.0**	**311.0**
Reserve	**0.0**	**112.0**
Total	**964.0**	**1229.0**

Tabelle 11 Auflistung der geleisteten Stunden

Wie in Tabelle 11 ersichtlich ist, wurden über 1200 Stunden für dieses Projekt geleistet.

Es wurde zwar von Anfang an mit einem Zeitaufwand von ca. 1000 Stunden gerechnet. Die Vorgabe von der HSR war ein Stundenaufwand von mindestens 720 Stunden zu erbringen.

Allerdings mussten wir uns in 2 Technologien (iPhone und OBD2) einarbeiten und zusätzlich noch die Fahrdatenanalyse, dass auch ein Bestandeil der Arbeit ist ein, erstellen.

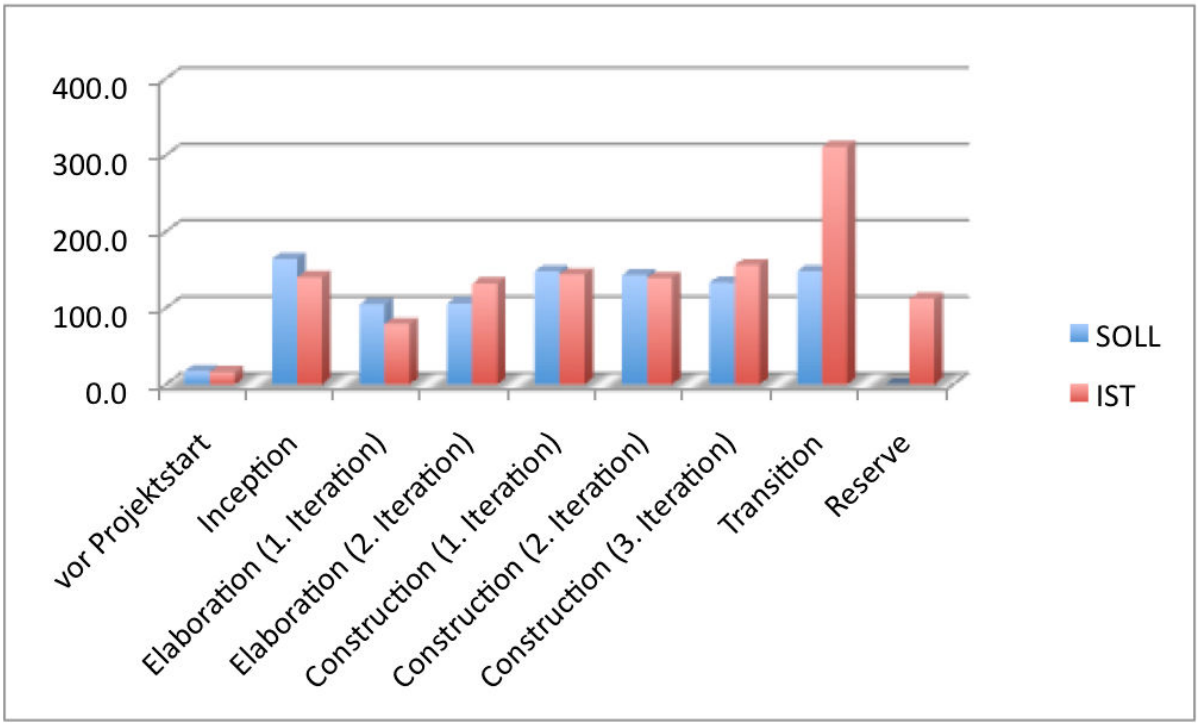


Abbildung 39: Balkendiagramm der geleisteten Stunden

Die Abbildung 39 veranschaulicht in welcher Phase des Projektes wie viele Stunden geleistet wurden. Man erkennt, dass die Soll und Ist Stunden in den meisten Phasen mehr oder weniger gleich sind nur am Schluss musste in der Transition und in der Reserve mehr geleistet werden als geplant. Der Hauptgrund für diesen Mehraufwand war, dass es ein Problem bestand die Verbindung zwischen dem Auto und des iPhones unterbruchfrei zu halten. Doch konnten wir das Problem dank dem Mehraufwand zum Schluss noch lösen.

11.7. Benutzeranleitung

11.7.1. Hardware Konfiguration

Folgende Komponenten sollten vorhanden sein:

-EcoHelper V1.0 auf dem iPhone installiert.

-Das Fahrzeug ist ein manuell geschalteter Benziner mit einer OBD2-Buchse.

-Der Kiwi-Wifi Adapter von PLX-Devices ist am Auto angeschlossen und eingeschaltet.

11.7.2. Anleitung

11.7.2.1. Registrierung

Die EcoHelper Applikation ist im Apple Store verfügbar.

Unter der Suche den Begriff „EcoHelper“ eingeben und die Applikation anschliessend auf dem iPhone installieren.

Anschliessend muss man sich ein Account beim EcoHelper Server beschaffen.

Die Registrierung kann unter http://www.tourlive.ch/ecohelper/ vorgenommen werden.

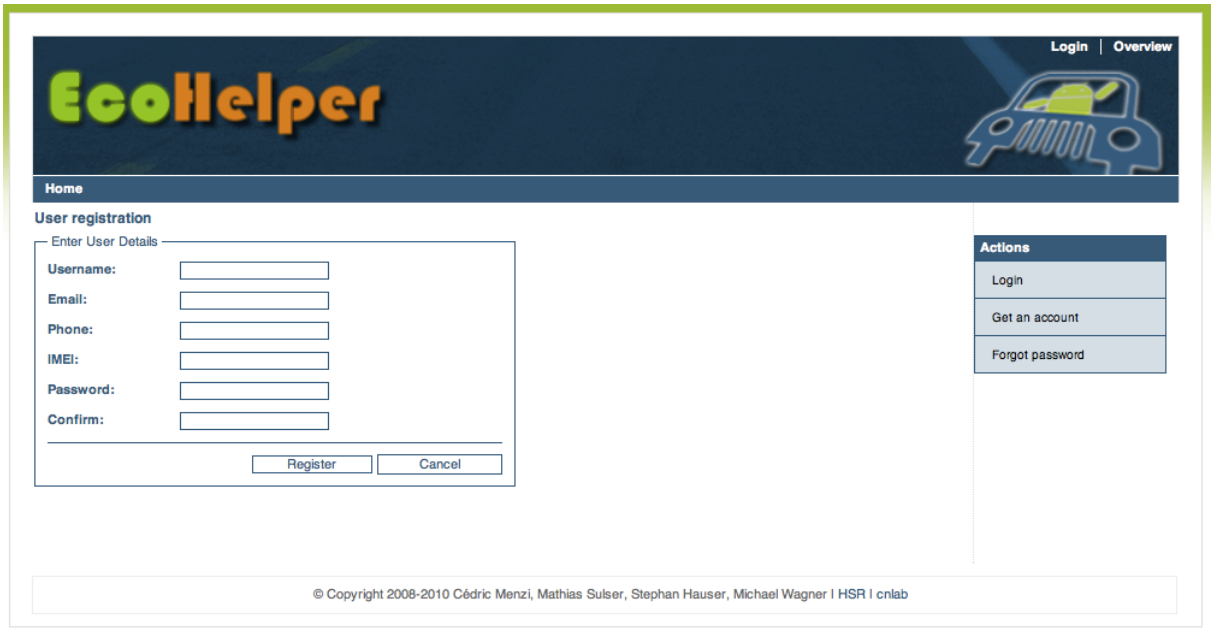


Abbildung 40 Registrierung auf dem TourLive Server

11.7.2.2. Installation des OBD-Adapters

Zuerst muss die OBD-Steckdose [Bild 1] im Auto gefunden werden. Die Steckdose sollte laut Richtlinien maximal 50cm vom Lenkrad entfernt sein. Falls man die OBD-Steckdose nicht auf Anhieb findet, sollte man in der Betriebsanleitung des Fahrzeuges nachschauen.

Anschliessend verbindet man den PLX Kiwi-Wifi OBD-Adapter [Bild 2] mit dem Fahrzeug über die OBD-Steckdose. Sobald alles richtig angeschlossen ist leuchtet das PLX-Logo rot und das Lämpchen "Link" blinkt grün.

WICHTIG: Der Motor muss eingeschaltet sein bevor in der EcoHelper Applikation eine neue Aufnahme gestartet wird.

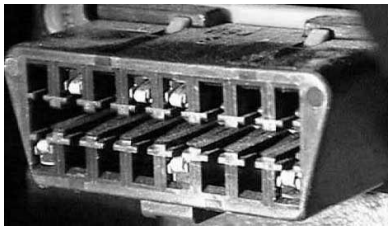


Bild 1: OBD Steckdose

Bild 2: Angeschlossener PLX Device

11.7.2.3. PLX auf dem iPhone konfigurieren

Bild 3: PLX-Wifi Adapter Einstellung

Damit der PLX Wifi Adapter korrekt mit der EcoHelper Applikation kommuniziert müssen die Wi-Fi Einstellungen richtig eingestellt werden.

Zuerst geht man in die Netzwerk-Einstellungen des iPhones [Bild 3]. Dort wählt man den "PLXDevices" als Wi-Fi aus und tippt auf den blauen Pfeil des "PLXDevices"-Wi-Fi.

In dieser Ansicht stellt man dann die IP-Addresse auf statisch und setzt folgende Felder:

IP-Adresse: **192.168.0.11**

Teilnetzmaske: **255.255.255.0**

Alle anderen Felder werden leer gelassen.

11.7.2.4. EcoHelper Applikation starten

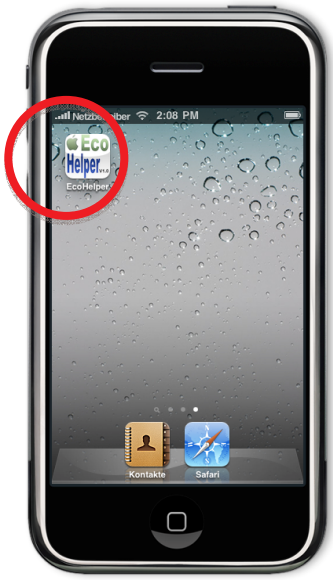


Bild 4 Installierte EcoHelper Applikation auf einem iPhone

Auf das obige Bild [Bild 4] sieht man die installierte EcoHelper Version1.0 Applikation auf einem iPhone. Um die Applikation zu starten, einfach auf das EcoHelper Icon tippen.

11.7.2.5. Profil erstellen / bearbeiten

Bild 5: Home Ansicht

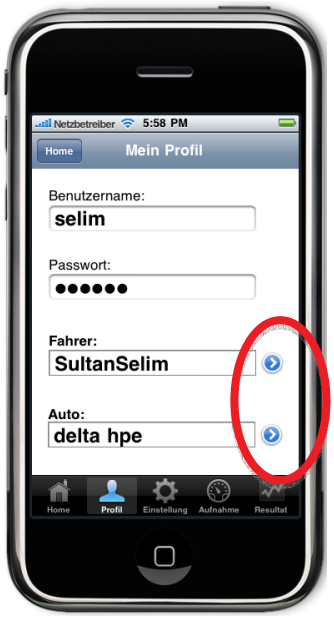


Bild 6: Profil

Bild 7: Tabbar Navigation

Nach dem Start der Applikation gelangt man zur Home Ansicht der Applikation [Bild 5]. Dort wird der aktuell gesetzte Fahrer und das aktuell gesetzte Autos angezeigt. Bei der 1. Inbetriebnahme sind noch keine Fahrer und Autos gesetzt.

Mittels der Tabbar [Bild 7] am unteren Ende der Applikation kann man durch die EcoHelper Software navigieren. Tippt man auf das Profil Icon, gelangt man auf die Seite wo man Fahrer, Autos, Benutzername und das Passwort bearbeiten kann [Bild 6].

In den Feldern Benutzername und Passwort gehören die von der Tourlive Webseite registrierten Zugangsdaten.

Durch das Tippen auf den kleinen Navigationspfeilen neben dem Auto- und Fahrer- Textfeld gelangt man auf die entsprechenden Verwaltungsansicht.

11.7.2.6. Fahrer erstellen / bearbeiten

Bild 8: Fahrer Verwaltung

Bild 9: Fahrer erstellen / bearbeiten

Auf der Fahrerverwaltungsseite [Bild 8] wird eine Liste der verfügbaren Fahrer angezeigt. Zu jedem gewählten Fahrer werden unten die wichtigsten Informationen angezeigt. Zu Beginn sind keine Fahrer erfasst. Will man neue Fahrer erfassen, bearbeiten oder gar löschen, so kann man dies über die obige Navigationsleiste tun. Das "+" steht für einen neuen Fahrer, der Edit-Button um einen bestehenden zu bearbeiten und das Papierkorb-Icon um einen Fahrer zu löschen.

Durch das Tippen auf den "+" oder "Edit" -Icon gelangt man auf die Erfassungs- / Bearbeitungsseite [Bild 9]. Auf dieser Seite kann man dann den Fahrer editieren. Mittels dem "Sichern"- Knopf oben rechts oder mittels dem "Abbrechen"-Knopf oben links gelangt man zurück auf die Fahrerverwaltungsseite.

11.7.2.7. Auto erstellen / bearbeiten

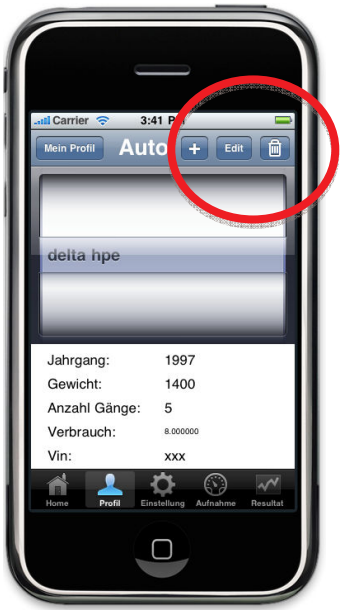


Bild 10: Auto Verwaltung

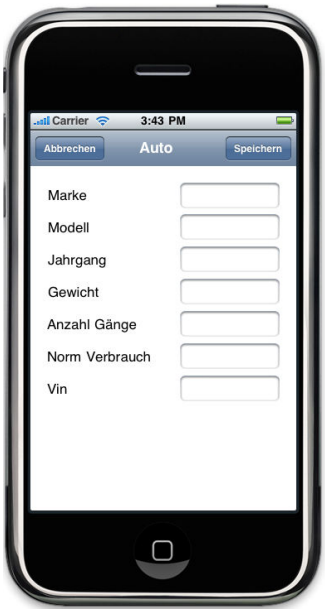


Bild 11: Auto erfassen / bearbeiten

Durch das Tippen auf den Navigationspfeil des Autos [Bild 6] gelangt man auf die Autoverwaltungsseite [Bild 10]. Auf der Autoverwaltungsseite wird eine Liste der verfügbaren Autos angezeigt. Zum gewählten Auto werden unten die wichtigsten Informationen angezeigt. Zu Beginn sind keine Autos erfasst. Will man neue Autos erfassen, bearbeiten oder gar löschen, so kann man dies über die obige Navigationsleiste tun. Das "+" steht für ein neues Auto, der „Edit“-Knopf um ein bestehendes zu bearbeiten und der Papierkorb-Knopf um eine Auto zu löschen.

Durch das Tippen auf den "+" oder "Edit" -Icon gelangt man auf die Erfassungs- / Bearbeitungsseite [Bild 11]. Auf dieser Seite kann man dann das Auto editieren. Mittels dem "Speichern"- Icon oben rechts oder mittels dem "Abbrechen"-Icon oben links gelangt man zurück auf die Autoverwaltungsseite.

11.7.2.8. Einstellungen vornehmen

Bild 12: Einstellungen

Tippt man auf der Tabbar Navigation unten auf das Einstellungen-Icon gelangt man auf die Einstellungsseite [Bild 12]. Auf dieser Seite können folgende Einstellungen vorgenommen werden.

Onlinemodus: Stellt man den Onlinemodus auf "Ein" so werden die gesammelten Daten während der Fahrt direkt an den Tourlive Server gesendet. Ist der Onlinemodus ausgeschaltet werden die Daten erst bei manueller Auswahl dem Tourlive zu einem späteren Zeitpunkt zugesendet. Es empfiehlt sich den Onlinemodus auszuschalten falls man kein geeignetes Datenabo mit seinem Handyabo abgeschlossen hat.

Hintergrundbeleuchtung: Zusätzlich ist es möglich die Hintergrundbeleuchtung einzuschalten. Das bedeutet, dass das iPhone während der Fahrt nicht in den Schlafmodus geschaltet wird.

Standardanzeige: Hier kann der Benutzer zwischen einer einfachen und einer detaillierten Erstansicht wählen. Natürlich kann zwischen den einzelnen Ansichten während der Fahrt direkt gewechselt werden.

11.7.2.9. Aufnahme starten

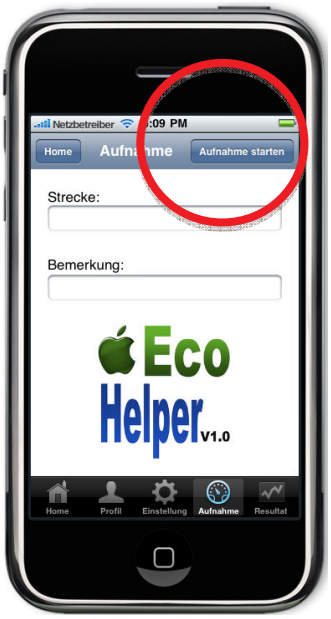


Bild 13: Aufnahme erfassen

Mittels der Tabbar Navigation kann man auf die Aufnahmen Ansicht [Bild 13] wechseln. Folgende Punkte müssen erledigt sein bevor eine neue Fahrt aufgenommen wird:

-PLX Device muss an die OBD-Steckdose angeschlossen sein

-Der Motor muss eingeschaltet sein

-Es muss ein Fahrer und ein Auto gesetzt sein

-Benutzername und Passwort müssen korrekt eingegeben werden

Sobald diese Punkte erfüllt sind kann man die Fahrt per Knopf "Aufnahme starten" beginnen.

11.7.2.10. Aufnahme einer Fahrt

Bild 14: Momentaufnahme während der Fahrt

Sobald man die Fahrt gestartet hat [Bild 13] wechselt man in die Breitenansicht des iPhone's. Dies ist die aktuelle Anzeige während der Fahrt [Bild 14].

Während der Fahrt kann problemlos zwischen der einfachen und der detaillierten Ansicht gewechselt werden. Die Fahrtaufnahme geht ohne Unterbruch weiter.

Sobald man die Fahrt beendet hat tippt man auf den Button "Auswertung". Sobald dieser Button gewählt wurde wird die Fahrtaufnahme beendet und man gelangt in die Auswertungsansicht.

11.7.2.11. Verbindungsverlust während der Fahrt

Da jedes Auto unterschiedliche Steuergeräte besitz, variieren die Antwortzeiten über die OBD2-Schnittstelle. Es kann auch passieren, dass die Verbindung von der OBD2-Schnittstelle wegen einer Zeitüberschreitung abgebaut wird. In diesem Fall sollte man die Verbindung zwischen dem PLX-Adapter und dem Auto überprüfen.

Als zusätzliche Kontrolle dient auch der PLX-Adapter. Dieser hat ein kleines Lämpchen "Link" [Bild 2]. Falls während der Fahrt das Lämpchen ständig leuchtet besteht eine Verbindung. Falls aber das Lämpchen während der Fahrt nur blinkt ist die Verbindung fehlgeschlagen.

11.7.2.12. Auswertung

Bild 15: Fahrten Verwaltung

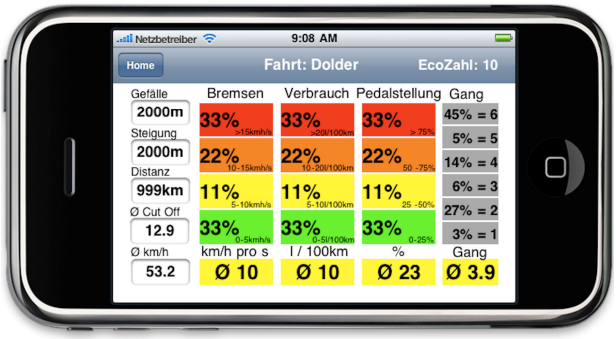


Bild 16: Statistische Auswertungsanzeige

Wird eine Fahrt beendet gelangt man automatisch auf der Fahrtenverwaltungsansicht [Bild 15]. In dieser Ansicht wird eine Liste der gespeicherten Fahrten angezeigt.

Man kann die einzelnen Fahrten auswählen und es werden unten immer automatisch die dazugehörigen Informationen angezeigt. Sobald man eine Fahrt ausgewählt hat, tippt man auf den Knopf "Fahrt auswerten".

Anschliessend gelangt man in die horizontale Ansicht, welche die ganze Fahrt statistisch auswertet [Bild 16]. In dieser Ansicht werden alle Durchschnitte und Summenhäufigkeiten der erfassten Daten angezeigt.

Durch das Tippen auf das "Home" Icon gelangt man zurück zur Home-Ansicht.

11.8. Sitzungsprotokolle

Die Sitzungen mit dem Team wurden mit entsprechenden Sitzungsprotokollen dokumentiert. Diese Dokumente befinden sich im Projektordner unter [E8]. Die Sitzungsprotokolle besitzen selber auch Versionsnummern, da diese von Betreuerseite immer wieder ergänzt wurden und wurden nach Datum archiviert.

11.9. Testprotokolle

Die Software wurde von 2 Testpersonen auf die Benutzung hin getestet. Diese Testprotokolle und Erkenntnisse wurden in Testprotokollen im Projektordner unter [E9] gespeichert.

11.10. Email

Der komplette Email-Verkehr betreffend dieser Bachelorarbeit wurde im Projektordner unter [E10] gespeichert.

11.11. Zeiterfassung

Die persönlichen Zeiterfassungsdokumente wurden im Projektordner unter [E11] gespeichert.

11.12. Präsentation

Die verschiedenen Präsentation die während der Arbeit durchgeführt wurden sind im Projektordner unter [E12] archiviert.