Bachelorarbeit

Daniel Lorenz

Lenkraddisplays eines Formula Student Rennwagens: von der Usability Analyse, über das Interface- und Funktionsdesign bis zur QT basierten Realisierung der Softwarearchitektur

Fakultät Technik und Informatik
Department Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Daniel Lorenz

Lenkraddisplays eines Formula Student Rennwagens: von der Usability Analyse, über das Interface- und Funktionsdesign bis zur QT basierten Realisierung der Softwarearchitektur

Bachelorarbeit eingereicht im Rahmen der Bachelorprüfung
im Studiengang Technische Informatik
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer : Prof. Dr. Franz Korf
Zweitgutachter : Prof. Olaf Zukunft

Abgegeben am 25. August 2008

Daniel Lorenz

Thema der Bachelorarbeit

Lenkraddisplay eines Formula Student Rennwagens: Von der Usability Analyse, über das Interface- und Funktionsdesign bis zur QT basierten Realisierung der Softwarearchitektur

Stichworte

Usability, Multifunktionslenkrad, QT, Embedded, HAWKS, Rennwagen, CAN, XML, CAN-Socket

Kurzzusammenfassung

Gegenstand dieser Arbeit ist die Entwicklung eines Multifunktionslenkrades mit Display für einen Rennwagen. Anhand Usability-Analysen werden die Anforderungen an das System ermittelt. Anschließend wird ein Prototyp der Bedienoberfläche entwickelt und ein Software-Design erstellt. Für die Umsetzung der Software wird QT Embedded verwendet. Zudem befasst dich diese Arbeit mit der Evaluierung der Software im Usability-Labor und der Verwendung eines CAN-Sockets.

Daniel Lorenz

Title of the paper

Steering wheel display of a formula student racing car: From the usability analysis, throw the interface- and functional-design up to the implementation of a software architecture basing on QT

Keywords

usability, multifunctional steering wheel, QT, embedded, HAWKS, racing car, CAN, XML, CAN-Socket

Abstract

The object of the assignment is the development of a multifunctional steering wheel for a racing car. On the basis of usability analysis the requirements for the system will be determined. Afterwards a prototype of user interface will be developed and a software-design created. For the implementation of the software QT Embedded is used. Furthermore this assignment occupying the evaluation of the software at the usability lab and the usage of CAN-socket.

Inhaltsverzeichnis

1 Einführung

In dieser Bachelor-Arbeit geht es um die *Usability* (siehe Seite 12) gerechte Gestaltung eines Lenkraddisplays und dessen Software. Dabei wird auf einzelne Techniken der Usability und Software-Engineering[G] eingegangen.

Abbildung 1.1: HAWK08

Zur Realisierung der Software für das Lenkrad-Display wird QT-Embedded[G] benutzt (auch unter dem Namen Qtopia bekannt), ein Grafik-Framework in der Programmiersprache C++, dass zum Laufen ein Linux Embedded[G] benötigt. Die Software soll über ein selbst gestaltetes Interface am Lenkrad steuerbar sein. Zudem wird in dieser Arbeit das Programmieren eines CAN-Sockets behandelt, welcher die Kommunikation über einen CAN-Bus[G] auf einem Linux Embedded erleichtert.

Das HAWKS Racing Team der Hochschule für Angewandte Wissenschaften Hamburg nimmt sich jedes Jahr vor einen eigenen Rennwagen für die Formula Student (siehe Kapitel 1.2) zu konstruieren. Dieses Jahr(2008) soll der Wagen eine eigens konstruierte Anzeige im Cockpit erhalten. Somit wurde die Informatik-Baugruppe des HAWKS Teams mit dem Entwurf der Elektronik und der Software des Lenkrades beauftragt.

Die Arbeit gliedert sich in die klassischen groben Usability-Methoden. Von der Analyse über die Modellierung und Spezifikation bis hin zur Realisierung und Evaluation dessen. In der Analyse werden die Benutzer und der Kontext untersucht. Die Planung, Konzeption und

das Layout folgen in den Kapiteln Projektplanung, Software Engineering und UI-Prototyping. Da Software Engineering grundlegend für die Software-Entwicklung ist und die Usability Hand in Hand mit der Software-Entwicklung geht, ist es wichtig sich eingehend mit dem Entwicklungs-Modell/Prozesses zu befassen.
Der Umsetzung und Evaluation sind Extra-Kapitel gewidmet.

Ziel dieser Arbeit ist es ein System mit Hilfe der Usability zu gestalten, dass es weitesgehend erlaubt sich den Anforderungen einer sich wandelnden Rennwagen-Telemetrie anzupassen und den Zielen der Formula Student, in Hinsicht auf die Bedienbarkeit im Rennsport, zu entsprechen.

1.1 Motivation

Informations-Technologie ist heutzutage in den verschiedensten Bereichen des alltäglichen Lebens wie auch im Automobilsport allgegenwärtig. Hier werden durch den Wettbewerb bedingt immer höhere Anforderungen an die Technik gestellt. Die IT soll durch Datenerfassung und Automatismen den Rennwagen schneller und sicherer gestalten, zudem Auswertungen bereit stellen und den Fahrer in vielen Tätigkeiten unterstützen oder sogar entlasten. Eines der wichtigsten Werkzeuge des Fahrers zum Steuern eines Fahrzeuges ist sein Lenkrad. In seiner reinen Funktionalität dient es zum Lenken des Fahrzeuges. Sogenannte Multifunktions-Lenkräder mit Knöpfen zum Steuern von verschiedensten Geräten, von Bordcomputern bis hin zu CD-Playern, werden immer häufiger in Automobile eingebaut und bieten dem Fahrer während der Fahrt neue Steuermöglichkeiten, ohne die Hände vom Lenkrad zu nehmen.

Solche Schnittstellen zwischen Mensch und Maschine müssen benutzerfreundlich entwickelt werden und intuitiv in der Handhabung sein. Ganz besonders in einem Automobil, bei dem die Sicherheit des Fahrers von einem Bruchteil einer Sekunde der Unaufmerksamkeit abhängt, sollte die Handhabung sehr gut durchdacht sein. Mit dieser Thematik beschäftigt sich die Usability.

Usability gewinnt immer mehr an Akzeptanz und Wichtigkeit, denn „Usability wird zum neuen Differenzierungsfaktor für Unternehmen und ihre Produkte“(Richter und Flückiger, 2007, S. 8). „Bei der Entwicklung von ... immer komplexer werdenden Consumer-Geräten wie ... Auto-Bordcomputer arbeiten immer mehr Mitarbeiter bei Herstellern oder Anbietern solcher Produkte ... Diese Mitarbeiter helfen, die Ergonomie und damit die Kundenakzeptanz, Produktivität und Wettbewerbsfähigkeit zu verbessern“(Dahm, 2006, S. 37).
Die Formula Student bewertet den Rennwagen nach Leistung und Aussehen, sowie der technischen Realisierung.Sie vergibt auch Punkte für gut realisierte und durchdachte Lösungen. Mit der Analyse der Anforderungen, der Umsetzung und der Evaluierung mittels der

Usability sollen noch mehr Punkte bei der Formula Student für das HAWKS Racing Team gewonnnen werden.

1.2 Formula Student

Folgendes Zitat erdeutlicht worum es sich bei Formula Student (siehe VDI, 2008) handelt und welches deren Aufgabenstellungen sind:

> Die Formula Student ist ein Projekt für Studenten der Ingenieurswissenschaften in dem es gilt einen einsitzigen Rennwagen (Monoposto) zu entwerfen und zu bauen. Das Projekt stellt normalerweise einen Teil der akademischen Studien dar und gipfelt in einem Wettbewerb, bei dem Teams aus der ganzen Welt zusammenkommen um gegeneinander anzutreten. Bei der Konstruktion müssen Regeln bezüglich der Rahmengestaltung (Sicherheit) und des Motors eingehalten werden, um das Wissen, die Kreativität und das Vorstellungsvermögen der Studenten zu testen. Viertaktmotoren mit bis zu 610 cm^2 können turbo- oder kompressorgeladen werden, um eine neue Herausforderung in der Motorenentwicklung hinzuzufügen. Die Fahrzeuge werden in drei Kategorien bewertet:
> STATISCH - Design- & Costreport, Marketingpräsentation
> DYNAMISCH I - Acceleration, Skid Pad
> DYNAMISCH II - Hot Lap, Endurance
> Die Studenten sollen annehmen, dass ein Produktionsbetrieb ihnen den Auftrag erteilt hat, einen Rennwagen zu entwickeln und einen Prototypen zur Beurteilung zu bauen. Der angestrebte Markt ist der der nichtprofessionellen Wochenendrennfahrer, Berg- und Sprintrennfahrer. Der Wagen muss daher hohe Beschleunigungs- und Bremsleistung sowie gute Handlingeigenschaften besitzen. Er muss günstig in der Anschaffung, bezahlbar im Unterhalt und vor allem zuverlässig sein. Des Weiteren wird der Marktwert des Fahrzeugs durch ein ansprechendes Design, Komfort und die Verwendung von Bauteilen aus der Massenproduktion gesteigert. Der Hersteller plant 1.000 Fahrzeuge pro Jahr zu fertigen und sie zu einem Stückpreis unter 25.000 Dollar an den Kunden zu bringen. Die Herausforderung für ein Team besteht in der bestmöglichen Erfüllung all dieser Anforderungen. Innerhalb eines Jahres entwickeln die Teams Prototypen, die dann auf den verschiedenen Wettbewerben (in den USA, Brasilien, Japan, England, Italien, Australien, Deutschland) miteinander verglichen werden. Weltweit gibt es inzwischen über 300 Teams, in Deutschland mit Beginn des Jahres 2006 über 30. (vgl. Schwarz, 2006)

1.3 HAWKS Racing Team

Das HAWKS Racing Team ist ein interdisziplinär arbeitendes Team von Studenten der verschiedensten Fachrichtungen der Hochschule für Angewandte Wissenschaften Hamburg, mit dem gemeinsamen Ziel einen Rennwagen für Formula Student zu bauen. Es wurde 2003 ins Leben gerufen und hat seitdem schon drei Rennwagen konstruiert und diese erfolgreich auf Veranstaltungen präsentiert. Die Studenten des Teams sind in verschiedene Baugruppen, entsprechend ihrer Fachrichtung, aufgeteilt. Dabei ergänzen sich die Baugruppen durch ständige Kommunikation und interdisziplinäre Arbeit am Rennwagen.

1.4 Zielsetzung

Für das HAWKS Racing Team soll ein Lenkrad mit eingebautem Display und Eingabemöglichkeit entwickelt werden. Um sich von der Konkurrenz abzuheben, soll das Lenkrad eine vollständig eigene Entwicklung des HAWKS Racing Team sein. Um den Anforderungen und den Wunsch des HAWKS Racing Team nach Punkten bei der Formula Student nachkommen zu können, soll kein kommerzielles Gerät benutzt werden.

Kernpunkt der Entwicklung und dieser Arbeit ist die Verwendung der Methoden der Usability. Es soll ein Multifunktions-Lenkrad mit Display unter verschiedensten Usability-Aspekten konstruiert werden.

Das alte System wird analysiert, eine Anforderungs-Analyse durchgeführt und anhand der Anforderungen und der Gestaltgesetze der Usability, von eigenen Anforderungen über Normen bis zu den gesetzlichen Vorschriften, ein neues System implementiert. Das neue System soll die im Rennwagen vorhandene Telemetrie nutzen und die Daten der Sensoren darstellen. Das Display und die dazugehörige Software soll einfach und benutzerfreundlich zu bedienen und die Anzeige auf dem Display für den Fahrer leicht lesbar sein. Dabei soll die Software aber auch modular erweiterbar sein, um neue Anforderungen leicht und schnell nachimplementieren zu können.

Paralell zu dieser Arbeit gibt es die Arbeit von Herrn Johann-Nikolaus Andreae (Andreae, 2008), die sich mit der Hardware und dem laufenden Betriebssystem für die Software beschäftigt. Es wird ein spezielles der Größe des Lenkrades angepasstes Microcontroller-Board entwickelt, dass mit einem ARM-9 Microprozessor bestückt ist. Als Betriebssystem wird Linux Embedded verwendet. Die Arbeit ist wie folgt aufgeteilt:

Aufgabe	Person
Usability	Lorenz
CAN Socket	Lorenz
Display Menü	Lorenz
CAN-Konfiguration	Lorenz
Interface-Design	Lorenz
Hardware-Design und Entwicklung	Andreae
Linux Embedded	Andreae
CAN Treiber	Andreae

Tabelle 1.1: Aufgabenverteilung an der Entwicklung des Lenkrades

1.5 Abgrenzung

Diese Bachelor-Arbeit beschäftigt sich mit der Entwicklung der Display-Software und des Interfaces für die Bedienung der Software. Wartbarkeit, Flexibilität, Stabilität und Performance der Software werden in den Vordergrund gestellt.

Die Software soll Informationen zu den einzelnen Sensoren wiedergeben und in der Art der Darstellung durch Eingabe veränderbar sein. Die Software soll daher weder Änderungen an der Motor-Steuerung oder anderen elektronischen Geräten und dessen Software ausüben können. Es soll vom CAN-Bus nur gelesen werden.

Der Schwerpunkt der Arbeit ist die Usability am System im Entwicklungs-, Verifikations- und Validationsprozess.

In dieser Arbeit wird auf die physiologischen und psychologischen Aspekte der Usability anhand von Daten unterschiedlicher Literatur eingegangen. Das Verifizieren dieser Daten ist in diesem Umfang nicht möglich. Die vielen Facetten der Usability können daher nicht gleich tief behandelt werden.

Viele der in dieser Arbeit erarbeiteten Empfehlungen das System zu gestalten, werden leider nicht am aktuellen Rennwagen umgesetzt. Das betrifft größtenteils die Hardware am Rennwagen und dessen Position.[1]

[1] **Wichtige Information**: alle mit einem hochgestellten „G“ markierten Begriffe werden im Glossar erörtert!

2 Usability-Analyse

In diesem Kapitel werden verschiedenste Analyse-Methoden der Usability erklärend aufgegriffen und durchgeführt. In folgenden Kapiteln werden die Analyse-Methoden angewandt:

- IST-Wert-Analyse: hier wird Hardware und Software des vorherigen Rennwagen analysiert und dessen Eigenschaften genau aufgelistet
- Normen und Gesetze: es werden Normen und Gesetze aufgelistet, die bei der Entwicklung eines solchen Systems berücksichtigt werden müssen
- Nutzungskontext: Analyse der Umgebung, Benutzer, Aufgaben, und Mittel
- Ablaufbeschreibung: die Ablaufbeschreibungen der Aufgabenstellungen an das System werden hier betrachtet
- Interviews: Analyse anhand von Befragungen
- Personas und Szenarien: es werden fiktive Personen erstellt und Szenarien zu den Personen durchgespielt
- Anforderungsanalyse: in diesem Unterkapitel werden die Anforderungen an das System erschlossen und die Möglichkeiten elaboriert
- Ziele: hier werden die Anforderungen für das System festgelegt, anhand derer Evaluiert werden soll

Bevor es aber an die Analyse geht, wird an dieser Stelle noch mal der Begriff der Usability im Groben beschrieben.
Usability ist eine noch sehr junge Disziplin der Wissenschaft. Wie der Name in die deutsche Sprache übersetzt schon suggeriert, beschäftigt sich die Usability mit der Gebrauchstauglichkeit/Benutzbarkeit von Software wie auch Maschinen. Dabei geht es nicht nur, wie viele denken könnten, um das Layout (hier Darstellung) einer Software und das Verschönern derer. Usability vereinigt Wissen aus vielen Wissenschaften und Themengebieten. Die am häufigsten herangezogenen Themengebiete sind folgende:

- Sozialwissenschaften
- Psychologie

- Physiologie
- Arbeitswissenschaften
- Informatik
- Design/Grafik

Die meisten Usability-Ergonomen kommen aus diesen Wissenschaften. Sie müssen in der Lage sein fächerübergreifend mit anderen Fachrichtungen zusammen zu arbeiten, um die Anforderungen lokalisieren und verstehen zu können. Dabei spielt nicht nur der Auftraggeber eines Projektes eine große Rolle, sondern auch der spätere Benutzer. Den benötigten Anforderungen nach, ist die Tiefe des Wissens anderer Fachrichtungen von Produkt zu Produkt unterschiedlich und projektspezifisch.

Im Vordergrund eines Usability-Ergonoms steht das benutzerorientierte Vorgehen. Usability sollte als Werkzeug zur Entwicklung gesehen werden und nicht nur analytischen Zwecken dienen. Sie soll das Produkt nicht nur kritisieren und beurteilen sondern dazu beitragen, die Qualität in Hinblick auf die Gebrauchstauglichkeit zu verbessern und somit den Entwicklungprozess beeinflussen. Ein Entwicklungsprozess zu beeinflussen heißt auch manchmal zu erkennen, dass die Konzeption in die falsche Richtung läuft, vieles verworfen werden muss und man von vorn beginnt. Das fordert eine kontinuierliche Überprüfung des Entwicklungsprozesses, was Zeit und wiederrum Geld raubend für ein Projekt sein kann.

Die Definition von der Gebrauchstauglichkeit ist in der Norm ISO 9142-11 (DIN EN ISO 9241-11, 1998) festgehalten. Die Norm definiert Gebrauchstauglichkeit wie folgt:

> „Das Ausmaß, in dem ein Produkt durch bestimmte Benutzer in einem bestimmten Nutzungskontext genutzt werden kann, um bestimmte Ziele effektiv, effizient und zufriedenstellend zu erreichen.“(DIN EN ISO 9241-11, 1998, Def. 3.1)

Die Norm ISO 9142-11 wurde zwar für ergonomische Anforderungen für Bürotätigkeiten mit Bildschirmgeräten entworfen, kann aber auch in einem anderen Nutzungskontext angewandt werden. Die Norm beschreibt Gebrauchstauglichkeit und gibt Informationen über Methoden zur Spezifikation, Anwendung, sowie Evaluierung von Produkten auf ihre Gebrauchstauglichkeit. Begriffe die in diesem Kontext in der Literatur sehr oft auftauchen sind **Effektivität, Effizienz** und **Zufriedenstellung**. Diese drei Säulen der Usability sind Indikatoren für Gebrauchtauglichkeit. Auf sie wird in diesem und den folgenden Kapiteln noch genauer eingegangen.

Für die Gebrauchstauglichkeit spielen viele Kriterien eine Rolle. Eines der wichtigsten ist die Mensch-Maschine-Kommunikation. Sie trägt im wesentlichen dazu bei, wie gut der Benutzer die Maschine steuern und, wie oder ob sein von der Maschine gefordertes Ziel erreicht

werden kann.

2.1 IST-Wert-Analyse

Der Rennwagen HAWK07 des letzten Jahres vom HAWKS Racing Team ist mit einem eigen konstruierten Lenkrad und einem darauf aufgesetzten kommerziellen Anzeigegerät ausgestattet. Das Anzeigegerät heißt „MyChron3 GOLD“(siehe Abb. 2.1) und wird von der Firma AiM Sports vertrieben. Das MyChron3 besteht aus einem Display mit einer ungefähren Größe von 6,06“ in der Bildschirmdiagonale, besitzt 4 Knöpfe für die Eingabe, 14 LED’s und ein zusatzliches kleines Ganganzeigen-Display. Die wichtigsten Funktionen einmal folgend aufgelistet:

Abbildung 2.1: Mychron3 Gold (hier im Bild rot umrandet)

- Umstellung auf die Lichtverhältnisse (Night vision)
- Ganganzeige in einem extra Display
- 10 LED’s zum anzeigen des Zeitpunktes zum Schalten der Gänge

- Drehzahl-Anzeige analog in Balkenform oder digital numerisch
- Temperatur-Anzeige numerisch wahlweise in ° Celsius oder Fahrenheit
- verschiedenste Winkel-Anzeigen wie Lenkrad-Winkel, Roll-Winkel oder Brems-Pedal-Winkel
- verschiedenste Druck-Anzeigen wie Öl-Druck oder Treibstoff-Leitungs-Druck in bar
- Geschwingikeits-Anzeige numerisch wahlweise in mph oder km/h
- Rundenzeit
- Batterie-Anzeige
- 4 LED's zum Anzeigen von Fehlern
- Wasserresistent
- Display über zusätzliche Software konfigurierbar
- Daten-Speichereinheit
- vier analoge Eingänge
- ein USB-Interface

Das binäre LC Display ist groß und im Layout sehr aufgeräumt. Die angezeigten Daten sind gut lesbar und in ihrer Art der Darstellung (Celsius oder Fahrenheit, mph oder km/h) konfigurierbar. Leider sind nicht alle angezeigten Daten mit einem Label versehen, sodass man nur Vermutungen darüber anstellen kann, was die Daten anzeigen, wenn man das Display zum ersten Mal sieht. Für die Erfassung der Daten, wird das Gerät direkt an die Motorsteuerung angeschlossen. Ein USB-Interace ermöglicht den Anschluss an einen PC zur Konfuguration oder dem Auslesen von Daten
Die Gang-Anzeige ist mit seiner extra LED-Siebensegmentanzeige von allen anderen Daten abgesondert dargestellt und ihr wird damit eine besondere Wichtigkeit zugesprochen. Die 10 LED's zur Anzeige des besten Schaltzeitpunktes sind über dem Display angebracht. Auch Sie sind per Software konfigurierbar. Die LED's liegen in den Farben Rot, Orange und Grün vor. Die grünen und die orangenen LED's von den 10 LED's über dem Display sollen den Fahrer informieren und darauf vorbereiten, dass der beste Schaltzeitpunkt inmuttelbar bevor steht. Die roten LED's der 10 LED's über dem Display geben den Schaltzeitpunkt an. Der Verlauf der LED's für den Schaltzeitpunkt geht von außen nach innen.
Die restlichen roten LED's am seitlichen Rand des Displays sind für Fehlermeldungen gedacht. Sie sollen warnen, wenn eine Anzeige, wie die Temperaturanzeige, einen voreingestellten Wert überschreitet.
Die Batterie-Anzeige warnt den Benutzer, dass die Batterien nur noch wenig Volt haben. Diese Anzeige geht also nur an, wenn die internen Batterien schwach, und gibt keinen Status

der Batterien. Außerdem gibt das Gerät nur den Status der internen Batterien an und nicht, wie man denken könnte, den Status der Autobatterie.
Als besonders herauszuheben ist die Daten-Speichereinheit zum speichern der Daten für spätere Auswertungen, die Anzeige der Rundenzeit, die mithilfe eines auf der Strecke montierten Infrarot-Receivers errechnet wird, und die Eigenschaft, dass das Gerät wasserresistent ist.
Das MyChron3 lässt sich in in seiner Anzeige und dessen Anordnung (Layout) zwar nicht konfigurieren, kann aber den Wert der Anzeige, die von den analogen Eingängen kommen, in der Einheit sowie der Darstellung bedingt anpassen.

2.2 Normen und Gesetze

Normen sind in technischen Bereichen ein wichtiger Bestandteil, damit man Produkte besser Beschreiben, Analysieren, Produzieren oder Klassifizieren kann. Normen helfen den Ingenieur im Entwicklungsprozess nicht alles neu erschließen oder beweisen zu müssen. Man erhält dadurch einen gemeinsamen Wortschatz und hat vordefinierte Schnittstellen. Ein USB-Stecker ist dann ein USB-Stecker und wird von allen verstanden. So gibt es auch Normen, die Usability-Ergonomen wichtige Informationen und damit Werkzeuge in die Hand geben, damit man Produkte hinsichtlich ihrer Usability messen oder entwickeln kann.

Für die Analyse und Umsetzung des Multifunktions-Lenkrads, werden in dieser Arbeit folgende Normen herangezogen:

- DIN EN ISO 9142 - Ergonomische Anforderungen für Bürotätigkeiten mit Bildschirmgeräten
  - Teil 11 - Anforderungen an die Gebrauchstauglichkeit
  - Teil 12 - Informationsdarstellung
  - Teil 110 - Grundsätze der Dialoggestaltung
  - Teil 3 - Anforderungen an visuelle Anzeigen
  - Teil 14 - Dialogführung mittels Menüs
- DIN EN ISO 13407 - Benutzer-orientierte Gestaltung interaktiver Systeme
- DIN EN ISO 15008 - Ergonomische Aspekte von Fahrerinformations- und Assistenzsystemen
- ISO 2575 - Strassenfahrzeuge - Symbole fuer Bedienteile, Anzeige- und Warngeraete

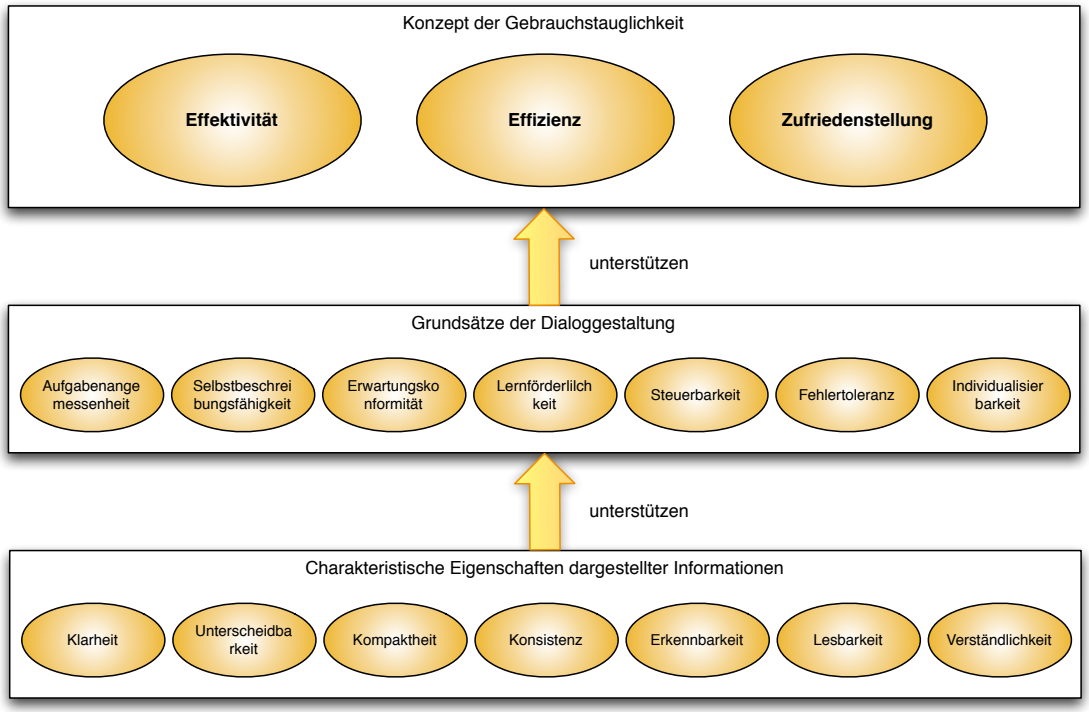

Abbildung 2.2: Beziehung zwischen Konzept, Grundsätzen und charakteristischen Eigenschaften der Teile der DIN-Norm 9241 (vgl Seite 22 DIN EN ISO 9241-110, 2006)

2.3 Interviews

Die aufgabenangemessene Gestaltung von Systemen ist ohne das Wissen der Benutzer nicht oder nur bedingt möglich. Daher gilt es diese Benutzer und die Personen die mit diesem System in irgendeiner Art zu tun haben zu finden und sie in einem sogenannten Interview zu befragen. Eine Art ein Interview zu halten ist die Methode „Contextual Inquiry“. Bei Contextual Inquiry beobachtet man einen möglichen späteren Benutzer bei seiner Erledigung der Aufgaben und stellt der Person ausgewählte Fragen. Leider ist diese Art der Befragung in dem Nutzungskontext nicht möglich. Die Stakeholder^{G} des Systems sind:

- Fahrer
- Ingenieure
- Sponsoren
- Zulieferer

- Juroren der Formula Student

In einem klassischen Interview mit den Fahrern des Rennwagens wurden Fragen zu der Darstellung und den Informationen auf dem Display des vorrigen Rennwagens HAWK03 gestellt. Die Auswertung der Antworten ergab,

- das MyChron3 war schwer zu Bedienen
- es gab leichte Einarbeitungsschwierigkeiten in das Gerät
- schwacher Kontrast, daher nur ausreichende Lesbarkeit
- Gerät fiel einmal aus, musste mal neu gestartet werden
- Drehzahl war schlecht ablesbar
- Ganganzeige und Runden-Anzeige wurde nicht genutzt
- versehentliches umstellen der Anzeige während der Fahrt
- aus Menüs kam man schwer wieder raus
- zu Viele Informationen auf einmal; wirkte überladen

Die Auswertung gibt erste Anhaltpunkte für die Anforderungs-Analyse des neuen Systems und zeigt, worauf die Fahrer achten und was für Probleme mit dem alten System aufgetaucht sind. Da es aber nur zwei Personen, nämlich die Fahrer, waren, die in diesem Interview befragt wurden, wird an dieser Stelle darauf hingewiesen, dass es Meinungen sind, die hier vertreten werden, und nicht wissenschaftlich erschlossene Untersuchungen.

Um mehr Informationen über die benötigten Daten für die Anzeige zu erhalten und die Team-Mitglieder auf das Thema Usability zu sensibilisieren, wurde ein Vortrag zur Usability vor dem gesammten Team mit anschließendem Fragebogen gehalten.

2.4 Nutzungskontext

Der Nutzungskontext umfasst "Die Benutzer, Arbeitsaufgaben, Arbeitsmittel (Hardware, Software und Materialien) sowie die physische und soziale Umgebung, in der das Produkt genutzt wird"(Def. 3.5 DIN EN ISO 9241-11, 1998).
Die Benutzer des Multifunktions-Lenkrads sind die späteren Fahrer des Fahrzeuges. Die Fahrer lassen sich in zwei verschiedene Gruppen unterteilen, die *Profi-Fahrer* und die Zielgruppe von Formula Student, nämlich den *ambitionierten Rennfahrer*. Dazu später in „Personas und Szenarien“mehr.

Die übergeordnete Arbeitsaufgabe ist das Fahren des Fahrzeuges. Diese Arbeitsaufgabe lässt sich in kleinere Aufgaben zerlegen, die sich in der Art und Weise in den verschiedenen Wettkampfdisziplinen der Formula Student wiederfinden lassen. Die Disziplinen sind „Acceleration“, „Endurance“, „Skidpad “, „Autocross“ und „Fuel-Consumption/Fuel-Economy“. Die folgenden Tabellen zeigen die erhobenen Daten zu den einzelnen Disziplinen.

Aufgabenbezeichnung	Endurance
Aufgabenhäufigkeit	oft (stellt das normale Rennen auf Zeit dar)
Aufgabendauer	22 km Fahrt bzw. 2 Fahrten jeweils 11 km
Kurzbeschreibung	Beim Endurance soll das Fahrzeug lange auf der Strecke bleiben und möglichst schnell zum Ziel kommen
Vorbedingungen/ Aufgabenabhängigkeiten	keine Fuel-Consumption
Sicherheitskritische Erfordernisse	Warning bei Überhitzung des Motors
Physische und mentale Anforderungen	gutes Durchhaltevermögen, Konzentration auf den Rennwagen sowie die Strecke

Tabelle 2.1: Arbeitsaufgabe : „Endurance“

Aufgabenbezeichnung	Fuel-Consumption
Aufgabenhäufigkeit	nicht so oft
Aufgabendauer	22 km Fahrt bzw. 2 Fahrten jeweils 11 km
Kurzbeschreibung	Beim Fuel-Consumption soll das Fahrzeug lange auf der Strecke bleiben und möglichst wenig dabei Verbrauchen
Vorbedingungen/ Aufgabenabhängigkeiten	keine Endurance
Sicherheitskritische Erfordernisse	gleiche wie beim Endurance
Physische und mentale Anforderungen	gutes Durchhaltevermögen, Konzentration auf den Rennwagen sowie die Strecke

Tabelle 2.2: Arbeitsaufgabe : „Fuel-Consumption“

Bei den Disziplinen Endurance und Fuel-Consumption handelt es um das gleiche Rennen nur mit anderen Zielen. In der Disziplin Endurance soll der Rennwagen 22 km lang mit einem Zwischenstopp von maximal drei Minuten fahren. In diesem Zwischenstopp soll der Fahrer gewechselt und der Motor ausgemacht und ohne Starthilfe wieder gezündet werden.
In der Disziplin Fuel-Consumption wird der Verbrauch des Wagens gemessen.

Aufgabenbezeichnung	Acceleration
Aufgabenhäufigkeit	oft
Aufgabendauer	sehr kurze Dauer
Kurzbeschreibung	Beim Acceleration geht es um eine schnelle Beschleunigung des Fahrzeuges
Vorbedingungen/ Aufgabenabhängigkeiten	der Motor sollte eine gute Betriebstemperatur haben keine
Sicherheitskritische Erfordernisse	keine
Physische und mentale Anforderungen	schnelle Reaktionsfähigkeit, starke Konzentration beim Schalten

Tabelle 2.3: Arbeitsaufgabe : „Acceleration“

Acceleration testet und beurteilt die Beschleunigung des Fahrzeuges. Der Rennwagen soll auf 75 Metern eine möglichst schnelle Geschwindigkeit erreichen.

Aufgabenbezeichnung	Skid-Pad
Aufgabenhäufigkeit	selten
Aufgabendauer	4 Runden (kurz)
Kurzbeschreibung	Beim Skidpad wird das auf einem Rundkurs die Zeit gemessen
Vorbedingungen/ Aufgabenabhängigkeiten	keine keine
Sicherheitskritische Erfordernisse	Warnungen bei Motorüberhitzung oder Spannungsabfall
Physische und mentale Anforderungen	hohe Konzentration auf die Strecke durch das viele Lenken

Tabelle 2.4: Arbeitsaufgabe : „Skid-Pad“

Der Skid-Pad ist ein Rundkurs auf einer Strecke die wie eine Acht aussieht. Jedes Team darf zwei Fahrer stellen die jeweils zwei Versuche haben. Die Aufgabe ist die beste Rundenzeit zu erlangen.

Aufgabenbezeichnung	Autocross
Aufgabenhäufigkeit	selten
Aufgabendauer	kurz (2-3 Minuten)

Kurzbeschreibung	Rundkurs über 800 Meter mit vielen Kurven. So schnell wie möglich durch den Rundkurs
Vorbedingungen/ Aufgabenabhängigkeiten	keine keine
Sicherheitskritische Erfordernisse	Warnungen bei Motorüberhitzung oder Spannungsabfall
Physische und mentale Anforderungen	hohe Konzentration auf die Strecke durch das viele Lenken

Tabelle 2.5: Arbeitsaufgabe : „Skid-Pad“

Beim Skid-Pad wie auch im Autocross muss der Rennwagen beweisen wie gut er in den Kurven liegt. Ziel beim Autocross ist, wie beim Skid-Pad, die beste Rundenzeit zu erlangen.

Die physische Umgebung (Cockpit) im Monoposto beinhaltet eine Sitzschale, ein Lenkrad und Seitenwände mit Armlehnen, die dem Fahrer wenig Raum für Bewegung lassen. Der Fahrer selbst trägt zudem einen feuerfesten Schutzanzug, Handschuhe und einen Helm mit verdunkelten Visir. Die Lautstärke des Fahrzeuges ist sehr hoch und hemmt den Fahrer andere akustische Signale wahrzunehmen. Die Sitzposition des Fahrers ist tief und gibt den Fahrer so eine gute Sicht auf das Lenkrad während der Fahrt. Er kann direkt auf das Display gucken, wenn das Display im Lenkrad verbaut ist.

2.5 Personas und Szenarien

2.5.1 Personas

Der Nutzungskontext des Systems lässt zwei Personas erschließen. Der erfahrene Fahrer, der an der Formula Student teilnimmt, und der Hobby-Fahrer.

Der erfahrene Fahrer (Profi-Fahrer) kennt den Rennwagen und macht Test- und Trainingsfahrten. Er nimmt sich die Zeit sich mit dem System auseinander zu setzen.
Seine Ziele sind Rennen zu gewinnen und das Optimum aus den Rennwagen raus zu holen. Er bringt sehr viel Erfahrung aus dem Rennsport mit.

Der zweite Fahrer, nämlich der Gelegenheits- oder Hobbyfahrer, hat ganz andere Zielvorstellungen. Zwar möchte er auch einen funktionstüchtigen schnellen Rennwagen haben, aber

mit dem Ziel Spass beim Fahren zu haben. Er will sich nicht zu lange mit den Internas des Rennwagens beschäftigen, sondern ihn hauptsächlich Fahren. Er will eine ästhetisch aussehende Menüsteuerung haben.

2.5.2 Szenarien

Für den Profi-Fahrer erschließt sich folgendes Szenario:
Der Fahrer steht bei einem Wettbewerb wie die Formula Student kurz vor dem Start eines Rundkurses. Die Instrumente bzw. die Anzeige mit den Daten wird geprüft, um nochmal sicher zu stellen, dass der Rennwagen ordnungsgemäß läuft.
Nachdem er gestartet ist, konzentriert er sich soweit er kann auf das Fahren. Die Daten sollten für ihn funktional und aufgabengemäß angezeigt werden und nicht ablenken.

Ganz anders sieht ein Szenario für den Hobbyfahrer aus:
Der Fahrer spielt mit der Software, weil er ausprobieren möchte was man damit alles machen kann. Er Fährt mit dem Rennwagen zum Spass und hat keinen Druck, etwas zu gewinnen. Gelassen geht er auch vom Gas um sich mal der Anzeige zu widmen und die Darstellung der Anzeige vielleicht zu ändern, oder nachzuschauen wie der Zustand des Wagens ist.

2.6 Anforderungsanalyse

In diesem Kapitel werden die Anforderungen an das System[1] betrachtet. Die Anforderungen sind wichtig, um einerseits ein gutes Software- wie auch Hardware-Design zu erstellen, als auch später mithilfe der Anforderungen Evaluieren zu können. In der Evaluierungsphase - den Testfahrten und im Usability-Labor - nimmt man sich die Anforderungen und überprüft Punkt für Punkt ob auch alle Anforderungen richtig umgesetzt wurden.
Zunächst werden Problemstellungen aus den Nutzungskontext des vorangegangenen Kapitels (2.4) erörtert. Es wird geprüft, inwiefern es möglich und welcher Aufwand damit verbunden ist, die Aufgabenstellungen zu lösen. Sodann wird eine Auflistung der benötigten Hard- bzw. Software und den technischen Möglichkeiten mittels einer Marktanalyse gemacht.

Die Anforderungen können aus den verschiedensten Bereichen kommen. Daher ist es sinnvoll vorher den Nutzungskontext zu untersuchen um die Problemstellungen und somit auch die Anforderungen, die an ein System gestellt werden, zu erkennen.
Anforderungen können von Aufgaben, von der Umwelt, den Mitteln, von den Personen

[1] damit ist das Lenkraddisplay gemeint

(physiologisch, psychisch, sozial) und vom System selbst abhängig sein. Anforderungen wie *soziales Umfeld* oder einige physiologische Aspekte (z.B. Größe und Gewicht einer Person) werden nicht betrachtet und als gegeben angesehen. Grund dafür ist das Einsatzgebiet des Rennwagens. Bei einen kleinen Rennwagen wie diesen kann nicht auf alle Menschen Rücksicht genommen werden.
Das Cockpit des Rennwagen des HAWKS Racing Team's wurde beispielsweise zu Ungunsten der Formula Student und zu Gunsten der Fahrer größer gebaut, damit mehr Fahrer in das Cockpit passen. Der Grund dafür ist, dass das HAWKS Racing Team ihre Philosophie verfolgt, mehr Personen die Möglichkeit zu bieten diesen Rennwagen fahren zu können. Das hat aber einen Verlust der Punkte in der Formula Student zur Folge.

Die Umwelt spielt immer eine Rolle, im Besonderen wenn das Umfeld des Nutzungskontexts sich außerhalb geschlossener Räume befindet. Die Umwelt ist im Rennsport eine Gewalt, gegen die immer gearbeitet wird. Zum einen sind das die Winde und Luftwiederstände, zum anderen die Wetterverhältnisse, wie Regen und Sonne. Um das System vor einem Wasserschaden in der Elektronik zu schützen, muss es *Wasserdicht* sein.
Bei prallem Sonnenschein ensteht auf einer Fahrbahn und im Rennwagen hohe Temperaturen die mit eingeplant werden müssen. Die Elektronik sollte daher *gut gekühlt* sein oder *hohe Temperaturen aushalten*.

Praller Sonnenschein hat aber auch noch einen anderen großen Nachteil der berücksichtigt werden muss, die Helligkeit. Displays, ganz besonders Displays die bei direktem Sonnenlicht abgelesen werden sollen, müssen entsprechend hell sein. „Bei Sonnenlicht ablesbare Displays sind für gewöhnlich durch eine Helligkeit von 1000 cd/m^2 oder höher definiert“(Steinbacher, 2006). Zu den Helligkeiten der Displays und den Techniken später im Kapitel(2.6.4) mehr.

Bei einem Rennwagen entstehen während der Fahrt Vibrationen und elektromagnetische Felder, die es zu absorbieren gilt. Die Vibrationen und Erschütterungen (z.B. durch kleine Unebenheiten wie Schlaglöcher auf der Rennstrecke) sollten bei einer empfindlichen Elektronik berücksichtigt werden.
Gegen elektromagnetische Felder is es schon schwerer anzugehen. Sie sind einerseits schwer zu messen, weil Sie meist während der Fahrt auftauchen, und andererseits nicht leicht abzuschirmen sind. Für die Abschirmung müssen die Frequenzen der Felder gemessen werden. Das Material für die Abschirmung sollte je nach der Frequenz gewählt werden. Im Fall des Rennwagens sollten empfindliche Elektronik wie Mircroprozessoren oder Speichermodule am besten weit weg von Generatoren wie die Lichtmaschine positioniert werden. Bei der Stromerzeugung können nämlich starke elektromagnetische Felder entstehen, die wiederum sogenannte Bitkipper^G (Bitfehler) in der Hardware verursachen

können.
Mögliche Lösung solcher Probleme sind Abschirmungen mit elektrisch leitenden Materialien, wie Metalle, die die elektromagnetischen Felder abschirmen (vgl. EMF-Portal, 2008).

2.6.1 Sensorik

Die Aufgabenstellung dieser Arbeit ist es, eine Anzeige zu gestalten, die die Informationen der Sensoren anzeigt, sodass der Fahrer zu jeder Zeit der Fahrt gut Informiert über den Status seines Wagens ist.
Die Wahrnehmung des Fahrer spielt dabei eine große Rolle. In dieser Arbeit wird sich auf die wesentlichen sensorischen Fähigkeiten in der Kommunikation zwischen Mensch und Computer beschränkt, dem Hör- und Sehvermögen (vgl. Seite 41 Dahm, 2006).

Die drei Stufen der Informationsverarbeitung

Die Informationsverarbeitung der sensorischen Wahrnehmung ist ein Teil der Psychologie. Der Ablauf läuft im groben wie folgt ab (siehe Abb. 2.3).

1. Ein Sinnesorgan nimmt einen Reiz auf. Die Reize werden Vorverarbeitet und gegebenenfalls gefiltert, damit das Gehirn nicht überreizt wird. Die Sinnesorgane des Menschen besitzen ein sensorisches Kurzzeitgedächtnis. Es hilft Reize verarbeiten zu können auch wenn der Reiz nur sehr kurz und schnell kam.
   Beispielsweise wenn bei der Fahrt im Auto auf das Tachometer geguckt wird, braucht man nur einen kurzen Moment auf das Tachometer gucken, weil der visuelle Reiz - das Tachometer - noch im sensorischen Kurzzeitgedächtnis geblieben ist. So kann man seine Aufmerksamkeit schnell wieder der Straße widmen.

2. Die zweite Stufe umfasst das Wahrnehmen und Erkennen von Reizen. In Perzeption^{G}s-Prozessen wird der wahrgenommene Reiz erkannt und interpretiert (vgl. Seite 40 Dahm, 2006). In der kognitiven Verarbeitung wird dann mittels der Erfahrung und der Fähigkeit des Gehirns Abstrahieren zu können Dinge, Vorgänge und Beziehungen erkannt. Es gibt also eine fließende Wechselwirckung zwischen der Kognition^G der Perzeptionsprozesse. Für die kognitive Verarbeitung werden die Informationen aus dem Kurzzeitgedächtnis wie auch aus dem Langzeitgedächtnis gebraucht.

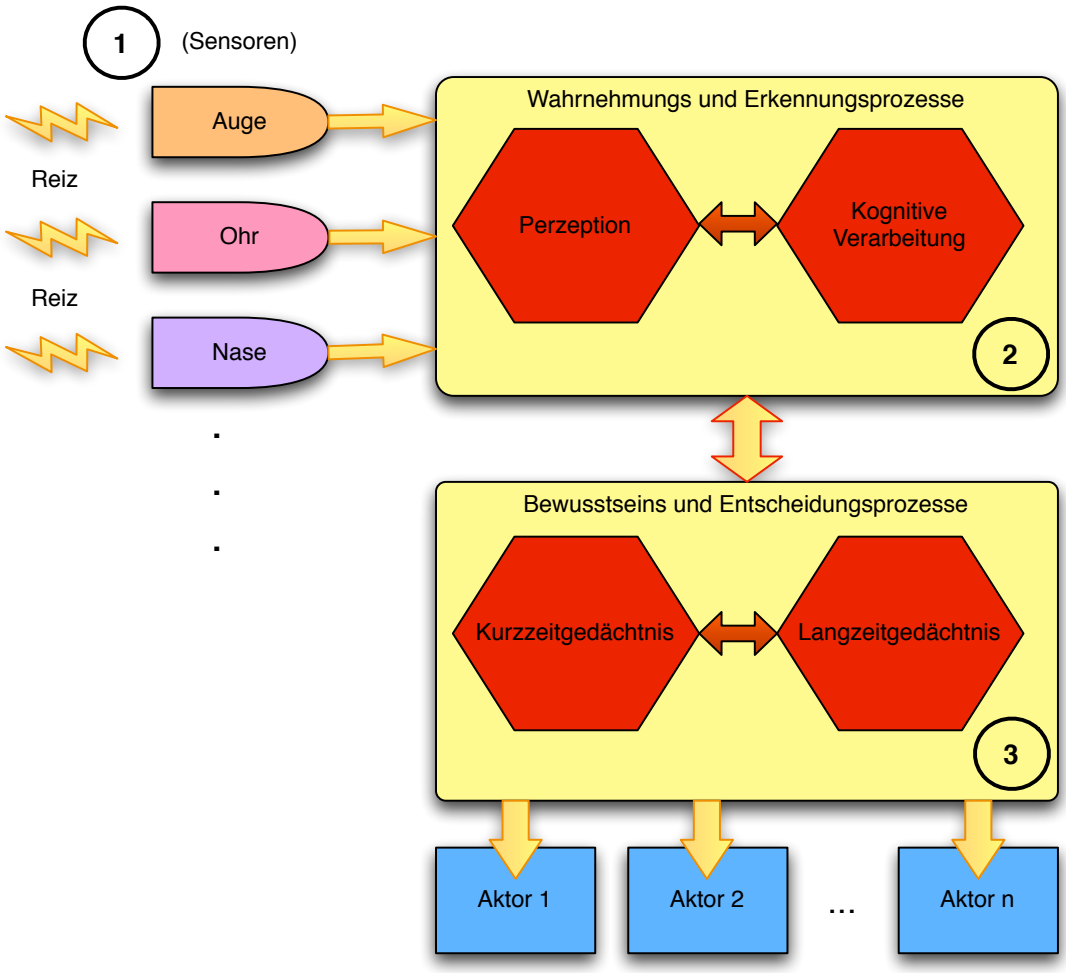

Abbildung 2.3: Stufen der sensorischen Wahrnehmung

3. Die letzte Stufe umfasst die Entscheidungs-Prozesse und das Befusste Lenken von Aktoren. In dieser Stufe wird der Kontext des Reizes analysiert. Es wird bewusst entschieden ob und wie auf den Reiz reagiert wird. Bei einer Reaktion wird ein Aktor - Füße, Hände, Mund u.s.w - gelenkt.

Für den Rennsport spielt die Reaktionszeit[G] des Fahrers eine wichtige Rolle. Man kann die Reaktionszeit anhand der drei Stufen der Informationsverabeitung erklären (siehe Abb. 2.3). Es spaltet sich auf in

- Wahrnehmungszeit
- Erkennungszeit

- Entscheidungszeit

Bei der Reaktionszeit spielen Faktoren wie Erfahrung, Menge der Reize, Vielfalt der Reize, mentaler und physischer Zustand eine Rolle. Bei einer Wahlreaktion[G] steigt die Reaktionszeit logarithmisch zu der Anzahl der Wahlmöglichkeiten (vgl. Dr. Sportwiss. Ralf Pfeifer, 2008). An der Abbildung 2.4 sieht man eine Kurve der Reaktionszeiten in Zusammenhang mit der Anzahl von Wahlmöglichkeiten. Bei wiederholter Durchführung, wie es der Fall eines Fahrers im Auto ist, verkürzt sich die Zeit wie es die untere Kurve in der Abbildung zeigt. Mehr zum Thema Reaktionszeit später im Kapitel 7 „Evaluation“, wo die Reaktionszeit einiger Testpersonen im Usability-Labor anhand der Software getestet wird.

Kruzzeitgedächtnis

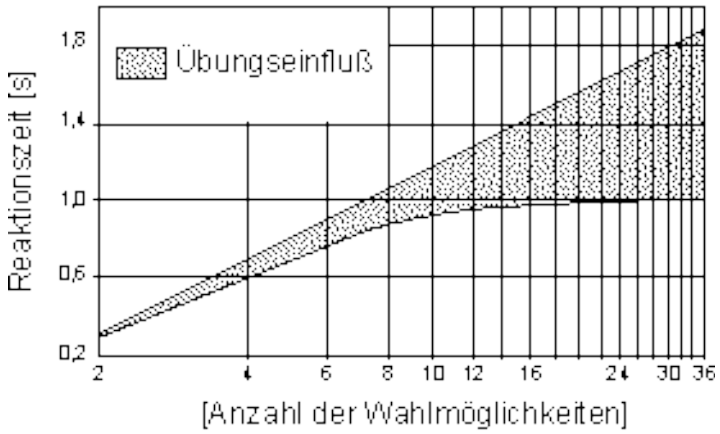

Abbildung 2.4: Reaktionszeiten in Zusammenhang der Anzahl an Wahlmöglichkeiten

Hören

Wie aus dem Kapitel 2.4 hervorgeht, ist die Umgebung während des Betriebs des Motors sehr laut, sodass die Wahrnehmung „Hören" des Fahrers stark eingeschränkt ist. Die Aufnahme von akustischen Signalen ist lediglich direkt am Ohr, unter dem Helm, möglich. Der Einsatz akustischer Signale würde den Fahrer bei der Fahrt entlasten immer wieder auf die Anzeige zu schauen und er könnte sich mehr auf die Straße konzentrieren. Über akustische Signale könnte man dem Fahrer Informationen wie den besten Schaltzeitpunkt oder Warnsignale übermitteln, die den Fahrer melden könnten auf die Anzeige zu schauen weil ein kritisches Ereignis eingetroffen ist.
Obwohl die Idee dem Fahrer akustische Signale als Informationträger für Informationen,

die eine Reaktion verlangen, zu übermitteln eine echte Entlastung für den Fahrer wäre, wird bei der Umsetzung darauf nicht zurück gegriffen. Es gibt zwei wesentliche Gründe, warum diese Möglichkeit nicht in Betracht gezogen wird. Zum einen ist es der zukünftige Wunsch nach einem Kommunikationssystem zwischen dem Fahrer und der Leitstelle an der Rennstrecke der es dem Fahrer sehr schwer machen würde die akustischen Signale war zu nehmen und gleichzeitig mit der Leitstelle zu kommunizieren. Zum anderen - und das ist der wesentlichere Grund - ist es der Aufwand eine entsprechende Hard und Software in der kurzen Entwicklungszeit zu implementieren.
Nichtsdestotrotz sollte für die Weiterentwicklung der Mensch-Computer Kommunikation diese Idee ins Auge gefasst werden.

Sehen

Um die Möglichkeiten des visuellen Wahrnehmens zu erötern, soll an dieser Stelle ein kleiner Exkurs über die Physiologie des Auges durchgeführt werden.
Das Auge ist das Organ mit dem der Mensch Visuelles wahrnehmen kann. Es nimmt verschiedenste Wellenlängen elektromagnetischer Strahlung auf und gibt es an das . Die Wellenlängen die der Mensch wahrnehmen kann liegen bei 400 bis 700nm (siehe Gegenfurtner, 2008). Über die verschiedenen Wellenlängen interpretiert der das Gehirn des Menschen die verschiedenen Farben. Für das scharf Sehen braucht das Auge einige Milisekunden um etwas zu fixieren. Wie lange hängt davon ab, wie weit das Objekt entfernt ist und wo das Auge vorher fixiert hat.

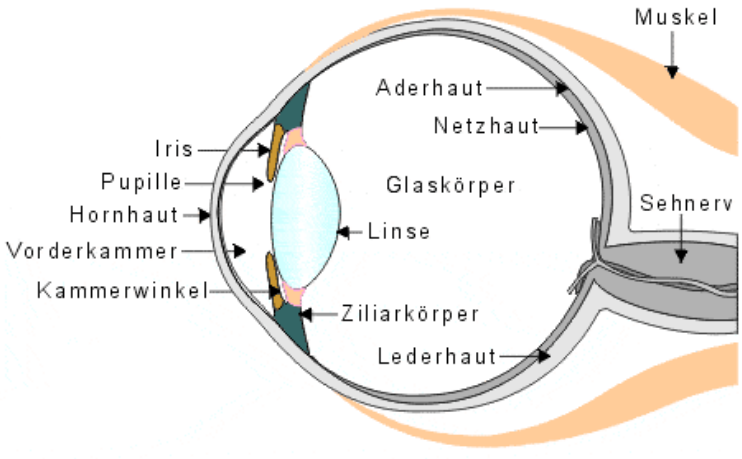

Abbildung 2.5: Das Auge (siehe Rauchmann, 2008)

Den Vorgang ein fixiertes Objekt im Auge scharf zu stellen nennt sich Akkommodation[G] Bei der Akkomodation wird die Linse kontrahiert oder entspannt und die Brechkraft derer somit verändert. Mit dem zunehmenden Alter werden die Ziliarmuskeln des Ziliarkörpers schwächer, sodass die Akkomodationsbreite[G] kleiner wird und man nahe Objekte immer schlechter scharf sehen kann. Der Nahpunkt (siehe Glossar „Akkommodationsbreite“) bei einem Judendlichen liegt ungefähr bei 7cm und bei einem 20jährigen bei etwa 10cm Entfernung zum Auge (siehe Wissenschaft-Online, 2008).
Die Iris umschließt die Pupille (siehe Abb. 2.5). Sie verengt die Pupille bei starker Helligkeit, Müdigkeit oder Nahsicht und erweitert die Pupille bei Dämmerung, Fernsicht und Stressreaktion. Diesen Vorgang nennt man Adaption[G]

2.6.2 Aufgabenanalyse

Anhand der Aufgaben in Kapitel 2.4 sind einige Anforderungen abzuleiten.
Der Fahrer muss beim Aussteigen, in der er nur 3 Minuten Zeit hat, frei in seiner Bewegung sein. Eine Konstruktion am Lenkrad erlaubt es das Lenkrad schnell von der Lenkstange zu nehmen und wieder aufzusetzen. Falls Elektronik in dem Lenkrad verbaut wird, ist darauf zu achten, das Lenkrad einerseits, beim Herausnehmen und wieder Aufsetzen vorsichtig zu behandeln und andererseits eine Elektronik zu wählen, die nicht so empfindlich auf Stöße reagiert.

Aus den Interviews geht hervor, dass die Fahrer gar nicht so viele Informationen während der Fahrt brauchen, weil sie sich stark auf die Straße und das Fahren konzentrieren müssen. Der Fahrer sollte nicht mit zu vielen Informationen belastet werden. Außerdem sollten die visuellen Informationen so positioniert sein, damit der Fahrer die Informationen schnell und klar ablesen kann, um sich schnell wieder auf das Fahren konzentrieren zu können.

Für die Postionierung einer Anzeige gibt es mehrere verschiedene Lösungen. Eine Lösung wäre, die Anzeige im Cockpit an den Amaturen neben oder hinter dem Lenkrad zu positionieren. Das hätte den Vorteil, dass die Anzeige starr an einer Stelle positioniert wäre und der Fahrer sich darauf einstellen könnte davon abzulesen. Leider ist in einem derartigen Monoposto wie sie für die Formula Student gebaut werden nicht genug Platz vorhanden.
Eine weitere Möglichkeit der Positionierung ist das Lenkrad. Zwar dreht sich die Anzeige während des Lenkens mit, aber in scharfen Kurven schaut der Fahrer eigentlich nicht auf die Anzeige. Weil der Fahrer in einer liegenden Haltung fährt, hat er eine gute Sicht auf die Anzeige, denn das Lenkrad sitzt in einer guten Entfernung und Winkel zum Auge, sodass er die Rennstrecke nicht lange aus den Augen hat, wie Abbildung 2.6 und 2.7 zeigen. Wegen der

Entfernung, wird die Akkommodation des Auges nicht so stark beansprucht. Je nach seiner Größe, schaut der Fahrer in einem Winkel von 20 ° bis 25 ° und mit einer Entfernung von etwa 46 cm auf das Display.

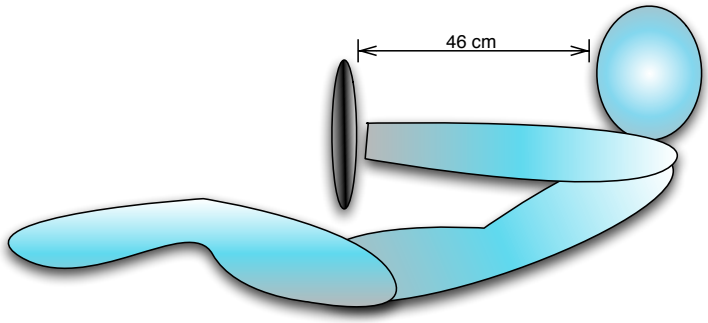

Abbildung 2.6: Entfernung vom Auge des Fahrers und dem Lenkrad

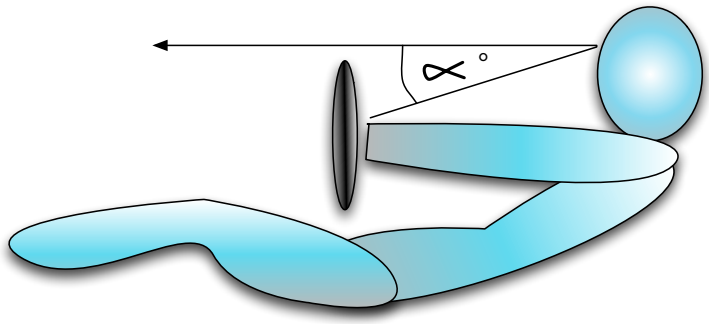

Abbildung 2.7: Der Winkel des Blickfeldes vom Fahrer

Ausgeschlossen wird die Positionierung der Anzeige am Helm. Die Anzeige, direkt am Helm, könnte den Fahrer in seiner Sicht stören oder die Sicht einschränken. Zudem ist in einem Helm nicht viel Platz zum verbauen einer Anzeige-Hardware. Ganz geschweige von der Sicherheit die vielleicht nicht mehr vom Helm gewährleistet werden kann, wenn im Innenraum harte Gegenstände verbaut werden und der Innenraum modifiziert wird. Außerdem ist die kurze Entfernung des Auges vom Fahrer zum Viesier suboptimal. Die andauernd starke Akkommodation des Auges beim Blick auf die Anzeige wäre eine zusätzliche starke Belastung für den Fahrer (siehe Abb. 2.8). Eine Vorrichtung mit Spiegeln, um die Akkommodation klein zu halten, zu bauen, wäre zudem ein viel zu großer Aufwand.

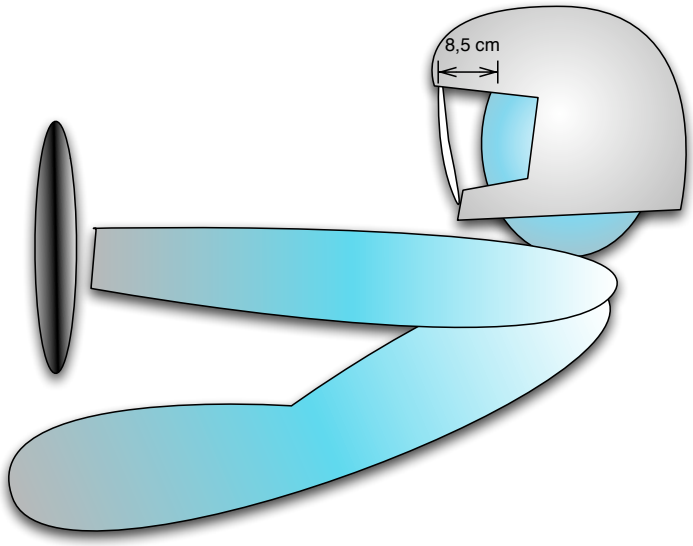

Abbildung 2.8: Entfernung vom Auge des Fahrer bis zum Visier des Helmes

2.6.3 Systemanalyse

Der Rennwagen besitzt eine Telemetrie. Sie besteht aus aus einem CAN-Bus und daran angebundenen Sensoren, einem WLAN-Modul und einer Data-Log-Einheit. So ist der CAN-Bus das Herzstück des Informationsaustausches im Rennwagen, über den Daten nach außen und umgekehrt gegeben werden. Eine Anzeige sollte demnach die vorhandene Datenstruktur nutzen und sich die Daten vom CAN-Bus holen. Wichtig dabei ist, dass die Software erweiterbar ist, wie der CAN-Bus und seine Architektur. Die Architektur des CAN-Busses wird in Kapitel 4.4 beschrieben.

2.6.4 Marktanalyse

Es gibt verschiedene Technologien für visuelle Anzeigen. Man hat die Wahl zwischen *analogen Anzeigen, Segmentanzeigen, vollgrafischen Displays* oder *Projektoren*.

Analoge Anzeigen

Unter den analogen Anzeigen gibt es z.B.

- Zeigeranzeigen : zeigen ein Wert mit einem Zeiger auf einer Skale an
- Pegelanzeigen : zeigen einen Füllstand an
- Thermometer : zeigen eine Themperatur an.

Die analogen Anzeigen haben den nachteil, dass die starr sind und nicht die Art der Darstellung verändern können, falls das gefordert ist.

Segmentanzeigen

Bei den Segmentanzeigen handelt es sich um eine zweifarbige Anzeige, die normalerweise Zahlen darstellt, wie die Sieben-Segment-Anzeige (siehe 2.9). Diese Art der Darstellung ist starr und beschrenkt in der Darstellung.

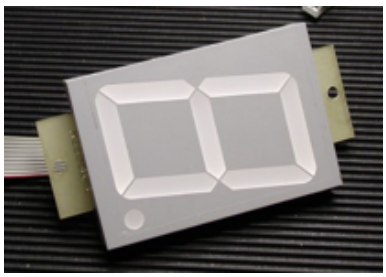

Abbildung 2.9: Sieben-Segment-Anzeige

vollgrafische Displays

Bei den vollgrafischen Displays gibt es so viele verschiedene Technologien, die jede ihre Nische hat. Folgend einmal aufgelistet, welche Displays-Technologien anhand ihrer Eigenschaften erörtert werden.

- CRT
- LCD
- OLED
- Plasma

- SED
- Ra-Tek

Für den Nutzungskontext sind die wichtigsten Eigenschaften die Helligkeit, der Kontrast, die Auflösung und der Energie- und Platzverbrauch.
Wie aus dem Kapitel 2.4 hervorgeht, steht das Display, falls es am Lenkrad oder im Cockpit installiert wird, gelegentlich unter direktem Sonnenlicht. Um die Daten vom Display trotzdem noch ablesen zu können, sollte die Helligkeit demensprechend stark sein. Für das Ablesen bei Tageslicht sollte die Helligkeit des Displays mindestens 1000cd/m^2[2] oder mehr betragen (vgl. Steinbacher, 2006).
Man unterscheidet in der Regel zwischen drei Lichtverhältnissen:

- Nacht^G - wenn in einer dunklen Umgebung die an einem Objekt zu messende maximale Beleuchtungsstärke 2 lx[3] nicht Überschreitet
- Tag^G - die gemessene Umgebungsbeleuchtung an der Anzeige soll 3 klx betragen.
- Sonnenlicht^G - wenn die an einem Objekt gemessene Beleuchtungsstärke $\geq$ 45 klx beträgt

(vgl. Seite 8 DIN EN ISO 15008, 2003)

Der Kontrast^G geht mit der Helligkeit einher und beschreibt das Verhältnis zwischen hell und dunkel in einem Bereich. Bei starkem Kontrast lassen sich Daten wie Zahlen oder Füllstände, soweit die Farben richtig gewählt wurden, gut ablesen. Gute Kontraste sind Farben die direkt zu ihren inversen Faben stehen (schwarz-weiss, blau-gelb u.s.w).
Die Auflösung gibt an, wieviel Pixel das Display besitzt. Um so mehr Pixel ein Display hat, um so feinere Darstellungen kann man auf dem Display darstellen.
Beispiel: Falls man einen einfachen Zeiger, der sich um 360° drehen kann, auf dem Display zeichnen möchte, sollte die Anzahl der Pixel so gewählt sein, dass der Zeiger wie ein gerader Strich aussieht. Ansonsten erhällt man ein Zickzack-Muster als Zeiger (siehe Abb. 2.10 und 2.11).

Der Energie- bzw. Stromverbrauch sollte möglichst gering sein. Im Rennwagen gibt es zwar einen Generator der die Bleibatterie im Rennwagen während der Fahrt wieder auflädt, nur verbraucht der Rennwagen durch stärkeren Energiebedarf mehr Treibstoff und der Verbrauch von Treibstoff sollte klein gehalten werden. Die Größe des Displays sollte möglichst groß gewählt werden, damit die Daten gut ablesbar sind und mindestens vier bis maximal sieben Daten gleichzeitig auf dem Display passen. Zudem sollte es aber, aus Gründen von Platzmangel, dünn sein.

[2] cd (Candela) : Messeinheit für die Lichtstärke
[3] lx (Lux): Einheit für Beleuchtungsstärke

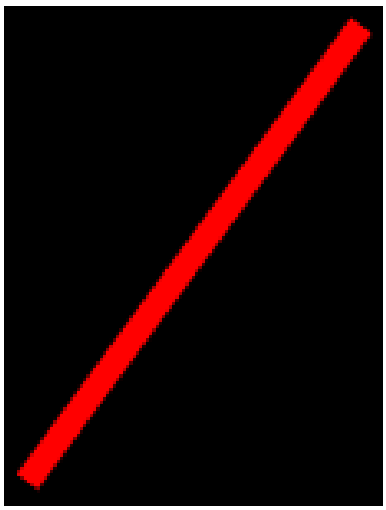

Abbildung 2.10: Auflösung 240x320

CRT Displays kommen daher nicht in Frage, weil sie zu viel Platz einnehmen und sehr schwer sind.

LCD^G sind eine gute Wahl, weil sie mit einigen Eigenschaften in das Profil der Anforderungen passen. Sie sind dünn, können eine relativ hohe Auflösung haben und sind zudem nicht so teuer.
Die Helligkeit und der Kontrast hängen stark von der vorzufindenen Technologie der Beleuchtung ab.

Es gibt drei Grundarten der Ausleuchtung von Displays (vgl. Organization, 2008).

- Transmissiv : ist eine Licht-durchlässige Art bei der eine Hintergrundbeleuchtung benötigt wird.
- Reflexiv : Das Umgebungslicht wird auf dem Bildschirm reflektiert und als Beleuchtung des Displays genutzt.
- Transreflexiv : Diese Art ist eine Mischung aus den transmissiven und reflexiven Displays. Bei schwachem Umgebungslicht steht eine Hintergrundbeleuchtung zur Verfügung. Zudem wird das Umgebungslicht genutzt und reflektiert und um auch bei starken Tageslicht etwas zu erkennen. Leider sind Displays mit diesr Art der Beleuchtung sehr kontrastarm.

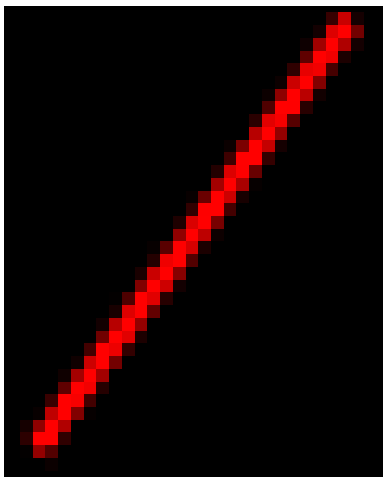

Abbildung 2.11: Auflösung 50x70

CCFL[4] oder LED's[5] werden bei LC-Displays als gängige Hintergrundbeleuchtungen genutzt. LED's werden immer häufiger dort eingesetzt wo wenig Platz vorhanden ist. Aber nicht nur ihre Größe sondern auch der kleine Energieverbauch im gegensatz zu CCFL, die gute Lichtausbeute und Farbtreue sprechen für die LED-Technik (vgl. Winter, 2007).

OLED^G's sind für den Gebrauch von Anzeigen gut geeignet. Sie brauchen keine Hintergrundbeleuchtung, weil die Anzeige selber leuchtet und können hauchdünn produziert werden. Außerdem ist deren Stromverbrauch noch geringer als bei LC-Displays. Der Nachteil dieser Technologie ist der relativ zum LCD hohe Anschaffungspreis und geringe Lebensdauer. Für den Embedded Bereich - somit auch in Automobilen - ist OLED bestens geeignet.

Plasma^G ist eine Technik mit Gasen, die auf große Displays ausgelegt ist. Der Vorteil dieser Technologie ist der starke Kontrast. Leider sind Plasma Dislay nicht für den embedded oder mobilen Bereich geeignet, weil sie nicht so dünn sind und auch nicht so scharf in der Bildqualität wie LCD.

SED[6] ist eine recht neue Technologie. Sie vereinigt die Brillianz der Bilder von CRT Displays, die flache Tiefe und den geringen Stromverbrauch der LCD. Diese Technik ist allerdings noch nicht zu kaufen und wird erst Ende 2008 von Canon und Toshiba auf dem angeboten. Außerdem ist diese Technik bisher nur für TV über 40“ Bilddiagonale und größer konzipiert (vgl. Strasser, 2008).

[4] CCFL = Cold cathode fluorescent lamps
[5] LED = light-emitting diode
[6] SED : Surface-Conduction Electron-Emitter Display

Abbildung 2.12: OLED Display im Auto

„Trotz des verhältnismäßig geringen Stromverbrauchs scheint der Einsatz in mobilen Endgeräten derzeit weniger sinnvoll, da noch Glas als Träger eingesetzt wird, was schwer und anfällig ist. Zudem dürften hier andere Techniken in den nächsten Jahren zu besseren Alternativen heranreifen, beispielsweise OLEDs, so Toshiba.“(siehe Ihlenfeld, 2008)

Grafische Frameworks

Es gibt viele grafische Frameworks in verschiedenen Programmiersprachen. All diese Frameworks zu erörtern würde an dieser Stelle den Rahmen der Arbeit sprengen. Es sollte eine Programmiersprache gewählt werden, die den Anforderungen entspricht und Hardware nah ist. Eine Programmiersprache für den Einsatz im embedded Bereich wäre z.B „C“ oder „C++“. Diese Sprachen sind perfomant und lassen im hohen Maße die Kontrolle über den Speicher zu. Speicher ist im Embedded Systemen wenig vorhanden, wie auch in unserem. Die Hardware, die in der Studienarbeit von Johann-Nikolaus Andreae (Andreae, 2008) entwickelt wird, hat einen Datenspeicher mit 256 MB und einen Arbeitsspeicher mit 64 MB. QT Embedded - auch unter den Namen Qtopia bekannt - ist ein Applikations-Framework von der Firma Trolltech. QT ist ein ausgreiftes Framework, dass schon seit Jahren für GUI^G-Entwicklung bei Desktop Anwendungen genutzt wird. Die Vorteile die QT Embedded bietet und sind folgende:

- Signal & Slot - eine threadsichere Implementierung eines Publisher-Subscriber-Konzepts[G]
- optimiert für Linux Embedded
- Implementiert in der Programmiersprache C++ mit vielen nützlichen erweiterten Klassen wie XML-Parser
- viele vordefinierte Grafik-Klassen
- ausführliche Documentation
- Backing Store - ein Mechanismus zum schnellen zeichnen von Widgets; besonders bei halb-transparenz oder transparenz (siehe Backing Store[7])
- Direct Painting - eine Möglichkeit die Hardware ohne Framebuffer direkt zu Manipulieren und darauf zu zeichnen (siehe Direct Painting[8])

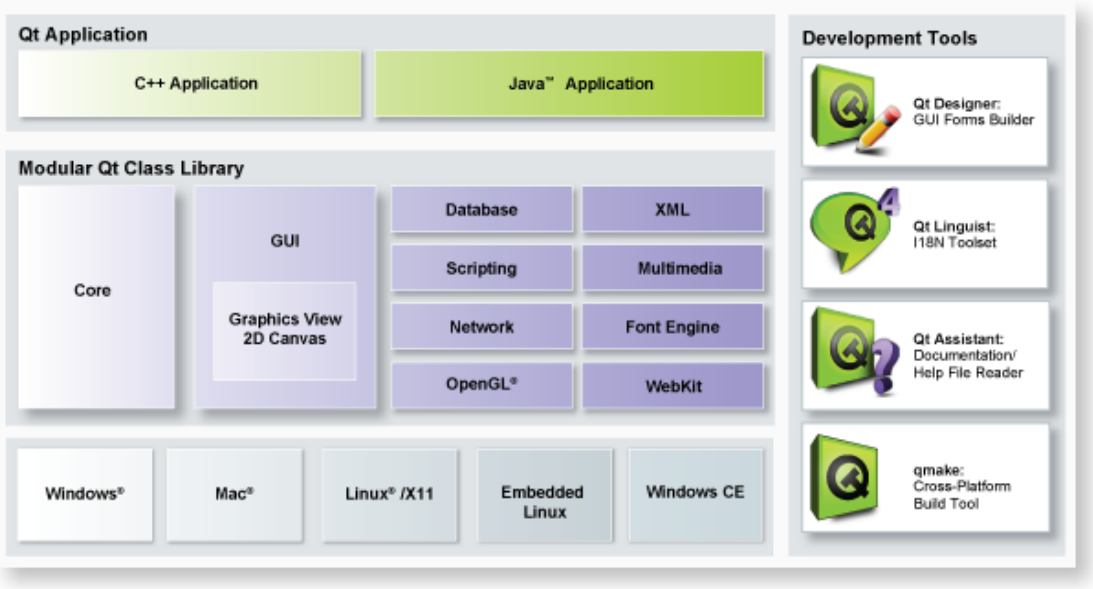

Abbildung 2.13: Grobe Übersicht über die QT-API

[7] http://doc.trolltech.com/qq/qq16-background.html
[8] http://doc.trolltech.com/4.3/qtopiacore-architecture.html#direct-painting

2.6.5 Anforderungen

Die oberste Priorität ist die Lesbarkeit der Daten auf dem Display. Der Fahrer muss die Daten in einem Bruchteil einer Sekunde erfassen können, um nicht zu sehr vom Fahren abgelenkt zu werden.
Die Anforderungen lassen sich in Hardware- und in Software-Anforderungen unterteilen.

Anforderungen an die Hardware und Umgebung

- die Hardware sollte Spritz-Wasserdicht sein
- Temperaturressistenz oder Kühlung
- starker Kontrast und möglichst hohe Helligkeit der Anzeige
- Schutz vor Vibrationen
- Schutz vor elektromagnetischen Feldern
- an die Software angepasstes Interface

Anforderungenn an die Software

Bei der Usability gerechten Entwicklung der Software sollte man sich nach den Richtlinien der DIN-Norm 9241 richten. Sie gibt Hinweise für die Gestaltung von Benutzerschnittstellen und Anforderungen für ein System, um effizient, effektiv und zufriedenstellend zu sein.

Die Abbildung 2.2 zeigt an, welche Grundsätze für die Gestalung von Dialogen berücksichtigt werden müssen. Die Dialogführung sollte

- Aufgabenangemessen
- Selbstbeschreibend
- Erwartungskonform
- Individualisierbar
- Lernförderlich
- Steuerbar
- Fehlertolerant

sein.

Die *funktionalen Anforderungen* an die Software sind:

- Anzeige der aktuellen Werte an den Sensoren
- visuelle Unterstützung für den Fahrer bei der Fahrt
- einstellen der Anzeige über ein Interface

2.7 Ziele

Die Ziele dieser Arbeit sind die Anforderungen an das System weitesgehend zu implementieren. Es soll eine wartungsfreundliche, angemessene, performante und stabile Software entwickelt werden.

3 Projektplanung

Die Projektplanung spielt eine sehr wichtige Rolle bei der Entwicklung. Vorrausgesetzt man plant die Zeit realistisch ein und teilt die Aufgaben bei der Entwicklung gut auf, kann die Projektplanung helfen Zeit zu sparen und Diskrepanzen bei der Entwicklung in der zeitlichen Reihenfolge vorzubeugen und die damit verbundenen Probleme zu bewältigen.
Gerade bei zeitkritischen Entwicklungen, wie es in dem Fall des Rennwagens des HAWKS Racing Team's ist, sollte eine gute Planung nicht fehlen. Die Entwicklung soll weniger als fünf Monate betragen. Dabei müssen Resourcen, wie Arbeitskräfte und Material, koordiniert, Fertigung und Beschaffung von Materialen bei Dritten eingeplant und zusätztliche Zeit für den „worst case", wie unerwartete Komplikationen und Änderungen, eingeräumt werden.
Folgend einmal aufgelistet, worauf bei der Planung geachtet werden soll:

- Arbeitskräfte
- Material
- Zeit für die Fertigung
- Zeit für die Beschaffung
- „Worst Case"

Zu Beginn sollten die Stakeholder, die wesentlich für die Entwicklung des Systems sind, gefiltert werden. Es sind die Zulieferer des Materials, externe Fertigung - falls welche benötigt wird - und die Team-Mitglieder, die direkt mit dem Telemetriesystem, dem Lenkrad oder dem Interieur zu tun haben, die in der Planung erfasst werden müssen.

Ganz besonders mit Johann-Nikolaus Andreae (Andreae, 2008) und dem Konstrukteur des Lenkrades sollte die Zusammenarbeit gut durchdacht und geplant sein. Johann-Nikolaus Andreae ist für die technische Hardware und das darauf laufende Betriebssystem zuständig (siehe Tab. 1.1). Der Konstrukteur des Lenkrades ist für die Maße und Form des Lenkrades verantwortlich.

An dieser Stelle soll nochmal darauf hingewiesen werden, dass nicht alle in dieser Arbeit erschlossenen Anforderungen umsetzbar waren. Der Grund für diesen Umstand sind

die Meinungsverschiedenheiten über die Gestaltung des Lenkrades für den Rennwagen. Infolgedessen wurden Anforderungen vom Konstrukteur des Lankrades gestellt. Diese Anforderungen werden in Kapitel 5 diskutiert.

Für den Prototypen der laufenden Software wird die Hardware im Entwicklungsprozess nicht benötigt. QT Embedded bietet einen virtuellen Framebuffer^G , der es erlaubt eine Simulation des Lankrades darzustellen. Die Abhängigkeiten zur Hardware ist daher nicht so stark. Der Entwicklungsprozess und Teile der Evaluierung der Software kann ohne die Hardware durchgeführt werden. Viele Prozesse der Entwicklung können so parallelisiert werden.
Aufgrund der Abhängigkeiten die aber noch bestehen, sollte eine Zeitplanung mit Meilensteinen erstellt werden. Es bietet sich daher ein Gantt-Diagramm für den kontrollierten Ablauf aller Vorgänge der Entwicklung an.

Als aller erstes wird eine Zeit für die Einarbeitung und Analyse eingeplant. Nach der Analyse, wenn die Anforderungen stehen und man die für die Entwicklung benötigten Komponenten identifiziert hat, wird mit dem Design des Systems begonnen.

Die Abbildung 3.1 zeigt die Zeitplanung an.

Ein wichtiger Meilenstein, der in der Abbligung 3.1 ein wenig untergeht, ist der Rollout des Rennwagens. Der Rollout ist ein Termin, an dem der Wagen als fahrbereit und fertig der Presse vorgestellt wird.

Wie man in der Abbildung 3.1 sieht, ist zum Rollout und zwischen dem Rollout und den wichtigen Events der Formula Student in Silverstone und Hockenheim noch Luft, um kleine Änderungen an Software oder Hardware verzunehmen. Diese Lücken sind dann für den „worst case“ eingeplant.

Die Zusammenarbeit mit dem Konstrukteur des Lenkrades erwies sich mehr als schwierig, weil es zum Teil Komplikationen bei der Konstruktion und Fertigung gab. Daraufhin wurde entschlossen, die Zeitplanung ohne den Konstrukteur durchzuführen und das System seperat anhand der Anforderungen des Konstrukteurs zu implementieren.

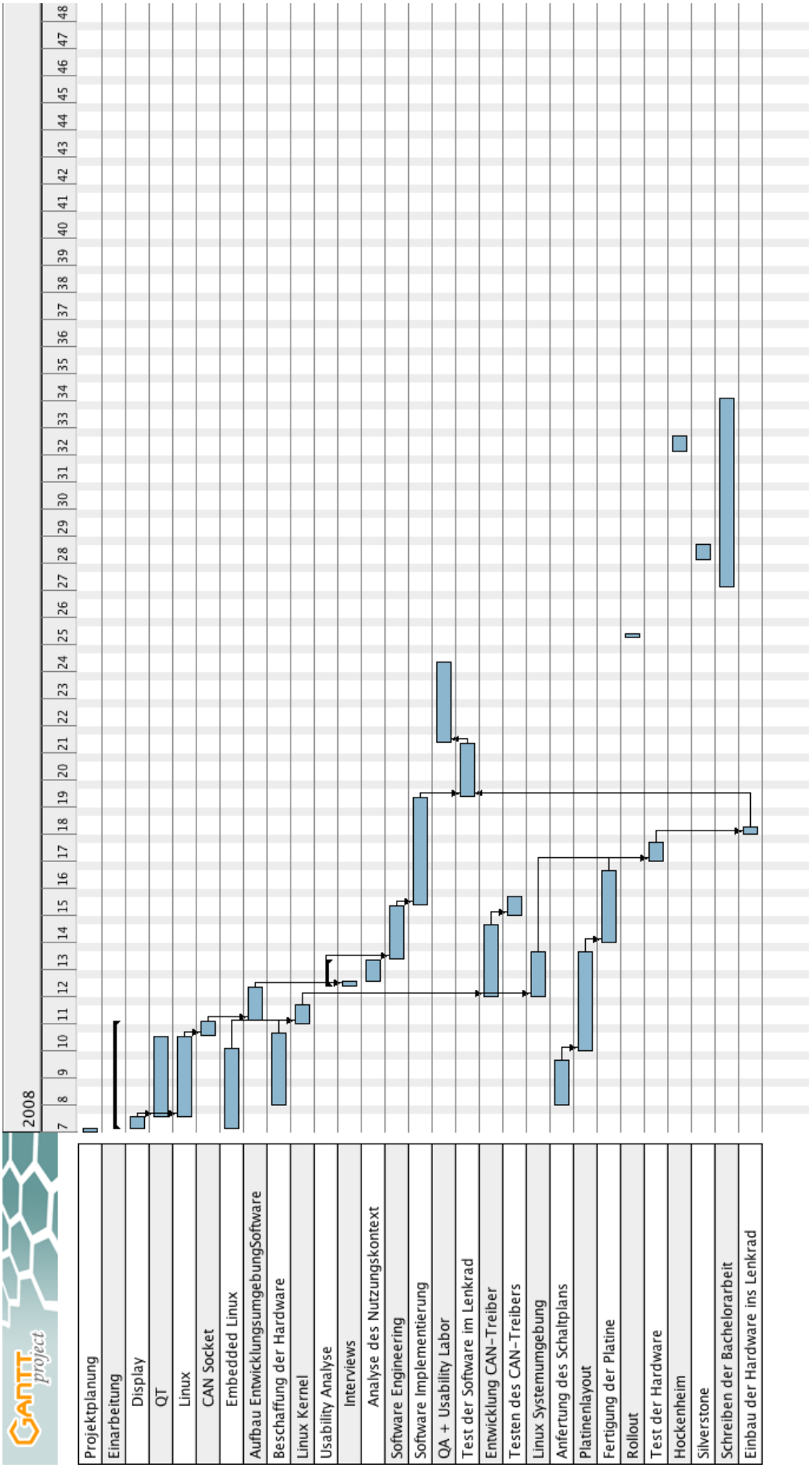

Abbildung 3.1: Zeitplanung mit einem Gantt-Diagramm

4 Die Umsetzung

In Kapitel 2 wurden die Anforderungen an das System erhoben. Anhand dieser Daten wird in diesem Kapitel festgelegt und erklärt wie das System aufgebaut wird.
Das System besteht aus einem Interface zur Steuerung der Anzeige und einem TFT-LCD-Display als Anzeige für die Daten. Das Display sollte möglichst weit oben am Lenkrad montiert werden, damit der Winkel bei der Sicht des Fahrers von der Straße zum Display möglichst klein ist 2.7. Die Abbildungen 4.3 und 4.2 zeigen einen ersten Versuch die Anforderungen umzusetzen.

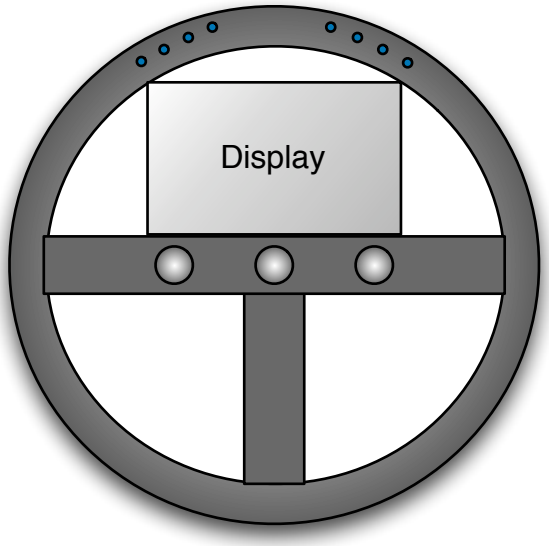

Abbildung 4.1: rundes Lenkrad mit Display

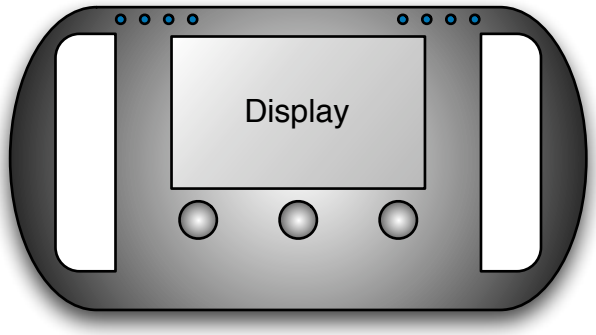

Abbildung 4.2: eckiges Lenkrad mit Display

4.1 Display

Damit möglichst viele Daten gleichzeitig auf dem Display untergebracht werden können und dabei die Lesbarkeit[G] für den Fahrer gut ist, muss das Display relativ groß gewählt werden. Ein Display, das zum Beispiel sechs Daten gleichzeitig anzeigen soll, muss dementsprechend größer gewählt werden, als ein Display, dass weniger anzeigt.
Die Norm DIN EN ISO 15008 (DIN EN ISO 15008, 2003) gibt Informationen dazu, welche Höhe alphanumerische Zeichen in Bezug zu der Entfernung, aus der gelesen wird, haben müssen, damit man die Zeichen gut lesen kann.

Radiant	Eignung
6,98 10^{-3}	Empfehlenswert
5,83 10^{-3}	akzeptabel, wenn Farbe als kodierende Größe eingesetzt wird
5,24 10^{-3}	akzeptabel, wenn Farbe nicht als kodierende Größe eingesetzt wird
4,36 10^{-3}	mit Einschränkungen [1]

Tabelle 4.1: Höhe der Zeichen (vgl. Seite 13 DIN EN ISO 15008, 2003)

Die Wahl der Größe von alphanumerischen und anderen Symbolen hängt nicht nur von der Entfernung des Fahrers zum Lenkrad ab, sondern auch von anderen Anforderungen wie die Lesegeschwindigkeit oder Genauigkeit mit der Abgelesen werden soll.
Wegen der großen Bedeutung der Lesbarkeit der alphanumerischen Zeichen auf dem Display, sollte die Zeichenmatrix[2] mindestens 7x9 (Breite/Höhe) Pixel groß sein (vgl. Seite 13 DIN EN ISO 15008, 2003).

[1] Wenn die Anforderungen an die Genauigkeit und Lesegeschwindigkeit gering sind oder wenn die Lesbarkeit eine untergeordnete Rolle spielt (z.B bei Indezes)
[2] die Matrix von Pixeln, die ein Zeichen ergeben

Die Vorgabe vom Lenkradkonstrukteur für die Größe des Displays auf dem Lenkrade fällt in dem Fall leider sehr klein aus. Das Display darf laut Konstrukteur nur 5cm Breite und 7cm Höhe betragen.

Abbildung 4.3: CAD Zeichnung des Lenkrades mit Vorgaben des Konstrukteurs

Daraufhin wurde nach einem Sponsor gesucht, der ein Display in dieser Größe anbot. Letztendlich wurde dem HAWKS Racing Team ein Display (Hitachi TX07D09VM1CBB) mit folgenden Specifikationen gesponsort:

Das Display erfüllt vielleicht nicht alle Anforderungen, ist aber mit seiner Helligkeit über 420cd/m^2 für Umgebungshelligkeiten wie Tageslicht (siehe Kapitel 2) gut geeignet. Bei Helligkeiten wie Sonnenlicht sollte aber schon ein tranreflexives Display in Betracht gezogen werden.

Für die Ablesbarkeit von alphanumerischen Zeichen ergeben sich nun für die Höhe folgende Rechnungen:

r = Radiant
d = Entfernung vom Auge zur Display-Mitte (siehe Abb. 2.7 und 2.6)
p = Höhe der gewählten Zeichenmatrix (7x9)

Eigenschaft	Hitachi TX07D09VM1CBB	Wünschenswert
Maße		
Gesammtfläche	50,54mm(Breite)x68,62mm(Höhe)	ca. 140mm(Breite)x100mm(Höhe)
Bildfläche	41,04mm(Breite)x54,72mm(Höhe)	ca. 140mm(Breite)x100mm(Höhe)
Pixelgröße	0,171mm x 0,171mm	/
Bildeigenschaften		
Auflösung	240 x 320 Pixel	240 x 320 Pixel
LCD Typ	transmissives vollgrafisches TFT[G]	transreflexives vollgrafisches LCD
Farben	18 bit Farben (262 000)	8 bit Farben oder mehr
HGB[3] Typ	weiße LED	weiße oder RGB-LED
HGB Helligkeit	420 cd/m^2	1000 cd/m^2
Bildwiederholrate	60 Hz	60 Hz oder höher
Kontrast	300:1	je nach Typ oder Beleuchtung anders
Spiegeleigenschaft	keine Informationen	entspiegelt (um nicht zu blenden)
Vibrationen	5 bis 100 Hz, 11,76 m/s^2	/
Blickwinkel	25°	> 20°

Tabelle 4.2: Vergleich Hitachi TX07D09VM1CBB mit den Anforderungen

q = Pixel-Höhe
h = minimale Höhe

$$h_1 = r * d \tag{4.1}$$

oder

$$h_2 = p * q \tag{4.2}$$

Wenn man das größere h von beiden Rechnungen nimmt, erhällt man die minimale Größe, die ein Zeichen auf dem Display haben sollte.
Für die Entfernung vom Auge zur Display-Mitte ist die waagerechte Länge vom Auge zum Lenkrad und die Höhe bekannt. Durch den Satz des Pythagoras erhällt man folgende Rechnung:

$$d = \sqrt{46^2 * 18^2} \tag{4.3}$$

$$h_1 = 6,98 * 10^{-3} * \sqrt{46^2 * 18^2} = 0,3447 cm \tag{4.4}$$

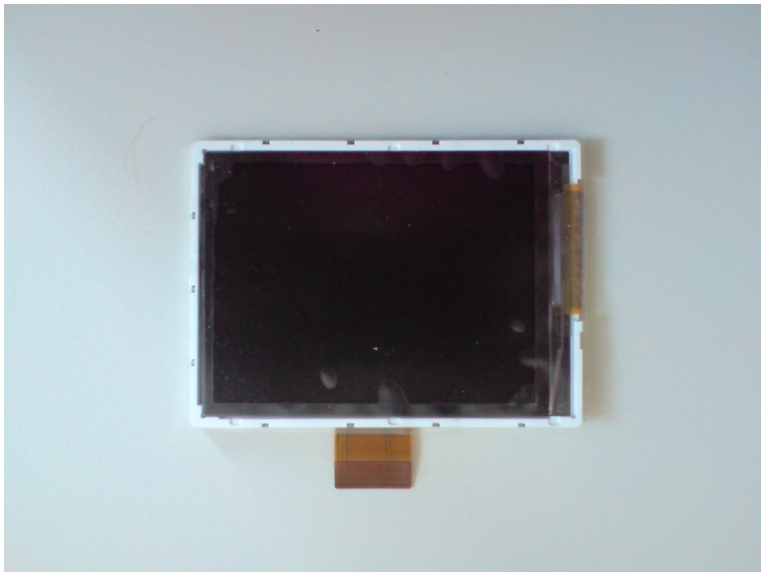

Abbildung 4.4: Das Lenkraddisplay Hitachi TX07D09VM1CBB

oder

$$h_2 = 0,171 * 9 = 1,539mm \tag{4.5}$$

$$h = max[h_1, h_2] \tag{4.6}$$

Um auf dem Display Hitachi TX07D09VM1CBB die alphanumerischen Zeichen gut ablesen zu können, sollte die Höhe den Rechnungen nach h = 3,5 mm betragen.

Die Umsetzung der GUI^G und dessen Farbkombinationen werden im Kapitel 5 ausgiebig besprochen.

Aus dem Grund, dass das Display kleiner ist, als vergeschlagen, muss das Layout so gewählt werden, dass man die Daten trotzdem schnell und gut ablesen kann. Abgesehen von den alphanumerischen Symbolen, sollen auch grafische Anzeigen die Daten anzeigen. Sie sollen die Daten durch Gestalt, Vertautheit und farblicher Hervorhebung für den Fahrer schneller und leichter erfassbar machen als alphanumerische Symbole (vgl. Seite 63 ff Dahm, 2006).

Es wird Empfohlen zwei bis maximal vier Daten gleichzeitig auf dem Display anzuzeigen, damit die Symbole oder grafischen Elemente groß genug dargestellt werden können und der Fahrer die Daten schnell und präzise genug ablesen kann.

4.2 Interface

Das Interface sollte so einfach wie möglich gestaltet werden, um den Fahrer nicht zu belasten oder von der eigentlichen Aufgabe, dem Fahren, abzuhalten. Um zu verhindern, dass der Fahrer während der Fahrt die Menüsteuerung benutzt, sollte das Interface in der Mitte des Lenkrades positioniert werden. Wenn das Interface nämlich nicht erreichbar ist ohne die Hände vom Lenkrad zu nehmen, wird der Fahrer weder beabsichtigt noch unbeabsichtigt des Interface benutzen.

Das Interface sollte mit Knöpfen ausgestattet sein, die einen relativ starken Druckpunkt besitzen, damit der Fahrer beim drücken mit Handschuhen, eine Rückmeldung fühlt. Die Position und Anzahl der Knöpfe soll an die Software angepasst werden. Um in jedem Zustand der Software zu erkennen, welche Aktion die Knöpfe auslösen, wird mit sogennanten Soft-Key-Knöpfen gearbeitet. Soft-Key-Knöpfe sind Knöpfe die entlang von Displays angebracht und durch die Anzeige auf dem Display dahingehend unterstützt werden, dass anzeigt wird, welche Aktion beim drücken der Knöpfe ausgelöst wird. Dabei muss beachtet werden, dass das Gestaltgesetz der Gruppierung durch Nähe eingehalten wird, damit man erkennt, welcher Knopf zu welcher Label auf dem Display gehört (vgl. Seite 60 Dahm, 2006).

Für das bloße Wechseln von der Darstellung der Daten auf dem Display und dem laden von Profilen reicht ein Interface von zwei bis drei Knöpfen. Ein oder zwei Knöpfe zum Wählen und ein Knopf für die Bestätigung.
In der Zukunft sollen aber noch Eingaben hinzukommen, sodass komplexere Menüführungen möglich sein sollen. Daher ist ein 3 Knöpfe Interface dir bessere Wahl. Die genaue Anordnung der Knöpfe wird im Kapitel 5 erörtert.

4.3 Schaltzeitpunkt

Der beste Schaltzeitpunkt[4] ist für den Fahrer außerordentlich wichtig. Wenn der Fahrer die Gänge in den richtigen Drehzahlen schaltet, kann der Rennwagen die optimale Kraftumsetzung auf die Straße überführen und dadurch wertvolle Sekunden in der Beschleunigung gewinnen.
Der Schaltzeitpunkt ist von zwei Faktoren abhängig:

- Drehzahl
- Gang

Die Drehzahl bei der geschaltet werden sollte, variiert von Gang zu Gang. Diese Daten müssen in der Software erfasst werden und konfigurierbar sein, falls man optimieren möchte oder Veränderungen am Wagen vorgenommen hat.

4.3.1 mittels Display

Das vollgrafische Display kann dabei helfen den Fahrer auf das Schalten vorzubereiten und ihm den optimalen Moment mitzuteilen. Es gibt verschiedene Gestaltungsmöglichkeiten dies zu tun.

1. Für den optimalen Moment des Schaltens blitzt das ganze Display in einem weißgelben Farbton kurz auf. Eine kurze Veränderung der Farbwerte sollte in einem Sichtwinkel von 20○ bis 25○ auffallen. Allerdings fällt so etwas nur auf, wenn es eine starke farbliche Veränderung ist. Das bedeutet, dass die Grundfarbe auf dem Display im optimalen Fall invertiert wird.
2. Eine weitere Möglichkeit wäre die grafische Darstellung. Man kann den Fahrer mit einer Skala und einem Zeiger oder einer Segmentanzeige einerseits darauf vorbereiten, dass der Schaltmoment kurz bevor steht, und andererseits den richtigen Moment grafisch anzeigen. Die grafisch dargestellten Elemente werden nochmal in Kapitel 5 aufgegriffen.
3. Zudem kann man die grafische Anzeige noch farblich unterstützen.

[4] damit ist das Gänge schalten im Wagen gemeint

4.3.2 mittels LED

Mit LED's wird im Rennsport gearbeitet.
Sie zeigen den Fahrern in einer Reihe aufgestellten den optimalen Schaltmoment, indem sie nacheinander leuchten.

LED haben den Vorteil, dass sie wenig Strom verbrauchen und sehr hell sein können. Außerdem kann man bei den LED's farblich kodieren und somit mit wenigen LED's viele Zustände anzeigen. In dieser Arbeit wird empfohlen vier LED's links oben und vier LED's rechts oben auf dem Lenkrad aufzureihen (siehe Abb. 4.1 & 4.2). Auf jeder Seite eine Reihe LED's, damit der Fahrer beim Einschlag des Lenkrades immer eine Reihe LED's sehen kann. In Abbildung 4.5 sieht man jeweils zwei Löcher auf jeder Seite.

Eine weitere Möglichkeit die LED's anzubringen wäre das Dashboard[G] Um die LED's aber auf dem Dashboard kontinuierlich zu sehen, müsste das Dashboard aber viel höher als im Rennwagen des HAWKS Racing Team's sitzen (siehe Abb. 4.5). Daher wird davon abgesehen, die LED's am Dashboard anzubringen.

Abbildung 4.5: Sicht des Fahrers aus dem Cockpit

In dem Fall des Rennwagens werden Duo-LED's verwendet. Sie haben zwei Farben, die angesteuert werden können, und sind dadurch fähig, drei Farben darzustellen

1. Farbe1
2. Farbe2
3. Farbe1+Farbe2

Man könnte auch RGB-LED's benutzen. Sie bieten durch die Farbmischungen aus Rot, Grün und Blau eine große Menge an Farben an. In der Regel sind sie meist nicht so hell und die Farbmischung nicht so einfach in der Ansteuerung. Weiterhin werden so viele Farben nicht benötigt, sodass von den RBG-LED's an dieser Stelle abgesehen wird.

Acht Duo-LED's in Blau und Rot sollten ausreichen, den Fahrer zu informieren. Folgende Sequenz wurde ausgedacht:

1. Vorbereitend die blaue Farbe der aüßertsten LED's
2. weiter mit 2 LED's auf jeder Seite
3. mit 3 LED's auf jeder Seite
4. alle LED's leuchten in Blau um außzusagen, dass der Moment kurz bevor steht
5. Der Schaltmoment wird mit der Farbe Rot bei allen LED's angedeutet (siehe Abb. 4.6).

Die Sequenz geht von Außen nach Innen. Dabei soll der Fahrer durch die Farbe Blau auf den Schaltmoment vorbereitet werden. Die rote Farbe soll den Fahrer dann verstärkt signalisieren zu Schalten.

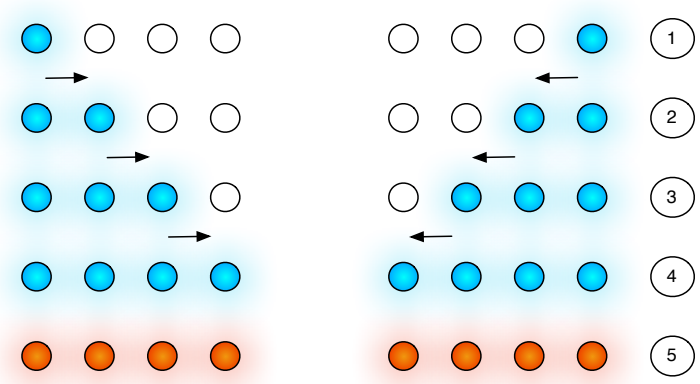

Abbildung 4.6: LED Reihenfolge

4.4 CAN-Bus

Der CAN-Bus im Rennwagen vom HAWKS Racing Team wurde von Sebastian Haase (Haase, 2007) und Simon Schuckert (Schuckert, 2007) entwickelt. Er basiert auf einer TTCAN^G-Architektur (time triggered CAN).

Der CAN-Bus vernetzt alle elektronischen Steuergeräte und Sensoren miteinander. Alle Daten die für die Anzeige auf dem Display gebraucht werden, kann man vom Bus lesen. Eine CAN-Nachricht ist grob in drei Teile unterteilt:

- CAN-ID : die ersten 2 Byte (eigentlich 11 bit) stehen für die CAN-ID. Anhand der CAN-ID kann man eine CAN-Nachricht identifizieren.
- Datenlänge : eine CAN-Nachricht kann verschieden lang sein. Ein Byte gibt an wie lang die Daten sind, die in der CAN-Nachricht mitgeliefert werden.
- Daten : Mittels der Datenlänge wird angegeben, wie lang die Daten sind. Eine CAN-Nachricht kann 0 bis 8 Byte Datenbeinhalten.

In sogenannten Matrixzyklen werden die CAN-Nachrichten zyklisch wiederholt. Das heißt, dass in einem Zyklus, dem Matrixzyklus, die Nachrichten ein oder mehrere Zeitfenster haben, in dem Sie gesendet werden. So ist die Geschwindigkeit des Datentransfers dadurch abhängig, wie lang ein Matrixzyklus ist und wieviele Zeitfenster einer CAN-Nachricht in dem Zyklus zugewiesen wurden. Für weitere Informationen über die Architektur des CAN-Busses wird auf die Bachelorarbeit von

Damit die einzelnen Komponenten, die auf dem CAN-Bus schreiben wollen, wissen wann ihr Zeitfenster ist, gibt es einen Timemaster. Der Timemaster ist eine Komponente die wiederholt CAN-Nachrichten schickt, die nur dafür da sind, alle Komponenten mit der Zeit des Timemasters zu synchronisieren. Für das bloße Lesen der CAN-Nachrichten vom Bus ist das aber unwichtig, daher soll an dieser Stelle nicht weiter auf die Architektur von TTCAN eingegangen werden.

Ein Matrixzyklus ist laut Simon Schuckert (siehe Seite 70 Schuckert, 2007) 40 ms lang. Demzufolge werden die Daten eines Sensors im idealen Fall mindestens 25 mal in der Sekunde auf dem CAN-Bus liegen. Das würde ausreichen, um eine grafische Anzeige auf dem Display fließend in der Bewegung aussehen zu lassen. Für den Eindruck einer Bewegung braucht man 22 Bilder pro Sekunde. Das PAL-Fernsehen als Beispiel, braucht 25 Bilder pro Sekunde (vgl. Seite 46 Dahm, 2006). Wenn man die grafischen Anzeigen auf dem Display

alle 40 ms neu Zeichnet, bekommt man das gefühl einer fließenden Bewegung.

Die Nachrichten auf dem CAN-Bus sind in Prioritäten aufgeteilt. Die Prioritäten sind in der CAN-ID kodiert als das erste Byte. Für das Lesen der Daten von den Sensoren, sind nur die Nachrichten mit der Priorität 6 von bedeutung. Das zweite Byte gibt die Quelle an. Jede Nachricht hat 8 Byte Datenlänge. In einer Nachricht sind Sensordaten von mehreren Sensoren vertreten. Die folgende Tabelle zeigt wie die Daten organisiert sind:

Die Daten stehen in der Byte-Reihenfolge Big-Endian in den Nachrichten. Das heißt, dass bei Daten die größer als ein Byte lang sind, darauf geachtet werden muss, dass das Byte mit den höchstwertigen Bits an der kleineren Adresse stehen - in unserem Fall in der Byte-Reihenfolge an der niedrigeren Stelle - gefolgt von dem Byte mit den niederwertigen Bits.
Wichtige Informationen für einen Sensor die aus der Tabelle zu entnehmen sind:

- Quelladresse - um bestimmen, in welcher CAN-Nachricht die Sensordaten sind
- Komponenten - um welchen Sensor es sich gerade handelt
- Datentyp - die Anzahl der Bytes für den Wert einer Komponente
- Byte-Offset - das erste Byte, das die Daten der Komponente beinhaltet
- Daten-Auflösung - die Daten-Auflösung der einzelnen Bits

Das lesen vom CAN-Bus wird in der Software abstrakt realisiert. Es wird über ein CAN-Socket gelesen, sodass man unabhängig vom lesenden Gerät ist. Der CAN-Socket ist so ähnlich aufgebaut wie ein TCP-Socket, mit dem Unterschied, dass alle Nachrichten Broadcast-Nachrichten sind. Man kann die Nachrichten nicht an ein bestimmtes Gerät schicken, sondern schickt eine Nachricht an alle. Was man aber kann, ist das filtern von Nachrichten nach der CAN-ID (vgl. Hartkopp, 2008). Der Vorteil dabei ist, dass der Code abstrakt bleibt. Man muss nicht mehr die Implementierung der Software an das Gerät anpassen. Seid der Linux Kernel Version 2.6.25 ist der CAN-Socket fester Bestandteil des Kernels.

Ein CAN-Frame ist folgendermaßen aufgebaut:

```
typedef __u32 canid_t;

struct can_frame {
    canid_t can_id;      /* 32 bit CAN_ID + EFF/RTR flags */
    __u8    can_dlc;     /* data length code: 0 .. 8 */
```

Nachrichten vom Typ Data Single (Byte-Reihenfolge: Big-Endian)

Komponenten / Werte	Quell-Adr.	B 0	B 1	B 2	B 3	B 4	B 5	B 6	B 7	Einheit	Bereich
Time Master Nr. 1 bis 7	0x01 - 0x07		u char u char u char u char	u char u char u char u char	u char u char u char u char			x x x x	0 1 2 3	Datum: dd.mm.yy Uhrzeit: hh.mm.ss Datum: dd.mm.yy Uhrzeit: hh.mm.ss	x = 0 -> BusMode: off x = 1 -> BusMode: record
Raddrehzahl vorne links	0x33		u short		u short		u short			RPM	0 bis ...
Raddrehzahl vorne rechts										km/h	0 bis 400
Geschw. vorne gemittelt											
Raddrehzahl hinten links	0x34		u short		u short		u short			RPM	0 bis ...
Raddrehzahl hinten rechts										km/h	0 bis 400
Geschw. hinten gemittelt											
EngineRoundCounter	0x35		u short		u short		u short		u char	1/bit	0 bis 65535
MAP										1mBar/bit	0 bis 65535
Phase1										0.1°/bit	0 bis 3553,5
TAir										0.6°/bit	-30 bis 129,375
TPS	0x36		u char	u short	u short	u short	u short			0,5%bit	0 bis 127,5
RPM										1rpm/bit	0 bis 65535
Klambda1										1e-4/bit	0 bis 99.9009
Inj1										1µS/bit	0 bis 65535
Inj2	0x37		u short		u short		u short		u char	1µS/bit	0 bis 65535
Inj3										1µS/bit	0 bis 65535
Inj4										1µS/bit	0 bis 65535
ValvPosition										0,4%bit	0 bis 100
Spark3	0x38		s short		s short		s short		u char	0,1°/bit	-3276,8 bis 3276,7
Spark4										0,1°/bit	-3276,8 bis 3276,7
Spark2										0,1°/bit	-3276,8 bis 3276,7
Sidestand										1/bit	0 bis 255
Spark1	0x39		s short	u char	u char	u char	s char	u char	u char	0,1°/bit	-3276,8 bis 3276,7
Lambda										0,01/bit	0 bis 2,55
VBatt										0,07V/bit	0 bis 18,05
DFarf										1/bit	-127 bis 128
TEngine										0.6°/bit	-30 bis 129,375
LambdaTarget										0,01/bit	0 bis 2,55
Speed	0x3A		u short		u char	u char	u char	u short		km/h	0 bis 255
Tipover										1/bit	0 bis 255
Gear										1/bit	0 bis 255
Active Block										1/bit	0 bis 255
BAP											
Lenkwinkel	0x3B		s short							Grad	-180 bis +180
Beschleunigung x (hinten)	0x3C		s short		s short		s short			mg	-6g bis 6g
Beschleunigung y (rechts)											
Beschleunigung z (oben)											
Drehrate	0x3D		s short		u short		s short		s char	°/s, positiv = UZS	12 bit
Drehratensensortemperatur										bit	12 bit, 1 K / 6.88 b
Drehratensensortemperatur										K	0 bis 595 K
Drehratensensortemperatur										°C	geschnitten auf ±127°C
Öldruck	0x3E		u char	u char	s short					dezibar	0 bis 10 bar
Öltemperatur										°C	0 bis ...
Ölschnorchelposition										° im UZS	0° bis 359°
Bremsdruck	0x3F		u char							bar	0 bis 200 bar
Gang	0x40		s char							0 neutr, -1 unbek.	0 bis 5

Abbildung 4.7: Organisation der Daten auf dem CAN-Bus

```
    __u8    data[8] __attribute__ ((aligned(8)));
};
```

Die Anwedung des CAN-Sockets ist analog zum TCP-Socket sehr einfach

```
unsigned int nbytes;
struct sockaddr_can addr;
struct ifreq ifr;
struct can_frame frame;

s = socket(PF_CAN, SOCK_RAW, CAN_RAW); // einen Socket öffnen

strcpy(ifr.ifr_name, "vcan0"); // Interface zuordnen

ioctl(s, SIOCGIFINDEX, &ifr); // Hardware konfigurieren und aktivieren

addr.can_family = AF_CAN;
addr.can_ifindex = ifr.ifr_ifindex;

bind(s, (struct sockaddr *)&addr, sizeof(addr)); // binden des Sockets
// an den CAN-Bus
```

Nun kann mittels der Methode read(...) vom CAN-Bus gelesen werden.

```
nbytes = read(s, &frame, sizeof(struct can_frame));
```

4.5 Profile

Über Profile, die man zu Beginn der Software laden kann, soll die *Individualisierbarkeit*, einer der Grundsätze der Dialoggestalung, erreicht werden. Aus dem Kapitel Personas (2.5.1) geht hervor, dass es allgemein zwei verschiedene Benutzer des Systems gibt. Um aber allen Benutzern - ganz besonders den Profi-Fahrern mit ihren eigenen Vorlieben - gerecht zu werden, soll das System Profile zum laden von vordefinierten Einstellungen besitzen.

Diese Profile sollen erlauben folgende Eigenschaften des Programms zu definieren:

- Farbschema

- Wahl zwischen drei oder mehreren verschiedenen Layout-Vorlagen, die größe und Positionierung angeben
- Positionierung der Anzeige-Elemente
- Wahl von verschiedenen Anzeigetypen
- Sliden von Seiten aktivieren/deaktivieren
- Zeit des Verschwindens der Menüsteuerung einstellen

Der Fahrer kann sich sein Profil für drei verschiedene Aufgaben einstellen. Für die *Aufgabenangemessenheit* der Anzeige gibt es „Driver-Modes“. Driver-Modes sollen eine Aufgabenstellung darstellen. Sie stellen die zur Aufgabe benötigten Anzeigen bereit. Sie bilden die Aufgaben wie folgt ab:

- Endurance
- Acceleration
- Training

Der Endurance-Driver-Mode soll die optimalen Anzeigelemente für die Aufgabenstellung „Rundenzeiten“ zu fahren beinhalten. Beim Acceleration-Mode soll die Anzeige grafische Elemente zur Unterstützung des besten Schaltmoments beinhalten, damit der Fahrer eine gute Beschleunigung erreicht. Der Training-Mode ist für Test- und Trainingsfahrten gedacht. Hier werden Anzeige-Elemente zum frühzeitigen Warnen von Problemen im Auto (zu hohe Temperatur) oder Informationen für ein Fahrertraining, wie Rundenzeit oder Beschleunigung, angezeigt.

Mehr zu den Driver-Modes, den grafischen Anzeige-Elementen und Empfehlungen zur Gestaltung im Kapitel 5.

4.6 Konfigurationsmöglichkeiten über XML

XML[G] ist ein strukturiertes Textdokument. Es gibt viele Syntax-Analyse- und Verarbeitungs-Werkzeuge, sogenannte Parser, für XML-Dokumente für etliche Programmiersprachen, wie Java, C#, C++, Visual Basic, JavaScript und noch viele mehr. Unter anderem bietet QT (siehe Abb. 2.13) drei verschiedene XML-Parser.

Der Vorteil von XML-Dokumenten ist die einfache Textformatierung, die es Lesbar für Menschen macht und angemessen verständlich ist. XML-Dokumente sind leicht zu erstellen und

können hierarchische komplexe Beziehungen darstellen.
XML-Dokumente sind daher ideal für den Einsatz als Konfigurationsdateien geeignet. Sie lassen sich einerseits von Menschen gut lesen und bearbeiten, und anderseits von Maschinen leicht interpretieren.

Wie es aus den vorherigen zwei Kapiteln hervorgeht, soll das System - hier ist das Programm gemeint - von außen konfigurierbar sein. Ein XML-Dokument soll als ein Profil dienen, dass die verschiedensten Einstellungen beinhaltet. Der Grobe Aufbau der XML ist in drei Teile unterteilt:

- allgemeine Einstellungen
- Konfiguration der Sensordaten
- Einstellung der „Driver-Modes“

4.6.1 Allgemeine Einstellungen

In den allgemeinen Einstellungen sind Informationen, die das System benötigt, um optimal und performant laufen zu können. Alle Informationen die nicht in der XML verhanden sind, werden auf ein Standardwert gesetzt. Die Tabelle 4.3 listet die Eigenschaften und dessen Untereigenschaften auf.

4.6.2 Konfiguration der Sensordaten

Die Sensordaten sind in CAN-Nachrichten über die CAN-ID und den ByteOffset (siehe Kapitel 4.4) zu identifizieren. Die Sensordaten sind von Sensor zu Sensor unterschiedlich groß. Aber nicht nur die Größe der Sensordaten sondern auch der Wertebereich und die Auflösung der Werte ist von Sensor zu Sensor verschieden.

Damit die Daten der Sensoren effektiv und individuell bearbeitet werden, soll für jeden Sensor eine Instanz im Programm existieren, die weiss wie mit den Daten umzugehen ist. Die Klasse die diese Instanzen beschreibt und die Routinen für das Bearbeiten der Daten beinhaltet heißt „CAN-Object“. Diese sogenannten CAN-Object’s sind durch die Attribute aus Tabelle 4.4 definiert.

Die Einstellung eines CAN-Object’s, welches die Daten vom Sensor für die Wassertemperatur bezieht, bearbeitet und an die jeweiligen Anzeige-Elemente auf dem Display weitergibt könnte folgendermaßen aussehen:

Eigenschaft	Wert	Beschreibung
ColorTheme	Standard / Gray-Theme / Contrast	vordefinierte Farbpaletten für die Anzeige
CAN	*CAN-ID, ID-Mask*	Filtereinstellungen für den CAN-Socket
CAN-ID	2 Byte Hexadezimal (Beispiel 0x063A)	hier wird die CAN-ID festgelegt, auf der die ID-Mask angewendet wird
ID-Mask	2 Byte Hexadezimal (Beispiel 0x00FF)	die Maske bestimmt, welche ID Bereiche ein bzw. ausgeschlossen sind
PageSliding	0 / >0	Der Wert 0 stellt den Effekt für das bewegte Seitenwechseln aus alle Werte über 0 aktivieren diesen Effekt(siehe Kapitel 5)
MenuHidingTime	X > 0	alle positiven Werte sind erlaubt. Gibt in Milisekunden an, wie lange die Soft-Key-Anzeige auf dem Display bleibt, bis es verschwindet
GearChangingMoment	Gear1 bis Gear4	hier werden die optimalen Schaltmomente eingetragen
Gear1	RPM > 0	hier wird die Drehzahl eingetragen, die den optimalen Schaltmoment vom 1 zum 2 Gang beträgt
Gear2	RPM > 0	hier wird die Drehzahl eingetragen, die den optimalen Schaltmoment vom 2 zum 3 Gang beträgt
Gear3	RPM > 0	hier wird die Drehzahl eingetragen, die den optimalen Schaltmoment vom 3 zum 4 Gang beträgt
Gear4	RPM > 0	hier wird die Drehzahl eingetragen, die den optimalen Schaltmoment vom 4 zum 5 Gang beträgt

Tabelle 4.3: Allgemeine Einstellungen der XML-Konfiguration

Eigenschaft	Wert	Beschreibung
Name	beliebig	legt den Namen des CAN-Object's fest, an dem man später identifiziert welcher Sensor vorliegt. Es sollte also ein aussagekräftiger Name gewählt werden
DataType	signedchar/ unsignedchar/ signedshort/ unsignedshort	legt fest, wie groß die Daten sind (char oder short) und ob das MSB als Vorzeichenmerkmal gesehen werden kann
CanId	2 Byte Hexadezimal (Beispiel 0x063A)	gibt die CAN-ID der Nachricht an, in der die Daten des jeweiligen Sensors sind
RangeMinimum	Ein Wert der in den Datentyp passt	kleinster Wert der auftreten darf. Ist der Wert kleiner als diese Grenze, wird er ignoriert
RangeMaximum	Ein Wert der in den Datentyp passt	größter Wert der auftreten darf. Ist der Wert größer als diese Grenze, wird er ignoriert
ByteOffset	1 bis 7	Offset, ab dem die Sensordaten in den CAN-Daten anfangen
DataResolution	> 0	gibt die Auflösung eines einzelnen Bits an
DataOffset	eine Zahl	der DataOffset wird auf dem Wert hinzuaddiert
IDP	ganzzahlige positive Zahl	IDP (internal decimal places) gibt die Nachkommastellen an, mit denen gerechnet werden soll. (Maßnahme um Float-Rechnungen wegen der Performanz zu umgehen)
Warning	LOWER, UPPER, WarningLevel	Informationen für das Warnsytem der Software (siehe 4.7)
LOWER	Zahl	wenn diese Grenze unterschritten wird, wird eine Warnroutine in der Software gestartet
UPPER	Zahl	wenn diese Grenze überschritten wird, wird eine Warnroutine in der Software gestartet
WarningLevel	DoNothing/ warning/ fatal	gibt an, in welchen Ausmaß die Warnung sein soll

Tabelle 4.4: CAN-Object Definition im XML-Dokument

```
<CanConfig name="waterTemp">
   <DataType>unsignedchar</DataType>
   <CanId>0x639</CanId>
   <RangeMinimum>0</RangeMinimum>
   <RangeMaximum>255</RangeMaximum>
   <ByteOffset>6</ByteOffset>
   <DataResolution>0.6</DataResolution>
   <DataOffset>-30</DataOffset>
   <IDP>2</IDP>
   <Warning>
      <LOWER>0</LOWER>
      <UPPER>200</UPPER>
      <WarningLevel>warning</WarningLevel>
   </Warning>
</CanConfig>
```

Zu erkennen ist, dass die Daten in einem vorzeichenlosen Byte an der Stelle 6 der CAN-Nachricht mit der CAN-ID 0x639 zu finden sind. Jedes Bit hat eine Auflösung von 0,6, der Sensor beginnt seine Messung bei -30 und kein Wert wird ignoriert, weil die Range von 0 bis 255 geht und damit alle Werte des vorzeichenlosen Bytes abdeckt. Die Daten vom Sensor sind folgendermaßen definiert: der kleinste DatenTyp-Wert multipliziert mit der Auflösung und dann addier mit dem DatenOffset (siehe Rechnung 4.7)

$$0 * 0,6 + (-30) = -30 \tag{4.7}$$

und der größte Wert wird analog gerechnet (siehe Rechnung 4.8)

$$255 * 0,6 + (-30) = 123 \tag{4.8}$$

Daraus ergibt sich, dass der Sensor für die Wassertemperatur von -30 bis +123° C misst. Das die Einheit Celsius oder Fahrenheit ist, ist hier noch nicht ersichtlich. Diese Information wird an die jeweiligen Anzeige-Elemente weiter gegeben. Dazu weiter unten mehr.

4.6.3 Einstellung der Driver-Modes

Die Einstellungen der Driver-Modes beziehen sich auf die Anzeige und darauf, woher die Daten bezogen werden.

Ein Driver-Mode hat drei Pages (engl. Seiten), Standard, Option1 und Option2. Jede Page hat hat ein Layout, dass angibt wieviele Widgets^G (Anzeige-Elemente) auf einer Page untergebracht werden können. Jedes Layout bestimmt die Position und große der Widgets. Mehr dazu in Kapitel 5. Jedes Widget wird mit den in der Tabelle 4.5 stehenden Attributen definiert.

Eigenschaft	Wert	Beschreibung
Name	beliebig	gibt den Bezug zu den vordefinierten Widgets. Jedes Widget hat einen vordefinierten Namen
CanInfo	beliebig	gibt den Bezug zu den CAN-Object's, von welchem die Daten erhalten werden
Optional	Name, CanInfo, ...	gibt den Bezug zu den vordefinierten Widgets. Optional ist ein Widget, das in dem übergeordneten Widget positioniert ist (ein Widget im Widget).
RangeMinLevel	Zahl	gibt den niedrigsten Wert an. Ist ein neuer Wert niedriger als RangeMinLevel, wird der aktuelle Wert des Widgets auf RangeMinLevel gesetzt
RangeMaxLevel	Zahl	gibt den höchsten Wert an. Ist ein neuer Wert höher als RangeMaxLevel, wird der aktuelle Wert des Widgets auf RangeMaxLevel gesetzt
Warning	LOWER, UPPER, WarningLevel	Informationen für das Warnsytem des Widgets (siehe 5)
LOWER	Zahl	das Widget definiert die Anzeige, falls diese Grenze unterschritten wird
UPPER	Zahl	das Widget definiert die Anzeige, falls diese Grenze überschritten wird
WarningLevel	DoNothing/ warning/ fatal	gibt an, in welchen Ausmaß die Warnung sein soll
Unit	beliebig	hällt Informationen der Beschriftung wie beispielsweise die Einheit

Tabelle 4.5: Einstellungen der einzelnen Widgets

4.6.4 Beispiel einer XML Datei

Exemplarisch soll an dieser Stelle eine Beispiel-XML mit nur einer CAN-Object-Konfiguration und nur einem Driver-Mode. Man kann sich schnell vorstellen welche Ausmaße das XML-Dokument annimmt, wenn 7 oder mehr CAN-Object-Konfigurationen und drei Driver-Modes in einem XML-Dokument enthalten sind.

Beispiel:

```
<root>
   <ColorTheme>Standard</ColorTheme>
   <CAN>
      <CAN-ID></CAN-ID>
      <ID-Mask></ID-Mask>
   </CAN>
   <PageSliding>1</PageSliding>
   <MenuHidingTime>2000</MenuHidingTime>
   <GearChangingMoment>
      <Gear1>12000</Gear1>
      <Gear2>12000</Gear2>
      <Gear3>12000</Gear3>
      <Gear4>12000</Gear4>
   </GearChangingMoment>
   <CanConfig name="gear">
      <DataType>unsignedchar</DataType>
      <CanId>0x63A</CanId>
      <RangeMinimum>0</RangeMinimum>
      <RangeMaximum>5</RangeMaximum>
      <ByteOffset>4</ByteOffset>
      <DataResolution>1</DataResolution>
      <DataOffset>0</DataOffset>
      <IDP>0</IDP>
      <Warning>
         <LOWER>0</LOWER>
         <UPPER>6</UPPER>
         <WarningLevel>DoNothing</WarningLevel>
      </Warning>
   </CanConfig>
```

```
<DrivingMode name="Acceleration" >
   <Page name="Standard">
      <Layout name="Layout1">
         <Position1>
            <Widget name="rpmindicator1">
               <CanInfo>rpm</CanInfo>
               <Optional name="gear_digital">
                     <CanInfo>gear</CanInfo>
                     <RangeMinLevel></RangeMinLevel>
                     <RangeMaxLevel></RangeMaxLevel>
               </Optional>
               <Unit>rpm</Unit>
            </Widget>
         </Position1>
         <Position2>
            <Widget name="speedindicator_digital">
               <CanInfo>speed</CanInfo>
               <RangeMinLevel></RangeMinLevel>
               <RangeMaxLevel></RangeMaxLevel>
            </Widget>
         </Position2>
      </Layout>
   </Page>
   <Page name="Option1">
      <Layout name="Layout2">
         <Position1>
            <Widget name="oil_barometer">
               <CanInfo>oilPressure</CanInfo>
               <Warning>
                   <LOWER>0</LOWER>
                   <UPPER>0</UPPER>
                   <WarningLevel>warning</WarningLevel>
               </Warning>
               <RangeMinLevel></RangeMinLevel>
               <RangeMaxLevel></RangeMaxLevel>
            </Widget>
         </Position1>
         <Position2>
            <Widget name="battery">
               <CanInfo>battery</CanInfo>
```

```
                <Warning>
                    <LOWER>0</LOWER>
                    <UPPER>0</UPPER>
                    <WarningLevel>warning</WarningLevel>
                </Warning>
                <RangeMinLevel></RangeMinLevel>
                <RangeMaxLevel></RangeMaxLevel>
            </Widget>
        </Position2>
    </Layout>
</Page>
<Page name="Option2">
    <Layout name="Layout2">
        <Position1>
            <Widget name="oil_temperature">
                <CanInfo>oilTemp</CanInfo>
                <RangeMinLevel></RangeMinLevel>
                <RangeMaxLevel></RangeMaxLevel>
            </Widget>
        </Position1>
        <Position2>
            <Widget name="water_temperature">
                <CanInfo>waterTemp</CanInfo>
                <Warning>
                    <LOWER>0</LOWER>
                    <UPPER>0</UPPER>
                    <WarningLevel>warning</WarningLevel>
                </Warning>
                <RangeMinLevel></RangeMinLevel>
                <RangeMaxLevel></RangeMaxLevel>
            </Widget>
        </Position2>
    </Layout>
</Page>
</DrivingMode>
</root>
```

Die Groß- und Kleinschreibung der Tag-Namen ist wichtig. Falls der Benutzer aber sich unsicher ist, hat er die Möglichkeit alle Tag-Namen klein zu schreibe.

4.7 Warnsystem

Das Warnsystem soll dem Fahrer mitteilen, dass etwas mit dem Wagen nicht stimmt. Dabei solles verschiedene Stufen der Fahlermeldungen geben.

- DoNothing - bei einen Fehlerfall soll nichts geschehen
- warning - in dieser Stufe soll der Fahrer gewarnt werden, dass etwas nicht stimmt, aber nicht an seiner Weiterfahrt gehindert werden. Er soll noch alle Informationen, die er ohne Fehlerfall erhällt, weiterhin erhalten
- fatal - in dieser Stufe soll der Fahrer mit allen erdenklichen Mitteln gewarnt werden, das ein schwerwiegender Fehler aufgetreten ist und der Wagen womöglich ausgeschaltet werden soll.

4.7.1 warning

Die Warnstufe *warning* ist für Fehlerfälle geeignet, bei denen der Fehler keine großen Auswirkungen auf den Fahrer oder den Rennwagen hat. Zum anderen kann man diese Warnstufe aber auch für schwerwiegende Fehlerfälle anwenden. Nämlich bei Fehlern die unter Umständen Schaden am Rennwagen ausüben würden, aber in einem bestimmmten Szenario in kauf genommen werden.
Beispiel: Der Fahrer fährt in einem echten Endurance und der Motor läuft viel zu heiß und droht Schaden zu nehmen. Der Fahrer und das Team wollen aber unbedingt des Rennen gewinnen und nehmen damit einen Schaden am Rennwagen in Kauf. Steht dabei die Sicherheit des Fahrers in Frage, darf das System für den jeweiligen Fehlerfall nicht mit dieser Warnstufe deklariert werden.

Um den Fahrer bei der Fahrt nicht zu beeinträchtigen, soll bei der Warnstufe *warning* auf den Fehler lediglich hingewiesen werden. Dabei sollen weiterhin alle Informationen dem Fahrer zur verfügung stehen. Damit das gelingt, wird Platz auf dem Display für Warnungen eingeplant. Dazu mehr im nächsten Kapitel.

Um die Aufmerksamkeit des Fahrers auf das Display zu lenken, damit er die Warnung warnimmt, soll die Warnung blinken[G] (DIN EN ISO 15008, 2003).

4.7.2 fatal

Bei Sicherheitskritischen Aspekten oder zur Prävention von möglichen Schäden am Rennwagen, sollte eine Warnung als *fatal* eingestuft werden. Der Fahrer soll bei dieser Warnstufe vom Weiterfahren gehindert werden ohne ihn aber in seinen sensorischen Eigenschaften zu behindern. Der Fahrer soll noch fahrtauglich bleiben und den Rennwagen so gut und schnell abstellen. Dabei sollen jegliche visuellen Informationen entfallen und nur die Warnung aufgaben- und fehlerangemessen dem Fahrer dargestellt werden.

In einem Fehlerfall der Warnstufe *fatal* sollen die LED's für das Anzeigen der Schaltmomente in der Farbe Rot blinken. Zudem soll das ganze Display mit der Warnung ausgefüllt werden und zwei Dinge anzeigen:

1. Fehler - gibt an, was für ein Fehler auftrat. („Motor zu heiß“, „Öldruck zu schwach“)
2. Meldung - gibt an, was getan werden soll („Bitte schalten Sie den Motor aus“)

Mehr über die Darstellung von Warnungen im nächsten Kapitel 5.

5 UI Prototyping

Beim User Interface Prototyping geht es um die Modellierung des späteren Produkts. Dieses Kapitel behandelt die grafischen Elemente der Software einerseits, sowie die Gestaltung des Interfaces andererseits.
Die grafische Anzeige und das Interface für die Steuerung sollen perfekt zu eineander passen, sozusagen verschmelzen. Der Aufwand des Benutzers, seine benötigten Informationen zu erschließen, soll gering gehalten werden. Zudem soll der Benutzer Freude an der Bedienung der Software haben, damit das Ziel der **Zufriedenstellung** erreicht wird.

„Neben den Software-Qualitäten der Funktion und der Benutzbarkeit kommt noch die Kategorie der hedonischen Qualität hinzu, eine Bewertung des Genusses und der Freude bei der Benutzung der Software“(siehe Seite 236 Dahm, 2006). Die Ästhetik der grafischen Elemente soll dabei helfen den „Joy of Use“, also die Freude am Benutzen, zu steigern.

Eine zusätzliche Anforderung an das Aussehen der Anzeige auf dem Display ist die „corporate identity“. Das HAWKS Racing Team soll damit, wo auch immer der Rennwagen steht, identifiziert werden. Die Farben des HAWKS Racing Team's sind Blau und Weiss. Die Farben sollen, wie das HAWKS-Logo in das Menü mit eingebunden werden.

5.1 Das Interface

Das Interface sollte schnell verständlich (*Selbstbeschreibungsfähigkeit*) und leicht zu erlernen sein (*Lernförderlichkeit*) (siehe Kapitel 2). Drei Taster am Lenkrad sollen das Interface ausmachen. Zwei Taster sind zum Selektieren von Objekten oder Blättern von Seiten zuständig. Der dritte Taster bestätigt die Eingabe oder macht einen Menüwechsel. Die Taster sollen keine weiteren Funktionen übernehmen. Somit ist eine Konsistenz in der Funktion der Tasten gegeben

Eins der wichtigen Aspekte der Usability ist die Sicherheit. An dieser Stelle ist die Sicherheit des Fahrers vor sich selbst gemeint. Die Positionierung der Taster soll verhindern, dass der Fahrer während der Fahrt die Menüsteuerung des Displays benutzt. Daher sollen die Taster weit innen am Lenkrad angebracht werden.

Die Anordnung der Knöpfe spielt eine große Rolle bei der Positionierung. Mit der Anordnung der Knöpfe kann man dem Fahrer andeuten, welche Funktion die Knöpfe haben oder welche Knöpfe ähnliche Funktionen besitzen.

Abbildung 5.1: Anordnung der Knöpfe am Lenkrad

In der Abbildung 5.1 sind die Taster fürs Selektieren und das Wechseln der Pages an der Seite des Displays und der Taster für die Eingabe und dem Menüwechsel unter dem Display angebracht. Die Taster an den seitlichen Rändern des Displays sollen für das Wechseln der Pages zuständig sein und somit eine Assoziation zu einem Buch, dass man durchblättert, herstellen. Leider ist diese Lösung suboptimal, weil die Wege der Hand bei der Menüführung sehr lang sind. Zudem brauchen die Taster, um als Softkey fungieren zu können, Labels^G (engl. Beschriftung) am Rand des Displays, damit ihre Funktion klar wird. Das Display ist aber sehr klein und man kann keine Labels an den Seiten unterbringen,

ohne viel Splatz dabei zu verlieren.

Abbildung 5.2: Anordnung der Knöpfe am Lenkrad unter dem Display auf einer Linie

Die Abblidung 5.2 zeigt eine bessere Anordnung als die Vorige. Bei der Bedienung sind kleinere Wege möglich und die Softkey-Labels können am unteren Rand des Displays, wie bei heutigen Handy's, plaziert werden. Dabei soll der Effekt des Wiedererkennens der Menüsteuerungen von Handys genutzt werden, um eine Art Analogie und dadurch *Konsistenz*(siehe Seite 7 DIN EN ISO 9241-12, 1998) zu schaffen.

Damit der Unterschied in den Funktionen der zwei Taster für die Auswahl und das Wechseln der Pages einerseits und den Taster für die Eingabe und dem Menüwechsel andererseits klar wird, werden die Knöpfe wie in Abbildung 5.3 angeordnet (siehe 5.6.1 Unterscheidung von Gruppen DIN EN ISO 9241-12, 1998). Die zwei Auswahl-Taster liegen waagerecht auf eine Linie und am rand des Displays, sodass sie gemeinsamkeiten in der anordnung haben und damit andeuten sollen, dass sie der gleichen Funktionsgruppe zugehören.
Man könnte die „Unterscheidung von Gruppen" auch mit der Form oder der Farbe der Knöpfe verstärken. Diese Methoden sollen aber nicht angewendet werden.

Ein weiterer Punkt ist die Beschaffenheit der Knöpfe. Der Druckpunkt^G des Knopfes sollte idealerweise tief sein, damit der Benutzer beim Drücken mit Handschuhen die Rückmeldung

Abbildung 5.3: Anordnung der Knöpfe am Lenkrad unter dem Display

spürt, gedrückt zu haben. Stattdessen kann auch ein Taster mit einem harten flachen Druckpunkt gewählt werden.

5.1.1 Softkey

Softkeys sind Taster, die ihre Funktion dynamisch wechseln können. Normalerweise sind Softkeys an den Zustand der Menüführung gekoppelt. Softkeys werden meist in mobilen oder eingebetteten Systemen, die wenig Platz für ein Interface bieten, eingesetzt. Ihre aktuelle Funktion wird in einem ihnen zugeteilten Bereich auf dem Display - meist nah am Rand und in der Nähe des Tasters - angezeigt.

Für alle drei Taster wird ein Bereich im unteren Rand des Displays zugewiesen. Bedingt durch den Platzmangel, bekommen die Label der beiden äußeren Taster weniger Platz auf dem Display, weil die Funktionen der Taster mittels Platz sparenden Pfeilen kodiert werden können. Dadurch erhällt man mehr Platz für das Label des mittleren Tasters.

Abbildung 5.4: Verschwinden der Softkey-Label-Leiste

Nach einer bestimmten Zeit x, die im XML-Dokument definiert ist, verschwinden die Labels in einer fließenden Bewegung in den unteren Rand des Dispays, damit mehr Platz für die Anzeige der Widgets verhanden ist (siehe Abb. 5.4). Die Bewegung soll dem Benutzer das Gefühl vermitteln, dass die Anzeige nicht einfach spurlos verschwindet, sondern immer präsent und abrufbar ist.
Sind die Labels der Softkeys inaktiv – also nicht zu sehen – können Sie durch drücken einer der drei Taster wieder sichtbar gemacht werden. Dabei wird keine Aktion der Taster durchgeführt, weil der Benutzer nicht sah welcher Funktion die Taster belegt waren. Erst wenn die Labels wieder aktiv sind, sind auch die Taster mit Funktionen belegt. Das soll vehindern, dass die Darstellung wechselt, obwohl der Benutzer es nicht gewollt hat.

Um einen Gulf of Execution^G des Benutzers zu verhindern, sind folgende Überlegungen umgesetzt worden:

- Es muss in jedem Zustand erkannbar sein, welche Aktionen möglich sind
- Der Zustand des Geräts muss stets erkennbar sein
- Das konzeptionelle Modell muss sich schnell erschließen lassen
- Der Benutzer muss den neuen Systemzustand mit seinen Zielen leicht vergleichen können

Diese Richtlinien resultieren aus den *sieben Handlungsschritten von Donald Norman* (vgl. Seite 99 Dahm, 2006).

Um zu erkennen auf welcher Page man sich gerade befindet, ist bei aktiver Softkey-Label-Leiste der Name der Seite am oberen Rand des Displays angezeigt. Wenn die Leiste

verschwindet, verschwindet auch die Anzeige am oberen Rand. Wenn die Softkey-Label-Leiste verschwunden ist, sieht man in der rechten unteren Ecke eine Anzeige mit der aktuellen Seitenzahl gefolgt von der Anzahl der Seiten.

Damit der Benutzer weiss, welche Aktionen in den jeweiligen Zustand möglich sind, gibt es die Softkey-Label-Leiste, die die aktuelle Funktionen der Taster anzeigt. Wenn eine Taste in einem Zustand keine Aktion bewirkt, wird auf dem Bereich des Tasters auf der Softkey-Label-Leiste nichts angezeigt.

Für die Lernförderlichkeit des Benutzers, leuchten die Labels bei Drücken der Taster kurz auf. Das ist eine Art der Rückmeldung des Systems die dem Benutzer helfen soll zu verstehen, dass die Labels zu den Tastern gehören.

5.2 Farbsystem

Die visuellen Informationen auf dem Display sollten bei unvorteilhaften Umgebungshelligkeiten - also sehr hellem Umgebungslicht - lesbar sein. Eine gute Farbkombination kann helfen, mit einem straken Kontrast die Informationen besser lesbar zu machen.
Farben können auch als Mittel der Kodierung dienen. „Sie sollten zusammen mit anderen Kodierverfahren redundant verwendet werden“(siehe Seite 15 DIN EN ISO 9241-12, 1998). Zusätzlich kann bei willkürlich verwendeter Farben, der Effekt enstehen, man habe einen „zugepackten“ oder „vollen“ Bildschirm (vgl. Seite 15 DIN EN ISO 9241-12, 1998).

Das Farbsystem der Software ist in Farbschemen organisiert. Ein vordefiniertes Farbschema aus vier Farben legt fest, in welchen Farben sich das GUI mit den Widgets präsentieren. Die vordefinierten Farbschemen sind in Hinblick auf einen starken Kontrast durch Farbkombinationen wohl durchdacht und sollen die Willkür des Benutzers in der freien Konfiguration durch das XML-Dokument unterbinden.

Ein Farbschema ist in folgenden vier Farbfunktionen unterteilt:

- Vordergrundfarbe
- Hintergrundfarbe
- Meldungsfarbe
- Warnfarbe

Hintergrundfarbe	**Farbe des Zeichens**						
	weiß	gelb	orange	rot	grün, cyan	blau, violett	schwarz
weiß	■	-	○	+	+	++	++
gelb	-	■	-	○	○	+	++
orange	○	-	■	-	-	○	+
rot	+	○	-	■	-	-	+
grün, cyan	+	○	-	-	■	-	+
blau, violett	++	+	○	-	-	■	-
schwarz	++	++	+	+	+	-	■
++ sehr gut + gut ○ akzeptabel bei signikanten Unterschieden in der Farbsättigung - nicht Empfehlenswert							

Tabelle 5.1: Zeichen/Hintergrund Farbkombinationen (siehe Tabelle 1 DIN EN ISO 15008, 2003)

Die Vordergrundfarbe sollte einen starken Kontrast zur Hintergrundfarbe haben. Das sind die Farben, die das GUI hauptsächlich anzeigt.
Die Meldungsfarbe ist für die Hervorhebung von Symbolen und alphanumerischen Zeichen zuständig. Diese Farbe wird auch für Warnungen benutzt.
Die Warnfarbe existiert nur für Warnungen, bei denen die Meldungsfarbe nicht ausreicht einen Kontext Hervorzuheben oder die Meldungsfarbe schon für einem anderen Informationskontext verwendet wird.

Nicht im Farbschema beinhaltet ist die Farbe Blau und spezielle Farbkodierungen der einzelnen Widgets. Die Farbe Blau ist in Kombination mit Weiss in den Labels der Softkeys, in dem Logo des HAWKS Racing Team's und in einigen Widgets vertreten. Das soll die „corporate identity" verstärken. Zu den Farbkodierungen der einzelnen Widgets im Kapitel 5.6 mehr.

Für eine ideale Farbkombination der Farbschemata wurde die Tabelle 5.1 zur Hilfe gezogen.

5.2.1 Farbschema Standard

„Standard" ist das empfohlende Farbschema. Falls kein Farbschema in XML-Dokument angegeben wurde, wird „Standard" standardmäßig genommen.

Farbfunktion	Farbe
Vordergrundfarbe	weiss
Hintergrundfarbe	schwarz
Meldefarbe	helles gelb
Warnfarbe	rot

Tabelle 5.2: Definition des Farbschemas „Standard“

Farbfunktion	Farbe
Vordergrundfarbe	schwarz
Hintergrundfarbe	weiss
Meldefarbe	dunkles gelb
Warnfarbe	rot

Tabelle 5.3: Definition des Farbschemas „Contrast“

5.2.2 Farbschema Contrast

„Contrast“ ist ein Farbschema, bei dem der Kontrast zwischen Vorder- und Hintergrundfarbe sehr groß ist.

5.2.3 Farbschema GrayTheme

Mit dem Farbschema „GrayTheme“ wird versucht, die in Kapitel 7.5.10 der Norm DIN EN ISO 9241-12 (1998) als gut deklarierte Hintergrundfarbe Hell-Grau, zu verwenden.

Farbfunktion	Farbe
Vordergrundfarbe	schwarz
Hintergrundfarbe	helles grau
Meldefarbe	dunkles gelb
Warnfarbe	rot

Tabelle 5.4: Definition des Farbschemas „GrayTheme“

5.3 Layout

Wie bei den Farbschemata, gibt es drei verdefinierte Layouts. Sie sollen die Größe, die Position und die Darstellung der Widgets bestimmen. Jedes Layout ist in Bereichen[G] unterteilt. Jedes Layout hat einen Meldungsbereich, in denen Statusmeldungen auftreten können. Bisher sind die einzigen Statusmeldungen Warnungen von Fehlern der Stufe „warning“. Man kann zukünftig dort auch andere Meldungen unterbringen.

Die anderen Bereiche definieren die Position1 bis n für die Widgets. Die Darstellung der Widgets - damit sind das Widget und die dazugehörigen Labels gemeint - ist für jede Position der Layouts festgelegt. Das heißt für den Programmierer, der ein neues Widget für das GUI entwickelt, dass das neue Widget für alle Positionen aller Layouts die Darstellung selbst definieren muss. Diese Maßnahme der vordefinierten Darstellungen ist notwendig, damit das Layout einerseits konsistent bleibt und andererseits den Benutzern helfen, leicht eine Page zu konfigurieren, ohne Kenntnisse der Darstellung von grafischen Anzeigen zu besitzen.

Alle Layouts folgen dem „Gesetz der Nähe“(vgl. Seite 10 DIN EN ISO 9241-12 (1998)). Das bedeutet, dass die grafischen Elemente durch ihre Nähe zu anderen Elementen als zusammengehörig definiert werden.

5.3.1 Layout1

Das Layout1 ist das von mir empfohlene Layout für die Standard/Haupt-Seite. Es zeigt zwei Informationen in verschiedenen größen an. Die wichtigere Information wird selbstverständlich an die Position für das größere Widget gesetzt. Leider bietet die Größe des Displays kein Platz für mehr Widgets, die groß genug und damit für den Fahrer schnell lesbar sind.
Das gelbe Warndreieck soll zeigen, wo die Meldebereiche der jeweiligen Layouts sind.

5.3.2 Layout2

Bei dem Layout2 ist für die Optionen-Seiten gedacht. Das Layout zeigt soweit die Widgets es zulassen das Widget mit einem Symbol für dafür und dessen Beschriftung. Beide Positionen sind gleich groß.

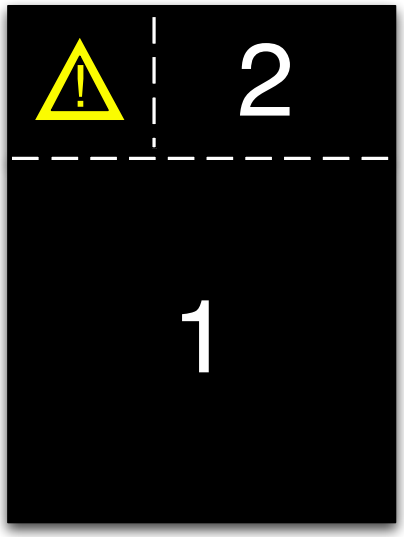

Abbildung 5.5: Layout1 mit der Position1 und 2

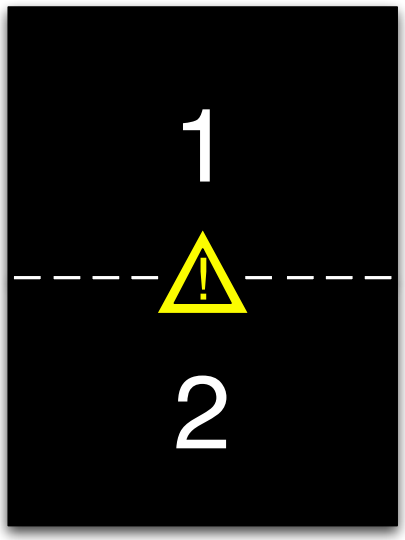

Abbildung 5.6: Layout2 mit der Position1 und 2

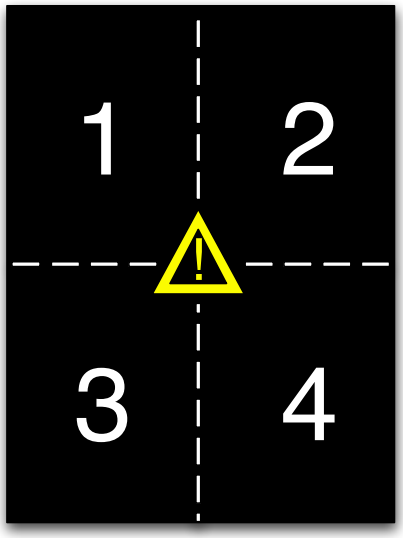

Abbildung 5.7: Layout2 mit der Position1 bis 4

5.3.3 Layout3

Das dritte Layout hat vier Bereiche. Es ist für Test- und Trainings-Fahrten konzipiert. Die Bereiche sind viel zu klein, damit der Fahrer sie während der Fahrt schnell ablesen kann.

5.4 Symbole

Bildschirmsymbole - auch Icons genannt - repräsentieren ein Objekt, eine Funktion oder eine Aktion. Sie helfen den Benutzer, falls er das Symbol verteht und interpretieren kann, schnell den Zusammenhang zu verstehen.
Bei Automobilen gibt es schon seit Jahren standarisierte und etablierte Symbole für Bedienteile, Anzeige- sowie Warngeräte die in der Norm 2575 aufgeführt werden ISO 2575 (1994). Von Jahr zu Jahr kommen immer mehr dazu.

Die Symbole für

- Batterieladezustand
- Öldruck
- Öltemperatur

- Temperatur der Motorkühlflüssigkeit

sind aus der Norm 2575 nachgezeichnet und für die Anzeige aufbereitet worden.

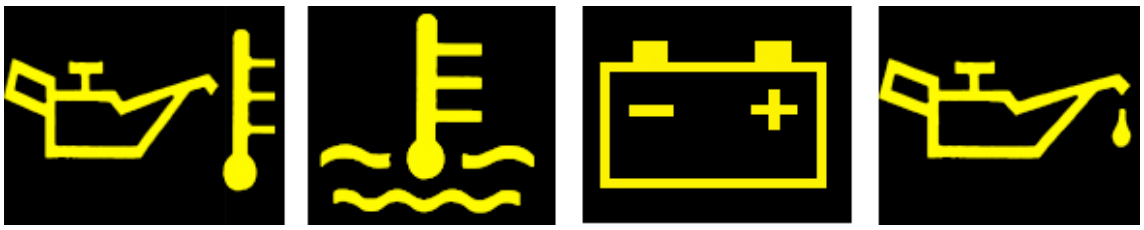

Abbildung 5.8: Symbole für Öltemperatur(links), Wassertemperatur(mitte-links), Batterieladezustand(mitte-rechts) und Öldruck(rechts)

Die Abbildung 5.8 zeigt die Symbole, die genutzt werden.

5.5 Listen

Für die Auswahl die Profile und den Driver-Modes werden Listen^G verwendet. Sie sind intuitiv in der Handhabung und listen alle Wahlmöglichkeiten auf. Um die hedonische Qualität zu steigern, wurde der Balken für die Auswahl im Aussehen der Softkey-Label-Leiste angepasst.

Falls mehr Elemente aufgeführt werden müssen als auf einer Seite passt, zeigt ein kleiner Pfeil nach unten, dass die Liste noch länger ist. Ist der Auswahlbalken ganz unten am Bildschirm und man drückt weiter nach unten, bewegt sich die Liste statt des Balkens. Man kann dann so lange nach unten drücken, bis der Pfeil nach unten in der Liste verschwindet. Verschiebt sich die Liste einmal nach oben, erscheint oben rechts in der Liste ein Pfeil der nach oben zeigt, um zu Zeigen, dass Listenelemente über den oberen Rand hinaus verschoben wurden.

Die Abbildung 5.9 zeigt eine Liste die mehr Elemente hat, als auf dem Bildschirm passen und den Pfeil nach unten.

Abbildung 5.9: Listenmenü

5.6 Widgets

Die Widgets sind in dem GUI die grafischen Anzeige-Elemente. Der Name Widget soll andeuten, dass die einzelnen Anzeige-Elemente austauschbar sind.
Widgets sind einzelne kleine Programme die Informationen bieten und teilweise auch Eingabemöglichkeiten haben.

Gemäß den Anforderungen aus dem Kapitel 2 wurden Anzeigen entwickelt, die verschiedene Informationen in einer Form darstellen, die es dem Fahrer einfach und schnell machen, die wesentlichen Informationen der Anzeigen zu entnehmen.

Der Fahrer bekommt die Information, die für ihn wichtig ist. Dem Fahrer sollte während der Fahrt nicht interessieren, welchen genauen Wert ein Sensor hat. Für das Fahren sind nur relativ einfache Informationen wichtig („bin ich schneller als zuvor?“, „ist der Batterieladezustand gut?“, „läuft der Motor zu heiß?“). Wenn der Fahrer Informationen in Zahlenform erhalten würde, müsste der Fahrer diese Information erst einmal interpretieren und bewerten. Die Wahlreaktion des Fahrers, wäre zusätzlich mit einer Zeit für die Kognition des Interpretierens belastet. Durch die Minimierung der Wahlmöglichkeiten soll die Reaktionszeit

auf die dargestellten Informationen klein gehalten werden.

Die folgenden Widgets sollen vorgestellt werden:

- Batterie
- Druck-Anzeige
- Drehzahl-Anzeige in analoger Zeigerform
- Drehzahl-Anzeige als Segment-Anzeige
- Temperatur-Anzeige
- Warnanzeige

5.6.1 Batterie

Die Batterie ist ein Widget, dass Informationen über den Füllstand der Autobatterie oder anderen Batterien im Rennwagen gibt. Die Batterie nutzt die Konventionen für Farbkodierung (vgl. Seite 15 DIN EN ISO 9241-12, 1998) eines Ampelsystems. Die Farbkodierung des Ampelsystems ist jedem Autofahrer bewusst. Die Farben stehen für

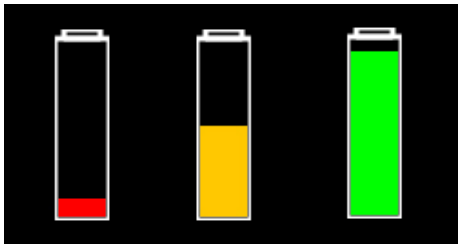

Abbildung 5.10: Batterie-Widget

- Rot : Vorsicht leer!
- Orange : Füllstand ist weniger als halb voll
- Grün : Der Füllstand der Batterie ist voll

Abbildung 5.11: Barometer-Widget

5.6.2 Barometer

Das Barometer Widget zeigt einen Druck an. Das Widget soll eine Analogie zu einem wahren Barometer sein; ein Zeiger der auf einer Skala zeigt auf der Werte sind. Das Widget wird bisher nur für den Öldruck-Sensor gebraucht.

5.6.3 Thermometer

Dieses Widget zeigt Temperaturen an. Die Form Des Widgets soll an ein Flüssigkeitsthermometer mit einer Pegelanzeige erinnern. Das Widget wird für die Anzeige der Öl- und Wasser-Temperatur genutzt.

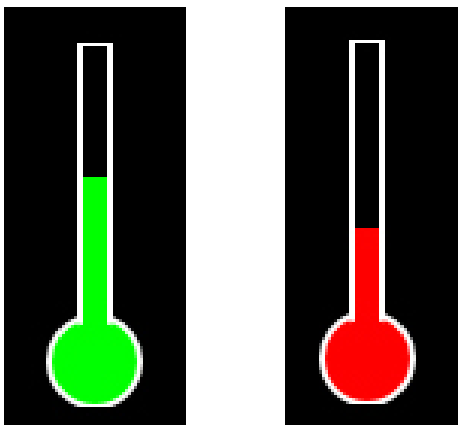

Abbildung 5.12: Thermometer-Widget

Das Widget besitzt zwei Wertegrenzen bei denen der Inhalt des Widgets, bei der Unterschreitung oder Überschreitung der jeweiligen Grenzen, rot kodiert wird. Liegt der Wert zwischen den Grenzen, ist der Inhalt des Widgets grün.

5.6.4 Drehzahl-Anzeige in analoger Zeigerform

Abbildung 5.13: Drehzahl-Widget (analog)

Das Drehzahl-Widget zeigt wie das Barometer-Widget eine analog Anzeige mit Zeiger und einer Skala. Es zeigt die Drehzahl in 1000ner Schritten an. Das Widget vereint die Informationen Drehzahl, Schaltmoment und Gang-Anzeige.
Die Skale ist in drei Abschnitte farblich unterteilt. Der erste Abschnitt ist in der Vordergrundfarbe gezeichnet und fordert den Fahrer zu keiner Reaktion auf.
Die Bereiche des zweiten und dritten Abschnittes sind durch die Einstellungen aus dem XML-Dokument festgelegt. Für jeden Gang können die Bereiche der Abschnitte in der Länge variieren. Der zweite Abschnitt ist in der Meldungsfarbe gezeichnet und soll den Fahrer andeuten, dass die optimale Drehzahl zum Schalten der Gänge erreicht wurde.
Der dritte und letzte Abschnitt ist in der Warnfarbe und zeigt, dass die Drehzahl schon weit über den besten Schaltmoment ist. Dieser Zustand wird durch ein Warndreieck in der Mitte des Widgets hervorgehoben.

Das Widget hat einen Bereich für ein optionales Widget das an der Stelle angezeigt wird. Dieser Bereich ist für die Gang-Anzeige optimiert worden.

Als zusätzliches Mittel, den besten Schaltmoment grafisch hervor zu heben, zeigt das Drehzahl-Widget von Außen nach Innen Ringe. Zwei Ringe sollen vorbereiten und der äußerste Ring den Fahrer zu einem Gangwechsel animieren (siehe Abb. 5.13).

Mein Vorschlag für den Driver-Mode „Endurance“ zeigen die folgenden Abbildungen 5.14, 5.15 und 5.16:

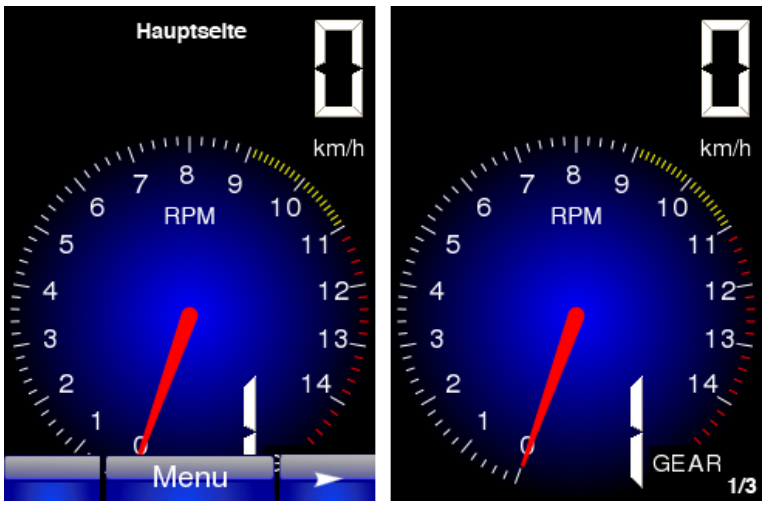

Abbildung 5.14: Layout: Hauptseite

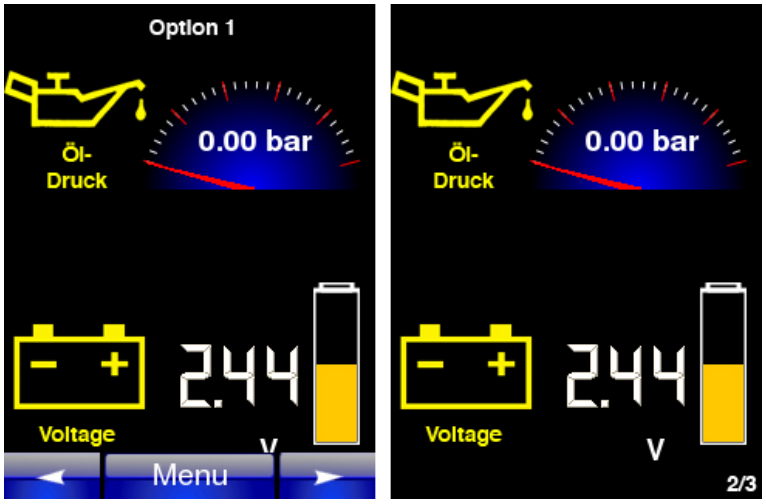

Abbildung 5.15: Layout: Options-Seite 1

5.6.5 Warnanzeige

Falls ein Fehler der Stufe „fatal“ auftritt, überdeckt das Widget für Warnungen mit einem großen Warndreieck in der jeweiligen Meldungsfarbe den Bildschirm.
Unter dem Warndreieck sollte dann der Fehler und eine Aufforderung etwas zu tun stehen (leider noch nicht in der Software implementiert).

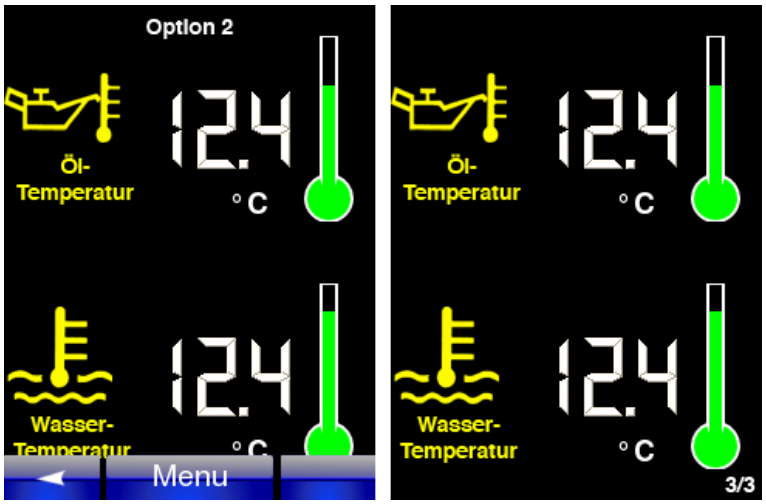

Abbildung 5.16: Layout: Options-Seite 2

6 Software Engineering

In diesem Kapitel wird das Design der Software und die Integration der Usability im Entwicklungsprozess behandelt. Mittels der Beschreibungssprache UML werden die Anforderungen an das System und die Problemstellungen dargestellt.

6.1 Ablaufbeschreibung

Die Entwicklung eines Systems kann in verschiedenen Etappen (Phasen) ablaufen. Es gibt verschiedenste Modelle, die Softwareentwicklungsprozesse zu unterteilen und durchzuführen. Hier einige Modelle zur phasenweise Entwicklung:

- Wasselfallmodell
- Spiralmodell
- V-Modell

Diese Modelle beschreiben ein iteratives Vorgehen in verschiedenen Phasen der Entwicklung. Für die Softwareenticklung ist ein strukturiertes Vorgehen von Vorteil, um die Planung und Kontrolle des Entwicklungsprozesses besser im Auge zu behalten.
Diese Arbeit ist in verschiedenen Phasen der Entwicklung unterteilt. Die Abbildung 6.1 zeigt den Ablauf der Entwicklung des Systems für diese Arbeit und in welchen Kapiteln die jeweiligen Phasen zu finden sind.
Die gelben Phasen beschreiben die Anforderungs-Analyse und den Grobentwurf. Sie stellen in der Usability das Vorgehen der Analysemethoden dar. In diesen Phasen muss der Analyst mit den späteren Benutzern eng zusammen arbeiten und die Anforderungen zusammentragen.
In den roten Phasen wird das System konkretisiert und modulweise implementiert und getestet. In diesen Phasen werden die Gestaltgesetze und Normen, unter berücksichtigung der allgemeinen und kontextspezifischen Anforderungen, herangezogen.
Abschließend wird in den blauen Phasen verifiziert und validiert, ob das System dem Design bzw. den Anforderungen entspricht.

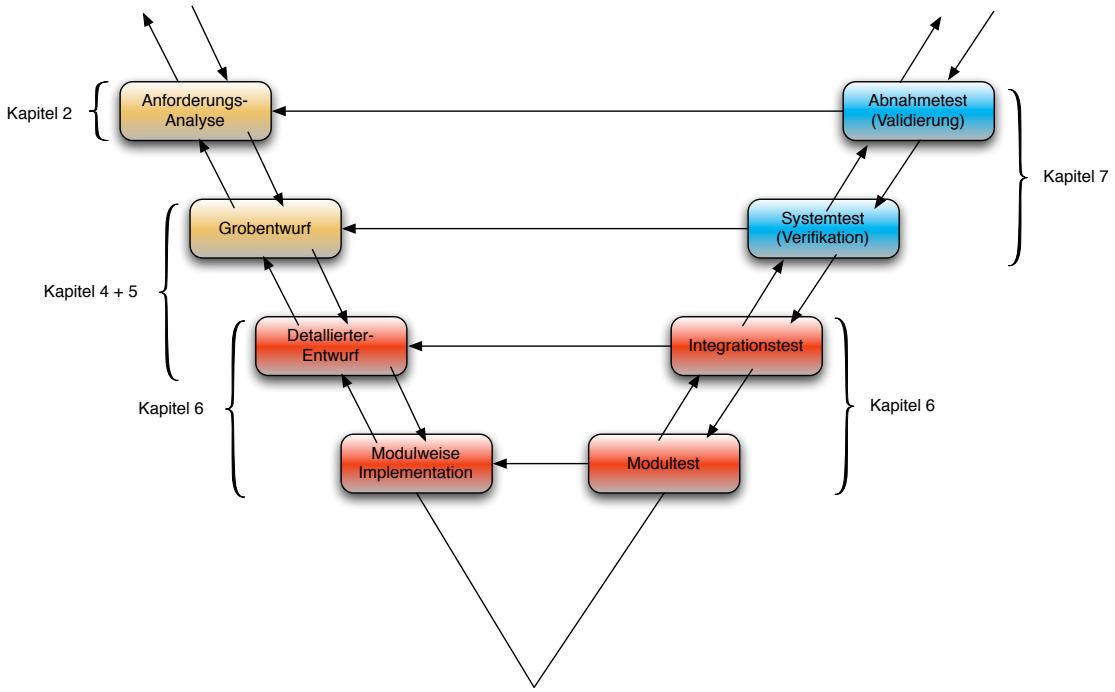

Abbildung 6.1: V-Modell mit der Zuweisung der jeweiligen Kapitel

Die Wahl fiel auf das V-Modell, weil es in einfachen iterativen Schritten zeigt, was getan wird und wie die Schritte in ihrer jeweiligen Detailstufe miteinander zusammenhängen. Dieses Vorgehensmodell ist einfach in die Planung zu integrieren. Ähnlich dem *Extreme Programming*[G], wurden die einzelnen roten Phasen aus Abbildung 6.1 für kleinere ausgearbeitete Funktionalitäten angewendet. Anhand sogenannter Storyboards[G] werden konkrete Funktionen nach und nach entwickelt und fügen sich dann zu einem größeren System.

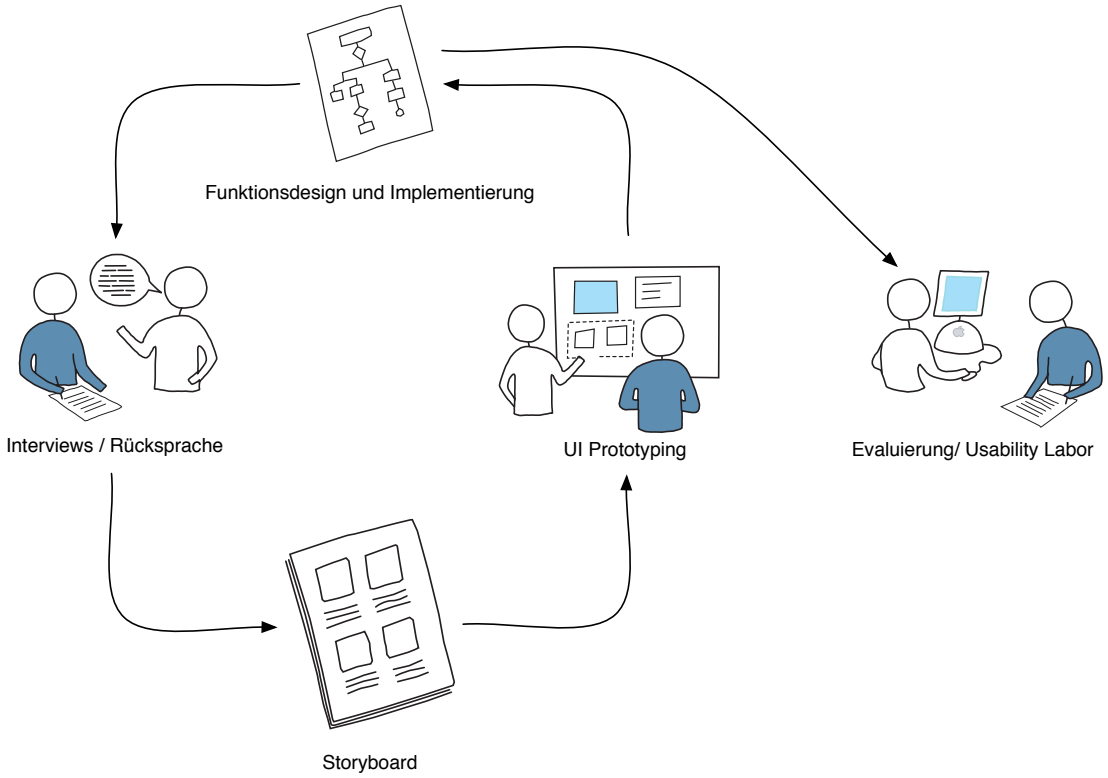

Abbildung 6.2: Integration der Usability in der Softwareentwicklung

Diese Herangehensweise soll im Entwicklungsprozess helfen, die Anforderungen des Kunden durch intensive Kommunikation besser zu erfassen und eine Rückkopplung des Kunden durch erneute Gespräche zu erhalten (siehe Abb. 6.2).

Ein anderer Vorteil des V-Modells ist das strukurierte Testen. Es wird inkrementell zu jeder Detailstufe getestet. Das heißt, dass Fehler, Irrtümer oder falsches Design den einzelnen Detailstufen zugeordnet werden können und man in die jeweiligen Stufen zurück gehen kann. Bei kleinen Projekten, wie diesem, kann man solch ein Modell gut einsetzen, weil das Design von der Größe nach relativ klein und leicht überschauber ist und spät erkannte Fehler im Design nicht so verheerende Außmaße annehmen, wie es bei großen Projekten der Fall sein kann.

6.2 Systemanforderungen und Softwaredesign

Die Software wurde den Anforderungen nach Wartbarkeit, Konfigurierbarkeit, Performanz, Stabilität und Fehlertoleranz entwickelt. Für die Konfigurierbarkeit der Software, wurde das XML-Dokument aus dem Kapitel 4 erstellt.

Die Systemumgebung ist eine relativ einfache Umgebung. Wie schon im vorigen Kapitel erörtert, hat das System ein drei Taster-Interface. Bisher ist das die einzige Mensch-Maschine-Schnittstelle zu der Software.

Die Aufgabenstellung der Software ist „einfach“ gesagt, nur das Anzeigen der Daten von den Sensoren. Der Benutzer drückt die Taster um die Darstellung auf dem Display seinem Vorhaben nach zu ändern.

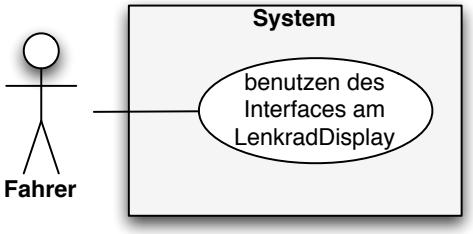

Abbildung 6.3: Interakton zwischen Mensch und Maschine

6.2.1 Menüsteuerung

Die Menüsteurung ist mit einem Zustandsautomaten realisiert. Die Abbildung 6.4 zeigt die Menüsteuerung.

Der Zustandsautomat steuert aber nicht nur wie in Abbildung 6.4 das Menü und das Lesen der XML, sondern auch die LED’s auf dem Lenkrad für den Schaltmoment. Der Zustandsautomat ist in der Klasse *MenuControl* implementiert, die die *CanObject*-Klasse und die *PageObject*-Klasse beinhaltet. Eine Instanz einer PageObject-Klasse stellt eine Seite mit Widgets dar. Für jeden geladenen Driver-Mode gibt es drei Instanzen von PageObject.

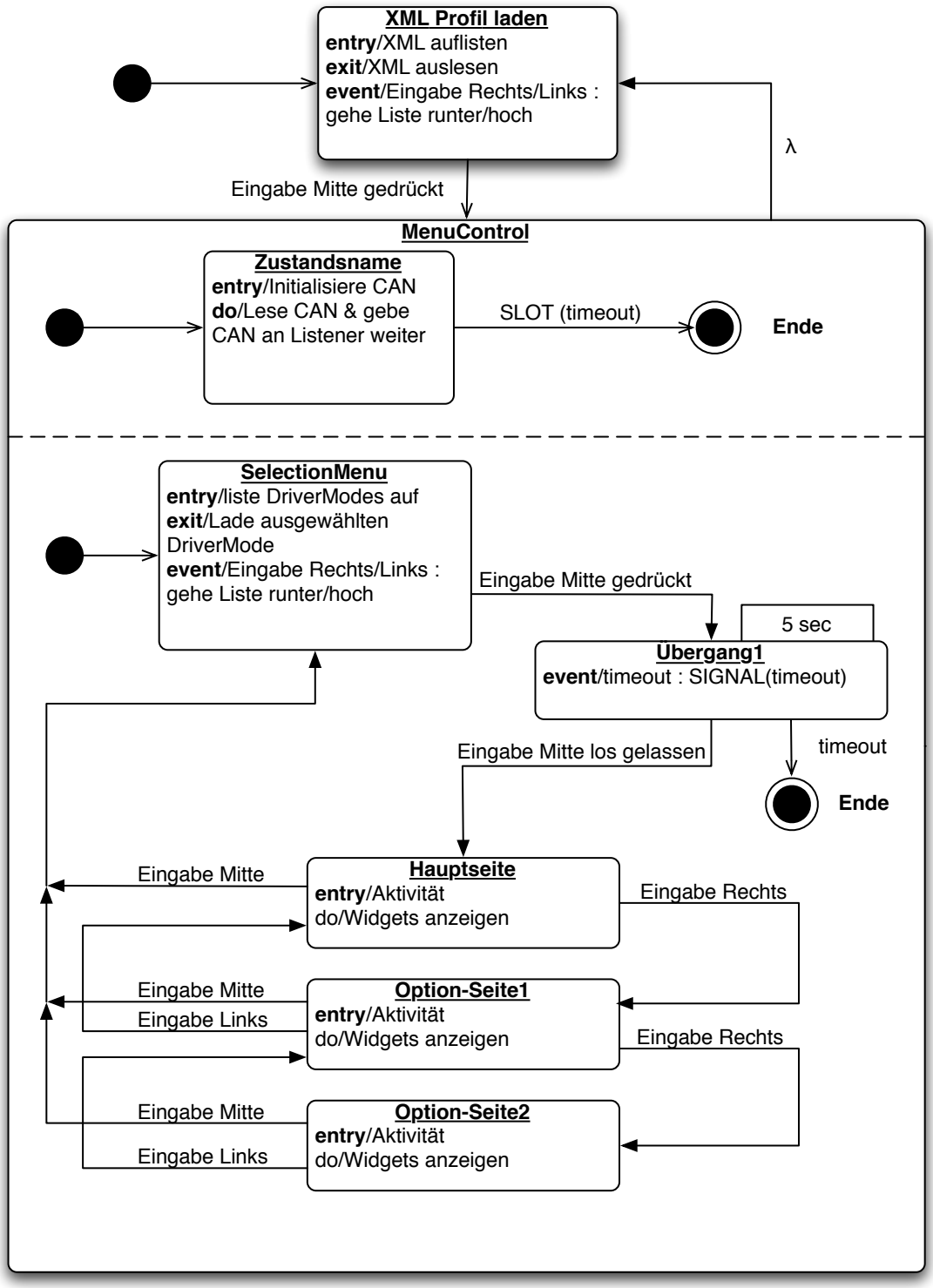

Abbildung 6.4: Menüsteuerung

6.2.2 Widgets

Die meisten Widgets die in dieser Software genutzt werden sind eigens entwickelte Widgets. Die Abbildung 6.5 zeigt die Widgets, die in dieser Software benutzt werden.

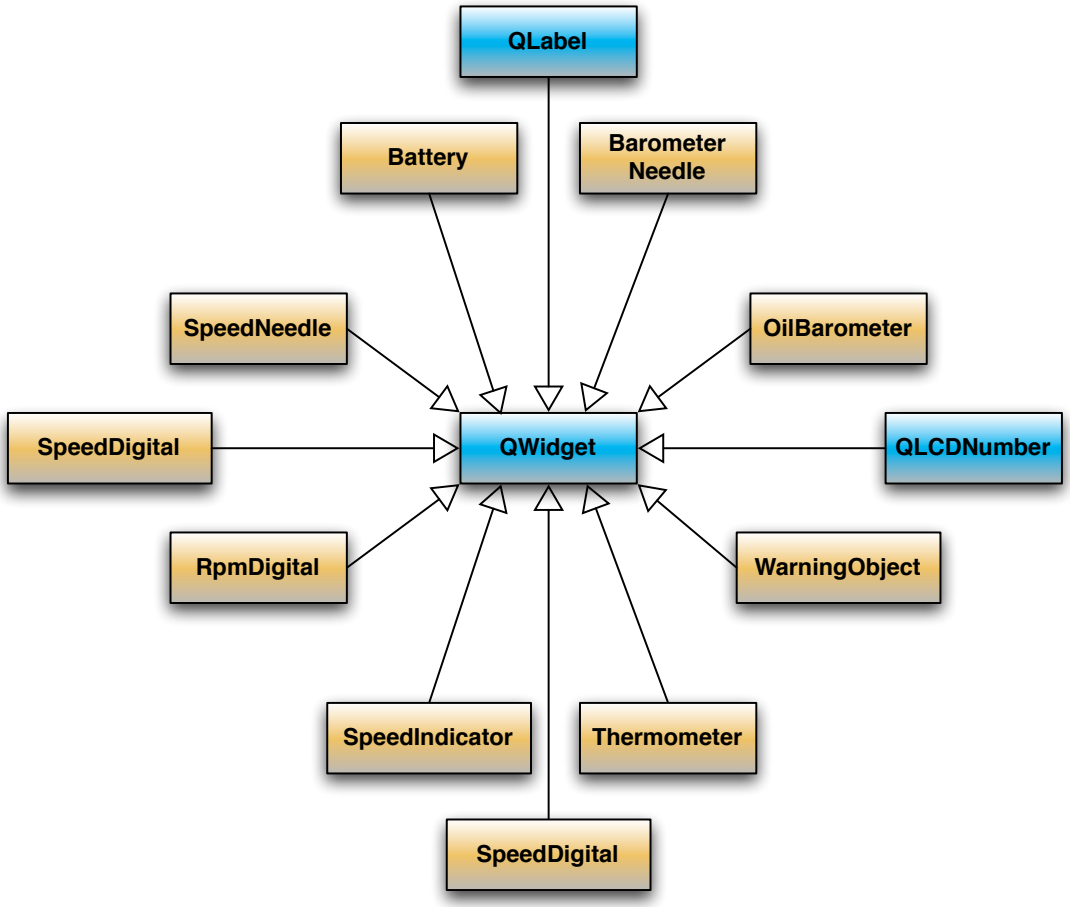

Abbildung 6.5: Vererbung der Klasse QWidget

Die in Orange dargestellten Widgets sind eigens erstellte Widgets die von der Klasse QWidget erben. QWidget ist eine Klasse im QT-Framework, die viele Methoden zum Zeichnen bereitstellt. Die blau dargestellten Widgets sind aus dem QT-Framework.

Alle eigens entwickelten Widgets besitzen eine Methode die ein Slot ist. Der Slot wird an ein Signal gebunden das auch eine Methode ist. Die Slots der Widgets werden an die Signals der CAN-Objekte gebunden, die die jeweiligen Informationen für die Widgets haben. Die

Slot Methode wird dann vom Framework aufgerufen, wenn sich bei dem angemeldeten CAN-Objekt der Zustand ändert.

Beipiel: Eine CAN-Objekt Instanz erhällt einen neuen Wert, wandelt den Wert für das Widget in einen verwertbaren Wert um und s

Für mehr Informationen über das QT-Framework siehe *http://doc.trolltech.com/*.

6.2.3 Lesen vom CAN-Bus

Das Lesen vom CAN-Bus ist über einen seperaten Thread realisiert. Dieser Thread bezieht die Daten über einen CAN-Socket. Die Klasse heißt *CanManager* und ist für das Lesen, Filtern und Verteilen der CAN-Nachrichten verantwortlich ist.

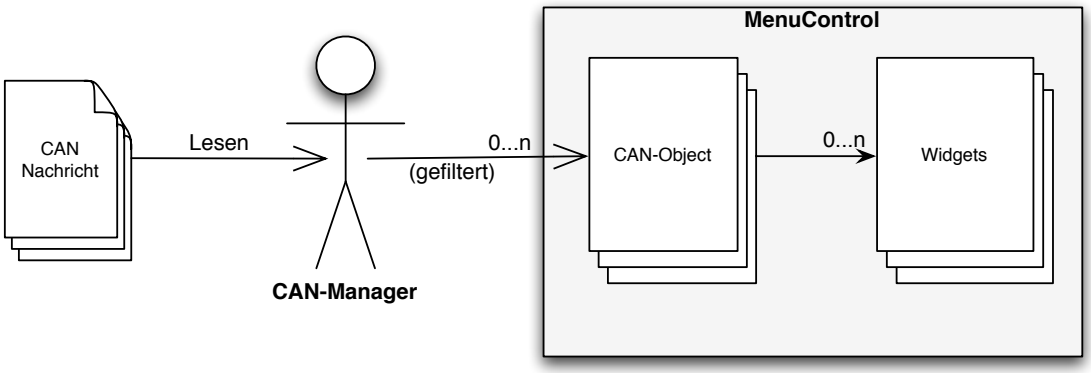

Abbildung 6.6: Der Weg der CAN-Nachrichten Verarbeitung

Der „CAN-Manager“ gibt die Nachrichten gefiltert nach CAN-ID’s an die Klasse MenuControl weiter. Die Klasse MenuControl besitzt eine Liste mit einer Indexstruktur, die mit einer Hashfunktion erzeugt wurde. Die Indices sind die CAN-ID’s und die Werte in der Liste sind wiederum Listen von CAN-Objekten. Anhand der Liste schaut die MenuControl Klasse nach den zuvor angemeldeten CAN-Objekten, die Interesse an den eintreffenden CAN-Nachrichten mit den jeweiligen CAN-ID haben.

Das heißt, dass mehrere CAN-Objekte an einer CAN-Nachricht interesse haben können und wiederum mehrere Widgets an den Wertänderungen der CAN-Objekte (siehe Abb. 6.7).

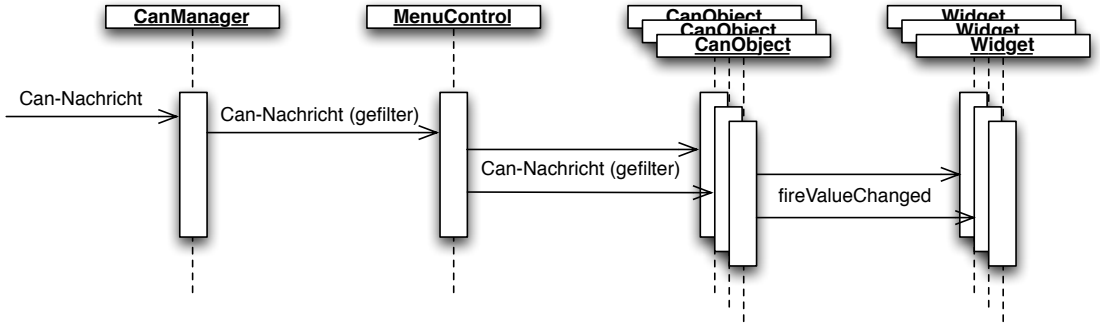

Abbildung 6.7: Kommunikation der einzelnen Komponenten über Signal & Slots

Der Anmeldevorgang der Widgets an den CAN-Objekten geschiet bei der Instanziierung der PageObjekte. Die Abbildung 6.8 zeigt in einem Sequenzdiagramm diesen Vorgang.

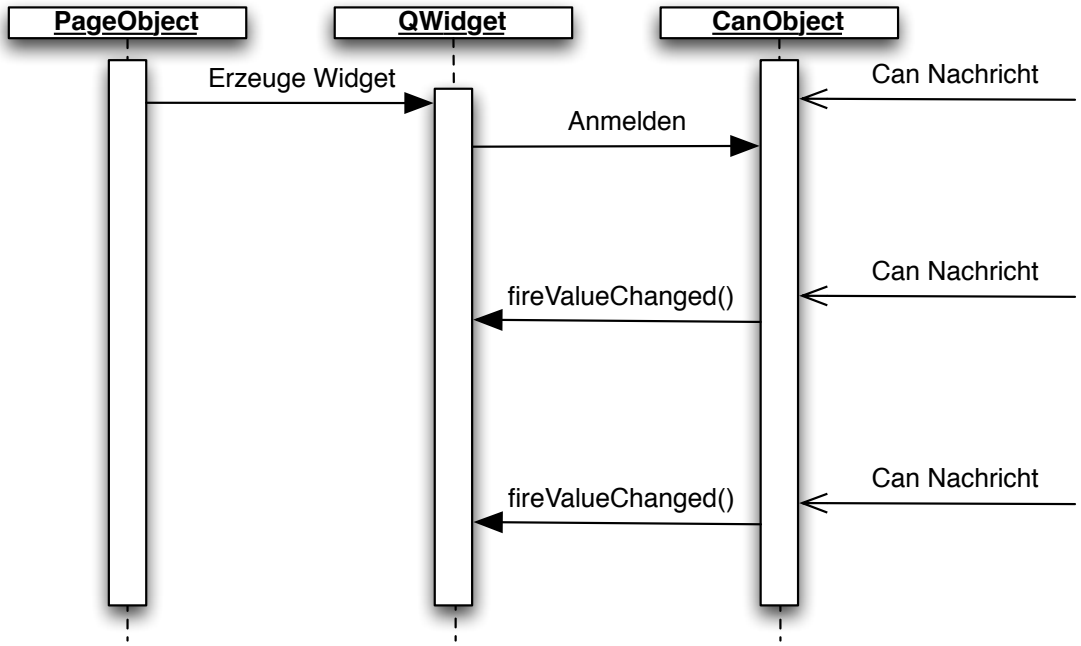

Abbildung 6.8: Anmeldung der Widgets an die CAN-Objekte

6.2.4 Konfigurierung der Software

Wie aus Kapitel 4.6 hervorgeht sollen die Darstellung und das Beziehen der Daten konfiguerierbar sein. Das Auslesen des XML-Dokuments geschieht in mehreren Schritten.

Zunächst werden die XML-Dokumente aufgelistet. Wenn der Benutzer ein Profil aus der Liste gewählt hat, werden die allgemeinen Daten und die Daten für die CAN-Objekte aus der XML gelesen. Mit diesen Daten wird die MenuControl-Klasse und alle in dem XML-Dokument definierten CAN-Objekte instanziiert.

Sind die Klassen instanziiert, hat der Benutzer die Wahl zwischen drei Driver-Modes. Erst wenn der Fahrer einen der drei Driver-Modes ausgewählt hat, werden auch die entsprechenden Daten aus dem XML-Dokument gelesen und die dazugehörigen PageObject Instanzen gebildet. Wechselt der Fahrer nun wieder in das Menü zum Laden der Driver-Modes, werden die PageObject Instanzen wieder gelöscht. Diese Maßnahme soll unzumutbar langes und speicherraubendes Laden von Widgets verhindern.

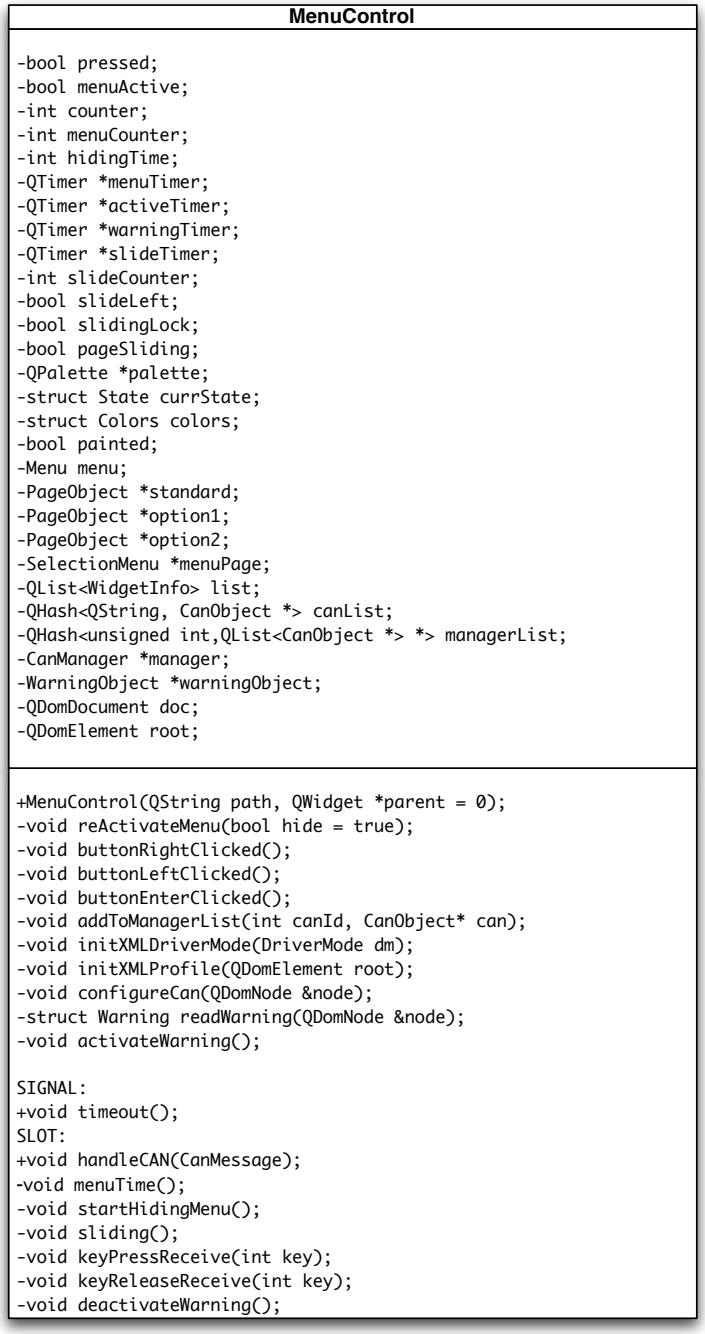

Abbildung 6.9: UML Klassendiagramm der MenuControl-Klasse

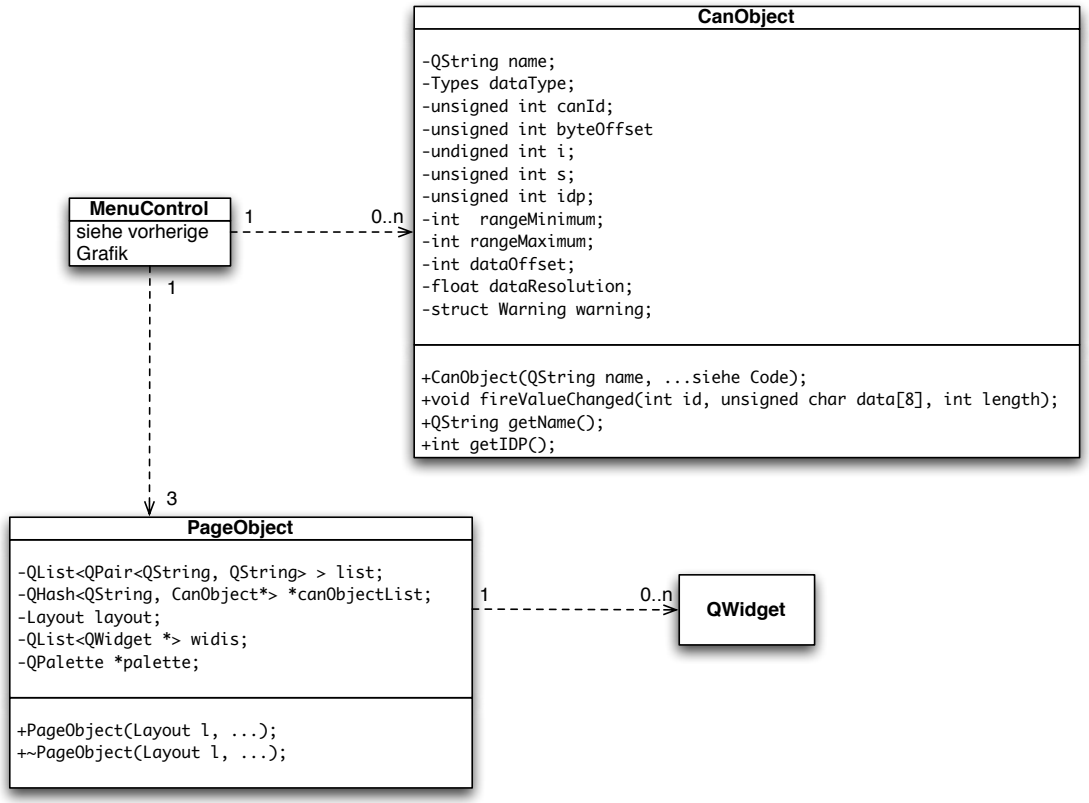

Abbildung 6.10: UML Klassendiagramm (2)

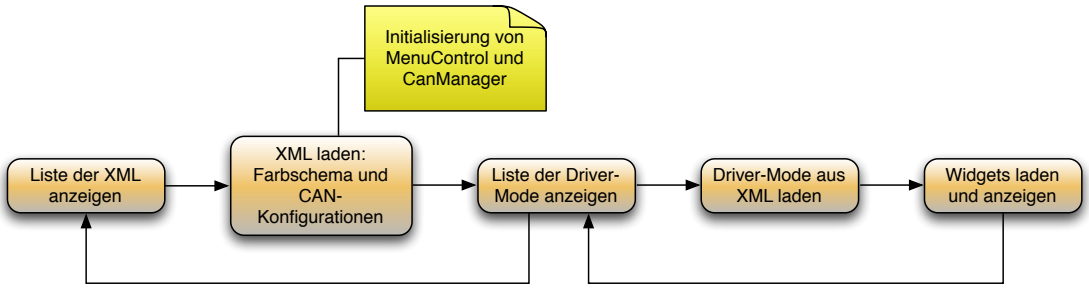

Abbildung 6.11: Lesen des XML-Dokuments

7 Evaluation im Usability-Labor

„Evaluation bezeichnet die systematische, datenbasierende Beschreibung und Bewertung von Software-Produkten“(siehe Konplan GmbH).

In diesem Kapitel wird die Software validiert. Es wird im Usbility-Labor geprüft, ob die Software und das Interface den Anforderungen aus dem Kapitel 2.6.5 entsprechen.
Das Usability-Labor ist zum Testen von Software konzipiert. Es besteht aus einem Testraum und einem Raum für die Testleitung und die Beobachter. Im Testraum werden die Testpersonen mit Kameras gefilmt, wie sie eine Software bedienen. Dabei gibt die Testleitung Aufgabenstellungen vor oder während des Tests.

Diese Tests sollen eine für das System typische Arbeitsumgebung bieten, damit die Ergebnisse im Labor den realen Situationen so weit wie möglich entsprechen. Die Arbeitsumgebung des Systems im Labor nachzubilden, gestaltet sich sehr schwierig. Zum einen gibt es die verschiedensten physischen Kräfte im Rennwagen, die auf das System und den Fahrer bei der Fahrt einwirken.
Zum anderen sollte die gleiche mentale Situation – wie Stress und starke Konzentration –, die sich auf dem Fahrer auswirkt, erzeugt werden.

Stärken der Usability-Tests im Labor sind (vgl. Seite 58 Richter und Flückiger, 2007):

- Unter Laborbedingungen können Schwachstellen der Benutzeroberfläche eindeutig nachgewiesen werden
- Validität kann weitgehend eingehalten werden
- die Beobachtungssituation ist in einem Usability-Labor optimal
- Die Methode ist für alle Beteiligten gut sichtbar. Die Bedeutung von Usability Engineering wird unmittelbar ersichtlich

7.1 Aufbau

Für den Usability-Test standen nur begingt viele Ressourcen zur Verfügung. Leider konnte das echte Lenkrad nicht genutzt werden, weil es zu der Zeit im Rennwagen gebraucht wurde. Das Problem wurde durch eine Simulation des Lenkrades auf einem 19“ Touch-Screen-Display gelöst, dass in einer Entfernung von etwa 46cm zur Testperson entfernt auf einem Tisch stand.

Zusätzlich hatte die Testperson einen großen 42“ Display. Auf diesem Display wurde der Testperson ein Autorennspiel gezeigt. Für das Lenken des spieles bekam die Testperson ein Spielelenkrad.
Die Testperson soll also das Spielelenkrad zum Lenken und das Simulierte Lenkrad zum ablesen der Daten und für die Menüsteuerung der Software benutzen.

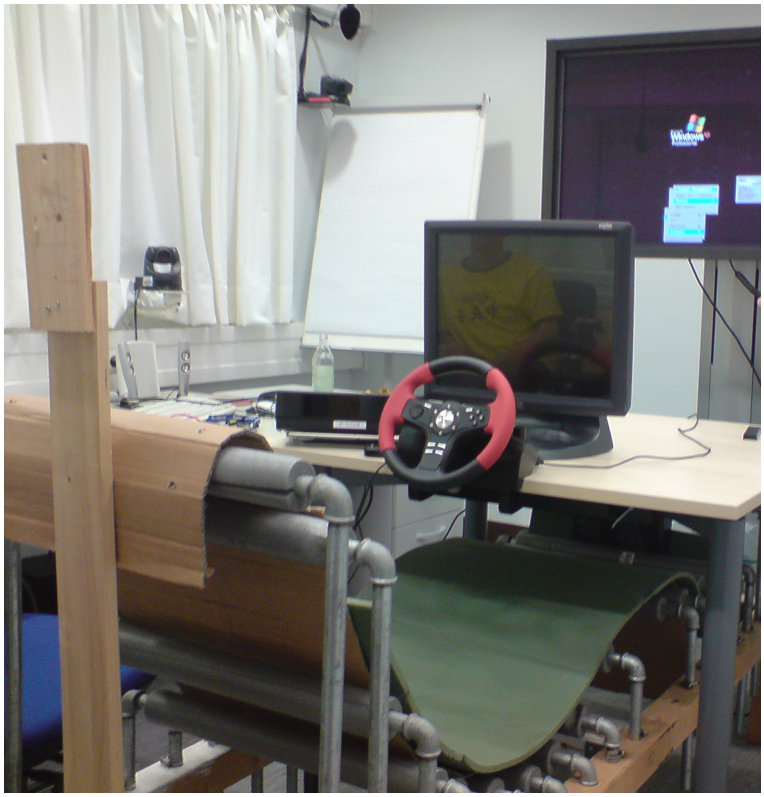

Abbildung 7.1: Erster Testaufbau

Während des Tests wurde der Fahrer kontinuierlich aus verschiedenen Perspektiven gefilmt

und eine vor der Testperson aufgestellte Eye-Tracker-Kamera verfolgte die Augen, wenn der Fahrer auf den Touch-Screen-Display geguckt hatte.

Die Eye-Tracker-Kamera sollte während der Fahrt erfassen, wie lange die Testperson seine Aufmerksamkeit dem Lenkrad zum Ablesen und steuern des Menüs widmet.

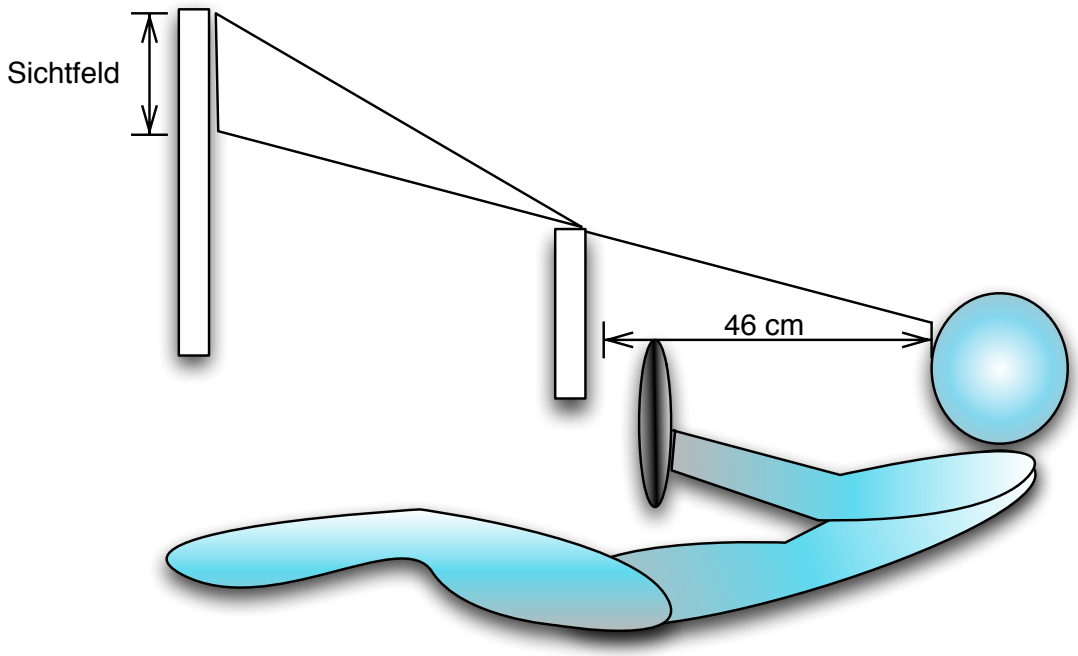

Abbildung 7.2: Eingeschränkte Sicht der Testperson auf das große Display

Ein Prototyp der Sitzschale des Rennwagens sollte genutzt werden, um die Fahrt im Rennwagen zu simulieren. Durch die liegende Position der Testperson ergaben sich leider Schwierigkeiten. Durch das relativ große Touch-Screen-Display auf dem Tisch vor dem Fahrer, war die Sicht auf dem großen Display eingeschränkt (siehe Abb. 7.2).

Daher entschieden wir uns einen normalen Bürostuhl zu nehmen und die Testperson aufrecht sitzen zu lassen. Zwar entstand dadurch ein größerer Blickwinkel zwischen Straße und Lenkrad – hiermit ist die Simulation auf dem Touch-Screen-Display gemeint –, aber der Fahrer konnte zumindest alles gut sehen.

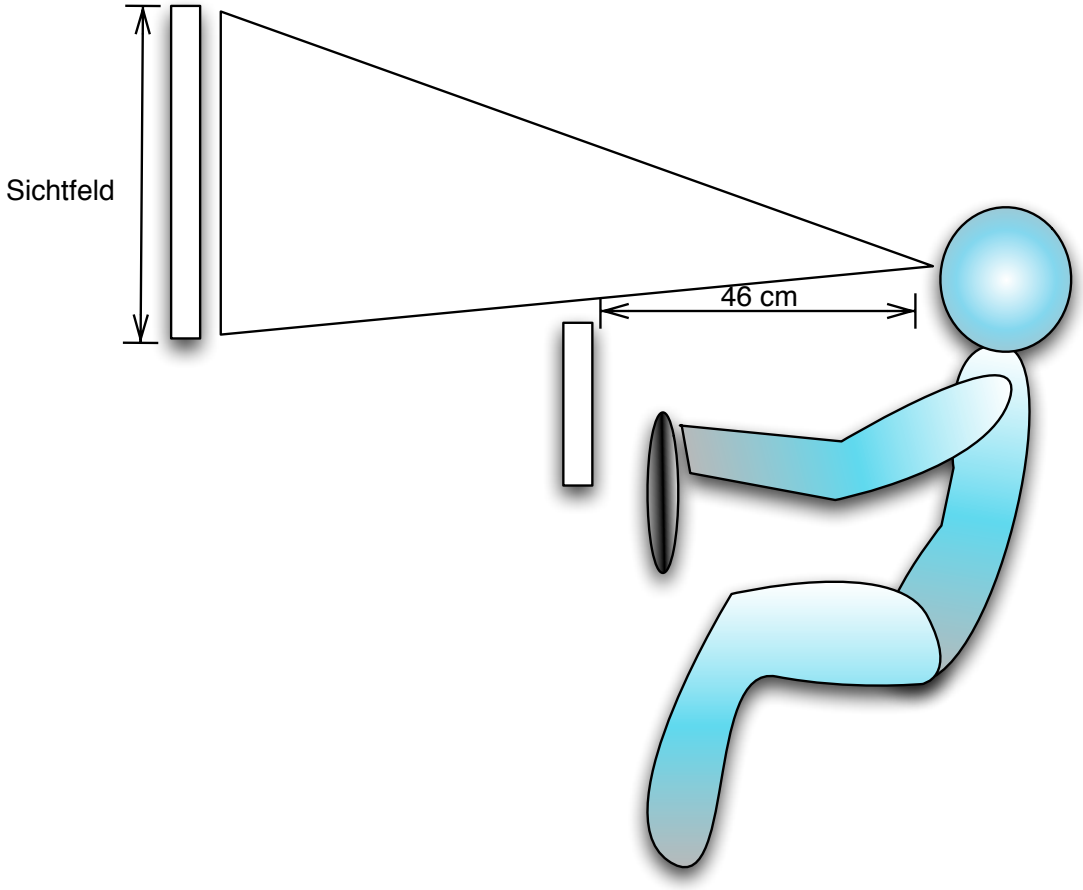

Abbildung 7.3: Gute Sicht der Testperson in der aufrecht sitzenden Position

Der Raum für die Testleitung ist mit verschiedenen Displays ausgestattet. Der Ansager hat ein Mikrofon mit dem er der Testperson Anweisungen gibt. Er verfolgt den Test über 4 Monitore, auf denen er sieht was die Testperson macht und was sich gerade auf dem großen Display abspielt.

Der Koordinator hat ein Kontrollpullt, einen Monitor der ein Abbild des Touch-Screen-Displays zeigt und einen weiteren Rechner, mit dem er verschiedene Eingriffe in die Software des Lenkrades machen kann.

Das Kontrollpullt hat verschiedene Taster. Mit den Tastern lassen sich zum einen Zeitmessungen und zum anderen über ein Mikrofon mündliche Notizen machen.

Die Zeitmessungen und mündlichen Notizen werden auf einen Rechner festgehalten. An dem gleichen Rechner wird die Kalibrierung des Eye-Trackers vorgenommen und die Aufnahmen der Kameras im Testraum gesteuert.

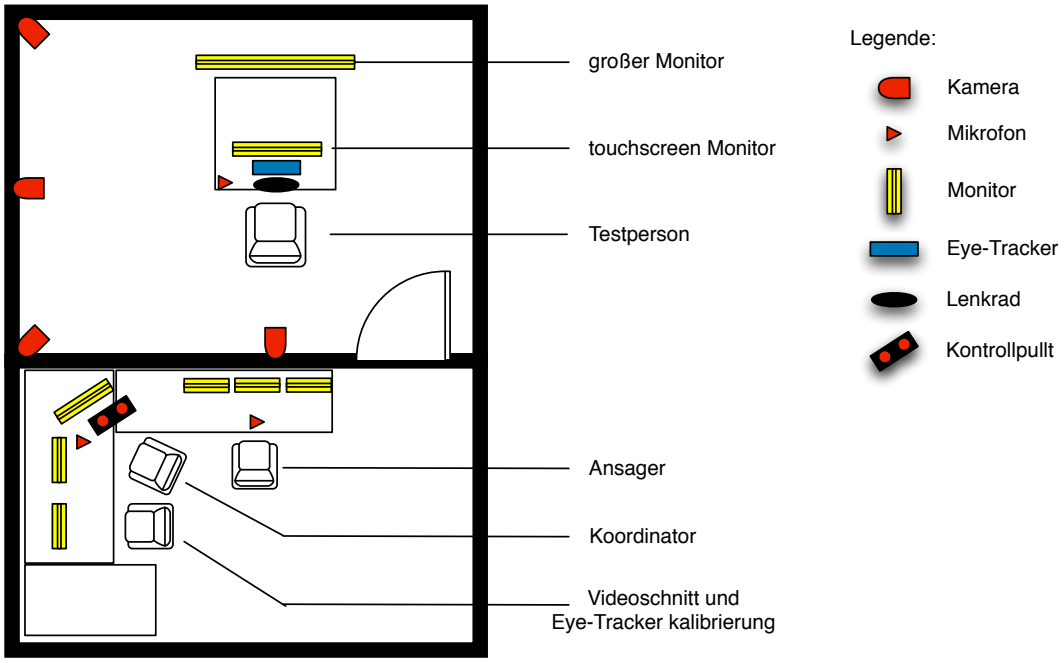

Abbildung 7.4: Aufbau der Testumgebung

7.2 Durchführung

Die Druchführung ist in mehreren Schritten aufgeteilt, die nacheinander ausgeführt werden.

1. Einführung der Testperson
2. Kalibrieren der Testumgebung
3. Testfahrt
4. Testen der Menüführung

5. Fragebögen ausfüllen

In der Einführung wird der Testperson die Menüführung der zu testenden Software gezeigt und erklärt. Zudem wird der Testperson die Durchführung erklärt, bei der sie angewiesen wird, laut zu denken und auffällige Veränderungen in der Simulation des Lenkrades sofort mitzuteilen.
In der Einführung kriegt die Person zwei Minuten Zeit das Rennspiel aus zu probieren.

Anschließend wird die Testumgebung auf die Testperson eingestellt. Die Entfernung des Touch-Screen-Displays sollte ungefähr auf Armlänge Entfernung liegen, damit die Testperson die Simulation noch bedienen kann. Daraufhin wird der Eye-Tracker kalibriert. Ist er einmal kalibriert, sollte die Testperson den Kopf möglichst nicht mehr bewegen, damit der Eye-Tracker richtig funktionieren kann.

7.2.1 Testfahrt

Nun kann der Fahrer seine Testfahrt beginnen. Bei geschlossenen Raum bekommt der Fahrer durch den Ansager Anweisungen.
Die Videoaufnahme und die Simulation auf dem Touch-Screen-Display wird gestartet und die Testperson wird angewiesen sich auf das Spiel zu konzentrieren.

Das Spiel soll die Testperson wie in einer richtigen Fahrt im Rennwagen fordern sich voll zu konzentrieren. Der Ansager gibt in folgender Reihenfolge anweisungen:

1. schaue auf die Geschwindigkeit
2. schaue auf die Drehzahl
3. schaue auf die Geschwindigkeit
4. schaue auf die Geschwindigkeit
5. schaue auf die Drehzahl
6. schaue auf die Batterie
7. schaue auf die Wasser-Temperatur

Die Anweisungen werden nur in Spielesituationen die es zulassen kurz auf die Anzeige zu schauen gegeben. Der Ansager achtet also darauf, dass die Testperson gerade auf einer Geraden fährt.
Die Testperson soll nach jeder Anweisung laut und schnell die Daten vom Display ablesen.

Zu jeder Anweisung macht der Koordinator mittels Kontrollpullt mündliche Notizen über Genauigkeit, Geschwindigkeit oder Auffälligkeiten beim Ablesen.

Zwischen den Anweisungen gibt der Koodinator mindestens eine Fehlermeldung auf das simulierte Lankraddisplay. Dabei misst der Koordinator die Zeit, die die Testperson braucht, die Fehlermeldung zu bemerken (perzeption), zu erkennen (kognition) und laut aufzusagen.

7.2.2 Testen der Menüführung

Zum testen der Menüführung bekommen alle Testpersonen drei Aufgabenstellungen vom Ansager durchgesagt.

1. Du stehst kurz vor dem Acceleration und möchtest die Öl-Temperatur überprüfen. Bitte navigiere im Menü zur entsprechenden Seite.
2. Du willst jetzt den Acceleration starten. Bitte gehe in die Hauptseite für den Start.
3. Du willst jetzt in den Endurance wechseln. Bitte wechsele zum Endurance-Mode.

Nachdem die Aufgaben durchgegeben wurden, misst der Koordinator die Zeit bis die Testperson die Aufgabe gelöst hat. Daraufhin macht der Koordinator mündliche Notizen zu Auffälligkeiten.

Die Aufgabenstellungen sollen zeigen, ob die Testperson die Menüführung verstanden hat und damit umgehen kann.

7.3 ISO-Fragebogen

Als Fragebogen wurden Teile aus dem ISO-Fragebogen ISONORM 9241/10 benutzt. Fragen zu Lernförderlichkeit, Erwartungskonfomität, Aufgabenangemessenheit und Selbstbeschreibungsfähigkeit des Programms wurden in dem Fragebogen untergebracht.

Der Fragebogen bestand demnach aus den Abbildungen 7.5, 7.6, 7.7 und 7.8.

Für mehr Informationen zum ISO-Fragebogen siehe Bräutigam (2008).

7.4 Testergebnisse

Die Tests ergaben viele neue Ansichten über die grafische Darstellung und den Auswirkungen auf die Aufmerksamkeit des Fahrers.

Die aufgenommenen Videos zeigten einige Schwächen und deuteten an bei welchen Vorgehen die Testpersonen vom Fahren abgelenkt waren.
Obwohl der Fahrer nicht während der Fahrt die Pages wechseln soll, wurde ihm der Auftrag indirekt gegeben. Alle Testkandidaten haben sofort die Pages gewechselt, nachdem sie die Anweisung bekamen eine Information laut zu sagen die nicht auf der jeweiligen Page war. Abgesehen vom erschwerten Fall der Bedienung eines Touch-Screen-Displays – wegen der nicht erfühlbaren Taster und dessen Druckpunkt –, haben alle Testkandidaten lange für das bedienen der Software gebraucht und somit ihre Aufmerksamkeit zu Strecke verloren.

Außerdem sind viele der Kandidaten nicht so schnell dahinter gekommen, dass wenn die Labels für die Taster gerade verschwunden sind (inaktiv), man einen Knopfdruck braucht um sie zu aktivieren und einen zweiten um eine Aktion durchzuführen. Das ist anscheind ein Schritt zuviel, auch wenn die Testpersonen schnell verstanden haben wie das Menü funktioniert.

Zusätzlich zu der Videoaufnahme gibt es noch die Aufzeichnungen des Eye-Trackers. Die Aufzeichnungen geben aufschluss über die Zeit, die die Testperson auf das Lankrad geguckt hat.

Das bloße Ablesen einer Information vom Lankraddisplay dauerte durchschnittlich 0,5 bis 0,7 Sekunden, wobei das Ablesen der Information mit vorrigem Wechsel der Page zwischen

3 bis 7 Sekunden schwankte.

Falls das wechseln der Pages gewollt ist, sollten die Taster zum wechseln in der Nähe der Hände am Lenkrad angebracht werden wie Abbildung 5.1 zeigt.

Die Ergebnisse des Fragebogens geben Informationen über einige Eigenschaften, die man noch verbesserm könnte (siehe Abb. 7.10).

Die Selbstbeschreibungsfähigkeit der Software sollte verbessert werden. Als Maßnahme für Erklärungen könnte ein zusätzlicher Menüpunkt „Hilfe“ weiter helfen. Außerdem wäre ein Handbuch über die Konfiguration und Bedienung der Software sehr hilfreich für den späteren Benutzer.

Aufgabenangemessenheit

Die Software ...	---	--	-	-/+	+	++	+++	*Die Software ...*
ist kompliziert zu bedienen.								ist unkompliziert zu bedienen.
bietet nicht alle Funktionen, um die anfallenden Aufgaben effizient zu bewältigen.								bietet alle Funktionen, um die anfallenden Aufgaben effizient zu bewältigen.
bietet schlechte Möglichkeiten, sich häufig wiederholende Bearbeitungsvorgänge zu automatisieren.								bietet gute Möglichkeiten, sich häufig wiederholende Bearbeitungsvorgänge zu automatisieren.
erfordert überflüssige Eingaben.								erfordert keine überflüssigen Eingaben.
ist schlecht auf die Anforderungen der Arbeit zugeschnitten.								ist gut auf die Anforderungen der Arbeit zugeschnitten.

Abbildung 7.5: Aufgabenangemessenheit (ISONORM-Fragebogen)

Erwartungskonformität

Die Software ...	---	--	-	-/+	+	++	+++	*Die Software ...*
erschwert die Orientierung, durch eine uneinheitliche Gestaltung.								erleichtert die Orientierung, durch eine einheitliche Gestaltung.
läßt einen im Unklaren darüber, ob eine Eingabe erfolgreich war oder nicht.								läßt einen nicht im Unklaren darüber, ob eine Eingabe erfolgreich war oder nicht.
informiert in unzureichendem Maße über das, was sie gerade macht.								informiert in ausreichendem Maße über das, was sie gerade macht.
reagiert mit schwer vorhersehbaren Bearbeitungszeiten.								reagiert mit gut vorhersehbaren Bearbeitungszeiten.
läßt sich nicht durchgehend nach einem einheitlichen Prinzip bedienen.								läßt sich durchgehend nach einem einheitlichen Prinzip bedienen.

Abbildung 7.6: Erwartungskonformität (ISONORM-Fragebogen)

Selbstbeschreibungsfähigkeit

Die Software ...	---	--	-	-/+	+	++	+++	*Die Software ...*
bietet einen schlechten Überblick über ihr Funktionsangebot.								bietet einen guten Überblick über ihr Funktionsangebot.
verwendet schlecht verständliche Begriffe, Bezeichnungen, Abkürzungen oder Symbole in Masken und Menüs.								verwendet gut verständliche Begriffe, Bezeichnungen, Abkürzungen oder Symbole in Masken und Menüs.
liefert in unzureichendem Maße Informationen darüber, welche Eingaben zulässig oder nötig sind.								liefert in zureichendem Maße Informationen darüber, welche Eingaben zulässig oder nötig sind.
bietet auf Verlangen keine situationsspezifischen Erklärungen, die konkret weiterhelfen.								bietet auf Verlangen situationsspezifische Erklärungen, die konkret weiterhelfen.
bietet von sich aus keine situationsspezifischen Erklärungen, die konkret weiterhelfen.								bietet von sich aus situationsspezifische Erklärungen, die konkret weiterhelfen.

Abbildung 7.7: Selbstbeschreibungsfähigkeit (ISONORM-Fragebogen)

Lernförderlichkeit

Die Software ...	---	--	-	-/+	+	++	+++	*Die Software ...*
erfordert viel Zeit zum Erlernen.								erfordert wenig Zeit zum Erlernen.
ermutigt nicht dazu, auch neue Funktionen auszuprobieren.								ermutigt dazu, auch neue Funktionen auszuprobieren.
erfordert, daß man sich viele Details merken muß.								erfordert nicht, daß man sich viele Details merken muß.
ist so gestaltet, daß sich einmal Gelerntes schlecht einprägt.								ist so gestaltet, daß sich einmal Gelerntes gut einprägt.
ist schlecht ohne fremde Hilfe oder Handbuch erlernbar.								ist gut ohne fremde Hilfe oder Handbuch erlernbar.

Abbildung 7.8: Lernförderlichkeit (ISONORM-Fragebogen)

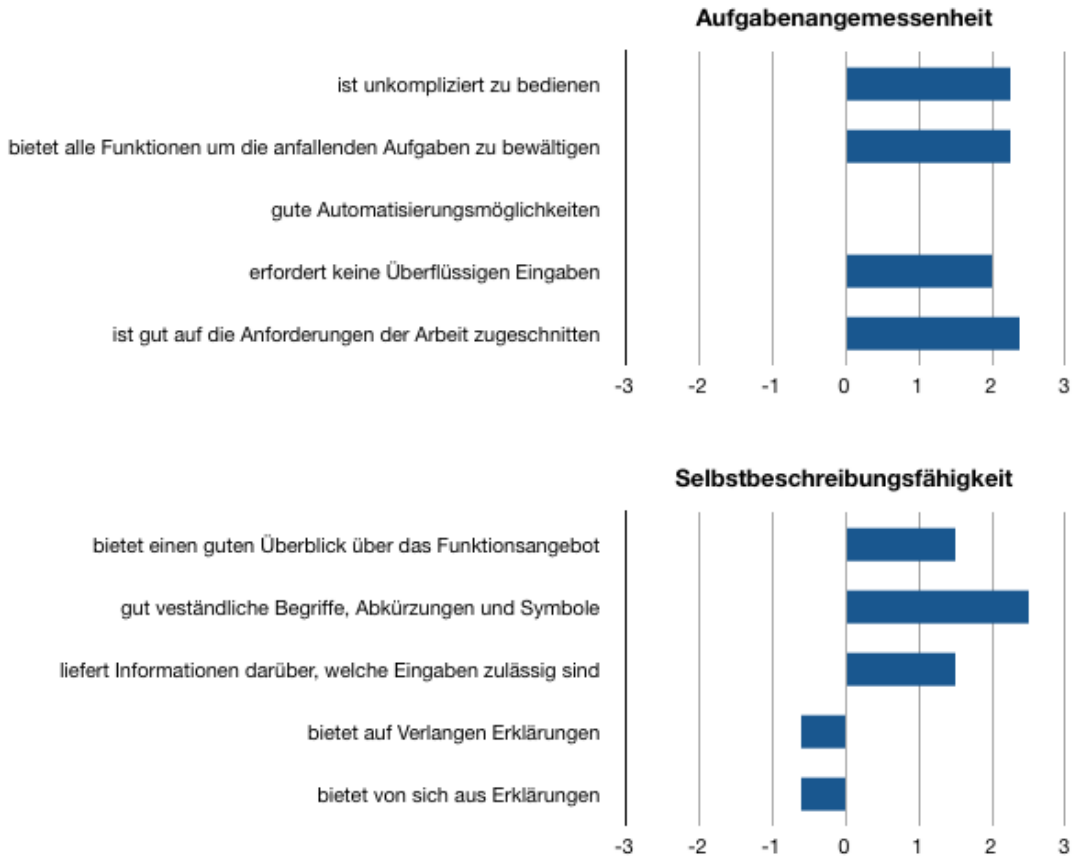

Abbildung 7.9: Auswertung der Fragebögen

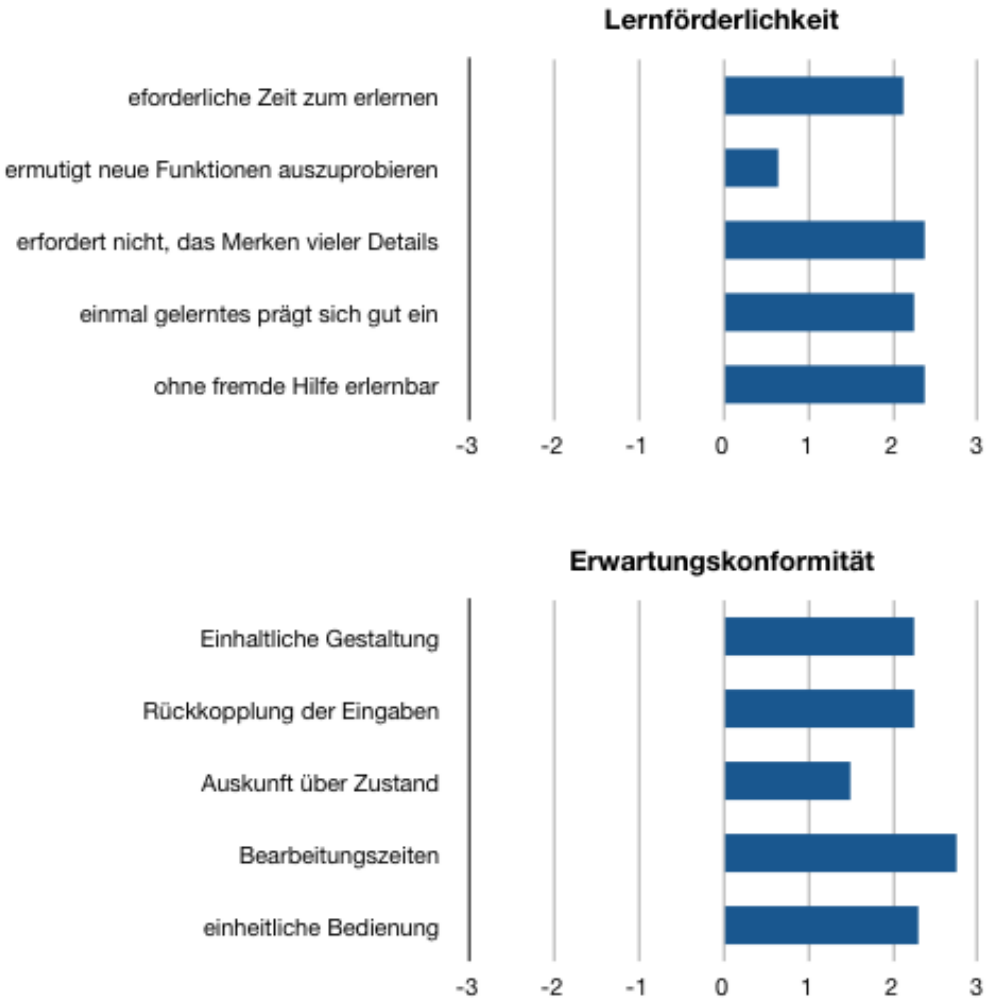

Abbildung 7.10: Auswertung der Fragebögen

8 Fazit

Abbildung 8.1: Das Multifunktionslenkrad des H04

Man kann am Ende sagen, dass das fertige System nach Usability-Gesichtspunkten entwickelt wurde. Es verfügt über eine Profilverwaltung, die jedem Benutzer erlaubt seine eigene Darstellung der Daten zu konfigurieren. Die Software wurde von der Hardware unabhängig programmiert.
Auch innerhalb der Software, ist das Layout von der Logik lose gekoppelt, sodass man die Software auch für großere Displays programmieren könnte. Zudem ist das Beziehen der Daten von der CAN-Bus-Telemetrie so konfigurierbar, dass Veränderungen der Daten auf dem CAN-Bus leicht einstellbar sind. Man kann sogar neue CAN-Daten definieren und ihnen grafische Anzeigen zuordnen. Das heißt, dass neue Sensoren leicht implementierbar sind ohne einen Programmieraufwand zu betreiben.

Die Software kann in ihrem Entwicklungsstand schon genutzt werden. Die Untersuchungen im Usability-Labor ergaben aber, dass die Software weiter entwickelt und verbessert werden kann.
Ganz besonders das Lenkrad und das Display sollten in der Form nocheinmal überarbeitet bzw. ausgetauscht werden. Sie genügen den in dieser Arbeit erschlossenen Anforderungen nicht.

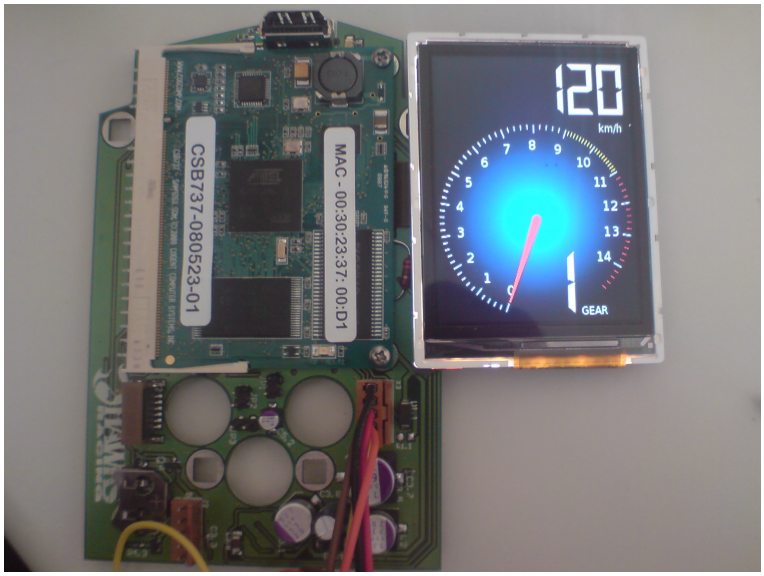

Abbildung 8.2: Das laufende System

Es gibt etliche Verbesserungen, die man an der Software noch machen kann. Im Usability-Labor ist aufgefallen, dass Benutzer, wenn Sie während der Fahrt die Pages wechseln, auf der Page bleiben auf der sie sich befinden, obwohl die eigentlichen Informationen für den jeweiligen Driver-Mode sich auf der Hauptseite befinden.
Ein automatischer Wechsel auf die Haputseite, der nach einer bestimmten Zeit x eintrifft, sollte das Problem lösen.

Weiterhin könnte die Ansteuerung der Grafik-Beschleuniger-Hardware im zusammenhang mit Direct Painting, einer Methode von QT direkt auf dem Bildschirm zu zeichnen, mehr grafische Möglichkeiten zur Verbesserung der Ästhetik bieten.

Bisher kommen die XML-Dokumente nur auf die Hardware, wenn die Hardware ausgebaut ist. Das Problem sollte man entweder über eine Service-Lucke für einen Datenträger wie eine SD-Karte oder über das direkte Flashen der Hardware vom CAN-Bus aus lösen.

Für das Lenkrad sollten zusätzliche Taster in der Nähe der Hände angebracht werden, damit der Fahrer auch während der Fahrt die Pages wechseln kann.

Eine mögliche Variante des nächsten Lenkrades könnte wie in Abbildung 8.3 aussehen.

Abbildung 8.3: Zeichnung eines Prototypen

8.1 Ausblick

Das in dieser Arbeit entwickelte System bietet keine Innovationen. Es ist aber eine Entwicklung mit vielen Funktionen und Überlegungen, die ausbaufähig sind und das System zu einem besonderen machen.

Zum Beispiel könnte man den besten Schaltzeitpunkt statt visuell, akustisch direkt im Helm des Fahrers mitteilen.

Die Rundenzeiten könnten von außen über Wlan in die CAN-Bus-Telemetrie gegeben und auf dem Display angezeigt werden. So könnte man auch eine Kommunikation zwischen Leitstand und Fahrer herstellen.

Für das Erstellen der XML-Dokumente muss man sich schon eingehender mit XML und der Rennwagen-Telemetrie beschäftigt haben. Um es aber mehr Benutzern einfacher zugänglich zu machen, könnte man eine Software entwickeln, die die Darstellung und Einstellungen über eine Art Wizard in ein XML-Dokument schreibt. Diese Software könnte mit Drag&Drop

Funktionalität den Benutzer das Einstellen der Layouts für das eigene Profil leichter machen und gleichzeitig damit ausschließen, dass der Benutzer keine ungültigen Eingaben im XML-Dokument vornimmt.

Der nächste Schritt der Evolution dieses Systems wäre dann das Schreiben auf die CAN-Bus-Telemetrie. Damit könnte man zum Beipiel den Drehzahlbegrenzer per Software festlegen oder Einstellungen an der Motorsteuerung während des Betriebes des Fahrzeuges vornehmen.

Literaturverzeichnis

[BROCKHAUS 2000] *Der Brockhaus*. Brockhaus GmbH, 2000. – ISBN 3-7653-3642-4

[AiM 2008] AiM, Sports: *Mychron3 Gold*. http://www.aimsports.com/products/m3-gold-auto/index.html. letzter Zugriff Juli 2008

[Andreae 2008] ANDREAE, Johann-Nikolaus: *Das Lenkraddisplays eines Formula Student Rennwagens: von der Analyse, über die Hardware- und Linuxtreiberentwicklung bis zum Prototypen*, Hochschule für Angewandte Wissenschaften Hamburg, Diplomarbeit, 2008

[Balzert Heidelberg 2001] BALZERT, Helmut: *Lehrbuch der Software-Technik. Bd.1. Software-Entwicklung*. Spektrum Akademischer Verlag, Heidelberg 2001. – ISBN 3-8274-0480-0

[Bräutigam 2008] BRÄUTIGAM, Lothar: *Beurteilung der Software-Ergonomie anhand des ISONORM-Fragebogens*. letzter Zugriff August 2008. – URL http://www.ergo-online.de/site.aspx?url=html/software/verfahren_zur_beurteilung_der/beurteilung_der_software_ergo.htm

[Buschmann 2008] BUSCHMANN, Frank: *Pattern: Publisher-Subscriber*. letzter Zugriff August 2008. – URL http://www.vico.org/pages/PatronsDisseny/Pattern%20Publisher%20Subscriber/index.html

[Dahm 2006] DAHM, Markus: *Grundlagen der Mensch-Computer-Interaktion*. 1. Pearson Studium, 2006. – ISBN 3-8273-7175-9

[Digital-World 2008] DIGITAL-WORLD: *Alles über Displays*. letzter Zugriff August 2008. – URL http://www.digital-world.de/videotv/trends/reports/displays/ratgeber_alles_ueber_displays/1258290/ratgeber_alles_ueber_displays.html

[DIN EN ISO 13407 ?] *Benutzer-orientierte Gestaltung interaktiver Systeme*. ?. – DIN Deutsches Institut für Normung e.V; Normenausschuß Informationstechnik

[DIN EN ISO 15008 2003] *Ergonomische Aspekte von Fahrerinformations- und Assistenzsystemen*. 2003. – DIN Deutsches Institut für Normung e.V; Normenausschuß Informationstechnik

[DIN EN ISO 9241-11 1998] *Anforderungen an die Gebrauchstauglichkeit - Leitsätze*. 1998. – DIN Deutsches Institut für Normung e.V; Normenausschuß Informationstechnik

[DIN EN ISO 9241-110 2006] *Grundsätze der Dialoggestaltung*. 2006. – DIN Deutsches Institut für Normung e.V; Normenausschuß Informationstechnik

[DIN EN ISO 9241-12 1998] *Informationsdarstellung*. 1998. – DIN Deutsches Institut für Normung e.V; Normenausschuß Informationstechnik

[DIN EN ISO 9241-14 ?] *Dialogführung mittels Menüs*. ?. – DIN Deutsches Institut für Normung e.V; Normenausschuß Informationstechnik

[DIN EN ISO 9241-3 ?] *Anforderungen an visuelle Anzeigen*. ?. – DIN Deutsches Institut für Normung e.V; Normenausschuß Informationstechnik

[Dr. Sportwiss. Ralf Pfeifer 2008] DR. SPORTWISS. RALF PFEIFER: *Reaktionszeiten*. letzter Zugriff Juli 2008. – URL http://www.arsmartialis.com/index.html?name=http://www.arsmartialis.com/technik/reaktion/reaktion.html

[EMF-Portal 2008] EMF-PORTAL: *Elektrische Felder*. letzter Zugriff Juli 2008. – URL http://www.emf-portal.de/lfu.php?l=g&detail=4#headline

[für Bildung und Forschung 2008] FORSCHUNG, Bundesministerium für Bildung und: *Organische Leuchtdioden - die Tapete als Lichtquelle?* letzter Zugriff Juli 2008. – URL http://www.bmbf.de/de/3604.php

[Gegenfurtner 2008] GEGENFURTNER, Prof. Karl R.: *Farbensehen beim Menschen*. letzter Zugriff Juli 2008. – URL http://www.allpsych.uni-giessen.de/karl/html/heidelberg/heidelberg.html

[Haase 2007] HAASE, Sebastian: *Telemetrie im Formula Student Rennwagen auf Basis von CAN Bus, DatenspeicherungundWireless LAN Technologien*, Hochschule für Angewandte Wissenschaften Hamburg, Diplomarbeit, 2007

[Hartkopp 2008] HARTKOPP, Oliver: *Low Level CAN Framework*. letzter Zugriff August 2008. – URL http://svn.berlios.de/svnroot/repos/socketcan/trunk/

[Hitachi 2006] Hitachi (Veranst.): *Customer's Acceptance Specifications TX07D09VM1CBB*. November 2006

[Ihlenfeld 2008] IHLENFELD, Jens: *SED, besser als Plasma-TV und LCD?* letzter Zugriff August 2008. – URL http://www.golem.de/0509/40252-2.html

[ISO 2575 1994] : *Strassenfahrzeuge - Symbole fuer Bedienteile, Anzeige- und Warngeraete*. 1994

[Konplan GmbH] KONPLAN GMBH: *Damit Ihre Software allen Qualitätsansprüchen gerecht wird*. Broschüre. – liegt der CD in PDF Form bei

[Organization 2008] ORGANIZATION, World Interlectual P.: *METHOD AND DEVICE FOR LCD-LABEL*. letzter Zugriff August 2008. – URL http://www.wipo.int/pctdb/en/wo.jsp?IA=SE1996001727&DISPLAY=DESC

[Rauchmann 2008] RAUCHMANN, Augenoptik: *Das Auge*. letzter Zugriff Juli 2008. – URL http://www.optik-rauchmann.de/Das_Auge.htm

[Richter und Flückiger 2007] RICHTER, Michael ; FLÜCKIGER, Markus: *Usability Engineering kompakt*. Spektrum Akademischer Verlag, 2007. – ISBN 978-3-8274-1837-1

[Schuckert 2007] SCHUCKERT, Simon: *Mikrocontrollerbasier te Telemetrie und Echtzeitauswer tung von Sensordaten im Formula Student Rennwagen*, Hochschule für Angewandte Wissenschaften Hamburg, Diplomarbeit, 2007

[Schwarz 2006] SCHWARZ, M.: *Die Idee der Formula SAE / Student*. 2006. – URL http://www.hawksracing.de/index.php?page=ueber_fsae

[Schäfers 2007] SCHÄFERS, Prof. M.: *Rechnerstrukturen*, HAW Hamburg, Vorlesungsfolie, 2007

[Shneidermann 2002] SHNEIDERMANN, Ben: *User Interface Design*. mitp-Verlag, 2002. – ISBN 3-8266-0753-8

[Sixtus 2008] SIXTUS, Mario: *Gemeinsam auf die Spitze*. letzter Zugriff August 2008. – URL http://www.zeit.de/2004/01/T-Extremprogrammierer?page=1

[Stangl 2008] STANGL, Werner: *Kognition*. letzter Zugriff Juli 2008. – URL http://www.stangl.eu/psychologie/definition/Kognition.shtml. – Johannes Kepler Universität Linz

[Stapelkamp 2007] STAPELKAMP, Torsten: *Screen- und Interfacedesign*. Springer-Verlag Heidelberg, 2007. – ISBN 978-3-540-32949-7

[Steinbacher 2006] STEINBACHER, Iris: Bessere Sicht bei Tageslicht. In: *Elektronik Praxis* (2006), November. – URL http://www.elektronikpraxis.vogel.de/themen/hardwareentwicklung/displays/articles/37680/

[Strasser 2008] STRASSER, Erich: *Was bedeutet SED-TV*. letzter Zugriff Juli 2008. – URL http://www.sed-fernseher.eu/was-bedeutet-sed-tv

[VDI 2008] VDI: *Formula Student*. letzter Zugriff Juli 2008. – URL http://www.formulastudent.de

[W3C 2008] W3C: *Extensible Markup Language (XML) 1.0*. letzter Zugriff August 2008. – URL http://www.edition-w3c.de/TR/2000/REC-xml-20001006/

[Winter 2007] WINTER, Matthias: LEDs erobern große Displays. In: *Elektronik Praxis* (2007), Mai. – URL http://www.elektronikpraxis.vogel.de/themen/hardwareentwicklung/optoelektronik/articles/64969/

[Wissenschaft-Online 2008] WISSENSCHAFT-ONLINE: *Akkomodationsbreite*. letzter Zugriff Juli 2008. – URL http://www.wissenschaft-online.de/abo/lexikon/bio/1713

Glossar

Adaption Anpassung der Empfindlichkeit der Augen an die Helligkeit des betrachteten visuellen Feldes

Akkommodation ist der Vorgang des Auges ein Objekt scharf zu stellen. Dabei wird die Linse des Auges über die Ziliarmuskeln des Ziliarkörpers kontrahiert oder entspannt um die Brechkraft der Linse zu ändern und somit die Gegenstände scharf auf die Netzhaut abzubilden.

Akkomodationsbreite meint den Bereich zwischen dem Punkt an dem man am nähersten(Nahpunkt) und den Punkt an dem man am weitesten(Fernpunkt) vom Auge weg scharf sehen kann. Die Einheit für die Akkommodation ist Dioptrin(dpt) und wird mit einem Akkodometer gemessen.

ARM-9 ARM-9 ist ein Microprozessor mit ARM-Architektur von der Familie der 32-Bit-RISC-Prozessoren. Der ARM-9 mit seiner Harvard Bus-Architektur arbeitet registerorientiert, d.h. er führt seine arithmetischen und logischen Befehle auf Registern aus. Durch seine eher RISC untypischen Ziele, wie geringer Stromverbrauch, ist der ARM-9 für den Einsatz im Embedded Bereich, wie beispielsweise in Automobilen ideal geeignet. Außerdem hat der ARM-9 eine geringe Siliziumfläche, was sich günstig im Preis auswirkt. (vgl. Schäfers, 2007)

Bereich Teil oder Fläche eines Bildschirms oder eines Fensters

Bitkipper ein Bitfehler der ein einzelnes Bit verdreht und damit die Daten verändert

blinken ‘beabsichtigte periodische Veränderung der Leuchtdichte einer Lichtquelle oder einer visuellen Information, üblciherweise vom ‘Aus’-Zustand zu einem bestimmten Wert, überlicherweise verwendet, um Ausfmerksamkeit zu wecken ‘(siehe Seite 5 DIN EN ISO 15008, 2003)

CAN Control Area Network : Ein serielles Feldbus-System von Bosch 1983 entwickelt. Konzipiert wurde es für Automobile, um die Verkablung zu verringern und dadurch gewicht zu sparen.

Dashboard englisch für Amaturenbrett. Meist hinter dem Lenkrad positioniert, beinhaltet das Dashboard Instrumente und Anzeigen für den Fahrer.

Druckpunkt beim Drücken eines Schalters oder Knopfes die Tiefe, bis zu der ein elektrischer Schalter aktiviert wird

Extreme Programming agile Softwareentwicklung, bei der die einzelnen Funktionalitäten mit den Kunden und Benutzern erötert und anhand derer implementiert wird, sodass eine ständige Kommunikation zwischen Kunden und Programmierer besteht, um in kleinen Schritten eine Rückkopplung zu erhalten (vgl. Sixtus, 2008).

Framebuffer ist ein Zwischenspeicher in dem die Daten des Displays vorgehalten werden, damit die Daten schneller auf das Display geladen werden können

GUI Graphical User Interface

Gulf of Execution der Anwender weiss keine Möglichkeit seine Ziele mit dem System umzusetzen

Kognition ist ein Begriff der alle Prozesse, die mit dem Erkennen einer Situation zusammenhängen. Folgende Fähigkeiten sind gemeint: Aufmerksamkeit, Wahrnehmungsfähigkeit, Erkenntnisfähigkeit, Schlussforgerung, Urteilsfähigkeit, Erinnerung, Lernfähigkeit, Abstraktionsvermögen und Rationalität. (vgl. Stangl, 2008)

Kontrastverhältnis Verhältnis zwischen der Leuchtdichte L_{high} eines Feldes in 'hellem' Zustand (wie z.B. die Striche eines Zeichens im Fall negativer Polarität (siehe Seite 5 DIN EN ISO 15008, 2003)) und der Leuchtdichte des gleichen Feldes L_{low} im 'dunklen' Zustand

Label Kurze beschreibende Bezeichnung oder Überschrift für ein Eingabe oder Anzeigefeld, eine Tabelle, ein Steuerungselement oder eine Objekt (siehe Seite 5 DIN EN ISO 9241-12, 1998)

LCD Liquid Crystal Display basiert auf flüssigen lichtdurchlässigen Kristallen, die sich in Kammern befinden und somit einen Pixel darstellen. In diesen Kammern gibt es bei farbigen Displays drei Subpixel die den Farben Rot, Grün und Blau (RGB) entsprechen. Mit der Eigenschaft der Kristalle zu regulieren wieviel Licht durchgelassen wird, kann so eine Farbe für ein einzelnes Pixel durch die hohe der Spannung angesteuert werden. (vgl. Digital-World, 2008)

Lesbarkeit bezieht sich auf die visuellen Eigenschaften eines Zeichens oder Symbols, die fürdie Einfachheit der Erkennung verantwortlich sind (siehe (Seite 6 DIN EN ISO 15008, 2003))

Linux Embedded eine abgespeckte Version eines Linux Betriebssystems

Listen Darstellung von Daten in waagerechter oder senkrechter Form in einer Anzeige, die sich üblicherweise in Abhängigkeit vom Zusatnd der Anwendung ändert (siehe Seite 5 DIN EN ISO 9241-12, 1998)

Nacht Umgebungsvermögen, unter der das Adaptionsvermögen des Fahrers vor allem beeinflusst wird durch:

- den Abschnitt der Straße vor dem Fahrzeug, der von der eigenen Fahrzeugbeleuchtung und den umgebenden Straßenbeleuchtung, sowie
- der Helligkeit der Anzeigen und Instrumente im Fahrzeug

(siehe Seite 7 DIN EN ISO 15008, 2003)

OLED Organic Light Emitting Device : sind Bauelemente aus ultradünnen organischen Schichten die beim Anlegen einer Spannung Licht aussenden (vgl. für Bildung und Forschung, 2008)

Perzeption Vorgang der Wahrnehmung eines Gegenstandes ohne bewusstes erfassen oder Identifizieren (siehe Seite 580 BROCKHAUS, 2000)

Plasma 'Die Plasma-Technik basiert auf einem fluoreszierenden Gemisch aus den Gasen Helium, Neon und Xenon. Das Gemisch ist in Kammern eingeschlossen. . . '(siehe Digital-World, 2008). Anhand dieser Gase enstehen Plasma-Entladungen, die eine Phosphorschicht zum leuchten bringen.

Publisher-Subscriber-Konzept Das Konzept ist ein Pattern zur Infrormationsweitergabe von Zustandsänderungen. Ein ausgewählte Komponente ist der 'publisher'. Die Komponente hällt eine Methode bereit, anhand der sich eine andere Komponente, der 'subscriber', an- oder abmelden kann. Ändert der 'publisher' nun seinen Zustand, teilt er dieses Ereignis allen seinen 'subscribern' mit (vgl. Buschmann, 2008).

QT-Embedded QT ist ein Applikations-Framework mit vielen grafischen Klassen für die GUI-Entwicklung

Reaktionszeit die zwischen Reiz und Reaktion verstreichende Zeitspanne(Latenz). Einfache Reaktionen (z.B. Tastendruck auf Lichtreiz) haben eine R. von 0,15 bis 0,3 Sekunden.

Software-Engineering ‘zielorientierte Bereitstellung und systematische Verwendung von Prinzipien, Methoden und Werkzeugen für die arbeitsteilige, ingenieurmäßige Entwicklung und Anwendung von umfangreichen Softwaresystemen’(siehe Seite 36 Balzert, Heidelberg 2001)

Sonnenlicht Umgebungslicht, unter der das Adaptionsvermögen des Fahrers vor allem durch das direkte Sonnenlicht beeinflusst wird (siehe Seite 7 DIN EN ISO 15008, 2003)

Stakeholder bezeichnet alle Personen, die mit dem System in in irgend einer Beziehun zu tun haben

Storyboard konkretisierung von eninzelnen Funktionalitäten und Abläufen der Software mit dem Kunden und Benutzern.

Tag Umgebungsbedingung, unter der das Adaptionsvermögen der Fahrers vor allem durch die externe Umwelt eines bewolkten Himmels beeinflusst wird (siehe Seite 7 DIN EN ISO 15008, 2003)

TFT Dünnschichttransistor (thin-film transistor engl.)

TTCAN steht für time triggered CAN und ist eine zeitgesteuerte CAN-Architektur. TTCAN wird verwendet um Echtzeitbedingungen gerecht zu werden.

Wahlreaktion bei einer Wahlreaktion muss man nicht auf alle, sondern nur auf bestimmte Situationen reagieren

Widgets wird in dieser Arbeit als Begriff für eine grafische Anzeige-Komponente - oft auch als Anzeige-Element bezeichnet - gebraucht.

XML XML (Extensible Markup Language) ist ein simples, sehr flexibles Text-Format basierend auf SGML (ISO 8879). Entwickelt von einer Arbeitsgruppe des W3C (World Wide Web Consortium), wurde es primär für das Web konzipiert (W3C, 2008).

Index

Tabellenverzeichnis

Abbildungsverzeichnis

Versicherung über Selbstständigkeit

Hiermit versichere ich, dass ich die vorliegende Arbeit im Sinne der Prüfungsordnung nach §24(5) ohne fremde Hilfe selbstständig verfasst und nur die angegebenen Hilfsmittel benutzt habe.

Hamburg, 25. August 2008	
Ort, Datum	Unterschrift