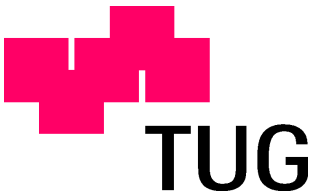
Magisterarbeit

# Drahtloses Kommunikationssystem für Diagnose und Monitoring in Kraftfahrzeugen

Norbert Thek

---

Institut für Technische Informatik
Technische Universität Graz
Vorstand: O. Univ.-Prof. Dipl.-Ing. Dr. techn. Reinhold Weiß




Begutachter: O. Univ.-Prof. Dipl.-Ing. Dr. techn. Reinhold Weiß
Betreuer: Dipl.-Ing. Dr. techn. Martin Schmid

Graz, im August 2006

# Kurzfassung

Mess- und Analyse-Systeme im automotiven Sektor, wie beispielsweise AVL-DRIVE, benötigen eine teure und arbeitsintensive Verkabelung der verschiedenen Sensoren und Monitoring Geräte. Dies ist insbesondere problematisch wenn Serienfahrzeuge analysiert werden, da bei der Verkabelung das Fahrzeug beschädigt werden kann (aufgeschnittene Tapezierung, Löcher in die Karosserie usw.) und dadurch an Wert verliert.
Ziel dieser Arbeit ist es, verschiedene drahtlose Kontakttechnologien auf ihre Nutzbarkeit im Automobilbereich zu analysieren und zu bewerten. Im Speziellen soll die Echtzeitfähigkeit und die Resistenz gegenüber Interferenzen untersucht werden. Basierend auf der ausgewählten Nahbereichsfunktechnologie wird eine neue smarte Benutzeroberfläche für AVL-DRIVE entwickelt, welche dem Benutzer auf möglichst einfache und intuitive Weise die Benutzung des AVL-DRIVE Systems während der Fahrt erlaubt. Zusätzlich wurde auch noch die Möglichkeit einer Sprachsteuerung in der Bewertung der verschiedenen Monitoring Geräte berücktsichtig und die Nutzbarkeit des neuen Benutzerinterfaces überprüft und getestet.

## Abstract

Measuring and analysis tools in the automotive sector, like AVL-DRIVE, require costly and laborious wiring of the different sensors and the monitoring devices. This is especially problematic when analyzing series-production cars because the wiring damages the car and reduces its value.
The aim of this work is to evaluate the usability of different wireless network technologies in the automotive domain to connect smart sensors or monitoring devices to the AVL-DRIVE system. A matter of particular interest is the real-time abilities and the interference resistance of the different technologies. Based on the selected transfer technology a new smart control interface for AVL-DRIVE is developed. The system shall allow the user to operate AVL-DRIVE software simply and intuitively without limiting the functionality of the program during test runs. In addition, the possibility of voice control is included in the assessment of different monitoring devices. Finally the usability of the new control interface is verified and tested.

## Danksagung

Diese Diplomarbeit wurde im Studienjahr 2005/06 am Institut für Technische Informatik an der Technischen Universität Graz durchgeführt.

Ich möchte mich bei Prof. Reinhold Weiß dafür bedanken, dass er es mir ermöglichte diese Arbeit zu verfassen. Dank auch an meinen Betreuer, Dr. Martin Schmid, für seine fachliche Betreuung und vor allem für seine Geduld. Auch bedanken möchte ich mich bei den Kollegen der Firma AVL List Abt. DV, die diese Arbeit überhaupt erst möglich gemacht haben.

Ein großes Dankeschön an meine Eltern, Geschwister und meiner Großmutter, ohne deren Unterstützung mein Studium nicht möglich gewesen wäre.
Weiters möchte ich mich bei meiner Freundin Malene für ihren Beistand während der Entstehung dieser Arbeit bedanken.

Schlussendlich bedanke ich mich auch bei Roland Hauser, Elisabeth Machacsek, Patrick Schöberl, Manfred Seper und Gerald Tausz für das gewissenhafte Korrekturlesen.

Graz, im August 2006 Norbert Alexander Thek

# Inhaltsverzeichnis

# Abbildungsverzeichnis

# Tabellenverzeichnis

# Kapitel 1

# Einleitung

## 1.1 Motivation

Kraftfahrzeuge (KFZ) werden in der heutigen Zeit mit einem hohen maschinellen Aufwand und hohem Automatisierungsgrad gefertigt. Trotz der genauen Festlegung der Eigenschaften aller Einzelkomponenten ist die Gesamteigenschaft des Produkts nicht so einfach vorauszusehen. Sehr stark trifft dies auf die Bewertung der Fahreigenschaften von KFZ zu.

Die Firma AVL List GmbH bietet ein Tool zur objektiven Bewertung der Fahrbarkeit an. Bis zu diesem Tool war das Bewerten der Fahrbarkeit rein subjektiv, abhängig von den Testfahrern der Automobilhersteller. Dieses Problem führte zur Entwicklung des Tools AVL-DRIVE$^{TM}$(kurz: DRIVE), mit dem eine „objektive“ Bewertung möglich ist.
DRIVE bewertet die Fahrbarkeit von PKW (seit 2002 auch von LKW und Busse) indem es die verschiedensten physikalischen Größen aufnimmt und ihre Auswirkungen auf eine Person bewertet. Abbildung 1.1 zeigt den prinzipiellen Aufbau des derzeitigen AVL-DRIVE-Systems

Eines der Probleme, die beim Arbeiten mit AVL-DRIVE aufkommen, ist die Komplexität der Benutzeroberfläche. Wie schon erwähnt, ist AVL-DRIVE nicht nur ein Mess- sondern auch ein Analysewerkzeug. Da der Benutzer die meiste Zeit mit der Analyse beschäftigt ist, ist das User Interface (UI) auch für diese Tätigkeit optimiert. Es wird sehr viel Information gleichzeitig dargestellt (siehe Abbildung 3.4). Dies ist für die Arbeit im Büro von Vorteil, im Feldtest wird es aber sehr umständlich, wenn eine Messung vorgenommen wird. Vor allem das Anpassen der Anwendung an ein neues Fahrzeug ist recht kompliziert, da einerseits verschiedene Fahrmanöver nötig sind (z.B. gleichmäßiges Beschleunigen, Notbremsung usw.), gleichzeitig aber das UI ständig beachtet werden muss, und zusätzlich noch einige Eingriffe nötig sind. Deswegen werden solche *Messfahrten* im Normalfall immer zu zweit durchgeführt. Aber auch bei späteren Messungen ist es notwendig, dass der Benutzer den Laptop kontrolliert und bedient. Weiters stellt ein geöffneter Laptop, auf dem DRIVE während der Autofahrt läuft, ein nicht zu unterschätzendes Sicherheitsrisiko dar. Es gibt zwar zur Zeit ein externes Display (siehe Abbildung 3.2), welches wichtige Informationen während einer Messung darstellt und dadurch den Blick des Fahrers nicht von der Fahrbahn ablenkt, dieses hat aber folgende Nachteile:

- Es unterstützt nur die Darstellung, es ist keine Interaktion möglich.
- Es existiert nur eine beschränkte Darstellungsmöglichkeit, da das Display nur monochrom arbeitet und eine geringe Auflösung hat.
- Die Anbindung erfolgt ausschließlich über Kabel.
- Es besteht eine Abhängigkeit von einem Anbieter, dadurch hat das Display einen relativ hohen Preis (vor allem im Bezug auf Preis/Leistung).

Ein weiterer Schwachpunkt des aktuellen AVL-DRIVE Systems besteht darin, dass zwar viele der Messwerte bereits direkt aus der Fahrzeugelektronik ausgelesen werden (meistens über den CAN-Bus, siehe Abschnitt 2.2.1), aber für viele Messgrößen noch zusätzliche externe Sensoren benötigt werden, da diese Messgrößen nicht erfasst werden (z.B. Schalldruck im Innenraum) bzw. nicht in der benötigten Qualität vorliegen (z.B. Beschleunigungssensoren für den Airbag arbeiten sehr grob). Da diese Sensoren aber über Kabel angebunden sind, muss bei einem Serienfahrzeug die Verkleidung abmontiert bzw. Löcher gebohrt werden, um die Leitungen zu verlegen. Wenn z.B. eine Leitung vom Motorraum in den Fahrgastraum verlegt wird, muss schlimmstenfalls sogar ein Loch in die Karosserie gebohrt werden. Dadurch können keine *normalen* Fahrzeuge (z.B. Mietfahrzeuge) verwendet werden, da diese natürlich bei solchen Eingriffen massiv an Wert verlieren.

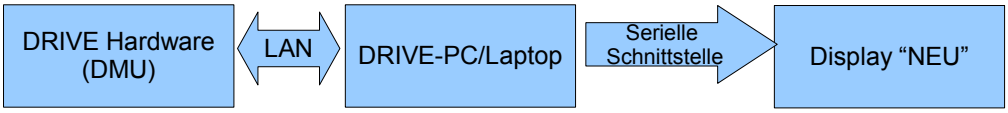



Abbildung 1.1: Übersicht des AVL-DRIVE Istzustandes

Aus diesen Gründen verfolgt diese Diplomarbeit zwei Ziele:

1. Es soll das vorhandene Display durch eine verbesserte Version ersetzt werden, die die oben genannten Nachteile nicht aufweist.
2. Es soll auch untersucht werden, ob es möglich und sinnvoll ist, die Sensoren drahtlos anzubinden.

## 1.2 Anforderungen und Aufgabenstellung

Abbildung 1.2 zeigt schematisch das abgeänderte AVL-DRIVE System als Ziel dieser Arbeit. Voraussetzung dafür ist, die drahtlose Verbindungstechnologie auf Störanfälligkeit bzw. auf ihr Echtzeitverhalten im automotiven Umfeld zu analysieren.

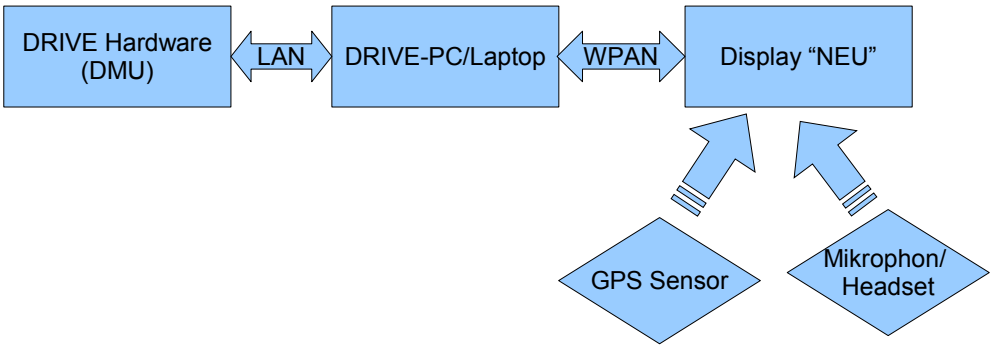



Abbildung 1.2: Anbindung des Zielsystems mittels drahtlosem Personal Area Network (WPAN) an das AVL-DRIVE System

Weiters soll die Möglichkeit einer Spracherkennung via Mikrophon am PDA untersucht werden. Diese wäre so auszulegen, dass einfache Sprachbefehle wie „Start“, „Stopp“ und „Kommentar“ zum PC geschickt werden können. Ein Beenden des Kommentars sollte selbstständig erkannt werden. Der Kommentar selbst wäre in komprimierter Form mit Zeitmarkierung abzulegen. Auf dem externen UI sind Kommunikationsbefehle des AVL-DRIVE zu implementieren und Status-Informationen sowie Messdaten und Bewertungen anzuzeigen. Für die Anzeige sollte ein generisches Framework implementiert werden, damit die Software für verschiedene Projekte nur neu konfiguriert werden muss (keine Source-Code Änderungen).

Die Verwendung von GPS soll evaluiert werden in Hinblick darauf, ob durch Sensorfusion eine Verbesserung der Messung erreicht werden kann.

Die Arbeitsinhalte umfassen folgende Schritte:

- Auswahl der Hardware-Komponenten (PDA, Headset, GPS), insbesonders auch eine Untersuchung hinsichtlich der Funktionalität im automotiven Umfeld
- Auswahl der Software-Entwicklungsumgebung
- Auswahl der Kommunikationstechnologie (Zielsystem $\leftrightarrow$ PC)
- Auswahl geeigneter Sensorik zur drahtlosen Anbindung
- Evaluierung der Kommunikation Voice, GPS, Steuerung von AVL-DRIVE
- Integration des Gesamtsystems im Testfahrzeug
- Erstellung der Dokumentation / des Benutzerhandbuchs.

### 1.2.1 Gliederung dieser Arbeit

Diese Arbeit gliedert sich in einen Theorieteil (Abschnitt 2), in welchem kurz über andere, ähnliche Produkte berichtet wird und der Begriff *Smart Devices* erklärt wird. Daran

anschließend folgt eine Übersicht über aktuelle draht- bzw. kontaktlose Verbindungstechnologien mit Diskussion und Vergleich von Eigenschaften. Es werden auch auch gängige Geräte, die für das Monitoring in Frage kommen, vorgestellt.
Im darauf folgenden Kapitel (siehe Abschnitt 3) wird eine Analyse des Ist-Zustands des AVL-DRIVE Systems vorgenommen und basierend darauf ein Lastenheft erstellt. An diesen Abschnitt anschließend wird die Auswahl der Hardware, Software usw. vorgenommen ehe das Produkt *DRIVE-Remote* im Groben vorgestellt wird.
Das darauf folgende Kapitel geht auf Implementierungsdetails ein. Danach folgt eine Evaluierung der Lösung, und schlussendlich wird ein Ausblick auf mögliche Erweiterungen des Systems gegeben.

# Kapitel 2

# Theorie

Wie bereits in der Einleitung erwähnt, soll ein neuartiges *Userinterface* zum Monitoring der Informationen dienen. Zuerst widmet sich diese Arbeit dem Begriff *Smart Devices*. Danach gibt es eine Übersicht über in Frage kommende Übertragungstechnologien bzw. Geräte für das Monitoring. Zum Schluss werden einige, aktuell am Markt befindliche Produkte, die einen ähnlichen Aufgabenbereich haben vorgestellt und auf ihre Verwendbarkeit bewertet.

## 2.1 Smart Devices

Der Begriff Smart Device wird in letzter Zeit im Bereich mobiler tragbarer Rechnersysteme oft verwendet. Beispielsweise bezeichnet Microsoft als Smart Devices sämtliche Geräte, auf denen ein *Windows CE* bzw. *Windows XP embedded* Betriebsystem läuft. Darunter fallen Pocket PCs, Handheld PCs, Smartphones, Smart Displays und andere.
Manchmal wird ein Smart Device als ein *information appliance device* bezeichnet, welches fähig ist Daten (in Form von Signalen/Messwerten, Graphiken, Animation, Audio usw.) aufzunehmen, zu bearbeiten und mit anderen Geräten und/oder Menschen zu kommunizieren. Dies unterscheidet Smart Devices von Geräten wie z.B. Digitalkameras (diese können nur aufzeichnen und dem Menschen die Daten kommunizieren), Mobiltelefone (diese können zwar kommunizieren aber Daten nicht aufzeichnen bzw. bearbeiten), TV Geräte und ähnliches.
Eine wichtige Fähigkeit eines Smart Devices besteht darin sich an die Umgebung *anzupassen*, sei es durch eine adaptive Programmierung oder durch simples Ersetzen der Software.
Zusammengefasst bestehen Smart Devices aus folgenden Komponenten (siehe Abbildung 2.1) [26]:

**Autonome Energieversorgung (Power component)** Das kann jede Art von Energieversorgung sein, die Mobilität unterstützt. Beispiele dafür sind Batterien, Solarzellen und Brennstoffzellen.

**Speicher (Memory component)** Ein Smart Device kann eigenständig Entscheidungen aufgrund der Umgebungsbedingungen treffen. Dazu müssen die möglichen Operationen in einem Speicher zur Verfügung stehen. Er wird aber auch benötigt, um Daten aufzunehmen und für später abrufbar zu halten. Physikalisch kann es flüchtiger aber auch permanenter Speicher sein.

**Recheneinheit (Processing component)** Nahe liegend benötigt ein Smart Device eine Recheneinheit, die die vorhandenen Daten bearbeiten kann. Diese ist fähig, Aufgaben durch Adaption zu lösen.

**Kommunikationsschnittstelle (Communication interface)** Einem Smart Device muss es möglich sein, mit seiner Umwelt in Verbindung zu treten. Darunter fallen nicht nur andere Smart Devices, sondern auch Menschen oder andere Computer (z.B. Server). Dies kann durch verschiedenste Kommunikationstechnologien erfolgen (digital aber auch analog, drahtgebunden oder drahtlos, usw.).

Ein Spezialfall von Smart Devices sind Smarte Sensoren. Ein Smarter Sensor besteht aus einem Sensor/Actuator (digital oder analog), einem Microcontroller und einem Kommunikationsinterface, welche in einer Einheit integriert sind.
Vom „Institute of Electrical and Electronics Engineers“ (IEEE) gibt es Bestrebungen einen gemeinsamen Standard für solche Sensoren (in Englisch „Smart Transducer“) zu definieren. Dafür gibt es die Standard Familie 1451, wobei sich die Gruppe 5[39] mit der Standardisierung von drahtlosen Smarten Sensoren befasst. Diese Spezifikation ist derzeit noch nicht verfügbar, soll aber auf bestehende Übertragungstechnologien wie 802.11(WLAN), 802.15.1 (Bluetooth) und 802.15.4 (ZigBee) aufbauen.
Im nächsten Abschnitt wird der Fokus auf drahtlose Netzwerke gelegt und deren Eignung für Smart Devices untersucht.

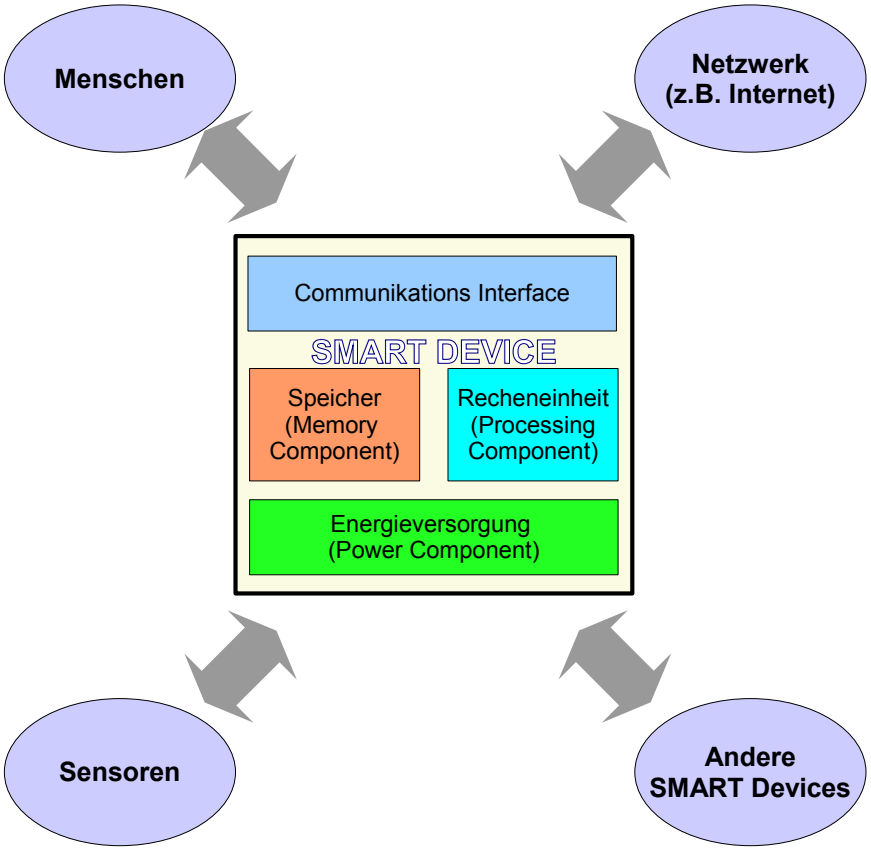



Abbildung 2.1: Struktur von Smart Devices

## 2.2 Drahtlose Netzwerke

Um Sensoren bzw. Monitore zu verbinden, bieten sich verschiedene Technologien an. Im Zuge dieser Diplomarbeit werden dafür bestehende Technologien verglichen. Zusätzlich wird zuerst noch das „Controller Area Network“ als Beispiel für eine weit verbreitete Übertagungstechnologie im KFZ-Bereich vorgestellt, um sich ein Bild von den Anforderungen im automotiven Bereich bilden zu können.
Folgende Standards drahtloser Netzwerke werden in der Folge untersucht und verglichen:

- Bluetooth (auch unter Berücksichtigung der neuen Version 2.0)
- WLAN IEEE 802.11
- Zigbee
- Wireless TTP/A
- Wireless Control Area Network (W-CAN)
- Sensor Motes
- Infrarot (IrDA)
- Near Field Communication (NFC)

### 2.2.1 Controller Area Network (CAN)

Das Controller Area Network (CAN bzw. auch CAN-Bus genannt) ist ein serielles, asynchrones Bussystem, das speziell für die Anwendung in Kfz entwickelt wurde. Es wird aber auch immer öfter in anderen Bereichen (Industrieautomatisierung, Haustechnik usw.) eingesetzt. Entwickelt wurde dieses System von Bosch für die Vernetzung von Steuergeräten in Automobilen, um die immer umfangreicheren Kabelbäume in den Fahrzeugen zu reduzieren und um damit Gewicht und Kosten zu sparen. Mit der Entwicklung wurde 1983 begonnen, 1986 wurde es offiziell vorgestellt und im Jahre 1993 von der ISO unter der Norm 11898 standardisiert.
CAN benutzt zur Verhinderung von Kollisionen auf dem Bus CSMA/CA (Carrier Sense Multiple Access/Collision Avoidance).
Eine spezielle Eigenschaft des CAN Busses ist die *Multi-Master Fähigkeit*, wodurch jeder einzelne Knoten eine Kommunikation einleiten kann. Dies ist auch aus Sicherheitsgründen relevant, denn der Ausfall eines Knotens hat keinen Einfluss auf den Rest des Busses. Zusätzlich ergibt sich dadurch noch eine ereignisgesteuerte Kommunikation, denn jeder Teilnehmer kann selbst entscheiden, wann er eine Übertragung durchführen möchte.
Die übertragenen Nachrichten sind mit einem Objektidentifier versehen, welcher die Nachricht und *nicht* den Absender kennzeichnet. Jeder Teilnehmer kann nun entscheiden, ob die Nachricht für ihn relevant ist oder nicht.
Für eine bessere Störresistenz gegenüber elektromagnetischen Einflüssen werden zumeist Kupferleitungen mit Differenzsignalen verwendet.

**Eckdaten von CAN**

- Datenübertragungsraten von bis zu 1 MBit/s
- Reichweite (max. Leitungslänge) abhängig von der Übertragungsrate
  - 40 m bei 1Mbit/s
  - 100 m bei 500kbit/s
  - 500 m bei 125kbit/s
- maximale Teilnehmeranzahl abhängig von den verwendeten Bustreiberbausteinen (Zur Zeit sind über 100 Teilnehmer möglich.).

### 2.2.2 WLAN IEEE 802.11

IEEE 802.11a/b ist die Technologie, die hinter dem allgemein bekannten Begriff *WLAN* steht. Genauer stehen für die Bezeichnung 802.11 mehrere Standards, die vom IEEE herausgegeben bzw. spezifiziert wurden.
Ziel der Standards war es, einen Ersatz für die Verkabelung von Computern zu schaffen[76]. Die erste Version wurde bereits 1997[62] veröffentlicht, konnte sich damals aber nicht durchsetzen. Erst mit der Version 802.11b (bzw. 802.11a in den USA) gelang der Durchbruch. Eine Übersicht über die vorhandenen Standards findet sich in der Tabelle 2.1:

| **Arbeitsgruppe** | **Arbeitsgebiet** |
|---|---|
| 802.11a | 54-Mbit/s-WLAN im 5 GHz-Band |
| 802.11b | 11-Mbit/s-WLAN im 2,4-GHz-Band |
| 802.11c | Wireless Bridging |
| 802.11d | "World Mode", regionspezifische Anpassung |
| 802.11e | QoS- und Streaming-Erweiterung für 802.11a/g/h |
| 802.11f | Roaming für 802.11a/g/h (Inter Access Point Protocol IAPP) |
| 802.11g | 54-Mbit/s-WLAN im 2,4-GHz-Band |
| 802.11h | 54-Mbit/s-WLAN im 5-GHz-Band mit DFS und TPC |
| 802.11i | Authentifizierung/Verschlüsselung für 802.11a/b/g/h (AES, 802.1x) |

Tabelle 2.1: Überblick über IEEE 802.11[58]

**Eigenschaften von WLAN nach IEEE 802.11**

Die Standards 802.11b und g verwenden für die Übertragung das freie ISM-Band (Industrial, Scientific, and Medical Band) bei 2400MHz, welches weltweit lizenzfrei genutzt werden darf, solange Auflagen bezüglich Sendeleistung und Störung benachbarter Frequenzbereiche eingehalten werden. Die Standards a bzw. h arbeiten im 5 GHz Bereich. In diesem Bereich sind strengere Auflagen gegeben. Deswegen darf 802.11a in Europa (Deutschland, Österreich, usw.) nicht verwendet werden. Erst die Erweiterung 802.11h bietet zusätzliche Spezifikationen, um zu gewährleisten, dass andere Dienste im gleichen Frequenzband nicht gestört werden. Diese sind Dynamic Frequency Selection (DFS) und Transmission Power

Control(TPC)). Im 2,4 GHz Bereich ist eine maximale Sendeleistung von 100mW EIRP[1] erlaubt, im 5GHz Band eine Leistung bis zu 1W EIRP.
Die Reichweite der Geräte wird von den meisten Herstellern mit 300 m angegeben, wobei dieser Wert nur bei optimalen Bedingungen, wie einer freien Sicht zwischen Sender und Empfänger, erreicht werden. Real sind Werte zwischen 30 m bis 100 m zu erwarten. Vor allem neuere Geräte erreichen höhere Reichweiten durch verbesserte Antennen.
Datenübertragungsraten der verschiedenen Standards (auch von noch nicht verabschiedeten) sind in Tabelle 2.2 zu finden.

| **Standard** | **Datenübertragunsrate** |
|---|---|
| IEEE 802.11 | 2 Mbps maximal |
| IEEE 802.11a | 54 Mbps maximal |
| IEEE 802.11b | 11 Mbps maximal |
| IEEE 802.11g | 54 Mbps maximal |
| IEEE 802.11n | 540 Mbps max. (Verabschiedung des Standards voraussichtlich erst Ende 2007) |
| 802.11b+ | 44 Mbps max. (nicht standardisiert, herstellergebunden) |
| 802.11a+/g+ | 108/125 Mbps max. (nicht standardisiert, herstellergebunden) |
| IEEE 802.16d | 75 Mbps maximal |
| IEEE 802.16e | 30 Mbps maximal |

Tabelle 2.2: Datenübertragungsraten

Diese Übertragungsraten sind aber nur für den optimalen Fall gültig (keine Störungen, kurze Distanz zwischen Sender und Empfänger). Die Transferraten lassen rapide nach, wenn keine optimalen Bedingungen mehr gegeben sind.
Der Medienzugriff wird über CSMA/CA geregelt.
Der Stromverbrauch wurde erst in letzter Zeit stark reduziert. Ältere Geräte wiesen Leistungsaufnahmen von über 1 Watt [29, S.40] auf, neuere Geräte, wie z.B. die WLAN Komponenten der Centrino$^{TM}$Plattform von Intel, kommen bereits mit einer Maximalleistung von 500 mW aus (typische Werte: Transmit: 300mW, Receive:170 mW, Sleep: 10mW[17]).

### Vorteile von IEEE 802.11

- Es ist auf dem Markt sehr stark verbreitet.
- Hohe Übertragungsraten sind möglich.
- Standard-Hardware ist verfügbar.
- Relativ hohe Reichweiten sind erreichbar.

[1] Effective isotropic radiated power, EIRP ist ein Wert um Richtantennen zu charakterisieren. EIRP gibt an, mit welcher Sendeleistung eine in alle Raumrichtungen gleich(isotrop) abstrahlende Antenne versorgt werden müsste, um in der Ferne die selbe Feldstärke zu erreichen wie eine Richtantenne in ihrer Hauptsenderichtung. Die Einheit ist $mW$ bzw. $dBm$ bzw. $dBW$ [48, Seite 60]

#### Nachteile

- Der Stromverbrauch ist immer noch relativ hoch.
- Es besteht keinerlei Quality of Service auf den unteren Schichten (erst die Erweiterung 802.11e unterstützt dies, dieser Standard wurde aber erst im September 2005 angenommen[62], ist also noch sehr neu)

### 2.2.3 Bluetooth

Der Industriestandard Bluetooth (IEEE 802.15.1)[55] dient zur drahtlosen Vernetzung von Geräten über kurze Distanz. Bluetooth bietet eine drahtlose Schnittstelle, über die sowohl mobile Kleingeräte (wie Mobiltelefone und PDAs), als auch Computer und Peripheriegeräte miteinander kommunizieren können.

Ursprünglich von Ericsson als Ersatz für die Kabelanbindung von Mobiltelefonen und Zusatzgeräten entwickelt, wurde 1998 die Entwicklung von der Bluetooth Special Interest Group (SIG)[2] übernommen, und Bluetooth als multifunktionelles *Wireless Personal Area Network* (WPAN) weiterentwickelt. Der Standard ist erst später von der IEEE für WPANs als IEEE 802.15.1 adaptiert worden.
Ziel der Entwickler war es, verschiedenste Geräte ohne bzw. mit wenig Konfiguration durch den Benutzer miteinander kommunizieren zu lassen. Weiterhin sollte der Transceiver billig, robust gegen Störungen und für multimediale Anwendungen verwendbar sein.

#### Eigenschaften von Bluetooth

Ein Bluetooth-Netz besteht aus einem Master und bis zu sieben weiteren aktiven Geräten als Slaves. Eine solche Gruppe von bis zu acht Geräten wird *Piconet* genannt. Es ist auch möglich, mehrere solcher Piconets miteinander zu verbinden. Dabei agiert ein Gerät als *Bridge* zwischen zwei Netzen. Ein Piconet kann insgesamt bis zu 255 Geräte enthalten, wobei aber nur acht gleichzeitig aktiv sein dürfen.
Der Master ist für den Ablauf der Kommunikation zuständig und vergibt Zeitslots für die Kommunikation an die Slaves.
Auch Bluetooth arbeitet im lizenzfreien ISM-Band, um weltweit zulassungsfrei betrieben werden zu können. Da sehr viele andere Geräte (WLAN, Mikrowellenherde, Fernsteuerungen, usw.) in diesem Bereich senden, musste Bluetooth gegen Störungen geschützt werden. Dafür wird ein Frequenzsprungverfahren verwendet, bei dem das Frequenzband in 79 Stufen eingeteilt und 1600 mal pro Sekunde gewechselt werden.
Im derzeit verbreiteten Standard 1.1 (bzw. auch 1.2) sind Datenübertragungsraten bis zu 723.1 kbit/s möglich. In der Version 2.0 mit Enhanced Data Rate (EDR) sind bis zu 2.1 Mbit/s erreichbar. Zukünftige Versionen sollen Transferraten von über 100 MBit/s erreichen, wobei dies zur Zeit nur unter Laborbedingungen erreicht wird.
Die Spezifikation sieht drei verschiedene Klassen von Geräten vor, die mit unterschiedlicher Sendeleistung arbeiten.

Die in Tabelle 2.3 angegebenen Reichweiten sind nur Richtwerte, bei geschickter Antennenwahl und freier Sichtverbindung sind auch größere Werte erzielbar[78].

[2] https://www.bluetooth.org/

| Klasse | Sendeleistung | Reichweite |
|---|---|---|
| I | 100 mW | ≈ 100 m |
| II | 2.5 mW | ≈ 20 m |
| III | 1 mW | ≈ 10 m |

Tabelle 2.3: Bluetooth Klassen und die damit erzielbaren Reichweiten

Um unterschiedlichste Geräte miteinander zu verbinden, gibt es die sogenannten Bluetooth-Profile. Damit kann festgestellt werden, welche Funktionalität ein Gerät anbietet.

**Vorteile von Bluetooth**

- Eine weite Verbreitung ist bereits vorhanden.
- Es verfügt über eine niedrige Sendeleistung.
- Standard Hardware ist verfügbar.
- Der Energieverbrauch ist sehr gering (vor allem bei Verwendung von Klasse II oder III).
- Ein Quality of Service ist bereits auf einer tiefen Protokollebene (L2CAP) verfügbar.

**Nachteile von Bluetooth**

- Die Bandbreite ist gering (obwohl durch EDR bis zu 2 MBit erreicht werden können).
- Ein Master kann nur eine begrenzte Anzahl von anderen Geräten (Nodes) adressieren (maximal acht aktive).

### 2.2.4 ZigBee

ZigBee ist ein Standard, der von der ZigBee Alliance definiert wurde, um Haushaltsgeräte und Sensoren miteinander zu verknüpfen. Der Standard zielt auf einen sehr niedrigen Energieverbrauch, geringe Komplexität (in Hardware und Software), hohe Zuverlässigkeit und geringeren Preis als vergleichbare WPANs (wie z.B. Bluetooth) ab.
Der Standard definiert eine höhere Protokollebene und basiert auf den Spezifikationen von IEEE 802.15.4.

**Eigenschaften[73]**

Auch ZigBee verwendet für die Datenübertragung das 2.4 GHz Frequenzband (weltweit frei verfügbar), zusätzlich aber auch das 915 MHz Band (Nordamerika) und 868 MHz Band (Europa). Die Reichweite liegt im Bereich von 10-75 m.
Die Datenübertragungsrate hängt vom Frequenzband ab. Sie beträgt 250 kb/s bei 2.4 GHz, 40 kb/s bei 915 MHz und 20 kb/s bei 868 MHz.
Grundsätzlich verwendet ZigBee für den Medienzugriff CSMA/CA, es gibt aber „Beacon-Netzwerke“, in denen die einzelnen Geräte nur zu bestimmten Zeiten aufwachen und senden (wichtig für die Verwendung in Realtime-Anwendungen mit garantierter Reaktionszeit).

Eine der wichtigsten Punkte in der Entwicklung war ein möglichst geringer Energieverbrauch. Dadurch ist es möglich, ZigBee-Geräte Wochen bis Monate ohne Aufladen bzw. Batteriewechsel zu betreiben.
Es gibt zwei Arten von Geräten in ZigBee-Netzen.

**Reduced Funktion Device (RFD) oder End Device:** Hier handelt es sich um einfache Geräte wie z.B. Lichtschalter, die nur einen geringen Teil des Protokolls implementieren. Sie können sich nur an einem Router (Coordinater) anmelden bzw. nur mit diesem kommunizieren und funktionieren daher auch nur in einem Netzwerk mit Stern-Topologie.

**Full Function Device:** Diese funktionieren in allen Netztopologien. Ein FFD kann die Rolle eines Koordinators übernehmen und mit beliebigen anderen Geräten kommunizieren.

Ein ZigBee Netzwerk kann aus bis zu 65.000 Knoten bestehen und verschiedenste Topologien aufweisen (siehe Abbildung 2.2).

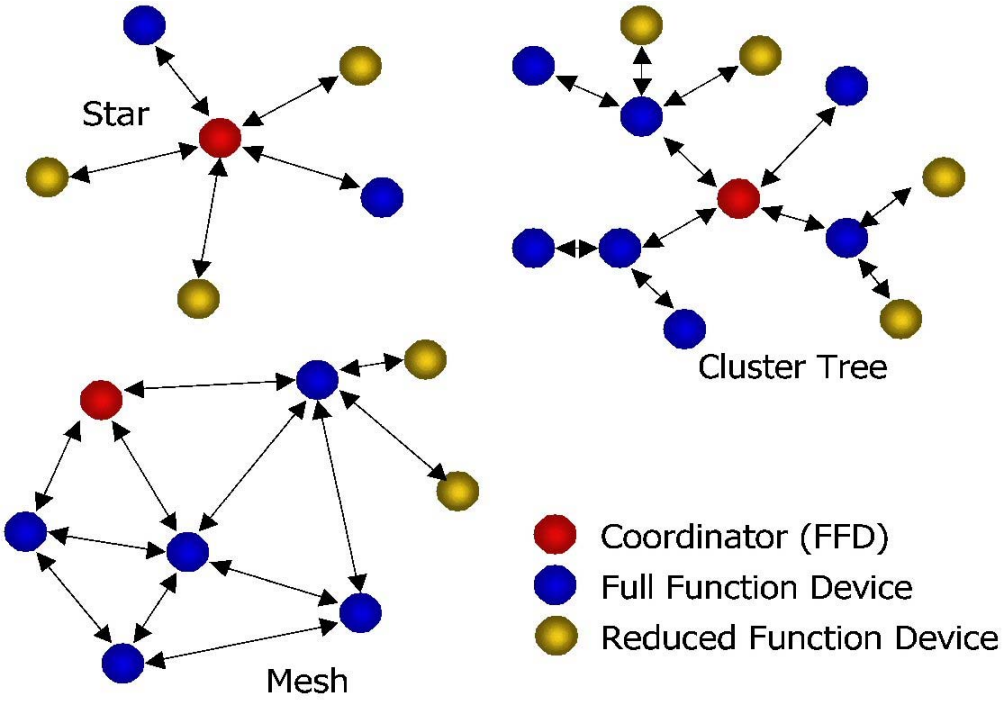



Abbildung 2.2: Mögliche ZigBee Netztopologien[50]

**Vorteile von ZigBee**

- Es besitzt einen sehr geringen Energieverbrauch.
- Es ist relativ unempfindlich gegenüber Störungen.

- ZigBee erlaubt Echtzeitanwendungen durch Verwendung von Zeitschlitzen (*Beacon Netzwerke*).
- Günstige Hardware ist bereits vorhanden. System-on-Chip Bauteile (Microcontroller mit Zigbee Transceiver) gibt es bereits für unter 4 USD[5].
- Eine große Anzahl von Knoten (> 60.000) ist möglich.

**Nachteile von ZigBee**

- Es verfügt nur über eine geringe Datenübertratungsrate.
- Der Standard ist noch sehr neu, auch noch nicht sehr weit verbreitet, daher gibt es wenig Erfahrungsberichte.

### 2.2.5 Wireless TTP/A

TTP/A (**T**ime-**t**riggered **p**rotocol Class **A**) ist ein echtzeitfähiges Sensor-Actuator-Bussystem. Es wurde für günstige Feldbussystemanwendungen entwickelt und kann bereits mit einem handelsüblichen 8-bit Microcontroller mit einem Standard UART (Universal Asynchronous Receiver Transmitter) implementiert werden. Das Protokoll wird oft für die Kommunikation in einem „Smart Transducer“ Cluster (Smartes Sensor Netz) verwendet.
TTP/A ist ein Master-Slave System, das feste Antwortzeiten garantiert. Es garantiert aber keine fehlerfreie Übertragung, weshalb sich das Protokoll im Wesentlichen nur zur Übertragung von Sensordaten mit großen Abtastraten eignet, da sich hier die einzelnen Fehler im Mittel nicht auswirken. Ein Netz kann aus bis zu 255 Slave Knoten (Nodes) bestehen, wobei jeder Knoten eine eindeutige Identifikationsnummer aufweist.
Wireless TTP/A ist eine Erweiterung von TTP/A, die eine drahtlose, echtzeitfähige Verbindung von einzelnen TTP/A Netzen ermöglicht. Die drahtlose Verbindung erfolgt wieder mittels Master/Slave Prinzip. Der Master stellt die gemeinsame Zeitbasis aller Nodes zur Verfügung und kontrolliert die Kommunikation zwischen den einzelnen Nodes[37].
Dieses Protokoll ist noch sehr neu und wird bis jetzt nur im akademischen Bereich eingesetzt. Basierend auf Standardkomponenten (wie z.B. einen Atmel ATmega 128 Microcontroller[3]) ist diese Technologie sehr kostengünstig. Die verwendeten Transceiver Module können eine Datentransferrate von bis zu 64 kBit/s erreichen (bei 433,92 MHz, einem europäischen lizenzfreien Frequenzband). Es können aber auch andere Übertragungsmodule verwendet werden, denn Wireless TTP/A definiert nur den logischen Teil des Protokolls.

**Vorteile von Wireless TTP/A**

- Es verfügt über einen geringen Energieverbrauch.
- Wireless TTP/A unterstützt Echtzeitanwendungen.
- Günstige Hardware ist verfügbar (durch Verwendung von Standardkomponenten).
- Es bestehen Auswahlmöglichkeiten für verschiedene Hardwaremodule, dadurch ist die Technologie sehr gut an verschiedenste Anforderungen anpassbar.

[3] http://www.atmel.fi/dyn/products/product_card.asp?part_id=2018

#### Nachteile von Wireless TTP/A

- Die Datenübertragungsrate ist relativ gering.
- Es ist kein wirklicher Standard, sondern als Lösungsansatz für drahtlose Echtzeitübertragung zu sehen.
  Dadurch ist keine Standardlösung mit Protokollstack verfügbar.

### 2.2.6 Sensor Motes

Sensor Motes[4] sind im eigentlichen Sinn keine Übertragunstechnologie. Sie stellen vielmehr einen Überbegriff für einzelne Sensoren in einen *Smart Dust Network* dar. Die Idee dahinter ist, dass ein ganzer Verband von winzigen Computern und Sensoren ein Ad-hoc Netzwerk bildet und damit verschiedenste Aufgaben erfüllen kann, wie z.B. die Überwachung eines Gebiets gegen Eindringlinge oder auch Verkehrsüberwachung.
Sensor Motes gehören zur Klasse der Smarten Sensoren (siehe Abschnitt 2.2.5 bzw. auch Abschnitt 2.1).
Die bekannteste Implementierung von Sensor Motes stammt von der Universität Berkley aus den USA; hier wurden auch die MICA Motes (siehe Abbildung 2.3) entwickelt, die käuflich von der Firma Crossbow erwerbbar sind[24].

Abbildung 2.3: MICA2 Mote der Firma Crossbow[24]

#### Netzwerk Technologie der Motes

Es gibt zur Zeit keinen einheitlichen Kommunikationsstandard für die verschiedenen Motes. Jeder Hersteller verwendet seine eigenen Chips bzw. Protokolle. Grundsätzlich haben aber alle das gleiche Ziel: möglichst energiesparend zu kommunizieren.
Als Beispiel dienen die Produkte der Firma Crossbow, welche im akademischen Bereich eine starke Verbreitung haben.
Aktuell bietet die Firma Module der Typen MICA2, MICA2DOT und MICAz an, wobei MICAz für die Kommunikation das bereits besprochene ZigBee Protokoll verwendet (siehe

[4](engl.) Mote = Staubkörnchen

Abschnitt 2.2.4). Zusätzlich bietet Crossbow noch integrierte Versionen von den MICA2 Radio/Prozessor Modulen an, um eigene Motes zu erstellen, die dann auch mit den MICA Motes zusammenarbeiten. MICA2DOT ist eine verkleinerte Version der MICA2 Motes, die von der Verbindungstechnik 100% kompatibel mit diesen ist, aber weniger Sensoren und kürzere Laufzeit (wegen kleinerer Batterien) hat. Tabelle 2.4 zeigt beispielgebend die Eigenschaften der Funkschnittstelle der MICA2 Motes.

| **Prozessor/Radio Board** | **MPR400CB** | **MPR410CB** | **MPR420CB** |
|---|---|---|---|
| **Frequenz** | 868/916 MHz | 433 MHz | 315 MHz |
| **Anzahl Kanäle** | 4/50 | 4 | 5 |
| **Übertragungsrate** | 38.4 Kbaud | 38.4 Kbaud | 38.4 Kbaud |
| **Empfangsempfindlichkeit** | -98 dBm | -101 dBm | -101 dBm |
| **Reichweite max.** | 150 m | 300m | 300m |
| **benötigter Strom** | | | |
| **Max.** | 27 mA | 25 mA | 25 mA |
| **Empfangzustand** | 10 mA | 8 mA | 8 mA |
| **Sleep-Mode** | $\langle$ 1 $\mu$A | $\langle$ 1 $\mu$A | $\langle$ 1 $\mu$A |

Tabelle 2.4: MICA2: Technische Spezifikationen der Funkschnittstelle[25]

### 2.2.7 IrDA

IrDA steht für *Infrared Data Association*[6], welche physisch und logisch ein Kommunikationsprotokoll für Infrarot[5] Schnittstellen zum Datenaustausch beschreibt. Der Standard ist für Entfernungen bis ca. 1 Meter und in der neuesten Version (IrDA 1.1), für Übertragungsraten bis 16 Mbit/s spezifiziert.
IrDA wurde - wie Bluetooth - für das Anwendungsgebiet von PAN entwickelt und enthält deswegen auch Spezifikationen für den Austausch von Business Cards (vCard), IrOBEX [6],IrLAN [7] und vieles mehr.

#### Vorteile von IrDA

- IrDA ist sehr stark verbreitet.
  2001 waren über 240 Millionen verkaufte Produkte im Umlauf[75]. Faktisch jeder PDA und jedes Notebook hat eine solche Schnittstelle eingebaut. Auch sehr viele Mobiltelefone unterstützen IrDA.
- IrDA ist sehr günstig.
- Es ist eine etablierte Technik (erste Version bereits 1993).
- IrDA weist einen äußerst geringen Energieverbrauch auf.

[5] Als Infrarotes Licht wird elektromagnetische Strahlung mit einer Wellenlänge von 780 nm bis 1mm bezeichnet (für IrDA kommt die Bereich von 850 nm bis 900nm zum Einsatz)
[6] Infrared OBject EXchange Protokoll[75, Seite 303]
[7] LAN zugriff über Infrarot

**Nachteile von IrDA**

- Es hat eine sehr kurze Reichweite (ca. 1m).
- Es benötigt eine Sichtverbindung zwischen den beiden Kommunikationspartnern (dadurch ergibt sich aber auch der Vorteil, dass sehr leicht kontrolliert werden kann, welche Geräte miteinander kommunizieren, was bei funkbasierenden Technologien aufwendiger ist).

### 2.2.8 Near Field Communication

Near Field Communication (NFC) ist eine standardisierte (ECMA 340[30] bzw. ISO 18092 [1]), drahtlose Verbindungstechnik für kurze Strecken. Ziel ist es, eine einfach zu handhabende und sichere Zwei-Wege-Kommunikation zwischen Geräten zu erlauben, um Benutzern kontaktlose Transaktionen oder Zugriffe auf digitale Inhalte zu ermöglichen. Weiters soll es als *Türöffner* für andere Verbindungstechnologien wie z.B. Bluetooth (siehe Abschnitt 2.2.3) oder WLAN (siehe Abschnitt 2.2.2) dienen. Die Funktionsweise wird anhand des folgenden Beispiels erklärt[47]:
Statt eines aufwändigen Pairing-Verfahrens mit diversen Einzelschritten hält man etwa einen Bluetooth-PDA und ein passendes Handy ganz einfach für einen Moment unmittelbar aneinander. Die Geräte identifizieren sich per NFC und starten die Datenverbindung per Bluetooth. Und das ohne die sonst übliche Funkraum-Suche, Gerätewahl, Dienstwahl und Passwortübergabe.

Die NFC Technologie wurde gemeinsam von Philips und Sony entwickelt, basiert auf *Mifare* und *FeliCa* RFID[8] Technologien und soll mit diesen weitgehend kompatibel sein (Mifare wurde von Philips, FeliCa von Sony entwickelt). NFC arbeitet in einem Frequenzbereich von 13,56 MHz, bietet eine Reichweite von max. 20 Zentimeter und eine Datenübertragungsrate von bis zu 424 kBit/s[30]. Eine Erweiterung auf 1 MBit/s ist in Vorbereitung[56]. Darüber hinaus ist NFC für einen geringen Stromverbrauch konzipiert (unter anderem begründet durch die sehr kurze Reichweite). Ein NFC Modul von Philips hat laut Spezifikation eine maximalen Stromaufnahme von 100mA[49]. Die weitere Entwicklung wird zur Zeit vom NFC Forum[57] vorangetrieben, welches sich auch für die Verbreitung dieser Technologie einsetzt. Dem NFC Forum gehören neben den Gründungsmitgliedern Philips, Sony und Nokia auch viele andere größere Unternehmen an (Microsoft, VISA, MasterCard, Samsung, um nur einige zu nennen).

**Vorteile von NFC**

- Es ist relativ Kostengünstig.
- Es gibt eine große Unterstützung durch verschiedenste Firmen (siehe NFC-Forum[57]).
- NFC ist von ECMA bzw. ISO standardisiert.

[8] Radio Frequency Identification (RFID), in der deutschen Fachliteratur gelegentlich Funkerkennung, ist eine Methode, um Daten auf einem Transponder berührungslos und ohne Sichtkontakt lesen und speichern zu können. Dieser Transponder kann an Objekten angebracht werden, welche dann anhand der darauf gespeicherten Daten automatisch und schnell identifiziert und lokalisiert werden können.

- Es besitzt nur einen geringer Energieverbrauch.

**Nachteile von NFC**

- Die Reichweite ist sehr beschränkt.
- Es ist noch nicht sehr weit verbreitet, da es relativ neu ist.
- Das Protokoll ist nicht für eine ständige Verbindung ausgelegt.
- Die Datenübertragungsraten sind gering (schnellere Spezifikationen noch nicht verfügbar).

### 2.2.9 Wireless Control Area Network

Wie auch bei den Sensor Motes (siehe Abschnitt 2.2.6) ist das Wireless Control Area Network (WCAN) kein Standard, sondern steht als Überbegriff für verschiedene Wege, den für drahtgebunde Übertragung definierten CAN Standard, auch drahtlos verwenden zu können. Forschung in diesem Bereich gib es bereits seit den 90er Jahren des letzten Jahrhunderts (siehe [36] oder [3]) in der es unter anderem darum ging, ob das Übertragungsprotokoll auf drahtlose Verbindungen angewendet werden kann bzw. welches physikalische Protokoll man verwenden sollte). Erwerbbare Produkte gibt es erst seit kurzem. Da es kein Standard ist, haben verschiedenste Anbieter unterschiedlichste Lösungen dafür gefunden. Beispielsweise gibt es von der Firma Matric eine „CAN-Bridge“[53] (siehe Abbildung 2.4), die es erlaubt, die Reichweite von CAN (normalerweise 40 m) zu erhöhen. Damit können Transferraten bis zu 500 kb/s erreicht werden. Diese *Bridges* sind aber nur für eine Punkt-zu-Punkt Verbindung geeignet, aber nicht um mehrere Sensoren drahtlos anzubinden.
Ein anderes Produkt bietet die Firma *Automation Artisans* an. „CANRF“[40] ist ein Treiber Baustein, der an Standard CAN Controller angeschlossen werden kann. Er bietet eine Transferrate von bis zu 20 kbit/s an und ist für das Anbinden von einzelnen Sensoren ausgelegt.

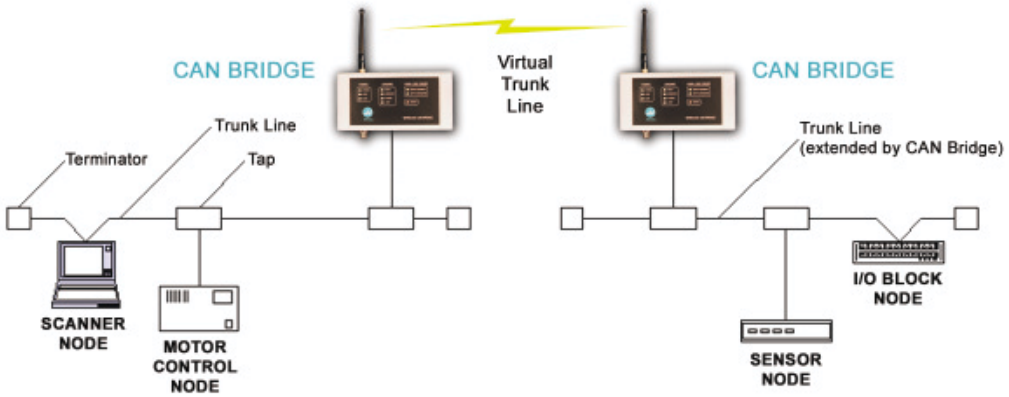



Abbildung 2.4: CAN-Bridge Anwendungsdiagram von Matric[53]

### 2.2.10 Vergleich der vorgestellten drahtlosen Übertragungstechnologien

In Tabelle 2.5 sind die wichtigsten technischen Eigenschaften der vorgestellten Übertragungstechnologien zusammengefasst. Tabelle 2.6 enthält eine Übersicht über vorhandene Softwareunterstützung sowie Kosten.

| | **Transferr. (netto)** | **Reichweite** | **Störres. geg. Interferenzen** | **Anzahl Geräte** | **QoS** |
|---|---|---|---|---|---|
| **Bluetooth** | - 2,1MBit/s | ca. 100 m | Gut, aber 2,4 Ghz Bereich | 7 aktive/ Master | Ja |
| **WLAN 802.11** | - 54 Mbit/s (über 100 in Vorber.) | - 300 m | Bedingt 2,4 Ghz Bereich | 79 Channels | Nein (nicht Low-Level) |
| **Zigbee** | - 250 kBit/s | - 75 m | Gut, altern. Bänder | - 16 Channels | Ja |
| **Wireless TTP/A** | - 10 kBit/s | - 200m | Umfangreich, altern. Bänder | Keine Angaben (theoretisch bis 65000) | Ja |
| **Wireless CAN** | Je nach Anbieter - 500 Kbits/s | Je nach Anbieter - 150 m | Je nach Anbieter | Je nach Anbieter (16) | Ja |
| **Smart Sensor Netw. Motes** | - 38 Kbit/s | - 300 m | Bedingt | -100 | Nein |
| **IrDA** | - 4 Mbit/s | - 1m | Gut da kein RF sondern Licht | 1 | Nein |
| **Near Field Comm. (NFC)** | - 424 kBit/s | Einige cm | Schlecht | 1 | Nein |

Tabelle 2.5: Übersicht der technischen Eigenschaften verschiedener kontaktloser Übertragungstechnologien.

| | **Software Unterstützung** | **Verfügbarkeit** | **Kosten** |
|---|---|---|---|
| **Bluetooth** | Vorhanden (bzw. Sockets verwendbar) | Sehr gut | günstige Hardware ca. 10 USD pro Einheit |
| **WLAN 802.11** | Standard Sockets | Sehr gut | etwas teurer als Bluetooth ca. 20 USD pro Einheit |
| **Zigbee** | Protokoll spezifiziert,aber aber extra zu kaufen | Gut, noch nicht weit verbreitet | günstig $<$ 5 USD pro Einheit |
| **Wireless TTP/A** | Nicht vorhanden, selbst zu entwickeln | Kein Produkt „von der Stange“, aber Verwendung von Standardkomponenten | Verwendetes RF-Modul BIM-433 Entwicklungsset $>$ 100 USD |
| **Wireless CAN** | Standard CAN (über Zusatz Software) | Verschiedene Anbieter, aber properitäre Lösungen | Bsp. CANRF Modul 89 USD |
| **Smart Sensor Netw. Motes** | Nicht vorhanden, selbst zu entwickeln | Mehrere Anbieter | Bsp. kompl. MICA2 Mote ca. 170 USD |
| **IrDA** | Gut spezifiziert (aber nur Punkt-zu-Punkt Kommunikation) | Sehr gut | sehr günstig $<$ 1 USD |
| **Near Field Comm.(NFC)** | Noch nicht vorhanden aber spezifiziert | Gut | günstig $<$ 5 USD pro Einheit |

Tabelle 2.6: Übersicht über Softwareunterstützung und Kosten der verschiedenen kontaktlosen Übertragungstechnologien

## 2.3 Mobile Devices

Durch die im Abschnitt 1.2 bereits vorgestellten Anforderungen an das neue Display ist es notwendig die Klasse mobiler Rechnersysteme näher zu untersuchen. Hier wird eine Übersicht über verschiedene mobile Devices gegeben, die für das neue externe Userinterface eingesetzt werden können bzw. als Ersatz für das derzeitige AVL-DRIVE Display dienen können.

### 2.3.1 Handheld Computers und Personal Digital Assistant

Unter einem Personal Digital Assistant (PDA) wird üblicherweise ein kleines tragbares Gerät verstanden, welches unmittelbar nach dem Einschalten zur Verfügung steht und mit verschiedensten Personal Information Management (PIM) Anwendungen ausgestattet ist. Unter PIM-Software ist eine Software zu verstehen, die Menschen beim Verwalten von persönlichen Daten helfen soll. Typischerweise fallen darunter Kontaktdaten, Termine, Aufgaben, usw.
PDAs gibt es in den verschiedensten Formen und Austattungen. Am meisten verbreitet sind Geräte mit Touchscreen und einigen Zusatztasten, wobei der größte Teil das Display einnimmt. Die meisten dieser Geräte basieren auf einem der beiden folgenden Betriebssystemen (es gibt zwar mehrere Plattformen, die restlichen Betriebssysteme haben aber am Markt faktisch keine Bedeutung [75, Seite 29 u. 34]):

- PalmOS
- Windows CE

Die Unterschiede zwischen den beiden Betriebssystemwelten verschwinden immer mehr. Grundsätzlich kann man sagen, dass die PALM OS Geräte billiger sind und mit leistungsschwächerer Hardware auskommen, Windows CE Geräte aber bessere Multimediafähigkeiten besitzen. Beispielsweise bekommt man den billigsten Palm OS PDA bereits um knapp über 100 Euro, das billigste, auf Windows CE basierende Gerät wird erst ab 200 Euro angeboten[43].

#### PalmOS basierte Geräte

PalmOS wurde von Palm Computing entwickelt. Zurzeit wird das Betriebssystem von PalmSource [65] weiterentwickelt.
Die ersten auf PalmOS basierenden PDAs (kurz Palms) kamen um 1996 auf den Markt und waren bald darauf marktbeherrschend auf diesem Gebiet. Die Palms zeichnen sich durch ein einfach zu erlernendes und ressourcenschonendes Betriebssystem sowie durch eine über einen langen Zeitraum nahezu unveränderte Hardware aus.
Die aktuellste Version von PalmOS ist zurzeit *Cobalt* (Vers. 6.1), die unter anderem Multitasking und -threading, bessere Multimedia Unterstützung, integrierte Kryptographie-Module bietet, gleichzeitig aber noch immer mit den Vorgängerversionen binärkompatibel ist. Zur Zeit gibt es aber noch relativ wenige Geräte am Markt, die auf PalmOS Cobalt basieren. Die meisten benützen noch immer die Vorgängerversion Garnet (5.4)

Für die Entwicklung gibt es von Palm Source Unterstützung über ein *Developer Program*, womit man Zugriff zu diversen Softwareentwicklung Kits (sogenannte SDK) und Entwicklungsumgebungen bekommt. Es gibt auch andere Anbieter von kostenpflichtigen Entwicklungsumgebungen (z.B. Metroworks Codewarrior). Direkt unterstützt von Palm Source wird nur die Sprache C. Es gibt aber auch C++, Java, Smalltalk und andere Compiler/Interpreter für die Plattform. Da es PalmOS schon relativ lange gibt und Software, die für ältere Versionen geschrieben wurde auch noch auf den neuen Geräten läuft, gibt es eine grosse Auswahl an vorhandenen Anwendungen und Tools.
Wie bereits erwähnt wurde der komplette Handheld-Markt von den von Palm entwickelten Geräten oder von Geräten anderer Firmen, die das Palm-Betriebssystem verwendeten, beherrscht. Erst durch den Markteinstieg von Microsoft mit ihren auf PocketPC-basierenden Produkten änderte sich das. Noch im Jahr 2000 hatte die Gruppe der Palm Geräte einen Marktanteil von über 70% [75, S. 29], der aber seitdem kontinuierlich sinkt. Laut neuesten Statistiken haben sie nur mehr einen Anteil von $\approx$ 18%[70]. Gründe dafür dürften zum einen der recht aggressive Markteinstieg von Microsoft sein, und zum anderen auch der Umstand, dass der gesamte PDA Markt zugunsten diverser Smartphones rückläufig ist.

#### WindowsCE (PocketPC) basierende Geräte

Als PocketPCs werden Geräte bezeichnet, die als Betriebssystem Microsoft WindowsCE einsetzen. Die neueste Generation trägt den Namen „Windows Mobile for PocketPC“.
WindowsCE lief bis zur PocketPC Version 2002 auf Geräten mit unterschiedlichen Prozessorarchitekturen, darunter Intel x86, MIPS, ARM, und Hitachi-SH-Prozessoren. Aufgrund des Aufwands für die Hard- und Softwareentwickler durch die dadurch entstandene Gerätevielfalt, wird das Windows Mobile Betriebssystem von Microsoft seit Version 2003 nur noch für ARM-basierende bzw. kompatible Geräte angeboten.
Die Hersteller der Geräte sind für die finale Anpassung des Betriebssystems an die Hardware zuständig wodurch man nach dem Kauf eines Geräts an den Hersteller bzw. den OEM bezüglich eines Updates gebunden ist. Neue Modelle werden in einem sehr kurzen Zyklus (oft unter einem Jahr) auf den Markt gebracht, die älteren Modelle werden in Bezug auf Betriebssystempflege in der Regel sehr bald nicht weiter unterstützt. Zum Beispiel bietet HP nur einen Upgrade Schritt an. Für die Geräte, die mit PocketPC 2002 geliefert wurden, gibt es ein Upgrade auf PocketPC 2003, aber keines mehr auf Windows Mobile for Pocket PC.
Software für den PocketPC ist auf normalen Windows PC's nicht lauffähig. Auch sind bei verschiedenen Versionen einer Applikation die Dateiformate nicht gleich (z.B. Word 2003 und PocketPC Word).
Es gibt aber eine breite Auswahl an Software für PocketPCs, da Microsoft nicht nur SDKs gratis zur Verfügung stellt, sondern auch Entwicklungsumgebungen (eMbedded Visual C++ 4.0 [18]) und Emulatoren für die verschiedenen Generationen des WindowsCE Betriebssystems.

### 2.3.2 Smartphones

Als Smartphones werden normalerweise Geräte bezeichnet, welche einerseits zwar als normales Mobiltelefon nutzbar sind, gleichzeitig aber auch Funktionalititäten von PDAs bie-

ten. Wie bereits bei den PDAs können auch hier die Geräte durch das verwendete Betriebssystem kategorisiert werden. Abgesehen von den proprietären Lösungen einiger Hersteller, die nur auf der vorgesehenen Hardware laufen, gibt es einige Betriebssysteme von größerer Bedeutung, die auf verschiedenen Geräten eingesetzt werden. Das sind folgende Systeme:

- Symbian OS[51]
- Betriebssystem von Research in Motion (RIM) für die Blackberry Geräte[66]
- Embedded Linux[52]
- PalmOS
- Windows CE
  - Windows Mobile 5.0 for Smartphones bzw. vormals Pocket PC Phone Edition
  - Windows Powered Smartphone (Geräte ohne Touchscreen)

Grundsätzlich haben Smartphones die gleichen Eigenschaften wie PDAs. Aus diesem Grund wird in dieser Arbeit nicht mehr speziell auf diese Kategorie von Geräten eingegangen.

### 2.3.3 Web Pad

Unter Web Pad (auch Web Tablet oder Smart Tablet genannt) versteht man ein drahtlos angebundenes, portables, günstiges, einfach bedienbares, tafelförmiges Gerät, welches einfachen und intuitiven Internetzugang erlaubt. Da diese Bezeichnung nicht wirklich definiert ist, versteht jede Firma unter WebPad etwas anderes. Beispielsweise ist für Xilinx[42] ein Web Pad ein zweiteiliges Gerät, bestehend aus einem Mobilteil mit Display und einer Basisstation mit einer fixen Verbindung ins Internet.
Als Beispiel für einen Web Pad dient die Atigo Reihe (siehe Abbildung 2.5) der Firma Xybernaut®. Die Tabelle 2.7 vergleicht die technischen Daten der verschiedenen angebotenen Modelle.

Abbildung 2.5: Atigo T von Xybernaut[27]

Alle Geräte der Atigo Reihe zeichnen sich durch eine höhere Robustheit aus. Funktionsgarantie wird für Temperaturen von 0°-40°C und einer Luftfeuchtigkeit von 0% - 90% gewährt. Auch sind die Geräte unempfindlich gegen Stöße, da sie in der Basisausstattung keinerlei bewegliche Teile enthalten (weder Festplatten noch Lüftungsventilatoren).

| Modell | Atigo T | Atigo S | Atigo M |
|---|---|---|---|
| **CPU** | Transmeta® Crusoe TM5800 1Ghz | AMD GEODE GX2 400-533 Mhz | Intel®XScale PXA255 -400 Mhz |
| **Speicher RAM** | 256 MB SDRAM | 128-256 MB SDRAM | 128 MB SDRAM |
| **Speicher FLASH/ROM** | 128 MB - 4 GB | 64MB - 4GB | 32 MB ROM + 64 Flash Opt. |
| **Display** | 8,4“ Touch Screen TFT Display, 800x600 Resolution, 24 Bit Color | | |
| **Betriebs-system** | Win XP Emb. & Prof., Div. Linux | Win CE.net 5.0, Linux, Win XP Prof. & Emb. | Windows CE.NET 4.2 |
| **Wireless Schnittstellen** | | über miniPCI Karte integrierbar | |
| **sonstige Schnittstellen** | 1 x USB 2.0 1 x USB 1.1 | 1 x USB 2.0 1 x USB 1.1 | 1 x USB 1.1 |
| **Steck-plätze** | PC Card (Type II) CF Type II | PC Card o. mini PCI CF oder interne HDD | PC Card Type II CF Type II |
| **Akku** | Interne Lithium-Ion Battery optional. ansteckbare externe Battery | | |

Tabelle 2.7: Technische Daten der Atigo Reihe von Xybernaut[54]

### 2.3.4 Smart Display

Ein Smart Display ist mit einem Web Pad bzw. einem Tablet-PC verwandt. Es ist vom Prinzip her eine Erweiterung eines Standard PCs. Der Begriff Smart Display wird eigentlich nur von der Firma Microsoft verwendet und bezeichnet einen *kabellosen Monitor*. Diese Geräte dienen zum Fernsteuern eines normalen Windows PCs (im speziellen Windows XP, es können aber auch ältere Versionen verwendet werden) über den Dienst der Remote-Steuerung.
Von der Hardware sind diese Geräte vergleichbar mit einem PDA bzw. einem Webpad, haben aber ein grösseres Display (im Normallfall 800 x 600 und mehr) und verfügen alle über eine WLAN Unterstützung. Das Betriebssystem auf diesen Geräten ist eine spezielle Version von Windows CE (Windows CE for Smart Displays)[20].
Als Beispiel für diese Art von Geräten dient das airsync V210 Wireless Display der Firma Viewsonic (siehe Abbildung 2.6)[23]. Die technischen Daten können der Tabelle 2.8 entnommen werden.

| | |
|---|---|
| **Prozessor** | 400MHz Intel XScale$^{TM}$ |
| **Speicher** | 64MB Rom und 128 MB Ram |
| **Betriebsystem** | Windows CE .NET 4.2 |
| **Wireless Schnittstellen** | WLAN 802.11b |
| **Batterie** | bis zu 4h Laufzeit |
| **Display** | 10” TouchScreen, Auflösung 800x600 mit True Color[9] |

Tabelle 2.8: Technische Daten des airsync V210

[9]True Color (engl. für Echtfarben) bezeichnet eine Farbtiefe von 24 Bit (16,78 Millionen Farben).

Abbildung 2.6: airsync$^{TM}$ V210 Wireless Display[23]

Diese Geräte sind nur mehr als Restposten lieferbar, da die Produktion eingestellt wurde. Deswegen wird auf sie nicht mehr näher eingegangen.

### 2.3.5 Tablet-PC

Ein Tablet-PC stellt einen mobilen PC dar, bei dem die Eingabe auch direkt über das Display (Touch Screen bzw. über einen speziellen Eingabestift) erfolgen kann. Erste Versionen solcher Geräte gab es schon Anfang der 90er Jahre z.B. den T100X-Dynapad von Toshiba[68].
Microsoft hat inzwischen dafür einen Standard definiert [21]. Zusammengefasst sind es mit einem Stift bedienbare Notebooks. Diese sind 100 % kompatibel zu normalen PCs. Daher kann Standard-Software auf den Geräten verwendet werden bzw. auch die umfangreichen PC-Entwicklungsumgebungen und SDK's.
Es wird zwischen zwei Typen von Tablet-PCs unterschieden:

**Convertibles:** Das sind klassische Notebooks, bei denen zusätzlich das Display so gedreht werden kann, dass die Tastatur verdeckt und die Stifteingabe möglich ist. Sonst unterscheiden sie sich nicht von einem normalen Notebook. Beispiel für ein Convertible ist der Acer TravelMate C110 (siehe Abbildung 2.7).

**Slate:** Slates sind Geräte, die auf geringes Gewicht und geringe Größe getrimmt sind. CPU, Speicher, Harddisk usw. sind direkt ins Display eingebaut. Diese lassen sich nur durch den Stift und einige spezielle Funktionstasten bedienen. Tastaturen (wenn verfügbar) können angesteckt werden. Dadurch sind sie vom Aufbau und Aussehen sehr ähnlich den oben vorgestellten Web Pads bzw. Smart Pads. Tablet-PC's haben aber normalerweise eine bessere Hardwareaustattung (CPU, Grafikkarte, Festplatte usw.). Ein Beispiel für ein Slate Modell ist der Fujitsu ST5021 (siehe Abbildung 2.7).

Seit kurzem gibt es auch noch Ultra Mobile PCs (UMPC), die von Microsoft und Intel als neue Geräteklasse von tragbaren Computern im Frühjahr 2006 vorgestellt worden sind. Die Geräte verfügen normalerweise über ein ca. 20 cm (7 Zoll) großes TFT-Display und werden über einen Touchscreen bedient. Laut Microsoft müssen sie mit WiFi und Bluetooth ausgestattet sein. Erste Geräte wie der Samsung-Q1 (siehe Abbildung 2.8) sind

(a) Convertible Tablet-PC, ein Acer TravelMate C110 [22]

(b) Slate Tablet-PC von Fujitsu [16]

Abbildung 2.7: Beispiele für verschiedene Tablet-PCs

bereits erhältlich und kosten rund 1300 €.
Auf diese Geräte wird nicht weiter eingegangen, da sie im Prinzip nur verkleinerte Tablet-PC's sind (gleiches Betriebssystem, normale PC-CPU usw.).


Abbildung 2.8: UMPC Samsung Q1[32]

### 2.3.6 Übersicht mobiler Devices

Die Tabelle 2.9 soll als Kurzübersicht der vorher aufgezählten Geräte dienen.

| | **PDA** | **Web Tablet Web Pad** | **Smart Display** | **Tablet-PC** | **Smart-phones** |
|---|---|---|---|---|---|
| **Schnitt-stellen** | verschiedenste (Preisfrage) | WLAN (USB bzw. CF vorhanden) | WLAN (USB bzw. CF vorhanden) | WLAN, Blue-tooth (USB bzw. PC-Card vorhanden) | verschiedenste (Preisfrage) |
| **Display** | QVGA[10] bzw. VGA | 800x600 bzw. höher | 800x600 bzw. höher | 800x600 bzw. höher | VGA, QVGA[10] bzw. weniger |
| **Preis** | ab 300 € | ab 500 € | ab 1300 € Nicht mehr am Markt | ab 1000 € | ab 300 € |
| **Eigen-schaften** | klein (Hemd-tasche), PIM Funktionalität, Standard OS (PALM bzw. Windows CE) | Drahtlose Internet-anbindung, Bedienung mit Stift, Tafelförmig | „Wireless Monitor“ Idee von Microsoft | „PC mit Touchscreen“ Convertibles & Slates | Handy mit PDA Eigen-schaften bzw. umgekehrt |
| **Vorteile** | billig, „Gute Größe“, große Auswahl, standardtisiert, viele Ent-wicklertools | verschiedens Modelle, Display | standardisierte Funktionalität | „Normaler PC“ dadurch Entwicklung mit „Standard-software“ | Größe, viele Anbieter |
| **Nachteile** | *rugged* Geräte teuer | Preis, weniger Anbieter | Produktion eingestellt | Preis Größe | Je nach Modell Display sehr klein bzw. ohne Touchscreen |

Tabelle 2.9: Vergleich der Eigenschaften mobiler Geräte

[10] QVGA im Hochformat bedeutet Auflösung von 240x320

## 2.4 Anforderungen für Smart Devices im automotiven Bereich

Wichtig für den Betrieb von sogenannten elektronischen Unterbaugruppen (EUB, das sind Geräte, die an das Bordnetz des KFZ angeschlossen sind) ist eine Überprüfung der elektromagnetischen Verträglichkeit (EMV)[7, 64-79]. Wie genau diese zu erfolgen hat, wird in der EU Richtlinie 72/245/EWG bzw. deren Anpassung 2004/104/EG genau dargelegt. Für andere Länder ausserhalb der EU gibt es eigene Richtlinien, die aber im Großen und Ganzen ähnliche Anforderungen haben und deswegen hier nicht näher aufgezählt werden. Auch ist in dieser Richtlinie das Diagramm zur EUB Klassifikation enthalten, das darstellt, ob die Richtlinie überhaupt zur Anwendung kommt (siehe Abbildung 2.9). Weiters wird noch festgelegt, dass diese Direktive nur für sicherheitsrelevante Bauteile, das sind z.B. Bauteile, die direkt für den Schutz der Insassen zuständig sind (Airbag, Rückhaltesystem) oder auch die Einfluss bzw. Kontrolle auf das Fahrzeug haben (Bremsen, Motor usw.), verwendet werden soll.

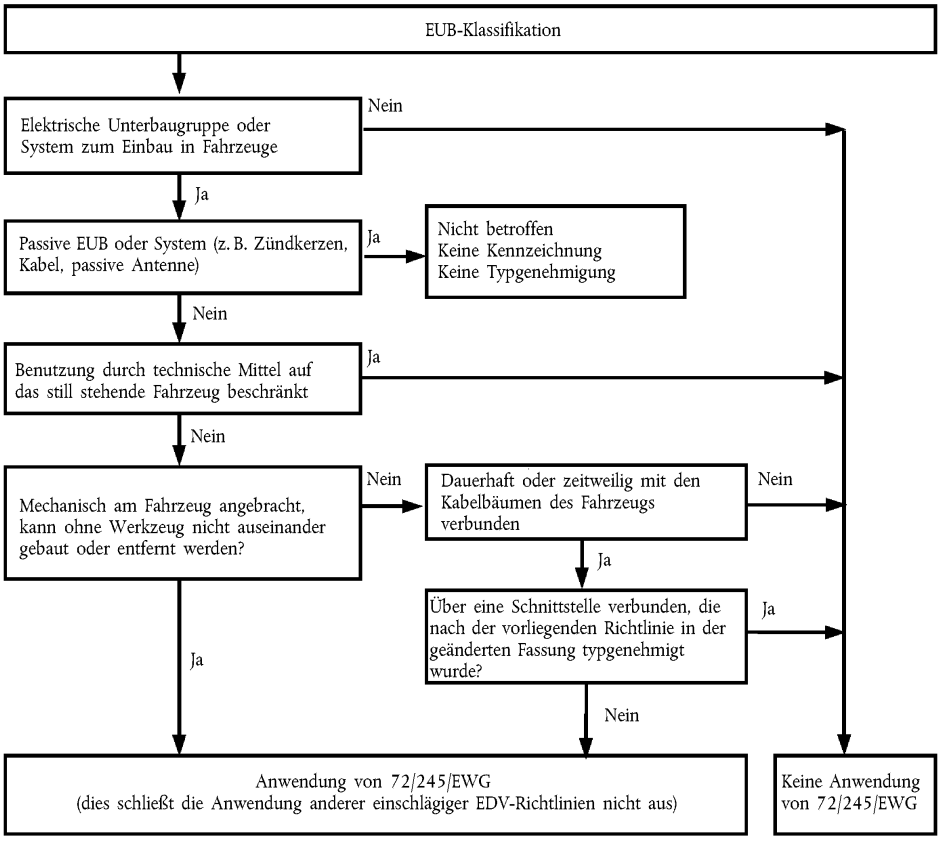



Abbildung 2.9: Klassifkation der EBUs bezüglich Anwendbarkeit der Richtlinie 2004/104/EG [4, S. 7]

Daraus kann geschlossen werden, dass für Smart Devices als Mess- und Monitoring Einheiten in KFZ keine speziellen Anforderungen bezüglich EMV gelten (die normalen

EMV Regeln, die für alle elektrischen Geräte gelten, wie beispielsweise die Richtlinie 89/336/EWG, müssen dennoch erfüllt werden).
Begründet kann das damit werden, dass einerseits die Geräte (falls eine eigene Energieversorgung über Batterie oder Akku verwendet wird) nicht an das Bordnetz des Fahrzeugs angeschlossen sind, teilweise auch nicht fix mit dem Fahrzeug verbunden sind (z.B. Mikrofon für die Lautstärkenmessung) und keinerlei Sicherheitsrelevanz besitzen (rein passive Messgeräte).

Dennoch müssen einige Überlegungen bezüglich des Gehäuses der Geräte angestellt werden. Die „Smart Transducer“ sollten eine gewisse Robustheit und vor allem Schmutz und Feuchtigkeitsunempfindlichkeit besitzen, um sie z.B. an der Fahrzeugachse oder im Radkasten einsetzen zu können. Für Geräte, die nur im Fahrzeuginnenraum Verwendung finden, (wie z.B. für Monitoring) ist auch das nicht wirklich nötig.

## 2.5 Verfügbare Produkte für das Monitoring

Hier wird ein kurzer Überblick über aktuelle, verfügbare Produkte im Bereich mobiles Monitoring gegeben. Ebenso wird kurz erklärt, warum sie für die Lösung nicht in Frage kommen.

### 2.5.1 ETAS ES780

Ein gutes Beispiel für ein Monitoring Device ist das ES780 Heads-Up Display mit Fernbedienung 2.10 von ETAS Inc. [11].

> Die ETAS Group, ... , liefert ein komplettes Spektrum einheitlicher Entwicklungs- und Diagnosewerkzeuge, die den gesamten Lebenszyklus eines Steuergeräts umfassen.[34]


Abbildung 2.10: Erweiterung ES780 für INCA von ETAS

Auch dieses Gerät wurde entwickelt, damit ein Versuchsingenieur der mit dem Softwarepaket INCA arbeitet, möglichst einfach die Applikation fernsteuern kann. Die Fernbedienung wird am Lenkrad befestigt, wobei die Befehle per Infrarotübertragung an das Display gesendet werden. Das Display selbst wird über TCP/IP an den PC, auf dem INCA läuft, bzw. an das Messsystem gekoppelt. Die Befehle werden dann über einen Treiber an die COM[12] API von INCA weitergeleitet.

[11] http://de.etasgroup.com/

[12] Das Component Object Model (Abkürzung COM) ist eine von Microsoft entwickelte proprietäre Technologie, um unter Windows Klassen aus DLLs (Dynamic Link Libraries) zu exportieren, die von allen 32-bittigen Microsoft-Betriebssystemen unterstützt wird.

**Nachteile**

- Die Anbindung des Displays erfolgt über Standard Ethernet. Beim jetztigen Aufbau von AVL-DRIVE werden die normalen Netzwerk Steckplätze aber bereits verwendet (RJ45/Ethernet Stecker wird für die DMU[13] verwendet). Dadurch müsste man einen Netz Hub oder Switch zusätzlich mitinstallieren.
- Eine beschränkte Darstellung (nur monochrom und nur bedingte graphische Möglichkeiten)
- Es gibt wieder Abhängigkeit von einer speziellen Hardware.
- Es könnte unter Umständen Probleme geben, wenn die Hardware in einem LKW verwendet wird, da die Reichweite von Infrarot zu kurz ist.

**Vorteile**

- Das Gerät ist sehr robust (z.B. Temperaturbereich von -40°C bis +85°C).
- Es ist einfach zu bedienen, da die Fernsteuerung direkt am Lenkrad befestigt wird.

### 2.5.2 Exor Electronic R&D

Exor Electronic ist ein Hersteller von sogennanten *Human Machine Interface* Panels (HMI, übersetzt Mensch-Maschine-Schnittstelle) für die Industrie-Automatisierung. Die Firma bietet eine große Auswahl von verschiedenen SPS Geräten, beginnend von einfachen Text-Matrix Panels (z.B. der ePAD03 ab ca. 390 USD) über monochrome Touchpanel (z.B. das eTOP05 um ca. 1.000 USD) bis zu vollständigen IndustriePC's. Die Geräte werden über eine spezielle Software, die „UniOP Designer-Software", programmiert, welche verschiedenste Grafikfunktionen schon inkludieren. Auch bieten sie Schnittstellen zu den verschiedensten Netzwerktypen an (Ethernet, ProfibusDB, Interbus und viele mehr).


(a) eTOP05 (b) eTOP1XX Windows CE basierend

Abbildung 2.11: Exor eTOP Panels [35]

[13] DMU= DRIVE Main Unit siehe Abbildung 3.1

**Nachteile**

- Die Panels sind nicht für den mobilen Einsatz gedacht. D.h. es wird ein extra Gehäuse benötigt.
- Die verschiedenen Geräte sind sehr teuer (eTOP05 kostet bereits ca. 1000 USD, zusätzlich kommen noch Kosten für die Designersoftware und mögliche Schnittstellen hinzu).
- Die Entwicklungsumgebung und auch die Panels sind ausgerichtet für Industrie-Automatisierung, z.B. erfolgt die Programmierung eher wie eine SPS und nicht in einer klassischen Sprache. Auch werden viele Features unterstützt, die überhaupt nicht benötigt werden (Echtzeitverhalten usw.)

**Vorteile**

- Die verschiedenen Geräte sind sehr robust (erfüllen unter anderem die Schutzklasse IP65).
- Sie sind erweiterbar mit verschiedenen Wireless-Schnittstellen (z.B. WLAN 802.11b).

# Kapitel 3

# DRIVE-Remote Design

## 3.1 Analyse des Ist-Zustands

AVL-DRIVE ist ein Mess- und Analyse-System zur objektiven Beurteilung der Fahrbarkeit und des Fahrzeugcharakters eines PKWs oder LKWs auf der Straße, auf der Rolle oder am Prüfstand. Der Begriff Driveability beschreibt das komplexe, subjektive Wahrnehmungsvermögen eines Fahrers. Die Aufgabe des AVL-DRIVE Systems ist es, ausgehend von den physikalischen Messungen die einzelnen unterschiedlichen Fahrzustände zu identifizieren, charakteristische Messgrößen online zu berechnen und mit objektiven DRIVE Noten zu bewerten.

Mit dem bestehenden Messsystem werden rund 15 verschiedene Signale wie Geschwindigkeit, Drehzahl, Pedalstellungen, Längsbeschleunigung, Querbeschleunigung, Laststrom und Vibrationen im Fahrzeug in Echtzeit ausgewertet. Diese Messkanäle werden sowohl vom ECU CAN Bus oder fahrzeugeigenen Sensoren, als auch von zusätzlich installierten Sensoren für Beschleunigungen, Vibrationen, Laststrom, Pedalstellungen abgegriffen. Das System ist in hohem Grad an spezielle Anforderungen adaptierbar, indem weitere Kanäle gemessen, Bewertungsgewichte angepasst und eigene Bewertungen durchgeführt werden. Dies führt zu individuellen Ergebnissen, die zwar den jeweiligen Bedarfsfall abdecken, aber die Vergleichbarkeit mit anders konfigurierten Messungen einschränkt.

### 3.1.1 Driveability

Das Wort Driveability beschreibt das durchaus komplexe subjektive Wahrnehmungsvermögen eines Menschen in Bezug auf verschiedene Betriebszustände eines Fahrzeugs. Ziel ist es, eine Bewertung zu erstellen, die:

- der eines realen Menschen sehr nahe kommt und somit die Subjektivität eines Menschen ausschließt,
- weit aus mehr Bewertungsaspekte betrachtet als ein einzelner Mensch im Stande wäre,
- die Bewertung zudem mit objektiven Daten/Ursachen verknüpft.

Zurzeit werden Fahrzeuge noch häufig von Gruppen von Testfahrern ausschließlich subjektiv bewertet. Dies ist einerseits sehr kostenintensiv, und andererseits kann die Bewertung einzelner Faktoren bei den Testfahrten je nach Tagesverfassung der Tester unterschiedlich interpretiert werden. Eine Auswertung ist langwierig und fehlerbehaftet, und es müssen meist mehr als ein Versuchsobjekt zur Verfügung stehen.
Dass dieses Thema im Laufe der Zeit immer mehr an Bedeutung gewonnen hat, liegt darin begründet, dass nicht nur das optische Erscheinungsbild eines Fahrzeugs wichtig ist, sondern auch immer mehr das oft zitierte „sich wohl fühlen“ oder auch die „Freude am Fahren“ in einem Fahrzeug von Kunden kritisch beurteilt wird. Das Fahrzeug aus Kundensicht zu betrachten wird zum wesentlichen Designfaktor.
Da es verschiedene Fahrzeugtypen wie z.B. Sportwagen, Limousine oder Kleinwagen gibt, und jede dieser Klassen andere Käuferschichten mit unterschiedlichen Ansprüchen ansprechen soll, ist es unumgänglich, diese in der Bewertung zu trennen. Aus diesem Grund hat man verschiedene Klassen definiert, die gleiche Faktoren unterschiedlich bewerten. Beispielsweise die Reaktion des Motorhochlaufs zur Fahrpedalstellung (bei einem Sportwagen exponentiell agressiv, in der Luxusklasse linear). Die Begriffe *Dynamik* und *Komfort* führen in diesem Beispiel zu gegenläufigen Bewertungen. Eine ruckartige Beschleunigung bei einem „Tip in“ führt in der sportlichen Klasse zu einer guten in der Luxus Klasse zu einer schlechteren Bewertung. Die Zielgruppe für das Fahrzeug beeinflusst auch die Bewertung. Dafür war es auch nötig, durch jahrelange Tests, Beobachtungen und Versuche diese Bewertungstabellen aufzubauen. Sie bilden den Kern eines AVL-DRIVE Systems und werden kontinuierlich weiterentwickelt.

### 3.1.2 Aufbau von AVL-DRIVE

Das Messsystem AVL-DRIVE besteht nicht nur aus der Analyse-Software, sondern auch aus einem Hardwareteil, der die eigentliche Messeinrichtung und eine Vielzahl von Sensoren und deren Verkabelung enthält.

#### Hardware[31]

Die Hardware wird aus der eigentlichen Messeinrichtung (DRIVE Main Unit, DMU, Abbildung 3.1) und einer Vielzahl von Sensoren gebildet. Als Sensoren kommen Beschleunigungssensoren, Vibrationssensoren, Stromwandler, Mikrofone, Wegpotentiometer, Gierratensensoren, Fahrbahnabstandslaser (Bestimmung des Wankwinkels), und Induktivgeber zum Einsatz. Des weiteren werden, soweit möglich, Messdaten aus dem fahrzeugeigenen CAN-Bus mit aufgezeichnet. Falls Sensordaten auf dem CAN-Bus verfügbar sind, die normalerweise durch externe Sensoren gemessen werden, ist es möglich, dass externe Sensoren durch diese ersetzt werden. Hierbei muss aber genauestens kontrolliert werden, ob diese Daten mit ausreichend hoher Genauigkeit vorliegen. Derzeit wird die eigentliche Messeinheit von der Firma IMC bezogen und mit leichten Modifikationen der Elektronik in einem Gehäuse der AVL verkauft. Hierbei handelt es sich um ein handelsübliches Gerät zur Erfassung von Messwerten in Echtzeit. Als Eingangsgrößen sind analoge Spannungen, CAN-Bus Signale und Pulssignale möglich. Für die analogen Spannungen können über eine von IMC mitgelieferte Software die Messbereiche $\pm$5V oder $\pm$10V definiert werden. Bei den Puls Eingängen gilt, dass sowohl Signale der fahrzeugeigenen Induktivgeber als

auch Signale einer TTL Logik gemessen werden können. Die Auflösung, jeweils bezogen auf den Messbereich, beträgt 12 bit. Über die mitgelieferte Software ist es möglich, die Abtastrate getrennt zu definieren. Das Maximum liegt bei 1000Hz pro Kanal bzw. bei 40kHz Summenabtastung. Intern werden die gemessenen analogen Werte mit denen des fahrzeugeigenen CAN-Busses zeitlich synchronisiert und über eine RJ45 Schnittstelle an den Messcomputer ausgegeben. Eine Übersicht über den Hardwareaufbau ist in Abbildung 3.3 zu finden.

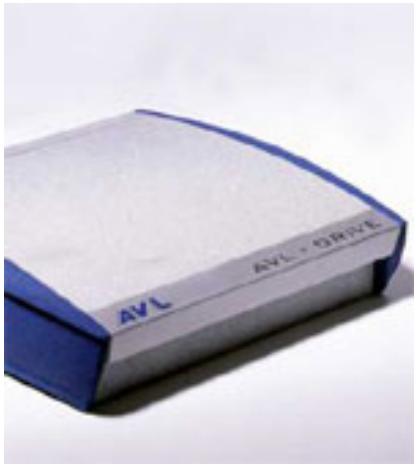



Abbildung 3.1: DRIVE MAIN UNIT (DMU)


Abbildung 3.2: AVL-DRIVE Display (Ist Zustand)

**Software**

Den Softwareteil von AVL-DRIVE kann man grob aufteilen in:

- Messdatenschnittstelle
- Betriebszustanderkennungslogik
- Driveability-Bewertungseinheit
- Ausgabe-Einheit (Userinterface)

Die Messdatenschnittstelle lässt den Zugriff auf verschiedene Hardwareschnittstellen und Offline-Messungen (von der Festplatte) zu. Letzteres dient der Simulation der Messungen. Das Analysesystem benutzt die erfassten Daten der Messeinheit zur Zustandserkennung. Dieser Teil (Triggermaschine) erkennt zur Zeit mehr als 75 Unterbetriebszustände (siehe Abbildung 3.5). Die erkannten Zustände werden dann anhand von verschiedensten Kriterien und den dazugehörigen Messwerten bewertet. Die sogennanten Detailkriteria werden dann wieder zu Subkriterien zusammengefasst und diese dann zu 15 Hauptkriterien (siehe Abbildung 3.6).
Die Erkennung der Betriebszustände erfolgt mittels Fuzzy Logik. Zum Beispiel ist der

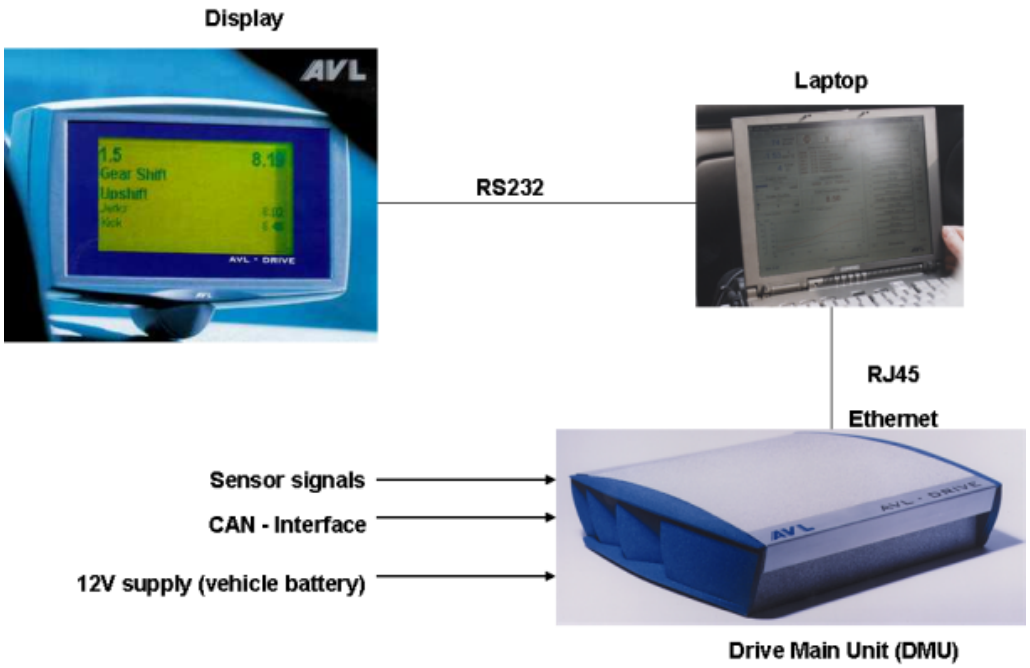



Abbildung 3.3: AVL-DRIVE Ist-Zustand

Leerlauf dadurch charakterisiert, dass kein Gang eingelegt ist, die Fahrpedalstellung unter einem bestimmten Pegel liegt usw. Bei der Nachbildung des subjektiven Empfindens mittels Computer müssen die Betriebszustände jedoch noch genauer beschrieben werden, und es muss aufgrund des subjektiven menschlichen Verhaltens auch auf die vorausgegangenen und die nachfolgenden Zustände Rücksicht genommen werden. Abhängig von der Intensität der menschlichen Aktionen wird die Toleranzschwelle gegenüber den auftretenden Fahrzeugreaktionen verschoben. Der Mensch ist gegenüber Unregelmäßigkeiten dann toleranter, wenn zuvor eine intensive Fahreraktion gesetzt wurde. Sehr kritisch werden daher die Betriebszustände Leerlauf und Konstantfahrt bewertet, in denen der Fahrer wenig oder keine Aktionen setzt. Die Nachbildung des äußerst komplexen subjektiven Empfindens erfolgt mit analytischen Verfahren wie Kennlinien und Kennfeldern sowie mit Hilfe von künstlichen neuronalen Netzen (KNN) (Artificial Neural Network, ANN). Für jedes Kriterium wird ein Driveabilityindex berechnet (ein Wert von 1 bis 10), wobei das Ergebnis eines Kriteriums durchaus als Eingangsgröße für andere Kriterien dienen kann. Die vielen Einzelbewertungen werden gruppiert und gemittelt. Diese Mittelung ist der menschlichen Bewertung nachempfunden und erfolgt teilweise auch mit neuronalen Netzen.
Zur Bewertung der Fahrzustände ist ein Bewertungsschlüssel festgelegt, der den Driveabilityindex von 1 bis 10 verbal erklärt und eine Akzeptanzschwelle festlegt. Eine übliche Schwelle ist ein Wert von 7, unter dem die Gesamtbewertung eines Fahrzeuges als nicht mehr völlig akzeptabel beurteilt wird (siehe Tabelle 3.1).
Das Userinterface (siehe Abbildung 3.4) ist, wie bereits erwähnt, für die Analyse der Messung und ihre Bewertung ausgelegt und deswegen für die Durchführung der Messung nur bedingt geeignet.

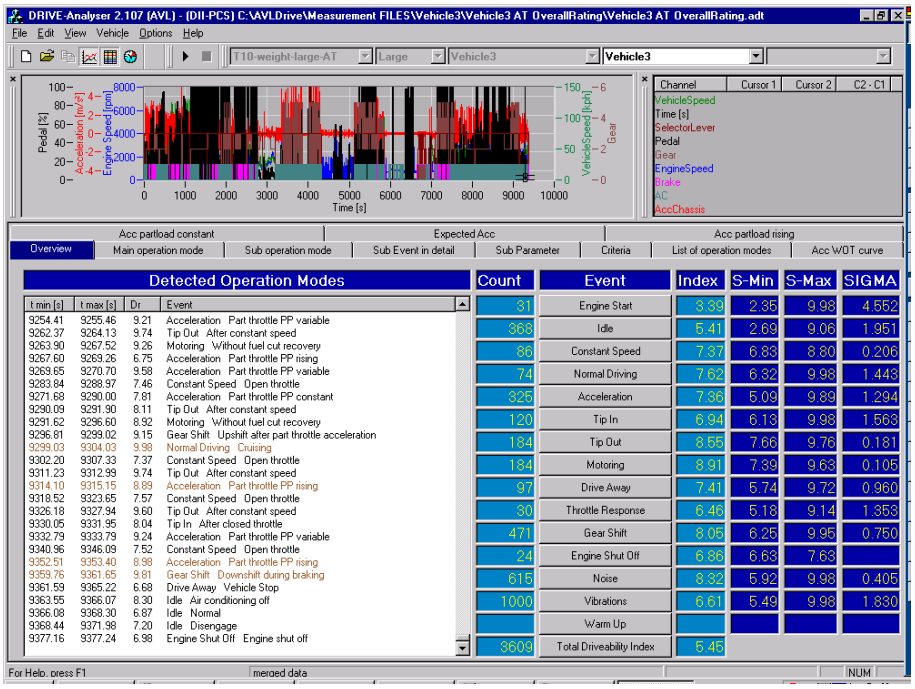



Abbildung 3.4: Screenshot von AVL-DRIVE, Ansicht einer Messung

### AVL-DRIVE Display Istzustand

Um die Verwendbarkeit von AVL-DRIVE während einer Messung zu erhöhen, gibt es ein externes Display, das an das Notebook angeschlossen wird. Das zur Zeit verwendete externe Display (siehe Abbildung 3.2) ist ein einfaches Dot-Matrix LCD, mit einer Auflösung von 160×80 Pixel. Das Display ist direkt über die serielle Schnittstelle mit dem PC verbunden und wird auch über diese mit Strom versorgt. Für die Anzeige läuft in AVL-DRIVE ein eigener Thread, der aus den aktuellen Daten eine Bitmap Grafik erzeugt und diese über die serielle Schnittstelle an das Display schickt, wo sie angezeigt wird.
Eine verbesserte Version des Displays verwendet einen sogenannten „CAR TV Monitor“ (siehe Abbildung 3.7), der über S-Video bzw. Composite Video am Notbook angeschlossen wird. Auf diesem *Zusatzmonitor* wird dann das bereits implementierte Testdisplay angezeigt. Diese Methode hat aber wie bereits das alte Display den Nachteil, dass es rein passiv ist. Zusätzlich müssen noch extra Kabel verwendet werden (für die Stromversorgung und das Videokabel). Aus diesem Grund ist diese Lösung für zukünftige Entwicklungen nicht mehr vorgesehen.

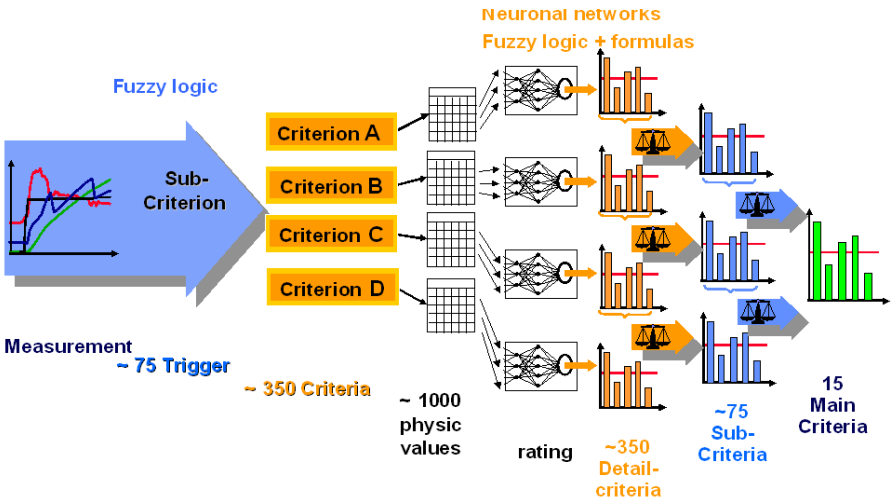



Abbildung 3.5: AVL-DRIVE Berechnung der DRIVE Note

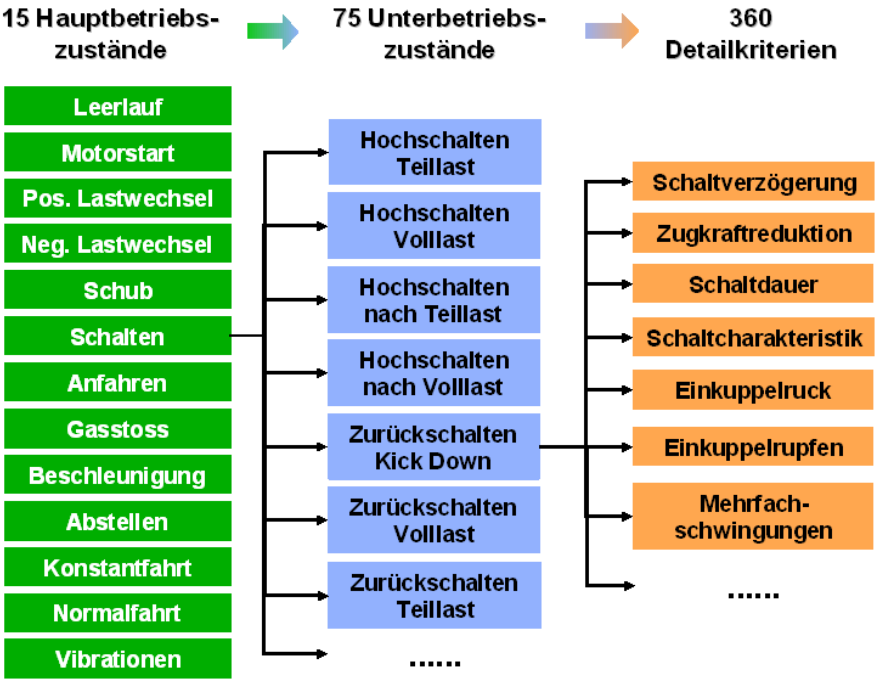



Abbildung 3.6: AVL-DRIVE Betriebszustände

| DR | Beschreibung | | Zone |
|---|---|---|---|
| 10 | Ausgezeichnet | Kunde ist begeistert | **Hervorragend** |
| 9 | Sehr gut | Fahrzeug erfüllt die Kundenerwartungen | |
| 8 | Gut | Fahrzeug verhält sich gut | **Zufriedenstellend** |
| 7 | Befriedigend | Fahrzeug weist leichte Schwächen auf | |
| 6 | Unangenehm | Vom Kunden als unangenehm empfunden | **Kritisch** |
| 5 | Störend | Vom Kunden als störend empfunden | |
| 4 | Ungenügend | Vom Kunden als sehr störend empfunden | **Mangelhaft** |
| 3 | Schlecht | Vom Kunden als Fehler reklamiert | |
| 2 | Sehr Schlecht | Fahrzeug funktioniert nur noch bedingt | **Inakzeptabel** |
| 1 | Keine funktion | Fahrzeug stellt ein Sicherheitsrisiko dar | |

Tabelle 3.1: Drivability Noten und ihre Bedeutung


Abbildung 3.7: Car TV Monitor

### 3.1.3 Begriffserklärung

Bei den folgenden Kapiteln werden verschiedenste Fachbegriffe und Bezeichnungen verwendet. Um den Text besser zu verstehen, werden hier die wichtigsten Begriffe erklärt:

**Kriterium (criteria)** Das ist die Ebene, die eigentlich die DRIVE Noten berechnet. Aus der Beschleunigung in X Richtung wird z.B. das Ruckeln erkannt und bewertet.

**Sub Events (Sub operation mode)** Damit wird ein erkannter Betriebszustand benannt, der softwaremäßig ein Event darstellt und zu Bewertungsberechnungen führt. Bsp.: Beschleunigung von 0 auf 100km/h.

**Hauptbetriebszustand (Main operation mode)** Dies wird auch Main Event genannt. Er dient zur Gruppierung von Subevents.

**Parameter (Sub)** Die Ursache für einen Event sind bestimmte Parameter, die in der Gruppe bestimmte Werte annehmen. Die Geschwindigkeit zu t0 = 0, t1 = 100 und der ununterbrochen positive Gradient der Geschwindigkeit führen zu dem Event „Beschleunigung von 0 auf 100km/h“

**Kanal (Channel)** Die Parameter werden aus Messkanälen berechnet. Für die Geschwindigkeitsparameter ist der Messkanal Geschwindigkeit notwendig.

**Experiment (Messung)** Für die Messungen sind veschiedenste Einstellungen des Fahrzeugs aber auch des Messsystems erforderlich. Diese werden im Rahmen der Messung im Messergebnis mit abgespeichert, um für spätere Betrachtungen oder Dokumentation zur Verfügung zu stehen.

**PCS** PC Simulation der Messung. Durch gespeicherte Daten wird eine Messung ohne angeschlossener Hardware nachsimuliert, zum Zweck der Veränderung der AVL-DRIVE-Noten mittels Variation des Experiments.

**Drivability Note** Eine Note zwischen 1 und 10 (siehe Tabelle 3.1), die an den einzelnen Events, bzw. der gesamten Messung vergeben wird (siehe Abschnitt 3.1.1).

## 3.2 Lastenheft (Anforderungen der Benutzer)

Wie bereits in der Einleitung erwähnt, ist einer der Hauptpunkte der Diplomarbeit das Ersetzen des bestehenden Displays durch eine allgemeine Lösung, die auch die Fernsteuerung von AVL-DRIVE erlauben soll (siehe auch Abbildung 3.8). Hier werden die groben Anforderungen aus der Einleitung detailliert beschrieben und spezifiziert.


Abbildung 3.8: DRIVE Soll-Zustand

### 3.2.1 Geforderte Komponenten

Die neue Lösung (das neue Display) *muss* folgende Anforderungen erfüllen, damit sie sinnvoll eingesetzt werden kann und einen wirklichen Mehrwert gegenüber der jetzigen Lösung ergibt:

#### Ersatz für das bestehende Display

Das neue Display muss zumindest das gleiche leisten, wie das bereits bestehende. Darunter fallen:

- Anzeigen des aktuellen „Events“ mit dazugehörigem „SubEvents“
- Anzeigen der zum Event zugehörigen Drivenote

- Anzeige von aktueller Geschwindigkeit, Gang, Pedalstellung des Fahrpedals (Gaspedal)
- Auswahlmöglichkeit zum Anzeigen vom:
  - Aktuelles „MainEvent“ + „SubEvents“
  - Aktuelles „MainEvent“ + „SubEvents“ + der „Detailevents“ mit dazugehörigen Drivenoten, aufsteigend sortiert nach der Note
  - Aktuelles „MainEvent“ + „SubEvents“ + mindestens die beiden vergangenen “MainEvent“ mit ihren Drivenoten + den Startzeitpunkt der “MainEvent“
- Die verschiedenen Events sollen auch für den Benutzer unmittelbar nach dem Auftreten angezeigt werden. Als Richtwert für die maximale Verzögerung wird 1 sek genommen. Auch kann Nachrichtenverlust vereinzelt (weniger als 1 % ) in Kauf genommen werden, da die Daten nur zur Information des Benutzers dienen und sonst nicht weiterverarbeitet werden.

Diese Punkte sind die Minimalanforderungen, die unbedingt erfüllt werden müssen, um das neue Display erfolgreich einsetzen zu können. Falls sie nicht erfüllt werden können, hat es keinen Sinn das bestehende Display zu ersetzen. Eine Ansicht des jetzigen Display wird zum Vergleich in Abbildung 3.2 dargestellt.

#### Fernsteuerung von DRIVE

Es muss möglich sein, eine Messung über das Userinterface zu starten bzw. zu stoppen.

#### Voice Kommentar für Messungen

Während einer Messung muss es möglich sein, Kommentare abzugeben, die einem Event oder einem Zeitpunkt während der Messung zugeordnet sind. Um es für den Benutzer einfach zu halten, soll der Kommentar als Audiostream aufgezeichnet werden (z.B. Wave File). Grundsätzlich wäre auch ein reiner Textkommentar möglich, was aber für die Verwendung während einer Fahrt mit einem Fahrzeug (LKW oder PKW) nicht handhabbar ist. Der Kommentar sollte danach in AVL-DRIVE aufscheinen und auch direkt von AVL-DRIVE heraus aufrufbar sein.

#### Vergabe eines Namens für die Messung

Wenn die Messung beendet wird (egal ob vom neuen Userinterface oder direkt von AVL-DRIVE aus) muss es die Möglichkeit geben, der Messung einen Namen zu geben. Standardmässig wird ein Default Name plus einer aufsteigenden Nummer verwendet (Bsp.: „DR00102“), der direkt am Display durch einen eigenen ersetzt werden kann (AVL-DRIVE vergibt den Default Namen am Beginn einer Messung). Der Name entspricht einem Verzeichnis, worin alle während einer Messung anfallenden Daten gespeichert werden. Am Ende der Messung wird dann das Verzeichnis auf den gewünschten Namen geändert.

#### Drahtlose Verbindung zwischen den AVL-DRIVE PC und dem neuen User Interface Device

Die Anbindung des Displays an den PC (Notebook) soll drahtlos erfolgen. Auch sollte das Display ohne externe Stromversorgung arbeiten können, damit keine zusätzliche Verkabelung nötig ist. Falls das aus technischen Gründen nicht möglich ist (z.B. Störungen, Interferenzen, Akkulaufzeiten oder ähnliches), ist auch eine kabelgebundene Lösung möglich.

### 3.2.2 Optionale Komponenten

Folgende Anforderungen an das neue Userinterface sind in die Kategorie „Nice to have“ einzuordnen und für ein funktionierendes System nicht verpflichtend.

#### Einfaches Userinterface

Das Userinterface des Displays soll so gestaltet werden, dass die Funktionen Start bzw. Stop einer Messung und das Aufzeichnen von Kommentaren direkt gestartet werden können (z.B. nur mit einem Klick, ohne zuerst ein Untermenü auszuwählen).
Zusätzlich soll gewährleistet werden, dass die Bedienung der häufigsten Tätigkeiten ohne Benutzung des PDA-Eingabestifts möglich ist (z.B. über Finger oder über die zusätzlichen Tasten am PDA). Diese Tätigkeiten sind, in Reihenfolge ihrer Wichtigkeit:

1. Starten und Stoppen einer Messung
2. Kommentar abgeben
3. Ansicht ändern
4. Erneuter Verbindungsaufbau zum PC (Notebook)
5. Ändern der überwachten Parameter

#### Mehrsprachfähigkeit

Das neue Userinterface soll Änderungen im Code die Möglichkeit bieten, die Sprache des Userinterface anzupassen. Darunter fallen auch Sprachen wie Chinesisch, Koreanisch und Japanisch. Um dies zu ermöglichen, sollten die Unicode-Zeichensätze verwendet werden.

#### Angezeigte Parameter für den User auswählbar machen

Dem Benutzer des neuen Displays soll die Möglichkeit gegeben werden, die Anzeige der Parameter selbst zu konfigurieren. Beim bisherigen Display waren diese immer fixiert (Drehzahl, Gang und Fahrpedal Stellung).
Weiters würde es von Vorteil sein, wenn die Liste der möglichen Parameter von AVL-DRIVE bezogen werden könnte und dort auch verwaltet werden würde.

#### Kalibrierung der verschiedenen Sensoren

Die Kalibrierung diverser Sensoren (vor allem sind das Beschleunigungs- und Vibrationssensoren) des Messsystem über das neue Userinterface ist auch ein erwünschtes Feature. Da diese Funktion zumeist nur bei Fahrtbeginn benötigt wird, braucht es nicht direkt erreichbar sein. Es reicht, wenn die Kalibrierung aus einem Untermenü aufgerufen werden muss.

#### Steuerung über Sprachkommandos

Eine sinnvolle und wünschenswerte Erweiterung der Funktionalität von DRIVE-Remote wäre die Möglichkeit der Kontrolle von AVL-DRIVE mittels Sprachkommandos. Es würden bereits die wichtigsten Kommandos (siehe Abschnitt 3.2.1) reichen. Eine Erweiterung um zusätzliche Kommandos sollte aber möglich sein.

#### Modularisierung des Userinterfaces

DRIVE wird von verschiedensten Kunden eingesetzt, die recht spezifische Anforderungen an das Produkt haben. Deswegen soll es möglich sein, das Userinterface für die verschiedensten Kunden (soweit gewünscht) individuell anpassbar zu machen.

#### Modularisierung des Verbindungsteils (Softwareseitig)

Da sich die kontaktlose Verbindungstechnologie relativ schnell ändert, muss das DRIVE-Remote leicht an neue Technologien anpassbar sein. Das soll einerseits durch definiertes API erreicht werden, andererseits soll auch das high-level Verbindungsprotokoll so allgemein gehalten sein, dass es unabhängig vom Transportträger ist.

#### Aufzeichnung von Positionsdaten

Eine weitere sinnvolle Erweiterung des Systems wäre „Global Positioning System“ (GPS) Daten aufzuzeichnen, damit für spätere Vergleiche die gefahrene Route vorliegt. Dies kann vorallem dazu genutzt werden, gemessene Events mit Teilstrecken in Verbindung zu bringen, damit sich z.B. erklären lässt, warum ein Event einmal bessere Drivability Noten als zu einem anderem Zeitpunkt besitzt (z.B. kann die Teilstrecke eine viel bessere Fahrbahn als eine andere haben. Dadurch sind natürlich auch die Fahrgeräusche geringer.). Zu diesem Punkt gehört nicht nur die Aufzeichnung der GPS-Daten, sondern auch die Auswertung der Daten. Vorzugsweise soll AVL-DRIVE selbst um ein Kartenmodul erweitert werden, in dem die aufgezeichneten Strecken eingeblendet werden.

#### Herstellerunabhängigkeit

Es soll versucht werden, das neue Userinterface Device herstellerunabhängig zu halten. Vor allem für die Hardware sollte es mehrere Anbieter geben. Dadurch bedingt muss auch der Softwareteil zu einem gewissen Teil plattformunabhängig sein. Optimal ist eine „write once, run anywhere“ Lösung. Es würde auch schon reichen, wenn durch ein geringfügiges Anpassen von einigen Display spezifischen Parametern (beispielsweise Anzahl und Funktionalität der am Gerät vorhanden Bedienknöpfe, Keyboard vorhanden, vorhandene

Touchscreen Funktionalität, Auflösung und Grösse des Displays ) die Software an die neue Hardware angepasst werden kann.

#### Test Client für PCs

Um das Testen zu vereinfachen soll es einen Client geben, der auf normalen PCs läuft und DRIVE-Remote *emuliert*.

### 3.2.3 Abgrenzung der Arbeit

Die displayseitige Software ist zu entwickeln. Zusätzlich soll mit den Entwicklern von AVL-DRIVE ein Kommunikationsprotokoll ausgearbeitet werden. Änderungen und Erweiterungen, die die AVL-DRIVE Software direkt betreffen, sind nicht Teil dieser Diplomarbeit. Sind solche notwendig, um DRIVE-Remote zu ermöglichen, so werden diese direkt von Mitarbeiten der Fa. AVL selbst durchgeführt.

## 3.3 Bewertung der Display Plattform

### 3.3.1 Anforderungen an das neue Userinterface Device

**Drahtlose Schnittstellen** Das Endgerät sollte drahtlose Übertragungen wie WLAN, BT bzw. die Möglichkeit von Erweiterungen über sonstige Schnittstellen, wie z.B. Compact Flash I/O[1] oder SD I/O[2], bieten.

**CPU Leistung und Speicheraustattung** Die Anforderungen an die CPU und an die Speicheraustattung sind grundstäzlich sehr gering, da das Display nur als reines Userinterface dient und selbst keinerlei komplexe Berechnungen durchführt.
Falls aber Extras (wie z.B. Steuerung über Sprache) gefordert werden, steigt dadurch natürlich der Bedarf an die Leistungsfähigkeit der Hardware (siehe Abschnitt 3.6).

**Display** Das Display muss bei schlechten Lichtverhältnissen gut lesbar sein (hoher Kontrast). Das bedeutet, dass die Geräte eine Hintergrundbeleuchtung benötigen, um gut lesbar zu sein bzw. dass sie Reflexionen verringernd beschichtet sind, damit sie auch bei starker Sonneneinstrahlung lesbar sind.
Außerdem muss das Display groß genug zu sein, um auch aus einiger Entfernung lesbar zu sein, da es in der Nähe der Windschutzscheibe bzw. am Armaturenbrett montiert werden soll. Dadurch ergibt sich eine Entfernung zwischen Device und Fahrer von bis zu 1 m (Bei LKW kann diese noch größer sein).

**Versorgung** Die Versorgung des Devices muss entweder über das Bordnetz des Fahrzeugs (12 Volt in PKWs bzw. 24 Volt in LKWs) möglich sein oder es gibt eine Akkuversorgung, die eine Laufzeit von mehreren Stunden (zumindest 3 h) Dauerbetrieb ohne neuerliche Aufladung) erlaubt.
Generell sollten für das AVL-DRIVE System beide Möglichkeiten erfüllt werden.

**Montagemöglichkeit in Fahrzeugen** Das Gerät muss in einem Fahrzeug derart montierbar sein, dass es den Handlungsraum des Fahrenden in keinster Weise einschränkt (Sicht, Lenkung, Schaltung etc.) und sich auch während der Fahrt keinesfalls lösen kann.

**Bedienbarkeit** Die möglichen Userinteraktionen (siehe Abschnitt 3.2.1) sollten möglichst einfach vorzunehmen sein (vorzugsweise Touchscreen).

**Kosten** Der Preis für Hardware und Software (pro Gerät) sollte im Bereich von ca. 500 € liegen.

### 3.3.2 Auswahl der Hardware

Verschiedene der vorgestellten Hardwareplattformen würden die geforderten Punkte erfüllen. Vor allem Web Pads und Handheld Computers (PDAs) sind sehr gut geeignet.

[1] Compact Flash I/O (oder auch Compact Flash +) ist eine Schnittstellen definition, die ursprünglich nur für Datenträger gedacht war, aber später für allgemeine Zwecke erweitert wurde. Sie ist weitgehend kompatibel mit den PC-Card Standard von PCMCIA (Personal Computer Memory Card International Association")[12].

[2] SDIO (Input/Output) card ist eine Schnittstelle die die Funktionalität von Geräten mit SDCard Slots erweitert[69].

Für erste Entwicklungen wurde ein WindowsCE basierender PDA gewählt. Dafür gibt es folgende Gründe:

**Hohe Verfügbarkeit**

Einer der Hauptpunkte für die Auswahl dieser Plattform ist die breite Verfügbarkeit von solchen PDAs. Sie werden von verschiedensten Herstellern (z.B. Hewlett Packard, Fujitsu Siemens, Dell) produziert. Im direkten Vergleich zu den PalmOS basierenden Geräten ist die Verfügbarkeit besser.

**Grösse und Montagemöglichkeit in einem KFZ**

Die PocketPCs sind kaum grösser als das bisherig verwendete externe Display. Auch gibt es eine breite Auswahl an Halterungen für die Montage in Fahrzeugen.

**Drahtlose Schnittstellen**

Faktisch alle am Markt befindlichen PocketPCs haben eine eingebaute Bluetooth Schnittstelle. WLAN ist bei den höherpreisigen Modellen auch Standard. Zusätzlich bieten sehr viele Modelle auch Erweiterungschnittstellen, wie z.B. SD-I/O oder Compact Flash an.

**Stromversorgung**

Die Stromversorgung der PDAs in einem KFZ stellt auch kein Problem dar. Für alle Anbieter gibt es Adapter, um sie über die Zigarettenanzünder-Buchse im Auto zu versorgen. Auch ein reiner Akkubetrieb ist möglich, da bei den verwendeten Geräten die Laufzeit bei verwendeter drahtloser Kommunikation weit über einer Stunde liegt. Zusätzlich gibt es auch größere Akkus bzw. Zusatzakkus für die meisten Geräte.

**Preis**

Ein weiterer Vorteil der PocketPCs, vor allem verglichen zu den Web Pads, ist der relativ günstige Preis. Billigste Geräte gibt es bereits für unter 300 € Geräte mit Bluetooth und Wifi Austattung sind ab ca. 400 € erhältlich. PalmOS basierende Geräte gibt es zwar schon ab ca. 150 €, diese haben aber keinerlei drahtlose Schnittstellen. Palm PDAs mit drahtlosen Schnittstellen kosten ungefähr genau so viel wie vergleichbare PocketPCs.

**Entwicklungsumgebungen, SDK usw.**

Für die Wahl von PocketPCs spricht auch die gute Austattung dieser Plattform mit Tools für die Entwicklung von Software. Von Microsoft direkt gibt es ein kostenloses SDK mit umfangreicher Dokumentation für die Entwicklung in C++, Visual Basic, Visual Basic .NET und C#. Zusätzlich gibt es auch kostenlose Emulatoren und Entwicklungsumgebungen für C++ und Visual Basic. Weiters existieren auch Visual Studio Versionen, die das Erstellen von PocketPC Applikationen (Smart Devices Projects) erlauben.
Durch die Kooperation des Instituts und auch der Firma AVL mit Microsoft steht Visual Studio 2005 Professional für die Entwicklung zur Verfügung (anfangs nur als Beta-Version,

da das endgültige Produkt erst Ende 2005 auf dem Markt kam)
Auch gibt es für PocketPC eine Java Runtime, die aber kostenpflichtig ist.

### Display

Die Qualität des Displays ist abhängig von Hersteller und Modell. Vor allem die Lesbarkeit bei starker Sonneneinstrahlung variiert sehr stark bei den verschiedenen Modellen. Höherpreisige Geräte bieten jedoch Displays, die den Ansprüchen genügen.

### Leistungsfähigkeit der Plattform (CPU-Performance)

PocketPCs gibt es mit verschiedenster CPU- und Hardware-Ausstattung. Ein Vorteil der PocketPCs ist die Binärkompatibilität der verschiedenen Geräte untereinander.
Durch das Einschränken auf eine CPU Plattform (ARM-Architektur) wurde erreicht, dass PocketPC Software auf allen am Markt befindlichen Geräten läuft, ohne neu compiliert oder sonstwie angepasst zu werden.
Die verschiedenen Geräte am Markt unterscheiden sich vor allem auch durch die verwendete CPU. So haben die billigsten PDAs 200Mhz XScale Prozessoren als Gegenstück dazu bietet DELL ihr Topprodukt mit einer 624 Mhz CPU an. Auch haben die billigen Geräte normalerweise nur 64MB RAM, teurere aber 128MB oder sogar noch mehr.
Für die Kernfunktionalität des neuen Displays (Steuern und Überwachen) reicht sicherlich die CPU Performance und Speicheraustattung der billigsten Geräte am Markt.
Die Verfügbarkeit von schnellen CPUs (über 400 Mhz) ist vor allem für eine mögliche Erweiterung mit Spracherkennungssoftware wichtig.

### Verwendete Hardware

Verwendet wurde der DELL Axim X50v (bzw. X51v) (siehe Bild 3.9). Die beiden Produkte sind von der Hardware faktisch gleich, nur dass der X51v mit Windows Mobile 5.0 ausgeliefert wird und der X50v noch mit PocketPC 2003.
Gewählt wurde er vor allem wegen seines guten Displays und seiner sehr guten technischen Ausstattung und des dennoch relativ günstigen Preises (Technische Daten siehe Tabelle 3.2).

| | |
|---|---|
| **CPU** | Intel®XScale-$^{TM}$PXA270 mit 624 MHz |
| **Speicher** | 64 MB SDRAM<br>256 MB Flash ROM |
| **Display** | Berührungsempfindliches und transflektives<br>16-Bit 3,7-Zoll-TFT-VGA-Display<br>Auflösung: 640 x 480 Pixel mit 65.536 Farben |
| **Betriebs-system** | Microsoft®Windows Mobile$^{TM}$5.0<br>mit Windows Media Player 10 Mobile |
| **Erweiterungsplätze** | 1 x CompactFlash Type II-Karte (3,3 V)<br>1 x Secure Digital-/SDIO Now-/MMC-Karten (3,3 V) |
| **Integrierte Schnittstellen** | Wi-Fi (802.11b)<br>Bluetooth$^{TM}$<br>IrDA 1.2<br>36-poliger Docking/Sync-Anschluss<br>3,5-mm-Anschluss für Kopfhörer/Headset |
| **Akku** | Lithium-Ionen-Akku mit 1100 mAh, |
| **sonstiges** | Intel® 2700G-Multimediabeschleuniger<br>mit 16 MB Grafikspeicher |

Tabelle 3.2: Technische Daten des Dell Axim x51v[41]


Abbildung 3.9: Dell Axim X51V [41]

## 3.4 Bewertung der Verbindungstechnologie

### 3.4.1 Anforderungen an die Kommunikationstechnologie

Im Folgenden werden Anforderungen an die kontaktlose Übetragungstechnologie in Zusammenarbeit mit dem AVL-DRIVE System erläutert.

#### Reichweite

Die Technologie muss auch dann noch funktionieren, wenn das Notebook z.B. im Kofferraum liegt und das Display an der Windschutzscheibe montiert ist. Das sind bei normalen PKWs ca 2m Abstand. Im *Worst case* wären es 18,75 Meter (das ist die maximale Länge eines Fahrzeuges in der EU [33, § 4 Abs. 6 KFG], wobei dies nur relevant ist, wenn Sensoren drahtlos angebunden werden). Für die Verbindung eines Monitoring Device mit dem Notebook hat diese Distanz keine Bedeutung, denn auch bei einem LKW ist das Notebook im Fahrgastraum untergebracht. Dadurch ergibt sich wieder eine maximale Entfernung von einigen Metern. Ebenso muss es auf alle Fälle funktionieren, wenn keine Sichtverbindung vorhanden ist.
Fast alle vorgestellten Übertragungstechniken erfüllen diese Anforderungen. Die meisten übertreffen es sogar um ein Vielfaches. Sicherlich muss auch berücksichtigt werden, dass die angegebenen Entfernungen nur theoretische Werte sind, dennoch haben alle genügend Reserven, um die geforderten Entfernungen (ein paar Meter) zu überbrücken.
Ausnahme sind aber IrDA (siehe Abschnitt 2.2.7) und NFC (siehe Abschnitt 2.2.8). Beide haben nur eine sehr kurze Reichweite (einige cm) die nicht ausreichend ist. Zusätzlich benötigt IrDA auch noch eine Sichtverbindung zwischen zwei Geräten was für diese Anwendung nicht garantiert werden kann. Aus diesem Grund können diese Technologien nicht für diese Aufgabe verwendet werden.

#### Übertragungsraten

Hier muss unterschieden werden, ob ein Sensor oder ein Monitoring Device angebunden werden soll. Dadurch ergeben sich unterschiedliche Anforderungen an die Übertragungsraten.

**Anforderungen für Sensoren:** Um eine Mindestübertragungsrate zu bestimmen, wird von folgenden Annahmen ausgegangen:

- Ein Messwert eines Sensors entspricht einen IEEE *double float* Wert. Das entspricht einer Länge von 8 Bytes[61, Seite 8].
- AVL-DRIVE verwendet in der Software (siehe Abschnitt 3.1.2) eine maximale Samplingrate von *1 kHz*, wobei das eher die Ausnahme ist, im Normallfall sind es nur 100 Hz.

Aus diesen zwei Punkten kommt man auf eine geforderte minimale Übertragungsrate von *8 KB* ($8\,Bytes \times 1kHz = 8000\,Bytes/Sec$) pro Sensor. Falls mehrere Sensoren verwendet werden, müssen deren Nettodatenraten addiert werden, da sie sich bei drahtloser Übertragung das Medium (die Luft, Äther) teilen. Auch wird ein Übertragungsprotokoll

benötigt, um die gesendeten Werte sinnvoll interpretieren zu können (z.B. Sensor ID, Timestamp, usw.). Weiters ist auch ein Anwendungsszenario denkbar, in dem der Sensor eine viel höhere Samplingrate besitzt als die oben erwähnten 1kHz, aber die Daten nur mit einer Wiederholrate von 1kHz (oder auch weniger) schickt. Stattdessen werden dafür aber Mittelwert und Standardabweichung gesendet (das wären dann $2 \times 8Bytes = 16Bytes$).
Andererseits verwendet AVL-DRIVE für die meisten Messwerte nur eine Samplingsrate von 100 Hz, wobei sogar noch geringere Raten vorstellbar sind (z.B. für einen Öltemperatur-Sensor reicht eine Samplingrate von 1Hz).
Die genaue Abschätzung der benötigten Datentransferrate ist nur möglich, wenn die genaue Sensorkonfiguration bekannt ist. Davon hängt dann auch die verwendbare Technologie ab. Beispielsweise ist die Wireless TTP/A Technik (siehe Abschnitt 2.2.5) für Signale mit hohen Samplingraten nicht ausreichend. Für einen, wie vorher bereits erwähnten, Öltemperatursensor genügt sie aber sicher.

**Anforderungen für das Userinterface:** Für das Userinterface kann nicht mehr von einem solch simplen Datenmodell wie vorher erwähnt ausgegangen werden, da viel mehr Daten benötigt werden. Andererseits ist aber die Updaterate viel geringer, da die Anzeige für einen Menschen gedacht ist und es daher keinen Sinn macht Messwerte öfters als 10 mal pro Sekunde zu erneuern. Ein Mensch kann Daten mit höherer Aktualisierungsrate nicht so schnell erfassen.
Durch Gespräche mit den Mitarbeitern der AVL und Vorführung von Prototypen konnten folgende Anforderungen an Übertragungsraten und Darstellung definiert werden:

- Es sollen maximal 10 verschiedene Messwerte gleichzeitig angezeigt werden (im Normallfall sollten es 3-4 verschiedene Messwerte sein).
- Messwerte sollten mit maximal 5Hz am Display upgedatet werden.
- Ein Messwert ist ein double float Wert (Länge 8 Bytes) plus einem Identifier (short int Wert länge 2 Bytes). In Summe hat ein einzelner Messwert eine Länge von 10 Bytes.
- Zusätzlich zu den Messwerten müssen noch erkannte „Betriebszustände“ (in Englisch Events) angezeigt werden (siehe Abbildung 3.6). Diese bestehen aus einem sogenannten *Main Event*, einem *Sub Event* und mehreren *Detail Events*. Für die Anzahl der Detail Events gibt es keine Einschränkung, zur Zeit wird aber kein Betriebszustand verwendet, der mehr als 20 Detail Events hat.
- Jedes dieser Events besteht aus einem Namen (mit maximal 200 Zeichen $\hat{=}$ 200 Bytes), einer Benotung mit einem Wert zwischen 1.0 und 10.0 (ausgenommen Main Event, welches keine Note besitzt) und einer Start- und Endzeit (relativer Wert, Sekunden seit Messungsbeginn). Die Benotung und die zwei Zeitpunkte sind ein *double float* Wert mit 8 Byte Länge.
- Ereignisse (Betriebszustände, Events) treten im Normallfall immer mit einen Abstand von einigen Sekunden auf. In Einzelfällen kann es vorkommen, dass mehrere Betriebszustände pro Sekunde auftreten. Dies ist die Ausnahme und in einem solchen Fall stellt es kein Problem dar, falls die Anzeige sich etwas (einige Sekunden)

verzögert. Zur Berechnung der Übertragungsrate wird von maximal einem Ereigniss pro Sekunde ausgegangen.

Daraus ergibt sich eine maximale Übertragungsrate zum Userinterface.

**Maximale Übertragungsrate für Messwerte:**
$\ddot{U}mess_{max} = AnzMesswerte_{max} \times Messwertegrösse \times SendeFrequenz_{max}$
$\ddot{U}mess_{max} = 10 \times 10 \times 5 = 500\ Bytes/sek$
**Maximale Übertragungsrate für Betriebszustände:**
$Eventgrösse_{max} = Maineventgrösse_{max} + Subeventgrösse_{max} + 20 \times Detaileventgrösse_{max}$
$Eventgrösse_{max} = 216 + 224 + 20 \times 224 = 4920\ Bytes$
**Maximale Übertragungsrate:**
$\ddot{U}max = Eventgrösse_{max} \times SendeFrequenz_{max}$
$\ddot{U}event_{max} = 4920 \times 1 = 4920\ Bytes/sek$
$\ddot{U}_{max} = \ddot{U}max + \ddot{U}mess_{max} = 500 + 4920 \approx 5500\ Bytes/sek$

Daraus ergibt sich eine Nettoübertragungsrate von ca. 5,5 KBytes/sec. Da immer ein Overhead durch das eingesetzte Übertragungsprotokoll auftritt, verdoppeln wir das Ergebnis, um dadurch auch einen Sicherheitspuffer zu erhalten.
Dadurch ergibt sich eine maximale Übertragungsrate von 11 KBytes/sec.
Auch für diesen Zweck haben alle getesteten Übertragungstechniken genug Reserven.

**Störanfälligkeit (Quality of Service, QoS):**

Wichtig für diesen Punkt ist die Unempflindlichkeit gegenüber Störimpulsen vom KFZ. Zusätzlich kann es aber auch zu Interferenzen (wie beispielsweise mit anderen drahtlosen Übertragunstechnologien) kommen
Auch hier sind die Anforderungen an die Übertragung nicht sehr hoch. Vereinzelte Aussetzer bei der Übertragung können akzeptiert werden. Viel schlimmer jedoch wäre die Verfälschung von Messwerten.

**Verfügbarkeit von Hardware und Softwarelösungen:**

WLAN nach dem 802.11 Standard, sowie Bluetooth sind sehr weit verbreitet und können ohne Probleme von verschiedensten Herstellern in verschiedensten Ausführungen bezogen werden. Interessant sind die beiden vor allem für das Userinterface, da viele mobile Endgeräte (siehe Abschnitt 2.3) diese Technologien bereits eingebaut haben. Für den Einsatz von Smarten Sensoren wird die Verwendung einer speziell darauf ausgerichteten Übertragungstechnologie wie beispielsweise ZigBee von Vorteil sein. Jedoch ist hier die Verfügbarkeit an Endgeräten, die eine solche Funkschnittstelle zur Verfügung stellen, noch nicht gegeben und der Implementierungsaufwand damit höher.

### 3.4.2 Auswahl der Verbindungstechnologie

Durch die Festlegung auf einen PocketPC als Zielplattform bleiben nur mehr WLAN nach IEEE 802.11 und Bluetooth übrig, da diese in vielen Geräten bereits fix eingebaut sind. Dies gilt natürlich auch für die PC Seite, denn die meisten Notebooks haben bereits WLAN

eingebaut und viele sogar auch Bluetooth. Für andere Technologien (wie z.B. ZigBee) müssten spezielle Erweiterungskarten gekauft oder sogar selbst entwickelt werden. Darüberhinaus müsste das nicht nur auf der PDA Seite sondern auch auf der PC Seite geschehen. Das würde aber den Implementierungsaufwand vervielfachen.
Zusätzlich haben WLAN und Bluetooth den Vorteil, dass beide bereits IP-Unterstützung anbieten. Dadurch kann Socket-Programmierung verwendet werden kann. Weiters ergibt sich dadurch auch eine gewisse Hardware-Unabhängigkeit, da bei Verwendung von Sockets, nicht dezidiert für eine physikalische Netzwerktechnologie entwickelt wird.
Aus diesem Grund wird eine Verbindung verwendet, die das IP Protokoll als Grundlage besitzt. Darunter fallen z.B. TCP, UDP aber auch High Level Kommunikationsprotokolle, wie CORBA [71, CORBA] oder auch DCOM[71, DCOM]. Zusätzlich erlaubt es auch auf kabelbasierende Lösungen zurückzugreifen (wenn z.B. der Stromverbrauch zu hoch ist oder es bei gewissen Anwendungsszenarien zu viele Störungen für den Betrieb mit drahtlosen Netzwerken gibt).

## 3.5 Auswahl der Entwicklungsumgebungen

### Verfügbare Entwicklungsumgebungen

Durch die Festlegung auf Microsoft Mobile bzw. PocketPC basierende Geräte stehen folgende Entwicklungsumgebungen bzw. Sprachen zur Verfügung :

- Java
  Es gibt verschiedenste Anbieter von Java für PocketPC, unter anderem IBM[38], Insignia (jetzt Esmertec[2]).
- C++
  Hier bietet Microsoft folgende Entwicklungswerkzeuge an:
  - eMbedded Visual C++ 4.0,
    welches gratis zur Verfügung steht[18].
  - Visual Studio 2005 (kurz VS2005), wobei hiervon zumindest die Standard Edition benötigt wird.
- Visual Basic
  Auch hier gibt es von Microsoft drei Varianten
  - eMbedded Visual Basic 3.0, welches in den "eMbedded Visual Tools 3.0" inkludiert ist, die es auch gratis zur Verfügung gibt[19].
  - VS 2005, wobei hier zumindest die Standard-Edition benötigt wird.
  - VS 2003, hiervon wird zumindest die Professional-Edition benötigt.

  *Achtung:* die Versionen sind nicht gleichwertig, da bei eMbedded Visual Basic noch eine ältere Version von Visual Basic (auch VB 6.0 genannt) verwendet wird, bei VS 2003 (und auch 2005) aber bereits Visual Basic .NET (manchmal auch VB 7.0 bzw. VB 8.0 oder neuer) verwendet wird. Grundsätzlich kann gesagt werden, dass VB.NET eine komplette Neuentwicklung von VB ist und deswegen auch nicht rückwärts kompatibel ist[28].
  VS 2005 unterscheidet sich von VS 2003 hauptsächlich durch die Unterstützung von .NET 2.0 anstelle von .NET 1.1.
- C#
  Zwei Varianten werden von Microsoft angeboten:
  - VS 2005, wobei hier zumindest die Standard-Edition benötigt wird
  - VS 2003, hiervon wird zumindest die Professional-Edition benötigt.

  Wie bereits vorher erwähnt unterstützt VS 2005 zusätzlich zum .NET Framework 1.1, welches es auch in VS 2003 gibt, auch noch .NET 2.0
- Sonstige
  Es existieren noch diverse „exotische" Entwicklungsumgebungen, wie zum Beispiel GoDB von Consigntech[13], die aber entweder für spezielle Anwendungsgebiete (wie z.B. Datenbankanwendungen) oder spezielle Hardware (siehe Abschnitt 2.5.2) angeboten werden

### Auswahl der Entwicklungsumgebung

Gegen Java spricht die eher schlechte Unterstützung auf der PocketPC Plattform. Beispielsweise ist Java Swing auf PocketPC kaum nutzbar[8]. Zusätzlich findet man allgemein weniger Information zum Entwickeln von Anwendungen in Java (vermutlich darin begründet, dass Microsoft dies nicht fördern will). Darüberhinaus ist es ungleich schwieriger auf eingebaute Funktionalitäten (wie z.B. das Displaykeyboard) zuzugreifen, da dafür „Low Level“ API Aufrufe benötigt werden. Außerdem fällt eine Lizenzgebühr bei den meisten Anbietern von Java-Entwicklungsplattformen an.

eMbedded Visual Basic 3.0 ist auch nicht sinnvoll zu verwenden, da es noch für die PocketPC 2002 Plattform ausgelegt ist. Darin entwickelte Programme laufen zwar auch auf PocketPC 2003 bzw. Windows Mobile 5.0, davon wird aber auch von Microsoft selbst abgeraten.
In die engere Auswahl kommen dadurch nur mehr eMbedded Visual C++ und .NET basierende Projekte (Visual Basic und C# sind beides .NET basierend und dadurch einander sehr ähnlich[45], deswegen werden sie gemeinsam betrachtet.)
Ein Vorteil, gleichzeitig aber auch einer der größten Nachteile von eMbedded Visual C++, ist, dass man sich sehr mit der WindowsCE (PocketPC) API-Programmierung beschäftigen muss. Diese API ist zwar der normalen Windows API (WIN32) nachempfunden, aber mit ihr nicht 100% kompatibel. Beispielsweise fehlen ihr einige Standardfunktionen für Zeit und Datumsberechnung (wie z.B. `strdate` oder `ctime`). Dadurch wird es schwieriger portablen Code zu schreiben. Der Hauptgrund, der gegen die Entwicklung mit C++ spricht, ist dass man mit der Microsoft Foundation Classes (MFC) arbeiten muss und von dieser nur ein Subset der Funktionalität zur Verfügung steht. Man ist sehr bald gezwungen auf native Windows API Funktionen zurückzugreifen. Auch ist die MFC eine Entwicklung aus dem Beginn der 90er Jahre, in der die objektorientierte Programmierung noch nicht weit verbreitet war. Dadurch ist die MFC recht kompliziert bzw. atypsich zu benutzen (vor allem in Vergleich[9] zu modernen Toolkits wie z.B. die STL (Standard Template Library) oder QT(Library von Trolltech[72])).

Aus obigen Gründen wurde als Basis .NET gewählt. Da bereits Erfahrung mit C# vorhanden war (von Diplomanden und auch seitens der Fa. AVL) wurde dies als Sprache Visual Basic vorgezogen.
Microsoft brachte bereits Ende 2004 die erste öffentliche Beta von VS 2005 heraus. Die Firma AVL (im speziellen auch die Abteilung DV, für die dieses Projekt durchgeführt wird) hat in näherer Zukunft vor, auf diese Plattform umzusteigen. Aus diesem Grund wurde diese Diplomarbeit gleich als Testprojekt für VS 2005 ausgesucht (weil diese Diplomarbeit als in sich geschlossen und nicht kritisch bei Terminverzögerungen eingestuft wurde). Anfangs wurde die VS 2005 Beta 1 getestet. Diese stellte sich als nicht nutzbar heraus, da es relativ oft zu Programmabstürzen kam und auch sonst sehr viele Fehler auftraten (gleiches Programm funktioniert, beim nächsten kompilieren wieder nicht, Dokumentation war unvollständig oder falsch, usw.). Die Beta 2, die am 17. April 2005 vorgestellt wurde, war zwar auch noch nicht zu 100% stabil (sie stürzte mehrmals in der Woche ab), man konnte aber damit bereits produktiv arbeiten. Zu den wichtigsten Verbesserungen in VS 2005 zählen für dieses Projekt die Einführung des .NET Compact Frameworks (.NET CF)

2.0. Dies bietet vor allem eine bessere Unterstützung und Kompatibilität mit dem Standard .NET Framework. Auswirkung hat dies insbesondere bei den Threads und für die Modularisierung der Oberfläche die Unterstützung von User Controls[46].
Auf die Vorteile dieser Features wird im Punkt Design noch genauer eingegangen.

## 3.6 Spracherkennungstools

### 3.6.1 Anforderungen an die Sprackerkennung

Für die Aufgabenstellung ist es nicht nötig eine komplette Spracherkennung zu haben. Es reicht bereits eine simple Erkennung von einigen Basiskommandos, wie z.B. Start/Stop usw. Andererseits muss die Software eine fortlaufende Spracherkennung unterstützen. Die Lösung, bei der nach einem Click die Aufnahme und Erkennung gestartet wird, ist nicht sinnvoll für das Anwendungsszenario, da es sehr wenige Kommandos gibt (ca. 5 wichtige, die anderen Befehle werden nur selten benötigt), und es für den Anweder kaum ein Vorteil ist, da er gleich durch direktes Auswählen des gewünschten Buttons die entsprechende Aktion ausführen kann.

### 3.6.2 Vorgehensweise

Da es den Rahmen dieser Arbeit sprengen würde, wird keine eigene Spracherkennung entwickelt. Es wird vielmehr eine Übersicht über vorhande Techniken, Programme und Software Libarys gegeben.

Richtige Spracherkennungstools gibt es viele verschiedene auf dem Markt. Wenn man aber welche sucht, die auf den PDAs laufen, wird die Auswahl bedeutend geringer, da es ungemein schwieriger ist, eine gute Erkennnungsrate mit der beschränkten Hardware zu garantieren. Zum Beispiel verlangt das Produkt „Dragon Naturally Speaking“ zumindest eine 500 Mhz CPU + 256 Mb Ram [67]. Eine derartige Hardwareaustattung gibt es natürlich bei mobilen Geräten nicht. Üblicherweise haben sie nur ca. 400 Mhz (meistens ohne eigene FPU) und nur 64 MB Ram, wobei dieser auch gleichzeitig als Hard Drive dient.
Drei verschiedene Hersteller wurden gefunden, deren Tools diese Aufgabe erfüllen.

- Conversay [14]
- Fonix [15]
- Nuance (vormals ScanSoft) [59]

Diese Produkte gibt es nur für PocketPC basierende Geräte, da sie alle relativ hohe Hardwareanforderungen stellen, (mindestens 400 Mhz CPU und mindestens 32 MB verfügbaren Speicher). Für Palms wurde kein Anbieter gefunden, der die Anforderungen erfüllt. Die Implementierung einer Spracherkennung wurde verschoben, da sich herausgestellt hatte, dass die Kosten für diese Werkzeuge relativ hoch sind (mehrere 1000 $) und es noch keinen echten Bedarf dahingehend gibt.

## 3.7 Bemerkung zur Vorgehensweise bei Design und Implementierung

Das hier dokumentierte Design und die Implementierung repräsentieren den Endstand der Entwicklung.
Bei der Entwicklung für dieses Tool wurde frei nach dem Spiralmodell mit Unterstützung von Prototypen vorgegangen.

### 3.7.1 Spiralmodell

Das Spiralmodell (es wird auch inkrementelles bzw. iteratives Vorgehensmodell genannt) ist ein Vorgehensmodell in der Softwareentwicklung, das im Jahr 1978 von Barry W. Boehm beschrieben wurde[10]. Es ist offen für bereits existierende Vorgehensmodelle. Das Management kann immer wieder eingreifen, da man sich spiralförmig voran entwickelt.

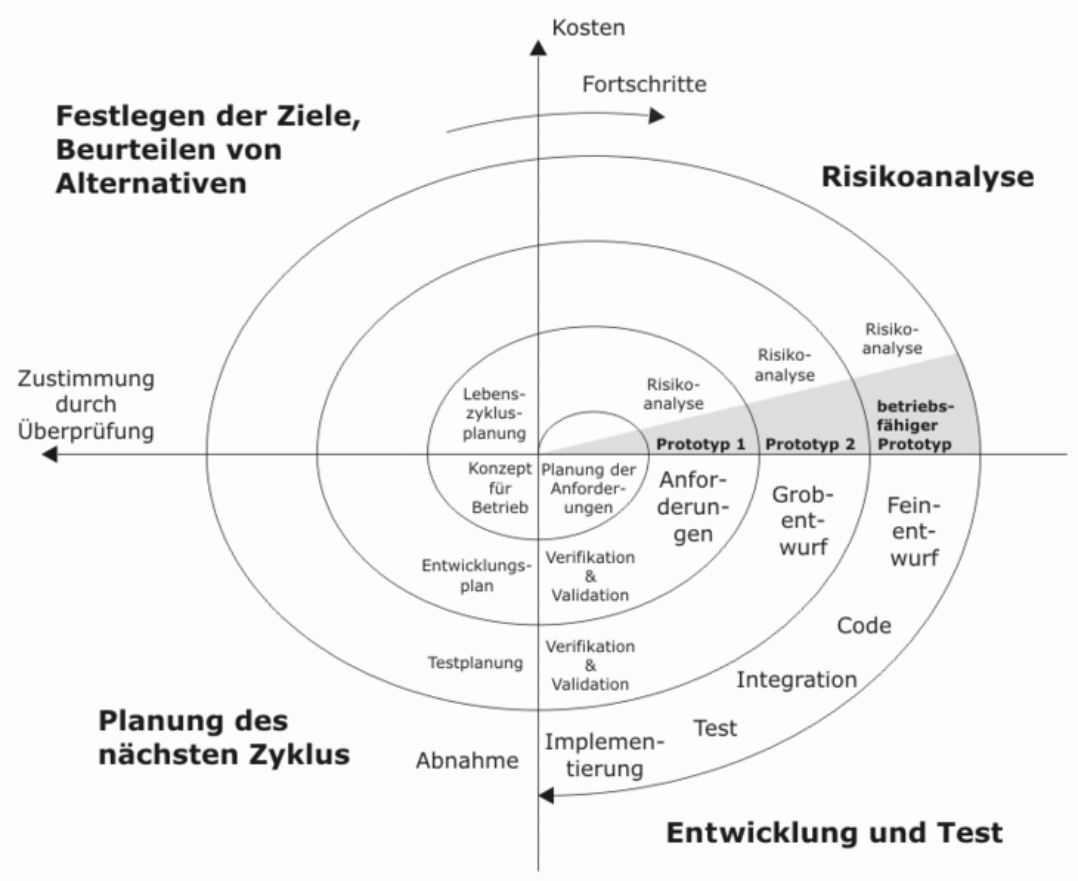



Abbildung 3.10: Spiralmodell nach Boehm (1978) [60]

Das Spiralmodell fasst den Entwicklungsprozess im Software-Engineering als iterativen Prozess auf, wobei jeder Zyklus folgende Aktivitäten enthält (siehe Abbildung 3.10):

1. Festlegung von Zielen, Alternativen und Rahmenbedingungen

2. Evaluierung der Alternativen und das Erkennen und Reduzieren von Risiken
3. Realisierung und Überprüfung des Zwischenprodukts
4. Planung der Projektfortsetzung.

Am Ende jeder Windung der Spirale steht ein Betrachten des Projektfortschritts (engl. Review). Dabei wird auch der Projektfortgang geplant und verabschiedet.

### 3.7.2 Prototyping

Allgemein ist ein Prototyp in der Technik ein Vorab-Exemplar für eine spätere Massenproduktion, um die verschiedensten Eigenschaften erproben zu können.
In der Softwareentwicklung versteht man unter Prototyping, dass schon recht früh im Entwicklungsprozess Prototypen der Software implementiert werden. Diese Prototypen sind keine fertigen Softwareprodukte, sondern dienen nur dazu, einzelne Punkte der Aufgabenstellung bzw. des Designs zu überprüfen, Lösungsansätze zu testen, Erfahrungen zu sammeln und dem Endbenutzer schon recht früh eine Vorstellung über die Software zu geben. Dieser letzte Ansatz wird sehr oft für die Entwicklung der Benutzeroberfläche (UI) verwendet, um den späteren Anwender rechtzeitig Möglichkeiten für ein Feedback zu geben.

#### Vorteile

- Risiko einer Fehlentwicklung sinkt, denn die Anforderungen der Anwender können laufend präzisiert und verifiziert werden.
- Unbeabsichtigte Wechselwirkungen zwischen einzelnen Komponenten des Produkts können früher erkannt werden.
- Der Fertigstellungsgrad ist besser verifizierbar.
- Die Qualitätssicherung kann frühzeitig eingebunden werden.

#### Nachteile

Prototyping verführt dazu, das Pflichtenheft nicht detailliert genug bzw. nicht korrekt zu erheben und später auch nicht gut zu dokumentierten. Das kann zu einem erheblich längeren Entwicklungsprozess führen.

### 3.7.3 Vorgehensweise

Nach der Feststellung des *IST* Zustands von AVL-DRIVE wurde grob spezifiziert, welche Ziele erreicht werden sollen. Danach wurde ein erster Prototyp erstellt, um den Benutzern zu zeigen, welche Erweiterungen erstellt werden können. Ein Beispiel eines solchen Prototypen sieht man in Abbildung 3.11. Einer der wichtigsten Gründen für das Erstellen von Prototypen lag auch darin begründet, dass die Benutzer und Projektleiter der Firma

AVL sich zuerst nicht wirklich vorstellen konnten, wie man mit DRIVE-Remote arbeiten soll. Aus diesem Grund bietet sich das Prototypen- basierende Entwicklungsmodell an, um schon von Beginn des Designprozesses an Rückmeldungen zu erhalten, welche Elemente bei einem Userinterface gewünscht werden.


Abbildung 3.11: Frühe Version eines UI Prototypen

## 3.8 Grobentwurf(Architectural-Design)

Im Kapitel *Anforderungen* (siehe Abschnitt 3.2) wurden die benötigten Punkte aus der Sicht des Anwenders beschrieben. Hier werden die Probleme technisch genauer spezifiziert und auch schon Lösungen vorgestellt, ohne zu sehr ins Detail zu gehen.

Wie bereits im Lastenheft festgehalten, wird hier nur die PDA bzw. Userinterface Seite entworfen. Änderungen bzw. Erweiterungen von AVL-DRIVE auf der PC-Seite werden hier nicht festgehalten. Ausnahme sind natürlich Bedingungen, die direkt auch die PDA Seite betreffen.

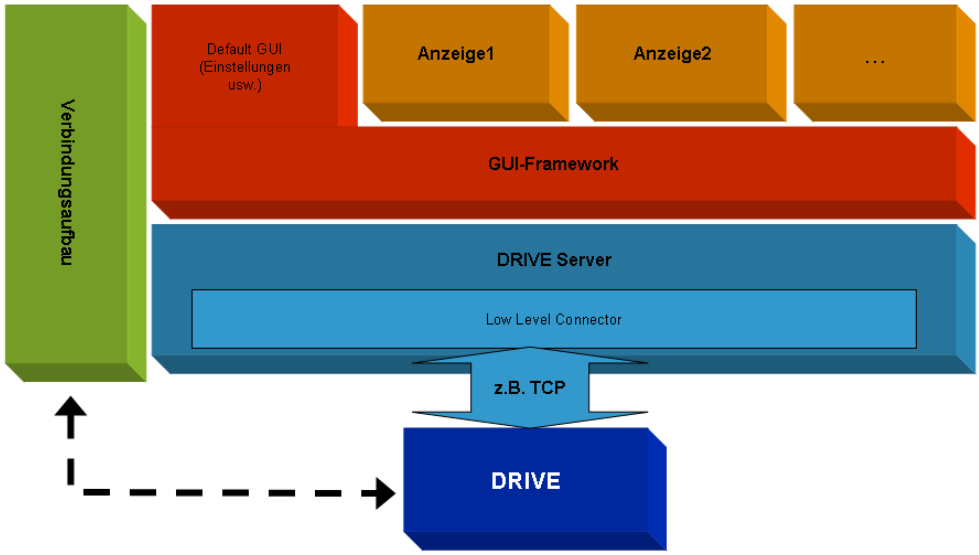



Abbildung 3.12: Grobdesign von DRIVE-Remote

Die Abbildung 3.12 soll einen groben Überblick liefern, wie DRIVE-Remote aufgebaut ist.
Zur Erfüllung des Punktes *Modularisierung des Verbindungsteils* (siehe Abschnitt 3.2.2), wird die Kommunikation mit AVL-DRIVE gekapselt und abstrahiert. Dieses Modul (DRIVE-Server in der Abbildung 3.2.2) beinhaltet alle Teile, die nötig sind, um mit AVL-DRIVE eine Verbindung aufzubauen, es fernzusteuern (siehe *Fernsteuerung von AVL-DRIVE* Abschnitt 3.2.1), Daten zu empfangen und zu senden. Das Modul Low-Level Kommunikation dient zur Kapselung der Netzwerkzugriffsfunktionalität (z.B Socket Programmierung). Ein weiterer Effekt dieser Kapselung ist eine zusätzliche Modularisierung des Verbindungsaufbaus. Damit könnte z.B. eine auf TCP basierende Verbindung relativ leicht durch eine UDP Verbindung ausgetauscht werden.

Das GUI Framework kontrolliert die Basisfunktionalität der Oberfläche (darunter fallen der Verbindungsaufbau und falls nötig auch IP-Adressen und Port Eingabe). Zusätzlich soll es auch erlauben, einzelne Anzeigenmodule (in der Abbildung 3.12 als „Anzeige1“ usw.)

einzubinden. Anfänglich sollte dies dynamisch bzw. über Konfigurationsfiles erfolgen. Dazu müsste man ein Plug-In Framework erstellen um zusätzliche GUI-Elemente während der Laufzeit zu laden. Für diese Aufgabe gibt es detaillierte Informationen von Microsoft selbst (Siehe [11] bzw. [64]) aber auch von anderen Quellen (Siehe [44]). Bei der Entwicklung stellte sich heraus, dass es gar nicht nötig ist, wie in der Einleitung gefordert (siehe Abschnitt 1.2), das Umstellen des GUI über Konfigurationsfiles zu erlauben. Für fast alle Kunden reicht ein Standard GUI, und falls Sonderwünsche erfüllt werden müssen, so ist es kein größeres Problem, dafür den Code zu ändern und eine neue Version zu releasen. Trotzdem wird die Anzeige in Module aufgeteilt, um den Code lesbar zu machen und solche Änderungen, falls gewünscht, einfach durchzuführen zu können.

Das Modul Verbindungsaufbau bzw. Konfiguration ist zuständig für die Details hinsichtlich der Verbindung (IP-Adresse, Port usw.) bzw. für die Kontrolle des Verbindungsaufbaus. Der Grund dafür ist diese Funktionalität nicht im GUI Modell zu implementieren um dadurch eine weitere Verbesserung der Modularisierung zu erreichen.

### 3.8.1 AVL-DRIVE-Schnittstelle

Für die Verbindung mit AVL-DRIVE wird eine TCP (bzw. auch UDP) Verbindung verwendet. Gründe, die für diese Entscheidung sprechen, waren:

1. AVL-DRIVE bietet bereits eine rudimentäre TCP-Anbindung für andere Programme an.
   Diese muss zwar für DRIVE-Remote angepasst und erweitert werden, aber eine Basisfunktionalität ist bereits vorhanden.

2. Die Entwickler der Abteilung DV hatten bereits negative Erfahrung mit DCOM gemacht, wenn verschiedene Versionen verwendet werden (VC++ 6.0 mit VS 2003). Aus diesem Grund wurde für die bereits vorhandenen Schnittstellen nicht DCOM verwendet.

3. Simple Object Access Protocol (SOAP)[75, Seite 251] oder andere High Level Übertragungsprotokolle wie z.B. CORBA haben den Nachteil, dass sie teilweise von VS 2005 unterstützt werden, aber nicht von Visual Studio 6.0. Deswegen würden zusätzliche Software Bibliotheken benötigt werden, die wieder extra Kosten verursachen würden.

4. Die Komplexität der übertragenen Daten ist recht gering. Dadurch kann ohne weiteres auch ein eigenes Protokoll entwickelt werden.

Die bereits vorhandene Schnittstelle war dazu ausgelegt, dass eine DRIVE-Messung von einem anderen Programm gestartet wird, nach einiger Zeit wieder gestoppt und danach Ergebnisse (Drivability Noten) über die Schnittstelle abgefragt werden. AVL-DRIVE ist in diesem Fall rein passiv (Server). Auch der Code für diese Kommunikation ist darauf ausgelegt, dass AVL-DRIVE auf bestimmte Ereignisse reagiert.
Für DRIVE-Remote werden aber auch während der Messung Werte benötigt, da es zum Überwachen einer Messung dienen soll. Zusätzlich sollte AVL-DRIVE die Werte von sich

aus schicken, ohne dass ein Client dauernd nachfragen muss (englisch: polling).
Deswegen bietet sich eine zusätzliche Schnittstelle (Port) an, über die Messdaten während einer Messung verschickt werden.
Das erlaubt weiters auch die Aufteilung der Kommunikation hinsichtlich Übertragungssicherheit in einen „sicheren“ und einen „unsicheren“ Kanal. Für einige Messungen (beispielsweise die Drehzahl) stellt es kein Problem dar, wenn einige Messdaten während der Übertragung verloren gehen, denn einzelne Pakete würde der Benutzer nicht erkennen können. Andererseits sollen die Kommandos, die an AVL-DRIVE abgesetzt werden, sicher ankommen.
Weiters sollte die Messdatenübertragung möglichst wenig zusätzlichen Overhead produzieren.

Aus diesen Gründen wird eine zusätzliche Schnittstelle eingefügt. Dafür bietet sich UDP an, da es geringen Overhead hat und für diese Aufgabe relativ leicht zu implementieren ist, denn die zu übertragenden Pakete liegen weit unter der maximalen Grösse von 64 KByte (siehe Abschnitt 3.4.1, es ist kein Aufsplitten von Datenpaketen notwendig).

## 3.9 Detailliertes Design

### 3.9.1 Übertragungsprotokoll

Das Übertragungsprotokoll gliedert sich in zwei Punkte:

1. Kommando-Kanal, TCP-basierte Kommunikation
   DRIVE-Remote sendet Daten und Kommandos an AVL-DRIVE
2. Messdaten-Kanal (Rückkanal), UDP-basierte Kommunikation
   AVL-DRIVE sendet Messwerte und Eventdaten an DRIVE-Remote

#### Kommando-Kanal (TCP)

Bei diesem Kanal ist AVL-DRIVE der Server, DRIVE-Remote der Client und aktive Teil. AVL-DRIVE reagiert nur auf Nachrichten vom Client, wird aber niemals eine Kommunikation von sich aus anfangen.
Nachdem der Client eine Nachricht an den Server geschickt hat, erwartet er immer eine Antwort. Im einfachsten Fall besteht sie nur aus dem Text „Kommando succeeded“. Bei Fehlern beginnt der Antworttext mit dem Text „ERROR:“ und danach folgt eine Beschreibung des Fehlers. Der Client wartet nur eine gewisse Zeit auf die Antwort. Die Zeitspanne für diesen *Timeout* ist zur Zeit fix auf 30 Sekunden codiert. Dieser Wert hat sich während der Entwicklung und bei Tests als sinnvoll herausgestellt. Er kann aber auch angepasst werden. Die Zeitspanne darf aber nicht zu hoch angesetzt werden, da der Benutzer auf eine Rückmeldung von seinem Kommando wartet.
Falls mehr als nur eine Erfolg/Misserfolgs-Nachricht als Ergebnis erwartet wird (siehe Beschreibung der Nachricht GETVALUE), enthält der Text den übertragenen Wert (auch als Klartext). Dies ist zur Zeit nicht implementiert (bzw. nur teilweise vorhanden), da es nicht benötigt wird.

#### Daten-Kanal (UDP)

Über diesen Kanal werden die Daten, die von DRIVE-Remote benötigt werden um Information während der Messung anzuzeigen, gesendet, aber auch allgemeine Information wie der Start bzw. das Ende einer Messung (beispielsweise durch Abbruch wegen Messfehlern), oder auch, dass AVL-DRIVE beendet wurde, werden über diese Verbindung gesendet. Hier wartet DRIVE-Remote nur auf den Empfang von Nachrichten und sendet keine Daten an AVL-DRIVE zurück.

### 3.9.2 Userinterfaces

Um die Punkte „Einfaches Userinterface“ (Abschnitt 3.2.2) und „Modularisierung des Userinterfaces“ (Abschnitt 3.2.2) zu erreichen, wurde der Grobaufbau des UI wie in Abbildung 3.13 gewählt.
Das Userinterface wurde in zwei Teile gegliedert. Rot umrandet ist die Kernkomponente, die immer vorhanden ist. Hier befindet sich der Button zum Starten bzw. Stoppen einer Messung. Dieser wurde so groß gemacht, um ihn einfach mit dem Finger betätigen zu

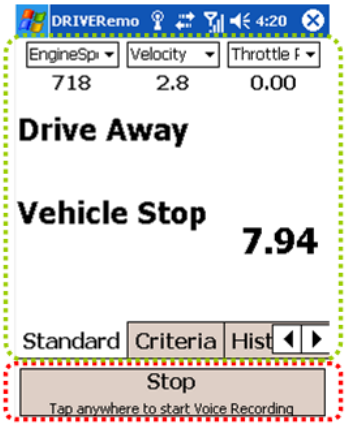



Abbildung 3.13: Standard Ansicht von DRIVE-Remote

können.
Der größte Teil der Oberfläche ist für die Informationsanzeige reserviert. Dafür wurde ein sogenanntes *Tab-Sheet Control* verwendet, welches erlaubt zwischen verschiedenen möglichen Anzeigen umzuschalten. Zusätzlich zu den drei Standard-Ansichten wurde noch ein Konfigurations-Tab eingefügt, auf dem man die DRIVE-Remote relevanten Einstellungen vornehmen kann. Zur Zeit gibt es nur ein Eingabefeld für die PC Adresse (IP-Adresse oder PC-Name) und den Port, der aber bis jetzt nur auf einen fixen Wert gesetzt ist. Die Tab-Felder wurden auch groß genug gestaltet, um mit Finger bedienbar zu sein.
An der Oberseite ist der *Taskbar* des PocketPC OS zu sehen. Dieser vom Betriebssystem verwaltete Bereich enthält neben dem Namen des im Vordergrund laufenden Programms den Startbutton (Windows-Symbol ganz links), Uhrzeit, diverse Status-Symbole (wie z.B. Zustand der drahtlosen Verbindung usw.) sowie auch den Minimierungs-Button (das Symbol ganz rechts), welche die aktuelle Applikation in den Hintergrund bringt. Auf diesen Teil hat man nur indirekt Einfluss. Jedoch wird es als „schlechtes Design“ gehalten, wenn man diesen Teil ohne triftige Gründe ausblendet. Deswegen wird er auch von DRIVE-Remote (in der Standard-Ansicht) nicht verändert.
Bei PocketPC Applikationen ist es üblich, einen Menübalken am unteren Ende einzublenden. Da DRIVE-Remote aber keine Menüs verwendet (da es zu klein ist, um es vernünftig mit dem Finger zu bedienen) wird dieser ganze Balken ausgeblendet. Ein Vorteil daraus ist, dass man etwas mehr Platz für die Darstellung von Information gewinnt. Vor allem kann dadurch eine etwas grössere Schrift verwendet werden, um die Lesbarkeit zu erhöhen. Nachteilig ist aber das Fehlen des Symbols (und Buttons) für die Steuerung des Eingabe-Bedienfelds (SIP, Simple Input Panel), was das Ein- und Ausblenden einer Display-Tastatur oder einer anderen Eingabeart erlaubt (z.B. Schrifterkennung). Da DRIVE-Remote faktisch keine Texteingabe (außer zur Konfiguration) benötigt, wird dieses Symbol nicht benötigt. Falls man explizit Zugriff auf die Eingabe benötigt, so wird die Displaytastatur von der Ablaufsteuerung bzw. über einen eigenen Button ein- bzw. ausgeblendet.

Um die Eingabe von Kommentaren zu ermöglichen (siehe Abschnitt 3.2.1) wurde eine

*eigene Lösung* gewählt. Anfänglich gab es neben dem Start/Stopp Button noch einen zweiten Button um eine Kommentaraufnahme zu starten. In Zuge der Entwicklung hat sich das aber als anfällig für Fehlbedienungen erwiesen. Es wurde öfters eine Messung gestoppt, wenn nur ein Kommentar abgeben werden sollte, bzw. umgekehrt, eine Kommentaraufzeichnung gestartet, obwohl man eine Messung beenden bzw. starten wollte. Deswegen wurde der Kommentar Button gänzlich eingespart. Dafür wurde aber die *gesamte* Oberfläche als Kommentar Button genutzt (siehe Text in Abbildung 3.13 ganz unten *Tap any free place to start recording*) bzw. als Fläche verwendet, die nicht schon selbst auf einen Tastendruck reagiert (wie die Buttons, ComboBoxen usw.). Abbildung 3.14 zeigt dies anschaulich.


Abbildung 3.14: Flächen die als Start Button für ein Audiokommentar dienen (Rot hinterlegt)

Um einen Audiokommentar aufzuzeichnen, muss man also auf die passende Fläche drücken. Danach ändert sich der Text in der untersten Zeile („Tap any free place...“) zu „Recording....“ und der Messungs Start/Stopp Button beginnt zu blinken. Damit hat der Benutzer einen kaum zu übersehenden Hinweis, dass eine Aufzeichnung läuft. Um den Kommentar zu beenden, muss nochmals auf die passende Fläche gedrückt werden, bzw. es wird nach 30 Sekunden der Kommentar automatisch beendet. Diese Zeit wurde gewählt, da es einerseits erlaubt genug Informationen aufzuzeichnen, andererseits die Audiodateien noch nicht zu groß werden. Zur Zeit wird mit 11khz, 8bit Mono, ohne Komprimierung aufgezeichnet. Dadurch ergibt sich eine maximale Dateigröße von ca. 300 KByte. Diese Größe ist wichtig, da nach dem Ende der Aufzeichnung die Datei an den AVL-DRIVE PC übertragen wird. Falls aber nur Bluetooth zur Verfügung steht, dauert die Übertragung dieses Kommentars einige Sekunden, in der die Verbindung komplett ausgelastet ist und deswegen keine (bzw. kaum) Information zur Messung angezeigt werden kann. Bei Verwendung von WLAN Verbindungen ist das nicht relevant, da die Datenübertragungsrate so hoch ist, dass der Transfer nur Sekundenbruchteile benötigt.

### 3.9.3 Anzeigen-Module

Zur Modularisierung der einzelnen Anzeigen werden *User Controls* verwendet, ein neues Feature von .NET CF 2.0. Für die Standard-Version des Frameworks gab es dieses Feature bereits ab Version 1.1. Neu hinzugekommen ist auch noch eine bessere bzw. erweiterte Unterstützung in der Entwicklungsumgebung Visual Studio 2005.
User Controls sind von der Framework-Seite gleichzusetzen mit „normalen“ Controls wie z.B. Textfeldern oder Button. Der Unterschied ist, dass diese Controls vom User frei definiert werden können. Eine solche Funktionalität gibt es für Windows schon seit den Anfängen der objektorientierten Userinterface-Entwicklung mit der MFC. Neu ist die viel bessere Integration in die Entwicklungsumgebung und eine bessere Kapselung. Ein eigenes User Control wird im Userinterface Designer von VS 2005 bereits zur Design Zeit richtig dargestellt. Damit ist eine viel einfachere Entwicklung möglich, da nicht mehr mittels Ausprobieren überprüft werden muss, ob die Ansicht passt. Auch lassen sich damit relativ leicht verschiedenste UI-Module entwerfen, um sie später, je nach gewünschter Funktionalität, in das Programm einzubinden (mittels „drag and drop“).

# Kapitel 4

# Implementierung

## 4.1 Implementierungsaspekte

Implementiert wurde DRIVE-Remote, wie bereits erwähnt, in C# unter Visual Studio Professional 2005. Anfänglich wurde noch unter 2003 gearbeitet, nachdem aber die Beta 2 von VS 2005 verfügbar war, wurde auf diese gewechselt um mit .NET 2.0 Compact Framework (CF) zu arbeiten. Grundsätzlich wäre auch eine Entwicklung mit Version 1.0 des CF möglich gewesen, aber 2.0 vereinfachte die Entwicklung.
Ein weiterer Vorteil der Entwicklung mit .NET CF ist die Kompatibilität mit dem normalen .NET Framework. Das Compact Framework ist ein Subset des normalen Frameworks. Es erlaubt, Programme für das Compact Framework zu schreiben und es auf einem normalen PC auszuführen (ausgenommen hiervon sind spezielle Assemblies die es nur auf WindowsCE Plattformen gibt, wie z.B. `Microsoft.WindowsCE.Forms`). Natürlich ist das Verhalten des Programms nicht komplett ident. Das wirkt sich vor allem beim Userinterface Design aus. Ein Userinterface, welches für das .NET CF entwickelt wurde, ist (abhänging von der Komplexität) nicht gut auf einem normalen PC zu bedienen, da die verschiedenen Elemente (Controls wie z.B. Buttons, Textfelder, Scrollbalken usw.) des Userinterface anders aussehen und sich auch teilweise anders verhalten. Aus diesem Grund bietet es sich an, das Programm in zwei Hauptteile aufzuteilen (in .NET auch Assemblies gennant):

**DRIVE-Remote Framework** beinhaltet den Kommunikationsteil (*DRIVEServer* siehe Abbildung 3.12) der plattformunabhängig ist, also auf dem PDA und auf dem PC ohne Änderung (sogar ohne neu zu kompilieren) funktioniert
Dieser Teil ist eine eigenständige DLL.

**DRIVE-Remote** beinhaltet den Rest der Applikation, insbesondere auch das Userinterface.
Von diesem Modul gibt es zwei Versionen, eine für den PDA und eine für den PC. Die PC Version dient nur als interne Testversion. Sie enthält nicht die komplette Funktionalität der PDA-Version. Vor allem gibt es nur rudimentäres Fehlerhandling und rudimentäre Information über die Benutzerführung. Es wird aber sichergestellt, dass das Testen des Verbindungsteils (DRIVE-Remote Framework) auch für den PC vollständig durchgeführt werden kann.

Dieser Teil benutzt das DRIVE-Remote Framework. Aus diesem wird dann die ausführbare Datei erstellt.

### 4.1.1 Implementierungsdetails zu den Übertragungsprotokollen

Im Sequenzdiagramm (siehe Abbildung 4.1) ist vereinfacht der Ablauf der einzelnen Nachrichten zu sehen. Die einzelnen Nachrichten werden in den folgenden Abschnitten beschrieben.

#### Kommando-Kanal (TCP)

**Allgemeine Message-Struktur**
Nachrichten, die über diesen Kanal gesendet werden, sind immer textkodiert. Insbesondere auch Zahlen werden als Text versendet. Eine Nachricht beginnt immer mit drei Raute Zeichen („###“) und endet auch damit. Weiters folgt die Sender-ID, danach ein Doppelpunkt. Darauf folgt das Kommando bzw. der Typ der Nachricht und darauffolgend die Daten. Mehrere Daten werden durch „;“ getrennt.
*Konvention:* Definitionen sind in der „Backus-Naur-Form“ (BNF) verfasst.
*Definition:*
###<SENDER-ID>:<MSGTYP> [<Datum1>{;<Datumx>}]###

SENDER-ID ist im einfachsten Fall „DRIVE“ für die AVL-DRIVE Anwendung und „PDAx“ für die „Fernbedienungen“, wobei „x“ ein Identifier ist, um mehrere Geräte zu unterscheiden. Diese Unterscheidung ist nötig, falls mehrere Geräte gleichzeitig an AVL-DRIVE angeschlossen werden.
„MSGTYP“ ist eines der im nächsten Kapitel definierten Kommandos bzw. Nachrichten-Typen.
„Datum“ steht für einen Wert, welcher eine Zahl, Text oder sonstiges sein kann und abhängig vom Message Typen ist.

**Übersicht über die Typen von Nachrichten**
In der Tabelle 4.1 sind alle Nachrichten zu sehen, die implementiert wurden. Anfänglich waren auch Nachrichten für die Übertragung von Konfigurationsdaten wie z.B. die AVL-DRIVE bekannten Messkanäle vorgesehen. Diese wurden aber wieder fallen gelassen, da dies zu viele Änderungen auf der Seite von AVL-DRIVE nach sich gezogen hätte.

**Details zu den Nachrichten-Typen**

**START:** Gibt AVL-DRIVE den Befehl die Messung zu starten.
**Beispiel:** ###PDA1: START###

**STOP:** Gibt AVL-DRIVE den Befehl die Messung zu stoppen.
**Beispiel:** ###PDA1: STOP###

**BYE:** Teilt AVL-DRIVE mit, dass DRIVE-Remote die Verbindung beendet.
**Beispiel:** ###PDA1: BYE###

| Befehl | Nachricht Typ |
|---|---|
| Start einer Messung | START |
| Stoppen einer Messung | STOP |
| Beenden einer Verbindung | BYE |
| Abonnieren von Events | ABBO_SUB_START |
| Beenden des Eventabos | ABBO_SUB_STOP |
| Abonnieren eines Parameters | ABBO_CHANNEL_START |
| Beenden eines Parameterabos | ABBO_CHANNEL_STOP |
| Senden von Daten an AVL-DRIVE | SEND_VALUE |
| Abfragen eines einzelnen Wertes | GETVALUE |

Tabelle 4.1: Übersicht über die verschiedenen Nachrichtentypen

**ABBO_SUB_START:** AVL-DRIVE soll ab diesem Zeitpunkt sämtliche Events an den Client senden. Das Senden der Events erfolgt über den anderen Verbindungskanal (UDP)
**Beispiel:** ###PDA1: ABBO_SUB_START###

**ABBO_SUB_STOP:** AVL-DRIVE soll ab diesem Zeitpunkt keine Events mehr an den Client senden.
**Beispiel:** ###PDA1: ABBO_SUB_STOP###

**ABBO_CHANNEL_START:** AVL-DRIVE soll ab diesem Zeitpunkt den ausgewählten Parameter an den Client senden. Das Senden der Parameterdaten erfolgt über den anderen Verbindungskanal (UDP).
**Daten:**
ID eines Parameters (im Normalfall eine ganzzahlige Nummer zwischen 1000 und 10000)
**Optionale Daten:**
Sende Intervall
Damit kann angegeben werden, in welchem Zeitintervall der Wert wiederholt gesendet wird. Das dient zur Reduzierung des Datenaufkommens, da einige Werte mit einer Samplingrate von über 1kHz aufgezeichnet werden. Falls nichts angegeben wird, wird ein Standardwert von 0,5 Sekunden angenommen.
Die Angabe von 0 Sekunden bedeutet, dass der Wert immer gesendet wird, wenn er sich ändert.
*ACHTUNG:* die Zeit dient nur als Richtwert.
**Beispiel:** ###PDA1: ABBO_CHANNEL_START 2030;1,8###
Mit dieser Nachricht wird der Parameter mit der ID 2030 abonniert. Es wird erwartet, dass der Wert alle 1,8 Sekunden gesendet wird.
*ACHTUNG:* AVL-DRIVE sendet einen Parameter immer nur einmal. Wenn ein Parameter nochmals abonniert wird, wird er trotzdem nur einmal geschickt. Für das Sendeintervall wird immer der zuletzt gesendete Wert angenommen. Daraus folgt, dass der Client sicherstellen muss, das die Parameter nicht mehrfach abonniert werden.

**ABBO_CHANNEL_STOP:** AVL-DRIVE soll zu dem angegebenen Parameter keine

neuen Werte mehr schicken.
**Daten:**
ID eines Parameters (im Normalfall eine ganzzahlige Nummer zwischen 1000 und 10000)
**Beispiel:** ###PDA1: STORNOPARAM 2030###

**VALUE:** Übermitteln eines Wertes von DRIVE-Remote an DRIVE.
**Daten:**
Parameternummer,
Wert des Parameters. Der Typ (Ganzzahl, Text, Fließkomma,... ) hängt implizit vom Paramter ab.
**Beispiel:** ###DRIVE: VALUE 170;3250###

**GETVALUE:** AVL-DRIVE soll den neuesten bzw. aktuellsten Wert zu dem angegeben Parameter senden. Das dient zum Abfragen von Einzelwerten vom Server. Es kann benutzt werden, wenn z.B. nur einmalig ein Messwert abgefragt werden soll (Öltemperatur oder ähnliches), oder aber um Konfigurationsdaten vom Server zu erhalten.
**Daten:** ID des gewünschten Wertes. Diese ID kann sowohl einen Parameter klassifizieren, als auch spezielle Werte, wie beispielsweise Konfigurationsdaten einer Messung (Zylinderanzahl des Motors, Anzahl der Gänge usw.) benennen.
**Beispiel:** ###PDA1: GETVALUE 230###

### Daten-Kanal (UDP)

#### Allgemeine Nachrichten-Struktur

Die Daten auf dieser Verbindung werden im Binärformat übertragen. Alle Pakete haben einen gemeinsamen Header bestehend aus einem Identifikator für den Typ der Nachricht und einem Zeitstempel (im AVL-DRIVE eigenen Format, siehe Tabelle 4.2). Je nach Typ der Nachricht können danach noch zusätzliche Informationen folgen.

| **Information** | **Beschreibung** | **Länge in Bytes** |
|---|---|---|
| Identifikationsbyte/ Typ der Nachricht | Folgende Typen sind bis jetzt definiert:<br>SUBEVENT = 1<br>PARAM = 2<br>MEASUREMENT_END = 3<br>MEASUREMENT_START = 4<br>PROGRAM_END = 5 | 1 |
| Zeitstempel (Timestamp) | AVL-DRIVE Format:<br>Anzahl hundertstel Sek.<br>seit Messungsbeginn | 4 |
| **Gesamtlänge:** | | 5 |

Tabelle 4.2: UDP-Header Format

**SUBEVENT:** Dieses enthält Daten, die zu einem Event gehören, und wird gesendet, nachdem der Event erkannt wurde.

**Daten:** Die Länge der Zusatzdaten ist vom jeweiligen Event abhängig. Es enthält den Main Event Namen, den Sub Event Namen, die Drivability Note und zusätzlich noch alle Detailevents mit ihren zugehörigen Drivability Noten.

**PARAM:** Diese Nachricht wird gesendet, wenn es neue Parameterwerte gibt, wie z.B. Drehzahl, Geschwindigkeit usw. Um den Overhead zu verringern, kann solch ein Paket mehrere Parameter gleichzeitig enthalten. Ein Parameter besteht aus einer 4 Byte langen Parameter ID.

**Daten:** Mindestens 1 bis zu maximal 255 Parameterwerte (bestehend aus 4 Byte Parameter ID und 8 Byte Parameter Wert)

**MEASUREMENT_END:** Diese Nachricht wird von AVL-DRIVE gesendet, wenn die Messung beendet wurde. Diese Nachricht kommt im Normalfall nur, nachdem DRIVE-Remote eine Stop-Nachricht gesendet hat (STOP). Sie kann aber auch empfangen werden, wenn AVL-DRIVE die Messung selbst beendet (entweder der Benutzer bricht die Messung auf der PC Seite ab oder ein Fehler ist aufgetreten).

**Daten:** Dieses Paket enthält zusätzlich noch die Gesamt-Drivability Note der Messung und den Namen der Messung.

**MEASUREMENT_START:** Diese Nachricht wird versendet, *nachdem* eine Messung gestartet wurde. Dies kann erfolgen, wenn die Messung direkt am PC gestartet wurde, oder nachdem die Nachricht START gesendet wurde.
**Achtung:** Die Antwort, die direkt über den TCP Kanal auf die Nachricht START kommt, ist nicht gleichwertig. Diese kommt bereits, wenn AVL-DRIVE anfängt die Messung zu starten, was einige Sekunden bis zu Minuten dauern kann, je nachdem, welche Schnittstellen zur Hardware initialisiert werden müssen. Diese Nachricht bedeutet, dass die Messung schon läuft!

**Daten:** Zusätzlich zum Header wird noch ein 4 Byte Timestamp im Unix-Zeitformat mitgeschickt. Dies wird als einfache Synchronisation der Messzeiten zwischen AVL-DRIVE und DRIVE-Remote verwendet.

**PROGRAM_END:** Mit dieser Nachricht wird signalisiert, dass AVL-DRIVE sich beendet hat. Wird zum Beenden der TCP Verbindungen und Benachrichtigen des Users verwendet.

**Daten:** Diese Paket enthält keine weiteren Daten.

### 4.1.2 Externe Tools und Libraries

Um den Entwicklungsaufwand zu verringern, wurden verschiedene Tools bzw. Libraries verwendet. Im Speziellen soll das *Smart Device Framework* von *OpenNETCF*[63] erwähnt

werden. Dieses Framework erleichert das Arbeiten mit dem .NET CF durch die Bereitstellung von zusätzlicher Funktionalität gegenüber dem .NET CF Framework. Es werden auch Klassen und Methoden zur Verfügung gestellt, die einige Low-Level Zugriffe auf die Windows CE API kapseln. Davon genutzt wurde vor allem die Unterstützung für Audio (Abspielen und Aufnahme). Verwendet wurde die Version 1.4 der Bibiliothek, da diese auch als Sourcecode zur Verfügung steht. Die Version 2.0, die Ende Mai 2006 fertiggestellt wurde, gibt es nur mehr gratis als Binaries. Für eine komplette Integration in das VS 2005 bzw. auch den Sourcecode muss eine Licence gekauft werden.
Die Verfügbarkeit des Sourcecodes ist wichtig, da nur einzelne Funktionalitäten der Library benötigt wurden, und diese Klassen einzeln herausgenommen wurden, was zu einer viel kleineren Binary führte (die OpenNETCF Libary hat insgesamt ca. 2MB).

## 4.2 Fazit zur Implementierung

Da die Software bereits im Einsatz ist, konnten schon einige Erfahrungsberichte über die Verwendung des neuen Displays gesammelt werden.

Die Anwender sind grundsätzlich sehr zufrieden mit der Software. Vor allem die Möglichkeit zusätzliche bzw. andere Messkanäle als beim alten Display zur Verfügung zu haben, wird sehr geschätzt und genützt. Auch die Verwendung von Standardkomponenten hat den Benutzern und Kunden sehr gefallen, da sie dadurch ihre eigenen PDAs nützen können.
Auch die Fähigkeit Audio/Voice-Kommentare zu einer Messung vorzunehmen wird gerne verwendet. Dies führte sogar zu dem Wunsch, dass bei AVL-DRIVE selbst die Möglichkeit eines Kommentares (dort nur als Text) nachträglich eingebaut wird.
Das anfänglich befürchtete Problem einer zu kurzen Laufzeit der PDAs stellte sich als unbegründet heraus. Neuere Standardmodelle haben Laufzeiten von über einer Stunde (inkl. eingeschaltetem Bluetooth bzw. WLAN). Mit zusätzlichen Batterien bzw. Akkus erreicht man sogar Zeiten von weit über drei Stunden.

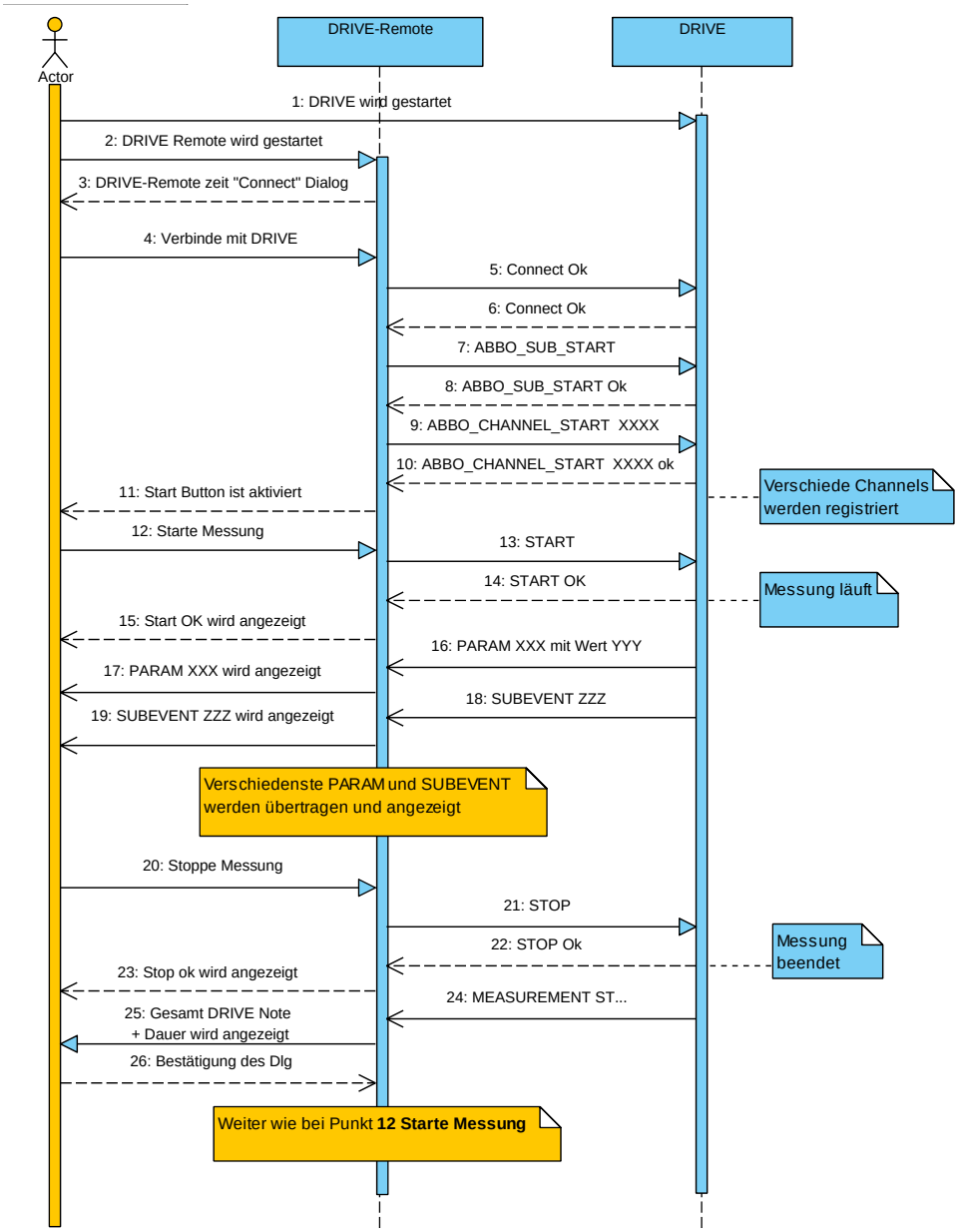



Abbildung 4.1: DRIVE-Remote Sequence Diagramm

# Kapitel 5

# Evaluierung

## 5.1 Robustheit der drahtlosen Netzwerke

Anfänglich wurde befürchtet, dass es zu oftmaliger Störung der drahtlosen Übertragung kommen würde. Dies stellte sich als vollkommen unbegründet heraus. Wenn die Verbindung einmal aufgebaut war (was in einigen Fällen recht kompliziert ist), gab es keine Unterbrechung der Übertragung (abgesehen von den Problem mit den HP iPAQs, siehe nächsten Abbschnitt 5.2). Die Anbindung über WLAN oder Bluetooth wurde auch am Rollenprüfstand bzw. auf einem Motorenprüfstand getestet, wo es sehr viel an Störstrahlung gibt. Auch hier gab es keinerlei merkbaren Aussetzer.

Der einzige wirkliche Nachteil der drahtlosen Übertragungstechnologien war der teilweise recht komplizierte Verbindungsaufbau. Beispielsweise war es in einem Fall nicht möglich, eine WLAN Verbindung zwischen einem Dell PDA und einem Dell Notebook herzustellen. Auch die Dell Hotline konnte nicht helfen und riet zu einer Neuinstallation des Notebooks, was schlussendlich das Problem löste.
Vor allem aber Bluetooth am PC ist sehr schlecht standardisiert (Benutzerführung der Software, unterstütze Profile usw.).

## 5.2 Probleme bei drahtloser Kommunikation

Der erste Kunde, der die PDA Lösung gekauft hat, verwendete als Plattform iPAQs von HP (Modell HP iPAQ hx2750). Diese Geräte erfüllen alle Anforderungen für DRIVE-Remote. Für die Kommunikation verwendeten sie WiFi (WLAN) anstatt Bluetooth. Begründet war dies darin, dass die Firmen-Notebooks des Kunden alle zwar WLAN Funktionalität eingebaut haben, aber kein Bluetooth, und sie nicht eine externe Bluetooth Lösung verwenden wollten (USB-Bluetooth Sticks stehen vom Gehäuse weg und sind deswegen störend).
Der Kunde hatte bei der Anwendung über WiFi das Problem, dass manchmal die Datenübertragung „einfror“ und einige Sekunden nichts übertragen wurde. Aufgefallen ist das vor allem dadurch, dass dann die Anzeigen am PDA nicht mit denen des Tachometers übereinstimmten. Dieses Verhalten ist nur im Stadtgebiet aufgetreten, nicht aber auf Landstraßen und Autobahnen. Um diesen Fehler zu reproduzieren, erhielten wir von den Kunden einen iPAQ, um selbst Tests damit durchzuführen. Nach mehreren Versuchen

wurde dieses Verhalten darauf zurückgeführt, dass sich andere unbekannte WiFi Netze in Reichweite befanden. Die iPAQ WLAN Software versucht nun immer mit den anderen Netzen eine Verbindung aufzubauen, und während eines solchen Versuch wird die bestehende Verbindung eingefroren. HP bestätigte nach Rückfrage dieses Verhalten, war aber nicht in der Lage, eine Lösung dafür anzubieten.
Bei Gesprächen für eine Lösungsfindung mit den Kunden stellte sich heraus, dass er auch mit einer kabelgebunden Lösung zufrieden ist, wenn damit auch eine Stromversorgung gewährleistet wird. (Dieser Kunde macht öfters extrem lange Testfahrten, länger als 4h, wobei der PDA Akku nicht lange genug hält). Aus diesem Grund wurde versucht, die Verbindung über die USB-Schnittstelle (ActiveSync) laufen zu lassen, da diese auch IP Verbindung zulässt. Bei Tests stellte sich heraus, dass zwar eine TCP Verbindung möglich ist (Steuerung der Messung), UDP-Pakete aber nicht übertragen werden (keine Messdaten wurden empfangen). Das konnte durch Recherche im Internet bestätigt werden (Suche in diversen Newsgroups und Online-Foren).
Um eine kabelgebundene Lösung dennoch zu ermöglichen, wurde die Messdatenverbindung geändert, um auch TCP zu benützen. Durch das modulare Design war das mit einem relativ geringen Aufwand möglich. (einige Stunden für den PDA und nochmals soviel für dir AVL-DRIVE Applikation). Diese Änderung wurde generell eingeführt, also auch bei Verwendung von WLAN oder BT, um nicht mehrfachen Code warten zu müssen. Mit dieser Änderung ist es jetzt möglich auch ein sogenanntes Sync-Kabel zur Datenübertragung zu benutzen. Nachteil dieser Übertragung ist natürlich das Kabel, das eine maximale Länge von nur 5 Meter haben darf([77]).
Andererseits ergeben sich durch die Kabel einige Vorteile. Darunter fällt zuerst die Stromversorgung über das Sync-Kabel, wodurch beliebig lange Messfahrten durchgeführt werden können. Aber auch der Verbindungsaufbau ist um vieles einfacher, da jetzt nur eine fixe Adresse eingetragen werden muss (virtueller Name *PPP_PEER*).

## 5.3 Drahtlose Anbindung von zusätzlichen Sensoren

Wie im Theorieteil bereits bestätigt, ist es grundsätzlich möglich drahtlos zusätzliche Sensoren an das AVL-DRIVE Messsystem anzuschließen. Mit der aktuell verwendeten DMU (siehe Abschnitt Hardware 3.1.2 bzw. Abbildung 3.1) ist das nicht möglich. Aber bereits das Nachfolgegerät, welches zur Zeit getestet wird, unterstützt die Kommunikation über WLAN. Es würde also die Verbindung zwischen Notebook und DMU drahtlos sein. Dies alleine reicht aber noch nicht aus, um die Verkabelung eines Fahrzeugs zu vereinfachen. Das jetzige Konzept einer DMU eignet sich auch nicht gut für die Verwendung von drahtlosen Technologien, denn die DMU fungiert auch als Analog-Digital Umsetzer. D.h. die Sensoren werden direkt an die DMU angeschlossen. Um jetzt einen Sensor drahtlos anzubinden, würde man direkt am Sensor einen Analog-Digital Signalwandler (A/D-Wandler) benötigen, diesen Datenstrom dann drahtlos übertragen und dieses Signal wieder analog zurückwandeln, um damit die DMU zu speisen. Vor allem bei der Verwendung von mehreren Sensoren lässt sich leicht erkennen, dass dies nicht wirklich sinnvoll ist.
Die Firma AVL-LIST ist aber gerade dabei, eine Alternative für ihre doch relativ alte DMU (faktisch gleichbleibende Hardware seit 1998) zu schaffen. Dieses neue System (welches auch in Zuge einer Diplomarbeit entwickelt wurde[31]), befindet sich zur Zeit kurz

vor der Serienproduktion. Dieses neue Messprodukt zeichnet sich dadurch aus, dass das Notebook über eine CAN-Schnittstelle mit dem Messsystem verbunden ist. Das Messsystem ist modular aufgebaut und kann durch zusätzliche Module erweitert werden (siehe Abbildung 5.1).

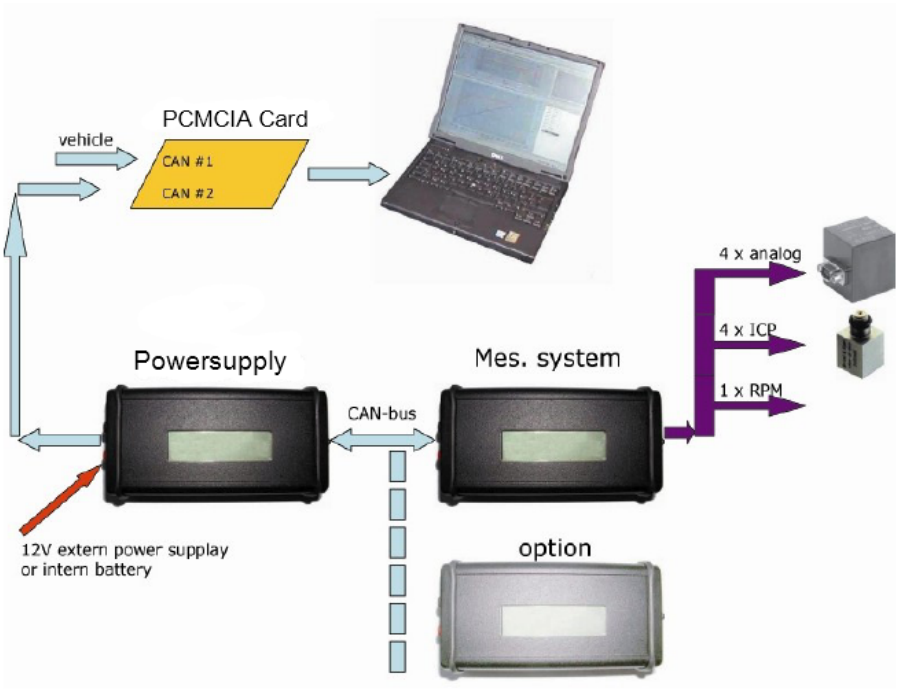



Abbildung 5.1: Schematische Darstellung des neuen Messsystems[31, Seite 49]

Die einzelnen Messmodule sind Eigenentwicklungen, die im Zuge der Arbeit erstellt wurden. Dadurch ergibt sich die Möglichkeit, einzelne Module zu modifizieren und drahtlos anzubinden. Zur Zeit besteht aber keine Notwendigkeit für diese Änderungen, da die jetzige Ausführung noch reicht.

# Kapitel 6

# Schlussbemerkungen

## 6.1 Zusammenfassung

In dieser Arbeit konnte gezeigt werden, dass bereits drahtlose Kommunikationstechnologien verfügbar sind, die den Anforderungen im automotiven Bereich genügen. Besonders in Verwendung mit AVL-DRIVE sind Technologien wie Bluetooth, ZigBee und auch WLAN nach 802.11 sehr gut geeignet, um Sensoren bzw. Monitoring Geräte zu verbinden. Da dies standardisierte Verfahren sind, ist keine Eigen- bzw. Neuentwicklung nötig und es kann auf allgemein verfügbare Komponenten zurückgegriffen werden.
Es zeigte sich, dass keines der am Markt verfügbaren Geräte den Anforderungen entsprach und als Ersatz für die bisher verwendeten Displays in Frage kam. Daher war für diesen Teil des Systems eine Neuentwicklung nötig. Auch hier konnte auf Standardkomponenten, wie z.B. einen PDA auf Windows CE 5.0 Basis (PocketPC) zurückgegriffen werden. Dadurch war es möglich die Entwicklungszeit kurz und die Kosten gering zu halten. Die Anforderungen der Firma AVL, Abteilung DV, für das neue Monitoring-Gerät konnten erfolgreich erfüllt werden.
Weiters wurde auch gezeigt, dass die drahtlose Anbindung von Sensoren grundsätzlich möglich ist. Es wurde aber auf eine Entwicklung verzichtet, da dies zurzeit nicht benötigt wird und noch keine Standardkomponenten für diese Anwendung am Markt existiert. Deshalb wird dieser Punkt verschoben und zumindest auf den neuen Standard IEEE 1451.5 für drahtlose Sensoren gewartet.

## 6.2 Ausblick

### 6.2.1 DRIVE-Remote Version 2.0?

Bei der jetzigen Version haben sich einige Schwachpunkte gezeigt, die durch einfache Änderungen kaum gelöst werden können. Beispielsweise hat es sich herausgestellt, dass für eine Remote-Kontrolle der Kalibrierung der Sensoren das jetzige Userinterface schlecht geeignet ist (vor allem die Benutzerführung und das Reagieren auf Events). Andererseits wird jetzt gerade AVL-DRIVE 3.0 entwickelt (bis jetzt Version 2.x), wodurch es zu vielen Änderungen kommt (z.B. werden die verschiedenen Schnittstellen aufgetrennt, die Anbindung der Messkanäle geändert usw.). Ein weiterer Wunsch ist es mehrere DRIVE-Remote Geräte mit einer AVL-DRIVE Applikation zu verbinden, wobei aber nur ein Gerät fern-

steuern kann, die anderen nur zur Anzeige benutzt werden.
Aus diesem Grund wird DRIVE-Remote 2.0 für das Konzept nochmals überarbeitet, und diesmal auch der Teil von AVL-DRIVE neuentwickelt.

# Literaturverzeichnis


[1] JTC 1/SC 6. *Telecommunications and information exchange between systems - Near Field Communication - Interface and Protocol (NFCIP-1).* ECMA International, Rue du Rhône 114, CH-1204 Geneva, Korean Standards Association #701-7 Yeoksam-dong Gangnam-gu KR-Seoul 135-513, 1 edition, 4 2004. Runtergeladen von http://standards.iso.org/ittf/PubliclyAvailableStandards/c038578_ISO_IEC_18092_2004(E).zip.

[2] Esmertec AG. Esmertec's wireless Java ME. Website. http://www.esmertec.com/, besucht am 11.06.2006.

[3] E.T. Powner A.Kutlu, H. Ekiz. Performance analysis of MAC protocols for wireless control area network. In *Second International Symposium on Parallel Architectures, Algorithms, and Networks*, pages 494–499. IEEE, 6 1996.

[4] Richtlinie der Kommission 2004/104/eg. Amtsblatt der Europäischen Union, 10 2004.

[5] Chipcon AS. World's First True System-on-Chip (SoC) ZigBee$^{TM}$Solution. Website, 9 2005. http://www.chipcon.com/index.cfm?dok_id=243\&kat_id=6, besucht am12.06.2006.

[6] The Infrared Data Association. About IrDA. Website, 05 2006. http://www.irda.org/displaycommon.cfm?an=1, besucht am 10.05.2006.

[7] Detsche Karl-Heinz Bauer Horst, Crepin Jürgen. *Autoelektrik, Autoelektronik.* Robert Bosch GmbH, 4 2002.

[8] Stan Berka. Java for PocketPC PDA's, 2006. http://www.berka.name/stan/jvm-ppc/java_for_pda.html, besucht am 31.03.2006 um 23:00.

[9] Pascal Audoux übersetzt von Philippe Fremy. Impressions on MFC vs Qt Programming. Website, 7 2005. http://phil.freehackers.org/kde/qt-vs-mfc.html, besucht am 15.06.2006.

[10] Barry W. Boehm. A Spiral Model of Software Development and Enhancement. *ACM SIGSOFT Software Engineering Notes*, 11:14–24, 1986.

[11] Jason Clark. Let users add functionality to your .NET applications with Macros and Plug-Ins. Website, 2 2005. http://msdn.microsoft.com/msdnmag/issues/03/10/Plug-Ins/ , besucht am 05.06.2005.

[12] Information about CompactFlash, 2004. http://www.compactflash.org/info/cfinfo.htm, besucht am 22.03.2006 um 14:00.

[13] GoDB Development Tool for Mobile, Pocket PC, Palm, Nokia, 2005. http://www.go-db.com/godb_home.asp, besucht am 31.03.2006 um 23:00.

[14] Conversational Computing Corporation. Mobile Conversay Software Development Kit. Website, 2006. http://www.conversay.com/Products/Embedded/MCSDK.asp besucht am 14.06.2006.

[15] Fonix Corporation. About Fonix VoiceIn® Standard Edition. Website, 2006. http://www.fonixspeech.com/pages/voiceIn_standard.php besucht am 14.06.2006.

[16] Fujitsu Computer Systems Corporation. Stylistic® ST5031 and ST5031D Tablet PC. Website, 1 2006. http://www.computers.us.fujitsu.com/images/pentablets/st5021_photogal/right2D.jpg, besucht am 02.02.2006 um 17:00.

[17] Intel Corporation. Intel PRO/Wireless 2011B LAN PC Card Specifications. Website. http://support.intel.com/support/wireless/wlan/pro2011b/lanpccard/sb/cs-008114.htm, besucht am 27.08.2005.

[18] Microsoft Corporation. Download eMbedded Visual C++ 4.0. Website, 12 2003. http://www.microsoft.com/downloads/details.aspx?familyid=1dacdb3d-50d1-41b2-a107-fa75ae960856.

[19] Microsoft Corporation. eMbedded Visual Tools 3.0 - 2002 Edition. Website, 1 2003. http://www.microsoft.com/downloads/details.aspx?FamilyID=f663bf48-31ee-4cbe-aac5-0affd5fb27dd, besucht am 10.06.2006.

[20] Microsoft Corporation. Microsoft Windows Powered Smart Displays. Website, 2005. http://www.microsoft.com/windows/smartdisplay/default.mspx, besucht am 05.10.2005 um 17:30.

[21] Microsoft Corporation. What is a Tablet PC? Website, 11 2005. http://www.microsoft.com/windowsxp/tabletpc/evaluation/about.mspx, besucht am 31.10.2005 um 11:45.

[22] Microsoft Corporation. Acer TravelMate® C110. Website, 1 2006. http://www.microsoft.com/windowsxp/images/tabletpc/evaluation/products/LargerImages/57043_400X286_AcerC110_F.jpg, besucht am 02.02.2006 um 17:00.

[23] ViewSonic Corporation. Smart Displays. Website, 10 2006. http://www.viewsoniceurope.com/DE/Products/Mobile_and_Wireless/v210.htm, besucht am 15.01.2006 um 15:45.

[24] Inc Crossbow Technology. Motes, Smart Dust Sensors, Wireless Sensor Networks. Website, 2 2006. http://www.xbow.com/Products/productsdetails.aspx?sid=3, besucht am 03.02.2006 um 17:00.

[25] Crossbow Technology, Inc, 4145 North First Street, San Jose, California 95134-2109. *MICA2 Datasheet*, 2 2006. http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/MICA2_Datasheet.pdf, besucht am 22.02.2006 um 14:00.

[26] Alan Davy. Components of a smart device and smart device interactions, 5 2003. http://www.m-zones.org/deliverables/d234_1/abstracts/davy-components-of-a-smart-device.php4.

[27] John R. Delaney. Xybernaut Atigo T/HB. Website, 1 2005. von http://www.eweek.com/cobrand/0,3223,a=143543\&s=25407\&ap=1,00.asp, besucht am 11.06.2006.

[28] Ami Desai. Difference between VB and VB.NET, 2005. http://www.dotnetspider.com/kb/Article1721.aspx, besucht am 31.03.2006 um 23:00.

[29] Jean-Pierre Ebert. *Energy-efficient Communication in Ad Hoc Wireless Local Area Networks*. PhD thesis, Technischen Universität Berlin, April 2004.

[30] ECMA International, Rue du Rhône 114, CH-1204 Geneva. *Standard ECMA-340, Near Field Communication Interface and Protocol (NFCIP-1)*, 12 2004. Runtergeladen von http://www.ecma-international.org/publications/standards/Ecma-340.htm.

[31] Patrick Falk. Re-design automotives Messsystem AVL-DRIVE. Master's thesis, Technische Universität Graz,Institut für Elektronik, 11 2005.

[32] Christian Stöcker Frank Patalong. Kleines, dickes Origami. Website, 03 2006. http://www.spiegel.de/netzwelt/technologie/0,1518,405100,00.html, besucht am 29.06.2006.

[33] Bundesgesetz über das Kraftfahrwesen 1967, BGBl.Nr. 267/1967 zuletzt geändert durch BGBl. I Nr. 117/2005.

[34] ETAS GmbH. Über ETAS Group. Website, 2 2006. http://de.etasgroup.com/about/about_etas.shtml, besucht am 2.06.2006.

[35] Exor GmbH. Visualisierung und Bediensysteme. Website, 2 2006. http://www.exor.de/, besucht am 03.02.2006 um 14:00.

[36] S. Hasnaoui. Wireless industrial networking using CAN MAC-sublayer. In *Conference Record of the 2001 IEEE Industry Applications Conference. Thirty-Sixth IAS Annual Meeting.*, pages 1303–1310. IEEE, 2001.

[37] Bernhard Huber. Wireless Real-Time Communication for Smart Transducer Networks. Master's thesis, Technischen Universität Wien ,Institut für Technische Informatik 182/1, Treitlstr. 3/3/182-1, 1040 Vienna, Austria, 08 2004.

[38] International Business Machines Corporation (IBM). Workplace Client Technology, Micro Edition. Website. http://www-306.ibm.com/software/wireless/wctme/bundle.html besucht am 10.06.2006.

[39] IEEE P1451.5 Working Group. Website, 2005. http://grouper.ieee.org/groups/1451/5/, besucht am 26.06.2006.

[40] Automation Artisans Inc. CANRF, a Wireless Controller Area Network (CAN) driver. Website, 2005. http://www.autoartisans.com/products/index.htm, besucht am 27.06.2006.

[41] Dell Inc. Axim x51v 624Mhz-Produktinformation. Website, 1 2006. http://www1.euro.dell.com/content/products/productdetails.aspx/axim_x51v, besucht am 31.01.2006 um 14:00.

[42] Xilinx Inc. Webpad / Web Tablet. Website, 2005. http://www.xilinx.com/esp/dvt/cdv/end_apps/webpad.htm, besucht am 06.12.2005.

[43] Preisvergleich Internet Services AG. Geizhals-Preisvergleich Österreich. Website, 1 2006. Palm Z22 http://geizhals.at/a169663.html, Acer n30 http://geizhals.at/a102274.html, beide besucht am 22.01.2006 um 14:45.

[44] Keith Jacquemin. Creating an Extensible User Interface with .NET, part 1. Website, 2 2005. http://www.codeproject.com/csharp/extensibleui.asp, besucht am 12.06.2005.

[45] May Ji. *Differences Between Microsoft Visual Basic .NET and Microsoft Visual C# .NET*. Microsoft Corporation, One Microsoft Way, Redmond, WA 98052-6399,United States of America, 2 2002. gefunden auf: http://support.microsoft.com/?kbid=308470, besucht am 03.04.2006 um 23:00.

[46] JW Hedgehog Inc. Jim Wilson. What's new in the .NET Compact Framework 2.0, 11 2005. http://msdn.microsoft.com/netframework/programming/netcf/default.aspx?pull=/library/en-us/dnnetcomp/html/whats_new_netcf2.asp, besucht am 02.04.2006 um 20:00.

[47] Christian Just. Near Field Communication: Helfer-Funk für Mobilgeräte. *heise Online*, 3 2004. http://www.heise.de/newsticker/meldung/46084, besucht am 10.04.2006 um 12:00.

[48] Otto Koudelka. Nachrichtensatelliten Skriptum, 1999.

[49] Holger Kunkat. *PN511 Transmission Module*. Philips Semiconductor, 2 2004. http://www.semiconductors.philips.com/products/identification/nfc/index.html, besucht am 14.04.2006 um 14:45.

[50] Khanh Tuan Le. ZigBee SoCs provide cost-effective solutions. *EETimes*, 2005. http://www.eetimes.com/showArticle.jhtml?articleID=173600329, besucht am 28.06.2006 um 14:00.

[51] Symbian Limited. Symbian OS home page. Website, 2006. http://www.symbian.com/symbianos/index.html.

[52] LinuxDevices. The Linux Mobile Phones Showcase. Website, 2 2006. http://linuxdevices.com/articles/AT9423084269.html, besucht am 09.06.2006.

[53] Matric. The Controller Area Network (CAN) has gone wireless! Website, 2003. http://www.matric.com/resources/Wireless_CAN_Bridge_Manual_v9.pdf, besucht am 22.06.2006.

[54] Anne Morris. *Atigo® Product Platform*. Xybernaut Corporation, 5175 Parkstone Drive, Suite 130 Chantilly, Virginia 20151-3832, 3 2005. gefunden auf http://www.xybernaut.com/assets/files/site_content/PDFs/Brochures/Atigo_platform8.pdf am 05.05.2006.

[55] Robert Morrow. *Bluetooth, Operation and use*. McGraw-Hill TELECOM Professional, 2002.

[56] Smart NFC. Was ist NFC? Website, 2006. http://www.smartnfc.com/index.php?option=com_content\&task=view\&id=58%&Itemid=29\&lang=de, besucht am 12.06.2006.

[57] NFC-Forum. About the NFC Forum. Website, 2005. http://www.nfc-forum.org/home besucht am 10.07.2006.

[58] Ronald Nitschke. IEEE 802.11. Website, 8 2003. http://www.ronaldnitschke.de/index.php?main=802.11b/002\&rechts=802.11b/802.11b_r, besucht am 29.08.2005 um 11:00.

[59] Inc. Nuance Communications. Nuance vocon. Website, 2006. http://nuance.com/vocon/, besucht am 14.06.2006.

[60] Conrad Nutschan. Spiralmodell. Website, 8 2006. http://de.wikipedia.org/wiki/Bild:Spiralmodell_nach_Boehm_%281988%29.png, besucht am 06.08.2006 um 11:45.

[61] Institute of Electrical and Inc. (IEEE) Electronics Engineers. IEEE 754-1985, Standard for Binary Floating-Point Arithmetic, 10 1985.

[62] Institute of Electrical and Inc. (IEEE) Electronics Engineers. 802.11 Timelines. Website, 2006. http://grouper.ieee.org/groups/802/11/802.11_Timelines.htm, besucht am 02.06.2006.

[63] OpenNETCF.org. OpenNETCF Smart Device Framework v1.4. Website. www.opennetcf.org, besucht am 10.06.2006.

[64] Roy Osherove. Creating a plug-in framework. Website, 12 2003. http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnaspp/html/pluginframework.asp, besucht am 05.06.2005.

[65] Inc. PalmSource. Palm OS. Website, 2006. http://www.palmsource.com/palmos/, besucht am 09.06.2006.

[66] Research In Motion. Website, 2006. http://www.rim.com/, besucht am 11.06.2006.

[67] Inc. ScanSoft. Scansoft - Dragon Naturally Speaking - Systemanforderungen. Website, 10 2005. http://www.scansoft.de/naturallyspeaking/standard/sysreqs.asp, besucht am 3.11.2005 um 23:45.

[68] Thierry Schembri and Olivier Boisseau. Toshiba T100-X Dynapad. Website, 11 2005. http://www.old-computers.com/museum/computer.asp?st=1\&c=801, besucht am 31.10.2005 um 11:45.

[69] About SDIO Card, 2004. http://www.sdcard.org/sdio/index.html, besucht am 23.03.2006 um 14:00.

[70] Sven-Olaf Suhl. PDAs ohne Handyfunktion kommen außer Mode. *heise Online*, 8 2005. http://www.heise.de/newsticker/meldung/62401, besucht am 21.01.2006 um 15:00.

[71] Mark R. Thomas. Common Object Request Broker Architecture. Website, 10 1999. http://krypton.mnsu.edu/~spiral/eta/glossary/indxGlossOOxml.html, besucht am 05.04.2006 um 17:45.

[72] Trolltech. Qt - Code less. create more. Website, 2006. http://www.trolltech.com/products/qt/index, besucht am 05.06.2006.

[73] Tutorial-Reports.com. Zigbee Introduction. Website, 2005. http://www.tutorial-reports.com/wireless/zigbee/zigbee-introduction.php, besucht am 19.07.2005.

[74] Bernd Heißing und Hans Jürgen Brandl. *Subjektive Beurteilung des Fahrverhaltens*. Vogel Fachbuch, 1 edition, 2002.

[75] Uwe Hansmann und Lothar Merk und Martin S. Nicklous und Thomas Stober. *Pervasive Computing*. Springer Professional Computing. Springer, 2 edition, 2001.

[76] Marc Kirsch und Oliver Welter. Wlan und Bluetooth, Unterschiede und Gemeinsamkeiten. Lehrveranstaltungsunterlagen an der TU München,Lehrstuhl f.Datenverarbeitung
Gefunden auf http://www.ldv.ei.tum.de/media/files/lehre/hauptseminar/ws0203/d2_BluetoothWlan.pdf, besucht am 28.08.2005, 1 2003.

[77] Inc. USB Implementers Forum. USB FAQ. Website. http://www.usb.org/developers/usbfaq/, besucht am 24.04.2006.

[78] Dusan Zivadinovic. Erstklassige Weiten. Elf Bluetooth-USB-Adapter mit großer Reichweite. *c't*, 25:158, 2004.