

Evaluierung und Implementierung einer grafischen Benutzeroberfläche für mobile Endgeräte in Fahrzeugen

Diplomarbeit zur Erlangung des akademischen Grades

Diplom-Informatikerin (FH)



Christiane Rettinger

Berlin, den 16. November 2004

Verfasserin: Christiane Rettinger
Matrikelnummer: s403525
E-Mail: xrettinger@web.de,
christiane.reittinger@first.fraunhofer.de

Studentin der: **Fachhochschule für Technik und Wirtschaft**
Fachbereich: Wirtschaftswissenschaften 2
Studiengang: Angewandte Informatik
Spezialisierung: Facility Management

1. Betreuer: Prof. Dr.-Ing. Thomas Jung
Fachhochschule für Technik und Wirtschaft
Treskowallee 8
10318 Berlin

Betreuendes
Unternehmen: **Fraunhofer Gesellschaft**
Institut für Rechnerarchitektur und Softwaretechnik

2. Betreuer: Dipl.-Inf. Ivo Haulsen
Fraunhofer Institut FIRST
Kekulestraße 7
12489 Berlin

Inhaltsverzeichnis

0.1	Abkürzungsverzeichnis	7
1	Einleitung	9
1.1	Ziel dieser Arbeit	9
1.2	Aufgabenstellung	10
2	SIRENA	11
2.1	Ziele des SIRENA-Projekts	11
2.2	Die Projektpartner	12
2.2.1	Motivation	13
2.3	Lösungsansatz des SIRENA-Projekts	13
2.3.1	Projektstatus	15
2.4	SIRENA-Technologien	16
2.4.1	UPnP – Universal Plug and Play	17
2.4.2	OSGI	20
2.4.3	Bluetooth	21
2.4.4	WLAN	22
2.4.5	CAN	22
2.5	Die Demonstratoren	23
2.5.1	Home-Demonstrator	23
2.5.2	Cross-Domain-Demonstrator	24
2.5.3	Automotive-Demonstrator	25
2.5.3.1	Direkt beteiligte Projektmitglieder	27
2.5.3.2	Automotive Web System - AWebS	27
2.5.3.3	Digital Audio Broadcast (DAB)	29

2.6	Fazit der Projektanforderungen	30
3	Usability-Design	32
3.1	Ziele des Usability-Design	32
3.2	Allgemeine Aspekte des Interface-Design	33
3.2.1	Software-Ergonomie	33
3.2.2	Usability	34
3.2.3	User-Centered-Design	35
3.3	Menschliche Wahrnehmung und ihre Grenzen	36
3.3.1	Methoden des Erkennens	39
3.3.2	Visuelle Wahrnehmung	40
3.4	Projektspezifische Aspekte des Interface-Design	42
3.4.1	Sicherheit	42
3.4.2	Eingabe	43
3.4.3	Modi – weniger ist mehr	45
3.5	Design	46
3.5.1	Konzept	47
3.5.1.1	Gemeinsamkeiten der SIRENA-Demonstratoren	47
3.5.2	Menüführung	48
3.5.3	Modellierung	50
3.6	Fazit des Usability-Design	50
4	Evaluationsstudie	51
4.1	Anforderungen der Benutzer	51
4.2	Die Methode	51
4.2.1	Theorie	51
4.2.2	Angewandte Methoden	52
4.3	Versuchsordnung	53
4.3.1	Übungsteil	53
4.3.2	Untersuchung	54
4.4	Ergebnisse	55
4.4.1	Allgemeines	55
4.4.2	Spezifische Ergebnisse	55
4.4.3	Das Fazit der Evaluationsstudie	56

5	Entwicklung	59
5.1	Ziele der Entwicklung	59
5.2	Laufzeitumgebung	59
5.3	Die Entwicklungsumgebung – das .NET Framework	60
5.3.1	C#	60
5.3.2	C# und seine Bedeutung im .NET Framework	62
5.3.3	Was ist .NET?	63
5.3.4	Das .NET Framework und seine Komponenten	63
5.3.4.1	Common Language Infrastructure (CLI)	66
5.3.4.2	Common Language Runtime (CLR)	66
5.3.5	Das .NET Compact Framework	67
5.3.6	Mono – ein Stück Unabhängigkeit	68
5.4	Programmierung	69
5.4.1	Zielsetzung	69
5.4.2	Die Gestaltungskomponenten	69
5.4.3	Benutzerverwaltung – Modi	71
5.4.4	Navigation	72
5.4.5	Farbige Buttons	73
5.4.6	Ausführung	73
5.4.7	Testphase	74
5.5	Das Fazit der Entwicklung	75
6	Usability-Tests	76
6.1	Ziele der Usability-Tests	76
6.2	Die Methode	76
6.2.1	Theorie	77
6.2.2	Angewandte Methode	77
6.3	Versuchsanordnung	78
6.3.1	Übungsteil	78
6.3.2	Untersuchung	80
6.4	Ergebnisse	81
6.4.1	Allgemeines	81
6.4.2	Spezifische Ergebnisse	81
6.4.3	Auswertung	81

Inhaltsverzeichnis

6.5	Die Untersuchungsmethode von Attrakdiff	85
6.5.0.1	Konfidenzintervall	85
6.5.0.2	Die Testmethode	86
6.5.1	Ergebnisüberblick	87
6.6	Das Fazit der Usability-Tests	89
7	Ergebnis	91
7.1	Fazit der Diplomarbeit	91
7.2	SIRENA Projektaussichten	91
A	Literaturverzeichnis	92
	Abbildungsverzeichnis	95
B	Glossar	97
C	Erhebungsbogen der Evaluationsstudie	100
D	Erhebungsbogen des Usability-Tests	104
E	Eigenständigkeitserklärung	112

0.1 Abkürzungsverzeichnis

AWebS	Automotive Web System
CAN	Controller Area Network
CLI	Common Language Infrastructure
CLR	Common Language Runtime
DAB	Digital Audio Broadcast
DPWS	Devices Profile for Web Services
ECMA	European Computer Manufacturers Association
EUREKA	Europäische Forschungs-Kooperation
FIRST	Fraunhofer Institut für Rechnerarchitektur und Softwaretechnik
FPP	Full Project Proposal
GENA	General Event Notification Architecture
GPRS	General Packet Radio Service
GPS	Global Positioning System
GSM	Global System for Mobile Communication
HMI	Human Machine Interface
HSCSD	High-Speed Circuit-Switched Data
HTTP	Hypertext-Transfer-Protocol
IEEE	Institute for Electrical and Electronic Engineers
IL	Intermediate Language
I/O	Input/Output
IP	Internet Protocol
IR	Infrared Radiation
IrDA	Infrarot Datenübertragung
ISO	International Standardization Organization
IT	Information Technology
ITEA	Information Technology for European Advancement
MES	Manufacturing Execution System
MMI	Mensch-Maschinen-Interface
MMS	Mensch-Maschinen-Schnittstelle
MSIL	Microsoft Intermediate Language
OSGI	Open Service Gateway Initiative
OS	Operating System

OSI	Open Systems Interconnection
OSI-RM	Open Systems Interconnection Reference Model
PDA	Personal Digital Assistant
PLC	Programmable Logical Controller
PMR	Professional Mobile Radio
P2P	Peer-to-Peer
QoS	Quality of Service
RAD	Rapid Application Development
RF	Radio Frequency
SDK	Standard Development Kit
SIRENA	Service Infrastructure for Real-Time Embedded Networked Applications
SLA	Service Level Agreement
SOAP	Simple Object Access Protocol
SSDP	Simple Service Discovery Protocol
TCP	Transmission Control Protocol
UDA	UPnP Device Architecture
UDP	User Datagram Protocol
UPnP	Universal Plug and Play
VoD	Video-on-Demand
WiFi	Wireless Fidelity
WLAN	Wireless Local Area Network
WP	Work Package
WSDL	Web Services Description Language
XML	Extensible Markup Language

1 Einleitung

By the year 2010, the Internet will have trillions of users it doesn't have today. Most of them will not be human beings.

– Harbor Research 2003 –

1.1 Ziel dieser Arbeit

Ausgangspunkt: Das europaweite Projekt SIRENA (**S**ervice **I**nfrastructure for **R**ead-Time **E**MBEDDED **N**ETWORKED **A**pplications) hat die Aufgabe, eine Protokollstruktur zu entwickeln, die es mobilen Endgeräten ermöglicht, ohne manuelle Konfiguration *Ad-Hoc*-Netze zu bilden und über einen gemeinsamen Kontrollpunkt gesteuert zu werden. Das Institut FIRST der Fraunhofer Gesellschaft hat bei diesem Projekt unter anderem die Aufgabe übernommen, einen Automotive-Demonstrator zu entwickeln. Radio, PC-Box, Navigationssystem und ein PDA (Personal Digital Assistent), der als Fernbedienung der anderen Geräte benutzt werden kann, sollen das entwickelte Framework in einem Automobil demonstrieren. Für diesen PDA war ein Mensch-Maschinen-Interface zu entwickeln. Die Planung, Programmierung und Implementierung des PDA-Interfaces sowie Usability-Tests sind Bestandteil dieser Arbeit.

Im Folgenden werden das Projekt SIRENA im Ganzen und der Automotive-Demonstrator im Speziellen näher erläutert. Danach folgt die Anforderungsanalyse an die Benutzeroberfläche. Hier wird besonderes Augenmerk auf Usability (Kapitel 3) gerichtet. Die gewonnenen Erkenntnisse werden in einem Konzeptentwurf vorgestellt. Es folgt die Dokumentation der Programmentwicklung basierend auf dem .NET

Compact Framework. Abschließend werden die durchgeführten Usability-Tests vorgestellt. Das Schlusswort rundet die Arbeit ab. Im Anhang finden sich Begriffserklärung, eine Liste der verwendeten Literatur, die Fragebögen der Evaluationsstudie und der Usability-Tests sowie die Eigenständigkeitserklärung.

1.2 Aufgabenstellung

Ziel der Arbeit ist es, eine effektive, anwendergerechte Benutzeroberfläche für einen PDA mit Windows-Pocket-PC-Betriebssystem als Laufzeitumgebung zu entwickeln. Die Aufgabe umfasst Anforderungsanalyse, Design mit Konzept- und Modellentwicklung, Programmierung, Test und Implementierung des Interface auf dem PDA.

Die Anforderungsanalyse teilt sich in die Ermittlung der zu implementierenden Hard- und Software, beschrieben in Abschnitt 2.5.3, und die Analyse der Benutzeranforderungen (Kapitel 3). Konzepterstellung und Modellierung basieren auf den Ergebnissen der Anforderungsanalysen und werden in Abschnitt 3.5 beschrieben. Programmiert wurde in der Sprache C# (gesprochen: c sharp) auf Basis des Microsoft .Net Compact Framework. Für die Usability-Tests wurde ein *Wizard-of-Oz*-Testszenario aufgebaut. Da keine der zu implementierenden Funktionen vor Beendigung dieser Arbeit verfügbar war, wurde der Funktionsumfang simuliert. Auch die Testfahrten fanden aus Sicherheitsgründen nicht im Straßenverkehr statt. Mehr zu Usability-Tests in Kapitel 6.

2 SIRENA

Dienste-Infrastruktur für echtzeitkritische, eingebettete, vernetzte Anwendungen

In diesem Kapitel wird das SIRENA-Projekt vorgestellt und in seinen einzelnen Komponenten erläutert. Anschließend werden der Automotive-Demonstrator und seine Hard- und Softwarekomponenten dargestellt. Dazu waren die Möglichkeiten bei FIRST selbst und den direkt beteiligten Projektmitgliedern zu erkunden. Hier galt es zu eruieren, welche Hardwarekomponenten, Dienste und welche gemeinsamen Designgrundlagen für den Demonstrator verwendet werden. Es waren eine Demonstrator-Szene zu entwerfen und die dafür notwendigen Komponenten zu akquirieren. Zeitgleich erfolgte die Analyse der Anforderungen der zukünftigen Nutzer. Da das Interface während der Fahrt benutzt werden soll, werden Benutzeranforderungen, Sicherheit und Wahrnehmung im Kapitel 3 besonders untersucht.

2.1 Ziele des SIRENA-Projekts

Das SIRENA-Projekt hat zum Ziel, ein Framework für eingebettete Dienst- und Laufzeitumgebungen verschiedener Domänen zu definieren und zu realisieren. Bei den Domänen handelt es sich um:

- Heimautomatisierung
- Industrie
- Automobilelektronik

Als verbindendes Glied wird ein Cross Domain Demonstrator entwickelt. Die marktübergreifende Ausbreitung der Domänen verlangt nach adäquaten Schnittstellen. SIRENA, ein Projekt des ITEA-Programms der Eureka-Initiative, hat die Aufgabe, hier eine stabile, übergreifende Lösung zu finden.

2.2 Die Projektpartner

Eine Auflistung der beteiligten Projektpartner findet sich in Abbildung 2.1.

Organisation	Nation	Domain
Fraunhofer FIRST	Deutschland	Automotive & Cross-Domain
Siemens SBS	Deutschland	Home
Paderborn University	Deutschland	Home
IXtronics	Deutschland	Automotive
Traveltainer	Deutschland	Automotive & Cross-Domain
Rostock University	Deutschland	Automotive & Cross-Domain
Materna	Deutschland	Cross-Domain
ESC	Deutschland	Automotive & Cross-Domain
INVERA	Deutschland	Cross-Domain
Dortmund University	Deutschland	Cross-Domain
Schneider Electric	Frankreich	Industrial
Cap Gemini	Frankreich	Industrial
Robotiker	Spanien	Industrial
ZIV	Spanien	Industrial

Abbildung 2.1: Mitglieder des SIRENA-Projekts

Die direkt am Automotive-Demonstrator beteiligten Projektpartner sind ESC und Traveltainer mit der AWebS-Box, siehe Abschnitt 2.5.3.2, und Web-Diensten im Auto. Die Universität Rostock stellt bluetoothfähige Sensoren zur Verfügung. Diese sollen zum Beispiel melden, wenn auf dem Beifahrersitz ein Kindersitz angebracht ist. Die Funktionen des Beifahrerairbags werden in diesem Fall abgeschaltet. Es werden bereits Kontakte zu Kindersitzherstellern geknüpft. Die Universität Dortmund ist zusammen mit der Firma Materna für das Sicherheitmanagement zuständig. Die Fir-

ma iXtronics entwickelt ein System für emissionsarmes Fahren. Das wird allerdings nur in den Ford Galaxy implementiert, das zweite SIRENA-Demonstrator-Fahrzeug. Dieser ist mit einem Hybrid-Motor ausgestattet.

2.2.1 Motivation

Komplexe Systeme verlangen nach Integration verschiedener Komponenten, Dienste und Anwendungen. Bisher isolierte Systeme wachsen zusammen und bieten neue und vernetzte Dienste und Anwendungen an. Die Zusammenarbeit von einer Domäne mit der anderen scheitert allerdings häufig daran, dass die spezifischen Eigenschaften einer Umgebung nachträgliche Erweiterungen oder Anpassungen nicht zulassen. Ein einheitlicher Rahmen fein strukturierter, anpassbarer Dienste über verschiedene Domänen hinweg wird hier mittel- wie langfristig erhebliche Produktivitätssteigerungen in der Systementwicklung zur Folge haben.

SIRENA hat sich zum Ziel gesetzt, diese Strukturierung zu leisten und eine flexible, skalierbare Dienste-Architektur zu definieren. Die Skalierbarkeit soll vom kleinen 8-bit-System über die gesamte Breite der gebräuchlichen Netzwerke gewährleistet sein. Die Entwicklung einer offenen Servicearchitektur und -infrastruktur, die sowohl das Anbieten allgemeingültiger als auch die Integration domänenspezifischer Services unterstützt, ist eine Herausforderung für alle Beteiligten.

2.3 Lösungsansatz des SIRENA-Projekts

In SIRENA wird der Lösungsansatz verfolgt, ein domänenunabhängiges Framework zur Realisierung von Anwendungen bereitzustellen. Dieses Framework beruht auf gemeinsamen Basisdiensten, die nicht domänenspezifisch sind, (z. B. Authentifizierung von Nutzern) und als Bausteine für die Realisierung höherwertiger, falls erforderlich, domänenübergreifender Anwendungen (z. B. Infotainment-Dienste im Auto und Haus) dienen. Dazu werden die Basisdienste eine Schnittstelle mit entsprechender formaler Beschreibung besitzen.

Der Fokus liegt daher auf der Entwicklung eines Dienste-Beschreibungsschemas, das anhand ausgewählter Beispiele aus den unterschiedlichen Anwendungsdomänen er-

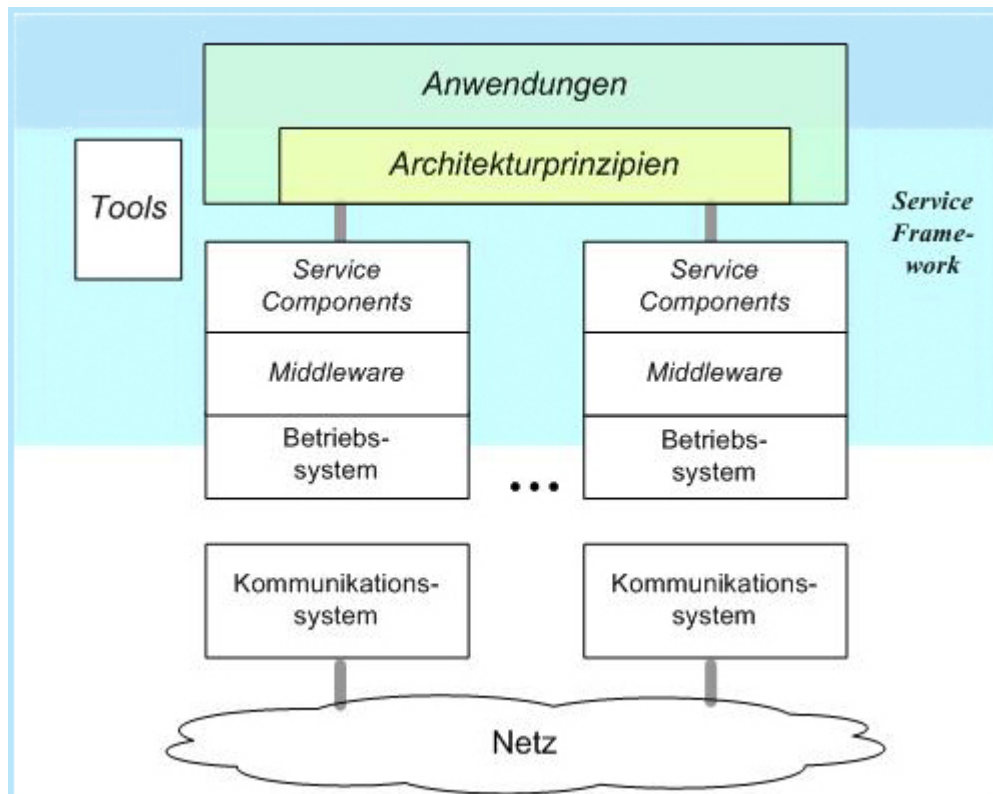


Abbildung 2.2: SIRENA-Framework Übersicht

arbeitet wird. Es geht nicht um die Entwicklung neuer Dienste, sondern um die Vereinheitlichung und Nutzung von Diensten, die bereits heute die Basis vieler Anwendungen sind, in der Regel aber immer neu und immer etwas anders entwickelt werden. Im Kern entspricht dieser Ansatz der Übertragung des bereits erfolgreichen Dienstebegriffs aus der Internet/Webtechnologie in die Welt der eingebetteten Systeme. Dabei sind natürlich ganz andere Anforderungen zu erfüllen, zum Beispiel im Hinblick auf Ressourcenverbrauch, Zuverlässigkeit und Performance. Eine Aufgabe besteht darin, sehr kleine Systeme interoperabel zu machen, wie ein Serverdienst in einem intelligenten Sensor. Die Basisdienste des SIRENA-Frameworks müssen deshalb unter anderem in hohem Maße skalierbar sein.

Wie Abbildung 2.2 zeigt, werden SIRENA-basierte Anwendungen von vernetzten Systemkomponenten getragen, die sich in Hardware und Betriebssystemen unterscheiden können. Die darüber liegende Middleware stellt grundlegende Unterstützungsfunktionen bereit und dient der Homogenisierung der Anwendungsplattform. Das

Service-Framework umfasst Erweiterungen zur Anpassung der Middleware an die Anforderungen der Service-Components und Tools zur erleichterten Planung, Konfiguration, Installation und Verwaltung der Anwendungen. Das Service-Framework wird bei der Entwicklung von Demonstratoren angewandt und erprobt.

2.3.1 Projektstatus

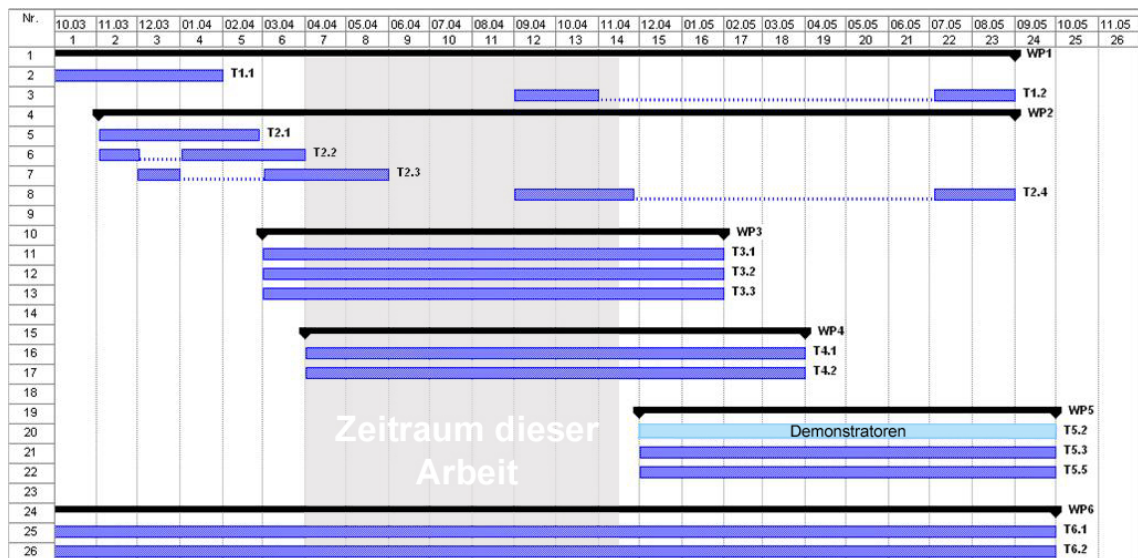


Abbildung 2.3: SIRENA-Projektplan

Das SIRENA-Projekt hat eine Laufzeit vom 1. Oktober 2003 bis zum 30. September 2005. Im ersten Drittel der Projektzeit wurden die im Folgenden vorgestellten Evaluierungen und Anforderungen erarbeitet. Ziel dieser Phase war, neben der Definition des eigentlichen Frameworks, die Festlegung einer Basistechnologie. In der weiteren Laufzeit wird diese verfeinert und ausgearbeitet, um schließlich in die Demonstratoren implementiert zu werden. Abbildung 2.3 zeigt die Planungszeit des Projekts mit ihren einzelnen Entwicklungsphasen. Der Zeitraum dieser Arbeit ist als hellgraues Feld markiert. Der geplante Entwicklungszeitraum für die Demonstratoren beginnt erst nach Abschluss dieser Arbeit. Das erklärt, warum die weiteren Komponenten des Automotive-Demonstrator auf sich warten lassen.

2.4 SIRENA-Technologien

Für das Gelingen des Projektes war es nicht wichtig, neue Technologien oder Protokolle zu entwickeln, sondern eher, Vorhandenes zu evaluieren und daraus eine gemeinsame, domänenübergreifende Basis zu schmieden. Die Auswahl der zu evaluierenden Technologien bezog sich in erster Linie auf solche, die eine spontane Vernetzung von Geräten erlauben und solche, die Service- und Gerätebeschreibungen ermöglichen. Die im Folgenden beschriebenen Dienste werden Bestandteile des SIRENA-Framework sein. Vor allem UPnP erfüllt alle Voraussetzungen, um als Basisprotokoll des SIRENA-Framework eingesetzt zu werden.

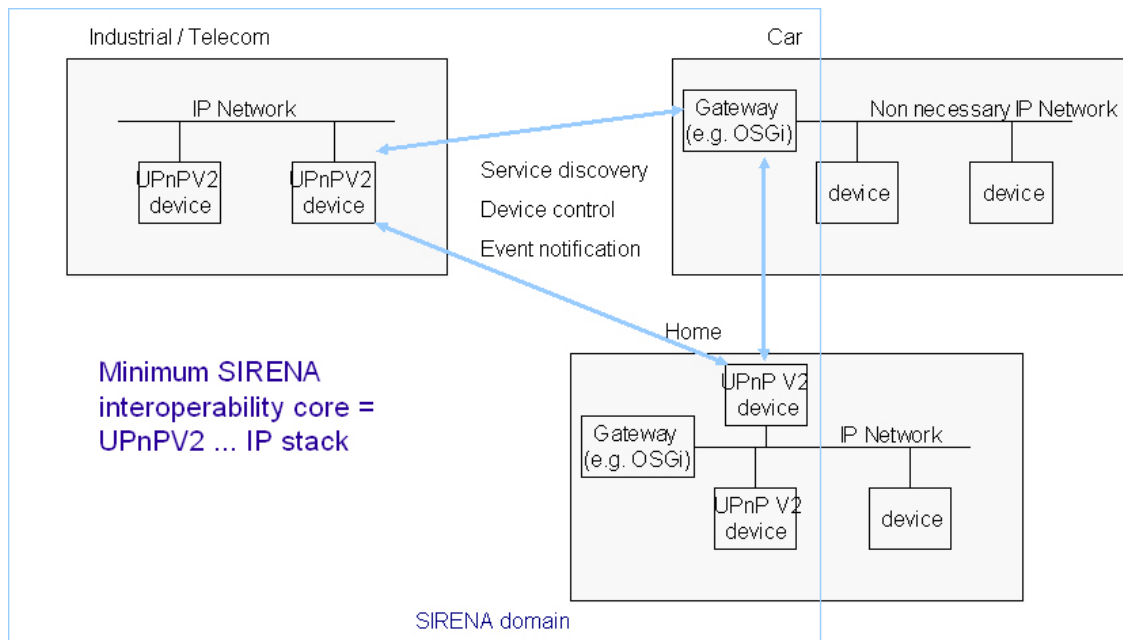


Abbildung 2.4: SIRENA-Konzept

Abbildung 2.4¹ vermittelt die geplante domänenübergreifende Funktionsweise des SIRENA-Framework auf Basis des UPnP Stack in der Version 2. Diese Version befindet sich noch in der Entwicklung. Dem SIRENA-Projekt wird voraussichtlich im Dezember 2004 ein UPnP v2 Stack zur Entwicklung des SIRENA-Framework zur Verfügung gestellt.

¹Die Grafik ist im Original auf der Homepage des SIRENA-Projekts zu finden [SIR].

Nach einer Entscheidung der Projektpartner im Oktober 2004 sollen die SIRENA-Demonstratoren auf Basis von UPnPv2 entwickelt werden. Aus dieser Entscheidung resultieren Konflikte in der weiteren SIRENA-Entwicklung. Als Technologie zur Ermöglichung von Softwareupdates während der Laufzeit oder als Gateway soll OSGI eingesetzt werden. Für OSGI besteht allerdings nur eine Schnittstelle zu UPnP 1.1, also der veralteten Version. Wann ein UPnPv2-kompatibler Stack bzw. Bundle erhältlich sein wird, ist noch offen. Die OSGI-Implementierung für SIRENA wird vom Fraunhofer Institut FIRST entwickelt. Eine Möglichkeit zur Lösung des Versionskonflikts wäre es, OSGI vorerst nur im SIRENA-Framework für den Automotive Demonstrator auf Basis der von UPnP 1.1 zu implementieren. UPnP-Version 2 wird unter dem Kürzel DPWS (Device Profile for Web Services) veröffentlicht. Das Konzept der Basisfunktionen von UPnP wird für DPWS beibehalten und nachfolgend genauer erläutert.

Für drahtlose Verbindungen werden Bluetooth und WLAN (Wireless Local Area Network) eingesetzt. Unterstützt wird außerdem das CAN-Protokoll, mit dem die Daten im Fahrzeug verbreitet werden. Nachfolgend werden die einzelnen Technologien genauer erläutert.

2.4.1 UPnP – Universal Plug and Play

UPnP ist ein offener Standard, der 1999 von Microsoft initiiert und von einem Industriekonsortium mit mehr als 650 Mitgliedern entwickelt wird. Ziel ist es, die Vernetzung und das Zusammenspiel von netzwerkfähigen Geräten zu erleichtern und so benutzerfreundlicher zu machen.

In der UPnP-Device-Architecture (UDA) wird die generelle Arbeitsweise der UPnP-Technologie festgelegt. Weiterhin werden für feste Geräteklassen, wie z. B. Scanner oder Drucker, allgemein gültige Beschreibungen definiert und veröffentlicht, so dass erkannte Geräte dieser Klasse über eine fest definierte Schnittstelle angesprochen werden können. Diese standardisierten Device-Templates einzelner Geräteklassen geben darüber Auskunft, welche Dienste ein Device anbieten muss und welche Dienste es darüber hinaus noch anbieten kann.

UPnP legt sich weder auf eine Programmiersprache noch auf eine Plattform, sondern nur auf die zu verwendenden Protokolle (SSDP, GENA, SOAP, HTTP, UDP, TCP,

IP) fest.

Die UPnP-Device-Architecture (UDA) definiert Protokolle, mit denen die Kommunikationsteilnehmer Nachrichten austauschen und setzt hierbei auf Standards wie HTTP und SOAP. UPnP-Geräte werden in Devices und Control-Points unterteilt, wobei ein Gerät auch beide Rollen einnehmen kann.

Ein Device kann mehrere Subdevices beinhalten. Devices sind in dieser Architektur Server und bieten Dienste an. Die ausgetauschten Nachrichten basieren auf XML und müssen dem jeweiligen Schema der entsprechenden Geräteklasse folgen. Ein Control-Point ist in der Lage, die von einem UPnP-Device angebotenen Dienste zu nutzen. In der UPnP-Device-Architecture werden sechs verschiedene Schritte beschrieben:

1. **Addressing**
2. **Discovery**
3. **Description**
4. **Control**
5. **Eventing**
6. **Presentation**

Im ersten Schritt, dem *Addressing*, erhält ein Device eine IP-Adresse. Im *Discovery* werden XML-Nachrichten verschickt, die ein Device und seine angebotenen Dienste bekannt machen. In einem weiteren Schritt kann über die *Description* eine genauere Beschreibung des Devices und der Dienste abgefragt werden. Nachdem ein UPnP-Gerät diese Schritte durchlaufen hat, können z.B. im Control-Schritt die eigentlichen Aktionsaufrufe erfolgen. Durch das *Eventing* werden Control-Points über Statusänderungen von Devices informiert. Für *Presentation* können *Devices* eigene HTML-Seiten anbieten, auf denen ihr Status ersichtlich ist, oder über die sie gesteuert werden können.

Sicherheitsvorkehrungen, um z. B. unberechtigte Zugriffe zu verhindern, sind in der UDA nicht definiert. Es liegen jedoch erweiternde Spezifikationen für sicherheitsbewusste Devices und Security-Konsolen vor. Jedes sicherheitsbewusste Device hat

einen Eigentümer. Dieser ist berechtigt, den Zugriff auf das Device zu reglementieren. Vor dem Ausführen einer Aktion prüft ein Device anhand der definierten Regeln, ob dieser Zugriff gestattet ist. Ein Control-Point, der als Eigentümer von Devices auftritt, heisst Security-Console und dient dem Management von sicherheitsbewussten Devices. UPnP ist derzeit in der Version 1.1 aktuell (Stand Oktober 2004), die Version 2.0 wird gerade entwickelt. Optimierungsbedarf besteht nicht nur in einem durchdachten Sicherheitskonzept, sondern auch bei der IP-Vergabe. Um den Anforderungen an das Internet und der wachsenden Zahl der Geräte gerecht zu werden, wird eine Erweiterung von der IPv4-Adressvergabe auf IPv6 implementiert. Die Adressierungsmöglichkeiten von IPv4 (4 Byte lang, darum IPv4 genannt) sind auf theoretisch maximal 2^{32} Adressen begrenzt. Durch die Verwendung von IPv6 (6 Byte) wird der Adressierungsraum auf 2^{48} theoretisch mögliche Adressen erweitert. Das mit UPnP vorgestellte Konzept wird in DPWS (Device Profile for Web Services) auf Basis von SOAP 1.2 und Web-Service-Addressing fortgeführt und um einige Komponenten erweitert.

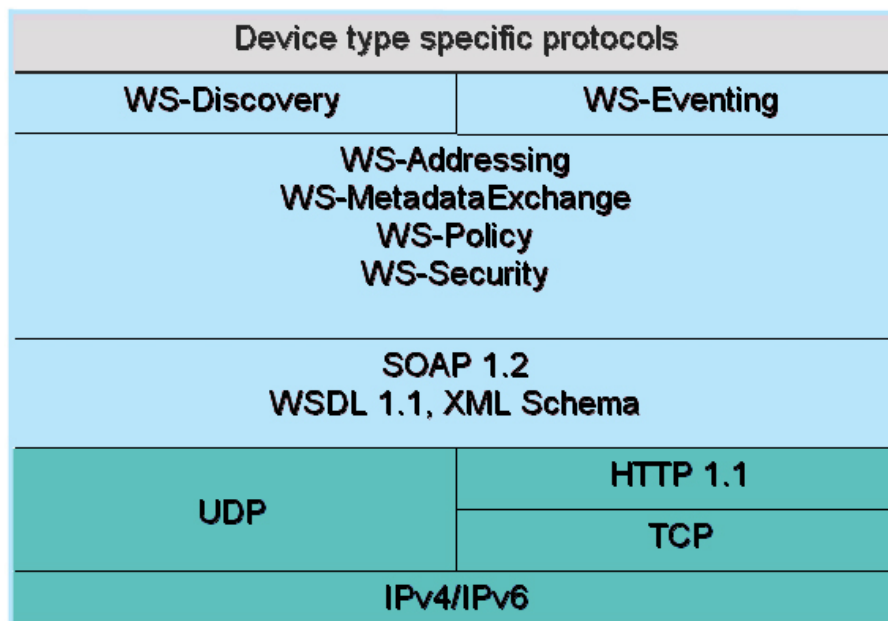


Abbildung 2.5: SIRENA-Protokollstack

Abbildung 2.5 beschreibt den Aufbau des SIRENA-Framework. Basierend auf IPv4- und IPv6-Adressen werden unter Verwendung der Standardprotokolle UDP, TCP

und HTTP 1.1 verschiedene Dienste, wie WS-Addressing, WS-Metadata-Exchange, WS-Discovery, WS-Policy, WS-Security, WS-Eventing unterstützt. Die Beschreibungssprachen werden XML und WDSL sein.

2.4.2 OSGI

Die OSGI (Open Service Gateway Initiative) Alliance wurde 1999 als ein non-profit Industriekonsortium gegründet. Bekannte Firmen, die der OSGI Allianz angehören, sind: IBM, Nokia, Bosch, Motorola, Philips, TeliaSonera, ProSyst und die Deutsche Telekom.

Ziel war es, eine Software-Plattform zu entwickeln, auf der Anwendungen und Dienste ausgeführt werden können. Inzwischen zum Java-Standard geworden, kommt das OSGI-Framework in Fahrzeugen, auf mobilen Endgeräten, in der Heimautomatisierung und -vernetzung oder in Industriebetrieben zum Einsatz. Dabei wird eine Umgebung zur Verfügung gestellt, auf der online ladbare Services ausgeführt werden können. Ein Service ist die kleinste ausführbare Programmeinheit und kann mit weiteren Services zu einem Bundle zusammengefasst werden.

Die Spezifikation der OSGI Service-Plattform definiert eine Java-basierte Laufzeitumgebung und Basisdienste. Ein bedeutendes Merkmal der Service-Plattform ist die Möglichkeit, dynamisch und kontrolliert Service-Anwendungen (sog. Bundles) zur Laufzeit einspielen und auch wieder entfernen zu können. Ein Einsatzgebiet von OSGI ist die integrierte Entwicklungsumgebung Eclipse. Plugins für Eclipse sind OSGI-Bundles. Abbildung 2.6 [All] skizziert die Service-Architektur von OSGI.

Eine OSGI Implementierung besteht in der Regel aus dem OSGI-Framework und einer Anzahl von Service-Bundles, die durch die modulare Architektur leicht hinzugefügt werden können. Dabei werden die Services, wie Java-Schnittstellen, beschrieben. Es gibt keine spezielle Form von Service-Beschreibungen. Es wird lediglich vorausgesetzt, dass jedes Bundle, das ein Service Interface zur Verfügung stellt, dieses korrekt implementiert. Abhängigkeiten zwischen den Bundles müssen explizit durch ein Import- oder Export-Statement hergestellt werden. Das Sicherheitsmanagement der OSGI-Service-Plattform basiert auf der Java-2-Spezifizierung. Dazu müssen die Java-Security-API's vom ausführenden Framework unterstützt werden. Es werden folgende Kontrollmöglichkeiten angeboten:

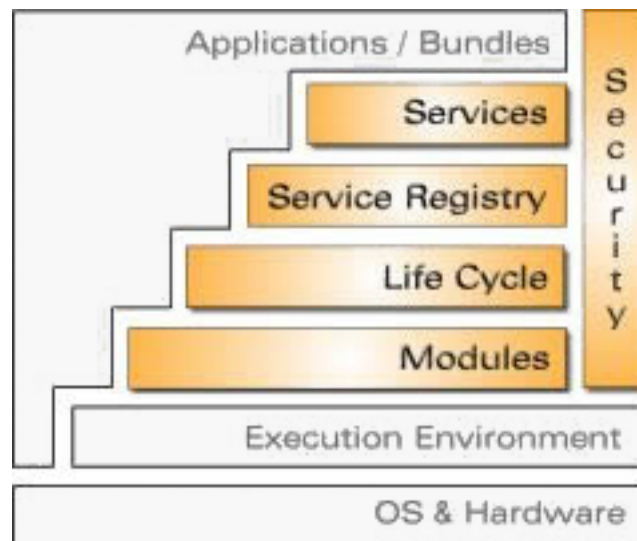


Abbildung 2.6: OSGI-Architektur

- **AdminPermission:**
Kontrolliert die Administration des Frameworks
- **ServicePermission:**
Kontrolliert Registrierung und die Anwendung von Services aus den Bundles
- **PackagePermission:**
Kontrolliert Im- und Export der einzelnen Datenpakete

OSGI unterstützt das UPnP-Protokoll. Damit ist es gut geeignet, im SIRENA-Framework eingesetzt zu werden.

2.4.3 Bluetooth

Bluetooth ist eine drahtlose Kommunikationstechnologie, welche auf kurzen Distanzen funktioniert und damit Verkabelungen von Geräten, wie Headsets, Mobiltelefonen und Laptops, vermeidet. Eine barrierefreie Sichtverbindung zwischen den kommunizierenden Geräten ist nicht notwendig, wie es etwa bei der Infrarottechnologie der Fall ist. Bluetooth nutzt das lizenzfreie 2.4-GHz-ISM-Band und unterstützt sowohl Daten- als auch Sprachübertragung bis zu 1 MBit/s brutto. Die Übertragungreichweite variiert je nach Geräteklasse zwischen 1 m, 10 m und 100

m. Ein Bluetooth-Pikonetz besteht aus maximal 256 Geräten, wovon höchstens acht aktiv sein können. Die restlichen befinden sich in einem inaktiven Zustand. Ein Gerät übernimmt die Rolle des Masters, die anderen sind Slaves und kommunizieren ausschließlich mit dem Master. Durch Überlagerungen von Pikonetzen lassen sich größere Netzwerke, die sogenannten Scatternetze, bilden.

Alternativ zur Bildung eines Scatternetzes kann ein Master die aktiven bzw. inaktiven Zustände in einem Pikonetz kontrollieren. Ein Gerätemanager übernimmt hier die zentrale Steuerung, indem Slaves gezielt geweckt bzw. in eine energiesparende Betriebsart versetzt werden. Die kurzen Distanzen, auf denen Bluetooth funktioniert, und die inzwischen weite Verbreitung machen den Einsatz auch in einem Kraftfahrzeug sinnvoll.

2.4.4 WLAN

Wireless LAN (Wireless Local Area Network) bezeichnet ein drahtloses lokales Funknetz-Netzwerk, wobei meistens ein Standard der IEEE 802.11-Familie gemeint ist. IEEE ist das *Institute for Electrical and Electronic Engineers*. Dieser Standard spezifiziert in der 802.11-Familie Protokolle des MAC-Bridging auf der 2. Schicht des OSI-Modells. WLAN nach 802.11b (maximal 11 MBit/s brutto) oder 802.11g (maximal 54 MBit/s brutto) funkt im 2,4-GHz-Band (Wellenlänge von ca. 13 cm). Kabellose Netzwerke spielen eine wesentliche Rolle bei der Ad-Hoc-Vernetzung unterschiedlicher Geräte in den verschiedenen SIRENA-Domänen.

2.4.5 CAN

Controller Area Network ist ein Protokoll, das 1981 von Bosch und Intel entwickelt wurde. Es findet seinen Einsatz auf den Daten-Bussen der meisten europäischen Fahrzeuge. CAN ist ein Standard der ISO (International Standardization Organization), der die untersten beiden Layer des OSI-Modells, den Physical-Layer und den Data-Link-Layer spezifiziert. Die wichtigsten Merkmale sind:

- Datenübertragungsgeschwindigkeit von 10 kBit/s bis 1MBit/s.
1 MBit/s reduzieren sich in der Realität auf effektive 500 kBit/s

- Buslänge bis 1 km (50 kBit/s)
- maximale Anzahl von Knoten im Bus: 32
bei Aufteilung in Teilnetze höhere Gesamtzahl möglich
- Multi-Master-Bus
- Kommunikation erfolgt ereignisgesteuert
- keine Empfängeradressen im Bus, jeder Teilnehmer hört alle Nachrichten (Broad-/Multicast)
- hohe Akzeptanz und weite Verbreitung durch Automobilhersteller

Über dieses Protokoll werden zum Beispiel Statusmeldungen des Fahrzeugs geschickt. Das passiert per Broadcast, jeder empfängt jede gesendete Nachricht. Das Abfragen dieser Statusmeldungen bringt interessante Aspekte des Fahrerkomforts mit sich. Im Nutzerprofil gespeicherte Einstellung der Außenspiegel sei nur ein Beispiel.

2.5 Die Demonstratoren

Ziel der Demonstratoren ist die Anwendung und Erprobung des SIRENA-Service-Rahmenwerks in einer realen Umgebung, die durch Anwendungen innerhalb der verschiedenen SIRENA-Domänen repräsentiert werden. Im Folgenden werden die Demonstratoren für die Automotive- und Home-Domäne beschrieben. Der Cross-Domain-Demonstrator wird die domänenübergreifende Interoperabilität des Sirena Rahmenwerks und seiner Dienste darlegen. Das Cross-Domain-Szenario ist noch vom Konsortium zu definieren.

2.5.1 Home-Demonstrator

Die Qualität eines Hotels definiert sich über die Ausstattung und Dienstleistungen, die dem Hotelgast geboten werden. Eine Ausstattung mit Telefon, Radio und Fernsehen wird als selbstverständlich vorausgesetzt. Eine Erweiterung dieser Ausstattung

ist ein Video-On-Demand-System. Dabei kann sich der Gast einen Film aus einer virtuellen Videothek aussuchen und auf dem TV-Gerät in seinem Zimmer abspielen. Der Vorteil für den Gast ist, dass die Filme nicht zu einem festen Zeitpunkt abgespielt, sondern individuell gesteuert werden können.

Herkömmliche Video-On-Demand-Systeme (siehe Seite 99) basieren auf einer zentralen Server-Komponente. Die Skalierbarkeit des Systems wird von dieser Komponente und der Netzwerk-Infrastruktur begrenzt. Ebenso müssen zur Sicherung der Verfügbarkeit viele Einzelkomponenten des Servers oder der gesamte Server redundant vorhanden sein. Die Beschaffenheit eines solchen Client-Server-Systems ist sehr kostenintensiv. Daher soll bei diesem Demonstrator ein kostengünstiges Video-On-Demand System mit hoher Verfügbarkeit auf Basis eines PC-Netzwerks entwickelt werden.

Mit der in SIRENA verfolgten serviceorientierten Vorgehensweise können lauffähige und einfach zu pflegende Systeme besonders effizient und kostengünstig geschaffen werden. Gleichzeitig werden Entwicklungs- und Wartungskosten reduziert. Die Applikation wird aus bestehenden Open-Source-Komponenten zusammengestellt. Bei Hinzufügen und Entfernen einzelner Clients muß Plug-and-Play gewährleistet sein.

Dazu wird die Peer-to-Peer-Technologie verwendet. Da in diesem Modell kein gesonderter Server existiert, sondern jeder Client zugleich Server ist, müssen geeignete effiziente Algorithmen zur Kontrolle und Steuerung des Datenverkehrs im Netzwerk sowie Strategien für die Datenhaltung und Verteilung der Videofilme entwickelt werden. Hierdurch soll eine erhöhte Verfügbarkeit des gesamten Systems erreicht werden. Die Benutzer des Systems sind Hotelgäste, deren EDV-Wissen und technisches Verständnis nicht vorausgesetzt werden können. Deshalb muss die personalisierte Benutzerschnittstelle einfach und intuitiv bedienbar sein. Darüber hinaus soll das System dem Hotelgast die direkte sowie durch Timer gesteuerte Aufnahme von TV-Sendungen ermöglichen.

2.5.2 Cross-Domain-Demonstrator

Der Cross-Domain-Demonstrator wird ebenfalls vom Fraunhofer Institut FIRST erstellt. Die genaue Szene ist noch nicht festgelegt. Demonstriert werden sollen die domänenübergreifenden Eigenschaften des SIRENA-Framework. Einen Entwurf da-

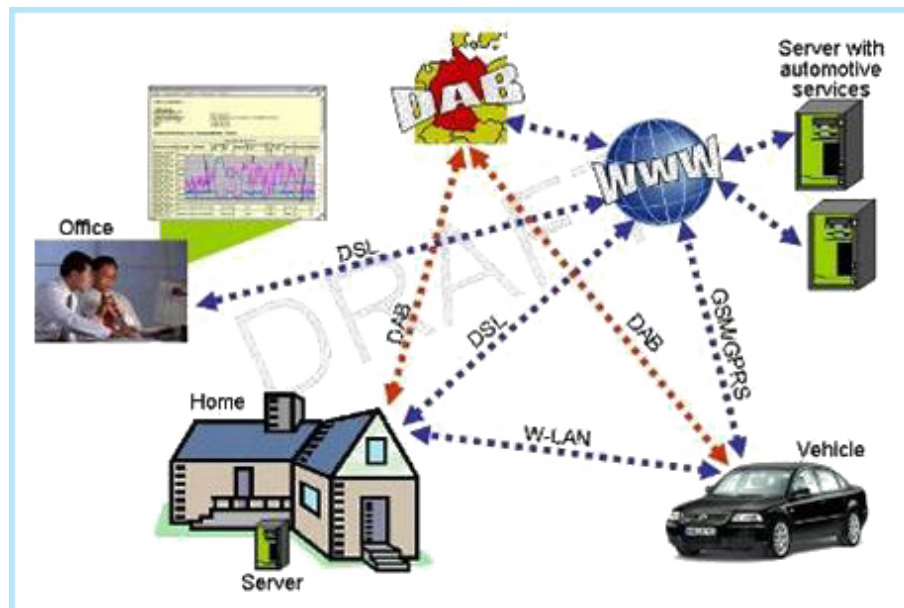


Abbildung 2.7: SIRENA-Cross-Domain-Demonstrator

zu zeigt Abbildung 2.7. Die domänenübergreifende Funktionalität von SIRENA wird anhand von Beispielanwendungen der Home- und Automotive-Domänen dargestellt.

2.5.3 Automotive-Demonstrator

SIRENA demonstriert das Framework an zwei Fahrzeugen: ein Ford Galaxy und das Fahrzeug der Mercedes-A-Klasse. Vorgestellt wird hier das Demonstrator-Fahrzeug des Fraunhofer Instituts FIRST, der A-Klasse-Wagen. Das Fahrzeug wird mit der AWebS-Box als zentrale Einheit ausgestattet. Darüber sind Internetzugang, Statusabfragen über den CAN-Bus und das von FIRST entwickelte Fahrtenbuch erreichbar. Das Woodstock-Autoradio bietet Digital-Audio-Broadcast-Technologie. Das geplante darzustellende Szenario zeigt Abbildung 2.8. Das SIRENA-Automobil besitzt ein interoperables, fahrzeugübergreifendes Netzwerk, in dem Services flexibel nach individuellen Bedürfnissen über einfach handhabbare Installationsmechanismen ergänzt werden können.

Die geschaffenen Interoperabilitätsmerkmale dieses Netzwerkes zwischen Mensch, Maschine und Umwelt eröffnen dem Fahrer und Mitreisenden neue Möglichkeiten zur Erweiterung und Personalisierung von Komfort- und Informationsdiensten. Die

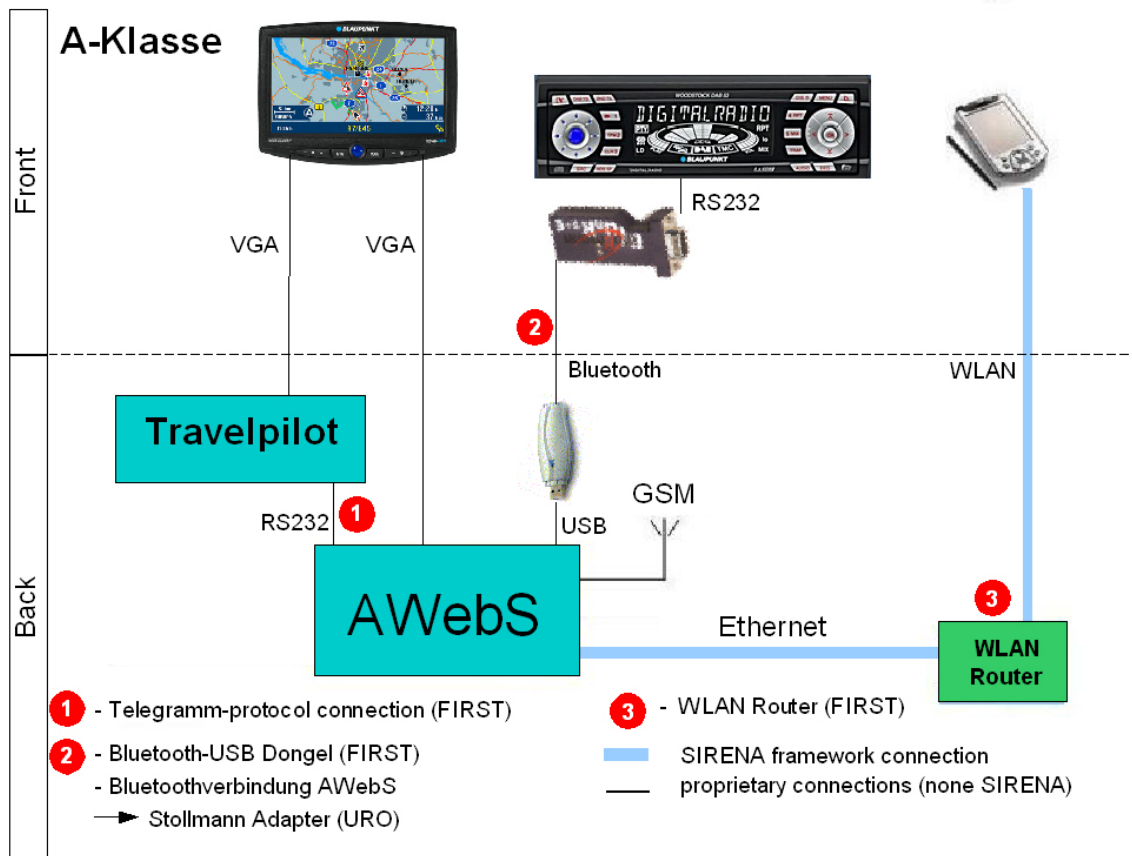


Abbildung 2.8: Automotive-Demonstrator-Szenario

Erfüllung der Marktanforderungen an Komfort- und Informationsdienste unter Verwendung des SIRENA-Frameworks führen zu tragbaren Entwicklungskosten und kurzen Innovationszyklen in der Automobilindustrie.

Mit der Bildung von Ad-Hoc-Netzen per UPnP in Client-Server-Manier unterstützt SIRENA die komfortable Handhabung verschiedenster Dienste. Adressierung und Konfiguration passieren dynamisch, d. h. ohne Interaktion des Benutzers. Der Fahrer kann sich auf die Bedienung einer Oberfläche konzentrieren, über die alle im Netzwerk für das Auto zur Verfügung gestellten Funktionen bereitstehen.

Praktisch entsteht hier die Möglichkeit, die Vielfalt der Services zentral zu verwalten und die Daten redundant wie mobil zu halten. Im Fahrzeug relevante Daten können auf verschiedene Arten auf andere Computer übertragen und aktualisiert werden. Zum kommerziellen Einsatz kommt diese Technologie z. B. bei Firmenfahrzeugen

für die Führung eines Fahrtenbuches. Kundenadressen werden aus dem Adressbuch übernommen, weitere Daten per Internet über AWebS auf dem Monitor des PDA angezeigt. Der Techniker ist stets aktuell informiert, sämtliche seiner Tätigkeiten sind dokumentiert, verarbeitete (Ersatz-) Teile werden per Datenbankabgleich registriert. Transparente Kundenbetreuung, automatisierte Rechnungen und Logistik ermöglichen maximale Effizienz der Arbeitsabläufe.

2.5.3.1 Direkt beteiligte Projektmitglieder

Die Firmen ESC und TraveTainer mit dem gemeinsamen Produkt, der AWebS-Box, und die Universität Rostock mit der SIRENA-Implementierung der Bluetooth-Komponenten sind die direkt am FIRST-Automotive-Demonstrator beteiligten Partner.

Zur Eruiierung der in der A-Klasse möglichen Statusabfragen war eine Vorführung des Wagens bei der Firma TraveTainer in Großwallstadt notwendig. Bei dieser Gelegenheit wurden parallel gemeinsame Designgrundlagen der Benutzeroberflächen der verschiedenen Demonstratoren besprochen. Ein Protokollauszug der Designvorschläge wird in Abschnitt 3.5.1.1 auf Seite 47, vorgestellt. Aufbau und Funktionen der AWebS-Box erklärt Abschnitt 2.5.3.2.

Die in Abbildung 2.8 dargestellte Systemarchitektur ist ein Beispiel, in dem ein SIRENA-Framework eingesetzt werden kann. Je nach Fahrzeugtyp und Ausstattung können die Systemkomponenten und Netzwerke variieren.

2.5.3.2 Automotive Web System - AWebS

Die AWebS-Box ist ein kleiner Computer, auf dem ein ganz normales Linux- oder Windows-Betriebssystem läuft. Da sie eine der Hauptkomponenten des Automotive-Demonstrators ist, bedarf sie einer detaillierten Vorstellung. AWebS bietet die Möglichkeit, im Fahrzeug eine Internetverbindung herzustellen und darüber verschiedene Web-Dienste in Anspruch zu nehmen.

Des Weiteren hat die ESC GmbH & Co. KG ein System zur Statusabfrage der über den CAN-Bus gesendeten Daten entwickelt. Die Daten von Fahrzeugsensoren werden zum Teil vom CAN gelesen, konventionell über Fahrzeugsensoren ausgewertet

oder simuliert. Diese Daten werden anderen Komponenten/Services in der Systemarchitektur zur Verfügung gestellt. So können Dienste beispielsweise interaktiv auf Fahrzeuginformationen reagieren. Dabei ist es je nach Fahrzeugtyp unterschiedlich, welche Geräte an den CAN-Bus angeschlossen sind und welche einen autarken Stromkreis haben oder rein mechanisch zu bedienen sind.

Die erreichbaren Informationen² der A-Klasse zeigt Abbildung 2.9. Verschiedene Services sind auf Fahrzeuginformationen, wie Geschwindigkeit, Position usw., angewiesen und andere benötigen diese zur optimalen ergonomischen und sicheren Benutzung.

ID	Byte	Bit	Wertebereich	Bemerkung
402	2	0...7	0 – 255	Bremsdruck
210	5	2	0 / 1	Bremse betätigt
308	2	0...7		Drehzahl highbyte, Umdrehungen pro Minute
308	3	0...7		Drehzahl lowbyte, Umdrehungen pro Minute
210	5	6	0 / 1	Gaspedal nicht betätigt
210	3	0...7	0 – 255	Gaspedalstellung
408	1	0...7	0 – 255	Tankinhalt in Liter

Abbildung 2.9: Statusinformationen der A-Klasse

Das AWebS bietet im Fahrzeug einen Zugang zum Internet über das GSM/GPRS-Mobilfunknetz und ermöglicht die Integration und Nutzung von Online-Services im Fahrzeug. Diese Kommunikationsmöglichkeit wird auch für Services des Cross-Domain-Demonstrator verwendet. Ein Browser im AWebS wird zur Darstellung von HTML-basierten Diensten und Menüs verwendet werden. Das Serviceangebot umfasst Verkehrsinformationen (Gefahrenstellen wie Stauenden und Baustellen), Wetter, Börse, WebCams, TV-Programm, E-Mail u. v. m. Im AWebS gehören Location-Based-Services zum Standard. So werden automatisch Hotels, Tankstellen oder Raststätten in der Standortumgebung gefunden oder beispielsweise die lokale Wettervorschau angezeigt. Im Automotive-Demonstrator werden die physikalischen Netze GSM/GPRS, DAB, Bluetooth, WLAN und CAN angewendet (siehe Glossar ab Seite 97).

²Mit freundlicher Genehmigung der Firmen ESC und Traveltainer.

Die flexible Erweiterbarkeit lokaler Fahrzeugfunktionen/Services soll im Bluetooth-Netzwerk durch Plug-and-Play-Erweiterungen demonstriert werden. Neue Komponenten im Bluetooth Netzwerk sollen automatisch erkannt werden und sich unter Berücksichtigung von Zulassungskriterien in das Fahrzeugnetzwerk einfügen lassen. Hierfür soll erstmals UPnP im Bereich der lokalen Fahrzeugvernetzung angewendet werden.

2.5.3.3 Digital Audio Broadcast (DAB)

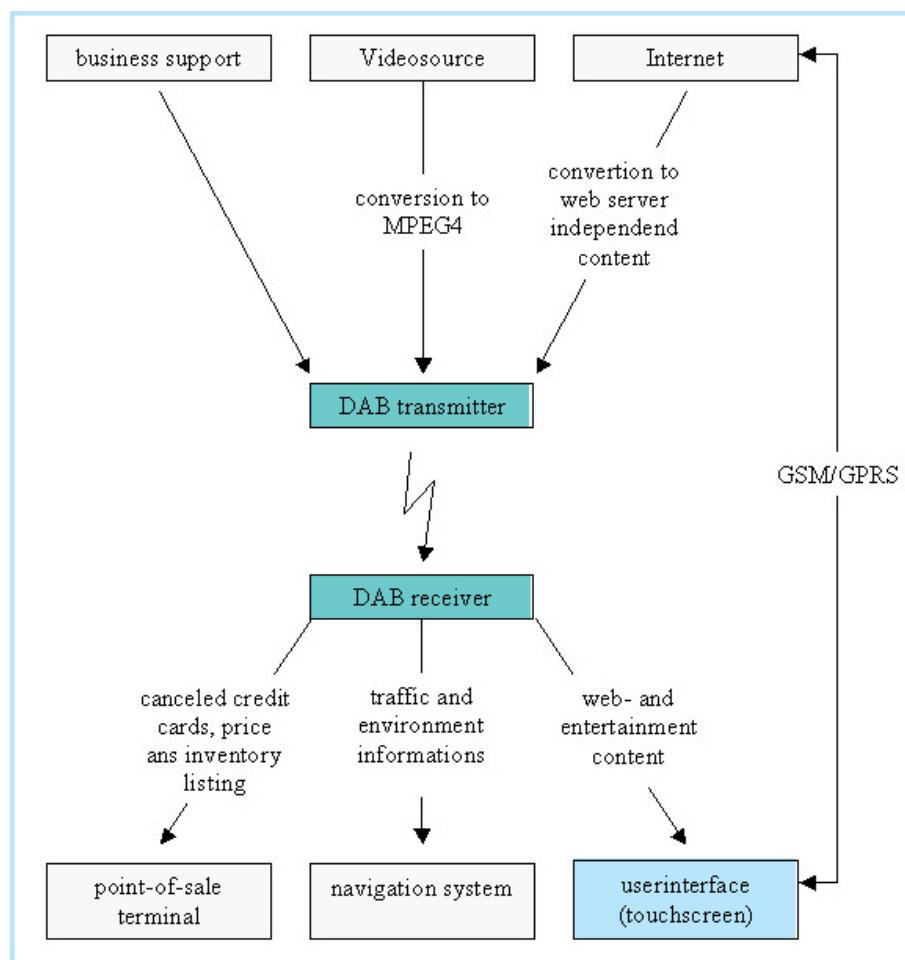


Abbildung 2.10: Digital-Audio-Broadcast

Dem Fraunhofer Institut steht ein Demonstrator-Fahrzeug der A-Klasse von der Firma Daimler-Chrysler zur Verfügung, des Weiteren das *Woodstock*-Autoradio der

Firma Blaupunkt mit Digital-Audio-Broadcast-Technologie (DAB).

Die Digital-Audio-Broadcast-Technologie sendet Radio in CD-Qualität. Mit dieser neuen Sendetechnik von Rundfunkdaten werden weitere Dienste ermöglicht. So werden jeweils aktuelle Informationen, wie Verkehrs- und andere Nachrichten, momentan gespielte Musik-Dateien, Bilder und Internetseiten, in einem Cache (Zwischenspeicher) abgelegt. Der Anwender kann diese Informationen individuell aufrufen und bei Bedarf speichern. Die Möglichkeiten und das Verfahren erklärt Abbildung 2.10 [SIR]. Informationen werden gesendet und über einen Receiver empfangen. Dieser leitet die Daten zur entsprechenden Anwendung. Das Angebot der verschiedenen Sendeanstalten ist variabel und vom Anwender über den PDA individuell abrufbar. Dabei werden die Daten via Bluetooth auf die AWebS-Box übertragen. Auf AWebS wird eine UPnP/DPWS-Bridge implementiert, die die Daten per WLAN auf den PDA sendet.

2.6 Fazit der Projektanforderungen

Die Hardwarekomponenten des Automotive-Demonstrator sind die AWebS-Box, das Woodstock-Autoradio und die Fernbedienung. Als Softwarekomponenten werden der TomTom-Navigator und das Fahrtenbuch implementiert. Die Benutzeroberfläche wird damit folgende Dienste anbieten können:

- **Fahrtenbuch (Logbook):**

- Zweck der Fahrt eingeben, weitere Fahrzeugdaten lesen

- Synchronisation mit AWebS

- Synchronisation mit PC

- **Radio:**

- Lautstärkeregelung

- Senderauswahl

- DAB-Dienste:**

- Nachrichten in Textformat

Nachrichten gesprochen

Bilder

Internetseiten

- **Navigator:**

Ziel eingeben

Karte anzeigen

...

- **AWebS:**

Verkehrsinformationen

Wetter

Tankstellen

Fast-Food-Ketten

3 Usability-Design

Was es nicht alles gibt, das ich nicht brauche!

– Aristoteles –

3.1 Ziele des Usability-Design

Ziel des Usability-Design ist die benutzergerechte Entwicklung einer Mensch-Maschinen-Schnittstelle.

Dieses Kapitel gibt vorab einen kurzen Überblick über die wissenschaftlichen Ansätze zu Gestaltungsgrundlagen interaktiver Systeme. Ein Einblick in die Wahrnehmungstheorien und die Möglichkeit des Menschen, Informationen zu verarbeiten, erklärt die kognitiven Möglichkeiten und Grenzen des Menschen.

Um den Anforderungen der zukünftigen Nutzer auch praktisch näher zu kommen, wurde bereits im Vorfeld eine Studie über Nutzerqualitäten vorhandener PDA-Benutzerschnittstellen durchgeführt. Auf dieser Basis werden die Usability-Aspekte¹ für die SIRENA-MMS eruiert. Das Menükonzept und ein Designentwurf der SIRENA-MMS runden dieses Kapitel ab.

¹Der Begriff *Usability* ist mit der Übersetzung von (engl.) *use* = Gebrauch, Nutzung und *ability* = Fähigkeit also Gebrauchsfähigkeit oder Benutzbarkeit zu erklären.

3.2 Allgemeine Aspekte des Interface-Design

Theorien, Normen und wissenschaftliche Grundlagen zur Gestaltung von Benutzeroberflächen gibt es viele. Gängige Ansätze sind:

- **Softwareergonomie**
- **Usability**
- **User-Centered-Design (UCD)**

Im Folgenden werden diese Herangehensweisen kurz beschrieben und ihre Relevanz in Bezug auf diese Arbeit erläutert.

3.2.1 Software-Ergonomie

Software-Ergonomie beschäftigt sich direkt mit Computerschnittstellen, während die Usability-Kriterien auf alle technischen Kontrollgeräte anwendbar sind.

Die Grundsätze der Software-Ergonomie wurden 1995 in der europäischen Norm DIN ISO 9241 festgelegt. Dafür wurde eine Arbeitsgruppe mit dem Ziel ins Leben gerufen, das weltweit verfügbare Wissen über Softwareergonomie zu sammeln, zu bewerten und in einer Norm festzulegen. Die Norm beinhaltet Hilfestellung bei der Konzeption, Gestaltung und Bewertung von Bildschirmarbeitsplätzen. Hier sind Mindestanforderungen für die ergonomische Gestaltung von Software beschrieben. Teil 10 der Norm enthält sieben Grundsätze, die als Leitlinien zur Leistungsbeschreibung, Bewertung und Gestaltung von Dialogsystemen angewendet werden sollen:

- Aufgabenangemessenheit
- Selbstbeschreibungsfähigkeit
- Steuerbarkeit
- Erwartungskonformität
- Fehlertoleranz

- Individualisierbarkeit
- Lernförderlichkeit

3.2.2 Usability

Diese Grundsätze finden sich ebenfalls in der Usability. Als die US-Air-Force im 1. Weltkrieg zunehmend Flugzeuge einsetzte, wurden leicht erlernbare und gut bedienbare Schnittstellen benötigt. Aus dieser Notwendigkeit entstand Usability. Jacob Nielsen [Nie93] nennt folgende fünf Attribute zur Berücksichtigung bei der Entwicklung von Benutzeroberflächen:

- **Learnability**
Das System soll leicht zu erlernen sein.
- **Efficiency**
Der User kann, nachdem er mit dem System vertraut ist, eine hohe Produktivität erzielen.
- **Memorability**
Der Benutzer soll sich schnell wieder am System zurechtfinden. Ein neuerliches Einarbeiten soll nicht notwendig sein.
- **Errors**
Das System soll eine geringe Fehlerrate aufweisen. Wenn ein Fehler auftritt, soll dieser mit einfachen Mitteln wieder ausgebessert werden können (Recovery). Katastrophale Fehler müssen im Vorhinein ausgeschlossen werden.
- **Satisfaction**
Das System soll angenehm zu bedienen sein. User sollen subjektiv gesehen das System gerne benutzen.

Für den projektspezifischen Aspekt der Sicherheit während der Fahrt wurden die Kriterien um folgende Punkte erweitert:

- **Aufmerksamkeit**
Das System soll mit geringst möglichem Aufwand an Aufmerksamkeit benutzbar sein.

- **Sicherheit**

Verkehrssicherheitsgefährdende Aktionen sind auszuschließen.

Die einzelnen Komponenten beeinflussen sich gegenseitig, so dass jede Ausprägung im globalen Zusammenhang betrachtet und entwickelt werden muss. Ein schwer bedienbares Produkt ist nicht effektiv, der Nutzer erreicht sein Ziel nicht und ist unzufrieden. Die Attraktivität des Produktes sinkt.

3.2.3 User-Centered-Design

Mit dem Ansatz des *User-Centered-Design*, das ebenso wie Usability seine Wurzeln bei den Arbeiten von Ben Shneiderman hat, findet sich Kritik an den starren Regeln der Usability. Einige Vorschläge würden voraussetzen, dass der Anwender grundsätzlich dumm sei (z. B. beim Web-Design jeden Link blau und unterstrichen zu markieren).

Allerdings war es durch solche festgelegten Grundlagen möglich, das World-Wide-Web in einer allgemein verständlichen Weise zu gestalten. Von Seiten der UCD wird der Usability aber Verrat an dem Grundsatz vorgeworfen, den Anwender von Anfang an als Individuum zu betrachten und nach seinen Fähigkeiten und Grenzen in die MMS-Entwicklung einzubeziehen.

Tatsächlich ist eine kritische Prüfung nötig. Führt man sich den Entwicklungsstand von Computern mit ihren Betriebssystemen und Anwendungen, wie sie vor 15 Jahren waren, vor Augen, so wird klar, dass einheitliche Designgrundlagen geschaffen werden mussten, um den Benutzer zu entlasten. Inzwischen ist der Umgang mit interaktiven Systemen selbstverständlich geworden. Auch die Systeme sind nicht nur in ihrer Bedienung, sondern auch in ihrer Ausstattung und Funktion effizienter und zufriedenstellender. Das bedeutet, dass der Anwender heutiger Systeme viel weniger technisches Wissen benötigt, gleichzeitig aber Erfahrungen im Umgang und mit allgemeinen Funktionsweisen interaktiver Systeme besitzt. So ist also nicht mehr stets vom *dümmsten anzunehmenden User (DAU)* auszugehen. In die Entwicklung der SIRENA-MMS sind die Anwender in drei Phasen direkt integriert:

1. Evaluationsstudie

2. Usability-Tests des ersten Prototypen
3. Usability-Tests der SIRENA-Demonstrator-Implementierung

Usability-Kriterien werden projektrelevant spezifiziert und bei Konzeption sowie Tests Entwicklungsgrundlagen für die SIRENA-MMS sein.

3.3 Menschliche Wahrnehmung und ihre Grenzen

Der Fahrer soll durch das SIRENA-MMI entlastet und unterstützt werden. Um das zu erreichen, ist es nötig zu wissen, wie der Mensch die Informationen, die er von der ihn umgebenden Welt empfängt, verarbeitet. Die SIRENA-MMS wird visuelle und auditive Reize aussenden. Bedient wird die Oberfläche mit dem Finger, dies löst beim Menschen den taktilen Reiz aus. Abbildung 3.1 gibt Auskunft über die vom Menschen wahrnehmbaren Bereiche dieser Reize (nach Luczak [Luc04]).

Wahrnehmungssystem	Bandbreite	Auflösung	Signal / Störung
Visuell	Wellenlänge: 400 – 700nm	Winkel: 1° - 30°	Kontrastverhältnis: 1:10
Auditiv	Frequenz: 20 Hz - 20 kHz; Schalldruckpegel: 0 – 140 dB	Frequenz: 1 Hz (bei 200 Hz) – 60 Hz (bei 10 kHz) Schalldruckpegel: 0,5 - 1,5 dB räumliche Lokalisierung: 1° (horizontal) – 10° (über Kopf)	20 dB
Taktil	<u>Berührungsreizschwelle</u> <u>Männer</u> : 9 mg (Lippe) bis 350 mg (Fußsohle), <u>Frauen</u> : 5 mg (Lippe) - 79 mg (Fußsohle) <u>Vibratorischer Schwellenwert</u> : 0,1µm (Fingerspitzen) bis 20 µm (Po), bei 200 Hz und 1 cm² Areal	<u>Berührung</u> : 14 - 18% bei einem Druck auf die Haut von 1 - 8 g/mm² <u>Vibratorische Amplitude</u> : ~20% konstant von 10 - 30 dB über Schwellenwert, bei 160 Hz an Fingerspitze; <u>Vibratorische Frequenz</u> : ~20 – 25% bei 20 - 300 Hz	

Abbildung 3.1: Schwellen menschlicher Wahrnehmung

Auditive Reize als Interaktionselemente werden von der SIRENA-MMS in Form kurzer Töne als informatives Feedback gegeben. Diese können nach Bedarf vom Nutzer abgestellt werden. Der Mensch kann eine eng begrenzte Anzahl symbolhafter

Töne unterscheiden, so wird es zwei Töne geben, der eine für Erfolgsmeldungen, der andere für das Anzeigen von Fehlern.

Manche Bewegungsabläufe lassen sich mit einem Minimum an Aufmerksamkeit durchführen. Beispiele dafür sind das *blinde* Greifen nach der Maus am Computer. Mit ein paar Tastebewegungen haben sich die Finger orientiert und die Finger bedienen mehrere Tasten. Dies ist auch für die SIRENA-MMS angestrebt. Als Orientierung des Tastsinns auf der PDA-Oberfläche dienen lediglich die Ecken und Kanten des Displays. Damit sind die oben und unten angeordneten Buttons leicht zu finden. Unterstützend würden hier Gravierungen oder Klebmarken in der Mitte der längsten Kante wirken, um die Lage der mittleren Buttons zu verdeutlichen.

Der Hauptanspruch an Aufmerksamkeit der SIRENA-MMS wird an das visuelle System des Anwenders gestellt. Rezeptoren für visuelle Reize sind die Stäbchen (Schwarz-Weiß-Sehen) und Zäpfchen (Farbsehen) auf der lichtempfindlichen Schicht des Augenhintergrunds, der Netzhaut. Mit einer Reaktionszeit von 200 - 400 ms (nach Luczak [Luc04]) sind sie sehr empfindlich für Bewegungen und Veränderungen im peripheren Gesichtsfeld, wobei der Mensch 7 ± 2 Dinge gleichzeitig im Kurzzeitgedächtnis erfassen oder unterscheiden kann. Das menschliche Sehfeld ist allerdings begrenzt. Genaue Auskunft darüber gibt die Abbildung 3.2 ([Kar02]).

Die Grafik macht deutlich, dass im horizontalen Sehfeld eine Farbunterscheidung bis zu einem Winkel von 60° möglich ist. Die Halterung des PDA im Auto wird im allgemeinen auf dem Amaturenbrett in einem horizontalen Winkel von $15 - 30^\circ$ angebracht. Der vertikale Winkel beträgt hier etwa $\pm 20^\circ$. So ist es dem Menschen möglich, den Verkehr zu fokussieren und gleichzeitig die Farbinformation des PDA wahrzunehmen. Ein grobes Modell von dem, was in den 200 - 400 ms der Verarbeitung visueller Reize passiert, zeigt Abbildung 3.3.

Entwickelt wurden unterschiedliche Modelle der Informationsverarbeitung, die sich im wesentlichen in Stufen- und Ressourcenmodelle unterteilen lassen. Die Stufenmodelle, wie in Abbildung 3.3 skizziert, gehen von einer sequentiellen Abarbeitung mehrerer Stufen der Verarbeitung vom empfangenen Reiz bis zur Reaktion aus. Ressourcenmodelle gehen von begrenzter Kapazität aus. Die Kapazität variiert dabei nach dem Erregtheitszustand. Ein Beispiel sei ein Gespräch mit dem Beifahrer während der Autofahrt. Beides kann zur gleichen Zeit verarbeitet werden. Hat der

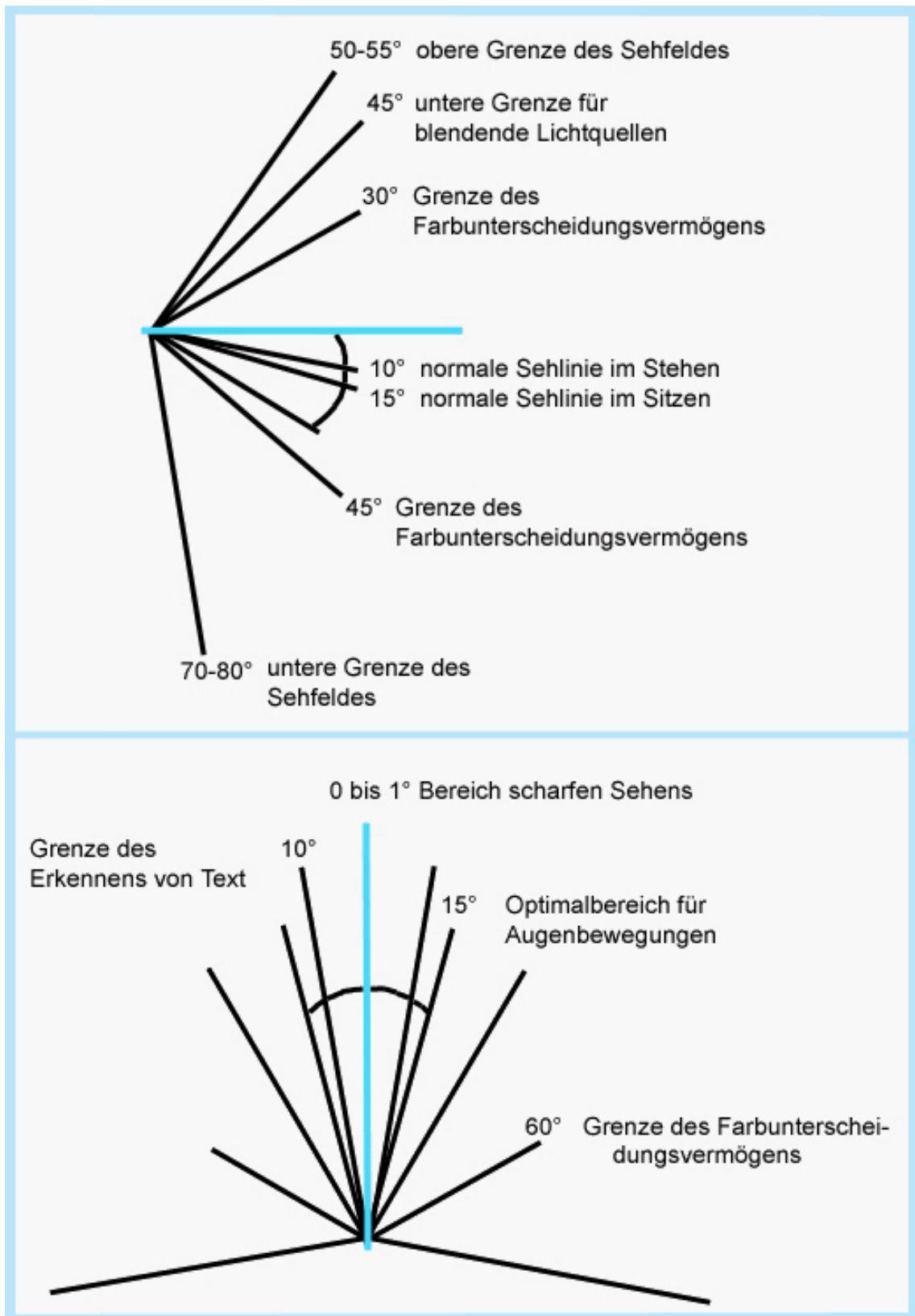


Abbildung 3.2: Vertikales und horizontales Sehfeld des Menschen



Abbildung 3.3: Stufen der Verarbeitung vom Reiz bis zu seiner Wahrnehmung

Autofahrer auf aktuelle Verkehrssituationen zu reagieren, die mehr Aufmerksamkeit verlangen, bricht das Gespräch ab. Die gesamte Kapazität wird für das Fahren benötigt.

3.3.1 Methoden des Erkennens

Für das Erkennen von Objekten werden zwei gleichzeitig ablaufende Prozessarten verantwortlich gemacht: Top-Down- und Bottom-Up-Prozesse. Top-Down-Prozesse bearbeiten den tatsächlich vorhandenen Reiz. Bottom-Up-Prozesse vergleichen mit Bekanntem, hier wird das Gedächtnis befragt. Diese Prozesse laufen parallel ab und sind nicht steuerbar. Daraus resultiert das Unvermögen die eigene Aufmerksamkeit zu bündeln. Ein erklärendes Beispiel hierzu basiert auf dem STROOP-Test (nach

[Luc04] S. 14), dargestellt in Abbildung 3.4.

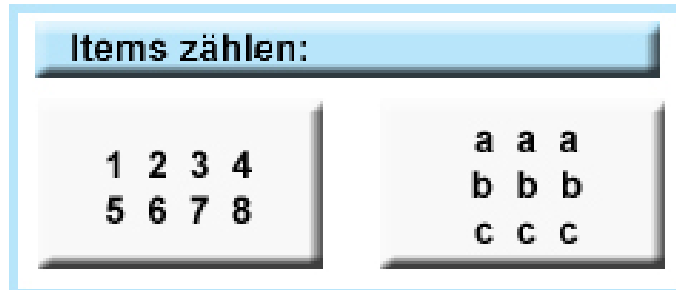


Abbildung 3.4: Trotz der Anweisung zu zählen, lässt sich das Lesen der Items nicht verhindern.

3.3.2 Visuelle Wahrnehmung

Bereits 1923 beschrieb Max Wertheimer in dem Buch *Untersuchung zur Lehre der Gestalt* Gestaltgesetze, die noch heute in erweiterter Form in der visuellen Wahrnehmung Gültigkeit besitzen. Sie erklären den Zusammenhang zwischen grafischer Darstellung und kognitiver Wahrnehmung. Einige davon sind:

1. **Das Prinzip der Ähnlichkeit**
Ähnliche Objekte scheinen eine Gruppe zu bilden.
2. **Das Gesetz der Nähe**
Nahe beieinander liegende Reize werden eher als zusammengehörig empfunden als voneinander entfernte Reize.
3. **Das Gesetz der Vertrautheit**
Dinge scheinen am ehesten eine Gruppe zu bilden, wenn die Gruppe vertraut oder bedeutsam erscheint.
4. **Gesetz der Geschlossenheit**
Nicht vorhandene Teile eines Reizganzen werden in der Wahrnehmung ergänzt, unvollständige Figuren als zusammengehörend wahrgenommen.

5. **Gesetz der Ähnlichkeit**

Einander ähnliche Elemente (Elemente gleicher Form oder Farbe) werden als zusammengehörig erlebt.

6. **Gesetz der guten Kurve**

Elemente, die ein gemeinsames Schicksal haben oder eine gute Kurve bilden (also sich z. B. gemeinsam bewegen oder gleichförmig verändern), werden als Einheit wahrgenommen.

7. **Gesetz der Kontinuität**

Elemente gleicher Form, die fortlaufend miteinander verbunden sind, werden als Einheit erlebt.

8. **Gesetz der Symmetrie**

Symmetrische Elemente werden als zusammengehörig erlebt.

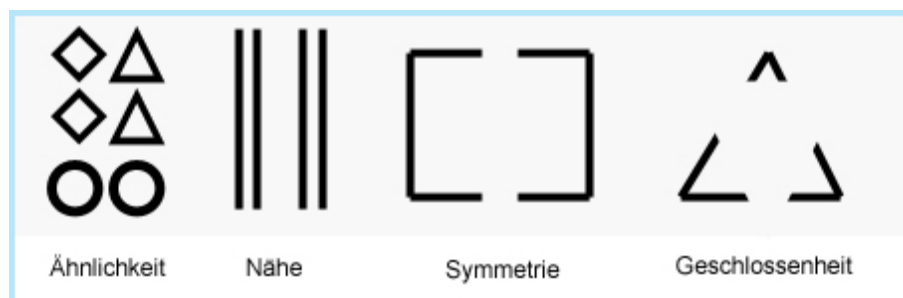


Abbildung 3.5: Beispiele zu den Gestaltgesetzen

Benutzeroberflächen bestehen aus einer Vielzahl von Details, die zusammen das Gesamtbild schaffen. Der Anwender wird sich ein mentales Modell der Darstellung ableiten. Ist diese Darstellung einheitlich und konsistent strukturiert, erfährt der Anwender optimale visuelle Unterstützung. Ein dargestelltes Muster kann das Erkennen der Gesamtheit behindern, wenn die Gesetze der Ähnlichkeit und der Nähe – verletzt werden.

3.4 Projektspezifische Aspekte des Interface-Design

3.4.1 Sicherheit

Geräte und Elemente, die neu entwickelt wurden und deshalb gesetzlich noch nicht erfasst sind, können nahezu ungeprüft im Fahrzeug benutzt werden². Erst wenn sich ein Anstieg von Unfällen oder anderer, sicherheitsbedenklicher Einschränkungen ergeben, wird der Gebrauch geprüft und gegebenenfalls verboten (bekanntestes Beispiel: Handybenutzung). Hier liegt die erste Verantwortung also beim Entwickler, die Sicherheit im Verkehr zu gewährleisten. Dazu stehen die bereits genannten Designtheorien und Gestaltgesetzen sowie Usabilitytestmöglichkeiten zur Verfügung. Doch selbst, wenn all diese Möglichkeiten ausgeschöpft werden, bedeutet das nicht, dass der Fahrer dieses System nutzen kann, ohne dass seine Aufmerksamkeit in sicherheitsbedenklichem Maße vom Verkehr abgelenkt wird. Dies wird am Beispiel des BMW 745i deutlich. Hier wurden bekannte Funktionen, wie z. B. Schaltknüppel und Handbremse durch Knopf bzw. Hebel am Lenkrad ersetzt. In der Mittelkonsole befindet sich dafür der Dreh- und Drückschalter zur Bedienung des On-Board-Systems. Vielleicht ist die neue Darstellung der Elemente leichter zu bedienen. Doch bis sich der Anwender an die Neuordnung gewöhnt hat, ist er verunsichert und muß unangemessen viel Aufmerksamkeit für die Bedienung der Instrumente aufwenden.

Auch die Bedienung des Autoradios während der Fahrt kann Unfälle auslösen. Um bei einer Vielzahl kleiner Tasten die richtige zu finden, behält der Fahrer das Gerät mindestens zwei Sekunden im Blick. Bei einem Tempo von 130 km/h legt das Fahrzeug in dieser Zeit 72 Meter zurück, auf denen die Verkehrssituation unbeachtet bleibt³.

So liegt es letztendlich in der Verantwortung des Fahrers, zu entscheiden, welche Tätigkeiten während der Fahrt nebenbei verrichtet werden und welche nicht.

Wünschenswert wäre eine Initiative, zum Beispiel des ADAC, die spezielle Testverfahren anbietet, so dass neu entwickelte On-Board-Systeme auf ihre Verwendbarkeit

²Notwendig ist der Nachweis der elektromagnetischen Verträglichkeit durch *CE*- oder *E*-Zeichen(siehe Straßenverkehrsordnung 23 Abs. 1 a Handyverbot)

³Aus der Sendung 7. Sinn vom 21.09.2004

während der Fahrt überprüft werden können. Es ist allerdings zu befürchten, dass die meisten Autoradios solchen Tests nicht standhalten können.

3.4.2 Eingabe

Zu diesem Problem gibt es mehrere Lösungsansätze, mit denen sich bereits führende Automobilhersteller auseinander gesetzt haben. So wurde zum Beispiel die Steuerung des Navigationssystems per Spracheingabe oder zusätzlichen Dreh-Drück-Schaltermodulen angeboten. Japanische Automobilhersteller bevorzugen die Eingabe über den Touchscreen, wobei die Eingabefelder so groß angelegt sind, dass sie mit den Fingern zu bedienen sind. Im Folgenden werden die gängigsten Eingabemethoden vorgestellt und projektspezifisch betrachtet:

- **Stifteingabe**

Ein PDA wird üblicherweise mit einem Stift bedient. Die Eingabefelder können durch die punktuelle Eingabe sehr klein gehalten werden. Speziell zur Eingabe von Text wird der Stift, zusammen mit dem PDA-Tastaturersatz, dem Eingabe-Panel, verwendet. Die aufzuwendenden feinmotorischen Bewegungsabläufe setzen Konzentration und visuelle Fokussierung voraus. Der Straßenverkehr wird während dieser Aktion sicher ausgeblendet. Allein die Möglichkeit, dass während der Fahrt der Stift herunterfällt ist nicht zu verantworten. Eine Eingabe per Stift während der Fahrt ist für das SIRENA-MMI also sicher auszuschließen. Daraus resultiert die Notwendigkeit, einen Fahrer- und einen Nichtfahrermodus einzurichten.

- **Sprache**

Die Sprachsteuerung stellt ein eigenes Interface dar, das gesondert entwickelt werden müsste und die SIRENA-MMS als Auswahloption optimieren würde. Da sich diese Eingabeform noch in einem Stadium der Entwicklung befindet, weil es an Komfort und intuitiver Bedienbarkeit mangelt, wird eine Implementierung für den Automotive-Demonstrator vorerst nicht in Betracht gezogen. Die Projektpartner der Universität Rostock entwickeln parallel an einer Universal-Plug-and-Play-gesteuerten Spracheingabe. Diese soll als eigenständiger Kontrollpunkt dienen. Eine Implementierung in die SIRENA-MMS des

Automotive-Demonstrator wird mit der Integration der Funktionen der endgültigen Demonstrator-Version geprüft.

- Dreh-Drückschalter

Zumeist in der Mittelkonsole untergebrachtes Eingabemodul zur Bedienung des Navigators und/oder anderer erreichbarer Geräte. Für ein PDA ist es wenig sinnvoll, diese Eingabemethode zu wählen. Sie liegt als zusätzliche Komponente fern des PDA und der Blickrichtung des Fahrers und benötigt Informationsverarbeitungsvorgänge, die ebenso eingespart werden können. Manueller und visueller Reiz haben mehrere Quellen, müssen koordiniert abgearbeitet werden und führen zu einem höheren sequentiellen Bedienaufwand.

- Touchpad

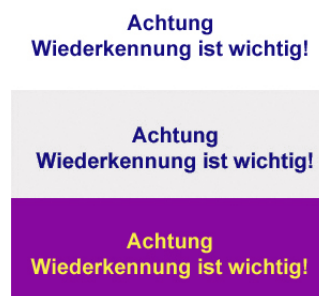
Den Touchpad zu benutzen, ist die zu eruiende Eingangs-idee zur Gestaltungsgrundlage des SIRENA-MMS. Die Buttons müßten so groß gestaltet sein, dass sie auch mit breiten Fingern leicht zu treffen sind. Das dient zudem der Übersichtlichkeit der Elemente. Geringe Anzahl, wie gleichbleibende Anordnung der Elemente, erfordert nur ein geringes Maß an Aufmerksamkeit. Die direkte Bewegung mit dem Finger auf die entsprechende Stelle lässt sich ohne Fokusänderung ausführen, sofern der PDA im wahrnehmbaren Bereich des Gesichtsfeldes angebracht ist (siehe 3.2).

- Fernbedienung

Eine am Lenkrad befestigte Fernbedienung, über die der PDA zu bedienen ist, soll als weitere Eingabemethode besonders betrachtet werden. Der erste Vorteil ist, dass die Hände während der Bedienung am Lenkrad bleiben können. Einfaches Vorwärts- und Rückwärtsnavigieren, verbunden mit einer Enter-Taste, machen die Bedienung einfach und intuitiv. Dazu wurden in Betracht kommende Geräte untersucht. Wichtige Kommunikationsvoraussetzung ist hier die Bluetooth-Unterstützung. Außerdem sollte ein SDK zur Entschlüsselung der Protokollstruktur zur Verfügung stehen. Diese Voraussetzungen erfüllte eine Fernbedienung der Firma Blaupunkt. Eine Implementierung ist geplant. Leider ist die bestellte Komponente nicht vor Beendigung dieser Arbeit eingetroffen.

3.4.3 Modi – weniger ist mehr

Ein weiterer Aspekt, den es zu betrachten gilt, ist der Lichtwechsel. Bei Nacht wirken weiße Elemente auf einem Display ähnlich wie eine Lampe, sie leuchten und können blenden. Sicherlich ist es richtig, unter dem Menüpunkt *Einstellungen* einen Moduswechsel anzubieten. Ist ein Alleinfahrer über mehrere Stunden auf der Autobahn unterwegs, kann zwischendurch schon mal die Sonne auf- oder untergehen. Da wäre ein intelligentes Interface nützlich, das von allein merkt, wann ein Moduswechsel nötig ist und ihn selbstständig durchführt. Dazu könnte man eine Statusanfrage an das Scheinwerferlicht anbinden (AWebS), das den Moduswechsel bei Ein- und Ausschalten aktiviert⁴. Eine weitere Möglichkeit ist die Kopplung an einen Lichtsensor, der im Innenraum des Wagens oder optimalerweise am PDA selbst befestigt ist. Der Moduswechsel wäre an die Überschreitung von vorher festgelegten Schwellwerten gebunden. Dadurch könnte flexibel auf sich ändernde Lichtverhältnisse reagiert werden (z. B. durch Dimmen der Displaybeleuchtung bis zum notwendigen Moduswechsel). Ein Nachtmodus dunkelt sehr helle Farben ab. Wichtig ist hier die Wiedererkennung. So sollten z. B. alle weißen Elemente durch die gleiche Farbe oder einen abgedunkelten Ton ersetzt werden, statt ein ganz neues Design zu kreieren. Sonst kann es passieren, dass der Fahrer nach Sonnenuntergang den Radiosender wechseln will und bei Tempo 180km/h die Oberfläche nicht wiedererkennt.



Des Weiteren wird der Usability-Aspekt der *leichten Erlernbarkeit* verletzt, wenn dem Beherrschen des Tagesmodus das Neulernen der Nacht-MMS folgt. Ein Fahrer, der nur eben den Radiosender wechseln will, ist hier überfordert. Die Sicherheit, der Attraktivitätswert der MMS und aller dahinterstehenden Funktionen sinkt. Deswegen wurde die Möglichkeit vorgezogen, diesen Moduswechsel zu vermeiden, indem

⁴Bei Reisen nach Skandinavien wäre der Nachtmodus durchgängig eingeschaltet.

das MMS gleich in einem Nachtmodus entwickelt wird, der auch bei Tage gut erkennbar ist.

So gibt es lediglich einen zusätzlichen Modus, den Fahrermodus. Ist der PDA mit der AWebS-Box eines Fahrzeugs verbunden und bekommt die Meldung, dass der Tachostand größer als Null ist, wird in den Fahrermodus geschaltet. Im Fahrermodus werden die sicherheitsgefährdenden Funktionen (z. B. Texteingabe, komplexe Internet-Seiten) in grau dargestellt, damit der Fahrer weiß, welche Funktionen erreichbar sind. Außerdem ertönt ein Signalton.

3.5 Design

Mit Design ist hier die visuelle Gestaltung gemeint, der in dieser Arbeit ein hoher funktionaler Wert zugesprochen wird. Gehen allerdings funktionale und visuelle Gestaltung nicht miteinander konform, wird die Bedienung der MMS als schwierig und frustrierend empfunden. Die Struktur des Designs sollte also auf die Struktur der implementierten Anwendungen abgestimmt sein. Wünschenswert ist ein hoher Wiedererkennungswert der einzelnen Anwendungen (Radio, Navigationsgerät etc.). Dazu werden firmeneigene Logos, Farben und Schriften verwendet. Informationen müssen mit einem Blick erfassbar sein, die folgende Aktion sollte möglichst *blind* ausgeführt werden können. Das Wiedererkennen von Anwendungen *auf einen Blick* wird mit Hilfe von Farben unterstützt. Zuordnung von Farben registriert das Gehirn wesentlich schneller als das Lesen von Buchstaben. Dazu ein kleiner Test:



Das Gehirn hat die Farbinformation bereits registriert, bevor die Buchstaben ihre Informationen liefern. Vorsicht ist hier geboten, zuviel Farbenspiel erscheint schnell chaotisch und unübersichtlich. Eine einheitliche Struktur und ein einfarbiger Hintergrund, in einer dunklen Farbe gehalten, wirken beruhigend und rahmen die Menüfelder ein. Ein dunkler Hintergrund hat den weiteren Vorteil, dass er während der Fahrt im Dunkeln nicht blendet.

3.5.1 Konzept

Das Konzept der SIRENA-Automotive-Demonstrator-Benutzeroberfläche beruht auf drei Säulen:

1. höchstens 6 Elemente pro Fenster, angeordnet in gleichbleibender Struktur
2. Navigation mit dem Finger und/oder einer Fernbedienung am Lenkrad
3. Farbkonzept zur leichteren Wiedererkennung

Basis des Farbkonzepts ist die Fähigkeit des Menschen, Farben schnell und in einem weiten Sehfeld wahrzunehmen. So wird auf der Hauptmenüseite jeder Funktion/Button eine Farbe zugeordnet: Das Fahrtenbuch in grün, Navigator magentarot, Radio gelb, AWebS-Dienste orange. Für die Navigationsbuttons und Texte in Status- oder Fehlermeldungen wird die Farbe Azurblau verwendet. Das Farbmenü dient der leichteren visuellen Orientierung.

3.5.1.1 Gemeinsamkeiten der SIRENA-Demonstratoren

Um eine Wiedererkennung der SIRENA-Demonstratoren zu gewährleisten, haben die Projektpartner nachfolgende Eigenschaften zur Gestaltung der Demonstrator-Oberflächen vorgeschlagen. Dabei handelt es sich um einen Auszug aus dem Projektprotokoll. Einige Vorschläge konnten aus konzeptionellen Gründen nicht umgesetzt werden. Diese sind entsprechend kommentiert.

- Hintergrund: Dunkelblau mit Sirena-Logo ohne Sirena-Schriftzug
Dunkelblau wurde als Hintergrundfarbe übernommen, ein Sirena-Logo wäre durch die Größe der Buttons als irritierende Farbunreinheit zwischen den Buttons erkennbar und wurde nicht implementiert.
- Schriftfarbe auf Hintergrund: Weiß
Da weiße Schriftfarbe in der Dunkelheit blenden würde, wurde kein Weiß eingesetzt.
- Schrift: Arial/Arial Black

- Fokusrahmen der Buttons orange (#FFB000)
- Icons mit eher technischer Symbolik, ähnlich dem Sirena-Design erstellen
- Kein Scrollen in den Menüs, sondern Menüführung auf mehrere Seiten verteilt
Dies wurde für die Menüführung übernommen. Bei der Auswahl der Radiosender zeigte sich eine Scrollbar allerdings als übersichtlichste Auswahlfunktion.
- 6 Menüpunkte pro Seite einschließlich vor und zurück bei Menüteilung
- *back*-Taste in jedem Untermenü

Damit ist der kreative Rahmen eng gesteckt. Die Platzierung der Menüpunkte wird gleichmäßig über die Fläche von 240 x 320 Pixel verteilt. Die vier oberen Buttons bilden das Menü, die zwei unteren navigieren entweder eine Ebene zurück oder auf die Startseite. Auf der Startseite selbst werden hier der *tools*-Button, der zum Hilfemenü und den Einstellungsmöglichkeiten führt, und der *exit*-Button, der die gesamte Anwendung schließt, platziert.

3.5.2 Menüführung

Ben Shneiderman [Shn02] hat acht goldene Regeln des guten Designs aufgestellt. Sie sind für jede Umgebung interpretierbar und gehen noch etwas weiter als die Leitlinien der Software-Ergonomie (ISO 9241-10). Darum werden sie hier vorgestellt und projektspezifisch angewandt. Aufbau der Benutzeroberfläche nach den acht goldenen Regeln:

1. **Streben nach Konsistenz**

Durch den strengen Aufbau der Seiten mit jeweils höchstens sechs Buttons, von denen zwei immer gleich bleiben (*home*, *back*), wird eine konsistente Navigation ermöglicht. Der Anwender weiß stets, was ihn erwartet.

2. **Regelmäßigen Benutzern die Verwendung von Short-Cuts bieten**

Geplant ist eine Favoriten-Seite, die vom Anwender mit vier favorisierten Menüpunkten ausgestattet werden kann.

3. Informatives Feedback geben

Die Buttons ändern ihre Farbe bei einem Click-Event. Diese Farbe bleibt so lange bestehen, bis der Befehl ausgeführt bzw. eine neue Seite geladen ist. Zusätzlich bestätigt ein leises Ticken die Ausführung der gewählten Funktion.

4. Entwerfen in sich geschlossener Dialoge

Auf jede Aktion erfolgt eine Reaktion. Sind die gewünschten Funktionen ausgeführt, beendet der Anwender den Menüpunkt mit dem Drücken des *home*-Buttons.

5. Fehlervermeidung und einfache Fehlerbehandlung

Auf Fehler folgen Messageboxen, die den Fehler und Methoden zu seiner Behebung beschreiben. Allerdings kann es lediglich durch fehlende Verbindung zu einem aufgerufenen externen Gerät zu Fehlern kommen.

6. Erlauben einer leichten Umkehr von Aktionen

Fehlerhafte Navigation in der Menüauswahl ist durch *home-* und *back*-Buttons jederzeit umkehrbar. Das von FIRST erstellte Fahrtenbuch erlaubt durch einen *back*-Button zu jeder Zeit eine Umkehr von Eingaben und die Rückkehr zum vorhergehenden Eintrag.

7. Unterstützung des internen Kontrollbedürfnis

Durch die konsistente Menüführung ist Vorhersehbarkeit gewährleistet. Statusmeldungen sind an einen Timer gebunden und schliessen nach einigen Sekunden selbstständig. Alle anderen Menüpunkte müssen explizit geschlossen werden.

8. Reduzieren der Belastung des Kurzzeitgedächtnis

Eingehend auf die besondere Situation des Fahrers sind die Fenster so angelegt, dass jeweils zwei feste Bestandteile (*back-* und *home*-Button) und höchstens vier weitere Elemente dargestellt werden. So wird forciert, dass sich der Anwender ein einfach strukturiertes, konsistentes, mentales Modell der Benutzeroberfläche ableiten kann.

3.5.3 Modellierung

Der Aufbau eines durchgängigen Farbkonzepts ist in mehrfacher Hinsicht schwierig. Zum ersten werden Farben individuell wie kulturell unterschiedlich empfunden. Zum zweiten ist der Menüumfang entscheidend, ob so ein Farbkonzept anwendbar ist oder nicht. Ist der Umfang zu groß, wirkt eine Vielzahl von Farben eher verwirrend als symbolisch leitend. Bei vier Hauptmenüpunkten mit höchstens vier Untermenüs bleibt das Konzept überschaubar.

Die ersten Prototypen entstanden auf dem Papier. Ein UML-Diagramm mit den zu implementierenden Funktionen war Vorlage der Menüstruktur (siehe Abbildung 5.7 auf Seite 69). Dabei stellt es sich als schwierig heraus, ein durchgängiges Farbdesign mit einem technisch anmutenden Design zu verbinden. Spiegelnde Oberflächen, wie Glas oder Metall, haben weiße Schatten, damit sich ihr Effekt entfaltet. Das würde bei Dunkelheit blenden und den Fokus der Aufmerksamkeit auf sich ziehen. Farbige Buttons wirken unruhig. Darum wurden schließlich dunkelblaue Buttons kreiert, die mit farbigem Schriftzug und gleichfarbigem Rand versehen sind. Ein Icon bildet den Blickfang und symbolisiert die Funktion. Die Systembuttons sind einheitlich in türkis gehalten, ebenso die Schriftfarbe bei Fehler- und Statusmeldungen. Das ist gut zu erkennen und blendet bei Dunkelheit nicht. Die anderen vier Menübuttons haben grün (Farbe des Fraunhofer Instituts), das Logo des Fahrtenbuchs ist das SIRENA-Automotive-Logo. Der Navigator, als helfendes Element, wurde die Farbe Rot/Magenta zugeordnet, dem Radio sonniges Gelb für Unterhaltung. Die AWebS-Dienste sind in Orange gehalten.

3.6 Fazit des Usability-Design

Damit ist ein ebenso simples wie eingängliches Design gelungen. Der geübtere Anwender benötigt das Farbkonzept nur noch als Anhaltspunkt. Darum ist es sinnvoll, ein paar anspruchsvollere Designs zur Verfügung zu stellen.

Die Anforderungen des Benutzers werden zuvor jedoch im Rahmen einer Evaluationsstudie genauer eruiert.

4 Evaluationsstudie

4.1 Anforderungen der Benutzer

Die Gestaltungsmöglichkeiten für eine Benutzeroberfläche sind vielfältig. Für dieses Projekt haben Übersichtlichkeit und intuitive Menüführung einen besonderen Stellenwert. Sie sind entscheidend für die Sicherheit im Straßenverkehr. In dieser Studie wird der Ist-Zustand untersucht. Kriterien der Usability wirken sich auf jeden Anwender anders aus. Der eine Anwender liest zuerst die Hilfe-Datei, bevor er seine Aufgaben löst, für einen anderen Benutzer ist es eine Herausforderung, alles allein herauszufinden. Die bei den Tests verwendeten Interfaces haben zum Teil schwere Usability-Defizite. Simple Funktionen wie Löschen eines Termins oder Schließen einer Anwendung zum Beispiel, sind in Untermenüs versteckt. Vor allem die Oberfläche des DAB-Services ist dem Anwender schwer zugänglich. Was einen PDA-Benutzer stimuliert und unterstützt, was ihn frustriert und ihn seine Tätigkeit als anstrengend empfinden lässt, wurde durch diese Untersuchung in Erfahrung gebracht.

4.2 Die Methode

4.2.1 Theorie

Das Benutzen eines PDA während der Autofahrt birgt ein hohes Sicherheitsrisiko. Gesetzliche Bestimmungen werden in der Regel erst verabschiedet, wenn sich Mängel in Unfällen realisieren und dadurch zum offensichtlichen Verkehrsrisiko geworden

sind. Das beste Beispiel dafür ist das Handyverbot während der Fahrt.

Nachfolgender Test soll herausfinden, ob das erstellte Konzept:

- höchstens 6 Buttons, davon 2 gleichbleibend zur Navigation
- Eingabe mit dem Finger und/oder einer Fernbedienung am Lenkrad

bei den zukünftigen Anwendern auf Akzeptanz stößt. Die Probanden waren aufgefordert, während der Tests laut zu denken. So kam es zu offenen Gesprächen, in denen das Konzept diskutiert wurde. Da die Anforderungen der Nutzer zeitgleich mit den Anforderungen an die Funktionalität eruiert wurden, konnten die hier gewonnenen Anregungen und Erkenntnisse optimal in die Systementwicklung integriert werden. Verschiedene Anwendungen wurden miteinander verglichen (manuelle, Windows-, DAB-Anwendung).

4.2.2 Angewandte Methoden

Um die Daten zu erfassen, wurden mehrere Methoden angewandt. Die Versuche fanden unter Einzelbetreuung statt. Jeder Proband bekam das gleiche PDA mit dem Windows-Pocket-PC-Betriebssystem, dazu einen Fragebogen mit 25 Fragen, unterteilt in drei Frageblöcke. Im ersten Teil des Fragebogens wurden kurze Aufgaben gestellt, die gleich nach der Ausführung zu beurteilen waren.

Die Bewältigung der Aufgaben wurde versteckt gefilmt. Die Kamera war sehr offensichtlich in einem Karton untergebracht und etwa 70 cm von der Testperson entfernt auf der Fensterbank platziert. Durch ein großes Loch an der Vorderseite des Kartons war die Kamera deutlich erkennbar. Trotzdem waren alle Probanden überrascht, als sie nachträglich um ihre Einwilligung zur Verwendung des Filmmaterials gebeten wurden.

Mit Hilfe der Aufzeichnungen per von Bild und *lautem Denken* konnte auch nachträglich noch differenziert ausgewertet und unterschieden werden, wo die Schwierigkeiten liegen.

Dem praktischen Teil der Fragen liegt ein geschlossenes Antwortschema zugrunde. Die Antwort-Items sind in vier Stufen eingeteilt, so dass die Probanden eine Aussage in Richtung gut oder schlecht wählen mussten (siehe 4.1)

sehr gut	gut	holprig	gar nicht
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Abbildung 4.1: Antwortitems in der Evaluationsstudie

Den Abschluss des praktischen Teils bildete eine qualitative Frage, hier waren die Probanden aufgefordert, positive wie negative Eindrücke der Anwendungen wiederzugeben. Absicht des praktischen Teils war es, eine gemeinsame Beurteilungsgrundlage für alle Probanden zu schaffen und die Schwierigkeiten in der manuellen Bedienung zu ermitteln.

Der zweite Frageblock bezieht sich auf das Benutzen des PDA während der Autofahrt. Hier waren Aussagen zu möglichen Eingabemethoden und zur Abschätzung der eigenen Fähigkeiten vorgegeben, denen die Probanden bei Übereinstimmung zustimmen sollten. Im letzten Frageblock wurden personenbezogene Fragen gestellt.

4.3 Versuchsanordnung

Die Testgruppe bestand aus 15 Personen. Alle waren Autofahrer. Auf Erfahrungen mit dem PDA wurde keinen Wert gelegt. Das Windows-Betriebssystem war allen Probanden bekannt. Akquiriert wurden die meisten Testpersonen durch eine Rundmail, in der alle Autofahrer angesprochen waren. Einige Teilnehmer wurden persönlich eingeladen. Die Tests fanden einzeln, an einem speziell hergerichteten Arbeitsplatz statt. Der Platz war mit dem PDA, einem Autoradio, das mit der DAB-Anwendung per Bluetooth angesprochen werden konnte und dem dreiseitigen Fragebogen ausgestattet. Als PDA-Eingabemethoden standen ein Eingabestift und der Wippschalter am Gerät zur Verfügung.

4.3.1 Übungsteil

Zunächst sollten die Teilnehmer den PDA mit seinen Eingabemöglichkeiten und einige einfache Anwendungen kennenlernen. Jede Frage enthielt Anweisungen, deren Ausführbarkeit anschließend zu beurteilen war. Die Aufgaben waren so gestellt, dass sie am Anfang schnell zum Erfolg führten. Am Ende fanden sich zwei Aufgaben, die

schwerer zu lösen waren (lesen des Radiotextes, der war als solcher nicht titulierte; löschen eines zuvor eingetragenen Termins im Kalender). Hier sollte der Umgang der Probanden mit Problemsituationen beobachtet werden.

4.3.2 Untersuchung

Im Fokus standen:

1. **Umgang mit dem Eingabestift**
Anmeldung per Kennwort, Navigation, Tastatur zur Texteingabe
2. **Eigenständiges Nutzen des Wippschalters**
Navigation
3. **Wiedererkennung des Windows-Betriebssystems**
Navigation, Programme finden, öffnen, schließen
4. **Applikationen in Windows ausführen**
Kalender öffnen, Termin notieren, später Termin wieder löschen
5. **DAB-Applikation**
öffnen, Radiotext finden und lesen, Applikation schließen

Der Test begann mit dem Einschalten des PDA, dem Finden des Stiftes und der Anmeldung per Kennwort. Danach wurde der Kalender gesucht und ein Termin eingetragen. Dieser Teil war einfach und sollte dem Probanden durch Erfolgserlebnisse das Gefühl von Sicherheit vermitteln. Testrelevant waren hier die verschiedenen Eingabemethoden. Anschließend sollten die DAB-Applikation geöffnet und der Radiotext gelesen werden. Auch diese Aufgabe war relativ einfach zu lösen; schwierig gestaltete sich das Erkennen des Radiotextes als solchen. Der Text läuft standardmäßig, ist aber nicht durch ein Label als solcher bezeichnet. Anschliessend wurde der eingetragene Termin wieder gelöscht. Das ist meist erst durch mehrmaliges Versuchen oder Verwenden der Hilfe möglich. Nach Beendigung des praktischen Teils wurde nach besonderen Eindrücken gefragt. Im zweiten Teil waren einige Aussagen bezüglich der Handhabung des PDA während der Autofahrt getroffen. Die Probanden sollten ankreuzen, falls sie zustimmten. Des Weiteren war interessant, ob die Hilfe benutzt

wurde; falls nicht, war eine Begründung erbeten. Am Ende wurden Fragen nach Alter, Geschlecht, Sehkraft und Erfahrungen im Umgang mit PDA und Autofahren gestellt.

4.4 Ergebnisse

4.4.1 Allgemeines

Die Testgruppe setzte sich aus 20 % Frauen und 80 % Männern zusammen, eine Zusammensetzung, die dem PDA-Benutzerprofil entspricht. Beruflich fanden sich Studenten, Ingenieure und Sekretärinnen. Auffällig war, dass bei knapp der Hälfte der Teilnehmer die Sehkraft eingeschränkt war. Die meisten waren kurzsichtig, Farbfehlsicht kam nicht vor. Der Deutsche Ärzteverband bestätigt die Ergebnisse, laut einer Studie sind über 50 % der über 16-jährigen Deutschen fehsichtig.

4.4.2 Spezifische Ergebnisse

Vergleichend getestet wurden die manuelle Eingabe per Eingabestift, was für 40 % der Testpersonen eine komplett neue Erfahrung war, die Windows-Anwendung, die als PC-Betriebssystem allen bekannt war, und die für alle Testpersonen neue und wenig intuitiv gestaltete DAB-Anwendung.

Interessanter Aspekt ist hier das relativ schlechte Abschneiden der Windows Anwendung. Diese war, zumindest vom PC her, allen bekannt. Trotzdem beschwerten sich viele Probanden über schwerfällige Wiedererkennung, unerwartet platzierte Elemente und wenig intuitive Menüführung.

Die manuelle Eingabe wurde positiv beurteilt. Auch diejenigen, die noch gar keine Erfahrung mit dem PDA hatten, kamen mit dem Eingabestift so gut zurecht, dass keiner auf die Idee kam, den Wippschalter am Gerät zum Navigieren zu nutzen. Die Probanden waren sich allerdings einig, dass sich die Eingabe mit dem Stift nicht für die Autofahrt eignet. Als Begründungen wurde angegeben, dass der Stift leicht herunterfallen könne und das Treffen des Touchscreens mit dem Stift zuviel Feinmotorik verlangen würde. Die Texteingabe sollte dem Fahrer während der Fahrt

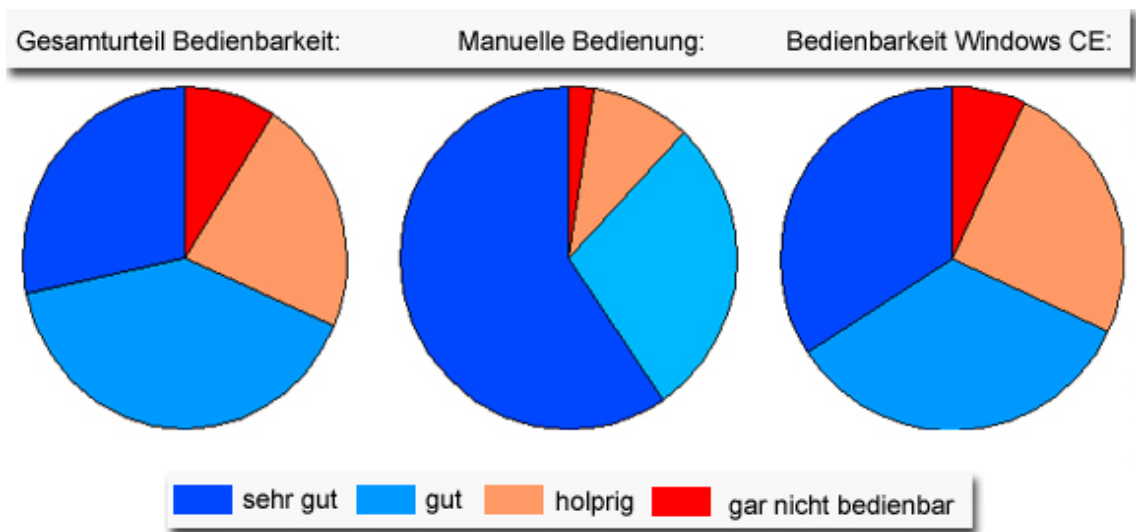


Abbildung 4.2: Ergebnisüberblick

nicht möglich sein, befanden 80 % der Testpersonen. Die DAB-Applikation zeigte sich wenig selbsterklärend. Da sich kaum einer der Probanden mit Digital-Audio-Broadcast-Technologie auskannte, war das Erfüllen dieser einfachen Aufgabe für die meisten Probanden frustrierend. Bemängelt wurde vor allem die fehlende Beschriftung. Das allgemeine Ergebnis lautet, dass sich der PDA mit den bestehenden Applikationen und Eingabemethoden auf keinen Fall für die Fahrt eignet. Im Umgang mit kritischen Situationen wurden unterschiedliche Verhaltensweisen beobachtet. Die eine Gruppe studierte ruhig und konzentriert die angebotenen Funktionen, um das Problem kognitiv zu lösen. Die andere exerzierte alle Möglichkeiten durch, davon ausgehend, dass die richtige Funktion dabei sein würde.

Eine Applikation mit wenigen großen Buttons pro Seite, die mit der Fingerspitze oder durch eine Fernbedienung am Lenkrad zu bedienen ist, konnten sich die Probanden als hilfreich während der Fahrt vorstellen.

4.4.3 Das Fazit der Evaluationsstudie

Die Auswertung der Fragen nach den Funktionen, die während der Fahrt ermöglicht bzw. verboten werden sollten, ist in Abbildung 4.3 dargestellt. Nach den hier erlangten Ergebnissen ist es nicht nötig, sich an ein windowsbasiertes Design anzulehnen,

Auswertung der Aussagen

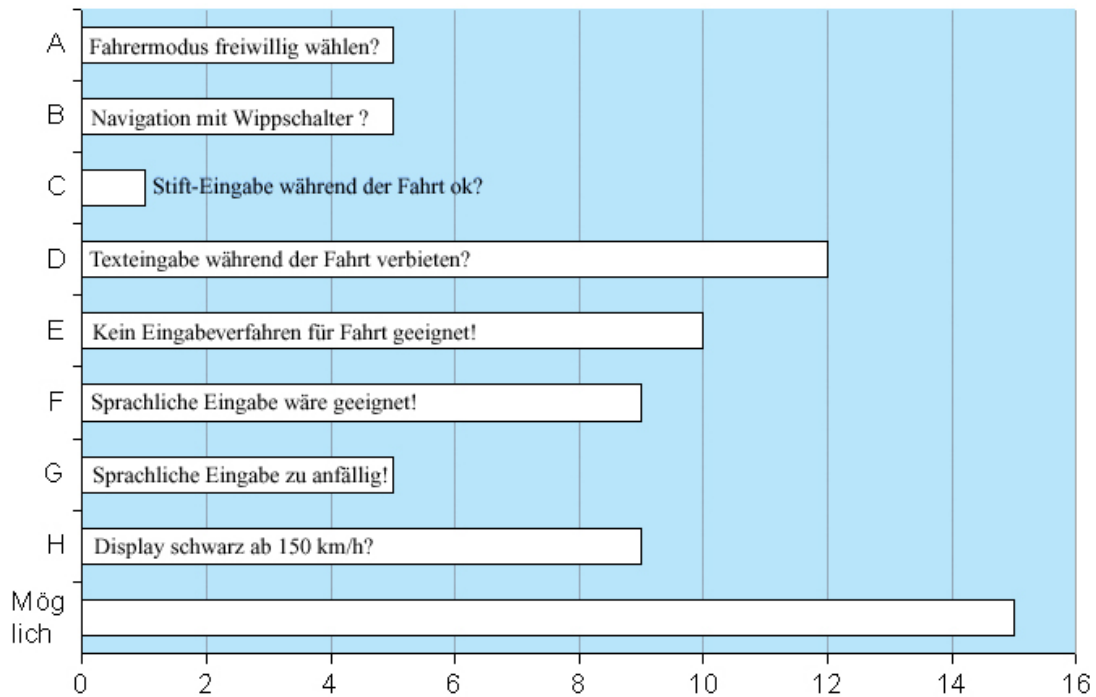


Abbildung 4.3: Übersicht der Aussagenbewertung

um intuitive Anwendung und hohen Wiedererkennungswert zu erreichen. Es sollte eher darauf geachtet werden, ein komplett neues Design zu entwickeln, um keine Erwartungen (positiv wie negativ) zu wecken, die nicht erfüllt werden. Die für die meisten Testpersonen neue, manuelle Eingabe mit dem Stift wurde rasch und ohne Probleme angenommen. Daraus folgt, dass eine einfache, eingängliche Eingabemethode schnell Akzeptanz findet, wenn die funktionale Umsetzung selbsterklärend und intuitiv ist.

Der hohe Anteil fehlsichtiger Anwender fordert eine vereinfachte Darstellungsweise der Elemente. Davon ausgehend, dass nicht jede Fehlsichtigkeit zu 100 % ausgeglichen ist, sollten die Menüpunkte ohne Fokussierung unterscheidbar sein. Da Menschen Farben schneller wahrnehmen können als Buchstaben zu lesen, wurde ein Farbkonzept zur leichteren Wiedererkennung entwickelt. Die vorgestellten Applikationen während der Fahrt zu nutzen, konnten sich die Probanden nicht vorstellen. Zu

viele Möglichkeiten, unübersichtliche Anordnung, mangelhafte Funktionalität oder eine schlechte Visualisierung machen das Nutzen der vorhandenen Applikationen während der Fahrt zu einer Gefahr für den Fahrer und alle anderen Verkehrsteilnehmer.

5 Entwicklung

Der ideale Computer ist noch zu entwickeln. Man gibt seine Probleme ein und sie kommen nie wieder heraus.

– Autor unbekannt –

5.1 Ziele der Entwicklung

Ziel der Entwicklung war es, eine für den PDA optimierte, effiziente Anwendung zu erstellen. Als Entwicklungsumgebung wurde das Microsoft-Visual-Studio-.Net gewählt.

In diesem Kapitel werden der Begriff .NET und das zugrunde liegende .NET Framework erläutert. Anschließend findet sich ein Vergleich der unterstützenden Funktionalitäten des .NET Framework mit denen des .NET Compact Framework, das eine Laufzeitumgebung für mobile Endgeräte zur Verfügung stellt. Im Abschnitt 5.4 werden die Programmstruktur und der Entwicklungsablauf des ersten Prototypen der SIRENA-MMS erläutert.

5.2 Laufzeitumgebung

Die Laufzeitumgebung der Oberfläche ist ein PDA mit Windows-CE-Betriebssystem. Das Windows-CE-Betriebssystem unterstützt das .Net Compact Framework. Auf dessen Basis erstellte Anwendungen sind durch Kopieren der Anwendungsdatei auf den PDA ausführbar. Es ist keine Installation notwendig.

- Display: Transfektiv-TFT-Farbdisplay, 240 x 320 Pixel, 6 x 8 cm groß
- Fingerabdrucklesegerät: Thermische Abtastungstechnik
- Speicher: 128 MB SDRAM, 48 MB ROM
- Betriebssystem: Windows Pocket PC 3.0
- .NET Compact Framework

5.3 Die Entwicklungsumgebung – das .NET Framework

Die Wahl der Programmiersprache zur Entwicklung der Benutzeroberfläche fiel auf C#. Diese noch junge, objektorientierte Programmiersprache wurde von Microsoft in Kooperation mit Anders Hejlsberg, dem Designer der Sprache Delphi, entwickelt. Verbunden mit dem .NET Framework (gesprochen: dot net) werden Flexibilität und Effizienz versprochen. Dieses Kapitel gibt einen Einblick in das dahinter stehende Konzept. Mit dem .NET Compact Framework stellt Microsoft eine Untermenge des .NET Frameworks als Laufzeitumgebung für Pocket-PC-Anwendungen zur Verfügung. Das .NET Framework ist eine komplexe, aber einfach anzuwendende Umgebung zum Programmieren, Kompilieren, Ausführen und Verteilen von Anwendungen.

5.3.1 C#

Dem Entwickler, der sich bereits mit Java und C-Programmierung befasst hat, wird ein Einstieg in C# (gesprochen: c sharp) relativ leicht fallen. Man kann es tatsächlich als Sprache, die aus den besten Eigenschaften von Java, Visual Basic und C++ zusammengesetzt ist, bezeichnen. Die Syntax ist ähnlich flexibel wie die von C++. Es lässt sich modular programmieren, doch um C# typischer zu machen, sind die Zeiger hier nicht übernommen worden. Es finden sich aber Techniken, wie man ähnlich vorgehen kann. Aus Java kennt man den Garbage-Collector, die Konstruktoren und das

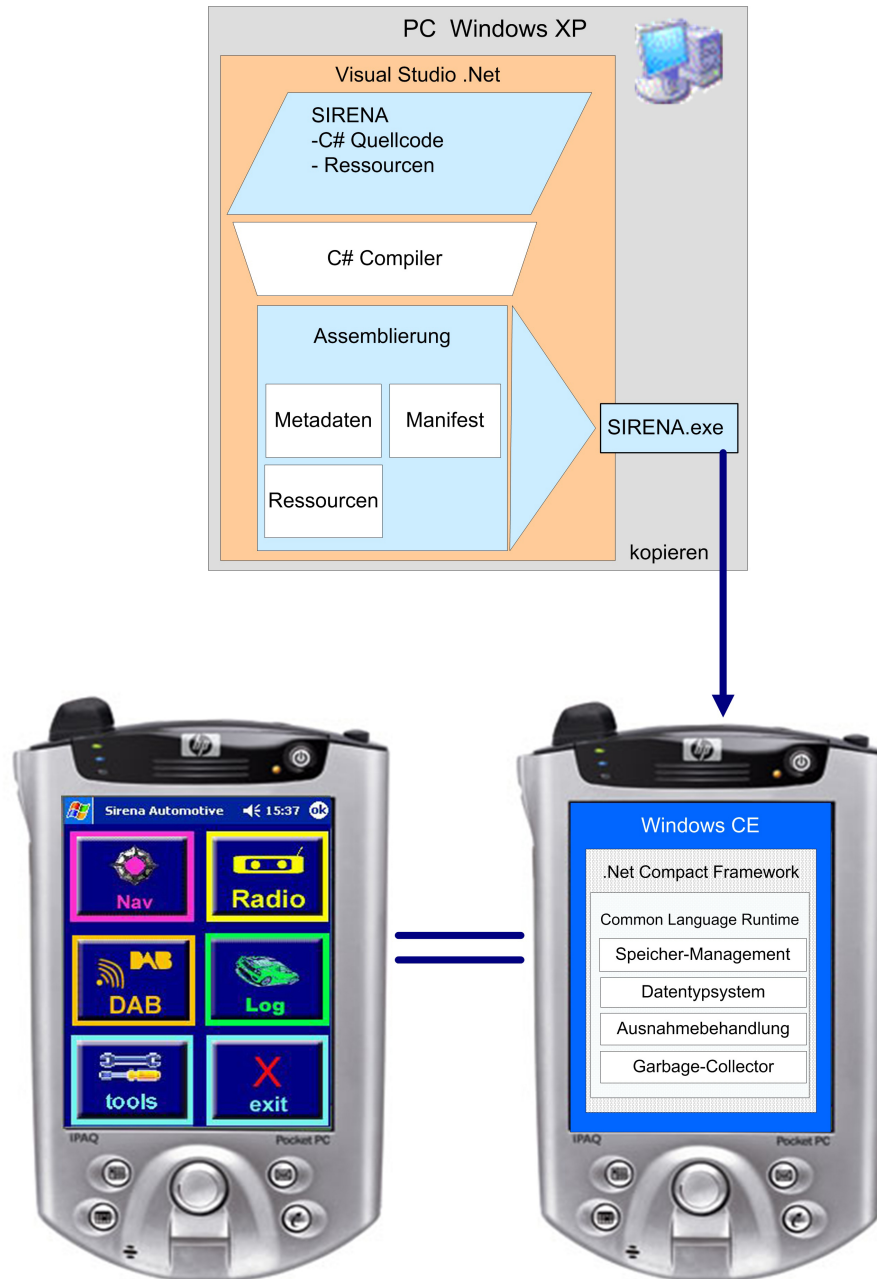


Abbildung 5.1: Funktion des .NET Framework am Beispiel SIRENA-MMS

Event-Handling. Die Größe der Klassenbibliotheken ist ähnlich umfangreich, doch transparenter durch seine baumartige Struktur. Die Wurzel dieser Namespaces ist *System*. Aus Visual Basic schließlich sind die RAD¹-Features bekannt. Damit wird auch dem Programmieranfänger über die ersten Hürden hinweggeholfen. Allerdings ist die große Ähnlichkeit zu anderen Sprachen insofern trügerisch, als dass C# seine ganz eigenen syntaktischen Eigenheiten, beispielsweise bei Funktionsaufrufen und Methodendeklaration, besitzt. Gerade was den stark abgespeckten Funktionsumfang des Compact Frameworks betrifft, ist die mächtige MSDN-Hilfe² zu komplex. Technische Artikel mit Beispielapplikationen zu bekannten Problemen erleichtern den Einstieg in die Syntax von C#. Im Internet hat sich bereits eine C#-Gemeinde gebildet, die Foren betreibt, in denen viele Fragen geklärt werden können.

5.3.2 C# und seine Bedeutung im .NET Framework

Am 14. Februar 2002 wurde C# von Microsoft im Rahmen des .NET Framework vorgestellt. Als gleichwertige Sprache steht C# neben Visual Basic, C++ und Visual J++ .NET im Visual Studio .NET zur Verfügung. Doch warum wurde mit C# eine neue Programmiersprache entwickelt, wenn sie tatsächlich äquivalent zu den anderen ist? Warum wurde das .NET Framework größtenteils in C# geschrieben und nicht mit C oder C++, wie die Windows-API? Ein guter Grund für die Entwicklung von C# war sicherlich, die positiven Eigenschaften aller Programmiersprachen zu vereinen. Dass Microsoft mit C# eine Konkurrenz zu Java von Sun-Microsystems auf den Markt bringen wollte, ist wohl auch nicht ganz auszuschließen.

Ein weiteres Argument ist allerdings, dass C# die erste von Microsoft entwickelte Programmiersprache ist, die zur Standardisierung eingereicht wurde. Im Dezember 2001 wurden C# und ein Teil von .NET (die Common Language Infrastructure) von der European Computer Manufacturers Association (ECMA) zum Industriestandard erhoben. Seit April 2003 sind C# und die CLI auch ISO-Standards. Die Beschreibung der Programmiersprache C# ist in dem Dokument ISO/IEC 23270 veröffentlicht, die der Laufzeitumgebung CLI in ISO/IEC 23271. Auf Basis dieser Standards ermöglicht Microsoft es jedem, einen C#-Compiler oder eine .NET-Implementierung zu entwickeln, denn die Beschreibungen sind (gegen Gebühr) öffentlich zugänglich.

¹Rapid Application Development = Schnelle Anwendungsentwicklung

²MSDN: Microsoft-Developer-Network, die Hilfedatei des Visual Studio

Microsoft geht damit einen wichtigen Schritt in Richtung Plattformunabhängigkeit. So wurde im Juni 2004 Mono (siehe Abschnitt 5.3.6) veröffentlicht, eine .NET-Implementierung für Linux. Die Tatsache, dass das .NET Framework zu großen Teilen in C# geschrieben ist, könnte bedeuten, dass Microsoft selber mit Effizienzverbesserung rechnet.

5.3.3 Was ist .NET?

Auseinandersetzung mit einem schwammigen Begriff

Seit etwa zwei Jahren taucht der .NET-Begriff in der IT-Branche immer häufiger auf. Doch trotz seiner Bekanntheit ist es schwer diesen Begriff zu definieren. Das liegt daran, dass die Microsoft-.NET-Initiative aus vielen Bereichen besteht und die allgemeine Vorstellung davon eine Mischung aus Realität, Vision und Marketing ist. Abbildung 5.2 zeigt eine Auflistung der einzelnen Bereiche. Im Folgenden sei mit *.NET* die .NET-Laufzeitumgebung, auf der C# aufsetzt, gemeint.

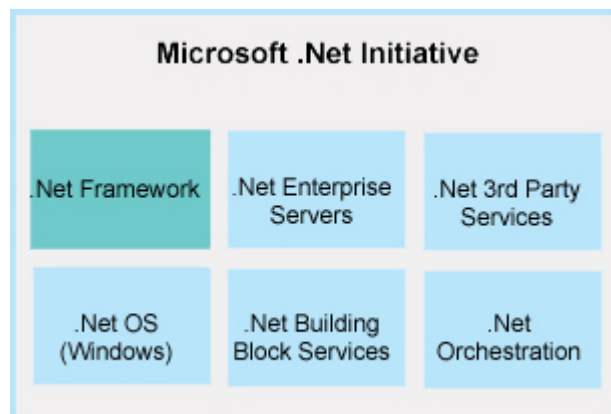


Abbildung 5.2: Die Bereiche der .NET-Initiative

5.3.4 Das .NET Framework und seine Komponenten

Das .NET Framework besteht aus verschiedenen Bereichen (z. B.: ADO.NET, ASP.NET, GDI+), die von jeder .NET-Programmiersprache verwendet werden können. Zur Entwicklung von Anwendungen dieser Bereiche steht eine umfangreiche Klassenbibliothek zur Verfügung. Das Zusammenspiel der Komponenten beschreibt

Abbildung 5.3. Die gewählte .NET-Programmiersprache greift auf die benötigten Klassen und Komponenten zu. Der jeweilige Compiler übersetzt den Code in einen Zwischencode, den MSIL-Code (Microsoft Intermediate Language), vergleichbar mit dem Java-Bytecode. Dieser ist unabhängig von der Programmiersprache, so dass Assemblies unterschiedlicher Programmiersprachen Teile des selben Programms sein können. Die Common Language Runtime übersetzt den Zwischencode zur Laufzeit in einen maschinenlesbaren Code.

Eine Liste der Features:

- Intermediate Language (IL) oder Zwischencode
Die Compiler der jeweiligen .NET-Sprachen übersetzen den Code in einen sprachunabhängigen Zwischencode. Dieser ist plattformunabhängig und wird von der jeweiligen Common Language Runtime (CLR) ausgeführt.
- Just-In-Time-Compiler (JIT)
Der JIT-Compiler des .NET Frameworks ist in die CLR integriert und kompiliert die Anwendung zur Laufzeit, also wenn der IL-Code ausgeführt wird. Erst dann entsteht prozessorlesbarer, OS-konformer Maschinencode. Die Effizienz wird durch Übersetzung nur der tatsächlich benötigten Programmmodule und intelligente Pufferung erhöht.
- Garbage-Collection (GC)
Garbage-Collection heißt übersetzt: Müllabfuhr. Dies ist ein bereits von Java und LISP bekannter Mechanismus, welcher dafür sorgt, dass Arbeitsspeicher wieder freigegeben wird, der von Objektinstanzen belegt wird, die nicht mehr referenziert sind. Man hat allerdings keinen Einfluss auf den Zeitpunkt, zu dem das Objekt aus dem Speicher entfernt wird.
- Manifest
Die Ausführungseinheiten werden durch den Compiler in sogenannte Assemblies übersetzt. Diese Assemblies enthalten in .NET selbstschreibende Metadaten. Im Manifest werden zum Beispiel Abhängigkeiten verwaltet, Versionsnummern definiert und (System-) Zugriffsrechte beschrieben, die für die Ausführung der Assemblierung benötigt werden. Die Assemblierung beschreibt sich damit vollständig selbst, Registry-Einträge entfallen.

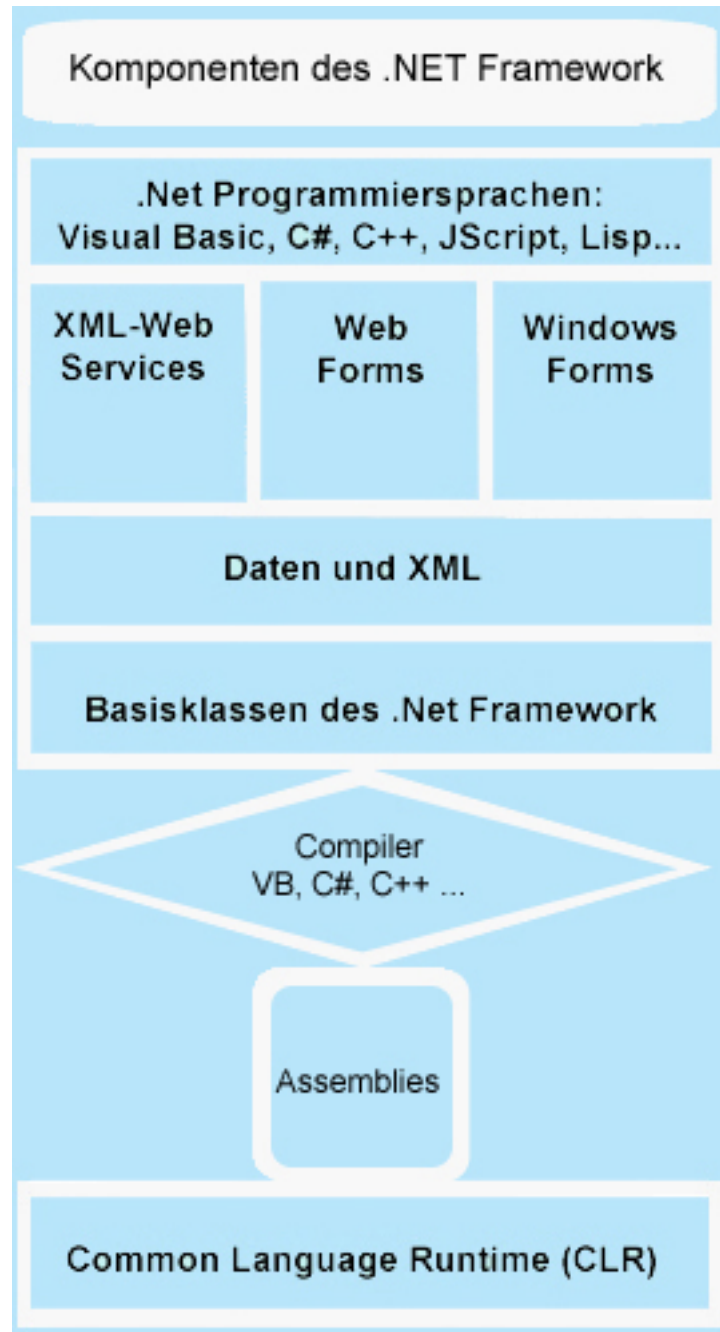


Abbildung 5.3: Komponenten des .NET Framework

- Code Access

Durchgängige Codesicherheit wird durch die Übersetzung in die Intermediate Language und die im Manifest enthaltenen Metadaten gewährt. Die .NET-Laufzeitumgebung kann dadurch beim Laden einer Anwendung feststellen, welche Aktionen ausgeführt werden dürfen.

- Reflection

Über ein Reflection-Interface erhält eine Anwendung Zugriff auf sämtliche Informationen zu ihren eigenen Datentypen. Mit den entsprechenden Sicherheitseinstellungen können auch Informationen aus Fremdklassen ausgelesen werden. Reflection ermöglicht außerdem, Typen zur Laufzeit dynamisch zu erzeugen.

5.3.4.1 Common Language Infrastructure (CLI)

Zur Common Language Infrastructure (CLI) (ECMA und ISO standardisiert) gehören u. a. das Common Type System (CTS) und die Common Language Spezifikation (CLS). Das CTS legt fest, welche Datentypen in einer Umgebung zur Verfügung stehen und wie diese Typen aufgebaut sind. Das bezieht sich nicht nur auf primitive Datentypen, sondern auch auf objektorientierte Typen, wie Klassen oder Schnittstellen. Grundsätzlich existieren hier nur Werttypen und Referenztypen, alle anderen sind spezielle Ableitungen.

In der CLS werden die Minimalanforderungen an die .NET-Programmiersprachen festgelegt. Dies gewährleistet Interoperabilität und Sprachunabhängigkeit. Das Kompilieren von Code in einer .NET-Programmiersprache erzeugt eine Datei, die den Zwischencode und Metadaten enthält. Durch die CLS ist es möglich, eine Anwendung in mehreren Programmiersprachen zu entwickeln, die dann mittels der Common Runtime Language zu maschinenlesbarem Code transformiert wird.

5.3.4.2 Common Language Runtime (CLR)

Die Common Language Runtime ist die Laufzeitumgebung der .NET-Programme. Sie umfasst:

- Ausführung des IL-Codes

- Sicherheitsüberprüfung
- Ausnahmebehandlung
- Aufruf von Methoden in Objekten
- Garbage-Collector
- Common Type System (CTS)

Mit diesem Framework ist es möglich, eine Anwendung in verschiedenen Programmiersprachen zu entwickeln und – sofern das .Net Framework installiert ist – plattformunabhängig zu implementieren.

5.3.5 Das .NET Compact Framework

Des Framework kleiner Bruder

Für Windows-Betriebssysteme auf Mobil- und Embedded-Geräten ist das .NET Framework als Compact-Version erhältlich. Es unterscheidet sich von dem für PCs verfügbaren Framework vor allem im Umfang der unterstützten Funktionalitäten. Das Compact Framework ist also weniger eine spezielle Ausarbeitung für die Belange von mobilen Endgeräten, als vielmehr eine Untermenge des .NET Framework. Es werden allerdings zusätzlich die speziellen Funktionen der Compact-Geräte unterstützt. Das sind:

- IrDA-Unterstützung
- SQL Server CE Klassen
- Gerätespezifische Controls (z. B. Input-Panel für Stifteingaben)

Weiterhin unterscheiden sich beide Frameworks in ihrer Größe und der Skalierungscharakteristik der Anwendungen. Das .NET Compact Framework unterstützt neben den Basisdatentypen auch die Basisklassen der IO-, Collections-, Reflection- und Drawing-Namensräume. Die Klassen-Bibliotheken des Compact Framework umfassen allerdings nur 25 % der auf Microsofts .NET Framework für PCs verfügbaren Menge. Das Compact Framework bietet Unterstützung für Netzwerkverbindungen, HTTP-Klassen und Webservices sowie den Datenzugriff mittels ADO.NET

auf SQL Server CE und SQL Server. Auch wird XML-Unterstützung geboten, diese beschränkt sich allerdings auf Klassen zum Lesen und Schreiben von Daten im XML-Format (z. B. XmlDocument, XmlNode). Die XML-Serialisierung des Desktop-Framework und Remoting gehören beispielsweise nicht dazu. Windows-Forms zur Gestaltung der Benutzeroberfläche werden unterstützt, dazu zählt das Layout, Drawing (Polygone, Linien, Ellipsen, Rechtecke, auch GIF-, JPEG- und BMP-Bilder), Text und die meisten Desktop-Controls. Insgesamt bilden aber auch sie auf dem .NET Compact Framework nur eine Untermenge der auf dem Desktop einsetzbaren Windows-Forms und lassen viele davon vermissen.

5.3.6 Mono – ein Stück Unabhängigkeit

Bei Mono 1.0 handelt es sich um eine Open-Source-Implementierung des Microsoft .NET-Frameworks. Bisher war die Entwicklung und Ausführung dieser Programme nur unter Windows möglich. Mit Mono ermöglicht die Novell-Tochter Ximian dies auch für alle anderen Betriebssysteme. Derzeit funktioniert das unter Unix-Derivaten wie Linux, Solaris und Mac OS X sowie unter Windows 2000 und XP. Mono 1.0 implementiert den vollen Umfang der ECMA-standardisierten Laufzeitumgebung CLI und enthält einen optimierenden Just-in-Time- sowie einen Ahead-of-Time-Compiler für den Zwischencode Intermediate Language (IL). Auch der mitgelieferte C#-Compiler entspricht dem ECMA-Sprachstandard. Bei den höheren APIs fährt Mono immer noch zweigleisig: Microsoft-kompatibel sind derzeit die Web-Anwendungen (ASP.NET) und der Datenbankzugriff (ADO.NET). Für GUI-Programme gibt es zunächst nur ein Mono-eigenes API namens Gtk# und eine Anbindung an die Vektorgrafik-Bibliothek Cairo. Windows-Forms werden noch nicht unterstützt. Eine Anwendung der hier entwickelten Oberfläche ist vorerst also nur auf Windows-Plattformen möglich.

5.4 Programmierung

5.4.1 Zielsetzung

Die zu entwickelnde Benutzeroberfläche soll im wesentlichen aus einer Hauptmenüseite mit Buttons zu den darunter liegenden spezifischeren Seiten bzw. Schnittstellen zu externen Diensten bestehen. Ausgeführt wurde dieses Konzept mit dem Designer des Microsoft-Visual-Studio .NET unter Verwendung der Windows-Forms.

5.4.2 Die Gestaltungskomponenten

Die Hauptkomponenten der SIRENA-MMS sind Windows-Form-Elemente und Buttons zur Navigation. Die Hauptmenüseite besteht aus einem Fenster mit sechs gleichgroßen Buttons. Die oberen vier dieser Buttons führen zu den in Abbildung 5.4 dargestellten Funktionen. Das Logbuch ist eine Implementierung des Fraunhofer Instituts FIRST. Hier werden Fahrzeugdaten für die Verwaltung und Kontrolle der Fahrzeugnutzung verwaltet. Zur Interaktion relevant ist die Eingabemöglichkeit des Zwecks der Fahrt. Alle anderen Daten werden automatisch erfasst. Diese Funktion wird nicht ausführbar sein, wenn das Auto fährt. Der Menüpunkt Logbuch bietet außerdem die Optionen, die Daten auf dem PDA mit der AWebS-Box oder dem PC zu synchronisieren. Die Funktionsweise des Navigators ist noch unbekannt. Die Simulation der Oberfläche in der Testanwendung bot die Möglichkeit zur Eingabe des Ziels (nur im Nichtfahrermodus) und die Optionen, die aktuelle Route aufzurufen. Der Menüeintrag *AWebS* führt zu den Web-Services, die Routenabhängige Gefahrenzonen etc. anzeigen. Die Radiolautstärke kann im Untermenü *Radio* per Scrollbalken geregelt werden. Die Auswahl der Radiosender erfolgt durch ein Scrollfeld. Die Digital-Audio-Broadcast-Technologie unterstützt folgende Dienste:

- *Web-Dienst*, hier werden die Internetseiten des gewählten Senders gezeigt,
- *Bilder*, die manche Sender anbieten,
- *Text-Info* zeigt Nachrichten in großer Schrift an,
- unter *News* werden Nachrichten vorgelesen.

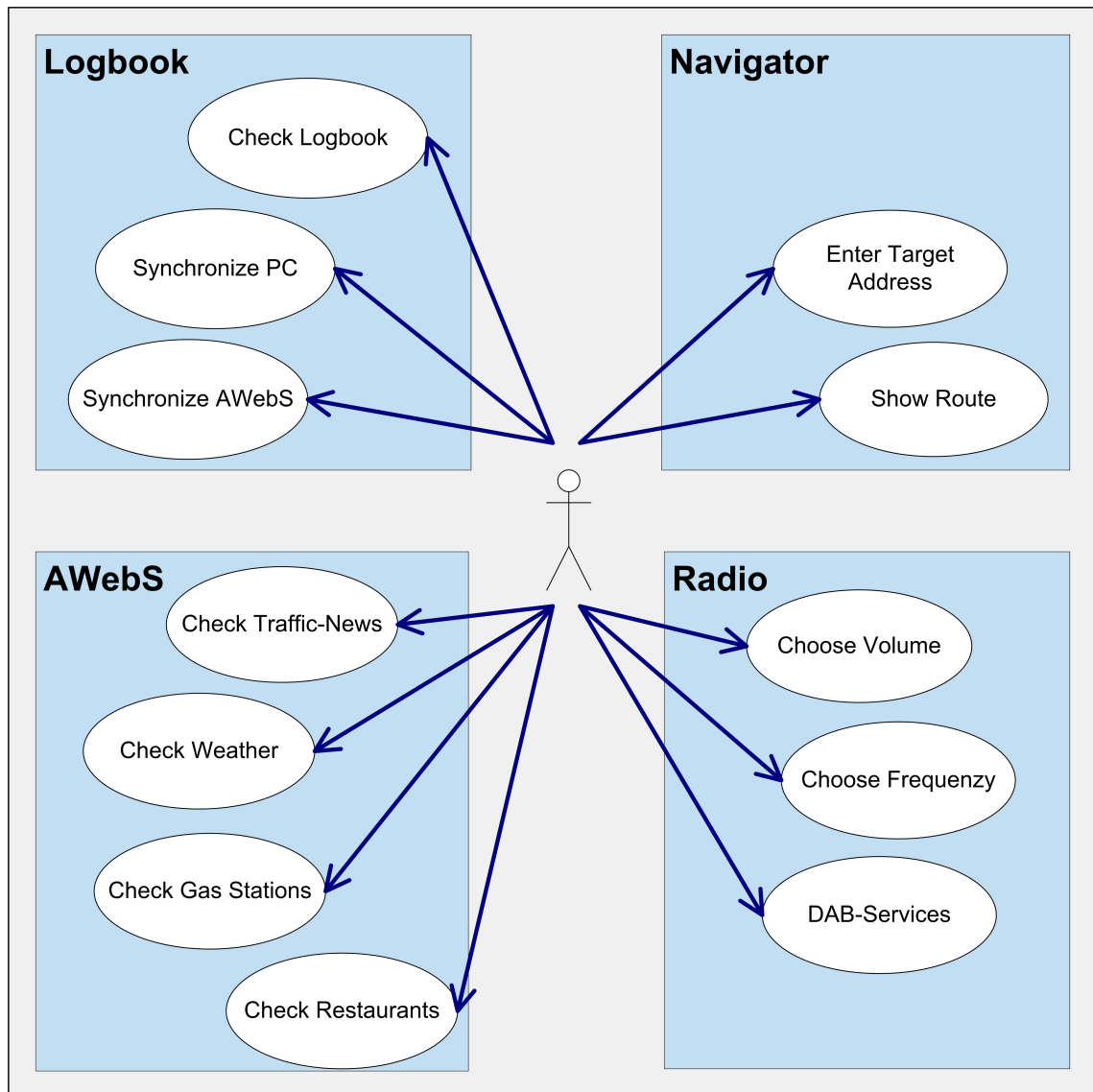


Abbildung 5.4: Usa-Case Diagramm des Hauptmenüs



Abbildung 5.5: DAB-Services

Der Menüumfang scheint zwar gering, trotzdem wurden für die Testimplementierung der SIERENA-Benutzeroberfläche 30 Form-Elemente und ca. 80 unterschiedliche Buttons verwendet.

5.4.3 Benutzerverwaltung – Modi

Die Implementierung eines Tag- und Nachtmodus wurde durch weitgehenden Verzicht weißer Elemente vermieden. Wichtig für die Sicherheit im Straßenverkehr sind ein Fahr- und Nichtfahrmodus. Diese wurden für den Test simuliert. Fenster, die verkehrsgefährdende Aktionen anbieten, haben die boolsche Variable *Fahrt*. Die nicht verfügbaren Funktionen werden in der Testversion der SIRENA-MMS normal ange-

zeigt. Es erscheint jedoch bei *Click* eine Fehlermeldung mit dem Hinweis, dass diese Funktionen nur bei haltendem Fahrzeug möglich sind. Diese Variante wurde für die Usability-Tests zur Simulation von Fehlermeldungen implementiert.

Später wird diese Variable *Fahrt* abhängig von der Fahrgeschwindigkeit sein. Ist das Tempo höher als 0, schaltet sich der Fahrermodus ein.

Die Buttons zu sicherheitsgefährdenden Funktionen werden im Fahrermodus in grau gehalten. Bei *Click* ertönt ein Signalton.

5.4.4 Navigation

Die Programmoberfläche hat einen übersichtlichen Aufbau. Baumartig strukturiert zweigen von dem Form-Element *Sirena1*, dem Hauptmenüfenster, alle weiteren Formen ab. Das Visual-Studio unterstützt dabei die Aktionen *hide* und *close*, um die Fenster zu verwalten, während sie nicht gebraucht werden. Schließt man jede Form mit *close*, wenn eine andere aufgerufen wird, ist das ressourcensparend, hat aber Wartezeiten durch erneutes Laden beim nächsten Gebrauch zur Folge. Das kann durch den Gebrauch von *hide* vermieden werden. Das Fenster wird einfach hinter dem aktuellen versteckt und offen gehalten. Dieses Vorgehen bei einer Anwendung mit 30 unterschiedlichen Form-Elementen blockiert den Speicher unnötig und erschwert die Aktualisierung von Daten. Effizienter ist es, eine einmal geladene Seite bis zum Wiederaufruf auf einem Stack abzulegen.

Bei Microsoft ist dieses Problem bekannt. In der MSDN-Library werden Beispielapplikationen mit verschiedenen Lösungsansätzen vorgestellt. Daraus entnommen sind die Klassen *StackForm* und *FormStack*. Diese ermöglichen es, eine flexible Navigation aufzubauen. Der Stack wurde mit einem *Array* realisiert. C# ermöglicht einen sehr flexiblen Umgang mit den in einem Array gespeicherten Elementen. In der Klasse *FormStack* wird der eigentliche Stack initiiert. Der Stack arbeitet, solange Forms enthalten sind. Dabei wird beim Aufruf einer Form geprüft, ob sie bereits im Stack ist. Ist sie da, wird sie gezeigt. Ist sie nicht da, wird sie geladen und gezeigt.

Von der Klasse *StackForm* erben alle Form-Elemente. Das ermöglicht ein konsistentes Framework, auf dessen Elemente dynamisch zugegriffen werden kann. Dabei unterstützt die Klasse *StackForm* die Form-Elemente mit einem Event-Handler, der das Closing-Event abfängt und die Form statt dessen auf den Stack schiebt. Ein

Load-Data-Worker-Thread und eine populate-Funktion unterstützen das Aktualisieren von Daten zur Laufzeit.

5.4.5 Farbige Buttons

Für jede Seite ist eine Form zu erstellen und mit Buttons oder entsprechend anderen Elementen zu füllen. Problematisch ist für diese Anwendung, dass das Compact Framework keine Möglichkeit bietet, die Buttons farbige oder mit Hilfe von Bildern zu gestalten. Gerade auf farbigen Buttons baut jedoch das Wahrnehmungskonzept der SIRENA-Oberfläche auf. Auch zu diesem Problem gibt es in den Technischen Artikeln der MSDN-Library Lösungen mit einigen Syntax-Beispielen. Mit der Möglichkeit, eine Grafik in Button-Design mit den Funktionen eines Buttons auszustatten, bietet die Klasse *ImageButton* benötigte Flexibilität für das SIRENA-HMI.

Das Click-Event eines Buttons zum Beispiel invertiert den Wert der booleschen Variable *bPushed*. Ist *bPushed* wahr, so wird das erste Pixel der Grafik gelesen und für alle Pixel dieser Farbe wird eine neue Farbe gesetzt. Verbunden mit dem Click-Event wird die Grafik, in Form eines Buttons, zu einem gedrückten Button. Auf diese Weise sind alle Buttons der SIRENA-MMS gestaltet.

5.4.6 Ausführung

Das Microsoft-Visual-Studio bietet durch RDA-Unterstützung eine komfortable Programmierumgebung. Die Klassen zur Navigation und zur Darstellung der farbigen Buttons bilden die Basis der Anwendung. Die einzelnen Fenster werden zuerst im Designer des Visual-Studio mit Grundkomponenten, wie Hintergrundfarbe, Toolbox etc., ausgestattet. Die von *ImageButton* abgeleiteten Grafiken werden durch den Designer nicht dargestellt. Im Gegenteil, wird die Klasse nach Implementierung der Image-Buttons noch einmal mit dem Designer geöffnet, versucht dieser, die Image-Buttons als .NET-spezifische Buttons zu implementieren und ändert den Quellcode. Wird das Fenster schließlich von *StackForm* abgeleitet, ist gar keine Designunterstützung mehr möglich. Es empfiehlt sich also, zuerst die Designkomponenten zu implementieren und die Form anschließend von *StackForm* abzuleiten.



Abbildung 5.6: SIRENA-MMS mit gedrücktem Button

Das Visual-Studio bietet eine Emulator-Funktion, die einen PDA mit Windows-CE-Betriebssystem simuliert und es so ermöglicht, die entwickelte Anwendung am PC zu testen (siehe Abbildung 5.7). Allerdings hat der Emulator nur eine beschränkte Speicherkapazität von 15 Mb RAM.

Die so entstandene Anwendung wird zu einer .exe-Datei kompiliert. Sämtliche Grafiken, Audio-Dateien und weitere Komponenten sind als eingebettete Ressourcen in der Datei enthalten.

5.4.7 Testphase

Um die Oberfläche zu testen, wurden die zukünftigen Dienste simuliert. Gespeicherte Audio-Dateien haben Nachrichten wiedergegeben, Bilder aus DAB-Diensten

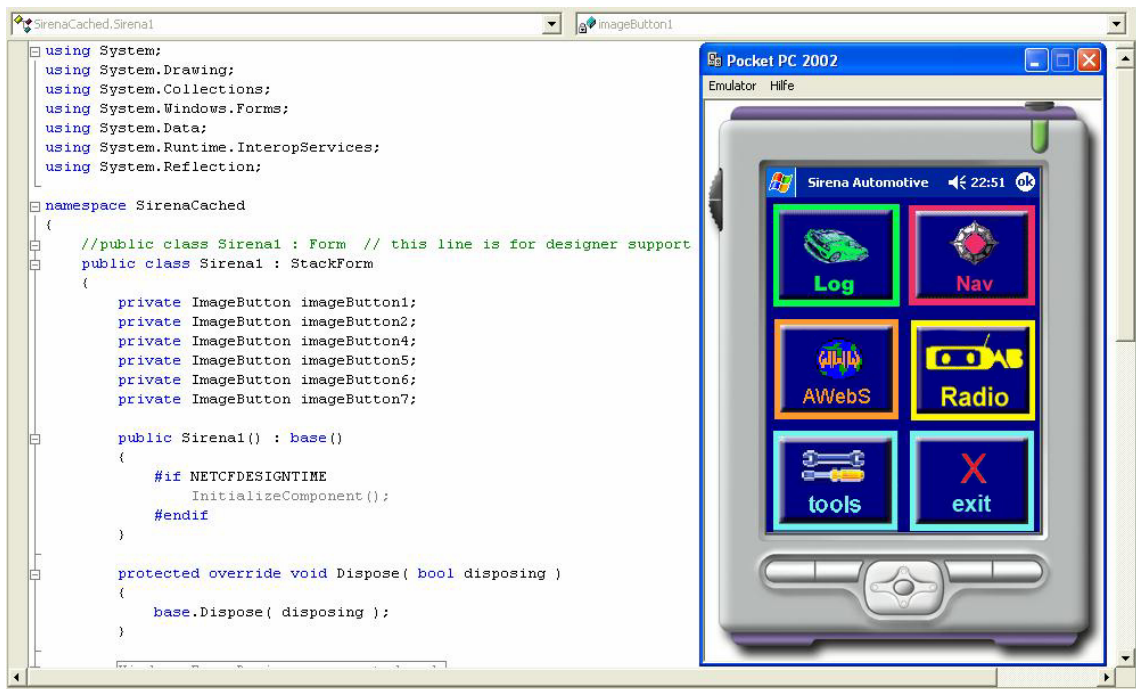


Abbildung 5.7: Entwicklungsumgebung mit Emulator

wurden eingebunden, Screen-Shots aus dem Computerspiel zur Fahrsimulation ergaben Stadtpläne, die Tankstellen und Fast-Food-Ketten anzeigen. Das Ergebnis war eine eigenständige Anwendung zur Darstellung eines konsistenten *Wizard-of-Oz*-Testszenarios.

5.5 Das Fazit der Entwicklung

Die SIRENA-Benutzeroberfläche wurde in Form eines testfähigen Prototypen erstellt. Dabei hat sich das Visual-Studio .Net mit C# als Programmiersprache als komfortable und unterstützende Entwicklungsumgebung gezeigt. Mit diesem ersten Prototyp der SIRENA-MMS ist eine effiziente, flexibel erweiterbare Anwendung entstanden. Damit werden optimale Voraussetzungen für die Implementierung der Hard- und Softwarekomponenten des Automotive-Demonstrator geboten.

6 Usability-Tests

Objektivität ist die Illusion, dass Beobachtungen ohne einen Beobachter gemacht werden können.

— Ernst von Glasersfeld —

6.1 Ziele der Usability-Tests

Die Sichtweise des Entwicklers weicht immer von der des Benutzers ab. Auch wenn Entwickler eines Systems hinterher durchaus Anwender desselben sein können, ist es nötig, zu überprüfen, ob das entwickelte System tatsächlich den Anforderungen und dem Verständnis des Benutzers entspricht. Ziel des Tests war es herauszufinden, inwieweit die Ansprüche von Autofahrern an eine PDA-Benutzeroberfläche, die sie während der Fahrt unterstützen soll, erfüllt werden konnten und wo Optimierungsbedarf besteht.

6.2 Die Methode

Die zu implementierenden Funktionen waren zur Zeit der Tests nicht verfügbar. Deshalb wurde ein sogenanntes *Wizard-of-Oz*-Testszenario entwickelt, das die zukünftigen Funktionen simuliert. Ein Betreuer hat während der praktischen Testphase die Aktionen der Probanden überwacht und entsprechend reagiert. Obwohl die Probanden vorher darüber informiert wurden, sind viele davon ausgegangen, dass tatsächlich das System reagiert.

6.2.1 Theorie

Die hier vorgestellte, heuristische Studie ist nicht repräsentativ, da die Gruppe der Probanden zufällig zusammengestellt wurde. Bedingung zur Teilnahme war der Besitz eines Führerscheins.

Zur Entwicklung des zu testenden Systems wurden im Vorfeld die Anforderungen der zukünftigen Nutzer eruiert. Anschliessend wurden diese unter den Aspekten der menschlichen Wahrnehmung und Informationsverarbeitung in die Gestaltung der SIRENA-Oberfläche integriert. Der vorliegende Prototyp der SIRENA-Benutzeroberfläche wurde hinsichtlich der Qualität folgender Eigenschaften untersucht:

- **Leichte Erlernbarkeit**
- **Effizienz**
- **Leichtes Wiedererkennen**
- **Fehlervermeidung / einfache Fehlerbehandlung**
- **Zufriedenheit**
- **Anforderungen an die Aufmerksamkeit**
- **Sicherheit**

Des Weiteren wurde die Möglichkeit wahrgenommen an einem online-Testverfahren teilzunehmen. Unter der Adresse:

<http://www.attrakdiff.de>

gelangt man zu einem kostenlosen (wissenschaftlich fundierten) Service, der interaktive Produkte auf Bedienbarkeit und Attraktivität prüft. Näheres zu Attrakdiff ab Seite 85.

6.2.2 Angewandte Methode

Diese erste Prüfung des Prototyps bedurfte einer möglichst realitätsnahen Testszene. Auf die (Autofahr-)Realität mußte wegen des Sicherheitsrisikos verzichtet werden. Um die Fahrsituation zu simulieren, wurde das Computerspiel *Midnight-Club*

eingesetzt. Dieses für Rennspiele konzipierte Spiel simuliert unter anderem auch bevölkerte Städte. Bedient mit Lenkrad und Pedalen (Simulation von Gaspedal und eines Automatikgetriebes), wurde das Spiel auf eine 2 m x 2,5 m grosse Leinwand projiziert, die im Abstand von 2 m zum Probanden stand. Ein Sessel vermittelte die richtige Sitzhöhe. Der PDA wurde auf einem Tisch montiert, an dem auch Lenkrad und ein Autoradio befestigt waren und die visuelle Testsituation abrundeten (siehe Abbildung 6.1).

6.3 Versuchsanordnung

6.3.1 Übungsteil

Die Versuche fanden unter Einzelbetreuung statt und dauerten jeweils etwa eine Stunde. Zuerst wurde jeder Proband in das Spiel eingewiesen und bekam die Möglichkeit es in Ruhe zu testen. Anschließend erfolgte die Erläuterung der zu beurteilenden Anwendung auf dem PDA. Danach wurde den Probanden Zeit gegeben, das System eigenständig zu erkunden und ein Gefühl dafür zu entwickeln. Im Testverlauf mußte ein Tempo von 50 km/h eingehalten und auf rote Ampeln geachtet werden. Zuerst wurde eine Testfahrt ausgeführt, in der der PDA nicht genutzt wurde. In der nächsten Fahrt wurde eine Reihe von Aufgaben gestellt:

1. Zieladresse und Zweck der Fahrt im Navigator eintragen
2. Fahrzeug starten – damit war der Fahrermodus aktiviert
3. Verkehrsnachrichten anhören
4. Tankstelle suchen und anfahren
5. Aktuelles Wetter anzeigen lassen
6. Radiosender einstellen
7. Fahrt beenden (Fahrermodus deaktivieren) und Fahrtenbuch kontrollieren



Abbildung 6.1: Usability-Test-Szenario

In einem Beobachtungsbogen wurde während beider Fahrten das Verhalten der Testpersonen bezüglich:

- Tempowechsel
- Spurwechsel
- Überfahrene rote Ampeln
- Häufigkeit verursachter Unfälle

dokumentiert.

6.3.2 Untersuchung

Nach Abschluss des praktischen Teiles bekamen die Probanden einen Arbeitsplatz zugewiesen, der mit 2 PDA's (einmal mit der SIRENA-Oberfläche im Fahrermodus, der andere im Nichtfahrermodus), je einem Fragebogen mit 24 Fragen und einem PC ausgestattet war, um an der online-Befragung teilzunehmen. Außerdem fand sich ein Informationsblatt, auf dem die Benutzeroberfläche und die simulierten Dienste noch einmal beschrieben waren.

Der Fragebogen enthielt 24 Fragen, 19 davon vorgegebene Aussagen, die durch ein Itemschema von 1 - 6 beurteilt werden sollten.

In den Fragen 20 - 23 wurden die Testteilnehmer aufgefordert, die eigene Meinung zu äußern. Es war nach guten wie schlechten Eigenschaften der SIRENA-Oberfläche sowie Optimierungsmöglichkeiten gefragt. In Frage 24 wurden statistische Angaben über Alter, Geschlecht, Ausbildung und Produkterfahrung ermittelt. In den Fragen 1 - 19 wurde zudem darauf geachtet, dass sich die Fragen nicht mit denen aus dem online-Angebot von Attrakdiff überschneiden.

6.4 Ergebnisse

6.4.1 Allgemeines

Die Testgruppe setzte sich wie in der Evaluationsstudie, aus 20 % Frauen und 80 % Männern zusammen, was dem PDA-Benutzerprofil entspricht. Beruflich waren Ingenieure, Studenten und Sekretärinnen vertreten. Auch hier war bei knapp 50 % der Probanden die Sehkraft eingeschränkt. Die meisten waren kurzsichtig, Farbfehlsicht kam nicht vor. Diese Ergebnisse decken sich mit denen aus der Anforderungsanalyse. Die Konsequenz des großen Anteils fehlsichtiger Erwachsener war, ein konsistentes Farbschema in die Menüführung zu implementieren. Ein Benutzen der Oberfläche aus dem Augenwinkel heraus wird dadurch forciert und unterstützt.

6.4.2 Spezifische Ergebnisse

Die Ergebnisse waren durchweg zufrieden stellend. Die Oberfläche wurde sehr positiv angenommen, dennoch wurden Optimierungsvorschläge gemacht, die im weiteren Projektverlauf implementiert werden können. Die Diagramme 1 - 5 in Abbildung 6.2, also Einfachheit des Erlernens, Effizienz, einfaches Wiedererkennen, Fehlervermeidung, einfache Fehlerbehebung und außerdem die Zufriedenheit des Nutzers, treffen Aussagen über die allgemeingültigen Kriterien eines gelungenen Mensch-Maschinen-Interfaces. Die Diagramme 6 und 7, Anforderungen an die Aufmerksamkeit des Fahrers/Benutzers und Sicherheit, gehen auf spezielle Anforderungen dieser Oberfläche ein.

6.4.3 Auswertung

Alle Ergebnisse befinden sich im überdurchschnittlich guten Bereich. Sehr positiv wurde sich zur leichten Erlernbarkeit geäußert. Große Buttons in übersichtlicher Anzahl mit konsistenter Platzierung lassen sich ebenso leicht bedienen wie erlernen.

Auch leichtes Wiedererkennen und Effizienz der SIRENA-Benutzeroberfläche wurden als überdurchschnittlich gut bewertet. Das bedeutet, dass die Oberfläche ihrem Zweck entsprechend gut funktioniert und eine hohe Nutzerakzeptanz erlangt hat.

6 Usability-Tests

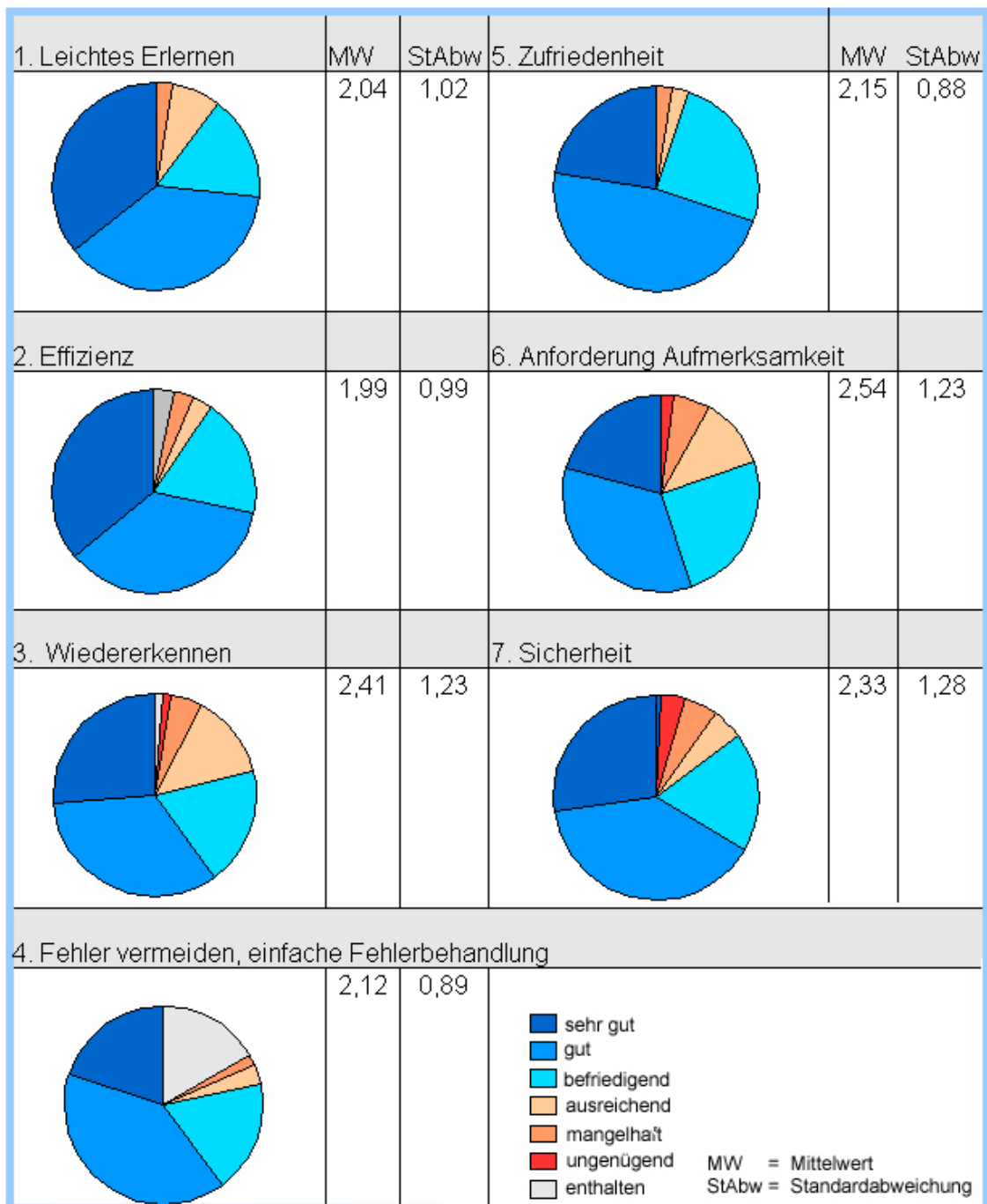


Abbildung 6.2: Auswertung Usability-Test der SIRENA-MMS

Die Untersuchung hinsichtlich des Optimierungspotenzials hat sich bei einigen Testpersonen die Meinung ergeben, dass die Platzierung der vier Hauptmenüpunkte auf der Eingangsseite einen Konflikt zwischen Anwender und Projektdarstellung darstellt. Die AwebS-Services werden nicht so häufig gebraucht wie die DAB-Dienste, die erst im Untermenü *Radio* zu erreichen sind. Durch eine Favoritenseite oder eine dynamische Hauptmenüseite wären sie schneller erreichbar. Eine sich ständig ändernde Hauptmenüseite wird jedoch eher zu Verwirrungen führen. Der Anwender soll Konsistenz erwarten können.

Ein weiterer Kritikpunkt war, dass sich tiefer liegende Funktionen nach einiger Zeit schließen sollten, um zum Hauptmenü zurück zu kehren. Auch selbstständiges Schließen von Untermenüs nach einer gewissen Zeit kann gerade im Straßenverkehr irritieren, wenn nach einer Aktion, die ungeteilte Aufmerksamkeit benötigt, wieder von vorn angefangen werden muss. Sinnvoll ist, unter dem Menüpunkt *Einstellungen* eine Favoritenseite, ebenso wie alternative *Skins*, individuell einstellbar zur Verfügung zu stellen.

Die Hilfefunktion wurde von keiner Testperson benutzt. Die Frage nach Fehlern und deren Behandlung konnten einige Testpersonen nicht beantworten, weil ihnen nichts aufgefallen war. Ansonsten wurden die konkreten Anweisungen der Fehlermeldungen sehr positiv angenommen. Ein Beispiel dazu zeigt Abbildung 6.3.

In Fragen der Sicherheit zeigen die Ergebnisse ebenfalls eine gute Bewertung. Die Aufzeichnungen im Beobachtungsbogen belegen jedoch Verbesserungsbedarf. Die geforderte Aufmerksamkeit führte bei einigen Probanden zu häufigerem Spurwechsel als bei der Fahrt ohne die Bedienung des PDA. Die Ursache hierfür ist in der geteilten Aufmerksamkeit der Testpersonen zu finden. Die Testumgebung, das Computerspiel und die zu testende Oberfläche waren neue Komponenten, die jeweils vollen Umfang an Aufmerksamkeit des Probanden gefordert haben. Vergleichbar ist die Situation mit der Bedienung des Autoradios in einem Leihwagen, unterwegs in einer fremden Stadt.

In der ersten Phase des Kennenlernens muss die Bedienung erlernt werden. Nach einer Zeit der Eingewöhnung sollte die Bedienung *blind* erfolgen können. Dieses Konzept kann nur in einer Langzeitstudie überprüft werden. Dabei werden vergleichende Beobachtungen des Fahrverhaltens in allen Phasen des Kennenlernens genauere



Abbildung 6.3: Fehleranzeige der SIRENA-MMS

Aussagen über die geforderte Aufmerksamkeit der SIRENA-MMS ermitteln.

Die elektronische und digitale Aufrüstung von Fahrzeugen ist noch lange nicht abgeschlossen. Zukünftig werden immer mehr Geräte um die Aufmerksamkeit des Fahrers buhlen. Die Entwicklung der SIRENA-Benutzeroberfläche, die alle im Fahrzeug erreichbaren Geräte zentriert steuert, ist die richtige Richtung, um sicherzustellen, dass die Aufmerksamkeit des Fahrers vom Wesentlichen, dem Verkehr, nicht unnötig abgelenkt wird.

6.5 Die Untersuchungsmethode von Attrakdiff

Attrakdiff ist ein Instrument zur Messung der Attraktivität von interaktiven Produkten [uMHUD].

Bisherige Evaluationsverfahren beschränken sich im wesentlichen auf Bedienbarkeit (pragmatische Qualität). Entwickelt wird ein Evaluationsverfahren, mit dem sowohl die wahrgenommene pragmatische, als auch die hedonische Qualität und die Attraktivität eines interaktiven Produktes erfasst werden kann. Durch eine breite Anwendung des Untersuchungsinstruments sollen weitere Erkenntnisse zur Qualität von Gütekriterien für Evaluationsverfahren und eine Weiterentwicklung des Untersuchungsinstruments erreicht werden.

AttrakDiff soll als Forschungsinstrument eingesetzt werden, z. B. um zu ermitteln, welche gestalterischen Maßnahmen zu einer verbesserten hedonischen Qualität führen. Weiter soll eine Diskussion darüber angestoßen werden, ob die Usability-Kriterien *Effektivität*, *Effizienz* und *Zufriedenstellung* wirklich ausreichend zur Bewertung von Produkten sein können, oder ob die hier beschriebenen Kriterien ebenso eine wichtige Rolle spielen.

AttrakDiff und die AttrakDiff-Website werden als Gemeinschaftsprojekt der User Interface Design GmbH und Marc Hassenzahl (Universität Darmstadt) entwickelt und bereitgestellt. [uMHUD]

6.5.0.1 Konfidenzintervall

Bei einer Untersuchung können niemals alle Personen, die ein Produkt nutzen, zur Beurteilung herangezogen werden. Bei der Auswahl der Testpersonen kann man nie 100 % sicher sein, dass die ausgewählten Personen für die Gesamtheit der Nutzer des Produkts repräsentativ sind. Es könnte also sein, dass sich die Beurteilung der Testpersonen, von der unterscheidet, die man erhalten würde, wenn man alle Nutzer befragen könnte. Das Konfidenzintervall gibt einen statistisch ermittelten Bereich an, in dem der *wahre* Wert liegt, den man erhalten würde, wenn alle Nutzer befragt werden könnten.

6.5.0.2 Die Testmethode

Die Testpersonen geben mit Hilfe gegensätzlicher Adjektivpaare an, wie das Produkt wahrgenommen wird. Ein Beispiel:

technisch	O O O O O O O	menschlich (PQ)
konventionell	O O O O O O O	originell (HQ-Identität)
lahm	O O O O O O O	fesselnd (HQ-Stimulation)
abstoßend	O O O O O O O	anziehend (Attraktivität)

Diese Adjektivpaare lassen sich den untersuchten Beurteilungsdimensionen zuordnen. Untersucht werden damit folgende Dimensionen:

1. Pragmatische Qualität (PQ)
Prüft die Benutzbarkeit eines Produkts
2. Hedonische Qualität - Stimulation (HQ-S)
Prüft, inwieweit das Produkt die persönliche Weiterentwicklung des Anwenders unterstützt
3. Hedonische Qualität - Identität (HQ-I)
Prüft, inwieweit sich der Nutzer mit dem Produkt identifizieren kann
4. Attraktivität (ATT)
Beschreibt die wahrgenommene Qualität

Diese Untersuchung ist produktunabhängig auf jedes interaktive System anwendbar. Dem Untersuchungsleiter steht lediglich die Möglichkeit zur Verfügung, Produkt- und Firmennamen anzugeben und die farbliche Gestaltung der Untersuchungsbögen anzupassen. Außerdem werden der Zeitraum der Untersuchung und die Art (Einzel- oder Vorher-Nacher-Untersuchung) angegeben. Die Anzahl der Teilnehmer ist auf 20 Personen begrenzt.



Abbildung 6.4: Ergebnisse Attrakdiff

6.5.1 Ergebnisüberblick

Die Portfolio-Darstellung ist in mehrere Quadranten unterteilt, anhand derer der Charakter der Anwendung abgelesen werden kann. Es gibt folgende Quadranten: überflüssig, zu selbstorientiert, zu handlungsorientiert, neutral, selbstorientiert, handlungsorientiert, begehrt. Je nach Ausprägung der Dimensionen hedonische Qualität und pragmatische Qualität, fällt das Produkt in einen oder mehrere Charakterbereiche. Je kleiner das Konfidenz-Rechteck, desto sicherer kann das Produkt einem bestimmten Ergebnisbereich zugeordnet werden.

Die pragmatische Dimension erreichte für die SIRENA-MMS mittlere Werte. Das bedeutet, die SIRENA-Oberfläche wurde als eher handlungsorientiert eingestuft. Es besteht also noch Verbesserungspotenzial hinsichtlich der Bedienbarkeit. Auch bei der hedonischen Qualität wurden mittlere Werte erreicht. Der Nutzer wird durch die Oberfläche zwar angeregt, jedoch lautet auch hier das Ergebnis: Es besteht noch Verbesserungspotenzial hinsichtlich der hedonischen Aspekte.

Das Konfidenz-Rechteck ist klein, das lässt erkennen, dass sich die Probanden in

ihrer Beurteilung einig waren. Das Konfidenz-Rechteck hat eine eher quadratische Form, die Probanden waren sich bei der Beurteilung der Dimensionen gleich einig.

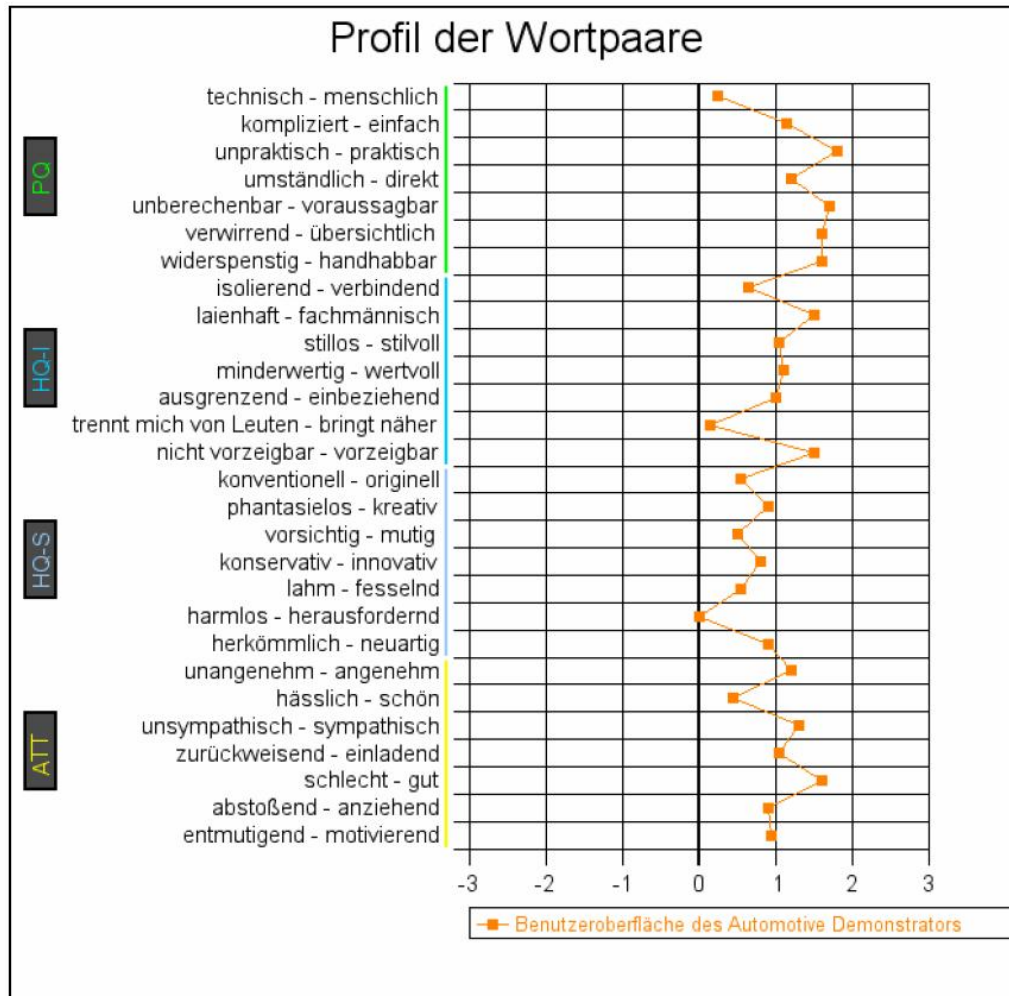


Abbildung 6.5: Das Profil der Wortpaare

Im Profil der Wortpaare sind die mittleren Ausprägungen der einzelnen Wortpaare in Bezug auf die SIRENA-Benutzeroberfläche dargestellt. Hier sind vor allem die Extremwerte interessant. Sie zeigen, welche Punkte besonders kritisch oder besonders gut gelöst sind. In der Beurteilung der SIRENA-MMS ist *harmlos* das einzige Beurteilungs-Item, das in den Bereich der kritischen Bewertung fällt. Die besonders positiv bewerteten Eigenschaften sind *praktisch*, *voraussagbar* und *gut*. Zusammenfassend wurde die SIRENA-MMS als gut voraussagbar, praktisch und gleichzeitig wenig herausfordernd bewertet.

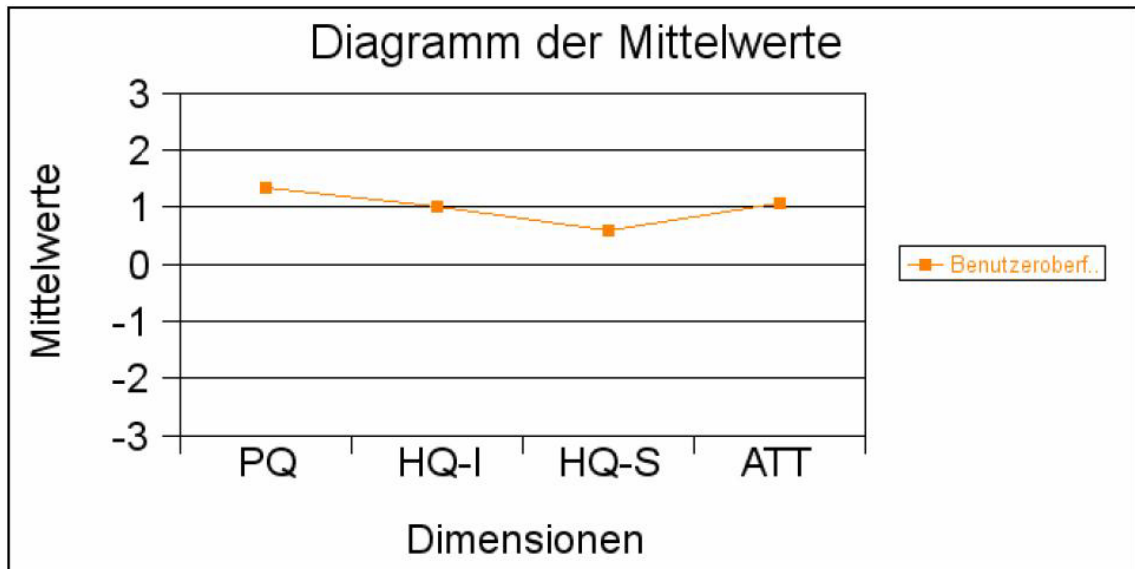


Abbildung 6.6: Abbildung der Mittelwerte

Das Diagramm der Mittelwerte stellt die mittlere Ausprägung der Dimensionen dar. Hier wird die hedonische Qualität zusätzlich nach den Gesichtspunkten Stimulation und Identität differenziert. Zusätzlich wird die Beurteilung der wahrgenommenen Attraktivität dargestellt.

Die Ausprägungen der hedonischen Qualität in den Dimensionen Identität und Stimulation befinden sich im durchschnittlichen Bereich. Um die Nutzer stärker an das Produkt zu binden, sie stärker zu motivieren und zu stimulieren, sollte auch hier eine Verbesserung angestrebt werden. Die Bewertung der wahrgenommenen Attraktivität und der pragmatischen Qualität befindet sich im überdurchschnittlichen Bereich. Das Produkt wird als attraktiv und effizient wahrgenommen. Verbesserungen sind hier nicht zwingend notwendig.

6.6 Das Fazit der Usability-Tests

Vergleichend wird festgestellt, dass sich die Ergebnisse von Attrakdiff mit den eigenen verifizieren, wobei die eigenen Untersuchungen produktspezifischere Ergebnisse liefern, die produktiver in die Weiterentwicklung einfließen können.

Die Tests fanden in der Phase des ersten Prototyps statt. Die hier erzielten Er-

gebnisse zeigen, dass die Oberfläche leicht zu verstehen und zu bedienen ist. Die Menüführung mit großen Buttons in übersichtlicher Anordnung und dem Bedienen mit dem Zeigefinger ist der richtige Ansatz für eine intuitive und eher beiläufige Nutzung während der Fahrt. Die gewünschte Optimierung des Nutzerkomforts kann mit der Implementierung der tatsächlichen Funktionen erfolgen. Vor Freigabe des endgültigen Produkts wird eine Langzeitstudie über mindestens eine Woche mit täglich etwa einer halben Stunde Übung je Proband empfohlen.

7 Ergebnis

7.1 Fazit der Diplomarbeit

Mit der SIRENA-MMS wurde ein Prototyp geschaffen, der das SIRENA-Framework in einem benutzeroptimierten Rahmen präsentiert. Die effiziente Programmierung unterstützt durch vordefinierte Threadfunktionen, die erforderliche Skalierbarkeit zur Implementierung der Demonstrator-Funktionen. Das auf Basis der Anforderungsanalyse erstellte Designkonzept hat bei den zukünftigen Anwendern hohe Akzeptanz erlangt. Damit sind optimale Voraussetzungen für die Realisierung der SIRENA-Automotive-Demonstrator-Mensch-Maschinen-Schnittstelle geschaffen worden.

7.2 SIRENA Projektaussichten

Das Projekt SIRENA hat die Aufgabe übernommen, eine zukunftsweisende Technologie zu entwickeln. Das Zusammenspiel von UPnP/DPWS und OSGI in einem Service-Framework ermöglicht ein ganz neues Niveau von Service-Anwendungen. Zur Zeit in der Entwicklung befindet sich die Protokollstruktur auf Basis von DPWS mit IPv6-Adressvergabe. Hier wird ein weiteres Treffen der Projektmitglieder entscheiden, in welcher Form OSGI implementiert werden kann. Auch die Sicherheitsaspekte und Authentifizierungsverfahren müssen noch ausgearbeitet werden. Alles in Allem kann mit einem erfolgreichen Projektabschluß im September 2005 gerechnet werden.

A Literaturverzeichnis

- [All] OSGi Alliance.
open service gateway initiative.
<http://www.osgi.org>.
- [Bay02] Juergen Bayer.
C# Nitty Gritty.
Addison-Wesley, 2002.
ISBN 3-8273-1856-4.
- [DB99] Dirk Roeckmann Dieter Bauer.
Methoden der Bewertung von Software.
Wirtschaftsverlag NW, 1999.
ISBN 3-89701-087-9.
- [Joh99] Steven Johnson.
Interface Culture.
Klett-Cotta, 1999.
ISBN 3-608-91980-5.
- [Joh00] Jef Johnson.
GUI Bloopers.
Academic Press, 2000.
ISBN 1-55860-582-7.
- [Kar02] Bernd Karstens.
Dialogsysteme und Software-Ergonomie, SE 4.

- , 2002.
<http://www.icg.informatik.uni-rostock.de/archiv/Semester/SS02/DSE>.
- [Luc04] Univ.-Prof. Dr.-Ing. Dipl.-Wirt.-Ing. Holger Luczak.
Menschliche Informationsverarbeitung.
Technische Hochschule Aachen Arbeitswissenschaften 2, 2004.
<http://www.iaw.rwth-aachen.de>.
- [Nie93] Jakob Nielsen.
Usability Engineering.
Academic Press, 1993.
ISBN 0-12-518406-9.
- [pro04] projectmono.
mono .net framework, 2004.
<http://www.mono-project.com>.
- [Ras01] Jef Raskin.
Das intelligente Interface.
Addison-Wesley, 2001.
ISBN 3-8273-1796-7.
- [Rub94] Jeffrey Rubin.
Handbook of Usability Testing.
John Wiley & Sons Ltd, 1994.
ISBN 0-471-59403-2.
- [Shn02] Ben Shneiderman.
User Interface Design (Deutsche Ausgabe).
MITP-Verlag Bonn, 2002.
ISBN 3-8266-0753-8.
- [Shn03] Ben Shneiderman.
Leonardo's Laptop.
MIT-Press, 2003.
ISBN 0-262-19476-7.

- [SIR] SIRENA.
homepage des sirena-projekts.
<http://www.sirena-itea.org>.
- [Soe03] Dirk Soeffker.
*Systemorientierte Modellbildung der wissensgeleiteten
Mensch-Maschine-Interaktion.*
Logos Verlag Berlin, 2003.
ISBN 3-8325-0251-1.
- [Spo02] Helma Spona.
C# – Der leichte Einstieg.
Markt + Technik Verlag, 2002.
ISBN 3-8272-6399-9.
- [uMHUD] User Interface Design GmbH und Marc Hassenzahl (Uni. Darmstadt).
attrakdiff.
<http://www.attrakdiff.de>.
- [US00] Dieter Werner Uwe Schneider.
Taschenbuch der Informatik.
Fachbuchverlag Leipzig, 2000.
ISBN 3-446-21331-7.
- [Wei02] Scott Weiss.
handheld usability.
John Wiley & Sons Ltd, 2002.
ISBN 0-470-84446-9.

Abbildungsverzeichnis

2.1	Mitglieder des SIRENA-Projekts	12
2.2	SIRENA-Framework Übersicht	14
2.3	SIRENA-Projektplan	15
2.4	SIRENA-Konzept	16
2.5	SIRENA-Protokollstack	19
2.6	OSGI-Architektur	21
2.7	SIRENA-Cross-Domain-Demonstrator	25
2.8	Automotive-Demonstrator-Szenario	26
2.9	Statusinformationen der A-Klasse	28
2.10	Digital-Audio-Broadcast	29
3.1	Schwellen menschlicher Wahrnehmung	36
3.2	Vertikales und horizontales Sehfeld des Menschen	38
3.3	Stufen der Verarbeitung vom Reiz bis zu seiner Wahrnehmung	39
3.4	Trotz der Anweisung zu zählen, lässt sich das Lesen der Items nicht verhindern.	40
3.5	Beispiele zu den Gestaltgesetzen	41
4.1	Antwortitems in der Evaluationsstudie	53
4.2	Ergebnisüberblick	56
4.3	Übersicht der Aussagenbewertung	57
5.1	Funktion des .NET Framework am Beispiel SIRENA-MMS	61
5.2	Die Bereiche der .NET-Initiative	63
5.3	Komponenten des .NET Framework	65

Abbildungsverzeichnis

5.4	Usa-Case Diagramm des Hauptmenüs	70
5.5	DAB-Services	71
5.6	SIRENA-MMS mit gedrücktem Button	74
5.7	Entwicklungsumgebung mit Emulator	75
6.1	Usability-Test-Szenario	79
6.2	Auswertung Usability-Test der SIRENA-MMS	82
6.3	Fehleranzeige der SIRENA-MMS	84
6.4	Ergebnisse Attrakdiff	87
6.5	Das Profil der Wortpaare	88
6.6	Abbildung der Mittelwerte	89
D.1	Testbegleitender Informationsbogen	111

B Glossar

Bluetooth

Bluetooth beschreibt einen Standard für die drahtlose Kommunikation unterschiedlichster Geräte wie Computer, Modem, Mobiltelefon und Drucker untereinander.

Bundle

Engl. für Bündel. Ein Bundle bezeichnet die Bündelung zweier Produkte zu einem Gesamtprodukt. So werden zum Beispiel Hardware-Komponenten mit spezieller Software, oder sich ergänzende Softwarepakete miteinander oder alte Versionen einer Software mit einer Update-Version gebündelt.

Bus

Eine Datenverbindung zwischen zwei oder mehreren Punkten. Die Geschwindigkeit eines Busses wird durch seine Breite (angegeben in Bit) und seine Frequenz (meist in MHz) bestimmt.

Cache

Als Cache wird ein schneller Zwischenspeicher bezeichnet, der häufig benutzte Daten puffert und dadurch den Zugriff darauf beschleunigt.

Controller Area Network (CAN)

Ein Protokoll, das auf dem Datenbus im Fahrzeug eingesetzt wird.

Digital Audio Broadcast (DAB)

DAB bezeichnet den neuen Rundfunkübertragungsstandard Digital-Radio. Vorteil dieser Übertragungsform ist störungsfreier Empfang in CD-Qualität. Durch den Empfang von Daten in digitaler Form ist es möglich, diese zu speichern und neben Audio-Daten noch weitere Informationen zu senden, die über ein Display anwählbar sind (z. B. Internetseiten, Verkehrsnachrichten).

ECMA – European Computer Manufacturers Association

Europäischer Industriestandard.

Echtzeit

Beschreibt die Fähigkeit eines Systems einen Sachverhalt unmittelbar zu verarbeiten.

Effizienz

Beschreibt die Kosten-Nutzen Relation eines Sachverhalts.

GPS

Abkürzung für Global Positioning System. Dabei handelt es sich um ein Satelliten-System, mit dem der Empfänger auf der Erde seine genaue Position ermitteln kann. Navigationssysteme für Autos basieren auf GPS in Verbindung mit einer Straßenkartensoftware.

GPRS – General Packet Radio Service

Daten werden mit Hilfe dieser Technologie wie im Internet in Paketen übertragen. GPRS-Handys sind immer online, empfangen die für den Teilnehmer bestimmten Daten aber nur von Zeit zu Zeit.

Intuition

Unmittelbares Erkennen, Erfassen ohne wissentliche Reflexion

PDA

Als Personal Digital Assisitent wird ein Computer in Taschenformat bezeichnet.

UMTS

Abkürzung für Universal Telecommunication System. UMTS ist ein Standard für Mobilfunknetze. In UMTS-Netzen können Datenübertragungsraten von bis zu 2 Mbit/s erreicht werden.

UPnP

Ein 1999 von Microsoft initiiertes Protokoll. Es spezifiziert die Bekanntgabe, die Suche und die Verwendung von Geräten bzw. Services. Dabei legt es sich weder auf eine Programmiersprache, noch auf eine Plattform, sondern nur auf die zu verwendenden Protokolle (SSDP, GENA, SOAP, HTTP, UDP, TCP, IP) fest.

Video on Demand

Übersetzt aus dem Englischen bedeutet es *Video auf Verlangen*. Gemeint sind online-Services, die Filme für eine bestimmte Zeit (meist 24 Stunden) freigeben. Dazu wird der Film auf die eigene Festplatte geladen. Nach Eingabe eines Schlüssels ist der Film für 24 h verfügbar. Danach kann ein neuer Schlüssel erworben werden.

Web Service (WS)

Ein Service, der für Clients über Web-basierte Technologien wie HTTP und SOAP erreichbar ist.

Wizard-of-Oz - Methode

Die Wizard-of-OZ-Methode wird angewandt, um nicht existente Systeme zu testen. Ein Mensch (Wizard) simuliert die Reaktionen des Systems und interagiert über die Benutzeroberfläche mit dem Anwender.

C Erhebungsbogen der Evaluationsstudie

Die Evaluationsstudie wurde am 16. Juni 2004 in den Räumen des Fraunhofer Instituts FIRST durchgeführt. Auf den nachfolgenden Seiten befinden sich die verwendeten Fragebögen.

Erhebungsbogen

für ein

PDA – Benutzerinterface

Der Elektronikmarkt bietet immer mehr Geräte an, die den Komfort des Fahrers während der Autofahrt erhöhen sollen. Um nun aber vom Fahrer kein Multitasking zu verlangen, ist es geplant, den PDA als eine Art Fernbedienung einzurichten. Vorhandene Geräte bilden dynamisch ein Netz, der PDA dient als Kontrollpunkt. Für diesen Kontrollpunkt wird die Benutzeroberfläche entwickelt. Sie, als zukünftiger Benutzer dieser Fernbedienung, werden hier um Ihre Meinung gebeten.



Bitte 'denken Sie laut' während des Tests, damit Ihre persönliche Meinung besser erfasst werden kann. Vielen Dank!

Der gesamte Test dauert nur 10-15 Minuten.

1. Aktivieren Sie bitte den PDA.

Wie hat das geklappt?

sehr gut	gut	holprig	gar nicht
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

2. Geben Sie das Kennwort ein: 1234

Wie hat das geklappt?

sehr gut	gut	holprig	gar nicht
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

3. Öffnen Sie den Kalender

Wie hat das geklappt?

sehr gut	gut	holprig	gar nicht
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

4. Tragen Sie nun den 23.06.04 als Abgabetermin dieser Befragung ein.

Wie hat das geklappt?

sehr gut	gut	holprig	gar nicht
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

5. Schliessen Sie bitte den Kalender und schalten den DAB - Dienst ein (Version 2003).

Wie hat das geklappt?

sehr gut	gut	holprig	gar nicht
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

6. Lesen Sie sich den Radiotext des Deutschlandfunks durch.

Wie hat das geklappt?

sehr gut	gut	holprig	gar nicht
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

7. Können Sie sich vorstellen, den DAB - Dienst während der Autofahrt, über den PDA als eine Art Fernbedienung, zu benutzen? (mit ein bisschen Übung)

sehr gut	gut	holprig	gar nicht
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

8. Öffnen Sie erneut den Kalender und löschen Sie den Abgabetermin wieder.

Wie hat das geklappt?

sehr gut	gut	holprig	gar nicht
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Schliessen Sie bitte alle Anwendungen.

➔ Damit ist der praktische Teil bereits geschafft!

9. Sie haben nun einen kurzen Einblick in die Menüführung des Microsoft Pocket PC Betriebssystems bekommen. Was ist Ihnen besonders aufgefallen, was war z.B. frustrierend, ärgerlich, besonders einfach oder sehr attraktiv?

10. Haben Sie die Hilfe benutzt?

ja / nein

Falls nein, warum nicht?

11. Lesen Sie sich bitte folgende Aussagen durch und kreuzen Sie alle an, denen Sie zustimmen.

ja

<input type="checkbox"/>	Wenn ich wählen könnte zwischen a) Fahrermodus, für die Fahrt optimiert aber im Funktionsumfang eingeschränkt b) Mitfahrermodus, der den vollen Funktionsumfang enthält würde ich mich für die Fahrt im Fahrermodus anmelden.
<input type="checkbox"/>	Mit dem Wippschalter am PDA komme ich prima zurecht. Damit kann ich selbst während der Fahrt problemlos durch ein übersichtliches Menü navigieren.
<input type="checkbox"/>	An die Eingabe mit dem Stift kann man sich so gut gewöhnen, dass er während der Fahrt problemlos zu benutzen wäre.
<input type="checkbox"/>	Eingabe von Schriftzeichen sollte während der Fahrt verboten sein.
<input type="checkbox"/>	Keines von den angebotenen Eingabemöglichkeiten ist geeignet, während der Fahrt benutzt zu werden.
<input type="checkbox"/>	Sprachliche Eingabe ist die einzige Methode, die die Fahrsicherheit nicht einschränkt.
<input type="checkbox"/>	Sprachliche Eingabe ist zu anfällig für Nebengeräusche und wird spätestens ärgerlich, wenn das Radio oder die Kinder im Hintergrund, laut sind.
<input type="checkbox"/>	Ab einem bestimmten Tempo (z.B. 150km/h) sollte das Display nur noch schwarz sein.

12. Abschliessend bitten wir Sie, einige Angaben zu sich und Ihrer Erfahrung mit dem Produkt zu machen.

Altersgruppe:	20-30	31-40	41-50	51-60	<u>Hinweis:</u> Diese Angaben werden nicht gespeichert und nicht an Dritte weitergegeben!
Geschlecht:	w / m				
Ausbildungsabschluß:					
Beruf:					
Fahren Sie öfter Auto?	ja / nein				
Haben Sie bereits Erfahrung im Umgang mit PDA's?	ja / nein				
Besteht eine Einschränkung Ihrer Sehkraft?	ja / nein				
Wenn ja, welche? Kurz- / Weit- / Farbfehlsicht	andere:				

Das war es schon, vielen Dank für Ihre Mühe!

D Erhebungsbogen des Usability-Tests

Die Usability-Tests wurden am 29. und 30. September 2004 in den Räumen des Fraunhofer Instituts FIRST durchgeführt. Auf den nachfolgenden Seiten befinden sich die verwendeten Fragebögen und ein Informationsblatt, das den Testpersonen zur Verfügung gestellt wurde.



Usabilitytest für ein PDA – Mensch - Maschinen – Interface

Die hier zu testende Benutzeroberfläche soll im Auto als Fernbedienung für die Steuerung verschiedener Geräte dienen, z.B.:

TomTom-Navigator

AwebS-Box

- anzeigen von Tankstellen und Fast-Food-Ketten u.v.m. in der Nähe

Radio

- Senderauswahl
- Lautstärke einstellen

Digital Audio Broadcast Dienste:

- anzeigen von Bildern und WEB-Seiten
- ansagen aktueller Verkehrsnachrichten

All die Dienste waren im Test nur simuliert. Zu beurteilen ist lediglich die Funktionalität von Menüführung und Design im Hinblick auf ihren Einsatz während der Autofahrt. Die Beantwortung der Fragen dauert nur ca 10 Minuten.

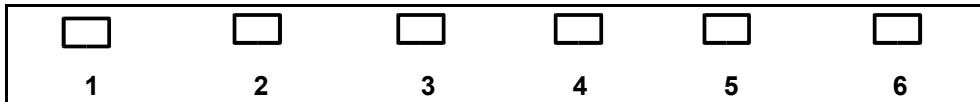
Dazu ist allerdings ein wenig Phantasie nötig. Stellen Sie sich vor, Sie wären für ein paar Tage in einer fremden Stadt, hätten einen Leihwagen und das getestete System wäre da, um Sie zu navigieren, zu unterhalten und zu informieren. Würden Sie sich damit anfreunden? Wie schnell kämen Sie gut mit dem PDA und der SIRENA – Oberfläche zurecht? Würden Sie schon bald, aus den Augenwinkeln heraus, allein an den Farbflächen auf dem PDA erkennen, in welchem Menüteil Sie sich befinden und 'blind' drauf los tippen können? Dann wäre die Oberfläche gelungen!

Die Bewertung bitte wie Schulnoten vergeben:

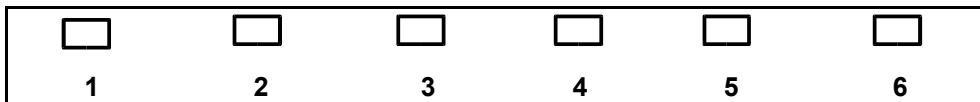
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
1	2	3	4	5	6

1 für 'sehr gut' bis 6 für 'ungenügend'

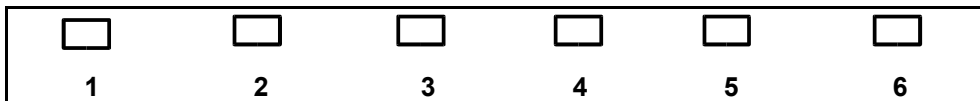
1. SIRENA ist auf dem Pocket PC leicht zu finden und zu öffnen:



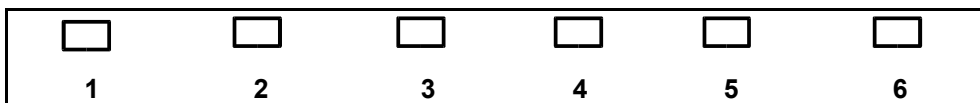
2. Die Bedienung der Oberfläche ist leicht erlernbar:



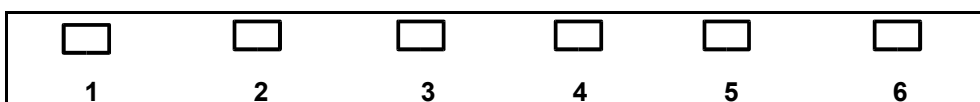
3. Design und Menüführung sind durchgängig konsistent:



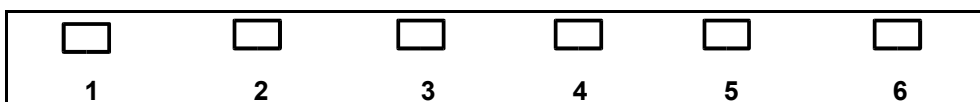
4. Die Fernbedienung ist nach kurzer Eingewöhnungszeit aus dem Augenwinkel heraus, mit einem Minimum an Aufmerksamkeit, bedienbar:



5. Die Oberfläche ist leichter zu bedienen als mein Autoradio:



6. Das Bedienen der Oberfläche erfordert weniger Aufmerksamkeit als die Bedienung meines Handys während der Fahrt:



7. Das Programm gibt nur die wirklich nötigen Arbeitsschritte vor:



8. Das Programm kann bestimmte Arbeitsschritte automatisch ausführen, um mich zu entlasten (z.B. Internetverbindung herstellen):

<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
1	2	3	4	5	6

9. Erläuterungen (Statusanzeigen, Fehlermeldungen) sind so zutreffend formuliert, dass sie in der jeweiligen Situation wirklich weiterhelfen:

<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
1	2	3	4	5	6

10. Die Hilfefunktion war notwendig:

<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
1	2	3	4	5	6

11. Die Wortwahl in den Rückmeldungen ist so gewählt, dass sie meinen Kenntnissen entspricht:

<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
1	2	3	4	5	6

12. Die Eingabemethode ist ideal für die Bedienung des PDA während der Fahrt:

<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
1	2	3	4	5	6

13. Die Geschwindigkeit mit der die Anwendung reagiert (1. Laden jeder Seite langsam, jedes weitere schnell) ist mir angenehm:

<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
1	2	3	4	5	6

14. Die Menge der angezeigten Daten ist an meine Anforderungen während der Fahrt angepasst:

<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
1	2	3	4	5	6

15. Das Programm gibt ausreichend Auskunft über besondere Systemzustände oder Verzögerungen:

<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
1	2	3	4	5	6

16. Die Fehlermeldungen geben mir Gründe an und erläutern mir Möglichkeiten, den Fehler zu korrigieren:

<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
1	2	3	4	5	6

17. Die gesteckten Ziele sind mit geringem Bedienungsaufwand zu erreichen:

<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
1	2	3	4	5	6

18. Die auf dem Bildschirm dargestellten Informationen sind für mich deutlich und schnell zu erkennen:

<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
1	2	3	4	5	6

19. Ich bewerte den Umgang mit dieser Software insgesamt als:

<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
1	2	3	4	5	6

20. Die Oberfläche während der Fahrt zu benutzen halte ich nicht für sinnvoll weil:

21. Folgende Eigenschaften der Oberfläche gefallen mir besonders gut:

22. Folgende Eigenschaften der Oberfläche sollten optimiert werden:

23. Folgendes sollte ergänzt werden:

24.Abschliessend bitten wir Sie, einige Angaben zu sich und Ihrer Erfahrung mit dem PDA zu machen:

Alter:	Geschlecht: w / m	<p>Hinweis: Diese Angaben werden nicht gespeichert und nicht an Dritte weitergegeben!</p>
Beruf:		
Ausbildungsabschluß:		
Fahren Sie regelmäßig Auto? ja / nein		
Haben Sie bereits Erfahrung im Umgang mit PDA's? ja / nein		
Besteht eine Einschränkung Ihrer Sehkraft? ja / nein		
Wenn ja, welche? Kurz- Weit- Farbfehlsicht andere:		

Das war es schon, vielen Dank für Ihre Mühe!

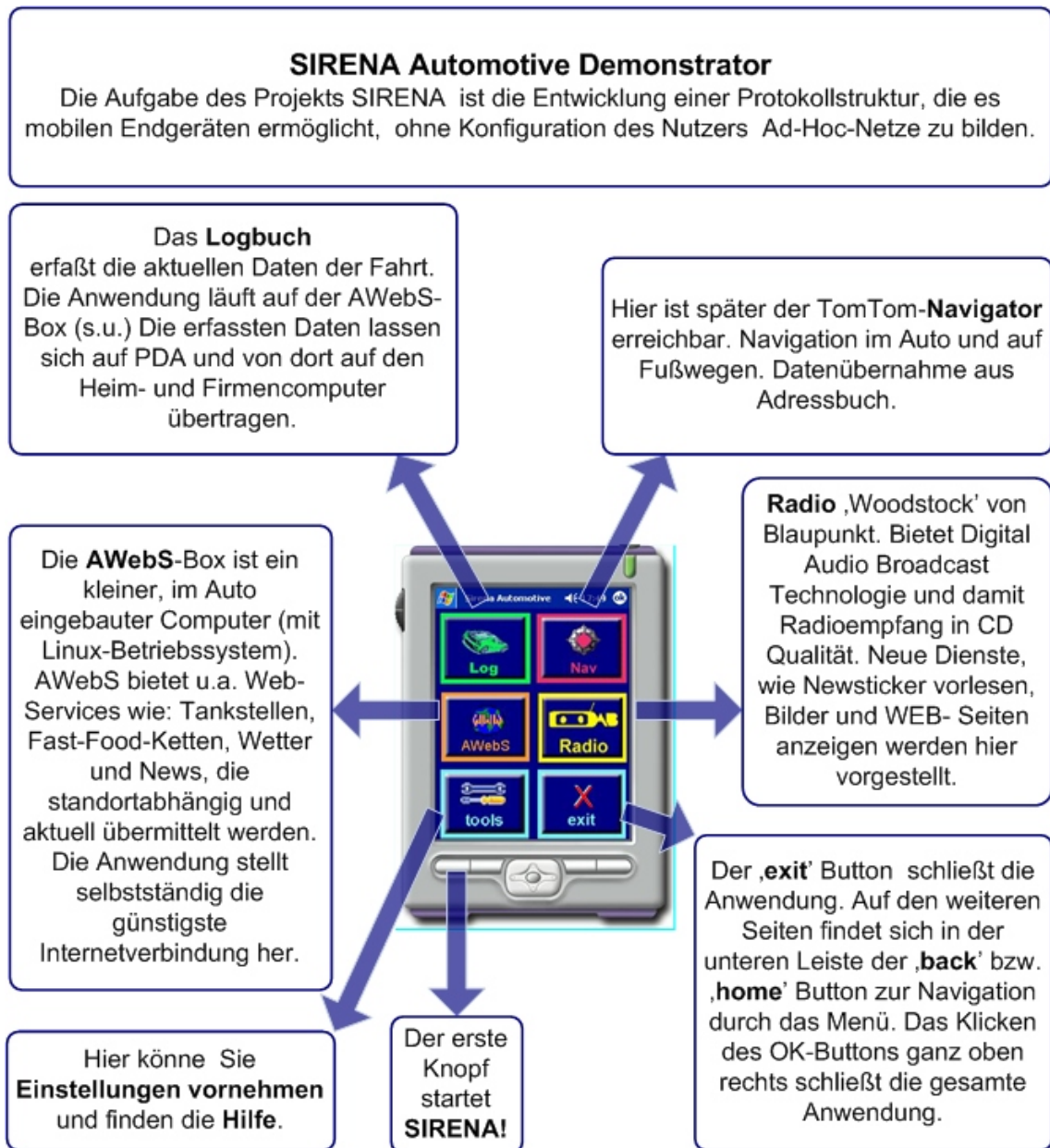


Abbildung D.1: Testbegleitender Informationsbogen

E Eigenständigkeitserklärung

Hiermit versichere ich, dass ich die vorliegende Diplomarbeit selbstständig und nur unter Verwendung der angegebenen Quellen und Hilfsmittel verfasst habe. Die Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen Prüfungsbehörde vorgelegt.

Christiane Rettinger
Berlin, den 17. November 2004