Hochschule Neubrandenburg
University of Applied Sciences

Diplomarbeit

zur Erlangung des akademischen Grades

„Dipl.-Ing.(FH) Geoinformatik“

Entwicklung einer Software zur Kartendarstellung auf mobilen Endgeräten

am Beispiel von Mobiltelefonen mit SymbianOS

Verfasser: Kerstin Breitlow

Betreuer:

Prof. Dr.-Ing. Andreas Wehrenpfennig
(Hochschule Neubrandenburg)

Dr. rer.-nat. Meinolf Asshoff
(geoGLIS oHG)

Neubrandenburg, den 28. Mai 2009

URN: urn:nbn:de:gbv:519-thesis 2009-0014-4

Eidesstatliche Erklärung

Ich erkläre an Eides Statt, dass ich die vorliegende Arbeit selbständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen nicht benutzt und die den benutzten Quellen wörtlich oder inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

Neubrandenburg, den 28. Mai 2009

Hinweis

Es soll an dieser Stelle darauf hingewiesen werden, dass in dieser Arbeit Soft- und Hardwarebezeichnungen verwendet werden, die den allgemeinen Markenschutz unterliegen. Die entsprechenden Marken- und Warenzeichen sind Eigentum der betreffenden Firmen.
Außerdem soll darauf hingewiesen werden, dass das bereitgestellte Kartenmaterial der „geoGLIS oHG“ ausschließlich zur Veranschaulichung und für Testzwecke in Zusammenhang mit dieser Arbeit verwendet werden darf.

Danksagung

An dieser Stelle möchte ich mich bei all denjenigen bedanken, die mich während des Studiums unterstützt haben.

Als ersten möchte ich mich ich bei meinen beiden Betreuern Prof. Dr.-Ing. Andreas Wehrenpfennig von der Hochschule Neubrandenburg und Dr. rer.-nat. Meinolf Asshoff von der „geoGLIS oHG“ bedanken, die mich während der Arbeit mit ihrer fachlichen Kompetenz unterstützt und mir den richtigen Weg gewiesen haben.

Ebenso möchte ich mich bei der gesamten „geoGLIS oHG“ bedanken, dafür dass sie mir diese Arbeit ermöglicht haben und für die gute Zusammenarbeit. Insbesondere möchte ich mich bei Sven Zanon der „geoGLIS oHG“ bedanken, der mir aufkommende Fragen zum Kartenmaterial hilfsbereit beantwortet hat.

Einen weiteren Dank möchte ich an Steve Richmann aussprechen, der mich mit guten Tipps und Ratschlägen bei der Programmierung unterstützt hat.

Des Weiteren möchte ich meinen Kommilitonen und Freunden Henrike Barkmann und Stefanie Prange danken, die mir mit nützlichen Tipps und einen guten Rat immer zur Seite gestanden haben.

Einen ganz besonderen Dank möchte ich an dieser Stelle an meinen Eltern und Großeltern (mütterlicherseits) richten, die mich während des Studiums nicht nur in finanzieller Sicht unterstützt haben, sondern mich auch immer wieder motiviert und ermutigt haben meine Ziele zu ereichen.

Inhaltsverzeichnis

Tabellenverzeichnis

Abbildungsverzeichnis

Abkürzungsverzeichnis

AMS	Anwendungs-Management-Software
API	Application Programming Interface (Programmierschnittstelle)
ATKIS	Amtliches Topographisch-Kartographisches Informationssystem
BKG	Bundesamt für Kartographie und Geodäsie
CDC	Connection Device Configuration
CLDC	Connected Limited Device Configuration
CSV	Comma-Separated Values (Dateiformat)
GPL	General Public License
GPS	Global Positioning System
GPX	GPS Exchange Format
GSM	Global Systems for Mobile Communication
GUI	Graphical User Interface
J2ME	Java 2 Mirco Edition
JAD	Java Application Descriptor
JDK	Java Development Kit
KML	Keyhole Markup Language
KVM	Kilobyte Virtual Machine
LBS	Location Based Service
NMEA	National Marine Electronics Association
OTA	Over The Air
PDA	Personal Digital Assistant
POI	Point Of Interest
Px	Pixel
RAM	Random Access Memory
RMS	Record Management System
ROM	Read Only Memory
SDK	Software Development Kit

UML	Unified Modeling Language
UMTS	Universelles Mobil Telekommunikations System
VM	Virtual Machine
WAP	Wireless Application Protocol
WGS84	World Geodetic System 1984
WMS	WebMapService
WTK	Java ME Wireless Toolkit
XML	Extensible Markup Language

1 Einleitung

1.1 Aufgabenstellung

Das Hauptziel dieser Diplomarbeit ist es, ein Grundkonzept einer Software zu entwickeln, mit der die Darstellung von Karten auf mobilen Endgeräten möglich ist. Als Datengrundlage für die Karten soll das georeferenzierte Kartenmaterial der „geoGLIS oHG“ dienen. Zusätzlich zu der Kartendarstellung soll mit der Software eine Aussage über die aktuelle Position des mobilen Endgerätes getroffen werden. Der ermittelte Standort des Gerätes ist dabei visuell zu veranschaulichen. Zusätzlich zu diesen beiden Funktionen soll es mit der Software möglich sein, neben der Karte und der Position, weitere Geodaten bereitzustellen. Bei der Umsetzung des Konzeptes ist darauf zu achten, dass es sich um eine plattformunabhängige Lösung handelt.

1.2 Grundidee / Motive

Wie im Kapitel 1.1 beschrieben, ist es das Ziel, ein Konzept für eine Software zu entwickeln, die es dem Anwender ermöglicht, ***Karten*** mit mobilen Endgeräten (Mobiltelefon, Smartphone, PDA oder auch BlackBerry) abzurufen und anzeigen zu lassen.
Das Abrufen der Karten soll auf zwei verschiedenen Wegen ermöglicht werden. Beim Abrufen der Karten wird unterschieden zwischen dem ***offline*** und dem ***online*** Laden des Kartenmaterials.

- ***Offline*** Laden der Kartenkacheln:

  Das offline Laden bedeutet, dass der Anwender keine Internetverbindung mit seinem mobilen Gerät herstellen muss, um an die entsprechenden Kartenkacheln zu gelangen. Der Nutzer der Software kann sich die Karten im Vorfeld besorgen, indem er sich die Kacheln aus dem Internet an einem handelsüblichen PC mit Internetanschluss herunterlädt und auf sein mobiles Endgerät überträgt.

- ***Online*** Laden der Kartenkacheln:

  Bei dem online Laden der Kacheln soll der Nutzer der Software die Möglichkeit erhalten, die Karten direkt mit seinem mobilen Endgerät aus dem Internet zu beziehen. Dadurch kann der Anwender den Zeitpunkt, an dem er sich die Karten aus dem Internet laden möchte, selbst bestimmen und ist nicht auf einen separaten PC angewiesen. Das Kartenmaterial kann an Ort und Stelle für die entsprechende Position dynamisch geladen werden. Bei diesem Weg muss jedoch einiges beachtet werden, wie zum Beispiel die Kosten und die Benutzerfreundlichkeit für den Anwender.

Die Kartendarstellung allein ist nicht sinnvoll, da der Betrachter der Kartenkacheln Auskunft über seinen momentanen Aufenthaltsort erhalten möchte. Aus diesem Grund ist es naheliegend eine Positionsbestimmung durchzuführen und dadurch dem Betrachter seine aktuelle Position innerhalb der Karte anzuzeigen. Da sich die Nutzer des mobilen Gerätes in ständiger Bewegung befinden können, ist eine *ständige* Positionsbestimmung erforderlich. Auf diese Weise kann die Aktualität der Positionsangabe gewährleistet werden.
Die Bestimmung der Koordinaten soll weitestgehend geräteunabhängig geschehen. Dabei müssen die verschiedenen Typen und Arten der mobilen Endgeräte beachtet werden. Einige der mobilen Geräte besitzen ein bereits integriertes GPS-Gerät. Neben Modellen von PDA - (Personal Digital Assistant) und Smartphone - Herstellern, bringen auch immer mehr Mobiltelefonhersteller Modelle mit einem eingebauten GPS-Gerät auf den Markt. Bei mobilen Geräten ohne einen internen GPS-Empfänger besteht in der Regel die Möglichkeit, ein solches Gerät über eine Bluetooth - Schnittstelle einzubinden. Bei der Bestimmung der Koordinaten ist nicht nur der Weg über ein GPS-Gerät möglich, es lässt sich außerdem über GSM oder UMTS realisieren. Diese Methode ist jedoch nicht so genau wie die Positionsbestimmung über GPS. Aus diesem Grund wird in dieser Arbeit ausschließlich auf die Möglichkeit der Koordinatenbestimmung mittels GPS eingegangen.

Neben der Kartendarstellung und der Positionsbestimmung sollen dem Nutzer noch verschiedene andere Funktionen zur Verfügung gestellt werden. Eine von diesen Funktionen ermöglicht das Einbinden von ***POI's (Points of Interest - dt. interessante Orte)***, wodurch für den Nutzer interessante Orte seiner Umgebung dargestellt werden können. Zu den POI's zählen unter anderem Museen, Campus-Informationen (wie die Bibliotheken oder die Cafeteria), Gastronomie, Parkhäuser oder etwa Tankstellen. Für die Funktionalität des Einbindens solcher Punkte ist es erforderlich, den Nutzern die Option des Imports zur Verfügung zu stellen.
Die gleiche Funktionalität (Import) wird bei den ***Waypoints (dt. Wegpunkte)*** benötigt. Dies bildet eine weitere Funktion, die den Anwendern bereitgestellt werden soll. Ein Waypoint ist eine isolierte geografische Position. Dabei handelt es sich um einen durch den Nutzer manuell festgelegten Punkt entlang einer Wegstrecke. In der Regel handelt es sich dabei um Orte, die der Nutzer bei seinen Outdoor - Aktivitäten entdeckt und festhalten möchte. Aus diesem Grund wird innerhalb der Funktion für den Nutzer die Möglichkeit geschaffen, solche Punkte zu setzen und zu exportieren.
Darüber hinaus soll die Software für das ***Tracking*** geeignet sein. Tracking ist das Aufzeichnen von so genannten GPS - Tracks. Die Tracks beinhalten die vom Nutzer zurückgelegte Wegstrecke. Wie bei den Waypoints soll auch hier die Möglichkeit für den Import und den Export geschaffen werden.

In der Abbildung 1 sowie in der Tabelle 1 werden die Funktionen, die in die Software integriert werden sollen, zusammenfassend dargestellt:

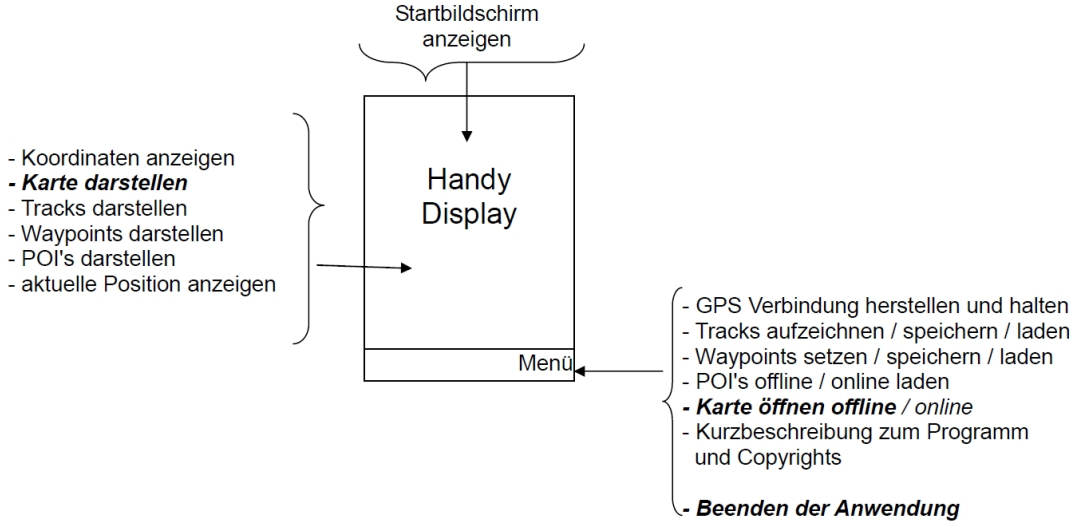


Abbildung 1: Funktionsübersicht der Software

Funktionen	Inhalt
Karten einbinden	Anzeigen von Karten Import der Kartenkacheln(offline/online)
Positionbestimmung	Ermitteln der aktuellen Postion Ausgabe der ermittelten Koordinaten Anzeige der aktuellen Position auf der Karte
Tracking	Aufnahme von Tracks Darstellen der Tracks Import/ Export der Daten
POI (Points Of Interest) anzeigen	Laden und Darstellen von POI's Import der Daten
Waypoints anzeigen	Laden und Darstellen von Waypoints Setzen von eigenen Waypoints Import/ Export der Daten
Kompassanzeige	Anzeige der Richtung (Nur bei Positionsänderung möglich)
Koordinatenanzeige	Anzeige der Koordinaten (Nur bei Positionsbestimmung möglich)

Tabelle 1: Übersicht über die Funktionen

Der Grund für solch eine Aufgabenstellung liegt praktisch auf der Hand: Die Nutzung von mobilen Endgeräten nimmt von Jahr zu Jahr zu. Weltweit besitzen mehr als 2 Milliarden Menschen ein Mobiltelefon [na09]. Wie aus der Statistik in Abbildung 2 ersichtlich, stieg die Anzahl der Mobilfunkanschlüsse alleine in Deutschland von 79,2 Millionen im Jahr 2005 auf 107,4 Millionen im Jahr 2008.

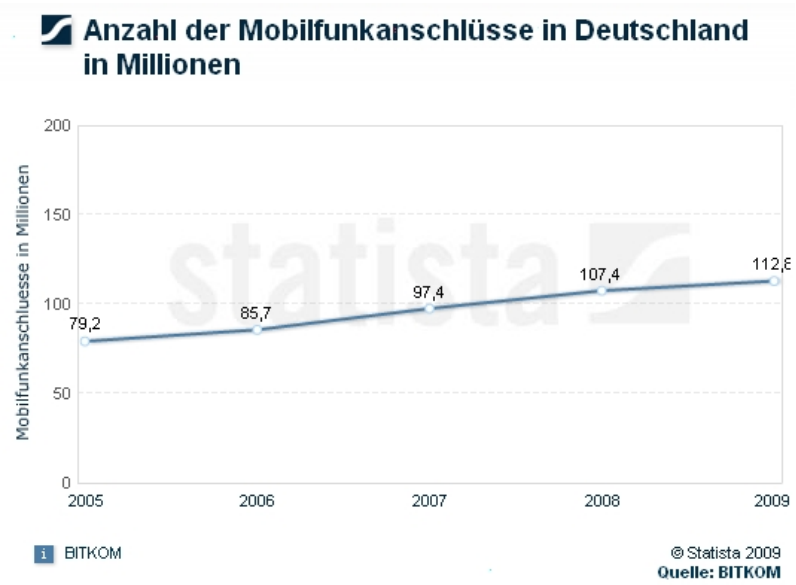


Abbildung 2: Statistik über die Anzahl der Mobilfunkanschlüsse in Deutschland
Quelle: [BIT09]

Um dem Anwender in seiner Mobilität noch mehr Möglichkeiten bereitzustellen, soll im Zusammenhang mit dieser Arbeit eine Software entwickelt werden, die dem Nutzer eine einfache und schnelle Bedienung erlaubt. Ein weiterer Grund solch eine Software zu entwickeln ist, dass der Anwender nur noch ein Gerät mit sich tragen muss. Er braucht sich

kein zusätzliches Navigationssystem anzuschaffen und kann auf die Papierkarten verzichten. Falls das Mobiltelefon über kein integriertes GPS-Gerät verfügt, ist jedoch zusätzlich eine GPS-Maus erforderlich. Diese ist bei weitem nicht so kostenintensiv wie ein Navigationsgerät. Die GPS-Maus kann auf Grund ihrer kleinen Größe problemlos in der Tasche verstaut werden und stellt so keinen zusätzlichen Ballast dar.
Das Hauptaugenmerk wird bei der Entwicklung auf das Einbinden von Kartenmaterial in die Software gelegt. Der Grund dafür ist, dass die meisten Freeware Programme die Option der Kartendarstellung nicht anbieten. Durch die Positionsanzeige auf der Karte findet sich der Nutzer der Software in seiner Umgebung besser zurecht. Dies ist zum Beispiel beim Geocaching, Tracking oder Sightseeing von Vorteil.
Im Vorfeld muss überlegt werden, auf welchem Weg dem Anwender die entsprechenden Geodaten (Karte, POI's, Waypoints) zur Verfügung gestellt werden sollen. Dafür kommen in der Softwareentwicklung für mobile Endgeräte zwei verschiedene Varianten von Anwendungen, mobile Dienste, in Betracht. Zum einen gibt es die Variante der „Browser basierten Anwendung". Bei dieser Anwendungsmöglichkeit ist eine Internetanbindung zwingend erforderlich. Zum anderen gibt es die Möglichkeit einer „Stand-Alone Anwendung". Hierbei kann auf eine Internetanbindung verzichtet werden. Bevor diese Software jedoch verwendet werden kann, muss sie durch den Anwender installiert werden.
Bei der „Browser basierten Anwendung" handelt es sich um einen WMS-Client der über einen mobilen Browser aufgerufen werden muss, um mit einen Server zu kommunizieren. Aus diesem Grund muss für den WMS-Client eine Netzwerkanbindung im mobilen Endgerät zur Verfügung stehen. Für mobile Endgeräte gibt es verschiedene Übertragungsarten. Bei einigen Geräten gibt es die Möglichkeit einer Netzwerkanbindung über WLAN (ortsgebunden). Neben WLAN gibt es die beiden Übertragungsarten GPRS und UMTS, die jedem mobilen Endgerät, das der Kommunikation dient, zur Verfügung stehen. Diese beiden Arten sind im Gegensatz zu dem WLAN nicht ortsgebunden. Abhängig vom Netzanbieter können diese beiden Varianten für den Internetzugriff sehr kostenintensiv werden.[Sta07]
Die Übertragungsarten haben unterschiedliche Übertragungsgeschwindigkeiten [Sta07]:

WLAN - 11 bis 600 Mbit/s, z.Z. üblich 54 Mbit/s

GPRS - 9,05 - 171,24 kbit/s

UMTS - 64 kbit/s - 2Mbit/s (HSDPA)

Damit ein WMS-Client auf mobilen Endgeräten lauffähig ist, müssen auf den Geräten verschiedene Softwareprodukte vorhanden sein. Für eine hardwareseitige Client-Anwendung muss auf dem Gerät eine Java-VM (JavaME-Virtual Machine) zur Verfügung stehen. Zusätzlich zu der VM(Virtual Machine) wird ein Internetbrowser (JavaScript fähig) für die serverseitige Client-Anwendung mit dynamischen Inhalten benötigt. Für die mobilen Browser müssen die Internetseiten entsprechend den Anforderungen mobiler Endgeräte angepasst werden, da diese nicht die gleichen Datenmengen empfangen können, wie herkömmliche PC's. [Sta07]

Die Diplomarbeit [Sta07] von Alexander Anders und Torsten Starke beschäftigt sich mit dieser Thematik ausführlicher.
Bei der „Stand-Alone Software“ handelt es sich um die Entwicklung einer Java-Applikation. Für die Lauffähigkeit dieser Applikation auf mobilen Endgeräten ist nur eine Java-VM auf dem jeweiligen Gerät nötig. Diese VM ist auf den meisten der Geräte bereits vorinstalliert. Eine Netzwerkverbindung und ein Internetbrowser sind in diesem Fall nicht notwendig. Aus diesen Gründen wird sich dafür entschieden, eine „Stand-Alone Software“ zu entwickeln.

2 Recherche und Analyse

2.1 Bedarfsanalyse

2.1.1 Anwender

Mit der Software wird beabsichtigt, eine Vielzahl von Anwendern anzusprechen.
Sie ist jedoch hauptsächlich für Zielgruppen gedacht, die auf den Offroad - Bereich Wert legen. Das umfasst auch die Gruppe von Leuten, die sich im Outdoor - Sport bewegen. Dazu gehören unter anderem Wanderer, Kanuten oder Fahrradfahrer.
Die Software ist für die Nutzer, die sich eine mobile Kartendarstellung wünschen und auf einen Faltplan verzichten möchten. Außerdem ist sie für Anwender, die sich abseits der Straße bewegen und dennoch auf eine Karte nicht verzichten möchten, um sich auch abseits der Straßen und Wege zu orientieren, wie zum Beispiel Freunde des Geocachings. Die Möglichkeit, sich auch abseits der Straße zu orientieren, ist stark vom Kartenmaterial abhängig. Deshalb sollte beim Kartenmaterial darauf geachtet werden, dass die Karte detailliert genug ist, um sich dort orientieren so können.
In der Software ist zusätzlich eine Positionsbestimmung vorgesehen. Die dabei ermittelte Position soll auf den dazugehörigen Kartenausschnitt dargestellt werden. Dadurch ist die Software für fast jeden Anwendungsbereich geeignet. Die im Hintergrund dargestellte Karte soll dem Anwender als Rasterkarte zur Verfügung gestellt werden. Eine vektorielle Darstellung der Karte wird von vornherein ausgeschlossen. Darauf wird in dem Kapitel „Geodaten 2.4.1“ näher eingegangen.
Neben der Kartendarstellung hat der Anwender die Möglichkeit GPS - Tracks aufzuzeichnen. Dadurch können Kajaktouren, Motorradtouren, Fahrradtouren oder Skiabfahrten am PC analysiert und dargestellt werden. Neben dem Aufzeichnen der zurückgelegten Strecken ist es dem Anwender auch möglich, für ihn interessante Orte, wie die Hütte für die Aprésski-Party, einen schön gelegenen Biergarten oder ein Museum als Waypoint zu setzen und zu speichern. Es soll neben dem Setzen solcher Punkte auch möglich sein, solche in die Software zu importieren. Dadurch ist es dem Anwender möglich, Informationen aus der Umgebung darzustellen, oder selbst Informationen zu sammeln. Aus diesem Grund wird die Software hauptsächlich im Bereich des Tourismus ihren Einsatz finden. Über diese Zielgruppe hinaus werden noch weitere Anwender angesprochen, wie zum Beispiel Neueinwohner einer Stadt oder Studenten des ersten Semesters.

In der Abbildung 3 wird die Software aus Sicht des Anwender beschrieben. Die Abbildung 3 zeigt auf, welche Funktionen dem Nutzer mit der Software bereitgestellt werden.

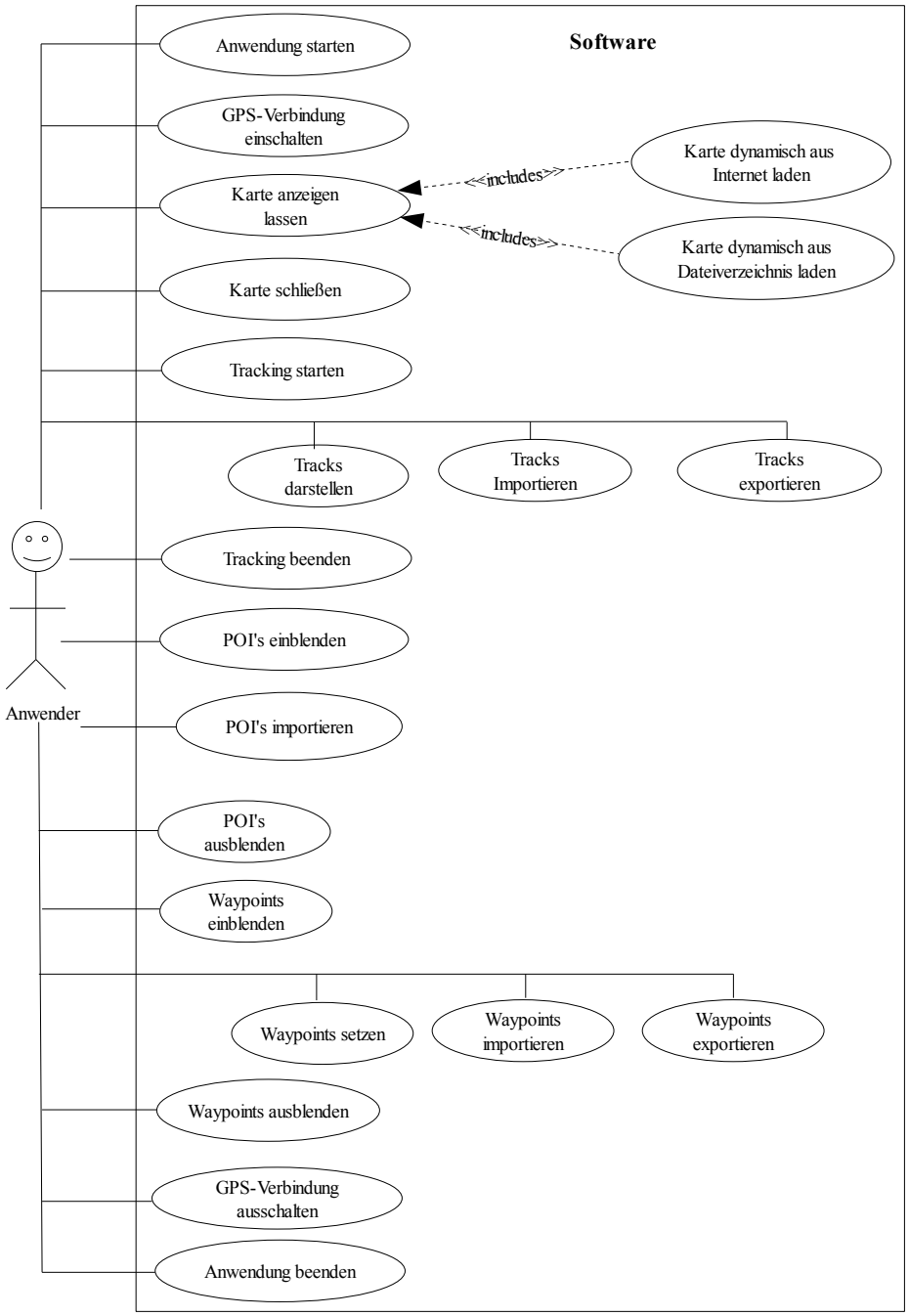


Abbildung 3: Use-Case-Diagramm der Software aus Sicht des Anwenders

2.1.2 Anwendungsbereiche

Die aus dem Kapitel „2.1.1 Anwender“ hervorgehenden Zielgruppen lassen sich in verschiedene Anwendungsbereiche zusammenfassen. Diese Anwendungsbereiche sind in der Tabelle 2 aufgelistet:

Anwendungsbereich	Anwender	Beispielanwendungen
Tourismus	Wanderer	Wanderkarte, Wanderroute (GPS-Track anzeigen / aufnehmen)
	Fahrradfahrer	Fahrradkarte, Fahrradtour (GPS-Track's anzeigen / aufnehmen)
	Kanuten	Kanukarte, Kanutour (GPS-Track's einblenden / aufnehmen)
	Stadttourist	Stadtinformation (Karte + POI's)
Beruf		
− > Ausbildung	Studenten	1. Semesterinformation (Karte + POI's)
− > mobile Berufe	Kurierfahrer	Umgebungsinformation, Karte +
	Straßenreinigung	zu fahrende Tour (GPS-Track's)
Einwohner	Neueinwohner	Stadtinformation (Karte + POI's)

Tabelle 2: Anwendungsbereiche der Software

Diese Anwendungsbereiche lassen sich in zwei Haupteinsatzgebiete aufschlüsseln. Dazu gehört zum einen die Verwendung der Software als „**mobile Wanderkarte**“. Dieses Gebiet findet seinen Einsatz im Bereich des Tourismus. Mit der Tracking - Funktion kann die gelaufene Route aufgezeichnet und gespeichert werden. Die Speicherung der Daten erfolgt in einem bestimmten Format (*.gpx*-Format), darauf wird in dem Kapitel 2.4.1 näher eingegangen. Durch das Speichern der Daten wird es dem Anwender ermöglicht, seine Daten zu sammeln und zu einem späteren Zeitpunkt auszuwerten. Darüber hinaus ist es auch möglich, sich die aufgezeichneten GPS - Tracks erneut in das Programm zu laden. Auf verschiedenen Internetportalen ist es möglich, die Daten mit Gleichgesinnten zu tauschen oder sie sich in bestimmten Kartenviewern anzeigen zu lassen. Ein Beispiel für solch eine Website ist unter folgendem Link zu finden [Kar09]. Durch das Laden von POI's und das Setzen von Waypoints kann die Wandergegend zusätzlich veranschaulicht werden. So kann die Software auf den Gebieten der Wandertouren, Fahrradtouren, Kanutouren oder anderen Outdoor - Aktivitäten zum Einsatz kommen.
Das zweite sich ergebende Haupteinsatzgebiet ist das „**mobile Informationssystem**“. Die POI's enthalten in der Regel neben der Positionsangabe noch zusätzliche Informationen, die dem Nutzer zur Verfügung gestellt werden können. Dazu gehören zum Beispiel Informationen zu Bibliotheken oder Tankstellen mit den Öffnungszeiten, Adressen und Telefonnummern. So können dem Studenten zum Beispiel Semesterinformationen bereitgestellt werden. Der Student soll so die Möglichkeit erhalten, sich leichter auf dem Hochschulgelände zu orientieren, indem er sich mit der Software Standorte wie etwa die Bibliothek oder Cafeteria anzeigen lassen kann. Für Erstsemestler, die neu in der Stadt sind, bzw. für Touristen sind die POI's mit den Stadtinformationen gedacht. Diese POI's sollen unter

anderem Standorte wie Museen, Sehenswürdigkeiten und Kinos enthalten.
Neben diesen Anwendungsbereichen gibt es noch die Möglichkeit, die Software in dem Bereich mobiler Arbeit zu nutzen. Dazu zählen unter anderem Kurierfahrer oder die Straßenreinigung. In diesem Bereich ist es möglich, sich die Arbeitsrouten (GPS - Tracks) und die entsprechende Karte des Arbeitsgebietes darzustellen. Dadurch ist es für den Arbeiter möglich genaue Informationen zu seinem Arbeitsumfeld zu erhalten.

Daraus lässt sich erkennen, dass die Software auf vielen verschiedenen Gebieten ihren Einsatz finden kann. Sie ist dadurch nicht auf einen Bereich festgelegt und kann so eine größere Bandbreite von Kunden, bzw. Nutzer ansprechen.

2.1.3 Lizenzierung: Open Source / Kommerziell

Bei der Software, die im Zusammenhang mit dieser Arbeit entwickelt wird, ist entschieden worden, diese den Anwendern vorerst frei zur Verfügung zu stellen. Kunden und andere interessierte Nutzer sollen sich die Software auf der Internetpräsens der **„geoGLIS oHG“** ([go09a], [go09b]) kostenlos herunterladen können.
Bei der Entwicklung der Software wird zum Teil auf bereits bestehende Open Source - Lösungen zurückgegriffen. Solche Open Source - Produkte haben den Vorteil, dass der Code offen liegt. Dieser Code ist so für jeden einsehbar und darf unter der Berücksichtigung verschiedener Bedingungen in der eigenen Software verwendet werden. Aus diesem Grund werden einige der Open Source - Produkte bei der Softwareentwicklung mit berücksichtigt und unter anderem mit verwendet. Die Open Source Software wird in der Regel unter der GNU General Public License (GPL) bereitgestellt. Unter dem Internetlink [Ger09] sind die rechtlichen Bedingungen der „GNU GPL“ aufgeführt. Nachfolgend soll kurz auf die grundlegenden Freiheiten der „GNU GPL“ eingegangen werden.

„Die GPL gewährt jedermann die folgenden vier Freiheiten als Bestandteile der Lizenz.

0 Das Programm darf ohne jede Einschränkung für jeden Zweck genutzt werden. Kommerzielle Nutzung ist hierbei ausdrücklich erlaubt.

1 Kopien des Programms dürfen kostenlos oder auch gegen Geld verteilt werden, wobei der Quellcode mitverteilt oder dem Empfänger des Programms auf Anfrage zum Selbstkostenpreis zur Verfügung gestellt werden muss. Dem Empfänger müssen dieselben Freiheiten gewährt werden - wer z.B. eine Kopie gegen Geld empfängt, hat weiterhin das Recht, diese dann kommerziell oder auch kostenlos zu verbreiten. Lizenzgebühren sind nicht erlaubt. Niemand ist verpflichtet, Kopien zu verteilen, weder im Allgemeinen, noch an irgendeine bestimmte Person - aber wenn er es tut, dann nur nach diesen Regeln.

2 Die Arbeitsweise eines Programms darf studiert und den eigenen Bedürfnissen angepasst werden.

3 Es dürfen auch die gemäß Freiheit 2 veränderten Versionen des Programms unter den Regeln von Freiheit 1 vertrieben werden, wobei dem Empfänger des Programms der Quellcode der veränderten Version verfügbar gemacht werden muss. Veränderte Versionen müssen nicht veröffentlicht werden; aber wenn sie veröffentlicht werden, dann darf dies nur unter den Regeln von Freiheit 1 geschehen.“[fE09c]

Auf der Website der **„geoGLIS oHG“**([go09a], [go09b]) wird dem Nutzer das gewünschte Kartenmaterial zum Download bereitgestellt. Dieses Kartenmaterial ist für den Anwender jedoch nicht frei verfügbar und muss käuflich erworben werden. Dafür handelt es sich um Karten, die sehr detailliert sind und ohne jegliche Werbung auskommen. Speziell für den Outdoor - Bereich enthalten die Karte sehr viel mehr Details, wie Landwege und Waldwege, als bei anderen Kartendienstanbietern.

2.2 Marktanalyse Software

2.2.1 Softwaretest

Vor dem Beginn der Softwareentwicklung wurden ausführliche Recherchen durchgeführt. Dabei wurde eine Vielzahl von Software im Internet und in Fachzeitschriften (z.B. c't [Döl08], [RD08]) gefunden, die eine Nutzung von GPS und Karten mit einem mobilen Endgerät ermöglichen. Diese Software wurde heruntergeladen und auf den dafür vorgesehenen mobilen Endgeräten installiert und getestet.
Die Software wird auf verschiedene Kriterien getestet. Anhand dieser Kriterien soll geprüft werden, welche verschiedenen Funktionalitäten die getestete Software bereitstellen. Diese Kriterien werden in der Liste 1 aufgeführt.

Liste 1: Kriterien für den Softwaretest:

- Art der Software
- Kosten der Software
- Lizenzart (Open Source)
- Demoversion
- Anwendungszweck
- für welche Geräte
- Betriebssystem
- wie Installieren
- Quellcode verfügbar
- Entwicklungsumgebung
- Karten einbinden
- Kartenformate
- GPS
- Positionsbestimmung
- Koordinatenanzeige
- Positionsanzeige auf Karte
- Kompassanzeige
- Tracking
- Formate der Tracks
- Darstellung Waypoints
- Setzen von Waypoints
- Im-/Export Waypoints
- Format f. Waypoints
- Darstellung POI's
- Import von POI's
- Format für POI's
- Routing möglich

Aus dieser Vielzahl von Kriterien haben sich die nachfolgenden Kriterien (Liste 2) herausgebildet, auf welche beim Testen der verschiedenen Software ein besonderes Augenmerk gelegt werden soll. In der im Zusammenhang mit dieser Arbeit entwickelten Software sollen diese Kriterien die Hauptfunktionen bilden, die dem Nutzer bereitgestellt werden sollen. Zunächst wird geprüft, ob die Software den funktionellen Anforderungen der Kriterien aus Liste 2 entsprechen.

Liste 2: Hauptkriterien für die Software: (siehe auch Tabelle 1)

- Karte einbinden
- Positionsbestimmung
- Tracking
- Waypoint - Funktionen
- POI - Funktionen
- Kompassanzeige
- Koordinatenanzeige

Erfüllte die Software diese Anforderungen, werden die restlichen Funktionen aus der obigen Liste 1 beim Testen mit einbezogen.
Wie aus der Liste 1 zu entnehmen, wird neben den Funktionalitäten der Software darauf geachtet, um welche Softwareart es sich handelt. Dabei wird unterschieden, ob es sich um

ein Open Source -, ein Freeware- oder ein Sharewareprodukt handelt. Bei den Sharewareprodukten wird - wenn verfügbar - auf Demoversionen zurückgegriffen. Dadurch ist es möglich, einen Einblick in die bereitgestellten Funktionen zu gewinnen.
Durch diesen ausführlichen Softwaretest konnte ein Überblick darüber gewonnen werden, welche Software bereits auf dem Markt existiert. Auf einzelne Software wird in dem nachfolgenden Kapitel 2.2.2 näher eingegangen. Zusätzlich kann so abgewogen werden, ob die Neuentwicklung einer Software sinnvoll ist. Darauf wird in dem Kapitel 2.2.3 näher eingegangen.

Eine ausführliche Aufstellung der getesteten Software anhand der Kriterien findet sich aufgrund ihrer Größe in einer Excel-Tabelle „*SoftwareTest.xls*“. Diese Tabelle ist auf der CD, die dieser Arbeit als Anlage beiliegt, enthalten.

2.2.2 Was gibt es bereits auf den Markt?

Bei den Recherchen wurde festgestellt, dass bereits eine große Menge von Shareware-, Freeware- und Open Source - Produkten auf den Markt existieren. Die meisten dieser Produkte stellen fast alle der gewünschten Funktionalitäten (siehe Tabelle: 1) bereit. Jedoch konnte man beim Testen der Software feststellen, dass bei der überwiegenden Menge der Freeware - und Open Source - Produkte das Einbinden von Karten nicht vorgesehen ist. In den Tabellen 3 und 4 ist eine Auswahl der getesteten Software aufgelistet. Auf diese Software wird im nachfolgenden näher eingegangen.

	Gerät	Software-art	Source-Code einsehbar	Doku-mentation	Code kom-mentiert
GPSwith-MAPS	PDA	Shareware	nein	Benutzer-oberfläche	-
VlkGPS	Mobil-telefon	Open-Source GNU - GPL	ja	Benutzer-oberfläche	teilweise
TrackMy-Journey (TMJ)	Mobiltelef. BlackBerry Windows CE	Freeware	nein	Benutzer-oberfläche	-
TrekBuddy	Mobiltelef. BlackBerry Windows CE Palm	Freeware	nein	nein	-
LocateME	Mobil-telefon	Open-Source GNU - GPL	ja	teilweise	teilweise
GPSTrack	Mobil-telefon	Open-Source GNU - GPL	ja	nein	teilweise

Tabelle 3: Software Auflistung

Name der Software	Funktionen						
	Karte	GPS	Tracking	POI	Way-point	Anzeige	
						Kompass	Koordi-naten
GPSwithMAPS	ja	ja	ja	nein	ja	nein	ja
VlkGPS	nein	ja	ja	nein	ja	ja	ja
TrackMyJourney (TMJ)	ja	ja	ja	nein	ja	ja	ja
TrekBuddy	ja	ja	ja	nein	ja	ja	ja
LocateME	nein	ja	nein	nein	ja	ja	ja
GPSTrack	nein	ja	ja	nein	nein	nein	ja

Tabelle 4: Erfüllt die Software die gewünschten Funktionen?

Im Gegensatz zur Open Source - Software, wurde bei den Recherchen zwei Freeware - Programme gefunden, die das Laden von Kartenmaterial unterstützen. Dazu gehören „**TrackMyJourney**“ [Tra09] und „**TrekBuddy**“ [For09]. Die andere bei den Recherchen gefundene Freeware unterstützt das Laden von Karten jedoch nicht. Bei diesen beiden Programmen ist es nicht möglich, eine Einsicht in den Source Code zu gewinnen. Hinzu kommt, dass das Kartenmaterial in einem speziellen Format vorliegen muss. Bei der Software „TrackMyJourney“ (TMJ) handelt es sich dabei um ein Format mit der Endung ***.tmj***.
Um die Software TMJ herunterladen zu können, ist es notwendig, sich zu registrieren. Dies ist auf der Website der Software [Tra09] möglich. Auf dieser Website bekommt der Nutzer außerdem eine ausführliche Einweisung in das Programm. Neben den Programmerläuterungen erhält der Anwender die Möglichkeit über diese Seite Kartenmaterial zu beziehen. Zum einen findet der Anwender bereits zum Download fertiges Kartenmaterial oder zum anderen die Erläuterung, wie man sich sein eigenes Kartenmaterial erstellt. Um sich jedoch sein eigenes Kartenmaterial erstellen zu können, ist ein Zusatzprogramm erforderlich, der „MapOrganizer“. Bei dem Kartenmaterial handelt es sich um Rasterbilder. Diese Rasterbilder werden unter Verwendung des Programms von dem „OpenStreetMap“ ([Ope09b], [Ope09a]) Server bezogen und können in dem vorgesehenen Format gespeichert werden. Die Software „TrackMyJourney“ stellt eine Vielzahl von Funktionen bereit. Dazu gehören alle in der Software gewünschten Funktionen (siehe Tabelle: 1), wie zum Beispiel die Positionsbestimmung und die Anzeige der Position auf der Karte. Die einzige Funktion, die von diesem Programm nicht unterstützt wird, ist die Darstellung von POI's. Dafür stellt das Programm jedoch die Funktion zur Darstellung von Waypoints bereit. Darüber hinaus wird die Software dem Markt ständig angepasst und ist somit immer aktuell. Die Software kann gerätespezifisch heruntergeladen werden. Das bedeutet, dass die Software für verschiedene Hersteller und deren verschiedenste Modelle angepasst wird. So bietet „TrackMyJourney“ zum Beispiel für die unterschiedlichen Nokia -, bzw. Sony Ericsson - Modelle verschiedene Programmdownloads. Dadurch wird es dem Anwender ermöglicht, das interne GPS-Gerät seines Mobiltelefons zu nutzen. Ein Beispiel für ein solches Mobiltelefon ist das Sony Ericsson W760i.
Eine weitere bei den Recherchen gefundene Freeware ist „TrekBuddy“[For09]. Diese Software stellt den Anwendern die gewünschten Funktionen zur Verfügung. Ausgenommen ist hier jedoch die Funktion zur Darstellung von POI's. „TrekBuddy“ unterstützt, ebenso wie die andere Freeware (TrackMyJourney), das Einbinden von Kartenmaterial und die Anzeige der Position mittels GPS. Im Gegensatz zu „TrackMyJounrney“ stellt „TrekBuddy“ nur eine universelle Version zum Downloaden bereit. Diese Version unterstützt ein externes GPS-Gerät ebenso wie einen geräteinternen GPS-Empfänger. Für die Einbindung von Kartenmaterial steht dem Anwender im Internet eine Website [awo09] zur Verfügung. Dort kann er sich den gewünschten Kartenausschnitt auswählen und in dem entsprechenden Dateiformat herunterladen. Auf dieser Seite erhält er auch die Möglichkeit Waypoints zu setzen und ebenso wie das Kartenmaterial herunterzuladen. Wie bei „TrackMyJourney“ besteht auch hier das Kartenmaterial aus Rasterbildern. Diese basieren bei „TrekBuddy“

auf „Google Maps“. Auf der Website [For09] kann das Programm heruntergeladen werden. Hier steht für den Nutzer der Software ein Forum bereit. In diesem Forum werden Funktionen und Besonderheiten der Software durch andere Anwender näher gebracht. Der Nachteil an dieser Seite ist, dass die einzelnen Informationen verstreut vorliegen und teilweise viel Zeit für gesuchte Informationen aufgebracht werden muss.
Es ist noch hervorzuheben, dass beide Freeware-Programme „TrackMyJourney“ und „TrekBuddy“ die unterschiedlichen Typen von mobilen Endgeräten berücksichtigen. Auf die unterschiedlichen Gerätetypen wird im Kapitel 2.3.1 näher eingegangen.
Neben den beiden Freeware-Programmen wird auch ein Shareware-Produkte in den Tabellen 3 und 4 aufgeführt, „**GPSwithMAPS**“[Ehr09b]. Das Softwareprogramm unterstützt ebenso wie die Freeware-Programme das Bestimmen der Position mittels GPS. Neben der Positionsbestimmung wird auch die Funktion zur Einbindung von Karten zur Verfügung gestellt. Für das Einbinden des Kartenmaterials stellt das Programm „GPSwithMAPS“ dem Nutzer drei verschiedene Möglichkeiten zur Verfügung. Die erste Variante ist das Integrieren des Kartenmaterials von Google Maps. Dazu stellt „GPSwithMAPS“ eine Internetadresse [Ehr09a] zur Verfügung, von der das Kartenmaterial bezogen werden kann. Für eine genaue Beschreibung zur Einbindung der Karte stellt „GPSwithMAPS“ eine Online-Hilfe zur Verfügung. Eine weitere Variante zur Einbindung des Kartenmaterials, ist das direkte Laden der Karte mit dem „PDA“ oder einem „Windows Mobile Gerät“ über das Internet. Dazu benötigen die mobilen Geräte jedoch einen WLAN - Zugang. Bei beiden genannten Varianten ist das Einmessen der Karten durch den Nutzer nicht mehr nötig. In der dritten Variante kann der Nutzer selbst wählen, woher er das Kartenmaterial bezieht. In diesem Fall muss dieses jedoch vor der Nutzung kalibriert werden. Eine ausführliche Beschreibung der Software findet sich auf der Website [Ehr09b] und in der Online-Hilfe. Bis auf die Einbindung der POI’s stellt die Software alle benötigten Funktionen zur Verfügung. Die Kosten für diese Software betragen etwa 34,00 Euro.
Die Preise für die Shareware - Produkte bewegen sich zwischen 15,00 Euro bis 30,00 Euro. Bei der gefundenen Shareware ist anzumerken, das diese hauptsächlich auf „PDA’s“ und „Windows Mobile Geräten“ laufen. Der größte Teil der Software für Mobiltelefone basiert auf Freeware, bzw. Open Source.
In den „Tabellen 3 und 4“ sind neben den kurz beschriebenen Programmen auch drei Open-Source-Produkte zu finden. Dazu gehören „VlkGPS“([Döl08], [Vlk09]), „GPSTrack“[Pet09] und „LocateME“ [Dev09]. Alle drei Programme werden dem Nutzer unter der GNU General Public License (GPL) zur Verfügung gestellt.
Mit jeder dieser Open Source Software ist es dem Anwender möglich eine Positionsbestimmung mittels GPS durchzuführen. Bei den Programmen „GPSTrack“ und „LocateME“ ist dies nur über ein externes Bluetooth - Gerät möglich. Bei keinem dieser drei Programme - „LocateME“, „GPSTrack“ und „VlkGPS“ - ist eine Funktion für das Darstellen von Karten integriert.
„LocateME“ ist ein sehr einfaches Programm, das nur für die Positionsbestimmung geeignet ist. Diese Position kann als Punkt gespeichert werden und mittels SMS an Freunde gesandt werden. In dem Programm „GPSTrack“ ist neben der Positionsbestimmung auch

noch das Tracking und das Setzen von Waypoint möglich. Diese Software ist genauso wie „LocateME“ ein sehr einfach gehaltenes Programm. Das Open Source - Programm „VlkGPS“ ist von seiner Funktionalität her sehr umfangreich. Es bietet bis auf das Darstellen von Karten und das POI Einbinden alle gewünschten Funktionen (siehe Tabelle: 1). Die Positionsbestimmung ist bei dieser Software auch über ein internes GPS-Gerät möglich.
Im Gegensatz zu den anderen beschriebenen Programmen, sind die drei Open Source - Produkte nur auf Mobiltelefonen betriebsfähig.

2.2.3 Wahl der Software

Die im Kapitel 2.2.2 erwähnte Open - Source - Software „VlkGPS“ und „LocateME“, „GPSTrack“ stehen unter der „GNU General Public License“ zur Verfügung. Dadurch ist es möglich, diese Software um eigene Funktionen zu erweitern und bereits bestehende Funktionen den eigenen Vorstellungen anzupassen.
Es war am Anfang dieser Arbeit geplant, die „VlkGPS“ - Software als Grundlage zu nehmen, da diese fast alle der gewünschten Funktionen (siehe Tabelle: 1) bereitstellt. Zu diesen Funktionen gehört das Aufzeichnen von Wegen (Tracking), das Exportieren gesetzter Waypoints zusammen mit einem Zeitstempel in den Datenformaten CSV, KML für Google Earth, GPX für diverse Geo-Tagging sowie das zu „geocaching.com“ [Geo09a] kompatible .LOC-Format. Wie aus der Tabelle 1 ersichtlich, stellt die Software lediglich die Funktion der POI- und der Karteneinbindung nicht zur Verfügung. Bei „VlkGPS“ handelt es sich um eine sehr umfangreiche Software. Es findet sich auf der Website von „VlkGPS“ zwar eine Dokumentation, diese fällt jedoch sehr spärlich aus. Diese beschreibt nur kurz die einzelnen Funktionen aus Sicht des Nutzers. Es sind keinerlei Beschreibungen des Source - Codes enthalten. Darüber hinaus ist der Quellcode nur teilweise und nicht ausführlich kommentiert. Auf Grund dessen wäre die Einarbeitungszeit in die Software zu umfangreich, um darauf in dieser Arbeit einzugehen.
Die erwähnte Open - Source - Software „LocateME“ und „GPSTrack“ wird in die Wahl für eine Software, die als Grundlage dienen soll, nicht näher in Betracht gezogen. Wie bei der Software „VlkGPS“ ist der Quellcode zwar teilweise kommentiert, es gibt jedoch keine Dokumentationen zu den beiden Programmen selbst. Auch hier wäre viel Zeit notwendig, um sich in diese Software einzuarbeiten.
Aus diesen Gründen erscheint es sinnvoll, eine eigene Software zu entwickeln. Das Hauptaugenmerk bei der Softwareentwicklung wird dabei auf die Funktionen für das „Karteneinbinden“ gelegt. Zum einen bezieht die im vorherigen Kapitel 2.2.2 beschriebene Software „TrekBuddy“ und „TrackMyJourney“ das Kartenmaterial von dem ***Google Maps Server*** oder dem ***OpenStreetMap Server***, welches nur in einem speziellen Format in die Software importiert werden kann. Zum anderen war das Darstellen von Karten in den restlichen der getestet Freeware- und Open Source Software gar nicht möglich. Bei den getesteten Sharewareprodukten ist das Kartendarstellen vorgesehen. Eine ausführliche Aufstellung dieser Software ist auf der beiliegenden CD in der Excel-Tabelle „SoftwareTest.xls“ zu finden.
Es soll eine Lösung gefunden werden, die das Darstellen von Kartenmaterial der „geoGLIS oHG“ ermöglicht (siehe Kapitel 2.4.1). Die anderen Funktionen (siehe Tabelle: 1) werden aus der erwähnten Open Source - Software bezogen oder sollen zu einem späteren Zeitpunkt in die Software eingearbeitet werden.
Außerdem ist es bei der Eigenentwicklung möglich, die Programmarchitektur selber zu gestalten. Dadurch kann sich innerhalb des selbst geschriebenen Quellcodes besser zu Recht gefunden werden. Außerdem wird die Fehlersuche erleichtert und auftretende Probleme können schneller gefunden und behoben werden.

2.3 Geräte- und Hardwareanalyse

2.3.1 Mobile Endgeräte

Auf dem Elektronikmarkt wird eine große Auswahl an mobilen Endgeräten angeboten. Bei diesen Geräten gibt es große Leistungsunterschiede. Diese Unterschiede reichen von einer minimalen Ausstattung über die Kamerafunktion bis hin zu einer Bluetooth - Schnittstelle oder der GPS-Fähigkeit. Daher bieten mobile Endgeräte ein sehr umfangreiches Einsatzgebiet. Auch die Leistungsfähigkeit dieser Geräte nimmt stetig zu. Neben dem Telefonieren bieten die Geräte unzählige weitere Möglichkeiten, wie etwa das Schreiben von SMS und MMS, das Planen von Terminen mittels Organizer, das Schreiben von E-Mails, das Abspielen von Musik oder das Ausführen „mobiler Dienste“ (z.B. Java-Applikationen). Aufgrund der stetigen Verkleinerung der Hardwarekomponenten der Geräte ist es möglich ein Mobiltelefon und ein PDA (Personal Digital Assistent) in einem Gerät zu vereinen. So haben sich aus dem Standard-Mobiltelefon die multimedialen Mobiltelefone und Smartphones entwickelt. Dadurch werden dem Nutzer neben der Telefonfunktion noch unzählige weitere Funktionen in einem einzigen Gerät bereitgestellt, wie etwa Kalender-, Adress- und Aufgabenverwaltung sowie Text- und Tabellenverarbeitung. Einige Geräte verfügen auch über einen integrierten GPS-Empfänger (z.B. Sony Ericsson W760i, Nokia N96). [fSidI09], [Gmb09a], [fE09a], [Gmb09a]

Daraus ist ersichtlich, dass die Mobiltelefone immer mehr zu einem Multifunktionsgerät werden. Die Tabelle 5 beinhaltet neben einer Auflistung der Funktionen, die die einzelnen mobilen Endgeräte dem Benutzer zur Verfügung stellen, auch die derzeit wichtigsten Geräte auf dem Markt.

Mobilgerät	Funktionen	Bedienung	Netzwerk
Standard-Mobiltelef.	Sprachkom. SMS	10er-Tastatur, Funktionstasten	GSM, HSCSD, GPRS
Multimediales Mobiltelef.	Sprachkom. SMS, MMS Fotos, Vidotel.,Ortung	10er-Tastatur, Funktionstasten	GSM, HSCSD, GPRS, UMTS, Bluetooth
Smartphones	Sprachkom., SMS Organizer, Datendienst	10er- oder Klein-tastatur, Funktionst.	GSM, HSCSD, GPRS, UMTS
PDA	Organizer, Ortung, Branchenlösungen	Stifteingabe Schrifterkennung	WLAN Bluetooth
BlackBerry	Sprachkom., SMS, E-Mail Datendienst, Internet	10er- oder Klein-tastatur, Funktionst.	GSM, GPRS UMTS

Tabelle 5: Funktionsübersicht mobiler Endgeräte
Quelle: [Gmb09a]

Die heutigen mobilen Endgeräte kann man als kleine tragbare Computer bezeichnen. Durch integrierte GPS-Empfänger ist es unter anderem auf PDA’s möglich, dieses durch Navigationssoftware zu erweitern. Der Nachteil dieser Geräte ist jedoch, dass sie die Telefonfunktion nicht unterstützen.

Durch das Aufkommen von Smartphones werden PDA's immer mehr verdrängt, das sie die gleichen Funktionalitäten besitzen. Darüber hinaus wird die Telefonfunktion dieser Geräte unterstützt. Neben diesen Funktionen stellt ein Smartphone alle weiteren Mobiltelefonfunktionen sowie die Funktion des mobilen Internets bereit (GSM, HSCSD, GPRS oder UMTS und HSDPA). Die Smartphones besitzen darüber hinaus fast alle einen integrierten GPS-Empfänger. [Gmb09g]
Neben den Smartphones und PDA's gibt es noch ein weiteres mobiles Endgerät, das ***BlackBerry***. BlackBerry ist ein geschütztes Warenzeichen der kanadischen Firma „Research In Motion" (RIM). Das BlackBerry Client-Server-Konzept läuft auf herkömmlichen Smartphones und besitzt auch dessen Funktionsumfang. Der BlackBerry - Client wurde hauptsächlich zum Schreiben und Empfangen von E-Mails konzipiert. Darüber hinaus wird von den BlackBerry - Geräten das Telefonieren und somit die Internetanbindung ermöglicht. [Gmb09f]
In der Abbildung 4 ist eine Aufstellung über die mobile Endgeräte dargestellt.

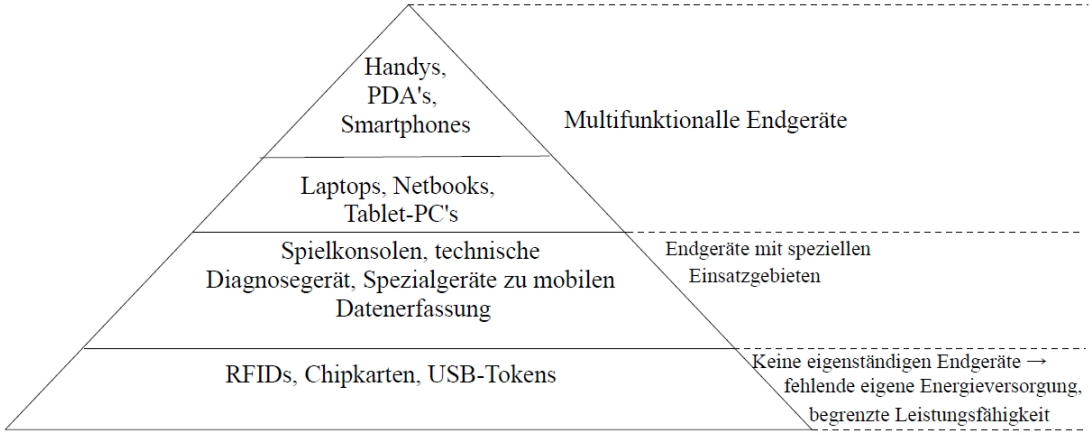


Abbildung 4: Übersicht über mobile Endgeräte
[fSidI09], Seite 6

2.3.2 Betriebssysteme der Geräte

Es soll eine Software gefunden werden, die nach Möglichkeit auf allen im Kapitel 2.3.1 eingegangenen mobilen Endgeräten ausgeführt werden soll, muss nach einer plattformunabhängigen Lösung gesucht werden. Die Systemumgebung kann sich jedoch von Gerät zu Gerät unterscheiden.
Mobile Endgeräte sind mit „Embedded Systems“(eingebettet Systeme) ausgerüstet. Durch solche Systeme ist es möglich, verschiedene Anwendungen abzuarbeiten. Die Hauptaufgabe dieser Systeme ist es, Geräte zu steuern, zu regeln oder zu überwachen. Die „Embeddet Systems“ sind für Geräte mit stark eingeschränkter Hard- und Software sowie einem stark reduzierten Betriebsystem ausgelegt [akt09], [Pla09], [Mar09].
Der grundsätzliche Hardwareaufbau ist bei allen betrachteten mobilen Endgeräten gleich. In der Abbildung 5 ist der allgemeine Aufbau von solch einem Gerät aus hardwareorientierter Sicht dargestellt. Wie aus Abbildung 5 ersichtlich, enthalten mobile Endgeräte im Gegensatz zu einen herkömmlichen PC spezielle Hardwarekomponenten. Diese werden für verschiedene Einsatzzwecke benötigt. Zu diesen Hardwarekomponenten gehören zum Beispiel ein Steckplatz für SD-Karten, Bluetooth - Schnittstelle oder ein GPS-Gerät.

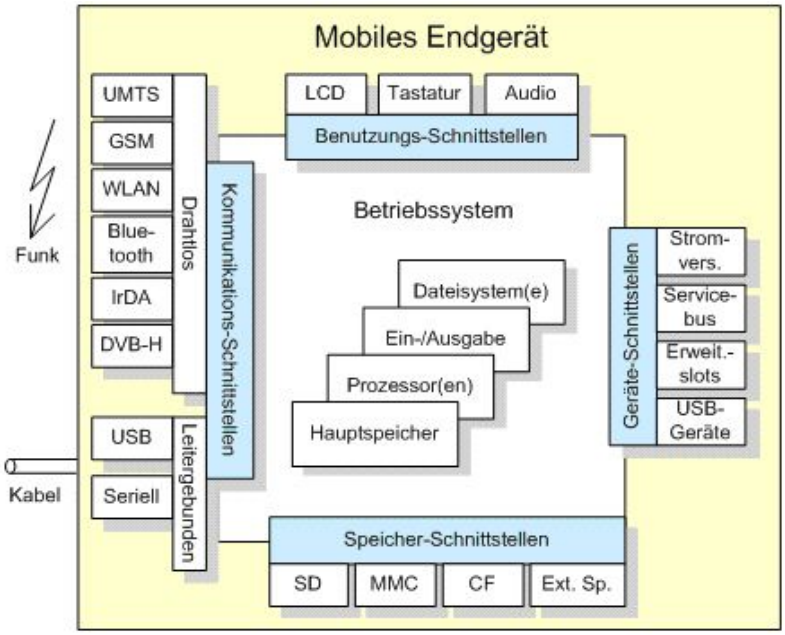


Abbildung 5: Allgemeiner Aufbau eines mobilen Endgerätes
Quelle: [fSidI09]

Für die mobilen Endgeräte wurden spezielle Betriebssysteme entwickelt, die den Anforderungen der Embedded Systems entsprechen. Die meisten Endgeräten enthalten keine Festplatten. Daher müssen die Systeme mit sehr wenig Speicher auskommen. Aufgrund dieser Speichereinschränkungen sowie Energieeinschränkungen und spezielle Benutzerschnittstellen wurden spezielle Betriebssysteme entwickelt. Die bekanntesten Betriebssysteme für mobile Endgeräte sind:

- SymbianOS
- Windows Mobile
- PalmOS
- Linux
- RIM (Reserch in Motion, BlackBerry)

Im Jahr 2006 hatte SymbianOS einen Marktanteil von 67%, bei Windows Mobile lag dieser bei 14%, bei RIM waren es 12% und bei Linux 7% [Mos06]. Wie aus dieser Statistik ersichtlich, verwenden die meisten Hersteller mobiler Endgeräte „SymbianOS“ als Betriebssystem in ihren Geräten. Nachfolgend soll auf die oben genannten Betriebssysteme näher eingegangen werden.

SymbianOS [1]

Das Betriebssystem SymbianOS ist sowohl auf Mobiltelefonen und Smartphones als auch auf PDA's lauffähig. Das Ziel war und ist es, eine einheitliche Betriebsplattform zu schaffen, um so die Geräte untereinander kompatibel zu machen. Dieser Anforderung wird das System gerecht, indem die Hersteller den Source Code allen zugänglich macht.
Außerdem bietet SymbianOS eine hohe Stabilität, ein multitasking- und multithreadfähiges System sowie ein effizientes Speichermanagement. Die Stabilität des Systems wird dadurch gewährleistet, dass jeder Prozess in einem eigenen Adressbereich ausgeführt wird. Aus diesem Grund sind Systemabstürze nahezu ausgeschlossen. Das Speichermanagement zeichnet sich dadurch aus, dass der Arbeitsspeicher (RAM) und der Festspeicher (Flash Memory und Erweiterungskarten) physikalisch unterteilt sind (siehe Abbildung 6).
Weiterhin werden verschiedenen Technologien von SymbianOS unterstützt, wie die Programmiersprachen **Java** und C++. Dadurch ist es möglich eigene Anwendung für das Betriebssystem zu entwickeln. Für die Entwicklung von Java-Applikationen stellt die Firma Sun[Inc08b] die J2ME - Entwicklungsumgebung bereit, worauf in dem Kapitel 4 ausführlich eingegangen wird.

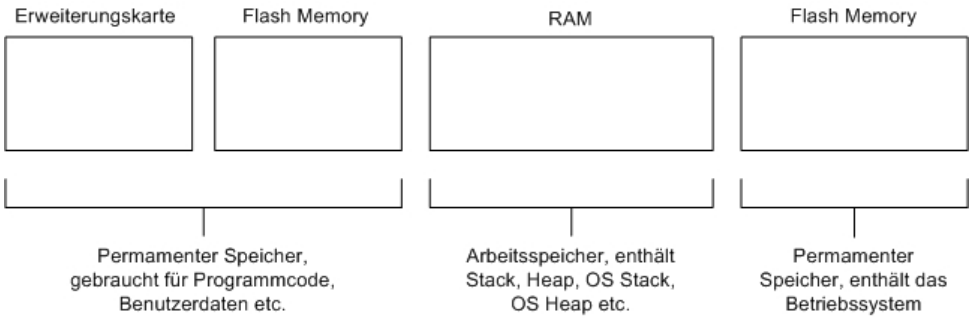


Abbildung 6: Speicheraufteilung bei Symbian OS
Quelle: [Mar09]

[1] Quelle: [Mar09], [wm09], [fSidI09]

Windows Mobile (Windows CE) [2]

Windows Mobile wird auch oft als Windows CE (Consumer Electronics) oder auch als Pocket Windows bezeichnet. Diese Plattform kommt hauptsächlich auf PDA's und Smartphones zum Einsatz.
Bei der Entwicklung wurde darauf geachtet, einige der bekannten Softwareanwendungen aus Windows auf die kleinere Version des Betriebssystems mit zu übernehmen (z.B. Windows Explorer, Pocket-Word, -Excel). Dabei wurde durch die Entwickler versucht, auf die Ressourcenknappheit mobiler Endgeräte zu achten. Jedoch hat das Betriebssystem durch die Treue zur Benutzeroberfläche (Abbildung: 7) und zu den Softwareanwendungen einen viel höheren Ressourcenverbrauch als die anderen betrachteten mobilen Betriebssysteme.

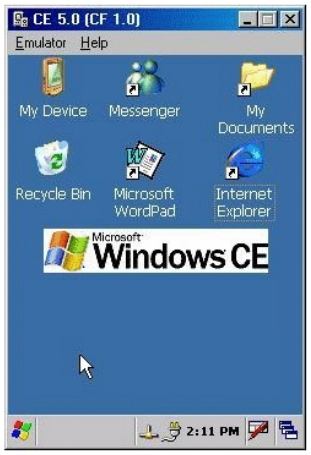


Abbildung 7: Oberfläche von Windows CE 5.0
Quelle: [Gmb09e]

Auch durch die preemptive Multitaskingfähigkeit kann es dem Nutzer beim zeitgleichen Ausführen mehrerer Anwendungen passieren, dass das System deutlich verlangsamt läuft. Bei der Speicherverwaltung setzt Windows Mobile auf ein in zwei Teile geteiltes RAM (Arbeitsspeicher, permanenter Speicher) und den ROM. Durch seine virtuelle Speicherverwaltung besitzt das System zusätzlich noch einen Speicherschutz.
Auch unter diesem Betriebssystem wird es dem Entwickler gestattet, seine eigenen Applikationen zu programmieren. Dafür steht eine Entwicklungsumgebung für C/C++ und dem Visuel BASIC-Bereich zur Verfügung. Neben diesen Hauptprogrammiersprachen für Windows Mobile gibt es auch einen ***Java***-Bereich (PersonalJava-Standard). Dafür wurde eine kleinere Java Version (Java 2 Micro Edition) mit der dazugehörigen Virtuel Machine in das System integriert.

[2] Quelle: [Mar09], [PC09], [Gmb09e], [fSidI09]

PalmOS [3]

Neben SymbianOS ist PalmOS das verbreitetste Betriebssystem auf dem Markt. Das Betriebssystem ist nur auf der Betriebssystemebene Multitasking fähig. Das RAM dient dem System als Arbeitsspeicher wie auch als permanenten Speicher von Daten und Programmen (siehe Abbildung 8).

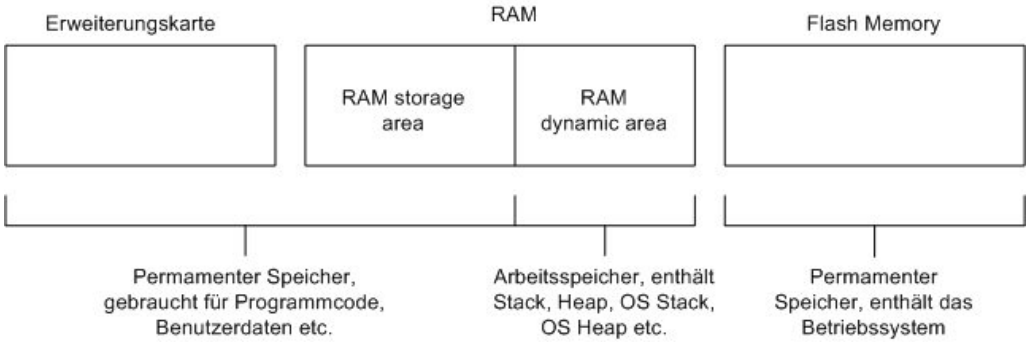


Abbildung 8: Speicheraufteilung beim Palm OS
Quelle: [Mar09], Seite 10

Für PalmOS stehen eine ganze Reihe von Programmiersprachen zur Verfügung, durch die eigene Applikationen für das Systen entwickelt werden können. Dazu gehören unter anderem eine vereinfachte Version von C (Pocket C) und Pascal. Innerhalb der Plattform ist keine Java-Umgebung vorhanden. Durch ein Softwarepaket [Hog09] von Sun [Mah09] kann das System jedoch um die J2ME-Laufzeitumgebung erweitert werden.

RIM (Research in Motion, BlackBerry) [4]

RIM wurde für die BlackBerry - Geräte entwickelt und läuft ausschließlich auf den BlackBerry Geräten. Für die Anwendungen, die das System bereitstellt, wird ein persistenter Speicher zur Verfügung gestellt. Mit der RIM - Plattform ist es möglich, mehrere Passwortgeschützte Benutzerbereiche einzurichten oder den Gerätespeicher zu verschlüsseln.
Die Firma RIM hat neben dem Betriebsystem auch die Softwareumgebung entwickelt. Die Programmiergrundlage für die RIM - Umgebungen bildet **Java** und C++. Dadurch ist es möglich, für dieses Betriebssystem Applikationen selber zu schreiben.

[3] Quelle: [Mar09], [fSidI09]
[4] Quelle: [fSidI09]

Linux [5]

Es gibt einige mobile Betriebssysteme, die auf Linux basieren. Dazu gehört unter anderem Embedix der Firma Lineo. Wie Linux selbst arbeitet es mit einem Linux-Kernel, jedoch in einer stark vereinfachten Version.
Ein weiteres auf Linux basierendes Betriebssystem, eCos, wurde von der Firma Red Hat auf dem Markt gebracht. Dabei handelt es sich um ein Open Source System. Bei diesem Betriebssystem ist besonders hervorzuheben, dass es in der Grundeinstellung extrem wenig Arbeitsspeicher verbraucht (ohne Echtzeitbetrieb).
Aktuell auf dem Markt gekommen ist das auf Linux basierende Open Source Betriebssystem Android von Google. Das System wird der Außenwelt unter der Apache-Lizenz (v2) zugänglich gemacht. Android setzt als Basis auf einen Linux-Kernel 2.6 auf. Dieser verwaltet den Speicher und ist für das Netz, sowie die benötigte Treiber zuständig. Für die Eigenentwicklung von Applikationen in **Java** stellt Android seine eigene Entwicklungsumgebung bereit, „Android SDK“. Im Gegensatz zu den anderen Betriebssystemen arbeitet der Compiler nur mit Java-Klassen die in JDK 5 oder 6 enthalten sind. Aus diesem Grund werden J2ME-Anwendungen von Android nicht unterstützt.

Aus der Betrachtung der einzelnen Betriebssysteme, die für mobile Endgeräte bereitgestellt werden, geht hervor, dass ein System allen Anforderungen die an die Software gestellt werden gerecht wird. Dabei handelt es sich um die Plattform SymbianOS. Aus diesem Grund, wird die Softwareprogrammierung in Zusammenhang mit dieser Arbeit auf dieses Betriebssystem ausgerichtet. Darüber hinaus ist dieses System am häufigsten auf dem Markt vertreten. Außerdem ist in SymbianOS im Gegensatz zu den anderen vier beschriebenen Systemen die J2ME-Laufzeitumgebung bereits integriert.

[5] Quelle: [Mar09], [hDAMS09]

2.3.3 Wahl der Programmiersprache

Aus der Betrachtung der einzelnen mobilen Betriebssysteme geht hervor, dass die Hersteller unterschiedliche Ansätze bei der Systemsoftware wählen. Außerdem wird deutlich, dass die Systeme den Anwendern die Möglichkeit bietet, eigene Applikationen zu entwickeln. Dabei soll darauf geachtet werden, dass die Applikation unter allen mobilen Betriebssystemen funktionsfähig ist, d.h. die Anwendung soll plattformunabhängig sein. Hinzu kommt, dass die Software eine gewisse Größe nicht überschreiten darf.
Aus diesen Gründen wird auf die Programmiersprache Java (Sun Microsystems) zurückgegriffen. Für den mobilen Bereich wurde von Sun eine ressourcensparende Version entwickelt - Java Micro Edition (J2ME). J2ME hat sich aus der Java 2 Plattform heraus gebildet [Esc03]. Aufgrund der Plattformunabhängigkeit von Java und das es sich um eine objektorientierte Programmiersprache handelt, wird diese Sprache für die Entwicklung gewählt.

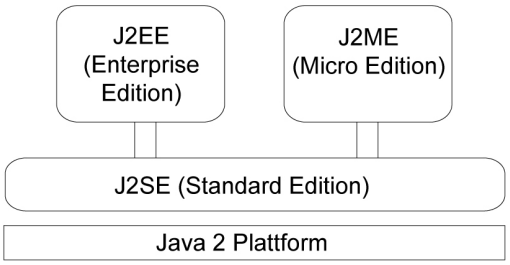


Abbildung 9: Übersicht der Java 2 Plattform.
Quelle: [Esc03]

Dabei ist die J2ME auf Geräte mit limitierten Ressourcen, wie die Einschrängungen in der Displaygröße, dem Speicher und die Prozessorleistung ausgerichtet [Stu08].
Die Laufzeitumgebung Java ME ist bereits auf den meisten mobilen Endgeräten integriert. Durch diese Laufzeitumgebung ist es möglich Java-Anwendungen auf dem mobilen Endgerät zu installieren und auszuführen. Aus diesen Gründen wird im Zusammenhang mit dieser Arbeit eine J2ME-Applikation entwickelt.
Die Ausnahme bildet jedoch das neue Betriebssystem Android. Auf diesem System sind keine J2ME-Anwendungen lauffähig. Aus diesem Grund wird in dieser Arbeit nicht näher auf dieses System eingegangen und bei der Wahl der Programmiersprache nicht berücksichtigt. Ein weiterer Grund dafür ist, dass das Betriebssystem zu neu auf dem Markt ist und es dadurch nicht genügend Erfahrungen durch Entwickler gibt.
Bei der Entwicklung einer Anwendung unter J2ME sind die unterschiedlichen Systeme mit ihren unterschiedlichen Systemanforderungen zu berücksichtigen. Je nachdem, unter welchem Betriebssystem die Applikation laufen soll, müssen unter J2ME spezielle Profile und Konfigurationen angepasst werden.

SymbianOS ist das mobile Betriebssystem, was sich in den vergangenen Jahren auf den Markt immer mehr durchgesetzt hat. Es ist das System, für das die Entwicklungsumgebung für Java ME-Applikationen am meisten ausgereift ist und welches speziell für solche Anwendungen ausgelegt ist. Aus diesen Gründen wird eine Software für mobile Endgeräten mit dem Betriebssystem SymbianOS entwickelt.

Für die Programmierung solcher Java-Applikationen werden spezielle Werkzeuge benötigt,

... das Java SE Development Kit (JDK) [Inc08a]

... das Java ME Wireless Toolkit (WTK) [Inc08b]

... einen beliebigen Editor.

Das JDK enthält die benötigten Komponenten wie den Compiler *javac*, den Interpreter *java*, sowie das Archivierungsprogramm *jar*. Diese Komponenten sind nötig, um eine lauffähige Java-Anwendung zu erzeugen. Das JDK ist nötig, um das Java ME Wireless Toolkit (WTK) benutzen zu können. Das WTK wurde von Sun entwickelt, um den Programmierer von J2ME-Anwendungen eine grafische Benutzerschnittstelle zur Verfügung zu stellen. [Mos06]
Für die Java Entwicklungsumgebung, empfiehlt sich die Eclipse - Plattform [Fou08a]. Durch ein Plug-In lässt sich Eclipse problemlos auf EclipseME erweitern [Fou08b]. In der EclipseME - Umgebung ist es möglich, das WTK einzubinden. Mit dieser Plattform werden alle nötigen Werkzeuge in einem Tool zusammengefasst (Editor, WTK). Darüber hinaus sind noch viele weitere nützliche Optionen und Funktionen, die den Entwicklungsprozess vereinfachen, enthalten.

2.3.4 Standortbestimmung

Für die Darstellung von Kartenmaterial der Umgebung in der sich der Anwender mit seinem mobilen Endgerät zurzeit aufhält, ist es sinnvoll, die aktuelle Position zu bestimmen. Aus diesem Grund soll eine Standortbestimmung ermöglicht werden, um die Position zu ermitteln. Für die Realisierung der Standortbestimmung stehen verschiedene Varianten zur Verfügung. Dazu gehören unter anderem die Positionsbestimmung über Bluetooth, GSM / UMTS oder GPS.
Nachfolgend soll auf die beiden Ortungsverfahren „Ortung im Mobilfunknetz (GSM / UMTS)“ und „Satellitenortung (GPS)“ eingegangen werden. Diese beiden Verfahren werden hauptsächlich bei der Positionsbestimmung von mobilen Endgeräten verwendet.

GSM / UMTS [6]

Bei GSM (Global Systems for Mobile Communication) und UMTS (Universelles Mobil Telecommunications System) handelt es sich um Standards auf dem Gebiet der Mobilfunkkommunikation. Mit diesen Standards wurde eine einheitliche Mobilfunktechnik geschaffen. Diese Systeme werden zur mobilen Sprachübertragung und mobilen Übertragung von Daten (z.B. SMS) verwendet.
Diese Systeme sind in Funkzellen mit jeweils einer Basisstation aufgeteilt. Die Anordnung dieser Zellen ähnelt einer Wabe (zellular).

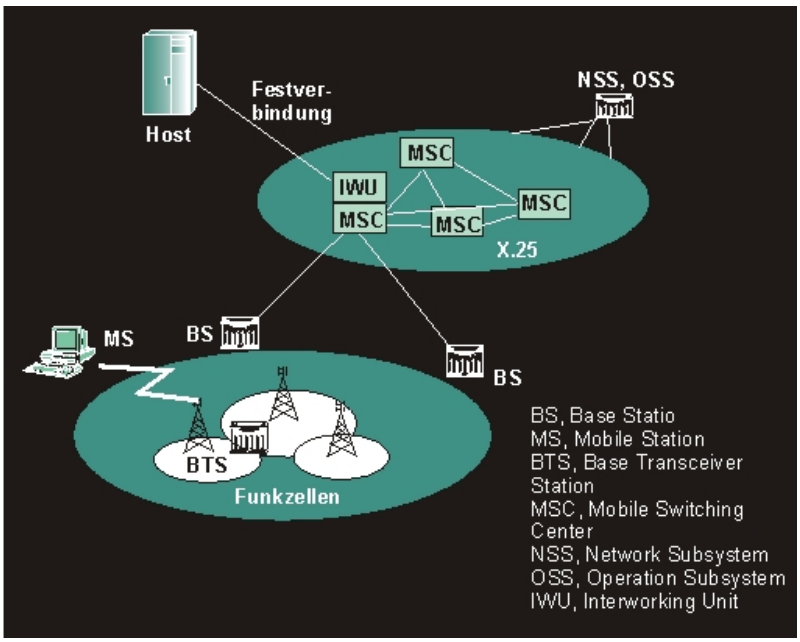


Abbildung 10: Architektur des GSM Netz
Quelle: [uDMW09]

[6] Quellen: [BfU09], [Gmb09d], [Gmb09b], [Kni 6]

Der Aufbau von UMTS ist dem des GSM sehr ähnlich, jedoch wurde die Zellengröße stark verringert. Dadurch wird bei Ortungen innerhalb von UMTS - Systeme eine höhere Genauigkeit und Datenübertragungsrate erreicht.

Positionsbestimmung mittels Mobilfunknetz-gestützter Systeme

Die Standortbestimmung Mobilfunknetz-gestützter Systeme erfolgt mit Hilfe von Informationsübermittlung oder aber durch die Ermittlung der Feldstärke.
Bei der Methode der Informationsübermittlung wird der Standort bestimmt, in dem die Position der - durch den Mobilfunkteilnehmer - genutzten Funkzelle bestimmt wird. Jede dieser Funkzellen enthält eine Zellenidentifikationsnummer (Cell-ID). Diese Cell-ID wird dem Mobilfunkteilnehmer zugeordnet. Die Ermittlung der Position erfolgt über einen Bogenschnitt. Die Genauigkeit dieser Methode ist Abhängig von der Zellengröße. Bei GSM liegt die Genauigkeit zwischen 50 m im städtischen Bereich und 500 m bis 30 km im ländlichen Bereich. Auf Grund kleiner Funkzellen kann mit UMTS eine höhere Genauigkeit (30 m - 50 m) erzielt werden. Jedoch ist UMTS noch nicht überall verfügbar. [Gmb09b]

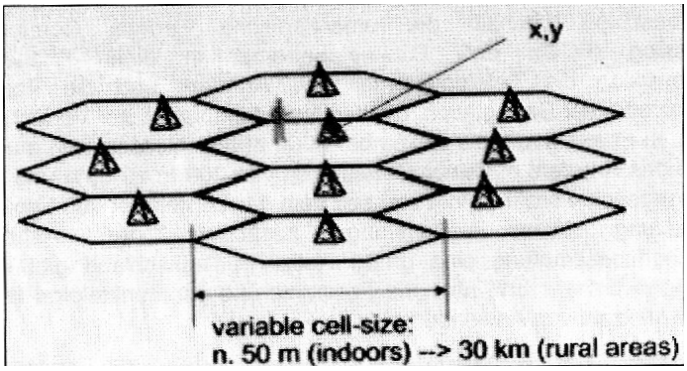


Abbildung 11: Lokalisierung über Funkzellenidentifikation
Quelle: [Kni 6]

Eine weitere Methode zur Positionsbestimmung ist die Messung der Signalstärke (Received Signal Strength - RSS) der Funkzelle. Durch das bestimmen der Felsstärken ist es möglich, die ungefähre Entfernung zum Sender (Basisstation) zu bestimmen. Dies erfolgt per Triangulation.

Im Gegensatz zur Positionsbestimmung mittels GPS (siehe Kapitel 2.3.4) ist die Mobilfunknetz-gestützte Ortung sehr ungenau. Darüber hinaus wird der Zugriff auf die Funkzelleninformation aus der Datenbank der jeweiligen Basisstation von einigen Mobilfunkanbietern nicht gestattet.

Aus diesem Grund wird im Zusammenhang mit dieser Arbeit die Positionsbestimmung mit Hilfe von GPS vorgenommen. Die Ortungsmethode über GSM und UMTS ist jedoch sinnvoll, um einen Mobilfunkteilnehmer für seine momentane Umgebung wichtige oder interessante Informationen zu übermitteln, zum Beispiel den am nächsten gelegenen Flughafen. Dieser Service wird auch als „Location Based Services“ (ortsabhängige Dienste) bezeichnet. In der Regel werden dem Mobilfunkteilnehmer die Daten über ein WAP - Protokoll (Wireless Application Protocol) zur Verfügung gestellt.

***GPS** (Global Positioning System)* [7]

GPS (Global Positioning System) ist ein satellitengestütztes Funknavigationssystem. Mit Hilfe des GPS ist es möglich, die Längen- und Breitengrade sowie die Höhe über dem Meeresspiegel zu bestimmen. Außerdem wurde durch das System ein einheitliches Bezugssystem geschaffen (WGS84). Das System setzt sich aus folgenden drei Segmenten zusammen:

1. **Raumsegment**
   Das Raumsegment wird durch eine Gruppe von Satelliten (min. 24 Satelliten) gebildet, die die Erde umkreisen. Die Abbildung 13 zeigt solch ein Satelliten.
   Die Satelliten wurden auf 6 Bahnen mit je 4 Satelliten angeordnet. Die Bahnhöhe beträgt ungefähr 20.000 km. Die Inklination (Bahnneigung) beträgt 55°. Durch die Anordnung der Satellitenbahnen wird gewährleistet, dass die Nutzung von mindestens 4 Satelliten von jedem Standpunkt aus und zu jeder beliebigen Zeit möglich ist. In der Abbildung 12 ist dieses Raumsegment anhand der GPS-Konstellation dargestellt.

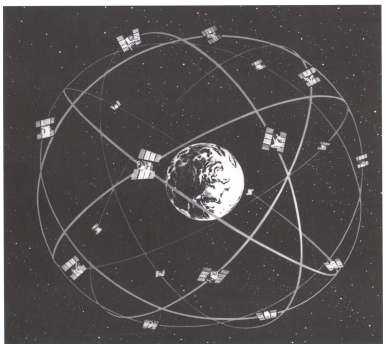


Abbildung 12: GPS-Konstellation
Quelle: [Bauge]

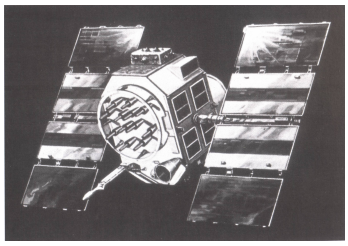


Abbildung 13: GPS-Satellit
Quelle: [Bauge]

2. **Kontrollsegment**
   Das Kontrollsegment bildet den Hauptbestandteil des GPS-Systems. Mit Hilfe dieses Segmentes werden die Satelliten beobachtet und verschiedene Einstellungen vorgenommen (u.a. atmosphärische Korrekturen, Synchronisation der Uhren).

[7] Quellen: [Pet], [Kni 6], [Bauge], [Web09a], [Web09b]

3. **Nutzersegment**
   Das Nutzersegment wird aus dem Nutzer und einem GPS-Empfänger gebildet. Diese Empfänger können je nach Einsatzgebiet stark variieren. Ein Vermesser benötigt aufgrund der Genauigkeit ein anderes GPS-Gerät, wie etwa ein Wanderer oder Radfahrer für eine einfache Positionsbestimmung mit Hilfe eines mobilen Endgerät (Mobiltelefon, Smartphone, PDA, BlackBerry). Aus diesem Grund werden in dieser Arbeit hauptsächlich GPS-Mäuse oder interne GPS-Geräte in Betracht gezogen.

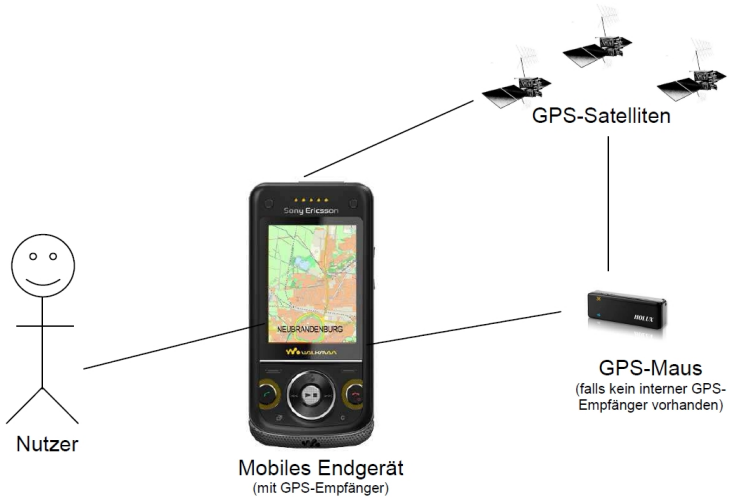


Abbildung 14: Darstellung des Nutzersegment
mit den verschiedenen Schnittstellen
Quellen für Bilder: [ube09],[Kni 6]

Positionbestimmung mittels GPS

Um mit einem GPS-Gerät eine Position zu bestimmen, muss zwischen diesem Empfänger und den Satelliten Sichtkontakt bestehen. Für eine Standortbestimmung mit der entsprechenden Höhenangabe, sind mindestens 3 Satelliten erforderlich.
Jeder Satellit sendet kontinuierlich 2 Signale (Trägerfrequenzen L1 = 1,57542 GHz und L2 = 1,2276 GHz) aus. Diese Signale sind mit den Werten 0 und 1 (Binärcode) belegt. Aus diesen Signalen gehen unter anderem folgende Daten hervor: die Satellitenposition, die Satellitenbahndaten, der Zeitpunkt der Aussendung des Signals.
Für eine Positionsbestimmung (siehe Abbildung 15) werden Informationen zur Signallaufzeit und zur Satellitenposition zu einem GPS-Empfänger auf der Erde benötigt. Die Signallaufzeit wird durch die Entfernungsmessung zwischen Empfänger und Satellit bestimmt. Dafür wird die Differenz zwischen dem Zeitpunkt der Aussendung beim Satelliten und dem Zeitpunkt des Empfangs bei einem GPS-Empfänger ermittelt. Der sich aus Berechnungen (räumlicher Bogenschnitt) ergebene Standort auf der Erde wird durch WGS84 - Koordinaten beschrieben.

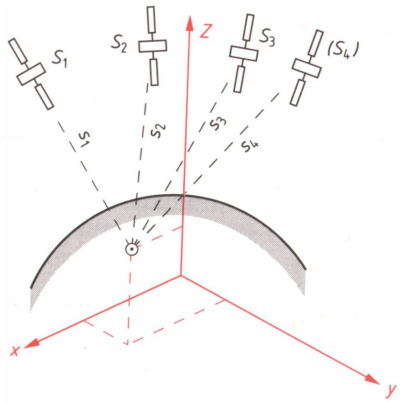


Abbildung 15: Punktbestimmung mit drei bzw vier Satelliten
Quelle: [Pet]

Bei der Verwendung von GPS-Mäusen und anderen displaylosen GPS-Empfängern handelt es sich beim Empfangen von GPS-Daten um **NMEA**-Datensätze (National Marine Electronics Association). Welche verschiedenen Informationen aus den NMEA - Datensätze bezogen werden können, wird in der Abbildung 16 veranschaulicht. Dazu gehören unter anderem die Weltzeit, die Datumsangabe, die Positionsangabe (Längen- und Breitengraden) sowie die Geschwindigkeit und die Richtung. [Gmb09c],[uDMW09]

```
$GPRMC,191410,A,4735.5634,N,00739.3538,E,0.0,0.0,181102,0.4,E,A*19
       |      | |           |           |   |   |      |     | |
       |      | |           |           |   |   |      |     | Prüfsumme
       |      | |           |           |   |   |      |     |
       |      | |           |           |   |   |      |     Modus (A,D,E,N,S)
       |      | |           |           |   |   |      |
       |      | |           |           |   |   |      magnetische
       |      | |           |           |   |   |      Deklination 0.4° E
       |      | |           |           |   |   |
       |      | |           |           |   |   Datum: 18.11.02
       |      | |           |           |   |
       |      | |           |           |   wahrer Kurs (ohne Bewegung 0)
       |      | |           |           |
       |      | |           |           Geschwindigkeit über Grund (Knoten)
       |      | |           |
       |      | |           007° 39.3538' östliche Länge
       |      | |
       |      | 47° 35.5634' nördliche Breite
       |      |
       |      Empfängerwarnung, A = Daten OK, V = Warnung
       |
Uhrzeit der Positionsbestimmung: 19:14:10 UTC-Zeit
```

Abbildung 16: Ausgabebeispiel von NMEA - Daten
GPRMC-Datensatz (RMC = recommended minimum sentence C)
empfohlener Minimumdatensatz (Quelle: [uDMW09])

Die Genauigkeit der GPS-Daten ist von verschiedenen Faktoren abhängig: Ausrichtung der GPS-Antenne, Abweichungsfehler oder freigeschaltetem Service. Abweichungsfehler der GPS-Daten können durch fehlerhaft eingestellte Satelliten oder auch durch Störeinflüsse der Ionosphäre herbeigeführt werden. Damit beim Transport des GPS-Empfängers keine größeren Schwankungen bei den GPS-Daten entstehen, sollte auf die Ausrichtung der Antenne und die Platzierung des Gerätes geachtet werden. Die Genauigkeiten der GPS-Position liegen im Normalfall, bei einer Abweichung um die 15 m.

GPS-Empfänger

Wie bereits erwähnt, kommen im Zusammenhang mit dieser Arbeit zwei verschiedene Arten von GPS-Geräten zum Einsatz. Dabei handelt es sich zum einen um sogenannte „GPS-Mäuse“ und zum anderen um ein internes GPS-Gerät.
Beide Empfängerarten haben gemeinsam, dass sie im Kern ein GPS-Modul enthalten. Die Grundbestandteile eines solchen GPS-Moduls sind ein GPS-Chip, ein Antennen-Chip, ein UART-Chip (serielle Schnittstelle) und ein Kondensator. [Web09a], [Web09b]
Bei mobilen Endgeräten werden die GPS-Module im Gehäuse integriert. Dadurch ist es möglich das mobile Endgerät mit seinen Standardfunktionen (siehe Tabelle 5) zu nutzen oder als GPS-Empfänger zu verwendet.

Bei einer GPS-Maus handelt es sich meist um kleine displaylose Geräte, die nur für den Empfang von GPS-Daten ausgelegt sind. Über eine Bluetooth - Schnittstelle werden diese Daten zu einen mobilen Endgerät übermittelt. Für die Stromversorgung ist in den GPS-Mäusen ein Akkumulator enthalten. Zum Laden des Akkumulators dient eine USB - Schnittstelle. [fE09d]

Abbildung 17: Beispiel für eine GPS-Maus
HOLUX GPSlim GR 240 Bluetooth Receiver (Quelle: [ube09])

Das Mobiltelefon kann zur Standortbestimmung mit einer Bluetooth GPS-Antenne verbunden werden. Ein Beispiel für eine solche GPS-Antenne ist der *HOLUX GPSlim GR 240 Bluetooth Receiver*, wie in Abbildung 17 dargestellt.

2.4 Geodaten

Die Verwendung von Geodaten wird dadurch begründet, da mit deren Hilfe ein Bezug zur realen Welt hergestellt werden kann. Es ist mit diesen Daten möglich eine Aussage darüber zu treffen, wo sich der aktuelle Standort des mobilen Endgerätes (GPS-Koordinaten) oder die Lage der Karte (Begrenzung des Kartenausschnittes durch Koordinaten) auf der Erde befindet. [Fri94]

2.4.1 Welche Daten sollen dargestellt werden?

Wie bereits in den vorherigen Kapiteln erwähnt, sollen den Anwendern innerhalb der Software verschiedene Daten zur Verfügung gestellt werden. Neben der Darstellung von Daten soll es dem Anwender auch gestattet werden, eigene Daten zu erfassen. Dabei handelt es sich um folgende Geodaten: GPS-Position, Karten, Waypoints, POI's, Tracking-Daten. Nachfolgend wird auf diese Daten näher eingegangen.

Karte

Wer die üblichen Straßen und Wege verlassen will, muss in der Regel auf eine Papierkarte und einen Kompass zurückgreifen. Für fast jeden Bereich gibt es solche Karten (Wanderkarten, Kanukarten, Radkarten), die es ermöglichen, sich abseits der Hauptverkehrswege zu orientieren. Mit der Software soll eine Möglichkeit geboten werden, auf Papierkarte und Kompass zu verzichten. Die Lösung ist die Darstellung digitaler georeferenzierter Karten auf einem mobilen Endgerät. Die Positionsbestimmung erfolgt mit Hilfe eines GPS-Empfängers, statt mit einer Papierkarte. Das Kartenmaterial muss den Anforderungen, wie an die speziellen Papierkarten, gerecht werden. Aus diesem Grund muss die Wahl des Kartenmaterials gründlich durchdacht sein.
Eine vektorielle Darstellung wird nicht in Betracht gezogen. Die Vektorkarten finden hauptsächlich ihren Einsatz in der Navigation, da sie für die Routenberechnung besonders gut geeignet sind. Es wird mit der entwickelten Software keine Navigation von A nach B angestrebt, so ist die Verwendung von Vektordaten überflüssig. Die Vektorkarten liegen einer Datenbank zugrunde, aus der diese generiert werden. Für die Routenberechnung wird auf die in der Datenbank enthaltenen Daten zugegriffen. Vektorkarten beziehen sich aufgrund ihres Einsatzgebietes hauptsächlich auf die Hauptverkehrswege. [RD08]
Die Unterschiede zwischen Raster- und Vektordaten, sind in der Tabelle 6 dargestellt.
Der Hauptgrund für die Verwendung von Rasterkarten der „geoGLIS oHG“ ist die Beschaffung und Darstellung der zugrunde liegenden geografischen Vektordaten. Die Beschaffung von Vektordaten ist eine sehr aufwändige und kostenintensive Aufgabe, da es sich bei diesen Daten um behördliche Daten handelt und diese nicht frei verfügbar sind.

	Rasterdaten	Vektordaten
Allgemein	- Realweltbeschreibung durch Flächen - beziehen sich dirket auf Fläche - Grundelement: Pixel - Datenerfassung: einfach und schnell	- Realweltbeschreibung durch Punkte und Linien - Grundelemente: Punkte, Linien - geschlossenen Linie definiert Fläche - geometrische Darstellung durch Lage der Knoten und Form der Linie >> bilden Grundlage für Routing
Vorteile	- Meist schönes, gut ablesbares Kartenbild. - blattschnittfrei - Tracks können direkt auf Karte erzeugt werden	- Sehr hohe Genauigkeit möglich - Oft Routingfähig - verschiedene Maßstabsanzeigen - Weniger speicherplatzaufwändig als Rasterkarten
Nachteile	- Nicht routingfähig - Kein einheitliches, standardisiertes Format	- Kartenbild meist unansehnlich - Kein einheitliches, standardisiertes Format - geografische Vektordaten, sehr kostenintensiv

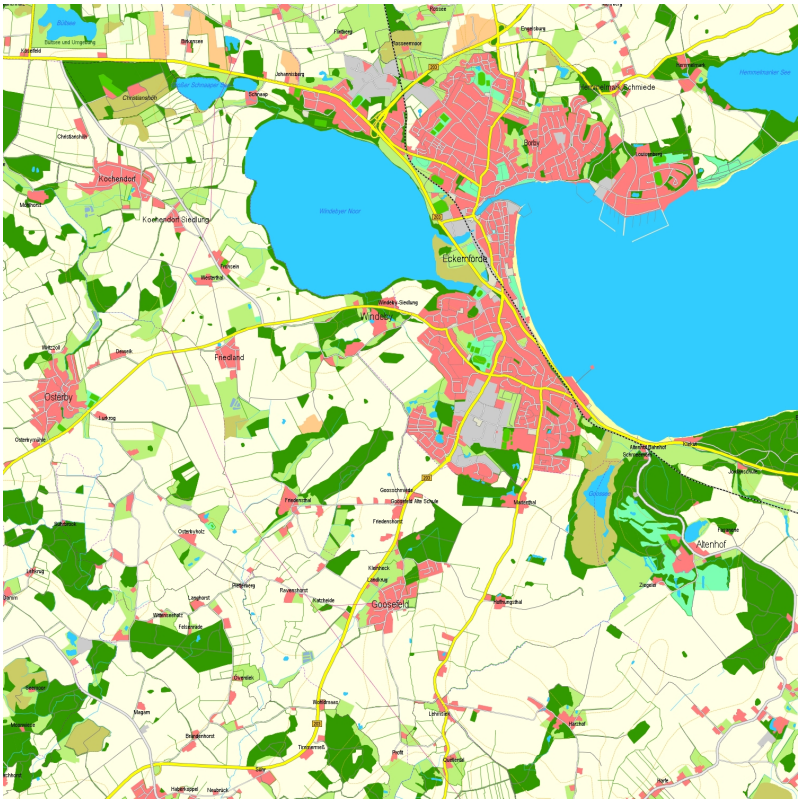
Tabelle 6: Gegenüberstellung: Rasterdaten und Vektordaten
Quellen: [rw09a], [Fri94], [rw09b]

Die auf der Webseite [go09b] zum Download bereitstehenden Rasterkarten der „geoGLIS oHG“ beruhen auf behördlichen Vektordaten - den ATKIS - Daten [dVdLdBD09]. Der ATKIS[8] - Datenbestand wird von den Landesvermessungsämtern bereitgestellt. Die Ämter gewinnen diese Daten unter anderem durch Digitalisierungen der Topografischen Karten (TK25, TK50, TK10). Jedoch werden heutzutage die entsprechenden Daten im Gelände in digitaler Form durch die Vermesser aufgenommen und können dadurch gleich weiterverarbeitet werden. Diese gewonnenen topografischen Geobasisdaten dienen dazu, alles auf einen einheitlichen Datenbestand und Raumbezug (Gauß-Krüger Koordinatensystem) zu bringen. Auf Bundesebene gibt es das BKG (Bundesamt für Kartographie und Geodäsie) [fKuG09], welches die Daten pflegt und koordiniert. [Sxh09], [R.S09]
Aus diesem Grund sind diese Kartenkacheln detaillierter und genauer als freies Kartenmaterial, wie etwa „Google Maps“[Deu09] oder „OpenStreetMap“ [Ope09a]. Dadurch sind in dem Kartenmaterial der „geoGLIS oHG“ (siehe Abbildung 18), im Gegensatz zu dem freien Kartenmaterial, viele Wald- und Forstwege sowie Feld- und Wanderwege enthalten.

PNG-Format [9]

PNG (Portable Network Graphic) ist ein Grafikformat für Rastergrafiken. Es hat den Vorteil, dass es weniger Komplex ist, als das TIFF - Format. Hinzukommt, dass das PNG - Format speziell für das World Wide Web entworfen wurde. Es ist ein vom „World Wide Web Consortium“ (W3C) anerkanntes Format.
Im Gegensatz zu anderen Grafikformaten bietet dieses Format für größere Grafiken eine deutlich bessere Komprimierung. Außerdem ist ist eine deutlich bessere Datenübertragung

[8] ATKIS = Amtliches Topographisch-Kartographisches Informationssystem
[9] Quellen: [Gun09],[Völ09]

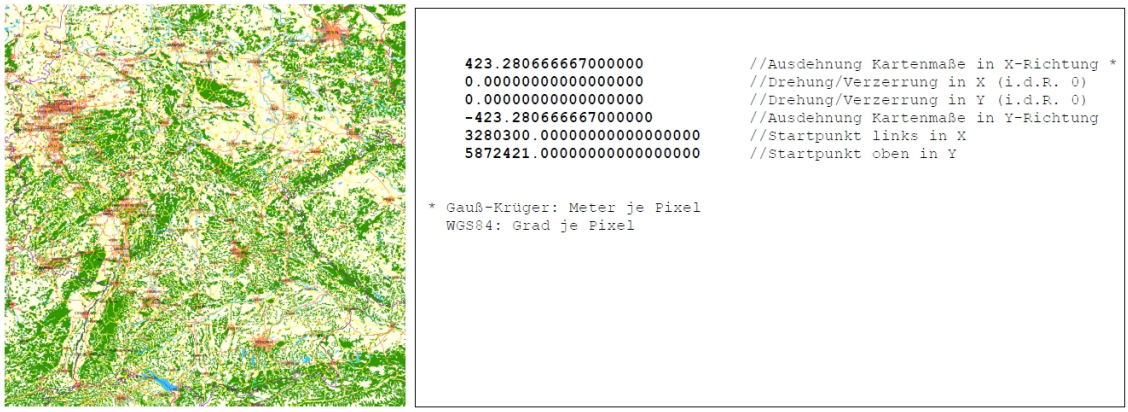


ATKIS (R) Basis-DLM, (C) Vermessungsverwaltungen der Länder und BKG 2005

Abbildung 18: Rasterkarte der geoGLIS oHG

möglich. Darüber hinaus bietet PNG die verlustfreie Darstellung sowohl von Echtzeitbildern im RGB - Farbmodell als auch von palettenbasierten Bildern. Außerdem werden durch das PNG - Format die Transparenz (Alphakanal) als auch Schattierungen unterstützt.

Die bei georeferenzierten Rasterkarte dazugehörige PGW - Datei enthält wichtige Information zu der entsprechenden Kachel. Die in der PGW - Datei enthaltenen Daten geben darüber Auskunft, welche Dimension ein Bildpixel in X- und Y-Richtung, Drehung, Verzerrung die Kartenkachel besitzt. Außerdem enthält die Datei die Startkoordinaten (x und y Koordinate) der Kartenkachel. Der Startpunkt befindet sich in der linken oberen Ecke. In der nachfolgenden Abbildung 19 wird eine georeferenzierte Rasterkarte mit der dazugehörigen PGW - Datei dargestellt.



```
423.280666667000000          //Ausdehnung Kartenmaße in X-Richtung *
0.00000000000000000          //Drehung/Verzerrung in X (i.d.R. 0)
0.00000000000000000          //Drehung/Verzerrung in Y (i.d.R. 0)
-423.280666667000000         //Ausdehnung Kartenmaße in Y-Richtung
3280300.00000000000000000    //Startpunkt links in X
5872421.00000000000000000    //Startpunkt oben in Y


* Gauß-Krüger: Meter je Pixel
  WGS84: Grad je Pixel
```

3280300.5237500.png 3280300.5237500.pgw

ATKIS (R) Basis-DLM, (C) Vermessungsverwaltungen der Länder und BKG 2005

Abbildung 19: PNG-Grafik mit der dazu gehörigen PGW-Datei

POI's

Neben der Kartendarstellung soll der Nutzer die Möglichkeit erhalten, weitere nützliche Informationen einzublenden. Dazu gehören unter anderem die POI's. Diese POI's können auf Wunsch durch den Nutzer der Software eingeblendet, bzw. ausgeblendet werden.
In der Regel werden POI's durch kleine Symbole dargestellt. Die Informationen für die POI's werden tabellenförmig in einer Datei abgelegt.
In der Abbildung 20 sind Beispiele von POI's dargestellt.

ATKIS (R) Basis-DLM, (C) Vermessungsverwaltungen der Länder und BKG 2005

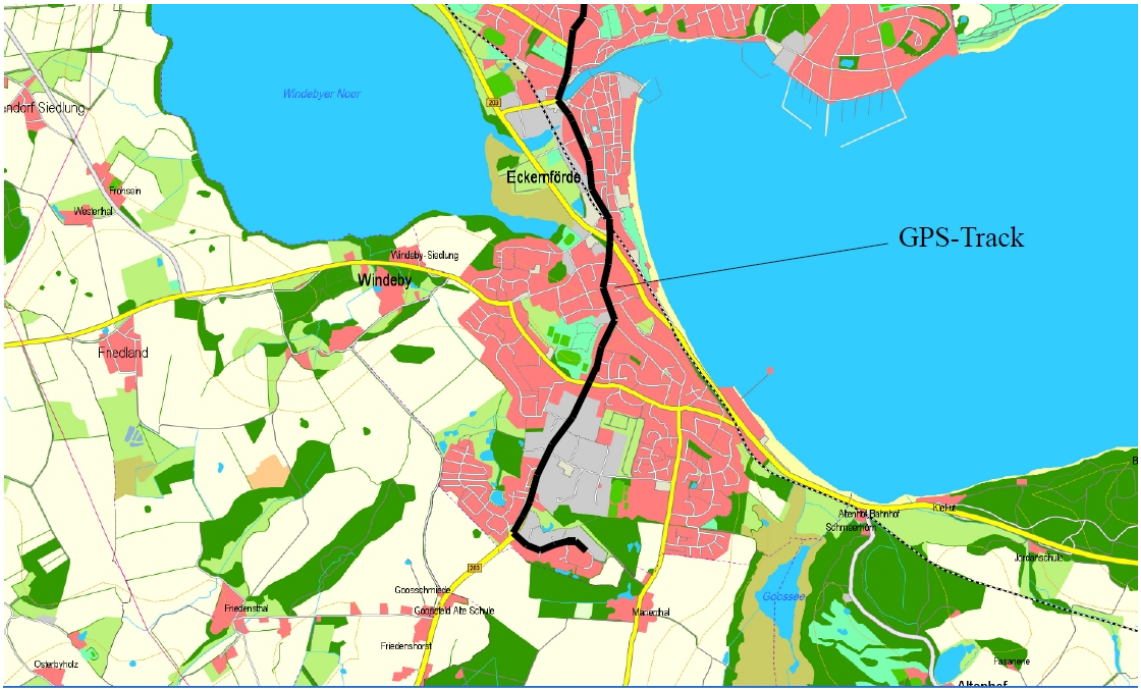
Abbildung 20: Darstellung von POI's auf der Karte

Waypoints

Weiterhin soll dem Nutzer die Möglichkeit geboten werden, Waypoints (Wegpunkte) manuell zu setzen. Dadurch ist, wie bereits erwähnt, das Markieren nutzerspezifischer Punkte möglich. Neben den Längen- und Breitengraden können noch zusätzliche Daten zum jeweiligen Punkt abgelegt werden wie etwa die Höhe oder die Geschwindigkeit.
Wie auch bei den POI's werden diese Punkte durch kleine Symbole auf der Karte gekennzeichnet. Auch diese Daten werden in einer Datei hinterlegt.

Tracking-Daten

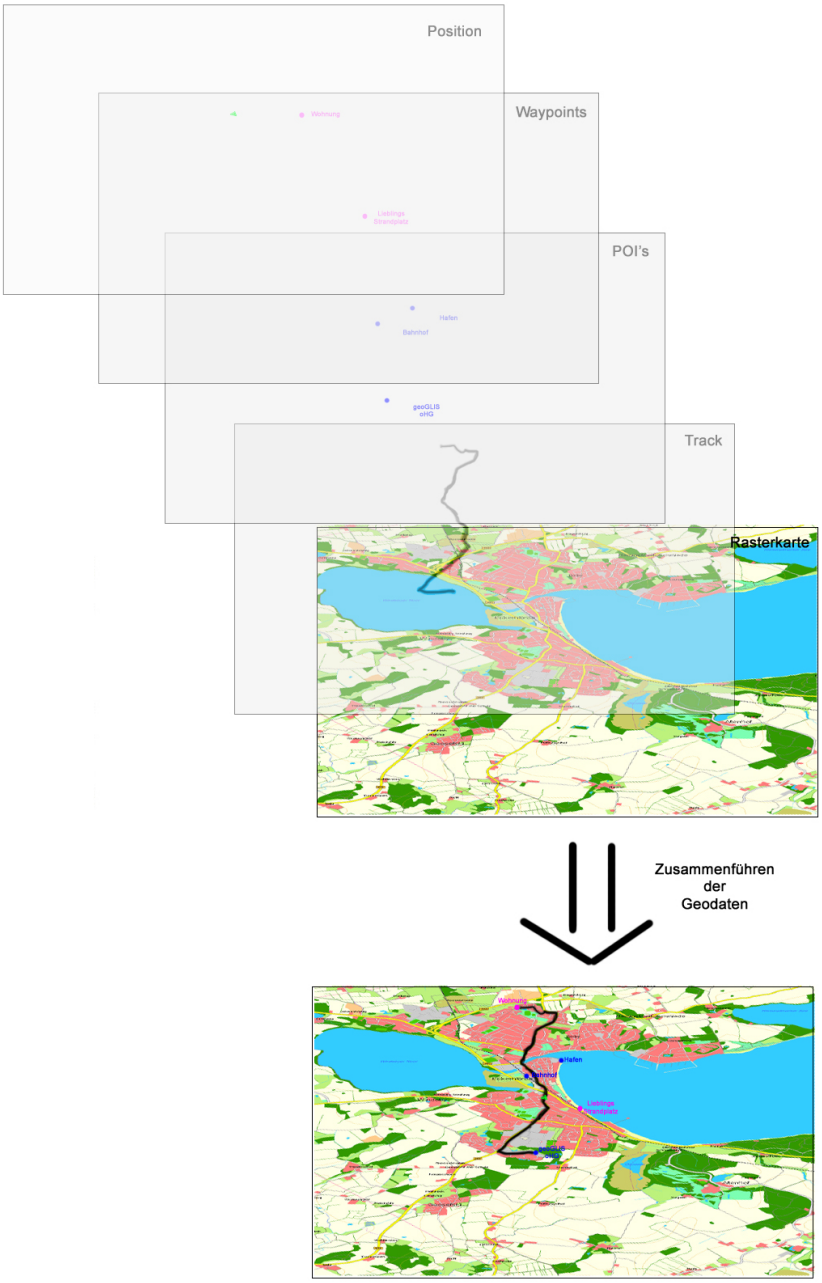
Unter Tracking-Daten versteht man die Daten, die zum Beispiel bei einer Wanderung aufgezeichnet werden. Durch die eingebaute Tracking - Funktion ist das Aufzeichnen der zurückgelegten Strecke möglich. Das Starten der Aufzeichnung erfolgt manuell. Die Aufzeichnung selbst geschieht automatisch.
Solche protokollierten Strecken werden auch als „GPS - Tracks“ bezeichnet. Ein Track besteht aus mehreren aufeinander folgenden geografischen Koordinaten. Neben den Längen- und Breitengraden wird auch die genaue Uhrzeit festgehalten. Dadurch kann im Nachhinein festgestellt werden, wo man sich zu welchem Zeitpunkt aufgehalten hat. Daneben können auch Informationen wie Geschwindigkeit und Richtung zu den jeweiligen Punkten gespeichert werden. In der Regel werden Trackpunkte alle 1 bis 20 Sekunden oder in einem Abstand von 1 bis 200 m aufgezeichnet und gespeichert.
Ein GPS - Track darf nicht mit einer Route verwechselt werden. Solch eine Route ist im Gegensatz zu einem Track eine geplante Strecke, die bereits im Vorfeld erstellt wird [RD08]. Jedoch werden beide durch eine Linie dargestellt. Durch das Aufzeichnen der zurückgelegten Strecke wird erreicht, dass diese im Nachhinein rekonstruiert werden können. So ist es möglich, zu einem späteren Zeitpunkt die zurückgelegte Strecke zu analysieren und darzustellen.
Für die Visualisierung solcher Tracking-Daten gibt es verschiedene Analyse-Programme (z.B.GPS-Track-Analyse.NET [gf09]) und eine Vielzahl von Internetportalen [GPS09]. Auf solchen Portalen ist es möglich, die Daten hochzuladen und die Tracks nachzubearbeiten (löschen, verschieben, einfügen weiterer Trackpunkte). Zur visuellen Unterstützung dienen Karten im Hintergrund. Bei GPSies.com [GPS09] handelt es sich dabei um Schnittstellen zu einer Google- und einer OpenStreetMap-API.
Solch ein GPS-Track wird in der Abbildung 21 angezeigt.



ATKIS (R) Basis-DLM, (C) Vermessungsverwaltungen der Länder und BKG 2005

Abbildung 21: Darstellung eines GPS-Tracks auf der Karte

Bei der Einbindung der GPS-Daten durch den Nutzer soll es möglich sein, diese gleichzeitig darzustellen. Das wird dadurch erreicht, dass die einzelnen Geodaten überlagert werden. Die Abbildung 22 zeigt die verschiedenen Layer, in denen die entsprechenden Daten enthalten sind. Bei der Darstellung wird die Karte immer in den Hintergrund gelegt. Dadurch ist die visuelle Unterlegung der übrigen Geodaten gewährleistet.



ATKIS (R) Basis-DLM, (C) Vermessungsverwaltungen der Länder und BKG 2005

Abbildung 22: Überlagerung der Geodaten

Dateiformate für POI's, Waypoints, Tracking-Daten

Damit das Austauschen zwischen verschiedenen Anwendungen (z.B. Java-Applikation) und den unterschiedlichen Internetportalen [GPS09] möglich ist, werden verschiedene GPS-Dateiformate entwickelt (z.B.ozi, .osm, .csv, .loc, .kml, .gpx). Diese Dateiformate ermöglichen das Speichern der Geodaten. Das derzeit am häufigsten verwendete Format ist das GPX - Dateiformat.

GPX-Dateiformat (GPS Exchange Format) [10]

Es ist ein offenes lizenzfreies Format, dessen Verwendung gebührenfrei ist. Die Unterstützung des Dateiformates wird durch die meisten GPS-Geräte und -Anwendungen gewährleistet. Dieses Format basiert auf XML - Standards. Dabei handelt es sich jedoch um eine minimierte Version von XML. Die Grundstruktur des GPX - Formates lässt sich wie folgt beschreiben Abbildung 23:

```
<gpx version="1.1 [1] ?" creator="xsd:string [1] ?">
      <metadata> metadataType </metadata> [0..1] ?
      <wpt> wptType </wpt> [0..*] ?
      <rte> rteType </rte> [0..*] ?
      <trk> trkType </trk> [0..*] ?
      <extensions> extensionsType </extensions> [0..1] ?
</gpx>
```

Abbildung 23: Grundstruktur des GPX-Dateiformat
Quelle: [Fos09a]

KML-Dateiformat (Keyhole Markup Language)[11]

KML dient, wie das GPX - Dateiformat, für die Darstellung geografischer Daten. Die verwendete Tag-Struktur enthält verschachtelte Elemente und Attribute. Genauso wie das GPX - Dateiformat verwendet KML den XML - Standard.
Das Dateiformat wurde als Austauschformat für die Client-Komponente von Google Earth entwickelt. In der Abbildung 24 wird die KML - Struktur anhand eines Beispiels dargestellt.

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://earth.google.com/kml/2.1">
      <Placemark>
            <name>Stonehenge, England</name>
            <description>Stonehenge wurde um 2500 v. Chr. erbaut</description>
            <Point>
                  <coordinates>1.826752,51.179045,0 </coordinates>
            </Point>
      </Placemark>
</kml>
```

Abbildung 24: Beispiel für ein Placemark-Element innerhalb eines KML Dokuments
Quelle: [Ear09]

[10] Quellen: [Fos09b], [fE09b]
[11] [Ear09]

2.4.2 Datenbeschaffung

Wie bereits erwähnt können mit Hilfe der Software verschiedene GPS-Daten importiert und dargestellt werden. Für den Import der Daten in die Software, müssen diese auf dem mobilen Endgerät vorhanden sein. Dafür stehen dem Nutzer bei der Datenbeschaffung zwei verschiedene Varianten zur Verfügung. Dabei handelt es sich zum einen um das offline Laden der Daten. Mit Hilfe dieser Variante werden die Daten durch den Nutzer im Vorfeld auf einem herkömmlichen PCs heruntergeladen und per Datenkabel auf das mobile Endgerät übertragen. Zum anderen handelt es sich um die Variante des online Ladens der GPS-Daten. Bei dieser Variante greift der Nutzer direkt auf das Internet über das mobile Endgerät zu, um die entsprechenden Daten zu beziehen.
Die Internetverbindung zu einem mobilen Endgerät wird über die Luftschnittstelle zur Verfügung gestellt. Aufgrund der Einschränkungen, denen die mobilen Endgeräte unterliegen, können die normalen HTML-Seiten nicht korrekt dargestellt werden. Aus diesem Grund wurde ein neuer Standard geschaffen, das Wireless Application Protocol (WAP). WAP stellt ein eigenes Dokumentenformat bereit, das WML (Wireless Markup Language). Es ist eine auf XML - basierende Sprache, die auch auf verschiedene HTML-Mechanismen zurückgreift. Für die Internetanbindung wird durch WAP das Wireless Session Protocol (WSP) zur Verfügung gestellt. Für die Datenübertragung wurden die Anzahl und das Volumen der Nachrichten reduziert, um den Ressourcenverbrauch auf mobilen Endgeräten so gering wie möglich zu halten [Tot09].

Beschaffen von POI's, Waypoints und Tracks

offline Laden
Wie bereits erwähnt, ist es bei dieser Variante der Datenbeschaffung erforderlich, das der Nutzer die Daten im Voraus über einen herkömmlichen PC (mit Internetanschluss) bezieht. Diese Daten werden durch den Nutzer über eine Datenschnittstelle auf dem mobilen Endgerät abgelegt.
Für das Laden der Geodaten, stehen dem Anwender eine Reihe von Internetportalen zur Verfügung. Unter „GPSies.com“[GPS09] stehen dem Anwender eine große Auswahl an GPS-Tracks zur Auswahl. Neben der Downloadfunktion erhält der Nutzer auf diesem Portal zusätzlich die Möglichkeit, Tracks in unterschiedliche Formate zu konvertieren sowie das Hochladen und das Bearbeiten eigener Tracks.
Ein weiteres großes Internetportal ist „NaviFriends.com“[Nav09]. Dieses Portal bietet eine sehr große POI - Datenbank. Es ist jedoch eine Registrierung durch den Nutzer erforderlich, um an die entsprechenden Daten zu gelangen. Die meisten POI's, die zum Downloaden stehen, werden in dem ASCII - Format zu Verfügung gestellt. Um diese in Daten in das entsprechende Dateiformat umzuwandeln, gibt es zum Beispiel einen POI - Converter [Con09b]. Ein weiteres Portal, das POI's zum Downloaden anbieten, ist my-poi.info. Von dieser Seite können die POI-Daten auch ohne Registrierung bezogen werden.
Solche Internetportale beruhen hauptsächlich auf gegenseitigem Austauschen von GPS-

Daten. Aus diesem Grund ist es sinnvoll, die selbst aufgenommen Daten auch anderen Nutzern bereit zu stellen.

online Laden

Bei dieser Variante ist es notwendig die vorhandenen Internetportale für die Darstellung auf mobile Browser anzupassen (WML). Dies ist jedoch zurzeit für die meisten Websites noch nicht realisiert.
Aus diesem Grund wird auf den Location Based Service (LBS), positionsabhängige Dienste, zugegriffen. Auch dieser Dienst wird über das WAP bereitgestellt. Mit diesem Service ist das Hinzuladen von standortbezogenen Daten möglich. Dadurch können wichtige Standorte die sich in der unmittelbaren Umgebung befinden, dargestellt werden. Location Based Services können außerdem Informationen zum Verkehr, Telefonbucheinträge, Kino- oder Theaterprogramm oder das Wetter enthalten.
Durch solch einen Service kann gewährleistet werden, dass dem Nutzer ständig Aktuelle Daten bereitgestellt werden können.

Daten aufnehmen

Die Daten (Waypoints und Tracking-Daten) können außerdem beschaffen werden, indem sie durch den Nutzer selber aufgenommen und gesammelt werden. Dadurch ist das erneute Laden und Darstellen von Waypoints oder Tracks zu einem späteren Zeitpunkt möglich.

Beschaffen des Kartenmaterials

Bei dem Kartenmaterial muss darauf geachtet werden, dass es sich um georeferenzierte Rasterkarten handelt. Solche Karten können unter anderem von der „geoGLIS oHG“ bezogen werden.

offline Laden

Für das Downloaden von georeferenzierten Karten ist es erforderlich, sich auf der Seite [go09b] der „geoGLIS oHG“ zu registrieren. Die Registrierung ist erforderlich, um den gewünschten Bestellvorgang auszulösen.
Die Auswahl eines Kartenausschnittes wird dem Nutzer über einen angepassten Webclient ermöglicht. Nachdem der gewünschte Ausschnitt bestimmt wurde, kann die Bestellung abgeschlossen werden. Danach erhält der Nutzer eine E-Mail, in der ein Download - Link enthalten ist. Dieser Link verweist auf eine Internetseite, auf der das Kacheln des gewählten Kartenausschnittes erfolgt. Nachdem erfolgreichen Erstellen der Kartenkacheln wird dem Nutzer ein Zip - File zum Download bereitgestellt.
Die Kartenkacheln, die im Zip -File enthalten sind, liegen in dem Dateiformat *.png* vor. Zusätzlich zu der .png - Datei enthält der Ordner eine *.pgw*-Datei. Diese Datei trägt den gleichen Namen wie, die .png - Datei (Beispiel: 3280300.5237500.png und 3280300.5237500 .pgw). Durch die Zusatzdatei .pgw erfolgt die Ausgabe als WGS84- oder Gauß-Krüger - codierte Rasterkarte. Weiter Dateiformate für georeferenzierte Karten können sein: .tiff-

mit der dazugehörigen .tfw-Datei.
Die durch die „geoGLIS oHG“ bereitgestellten Kacheln besitzen eine Pixelgröße von 1500 x 1500 Px (Pixel). Diese Kachelgröße kann, je nach gewählten Kartenausschnitt, jedoch variieren.

online Laden
Das online Laden des Kartenmaterials erfolgt über einen mobilen Browser des Endgerätes. Der Browser ruft einen WMS - Service im Internet auf. Dadurch ist das downloaden von Kartenmaterial an Ort und Stelle möglich.
Die Datenübertragung zwischen mobilen Endgeräten und dem Internet kann bei großen Datenmengen noch sehr langsam sein. Ausgenommen ist hier jedoch die Internetverbindung über eine WLAN - Schnittstelle. Diese ist jedoch nicht überall verfügbar. Dadurch kann also abhängig von der Datenmenge der Ladevorgang von Kartenmaterial über einen mobilen WMS - Dienst sehr lange dauern. Aus diesem Grund können, je nach Datentarif des Mobilfunkanbieters, höhere Mobilfunkgebühren anfallen.

2.5 Zusammenfassung

Im Zusammenhang mit dieser Arbeit wurden verschiedene Analysen durchgeführt, die in den vorherigen Kapiteln ausführlich beschrieben wurden. Diese Analyse waren notwendig, um festzulegen welche Voraussetzungen erfüllt sein müssen und welche Anforderungen bestehen, um eine Anwendung für mobile Endgeräte zu entwickeln.
Diese Analysen umfassen die „Bedarfsanalyse“ (Kapitel 2.1), die „Marktanalyse“ (Kapitel 2.2), die „Geräte- und Hardwareanalyse“ (Kapitel 2.3) sowie die „Datenanalyse“ (Kapitel 2.4). Aus diesen verschiedenen Analysen geht hervor, das bei der Entwicklung einer Software eine Reihe von Kriterien beachtet werden müssen.
Mit der Durchführung der Bedarfsanalyse sollte geklärt werden, ob der Bedarf an einer weiteren Software auf dem Markt existiert und welche Anwender mit der entwickelten Software angesprochen werden sollen. Mit der Software werden eine Reihe von Anwender angesprochen. Dazu gehören vor allem Anwender im Outdoor - Bereich, wie zum Beispiel Wandern und Fahrrad fahren. Durch die Software soll die Möglichkeit geschaffen werden, die benötigten Materialien (Karte, Positionsbestimmung) in einem Gerät zu vereinen. Mit der Anwendung soll die Positionsbestimmung des Nutzers durchgeführt werden und auf einer georeferenzierten Karte dargestellt werden. Dadurch ist eine Orientierung auch abseits der Straße möglich.
Durch die Marktanalyse sollte gezeigt werden, welche verschiedenen Arten von Software bereits auf dem Markt existieren. Dabei wurde festgestellt, das bereits eine Reihe von Softwareprodukten auf den Markt angeboten werden. Dabei handelt es sich in der Regel um Sharewareprodukte. Es wurde bei den Recherchen auch zwei Freewareprodukte gefunden, die das Kartendarstellen ermöglichen. Das Kartenmaterial dieser Software wird hauptsächlich von dem Google- oder OpenStreetMap-Server bezogen.
Das Kartenmaterial für die mit der Arbeit entwickelte Software wird von dem WMS-Client der „geoGLIS oHG“ bereitgestellt. Als Grundlage für die Karten dient Datenmaterial der Landesvermessungsämter (ATKIS - Daten). Darüber hinaus enthält das Kartenmaterial der „geoGLIS oHG“ keinerlei Werbung. Aus diesem Grund ist das Kartenmaterial genauer und detaillierter als anderes Kartenmaterial. Da die Darstellung solcher Karten innerhalb einer mobilen Anwendung möglich sein soll, wird in Zusammenhang mit dieser Arbeit eine neue Software entwickelt.
Mit Hilfe der Geräte- und Hardwareanalyse konnte gezeigt werden, welche verschiedenen Typen von mobilen Endgeräten durch die Hersteller auf dem Markt angeboten werden. Dabei haben sich vier Hauptgruppen an mobilen Endgeräten herausgebildet, dazu gehören das Mobiltelefon, das Smartphone, der PDA sowie das BlackBerry. Ein weiterer wichtiger Punkt ist, dass für die verschiedenen mobilen Endgeräte sich unterschiedliche Betriebssysteme gebildet haben (SymbianOS, Windows Mobile, PalmOS, RIM). Wie aus der Geräte- und Hardwareanalyse hervorgeht, wurde sich für die plattformunabhängige Programmiersprache Java entschieden. Es wurde durch Sun eine Java-Version entwickelt, die speziell auf die eingeschränkten Ressourcen mobiler Endgeräte angepasst wurde. Dabei handelt es sich um die Programmiersprache J2ME. Auf Grund der unterschiedlichen Pro-

grammiereinstellungen für die verschiedenen mobilen Endgeräte, wurde sich bei der Entwicklung der Software auf das Betriebssystem SymbianOS festgelegt.
Neben der bereits erwähnten Darstellung von georeferenzierten Kartenmaterial, soll es mit der Software außerdem möglich sein, weitere Daten anzuzeigen. Dabei handelt es sich um GPS - Track's, POI's und Waypoints. Aus diesem Grund wurde eine Datenanalyse durchgeführt. Dadurch wurde festgelegt, welche Daten benötigt werden und woher diese Daten stammen. Da neben der Darstellung von Daten innerhalb der Software auch das Aufnehmen von Datensätzen eine große Rolle spielt, wird zusätzlich eine Export-Funktion integriert.

Es soll eine Java-Anwendung entwickelt werden, die das offline Anzeigen georeferenzierter Rasterkarten ermöglicht. Des Weiteren soll mit Hilfe der Software eine Positionsbestimmung mittels GPS durchgeführt werden.

3 Softwarekonzept

Es wird mit dieser Arbeit ein Softwarekonzept entwickelt, welches auf alle mobilen Endgeräte angewandt werden kann.
Bei der Softwareentwicklung gibt es eine Menge von Faktoren die zu berücksichtigen sind. Die Abbildung 25 zeigt die Systemumgebung. Alle dargestellten Faktoren in dieser Abbildung müssen bei der Entwicklung einer Java-Anwendung für mobile Endgeräte berücksichtigt werden.

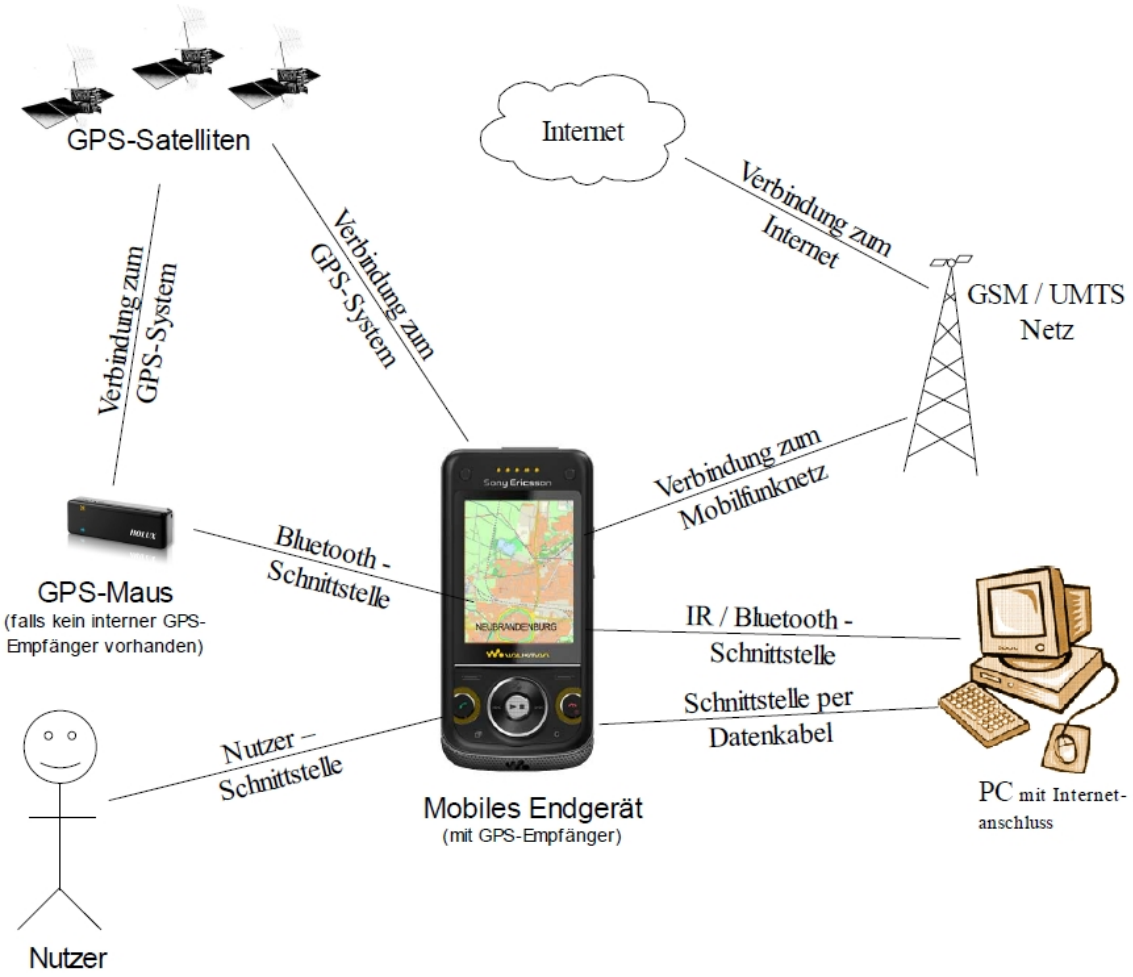


Abbildung 25: Systemumgebung
Quellen der Bilder: [ube09], [Kni 6], [Lib09]

In der Abbildung 25 werden die verschiedenen Schnittstellen aufgezeigt, die zu einen mobilen Endgerät existieren. Eine sehr wichtige Schnittstelle bildet die Nutzerschnittstelle. Durch diese Schnittstelle hat der Nutzer die Möglichkeit mit dem Gerät zu interagieren. Auf Grund dessen kann der Nutzer den Anwendungsablauf auf dem Endgerät beeinflussen. Innerhalb der Anwendung wird ein Menü eingebunden, in dem eine Navigation möglich ist. Mit der Auswahl eines Menüpunktes können verschiedene Prozesse gestartet oder beendet werden.

Solch ein Prozess kann unter anderem den Aufbau einer Schnittstelle zu einem GPS-Empfänger darstellen. Damit eine Positionsbestimmung möglich ist, muss der Empfänger eine Verbindung zu den GPS-Satelliten aufbauen. Es kann auf zwei verschiedene Arten eine Verbindung zu den Satelliten hergestellt werden. Verfügt das mobile Endgerät über ein internes GPS-Gerät, so kann darüber eine direkte Verbindung aufgenommen werden. Bei einem Gerät ohne integriertes Gerät, kann über eine Bluetooth - Schnittstelle ein GPS-Empfänger (GPS-Maus) angeschlossen werden.
Weiterhin existieren verschiedene Schnittstellen, um das mobile Endgerät mit einem herkömmlichen PC zu verbinden. Es kann eine Verbindung über ein Datenkabel, über Bluetooth oder Infrarot mit dem PC hergestellt werden. Diese ist erforderlich, wenn ein Datenaustausch zwischen den beiden Geräten stattfinden soll. Dadurch ist es möglich, die Software, das Kartenmaterial oder die restlichen Geodaten auf das mobile Endgerät zu übertragen.

Aus der Abbildung 25 geht hervor, welche Randbedingungen bei der Programmierung zu berücksichtigen sind. Zur Veranschaulichung der einzelnen Programmabläufe wird die UML - Darstellung (Unified Modeling Language) gewählt. Diese Darstellungsart wird verwendet, da sie sich besonders zur Veranschaulichung von objektorientierte Abläufen eignet. Die UML - Notation stellt verschiedene Diagrammtypen zur Visualisierung bereit.
In Zusammenhang mit dieser Arbeit wird das Aktivitätsdiagramm zur visuellen Darstellung der verschiedenen Abläufe gewählt. Dadurch können die verschiedenen Bereiche, auf die die Anwendung zugreift, veranschaulicht werden.
Diese unterschiedlichen Programmabläufe werden in dem nachfolgenden Kapitel ausführlich erläutert. Die Abbildung 26 stellt ein Übersichtsdiagramm über das Gesamtkonzept sowie des Hauptprogramms der Software-Applikation dar.

3.1 Hauptprogramm

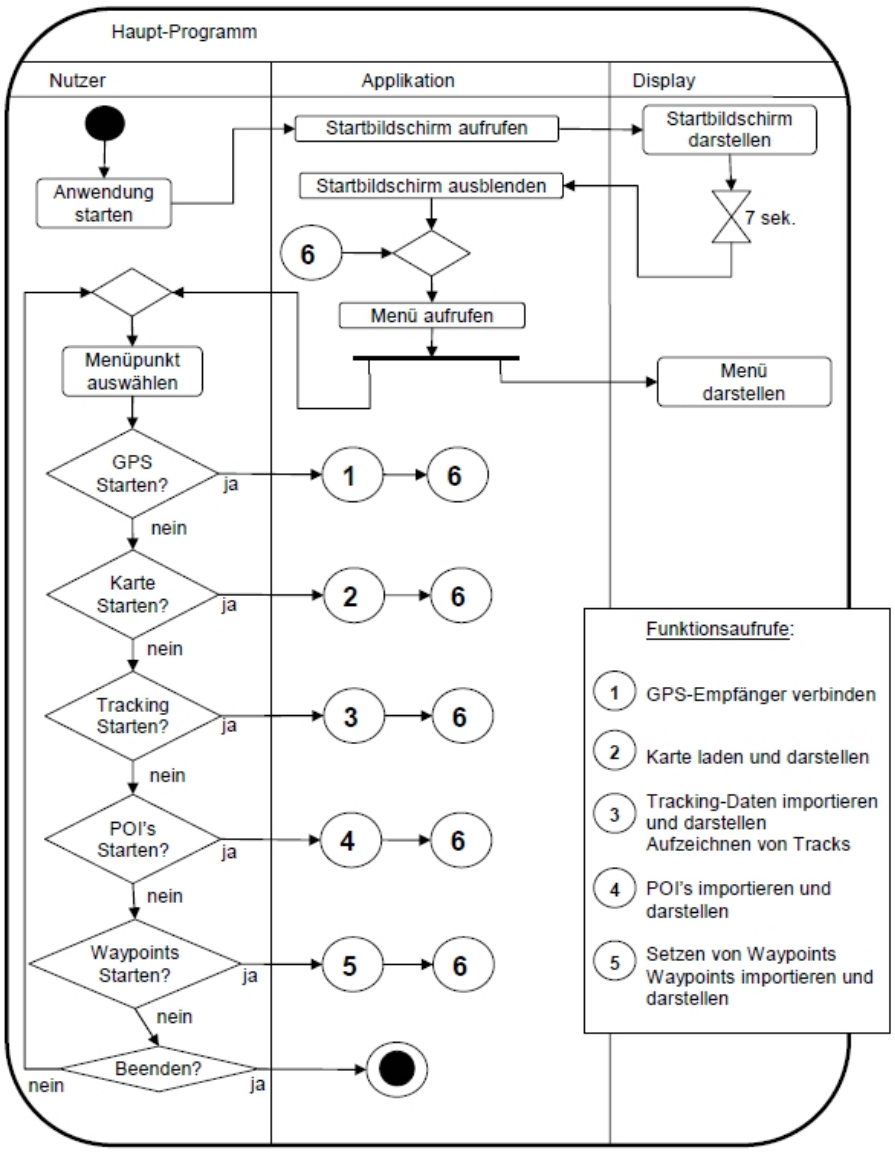


Abbildung 26: Aktivitätsdiagramm des Hauptprogramms

Das abgebildete Aktivitätsdiagramm 26 veranschaulicht das Ablaufkonzept des Hauptprogramms. Es bildet das Grundgerüst der entwickelten Software. Durch das Hauptprogramm werden die restlichen Anwendungen eingebunden und bereitgestellt.
Aus dem Diagramm ist ersichtlicht, dass nach dem Programmstart ein Startbildschirm eingeblendet werden soll. Nach einer fest definierten Zeit, soll das Bild wieder ausgeblendet werden. Das definieren der Zeit ist notwendig, da sonst je nach Gerätetyp die Anzeigedauer auf dem Display unterschiedlich lang sein würde. Dies gilt für die Bildschirmanzeige mittels eines Alert unter J2ME. Darauf wird in dem Kapiteln 4.4 näher eingegangen.

Nach dem Einblenden eines Startbildschirmes auf dem Display wird ein Menü dargestellt. Mit Hilfe des Menüs ist es dem Anwender möglich, durch die einzelnen Funktionen der Software zu navigieren. Dabei sind folgende Menüaufrufe möglich:

1 Verbindung zu einem GPS-Empfänger herstellen

2 Kartenmaterial laden und darstellen

3 Tracking-Daten laden und darstellen

4 POI's laden und darstellen

5 Waypoints setzen
Waypoints laden und darstellen

Durch die Auswahl des entsprechenden Menüpunktes wird es ermöglicht die oben genannten Funktionen aufzurufen und die dazugehörige Anwendung zu starten. Die einzelnen Funktionsaufrufe sind in dem Aktivitätsdiagramm (Abbildung 26) durch die Ziffern 1 - 5 gekennzeichnet. Auf die einzelnen Funktionen wird in den nachfolgenden Kapiteln näher eingegangen. Nachdem der Nutzer durch das Auswählen eines Menüpunktes in das jeweilige Unterprogramm gewechselt hat, soll es ihm möglich sein, jederzeit zum Hauptmenü zurück zukehren. Dies wird durch die Nummer 6 im Aktivitätsdiagramm dargestellt. Die Rückkehr in das Hauptprogramm kann auf zwei verschiedene Arten geschehen. Zum einen durch das Beenden eines aufgerufenen Ablaufes oder zum anderen durch die Aufforderung des Nutzer.
Neben den unterschiedlichen Funktionen steht dem Nutzer innerhalb des Hauptprogramms auch die Möglichkeit zur Verfügung, das Programm ordnungsgemäß zu beenden.

3.2 Verbindung zu einem GPS-Empfänger

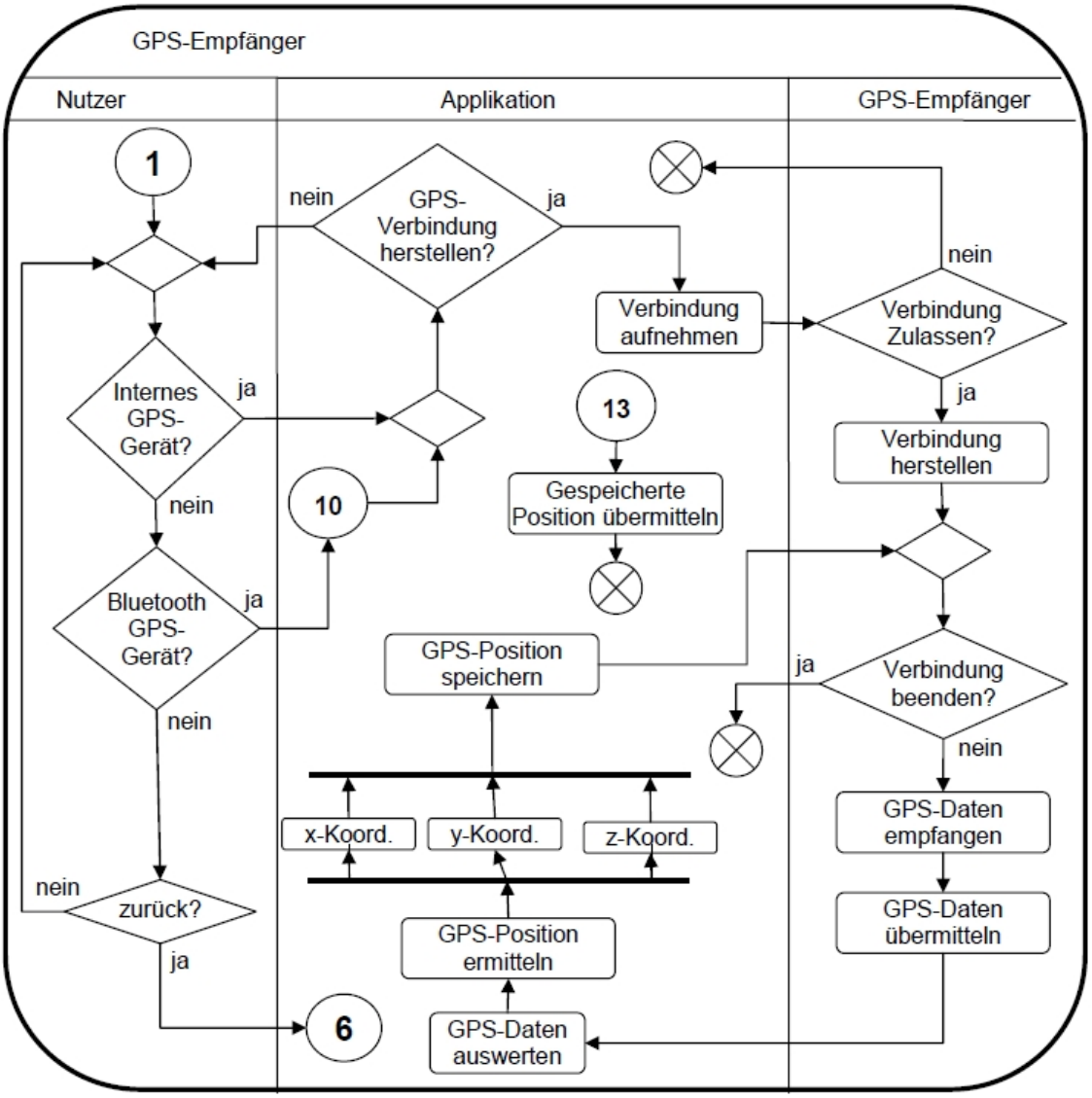


Abbildung 27: Aktivitätsdiagramm - Verbindung zu einen GPS-Empfänger

In dem Programm ist es erforderlich eine Verbindung zu einem GPS-Empfänger herzustellen. Dies ist notwendig, um die aktuelle Position des Gerätes zu bestimmen. Der Verbindungsaufbau wird in dem Aktivitätsdiagramm in der Abbildung 27 dargestellt.
Verfügt das mobile Endgerät über einen integrierten GPS-Empfänger, baut die Anwendung zu diesem Gerät eine Verbindung auf. Sollte das mobile Gerät jedoch keinen internen GPS-Empfänger besitzen, so kann über eine Bluetooth - Schnittstelle (Ziffer: 10) solch ein Gerät angebunden werden. Dieser Prozess wird in dem Diagramm 28 näher erläutert.

Die Abbildung 28 zeigt, das zunächst geprüft wird, ob das interne Bluetooth - Gerät eingeschaltet ist. Sollte dies nicht der Fall sein, wird dem Nutzer eine Mitteilung auf dem Display angezeigt. Ist das BT-Gerät jedoch bereits eingeschaltet, wird eine Verbindung zu diesem Gerät hergestellt und anschließend der Suchvorgang gestartet. Nachdem ein benachbartes GPS-Gerät gefunden wurde, wird es auf dem Display angezeigt. Werden keine Geräte mehr in der näheren Umgebung gefunden, wird der Suchvorgang beendet. Anschließend kann der Nutzer der Software aus den Geräten das entsprechende Gerät auswählen. Danach wird eine Verbindung zu dem Gerät aufgebaut und zur Ausgangsfunktion zurückgekehrt.

Nachdem eine Verbindung zu einem GPS-Empfänger hergestellt wurde, können mit Hilfe des Gerätes GPS-Daten empfangen werden. Wie aus dem Diagramm aus Abbildung 27 ersichtlich, können diese Daten an die entsprechenden Funktionen übermittelt werden. Da es sich bei den GPS-Daten wie bereits erwähnt um NMEA - Datensätze (siehe Kapitel 2.3.4, Abbildung 15) handelt, müssen die benötigten Daten zunächst aus den Datensätzen herausgefiltert werden. Bei den benötigten Daten handelt es sich um die GPS-Koordinaten, die sich aus den geografischen Längen- und Breitengrad zusammensetzen. Es kann für bestimmte Anwendungen, wie die Tracking- und Waypoint - Funktion erforderlich sein, zusätzlich die Höhe aus den GPS-Datensätzen zu filtern. Nach dem Ermitteln der erforderlichen Daten, können diese an die entsprechende Funktion übermittelt werden, von der die Daten benötigt werden. Dazu werden die Daten in einem Zwischenspeicher gehalten. Auf diesen Speicher kann dann durch die anderen Funktionen zugegriffen und die gewünschten Positionsdaten abgerufen werden. Nachdem eine Verbindung zu einem GPS-Empfänger hergestellt wurde, läuft der Empfang der GPS-Daten im Hintergrund weiter. Aus diesem Grund wird direkt in das Hauptprogramm des Menüs zurückgekehrt, ohne das der Ablauf beim Verlassen der Funktionen beendet wird.

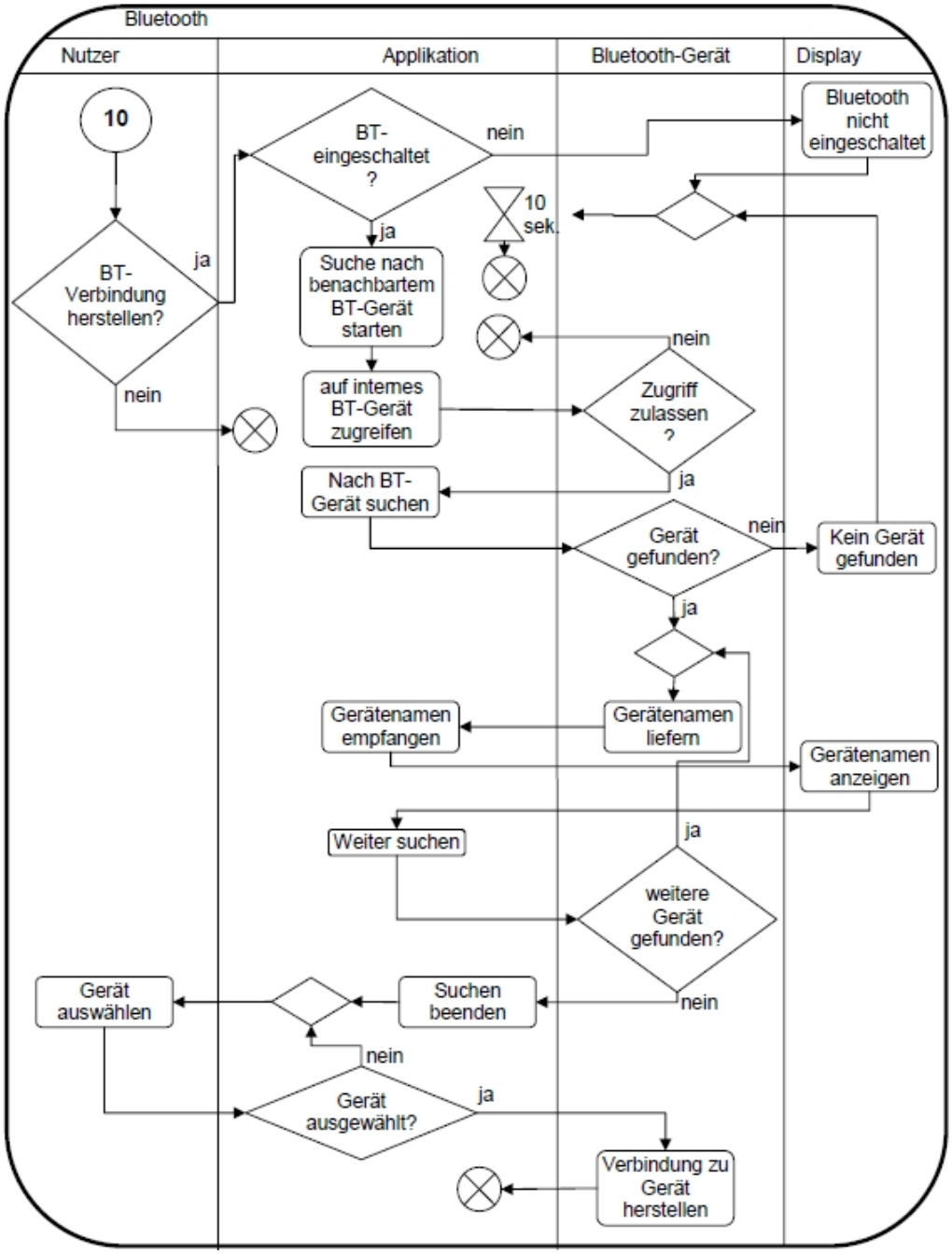


Abbildung 28: Aktivitätsdiagramm zum herstellen einer Blutoothschnittstelle

3.3 Kartendarstellung

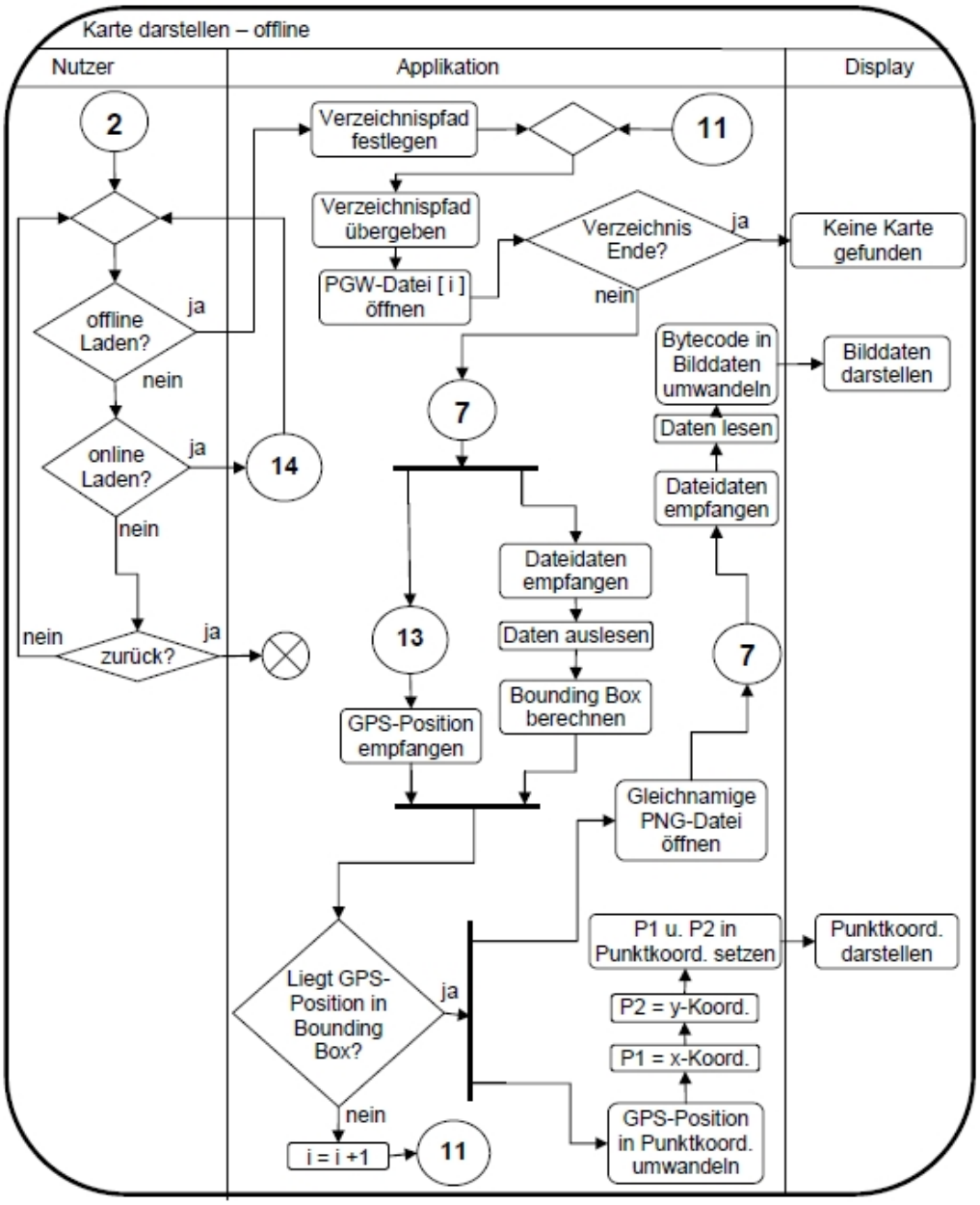


Abbildung 29: Aktivitätsdiagramm - Darstellen von Karten - offline Laden

Wie bereits mehrfach erwähnt, soll dem Nutzer das Darstellen von Kartenmaterial ermöglicht werden. Die Abbildungen 29 und 30 stellt den Ablauf für den Import von Kartenmaterial in die Anwendung dar. Dabei wird zwischen den beiden Optionen „offline Laden“ und „online Laden“ unterschieden.
Wählt der Nutzer die Variante des „offline Ladens“, greift die Anwendung auf das Dateisystem des mobilen Endgerätes zu, um das Kartenmaterial in die Anwendung zu importieren. Dazu wird zunächst ein vordefinierter Verzeichnispfad an das Dateisystem übergeben. Der Zugriff auf das Dateisystem wird in einer extra Funktion ausgeführt. Dies wird durch die Ziffer 7 kenntlich gemacht (siehe Kapitel 4.6.1).

Wie bereits erwähnt, besteht das gelieferte Kartematerial aus einer PGW - sowie einer PNG - Datei. Bevor der entsprechende Kartenausschnitt geladen werden kann, wird die dazugehörige PGW - Datei geladen. Aus dieser Datei werden die benötigten Daten zur Berechnung der Bounding Box gelesen. Diese definiert die Ausdehnung der dazugehörigen Kachel. Eine Bounding Box enthält die Eckkoordinaten (oben Links; unten Rechts) der Karte. Dadurch kann geprüft werden, ob die mit einem GPS-Empfänger (Ziffer 13) übermittelte Position innerhalb des Kartenbereiches liegt. Sollte die Position innerhalb des Bereiches liegen, wird die gleichnamige PNG - Datei geladen und auf dem Display des mobilen Endgerätes dargestellt und die ermittelte Position auf ihr abgebildet. Liegt die GPS-Position jedoch außerhalb des Kartenbereiches, so wird die nächste indem Verzeichnis befindliche PGW - Datei geöffnet. Dieser Vorgang wird solange wiederholt, bis die entsprechende Kartenkachel gefunden wurde. Sollte in dem Verzeichnis kein entsprechender Kartenausschnitt vorhanden sein, so wird dem Nutzer eine Mitteilung auf dem Display ausgegeben.

Bei der Variante des „online Ladens“ (Abbildung 30), wird das Kartenmaterial aus dem Internet bezogen. Dazu muss das entsprechende Kartenmaterial auf einem Server liegen, damit mit dem mobilen Endgerät darauf zugegriffen werden kann. Bevor über die Anwendung eine Verbindung zum Internet hergestellt werden kann, muss dieses zunächst gestartet werden. Anschließend kann der Name der URL übergeben und angewählt werden. Danach kann das Kartenmaterial von dem Server bezogen werden. Nachdem geprüft wurde, ob die aktuelle GPS-Position auf einer der Kartenkacheln liegt, erfolgt der Download des jeweiligen Kartenausschnitts. Danach können die Daten von der Anwendung empfangen und gelesen werden. Abschließend wird die Karten auf dem Display dargestellt. Das bestimmen der Position auf der Karte erfolgt genau wie beim „offline Laden“ des Kartenmaterials.

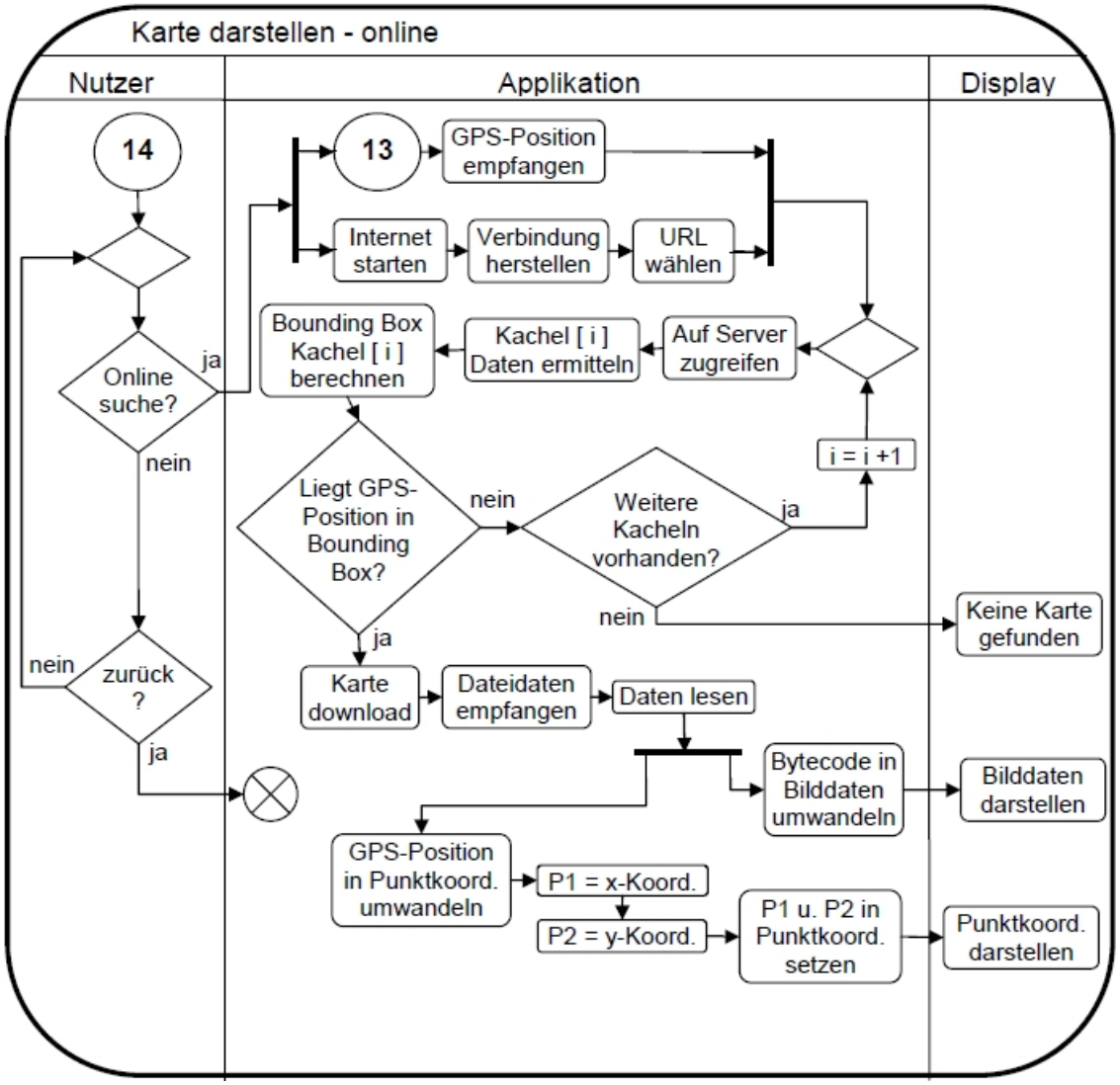


Abbildung 30: Aktivitätsdiagramm - Darstellen von Karten - online Laden

3.4 Tracking-Daten

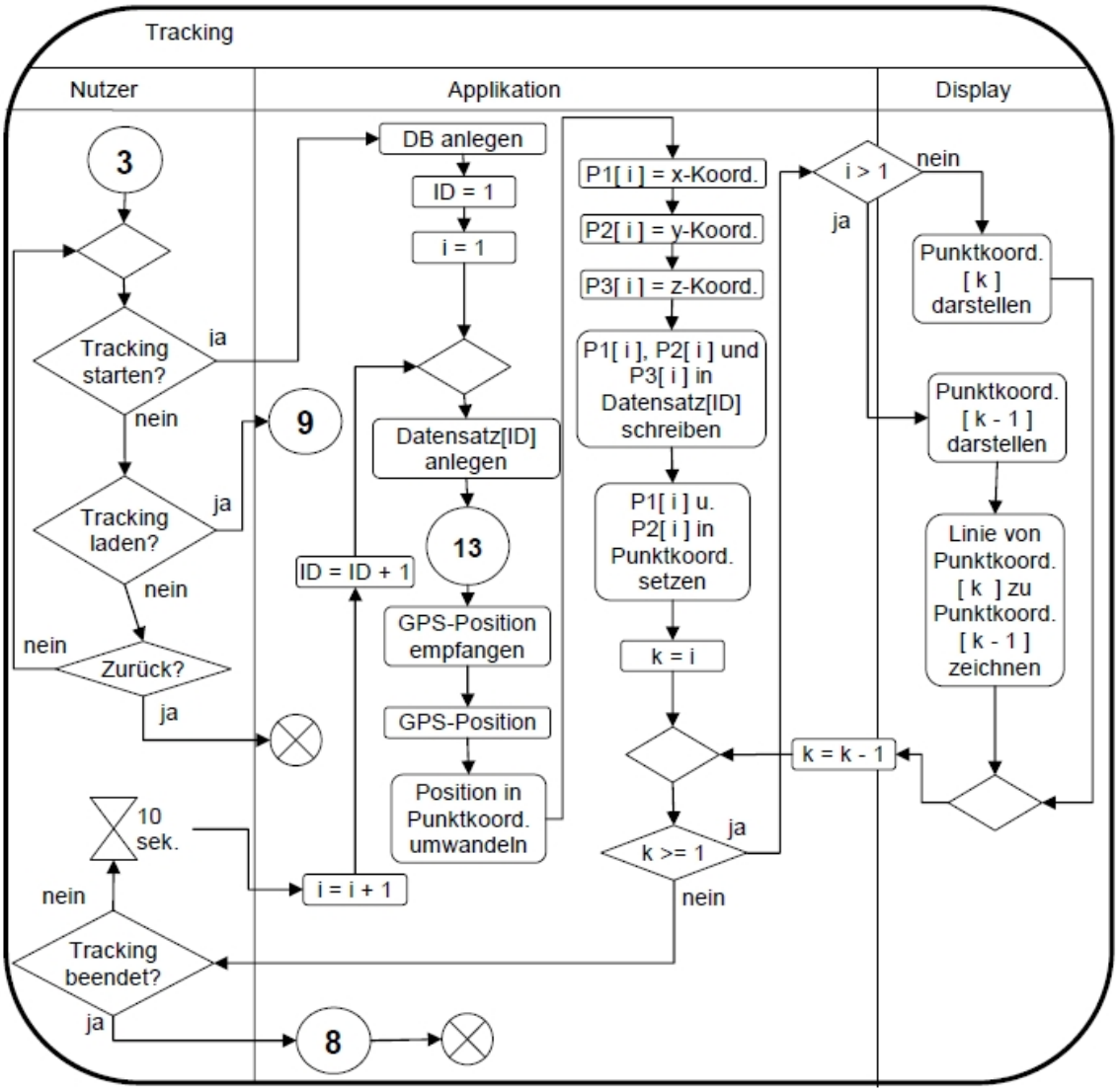


Abbildung 31: Aktivitätsdiagramm - Aufzeichnen von Tracks

Durch die Auswahl des Menüpunktes „Tracking starten“ im Hauptprogramm wird dem Nutzer ein neues Menü eingeblendet. Innerhalb des Untermenüs stehen dem Nutzer zwei Optionen zur Auswahl. Zum einen hat er die Möglichkeit, neue Tracking-Daten aufzuzeichnen. Zum anderen besteht für den Nutzer die Option, bereits vorhandene Tracking-Daten in die Anwendung zu importieren. Dieser Vorgang wird in dem Aktivitätsdiagramm 32 näher erläutert.

Das Diagramm in der Abbildung 31 beschreibt, den Ablauf für das Aufzeichnen von GPS-Track's. Aus diesem Diagramm ist ersichtlich, das die aufgenommenen Daten in einer Datenbank abgelegt werden. Nach dem Beenden des Vorgangs werden die Datenbank-Datensätze in einer Datei abgelegt. Bevor jedoch die Daten abschließend in einer Datei gespeichert werden können, sind eine Vielzahl von Arbeitsschritte notwendig.

Diese Schritte dienen dazu, um die erforderlichen Daten zu erfassen. Zu diesen Daten gehören die Längen- und Breitengrade der aktuellen Position. Um die Position zu bestimmen ist es als erstes erforderlich, eine Verbindung zu einem GPS-Empfänger herzustellen. Aus diesem Grund greift die Funktion des „GPS-Tracks aufzeichnen“ auf eine extra Anwendung zu. Dieser Vorgang ist durch die Ziffer 13 gekennzeichnet. Die Verbindung zu einen GPS-Empfänger wurde bereits in Kapitel 3.2 ausführlich beschrieben. Durch diese Funktion werden die Daten empfangen. Anschließend wird die GPS-Position an die aufrufende Anwendung übergeben. Nachdem die Daten übergeben wurden, ist eine Weiterverarbeitung der Daten möglich.
Dabei werden jeweils die geografische Breite (x) und die geografische Länge (y) in einen Datenbank-Datensatz geschrieben. Darüber hinaus wird die GPS-Position in eine Punktkoordinate umgewandelt, um diese dann auf dem Display des mobilen Gerätes darzustellen. Der Vorgang des Empfangens der GPS-Position, das Schreiben in einen Datensatz und das anschließende Darstellen auf dem Display wird in einem bestimmten Zeitintervall wiederholt. Dies geschieht solange, bis der Ablauf durch den Anwender beendet wird. Sobald mehrere GPS-Positionen empfangen und in die Datenbank abgelegt wurden, wird eine Linie zwischen den einzelnen Punkten gezeichnet. Dadurch wird die visuelle Darstellung der bereits zurückgelegten Strecke realisiert.
Nachdem das Tracking durch den Anwender beendet wurde, werden die in der Datenbank erfassten Datensätze in einer Datei abgelegt. Für das Speichern der Daten existiert eine extra Funktion, die das Speichern von Daten erlaubt. Der Funktionsaufruf ist in dem Aktivitätsdiagramm durch die Ziffer 8 dargestellt. Wurde die Datei erfolgreich gespeichert, kehrt die Anwendung in das Hauptprogramm zurück.

Im nachfolgenden Abschnitt soll auf das Importieren von GPS-Track's näher eingegangen werden. Der Ablauf für den Vorgang des Importierens wird in der Abbildung 32 dargestellt.
Für das Laden bereits vorhandener Tracking Daten muss die Anwendung auf das Dateisystems des mobilen Endgerätes zugreifen. Das Öffnen einer ausgewählten Datei findet in einer extra Funktion statt. Dies wird durch die Ziffer 7 in dem Aktivitätsdiagramm 32 gekennzeichnet. In dem Kapitel 4.6.1 wird das Öffnen von Dateien näher erläutert. Mit Hilfe der Funktion für das Öffnen von Dateien können die Dateidaten an die Import-Funktion übergeben werden. Nachdem die Dateidaten übermittelt wurden, können sie durch die Import-Funktion ausgewertet werden. Das Aktivitätsdiagramm zeigt, dass das Auslesen der Daten mit Hilfe eines XML - Parser's geschieht. Solch ein Parser wird verwendet, da die zu ermittelnden Daten in einer XML - Struktur vorliegen (siehe Kapitel 2.4.1).

Das Auslesen der Daten erfolgt elementeweise. Bei den jeweiligen Elementen handelt es sich um die Angabe der Längen- und Breitengrade. Diese Werte werden auf einer dafür definierten Variablen abgelegt. Nachdem innerhalb der Datei keine weiteren XML - Elemente mehr vorhanden sind, wird der Lesevorgang der Daten beendet. Anschließend werden die Punkte auf dem Display visualisiert. Dies geschieht, in dem die einzelnen Punkte auf dem Display dargestellt werden. Zur Veranschaulichung der Strecke, wird eine Linie zwischen den einzelnen Punkten gezeichnet.

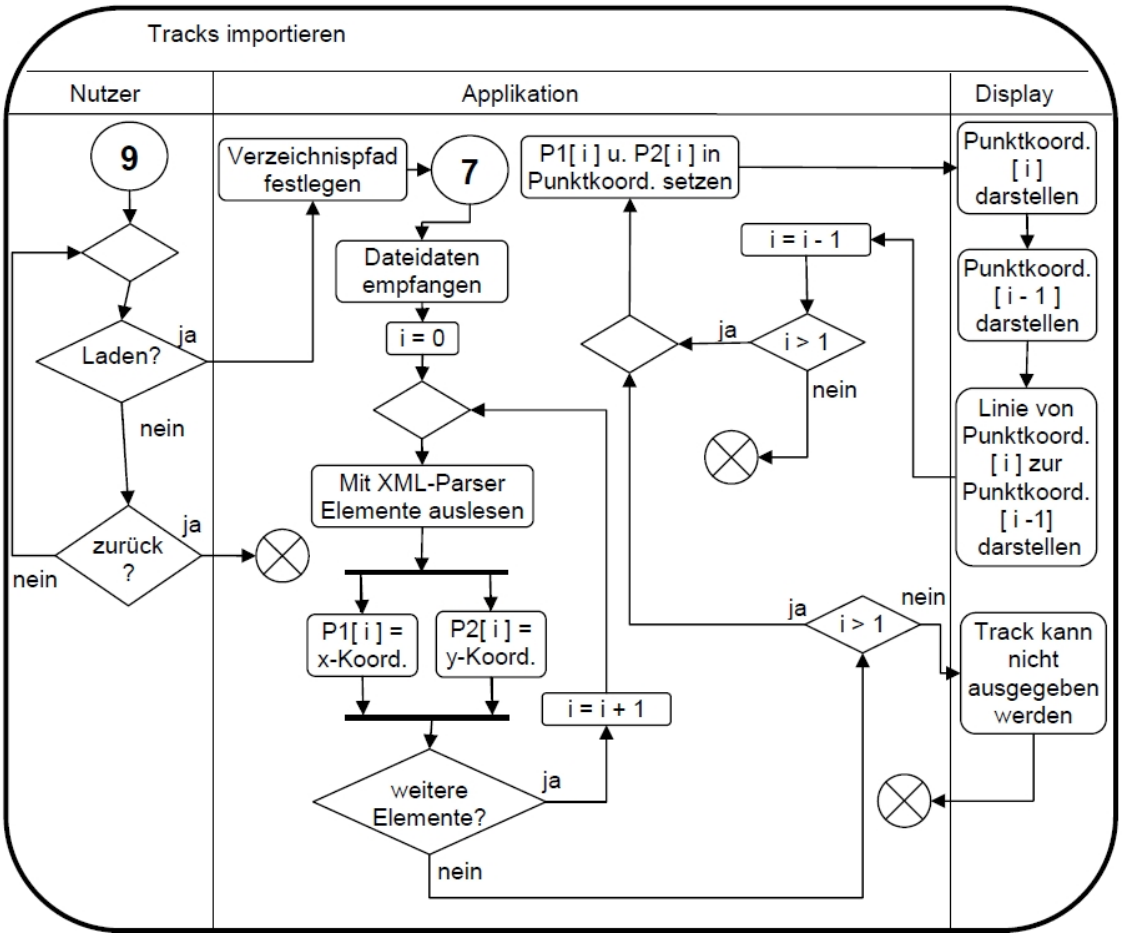


Abbildung 32: Aktivitätsdiagramm - Importieren von Tracks

3.5 POI's und Waypoints

Import von POI's und Waypoints

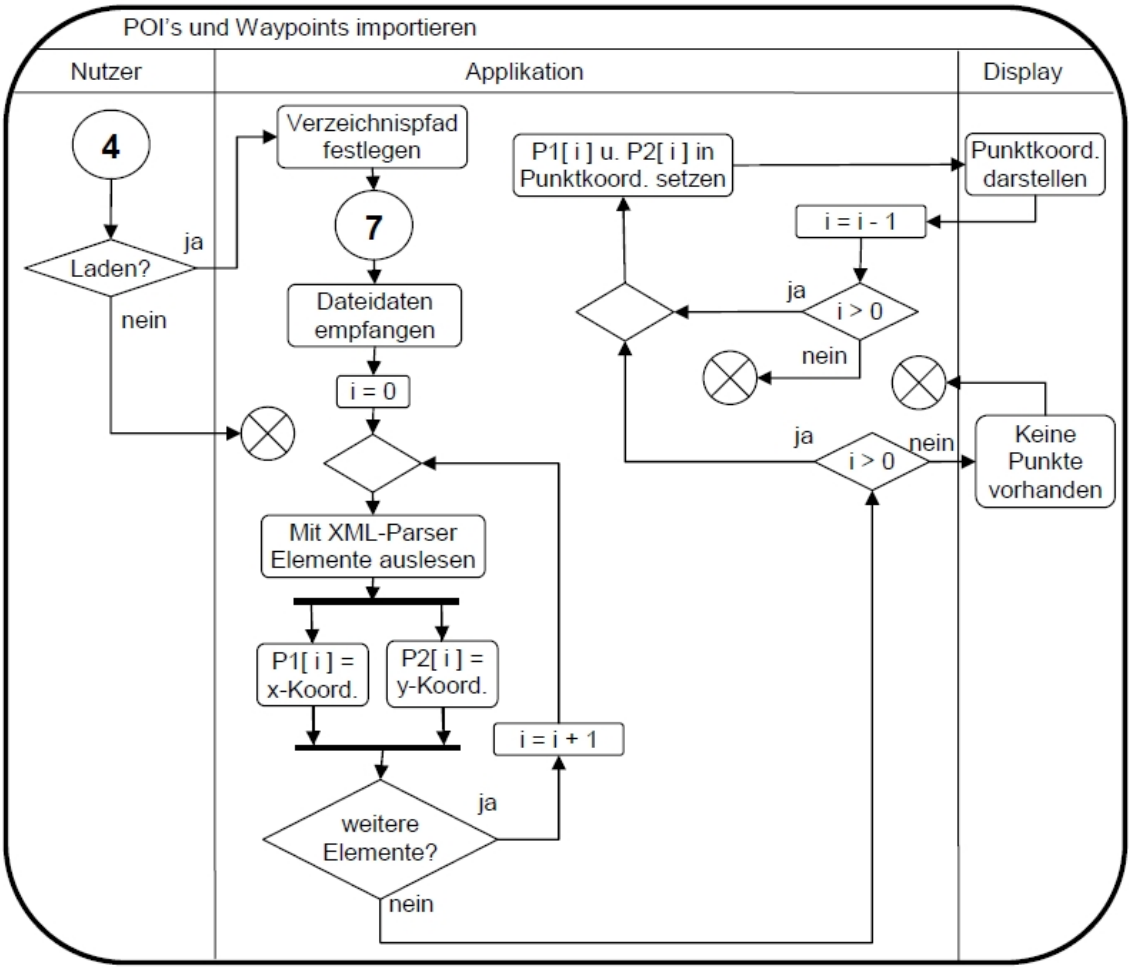


Abbildung 33: Aktivitätsdiagramm - Importieren von POI's bzw. Waypoints

Das Aktivitätsdiagramm in Abbildung 33 stellt den Ablauf für den Import von POI's und Waypoints dar. Die entsprechenden Daten werden von dem Dateisystem des mobilen Endgerätes bezogen. Zum Öffnen der Datei wird auf eine extra Funktion zugegriffen. Dies wird durch die Ziffer 7 kenntlich gemacht.
Nachdem die Datei geöffnet und die Dateidaten übermittelt wurden, können die die XML - Daten genau wie bei den Tracking-Daten elementeweise ausgelesen werden. Anschließend erfolgt die Darstellung der Daten auf dem Display.
Die Funktionsweise des Hereinladens der Daten funktioniert genauso wie bei der Import-Funktion für das Tracking, sie unterscheiden sich lediglich in der Darstellung der Daten auf dem Display. Aus diesem Grund sollte überlegt werden, die Import-Funktion in einen gesonderten Ablauf zu realisieren. Auf Grund der Übersichtlichkeit, wird sich in Zusammenhang mit dieser Arbeit dazu entschlossen, den Ablauf in zwei getrennten Diagrammen darzustellen.

Setzen von Waypoints

Neben den Import von Waypoints, soll der Nutzer auch die Möglichkeit haben, eigene Punkte zu setzen. Der Ablauf für das Setzen von Waypoints wird in dem Aktivitätsdiagramm in der Abbildung 34 näher erläutert.
Wie aus dem Diagramm ersichtlich, wird dem Nutzer ein Untermenü eingeblendet. Mit Hilfe des Menüs kann der Nutzer zwischen zwei Optionen wählen. Dem Nutzer wird die Möglichkeit gegeben „Waypoints zu setzen“ oder aber „Waypoints zu importieren“. Wählt der Nutzer die Option für das Importieren von Waypoints, so wird die gleiche Anwendung gestartet, wie sie im vorherigen Abschnitt beschrieben wurde (siehe Abbildung 33). Dieser Vorgang wird in dem Diagramm 34 durch die Ziffer 4 gekennzeichnet.
Mit der Option „Waypoint setzen“ erhält der Anwender der Software die Möglichkeit eigene Waypoints aufzunehmen und anschließend in einer Datei abzulegen. Wie bereits erwähnt, wird dieser Ablauf in der Abbildung 34 detaillierter dargestellt. Das Aktivitätsdiagramm (Abbildung 34) zeigt, dass die GPS-Position der Anwendung durch die Funktion „GPS-Empfänger“ (Ziffer 13) bereitgestellt wird. Nach dem Empfangen der Position, werden diese Daten jeweils in eine Datenbank geschrieben. Dabei wird jeder Koordinatenwert (x, y, z) in einer extra Spalte abgelegt. bevor die Daten jedoch in eine Datenbank gespeichert werden können, muss diese vorher erzeugt werden. Neben den Längen- und Breitengraden sowie der Höhe, kann durch den Nutzer auch eine Beschreibung für den Punkt erfasst werden.
Abschließend werden die erfassten Daten aus der Datenbank gelesen und in einer Datei gespeichert (Funktion mit der Ziffer 8). Nachdem dieser Vorgang abgeschlossen wurde, wird die Funktion beendet und in das Hauptmenü zurückgekehrt.

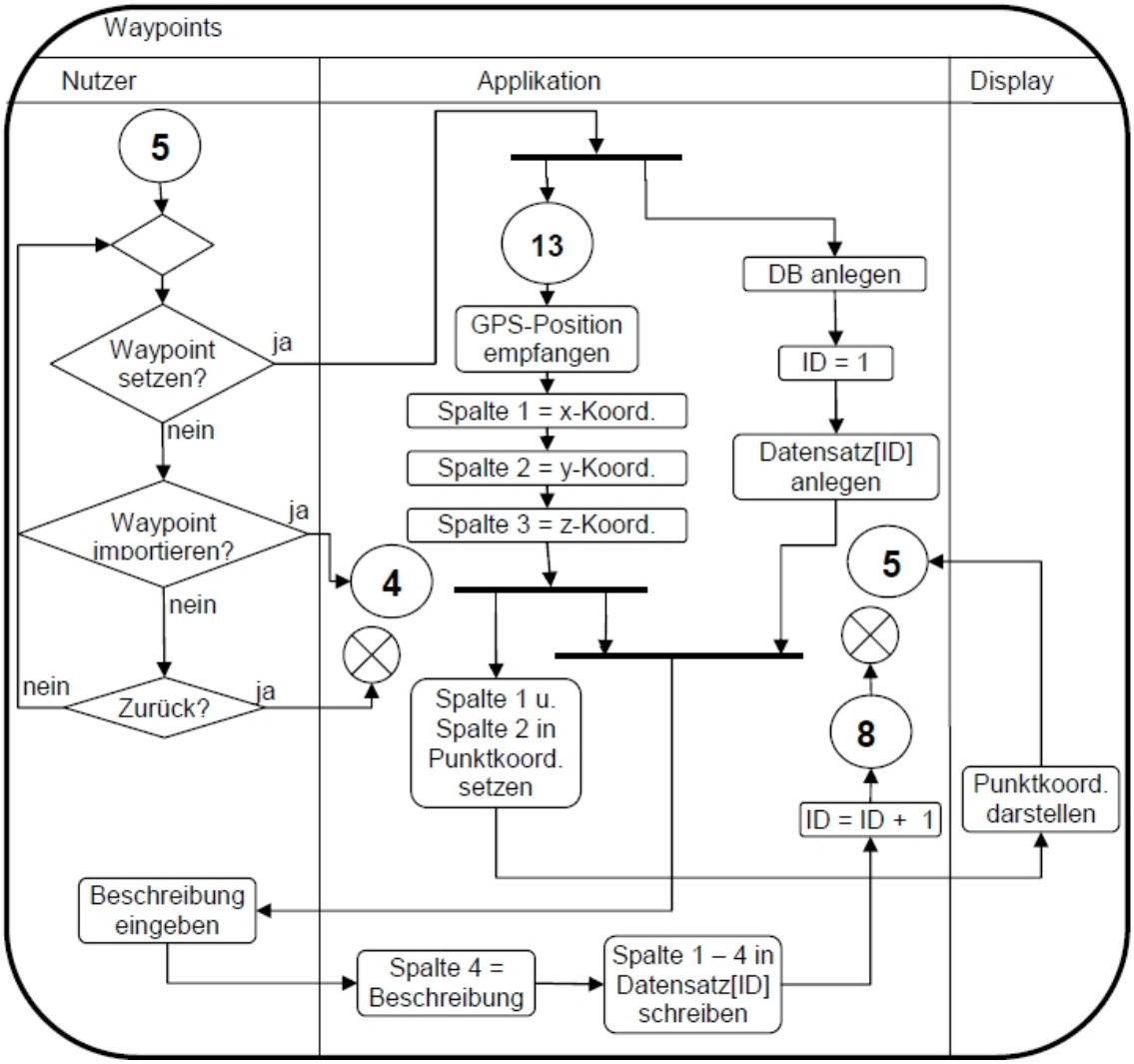


Abbildung 34: Aktivitätsdiagramm - Setzen von Waypoints

3.6 Öffnen von Dateien

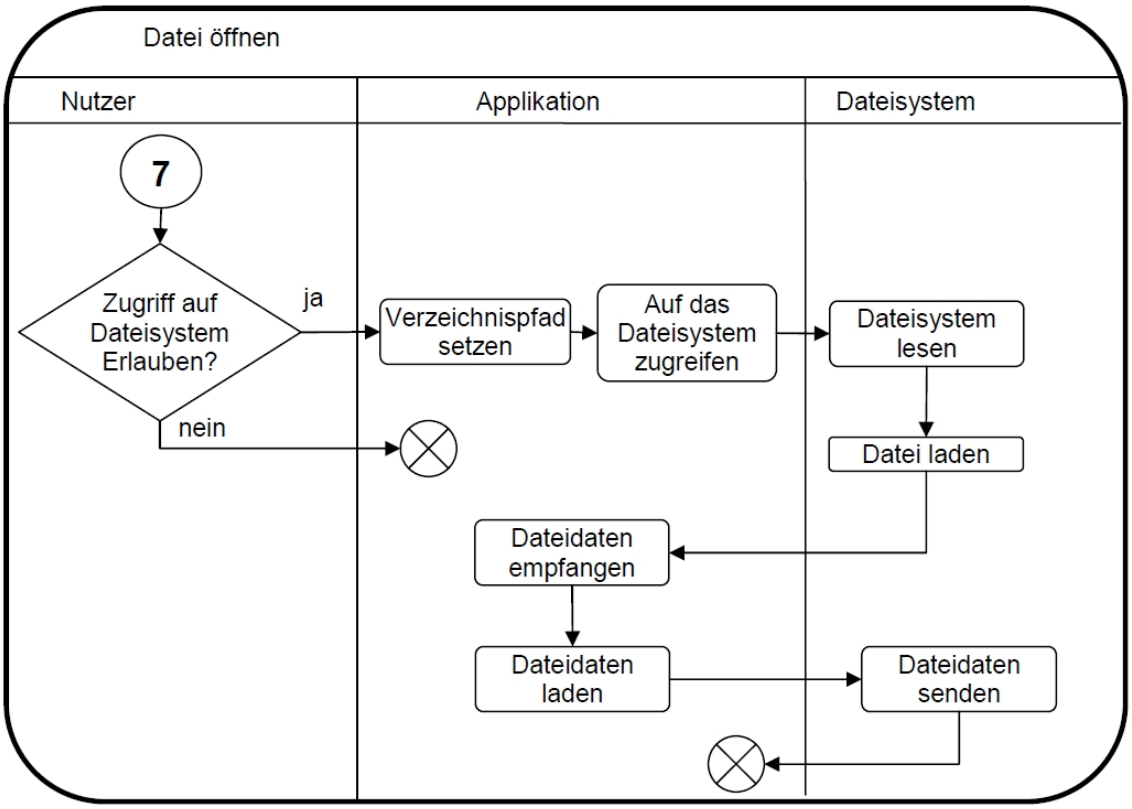


Abbildung 35: Aktivitätsdiagramm - Allgemeines Öffnen einer Datei

Das in der Abbildung 35 dargestellte Aktivitätsdiagramm beschreibt den Ablauf für das Öffnen von Dateien. Diese Funktion wird von mehreren anderen Unterprogrammen innerhalb der Software benötigt. Aus diesem Grund wird diese in ein separates Diagramm geschrieben, um so die mehrfache Darstellung zu vermeiden.
Wie aus dem Aktivitätsdiagramm ersichtlich, wird eine Verbindung zu dem Dateisystem des mobilen Endgerätes hergestellt. Dies ist notwendig, um eine Datei, die sich auf dem Dateisystem befindet, zu öffnen. Damit die Anwendung auf das Dateisystem zugreifen kann, muss dieser durch den Nutzer zugelassen werden. Sollte dies nicht der Fall sein wird die Anwendung beendet und zu der Funktion zurückgekehrt, welche die Anwendung aufgerufen hat.
Nachdem der Zugriff durch den Nutzer bestätigt wurde, können die auf dem System vorhandenen Dateien und Verzeichnisse durch die Applikation empfangen werden. Durch die aufrufende Funktion wird der Pfad der Verzeichnisse übergeben. Auf diesen wird anschließend zugegriffen. Danach wird die geforderte Datei geladen und die Dateidaten an die entsprechende Funktion übermittelt. Nachdem die Daten an die Anwendung übergeben wurden, wird die Funktion des Datei Öffnens beendet.

3.7 Speichern einer Datei

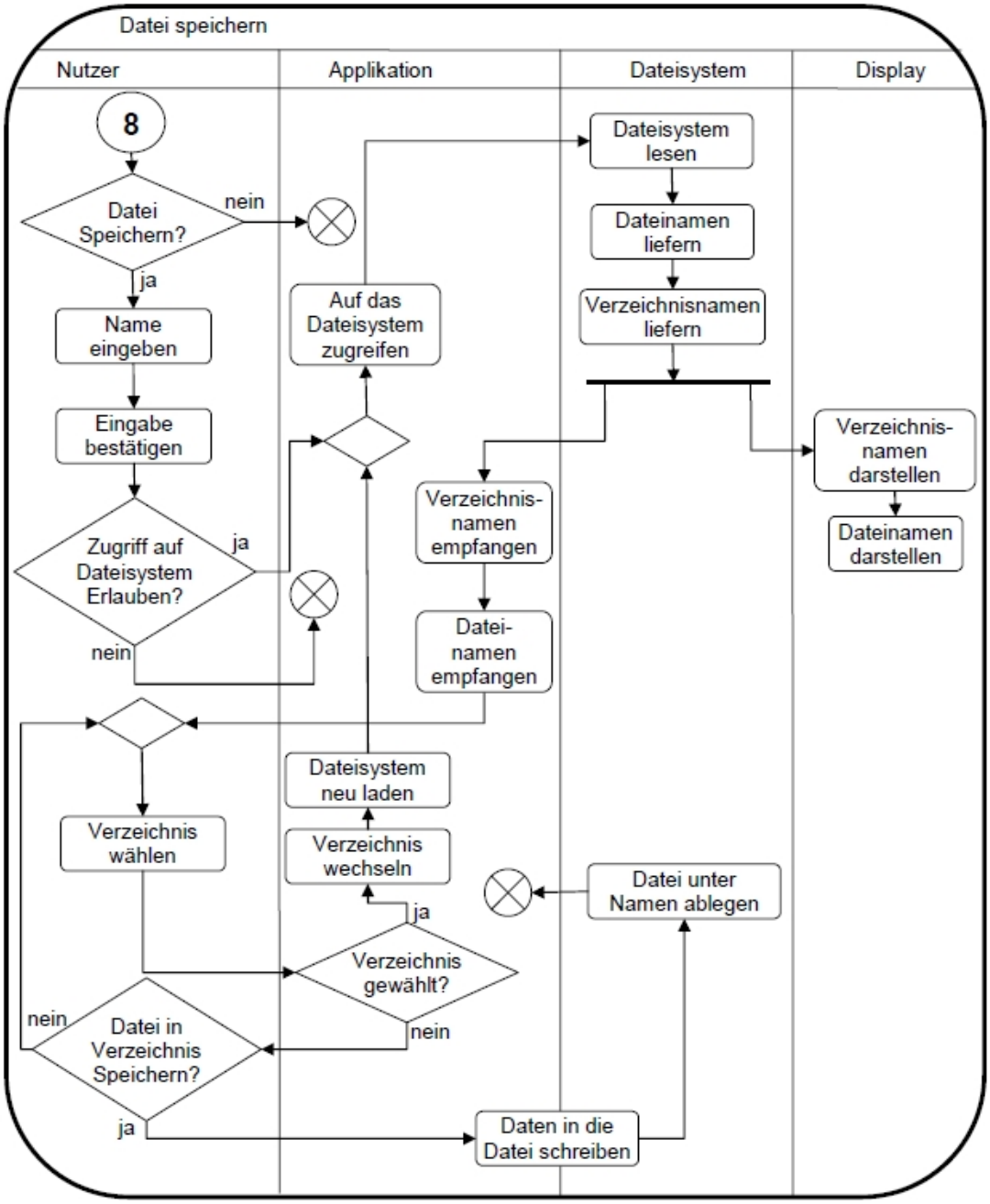


Abbildung 36: Aktivitätsdiagramm - Allgemeines Speichern einer Datei

In einigen Funktionen innerhalb der Software ist das Speichern von Dateien vorgesehen. Dadurch wird es ermöglicht, die in einer Datei gespeicherten Daten zu einem späteren Zeitpunkt erneut in das Programm zu laden oder die Daten mit einer Analysesoftware auszuwerten.

Der Ablauf für das Speichern der Daten wird in dem Aktivitätsdiagramm (Abbildung 36) dargestellt. Bevor die Daten gespeichert werden können, muss der Nutzer der Software ein Name eingegeben, unter der die jeweilige Datei abgelegt werden soll. Nach der Eingabe des Namens wird durch die Applikation auf das Dateisystems des mobilen Endgerätes zugegriffen. Dieser Zugriff muss durch den Nutzer der Software zugelassen werden.

Anschließend kann die Applikation die Daten auf dem Dateisystem lesen. Dadurch ist es möglich, die Verzeichnisnamen und die Dateinamen auf dem Display anzuzeigen. Der Nutzer hat nun die Möglichkeit, zwischen den Verzeichnissen zu wechseln. Dadurch ist der Speicherort frei wählbar. Nachdem der Speicherort gewählt wurde, kann die Datei durch Bestätigung des Nutzers auf dem Dateisystem abgelegt werden. Ist der Speichervorgang beendet, wird die Anwendung geschlossen und zu der Anwendung zurückgekehrt, die das Speichern einer Datei ausgelöst hat.

3.8 Systemvoraussetzungen

Für eine Softwareanwendung auf mobilen Endgeräten müssen verschiedene Voraussetzungen erfüllt sein, damit die Anwendung auf ihnen lauffähig ist. Wie bereits in dem Kapitel 2.3.1 erwähnt, unterliegen die mobilen Endgeräte eine Reihe von technischen Einschränkungen. Aufgrund der heterogenen Hardware der mobilen Endgeräte gibt es kein einheitliches Design.
Im Zusammenhang mit dieser Arbeit wird eine J2ME-Applikation entwickelt, die auf Mobiltelefone unter SymbianOS läuft. Aus diesem Grund wird nachfolgenden auf die Systemanforderungen, die ein Mobiltelefon mindestens erfüllen muss, eingegangen. Eine Übersicht über die Anforderungen sind in der Tabelle 7 dargestellt:

Displayauflösung:	zw. 84 x 48 Px und 240 x 320 Px
Display:	Farbdisplay
Farbtiefe (Bit):	1, 4, 12, 16
Bandbreite in Bit/s:	14k - 72k
Speicher RAM (MB):	16 bzw. 32
Akkulaufzeit:	bis zu 400 h
Laufzeitumgebung:	Java Me
Betriebssystem:	SymbianOS
Bluetooth (BT):	ja
GPS	extern (BT) oder intern
USB-Unterstützung	ja
Internet-Zugriff	ja
SD-Karte	ja

Tabelle 7: Systemvoraussetzungen für Mobiltelefone
Quellen: [Müc09], [Tot09]

Ein wichtiger Faktor bei mobilen Endgeräten ist die Speicherkapazität. Der interne Gerätespeicher liegt bei Mobiltelefonen um die 40 MB. Bei neuartigen Geräten lässt sich dieser Speicher durch SD-Karten erweitern, in der Regel auf maximal 4 GB. Der Arbeitsspeicher, der Mobiltelefonen zur Verfügung steht, wird zusätzlich durch verschiedene Faktoren beeinflusst. Zu diesen Faktoren gehören das Ausführen von Anwendung sowie das Laden von Grafiken, da durch den geringen Arbeitsspeicher ein nur mäßiges Caching unterstützt wird. Daher sollte beim Entwickeln darauf geachtet werden, dass so wenige Ressourcen wie nötig beansprucht werden.
Ein weiteres Problem sind die Akkulaufzeiten der Mobiltelefone. Diese betragen im Standby ca. 400 Stunden. Durch das Ausführen von Anwendung und das Benutzen eines GPS-Empfängers wird diese stark verringert. Aus diesem Grund sollte darauf geachtet werden, das der Akku aufgeladen ist.
Da eine auf Java basierende Software entwickelt wird, ist es eine Grundvoraussetzung, dass das Mobiltelefon über eine J2ME-Laufzeitumgebung verfügt. Außerdem sollte das Gerät mindestens eine Bluetooth Schnittstelle bereitstellen, damit ein externer GPS-Empfänger angebunden werden kann. Auf diese Schnittstelle kann jedoch verzichtet werden, wenn das Mobiltelefon einen integrierten GPS-Empfänger besitzt.

In der Abbildungen 37 wird Dargestellt, welche Komponenten auf alle Fälle zur Verfügung stehen sollten:

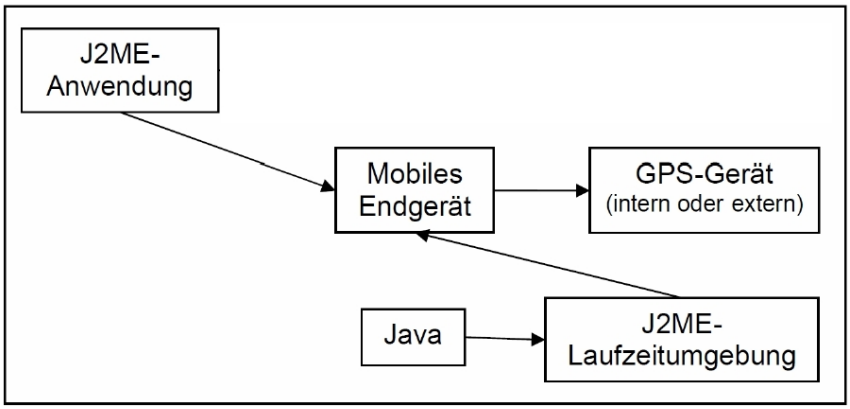


Abbildung 37: Grundkonzept

4 Umsetzung

4.1 J2ME (Java 2 Mirco Edition)

In dem Kapitel 2.3.3 wurde bereits erwähnt, dass die Java 2 Mirco Edition eine Programmiersprache ist, die speziell auf die Anwendungsentwicklung mobiler Endgeräte ausgelegt ist. Die Entwicklung solch einer Sprache war nötig, da bei mobilen Geräten Einschränkungen hinsichtlich Displaygröße, Speicherausstattung sowie der Prozessorleistung bestehen. Um den Einschränkungen gerecht zu werden, wurde auf bestimmte Java - Funktionalitäten verzichtet. Neben diesen Einschränkungen ist zu beachten, dass die heutigen mobilen Endgeräte über spezielle Hardware-Komponenten verfügen, wie etwa ein Kartenlesegerät. Zusätzlich ist zu beachten, dass es eine Vielzahl unterschiedlicher Endgeräte mit verschiedener Systemarchitektur auf dem Markt gibt. [Esc03], [Mos06]
Um diesen Anforderungen und Einschränkungen mobiler Endgeräte gerecht zu werden, benötigt die J2ME eine besondere Struktur, wie in Abbildung 38 dargestellt.

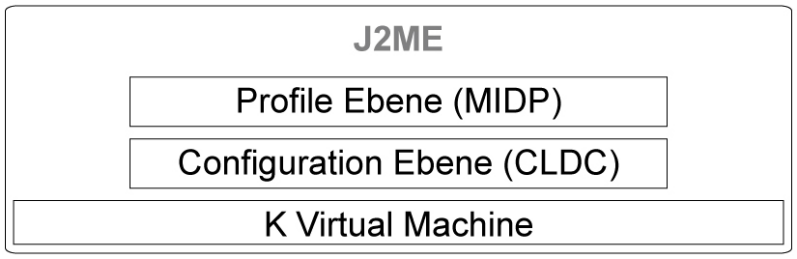


Abbildung 38: Struktur von J2ME (für CLDC Configuration).
Quelle: [Esc03]

Die Abbildung 38 zeigt, das die Entwicklung unter J2ME auf zwei Ebenen basiert. Dabei handelt es sich um die „Configuration-Ebene“ und die „Profil-Ebene“. Außerdem ist aus der Abbildung ersichtlich, dass die Profilebene über der Konfigurations-Ebene liegt. Auf beiden Ebenen werden dem Entwickler verschiedene API's zur Verfügung gestellt. Bei der Profil-Ebene handelt es sich dabei um eine „Low-Level-API“ und bei der Konfiguration um eine „High-Level-API“.
Die Konfigurations-Ebene dient dazu den verschiedenen Hardwareausstattungen der mobilen Endgeräte gerecht zu werden. Auf dieser Ebene wird momentan in zwei Konfigurationen unterschieden. Es handelt sich zum einen um CLDC (Connected Limited Device Configuration) und zum anderen um CDC (Connection Device Configuration). Für die CLDC - Konfiguration existierten zurzeit zwei Versionen 1.0 und 1.1. Zu dem Einsatzbereich für die CLDC - Konfiguration gehören die Mobiltelefone. Für diese Konfiguration wird unter J2ME eine VM bereitgestellt, die auf die starke Limitierung der mobilen Endgeräte ausgelegt ist, die KVM (Kilobyte Virtual Machine). Es werden außerdem viele Systemfunktionen durch die CLDC - Bibliotheken zur Verfügung gestellt. Darüber hinaus ist das dynamische Laden von Programmen sowie das Laden von interaktiven Inhalten und Anwendungen unter dieser Konfiguration möglich. [Esc03], [Mos06], [Use08], [Stu08]

„**CLDC - Anforderungen:**

- 16- oder 32-Bit-Prozessor
- Für die KVM und die CLDC-Bibliotheken sollen mindestens 160 Kb nicht-flüchtiger Speicher vorhanden sein, ein Speicher also, dessen Zustand sich durch Ausschalten des Gerätes nicht ändert
- Wenigstens 32 Kb flüchtiger Speicher (RAM) müssen als Arbeitsspeicher für die JVM vorhanden sein.“[Mos06]

Die andere Konfiguration, CDC, findet hauptsächlich seine Anwendung bei Applikationen für PDA's und Smartphones. Dabei unterliegt die CDC-Konfiguration nicht solchen starken Einschränkungen wie die CLDC. Unter der CDC-Konfiguration wird eine CVM (C Virtual Machine) integriert. Diese VM ist nicht so stark limitiert wie die KVM, sie ist fast genauso umfangreich wie eine herkömmliche JVM (Java SE). [Esc03], [Mos06]
Auf Grund der Hardwareausstattung lassen sich die mobilen Endgeräte in unterschiedliche Gruppen (siehe Kapitel 2.3.1) unterteilen. Die Geräte können sich innerhalb einer Gruppe zusätzlich unterscheiden. Ein Beispiel sind die unterschiedlichen Displaygrößen der Geräte. Es ist möglich für solche Fälle weitere Anpassungen vorzunehmen. Wie bereits erwähnt, wird dem Entwickler dafür die Profil-Ebene bereitgestellt. Durch diese Ebene werden die API's der Konfigurations-Ebene erweitert. Es existieren vier verschiedene Profile. Dazu zählen unter anderem das Fundation Profil, das Personal Profil und das PDA Profil. Diese werden für die Verwaltung der Benutzerschnittstelle, die Speicherung persistanter Daten und die Steuerung des Lebenszyklus einer Anwendung benötigt. Das gebräuchlichste Profil ist das „Mobile Information Device Profile“, kurz MIDP. Wie aus Abbildung 38 ersichtlich setzt MIDP auf die CLDC - Konfiguration auf. Daher ist dieses Profil, wie auch die CLDC - Konfiguration, speziell auf Mobiltelefone ausgelegt. MIDP stellt den Entwickler eine Reihe von API's zur Verfügung. Neben weiteren Funktionen legt dieses Profil die Mindestanforderungen an die Hardware fest. [Esc03], [Mos06], [Use08], [Stu08]

Mit dem MIDP - Profil lassen sich folgende Funktionsbereiche umsetzen [Mos06]:

„- Lieferung von Anwendungen
- Steuerung des Lebenszyklus einer Anwendung
- Signieren und Sicherheitsmodell
- Netzwerkverbindung und -protokolle
- Persistenter Speicher (RecordStore)
- Erzeugung von Tönen
- Timer
- Unterstützung von HTTPS (HTTP verschlüsselt)
- Registrierung von Server-Push-Diensten
- Benutzungsinterface (Bildschirm, Tastatur)“

Die im Zusammenhang mit dieser Arbeit entwickelte Anwendung verwendet die CLDC - Konfiguration 1.1 und das MIDP - Profil 2.0. Programmierte Anwendungen werden unter J2ME als MIDlet's bezeichnet. Das MIDlet besteht dabei aus mindestens einer Klasse und wird auf der Profil Ebene ausgeführt. Dabei dürfen in einem MIDlet nur API's des gleichen Profils verwendet werden. Die Nachfolgende Abbildung 39 zeigt den Lebenszyklus eines solchen MIDlet's.

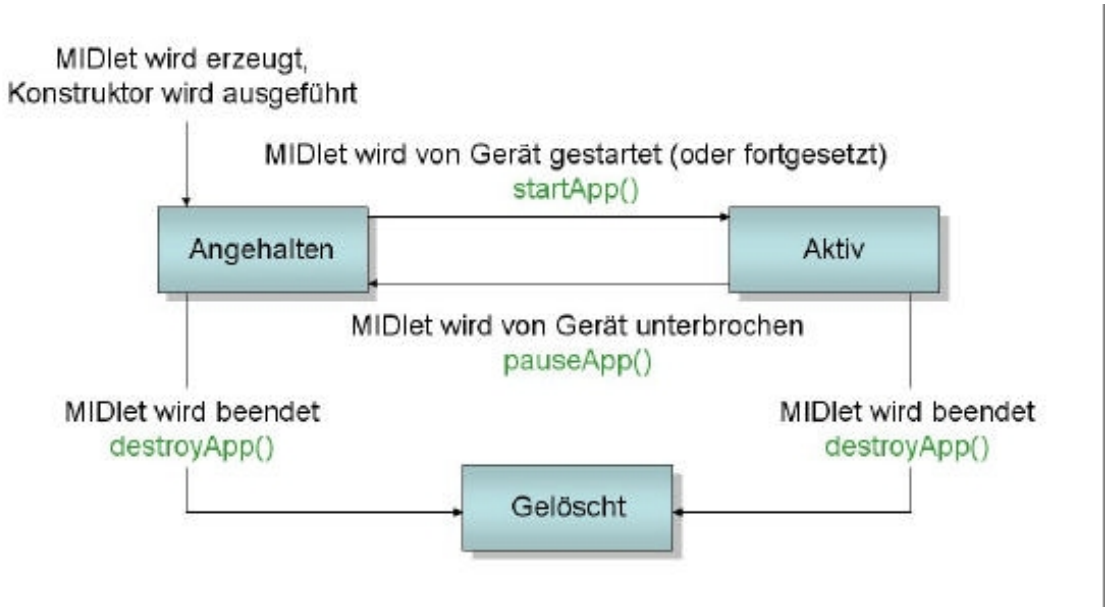


Abbildung 39: Lebenszyklus eines MIDlet's
Quelle: [Töd09]

.

4.2 WTK 2.5.2 (Java Wireless Toolkit)

Wie bereits erwähnt bietet das WTK von Sun den Programmierer von J2ME-Anwendungen eine grafische Benutzerschnittstelle. Dadurch wird das Entwickeln der J2ME-Anwendungn sehr stark vereinfacht.

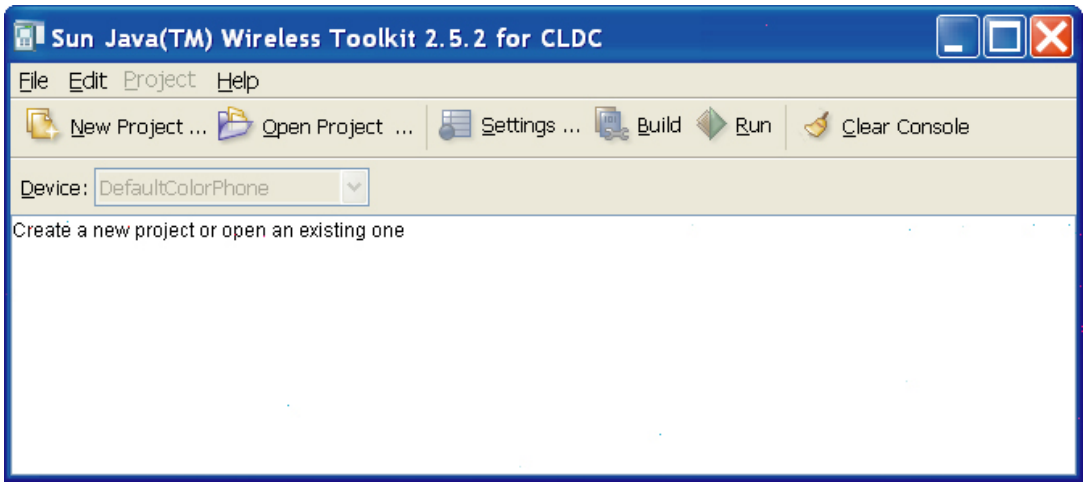


Abbildung 40: Java Wireless Toolkit der Firma Sun.

Das WTK umfasst eine Reihe von Komponenten, die für die Entwicklung von J2ME-Anwendungen benötigt werden. Um eine laufende J2ME-Anwendung zu erhalten, sind verschiedene Schritte nötig. Dazu gehören das Pre-verifizieren und Kompilieren, das abschließende Erstellen eines JAR - Archives und der dazugehörigen JAD - Datei sowie das Ausführen der Anwendung im entsprechenden Emulator. Zusätzlich zu den einzelnen Komponenten stellt das WTK die so genannte *KToolbar* zur Verfügung. Die KToolbar stellt die GUI des WTK dar. Dadurch wird der Zugriff auf die Komponenten vereinfacht. [Esc03], [Mos06]

Emulatoren

Emulatoren (Emulation, dt. Nachbildung) bilden Systeme nach und ahmen diese auf anderen Plattformen nach. Das WTK stellt verschiedene Emulatoren bereit, die die Betriebssysteme mobiler Endgeräte nachbilden. Dadurch wird es dem Programmierer ermöglicht, J2ME-Anwendungen auf den PC mit einem entsprechenden Emulator auszuführen und zu simulieren. So können diese vor der Installation auf dem mobilen Endgerät getestet werden.

Abbildung 41: WTK 2.5.2 Emulatoren
(DefaultColorPhone, DefaultGrayPhone, MediaControlSkin, QwertyDevice)

Zusätzlich zu den bereits vorhandenen Emulatoren können im Internet weitere Emulatoren herruntergeladen werden. Dadurch können unter anderem Handysysteme der Hersteller Nokia, Siemens und Sony Ericsson emuliert werden.
Im WTK 2.5.2 sind standardmäßig vier Emulatoren vorinstalliert. In Abbildung 41 werden die verschiedenen Emulatoren dargestellt. Diese unterscheiden sich in der Darstellung der Displayfarben, sowie den einzelnen Tastenfunktionen. [Esc03], [Mos06]

4.3 Klassendiagramm

Die Abbildung 42 stellt den Entwurf eines Klassendiagramms der in Zusammenhang mit dieser Arbeit entworfenen Software dar.

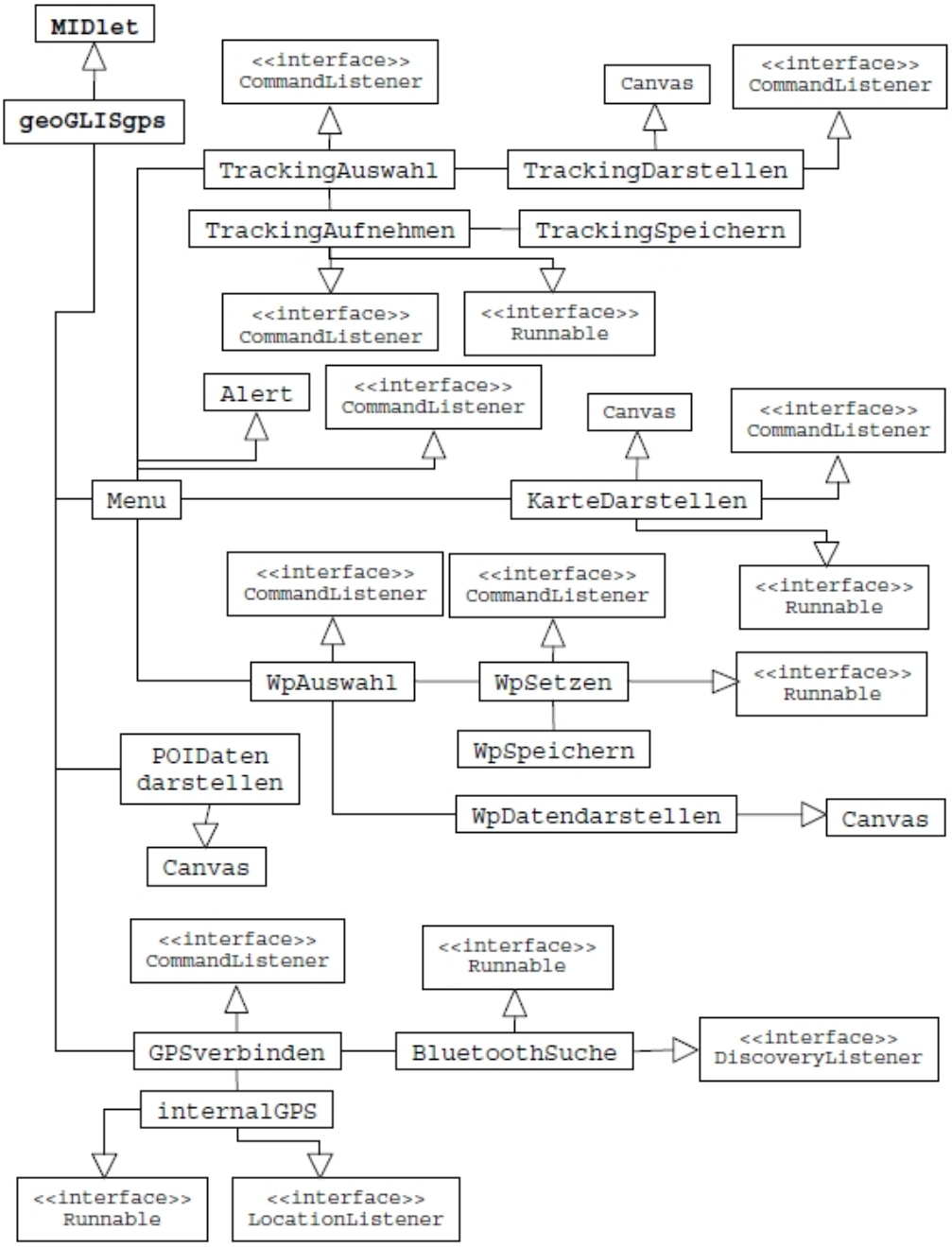


Abbildung 42: Klassendiagramm der Software

4.4 Hauptprogramm

Die Umsetzung der Software auf Basis des Konzeptes sowie die Ausführungen zu den Programmcodes wurde mit folgenden Quellen [Esc03] und [Mos06], [SS09] realisiert. Darüber hinaus wurden sich Anregungen aus den erwähnten Open Source Produkten [Pet09], [Dev09] und [Vlk09] geholt.

Das Hauptprogramm bildet das Grundgerüst der J2ME-Applikation, da dieses MIDlet beim Starten der Anwendung als erstes aufgerufen wird. Aus diesem Grund muss die folgende Grundstruktur in diesem MIDlet vorhanden sein:

- `startApp()`
- `pauseApp()`
- `destroyApp()`

Für das Ansprechen von verschiedenen Anwendungen und Funktionen, ist es notwendig, die drei oben genannten Methoden in das Start - MIDlet zu implementieren. Beim Starten des Programms wird als erstes auf die `startApp()`-Methode zugegriffen, um die Software ordnungsgemäß auszuführen. Darüber hinaus werden durch diese Methode alle durch die J2ME-Anwendung benötigten Ressourcen angefordert.
Es kann bei der Ausführung von Anwendungen auf mobilen Endgeräten zu Unterbrechungen kommen. Dazu zählen etwa das Eintreffen einer SMS oder ein ankommender Anruf. In solchen Situationen ist es notwendig, dass die ausgeführte Anwendung in einen Pause-Zustand wechselt. Aus diesem Grund wird die Methode `pauseApp()`-Methode implementiert. Außerdem sorgt diese Methode dafür, dass die Anwendung nach Beenden der Unterbrechung (Telefonat beendet, SMS empfangen) ordnungsgemäß fortgesetzt wird.
Damit die Applikation ordnungsgemäß beendet werden kann, wird die Methode `destroyApp` benötigt. Durch das Aufrufen dieser Methode werden genutzten Ressourcen wieder freigegeben. Außerdem können innerhalb des Destroy - Zustandes Informationen gespeichert werden. Diese Daten können der Anwendung beim erneuten Ausführen so wieder bereitgestellt werden.
In dem Listing 4.1 wird das Grundgerüst solch eines MIDlet's veranschaulicht. Daraus ist ersichtlich, dass das MIDlet von der Klasse `MIDlet` erbt. Dadurch werden die oben genannten Methoden bereitgestellt, um so die verschiedenen Zustände anzusprechen.
Eine J2ME-Applikation in der mehrere MIDlet's enthalten sind, wird als MIDlet - Suite bezeichnet. Innerhalb solch einer Suite darf nur ein MIDlet existieren, das von der Klasse `MIDlet` erbt.

```
package de.geoglis.geoglisgps;

import javax.microedition.midlet.*;

public class geoGLISgps extends MIDlet{

  public geoGLISgps() { }

  protected void startApp() { }

  protected void pauseApp() { }

  protected void destroyApp(boolean b) { }
}
```

Listing 4.1: Hauptprogramm der MIDlet-Suite

Nach dem Starten des Programms wird die `startApp()`-Methode aufgerufen. In dieser Methode ist der Funktionsaufruf `display(hmenu)` enthalten. Durch diesen Aufruf wird der Zugriff auf ein weiteres MIDlet innerhalb der Suite ermöglicht. Das MIDlet `Menu` ermöglicht es, auf dem Display ein Hintergrundbild sowie ein Benutzermenü einzublenden (siehe Abbildung 43).
Für das Ansprechen der Bildschirme mobiler Endgeräte wird die MIDP-GUI-API verwendet. Dafür steht unter J2ME die Klasse `javax.microedition.lcdui.Display` bereit. Durch diese Klasse wird es ermöglicht, das Display der mobilen Endgeräte anzusprechen. Damit die verschiedenen Funktionen dieser Klasse genutzt werden können, ist es erforderlich, ein Objekt dieser Klasse zu erzeugen.

```
private static Display sdisplay;
```

Außerdem wird durch das erzeugen solch eines Display-Objektes der Displaymanager bereitgestellt. Um das Display-Objekt zu ermitteln, wird die statische Funktion `getDisplay (this)` verwendet. Diese Funktion wird in der Regel innerhalb der `startApp()`-Methode aufgerufen. der Funktionsaufruf sieht in dem Programm selbst wie folgt aus:

```
sdisplay = Display.getDisplay(this);
```

Die Aktuelle Darstellung auf dem Bildschirm wird als `Screen` bezeichnet. Es gibt unter J2ME vier verschiedene Klassen, die als *Screen* fungieren können. Dazu gehören die Klassen `TextBox, List, Alert` und `Form`. Diese Klassen finden auch innerhalb der Applikation ihre Anwendung. Bei der Darstellung von Screens ist zu beachten, das auf dem Display jeweils ein `Screen` angezeigt werden kann.

Abbildung 43: Menüanzeige auf dem Mobiltelefon

Es ist innerhalb der Applikation erforderlich, die Bildschirmanzeige zu aktualisieren. Dafür existiert in der Display-Klasse die Methode `setCurrent()`. Jedem MIDlet ist genau ein Display-Objekt zugeordnet. Aus diesem Grund wird die Methode

```
sdisplay.setCurrent(able);
```

in einer separaten Funktion geschrieben. Dadurch ist es möglich, dass andere MIDlet's darauf zugreifen können, und eine einheitliche Darstellung auf dem Display gewährleistet wird, da innerhalb der Applikation auf nur ein Display-Objekt zugegriffen werden kann. Darüber hinaus kann so jederzeit zu dem Ausgangsbildschirm zurückgekehrt werden.
Wie bereits erwähnt, wird innerhalb des Hauptprogramms `geoGLISgps()` ein MIDlet mit der Klasse `Menu` aufgerufen. Dieses MIDlet erbt von der Klasse `Alert`. Es werden durch Java keine Mehrfachvererbungen unterstützt. Aus diesem Grund können weitere Klassen implementiert werden.

```
public class Menu extends Alert implements CommandListener {}
```

Um innerhalb einer Anwendung auf bestimmte Events reagieren zu können, ist es notwendig, das Interface `CommandListener` zu implementieren.

Wenn dieses Interface in eine Klasse implementiert wird, ist es erforderlich die Methode

```
public void commandAction(Command c, Displayable d)
```

einzubinden. Diese dient zur Behandlung von Ereignissen, da sie bei jedem Auftreten eines Events aufgerufen wird. Innerhalb dieser Methode wird festgelegt welche Reaktion auf ein bestimmtes Ereignis folgt. Dies geschieht über eine normale `if`-Abfrage.
Um auf verschiedene Ereignisse reagieren zu können, müssen diese zuvor festgelegt werden. Dies geschieht über die Klasse `Command`.

```
public Command(String label, int commandType, int priority);
```

Es müssen drei verschiedene Parameter an diese Klasse übergeben werden. Durch den ersten Parameter wird ein `String` festgelegt, der auf dem Display angezeigt werden soll. Mit der Angabe des zweiten Parameters wird der Kommandotyp näher festgelegt. Es werden folgende Typen durch die Klasse bereitgestellt: `BACK, CANCEL, EXIT, HELP, ITEM, OK, SCREEN und STOP`. Der dritte Parameter dient dazu, die Prioritätsstufe des Kommandos festzulegen. Dabei ist die 1 die höchste Prioritätsstufe. Mit dem Steigen des Zahlenwertes nimmt die Priorität ab.
Damit die Ereignisse die zur Wahl stehen auf dem Display dargestellt werden, muss die Methode

```
public void addCommand(Command c)
```

eingebunden werden. Soll nun ein Ereignis durch Auswahl eines Menüpunktes ausgelöst werden, müssen sich diese bei dem `setCommandListener(CommandListener c1)` registrieren. Durch diesen `Listener` kann auf die verschiedene Ereignisse ausgelöst und die jeweilige Reaktion ausgeführt werden.
Ein solches Ereignis kann die Auswahl des Menüpunktes „Beenden“ sein. Wird dieses Event ausgelöst, wird die in dem Hauptprogramm definierte Methode `public static void beenden()` aufgerufen. Durch diese Methode wird es ermöglicht, die Applikation ordnungsgemäß zu beenden. Dazu werden die beiden Funktionsaufrufe

```
geoglis.destroyApp(true);
geoglis.notifyDestroyed();
```

in die J2ME-Applilation integriert.
In dem nachfolgenden Programmausschnitt 4.2 wird veranschaulicht, wie die Umsetzung des `CommandListener` realisiert wird.

```
public class Menu extends Alert implements CommandListener {
  private Command menuBeenden;
  [....]

  public Menu() {
    [....]
    addCommand(menuBeenden = new Command("Beenden", Command.EXIT, 0));
    setCommandListener(this);
  }

  public void commandAction(Command c, Displayable d) {
    if (c == menuBeenden) {
      geoglis.beenden();
    }else if(c == menuKarte){
      [....]
    }
  }
}
```

Listing 4.2: CommandListener

Für das Einbinden und Darstellen des Menüs und des Hintergrundbildes wird die Klasse `Alert` eingebunden. Dies geschieht, indem das MIDlet von dieser Klasse erbt. Dadurch ist ein schneller Zugriff auf die Funktionen des `Alert's` möglich. Ein `Alert` wird verwendet, um bestimmte Informationen oder Bilder auf dem Display auszugeben. Für das Einbinden eines Bildes innerhalb eines `Alert's` wird folgender Programmcode verwendet:

```
setImage(Image.createImage("/geoglis.png"));
```

Zum Darstellen eines Bildes, wird zunächst ein Objekt der Klasse `Image` gebildet. In den oberen Programmcode, ist `setImage()` eine Funktion des `Alert's`, die für das darstellen von Bildern notwendig ist. Zur Generierung eines Bildes wird die Methode `createImage()` der Klasse `Image` benötigt. Um das gewünschte Bild darzustellen wird der Methode der Bildpfad übergeben. Dieser Pfad bezieht sich lediglich auf die Bilder, die im Projektverzeichnis enthalten sind.

4.5 Verbindung zu einen GPS-Empfänger

Der Nutzer soll mit der entwickelten Anwendung die Möglichkeit haben, die aktuelle GPS-Position zu bestimmen. Dies soll wie bereits erwähnt auf zwei verschiedene Arten möglich sein. Es soll der Zugriff auf die GPS-Daten über einen Bluetooth GPS-Empfänger (GPS-Maus) stattfinden, wenn das mobile Endgerät über keinen internen Empfänger verfügt. Besitzt das Gerät jedoch einen integrierten GPS-Empfänger, soll dazu eine entsprechende Schnittstelle aufgebaut werden.

Wie aus der Abbildung 43 und dem Konzept im Kapitel 3.1 ersichtlich, kann innerhalb des Hauptmenüs der Anwendung der Menüpunkt „GPS“ aufgerufen werden. Anschließend wird auf dem Display ein Untermenü dargestellt. In der Abbildung 44 wird der Quellcode mit der dazugehörigen Displayausgabe dargestellt. Um die Menüauswahl auf dem Display zu generieren wird eine „Listendarstellung“ gewählt. Dies wird realisiert, in dem das MIDlet `GPSverbinden` von der Klasse `List` erbt. Als Typ für die Liste wird *IMPLICIT* gewählt, da es sich um eine einfache Listendarstellung handelt. Darüber hinaus ist es mit diesem Listentyp möglich, verschiedene Ereignisse auszulösen. Damit auf diese Ereignisse eine entsprechende Reaktion stattfinden kann, wird in das Hauptmenü als Interface der `CommandListener` implementiert. Ein Ereignis wird ausgelöst, indem der Nutzer seine Menüwahl bestätigt. Anschließend wird die Methode `CommandAction()` aufgerufen. Mit Hilfe von `getSelectedIndex()` wird der Indexwert des Listenobjektes bestimmt. Danach wird das jeweilige Ereignis ausgeführt.
Bei der Wahl des Menüpunktes „internes GPS“ beträgt der Index 0. Mit Hilfe einer „Switch-Case-Anweisung“ wird es ermöglicht, dem Indexwert ein Ereignis zu zuordnen.

```
private static final String GPSauswahl[]={"internes GPS","Bluetooth GPS"};

public GPSVerbinden() {
  super("GPS verbinden", List.IMPLICIT, GPSauswahl, null);

  addCommand(zuruck = new Command("Zurück", Command.BACK, 2));
  setCommandListener(this);
}
public void commandAction(Command c, Displayable d) {
  if (c == List.SELECT_COMMAND){
    switch (getSelectedIndex()) {
      case 0:
        geoglis.display(new internalGPS());
          break;
      case 1:

      [...]

}
```

Liste definieren

Liste setzen

Aufruf, wenn Listeneintrag gewählen

Funktionsaufruf

Abbildung 44: Auswahl des Verbindungstyp zu einem GPS-Empfänger

Nachdem durch den Nutzer eine Auswahl getroffen wurde, wird ein weiteres MIDlet innerhalb der MIDlet-Suite aufgerufen. Durch dieses MIDlet kann eine Verbindung zu einem GPS-Empfänger hergestellt werden. In der Abbildung 44 wird beispielhaft der Aufruf des Menüpunktes „internes GPS“ gezeigt. Diese Auswahl ruft die Klasse `internalGPS` auf. Nachdem Aufrufen der Klasse stellt diese anschließend eine Verbindung zu einem geräteinternen GPS-Empfänger her.
In dem nachfolgenden Abschnitt soll zunächst auf die Anbindung eines Bluetooth GPS-Empfängers eingegangen werden.

Bluetooth - Schnittstelle

Um eine Schnittstelle zu einem Bluetooth Gerät aufzubauen stellt J2ME eine spezielle „Bluetooth - API“ (JSR-82, [Sun09b]) zur Verfügung. Da bereits eine Vielzahl von Bluetooth - Anwendungen existieren, wird diese bei der Softwareumsetzung nicht realisiert. Bei der Umsetzung wird das Augenmerk auf die Anbindung eines internen GPS-Empfängers gelegt.
Es soll jedoch nachfolgend auf die Funktionsweise der Bluetooth - API eingegangen werden. Dabei wird sich an einem Bluetooth Beispiel der Entwicklerseite von Sony Ericsson orientiert [AB09]. Dieses Beispiel wurde von „Oscar Vivall“ entwickelt.
Für die Beschreibung der Bluetooth - Schnittstelle wurde auf folgende Quellen zurückgegriffen: [Mos06], [Sun09b], [Wil09].

Bevor zu einem Bluetooth - Gerät eine Verbindung hergestellt werden kann, muss nach Geräten, die sich in unmittelbarer Nähe zu dem eigenen Gerät befinden, gesucht werden. Dazu ist es zunächst notwendig, auf die Einstellungen des Bluetooth - Gerät im eigenen mobilen Endgerät zuzugreifen. Dies geschieht durch den Zugriff auf die Klasse `LocalDevice`.

```
LocalDevice localdevice = LocalDevice.getLocalDevice();
```

Auf die verschiedenen Suchalgorithmen, die die API bereitstellt, können durch die Klasse `DiscoveryAgent` zugegriffen werden. Darüber hinaus, kann durch diese Klasse, die Suche nach den Bluetooth - Geräten gestartet werden. Das Listing 4.3 zeigt einen Ausschnitt über das Starten der Suche nach einen Bluetoot - Gerät.
Für das Überwachen von Ereignissen (Geräte finden, Service finden) wird der `Discovery Listener` als Interface implementiert. Sobald ein Gerät gefunden wurde, reagiert der `DiscoveryListener` auf dieses Ereignis, indem er das Gerät an die Applikation übergibt, durch die die Suche ausgelöst wurde.

```
[...]

private DiscoveryAgent discoveryAgent;

[...]

run{
   [...]
   LocalDevice localDevice = LocalDevice.getLocalDevice();

   //suche nach BT-Gerät starten
   discoveryAgent = localDevice.getDiscoveryAgent();
   discoveryAgent.startInquiry(DiscoveryAgent.GIAC, this);
   [...]
 }

 [...]
```

Listing 4.3: Bluetooth-Gerätesuche starten

Nachdem die Gerätesuche abgeschlossen ist, wird eine Servicesuche gestartet. Diese ist erforderlich, um festzustellen, welche Dienste durch die gefundenen Geräte bereitgestellt werden. Durch die Servicesuche wird eine URL ermittelt. Mit dieser Service - URL wird es ermöglicht, die jeweiligen Dienste des Gerätes einzubinden. Der im Listing 4.3 dargestellte Quelltext wird dazu um folgende Anweisung erweitert:

```
discoveryAgent.searchServices(null, uuidSet, btDevice, this);
```

Mittels dieser Funktion wird die „Transaktion-ID“ zurück geliefert, durch welche der Dienst eindeutig identifiziert wird. Die oben beschriebene ServiceURL erhält man durch das aufrufen der Methode `getConnectionURL()`.
Nachdem erfolgreichen beenden beider Suchen, kann eine Verbindung mit dem Dienst hergestellt werden. Handelt es sich bei dem gefundenen Bluetooth - Gerät um eine GPS-Maus, können anschließend die GPS - Daten durch die Anwendung empfangen und ausgewertet werden. Genau wie die beiden Suchen, wird dieser Vorgang in einem `Thread` ausgeführt. Dafür ist es notwendig, das `Runnable`-Interface einzubinden. Nachdem Start des `Threads`, wird die `run()`-Methode aufgerufen. Durch diese Methode kann der `Thread` gesteuert werden. In dem Listing 4.4 ist die Definition und das implizieren des `run()` durch die `start()`-Methode dargestellt.

```
[...]
Thread t = new Thread(this);\\
t.start();\\
[...]
run{
   [...]
}
```

Listing 4.4: Thread starten

In dem Fall eines GPS - Empfänger wird das empfangen von GPS-Daten gesteuert. Nachdem Aufrufen der `run()`-Methode wird ein `StreamConnection` geöffnet. Die nachfolgende Anweisung ist dafür zuständig, um den Datenstrom zum Empfangen der Daten zu öffnen.

```
StreamConnection conn = (StreamConnection)Connector.open(ServiceURL);
```

Anschließend wird mit einen `InputStream` dafür gesorgt, das die Daten zum Lesen geöffnet werden. Danach können die Daten mit Hilfe der `read()`-Methode ausgelesen werden.

```
InputStream inputstream = conn.openInputStream(); inputstream.read();
```

Interner GPS-Empfänger

Im Gegensatz zu der Anbindung einer GPS-Maus über eine Bluetooth - Schnittstelle, ist der Zugriff auf einen integrierten GPS-Empfänger deutlich einfacher. Die J2ME-Plattform stellt dafür die „Location - API“ (JSR-79) bereit. Die Verwendung der API ist nur möglich, wenn auf dem Endgerät mindestens die CLDC 1.1 und die MIDP 2.0 vorhanden ist. Beispiele für solche Endgeräte sind unter anderem das Sony Ericsson W760i und das Nokia N95. Im nachfolgenden wird auf die Funktionsweise der Location - API im Einzelnen näher eingegangen.
Die Umsetzung und die Beschreibung der Location - API erfolgte mit Hilfe folgender Quellen: [Mos06], [uSO09], [Fre09], [Inc09].

Bevor eine Lokalisierung gestartet werden kann, ist es notwendig die Anforderungen an den Dienstanbieter (`LocationProvider`) zu definieren. Dafür wird durch die Location - API der Klasse `Criteria` bereitgestellt. Mit Hilfe dieser Klasse kann unter anderem festgelegt werden, mit welcher Genauigkeit die Positionsbestimmung erfolgen soll (in Meter). Des Weiteren kann der Energieverbrauch (hoch, mittel, niedrig) bei der Lokalisierung bestimmt werden. Eines der wichtigsten Kriterien, ist die Festlegung der Kosten, die bei der Lokalisierung entstehen dürfen.
Die definierten Anforderungen werden anschließend an den `LocationProvider` übergeben. Nachdem geprüft wurde, welcher Provider die Kriterien erfüllt, wird die Positionsbestimmung gestartet. Damit die Position aktualisiert werden kann, wird durch die API das Interface `LocationListener` bereitgestellt. Zum Aufrufen des `Listener`, wird die `setLocationListener()`-Methode benötigt. Diese wird innerhalb der `run()`-Methode ausgeführt.

```
setLocationListener(LocationListener listener, int interval,
                    int timeout, int maxAge);
```

Der Methode `setLocationListener()` werden vier Parameter übergeben. Durch den ersten Parameter wird der `Listener` registriert. Außerdem kann bestimmt werden, in welchem Zeitintervall (in Sekunden) der `LocationListener` aufgerufen werden soll. Dazu dient der zweite Parameter. Durch den dritten Parameter wird bestimmt, nach welcher Zeit die Suche nach einem entsprechenden Provider abgebrochen werden soll. Mit dem vierten und letzten Parameter wird festgesetzt, wie alt ein Positionsdatensatz maximal sein darf.
Das Setzen der Kriterien des Provider sowie des `LocationListener` erfolgt innerhalb der `run()`-Methode des `Threads`, in dem die Lokalisierung ausgeführt wird. Dies wird mit dem Listing 4.5 veranschaulicht.

```
try{
  //Kriterien für die Auswahl einer Methode zur Positionsbestimmung setzen
  Criteria Kriterien = new Criteria();
  //wie genau (in Metern) müssen Breiten- und längengrade bestimmt werden
  Kriterien.setHorizontalAccuracy(10);
  Kriterien.setVerticalAccuracy(10);
  //legt fest, ob bei der Lokalisierung kosten entstehen dürfen
  Kriterien.setCostAllowed(false);

  LocationProvider Provider = LocationProvider.getInstance(Kriterien);
  Location location = Provider.getLocation(60);

  if(location != null){
    Provider.setLocationListener(this, 10, -1, -1);
  }
}catch(Exception ex){
  setString(ex.toString());
```

Listing 4.5: Lokalisierung starten

Zum Bestimmen der Positionsinformationen wird auf die `Location`-Klasse zugegriffen. Zu diesen Informationen gehören etwa der Zeitstempel (`getTimestamp()`), die momentane Geschwindigkeit (`getSpeed()`in m/s) und die Richtungsorientierung (`getCourse()`). Mit der `getQualifiedCoordinates()`-Methode werden die GPS-Koordinaten abgefragt. Die Angabe der Position erfolgt in WGS84-Koordinaten. Die Ermittlung der GPS-Position wird in dem Listing 4.6 aufgeführt. Nachdem die gewünschten Informationen ermittelt wurden, können diese auf dem Display ausgegeben oder an die aufrufende Anwendung übergeben werden.
Das Ausgeben der Koordinaten wird mit der Klasse `Canvas` realisiert. Sobald sich die Positionsdaten ändern, werden diese durch Aufrufen der `repaint()`-Methode neu gezeichnet.

```
public void locationUpdated(LocationProvider Provider, Location location) {

  Coordinates Koord = location.getQualifiedCoordinates();

  if(Koord != null){
    String breitengrad = Double.toString(Koord.getLatitude());
    String laengengrad = Double.toString(Koord.getLongitude());
    String hoehe = Double.toString(Koord.getAltitude());

    String geschwindigkeit = Double.toString(location.getSpeed());
    String richtung = Double.toString(location.getCourse());
    String zeitstempel = Double.toString(location.getTimestamp());

    [...]
    //Darstellung auf dem Display
  }
}
```

Listing 4.6: Aktualisieren der Position mittels Methode: `locationUpdated()`

Es ist noch zu erwähnen, dass die Positionsbestimmung auch ohne einen `LocationListener` stattfinden kann. Dies ist sinnvoll, wenn die Position in einer anderen Klasse benötigt wird, wie etwa beim Darstellen der Position innerhalb einer Karte.
Neben der Lokalisierung des mobilen Endgerätes über das GPS-System erlaubt die Location - API auch die Lokalisierung durch Mobilfunk-gestützte Ortung. Dadurch ist es außerdem möglich mit der API ohne einen integrierten GPS-Empfänger die Position zu bestimmen. Darüber hinaus ist es mit dieser API möglich den „Location Based Service“ zu realisieren.

4.6 Einbinden der Daten

4.6.1 Öffnen von Dateien

Wie bereits erwähnt soll es mit der Software möglich sein, verschiedene Daten in die Anwendung einzubinden. Um den Import der Daten zu ermöglichen, ist es zunächst einmal erforderlich, einen Zugriff auf das gewünschte Datenmaterial herzustellen.
Diese Daten befinden sich in der Regel auf dem Dateisystem der mobilen Endgeräte. Damit die gewünschten Daten geladen werden können, muss eine Schnittstelle zwischen der Anwendung und dem Dateisystem des Endgerätes hergestellt werden. Dafür stellt die J2ME-Plattform, die „FileConnenction-API“ (JSR-75, [Sun09a]) bereit.
Auf einem mobilen Endgerät existieren zwei Speicher. Zum einen der interne Telefonspeicher, zum anderen der Speicher, der durch eine Speicherkarte (SD-Karte) zusätzlich zu dem Telefonspeicher bereitgestellt werden kann. Der Zugriff auf den Telefonspeicher und der Speicherkarte erfolgt unter der Angabe des Laufwerkbuchstaben (Telefonspeicher: „c“; Speicherkarte: „e“), bzw. der Angabe des Laufwerknamens („Telefonspeicher“; „Speicherkarte“).
Durch die Verwendung der `FileConnection` ist es möglich, die Größe des .jar-Files gering zu halten, da so benötigte Daten ausgelagert werden können. Dies ist besonders bei dem Kartenmaterial von großen Vorteil. Dadurch kann es problemlos erweitert, bzw. ausgetauscht werden. Es ist jedoch darauf zu achten, dass die zu ladende Karte eine bestimmte Dateigröße nicht überschreiten dürfen, da sonst die Kartendaten nicht geladen werden können (Kartenkacheln $<$ 1MB).
In Zusammenhang mit dieser Arbeit soll das Datenmaterial unter einem fest definierten Pfad (z.B. Verzeichnis = ”file:///Speicherkarte/geoGLISgps/Karten/”) abgelegt werden. Sollte dieser nicht existieren, soll er entweder durch den Nutzer oder durch die Anwendung selbst angelegt werden.
Das Listing 4.7 zeigt, wie ein Zugriff auf ein Verzeichnis hergestellt werden kann. Anschließend wird der Inhalt des Verzeichnisses auf dem Display angezeigt (Abbildung 45). Wie aus dem Listing 4.7 ersichtlich, wird zunächst ein `FileConnection` - Objekt definiert und initialisiert. Anschließend wird mit Hilfe dieses Objektes das Verzeichnis zum Lesen geöffnet. Dies geschieht durch `Connector.open(verzeichnis)`. Danach kann der Inhalt des Verzeichnisses ausgelesen werden. Dafür wird im Voraus ein `Enumeration` definiert. Dort wird der Inhalt der `FileConnection` abgelegt. Anschließend wird der Inhalt aus der `Enumeration`- Liste ausgelesen. Dies geschieht mit Hilfe einer `While`-Anweisung. Innerhalb dieser Anweisung wird geprüft, ob die Liste noch ein weiteres Element enthält. Sollte das der Fall sein, wird der Dateiname auf einer `String`-Variable gespeichert. Nachdem ein Dateiname gefunden wurde, wird geprüft, ob dieser auf „.png“ endet. Dadurch werden alle nicht benötigten Dateien aussortiert und dem Nutzer nicht angezeigt. Nachdem Zugriff auf das `FileConnection`-Objekt, wird dieses anschließen wieder geschlossen.

```
Enumeration eFileList;
String tmpDir
FileConnection fc = (FileConnection)Connector.open( verzeichnis);

if( fc.isDirectory() ) {
   eFileList = fc.list();
   while( eFileList.hasMoreElements() ) {
     tmpDir = ((String)eFileList.nextElement());
     if( tmpDir.endsWith( ".png" )) {
       //Verzeichnisinhalt auf Display anzeigen
       append(" " + tmpDir + "\n");
     }
   }
 }
 fc.close();
```

Listing 4.7: Verzeichnisinhalte anzeigen

Bevor jedoch eine Verbindung zu dem Dateisystem hergestellt werden kann, muss der Zugriff durch den Nutzer bestätigt werden. Dies ist in der Abbildung 45 dargestellt.
Der im Listing 4.7 enthaltene Ablauf dient lediglich dazu, Inhalte auf dem Display darzustellen. Damit eine Lesezugriff auf eine Datei augeübt werden kann, muss beim Öffnen der `FileConnection` neben dem Verzeichnispfad ein weiter Parameter übergeben werden.

```
fc = (FileConnection) Connector.open (verzeichnis, Connector.READ);
```

Abschließend können die verschiedenen Daten durch die Anwendung gelesen und auf dem Display dargestellt werden. In den nachfolgenden Kapiteln 4.6.2 und 4.6.3 wird auf das Einbinden und die Darstellung der einzelnen Geodaten näher eingegangen.

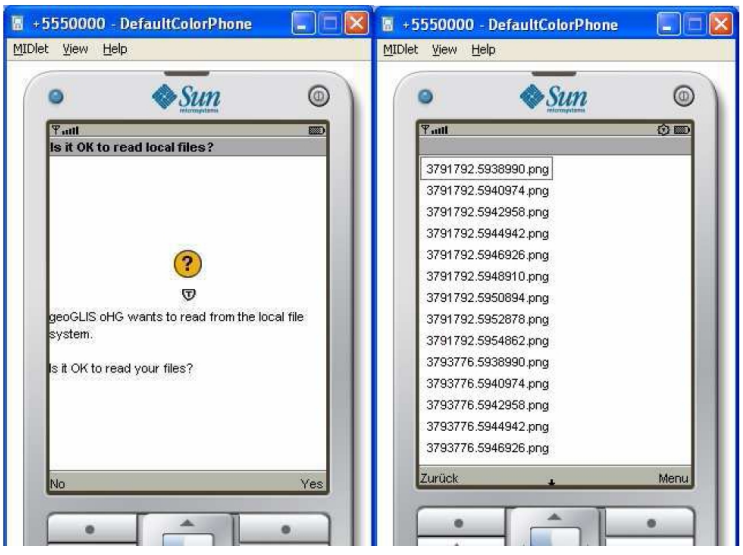


Abbildung 45: Zugriff auf Dateisystem mobiler Endgeräte

4.6.2 Karte

Für das Darstellen des georeferenzierten Kartenmaterials ist es erforderlich, eine „.png“-Datei sowie die dazugehörige „.pgw“ - Datei zu öffnen. Dieses Kartenmaterial soll durch den Nutzer auf der SD-Speicherkarte des mobilen Endgerätes unter dem Verzeichnis „/Speicherkarte/geoGLISgps/Karten/...“ abgelegt werden. Um eine Verbindung zu dem Verzeichnisinhalt herzustellen, wird wie im vorherigen Kapitel 4.6.1 beschrieben, die `File Connection` - API zur Hilfe genommen.
Die Auswahl der benötigten Kachel soll dabei dynamisch erfolgen. Dazu ist es notwendig, dass die aktuelle Position durch einen entsprechenden GPS-Empfänger bestimmt wird. Nachdem die Positionsdaten abgerufen wurden, wird geprüft, in welcher Kachel sich die aktuelle Position befindet. Dazu ist es erforderlich, die Bounding Box (dt.: Zeichen-Box) zu bestimmen. Die Bounding Box wird durch die Angaben der linken oberen Eckkoordinate sowie der rechten unteren Eckkoordinate der Kachel definiert. Um eine Aussage über diese Koordinaten treffen zu können, müssen diese zunächst ermittelt werden. Für das Berechnen dieser Daten ist der Zugriff auf die „.pgw“ - Datei nötig. Auf den Aufbau dieser Datei wurde bereits in dem Kapitel 2.4.1 kurz eingegangen. In der nachfolgenden Aufzählung wird der Inhalt Zeile für Zeile erläutert:

1. Zeile - Dimension (in Koordinateneinheit) eines Bildpixels in X-Richtung
2. Zeile - Drehung/Verzerrung in X
3. Zeile - Drehung/Verzerrung in Y
4. Zeile - Dimension (in Koordinateneinheit) eines Bildpixels in Y-Richtung
5. Zeile - Startkoordinate links oben in X
6. Zeile - Startkoordinate links oben in Y

Wie aus dieser Aufzählung hervorgeht, lässt sich die Koordinate der linken oberen Ecke durch die beiden letzten Werte in der „.pgw“-Datei beschreiben. Um nun die linke untere Ecke zu berechnen, ist es notwendig die Ausdehnung der Karten in X- bzw. in Y-Richtung zu kennen. Diese Werte sind in der ersten und vierten Zeile der „.pgw“-Datei zu finden. In dem Koordinatensystem WGS84 gibt die Dimension eines Bildpunktes Auskunft darüber, welche räumliche Ausdehnung ein Pixel besitzt. Zusätzlich muss die Kachelgröße in Pixel bekannt sein.

$$xRechtsunten = AusdehnungX * Bildmaße + xLinksoben$$

$$yRechtsunten = AusdehnungY * Bildmaße + yLinksoben$$

Bezieht man die Kartenmaterial von der „geoGLIS oHG“-Website [go09b] erhält man die Kacheln mit den Bildmaßen 1500 x 1500 Px. Dadurch besitzen diese Kacheln eine Dateigröße von über 1 MB. Es wurde versucht diese Kacheln mit der mobilen Applikation zu öffnen und darzustellen. Dabei wurde festgestellt, dass die Bilder die Dateigröße von 1 MB nicht übersteigen dürfen. Aus diesem Grund war es am Anfang nicht möglich, die Karten auf dem Display des mobilen Endgerätes darzustellen. Nach Rücksprache mit der Firma „geoGLIS oHG“, wurden daraufhin angepasstes Kartenmaterial mit den Abmaßen 500 x 500 Px bereitgestellt.
Nachdem die Eckkoordinaten berechnet wurden, wird geprüft, ob sich die aktuelle Position innerhalb dieser Kachelkoordinaten befindet. Dies geschieht über folgende Abfrage:

$$xGPSPos >= xLinksoben$$

$$xGPSPos <= xRechtsunten$$

$$yGPSPos >= yLinksoben$$

$$yGPSPos <= yRechtsunten$$

Sollte die durch einen GPS-Empfänger ermittelte Position außerhalb dieser Kachel liegen, wird die nächste Kachel in dem Verzeichnis auf dieselbe Weise überprüft. Sollte keine der vorhandenen Kacheln diese Kriterien erfüllen, soll dem Nutzer eine entsprechende Mitteilung auf dem Display angezeigt werden.
Bei einer erfolgreichen Suche wird die Kartenkachel aus dem Dateisystem geladen und auf dem Bildschirm dargestellt. Dazu übermittelt man den Bildnamen der analysierten Kachel an die `FileConnection`. Anschließend ist es notwendig, den Bytecode des Bildes mit Hilfe eines `InputStreams` aufzunehmen. Danach wird dieser `Stream` einer Variable vom Typ `Image` übergeben. Abschließend wird die Methode `paint()` aufgerufen, um die Karte zu zeichnen. Damit diese Methode verwendet werde kann, ist es notwendig, dass das MIDlet von der Klasse `Canvas` erbt. Dieser Ablauf wird in dem Listing 4.8 näher dargestellt.
Damit die Karte auf dem Display platziert werden kann, sind einige Berechnungen notwendig. Das Platzieren der Karte ist erforderlich, damit die angezeigte GPS-Position mit der Position auf der Karte übereinstimmt. Dazu ist es zunächst notwendig, die GPS-Koordinaten in Pixelkoordinaten des Bildschirms umzuwandeln. Anschließend kann die Position der Karte auf dem Display ermittelt werden. Auf die einzelnen Rechenschritte, wird im nachfolgenden näher eingegangen.

```
 [...]
//Dateisystem zum lesen der Daten öffnen
FileConnection fc = (FileConnection)Connector.open(Bildname, Connector.READ);
//Datei Input Stream öffnen
InputStream fis = fc.openInputStream();
Karte = Image.createImage(fis);
//aufrufen der paint()-methode
repaint();
[...]
paint(Graphics g){
 [...]
 //zeichnen der Karte und der Position auf den Bildschirm
 g.drawImage(Karte, xKartenposition, yKartenposition, Graphics.LEFT | Graphics.
     TOP);
 g.drawImage(position,xDisplayPixelPosition, yDisplayPixelPosition, Graphics.
     HCENTER | Graphics.VCENTER);
 [...]
}
```

Listing 4.8: Karte Laden und anschließendes Zeichnen der Karte

Für das Ermitteln der Bildschirmpixelkoordinaten, wird als erstes der Auflösungsfaktor berechnet. Dieser gibt die Koordinatenausdehnung in Grad je Bildschirmpixel an. Für die Berechnung werden die zuvor bestimmten Eckkoordinaten benötigt sowie die Displaybreite bzw. -höhe. Innerhalb eines MIDlet's, dass von der Klasse `Canvas` erbt, erfolgt das bestimmten der Displayhöhe und der -breite durch die beiden Methoden `getWidth()` und `getHeight()`.

$$Aufloesungsfaktor X = \frac{xRechts - xLinks}{Displaybreite}$$

$$Aufloesungsfaktor Y = \frac{yUnten - yOben}{Displayhoehe}$$

Nachdem der Auflösungsfaktor bestimmt wurde, kann der Abstand zu dem oberen und dem rechten Displayrand bestimmt werden. Dazu wird neben den rechten x-Koordinate und der oberen y-Koordinate auch die Angabe der aktuellen GPS-Position benötigt. Dadurch ist es möglich, die Pixelposition auf dem Display des mobilen Endgerätes zu bestimmen.

$$AbstandDisplayRechts = \frac{xGPSPos - xRechts}{Aufloesungsfaktor X}$$

$$xDisplayPixelPosition = Displaybreite - AbstandDisplayRechts$$

$$AbstandDisplayUnten = \frac{yGPSPos - yUnten}{Aufloesungsfaktor Y}$$

$$yDisplayPixelPosition = Displayhoehe - AbstandDisplayUnten$$

Nachdem die Pixelkoordinaten für das Display des mobilen Endgerätes bestimmt worden sind, kann die Karte platziert werden. Dazu ist es notwendig, die GPS-Position in Pixelkoordinaten der Kartenkachel umzuwandeln. Dies geschieht auf die gleiche Weise, wie auch bei der Pixelkoordinate für das Display. Der Auflösungsfaktor muss jedoch nicht mehr berechnet werden, da dieser Wert aus der PNG - Datei entnommen werden kann.

$$Aufloesungsfaktor X = DimensionBildpunktX$$

$$Aufloesungsfaktor Y = DimensionBildpunktY$$

$$AbstandKachelRechts = \frac{xGPSPos - xRechts}{Aufloesungsfaktor X}$$

$$xKachelPixelPosition = Kachelbreite - AbstandKachelRechts$$

$$AbstandKachelUnten = \frac{yGPSPos - xRechts}{Aufloesungsfaktor X}$$

$$yKachelPixelPosition = Kachelhoehe - AbstandKachelUnten$$

Damit anschließend die Karte positioniert werden kann, wird die Differenz zwischen den ermittelten Pixelkoordinaten der Kachel und des Display berechnet.

$$xKartenposition = xKachelPixelPosition - xDisplayPixelPosition$$

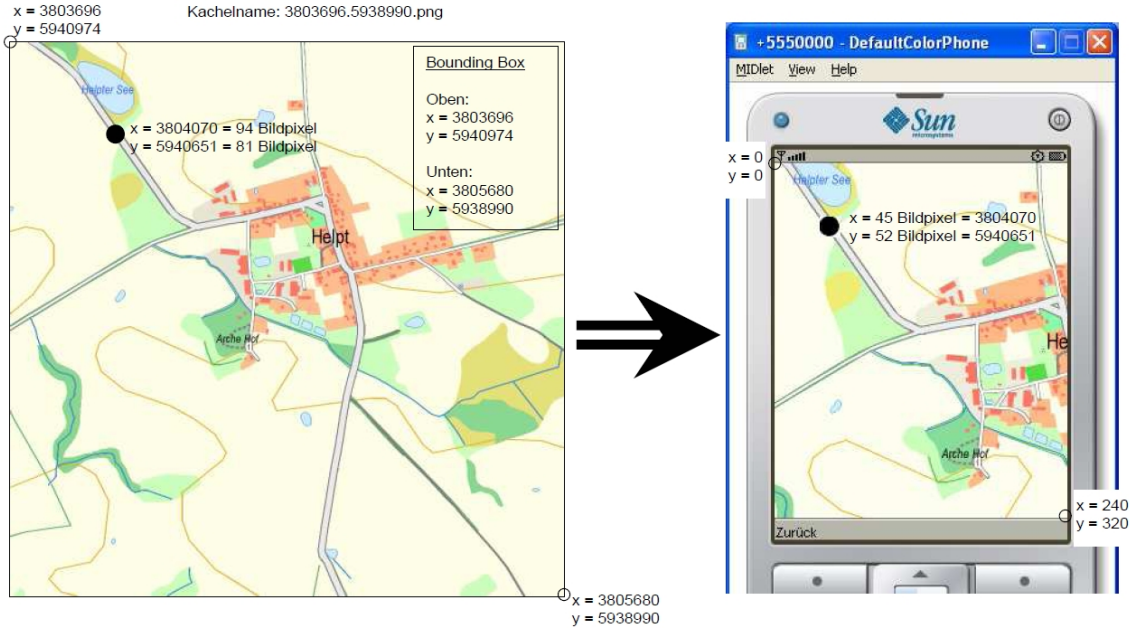
$$yKartenposition = yKachelPixelPosition - yDisplayPixelPosition$$

Die ermittelten Werte werden abschließend als Startpunkt für das Kartenzeichnen gesetzt. Das Setzen der ermittelten Werte für die GPS-Position, sowie die Kartenposition wird in dem Listing 4.9 dargestellt.

```
  [...]
g.drawImage(Karte, xKartenposition, yKartenposition, Graphics.LEFT | Graphics.TOP
    );
g.drawImage(position,xDisplayPixelPosition, yDisplayPixelPosition, Graphics.
    HCENTER | Graphics.VCENTER);
  [...]
```

Listing 4.9: Zeichnen der ermittelten Pixelkoordinaten

Bei den Berechnungen ist zu beachten, dass die GPS-Position durch den Empfänger in WGS84 - Koordinaten geliefert werden. Dabei erfolgt die Angabe der Koordinaten als Dezimalzahl. Diese Zahl muss auf 5 Stellen nach dem Komma gerundet werden und anschließend in eine ganze Zahl umgewandelt werden. Dies geschieht, indem man den ermittelten Wert mit 100000 multipliziert. Dadurch wird gewährleistet, dass die durch den GPS-Empfänger ermittelte Position mit den Koordinatenangaben in der PGW - Datei übereinstimmen. Außerdem ist beim Beschaffen der Kartenkacheln darauf zu achten, das diese in WGS84-Koordinaten vorliegen.



ATKIS (R) Basis-DLM, (C) Vermessungsverwaltungen der Länder und BKG 2005

Abbildung 46: Beispiel für die Positionbestimmung auf einer Kartenkachel

Das Zeichnen der Karte und der Position auf dem Display soll in einer `Thread`-Umgebung stattfinden, damit bei Änderungen der GPS-Koordinaten die Position auf dem Display neu berechnet werden kann. Anschließend ist es dadurch möglich, den darzustellenden Kartenausschnitt neu zu bestimmen. Zum neu Zeichnen des Displays ist es erforderlich innerhalb der `run`-Methode des `Threads` die Methode `repaint()` aufzurufen. Diese Methode sorgt dafür, dass `paint()` erneut aufgerufen wird und somit die entsprechenden Daten neu gezeichnet werden.

Wie bereits erwähnt, war es bei der Suche nach der darzustellenden Kachel angedacht, auf die gleichnamige PGW - Datei zuzugreifen. Nachdem versucht wurde dies mit J2ME zu realisieren, wurde festgestellt, dass es nicht möglich ist, die Datei Zeilenweise auszulesen. Es war lediglich möglich, den Inhalt mit einen `InputStream` auszulesen und anschließend auf dem Display darzustellen.
Es wurde bisher keine Möglichkeit gefunden, die benötigten Werte für die Berechnungen aus der PGW - Datei zu beziehen. Da diese Daten jedoch zwingend für die Berechnung der „Bounding Box“ benötigt werden, ist es sinnvoll, diese Daten in eine XML - Struktur zu schreiben. Anschließend können diese Daten mit Hilfe eines XML - Parsers ausgelesen und auf eine Variable abgelegt werden. Ein möglicher Aufbau solch einer XML - Struktur ist im Listing 4.10 dargestellt.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes">
  <Kachelinformationen>
    <Kachelname>....</Kachelname>
    <Ausdehnung>
      <xRichtung>....</xRichtung>
      <yRichtung>....</yRichtung>
    </Ausdehnung>
    <Drehung>
      <inX>....</inX>
      <inY>....</inY>
    </Drehung>
    <Startpunkt>
      <inX>....</inX>
      <inY>....</inY>
    </Startpunkt>
  </Kachelinformationen>
```

Listing 4.10: Aufbau einer XML-Struktur

Durch einen XML - Parser kann auf die in der XML - Struktur enthaltenen Elemente ein direkter Zugriff ausgeübt werden. Für die J2ME werden dem Entwickler bereits verschiedene XML - Parser [Hau09] bereitgestellt, die den Anforderungen mobiler Endgeräte entsprechen.
Eine ausführliche Erläuterung der XML-Struktur und XML-Parsern wird mit der Seminararbeit von René-Pierre Vouri [Vou09] bereitgestellt.

Neben dem Zugriff auf das Dateisystems, um an das benötigte Kartenmaterial zu gelangen, gibt es noch eine weitere Möglichkeit. Wie in dem Konzept des Kapitels 3.3 dargestellt, sollen die Karten auch über das Internet bezogen werden. Dafür stellt die J2ME eine spezielle Klasse bereit, die `HttpConnection`. Mit dieser Klasse ist es möglich, eine Verbindung zu einer Internetseite aufzubauen und Daten von dieser Seite zu beziehen. Jedoch wird zurzeit noch kein angepasster Webclient für mobile Dienste durch die „geoGLIS oHG“ angeboten. Aus diesem Grund, wird die konkrete Umsetzung dieser Klasse nicht näher betrachtet.
Es sollte jedoch darüber nachgedacht werden, dies in naher Zukunft zu ändern, da die Internetnutzung durch mobile Endgeräte von Jahr zu Jahr wächst [New09]. Ein Beispiel für solch einen mobilen WMS - Client ist Google Maps Mobile [Mob09].

4.6.3 Geodaten einbinden: Waypoints, Tracking-Daten, POI's

Neben der Darstellung von Kartenmaterial und der GPS-Position, soll es mit der Anwendung möglich sein die restlichen Geodaten auf dem Display anzuzeigen. Dazu zählen die Waypoints, die Tracking-Daten und die POI's.
Das Darstellen der jeweiligen Daten, macht es erforderlich die Datei, in der sich die Daten befinden, zu Öffnen. Dies geschieht wie im Kapitel 4.6.1 beschrieben über den Zugriff auf das Dateisystems mit der `FileConnection` - API. Anschließend können die Daten aus der Datei geladen werden. Dabei ist zu beachten, dass die Daten in den Dateiformaten .gpx bzw. .kml vorliegen können. Auf diese beiden Formate wurde bereits in dem Kapitel 2.4.1 näher eingegangen. Wie aus dem Kapitel 2.4.1 hervorgeht, besitzen beide Formte die XML - Struktur. Damit die entsprechenden Daten aus dieser Datei gelesen werden können, wird auf einen XML - Parser [Hau09] zurückgegriffen, der dazu dient, die Datei Elementweise auszulesen (siehe Kapitel 4.6.2). Dadurch ist es möglich, die gewünschten Daten aus der Datei herauszufiltern und auf einer Variablen abzulegen. Das Herauslesen der einzelnen Elemente, erfolgt bei den Geodaten solange, bis keine Elemente mehr gefunden werden. Es soll im nachfolgenden jedoch ausschließlich das GPX - Dateiformat näher betrachtet werden.
Dabei sind die Struktur und der Aufbau der verschiedenen Geodaten zu beachten. In dem Listing 4.11 ist die Struktur einer Tracking - Datei auszugsweise dargestellt. Das Element auf das in diesen Zusammenhang ein Zugriff ausgeübt werden soll, wird durch `<trkpt lat="...."lon="....">... </trkpt>` beschrieben. Durch dieses Element wird ein Trackingpunkt definiert. Aus diesem Element sollen anschließend die Längen- und Breitengrade herausgefiltert werden. Neben diesen beiden Daten können auch noch die Höhe und der Zeitstempel aus der GPX - Datei gewonnen werden.
Der Aufbau der XML - Struktur ist für Waypoints und POI's der gleiche. Deshalb kann das Einlesen dieser Geodaten durch dasselbe MIDlet erfolgen. Die Grundstruktur für solche Punkte ist in dem Listing 4.12 dargestellt. Das Element wird in dieser Struktur durch `<wpt lat="...."lon="...."> ... </wpt>` definiert. Neben den Längen- und Breitengraden können diese Daten noch zusätzliche Informationen wie den Namen enthalten.

```
[...]
 <trk>
   <name>fürstenwerder radtour on GPSies.com</name>
   <link href="http://www.gpsies.com/map.do?fileId=slxpylfeeiewqdnk"/>
   <trkseg>
     <trkpt lat="53.38714730" lon="13.57737200">
       <ele>104.00000</ele>
       <time>2009-04-03T11:57:52Z</time>
     </trkpt>
     <trkpt lat="53.38599550" lon="13.57432500">
       <ele>105.00000</ele>
       <time>2009-04-03T11:59:18Z</time>
     </trkpt>
     [...]
   </trkseg>
 </trk>
</gpx>
```

Listing 4.11: GPX-Struktur von GPS-Tracks

```
[...]
 <link href="http://www.my-poi.info">
   <text>my-poi.info</text>
 </link></metadata>
 <wpt lat="54.51166" lon="9.54138">
  <ele>0</ele>
  <time>1970-01-01T01:00:00.000Z</time>
  <name>Schloss Gottorf</name>
  <sym>Waypoint</sym>
 </wpt>
 [...]
</gpx>
```

Listing 4.12: GPX-Struktur von POI's und Waypoints

Innerhalb solch einer GPX - Datei können wie aus den beiden Listings 4.11 und 4.12 ersichtlich mehrere der beschriebenen Elemente enthalten sein. Beim Auslesen dieser Elemente können daher große Datenmengen entstehen. Um einen Datenverlust beim Lesen der Daten vorzubeugen, ist es sinnvoll diese persistent zu speichern.
Dafür wird durch die J2ME das „Record Management System „(RMS) bereitgestellt. Mit Hilfe des RMS werden die Daten in einem `RecordStore` (Datenbank) abgelegt. Dafür steht dem Entwickler die gleichnamige Klasse `RecordStore` innerhalb des RMS zur Verfügung. Diese Klasse ermöglicht das Speichern der gewünschten Datensätze. Die Daten werden als Byte-Array abgelegt. Damit die Daten in die Datenbank abgelegt werden können, muss man diese zunächst anlegen. [Mos06]

In dieser Datenbank können beliebig viele Datensätze enthalten sein. Solch ein Datensatz ist durch eine eindeutige ID gekennzeichnet. Außerdem ist es möglich, innerhalb eines MIDlet's mehrere `RecordStores` anzulegen. Dadurch kann innerhalb einer Anwendung sowohl für Waypoints und POI's als auch für die Tracking-Daten solch ein `RecordStore` erzeugt werden.
Damit ein `RecordStore` angelegt werden kann, muss zunächst ein Objekt davon geöffnet werden.

```
RecordStore RS = RecordStore.openRecordStore("Name", true)
```

Diesem Objekt werden zwei Parameter übergeben. Dabei handelt es sich zum einen um den Namen (max. 32 Zeichen) unter dem der `RecordStore` abgelegt werden soll. Zum anderen wird ein zweiter Parameter übergeben, dieser sorgt dafür, ob eine Datenbank neu angelegt werden soll, falls diese noch nicht existiert. Anschließend können die jeweiligen Daten in den `RecordStore` geschrieben werden. Dies geschieht über einen `OutputStream` und der Methode `write()`, der die Daten übergeben werden. Nachdem die Datensätze in die Datenbank abgelegt wurden, muss diese wieder geschlossen werden.

```
RS.closeRecordStore()
```

Damit die Tracks, POI's oder Waypoints auf den Display dargestellt werden können, müssen die Daten anschließend wieder aus den `RecordStore` gelesen werden. Dies erfolgt mit Hilfe eines `InputStream` und der Methode `read()`.
Um die Geodaten auf dem Display des mobilen Endgerätes darstellen zu können, müssen die Längen- und Breitengrade der Punkte (Tracking - Punkte, Waypoint, POI) in Pixelkoordinaten umgewandelt werden. Dies geschieht auf die gleiche Art und Weise wie die Umrechnung der GPS-Position (siehe Kapitel 4.6.2). Daher sollte eine eigene Klasse definiert werden, die für das Umrechnen von Koordinaten in Bildschirmpixelpositionen zuständig ist. Dadurch wird eine Folgenutzung von bestimmten Programmteilen ermöglicht.
Damit die Geodaten auf dem Display dargestellt werden können, sollte bereits eine Karte geladen sein. Ist dies nicht der Fall, wird die benötigte Kartenkachel anhand der Tracking - Punkte bestimmt. Dazu ermittelt man die Bounding Box dieser Daten. Dies geschieht, in dem geprüft wird, welcher der Punkte (Tracking - Punkte, Waypoints, POI's) am weitesten links oben und welcher rechts unten liegt. Das Positionieren der Karte erfolgt auf dieselbe Art und Weise, wie in dem Kapitel 4.6.2 beschrieben. Für das Ermitteln der Position wird bei mehreren Koordinaten die Koordinate gewählt, die in der linken oberen Ecke der Kartenkachel liegt. Nachdem die benötigten Berechnungen durchgeführt wurden, können abschließend die Daten auf dem Display angezeigt werden.
Eine ausführliche Beschreibung der Funktionsweise des RMS wird in den Quellen [Esc03] und [Mos06] bereitgestellt.

4.7 Aufzeichnen von Daten

4.7.1 Speichern von Dateien

Mit der Anwendung soll der Nutzer auch die Möglichkeit erhalten eigene Daten aufzuzeichnen. Damit diese Daten dauerhaft gespeichert werden können, werden sie zunächst mit Hilfe des RMS in einer Datenbank abgelegt (siehe Kapitel 4.6.3). Mit dem RMS gespeicherte Daten können durch die Anwendung gelesen werden. Die RMS - Datei existiert solange auf dem mobilen Endgerät, bis die MIDlet - Suite von dem Gerät gelöscht wird. Aus diesem Grund müssen die Daten separat auf dem Dateisystem des Endgerätes abgelegt werden. Darüber hinaus sollen die Daten in dem GPX - bzw. KML - Dateiformat abgelegt werden. Dadurch wird gewährleistet, dass die aufgenommenen Daten durch andere Anwendungen genutzt werden können. Außerdem können so die Daten anschließend analysiert und ausgewertet werden.
Ein Beispiel für das Schreiben der Struktur in einen Datensatz wird in dem Listing 4.13 näher erläutert. Das Schreiben in einen `RecordStore` erfolgt wie bereits in dem vorherigen Kapitel 4.6.3 beschrieben mit einen `OutputStream` und der Methode `write()`. Anschließend können die in der Datenbank enthaltenen Daten ausgelesen und in eine Datei geschrieben werden. Beim Auslesen der Daten aus einem `RecordStore` ist zu beachten, dass die Grundstruktur einer GPX- bzw. einer KML - Datei eingehalten wird. Für das Speichern von Daten wird auf das Dateisystem des mobilen Endgerätes mit der `FileConnection` - API zugegriffen.

```
os.write( [...] );
[...]
os.write( ( waypoint.latToString( .... ) + waypoint.lonToString( .... )).getBytes
    () );
[...]
os.write( [...] );
```

Listing 4.13: GPX - Grundstruktur in `RecordStore` schreiben. Quelle: [Vlk09]

Die Überlegungen für das Speichern von aufgezeichneten Geodaten wurde durch einen Blick in die GNU-GPL Software „Vlk-GPS“ [Vlk09] bestätigt. Innerhalb dieser Software wurde das Setzen von eigenen Daten auf ähnliche Weise realisiert.

4.7.2 Geodaten aufzeichen: Waypoints, Tracking-Daten

Das Setzen von Waypoints und das Starten des Tracking setzen voraus, dass eine Verbindung zu einem GPS-Empfänger besteht. Dadurch können die aktuellen Positionsdaten des Standortes ermittelt werden. Dies wurde im Kapitel 4.5 näher erläutert.
Wird durch den Nutzer der Menüpunkt „Waypoint setzen“ aufgerufen, wird auf das MIDlet für die Positionsbestimmung zugegriffen und die GPS-Position an die aufrufende Methode übergeben. Die übermittelten Längen- und Breitengrade sowie die Höhe wird wie bereits erwähnt in einen `RecordStore` abgelegt. Anschließend kann durch den Nutzer eine Beschreibung des Waypoints eingegeben werden. Nutzereingaben lassen sich in J2ME mit einer `TextBox` realisieren. Für solch eine `TextBox` kann festgelegt werden, welche Eingabebeschränkungen bestehen, d.h. welche Zeichen durch den Nutzer eingeben werden dürfen und wie lang ein Zeichensatz maximal sein darf. Auch die Beschreibung wird in der Datenbank hinterlegt. Nach der Eingabe einer Beschreibung wird der erfasste Waypoint aus dem `RecordStore` gelesen und auf dem Dateisystem des mobilen Endgerätes gespeichert.
Beim Aufzeichnen von Tracks, werden die Positionsdaten von dem GPS-Empfänger bezogen und wie die Waypoint - Daten in einen `RecordStore` abgelegt. Dieser Vorgang wird solange wiederholt bis das Tracking durch den Nutzer beendet wird. Abschließend werden die gewonnen Positionsinformationen in einer GPX - oder KML - Datei auf dem Dateisystem des mobilen Endgerätes gespeichert.

4.8 Übersicht der Umgesetzen Funktionen

In den vorherigen Kapiteln wurde beschrieben, welche Möglichkeiten die Java Micro Edition bietet, die verschiedenen Funktionen auf Basis des Konzeptes zu realisieren. Die Abbildung 47 zeigt, welche der gewünschten Funktionen mit der Software umgesetzt werden konnten. Bei der Umsetzung konnte der Zugriff auf einen integrierten GPS-Empfänger eines mobilen Endgerätes (Sony Ericsson W760i) mit Hilfe der Location - API hergestellt werden. Dabei werden die Positionsdaten zunächst auf dem Display des Gerätes ausgegeben. Dabei konnte realisiert werden, dass die Verbindung zu dem Empfänger bestehen bleibt, wenn in eine andere Funktion gewechselt wird.

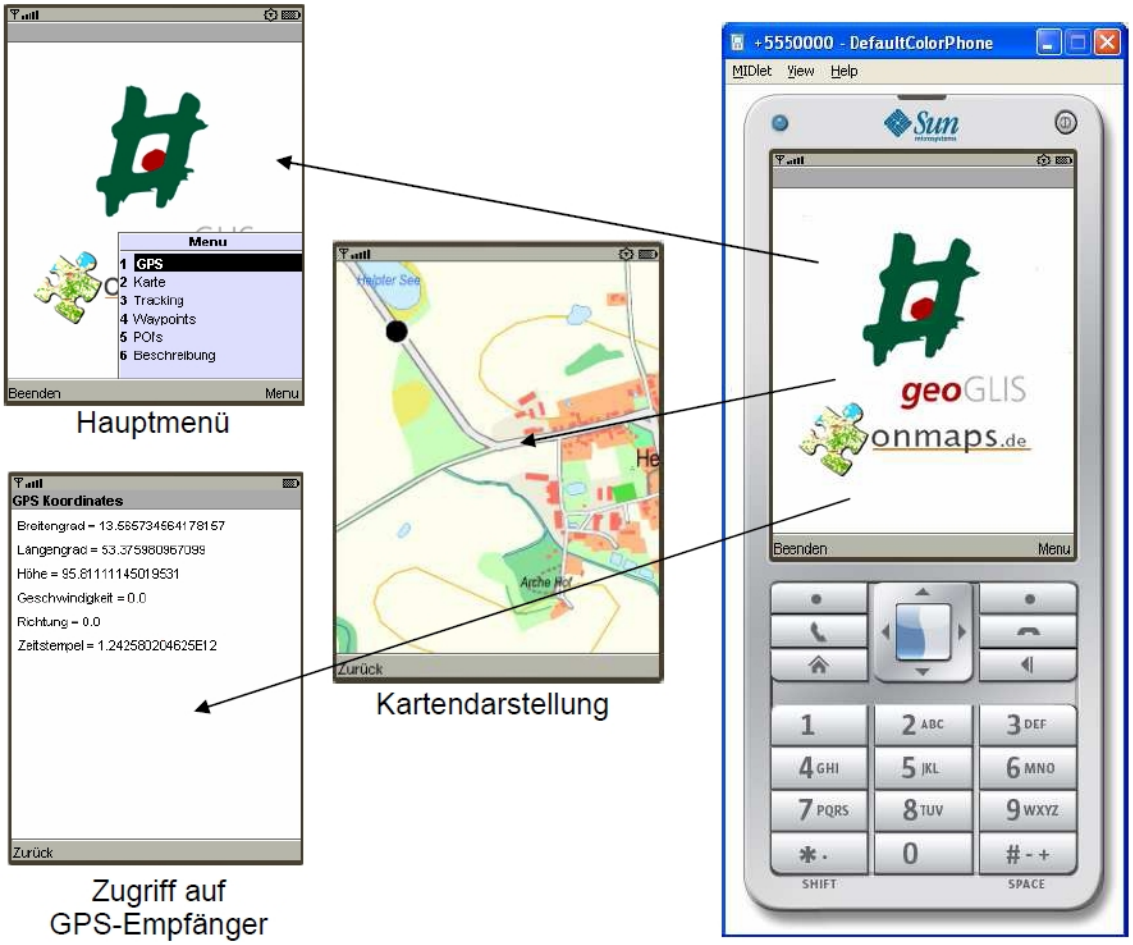


Abbildung 47: Übersicht der Umgesetzen Funktionen

Des Weitern konnte der Zugriff auf das Dateisystems des mobilen Endgerätes umgesetzt werden. Dadurch ist es möglich eine Karte aus einen Vorgegebenen Pfad auf dem Display des mobilen Endgeräts darzustellen. Der Zugriff auf das Dateisystems wird wie bereits erwähnt mit der FileConnection-API ermöglicht. Die in dem Kapitel 4.6.2 beschriebenen Berechnungen konnten bis zum jetzigen Zeitpunkt noch nicht in die Anwendung eingebunden werden. Anhand einer selbst gewählten Kartenkachel und GPS - Positionen wurden die Berechnungen nachvollzogen. Anhand der berechneten Werte wurden die Kartenposition und die GPS-Position in der J2ME-Applikation manuell gesetzt. Dadurch konnte

festgestellt werden, das die Überlegungen zu den Berechnungen der einzelnen Werte korrekt funktionieren.
Es konnten auf Grundlage des entwickelten Konzeptes nur Teile der gewünschten Funktionen umgesetzt werden. Aber durch die Arbeit konnte die Basis für die Softwareentwicklung einer mobilen Anwendung geschaffen werden. Darüber hinaus wurde in Zusammenhang mit dieser Arbeit ein stabil laufendes Grundgerüst einer J2ME-Applikation entwickelt.

4.9 Installation auf dem mobilen Endgerät

Bevor die Software auf dem Endgerät installiert werden kann, muss die MIDlet - Suite in einem .jar-File verpackt werden. Bei der .jar-Datei handelt es sich in der Entwicklung von Java-Applikationen um die ausführbare Datei. Durch diese Datei ist die Installation auf dem entsprechenden Gerät möglich.
In einem JAR - File sind unter anderem das MIDlet und die benötigte Ressourcen (wie Bilder, Sound und Hilfe-Dateien) enthalten. Bei dem Erstellen eines JAR - Files wird außerdem eine JAD - Datei erzeugt. In dieser Datei werden alle wichtigen Eigenschaften eines MIDlet's abgelegt. Diese Datei wird auch als Applikationsdeskripter bezeichnet. In der Datei sind unter anderem Informationen dazu enthalten, welche Konfiguration (CLDC 1.1 oder CDC) und welches Profil (MIDP, Fundation Profil) verwendet wurde. Darüber hinaus gibt die Datei Auskunft welche Größe das .jar-File besitzt und um welche Version es sich handelt. Für das Generieren dieser Dateien (JAR und JAD) steht in der Entwicklungsumgebung Eclipse eine Funktion zur Verfügung („Create Package“).
Zum Installieren der Datei auf dem mobilen Endgerät, muss diese zuvor auf das Gerät übertragen werden. Dazu stehen dem Nutzer zwei unterschiedliche Wege zur Verfügung. Zum einen existiert eine „Over The Air“-Schnittstelle (OTA). Bei dieser Methode bezieht der Nutzer die J2ME-Applikation direkt mit seinem mobilen Endgerät aus dem Internet. Für die OTA Variante ist es nötig, dass das MIDlet auf einem Web-Server zur Verfügung zu stellen. In Abbildung 48 wird die Übertragung des MIDlet's mittels OTA - Schnittstelle und das nachträgliche Installieren dargestellt. [Esc03], [Mos06]

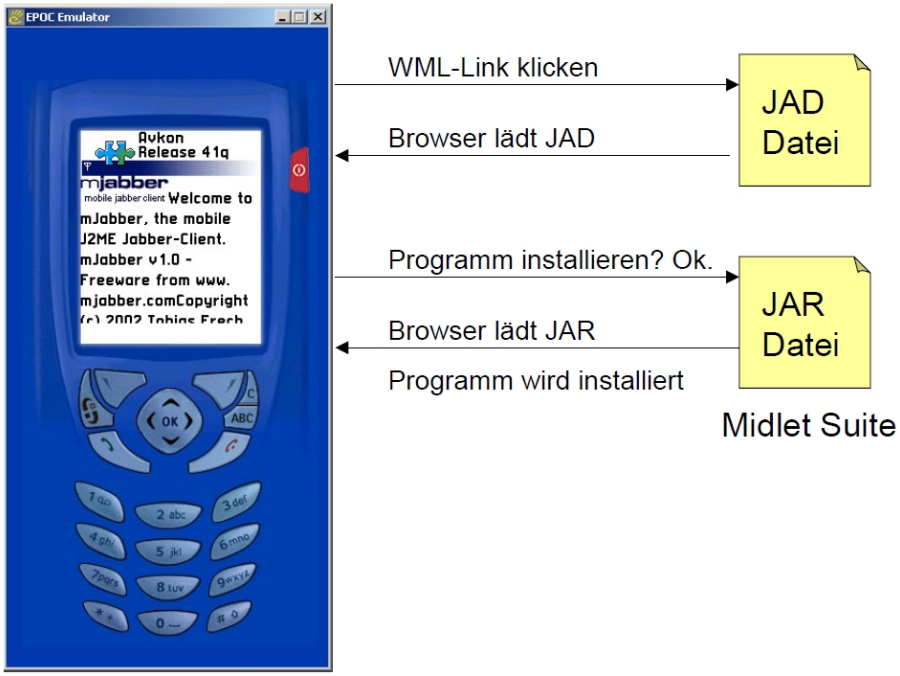


Abbildung 48: Übertraung der Dateien über das Internet (OTA)
Quelle: [Tot09]

Zum anderen besteht für den Nutzer die Möglichkeit, über eine PC-Schnittstelle, die J2ME-Anwendung auf das mobile Endgerät zu übertragen. Hierfür stehen dem Anwender mehrere PC-Schnittstellen zur Verfügung. Eine Schnittstelle kann über das Datenkabel (i.d.R. USB-Kabel) gebildet werden. Darüber hinaus können dem Anwender noch weitere Schnittstellen zur Verfügung stehen, eine Infrarot- oder eine Bluetooth - Schnittstelle.
Nach dem erfolgreichen Übertragen des MIDlet's auf das mobile Endgerät, kann es durch den Nutzer installiert und anschließend ausgeführt werden.

4.10 Anwendung auf dem Endgerät

4.10.1 Testgeräte

Bevor die J2ME-Applikation, wie im Kapitel 4.9, auf dem mobilen Endgeräte übertragen und installiert wurde, ist diese zunächst auf den Emulator des WTK's (siehe Kapitel 4.2) ausgeführt und getestet worden. Dadurch war es möglich, auftretende Fehler im Vorfeld zu beheben. So konnten unnötige Installationen auf dem mobilen Endgerät vermieden werden. Aus diesem Grund wird der Emulator in die Liste der Geräte der Tabelle 8 mit aufgenommen. Nachdem die Software auf den Emulator fehlerfrei lief, wurde diese auf das mobile Endgerät übermittelt. Die Tabelle 8 enthält eine Übersicht, über die Geräte auf denen die Software installiert und getestet wurde.

	Sony Ericsson W760i	WTK Emulator DefaultColorPhone
Akkulaufzeit (Standby)	bis zu 400 Std	unbegrenzt
Displayauflösung	240 x 320 Px	240 x 320
Farbdisplay	ja	ja
Farben	262.144	4096
GPS	ja	ja (Simulation)
Bluetooth	ja	ja (Simulation)
USB-Unterstützung	ja	nicht nötig
Java-fähig	ja	ja
	CLDC 1.1	CLDC 1.1
	MIDP 2.0	MIDP 2.1
Betriebsystem	Eigententwicklung	

Tabelle 8: Testgeräte
Quellen: [hm09b], [hm09a]

In dem nachfolgenden Kapitel soll zunächst die Funktionsweise der Applikation auf den Emulator und anschließend auf dem realen Gerät beschrieben werden. Bei dem Emulator ist zu beachten, dass dieser alle wichtigen Funktionen der gängigen Betriebsysteme zusammenfasst. Aufgrund dessen kann es zu Unterschieden beim Ausführen der programmierten J2ME-Anwendung kommen.

4.10.2 Testen der Software

Emulator

Nachdem die Software auf dem Emulator gestartet wurde, wird diese ordnungsgemäß ausgeführt. Das Hintergrundbild sowie das Menü werden dem Anwender wie gewünscht eingeblendet. Der Aufruf der einzelnen Menüpunkte kann problemlos ausgeführt werden. Nach der Wahl des Menüpunktes „GPS“ wird eine Auswahlliste geöffnet, aus der gewählt werden kann, auf welche Art eine Verbindung zu einem GPS-Empfänger hergestellt werden soll. Wie bereits in der Umsetzung (siehe Kapitel 4.5) erwähnt, wurde die Bluetooth - Schnittstelle nicht realisiert.
In einen Emulator lassen sich „External Events“ simulieren. Dazu zählt auch die Simulation eines internen GPS - Gerätes, wie in der Abbildung 49 dargestellt. Damit diese Funktion getestet werden kann, ist es erforderlich eine Textdatei mit verschieden Punktinformationen anzulegen.

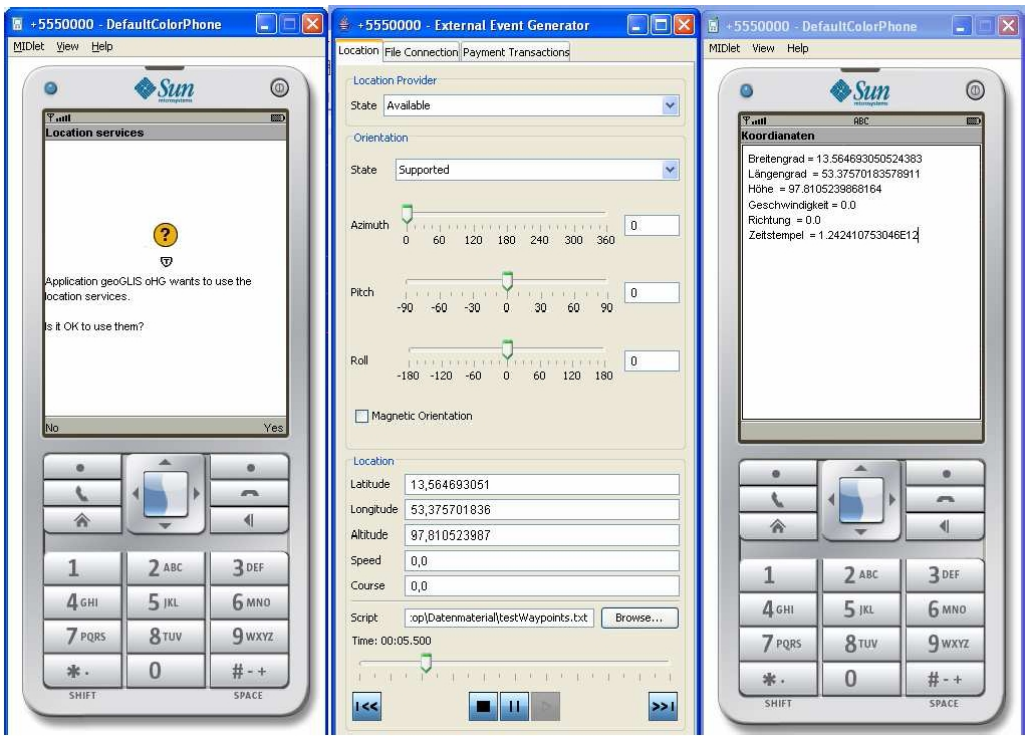


Abbildung 49: Simulation eines internen GPS-Empfängers

Die in dieser Datei enthaltenen Koordinaten werden korrekt durch die Applikation empfangen und auf dem Display ausgegeben. Anschließend kann der Nutzer in das Hauptmenü des Programms zurückkehren. Die Positionsbestimmung läuft im Hintergrund weiter. Dies lässt sich durch wiederholtes Aufrufen des Menüpunktes feststellen. Mit dieser Funktion konnte gezeigt werden, dass die Verbindung zu einem integrierten GPS-Empfänger möglich ist.
Ein weiterer Menüpunkt „Karte“ öffnet dem Nutzer der Anwendung eine Karte. Das Laden der Karte erfolgt momentan noch nicht dynamisch. Auch die dargestellte Position wurde manuell durch die im Kapitel 4.6.2 beschriebenen Berechnungen ermittelt. Zum Prüfen,

ob die ermittelte Pixelkoordinate der GPS-Position und die Ausrichtung der Karte korrekt ist, wurde dies mit dem Deutschland Viewer der „geoGlIS oHG“ [go09b] verglichen. Außerdem ist daraus ersichtlich, dass der Zugriff auf das Dateisystems innerhalb des Emulator fehlerfrei läuft, da die Karte auf dem Display gezeichnet wird. Innerhalb dieser Funktion ist es dem Nutzer erlaubt, jederzeit in das Hauptmenü zuwechseln.
Neben diesen beiden Hauptfunktionen hat der Anwender noch die Möglichkeit sich eine Kurzbeschreibung der Software auf dem Display anzeigen zu lassen. Anschließend kann er auch hier zum Startbildschirm zurückkehren, um einen Zugriff auf das Hauptmenü zu erhalten.
Sollte eine Funktion innerhalb der J2ME-Applilation noch nicht umgesetzt worden sein, wird der Nutzer mit einem Informationsbildschirm darüber in Kenntnis gesetzt.
Für das ordnungsgemäße Beenden der Anwendung wurde in das Hauptprogramm der Menüpunkt „Beenden“ integriert. Das Programm kann innerhalb des Emulator ohne Probleme geschlossen werden.

Testgerät

Mit Hilfe einer Bluetooth - Schnittstelle, konnten die zur Installation auf dem mobilen Endgerät benötigten Dateien (JAR, JAD) auf das Gerät übertragen werden. Nachdem die Dateien übertragen wurden, war es möglich die Installation zu starten. Nach einem erfolgreichen Abschluss der Installation konnte die Anwendung ausgeführt werden.
Auf dem Testgerät Sony Ericson können alle Funktionen, die durch die Applikation bereitgestellt werden. ohne weiteres genutzt werden. Dabei konnte unter anderem eine Verbindung zu den internen GPS-Empfänger des Sony Ericsson W760i hergestellt werden. Anschließend wurden die Empfangenen Positionsdaten auf dem Display angezeigt.
Nach dem Test des Zugriff auf ein internes GPS-Gerät wurde der Menüpunkt „Karte“ näher betrachtet. Wie bei den Emulator konnte eine Karte sowie die GPS-Position durch den Zugriff auf das Dateisystem geladen und auf dem Display gezeichnet werden. Die Berechnung der Pixelkoordinaten erfolgte auch hier im Vorfeld manuell.

Die J2ME-Applikation lief sowohl auf den Emulator als auch auf dem realen Testgerät ohne das Auftreten von Fehlern.

5 Zusammenfassung und Ausblick

Das Ziel der Arbeit, ein Konzept für die Kartendarstellung auf einem mobilen Endgerät zu entwerfen, wurde erfolgreich umgesetzt. Darüber hinaus konnte in Zusammenhang mit dieser Arbeit ein Grundgerüst einer J2ME-Applikation geschaffen werden. Mit Hilfe des Grundgerüsts ist es möglich die gewünschten Funktionen nach und nach zu integrieren. Auf Basis des entworfenen Konzeptes konnten bereits die Ansätze für die Umsetzung mit der Java Mirco Edition erläutert werden. Außerdem konnten einige Teile (internes GPS, Kartendarstellung) des entwickelten Konzeptes realisiert werden.
Für die Umsetzung der mobilen Anwendung wurde sich mit der Programmiersprache J2ME und der damit zusammenhängenden objektorientieren Programmierung auseinander gesetzt. Dabei konnte ein Überblick gewonnen werden, welche Möglichkeiten auf dem Gebiet der Entwicklung mobiler Anwendungen existieren.
Der Markt für mobile Anwendungen ist sehr schnelllebig und wächst dabei immer weiter. Im Laufe der Arbeit sind unzählige weitere mobile Anwendungen auf den Markt erschienen. Auch die Nutzung des mobilen Internets hat in dieser Zeit stark zugenommen [New09]. Die Möglichkeiten mobiler Anwendungen und Dienste sind noch lange nicht ausgeschöpft. Ein Gebiet, der im Bereich mobiler Endgeräte immer mehr zunimmt ist die zusätzliche Nutzung als Navigationsgerät [Lüd09]. Die Navigationssoftware für Mobiltelefone und Smartphones reichen jedoch zurzeit noch nicht an die „Stand-Alone“ - Geräte herran. Der Markt ist für diesen Bereich geöffnet und wird daher in Zukunft noch stärker wachsen. Ein weiterer Grund für dieses Wachstum ist, dass immer mehr mobile Endgeräte mit einen GPS-Gerät ausgestattet werden. Man darf also gespannt sein, was die Zukunft auf diesem Gebiet noch mit sich bringt.
In der Entwicklung mobiler Anwendung sollte die Nutzung des Internets sowie das berücksichtigen von Positionsdaten (GPS-Koordinaten) weiter verfolgt werden.
Abschließend lässt sich sagen, dass es sich bei der gestellten Aufgabe um ein sehr spannendes Thema handelt. Da das Thema sehr umfangreich ist, konnte im Rahmen dieser Arbeit einiges nur Angeschnitten werden. Es konnten jedoch die verschieden Möglichkeiten, die die Entwicklung mobiler Anwendungen bietet, aufgezeigt werden. Die Möglichkeiten mobiler Anwendungen sind bei weitem noch nicht ausgeschöpft, daher ist die Entwicklung mobiler Anwendung zukunftssicher. Außerdem ist der Markt zurzeit noch am wachsen und entwickelt sich ständig weiter.

Literaturverzeichnis

[AB09] Sony Ericsson Mobile Communications AB. *BlueGPS - a Bluetooth GPS Sample Application*. [http://developer.sonyericsson.com/util/SearchCMS.do?criteria=GPS&search=go], Letzer Zugriff: 08.05.2009.

[akt09] Hardware aktuell. *Hardware - Eingebettetes System*. [http://www.hardware-aktuell.com/lexikon/Eingebettetes_System/], Letzer Zugriff: 23.02.2009.

[awo09] awokenmind.de. *GoogleMaps 2 TrekBuddy*. [http://gm2tb.awokenmind.de//], Letzer Zugriff: 18.02.2009.

[Bau07] Patrick Grässle, Henriette Baumann, Philippe Baumann. *UML 2 - projektorientiert*. 4. aktualisierte Auflage. Galileo Computing, 2007.

[Bauge] Manfred Bauer. ***Vermessung und Ortung mit Satelliten*** - *GPS und andere satellitengestützte Navigationssysteme*. Wichmann, 2002, 5. Auflage.

[BfU09] Wald und Landschaft BUWAL und vom Bundesamt für Metrologie und Akkreditierung METAS Bundesamt für Umwelt. *Nichtionisierende* ***Strahlung Mobilfunk-Basisstationen (GSM)*** *Messempfehlung*. [http://www.umwelt.tg.ch/documents/Mobilfunk-Basisstationen%20(GSM).pdf], Letzer Zugriff: 23.03.2009.

[BIT09] © Statista 2009 Quelle: BITKOM. *Anzahl der Mobilfunkanschlüsse in Deutschland in Millionen*. [http://de.statista.com/statistik/daten/studie/3907/umfrage/anzahl-der-mobilfunkanschluesse-in-deutschland/#info], Letzer Zugriff: 25.03.2009.

[Con09a] World Wide Web Consortium. *Leading the Web to Its Full Potential...* [http://www.w3.org/], Letzer Zugriff: 02.04.2009.

[Con09b] POI Converter. *POI Converter*. [http://rjdavies.users.btopenworld.com/html/poiconverter.html], Letzer Zugriff: 03.04.2009.

[Deu09] Google Deutschland. *Google Maps Deutschland*. [http://maps.google.de/maps?hl=de&tab=wl], Letzer Zugriff: 11.03.2009.

[Dev09] Silent Development. *LocateMe - Free J2ME GPS Tracking Software*. [http://silentdevelopment.blogspot.com/2008/03/locateme-free-j2me-gps-tracking.html], Letzer Zugriff: 19.01.2009.

[Döl08] Mirko Dölle. Elektronische Schatzkarte. *c't 11/2008*, 2008.

[Dud03] *Schülerduden - Informatik*. Duden, 2003.

[dVdLdBD09] AdV-Arbeitsgemeinschaft der Vermessungsverwaltungen der Länder der Bundesrepublik Deutschland. *ATKIS-Amtliches Topographisch-Kartographisches Informationssystem.* [http://www.atkis.de/], Letzer Zugriff: 12.03.2009.

[Ear09] Google Earth. *Was ist KML?* [http://earth.google.de/kml/whatiskml.html], Letzer Zugriff: 02.04.2009.

[Ehr09a] Dr. Wilfried Ehrensperger. *GPSwithMAPS - Kartendownload.* [http://gpswithmaps.de/map1.html], Letzer Zugriff: 18.02.2009.

[Ehr09b] Dr. Wilfried Ehrensperger. *GPSwithMAPS.* [http://gpswithmaps.de/], Letzer Zugriff: 26.01.2009.

[Esc03] Sebastian Eschweiler. *Das Einsteigerseminar - Java 2 Micro Edition.* bhv, 2003.

[fE09a] Wikipedia Die freie Enzyklopädie. *Mobiltelefon.* [http://de.wikipedia.org/wiki/Handy], Letzer Zugriff: 01.02.2009.

[fE09b] Wikipedia Die freie Enzyklopädie. *GPS Exchange Format.* [http://de.wikipedia.org/wiki/GPX], Letzer Zugriff: 01.04.2009.

[fE09c] Wikipedia Die freie Enzyklopädie. *GNU General Public License.* [http://de.wikipedia.org/wiki/GNU_General_Public_License#cite_note-1], Letzer Zugriff: 23.05.2009.

[fE09d] Wikipedia Die freie Enzyklopädie. *GPS-Maus.* [http://de.wikipedia.org/wiki/GPS-Maus], Letzer Zugriff: 23.05.2009.

[fKuG09] BGK Bundesamt für Kartographie und Geodäsie. *Bundesamt für Kartographie und Geodäsie.* [http://www.bkg.bund.de/], Letzer Zugriff: 12.03.2009.

[For09] TrekBuddy Forum. *TrekBuddy - J2ME application for GPS navigation and tracking.* [www.trekbuddy.net], Letzer Zugriff: 26.01.2009.

[Fos09a] Dan Foster. *GPX 1.1 Schema Documentation.* [http://www.topografix.com/GPX/1/1/], Letzer Zugriff: 01.04.2009.

[Fos09b] Dan Foster. *GPX: the GPS Exchange Format.* [http://www.topografix.com/gpx.asp], Letzer Zugriff: 01.04.2009.

[Fou08a] Eclipse Foundation. *Eclipse IDE for Java Developers (Eclipse Version: Eclipse Europa 3.2).* [http://www.eclipse.org/downloads/], Letzer Zugriff: 24.06.2008.

[Fou08b] Eclipse Foundation. *Installing EclipseME.* [http://eclipseme.org/docs/installation.html], Letzer Zugriff: 24.06.2008.

[Fre09] Thomas Frenken. *JSR-179: Location Based Mobile Applications auf der JavaME-Plattform.* [http://www.mi.fh-wiesbaden.de/~barth/mobile/ws0607/Location.pdf], Letzer Zugriff: 06.05.2009.

[Fri94] Bill, Fritsch. *Grundlagen der Geo-Informationssysteme - Band 1 Hardware, Software und Daten - 2. Auflage.* Wichman, 1994.

[fSidI09] Bundesamt für Sicherheit in der Informationstechnik. *Mobile Endgeräte und mobile Applikationen.* [http://www.bsi.de/literat/doc/mobile/mobile_endgeraete.pdf], Letzer Zugriff: 27.01.2009.

[Geo09a] a Groundspeak Project Geocaching. *Geocaching - The Official Global GPS Cache Hunt Site.* [http://www.geocaching.com/], Letzer Zugriff: 25.05.2009.

[geo09b] geocaching.de. *Geocaching.de - der deutschen Geocaching-Seiten.* [http://www.geocaching.de/], Letzer Zugriff: 23.05.2009.

[Ger09] Peter Gerwinski. *GNU General Public License.* [http://www.gnu.de//documents/gpl.de.html], Letzer Zugriff: 17.02.2009.

[gf09] gps freeware.de. *GPS-Track-Analyse.NET - Download Installations-Archiv.* [http://www.gps-freeware.de/DownloadStart.aspx], Letzer Zugriff: 26.05.2009.

[Gmb09a] DATACOM Buchverlag GmbH. *IT Wissen - Das große Online-Lexikon für Informationslexikon - Handy.* [http://www.itwissen.info/definition/lexikon/Handy-cellphone.html], Letzer Zugriff: 22.02.2009.

[Gmb09b] DATACOM Buchverlag GmbH. *IT Wissen - Das große Online-Lexikon für Informationslexikon - **Ortung**.* [http://www.itwissen.info/definition/lexikon/Ortung-locating.html], Letzer Zugriff: 22.03.2009.

[Gmb09c] DATACOM Buchverlag GmbH. *IT Wissen - Das große Online-Lexikon für Informationslexikon - **NMEA (national marine electronics association)**.* [http://www.itwissen.info/definition/lexikon/NMEA-national-marine-electronics-association.html], Letzer Zugriff: 23.03.2009.

[Gmb09d] DATACOM Buchverlag GmbH. *IT Wissen - GSM-Netz.* [http://www.itwissen.info/definition/lexikon/GSM-Netz-GSM-network.html], Letzer Zugriff: 23.03.2009.

[Gmb09e] DATACOM Buchverlag GmbH. *IT Wissen - Das große Online-Lexikon für Informationslexikon - Windows CE.* [http://www.itwissen.info/definition/lexikon/Windows-CE.html], Letzer Zugriff: 24.02.2009.

[Gmb09f] DATACOM Buchverlag GmbH. *IT Wissen - Das große Online-Lexikon für Informationslexikon - BlackBerry.* [http://www.itwissen.info/definition/lexikon/BlackBerry-BlackBerry.html], Letzer Zugriff: 27.05.2009.

[Gmb09g] DATACOM Buchverlag GmbH. *IT Wissen - Das große Online-Lexikon für Informationslexikon - Smartphone.* [http://www.itwissen.info/definition/lexikon/Smartphone-smart-phone.html], Letzer Zugriff: 27.05.2009.

[go09a] geoGLIS oHG. *Herzlich Willkommen bei der geoGLIS, Ihrem Dienstleister für Geo-Informationssysteme!* [http://www.onmaps.de], Letzer Zugriff: 17.02.2009.

[go09b] geoGLIS oHG. *Mit onmaps haben Sie gute Karten...* [http://www.onmaps.de], Letzer Zugriff: 17.02.2009.

[GPS09] Bechtold Internet Solutions GPSies.com. *GPSies - Tracks for Vagabonds.* [http://www.GPSies.com/], Letzer Zugriff: 15.03.2009.

[Gun09] Data2Map Dipl.-Ing. Manfred Guntz. *PNG.* [http://www.data2map.de/infothek/dateiformate/png/index.html], Letzer Zugriff: 02.04.2009.

[Hau09] Stefan Haustein. *kXML 2 (XML-Parser).* [http://kxml.sourceforge.net/], Letzer Zugriff: 13.05.2009.

[hDAMS09] heise Developer Autor: Markus Stäuble. *Googles mobile Plattform Android.* [http://www.heise.de/developer/artikel/print/120124], Letzer Zugriff: 26.02.2009.

[hm09a] handy mc.de. *Datenblatt Nokia 6230i.* [http://www.handy-mc.de/datenblatt/handy/nokia-6230i.html], Letzer Zugriff: 06.05.2009.

[hm09b] handys mobile.de. *Sony-Ericsson W760i - Technische Daten.* [http://www.handys-mobile.de/sony-ericsson-w760i-datenblatt.html], Letzer Zugriff: 06.05.2009.

[Hog09] Softtonic Autor: Shawn Hogan. *MIDP for Palm OS 1.0.* [http://midp-for-palm-os.softonic.de/palm], Letzer Zugriff: 25.02.2009.

[IJ09] W3C.DE/AT Ian Jacobs, Head of W3C Communications; Deutsche Übersetzung: Thomas Tikwinski. *Über das World Wide Web Consortium (W3C).* [http://www.w3c.de/about/overview.html], Letzer Zugriff: 02.04.2009.

[Inc08a] Sun Microsystems, Inc. *Java SE Downloads Version 1.5.0.* [http://java.sun.com/javase/downloads/index.jsp], Letzer Zugriff: 21.08.2008.

[Inc08b] Sun Microsystems Inc. *Sun Java Wireless Toolkit for CLDC, 2.5.2.* [http://java.sun.com/products/sjwtoolkit/download.html], Letzer Zugriff: 22.08.2008.

[Inc09] Eray Ince. *GPS in J2ME Starter Example.* [http://blog.erayince.com/?tag=gps], Letzer Zugriff: 15.05.2009.

[Kar09] Helmut Karger. *Wandern mit GPS: GPS Track Viewer - Anzeige von GPS Tracks, Routen und Waypoints im Kartenfenster.* [http://www.gpswandern.de/gpxviewer/gpxviewer.shtml], Letzer Zugriff: 17.02.2009.

[Kni 6] Elfriede T. Knickmeyer. ***Einführung in die Navigation*** *für GeoinformatikerInnen und VermesserInnen.* Schriftemreihe der Fachhochschule Neubrandenburg, März 2003, Reihe B: Band 6.

[Lüd09] Daniel Lüders. Alles an bord! *c't 10/2009*, 2009.

[Lib09] Nevins Memorial Library. *Internet Access & Computers at the Library.* [http://www.nevinslibrary.org/reference/internetpolicy.html], Letzer Zugriff: 08.04.2009.

[Mah09] Sun Developer Network Autor: Qusay Mahmoud. *MIDP for Palm OS 1.0: Developing Java Applications for Palm OS Devices.* [http://developers.sun.com/mobility/midp/articles/palm/index.html], Letzer Zugriff: 25.02.2009.

[Mar09] Cyril Marti. *Embedded-Betriebssysteme.* [http://www.iam.unibe.ch/~rvs/teaching/SS02/bs/arbeiten/Cyril_Marti-Embedded_OS.pdf], Letzer Zugriff: 23.02.2009.

[Müc09] Gerald Mücke. *Möglichkeiten und Grenzen der Oberflächengestaltung auf mobilen Endgeräten.* [http://wwwstud.rz.uni-leipzig.de/~mai00cjs/data/OberflaechengestaltungMobilerEndgeraete.pdf], Letzer Zugriff: 01.05.2009.

[Mob09] Google Mobile. *Probieren Sie Google Maps mithilfe der interaktiven Demo aus.* [http://www.google.de/mobile/gmm/demo.html], Letzer Zugriff: 06.04.2009.

[Mos06] Ulrich Breymannn, Heiko Moseman. *JavaME - Anwendungsentwicklung für Handy's, PDA und co.* 2.Auflage. Hanser, 2006.

[na09] news.at (apa). *Marktstudie: 2008 weltweit zwei Milliarden Handynutzer.* [http://www.news.at/articles/0407/548/74754/marktstudie-2008-milliarden-handynutzer], Letzer Zugriff: 25.03.2009.

[Nav09] NaviFriends.com. *POI Datenbank.* [http://www.navifriends.com/nfpois/home.php], Letzer Zugriff: 03.04.2009.

[New09] Mobilfunk News. *Mobile Internet-Nutzung nimmt zu.* [http://www.handy-mc.de/mobilfunk-news/artikel/2008/09/17/mobile-internet-nutzung-nimmt-zu.html], Letzer Zugriff: 06.04.2009.

[Ope09a] OpenStreetMap. *OpenStreetMap - Die freie Wiki-Weltkarte.* [http://www.openstreetmap.de/], Letzer Zugriff: 18.02.2009.

[Ope09b] OpenStreetMap. *OpenStreetMap - The Free Wiki World Map.* [http://www.openstreetmap.org/], Letzer Zugriff: 18.02.2009.

[PC09] Club Pocket PC. *Windows CE - Das Betriebssystem.* [http://www.pocketpc-users.de/ceinfo.htm], Letzer Zugriff: 25.02.2009.

[Pet] Günter Petrahn. *Taschenbuch Vermessung - Grundlagen der Vermessungstechnik.* Cornelsen Verlag, Berlin.

[Pet09] Dana Peters. *GPS Track 1.1 for J2ME.* [http://www.qcontinuum.org/gpstrack], Letzer Zugriff: 19.01.2009.

[Pla09] Dr. Frank Plagge. *Software in eingebetteten Systemen.* [http://www.uni-kassel.de/fb16/fsg/dateien/ses/ss2005/02%20-%20Eingebettete%20Systeme.pdf], Letzer Zugriff: 23.02.2009.

[RD08] Peter Röbke-Doerr. Pfadfinder - Siebzehn Programme für die Off-Road-Navigation. *c't 25/2008*, 2008.

[R.S09] R.Stahl. *Basisdaten der Vermessungsverwaltungen.* [http://www.giub.uni-bonn.de/gistutor/theorie/daten/basisdat/basisdat.htm], Letzer Zugriff: 12.03.2009.

[rw09a] radreise wiki.de. *Rasterkarten.* [http://radreise-wiki.de/Rasterkarten], Letzer Zugriff: 08.03.2009.

[rw09b] radreise wiki.de. *Vektorkarten.* [http://radreise-wiki.de/Vektorkarten], Letzer Zugriff: 08.03.2009.

[SS09] Copyright 2003 08 Demo Source and Support. *Java Tutorial - J2ME.* [http://www.java2s.com/Tutorial/Java/0430__J2ME/Catalog0430__J2ME.htm], Letzer Zugriff: 26.05.2009.

[Sta07] Alexander Anders, Torsten Starke. *Diplomarbeit - Entwicklung eines WMS-Clients für ein mobiles Endgerät.* HTW Dresden, 2007.

[Ste01] Ralph Steyer. *Java 2 Das Programmier-Handbuch.* Markt+Technik, 2001.

[Stu08] M. Frei, R. Wittwer, B. Studer. *J2ME-Tutorial (Java 2 Micro Edition).* [http://www.bstuder.ch/sppmc/v20_Tutorial_J2ME_ohne_loe.pdf], Letzer Zugriff: 03.07.2008.

[Sun09a] Sun. *Getting Started with FileConnection APIs.* [http://developers.sun.com/mobility/apis/articles/fileconnection/], Letzer Zugriff: 08.05.2009.

[Sun09b] Sun. *JSR 82 - Bluetooth API 1.1.* [http://java.sun.com/javame/technology/msa/jsr82.jsp], Letzer Zugriff: 08.05.2009.

[Sxh09] Dr. Frank Sxholles. *ATKIS®-Topografische und kartografische Informationen.* [http://www.laum.uni-hannover.de/ilr/lehre/Isv/Isv_Verm.htm#atkis], Letzer Zugriff: 12.03.2009.

[Töd09] Kai Tödter. *J2ME MIDlet-Entwicklung mit Eclipse, Ant und Antenna.* [http://www.toedter.com/download/J2ME-Eclipse-Antenna.pdf], Letzer Zugriff: 10.04.2009.

[Tot09] *Mobile Endgeräte.* [http://www.linecity.de/downloads/kap_4.pdf], Letzer Zugriff: 03.04.2009.

[Tra09] TrackMyJounrey. *TrackMyJounrey.* [http://www.trackmyjourney.co.uk], Letzer Zugriff: 19.01.2009.

[ube09] ubergizmo. *Holux GPSlim 240.* [http://www.ubergizmo.com/de/2006/10/general/holux_gpslim_240.php], Letzer Zugriff: 27.03.2009.

[uDMW09] Dr. Anja Köhne und Dr. Michael Wößner. *GPS-Infos - NMEA-0183 Daten.* [http://www.kowoma.de/gps/zusatzerklaerungen/NMEA.htm], Letzer Zugriff: 23.03.2009.

[Use08] Claus Alexander Usener. *Grundlagen der MIDlet-Programmierung.* [http://www.wi.uni-muenster.de/pi/lehre/ws0607/skiseminar/ausarbeitungen/04-Programmierung.pdf], Letzer Zugriff: 25.08.2008.

[uSO09] Dominik Gruntz und Severin Olloz. *Erfahrungen mit dem Location API (JSR 179).* [http://www.gruntz.ch/papers/LocationAPI.pdf], Letzer Zugriff: 06.05.2009.

[Völ09] Max Völkel. *Vortrag 12: Das Grafikdateiformat PNG „Portable Network Graphics“.* [http://goethe.ira.uka.de/seminare/redundanz/vortrag12/], Letzer Zugriff: 02.04.2009.

[Vlk09] VlkGPS. *vlkGPS - simple GPS navigation for java phones - best for geocaching, walking, bicycle, sport, ...* [http://vlkgps.bielyvlk.sk/?dest=vlkgps], Letzer Zugriff: 26.01.2009.

[Vou09] René-Pierre Vouri. *Kommunikation mit Mobilen Endgeräten.* [http://krottmaier.cgv.tugraz.at/docs/seminar/sem2002_kommunikation.pdf], Letzer Zugriff: 13.05.2009.

[Web09a] Joern Weber. *Naviuser - Das Forum für GPS-Einsteiger und -Experten - **GPS-Empfänger, prinzipieller technischer Aufbau**.* [http://www.naviuser.at/forum/showthread.php?p=1307], Letzer Zugriff: 24.03.2009.

[Web09b] Joern Weber. *Naviuser - Das Forum für GPS-Einsteiger und -Experten - **GPS-Modul, prinzipieller Aufbau**.* [http://www.naviuser.at/forum/showthread.php?t=359], Letzer Zugriff: 24.03.2009.

[Wil09] Christian Rathemacher, Friederike Wild. *Kommunikation zwischen mobilen Devices mit Bluetooth.* [http://www.mi.fh-wiesbaden.de/~barth/mobile/ws0607/Bluetooth.pdf], Letzer Zugriff: 21.05.2009.

[wm09] www.at mix.de. *EPOC 32.* [http://www.at-mix.de/epoc_32.htm], Letzer Zugriff: 23.02.2009.

[Zen05] Chris Rupp, Jürgen Hahn, Stefan Queins, Mario Jeckle, Barbara Zengler. *UML 2 - glasklar - Praxiswissen für die UML-Modellierung und -Zertifizierung.* 2.Auflage. Hanser, 2005.

Glossar

Akkumulator

Bei einen Akkumulator handelt es sich um einen Speicher für elektrische Energie.

Bounding Box

Bounding Box bedeutet zu deutsch Zeichenbox. In Zusammenhang mit dieser Arbeit stellt solch eine Box die räumliche Begrenzung einer Kartenkachel dar. Dies geschieht durch die Angaben von Koordinaten der linken oberen Ecke, sowie der rechten unteren Ecke einer Kachel.

Geocaching

Unter Geocaching versteht man eine moderne Art der Schnitzeljagd. Auf verschiedenen Internetportalen findet der Interessierte die Informationen über die Koordinaten eines versteckten Schatzen, dem so genannten Geocache. Mit Hilfe der Koordinaten, kann mit der suche nach den Schatz begonnen werden. Dieser ist an ungewöhnlichen Plätzen versteckt. Nach erfolgreichen finden des Geocaches, wird dieser durch den Finder gehoben und in der Regel erneut versteckt.
Quelle: [geo09b]

GPL (GNU General Public License)

GNU General Public License (oft abgekürzt GPL) ist eine von der Free Software Foundation herausgegebene Lizenz mit Copyleft für die Lizenzierung freier Software.
Quelle: [fE09c]

GPS-Maus

Mit Hilfe einer GPS-Maus können GPS-Positiondaten empfangen werden. Sie kann über eine Schnittstelle mit einem mobilen Endgerät verbunden werden. Die GPS-Maus dient lediglich zum empfangen der GPS-Daten. Diese Daten werden durch das Gerät in ein definiertes Datenformat umgewandelt (i.d.R. NMEA). Quelle: [fE09d]

JDK

Das JDK ist eine von Sun geschaffene Entwicklungsumgebung. Mit Hilfe dieser Entwicklungsumgebung ist es möglich, Java-Applikationen, Java-Applets und Java-Komponenten zu erstellen.
Das JDK enthält neben dem Java Runtime Environment(JRE), folgende Komponenten:

- *javac* (Java-Compiler)
- *javadoc* (zur Erstellung der Dokumentation)
- *jar* (zur Erstellung von Java-Archive)
- *jarsigner* (zum Signieren von Java-Anwendungen oder Bibliotheken)

- *htmlconverter* (Java Plug-in HTML Converter)
- *appletviewer*

Quelle: [Ste01]

W3C - World Wide Web Consortium

Das 'W3C' ist ein internationales Konsortium für das entwickeln von Web-Standards und Richtlinien. Das Hauptziel des W3C ist es: 'Dem World Wide Web dadurch seine vollen Möglichkeiten zu erschließen, dass Protokolle und Richtlinien entwickelt werden, die ein langfristiges Wachstum des Web sichern.'[IJ09] Quelle:[Con09a]

WGS84

Das ***World Geodetic System 1984*** *(WGS84)* ist das Referenzsystem für GPS. Dieses geozentrische dreidimensionale Koordinatensystem wird also für die dreidimensionale Positionsbestimmung benötigt. Das WGS84 wurde himmelsfest realisiert. Neben dem WGS84 Referenznetz gehören unter anderem noch:

- ein mathematisches Modell des Schwerefeldes der Erde,
- die Lichtgeschwindigkeit im Vakuum,
- das Produkt aus der Gravitationskonstante und der Masse der Erde.

Quellen: [Bauge] und [Pet]

WMS-Client

Ein WMS-Client (Web Map Service-Client) ist eine rechnergestützte Anwendung für die Online-Darstellung von dynamischen oder statischen Karten. Quelle: [Sta07]

A Inhalt der CD

Die dieser Arbeit beiliegende CD enthält folgenden Inhalt:

- die Diplomarbeit im PDF - Format „*Diplomarbeit-Breitlow-2009.pdf*“.
- der Source Code des Entwickelten Programms: „*geoGLISgps.zip*“
- eine Excel Tabelle, die die in Zusammenhang mit der Arbeit getesteten Software „*SoftwareTest.xls*“.
- die Entwicklungsumgebung für die Programmierung mit:
  - Eclipse: „*eclipse-java-ganymede-win32.zip*“
  - WTK 2.5: „*sun_java_wireless_toolkit-2_5_2-windows.exe*“
- außerdem nützliche Hilfsprogramme zum Bearbeiten von Geodaten
  - POI Konverter: „*POIConverter_setup408.zip*“
  - GPS Babel: „*gpsbabel-1.3.6.zip*“
  - GPS-Track Analyse: „*GPS-Track-Analyse.NET.EXE*“
  - EasyGPS: „*SetupEasyGPS.exe*“
- Source Code Beispiele
  - VlkGPS: „*vlkgps-0.8.0-src.tar*“
  - GPSTrack: „*GpsTrackSource11.zip*“
  - LocateME: „*locateme_source_1.0.zip*“
  - interes GPS: „*gps_tst.rar*“
  - Bluetooth GPS: „*GPS.zip*“
- genutztes Datenmaterial
  - Kartenmaterial der geoGLIS oHG: „*020_kerstin_woldegk.zip*“, „*kacheln_neubrandenburg.zip*“