Peter Goda, Bakk.techn.

Bereitstellung von Infrastruktur für die WLAN Indoor-Positionierung mit einem Smartphone

MASTERARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur/in

Masterstudium Geomatics Science

Technische Universität Graz

Betreuer:

Ao.Univ.-Prof. Dipl.-Ing. Dr.techn. Wieser Manfred

Mitbetreuer:

Dipl.-Ing. Thomas Moder

Institut für Navigation

Graz, Juni 2014

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

Graz, am ....

(Unterschrift)

Statutory Declaration

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

....

date (signature)

Kurzfassung

Diese Abhandlung beschreibt die funktionsfähige Infrastruktur für die Kommunikation zwischen einem Smartphone und einem Webservice mit dem Ziel, eine mobile Indoor-Positionsbestimmung durchzuführen.
Im Zuge dieser Arbeit wird ein Webserver mit einer PostgreSQL Datenbank eingerichtet. Ein Apache HTTP Server wird konfiguriert und für den Zugriff aus dem Internet freigegeben. Es wird ein PHP Webservice programmiert, welches die Anfragen vom Smartphone empfängt und bearbeitet. Besonderes Augenmerk wird auf die Entwicklung einer Smartphone-Applikation gelegt, welche einerseits der Messung der WLAN Signalstärke an einem örtlich lokalisierten Referenzpunkt dient und andererseits die Positionsbestimmung mittels WLAN Fingerprinting beherrscht. Abschließend wird eine webbasierte Verwaltungsoberfläche erstellt.
Unter Zuhilfenahme dieser Infrastruktur können die gemessenen Referenzpunkte der Datenbank hinzugefügt und anschließend in Echtzeit für die Positionierung innerhalb eines Gebäudes verwendet werden.
Für weiterführende Untersuchungen kann die Methode der Positionsberechnung durch alternative Ansätze erweitert werden. Ebenfalls können zusätzlich gemessene Parameter im Webservice implementiert werden.

Abstract

This paper describes a functional structure for the communication between a Smartphone and a Web service with the aim to undertake a mobile indoor positioning.
As part of this work a Web server with a PostgreSQL database is established. An Apache HTTP Server is set up and accessible from the Internet. A PHP Web service which receives and processes the requests from the Smartphone is programmed. Special attention is put on the development of a Smartphone application which, on the one hand, measures the signal strength of the WLAN from a local reference point and, on the other hand, is able to do a positioning through WLAN Fingerprinting. And finally, a web-based management interface is developed.
With the aid of this infrastructure the measured reference points can be added to the database and afterwards used for positioning inside a building in real-time.
For further examinations the method of position calculation can be expanded with alternative approaches. Also, additional parameters can be implemented in the Web service.

Danksagung

Ich möchte mich bei allen herzlich bedanken, die mir bei der Erstellung meiner Masterarbeit mit Rat und Tat zur Seiten gestanden sind.
Mein besonderer Dank gilt Herrn Ao. Univ.-Prof. Dipl.-Ing. Dr.techn. Wieser Manfred für die Themenstellung und die Betreuung während der gesamten Zeit.
Des Weiteren bedanke ich mich bei meinen Kollegen Daniel Koch und Thomas Moder für den intensiven Ideenaustausch sowie die hilfreichen und konstruktiven Gespräche. Besonders hervorheben möchte ich Thomas, der mir wichtiges Material und Informationen für meine Masterarbeit zur Verfügung gestellt und mich intensiv betreut hat.
Ebenfalls richte ich meinen Dank an Herrn Robert Mayr, der mich stets motiviert und speziell beim Konfigurieren des Webservers ein wichtiger Begleiter war.
Insbesondere möchte ich mich bei Frau Andrea Hofer und meinem Freund Gerry für die kritische, konstruktive und präzise Durchsicht dieser Masterarbeit bedanken.
Besonders danke ich meiner Familie und meinen Freunden, die mir im Laufe meines Studiums eine große Stütze gewesen sind.
Schlussendlich möchte ich mich ganz herzlich bei meiner Verlobten Tanja bedanken, ohne sie wäre ich niemals fertig geworden. Sie hat mich in allen Phasen meines Studiums und ganz speziell beim Schreiben meiner Masterarbeit tatkräftig unterstützt und ist mir bei Problemen mit der Formulierung immer zur Seite gestanden.

Abkürzungsverzeichnis

AJAX	Asynchronous JavaScript and XML
AoA	Angle of Arrival
CoO	Cell of Origin
dB	Dezibel
dBm	Dezibel Miliwatt
ESA	European Space Agency
GLONASS	Globalnaja nawigazionnaja sputnikowaja sistema
GPS	Global Positionig System
GSM	Global System for Mobile Communications
HTTP	Hypertext Transfer Protocol
IEEE	Institute of Electrical and Electronics Engineers
IMU	Inertial Measurament Unit
INAV	Indoor Navigation
JSON	JavaScript Object Notation
LBS	Loction-Based Service
MAC	Media Access Control
OFDM	Orthogonal Frequency-Division Multiplexing
PDR	Pedestrian Dead Reckoning
PHP	PHP Hypertext Preprocessor
RFID	Radio-Frequency Identification
RSSI	Received Signal Strength Indicator
SMS	Short Message Service
SQL	Structured Query Language
SSID	Service Set Identifier
TDoA	Time Difference of Arrival
ToA	Time of Arrival
UML	Unified Modeling Language
UWB	Ultra Wide Band
WGS84	World Geodetic System 1984
WLAN	Wireless Local Area Network
XML	Extensible Markup Language

Inhaltsverzeichnis

1. Einleitung

Heutzutage werden Navigationsgeräte zur Orientierung und zur Positionsbestimmung verwendet. Mit Hilfe von GNSS Satelliten, die sich im Weltall befinden, kann überall auf der Welt die Position bestimmt werden.
Diese Methode hat allerdings den Nachteil, dass sie innerhalb von Gebäuden nicht funktioniert, da keine direkte Sichtverbindung zwischen Empfänger und Satelliten besteht.
Infolge der technologischen Entwicklung erlaubt ein Smartphone, ein mit vielen Sensoren ausgestatteter „Mini-Computer“, die Positionsbestimmungsmöglichkeiten zu erweitern und für viele Menschen kostengünstig nutzbar zu machen.
Da heutzutage immer größer werdende Gebäudekomplexe mit hoher Personen-Frequenz entstehen (beispielsweise Shopping-Zentren, Museen, U-Bahnstationen oder Krankenhäuser) wird es zunehmend schwieriger, sich in diesen zurecht zu finden.
Indoornavigation kann dort helfen, wo die Funktionalität herkömmlicher Positionsbestimmungsmethoden (beispielsweise mittels Satelliten) stark eingeschränkt ist.
Die Aufgabe dieser Arbeit besteht darin, die notwendige Hard- und Softwareumgebung für einen Testaufbau zur Indoorpositionierung mittels WLAN zu schaffen. Die Positionsberechnung soll auf einem Webserver durchgeführt werden und die dafür notwendigen Schnittstellen sind zu definieren. In weiterer Folge wird eine Smartphone-Applikation programmiert, welche die Datenaufnahme einer Referenzumgebung sowie die anschließende Positionierung innerhalb von Gebäuden unter Zuhilfenahme einer Datenbank unterstützt.

1.1. Detaillierte Aufgabenstellung

Die Kernaufgabe beschäftigt sich mit dem Aufbau eines Systems, welches seine eigene Position in einem Gebäude mit Hilfe von WLAN Accesspoints, gemessenen Referenzpunkten (welche einer Datenbank zu entnehmen sind) und einem Smartphone in Echtzeit bestimmen kann.
Folgende Hardware ist als Voraussetzung notwendig:

- Webserver für die Datenbank und die Webapplikation
- Smartphone zur Positionsbestimmung und Aufnahme der Referenzpunkte
- WLAN Accesspoints zur Positionsbestimmung und Definition der Referenzpunkte

Zusätzlich werden folgende Softwarekomponenten entwickelt:

- Android-Applikation für die Messdatenaufnahme der Referenzpunkte sowie für die Anzeige der aktuell ermittelten Position
- Webservice zur Berechnung der aktuellen Position sowie als Schnittstelle zu einer Datenbank, welche die gemessenen Referenzpunkte enthält
- Grafische webbasierte Oberfläche zur Datenbankverwaltung

Für die Positionsberechnung wird die Methode des Fingerprinting angewandt. Da die Applikation für weitere Untersuchungen verwendbar sein muss, sind eine gute Dokumentation sowie ein modularer Aufbau der Software notwendig.

1.2. Struktur der Arbeit

In Kapitel 2 werden die theoretischen Grundlagen für die Indoor-Positionierung behandelt. Es werden unterschiedliche Methoden vorgestellt, um in einem Gebäude die aktuelle Position zu bestimmen. Man bekommt einen Überblick über Wireless LAN sowie allgemeine Positionsbestimmungstechniken.
In Kapitel 3 werden die notwendigen Anforderungen an die Hard- und Software aufgelistet.
Kapitel 4 beschäftigt sich im Detail mit den Schnittstellen zwischen der Smartphone-Applikation und dem Webservice.
Darauf aufbauend behandelt Kapitel 5 die Installation des Webservers und der Datenbank.
Kapitel 6 beschreibt den Aufbau der Smartphone Applikation und deren Funktionalität.
Nach Erstellen der Applikation wird die Funktionalität der Positionsbestimmung mittels WLAN implementiert, womit sich Kapitel 7 im Detail beschäftigt.
Anschließend werden in Kapitel 8 der Aufbau der Datenbank sowie der webbasierte Verwaltungsoberfläche skizziert.
Abschließend werden in Kapitel 9 die Aufnahme der Referenzpunkte, der Webserver sowie die Android-Applikation auf deren Funktionalität getestet.

2. Theoretische Grundlagen

Im Folgenden werden die notwendigen Grundlagen dieser Arbeit erläutert. Im Speziellen geht es um die verschiedenen Methoden einer Positionsbestimmung und man bekommt einen kurzen Überblick über WLAN.

2.1. Positionsbestimmung mittels Radiowellen

Dieses Kapitel führt in die Thematik der Positionsbestimmung ein. Grundlegend gibt es verschiedene Methoden, um die aktuelle Position zu bestimmen. Anschließend werden ausgewählte Methoden für die Bestimmung der Positionen im Allgemeinen kurz beschrieben.

2.1.1. Messung der Laufzeit - Time of Arrival (ToA)

Bei dieser Methode wird die Distanz zwischen zwei Punkten gemessen. Die Signalausbreitungsgeschwindigkeit jedes Signals entspricht der Lichtgeschwindigkeit und ist somit eine endliche Größe. Auf Grund der gemessenen Sende- und Empfangszeit ergibt sich aus der Differenz die Signallaufzeit, wodurch man die Entfernung des Empfängers vom Sender berechnen kann. Um diese Methode erfolgreich ausführen zu können, ist eine exakte Messung der Signallaufzeit vom Sender zum Empfänger notwendig. Abbildung 2.1 zeigt schematisch die Distanzmessung durch ToA.
Zur Positionsbestimmung mittels ToA müssen von verschieden lokalisierten Sendern, welche untereinander ständig zeitlich synchron gehalten werden müssen, Daten empfangen werden [1]. Zur Lokalisierung im Raum werden mindestens vier empfangene Signale benötigt, für Bestimmungen in der Ebene sind drei ausreichend. Durch zusätzliche Messdaten können Störeinflüsse minimiert werden.

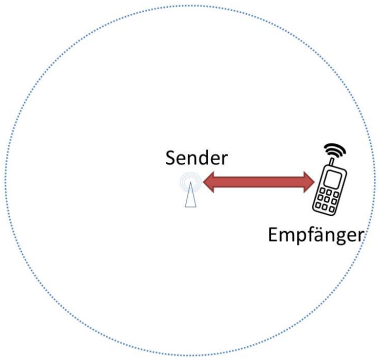


Abbildung 2.1.: Distanzmessung durch ToA

2.1.2. Laufzeitdifferenz - Time Difference of Arrival (TDoA)

Bei dieser Methode senden mehrere Sender zeitgleich ein Signal aus. Auf Grund der unterschiedlich gemessenen Laufzeitdifferenzen kann die Position des Geräts näherungsweise bestimmt werden. Es kann eine höhere Genauigkeit als bei der ToA Methode erzielt werden. Der Vorteil gegenüber der ToA Methode ist, dass zwei synchronisierte Sender Signale übermitteln. Sind die Standorte der Sender bekannt, so liegt der Empfänger auf einer hyperbolische Kurve. Um die Genauigkeit zu steigern, können weitere Messungen zu anderen Referenzpaaren durchgeführt werden. Auf dem Schnittpunkt der Hyperbeläste aller gemessenen Referenzpaare liegt der Empfänger [1]. Abbildung 2.2 zeigt die Distanzmessung durch TDoA.

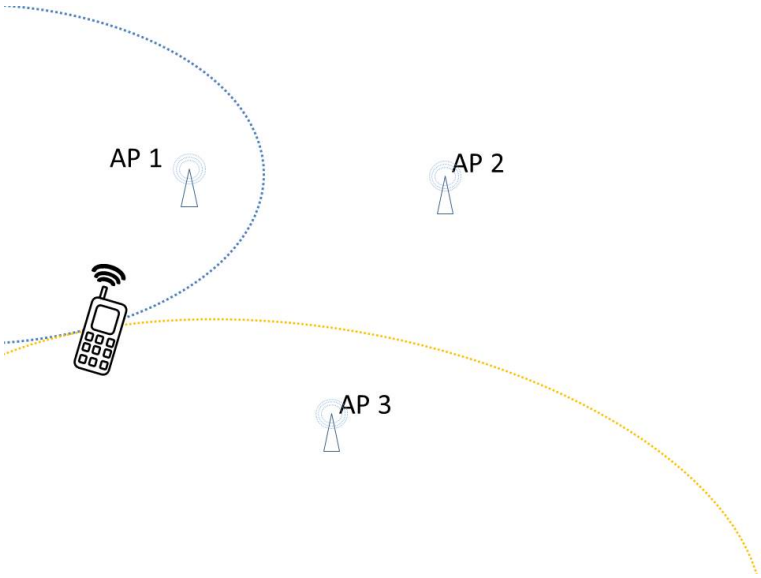


Abbildung 2.2.: Distanzmessung durch TDoA

2.1.3. Eingangswinkel eines Signals - Angle of Arrival (AoA)

Man verwendet bei dieser Methode die Information über den Eingangswinkel des empfangenen Signals und kann somit die Lage des Senders eruieren. Ist die Richtung aus der ein Signal gesendet wird bekannt, so besteht die Möglichkeit einer zweidimensionalen Ortsbestimmung. Voraussetzung dafür ist die exakte Kenntnis über die Position der Sender. Des weiteren muss der Empfänger eine Winkelmessung der Signale beherrschen. In den meisten Fällen ist diese Eigenschaft jedoch nicht vorhanden [1]. Beispielsweise wird diese Methode in der Luft- und Schifffahrt angewandt. In Abbildung 2.3 sieht man die schematische Darstellung einer Positionierung mittels Winkelberechnung.

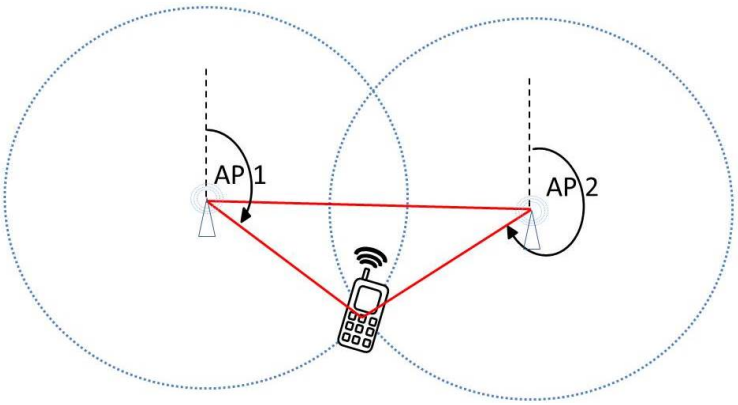


Abbildung 2.3.: Positionierung durch Winkelberechnung

2.1.4. Zellenreichweite eines Senders - Cell of Origin (CoO)

Der Sender übermittelt seine einzigartige Netzwerkkennung an den Empfänger. Wenn man die Lage des Senders kennt, weiß man in etwa die Position des Empfängers, wobei die Genauigkeit von der Größe der Zelle abhängt. Je kleiner die Zellen sind, desto präziser kann man den Standort des Empfängers bestimmen. Zum Beispiel kann man mit den Funkzellen des Handynetzes in Städten eine Genauigkeit von 100 bis 300 Metern erreichen. In ländlichen Gebieten hingegen können sich die Zellen über mehrere Kilometer ausdehnen. Daraus kann man schließen, dass es sich um eine sehr ungenaue Positionsbestimmung handelt [1]. Abbildung 2.4 zeigt ein zellbasiertes Verfahren.

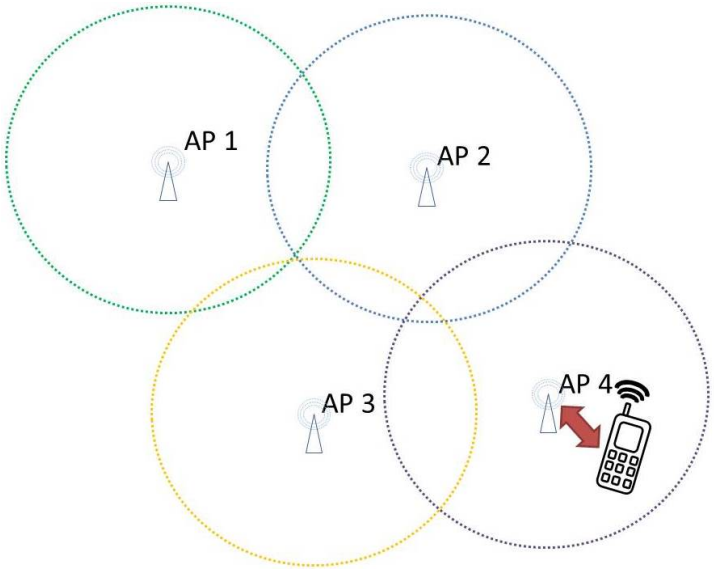


Abbildung 2.4.: Zellenbasiertes Verfahren

2.1.5. Signalstärke - Received Signal Strength Indicator (RSSI)

Die RSSI Methode bietet eine weitere Form, um den Abstand zwischen einem Sender und dem Empfänger zu ermitteln, indem man zur Berechnung des Abstandes die Signalstärke benutzt. Die Kenntnis über das Verhältnis des Abstandes der Signalstärke zum Sender ermöglicht Rückschlüsse über die Entfernung zum Empfänger. Dabei ist zu berücksichtigen, dass die Stärke des Signals quadratisch mit der Entfernung vom Sender abnimmt, da man von einer kugelförmigen, isotropen Ausbreitung ausgeht. Wird die Feldstärke an verschiedenen Referenzpunkten gemessen, können diese Werte zur Positionsbestimmung verwendet werden. Problematisch ist, dass Mauern und Gegenstände das Signal zusätzlich abschwächen und reflektieren, wodurch Mehrwegeffekte (multipath) entstehen [1].

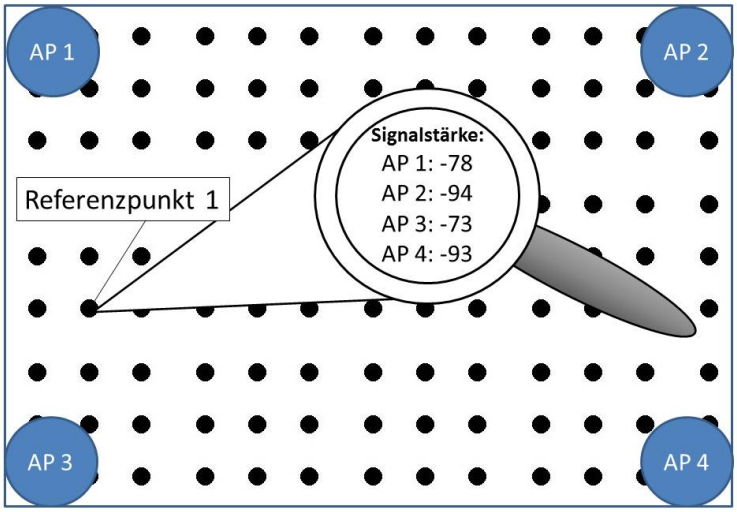


Abbildung 2.5.: RSSI Messung an einem Referenzpunkt

Pegelwerte und Dezibel

In der Nachrichtentechnik werden Signalstärken im logarithmischen Maß, dem Dezibel (dB) angegeben. Unter Dezibel versteht man eine Pseudoeinheit, die aus einem Verhältnis zweier Pegelwerte P_1/P_0 errechnet wird. Die Pegeldifferenz ΔL ist definiert als Logarithmus zur Basis 10 und wird wie folgt berechnet:

$$\Delta L = 10 \log_{10} \frac{P_1}{P_0} \tag{2.1}$$

Das Leistungsverhältnis P_q jedes Pegelwertes lässt sich daraus wie folgt umkehren

$$P_q = 10^{\Delta L/10} \tag{2.2}$$

Aus der Formel lässt sich herauslesen, dass es sich bei der Berechnung des Leistungsverhältnisses um ein Zehntel der Pegeldifferenz handelt. Da die Pegeldifferenz jedoch ohne physikalische Einheit ist, wurde Dezibel (Zehntel -Bel) als Pseudoeinheit, benannt nach Alexander Graham Bell, dafür festgelegt.
Des Weiteren ist zu erwähnen, dass positive dB-Werte einer Leistungssteigerung und negative dB-Werte einer Leistungsminderung entsprechen.
In der Nachrichtentechnik wird die Sendeleistung in Milliwatt (mW) angegeben. Daraus wird das Leistungsverhältnis in der Einheit dBm (Dezibel Milliwatt) verwendet (beispielsweise gilt: $30dBm = 10 \log_{10} \frac{1000mW}{1mW}$) [2].
Bei einem WLAN-Signal variiert das Verhältnis der Dämpfung abhängig vom durchdrun-

genen Material und kann aus Tabelle 2.1 entnommen werden.

Material	2,4-GHz-Dämpfung	5-GHz-Dämpfung
Leichtbeton 11,5 cm	≈ 12 dB	≈ 19 dB
Hochlochziegel 11,5 cm	≈ 7 dB	≈ 10 dB
Lehmstein 11,5 cm	≈ 22 dB	≈ 36 dB
Kalksandstein 24 cm	≈ 9.5 dB	≈ 23 dB
Leichtbeton 30 cm	≈ 26 dB	≈ 35 dB
Stahlbeton 16 cm	≈ 20 dB	≈ 32 dB
Hochlochziegel 36 cm	≈ 26 dB	≈ 50 dB
Tondachziegel 1,3 cm	≈ 3 dB	≈ 8 dB
2 - fache Wärmeschutzverglasung	≈ 33 dB	≈ 27 dB

Tabelle 2.1.: Dämpfungswerte von Baumaterialien bei einem WLAN Signal [2]

Wie aus Tabelle 2.1 entnommen werden kann, besteht für unterschiedliche Baumaterialien kein linearer Zusammenhang zwischen der Dämpfung und der Frequenz des gesendeten Signals.

2.1.6. Überblick der Positionierungsmethoden

Die oben genannten Methoden zur Positionsbestimmung können zu zwei grundsätzlichen Methoden zusammengefasst werden. Zum einen zu Trilateration, welche mittels Distanzmessung die Position bestimmt und zum anderen die Triangulation, welche die Position mit Hilfe von Winkelangaben berechnet. Die oben angeführte Aufzählung ist jedoch nicht vollständig, da noch weitere Methoden, wie beispielsweise die visuelle Positionsbestimmung, existieren. Für die Indoorpositionierung sind diese jedoch nicht relevant, daher wird nicht näher darauf eingegangen [3].
In Abbildung 2.6 wird ein Überblick der Positionstechniken dargestellt.

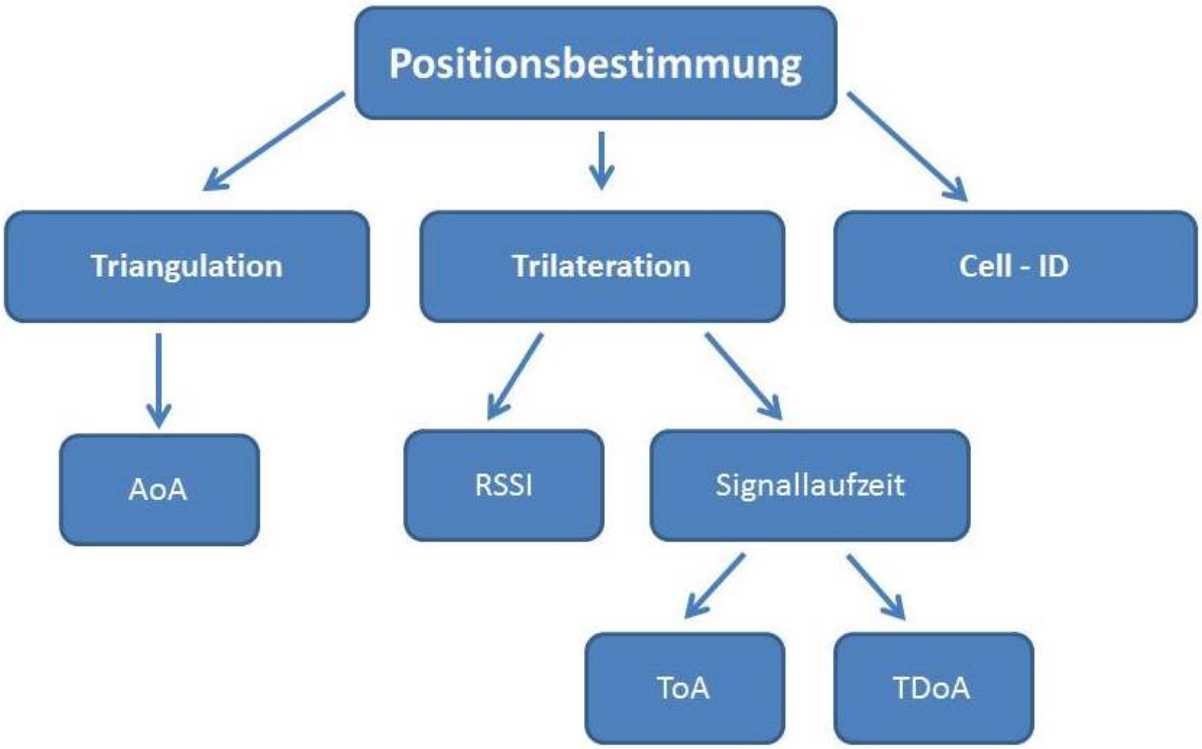


Abbildung 2.6.: Überblick über die Positionierungstechniken

In Tabelle 2.2 werden die einzelnen Methoden mit ihren Vor- und Nachteilen gegenübergestellt.

Methode	Vorteile	Nachteile
AoA	nur zwei Basisstationen werden benötigt; keine Synchronisation der Uhren notwendig	komplexe Hardware; Qualität der Antenne spielt eine wesentliche Rolle
CoO	einfache Methode; kompatibel mit vorhandener Hardware	geringe Positionsgenauigkeit
RSSI	kompatibel mit vorhandener Hardware; keine Synchronisation der Uhren notwendig; hohe Genauigkeit möglich	Datenbank muss vorhanden sein; Mauern und Gegenstände schwächen das Signal ab
TDoA	sehr hohe Genauigkeit kann erreicht werden; keine zeitliche Synchronisation der Empfänger notwendig	an den Basisstationen sollten Uhren (Atomuhren) mit hoher Genauigkeit vorhanden sein
ToA	sehr hohe Genauigkeit	zeitliche Synchronisation der Sender notwendig

Tabelle 2.2.: Übersicht der Positionierungsmethoden [4], [5], [6], [7]

2.2. Positionsbestimmung mit Hilfe von Satelliten

In diesem Kapitel wird ein kurzer Überblick über die verschiedenen Technologien der Positionsbestimmung mit Hilfe von Satellitensystemen aufgezeigt. Aus den Postionen der Satelliten lässt sich anhand von Laufzeitmessungen die Position des Empfängers jederzeit und überall auf der Welt und mit hoher Genauigkeit bestimmen. Die einzelnen Systeme wie GPS, GALILEO und GLONASS werden nachfolgend vorgestellt.

2.2.1. GPS

GPS (Global Positioning System) ist ein System, welches dem Nutzer erlaubt, überall auf der Erde seine exakte Position zu bestimmen. Es besteht nominell aus 24 Satelliten in zirka 20.000 km Höhe. Die Satelliten senden kontinuierlich ihre sich ständig ändernden Positionsdaten, sowie die genaue Uhrzeit. Aufgrund der Signallaufzeiten können GPS-Empfänger ihre eigene Position und Geschwindigkeit berechnen. Die Satelliten sind in sechs Bahnebenen mit einer Umlaufzeit von etwa zwölf Stunden so verteilt, dass über jedem Punkt der Erde mindestens vier Satelliten gleichzeitig am Himmel stehen. Mit GPS erreicht man bei idealen Bedingungen (zum Beispiel keine Abschattung) eine Genauigkeit

von etwa drei Metern in der Lage und etwa zehn Metern in der Höhe. [8]

Abbildung 2.7.: GPS Satelliten umrunden die Erde (Foto: NOAA)
[9]

2.2.2. GALILEO

GALILEO ist ein europäisches Satellitensystem, welches von der Europäischen Union und der europäischen Raumfahrtbehörde ESA 1994 initialisiert worden ist. Es ist das erste System, welches nicht auf militärischer Basis, sondern für zivile Nutzung entwickelt worden ist. Dieses System besteht bei Vollendung aus dreißig Satelliten, welche mit zwei für zivile Nutzung verfügbaren Frequenzen senden. Dadurch kann eine höhere Genauigkeit als bei anderen satellitengestützten System erreicht werden. Hauptaugenmerk dieses Projektes liegt auf der Verfügbarkeit und Nutzung für sicherheitsrelevante Applikationen, wie beispielsweise Flugzeuge beim Landeanflug. [8]

Abbildung 2.8.: GIOVE-B Satellit von GALILEO (Foto: ESA 2012)
[10]

2.2.3. GLONASS

GLONASS (Globalnaja nawigazionnaja sputnikowaja sistema) ist ein russisches Satellitensystem, welches GPS sehr ähnlich ist. Im Gegensatz zu GPS Satelliten sendet jeder GLONASS Satellit das Signal auf einer eigenen Frequenz, aber mit identischer Codierung. GLONASS kann nicht nur für militärischen Gebrauch, sondern auch für zivile Nutzung verwendet werden. Zwischenzeitlich waren zu wenig Satelliten für eine Positionsbestimmung verfügbar. In den letzten Jahren wurde die notwendige Mindestanzahl an Satelliten wieder hergestellt und das System reaktiviert.

2.3. Technologien zur Positionsbestimmung innerhalb von Gebäuden

Heutzutage wird es für Menschen immer wichtiger, sich ständig mitzuteilen und andere an ihrem Leben teilhaben zu lassen. Als Beispiel sei die Internetseite „Facebook“ genannt, die dem gläsernen Menschen ein großes Stück näher kommt. Neben sogenannten Statusmeldungen kann man weiteren Personen auch den persönlichen Standort mitteilen. Auch diese Internetplattform nutzt bereits eine Form der Indoorpositionierung, da GPS-Navigation in den meisten Gebäuden nicht oder nur sehr eingeschränkt möglich ist. Aufgrund der Dämpfung durch beispielsweise Wände und Decken können die Satellitensignale nicht mehr empfangen werden.
Innerhalb von Gebäuden kann die vorhandene Infrastruktur kostengünstig für die Posi-

tionsbestimmung genutzt werden. Mittels geringer Investitionskosten können an Decken und Wänden neue Sensoren und Geräte angebracht werden, wobei vorhandene Strom- und Datennetzwerke verwendet werden können. Dem gegenüber steht ein hoher finanzieller Aufwand, um beispielsweise Satelliten in das Weltall zu schießen.
In den nachfolgenden Abschnitten folgt eine Beschreibung von bereits bestehenden Technologien, welche eine Indoorpositionierung ermöglichen und nicht nur in geschlossenen Gebäuden angewandt werden können.

2.3.1. Infrarot

Infrarot ist bereits seit einigen Jahrzehnten in Verwendung und stellt zudem eine kostengünstige Variante dar. Hierbei wird die geringe Reichweite des Signals sowie die Begrenzung durch Wände und dergleichen ausgenutzt. Ein Infrarotsender strahlt Signale aus, die von einem mobilen Gerät empfangen und decodiert werden. Das ausgesendete Signal beinhaltet eine eindeutig codierte Kennung. Voraussetzung für die Kommunikation ist Sichtkontakt zwischen Sender und Empfänger, da sich Infrarot nicht durch Gegenstände hindurch ausbreiten kann. [11]

2.3.2. Visuelle Methoden

Bei der visuellen Methode zur Positionsbestimmung innerhalb von Gebäuden wird eine Kamera verwendet. Man kann zwischen zwei grundlegenden Methoden unterscheiden. Zum einen hat man die Möglichkeit die Postion mittels „Tags Detection“ zu bestimmen und zum anderen kann man eine bildbasierte Positionierungsmethode mittels „Fingerprinting“ verwenden.

- *Tags Detection* - Bei dieser Methode werden Tags verwendet. Tags sind quadratische Felder, die aus beispielsweise 36 schwarz-weißen Quadraten bestehen und somit 36 Bit an Informationen beinhalten. Die Tags werden an speziellen Orten innerhalb eines Gebäudes platziert. Eine dafür entwickelte Software auf einem Smartphone kann mit der vorhandenen Kamera diese Tags identifizieren und somit deren Position ermitteln. Auf diese Art und Weise ist es nicht möglich eine kontinuierliche Positionierung zu gewährleisten, da nicht durchgehend Sichtkontakt zu Tags garantiert ist. Daher dient diese Methode als Unterstützung zur Bestimmung einer absoluten Positionskoordinate, welche beispielsweise für Inertialsensoren benötigt wird.

- *Bildbasierte Positionierungsmethode mittels „Fingerprinting“* Bei dieser Methode ist eine kontinuierliche Positionierung möglich. Es werden als Referenzpunkte Fotos auf-

genommen, die in weiterer Folge zusammen mit den Koordinaten in einer Datenbank gespeichert wurden. Da Fotos viel Speicherplatz benötigen, können mit Hilfe eines „Computer Vision Algorithmus“ bestimmte Merkmale extrahiert und diese anstatt der Fotos in der Datenbank gespeichert werden. Während der Positionsbestimmung werden die charakteristischen Merkmale von aktuell aufgenommenen Fotos mit jenen der gespeicherten Referenzaufnahmen verglichen. Im Vergleich zur „Tags Detection“ ist diese Methode wesentlich komplexer und wird bei Indoornavigation unter anderem bei Robotern verwendet.

Heutzutage haben Smartphones qualitativ hochwertigere Kameras und die Prozessorgeschwindigkeit der Geräte erreicht eine sehr hohe Leistung. Aufgrund dieser technischen Entwicklung sind Smartphones bereits für diese Methode einsetzbar. [12]

2.3.3. Funkwellen

Da Infrarot einige Nachteile mit sich bringt, bieten Funkwellen eine Alternative zur Positionsbestimmung. Will man mit Hilfe dieser elektromagnetischen Wellen Entfernungsmessungen durchführen, so werden die Laufzeit und die Signalstärke gemessen. Es werden hohe Anforderungen an die Hardware gestellt, da sich Funkwellen mit Lichtgeschwindigkeit ausbreiten. Dafür werden hochfrequente Signale verwendet, welche durch Wände und Gegenstände dringen. Dabei können codierte Informationen (aktuelle Position des Senders und weitere relevante Daten) mitgesendet werden. Ein ebenfalls auf Funkwellen basierendes Positionsbestimmungssystem stellt das satellitengestützte GPS dar. Dieses dient zur Lokalisierung im Außenbereich und ist bereits weltweit im Einsatz. Da für diese Positionsbestimmung mindestens vier Satelliten und eine direkte Sichtlinie verfügbar sein müssen, ist dieses System im Indoorbereich nur eingeschränkt bis gar nicht verwendbar. Zu den Funktechnologien, die auch für Indoornutzung geeignet sind, zählen unter anderem die bereits weit verbreiteten WLAN Verbindungen, Bluetooth, UWB (Ultra Wide Band) sowie RFID (Radio-Frequency Identification).

2.3.4. High-Sensitivity GPS

In Gebäuden können GPS Signale von herkömmlichen Navigationsgeräten nur schwer empfangen werden. Aus diesem Grund werden hoch sensible Empfänger (High-Sensitivity GPS) eingesetzt. Im Vergleich zu GPS ist deren Genauigkeit wesentlich geringer, da auch reflektierte Signale noch stark genug sind, um zur Positionsbestimmung verwendet zu werden. [13]

2.3.5. Inertiale Navigation

Für die „Inertial Navigation“ ist eine IMU (Inertial Measurament Unit) mit drei Beschleunigungsmessern und drei Gyroskopen notwendig. Die Messungen aus den Sensoren werden aufintegriert, um Position und Geschwindigkeit zu erhalten. „Intertial Navigation“ bietet eine sehr hohe Genauigkeit bei der Positionsbestimmung. Über einen längeren Zeitraum nehmen die Fehler durch die Beschleunigungsmesser quadratisch beziehungsweise durch die Gyroskope kubisch zu, was zu einer fehlerhaften Positionsberechnung führt. Aus diesem Grund ist regelmäßig eine Positionsbestimmung von außerhalb notwendig. Dazu eignen sich neben GPS auch andere Positionsbestimmungsarten, die eine absolute Position liefern und die Messwerte aktualisieren können. Eine Erweiterung des inertialen Navigationssystems ist PDR (Pedestrian dead Reckoning). Dazu wurden weitere Sensoren wie Schrittzähler und Magnetometer mit dem Inertialen Navigationssystem konstruiert, um die Postion zu berechnen. Dadurch kann eine noch höhere Genauigkeit erzielt werden. [13]

2.3.6. Ultraschall

Statt Infrarot oder Funkwellen können auch Schallwellen, häufig in Form von Ultraschall, als Grundlage für die Positionsbestimmung verwendet werden. Schallwellen breiten sich in Luft mit circa $300\frac{m}{s}$ aus und haben eine Frequenz zwischen 20 kHz und 1 GHz. Zu beachten gilt, dass Schallwellen bei Hindernissen in Abhängigkeit von der Dichte des Mediums reflektiert oder absorbiert werden beziehungsweise durch den Stoff dringen. Schallwellen sind für die Positionsbestimmung mittels der Methode „Time of Arrival“ wesentlich besser geeignet als beispielsweise Funkwellen, da hierbei die Übertragungszeit des Signals deutlich länger ausfällt. Der Vorteil besteht darin, dass sowohl im Empfänger als auch im Sender keine exakte Zeitdifferenzbestimmung notwendig ist. Andererseits ist die Ausbreitungsgeschwindigkeit stark von der Temperatur des Mediums abhängig, was wiederum einen Nachteil darstellt und bei der Berechnung der aktuellen Position berücksichtigt werden muss. [3]

2.4. Methoden der netzwerkgestützten Positionsbestimmung

Zu den Methoden der netzwerkgestützten Positionsbestimmung zählen unter anderem die Funknetze. Im folgenden Kapitel wird das für die Arbeit wichtige Funknetz WLAN

erläutert. Eine weitere Möglichkeit die Postion zu bestimmen, bietet das vorhandene Mobilfunknetz GSM.

2.4.1. Wireless local areal networks (WLAN)

Wireless local areal networks (WLAN) sind drahtlose LAN Verbindungen, die über Funk, basierend auf dem IEEE 802.11 Standard, arbeiten. Bevorzugt werden Datenübertragungen im 2,4 GHz lizenzfreien Frequenzband durchgeführt [14]. Ein weiterer Frequenzblock ist der Bereich von 5,15 GHz bis 5,725 GHz . Ein Netzwerk ist aus mehreren Basisstationen und WLAN-fähigen Empfängergeräten aufgebaut. Diese Basisstationen werden „Accesspoints“ genannt [15].
Digitale Informationen werden mit hoher Datenrate mittels „orthogonalem Frequenzmultiplexverfahren“ (OFDM) von einem Gerät zu einem entfernten gesendet [16]. Es kann eine Übertragungsweite von ein paar hundert Metern ermöglicht werden. Diese Abdeckungsfläche der einzelnen Accesspoints wird „Basic Service Area“ genannt [15].
Die IEEE 802.11g Norm ist seit 2003 zur Verwendung freigegeben und wird seitdem als Standard für funklose LAN Verbindung genutzt. Sie bietet eine Datenübertragungsrate von 54 Mbit/s. Der Standard wurde weiterentwickelt und an den neuesten Stand der Technik angepasst. Es konnte eine Erhöhung des Datendurchsatzes erzielt werden, womit im Standard 802.11n eine Datenübertragungsgeschwindigkeit von 600 Mbit/s, beziehungsweise im Standard 802.11ac jene von 1,3 Gbit/s zur Verfügung steht. Im Jahr 2014 wurde der Standard 802.11ad freigegeben, bei dem bis zu 7 Gbit/s möglich sind. [17]

2.4.2. GSM

GSM (Global System for Mobile Communications) ist ein standardisiertes, digitales Mobilfunknetz. Seine Verwendung findet es hauptsächlich in der Telefonie, in der Internetnutzung und im Senden von Kurzmitteilungen (SMS). Das Mobiltelefon verbindet sich mit einer Basisstation, die sich an einem bekannten Standort befindet. Daraus lässt sich mit Hilfe der Methode CoO (siehe Kapitel 2.1.4) die Position bestimmen. Um eine genaue Positionsbestimmung zu erhalten, sind die meisten Basisstationen mit Richtfunkantennen ausgestattet, mit denen man über die Triangulation mittels der Methode ToA (siehe Kapitel 2.1.1) eine höhere Genauigkeit erzielt. Bezugnehmend auf den aktuellen Stand der Technik liefert diese Methode eine Genauigkeit von ±20 Metern im städtischen sowie ±250 Metern im ländlichen Bereich. Verwendet wird diese Methode beispielsweise beim Rettungsdienst, um den Aufenthalt von Unfallopfern näherungsweise bestimmen zu können. [1]

2.5. Möglichkeiten zur Positionsbestimmung

Die folgende Abbildung 2.9 bietet eine übersichtliche Darstellung, der zuvor genannten Methoden zur Positionsbestimmung.

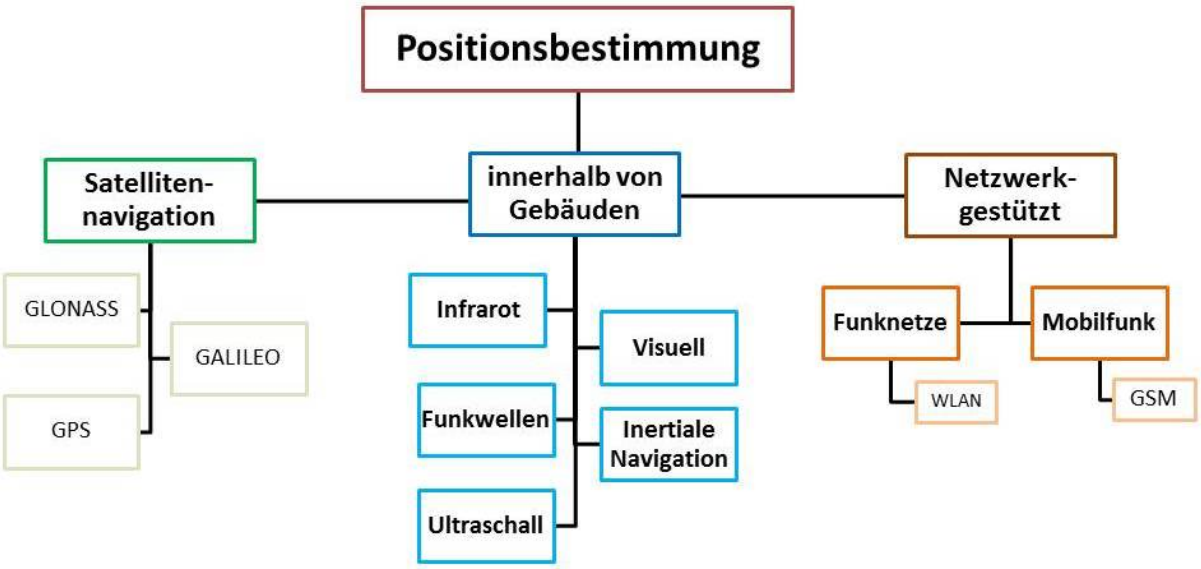


Abbildung 2.9.: Positionsbestimmung

2.6. Positionsbestimmung in einem WLAN Netzwerk

Da bei dieser Arbeit die Positionsbestimmung mittels WLAN erfolgt, soll diese Methode näher betrachtet werden. Im Wesentlichen kommen bei der Positionsbestimmung in einem WLAN Netzwerk zwei Varianten zum Tragen. Nachfolgend werden diese Methoden genauer erläutert.

2.6.1. Zellenbasiertes Verfahren

Das zellenbasierte Verfahren ist in seiner Funktionsweise dem „Cell of Origin“ ähnlich. Technisch gesehen ist es sehr einfach anzuwenden, jedoch liefert es ungenaue Werte. WLAN Funknetze sind in zellularen Strukturen aufgebaut. Bei diesem Verfahren wird die Eigenschaft des Funknetzes ausgenutzt, um über die Position des Accesspoints und dessen Zellengröße die Position des Empfängers einzugrenzen. Bei mehreren Sendern verbindet sich der mobile Empfänger, zum Beispiel ein Smartphone, immer mit der stärksten Basisstation [1]. Ziel ist es, die Accesspoints so zu positionieren, dass ihre Strukturzellen

sich möglichst wenig überlappen. Das führt jedoch zu einer relativ ungenauen Positionsbestimmung.

2.6.2. Fingerprinting Methode

Die Fingerprinting Methode, auch tabellenbasiertes Verfahren genannt, nutzt die WLAN Signalstärke zur Positionsbestimmung. Bei diesem Verfahren werden die Dämpfungs- und Reflektionseigenschaften der Signale genutzt, da diese an jedem Ort eine unterschiedliche Charakteristik aufweisen. Die Berechnung der Postion eines Nutzers findet in zwei Phasen (Online und Offline) statt.
Während der Offline-Phase werden Referenzpunkte koordinativ festgelegt und anschließend werden an diesen die Signalstärken aller verfügbaren Accesspoints gemessen. Durch Berechnung des arithmetischen Mittels aus mehreren Messreihen wird ein Signalstärkenvektor ermittelt, welcher in einer zentralen Datenbank abgespeichert wird.
Es ist bei diesem Verfahren von großer Bedeutung, dass die entstehenden Dämpfungen und Reflektionen (beispielsweise durch Möbel, Wände, etc.) dazu genutzt werden, um eine hohe Genauigkeit erzielen zu können. Je unterschiedlicher die Referenzprofile sind, desto besser sind diese bei der Positionsbestimmung voneinander zu unterscheiden. So wie der Fingerabdruck eines Menschen zur Identifizierung dient, so sind bei dieser Methode die unterschiedlichen Signalstärken ähnlich einem Fingerabdruck, woraus sich auch der Name der Methode ableitet. [18]

Offline-Phase

Die Offline-Phase kann als Kalibrierung gesehen werden. Die Anzahl der Referenzpunkte (Kalibrierungspunkte) ist abhängig vom Aufbau und der Größe des Gebäudes. An jedem dieser Kalibrierungspunkte wird eine Messung durchgeführt. Es ist dabei zu beachten, dass die korrekte Ausrichtung des Benutzers die RSSI Werte beeinflusst.
Befindet sich beispielsweise der Benutzer zwischen dem Accesspoint und dem mobilen Gerät (Smartphone), so wird die Signalstärke abgeschwächt. Im Gegensatz dazu erhält man bei direkter Sichtlinie zwischen dem Accesspoint und dem mobilen Gerät einen höheren Wert. [19]
Aus diesem Grund werden mindestens vier Himmelsrichtungen vermessen und diese Werte in der Datenbank abgespeichert. Die Differenz zweier gemessener Werte in entgegengesetzter Himmelsrichtung kann bis zu fünf Dezibel betragen. [20]
Ziel dieser einzelnen Messungen ist, an jedem Referenzpunkt die Signalstärke zu jedem Accesspoint und mit jeder Orientierungsrichtung zu messen und zu speichern. Auf Grund

der Tatsache, dass die empfangenen Signale von vielen Faktoren beeinflusst werden, müssen zahlreiche Messungen durchgeführt und statistisch zusammengefasst werden.

Online-Phase

In der Online-Phase empfängt das mobile Gerät periodisch RSSI Messungen und vergleicht diese mit jenen aus der Datenbank, welche in der Offline-Phase erstellt worden sind. Daraus wird die Position des mobilen Gerätes zu dem nächstgelegen Referenzpunkt ermittelt.
Es gilt

$$R_{index} = \min_{j=R_{act}} \sqrt{\sum_{i=AP(j)} \Big[SS_{RM}(i,j) - SS_{MEAS}(i)\Big]^2} \tag{2.3}$$

wobei

- R_{index} den Index des Referenzpunktes an dem aktuellen Standort,
- R_{act} sämtliche Referenzmessungen, welche die aktuell sichtbaren Accesspoints beinhalten,
- $AP(j)$ die verfügbaren Accesspoints beim Referenzpunkt j,
- $SS_{RM}(i,j)$ die hinterlegte Offline-Signalstärke an einem Referenzpunkt j von einem Accesspoint i,
- $SS_{MEAS}(i)$ die aktuelle online gemessene Signalstärke von einem Accesspoint i,

darstellt.
Eine Voraussetzung für einen reibungslosen Ablauf der Online-Phase ist eine Internetverbindung zur Datenbank. Weiters ist zu beachten, dass jedes Gerät in seiner Handhabung und Hardware unterschiedlich ist und sich daher die Messwerte geräteabhängig unterscheiden. [18]
In Abbildung 2.10 ist die Online-Phase schematisch dargestellt.

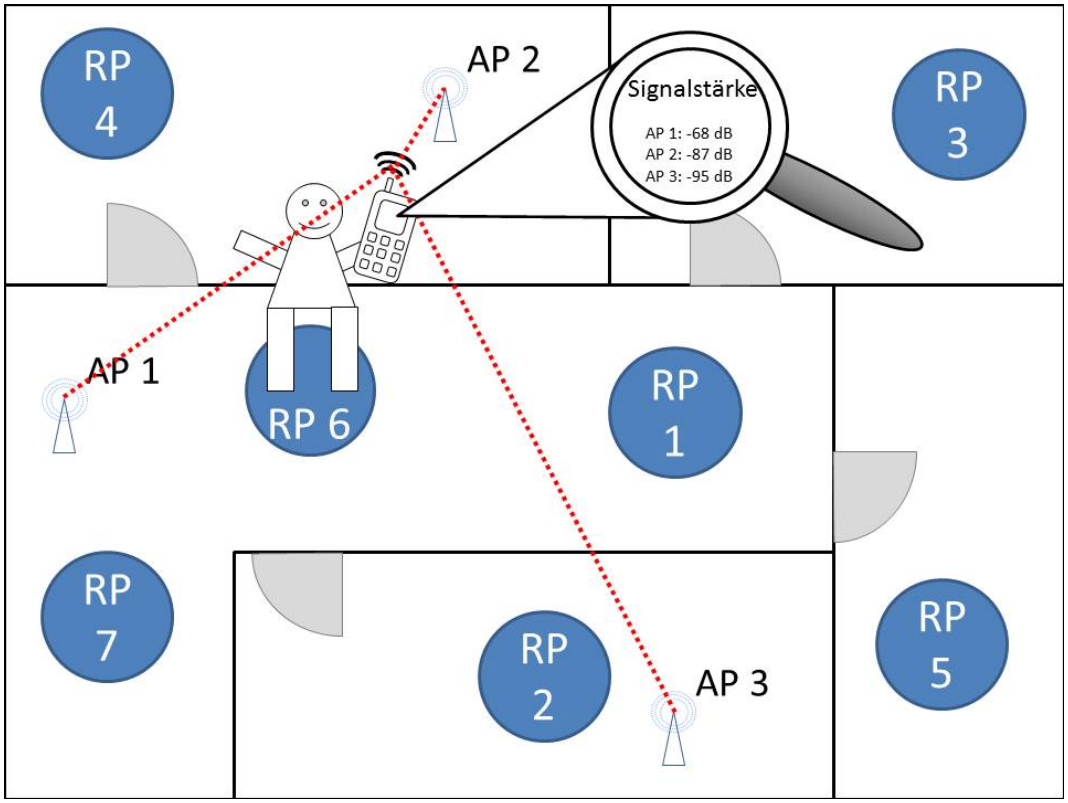


Abbildung 2.10.: schematische Darstellung der Online-Phase

Genauigkeit der Positionsbestimmung

Bezugnehmend auf die Untersuchungen von „Martin et al“ kann man sagen, dass durch verschiedene Messungen mit deren Smartphone Applikation eine Positionsgenauigkeit von bis zu 1,5 Metern erreicht werden kann [21]. In weiterer Folge zeigen die Tests von „Honkavirta et al“, dass die Positionsgenauigkeit nicht von der Dauer der Referenzpunktmessung abhängig ist, denn ab zehn Sekunden Messzeit kann keine relevante Verbesserung der Genauigkeit erzielt werden. Jedoch wirkt sich die Messung in mehrere Himmelsrichtungen positiv auf die Genauigkeit aus. Des weiteren hat „Honkavirta et al“ festgestellt, dass auch ein hohe Anzahl an Accesspoints (mehr als fünf) keine Verbesserung in der Genauigkeit der Positionsbestimmung des Empfängers liefert. [22]

3. Konzeptionierung

Bevor es zur Umsetzung der Software kommt, steht die Planung des gesamten Systems im Vordergrund.

3.1. Anforderungen eines Indoor-Positionierungsdienstes

Zu Beginn muss eine allgemeine Definition des Vorhabens mittels einer Anforderungsanalyse, einem so genannten Soll-Konzept, mit den fachlichen Anforderungen an Funktionen und Prozessen entwickelt werden. Es ist daher zweckmäßig, einen Anforderungskatalog zu erstellen, der sowohl die notwendigen Grundprozesse, die am Anfang zur Verfügung stehen, als auch die absehbaren zukünftigen Anforderungen auflistet.
Erst im Anschluss daran kann mit der Auswahl der Programmiersprache, der Datenbank und der Hardware begonnen werden. Die folgenden Unterabschnitte beschreiben die wesentlichen Anforderungen, die zuerst erfüllt werden müssen.

3.1.1. Anforderungen für die Verwaltung der Daten

Alle gemessenen und zu verarbeitenden Daten werden zentral in einer Datenbank gespeichert. Da es bei der Positionierung auch um Berechnungen von Koordinaten geht, soll die Datenbank über die Möglichkeiten einer „Spatial“-Erweiterung verfügen. Ein Teil der daraus resultierenden wesentlichen Anforderungen sind:

- zentrale, redundanzfreie Datenerhaltung der Messpunkte, Referenzpunkte und Accesspoints
- Speicherung von Koordinaten und Berechnungen im Raum
- Benutzerfreundlichkeit der Software sowohl in der Anwendung als auch in der Administration
- geringe Investitionssumme für Software und Hardware
- Anforderungen der Hardware sollen den geplanten Kapazitäten entsprechen

3.1.2. Anforderungen an den Webserver

Der Webserver ist der zentrale Drehpunkt im gesamten System. Es ist von großer Bedeutung, dass Anfragen schnell bearbeitet werden und in Folge Berechnungen zur Positionsbestimmung rasch durchgeführt werden können. Um diese Kriterien erfüllen zu können, sind folgende Anforderungen notwendig:

- schnelle Internetverbindung
- hohe Rechenleistung der Hardware
- Benutzerfreundlichkeit der Software sowohl in der Anwendung als auch in der Administration
- geringe Investitionssumme für Software und Hardware
- Anpassung und Erweiterungsfähigkeit des Webservers

3.1.3. Anforderungen an das Smartphone

Die Positionierung soll mittels eines Smartphones erfolgen. Daher muss das Gerät auch gewisse Voraussetzungen erfüllen:

- WLAN-fähig
- Internetzugang
- lange Akkulaufzeit
- optional einen Barometer beziehungsweise ein Gyroskop

Die Software des Smartphones muss Folgendem gerecht werden:

- die Benutzerfreundlichkeit der Software am Smartphone muss gegeben sein
- Kommunikation mit dem Webserver
- Erstellung eines Referenzfeldes - einzelne RSSI Werte werden an Referenzpunkten gemessen und die Werte werden in der Datenbank gespeichert
- Positionsbestimmung

3.1.4. Anforderungen an das Webservice

Das Webservice ist für die Kommunikation zwischen dem Programm am Smartphone und der Datenbank zuständig. Hier befindet sich unter anderem der Code für die Positionsberechnung. Dieser Quellcode soll für zukünftige Entwicklungen bearbeitet, verbessert und sogar ausgetauscht werden können, um weitere Funktionalitäten hinzufügen zu können und die Genauigkeit zu steigern.
Dafür muss das Webservice folgende Kriterien erfüllen:

- plattformunabhängiger Zugriff
- Erweiterbarkeit - für zukünftige Anforderungen modular erweiterbar
- Performance - die Verarbeitungsgeschwindigkeit soll bei mehreren gleichzeitig erfolgenden Zugriffen trotzdem schnell sein
- Verwaltung der Referenzpunkte, Messwerte und Accesspoints
- Positionsbestimmung mittels übergebenen Parametern (Mac Adresse, RSSI Wert, Zeit, Datum, etc.)

3.1.5. Anforderungen an die Webseite

Es wird eine übersichtliche Webseite für die Datenbankverwaltung programmiert, damit ein Zugriff auf die Daten schnell und vor allem ortsunabhängig erfolgen kann.
Um diesen Voraussetzungen gerecht zu werden, ist Folgendes zu gewährleisten:

- Verwaltung der Referenzpunkte, Messwerte, Accesspoints
- Darstellung und Editierbarkeit der Daten aus der Datenbank
- Erstellen und Editieren von Referenzpunkten, Messungen und Accesspoints
- Benutzerverwaltung

4. Schnittstellen

Im folgenden Kapitel werden die Schnittstellen einer externen Applikation zur Datenbank sowie zur Berechnung für die Positionsbestimmung beschrieben. Mit Hilfe dieser Schnittstellen können externe Applikationen auf die Werte der Datenbank, in der sich die „Fingerprinting“ Daten befinden, zugreifen und ebenso die Funktion zur Berechnung der aktuellen Position nutzen.

4.1. Definition - Webservice

Um einen abstrahierten Datenaustausch zu ermöglichen, wird ein Webservice eingerichtet, welches auf Anfragen aus dem Internet reagiert. Der verwendete Server ist unter der Adresse `http://www.inav.tugraz.at:8080/service` zu finden.
Wird ein HTTP Request an diese Adresse gesendet, versucht das Webservice die Anfrage abzuhandeln und die geforderten Ergebnisse zu liefern. In diesem Kapitel wird beschrieben, in welcher Form die Anfragen an das Webservice gesendet werden müssen, um diese beantworten zu können.
Das Datenformat bei dem Datenaustausch ist JSON. JSON steht für Javascript Object Notation und wurde von Douglas Crockford definiert. Es handelt sich bei JSON um eine Struktur, die für Menschen und Computer leicht lesbar ist. Es ist im Gegensatz zu XML leichter zu lesen und hat weniger Overhead, dadurch gewinnt es beim Datenaustausch im Web immer mehr an Bedeutung [23] .

4.1.1. Dispatcher

Der Dispatcher ist sozusagen die Einstiegsseite für die Verarbeitung der Anfragen an das Webservice. Dieser dient dazu, Aufrufe auf die jeweiligen Klassen und Methoden automatisch zu verteilen und eine Antwort zu liefern, wenn die Bearbeitung des Aufrufes erfolgreich oder nicht erfolgreich abgeschlossen wurde. Im Speziellen wird der vorhandene HTTP Header ausgelesen. In diesem Objekt steht der übergebene Dienst, welcher aufgerufen werden soll sowie die eventuell notwendigen Parameter. Nach dem Durchlauf des Programmes wird eine Antwort generiert und diese wird dem aufrufenden Request angezeigt.

Die Dispatcher Klasse kann HTTP POST und HTTP GET verarbeiten und mit den darin enthaltenen Variablen die gewünschten Dienste starten [24].
Der genaue Aufbau der Klasse mit allen Methoden ist unter
`http://www.inav.tugraz.at:8080/service/dokumentation.php` zu finden.

Aufruf

Um das Webservice zu starten, ruft man die Adresse
`http://www.inav.tugraz.at:8080/service/dispatcher.php` auf. An diese Adresse werden dann weitere Parameter bei einem GET Aufruf angehängt. Um einen Dienst zu starten, deklariert man **$dienst = Name des Dienstes**. Namen, die sich dafür eignen, werden nachfolgend aufgelistet. Jeder Dienst hat Methoden (Funktionen), die aufgerufen werden müssen. Diese Methoden werden wie folgt **$operation = Name der Operation** aufgerufen.

4.1.2. Accesspoints

Übersicht

Der Dienst „Accesspoints“ beinhaltet alle notwendigen Funktionen, die die Datenbanktabelle „accesspoints“ betreffen. In der Tabelle „accesspoints“ werden die Eigenschaften der Accesspoints wie MAC-Adresse, Kanal, SSID gespeichert. In der Tabelle 4.1 ist eine Zusammenfassung der Methoden, Konstruktoren und Eigenschaften der Klasse *Accesspoints* ersichtlich.

Tabelle 4.1.: Zusammenfassung - Accesspoints

Public Methoden	
boolean	checkAccesspointExists(String mac)
boolean	insertNewAccesspoint(Array DataAccesspoint)

4.1.3. Methoden von Accesspoints

checkAccesspointExists(String mac)

Die Methode „checkAccesspointExists“ prüft, ob die MAC-Adresse schon in der Datenbank vorhanden ist. Davor wird der übergebene Parameter geprüft, um sicher zu gehen, dass es eine gültige MAC-Adresse ist. Als Übergabeparamter muss die MAC-Adresse als String übergeben werden. Wenn die MAC-Adresse in der Datenbank existiert, wird *true*

zurückgegeben, ansonsten *false*. Weiters wird bei einer existierenden MAC-Adresse die ID der MAC-Adresse und die MAC-Adresse in die Eigenschaft *result* geschrieben.

Example - checkAccesspointExists

Der Aufruf für die Abfrage ob ein Accespoint existiert sieht wie folgt aus:
http://www.inav.tugraz.at:8080/service/dispatcher.php?$dienst=accesspoints&$operation=checkAccesspointExists&mac=00:24:51:cb:95:c0
Die Antwort vom Webservice sieht folgendermaßen aus:

```
{
    "checkAccesspointExists":[{
        "mac":"00:24:51:cb:95:c0",
        "0":"00:24:51:cb:95:c0",
        "id":2,
        "1":2}]
}
```

Listing 4.1: Example: Ausgabe der Methode checkAccesspointExists

insertNewAccesspoint(Array DataAccesspoint)

Die Methode „insertNewAccesspoint(Array DataAccesspoint)“ legt in der Datenbank einen neuen Accesspoint an. Als Übergabeparameter dient ein Array, welches folgendermaßen aufgebaut ist:

```
{
    "b4:07:f9:71:5b:6e": {
        "ssid": "AndroidAP",
        "mac": "b4:07:f9:71:5b:6e"
    }
}
```

Listing 4.2: Example: Übergabeparameter, um neuen Accesspoint einzutragen

Wenn das Einfügen des Datensatzes in die Tabelle erfolgreich war, speichert die Methode in die Variable *result* den Rückgabewert aus der Datenbank.

4.1.4. Referenzpunkte

Der Dienst „Referenzpunkte“ beinhaltet alle notwendigen Funktionen, die die Tabelle „referencepoints“ betreffen. In der Tabelle „referencepoints“ sind die Eigenschaften der Referenzpunkte wie Name, Beschreibung, Koordinaten und Status gespeichert. In der Tabelle 4.2 ist eine Zusammenfassung der Methoden, Konstruktoren und Eigenschaften der Klasse *Referenzpunkte* zu finden.

Tabelle 4.2.: Zusammenfassung - Referenzpunkte

Konstruktoren	
	_construct(Object _db)
Eigenschaften	
Array	result
Object	db
Public Methoden	
Array	getResult
Array	getResult
boolean	getNeueReferenzpunkte
boolean	getAllReferenzpunkte
boolean	getReferenzpunktByID(Integer id)
boolean	getReferenzpunktByName(String name)
boolean	getReferenzpunktDetails(Integer id)

4.1.5. Konstruktoren von Referenzpunkten

_construct(Object _db)

„_db“ ist eine Konstruktor-Methode, der die Datenbankverbindung erwartet. Damit ist gewährleistet, dass diese Klasse eine Verbindung zur Datenbank herstellen kann.

4.1.6. Eigenschaften von Referenzpunkten

result

In dieser Variablen werden Ergebnisse der einzelnen Methoden gespeichert. Diese Variable ist eine private Variable und lässt sich mit der Getter-Funktion getResult auslesen.

db

In diesem Objekt ist die aktuelle Datenbankverbindung gespeichert. Dieses Objekt ist notwendig, wenn ein Datenbankaufruf in einer Methode benötigt wird.

4.1.7. Methoden von Referenzpunkten

getResult

Mit Hilfe dieser Methode kann man die privaten Variable „result“ auslesen.

getNeueReferenzpunkte

Die Methode „getNeueReferenzpunkte“ holt die Referenzpunkte aus der Datenbank, die noch keine Messwerte in der Tabelle *Measurements* vorweisen. Wenn Werte vorhanden sind, werden diese in *result* gespeichert und ein Boolscher-Wert zurückgegeben, dass die Datenbankabfrage erfolgreich war.

Example - getNeueReferenzpunkte

Der Aufruf für die Abfrage der neuen Referenzpunkte sieht wie folgt aus:
`http://www.inav.tugraz.at:8080/service/dispatcher.php?$dienst=referenzpunkte&$operation=getNeueReferenzpunkte`
Die Antwort vom Webservice sieht folgendermaßen aus:

```
{      "getNeueReferenzpunkte": [{
    "id": 278,
    "0": 278,
```

```
4     "name": "Test",
5     "1": "Test",
6     "beschreibung": "Test",
7     "2": "Test",
8     "koordinaten": "01010000000B5D8940F5E72E40DA1EBDE13E884
        740",
9     "3": "01010000000B5D8940F5E72E40DA1EBDE13E884740",
10    "created": "2013-08-05 13:13:11",
11    "4": "2013-08-05 13:13:11",
12    "modified": "2013-08-05 13:13:11",
13    "5": "2013-08-05 13:13:11"
14  }],
15  "status": "success"}
```

Listing 4.3: Example: Ausgabe der Methode getNeueReferenzpunkte

getAllReferenzpunkte

Die Methode „getAllReferenzpunkte“ holt alle Referenzpunkte aus der Datenbank. Man bekommt ein Array mit allen Attributen der Referenzpunkte in der Variablen result, sofern der Aufruf erfolgreich war, zurückgeliefert.

Example - getAllReferenzpunkte

Der Aufruf für die Abfrage aller Referenzpunkt sieht wie folgt aus:
`http://www.inav.tugraz.at:8080/service/dispatcher.php?$dienst=referenzpunkte&$operation=getAllReferenzpunkte`

Die Antwort vom Webservice sieht folgendermaßen aus:

```
1  {
2        "getAllReferenzpunkte": [{
3     "id": 278,
4     "0": 278,
5     "name": "Test",
6     "1": "Test",
7     "beschreibung": "Test",
8     "2": "Test",
```

```
    "koordinaten": "01010000000B5D8940F5E72E40DA1EBDE13E884
       740",
    "3": "01010000000B5D8940F5E72E40DA1EBDE13E884740",
    "created": "2013-08-05 13:13:11",
    "4": "2013-08-05 13:13:11",
    "modified": "2013-08-05 13:13:11",
    "5": "2013-08-05 13:13:11"
  }],
  "status": "success"
}
```

Listing 4.4: Example: Ausgabe der Methode getAllReferenzpunkte

getReferenzpunktByID(Integer id)

Die Methode „getReferenzpunktByID(Integer id)“ holt den Referenzpunkt mit der übergebenen *ID* aus der Datenbank. Man bekommt ein Array mit allen Attributen des Referenzpunktes in der Variablen „result“, wenn der Aufruf erfolgreich war, zurückgeliefert.

Example - getReferenzpunktByID

Der Aufruf für die Abfrage aller Referenzpunkte sieht wie folgt aus:

`http://www.inav.tugraz.at:8080/service/dispatcher.php?$dienst=referenzpunkte&$operation=getReferenzpunktByID&id=278`

Die Antwort vom Webservice sieht folgendermaßen aus:

```
{
    "getReferenzpunktByID":[{
        "id":278,
        "0":278,
        "name":"Test",
        "1":"Test",
        "beschreibung":"Test",
        "2":"Test",
        "koordinaten":"01010000000B5D8940F5E72E40DA1EBDE13E
           884740",
        "3":"01010000000B5D8940F5E72E40DA1EBDE13E884740",
```

```
11          "created":"2013-08-05 13:13:11",
12          "4":"2013-08-05 13:13:11",
13          "modified":"2013-08-05 13:13:11",
14          "5":"2013-08-05 13:13:11"}],
15          "status":"success"
16  }
```

Listing 4.5: Example: Ausgabe der Methode getReferenzpunktByID

getReferenzpunktByName(String name)

Die Methode „getReferenzpunktByName(String name)“ holt den Referenzpunkt mit dem übergebenen *Namen* aus der Datenbank. Man bekommt ein Array mit allen Attributen des Referenzpunktes in der Variablen „result“, wenn der Aufruf erfolgreich war, zurückgeliefert.

Example - getReferenzpunktByName

Der Aufruf für die Abfrage aller Referenzpunkte sieht wie folgt aus:

`http://www.inav.tugraz.at:8080/service/dispatcher.php?$dienst=referenzpunkte&$operation=getReferenzpunktByName&name=Test`

Die Antwort vom Webservice sieht folgendermaßen aus:

```
1   {
2       "getReferenzpunktByName":[{
3           "id":278,
4           "0":278,
5           "name":"Test",
6           "1":"Test",
7           "beschreibung":"Test",
8           "2":"Test",
9           "koordinaten":"01010000000B5D8940F5E72E40DA1EBDE13E
                884740",
10          "3":"01010000000B5D8940F5E72E40DA1EBDE13E884740",
11          "created":"2013-08-05 13:13:11",
12          "4":"2013-08-05 13:13:11",
13          "modified":"2013-08-05 13:13:11",
```

```
14          "5":"2013-08-05 13:13:11"}],
15          "status":"success"
16  }
```

Listing 4.6: Example: Ausgabe der Methode getReferenzpunktByName

getReferenzpunktDetails(Integer id)

Die Methode „getReferenzpunktDetails(Integer id)“ holt für den Referenzpunkt mit der übergebenen *ID* alle Attribute der Tabelle „referenzpunkte“ und die dazugehörigen Messwerte, sofern diese existieren, aus der Tabelle „measurements“. Dies geschieht mit Hilfe der „getMesurementsByReferencepoint“ Methode, die auf die Datenbank zugreift. Man bekommt ein Array mit allen Attributen des Referenzpunktes in der Variable „result“, wenn der Aufruf erfolgreich war, zurückgeliefert.

Example - getReferenzpunktDetails

Der Aufruf für die Abfrage aller Referenzpunkte sieht wie folgt aus:
`http://www.inav.tugraz.at:8080/service/dispatcher.php?$dienst=referenzpunkte&$operation=getReferenzpunktDetails&id=278`

Die Antwort vom Webservice sieht folgendermaßen aus:

```
1  {
2   "getReferenzpunktDetails": [{
3       "id": 278,
4       "0": 278,
5       "name": "Test",
6       "1": "Test",
7       "beschreibung": "Test",
8       "2": "Test",
9       "koordinaten": "0101000000072E40DA1EBDE13E884740",
10      "3": "0101000000072E40DA1EBDE13E884740",
11      "created": "2013-08-05 13:13:11",
12      "4": "2013-08-05 13:13:11",
13      "modified": "2013-08-05 13:13:11",
14      "5": "2013-08-05 13:13:11",
15      "mesurements": {
16          "b4:a4:e3:59:49:e2": {
```

```
            "id": 46,
            "0": "667",
            "accesspoint_id": 46,
            "1": 46,
            "referencepoint_id": 278,
            "2": 278,
            "rssi": "-86",
            "3": "-86",
            "channel": 2412,
            "4": 2412,
            "created": "2013-08-12 16:40:56.861",
            "5": "2013-08-12 16:40:56.861",
            "modified": "2013-08-12 16:40:56.861",
            "6": "2013-08-12 16:40:56.861",
            "status_id": 1,
            "7": 1,
            "8": 46,
            "mac": "b4:a4:e3:59:49:e2",
            "9": "b4:a4:e3:59:49:e2",
            "ssid": "POI",
            "10": "POI"
        },
          "a0:cf:5b:9f:76:3f": {
              "id": 76,
              "0": "657",
              "accesspoint_id": 76,
              "1": 76,
              "referencepoint_id": 278,
              "2": 278,
              "rssi": "-62.904220779221",
              "3": "-62.904220779221",
              "channel": 5180,
              "4": 5180,
              "created": "2013-08-12 16:40:56.707",
              "5": "2013-08-12 16:40:56.707",
              "modified": "2013-08-12 16:40:56.707",
```

```
53                "6": "2013-08-12 16:40:56.707",
54                "status_id": 1,
55                "7": 1,
56                "8": 76,
57                "mac": "a0:cf:5b:9f:76:3f",
58                "9": "a0:cf:5b:9f:76:3f",
59                "ssid": "tug",
60                "10": "tug"
61              },
62            "a0:cf:5b:9f:76:3e": {
63                "id": 79,
64                "0": "663",
65                "accesspoint_id": 79,
66                "1": 79,
67                "referencepoint_id": 278,
68                "2": 278,
69                "rssi": "-62.621212121212",
70                "3": "-62.621212121212",
71                "channel": 5180,
72                "4": 5180,
73                "created": "2013-08-12 16:40:56.806",
74                "5": "2013-08-12 16:40:56.806",
75                "modified": "2013-08-12 16:40:56.806",
76                "6": "2013-08-12 16:40:56.806",
77                "status_id": 1,
78                "7": 1,
79                "8": 79,
80                "mac": "a0:cf:5b:9f:76:3e",
81                "9": "a0:cf:5b:9f:76:3e",
82                "ssid": "eduroam",
83                "10": "eduroam"
84            },
85    }
86  }],
87        "status":"success"}
```

Listing 4.7: Example: Ausgabe der Methode „getReferenzpunktDetails“

4.1.8. Messung

Der Dienst „Messung“ beinhaltet alle notwendigen Funktionen, die die Tabelle „measurements“ betreffen. In der Tabelle „measurements“ werden die unterschiedlichen Messwerte von einem Referenzpunkt zu den verfügbaren Accesspoints eingetragen. Weiters sind die Accespoints ID, Referencepoint ID, der gemessene RSSI Wert, der Kanal und der aktuelle Status der Messung in dieser Tabelle gespeichert. Die Tabelle 4.3 zeigt eine Zusammenfassung der Methoden, Konstruktoren und Eigenschaften der Klasse *Messung*.

Tabelle 4.3.: Zusammenfassung - Messung

Public Methoden	
boolean	getMesurementsByReferencepoint(integer referenzpunktId)
boolean	saveNewMesurements(Array messwerte)

4.1.9. Methoden von Messung

getMesurementsByReferencepoint(integer referenzpunktId)

Die Methode „getMesurementsByReferencepoint(integer referenzpunktId)“ holt alle Messungen aus der Datenbank, die in den vorliegenden Referenzpunkten getätigt wurden. Folgende Werte werden, sofern Werte vorhanden sind, in die Variable „result“ gespeichert und anschließend wird ein „Boolscher Wert“ zurückgegeben, um festzustellen, ob die Datenbankabfrage erfolgreich war.

Example - getMesurementsByReferencepoint

Der Aufruf für die Abfrage eines Referenzpunktes sieht wie folgt aus:

```
http://www.inav.tugraz.at:8080/service/dispatcher.php?$dienst=messung
&$operation=getMesurementsByReferencepoint&referenzpunktId=278
```

Die Antwort vom Webservice sieht folgendermaßen aus:


```
{
  "getMesurementsByReferencepoint": {
    "b4:a4:e3:59:49:e2": {
        "id": 46,
        "0": "667",
        "accesspoint_id": 46,
        "1": 46,
        "referencepoint_id": 278,
        "2": 278,
        "rssi": "-86",
        "3": "-86",
        "channel": 2412,
        "4": 2412,
        "created": "2013-08-12 16:40:56.861",
        "5": "2013-08-12 16:40:56.861",
        "modified": "2013-08-12 16:40:56.861",
        "6": "2013-08-12 16:40:56.861",
        "status_id": 1,
        "7": 1,
        "8": 46,
        "mac": "b4:a4:e3:59:49:e2",
        "9": "b4:a4:e3:59:49:e2",
        "ssid": "POI",
        "10": "POI"
    },
    "a0:cf:5b:9f:76:3f": {
        "id": 76,
        "0": "657",
        "accesspoint_id": 76,
        "1": 76,
        "referencepoint_id": 278,
        "2": 278,
        "rssi": "-62.904220779221",
        "3": "-62.904220779221",
        "channel": 5180,
```

```
        "4": 5180,
        "created": "2013-08-12 16:40:56.707",
        "5": "2013-08-12 16:40:56.707",
        "modified": "2013-08-12 16:40:56.707",
        "6": "2013-08-12 16:40:56.707",
        "status_id": 1,
        "7": 1,
        "8": 76,
        "mac": "a0:cf:5b:9f:76:3f",
        "9": "a0:cf:5b:9f:76:3f",
        "ssid": "tug",
        "10": "tug"
    },
    "a0:cf:5b:9f:76:3e": {
        "id": 79,
        "0": "663",
        "accesspoint_id": 79,
        "1": 79,
        "referencepoint_id": 278,
        "2": 278,
        "rssi": "-62.621212121212",
        "3": "-62.621212121212",
        "channel": 5180,
        "4": 5180,
        "created": "2013-08-12 16:40:56.806",
        "5": "2013-08-12 16:40:56.806",
        "modified": "2013-08-12 16:40:56.806",
        "6": "2013-08-12 16:40:56.806",
        "status_id": 1,
        "7": 1,
        "8": 79,
        "mac": "a0:cf:5b:9f:76:3e",
        "9": "a0:cf:5b:9f:76:3e",
        "ssid": "eduroam",
        "10": "eduroam"
    },
```

```
72    },
73    "status": "success"
74  }
```

Listing 4.8: Example: Ausgabe der Methode getMesurementsByReferencepoint

saveNewMesurements(Array messwerte)

Die Methode „saveNewMesurements(Array messwerte)“ dient zur Vorbereitung und Speicherung der neuen Messungen in der Datenbank. Als Erstes werden die gelieferten Messwerte auf Fehler geprüft. Anschließend werden die Messwerte in die Datenbank eingetragen. Bei erfolgreicher Speicherung wird ein „Boolscher-Wert“ zurückgegeben.

Der Aufruf für die Speicherung der Messung muss wie folgt aufgebaut sein:

```
1  {
2   "messwerte": [
3    {
4     "mesurements": {
5       "00:24:17:7a:80:4b": {
6         "bssid": "00:24:17:7a:80:4b",
7         "scanned_time": "19:37:15:68",
8         "orientierung": {
9           "west": {
10            "rssi": -88
11          },
12          "nord": {
13            "rssi": -88
14          },
15          "sued": {
16            "rssi": -89
17          }},
18        "scanned_date": "2013-08-12",
19        "ssid": "Thomson402720",
20        "mac": "00:24:17:7a:80:4b",
21        "rssi": -88.75,
22        "channel": "2462"
23      },
```

```
    "00:1f:3f:a3:ed:47": {
      "bssid": "00:1f:3f:a3:ed:47",
      "scanned_time": "19:37:10:803",
      "orientierung": {
        "ost": {
          "rssi": -66.5
        },
        "west": {
          "rssi": -66.2
        },
        "nord": {
          "rssi": -67.6
        },
        "sued": {
          "rssi": -66
        }},
      "scanned_date": "2013-08-12",
      "ssid": "Nachtigal",
      "mac": "00:1f:3f:a3:ed:47",
      "rssi": -66.57499999999999,
      "channel": "2412"
    }
   },
    "id": 278,
    "0": 278,
    "name": "Test",
    "1": "Test",
    "beschreibung": "Test",
    "2": "Test",
    "koordinaten": "01010000000B5D8940F5E72E40DA1EBDE13E88
       4740",
    "3": "01010000000B5D8940F5E72E40DA1EBDE13E884740",
    "created": "2013-08-05 13:13:11",
    "5": "2013-08-05 13:13:11",
    "modified": "2013-08-05 13:13:11",
    "4": "2013-08-05 13:13:11"
```

```
59    }]
60 }
```

Listing 4.9: Example: Übergabeparameter für saveNewMesurements

Wenn in einer Richtung keine Messwerte zu einem der Accespoints existieren, werden diese ausgelassen.

Die Antwort vom Webservice sieht folgendermaßen aus:

```
1  {
2   "saveNewMesurements": {
3    "00:1f:3f:a3:ed:47": {
4        "id": "669",
5        "created": "2013-08-12 19:37:20.758",
6        "status_id": 1,
7        "referencepoint_id": 278,
8        "rssi": "-66.575",
9        "accesspoint_id": 35,
10       "channel": 2412,
11       "modified": "2013-08-12 19:37:20.758"
12   },
13   "00:24:17:7a:80:4b": {
14     "id": "668",
15     "created": "2013-08-12 19:37:20.73",
16     "status_id": 1,
17     "referencepoint_id": 278,
18     "rssi": "-88.75",
19     "accesspoint_id": 34,
20     "channel": 2462,
21     "modified": "2013-08-12 19:37:20.73"
22   }
23  },
24    "status": "success"
25 }
```

Listing 4.10: Example: Ausgabe der Methode saveNewMesurements

4.1.10. Richtungssatzmessungen

Der Dienst namens „Orientierung“ beinhaltet alle notwendigen Funktionen, welche die Tabelle „orientations“ betreffen. In der Tabelle „orientations“ werden für jeden Referenzpunkt die gemessenen Signalstärken von allen verfügbaren Accesspoints für alle vier Himmelsrichtungen eingetragen. Zusätzlich werden die IDs der Accesspoints, und des Referencepoints, der Kanal und der aktuelle Status der Messung in dieser Tabelle gespeichert. Die Tabelle 4.4 zeigt eine Zusammenfassung der Methoden der Klasse *Orientierung*. Bei diesem Dienst geht es darum, die Orientierungsparameter in der Datenbank speichern, auslesen und verändern zu können. Die Orientierungsparameter werden bei der Messung ermittelt und beim Speichern der Messwerte in der Datenbank eingetragen.

Tabelle 4.4.: Zusammenfassung - Orientierung

Public Methoden	
boolean	getOrientationsByMesurementID(int mesurementId)
boolean	getAllCourses

getOrientationsByMesurementID(int mesurementId)

Die Methode „getOrientationsByMesurementID(int mesurementId)“ holt aus der Datenbank die gemessenen Werte der Orientierung für die jeweilige Messung. Die Messung bezieht sich meistens von einem Referenzpunkt zum Accespoint. Als Übergabeparameter dient die ID der Messung. Der Aufruf sieht im Beispiel wie folgt aus:
`http://www.inav.tugraz.at:8080/service/dispatcher.php?$dienst=orientations&$operation=getOrientationsByMesurementID&mesurementId=3292`
und die Antwort vom Webservice ist:

```
{"getOrientationsByMesurementID":
    {"west":
        {"id":10516,
        "rssi":"-91.65",
        "created":"2013-08-23 12:46:14.925",
        "modified":"2013-08-23 12:46:14.925",
        "mesurement_id":3292,
        "course_id":4,
        "name":"west"
        ,"status_id":1},
```

```
11      "nord":{
12          "id":10517,
13          "rssi":"-68",
14          "created":"2013-08-23 12:46:14.925",
15          "modified":"2013-08-23 12:46:14.925",
16          "mesurement_id":3292,
17          "course_id":1,
18          "name":"nord",
19          "status_id":1},
20      "sued":{
21          "id":10518,
22          "rssi":"-89.6",
23          "created":"2013-08-23 12:46:14.94",
24          "modified":"2013-08-23 12:46:14.94",
25          "mesurement_id":3292,
26          "course_id":3,
27          "name":"sued",
28          "status_id":1,}},
29      "status":"success"}
```

Listing 4.11: Example: Aufbau des Arrays für das Überprüfen der Messung

getAllCourses

Diese Methode holt alle möglichen Orientierungen aus der Datenbank.

```
1 {"getAllCourses":
2     [{"id":1,"name":"nord"},
3     {"id":2,"name":"ost"},
4     {"id":3,"name":"sued"},
5     {"id":4,"name":"west"}],
6 "status":"success"}
```

Listing 4.12: Example: Alle verfügbaren Orientierungen

4.1.11. Positionierung

Der Dienst „Positionierung“ ist eine Kernfunktion der Masterarbeit. In dieser Funktion werden die gemessenen Signalstärken mit den in der Datenbank gespeicherten Signalstär-

ken verglichen und dadurch eine Position bestimmt. In der nachstehenden Tabelle sind Methoden für den Aufruf aus dem Internet angeführt.

Tabelle 4.5.: Zusammenfassung - Positionierung

Public Methoden	
Array	positionBerechnen(Array messung)
Array	getKoordinateforReferencepointById(Int id)

positionBerechnen(Array messung)

Diese Methode berechnet aus den verfügbaren RSSI Werten und den MAC Adressen die aktuelle Position. Die Anfrage sieht wie gefolgt aus.

```
{
   "$operation": "positionBerechnen",
   "$dienst": "position",
   "params": {
     "messung": [
       [
         {
           "bssid": "00:1f:3f:a3:ed:47",
           "orient": "default",
           "scanned_time": "21:58:24:1",
           "scanned_date": "2014-02-06",
           "ssid": "Nachtigal",
           "mac": "00:1f:3f:a3:ed:47",
           "rssi": -53,
           "channel": 2412
         }
       ],
       [
         {
           "bssid": "a4:b1:e9:68:82:59",
           "orient": "default",
           "scanned_time": "21:58:24:1",
           "scanned_date": "2014-02-06",
           "ssid": "A1-688259",
           "mac": "a4:b1:e9:68:82:59",
```

```
            "rssi": -83,
            "channel": 2462
          }
        ]
      ]
    },
    "format": "json",
    "header": {
      "deviceVersion": 18,
      "deviceType": "Android",
      "language": "Deutsch"
    }
  }
```

Listing 4.13: Example: Position berechnen JSON Anfrage

Die Antwort des Webservice beinhaltet die Koordinaten der berechneten Referenzpunkte, sowie deren ID und Namen. Weiters wird bei der Positionsberechnung eine Standardabweichung berechnet. Dieser Wert wird ebenfalls zurückgegeben. Die Antwort des Webservices findet man in der Listing positionBerechnen(Array messung).

```
{
  "positionBerechnen": {
    "refpunkt_id": 278,
    "refpunkt_name": "Test",
    "ssr": 37.7188291102,
    "messwerte": 0,
    "koordinaten": "POINT(15.453043 47.064419)",
    "lat": "47.064419",
    "lng": "15.453043"
  },
  "status": "success"
}
```

Listing 4.14: Example: Position berechnen JSON Anfrage

getKoordinateforReferencepointById(Int id)

Mit dieser Methode kann man die Koordinaten eines Referenzpunktes aus der Datenbank abfragen. Benötigt wird hierfür die ID des Referenzpunktes, die bei der Abfrage übergeben

wird. Als Antwort erhält man ein JSON Array. Dieser sieht wie folgt aus:

```
{
"getKoordinateforReferencepointById": [{
  "id": 278,
  "koordinaten": "POINT(15.453043 47.064419)",
  "lng": "15.453043",
  "lat": "47.064419",
  "name": "Test"
}],
"status": "success"
}
```

Listing 4.15: Example: Koordinaten eines Referenzpunktes

5. Einrichten des Webservers

Um das Webservice betreiben zu können und für weitere Tests für die Weiterentwicklung der Masterarbeit benutzen zu können, wurde eigens ein Webserver an der Technischen Universität Graz eingerichtet, der genau für diesen Aufgabenbereich zuständig ist. Die Konfiguration und Einrichtung des Servers ist ein Teil der Masterarbeit. (Anforderungen siehe 3.1.4) Folgende Überlegungen wurden getroffen und anschließend aufgebaut:

5.1. Überlegungen

Als Erstes wurde überlegt, welche Software auf dem Server laufen soll und welche Leistung dieser haben soll. Dazu wurde Robert Mayr von der ITG Graz, der für mehrere Webserver für den „Magistrat Graz“ zuständig ist, befragt, um seine Erfahrungswerte aus der Praxis nutzen zu können. An dieser Stelle nochmals herzlichen Dank für die Unterstützung. Als Server steht ein *Intel(R) CORE(TM)2 DUO CPU E4500 mit 2.20GHz mit 2Gb RAM* zur Verfügung. Die Anbindung an das Internet ist über das TU Graz Netzwerk gewährleistet. Hier steht eine *100Mbit* Leitung für Upload und Download zu Verfügung. Im Server ist eine *500GB* große Festplatte verbaut. Das Betriebssystem des Servers ist *Windows 7 64Bit Enterprise*.
Der Webserver ist ein Computer mit einer Webserver-Software, die dazu dient, Dokumente an Clients (Browser) zu übertragen.

5.2. Installation

Als Nächstes folgen die Installationsparameter der verwendeten Software, im speziellen für den Webserver und die Datenbank.

5.2.1. HTTP Server

Als HTTP Server dient der Apache 2.2 Webserver. Dieser ist ein „Quell-offenes“ und wird von *Apache Software Foundation* entwickelt und ist der meistbenutzte Webserver im Internet [25] [26].
Der Server wird unter *Apache License 2.0* vertrieben, die eine Lizenz für freie Software ist und mit der *GNU General Public License* kompatibel ist [27] . Siehe `http://www.gnu.`

`org/licenses/license-list.html#apache2`.

Installation Apache

Auf der Projektseite `http://httpd.apache.org/download.cgi` kann die letzte Version heruntergeladen werden. Die Installation wird mit Hilfe des Installationsassistenten gestartet.
Nach Akzeptieren der Lizenzbedingungen kommt man auf die Seite, wo man die Serverinformationen einträgt. Zuerst müssen ein Domainname, ein Servername und eine E-Mail-Adresse für den Administrator angegeben werden. Wenn man den Apache in einem lokalen Netz zu Testzwecken einrichten möchte, kann man *localhost* und eine beliebige E-Mail-Adresse eintragen. In Abbildung 5.1 sieht man die für den Server eingetragenen Werte.

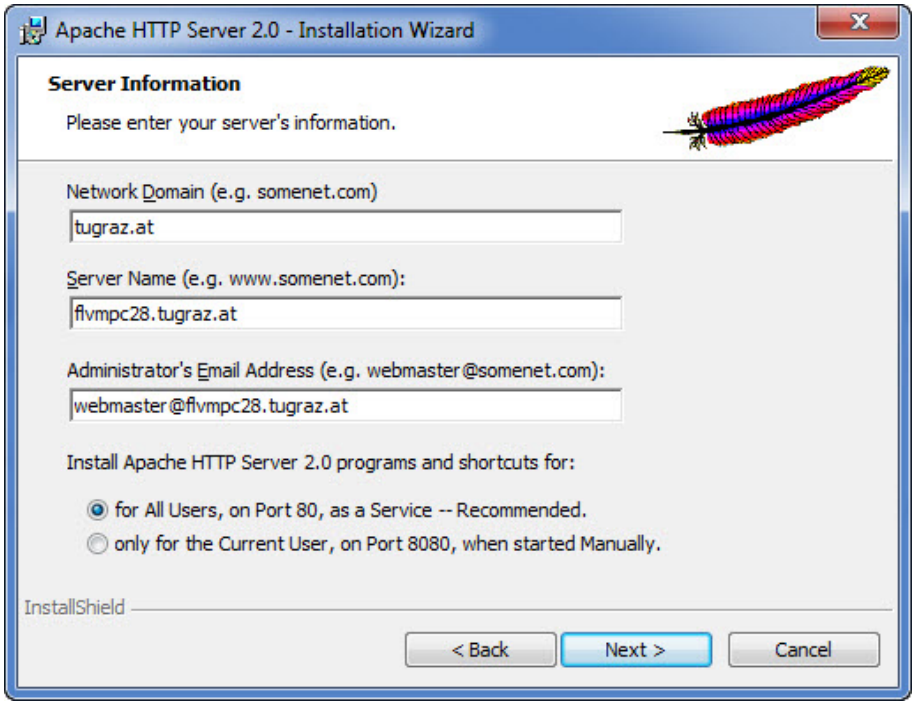


Abbildung 5.1.: Apache - Server Information

Es wird ebenfalls ausgewählt, dass Apache als „Dienst“ für alle Benutzer läuft. Als nächstes wählt man aus, ob eine typische Installation oder eine benutzerdefinierte Installation erfolgen soll. Mit der benutzerdefinierten Installation kann ausgewählt werden, welche Softwarebestandteile man installieren möchte oder nicht. Im Beispiel wurde eine typische Installation gewählt. Abschließend wird der Ort der Installation festgelegt. Standardmäßig ist es unter „C:\Programme Files (X86)\Apache Software Foundation“

Die Schritte laufen auf den verschiedenen Windows Versionen gleich ab. Wenn die Installation erfolgreich ist, meldet sich der Assistent nochmals, dass der Vorgang korrekt abgeschlossen wurde. Ein Neustart des Systems ist nicht notwendig. Abschließend wird der Webserver automatisch gestartet. Der Apache läuft als Hintergrunddienst auf dem Rechner und kann in einem Browser unter `http://localhost` oder `http://127.0.0.1` aufgerufen werden. Es wird ein Standarddokument geöffnet. Die Dokumente, die der Webserver anzeigen soll, befinden sich im Unterordner der Installation *„htdocs“*.

Konfiguration Apache Webserver

Es bedarf der Konfiguration des Webservers, um produktiv arbeiten zu können und den Webserver auch aus dem Internet ansprechen zu können. Da im Internetnetz der Technischen Universität Graz bestimmte Ports gesperrt sind, war es notwendig, den Port vom „Zentralen Informationsdienst“ freigeben zu lassen. Dies ist der Port 8080. Die Konfigurationsdatei des Webservers befindet sich in der „httpd.conf“ Datei im Unterordner „conf“ der Apache Installation.
Apache ist modular aufgebaut, so können in der Konfigurationsdatei Module freigeschalten werden, die sich im Ordner „modules“ befinden.

- Port - Firewall Einstellungen
- Module, die freigegeben wurden

PHP

Damit nicht nur die statischen Inhalte dargestellt werden, wird Skirptsprache „PHP“ installiert (zum Downloaden unter `http://php.net/downloads.php`). Mit „PHP“ lassen sich dynamische Webseiten oder Webanwedungen erstellen. „PHP“ wird als freie Software („Open Source“) unter der PHP-Lizenz vertrieben (zu finden unter `http://www.php.net/license/`). In der aktuellen Version „PHP 5“ wird ein verbessertes Objektmodell für die objektorientierte Programmierung und erweiterte Funktionalitäten bei XML und DOM-Handhabung geboten. „PHP“ bietet eine breite Datenbankunterstützung, unter anderem auch für die im Projekt ausgewählte Datenbank „PostgreSQL“. Es müssen für den Webserver „Apache“ und für die Datenbank einige Konfigurationsschritte getätigt werden, bevor diese miteinander interagieren können.
Eine detaillierte Installationsanleitung für „PHP“ für unterschiedliche Webserver und Betriebsysteme ist unter `http://php.net/manual/de/install.php` zu finden.
Für den Apache 2.2 Server auf der Windows Plattform sind die Schritte nachfolgend erklärt. Der unten angeführte Konfigurationscode muss unter „httpd.conf“ eingetragen

werden.

```
1     LoadModule php5_module "c:/php/php5apache2_2.dll"
2     AddHandler application/x-httpd-php .php

4     # configure the path to php.ini
5     PHPIniDir "C:/php"

7     <FilesMatch \.php\$>
8       SetHandler application/x-httpd-php
9     </FilesMatch>
```

Listing 5.1: PHP und Apache 2.2 Konfiguration

5.2.2. Datenbank

Als Datenbank dient „PostgreSQL“, welche eine leistungsstarke, objekt-relationale Datenbank ist. Sie ist unter einer „BSD-artigen Open-Source Lizenz“ veröffentlicht.
„PostgreSQL“ bietet eine Erweiterung, die „POSTGis“ genannt wird. Diese Erweiterung unterstützt geographische Objekte und Funktionen. Diese Unterstützung war bei der Anforderungsanalyse erforderlich. Nachfolgend die Installation von „PostgreSQL“ mit der „POSTGis Erweiterung“ unter Windows:

Installation von PostgreSQL unter Microsoft Windows

Für Microsoft Windows sind auf der PostgreSQL-Projektseite die neuesten Installationsdateien zum Herunterladen vorhanden (`http://www.postgresql.org/download/windows/`).
Die Installation unter Microsoft Windows erfordert einige Voraussetzungen:

- Administrationsrechte auf dem lokalen Rechner
- alle Updates und Servicepakete müssen installiert sein
- PostgreSQL benötigt für die Installation eine NTFS formatierte Partition

Installationsschritte:
Nach dem Starten der Installation mit *postgresql-<Versionsnummer>.msi* öffnet sich der Installationsassistent und fragt die wichtigsten Voreinstellungen ab. Dieser Vorgang ist in Abblidung 5.2 ersichtlich.

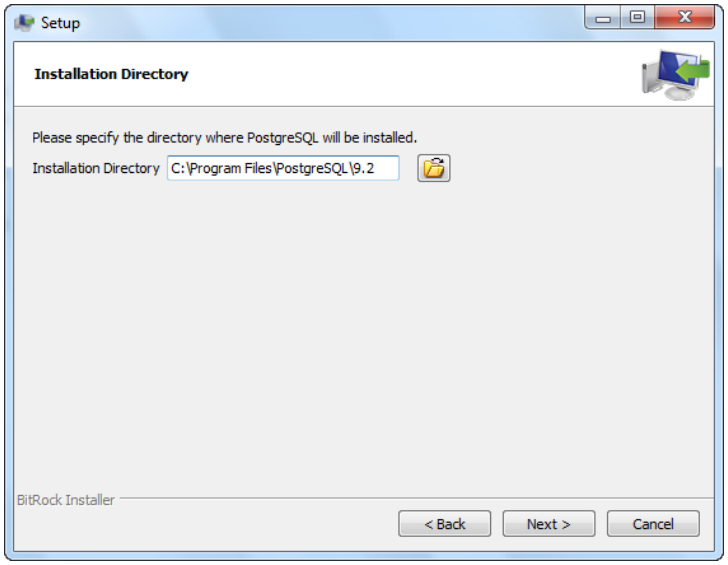


Abbildung 5.2.: PostgreSQL Installationsordner

In Abbildung 5.3 wird der Speicherort für die Daten der Datenbank „PostgreSQL“ festgelegt.

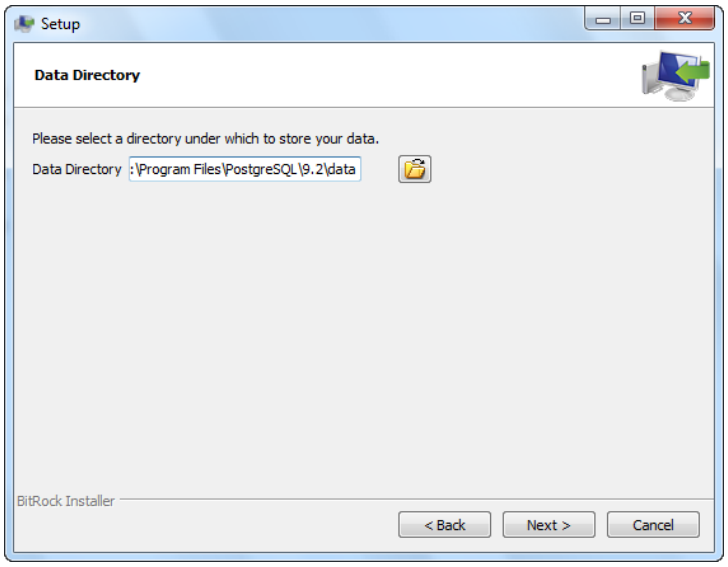


Abbildung 5.3.: PostgreSQL Datenbank Speicherordner

In Abbildung 5.4 sieht man die Installationsoptionen sowie weitere zu installierende Software.

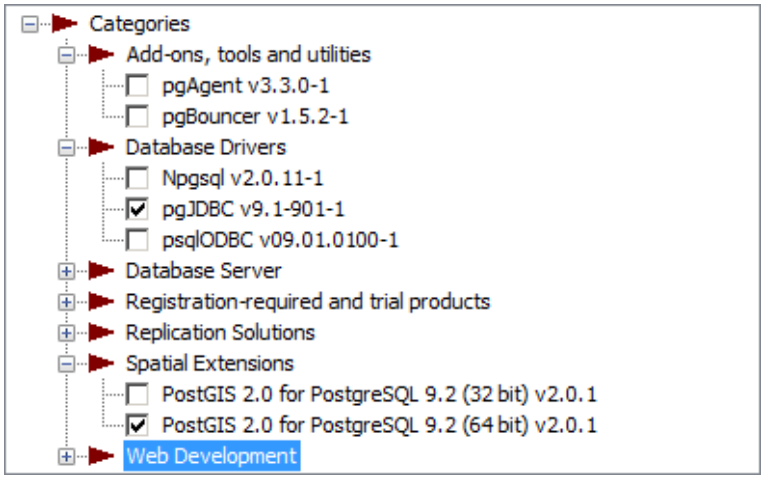


Abbildung 5.4.: PostgreSQL Installation von Erweiterung

5.2.3. Installation PostGIS

Zu Beginn wird das Paket „PostGIS“ von `http://postgis.net/install` heruntergeladen. Anschließend wird die Installation gestartet. Am besten verwendet man durchgehend den Installationsassistenten.
In Abbildung 5.5 wird die Komponente „PostGIS“ und „Create spatial database“ ausgewählt und die Installation fortgesetzt.

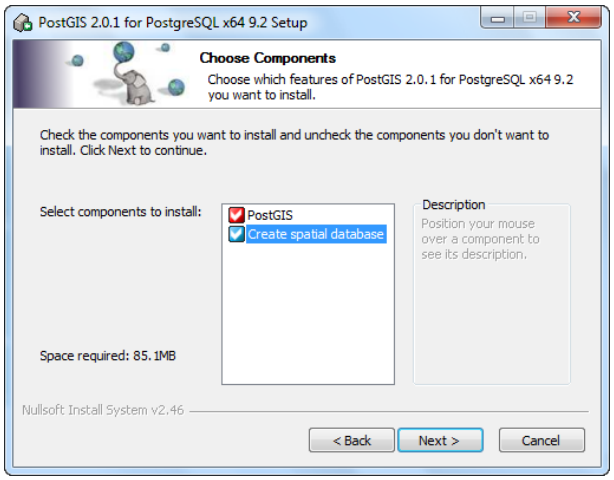


Abbildung 5.5.: Auswahl „spatial database“

Im nächsten Dialogfenster wird das Standardverzeichnis unter „PostgreSQL“ aufgerufen und in weiterer Folge werden die Zugangsdaten des Datenbankservers eingegeben. Diesen Vorgang sieht man in Abbildung 5.6.

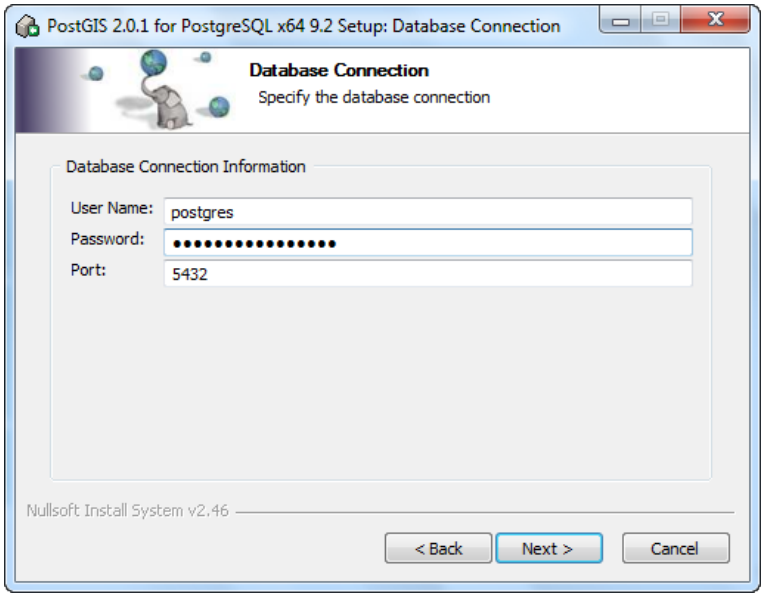


Abbildung 5.6.: PostGIS - Datenbankverbindungsdaten

Nach Eingabe der Standarddatenbank für „PostGIS“ ist die Installation abgeschlossen. Bei diesem Projekt lautet der Name der Datenbank „indoor“.

PostgreSQL konfigurieren

Um die Datenbank optimal nutzen zu können, werden in der Konfigurationsdatei „Postgresql.conf“ weitere Einstellungen vorgenommen, da die Standardinstallation von „PostgreSQL“ nicht für Anwendungen an einem Produktivsystem geeignet ist. In der nachfolgenden Aufzählung befinden sich die wichtigsten Parameter, die geändert werden können.

- *port* - Der „TCP Port“ der Datenbank ist für die Netzwerkzugriffe (standardmäßig 5432) verantwortlich. Der Port muss freigegeben sein, damit aus dem Internet zugegriffen werden kann.
- *max_connections* - Hier wird die maximale Anzahl an Verbindungen, die gleichzeitig auf die Datenbank zugreifen dürfen, angegeben.
- *superuser_reserved_connections* - Für den Administrator werden eine bestimmte Anzahl an Zugriffen auf die Datenbank festgelegt.
- *shared_buffers* - Dieser Wert steht für alle „PostgreSQL“ Prozesse zur Verfügung. Für einen Datenbankserver ist der Wert von einem Viertel bis zu einem Drittel des verfügbaren Hauptspeichers empfehlenswert. Bei Systemen mit hohen Lesezugriffen kann ein kleinerer Wert definiert werden. Im Gegensatz dazu kann bei vielen

Schreibzugriffen ein höherer Wert festgelegt werden. Auf keinen Fall darf hier der gesamte Speicher des Servers konfiguriert werden.

- *work_mem* - Mit dieser Einstellung wird die Obergrenze des zur Verfügung stehenden Arbeitsspeichers für diverse Operationen, wie Sortieroperationen oder Verknüpfungsalgorithmen festgelegt. Über den festgelegten Wert werden solche Operationen auf der langsameren Festplatte in eine temporäre Datei zwischengespeichert. Je höher dieser Wert ist, desto mehr Operationen können im RAM durchgeführt werden.
- *maintenance_work_mem* - Diese Eigenschaft legt die Speichergröße für Wartungsarbeiten an der Datenbank, wie zum Beispiel „Create Index“, fest.
- *log* - Log wird benötigt, um Fehler in der Datenbank aufzeigen zu können. In diesem Parameter können unter anderem der Speicherort und der Name der Datei gewählt werden. Weiters können die Art und die Wichtigkeit der Fehlermeldung eingestellt werden. [26]

Um die Datenbank für andere Hosts freigeben zu können, müssen vorab einige Einstellungen vorgenommen werden. In „postgresql.conf“ muss der Parameter „listen_adresses“ angepasst werden. Dieser definiert den IP-Adressbereich, von welchem Datenbankverbindungen erlaubt sind. Im Idealfall sind Verbindungen von allen IP-Adressen möglich. Diese Einstellung wird über den Parameter **listen_addresses='*'** festgelegt. Weiters müssen in der Datei „pg_hbd.conf“ Berechtigungen für die jeweilige Datenbank und den Benutzer freigegeben werden. Die Einstellungen für dieses Projekt sind in Abbildung 5.7 ersichtlich.

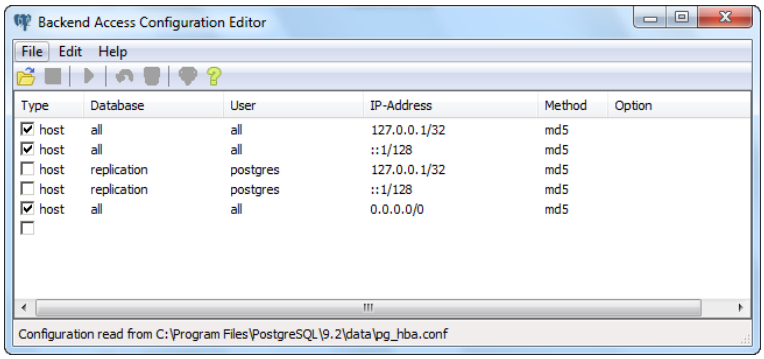


Abbildung 5.7.: hbd Konfiguration der Datenbank

PostgreSQL verwalten

Nach der Standardinstallation unter Windows steht die Software „pgAdmin III“ zur Verfügung. Diese ist eine grafische Benutzeroberfläche zur Administration von „PostgreSQL“. In Abbildung 5.8 sieht man die verfügbaren Datenbanken sowie Tabellen des Projektes.

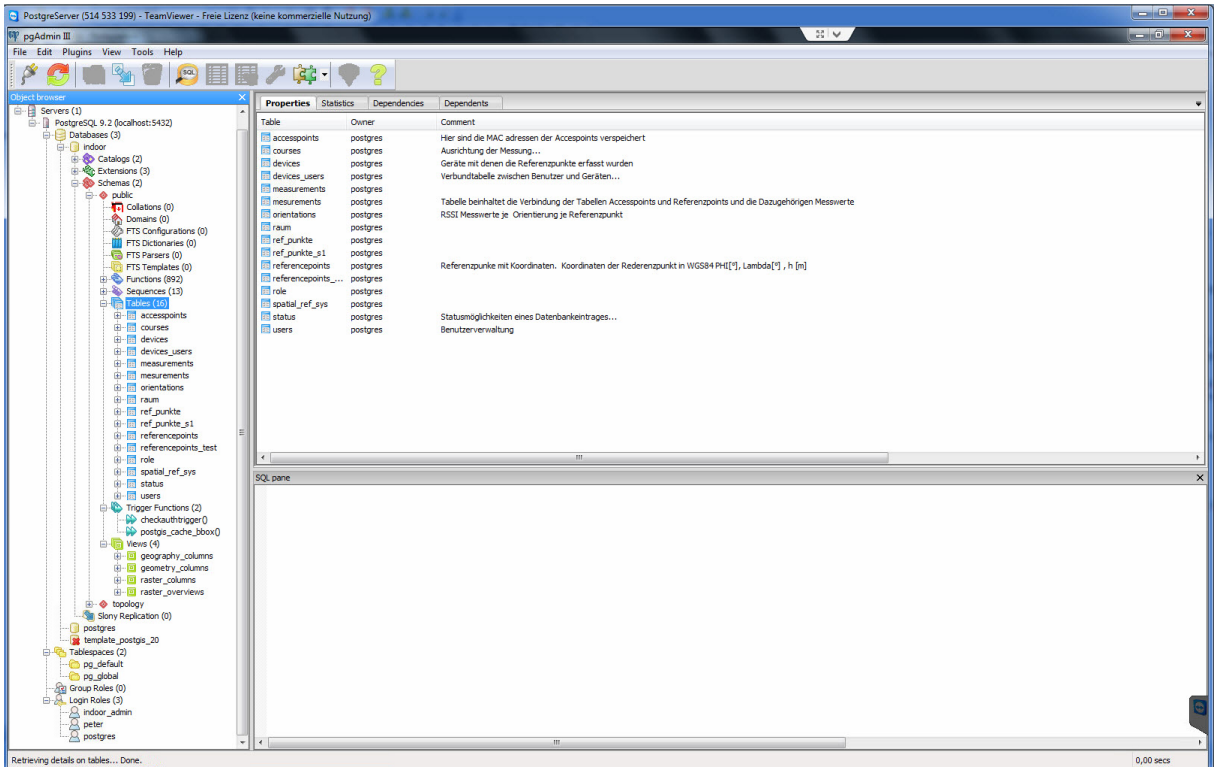


Abbildung 5.8.: pgAdmin III

Es können Datenbanken, Tabellen und SQL-Befehle über die grafische Oberfläche erstellt werden. Ebenfalls können die Inhalte der Tabellen angezeigt und verändert werden. Des weiteren können neue Benutzer und Benutzergruppen angelegt und definiert werden.
Um mit „PHP“ auf „PostgreSQL“ zugreifen zu können, sind in den Konfigurationsdateien des Webservers und des „PHP“ Änderungen notwendig.
In den Konfigurationsfile „httpd“ vom Apache Webserver muss folgendes eingetragen werden:

```
1 LoadFile ""C:\\php\\php5gs.dll"
2 LoadFile "C:\\php\\libpq.dll"
```

Listing 5.2: Apache 2.2 - mit PostgreSQL-Konfiguration

In der PHP Konfigurationsdatei „php.ini“ muss folgendes eingetragen werden:

```
1 extension=php_pdo_pgsql.dll
2 extension=php_pgsql.dll
```

Listing 5.3: PHP - mit PostgreSQL-Konfiguration

6. Smartphone Applikation

Im Folgenden wird die Funktionsweise der programmierten Smartphone-Applikation INAV (Indoor Navigation) erklärt. Ebenfalls werden die wesentlichen Programmteile erläutert und deren Handhabung beschrieben.

6.1. Applikation

Um die Indoornavigation nutzen zu können, muss man das Programm von der Homepage `http://inav.tugraz.at` downloaden und anschließend auf dem Smartphone installieren. Außerdem muss eine Internetverbindung vorhanden sein und die WLAN Funktion am Smartphone aktiviert sein, um alle Funktionen ausführen zu können. Beim Android Smartphone erscheint folgende Startoberfläche:

Abbildung 6.1.: Startoberfläche der Applikation

Von der Startseite aus kann man die Buttons „Referenzpunktverwaltung“, „Navigation starten“ oder „Karte“ aufrufen.

6.1.1. Referenzpunktverwaltung

Unter „Referenzpunktverwaltung“ kann man die Referenzpunkte aus der Datenbank, die sich auf dem Server befinden, laden und auch nach Datensätzen suchen, sofern man über den Namen des Bezugspunktes Bescheid weiß. In Abbildung 6.2 sieht man eine Auflistung der Referenzpunkte aus der Datenbank.

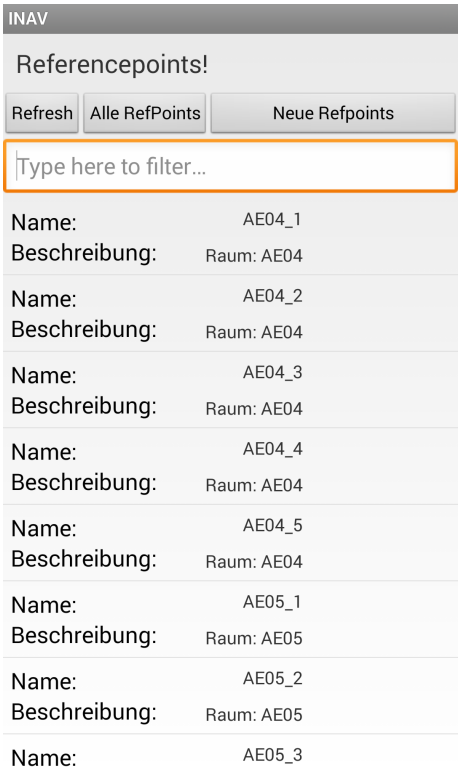


Abbildung 6.2.: Referenzpunktverwaltung

Im folgenden Kapitelabschnitt werden nähere Details über die „Referenzpunktverwaltung“ erklärt.

Referenzpunktdetails

Nachdem man einen Referenzpunkt ausgewählt hat (durch antippen/anklicken), bekommt man aus der Datenbank detaillierte Informationen über den selektierten Bezugspunkt. Zu diesen Informationen zählen der Name des Punktes, eine kurze Beschreibung der Lage des Punktes, das Erfassungs- und Änderungsdatum sowie eventuell vorhandene Messungen zu Accesspoints. Weiters hat man die Möglichkeit, die Werte zu aktualisieren. In Abbildung 6.3 kann man die Details zu einem ausgewählte Referenzpunkt ablesen.

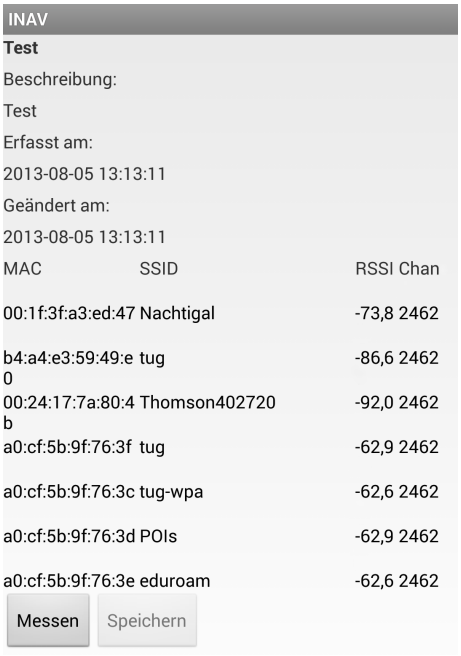


Abbildung 6.3.: Detailfenster Referenzpunkt

Referenzpunktmessung

Wenn entweder noch keine Messung vorhanden ist oder man diese Position erneuern möchte, geht man auf den Button „Messen“, um diese Aktion ausführen zu können. Es werden in vier Himmelsrichtungen RSSI Werte zu den Accesspoints gemessen, gemittelt und vor Speicherung der Daten angezeigt. Man hat die Möglichkeit, die Messung zu wiederholen oder die Werte in die Datenbank zu übertragen.
In Abbildung 6.4 wird eine Messung in Richtung „Osten“ gezeigt. Man erkennt, welche Accesspoints (MAC) gemessen werden und welche Signalstärken (RSSI) vorherrschen. Mit dem „Save Button“ beendet man die laufende Messung zur aktuellen Himmelsrichtung. Danach wird die nächste Himmelsrichtung freigeschaltet und der Vorgang kann wiederholt werden.

Abbildung 6.4.: Messung

6.1.2. Navigation starten

Im Menüpunkt „Navigation starten“ zeigt die Applikation in Echtzeit die aktuelle Position des Smartphones. Hierzu werden die RSSI Werte an den Server geschickt und dieser berechnet nach einem Algorithmus (siehe Kapitel 7) die aktuelle Position und sendet diese an das Smartphone zurück. Als Benutzer bekommt man den nächstgelegenen Referenzpunkt als Text ausgegeben, wie in Abbildung 6.5, mit dem Referenzpunkt „Test“ dargestellt wird. Als Zusatzinformation wird die aktuelle Messung (Uhrzeit, Datum, verfügbare Accesspoints) mit den dazugehörigen RSSI Werten angezeigt.

Abbildung 6.5.: aktuelle Positionsbestimmung

6.1.3. Karte

Im Menüpunkt „Karte“ wird die aktuelle Position auf einer „Google Maps Karte“ angezeigt. Die Berechnung der Position erfolgt wie in 6.1.2 beschrieben. Die Vorgehensweise der Implementierung einer „Google Maps Karte“ in die Android Applikation ist im Anhang A zu finden. In Abbildung 6.6 sieht man in Echtzeit die berechnete Indoor-Position auf einer „Google Maps Karte“.

Abbildung 6.6.: Google Maps Karte

6.2. Service

Service sind Hilfsprogramme, die gewisse Routinen im Hintergrund erledigen und die Werte in das vorgesehene Format, je nach Aktion bringen, um sie weiter verarbeiten zu können.

6.2.1. JSONWebservice

Die Klasse „JSONWebservice“ beinhaltet das Zusammensetzen der JSON Objekte und der eventuellen Parameter (JSON Array) und übergibt das JSON Objekt an die Klasse

„HttpClient“. Diese sendet anschließend die JSON-formatierten Daten an das definierte Webservice. Die Datenformatierung wird in „Kapitel 4 - Schnittstellen“ beschrieben.

Methoden von JSONWebservice

Die JSONWebservice-Klasse hat drei Methoden, die für die Formatierung der JSON Objekte zur Verfügung stehen.

connectToDispatcher

Diese Methode bekommt folgende Parameter, die formatiert werden:

- *URL <string>* - ist die http Adresse des Webservices
- *Dienst <string>* - ist der aufgerufene Name der Klasse im Webservice
- *Operation <string>* - ist die Methode der genannten Klasse

connectToDispatcher

Diese Methode unterscheidet sich durch die Anzahl der Parameter und des Datentyps. Hier kann man ein JSONObject als Parameterwert verwenden:

- *URL <string>* - ist die http Adresse des Webservices
- *Dienst <string>* - ist der aufgerufene Name der Klasse im Webservice
- *Operation <string>* - ist die Methode der genannten Klasse
- *Params <string>* - ist der Name des Übergabeparameters der Methode
- *JSONObjParams <JSONObject>* - ist ein Parameter als JSON Objekt

connectToDispatcher

Diese Methode unterscheidet sich durch die Anzahl der Parameter und bekommt folgende Übergabewerte:

- *URL <string>* - ist die http Adresse des Webservices
- *Dienst <string>* - ist der aufgerufene Name der Klasse im Webservice
- *Operation <string>* - ist die Methode der genannten Klasse
- *Params <string>* - ist der Name des Übergabeparameters der Methode
- *ParamsValue <string>* - ist der Wert des Parameters

6.2.2. WifiAndInternetState

In dieser Klasse wird der Status des WLANs ermittelt und überprüft, ob eine Internetverbindung vorhanden ist.

Methoden von WifiAndInternetState

Folgende zwei Methoden stehen in der Klasse „WifiAndInternetState“ zur Verfügung:

- *isWifiEnabled*- Diese Methode liefert den Status der WLAN Verbindung.
- *isConnectingToInternet* - Hier wird überprüft, ob eine Internetverbindung vorhanden ist. Es ist nicht von Bedeutung, ob über WLAN oder HSDPA, GPRS Internetempfang existiert. Wichtig ist, dass eine Verbindung zum Internet besteht.

6.2.3. HttpClient

Diese Klasse sendet den übergebenen URL und das vom JSON Webservice vorbereitete JSON Objekt an den Webserver und verarbeitet im Anschluss daran die Antwort des Webservers. Mit Hilfe folgender Methode wird die Aktion ausgeführt:

SendHttpPost

- *URL <string>* - ist die http Adresse des Webservices
- *jsonObjSend <JSONObject>* - das vom JSON Webservice formatierte Objekt

6.2.4. ReferencepointDBAdapter

Diese Klasse speichert die bereits vorhandene Referenzpunktliste aus der Datenbank lokal auf dem Smartphone. Mit Hilfe der Methoden dieser Klasse wird die Referenzpunktliste verwaltet. Es werden Punkte hinzugefügt, ausgelesen, gelöscht und aktualisiert. In der nachfolgenden Aufzählung sind die zwei wesentlichen Methoden, die per Button von der Applikation ausgeführt werden können, aufgelistet:

- *fetchAllReferencepoints* - Mit dieser Methode werden die Daten zu allen Referenzpunkten aus der Datenbank geholt.
- *fetchReferencepointsByName* - Ein für die weitere Verarbeitung notwendige Methode, da sie die Informationen zu den Referenzpunkten aus dem Speicher des Smartphones liest.

6.2.5. Berechnungen

In dieser Klasse befinden sich etwaige Berechnungen, die für das Aufbereiten der Daten notwendig sind. Unter anderem befinden sich in dieser Liste die Berechnungen des Mittelwertes der RSSI Werte, sowie ein Iterator, der durch jede Himmelsrichtung pro gemessener MAC Adresse geht und den Mittelwert berechnet. In weiterer Folge bereitet der Iterator die Werte für die Übergabe an die Datenbank vor.

- *jsonObjAndArrIteratorForRefPunkt* - Diese Methode iteriert durch die aufgezeichneten Messergebnisse. Wenn diese bei den Epochen pro MAC Adresse angelangt ist, wird ein Mittelwert berechnet. Wenn dies erfolgreich ist, wird unter Berücksichtigung des berechneten Mittelwerts und den dazugehörigen Informationen (bssid, channel, ssid und mean_rssi) ein JSON Objekt erstellt.
- *berechneMittelwert* - Berechnet den Mittelwert eines übergebenen Arrays mit Integer Werten.
- *berechneMittelwertDouble* - Berechnet ebenfalls den Mittelwert eines übergebenen Arrays, jedoch mit Double Precision-Werten.

6.2.6. Messung

In dieser Klasse erfolgt die Messung der WLAN Daten und in weiterer Folge die lokale Speicherung auf dem Smartphone. Das Zeitintervall, in dem die Daten ausgelesen werden, wird aus der „Ressourcen Datei“ geladen. Es wird ein Thread gestartet, der in bestimmten Zeitabständen die WLAN Daten von der WLAN Karte ausliest. Nachfolgend sind die primären Methoden angeführt.

- *starteMessung* - Diese Methode beginnt mit der Messung der WLAN Daten.
- *Timer_ Tick* - Es werden die aktuellen Messwerte von der WLAN Karte ausgelesen, das Ergebnis wird auf dem Display dargestellt und für die weitere Verarbeitung in einem Array gespeichert.

6.2.7. Zusammenfassung

Zusammenfassend ist zu erwähnen, dass die Smartphone-Applikation modular entwickelt worden ist und deshalb relativ einfach um weitere Funktionalitäten erweitert werden kann. Für die Kommunikation mit dem Webservice ist die Klasse JSONWebservice ausschlaggebend. In Abbildung 6.7 wird die Interaktion der wichtigsten Methoden mit dem Webserver und der dahinterliegenden Datenbank schematisch dargestellt.

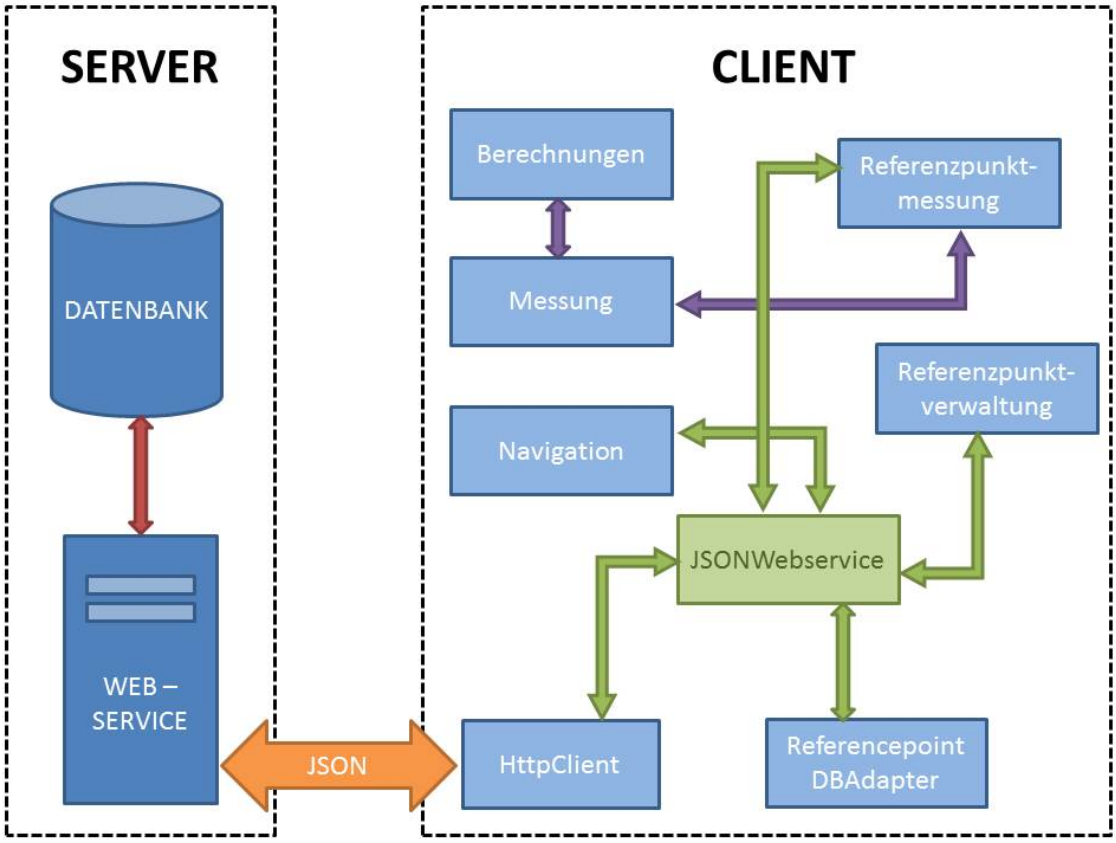


Abbildung 6.7.: Interaktion Webserver mit Client

7. Fingerprinting Algorithmus

Im Folgenden wird die Umsetzung des Fingerprinting Algorithmus in „PHP“ detailliert erläutert. Wie in Kapitel 2.6.2 erwähnt, soll aus den aktuell gemessenen RSSI Werten mit Hilfe der bereits vorhandenen Werte aus der Datenbank die Position des mobilen Gerätes zu dem nächstgelegen Referenzpunkt bestimmt werden. Dieser Vorgang basiert auf einem Webservice unter Verwendung eines Webservers, der wie in Kapitel 4.1.11 Übergabeparameter zur Positionsbestimmung erwartet.
Diese Art der Positionsbestimmung hat den Vorteil, dass nicht ausschließlich die im Rahmen dieser Masterarbeit programmierte Android-Applikation zur Positionsbestimmung genutzt wird, sondern jede beliebige Software über die definierte Schnittstelle die Position berechnen kann.
Ein weiterer Vorteil ist, dass der Algorithmus, der zur Berechnung verwendet wird, auf dem Webserver liegt und somit leicht zu ändern, verbessern und erweitern ist.
Nachteil dieser Variante ist, dass eine ständige Verbindung zum Internet Voraussetzung für die Bestimmung der Position ist.
Zur Positionsberechnung werden die Messwerte an das Webservice übergeben.
In weiterer Folgen wird eine SQL-Abfrage getätigt und zu einem String zusammengesetzt. Dieser Vorgang geschieht in der Funktion „SQL-Stirng SSID“ und liefert die MAC Adressen zurück.
Anschließend erfolgt eine Datenbank Abfrage, indem alle Referenzpunkte abgefragt werden, in denen die MAC-Adressen gemessen wurden.
Im Anschluss daran findet die eigentliche Berechnung mit Hilfe der „Fingerprinting“-Methode statt. Als Übergabeparameter werden die Referenzpunktliste und die Messung übermittelt. In der Methode „Fingerprinting“ wird die Position des mobilen Gerätes mittels einfachen „Fingerprinting Algorithmus“ bestimmt. Dazu werden jene Referenzpunkte, in denen die gemessenen MAC-Adressen vorhanden sind, zum Vergleich herangezogen.
In weiterer Folge werden die bereits in der Offline-Phase gemessenen Daten des Referenzpunktes, die in der Datenbank gespeichert sind, mit den aktuellen Messwerten verglichen. Dabei wird mit Hilfe der Formel 2.3 die kleinste Abweichung gesucht.
Wenn die Liste der Referenzpunkte abgearbeitet ist, steht am Ende des Algorithmus der Index des Minimums fest. Somit kann die aktuelle Position zu einem Referenzpunkt zugeordnet werden. Das Webservice gibt als Antwort den Namen des Referenzpunktes, die Koordinaten und die Standardabweichung zurück.

```
2  require_once('interface_service.php');
3  require_once('class_jsonhandler.php');
4  require_once "class_functions.php";

6  /**
7   * @class Position - Klasse für die Berechnung der Position
8   * @package at.petergoda.tug.inav.Position
9   */
10 class Position implements service{

12   /**
13    * Private Variable $db speichert die Datenbankverbindung
14    * @var object $db
15    */
16   private $db;
17   /**
18   * private Variable $result speichert den Ergebnisarray ab
19   * @var array $result
20   */
21   private static $result;
22   /**
23    * Konstruktor Methode der die Datenbankverbindung erwartet.
24      * Damit ist gewährleistet, dass diese Klasse eine Verbindung
25      * zur Datenbank herstellen kann.
26    * @method __construct Konstruktor
27    * @param object $_db Datenbankobjekt
28    */
29   public function __construct($_db){
30     #speichert den Datenbankverbindungsobjekt in den eigenen Instanz (this->db) ab
31     $this->db = $_db;
32   }
33   /**
34   * @method getResult - liefert das
35   * @return $result - Resultat wird zurückgegeben
36   */
37   public function getResult(){
38     return self::$result;
39   }
40   /**
41   * @method filterMessung - Methode der die Schwächsten Signale entfernt
42   *
43   *
44   */
45   private function filterMessung($messung){

47   // Obtain a list of columns
48   foreach ($messung as $key => $row) {
49     $rssi[$key]  = $row['rssi'];

51   }

53 // Sort the data with volume descending, edition ascending
54 // Add $data as the last parameter, to sort by the common key
55   array_multisort($rssi, SORT_DESC, $messung);
```

```
57    $j=0;
58    for ($i=0; $i<sizeof($messung)-1;$i++){

60        if($messung[$i][0]['rssi']>-90){

62          #$messung_neu[$j][0]=$messung[$i][0];
63          array_push($messung_neu, $messung[$i]);

65        }
66        if(sizeof($messung)-1>5 && $i==5){
67          break;
68        }
69      }
70      return $messung_neu;
71    }
72    /**
73    * @method positionBerechnen - Methode für das Berechnen der
74      *           Position aus dem Übergeben MAC Adressen und RSSI Werten
75    * @param $messung <array>  - Die Array $messung beinhaltet die MAC
76                                    Adresse und die dazugehörige RSSI Werte
77    */
78    public function positionBerechnen($messung){

80        $messung_filtered = self::filterMessung($messung);
81        // Teil SQLString für die Macadressen
82          $sqlMac=self::sqlStringSsid($messung_filtered);
83        // Datenbankabfrage zusammenfügen
84        try {

86          $stmt = $this->db->prepare("SELECT
87                          DISTINCT(rp.id),
88                          rp.name,
89                          rp.beschreibung
90                        FROM
91                          public.".AP." ap
92                          LEFT JOIN public.".MS." ms ON (ms.accesspoint_id=ap.id)
93                          LEFT JOIN public.".RP." rp ON (rp.id=ms.referencepoint_id)
94                        WHERE
95                           ".$sqlMac);
96          // Wenn das SQL Statement nicht erstellt werden konnte wir ein Fehler geworfen
97          if(!$stmt) throw new SQLException($db->error,$db->errno);
98          $stmt->execute();

100       // in $referencePunktmitMAC befinden sich alle Referenzpunkt in
101         // dem die gemessenen MAC adressen gemessen wurden
102         $referencePunktmitMAC=$stmt->fetchAll();
103         //print_r($referencePunktmitMAC);
104         //Das SQLStatement wird geschlossen/beendet
105         $stmt->closeCursor();
106         // Rückgabe das es geklapp hat

108       }catch(Exception $e){
109         if($stmt) $stmt->close();
110         //throw $e;
```

```
111           $_SESSION['error'].=$e;
112           return false;

114         }//ende catch
115         $result=self::fingerprinting($referencePunktmitMAC,$messung_filtered);
116         if(!empty($result)){
117           self::$result=$result;
118           return true;
119         }else{
120           return false;
121         }

123     }// Ende positionBerechnen

125     /**
126     * @method private sqlStringSssid
127     * @param array $macAdressen Beinhaltet die MAC Adressen in einem array
128     * array(array(mac=>'00:24:51:cb:95:c0'),array(mac=>'b4:a4:e3:59:49:e0'.....))
129     * return $text - SQL Teilstring Baustein für die Abfrage in der Accespointsdatenbank
130     */
131     private function sqlStringSsid($macAdressen){
132       $text=''; // Variable Text null
133       // Baut den suchstring für den SSID (name der AP) zusammen
134       for ($i=0; $i<sizeof($macAdressen)-1;$i++){

136       $text."ap.mac='". $macAdressen[$i][0]['mac']."' OR ";

138         }
139       $text."ap.mac='".$macAdressen[$i][0]['mac']."'";

141       return $text;

143     }
144     /**
145     * @method fingerprinting - In dieser Methode wird die Position des Smartphones
146     *                          mittels einfachen Fingerprinting bestimmt
147     * @param <array> $referencePunktmitMAC
148     * @param <array> $messung
149     *   @return <array> $resultsarray  -  Beinhaltet die ID des nächstegelegenen
150     *                   Referenzpunktes und die geschätzte Standardabweichung
151     *                   und den Anzahl der passenden Messwerte
152     */
153     protected function fingerprinting($referencePunktmitMAC,$messung){

155       $ssr_temp_v=100.0; // Standartabweichung auf 100 gesetzt Temporäre Variable
156       $max_messwert=0; // Anzahl der gefundenen passenden Referenzpunkte

158   // für jeden Referenzpunkt werden die Daten geholt und mit
159   // den gemessene RSSI Werte verglichen
160         foreach($referencePunktmitMAC as $key => $value){
161           try {
162             # Holen der Punkte in dem der gemessene Referenzpunkt
163             $stmt = $this->db->prepare("SELECT
164                           ms.accesspoint_id,
165                           ms.referencepoint_id,
```

```
166                          ms.rssi as rssi ,
167                          ap.mac as mac
168                         FROM
169                           public.".MS." ms
170                         LEFT JOIN public.".AP." ap ON (ap.id=ms.accesspoint_id)
171                         WHERE
172                            ms.referencepoint_id = :id");
173            // Wenn das SQL Statement nicht erstellt werden konnte wir ein Fehler geworfen
174            if(!$stmt) throw new SQLException($db->error,$db->errno);
175            // Die Parameter werden an den vorbereiteten SQLStatement gebunden
176                     // dies dient zur Sicherheit- SQL Injections sind somit nicht möglich
177            $stmt->execute(array(':id'=>$referencePunktmitMAC[$key]['id']));

179            // in $MessungMitDaten befinden sich alle Referenzpunkt in dem
180                     // die gemessenen MAC adressen gemessen wurden
181            $MessungMitDaten=$stmt->fetchAll();
182            //Das SQLStatement wird geschlossen/beendet
183            $stmt->closeCursor();


186          }catch(Exception $e){
187            if($stmt) $stmt->close();
188            //throw $e;
189            $_SESSION['error'].=$e;

191          }//ende catch
192          $anzahl_vom_messwerte=0;
193          $ssr=0.0; // Standardabweichung
194  // Vergleichen der RSSI werte mit jeder gemessenen AP Wert
195  // und berechnet die Standartabweichung $ssr
196          foreach($messung as $mkey => $mvalue){
197            //Prüfen ob Messwert im Datenbankarray vorhanden ist
198            if (multidimensional_search($MessungMitDaten,
199                     array('mac'=>$mvalue[0]['mac'],'rssi' =>$mvalue[0]['rssi'] )) ) {
200                $anzahl_vom_messwerte++;
201               }

203             $rssi_db=multidimensional_search_retint($MessungMitDaten,
204                     array('mac'=>$mvalue[0]['mac']));

206             // RSSI Wert aus der Datenbank
207             $rsi_db=floatval($MessungMitDaten[$rssi_db]['rssi']) ;
208             $rsi_gm=floatval($mvalue[0]['rssi']); // RSSI Wert gemessen
209            // Quadriert die differenz der RSSI_DB-RSSI_GM
210             $ssr_temp= pow($rsi_db-$rsi_gm,2);

212             $ssr+=$ssr_temp; // zusammensetzten der Standartabweichung für alle AP RSSIs

214          }# ende foreach2

216           $ssr=sqrt($ssr); // Wurzel der Standardabweichung
217                   // Zuweisung Anzahl der Messwerte zu der jeweiligen Datensatz
218           $MessungMitDaten['messwerte']=$anzahl_vom_messwerte;
219                  // Zuw der Standardabweichung zu der jeweiligen Datensatz
220           $MessungMitDaten['ssr']=$ssr;
```

```
222             //Zusammenfassen der Standpunkte mit ihren Werten
223           //$werte[$referencePunktmitMAC[$key]['id']]=$MessungMitDaten;

225   // Vergleichen ob die Standardabweichung zu den Anderen Referenzpunkten.
226   // Wenn dies zutrifft wird die neue Standardabweichung mit den alten überschrieben
227           if($ssr<=$ssr_temp_v){
228             $ssr_temp_v=$ssr;
229             $min_punkt_nr=$referencePunktmitMAC[$key]['id'];
230             $min_punkt_name=$referencePunktmitMAC[$key]['name'];
231             $min_punkt_beschreibung=$referencePunktmitMAC[$key]['beschreibung'];

233             }
234             if($anzahl_vom_messwerte>=$max_messwert){ // Anzahl gefundene Messwerte
235              $max_messwert=$anzahl_vom_messwerte;

237             }
238         } # ende foreach 1
239   // zusammensetzten des Resultatsarrays was zurückgegeben wird
240         $resultsarray['refpunkt_id']=$min_punkt_nr; // referenzpunkt ID - nächstgelegene Position
241   // referenzpunkt ID - nächstgelegene Position
242         $resultsarray['refpunkt_name']=$min_punkt_name." ".$ssr_temp_v.$min_punkt_beschreibung;
243         $resultsarray['ssr']=$ssr_temp_v; // die kleinste Standardabweichung
244         $resultsarray['messwerte']=$max_messwert;
245         if(self::getKoordinateforReferencepointById($min_punkt_nr)){
246           $resultsarray['koordinaten']=self::$result[0]['koordinaten'];
247           $resultsarray['lat']=self::$result[0]['lat'];
248           $resultsarray['lng']=self::$result[0]['lng'];
249           $resultsarray['anz']=1;
250         }else{
251           $resultsarray['koordinaten']='';
252           $resultsarray['lat']='';
253           $resultsarray['lng']='';
254         }
255         return $resultsarray;
256     } // ende funktion fingerprinting

258       /**
259       * Holt die Koordianten für eine Referenzpunkt aus der Datenbank
260       * @param $id - Id des gesuchten Referenzpunktes
261       */
262       public function getKoordinateforReferencepointById($id){
263         try {
264             # Holen der Punkte in dem der gemessene Referenzpunkt
265             $stmt = $this->db->prepare("SELECT
266             rp.id,ST_ASTEXT(koordinaten) as koordinaten,
267                       ST_X(koordinaten) as lng, ST_Y(koordinaten) as lat,name
268                           FROM
269                             public.".RP." rp
270                           WHERE
271                              rp.id = :id");
272             // Wenn das SQL Statement nicht erstellt werden konnte wir ein Fehler geworfen
273             if(!$stmt) throw new SQLException($db->error,$db->errno);
274   // Die Parameter werden an den vorbereiteten SQLStatement gebunden -
275   // dies dient zur Sicherheit- SQL Injections sind somit nicht möglich
```

```
276             $stmt->execute(array(':id'=>$id));

278 // in $MessungMitDaten befinden sich alle Referenzpunkt in dem
279 // die gemessenen MAC Adressen gemessen wurden
280             $koordinatenAsText=$stmt->fetchAll();

282             //Das SQLStatement wird geschlossen/beendet
283             $stmt->closeCursor();


286           }catch(Exception $e){
287             if($stmt) $stmt->close();
288             //throw $e;
289             $_SESSION['error'].=$e;

291           }//ende catch

293           if(!empty($koordinatenAsText)){
294           self::$result=$koordinatenAsText;
295             return true;
296           }else{
297             return false;
298           }
299       }
300 }
```

Listing 7.1: Beispiel - PHP „Klasse Position“ zur Berechnung der Position mittels WLAN Fingerprinting Methode

8. Realisierung

8.1. Datenbankstruktur

Die Messdaten und weitere zugehörige Daten sind in einer PostgreSQL Datenbank gespeichert. Die Struktur der Datenbank wurde so geplant, dass sie den Normalisierungsformen entsprechen. In den einzelnen Tabellen stehen die Daten, die aus folgenden Anforderungen abgeleitet werden:

- Benutzerverwaltung (Rechte und Rollen der Benutzer)
- Umgebungsinformationen (verfügbare Accesspoints, Räumlichkeiten)
- Signalinformationen (gemessene Signalstärken und weitere Parameter)
- Zusatzinformationen (Orientierung, Status, Geräte zur Messung der Parameter)

Die genauen Inhalte der Tabellen werden in den nächsten Abschnitten erklärt.

8.1.1. Tabelle 'accespoints'

Field	Typ	Null	Extra
id	serial	not NULL	primary key, autoincrement
mac	macaddr	not NULL	
ssid	character varying(255)		
status_id	integer	not NULL	Default 1
created	timestamp	not NULL	
modified	timestamp	not NULL	

Tabelle 8.1.: Tabelle 'accesspoints'

Die Tabelle 'accesspoints' beinhaltet die Informationen zu den Accesspoints. Dazu wird die eindeutige MAC Adresse im Attribut „mac“ gespeichert. Der Name des Accespoints steht im Attribut „ssid“. Ein Fremdschlüssel der Tabelle „status“ ist „status_id“, in dem der aktuelle Status des Accesspoints festgelegt werden kann. Das Attribut „id“ ist der

Primärschlüssel der Tabelle. Mit „created“ und „modified“ wird festgelegt, wann ein Datensatz erzeugt beziehungsweise verändert wurde. Diese drei Attribute kommen in fast alle weiteren Tabellen vor und werden nicht mehr näher erläutert.

8.1.2. Tabelle 'courses'

Field	Typ	Null	Extra
id	serial	not NULL	primary key, autoincrement
name	character varying(255)		

Tabelle 8.2.: Tabelle 'courses'

Die Tabelle 'courses' ist eine Hilfstabelle, in welcher die Himmelsrichtungsmessungen für Nord-Ost-Süd-West stehen. Diese Tabelle entspricht der vorgegebenen Normalform.

8.1.3. Tabelle 'devices'

Field	Typ	Null	Extra
id	serial	not NULL	primary key, autoincrement
os	character varying(255)		
version	character varying(255)		
app_version	character varying(255)		
status_id	integer	not NULL	
device	character varying(255)		
model	character varying(255)		
product	character varying(255)		
manufacturer	character varying(255)		
displaysize	character varying(255)		
created	timestamp	not NULL	
modified	timestamp	not NULL	

Tabelle 8.3.: Tabelle 'devices'

In der Tabelle 'devices' stehen Informationen zu den einzelnen Geräten, mit denen die Referenzmessungen durchgeführt werden. In der nachfolgenden Aufzählung befindet sich die Beschreibungen der einzelnen Attribute.

- *os* - Betriebssystem des Gerätes

- *version* - aktuelle Version des Betriebssystems
- *app_ version* - Version der Applikation, mit der die Messungen erfolgen
- -status_id - aktueller Status des Gerätes
- *device* - Name des Gerätes (Smartphones)
- *product* - Produktname, der vom Hersteller des Gerätes festgelegt ist
- *manufacturer* - Herstellername
- *displaysize* - Größe des Displays

8.1.4. Tabelle 'devices_users'

Field	Typ	Null	Extra
id	serial	not NULL	primary key, autoincrement
device_id	integer		
user_id	integer		

Tabelle 8.4.: Tabelle 'devices_users'

Die Tabelle 'devices_users' verknüpft den Benutzer mit einem Gerät. Im Attribut „device_id“ steht die ID des Gerätes und im „user_id“ steht die dazugehörige ID eines Benutzers aus der Tabelle 'users'.

8.1.5. Tabelle 'measurements'

Field	Typ	Null	Extra
id	serial	not NULL	primary key, autoincrement
accesspoint_id	integer	not NULL	
referencepoint_id	integer	not NULL	
rssi	double precision	not NULL	
channel	integer		
status_id	integer	not NULL	default 1
created	timestamp	not NULL	
modified	timestamp	not NULL	

Tabelle 8.5.: Tabelle 'measurements'

Die Tabelle 'measurements' beinhaltet die Messungen der WLAN Daten an den Referenzpunkten. Diese Tabelle wird auch für die Berechnung der Position herangezogen. In der nachfolgenden Auflistung stehen die einzelnen Attribute der Tabelle.

- *accesspoint_id* - Fremdschlüssel zu der Tabelle 'accesspoints'
- *referencepoint_id* - Fremdschlüssel zu der Tabelle 'referencepoints'
- *rssi* - gemittelte Signalstärke über alle verfügbaren Himmelrichtungsmessungen
- *channel* - Kanal auf dem der Acesspoint bei der Messung sendet
- *status_id* - Status der Messung

8.1.6. Tabelle 'orientations'

Field	Typ	Null	Extra
id	serial	not NULL	primary key, autoincrement
mesurement_id	integer	not NULL	
course_id	integer		
status_id	integer		
rssi	double precision		
status_id	integer	not NULL	default 1
created	timestamp	not NULL	
modified	timestamp	not NULL	

Tabelle 8.6.: Tabelle 'orientations'

In den Hilfstabellen von 'measurements' und 'orientations' befinden sich die Werte der gemessenen Signalstärken pro Orientierung (Himmelsrichtung). Diese Tabelle ist über den Fremdschlüssel 'mesurements_id' zu den dazugehörigen Messungen in der Tabelle 'measurements' verknüpft. Weiters wird in den Attributen 'course_id' der Orientierungsschlüssel gespeichert sowie in 'status_id' der Status der Messung.

8.1.7. Tabelle 'referencepoints'

Field	Typ	Null	Extra
id	serial	not NULL	primary key, autoincrement
name	character varying(255)	not NULL	UNQUE
beschreibung	text		
status_id	integer	not NULL	default 1
koordinate	geometry		$WGS84$ ϕ[°], λ[°] , h [m]
created	timestamp	not NULL	
modified	timestamp	not NULL	

Tabelle 8.7.: Tabelle „referencepoints“

In der Tabelle „referencepoints“ sind die auf dem Reißbrett festgelegten Messpunkte gespeichert. Diese werden über die webbasierte Verwaltung eingetragen und verwaltet. Diese Tabelle beinhaltet neben dem Namen der einzelnen Referenzpunkte „name“ sowie Beschreibung „beschreibung“ auch die Koordinaten des Punktes im WGS84, die im System als ϕ[°], λ[°] und h [m] abgespeichert werden.

8.1.8. Tabelle 'status'

Field	Typ	Null	Extra
id	serial	not NULL	primary key, autoincrement
name	character varying(255)		
created	timestamp	not NULL	
modified	timestamp	not NULL	

Tabelle 8.8.: Tabelle „status“

In der Tabelle 'status' befinden sich die möglichen Statuswerte. Diese wären zum Beispiel - „aktiv“ - „inaktiv“ - „Bearbeitung“. Im Attribut 'name' steht genau dieser Wert. Diese Tabelle wird von mehreren Tabellen der Datenbank genutzt, um den aktuellen Status des Wertes verändern zu können und dadurch die Möglichkeit zu schaffen, die Werte zu filtern.

8.1.9. Tabelle 'users'

Field	Typ	Null	Extra
id	serial	not NULL	primary key, autoincrement
username	character varying(255)	NOT NULL	UNIQUE
email	character varying(255)	NOT NULL	UNIQUE
password	character varying(255)	NOT NULL	UNIQUE
status_id	integer	not NULL	Default 1
created	timestamp	not NULL	
modified	timestamp	not NULL	

Tabelle 8.9.: Tabelle 'users'

Die Tabelle 'users' beinhaltet die Benutzer, die das Verwaltungssystem administrieren dürfen.

- *username* - Benutzername muss eindeutig sein
- *email* - E-Mail Adresse des Benutzers muss ebenfalls eindeutig sein
- *passwort* - Verschlüsseltes Passwort des Benutzers

8.2. INAV - Datenverwaltung

Für die Datenverwaltung wurde eine webbasierte Lösung entwickelt und mittels „PHP“ „Framework“ „CakePHP“ umgesetzt. Die Verwaltung ist vorwiegend darauf ausgelegt, dass Referenzpunkte eingetragen werden können. Außerdem ist es vorgesehen, alle Daten aus der Datenbank anzeigen zu lassen und diese zu editieren. Die Verwaltung ist modular aufgebaut und kann mit weiteren Tabellen verknüpft beziehungsweise können weitere Funktionalitäten hinzugefügt werden.
In den nachfolgenden Grafiken wird ein Überblick der Funktionalität der Verwaltung gegeben.
Die Verwaltung ist unter `http://www.inav.tugraz.at:8080/website/` erreichbar und ist mit Hilfe eines Benutzerverwaltungssystems vor Zugriffen von unbefugten Personen geschützt. Siehe Abbildung 8.1

Abbildung 8.1.: Login in das Verwaltungssystem

Nach der Eingabe des Benutzernamens und des Passworts bekommt man die Möglichkeit, die Inhalte der einzelnen Tabellen anzeigen zu lassen. Dazu hat man, wie in Abbildung 8.2 ersichtlich ist, die Möglichkeit, eine der Tabellen auszuwählen.

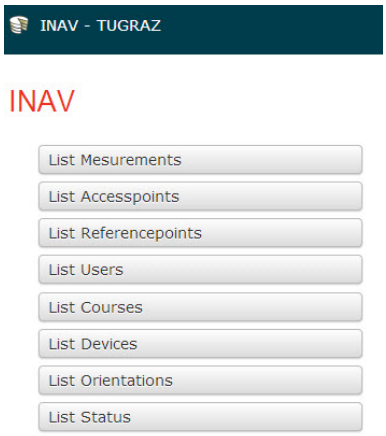


Abbildung 8.2.: Auswahl der Tabellen nach dem Login

Nach Auswahl einer Tabelle bekommt man die einzelnen Werte der Tabelle angezeigt und kann über den „View - Button“ die Details der Zeile anzeigen lassen. Ebenso ist es möglich, über den „Edit- Button“ die Werte zu bearbeiten. Anschließend gibt es noch die Möglichkeit, den Datensatz über den „Delete-Button“ zu löschen. Als Beispiel in der Abbildung 8.3 wurde die Tabelle „Measurements“ ausgewählt.

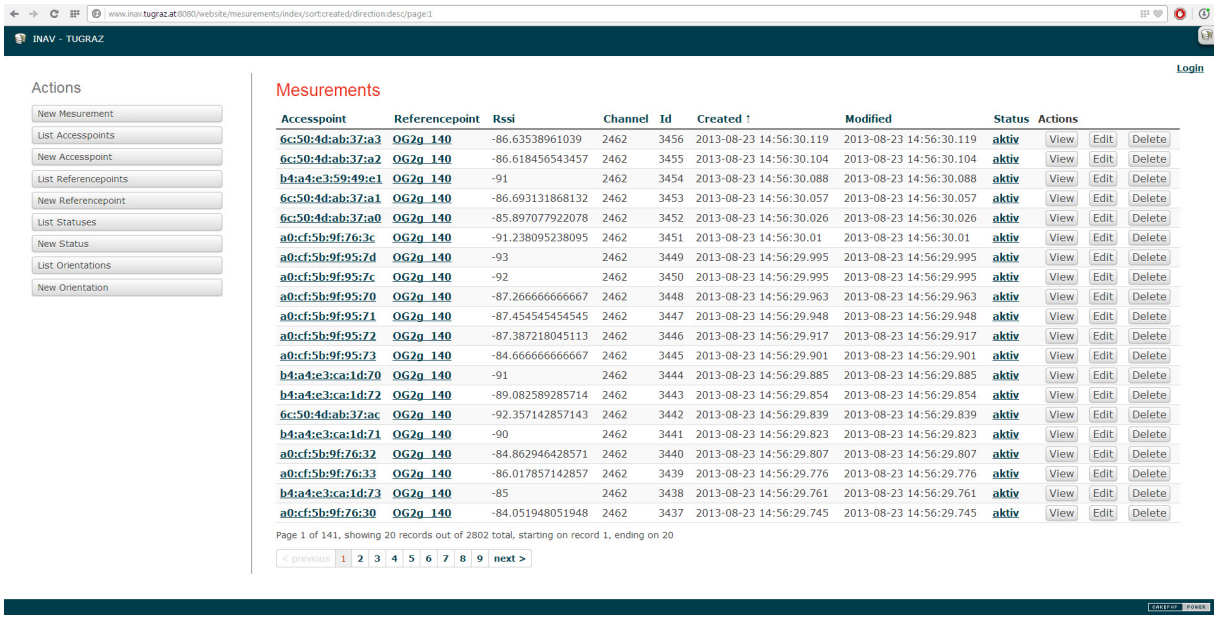


Abbildung 8.3.: Liste der Messungen aus der Tabelle „measurements“

Wenn man eine Messung ausgewählt hat, bekommt man auch die verknüpften Informationen zur Messung. In diesem Fall wären dies die Messwerte aus der Tabelle „orientations“, wo sich die Messungen zu den jeweiligen Himmelsrichtungen befinden. Weiters erlaubt die Verwaltung, zu den Informationen des gemessenen Accesspoints zu gelangen und zeigt ebenso die Details über den aktuellen Referenzpunkt. In der Abbildung 8.4 ist ein Beispiel einer Messung aufgelistet.

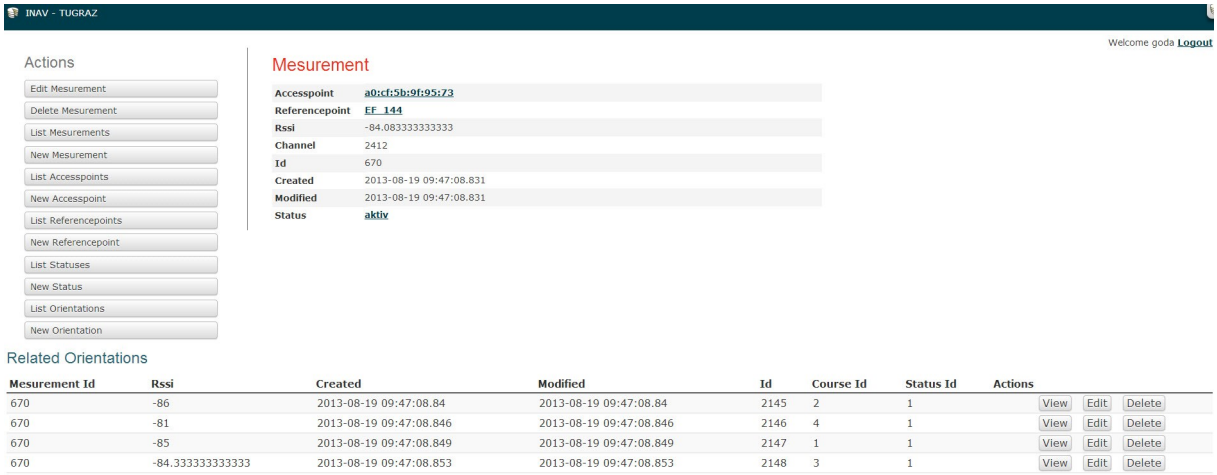


Abbildung 8.4.: Details zu einer Messung

Als Beispiel sei noch die Anlage eines neuen Referenzpunktes erwähnt. Man gelangt über

den Button „New Referencepoint“ zu der Oberfläche, siehe Abbildung 8.5, in der ein neuer Referenzpunkt eingetragen werden kann. Hier werden der Name des Referenzpunktes, die Koordinaten und mögliche Beschreibungen eingetragen.

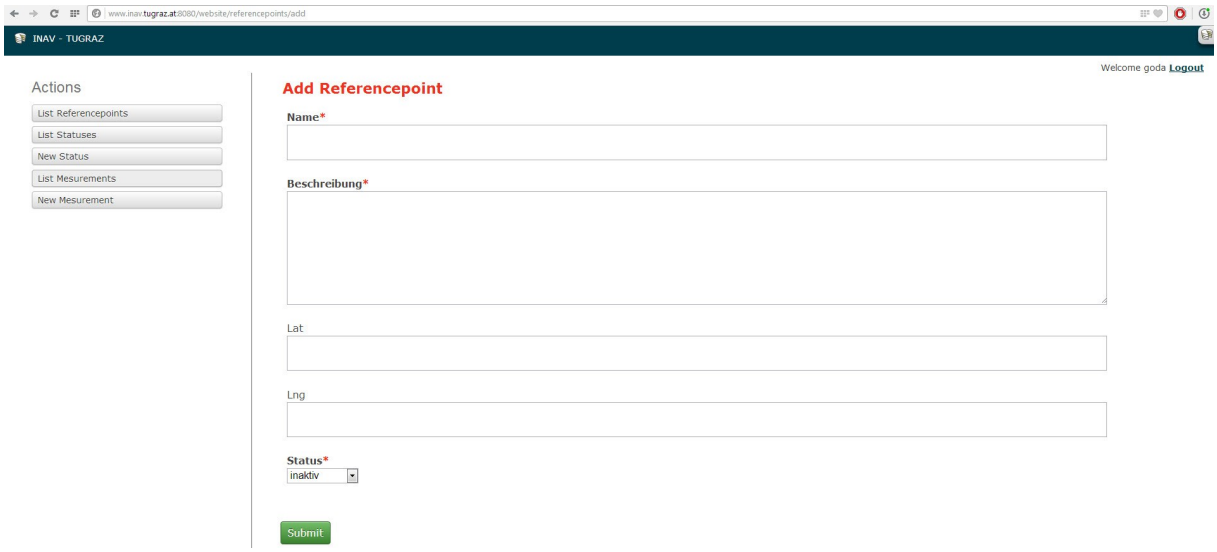
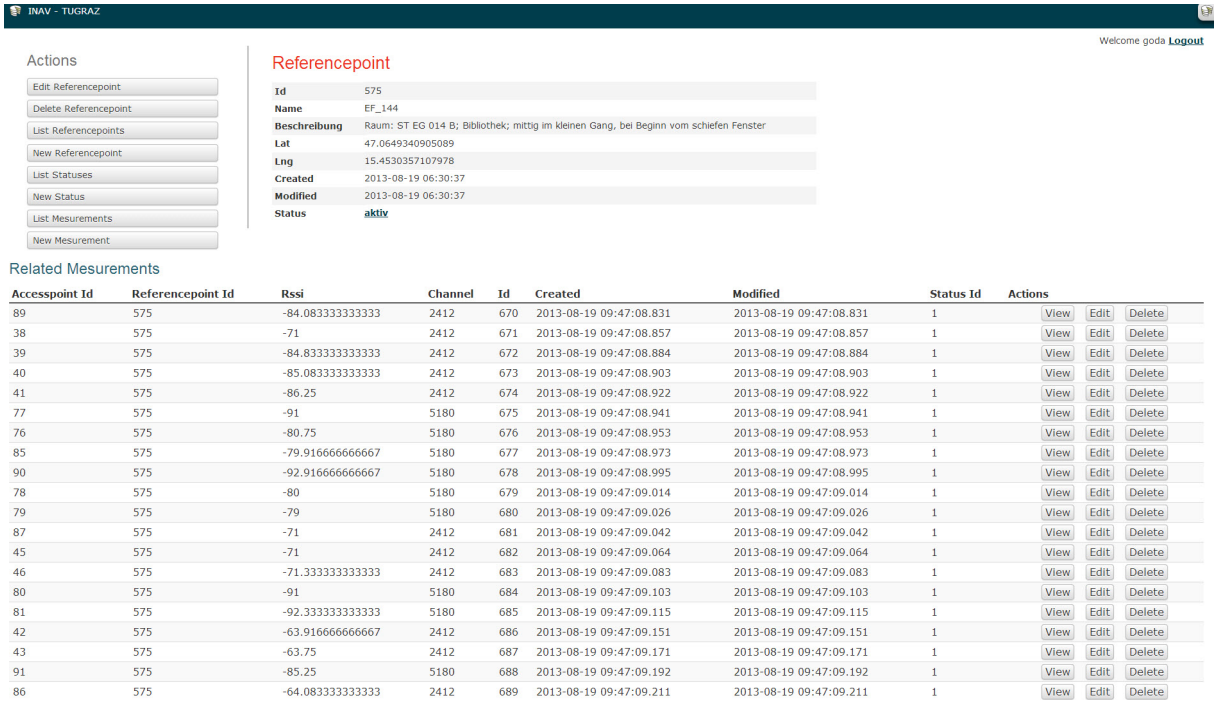


Abbildung 8.5.: neuen Referenzpunkt eintragen

Wenn man die Details von einem Referenzpunkt auswählt, an dem schon Messungen getätigt wurden, bekommt man eine Seite mit den Informationen zu den Referenzpunkten und es werden auch die dazugehörigen Messwerte aus der Tabelle „measurements“ geholt. Eine Beispielseite ist in der Abbildung 8.6 ersichtlich.



Accesspoint Id	Referencepoint Id	Rssi	Channel	Id	Created	Modified	Status Id	Actions
89	575	-84.083333333333	2412	670	2013-08-19 09:47:08.831	2013-08-19 09:47:08.831	1	View Edit Delete
38	575	-71	2412	671	2013-08-19 09:47:08.857	2013-08-19 09:47:08.857	1	View Edit Delete
39	575	-84.833333333333	2412	672	2013-08-19 09:47:08.884	2013-08-19 09:47:08.884	1	View Edit Delete
40	575	-85.083333333333	2412	673	2013-08-19 09:47:08.903	2013-08-19 09:47:08.903	1	View Edit Delete
41	575	-86.25	2412	674	2013-08-19 09:47:08.922	2013-08-19 09:47:08.922	1	View Edit Delete
77	575	-91	5180	675	2013-08-19 09:47:08.941	2013-08-19 09:47:08.941	1	View Edit Delete
76	575	-80.75	5180	676	2013-08-19 09:47:08.953	2013-08-19 09:47:08.953	1	View Edit Delete
85	575	-79.916666666667	5180	677	2013-08-19 09:47:08.973	2013-08-19 09:47:08.973	1	View Edit Delete
90	575	-92.916666666667	5180	678	2013-08-19 09:47:08.995	2013-08-19 09:47:08.995	1	View Edit Delete
78	575	-80	5180	679	2013-08-19 09:47:09.014	2013-08-19 09:47:09.014	1	View Edit Delete
79	575	-79	5180	680	2013-08-19 09:47:09.026	2013-08-19 09:47:09.026	1	View Edit Delete
87	575	-71	2412	681	2013-08-19 09:47:09.042	2013-08-19 09:47:09.042	1	View Edit Delete
45	575	-71	2412	682	2013-08-19 09:47:09.064	2013-08-19 09:47:09.064	1	View Edit Delete
46	575	-71.333333333333	2412	683	2013-08-19 09:47:09.083	2013-08-19 09:47:09.083	1	View Edit Delete
80	575	-91	5180	684	2013-08-19 09:47:09.103	2013-08-19 09:47:09.103	1	View Edit Delete
81	575	-92.333333333333	5180	685	2013-08-19 09:47:09.115	2013-08-19 09:47:09.115	1	View Edit Delete
42	575	-63.916666666667	2412	686	2013-08-19 09:47:09.151	2013-08-19 09:47:09.151	1	View Edit Delete
43	575	-63.75	2412	687	2013-08-19 09:47:09.171	2013-08-19 09:47:09.171	1	View Edit Delete
91	575	-85.25	5180	688	2013-08-19 09:47:09.192	2013-08-19 09:47:09.192	1	View Edit Delete
86	575	-64.083333333333	2412	689	2013-08-19 09:47:09.211	2013-08-19 09:47:09.211	1	View Edit Delete

Abbildung 8.6.: Detailseite eines Referenzpunktes

9. Testen der Implementierung

Im folgenden Kapitel werden Details zu den einzelnen Tests aufgezeigt. Diese werden in den Bereichen „Webserver“, „Smartphone“ und „Funktionalität zur Positionsbestimmung“ durchgeführt.

9.1. Testen der Positionsbestimmung

Die Positionsbestimmung soll im Gebäude der Technischen Universität Graz erfolgen. Dazu werden die Pläne von der „Steyrergasse 30“ verwendet. Die Grundrisse des Gebäudes stellt die „Bundesimmobiliengesellschaft“ digital zur Verfügung. Sie sind im Rahmen des Projektes „LOBSTER“ georeferenziert worden und werden für diese Arbeit zur Verwendung freigegeben.

9.1.1. Vorbereitungen zur Referenzpunktmessung

Um eine Referenzpunktmessung durchführen zu können, müssen im Vorfeld einige Vorbereitungen getroffen werden. Dazu werden am „Gebäudeplan“ die einzelnen Referenzpunkte eingetragen. Für diese Vorhaben wird die Software „QuantumGIS“ benutzt. Diese Software kann den georeferenzierten Plan darstellen sowie Punkte erstellen und diese zur weiteren Verarbeitung in die Datenbank übertragen. In Abbildung 9.1 ist dieser Vorgang abgebildet.

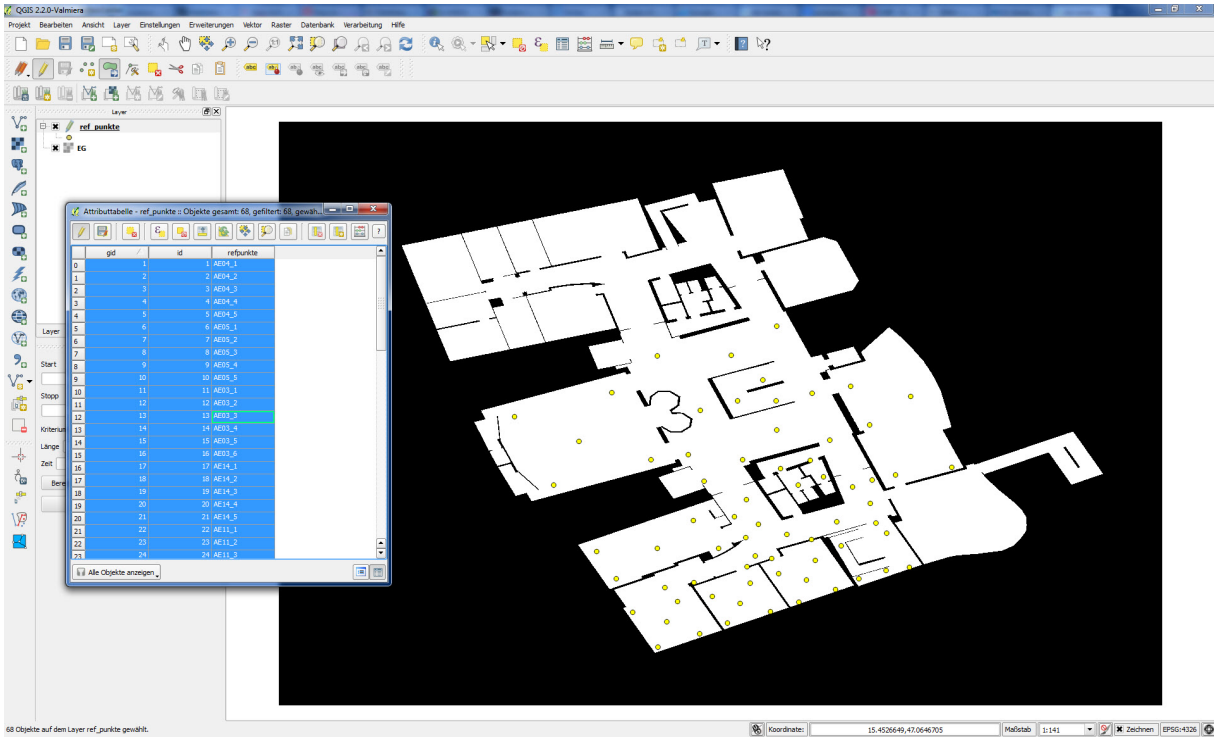


Abbildung 9.1.: Referenzpunkterstellung mittels QuantumGIS

Nach Erstellen der Referenzpunkte auf dem Gebäudeplan wurden die Entfernungen zu den nächstgelegenen Gebäudemauern gemessen und in einer Excel Liste eingetragen. In Abbildung 9.2 wird ein kleiner Ausschnitt der Excel Tabelle gezeigt.

Entfernungen zur nächstgelgenen Wand in Meter

ReferenzPunkt	Nord	Ost	Süd	West
A109_1	1.60	2.20		
A109_2	1.70			2.00
A109_3			2.00	2.00
A109_4		2.40	1.40	
A110_1	1.60	1.60		
A110_2		1.40	1.60	
A111_1		1.30	1.40	
A111_2		1.40	1.30	
A111_3	1.25	1.50		
A111_4	1.40			1.60
A111_5	3.00	5.00	3.60	5.00

Abbildung 9.2.: Excel Liste mit Distanzangaben zur nächstgelegenen Wand

9.1.2. Durchführung der Referenzpunktmessung

Die Durchführung der Referenzpunktmessung findet im Gebäude der Technischen Universität Graz in der Steyrergasse 30 im „Erdgeschoß“ und im „1. Stock“ statt. Es ist eine

„Radio map“ erstellt worden.
Dazu ist folgende Hardware benutzt worden:

Smartphone

Als Gerät wurde das „Samsung Galaxy S4“ verwendet. In Abbildung 9.3 ist das Smartphone abgebildet.

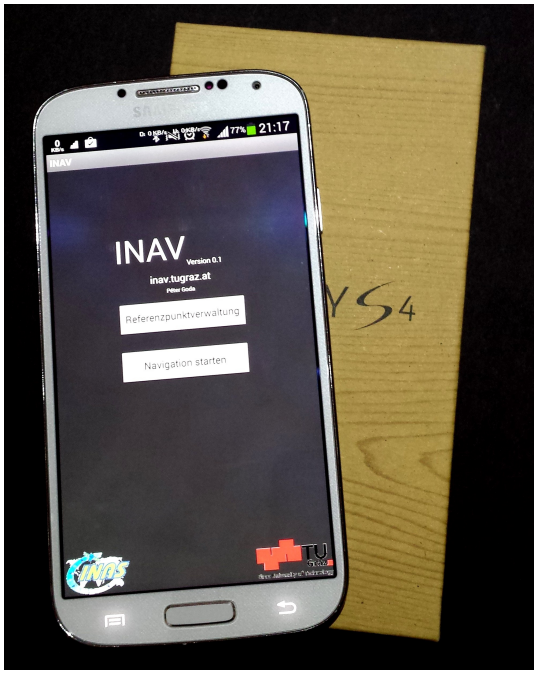


Abbildung 9.3.: Samsung Galaxy S4

Datenblatt:

- *Hersteller* - Samsung
- *Modellname* - Galaxy S4
- *Modellnummer* - GT - I9505
- *Plattform* - Android Phone
- *Plattform Version* - Android 4.3
- *Taktfrequenz* - Quad Core, 1,9 GHz
- *Speicher* - 2 GB RAM und 16 GB interner Speicher

Distanzmessgerät

Zur Messung der Entfernungen der einzelnen Referenzpunkte zu den nächstgelegen Wänden wurde ein Distanzmessgerät, wie in Abbildung 9.4 gezeigt, verwendet.

Abbildung 9.4.: Meister - UM 15 C

Datenblatt:

- *Hersteller* - Meister
- *Modellname* - UM - 15C
- *Messreichweite* - 0,5-15m
- *Genauigkeit* - 0.01m/1m
- *Messmodus* - Ultraschall

WLAN Accesspoints

Das folgende Datenblatt bezieht sich auf die vom ZID installierten Accesspoints. Man kann auch Signale von anderen Accesspoints im Gebäude empfangen, jedoch lässt sich nicht feststellen, um welche Geräte es sich handelt.

Datenblatt:

- *Hersteller* - Cisco
- *Modellname* - Aironet
- *Modellnummer* - 1140
- *Integrierte Antenne* - 2,4GHz mit horizontaler Strahlbreite von 360° und 5GHz mit horizontaler Strahlbreite von 360°
- *Maximale Übertragungsleistung* -
  - *2,4 GHz* - 20dBm
  - *5 GHz* - 20dBm
- *Unterstützte Funkstandards* - 802.11 a/b/g/n
- *Speicher* - 128 MB DRAM und 32MB Flash

Die Messung der Referenzpunkte (Radiomap) erfolgt an einem Tag. In der Offline Phase wurde in vier Himmelsrichtungen gemessen und die Werte gemittelt. Die Messung wurde mit dem Smartphone, das unmittelbar vor der Brust des Benutzers gehalten wurde, durchgeführt. Um die genaue Position des Referenzpunktes bestimmen zu können, wurde mit Hilfe eines digitalen Distanzmessgerätes die Entfernung zu zwei Mauern gemessen und somit die aktuelle Position ermittelt.

9.2. Testen des Webservers

Im nächsten Schritt wird der Webserver getestet. Im Speziellen wird die Ladezeit der Webseite beziehungsweise der Positionsberechnung näher analysiert. Zu diesem Zweck werden auf dem Webserver Zugriffe simuliert, da die Ladezeit von 0,1 oder 0,05 Millisekunden schwer messbar ist. Aus diesem Grund erfolgt die Simulation mit der Software „Apache Benchmark“. Bei der Installation von Apache ist diese Software bereits mit installiert worden. Der Test wird von einem unabhängigen Server aufgerufen, damit die Zugriffszeiten und die Netzwerkauslastung zusätzlich getestet werden können. Unterschiede können erst dann festgestellt werden, wenn zwischen 100 und 1000 Benutzer gleichzeitig auf die Seite zugreifen.
Folgende Möglichkeiten zur Messung stehen zur Auswahl:

- *statischer Content* - man testet eine html-Datei oder ein Bild auf dem Server
- *dynamischer Content, ohne Datenbank* - ein einfacher PHP Aufruf

- *dynamischer Content, mit Datenbank* - ein einfacher PHP Aufruf mit einer Datenbankanfrage

Folgende Optionen sind für den Test mit dem „Apache Benchmark“ möglich und werden in der Konsole eingegeben:

- *„-n“ requests* - Anzahl der Anfragen, die durchgeführt werden sollen
- *„-c“ concurrency* - Anzahl der parallelen Anfragen
- *„-t“ timelimit* - Maximale Wartezeit auf die Antwort in Sekunden
- *„-p“ postfile* - Datei mit den Daten, die per POST gesendet werden sollen
- *„-T“ content-type* - Content-type, der bei einem POST benutzt werden soll
- *„-v“ verbosity* - Stufe für die Ausgabe von Informationen
- *„-w“* - Ergebnis des Tests als HTML-Tabelle ausgeben
- *„-i“* - Verwendet HEAD anstelle von GET
- *„-x“ attributes* - Attribute für die Tabelle
- *„-y“ attributes* - Attribute für die Tabellenzeilen (tr)
- *„-z“ attributes* - Attribute für die Tabellenzellen (td und th)
- *„-C“ attribute* - Cookie hinzufügen, z.B. „Apache=1234“. Kann wiederholt werden.
- *„-H“ attribute* - Fügt zusätzlichen Header in die Anfrage ein, beispielsweise Accept-Encoding: zip
- *„-A“ Authentication username:password* - Benutzer WWW-Authentisierung. Benutzername und Passwort müssen durch einen Doppelpunkt getrennt angegeben werden.
- *„-P“ Proxy Authentication username:password* - Benutze Proxy-Authentisierung. Benutzername und Passwort müssen durch einen Doppelpunkt getrennt angegeben werden.
- *„-V“* - Ausgabe der Versionsnummer
- *„-k“* - Benutze HTTP-KeepAlive
- *„-h“* - Anzeigen von Benutzungsinformationen (diese Tabelle)

[28]

Weitere Informationen und nähere Details zur „Apache Software“ findet man unter `http://httpd.apache.org/docs/programs/ab.html`.

9.2.1. Beispiel für einen Testaufruf

Es wird mit folgenden Befehl ein Testaufruf durchgeführt, wo 1000 Aufrufe mit jeweils 100 parallelen Anfragen an einem URL getestet werden. (siehe Abbildung 9.5)
`ab-n1000-c100http://inav.tugraz.at:8080/service/speedtest.php`

```
c:\wamp\bin\apache\apache2.2.22\bin>ab -k -n 1000 -c 100 http://inav.tugraz.at:8080/service/speedtest.php
This is ApacheBench, Version 2.3 <$Revision: 655654 $>
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/

Benchmarking inav.tugraz.at (be patient)
Completed 100 requests
Completed 200 requests
Completed 300 requests
Completed 400 requests
Completed 500 requests
Completed 600 requests
Completed 700 requests
Completed 800 requests
Completed 900 requests
Completed 1000 requests
Finished 1000 requests


Server Software:        Apache/2.2.22
Server Hostname:        inav.tugraz.at
Server Port:            8080

Document Path:          /service/speedtest.php
Document Length:        14 bytes

Concurrency Level:      100
Time taken for tests:   7.007 seconds
Complete requests:      1000
Failed requests:        0
Write errors:           0
Keep-Alive requests:    1000
Total transferred:      238100 bytes
HTML transferred:       14000 bytes
Requests per second:    142.71 [#/sec] (mean)
Time per request:       700.740 [ms] (mean)
Time per request:       7.007 [ms] (mean, across all concurrent requests)
Transfer rate:          33.18 [Kbytes/sec] received

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:        0    5  14.5      0      80
Processing:    52  400 817.8    127    4779
Waiting:       52  400 817.8    127    4779
Total:         53  405 829.5    127    4825

Percentage of the requests served within a certain time (ms)
  50%    127
  66%    134
  75%    159
  80%    290
  90%   1013
  95%   2463
  98%   3920
  99%   4406
 100%   4825 (longest request)
```

Abbildung 9.5.: einfacher Servertest - 1000 Aufrufe mit 100 parallelen Anfragen

Das Ergebnis dieser Anfrage liefert folgende wichtige Informationen:

- *Complete Requests* - erfolgreich abgearbeitete Anfragen

- *Failed Requests* - fehlgeschlagene Anfragen
- *Requests per second* - Anfragen pro Sekunde
- *Transfer Rate* - Übertragungsrate

Aus diesem Beispiel wird ersichtlich, dass bei 100 parallelen Benutzern der Webserver 142 Anforderungen pro Sekunde bearbeiten kann. Die durchschnittliche Verarbeitungszeit beträgt sieben Millisekunden. Die Übertragungsrate ist 33 KBytes pro Sekunde.

9.2.2. Szenario - einfacher PHP Aufruf

In diesem Abschnitt wird das Verhalten des Webservers mit unterschiedlichen Anzahlen der parallelen Zugriffen evaluiert. Dabei wird eine einfache dynamische PHP Seite aufgerufen.

Auslastung bei zehn parallelen Zugriffen

In Abbildung 9.6 sieht man das Ergebnis der Auslastung des Webservers bei zehn parallelen Zugriffen. Die Zugriffszeit beträgt 5 Millisekunden und 183 Anfragen pro Sekunde können abgearbeitet werden.

```
Document Path:          /service/speedtest.php
Document Length:        14 bytes

Concurrency Level:      10
Time taken for tests:   5.455 seconds
Complete requests:      1000
Failed requests:        0
Write errors:           0
Keep-Alive requests:    994
Total transferred:      237728 bytes
HTML transferred:       14000 bytes
Requests per second:    183.31 [#/sec] (mean)
Time per request:       54.553 [ms] (mean)
Time per request:       5.455 [ms] (mean, across all concurrent requests)
Transfer rate:          42.56 [Kbytes/sec] received
```

Abbildung 9.6.: Ergebnis Servertest - 1000 Aufrufe mit 10 parallelen Anfragen

In Abbildung 9.7 ist die Netzwerkauslastung bei 1000 Aufrufen mit 10 parallelen Anfragen zu sehen. Man sieht eine sehr geringe Beanspruchung des Netzwerkes. Sie liegt unter 0,25 Prozent der maximalen Auslastung.

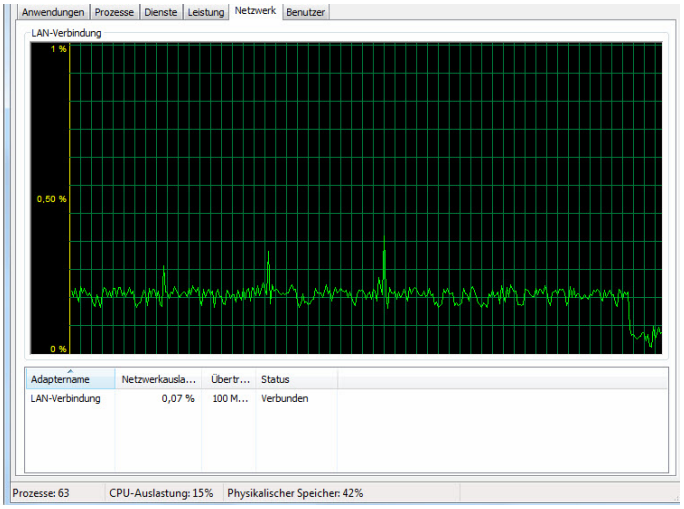


Abbildung 9.7.: Grafische Darstellung der Netzwerkauslastung des Webservers

Auslastung bei 50 parallelen Zugriffen

In Abbildung 9.8 sieht man das Ergebnis der Auslastung des Webservers bei 50 parallelen Zugriffen. Die Zugriffszeit beträgt 4,8 Millisekunden und 205 Anfragen pro Sekunde können abgearbeitet werden.

```
Document Path:          /service/speedtest.php
Document Length:        14 bytes

Concurrency Level:      50
Time taken for tests:   4.857 seconds
Complete requests:      1000
Failed requests:        0
Write errors:           0
Keep-Alive requests:    1000
Total transferred:      238050 bytes
HTML transferred:       14000 bytes
Requests per second:    205.88 [#/sec] (mean)
Time per request:       242.864 [ms] (mean)
Time per request:       4.857 [ms] (mean, across all concurrent requests)
Transfer rate:          47.86 [Kbytes/sec] received
```

Abbildung 9.8.: Ergebnis Servertest - 1000 Aufrufe mit 50 parallelen Anfragen

Auslastung bei 150 parallelen Zugriffen

In Abbildung 9.9 sieht man das Ergebnis der Auslastung des Webservers bei 150 parallelen Zugriffen. Die Zugriffszeit beträgt 17 Millisekunden und 57 Anfragen pro Sekunde können abgearbeitet werden.

```
Document Path:          /service/speedtest.php
Document Length:        14 bytes

Concurrency Level:      150
Time taken for tests:   17.314 seconds
Complete requests:      1000
Failed requests:        107
   (Connect: 0, Receive: 0, Length: 107, Exceptions: 0)
Write errors:           0
Keep-Alive requests:    893
Total transferred:      212791 bytes
HTML transferred:       12502 bytes
Requests per second:    57.76 [#/sec] (mean)
Time per request:       2597.099 [ms] (mean)
Time per request:       17.314 [ms] (mean, across all concurrent requests)
Transfer rate:          12.00 [Kbytes/sec] received
```

Abbildung 9.9.: Ergebnis Servertest - 1000 Aufrufe mit 150 parallelen Anfragen

Zusammenfassend kann man sagen, dass je höher die Anzahl der gleichzeitigen Zugriffen ist, die Anzahl der zu beantwortenden Anfragen pro Sekunde sinkt und auch die Verarbeitungszeit steigt. Deswegen wurde in der Konfiguration des Webservers ein Limit von 150 gleichzeitigen Zugriffen gesetzt. Weiters ist zu erwähnen, dass die Netzwerkauslastung mit Anzahl der parallelen Zugriffe nur gering ansteigt.

9.2.3. Szenario - mit Datenbank Abfrage

In diesem Abschnitt wird das Verhalten des Webservers mit unterschiedlicher Anzahl von parallelen Zugriffen evaluiert. In diesem Fall wird auf die Datenbank zugegriffen, wo ein Referenzpunkt mit den dazugehörigen Messwerten aufgerufen wird. Der Aufruf schaut wie folgt aus: `ab-n1000-c100http://www.inav.tugraz.at:8080/website/referencepoints/view/575`

Auslastung bei zehn parallelen Zugriffen

In Abbildung 9.10 sieht man das Ergebnis der Auslastung des Webservers bei zehn parallelen Zugriffen. Die Zugriffszeit beträgt 186 Millisekunden und 5 Anfragen pro Sekunde können abgearbeitet werden.

```
Document Path:          /website/referencepoints/view/575
Document Length:        58561 bytes

Concurrency Level:      10
Time taken for tests:   186.742 seconds
Complete requests:      1000
Failed requests:        999
   (Connect: 0, Receive: 0, Length: 999, Exceptions: 0)
Write errors:           0
Keep-Alive requests:    997
Total transferred:      59532043 bytes
HTML transferred:       59185189 bytes
Requests per second:    5.35 [#/sec] (mean)
Time per request:       1867.417 [ms] (mean)
Time per request:       186.742 [ms] (mean, across all concurrent requests)
Transfer rate:          311.32 [Kbytes/sec] received
```

Abbildung 9.10.: Datenbankservertest - 1000 Aufrufe mit 10 parallelen Anfragen

Auslastung bei 100 parallelen Zugriffen

In Abbildung 9.11 sieht man das Ergebnis der Auslastung des Webservers bei 100 parallelen Zugriffen. Die Zugriffszeit beträgt 200 Millisekunden und 5 Anfragen pro Sekunde können abgearbeitet werden.

```
Document Path:          /website/referencepoints/view/575
Document Length:        59187 bytes

Concurrency Level:      100
Time taken for tests:   199.873 seconds
Complete requests:      1000
Failed requests:        998
   (Connect: 0, Receive: 0, Length: 998, Exceptions: 0)
Write errors:           0
Non-2xx responses:      11
Keep-Alive requests:    989
Total transferred:      59256915 bytes
HTML transferred:       58908260 bytes
Requests per second:    5.00 [#/sec] (mean)
Time per request:       19987.343 [ms] (mean)
Time per request:       199.873 [ms] (mean, across all concurrent requests)
Transfer rate:          289.52 [Kbytes/sec] received
```

Abbildung 9.11.: Datenbankservertest - 1000 Aufrufe mit 100 parallelen Anfragen

Auslastung bei 150 parallelen Zugriffen

In Abbildung 9.12 sieht man das Ergebnis der Auslastung des Webservers bei 100 parallelen Zugriffen. Die Zugriffszeit beträgt 211 Millisekunden und 4,7 Anfragen pro Sekunde können abgearbeitet werden.

```
Document Path:          /website/referencepoints/view/575
Document Length:        59183 bytes

Concurrency Level:      150
Time taken for tests:   211.650 seconds
Complete requests:      1000
Failed requests:        1020
   (Connect: 0, Receive: 0, Length: 1020, Exceptions: 0)
Write errors:           0
Non-2xx responses:      171
Keep-Alive requests:    862
Total transferred:      59793221 bytes
HTML transferred:       59424758 bytes
Requests per second:    4.72 [#/sec] (mean)
Time per request:       31747.516 [ms] (mean)
Time per request:       211.650 [ms] (mean, across all concurrent requests)
Transfer rate:          275.89 [Kbytes/sec] received
```

Abbildung 9.12.: Datenbankservertest - 1000 Aufrufe mit 150 parallelen Anfragen

Zusammenfassend kann man sagen, dass die Geschwindigkeit des Servers mit der Anzahl der parallelen Zugriffe sinkt. Die Auslastung des Arbeitsspeichers bei 150 gleichzeitigen Zugriffen ist bei 2 GB RAM erreicht. Mit mehr RAM könnten mehr Benutzer zugleich zugreifen, da die Internetübertragungsrate der Technischen Universität es zulässt. In Abbildung 9.13 sieht man die maximale „Download“- und „Upload“ Geschwindigkeit des Servers. Der Speedtest ist mit `http://www.speedtest.net` durchgeführt worden.

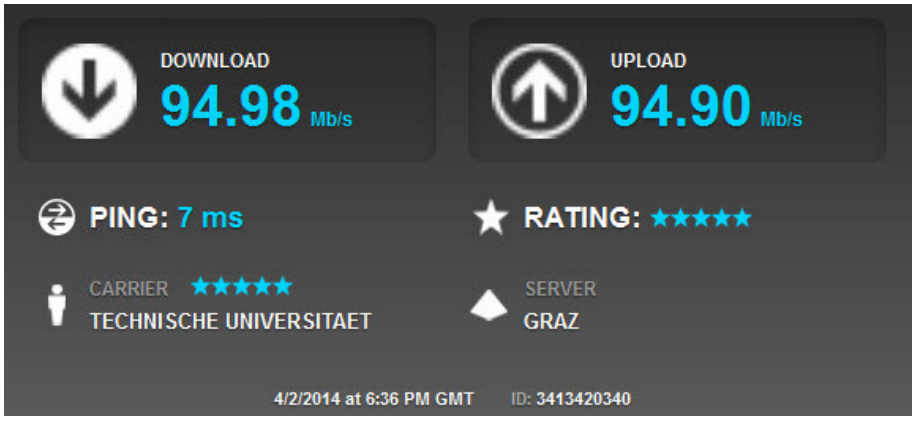


Abbildung 9.13.: Speedtest Server

10. Zusammenfassung

Aufgabenstellung der Masterarbeit war die Entwicklung eines Algorithmus zur Positionsbestimmung in Gebäuden unter Verwendung bereits existierender WLAN-Netze sowie dessen Implementierung auf einem Smartphone. Zu diesem Zweck ist ein „Fingerprinting Algorithmus“ adaptiert worden, welcher die aktuelle Position mit Hilfe zuvor aufgenommener Referenzpunkte bestimmt. Diese sind unter Zuhilfenahme eines Webservers in einer Datenbank abgelegt und werden von einer entwickelten Smartphone-Applikation über eine definierte Schnittstelle zur Lokalisierung abgerufen.
Neben einer grafischen Oberfläche zur Aufnahme der Referenzpunkte sowie zur Datenbankverwaltung kann die aktuelle Position auf einer Karte in Echtzeit dargestellt werden. Abschließend sind die entwickelten Komponenten auf deren Funktionalität getestet worden.

10.1. Ausblick

Um die Indoornavigation mit einen Smartphone zu verbessern besteht die Möglichkeit, zusätzliche verfügbare Sensoren zu nutzen. Beispielsweise können Beschleunigungssensoren die Positionsänderung detektieren oder ein Barometer mögliche Höhenänderungen (Stockwerke) am veränderten Luftdruck erkennen.
Die horizontale Ausrichtung des Smartphones, welche einen Einfluss auf das Messergebnis ausübt, kann durch einen Kompass oder ein Gyroskop bestimmt werden. Durch eine Verbesserung des implementierten Algorithmus zur Positionsbestimmung können beispielsweise vorhergehende Messergebnisse zusätzlich berücksichtigt werden (beispielsweise mit Hilfe eines Kalman-Filters). Aufgrund der steigenden Rechenleistung von Smartphones können unter Zuhilfenahme einer integrierten Kamera bildverarbeitende Methoden zusätzlich herangezogen werden. In Randbereichen von Gebäuden kann Indoornavigation mit herkömmlicher GPS- oder Funkzellenortung kombiniert werden.

Die Untersuchung der Positionsgenauigkeit bei sich verändernder Testumgebung (beispielsweise Variation der Anzahl eingeloggter WLAN-Clients oder unterschiedlicher Anzahl an Personen im Raum) kann Gegenstand zukünftiger Forschungstätigkeiten sein.

A. Anhang - Implementierung einer „Google Maps Karte“

Diese Aufzählung beschreibt die wichtigsten Schritte, zur Implementierung einer „Google Maps V2 Karte“ in eine Android Applikation in der Entwicklungsumgebung „Eclipse“.

A.1. Einbinden von Google Play Service

Google verwendet für die Anwendung der neuen „Google Maps V2 API“ „Google Play Service“. Als Erstes wird über den Android SDK Manager unter „Extras“ „Google Play Service“ installiert. Dieser Vorgang ist in Abbildung A.1 ersichtlich.

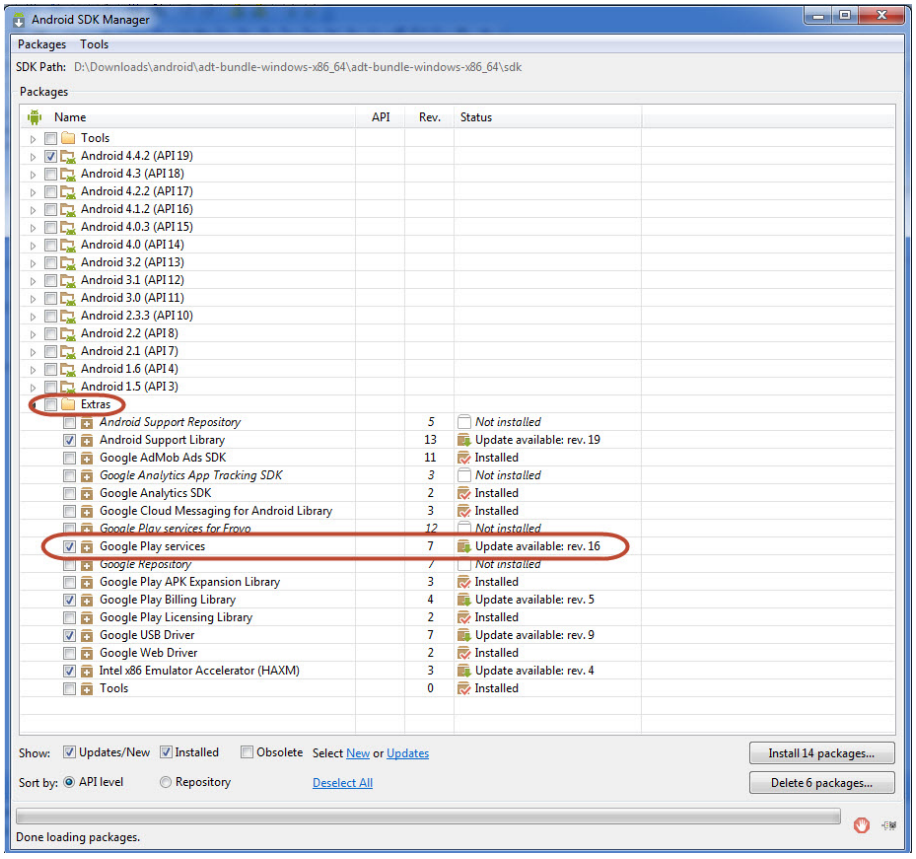


Abbildung A.1.: Download „Google Play Service“

Wenn der Android SDK Manager nicht über „Eclipse“ gestartet worden ist, muss „Google

Play Service“ zu „Workspace“ hinzugefügt werden.

A.2. „Google Maps API Key“

Es gibt zwei Möglichkeiten, um den „Google Maps API Key“ zu bekommen. Als erste Variante dient die „Eingabeaufforderung“ unter Windows beziehungsweise „Terminal“ unter Linux oder Mac OS. Folgender Befehl muss ausgeführt werden:

Windows

keytool -list -v -keystore "%USERPROFILE%\.android\debug.keystore" -alias androiddebugkey -storepass android -keypass android

Linux oder Mac OS

keytool -list -v -keystore ~/.android/debug.keystore -alias androiddebugkey -storepass android -keypass android
Als Ergebnis erhält man den in Abbildung A.2 generierten „SHA 1 Fingerprint“, welcher für jeden Computer einzigartig ist.

```
Aliasname: androiddebugkey
Erstellungsdatum: 14.01.2013
Eintragstyp: PrivateKeyEntry
Zertifikatkettenlönge: 1
Zertifikat[1]:
Eigent³mer: CN=Android Debug, O=Android, C=US
Aussteller: CN=Android Debug, O=Android, C=US
Seriennummer: 6f1dbc3c
G³ltig von: Mon Jan 14 06:31:21 CET 2013 bis: Wed Jan 07 06:31:21 CET 2043
Zertifikat-Fingerprints:
         MD5:  5C:29:23:DE:62:0F:F3:85:FD:46:7C:0E:6A:C9:1E:AF
         SHA1: B0:E8:FB:66:F4:58:77:02:C4:36:7A:B6:B1:35:FF:4B:F9:AD:F9:DA
         SHA256: 49:72:3F:D8:42:28:C8:71:5E:7D:D7:43:E8:A8:1A:B8:32:BE:A4:10:81:21:76:C7:8A:31:B5:70:63:1A:E7:2A
         Signaturalgorithmusname: SHA256withRSA
         Version: 3
```

Abbildung A.2.: „SHA 1 Fingerprint“

Als zweite Variante dient in „Eclipse“ die Einstellungsmaske von **Android - Build**. Zu finden unter **Eclipse** => **Windows** => **Preferences**. Abbildung A.3 zeigt diese Einstellungsmaske.

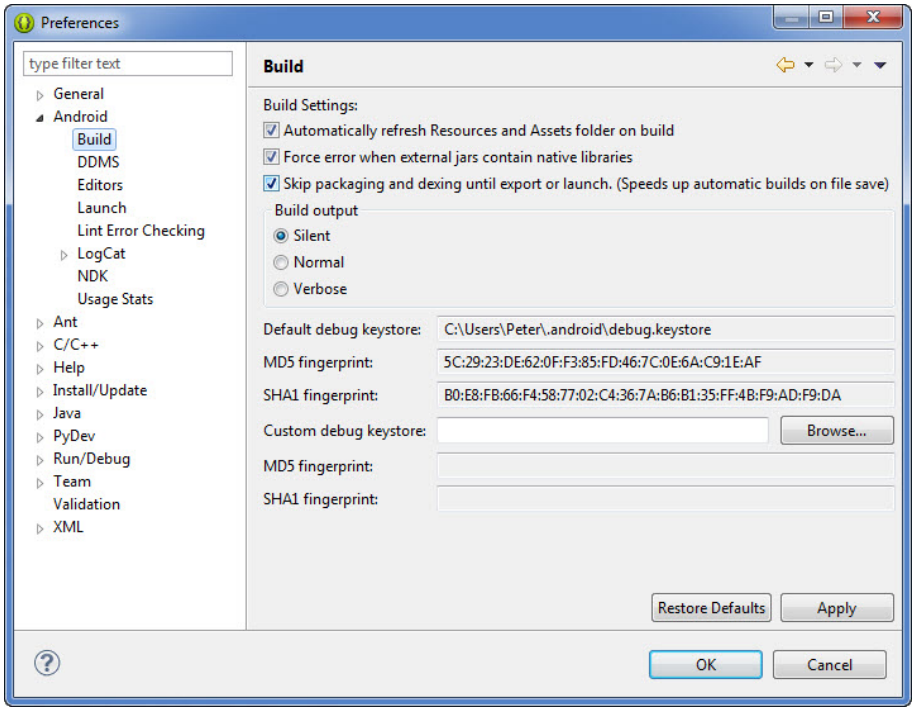


Abbildung A.3.: Eclipse Einstellungen mit „SHA 1 Fingerprint“

A.2.1. Google Maps API

Unter `https://code.google.com/apis/console/` wird die „Google API Console“ aufgerufen. Hier muss ein neues Projekt unter „Projects“ erstellt werden. In weiterer Folge muss zu diesem Projekt anschließend „Google Maps Android API V2“, unter „**APIs & Auth**“, eingeschalten werden. Siehe Abbildung A.4.

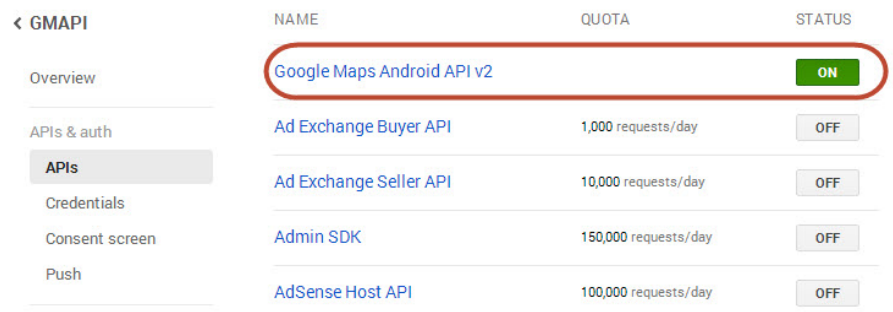


Abbildung A.4.: „Google Maps Android API V2“

Als Nächstes erstellt man unter „**Credentials**“ einen neuen „Public API Access Key“ als „Android Key“. Dazu fügt man den „SHA1 Wert“ in das vorgesehene Feld und trägt weiters den Name, mit einem Semikolon getrennt, des Paketes, in die Android Applikation ein. Ein Beispiel ist in Abbildung A.5 zu sehen.

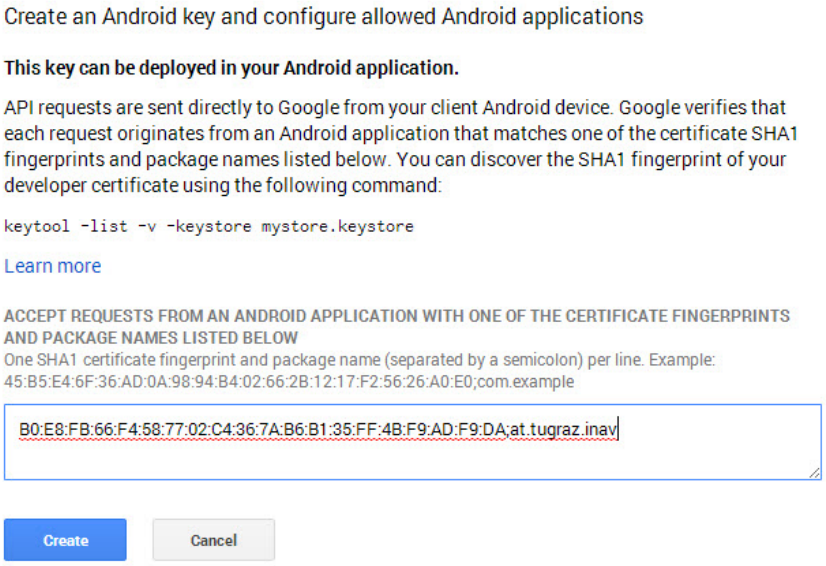


Abbildung A.5.: Erstellen eines „Android Keys“

Abschließend erhält man den „Goggle Maps API Key“. Siehe Abbildung A.6.

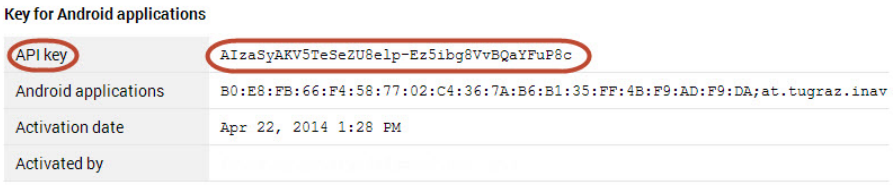


Abbildung A.6.: „Goggle Maps API Key“

Dieser „Key“ ist notwendig, um die Android Applikation mit Google Maps zu autorisieren.

A.3. Android Applikation mit Google Maps

Damit die Android Applikation den „Google Maps API Key“ nutzen kann, muss es im Paket „Google Play Services“ eingetragen werden. Dies geschieht über die Eigenschaften des Paketes (**rechter Mausklick => Properties => Android**). In Abbildung A.7 sieht man auf der rechten Seite unter „Library“ den „add“ Button, mit dem „google-play-services_lib“ auswählt wird.

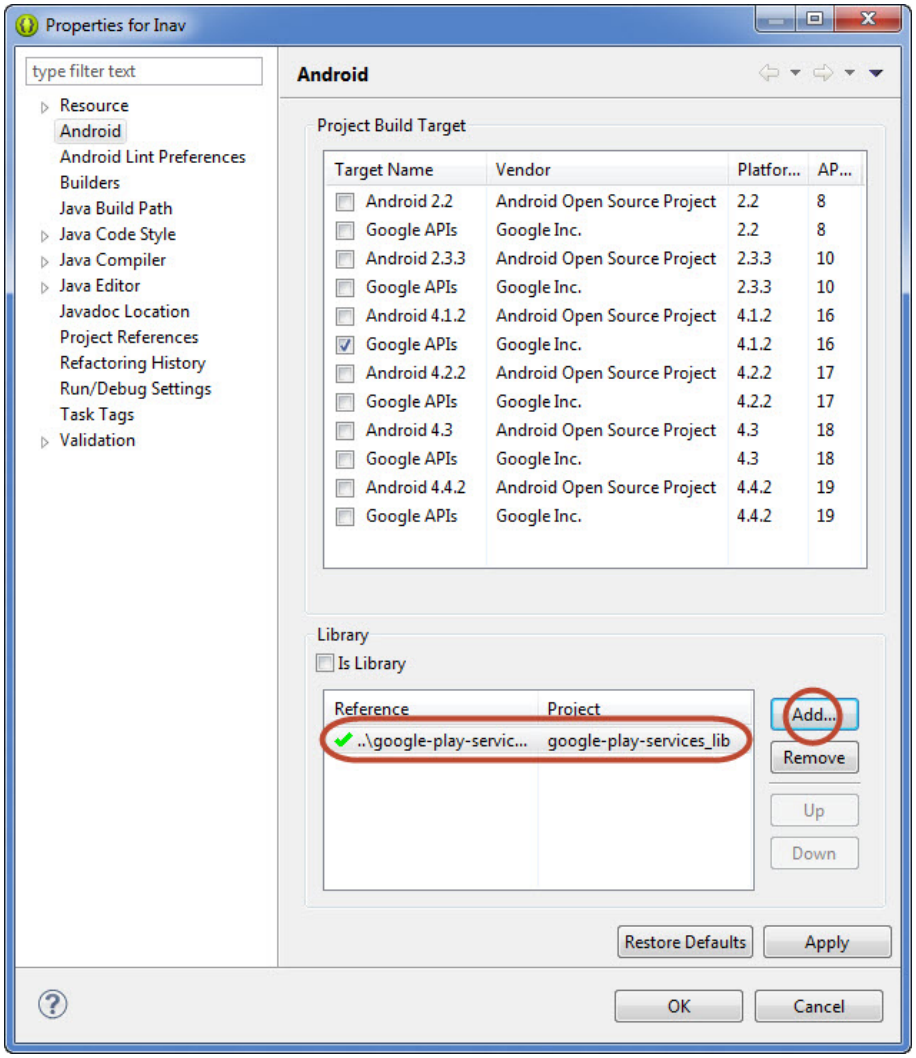


Abbildung A.7.: Eigenschaften des Projektes mit „Google Play Service“

Als nächster Schritt muss der „API Key“ in das Manifest eingetragen werden. Dazu wird **AndroidManifest.xml** geöffnet und folgender Code unter der XML Tag „applikation“ eingetragen:

```
2  <meta-data
3        android:name="com.google.android.maps.v2.API_KEY"
4        android:value="AIzaSyAKV5TeSeZU8elp-Ez5ibg8VvBQaYFuP8c" />
```

Listing A.1: Google Maps API Key

Wobei der Wert von „android:value“ mit dem eigenen Wert der „Google Konsole“ eingesetzt wird.
Weiters benötigt Google Maps folgende Berechtigungen und Features:

- ACCESS_NETWORK_STATE

- INTERNET
- WRITE_EXTERNAL_STORAGE
- ACCESS_COARSE_LOCATION
- ACCESS_FINE_LOCATION
- OpenGL ES V2

Wobei „Open GL ES V2“, wie in Listing 4 gezeigt wird, eingefügt werden muss.

```
1       <uses-feature
2           android:glEsVersion="0x00020000"
3           android:required="true" />
```

Listing A.2: Open GL ES V2

Zu den Permissions muss ebenfalls noch MAPS_RECEIVE hinzugefügt werden. Dies schaut wie folgt aus:

```
1   <permission
2           android:name="at.tugraz.inav.permission.MAPS_RECEIVE"
3           android:protectionLevel="signature" />

5       <uses-permission android:name="at.tugraz.inav.permission.MAPS_RECEIVE" />
```

Listing A.3: MAPS_RECEIVE Permission

Jetzt kann Google Maps in das Layout eingebunden werden. Google Maps benutzt dafür „MapFragments“, welche eine Unterklasse von „Fragments“ ist. In der „Layoutdatei“ wird folgender Beispiel Code für das Anzeigen der Karte implementiert:

```
1       <?xml version="1.0" encoding="utf-8"?>
2   <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
3       android:layout_width="fill_parent"
4       android:layout_height="fill_parent" >

6       <fragment
7           android:id="@+id/map"
8           android:name="com.google.android.gms.maps.MapFragment"
9           android:layout_width="match_parent"
10          android:layout_height="match_parent"/>

12  </RelativeLayout>
```

Listing A.4: Layout Beispiel - Mapfragments

Abschließend wird in einer „Activity“, hier als Beispiel „Karte“ gewählt, die „Google Maps“ Karte wie folgt aufgerufen.

```
1   public class Karte extends Activity {

3       // Google Maps
4       private GoogleMap googleMap;
```

```
6       @Override
7       protected void onCreate(Bundle savedInstanceState) {
8           super.onCreate(savedInstanceState);
9           setContentView(R.layout.activity_main);

11          try {
12              // Laden der Karte
13              initilizeMap();

15          } catch (Exception e) {
16              e.printStackTrace();
17          }

19      }

21      /**
22       * Funktion um die Karte zu laden, wenn es noch nicht geladen wurde, wird diese
23       * Funktion geladen.
24       * */
25      private void initilizeMap() {
26          if (googleMap == null) {
27              googleMap = ((MapFragment) getFragmentManager().findFragmentById(
28                      R.id.map)).getMap();

30              // check if map is created successfully or not
31              if (googleMap == null) {
32                  Toast.makeText(getApplicationContext(),
33                          "Sorry! Unable to create Google Maps", Toast.LENGTH_SHORT)
34                          .show();
35              }
36          }
37      }

39      @Override
40      protected void onResume() {
41          super.onResume();
42          initilizeMap();
43      }

45  }
```

Listing A.5: Beispiel - Android Applikation mit Google Maps

Die Google Maps Karte erscheint beim Aufrufen der Android Applikation. Diese Anleitung bezieht sich auf die Seite `http://www.androidhive.info/2013/08/android-working-with-google-maps-v2/`, wo auch weitere Einstellungen zur Karte zu finden sind. Ebenfalls erhält man unter Google Dokumentation `https://developers.google.com/maps/documentation/android/start#installing_the_google_maps_android_v2_api` weitere vertiefende Informationen.

Literaturverzeichnis

[1] Clemens Strauss: Location Based Services Vorlesung, Institut für Geoinformation, Technische Universität Graz. (2009)

[2] Rech, J.: *Wireless LANs: 802.11-WLAN-Technologie und praktische Umsetzung im Detail ; 802.11a/h, 802.11b, 802.11g, 802.11i, 802.11e.* Heise, 2004. – ISBN 9783936931044

[3] Roth, J.: *Mobile Computing: Grundlagen, Technik, Konzepte.* Dpunkt.Verlag GmbH, 2005 (Dpunkt Lehrbuch). – ISBN 9783898643665

[4] Renaudin, Valerie ; Yalak, Okan ; Tome, Phillip ; Merminod, Bertrand: Indoor Navigation of Emergency Agents. In: *European Journal of Navigation* 5 (2007), Nr. 3, 36–45. http://www.reedbusiness-geo.nl/

[5] Dardari, D. ; Luise, M. ; Falletti, E.: *Satellite and Terrestrial Radio Positioning Techniques: A signal processing perspective.* Elsevier Science, 2011. – ISBN 9780123820853

[6] Liu, Hui ; Darabi, H. ; Banerjee, P. ; Liu, Jing: Survey of Wireless Indoor Positioning Techniques and Systems. In: *Trans. Sys. Man Cyber Part C* 37 (2007), November, Nr. 6, 1067–1080. http://dx.doi.org/10.1109/TSMCC.2007.905750. – DOI 10.1109/TSMCC.2007.905750. – ISSN 1094–6977

[7] Zeimpekis, Vasileios ; Giaglis, George M. ; Lekakos, George: A taxonomy of indoor and outdoor positioning techniques for mobile location services. In: *SIGecom Exchanges* 3 (2003), Nr. 4, 19-27. http://dblp.uni-trier.de/db/journals/sigecom/sigecom3.html#ZeimpekisGL03

[8] Hofmann-Wellenhof, Bernhard ; Lichtenegger, Herbert ; Wasle, Elmar: *GNSS: Global Navigation Satellite Systems: GPS, GLONASS, Galileo & more.* Springer, 2008. – ISBN 9783211730171

[9] Commerce, National O. o. ; Administration, Atmospheric: *NOAA 200th: Transformations: What is the Global Positioning System?* http://celebrating200years.noaa.gov/transformations/gps/side1.html. – Aufruf am 02.05.2014 um 22:12

[10] ESA: *Mission accomplished, GIOVE-B heads into deserved retirement / Navigation / Our Activities / ESA*. http://www.esa.int/Our_Activities/Navigation/Mission_accomplished_GIOVE-B_heads_into_deserved_retirement. – Aufruf am 02.05.2014 um 22:30

[11] Kolodziej, Krzysztof W. ; Hjelm, Johan: *Local Positioning Systems: LBS Applications and Services*. Taylor & Francis, 2006. – ISBN 9780849333491

[12] Mulloni, Alessandro ; Wagner, Daniel ; Barakonyi, Istvan ; Schmalstieg, Dieter: Indoor Positioning and Navigation with Camera Phones. In: *IEEE Pervasive Computing* 8 (2009), April, Nr. 2, 22–31. http://dx.doi.org/10.1109/MPRV.2009.30. – DOI 10.1109/MPRV.2009.30. – ISSN 1536–1268

[13] Wieser, Andreas: High-Sensitivity GNSS: The tr ade-off between availability and accuracy. In: *Proc: 3rd IAG Symposium Geodesy for Geotechnical and Structural Engineering,Austria* (2006)

[14] Goldsmith, A.: *Wireless Communications*. Cambridge University Press, 2005. – ISBN 9780521837163

[15] *WLAN Positioning Systems: Principles and Applications in Location-Based Services*. Cambridge University Press, 2012. – ISBN 9781139503877

[16] Prasad, R.: *OFDM for Wireless Communications Systems*. Artech House, 2004 (Artech House universal personal communications series). – ISBN 9781580537995

[17] Palmer, M.: *Hands-On Networking Fundamentals*. Cengage Learning, 2012. – ISBN 9781285402758

[18] Mok, E. ; Retscher, G.: Location Determination Using WiFi Fingerprinting Versus WiFi Trilateration. In: *J. Locat. Based Serv.* 1 (2007), Juni, Nr. 2, 145–159. http://dx.doi.org/10.1080/17489720701781905. – DOI 10.1080/17489720701781905. – ISSN 1748–9725

[19] Moser, Retscher G.and E. ; Vredeveld, D. ; Heberling, D.: *Performance and Accuracy Test of the WLAN Positioning System ipos*. Papers presented at the 3rd Workshop on Positioning, Navigation and Communication WPNC 2006, 2006

[20] V. Padmanabhan, P. B.: *An In-Building RF-based User Location and Tracking System*. In INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings, 2000., 2000

[21] Martin, Eladio ; Vinyals, Oriol ; Friedland, Gerald ; Bajcsy, Ruzena: Precise indoor localization using smart phones. In: Bimbo, Alberto D. (Hrsg.) ; Chang, Shih-Fu (Hrsg.) ; Smeulders, Arnold W. M. (Hrsg.): *ACM Multimedia*, ACM, 2010. – ISBN 978–1–60558–933–6, 787-790

[22] Honkavirta, Ville ; Perälä, Tommi ; Ali-Löytty, Simo ; Piché, Robert: A Comparative Survey of WLAN Location Fingerprinting Methods. In: *Proceedings of the 6th Workshop on Positioning, Navigation and Communication 2009 (WPNC'09)*, 2009, 243–251

[23] Carl, Denny: *Praxiswissen Ajax*. 1. Beijing [u.a.] : O'Reilly, 2006 (O'Reillys basics). – ISBN 978–3897214514

[24] Lahres, Bernhard ; Raýman, Gregor: *Objektorientierte Programmierung: Das umfassende Handbuch ; [objektorientierte Programmierung verständlich erklärt ; von den Prinzipien über den Entwurf bis zur Umsetzung ; Praxisbeispiele in UML, Java, C♯, C++, JavaScript, Ruby, Python und PHP]*. 2. Bonn : Galileo Press, 2009 (Galileo computing). – ISBN 9783836214018

[25] Netcraft: *Web Server Survey | Netcraft*. `http://news.netcraft.com/archives/category/web-server-survey/`. – Aufruf am 03.03.2014 um 14:30

[26] Scherbaum, A.: *PostgreSQL: Datenbankpraxis für Anwender, Administratoren und Entwickler*. Open Source Press, 2009. – ISBN 9783937514697

[27] GNU: *Verschiedene Lizenzen und Kommentare - GNU-Projekt - Free Software Foundation*. `http://www.gnu.org/licenses/license-list.html/`. – Aufruf am 03.03.2014 um 14:45

[28] Lehrbuch Verlag, TEIA AG Internet A.: *Apache kostenlos lernen:5.4. Leistungstest (Benchmark)*. `http://www.teialehrbuch.de/Kostenlose-Kurse/Apache/15413-Leistungstest.html`. – Aufruf am 18.03.2014 um 16:50

Abbildungsverzeichnis