Location Based Services mit OpenStreetMap Daten

Masterarbeit

Fachhochschule Mainz Fachbereich I

Master-Studiengang Geoinformatik und Vermessung

Pascal Neis
(MatrikelNr.: 505007)

Stand-Nr.: KM0002

Betreuer : Prof. Dr. Alexander Zipf

Bearbeitungszeitraum: 29.02.2008 - 29.08.2008

Danksagung

Mein erster Dank geht an Herrn Prof. Dr. Alexander Zipf für die gute Betreuung während der Durchführung dieser Masterarbeit.

Herrn Wael Alnasrallah danke ich für die Hilfe bei der Einrichtung des Servers, der Research Group Cartography des Geographischen Instituts der Universität Bonn und der OSM Community für die Anregungen und Verbesserungsvorschläge bei der Entwicklung dieser Arbeit.

Mein spezieller Dank gilt meiner Familie und meiner Freundin, ohne deren Unterstützung und Geduld ich sicherlich auch dieses mal nicht wieder so weit gekommen wäre.

Inhaltsverzeichnis

Abkürzungsverzeichnis **iii**

1 Einleitung **1**

1.1 Zielsetzung 2

1.2 Anmerkungen zum Aufbau der Masterarbeit 3

1.3 Motivation 4

2 Location Based Services **7**

2.1 Geschichtliche Entwicklung 7

2.2 Grundprinzip, Kategorien & Marktpotential 8

2.3 Open Geospatial Consortium 11

2.4 OpenGIS Location Services 11

2.5 Web Feature Service & Web Map Service 15

2.6 Erreichbarkeitsanalyse 18

3 OpenStreetMap - The Free Wiki World Map **19**

3.1 Was ist OpenStreetMap? 19

3.2 Überblick der Komponenten & das Datenmodell 23

3.3 Map Features 28

3.4 Editoren, Karten & die Lizenz 29

3.5 Abdeckung & Potential 30

4 Aufbereitung der OSM Daten für ... **32**

4.1 ... OLS Route Service 32

4.2 ... Web Feature Service & Web Map Service 38

4.3 ... Accessibility Analysis Service 40

4.4 ... OLS Location Utility Service 41

4.5 ... OLS Directory Service 43

4.6 Statistiken der Datenaufbereitung 44

5 Verwendete Software **47**

5.1 PostgresSQL 47
5.2 WFS & WMS (GeoServer) 48
5.3 OpenLayers & TileCache 50

6 Weiterentwickelte & neu implementierte Software 53
6.1 OpenGIS Location Services 54
6.1.1 Architektur 55
6.1.2 Route Service 57
6.1.3 Location Utility Service 65
6.1.4 Directory Service 67
6.1.5 Presentation Service 68
6.2 Weitere Dienste 69
6.2.1 Accessibility Analysis Service 69
6.2.2 Emergency Route Service 70
6.2.3 Extended Route Service 71
6.2.4 RESTful OLS 73
6.3 OpenRouteService.org 73

7 Nutzung & Installation 77
7.1 Datenaufbereitungs-Tools 77
7.2 Installation auf einem Server 78

8 Zusammenfassung & Ausblick 82

Literaturverzeichnis I

Abbildungsverzeichnis VI

Listings VIII

Tabellenverzeichnis IX

A Request- & Response Beispiele a
A.1 OLS Route Service . . . . . a
A.2 OLS Location Utility Service . . . . . d
A.3 OLS Directory Service . . . . . f
A.4 Accessibility Analysis Service . . . . . h

B Inhaltsverzeichnis CD j

Eidesstattliche Erklärung A

Abkürzungsverzeichnis

ADT	Abstract Data Type
AOI	Area of Interest
API	Application Programming Interface
DB	Datenbank
DTD	Document Type Definition
EPSG	European Petroleum Survey Group
ERS	Emergency Route Service
ESRI	Environmental Systems Research Institute
GIS	Geoinformationssystem
GML	Geography Markup Language
GPL	General Public License
GUI	Graphical User Interface
HTML	Hyper Text Markup Language
HTTP	Hyper Text Transfer Protocol
KML	Keyhole Markup Language
JAR	Java Archive
JDBC	Java Database Connectivity
JDK	Java Development Kit
JRE	Java Runtime Environment
JTS	Java Topology Suite
JVM	Java Virtual Machine
LBS	Location Based Services
OGC	Open Geospatial Consortium
OL	OpenLayers
OLS	OpenGIS Location Services
OpenLS	OpenGIS Location Services
OSM	OpenStreetMap

ORS	OpenRouteService
OSS	Open-Source Software
PNG	Portable Network Graphics
POI	Point of Interest
RS	Route Service
SLD	Styled Layer Descriptor
SRS	Spatial Reference System
SVG	Scalable Vector Graphics
UML	Unified Modeling Language
URL	Uniform Resource Locator
WFS	Web Feature Service
WFS-T	Web Feature Service Transactional
WMS	Web Map Service
WPS	Web Processing Service
WWW	World Wide Web
XLS	XML for Location Services
XML	Extensible Markup Language
XSD	XML-Schemata

1 Einleitung

Wer kennt dies nicht:
Nächstes Wochenende soll es in eine fremde Stadt gehen und man kennt sich dort überhaupt nicht aus!

Durch das *World Wide Web* (WWW) oder die immer größerer Verbreitung von Navigationsgeräten, nutzen heutzutage schon viele Geodaten und *Location Based Services* (LBS). Anwendungen wie Google Maps/Earth, Preissenkungen bei den tragbaren Weggehilfen oder immer leistungsfähigerer Mobiltelefone haben ihr übriges dazu beigetragen. Auch erfolgt teilweise die Nutzung der Daten und Dienste, ohne das sich die Nutzer deren überhaupt bewusst sind. Es wird über das Internet mit einem beliebigen Routensuchdienst schnell mal eine Route vom Wohnort zum nächsten Ausflug geplant und dabei noch ein passendes Hotel gesucht. Über die Telefonauskunft wird nach dem nächstmöglichen Pizza-Restaurant in der Umgebung gefragt oder mit dem tragbaren Navigationsgerät über die Adresse die Anfahrt zum Freund koordiniert. Hinter all diesen und noch weiteren Anwendungen oder Services stecken Geodaten und Location Based Services. Dabei werden Geodaten unter anderem in Karten, bei der Routenplanung oder in Branchenverzeichnissen verwendet.

Nicht nur die genutzten Anwendungen spiegeln sich im Ergebnis der Suche wieder, sondern auch die Qualität der Geodaten haben eine wichtige Rolle. Die Geodaten, also die digitalen Informationen, können in ihrem Umfang, ihrer Vollständigkeit, der Aktualität und Genauigkeit das Resultat beeinflussen. Im Gegensatz zu den USA, wo zumindest der Grunddatenbestand von Geodaten frei zugänglich ist, haben beziehungsweise hatten wir es bisher in Europa schwer, an freie Geodaten heranzukommen. Allerdings ist die Qualität der amerikanischen freien Geodaten nicht mit denen der öffentlichen Verwaltung in Deutschland vergleichbar. Die vier Faktoren: Umfang, Vollständigkeit, Aktualität und Genauigkeit der Daten, finden sich jedoch auch im Preis wieder, wenn es darum geht, die Daten für ein mögliches Projekt zu erwerben. Dabei fallen Kosten für den Kauf und die möglichen weiteren Verwendungen, wie zum Beispiel Internet-Nutzung, an.

Im Jahre 2004 startete das Projekt „*OpenStreetMap (OSM) - The Free Wiki World Map*“ mit dem Ziel, eine frei verfügbare Karte der ganzen Welt zu erstellen. Es kann jeder dazu beitragen, die Karte zu vervollständigen oder zu korrigieren. Das somit entstehende Kartenmaterial, beziehungsweise die somit entstehenden Geodaten, dürfen - entsprechend einer Lizenz - in jeglicher Form weiterverwendet und veröffentlicht werden. Die Weiterverwendung und Veröffentlichung und der Versuch eine Karte der ganzen Welt zusammenzutragen, sprechen für eine genauere Untersuchung der Möglichkeiten von OSM Daten. Der Titel dieser Masterarbeit lautet: ***„Location Based Services mit OpenStreetMap Daten“***

1.1 Zielsetzung

Das Ziel der Masterarbeit kann in zwei Teile gegliedert werden:

- Evaluierung und Aufbereitung von OpenStreetMap Daten für LBSs
- Weiterentwicklung und Einrichtung von LBSs mit Hilfe der OpenStreetMap Daten

Die Evaluierung soll dabei zeigen, welche OpenStreetMap Daten genutzt, wie sie vielleicht bearbeitet, gefiltert und oder erweitert werden müssten, damit eine Nutzung derer in Ortbezogenen-Diensten realisiert werden kann. Die Weiterentwicklung und Einrichtung der Anwendungen bezieht sich in erster Linie auf den von Neis (2006) implementierten *OpenGIS Location Service* (OLS) *Route Service* (RS), der bereits in verschiedenen Projekten Anwendung findet. Bisher musste der OLS RS nur mit Datenmengen in Größen einer Stadt wie Heidelberg, Osnabrück und Hamburg oder im Einzelfall auch mal mit denen eines Bundeslandes wie Rheinland-Pfalz zurechtkommen. Durch die OpenstreetMap könnten somit größere Tests durchgeführt werden, in denen sich eventuelle Probleme bei der Implementierung lokalisieren lassen würden, die unter Verwendung von „kleineren“ Datensätzen nicht oder nicht spürbar auftreten. Beispielsweise könnten genauere Aussagen getroffen werden, ob bei der Datenhaltung des Routing-Graphen oder beim Routing selbst Optimierungen angebracht werden müssen. Ein weiteres Ziel ist die Installation des Route Service auf einem Server und die Möglichkeit der Nutzung über eine zu entwickelnde Webseite. Als Ergänzung zu dem OLS RS soll noch versucht werden, einen *Open Geospatial Consortium* (OGC) konformen Kartendienst (*Web Mapping Service* (WMS) inklusive *Web Feature Service* (WFS)) mit OSM Daten einzurichten und ihn ebenfalls über die Webseite nutzbar zu machen.

1.2 Anmerkungen zum Aufbau der Masterarbeit

Die Masterarbeit gliedert sich in einen Hauptteil mit acht Kapiteln und einen Anhang aus zwei Teilen.

Kapitel 1.: Einleitung - Dieses Kapitel

Kapitel 2.: Location Based Services - Was verbirgt sich hinter LBS, welche Möglichkeiten und Spezifikationen bieten sie?

Kapitel 3.: OpenStreetMap - Vorstellung des OpenStreetMap Projektes

Kapitel 4.: Aufbereitung der OSM Daten für ... - Wie müssen OpenStreetMap Daten aufbereitet werden, um sie am Beispiel von OLS nutzen zu können?

Kapitel 5.: Verwendete Software - Erläuterung der verwendeten Software

Kapitel 6.: Weiterentwickelte & neu implementierte Software - Beschreibung der Änderungen an der bestehenden und neu implementierten Software

Kapitel 7.: Nutzung & Installation - Beschreibung, wie die entwickelten Tools für die Datenaufbereitung genutzt und die entstandenen Services auf einem Server installiert werden können

Kapitel 8.: Zusammenfassung & Ausblick - Abschließende Betrachtung und mögliche Weiterentwicklungen der Arbeit

A Request & Response Beispiel: Beispiele für Serviceanfragen und -antworten

B Inhaltsverzeichnis CD

1.3 Motivation

Wie bereits in der Einleitung beschrieben, war es bis heute in Europa schwierig an freie Geodaten - Geoinformationen - heranzukommen. Doch durch das zunehmende Aufkommen des Web 2.0 (O'Reilly, 2005), dem sogenannten „Mitmach-Web", entstehen vermehrt raumbezogene oder verortbare Daten. Diese werden zum Beispiel über Mash-Ups (Verknüpfungen oder Kombinationen von Daten) mit anderen interaktiven Inhalten in Anwendungen wie Google Earth[1], Google Maps[2] etc. verbunden und frei online gestellt (Zipf, 2007). Gerade diese beiden genannten Anwendungen unterliegen aber strengen Lizenzbedindungen, so dürfen zum Beispiel die Ergebnisse von Mash-Ups oder die einer Routenplanung nicht kommerziell verwendet werden. Auch derjenige, der die *Application Programming Interface* (API) von Google Maps für eine eigene Anwendung auf der Website nutzen möchte, kommt nicht um eine Registrierung und Akzeptierung der Allgemeinen Geschäftsbedingungen herum. Abgesehen von diesen oder auch anderen Lizenzbedingungen ist es heute bereits trotzdem möglich, das auch nicht-Programmier- oder nicht-Softwareexperten solche APIs nutzen und ihre eigenen Daten im WWW veröffentlichen können.

Nutzung von freien Geodaten

Seit den letzten Jahren, bedingt durch preiswerte und handliche GPS-Empfänger und eine immer besser werdende Abdeckung von Breitbandinternet, passiert etwas Besonderes im WWW: Geodaten werden durch Gemeinschaftsarbeit (collaborative effort) erfasst und im Projekt OpenStreetMap zur Verfügung gestellt. Über die Entstehung, den Wachstum und vor allem den Inhalt wird später im Kapitel *3 OpenStreetMap* ausführlicher eingegangen. Das die, durch eine Gemeinschaftsarbeit von Nutzern, gesammelten Geodaten eine viel versprechende und innovative Grundlage für raumbezogene Anwendungen bilden könnte, demonstrierten auch Holone, Misund und Holmstedt (2007). Zwischen der Gründung von OSM im Jahre 2004 und dem eigentlich Start 2006, zeigten unter anderem Zipf und Tschirner (2005), wie ein Portal zum Austausch von freien Geodaten aussehen könnte. Das allerdings OSM oder ähnliche Projekte eine so große Akzeptanz und Entwicklung nehmen könnte, konnten wahrscheinlich nur wenige vorhersehen.

Natürlich kann aber auch die Frage gestellt werden: *Brauchen wir überhaupt OSM, es gibt doch Google?* Es gibt mehrere Vorteile, die für OSM sprechen: Google-Anwendungen, oder auch Anwendungen anderer Kartenanbieter dürfen zwar größtenteils kostenlos genutzt werden, deren Karten zu vervielfältigen, ist aber in den meisten Fällen nicht erlaubt. Auch sind die Geodaten der Karte bei Google nicht „frei" erhältlich. OpenStreetMap

[1] Google Earth - http://earth.google.de
[2] Google Maps - http://maps.google.de

dagegen „bietet auch die „rohen“ Geodaten an, damit jeder sie so nutzen kann, wie er möchte.“[3] Speziell letztere Gründe sprechen für eine Evaluierung und den Versuch mit diesen Daten verschiedene Web Service aufzubauen.

Noch ein Routenplaner im Internet?

Routenplaner gibt es viele im Internet (map24, Falk oder Google Maps). In der Regel unterliegen diese aber strengen Nutzungsbedingungen, die zum Beispiel untersagen, das Nutzer keine, oder nicht ohne weiteres, Abbildungen oder Karten an andere weitergeben dürfen. Mit dieser Arbeit soll versucht werden, mit Hilfe der Daten des OpenStreetMap Projektes, einen Routenplaner einzurichten, der nicht solchen Nutzungsbedingungen unterliegt. Dabei muss der vorhandene Routenplaner (Neis, 2006) gegebenenfalls weiterentwickelt werden, um mit den OSM Daten und auch der erhöhten Datenmenge zurecht zukommen. Und ihn möglicherweise auch über das Internet nutzbar zu machen.

Zu Beginn dieser Masterarbeit war nur eine Website auffindbar, die einen Routenplaner[4] mit Hilfe von OSM Daten, anbot. Dieser Routenplaner ist allerdings in seinen Fähigkeiten sehr eingeschränkt. Er bietet nur Routing für Freiburg an, hat keine Unterscheidung bei der Routenplanungs-Eigenschaft, wie zum Beispiel Auto oder Fussgänger, beachtet keine Einbahnstraßen und der Nutzer erhält nach Berechnung einer Route auch keine Fahranweisungen. Auch ist die Angabe des Start- und Zielpunktes nur über das Setzen des jeweiligen Punktes in der Karte möglich. Eine Schnittstelle, um den Service von anderen Anwendungen nutzen zu können, ist auf der Website nicht veröffentlicht.
Im Laufe dieser Masterarbeit kamen noch zwei Webseiten hinzu. Eine davon ist das „Projekt II - GIS: Freies Online Routing“[5] der FH Oldenburg. Ähnlich wie die vorherige Webseite, wurde auch diese von Studenten erstellt. Sie bietet im Gegensatz zu der ersten deutschlandweites Routing an. Wichtig ist jedoch zu erwähnen, dass auch hierbei nicht alle Straßen von OSM, die für das Routing vorgesehen sind, genutzt werden. Bei „OpenStreetMap routing service demo“[6] kann hingegen die Fortbewegungsart (Auto, Fussgänger oder Fahrrad) ausgewählt werden. Diese Seite ermöglicht allerdings nur das Routing in den Niederlanden.
Alle eben vorgestellten Webseiten visualisieren zwar die ermittelte Route in der OSM Karte, die daraus resultierenden Fahranweisungen bietet allerdings keine an. Des Weiteren ist keiner von ihnen über die OGC OLS Route Service Spezifikation nutzbar.

[3] OSM FAQs: Fragen und Antworten - http://www.openstreetmap.de/faq.html
[4] Routing on Openstreetmap 0.4 - University Freiburg, online unter: http://www.osm-routing.de
[5] http://fbvpc066.fh-oldenburg.de/~projektgis/ProjektGIS/start.html
[6] http://tile.openstreetmap.nl/~lambertus/routing/

Neben den Anwendungen, die aus OSM Daten Karten erstellen können (Osmarender[7] oder Mapnik[8]), gibt es den OSM Namefinder[9]. Er ist ein Web Service, dessen Funktionalitäten mit einen Geocoder/Reverse-Geocoder und einer Art Branchenbuch, beides in einer einfachen Form, verglichen werden kann. Die verschiedenen Services, um die OSM Karte auf einer eigenen Website einbinden zu können, oder auch der Namefinder nutzen keine standardisierten Schnittstellen, beispielsweise die des *Open Geospatial Consortium* (OGC). Warum OSM OGC Standards nicht nutzt, wird wie folgt erläutert: *OpenStreetMap begann mit Hilfe des einfachsten Ansatzes Karten zu generieren. Dabei verfolgten sie für die Eingabe und Pflege der Daten den Wiki-Ansatz. Weil die standardisierten Anwendungen dies nicht nach den Wünschen von OpenStreetMap unterstützten, entschieden sie sich gegen eine Nutzung von OGC Standards.*[10] Nichtsdestotrotz sind OGC Standards in der heutigen Zeit zu wichtig geworden, vor allem wegen der Interoperabilität. Mit Hilfe von OGC Anwendungsschnittstellen können auch OSM Daten verfügbar und nutzbar gemacht werden. Beides soll, neben der Aufbereitung der OpenStreetMap Daten, mit dieser Arbeit versucht werden.

[7] http://wiki.openstreetmap.org/index.php/De:Osmarender
[8] http://wiki.openstreetmap.org/index.php/De:Mapnik
[9] http://gazetteer.openstreetmap.org/namefinder/
[10] http://wiki.openstreetmap.org/index.php/FAQ

2 Location Based Services

In diesem Kapitel werden Anfangs *Location Based Services* (LBS) vorgestellt. Danach wird die *OpenGIS Location Services* (OLS) Spezifikation des OGC mit ihren Schnittstellen-Definitionen für Ortsbezogene-Dienste erläutert. Am Ende werden weitere Dienste beschrieben, die im Kontext mit LBS von Interesse sind.

2.1 Geschichtliche Entwicklung

Positions-Dienste (Location Services) oder Positionsbestimmungs-Dienste haben eine lange Tradition. Bereits vor vielen Jahren waren sie zum Beispiel für die Schifffahrt von essentieller Bedeutung. Die wohl unerlässlichste Fragestellung war: *„Wo bin ich?“*. An dieser wichtigen Frage hat sich bis heute nicht viel verändert, außer das noch weitere Fragen bzgl. der Position hinzugekommen sind. Damals wurde die Position noch über die Sterne bestimmt und in eine Karte übertragen.
Mit der Entwicklung eines *Global Positioning System* (GPS) während der 70er Jahre, wo eine Position mit Hilfe von Satelliten bestimmt werden konnte, änderte sich aber einiges. Das damalige GPS wurde anfänglich vom US-Verteidigungsministerium für militärische Zwecke erschaffen. 1995 wurde die volle Funktionsfähigkeit des US-GPS bekannt gegeben. Richtig nutzbar wurde GPS aber erst im Jahre 2000, als die künstliche Verschlechterung der Positionsbestimmung abgeschaltet wurde. Danach konnte eine Position mit einer Genauigkeit von +/-10 Metern bestimmt werden. Bedingt durch die „Verschlechterung“ des Signals lag die Genauigkeit zuvor bei +/-100 Metern.[1] Richtige Ortsbezogene-Dienste (Location Based Services) und die zu grundeliegende Technologie entstanden erst Ende der 90er Jahre mit dem Wachstum der Mobilfunktnetze und dem Interesse deren Betreiber (Schiller & Voisard, 2004).

Aber: ***Was genau sind eigentlich Location Based Service?***
Virrantaus, Markkula, Garmash und Terziyan (2001) beschreiben Location Based Service als erreichbare Dienste, die Informationen für mobile Geräte über das Mobilnetz und unter Nutzung der Position des mobilen Gerätes zur Verfügung stellen. Raper, Gartner,

[1] Mehr Informationen beispielsweise unter: http://de.wikipedia.org/wiki/Global_Positioning_System

Karimi und Rizos (2007) definieren LBS auf einem höheren Level der Abstraktion als Computeranwendungen, die Informationen abhängig von der Position des Gerätes und Nutzers, übermitteln. Nach Brimicombe (2002) sind LBSs eine Schnittmenge von drei Technologien. Wie in der Abbildung 2.1 zu sehen ist, sind diese Technologien: das *Internet*, *Geographische Informations Systeme* (GIS) mit räumlichen Datenbanken und die *New Information and Communication Technologies* (NICTS), wie mobile Telekommunikationssysteme und tragbare Geräte.

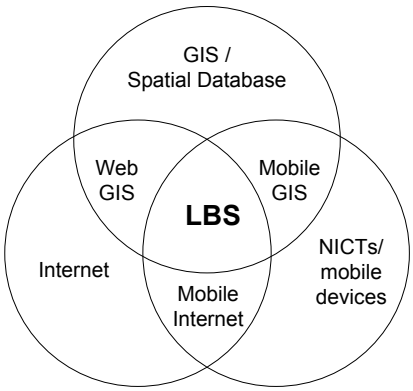


Abbildung 2.1: LBS als eine Verschneidung von Technologien (Brimicombe, 2002)

Einfache Beispiele für LBS sind, wie bereits erwähnt, Positions- oder Navigationsdienste und Branchenverzeichnisse. Eine Anfrage an solche Dienste könnte wie folgt aussehen:

- „Wo bin ich?“
- „Wo finde ich im Umkreis von 2 km das nächste Restaurant?“
- „Wo ist die Goetheallee?“
- „Wie komme ich von meiner Position dorthin?“
- „Zeige mir das auf der Karte!“

2.2 Grundprinzip, Kategorien & Marktpotential

Das Grundprinzip, wie ortsbezogene Dienste funktionieren, ist in Abbildung 2.2 zu sehen. Genutzt werden können sie von den unterschiedlichsten Geräten, zum Beispiel von einem Mobiltelefon, PDA, Notebook oder PC. Generell benötigen die Geräte dabei einen Netzwerkanschluß und Positionstechnologien, wovon beide wiederum eine Infrastruktur mit geringer Wartezeit für eine größere Zahl von Nutzern bereitstellen müssen (Raper et al., 2007). Bei einer Anfrage an einen Location Service muss in der Regel die Position oder die Adresse des Nutzers mit übergeben werden, um eine Karte von der Umgebung

erstellen oder eine Umkreissuche ausführen zu können. Die Anfragen werden von den Diensten unter Verwendung von unterschiedlichen GeoDaten und sonstigen Informationen bearbeitet. Das Ergebnis wird anschließend an den Nutzer zurück gesendet (siehe Abbildung 2.2).

Abbildung 2.2: Grundprinzip von Location Based Services (vereinfacht)

Neben der eben erwähnten sogenannten *Service-Oriented Architektur* (SOA) von LBSs können auch Stand-Alone Geräte für die Nutzung von LBSs zum Einsatz kommen, diese enthalten dann die Informations-Architektur. Auf dem Gerät sind somit das Positionierungssystem (zum Beispiel GPS) und die GeoDaten für Karten oder Umkreissuchen (zum Beispiel auf einer Speicherkarte) vorhanden und können ohne weitere Dienste direkt verwendet werden.

LBSs können in verschiedene **Kategorien** eingeteilt und genutzt werden. In der Regel werden LBSs unter Verwendung der Position in Anspruch genommen. Ebenfalls ist sie in den meisten Fällen für die eigentliche Nutzung der LBS Kategorien, in der folgenden Aufzählung aus Fritsch und Muntermann (2005), notwendig.

- ***Navigation:*** *Navigationsdienste:* Navigationsdienste per Handy. Vertreter der interaktiven Informationsdienste; *Tourismus:* Freizeitdienst für nicht-alltägliche Umgebungen.
- ***Unterhaltung:*** *Spiele*: Mobile Spiele mit Ortskomponente zur Freizeitgestaltung.
- ***Informationsdienste:*** *Kulturinformation:* Informationsdienst zur Freizeitgestaltung. *Finanzdienste:* ortsabhängige Finanzinformationen und -dienstleistungen.
- ***Sicherheits- und Notfalldienste:*** *Medizinische Notdienste:* Ortungsdienste zur Bewältigung medizinischer Notlagen; *Notfalldienste:* Notrufortung, Katastrophenschutz und andere Dienste zum Schutz von Leib und Leben bei drohender Gefahr als hoheitlicher Bürgerservice; *Strafverfolgung:* Ortungsdienste zur Erleichterung der Strafverfolgung.
- ***Community-Dienste:*** *Freunde finden:* Freizeitdienst mit hoher sozialer Bindung; *Dating:* Ortsbasierte, mobile Partnervermittlung zur Freizeit- & Lebensgestaltung.

Wie bereits erwähnt, kommen für die Nutzung von LBSs unterschiedliche Geräte zum Einsatz. Eine wichtige Rolle kann dabei die Bestimmung der Position spielen. Bei der Er-

mittlung können verschiedene Technologien verwendet werden (aus Raper et al. (2007)): Mobilfunksignale oder eine Reihe von Kurzstrecken-Kommunikationen wie WiFi, Bluetooth, Zigbee, UWB, RFID und so weiter. Die wichtigste und auch meist verwendete ist aber bis heute immer noch GPS. Es ist weltweit, 24 Stunden und sieben Tage die Woche, kostenlos nutzbar. Im Bezug auf die Genauigkeit der Positionsbestimmung liegen hier aber auch die Grenzen bei LBSs. Reichen für Autofahrer die über GPS bestimmten Koordinaten aus, kann es bei Fußgängern zu einigen Problemen kommen. Die Möglichkeit eine Position zu bestimmen, ist aber wichtiger, als die Genauigkeit (Raper et al., 2007).

Marktpotential & Aussichten

Zur Jahrtausendwende und mit der Einführung von Mobilen Mehrwertdiensten wurde von Durlacher Research Ltd. (1999) prognostiziert, das der Umsatz mit Location Based Services in Europa für das Jahr 2003 bei ca. 3,5 Mrd. Euro liegen werde. Diese Vorhersage wurde jedoch in den folgenden Jahren stark nach unten korrigiert. Die veröffentlichte Studie von der Marktforschungsgesellschaft Strategy Analytics erwartete für das Jahr 2003 dann lediglich nur noch einen weltweiten Umsatz von 1,1 Mrd. US$ durch LBS (Europa 0,13 Mrd. US$) (Strategy Analytics, 2003). Im Jahr 2006 gaben knapp 20 Millionen Nutzer rund 640 Millionen Euro für LBS aus. Die immer wieder nach unten korrigierten Zahlen belegen, dass die Erwartungen von den Gewinnen und Umsätzen durch LBS weit unter den Vorstellungen geblieben sind. Laut Fritsch und Muntermann (2005) dürfte die Nicht-Verwendung der Dienste am fehlenden Datenschutz, an Privatsphärenbedenken und an der unsicheren Technik liegen. Aber auch das zum Beispiel viele Leute nicht mit den Geräten oder den Anwendungen umgehen können oder deren Nutzen einfach noch nicht erkannt haben können erschwert die Akzeptanz von Location Based Service und kann unter Umständen noch eine Weile in Anspruch nehmen. Ergänzend zählen diese Gründe auch zu den Risiken von LBSs. „Bis 2011 soll die Zahl der Benutzer jedoch, laut Strategy Analytics, weltweit auf 95 Millionen klettern und dabei einen Umsatz von 4,3 Milliarden Euro generieren.“(YellowMap, 2008)

2.3 Open Geospatial Consortium

Im Zusammenhang von Location Based Services stößt man in der Regel immer wieder mal auf den Begriff: OGC. Dies ist die Abkürzung für *Open Geospatial Consortium* (früher: Open GIS Consortium). Das OGC ist ein internationales Industrie-Konsortium von Unternehmen, Behörden und Universitäten. 1994 von 20 Mitgliedern gegründet, sind es heute weit mehr als 360 Mitglieder. Sie verfolgen das Ziel, die Entwicklung von raumbezogener Informationsverarbeitung (insbesondere Geodaten) auf der Basis allgemeingültiger und frei verfügbarer Standards zum Zweck der Interoperabilität festzulegen. Diese Standards sind „Spezifikationen, die von abstrakten Beschreibungen des Aufbaus, der Komponenten und der Funktionsweise eines dienstebasierten GIS im Sinne des OGC bis hin zu detaillierten Spezifikationen der Implementation dieser Dienste reichen. Hierbei wird jedoch nicht die konkrete Umsetzung der Software vorgeschrieben, sondern die verschiedenen Schnittstellen eines Dienstes, deren Eigenschaften und Verhalten festgelegt. Der Weg zu diesen Spezifikationen läuft über einen langen Diskussionsprozess im OGC, dessen Ergebnis schließlich in einer Spezifikation resultiert.“[2] Wichtig und kritisiert werden muss: Dieser Prozess findet unter Ausschluß der Öffentlichkeit statt.

2.4 OpenGIS Location Services

OpenGIS Location Services (OpenLS oder OLS) ist eine im Jahr 2000 ins Leben gerufene Initiative des OGC. Sie hat sich der Entwicklung und Beschreibung von Schnittstellen und Protokollen gewidmet, um Location Based Services zu standardisieren. Das Ziel dieser Initiative ist es, freie Spezifikationen für interoperable Location Based Services zu erarbeiten und Entwicklern zur Verfügung zu stellen. Bei der Erstellung der OpenLS Spezifikation wurden bereits bestehende Spezifikationen und Infrastrukturen von Telekommunikations- und Internetservices beachtet und mit einbezogen (Neis, 2006). Die Version 1.0 der Spezifikation erschien im Januar 2004. Im Mai 2005 folgte Version 1.1 mit kleineren Überarbeitungen an den Service Schemata und ohne Änderungen an der Anzahl der Services oder deren Aufgabe. Die Version 1.2 der Spezifikation soll noch in diesem Jahr erscheinen. Als weiterer Dienst im OLS Kontext wird der Tracking Service hinzukommen. Vor der 1.0 Version der Spezifikation war noch ein weiterer Dienst im Gespräch, ein sogenannter: Navigation Service. In der Version 1.3, an der bereits gearbeitet wird, soll dieser Service als Erweiterung des Route Service mit einfließen.
Die OLS Spezifikation enthält somit mehrere Beschreibungen für LBS. Sie können als

[2] Open Geospatial Consortium aus Wikipedia - http://de.wikipedia.org/wiki/Open_Geospatial_Consortium

eine detaillierte technische Beschreibung der Schnittstellen bezeichnet werden oder auch als Bauplan für OLS konforme Web Services. In der folgenden Abbildung 2.3, sind alle Services der OLS Spezifikation 1.1 zu sehen. Im Rahmen dieser werden sie als sogenannte *Core Services* bezeichnet.

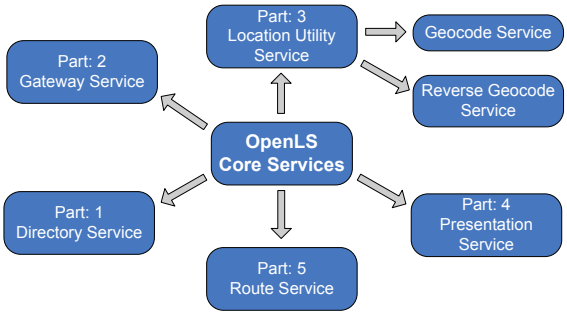


Abbildung 2.3: Die Core Services der OpenLS-Spezifikation Version 1.1

Eine Komponente, die aus den OLS Core Services bereitgestellt werden kann, heißt *GeoMobility Server* (GMS). Er nutzt offene Schnittstellen für den Zugriff auf Positionsdaten eines mobilen Users (zum Beispiel unterstützt durch ein *Gateway Mobile Location Center* (GMLC)) und bietet eine weitere Anzahl von Schnittstellen an. Anwendungen, die auf diesem oder einem anderem Server liegen, erhalten damit Zugriff zu den OLS Core Services. Weitere Anwendungen im GMS können dann diese Basis-Dienste nutzen. Das nachfolgende Bild (Abbildung 2.4) zeigt das Konzept wie der GeoMobility Server zu den anderen Elementen von der LBS Architektur in Beziehung steht.

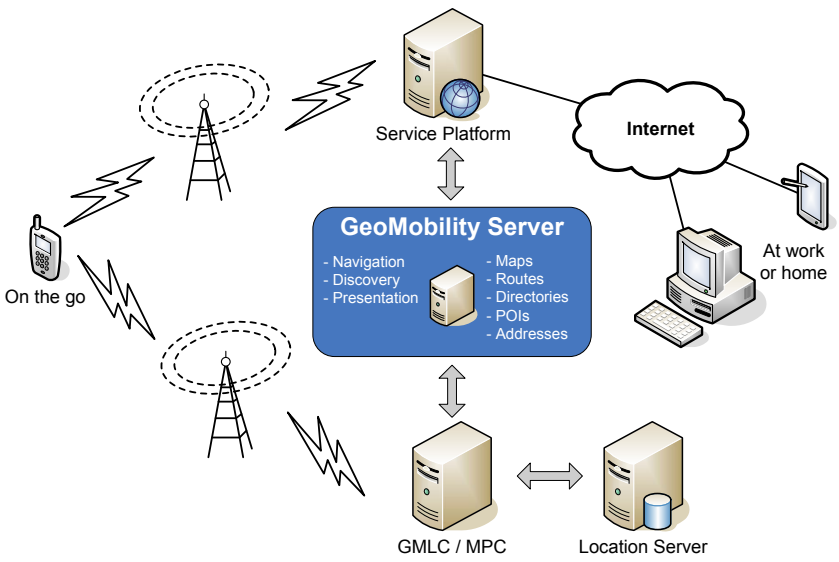


Abbildung 2.4: Der GeoMobility Server aus OLS (2005)

In der Abbildung sind neben dem GMS noch andere Komponenten zu sehen. Das Internet spielt dabei die Rolle des Netzwerkes, über welches die Services unter anderem über

einen PC oder PDA erreichbar sind. „On the go“ kann ein User den GMS auch über einen GMLC, also einen Mobilfunkserver, nutzen. In den folgenden Abschnitten werden die einzelnen Core Services aus Abbildung 2.3 vorgestellt.

Directory Service

Der Directory Service ermöglicht einen netzwerkbasierten Zugriff auf Online-Branchenverzeichnisse, wie z.B. „Gelbe Seiten“. Mit den Verzeichnissen lassen sich Orte, Produkte oder Dienste suchen, die sich um die eigene Position herum befinden. Oder es kann die Position von bekannten Orten/Produkten/Diensten, unabhängig vom Standort des Nutzers, geliefert werden. Bei einer Anfrage an einen Directory Service können zwei Grundformen unterschieden werden:

- Der Pinpoint Directory Service dient der Suche nach einem Ort, Produkt oder Dienst, die jedoch alle nichts mit der aktuellen Position zu tun haben, z.B. „Suche Nassauer Hof in Wiesbaden!“
- Ein Proximity Directory Service ermöglicht die Suche nach dem nächstgelegenen Ort, Produkt oder Dienst mit Hilfe der aktuellen Position, z.B. „Suche nächstes Hotel um meine aktuelle Position herum!“

Meistens kommt jedoch eine Mischform aus beiden Arten vor, die sowohl inhaltliche, als auch räumliche Vorgaben beinhaltet, wie beispielsweise „Suche nächstes McDonalds Restaurant im Umkreis von 2 Kilometern!“. Mehr Informationen vgl. Neis (2006), Neis (2008) und OLS (2005).

Gateway Service

Mit Hilfe eines Gateway Service kann die aktuelle Position eines Nutzers ermittelt werden. Dies geschieht durch einen Zugriff auf einen Mobilfunkserver, der die Positionsdaten eines bekannten mobilen Gerätes (beispielsweise eines Mobiltelefons) auf Anfrage zurückliefert (Abbildung 2.5).

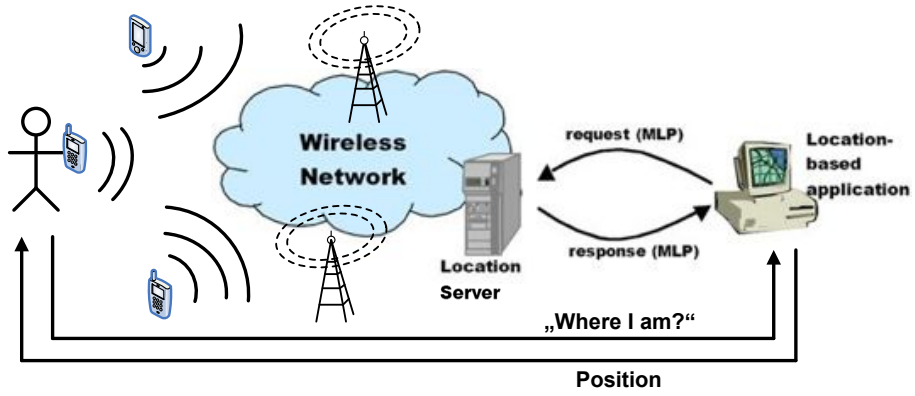


Abbildung 2.5: Gateway Service - Zugriff auf Positionsdaten mit MLP (geändert, aus MLP (2002))

Ein Location Server, wie in der obigen Abbildung 2.5 zu sehen, wird von Mobilfunknetzbetreibern zur Verfügung gestellt und kann die Position eines mobilen Gerätes innerhalb des zugehörigen Mobilfunknetzes berechnen. Die benutzte Schnittstelle entspricht dabei der *Mobile Location Protocol Specification* (MLP) (MLP, 2002). Sie ist eine API und definiert den Zugriff auf die Positionsdaten mobiler Geräte. Unabhängig vom darunter liegenden Netzwerk können Positionsdaten des Nutzers eines beliebigen Telefonnetzwerkes (z.B. GSM und UMTS) zwischen dem Gateway Service und dem Location Server ausgetauscht werden. Mehr Informationen vgl. Neis (2006), Neis (2008) und OLS (2005).

Location Utility Service

Der Location Utility Service ist im OLS Kontext ein Oberbegriff für die beiden Services, die Positionskoordinaten und verbale Ortsangaben ineinander umwandeln. Er unterteilt sich in zwei „Dienste“: den *Geocoder Service* und den *Reverse Geocoder Service*. Die Aufgaben der beiden Services sind vergleichbar, aber leicht unterschiedlich.

Der *Geocoder Service* gibt zu einem Namen oder einer Adresse in normalisierter Beschreibung eine oder mehrere geographische Positionen (Koordinaten) zurück. Die Antwort des Service enthält eine Liste von Adressen mit den dazugehörigen Positionen und sortiert nach der besten Übereinstimmung mit der übergebenen Adresse.

Der *Reverse Geocoder* macht das Ganze in umgekehrter Reihenfolge. Ihm wird eine geographische Position (Koordinate) übergeben und er antwortet mit einer normalisierten Ortsangabe, meist einer Straßenadresse. Mit der Anfrage an den Reverse Geocoder kann eine bestimmte Form der Ortsangabe angefordert werden. Als Antwort zu einer Position wären z.B. nahe liegende Kreuzungen oder POIs möglich. Mehr Informationen vgl. Neis (2006), Neis (2008) und OLS (2005).

Presentation Service

Der Presentation Service ist für die graphische Präsentation von Karten zuständig. Er ist vergleichbar mit einem OGC *Web Map Service* (WMS), unterscheidet sich jedoch in ein paar Punkten von ihm. So können für die Definition des Bildausschnittes andere Parameter, wie die Kombination von Bildmittelpunkt mit einem Radius oder der Displaymaßstab, angegeben werden. Auch kann bei einer Karten-Anfrage eine Bildschirmauflösung berücksichtigt werden. Die Karte wird über eine angegebene BoundingBox und mit allen verfügbaren oder den angegebenen Layern erstellt. Optional kann in der Karte auch ein mit in der Anfrage mitgelieferter *Point Of Interest* (POI) dargestellt werden. Es können aber auch Adressen oder eine berechnete Route, inklusive deren Richtung und Manöver, mit Beschreibung (wo beispielsweise abgebogen werden muss) in die Karte aufgenommen werden. Es gibt zwei verschiedene Arten von Requests an einen Presentation Service (vgl. Neis (2006), Neis (2008) und OLS (2005)):

- Der *GetPortrayMapCapabilitiesRequest* fragt eine Liste der Eigenschaften und Fähigkeiten des Presentation Services ab, beispielsweise welche Ausgabeformate oder Layer vorhanden sind.
- Der *PortrayMapRequest* kann eine Karte unter Vorgabe bestimmter Layer oder Styles anfordern.

Route Service
Mit Hilfe des Route Service kann eine OLS konforme Anwendung Fahr- oder Gehrouten ermitteln lassen. Dabei gibt es zwei verschiedene Arten von Routen-Requests.

1. Eine Route-Anfrage, die die Berechnung einer Route zur Folge hat.
2. Oder eine Neuberechnung einer bereits auf dem Server gespeicherten Route.

Bei einem neuen Route-Request müssen mindestens Start- und Zielpunkt sowie die Art der Routensuche, wie z.B. „Schnellste“, „Kürzeste“ oder „Zu Fuß“, enthalten sein. Alle anderen Parameter, wie zum Beispiel Zwischenpunkte oder Vermeidungsgebiete, können optional bei der Anfrage angegeben werden. Die Antwort von einem Route Service besteht mindestens aus einer Route-Zusammenfassung. Diese enthält beispielsweise die Reisezeit und eine Streckenbeschreibung. Fahranweisungen, Karte(n) und Geometrien der Route können optional angefordert und erhalten werden (mehr Informationen vgl. Neis (2006), Neis (2008) und OLS (2005)).

2.5 Web Feature Service & Web Map Service

Um Karten beispielsweise in einer Anwendung oder auf dem Mobiltelefon anzeigen lassen zu können, wird ein Programm benötigt, das diese, beziehungsweise die „visuelle Repräsentation von Geodaten“[3] erstellen kann. Hierfür wurde vom OGC die „Web Map Service“ (WMS) Spezifikation verabschiedet. Diese bietet vereinfacht gesagt: Zugriff auf einen Dienst, der Landkarten über ein Netzwerk, wie zum Beispiel das Internet, erstellen kann.

Die Spezifikation und der technische Hintergrund (WMS)
In ihr ist standardisiert, wie eine WMS-konforme Anwendung anzusprechen ist und wie die Antwort von ihr auszusehen hat. Das bedeutet: „Verwendet ein Client oder eine Server-Software die WMS Spezifikation, können Karten von jeden beliebigen Server genutzt werden, der auch die WMS Spezifikation unterstützt.“ (WMS, 2001) Die erstellte Karte wird in Form eines Bildes in vorgegebenen Dateiformaten zurückgesendet. Vom einfachen Raster-Grafikformat oder je nach Implementierung des Services auch als *Scalable*

[3] Web Map Service (WMS) Implementation Specification - http://www.opengeospatial.org/standards/wms

Vector Graphics (SVG) Dateiformat. Ein WMS kann laut der Spezifikation drei verschiedene Operationen unterstützen, zwei davon muss er anbieten (vgl. WMS (2001)):

1. GetCapabilities (required) - Frage nach den Fähigkeiten des WMS. Die Antwort erfolgt über ein XML-Dokument mit Metainformationen, das unter anderem Angaben zum Anbieter, die unterstützten Ausgabeformate der Karte und abfragbare Layer beinhaltet.
2. GetMap (required) - Liefert die angefragte Karte als Bild oder als ein Satz von Features zurück.
3. GetFeatureInfo (optional) - Gibt thematische Informationen über Daten an einer Position der Karte zurück.

Damit Geodaten von verschiedenen Systemen oder Anwendungen genutzt werden können, bietet sich deren Speicherung und Datenhaltung in einem OGC *Web Feature Service* (WFS) an. Zusammenfassend gesagt: „Die Spezifikation des WFS definiert Schnittstellen, um den Zugriff und die Änderungen von geographischen Features über ein Netzwerk (zum Beispiel das Internet) zu ermöglichen.“ (WFS, 2002) Somit können Anwender und Anwendungen, die die WFS Schnittstelle nutzen, auf jeden erreichbaren WFS Server zugreifen und dessen Geodaten kombinieren, verwenden und organisieren.

Die Spezifikation und der technische Hintergrund (WFS)
Gemäß der WFS Spezifikation[4] muss zwischen zwei WFS-Typen unterschieden werden: Ein „Basic“ WFS erlaubt die Bereitstellung und Abfrage von Features[5]. Der „Transactional“ WFS erlaubt die Erstellung, Löschung und Änderung von Features (manchmal abgekürzt als WFS-T).

Folgende Operationen muss der „Basic“ WFS anbieten (vgl. Neis (2006) und (WFS, 2002)):

1. *GetCapabilities* - Mit dieser Operation wird nach den Fähigkeiten des WFS gefragt. Als Antwort erfolgt ein XML-Dokument mit Angaben zum Anbieter des WFS, abfragbaren Feature Types und möglichen Operationen.
2. *DescribeFeatureType* - Ist eine Anfrage, bei der Informationen zur Struktur der einzelnen Feature Types zurückgegeben werden.
3. *GetFeature* - Beinhaltet eine Anfrage, bei der die eigentlichen (gefilterten) Geodaten zurückgegeben werden.

[4] Web Feature Service (WFS) Implementation Specification - http://www.opengeospatial.org/standards/wfs
[5] „Unter einem Feature versteht man hierbei die allgemeine Abstraktion eines Fakts der Realität („real world phenomena“)“ - http://de.giswiki.net/wiki/WFS

Beim WFS-T sind folgende Operationen möglich:

1. *Transaction* (required) - Diese Operation stellt die Möglichkeiten Insert, Update und Delete bereit. Damit können Features in der Datenbasis eingefügt, aktualisiert oder gelöscht werden.
2. *LockFeature* (optional) - Bietet eine Option, mit der bei einer Transaction das Feature nicht von einer anderen Operation geändert werden kann.

Auf die einzelnen Parameter der verschiedenen Operationen wird hier nicht mehr weiter eingegangen, mehr Informationen sind zu finden unter WFS (2002).

Styled Layer Descriptor

Im Kontext eines WMS muss auch die *Styled Layer Descriptor* (SLD) Spezifikation des OGC erwähnt werden. Mit ihrer Hilfe ist es möglich, die Kartendarstellung eines WMS zu beeinflussen. Die Karte eines WMS baut sich in der Regel aus mehreren Schichten (Layern) zusammen. Beispielsweise ein Layer für die Straßen, ein Layer für Gebäude und so weiter. Unter Umständen gibt es für die Art der Darstellungen der Layer immer nur Default-Styles, also einheitliche Farben für Straßen (Linien) oder Gebäude (Polygone). Über SLD ist aber eine sogenannte User-definierte Farbgestaltung oder auch Symbolisierung von Karten möglich, das heißt: einzelne Layer können durch SLD klassifiziert werden. Somit können Inhalte der Layer anhand ihrer Attribute in verschiedenen Klassen sortiert oder gefiltert werden. In der Abbildung 2.6 ist der Vorteil zu sehen. Beim Straßen-Layer wurden die verschiedenen Straßen anhand ihres Types unterschiedlich in der Breite und Farbe gezeichnet und Fußgängerwege komplett rausgefiltert. Ähnliches gilt für Nutzungsflächen, die in unterschiedlichen Farben dargestellt werden.

(a) Mit SLD gestyled (b) Mit Default-Style

Abbildung 2.6: Karten mit und ohne SLD gestyled (OpenStreetMap Daten Bonn)

Die einzelnen SLD Elemente werden nicht näher beschrieben (mehr Informationen unter SLD (2002)). Neubauer und Zipf (2007) zeigten wie SLD für 3D oder Dietze und Zipf (2007) demonstrierten wie SLD für die thematische Kartographie erweitert werden kann.

2.6 Erreichbarkeitsanalyse

Eine für viele Planungs- und Standortanalysen relevante GIS Analyse ist, die Erreichbarkeit von Regionen von einer vorgegebenen Position aus zu ermitteln. Sie haben in der Geoinformatik-Forschung eine lange Geschichte (vgl. Juliao (1998), Miller (1999)). Es gibt eine Anzahl verschiedener Definitionen und außerdem sind diverse mehr oder weniger komplexe Modelle verfügbar. Bei dem von Neis, Dietze und Zipf (2007) realisierten Web Service wurde kein besonders komplexes Erreichbarkeitsmodell (zum Beispiel im Sinne einer Interaktionsmodellierung) verwendet. „Die Analyse bestimmt ein Gebiet als Polygon, das von einer als Parameter zu übergebenden Position auf Basis eines attributierten Strassengraphens diejenigen Orte und Strassenabschnitte ermittelt, die innerhalb einer vorgegebenen Zeit oder einer metrischen Distanz erreicht werden können“ Neis, Zipf, Helsper und Kehl (2007).

Als Ergebnis konnte die Fläche über drei unterschiedliche Verfahren visualisiert werden:

(a) Isolinienberechnung (b) Konvexe Hülle (c) Gepufferte Straßen

Abbildung 2.7: Ergebnisse der Erreichbarkeitsanalyse nach Neis et. al. (2007)

Das Besondere an der gezeigten Erreichbarkeitsanalyse war, dass mit ihr nicht nur Gebiete bestimmt werden können, sondern auch Zeiten und Entfernungen, z.B. der Orte oder *Point of Interest* (POI), die sich innerhalb dieser möglichen Erreichbarkeit befinden. Das Erreichbarkeitsanalysen auch im Zusammenhang mit Location Based Services eine wichtige Entscheidungsgrundlage für den Nutzer bieten können, zeigte Petsching (2008). Er nutzte den implementierten Erreichbarkeitsdienst von Neis et al. (2007) in einer auf Fußgänger angepassten Version für die Planung von Besichtigungstouren. Zipf und Röther (2000) erläuterten und zeigten bereits eine ähnliche Anwendung mit Hilfe der „Network Analyst“ Extension von ESRI ArcGIS.

3 OpenStreetMap
The Free Wiki World Map

Dieses Kapitel stellt alles wichtige dar, was zum Projekt *OpenStreetMap - The Free Wiki World Map*[1] zählt. Angefangen mit der geschichtlichen Entwicklung des Projektes, den Bestandteilen, warum und wie es funktioniert.

3.1 Was ist OpenStreetMap?

Für die Frage, was denn *OpenStreetMap* (OSM) ist und warum es von nutzen sein könnte, findet sich auf der Startseite des OSM Wikis folgende Definition (OSM, 2004):

> OpenStreetMap creates and provides free geographic data such as street maps to anyone who wants them. The project was started because most maps you think of as free actually have legal or technical restrictions on their use, holding back people from using them in creative, productive or unexpected ways.

Mit anderen Worten ausgedrückt heißt dies: OpenStreetMap erstellt und bietet „freie“ geographische Daten für jeden, der sie haben möchte, zum Beispiel für Stadtpläne. Das Projekt wurde gestartet, weil die meisten bekannten Karten rechtlichen oder technischen Beschränkungen für ihre Verwendung unterliegen. Dies hält Anwender zurück, die Karten oder Daten in kreativer, produktiver oder unerwarteter Weise zu nutzen. „Frei“ kann hier genauso wie bei „Freier Software“ verstanden werden. „Freie Software bedeutet die Freiheit des Benutzers, die Software zu benutzen, zu kopieren, sie zu vertreiben, zu studieren, zu verändern und zu verbessern.“[2] Oder von der deutschen OSM Website: „OpenStreetMap ist ein Projekt mit dem Ziel, eine freie Weltkarte zu erschaffen .... Weil wir die Daten selbst erheben und nicht aus existierenden Karten abmalen, haben wir selbst auch alle Rechte daran. Die OpenStreetMap-Daten darf jeder lizenzkostenfrei einsetzen und beliebig weiterverarbeiten.“[3]

[1] OpenStreetMap - The Free Wiki World Map - http://www.openstreetmap.org/
[2] The Free Software Definition - GNU Project http://www.gnu.org
[3] OpenStreetMap FAQs: Fragen und Antworten - http://www.openstreetmap.de/faq.html

OpenStreetMap ist, wie bereits bei der Motivation zu dieser Arbeit (Kapitel 1.3) beschrieben, eine Gemeinschaftsarbeit (collaborative effort). Dabei verfolgt das OSM Projekt den Wikipedia-Ansatz, das heißt: Jeder kann mitmachen, jeder kann Daten, die er für wichtig hält, eingeben, seine oder andere Daten verändern, verbessern oder auch löschen, alles nach einer vorherigen Anmeldung beim Projekt. Während des gesamten Projektes werden Probleme dann gelöst, wenn sie auftauchen.
Im Zusammenhang mit der Vorgehensweise beim OSM Projekt wird gerne auch Ward Cunningham zitiert:"The simplest thing that could possibly work." In gewissen Bereichen kann somit das OpenStreetMap Projekt mit der freien Enzyklopädie Wikipedia[4] verglichen werden. Wo OSM allerdings noch etwas hinten dran ist: „es fehlt vor allem die Möglichkeit, die Veränderungen an der Karte in einer für Menschen leicht verständlichen Form darzustellen."(Ramm & Topf, 2008) Auch wenn Daten aus der Karte ausversehen oder absichtlich gelöscht wurden, sind diese nur wieder mit Handarbeit herzustellen. Daraus entstehen allerdings auch andere Probleme: Anders als bei Wikipedia, das aus einer Anzahl von vielen Artikeln besteht, bilden bei OSM alle Daten eine Karte. Grundsätzlich ist das Projekt somit nicht unbedingt gegen Vandalismus geschützt. Darauf wird aber später in diesem Kapitel noch eingegangen.

Stellt sich nun die Frage: ***Warum OSM statt Google Maps oder Map24 verwenden?***
Wie ebenfalls bereits bei der Motivation zu dieser Arbeit beschrieben, können zwar die meisten Kartenanbieter im WWW kostenlos genutzt werden, die Verwendung oder Weitergabe deren Abbildungen oder Karten sind aber in der Regel keinesfalls kostenlos nutzbar.[5] [6] Es gibt viele Vorteile von OSM, der für diese Arbeit aber wichtigste dürfte wohl sein, dass Zugriff auf die „rohen" Geodaten der Karten besteht. Allgemein bedeutet dies: durch den Erhalt dieser Geodaten können sie in der verschiedensten Weise genutzt werden. Zum Beispiel, um eigene Karten mit beliebigem Layout zu erstellen oder sie in unterschiedlichen Varianten in einem *Geographischen Informationssystem* (GIS) zu nutzen. Oder wie bei dieser Arbeit versucht werden soll, sie in einem Routenplaner, Karten- und Geodatendienst anzuwenden. Bei proprietärem Kartenmaterial á la NAVTEQ[7] oder Tele Atlas[8] müssten diese Geodaten erst über eine Lizenz eingekauft werden. Unter Umständen können dann sehr hohe Kosten für die Nutzungsart oder auch den Umfang der Daten entstehen. Ein weiterer Vorteil von OSM gegenüber anderen Kartenanbietern ist: Es werden weltweite aktuelle Geodaten produziert, die von Leuten mit Ortskenntnis und Interesse erfasst werden. Die letzteren beiden Punkte bilden einen nicht unerheblichen

[4] Wikipedia - Wikipedia:Hauptseite http://de.wikipedia.org/wiki/Wikipedia:Hauptseite
[5] Google Maps - AGBs - http://www.google.com/intl/de_de/help/terms_maps.html
[6] Map24 Nutzungsbedingungen - http://www.de.map24.com/eula
[7] NAVTEQ - http://www.navteq.com
[8] Tele Atlas - Digital Mapping & Navigation Solutions - http://www.teleatlas.com

Unterschied zu Mitarbeitern von Kartenfirmen.

Durch die Vielzahl der unterschiedlichsten Daten, die im OSM Projekt enthalten oder am entstehen sind, können in vorhandenen aber auch in neuen Anwendungsfeldern genutzt werden. Verwendet werden können sie beispielsweise in den üblichen Fachanwendungen von Kommunen oder Tourismus. Neue Anwendungsfelder bilden sich zum einen im Bereich der privaten GIS-Anwendungen, wie zum Beispiel in der eigenen Webseite. Zum anderen in speziellen Anwendungen wie: Routenplaner für Wanderer, Rollstuhlfahrer oder sehbehinderten Menschen. Solche Anwendungen waren bis jetzt aufgrund der fehlenden Daten nur bedingt realisierbar.

Tabelle 3.1: Geschichtliche Entstehung und Entwicklung des OSM Projektes

Datum	Ereignis
Aug. 2004	Steve Coast startet das Projekt OpenStreetMap in England
Dez. 2005	1000 registrierte Benutzer
Mär. 2006	Die Infrastruktur ist bereit, um große Gebiete zu kartografieren
Mär. 2006	Osmarender
Apr. 2006	Gründung der OpenStreetMap Foundation
Mai 2006	Isle of Wight Mapping Party - Treffen von Mappern aus ganz Europa, um die Insel in OSM zu kartografieren
Jul. 2006	ca. 2500 registrierte Benutzer & ca. 9 Mio. GPS Wegpunkte in der Datenbank
Nov. 2006	Slippy Map
Feb. 2007	Stadt Cambridge in England ist in OSM „fertig“
Jul. 2007	1ste Internationale OSM Konferenz: *The State of the Map* (SOTM) 2007
Jul. 2007	ca. 10.000 registrierte Benutzer & ca. 90 Mio. GPS Wegpunkte in der DB
Jul. 2008	2te Internationale OSM Konferenz: *The State of the Map* (SOTM) 2008
Jul. 2008	ca. 49.000 registrierte Benutzer & ca. 362 Mio. GPS Wegpunkte in der DB. Größe des weltweiten Datenbestandes: 4.2GB (bz2-Komprimierung) auf http://planet.openstreetmap.org

Wie in der Tabelle 3.1 zu sehen, wurde im April 2006 eine Stiftung – die OpenStreetMap Foundation – gegründet. Sie ist eine Non-Profit-Organisation und versucht durch Spenden und sonstige Gelder die Infrastruktur aufrecht zu erhalten. Die Hauptinfrastruktur steht bis heute in Großbritannien und alle Arbeiten am Projekt werden unentgeltlich von Freiwilligen erledigt. „Die Server, auf denen die Datenbank und andere zentrale Dienste laufen, sind zum Teil gesponsort, zum Teil von einzelnen Projektmitgliedern aus eigener Tasche bezahlt. Den Serverbetrieb (Strom, Datenverkehr, usw.) bezahlen eine englische Universität und eine amerikanische Forschungseinrichtung.“[9]

[9] OpenStreetMap FAQs: Fragen und Antworten - Wie finanziert sich das? http://www.openstreetmap.de/faq.html

Wie und welche Daten können in OSM genutzt und eingegeben werden?
Als Datenquellen für OSM Daten können folgende Dinge in Frage kommen:

1. GPS Tracks von Navigationsgeräten
2. Frei-nutzbare Satellitenbilder (NASA Landsat, Yahoo)
3. Frei-verfügbare Geodaten, siehe Beispiel TIGER-Daten[10]
4. Karten deren Copyright abgelaufen ist
5. Vor-Ort-Kenntnisse

Dabei ist die ursprünglichste und auch häufigste Weise (Punkt 1), die über die aufgezeichneten Wege des GPS-Gerätes. Für die Punkte 2 und 4 ist sehr wichtig zu erwähnen: „das Abzeichnen von urheberrechtlich geschütztem Kartenmaterial, wie Google Maps, Stadtplandiensten oder gedruckten Straßenkarten, ist nicht zulässig!“[11] Wobei sich auch hier Ausnahmen finden, beispielsweise in den von Yahoo für OSM freigegebenen Luftbildern oder in den Bildern der US-Regierung, die nicht einer Lizenz unterliegen. Ebenfalls wurden bereits frei verfügbare Geodaten in das Projekt integriert (Punkt 3). Zum Beispiel sind die TIGER-Daten der US-Regierung, die Daten des FRIDA-Projektes von Osnabrück oder Daten eines Niederländischen Unternehmens bereits in OSM übernommen worden. Nicht vergessen werden dürfen die Vor-Ort-Kenntnisse (Punkt 5), die meist jeder mitbringt, der Daten in das Projekt einpflegt.

Möchte man sich am Projekt beteiligen, muss man sich zuerst beim OSM Projekt registrieren. Danach empfiehlt sich folgende ***Vorgehensweise*** (vereinfacht):

1. GPX-Tracks im GPS-Gerät mitloggen (Fotos, Notizen ...)
2. GPX-Tracks auf den OSM Server hochladen
3. Eintragen/“Zeichnen“ der Daten in die Karte: JOSM oder über Potlatch
4. Attributierung der Daten mit JOSM oder über Potlatch
5. Karten selbst rendern oder rendern lassen, danach können sie bei tiles@home oder mapnik betrachtet werden

Bei der Datensammlung mit dem GPS-Gerät können die unterschiedlichsten Fortbewegungsvarianten verwendet werden: zu Fuß, mit dem Fahrrad, dem Auto oder der Bahn. Auf die verschiedenen Editoren und die Programme,die für das Zeichnen der Karten zuständig sind, wird später noch etwas in diesem Kapitel genauer eingegangen.

[10] Topologically Integrated Geographic Encoding and Referencing - „... TIGER-Daten sind kostenlos verfügbar, da ein Werk der Regierung der Vereinigten Staaten ..“ http://de.wikipedia.org/wiki/TIGER
[11] OSM Wiki De:Newbie - http://wiki.openstreetmap.org/index.php/De:Newbie

Die eben beschriebene Vorgehensweise ist in der folgenden Abbildung 3.1 veranschaulicht:

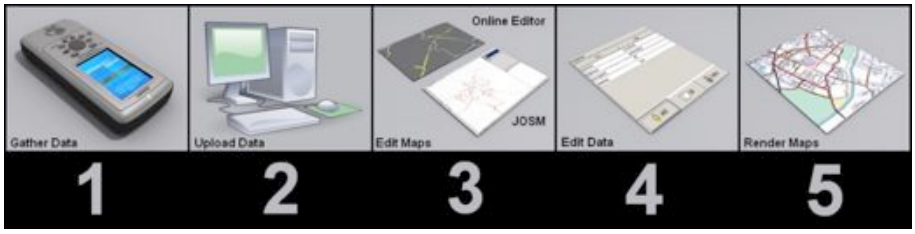


Abbildung 3.1: „The 5 steps to making a map“ (aus OSM Wiki)

Berechtigte Frage: ***Wie sieht es mit der Genauigkeit aus?***
Im Kapitel *Location Based Service 2.1* wurde bereits ein wenig auf das *Global Positioning System* (GPS) eingegangen. Dabei wurde auch erwähnt, dass die Genauigkeit früher noch bei +/-100 Metern lag. Aber durch die Aufhebung der künstlichen Verschlechterung des Signals durch die US-Regierung, war danach eine Genauigkeit von +/-10 Metern möglich. Heutzutage liegt, durch immer bessere GPS-Chipsätze, die Genauigkeit bei Rund +/- 5 Metern. Durch verschiedene Aufbauten, zum Beispiel *Differential GPS* (DPGS), wären noch genauere Positionsbestimmungen möglich, aber die Frage ist: *Reichen nicht auch diese +/- 5 Meter aus?* Sie kann mit ja beantwortet werden. OSM Karten können natürlich nicht mit Kataster- und Liegenschaftskarten verglichen werden. „Aber für die Stadt- und Landkarten, die OSM produziert, ist es allemal genau genug.“„Ob die Straße nun ein paar Meter weiter nördlich ist, spielt eine geringe Rolle - viel wichtiger ist, dass sie topologisch richtig erfasst ist.“(Ramm & Topf, 2008) Und genau dieser Punkt, topologisch richtige Daten, ist die entscheidende Grundlage, um einen Routenplaner anhand von Straßendaten beziehungsweise OSM Straßendaten aufzubauen.

3.2 Überblick der Komponenten & das Datenmodell

Bevor auf das Datenmodell des OpenStreetMap Projektes eingegangen wird, sollen zuvor die Komponenten im Überblick gezeigt werden. Die Abbildung 3.2 zeigt, dass im wesentlichen das OSM Projekt aus den folgenden vier Komponenten besteht. Erstens, den Editoren, wie zum Beispiel Potlatch (Flash-Online-Editor für den Webbrowser) oder JOSM (Java OpenStreetMap Editor). Zweitens, der wichtigsten Komponente - der MySQL Datenbank. Drittens, aus verschiedenen Renderen, die für die Karten, besser gesagt für deren Erstellung, zuständig sind. Und viertens den Viewern, die in verschiedenen Anwendungen oder Browsern genutzt werden, um die OSM Karte anzuzeigen.

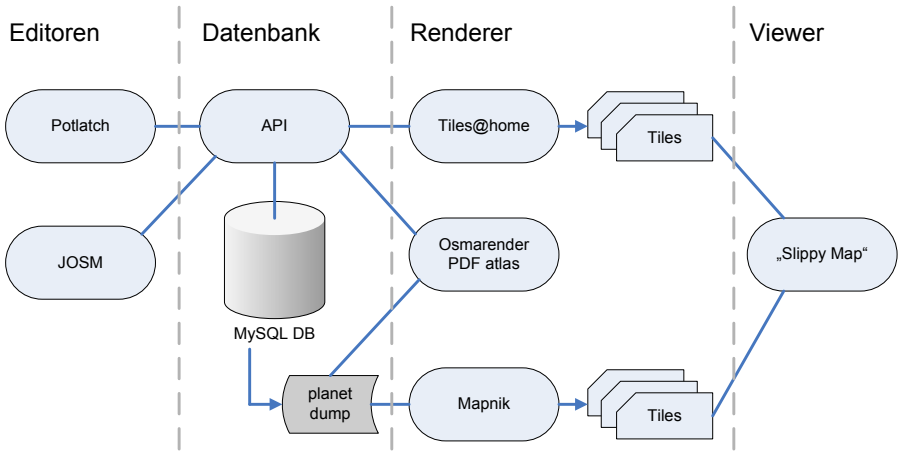


Abbildung 3.2: Überblick der OSM Komponenten (geändert aus Ramm (2008))

Wie in der Abbildung 3.2 zu sehen, gibt es einen zentralen Datenbankserver. Dieser steht in einem Serverraum der University College of London. Es bestehen prinzipiell zwei Möglichkeiten auf OSM Daten zuzugreifen: Entweder direkt auf die *Datenbank* (DB) über die *OSM-API*[12] oder es wird ein regelmäßig erzeugter Datenbankabzug (Dump) verwendet. Bei beidem werden die OSM Daten immer im selben XML-Dateiformat geliefert. Dieses wird anschließend beschrieben. Bei der Nutzung der OSM-API, um OSM Daten zu erhalten, wird ein geographischer Bereich angegeben, der vom Server geladen werden soll. Bei dieser Nutzung gibt es allerdings Beschränkungen: entweder können maximal 50.000 Nodes abgefragt werden oder der Bereich darf nicht größer als 0.25 Quadrat-Grad sein (in Deutschland etwa 50x50km). Es gibt noch eine weitere API, die keinerlei Beschränkungen aufweist: Osmxapi[13]. Die Dumps der kompletten OSM DB (also der gesamten Welt) werden wöchentlich erstellt und als ein sogenanntes *Planet File*[14] zum Download angeboten. Daneben gibt es zusätzlich noch Änderungsdateien, die die täglichen, stündlichen und sogar minütlichen Änderungen zum Planet File enthalten. Um nicht immer mit dem kompletten Datensatz zu arbeiten, bieten verschiedene Webseiten Auszüge aus dem Planet File zum Download an. Für diese Arbeit wurden größtenteils die OSM-Dateien (*.osm) und Shape-Dateien (*.shp) von der Geofabrik[15] verwendet. Auf die anderen Komponenten aus der Abbildung 3.2 wird später in diesem Kapitel eingegangen.

[12] OSM Protocol Version 0.5 - http://wiki.openstreetmap.org/index.php/OSM_Protocol_Version_0.5
[13] Osmxapi - http://wiki.openstreetmap.org/index.php/Osmxapi
[14] Planet File - http://planet.openstreetmap.org/
[15] Downloads auf geofabrik.de - http://download.geofabrik.de

Das OSM Datenmodell

Es bildet die Grundlage für die Datenbank und für das XML-Austauschformat der OSM Daten. Folgende drei Objekte sind die grundlegenden Objekttypen in OpenStreetMap: ***Node***, ***Way*** und ***Relation***. Jedes dieser Objekte kann Attribute besitzen, diese werden *Tags* genannt. Ein Tag besteht aus einem *Key* und einem *Value*. Über diesen sogenannten „Key=Value“ können den oben genannten Typen weitere Beschreibungen zugeordnet werden. Die folgende Abbildung ist eine vereinfachte Darstellung des aktuellen Datenmodells:

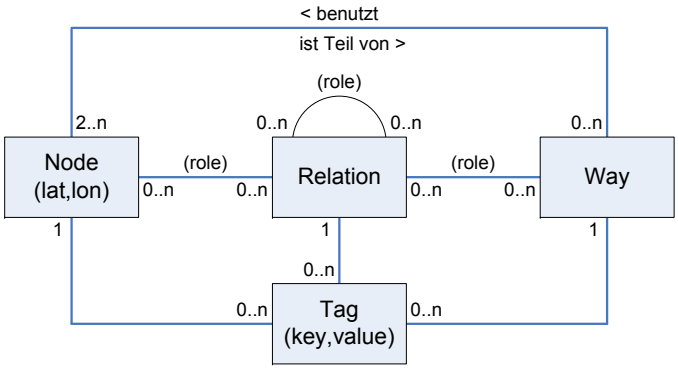


Abbildung 3.3: Vereinfachte Darstellung des OSM Datenmodells aus (Ramm & Topf, 2008)

Jede Node, jeder Way und jede Relation erhalten in der OSM Datenbank bei der Erzeugung einen eigenen, eindeutigen und numerischen *Identifikator* (ID). Über diese ID können die Objekte abgerufen, bearbeitet und gelöscht werden. Nach einer Löschung eines Objektes wird die ID nicht wieder verwendet.

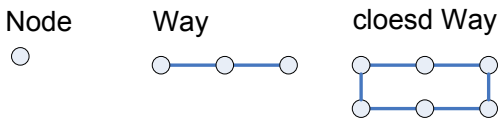


Abbildung 3.4: OSM Objekttypen: Node und Way (und closed Ways)

Eine **Node** ist das Grundelement der OpenStreetMap Daten. Dabei muss es folgende Attribute enthalten: `id` (ID), `lat` (geographische Länge) & `lon` (geographische Breite). Folgende Attribute sind laut der OSM *Document Type Definition* (DTD) 0.5[16] optional, aber trotzdem in der Regel bei den Daten enthalten: `timestamp` (Zeitstempel der letzten Änderung) und `user` (Benutzerkennung des Bearbeiters). In dem nachfolgenden Beispiel (Listing 3.1) sind drei Tags enthalten: Zum Einen der Name (Zeile 3), mit welchen Editor

[16]OSM DTD 0.5 - http://wiki.openstreetmap.org/index.php/OSM_Protocol_Version_0.5/DTD

die Node erstellt wurde (Zeile 4) und zu welcher Art Map Feature es gehört (hier: „Nutzung=Tankstelle“). Was können Nodes noch alles darstellen und beschreiben (vgl. OSM Wiki[17]:

- einen einzelnen geografischen Punkt oder Ort, beispielsweise Sehenswürdigkeiten, Ortschaften, Städte oder POIs;
- Start- und Endpunkt und alle Zwischenpunkte einer Linie (Way);
- alle Punkte des Umrisses einer Fläche.

```
1  <node id="279148121" timestamp="2008-07-18T09:30:53Z" user="pitscheplatsch"
2        lat="50.2483693" lon="8.1942967">
3    <tag k="name" v="Freie Tankstelle"/>
4    <tag k="created_by" v="Potlatch 0.9c"/>
5    <tag k="amenity" v="fuel"/>
6  </node>
```

Listing 3.1: OSM Node

Ein *Way* ist ein Linienzug, der aus mehreren durch Geraden verbundenen Nodes besteht. Durch Ways werden hauptsächlich Straßen und Wege, aber auch andere Elemente, wie Flächen (closed Way), Flüsse, Eisenbahnlinien oder noch einiges mehr, abgebildet. Dabei muss ein **Way** ein *id* (ID) Attribut und mindestens zwei Node-Referenzen (`nd`-Element mit *ref*-Attribut) enthalten. Die anderen Attribute und Elemente sind, wie bei der Node, laut der OSM DTD 0.5 optional. Im nachfolgenden Beispiel (Listing 3.2) sind ebenfalls wieder drei Tags enthalten: neben dem bereits bekannten Tag `created_by`, sind noch `name` (Zeile 4) des Ways und Straßenart (Zeile 6 `highway=residential`) aufgeführt. Wie in einem Way eine Einbahnstraße angegeben werden kann und wie die weiteren wichtigen *Map Feature* aussehen, also *Key*=*Value* Paare, werden im Kapitel 3.3 beschrieben.

```
1  <way id="25611383" timestamp="2008-07-17T15:45:33Z" user="pitscheplatsch">
2    <nd ref="279149781"/>
3    <nd ref="268501735"/>
4    <tag k="name" v="Auf der Langwies"/>
5    <tag k="created_by" v="Potlatch 0.9c"/>
6    <tag k="highway" v="residential"/>
7  </way>
```

Listing 3.2: OSM Way

Eine **Relation** dient zur Modellierung von Beziehungen zwischen OSM Objekten (Node, Way oder Relation). Die Rolle (`role`), die jedem Teilnehmer (`member`) zugewiesen werden kann, ist ein beliebiger Text. Das nachfolgende Beispiel (Listing 3.3) zeigt eine

[17] De:Elemente - http://wiki.openstreetmap.org/index.php/De:Elemente

Abbiegevorschrift: von Way (`ref=211149754`), über Node (`ref=279149781`) und nach Way (`ref=279149733`) darf nicht rechts abgebogen werden. Bis jetzt findet das Relation-Objekt noch keine hohe Verwendung bei der Datenerhebung im OSM Projekt.

```
<relation id="2400" timestamp="2008-07-17T15:45:33Z" user="pitscheplatsch">
  <member type="node" ref="279149781" role="via"/>
  <member type="way" ref="279149733" role="to"/>
  <member type="way" ref="211149754" role="from"/>
  <tag k="restriction" v="no_right_turn"/>
  <tag k="created_by" v="JOSM"/>
  <tag k="type" v="restriction"/>
</way>
```

Listing 3.3: OSM Relation

XML-Datenformat

Die eben gezeigten Beispiele für OSM Objekte (Listing 3.1, 3.2 und 3.3) sind bereits im XML-Format angegeben. Wird mit Daten über eine der beiden OSM APIs oder mit Daten eines anderen Servers gearbeitet, sind sie in diesem Format (*.osm). Der Aufbau der Datei ist durch die OSM *Document Type Definition* (DTD) festgelegt, zur Zeit in Version 0.5. Das nachfolgende Schema-Diagramm (Abbildung 3.5) wurde über eine Konvertierung dieser DTD (*.dtd) zu einem XML Schema (*.xsd) erstellt.

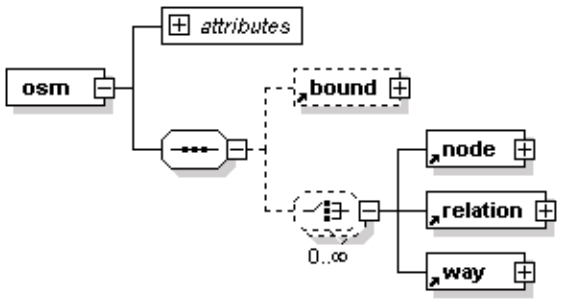


Abbildung 3.5: Schema-Diagramm einer OSM Datei Version 0.5 (*.osm)

Wie in der Abbildung 3.5 zu sehen, besteht eine OSM-XML-Datei aus dem Hauptelement `osm`. Die Attribute wie `version` und `generator` und das Element `bound` können optionalen enthalten sein. Der eigentliche Inhalt ist aber eine beliebige Anzahl der `node`, `way` und `relation`-Elemente. Sie werden auch in dieser Reihenfolge in der Datei aufgeführt.

3.3 Map Features

Als Map Features wird die Liste bezeichnet, die alle Objekte (Tags) enthält, die auf der Karte eingezeichnet werden. Zur Zeit werden Nodes und Ways (inklusive Flächen) über diese Tags modelliert. An der Visualisierung von Relations wird gerade gearbeitet. Die Tags der Objekte, die in einer beliebigen Anzahl bei jeder Node und jedem Way angegeben werden können, bestehen, wie bereits erklärt, immer aus einem Schlüssel und einem Wert („Key=Value“). Wichtig zu erwähnen: Beide können beliebig gewählt und angegeben werden. Das heißt: „OpenStreetMap lässt prinzipiell alle möglichen „Schlüssel“ oder „Werte“ zu. Trotzdem ist es natürlich für die Anwendung der OSM-Daten von Vorteil, wenn Einigkeit herrscht, welche Basiseigenschaften durch welche Datenrepräsentation ausgedrückt werden.“[18]

Eine Liste der möglichen Map Features befindet sich im OSM Wiki[19]. In diesem Kapitel werden nur die wichtigsten Hauptgruppen kurz erwähnt. Detaillierte Informationen können im Wiki eingesehen werden. Neue oder wünschenswerte Map Features (Proposed Features) werden auf einer separaten Seite vorgeschlagen und diskutiert.[20]

Tabelle 3.2: OSM Map Features: Übersicht der wichtigsten Keys

Key	Beschreibung: Art und Weise um ...
highway	Straßen/Wege und deren Objekte zu kennzeichnen
waterway	Wasserwege und deren Objekte zu kennzeichnen
railway	Schienenwege und deren Objekte zu kennzeichnen
leisure	Freizeitflächen und deren Objekte zu kennzeichnen
amenity	die Nutzung zu kennzeichnen
landuse	die Landnutzung zu kennzeichnen
natural	eine Naturfläche zu kennzeichnen
place	Ortschaften/Städte/Länder etc. zu kennzeichnen

Die Tabelle 3.2 gibt nur einen geringen Teil der möglichen Keys und vor allem der sehr großen Anzahl von möglichen Values im Rahmen der OSM Map Features wieder. Welche Map Features bei den einzelnen Location Based Services zum Einsatz kommen, wird im Kapitel 4 beschrieben. Nützliche Übersichten der verwendeten Tags und die Anzahl deren Nutzung in Europa oder den einzelnen Ländern in Europa können *OpenStreetMap Tagwatch*[21] entnommen werden.

[18] OSM Wiki De:Map Features - http://wiki.openstreetmap.org/index.php/De:Map_Features
[19] OSM Wiki: Map Features - http://wiki.openstreetmap.org/index.php/Map_Features
[20] OSM Wiki: Proposed features - http://wiki.openstreetmap.org/index.php/Proposed_Features
[21] OpenStreetMap Tagwatch - http://tagwatch.stoecker.eu

3.4 Editoren, Karten & die Lizenz

Um Daten in das OpenStreetMap Projekt einzugeben oder zu bearbeiten, gibt es mehrere verschiedene Editoren. Die zwei bekanntesten Editoren sind: Potlatch und JOSM. Im Folgenden werden sie kurz vorgestellt. ***Potlatch*** ist ein flashbasierter Online-Editor und mit einem Webbrowser über den 'Edit'-Reiter auf der Hauptkarte von OSM[22], nutzbar. Er ist ideal für schnelle und einfache Änderungen der Daten. Satellitenbilder von Yahoo sind als Hintergrund hinterlegbar, ermöglichen ein Nachzeichnen und die Prüfung von Daten. Da Kartensymbole nur eingeschränkt und mit wenigen Eigenschaften verfügbar sind, wird ein Nacharbeiten in JOSM erforderlich (mehr Informationen unter Potlatch (2007)). ***JOSM*** ist derzeit der populärste OSM Editor und eine Standalone-Java-Applikation. Er eignet sich für alle Arbeiten, ist sehr komfortabel/übersichtlich und wird ständig weiterentwickelt. Des Weiteren gibt es bereits eine Vielzahl von Plugins, die JOSM um sinnvolle Funktionen erweitert (mehr Informationen unter JOSM (2006)).
Mehr Informationen zu allem hier vorgestellten finden sich unter den jeweils angegeben Wiki Seiten des OSM Projektes. Ein guter Vergleich der Editoren ist auch im OSM Wiki[23] zu finden.

Die wohl bekanntesten Kartenrender, die aus OSM Daten Karten erstellen, sind: der Osmarender und Mapnik. Mit dem ***Osmarender*** können aus OSM Daten SVG-Vektorgrafiken erzeugt werden. Das Programm nutzt als Eingabe die OSM Daten und eine Datei, die Regeln beinhaltet, die das Aussehen der einzelnen OSM Map Features beeinflusst. Als Ergebnis erzeugt die Anwendung eine SVG-Grafik, die entsprechend der in den Regeln definierten Styles gerendert wurde (vgl. Osmarender (2006)). Osmarender wird auch für das Tiles@home Projekt (verteiltes Rendern der Kartenkacheln für die „Slippy Map“) verwendet (Tiles@home, 2007). ***Mapnik*** ist ein Open-Source C++ Kartenrenderer und wurde unabhängig von OSM entwickelt. Es wird primär zum Rendern der „Slippy Map“ für OSM verwendet. Damit Mapnik aus OSM Daten Karten erstellen kann, müssen zuvor die Daten in eine PostgreSQL/PostGIS Datenbank importiert werden (mehr Informationen unter Mapnik (2005)).

OSM-Karten im Web und die Slippy Map
Google Maps hat mit seiner intuitiven und einfachen Bedienung der Karte, in der „jeder“ versteht wie er sie zu bewegen hat, sie vergrößern und verkleinern kann, neue Maßstäbe im WWW gesetzt. Somit sind Karten á la Google zu einem „Quasi-Standard“ geworden. Diese Art der verschiebbaren Karte wird im OSM-Jargon als „Slippy Map“ („flinke Karte“) bezeichnet (vgl. Ramm und Topf (2008)). Die „Slippy Map“ ist somit ein Web-

[22] OpenStreetMap - http://www.openstreetmap.org/
[23] OSM Wiki: Comparison of editors - http://wiki.openstreetmap.org/index.php/Comparison_of_editors

Interface, um sich OSM Karten im WWW anzuschauen. Als Kartenrenderer und Kartenlieferanten gibt es verschiedene Varianten, deren Karten auch unterschiedlich aussehen: den Mapnik-Tile-Server und das Projekt Tiles@home. Erstes verwendet den vorgestellten Mapnik Kartenrenderer und das Projekt Tiles@home nutzt den Osmarender.

Um Karten aus Vektordaten erstellen zu können, wird viel Rechenzeit benötigt. Aus diesem Grund werden heutzutage Karten für das WWW bereits stückweise vorgerechnet und als Grafik auf einem Server abgelegt (genau wie bei Google Maps). Dabei wird folgendermaßen vorgegangen: Die Welt wird in Reihen und Spalten zerlegt, sodass jeweils „Kacheln“ (Tiles) für quadratische Bereiche der Karten entstehen. Eine etwas ausführlichere Beschreibung über die Erstellung der Tiles erfolgt im Kapitel 5.3, wo ein TileCache-Server vorgestellt wird, der für diese Arbeit verwendet wurde. Erfolgen Änderungen oder Ergänzungen an den OSM Daten, sind diese nicht direkt in den Karten sichtbar. Die Tiles des Mapnik Servers werden mindestens einmal pro Woche aktualisiert, bei Tiles@home (Osmarender) kann die Aktualisierung der Tiles angestoßen werden (mehr Informationen unter Tiles@home[24]).

Die Lizenz die hinter den Daten steht?
Alle Daten, die in der OpenStreetMap-Datenbank vorhanden sind, stehen unter der Lizenz *Creative Commons Attribution-Share Alike 2.0* (CC-BY-SA 2.0)[25]. „Die Lizenz besagt, dass jegliche Art der Nutzung von OSM-Daten, auch gewerblich, zulässig ist. Es muss jedoch angegeben werden, woher die Daten stammen, und jedes Werk, das aus OSM-Daten abgeleitet ist, muss wiederum unter der CC-BY-SA-Lizenz stehen.“[26] Das die Lizenz nicht so gut für Geodaten geeignet ist, hat sich im Laufe der Zeit herausgestellt, da sie viele Fragen offen lässt. Eine Änderung der Lizenz ist allerdings nicht so leicht durchführbar, da jeder der Daten zu OSM beigetragen hat, um Zustimmung gefragt werden müsste (vgl. Ramm und Topf (2008)).

3.5 Abdeckung & Potential

Genaue Aussagen über die Abdeckung, die Vollständigkeit oder den Detaillierungsgrad zu machen, ist etwas schwierig. Das fängt bei einer weltweiten Betrachtung an und endet in jeder beliebigen Stadt, Dorf oder Straße. Jeder definiert „Vollständigkeit“ unterschiedlich: „Was ist „vollständig“? Alle Autobahnen und Bundesstraßen? Oder alle Radwege und Briefkästen? Jede einzelne Hausnummer und jede Parkbank?“ Dennoch sind bereits

[24] OSM Wiki: Tiles@home - http://wiki.openstreetmap.org/index.php/Tiles%40home
[25] Creative Commons Attribution-Share Alike 2.0 license - http://creativecommons.org/licenses/by-sa/2.0/de
[26] OpenStreetMap FAQs: Wie ist das mit der Lizenz? - http://www.openstreetmap.de/faq.html

heute in Deutschland OSM Karten, beziehungsweise OSM Daten, in vielen Städten schon besser als die meisten proprietären Karten. Genauso gut kann es anderswo aber noch vorkommen, dass ein weißer Fleck oder nur eine Durchgangsstraße, wo eigentlich ein ganzer Ort hingehört, vorhanden sind.[27] Trotzdem dürfen die immense Aktualität, der schnelle Wachstum und die Tausenden von engagierten Erfassern und Anwendern keinesfalls außer acht gelassen werden. Auch wenn Falscheingaben (bewusst oder unbewusst) von jedem der Daten eingibt oder bearbeitet, möglich sind, ist die Datenqualität von der Sorgfalt des Kartierers abhängig.

Statistiken über das OSM Projekt gibt es einige im OSM Wiki[28]. Eine recht beeindruckende ist die folgende Abbildung 3.6. Sie zeigt den Anstieg der Anzahl von hochgeladenen GPS Tracks und registrierten OSM Nutzern. Allerdings muss dazu erwähnt werden, dass der Anstieg im März und Juni 2008, bedingt durch verschiedene Zeitungsartikel und Fernsehbeiträge, verstärkt wurde.

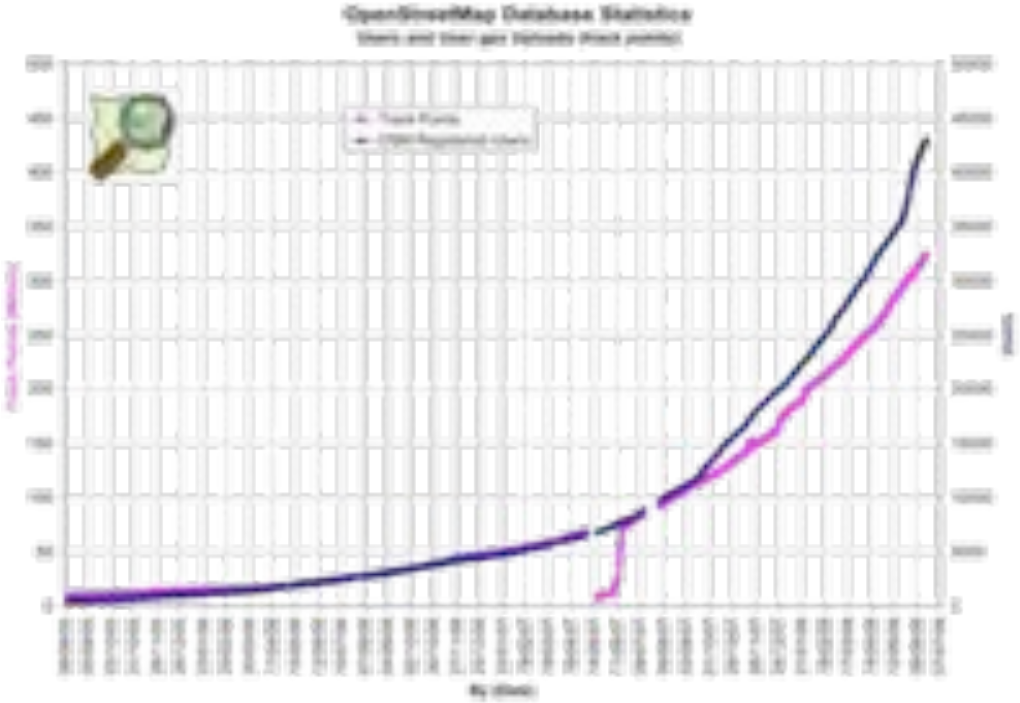


Abbildung 3.6: OpenStreetMap Datenbank Statistik

[27] OpenStreetMap FAQs: Wie vollständig sind die Daten? - http://www.openstreetmap.de/faq.html
[28] OSM Wiki: Stats - http://wiki.openstreetmap.org/index.php/Stats

4 Aufbereitung der OSM Daten für ...

Das erste Ziel dieser Masterarbeit war die Datenaufbereitung der OpenStreetMap Daten, um sie im OLS Route Service von Neis (2006) nutzen zu können. Anschließend sollte versucht werden, die Daten entsprechend für eine Nutzung in einem OGS Web Map Service und Web Feature Service aufzubereiten. Durch schnelle Erfolge bei der Konvertierung der OSM Daten für kleine Bereiche, wurde die Aufbereitung auf ganz Deutschland ausgedehnt. Ergänzend wurden weitere Daten für andere Location Based Services vorbereitet. Was genau bei diesen Aufbereitungen durchgeführt wurde, und wie die verwendeten Daten für die Dienste aussehen, wird in diesem Kapitel gezeigt.

Allgemein: ***Woher kommen die OSM Daten?***
Um die „reinen“ OSM Geodaten zu erhalten, gibt es verschiedene Wege. Die einfachste Weise ist zum Beispiel bei der Webseite der Geofabrik[1] die Daten herunterzuladen. Hier werden die OSM Daten in unterschiedlichen Varianten, als OSM-Datei (*.osm) oder Shape-Datei (*.shp) und für verschiedene Länder bereitgestellt. Weiterhin gibt es noch andere Varianten: Es könnte ein beliebiger Ausschnitt aus dem Planet-File[2] mit Hilfe des Osmosis[3] Tool ausgeschnitten werden. Die Daten könnten aber auch direkt mit Hilfe der Osmxapi[4] (die im Gegensatz zur normalen OSM API keine Beschränkungen besitzt) heruntergeladen werden. Für diese Arbeit wurden größtenteils die OSM Daten der Geofabrik verwendet.

4.1 ... OLS Route Service

Wie bereits erwähnt, soll der vorhandene und inzwischen auch weiterentwickelte OLS Route Service als Routensuchdienst für diese Arbeit verwendet und gegebenenfalls optimiert werden. Was genau geändert wurde, wird im Kapitel *6.1.2 Route Service* beschreiben. Damit Routen berechnet werden können, muss ein Straßennetz vorhanden sein. Da-

[1] GEOFABRIK - http://www.geofabrik.de
[2] Planet.osm - http://wiki.openstreetmap.org/index.php/Planet.osm
[3] Osmosis - http://wiki.openstreetmap.org/index.php/Osmosis
[4] Osmxapi - http://wiki.openstreetmap.org/index.php/Osmxapi

bei gibt es eine Vielzahl von möglichen Straßen, Wegen oder auch Feldwegen, die für die unterschiedlichsten Routenberechnungen verwendet werden könnten. Wird im weiteren Kontext von OSM Straßen geschrieben, sind immer Straßen und Wege gemeint.

Welche Straßen von OSM werden wo im Route Service verwendet?
Vor Beginn dieser Arbeit bot der vorhandene Route Service „nur“ Auto- und Fußgänger-Routenplanung an. Da aber im OSM Projekt unter anderem auch Wege für Fahrradfahrer mit aufgenommen werden, wurden neben den unterschiedlichen Straßen für Autos und Fußgänger, auch die für Fahrräder mit in die Datenbasis für den Route Service aufgenommen. Wie bereits im Kapitel *3.3 Map Features* in Tabelle 3.2 beschrieben, sind alle möglichen Straßen und deren Objekte über den `highway`-Key in den OSM Daten gekennzeichnet. Entsprechend dieses Keys gibt es dann ein Vielzahl von möglichen Values, um eine Straße oder ein Objekt der Straße zu kennzeichnen und in der OSM Karte visualisieren zu lassen. Für die Auto-Routenplanung werden folgende OSM Values des `highway`-Keys verwendet:

Tabelle 4.1: OSM Straßen (`highway`-Key) für OLS Route Service (Auto)

Value	Beschreibung der Straßenart
motorway & motorway_link	Autobahn und Autobahn-Zubringer
trunk & trunk_link	Schnellstraße und Schnellstraßenanschlußstelle
primary & primary_link	Bundesstraßen und Bundesstraßenanschlußstelle
secondary	Land- (, Staats-), und Kreisstraße
tertiary	Gemeindeverbindungsstraße
unclassified	Gemeindestraßen
residential	Straße an und in Wohngebieten
living_street	Verkehrsberuhigter Bereich oder Wohnstraße
service	Erschließungsweg (nur wenn für Auto freigegeben)

Für die Fussgänger-Routenplanung werden neben den Straßen der Auto-Routenplanung (siehe Tabelle 4.1, ausgenommen `motorway` & `motorway_link`) noch die folgenden OSM Values des `highway`-Keys verwendet:

Tabelle 4.2: OSM Straßen (`highway`-Key) für OLS Route Service (Fussgänger)

Value	Beschreibung der Straßenart
Values der Tabelle 4.1 außer motorway & motorway_link	Beschreibungen siehe Tabelle 4.1
service	Erschließungsweg
track	Feld- oder Waldweg
pedestrian	Fußgängerzone
cycleway	beschilderter Radweg (nur wenn Fußgänger freigegeben)
footway	beschilderter Fußweg
bridleway	beschilderter Reitweg (wenn nicht explizit untersagt)
steps	Treppen auf Fuß-/Wanderwegen

Bei der Fahrrad-Routenplanung werden, ähnlich wie bei Fußgänger-Routenplanung, teilweise auch die Straßen der Auto-Routenplanung verwendet:

Tabelle 4.3: OSM Straßen (`highway`-Key) für OLS Route Service (Fahrrad)

Value	Beschreibung der Straßenart
Values der Tabelle 4.1 außer motorway & motorway_link	Beschreibungen siehe Tabelle 4.1
service	Erschließungsweg
track	Feld- oder Waldweg
cycleway	beschilderter Radweg
footway	beschilderter Fußweg (nur wenn Fahrrad freigegeben)
bridleway	beschilderter Reitweg (wenn nicht explizit untersagt)

Neben den eben erwähnten Values des `highway`-Key, gibt es noch verschiedene andere OSM Tags, die für eine Routenplanung wichtig sind.

- ***Einbahnstraßen*** werden über den `oneway`-Key angegeben und mit folgenden belegt: 'true'/'false' , 'yes'/'no' oder '1'/'-1'. Die Werte geben eine Aussage darüber, in welche Richtung die Einbahnstraße verläuft. 'true', 'yes' und '1' bedeutet: die Einbahnstraße verläuft in Richtung der Digitalisierung ab, die anderen Werte entsprechend entgegengesetzt dieser Richtung. Beispiel: *oneway=-1* oder *oneway=yes*
- ***Beschränkungen*** einer Straße für zum Beispiel Autos, Fußgänger oder Fahrradfahrer werden über die Tags: `motorcar`, `foot` und `bicycle` geregelt. Auch hier sind wieder ein paar verschiedene Werte erlaubt, wobei für diese Arbeit erstmal nur 'yes' und 'no' Verwendung finden. Mit Hilfe dieses Tags kann somit für jeden beliebigen Straßentyp ein Beschränkung für den jeweiligen Nutzer (Auto, Fußgänger oder Fahrrad) erteilt werden. Beispiel: *motorcar=yes* oder *foot=no*
- Generelle ***Zutrittsbeschränkungen*** einer Straße können über den `access`-Tag geregelt werden. Auch hier sind wieder ein paar verschiedene Werte erlaubt, wobei erstmal nur 'yes' und 'no' verwendet werden. Auswirkungen hat dies nur auf die Routenplanung von Autos, bei Fußgängern oder Fahrradfahrern kommt es nicht zum Einsatz. Beispiel: *access=no*
- Ein ***Kreisverkehr*** wird über das `junction`-Tag angegeben. Dadurch wird automatisch *oneway=yes* vergeben.

Nicht zu vergessen ist der jeweilige Name der Straße. Der Name wird entweder über dem `name`-Tag oder aber, bei nicht Angabe, dem `ref`-Tag (Straßennummerierung) entnommen. Beispiel: *name=Hühnerstraße* oder *ref=B 417*. Dessen ungeachtet können Probleme durch verschiedene Schreibweisen der Namen oder Straßennummerierungen auftreten. Auch wenn dies eigentlich im OSM Wiki „geregelt“ ist, hängt es doch wesentlich von demjenigen ab, der die Daten eingibt. So werden zum Beispiel Abkürzungen verwendet: *Str.* statt *Straße* und manche nutzen auch „ss“ statt „ß“. Ebenfalls kann es zu Problemen bei der Straßennummerierung kommen: manche erfassen Straßennamen mit und manche ohne Leerzeichen, Beispiel: „B417“ und „B 417“. Beide genannten Probleme werden wie folgt beseitigt: generell werden Straßenabkürzungen („Str.“ oder “str.“) durch eine vollständige Schreibweise ersetzt („Strasse“ oder „strasse“). Das Gleiche gilt für ein „ß“, das immer durch ein „ss“ ausgetauscht wird (Hintergrund: In anderen Ländern wird kein „ß“ verwendet). Auch die Probleme bei der Straßennummerierung dürften durch entsprechende Filter behoben sein, so dass die Nummerierung immer durch ein Leerzeichen vom Buchstaben getrennt ist.

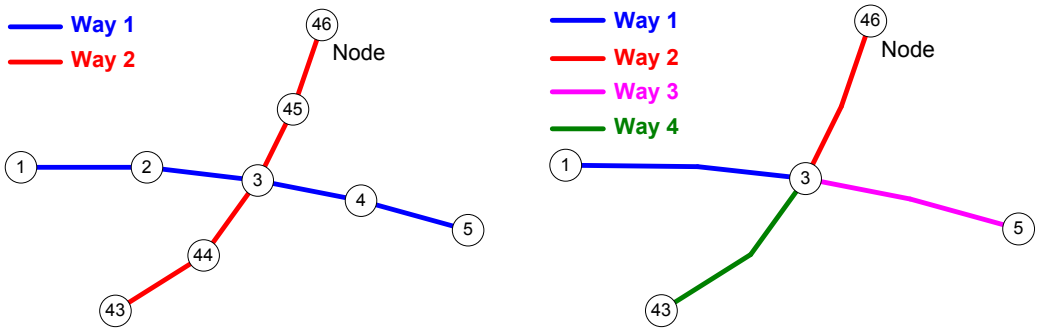
Es gab bis zum Ende dieser Arbeit bereits weitere Tags oder Values, die zur Routenplanung oder für die Fahranweisungen verwendet werden könnten oder auch müssten. Hier ist gut zu sehen, dass sich das OpenStreetMap Projekt unter einer stetigen Weiterentwicklung befindet und sich deren Nutzer ständig Gedanken über Verbesserungen oder Neuerungen machen. Auch sind andere Kombinationen der Tags durchaus denkbar, doch im wesentlichen sind die oben genannten die wichtigsten und vor allem auch die, welche von den OSM Nutzern Verwendung bei Objekten finden. Eine gute Übersicht über die im OSM Projekt genutzten Tags und vor allem deren Häufigkeit, zum Beispiel in Deutschland, bietet hier *Tagwatch*[5].

Was muss an den OSM Straßen Daten geändert werden?

Das wichtigste für eine Routing-Anwendung sind die Daten, insbesondere der Routing-Graph. Ein Routing-Graph repräsentiert dabei das Straßennetz, was in Knoten und Kanten in einem Modell abgelegt ist. Entscheidend ist dabei, dass der Graph über die richtige Topologie aufgebaut wird. Das heißt: Kreuzungen von Straßen werden als Knoten und Straßen als Kanten abgebildet. Weil bei der Erzeugung, wie auch bei der Pflege, von OSM Daten die Nutzer darauf achten topologisch richtige Daten zu erzeugen, ist bereits eine gute Grundlage bei den Daten für einen Routing-Graph vorhanden. Die maßgebliche Aufbereitung der OSM Daten für ein mögliches Routing ist somit: ***aus der vorhandenen Topologie einen Routing-Graphen mit Knoten und Kanten zu erstellen!***

[5] OpenStreetmap Tagwatch - http://tagwatch.stoecker.eu

Wie mögliche Ausgangsdaten und das Resultat aussehen, ist in der folgenden Abbildung 4.1 zu sehen.

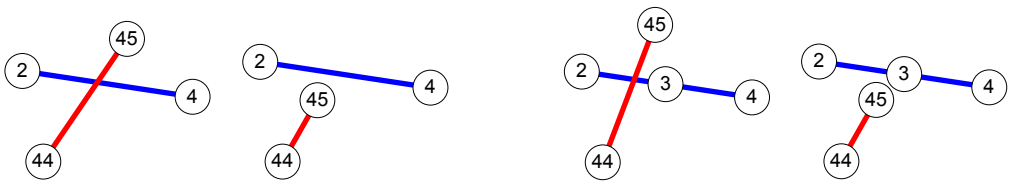


(a) Original OSM Daten (b) Aufbereitete OSM Daten

Abbildung 4.1: Aufbereitung der OSM Daten für OLS Route Service

Wie in der Abbildung 4.1 zu sehen, muss also in der vorhandenen Topologie untersucht werden, an welchen Knoten (im Beispiel Node Nr. 3) sich Ways kreuzen beziehungsweise eine gemeinsame Node besitzen und diese sozusagen eine Kreuzung bilden. Im Original OSM Datensatz (Abbildung 4.1 (a)) waren es zwei Ways (Way Nr. 1 mit Nodes 1,2,3,4 & 5 und Way Nr. 2 mit Nodes 43, 44, 3, 45 & 46) mit insgesamt neun Nodes und einer gemeinsamen Node. Durch diese gemeinsame Node, müssen die zwei Ways an dieser geteilt werden. Dabei entstehen zwei „neue“ Ways (3 & 4) und die Nodes 2, 4, 44 und 45 „fallen weg“ (Abbildung 4.1 (b)). Durch diese Bearbeitung werden hier im Beispiel aus zwei Ways vier Ways mit einer gemeinsamen Node. Was dies für einen gesamten Datensatz ausmacht, zum Beispiel für Deutschland, wird am Ende dieses Kapitels gezeigt.

Genau genommen müssten eigentlich noch weitere Maßnahmen getroffen werden, um einen „vollständigen“ Graphen aus OSM Daten zu erstellen. Unter Umständen kann es immer mal vorkommen, dass folgende Probleme bei Nodes und Ways auftreten könnten:

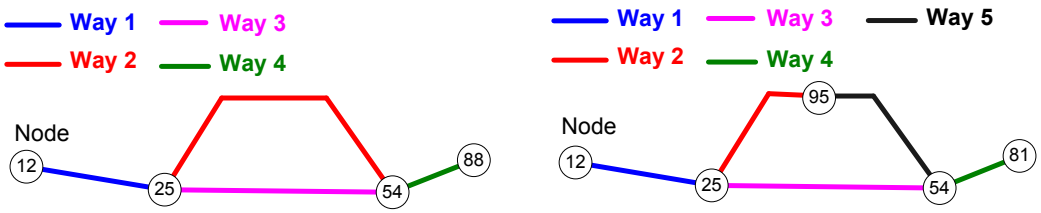


(a) Kein „Schnittpunkt“/Node vorhanden (b) Keine gemeinsame Node vorhanden

Abbildung 4.2: Mögliche andere Probleme bei der Aufbereitung von OSM Daten

Die ersten Tests am Anfang dieser Arbeit zeigten jedoch, dass diese Fälle eher sehr selten auftreten. Sicherlich auch, weil viele Beteiligte am OSM Projekt versuchen ihre Daten topologisch und sauber einzuarbeiten. Was allerdings teilweise zu Problemen

führt ist, wenn eine Straße im OSM Datenbestand aus zwei identischen Nodes besteht. Das heißt eine Straße mit identischer Start- und End-Node muss ebenfalls im Aufbereitungsschritt entfernt werden. Weiterhin mussten aufgrund der genutzten Graph- und Routing-Algorithmus-Implementierung noch weitere Änderungen an den Daten erfolgen. Beispielsweise kann es vorkommen, dass mehrere Straßen von einer identischen Node zu einer anderen identischen Node führen. Die Abbildung 4.3 veranschaulicht dies: Das Problem entsteht bei den Ways Nr. 2 und 3, also von Node 25 nach 54. Sie haben beide unterschiedliche Geometrien, besitzen dennoch beide die gleichen Nodes (25 & 54). Dies kann zu Problemen während des Routings führen, dementsprechend muss die eine von beiden Straßen geteilt werden.



(a) Original OSM Daten (b) Aufbereitete OSM Daten

Abbildung 4.3: Aufbereitung der OSM Daten für OLS Route Service, Part II

Beschreibung der Ursache im Routing-Algorithmus: Bei der Berechnung der Route wird jeweils von einem zum nächsten Knoten (Node) geroutet. Dabei wird das Gewicht der Kante über die Abhängigkeiten der Nodes bezogen. Der implementierte Graph kann also nicht mehrere Kanten, die von einer Node zu einer anderen Node verlaufen, speichern und verwenden. Er gebraucht immer die bei der Initialisierung des Graphen zuletzt eingefügte Kante. In der Abbildung 4.3 würde zum Beispiel einmal die obere Kante (Way 2) und an einer anderen Node mit der gleichen Situation vielleicht mal die untere Kante (Way 3) verwendet werden, je nachdem welche dem Graph zuletzt hinzugefügt worden wäre.

Zu Beginn dieser Arbeit gab es keinerlei Informationen, wie aus OSM Daten routingfähige Graphen erstellt werden können. Erst zum Ende dieser Arbeit entstanden verschiedene Wiki-Seiten, wie OSM Daten für ein Routing optimiert werden müssten: *OSM Wiki: Routing with osm data*[6] oder *OSM Routing Data Layer*[7]

Was wurde jetzt genau gemacht und was beinhalten die fertig aufbereiteten Daten?
Um die Daten für das Routing aufzubereiten, wurde eine Java-Konsolenanwendung entwickelt, die lediglich einen beliebigen OSM-Datenssatz benötigt. Dabei spielt es keine Rolle, ob es sich nur um OSM Daten einer Gemeinde oder eines Landes handelt. Das

[6]OSM Wiki: Routing ... - http://wiki.openstreetmap.org/index.php/Routing_with_osm_data
[7]OSM Wiki: OSM Routing ... - http://wiki.openstreetmap.org/index.php/OSM_Routing_Data_Layer

Java-Tool liest die Daten ein, überprüft welche Nodes von welchen Ways verwendet werden und baut dementsprechend einen Routing-Graphen auf. Anschließend werden noch die eben erwähnten Filter angewendet, die Daten optimiert und das Resultat in einem Shapefile abgespeichert. Alternativ wurde auch eine Lösung für die direkte Importierung in eine Datenbank entwickelt. Das resultierende Shapefile, beziehungsweise die Tabelle, enthält dann folgende Attribute für den Routing-Graph im OLS Route Service:

Tabelle 4.4: Tabellenschema für OLS Route Service

Attribut	Datentyp	Beschreibung
id	INTEGER	ID der Straße (Kante)
name	STRING	Name der Straße
type	STRING	Straßentyp
oneway	STRING	Einbahnstraße: ja oder nein
length	DOUBLE	Länge der Straße in Metern [m]
motorcar	STRING	Nutzungserlaubnis für Autos
pedestrian	STRING	Nutzungserlaubnis für Fußgänger
bicycle	STRING	Nutzungserlaubnis für Fahrräder
tracktype*	STRING	Oberflächenbeschaffenheit der Straße
maxspeed*	STRING	zulässige Höchstgeschwindigkeit in [km/h]
GEOMETRY	GEOMETRY	Geometrie der Straße (MULTILINESTRING)
* in Tabelle vorhanden, wird aber noch nicht im Route Service verwendet		

Auf weitere möglichen Tags, die in einer Weiterentwicklung der OLS Route Service verwendet werden könnten, wird im Ausblick zu dieser Arbeit eingegangen (Kapitel *8 Zusammenfassung & Ausblick*). Eine Statistik von der Entwicklung der OSM Daten bei dieser Arbeit für den Route Service und genaue Zahlen der verwendeten Straßen befindet sich am Ende des Kapitels in *4.6 Statistiken der Datenaufbereitung*.

4.2 ... Web Feature Service & Web Map Service

Um OpenStreetMap Daten in einen *Web Feature Service* (WFS) und *Web Map Service* (WMS) verfügbar zu machen, gibt es verschiedene Lösungswege, die aber auch vom Resultat nicht ganz gleich sind. Sind Geodaten einmal in einer PostgreSQL/PostGIS Datenbank oder als Shapefile vorhanden, können sie in der Regel ohne Probleme in einen WMS und WFS, wie zum Beispiel dem GeoServer[8], verfügbar gemacht und genutzt werden. Im Folgenden werden zwei mögliche Varianten für die Datenaufbereitung vorgestellt: 1. Nutzung des Tools: osm2pgsql und 2. Nutzung von verfügbaren Shapefiles.

[8] GeoServer - http://geoserver.org

osm2pgsql

Im Kontext von Mapnik (2005) gibt es ein Tool namens *osm2pgsql*, welches OSM Daten direkt in eine PostgreSQL/PostGIS Datenbank konvertieren kann. Es ist für Linux als auch für Windows verfügbar. Aus einem beliebigen OSM Datensatz werden folgende vier Tabellen erstellt:

1. planet_osm_line - In dieser Tabelle sind alle Way-Objekte des angegebenen OSM Datensatzes enthalten, wie zum Beispiel: highway, waterway oder railway.
2. planet_osm_point - Die Tabelle enthält alle Node-Objekte, die mit weiteren Tags belegt sind, so zum Beispiel: places oder POIs.
3. planet_osm_polygon - Enthält alle möglichen Flächen, zum Beispiel: Landnutzung oder Gebäude.
4. planet_osm_roads - Tabelle die nur Straßen/Wege enthält.

Der Vorteil dieser Lösung ist, dass sehr viele Attribute aus dem OSM Datensatz mit in die Tabellen übernommen werden. Allerdings ist eine beispielsweise differenzierte Suche nach POIs nur in Verbindung mit einer Filterung bei der Abfrage der Point-Tabelle möglich. Je nach Datensatzgröße könnte es somit vorteilhafter sein, wenn zum Beispiel eine einzelne Tabelle nur mit POIs verfügbar wäre.

Shapefiles

Der zweite mögliche Lösungsweg der hier vorgestellt wird, ist die Nutzung von verfügbaren Shapefiles der Geofabrik, die auf deren Webseite zum Download angeboten werden. Im Download-Bereich können immer tagesaktuelle Shapefiles für verschiedene Länder Europas oder für die einzelnen Bundesländer Deutschlands heruntergeladen werden. In einem Shapefile-„Paket“ für Deutschland waren zur Zeit dieser Arbeit folgende Daten enthalten: *waterways* (Wasserwege), *roads* (Straßen/Wege), *railways* (Eisenbahnlinien), *points* (Punkte, wie z.B. POIs), *natural* (Naturflächen) und *buildings* (Gebäude). Diese Shapefiles können entweder direkt in einem WFS/WMS (zumindest im Falles des GeoServers) genutzt werden, oder, was der bessere Weg ist, sie werden in eine PostgreSQL/-PostGIS Datenbank importiert (shp2pgsql[9]). Gerade die Datenhaltung von größeren Datensätzen in Datenbanken bringen einen wichtigen Performancegewinn. Der Vorteil von diesem Lösungsweg ist, dass durch die verschiedenen Dateien bereits getrennte Layer für Straßen, Gebäude oder auch POIs möglich sind. Nachteile hingegen sind das die Daten vereinzelt lückenhaft oder Features ohne Geometrien vorhanden waren.

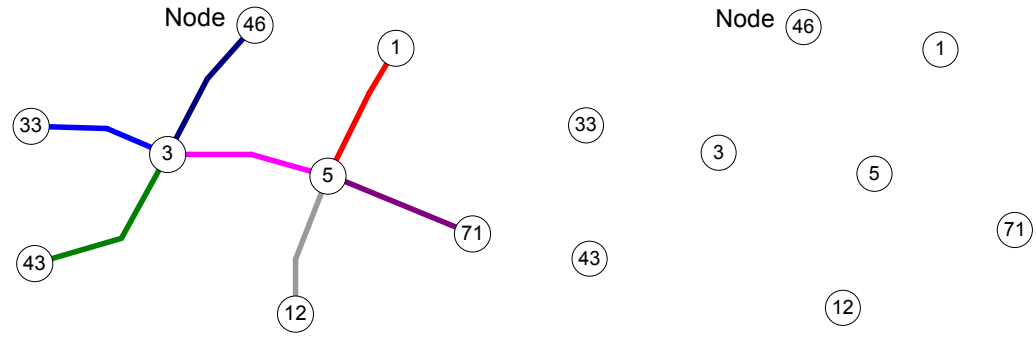
Anfangs erfolgte die Realisierung eines WFSs und WMSs unter Verwendung dieser Shapefiles. Aber nach immer fortschreitender Entwicklung der Datenaufbereitung für den

[9] shp2pgsql - http://www.postgis.fr/tarball/postgis-cvs/loader/README.shp2pgsql

OLS Route Service zeichneten sich als Nebenprodukte eigene Tools auf, mit deren Hilfe mehrere und teilweise vollständigere Daten-Layer aus den „rohen“ OSM-Daten erstellt werden können. Somit sind ebenfalls wieder Java-Konsolenanwendungen für die Aufbereitung vorhanden. Folgende Layer können mit Hilfe dieser erstellt werden: Gebäude, Naturflächen/Landnutzung, Ortsnamen, POIs, Wasserwege, Straßen und Ampeln/Mini-Kreisel. Wie die entwickelten Tools genutzt werden können, wird im Kapitel *7.1 Datenaufbereitungs-Tools* beschrieben.

4.3 ... Accessibility Analysis Service

Wie im Kapitel *2.6* beschrieben, handelt es sich in dem von Neis et al. (2007) realisierten *Accessibility Analysis Service* (AAS) um eine Web-basierte Erreichbarkeitsanalyse, die als Grundlage ein Routing-fähiges Straßennetz benötigt. Dieses ist bereits durch die Datenaufbereitung für den OLS Route Service vorhanden. Weiterhin werden, um den vollen Funktionsumfang zu bieten, noch zwei weitere Datensätze benötigt. Der wichtigere von beiden enthält alle Knotenpunkte (Nodes) des Routing-Netzes (siehe Abbildung 4.4). Diese sind für die Berechnung für das Polygon, das die Erreichbarkeit repräsentiert, erforderlich.



(a) OSM Routing-Netz für OLS Route Service (b) Knoten des Routing-Netzes für AAS

Abbildung 4.4: OSM Daten für Accessibility Analysis Service

Wie bei der Vorstellung erwähnt, war eine der Besonderheiten des Services, dass er auch die Strecken und Zeiten von beliebigen Punkten (zum Beispiel POIs oder Städten/Orten), die sich innerhalb des Polygon befinden, zurückgeben kann. Hierfür kann entweder das vorhandene POI- oder Ortsnamen-Layer des WFS verwendet werden.

4.4 ... OLS Location Utility Service

Der OLS Location Utility Service ist ein Geocoder und Reverse Geocoder (siehe Kapitel *2.4 OpenGIS Location Services*). Es kann mit dem Dienst eine Ortsangabe (Adresse) in eine Koordinate und eine Koordinate in eine Ortsangabe umgewandelt werden. Als Datengrundlage für die Adress-Suche des Geocoders wurde unter anderem eine Art *„Adressbuch“* entwickelt, dieses besteht aus folgendem Tabellenschema:

Tabelle 4.5: Tabellenschema für OLS Location Utility Service (Geocoder)

Attribut	Datentyp	Beschreibung
id	INTEGER	ID
countrycode	STRING	Name des Landes (CountryCode), Beispiel: ’de’ für Deutschland
countrys	STRING	Bundesland (CountrySubdivision), Beispiel: ’Nordrhein-Westfalen’
postalcode	STRING	Postleitzahl (PostalCode), Beispiel: ’53115’
municipal	STRING	Ortname (Municipality), Beispiel: ’Bonn’
municipals*	STRING	Ortsteilname (MunicipalitySubdivision)
strname	STRING	Straßenname (StreetName), Beispiel: ’Meckenheimer Allee’
housenr*	STRING	Hausnummer (Housenumber), Beispiel: ’172’
sdivision*	STRING	Hausnummernzusatz (Subdivision), Beispiel: ’a’
GEOMETRY	GEOMETRY	Geometrie der Adresse (POINT)
* in Tabelle vorhanden, wird aber noch nicht vom Service verwendet		

Ziel der Datenaufbereitung für den Geocoder ist es, die Tabelle anhand der OSM Daten auf verschiedene Weise zu befüllen. Folgende Möglichkeiten oder auch Kombinationen einer Suche nach einer Adresse sollen verfügbar sein: Suche nach einem ...

1. Bundesland
2. Postleitzahl
3. Ortsnamen wie Stadt, Dorf oder auch Stadtteil
4. Straßenname

Da Postleitzahlen und auch deren Polygon noch nicht richtig bei OSM verwendet werden, wurde für die Aufbereitung des Adressbuches ein Bundesland- und ein Postleitzahlendatensatz von *ESRI Data & Maps 2004*[10] verwendet. Bei der Datenaufbereitung werden die Nodes aus den OSM Daten, die den in der Tabelle 4.6 gekennzeichneten `place`-Key und

[10] Redistribution rights - http://webhelp.esri.com/arcgisdesktop/9.1/index.cfm?ID=2039&TopicName=Redistribution%20rights&rand=636&pid=2036

einen `name`-Key enthalten, dem Adressbuch hinzugefügt.

Tabelle 4.6: OSM *„Key-Value“*-Paare für OLS Location Utility Service

Key	Value	Beschreibung
place	city	Großstadt mit > 100.000 Einwohner
	town	Stadt mit > 10.000 Einwohner
	village	Dorf mit < 10.000 Einwohner
	hamlet	Weiler (Ansammlung einiger Häuser)
	suburb	Stadtteil, Ortsteil
name	Benutzerdefiniert	allgemeine Bezeichnung, hier: Name des Ortes (place-Value) oder Name der Straße
ref	Benutzerdefiniert	Straßennummerierung

Ways, die den `highway`-Key und den `name`- oder `ref`-Key enthalten, kommen ebenfalls in das Verzeichnis. Jedem Eintrag im Adressbuch werden über Räumliche-Suchen die fehlenden Attribute hinzugefügt. Das bedeutet: Anhand der Koordinate des Objektes und der Point-In-Polygon-Methode kann in den Datensätzen der Bundesländer und der Postleitzahlenbereichen herausgefunden werden, in welchem Bundesland und Postleitzahlenbereich sich das Objekt befindet. Bei den Orten werden auf diese Art und Weise folgende Attribute hinzugefügt: Name des Staates, des Bundeslandes und die Postleitzahl. Den Straßen müssen die gleichen Attribute, wie auch der Ortsname, in dem sich die Straßen befindet, bei jedem Eintrag ergänzt werden. Wie bereits bei der Datenaufbereitung für den Route Service, müssen auch hier wieder die Namen der Straßen mit Abkürzungen („str.“ oder „Str.“) korrigiert werden.

Für die Reverse-Geocodierungs Funktionalität des Location Utility Service wurden noch zwei weitere Tabellen erstellt. Zum einen ein sogenanntes „Straßenbuch“, das die gleichen Attribute wie das „Adressbuch“ enthält, sich allerdings von der Geometrie unterscheidet. War es beim Adressbuch eine Punktgeometrie für jede Adresse, sind es beim Straßenbuch Liniengeometrien, die eine verbesserte Suche beim Reverse-Geocodieren ermöglichen. Daneben gibt es noch eine weitere Tabelle: das „Postleitzahlenbuch“. Dieses findet aber nur Anwendung, wenn keine Straße in einem gewissen Abstand zur angegebenen Koordinate im „Straßenbuch“ gefunden werden kann.

Die OSM Daten bieten auch für diesen Service wieder ein größere Anzahl von möglichen Tags an, unter anderem zum Beispiel Hausnummern. Weil diese aber zu der Zeit dieser Arbeit noch keine ausreichende Verwendung im OSM Projekt fanden, wurden sie hier nicht beachtet. Das Tool, um die Daten aufzubereiten, ist wieder eine Java-Konsolenanwendung und die Nutzung wird im Kapitel *7.1 Datenaufbereitungs-Tools* beschrieben.

4.5 ... OLS Directory Service

Die OSM Daten für einen OLS Directory Service (Online-Branchenverzeichnis) können ohne großen Aufwand aus den OSM Daten ausgelesen werden. Dabei bieten die *Map Features* (siehe Kapitel *3.3*) eine vielseitige und ständig zunehmende Auswahl von möglichen Inhalten für einen Verzeichnisdienst. Die hier vorgestellten und verwendeten waren während dieser Arbeit die vorhandenen und meist genutzten. Sie können entweder im Node-Objekt oder im Way-Objekt im OSM Datensatz vorkommen. Des Weiteren wurden sie für den in dieser Arbeit realisierten OLS Directory Service noch in Kategorien eingeteilt, um dem Nutzer zum Beispiel nicht nur die Suche nach der nächsten Bushaltestelle zu ermöglichen, sondern auch nach anderen in der Nähe befindlichen öffentlichen Verkehrsmitteln (siehe Tabelle 4.7 Spalte *Kategorie*).

Tabelle 4.7: OSM *„Key-Value“*-Paare für OLS Directory Service

Key	Beschreibung (Value)	Kategorie
highway	Bushaltestelle (bus_stop)	public_transport
railway	Bahnhof (station), Straßenbahnhaltestelle (tram_stop), U-Bahn Eingang (subway_entrance)	public_transport
amenity	Geldautomat (atm), Bank (bank), Geldwechselbüro (bureau_de_change), Biergarten (biergarten), Busstation (bus_station), Cafe (cafe), Kino (cinema), College (college), Gericht (courthouse), Schnell-Restaurant (fast_food), Tankstelle (fuel), Krankenhaus (hospital), Bibliothek (library), Nachtclub/Disco (nightclub), Parkplatz (parking), Apotheke (pharmacy), Anbetungsort (place_of_worship), Polizeistation (police), Briefkasten (post_box), Postamt (post_office), Kneipe (pub), Öffentliches Gebäude (public_building), Restaurant (ohne Fast-Food) (restaurant), Schule (school), Taxi(taxi), Telefon(telephone), Theater (theatre), Öffentliche Toilette (toilets), Rathaus (townhall), Universität (university)	amenity
shop	Supermarkt (supermarket), Lebensmittelgeschäft (convenience), Bäckerei (bakery), Metzger (butcher), Kiosk (kiosk)	shop
tourism	Touristeninformation (information), Hotel (hotel), Motel (motel), Gästehaus (guest_house), Jugendherberge (hostel)	tourism

4.6 Statistiken der Datenaufbereitung

Im Folgenden werden zwei Datensätze für Deutschland gegenüber gestellt, um die Entwicklung der OpenStreetMap Daten für Deutschland und die mit dieser Arbeit bereitgestellten Location Based Services zu zeigen. Der erste OSM Datensatz für Deutschland ist vom Anfang der Arbeit (19.03.2008). Durch bz2-Komprimierung hat er eine Größe von ca. 90 MB. Die komplette Datenaufbereitung wurde mit einem Desktop Rechner (Quad CPU @ 2.40GHz und 3.25GB RAM) durchgeführt und dauerte, inklusive Importierung der Daten in eine Datenbank, 1 Stunde und 12 Minuten.

Wichtig: Die folgenden Zahlen entstanden durch die Aufbereitung der OSM Daten mit eigenen entwickelten Anwendungen und für die in dieser Arbeit verwendeten und neu implementierten Services. Die tatsächliche Anzahl der jeweiligen Features kann von den hier stehenden abweichen. Bedingt zum Beispiel durch eine nicht „korrekte Schreibweise“ oder Problemen beim Export der Geometrien (z.B. nicht geschlossenes Polygon).

Tabelle 4.8: OSM Deutschland Datensatz vom 19.03.2008

Layer (Gesamtzahl)	Enthaltende Typen (Anzahl)	Dauer
Gebäude (11080)		33 Sec.
Naturflächen & Landnutzung (43097)	forest (12450), water (10915), wood (6007), residential (4659), park (4031), industrial (1710), allotments (1533), commercial (593), retail (582), riverbank (316), scrub (277), fell (24)	32 Sec.
Ortsnamen (27393)	village (18932), hamlet (3135), suburb (2969), town (2259), city (98)	26 Sec.
POIs (50645)	44 verschiedene, die 10 meist verwendeten: parking (13416), bus_stop (3830), place_of_worship (3336), railway_station (3186), fuel (3125), restaurant (2737), school (2650), supermarket (2551), post_box (1773), telephone (1724)	38 Sec.
Straßen & Wege (551956)	224 verschiedene, die 16 meist verwendeten: residential (223152), footway (59787), secondary (50364), unclassified (45834), track (33851), tertiary (29367), service (28884), primary (20514), cycleway (15668), motorway_link (11612), motorway (9977), pedestian (5284), steps (4176), trunk (3854), trunk_link (3539), primary_link (3332)	44 Sec.
Objekte an Kreuzungen (10355)	traffic_signals (8960), mini_roundabout (728), crossing (551), stop (116)	24 Sec.

Straßen & Wege für Routing (1149410)	residential (477775), secondary (144145), footway (100784), unclassified (87926), tertiary (69971), primary (62127), track (60789), service (48695), cycleway (29757), motorway (19728), motorway_link (14967), pedetrian (10449), trunk (7827), trunk_link (4449), steps (4441), primary_link (4260), living_street (954), bridleway	24 Sec.
Straßen-Knoten(742989)		53 Sec.
Wasserwege (3556)	river (1722), stream (1324), canal (433), drain (77)	4 Sec.
Adressbuch (224629)		23 Min. 15 Sec.
Straßenbuch (304223) & Postleitzahlen (8719)		27 Min. 53 Sec.

Im Laufe dieser Arbeit wuchs der OSM Deutschland Datensatz bereits so stark an, dass er nicht mehr mit dem Desktop Rechner des ersten Datensatzes (zumindest unter Windows XP Pro SP2) verarbeitet werden konnte. Die Aufbereitung der Daten wurde ca. ab Mitte dieser Arbeit auf einen Server (2 x Quad CPU @ 2.83GHz und 32GB RAM) ausgelagert und benötigte für den Datensatz (29.07.2008 und ca. 188 MB) eine Zeit von 1 Stunde 52 Minuten. In der folgenden Tabelle können die einzelnen Anzahlen der Layer wieder entnommen werden.

Tabelle 4.9: OSM Deutschland Datensatz vom 29.07.2008

Layer (Gesamtzahl)	Enthaltende Typen (Anzahl)	Dauer
Gebäude (45720)		1 Min. 39 Sec.
Naturflächen & Landnutzung (87602)	forest (23953), water (20950), residential (11953), park (10499), wood (9697), industrial (3844), allotments (3187), commercial (1321), retail (1277), scrub (544), riverbank (350), fell (27)	1 Min. 35 Sec.
Ortsnamen (37126)	village (24023), hamlet (6378), suburb (4489), town (2148), city (91)	39 Sec.
POIs (120821)	44 verschiedene, die 10 meist verwendeten: parking (29235), bus_stop (12681), place_of_worship (7869), restaurant (7247), supermarket (6608), school (6415), fuel (5972), post_box (5041), railway_station (4021), telephone (3845)	1 Min. 38 Sec.
Straßen & Wege (1104452)	225 verschiedene, die 16 meist verwendeten: residential (435849), footway (138143), track (114246), unclassified (77418), secondary (72121), service (69011), tertiary (48941), cycleway (35260), primary (26233), motorway_link (15406), motorway (14501), living_street (12009), steps (10777), pedestrian (9201), trunk (5634), trunk_link (5250),	1 Min. 49 Sec.

Objekte an Kreuzungen (20533)	traffic_signals (17531), crossing (1507), mini_roundabout (1175), stop (319)	1 Min. 40 Sec.
Straßen & Wege für Routing (2365274)	residential (962365), secondary (240957), footway (238839), track (209991), unclassified (159082), tertiary (143892), service (120159), primary (92018), cycleway (72516), motorway (25998), living_street (22824), pedetrian (20335), motorway_link (19999), steps (11619), trunk (10771), trunk_link (6706), primary_link(6090), bridleway (1107)	5 Min. 22 Sec.
Straßen-Knoten (1456288)		18 Sec.
Wasserwege (7000)	stream (3366), river (2593), canal (826), drain (215)	1 Min. 44 Sec.
Adressbuch (422837)		25 Min. 7 Sec.
Straßenbuch (589876) & Postleitzahlen (8719)		29 Min. 15 Sec.

Aus diesen beiden Ergebnissen der Datenaufbereitung können allgemeine wie auch Typspezifische Werte für die Entwicklung der OSM Daten für Deutschland entnommen werden. Sie wurden zwar nicht auf Basis der OSM Datenbank erzeugt, dürften jedoch trotzdem mehr als eine grobe Analyse und Statistik für die Entwicklung des OSM Deutschland Datenbestandes für den Zeitraum Mitte März bis Ende Juli 2008 sein.

Tabelle 4.10: Statistik für OSM Deutschland Daten vom 19.03.2008 bis 29.07.2008

Layer	19.03.2008 Anzahl	29.07.2008 Anzahl	Zuwachs in %
Gebäude	11080	45720	+312%
Naturflächen & Landnutzung	43097	87602	+103%
Ortsnamen	27393	37126	+ 36%
POIs	50645	120821	+138%
Straßen & Wege	551956	1104452	+100%
Objekten an Kreuzungen	10355	20533	+ 98%
Straßen für Routing	1149410	2365274	+105%
Wasserwege	3556	7000	+ 96%
Adressbuch	224629	422837	+ 88%
Straßenbuch	304223	589876	+ 94%

Wie in der Tabelle 4.10 zu sehen ist, hat sich die Datenmenge im gesamten Durchschnitt etwa verdoppelt. Lediglich ein verstärkter Zuwachs von Gebäuden und nur ein kleiner Zuwachs von Ortsnamen heben sich hervor. Die Auswirkungen der generellen Datenaufbereiung der OSM Daten für das Routing können etwa mit der Verdopplung der Straßen-Datenmenge (*Straßen & Wege* vs. *Straßen für Routing*) angenommen werden.

5 Verwendete Software

In diesem Kapitel werden jeweils kurz die für diese Arbeit wichtigen und verwendeten Anwendungen, wie zum Beispiel das Datenbanksystem, die verwendete Web Map und Web Feature Service Implementierungen oder ein Map-Viewer für die Ansicht in einem Webbrowser vorgestellt.

5.1 PostgresSQL

Räumliche Datenbanksysteme gibt es heutzutage einige. Seit den letzten Jahren hat sich aber verstärkt die Nutzung PostgreSQL[1] mit der Erweiterung PostGIS[2] verbreitet. PostgreSQL ist *Open-Source Software* (OSS) für ein *objektrelationales Datenbankmanagementsystem* (ORDBMS). Die Entwicklung begann Ende 1970 als universitäres Projekt an der University of California at Berkeley Computer Science Department. Heute wird PostgreSQL von verschiedenen internationalen Gruppen von Befürwortern von Open-Source Software weiterentwickelt (Douglas & Douglas, 2005).

Durch eine Erweiterung Names PostGIS, bietet PostgreSQL ein Unterstützung von geografischen Objekten. Dies bedeutet, durch die Erweiterung kann PostgreSQL als eine räumliche (spatial) Datenbank verwendet werden. Somit können *Geo*-Typen, wie zum Beispiel Point, Line oder Polygon (vgl. SFS (1999), in der Datenbank gespeichert und vor allem über die angebotenen *Geo*-Operationen verwendet werden. Über diese Operationen werden zahlreiche räumliche Analyse- und Abfragefunktionen zur Verfügung gestellt, wie zum Beispiel: Funktionen für die Berechnung von Flächen und Distanzen, Verschneidung, Berechnung von Pufferzonen oder räumliche Operatoren wie Overlaps, Within, Contains etc. Die Daten können weiterhin über ein R-tree räumlich indiziert und in verschiedenen Projektionen genutzt werden. Für diese Arbeit kamen für die Aufbereitung der OSM Daten und Entwicklung der unterschiedlichen Anwendungen, Postgres 8.2 & 8.3 und PostGIS 1.3.2 zum Einsatz.

[1] PostgreSQL - http://www.postgresql.org/
[2] PostGIS - http://postgis.refractions.net/

Es gibt auch ein Modul namens pgRouting[3], das PostGIS die Funktionalitäten für Routenplanungen hinzufügt. Es wird im Rahmen des Projektes PostLBS[4] entwickelt. Dieses Projekt hat das Hauptziel, Routing und Geocodierung für LBS als OSS zur Verfügung zu stellen. Für diese Arbeit wurde dieses Modul untersucht und die Ergebnisse werden im Kapitel *6.1.2 Route Service* beschrieben.

5.2 WFS & WMS (GeoServer)

Für was und warum überhaupt ein WFS und WMS?
Aufgrund der bereits vorhanden und ähnlichen Services des OpenStreetMap Projektes ist dies keine unberechtigte Frage. Keiner des OSM Services ist OGC konform, was allerdings von Seiten des OSM Projektes auch so gewollt ist (siehe Kapitel 1.3 Motivation). Ein OGC WFS kann in vielen freien und proprietären GI-Systemen eingebunden, für attributelle und räumliche Analysen/Abfragen genutzt werden. Das Gleiche gilt auch für den WMS. Ein weiterer Grund den WMS einzurichten war der OLS Route Service. Dieser, beziehungsweise der OLS Presentation Service, benötigt einen WMS, um eine Karte mit der eingezeichneten Route erstellen zu können. Weiterhin besitzt ein SLD-fähiger WMS die Besonderheit, dass „beliebige“ Karten über SLD und nach eigenen Wünschen generiert werden können.

WFS und WMS Implementierungen sind durch GeoServer[5] oder deegree[6] als OSS vorhanden. Durch eigene und bisherige positive Erfahrungen wurde für diese Arbeit ein GeoServer verwendet. Er ist in den Bereichen wie die einfache Installation, Konfiguration oder verschiedener Nutzungsfunktionalitäten anderen Implementierungen etwas im voraus. Auch im Vergleich der Performance schneidet der GeoServer entsprechend gut gegenüber anderen OSS WMS ab (vgl. Bremm (2007)).

Bei der Konfiguration des GeoServer für diese Arbeit wurden folgende Änderungen vorgenommen:

- Nicht verwendete Layer (FeatureTypes), die standardmäßig bei einer Installation vorhanden sind, wurden entfernt.
- Die Anzahl der Feature, die maximal bei einer GetFeature-Anfrage erhalten werden können, wurde auf 1000 begrenzt. Hintergrund: Würde hier keine Begrenzung eingefügt werden oder die Grenze beispielsweise bei 50.000 liegen, könnte über die

[3] pgRouting - http://pgrouting.postlbs.org/wiki
[4] PostLBS - http://www.postlbs.org
[5] GeoServer - http://geoserver.org
[6] deegree - http://www.deegree.org/

GetFeature-Anfrage ein komplettes Layer aus dem WFS abgerufen werden.

- Die generelle WFS-T Funktionalität des GeoServers wurde deaktiviert, damit kein unbeabsichtigtes ändern der Daten möglich ist.
- Die Log-Stufe wurde auf „Production“ gesetzt und die Standard-Out der Anwendung deaktiviert, um unnötige Log-Einträge und große Log-Dateien zu unterbinden.
- Als letztes wurde die Startseite, um den GeoServer zu konfigurieren, umbenannt.

Mit Hilfe der unter Kapitel *4.2 WFS & WMS* aufbereiteten Shapefiles aus den OSM Daten und mit deren Importierung in eine PostGIS Datenbank, können die Daten ohne größere Probleme im GeoServer verfügbar gemacht werden. Für die Nutzung der Daten im WMS wurden zusätzlich für jedes Layer eine SLD Datei erstellt (siehe unter Kaptiel *2.5* Abschnitt *Styled Layer Descriptor (SLD)*). In einer SLD Datei kann jeweils für ein Layer folgendes festgelegt werden:

1. *In welchem Maßstabsbereich sollen die Features gezeichnet werden?* alternativ auch: *Bis zu welchem Maßstab* oder *Ab welchem Maßstab?*
2. *Wie sollen Features gezeichnet werden?* Farbe, Punktgröße, Linien-Art/Breite, Transparenz etc. Weiterführend kann auch die Darstellungsart vom Maßstab abhängig gemacht werden.
3. *Sollen sie eine Beschriftung erhalten?* Schriftart, Schriftstärke, etc.
4. *Darstellung über eine Alternative, zum Beispiel Symbole?* Symbole müssen entsprechend aufbereitet werden (Größe, Transparenz)

Um mit Hilfe des WMS ein gutes Kartenbild zu erhalten, wurden, neben der Erstellung der SLDs, zwei Layer zusätzlich durch eine Generalisierung überarbeitet. Durch die große Anzahl der Features und dem Detailreichtum der Geometrien waren manche Kartenabschnitte zu überladen und dauerten beim Rendering durch den WMS zu lange. Für die Generalisierung von Linien wurde eine vorhandene und eigene Implementierung des Douglas-Peucker-Algorithmus verwendet (vgl. Douglas und Peucker (1973)). Für Polygone wurde diese entsprechend erweitert, sodass sie auch bei Flächen verwendet werden kann.

Im Rahmen dieser Arbeit kamen unterschiedliche Versionen des GeoServers zum Einsatz. Arbeitete die GeoServer Installation der Version 1.5.3 anfangs noch ohne Probleme, zeigten sich durch die vor allem verstärkte Nutzung des WMS verschiedene Probleme. Aber auch bei den folgenden Versionen kamen nach der lokalen Installation auf einem PC oder einem Server, immer wieder unterschiedliche Probleme während des Betriebes

zum Vorschein:

- *Version 1.5.3* - Während des Betriebes über mehrere Tage und normaler bis intensiver Nutzung baut der GeoServer eine Vielzahl von Verbindungen zu der Datenbank auf, in der die Daten vorliegen. Dies führt zum einen zu einem erhöhten RAM-Verbrauch und unter Umständen zu einer Übertretung der erlaubten Verbindungsanzahl der Datenbank.
- *Version 1.6.3* - Es können zwar verschiedene Stufen des Logging bei einer GeoServer Installation eingestellt werden, jedoch kam es bei dieser Version vor, dass unnötige Standard-Outs des GeoServers die Log-Datei des Tomcat Servers füllten. Weiterhin gab es Probleme mit der Unterstützung von Daten, die in einer Datenbank in EPSG:4326[7] vorliegen, und als EPSG:900913[8] Karte abgerufen werden sollen.
- *Version 1.6.4a* - Traten in dieser Version die Probleme der anderen Versionen nicht mehr auf, gab es ein neues Problem: es konnten keine Symbole mehr via SLD in die Karte gezeichnet werden.

Zuletzt wurde die *Version 1.6.4b* verwendet, bei der bis zum Ende dieser Arbeit keine Probleme mehr aufgefallen sind.

5.3 OpenLayers & TileCache

Für die Visualisierung von Karten und deren Bereitstellung über das Internet gibt es verschiedene Anwendungen. Als möglichen Kartendienst wurde bereits der GeoServer WMS vorgestellt. Im Folgenden wird zunächst der zum Einsatz gekommene Map Client vorgestellt.

OpenLayers[9] ist ein frei verfügbarer Map-Client für beliebige Webbrowser und basiert auf JavaScript. Er kann OGC konforme Web Map und Web Feature Services, wie den UMN MapServer oder den wie bei dieser Arbeit verwendeten GeoServer, ansprechen, kombinieren und Karten oder andere Daten von ihnen anfordern. Ein großer Vorteil ist, dass es sich um ein sehr aktives Projekt handelt, es in einigen Projekten verwendet und ebenfalls von der OSS Gemeinde weiterentwickelt wird. Vom Erscheinungsbild im Web Browser ähnelt es Google Maps, jedoch unterscheidet sich OpenLayers von Google durch eine weit größere Funktionsvielfalt. Zu Beginn dieser Arbeit lag OpenLayers noch in Version 2.5 vor, nach kurzer Zeit gab es aber den Wechsel zu Version 2.6, die auch hier ver-

[7] EPSG:4326 = WGS84
[8] EPSG:900913 = sphärische Mercator-Projektion die von TileCache oder Google Maps oder Earth verwendet wird
[9] OpenLayers - http://openlayers.org/

wendet wurde. Wie das Ergebnis der entwickelten Webseite mit OpenLayers aussieht und welche Änderungen und Entwicklungen noch gemacht werden mussten, wird im Kapitel *6.3 OpenRouteService.org* gezeigt. Mit der Onlinestellung der entwickelten Website und der damit verbundenen Verwendung des eingerichteten GeoServers zeigte sich jedoch, dass dieser nicht für die Integration auf einer Webseite geeignet ist. Karten müssen immer wieder gerendert werden, obwohl sie vielleicht gerade ein paar Sekunden zuvor durch einen anderen Nutzer abgerufen wurden. Auch die Performance der Kartenvisualisierung und die Interaktion mit einen WMS in Abhängigkeit mit dem Umfang des Karteninhaltes, ist nicht unbedingt optimal auf einer gut besuchten Webseite. Eine Alternative bietet hier die Installation eines TileCache.

TileCache

Wie bereits in Kapitel 3.4 bei der Beschreibung der OSM Karten erwähnt, benötigt das Rendern von Karten Rechenpower und -zeit. Üblicherweise werden aus diesem Grund Karten vorgerechnet und als Bild auf einem Server abgelegt. Der entscheidende Vorteil, der daraus entsteht, ist folgender: ein Map Client, der von einem solchen Server Karten anfordert, kann sie direkt von ihm erhalten, es entsteht die „Slippy Map“. Dies ist ein großer Vorteil gegenüber einer Realisierung durch einen WMS. Bei diesem wird in der Regel erst nach einer Anfrage die Karte gerendert. TileCache[10] ist ein *Web Map Service-Client* (WMS-C) oder *Tile Map Service* (TMS). Er speichert „vorweg berechnete“ Karten in einer Verzeichnisstruktur ab und kann diese somit schnell und einfach auf eine Anfrage zurückliefern. Die Welt-Karte oder jeder anderer beliebiger Ausschnitt wird dabei in Reihen und Spalten zerlegt, wodurch immer gleichgroße und quadratische „Kacheln“ (Tiles) für verschiedene Zoomstufen entstehen. Abbildung 5.1 zeigt die Zerlegung am Beispiel von Europa und danach Deutschland:

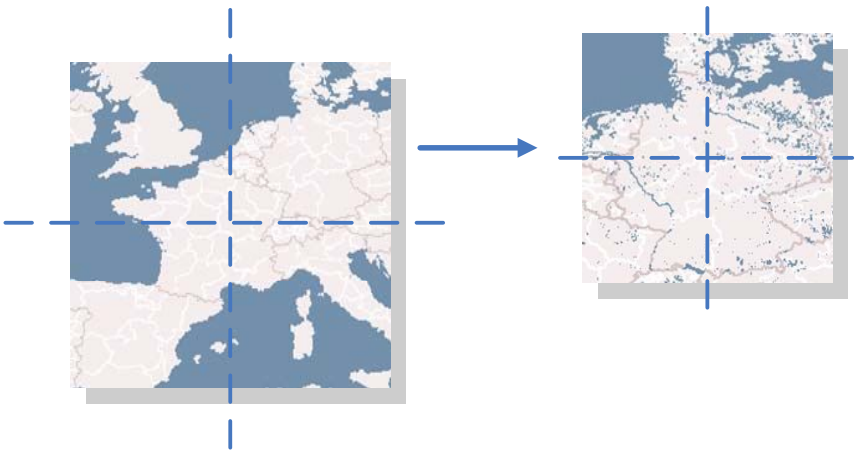


Abbildung 5.1: Funktionsweise TileCache

[10] TileCache - http://www.tilecache.org

Ein TileCache Server kann auf zwei unterschiedliche Weisen mit Kacheln „befüllt“ werden. Entweder durch eine komplette Vorwegberechnung (Skriptaufruf mit Boundingbox- und Zoomlevel-Parameter) oder durch Speicherung des erstmaligen Aufrufs einer Kachel. In beiden Fällen steht jedoch immer noch eine Anwendung im Hintergrund, die die Karten rendern muss. Nicht ganz außer acht gelassen werden darf dabei, dass diese Datenhaltung der Karten, je nach Boundingbox und Anzahl von Zoomstufen, einen nicht unerheblichen Speicherplatzbedarf erfordern kann. Für diese Arbeit wurde die TileCache Version 2.04 verwendet. Dadurch das hinter TileCache und OpenLayers die selben Entwickler stehen, ist eine gute Zusammenarbeit zwischen den Programmen gewährleistet.

6 Weiterentwickelte & neu implementierte Software

Wurden im letzten Kapitel die vorhandenen und verwendeten Anwendungen dargestellt, werden in diesem Kapitel alle weiterentwickelten und neu implementierten Anwendungen beschrieben. Aufgrund recht schneller Erfolge bei der Datenaufbereitung für den OLS Route Service, den Web Feature und Web Map Service und der Weiterentwicklung und Optimierung des Route Services wurden noch weitere Dienste mit OpenStreetMap Daten realisiert. Die folgende Abbildung 6.1 zeigt eine Gesamtübersicht von allen die in dieser Arbeit verwendeten, weiterentwickelten und implementierten Dienste.

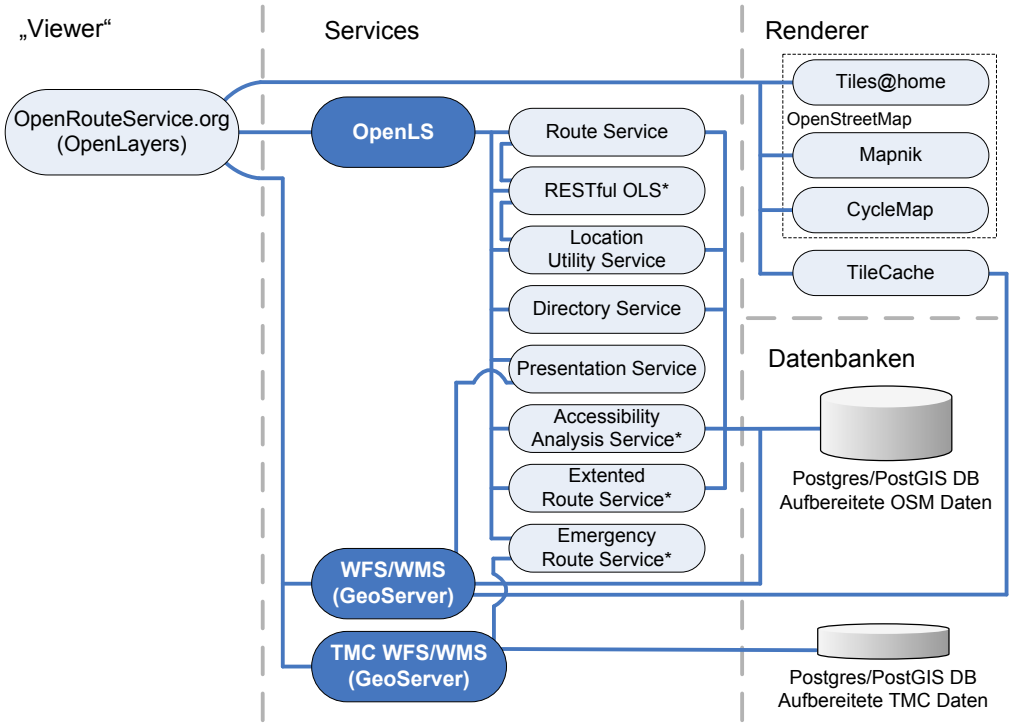


Abbildung 6.1: Gesamtarchitektur der verwendeten, weiterentwickelten und implementierten Dienste

Wie in der Abbildung 6.1 zu sehen, werden alle aufbereiteten OSM Daten für die Services von dieser Arbeit in einer PostGIS Datenbank vorgehalten. Die „Services“ in der Mitte

der Abbildung bestehen zum einem aus dem in dieser Arbeit entstandenen „OpenLS-Framework“ (OpenLS Service), aus dem mit Hilfe der aufbereiteten OSM Daten aufgesetzten GeoServer (WFS/WMS) und aus dem von Mayer (2008) entstandenen GeoServer mit *Traffic Message Channel* (TMC) Daten (mehr dazu im Unterkapitel *6.2.2 Emergency Route Service & 6.3 OpenRouteService.org*). Die in der Abbildung mit * gekennzeichneten Dienste des OpenLS Frameworks zählen nicht zu den Core Services, bauen aber durch deren Implementierung auf dem Framework auf und sind aus diesem Grund auch in dessen enthalten. Alle entstandenen und aufgesetzten Dienste dieser Arbeit, können über die entwickelte Webseite („Viewer“), die unter http://OpenRouteService.org online zufinden ist, genutzt werden. Für die Webseite stehen vier verschiedene Karten („Renderer“) zur Auswahl: entweder Tiles@home, Mapnik, die CycleMap (OSM Karte mit speziellen Styling-Parametern für Fahrradrouten) vom OpenStreetMap Projekt oder die Karte des aufgesetzten WMS TileCaches.

6.1 OpenGIS Location Services

Im Kapitel *2.4 OpenGIS Location Services* wurden bereits die Core Services vorgestellt. Die Grundlage für den Route Service bildete die Diplomarbeit von Neis (2006). Über die letzten zwei Jahre wurde dieser teilweise weiterentwickelt und fand in den Projekten MoNa3d[1], GDI-3D[2], OKGIS[3], RIMAX[4] und bei ReWoB[5] anwendung. Bei letzteren Projekt (ReWoB) kam allerdings nicht der RS zum Einsatz, sondern er wurde als Grundlage für die Entwicklung eines Web Erreichbarkeitsanalyse-Dienstes verwendet (Neis et al., 2007). Im Rahmen des Projektes OK-GIS und GDI-3D wurden einfache Versionen des OLS Location Utility (Geocoder/Reverse Geocoder) und Presentation Services entwickelt. Beides konnte als Grundlage für diese Arbeit verwendet werden, um weitere Untersuchungen einer möglichen Verwendung von OSM Daten durchzuführen. Vom OLS Diretory Service gab es keine vorhandene Umsetzung und dieser musste demzufolge komplett neu implementiert werden. Eine Realisierung des Gateway Services wäre zwar interessant und auch nützlich gewesen, allerdings benötigt er Zugriff auf einen Mobilfunkserver. Einen solchen Zugang von einem Mobilfunkbetreiber zu erhalten, dürfte äußerst schwierig sein. Ergänzend gibt es noch weitere Entwicklungen, die den OLS

[1] http://www.MoNa3D.de - „ Mobile Navigation 3D - 3D-Stadtmodelle für mobile Navigationssysteme“ (MoNa3D)

[2] http://www.gdi-3d.de - „Geodateninfrastruktur für 3D-Geodaten ; Auf dem Weg zu dienstebasierten interoperabelen 3D-Stadtmodellen am Beispiel von Heidelberg“

[3] http://www.OKGIS.de - „Offenes Katastrophenmanagement mit freiem GIS“ (OKGIS)

[4] http://www.rimax-hochwasser.de - „Risikomanagement extremer Hochwasserereignisse“ (RIMAX)

[5] http://www.ReWoB.de - „Regionalisierte Wohnungsmarktbeobachtung Rheinland-Pfalz“ (ReWoB)

Route Service als Grundlage verwenden. So zeigten Weiser, Neis und Zipf (2006), wie der Route Service im Kontext des Katastrophenmanagements als ein sogenannter *Emergency Route Service* (ERS) genutzt werden könnte (mehr zum ERS in Kapitel *6.2.2*). Ebenfalls unter Verwendung der Core Services zeigten Neis und Zipf (2007), wie „Focus-Maps“ im Zusammenhang mit der Visualisierung einer berechneten Route und der Integration von Landmarken verwendet werden könnte (*Extended Route Service* - mehr Informationen im Kapitel 6.2.3). Der RESTful OLS soll ein erster möglicher Prototyp sein, der eine vereinfachte Nutzung von verschiedenen OLS Services gemäß des *Representational State Transfer* (Akronym REST) bietet (Kapitel 6.2.4).

6.1.1 Architektur

Vor Beginn dieser Masterarbeit waren der Route Service, Location Utility Service und Presentation Service noch eigene und selbständige Dienste. Im Laufe dieser Arbeit wurden sie zusammengeführt und bilden jetzt, mit folgenden und weiteren nicht Core Services, das ***OpenLS Framework*** (siehe Abbildung 6.1 Gesamtarchitektur).

- OLS Route Service
- RESTful OLS
- OLS Location Utility Service (Geocoder/Reverse Geocoder)
- OLS Directory Service
- OLS Presentation Service
- Accessibility Analysis Service
- Extended Route Service (Route Service with Landmarks)
- Emergency Route Service

Die folgende Abbildung 6.2 zeigt den ***OpenLS Service*** der aus dem OpenLS Framework entstandenen ist und alle eben erwähnten Dienste enthält.

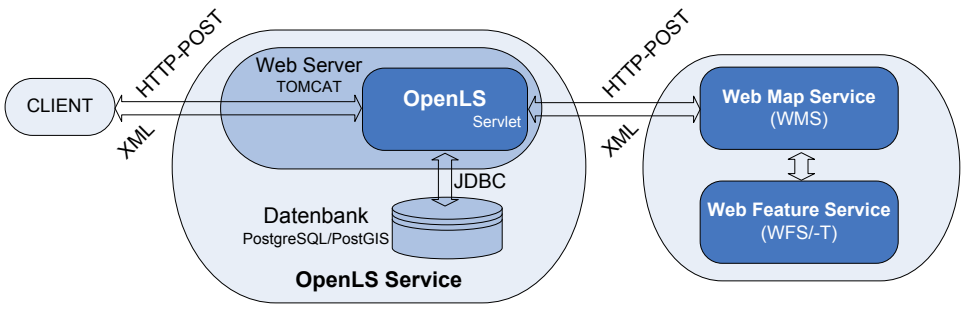


Abbildung 6.2: Architektur des OpenLS Service

Der Service ist in der Java Version 6.0 realisiert und läuft als Servlet in einem Tomcat[6] Server ab Version 5.5.x. Die Kommunikation erfolgt OLS typisch, über das *Hyper Text Transfer Protocol* (HTTP) POST und in Form von *Extensible Markup Language* (XML) Anfragen und Antworten ab. Alle benötigten Daten des Dienstes sind in einer PostgreSQL/PostGIS DB gespeichert. Der Informationsaustausch zwischen dem OpenLS Service und der Datenbank findet über die *Java Database Connectivity* (JDBC) statt. Weiterhin ist für die Visualisierung von Ergebnissen der Dienste ein WMS und für andere Datenoperationen ein WFS erforderlich.

Bei der Zusammenlegung der bestehenden Dienste wurden folgende generelle Änderungen durchgeführt:

- Die Daten für die Services müssen in WGS-84 Format vorliegen.
- Die interne Datenverarbeitung (zum Beispiel Distanzberechnungen) wurde entsprechend auf WGS-84 angepasst.
- Update der verwendeten GeoTools[7] Bibliothek von der Version 2.2 auf 2.4.2 und Update der *Java Topology Suite* (JTS) Bibliothek von Version 1.6 auf 1.8.
- Umstellung des Logging von der Java Logging API[8] auf log4j[9].
- Änderung der GeoTools Postgres/PostGIS EPSG[10] Koordinatentransformation auf eine in Java programmierte GeoTools SQL-Datenbank (HSQL).

Die Änderung der Datenhaltung und -verarbeitung wurde aus verschiedenen Gründen notwendig: Zum einen sind die OpenStreetMap Daten in WGS-84 vorhanden und zum anderen ist WGS-84 das „Standard-Referenzsystem“ für Geodaten. Diese Änderung musste zum Beispiel in Distanzberechnungen oder anderen Rechen- oder Suchoperationen entsprechend angepasst werden. Da es in der Vergangenheit manchmal Probleme mit der verwendeten Java Logging API gab, wurde diese komplett durch log4j ausgetauscht. Log4j überzeugt vor allem durch eine einfache Konfiguration, aber auch durch sehr viele mögliche und vielseitige Einstellungen. Es können zum Beispiel über eine Konfigurationsdatei verschiedene Datei-Logger, je Logstufe, Package oder Klasse, erzeugt werden. Auch das Format der einzelnen Log-Einträge kann beliebig für jeden Logger angepasst werden. Somit können für alle Dienste im OpenLS Service die jeweiligen Logger beliebig angepasst und eingestellt werde. Neben dem obligatorischen Error-Logger, der auftretende Fehler mitloggt, wurde für jeden Dienst ein Logger integriert, der die Zugriffe aufzeichnet. Bei

[6] Apache Tomcat - http://tomcat.apache.org
[7] GeoTools - The Open Source Java GIS Toolkit http://geotools.codehaus.org/
[8] Java Logging Technology - http://java.sun.com/javase/6/docs/technotes/guides/logging/index.html
[9] Apache log4j - http://logging.apache.org/log4j/
[10] European Petroleum Survey Group Geodesy (EPSG) - http://www.epsg.org/

einem Request an einen Service werden somit Datum, Uhrzeit, IP des Nutzers, Art der Anfrage (Parameter der Anfrage) und die Dauer der Bearbeitung mitgeloggt. Dieses soll insbesondere dazu dienen, genauere Informationen darüber zu erhalten, welche Arten von Anfragen gestellt werden und bei welcher Anfrage ein Fehler oder Problem aufgetreten ist. Bei der früheren Version des RS erfolgte die Koordinatentransformation mit Hilfe einer GeoTools Postgres/PostGIS EPSG Transformationsdatenbank. Weil aber nach der Installation des OpenLS Service auf einen Linux-Server diese nicht ohne Probleme zu nutzen war, wurde im OpenLS Service die Datenbank durch eine andere, von GeoTools bereitgestellte, ausgetauscht (GeoTools SQL-Datenbank (HSQL)).

6.1.2 Route Service

Vor der Beschreibung der Optimierungen, die an der vorhandenen Implementierung erfolgt sind, wird der OLS *Route Service* (RS) nochmals kurz vorgestellt.

Wie bereits im Kapitel *2.4 Route Service* beschrieben, ist er ein netzwerkbasierter Routenplanungs-Dienst. Dabei muss zwischen zwei verschiedenen Arten von Route-Requests unterschieden werden: Berechnung einer Route (Neuanfrage) und Neuberechnung einer bereits auf dem Server gespeicherten Route. Bei einer Neuanfrage müssen mindestens Start- und Zielpunkt sowie die Art der Routenplanung angegeben werden (zum Beispiel „Fastest“, „Shortest“ oder „Pedestrian“). Dabei können die Punkte als Adresse, POI, Punktkoordinate oder in Form einer anderen Geometrie (Polygon, Ellipse, Kreis etc.) angegeben werden. Zusätzlich können die Geometrien jeweils in einem unterschiedlichen *Spatial Reference System* (SRS) übergeben werden. Andere Parameter, wie Zwischenpunkte, Vermeidungsgebiete, Fahranweisungen, Karte(n) etc., sind bei der Anfrage optional.
Die Antwort des Route Service besteht mindestens aus einer Routen-Zusammenfassung. Diese enthält unter anderem die gesamte Reisezeit und -strecke. Optional können die eben erwähnten Fahranweisungen (RouteInstruction), Geometrien der Route (RouteGeometry) und Karte(n) (RouteMaps) enthalten sein. Einigen Parametern können zusätzliche Attribute bei einer Anfrage hinzugefügt werden, zum Beispiel: in welcher Sprache die Fahranweisungen erfolgen sollen, in welcher Einheit Streckenlängen angegeben werden oder in welchen SRS die Geometrie der Route zurückgegeben wird oder ob sie zusätzlich generalisiert werden muss. Weitere Informationen unter Neis (2006), Neis (2008) oder OLS (2005). Eine Beispiel Anfrage und die daraus resultierende Antwort von einem OLS RS wird im Anhang A.1 gezeigt. Die folgende Abbildung 6.3 enthält ein Sequenzdiagramm einer Anfrage an den OLS Route Service. Dabei können je nach Angabeart des Start- oder Endpunktes, der OLS Location Utility Service für die Geocodierung einer Adresse oder

der OLS Directory Service für eine Branchenverzeichnisabfrage nach einem POI zum Einsatz kommen. Der OLS Presentation Service ist für die optionale Kartenerstellung zuständig.

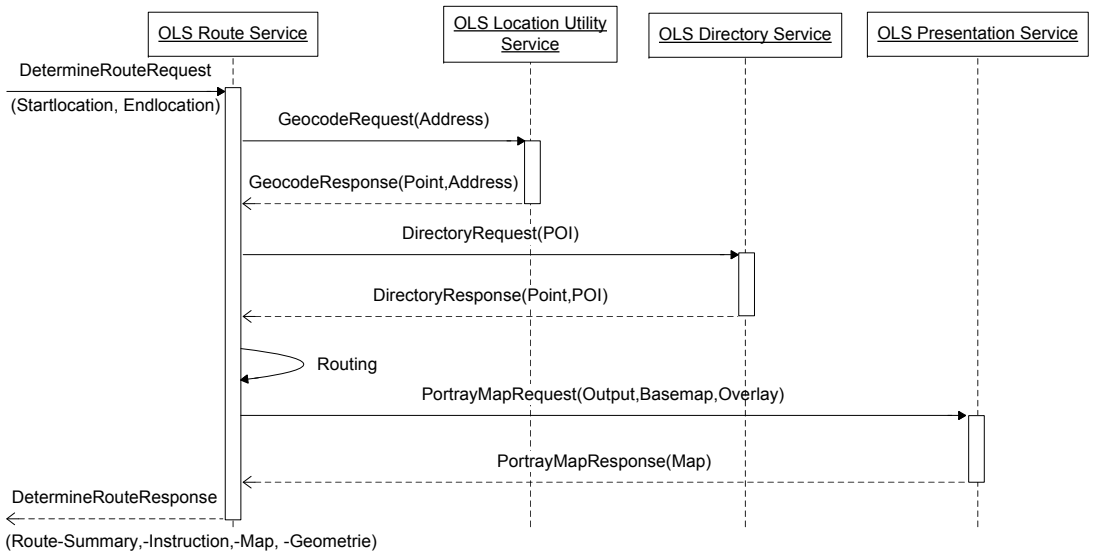


Abbildung 6.3: Sequenzdiagramm des OLS Route Service

Optimierungen und Überarbeitungen ...

Vorrangig sollte die vorhandene Implementierung des RS dahingehend getestet werden, ob sie mit großen Straßengraphen, in Form eines Datensatzes, für ein komplettes Land zurecht kommt. Nach der Aufbereitung der Routing-fähigen OSM Daten (siehe Kapitel 4.1) und der Integration zeigte sich jedoch schnell, dass der implementierte Routing-Algorithmus anscheinend nicht optimal arbeitet und somit ausgetauscht werden muss. Auch zeigten sich weitere Probleme bei der Beachtung von Einbahnstraßen am Start- oder Zielpunkt. Die Erstellung der resultierenden Fahranweisungen, inklusive der verwendeten Sprachen, wurden ebenfalls überarbeitet. Im Folgenden werden die wichtigsten Optimierungen, die am OLS Route Service erfolgt sind, vorgestellt.

Bei dem verwendeten Algorithmus handelte es sich fast um den Original aus der GeoTools-Bibliothek stammenden Dijkstra (1959) Routing-Algorithmus. Nach genauerer Untersuchung wurde folgendes Problem im Algorithmus entdeckt: Bei einer Routenberechnung wurde stets der komplette Graph durchgerechnet. Es erfolgte während der Berechnung keine Überprüfung ob der Zielknoten bereits erreicht ist und die Berechnung vorzeitig beendet werden kann. Die Kontrolle wurde nachgetragen, brachte allerdings nicht lange den gewünschten Erfolg. Im Laufe dieser Arbeit wuchs bereits der verwendete OSM Deutschland Straßensatz so stark an (siehe Kapitel 4.6), dass auch die Berechnung von kurzen Strecken keinesfalls „schnell“ erfolgte. Durch verschiedene Recherchen konnte eine fast vollständige Umsetzung des A*(„A Stern“)-Routing-Algorithmus, der im Rahmen

der nächsten GeoTools Version integriert wird, gefunden werden[11]. Der A*-Algorithmus (Hart, Nilsson & Raphael, 1968) nutzt im Gegensatz zum Dijkstra die Hilfe einer Heuristik bei der Berechnung der Route. Als Heuristik kann bei der Routenermittlung die berechnete Luftlinie des aktuellen Knotens zum End-Knoten verwendet werden. Da die gefundene Umsetzung bereits für die Integration in GeoTools vorgesehen war, musste unter anderem eine Klasse für die Bestimmung der eben beschrieben Heuristik implementiert werden. Weiterhin wurde von dem „neuen“ A-Stern Algorithmus eine erweiterte Version entwickelt, der ebenfalls wie bei Neis (2006) eine Liste von Kanten-IDs übergeben werden kann, die beim Routing nicht verwendet werden sollen. Das waren aber nicht die einzigen Änderungen an der eigentlichen Routenberechnung. Wurden früher die einzelnen Kantengewichte während des Routings berechnet, erfolgt nun die Ermittlung und die Speicherung in einer HashMap bei der Initialisierung des Route Services. Somit werden unnötige Berechnungen im Routing-Algorithmus vermindert und es bietet mehr Performancegewinne. Die Verwendung der vorwegberechneten Gewichte wurde im A-Stern- und im überarbeiten Dijkstra-Algorithmus integriert.

Als Alternative zum optimierten und verwendeten GeoTools Routing-Algorithmus wurde eine mögliche Nutzung von pgRouting untersucht. Hierfür gab es allerdings zur Testzeit keine brauchbare Installationsanleitung. Auch die als Stable-gekennzeichnete Version ließ sich nicht ohne Änderungen der SQL-Funktionen verwenden. Bei den Tests wurde der zur Anfangszeit verfügbare OSM Deutschland Datensatz verwendet. Für kleine Routenberechnungen können ähnliche Zeiten erreicht werden, wie mit dem eigenen GeoTools Routing-Algorithmus. Wurden allerdings Routen, beispielsweise von Frankfurt nach München (ca. 400 km), berechnet, erhöhte sich die Berechnungszeit von pgRouting in den zweistelligen Sekunden Bereich. Hier ist die vorhandene Umsetzung schneller. Ähnliche Erfahrungen, bezüglich der Routenberechnungen mit großen Datensätzen und längeren Routen, konnten von anderen Nutzern nicht gefunden werden. Lediglich bei der Nutzung der Beispiele[12] viel auf, dass bei längeren Routen (>100 km) ebenfalls eine längere Bearbeitungszeit (>8 Sekunden) entsteht oder Routenlängen gar auf eine maximale Länge von 20 oder 200 km begrenzt sind. Ein klarer Vorteil von pgRouting ist, der geringe Ressourcenverbrauch des Arbeitsspeicher. Werden die erzielten Ergebnisse mit den pgRouting Demo Seiten im WWW verglichen und sollte die Installation und die Nutzung von pgRouting bei dieser Arbeit richtig erfolgt sein, ist es keine Alternative zu der eigenen GeoTools Routing-Umgebung. Um „kleinere“ Routen (<100 km) ressourcensparend zu berechnen, ist es wiederum eine sehr gute Ausweichmöglichkeit.

[11] GeoTools - A Star algorithm http://jira.codehaus.org/browse/GEOT-1715
[12] pgRouting Demo - http://pgrouting.postlbs.org/wiki/pgRoutingDemo

Neben dem Routing-Algorithmus, wurden die verschiedenen Arten der Routenplanung um die Möglichkeit für Fahrradfahrer erweitert. Somit stehen nun folgende Varianten für eine Routenplanung zur Verfügung:

- Autofahrer: *Fastest*
- Autofahrer: *Shortest*
- Fußgänger: *Pedestrian*
- Fahrradfahrer: *Bicycle*

Für die zusätzliche Möglichkeit Fahrradfahrer *Bicycle*, wurde die OGC OLS (2005) Spezifikation „erweitert“. In diesem Fall (Element-`RoutePreference`) ist es aber von Seiten der Spezifikation vorgesehen, eigene Werte für weitere Varianten einzutragen.

Der Routing-Graph wurde früher jeweils im Kontext der Route Service und der Accessibility Analysis Service Implementierung erstellt und verwaltet. Durch die „Zusammenlegung“ von RS und AAS in ein Framework zeigte sich, dass es unnötig ist, den Routing-Graphen zweimal vorzuhalten. So wurde er aus beiden Services entnommen und wird zukünftig über ein neues Package initialisiert und kann von beiden Services genutzt werden. Innerhalb dieses Package werden drei Graphen für die eben erwähnten Varianten der Routenplanung erstellt. Dieses wurde aufgrund der unterschiedlich zu verwendenden Straßentypen, je Variante, so realisiert (Welche Straßentypen werden wo verwendet? Siehe Kapitel *4.1 Datenaufbereitung OLS Route Service*). Weiterhin bringt es den Vorteil, dass der AAS zukünftig auch für Fußgänger oder Radfahrer genutzt werden könnte. Momentan findet er nur Verwendung für Autofahrer. Wegen der Fülle von verschiedenen Straßentypen bei den OpenStreetMap Daten, wurde die Anzahl der im RS verwendeten entsprechend angepasst. Weil bei den OSM Daten nicht immer die Geschwindigkeiten bei jeder Straße mitangegeben werden und auch bei anderen Datensätzen, die im RS eingesetzt werden, nicht immer Geschwindigkeiten vorhanden sind, können diese über eine Konfigurationsdatei den individuellen Anforderungen angepasst werden.

Beim OSM Deutschland Datensatz vom 29.07.2008 entstanden durch die Aufbereitung für den RS insgesamt 2.365.274 Straßen & Wege. Weil nicht jede Straße bei jeder Routenplanung verwendet werden darf, entstehen folgende Kantenanzahlen für die einzelnen Varianten:

- Routing-Graph für Autofahrer (29.07.2008): *1.805.253* Kanten
  (im Vergleich Datensatz vom 19.03.2008 - *940.183* Kanten)
- Routing-Graph für Fußgänger (29.07.2008): *2.336.557* Kanten
  (im Vergleich Datensatz vom 19.03.2008 - *1.113.748* Kanten)

- Routing-Graph für Fahrradfahrer(29.07.2008): *2.109.699* Kanten
  (im Vergleich Datensatz vom 19.03.2008 - *1.012.460* Kanten)

Für die folgenden Performancevergleiche, der verwendeten und überarbeiteten Routing-Algorithmen, wurde der Deutschland OSM Datensatz vom 19.03.2008 verwendet. Der neuere Datensatz vom 29.07.2008 konnte aufgrund seiner Größe und dem lokalen System, bestehend aus Quad CPU (@ 2.40GHz), 3.25GB RAM und Windows XP SP2, nicht für den Vergleich verwendet werden. Die in der Tabelle stehenden Zeiten für das Routing spiegeln die Zeit wieder, die nur für die eigentliche Routenberechnung benötigt wird. Also vom Aufruf des Algorithmus bis zur Rückgabe des berechneten Pfades. Die Zeiten für die Erstellung von Fahranweisungen, der Geometrie der Route oder von Karten sind darin nicht enthalten.

Tabelle 6.1: Performancevergleich Routing-Algorithmen (Datensatz 19.03.2008)

Vergleich	I	II	III	IV
Streckenlänge	1.4 km	15.8 km	121.8 km	429.1 km
Anzahl der Kanten im Ergebnis	12	89	180	292
Original Dijkstra-Algorithmus	6.85 s	6.86 s	6.82 s	6.87 s
Überarbeiteter Dijkstra-Algorithmus	0.85 s	0.87 s	1.58 s	4.65 s
A-Stern-Algorithmus	0.09 s	0.12 s	0.45	1.98 s

Wie in der Tabelle 6.1 zu sehen ist, wurden vier Vergleiche mit den drei vorhandenen Routing-Algorithmen und Distanzen von ca. 1, 16, 122 und 429 km durchgeführt. Die angegebenen Zeiten für die Berechnungen sind jeweils eine Mittlung aus fünf Routenberechnungen. Zudem wurden alle Messungen mit dem Autofahrer Routing-Graph und der Option „Fastest“ durchgeführt. Im folgenden Liniendiagramm (Abbildung 6.4) sind ebenfalls die Unterschiede zwischen den verschiedenen und vorhandenen Routing-Algorithmen zu sehen.

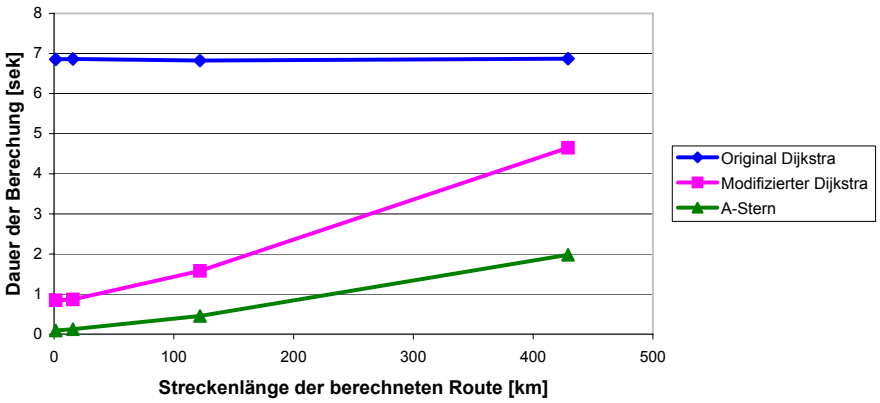


Abbildung 6.4: Performancevergleich zwischen den Routing-Algorithmen

Dabei ist gut zu erkennen, dass der „Original“ Dijkstra, egal bei welchem Abstand zwischen Start und Ziel, bei einer Routenberechnung stets den ganzen Graphen durchgerechnet hat. Der Zeitunterschied zwischen dem überarbeiteten Dijkstra und dem verwendeten A-Stern beim Vergleich I (Streckenlänge 1.4 km) deutet eigentlich immer noch darauf hin, dass etwas im Dijkstra Algorithmus nicht optimal läuft. Da ein A-Stern-Algorithmus aber für eine Routenplanung zwischen zwei Punkten die bessere Wahl ist, wurde der vorhandene Dijkstra-Algorithmus nicht weiter untersucht. Die Tabelle 6.2 zeigt die entstehenden Zeitunterschiede, die durch die OGC OLS RS Schnittstelle (XML-lesen und schreiben) und der Erstellung der Fahranweisungen, der Geometrie der Route und einer Karte entsteht. Da diese Unterschiede vom verwendeten Algorithmus unabhängig sind, wurden die Messungen mit dem A-Stern-Algorithmus durchgeführt.

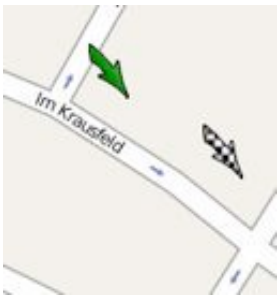
Tabelle 6.2: Antwortzeiten des OLS Route Service (Datensatz 19.03.2008)

Messung	I	II	III	IV
Streckenlänge	1.4 km	15.8 km	121.8 km	429.1 km
Nur Routing mit A-Stern-Algorithmus	0.09 s	0.12 s	0.45	1.98 s
Antwortzeiten des OLS RS (mit A-Stern-Algorithmus)	0.09 s	0.13 s	0.48	2.09 s
Zeitunterschied	0 s	+ 0.01 s	+ 0.03 s	+ 0.11 s

Die Messungen wurden ebenfalls wieder unter Verwendung des OSM Deutschland Datensatzes 19.03.2008 und dem lokalen System (Quad CPU @ 2.40GHz, 3.25GB RAM und Windows XP SP2) ausgeführt. Bei der Anfrage zur Routenplanung wurde eine Fahranweisung, die Geometrie der Route und eine „Karte“ angefordert. Die Antwortzeiten in der vierten Zeile sind jeweils wieder eine Mittlung aus fünf Anfragen an den RS. Start- und Zielpunkt wurden in WGS-84 Koordinaten übergeben und die Geometrie der Route wurde ebenfalls in diesem SRS angefordert. Als Karte wurde keine Rastergrafik sondern eine *Keyhole Markup Language* (KML) Datei verlangt. Eine Rasterkarte mit der berechneten Route kann je nach Größe die Antwortzeit des RS erhöhen. Weiterhin wurden keine Zwischenpunkte oder eine AvoidList in der Anfrage verwendet.

Routenberechnungen erfolgen unter Zuhilfenahme der Graphentheorie. Dabei wird über Knoten und deren Beziehungen über Kanten und einem Gewicht die kürzeste oder schnellste Verbindung zwischen zwei Knoten ermittelt. Demnach muss in der Regel zum Start- und Zielpunkt der nächstmögliche Knoten gefunden werden, um eine Route berechnen zu können. Bisher wurde die angegebene Position des Start- und Zielpunkts bei einer Routenberechnung automatisch auf den nächstmöglichen Knoten verschoben (siehe Abbildung 6.5 (b)), um von dort die Berechnungen starten und enden zu lassen. Das bringt natürlich den Nachteil der Änderung der vorgegebenen Positionen mit sich, was sich gera-

de bei Fußgängerrouting negativ bemerkbar macht. Die Abbildung 6.5 (a) und (b) zeigen die einzelnen Situationen.



(a) Start- und Zielpunkt an einer Straße (Ausgangssituation)

(b) Start- und Zielpunkt werden auf nächsten Knoten verschoben (Alt)

(c) Start- und Zielpunkt werden wie angegeben verwendet (Neu)

Abbildung 6.5: „Problem“ bei der Start- & Zielangabe bei einer Routenberechnung

Im Bild (c) ist das Ergebnis vom überarbeiteten RS zu sehen. Dabei werden die vorgegebenen Positionen nicht verändert und jeweils ein Wegstück wird von der Start-/Zielposition der nächstmöglichen Straße hinzufügt. Für die Routenplanung von Fußgängern war dies eine wichtige Änderung. Weitere Probleme entstanden wenn die Start- und Zielposition an einer Einbahnstraße liegen, die Positionen sich innerhalb einer kurzen Entfernung an einer Straße oder Einbahnstraße befinden oder wenn die Positionen durch eine Kreuzung voneinander getrennt sind. Diese „besonderen Situationen“ wurden bisher bei der Wegberechnung, vor allem für Autofahrer, nicht beachtet. Da sich die Positionen von Start und Ziel an einem Knoten oder innerhalb einer Kante befinden, kann der eigentlich Routing-Algorithmus dort nicht verwendet werden. Die Situationen müssen vor der Berechnung durch den Routing-Algorithmus überprüft, gegebenfalls abgefangen und berechnet werden. Was passiert, wenn dies nicht erfolgt, ist in Abbildung 6.6 zu sehen.

(a) Die Route verläuft fälschlicherweise entgegen der Einbahnstraße

(b) Start- und Zielpunkt sind über eine Kreuzung voneinander getrennt

(c) Wie (b), aber zweimal entgegengesetzt der Einbahnstraßen

Abbildung 6.6: Besondere Routing-Situationen (Alt)

Unter Umständen kann es bedeuten, dass eine Fahrt entgegen der Einbahnstraße beginnt

(Bild (a) & (b)) und/oder auch so endet (c). In den folgenden Abbildungen 6.7 sind die Ergebnisse der überarbeiteten Version des RS bezüglich der Handhabung solcher Situationen dargestellt. Weiterhin sind noch andere Kombinationen und Varianten möglich, die durch die implementierte Überprüfung abgefangen und berechnet werden.

(a) Die Route verläuft entsprechend der Einbahnstraße

(b) Start- und Zielpunkt sind über eine Kreuzung voneinander getrennt

(c) Wie (b), aber ebenfalls entsprechend der Einbahnstraßen

Abbildung 6.7: Besondere Routing-Situationen (Neu)

Beim RS kann optional eine sogenannte AvoidList bei einer Routenberechnung angegeben werden. Die Liste kann unter anderem Polygone enthalten, die bei der Routenberechnung vermieden werden sollen. Diese Option wurde bisher nicht konsequent im RS durchgeführt und im Umfeld des Start- und Zielpunktes wurden die angegebenen Vermeidungsgebiete unter Umständen nicht beachtet. So konnte es also bisher passieren, dass eine Route am Anfang oder am Ende durch eine solches Gebiet verläuft. In Abbildung 6.8 ist das alte Problem und das neue Ergebnis zu sehen (mehr zum Thema *AvoidList* im Kapitel *6.2.2 Emergency Route Service*).

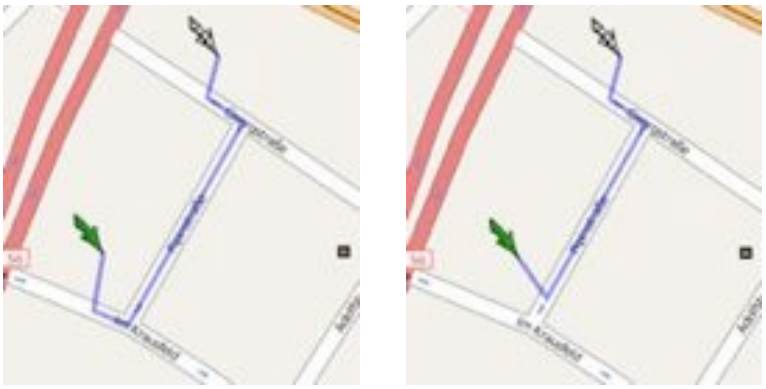
(a) Die Route verläuft durch das Vermeidungsgebiet (Alt)

(b) Die Route verläuft außen herum (Neu)

Abbildung 6.8: Besondere Routing-Situationen - *AvoidList*

Bei einer Anfrage an einem OLS RS können nicht nur Punktkoordinaten für Start und Ziel angegeben werden, sondern auch Adressen. Damit bei einer Adressangabe auch wirklich an der entsprechenden Straße gestartet oder die Route beendet wird, mussten auch hier wieder verschiedene Optimierungen erfolgen. In der folgenden Abbildung 6.9 (a) beginnt die Route auf der Straße („Im Krausfeld“), die zu der Koordinate am nächsten liegt. Die

Adresse bezog sich auf die „Pipinstraße“. Im Bild (b) ist das Ergebnis des geänderten Route Service zu sehen.



(a) Die Route beginnt auf der falschen Straße (Alt)

(b) Die Route beginnt auf der angegebenen Straße (Neu)

Abbildung 6.9: Start (oder Ziel) durch Adresse angegeben

Zugleich wurden auch die Fahranweisungen des RS überarbeitet. Neben kleinen Änderungen, wie verbesserte Berechnungen der Zeiten sowie der Strecken für jede einzelne Fahranweisung, wurden zusätzlich die Textbausteine für die verschiedenen Sprachen überarbeitet. Beispielhafte Anfragen an und Antworten vom RS und die dabei unterstützten Parameter werden im Anhang *A.1 Request- & Response Beispiel für OLS Route Service* gezeigt.

6.1.3 Location Utility Service

Von dem OLS Location Utility Service war bereits vor dieser Arbeit eine Implementierung vorhanden. Sie unterstützte allerdings nicht die komplette OLS Spezifikation v1.1. Gerade die wichtige Option, eine sogenannte *FreeForm*-Adresse geocodieren zu können, war nicht vorhanden. Insgesamt besteht der Location Service grob aus drei Teilen: der XML-Schnittstelle, dem Teil für das Geocodieren und dem Teil, der für das Reverse-Geocodieren zuständig ist. Die XML-Schnittstelle wurde ohne größere Änderungen so belassen. Auch der Teil für das Reverse-Geocodieren, also dem Teil, der aus einer Position eine Adresse macht, wurde nur an manchen Stellen überarbeitet und optimiert. Er besteht im Wesentlichen nur aus einer räumlichen Anfrage an die Datenbank und aus der Rückgabe der Adresse in einer strukturierten Form. Der Teil für das Geocodieren wurde hingegen fast komplett neu implementiert. Wenn eine Adresse geocodiert werden soll, bietet der Service zwei Möglichkeiten an: entweder die Angabe der Adresse in einer strukturierten Form oder, wie bereits erwähnt, in einer sogenannten FreeForm-Variante.

Das folgende Listing 6.1 ist ein Beispiel für eine strukturierte Form einer Adresse durch das `StreetAddress`-Element in einer Anfrage an den OLS Location Service:

```
<xls:Address countryCode="DE">
  <xls:StreetAddress>
    <xls:Building number="36"></xls:Building>
    <xls:Street officialName="Holzstraße"></xls:Street>
  </xls:StreetAddress>
  <xls:Place type="CountrySecondarySubdivision">Mainz</xls:Place>
  <xls:PostalCode>55116</xls:PostalCode>
</xls:Address>
```

Listing 6.1: XML-Beispiel für eine strukturierte Adressangabe

Die Unstrukturierte Form einer Adresse durch das `freeFormAddress`-Element wird in folgendem Listing 6.2 dargestellt:

```
<xls:Address countryCode="DE-RP">
  <xls:freeFormAddress>55116 Mainz Holzstraße 36</xls:freeFormAddress>
</xls:Address>
```

Listing 6.2: XML-Beispiel für das freeFormAddress-Element

Eine Geocodierung gemäß einer strukturierten Adress-Anfrage ist prinzipiell nicht so aufwendig zu implementieren, da bereits die Vorgaben, was zum Beispiel die Postleitzahl oder der Straßenname ist, angegeben sind (verschiedene Schreibweisen und Rechtschreibfehler außer acht gelassen). Etwas umfangreicher wird es dagegen, eine Adresse zu geocodieren, die in einer beliebigen Form und in einem beliebigen Umfang angegeben wird. Hierfür wurde ein Algorithmus entwickelt und im Location Service implementiert, der eine fast „beliebige“ Form einer Adresse geocodieren kann. Dabei werden zuerst, wenn angegeben, die Abkürzungen aus der Adresse durch die ausgeschriebenen Wörter ausgetauscht. Zum Beispiel: „str.“ wird mit „Straße“ ersetzt. Wie die OSM Daten für den Location Service aufbereitet wurden, ist im Kapitel 4.4 *OLS Location Utility Service* beschrieben. Weiterhin ist die realisierte Suche einer Adresse „Case Insensitivity“ und Sonderzeichen, wie Kommas oder Anführungszeichen, werden entfernt. Der Adress-Such-Algorithmus zerlegt die zu geocodierende Adresse in verschiedene Einzelteile. Dabei untersucht er, wo sich eventuell eine Postleitzahl befinden könnte. Danach wird versucht durch kombinieren der Einzelteile und Anfragen an die Datenbank, die angefragte Adresse in der Datenbank zu finden. Ein Beispiel für den groben Ablauf der Bearbeitung einer Anfrage:

1. Anfrage an den Location Utility Service
   folgende Adresse soll geocodiert werden: „53115 Bonn, beringstr.“

2. Entfernen des Kommas und ersetzen der Abkürzung „str.“ durch „straße“
   ⇒ „53115 Bonn beringstraße“
3. Zerlegung der Adresse in folgende Teile: „53115“, „Bonn“ und „beringstraße“
4. Erstellung und Kombination der Datenbankabfrage
5. Aufbereitung des gefundenen Datenbankeintrages (Adressbucheintrags) zu: *PostalCode*=53115, *Municipality*=Bonn, *Street officialName*=Beringstraße und *CountrySubdivision*= Nordrhein-Westfalen und Rückgabe der Adresse

Somit ist mit Hilfe des Location Utility Service für Deutschland die Suche nach einem Bundesland, einer Postleitzahl, Ortsnamen und Straßennamen möglich. Ergänzend ist eine Kombination von Postleitzahl, Ortsname und Straßennamen in der strukturierten und freien Adressform möglich. Bei der Geocodierung einer strukturierten Adresse, wurde der vorhanden Levenshtein-Distanz-Algorihtmus wieder integriert (Levenshtein, 1965 (Russisch)). Dieser berechnet ein Maß (Distanz) für den Unterschied zwischen zwei Zeichenketten[13] und kann aus diesem Grund für eine Rechschreibkorrektur bei einer fehlerhaften oder unvollständigen Adresssuche genutzt werden. Hausnummern werden wegen der nicht ausreichenden Nutzung im OpenStreetMap Projekt noch nicht verwendet. Damit die Suche nach Adressen schnell erfolgen kann, sind alle Spalten der verwendeten Tabelle (des Adressbuches) indiziert. Des Weiteren wurden für die Suche in der Datenbank ein Postgres-Verbindungs-Manager in Java entwickelt, der eine vorgebbare Anzahl von offenen Verbindungen zu einer Datenbank verwaltet und bei Bedarf eine freie Verbindung liefert. Das bringt den Vorteil, dass immer eine offene Verbindung zu einer DB vorhandenen ist und nicht erst noch geöffnet werden muss. Beispielhafte Anfragen an und Antworten vom OLS Geocoder und Rerverse Geocoder und die dabei unterstützten Parameter sind wieder im Anhang *A.2 Request- & Response Beispiel für OLS Location Utility Service* zu finden.

6.1.4 Directory Service

Der mit dieser Arbeit entstandene OLS Directory Service ist eine Neuimplementierung. Von vier verschiedenen Arten bei der Anfrage wird aber vorerst nur die dritte unterstützt.

1. Suche eines POI über die vorgegebene Adresse (`Address`)
2. Suche der nächstmöglichen POIs (`Nearest`)
3. Suche nach POIs, die sich inner- oder außerhalb einer vorgegebenen Entfernung befinden (`WithinDistance`)

[13] Levenshtein-Distanz - http://de.wikipedia.org/wiki/Levenshtein-Distanz

4. Suche nach POIs, die sich innerhalb eines vorgegebenen Bereiches befinden (`WithinBoundary`)

Die genau verwendeten POI-Typen und welche Kategorien beim Verzeichnisdienst, beziehungsweise bei der Umkreissuche, genutzt werden können, ist im Kapitel *4.5 OLS Directory Service* bei der Datenaufbereitung beschrieben. Bei der Entwicklung dieses Services wurde ebenfalls wieder der im Zusammenhang beim Location Utility Service entstandene Verbindungs-Manager für Postgres Datenbanken verwendet. Wie eine Anfrage und die Antwort des Services aussieht, ist im Anhang A.3 beschrieben. Die Abbildung 6.10 zeigt das Ergebnis einer Umkreissuche in Bonn, in dem im Abstand von 500m zur Position alle Apotheken angezeigt werden.

Abbildung 6.10: Ergebnis einer Umkreissuche in Bonn

6.1.5 Presentation Service

Eine zumindest teilweise vorhandene Implementierung des OLS Presentation Service war bereits vor dieser Arbeit vorhanden. So bestand lediglich die Aufgabe darin, den bestehenden Dienst in das entstandene OpenLS Framework (siehe Kapitel 6.1.1) zu integrieren. Bei der vorhandenen Umsetzung handelt es sich nicht um einen vollwertigen Kartendienst, sondern eher um eine Art „Proxy“ zu einem SLD-fähigen WMS. Bisher wurde hierbei ausschließlich ein GeoServer (siehe Kapitel 5.2) verwendet. Der OLS Presentation Service nimmt Anfragen entgegen, konvertiert diese in einen GetMap-Request und sendet sie an den WMS weiter. Bislang hat diese Art und Weise der Funktionalität ausgereicht, allerdings wurde der Presentation Service bis jetzt noch nicht für Mobile Geräte genutzt. Auch im Kontext dieser Arbeit kommt er nicht richtig zum Einsatz, weshalb keine weiteren Ergänzungen oder Optimierungen an ihm erfolgten.

6.2 Weitere Dienste

Wie in der Abbildung *6.1 Gesamtarchitektur der verwendeten, weiterentwickelten und implementierten Dienste* zu sehen, sind neben den OLS Core Services weitere nicht OLS-konforme Dienste im OpenLS Framework enthalten. In diesem Unterkapitel werden die einzeln vorgenommen Änderungen an den Diensten oder Umsetzungen beschrieben.

6.2.1 Accessibility Analysis Service

Der vorhandene *Accessibility Analysis Service* (AAS) von Neis et al. (2007) war eine Software-Entwicklung aus dem Forschungsprojekt ImmoSDSS_RLP[14]. Für die Aufnahme in das OpenLS Framework wurde er an verschiedenen Stellen überarbeitet. Genauso wie beim RS wurde das Logging auf log4j und die Datenhaltung und -verarbeitung auf WGS-84 umgestellt und der verwendete Dijkstra-Algorithmus korrigiert. Weil der Graph für die Ermittlung der Erreichbarkeit identisch zum Routing-Graph des RS ist, wurde der Graph aus dem AAS entnommen. Der Service nutzt jetzt den gleichen Graph wie der RS. Die folgende Abbildung 6.11 zeigt das Ergebnis einer Erreichbarkeitsanalyse mit Nutzung der aufbereiteten OSM Daten. Die hellgrüne Fläche ist dabei das Gebiet, das innerhalb von 15 Minuten von der Bonner Innenstadt erreicht werden kann.

Abbildung 6.11: Ergebnis einer Erreichbarkeitsanalyse für Bonn

Weitere Informationen über die genaue Funktionsweise für die Erstellung des Erreichbarkeitspolygon oder der Parameter für Anfragen und Antworten sind unter Neis et al. (2007) zu finden. Ein Beispiel für eine Anfrage an den Service und die dazugehörige Antwort ist

[14] ImmoSDSS_RLP - http://www.i3mainz.fh-mainz.de/Article300.html

im Anhang A.4 aufgeführt. Die Schnittstelle des AAS entstand mit dem damaligen Projekt und ist nicht OGC konform. Als Folgearbeit wäre eine Umstellung des Dienstes auf die *Web Processing Service* (WPS) Schnittstelle des OGCs möglich.

6.2.2 Emergency Route Service

Wie bereits Neis (2006) zeigte, besitzt die OpenLS Spezifikation die Besonderheit, dass bei einer Routenplanung eine sogenannte *AvoidList* angegeben werden kann. Die Implementierung des *Emergency Route Service* (ERS) war bereits vor dieser Arbeit vorhanden und wurde ebenfalls unter kleinen Änderungen in das OpenLS Framework aufgenommen. Von der Funktionsweise ist es kein vollwertiger Route Service, sondern eher eine Art „Proxy“ zu einem OpenLS konformen Routensuchdienst. Die Besonderheit ist dabei, dass der ERS bei einer Anfrage Vermeidungsgebiete aus einem WFS (via *GetFeature*-Anfrage mit *BoundingBox*) abfragt, diese in die Anfrage integriert und sie weiter an einen OLS RS sendet. Der OLS RS berechnet dann unter Vermeidung der angegebenen Polygone eine Route. Die folgende Abbildung 6.12 zeigt diese Sequenz:

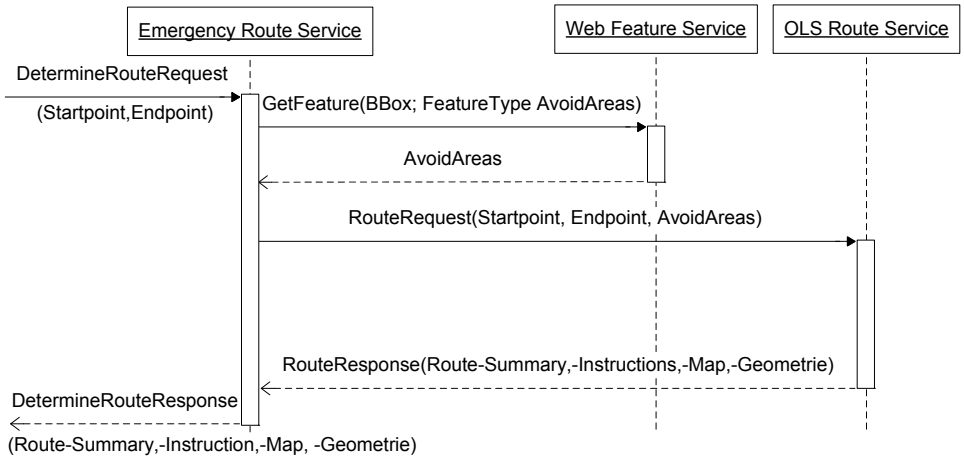


Abbildung 6.12: Sequenzdiagramm des Emergency Route Service

Haase, Zipf, Neis und Camargo (2008) demonstrierten, wie die Funktionalität des ERS zur Evakuierung der Bevölkerung in Überschwemmungsgebieten genutzt werden könnte. Wie der OLS RS und der ERS für 3D erweitert werden könnte, zeigten Neis, Schilling und Zipf (2007). Mayer (2008) präsentierte in seiner Diplomarbeit ein Konzept und eine Implementierung zur Bereitstellung von TMC-Daten für Geodateninfrastrukturen auf standardisiertem Weg. TMC-Meldungen werden als Sensorbeobachtungen betrachtet und mit Hilfe der OGC Sensor Web Technologie bereitgestellt. Somit sind über einen OGC *Sensor Observation Service* (SOS) immer aktuelle Verkehrsmeldungen abrufbar. Über

die OGC WPS Schnittstelle werden die TMC-Rohdaten aufbereitet und daraus Geometrien erzeugt, die die Orte der Verkehrsereignisse repräsentieren. Um diese Geometrien über das Web zugänglich zu machen, erfolgt eine Speicherung dieser in einem WFS. Die zuvor erwähnten Prozessschritte werden in einem regelmäßigem Zeitabstand ausgeführt, sodass immer die aktuellen TMC-Geometrien verfügbar sind. Durch die Speicherung der TMC Daten in einem WFS, kann der ERS auf die Meldungen zugreifen. Als Folge dessen können bei einer Routenberechnung aktuelle Verkehrsmeldungen einfließen. Weiterhin können die TMC-Daten durch die Bereitstellung in einem WMS auch zur Visualisierung genutzt werden. Die Abbildung 6.13 zeigt eine Routenberechnung unter Verwendung der TMC Daten über den ERS. Dabei ist gut zu sehen, dass die Baustelle mit dem Stau, die normalerweise auf dem Weg liegen würde, umfahren wird. Nachfolgend sind noch weitere visualisierte TMC Meldungen darstellt (Straßenschilder und rot-transparente Linien).

Abbildung 6.13: Routenberechnung mit Verwendung des ERS & TMC

6.2.3 Extended Route Service

Das sogenannte „Straßengegenstände“ (wie Ampeln, Fußgängerüberquerungen oder Kreisel) ein wichtige Rolle für Autofahrer spielen, beschrieben May und Ross (2006). Wie Landmarken im Zusammenhang mit OpenLS Diensten genutzt werden könnten, zeigten bereits Neis und Zipf (2007) und Neis und Zipf (2008a). Weil im OSM Projekt die eben genannten Straßengegenstände vorhanden sind, wurde die prototypische Version von

Neis und Zipf (2008a) geändert beziehungsweise erweitert. Bei der vorhandenen Version wurden bisher nur Landmarken im Umkreis von Kreuzungen in den Fahranweisungen erwähnt. Nach einer Änderung werden jetzt vorhandene Ampeln oder Kreisel an einer Kreuzung in den Fahranweisungen erwähnt. Die Abbildung 6.14 zeigt eine berechnete Route, an der eine Ampel und ein Kreisel an einer Kreuzung vorhanden sind.

Abbildung 6.14: Ergebnis einer Routenplanung (Extended Route Service)

Bei der Erstellung der Fahranweisungen wird an jeder Kreuzung der berechneten Route überprüft, ob eine Ampel oder ein Kreisel vorliegen. Sollte eines von beiden vorhanden sein, wird es in die Anweisung mit aufgenommen. Das folgende Beispiel enthält die Fahranweisungen der berechnete Route aus Abbildung 6.14 vom normalen Route Service:

- *Nr. 3 - Bitte fahren Sie geradeaus auf Heerstrasse für 0.2 km ...*
- *Nr. 4 - Bitte fahren Sie rechts auf Georgstrasse für 0.2 km ...*
- *Nr. 5 - Bitte fahren Sie links auf Adolfstrasse für 0.1 km ...*

Die folgenden Anweisungen sind vom Extended Route Service. Dabei sind in der Anweisung Nr. 3 und 5 die erwähnten „Straßengegenstände“ inkludiert.

- *Nr. 3 - Bitte fahren Sie geradeaus **an der Ampel** auf Heerstrasse für 0.2 KM ...*
- *Nr. 4 - Bitte fahren Sie rechts auf Georgstrasse für 0.2 KM ...*
- *Nr. 5 - Bitte fahren Sie links **im Kreisel** auf Adolfstrasse für 0.1 KM*

Mit der Implementierung dieses Services wurden weitere Vorbereitungen getroffen, um in einer zukünftigen Version des Extended Route Service auch Landmarken wieder in die Fahranweisungen aufnehmen zu können. Das war im Rahmen dieser Arbeit nicht mehr möglich.

6.2.4 RESTful OLS

Um eine vereinfachte Nutzung des OLS RS und OLS Geocoders und Reverse Geocoders zu bieten, wurde zusätzlich ein weiterer prototypischer Web Service im OpenLS-Framwork entwickelt. Dieser bietet die Funktionalität, die zuvor genannten Services über den *REpresentational State Transfer* (Akronym REST) Architekturstil (Fielding, 2000) anzusprechen. Dabei werden die einzelnen Bestandteile einer URL dazu verwendet, um einen Dienst anzusprechen und um Informationen für die Anfrage zu übertragen. Im folgenden Beispiel wird der RESTful OLS Service für eine Geocodierung eines Ortsnamen über die URL angesprochen: *http://www.MyOpenLS.com/openls/restful/geocode/Bonn*. Entscheidend sind die Parameter */geocode* und */Bonn*. Ersteres für die Signalisierung der Geocodierung und „Bonn“ als die zu Geocodierende Adresse. Somit sind folgende Beispiele mit dem RESTful OLS Service für die Geocodierung, Reverse Geocodierung oder eine Routenplanung möglich:

- *http://www.MyOpenLS.com/openls/restful/geocode/53111*
- *http://www.MyOpenLS.com/openls/restful/geocode/reverse/7.099 50.7793*
- *http://www.MyOpenLS.com/openls/restful/route/7.09 50.74/7.19 50.73/Fastest*

6.3 OpenRouteService.org

Mit Hilfe von OpenLayers (siehe Kapitel 5.3) wurde eine Webseite erstellt, über die so gut wie alle Services von dieser Arbeit genutzt werden können. Eine erste Version wurde bereits anderthalb Monate nach Beginn unter der Adresse: *http://OpenRouteService.org* (ORS) veröffentlicht. In der Zeit bis zum Ende dieser Arbeit wurde sie immer für die dazukommenden Services weiterentwickelt. Anfängliche Probleme gab es durch nicht vorhandene *JavaScript* (JS)-Kenntnisse und nicht so ausführliche Dokumentationen oder Beispiele von OpenLayers. Gerade für die Integration der entwickelten Services in die Webseite, mussten die kompletten Anbindungen erstellt werden. Somit entstanden JS-Funktionen, um OLS Dienste anzusprechen, die Antworten auszulesen und die Ergebnisse mit Hilfe von OpenLayers zu präsentieren. Weiterhin wurden für die Kommunikation zwischen den JS-Funktionen und den Services noch PHP-Seiten und -Funktionen entwickelt. Die Kommunikation zwischen ORS und dem OpenLS Service ist im oberen Teil der Abbildung 6.15 enthalten. Im unteren Teil der Abbildung 6.15 sind die verschiedenen Kartendienste zu sehen: angefangen vom OSM Projekt, über den ORS WMS TileCache und den WMS mit den TMC Daten, die unter Zuhilfenahme der OpenLayers Bibliothek in der Webseite integriert sind. Einige Probleme entstanden, als die Webseite für verschie-

dene Webbrowser kompatibel gemacht wurden. Folgende Browser werden unterstützt: Firefox (2.0.0.16 & 3.0.1), Internet Explorer (7.0.5730.13) und Opera (9.51).

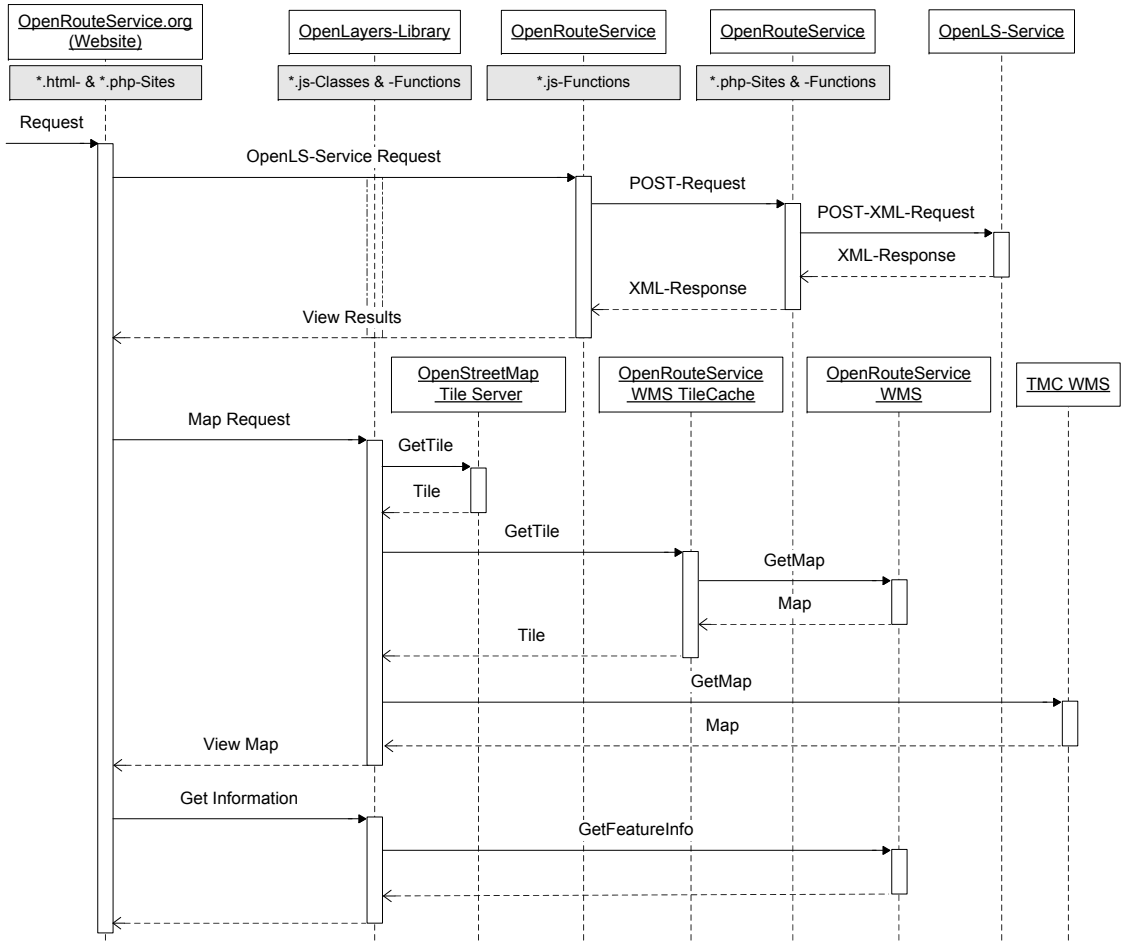


Abbildung 6.15: Sequenzdiagramm von OpenRouteService.org

Die Webseite unterteilt sich in drei Bereiche (siehe Abbildung 6.16):

- I. Bereich (Rot-gestrichelt) - Der Abschnitt, in dem die Karte und die Ergebnisse der Dienste angezeigt werden. Rechts oben über das '+' öffnet sich ein Auswahlfenster für die anderen möglichen Karten.
- II. Bereich (Gelb-gestrichelt) - In diesem Abschnitt können die Dienste verwendet oder verschiedene Parameter für die Nutzung eingestellt werden.
- III. Bereich (Grün-gestrichelt) - Die textuellen Ergebnisse der Dienste werden in diesem Abschnitt ausgegeben.

Abbildung 6.16: Webseite OpenRouteService.org

Welche Dienste werden auf OpenRouteService.org angeboten?

Für Deutschland stehen mit den aufbereiteten OSM Daten folgende zur Verfügung:

- Routenplaner für
  - Autofahrer (*Schnellste & Kürzeste*)
  - Fußgänger
  - Fahrradfahrer
- Geocoder & Reverse Geocoder
- Umkreissuche (POI-Suche)
- Erreichbarkeitsanalyse
- Weitere Routenplaner: zum Beispiel mit erweiterten Fahranweisungen oder der Möglichkeit, Vermeidungsgebiete zu digitalisieren.

Für die Berechnung einer Route mit dem Routenplaner, kann die Start- und Zielposition über ein Klicken auf die Karte oder über die Eingabe einer Adresse in ein Suchfeld gesetzt werden. Für die Auswahl der Sprache der Fahranweisungen, die Längeneinheit und die Art der Routenplanung (Auto, Fußgänger oder Fahrrad) stehen jeweils Dropdown-Menüs zu Verfügung. Weiterhin kann optional die Verwendung der aktuellen Verkehrsmeldungen oder die Vermeidung von Autobahnen aktiviert werden. Die berechnete Route wird

anschließend in der Karte eingezeichnet. Streckeninformationen, Fahranweisungen und der Download der Route als GPX-Datei werden im entsprechenden Bereich angezeigt. Eine Besonderheit bietet die Option „Extended Routing Version“: Damit können beliebige Gebiete in der Karte digitalisiert werden, durch welche die zu berechnende Route nicht verlaufen soll. Im Suchfeld des Geocoders können verschiedene Arten einer Adresse eingegeben werden. Nach der Geocodierung der Adresse durch den Dienst, verschiebt sich die Karte automatisch zu der Position der Adresse. Soll zu einer Position in der Karte die Adresse gesucht werden, kann diese über die Funktion „Where I am?“ bei dem Reverse Geocoder angefragt werden. Für die Umkreissuche muss eine beliebige Position in die Karte gesetzt werden. Neben der Entfernung für die Suche um die Position, muss entweder die Kategorie oder der POI-Typ, nach dem gesucht werden soll über das Dropdown-Menü ausgewählt werden. Über „Accessibility Analysis“ kann die Erreichbarkeitsanalyse auf der Webseite verwendet werden. Für die Nutzung muss eine Position festgelegt und eine Zeit für die Erreichbarkeit eingetragen werden.

Resonanz auf die Webseite

Die erste Version der ORS Website ging Anfang April während dieser Arbeit online. Durch die Veröffentlichung der URL auf verschiedenen OSM Maillinglisten (*talk*, *talk-de* & *routing*)[15], in Publikationen durch Neis und Zipf (2008b), Vorträgen (Neis, Zipf & Schmitz, 2008) oder anderen Zeitungs- und Medienberichten (zum Beispiel Dworschak (2008)), entstanden die in Tabelle 6.3 stehenden Besucherzahlen und Seitenzugriffe. Die Herkunft der Besucher war dabei über die gesamte Welt verteilt. Die Spalten RS (*Route Service*), LUS (*Location Utility Service*), AAS (*Accessibility Analysis Service*) und DS (*Directory Service*) enthalten die Zahlen, wie oft die Services genutzt wurden.

Tabelle 6.3: Statistiken der Webseiten-Nutzung von OpenRouteService.org

Monat/2008	Besucher	Seitenaufrufe	RS	LUS	AAS	DS
April	ca. 800	ca. 1.400	ca. 1.500	-	-	-
Mai	ca. 4.400	ca. 6.600	ca. 6.000	-	-	-
Juni	ca. 4.200	ca. 6.800	ca. 5.700	ca. 10.300	ca. 200	-
Juli	ca. 5.000	ca. 9.000	ca. 12.000	ca. 20.000	ca. 400	ca. 2.200

[15] lists.openstreetmap.org Mailing Lists - http://lists.openstreetmap.org/listinfo

7 Nutzung & Installation

In diesem Kapitel wird die Nutzung der entwickelten Programme für die unterschiedlichen Aufbereitungen der OSM Daten beschrieben. Danach wird die Einrichtung des entstandenen OpenLS Service auf einem Server erläutert.

7.1 Datenaufbereitungs-Tools

Um die OSM Daten für die verschiedenen Dienste von dieser Arbeit aufzubereiten (siehe Kapitel *4 Aufbereitung der OSM Daten für ...*), kann die entwickelte Konsolenanwendung (*OSM2Shape.jar*) genutzt werden. Das Tool bietet durch unterschiedliche Nutzung entweder die Möglichkeit, die Daten in ein Shapefile abzuspeichern oder sie direkt in eine PostGIS-DB zu schreiben. Vor der Verwendung müssen eventuell drei Konfigurationsdateien angepasst werden, die sich immer im gleichen Verzeichnis wie die Anwendung befinden müssen.

1. *osm.properties.xml* - Sollen die OSM Daten in eine Datenbank geschrieben werden, müssen in dieser Datei die notwendigen Zugangsparameter eingetragen werden. Benötigt wird eine PostgreSQL[1] ab Version 8.1 und PostGIS[2] ab 1.3.2 Installation.
2. *config.log.xml* - Über die Konfigurationsdatei kann der Logger der Anwendung eingestellt werden. Neben den verschiedenen Log-Levels kann auch das Format der Log-Einträge geändert werden.
3. *addressbook.properties.xml* - Diese Datei enthält die Zugangsparameter für die DB, die für die Datenaufbereitung für den OLS Location Utility Service wichtig sind.

Das folgende Beispiel 7.1 zeigt den Aufruf der Anwendung über die Konsole. Dabei spielt es keine Rolle, ob es sich um eine Windows- oder Linux-Plattform handelt. Einzige Voraussetzung: Java 6.0[3] muss eingerichtet und installiert sein. Angefangen bei *java*, über die Zuweisung des maximal verwendeten Speichers (*-Xmx1536m*) und der Bekanntgabe

[1] PostgreSQL - http://www.postgresql.org/
[2] PostGIS - http://postgis.refractions.net/
[3] Java - http://www.java.com/

des Klassenpfades (*-cp OSM2Shape.jar*), erfolgt der eigentliche Programmaufruf durch *osm2shape.console.All*. Der letzte Parameter gibt die zu verwendende *.osm Datei an.

```
1        java -Xmx1024m -cp OSM2Shape.jar osm2shape.console.All germany.osm
```

Listing 7.1: Aufruf der Java Konsolenanwendung

Über den oben gezeigten Aufruf werden fast alle möglichen Datensätze aufbereitet. Lediglich die Daten für den OLS Location Utility Service müssen über einen separaten Aufruf (**.AddressBook*) aufbereitet werden. Sollen zum Beispiel nur die Daten von Gebäuden aus der *.osm Datei ausgelesen werden, kann dies durch Änderung des Aufrufs erfolgen: ersetzen von **.All* durch **.Buildings* beim Aufruf *osm2shape.console*. Selbiges gilt wenn nur die anderen Daten konvertiert werden sollen: **.Naturals*, **.Places*, **.Roads*, **.IntersectionFeatures*, **.RoadsForRouting* oder **.Waterways*.

Als Resultat, bei der im Beispiel 7.1 ausgeführten Konsolenanwendung, gibt es für jedes Layer ein Shapefile. Es kann anschließend mit Hilfe der Tools *osm2pgsql* und *psql* von PostgreSQL/PostGIS konvertiert und in eine PostGIS DB importiert werden.

Es ist aber auch möglich, die Daten direkt nach der Aufbereitung in eine PostGIS DB zu schreiben. Dies kann neben der Konfiguration der *osm.properties.xml*-Datei einfach über die Änderung des Konsolenbefehls ausgeführt werden: es muss *osm2shape.console.All* durch *osm2pgsql.console.All* ausgetauscht werden. Die Variante mit direktem Import in eine DB machte aber, aus bis jetzt unerklärbaren Gründen, auf dem Red Hat Server Probleme, weshalb dort nur die Variante über Shapefiles zum Einsatz kommt.

7.2 Installation auf einem Server

Der entwickelte OpenLS Service (siehe Kapitel 6) benötigt für die Installation auf einem Server oder einem Desktop Rechner folgende Komponenten: Java 6.0, Apache Tomcat ab 5.5.x und PostgreSQL ab Version 8.1 und PostGIS ab 1.3.2. Erfolgreiche Installationen wurden bis jetzt auf folgenden Systemen ausgeführt: Microsoft Windows XP SP2, Microsoft Windows Server 2003 Enterprise Edition x64 SP1, Ubuntu 8.04 (Hardy) LTS Server Edition und Red Hat Enterprise Linux 5 Server.

Bei der Installation auf den beiden Linux Servern traten weitere Probleme auf. Unter anderem gab es Schwierigkeiten bei der Verarbeitung von Shapefiles oder der Anbindung der Koordinaten-Transformations-DB. Beides konnte jedoch bis zum Ende dieser Arbeit beseitigt werden.

Beim OpenLS Service handelt es sich, wie bereits erwähnt, um ein Java Servlet für die In-

stallation in einem Tomcat Servlet-Container. Wie im Kontext eines Tomcat üblich, findet die komplette Konfiguration über XML-Dateien statt. Nach der Integration der OpenLS Webapplication (*.war) in das */webapps* Verzeichnis der Tomcatinstallation, kann der OpenLS Service über folgende XML-Dateien im */WEB-INF* Ordner konfiguriert werden.

- *web.xml* („Deployment Descriptor“) - In dieser Datei werden die Namen, Main-Klassen, URL-Muster und Startreihenfolgen der einzelnen Services festgelegt. Sie ist wichtig, weil in ihr genau festgelegt wird, ob und wann jeder einzelne Dienst im Servlet initialisiert wird.
- *config.log.xml* - Ist die Konfigurationsdatei für den Logger von allen Diensten des OpenLS Service. Neben verschiedenen Log-Levels und dem Format der Log-Einträge, können die Pfade, unter denen die Log-Dateien gespeichert werden sollen, geändert werden.
- *AccessibilityAnalysisService.properties.xml* - Ist die Datei, um den AAS zu konfigurieren. In der XML-Datei müssen neben den verschiedenen Parametern für die Verbindung zur PostGIS, auch der Pfad zu einem OLS Presentation Service für die Kartenerstellung oder das Verzeichnis für die Speicherung der Ergebniskarten, angegeben werden.
- *DirectoryService.properties.xml* - Mit dieser Datei kann der OLS Directory Service konfiguriert werden. Im Wesentlichen sind die Verbindungsparameter für die DB enthalten, sowie die Distanzen für die mini- oder maximale Umkreissuche.
- *EmergencyRouteService.properties.xml* - Ist die Datei, um den ERS zu konfigurieren. Sie enthält Pfade, unter denen der zu verwendende OLS RS und WFS für die Integration der Vermeidungsgebiete zu erreichen sind.
- *LocationUtilityService.properties.xml* - Ist die Konfigurationsdatei für den OLS Location Utility Service. Sie enthält die Verbindungsparamter zur DB für die Adress-Suche.
- *PresentationService.properties.xml* - Ist die Datei, um den OLS Presentation Service zu konfigurieren. Für die Nutzung dieses Services ist eine GeoServer Installation erforderlich.
- *RESTful.properties.xml* - Ist die Konfigurationsdatei für den RESTful OLS Service.
- *RouteService.languages.xml* - In dieser Datei können die Sprachen der Fahranweisungen des OLS RS eingefügt, geändert oder gelöscht werden. Die Struktur dieser Datei gibt eine eigens dafür erstellte XSD vor.
- *RouteService.properties.xml* - Ist die Datei, um den OLS RS zu konfigurieren. Sie

enthält unter anderen Pfade zu anderen OLS Diensten, wie zum Beispiel: OLS Location Utility Service, Directory Service und Presentation Service.

- *RouteServiceLandmarks.properties.xml* - Enthält die Parameter um den Extended Route Service einzurichten.
- *FileDelete.properties.xml* - Ist ein separater Thread, der ein Verzeichnis überwacht, in dem temporär Dateien abgespeichert und nach einer vorgegebenen Zeit vom Thread gelöscht werden. In der Konfigurationsdatei können die Zeiten, wann der Thread aufgerufen wird und wann Dateien gelöscht werden sollen, eingestellt werden.
- *Graph.properties.xml* - Diese Datei ist für die Konfiguration des Routing Graphen des RS und AAS zuständig. In ihr können neben den Verbindungsparameter zu der DB, auch die einzelnen Geschwindigkeiten für verschiedenen Straßentypen, Fußgänger und Fahrradfahrer angegeben werden.

Durch die Installation des OpenLS Service auf einem Server oder Desktop Rechner, können die Dienste über folgende Adressen genutzt werden (hier am Beispiel des Servers dieser Arbeit):

Tabelle 7.1: URLs des OpenLS Service und der weiteren Dienste dieser Arbeit

Service Name	Service URL
Accessibility Analysis Service	http://openls.giub.uni-bonn.de/*openls-osm/analyse*
ERS mit TMC	http://openls.giub.uni-bonn.de/*openls-osm/avoiddetermineroute*
Extended Route Service	http://openls.giub.uni-bonn.de/*openls-osm/determinerouteextent*
OLS Directory Service	http://openls.giub.uni-bonn.de/*openls-osm/directory*
OLS Location Utility Service	http://openls.giub.uni-bonn.d/*openls-osm/geocode*
OLS Presentation Service	http://openls.giub.uni-bonn.d/*openls-osm/getmap*
OLS Route Service	http://openls.giub.uni-bonn.de/*openls-osm/determineroute*
RESTful OLS	http://openls.giub.uni-bonn.de/*openls-osm/restful/**
Web Feature Service	http://openls.giub.uni-bonn.de/*geoserver-osm/wfs*
Web Map Service	http://openls.giub.uni-bonn.de/*geoserver-osm/wms*
WMS TileCache	http://openls.giub.uni-bonn.de/*ors-tilecache/tilecache.py?*
Webseite	http://OpenRouteService.org

Damit die Daten automatisiert auf dem Server aktualisiert werden können, wurde zusätzlich ein Shell-Skript geschrieben, was die verschiedenen Programme aufruft. Im folgenden Listing 7.2, ist eine etwas vereinfachte Variante des Update-Skripts zu sehen. Das Skript beginnt mit dem Download und dem Entpacken der OSM Daten (Zeile 2&3). Nach der Erstellung einer temporären Datenbank für das Update, wird das entwickelte Datenaufbereitungs-Tool ausgeführt. Anschließend werden mit Hilfe der Tools *shp2pgsql*

und *psql* die Shapefiles konvertiert und in die PostGIS DB importiert (Zeile 6-9). Ab der Zeile 10 wird das angesprochene Adressbuch für den OLS Location Utility Service erstellt, anschließend konvertiert und danach in die DB importiert. Als vorletztes werden durch vorhandene SQL-Skripte zum Beispiel noch verschiedene Indizes und Berechtigungen an den Tabellen eingerichtet. Abschließend wird die vorhandene Tabelle mit den alten Daten gelöscht und die Temporäre umbenannt. Somit ist fast ein Update der Daten im laufenden Betrieb möglich. Nach dem Update der Daten müssen lediglich die Services neu initialisiert werden. Auch dieses kann über einen einfachen Skript-Aufruf erfolgen.

```
1  #! /bin/bash
2  echo "*** Start Update Script ***"
3   wget -N -Y off http://download.geofabrik.de/osm/europe/germany.osm.bz2
4   bunzip2 -k -f /srv/data/germany.osm.bz2
5   createdb -E UTF8 -U openlsfoo -T template_postgis osm_auto_tmp -W PASSWORD
6   java -Xmx7000m -cp /srv/data/OSM2Shape.jar osm2shp.console.All germany.osm
7   shp2pgsql -c -I buildings.shp -s 4326 buildings osm_auto_tmp > buildings.sql
8    ....
9   psql -h localhost -p 5432 -U openlsfoo -q -d osm_auto_tmp -f buildings.sql
10   ....
11  java -Xmx4000m -cp OSM2Shape.jar osm2shp.console.AddressBook
12  shp2pgsql -c -I addressbook.shp -s 4326 addressbook osm_auto_tmp > addressbook.sql
13   ....
14  psql -h localhost -p 5432 -U openlsfoo -q -d osm_auto_tmp -f addressbook.sql
15   ....
16  psql -h localhost -p 5432 -U openlsfoo -q -d osm_auto_tmp -f Grant.sql
17  psql -h localhost -p 5432 -U openlsfoo -q -d dummy -f Rest.sql
18 echo "*** End ***"
```

Listing 7.2: Update-Skript für Linux Server (vereinfacht)

8 Zusammenfassung & Ausblick

Die Ziele dieser Arbeit, OSM Daten für Location Based Services aufzubereiten und verschiedene Web Services mit den Daten einzurichten, weiterzuentwickeln oder neu zu implementieren, ist in überaus vielseitiger Weise gelungen. Die durch Gemeinschaftsarbeit gesammelten weltweiten Geodaten des OpenStreetMap Projektes konnten, durch die Entwicklung von verschiedenen Anwendungen, so aufbereitet werden, dass sie in unterschiedlichen Location Based Services verwendet werden können. Während der Aufbereitung war es anfangs schwer, einen Überblick über die zahlreichen OSM Tags und deren unterschiedliche Verwendung zu erhalten. Des Weiteren kommen bei der Bearbeitung und Filterung im Laufe der Zeit relativ große Datenmengen zustande. Erschwert wurde die Verarbeitung dadurch, dass es keine „standardisierten“ Eingaberichtlinien im OSM Projekt gibt. Jeder kann seine Daten eingeben wie er möchte. Es gibt lediglich „Vorgaben“ wie Daten (Tags) aussehen müssen, damit sie in der Karte eingezeichnet werden können. Dies bereitete an manchen Stellen bei der Filterung und Konvertierung der OSM Daten Probleme. Die Abdeckung der OSM Daten kann stark unterschiedlich sein. Im Gegensatz zu kommerziellen Kartenanbietern sind in Deutschland viele Städte im OSM Projekt umfangreicher erfasst. In ländlichen Gebieten kann es dagegen hin und wieder vorkommen, dass ein ganzes Dorf, bis auf die Hauptstraße, nicht in den OSM Daten vorhanden ist. Hieraus ergeben sich folgende Vor- und Nachteile für die Daten des OpenStreetMap Projektes:

Tabelle 8.1: Vor- und Nachteile der OSM Geodaten

Vorteile von OSM (+)	Nachteile von OSM (-)
+ produziert weltweit „freie“ Geodaten	- Unterschiedliche weltweite Abdeckung
+ große Aktualität und schneller Wachstum	- Qualität ist abhängig vom Kartierer
	- Falscheingaben

Das Ziel, OSM Daten für die Verwendung im vorhandenen Route Service und für die Einrichtung eines Web Feature und Web Map Service aufzubereiten, wurde dahingehend übertroffen, dass noch weitere OSM Daten für andere Dienste gefiltert und konvertiert werden konnten. Neben der Einrichtung in bestehende Anwendungen, wie dem GeoSer-

ver (WFS & WMS) oder dem WMS TileCache von MetaCarta, entstand mit dieser Arbeit ein OpenLS Framework (der *OpenLS Service*) aus dem Zusammenschluß von eigenen vorhandenen, im Laufe der Arbeit überarbeiteten, weiterentwickelten oder neu implementierten Web Services. Der OpenLS Service enthält vier der fünf möglichen OLS Core Services und eine Anzahl von weiteren nicht OLS-konformen Services.

Die **folgenden bestehenden Anwendungen** konnten für diese Arbeit verwendet werden:

- für den *WFS & WMS* eine GeoServer Installation
- und für den *WMS TileCache* die Implementierung von MetaCarta.

Die **folgenden eigenen Implementierungen** wurden unter kleinen Änderungen in den OpenLS Service übernommen:

- *OLS Presentation Service*, der vergleichbar mit einem WMS ist
- und der *Emergency Route Service*, der bei einer Routenplanung Gefahrengebiete, die von einem WFS bereitgestellt werden, beachtet.

Größere **Änderungen** erfolgten an folgenden Diensten, die in den OpenLS Service integriert wurden:

- am *OLS Route Service* für die Routenplanung,
- für die Geocodierung von Adressen und Reverse Geocodierung von Positionen am *OLS Location Utility Service*,
- der Erreichbarkeitsanalyse mit Hilfe des *Accessibility Analysis Service*
- und am *Extended Route Service* für erweiterte Fahranweisungen.

Komplett neu implementiert wurden für den OpenLS Service:

- der *OLS Directory Service* für eine POI-Umkreissuche
- und ein *RESTful OLS*, der eine vereinfachte Nutzung von OLS Diensten bietet.

Ein Hauptziel war, den OLS Route Service von Neis (2006) mit großen Datensätzen zu testen und gegebenenfalls zu verbessern. Nach den Untersuchungen mit großen Datensätzen konnte er in vielen Punkten optimiert und weiterentwickelt werden. Die wichtigsten und größten Änderungen betrafen die Routing-Graphen und den Routing-Algorithmus. Waren früher nur Auto- und Fußgänger-Routing möglich, konnte der RS für Fahrradfahrer erweitert werden. Bei der Änderung des Routing-Algorithmus wurde erst der bestehende Dijkstra überarbeitet, welcher wegen nicht Erreichen des gewünschten Erfolges später durch den A-Stern-Algorithmus ausgetauscht wurde. Der OLS Location Utility Service wurde ebenfalls an vielen Stellen überarbeitet und vervollständigt. Neben einer strukturierten Adress-Angabe kann jetzt eine sogenannte FreeForm-Variante angegeben werden.

Sie ermöglicht eine Kombination bei der Angabe von Postleitzahl, Ortsnamen und/oder Straßennamen für die Geocodierung einer Adresse. Vom OLS Directory Service war für eine POI-Umkreissuche keine Implementierung vorhanden. Die entstandene Umsetzung unterstützt vorerst nur die wichtigsten Elemente für ein Umkreissuche, sie kann aber in Folgearbeiten noch weiter ausgebaut werden. Durch den Extended Route Service ist eine erste prototypische Version eines Routenplaners entstanden, der in seiner jetzigen Version „Straßengegenstände“, wie Ampeln oder Mini-Kreisel, in seinen Fahranweisungen erwähnt. Im Service sind noch weitere Vorbereitungen getroffen worden, damit nachträglich auch andere Landmarken in den Fahranweisungen integriert werden können. Um die OLS Core Services auch vereinfacht nutzbar zu machen, wurde der RESTful OLS Dienst entwickelt. Er bietet den Vorteil gegenüber einem „normalen“ OLS Dienst, dass er komplett über die URL angesprochen werden kann, somit entfällt zumindest ein XML-Dokument für die Anfrage an einen Service.

Zusammenfassend ist mit dieser Arbeit ...
durch die eben erwähnten Dienste, deren Einrichtung auf einem Server und der Entwicklung einer Webseite für die Nutzung (veröffentlicht unter http://OpenRouteService.org), nach eigenem Kenntnisstand **der erste & freie ...**

- **Route Service,** (komplett auf OGC OpenLS Standard):
  - Auto Routenplaner (*Fastest* & *Shortest*)
  - Fußgänger Routenplaner (inkl. Treppen und Unterführungen)
  - Fahrrad Routenplaner
- **Web Feature & Web Map Service,** (OGC WFS/WMS Standard)
- **Geocoder & Reverse-Geocoder,** (OGC OpenLS Standard)
- **POI-Umkreissuche,** (OGC OpenLS Standard)
- **Erreichbarkeitsdienst,** (Accessibility Analysis Service)
- Weitere Routenplaner, zum Beispiel mit erweiterten Fahranweisungen oder der Möglichkeit, Vermeidungsgebiete zu digitalisieren oder zu nutzen.

basierend auf freien & durch Gemeinschaftsarbeit erhobenen Geodaten für Deutschland entstanden.

Die Besonderheit dabei ist, dass so gut wie alle Services über standardisierte Schnittstellen des OGC genutzt werden können. Somit können die Services in bestehende Systeme, welche die verwendeten Spezifikationen unterstützen, integriert werden.

Ergänzend werden die von Mayer (2008) bereitgestellten TMC-Meldungen auf OpenRou-

teService.org visualisiert und können bei einer Routenplanung (in prototypischer Form) berücksichtigt werden. Interessant ist die Untersuchung der Logdateien des Route Service. Traten am Anfang, nach der Veröffentlichung auf ORS, öfters Probleme bei einer Routenplanung (ca. bei jeder 100sten) wegen nicht verbundenen Straßen (Kanten) im Routing-Graph auf, so wurden diese Probleme bis zum Ende der Arbeit immer seltener. Dies lässt sich darauf zurückführen, dass einige OSM Nutzer mit Hilfe des Route Service die OSM oder ihre eigenen eingearbeiteten Daten validieren und sie gegebenenfalls bei Fehlern korrigierten.

Das OpenRouteService.org auch in anderen Bereichen, wie zum Beispiel im Bereich von öffentlichen Straßen, genutzt werden kann, zeigen die folgenden Abbildungen. Die Route von Abbildung 8.1 (a) verläuft im Gebäude der FH Hannover von einem Ort innerhalb des Gebäudes zur Cafeteria. Das ist ein Beispiel, in dem Wege im Bereich eines Gebäudes ins OSM Projekt eingearbeitet wurden und somit sogar für „Indoor“-Routing auf der Webseite genutzt werden kann (Anfang Juli 2008 war das Bild „Image of the week“ im OSM Wiki). Ein weiteres gutes Beispiel für ein geplante Strecke ist die Abbildung 8.1 (b). Dort wurde eine Route von den Büffeln, über die Robben, hin zu den Pinguinen im Berliner Zoo ermittelt.

(a) FH Hannover (b) Berliner Zoo

Abbildung 8.1: OpenRouteService.org in der FH Hannover & im Berliner Zoo

Erweiterung der Abdeckung von OpenRouteService.org

Kurz vor Ende der Arbeit konnten die Datenaufbereitungs-Tools für weitere Länder von Europa verwendet werden. Allerdings musste die Funktionsweise der Adressbuch-Erstellung für den Geocoder etwas geändert und weiterentwickelt werden, damit sie auch für andere Länder nutzbar wird. Dadurch bietet OpenRouteService.org nun nicht mehr nur alle erwähnten Services für Deutschland an, sondern auch für Österreich, die Schweiz, Italien,

Dänemark und Liechtenstein. Die folgende Abbildung 8.2 zeigt eine Routenplanung von Bern (Schweiz) nach Wien (Österreich).

Abbildung 8.2: Routing mit OpenRouteService.org von Bern nach Wien

Ausblick

Genauso vielseitig wie diese Arbeit ausgefallen ist, bieten sich auch Möglichkeiten für Erweiterungen oder Ausbaustufen der Services an. Angefangen beim Web Map Service, in dem noch nicht alle OSM Daten integriert sind, über die Abdeckung der unterstützten Länder, bis hin zum OLS Route Service, der aufgrund der vielen OSM Tags für die verschiedensten Routing-Varianten genutzt werden könnte. Im Folgenden werden zu verschiedenen Diensten dieser Arbeit mögliche Verwendungen oder Weiterentwicklungen beschrieben.

Wie bereits eben erwähnt, konnte OpenRouteService.org bereits kurz vor Ende der Arbeit auf weitere Länder ausgedehnt werden. Ein möglicher Schritt in der Zukunft wäre, die Abdeckung zusätzlich auf andere Länder oder ganz Europa zu erweitern. Jedoch müssten noch einige Ergänzungen, wie zum Beispiel die Unterstützung von Fähren, erfolgen. Sie müssten jeweils bei der Aufbereitung der Daten, im Routing-Algorithmus und bei der Erstellung von Fahranweisungen im RS Berücksichtigung finden.

Beim *WFS & WMS* würde sich anbieten, dass noch weitere darzustellende Objekte aus den OSM Daten ausgelesen und in die Services integriert werden. Dadurch könnte mit noch mehr unterschiedlichen Daten gearbeitet werden. Ein großer Vorteil von der OSM Datenhaltung in einem WFS ist, dass damit Räumliche-Operationen, wie zum Beispiel Verschneidungen mit anderen Daten, möglich werden. Durch die Einrichtung des WMS bieten sich ebenfalls Vorteile. Zum Beispiel können mit Hilfe von SLDs unterschiedliche Karten, je nach Verwendung, Serverseitig erstellt werden.

Für den *OLS Directory Service* könnten ebenfalls noch weitere OSM Daten genutzt werden. Daneben könnten die fehlenden Elemente, zur der Unterstützung der OLS Spezifikation, im Service implementiert werden. Interessante Möglichkeiten für weitere Verwendungen der Umkreissuche ergeben sich, wenn weitere OSM Tags bei der Dateneingabe von Nutzern verwendet werden. Zum Ende dieser Arbeit gab es bereits Tags für Kontaktdaten (Telefonnummer) und Öffnungszeiten. Stehen solche Daten in größeren Gebieten zur Verfügung, könnten bei der Suche bereits die Öffnungszeiten einfließen, Beispiel: *Welcher Bäcker hat Sonntags morgens im Umkreis von 2 Kilometern geöffnet?*

Ebenso bietet der *OLS Location Utility Service* verschiedene Möglichkeiten für Folgearbeiten. Einer der interessantesten dürfte dabei sein, das genaue geocodieren von Hausnummern zu realisieren. Wurden zu Beginn dieser Arbeit nicht viele Hausnummern in den OSM Datenbestand aufgenommen, so hat sich das bis zum Ende hin bereits geändert. Weiterhin könnte auch die Verwendung der Rechtschreibkorrektur über den Levenstein-Algorithmus in der Suche nach einer FreeForm-Adresse integriert werden. Momenten wird er nur bei der Suche nach einer strukturierten Adresse verwendet. Um den Geocoder in weiteren Ländern verwenden zu können, muss zuvor dessen Datenaufbereitungs-Tool weiterentwickelt werden, da die Struktur und Bestandteile einer Adresse je nach Land unterschiedlich sein können.

Im *OLS Route Service* könnte zum Beispiel die Nutzung der OSM Relations (Abbiegevorschriften) realisiert werden. Darüber hinaus gibt es weitere OSM Tags, die bis jetzt keine Beachtung finden, zum Beispiel: *maxspeed* für die maximal-erlaubte Geschwindigkeit auf einer Straße oder eine erweiterte Nutzung des *access*-Tags, das für Zutrittsbeschränkungen einer Straße verwendet wird. Die Fahranweisungen im RS oder die des Extended Route Service bieten gleichermaßen Bereiche, in denen Verbesserungen oder Optimierungen erfolgen könnten. Die Möglichkeiten verschiedene Routing Varianten aus den OSM Daten zu realisieren sind zahlreich. Zum Beispiel einen Routenplaner für Rollstuhlfahrer, Sehbehinderte, Schulkinder, LKW/Schwertransporte oder speziell für Radfahrer oder Wanderer. Insbesondere die Routenplanung für Radfahrer und Wanderer ist besonders vielseitig. Für beide könnten Parameter wie: Steigungen, Straßenbelag, Fortbewegungstyp (zum Beispiel Art des Fahrrads), Straßentyp, Landschaft und so weiter bei der Berechnung einer Route Einfluss nehmen. Die Anforderungen an die verschiedenen Routing-Anwendungen sind somit sehr individuell. Das Wichtigste dabei ist jedoch, dass sie dabei vor allem auf eine präzise Abbildung der Realität angewiesen sind.

Auch die entwickelte Webseite *OpenRouteService.org* könnte in Bezug auf die unterstützten Optionen der Dienste weiterentwickelt werden. Zum Beispiel wäre es möglich, die Eingabe von Zwischenpunkten für die Routeplanung oder die Angabe einer Start- oder

Zielzeit zu integrieren. Komfortabel wäre eine Änderung der Weboberfläche, damit die Angabe des Start- und Zielpunktes über einen Rechts-Klick mit der Maus in der Karte realisiert werden könnte. Bei der Entwicklung der Webseite war das mit der Openlayers Version 2.6 nicht möglich. Die leicht überarbeitete Erreichbarkeitsanalyse (Accessibility Analysis Service) könnte ebenfalls für weitere Fortbewegungsmittel erweitert werden. Momentan wird die Analyse nur für Autofahrer angeboten. Es wäre sinnvoll, die aktuelle Schnittstelle des AAS in die eines WPS zu ändern. Der Service könnte dann ebenfalls über eine OGC konforme Schnittstelle angeboten werden.

Schilling, Lanig, Neis und Zipf (2008) zeigten unter anderem, wie mit Hilfe von *Shuttle Radar Topography Mission* (SRTM)[1] Daten ein Höhenmodell und daraus ein Höhenprofil für eine Routenplanung erstellt werden kann. Dies wäre ein nützliche Ergänzung für den OLS Route Service oder für die Webseite ORS. Kurz vor dem Ende der Arbeit wurde der eingerichtete WMS mit den OSM Daten in das Projekt GDI3D[2] eingebunden. Die Abbildung 8.3 zeigt einen Ausschnitt aus der Anwendung XNavigator des Projektes, in der die Karte des OSM WMS auf das Höhenmodell gemappt wurde. Des Weiteren wurde der entwickelte OLS Directory Service für eine POI Umkreissuche mit den OSM Daten in den XNavigator integriert.

Abbildung 8.3: OSM WMS im XNavigator des Projektes GDI3D

Abschließend kann mit dieser Arbeit bestätigt werden, dass das OpenStreetMap Projekt mit seinen „frei“-verfügbaren, weltweiten Geodaten und mit den in dieser Arbeit entstanden Location Based Services, gutes Potential für viele Anwendungsmöglichkeiten im privaten, öffentlichen oder wirtschaftlichen Bereich hat beziehungsweise noch haben wird.

[1] SRTM - http://www.dlr.de/srtm/
[2] GDI3D - 3D-Geodateninfrastruktur für Heidelberg http://www.gdi-3d.de/

Literaturverzeichnis

Bremm, C. (2007). *Evaluation web-basierter Geoinformationssysteme.* Unveröffentlichte Diplomarbeit, Fachhochschule Mainz Fachbereich I - Studiengang Geoinformatik und Vermessung.

Brimicombe, A. J. (2002). *GIS - Where are the frontiers now?* Proceedings GIS 2002., Bahrain.

Dietze, L. & Zipf, A. (2007). Extending OGC Styled Layer Discriptor (SLD) for Thematik Cartography - Towards the ubiquitous use of advanced mapping functions through standardized visualization rules. In *4th Int. Symp. on LBS and Telecartography 2007.*

Dijkstra, E. W. (1959). *A note on two problems in connexion with graphs.* Numerische Mathematik. 1 (1959).

Douglas, D. & Peucker, T. (1973). Algorithms for the Reduction of the Number of Points Required to Represent a Digitised Line or its Caricature. In *Cartographica: The International Journal for Geographic Information and Geovisualization* (S. 112-122).

Douglas, K. & Douglas, S. (2005). *PostgreSQL. Developer's Library.* Sams Publishing.

Durlacher Research Ltd. (1999). *Mobile commerce report* (Tech. Rep.).

Dworschak, M. (2008). *Geografie: Wikipedia der Navigation.* Der Spiegel 22/2008. Online unter: http://www.spiegel.de/spiegel/0,1518,555174,00.html.

Fielding, R. T. (2000). *Architectural Styles and the Design of Network-based Software Architectures.* Unveröffentlichte Dissertation, University of California, Irvine.

Fritsch, L. & Muntermann, J. (2005). Aktuelle Hinderungsgründe für den kommerziellen Erfolg von Location Based Service-Angeboten. In *Konferenz Mobile Commerce Technologien und Anwendungen (MCTA).*

Haase, M., Zipf, A., Neis, P. & Camargo, V. de. (2008). Interoperable Routing Services in the Context of Evacuation Schemes due to Urban Flooding. In *EnviroInfo 2008 Conference. Environmental Informatics and Industrial Ecology. Lueneburg,*

Germany.

Hart, P. E., Nilsson, N. J. & Raphael, B. (1968). *A Formal Basis for the Heuristic Determination of Minimum Cost Paths.* IEEE Transactions on Systems Science and Cybernetics SSC4 (2).

Holone, H., Misund, G. & Holmstedt, H. (2007). Users Are Doing It For Themselves: Pedestrian Navigation With User Generated Content. In *Next Generation Mobile Applications, Services and Technologies, 2007. NGMAST '07. The 2007 International Conference on* (S. 91-99).

JOSM. (2006). *Java OpenStreetMap Editor.*
Online unter: http://josm.openstreetmap.de.

Juliao, R. (1998). Measuring Accessibilty. A GIS Based Methodology for Accessibility Evaluation. In *GIS PLANET 1998 Annual Conference Proceedings, Lisbon, Portugal.*

Levenshtein, V. I. (1965 (Russisch)). *Binary codes capable of correcting deletions, insertions, and reversals.* Doklady Akademii Nauk SSSR, 163(4) S. 845-848, 1965 (Russisch). Englische Übersetzung in: Soviet Physics Doklady, 10(8) S. 707-710, 1966.

Mapnik. (2005). *Mapnik C++/Python GIS Toolkit.*
Online unter: http://mapnik.org.

May, A. & Ross, T. (2006). Presence and quality of navigational landmarks: effect on driver performance and implications for design. In *Human Factors, 48* (S. 346-361).

Mayer, C. (2008). *Nutzung von Verkehrsfunkdaten des Traffic Message Channel über OGC Sensor Web.* Unveröffentlichte Diplomarbeit, Fachhochschule Mainz Fachbereich I - Studiengang Geoinformatik und Vermessung.

Miller, H. J. (1999). Measuring space-time accessibility benefits within transportation networks: Basic theory and computational procedures. In *Geographical Analysis, Volume 31* (S. 187-212).

MLP. (2002). *Mobile Location Protocol Specification. Verison 3.0.*
Online unter: http://www.openmobilealliance.org/tech/affiliates/lif/lifindex.html.

Neis, P. (2006). *Routenplaner für einen Emergency Route Service auf Basis der OpenLS Spezifikation.* Unveröffentlichte Diplomarbeit, Fachhochschule Mainz Fachbereich I - Studiengang Geoinformatik und Vermessung.

Neis, P. (2008). *(in Arbeit) Die OpenLS Core Services - Reihe: OpenGIS Essentials*

- Spezifikationen des OGC im Überblick. Hrsg: Andrae, C., Fitzke, J., Zipf, A. Wichmann Hüthig Verlag. Heidelberg.

Neis, P., Dietze, L. & Zipf, A. (2007). A Web Accessibility Analysis Service based on the OpenLS Route Service. In *10th AGILE International Conference on Geographic Information Science 2007, Aalborg University, Denmark.*

Neis, P., Schilling, A. & Zipf, A. (2007). 3D Emergency Route Service (3D-ERS) based on OpenLS Specifications. In *GI4DM 2007. 3rd International Symposium on Geoinformation for Disaster Management. Toronto, Canada.*

Neis, P. & Zipf, A. (2007). Realizing Focus Maps with Landmarks using OpenLS Services. In *4th International Symposium on LBS and Telecartography 2007. Hongkong.*

Neis, P. & Zipf, A. (2008a). *Extending the OGC OpenLS Route Service to 3D for an Interoperable Realization of 3D Focus Maps and Landmarks.* International Journal of Location Based Services (JLBS). (Special Issue on Telecartography).

Neis, P. & Zipf, A. (2008b). OpenRouteService.org is three times „Open“: Combining OpenSource, OpenLS and OpenStreetMaps. In *GIS Research UK (GISRUK). Manchester.*

Neis, P., Zipf, A., Helsper, R. & Kehl, A. (2007). Webbasierte Erreichbarkeitsanalyse - Vorschläge zur Definition eines Accessibility Analysis Service (AAS) auf Basis des OpenLS Route Service. In *REAL CORP 2007. Wien, Austria.*

Neis, P., Zipf, A. & Schmitz, S. (2008). *OpenRouteService.org - Combining Open Standards and Open Geodata.* The State of the Map. 2nd Open Street Maps Conference, Limerik. Ireland.

Neubauer, S. & Zipf, A. (2007). Suggestions for Extending the OGC Styled Layer Descriptor (SLD) Specification into 3D - Towards Visualization Rules for 3D City Models. In *Urban Data Management Symposium. UDMS 2007. Stuttgart.*

OLS. (2005). *OpenGIS Location Services (OLS) Version 1.1.*
Online unter: http://www.opengeospatial.org/standards/olscore.

O'Reilly, T. (2005). *What is web 2.0 - design patterns and business models for the next generation of software.*
Online unter: http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html.

OSM. (2004). *OpenStreetMap (OSM) - The Free Wiki World Map.*
Online unter: http://www.openstreetmap.org.

Osmarender. (2006). *Osmarender - Regelbasiertes Wiedergabe-Programm für SVG-Grafiken aus OSM-Daten.*
Online unter: http://wiki.openstreetmap.org/index.php/Osmarender.

Petsching, M. (2008). *Besichtigungstouren auf Basis von OpenLS-Diensten.* Unveröffentlichte Diplomarbeit, Fachhochschule Mainz Fachbereich I - Studiengang Geoinformatik und Vermessung.

Potlatch. (2007). *Online-Editor für OpenStreetMap.*
Online unter: http://wiki.openstreetmap.org/index.php/Potlatch.

Ramm, F. (2008). *OpenStreetMap - Die Frei Weltkarte.* Vortrag.
Online unter: http://www.wheregroup.com/files/osm_vortrag_ramm.pdf.

Ramm, F. & Topf, J. (2008). *OpenStreetMap - Die freie Weltkarte nutzen und mitgestalten.* Lehmanns Media, Berlin.

Raper, J., Gartner, G., Karimi, H. & Rizos, C. (2007). A critical evaluation of location based services and theis potential. *Journal of Location Based Services*.

Schiller, J. & Voisard, A. (2004). *Location-based Services.* Morgan Kaufmann.

Schilling, A., Lanig, S., Neis, P. & Zipf, A. (2008). DEM Processing and 3D Navigation using open standards and free geo data. In *3rd International Workshop on 3D Geo-Information. Seoul, South Korea.*

SFS. (1999). *OGC Simple Features Specification For SQL.*
Online unter: http://www.opengeospatial.org/standards/sfs.

SLD. (2002). *Styled Layer Descriptor Profile of the Web Map Service Implementation Specification.*
Online unter: http://www.opengeospatial.org/standards/sld.

Strategy Analytics. (2003). *Location Based Services: Strategic Outlook for Mobile Operators and Solutions Vendors* (Tech. Rep.). http://www.strategyanalytics.com.

Tiles@home. (2007). *Projekt Tiles@home - Programm zum verteilten Rendern von Kacheln für die Slippy Map.*
Online unter: http://tah.openstreetmap.org/.

Virrantaus, K., Markkula, J., Garmash, A. & Terziyan, Y. (2001). Developing GIS-Supported Location-Based Services. In *Proc. of WGIS 2001 Ű First International Workshop on Web Geographical Information Systems., Kyoto, Japan* (S. 423-432).

Weiser, A., Neis, P. & Zipf, A. (2006). Orchestrierung von OGC Web Diensten im Katastrophenmanagement - am Beispiel eines Emergency Route Service auf Basis

der OpenLS Spezifikation. *GIS - Zeitschrift für Geoinformatik*, 35-41.

WFS. (2002). *Web Feature Service (WFS) Implementation Specification.* Online unter: http://www.opengeospatial.org/standards/wfs.

WMS. (2001). *Web Map Service (WMS) Implementation Specification.* Online unter: http://www.opengeospatial.org/standards/wms.

YellowMap. (2008). *Technologie - Überblick Mobile Systeme.* Online unter: http://www.yellowmap.com/technologie-mobilesysteme.html - abgerufen am 10.07.08.

Zipf, A. (2007). Google Maps and Google Earth in Kommunen!? Wieviel Web 2.0 - 3D brauchen Kommunen und gibt es interoperable Alternativen? *geoNews. 02/2007. RM-Data. Austria*.

Zipf, A. & Röther, S. (2000). *Tourenvorschläge für Stadttouristen mit dem ArcView Network Analyst.* Liebig (Hrsg.)(2000): ArcView Arbeitsbuch. Hüthig Verlag. Heidelberg.

Zipf, A. & Tschirner, S. (2005). Finding GI-datasets that otherwise would have been lost - GeoXchange - a OGC standards-based SDI for sharing free geodata. In *2nd International Workshop on Geographic Information Retrieval (GIR 2005) Bremen, Germany. Springer.*

Abbildungsverzeichnis

2.1 LBS als eine Verschneidung von Technologien (Brimicombe, 2002) . . . 8
2.2 Grundprinzip von Location Based Services (vereinfacht) 9
2.3 Die Core Services der OpenLS-Spezifikation Version 1.1 12
2.4 Der GeoMobility Server aus OLS (2005) 12
2.5 Gateway Service - Zugriff auf Positionsdaten mit MLP (geändert, aus MLP (2002)) 13
2.6 Karten mit und ohne SLD gestyled (OpenStreetMap Daten Bonn) 17
2.7 Ergebnisse der Erreichbarkeitsanalyse nach Neis et. al. (2007) 18

3.1 „The 5 steps to making a map“ (aus OSM Wiki) 23
3.2 Überblick der OSM Komponenten (geändert aus Ramm (2008)) 24
3.3 Vereinfachte Darstellung des OSM Datenmodells aus (Ramm & Topf, 2008) 25
3.4 OSM Objekttypen: Node und Way (und closed Ways) 25
3.5 Schema-Diagramm einer OSM Datei Version 0.5 (*.osm) 27
3.6 OpenStreetMap Datenbank Statistik 31

4.1 Aufbereitung der OSM Daten für OLS Route Service 36
4.2 Mögliche andere Probleme bei der Aufbereitung von OSM Daten 36
4.3 Aufbereitung der OSM Daten für OLS Route Service, Part II 37
4.4 OSM Daten für Accessibility Analysis Service 40

5.1 Funktionsweise TileCache 51

6.1 Gesamtarchitektur der verwendeten, weiterentwickelten und implementierten Dienste 53
6.2 Architektur des OpenLS Service 55
6.3 Sequenzdiagramm des OLS Route Service 58
6.4 Performancevergleich zwischen den Routing-Algorithmen 61
6.5 „Problem“ bei der Start- & Zielangabe bei einer Routenberechnung . . . 63
6.6 Besondere Routing-Situationen (Alt) 63
6.7 Besondere Routing-Situationen (Neu) 64

6.8 Besondere Routing-Situationen - *AvoidList* 64
6.9 Start (oder Ziel) durch Adresse angegeben 65
6.10 Ergebnis einer Umkreissuche in Bonn 68
6.11 Ergebnis einer Erreichbarkeitsanalyse für Bonn 69
6.12 Sequenzdiagramm des Emergency Route Service 70
6.13 Routenberechnung mit Verwendung des ERS & TMC 71
6.14 Ergebnis einer Routenplanung (Extended Route Service) 72
6.15 Sequenzdiagramm von OpenRouteService.org 74
6.16 Webseite OpenRouteService.org 75

8.1 OpenRouteService.org in der FH Hannover & im Berliner Zoo 85
8.2 Routing mit OpenRouteService.org von Bern nach Wien 86
8.3 OSM WMS im XNavigator des Projektes GDI3D 88

Listings

3.1 OSM Node 26
3.2 OSM Way 26
3.3 OSM Relation 27

6.1 XML-Beispiel für eine strukturierte Adressangabe 66
6.2 XML-Beispiel für das freeFormAddress-Element 66

7.1 Aufruf der Java Konsolenanwendung 78
7.2 Update-Skript für Linux Server (vereinfacht) 81

A.1 OLS Route Service Request (vereinfacht) . . . . . a
A.2 OLS Route Service Response (vereinfacht) . . . . . b
A.3 OLS Location Utility Service Request (Geocoder) (vereinfacht) . . . . . d
A.4 OLS Location Utility Service Response (Geocoder) (vereinfacht) . . . . . d
A.5 OLS Location Utility Service Request (RevserseGeocoder) (vereinfacht) . e
A.6 OLS Location Utility Service Response (RevserseGeocoder) (vereinfacht) e
A.7 OLS Directory Service Request (vereinfacht) . . . . . f
A.8 OLS Directory Service Response (vereinfacht) . . . . . g
A.9 Accessibility Analysis Service Request (vereinfacht) . . . . . h
A.10 Accessibility Analysis Service Response (vereinfacht) i

Tabellenverzeichnis

3.1 Geschichtliche Entstehung und Entwicklung des OSM Projektes 21
3.2 OSM Map Features: Übersicht der wichtigsten Keys 28

4.1 OSM Straßen (`highway`-Key) für OLS Route Service (Auto) 33
4.2 OSM Straßen (`highway`-Key) für OLS Route Service (Fussgänger) 33
4.3 OSM Straßen (`highway`-Key) für OLS Route Service (Fahrrad) 34
4.4 Tabellenschema für OLS Route Service 38
4.5 Tabellenschema für OLS Location Utility Service (Geocoder) 41
4.6 OSM *„Key-Value“*-Paare für OLS Location Utility Service 42
4.7 OSM *„Key-Value“*-Paare für OLS Directory Service 43
4.8 OSM Deutschland Datensatz vom 19.03.2008 44
4.9 OSM Deutschland Datensatz vom 29.07.2008 45
4.10 Statistik für OSM Deutschland Daten vom 19.03.2008 bis 29.07.2008 . . 46

6.1 Performancevergleich Routing-Algorithmen (Datensatz 19.03.2008) . . . 61
6.2 Antwortzeiten des OLS Route Service (Datensatz 19.03.2008) 62
6.3 Statistiken der Webseiten-Nutzung von OpenRouteService.org 76

7.1 URLs des OpenLS Service und der weiteren Dienste dieser Arbeit 80

8.1 Vor- und Nachteile der OSM Geodaten 82

A Request- & Response Beispiele

A.1 OLS Route Service

Der weiterentwickelte RS unterstützt nahezu alle möglichen Elemente und Attribute der OLS Spezifikation Version 1.1 (vgl. Neis (2006)). Das nachfolgende XML-Beispiel ist eine etwas vereinfachte Anfrage an einen OLS RS, in dem nicht alle möglichen Elemente und Attribute verwendet werden. Über das *xls:lang*-Attribut (Zeile 2) kann die Sprache der Fahranweisung angegeben werden. In Zeile 5 kann die Längeneinheit (*distanceUnit*) vorgegeben und das Speichern der Route auf dem Server veranlasst werden (*provideRouteHandle*). Über das *RoutePreference*-Element in Zeile 7 kann die Art der Routenplanung geändert werden. Die Koordinaten des Starts und Ziels sind in den Zeilen 7 & 16 zu finden. Das genutzte SRS sollte angegeben werden, weil bei Nicht-Angabe standardmäßig bei allen Geometrien WGS-84 (EPSG:4326) angenommen wird. Durch die Zeilen 21 bis 25 können die optionalen Ergebnisse zur berechneten Route angefordert werden: Fahranweisungen (Zeile 21), Geometrie der Route (Zeile 22) und Karte mit eingezeichneter Route (Zeile 23 bis 25). Kleine Besonderheit des RSs von dieser Arbeit: es wird das KML-Format unterstützt, weiterhin sind auch andere Formate wie png oder jpeg möglich. Weitere mögliche Elemente oder Attribute, die nicht im Beispiel vorkommen sind: Angabe von Start- oder Zielzeit (*expectedStartTime*/*expectedEndTime*), Zwischenpunkte (*ViaPoints*) und eine Vermeidungsliste (*AvoidList*). Bei den Elementen *RouteInstructionsRequest*, *RouteGeometryRequest* und *RouteMapRequest* sind ebenfalls noch weitere Attribute möglich, vgl. dazu OLS (2005).

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <xls:XLS .... version="1.1" xls:lang="de">
3 <xls:RequestHeader/>
4 <xls:Request methodName="RouteRequest" requestID="123456789" version="1.1">
5   <xls:DetermineRouteRequest distanceUnit="KM" provideRouteHandle="true">
6     <xls:RoutePlan>
7       <xls:RoutePreference>Fastest</xls:RoutePreference>
8       <xls:WayPointList>
9         <xls:StartPoint><xls:Position>
```

```
10          <gml:Point srsName="EPSG:4326">
11             <gml:pos>7.0999616 50.7254044</gml:pos>
12          </gml:Point></xls:Position>
13        </xls:StartPoint>
14        <xls:EndPoint><xls:Position>
15          <gml:Point srsName="EPSG:4326">
16             <gml:pos>7.1011011 50.7254122</gml:pos>
17          </gml:Point></xls:Position>
18        </xls:EndPoint>
19      </xls:WayPointList>
20    </xls:RoutePlan>
21    <xls:RouteInstructionsRequest format="text/plain"/>
22    <xls:RouteGeometryRequest/>
23    <xls:RouteMapRequest>
24       <xls:Output format="kml" height="400" width="400" BGcolor="#ffffff"/>
25    </xls:RouteMapRequest>
26  </xls:DetermineRouteRequest>
27 </xls:Request>
28 </xls:XLS>
```

Listing A.1: OLS Route Service Request (vereinfacht)

Aufgrund der Anfrage (XML-Beispiel A.1) berechnet der RS die Route und gibt das Ergebnis in XML formatiert zurück (siehe vereinfachtes XML-Beispiel A.2). In der Antwort muss mindestens die Zusammenfassung der Route enthalten sein (Zeile 7-13 - *RouteSummary*). Die Geometrie der Route steht im *RouteGeometry*-Element (Zeile 14-20). In den Zeilen 21-27 ist die Liste von Fahranweisungen vorhanden. Bei jeder Fahranweisung (Element *RouteInstruction*) sind weitere Attribute wie Dauer, Distanz und die eigentliche Anweisung enthalten. Das letzte mögliche Element ist für die Karte zuständig (*RouteMap*, Zeile 28-35). Dort sind verschiedene Informationen wie Dateiformat, Breite/Höhe und die URL, wo die Karte abgerufen werden kann, beinhaltet.

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <xls:XLS .... >
3    <xls:ResponseHeader/>
4    <xls:Response requestID="123456789" version="1.1" numberOfResponses="1">
5      <xls:DetermineRouteResponse>
6        <xls:RouteHandle routeID="1218061479598" serviceID="1.1"/>
7        <xls:RouteSummary>
8          <xls:TotalTime>PT17S</xls:TotalTime>
9          <xls:TotalDistance uom="KM" value="0.1"/>
10         <xls:BoundingBox srsName="EPSG:4326">
11           <gml:pos>7.09996 50.72523</gml:pos><gml:pos>7.10110 50.72555</gml:pos>
12         </xls:BoundingBox>
```

```
13      </xls:RouteSummary>
14      <xls:RouteGeometry>
15        <gml:LineString srsName="EPSG:4326">
16          <gml:pos>7.0999616 50.7254044</gml:pos>
17            ....
18          <gml:pos>7.1011011 50.7254122</gml:pos>
19        </gml:LineString>
20      </xls:RouteGeometry>
21      <xls:RouteInstructionsList xls:lang="de">
22        <xls:RouteInstruction duration="PT0S" description="Anweisungsnr. 1">
23          <xls:Instruction>Sie starten auf: Weberstrasse</xls:Instruction>
24          <xls:distance value="0" uom="KM"/>
25        </xls:RouteInstruction>
26        ....
27      </xls:RouteInstructionsList>
28      <xls:RouteMap description="Overview">
29        <xls:Content format="kml" height="400" width="400">
30          <xls:URL>
31                http://openls.giub.uni-bonn.de/openls-osm/tmp/1218061479598.kml
32          </xls:URL>
33        </xls:Content>
34         ....
35        </xls:RouteMap>
36    </xls:DetermineRouteResponse>
37  </xls:Response>
38 </xls:XLS>
```

Listing A.2: OLS Route Service Response (vereinfacht)

A.2 OLS Location Utility Service

Bei einem Request an den OLS Location Utility Service können die hier in den verschiedenen XML-Beispielen verwendeten Elemente und Attribute genutzt werden. Es gibt zwar noch weitere (vgl. Neis (2008) und OLS (2005)), aber diese waren für diese Arbeit und die hier vorgesehene Nutzung nicht relevant. Eine Anfrage für eine Geocodierung, ist im XML-Beispiel A.3 zu sehen. Das einzig wichtige Element dabei ist das *freeFormAddress*-Element, das die Adresse enthält (Zeile 7-9). Weitere Informationen über Request- und Response-Parameter siehe Neis (2008) und OLS (2005).

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <xls:XLS .... version="1.1">
3    <xls:RequestHeader/>
4    <xls:Request methodName="GeocodeRequest" requestID="123456789" version="1.1">
5      <xls:GeocodeRequest>
6        <xls:Address countryCode="DE">
7          <xls:freeFormAddress>53115 bonn, beringstr.</xls:freeFormAddress>
8        </xls:Address>
9      </xls:GeocodeRequest>
10   </xls:Request>
11 </xls:XLS>
```

Listing A.3: OLS Location Utility Service Request (Geocoder) (vereinfacht)

Die Antwort des Services erfolgt ebenfalls wieder in XML. Neben der Koordinate (Zeile 9) wird die geocodierte Adresse in einer strukturierten Form zurückgegeben (Zeile 11-18). Zusätzlich ist in Zeile 19 im *GeocodeMatchCode*-Element ein Wert für die Übereinstimmung der angefragten und zurückgelieferten Adresse enthalten.

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <xls:XLS .... version="1.1">
3    <xls:ResponseHeader/>
4    <xls:Response requestID="123456789" version="1.1" numberOfResponses="1">
5      <xls:GeocodeResponse>
6        <xls:GeocodeResponseList numberOfGeocodedAddresses="1">
7          <xls:GeocodedAddress>
8            <gml:Point>
9              <gml:pos srsName="EPSG:4326">7.0921596 50.7272263</gml:pos>
10           </gml:Point>
11           <xls:Address countryCode="de">
12             <xls:StreetAddress>
13               <xls:Street officialName="Beringstraße"/>
14             </xls:StreetAddress>
15             <xls:Place type="CountrySubdivision">Nordrhein-Westfalen</xls:Place>
```

```
16            <xls:Place type="Municipality">Bonn</xls:Place>
17            <xls:PostalCode>53115</xls:PostalCode>
18          </xls:Address>
19          <xls:GeocodeMatchCode accuracy="1.0"/>
20        </xls:GeocodedAddress>
21      </xls:GeocodeResponseList>
22    </xls:GeocodeResponse>
23  </xls:Response>
24 </xls:XLS>
```

Listing A.4: OLS Location Utility Service Response (Geocoder) (vereinfacht)

Bei einer Anfrage zur Reverse Geocodierung ist lediglich die Position im XML-Request enthalten (siehe Zeile 9).

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <xls:XLS .... version="1.1">
3    <xls:RequestHeader/>
4    <xls:Request methodName="ReverseGeocodeRequest"
5        requestID="123456789" version="1.1">
6      <xls:ReverseGeocodeRequest>
7        <xls:Position>
8          <gml:Point>
9            <gml:pos>7.098375 50.735794</gml:pos>
10         </gml:Point>
11       </xls:Position>
12     </xls:ReverseGeocodeRequest>
13   </xls:Request>
14 </xls:XLS>
```

Listing A.5: OLS Location Utility Service Request (RevserseGeocoder) (vereinfacht)

Die Antwort des Location Utility Service enthält wieder das *Point*- und *Address*-Element. In Zeile 18 ist noch das *SearchCentreDistance*-Element enthalten, das die Distanz zwischen zurückgegebener Adresse und der angefragten Position wiedergibt.

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <xls:XLS .... version="1.1">
3    <xls:ResponseHeader/>
4    <xls:Response requestID="123456789" version="1.1" numberOfResponses="3">
5      <xls:ReverseGeocodeResponse>
6        <xls:ReverseGeocodedLocation>
7          <gml:Point>
8            <gml:pos srsName="EPSG:4326">7.0983721 50.7357774</gml:pos>
```

```
9         </gml:Point>
10        <xls:Address countryCode="de">
11          <xls:StreetAddress>
12            <xls:Street officialName="Sternstraße"/>
13          </xls:StreetAddress>
14          <xls:Place type="CountrySubdivision">Nordrhein-Westfalen</xls:Place>
15          <xls:Place type="Municipality">Bonn</xls:Place>
16          <xls:PostalCode>53111</xls:PostalCode>
17        </xls:Address>
18        <xls:SearchCentreDistance uom="M" value="2.0"/>
19      </xls:ReverseGeocodedLocation>
20    </xls:ReverseGeocodeResponse>
21  </xls:Response>
22 </xls:XLS>
```

Listing A.6: OLS Location Utility Service Response (RevserseGeocoder) (vereinfacht)

A.3 OLS Directory Service

Bei einem Request (siehe XML-Beispiel A.7) können anfangs die gewünschte Längeneinheit sowie das Sortierkriterium angegeben werden (Zeile 4). In Zeile 9 steht die Koordinate und in Zeile 12 die Distanz, innerhalb welcher gesucht werden soll (*Minimum-Distance*-Element). Im *POIProperties*- und *POIProperty*-Element sind das Verzeichnis (hier „OSM“) und die für diese Arbeit realisierte Kategorie „public_tran“ enthalten. Im letzteren Element sind gemäß der Datenaufbereitung entweder der POI-Type oder die Kategorie im *value*-Attribut vorhanden (vgl. Tabelle *4.7 OSM „Key-Value“-Paare für OLS Directory Service*).

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <xls:XLS .... version="1.1">
3   <xls:RequestHeader/>
4   <xls:Request methodName="DirectoryRequest" requestID="123456789" version="1.1">
5     <xls:DirectoryRequest distanceUnit="M" sortCriteria="Distance">
6       <xls:POILocation>
7         <xls:WithinDistance>
8           <xls:Position>
9             <gml:Point>
10              <gml:pos>7.092298 50.733306</gml:pos>
11            </gml:Point>
12          </xls:Position>
13        <xls:MinimumDistance value="500" uom="M"/>
```

```
14          </xls:WithinDistance>
15        </xls:POILocation>
16        <xls:POIProperties directoryType="OSM">
17          <xls:POIProperty name="Keyword" value="public_tran"/>
18        </xls:POIProperties>
19      </xls:DirectoryRequest>
20    </xls:Request>
21  </xls:XLS>
```

Listing A.7: OLS Directory Service Request (vereinfacht)

Die Antwort enthält, je nach Anfrage und Distanz, mehrere *POIContext*-Elemente. Jedes Element beinhaltet neben der Position (Zeile 9) den Namen, eine Beschreibung und die ID des POIs (Zeile 6). Als letztes wird bei jedem gefundenen POI noch die Distanz von der angefragten Koordinate bis zum jeweiligen POI zurückgegeben (Zeile 12). Mehr Informationen über weitere Request- und Response-Parameter siehe Neis (2008) und OLS (2005).

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <xls:XLS .... version="1.1">
3    <xls:ResponseHeader/>
4    <xls:Response requestID="123456789" version="1.1" numberOfResponses="3">
5      <xls:DirectoryResponse>
6        <xls:POIContext>
7          <xls:POI POIName="Herwarthstraße" description="bus_stop" ID="265956872">
8            <gml:Point>
9              <gml:pos srsName="EPSG:4326">7.092758 50.7330345</gml:pos>
10           </gml:Point>
11         </xls:POI>
12         <xls:Distance uom="M" value="59"/>
13       </xls:POIContext>
14       ....
15     </xls:DirectoryResponse>
16   </xls:Response>
17 </xls:XLS>
```

Listing A.8: OLS Directory Service Response (vereinfacht)

A.4 Accessibility Analysis Service

Wie im XML-Beispiel (A.9) für die Anfrage an den AAS zu sehen ist, ist die Struktur und auch die Element-/Attributbezeichnung an die OGC OLS Spezifikation angelehnt. Die wichtigsten zwei Elemente sind: *LocationPosition* (Zeile 10-16) und *PolygonPreference* (Zeile 20-22). Im ersten Element wird die Position angegeben, von der die Erreichbarkeit ermittelt werden soll. Im zweiten Element wird die Eigenschaft für die Darstellung des Polygons festgelegt. Drei Varianten sind möglich: Isolinien, konvexe Hülle und gepufferte Straßen (Unterschiede siehe Abbildung 2.7).

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <aas:AAS .... version="1.0">
3    <aas:RequestHeader/>
4    <aas:Request methodName="AccessibilityRequest" requestID="1234567" version="1.0">
5      <aas:DetermineAccessibilityRequest>
6        <aas:Accessibility>
7          <aas:AccessibilityPreference>
8            <aas:Time Duration="PT0H02M00S"></aas:Time>
9          </aas:AccessibilityPreference>
10         <aas:LocationPoint>
11           <aas:Position>
12             <gml:Point xmlns:gml="http://www.opengis.net/gml" srsName="EPSG:4326">
13               <gml:pos>7.08845 50.73991</gml:pos>
14             </gml:Point>
15           </aas:Position>
16         </aas:LocationPoint>
17       </aas:Accessibility>
18       <aas:AccessibilityOutputRequest Coordinate="true" Distance="true"
19         DistanceUnit="KM" Name="true" Time="true"/>
20       <aas:AccessibilityGeometryRequest>
21         <aas:PolygonPreference>Detailed</aas:PolygonPreference>
22       </aas:AccessibilityGeometryRequest>
23     </aas:DetermineAccessibilityRequest>
24   </aas:Request>
25 </aas:AAS>
```

Listing A.9: Accessibility Analysis Service Request (vereinfacht)

Die Antwort des AAS enthält im wesentlichen ein oder mehrere Polygone, die das Gebiet der Erreichbarkeitkeit repräsentieren (Zeile 13-26).

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <aas:AAS .... version="1.0">
3    <aas:ResponseHeade/>
4    <aas:Response requestID="1234567" version="1.0">
5      <aas:AccessibilityResponse>
6        <aas:AccessibilitySummary>
7          <aas:NumberOfLocations>0</aas:NumberOfLocations>
8          <aas:BoundingBox srsName="EPSG:4326">
9            <gml:pos>7.0630597 50.7239076</gml:pos>
10           <gml:pos>7.1086217 50.7556472</gml:pos>
11         </aas:BoundingBox>
12       </aas:AccessibilitySummary>
13       <aas:AccessibilityGeometry>
14         <gml:Polygon srsName="EPSG:EPSG:4326">
15           <gml:exterior>
16             <gml:LinearRing>
17               <gml:pos>7.1079614 50.7379072</gml:pos>
18               <gml:pos>7.1081132 50.7377997</gml:pos>
19               ....
20               <gml:pos>7.1079614 50.7379072</gml:pos>
21               <gml:pos>7.1079614 50.7379072</gml:pos>
22             </gml:LinearRing>
23           </gml:exterior>
24         </gml:Polygon>
25         ....
26       </aas:AccessibilityGeometry>
27     </aas:AccessibilityResponse>
28   </aas:Response>
29 </aas:AAS>
```

Listing A.10: Accessibility Analysis Service Response (vereinfacht)

B Inhaltsverzeichnis CD

1. Testdaten für Nordrhein-Westfalen - enthält zwei Ordner:

Original OSM Daten

Aufbereitete OSM Daten als Shapefiles und SQL-Skripte

2. Software - enthält vier Ordner:

Datenaufbereitungs-Tool

OpenLS Service

Webseite OpenRouteService.org

SLD Dateien und Symbole für den WMS

3. Masterarbeit als PDF-Dokument

Erklärung

Hiermit erkläre ich, dass ich die vorliegende Masterarbeit selbständig angefertigt habe. Es wurden nur die in der Arbeit ausdrücklich benannten Quellen und Hilfsmittel benutzt. Wörtlich oder sinngemäß übernommenes Gedankengut habe ich als solches kenntlich gemacht.

....

(Ort, Datum) (Unterschrift)