Übernahme von Geodatenbeständen aus Open Street Map und Bereitstellung einer effizienten Zugriffsmöglichkeit für ortsbezogene Dienste

Jörg Roth

Fakultät für Informatik
Ohm-Hochschule Nürnberg
Kesslerplatz 12, 90489 Nürnberg
Joerg.Roth@Ohm-Hochschule.de

Zusammenfassung

Unter anderem durch das Open-Street-Map-Projekt stehen mittlerweile umfangreiche Geodatenbestände zur Verfügung. Bei der Verwendung in eigenen Projekten entstehen keine Lizenzkosten, so dass sich entsprechende Datenbestände auch für kleinere Projekte eignen. In diesem Beitrag wird ein Ansatz vorgestellt, wie ein Geodatenbestand für eigene ortsbezogene Dienste übernommen und durch ein effizientes Zugriffsverfahren eingesetzt werden kann.
Die Datenübernahme erfordert nicht nur eine Anpassung von Formaten. Neben der Geometrie ist insbesondere die Klassifikation von Geo-Objekten ein wichtiges Merkmal für die automatische Auswertung durch Dienste. Eine grundlegende Funktion der Datenübernahme ist daher, die umfangreichen und teilweise inkonsistenten Klassifikationsmöglichkeiten von Open Street Map in ein einfaches und dennoch aussagekräftiges Klassifikationsschema zu konvertieren.
Die hier vorgestellte Datenhaltung und der Zugriff auf die Datenbestände basieren auf Standards. Zur Speicherung wurde eine konventionelle relationale SQL-Datenbank *ohne* räumliche Erweiterung verwendet. Für die Speicherung von Geometrien wurde das Standard-Format *Well-Known Binary* eingesetzt. Dieses ermöglicht eine einfache Weiterverarbeitung, da hierfür Software-Pakete für viele Plattformen existieren. Die geometrische Suche wird durch einen neuartigen Indizierungsmechanismus ermöglicht, der vollständig in SQL formuliert werden kann und traditionelle Index-Spalten verwendet.
Die Effektivität der Datenhaltung wurde anhand von Performance-Tests überprüft. Einige Studentenprojekte demonstrieren, dass man den Datenbestand ohne großen Einarbeitungsaufwand für eigene ortsbezogene Dienste einsetzen kann.

1 Übernahme der Open-Street-Map-Datenbestände

Das *Hybris*-Projekt [Roth08] hat zum Ziel, eine allgemeine Plattform für den Import, die Bearbeitung sowie die Bereitstellung von Geodaten für ortsbezogene Dienste anzubieten. Die im Projekt entwickelte Umgebung unterstützt zahlreiche Import-Formate; in diesem Beitrag wird die Datenübernahme aus dem Open-Street-Map-Datenbestand [OSM] beschrieben. Open Street Map (im Folgenden auch *OSM* genannt) unterstützt prinzipiell drei Arten der Nutzung:

- Vorberechnete Karten können als Bilddateien übernommen werden. Diese können in ortsbezogene Kartenanwendungen eingeblendet werden, erlauben aber keine Auswertung der Objektdaten.
- Über eine HTTP-Schnittstelle können Vektor-Daten in einer Fläche von bis zu ca. 1000 km^2 direkt aus der Open-Street-Map-Datenbank geladen werden.
- Über so genannte *Planet Files* [OSMPLA] kann der gesamte Vektor-Datenbestand in einem XML-Format übertragen werden. Planet Files gibt es für verschiedene Ausschnitte des Datenbestandes, so z.B. für einzelne Staaten oder Teile von Staaten (in Deutschland z.B. für Bundesländer).

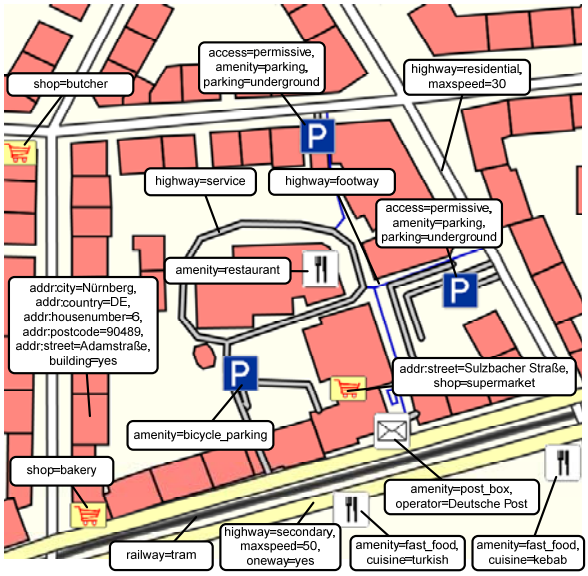
In diesem Beitrag wird der letzte Weg beschrieben, der den vollen Zugriff auf alle gesammelten Daten in eigenen Anwendungen erlaubt. Einen eigenen Datenbestand zu pflegen, erfordert zwar gegenüber dem Online-Zugriff einen erhöhten Aufwand, man hat aber diverse Vorteile:

- Man kann verschiedene geometrische und inhaltliche Suchen realisieren und ist nicht auf die Möglichkeiten der Online-Schnittstelle von OSM angewiesen.
- Man kann die eigene Server-Umgebung an die Anforderungen des Dienstes anpassen, z.B. bezüglich Zugriffsregelungen, Performance oder Ausfallsicherheit.
- Man kann den Datenbestand modifizieren und an eigene Anforderungen anpassen. Darüber hinaus kann man den OSM-Datenbestand mit weiteren Datenbeständen verknüpfen.
- Man ist zeitlich von den ständigen Änderungen des OSM-Datenbestandes entkoppelt. Insbesondere die schnelle Änderungsrate der OSM-Objektklassifikation ist für eigene Projekte kritisch, da Änderungen dazu führen können, dass Dienste nicht mehr wie erwartet laufen.

Die Übernahme des Datenbestandes ist nicht trivial und betrifft neben rein geometrischen Konvertierungen vor allem die Objektklassifikation.

1.1 Klassifikation von Objekten

Durch eine sinnvolle Objektklassifikation sollte eine Anwendung oder ein Dienst leicht solche Objekte selektieren können, die für eine spezielle Funktion interessant sind. So möchte beispielsweise eine Assistenz-Anwendung für Touristen die Unterkünfte, historischen Bauten oder Museen aus der Datenbank laden, während eine Wegeplanung sich vorwiegend für linienförmige Objekte des Straßenverkehrs interessiert.
Viele Ansätze der Objektklassifikation konzentrieren sich auf rein geometrische Eigenschaften und bilden inhaltliche Klassen nur in Form einiger weniger Oberklassen ab [Fra97]. Für ortsbezogene Anwendungen ist aber oft eine Klassifikation von konkreten inhaltlichen Eigenschaften notwendig. Darüber hinaus werden durch den Anbieter von Geodaten ständig neue Objekte als interessant eingestuft und deren Position erfasst. Starre Klassifikationen veralten daher schnell.
Aus diesen Gründen betrachtet Open Street Map auch die Klassifikation von Objekten als Community-Prozess [OSMMF]. Das ist einerseits ein Vorteil, da sich das Schema an den schnell wachsenden Bestand neuer Objekttypen anpassen kann. Anderseits ist es ein Nachteil, wenn das Schema von Software ausgewertet werden soll, da auch diese ständig angepasst werden muss.
Open Street Map bietet ein einfaches Verfahren an Objekte zu klassifizieren, indem Geo-Objekten beliebig viele Paare von *Schlüssel-Wert* (jeweils eine Zeichenkette) zugeordnet werden. Diese Paare unterliegen keiner Restriktion, müssen also nicht aus einer festen Liste stammen. Jeder Teilnehmer, der einen Datensatz beisteuert, kann beliebige Paare eintragen. Bezüglich vieler Objekttypen gibt es Konsens über sinnvolle Richtlinien für die Zusammensetzung von Paaren [OSMMF], die Einhaltung der Richtlinien wird aber nicht technisch überprüft. Die vielfältigen Möglichkeiten der Beschreibung sind in Abbildung 1 (links) illustriert.



Haupt-Schlüssel-Wert-Paar	Bedeutung	Anteil %
`highway=residential`	Innerortsstraße	14,36
`building=yes`	Gebäude	9,14
`highway=track`	Wald-, Feldweg	8,65
`highway=service`	Auffahrten	3,96
`highway=footway`	Fußweg	3,54
`highway=unclassified`	Außerörtliche unklassif. Str.	2,32
`bicycle=yes`	Radweg	2,09
`power=tower`	Strommast	1,91
`highway=secondary`	Landstraße	1,63
`highway=tertiary`	Kreisstraße	1,62
Weitere Haupt-Paare		50,78
Zahl aller vorkommender verschiedener Haupt-Paare: 52085		100

Abbildung 1: Beispiele von OSM-Klassifikationen (links) und Verteilung der zehn häufigsten Haupt-Schlüssel-Wert-Paare (rechts)

Es ist offensichtlich, dass eine automatische Weiterverarbeitung von Geo-Objekten nur dann funktioniert, wenn eine konsistente Auszeichnung der Objekte durch die Ersteller erfolgt. Der Vorschlag für eine Vereinheitlichung wird mit Hilfe von Diskussionsforen ständig weiterentwickelt und kann wie folgt skizziert werden:

- Es gibt einen Hauptschlüssel, der die allgemeine Klassifikation des Objektes beschreibt, z.B. **`highway`** für Straßen oder **`amenity`** für beliebige Einrichtungen.
- Zu dem ersten Schlüssel gibt es jeweils einen vorgeschlagenen Satz von Werten. Zu **`highway`** gibt es beispielsweise die Vorschläge **`highway=service`** für Auffahrten, **`highway=secondary`** für Landstraßen oder **`highway=residental`** für Innerortstraßen.
- Weitere Klassifikationen zum Objekt können in zusätzlichen Paaren hinterlegt werden. Zu bestimmten Haupt-Paaren gibt es konkrete Vorschläge für weitere Paare. Beispielsweise kann bei **`highway`** das Paar **`maxspeed=…`** für die Höchstgeschwindigkeit angegeben werden.

Im Datenbestand von Deutschland gibt es alleine ca. 52 000 verschiedene Variationen für das Haupt-Schlüssel-Werte-Paar. Leider gibt es darunter eine große Zahl inkonsistenter oder mehrdeutiger Auszeichnungen. Als Beispiel: *Bistros* werden im Moment nicht explizit als Objekt-Klasse vorgeschlagen und zur Klasse Café beigefügt. Daher ist die häufigste Klassifikation entweder **`amenity=cafe`** oder **`shop=coffee`**. Es treten aber auch die Klassifikationen **`shop=bistro`**, **`amenity=bistro`** und **`amenity=restaurant/cuisine=bistro`** auf. Das sind insgesamt fünf verschiedene Arten für denselben Objekttyp. Hierdurch wird deutlich, dass die Klassifikation von Objekten durch einen Community-Prozess auch Nachteile hat. Hauptgründe für die unterschiedlichen Klassifikationen sind dabei:

- Es gibt keinen Konsens und die Diskussion ist noch im Gange oder bestimmte Objekttypen sind so neu, dass noch keine Diskussion begonnen wurde.
- Beitragende kennen die aktuellen Absprachen nicht (die sich leider ständig ändern) und verwenden eine eigene oder veraltete Klassifikation.
- Die Zuordnung ist ein inhaltliches Problem, da ein Objekt mehrere Eigenschaften gleichzeitig haben kann. Als Beispiel: ein Fußweg, auf dem auch Radfahren möglich ist, kann durch **`highway=cycleway/footway=yes`** oder **`highway=footway/cycleway=yes`** beschrieben werden.

Für einen eigenen Datenbestand ist diese Art der Auszeichnung daher ungeeignet. Neben der Mehrdeutigkeit gibt es weitere Nachteile:

- Eine Suche nach bestimmten Klassen ist immer eine Suche nach Zeichenketten. Es gab zwar einen Vorschlag von OSM, wiederkehrende Bezeichnungen durch Nummern zu ersetzen, das schränkte jedoch die Möglichkeiten der Auszeichnung zu sehr ein, so dass man derzeit eine textbasierte Auszeichnung vornimmt.
- Es gibt keine einfache Beziehung der Schlüssel zu Oberbegriffen, wie *Gebäude*, *Verkehr*, *Vegetation*. Für bestimmte Anwendungen müssen aber bestimmte Oberbegriffe recherchierbar sein.
- Die Klassifikation ist viel zu feingranular und erlaubt zu detaillierte Unterscheidungen, die für viele Anwendungen uninteressant sind.

Aus diesen Gründen wird für den importierten Datenbestand ein eigenes Hybris-Klassifikationsschema verwendet. Es wird permanent weiterentwickelt, indem periodisch die OSM-Klassifikationen ausgewertet und häufig vorkommende Schlüsselpaare untersucht werden. Darüber hinaus orientiert sich das eigene Schema an dem Aufbau des ATKIS-Katalogs [ATKIS]. Dieser ist für Landesvermessungsämter ein verbindliches Klassifikationsschema, für unsere Zwecke ist er aber zu undifferenziert und unterscheidet zu wenige Klassen. Wie ATKIS verwendet das eigene Klassifikationsschema aber einen einzelnen Zahlenwert, um Klassen zu definieren (Abbildung 2). Die erste Ziffer gibt eine Hauptklasse an, z.B. *organisatorisch*, *bebaute Fläche* oder *offene Fläche*. Ist für ein Objekt eine genauere Klassifikation möglich, so wird eine weitere Ziffer verwendet. Man kann bis zur maximalen Stellenzahl (derzeit fünf) weitere Stellen verwenden und erhält eine immer genauere Klassifikation. Durch die Ziffernvergabe entsteht eine einfache baumartige Zuordnung von Klassen und Unterklassen. Derzeit werden 565 Klassen unterschieden.

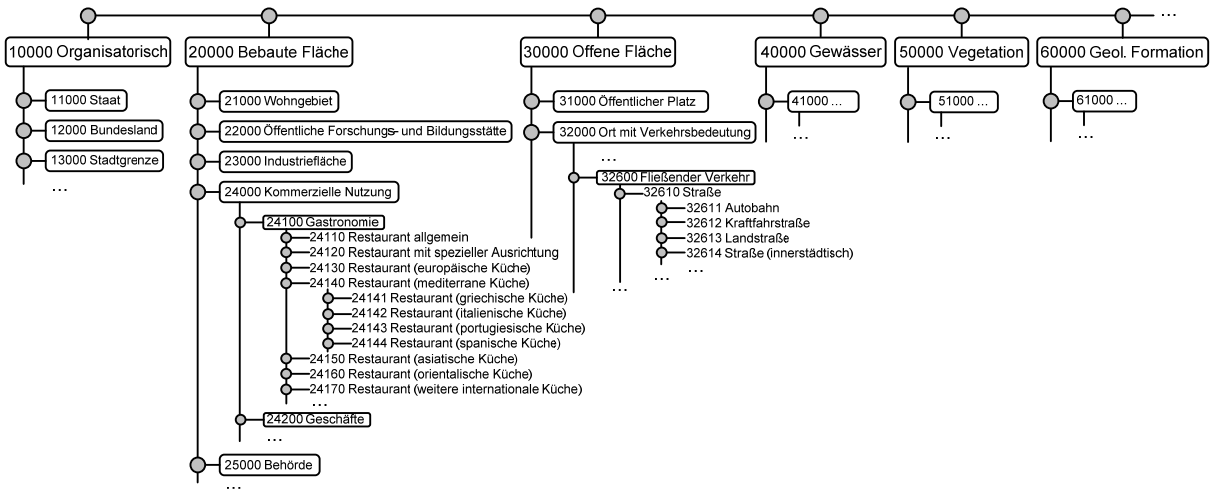


Abbildung 2: Ausschnitt des Hybris-Klassifikationsschemas

Die Klassennummer kann in eine Indexspalte der Datenbank geschrieben werden, so dass eine schnelle Suche nach Objekten von bestimmten Klassen durchgeführt werden kann. Dadurch dass sich die Aufteilung in Klassen und Unterklassen auf die Nummernkreise überträgt, kann auch effektiv nach bestimmten Teilklassen gesucht werden, was auf den ursprünglichen OSM- Klassifikationen schwierig wäre. Als Beispiel:

- Suche nach italienischen Restaurants: **`SELECT * WHERE CLASS=24142`**
- Restaurants mit mediterraner Küche: **`…WHERE CLASS BETWEEN 24140 AND 24149`**
- beliebige Restaurants: **`…WHERE CLASS BETWEEN 24110 AND 24179`**

In dem importierten Datenbestand wird die OSM-Klassifikation zwar mitgespeichert, für effiziente Suchen wird aber die Hybris-Klassifikation eingesetzt. Die OSM-Klassifikation wird erst verwendet, wenn die Objekte *nach* einer Suche aus der Datenbank geladen wurden um weitere Eigenschaften über Objekte zu erfahren, beispielsweise die Höchstgeschwindigkeit auf Straßen.

1.2 Datenübernahme aus Open Street Map

Open Street Map verwendet für den Datenexport über die Planet Files ein XML-Format. Jede Datei enthält Einzelpunkte (genannt *nodes*), Linienzüge und Flächen (genannt *ways*) sowie zusammengesetzte Objekte (genannt *relations*). Die Speicherung ist redundanzfrei; Linien enthalten nur die Referenzen auf die Einzelpunkte, wobei bei gemeinsamen Grenzen jeder Einzelpunkt nur einmal abgelegt wird. Die relations sind Zusammenfassungen von Linien, Flächen oder Punkten zu inhaltlich zusammengefassten Objekten. Grenzverläufe (bestehend aus Einzelgrenzen) oder inhaltlich zusammengefasste Wege (z.B. Radwanderwege) fallen darunter. Auch die Zusammenfassung erfolgt über die Referenzen auf die Einzelobjekte.
Die Planet-Datei *Deutschland* (Stand März 2010) beschreibt ca. 7 Mio. Geo-Objekte und bildet die Grundlage für die Statistiken in diesem Beitrag. Sie ist 8,2 GB groß, enthält 39 Mio. Einzelpunkte, 5,3 Mio. Linien und Flächen sowie ca. 65 000 Relationen.
Für eine schnelle geometrische Suche in Datenbanken ist die redundanzfreie Speicherung der Geometrien durch Einzelpunkte und Linienzüge ungeeignet. Für eine geometrische Suche müssten zum Testen jeder geometrischen Bedingung zur Laufzeit einzelne Positionen aus separaten Tabellen besorgt werden, um Flächen oder Linienzüge aufzubauen. Der Hybris-Import baut daher in sich vollständige Geometrien auf. Dies ist zwar nicht redundanzfrei, ermöglich aber eine effiziente Beschleunigung durch einen geometrischen Index. Insgesamt werden die folgenden Arbeiten ausgeführt:

- In Linienzügen, Flächen und zusammengesetzten Objekte werden die Referenzen auf Position und Flächen durch Kopien ersetzt. Als Resultat wird jedes Geo-Objekt durch eine eigene vollständige Geometrie beschrieben.
- In Open Street Map sind Flächen von Linienzügen nur dadurch unterscheidbar, dass Anfangs- und Endpunkte einer Punktfolge zusammenfallen. Anhand von OSM-Schlüsseln oder Objekttypen kann dies nicht entschieden werden. Dieser Test wird beim Import durchgeführt und Geometrien entsprechend gekennzeichnet.
- Punkt- und linienförmige Geometrien besitzen keine flächenartige Ausprägung. Viele Such-Operationen erfordern jedoch die Hinterlegung einer Fläche. Daher wird zu solchen Objekten ein zusätzliches Flächenobjekt konstruiert, indem die Geometrie durch die so genannte *Puffer*-Operation 'aufgeblasen' wird. Die Linien- oder Punktgeometrien bleiben jedoch erhalten, so dass eine Anwendung sich später die für den Anwendungsfall geeignete geometrische Repräsentation aussuchen kann.
- Open Street Map kann weder Multipolygone noch mehrfache Linien in einzelnen Linien- oder Flächenobjekten darstellen (erst durch die relations). Darüber hinaus werden ausgedehnte Linienobjekte (z.B. Autobahnen) in viele Teilstücke zerlegt. Beim Importieren wird versucht, passende Einzelobjekte wieder zu einem Gesamtobjekt zusammenzufassen. Dies ist eine sehr komplizierte Funktion, da die Zusammengehörigkeit heuristisch durch Namensgleichheit und geometrische Nähe festgestellt werden muss. Zusammengesetzte Objekte werden in einer zweiten Tabelle verwaltet, die Einzelobjekte stehen aber weiter in der Haupttabelle zur Verfügung.
- Die zeichenbasierte Klassifikationsbeschreibung wird über ein Regelwerk interpretiert und auf das eigene nummernbasierte Klassifikationsschema abgebildet.
- Zur weiteren Segmentierung der großen Datenmengen wird ein so genannter *Strukturbaum* verwendet [Roth06a]. Der Strukturbaum kann im weitesten Sinn wie ein Verzeichnisbaum eines Dateisystems betrachtet werden und erlaubt, dass man Unterbäume separat

durchsuchen oder Teile eines Baumes in eigene Datenbestände exportieren kann. Beim Import wird der Standort eines Objektes im Strukturbaum festgelegt.

- Weitere Felder (Autor, Name, Web-Link) werden aus dem Datensatz extrahiert und gespeichert. Liegt kein Objektname vor, wird ein automatisch generierter Name eingetragen.

Zur Steuerung der oben genannten Zuordnungen wurde ein Regelinterpreter entwickelt, der durch Regeln der Form

<Bedingung$_1$>, <Bedingung$_2$>, … <Bedingung$_n$> -> <Klassenzuordnung>, <Breite>, <Position im Strukturbaum>, <Namensvergabe>

gesteuert wird. Abbildung 3 zeigt einen kleinen Ausschnitt aus dem Regelwerk. Insgesamt liegen derzeit ca. 500 Regeln vor.

```
geometry==line, highway==motorway    -> ATKIS=3101, CLASS=32611, WIDTH=15,
      tree=roads.traffic, name=schema2/Autobahn                          # Autobahn
geometry==line, highway==primary     -> ATKIS=3101, CLASS=32613, WIDTH=5,
      tree=roads.traffic, name=schema2/Bundesstrasse                     # Bundesstrasse
geometry==line, highway==cycleway    -> ATKIS=3102, CLASS=32616, WIDTH=1,
      tree=roads.traffic, name=schema2/Fahrradweg                        # Fahrradweg
geometry==line, cycleway==lane       -> ATKIS=3102, CLASS=32616, WIDTH=1,
      tree=cycleway.traffic, name=schema2/Radwegspur                     # Radfahrspur
geometry==line, cycleway==track      ->
      ATKIS=3102, CLASS=32616, tree=cycleway.traffic, name=schema2/Radwegspur
geometry==area/point, amenity==parking  -> ATKIS=3103, CLASS=32110,
      tree=urban.org, name=schema1
...

geometry==area/point, amenity==restaurant,cuisine==regional       ->          # Verschiedene
      ATKIS=2315, CLASS=24111, tree=urban.org, name=schema2/Restaurant       # Restaurants
geometry==area/point, amenity==restaurant,cuisine==national       ->
      ATKIS=2315, CLASS=24112, tree=urban.org, name=schema2/Restaurant
geometry==area/point, amenity==restaurant,cuisine==international  ->
      ATKIS=2315, CLASS=24113, tree=urban.org, name=schema2/Restaurant
geometry==area/point, amenity==restaurant,cuisine==pizza          ->
      ATKIS=2315, CLASS=24121, tree=urban.org, name=schema2/Pizzaria
geometry==area/point, amenity==restaurant,cuisine==fish           ->
      ATKIS=2315, CLASS=24122, tree=urban.org, name=schema2/Restaurant
geometry==area/point, amenity==restaurant,cuisine==seafood        ->
      ATKIS=2315, CLASS=24122, tree=urban.org, name=schema2/Restaurant
geometry==area/point, amenity==restaurant,cuisine==vegetarian     ->
      ATKIS=2315, CLASS=24123, tree=urban.org, name=schema2/Restaurant
geometry==area/point, amenity==restaurant,cuisine==steak          ->
      ATKIS=2315, CLASS=24124, tree=urban.org, name=schema2/Steakhaus
geometry==area/point, amenity==restaurant,cuisine==steak_house    ->
      ATKIS=2315, CLASS=24124, tree=urban.org, name=schema2/Steakhaus
...
```

Abbildung 3: Ausschnitt des Regelwerks zum Steuern des Imports

Bedingungen enthalten Aussagen über die geometrische Ausprägung (**`area`** für Fläche, **`line`** für Linienobjekt, **`point`** für Einzelpunkt) oder Vorkommen von OSM-Klassifikationen. Treffen alle Bedingungen zu, werden die geforderten Einträge für die ATKIS- und Hybris-Klasse sowie die Position im Strukturbaum im Objekt vorgenommen. Für Linienobjekte, bei denen keine Breiteninformation in der Geometrie abgelegt ist, wird eine Breite in Metern definiert. Schließlich wird, wenn das Objekt noch keinen Namen hat, eine automatische Benennung gemäß verschiedener Schemata vorgenommen. Das Schema **`schema2`** definiert beispielweise, dass der Name aus der angegeben Zeichenkette sowie der eindeutigen Objektnummer aufgebaut wird. So entsteht beispielsweise der Name "**`Fahrradweg_123456`**", falls kein anderer Name definiert wurde.

1.3 Die Datenbank-Struktur

Die aus dem OSM-Datenbestand extrahierten Geo-Objekte werden in einer relationalen Datenbank gespeichert. Bezüglich der Speicherung wurden folgende Entscheidungen getroffen:

- Die Datenbank ist eine relationale Standard-Datenbank *ohne* räumliche Erweiterung. Gründe dafür waren: 1. Es sollen beliebige Server-Systeme einsetzbar sein, insb. auch Server, die selbst auf mobilen Endgeräten laufen (z.B. eine SQLite-Datenbank). Hierdurch soll es möglich sein, Teile der Geo-Datenbestände auf ein mobiles Endgerät auszulagern. 2. Es sollen beliebige Client-Umgebungen für Standard-SQL unterstützt werden (z.B. basierend auf ODBC oder JDBC), ohne dass man auf zueinander inkompatible räumliche SQL-Erweiterungen angewiesen ist. Darüber hinaus gibt es verschiedene Client-Plattformen, für die es überhaupt keine Schnittstellen für räumliche Datenbanken gibt.
- Pro Geo-Objekt soll nur eine einzige Datenzeile gespeichert werden. Insbesondere werden die Geometrien nicht weiter in Linien und Punkte zerlegt und auf weitere Tabellen verteilt abgelegt. Geometrien werden im Standard-Format *Well-Known Binary* abgelegt, das durch das *Open Geospatial Consortium* (*OGC*) spezifiziert wurde [Herr05].
- Informationen zur räumlichen Indizierung werden auch in der betreffenden Datenzeile eines Geo-Objektes gespeichert. Es sollen keine weiteren Strukturen notwendig sein, um effizient eine geometrische Suche durchzuführen.

Der letzte Punkt erfordert einen neuartigen Indizierungsmechanismus, wie er in [Roth09b] vorgestellt wurde. Er wird in Abschnitt 1.4 skizziert. Es ergibt sich eine Tabellenstruktur wie in Tabelle 1 dargestellt.

Tabelle 1: Spalten der Geodatenbank

Spalte	Beschreibung	Oberbegriff
Name	Name des Objektes (nicht eindeutig), z.B. der Straßenname im Ort	thematisch
Hybris-Klasse	Klassifikationsnummer (Abschnitt 1.1)	
ATKIS-Klasse	Klassifikationsnummer nach ATKIS-Katalog	
OSM-Klassifikation	Beschreibungen nach OSM als Zeichenkette	
Struktur-Beschreibung	Position im Strukturbaum	strukturell
Polygon-Geometrie	OGC-Geometrie *Multipolygon* als Well-Known Binary. Diese Geometrie wird *immer* gespeichert.	geometrisch
Linien-Geometrie	OGC-Geometrie *MultiLineString* als Well-Known Binary – nur bei linienförmigen Objekten	
Punkt-Geometrie	OGC-Geometrie *Point* als Well-Known Binary – nur bei punktförmigen Objekten	
Puffer-Größe	Ausdehnung einer punktförmigen Geometrie oder Breite einer linienförmigen Geometrie	
Flächeninhalt	Vorberechneter Flächeninhalt der Polygon-Geometrie	
Autor	Ersteller des Datenobjektes	organisatorisch
ImportID	ID, die von der Import-Quelle vergeben wird und während der Lebenszeit des Objektes erhalten bleiben muss	
Import-Modifikation	Wurde der Datensatz beim letzten Import 1. neu angelegt, 2. unverändert übernommen oder 3. verändert?	
Änderungs-Flag	Wurde das Objekt *nach* dem letzten Import in der lokalen Datenbank verändert?	
Web-Link	URL zu weiteren Informationen zum Objekt	Metadaten
Split Index	Indexnummer nach dem Extended-Split-Index-Verfahren	Räumlicher Index
Shifted Split Index	Nummer des verschobenen Indexes	
Minimal Bounding Rect1	*Zwei* Rechtecke, die in Summe die Geometrie umschließen	
Minimal Bounding Rect2		

In dieser Tabelle befinden sich keine topologischen Eigenschaften. Diese werden bei Bedarf, z.B. bei der Routenplanung in eigenen Tabellen abgelegt. Einige der dargestellten Eigenschaften dienen *organisatorischen* und *strukturellen* Eigenschaften eines Objektes [Roth06b]. So gibt es Spalten die einzig für den Import und Re-Import der Daten zuständig sind und beispielsweise erlauben, dass bei einem Import die Veränderungen in der Datenquelle erkannt werden und potentielle Import-Konflikte behandelt werden können.
Für die Speicherung der geometrischen Eigenschaften stehen fünf Spalten zur Verfügung. Der Flächeninhalt wird vorberechnet in einer Spalte abgelegt. Damit kann man bei der Suche den Flächeninhalt einschränken, ohne dass dieser zur Laufzeit berechnet werden muss. Da Polygon-, Linien- und Punkt-Geometrien als Well-Known Binary gespeichert sind, können sie auf einfache Weise weiterverarbeitet werden. Innerhalb einer SQL-Anweisung kann nicht auf geometrische Interna einer Geometrie (z.B. auf die Einzelpunkte eines Polygons) zugegriffen werden. Geometrische Anfragen müssen daher immer die geometrischen Index-Spalten verwenden. Wurde die Geometrie erst einmal durch eine Suche aus der Datenbank deserialisiert, kann sie mit verschiedenen OGC-konformen Werkzeugen (z.B. JTS [Aq03]) weiterverarbeitet werden.

1.4 Schneller Zugriff über den Extended Split Index

Beim Zugriff auf Geo-Objekte ist neben der Objektklasse oder dem Namen eine geometrische Einschränkung die wichtigste Bedingung. Typische Abfragen laden beispielsweise alle Geo-Objekte in einem vordefinierten Rechteck für eine Kartendarstellung oder laden Objekte eines bestimmten Typs im Umkreis um den aktuellen Standort. Damit Indexinformationen effizient in der Datenzeile des Objektes gespeichert werden können, wird das Extended-Split-Index-Verfahren eingesetzt [Bey09, Roth09b]. Geometrische Abfragen verlaufen in zwei Stufen:

- Jeder Geometrie werden zwei Nummern zugeordnet (*Split Index* und *Shifted Split Index*). Diese definieren die Lage *und* Größe der Geometrie durch eine rechteckige Annäherung. Man verwendet zwei nach einem bestimmten Verfahren zueinander verschobene Index-Nummern, um bestimmte Verschnittprobleme mit einer einzelnen Nummer zu kompensieren [Ker10]. Im Gegensatz zum Minimal Bounding Rect (MBR) können diese zwei Nummern unabhängig voneinander auf eindimensionale Indizes der Datenbank abgebildet werden und erlauben bei einer Anfrage eine sehr effiziente geometrische Suche.
- Durch die erste Stufe kann schon eine Kandidatenmenge berechnet werden. Diese kann aber bei einem exakten geometrischen Vergleich noch zu groß sein. Damit aber nicht alle Kandidaten aus der Datenbank heraus transportiert werden müssen, erfolgt in einer zweiten Stufe eine weitere Einschränkung. Diese Stufe verwendet eine Annäherung der Geometrie durch *zwei* Minimal Bounding Rects (MBR). Zwei Rechtecke erlauben eine wesentlich günstigere Annäherung von z.B. L-förmigen Objekten als ein einzelnes. Für die Berechnung von zwei minimalen Rechtecken, die in Summe eine Geometrie umschließen, existiert ein Verfahren, das bei n Geometriepunkten in $O(n \cdot \log n)$ Schritten läuft. Bei typischen Geometrien reduziert sich die abgedeckte Fläche bei zwei Rechtecken im Vergleich zu einem MBR um Faktor 1,4 bei Polygonen bzw. 2,1 bei Linienobjekten. Die Darstellung durch zwei statt einem umgebenden Rechteck bringt daher eine deutlich verbesserte Annäherung der Geometrie bei geringen Kosten [Roth10].

Jede geometrische Bedingung kann so auf nicht-geometrische Spalten-Bedingungen abgebildet werden, die durch Standard-SQL ausdrückbar sind und eine Beschleunigung durch Standard-Index-Spalten erlauben. Zusätzlich können weitere nicht-geometrische Bedingungen angehängt werden. Der geometrische Anteil der Bedingungen ist eine Kette von **`INDEX BETWEEN … OR INDEX BETWEEN … OR …`**. Diese Kette wird von der anfragenden Anwendung durch eine kleine Hilfsfunktion berechnet. Implementierungen dieser Hilfsfunktion existieren für diverse Software-Plattformen.

Die beschriebenen Indexierungsmaßnahmen führen zu einer Trefferquote von 97% bei durchschnittlichen Abfragen, d.h. nur 3% der Kandidaten sind *false positive* und entsprechen nicht einer exakten geometrischen Bedingung [Ker10]. Bei typischen Anwendungen ist diese Quote hinreichend groß, da überflüssige Resultate in der Regel durch weitere Verarbeitungsschritte verworfen werden. Müssen geometrische Bedingungen aber exakt überprüft werden, kann die Anwendung dies explizit tun. Hierzu existieren effiziente Bibliotheken für zahlreiche Plattformen, die das Well-Known-Binary-Format interpretieren können.

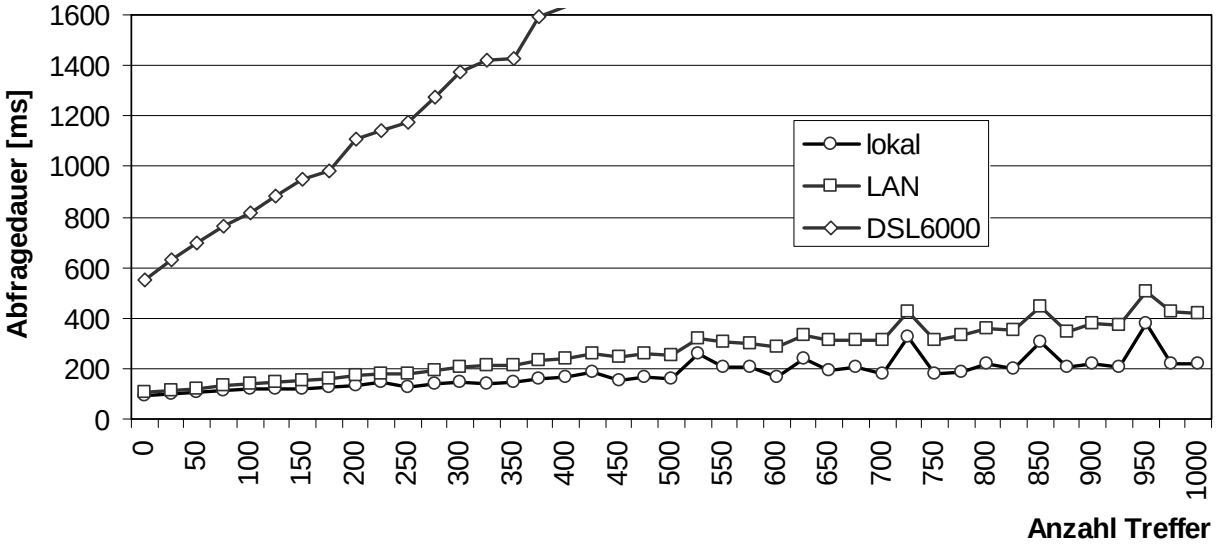


Abbildung 4: Zugriffszeiten auf den Geodatenbestand

Abbildung 4 zeigt das Laufzeitverhalten bei typischen Anfragen. Hierzu wurden ca. 37 000 Anfragen durch das Werkzeug zur Generierung von Kartenkacheln (siehe unten) gemacht. Diese laden Objekte, die mit vorgegebenen rechteckigen Flächen in unterschiedlichen Orientierungen und Größen überlappen. Die Anfragen wurden lokal, im LAN über GBit-Ethernet sowie von einem Arbeitsplatz, der über DSL (6 MBit) angebunden ist, durchgeführt. Der Server verfügte über zwei Quad-Prozessoren (XEON L5506) mit 2,13 GHz (Windows Server 2008).
Im Durchschnitt wurden 59,1 Geo-Objekte pro Anfrage zurückgegeben. Die durchschnittliche Dauer für eine Anfrage betrug 100,6 ms (lokal) 110,9 ms (LAN) 669,3 ms (DSL6000). Die Zeiten entsprechen typischen nicht-geometrischen Abfragen und hängen im Wesentlichen von der Bandbreite der Netzwerkanbindung und von der Menge der zurück gelieferten Objekte ab. Für typische ortsbezogene Dienste, die Anfragen mit einer überschaubaren Resultatmenge durchführen, sind die Zeiten durchweg akzeptabel.

1.5 Re-Import und Authoring des Datenbestandes

Der Open-Street-Map-Datenbestand ändert sich ständig. Die häufigsten Änderungen betreffen neue Geo-Objekte. Bei existierenden Objekten werden oft die Geometrien korrigiert, aber auch Klassifizierungsmerkmale verändert.
Durch die Übernahme in eine lokale Datenbank kann man eigene Projekte zwar von den permanenten Änderungen entkoppeln, periodisch sollte der eigene Datenbestand aber aktualisiert werden. Durch die Entkopplung kann man auf stabile Stände warten (insbesondere bezüglich der Klassifizierungen) und eine Erneuerung sorgfältig planen.
Durch verschiedene Spalten wird ein Re-Import gesteuert. Durch eine eindeutige Identifikationsnummer von OSM kann ein Objekt auch in der eigenen Datenbank identifiziert werden (Spalte *ImportID*, Tabelle 1). Damit wird bei einem Re-Import festgestellt, ob ein Objekt neu hinzugekommen ist oder nur aktualisiert wurde. Darüber hinaus wird festgehalten, ob ein

Objekt in der Zwischenzeit lokal verändert wurde (Spalte *Änderungs-Flag*, Tabelle 1). Schließlich kann nach jedem Import nachvollzogen werden, ob ein Datensatz beim letzten Import neu angelegt, unverändert übernommen oder verändert wurde (Spalte *Import-Modifikation*, Tabelle 1).

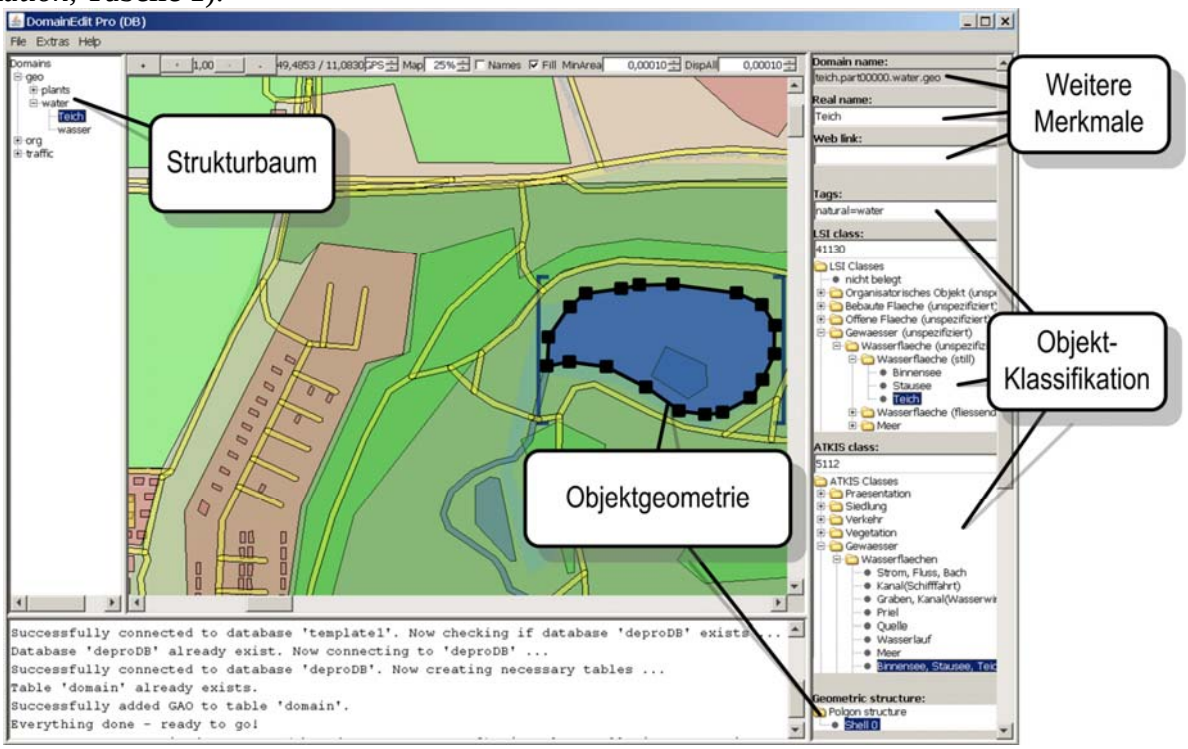


Abbildung 5: Die depro-Authoring-Umgebung

Für die Modifikation der lokalen Datenbank steht die Authoring-Umgebung *depro* zur Verfügung (Abbildung 5, [Roth08]), mit der komfortabel die Geometrien oder Klassifikationen geändert werden können. Über ein Plug-In-Konzept können Entwickler eigene Funktionen zur Modifikation des Datenbestandes integrieren. Zurzeit existieren schon Plug-Ins zur statistischen Auswertung des gesamten Datenbestandes nach geometrischen und inhaltlichen Gesichtspunkten, zum Refactoring des Strukturbaums sowie zum Abspalten von Datenbeständen aus dem Gesamtdatenbestand. Das letztgenannte Plug-In kann z.B. verwendet werden um kleinere Datenbanken zu Testzwecken zu generieren oder extern zu installieren. Darüber hinaus kann durch Import-Plug-Ins der Datenbestand durch andere Quellen erweitert werden.

2 Aufbauende Arbeiten

Auf der Basis des Datenbestandes von der beschriebenen Datenübernahme aus Open Street Map, sind verschiedene Arbeiten durchgeführt wurden. Viele Anwendungen sind während der Lehrveranstaltung "Ortsbezogene Anwendungen und Dienste" im Wintersemester 2009/2010 an der Ohm-Hochschule Nürnberg entstanden. Im Rahmen dieser 4-stündigen Vorlesung sollte in einer Projektarbeit der Umgang mit Geodaten gelernt werden. Hierzu wurden folgende Aufgaben vergeben:

- Fußgänger-Navigation im Stadtbereich,
- Malen von Karten mit unterschiedlichen Vergrößerungsstufen,
- Automatische Generierung von JPG-Beschreibungen zu georeferenzierten Fotos,
- Automatische Generierung von Wegbeschreibung aus GPS-Logs.

Einige Resultate sind in Abbildung 6 illustriert.

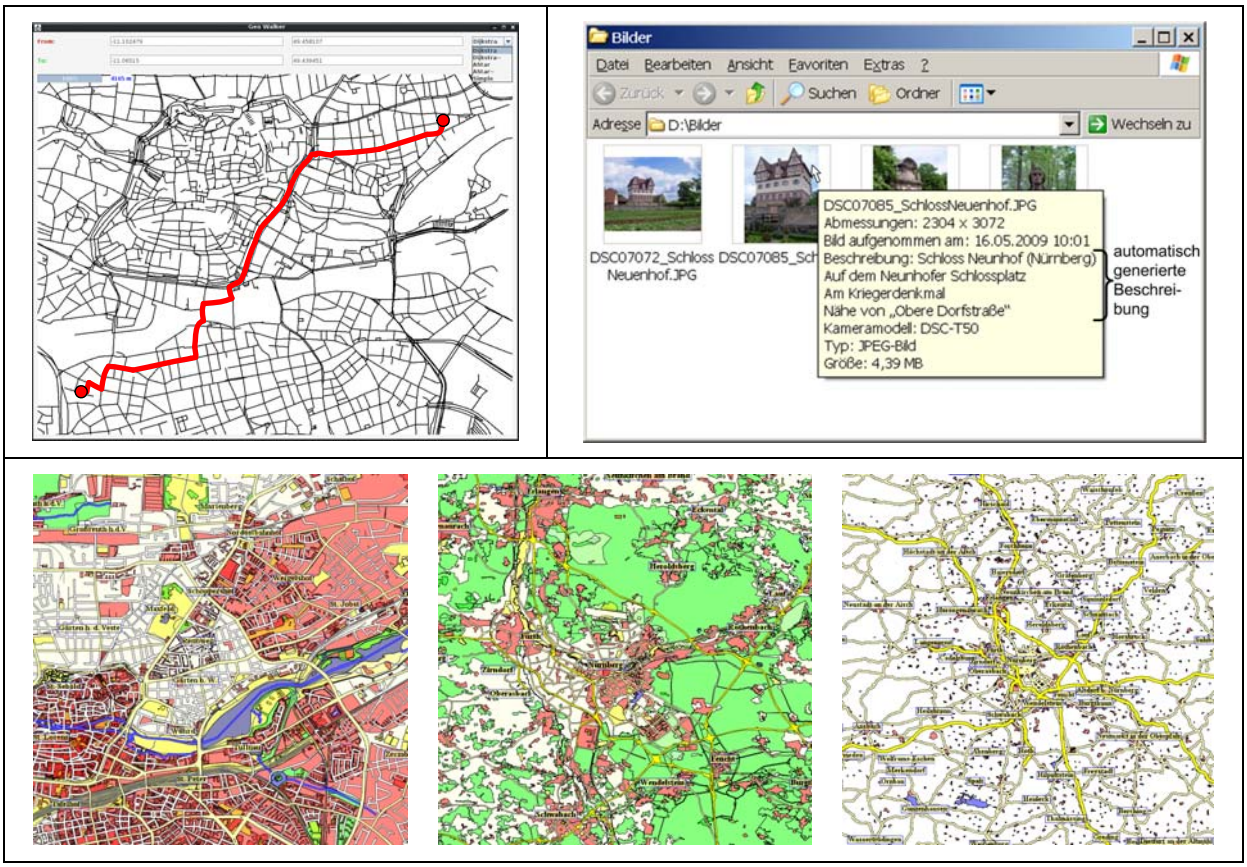


Abbildung 6: Beispiele für ortsbezogene Anwendungen und Plattformdienste: Navigation (oben links), automatische Beschreibungsgenerierung von Fotos (oben rechts), Generierung von Karten (unten)

Zurzeit werden weitere Plattformdienste entwickelt. Solche Dienste kann man als Baukasten verwenden und damit die weitere Entwicklung verkürzen. Als Plattformdienste sind bisher eine in weiten Grenzen konfigurierbare Routenplanung, eine Rendering-Engine für Kartenkacheln sowie eine Reverse-Geocoding-Plattform entstanden.
Insbesondere die kurze Einarbeitungszeit im Rahmen der Lehrveranstaltung demonstrierte, dass der Geodatenbestand in der aktuellen Form sehr gut geeignet ist, relativ schnell eigene ortsbezogene Dienste zu entwickeln. Von den Studenten wurde darüber hinaus begrüßt, dass der Zugriff auf den zentralen Geodatenbestand von vielen Betriebssystemplattformen aus stabil funktionierte. Auch die Entwicklung auf einem Heimarbeitsplatz über DSL war dabei möglich.
Um den Komfort weiter zu erhöhen und damit die Einarbeitungszeit weiter zu verkürzen existiert für die Programmiersprache Java eine Einblendung der Geo-Objekte aus der Datenbank in den Objektraum der Anwendung [Roth09a]. Für die direkte Verarbeitung von Objektgeometrien im Well-Known-Binary-Format können darüber hinaus Rahmenwerke wie beispielsweise Hibernate Spatial [Hib] eingesetzt werden.

3 Fazit

Durch die Geodaten-Bestände von Open Street Map stehen viele Möglichkeiten offen, ortsbezogenen Anwendungen relativ kostengünstig zu realisieren. Diverse Festlegungen unterliegen einem Community-Prozess. Insbesondere die Freiheiten, Geo-Objekte zu klassifizieren, schränken jedoch die weitere automatische Auswertung durch die Anwendung stark ein.
In diesem Beitrag ist ein Ansatz dargestellt worden, den Open-Street-Map-Datenbestand in eine eigene Datenbank zu übernehmen. Hierbei werden geometrische und inhaltliche Eigenschaften sowie die Klassifikation auf ein eigenes Schema abgebildet. Möglichkeiten der Weiterverarbeitung und Fusionierung mit weiteren Datenbeständen stehen zur Verfügung.

Schließlich wird ein effizienter Abfragemechanismus zur Verfügung gestellt, der auf Standard-SQL basiert, daher für viele Client- und Server-Plattformen einsetzbar ist. Die Effektivität des Ansatzes wurde durch verschiedene Studentenprojekte demonstriert.

Referenzen

[Aq03] Aquino, J.: JTS Topology Suite, Technical Specifications, Vivid Solutions, 2003

[ATKIS] Arbeitsgemeinschaft der Vermessungsverwaltungen der Länder der Bundesrepublik Deutschland (AdV): ATKIS Homepage, http://www.atkis.de/

[Bey09] Beyer, M.: Konzeption und Realisierung eines Geospatial Add-Ons für SQL, Diplomarbeit, Ohm-Hochschule Nürnberg, März 2009

[Fra97] Frank, A.: Spatial Ontology: A Geographical Information Point of View, Spatial and Temporal Reasoning, Part II, Springer, 135-153

[Herr05] Herring, J. (ed.): OpenGIS Implementation Specification for Geographic information – Simple feature access - Part 1: Common architecture, OGC, 2005

[Herr06] Herring, J. (ed.): OpenGIS Implementation Specification for Geographic information – Simple feature access - Part 2: SQL option, OGC, 2006

[Hib] Hibernate Spatial Tutorial, http://www.hibernatespatial.org

[Ker10] Kerschner, M.: Effizienzsteigerung des räumlichen Extended Split Index, Bachelorarbeit, Ohm-Hochschule Nürnberg, April 2010

[OSM] Open Street Map, http://www.openstreetmap.org/

[OSMMF] OSM Map Features, http://wiki.openstreetmap.org/wiki/Map_Features

[OSMPLA] OpenStreetMap excerpts for Europe, http://download.geofabrik.de/osm/

[Roth06a] Roth, J.: Detecting Identifiable Areas in Mobile Environments, The 21st Annual ACM Symposium on Applied Computing, 23.-27. April 2006, Dijon, (Frankreich), ACM Press, 986-991

[Roth06b] Roth, J.: Modelling Geo Data for Location-based Services, 3. GI/ITG KuVS Fachgespräch "Ortsbezogene Anwendungen und Dienste", 7.-8. Sept. 2006, Berlin, Institut für Informatik der Freien Universität Berlin

[Roth08] Roth, J.: Managing Geo Data for Location-based Services – The Hybris Framework
4th Annual Meeting on Information Technology & Computer Science (ITCS 2008), Stuttgart, 20. Feb. 2008

[Roth09a] Roth, J.: *Verwaltung geographischer Daten mit Hilfe eines Add-ons für Standard-Datenbanken,* Verwaltung, Analyse und Bereitstellung kontextbasierter Informationen, GI Informatik 2009, Lübeck, 29.9.2009, 2041-2055

[Roth09b] Roth, J.: The Extended Split Index to Efficiently Store and Retrieve Spatial Data With Standard Databases, IADIS International Conference Applied Computing 2009, Rom (Italien), 19.-21. Nov. 2009, Vol. I

[Roth10] Roth, J.: The Approximation of Two-Dimensional Spatial Objects by Two Bounding Rectangles, Spatial Cognition and Computation: An Interdisciplinary Journal, akzeptierter Beitrag