TECHNISCHE UNIVERSITÄT DRESDEN

Fakultät Informatik Institut für Systemarchitektur, Professur Rechnernetze

Diplomarbeit

EIN BEWERTUNGSSYSTEM FÜR OPENSTREETMAP

Christoph Wagner
Mat.-Nr.: 3120405

Betreut durch:
Dr.-Ing. Stephan Groß
Dr.-Ing. Sandra Steinbrecher
Verantwortlicher Hochschullehrer:
Prof. Dr. rer. nat. habil. Dr. h. c. Alexander Schill

Eingereicht am 31. Dezember 2010

SELBSTÄNDIGKEITSERKLÄRUNG

Ich versichere, dass ich die vorliegende Diplomarbeit ohne fremde Hilfe selbständig verfasst und nur die angegebenen Quellen und Hilfsmittel benutzt habe. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich gemacht. Die Arbeit wurde bisher in gleicher oder ähnlicher Form weder veröffentlicht, noch einer anderen Prüfungsbehörde vorgelegt.

.....
gez. Christoph Wagner Dresden, 31. Dezember 2010

INHALTSVERZEICHNIS

ZUSAMMENFASSUNG

Um die Vertrauenswürdigkeit der Daten des freien Kartierungs-Projektes Openstreetmap besser einschätzen zu können, soll ein System entworfen werden, bei dem die Beteiligten in der Lage sind, ihr Vertrauen in die Richtigkeit einzelner Datensätze für andere nachvollziehbar ausdrücken zu können. Dabei soll neben der Vertrauenswürdigkeit der Datensätze auch die Vertrauenswürdigkeit der Bewerter betrachtet werden.
Um die Zugehörigkeit von Vertrauensaussagen zu Bewertern von unterschiedlicher Vertrauenswürdigkeit zu sichern und die Vertrauensaussage selbst zu beurteilen, werden die Wertungen mithilfe von digitalen Signaturen implementiert.

1 MOTIVATION

Dank der zunehmenden Vernetzung und vielfältigen Kommunikationsmöglichkeiten durch das Internet gewinnen Web-2.0-Projekte zunehmend an Bedeutung und Attraktivität. Im Vordergrund steht das kollaborative Zusammenwirken vieler Menschen an einem Projekt mit einem gemeinsamen Ziel. Das bekannteste Beispiel eines erfolgreichen Projektes dieser Art ist wahrscheinlich Wikipedia[1]. Nutzer können hierbei Artikel nahezu beliebig erstellen, editieren und weiterentwickeln um eine große Enzyklopädie zu erschaffen.
Ein weiteres, noch nicht so weit verbreitetes Projekt, welches sich wie Wikipedia mit der Sammlung von freiem Wissen beschäftigt, ist Openstreetmap[2]. Das Openstreetmap-Projekt hat sich auf die Sammlung von Geodaten spezialisiert und möchte mithilfe vieler mitwirkender Nutzer, den sogenannten Mappern, eine freie Weltkarte erstellen. Das heißt, sowohl die Geodaten als auch die daraus erzeugten Karten sollen für jedermann kostenlos zugänglich und verwendbar sein. Geodaten sind Daten mit geographischem Bezugspunkt, angegeben durch Koordinaten auf der Erde. Grundlage des Openstreetmap-Projektes ist eine zentrale Geodatenbank, welche von den Mappern beliebig erweitert und bearbeitet werden kann. Das Sammeln der Geoinformationen erfolgt meist vor Ort mit einem GPS-Gerät und weiteren zusätzlichen Geräten, wie Digitalkameras, Audiorecordern etc.
Des weiteren werden Luftbilder und geographisches Wissen von Anwohnern bei der Erfassung der Daten genutzt.

Die gesammelten Rohdaten werden vektorisiert, also durch graphische Primitive wie Punkte und Linien dargestellt, und mit zusätzlichen Informationen, den sogenannten Tags, angereichert. Bei der Erfassung einer Straße beispielsweise können somit neben dem Straßenverlauf auch Informationen wie Straßenklassifizierung, Straßenname, Oberflächenbeschaffenheit, Beleuchtung und vieles mehr erfasst werden.
Die Mitglieder von Openstreetmap haben sich im Laufe der Zeit darauf geeinigt, welche Tags für welche realen Objekte verwendet werden sollen. Ein großer Vorteil von Openstreetmap ist, dass diese Liste von Tags beliebig erweitert und verändert werden kann, wodurch Openstreetmap in der Lage ist, auf wechselnde Anforderungen schnell zu reagieren. Dies wurde eindrucksvoll deutlich als im Januar 2010 in Haiti ein starkes Erdbeben die Hauptstadt Port-au-Prince erschüttert hatte und mehrere engagierte Mapper neue Tags für Erdbebenschäden entworfen haben[3] um damit eine Katastrophenkarte der Krisenregion zu erstellen.

1. Wikipedia: http://wikipedia.org/
2. Openstreetmap: http://www.openstreetmap.org/
3. Heise:
http://www.heise.de/newsticker/meldung/OpenStreetMap-Projekt-als-Katastrophenhelfer-908472.html,
20.01.2010

Openstreetmap kann mit einem solchen freien Taggingschema prinzipiell alle Bereiche von Geodaten abdecken und in viele verschiedene Richtungen wachsen, sofern genügend Interesse der Mapper daran besteht.
Ein Nachteil des freien Taggingschemas sind die teilweise sehr langen und manchmal im Sande verlaufenden Diskussionen über die beste Art und Weise Informationen an den Objekten mit Tags unterzubringen. Es treten gelegentlich inkompatible oder rivalisierende Tags auf. Mit der Zeit stellt sich allerdings meist eine Variante als die beliebteste heraus, während die anderen verschwinden.

Die durch das flexible Taggingschema und die beliebige Editierbarkeit der Daten gewonnene Freiheit wird leider auch oftmals kritisch betrachtet, wenn es darum geht wie verlässlich die Openstreetmap-Daten eigentlich sind. So argumentieren beispielsweise Ämter oder Firmen, die um eine Datenspende für Openstreetmap gebeten werden, dass sie die Qualität ihrer Daten nicht mehr zusichern könnten, wenn diese in einer freien Datenbank von jedem bearbeitet und somit eventuell auch verschlechtert werden könnten. Als Spender möchten sie allerdings oft auch auf ihre Namensnennung bei der Nutzung der Daten nicht verzichten. Datenverschlechterungen nach dem Import sollen andererseits nicht auf die Spender zurückfallen.
Im jetzigen System von Openstreetmap können von der Openstreetmap-Community keinerlei Zusagen dieser Art gemacht werden. In der Folge kommt eine Zusammenarbeit von Openstreetmap und Ämtern bzw. Firmen oft nicht zustande.

Weiterhin gibt es in Openstreetmap Themenbereiche, bei denen von Natur aus wenig Flexibilität nötig ist und die eingetragenen Daten im Normalfall nicht verändert werden müssen. Ein Beispiel hierfür sind politische Grenzen, die aus öffentlichen Quellen bezogen wurden, die amtlich genau sind. Einmal korrekt in die Datenbank eingetragen, gibt es in der Regel keinen Grund diese Grenze zu verändern. Da sich politische Grenzen in den meisten Regionen der Welt eher selten verschieben, sollte die Grenze in der Datenbank dies auch nicht tun. Die Gefahr, dass Nutzer aus Versehen diese Grenze editieren oder andere Fehler auftreten, ist verhältnismäßig hoch. Aus diesem Grunde wurde innerhalb der Community schon häufig diskutiert, bestimmte Objekte für das Editieren zu sperren. Dies widerspräche allerdings den Prinzipien von Openstreetmap. Zum einen müsste es privilegierte Personen geben, die Objekte sperren und im Ernstfall wieder entsperren können, und zum anderen müssten diese Personen erst einmal absichern können, dass ein Objekt korrekt in die Datenbank eingetragen wurde, wenn sie es sperren.

Ein anderer Ansatz besteht darin, alle Dateneingaben - auch die fehlerhaften - zuzulassen und sie im Nachhinein zu überprüfen. Es gibt bereits verschiedene Programme, die die Openstreetmap-Datenbank regelmäßig automatisch auf Fehler überprüfen.[4] Dabei wird nach bekannten Fehlermustern - wie beispielsweise doppelten Knoten oder unverbundenen Wegen - gesucht, die beim Mappen auftreten können. Die gefundenen Fehler werden anschließend meistens von Mappern kontrolliert und korrigiert; selten und nur in eindeutigen Fällen geschieht dies automatisiert.
Leider stoßen automatische Verfahren an ihre Grenzen, wenn es darum geht, die Openstreetmap-Daten inhaltlich und nicht nur strukturell zu überprüfen. Ein automatisches Verfahren wird beispielsweise nicht feststellen können, ob eine eingetragene Straße tatsächlich existiert oder ob Eigenschaften wie Höchstgeschwindigkeit und Fahrbahnbreite zutreffend gesetzt sind. Das liegt daran, dass automatische Programme bisher keinen Abgleich mit der Wirklichkeit vornehmen können. Diese Aufgabe bleibt (noch) den Mappern überlassen.

Eine Mitwirkung von Menschen ist also immer dann erforderlich, wenn Daten inhaltlich kontrolliert werden müssen.
Vor dem gleichen Problem stehen auch kommerzielle Anbieter von Geodaten wie Navteq oder Teleatlas. Die Mitwirkung der Nutzer besteht dort jedoch lediglich im Melden von Problemen. Teleatlas stellt dafür einen Onlinedienst namens Map Insight[5] zur Verfügung. Die Meldungen sind laut Teleatlas eine von vielen Quellen, die für die Aktualisierung der Daten verwendet werden. Das eigentliche Editieren der Daten ist aber wenigen privilegierten Personen - den Teleatlas-Mitarbeitern - überlassen, die aus Unternehmenssicht als vertrauenswürdig gelten können. Auch das Unternehmen Navteq bietet einen Onlinedienst zur Datenkorrektur an, der sehr ähnlich wie

4. Openstreetmap Qualitätssicherung, http://wiki.openstreetmap.org/wiki/Qualitätssicherung
5. Teleatlas Map Insight, http://mapinsight.teleatlas.com/

der von Teleatlas arbeitet. Der NAVTEQ Map Reporter[6] kann jedoch gezielter genutzt werden, indem er die genaue Auswahl eines Datenobjektes zur Fehlermeldung zulässt. In der Navteqhilfe ist zu lesen: *„Wir haben eine Technologie entwickelt, mit deren Hilfe eine Autokorrektur einiger Map Reports möglich ist. Andere Aktualisierungsvorschläge müssen durch unser Field-Team überprüft und verifiziert werden."*
Es ist durchaus denkbar, dass hier automatische Plausibilitätstests angewandt werden, um nicht zu viel Aufwand für die angestellten Dateneditoren zu verursachen. Da von dem Melder eines Fehlers außer der freiwilligen Angabe einer E-Mailadresse keine persönlichen Daten verlangt werden, fließen offenbar keine Informationen über den Melder in die automatischen Prüfungen mit ein.

Auch bei Openstreetmap können Nutzer der Daten Verbesserungsvorschläge machen und Kartenfehler melden. Mit dem Onlinedienst Openstreetbugs[7] lassen sich Marker auf einer Openstreetmap-Karte setzen und mit einem Kommentar versehen. Die Mapper des Openstreetmap-Projektes können sich diese Fehler ansehen, kommentieren und die Daten entsprechend editieren. Der wesentliche Unterschied zu Teleatlas und Navteq ist, dass die Mapper keine privilegierten Personen sind. Jeder kann die Daten beliebig editieren; jeder kann Mapper sein. Weiterhin hat Openstreetmap eine viel höhere Transparenz, da durch das Speichern und Veröffentlichen einer Versionsgeschichte von Objekten jeder nachvollziehen kann, wer wann wo was wie editiert hat.
Ein Problem dabei ist es, die Masse an Änderungen zu überblicken und systematisch inhaltlich zu überprüfen. Mit Diensten wie Openstreetbugs ist das zur Zeit noch nicht möglich. Einer systematischen Überprüfung der Daten stehen folgende Probleme im Weg:

- Bisher sind nur negative Bewertung möglich: Wird bei Openstreetbugs ein Fehler entdeckt, kann er gemeldet werden. Bei fehlenden Bugs in einem Gebiet ist es unmöglich festzustellen, ob das Gebiet bereits auf Fehler kontrolliert wurde und schlicht kein Handlungsbedarf besteht oder ob sich bisher niemand weiter für das Gebiet interessiert hat und somit noch nie nach Fehlern gesucht wurde.
- Es ist keine zeitliche Kontrolle möglich: Die Fehlersuche und Korrektur wird immer nur an der aktuellen Version durchgeführt. Wenn in der Zukunft erneut Fehler eingebracht werden, wird das eher selten auffallen.

Ziel dieser Arbeit ist es nun ein System zu schaffen, welches eine solche systematische Datenkontrolle ermöglicht. Die Datenkontrolle soll in Form einer Bewertung stattfinden, die Nutzer für bestimmte Openstreetmap-Daten abgeben können. Dabei sollen folgende Kriterien berücksichtigt werden:

- Es gibt keine privilegierte Personen. Jeder kann editieren; jeder kann bewerten.
- Bewertungen sollen optional Informationen über Bewerter miteinbeziehen - werden also diesen zugeordnet.
- Nutzer sollen die Bewertungen anderer Nutzer einsehen können.
- Bewertungen sollen nicht gefälscht werden können.
- Bewertungen sollen verfallen bzw. zurückgezogen und korrigiert werden können.
- Bewerter sollen ihre Privatsphäre durch Pseudonyme schützen können.

Die wesentlichen Vorteile einer solchen Herangehensweise zur Qualitätssicherung der Openstreetmap-Daten liegen auf der Hand:

6. Navteq Map Reporter, http://mapreporter.navteq.com/
7. Openstreetbugs, http://www.openstreetbugs.org

- Eine inhaltliche Datenverifikation ist durch Einbeziehung von Menschen möglich.
- Die Veröffentlichung der Bewertungen ermöglicht eine systematische Qualitätskontrolle.
- Die Bewertungsqualität kann durch eine Einschätzung der Kompetenz des Bewerters erhöht werden.
- Das System ist nicht restriktiv, sondern erweitert die Möglichkeiten zur Qualitätseinschätzung von OSM-Daten.

Je mehr verschiedene Bewerter ein Objekt und seine Eigenschaften signieren, desto größer ist die Wahrscheinlichkeit, dass die in der Datenbank gespeicherten Daten für dieses Objekt richtig sind und je verbreiteter das Bewertungssystem ist, desto mehr Daten können auf diese Weise verifiziert werden.
Um das System massentauglich zu machen, werden einfache Benutzerschnittstellen und Visualisierungen der Bewertungen benötigt, was in dieser Arbeit aber nicht im Vordergrund stehen soll.

2 ANALYSE

Das Openstreetmap-Projekt kann nach [Ste08] als Projekt einer Internetcommunity verstanden werden, deren Ziel jedoch nicht hauptsächlich im Beziehungsaufbau der Mitglieder liegt, sondern in der Schaffung eines gemeinsamen Werkes - der freien Weltkarte. Die zum Erreichen des Ziels notwendige Kommunikation findet in der Regel über Wikis, Mailinglisten, Internetforen oder Blogs statt. Die Community ist grundsätzlich offen, das heißt, jeder kann sich am Verbessern der Karte, am Entwickeln und Erhalten der Projektinfrastruktur oder an den Communityabsprachen über grundsätzliche Regeln etc. beteiligen und damit Teil der Community werden.
Neben den aktiven Mitgliedern der Community, die dadurch definiert sind, dass sie Inhalte in der Community beitragen, gibt es jedoch auch passive Nutzer, die die von der Community zur Verfügung gestellten Inhalte nutzen möchten. Diese Einteilung kann als Rollenzuweisung verstanden werden, denn selbstverständlich treten auch aktive Mitglieder in der Rolle der passiven Nutzer auf und umgekehrt.
Dabei ist die Nutzung der Openstreetmap-Daten auf viele verschiedene Arten möglich. Zahlreiche thematische Karten im Internet[1] stellen die Daten in aufbereiteter Form mit unterschiedlichen Schwerpunkten für die Nutzer dar. Anwendungen für diverse mobile Endgeräte dienen der Navigation entlang berechneter Routen. Aber auch die Daten selbst können analysiert und verarbeitet werden.
Aufgrund der großen Offenheit und der zunehmenden Beliebtheit des Projektes, wächst jedoch die Gefahr von Missbräuchen. Besteht das eigentliche Ziel der Openstreetmapmitglieder darin, korrekte geographische Informationen an die Nutzer weiterzugeben, so bedeutet ein Missbrauch dem Nutzer fehlerhafte Informationen zu präsentieren, welche er nicht als solche erkennt. Fehlerhafte Informationen können hierbei bewusst erzeugt werden oder ungewollt entstehen. Sie sind jedoch nicht auszuschließen.
Da der Sinn einer Karte darin besteht, einen Nutzer mit für ihn unbekannten geographischen Gegebenheiten bekannt zu machen, ist der Nutzer im Vorhinein nicht in der Lage, die Gültigkeit der Informationen über diese Gegebenheiten zu überprüfen. Dieses Informationsdefizit birgt das Risiko von realen Fehlentscheidungen, deren Kosten in den vielfältigen Einsatzmöglichkeiten der Karten unterschiedlich hoch sein können, wie Beispiel 2.1 zeigt.

Beispiel 2.1 (Rettungswagen) *Ein Rettungswagenfahrer möchte die schnellste Strecke zu seinem Einsatzort mit Openstreetmap-Daten berechnen lassen. Falls dieser aufgrund eines Kartenfehlers in Openstreetmap verspätet seinen Zielort erreicht, können die Kosten des Fehlers im schlimmsten Fall Menschenleben bedeuten.*

1. Verschiedene OSM-Karten: http://www.openstreetmap.de/schaufenster/

2.1 EIN BEWERTUNGSSYSTEM FÜR OPENSTREETMAP

Ein Nutzer der Openstreetmap-Daten erwartet zunächst, dass diese korrekt sind. Dabei möchte er verständlicherweise möglichst wenig enttäuscht werden. Diese Erwartungshaltung kann auch als Vertrauen des Nutzers in die Openstreetmap-Daten beschrieben werden. Der Duden beschreibt Vertrauen als *„festes Überzeugtsein von der Verlässlichkeit, Zuverlässigkeit einer Person oder Sache"*. In diesem Sinne möchte sich der Nutzer von der Zuverlässigkeit der Openstreetmap-Daten überzeugen, somit also Vertrauen gewinnen.
Die Zuverlässigkeit einer Karte bedeutet für einen Nutzer nicht nur, dass die Daten in für ihn wichtigen Eigenschaften korrekt sind, sondern auch, ob eventuell auftretende Fehler für den Nutzer verantwortbar sind. So wird der Rettungswagenfahrer aus Beispiel 2.1 die Nutzung von Openstreetmap-Daten sehr wahrscheinlich ablehnen, da es bisher schwierig ist die Zuverlässigkeit auf hohem Niveau sicherzustellen und ein Fehler schwer wiegen könnte.

Um einen Eindruck von der Zuverlässigkeit der Openstreetmap-Karten zu bekommen bieten sich dem Nutzer im Moment beispielsweise folgende Möglichkeiten:

- **Ortskenntnis:** Besitzt ein Nutzer bereits geographisches Wissen über einen bestimmten Bereich, den die Karte abdeckt und stimmt die Karte mit den Kenntnissen des Nutzers überein, so wird es diesem nicht schwer fallen, der Karte auch in für ihn unbekannten Gebieten glauben zu schenken.
- **Kartenvergleich:** Um das Vertrauen in die Karte zu steigern, kann der Nutzer alternative Kartenwerke zu Rate ziehen, denen er mehr vertraut und vergleichen, ob die Informationen übereinstimmen.
- **Erfahrungen mit der Karte:** Der Nutzer kann die Karte auch direkt in der Realität testen und dabei feststellen, ob die Karte für ihn tauglich ist.

Problematisch bei diesen Ansätzen ist, dass zum einen der Aufwand für den Nutzer steigt und zum anderen eine Risikoabschätzung gerade bei Openstreetmap-Karten dennoch sehr schwierig ist. Dies liegt darin begründet, dass die Zuverlässigkeit der Openstreetmap-Karten lokal sehr unterschiedlich sein kann. Eine Openstreetmap-Karte setzt sich ja gerade aus den verschiedenen Beiträgen von unterschiedlichen Mappern zusammen. Hat man als Vergleichsgebiet einen Bereich gewählt, in dem ein sehr sorgfältig arbeitender Mapper beiträgt, so kann man durchaus zu fehlerhaften Annahmen über einen noch unbekannten Bereich kommen, in dem womöglich ein etwas oberflächlicherer Mapper arbeitet.
Bei kommerziellen Kartenanbietern hingegen kann man eher eine gleichmäßige Verteilung der Fehler erwarten, da von einheitlichen Standards innerhalb des Unternehmens auszugehen ist und die Zahl der Mitarbeiter prinzipiell sehr viel niedriger ist. Da sich Unternehmen aussuchen, wen sie einstellen, ist auch davon auszugehen, dass die Kompetenz der Mitarbeiter vergleichbar hoch ist, während bei Openstreetmap jeder, unabhängig von der Kompetenz, mitarbeiten kann und somit auch große Unterschiede in den Fähigkeiten der Nutzer bestehen, die sich wiederum auf die Daten auswirken können.

Zusätzlich zu den bisher genannten Möglichkeiten, das Kartenmaterial selbst zu überprüfen, kann ein Nutzer auch Personen vertrauen, die Aussagen über das Kartenmaterial treffen. Für gewöhnlich tut ein Nutzer das, wenn er kommerzielles Kartenmaterial kauft, indem er dem Unternehmen, welches die Karten herstellt ein gewisses Maß an Vertrauen entgegenbringt. Dieses Vertrauen ist nicht grundlos, wenn man unterstellt, dass Unternehmen ein wirtschaftliches Interesse daran haben ihr Produkt gut genug herzustellen.
Der Begriff „Person" soll hier und im Folgenden als juristische Person verstanden werden und somit auch Institutionen einbeziehen.

Beispiel 2.2 (Landesvermessungsamt) *Ein Nutzer, der eine Karte vom Sächsischen Landesvermessungsamt gekauft hat, erwartet, dass dieses Kartenmaterial zuverlässig ist. Er kennt das Landesvermessungsamt nicht selbst, aber glaubt, dass das Amt bei der Erfassung der Daten mit amtlicher Sorgfalt vorgeht und verlässt sich auf den guten Ruf des Amtes.*

Im Beispiel 2.2 wird deutlich, dass das Vertrauen zu einer Person nicht unbedingt persönliche Erfahrungen mit dieser voraussetzt. Setzt man voraus, dass die vom Nutzer gekaufte Karte authentisch ist, also wirklich vom Landesvermessungsamt stammt, so sind im Beispiel zwei weitere Dinge ausschlaggebend für das entgegengebrachte Vertrauen:

- **Glaubwürdigkeit:** Die Bereitschaft des Nutzers den Aussagen einer anderen Person, in diesem Fall des Amtes gelten zu lassen, bezeichnet man als Glaubwürdigkeit. Das Amt sagt von sich selbst: *„Der Staatsbetrieb Geobasisinformation und Vermessung Sachsen stellt die Geodatendienste mit der zur Erfüllung seiner öffentlichen Aufgaben erforderlichen Sorgfalt bereit."*[2] Ob der Nutzer das Amt nun für glaubwürdig hält, hängt unter anderem von den persönlichen Erfahrungen und der Überzeugungskraft des Amtes ab.
- **Reputation:** Ebensowichtig wie die Selbstdarstellung des Amtes ist seine Reputation, also welche Eigenschaften und Verhaltensweisen ihm von anderen Personen nachgesagt werden. Dabei können auch die Karten direkt eine Reputation erhalten. So findet man beispielsweise in einem Onlineshop für Karten die Aussage: *„Die Karten sind außerordentlich exakt und lassen keine Zweifel aufkommen."*[3]

Das Amt sowie jedes traditionelle Kartenunternehmen legen Wert darauf, dass Glaubwürdigkeit und Reputation möglichst gut sind. Die Reputation des Unternehmens hängt zudem häufig mit der Reputation des Produktes, also dem Kartenmaterial zusammen. Fehler des Kartenmaterials und damit Enttäuschungen für den Nutzer bergen die Gefahr für das Unternehmen, die Werte für Glaubwürdigkeit und Reputation zu verschlechtern - das Unternehmen steht also mit seinem Namen ein.

Bei dem Gemeinschaftsprojekt Openstreetmap sieht die Situation ein wenig anders aus. Die Community an sich kann nur mutmaßliche Aussagen über die Gültigkeit der eigenen Daten treffen, da es keinen Reviewprozess oder ähnliches gibt, um die Beiträge von Mitgliedern zu validieren. Es ist also wenig zielführend bei der Frage, ob ein bestimmtes Objekt in Openstreetmap vertrauenswürdig ist, die Frage nach der Glaubwürdigkeit bzw. Reputation von Openstreetmap zu stellen. Vielmehr müsste hinterfragt werden, welche Glaubwürdigkeit bzw. Reputation der Mapper besitzt, der das besagte Objekt eingetragen hat. Dies lässt sich jedoch nicht ohne weiteres klären, denn jeder Mapper kann sich ein beliebiges Pseudonym zulegen, unter welchem er editiert. Bis 2009 war es sogar möglich, die Karte komplett anonym zu editieren[4], also ohne dass die einzelnen Editiervorgänge verknüpfbar wären. Jetzt sind sie zumindest einem Pseudonym zuzuordnen und damit häufig einer Person, sofern die Person nicht für jeden Editiervorgang einen anderen Account benutzt, was aufwändig wäre, oder sich den Account mit anderen Personen teilt.
Meistens liegen einem Nutzer von Openstreetmap jedoch nur wenig bis gar keine Informationen über ein Pseudonym, also einen Openstreetmap-Account vor. Fasst man die gemappten Daten eines Pseudonyms als dessen Aussagen auf und hat man keine weiteren Informationen, so ist es nur schwer möglich, diese Aussagen begründet als glaubwürdig anzusehen. Eine Reputation von Mappern, die ihre Identität nicht preisgeben bzw. unter ihrem Pseudonym nicht in der Öffentlichkeit auftreten, ist schwer zu erhalten. Es ist im Übrigen auch gar nicht notwendig, dass die Mapper Informationen über sich preisgeben, um Informationen über die Zuverlässigkeit der eingetragenen Daten zu erhalten.

2. LVA Sachsen, http://www.landesvermessung.sachsen.de/inhalt/geo/basis/basis_nutz.html
3. MapFox, http://www.mapfox.de/WG_107.php
4. Anonymes Editieren, http://wiki.openstreetmap.org/wiki/Anonymous_edits

Eine vom Ersteller der Daten unabhängige Möglichkeit, Vertrauen in die Daten zu erlangen, besteht nämlich darin, den Daten selber eine gewisse Reputation zu verleihen. Weil Daten nicht interagieren können, können sie auch nicht aktiv Reputation sammeln. Die Reputation kommt beispielsweise dadurch zustande, dass Personen die Daten bewerten und ihre Bewertungen den Nutzern zugänglich machen. Die einzelnen Bewertungen werden mithilfe einer Reputationsfunktion zu einem Reputationswert aggregiert. Dabei ist zunächst zu klären, ob diese Funktion global ist und jeder Nutzer einem konkreten Objekt den gleichen Reputationswert zuordnet oder ob dies lokal geschieht und jeder Nutzer aufgrund unterschiedlicher Informationen, Wünschen und Anforderungen seine Reputationsfunktion selbst gestaltet. Im ersten Fall ist der Aufwand für den Nutzer geringer und die Daten werden hinsichtlich ihrer Reputation objektiv, also vom Nutzer unabhängig miteinander vergleichbar. Im zweiten Fall kann der Reputationswert für den einzelnen Nutzer aussagekräftiger werden, da dieser auch Informationen, die nicht global vorliegen, wie beispielsweise persönliche Bekanntschaft mit einem Bewerter, einbringen kann. Wichtig dafür ist, dass alle Bewertungen dem Nutzer zugänglich sind.
Es ist prinzipiell nicht ausgeschlossen, dass sowohl eine globale als auch mehrere lokale Reputationsfunktionen nebeneinander existieren.

Zusammenfassend veranschaulicht Abbildung 2.1 einige der genannten Möglichkeiten eines Nutzers, Vertrauen auf die Richtigkeit der Karte zu gewinnen. Sowohl persönliches Prüfen der Karte als auch die Aussagen von bewertenden Personen fließen in die eigene Einschätzung der Daten durch den Nutzer mit ein. Der eigentliche Ersteller bzw. der letzte Openstreetmapnutzer, der die betreffenden Daten editiert hat, könnte auch als spezieller Bewerter aufgefasst werden, ohne dass er zusätzlich eine Bewertung abgibt. Mit der Entscheidung, die Daten in dieser Art und Weise zu erstellen, hat er implizit ausgesagt, dass diese aus seiner Sicht so korrekt sind.

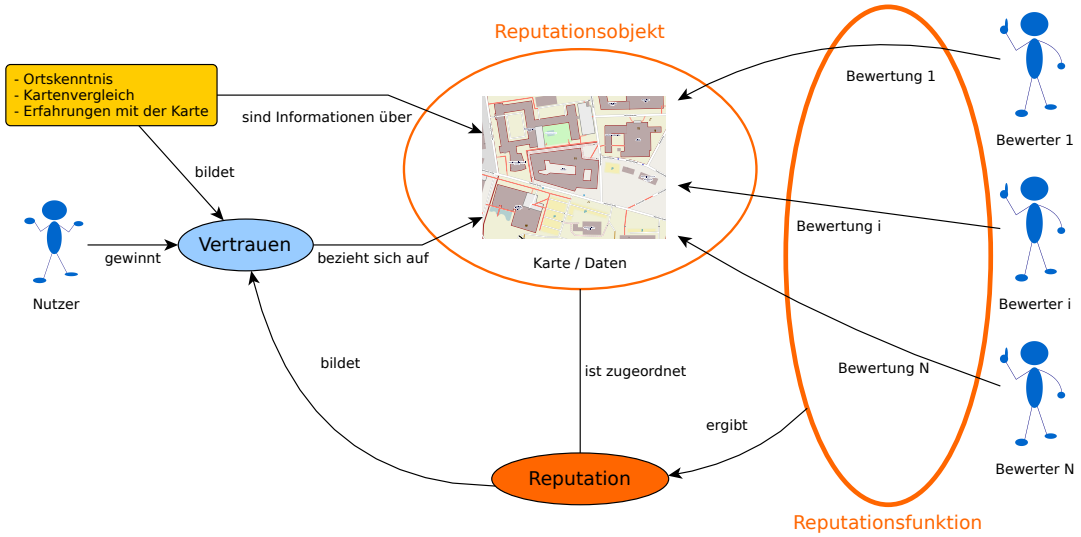

Abbildung 2.1: Der Nutzer gewinnt Kartenvertrauen durch eigene Informationen über die Karte oder durch Bewertungen anderer Personen, die mithilfe einer Reputationsfunktion die Reputation der Karte bilden.

2.2 KLASSIFIKATION VON REPUTATIONSSYSTEMEN

Bei Projekten, die das Erstellen gemeinschaftlicher öffentlicher Daten bzw. Inhalte zum Ziel haben, ist es sinnvoll zwei verschiedene Reputationssysteme voneinander abzugrenzen. Grundsätzlich interessiert den Nutzer des Systems natürlich der Reputationswert der erstellten Inhalte, welcher ihm durch ein Reputationssystem für den Inhalt mitgeteilt wird.
Neben diesem Reputationssystem ist es oft nützlich ein zweites Reputationssystem zu betreiben, welches den Erstellern oder Begutachtern der Inhalte eine Reputation verleiht. Diese Reputation wiederum kann zur Erstellung des Reputationswertes des Inhaltes beitragen, der von Nutzern mit eigener Reputation begutachtet oder editiert wurde. Die Berechnung erfolgt hierbei durch eine Reputationsfunktion.

Ein Reputationssystem besteht laut [SGM09] aus folgenden Komponenten:

- Einem Bewertungsalgorithmus der festlegt, welche Daten wie bewertet werden,
- einer Reputationsfunktion oder Reputationsalgorithmus als Zusammenfassung der Bewertungen,
- einer Verteilungsstrategie von Reputation und Bewertungen,
- der Speicherung von Reputation und Bewertungen und
- der Auswertung der Reputation durch einen Nutzer.

Die Gestaltung dieser Komponenten kann bei der Einteilung von Reputationssystemen eine wesentliche Rolle spielen. Im Folgenden sollen neben diesen Komponenten weitere Aspekte betrachtet werden, anhand derer ein Reputationssystem klassifiziert werden kann.

2.2.1 Reputationsfunktion

Ein wichtiger Parameter kann dabei die Reputationsfunktion sein, also der Algorithmus, der aus den reputationsrelevanten Daten einen Reputationswert berechnet. Reputationsrelevant sind beispielsweise abgegebene Bewertungen, zusätzliche Informationen, wie beispielsweise die Reputation über die beteiligten Bewerter oder einfach nur Verhaltensweisen bzw. Interaktionen zwischen Nutzern und dem Reputationsobjekt aus denen implizit Informationen über die Reputation des Objektes gewonnen werden können.
Die Reputationsfunktion kann beispielsweise eine einfache Durchschnittsbildung der Bewertungen sein oder probabilistische Modelle verwenden. Der Einfachheit halber könnten nur diskrete Werte wie „vertrauenswürdig“ oder „nicht vertrauenswürdig“ mit zusätzlichen Abstufungen zurückgegeben werden. Oft entstehen Zahlen als Ergebnis einer solchen beliebig komplizierten Funktion, die jedoch miteinander vergleichbar sein müssen, um den Reputationswert abwägen zu können.
Die Vielfalt an Reputationsfunktionen ist groß und ein genaues Festlegen, ist für die Entwicklung des Systems zur Reputation der Openstreetmap-Daten nicht nötig. Dieser Parameter ist flexibel wählbar und für diese Arbeit kein ausschlaggebendes Kriterium.

2.2.2 zentral versus dezentral

Ein gerne verwendetes Klassifizierungsmerkmal von Reputationssystemen ist die Handhabung der Verteilung der Bewertungsinformationen und der Reputation an die Nutzer. Es kann grundsätzlich zwischen einem zentralen und einem dezentralen Reputationssystem unterschieden werden [JIB07].
Von einem zentralen Reputationssystem spricht man hierbei, wenn die Bewertungen einer gemeinsamen Authorität, in Online-Szenarios also einem allen zugänglichen Server, übergeben bzw. von dort abgerufen werden können. Der Server kann dabei einen globalen Reputationswert aus den Bewertungen berechnen und den Nutzern auf Anfrage mitteilen. Die zentrale Verwaltung der Bewertungen ermöglicht einen gemeinsamen Kenntnisstand aller Nutzer des Systems auf die verfügbaren Bewertungsinformationen. Daraus folgende Beurteilungen können natürlich dennoch variieren.
Bei dezentralen oder auch verteilten Reputationssystemen werden im Gegensatz dazu die Bewertungen nicht zusammengeführt, sondern bei den Bewertern bzw. kleineren Gruppen von Bewertern gespeichert. Ein Nutzer kann das verteilte Netz nach hilfreichen Bewertungen durchsuchen und diese in seine Betrachtungen einbeziehen. Bei der Suche können verschiedene Nutzer zu verschiedenen Ergebnissen kommen und somit ihre Einschätzungen der Vertrauenswürdigkeit auf unterschiedlicher Faktenbasis treffen. Eine dritte Instanz wie ein zentraler Server, dem je nach Modell mehr oder weniger vertraut werden muss, entfällt hierbei. Der Nutzer kann die Entitäten, von denen er Bewertungen bzw. Reputation bezieht, selbst auswählen.

Eine spezielle Form eines dezentralen Systems ist das subjektive Reputationssystem [Ste08], bei dem nur die eigenen Erfahrungen zur Reputationsbildung genutzt werden. Es benötigt Interaktionen zwischen Bewerter und Reputationsobjekt und ist für ein System zur Bewertung von Daten ungeeignet. Denn Daten besitzen keine Verhaltensweisen aus denen sich Reputation erhalten lässt. Für die Bewertung von Begutachtern, also das zweite eingesetzte Reputationssystem kommt es nur dann in Frage, wenn zwischen den Benutzern Interaktionen stattfinden. Da Mapper bei Openstreetmap nicht interagieren müssen, ist es auch hier eher nicht relevant.
Weiterhin wird in [Ste08] eine Systemarchitektur mit lokaler Speicherung der Reputation erwähnt. Die Reputationsobjekte sammeln Bewertungen und speichern ihre Reputation selbst. Sofern die Fälschungssicherheit der Bewertungen sichergestellt ist, kann diese Form sowohl bei dem Reputationssystem für die Daten als auch für das der Begutachter zum Einsatz kommen.

Das in dieser Arbeit betrachtete Bewertungssystem soll grundsätzlich zentral konstruiert werden. Dies ist naheliegend, da die Reputationsobjekte des Inhaltsreputationssystems, also die Openstreetmap-Daten auch schon zentral verwaltet werden. Die Bewertungen sollen möglichst vielen Benutzern zugänglich sein und eine allgemeine Faktenbasis schaffen.
Eine ganz strikte Einordnung in eine der genannten Architekturen ist jedoch nicht notwendig. Es kann sehr wohl zu Mischformen kommen. So soll zwar ein zentraler Server existieren, aber eine Bewertung von Daten oder anderen Bewertern und eine Weitergabe dieser soll auch ohne diesen Server möglich sein, wenngleich sie dann möglicherweise nicht an alle Nutzer verbreitet wird. Vor allem für das Bewerterreputationssystem kann eine dezentrale Lösung in Frage kommen, wenn durch die Verbreitung der Begutachterbewertung an alle Nutzer Nachteile entstehen können. Wird ein Benutzer beispielsweise von einem anderen Benutzer bewertet, so kann aus dieser Bewertung eine Beziehung zwischen den Nutzern hergestellt werden, die ein Angriff auf die Privatsphäre darstellen könnte.

2.2.3 inhaltsorientiert versus nutzerorientiert

Weiterhin lassen sich zwei grundlegende Ansätze von Reputationssystemen für internetbasierte Kollaborationsprojekte unterschieden - die inhaltsorientierten (content-driven) und die nutzerorientierten (user-driven) Systeme [AdAKP10].
Basiert die Reputationserstellung auf expliziten Bewertungen und Rückmeldungen von Nutzern,

so handelt es sich um nutzerorientierte Systeme. Nutzer bewerten also aktiv Daten oder andere Nutzer.
Demgegenüber stehen inhaltsorientierte Reputationssysteme, bei denen Reputation anhand des beigetragenen Inhaltes von Nutzern automatisch generiert wird. Grundlage ist eine Analyse von Interaktionen der Nutzer mit Inhalten oder anderen Nutzern, die allerdings nicht primär zum Zwecke der Reputationsgewinnung stattfinden.

Die Betrachtung von nutzer- und inhaltsorientierten Systemen ist für diese Arbeit deswegen so interessant, weil es bereits ein im Einsatz befindliches inhaltsorientiertes Reputationssystem für das Communityprojekt Wikipedia namens WikiTrust gibt [AD07, AdAKP10], und Wikipedia und Openstreetmap viele Gemeinsamkeiten besitzen.

Im Folgenden werden daher einige Eigenschaften, Vor- und Nachteile von nutzer- und inhaltsorientierten Systemen beschrieben und im Zusammenhang mit Openstreetmap betrachtet sowie anschließend die Anwendung eines Systems wie WikiTrust für Openstreetmap diskutiert.

Nutzerorientierte Systeme werden bereits häufig auf Onlinemarktplätzen wie bei Ebay oder Amazon eingesetzt. Dort bewerten Nutzer direkt andere Nutzer bzw. Produkte und verleihen ihnen damit Reputation. In solchen Marktplatzszenarios haben Nutzer mit höherer Reputation oft einen wirtschaftlichen Vorteil gegenüber Konkurrenten mit niedrigerer Reputation. Denn aus dem Vertrauen, was Handelspartner ihnen aufgrund der Reputation entgegenbringen, entstehen mehr Geschäftsbeziehungen.
Jeder Nutzer ist also bestrebt eine möglichst gute Reputation zu erhalten. Dabei kann er den durch das Reputationssystem vorgesehenen Weg gehen und beispielsweise durch gute Geschäfte etc. versuchen gute Bewertungen zu erreichen. Eine andere Möglichkeit wäre das Reputationssystem anzugreifen [PR08] und bei Erfolg gutgläubige Nutzer zu täuschen.
Ein Problem bei nutzerorientierten Systemen besteht darin, dass die abgegebenen Bewertungen nicht immer die wirklichen Ansichten des Bewerters ausdrücken müssen. So kann ein Bewerter aus Angst vor negativen Rachebewertungen einem Nutzer eine bessere Bewertung verleihen, als er eigentlich für richtig hält [DFW05]. Denn von positiven Bewertungen profitieren beide.
Ein solches Vorgehen ist bei inhaltsorientierten Systemen schwieriger, da Bewertung und Interaktion nicht getrennt sind. Wöllte man Reputationen fälschen, so müsste man Interaktionen gezielt manipulieren, was aufwändiger wäre.
Das Probem von falschen Bewertungen ist für ein nutzerorientiertes Reputationssystem für Openstreetmap im Unterschied zu Reputationssystemen für Onlinemarktplätze dahingehend weniger groß, dass eine positive Reputation keinen offensichtlichen ökonomischen Vorteil bringt. Die grundlegende Motivation zur Teilnahme an einem Reputationssystem für Openstreetmap ist nämlich nicht der eigene Vorteil durch hohe Reputation, sondern die Vermittlung von Informationen an andere Nutzer. Ein Missbrauch von hoher Reputation führt eher zur Schädigung anderer Nutzer als zum Erlangen eines eigenen Vorteils. In wenigen konstruierten Fällen könnte das einher gehen, aber es entspricht nicht der Regel.
Doch genau dieser eher altruistische Ansatz des Reputationssystems für Openstreetmap-Daten führt zu einem weiteren Problem nutzerorientierter Systeme, das hierbei besonders stark zum Tragen kommen könnte. Nutzer müssen aktiv handeln, um eine Bewertung abzugeben. Bei Ebay werden etwas mehr als die Hälfte aller Auktionen bewertet [JIB07]. Diese beachtliche Anzahl wird in [DFW05] mit einer Mischung aus altruistischem, egoistischem und bewertungserwiderndem Verhalten der Nutzer erklärt.
Das Problem der Bewertermotivation könnte auch mit zusätzlichen Anreizen wie materiellen Belohnungen oder Vorteilen des Bewerters gemildert werden.
Bewerter des Systems für Openstreetmap können Anerkennung und Glaubwürdigkeit erlangen. Ein gewisser Eigennutz besteht zusätzlich darin, dass ihr Beitrag das System am Leben hält, von dem sie auch selbst als Nutzer profitieren können. Diese Einstellung ist in der Openstreetmap-Community verbreitet und wird auch häufig als Begründung für das Beitragen von Daten überhaupt angeführt.

Ein Motivationsproblem der Bewerter gibt es bei inhaltsorientierten Reputationssystemen nicht. Der große Vorteil solcher Systeme besteht nämlich darin, dass sie nicht auf die explizite Mithife

Wikipedia-Artikel	Openstreetmap-Objekt
Gemeinsamkeiten:	
besitzen Historie	
frei editierbar	
Inhalt kann Bearbeitern zugeordnet werden	
Unterschiede:	
nicht ausschließlich Fakten, nicht immer prüfbar	prüfbare Fakten (ground truth)
enthält überwiegend Prosatexte	besteht aus strukturierten Daten
oft umfangreiche Texte	wenige Attribute und Metadaten
hauptsächlich unbegrenzt gültige Inhalte	Inhalte dynamisch
werden häufig geändert	werden selten geändert
werden über Namen adressiert	werden durch ID adressiert

Tabelle 2.1: Wichtige Gemeinsamkeiten und Unterschiede von Wikipedia-Artikeln und OSM-Objekten.

von Nutzern angewiesen sind. Die Reputation fällt einfach neben den gewöhnlichen Interaktionen mit den Daten an. Dadurch ist jeder, der mit den Daten nachvollziehbar interagiert, automatisch Teil des Bewertungssystems. Liegen bereits Daten über Interaktionen vor, wie es bei Wikipedia in Form der Artikelhistorien der Fall ist, dann lässt sich die Reputation für jeden Nutzer sofort daraus berechnen. Das System benötigt also keine Anlaufzeit wie ein nutzerorientiertes System, in der es ersteinmal etabliert werden muss.
Problematisch bei inhaltsorientierten Reputationssystemen sind zum einen die getroffenen Annahmen, um Reputation aus den Interaktionen zu gewinnen, und zum anderen die Vermittlung der Ergebnisse an die Endnutzer. Welche Schlüsse ein Endnutzer aus einem mitgeteilten Reputationswert ziehen kann und wie er sie beurteilt, hängt stark davon ab, wie gut der Nutzer das System und die zugrundeliegenden Annahmen verstanden hat. Nutzerorientierte Systeme haben hier den Vorteil, dass ein Reputationswert, der aus einer klaren Äußerung eines Nutzers resultiert, oft leichter verständlich ist, als einer, der sich aus einer Berechnung mit Informationen aus Nutzerinteraktionen ergibt. Beispielsweise ist für die Aussage „Drei Begutachter haben dieses Datum positiv bewertet." weniger Systemkenntnis für das Verständnis erforderlich als bei einer Aussage wie „Für dieses Datum wurde der Reputationswert drei berechnet."
Auch wenn bei nutzerorientierten Systemen aus den Bewertungen ein Reputationswert gebildet wird, so sind dennoch die Grundlagen dieses Wertes, also die einzelnen Bewertungen, für viele Nutzer leichter nachvollziehbar als die Analyse von Nutzerinteraktionen.

Anwendung des WikiTrust-Prinzips bei Openstreetmap

Ein Beispiel für ein inhaltsorientiertes Reputationssystem ist WikiTrust [AD07]. Als Plugin für den Firefox Browser verfügbar bietet WikiTrust ein Reputationssystem, um Nutzern zu helfen, die Zuverlässigkeit von Textinhalten in der Wikipedia einschätzen zu können. Für jeden Nutzeraccount, mit dem in der Wikipedia editiert wurde, kann ein Reputationswert berechnet werden. Für einen Textbeitrag in einem Artikel erhält ein Nutzer positive Reputation, wenn dieser Beitrag von nachfolgenden Editoren unverändert übernommen wird. Wird der Beitrag jedoch geändert oder gar gelöscht, kann der Nutzer Reputation verlieren. Dabei wird vorausgesetzt, dass jemand, der bei einem Textabschnitt Änderungen vornimmt, den Teil, den er belässt als richtig einschätzt. Um den ursprünglichen Ersteller eines Textabschnittes herauszufinden, werden die Historie eines Artikels untersucht und Artikelversionen miteinander verglichen.
Die Reputation des Nutzers wirkt sich wiederum auf Beiträge aus, die der Nutzer editiert. Auch hier werden also zwei Reputationssysteme parallel eingesetzt.

Die Idee eines inhaltsorientierten Reputationssystems wie WikiTrust ließe sich grundsätzlich auch auf Openstreetmap anwenden, da es zwischen Wikipedia und Openstreetmap grundlegende Gemeinsamkeiten gibt. Beide Projekte bestehen aus einer Gruppe von Menschen, deren Ziel es ist,

ihr Wissen zu teilen und mit anderen gemeinsam freie öffentliche Inhalte zu erzeugen. Bei beiden kann jeder Inhalte beitragen und vorhandene Inhalte ändern. Statt Wikipedia-Artikeln würden durch das inhaltsorientierte System geographische Objekte und deren Bearbeiter analysiert, um Reputation zu berechnen. Interaktionen mit den Daten finden dann statt, wenn Bearbeiter Tags oder die Position von Openstreetmap-Objekten verändern und damit einen neuen Eintrag in der Historie dieses Objektes erzeugen.
In Tabelle 2.1 werden Wikipedia-Artikel und Openstreetmap-Objekt miteinander verglichen. Beide besitzen eine Historie, die jede Änderung erkennbar und einem entsprechenden Nutzer zuordenbar macht, was Voraussetzung für ein inhaltsorientiertes Reputationssystem ist. Es gibt jedoch wesentliche Unterschiede, die einer Anwendung eines solchen Systems entgegenstehen.

Die Kritikpunkte sind:

- **Objektreferenzen können leicht verloren gehen.** Anders als in Wikipedia, wo Artikel über ihren Titel referenziert werden, werden in Openstreetmap Datenobjekte mit einer eindeutigen Identifikationsnummer (ID) gekennzeichnet. Diese IDs haben jedoch anders als ein Artikelname keinerlei inhaltlichen Bezug zum jeweiligen Datenobjekt. Wird ein Datenobjekt, z.B. eine Straße, erst gelöscht und später wieder neu angelegt, besitzt das neue Datenobjekt eine neue ID und die Historie des Vorgängerobjektes ist nicht mit ihm verknüpft. Bei inhaltlichen Datenkontrollen fällt solch eine Manipulation nicht auf, da aktuell vorhandene Daten dadurch nicht verschlechtert wurden. Für ein inhaltsorientiertes Reputationssystem stellt dies jedoch ein ernsthaftes Problem dar.

- **Annahmen zur Reputationserstellung aus Interaktionen sind bei Openstreetmap zweifelhaft.** Gemäß des WikiTrust-Prinzips würden Eigenschaften eines Objektes, die einen Editiervorgang eines Mappers unangetastet überstehen, positive Reputation erhalten. Es wird hier unterstellt, dass ein Mapper diese Eigenschaften deswegen nicht ändert, weil er sie als korrekt ansieht. Bei Openstreetmap liegen einem Mapper nicht immer alle möglichen Informationen zu einem Objekt vor, so dass er die bereits eingetragenen gar nicht prüfen kann. Jemand der bei einer Straße beispielsweise die zulässige Höchstgeschwindigkeit kennt und eintragen will, hat womöglich nicht auf den Straßennamen geachtet. Oder jemand, der eine neu erfasste an eine bereits bestehende Straße anschließt und letztere somit verändert, muss die bestehende Straße nicht einmal anschauen, um sie zu ändern.
  Bei der Erweiterung von Texten wie in der Wikipedia ist es eher üblich, den bestehenden Text zumindest zu lesen und gegebenenfalls abzuändern und umzustrukturieren, um seinen eigenen Beitrag passend einzufügen. Auch hier können dem Bearbeiter Informationen zu bereits vorhandenem Text fehlen, allerdings stellt sich die Frage, ob er ihn dann wirklich verbessern kann.

- **Überlebensfähigkeit von OSM-Daten ist keine geeignete Reputationsgrundlage.** Der Logik des WikiTrust-Prinzips zufolge verdienen Mapper, deren Daten entweder eine längere Zeitspanne oder eine höhere Anzahl an Editierungen überleben, mehr Reputation. Da sich die Objekte der Realität, auf die sich die OSM-Objekte beziehen, auch gelegentlich ändern können, ist es bei einigen Objekten mit höherer Änderungshäufigkeit so, dass eine lange Zeitspanne die Gültigkeit der Information eher ungewisser macht. Wird die Anzahl der Editierungen, die bestimmte Eigenschaften von Objekten überstanden haben, als Grundlage für die Überlebensfähigkeit verwendet, so kann zumindest die Hoffnung bestehen, dass der editierende Mapper die Daten geprüft hat. Ändert ein Mapper gewisse Daten, so heißt das aber nicht automatisch, dass der Vorgänger diese Daten falsch eingetragen hat und deswegen schlechte Reputation verdient. Die Faktenlage kann sich einfach geändert haben. Wikipedia-Artikel beschreiben jedoch häufig allgemeingültige statische Sachverhalte. Nur wenige Texte müssen dynamisch angepasst werden z.B. Lebensläufe noch lebender Personen oder Artikel mit Bezug zu aktuellen Geschehnissen. Und auch dort können die Inhalte nach einer gewissen Zeit in der Regel fixiert werden.
  Durch den Anspruch, aktuelles Kartenmaterial zu liefern, und den Umstand einer sich wandelnden Realität besitzt Openstreetmap diese Eigenschaft der langfristigen Fixierung nicht, und Daten werden, wenn auch nur selten, geändert werden müssen, um aktuell zu bleiben.

- **Reputation entsteht nur durch Editieren.** Bereits bestehende Openstreetmap-Daten werden in der Regel wesentlich weniger häufig editiert, als Wikipedia-Artikel. Während Openstreetmap-Objekte im Schnitt 1,5 Mal editiert werden, gibt es bei Wikipedia-Artikeln mehr als 26 Bearbeitungen pro Artikel[5]. Ein Wikipedia-Artikel wird gewöhnlich iterativ entwickelt und verbessert. Dabei finden viele Interaktionen der Bearbeiter mit dem Artikel statt. Ein Wikipedia-Artikel lebt geradezu von den Bearbeitungen, die Nutzer an ihm durchführen. Da Openstreetmap-Objekte aber meistens knappe Fakten darstellen, können diese oft mit wenigen Bearbeitungen erstellt werden. Die strukturierten Daten benötigen auch keine Stil- oder Formverbesserungen, wie Prosatexte in der Wikipedia. Zwar werden hin und wieder zusätzliche Eigenschaften hinzugefügt oder bestehende Eigenschaften korrigiert, aber dies geschieht dennoch im Vergleich zum neu Anlegen von Objekten recht selten.
  Sieht ein Mapper keinen Grund ein Objekt zu editieren, so wird er ihm auch keine Reputation verleihen, bzw. ein vorheriger Bearbeiter des Objektes keine bekommen können.
- **Automatische Editierungen verfälschen die Reputation.** Nicht hinter jedem in Openstreetmap aktiven Account steht eine Person. Die strukturierten Daten werden gelegentlich auch von sogenannten „bots“ , also Computerprogrammen editiert. Dies geschieht beispielsweise bei Datenimporten oder großflächigen Korrekturen, die jedoch eher selten auftreten. Diese Programme können vorhandene Inhalte nicht überprüfen und könnten damit ungewollte Reputation erhalten bzw. hervorrufen.

Ein inhaltsorientiertes Reputationssystem, das die Editierungen der Mapper analysiert, wäre für Openstreetmap denkbar, hätte aber die aufgezählten Nachteile. Leider stehen andere analysierbare Interaktionen von Personen mit den Daten, wie z.B. die Nachfrage bestimmter Daten, Gebiete etc., nicht direkt zur Verfügung.

In dieser Arbeit wird deswegen ein nutzerorientiertes System, also ein Bewertungssystem entwickelt.

2.3 ANFORDERUNGEN AN DAS BEWERTUNGSSYSTEM

Um den Openstreetmap-Daten Reputation zu verleihen, soll ein Werkzeug zur Abgabe und Analyse von Bewertungen geschaffen werden, welches sowohl von Bewertern als auch von Auswertenden, also den Nutzern der Openstreetmapdaten, verwendet werden kann. Die Bewerter sollen in der Lage sein, ihre Informationen über Openstreetmap-Daten auszudrücken und in definierter Form zu veröffentlichen. Auswertende können diese nutzen, um Vertrauen in die Daten zu gewinnen, wie in Abschnitt 2.1 erklärt.

2.3.1 Funktionale Anforderungen

Im Folgenden werden zunächst grundsätzliche funktionale Anforderungen an das Bewertungssystem für Openstreetmap-Daten ermittelt und diskutiert. Sie definieren, welche Aufgaben das System übernimmt und wie die Teilnehmer dabei mit dem System interagieren können.

- **Bewertungsabgabe:** Mehrere Personen sollen in der Lage sein, ihr Wissen über die Richtigkeit der Openstreetmap-Daten in Form einer Bewertung mitzuteilen. Das System nimmt diese Bewertung in einer bestimmten Form und unter Einhaltung gewisser Regeln entgegen.

5. Statistik deutsche Wikipedia (abgerufen November 2010), http://de.wikipedia.org/wiki/Spezial:Statistik

- **Bewertungsspeicherung:** Alle abgegebenen Bewertungen sollen im System gespeichert werden.
- **Bewertungsabfrage:** Die gespeicherten Bewertungen sollen jederzeit von jedem abgerufen werden können.
- **Bewertungsauswertung:** Ein Nutzer des Systems soll die Möglichkeit einer Auswertung der Bewertungen haben. Gültige und ungültige Bewertungen sollen unterscheidbar sein. Mithilfe des Systems soll der Nutzer zu einer Gesamteinschätzung der Zuverlässigkeit des interessierenden Datenobjektes gelangen.
- **Bewertungswiderruf:** Ein Bewerter soll in der Lage sein, seine eigene Bewertung für ungültig zu erklären bzw. zu erneuern.

2.3.2 Nebenbedingungen

- **Skalierbarkeit:** Das System soll bei wachsender Zahl der Bewertungen in seiner Nützlichkeit nicht eingeschränkt werden. Problematisch dabei ist zum einen der wachsende Ressourcenverbrauch (Speicher, Kommunikationsaufwand, etc. ...), aber auch Schwierigkeiten des Nutzers, aus der wachsenden Menge von Bewertungen eine aussagekräftige Abschätzung der Zuverlässigkeit der Daten zu erhalten.
- **Pluginansatz:** Das gesamte Bewertungssystem soll für Openstreetmap als Ergänzung dienen und möglichst nicht invasiv in das bestehende System eingreifen. Die Realisierung eines Bewertungseditors als Plugin für eine OSM-Dateneditor-Software stärkt die Akzeptanz und ermöglicht eine für das restliche System unkritische Entwicklung. Demzufolge sollen OSM-Daten weiterhin ungehindert ergänzt werden können. Auch bereits bewertete Daten sollen bei Bedarf korrigiert werden können. Dies wird zwar Auswirkungen auf die Gültigkeit der abgegebenen Bewertungen haben, aber stellt kein Hindernis dar und schränkt Openstreetmap in der bisherigen Flexibilität nicht ein.
- **Gebrauchstauglichkeit:** Bei der Erfüllung seiner Aufgaben sollte das Bewertungssystem nicht zu viele Voraussetzungen an seine Benutzer stellen. Die wesentlichen Aufgaben müssen verständlich sein und die nötige Interaktion mit dem Nutzer möglichst intuitiv. Der Nutzer muss auf sicherheitskritische Einstellungen und Abläufe hingewiesen, aber darf dennoch nicht mit zu viel Detailinformationen überfordert werden.

2.3.3 Sicherheitsanforderungen

Das Bewertungssystem für Openstreetmap-Daten soll im Folgenden unter den Aspekten von mehrseitiger Sicherheit diskutiert werden. Das bedeutet, dass die Sicherheitsinteressen aller Beteiligten einbezogen und im Konfliktfall gegeneinander abgewogen werden. Beteiligte sind hierbei die Bewerter, sowie die Nutzer der Bewertungen.

Die in Betracht gezogenen Interessen können in folgende Schutzziele gefasst werden:

- Verfügbarkeit der Bewertungen: Nutzer möchten die Vertrauenswürdigkeit von OSM-Daten anhand vorhandener Bewertungen einschätzen können und benötigen deshalb Zugang zu den Bewertungen.
- Integrität der Bewertungen: Abgegebene Bewertungen sollen nicht unauthorisiert manipuliert werden können.
- Vollständigkeit der Bewertungen: Nutzer möchten alle abgegebenen Bewertungen einsehen können.

- Zurechenbarkeit von Bewerter zu Bewertungen: Nutzer möchten, dass Bewertungen von den Bewertern stammen, von denen sie zu stammen scheinen.
- Anonymität der Bewerter: Bewerter möchten möglichst anonym bleiben und das Erstellen von beispielsweise personalisierten Interessensprofilen aufgrund ihrer Bewertungen verhindern.

Die Schutzziele sollten, wenn möglich, mit den funktionalen Anforderungen aus Abschnitt 2.3.1 sowie den Nebenbedingungen aus Abschnitt 2.3.2 in Einklang stehen. Dabei werden nur bestimmte Anforderungen berührt und womöglich konkretisiert.

So betrifft die **Verfügbarkeit** der Bewertungen vor allem die Bewertungsspeicherung und die Bewertungsabfrage. Daten müssen so gespeichert werden, dass der Zugriff darauf jederzeit durch jeden Nutzer möglichst schnell erfolgen kann. Im Sinne der Skalierbarkeit muss ein Mehrbenutzerbetrieb ermöglicht werden. Bei einem zentralen Ansatz, also beispielsweise der Einsatz eines zentrale Servers zur Speicherung der Bewertungen sollte weiterhin die Ausfallsicherheit durch Backups und Ersatzserver gesichert werden. Die absolute Verfügbarkeit findet ihre Grenzen natürlich in den technischen und physikalischen Möglichkeiten der Nutzung der begrenzten Ressourcen. Vor allem aber auch in der finanziellen Knappheit eines Freizeitprojektes wie Openstreetmap, die den Unterhalt von hilfreichen, aber nicht unbedingt notwendigen Ressourcen nicht erlaubt.

Die **Integrität** der Bewertungen ist vor allem auf den Kommunikationswegen zwischen Bewerter und Server bzw. Server und Nutzer gefährdet. Die Bewertungsabgabe an sich findet auf dem Rechner des Bewerters statt, der als vertrauenswürdiger Bereich angesehen werden muss. Zum Zeitpunkt der Abgabe kann die Bewertung natürlich beliebig manipuliert werden. Schadsoftware wie Trojaner könnten den Rechner des Bewerters befallen und diese Bewertungen ohne Kenntnis des Bewerters erstellen. Zusätzliche Maßnahmen wie die Verwendung externer vertrauenswürdiger zertifizierter Geräte wären für das System schlicht unangemessen und stehen in keinem Verhältnis zu den eher geringen Schäden durch manipulierte Bewertungen. Der Rechner des Bewerters unterliegt zudem dem Verantwortungsbereich des Bewerters der beliebige Sicherheitsmaßnahmen ergreifen kann, um die Kontrolle über den Rechner zu behalten.
Diese Arbeit wird somit Maßnahmen zur Integritätssicherung vorstellen, die erst nach der Bewertungsabgabe, während der Bewertungsspeicherung und vor allem auch bei der Bewertungsabfrage greifen. Ein Angriff der Integrität sollte spätestens während der Bewertungsauswertung aufgedeckt werden können.

Die **Vollständigkeit** der Bewertungen betrifft auch vor allem die Kommunikationswege zwischen Abgabe, Speicherung und Abfrage. Besonders interessant ist die Frage, ob einem zentralen Server bei der Speicherung der Bewertungen vertraut werden muss oder nicht. Bewerter können prüfen, ob ihre Bewertung aufgenommen wurde, indem sie die Bewertungen vom Server abfragen und testen, ob ihre dabei ist. Der Server darf an dieser Stelle nicht wissen, wer gerade die Bewertungen abruft, da er die Antwort sonst wieder speziell manipulieren könnte, um keinen Verdacht zu erwecken. Eine anonyme Abfrage muss also möglich sein. Wenn auch Nutzer in der Lage sein sollen zu überprüfen, dass die Antwort des Servers vollständig ist und alle verfügbaren Bewertungen enthält, müssen zusätzliche Vorkehrungen getroffen werden. Notwendig ist dies nicht, wenn man unterstellt, dass der Server gegen externe Angriffe ausreichend gesichert ist und die Betreiber des Servers wenig Interesse daran haben, dem System zu schaden. Da zudem die Kontrollmöglichkeiten der Bewerter gegeben sind und der Server nicht zwischen Bewerter und Nutzer bei der Abfrage unterscheiden können soll, ist eine gezielte Manipulation kaum möglich und nur mit hohem Risiko der Entdeckung verbunden.

Zurechenbarkeit ist wichtig, um bei einer Bewertungsauswertung die richtige Bewerterreputation einfließen zu lassen. Die Bewerterreputation wird an der Bewerteridentität festgemacht. Ordnet ein Nutzer eine Bewertung einer falschen Bewerteridentität zu, verwendet er demzufolge auch deren Bewerterreputation, was zu falschen Ergebnissen führen kann.
Zurechenbarkeit ist zudem aus Sicht des Bewerters notwendig, der in der Lage sein muss, sich

als Urheber einer Bewertung, die er widerrufen möchte, auszuweisen. Anderenfalls könnte der Bewertungswiderruf zur unerwünschten Manipulation missbraucht werden.

Anonymität kann an allen Stellen des Systems ohne weiteres sichergestellt werden, wenn keine Funktionalität beeinträchtigt ist. So kann die Bewertungsabfrage komplett anonym erfolgen. Wer die Bewertungen speichert, muss auch nicht bekannt sein, so lange sie auffindbar bzw. adressierbar sind, was die Verwendung von Pseudonymen erfordert. Dies spielt vor allem bei dezentralen System eine Rolle, bei der die Bewertungen in einem verteilten Netz gespeichert sind.

Abwägung der Sicherheitsinteressen gegeneinander

Auch die Schutzziele selber können Gegensätze darstellen. Aus den verschiedenen Zielen sowie den übrigen Systemanforderungen und Nutzerinteressen werden daher nun einige Konfliktpunkte vorgestellt und diskutiert.

- **Verfügbarkeit und Vollständigkeit gegen Bewerterprivatsphäre**
  Bewerter könnten ein Interesse daran haben, dass nicht alle ihre Bewertungen jedem zugänglich sind, da aus den Bewertungen möglicherweise Rückschlüsse auf den Bewerter bzw. seine Interessen oder Gewohnheiten gezogen werden können. Die Bewertungen könnten erst nach vorheriger Anmeldung an einem System abrufbar sein, wobei der Bewerter Regeln aufstellen könnte, wer seine Bewertungen abrufen darf.
  Dieses Vorgehen widerspricht jedoch grundlegend dem Interesse der Nutzer des Systems möglichst alle Bewertungen einzusehen, um eine gute Abschätzung der Zuverlässigkeit der Daten zu erreichen. Kompromisslösungen, wie das Mitteilen des resultierenden Reputationswertes durch eine dritte Instanz ohne Herausgabe der zugrundeliegenden Bewertungen, setzen zusätzliches Vertrauen in diese Instanz voraus. Das Resultat ist höchstens so glaubwürdig wie diese dritte Instanz.
  Aber auch aus Sicht der Bewerter spricht etwas dafür, die Bewertungen möglichst ohne Hindernisse für die Nutzer zu veröffentlichen. Je mehr Nutzer die Bewertungen einsehen können, desto hilfreicher sind diese. Setzt man voraus, dass es die grundlegende Motivation eines Bewerters ist, möglichst vielen Benutzern bei der Beurteilung der Qualität der Openstreetmap-Daten zu helfen, dann möchte er natürlich auch, dass seine Bewertungen jedermann zur Verfügung stehen. Bewertungen, die nur ausgewählten Personen dienen, könnten zwar in bestimmten Fällen gewünscht sein, widersprechen aber dem Wesen eines gemeinschaftsgetriebenen Opensourceprojektes wie Openstreetmap, das möglichst vielen Leuten ein Profitieren von gesammelten Informationen ermöglichen will.
  Die Abwägung für das System fällt also klar zu Gunsten von Verfügbarkeit und Vollständigkeit. Das berechtigte Interesse der Bewerterprivatsphäre kann statt durch Einschränken bzw. Erschweren der Verbreitung der Bewertung vielmehr durch Anonymisierung oder Pseudonymisierung erreicht werden.

- **Integrität bei Datenänderung**
  Damit die Aussagekraft einer Bewertung erhalten bleiben kann, muss gewährleistet sein, dass diese nach Abgabe nicht beliebig verändert werden kann. Dabei muss zum einen die Zuordnung von Bewertung zu bewerteten Daten gewahrt bleiben und zum anderen die Aussage der Bewertung selbst. Problematisch ist, dass bei Openstreetmap auch die bewerteten Daten weiterhin änderbar sein sollen, was zunächst einem Angriff auf die Integrität der Gesamtbewertung entspricht, da die Integrität der bewerteten Daten ebenso wichtig ist wie die Bewertung selbst.
  Weil Openstreetmap über eine Historie von Zuständen bzw. Versionen eines Objektes verfügt, lässt sich jedoch eine Datenänderung bei Openstreetmap von einer Datenänderung der bewerteten Daten entkoppeln. Die Integrität der Daten bezieht sich demzufolge auf einen bestimmten Stand eines Objektes bzw. einer Eigenschaft eines Objektes. Dieser Konflikt lässt sich also auflösen. Ob für einen alten Stand eines Objektes abgegebene Bewertungen für die Bewertung des aktuellen Standes des Objektes nützlich sein können, wird später in Abschnitt 2.6 genauer betrachtet.

- **Zurechenbarkeit gegen Anonymität**
  Bei der Abwägung der Interessen von Bewertern und Nutzern bilden die Anforderungen an Zurechenbarkeit und Anonymität einen Gegensatz.
  Für die Nutzer, die aus den abgegebenen Bewertungen von Daten, Informationen über deren Richtigkeit erhalten wollen, ist es von Vorteil, wenn die Bewerter möglichst klar auszumachen sind. Im Kontext der Bewerter kann deren Bewertung von den Nutzern besser eingeschätzt werden. Dabei muss nicht unbedingt der echte Name in Erscheinung treten. Es kann möglicherweise auch ein Pseudonym (Nickname) verwendet werden. Über die Verknüpfbarkeit zu anderen abgegebenen Bewertungen des Bewerters ist der Nutzer besser in der Lage, die Glaubwürdigkeit des Bewerters einzuschätzen.
  Auf der anderen Seite ist der Bewerter gefährdet, zu viel von sich preiszugeben und muss dadurch einen Angriff auf seine Privatsphäre befürchten. Beinhalten die Bewertungen beispielsweise Informationen wie Art und Lokalisation der bewerteten Daten und die Uhrzeit der Bewertung, so kann daraus ein Interessensprofil, ein Aufenthaltsprofil oder ein Aktivitätsprofil erstellt werden.
  Ein Bewerter will also so wenig Informationen preisgeben wie möglich, aber so viele, wie nötig sind, um andere von seiner Glaubwürdigkeit zu überzeugen. Die Regulierung, wieviel und welche Informationen ein Bewerter über sich preisgeben will, sollte am besten vom Bewerter selbst durchgeführt werden.
  Dabei ist im Sinne der Zurechenbarkeit darauf zu achten, dass ein Bewerter nicht die Identität eines anderen Bewerters vortäuschen kann.

2.4 BEWERTUNGSPARAMETER

2.4.1 Bewerter

Um eine Bewertung durchzuführen, braucht es zunächst jemanden, der dies tut. Es besteht durchaus die Möglichkeit, an den Bewerter gewisse Bedingungen zu stellen, wie zum Beispiel die Anmeldung bei Openstreetmap oder eine bestimmte Mindestanzahl gemappter Objekte. Dies würde eine Vorauswahl treffen und die Missbrauchsschwelle anheben.
Im Gegensatz dazu lassen sich sicher deutlich mehr Bewerter finden, wenn keine Bedingungen gestellt werden und prinzipiell jeder eine Bewertung abgeben kann. Es ist leicht vorstellbar, dass solch ein System von Angreifern einfach ausgenutzt werden kann, indem automatische Bewertungen in großer Stückzahl generiert werden und somit großes Interesse mehrerer Bewerter vorgetäuscht wird. Um diese Missbrauchsmöglichkeit wiederum abzumildern, könnte den Bewertungen ein zusätzliches Erkennungsmerkmal hinzugefügt und daran eine unterschiedliche Gewichtung der Bewertungen vorgenommen werden. Bewerter beziehungsweise deren Bewertungen könnten beispielsweise durch andere Instanzen zertifiziert sein und eine Gewichtung aufgrund der verschiedenen Zertifikate vorgenommen werden.

2.4.2 Bewertungsgewicht

Wie oben bereits erwähnt, kann es durchaus sinnvoll sein, den abgegebenen Bewertungen eine unterschiedliche Bedeutung beizumessen. Dazu werden allerdings zusätzliche Informationen benötigt, die entweder der Bewertung direkt beigefügt sind oder zu dem Bewerter vorliegen, sofern dieser bekannt ist.
Wie in Abschnitt 2.2 erläutert, können diese Informationen aus einem zweiten Reputationssystem für die Bewerter resultieren. Beispielsweise dienen diese Informationen dazu, die Kompetenz eines Bewerters zu belegen, über das Bewertungsobjekt eine Aussage zu treffen. Sie können aber auch einen gewissen Status signalisieren, den sich der Bewerter zuvor erarbeiten musste. Dies könnte sehr einfach die Dauer der Mitgliedschaft bei Openstreetmap oder die Anzahl der dort eingetragenen Daten sein.
In dieser Arbeit soll allerdings keine genaue Aussage getroffen werden, welche Informationen wie in das Bewertungsgewicht einfließen. Vielmehr soll die Möglichkeit geschaffen werden, jede beliebige Information mit aufnehmen zu können und letztendlich dem Nutzer zu überlassen, welche Information er wie bewertet und welchen Wert eine vorliegende Bewertung somit für ihn hat.
Die Informationen sollen in Form von digitalen Zertifikaten direkt an den Bewertungen untergebracht sein. Somit ist es möglich, auch ohne den Bewerter zu identifizieren, das beiliegende Zertifikat einer Bewertung zu überprüfen und daraus ein Bewertungsgewicht abzuleiten.
Die zertifizierenden Instanzen können verschiedenartig sein und müssen nicht jede Bewertung einzeln zertifizieren. Sie könnten auch dem Bewerter ein Zertifikat ausstellen, das er seinen Bewertungen beilegt. Vorstellbar sind Zertifikate durch reale Institutionen, wie Ämter oder Vereine, oder geographische Zertifikate, die bestätigen, dass sich ein Bewerter in einem gewissen geographischen Bereich gut auskennt oder Zertifikate von OSM-Stammtischen, die einfach nur aussagen, dass die Person im OSM Umfeld bekannt ist und womöglich weniger Anfängerfehler macht oder einfach nur durch persönlichen Kontakt vertrauenswürdiger ist, und so weiter. Die zertifizierenden Institutionen können sich auch gegenseitig die Zertifikate bestätigen und somit eine Art Vertrauensnetzwerk schaffen.

2.4.3 Bewertungsart

Die Bewertungsart beschreibt grundsätzliche Eigenschaften einer Bewertung, welche inhaltliche Aussagen getroffen werden können und welche Struktur dafür notwendig ist. Einige verschiedene Eigenschaften von Bewertungsarten und deren Interpretationsmöglichkeiten seien im Folgenden genannt:

- nur positiv (etwas stimmt)
- nur negativ (etwas stimmt nicht)
- binär (etwas stimmt oder nicht)
- mehrdimensional als Bewertung mehrerer Eigenschaften (z.B. Name falsch geschrieben, Lage korrekt, Adresse fehlt, ...)
- Verteilen von Punkten (z.B. auf einer Skala von 1-10 bin ich mir mit 9 Punkten sicher, dass dort eine Autobahn ist)

Wie bereits in der Einleitung beschrieben, existieren für Openstreetmap schon Möglichkeiten Fehler zu melden, also eine nur negative Bewertung abzugeben. Problematisch ist die fehlende Systematik von Datenüberprüfungen, da die Feststellung von Fehlerfreiheit nicht vermerkt wird. Für Openstreetmap nützlicher ist ein System, welches auch positive Bewertungen zulässt. Ein Bewertungssystem mit Verteilung von Punkten könnte zwar in manchen Fällen etwas mehr Informationen beitragen, aber ist oftmals nicht notwendig. Bei Openstreetmap werden im Wesentlichen überprüfbare Fakten aus der Realität gesammelt, und entweder ist man sich bezüglich eines Faktums sicher, weil man den Abgleich mit der Realität vorgenommen hat, oder man ist sich nicht sicher und braucht auch keine Bewertung abzugeben.
Zur weiteren Vereinfachung des Systems ist es sogar ausreichend, ausschließlich positive Bewertungen vorzunehmen, also nur zu dokumentieren, dass ein Datum überprüft und für korrekt befunden wurde. Sollte bei der Überprüfung ein Fehler auffallen, so könnte der womöglich sofort korrigiert oder zumindest ein Openstreetbug an der entsprechenden Stelle gesetzt und in dessen Kommentarfeld beliebige Mängel aufgeführt werden. Eine Behandlung von negativen Bewertungen ist also für das einzuführende Bewertungssystem überflüssig.

2.4.4 Bewertungsobjekt

Die Organisation des Bewertungsobjektes, also der Dateneinheit von Openstreetmap, die mit einer Bewertung abgedeckt wird, ist auf verschiedenen Ebenen denkbar. Diese sind:

- Kartenausschnitt (z.B. Campusgelände der TU in Dresden)
- Objektkategorien eines Ausschnitts (z.B. alle Straßen eines Dorfes oder alle Autobahnen in Deutschland...)
- Objektebene (z.B. ein bestimmter Autobahnabschnitt mit Geschwindigkeitsangaben, Anzahl der Fahrspuren etc.)
- Tagebene (jedes Tag eines Objektes einzeln, z.B. Höchstgeschwindigkeit der Autobahn)

Bei der Bewertung von kompletten Kartenausschnitten, also definierten geographischen Bereichen, ist die Granularität am gröbsten. Ein großer Vorteil dieses Verfahrens liegt in der schnellen Bewertung pro Fläche. Allerdings lässt sich durch die Bewertungen hier höchstens eine Tendenz der Korrektheit der Daten in dem entsprechenden Ausschnitt ableiten. Kleinere Fehler könnten in einem gut gemappten Gebiet entweder untergehen oder zu einer schlechten Bewertung des ganzen Ausschnitts beitragen. Beides wäre wenig hilfreich, da der Fehler nicht genau lokalisiert werden kann und Korrekturen demzufolge nicht wesentlich erleichtert wären. Im ersten Fall kann nicht mal ein Fehler festgestellt werden, und im zweiten Fall werden zunächst auch die korrekten Daten als potenziell fehlerbehaftet angesehen. Im Falle eines Campusgeländes, das wegen der fehlenden Eintragung eines Fakultätsgebäude schlecht bewertet ist, auch wenn die Straßen und Wege alle korrekt erfasst sind, würde diese Bewertung einen Nutzer, der nur die Wege benutzen möchte, verwirren.
Für den Bewerter ergibt sich zudem das Problem, dass er mit großer Wahrscheinlichkeit nicht für alle eingetragenen Dinge in dem Bereich gleichermaßen kompetent genug ist, um die Richtigkeit einschätzen zu können. Die Aussagekraft der Bewertung kann somit auch stark schwanken.

Ein wenig eingrenzen lassen sich die Nachteile, wenn statt einem kompletten Ausschnitt nur bestimmte Kategorien von Objekten in einem Ausschnitt betrachtet werden. Ein Bewerter könnte je nach Interessenlage und Kompetenz verschiedene Kategorien bewerten. Ein Briefträger könnte beispielsweise die Richtigkeit der Adressen und Briefkästen eines Ausschnitts beurteilen, während er über die Grünflächen keine Aussage treffen möchte. Mögliche Fehler sind auf eine Kategorie beschränkt und somit für Mapper leichter zu finden.
Eine feinere Beurteilung der Qualität lässt sich durch die Bewertung jedes Objektes einzeln durchführen. Dies ist zwar verhältnismäßig aufwändig, aber lässt eine genauere Fehlererkennung zu.

Ein großes Problem für alle drei bisher genannten Bewertungsebenen ist, dass es im Falle von Erweiterungen der Openstreetmap-Daten zu Schwierigkeiten mit der Gültigkeit der Bewertung kommen kann. Die Bewertung müsste nicht nur einen Bereich oder ein Objekt umfassen, sondern zusätzlich die Versionsgeschichte berücksichtigen. Für einen Bereich wird das sehr aufwändig. Für ein einzelnes Objekt zwar noch machbar, aber unpraktikabel. Zudem nützt es dem Auswerter nicht sehr viel, wenn beispielsweise die Autobahn in einer früheren Version gültig war und er sich zunächst alle Änderungen seit diesem Zeitpunkt anschauen muss, um zu einem Schluss zu kommen.

Um diesem Problem aus dem Weg zu gehen, ist es sinnvoll, eine Bewertung auf Grundlage der kleinsten Dateneinheit von Openstreetmap, den Tags und Nodes bzw. Relations eines Objektes, durchzuführen. Diese Dateneinheiten werden in dieser Arbeit fortan als semantische Einheiten bezeichnet.
Ein Bewerter kann die einzelnen semantischen Einheiten eines Objektes separat bewerten. Der Zusammenhang mit dem Objekt darf dabei aber nicht verloren gehen, da die Bewertung ja nur für eine bestimmte Einheit dieses Objektes und keines anderen gilt. Dem Objekt können unabhängig von den bereits bewerteten Einheiten neue semantische Einheiten, wie Tags hinzugefügt werden. Die Historie eines Objektes spielt nur dann eine Rolle, wenn bereits bewertete Einheiten verändert oder entfernt werden.

Wesentlich für die Daten von Openstreetmap sind neben den Objekteigenschaften die Geoinformationen. Sie dienen als Kontext für die Bewertung der restlichen Objekteigenschaften. Als Beispiel diene die Waldschlösschenbrücke in Dresden, die sich derzeit noch in der Bauphase befindet. Sie ist in Openstreetmap erfasst und enthält unter anderem Eigenschaften, die sie als Baustelle auszeichnen. Wären diese Eigenschaften durch Bewerter bestätigt, ohne auf die geographische Lage der Brücke zu achten, so ließe sich die Brücke an eine völlig andere Stelle bewegen und die Eigenschaften behielten ihre Gültigkeit. Die Frage, ob die bewerteten Eigenschaften des Objektes aufgrund der neuen Position als falsch anzusehen sind, ist die Frage nach der Interpretation der Bewertungsaussage. Eine Aussage der Form: „Es gibt eine Brücke namens Waldschlösschenbrücke, die im Bau ist und sich irgendwo auf der Welt befindet." ist viel schwächer als: „Es gibt eine Brücke namens Waldschlösschenbrücke, die im Bau ist und sich in Dresden

an der Elbe befindet.", wobei Dresden an der Elbe mit der genauen Position in Form der Koordinaten ersetzt werden könnte.
Möchte man also die zweite stärkere Aussage generieren, so muss der geographische Kontext bei der Bewertungsabgabe berücksichtigt werden.
Im Unterschied zu den durch Tags gesetzten Objekteigenschaften sind die Objektgeometrien, also die Positionsangaben auf der Erde, nicht exakt feststellbar, denn Positionsmessgeräte wie zum Beispiel GPS-Geräte besitzen nur eine begrenzte Genauigkeit. Die Geometrie von Objekten wird deshalb in Openstreetmap häufig verändert, wenn genauere Positionsdaten vorliegen oder einfach nur unvollständige Daten verbessert oder ergänzt werden z.B. bei Anschluss eines neuen Weges an ein bestehendes Wegenetz.
Wäre die Bewertung einer Objekteigenschaft nun abhängig von der zu diesem Zeitpunkt gesetzten Geometrie des Objektes, so würde jede kleinste Änderung an der Geometrie die Bewertung der Objekteigenschaft invalidieren. Wird die Geometrie nicht in die Bewertung mit einbezogen, kann man entweder, wie oben bereits genannt, nur schwächere Aussagen treffen oder man stellt den Zusammenhang über den Zeitpunkt der Bewertung her und vergleicht die zu diesem Zeitpunkt vorhandene Geometrie aus der Historie des Objektes, was wiederum erhöhten Aufwand bedeuten würde.
Ein Kompromiss ist, die Geometrie separat, aber zum gleichen Zeitpunkt zu bewerten, wie eine Objekteigenschaft. Änderungen der Geometrie würden zwar auffallen, aber noch nicht automatisch die Bewertung einer Eigenschaft invalidieren. Die Entscheidung, ob die Geometrie bewertet werden soll, kann jedoch weiter dem Nutzer überlassen werden. Wenn dieser den geographischen Kontext in einem konkreten Fall für unwichtig hält, soll er nicht gezwungen werden diesen einzuschätzen.
Um der Unschärfe der Positionsdaten von Objekten gerecht zu werden, könnten auch die Bewertungen nur unscharfe Aussagen zulassen. So könnten Koordinaten nur bis zu einer bestimmten Anzahl an Stellen, also einer gewissen Genauigkeit bewertet werden. Alternativ könnten der Bewertung auch zusätzliche Angaben, wie ein Gültigkeitsbereich hinzugefügt werden, was im Beispiel 2.3 verdeutlicht wird. Möglich ist auch, dass der Auswertende festlegt, welche Abweichungen von der ursprünglichen Koordinate er noch für akzeptabel hält. Er hat dabei jedoch keine Information über die Einschätzung der Genauigkeit durch den Bewerter und kann daher nur allgemeine Annahmen treffen.

Beispiel 2.3 (Bushaltestelle) *Ein Bewerter möchte die Lage einer in OSM erfassten Bushaltestelle bestätigen. Beim Warten auf den Bus hat er die Koordinaten mit seinem GPS-Gerät nachgeprüft. Das Gerät wies bei der Erfassung eine mögliche Ungenauigkeit von 5 Metern auf. Zu Hause angekommen, bewertet er die Eigenschaft „highway=bus_stop" des Objektes als richtig. Sofern noch keine Genauigkeitsaussagen über die Position vorliegen, kann er das Objekt auf seinen Messpunkt verschieben und anschließend die Position mit einer Toleranz von 5 Metern bestätigen. Liegen bereits Aussagen vor, kann er testen, ob diese seiner Messung widersprechen und gegebenenfalls vorangegangene Bewerter kontaktieren, um eine Einigung zu finden.*

2.4.5 Bewertungsaussage

Durch die Interpretation einer Bewertung können folgende Informationen gewonnen werden. Zu einem bestimmten Zeitpunkt bestätigt ein Bewerter, dass die bewertete Eigenschaft des Objektes tatsächlich in der Realität vorhanden ist und somit korrekt erfasst wurde. Liegen der Bewertung zusätzliche Informationen wie beispielsweise Genauigkeit der Geometrie, Quelle der Information oder Bewerterreputation bei, so kann ein Auswertender diese in seine Betrachtungen mit einbeziehen.

Ziel der Bewertung ist es, einen Auswertenden in der Einschätzung der Zuverlässigkeit der Openstreetmap-Daten zu unterstützen. Die vielfältigen Informationen lassen sich auf einen Zahlenwert, den Reputationswert abbilden, um die Auswertung zu vereinfachen.

2.5 TECHNISCHER AUFBAU VON OPENSTREETMAP

Um eine detailliertere Beschreibung des Bewertungsverfahrens vorzunehmen und auf Stärken und Schwächen des Systems eingehen zu können sowie ein passendes Angreifermodell zu entwickeln, ist es zunächst erforderlich, einen tieferen Einblick in den technischen Aufbau der Openstreetmap-Daten sowie deren Bearbeitung und dabei auftretende Fehlerquellen zu geben.

2.5.1 Datenmodell von Openstreetmap

Openstreetmap abstrahiert die Realität in einem speziellen Datenmodell, welches drei verschiedene Arten von Objekten kennt:

- Knoten (Nodes): Punktobjekte, die eine geographische Position besitzen
- Wege (Ways): Linienobjekte, die aus einer Liste von Knoten bestehen
- Relationen (Relations): Beziehungen zwischen Knoten, Wegen und/oder Relationen zur Modellierung komplexerer Objekte

Alle diese Objekte haben gemeinsam, dass sie mit beliebigen Tags ausgestattet werden können, aber nicht müssen. Tags bestehen aus Schlüssel-Wert-Paaren der Form *Key* = *Value*, die laut aktueller OSM-API[6] beliebige Unicode-Strings mit bis zu 255 Zeichen enthalten können.
Elemente von Relationen, sogenannte member können zusätzlich mit Rollen versehen werden.
Zu jedem Objekt in Openstreetmap sind weiterhin eine eindeutige Identifikationsnummer (OSM-ID), sowie eine Versionsnummer, der Zeitstempel der letzten Bearbeitung, der letzte Bearbeiter mit Bearbeiterkommentar und die komplette Historie des Objektes gespeichert. Die Openstreetmap-Identifikationsnummer, die bei jedem Objekt gespeichert ist, ist jedoch nur eindeutig bei gleichem Objekttyp. Um eine eindeutig Openstreetmap-Identifikationsnummer (OID) über alle Objekte zu erhalten, muss der Objekttyp mitkodiert werden.
Eine wesentliche Eigenschaft von Knoten und Wegen ist die Geometrie, welche die Lage des Objektes auf der Erdoberfläche beschreibt. Eine Knotengeometrie besteht aus einer geographischen Länge (Longitude) und einer geographischen Breite (Latitude), die bei Openstreetmap im internationalen geodätischen Referenzsystem World Geodetic System 1984 (WGS 84) abgespeichert werden.
Die Geometrie von Wegen wird über eine geordnete Liste von solchen Knoten beschrieben und ist aufgrund dieser Ordnung gerichtet.
Anders als Knoten und Wege besitzen Relationen keine eigene Geometrie. Ihre Lage wird nur durch die Lage ihrer Mitglieder bestimmt.

In Tabelle 2.2 sind die Tags des Gebäudes der Fakultät Informatik in Openstreetmap beispielhaft dargestellt. Das Fakultätsgebäude ist als Fläche erfasst, was in Openstreetmap bedeutet, dass es aus einem geschlossenen Weg besteht. Geschlossen bedeutet, dass der Startknoten gleich dem Endknoten ist. Dieser Weg besitzt spezielle Tags z.B. building=yes, die ihn als Fläche interpretierbar machen.
Neben der Liste von Tags, die hauptsächlich Adressinformationen beinhalten, existiert noch eine geordnete Liste mit Knoten, welche die Geometrie des Weges abbilden.
Die gesamten Daten des Objektes sind unter folgender URL komplett einzusehen:
http://www.openstreetmap.org/browse/way/23290301

Die erfassten Daten eines Objektes stellen im Wesentlichen eine Abbildung von Fakten aus der Realität dar. Allerdings gibt es bei bestimmten Eigenschaften Interpretationsspielraum. So lässt sich die genaue Geometrie eines Objektes aufgrund von Fehlern der Messtechnik nicht zweifelsfrei feststellen. Korrekturen im Nachhinein durch genauere Messungen kommen häufig vor.

6. OSM-API v0.6, http://wiki.openstreetmap.org/wiki/API_v0.6

Key	Value
addr:city	Dresden
addr:country	DE
addr:housenumber	46
addr:postcode	01187
addr:street	Nöthnitzer Straße
amenity	university
building	yes
layer	1
name	Fakultät Informatik (INF)

Tabelle 2.2: OSM Tagging Schema Beispiel: Fakultät Informatik der TU-Dresden

2.5.2 Editieren

Die Editierungen werden in Form von Changesets[7] seit API 0.6 auf den bestehenden Datenbestand angewendet. Dabei wird eine Reihe von Änderungen in einer Gruppe zusammengefasst, die zusätzliche Metainformationen, wie Autor, verwendeter Editor, Datenquelle, betreffendes Gebiet etc. enthält. Changesets können komplett rückgängig gemacht werden.
Neben dem technischen Vorgang führt ein Mapper inhaltlich beim Editieren von Openstreetmap-Daten im Wesentlichen zwei Aktionen durch:

- Ergänzung völlig neuer Objekte unabhängig von den bereits existierenden Daten
  (z.B. Gebäude, Points of Interests etc. eintragen)
- Korrigieren und Ergänzen der existierenden Daten
  (z.B. Hinzufügen von zusätzlichen Tags wie Straßenbreite, Adressen, Namen,... bzw. Anschluss von neuen Wegen an bereits bestehende oder Korrigieren von Geometrien aufgrund genauerer Daten)

Ein hierbei offensichtliches Problem ist, dass falsche Daten einfach eingegeben beziehungsweise bereits vorhandene, vermutlich korrekte Daten verfälscht werden können - beabsichtigt oder unbeabsichtigt. Die Unterscheidung in neu eingegebene und geänderte Daten ist in sofern sinnvoll, als dass im Gegensatz zu neuen Daten bereits existierende schon eine Bewertung erhalten haben könnten.
Ein Nutzer der freien Geodaten von Openstreetmap möchte gerne auf die Richtigkeit der verfügbaren Daten vertrauen können, zumindest in einem für ihn wichtigen Kartenausschnitt. Das bedeutet im bisherigen System, dass er einfach allen in dem interessierenden Bereich tätigen Mappern vertraut. Das können in einem Stadtgebiet beispielsweise sehr viele sein (z.B. in Dresden 200 - 300), die man sehr wahrscheinlich nicht alle persönlich kennt. Es gibt also zunächst einmal keine Vertrauensgrundlage.
Ziel eines Reputationssystems für Openstreetmap ist, die Qualität der Openstreetmap-Daten systematisch zu bewerten und die Vertrauenswürdigkeit der Objekte über die Reputation der bewertenden Menschen besser einschätzen zu können. Fehlerbeseitigungen könnten dadurch gezielter erfolgen. Die Fehler, die in Openstreetmap auftreten können, sollen in Abschnitt 2.5.3 zunächst genauer vorgestellt werden.

7. Änderungssätze (Changesets) bei OSM, http://wiki.openstreetmap.org/wiki/DE:Changeset

2.5.3 Fehlerquellen bei Openstreetmap

Wenn Daten bei Openstreetmap neu erfasst bzw. bereits erfasste Daten verändert werden, so besteht immer die Möglichkeit, dass die resultierenden Daten fehlerhaft sind. Ein Reputationssystem für Openstreetmap kann solche Fehler nicht verhindern, möchte aber dazu beitragen, dass potentiell richtige Daten von potentiell falschen Daten mit wenig Aufwand unterschieden werden können.
Um herauszufinden, welche Daten als falsch angesehen werden müssen, ist es erforderlich, die möglichen Fehler und deren Ursachen genauer zu kennen. Damit das System möglichst effektiv arbeitet, soll das Hauptaugenmerk hierbei auf häufig auftretenden Fehlern liegen. Einige Fehlerquellen ohne Anspruch auf Vollständigkeit sollen deshalb kurz genannt und entsprechend ihrer Auftrittshäufigkeit getrennt werden. Zudem wird die erfahrungsgemäße Erkennung dieser Fehlerart kurz kommentiert.

- häufig auftretende Fehler
  - unzureichendes Verständnis und zu wenig Erfahrung mit Openstreetmap oder Fehlbedienung des Editors (meist Anfängerproblem, für erfahrene User oft erkennbar)
  - Verwendung schlechter Ausgangsdaten, aufgrund von schlechtem GPS-Empfang, verzerrten Luftbildern, Fehlinformationen und weiteren unerwarteten Störquellen (nicht so einfach festzustellen und oft strittig)
- seltene Fehler
  - Fehlfunktion der Editiersoftware (Fehler oft gut automatisiert zu beheben, da eventuell erkennbares Fehlermuster)
  - Vandalismus (je nachdem, wie geschickt durchgeführt, leichter oder schwerer zu erkennen und zu beheben)
  - Daten verlieren plötzlich an Gültigkeit, da reales Bezugsobjekt verändert wird z.B. durch Straßensperrungen, Verkehrsumstellung, Ladenschließung, Häuserabriss, etc. (je nach gesellschaftlicher Relevanz wird Änderung früher oder später bemerkt und Daten aktualisiert)

Ein Fehler aufgrund von unzureichendem Verständnis oder kurz gesagt durch Nutzerfehler tritt deshalb häufig auf, weil er prinzipiell alle beteiligten Mapper betreffen kann. Vor allem Anfänger sind mit der Komplexität von Openstreetmap schnell überfordert und erzeugen unbewusst fehlerhafte Daten. Viele dieser Fehler wie zum Beispiel eine falsche Straßenkategorie (Autobahn statt Fußweg) oder nicht verbundene Wege fallen erfahrenen Nutzern auf, sofern sie die Daten überprüfen oder werden von vielen der zahlreichen Werkzeuge zur Qualitätssicherung[8] automatisch erkannt.
Aber auch erfahrene Nutzer machen Fehler, wenn auch nicht so häufig, denn Irren ist menschlich. So könnten beispielsweise Absprachen über Eintragungen, die ein Großteil der Community getroffen hat, nicht alle Mapper erreichen, woraufhin entgegen dieser anerkannten Absprachen editiert werden könnte. Meistens sind jedoch die erfahrensten aktiven Mapper auch in den Kommunikationsnetzwerken der Community aktiv.
Wichtig, um solche Fehler zu bekämpfen, ist eine kurze systematische Nachkontrolle der Daten vor allem bei Anfängern. Das hier entwickelte System ermöglicht solch eine Systematik.

Neben den menschlichen Fehlern sind die zufälligen und systematischen Fehler der Messgeräte und Ausgangsdaten eine weitere häufige Ursache von Fehlern, die jedoch häufig die Geometrie und weniger die Tags der Daten betreffen. Zufällige Fehler, also solche, die sich bei wiederholten Messungen im Mittel aufheben, sind häufig bei GPS-Geräten vorzufinden. Sie sind beispielsweise von der Position der GPS-Satelliten von der Umgebung oder vom Wetter abhängig, wie eine

8. Qualitätssicherung, http://wiki.openstreetmap.org/wiki/Qualit%C3%A4tssicherung

Studie von M. Modsching in Görlitz zeigt [MKtH06]. Der Messfehler wird vor allem bei Hindernissen in der Umgebung, wie Schluchten, Häuserwänden, Wäldern oder Bergkuppen größer. Da diese Hindernisse nicht zufällig sind, treten hierbei auch systematische Fehler auf, das heißt, dass beispielsweise neben Hauswänden bestimmte Fehlertendenzen über mehrere Messungen nicht im Mittel aufgehoben werden können.
Andere Datenquellen wie digitale Orthophotos, also georeferenzierte Luftaufnahmen, können auch Verzerrungen enthalten, je nachdem wieviel Aufwand bei der Herstellung betrieben wurde. Da viele Luftbilder, vor allem die sehr genau ausgerichteten, für Openstreetmap aus rechtlichen Gründen als Quelle nicht zur Verfügung stehen, können fehlerhafte Positionsdaten aus den wenigen verbliebenen, gelegentlich fehlerhaften aber verwendbaren Luftbildern, wie beispielsweise Yahoo!-Luftbildern[9] resultieren. Die Luftbildgenauigkeit ist in den meisten Fällen dennoch deutlich besser als die von gebräuchlichen GPS-Geräten.
Bei Fehlern dieser Art könnte das Bewertungssystem in sofern helfen, dass eine geschätzte Genauigkeit der Daten bei der Datenkontrolle mit angegeben wird, um so eine eventuelle Verschlechterung im Nachhinein zu vermeiden.

Treten Fehler in der Editiersoftware auf, so kann dies auch zu Datenfehlern in Openstreetmap führen. Diese Fehler sind deswegen selten, weil die Software in der Regel getestet wird, bevor ein größerer Nutzerkreis sie verwendet. Zudem werden Fehler, die auf diese Art entstehen, in der Regel schnell bemerkt, und ihre Ursache kann im Gegensatz zu menschlichen Fehlern oder Messfehlern schnell und gründlich beseitigt werden. Ein Bewertungssystem hilft hierbei nicht wirklich viel.

Problematischer ist gezielter Vandalismus[10]. Da dieser erfahrungsgemäß nur von wenigen Leuten ausgeht, ist auch hier von seltenen Fehlern auszugehen. Selbstverständlich können wenige Leute auch viel zerstören, aber größere Verfälschungen der Daten fallen erstens schneller auf und besitzen zweitens häufig erkennbare Fehlermuster, z.B. ein gleiches Changeset, so dass sie recht einfach rückgängig gemacht werden können, indem eine ältere Version der OSM-Historie wieder hergestellt wird.
Gegen geschickt gemachten Vandalismus in kleinem Maßstab, kann sich das Projekt nur mit systematischen Kontrollen der eingetragenen Daten schützen. Das ist unter anderem ein Hauptanliegen des Bewertungssystems dieser Arbeit.

Eine schwierige Problematik, die später in Abschnitt 2.6 noch ausführlicher erläutert wird, besteht darin, dass die Openstreetmap-Daten dynamisch sein müssen, da sich die zu Grunde liegende Realität verändern kann. Dagegen bietet das Bewertungssystem keinen Schutz, aber es muss mit dieser Situation umgehen können.

2.5.4 Fehlerquellen bei der Bewertung der Daten

Nicht nur beim Erfassen und Eintragen der Daten können Fehler auftreten, sondern auch beim Bewerten dieser. So muss ein Bewerter erst einmal in der Lage sein, die Informationen nachzuprüfen und dies auch tun. Der Vorteil von Openstreetmap ist die Existenz einer überprüfbaren Faktenbasis (groundtruth). Diese kann mit Sinnesorganen und Messgeräten wahrgenommen werden. Fehler sind nicht ausgeschlossen, aber die Daten können in den meisten Fällen dennoch mit genügend Aufwand validiert werden.
Schwieriger wird es, wenn Daten aus Drittquellen hinzugefügt werden. So werden in Openstreetmap beispielsweise auch Kontaktinformationen öffentlicher Einrichtungen, wie Telefonnummern, E-Mail- und Webadressen oder Öffnungszeiten erfasst. Stammen diese beispielsweise aus einer durch Onlinerecherche gefundenen Drittquelle und nicht vom Betreiber selbst, so ist hier eine neue fehleranfällige Ebene eingebracht. Ist die Drittquelle womöglich sogar die einzige Quelle der Information, so könnten die Bewerter möglicherweise alle die selbe falsche Quelle verwenden

9. Yahoo!-Luftbild Genauigkeit, http://wiki.openstreetmap.org/wiki/Yahoo!_Aerial_Imagery/Accuracy
10 Vandalismus bei OSM, http://wiki.openstreetmap.org/wiki/Vandalism

und somit falsche Informationen als richtig bewerten.
Es wäre also sinnvoll bei einer Bewertung auch die Art der Prüfung bzw. die Quelle der Information mit angeben zu können.

Eine weitere Fehlerquelle bei der Bewertung der Daten kann dadurch zustande kommen, dass der Bewerter nicht die aktuellen Openstreetmap-Daten als Grundlage nimmt. Möglicherweise bewertet er einen alten Datenstand und ist sich darüber nicht im Klaren. Eine Bewertung älterer Daten trotz Kenntnis von Aktualisierungen ist hingegen nicht sinnvoll. Ist ein Bewerter der Ansicht, dass das aktuelle Objekt in Openstreetmap nicht korrekt ist, jedoch eines aus der Historie, so wird er das ältere Objekt wieder herstellen und anschließend möglicherweise bewerten. Die Bewertung älterer Daten ohne Wiederherstellung dieser hilft einem Nutzer von Openstreetmap, der Informationen über die aktuelle Karte erhalten will, nicht weiter.

2.5.5 Korrektheit der Openstreetmap-Daten

Bisher wurde der Blick nur auf Fehler der Openstreetmap-Daten gerichtet ohne zu definieren, wann Openstreetmap-Daten als korrekt gelten können.
Das ist keineswegs immer sehr einfach festzustellen, da es durchaus unterschiedliche Möglichkeiten gibt, die Realität in Openstreetmap abzubilden. Bei meiner Definition von „korrekte Daten“ sollen aber alle diese Möglichkeiten unter gewissen Bedingungen durchaus gültig sein.
Openstreetmap arbeitet wie jede andere Karte auch mit einer Abstraktion der Realität. Dabei werden wesentliche Fakten extrahiert und entsprechend der OSM-Datenstruktur aufbereitet. Die Art und Weise, wie dies geschieht, wurde zuvor meistens von den Mappern diskutiert und im Openstreetmap-Wiki[11] festgehalten. Allerdings sind die dort aufgeführten Taggingschemas nur eine Richtlinie und können durch eigene Taggingmodelle erweitert werden. Falls es inhaltliche Kollisionen mit bereits bestehenden Tags gibt, sollten diese ausdiskutiert werden und das Ergebnis wiederum im Wiki notiert werden.
Die Map-Features-Seite im OSM-Wiki ist Grundlage für fast alle Anwendungen von Openstreetmap und hat somit genug Bedeutung, um als Grundlage für die Definition von „korrekten OSM-Daten“ zu dienen.

Nehmen wir an, es gibt eine Menge R von Fakten in einem bestimmten geographischen Bereich der Realität. Die Größe des gewählten Bereichs entspricht hierbei der akzeptierten geographischen Genauigkeit der Openstreetmap-Daten, also in der Praxis ein zweistelliger Meterbereich[12].
Nach der Abstraktion bleibt nur eine Teilmenge von Fakten $A \subseteq R$ daraus übrig. Weiterhin sei D die Menge aller im gleichen Bereich vorhandenen Openstreetmap-Daten.
Aus dem OSM-Wiki lassen sich offizielle Abbildungen o entnehmen die den Openstreetmap-Daten abstrahierte Fakten zuordnen: $o : D \to A$.
Diese Zuordnung sollte eine Funktion sein. Ist die Zuordnung mehrdeutig, so ist nicht klar, was mit dem entsprechenden Openstreetmapdatum eigentlich in der Realität gemeint ist. Zudem wäre es wünschenswert, wenn die Funktion injektiv ist, denn das bedeutet, dass es ein einheitliches Taggingschema gibt, an das sich alle Mapper halten. Dies wird allerdings in einem Projekt wie Openstreetmap nicht erzwungen, damit genug Spielraum zum ausprobieren neuer Tags bleibt.
Zusätzlich zu den dem Wiki entnehmbaren Abbildungen o existieren weitere inoffizielle Abbildungen i, die jedoch nur dann gültig sein sollen, wenn das Urbild der Menge A unter der Funktion o keine Gemeinsamkeiten mit dem Urbild der Menge A unter der Funktion i hat:

$$o^{-1}(A) \cap i^{-1}(A) = \emptyset$$

Die im Wiki festgelegte Bedeutung der vorhandenen Daten soll also durch eigene Abbildungen nicht verändert werden. Ein vorhandenes Openstreetmapdatum d soll dann korrekt sein, wenn

$$o(d) = a \vee i(d) = a \text{ mit } d \in D; a \in A$$

gilt.
In Worte gefasst muss es eine Abstraktion der Realität des geographischen Bereiches geben, in welchem das OSM-Datum sich befindet, so dass sich das Datum diesem abstrakten Faktum zuordnen lässt - entweder mithilfe der Regeln aus dem Wiki oder mit eigenen erweiterten Regeln, die mit ersteren nicht kollidieren dürfen.

11. OSM-Wiki: Map Features, http://wiki.openstreetmap.org/wiki/DE:Map_Features
12 GPS-Genauigkeit, http://wiki.openstreetmap.org/wiki/DE:Genauigkeit_von_GPS-Daten

2.6 DYNAMIK DER OPENSTREETMAP-DATEN

Openstreetmap-Daten sind nicht statisch und ändern sich über die Zeit. Für jedes OSM-Objekt existiert eine eigene Historie. Bei jedem neuen Upload einer geänderten Version eines Objektes zum OSM-Server wird die Versionsnummer des Objektes um 1 erhöht. Jedes Objekt beginnt mit der Versionsnummer 1 mit der es angelegt wird. Eine Änderung eines Objektes bedeutet ein Attribut oder ein Teil der Geometrie wurde verändert, gelöscht oder neu hinzugefügt.
Um einen groben Überblick über die Änderungshäufigkeit von OSM Objekten zu bekommen, lässt sich die durchschnittliche Versionsnummer für Objekte ermitteln. Anfang Oktober 2010 ist die durchschnittliche Versionsnummer der Objekte für ein planet file bzw. ein Deutschlandauszug daraus wie folgt:

	Node	Way	Relation
Planet	1,470	1,672	3,573
Deutschland	1,857	2,510	7,630

Quelle: http://www.h-renrew.de/h/osm/osmchecks/

Das Auftreten von Änderungen kann regional sehr verschieden sein. Da Deutschland im Vergleich mit anderen Ländern schon sehr gut erfasst ist und eine sehr aktive Openstreetmap-Community besitzt, treten hier häufiger Änderungen auf als im weltweiten Durchschnitt. Je mehr Geodaten bereits erfasst sind, desto mehr konzentriert sich die Arbeit auf das Aktualisieren und Verbessern dieser.
Leider lässt sich aus der Versionsnummer eines Objektes allein noch nicht ableiten, ob bzw. welche Eigenschaften des Objektes eigentlich verändert wurden. Statistische Informationen darüber ließen sich nur aus einer tieferen Analyse der kompletten Historie jedes Objektes in Openstreetmap erhalten. Jede Version müsste mit ihrer Vorgängerversion verglichen werden, um die Änderungen zu bestimmen.
Doch selbst, wenn genaue Statistiken über die Änderungen der einzelnen Daten zur Verfügung ständen, so ließe sich noch nicht feststellen aus welchem Grund diese geändert wurden.

2.6.1 Erwünschte Änderungen

Die Änderung von einmal eingetragene Daten kann aus verschiedenen Gründen gewollt sein:

- **Änderungsgrund:** Die Realität hat sich geändert und die Daten sind somit veraltet.
  Beispiel: Ein in OSM erfasstes Haus wird abgerissen.
- **Änderungsgrund:** Die Abbildung der Realität hat sich geändert. Die Daten sollen einem neuen Taggingschema angepasst werden.
  Beispiel: Poller wurden früher mit "highway=bollard" getaggt. Mittlerweile existiert der key "barrier" und somit ist "barrier=bollard" zu verwenden und "highway=bollard" zu entfernen.
- **Änderungsgrund:** Die Erkenntnisse über die Realität haben sich geändert. Die früher eingetragenen Daten werden als mangelhaft erkannt.
  Beispiel: "highway=road" kann verwendet werden, um anzuzeigen, dass eine Strasse vorhanden ist ohne sie genau einem Typ zuordnen zu können. Stellt sich später heraus, dass es sich beispielsweise um eine Straße in einem Wohngebiet handelt, wird das Attribut zu "highway=residential" geändert.

- **Änderungsgrund:** Informationen sollen ergänzt werden und beeinflussen bereits bestehende Daten.
  Beispiel: Zu einer Straße mit zwei Knotenpunkten soll in der Mitte eine Seitenstraße ergänzt werden. Es wird mittig ein Anschlussknotenpunkt hinzugefügt, an den die Seitenstraße angeschlossen wird. Die ursprünglich vorhandene Straße besteht nun aus drei Knotenpunkten.

Bei den vorgestellten Änderungsgründen ist zu bedenken, dass diese dann akzeptabel für einen Nutzer sein können, wenn sie sich aus einem Konsens der Community heraus ergeben. Dabei ist es vergleichsweise einfach über Fakten in der Realität einen Konsens zu finden, da diese von jedermann nachgeprüft werden können. Deren Abbildung in OSM-Daten ist da oftmals wesentlich strittiger. Gibt es verschiedene Ansichten, wie die Informationen der Realität eingetragen werden sollen und bestehen die Verfechter dieser Ansichten auf ihre Art die Daten einzutragen, so kann die Situation im schlimmsten Fall in einem sogenannten Edit-War[13] enden, indem jede Seite so schnell wie möglich die Änderungen der anderen Seite erneut nach den eigenen Ansichten ändert. Solche Änderungen gelten als Vandalismus und sind damit unerwünscht. Weitere Formen des Vandalismus wären willkürliche oder gezielte Löschungen oder Änderungen von korrekten Informationen.

Ist die Motivation eines Mappers grundsätzlich die eine erwünschte Änderung vorzunehmen, so heißt das leider noch nicht, dass das Resultat tatsächlich eine gewünschte Änderung ist. Einige der in Abschnitt 2.5.3 genannten ungewollten Fehler können dennoch auftreten. Eine Überprüfung der geänderten Daten muss demzufolge ohnehin stattfinden. Somit lassen sich im Nachhinein durch Bewerter Anhaltspunkte erzeugen, ob eine Änderung gewollt oder eher ungewollt war und entsprechende Maßnahmen, wie erneute Änderungen bzw. Fehlermarkierungen ergreifen.

2.6.2 Umgang mit Datenänderungen

Ein Signatursystem dient in der Regel dazu Veränderungen an signierten Daten anzuzeigen. Es ist leider relativ aussichtslos für ein solches System erwünschte (siehe Abschnitt 2.6.1) von unerwünschten (siehe Abschnitt 2.5.3) Änderungen automatisch zu unterscheiden. Dennoch benötigt das System eine Strategie, wie es mit geänderten Daten verfährt zu denen Signaturen vorliegen. Dazu muss eine Abwägung zwischen älteren geprüften Daten und neueren ungeprüften Daten stattfinden.

Folgende Möglichkeiten mit Signaturen von alten Daten umzugehen sind vorstellbar:

- **Verwerfen:** Alle Signaturen, die sich nicht auf die aktuelle Version des Objektes beziehen, werden ignoriert.
- **Negieren:** Signaturen eines älteren Datums werden als negative Bewertung des aktualisierten Datums angesehen.
- **Vergleichen:** Die Signaturen des neueren Datums und die des älteren Datums bilden jeweils einen Reputationswert. Der höhere Reputationswert zeigt an welchem Datum mehr vertraut werden kann.

Das Verwerfen älterer Signaturen ist eine einfach zu realisierende Variante, die allerdings keinen Schutz gegen Vandalismus bietet. Eine einzige Bearbeitung der Daten genügt und alle bisher gesammelten Bewertungen sind für die Einschätzung des geänderten aktuellen Datums bedeutungslos. Zwar wird das neue Datum hierbei zunächst keine positive Bewertung haben, bis es von Bewertern erneut überprüft wurde, allerdings ist es fragwürdig, ob das geänderte Datum

13 OSM-Wiki: Streitfälle, http://wiki.openstreetmap.org/wiki/Disputes

unabhängig vom früheren Reputationswert genau so angesehen werden sollte, wie ein neu angelegtes ohne Vorgeschichte.

Würde man Signaturen des älteren Datums negieren und als schlechte Bewertung des aktuellen Datums ansehen, so würden geänderte Objekte, falls sie signiert waren zunächst schlechter bewertet sein als neue Objekte. Dieser Variante liegt die Annahme zugrunde, dass gewollte Änderungen bei gut bewerteten Daten eher selten auftreten und Änderungen somit prinzipiell umso schlechter zu bewerten sind, je besser die vorherigen Daten bewertet waren. Zweifelhaftere Daten, die also gar nicht oder weniger gut bewertet sind, hätten demnach eine höhere Wahrscheinlichkeit, dass sie noch einmal verbessert, also geändert werden müssten.
Problematisch bei diesem Ansatz ist, dass sich die getroffenen Bewertungsaussagen auf das alte Datum beziehen und nicht auf das neue, auf dass sie aber dennoch negativ angewendet werden. Jemand der beispielsweise aussagt, dass sich momentan ein bestimmtes Geschäft an einer bestimmten Stelle befindet, sagt damit nicht automatisch aus, dass sich das Geschäft in einem Jahr immer noch dort befindet und kein anderes Geschäft.

Bewertungen beziehen sich also nicht nur auf ein bestimmtes Datenobjekt, sondern auch auf einen gewissen Zeitpunkt, also eine Version in der Historie des Objektes. Bleibt man bei der Frage, ob das aktuelle Datum als zuverlässig angesehen werden kann, so ist es eher schwierig aus den Bewertungen vergangener Daten nützliche Informationen diesbezüglich zu extrahieren. Erweitert man die Frage jedoch darauf, welche Version des Datums als zuverlässiger gesehen werden kann, so ist natürlich die Historie der Bewertungen sehr nützlich. Jede Version des Objektes könnte aufgrund der jeweils abgegebenen Bewertungen seinen eigenen Reputationswert bilden. Die Reputationswerte der verschiedenen Versionen könnten miteinander verglichen und somit die Version mit dem höchsten Reputationswert ermittelt werden. Um dem Problem der Veralterung der Aussagen gerecht zu werden, könnte ein Zeitfaktor eingeführt werden, der Reputationswerte älterer Versionen entsprechend der vergangenen Zeit seit dem Ändern der Version geringer gewichtet. Es sei an dieser Stelle noch einmal darauf hingewiesen, dass mit Version nicht die Versionsnummer eines Openstreetmap-Objektes gemeint ist, sondern die Version eines einzelnen Tags, die momentan nicht separat den Openstreetmap-Daten beiliegt.

Ein grundsätzliches Problem der letzten beiden Varianten, die in irgendeiner Form die Bewertungen älterer Daten mit einbeziehen, liegt jedoch darin begründet, dass die Zuordnung der Bewertungen zu den entsprechenden Openstreetmap-Daten nur über die eindeutige Identifikationsnummer von Openstreetmap-Objekten erfolgen kann. Wird ein Openstreetmap-Objekt einfach gelöscht und neu angelegt, geht diese Zuordnung verloren. Da jeder Mapper in der Lage ist Objekte zu löschen, könnte ein Vandalist dies auch tun, um seine Änderung eben nicht als Änderung, sondern als neu anlegen von Daten zu tarnen und somit ein unbewertetes neues Datum mit neuer Identifikationsnummer zu erzeugen.
Da die Openstreetmap-Datenbank auch gelöschte Objekte weiterhin speichert, kann ein Löschvorgang natürlich nachvollzogen werden. Nur die Verbindung vom alten zum neu angelegten Objekt kann nicht mehr einfach automatisiert hergestellt werden. Aufwändige Vergleiche von Objekten in einem bestimmten geografischen Bereich zum gelöschten Objekt wären nötig und könnten unter gewissen Plausibilitätsannahmen so etwas schaffen.
Das Löschen und anschließende neu Anlegen von Openstreetmap-Objekten muss seine Ursache jedoch nicht immer im Vandalismus haben. Nicht alle Mapper verstehen, wie die Daten in der Datenbank gespeichert werden und kennen den Wert einer Historie eines Objektes. So kommt es auch vor, dass ganz ohne böse Absicht Daten einfach gelöscht und neu angelegt werden. In manchen Fällen meinen bestimmte Mapper, dass es schneller und einfacher geht ein Objekt neu anzulegen, als an dem alten so lange Änderungen vorzunehmen, bis der gewünschte Zustand erreicht ist. Man stelle sich eine größere Fläche mit vielen Knoten vor, die aber alle nur ungenau platziert wurden. Es kann sich hier lohnen, die Fläche komplett neu zu zeichnen, als jeden einzelnen Punkt an eine bessere Position zu verschieben.
Es kann jedoch neben diesen pragmatischen Gründen noch andere Gründe geben, auch größere Gebiete zu löschen und sofort neu zum Server hochzuladen. Beispielsweise spielt es eine Rolle wer der erste Bearbeiter, also der Urheber eines Objektes ist. Dies kann vor allem bei Fragen von Lizenzänderungen eine Rolle spielen. Aber auch der Wettbewerb von Nutzern untereinander, die

darauf aus sind, möglichst viele Änderungen gemacht bzw. Daten beigetragen zu haben, kann zu einem solchen Verhalten führen.
Je nach Ansichtssache können diese Fälle natürlich auch als Vandalismus eingestuft werden.

Da es jedoch diese Möglichkeiten gibt, in die Historie von Objekten einzugreifen, ist der einfachste Ansatz mit Änderungen umzugehen, indem alte Bewertungen ignoriert werden, womöglich der zweckmäßigste. Auf einer Webseite oder sogar einer Karte könnte gezeigt werden, wo überall bzw. welche Objekte genau gelöscht oder geändert wurden und wieviel positive Bewertungen diese hatten. Löschen und Verändern kann in diesem Fall gleich aufgefasst werden.

2.7 ZUSAMMENFASSUNG ANALYSE

Reputationssysteme können das Vertrauen der Nutzer in das Kartenmaterial von Openstreetmap steigern, wie in Abschnitt 2.1 dargelegt wurde. Grundlage ist es Informationen von Dritten über die Gültigkeit der Daten an die Nutzer weiterzugeben.
In Abschnitt 2.2 wurde erläutert, dass es sinnvoll ist dafür zwei Reputationssysteme zu betrachten, eines trifft Aussagen über die Vertrauenswürdigkeit der Daten, also das eigentliche Ziel und eines über die Vertrauenswürdigkeit der Bewerter, die die Daten einschätzen.
Die Reputationsfunktion 2.2.1 der Systeme wurde dabei zunächst nicht näher festgelegt.
Abschnitt 2.2.2 erklärt, dass die Reputationssysteme einen zentralen Server benutzen sollen, aber nicht müssen. Sie brauchen nicht strikt zentral organisiert sein, sondern können auch dezentrale Elemente erlauben.
Die Reputation an sich soll im Sinne eines nutzerorientierten Reputationssystems durch aktive Bewertungsaussagen von Bewertern zu Stande kommen. Ein inhaltsorientiertes System wie WikiTrust ist für Openstreetmap weniger geeignet, wie Abschnitt 2.2.3 erläutert.
Die Funktionen, die das Bewertungssystem erfüllen soll, sind in Abschnitt 2.3.1 beschrieben. Dabei wird auf verschiedene Nebenbedingungen wert gelegt, wie Abschnitt 2.3.2 erklärt. Besondere Aufmerksamkeit erhalten die Sicherheitsinteressen der beteiligten Personen und deren Abwägung im Sinne des Konzeptes von mehrseitiger Sicherheit im Abschnitt 2.3.3.
Verschiedene Parameter des Systems, sowie Ansätze zu ihre konkreten Umsetzung werden in Abschnitt 2.4 gegeben.

Um die genaue Funktionsweise des Bewertungssystems verständlich machen zu können, ist ein grobes Verständnis der technischen Strukturen sowie der Erstellung und Bedeutung der Daten von Openstreetmap erforderlich. Abschnitt 2.5 enthält daher eine kurze Einführung in die Grundlagen von Openstreetmap. Dabei werden das Datenmodell in Abschnitt 2.5.1 und der Vorgang des Editierens in Abschnitt 2.5.2 vorgestellt. Fehler, die beim Editieren entstehen können und sich auf die Gültigkeit der Daten auswirken, werden in Abschnitt 2.5.3 genauer analysiert und diskutiert. Das Bewertungssystem kann mit den verschiedenen Fehlerquellen unterschiedlich umgehen und beim Finden der Fehler behilflich sein. Besonderer Wert wird dabei auf eher häufig auftretende Fehler gelegt. Abschnitt 2.5.5 definiert, was in dieser Arbeit überhaupt als Fehler der Openstreetmap-Daten betrachtet wird.
Das Problem des Bewertungssystems, dass sich die Bewertungsobjekte bzw. deren Gültigkeit dynamisch ändern können, wird in Abschnitt 2.6 aufgeworfen. Möglichkeiten, wie das System damit verfährt, werden in Abschnitt 2.6.2 aufgezeigt.

3 ENTWURF

Im folgenden Entwurf soll ein Bewertungssystem vorgestellt werden, welches die in Abschnitt 2.3 genannten Anforderungen weitestgehend erfüllt.

3.1 SYSTEMKOMPONENTEN

Um die Bewertungen der Openstreetmap-Daten vorzunehmen oder auszuwerten bekommen die Benutzer eine Anwendung zur Hand, die für diese Zwecke entwickelt wird. Diese Anwendung ist als optionale Zusatzanwendung für die Benutzung von Openstreetmap konzipiert.

Abbildung 3.1 veranschaulicht alle Komponenten des Systems, die im Folgenden ausführlich diskutiert werden.
Der Zugriff auf die Openstreetmap-Daten, sowie die graphische Ausgabe an den Nutzer erfolgt mithilfe eines bereits existierenden Openstreetmapeditors. Einer der bekanntesten ist derzeit JOSM. Dieser verfügt über eine Pluginschnittstelle, die es ermöglicht die zusätzliche Anwendung zur Benutzung des Bewertungssystems als Reputationsplugin zu integrieren.

Das Reputationsplugin verbindet die Funktionalitäten der Systemkomponenten und sichert die Anbindung an den OSM Editor.

Welche Funktionen die einzelnen Komponenten erfüllen, stellt Tabelle 3.1 kurz dar.

3.2 OSM SERVER

Der Openstreetmapserver speichert alle Openstreetmap-Datenobjekte zentral einschließlich ihrer Objekthistorien und Metadaten in einer Datenbank ab. Über eine Programmierschnittstelle (API) können die Daten direkt abgefragt bzw. aktualisiert werden. Das Reputationssystem soll stets mit den aktuellen Openstreetmap-Daten arbeiten und somit sollten die Daten direkt vom OSM Server geladen werden können, bevor sie möglichst zeitnah bewertet werden. Das Reputationssystem kann prinzipiell auch mit älteren Datenständen arbeiten. Hierbei könnten jedoch Konflikte entstehen, wenn Daten bewertet werden, die längst nicht mehr aktuell sind. Denn aktuell ist die Version eines Objektes, die zuletzt auf dem Server bearbeitet wurde. Die Problematik bei der Bewertung älterer Datenstände ist in Abschnitt 2.5.4 bereits beschrieben.

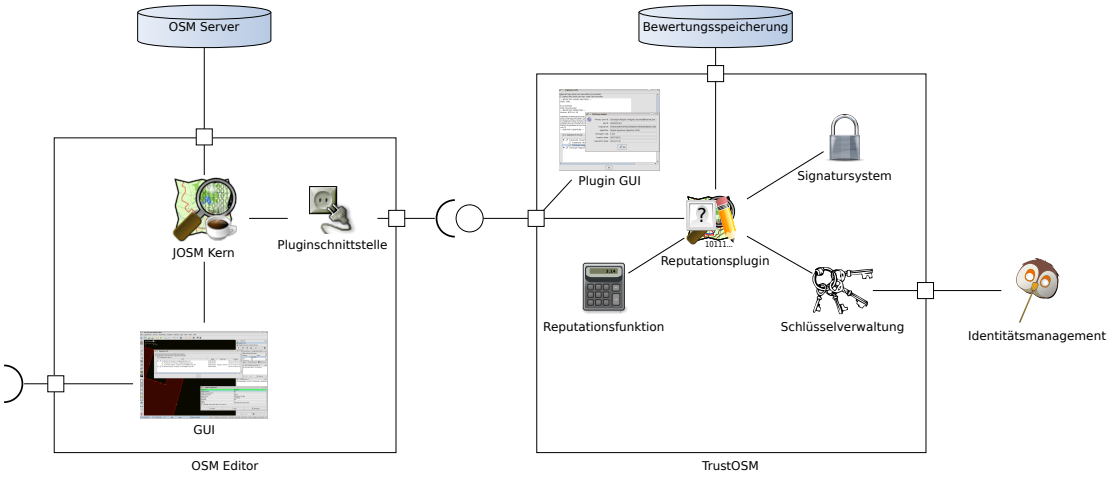

Abbildung 3.1: Komponenten des Bewertungssystems

Komponente	Funktion
OSM Server	Speicherung der OSM Daten
OSM Editor	graphische Bearbeitung und Auswertung der OSM-Daten
Reputationsplugin	Steuerung und Anbindung des Bewertungssystems
Plugin GUI	graphische Aufbereitung und Bedienung der Anwendung
Reputationsfunktion	Auswertung verfügbarer Bewertungen und Zusatzinformationen
Signatursystem	Erzeugen und Prüfen digitaler Signaturen von OSM-Daten
Schlüsselverwaltung	Erzeugung, Verbreitung und Speicherung von Schlüsseln
Identitätsmanagement	Benutzung verschiedener Bewerteridentitäten
Bewertungsspeicherung	Speicherung und Verteilung der erstellten Bewertungen

Tabelle 3.1: Kurze Übersicht über die Bedeutung der Systemkomponenten der Systemkomponenten

Um stets die aktuellen Daten vom Server abrufen zu können, muss die Anwendung mit dem Server kommunizieren können. Für diesen Teil der Anwendung werden jedoch bereits existierende Funktionalitäten des OSM Editors verwendet.

Der OSM Server hat eine weitere für das Bewertungssystem bedeutsame Aufgabe. Er vergibt die eindeutigen Referenznummern der Openstreetmap-Objekte (OID). Objekte, die in einem offline arbeitenden OSM Editor neu angelegt werden erhalten erst nach dem Hochladen zum Server eine neue OID. Wie im Abschnitt 3.3 noch genauer beschrieben wird, ist die Identifikationsnummer notwendig, um die abgegebenen Bewertungen einem Openstreetmap-Objekt bzw. dessen Eigenschaften zuzuordnen.
Dies hat technisch zur Folge, dass offline angelegte Objekte nicht sofort bewertet werden können. Sie müssen erst eine Identifikationsnummer vom OSM Server erhalten.

3.3 SIGNATURSYSTEM

Die Abgabe einer Bewertung findet in Form einer digitalen Signatur statt. Dies entspricht einer Bewertungsart, die nur positive Aussagen zulässt, wie in Abschnitt 2.4.3 gezeigt. Digitale Signaturen werden durch asymmetrische Kryptographiesysteme erzeugt. Der genaue Vorgang kann im „Handbook of Applied Cryptographie„ in Kapitel 11 nachgelesen werden [Riv97]. Notwendig ist ein digitales Schlüsselpaar bestehend aus einem öffentlichen Testschlüssel und einem geheimen Signierschlüssel. Mit dem Signierschlüssel wird zu einer Nachricht ein Wert berechnet, die sogenannte Signatur. Mithilfe des öffentlichen Testschlüssels kann jeder Nutzer prüfen, ob die Signatur zu der Nachricht passt. Ist dies nicht der Fall wurden entweder Nachricht oder Signatur verändert oder der verwendete Testschlüssel korrespondiert nicht mit dem zur Signaturerzeugung genutzten Signierschlüssel. Die Signatur, die zu einem Testschlüssel passt, kann niemand außer der Besitzer des korrespondierenden Signierschlüssels erzeugen. Sie bietet damit eine Fälschungssicherheit.

Durch den Einsatz eines Signatursystems kann demzufolge das Schutzziel der Integrität wie in Abschnitt 2.3.3 formuliert sehr leicht erfüllt werden.
Auch die Zurechenbarkeit von Bewerter zu Bewertung ist möglicherweise herzustellen. Genauer gesagt, kann eine Bewertung über das Signaturpaket sicher einem bestimmten öffentlichen Testschlüssel zugeordnet werden. Je nachdem, welche Informationen über den Testschlüssel zur Verfügung stehen, kann die Identität des Besitzers des korrespondierenden Signierschlüssels festgestellt werden.

Das Erzeugen einer Signatur über die Openstreetmap-Daten entspricht der Abgabe einer Bewertung. Das Validieren dieser Signatur ist erforderlich für die Auswertung der Bewertung. Ist diese erfolgreich, so können die weiteren der Signatur beiliegenden oder mit dem Testschlüssel verknüpften Informationen ausgewertet werden.

Die verwendeten Begriffe Signatur, Signaturpaket und Bewertung sollen an dieser Stelle klar voneinander abgegrenzt werden. Eine digitale Signatur ist wie oben beschrieben das Ergebnis eines mathematischen Verfahrens also ein Zahlenwert. Ein Signaturpaket enthält Zusatzinformationen, die die Signatur näher beschreiben, sowie die Signatur selbst. Die Informationen können dabei durch die Signatur abgedeckt sein, werden also zusammen mit der Signaturnachricht signiert. Eine Bewertung enthält das Signaturpaket zusammen mit der Signaturnachricht. Zur Auswertung einer Bewertung wird zusätzlich der zugehörige öffentliche Testschlüssel benötigt. Die Verteilung dieses Testschlüssels kann jedoch auf anderen Wegen erfolgen, als zusammen mit der Bewertung. Er wird daher nicht als Teil der Bewertung verstanden. In diesem Entwurf wird davon ausgegangen, dass dieser Testschlüssel zum Zeitpunkt der Bewertungsauswertung dem Nutzer vorliegt.

Der gesamte Ablauf einer Bewertung ist in Abbildung 3.2 dargestellt. Aus einem OSM Objekt werden zunächst semantische Einheiten gebildet, die zusammen mit den Metainformationen

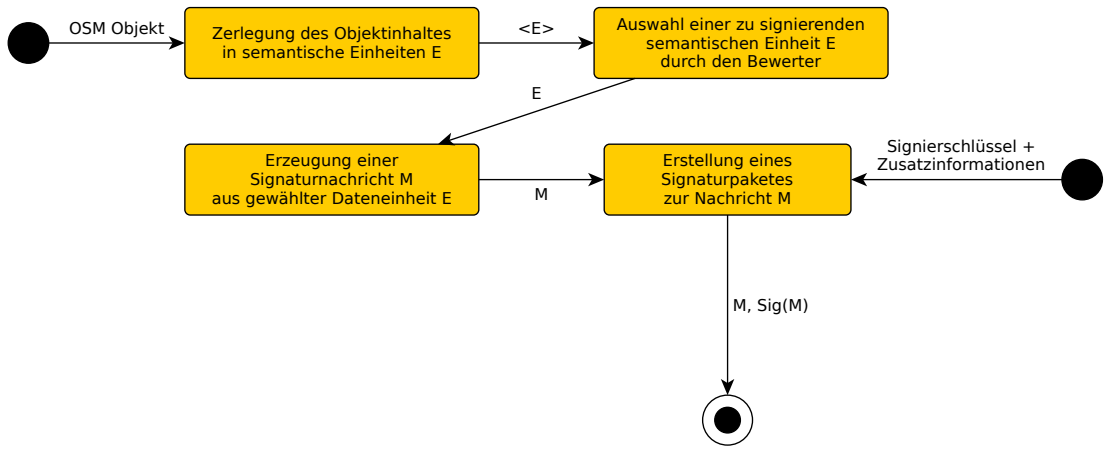

Abbildung 3.2: Prinzipieller Ablauf einer Bewertungsabgabe zu einem Openstreetmap-Objekt

das Objekt repräsentieren. Die semantischen Einheiten sind dadurch gekennzeichnet, dass sie voneinander unabhängig verändert werden können. Von einem Bewerter können sie unabhängig voneinander bewertet werden und die Gültigkeiten der Bewertungen können unabhängig voneinander geprüft werden. Semantische Verbindungen zwischen diesen Einheiten können existieren, werden aber vom Nutzer selber hergestellt. Das bedeutet ein Nutzer kann die Bedeutung von Bewertungen einer semantischen Einheit abhängig von einer anderen machen.

Ein Bewerter wählt aus den semantischen Einheiten eine aus, welche er bewerten möchte. Aus dieser wird eine Signaturnachricht gebildet, das heißt, die Daten der semantischen Einheit werden in Form einer Textnachricht dargestellt. Sie bleiben dadurch menschenlesbar und können wie ein gewöhnlicher Text, der digital signiert wird, behandelt werden. Zusätzlich werden Metainformationen wie beispielsweise die OSM ID hinzugefügt, um sicherzustellen, dass die Bewertung einem bestimmten Openstreetmap-Objekt wieder exakt zugeordnet werden kann.

Für die Textnachricht wird des Weiteren ein Signaturpaket erzeugt. Grundlage ist der Bewertersignierschlüssel der von der Schlüsselverwaltung zur Verfügung gestellt wird. In das Signaturpaket gehen Zusatzinformationen ein, die teilweise durch den Signiervorgang festgelegt werden z.B. ID des zugehörigen Testschlüssels, Signierdatum, Algorithmen, etc. und teilweise durch den Bewerter festgelegt werden können z.B. Toleranzbereich oder verwendete Quellen bei der Bewertung. Die Zusatzinformationen beschreiben die Umstände der Bewertung näher und können auch Informationen zur Gültigkeit enthalten.

Für den Betrieb des Signatursystems müssen folgende Dinge genau festgelegt werden:

- Was sind die semantischen Einheiten und wie wird eine Signaturnachricht gebildet?
- Welche Zusatzinformationen werden dem Signaturpaket hinzugefügt?

Was ein Openstreetmap-Objekt ist und wie es aufgebaut ist, wurde in Abschnitt 2.5.1 genauer erklärt. Abbildung 3.3 stellt die für das Signatursystem interessanten und wesentlichen Teile eines Openstreetmap-Objektes noch einmal dar. Ein Openstreetmap-Objekt kann ein Node ein Way oder eine Relation sein. Jedes Openstreetmap-Objekt besitzt genau eine eindeutige Identifikationsnummer, sowie verschiedene weitere Metadaten wie Bearbeiter, Bearbeitungszeitpunkt, Version etc., die jedoch keine so bedeutsame Rolle spielen, wie die ID. Alle diese Metadaten können nicht von Benutzern direkt editiert werden. Sie werden vom OSM Server nach dem Hochladen der Daten gesetzt.

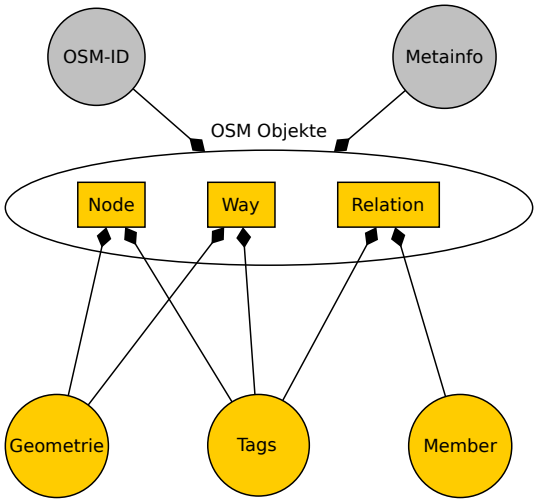

Abbildung 3.3: Die drei OSM Objektarten mit den Eigenschaften, die ihnen zugeordnet werden können.

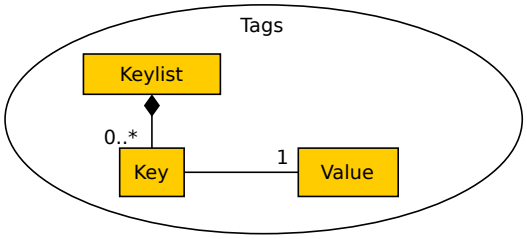

Abbildung 3.4: Tags von OSM Objekten bestehen aus beliebig vielen Key-Value-Paaren.

Je nachdem welches OSM Objekt genau vorliegt sind verschiedene weitere Eigenschaften wie Tags, Geometrie oder Members vorhanden. Diese zeichnen den wesentlichen Inhalt des Objektes aus und sind vom Nutzer editierbar. Aus ihnen werden daher die semantischen Einheiten gebildet.

Welche Informationen bei den drei inhaltstragenden Elementen Tags, Geometrie und Members jeweils als semantische Einheiten verstanden werden und wie der zu signierende Text erzeugt wird, behandeln die nachfolgenden Abschnitte.

3.3.1 Tags

Wie aus Abbildung 3.3 hervorgeht, können Tags an jedes Openstreetmap-Objekt angebracht werden. Sie beschreiben das Objekt näher und klassifizieren es inhaltlich.

Abbildung 3.4 verdeutlicht, dass Tags an einem Openstreetmap-Objekt eine beliebig lange Liste von Key-Value-Paaren sind. Diese Liste kann insbesondere auch leer sein. Ein Tag, also ein Eintrag dieser Liste wird über seinen Key (Schlüssel) adressiert. Die Keys sind dabei eindeutig, das heißt, es kann keine zwei gleichen Keys geben. Jeder vorhandene Key hat genau einen ihm zugeordneten Value (Wert).
Key und Value bilden eine semantische Einheit. Verschiedene Key-Value Paare können getrennt voneinander bearbeitet werden. Key und Value bestehen aus reinem Ascii-Text. Sie können somit direkt in der Signaturnachricht verwendet werden. Um sie voneinander abzugrenzen sind

zusätzliche Texttrennzeichen notwendig, die in den bestehenden Key und Value Texten gesondert gekennzeichnet (escaped) werden müssen. Das genaue Format wird in der Implementierung festgelegt.

Eine Signaturnachricht für einen Tag umfasst somit:

- OSM Identifikationsnummer
- Key
- Value
- Definierte Trennzeichen

Aus der Signaturnachricht muss sich eindeutig rekonstruieren lassen, welche Daten der Bewerter signiert hat. Lässt man nur eines der Elemente weg, so ist dies nicht mehr sichergestellt. Auch das Hinzufügen anderer Elemente verbessert die Situation nicht.

Am einfachsten lässt sich die Notwendigkeit der OSM Identifikationsnummer erläutern. Eine Signatur über ein Key-Value-Paar allein ist so gut wie bedeutungslos, denn dieses könnte jedem beliebigen Openstreetmap-Objekt zugeordnet sein. Sicher ließe sich die Zahl der in Frage kommenden Objekte einschränken, wenn man nur die betrachtet, die gegebenes Key-Value-Paar überhaupt gesetzt haben beziehungsweise im Verlaufe ihrer Historie einmal gesetzt hatten. Nutzt man zusätzlich die Zusatzinformation des Zeitpunktes der Erzeugung der Signatur, so lässt sich auch der Zeitrahmen in der Historie weiter einschränken, der durchsucht werden muss. Dennoch ist dies offensichtlich nicht praktikabel und aufgrund von möglichen Synchronisationsverzögerungen nicht sicher. Die Version auf dem Server könnte beispielsweise schon eine andere sein, als die, die man zu einem bestimmten Zeitpunkt signiert.
Nimmt man die geografische Lage eines Objektes hinzu, so lassen sich mit Key und Value zusammen die in Frage kommenden Objekte sehr stark eingrenzen. Die Position eines Objektes ist dennoch keine eindeutige Referenz auf ein Objekt, denn es ist möglich, dass mehrere Objekte an genau der gleichen Stelle mit genau den gleichen Tags vorkommen. Neben der uneindeutigen Zuordenbarkeit ist eine Positionsangabe in der Regel länger als eine Identifikationsnummer. Ein weiterer ausschlaggebender Grund die OSM ID nicht zu ersetzen ist, dass die Geometrie und alle anderen Objektinhalte veränderbar sind, während die ID für ein einmal erstelltes Objekt bestehen bleibt.

Es ist weiterhin leicht einzusehen, dass das Weglassen des Keys zu Unklarheiten bezüglich der bewerteten Daten auf Objektebene führen würde. Das angeben des Values zusammen mit der ID ist nur dann eindeutig, wenn es keinen weiteren Key an einem OSM Objekt gibt oder gegeben hat, der den selben Value hat. Vor allem Tags, die als Value „yes“ oder „no“ haben, werden häufig zugleich an ein Objekt angebracht. Eine Tankstelle mit „amenity=fuel“ besitzt beispielsweise Tags wie „fuel:diesel=yes“ und „fuel:lpg=yes“ nebeneinander, die angeben, welcher Treibstoff erworben werden kann. Wenn nur der Value „yes“ zusammen mit der ID signiert wurde, sagt das nicht aus, welche Treibstoffart genau vom Bewerter überprüft wurde. Der zugehörige Key ist nicht mehr rekonstruierbar.

Interessanter ist die Frage, ob der Value in der Signaturnachricht weggelassen werden kann. Wie bereits beschrieben sind Keys eindeutig. Zu jedem Objekt kann demnach der Value durch einen gegebenen Key ermittelt werden. Das Problem an dieser Stelle ist jedoch, dass über verschiedene Versionen eines Objektes betrachtet ein Key unterschiedliche Values haben kann. Zwar könnte wieder über den Signaturzeitpunkt versucht werden, die richtige Version des Objektes und damit der ursprüngliche bewertete Value zu finden, allerdings können auch hier die weiter oben bereits angesprochenen Synchronisationsprobleme auftreten und die Sicherheit der Bewertung gefährden. Die richtige Version eines Objektes ließe sich durch Angeben der Versionsnummer in der Signatur auffindbar gestalten. Zugegeben könnten sich in diesem Fall, also der Nachricht aus ID,

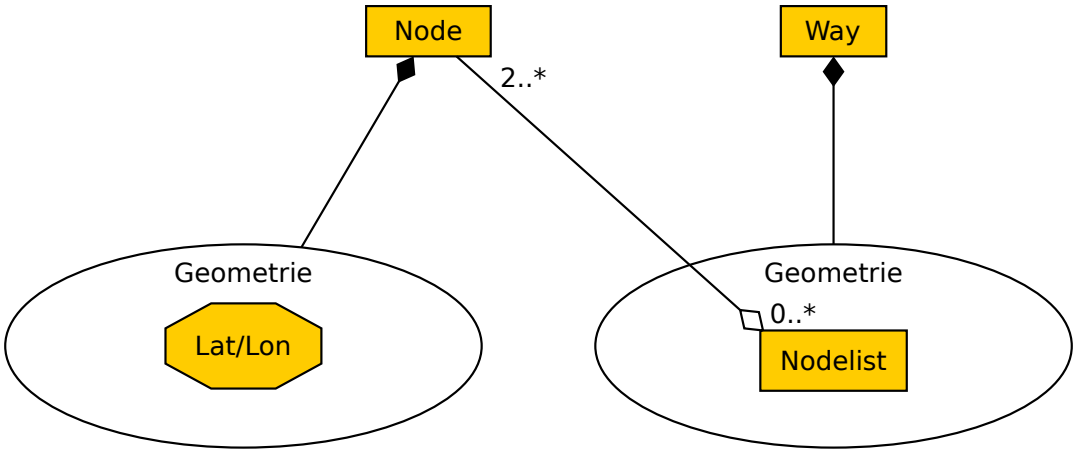

Abbildung 3.5: Aufbau der Geometrie von Knoten (links) bzw. Wegen (rechts).

Key und Version, die Ausgangsdaten exakt rekonstruieren lassen. Allerdings setzt das immer eine Abfrage der Historie eines Objektes voraus. Ohne diese ließe sich beispielsweise auch nicht feststellen, ob zwei Bewerter den selben Tag eines Objektes bewertet haben. Die Versionsnummer des Objektes kann sich ändern, während der Tag an sich gleich bleibt. Eine Versionierung der einzelnen Tags gibt es in Openstreetmap leider nicht.

Definierte Trennzeichen sind technisch notwendig, und bei jeder Variante von Textnachrichten mit mehr als einem Element erforderlich. Zusammenfassend lässt sich sagen, dass die getroffene Auswahl von zu signierenden Elementen die praktikabelste ist.

3.3.2 Geometrie

Unter Geometrie wird in dieser Arbeit die Verortung, also die Position der Openstreetmap-Objekte auf der Erdoberfläche verstanden. In Abbildung 3.3 wird deutlich, dass nur Nodes und Ways eine Geometrie besitzen, Relations jedoch nicht. Relations modellieren Beziehungen zwischen Openstreetmap-Objekten und diese Beziehungen haben keinen geografischen Bezug. Über die Lage der Mitglieder einer Relation könnte ein geografischer Bezug hergestellt werden, sofern die Mitglieder nicht wieder selber Relationen sind. Dies ist aber für die Bewertung von Relationen unnötig.

Die Geomtrie soll daher nur für Punkt- und Linienobjekte in Openstreetmap bezüglich ihres zu bewertenden Inhaltes betrachtet werden. Abbildung 3.5 zeigt die Unterschiede in der Geomtrie von Nodes und Ways. Die Geometrie eines Nodes wird durch geographische Koordinaten auf der Erde in Form der geographischen Breite (Latitude) und der geographischen Länge (Longitude) festgelegt. Jeder Node besitzt dabei genau eine Latitude und eine Longitude, die jeweils in Dezimalgrad angegeben werden. Vom Openstreetmap-server werden derzeit 7 Nachkommastellen bei der Gradzahl berücksichtigt.

Die Geometrie eines Ways in Openstreetmap ist in technischer Hinsicht etwas völlig anderes als die eines Nodes. Ein Way führt lediglich eine geordnete Liste mit Referenzen auf andere Nodeobjekte. Die Geometrie der Nodeobjekte, also die konkreten geographischen Koordinaten dieser bilden somit die Stützstellen des Ways. Die Essenz der Geometrieinformationen eines Ways liegt somit in einer Vielzahl von Punktkoordinaten, sowie die Reihenfolge, in der sie von dem Way durchlaufen werden. Die Liste der Nodes muss dabei mindestens 2 und darf laut

OSM-Spezifikation[1] höchstens 2000 Elemente enthalten. Ein Node kann mehrfach in der Liste auftauchen.

Da sich die Geometrien von Node und Way im Wesen unterscheiden, sollen sie im folgenden Abschnitt mit Blick auf die Bildung einer zu signierenden Nachricht einzeln genauer analysiert werden.

Node

Die Geometrie eines einzelnen Nodes ist durch die Latitude und die Longitude inhaltlich vollständig beschrieben. Eine zu signierende Textnachricht enthält demnach folgende Elemente:

- OSM Identifikationsnummer
- Latitude
- Longitude
- Definierte Trennzeichen

Wieder wird die OSM Identifikationsnummer benötigt, um die signierten Koordinaten später wieder eindeutig einem OSM-Objekt, genauer einem Node zuordnen zu können. Dies ist von entscheidender Bedeutung, denn über die ID können Informationen über den Kontext des Objektes, also seine Tags abgerufen werden. Ohne diesen Kontext reduziert sich die Aussage einer Signatur über eine Nodegeometrie von *„Das Objekt mit der Nummer 1234 befindet sich an dieser Stelle."* zu *„An dieser Stelle gibt es einen OSM Node"*. Die Gültigkeit des Kontextes des referenzierten Objektes wird separat durch Signaturen der Tags abgesichert. Beispiel 3.1 verdeutlicht die Bedeutung des Zusammenhangs von Tags und Geometrie.

Beispiel 3.1 (ASCII) *Ein Student möchte bestätigen, dass sich das Informatiker Cafe ASCII mit der OSM-ID „448002298" und dem Tag „amenity=cafe" an den Koordinaten „Lat:51.0251243°, Lon:13.7232094°" befindet. Er erstellt eine Signatur über die ID und das Tag. Er bestätigt damit, dass es sich bei dem Objekt mit ID 448002298 um ein Cafe handelt. Anschließend signiert er ID und Koordinaten. Er bestätigt damit die Position des Objektes. Zusammen ergibt das die gewünschte Bestätigung, dass sich das Cafe an dieser Stelle befindet.*

Abschnitt 2.4.4 erklärt, warum die Signatur nicht über die Tags und die Geometrie gleichzeitig erstellt wird, sondern separat.

Zu hinterfragen ist an dieser Stelle, ob die eindeutige Zuordnung zu einem Objekt über die ID erfolgen muss oder andere Möglichkeiten zur Verfügung stehen.
Die Position beschrieben durch Latitude und Longitude allein reicht nicht aus, um ein OSM-Objekt eindeutig zu referenzieren. Zwei verschiedene Nodes könnten möglicherweise genau übereinanderliegen. Zudem kann es vorkommen, dass ein Node verschoben wird und somit zum Finden des signierten Nodes die Historie aller Nodes, die an der gegebenen Position gewesen sein könnten, untersucht werden muss. Das betrifft alle Nodes mit einer Versionsnummer von 2 aufwärts. Der Aufwand dafür wäre sehr hoch. Er lässt sich wiederum durch Einbeziehen der Versionsnummer mindern, ist aber immernoch unnötig groß.

Statt Latitude und Longitude wäre es möglich die OSM ID zusammen mit der Versionsnummer zu signieren. Die entsprechende Geometrie lässt sich in diesem Fall wieder eindeutig aus der Historie konstruieren. Die selben Argumente aus Abschnitt 3.3.1, die gegen eine Verwendung der

1. OSM Elemente, http://wiki.openstreetmap.org/wiki/Elements

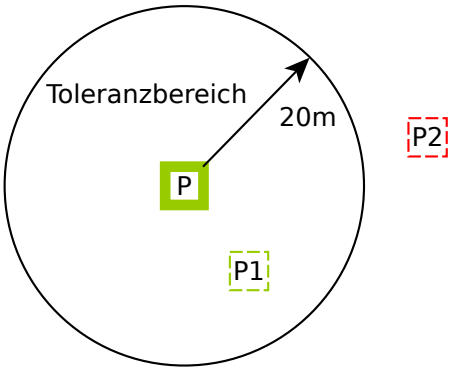

Abbildung 3.6: Toleranzbereich für Gültigkeit der Bewertung von Node P. Bei Verschiebung von P nach P1 gilt die Position des Nodes weiterhin als bestätigt, nicht mehr jedoch bei P2.

Versionsnummer statt des eigentlichen Inhaltes sprechen, gelten auch hier. Anhand der signierten Nachricht ließe sich nicht sofort feststellen, ob zwei Bewerter eigentlich die selbe Information bewertet haben oder nicht. Zudem ist eine Abfrage der OSM Historie immer aufwändiger als die bloße Abfrage eines OSM Objektes mit seinen aktuellen Eigenschaften zu einer gegebenen ID.

Latitude und Longitude einzeln für sich genommen sind offensichtlich auch wertlos. Sie müssen entweder beide zusammen oder keine von beiden in die Signaturnachricht eingehen. Für die definierten Trennzeichen gilt genau das selbe, wie bei den Tagsignaturnachrichten (Abs. 3.3.1).

Toleranzbereich Am Ende von Abschnitt 2.4.4 wurde aufgrund der Unschärfe der Positionsbestimmung vorgeschlagen einen Toleranzbereich bei der Bewertung von Koordinaten einzuführen. Für einen einzelnen Punkt kann dies als Zusatzinformation in der Signatur vermerkt werden.

Der Student aus Beispiel 3.1 kann die Position des ASCII Cafes möglicherweise realistisch ohne weitere Messungen nur auf 20 Meter genau schätzen.

Abbildung 3.6 stellt diesen Sachverhalt grafisch dar. Der Punkt P könnte das ASCII Cafe sein, welches der Student bewertet hat. Bei einer Auswertung der Signatur wird zunächst geprüft, ob die Signatur valide ist. Dazu muss neben der Signatur die zugehörige Nachricht vorliegen, die der Student signiert hat. Anschließend wird geprüft, ob die aktuellen Koordinaten des Punktes innerhalb des Toleranzbereichs liegen. Ist dies der Fall, wie in der Abbildung bei Punkt P1, gilt die Position des Nodes durch die Signatur als bestätigt. Wird der Node so weit verschoben, dass er sich außerhalb des angegebenen Toleranzbereiches befindet z.B. an der Position P2 in der Abbildung, kann seine Position durch die Bewertung nicht mehr als bestätigt angesehen werden. Aus Gründen der Komplexitätsreduzierung wird der Toleranzbereich in dieser Arbeit durch einen Kreis um den signierten Punkt definiert, dessen Radius vom Bewerter angegeben werden kann. Somit ist bei Verschiebung lediglich der Abstand des aktuellen Punktes zum signierten Punkt zu überprüfen. Vorstellbar sind auch Freihandtoleranzbereiche bzw. Bereiche, die sich an anderen Objekten orientieren (z.B. das Cafe ASCII befindet sich irgendwo in dem Gebäude der Fakultät Informatik).

Signieren mehrere Bewerter ein Nodeobjekt an unterschiedlichen Koordinaten entstehen mehrere verschiedene Toleranzbereiche. In Abbildung 3.7 werden diese Toleranzbereiche für ein einziges Nodeobjekt visualisiert. Angenommen der Punkt P befindet sich aktuell zwischen der Position P1 und P3 (in der Abbildung der mit P beschriftete Punkt mit starkem Rahmen) und es sei dieser Punkt in der Vergangenheit an den unterschiedlichen Positionen P1 bis P4 jeweils mit dem angegebenen Toleranzradius bewertet worden, so ergibt sich das vorliegende Schaubild. Angenommen die Validierung der eigentlichen Signatur an den verschiedenen Punkten wäre erfolgreich, so

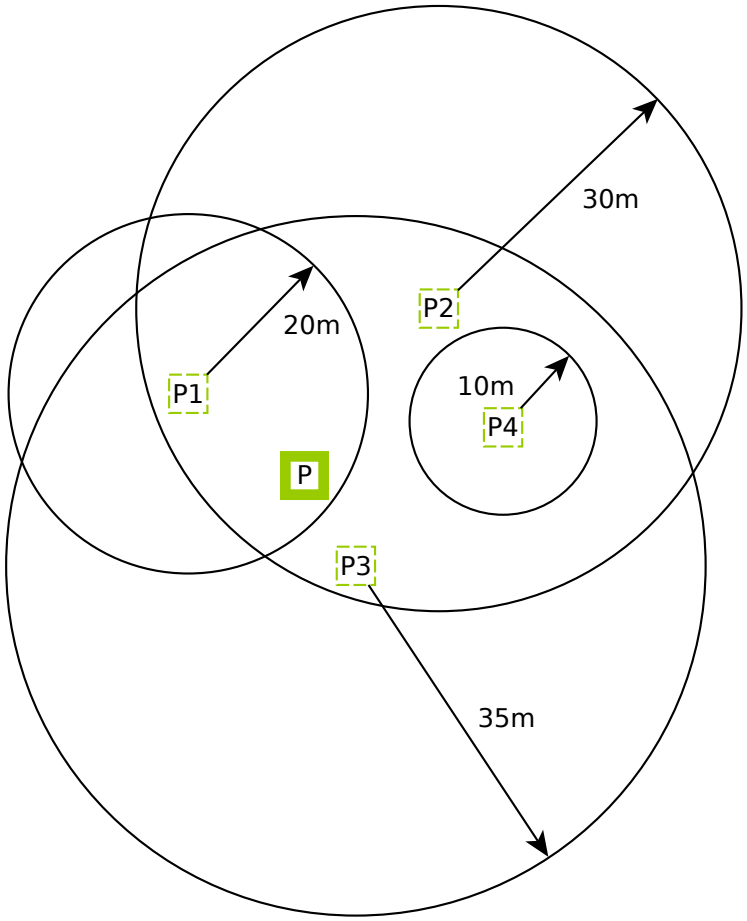

Abbildung 3.7: Überlappung mehrerer Toleranzbereiche eines Nodes.

muss vom Auszuwertenden geprüft werden, ob die Toleranzbereiche eingehalten wurden.
Die aktuelle Position P des Objektes wird durch die Signaturen an P1, P2 und P3 bestätigt. Die Bewertung an der Position P4 spricht dagegen. Je nachdem welche Reputation die jeweiligen Bewerter hatten, könnte für die aktuelle Position P nun ein Reputationswert ermittelt werden. Einfließen könnte auch der angegebene Toleranzradius. Kleinere Radien, also genauere Bewertungsaussagen sind womöglich wertvoller als größere Radien.
Die abgegebenen Bewertungen könnten neben der Einschätzung der Position des aktuellen Punktes durch Nutzer der Daten auch Mappern dazu dienen, den Punkt besser zu positionieren. Als Optimierungskriterium könnte der Reputationswert des Punktes genutzt werden. Dieser wäre beispielsweise besonders hoch, wenn der Punkt in einem Überschneidungsgebiet möglichst vieler Toleranzbereiche liegt. In Abbildung 3.7 wären demnach die Punkte P2 und P3 schlechter bewertet, als P1 oder P. Der Punkt P4 könnte aufgrund seiner höheren Bewertungsgenauigkeit besser sein als P oder P1. Möglicherweise ist aber auch die Bewerterreputation des Bewerters von P4 schlechter als die von P1, was widerum gegen P4 sprechen könnte. Der Reputationswert und dessen Zusammensetzung ist jedoch Sache der Reputationsfunktion und wird an späterer Stelle noch einmal behandelt.
Der konstruierte Fall der abgegebenen Bewertungen bietet jedenfalls keine Möglichkeit eines Konsens der verschiedenen Bewerter an, da sich die Aussagen von P1 und P4 widersprechen, während die Aussagen von P2 und P3 im Überschneidungsgebiet vereinbar sind. Dieser Konflikt kann nicht durch das Bewertungssystem, sondern muss von den Mappen aufgelöst werden. Das Bewertungssystem kann jedoch diesen Konflikt überhaupt erst sichtbar machen. Bei Einigung kann der Bewerter, dessen Messung oder Schätzung ungültig war seine Bewertung zurückziehen.

An dieser Stelle sei darauf hingewiesen, dass es sich durch die Angabe von Zusatzinformationen wie einem Toleranzbereich bei der Bewertung einer Geometrie eines OSM Objektes nicht mehr nur um eine einfache Positivaussage handelt wie: „Das Objekt ist genau an dieser Position." Eine Änderung der Position bedeutet eben nicht automatisch, dass die Bewertung nicht mehr für die neue Position gilt. Anders als in Abschnitt 2.6.2 betrachtet, spielen bewertete ältere Positionsdaten daher bei der Beurteilung der aktuellen Positionsdaten auch eine wesentliche Rolle und sollten nicht ignoriert werden. Die Unschärfe der Geometrieinformationen ist ein wesentlicher Unterschied zu den Tags bei denen die Gültigkeit klar entscheidbar ist. Auch die Bewertungen unterscheiden sich demzufolge.

Way

Um die Geometrie eines Ways zu signieren, ist es erforderlich, wie bei den Tags und Nodes auch, den wesentlichen Inhalt zu erfassen und die Elemente einer Signaturnachricht zu definieren. Im Unterschied zu dem sehr einfach gestalteten Nodeobjekt gibt es bei einem Way sehr viel mehr Möglichkeiten eine Bewertungsaussage zu treffen. Das liegt unter anderem daran, dass mit einer Waygeometrie deutlich mehr unterschiedliche Editieroperationen durchgeführt werden können.
Zunächst besteht die Wahl, ob die ganze Waygeometrie als solche signiert oder sie in Teilen signiert wird, wobei durch die signierten und somit bewerteten Teile eine Reputationsaussage über die gesamte Geometrie getroffen werden könnte.

Bei der Bildung einer Signaturnachricht ist es natürlich von großer Wichtigkeit welche geographische Position die einzelnen Nodes besitzen, die durch die Liste der Nodes in der Waygeometrie referenziert werden. An dieser Stelle reicht es nicht nur die Referenzen auf die einzelnen Nodes zu signieren, da sich die konkreten Koordinaten somit unbemerkt ändern könnten, während die Signatur über die Node ID intakt bleibt. Dieses Verhalten ist nicht gewünscht. Als Elemente für die Signaturnachricht kommen also die ganz konkreten Werte von Latitude und Longitude der referenzierten Koordinaten in Frage. Die Versionsnummer der Nodes vermag diese nicht zu ersetzen, da eine Versionsänderung auch durch Setzen von Tags an den Nodes verursacht werden kann und nicht eine Positionsänderung bedeuten muss.

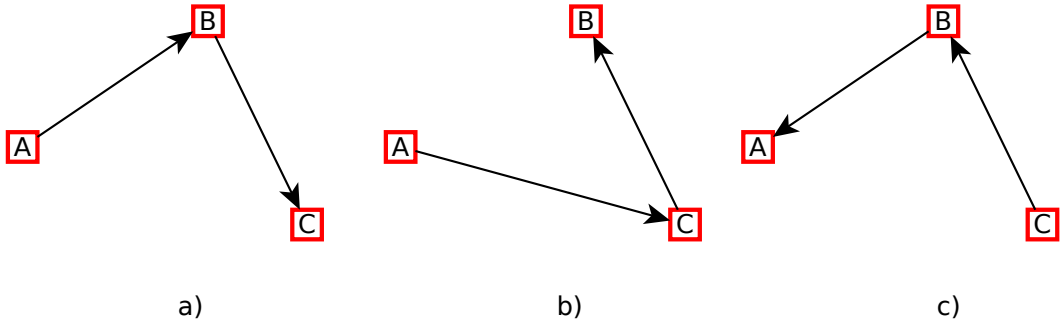

Abbildung 3.8: Die selben drei Knotenpunkte A, B und C können verschiedene Waygeoemtrien bilden, wenn sie in unterschiedlicher Weise durchlaufen werden.

Zunächst wird der Fall betrachtet, dass die gesamte Waygeometrie als Ganzes signiert wird. Die Signaturnachricht muss demzufolge die Koordinaten sämtlicher Nodes des Ways enthalten, wobei deren Reihenfolge in der Signaturnachricht bedeutsam ist. Abbildung 3.8 zeigt beispielhaft drei verschiedene Ways, die jedoch aus den gleichen Knoten A, B und C bestehen. Lediglich deren Reihenfolge in der Nodeliste des Ways ist verschieden. Die Pfeile weisen darauf hin, dass Wege im Openstreetmap-Datenmodell gerichtet sind. Während die Geometrien in Fall a) und b) unterschiedliche Verbindungen zwischen den Nodes herstellen, sind die Verbindungen im Vergleich von Fall a) und c) zwischen den Nodes gleich. Lediglich deren Richtung hat sich invertiert. In der Praxis spielt die Richtung des Ways tatsächlich in vielen Fällen keine Rolle. Bedeutsam ist sie jedoch, um zum Beispiel Einbahnstraßen, Fließrichtungen von Gewässern oder andere richtungsabhängige Gegebenheiten darzustellen, die unter allen Linientypen im Moment allerdings nicht sehr häufig vorkommen. Der Fall a) und c) muss im Allgemeinen allerdings unterschieden werden.

Eine Signaturnachricht für eine Waygeomtrie als Ganzes könnte also so aussehen:

- OSM Identifikationsnummer des Ways
- Latitude und Longitude aller referenzierten Knoten in der Reihenfolge der geordneten Liste des Ways
- Definierte Trennzeichen

Wie üblich wird die OSM Identifikationsnummer des Ways zum referenzieren des zugehörigen OSM Objektes genutzt, an dem weitere Informationen durch Tags angebracht sein können. Die Situation ist im Falle der Identifikationsnummer ähnlich wie bei den Nodes und wird daher nicht noch einmal gesondert begründet. Die Liste der Koordinaten, sowie die Trennzeichen wurden ebenfalls bereits besprochen und werden daher nicht weiter behandelt.

Viel wichtiger ist hier die Diskussion, wie sich nun die verschiedenen Bearbeitungsmöglichkeiten des Ways auf die Signatur auswirken, welches Verhalten gewünscht ist und ob eine Signatur von Teilgeometrien möglich und sinnvoll ist. Das Hinzufügen eines Nodes zu einem bestehenden Way ist eine sehr häufig vorkommende Änderung von Waygeometrien. Das Beispiel der neuen Straße, die an eine bereits bestehende Straße angeschlossen wird, wurde in dieser Arbeit bereits öfter benannt. Weiterhin könnten Knoten hinzugefügt werden, um Objekte direkt auf dem Weg zu modellieren. Bei einer Straße sind das zum Beispiel Poller, Tore, Ampeln, Zebrastreifen etc. Das Problem aus Sicht des Bewertungssystems entsteht dann, wenn diese Dinge bei einem bestehenden Way nachgetragen werden. Abbildung 3.9 zeigt, wie sich die Geometrie eines Ways von A über B nach C ändert, wenn zwischen B und C ein neuer Node D hinzugefügt wird. Die

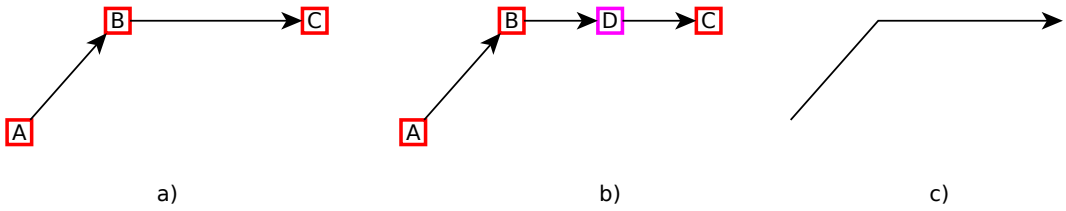

Abbildung 3.9: Einem bestehenden Way a) durch die Knoten A, B und C wird zwischen B und C ein neuer Node D hinzugefügt. Es entsteht Way b). Der prinzipielle Verlauf des Linienzuges c) bleibt bestehen.

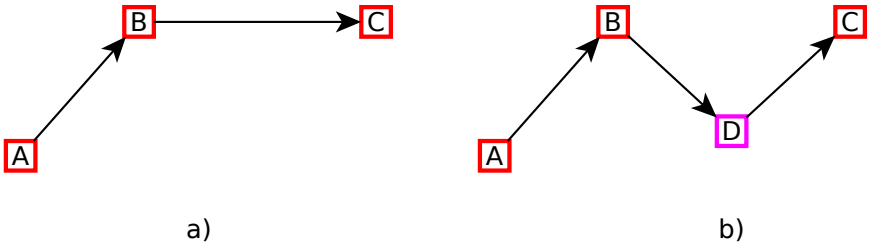

Abbildung 3.10: Eingefügter Node D ändert Verlauf des Linienzuges.

Verbindung zwischen B und C wird aufgetrennt und stattdessen entsteht eine neue von B nach D und von D nach C. Obwohl sich die Geometrie des Ways damit technisch geändert hat, hat sich die inhaltliche Aussage der Geometrie nicht geändert. Der prinzipielle Verlauf der Linien der Ways a) und b) ist genau gleich und entspricht c). Bei der Auswertung der Signatur über die gesamte Geometrie sollte dieser Umstand berücksichtigt werden. Die Bewertung der Geometrie sollte also trotz eingefügten Nodes weiterhin gültig bleiben.

Es kann jedoch auch vorkommen, dass der eingefügte Node den Linienzug verändert, wie beispielhaft in Abbildung 3.10 demonstriert. Auch bestehende Knoten können verschoben werden, was das selbe Ergebnis, nämlich einen inhaltlich veränderten Linienzug zur Folge hätte. An dem Beispiel soll jedoch folgender Aspekt näher betrachtet werden: Der eingefügte Node D ändert zwar die Geometrie als Ganzes, allerdings bleiben Teile intakt. Die Verbindung von B nach C nimmt zwar in Way b) einen anderen Verlauf als in Way a), allerdings bleibt die Verbindung von Node A nach Node B davon völlig unberührt. Es stellt sich die Frage, ob die Geometrie in logische Einheiten unterteilt werden kann, die wiederum einzeln für sich signiert und ausgewertet werden können. Gibt es die Möglichkeit kleinere Einheiten der Geometrie zu bewerten, so muss der Bewerter nicht immer die komplette Geometrie prüfen. In Abbildung 3.10 wäre es also denkbar, dass ein Bewerter nur das Teilstück zwischen A und B des Ways vermisst und über den Rest des Weges von B nach C keine Aussage treffen kann und möchte.
Kleinere Einheiten des Ways auszuwerten ermöglicht das Übernehmen von Bewertungen unbearbeiteter Teile des Ways. Das Verfeinern der Granularität verursacht allerdings größere durch die Bewertung anfallende Datenmengen. Dem steht entgegen, dass eventuell mehr Bewertungen von Datenänderungen verschont bleiben und deren Nutzen sich dadurch erhöht.

Um den Way in inhaltlich kleinere Untereinheiten aufzuteilen, gilt es zunächst einmal festzustellen, welche Einheiten das sein können.

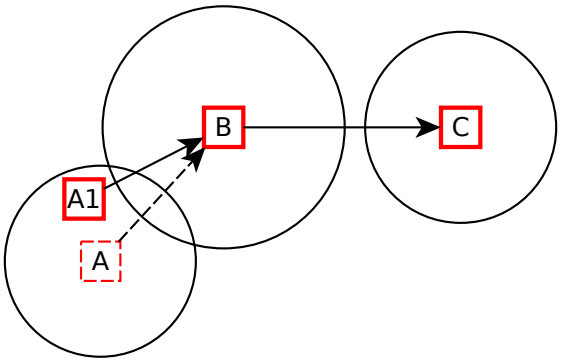

Abbildung 3.11: Toleranzbereich einzeln signierter Nodes eines Ways. Obwohl Punkt A nach A1 verschoben wurde gilt Waygeometrie als bestätigt.

Einzelnodesignaturen: Eine Möglichkeit wäre jeden Node des Ways als Einheit zu betrachten und demnach einzeln zu signieren. Dies hätte den Vorteil der Einfachheit. Die Signatur eines Ways kann dadurch auf die Signatur von mehreren einzelnen zu dem Way gehörenden Nodes zurückgeführt werden. Ein dem Way neu hinzugefügter Node wäre einfach nicht signiert, würde aber die Signaturen der bereits vorhandenen Nodes nicht beeinflussen. Ähnlich wie bei den Signaturen einzelner Nodeobjekte in Abschnitt 3.3.2, könnte ein Toleranzbereich für jeden Node der Geometrie festgelegt werden. Die vorhandenen Nodes hätten jeder für sich einen eigenen Toleranzbereich, der in Abbildung 3.11 durch den Kreis um die Nodes dargestellt ist. Node A könnte in dieser Abbildung an die Stelle A1 verschoben werden und seine Bewertung wäre weiterhin gültig, wodurch die ganze Geometrie weiterhin als durch die Bewertung bestätigt angesehen werden kann.
Eine Signaturnachricht für einzelne Nodes eines Ways besteht aus folgenden Elementen:

- OSM Identifikationsnummer des Ways
- OSM Identifikationsnummer des Nodes
- Latitude des Nodes
- Longitude des Nodes
- Definierte Trennzeichen

Die ID des Weges wird wie bisher benötigt, um das richtige Way-Objekt zu referenzieren für dessen Geometrie eine Bewertung vorgenommen wird. Mit der Node ID und der Latitude und Longitude wird der Node aus der Liste der zum Way gehörigen Nodes beschrieben, für den die Einzelnodesignatur gilt. Die ID dient dazu die Signatur dem richtigen Node zuzuordnen.
Besonders problematisch an diesem Ansatz der Einzelnodesignaturen ist jedoch, dass die Reihenfolge der Nodes nicht beachtet wird. Dies führt dazu, dass die in Abbildung 3.8 dargestellten Wayvarianten durch die gleichen Nodes alle als durch die Signatur abgedeckt betrachtet werden, sofern die Nodes signiert sind.
Wie schwerwiegend dieses Problem ist, hängt unter anderem davon ab, wie oft Wege derart verändert werden, dass die Nodes in einer anderen Reihenfolge durchlaufen werden. Bei Openstreetmap ist dies eher selten der Fall, da es kaum sinnvolle Anwendungen dafür gibt. Das Ignorieren der Nodereihenfolge bei der Bewertung öffnet jedoch Sicherheitslücken, die ein Vandale ausnutzen könnte, um das Bewertungssystem auszuhebeln. Bei der absichtlichen Verfälschung des Ways unter Beibehaltung der Gültigkeit der Bewertungen wäre ein Angreifer immerhin nicht völlig frei, da er zumindest die signierten Nodes in der Wayliste lassen muss - in welcher Reihenfolge auch immer. Ein möglicher Schutz gegen solche Angriffe würde einiges an zusätzlichem

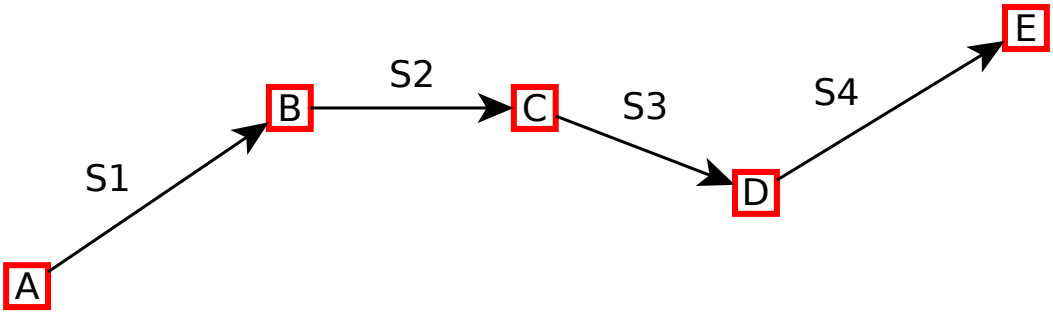

Abbildung 3.12: Der Way mit 5 Nodes von A bis E kann als Menge von 4 Segmenten S1 bis S4 aufgefasst werden.

Aufwand erzeugen, da neben den einzelnen Nodes deren geordnete Liste signiert werden muss, um auch die Reihenfolge abzusichern. Eventuell lassen sich hier auch Teilreihenfolgen signieren, um nicht wieder Aussagen über den gesamten Way zu erzwingen. Es müssen dann allerdings die Fragen beantwortet werden, ob eine Nodesignatur in einer Waygeometrie ohne eine Signatur der Liste überhaupt gültig ist, wie es mit der Validität der Reihenfolge zugeht, wenn Nodes gelöscht oder hinzugefügt werden und wie die Bewertungsinformationen möglichst kompakt gespeichert werden können. Ist also eine Signatur der Liste bei jeder Nodesignatur erneut erforderlich und wird mit dieser zusammen abgespeichert oder kann auf bereits bestehende Listensignaturen verwiesen werden?

Segmentsignaturen: Neben dem Ansatz der Einzelnodesignaturen, der im Detail vor allem bei der Reihenfolge der Nodes problematisch ist, gibt es weitere Möglichkeiten eine Waygeometrie logisch aufzuteilen und zu signieren. Fasst man eine Waygeometrie nicht als Liste geordneter Nodes auf, sondern als Menge von Segmenten, so kann man diese Segmente als Untereinheiten des gesamten Ways verstehen. Abbildung 3.12 stellt die Segmente S1 bis S4 eines Ways anschaulich dar. Ein Segment ist dabei die Verbindung von zwei benachbarten Nodes des Ways. Das Segment ist gerichtet und wird durch die Lage seines Anfangs- und Endnodes bestimmt. In diesem Modell besteht also der Way nicht aus Nodes, sondern aus Verbindungen zwischen Nodes. Die Reihenfolge dieser Verbindungen, also der Segmente, spielt dabei keine Rolle. Die Menge aller Segmente beschreibt die Geometrie des Ways vollständig.
Die inhaltliche Beschreibung eines Segmentes durch seine zwei Knoten führt zu folgender Signaturnachricht:

- OSM Identifikationsnummer des Ways
- Latitude des Startnodes
- Longitude des Startnodes
- Latitude des Endnodes
- Longitude des Endnodes
- Definierte Trennzeichen

Die OSM ID wird benötigt, um das signierte Element dem entsprechenden Way-Objekt zuzuordnen. Latitude und Longitude von Start- und Endnode beschreiben wie bereits erwähnt die Lage des Segmentes. Trennzeichen grenzen die einzelnen Werte wie üblich voneinander ab.

Das Auftrennen von Ways stellt für das Bewertungssystem ein schwieriges Problem dar. Bei dieser Editieroperation wird ein Wegobjekt an einem Node aufgespalten, so dass hinterher zwei eigenständige Wayobjekte vorhanden sind. Abbildung 3.13 veranschaulicht diesen Vorgang. Bei der Auftrennung des Ways von Node A bis E mit ID=1 an Node C findet intern folgender Vorgang statt:

- Anlegen eines neuen Way-Objektes
- Kopieren der Nodeliste ab dem Trennnode aus dem ursprünglichen Way-Objekt in das neu angelegte
- Entfernen der kopierten Nodes aus dem ursprünglichen Way-Objekt
- Kopieren aller Tags und Relationsmitgliedschaften des alten Wayobjektes in das neue.

Das Resultat ist ein Wayobjekt, welches die ID des ursprünglichen Objektes behält, in Abbildung 3.13 der linke Teil mit der ID=1, und ein neues Wayobjekt mit einer neu vergebenen ID, im Beispiel der rechte Teil mit ID=2. Die Wayobjekte sind aber weiterhin über den gemeinsamen Node C verbunden, was in der Grafik jedoch nicht so dargestellt ist. Inhaltlich ändert sich durch die Auftrennung des Ways jedoch gar nichts. Lediglich die OSM Historie des neuen Teilstückes wird neu begonnen. Durch wieviele Wayobjekte eine reale Straße bzw. linienartige geographische Begebenheit in Openstreetmap abgebildet wird, spielt für die Aussage der Abbildung keine Rolle. Lediglich das Editieren kann durch längere Wegstücke etwas einfacher sein, als durch viele kleinere. Eine Auftrennung von Ways ist jedoch immer dann nötig, wenn Eigenschaften nur für einen bestimmten Teil eines Ways gelten. Beispielsweise gelten Geschwindigkeitsbeschränkungen oft nur für einen kleinen Teil einer Straße. Auch eine Brücke kann nur an den Teil der Straße getaggt werden, der tatsächlich als Brücke gebaut ist. Es gibt viele Fälle in denen bestehende Ways aufgetrennt werden müssen.
Dies hat negative Konsquenzen für die bereits abgegebenen Bewertungen. Da die Signatur der Geometrie immer über die Way ID mit dem entsprechenden Objekt verbunden ist, gelten die Bewertungen plötzlich nur noch für den Teil der Geometrie, dessen ID gleichgeblieben ist. Aus Sicht dieses Objektes wird ein Teil der Geometrie einfach gelöscht, was die dafür abgegebenen Bewertungen invalidieren muss. Es wäre inhaltlich wünschenswert, wenn die abgegebenen Bewertungen für den Teil der Geometrie, der in das neue Objekt kopiert wird, einfach mit übernommen werden könnten. Die Bindung an die Way ID erlaubt dies jedoch nicht.
Das selbe Problem findet leider auch bei den Tags statt. Die in das neue Way-Objekt kopierten Tags sind in dem neuen Objekt nicht bewertet, obwohl sie es in dem alten waren. Wünschenswert wäre eine Übertragung der Bewertung, aber nur in diesem Fall.
Das Problem kann leider in dieser Diplomarbeit nicht gelöst werden. Würde man die Bewertungen übertragbar gestalten, z.B. durch Nichteinbeziehung der ID, dann kann dieser Umstand missbraucht werden und Bewertungen können mit Objekten assoziiert werden, bei denen dies nicht gerechtfertigt ist.

3.3.3 Members

Members spielen nur im Zusammenhang mit dem OSM Objekttyp Relation eine Rolle. Relations führen eine Liste mit Referenzen auf andere OSM Objekte. Jedes Element dieser Liste ist ein Member der Relation[2]. In der Struktur unterscheidet sich eine solche Liste von der Nodeliste bei Ways darin, dass alle drei OSM Objekttypen referenziert werden können und dass dieser Referenz eine Rolle zugewiesen werden kann. Die Reihenfolge der Member in der Liste wird zwar persistent gespeichert, ist aber nicht bei allen Anwendungen von Bedeutung. Die Sortierung der Objekte ist beispielsweise bei Buslinienrelationen wichtig. Wege und Haltestellen werden in der Reihenfolge referenziert, wie sie vom Bus befahren werden. Allerdings ist bei der Reihenfolge,

2. Relationen in OSM http://wiki.openstreetmap.org/wiki/DE:Relationen

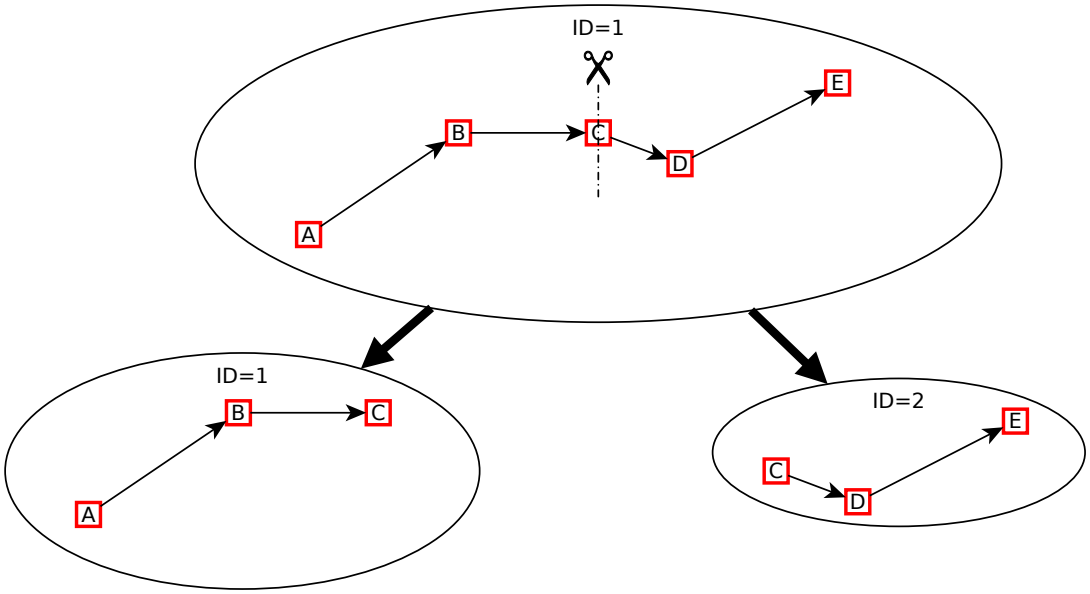

Abbildung 3.13: Das splitten des Ways durch die Punkte A bis E erzeugt zwei separate Wayobjekte, wobei das von A bis C die ID behält, während das von C bis E eine neue bekommt.

selbst in den Fällen, in denen sie von Bedeutung ist, nicht genau definiert, welche Sortierung inhaltlich richtig ist und welche nicht. Bei dem Beispiel Buslinie ist also nicht klar, ob zuerst der Way in der Relationsliste, auf dem der Bus eine Bushaltestelle anfährt oder zuerst der Node, der die Bushaltestelle darstellt.

In diesem Entwurf wird daher die Reihenfolge von Members in der Relation nicht berücksichtigt. Ähnlich wie Tags, werden die Members zusammen mit ihrer Rolle in der Relation einzeln signiert. Eine Signaturnachricht für ein Member sieht demnach so aus:

- OSM Identifikationsnummer der Relation
- OSM Identifikationsnummer und Objekttyp des Members
- Rolle des Members in der Relation
- Definierte Trennzeichen

3.3.4 Probleme des OSM-Datenmodells

Bei der Erstellung der Signaturnachricht treten viele Schwierigkeiten auf, die sich teilweise auf Schwächen des OSM-Datenmodells zurückführen lassen. Dieser Zusammenhang soll kurz erläutert werden, wird in dieser Diplomarbeit aber hingenommen, da das Bewertungssystem als Erweiterung des bestehenden OSM Systems konzipiert wurde und keine Eingriffe beispielsweise in die OSM Datenstruktur erfordern soll.

Als Beispiel zur Erläuterung diene erneut das Informatiker Cafe ASCII, dessen Tagging in Tabelle 3.2 einschließlich der Koordinaten aufgeführt ist. Das Tagging wurde für das Beispiel jedoch verändert. In der OSM-Datenbank hat das Nodeobjekt, welches das Cafe repräsentiert keine eigenen Adressinformationen. Diese befinden sich an dem Objekt, welches das Gebäude der Informatik darstellt und sind nicht formal mit dem Nodeobjekt des Cafes assoziiert.

Key	Value
addr:city	Dresden
addr:country	DE
addr:housenumber	46
addr:postcode	01187
addr:street	Nöthnitzer Straße
amenity	cafe
name	ASCII
drink:club-mate	yes
Latitude	Longitude
51.0251243°	13.7232094°

Tabelle 3.2: OSM Beispiel Cafe ASCII

Die erste Schwäche des OSM Datenmodells ist eine **fehlende formale Assoziation, bei vorhandener semantischer Assoziation**. Dieses Problem tritt gleich auf mehreren Ebenen auf. Die erste ist die **Interobjektebene**, die bereits angedeutet wurde. Der Node des Informatiker Cafes ASCII ist semantisch assoziiert mit dem Fakultätsgebäude in dem es sich befindet. Es besitzt wie bereits erwähnt die gleiche Adresse und seine Existenz ist an die Existenz des Informatikgebäudes geknüpft. Würde das Informatikgebäude abgerissen, würde auch das Cafe verschwinden. Dieser Zusammenhang ist im aktuellen Openstreetmap-Datenmodell nicht wirklich erfassbar. Mit dem Objekttyp der Relation könnte zwar der Node des Cafes mit dem Way des Fakultätsgebäudes verknüpft werden, aber diese Relation müsste explizit von Programmen, die die OSM Daten auswerten, berücksichtigt werden. Ein Programm, welches bei dem aktuellen Datenmodell die Adresse des Informatiker Cafes ASCII aus den OSM Daten gewinnen möchte, muss teils rechenintensive, teils ungenaue geometrische Suchverfahren anwenden wie zum Beispiel das Finden der nächstgelegenen Adresse etc.
Dem Beispiel in Tabelle 3.2 wurden die Adressinformationen an den Node des ASCII Cafes explizit als Tags hinzugefügt, um neben der Interobjektebene die **Innerobjektebene** zu verdeutlichen, bei der das Fehlen formaler Assoziationen Probleme verursacht. Semantisch zusammengehörig sind die Tags mit den Adressinformationen, deren keys mit addr: beginnen. Wie man sieht wurde bei der Erarbeitung des Adressschemas bereits versucht dem Problem der fehlenden formalen Assoziationen von Tags zu begegnen, indem man Namensräume einführt. Eine weitere semantische Assoziation bildet der Tag „amenity=cafe“ und der Namenstag „name=ASCII“. Würde das Cafe ausziehen, würde der Node sicherlich nicht mehr ASCII heißen. Auch der Tag „drink:clubmate=yes“, der anzeigt, ob das Getränk Club Mate dort verkauft wird, ist nur im Zusammenhang mit dem Tag „amenity=cafe“ gültig. Es liegt jedoch lediglich an der inhaltlichen Bedeutung der Tags, ob diese eine starke Assoziation besitzen oder nicht. Die Assoziation bzw. Gruppierung von zusammengehörigen Eigenschaften richtet sich im Wesentlichen danach, wie stark eine Eigenschaft die Bedeutung einer anderen beeinflusst oder von ihr abhängt. Bisher wurde die geografische Position dabei außer Acht gelassen. Doch vor allem diese spielt eine wesentliche Rolle für die Bedeutung der Tags. Die Information über das ASCII Cafe verliert an einer anderen Position, je nachdem wie weit entfernt, mehr oder weniger stark an Bedeutung. Auch die Gültigkeit der Adressinformationen steht dann in Frage.
Unabhängig wäre jedoch beispielsweise die Existenz des ASCII Cafes an der festgelegten Position von der Adressinformation an dieser Stelle. Selbst wenn die Hausnummern geändert würden oder sich Postleitzahlenbereiche ändern, wäre das ASCII Cafe immernoch an der gleichen Stelle gültig.

Das Beispiel zeigt, dass die Abhängigkeiten und semantischen Zusammengehörigkeiten der einzelnen Eigenschaften von Objekten untereinander bzw. verschiedener Objekte miteinander im OSM Datenmodell nicht erfassbar sind. Für das Bewertungssystem ist es folglich schwierig die Gültigkeit von Bewertungen für bestimmte Eigenschaften festzustellen.

Das Problem der fehlenden formalen Assoziationen zeigt sich auch in den Schwierigkeiten bei der Auftrennung von Ways. Wird ein Way getrennt, so entstehen zwei Objekte, die formal nicht

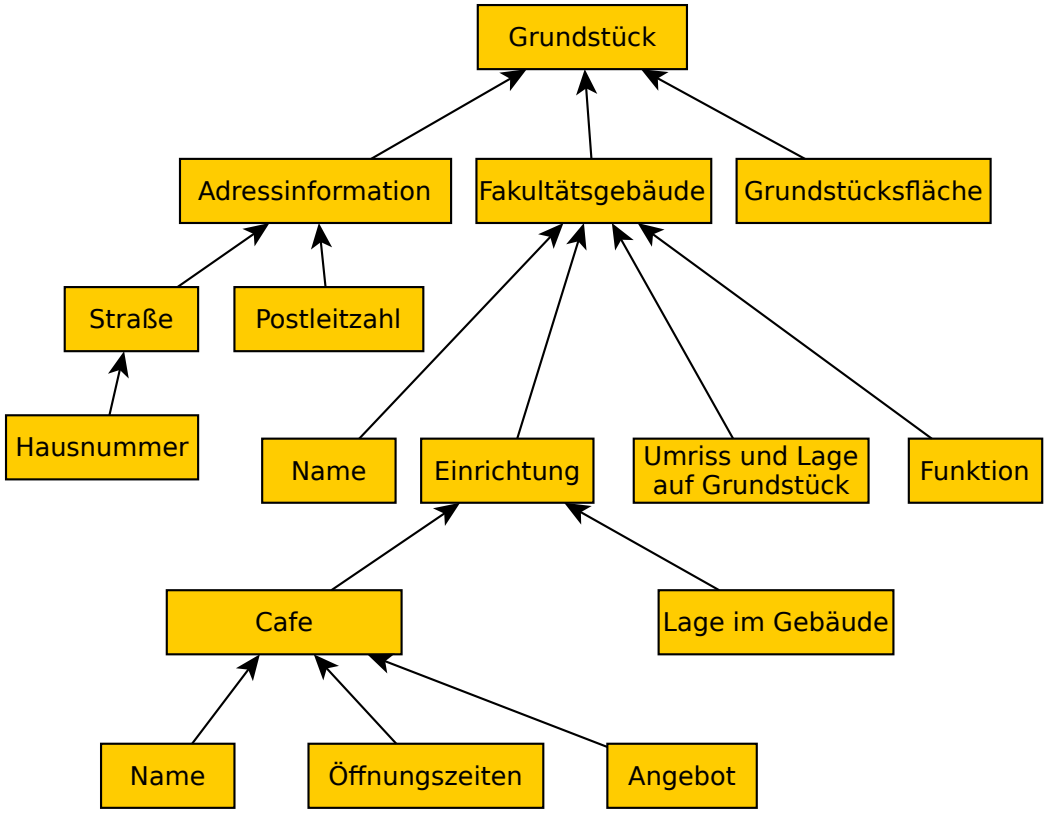

Abbildung 3.14: Hierarchische Datenstruktur des Cafes im Gebäude der Fakultät Informatik.

assoziiert sind, obwohl sie es semantisch sind. Sie besitzen nämlich beide die gleichen Tags und sind Teile eines realen Weges. Müsste ein Straßenname geändert werden von einem relen Weg der beispielsweise aufgrund unterschiedlicher Geschwindigkeitsbegrenzungen in mehrere Wayobjekte aufgeteilt wurde, so muss der Name bei jedem dieser Teilobjekte einzeln geändert werden. Auch hier gab es bereits Überlegungen von aktiven Mappern die Problematik mithilfe von Relationen anzugehen, die jedes Teilstück des Ways als Member führen. Gemeinsame Tags, wie der Straßenname würden in der Relation gespeichert.

Eine Möglichkeit das Openstreetmap-Datenmodell zu verbessern, könnte die Einführung von Hierarchien sein. Abbildung 3.14 zeigt, wie das Beispiel des Cafes innerhalb des Informatikgebäudes in einer Hierarchie strukturiert sein könnte. Die Pfeile weisen auf den Eintrag der nächsthöheren Hierarchie, von welchem ein Eintrag direkt abhängig ist. Einträge auf der gleichen Hierarchieebene, also solche, die auf den selben Eintrag verweisen, sollen möglichst unabhängig voneinander sein. So ist das Fakultätsgebäude an sich mit seinen Eigenschaften und seinen Einrichtungen von einer Adressänderung des Grundstückes auf dem es sich befindet unabhängig. Das Cafe ASCII ist eine Einrichtung mit einer Lage innerhalb des Fakultätsgebäudes. Die Einrichtung, also ein Raum des Fakultätsgebäudes existiert nur, wenn das Fakultätsgebäude existiert. Das Cafe existiert nur, wenn der Raum existiert und so weiter. Wird ein Eintrag entfernt, so wird der gesamte Unterbaum mit ihm entfernt, da er direkt abhängig ist. Wird also das Cafe geschlossen und stattdessen ein Lagerraum eingerichtet, verlieren die Öffnungszeiten,der Name und das Angebot des Cafes ihre Bedeutung.
Da Openstreetmap-Daten iterativ ergänzt werden, wäre es für die Funktionsweise der hierarchischen Datenstruktur erforderlich, dass immer wieder neue Hierarchien ergänzt und vorhandene Abhängigkeiten geändert werden können. Es ist somit möglich, dass ein Mapper das Cafe einträgt ohne, dass das Informatikgebäude in den Daten vorhanden ist. Er würde das Cafe direkt in Abhängigkeit des Grundstückes eintragen. Wird das Informatikgebäude später ergänzt, kann

Objektteil	OSM-Objekt	Signaturnachricht enthält	Signaturzusatzinfo
Tag	Node, Way, Relation	OID, Key, Value	Source
Geometry	Node	OID, Lat, Lon	Tolerance, Source
Geometry	Way	OID, Segment	Tolerance, Source
Member	Relation	OID, OID(Mem), Rolle(Mem)	Source

Tabelle 3.3: Entwurf der Signaturnachrichten der OSM Objekte

das Gebäude und die Einrichtung als Hierarchie hinzugefügt werden und die Referenz des Cafes geändert werden.
Möglicherweise wäre es für so ein Datenmodell sinnvoll Positionen relativ zu einer höheren Hierarchie anzugeben, für die wiederum Lagedaten existieren. Es stellt sich weiterhin die Frage, welches die oberste Hierarchie ist (z.B. Welt) und ob es nur einen Baum geben sollte oder mehrere oder womöglich ein gerichtetes Netz.

Eine Bewertung solcher hierarchischer Daten könnte sich auf einzelne Pfade oder Abschnitte eines Baumes beziehen. Ein Bewerter könnte demnach Aussagen treffen wie: „Im Fakultätsgebäude gibt es eine Einrichtung, die ein Cafe ist und den Namen ASCII trägt." Ändert sich der Name des Cafes wäre die Aussage, dass es eine Einrichtung des Fakultätsgebäudes ist immer noch zutreffend.

Es lässt sich zusammenfassend sagen, dass Bewertungsansätze, die auf einem verbesserten Openstreetmap-Datenmodell aufbauen vielversprechende Aussichten bieten Probleme wie die Feststellung der Gültigkeit von Bewertungen zu lösen. Im Rahmen des aktuellen Datenmodells lassen sich Assoziationen und Abhängigkeiten von Daten vor allem auf der semantischen Ebene feststellen. Semantische Einheiten wie Tags und Lagedaten sind einzig über das Objekt zu welchem sie gehören formal gruppiert. Da sich keine Haupttags finden lassen und auch die Lagedaten nicht immer als definierende Haupteigenschaften eines Objektes gelten können, werden alle semantischen Einheiten als gleichbedeutend erachtet. Ob die Änderung einer semantischen Einheit Auswirkungen auf die Gültigkeit von anderen semantischen Einheiten hat, kann immernoch bei der Auswertung und Feststellung der Gültigkeit von Bewertungen entschieden werden, wird aber zunächst nicht angenommen.

Für diesen Entwurf fasst Tabelle 3.3 zusammen, welche Daten des Openstreetmap-Objektes in die jeweiligen Bewertungen für Tags, Geometry und Members einfließen. Um die verwendeten Daten voneinander abzugrenzen, werden jeder Signaturnachricht Texttrennzeichen, wie Zeilenumbrüche, Gleichheitszeichen oder Doppelpunkte hinzugefügt.

3.4 BEWERTUNGSSPEICHERUNG

Die abgegebenen Bewertungen müssen persistent gespeichert werden, damit Nutzer darauf zugreifen können. Abschnitt 2.2.2 kommt zu dem Schluss, dass dafür ein zentraler Server eingesetzt werden kann, der diese Speicherung übernimmt. Zusätzlich gibt es auch die Möglichkeit, die Daten ohne diesen Signaturserver weiterzugeben. Der Server dient in erster Linie der effizienten Verbreitung der Bewertungen.

Zentraler Server: Um die im Bewertungsplugin TrustOSM angefallenen Bewertungsdaten auf dem zentralen Signaturserver zu speichern, bzw. bereits dort gespeicherte Bewertungen zu erhalten oder zu widerrufen muss Kommunikation zwischen dem Client und dem Server stattfinden. Diese ist in Abbildung 3.15 dargestellt. Sollen Bewertungen gespeichert werden, so wird der Server mittels der put() Mitteilung dazu aufgefordert. Das übergebene Argument ist eine Liste von

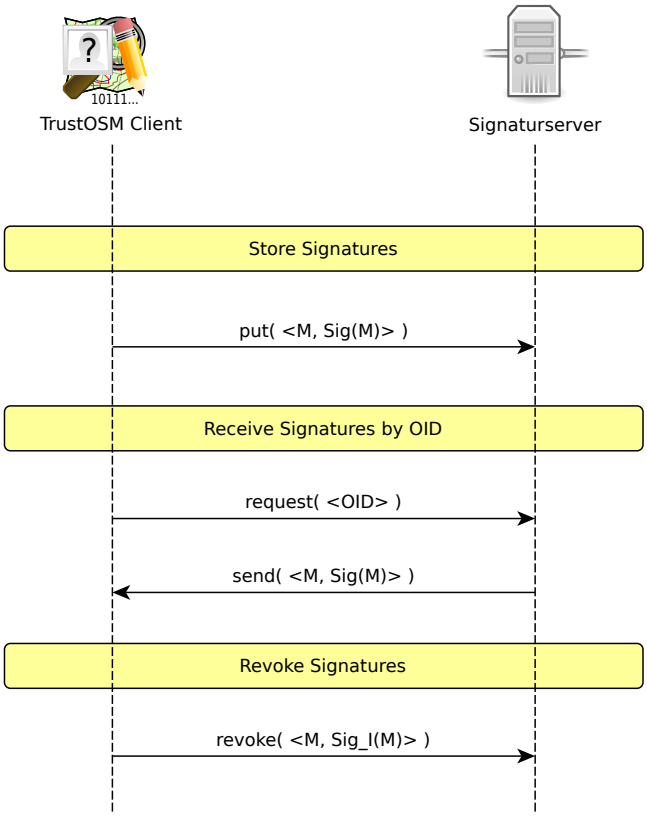

Abbildung 3.15: Kommunikation der OSM Client Applikation mit dem Signaturserver bei Speicherung bzw. Abfrage oder Widerruf der Bewertungen.

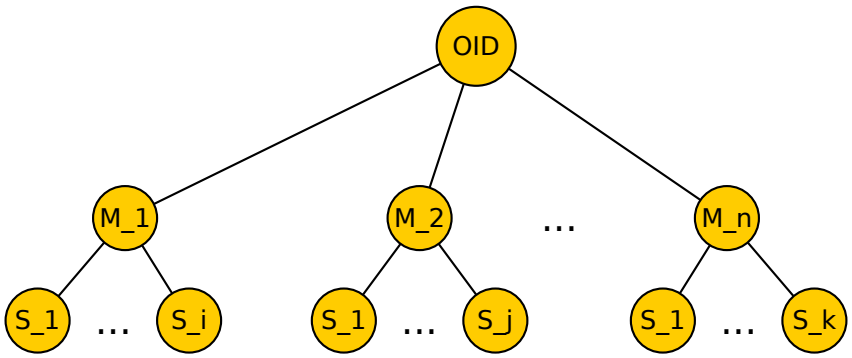

Abbildung 3.16: Struktur der Bewertungsspeicherung auf dem Signaturserver

Bewertungen, angedeutet durch die spitzen Klammern <>. Eine Bewertung besteht aus der Signaturnachricht M und der digitalen Signatur zu dieser Nachricht. Die Signaturnachricht M ist die Klartextnachricht, die aus dem Inhalt der konkreten OSM Daten gebildet wird, wie in Abschnitt 3.3 ausführlich beschrieben. Die digitale Signatur Sig(M) wird von einem Bewerter mit seinem geheimen Schlüssel über die Signaturnachricht M erstellt und dient zur Absicherung von M gegen Manipulation, sowie als Träger von Zusatzinformationen über die Bewertung, wie beispielsweise Toleranz oder Quellen der Prüfung etc. Aus der Signaturnachricht M lässt sich die OSM Identifikationsnummer (OID), zu der die Bewertungen gespeichert werden sollen, vom Server extrahieren. Sie ist bei der Speicherung wichtig, um auf Bewertungen eines bestimmten OSM Objektes schnell zugreifen zu können.

Zu einer OID können viele Signaturnachrichten vorliegen, da viele verschiedene Tags, Geometrien oder Members bei einem Objekt bewertet werden können. Auf dem Server müssen zu jeder Signaturnachricht beliebig viele Signaturen gespeichert werden können, denn verschiedene Bewerter können die selben Daten bewerten und somit die selbe Signaturnachricht M bilden und signieren. Abbildung 3.16 stellt die Speicherhirarchie auf dem Signaturserver entsprechend dar. Die Blätter der Baumstruktur sind die Signaturen S, die zu einer entsprechenden Nachricht M jeweils mit unterschiedlichen Signierschlüsseln erstellt worden sind. Wird die selbe Nachricht mit dem selben Schlüssel mehrmals signiert, so überschreibt die letzte Signatur die vorher gespeicherte. Mehrere Signaturen von dem selben Bewerter gleichzeitig zu speichern ist sinnlos. Für einen Bewerter kann es z.B. dann sinnvoll sein eine bereits abgegebene Signatur erneut zu tätigen, wenn er die in der Signatur gespeicherten Zusatzinformationen aktualisieren möchte.

Sollen zu einer gegebenen Liste von Openstreetmap-Objekten, gekennzeichnet durch ihre OID, alle vorliegenden Bewertungsinformationen abgerufen werden, so wird eine request() Anfrage mit der Liste der OIDs der Objekte an den Server geschickt. Wie in Abbildung 3.15 ersichtlich, sendet der Server daraufhin eine Liste von Bewertungen zurück, die ebenso strukturiert ist, wie die Liste der put() Mitteilung. Die Liste kann beliebig groß, je nachdem wieviele Bewertungen vorliegen und insbesondere auch leer sein, wenn zu allen gegebenen OIDs keine Bewertungen gespeichert wurden. Der Client kann mithilfe der in den Signaturnachrichten M gespeicherten OID, die Bewertungen den OSM Objekten seiner Anfrage zuordnen.

Eine weitere Operation, die ein Bewerter mit dem TrustOSM Client durchführen kann, ist der Widerruf einer bereits abgegebenen und auf dem Server gespeicherten Bewertung. Mit der revoke() Mitteilung wird der Signaturserver davon in Kenntnis gesetzt, dass Signaturen von bestimmten Signaturnachrichten für ungültig erklärt werden. Dazu wird eine spezielle Signatur Sig_I(M) über die entsprechende Signaturnachricht M angefertigt. Sig_I enthält dabei Zusatzinformationen, die anzeigen, dass es sich um eine Invalidierung einer bereits vorhandenen Signatur handelt. Mit der Nachricht M, der darin gespeicherten OID und Sig_I(M) kann die entsprechende zu widerrufende Signatur Sig(M) auf dem Signaturserver adressiert werden. Der Server muss dabei prüfen, dass

Sig(M) und Sig_I(M) mit dem gleichen Signierschlüssel erzeugt wurden. Somit wird sichergestellt, dass nur der Besitzer des Schlüssels, mit dem die zu widerrufende Signatur ursprünglich erzeugt wurde, eine Signatur und somit eine Bewertung widerrufen kann. Mit dem Vorzeigen der Signatur Sig_I(M) kann der Server weiteren Personen nachweisen, dass Sig(M) tatsächlich von seinem ursprünglichen Ersteller invalidiert wurde.
Neben der Prüfung des Schlüssels von Sig(M) und Sig_I(M) ist es außerdem von Bedeutung, dass Sig_I(M) neuren Datums als Sig(M) ist. Da das Datum ebenfalls als Zusatzinformation in der Signatur gespeichert ist, fällt diese Prüfung leicht. Möchte ein Bewerter, dass seine Signatur Sig(M) doch wieder gültig ist, so signiert er M einfach erneut und speichert die Signatur mittels einer gewöhnlichen put() Nachricht auf dem Server. Dadurch, dass die neu erzeugte Signatur Sig(M) aktueller ist als Sig_I(M), ist Sig_I(M) unwirksam und Sig(M) wieder gültig.

Lokale Speicherung: Mit TrustOSM ist es weiterhin möglich abgegebene Bewertungen lokal in eine Datei zu speichern. Die Bewertungsinformationen können somit auch ohne den Server und demzufolge auch offline an andere Personen weitergegeben werden. Die Informationen, die in die Datei gespeichert werden können, sind die selben, die auf den Server übertragen werden können. Wählt ein Bewerter diesen dezentralen Verbreitungsansatz seiner Bewertungen, muss er jedoch befürchten, dass vor allem ein Widerruf seiner Bewertung nicht alle Nutzer seiner Bewertungen erreicht. Bei einem zentralen Server, von dem alle Nutzer Bewertungen beziehen, ist dieses Risiko kleiner. Mit der Kombination von zentraler und dezentraler Weitergabe der Widerrufssignatur werden die meisten Nutzer erreicht.

3.5 IDENTITÄTSMANAGEMENT

Eine in Abschnitt 2.3.3 formulierte Sicherheitsanforderung beschäftigt sich mit der Privatsphäre des Bewerters. Bei der Abgabe einer Bewertung möchte der Bewerter selbst entscheiden, wieviele und welche Informationen er über sich preis gibt. Er soll insbesondere in der Lage sein Bewertungen anonym bzw. unter Nutzung eines Pseudonyms abzugeben. Nutzt er ein Pseudonym für genau eine Bewertung kann diese als anonym angesehen werden, da kein direkter Bezug zu anderen Bewertungen hergestellt werden kann.
Die Verwaltung von Pseudonymen mit möglicherweise verschiedenen verknüpften Informationen ist Aufgabe des Identitätsmanagements. Dieses betreibt der Bewerter selbst, das heißt, er kann sich eigene beliebige Identitäten anlegen und verwenden, wie er möchte. Dabei kann er von Identitätsmanagementsystemen (IDMS) unterstützt werden.
Eine Identität wird über das verwendete Schlüsselpaar aus Signier- und Testschlüssel mit einer Bewertung verknüpfbar. Dabei ist es gängige Praxis Informationen der Identität, wie z.B. Name, E-mailadresse oder Foto zusammen mit dem öffentlichen Schlüssel zu signieren und zu verbreiten.

Abbildung 3.17 verdeutlicht, wie die öffentliche Identität, die sich ein Bewerter zulegt, in das System eingebunden ist. Ein Bewerter kann sich eine öffentliche Identität erstellen und an ein Schlüsselpaar knüpfen, zu dem er den privaten Schlüssel besitzt. Kern dieser Identität ist der öffentliche Schlüssel. Zu jedem Zeitpunkt, auch nach Abgabe von Bewertungen, können die Eigenschaften der Identität vom Bewerter verändert werden. Einzig der öffentliche Schlüssel an sich muss unverändert bleiben. Zu beachten ist allerdings, dass Zertifikate von diesen Identitätsänderungen auch betroffen sein könnten und ihre Gültigkeit verlieren, sofern sie die geänderten Eigenschaften abdecken.
Damit zu einer abgegebenen Bewertung der zugehörige öffentliche Schlüssel gefunden werden kann, wird eine Referenz auf diesen im Signaturpaket der Bewertung abgespeichert. Diese Referenz ist auch als Fingerabdruck (fingerprint) bekannt und entsteht durch Anwendung einer Hashfunktion auf den öffentlichen Schlüssel. Dass zu einer Bewertung der passende öffentliche Schlüssel mit seinen Identitätsinformationen vorliegt, wird durch einen erfolgreichen Test der Signatur zur Signaturnachricht sichergestellt.

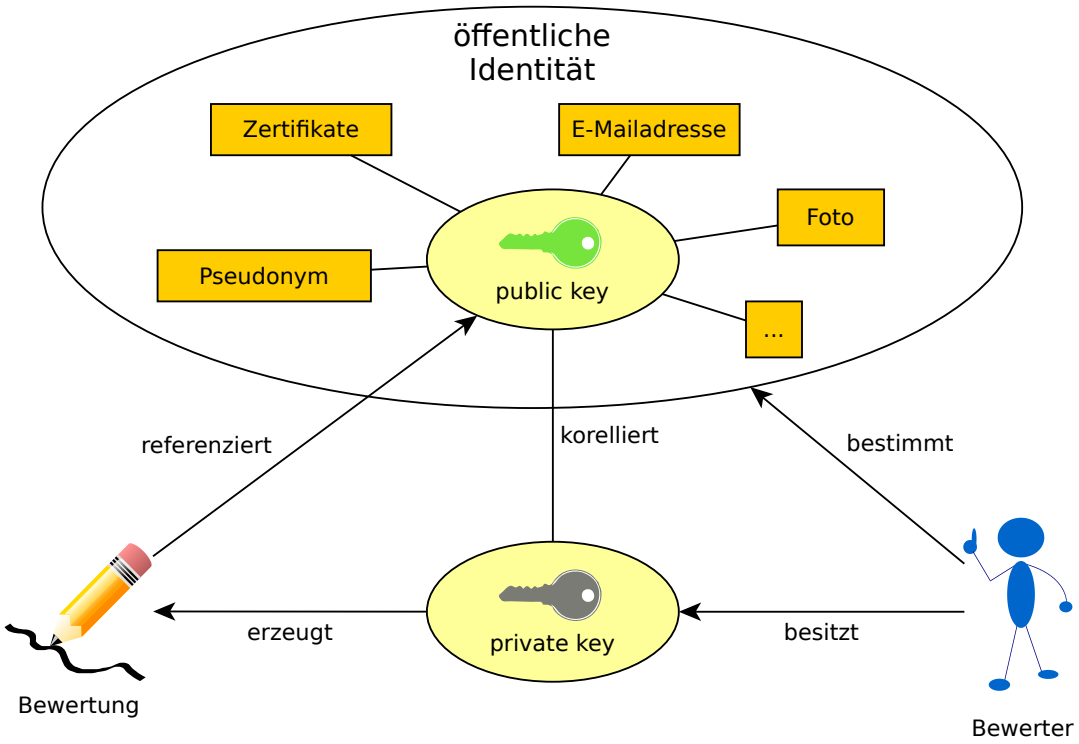

Abbildung 3.17: Zusammenhänge zwischen Bewerter, Identität, Schlüsselpaar und Bewertung.

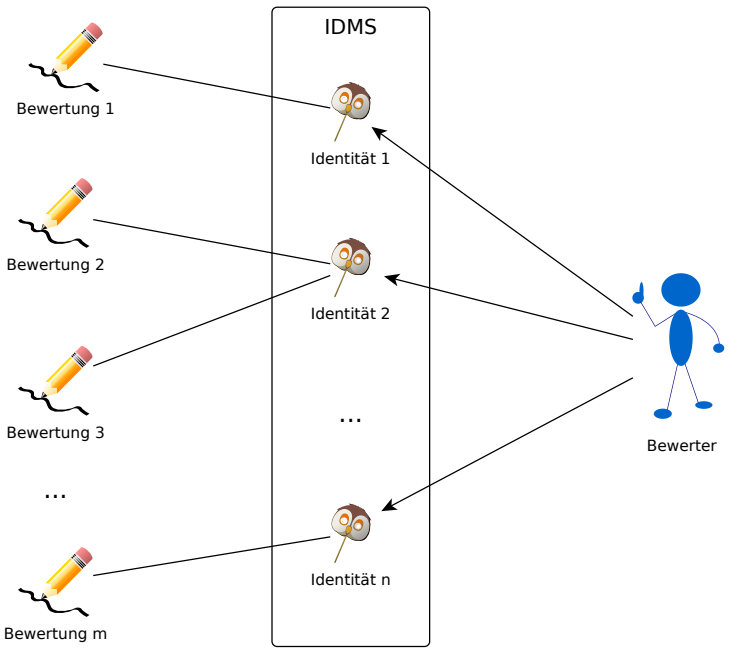

Abbildung 3.18: Ein Bewerter verwaltet mehrere Identitäten, mit denen er Bewertungen abgibt.

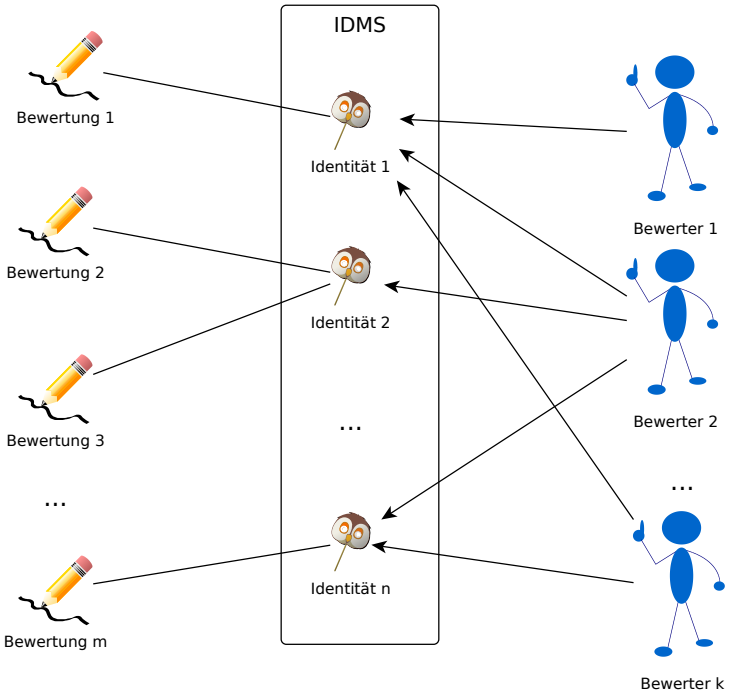

Abbildung 3.19: Mehrere Bewerter teilen sich Identitäten.

Möchte der Bewerter mehrere Identitäten für verschiedene Bewertungen einsetzen, wie in Abbildung 3.18 dargestellt, so lohnt sich der Einsatz eines Identitätsmanagementsystems. Mithilfe eines solchen Systems kann der Bewerter nicht nur Identitätsinformationen erstellen, sondern er kann auch Regeln definieren, wie und in welchem Zusammenhang eine Identität eingesetzt werden soll. Dabei kann der Bewerter verschiedene Rollen festlegen, die er z.B. abhängig von den Daten, die er gerade bewertet, einnimmt.
Während in Abbildung 3.18 genau ein Bewerter mehrere Identitäten nutzt, mit denen wiederum beliebig viele Bewertungen getätigt werden können, zeigt Abbildung 3.19 welche Möglichkeiten ein IDMS noch bietet. So könnten mehrere Benutzer die gleiche Identität nutzen, um Bewertungen zu erstellen. Das IDMS prüft in diesem Fall die Berechtigung eines Bewerters eine Identität zu nutzen.

Da eine Identität von demjenigen kontrolliert wird, der den privaten Schlüssel zum öffentlichen Schlüssel der Identität besitzt, richten sich Angriffe gegen den privaten Schlüssel gleichzeitig gegen die verbundene Identität. Wird ein geheimer Schlüssel gestohlen, kann dadurch die Identität missbraucht werden. Eine Identität kann aber auch unter Kooperation des Identitätsbesitzers durch Übergabe des geheimen Schlüssels an andere Personen weitergegeben werden.
Ein Löschen der Identität entspricht dem Verbreiten eines Widerrufszertifikates für den öffentlichen Schlüssel. Dabei wird der öffentliche Schlüssel nicht gelöscht, sondern lediglich für ungültig erklärt. Tätigkeiten, die im Namen der betroffenen Identität durchgeführt wurden oder noch werden, verlieren ganz oder teilweise ihre Gültigkeit, je nachdem ob der Zeitpunkt der Tätigkeit vor oder nach der Ausstellung des Widerrufs liegt und ob dies zweifelsfrei nachgewiesen werden kann.

Für diesen Entwurf ist das Identitätsmanagementsystem optional. Die Arbeit mit wenigen Identitäten und nur einem Benutzer kann auch direkt mit der Schlüsselverwaltung erfolgen.

3.6 SCHLÜSSELVERWALTUNG

Digitale Schlüssel sind bei mehreren Vorgängen im System erforderlich. Bei der Abgabe einer Bewertung wird ein geheimer Signierschlüssel des Bewerters benötigt. Bei der Auswertung einer Bewertung muss der passende öffentliche Testschlüssel vorliegen. Ein weiterer Einsatz für den geheimen Signierschlüssel ist das Erstellen von Zertifikaten zu öffentlichen Schlüsseln, wobei zum Prüfen des Zertifikates wiederum der passende öffentliche Testschlüssel vonnöten ist.
Die jeweils erforderlichen Schlüssel werden durch eine Schlüsselverwaltung zur Verfügung gestellt. Die Hauptaufgaben dieser Schlüsselverwaltung sind:

- Erzeugung von Schlüsselpaaren
- sichere Speicherung von privaten Schlüsseln
- Verbreitung von öffentlichen Schlüsseln
- Beschaffung von fremden öffentlichen Schlüsseln
- Erzeugung und Verbreitung von Widerrufszertifikaten

Die Schlüsselverwaltung kann dabei eng mit einem Identitätsmanagementsystem zusammen arbeiten. Dieses steuert dann Aufgaben, wie die Schlüsselpaarerzeugung bzw. die Auswahl geeigneter Schlüssel für die oben beschriebenen Vorgänge. Ohne das IDMS steuert der Bewerter die Schlüsselverwaltung direkt.

Bei der Erzeugung von Schlüsselpaaren sind zum einen die zusätzlichen Informationen anzugeben, die dem öffentlichen Schlüssel beigelegt werden, sowie die verwendeten Algorithmen und Schlüssellängen zu definieren, die mit diesem Schlüssel zum Einsatz kommen. Das Schlüsselpaar muss selbstverständlich zum digitalen Signieren geeignet sein.
Der Schutz der privaten Schlüssel eines Bewerters vor unberechtigtem Zugriff ist sehr wichtig für die Sicherheit des Systems. Die Schlüsselverwaltung kann nur einen Teil dazu beitragen, indem der private Schlüssel beispielsweise selbst mit einem symmetrischen Verfahren wie AES verschlüsselt und durch ein Passwort geschützt abgespeichert wird. Ein großer Teil des Schutzes ist allerdings vom Bewerter durch zusätzliche Maßnahmen zur Absicherung seines verwendeten Computersystems zu organisieren.
Der öffentliche Schlüssel kann von der Schlüsselverwaltung gemäß den Wünschen des Nutzers verbreitet werden. Geeignet sind beispielsweise öffentliche Schlüsselserver mit denen die lokale Schlüsselverwaltung kommunizieren kann. Öffentliche Schlüssel können sowohl auf die Server geladen als auch von dort bezogen werden. Anders als bei Signaturen zur Absicherung von Kommunikationsmitteilungen muss an dieser Stelle keine Prüfung des gelieferten öffentlichen Schlüssels durch den Nutzer stattfinden, indem beispielsweise auf anderen Kanälen mit dem Eigentümer des Schlüssels die Schlüssel ID verglichen wird. Der Schlüssel muss lediglich zu einer vorliegenden Bewertung passen.
Eine weitere Aufgabe der Schlüsselverwaltung liegt in der Erstellung und Verbreitung von Widerrufszertifikaten. Ein Widerrufszertifikat kann dabei Informationen über die Ursache des Widerrufs beinhalten, sowie welche Signaturen betroffen sein sollen. Dies kann alle mit dem zugehörigen Schlüssel erstellten Signaturen und demzufolge Bewertungen betreffen oder nur bestimmte ab einem gewissen Zeitpunkt oder aufgrund anderer Kriterien. Meist wird ein Widerrufszertifikat direkt nach Erzeugung eines Schlüsselpaares erstellt und getrennt vom privaten Schlüssel aufbewahrt. Dieses invalidiert bei Veröffentlichung alle Signaturen des benutzten Schlüssels.

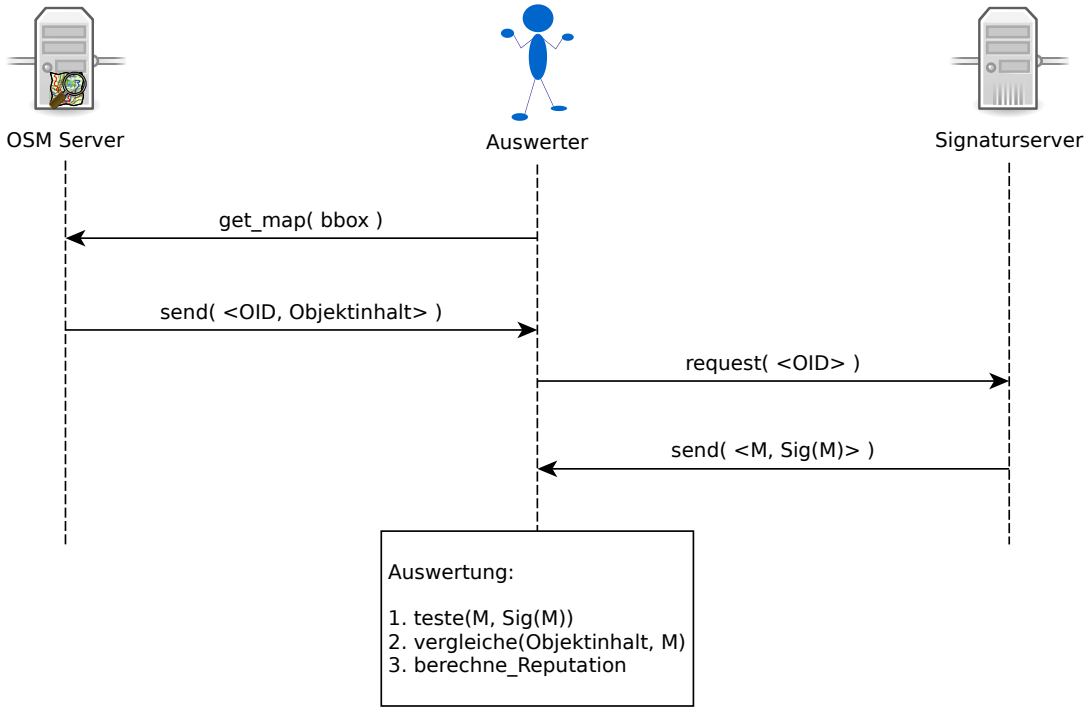

Abbildung 3.20: Ablauf einer Bewertungsauswertung

3.7 AUSWERTUNG DER BEWERTUNGEN

Bisher lag ein Hauptaugenmerk auf der Erstellung von Bewertungen durch die Bewerter. Nun soll die Auswertung der Bewertungen durch die Nutzer im Mittelpunkt stehen. Abbildung 3.20 stellt dabei die grundlegenden Abläufe vor und bei einer Auswertung dar. Der Auswerter, also der Nutzer der Openstreetmap-Daten möchte die Vertrauenswürdigkeit von Openstreetmap-Daten einschätzen. Als erstes benötigt er die Openstreetmap-Daten der interessierenden Region. Er führt daher eine get_map Anfrage an den Openstreetmap-Server durch, in der er ihm die Koordinaten des interessierenden Bereiches in Form einer rechteckigen Bounding Box (bbox) mitteilt. Der Server übermittelt daraufhin die ihm vorliegenden Openstreetmap-Daten in einer Liste mit Openstreetmap-Objekten, die durch ihre Openstreetmap-Identifikationsnummer (OID), sowie ihren restlichen Objektinhalt gekennzeichnet sind. Der Auswerter kann sich nun an den Signaturserver wenden und versuchen zu den ihn interessierenden Openstreetmap-Objekten Bewertungen abzufragen. Da er mit den OIDs der interessierenden Objekte durch den OSM Server bekannt gemacht wurde, kann er diese nun als Liste an den Signaturserver schicken. Dieser sendet daraufhin die Bewertungen zurück.

Nun kann der Nutzer mit der eigentlichen Auswertung der Bewertungen beginnen.

3.7.1 Signaturtest

Zunächst stellt er sicher, dass die ihm übermittelten Bewertungen nicht manipuliert wurden und testet, ob die Signatur im Signaturpaket Sig(M) zu der Signaturnachricht M passt. Den dazu benötigten Testschlüssel bezieht er aus seiner Schlüsselverwaltung. Ist diese Prüfung erfolgreich, so beginnt er den nächsten Schritt der Auswertung. Falls nicht, ist die Bewertung zu ignorieren und der Nutzer kann andere Personen auf diesen Umstand aufmerksam machen, um einen eventuellen Angriff aufzudecken.

3.7.2 Objektvergleich

Bei gültiger Signaturprüfung vergleicht der Auswerter die aktuellen Openstreetmap-Daten eines Objektes, wie er sie vom Openstreetmap-Server erhalten hat, mit den der Signaturnachricht entnehmbaren Daten. Sind sie gleich, so können die Daten des OSM Servers als durch die Bewertung bestätigt angesehen werden. Unterscheiden sie sich, so können sie dennoch in manchen Fällen als bestätigt betrachtet werden. Zusätzliche dem Signaturpaket entnehmbare Informationen, wie beispielsweise ein Toleranzbereich für Geometriedaten geben die Kriterien dafür aus Sicht des Bewerters an. Auch aus Sicht des Auswerters könnten Abweichungen von den ursprünglich bewerteten Informationen unter selbst gewählten Kriterien akzeptabel sein und für ihn die Bedeutung der Bewertung weiter bestehen lassen.

3.7.3 Reputationsfunktion

Nachdem festgestellt wurde, welche Bewertungen die OSM Daten bestätigen, so muss im letzten Schritt geprüft werden, wie vertrauenswürdig diese Bewertungen sind und welche Glaubwürdigkeit die aktuellen OSM Daten dadurch letztendlich bei dem Nutzer erwerben. Ausdruck dafür kann der Reputationswert sein. Miteinbezogen werden hierbei vor allem die Bewerterzertifikate, die mit dem öffentlichen Testschlüssel verknüpft sind. Aus ihnen kann ein Reputationswert für die Bewerter gebildet werden, der in die Berechnung des Reputationswertes der Daten miteinfließt. Wie diese Reputationswerte erstellt werden, wird durch die Reputationsfunktion geregelt, welche im Folgenden näher beschrieben werden soll.

Die Wahl der Reputationsfunktion bestimmt, welche Aspekte der abgegebenen Bewertungen in die Gesamteinschätzung des Datenobjektes mit welcher Bedeutung eingehen.

Es wäre denkbar, dass der Nutzer für sich selbst eine informelle Reputationsfunktion benutzt, dass heißt er betrachtet einige oder alle abgegebenen Bewertungen und entscheidet anhand der vorliegenden Informationen, wie sehr er auf die Richtigkeit des Datenobjektes vertrauen möchte. Die Reputationsfunktion kann er dabei auch spontan und nach Gefühl bilden. Diese Form der Bewertungsaggregation ist sehr flexibel und kann komplexere Zusammenhänge einbeziehen. Beispiel 3.2 zeigt, dass schwer abzubildende Fakten, wie Freundschaft leicht vom Nutzer berücksichtigt werden können.

Beispiel 3.2 (Persönliche Beziehung) *Ein Restaurant ist in Openstreetmap mit der Eigenschaft „cuisine=vegan" ausgestattet und dieser Fakt, dass es dort hauptsächlich vegane Speisen gibt, wurde von mehreren Bewertern signiert. Ein Nutzer, der das Restaurant nicht kennt, möchte wissen, ob es dort tatsächlich veganes Essen gibt und prüft die abgegebenen Bewertungen. Ihm fällt auf das eine Signatur, die seines Freundes ist. Da er diesem aufgrund einer persönlichen Beziehung vertraut, sind die restlichen Signaturen für ihn von geringerer Bedeutung und er gewichtet die Aussage seines Freundes höher und vertraut somit den Daten.*

Allerdings erfordert die manuelle Bewertungsaggregation mehr Aufwand des Nutzers, da dieser sich selber über die Details der Bewertungen informieren muss. Zudem sind die Ergebnisse schlecht vergleichbar, da der Nutzer bei gleicher Faktenlage zu unterschiedlichen Ergebnissen kommen kann. Wächst die Anzahl der Bewertungen und damit auch die Anzahl der zu untersuchenden Informationen, kann der Nutzer schnell den Überblick verlieren und wesentliche Informationen könnten übersehen werden. Die Skalierbarkeit ist also bei manuellen Verfahren eher schlecht.

Um aus den abgegebenen Bewertungen eine automatisierte Abschätzung der Richtigkeit der Daten zu generieren, werden die Informationen der Bewertungen auf Zahlenwerte abgebildet.

Dabei muss klar definiert werden, welche Informationen in die Berechnung eingehen und was sie bedeuten. Ziel der Berechnung ist die Zahlenwerte der Bewertungen auf einen aggregierten Zahlenwert zu bringen, der als ein Maß für die Vertrauenswürdigkeit des bewerteten Datums interpretiert werden kann. Dieser Wert soll im Folgenden R heißen. Für den Nutzer ist dieser Wert lediglich als Hilfestellung bei der Entscheidung zu verstehen, ob er den Daten vertrauen möchte oder nicht. Er kann durch zusätzliche Informationen andere Erkenntnisse gewinnen, als der berechnete Zahlenwert vermittelt. Wenn die Reputationsfunktion flexibel genug ist, reicht es womöglich einige Parameter zu ändern, um die eigenen Informationen einzuarbeiten. Reputationsfunktionen, die ohne zusätzliche durch den Nutzer zu definierende Parameter auskommen, könnten auch als globale Reputationsfunktionen eingesetzt werden, das heißt, sie könnten für jeden Nutzer gleiche Ergebnisse liefern und die Daten somit objektiv hinsichtlich ihrer Vertrauenswürdigkeit vergleichbar machen.

Zunächst ist es notwendig die zur Verfügung stehenden Informationen der Bewertung auf Zahlenwerte abzubilden. Die möglichen Aussagen von Bewertungen werden durch das Bewertungssystem vorgegeben. Findet eine Bewertung in Form einer digitalen Signatur statt, so kann der Bewerter mit seiner Signatur nur ausdrücken, dass er das betreffende Datum für korrekt befindet. Entdeckt er einen Fehler so kann er diesen entweder sofort selber korrigieren und anschließend das korrigierte Datum signieren oder einen Fehler an dieser Stelle mithilfe von Openstreetbugs markieren und den Fehler (bug) signieren oder nichts tun.

Jede Bewertung i bekommt nun eine Zahl $x_i \in \{0, 1\}$ zugewiesen. Dabei ist $x_i = 1$, wenn ein Openstreetmap-Datum signiert wurde und $x_i = 0$, wenn ein Openstreetbug signiert wurde.
Der Wert $x_i = 1$ entspricht demnach der Aussage: „Die vorhandenen Daten sind korrekt." bzw. „Die vorhandenen Daten wurden korrigiert und sind jetzt korrekt." und $x_i = 0$ bedeutet: „Die vorhandenen Daten sind nicht korrekt. Ein Fehler wurde gefunden, aber noch nicht behoben." Die Zuordnung von bug zu Daten erfolgt, wie bei Openstreetbugs üblich über die Geokoordinaten des Bugs und der Daten und lässt sich damit nur unscharf zuordnen, da nicht klar ist auf welche Tags sich der bug genau bezieht bzw. auf welches Objekt genau, denn es können auch mehrere Objekte in einen bug involviert sein. Die genau Beschreibung im Kommentarfeld des bugs lässt das Problem aber meist eindeutig vorhandenen Daten zuordnen.
Zur Vereinfachung wird an dieser Stelle angenommen, dass die Zuordnung zwischen bug und Openstreetmap-Daten sowie umgekehrt klar und objektiv auszumachen ist und zu einem Openstreetmap-Datum immer alle positiven Bewertungen $x_i = 1$ und alle negativen Bewertungen $x_i = 0$ in Form der signierten bugs vorliegen.

Neben der eigentlichen Bewertungsaussage können über in der Signatur gespeicherte Zusatzattribute dem Nutzer weitere Informationen zugänglich gemacht werden. Ein Beispiel ist ein beigelegtes Zertifikat oder Kommentare des Bewerters. Deren Umsetzung in Zahlenwerte kann bei den Reputationsfunktionen unterschiedlich gehandhabt werden und ist Teil des Reputationsalgorithmus.

Im Folgenden sollen einige beispielhafte Reputationsfunktionen vorgeschlagen und diskutiert werden, die für ein Reputationssystem für Openstreetmap in Frage kommen. Dabei ergibt sich der Reputationswert R immer als Funktion f von der Menge aller abgegebenen Bewertungen $\{x_i\}$. Die Reihenfolge der Verarbeitung der Bewertungen sollte dabei keine Rolle spielen. Der Reputationswert berechnet sich also allgemein mit:

$$R = f(\{x_i\})$$

Mittelwert

Eine sehr einfache Reputationsfunktion bildet aus allen N abgegebenen Bewertungen den Mittelwert.

$$f(\{x_i\}) = \frac{\sum_i x_i}{N}$$

Wenn keine Bewertung abgegeben wurde, soll der Wert $f(\emptyset) = 0{,}5$ sein.

Da für x_i nur 0 oder 1 möglich sind kann der Mittelwert auch nur im Intervall zwischen 0 und 1 liegen. Liegt er näher an der 1 oder ist er genau 1, so ist das Datum eher vertrauenswürdig und liegt er näher an der 0 oder ist genau 0 so ist das Datum vermutlich fehlerhaft.

Stehen den positiven genausoviele negative Bewertungen gegenüber, ist der Reputationswert genausogroß, als wäre keine Bewertung abgegeben worden. Werden nur negative oder nur positive Bewertungen abgegeben ist der Reputationswert genau 0 bzw. genau 1. Die Gesamtanzahl der Bewertungen spielt dabei keine Rolle. Ebensowenig fließen hierbei zusätzliche Informationen etwa über die Bewerter mit ein.
Der besondere Reiz liegt in der Einfachheit und Übersichtlichkeit der Reputationsfunktion, sowie deren Unabhängigkeit von lokalen Parametern des Nutzers. Dies ermöglicht ihren Einsatz als globale Reputationsfunktion.
Ihre Aussagekraft ist jedoch begrenzt zum einen durch das Fehlen der Bewerterreputation, also den Zusatzinformationen über die Bewerter, zum anderen wegen des fehlenden Einflusses der Bewertungsanzahl.

Summation

Um die Gesamtzahl der Bewertungen zu würdigen, wäre auch eine einfache Summation der x_i möglich. Ein Objekt, welches drei positive Bewertungen hat, sollte demnach einen höheren Reputationswert bekommen, als eines mit nur einer positiven Bewertung. Bei der Summation sollen die negativen Bewertungen den Reputationswert um eins senken, während die positiven ihn um eins erhöhen. Eine Reputationsfunktion, die dies bewirkt kann so formuliert werden:

$$f(\{x_i\}) = \sum_i (2x_i - 1)$$

Ein nicht bewertetes Objekt hat den Reputationswert $f(\emptyset) = 0$. Die Bildmenge von f sind hierbei jedoch alle ganzen Zahlen. Es gibt keine Beschränkung.
Auch hier eignet sich die Reputationsfunktion als globale Funktion, da wiederum keine weiteren Parameter zu setzen sind. Wie auch beim Mittelwert mangelt die Aussagekraft jedoch an der fehlenden Bewerterreputation. Wenige kompetente Bewerter könnten somit von vielen weniger kompetenten Bewertern überstimmt werden.

Gewichteter Mittelwert

Damit die Reputationsfunktion differenzierter zwischen den Bewertungen verschiedener Bewerter unterscheiden kann, müssen die dafür notwendigen zusätzlichen Informationen auf Zahlenwerte abgebildet und anschließend in die Berechnung mit aufgenommen werden. Die Erstellung und Auswertung der Bewerterreputation ist keineswegs trivial und ein größeres Problem für sich. An dieser Stelle soll die vereinfachte Betrachtungsweise genügen, dass jeder Bewertung x_i eine Bewerterreputation b_i zugeordnet werden kann, die angibt wie glaubwürdig der Bewerter i mit seiner Bewertungsaussage ist. Dabei soll weiterhin gelten, dass $b_i \in \mathbb{R}, 0 \leq b_i \leq 1$ ist. Je näher der Wert an der 1 liegt, desto eher sagt der entsprechende Bewerter bei seiner Bewertung die Wahrheit. Wird einem Bewerter die Bewerterreputation 0 zugeordnet, so lügt er immer, ist es die 1, so entspricht seine Aussage immer der Wahrheit.

Um die Bewertungen x_i zu gewichten werden sie zunächst auf $-\frac{1}{2}$ und $\frac{1}{2}$ abgebildet durch Subtraktion von $\frac{1}{2}$. Anderenfalls hätten die Gewichte bei positiven Bewertungen andere Auswirkungen als bei negativen. Die Gewichte sind Faktoren vor den verschobenen Bewertungsaussagen. Diese Faktoren sind die ebenso um $\frac{1}{2}$ verschobenen b_i . Nach Gewichtung und Normierung auf die Anzahl N der Bewertungen wird das Ergebnis wieder von dem Intervall $[-\frac{1}{2}...\frac{1}{2}]$ durch Addition von $\frac{1}{2}$ auf das Intervall [0..1] zurückverschoben.

$$f(\{x_i\}) = \frac{\sum_i \left((b_i - \frac{1}{2})(x_i - \frac{1}{2})\right)}{N} + \frac{1}{2}$$

Interessant ist, dass ein guter Lügner mit $b_i < \frac{1}{2}$, von dem also bekannt ist, dass er lügt, mit seiner Aussage x_i ebenso in die Berechnung eingeht, als würde er das Gegenteil $(1 - x_i)$ behaupten und wäre dabei mit dem Wert $(1 - b_i)$ glaubwürdig. Das gilt natürlich ebenso für einen glaubwürdigen Bewerter.

Denn es gilt:

$$\begin{aligned}((1 - b_i) - \frac{1}{2})((1 - x_i) - \frac{1}{2}) &= (\frac{1}{2} - b_i)(\frac{1}{2} - x_i) \\ &= -(-\frac{1}{2} + b_i) \cdot (-(-\frac{1}{2} + x_i)) \\ &= (b_i - \frac{1}{2})(x_i - \frac{1}{2})\end{aligned}$$

Ein Grundproblem der Aussagekraft des gewichteten Mittelwertes besteht darin, dass die Wirkung von sehr glaubwürdigen Bewertungen durch die von weniger glaubwürdigen aufgrund der Mittelung abgeschwächt wird.

Reputationsberechnung mittels Wahrscheinlichkeiten

Fasst man das durch den Reputationswert ausgedrückte Vertrauen als eine Wahrscheinlichkeit auf die angibt, ob ein Bewerter i mit der Einschätzung x_i der Daten recht behält, so lässt sich im Falle einer binären Einschätzung das Gesamtvertrauen $P(X|\{x_i\})$ darauf, dass eine bewertetes Openstreetmap-Datum unter der Einschätzung der Bewerter richtig ist, angeben:

$$P(X|\{x_i\}) = \frac{1}{1 + \frac{1-P(X)}{P(X)} \cdot \prod_i \frac{P(x_i|1-X)}{P(x_i|X)}}$$

Dabei gilt $x_i \in \{0, 1\}$.
Mit $P(X)$ wird die A-priori-Wahrscheinlichkeit bezeichnet, dass die Openstreetmap-Daten korrekt sind. Die Wahrscheinlichkeiten $P(x_i|1)$ und $P(x_i|0)$ drücken aus, wie gut eine Person in der Lage ist richtige bzw. fehlerhafte Daten als solche zu erkennen.

Eine Herleitung der Formel, sowie Beispielberechnungen finden sich im Anhang A.1.

Annahmen:

Im Folgenden werden ein paar Annahmen für die Berechnung des Reputationswertes diskutiert. Mit der Beschränkung auf nur zwei Ereignisse, wird unterstellt, dass es möglich ist ein bestimmtes Openstreetmap-Datum als „richtig" oder „falsch" einzuordnen. Diese Einordnung muss objektiv sein und die Einschätzungen von Dritten müssen objektiv verifizierbar sein. Eine allgemeine Verständigung auf grundlegende Regeln, so wie so zum Beispiel im Map Features Katalog im Openstreetmap-Wiki zu finden sind, ist dazu notwendig.
Leider herrscht in der Praxis ab und zu Uneinigkeit, ob ein gewisses Openstreetmap-Datum als richtig angesehen werden kann. In diesem Fall muss zunächst klargestellt werden unter welchen Annahmen und Regeln eine Einschätzung getroffen wurde. Für einen Nutzer, der indirektes Vertrauen durch dritte Personen sammeln will, müssen die Ansichten, wann etwas als „richtig" oder „falsch" zu bewerten ist abgeglichen werden, sonst ist die Einschätzung nutzlos.

Ein weiteres Problem stellt die Schätzung der A-priori-Wahrscheinlichkeit $P(X)$ dar. Die genaue Fehlerrate der Daten kennt sicher niemand, da keiner in der Lage ist, alle Openstreetmap-Daten zu verifizieren. Allerdings kann $P(X)$ durch repräsentative Stichproben angenähert werden. Der Rahmen in dem die Stichprobe stattfindet, also deren Grundgesamtheit, kann dabei allerdings eine beliebige Teilmenge der vorhandenen Openstreetmap-Daten sein. Eine repräsentative Stichprobe für die gesamte Welt ist schon aufgrund der sehr unterschiedlichen Datenlage nur bedingt nützlich in einem kleineren Bereich. Es gibt Gebiete mit hoher Mapperaktivität, z.B. Ballungszentren wie Städte und Gebiete mit niedriger Aktivität z.B. ländliche Regionen. Es gibt Gebiete, die hauptsächlich aus importierten Daten bestehen und somit schon eine hohe Datendichte, aber eventuell trotzdem kaum aktive Mapper besitzen z.B. die USA mit dem Tiger-Import[3]. Somit können die fehlerverursachenden Faktoren von Ort zu Ort verschieden sein.
Die Fehlerrate der Openstreetmap-Daten kann aufgrund der großen Vielfalt zwischen Gebieten unterschiedlich schwanken. Es ist also zum einen aufgrund der besseren Durchführbarkeit und zum anderen aufgrund der höheren Genauigkeit sinnvoll, die Fehlerrate nur für ein kleineres Gebiet wie zum Beispiel die Stadt Dresden zu ermitteln. Die Auswahl der Grundgesamtheit der Stichprobe kann neben geographischen auch thematische Kriterien berücksichtigen. So könnte beispielsweise die Fehlerwahrscheinlichkeit für Straßennamen oder Briefkästen in Openstreetmap geschätzt werden.
Zu berücksichtigen ist, dass die A-priori-Wahrscheinlichkeiten, die nur auf Grundlage eines eingeschränkt betrachteten Bereichs der Openstreetmap-Daten ermittelt wurden, nur in diesem Bereich gelten können, sofern keine statistische Unabhängigkeit zu einem anderen Bereich nachgewiesen werden kann.

Die eigentliche Grundlage für die Berechnung, der Wahrscheinlichkeit, dass die Openstreetmap-Daten korrekt sind unter Berücksichtigung der abgegebenen Bewertungen, bilden die jeweiligen Wahrscheinlichkeiten, dass eine dritte Person richtige von falschen Daten unterscheiden kann. Die Auffassung, dass diese Wahrscheinlichkeiten Ausdruck des indirekten Vertrauens sind, ist zunächst gewöhnungsbedürftig. Dabei ist es wichtig herrauszustellen, dass sich das Vertrauen nicht etwa auf die dritte Person, sondern auf die Daten bezieht, welche einzuschätzen sind.
Angenommen eine dritte Person hätte eine bestimmte Wahscheinlichkeit richtige Daten als solche zu erkennen also $P_0(x_i = 1|X = 1)$ bzw. falsche Daten als falsche also $P_0(x_i = 0|X = 0)$. Der Index 0 signalisiert, dass es sich nicht um die Wahrscheinlichkeiten $P(x_i = 1|X = 1)$ und $P(x_i = 0|X = 0)$ handelt, die der Nutzer jeder Person i zuordnet.
Im Folgenden sollen mit P die beiden Wahrscheinlichkeiten $P(x_i = 1|X = 1)$ und $P(x_i = 0|X = 0)$

3. TIGER-Import USA, http://wiki.openstreetmap.org/wiki/TIGER

bezeichnet werden und mit P_0 analog die Wahrscheinlichkeiten $P_0(x_i = 1|X = 1)$ und $P_0(x_i = 0|X = 0)$.
Die Wahrscheinlichkeiten P_0 sind dem Nutzer zunächst nicht bekannt und er kann lediglich Annahmen P darüber treffen. Je mehr Informationen einem Nutzer über die dritte Person vorliegen, desto besser wird es ihm möglich sein die Einschätzung P an die vorhandene Grundwahrscheinlichkeit P_0 anzunähern. Im Idealfall $P = P_0$ ist also das Vertrauen, was ein Nutzer mithilfe der dritten Person i in die Daten bekommen kann so hoch, wie die Wahrscheinlichkeit das diese Person die Daten richtig einschätzt.
Die Informationen, die ein Nutzer über eine dritte Person bekommen kann, können sehr vielfältig sein. So ist es beispielsweise interessant, wie oft ein Nutzer in der Vergangenheit mit seinen Aussagen richtig gelegen hat. Im einfachsten Fall könnte aus dieser relativen Häufigkeit der richtigen Prognosen im Verhältnis zu den abgegebenen Prognosen, die Wahrscheinlichkeiten P geschätzt werden. Liegen diese relativen Häufigkeiten nicht vor, weil ein Nutzer auf seine Privatsphäre achtet oder einfach noch nichts bewertet hat, so könnten andere Informationen zur Schätzung der Wahrscheinlichkeiten P beitragen. Kann ein Bewerter beispielsweise ein Zertifikat vorlegen z.B. ein „OSM-Stammtischzertifikat", so könnte die Wahrscheinlichkeit über die Häufigkeiten, dass ein OSM-Stammtischmitglied die Daten richtig einschätzt angenähert werden.
Ziel des Nutzers ist es, die Wahrscheinlichkeiten P, die er den jeweiligen Bewertern zuordnet mit den zur Verfügung stehenden Informationen so gut es geht den Wahrscheinlichkeiten P_0 anzunähern. Alle Informationen eines Bewerters, die dazu beitragen, können als Reputation des Bewerters angesehen werden.
Hat ein Nutzer keine Informationen über einen Bewerter so können allgemeine Statistiken über Bewerter für eine Grundschätzung der Wahrscheinlichkeiten P zurate gezogen werden. Dies können beispielsweise die relativen Trefferhäufigkeiten aller Bewerter sein. Alternativ kann auch eine pessimistische Annahme getroffen werden, die dem Nutzer zufälliges bewerten unterstellt also $P(x_i = 1|X = 1) = 0{,}5$ und $P(x_i = 0|X = 0) = 0{,}5$.

3.8 ZUSAMMENFASSUNG ENTWURF

Der Entwurf stellt hauptsächlich dar, aus welchen Komponenten das entwickelte Bewertungssystem für Openstreetmap-Daten besteht und wie diese aufgebaut sind und genutzt werden können.
In Abschnitt 3.1 werden die Komponenten eingeführt und kurz ihre Aufgabe genannt, um einen ersten groben Überblick über das System zu gewähren.
Abschnitt 3.2 erläutert die Bedeutung des Openstreetmap-Servers für das Bewertungssystem.
Ein Kernbestandteil des Bewertungssystems ist das Signatursystem, welches in Abschnitt 3.3 daher ausführlich diskutiert wird. Im Rahmen des Signatursystems wird festgelegt, welche Inhalte in welcher Form in die Bewertung miteinfließen und diese Festlegung begründet. Dabei wird erläutert wie die semantischen Einheiten gebildet werden können und wie diese in Signaturnachrichten bewertet werden. Da es in direktem Zusammenhang steht, ob Bewertungen bei der Auswertung als gültig angesehen werden, wird auch dieser Aspekt miteinbezogen.
Das Entwerfen der Signaturnachrichten brachte Probleme mit sich, deren Ursache in dem aktuellen Openstreetmap-Datenmodell zu finden sind. Die Probleme werden in Abschnitt 3.3.4 erklärt. Zudem wird im gleichen Abschnitt ein alternatives hierarchisches Datenmodell für Openstreetmap betrachtet, was als Idee zur Verbesserung von Openstreetmap für die Zukunft zu verstehen ist.

Die erstellten Bewertungen müssen gespeichert und verteilt werden, womit sich thematisch Abschnitt 3.4 befasst.
Damit ein Bewerter seine Privatsphäre schützen kann, beschreibt Abschnitt 3.5 die mögliche Benutzung eines Identitätsmanagementsystems. Dieses ist für das Bewertungssystem an sich optional.
Notwendig ist jedoch eine Schlüsselverwaltung, die direkt durch das Identitätsmanagementsystem gesteuert oder direkt durch den Bewerter genutzt werden kann, wie Abschnitt 3.6 ausführt.

Abschnitt 3.7 erläutert den Prozess der Auswertung von Bewertungen. Dieser ist in drei Stufen geteilt, die nacheinander durchgeführt werden. Zunächst wird die Signatur geprüft, anschließend wird das bewertete Objekt mit dem zugehörigen aktuell in Openstreetmap vorhandenen verglichen, um eine Gültigkeit der Bewertung festzustellen und zuletzt wird ein Reputationswert berechnet. Für die Gestaltung einer Reputationsfunktion für Openstreetmap-Daten liefert der Unterabschnitt 3.7.3 verschiedene Ideen. Eine weiter ausgeführte Idee basiert auf der Verwendung der Wahrscheinlichkeitsrechnung und wurde im Anhang A.1 hergeleitet.

4 IMPLEMENTIERUNG

Im Rahmen dieser Diplomarbeit wurde der Prototyp einer Applikation entwickelt mit der Bewertungen für Openstreetmap-Daten abgegeben, ausgewertet und gespeichert werden können. Die Applikation heißt TrustOSM, benutzt OpenPGP für den Umgang mit digitalen Signaturen und wurde als Plugin für den verbreiteten Openstreetmap-Editor JOSM[1] programmiert.
Die Software (sowohl Editor als auch Plugin) steht unter der GNU General Public License[2] Version 2.
Der Quellcode des Plugins, sowie eine lauffähige Version des Editors liegen der Diplomarbeit auf einer CD-Rom bei. Die neueste Version des Quellcodes des TrustOSM-Plugins kann aus dem Subversion-Repository[3] des Openstreetmap-Projektes ausgecheckt werden.

In diesem Kapitel werden die verwendeten Technologien sowie wichtige Details der Implementierung des Plugins erläutert.

4.1 JOSM

JOSM ist ein in der Programmiersprache Java geschriebener Editor für Openstreetmap-Daten. JOSM wurde 2005 von Immanuel Scholz[4] entwickelt und wird seitdem stetig von der Community erweitert. Aktuell wird das Projekt von Dirk Stöcker betreut und von mehreren Programmierern in ihrer Freizeit gepflegt. Der Programmcode ist frei zugänglich und steht unter der GNU General Public License Version 2. Der Editor erfreut sich großer Beliebtheit bei den Mappern von Openstreetmap und bietet sehr viele verschiedene Möglichkeiten, gesammelte Geodaten zu verarbeiten und Daten für Openstreetmap zu erzeugen. Einige Grundfunktionen sind:

- Download/Upload von Openstreetmap-Daten vom/zum Openstreetmap-Datenbankserver
- offline Anlegen, Verändern, Löschen, Anzeigen, Analysieren und Exportieren der Daten
- Erweiterung mit neuen Funktionen über Pluginschnittstelle, z.B. Luftbilder (WMS-plugin), Validierung (validator-plugin), etc.

1. Java OSM Editor, http://josm.openstreetmap.de/
2. GPL v2, http://www.gnu.org/licenses/gpl-2.0.html
3. TrustOSM im OSM Repository, http://svn.openstreetmap.org/applications/editors/josm/plugins/trustosm/
4. I. Scholz, Mitarbeiterprofil TU Dresden, 2009, http://www.inf.tu-dresden.de/index.php?node_id=2043&ln=de

Da über die Pluginschnittstelle viele Funktionen des Editors benutzbar sind, die für das Signatursystems benötigt werden, wurde für diese Diplomarbeit ein Prototyp einer Signatursoftware für Openstreetmap als JOSM-Plugin entwickelt.

Die für das Bewertungssystem wichtigen Funktionen sind:

- Bereitstellung einer Basis-GUI zum Verwalten von Openstreetmap-Daten
- Einfacher Transfer der Openstreetmap-Daten vom und zum Server
- Selektion von Openstreetmap-Daten durch den Nutzer

Viele der Funktionen von JOSM, sowie Hinweise zur Installation und grundlegenden Bedienung des Editors finden sich im Openstreetmap-Wiki[5]. Die nachfolgenden Abschnitte beschäftigen sich daher mit der konkreten Einbindung und Nutzung des in dieser Diplomarbeit entwickelten Plugins.

4.1.1 Installation des Plugins

Das Plugin wird in Form eines Jar-Pakets bereitgestellt, das alle für die Ausführung des Plugins benötigten Dateien enthält. Das Jar-Paket wird in das Plugin-Verzeichnis des JOSM Editors kopiert (unter Linux z.B. ~/.josm/plugins/) und beim Start des Editors erkannt und entpackt. Um das Plugin zu verwenden, muss es im Editor zusätzlich aktiviert werden. Nach einem Neustart des Editors wird das Plugin geladen und steht anschließend zur Verfügung. Es folgt eine Schritt-für-Schrittanleitung zum Aktivieren und Nutzen des Plugins. Voraussetzung sind die trustosm.jar-Datei im Plugin-Verzeichnis von JOSM, sowie eine funktionierende Installation des Editors selbst. Nach dem Start des Editors sind folgende Schritte notwendig:

1. JOSM Einstellungsdialog aufrufen
2. Pluginverwaltung anwählen
3. trustosm aktivieren, indem Haken in Checkbox gesetzt wird
4. JOSM beenden und neu starten
5. Fenster zur Erstellung der Signaturen einblenden
6. Das erscheinende Signaturfenster nutzen

Die für die Aktivierung des Plugins nötigen Schaltflächen in JOSM sind in Abbildung 4.1 durch entsprechende Zahlen markiert.

4.1.2 Nutzung des Plugins

Nach erfolgreicher Aktivierung des Plugins stehen folgende neue Funktionalitäten zur Verfügung:

- Es ist neuer Dateityp (*.txml) im Öffnen-Dialog des Editors verfügbar, über den in einer Datei gespeicherte Bewertungen importiert werden können.
- Ein neues GPG-Menü ermöglicht das Exportieren von Bewertungen in eine XML-Datei.
- Im Einstellungen-Dialog des Editors erscheint ein neuer Menüpunkt, der die Konfiguration des Plugins durch den Nutzer ermöglicht.

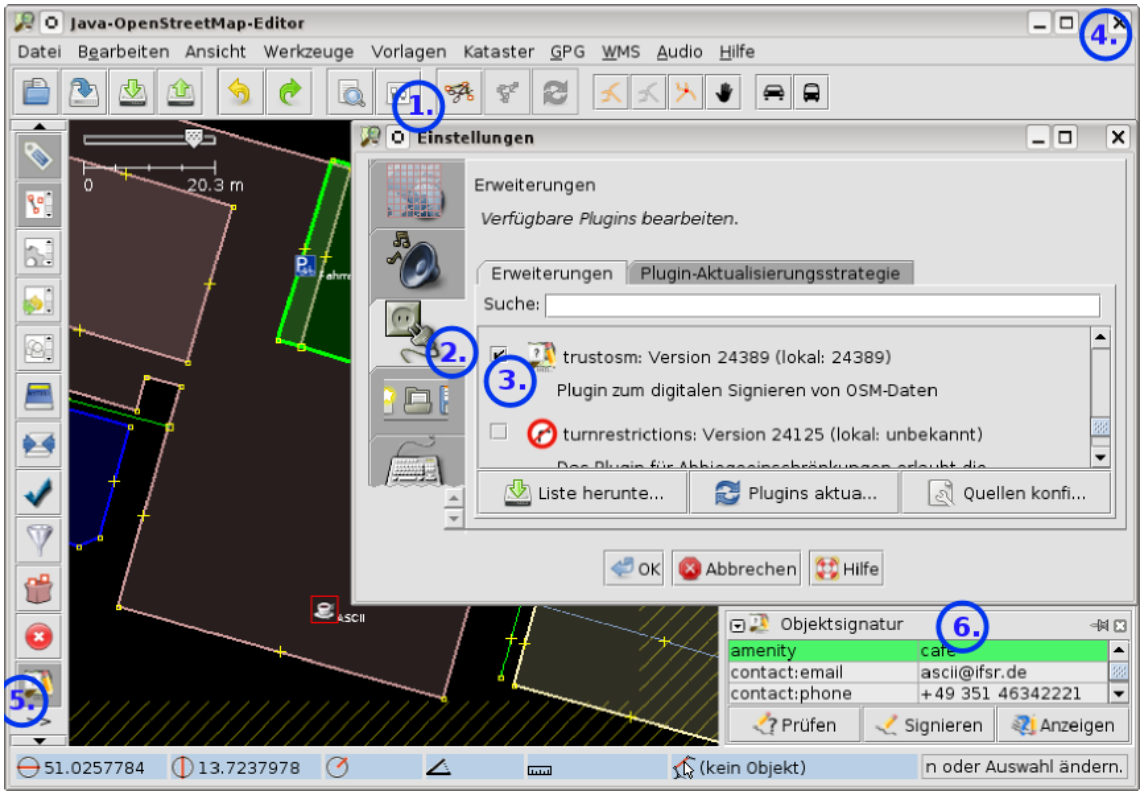

Abbildung 4.1: Schritte zur Aktivierung des trustosm-Plugins.

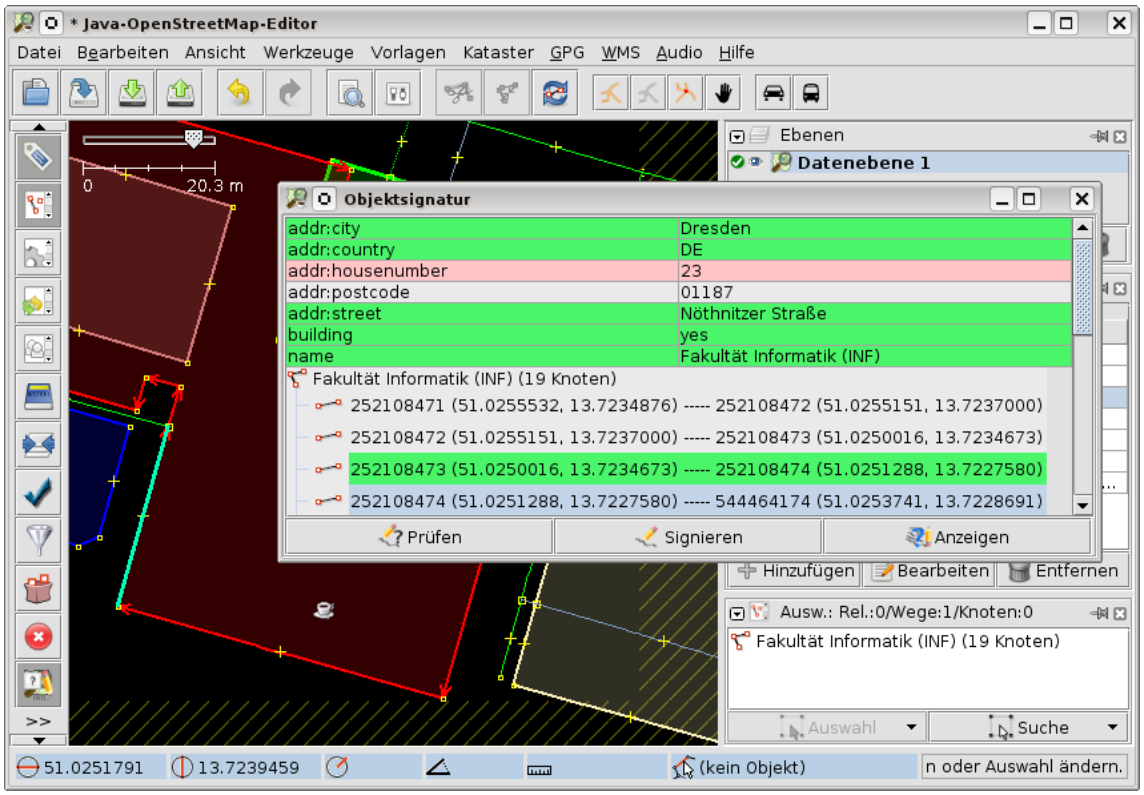

Abbildung 4.2: Dialog zur Auswahl der zu signierenden oder zu analysierenden semantischen Einheiten eines Objektes

Abbildung 4.3: Schlüsselauswahldialog des Pluginprototypen

- Im MapView-Modus des Editors (wenn Daten geladen wurden) erscheint ein neuer Dialog (siehe Abbildung 4.2) um Bewertungen zu selektierten Daten abzugeben oder anzuzeigen.

Der Screenshot in Abbildung 4.2 zeigt den durch das Plugin bereitgestellten Objektsignatur-Dialog, der zu den im Editor selektierten Objekten die semantischen Einheiten darstellt. Farblich wird hervorgehoben, ob die vorliegenden Bewertungen eines Datums dieses bestätigen und gültig sind. Grün bedeutet, dass alle Bewertungen gültig sind und das Objekt bestätigen. Rot bedeutet, dass mindestens eine Bewertung das entsprechende Datum nicht bestätigen kann und grau bedeutet, dass keine Bewertungen vorliegen.

Der Dialog besitzt drei verschiedene Buttons. Der Button „Prüfen“ startet die Überprüfung aller vorliegenden Bewertungen. Das bedeutet, die Signaturen werden geprüft, die Aussage der Bewertung analysiert und ein Reputationswert errechnet. Es wäre denkbar diese Prüfung in Zukunft automatisch durchführen zu lassen und den Button damit einzusparen. Ein Nutzer könnte dann davon ausgehen, dass die Bewertungen alle automatisch vom Plugin geprüft wurden - entweder direkt nach dem Importieren oder direkt vor der Anzeige. Im Prototypen des Plugins muss er die Prüfung jedoch noch selbst starten.

Der Button „Signieren“ startet die Erstellung einer digitalen Signatur der ausgewählten semantischen Einheit, also beispielsweise eines Tags. Zunächst wird der Schlüsselauswahldialog (siehe Abbildung 4.3) angezeigt, durch den der Nutzer grundlegende Aufgaben der Schlüsselverwaltung wahrnehmen kann. Er kann neue Schlüsselpaare anlegen oder Signierschlüssel aus seinem privaten Schlüsselbund auswählen und sich Details zu einem Schlüssel anzeigen lassen. Die Auswahl des Schlüssels kann manuell oder zufällig erfolgen. Möchte ein Bewerter immer den gleichen Schlüssel zum Signieren verwenden und nicht jedes Mal nach dem zu verwendenden Schlüssel gefragt werden, kann er das Erscheinen des Dialogs deaktivieren. Im Anschluss wird der Bewerter noch nach dem Passwort zu dem gewählten Schlüssel gefragt und der Signaturvorgang wird mit dem Berechnen der Signatur abgeschlossen.

Hat ein Bewerter noch kein eigenes Schlüsselpaar oder klickt er auf den Button zur Erzeugung eines neuen Schlüssels (Abbildung 4.3) erscheint ein Dialog zum Erzeugen eines neuen Schlüsselpaares wie in Abbildung 4.4. Der Bewerter kann ein beliebiges Pseudonym als User-ID nutzen, einen Signieralgorithmus wählen (zur Zeit RSA oder DSA), im Falle von RSA eine Bitlänge des Schlüssels auswählen und einen Verschlüsselungsalgorithmus zum Schutz des privaten Schlüssels sowie ein Ablaufdatum des Schlüsselpaares festlegen.

Mithilfe des Buttons „Anzeigen“ aus dem Objektsignatur-Dialog des Plugins ist es möglich detaillierte Informationen zu abgegebenen Bewertungen einer semantischen Einheit abzufragen.

5. JOSM Anleitung, http://wiki.openstreetmap.org/wiki/DE:JOSM/Anleitung

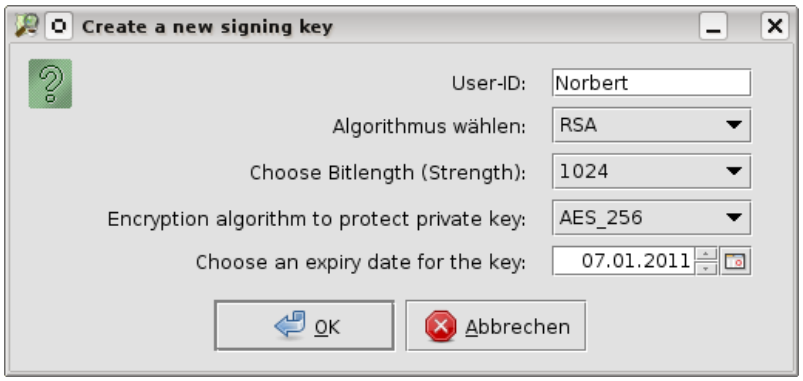

Abbildung 4.4: Dialog zum Erzeugen eines neuen Schlüsselpaares

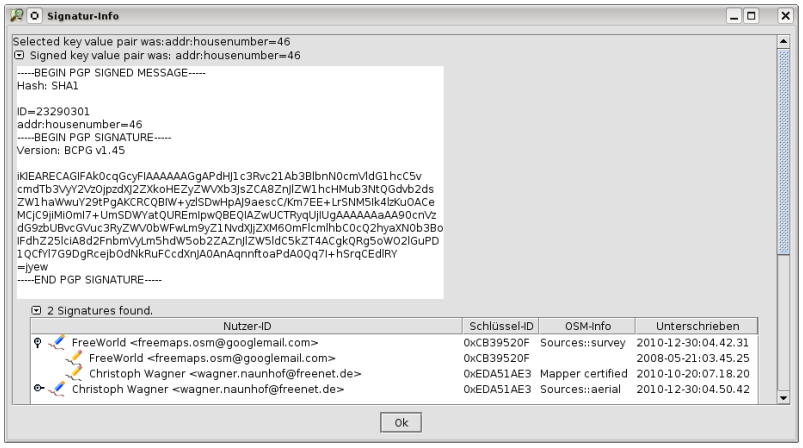

Abbildung 4.5: Signaturinfodialog des Pluginprototypen

Die entsprechende semantische Einheit, z.B. ein Tag, muss dazu selektiert sein. Abbildung 4.5 zeigt beispielhaft zwei Bewertungen für den Tag „addr:housenumber=46“ der Fakultät Informatik. Es sind sowohl die komplette OpenPGP Nachricht als auch Informationen zu den Bewertern verfügbar.

4.1.3 Pluginüberblick - Entwicklersicht

Der Aufbau des Basisverzeichnises des Plugins entspricht der Standardverzeichnisstruktur eines JOSM-Plugins. Die verschiedenen Verzeichnisse sollen in nachstehender Tabelle mit ihrem Inhalt kurz genannt werden:

.	Lizenz, README, ant-build-files
build	kompilierte Klassen und alle Dateien zur Erzeugung des finalen Jar-Pakets
images	im Programm benutzte Grafiken
jce	Java JCE Unlimited Strength Policy Files
lib	externe Bibliotheken
resources	vom Programm verwendete statische Daten
src	Quelltexte in Packagestruktur
svg_images	eigene Bilder im SVG-Format

Die Klassen des Plugins werden in verschiedene Java-Packages aufgeteilt um eine übersichtliche Grundstruktur herzustellen. Haupteigenschaft eines Packages ist der definierte Namensraum. Die Packages orientieren sich dabei an den vorhandenen Packages der JOSM-Kernapplikation.
Eine Liste aller Packages des Plugins, den enthaltenen Klassen, sowie eine kurze Erläuterung ihrer Bedeutung sind im Folgenden aufgeführt:

- **org.openstreetmap.josm.plugins.trustosm**
  Hauptpackage des Plugins.
  - TrustOSMplugin.java : Initialisierung des Plugins
- **org.openstreetmap.josm.plugins.trustosm.actions**
  Enthält Actions siehe org.openstreetmap.josm.actions
  - ExportSigsAction.java : Startet Export der Bewertungen in eine Datei
  - GetMissingDataAction.java : Startet Nachladen von OsmPrimitives, die von importierten Bewertungen referenziert werden, aber noch nicht geladen wurden
- **org.openstreetmap.josm.plugins.trustosm.data**
  Package für Datenklassen
  - TrustNode.java : Speichert Bewertungen zu einem Node
  - TrustWay.java : Speichert Bewertungen zu einem Way
  - TrustRelation.java : Speichert Bewertungen zu einer Relation
  - TrustSignatures.java : Repräsentiert Bewertungen einer semantischen Einheit eines OsmPrimitives
- **org.openstreetmap.josm.plugins.trustosm.gui**
  GUI-Klassen des Plugins
  - DownloadSignedOsmDataTask.java : Steuert das Nachladen von OsmPrimitives und visualisiert Ergebnisse
  - KeyGenerationTask.java : Organisiert Schlüsselerzeugung und gibt graphisches Feedback (Fortschrittsbalken)
  - KeyTreeTableModel.java : Definiert das Aussehen der TreeTable zur Anzeige der verwendeten öffentlichen Schlüssel und deren Zertifikate und Eigenschaften
- **org.openstreetmap.josm.plugins.trustosm.gui.dialogs**
  Klassen, die Dialoge oder Teile davon zur Verfügung stellen
  - JCollabsiblePanel.java : Ein einfaches Panel zum Auf- und Zuklappen

  - TrustDialog.java : Hauptdialog des Plugins (nur sichtbar im MapView Modus von JOSM) zur Steuerung der Bewertung
  - TrustSignaturesDialog.java : Visualisierung von abgegebenen Bewertungen einer semantischen Einheit mit vollständiger OpenPGP-ASCII-Armored-Message und Bewertungsinformationen
  - TrustPreferenceEditor.java : Konfigurationsmenü des Plugins

- **org.openstreetmap.josm.plugins.trustosm.io**
  Klassen zum Importieren und Exportieren von Daten
  - SigExporter.java : Steuert den Export von Bewertungen in eine Datei
  - SigImporter.java : Steuert den Import von Bewertungen aus einer Datei
  - SigReader.java : Liest als XML abgespeicherte Bewertungen ein und erzeugt entsprechende Javaobjekte
  - SigWriter.java : Generiert aus Javaobjekten der Bewertungen XML-Text

- **org.openstreetmap.josm.plugins.trustosm.util**
  Werkzeugklassen, die kompakte, weitestgehend unabhängige Aufgaben erfüllen
  - NameGenerator.java : Erzeugt zufällige Namen
  - SpringUtilities.java : Hilfsklasse zur graphischen Anordnung von Elementen mit Swing
  - TrustGPG.java : Steuert Schlüsselverwaltung, signiert und verifiziert Nachrichten
  - TrustAnalyzer.java : Untersucht Bewertungen auf Gültigkeit und berechnet Reputation

Neben den selbst erstellten Klassen werden zusätzliche Bibliotheken eingebunden. Die Jar-Files der Bibliotheken befinden sich im Unterverzeichnis lib im Plugin-Verzeichnis und haben die im Folgenden beschriebenen Aufgaben:

- bcpg-jdk16-145.jar : BouncyCastle-Klassen zur Nutzung des OpenPGP-Standards
- bcprov-jdk16-145.jar : BouncyCastle-Provider zur Nutzung der Java Cryptography Architecture
- jcalendar-1.3.3.jar : Hilfsklassen, die eine graphische Kalenderfunktion zur Verfügung stellen
- looks-2.0.1.jar : Themes für die Darstellung des Kalenders
- swingx-core-1.6.2.jar : Erweiterte Swingklassen, die komplexe Grafikkomponenten wie z.B. TreeTables bereitstellen

4.2 OPENPGP

Für die Implementierung der digitalen Signatur bei der Bewertungsabgabe wird OpenPGP verwendet. Mit OpenPGP existiert ein frei verwendbarer Internet-Standard [CDF+07] zur Erzeugung einer solchen Signatur. OpenPGP ist weit verbreitet und wird unter anderem zum Verschlüsseln und Signieren von E-Mails benutzt. Grundlage ist ein asymmetrisches Kryptosystem bei dem jeder Teilnehmer ein Schlüsselpaar mit einem privaten und einem öffentlichen Schlüssel erzeugt.
OpenPGP bietet verschiedene Algorithmen an, die zum Verschlüsseln (RSA-E, Elgamal) zum Signieren (DSA, RSA-S) oder für beides (RSA) verwendet werden. Da für diese Diplomarbeit nur die Signaturerzeugung mit OpenPGP wichtig ist, wird auf die Verschlüsselungsmöglichkeiten mit

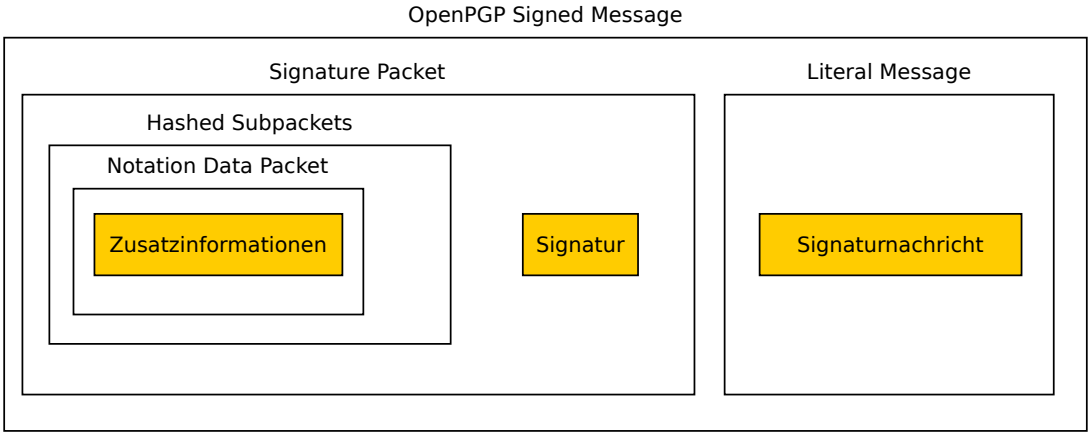

Abbildung 4.6: Prinzipieller Aufbau einer OpenPGP Signed Message.

OpenPGP nicht weiter eingegangen. Mit den Signieralgorithmen DSA (Digital Signature Algorithm) und RSA (bennannt nach den Erfindern Rivest, Shamir und Adleman) werden zwei standardisierte Algorithmen aus dem Digital Signature Standard [NIS09] verwendet. Die Integration von ECC (Elliptic Curve Cryptography) in OpenPGP ist zusätzlich bereits in Planung [Jiv10].

OpenPGP spezifiziert unter anderem die Formate der Schlüssel und Signaturen. Die wesentlichen Informationen werden in Packets untergebracht, die in definierter Weise verschachtelt und aneinandergereiht werden können. In Abbildung 4.6 sind einige für das Bewertungssystem verwendete Packets, sowie ihre Verschachtelung dargestellt. Eine OpenPGP Signed Message entspricht dabei einer oder mehrerer Bewertungen, abhängig von der Anzahl der vorhandenen Signature Packets. In einer OpenPGP Signed Message können also mehrere Signature Packets vorkommen, wenn zu einer Klartextnachricht mehrere Signaturen vorliegen. Die Klartextnachricht, in der Arbeit auch Signaturnachricht genannt, ist in einer Literal Message eingebettet.
Ein Signature Packet besitzt neben Eigenschaften wie der Schlüssel-ID, mit der die Signatur erzeugt wurde, dem Erzeugungszeitpunkt, den verwendeten Algorithmen und Formatversionen noch Platz für beliebig viele sogenannte Hashed Subpackets, sowie die eigentliche Signatur selbst. Neben den Hashed Subpackets könnten auch Unhashed Subpackets angegeben werden, die jedoch für diese Arbeit nicht verwendet werden, weil sie nicht durch die Signatur geschützt sind.
Bei der Erstellung der Signatur wird zunächst ein Hashwert aus der Klartextnachricht sowie aus den bereits genannten Eigenschaften des Signature Packets einschließlich der Hashed Subpackets berechnet, einzig die Unhashed Subpackets bleiben davon ausgeschlossen. Über diesen Hash wird anschließend die digitale Signatur angefertigt und im Signature Packet gespeichert.
In den Hashed Subpackets können verschiedene und beliebig viele Subpackets aneinandergereiht werden, die Informationen über den Schlüssel bzw. den zugehörigen Signierer liefern wie z.B. die Signer's User ID, also ein String, der üblicherweise einen Namen und eine E-Mailadresse enthält. Es können auch Informationen über bevorzugte Keyserver, Kompressionsalgorithmen und vieles mehr angegeben werden. Besondere Bedeutung soll hier das sogenannte Notation Data Packet erhalten, mit dem es möglich ist frei wählbare zusätzliche Bemerkungen einem Signature Packet hinzuzufügen. Dieses wird benutzt, um die schon öfter genannten Zusatzinformationen zu speichern.

4.2.1 Notation Data

Die Zusatzinformationen wie beispielsweise die Toleranz sollen in einem Notation Data Packet untergebracht werden. Alternativ könnten sie auch einfach der schon vorhandenen Signaturnachricht hinzugefügt werden. Dies hätte allerdings den Nachteil, dass die Bewertungen, die sich auf

die gleichen Openstreetmap-Daten beziehen bei unterschiedlichen Zusatzinformationen unterschiedliche Signaturnachrichten bilden würden und somit nicht in der selben OpenPGP Signature Message unterkommen können. Es wird dadurch schwieriger festzustellen, ob zwei Bewertungen zu den selben Daten abgegeben wurden. Zudem sind die Zusatzinformationen als Ergänzung zu verstehen, die die Interpretation der Signatur beeinflussen. Da sie optional sind, kann eine eventuell eingeschränktere Auswertung der Signatur zu den bewerteten Daten auch ohne diese Informationen stattfinden.

Der Aufbau und die Verwendung des Notation Data Packets wird im Folgenden beschrieben. Ein Notation Data packet besteht aus Schlüssel-Wert-Paaren, die jeweils aus Strings bestehen. Der OpenPGP-Standard beschreibt für den Schlüsselraum zwei mögliche nutzbare Namensräume. Einer wird von der Internet Engineering Task Force (IETF) verwaltet und umfasst alle Strings, die kein @ Symbol enthalten. Daneben gibt es die Möglichkeit einen nutzerdefinierten Namensraum zu benutzen, indem ein beliebiger UTF-8 String gefolgt von einem @ und einem DNS domain name angegeben wird.
Bei der Implementierung des Prototypen wurde letztere Variante gewählt. Der gewählte Key des Notation Data subpackets, das ein bestimmtes Openstreetmap-Zertifikat anzeigen soll lautet:
„trustosm@openstreetmap.org“
Vor der allgemeinen Inbetriebnahme des Systems muss dies zunächst mit den Eigentümern der verwendeten Domain abgesprochen werden.

Ein Notation Data packet kann sowohl bei einer Zertifizierung eines öffentlichen Schlüssels angegeben werden, sowie bei der Signatur von Openstreetmap-Daten. Der Value des packets kann dabei natürlich verschieden sein und unterschiedliche Bedeutung haben.

Bewerterzertifikat

Wird ein Notation Data packet bei der Beglaubigung eines öffentlichen Schlüssels eingesetzt, so ist es möglich nicht nur die Korrektheit und Zuordnung eines Schlüssels zu einem Nutzer zu bestätigen sondern auch bestimmte Eigenschaften des Schlüsseleigentümers zu zertifizieren und an den öffentlichen Schlüssel zu binden. Allein die Existenz einer Openstreetmap-Notation lässt darauf schließen, dass es sich um eine für Openstreetmap relevante Zertifizierung handelt. Wie wertvoll diese ist, wird durch die Reputation des Zertifizieres festgelegt. Über einen Kommentar kann der Zertifizierer zusätzlich Angaben machen, welche Eigenschaften er genau zertifiziert, wenn dies nicht schon implizit durch den Zertifizierer deutlich ist.

Der Aufruf des Kommandozeilenprogrammes GnuPG kann ein entsprechendes Notation Data packet wie folgt einer Schlüsselbeglaubigung hinzufügen:

```
gpg --cert-notation "trustosm@openstreetmap.org"="Note:OSM Experte" \
  --sign-key <ID vom Dresdener OSM Stammtischschlüssel>
```

Im Beispiel signiert der Dresdner OSM Stammtisch, dass ein bestimmter Nutzer ein OSM Experte ist.

OpenPGP kann subpackets als kritisch markieren, wenn sie für die Aussage der Signatur von entscheidender Bedeutung sind. Implementationen, welche diese subpackets dann nicht auswerten können, werden somit gewarnt die Signatur zu benutzen. Das spezielle Notation Data subpacket ist jedoch in diesem Fall nicht kritisch für die Signatur. Die Bedeutung der Signatur reduziert sich bei ignorieren des subpackets lediglich auf die Überprüfung des Schlüssels, die natürlich von jeder Zertifizierungsstelle, die Openstreetmap-Zertifikate ausstellt, zuvor vorgenommen werden muss. Ein Openstreetmap-Zertifikat enthält also neben der Aussage des Zertifikates zusätzlich eine Aussage, wie gut der zu zertifizierende öffentliche Schlüssel überprüft wurde.

Bewertungszusatzinformationen

Ein Notation Data packet kann auch Zusatzinformationen in einer Bewertung unterbringen. Dabei werden nicht die signierten Daten verändert, sondern lediglich die Interpretation einer gültigen Signatur näher spezifiziert. Ein Beispiel für eine solche Zusatzinformation ist die Toleranz einer Nodeposition. Das Notation Data packet enthält beispielsweise folgende Information:

```
"trustosm@openstreetmap.org"="Tolerance:10m"
```

Die Signatur legt also die signierten Daten in diesem Fall nicht exakt fest, sondern lässt gewisse Abweichungen zu.

Auch hier ist das subpacket nicht kritisch, sofern die Gültigkeit der Signatur erweitert und nicht eingeschränkt wird. Ein Ignorieren des subpackets würde in diesem Fall eventuell mehr ungültige Bewertungen verursachen als nötig. Schwerer wiegt jedoch, wenn eine Bewertung als gültig erkannt wird, obwohl sie es durch einschränkende subpackets nicht wäre.

Auswertbarkeit der Zusatzinformationen

Prinzipiell kann der Value-String des Notation Data Packets beliebig sein. Möchte man jedoch eine automatische Auswertung ermöglichen ist es notwendig sich auf bestimmte Werte festzulegen. Nicht alle Value-Strings müssen jedoch automatisch ausgewertet werden. Kommentare eines Bewerterzertifikates können auch auf semantischer Ebene vom Endnutzer interpretiert werden. Werte wie ein Toleranzbereich hingegen, lassen sich hervorragend automatisch nutzen und weiterverarbeiten. Bisher sind im Prototypen folgende Werte verwendet:

- **Note:** Hierbei kann angezeigt werden, dass es sich um einen Freitextkommentar handelt, der nicht automatisch ausgewertet werden braucht.
- **Tolerance:** Gibt eine Toleranz für geografische Positionen an, für innerhalb derer die Bewertung für verschobene Positionen ebenfalls gelten soll. Aktuell zulässige Werte sind Zahlenwerte gefolgt von einer Längeneinheit, also z.B. **Tolerance:10m**
- **Sources:** Gibt einen Hinweis darauf aus welchen Quellen sich ein Bewerter informiert hat um die Richtigkeit des bewerteten Datums zu bestätigen. Es können dabei mehrere Quellen angegeben werden. Aktuell möglich sind :survey, :aerial, :web und :trusted. Survey bedeutet, dass der Bewerter sich die Lage vor Ort angeschaut hat. Bei Aerial hat der die Informationen aus einem Luftbild entnommen. Web bedeutet er hat lediglich eine Internetrecherche angestellt und dabei Informationen gefunden, die das Openstreetmap-Datum bestätigen. Trusted steht für dritte aus Sicht des Bewerters vertrauenswürdige Personen, die dem Bewerter die Informationen gegeben haben z.B. Freunde, Verwandte und Ortsansässige. Hat ein Bewerter im Internet recherchiert und war gleichzeitig vor Ort, so sieht der Value-String wie folgt aus: **Sources::web:survey**

Falls zu dem vereinbarten Notation Key **trustosm@openstreetmap.org** andere Values als die hier genannten auftreten, können diese im Programm angezeigt werden. Notation Data Packets mit anderem Key werden ignoriert.

4.2.2 BouncyCastle

Die Nutzung von OpenPGP in Java wurde mit dem Framework BouncyCastle[6] realisiert, welches unter einer adaptierten „MIT X11 Lizenz“[7] steht. Das Framework setzt auf die JCA (Java Cryptography Architecture)[8] auf und liefert einen eigenen Cryptographic Service Provider dafür mit, welcher mit folgender Codezeile registriert werden kann:

```
Security.addProvider(new BouncyCastleProvider());
```

Mit der JCA bzw. dem Kryptographieframeworkteil JCE (Java Cryptography Extension) ist eine Vielzahl von kryptographischen Algorithmen nutzbar, die auch für OpenPGP verwendet werden können. Die JCE bietet Funktionen für kryptographische Tätigkeiten, wie Schlüsselpaarerzeugung, Hashberechnung, Signaturerstellung, Verschlüsselung z.B. von Passwörtern und vieles mehr. Mit dem BouncyCastle Framework werden diese Grundfunktionen angewendet, um OpenPGP-Schlüssel zu erzeugen und diese z.B. zum Erstellen und Testen von OpenPGP-Signaturen zu benutzen.

Jurisdiction Policy Files

Die volle kryptographische Stärke der JCE ist in der Standard JVM (Java Virtual Machine) leider beschränkt. In manchen Ländern gibt es gesetzliche Einschränkungen für kryptographische Algorithmen, z.B. Höchstgrenzen für Schlüssellängen. Die JCE legt diese Einschränkungen in Jurisdiction Policy Files[9] fest, also Dateien, die von der JVM zur Laufzeit ausgewertet werden und die Nutzung des Kryptografieframeworks durch Programme beschränken. Die Regeln in der Standardausführung dieser Files, den so genannten „Strong Policy Files“ beschränken die meisten Algorithmen auf 128 Bit Schlüssellänge. Die Schlüssellänge für RSA ist durch diese mitgelieferten Files ausnahmsweise jedoch unbegrenzt.
Dennoch ist die OpenPGP-Schlüsselgenerierung des TrustOSMPlugins von den Beschränkungen betroffen. Neben den möglichen Beschränkungen von DSA und in Zukunft auch ECDSA wird von OpenPGP ein symmetrischer Verschlüsselungsalgorithmus benutzt, um den geheimen Schlüssel durch ein Passwort zu schützen. Meistens kommt hier AES (Advanced Encryption Standard) zum Einsatz, für den Schlüssellängen von 128, 192 und 256 Bit wählbar sind und der durch die „Strong Policy Files“ lediglich in seiner 128 Bit Schlüsselvariante nutzbar wäre.
Lebt man in einem Land, welches keine gesetzlichen Beschränkungen von Kryptografie aufweist, so kann man die sogenannten „Unlimited Strength Policy Files“ von der Java-Downloadseite[10] beziehen und sich in seine JVM installieren. Die kryptographischen Algorithmen sind anschließend ungehindert und mit voller zur Verfügung stehender Schlüssellänge nutzbar.
Das Plugin TrustOSM prüft bei seiner Initialisierung, ob Kryptographie unbeschränkt nutzbar ist, indem versucht wird einen 192 Bit starken AES Schlüssel anzulegen. Scheitert dies, liegen Beschränkungen vor und eine Warnung wird ausgegeben, dass nicht alle kryptographischen Möglichkeiten zur Verfügung stehen.
Eventuell könnten die Unlimited Strength Policy Files mit dem Plugin mitgeliefert und eine Installation auf Wunsch des Nutzers einfach veranlasst und automatisch durchgeführt werden. Die aktuelle Implementierung unterstützt dies jedoch noch nicht.

OpenPGP Nutzung

Die wesentlichen Funktionen, die das BouncyCastle-Framework für die Nutzung des OpenPGP-Formates anbietet, sind im Java package org.bouncycastle.openpgp[11] aufgeführt. Die Klassen des

6. BouncyCastle, http://www.bouncycastle.org/java.html
7. BouncyCastle Lizenz, http://www.bouncycastle.org/licence.html
8. JCA, http://download.oracle.com/javase/6/docs/technotes/guides/security/crypto/CryptoSpec.html
9. Policy Files, http://download.oracle.com/javase/6/docs/technotes/guides/security/crypto/CryptoSpec.html#AppB
10. Java Download, http://www.oracle.com/technetwork/java/javase/downloads/index.html
11. BouncyCastle API, http://www.bouncycastle.org/docs/pgdocs1.6/index.html

Frameworks werden hauptsächlich in der Utilityklasse TrustGPG des TrustOSMPlugins verwendet. Die Klasse TrustGPG übernimmt dabei viele Aufgaben die im Folgenden aufgelistet sind. Zu jeder Aufgabe sind die Methoden von TrustGPG aufgeführt und fett gedruckt, die diese Aufgabe erfüllen. Ihr Rückgabewert ist vorangestellt. Void bedeutet es gibt keinen Rückgabewert.

- Verwaltung der öffentlichen und privaten OpenPGP-Schlüssel
  - PGPPublicKey **getPublicKeyFromRing**(long keyID)
  - String **secKeytoString**(PGPSecretKey k)
  - void **readGpgFiles**()
  - void **writeGpgFiles**()
- Auswahl und Entsperren eines geheimen Schlüssels
  - void **readSecretKey**()
  - void **getPasswordfromUser**()
- Erzeugen eines Schlüsselpaares
  - PGPSecretKey **generateKey**()
- Signieren von Textnachrichten
  - TrustWay **signWay**(TrustWay trust)
  - PGPSignature **signSegment**(TrustWay trust, List<Node> nodes)
  - PGPSignature **signNode**(Node node)
  - boolean **signTag**(TrustOsmPrimitive trust, String key)
  - PGPSignature **sign**(String tosign)
- Verifizieren von Signaturen zu Textnachrichten
  - boolean **verify**(String sigtext, PGPSignature sig)
- Erstellung von Signaturpaketen mit Zusatzinformationen
  - PGPSignatureSubpacketGenerator **chooseAccuracy**()
  - PGPSignatureSubpacketGenerator **chooseInformationSource**()
- Auslesen von Signaturpaketen
  - double **searchTolerance**(PGPSignature sig)
- Anzeigen von Informationen zu einem öffentlichen Schlüssel
  - void **showKeyDetails**(PGPPublicKey key)

Wichtigste BouncyCastle-Klasse ist für das Trustosm-Plugin die Klasse PGPSignature, die ein komplettes Signaturpaket enthält. Eine PGPSignature wird von der Methode sign(String tosign) erzeugt, der ein zu signierender String, die Signaturnachricht übergeben wird. Die anderen sign-Methoden rufen diese Methode auf, nachdem sie die übergebenen OSM-Daten verarbeitet und die semantischen Einheiten in Signaturnachrichten geformt haben.

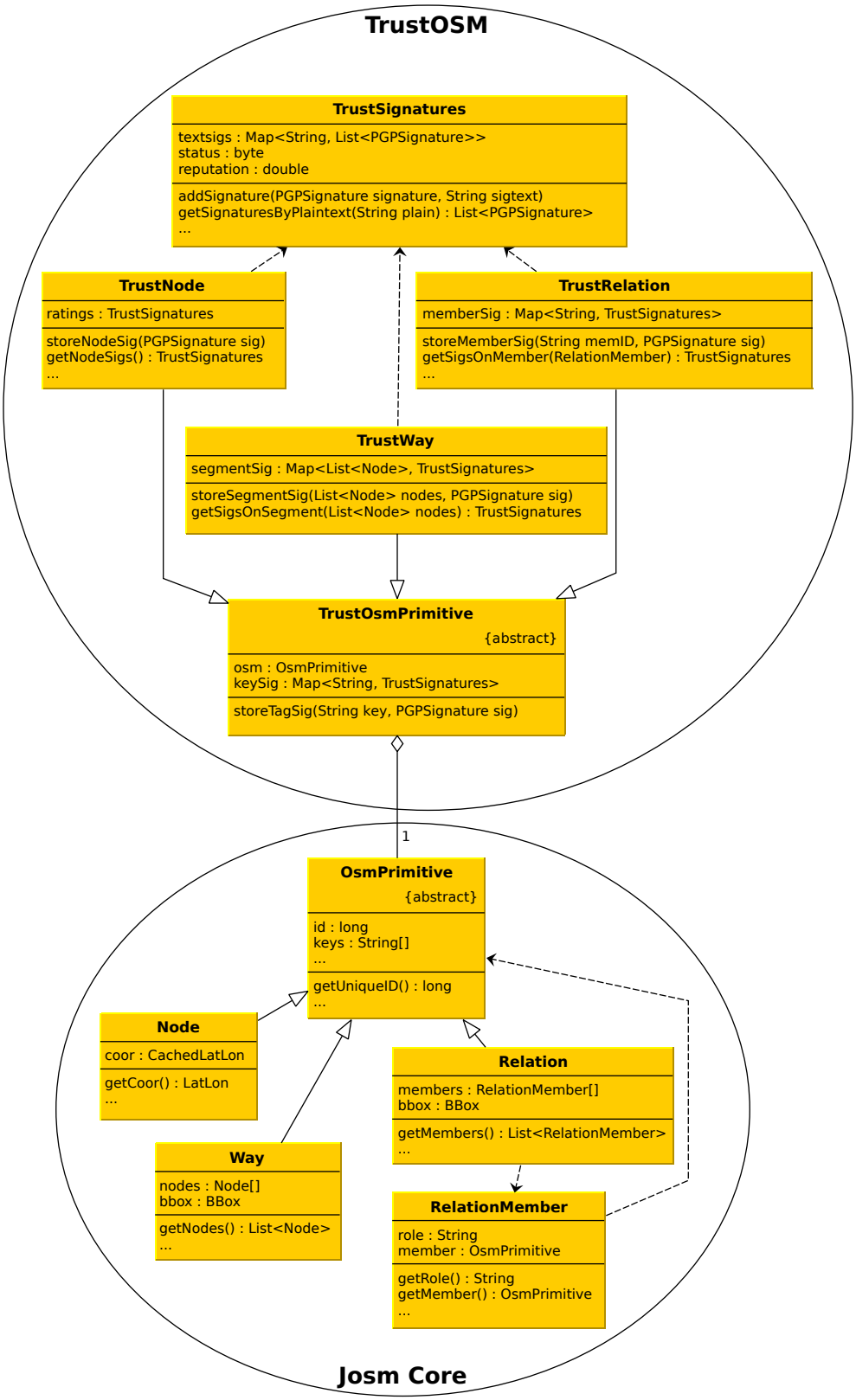

Abbildung 4.7: Wichtigste Klassen zur Verwaltung der OSM-Objekt- und Signaturdaten.

4.3 DATENSTRUKTUREN

Für die Implementierung der Signaturerweiterung wurden zu den bereits existierenden Openstreetmap-Objektklassen in JOSM neue Klassen hinzugefügt, die auf diese verweisen. Abbildung 4.7 stellt den prinzipiellen Zusammenhang der vorhandenen Klassen mit den wichtigsten neuen Datenklassen in einem Klassendiagramm dar.
Die abstrakte Klasse OsmPrimitive stellt dabei ein OSM Objekt dar, dessen konkrete Instanzen durch die Klassen Node, Way oder Relation erzeugt werden. In der abstrakten Klasse OsmPrimitive sind alle Attribute und Methoden definiert, die alle Openstreetmap-Objekte gemeinsam haben. Als Beispiel sei die eindeutige id genannt, die zusammen mit dem Objekttyp ein Objekt eindeutig definiert. Weiterhin hat jedes OSM Objekt einen Liste von Tags, gespeichert in dem String-Array keys. Die Besonderheiten jeder OSM-Objektart werden in der jeweiligen konkreten Klasse definiert. Ein Node besitzt demnach genau eine geographische Koordinate vom Typ LatLon, ein Way speichert eine Liste von Nodes und eine Relation eine Liste von Mitgliedern (RelationMember).

4.3.1 TrustOsmPrimitive

Zu jedem konkreten Openstreetmap-Objekt soll eine Bewertung gespeichert werden. Dazu werden neue Klassen geschaffen, die den verschiedenen Bewertungen bei verschiedenen OSM-Objekttypen gerecht werden. So müssen zu jedem OSM-Objekt Tagsignaturen gespeichert werden. Bei Nodeobjekten werden zusätzlich einzelne Nodesignaturen gespeichert, bei Wayobjekten Segmentsignaturen und bei Relations die Membersignaturen. Die konkreten Klassen TrustNode, TrustWay und TrustRelation bewerkstelligen genau das. Sie erben wiederum von der abstrakten Klasse TrustOsmPrimitive, die das Speichern der Tagsignaturen implementiert.

Dabei sind TrustNode, TrustWay und TrustRelation keine erweiterten Klassen von Node, Way und Relation, sondern halten lediglich eine Referenz auf das entsprechende OsmPrimitive-Objekt. Somit ist es möglich von jedem signierten Inhalt aus, vermerkt in den Klassen TrustNode, TrustWay und TrustRelation auf das zugehörige OSM-Objekt zuzugreifen. Wichtig ist auch die Referenz andersherum. Es ist also wichtig für ein gegebenes OsmPrimitive-Objekt zu erfahren, ob Bewertungen vorliegen, also ein entsprechendes TrustOsmPrimitive Objekt existiert. Hierfür wird eine globale Liste in der Hauptklasse des Plugins TrustOSMplugin geführt, welche über einen String, der die Openstreetmap-ID zusammen mit dem Typ eines Objektes repräsentiert, ein eventuell vorhandes TrustOsmPrimitive zugreifbar macht. Es müssen dabei lediglich die Bewertungen in der Liste des Plugins vorliegen, zu denen auch aktuell Daten im Editor vorhanden sind. Üblicherweise wird nur ein kleiner Ausschnitt der Openstreetmap-Daten in den Editor geladen und bearbeitet. Nur die bewerteten Daten dieses Ausschnittes tauchen in der Liste des Plugins auf.

Die Liste wird wie folgt instanziiert:

```
public static final Map<String, TrustOsmPrimitive> signedItems =
            new HashMap<String, TrustOsmPrimitive>();
```

Die Zuordnung von Openstreetmap-ID, die als String vorliegt und einem TrustOsmPrimitive wird als Map implementiert. Zu einem Key in einer Map, in diesem Fall der String der Openstreetmap-ID kann jeweils genau ein Value vorliegen, also ein TrustOsmPrimitive. HashMap ist eine konkrete Implementierung des Java Map Interfaces mithilfe einer Hash-Tabelle, die eine konstante Zugriffszeit für die verschiedenen Einträge bietet.
Um nun also zu erfahren, ob zu einem konkreten Openstreetmap-Objekt Bewertungen vorliegen, muss so vorgegangen werden, wie im folgenden Codebeispiel gezeigt:

```
public boolean existsTrustOsmPrimitive(OsmPrimitive osm) {
   String oid = TrustOsmPrimitive.createUniqueObjectIdentifier(osm);
   return TrustOSMplugin.signedItems.containsKey(oid);
}
```

Es wird also lediglich geprüft, ob in dieser globalen Liste zu einer Openstreetmap-ID ein entsprechendes TrustOsmPrimitive existiert, durch welches auf Bewertungen für Daten des gegebenen OsmPrimitives zugegriffen werden kann.
Die globale Liste der signedItems wird immer dann erweitert, wenn Bewertungen in die Anwendung importiert oder Bewertungen in der Anwendung zu gewählten OSM Daten abgegeben werden.

UniqueObjectIdentifier

An dieser Stelle wird der Schlüsselstring der Liste mit den Bewertungen genauer betrachtet. Wie im Codebeispiel gezeigt, entsteht dieser String durch Aufruf der statischen Methode createUniqueObjectIdentifier(OsmPrimitive osm) von der Klasse TrustOsmPrimitive, welche in folgendem Codeauszug beschrieben ist:

```
public static String createUniqueObjectIdentifier(OsmPrimitive osm) {
   String id = "";
   if(osm instanceof Node) {
     id = "n";
   } else if(osm instanceof Way) {
     id = "w";
   } else if(osm instanceof Relation) {
     id = "r";
   }
   id += osm.getUniqueId();
   return id;
}
```

Um ein Openstreetmap-Objekt eindeutig zu referenzieren, muss der Objekttyp (Node, Way oder Relation) sowie die Identifikationsnummer bekannt sein. Die Identifikationsnummer allein ist nur eindeutig für Objekte gleichen Typs. Die beschriebene Methode kodiert den Objekttyp in einen Buchstaben, der der Identifikationsnummer voran gestellt wird. Der resultierende String enthält alle Informationen, um OSM-Objekte eindeutig zu referenzieren. Mit ihm kann ein OsmPrimitive rekonstruiert bzw. gefunden oder vom OSM Server nachgeladen werden. Er wird zudem zur Erzeugung der Signaturnachrichten verwendet, und entspricht der im Entwurf verwendeten OID.

Trust-Klassen im Detail

Auf die Umsetzung der in Abbildung 4.7 erwähnten Klassen des TrustOSMPlugins wird in den folgenden Absätzen näher eingegangen. Prinzipiell sind die Klassen TrustNode, TrustWay und TrustRelation die Verbindung zwischen den eigentlichen OSM-Objekten, implementiert in den Klassen Node, Way und Relation, und den abgegebenen Bewertungen, implementiert in der Klasse TrustSignatures.

Die Klasse TrustOsmPrimitive implementiert die Speicherung der Bewertung für Tags in einer Map wie folgt:

```
private final Map<String, TrustSignatures> keySig =
      new HashMap<String, TrustSignatures>();
```

Der Schlüsselstring der Map ist in diesem Fall der Name des keys eines bewerteten Tags (key-value-Paar). Die Klasse implementiert Methoden, um Bewertungen zu keys hinzuzufügen, bzw. gespeicherte Bewertungen abzufragen. Weiterhin enthält sie statische Methoden, um aus einem gegebenen OsmPrimitive und einem key eine Signaturnachricht für den somit referenzierten Tag zu erzeugen, bzw. um aus einer gegebenen Signaturnachricht das key-value-Paar wiederherzustellen.

Die Klasse TrustNode erbt von der Klasse TrustOsmPrimitive und kann demzufolge ebenso Bewertungen für Tags abspeichern. Zusätzlich ist ein Bewerter in der Lage die Position eines Nodes zu bewerten. Da ein Node genau eine Position hat ist genau ein TrustSignatures Objekt erforderlich die Bewertungen dafür abzuspeichern. Das Attribut ratings steht dafür zur Verfügung. In der Klasse TrustNode sind außerdem statische Methoden implementiert, die zu einem gegebenen Nodeobjekt eine Signaturnachricht der Position erzeugen bzw. aus einer solchen Signaturnachricht ein Nodeobjekt rekonstruieren.
Bei der Erzeugung eines TrustNodes muss ein bereits existierender Node angegeben werden, der von dem TrustNode referenziert werden kann.

Neben dem TrustNode erbt die Klasse TrustWay ebenso von TrustOsmPrimitive. Neben den Tags sind bei einem Way Bewertungen einzelner Segmente möglich. Ein Segment wird durch eine Liste von Nodes repräsentiert. In dieser Implementierung wird davon ausgegangen, dass diese Liste zweielementig ist. Zu jedem dieser Segmente speichert der TrustWay Bewertungen in einer Map ab:

```
private final Map<List<Node>, TrustSignatures> segmentSig =
      new HashMap<List<Node>, TrustSignatures>();
```

Ein TrustWay bietet Methoden Signaturen zu gegebenen Segmenten abzuspeichern bzw. abzufragen. Mit statischen Methoden können Segmente in Signaturnachrichten geformt, bzw. aus Signaturnachrichten Segmente erzeugt werden.

In ähnlicher Weise wie TrustNode und TrustWay erbt die Klasse TrustRelation von TrustOsmPrimitive und stellt grundsätzliche Methoden zur Verwaltung von Relationsmitgliedern zur Verfügung. Gespeichert werden diese in einer Map:

```
private final Map<String, TrustSignatures> memberSig =
      new HashMap<String, TrustSignatures>();
```

Der Stringschlüssel in dieser Map ist die OID des jeweils bewerteten RelationMembers. RelationMembers können mithilfe von statischen Methoden in Signaturnachrichten bzw. umgekehrt gewandelt werden.

4.3.2 TrustSignatures

Die Speicherung der eigentlichen Bewertungen für die verschiedenen bewertbaren Teile der OSM-Objekte übernimmt die Klasse TrustSignatures. Diese wird von allen TrustOsmPrimitives benutzt. Bewertungen haben zwei Teile. Zum einen die Signaturnachricht, die bewertet wurde und zum anderen die digitale Signatur. Zu einer Signaturnachricht können mehrere verschiedene Signaturen vorliegen, aber jede Signatur bezieht sich auf genau eine Signaturnachricht. Die verwendete Datenstruktur, die diesen Zusammenhang abbildet ist die folgende:

Status	Interpretation
SIG_UNKNOWN	Status der Bewertungen unbekannt oder ungeprüft
SIG_VALID	Alle Bewertungen sind gültig und bestätigen die aktuellen Daten
SIG_BROKEN	Mindestens eine Bewertung ist für die aktuellen Daten ungültig
ITEM_REMOVED	Bewertungen beziehen sich auf Daten, die nicht mehr vorhanden sind

Tabelle 4.1: Statusnachrichten eines TrustSignatures-Objektes und deren Bedeutung.

```
private final Map<String, List<PGPSignature>> textsigs =
      new HashMap<String, List<PGPSignature>>();
```

Jeder Schlüsselstring ist eine Signaturnachricht zu der eine Liste von Signaturen gespeichert wird. Die verschiedenen Signaturnachrichten repräsentieren die verschiedenen Werte einer semantischen Einheit, die bewertet wurden. Beispiel 4.1 zeigt, wie es dazu kommen kann.

Beispiel 4.1 (Bewertung eines Straßenattributes) *Eine Straße ist mit dem Key-Value-Paar „surface=unpaved" getaggt, was eine nicht versiegelte unbefestigte Straßenoberflächenbeschaffenheit anzeigt. Wird dieses Tag signiert, wird bei einem TrustWay-Objekt in der Map „keySig" unter dem Schlüssel „surface" ein TrustSignatures-Objekt angelegt und gespeichert. Dieses TrustSignatures-Objekt enthält nun eine Signaturnachricht bestehend aus der ID des Ways, sowie dem kompletten Tag „surface=unpaved".*
Ein Mapper konkretisiert den Tag und ändert ihn zu „surface=sand", um anzuzeigen, dass es sich um einen Sandweg handelt. Wird der neue Wert ebenso signiert, wird im bereits vorliegenden TrustSignatures-Objekt, welches vom bestehenden TrustWay-Objekt für den Key „surface" gespeichert wird, eine neue Signaturnachricht in die Map „textsigs" aufgenommen, die nun wiederum die Signaturen für diesen Wert des Tags in einer Liste speichert.

Ein ähnliches Vorgehen bei der Speicherung von Bewertungen liegt bei Nodes vor. Die verschiedenen Signaturnachrichten zu einem Node entstehen, wenn dieser verschoben und anschließend erneut signiert wird. Genauso können Segmente verschoben werden. Bei Relationsmitgliedern könnten sich möglicherweise die Rollen ändern, die diesen in der Relation zugewiesen sind.

Reputation

Der Reputationswert einer semantischen Einheit wird neben den Bewertungen ebenfalls in dem zugehörigen TrustSignatures-Objekt gespeichert. Da die implementierte Reputationsfunktion noch nicht sehr aussagekräftig ist, wird im Moment zum farblichen Hervorheben der unterschiedlich bewerteten semantischen Einheiten im Objektsignatur-Dialog neben dem Reputationswert eine sogenannte Statusnachricht geführt, die lediglich vier verschiedene Zustände haben kann, welche in Tabelle 4.1 einschließlich ihrer Interpretation aufgeführt sind.
Zukünftige Entwicklungen des Programmes könnten diese Statusnachricht überflüssig machen und eine farbliche Anzeige der semantischen Einheiten direkt vom Reputationswert abhängig machen. In den Reputationswert können mehr Informationen einfließen, als die bloße Gültigkeit einer Bewertung für eine semantische Einheit. Wenn alle Bewertungen gültig sind, kann der Reputationswert weitere Abstufungen vornehmen, wie vertrauenswürdig eine semantische Einheit unter Berücksichtigung der Bewertungen ist. Ob Bewertungen einfließen, die die Gültigkeit der aktuellen semantischen Einheit nicht untermauern, ist dem Reputationsalgorithmus überlassen.

4.4 BEWERTUNGSAUSWERTUNG

Für eine Auswertung der vorhandenen Bewertungen stellt die Klasse TrustAnalyzer Methoden zur Verfügung. Dabei wird genauso vorgegangen, wie in Abschnitt 3.7 dargelegt. Zunächst wird geprüft, ob die Signaturen zu den Signaturnachrichten passen. Diese Prüfung erfolgt in der globalen Utilityklasse TrustGPG, die grundlegende OpenPGP-Funktionalitäten bereitstellt und durch die globale Variable TrustOSMplugin.gpg zugreifbar ist.

```
TrustOSMplugin.gpg.verify(signedPlaintext, sig)
```

Übergeben wird die PGPSignature sig sowie der String signedPlaintext, der die Signaturnachricht darstellt. Ist die Prüfung erfolgreich folgt der zweite Schritt, der Vergleich der Daten der Signaturnachricht mit den aktuell in Frage stehenden Daten. Dieser Vergleich soll als Implementierung beispielhaft im Folgenden für die Bewertung der Position eines Nodes aufgeführt werden:

```
public static boolean isNodeRatingValid(TrustNode trust,
                                        String signedPlaintext,
                                        PGPSignature sig) {
  /** Rating is valid if Node from signed plaintext
  ** is inside Tolerance given in Signature
  */
  Node signedNode = TrustNode.generateNodeFromSigtext(signedPlaintext);
  Node currentNode = (Node)trust.getOsmPrimitive();
  double dist = signedNode.getCoor().greatCircleDistance(currentNode.getCoor());
  /** is distance between signed Node and current Node inside tolerance? */
  return dist<=TrustGPG.searchTolerance(sig);
}
```

Zunächst wird der signierte Node aus der übergebenen Signaturnachricht rekonstruiert. Der aktuelle Node wird anhand der gespeicherten Referenz im übergebenen TrustNode in Erfahrung gebracht. Zwischen der signierten und der aktuellen Position wird die Entfernung auf der Erdoberfläche berechnet. In der PGPSignatur sig wird nach Werten für eine Toleranz gesucht und anschließend verglichen, ob die aktuelle Position des Nodes innerhalb der erlaubten Abweichung liegt. Ist keine Toleranz angegeben, so liefert TrustGPG.searchTolerance(sig) den Wert 0 zurück und die Bewertung wäre somit nur dann gültig, wenn sich die Position nicht geändert hat.

Für alle nach diesen zwei Prüfungen gültigen Bewertungen eines TrustSignatures-Objektes wird im Anschluss ein Reputationswert gebildet. In der aktuellen Implementation ist dies lediglich die Anzahl der abgegebenen gültigen Bewertungen.

4.5 IMPORT/EXPORT

Die abgegebenen Bewertungen sollen außerhalb des Plugins gespeichert werden können. Damit auch andere zukünftige Anwendungen die Daten problemlos einlesen können, wurde das Austauschformat mit XML (Extensible Markup Language) [BPM+08] realisiert.
Für diese Diplomarbeit wurde noch kein geeigneter zentraler Datenbankserver zur Bewertungsspeicherung, wie in Abschnitt 3.4 beschrieben, implementiert. Daher bietet das Plugin zu diesem Zeitpunkt lediglich eine Speicherung in eine lokale Datei an. XML bietet hierbei den Vorteil, dass die exportierten Dateien mit einem Texteditor betrachtet werden können und dabei menschenlesbar sind. Zudem lässt sich der Aufbau einer XML Datei recht einfach mit Hilfe einer DTD (Dokumenttypdefinition) formal beschreiben. In dieser Diplomarbeit wurde eine solche DTD entwickelt, die die Datenstruktur des Programmes in einer trustXML Datei nachbildet:

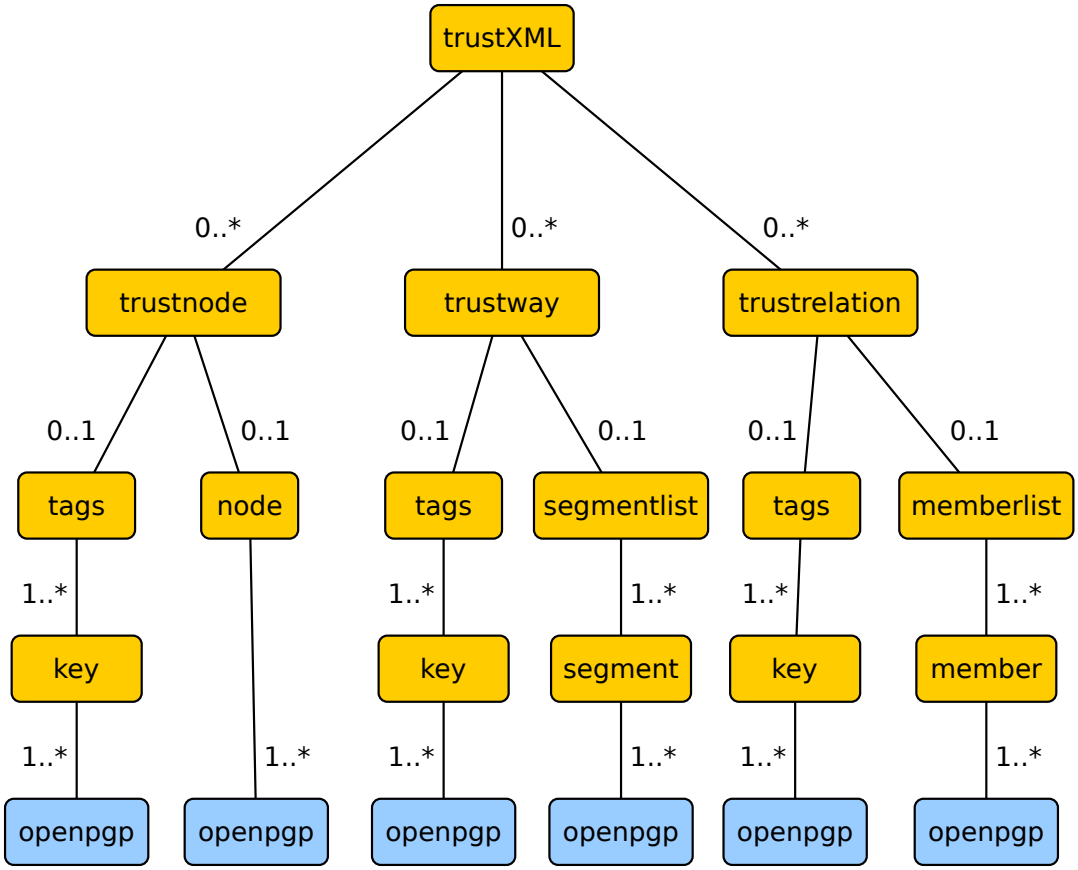

Abbildung 4.8: Darstellung des XML Baumes eines trustXML Dokumentes.

```
<!DOCTYPE trustXML [
 <!ELEMENT trustXML (trustnode|trustway|trustrelation)*>
 <!ATTLIST trustXML version CDATA #IMPLIED creator CDATA #IMPLIED >
 <!ELEMENT trustnode (tags?,node?)>
 <!ELEMENT trustway (tags?,segmentlist?)>
 <!ELEMENT trustrelation (tags?,memberlist?)>
 <!ATTLIST trustnode osmid CDATA #IMPLIED >
 <!ATTLIST trustway osmid CDATA #IMPLIED >
 <!ATTLIST trustrelation osmid CDATA #IMPLIED >
 <!ELEMENT tags (key)+>
 <!ELEMENT key (openpgp)+>
 <!ATTLIST key k CDATA #IMPLIED >
 <!ELEMENT node (openpgp)>
 <!ELEMENT segmentlist (segment)*>
 <!ELEMENT segment (openpgp)+>
 <!ELEMENT memberlist (member)*>
 <!ELEMENT member (openpgp)+>
 <!ELEMENT openpgp (#PCDATA)*>
]>
```

Abbildung 4.8 stellt den durch die DTD beschriebenen Aufbau des XML Dokumentes in einer Baumstruktur dar. Jeder Knoten entspricht einem XML-Tag und hat eine bestimmte Anzahl Kindknoten. Die mögliche Anzahl der Kindknoten ist in der Abbildung an den Kanten beschriftet. Als

Blattknoten treten lediglich openpgp-Tags auf. Sie besitzen keine weiteren Unterknoten und enthalten die Daten der jeweiligen Bewertung in Form eine OpenPGP clear signed message bestehend aus der Klartextnachricht und einer oder mehreren OpenPGP signed messages. Die XML-Elemente key, node, segment und member können ein oder mehrere openpgp-Knoten besitzen, je nachdem, wieviele verschiedene Klartexte zu einer durch sie referenzierten semantischen Einheit vorliegen. Verschiedene Klartexte zu einer semantischen Einheit entstehen immer dann, wenn eine bewertete Einheit geändert und erneut bewertet wird.

4.6 SICHERHEITSBETRACHTUNGEN

Das implementierte Bewertungssystem ist unter anderem aufgrund vieler Vereinfachungen wie z.B. der schwachen Reputationsfunktion, das Fehlen einer automatischen Analyse der Bewerterzertifikate etc. anfällig gegen Angriffe. Einige der in diesem Kapitel vorgestellten Angriffe lassen sich einfacher abwehren, andere sind nur schwer oder gar nicht abzuwenden.
Angreifer können dabei verschiedene Ziele verfolgen. Einige davon seien kurz genannt:

- Schädigung des Openstreetmap-Projektes in irgendeiner Form z.B. durch
  - Zerstörung der Arbeit Anderer (Daten, Bewertungen)
  - unbefugter Import von lizenzrechtlich geschütztem Datenmaterial
  - Beschädigung der Infrastruktur
- Schädigung der Projektmitarbeiter z.B. durch
  - Verletzung ihrer Privatsphäre
  - Rufschädigung
- Schädigung der Openstreetmap-Nutzer z.B. durch
  - gezielte Täuschung von Nutzern mit falschen Daten und Bewertungen
  - Behinderung bei der Nutzung der Daten und Bewertungen

Das Bewertungssystem schützt nicht vor der Verfälschung von Openstreetmap-Daten. Es kann jedoch der Aufklärung dienlich sein, ob eine Verfälschung vorliegt. Es wird demnach schwieriger die Arbeit Anderer unbemerkt zu zerstören. Die Einführung des Bewertungssystems eröffnet insofern ein neues Angriffsziel, als dass neben den Openstreetmap-Daten möglicherweise auch Bewertungen zerstört oder gefälscht werden können.
Hinsichtlich unbefugter Datenimporte kann das Bewertungssystem auch bei der Erkennung solcher helfen. Genehmigte Datenimporte könnten einfach durch den Lizenzinhaber signiert werden, um sie zu kennzeichnen.
Die Beschädigung der Projektinfrastruktur kann durch das System nicht verhindert werden. Da neue Infrastruktur hinzugefügt wird, wie z.B. der zentrale Server zu Bewertungsspeicherung, entsteht somit ein weiteres Angriffsziel.
Die Möglichkeit der Schädigung von Projektmitarbeitern wird häufig übersehen. Eine Verletzung der Privatsphäre kann stattfinden, wenn Interessenprofile etc. aus den beigetragenen Daten erstellt werden. Das Bewertungssystem ist jedoch nicht auf den Schutz der Mapper ausgelegt. Stattdessen bietet es Schutzmöglichkeiten der Bewerter da auch diese als Projektmitarbeiter in ihrer Privatsphäre angreifbar sind.
Eine Rufschädigung kann ebenso auf Mapper und Bewerter abzielen, wobei wiederum nur die Bewerter durch das System geschützt werden können.
Das Bewertungssystem unterstützt vor allem Nutzer von Openstreetmap, indem es die Vertrauenswürdigkeit von Daten kennzeichnet. Eine Täuschung wie bisher durchführbar, indem einfach

falsche Daten erzeugt werden, ist natürlich weiterhin möglich, sofern der Nutzer die Daten trotzdem verwendet. Er kann bei der Nutzung unbewerteter Daten jedoch vorsichtiger sein. Findet das Bewertungssystem eine große Verbreitung kann er eventuell sogar ganz auf die Nutzung von unbewerteten Daten verzichten ohne allzugroße Qualitätseinbußen bei der Kartennutzung in Kauf nehmen zu müssen. Gerade für die Anlaufzeit des Bewertungssystems ist das jedoch unrealistisch.

Zur Einschätzung der genannten Schutzmöglichkeiten des Bewertungssystems, werden nachfolgend Angriffe auf das Bewertungssystem vorgestellt, die letztendlich eines der genannten Angriffsziele erreichen sollen. Dabei wird zunächst ein Angreifermodell entwickelt, um die Eigenschaften und Möglichkeiten eines potentiellen Angreifers festzulegen.

4.6.1 Angreifermodell

Es soll von einem starken Angreifer ausgegangen werden, der Openstreetmap gut kennt, programmieren kann und ein Insider in dem Sinne ist, dass er bei Openstreetmap angemeldet ist und natürlich das Bewertungssystem uneingeschränkt nutzen kann.
Der Angreifer möchte unerkannt bleiben, damit er sein Werk lange fortsetzen kann. Auch seine Bearbeitungen, die gezielte Fehlinformationen beinhalten können, sollten nicht zu auffällig sein, damit sie nicht sofort von anderen Mappern wieder korrigiert werden. Sie sehen dementsprechend möglichst glaubwürdig aus und sind nicht leicht aufzuspüren.
Der Angreifer ist komplexitätstheoretisch beschränkt und kann somit keine gängigen als sicher geltenden asymmetrischen Kryptographiesysteme brechen. Er kann demzufolge keine digitalen Signaturen fälschen bzw. sich den privaten Schlüssel aus der Signatur berechnen.

4.6.2 Angriffe

Es wird im Folgenden untersucht, ob ein solcher Angreifer das Bewertungssystem zu seinem Vorteil missbrauchen kann. Setzt der Angreifer Attacken gegen das Reputationssystem ein, kann er damit versuchen die Glaubwürdigkeit der OSM-Daten zu beeinträchtigen oder eigene falsche Daten als glaubwürdig erscheinen zu lassen. Typische Angriffe auf Reputationssystem sind bei Peters [PR08] oder Hoffman [KNR07] beschrieben. Einige für das Bewertungssystem für Openstreetmap-Daten relevante Angriffe werden hier kurz vorgestellt.

Sybil-Attack

Die Sybil-Attack ist ein Angriff auf ein Reputationssystem, bei dem der Angreifer durch Erzeugen von vielen Identitäten versucht mehr Einfluss auf die Reputation zu erlangen. Da es beim implementierten System sehr leicht möglich ist neue Identitäten anzulegen, indem ein neues Schlüsselpaar erzeugt wird, ist diese Attacke vergleichsweise einfach durchzuführen.
Ein Angreifer kann die Schwachstelle ausnutzen, indem er beliebige falsche Daten in die Openstreetmap-Datenbank einträgt und diese mit den vielen Identitäten, die er besitzt positiv bewertet. Da die aktuelle Reputationsfunktion lediglich die Anzahl der unterschiedlichen Bewertungen ausdrückt, kann ein Nutzer getäuscht werden und die falschen Daten womöglich als glaubwürdig ansehen.

Ein möglicher Schutz ist durch Bewerterzertifikate möglich, die in den Reputationsalgorithmus mit eingehen. Allein die Anzahl der Zertifikate kann auch hier wieder keine Aussage über die Glaubwürdigkeit eines Bewerters treffen, denn der Angreifer könnte sich seine vielen Identitäten gegenseitig zertifizieren lassen auch „ballot stuffing“ genannt. Aus Sicht des Nutzers muss ein Bewerterzertifikat daher zur wirksameren Bekämpfung von Sybil-Attacken von einer vertrauenswürdigen Instanz stammen, die einen Bewerter im Sinne des Nutzers ausreichend überprüft hat.

Rest-on-the-laurels

Ein Angriff basierend auf einer ROTL (Rest-on-the-laurels) Strategie - zu deutsch „sich auf den Lorbeeren ausruhen“ - bezeichnet im Zusammenhang mit Reputationssystemen den Aufbau einer positiven Bewertung, um diese plötzlich negativ zu missbrauchen.
Die Möglichkeiten sich vor solchen Angriffen zu schützen, hängen im Wesentlichen davon ab, wie einfach es für einen Angreifer ist sich gute Reputation aufzubauen. Je nach dem was in den Reputationsalgorithmus eingeht, muss ein Angreifer dazu beispielsweise viele sinnvolle und richtige Bewertungen tätigen oder versuchen ein oder mehrere Bewerterzertifikate zu erhalten, denen ein zu täuschender Nutzer vertraut.

Ein Großteil der Verantwortung liegt zur Bekämpfung solcher Angriffe daher bei den zertifizierenden Instanzen. Abhängig davon, wie schnell und einfach diese getäuscht werden können bzw. sie positive Zertifikate ausstellen, ist der Angriff einfach durchführbar.
Denkbar für zukünftige Systeme ist es auch, dass die zertifizierenden Instanzen ihre ausgestellten Zertifikate widerrufen können, um die Wirkung einer ROTL-Attacke etwas einzudämmen.

Rufmord

Dieser Angriff zielt darauf ab die Glaubwürdigkeit von Bewertern zu senken, indem ihr Ruf beschädigt wird. Dies kann beispielsweise erreicht werden, indem im Namen eines Bewerters unsinnige Bewertungen abgegeben werden. Das Bewertungssystem schützt jedoch vor solchen Angriffen, sofern der geheime Signierschlüssel eines Bewerters sicher aufbewahrt wird und dem Angreifer nicht zugänglich ist.

Eine weitere Möglichkeit den Ruf von Bewertern zu schädigen, ist den Ruf der sie zertifizierenden Instanzen zu schädigen. Bekommt ein Angreifer ein Zertifikat der gleichen Instanz ausgestellt, die den anzugreifenden Bewerter zertifiziert hat und missbraucht es offensichtlich, könnte das den Ruf dieser Instanz schädigen.
Auch hierbei kann die Möglichkeit eines gezielten Zertifikatwiderrufs den Schaden etwas eindämmen.

Denial-of-Service

Ein DOS (Denial-of-Service) Angriff, bringt ein System dazu den „Dienst zu verweigern“, indem eine große Menge von Aufgaben generiert wird, die das System überlasten. Ein solcher Angriff ist im Bewertungssystem für Openstreetmap-Daten gleich an mehreren Stellen möglich. Zum einen kann der zentrale Server angegriffen werden, der die Bewertungen speichert und abrufbar macht. Dieser Angriff auf die Verfügbarkeit des Systems kann z.B. durch Mirrorserver entschärft werden, die den Inhalt des zentralen Servers spiegeln und anbieten. Außerdem können die Bewertungen dezentral weitergegeben werden z.B. über die Import/Export Funktion des Plugins.

Eine weitere Möglichkeit einen DOS Angriff auf das Bewertungssystem zu starten ist die Generierung einer Vielzahl von Bewertungen in der Hoffnung Kapazitätsgrenzen zu überschreiten oder das Plugin mit der Auswertung der Bewertungen zu überfordern. Der praktische Nutzen dieses Angriffs hängt davon ab, wieviel Rechenaufwand die Auswertung einer Bewertung im Vergleich zu ihrer Erstellung verursacht. Dieser Angriff kann auch als Angriff gegen die Skalierbarkeit des Systems angesehen werden. Je mehr Ressourcen ein System zur Verfügung hat, desto schwieriger wird es, es auszulasten.

Gegen DOS Angriffe auf das Bewertungssystem in der beschriebenen Art kann eine Milderung durch das Festlegen von Abbruchbedingungen bei der Auswertung von Bewertungen oder dem Importieren von Bewertungen in die Anwendung erreicht werden. Zwar ist die Funktion des Systems dann trotzdem eingeschränkt, allerdings findet die Behandlung des Problems auf einer höheren Ebene statt, als beispielsweise bei einem Speicherüberlauf oder sehr langen Programmablauf, der oft nur durch Beendigung des Programmes abzubrechen ist.

Zerstörung positiver Reputation

Eine Schwachstelle des implementierten Systems ist die einfache Zerstörbarkeit von abgegebenen Bewertungen und damit von Reputation. Da für die Auffindbarkeit und Gültigkeit der Bewertungen die Identifikationsnummer eines Openstreetmap-Objektes wichtig ist und diese durch Löschen und Neuanlegen eines Objektes geändert wird, können die zu diesem Objekt abgegebenen Bewertungen damit praktisch gelöscht werden. Das Neuanlegen des Objektes verschleiert hierbei, dass das ursprüngliche bewertete Objekt entfernt wurde. Ein Angreifer kann damit zwar keine falschen Daten positiv bewertet bekommen, aber immerhin die Vertrauenswürdigkeit von Daten senken.

Solche Angriffe können durch regelmäßige möglicherweise auch automatische Kontrollen der bewerteten Daten jedoch aufgedeckt werden.

4.7 ZUSAMMENFASSUNG IMPLEMENTIERUNG

Im Implementierungskapitel werden Details des für diese Diplomarbeit implementierten JOSM-Plugins dargelegt. Der Abschnitt 4.1 gibt zunächst einen kurzen Überblick in den OSM-Editor JOSM, wobei die Schwerpunkte auf der Einbindung (Abschnitt 4.1.1), der Nutzung (Abschnitt 4.1.2) und einem kurzen Überblick über die interne Struktur (Abschnitt 4.1.3) des Plugins liegen.

In Abschnitt 4.2 wird der Einsatz von OpenPGP als Format für die Signaturen der Bewertung vorgestellt. Hierbei wird näher auf die Nutzung des Notation Data Packets zur Unterbringung von Zusatzinformationen bei der Bewertung von Daten oder bei der Erstellung von Bewerterzertifikaten eingegangen (Abschnitt 4.2.1). Die Nutzung der OpenPGP Grundfunktionen geschieht mit dem freien Java Framework BouncyCastle, zu dem wesentliche Informationen bezüglich der Verwendung im Trustosm-Plugin in Abschnitt 4.2.2 genannt werden.

Ein für die Implementierung wichtiger Bestandteil des Plugins sind die Datenstrukturen zur Verwaltung der Bewertungen und deren Zuordnung zu den Openstreetmap-Daten. Die Datenstrukturen werden in Abschnitt 4.3 beschrieben. Die Klassen TrustOsmPrimitive und TrustSignatures spielen dabei eine zentrale Rolle.

In knapper Ausführung wird auf die Auswertung von Bewertungen durch die Klasse TrustAnalyzer im Abschnitt 4.4 eingegangen.

Für die Speicherung der Bewertungen wurde ein Austauschformat mit XML entwickelt, welches in Abschnitt 4.5 genau beschrieben wird.

Der Abschnitt 4.6 stellt einige Sicherheitsbetrachtungen zum implementierten System an. Neben den möglichen Angriffszielen wird hierbei ein Angreifermodell (Abschnitt 4.6.1) aufgestellt und einige Angriffe auf das Bewertungssystem, sowie mögliche Schutzvorkehrungen erläutert.

5 FAZIT UND AUSBLICK

Das Ziel dieser Diplomarbeit, ein Bewertungssystem für Openstreetmap-Daten zu entwickeln, wurde erreicht. Mit dem System sind Anwender erstmals in der Lage formal auszudrücken, dass von ihnen geprüfte OSM-Daten nach ihrer Einschätzung richtig sind. Nutzer des Openstreetmap-Kartenmaterials bekommen dadurch Anhaltspunkte geliefert, wie vertrauenswürdig die Openstreetmap-Daten im Detail sind.
Mit dem Einsatz von OpenPGP kann die abgegebene Bewertung gegen Manipulation geschützt werden. Mit den Möglichkeiten einer Zertifizierung von OpenPGP-Schlüsseln kann ein Nutzer zusätzliche Reputation durch andere Nutzer erhalten und seinen Bewertungen damit mehr Glaubwürdigkeit verleihen. Dabei wurde das bewährte Konzept eines sogenannten „Web of Trust“ für die Zertifizierung von Openstreetmap-Bewertern adaptiert. Jeder Nutzer kann jeden anderen zertifizieren.
Welche Zertifikate bei der Analyse einer Bewertung wie gewichtet werden, wie Bewertungen mit gegensätzlichen Aussagen behandelt werden, und welche zusätzlichen Informationen in den Reputationswert miteinfließen, entscheidet die Reputationsfunktion. Da diese Aufgabe sehr komplex ist und den Umfang dieser Diplomarbeit überstiegen hätte, wurde nur eine sehr einfache und praktisch wenig hilfreiche Reputationsfunktion implementiert, die lediglich die Anzahl der für ein Datum abgegebenen gültigen Bewertungen darstellt. In dieser Arbeit wurden jedoch weitergehende Ideen zu Reputationsfunktionen ausgeführt und eine Formel zur Berechnung von Reputation für Openstreetmap-Objekte im Anhang entsprechend hergeleitet. Dabei wurde das Problem der Bewerterreputation jedoch noch nicht berührt.
Eine praktisch relevante Reputationsfunktion sollte in der Lage sein, die Bewerterzertifikate bzw. weitere Informationen des Bewerters, die über den öffentlichen OpenPGP-Schlüssel zur Verfügung stehen, miteinzubeziehen. In diesem Zusammenhang müsste die genaue Aussage einer Bewertung und der Inhalt noch einmal genauer überdacht und gegebenenfalls angepasst werden.

Im Zuge dieser Arbeit wurden Möglichkeiten aufgezeigt, wie sich ein Bewerter schützen kann, der aufgrund der Daten, die er hinterlässt, einen Angriff auf seine Privatsphäre fürchtet. Der Einsatz verschiedener OpenPGP-Schlüssel bei verschiedenen Bewertungen unter Nutzung von Pseudonymen kann zu diesem Schutz beitragen. Ein Identitätsmanagementsystem unterstützt den Nutzer optional bei der Erstellung von Identitäten und den Entscheidungen ihres Einsatzes. Die Anbindung eines solchen Systems an das Bewertungssystem ist eine Aufgabe zukünftiger Entwicklungen.

Der Prototyp des Trustosm-Plugins für den Openstreetmap-Editor JOSM zeigt, dass eine Bewertung und Analyse einzelner Teile eines Openstreetmap-Objektes möglich ist und ohne viel zusätzlichen Aufwand von OSM-Anwendern durchgeführt werden kann. Der OSM-Anwender sollte jedoch mit der grundsätzlichen Bedienung des Editors JOSM vertraut sein, was auf die meisten

Mapper zutrifft. Sollte das System in Zukunft breitere Anwendung finden, lassen sich auch neue eigenständige Applikationen entwickeln, die speziell auf die Bewertungsanforderungen ausgerichtet sind. Denkbar sind unter anderem Webanwendungen, wobei bei einer Bewertungsabgabe beispielsweise mit dem Webbrowser die Sicherheitsstrategien erneut überdacht werden müssen. Zur Visualisierung von globalen Reputationswerten, errechnet aus den abgegebenen Bewertungen, wären Onlinekarten jedoch bestens geeignet.

Der Plugin-Prototyp erfüllt bisher noch nicht alle Anforderungen, die in der Analyse festgelegt wurden. So sollte in Zukunft ein Rückruf abgegebener Bewertungen eingebaut werden. Weiterhin fehlt im Moment noch eine vollständige Schlüsselverwaltung, die in der Lage ist öffentliche Schlüssel zu importieren und zu exportieren. Die rudimentäre bisher implementierte Schlüsselverwaltung des Prototypen ist in der Lage, neue Schlüssel anzulegen, Informationen über Schlüssel anzuzeigen und Schlüssel zur Abgabe einer Bewertung auszuwählen. Da die Dateien der Schlüsselverwaltung jedoch vollständig kompatibel zu dem frei verfügbaren Programm GnuPG sind, kann die Schlüsselverwaltung in der Zwischenzeit auch über dieses Programm durchgeführt werden und ist somit nicht essenziell für die Nutzung des Bewertungssystems durch das Plugin. Bei der Nachrüstung des Plugins mit den genannten Funktionen werden allerdings keine Probleme erwartet.

Schwierigkeiten sind bisher vor allem bei der genauen Festlegung der zu signierenden Daten aufgetreten. In dieser Arbeit wurde versucht, den Inhalt eines Openstreetmap-Objektes in möglichst kleine semantische Einheiten aufzuteilen, die einzeln für sich bewertet werden können. Eine semantische Einheit ist in diesem Fall ein einzelner Tag eines OSM-Objektes, ein Member einer Relation, die Lage eines Nodes oder ein Segment eines Ways. Über die OSM-ID werden die Einheiten miteinander in Verbindung gebracht.
Schwächen des Openstreetmap-Datenmodells sowie Verbesserungsvorschläge wurden in diesem Zusammenhang in der Arbeit ebenfalls untersucht.

Zukünftige Praxistests werden zeigen, ob das System auch so, wie es implementiert ist zusammen mit dem aktuellen Datenmodell von Openstreetmap sinnvoll genutzt wird. Damit jedoch das System praktisch eingesetzt werden kann, sollte zunächst ein zentraler Server zur Speicherung der Bewertungen aufgesetzt werden. Vorschläge wie dessen API gestaltet sein könnte und welche Speicherstrukturen in Frage kommen, wurden in dieser Arbeit gemacht.
Ebenso wurde ein XML Austauschformat für die Bewertungsdaten festgelegt mit dem sowohl eine Datenübertragung zu diesem Server stattfinden kann, als auch eine Speicherung in Dateien möglich ist. Letzteres wurde bereits implementiert.

Dass das entwickelte System gegen Angriffe verwundbar ist, konnte in dieser Arbeit ebenso gezeigt werden. Es wird eine zukünftige Aufgabe sein, die Schutzmechanismen gegen die Angriffe weiter zu erhöhen, wo das notwendig ist. Nicht jeder Angriff lohnt seinen Aufwand. Nicht aufwändige und dennoch effektive Angriffe gegen das implementierte System, wie z.B. die Sybil-Attack können beispielsweise durch eine verbesserte Reputationsfunktion entschärft werden.

Wenn das System in den wichtigen genannten Funktionen verbessert und von der Openstreetmap-Community akzeptiert und eingesetzt wird, kann es sich zu einem wertvollen Werkzeug der Qualitätssicherung in Openstreetmap entwickeln.

A ANHANG

Der folgende Anhang zeigt die Herleitung einer möglichen Formel zur Berechnung von Reputation mit Methoden der Wahrscheinlichkeitsrechnung auf. Die Formel wurde nicht implementiert und stellt lediglich eine Idee zur Lösung der Reputationsproblematik dar. Der Idee liegt zugrunde, dass die Glaubwürdigkeit von Bewertern in einer Wahrscheinlichkeit ausgedrückt werden kann mit der die Aussage ihrer Bewertung bei einer Prüfung eintritt. Dieser Wert kann auch als Bewerterreputation angesehen werden. Die Reputationsfunktion versucht nun aus einer Schätzung, wieviel Prozent der Openstreetmap-Daten als richtig angenommen werden und den Aussagen der Bewerter, sowie der zugehörigen Bewerterreputation einen Reputationswert für ein Openstreetmap-Datum zu berechnen. Dabei fließt die Fähigkeit der Bewerter richtige von falschen Daten zu unterscheiden und zu kennzeichnen in den Reputationswert mit ein.

A.1 HERLEITUNG DER REPUTATIONSFUNKTION

Gegeben ist ein Zufallsexperiment mit zwei möglichen Ereignissen. Es bezeichne X eine zweiwertige Zufallsvariable, die den zwei Ereignissen des Bernoulli-Experimentes wie folgt zugeordnet ist:

Das Openstreetmap-Objekt ist korrekt $\rightarrow \quad X = 1$
Das Openstreetmap-Objekt ist falsch $\rightarrow \quad X = 0$

Weiterhin seien endlich viele Personen gegeben, die eine Einschätzung $x_i \in \{0, 1\}$ des Openstreetmap-Objektes abgegeben haben, wobei die Bedeutung der Werte von x_i denen von X entspricht. Es soll angenommen werden, dass die Einschätzungen der Personen unabhängig voneinander getroffen werden. Die Ereignisse x_i sind also alle voneinander stochastisch unabhängig.
Jeder Person i können weiterhin zwei Wahrscheinlichkeiten zugeordnet werden. Zum einen die Wahrscheinlichkeit mit der Einschätzung x_i richtig zu liegen, unter der Bedingung, dass $X = 1$ ist, also $P(x_i|1)$ und zum anderen die Wahrscheinlichkeit mit der Einschätzung x_i richtig zu liegen unter der Bedingung, dass $X = 0$ ist, also $P(x_i|0)$. Diese Wahrscheinlichkeiten sind Ausdruck des indirekten Vertrauens des Nutzers gegenüber dieser dritten Person i.
Zudem seien die A-priori-Wahrscheinlichkeiten $P(X)$ gegeben, die sich aus den Openstreetmap-Daten durch Stichproben annähern lassen und angeben, wie hoch die Wahrscheinlichkeit ist auf ein richtiges bzw. falsches Openstreetmap-Datum zu treffen.

Gesucht ist nun die Wahrscheinlichkeit $P(X|\{x_i\})$, also die Wahrscheinlichkeit dafür, dass die Openstreetmap-Daten richtig bzw. falsch sind unter der Bedingung, dass eine Menge von Einschätzungen $\{x_i\}$ dritter Personen i bekannt ist.

Die **Berechnung** erfolgt über den Satz von Bayes:

$$P(X|\{x_i\}) = \frac{P(\{x_i\}|X) \cdot P(X)}{P(\{x_i\})} \tag{A.1}$$

Das Ereignis $\{x_i\}$ setzt sich zusammen aus allen Teilereignissen x_i:

$$\{x_i\} = \bigcap_i x_i$$

Es gilt weiterhin:

$$P(\{x_i\}|X) = P\left(\bigcap_i x_i|X\right) = P\left(\bigcap_i (x_i|X)\right)$$

Aus der vorausgesetzten stochastischen Unabhängigkeit der Einzelereignisse x_i folgt die stochastische Unabhängigkeit der Ereignisse $(x_i|X)$. Somit kann die Verbundwahrscheinlichkeit durch Multiplikation der Wahrscheinlichkeit der Teilereignisse gebildet werden:

$$P(\{x_i\}|X) = P\left(\bigcap_i (x_i|X)\right) = \prod_i P(x_i|X) \tag{A.2}$$

Die Wahrscheinlichkeit für das Ereignis $\{x_i\}$ ergibt sich aus den bedingten Wahrscheinlichkeiten mit dem Gesetz der totalen Wahrscheinlichkeit wie folgt:

$$P(\{x_i\}) = P(\{x_i\}|X) \cdot P(X) + P(\{x_i\}|\bar{X}) \cdot P(\bar{X}) \tag{A.3}$$

Der Wert der Zufallsvariablen des Gegenereignisses $\bar{X}$ zum Ereignis X lässt sich leicht berechnen:

$$\bar{X} = 1 - X$$

Damit und unter Zuhilfenahme der Formel A.2 lässt sich $P(\{x_i\})$ durch Einsetzen in die Gleichung A.3 berechnen:

$$P(\{x_i\}) = \prod_i P(x_i|X) \cdot P(X) + \prod_i P(x_i|1-X) \cdot P(1-X) \tag{A.4}$$

Die Wahrscheinlichkeit $P(1-X)$ lässt sich aufgrund der Bernoulli-Verteilung berechnen mit:

$$P(1-X) = 1 - P(X)$$

Bedient man sich noch einmal der Gleichung A.2 und setzt die Gleichung A.4 in unsere Ausgangsgleichung A.1 ein ergibt sich:

$$P(X|\{x_i\}) = \frac{\prod_i P(x_i|X) \cdot P(X)}{\prod_i P(x_i|X) \cdot P(X) + \prod_i P(x_i|1-X) \cdot (1-P(X))} \tag{A.5}$$

durch Umformung ergibt sich:

$$P(X|\{x_i\}) = \frac{1}{1 + \frac{1-P(X)}{P(X)} \cdot \prod_i \frac{P(x_i|1-X)}{P(x_i|X)}} \tag{A.6}$$

	Bewerter 1		Bewerter 2		Bewerter 3	
X \ x_i	1	0	1	0	1	0
1	0,9	0,1	0,9	0,1	0,8	0,2
0	0,9	0,1	0,3	0,7	0,1	0,9

Tabelle A.1: Die Tabelle listet alle bedingten Wahrscheinlichkeiten der drei Beispielbewerter.

A.2 RECHENBEISPIELE

Für die Rechenbeispiele wird angenommen, dass 90% aller Openstreetmap-Daten korrekt sind. Die A-priori-Wahrscheinlichkeit ist demnach $P(X = 1) = 0{,}9$.
Es seien drei Bewerter gegeben, die unterschiedliche Bewertungsstile haben.
Bewerter 1 trifft seine Einschätzungen zufällig. Bewerter 2 neigt dazu die Daten zu überschätzen und Bewerter 3 unterschätzt sie gerne. Die genauen Wahrscheinlichkeiten sind in Tabelle A.1 festgelegt.

Bewerter 1 benutzt einen Ikosaeder (zwanzigseitiger Spielwürfel) und gibt die Einschätzung von 0 immer dann ab, wenn er bei einem Wurf eine 1 oder eine 20 erzielt. Das heißt, er wird mit einer Wahrscheinlichkeit von $P(x_i = 1|X = 1) = 0{,}9$ mit der Einschätzung, dass ein Datum korrekt ist richtig liegen. Dafür ist er beim Finden von Fehlern sehr schlecht und 90% der Fehler schätzt er als korrekte Daten ein: $P(x_i = 1|X = 0) = 0{,}9$.
Die Bewerter geben ihre Einschätzungen eines Objektes wie folgt ab:

$$\{x_1, x_2, x_3\} = \{1, 1, 0\}$$

Es soll nun die Wahrscheinlichkeit berechnet werden, dass das zu untersuchende Openstreetmap-Datum unter Berücksichtigung der Einschätzungen mit der jeweiligen Wahrscheinlichkeit aus Tabelle A.1 richtig ist.

$$\begin{aligned} P(1|\{1,1,0\}) &= \frac{1}{1 + \frac{1-P(1)}{P(1)} \cdot \prod_i \frac{P(x_i|0)}{P(x_i|1)}} \\ &= \frac{1}{1 + \frac{1-0{,}9}{0{,}9} \cdot \frac{0{,}9}{0{,}9} \cdot \frac{0{,}3}{0{,}9} \cdot \frac{0{,}9}{0{,}2}} \end{aligned}$$

Da Bewerter 1 nur rät, spielt sein Faktor bei der Berechnung des Produktes offensichtlich keine Rolle und ergibt 1. Dabei wäre es auch unerheblich mit wievielen Zahlen er sich beim würfeln für seine Einschätzung entscheidet.

$$\begin{aligned} &= \frac{1}{1 + \frac{1}{9} \cdot 1 \cdot \frac{3}{9} \cdot \frac{9}{2}} = \frac{1}{1 + \frac{1}{6}} = \frac{6}{7} \\ &= 0{,}\overline{857142} \end{aligned}$$

Die Wahrscheinlichkeit, dass das Openstreetmap-Datum korrekt ist, hat sich also mithilfe der Einschätzungen von der A-priori-Wahrscheinlichkeit von 0,9 auf weniger als 0,86 verringert. Grund dafür ist Bewerter 3, der das Datum als falsch eingeschätzt hat und dessen Einschätzungen im Verhältnis besser sind, als jene von Bewerter 2.

Würde man die Aussagen von Bewerter 2 und 3 invertieren ergäbe sich die Wahrscheinlichkeit wie folgt:

$$
\begin{aligned}
P(1|\{1,0,1\}) &= \frac{1}{1+\frac{1-P(1)}{P(1)}\cdot\prod_i \frac{P(x_i|0)}{P(x_i|1)}} \\
&= \frac{1}{1+\frac{1-0,9}{0,9}\cdot\frac{0,9}{0,9}\cdot\frac{0,7}{0,1}\cdot\frac{0,1}{0,8}} \\
&= \frac{1}{1+\frac{1}{9}\cdot 1\cdot 7\cdot\frac{1}{8}} = \frac{1}{1+\frac{7}{72}} = \frac{72}{79} \\
&= 0,\overline{9113924050632}
\end{aligned}
$$

In diesem Falle wäre die Wahrscheinlichkeit, dass das Objekt korrekt ist mithilfe der Bewerterinformationen auf mehr als 0,91 leicht gestiegen, gegenüber der Ausgangswahrscheinlichkeit von 0,9. Bewerter 3 fällt wiederum mehr ins Gewicht.

Bliebe noch die Frage wie die Wahrscheinlichkeit steht, wenn Bewerter 2 und 3 beide das Openstreetmap-Objekt als falsch einschätzen. Zum Verdeutlichen, dass die Einschätzung vom zufällig ratenden Bewerter 1 keine Rolle spielt, wird sein Tip jetzt auch als 0 angenommen. Es ergibt sich:

$$
\begin{aligned}
P(1|\{0,0,0\}) &= \frac{1}{1+\frac{1-P(1)}{P(1)}\cdot\prod_i \frac{P(x_i|0)}{P(x_i|1)}} \\
&= \frac{1}{1+\frac{1-0,9}{0,9}\cdot\frac{0,1}{0,1}\cdot\frac{0,7}{0,1}\cdot\frac{0,9}{0,2}} \\
&= \frac{1}{1+\frac{1}{9}\cdot 1\cdot 7\cdot\frac{9}{2}} = \frac{1}{1+\frac{7}{2}} = \frac{2}{9} \\
&= 0,\overline{2}
\end{aligned}
$$

Offenbar ist die Wahrscheinlichkeit, dass das Objekt richtig ist durch die übereinstimmenden Einschätzungen der Bewerter, dass es falsch sei, stark gesunken.

LITERATURVERZEICHNIS

[AD07] B.T. Adler and L. De Alfaro. A content-driven reputation system for the wikipedia. In *Proceedings of the 16th international conference on World Wide Web*, pages 261–270. ACM, 2007.

[AdAKP10] B. Adler, L. de Alfaro, A. Kulshreshtha, and I. Pye. Reputation systems for open collaboration. *Journal of the ACM*, 2010.

[BPM+08] Tim Bray, Jean Paoli, Eve Maler, François Yergeau, and C. M. Sperberg-McQueen. Extensible markup language (XML) 1.0 (fifth edition). W3C recommendation, W3C, November 2008. http://www.w3.org/TR/2008/REC-xml-20081126/.

[CDF+07] J. Callas, L. Donnerhacke, H. Finney, D. Shaw, and R. Thayer. Openpgp message format. RFC 4880, Internet Engineering Task Force, November 2007.

[DFW05] C. Dellarocas, M. Fan, and C.A. Wood. Self-interest, reciprocity, and participation in online reputation systems. *Technical Report Paper 205*, February 2005.

[JIB07] Audun Jøsang, Roslan Ismail, and Colin A. Boyd. A survey of trust and reputation systems for online service provision. *Decision Support Systems*, 43(2):618–644, March 01 2007.

[Jiv10] A. Jivsov. ECC in OpenPGP (IETF Internet-Draft <draft-jivsov-openpgp-ecc-06.txt>). An online version is available at `http://tools.ietf.org/html/draft-jivsov-openpgp-ecc-06` (23.12.2010), September 2010. Expires March 17, 2011.

[KNR07] David Zage Kevin Hoffman and Cristina Nita-Rotaru. A survey of attack and defense techniques for reputation systems. Technical Report CSD TR #07-013, Purdue University, 2007.

[MKtH06] M. Modsching, R. Kramer, and K. ten Hagen. Field trial on gps accuracy in a medium size city: The influence of built-up. In *3rd Workshop on Positioning, Navigation and Communication*, 2006.

[NIS09] NIST. *Digital Signature Standard (DSS) (FIPS PUB 186-3)*. National Institute of Standards and Technology, June 2009.

[PR08] R. Peters and I. Reitzenstein. Robuste reputationssysteme auf elektronischen märkten. *Multikonferenz Wirtschaftsinformatik (MKWI)*, 2008.

[Riv97] Alfred J. Menezes and Paul C. Van Oorschot and Scott A. Vanstone and R. L. Rivest. *Handbook of Applied Cryptography*. CRC Press, 1997.

[SGM09] Sandra Steinbrecher, Stephan Groß, and Markus Meichau. Jason: A scalable reputation system for the semantic web. In Dimitris Gritzalis and Javier Lopez, editors, *Emerging Challenges for Security, Privacy and Trust*, volume 297 of *IFIP Advances in Information and Communication Technology*, pages 421–431. Springer Boston, 2009. 10.1007/978-3-642-01244-0_37.

[Ste08] Sandra Steinbrecher. *Mehrseitige Sicherheit in Reputationsystemen*. PhD thesis, TU Dresden, April 2008.