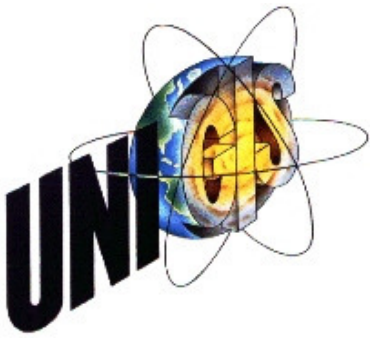


Master Thesis

im Rahmen des
Universitätslehrganges „Geographical Information Science & Systems“
(UNIGIS MSc) am Zentrum für GeoInformatik (Z_GIS)
der Paris Lodron-Universität Salzburg

zum Thema

„WFS Simple - OpenGIS für den Massenmarkt“

vorgelegt von

Martin Rinner
U1341, UNIGIS MSc Jahrgang 2007

Zur Erlangung des Grades
„Master of Science (Geographical Information Science & Systems) – MSc(GIS)”

Gutachter:
Ao. Univ. Prof. Dr. Josef Strobl

Regensburg, 27. Oktober 2009

Erklärung der eigenständigen Abfassung der Arbeit

Ich versichere, diese Master Thesis ohne fremde Hilfe und ohne Verwendung anderer als der angeführten Quellen angefertigt zu haben, und dass die Arbeit in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen hat. Alle Ausführungen der Arbeit, die wörtlich oder sinngemäß übernommen wurden, sind entsprechend gekennzeichnet.

Regensburg, 27. Oktober 2009

Kurzfassung

Mit der Veröffentlichung von Google Maps Anfang 2005 (Google 2005) setzte ein Boom der Nutzung geographischer Inhalte ein. Dieser sogenannte Geomassenmarkt entwickelt sich schnell. Dennoch existieren kaum Standards zum Datenaustausch über Web Services.

Das OGC, das Standards zum Datenaustausch in Geodateninfrastrukturen entwickelt, hat den WFS Simple als Dienst zum Datenaustausch im Geomassenmarkt entworfen.

In der vorliegenden Arbeit soll überprüft werden, ob der Standard WFS Simple den Anforderungen des Geomassenmarkt entspricht. Die Überprüfung gliedert sich in drei Schritte:

Definition. Nach der Definition des Begriffs Geomassenmarkt werden dessen Merkmale erarbeitet.

Anforderungsanalyse. Durch die Analyse bestehender Webanwendungen werden die Anforderungen des Geomassenmarkts an einen Standard zur Bereitstellung von Vektordaten aufgezeigt.

Prototypische Implementierung. Die bei der Implementierung gewonnenen Erfahrungen sollen ebenfalls bei der Bewertung des WFS Simple einfließen.

Abstract

As Google Maps was published in 2005 (Google 2005), a sudden increase in the use of geospatial content could be noticed. This so called geo mass market is still growing rapidly. Despite these facts, standards of exchanging data via web services still don't exist.

The OGC which is working on the implementation of these standards created the WFS Simple as a tool for the geo mass market to exchange data.

The purpose of this thesis is to examine the WFS Simple. Is a standard fitting in the demands of the geo mass market? The examination is structured into three steps:

Definition. A definition of the geo mass market and its characteristics.

Analysis. By analyzing existing web applications the demands of the geo mass market on a standard for vector data will be pointed out.

Implementation. Experiences made by the implementation will help to evaluate the WFS Simple.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

Ajax	Asynchronous JavaScript and XML
API	Application Programming Interface
GeoJSON	Geographic JavaScript Object Notation
GeoRSS	Geographic Really Simple Syndication
GML	Geographic Markup Language
JSON	JavaScript Object Notation
KML	Keyhole Markup Language
KML SWG	KML Standards Working Group
MMWG	Mass Market Working Group
OGC	Open Geospatial Consortium
RSS	Really Simple Syndication
WFS	Web Feature Service
WFS Simple	Web Feature Service Simple
WMS	Web Mapping Service

Kapitel 1

Einleitung

Mit der Veröffentlichung von Google Maps im Februar 2005 (Google 2005) und Google Earth im Juni 2005 (Heise 2005:o.S.) setzte ein Boom ein: auf einmal war das Interesse einer breiten Masse an geographischen Inhalten geweckt. Infolge dieses Booms finden sich auf vielen Websites kleine eingebettete Karten, es entstehen virtuelle Städte und andere Internetuser betrachten nur die Erdoberfläche, teilen anderen Nutzern sehenswerte Objekte mit und entdecken auch mal römische Villen (vgl. Höffken 2009).

Dieser Geomassenmarkt ist dynamisch entstanden, entsprechend gibt es eine Menge an Austauschformaten und –standards. Bei einem Teil von ihnen wacht, anders als bei Standards des Open Geospatial Consortiums (OGC), kein Gremium über deren Weiterentwicklung.

Auf der anderen Seite haben sich Standards, allen voran die des OGC, als hilfreich erwiesen, um Interoperabilität zwischen verschiedenen Datenlieferanten und –nutzern zu erreichen. Einer der Standards ist der „Web Feature Service Simple“, der vom OGC für den Datenaustausch im Geomassenmarkt konzipiert wurde.

1.1 Zielsetzung

Die vorliegende Arbeit untersucht, ob die Spezifikationen des WFS Simple für die Anforderungen des Geomassenmarkts an einen Dienst zur Bereitstellung von Vektordaten geeignet sind.

1.2 Gliederung der Arbeit

In **Kapitel 2** werden die Grundlagen für die Arbeit beschrieben.

Nach der Definition des Begriffs „Geomassenmarkt“ (Kapitel 2.1) wird die Beziehung zwischen Geomassenmarkt und OGC erläutert (Kapitel 2.2). Im Anschluss daran werden in Kapitel 2.3 die Kennzeichen des Geomassenmarkts an Hand von Technologien, Trends, Datenquellen und -formaten beschrieben.

Kapitel 3 behandelt die Bereitstellung von Vektordaten im Geomassenmarkt.

In der ersten Hälfte des Kapitels werden verschiedene Möglichkeiten einer Bereitstellung von Geodaten beschrieben. Im zweiten Teil schließt sich eine Analyse bestehender Anwendungen mit dem Ziel an, Anforderungen des Geomassenmarkts an einen Standard zur Bereitstellung von Vektordaten abzuleiten.

Im **Kapitel 4** wird der WFS Simple vorgestellt.

In diesem Kapitel werden die Grundzüge des Web Feature Service Simple-Standards erläutert sowie Unterschiede zum Web Feature Service erarbeitet.

Ein Prototyp des WFS Simple wird in **Kapitel 5** realisiert.

Es werden die einzelnen Schritte vom Entwurf bis zur Implementierung und Evaluierung beschrieben. Schwerpunkte sind dabei das Klassendesign des Prototypen, die Einbindung des Microsoft SQL Server 2008 als Datenquelle sowie die Filterung mittels Regulärer Ausdrücke. Abschließend werden noch Defizite des Prototypen dargestellt.

Im **Kapitel 6** erfolgt die Bewertung des WFS Simple-Standards.

Nach der Diskussion der Spezifikationen wird überprüft, ob der WFS Simple-Standard den in Kapitel 3 erarbeiteten Anforderungen des Geomassenmarkts genügt.

Kapitel 2

Definition und Kennzeichen des Geomassenmarkts

2.1 Definition

Um zu überprüfen, ob der WFS Simple-Standard den Anforderungen des Geomassenmarkts gerecht wird, muss zuerst der Begriff „Geomassenmarkt“ definiert werden. Dabei ist von Interesse, wie das OGC als Standardisierungsgremium den Begriff Geomassenmarkt definiert.

Allgemeine Definition „Massenmarkt“

Als Massenmarkt wird meist ein Markt gesehen, der durch eine hohe Nachfrage und durch hohe Stückzahlen gekennzeichnet ist.

Definition des OGC

Mit der Einführung von KML als offizieller OGC-Standard definiert das OGC den Begriff „Geomassenmarkt“. Dieser wird unter Anderem durch folgende drei Punkte gekennzeichnet (vgl. Open Geospatial Consortium Inc. 2009:vii):

- Der Geomassenmarkt besteht aus Millionen von Usern, von denen der Großteil keine Experten im Bereich von Geoinformationen ist

- Es existiert eine große Zahl an KML-Dateien und Anwendungen, die diese nutzen
- Die Zahl der Nutzer, der Geodaten und der Anwendungen wächst schnell.

Diese erste Definition des OGC greift jedoch zu kurz. Zum Einen existieren neben KML noch weitere massenmarkttaugliche Vektorformate (siehe Kapitel 2.3.4). Vor Allem aber lässt die Definition des OGC viele Faktoren, die den Geomassenmarkt prägen und ihn gegenüber den etablierten Geoinformationsmärkten unterscheidet, außen vor.

Bevor die den Geomassenmarkt prägenden Kennzeichen besprochen werden, soll zunächst eine eigene, umfassendere Definition des Begriffs „Geomassenmarkt" gegeben werden.

Eigene Definition

Der Begriff Geomassenmarkt bezeichnet eine neuartige Form der Nutzung von geographischen Inhalten, die eng an das Internet als technische Basis und Kommunikationsplattform gebunden ist.

Als Akteure kommen alle Nutzer des Internets in Betracht, unabhängig von ihrem Wissen im Bereich der Geoinformationen, ihrem Bewusstsein für geographische Inhalte und ihrer Nutzungsform des Internet. Von den Beteiligten des Geomassenmarkts wird eine große Zahl an Inhalten in den unterschiedlichsten Datenquellen und -formaten bereitgestellt, um in einer Vielzahl von Anwendungen genutzt und visualisiert zu werden.

Die Kennzeichen, die den Geomassenmarkt nach dieser Definition ausmachen und prägen, werden später in diesem Kapitel erarbeitet. Zuvor ist es jedoch erforderlich, mögliche Grenzen und Gemeinsamkeiten zwischen dem OGC und dem Geomassenmarkt aufzuzeigen.

2.2 OGC und Geomassenmarkt

2.2.1 Abgrenzung OGC gegen Geomassenmarkt

Ein zentraler Punkt obiger Definition ist die Gruppe der Akteure, die einmal aus allen Nutzern des Internet besteht und damit sehr groß ist, und zum Anderen das meist nicht vorhandene Expertenwisssen der Akteure im Bereich der Geoinformationstechnologie. Daraus ergibt sich eine Zweiteilung des GI-Marktes, der aus dem Bereich der etablierten GI-Standards und aus dem sich schnell entwickelndem Geomassenmarkt. besteht (vgl. Parsons 2006:o.S.).

Diese Zweiteilung wird auch in der Auswahl der verwendeten Technologien deutlich. Während der Bereich der etablierten GI-Systeme auf OGC-Standards setzt, nutzt der Geomassenmarkt vielfach andere Datenformate und –Standards (siehe Kapitel 2.3.2.

2.2.2 Berührungspunkte zwischen OGC und Geomassenmarkt

Trotz dieser Zweiteilung gibt es Berührungspunkte zwischen dem Geomassenmarkt und dem OGC. Zwei davon sind die Mass Market Working Group und KML.

Mass Market Working Group (MMWG)

Die MMWG wurde im Dezember 2006 (MMWG 2009:o.S.) und damit knapp zwei Jahre nach der Veröffentlichung von Google Maps gegründet. Ziel der MMWG ist es u. a., eine Folge von auf die Bedürfnisse des Geomassenmarkts zugeschnittener Dienste zu definieren (vgl. ebd.).

Unter der Führung der MMWG werden hauptsächlich zwei Standards entwickelt: der Vorschlag eines „Web Feature Service Simple“ sowie eines „Web Map Tiling Service Standard“ (Singh 2008).

Keyhole Markup Language (KML)

KML, ein im Geomassenmarkt weit verbreitetes Datenformat, wurde als Industriestandard von Galdos entwickelt und erst später als OGC-Standard anerkannt. Der Prozess der Anerkennung wurde von der KML Standards Working Group (KML SWG) bearbeitet (vgl. Open Geospatial Consortium Inc. 2008a:o.S.).

Deren Aufgabe war es, die von der Öffentlichkeit eingebrachten Kommentare zu bearbeiten und die Konformität zum Regelwerk des OGC und die Kompatibilität zu älteren KML-Versionen sicherzustellen. (vgl. ebd.).

2.3 Kennzeichen des Geomassenmarkts

In Kapitel 2.1 wurde der Begriff „Geomassenmarkt“ definiert, wie er im weiteren Verlauf dieser Arbeit verwendet werden soll. Wesentlicher Kern dieser Definition ist die Nutzung des Internets sowohl als technische Basis als auch als Kommunikationsplattform.

Die Faktoren, die den Geomassenmarkt kennzeichnen und in Schlagworte und Trends sowie in Technologien unterteilt werden können, werden im Folgenden besprochen.

2.3.1 Schlagworte und Trends

2.3.1.1 Web 2.0

In seinem Artikel „The Year of Web Services“ vom Dezember 2003 stellt Eric Knorr (vgl. 2003:o.S.) die bevorstehende Bedeutung von Webservices dar. Die Nutzung von Webservices sei nichts anderes als das, was Scott Dietzen als Web 2.0 bezeichnet habe (vgl. ebd.:o.S.).

Tim O’Reilly greift den Begriff „Web 2.0“ in seinem Artikel “What is Web 2.0” auf. Er bezeichnet Web 2.0 als Konzept (vgl. O’Reilly 2005:1), das keine neue Technologie darstellt, sondern eine neue Nutzungsart von bestehenden Technologien und Standards.

Ausgangspunkt seiner Definition ist die Feststellung, dass das Internet als Plattform für alle Arten von Anwendungen dient. Durch den Einsatz von Technologien wie Flash, Flex und AJAX entstehen Anwendungen, die eine ähnliche Usability wie Desktopanwendungen aufweisen (vgl. Koops 2008:24). Auch die Grenzen, die einzelne Geräteplattformen voneinander abgrenzen, verschwinden.

Ein weiterer zentraler Punkt ist der Begriff „Harnessing Collective Intelligence“ (O’Reilly 2005:o.S.), auch als „Mitmachweb“ bekannt. User erstellen Inhalte, sei es in Form von Blogs, Beiträgen in Wikipedia oder Produktbeschreibungen und Bewertungen bei Amazon. Durch Verknüpfungen entstehen Netzwerke, die der Schlüssel zu

Marktdominanz im Web 2.0 sind (vgl. ebd). Folglich sind die Daten der Faktor für den Wert einer Anwendung und eines Geschäftsmodells.

Die Verbreitung und der Erfolg von vielen Diensten liegen darin, dass sie einfache, aber mächtige Application Programming Interfaces (API) und Dienste zur Verfügung stellen. Dieses „Leightweighted Programming Models" (vgl. ebd.) genannte Prinzip führt zu neuen, innovativen Anwendungen durch Kombination verschiedener Dienste. Allerdings führt dies teilweise zu Anwendungen, deren Entwicklung sich ständig in der Betaphase befindet (vgl. ebd.). Dadurch können User auch als Mitentwickler betrachtet werden.

Kritik am Begriff „Web 2.0"

Der Begriff „Web 2.0" umfasst einige Prinzipien, die O'Reilly (2005) in seinem Artikel mit Beispielen belegt. Eine scharfe Abgrenzung zum „Web 1.0" als logischer Vorgänger ist nicht ohne Weiteres möglich. In Foren und Mailinglisten beispielsweise können User auch Inhalte erstellen und Kommentare austauschen – sie existieren aber wesentlich länger als der Begriff Web 2.0. Alby (2008:16) sieht in der „Dehnbarkeit des Begriffs" Grund für „heftige Diskussionen" (vgl. ebd.).

Letztlich sei, so Koops (vgl. 2008:25), Web 2.0 ein eine Modebegriff, der nichts weiter als die Nutzung bestimmter Techniken beschreibe. In seinen Augen könne ein Webentwickler auch ohne Kenntnis dieses Begriffs Web 2.0-Anwendungen entwickeln.

Trotz dieser Kritik ist der Begriff Web 2.0 wichtig für die vorliegende Arbeit, da dieser Begriff den für den Geomassenmarkt wichtigen Elemente – Webservices, aktive Teilnahme der Nutzer und die Verbreitung von innovativen Anwendungen – einen prägnanten Namen gibt.

2.3.1.2 Mashups

Mashups sind Anwendungen, die Daten unterschiedlicher Herkunft zu einer neuen Webanwendung „vermischen“ und kombinieren. Dabei können die Datenquellen unterschiedlicher Herkunft sein: eigene Daten können genauso eingebunden werden wie Daten, die von anderen Diensten, von Flickr, Google Maps und vielen anderen stammen.

Mashups sind eine der bekanntesten Ausprägungen des Web 2.0. Sie erfüllen viele der sieben von O’Reilly (vgl. 2005:o.S.) genannten Punkte, die seiner Meinung nach das Web 2.0 ausmachen.

- User können ihren selbst generierten Inhalt einbringen. Wird der eigene Inhalt beispielsweise als Text in einem Blog oder als Foto in Flickr veröffentlicht, so können andere User in Form von Kommentaren darauf antworten. Dies stellt eine Form der „Kollektiven Intelligenz“ (vgl. ebd.:o.S.) dar.
- „Fremder“ Inhalt ist oft die Grundlage eines Mashups. Populär wurden in den Anfangszeiten des Web 2.0 gerade Mashups auf Basis von Google Maps. Die Erstellung von Mashups ist jedoch nur möglich, da User „Lightweight Programming Models“ (vgl. ebd.:o.S.) in Form von Webdiensten und APIs nutzen können.

O’Reilly (vgl. ebd.:o.S.) nennt dieses Prinzip Neuerung durch Zusammenbau. In seinen Augen kann ein Mehrwert erzeugt werden, wenn gängige Komponenten ausreichend vorhanden sind und auf neuartige oder effektive Weise kombiniert werden (vgl. ebd.:o.S.).

2.3.1.3 Geoweb

Geoweb bzw. Geospatial Web ist ein zentraler Begriff im Umfeld von Geoinformatik und Internet. Maguire (vgl. 2008:1) sieht den Begriff jedoch als Platzhalter für Geodaten und Anwendungen, die über das Internet genutzt werden. Dies und die schnelle Entwicklung des Geoweb in den letzten Jahren habe sowohl in den Bereichen des

professionellen GIS als auch der Mainstream Nutzung neue und innovative Wege der Nutzung von geography geöffnet (vgl. ebd.:1).

Die Autoren von (Turner, A. & Forrest, B. 2008) präzisieren die Rolle des Internet im Geoweb als Netzwerk von entdeckbaren Dokumenten, Datenbanken und Diensten mit geographischen Inhalten (vgl. ebd.:2). Maguire (vgl. 2008:1) sieht nicht nur eine Veränderung in Umfang und Zahl von im Internet erreichbaren Karten, sondern auch in der vermehrten Bereitstellung der Daten durch Webservices, die auch von einer steigenden Zahl von „web geographers" (vgl. ebd.:1) genutzt werden können.

Die wachsende Zahl an Geodaten, verbunden mit den von Web 2.0-Usern erstellten Inhalten und mit der steigenden Verbreitung von GPS und breitbandigem Internetzugang auf mobilen Geräten bilden einen neuen Trend, „Where 2.0" genannt (Turner, A. & Forrest, B. 2008:2). Maguire (vgl. 2008) prägt hingegen den Begriff des Geoweb 2.0, der Eigenschaften des Web 2.0 auf das Geoweb überträgt, wodurch Maguire (vgl. ebd.:3ff) eine Geodateninfrastruktur erhält, die auf Standardprotokollen wie SOAP, XML oder OGC-Standards basiert, dynamische Clients für die Visualisierung nutzt und die Erstellung von Mashups ermöglicht. Somit stellt das Geoweb ein wesentliches Merkmal des Geomassenmarkts dar.

2.3.1.4 Geotagging

Der Begriff „Geotagging" setzt sich aus zwei Teilen zusammen: „Tagging" und „Geo". Beim Tagging werden einzelne Resourcen – Bilder, Blogeinträge – mit mindestens einem Schlagwort, den sogenannten Tags, versehen. Dieses Verfahren ähnelt den Anfängen von Yahoo!, in dessen Verzeichnis Websites von Menschen in Kategorien einsortiert wurden (vgl. Alby 2008:121). Beim Tagging hingegen werden von den Nutzern selber die Inhalte verschlagwortet (vgl. ebd:127).

Die Bilddatenbank und Fotocommunity „Flickr" (http://www.flickr.com) ermöglicht Usern, ihre hochgeladenen Fotos mit Tags zu versehen. Um die Fotos nicht nur nach deren Inhalten durchsuchen zu können, fingen Nutzer an, geographische Informationen

mit anzugeben, die Bilder zu verorten. Verwendet werden dabei die drei Tags geotagged, geo:lon und geo:lat.

Aufnahmen der Steinernen Brücke in Regensburg, die mit den Tags

```
geotagged geo:lon=12.0978 geo:lat=49.0218
```

verschlagwortet sind, werden in einer Karte (http://www.flickr.com/map) georeferenziert angezeigt.

Die Lageinformationen können nicht nur durch den User manuell hinzugefügt werden. Durch die Einführung von GPS-Kameras und Handys mit integrierter Kamera und GPS kann der Aufnahmestandort auch im Bild in den sogenannten EXIF-Daten gespeichert werden. EXIF-Daten (Exchangeable Image File Format) enthalten zusätzliche Informationen wie Belichtungsdauer, Blende oder Kameramodell. Diese Informationen werden beim Upload ausgelesen und zusammen mit dem Bild gespeichert (vgl. Flickr 2004:o.S.).

2.3.2 Technologien

Bei der Erläuterung des Konzepts des Web 2.0, bei der Einführung in Mashups und der Vorstellung des Geoweb wurde bereits der verstärkte Einsatz einiger Technologien angesprochen. Diese Technologien werden im Folgenden detailliert vorgestellt.

2.3.2.1 Web Services

Webservices sind Softwarekomponenten, die eine plattformübergreifende Kommunikation zwischen Computern ermöglichen sollen. Die Kommunikation erfolgt meist über das HTTP(S)-Protokoll. Ein Webservice stellt bestimmte Funktionalitäten zur Verfügung, die von anderen Computern (Clients) genutzt werden können.

Für die Kommunikation wird in der Regel Xml verwendet (vgl. W3C 2004). Dies wird jedoch nur bedingt eingehalten. Ein Web Map Service beispielsweise gibt als Antwort auf den GetMap-Request kein Xml, sondern ein Bild zurück, andere Webservices wiederum nutzen JSON u. ä. zur Ausgabe (siehe Kapitel 3.3.1 und 3.3.3).

Web Services sind die Basis für viele Funktionen, die das Web 2.0 und den Geomassenmarkt prägen. Daten, die an verteilten Standorten über Web Services bereitgestellt werden, können in Mashups eingebunden und so genutzt werden.

2.3.2.2 Ajax

Ajax, eine Abkürzung für „Asynchronous JavaScript and XML", ist keine neue Technologie, sondern besteht aus einigen Technologien, die auf eine effektive Art kombiniert werden (vgl. Garret 2005:o.S.).

Ajax basiert hauptsächlich auf dem XMLHttpRequest, einer JavaScript-Funktion, die im Hintergrund eine Anfrage an einen Webserver stellt, und die Antwort empfängt. Die zweite Technologie hinter Ajax ist die Änderung des DOM (Document Object Model) der HTML-Seite im Browser. Dadurch können Teile des bereits vom Browser gerenderten HTML-Dokuments verändert werden, ohne dass die gesamte Seite neu vom Webserver geladen werden muss (vgl. Gehtland et al. 2006:53ff). In Ajax werden beide Funktionen miteinander verknüpft, so dass es möglich wird, die Seite im Browser mit den empfangenen Antworten zu manipulieren.

Dieses Abrufen von Daten im Hintergrund wird als asynchrones Verhalten bezeichnet. Das unterschiedliche Verhalten traditioneller Webanwendungen und von Ajax-basierten Anwendungen kann in nachfolgender Abbildung (Abb. 1) aufgezeigt werden.

Bei der Nutzung klassischer Anwendungen wird stets die gesamte Seite neu geladen, was dazu führt, dass während der Datenübertragung der User die Anwendung nicht weiter nutzen kann. Werden die Daten asynchron übertragen, so erfolgt der Abruf und das Empfangen der Daten im Hintergrund, so dass der Nutzer in der Zwischenzeit mit seiner Webanwendung weiterarbeiten kann.

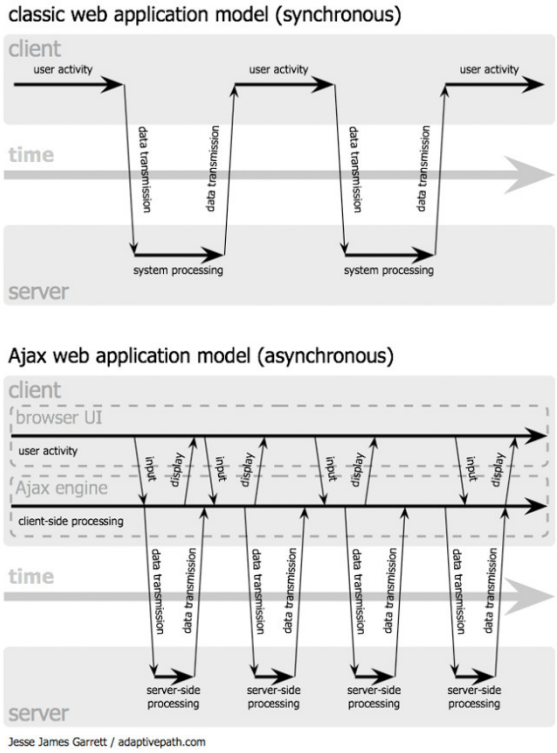


Abb. 1: Synchrones und asynchrones Verhalten von Webanwendungen (vgl. Garret 2005:o.S.).

Der Einsatz von Ajax ermöglicht dynamische Webanwendungen, wie am Beispiel von Google Suggest (http://www.google.de/webhp?complete=1) erläutert werden soll. Google Suggest schlägt dem User während der Eingabe des Suchbegriffs bereits Suchbegriffe vor, die mit den bereits eingegebenen Buchstaben beginnen (siehe Abb. 2). Ermöglicht wird dies durch Ajax: nach jedem eingegebenen Buchstaben wird ein Request an einen Webservice gestellt, der dann eine Liste mit Begriffen zurückgibt. Durch Manipulierung des DOM der Seite werden die Begriffe in einer Liste angezeigt.

Abb. 2: Screenshot Google Suggest (vgl. Google 2009:o.S.)

2.3.2.3 Tiling

Während im Bereich der Desktop-GI-Systeme und bei einigen webbasierten GIS-Viewern immer der gesamte sichtbare Kartenausschnitt als eine Grafik auf Anforderung gerendert werden, setzen neuere Mapping-APIs auf das sogenannte „Tiling“. Dabei wird die gesamte Karte nach einem vorgegebenen Schema in einzelne, quadratische Tiles (Kacheln) zerlegt. Der Client ruft die erforderlichen Tiles vom Server ab und setzt diese zu einer großen Karte zusammen.

Funktionsweise

Da meist nur ein Teil der maximalen Kartenausdehnung sichtbar ist, werden nur die Tiles geladen, die ganz oder teilweise im sichtbaren Bereich liegen. Um beim Verschieben der Karte nicht auf das Nachladen von benachbarten Kacheln warten zu müssen, werden zusätzlich die dem sichtbaren Bereich benachbarten Kacheln geladen. (vgl. Langfeld 2006:39ff, Kunze 2006:80f).

Tiling ermöglicht kein stufenloses Zoomen. Der Grund dafür liegt im Einsatz von Zoomstufen, die im Tilingschema definiert werden. Ausgehend von einer Zoomstufe, die die maximale Ausdehnung in einer Kachel abbildet, wird der Maßstab beim

Heranzoomen um eine Stufe halbiert; die Fläche, die zuvor ein Tile abgedeckt hat, wird nun von vier Tiles abgedeckt.

2.3.3 Mappingumgebungen

In Kapitel 2.1 wurde das Umfeld des Geomassenmarkts skizziert, das durch Elemente des Web 2.0 und einem verstärkten Bewusstsein für geographische Inhalte gekennzeichnet ist. Auf technischer Seite wird, wie in Kapitel 2.2 erarbeitet, der Geomassenmarkt durch den Einsatz von Webservices und Ajax geprägt. Dies führt zu neuartigen Möglichkeiten der Navigation in Karten.

Um den von „Neogeographern" erstellten Inhalt visualisieren zu können, werden Komponenten benötigt, die zum Einen eine Kartengrundlage zusammen mit Navigationsfunktionen bereitstellen und zum Anderen Funktionen bieten, um Geodaten hinzuzufügen. Grundsätzlich lassen sich alle Komponenten in zwei Gruppen, den sogenannte „Virtuelle Globen" und den „Webmapping-APIs", aufteilen.

2.3.3.1 Virtual Globes

Virtuelle Globen, auch als Earth Viewer, Earth Browser oder Geobrowser bezeichnet, bieten einen dreidimensionalen Blick auf die Erde. Sie bieten den Usern die Möglichkeit, die Erdoberfläche aus einem Höhenmodell mit überlagertem Satelliten- bzw. Luftbild bestehend zu erkunden. Dazu stehen ihm zusätzlich zum Zoomen und Pannen noch weitere Werkzeuge für die Navigation im Raum zur Verfügung. Dadurch kann ein User sowohl die Blickrichtung als auch den Blickwinkel ändern (vgl. Google 2009a:o.S.).

Neben dem bekanntesten Vertreter Google Earth stehen auch noch andere Virtual Globes zur Verfügung, z. B. NASA World Wind und Microsoft Virtual Earth. All diese Produkte können, zumindest in einer Grundversion, kostenlos verwendet werden. Google Earth hat von allen den umfassendsten Funktionsumfang (vgl. Öfele & Luderschmid 2006:o.S.); dort stehen den Benutzern auch 3D-Gebäudemodelle größerer

Städte zur Verfügung, auch können externe Daten, beispielsweise in Form von KML-Dateien, einbinden lassen [vgl. Höffken 2009:40f].

ESRI hat ebenfalls als kostenfreies Produkt den ArcGIS Explorer entwickelt, der sich nahtlos in die ArcGIS-Produktpalette einfügt. Nutzer können von ESRI bereitgestellte Basiskarten nutzen und zusätzliche Daten, die von einem ArcGIS Server bereitgestellt werden, einbinden (vgl. ESRI 2009c:o.S.). Mit Hilfe eines ArcGIS Server können auch räumliche Analysen durchgeführt werden (vgl. ESRI 2009d:o.S.).

2.3.3.2 Webmapping-APIs

Anders als Virtuelle Globen kommen Webmapping-APIs ohne Installation von Desktopanwendungen auf dem Client aus. Zur Ausführung einer Webmapping-Anwendung genügt meist ein Browser mit aktiviertem JavaScript. Bei einigen auf Adobe Flash bzw. Flex basierenden Anwendungen ist die Installation eines Browserplugins jedoch Voraussetzung. In dieser Arbeit sollen jedoch nur die JavaScript-basierten APIs betrachtet werden. Für auf Flash bzw. Flex basierenden APIs gelten vergleichbare Konzepte.

Google brachte im Februar 2005 (vgl. Ramsey 2008:o.S.) als erstes Unternehmen eine Webmapping-API auf den Markt, die auch abseits der GI-Branche Beachtung fand. Andere große Unternehmen wie Microsoft und Yahoo! zogen nach. Obwohl es eine Vielzahl an APIs gibt, unterscheiden sie sich, wie aus Abb. 3 ersichtlich, im Funktions- und Datenumfang nicht wesentlich (vgl. Öfele & Luderschmid 2006:o.S.).

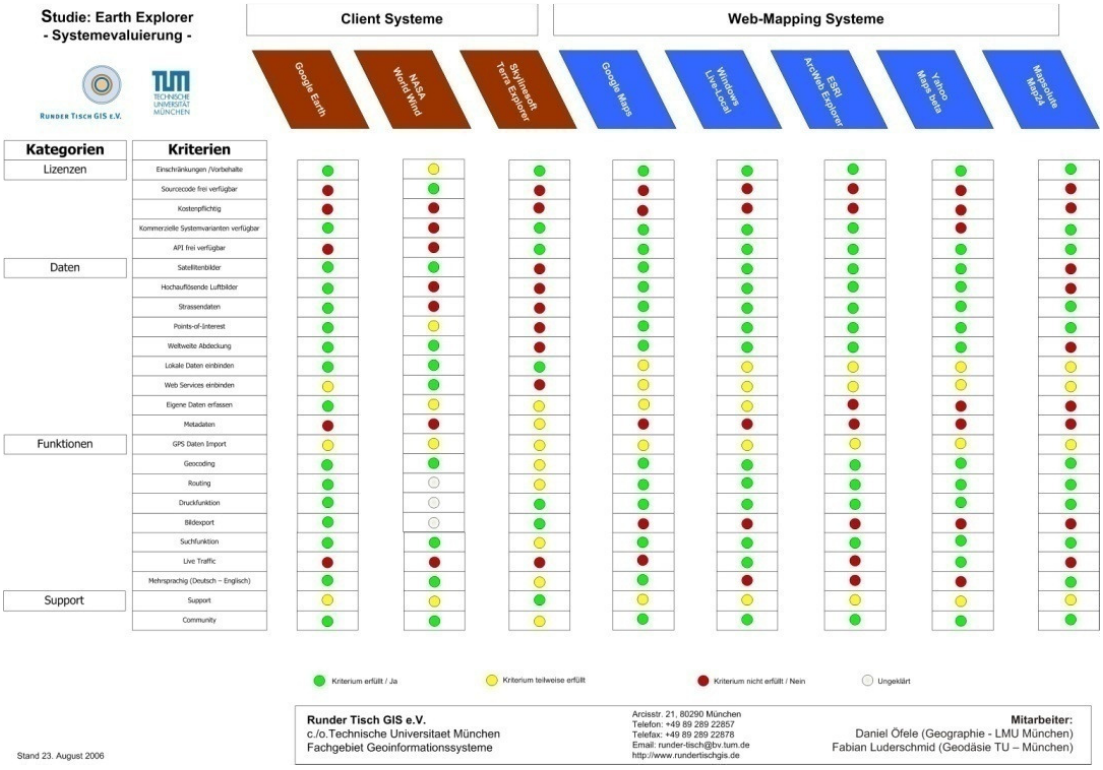


Abb. 3: Ergebnisse der Systemevaluierung verschiedener Earth Explorer (vgl. Öfele & Luderschmid 2006:o.S.)

Verfügbare Geodaten

Die meisten Webmapping-APIs bieten Karten, Satelliten- bzw. Orthophotos und Hybridansichten, die aus Satelliten- bzw. Orthophoto und einer überlagen Karte bestehen, an. Bing Maps hat zusätzlich als Alleinstellungsmerkmal Schrägansichten vieler größerer Städte.

Funktionsumfang

Einen Überblick über die Funktionalität bieten Öfele & Luderschmid (vgl. 2006:o.S.). Zusätzlich finden sich im Internet viele Beispiele, wie nicht von der API unterstützte Datenquellen eingebunden werden können. Die Einbindung eines WMS-Dienstes beispielsweise in Google Maps zeigt Mulka (vgl. 2005:o.S.).

OpenLayers

Aus der Zahl der Webmapping-APIs sei an dieser Stelle OpenLayers (http://www.openlayers.org) hervorgehoben. Diese unter der BSD lizensierte Bibliothek hat viele Funktionen, die sie von den Konkurrenten abhebt. OL ermöglicht die Einbindung verschiedener APIs wie Google Maps, Bing Maps etc., die Nutzung von Web Map Services, Lese- und Schreibzugriff auf Web Feature Services und vieles mehr.

2.3.4 Datenformate

Während in professionellen Geodateninfrastrukturen zum Datenaustausch in der Regel GML verwendet wird, greifen Anwendungsentwickler im Geomassenmarkt meistens auf andere Formate zurück.

2.3.4.1 GeoRSS

Mittels RSS (Really Simple Syndication) können Nachrichten, bestehend aus Titel, Beschreibung bzw. Anriss und einem Link zum vollständigen Artikel in einem XML-basierten Format ausgetauscht werden. Diese sogenannten Feeds beinhalten neben Name und Mailadresse des Autors Titel, Beschreibung bzw. Anriss der einzelnen Einträge, Veröffentlichungsdatum und einen Link zum vollständigen Artikel.

Ein exemplarischer RSS-Feed mit einem einzigen Eintrag schaut folgendermaßen aus:

```
<?xml version="1.0" encoding="utf-8"?>
<rss version="2.0">
  <channel>
    <title>Berichte meiner Reise</title>
    <link>reise.domain.de</link>
    <description>Kurzberichte und Erlebnisse meiner Reise</description>
    <language>de-de</language>
    <copyright>Autor</copyright>
    <pubDate></pubDate>
    <item>
      <title>Regensburg</title>
```

```
      <description>Meine Eindrücke aus der Welterbestadt
Regensburg</description>
      <link>http://reise.domain.de/2009/regensburg</link>
      <author>Autor, autor@domain.de</author>
      <pubDate>Tue, 28 Jul 2009 12:00:00 +0100</pubDate>
    </item>
  </channel>
</rss>
```

In dem Maße, wie RSS zur Veröffentlichung von Inhalten immer wichtiger wird, so wird es auch immer wichtiger, einen Ortsbezug zu beschreiben. Dies muss auf eine Art geschehen, die es Anwendungen ermöglicht, georeferenzierte Feeds anzufordern, zu veröffentlichen und zu visualisieren (vgl. GeoRSS 2009:o.S.).

Um Einträge eines RSS-Feeds zu verorten, kann der Autor des Feeds aus drei Möglichkeiten wählen:

W3C

Mit diesem älteren Standard lassen sich nur Punkte im Bezugssystem WGS84 darstellen. (Geodateninfrastruktur Deutschland 2009:o.S.)

GeoRSS Simple

Mit diesem Standard können grundlegende Geometrien (Punkt, Linie, Rechteck, Polygon (vgl. GeoRSS 2009:o.S.) abgebildet werden. Referenzsystem ist ebenfalls WGS84.

Dazu muss der GeoRSS-Namespace eingebunden werden:

```
<rss version="2.0" xmlns:georss="http://www.georss.org/georss/">
```

Der Eintrag des Feeds wird dann noch um das Geometrieelement ergänzt:

```
<item>
  <title>Regensburg</title>
  <description>Meine Eindrücke aus der Welterbestadt
Regensburg</description>
  <link>http://reise.domain.de/2009/regensburg</link>
  <author>Autor, autor@domain.de</author>
  <pubDate>Tue, 28 Jul 2009 12:00:00 +0100</pubDate>
  <georss:point>49.0218 12.0978</georss:point>
</item>
```

GeoRSS GML

Durch den Einsatz des XML-Applikationsschemas GML (Geographic Markup Language) können wesentlich komplexere Geometrieen abgebildet werden. Es liegt keine Beschränkung auf WGS84 als Referenzsystem vor.

Dazu muss zusätzlich der GML-Namespace eingebunden werden:

```
xmlns:gml="http://www.opengis.net/gml"
```

Der Eintrag des Feeds wird dann noch um das Geometrieelement ergänzt:

```
<item>
  <title>Regensburg</title>
  <description>Meine Eindrücke aus der Welterbestadt
Regensburg</description>
  <link>http://reise.domain.de/2009/regensburg</link>
  <author>Autor, autor@domain.de</author>
  <pubDate>Tue, 28 Jul 2009 12:00:00 +0100</pubDate>
  <georss:where>
    <gml:Point>
      <gml:pos>49.0218 12.0978</gml:pos>
    </gml:Point>
  </georss:where>
</item>
```

Das Referenzssystem in diesem Beispiel ist WGS84 und wird deswegen nicht angegeben.

Für die Darstellung in einer Karte ist die Nutzung einer Mapping-Komponente wie in Kapitel 2.3.3.2 beschrieben erforderlich.

2.3.4.2 GeoJSON

Ein weiteres Format zum Austausch geographischer Daten ist GeoJSON (http://geojson.org), das auf JSON (http://json.org) basiert (vgl. Howard et al. 2008:o.S.).

JSON (JavaScript Object Notation) (vgl. JSON o.J.:o.S.)

JSON basiert auf zwei Elementen: Objekte und Arrays.

- Objekte sind von geschweiften Klammern umschlossene, ungeordnete Listen, bestehend aus Schlüssel-Wert-Paaren (Key-Value-Pairs). Schlüssel und Wert werden durch einen Doppelpunkt voneinander getrennt.
- Arrays sind geordnete, von eckigen Klammern umschlossene Listen, deren Elemente mit Kommata voneinander getrennt werden.

Als mögliche Werte können verwendet werden:

- Objekt
- Array
- Daten vom Typ Number
- Daten vom Typ String (in Anführungszeichen)
- Literal: "true", "false", "null"

Als Beispiel sei hier eine Liste mit zwei Orten und deren Einwohnerzahl als JSON aufgeführt:

```
{
   "Anzahl":2,
   "Trefferliste":[
      {
         "Name":"Regensburg",
         "Einwohnerzahl":150000
      },
      {
         "Name":"München",
         "Einwohnerzahl":1200000
      }
   ]
}
```

GeoJSON

GeoJSON ist ein Format zur Codierung geographischer Daten (vgl. Howard 2008:o.S.).

Jedes GeoJSON-Objekt muss den Schlüssel „type“ enthalten, dessen Wert den Typ des Objekts angibt. Zur Verfügung stehen diverse Geometrietypen („Point“, „MultiPoint“, „LineString“, „MultiLineString“, „Polygon“ und „MultiPolygon“) sowie „GeometryCollection“, „Feature“, oder „FeatureCollection“.

Zusätzlich können Objekte auch einen Schlüssel „crs“ für die Angabe eines Referenzsystems und einen Schlüssel „bbox“ für die Angabe einer Boundingbox besitzen. (vgl. ebd.:o.S.)

Als Beispiel sei ein einfaches GeoJSON-Objekt mit einer Punktgeometrie aufgeführt:

```
{
   "type":"Feature",
   "Name":"Regensburg",
   "Einwohnerzahl":150000,
   "geometry":{
      "type":"Point",
      "coordinates":[
         102.587890625,
         -1.114990234375
      ]
   }
}
```

2.3.4.3 KML

KML (Keyhole Markup Language) wurde 2001 von der Firma Keyhole entwickelt (Wernecke 2009:2). Nach der Übernahme von Keyhole durch Google wurde am 04. April 2008 KML in der Version 2.2 als OGC-Standard angenommen (Open Geospatial Consortium Inc. 2008b:o.S.).

Die Präambel der KML-Spezifikationen (vgl. Open Geospatial Consortium Inc. 2008c:ii) enthält zwei heraushebenswerte Punkte:

- Der Schwerpunkt von KML liegt auf der Darstellung von geographischen Daten und Anmerkungen. Visualisierung beinhaltet nicht nur die Darstellung geographischer Daten, sondern auch Vorgaben zur Navigation und Betrachtungsweise von Objekten.
- Es wird in der Präambel vereinbart, dass sich KML sich an internationale Standards angleicht, um dadurch eine größere Verbreitung und Interoperabilität von Earth Browsern zu ermöglichen.

An beiden Punkten wird deutlich, dass KML ursprünglich kein OGC-Standard war. Der Schwerpunkt von KML der Visualisierung steht im Gegensatz zu GML, dessen Schwerpunkt auf einer effizienten Interoperabilität zwischen verschiedenen Systemen

liegt. Zur Visualisierung stehen andere Standards, beispielsweise SLD im Bereich der WMS-Dienste, zur Verfügung. Der zweite Aspekt, die Annäherung an internationale Standards, zeigt Defizite von KML hinsichtlich der Unterstützung von OGC-Standards auf. Nach Open Geospatial Consortium Inc. (2008c:4) implementiert KML die Spezifikationen des Simple Feature Access nicht.

Aufbau von KML (vgl. Wernecke 2009)

Jede KML-Datei beginnt mit dem Xml-Header, gefolgt von der Deklaration des KML-Namespaces „http://www.opengis.net/kml/2.2“:

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2">
```

Als Kindelemente des Knoten <kml> können nun alle Objekte hinzugefügt werden. Meist wird jedoch noch der Knoten <Document> eingefügt, um das KML-Dokument zu gliedern. Als Objekte können Placemarks, Overlays, Styles und andere Elemente verwendetet werden.

Placemark

Zentrales Element von KML ist das „Placemark“-Element, das ein einzelnes Objekt der Realität bestehend aus Lage, Name, Beschreibung u. a. repräsentiert.

```
<Placemark>
  <name>Regensburg</name>
  <description>Welterbe an der Donau mit Dom, Steinerner Brücke und
schöner Altstadt</description>
  <Point>
    <coordinates>49.0218,12.0978,0</coordinates>
  </Point>
</Placemark>
```

Im Folgenden werden zwei Elemente näher beschrieben, die für die Erstellung des Prototypen von Interesse sind. Dies sind die Elemente „ExtendedData“ und „Geometry“.

ExtendedData

Um zu einem Placemark-Element benutzerdefinierte Daten hinzuzufügen, wird das ExtendedData-Element verwendet. Dabei stehen nach Open Geospatial Consortium Inc. (vgl. 2008c:41ff) drei Varianten zur Verfügung:

- kml:Data-Element mit Schlüsselfeldern und Werten
- Typisierte Daten unter Verwendung eines kml:Schema-Elements
- Beliebiger Xml-basierter Inhalt, organisiert außerhalb des KML-Namespaces

Bei der Realisierung des WfsSimple-Prototypen wird die erste Variante verwendet, da dieses Verfahren einen einfachen, aber mächtigen Mechanismus (vgl. Wernecke 2009:246) darstellt. Die Schlüsselfeld/Wert-Paare werden in Google Earth standardmäßig in einem „Balloon“ als Tabelle angezeigt (vgl. ebd. 2009:249).

Der Aufbau ist wie folgt:

```
<Placemark>
  <name>Regensburg</name>
  <description>Welterbe an der Donau mit Dom, Steinerner Brücke und Altstadt</description>
  <Point>
    <coordinates>49.0218,12.0978,0</coordinates>
  </Point>
  <ExtendedData>
    <Data name="Einwohner"><value>150000</value></Data>
    <Data name="Hotelbetten"><value>4000</value></Data>
  </ExtendedData>
</Placemark>
```

Bei beiden anderen Varianten werden von Earth Browser die zusätzlichen Informationen (vgl. ebd.:246) ignoriert; sie eignen sich allerdings gut zum Datenaustausch (vgl. ebd.:246).

Geometry

Zur Abbildung von geometrischen Objekten stehen fünf Elemente zur Verfügung (vgl. Open Geospatial Consortium Inc 2008c):

- <Point>
- <LineString>
- <LinearRing>
- <Polygon>
- <MultiGeometry>

Im KML-Standard ist ebenfalls die Definition eines „Punktes“ festgelegt. Die Werte für longitude (Länge), latitude (Breite) und altitude (Höhe) werden durch Kommata getrennt als Koordinatentupel verwendet. Dies unterscheidet sich vom Well Known Text-Format (WKT) des Simple Feature Access - Standard des OGC (vgl. Open Geospatial Consortium Inc 2005b:29): dort werden die einzelnen Werte durch Leerzeichen getrennt. Im Detail werden die Geometrieen wie folgt gebildet (vgl. Open Geospatial Consortium Inc 2008c):

- <Point>

  Der Elternknoten <Point> beinhaltet das Child <coordinates>, das die Koordinaten des Punktobjekts enthält.
- <LineString>

  Der Elternknoten <LineString> beinhaltet das Child <coordinates>, das mindestens zwei Koordinaten der Linie enthält. Die einzelnen Koordinatentupel werden im Gegensatz zum WKT-Format durch Leerzeichen getrennt.
- <LinearRing>

  Ein LinearRing ist ein geschlossener Linienzug und enthält demzufolge mind. drei Koordinatentupel, wobei der Endpunkt gleich dem Startpunkt sein muss.
- <Polygon>

  Ein Polygon besteht grundsätzlich aus einer <OuterBoundary>, die ihrerseits einen <LinearRing> als Begrenzung des Polygons enthält. „Löcher“ im Polygon können als <InnerBoundary> mit untergeordnetem <LinearRing> wiedergegeben werden.
- <MultiGeometry>

  Als Multigeometry werden all die Objekte bezeichnet, die sich aus mehreren Geometrieen zusammensetzen. Diese Klasse umfasst also auch all die Objekte, die nach dem Simple Feature Access-Standard als MultiPoint usw. zu modellieren sind (vgl. ebd:76f).

  Ein <MultiGeometry>-Knoten setzt sich wiederum aus vollständigen Geometrieelementen zusammen.

Style

Mit Hilfe von Styles kann der Ersteller eines KML-Dokuments die Symbolisierung der enthaltenen Objekte festlegen. Für die verschiedenen Geometrieklassen stehen entsprechende Styles zur Verfügung – für Punkte der iconStyle, für PolyLines der LineStyle usw. (vgl. Wernecke 2009).

Overlay, GroudOverlay, NetworkLink, ViewPoint

KML bietet wesentlich mehr Möglichkeiten als die bisher beschriebenen. So können auch Rasterdaten als Overlays eingebunden werden, dynamische KML-Quellen als NetworkLink eingebunden und Ansichten von Objekten als ViewPoints definiert werden. Diese Möglichkeiten sind für den Entwurf und die Implementierung eines WfsSimple-Prototypen nicht von Bedeutung und werden deshalb nicht weiter beschrieben.

2.3.5 Datenquellen

Bisher wurden in diesem Kapitel technische Kennzeichen des Geomassenmarkts besprochen. In diesem Zusammenhang wurden Trends und Plattformen angesprochen, die als Datenlieferanten fungieren können.

2.3.5.1 Soziale Netzwerke

Eng verbunden mit dem Schlagwort „Web 2.0“ ist die Nutzung sogenannter „Sozialer Plattformen“ bzw. Netzwerke. Während bei einigen Portalen wie Facebook die Vernetzung der Mitglieder im Vordergrund steht, dienen andere der Bereitstellung von eigenem Inhalt. Enthält dieser Inhalt einen direkten oder indirekten Lagebezug, so lässt er sich unter Zuhilfenahme einer API wie Google Maps als Mashup präsentieren (Rousse 2007: 155f).

Aus der Vielzahl an Sozialen Netzwerken sollen nun zwei exemplarisch vorgestellt werden. Bei Panoramio und Flickr können Benutzer Bilder hochladen, in Sammlungen organisieren und mit Schlagworten versehen. Zusätzlich können andere User die Bilder kommentieren und sich in Gruppen mit bestimmten Interessen zusammenschließen. Grund für die Wahl der beiden Plattformen ist einmal der hohe Bekanntheitsgrad und vor Allem die Integration geographischer Daten.

Panoramio

Panoramio (http://www.panoramio.com), eine zu Google gehörende Plattform, unterstreicht die Bedeutung der Verortung der Bilder bereits auf ihrer Startseite: neben einem Mashup aus Google Maps mit bei Panoramio gespeicherten Bilder ergeht an die Nutzer die Aufforderung: „Share your favourite places". Diese Bilder können auch als Overlay bei Google Maps (http://maps.google.com) direkt hinzugeschaltet werden. Eine Verwendung in eigenen Mashups ist durch die Verwendung der API *(http://* www.panoramio.com/api) möglich.

Flickr

Flickr ist ebenfalls eine Plattform zum Veröffentlichen von eigenen Photos. Die Bedeutung des Aufnahmeorts wird deutlich durch die Prägung des Begriffs „Geotagging" (siehe Kapitel 3.2.).

Der Zugriff auf die eingestellten Inhalte erfolgt nicht nur über die Website von Flickr, auch über die API (http://www.flickr.com/services/api) kann auf sie zugegriffen werden. Dazu ist es notwendig, sich bei Yahoo, der Eigentümerin von Flickr, und bei Flickr selber zu registrieren und zusätzlich den Flickr API Key anzufordern (vgl. Flickr 2009:o.S.). Unter (http://www.flickr.com/services/api) befindet sich die gesamte Dokumentation zur API.

Um eine Suchanfrage an Flickr zu richten, genügt es, einen REST-basierten Request (GET oder POST) an den Endpunkt http://api.flickr.com/services/rest/ zu senden. Eine räumliche Suchanfrage nach Photos in der Umgebung Regensburgs schaut beispielsweise wie folgt aus:

```
http://api.flickr.com/services/rest/?method=flickr.photos.search&
bbox= 48.5,11.5,49.5,12.5&api_key=sdfgsgs
```

An den Endpunkt werden die für die Suche erforderliche und optionalen Parameter angehängt. Im Beispiel sind dies:

- Der Parameter method mit dem Wert flickr.photos.search für die Suche nach Photos
- Der API-Key für die Protokollierung der Nutzung der API (vgl. ebd.:o.S.)
- Der Parameter bbox zur räumlichen Eingrenzung der Suche

Als Ergebnis wird ein Dokument (standardmäßig als XML) zurückgegeben, das diverse Informationen zu den gefundenen Fotos, aber keinen direkten Links, enthält. Aus diesen Informationen kann dann der Link zum Foto oder dem Photostream des Users erstellt werden.

2.3.5.2 „Kartensammlungen“

Das Web 2.0 und das Geoweb 2.0 leben von der Beteiligung der User. Diese können nicht nur in Blogs schreiben und Photos austauschen, sondern auch eigene Karten erstellen und mit anderen austauschen.

Google Maps Verzeichnis

Google Maps fordert die Nutzer auf: „Kartografieren Sie Ihnen bekannte Orte und Routen. Fügen Sie Text, Fotos oder Videoaufnahmen hinzu und geben Sie die Ergebnisse frei“ (Google 2009d:o.S.). Freigegebene Karten aller User können im „Google Maps Verzeichnis“ durchforstet und zu Google Maps hinzugefügt werden (Google 2009b:o.S.).

Google Earth Community

Die Google Earth Community (bbs.keyhole.com) besteht aus einer Vielzahl von moderierten, nach Themen getrennten Foren, in denen User geographische Informationen aller Art als Kmz-Datei posten können. Auch nicht registrierte Nutzer können die KMZ-Dateien herunterladen bzw. sich in Google Earth anzeigen lassen.

ESRI ArcGIS Online

Auch ESRI bietet mit der Portal ArcGIS Online (http://www.esri.com/arcgisonline) eine Plattform zur Erstellung und zur Veröffentlichung von Karten. Die Besonderheit dieses noch in der Beta-Phase stehenden Projektes ist, dass der Workflow Erstellung – Veröffentlichung – Einbindung durchgehend auf Produkten von ESRI basieren (vgl. ESRI 2009a:o.S.). Zur Erstellung der Karten ist die Nutzung von ArcGIS Desktop Voraussetzung. Die Veröffentlichung erfolgt bei ESRI als Service auf der Basis von ArcGIS Server (AGS) und der JavaScript API (vgl. ebd.:o.S.). Der Ersteller hat dabei die Möglichkeit, eine Karte allen zugänglich zu machen oder über eine Gruppenverwaltung nur eingeschränkten Benutzerkreisen bereitzustellen (ESRI 2009b:o.S.).

2.3.5.3 Geographische Namen und Geocoding

Liegt nur ein indirekter Raumbezug eines Objekts vor, d. h nur der Name oder die Adresse, kann dieser in einen direkten Raumbezug, eine Koordinate, umgesetzt werden. Dazu kann auf verschiedene Dienste zurückgegriffen werden.

Geocoding

Ein hausnummerngenaues Geocoding bieten die meisten Anbieter von Mapping-APIs wie Google Maps API, Yahoo! Maps API und Microsoft Bing an.

Google als einer der Anbieter beispielsweise ermöglicht zwei Arten des Zugriffs an: einen direkten Zugriff über einen http-Request auf die Server oder über die in der API enthaltenen JavaScript-Funktionen (vgl. Majewski 2006:o.S.).

GeoNames

GeoNames (http://www.geonames.org) hingegen ist ein Gazetteer-Dienst, der nach eigenen Angaben über acht Millionen geographische Namen, unterteilt in neun Featureklassen, kennt (vgl. OpenGeoNames o.J.a:o.S.). Um auch die Art des geographischen Namens (Ortschaft, administrative Einheit, Fluss, Straße usw.) zu spezifizieren, werden insgesamt 645 sogenannter FeatureCodes verwendet (vgl. ebd:o.S.).

Interessant an GeoNames sind vor Allem die Lizenzbedingungen: die gesamte Datenbank ist unter der Creative Commons Attribution 3.0 License lizensiert (vgl. ebd:o.S.); sie ermöglicht unter der Nennung des Rechteinhabers die Weiterverbreitung, Veröffentlichung und Bearbeitung des Werkes (vgl. Creative Commons o.J.:o.S.).

Die Nutzung von GeoNames kann auf drei Arten erfolgen: zum Einen kann der Nutzer über ein Suchformular die Datenbank abfragen, und zum Zweiten die Datenbank über

Webservices abfragen. Die Ergebnisse liegen meist als Xml- oder JSON-Dateien vor; eine Übersicht über die möglichen Webservices und Ausgabeformate bietet (vgl. OpenGeoNames o.J.b:o.S.)

Da GeoNames die Datenbanken nach Ländern getrennt zum Download anbietet, können User mit diesen Daten auch darüber hinaus eigene Dienste und Anwendungen aufbauen.

2.3.5.4 „Geocoding by IP Adress“

Eine dritte Art des Geocoding ist das sogenannte Geocoding by IP Adress, auch bekannt als Geotargeting. Darunter versteht man die Technik, über die öffentliche IP-Adresse eines Internetusers dessen Standort zu bestimmen (Turner 2004:o.S.).

Die Einsatzmöglichkeiten sind vielfältig:

- Benutzern werden Webseiten in der ihres Standorts angezeigt.
- Im Geomarketing kann Benutzern standortbezogene Werbung angezeigt werden.
- CinemaNow, eine Onlinevideothek, kann über IP-bezogene Standortbestimmung die „Verwertung digitaler Inhalte an territoriale Grenzen […] binden“ (Höhnke o.J.:o.S.).

2.3.5.5 Sonstige Daten

Neben den bisher erwähnten Datenquellen existieren eine Vielzahl weiterer Quellen von Geodaten. Im OpenStreetMap-Projekt (http://www.openstreetmap.org) bereitgestellte Informationen wie POI können ebenso als Quelle verwendet werden wie Daten über die letzten Erdbeben. Diese werden vom USGS (U.S. Geological Survey) (http://earthquake.usgs.gov) bereitgestellt und enthalten Zeitpunkt, Stärke und Ort des Bebens. Auch die Wasser- und Schifffahrtsverwaltung stellt Informationen zu aktuellen Pegelständen und den Koordinaten der Messstellen zur Verfügung (http://www.pegelonline.wsv.de/gast/pegelinformationen).

Kapitel 3

Nicht OpenGIS-konforme Bereitstellung von Vektordaten im Geomassenmarkt

In Kapitel 2 der Arbeit wurde gezeigt, aus welchen Quellen räumliche Daten stammen können. Allen Quellen ist gemeinsam, dass die Geodaten auf nicht OpenGIS-konforme Art und Weise bereitgestellt werden. In diesem Kapitel sollen nun verschiedene Möglichkeiten erläutert werden.

In den Kapiteln 3.1, 3.2 und 3.3 werden verschiedene Formen des Zugriffs auf Vektordaten dargestellt und Anhand von Webanwendungen beschrieben. Abschließend erfolgt in Kapitel 3.4 eine Analyse der Anforderungen des Geomassenmarkts an einen Standard zur Bereitstellung Vektordaten.

3.1 Dateibasierter Zugriff

Soll auf statische Daten zugegriffen werden, so bietet sich zunächst eine Bereitstellung der Daten in Dateiform an.

Erstellung der Daten

Liegen die Daten bereits in digitaler Form vor, so können diese mittels eines Konverters in eines der unter 3.4 aufgeführten Formate umgewandelt werden. Für die verschiedensten Ausgangsformate existieren mittlerweile eigene Konverter. Aus Excel-Dokumenten können mit den Tools KaMeLwriter (http://www.kamelwriter.com) oder ExcelToKml (http://www.earthpoint.us/ExcelToKml.aspx) beispielsweise KML-Dateien erstellt werden. Die Liste von GI-Software, die einen Export als KML oder geoRSS ermöglicht, ist lang – ArcGIS, das einen KML-Export ab der Version 9.2 unterstützt, sei als Vertreter der kommerziellen Produkte genannt.

Auch im Bereich der Open Source Software gibt es einige Möglichkeiten zur Erzeugung von KML-Dateien.

Desktop-GIS-Anwendungen bieten die Möglichkeit, Layer als KML zu exportieren. In gvSIG (http://www.gvsig.gva.es) beispielsweise ist eine solche Exportfunktion bereits enthalten, in anderer GIS-Software kann diese Funktionalität als Plugin hinzugeladen werden.

Auch über die Ausführung von Kommandozeilenprogrammen ist die Erstellung von KML-Dateien möglich. Mit der OGR Simple Feature Library (http://www.gdal.org/ogr) können knapp 40 verschiedene Vektorformate und Datenquelle gelesen und teilweise auch geschrieben werden (vgl. GDAL o.J.:o.S.). Mit dem Tool ogr2ogr lassen sich einfache Formatumwandlungen durchführen.

Mit dem Befehl

```
ogr2ogr -f KML output.kml input.shp
```

wird aus dem Shapefile input.shp eine KML-Datei output.kml erstellt.

Auch ein manuelles Erzeugen bzw. Editieren von Xml-basierten Daten ist mittels Editoren möglich. Der kmleditor (http://www.northgates.ca/kmleditor) von NorthGate beispielsweise erleichtert die Eingabe und Validierung der Daten erheblich.

Bereitstellung der Daten

Die Bereitstellung der erzeugten Daten kann auf einem Laufwerk eines PCs bzw. Servers erfolgen. Im Regelfall werden die Vektordaten jedoch auf einem Webserver veröffentlicht, wodurch Zugriff und Nutzung über das Internet möglich werden. Die Veröffentlichung kann auch in Foren und Plattformen wie der Google Earth Community (siehe Kapitel 2.3.5.2) erfolgen.

Einbinden der Vektordaten

Wie die Vektordaten letztlich in eine Karte eingebunden werden, ist von der benutzten API bzw. der jeweiligen Software abhängig.

Sollen dateibasierte Vektordaten über eine API eingebunden werden, so wird im Quelltext der Anwendung die Datei fest als eigener Layer hinzugefügt. Die jeweilige Syntax ist dabei von API zu API unterschiedlich. In OpenLayers wird mit Hilfe der JavaScript-Funktion

```
map.addLayer(new OpenLayers.Layer.GML("KML", "lines.kml",
  { format: OpenLayers.Format.KML,
    formatOptions: {extractStyles: true}
  };
```

die statische KML-Datei lines.kml als neuer Layer zum bereits instanziiertem Objekt map hinzugefügt. Dabei spielt es keine Rolle, ob die Datei auf einem Laufwerk oder einem Webserver gespeichert ist.

Beim Einsatz von Desktopanwendungen wie Google Earth und ESRI ArcGIS Explorer können KML- und andere Dateien geöffnet und als Layer hinzugefügt werden (vgl. ESRI 2009c:o.S.).

Beispielanwendung: Fahrradverleih nextbike

Über das Fahrradverleihsystem nextbike (http://www.nextbike.de) können sich Kunden ein Leihrad mittels Handy in einigen Städten Deutsch ausleihen. Dies kann, ebenso wie die Rückgabe, nur an vorgegebenen Orten geschehen (vgl. nextbike o.J.a:o.S.). Auf der Homepage der Firma gibt es eine Standortübersicht, die für jede Stadt eine einzelne KML-Datei mit den Ausleihorten bereitstellt (vgl. nextbike o.J.b:o.S.). Potentielle Kunden können sich die KML-Dateien in einem Earth Browser anzeigen lassen. Eine Gemeinde, in der nextbike vertreten ist, könnte so in einem Informationssystem für Bürger und Touristen die Verleihstandorte einbinden.

Diese KML-Dateien enthalten nur Informationen über die Lage der Standorte; dynamische Informationen wie die Zahl der aktuell verfügbaren Räder sind auf Grund der statischen Bereitstellung nicht möglich.

Beispielanwendung: interaktiver Ortsplan Pfronten

Auf dem interaktiven Ortsplan der stark vom Tourismus geprägten Gemeinde Pfronten im Allgäu werden Gastgeber – Hotels, Ferienwohnungen, Bauernhöfe usw. – dargestellt (vgl. Pfronten o.J.a:o.S.). Über einen Link „In Google Earth öffnen“ kann eine KML-Datei geöffnet werden, die weitere statische KML-Dateien über mehrere „NetworkLink“ einbindet (vgl. Pfronten o.J.b:o.S.):

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<kml xmlns="http://earth.google.com/kml/2.2">
  <Folder>
    <name>Gastgeber Pfronten</name>
    <visibility>1</visibility>
    <open>1</open>
```

```
    <description>Gastgeber aus Pfronten gruppiert anhand ihrer
Betriebskategorie</description>
    <NetworkLink>
      <name>Hotel</name>
      <Link>
        <href>http://www.pfronten.de/se_data/geodata/hotel.kml</href>
      </Link>
    </NetworkLink>
```

Fazit

Um dateibasiert Vektordaten bereitzustellen, wird einfachste, standardisierte Technik benötigt. Von Vorteil ist vor Allem, dass auf die Nutzung einer Datenbank und serverseitiger Skripte unter asp.net, php u.ä. als Datenbankfrontend verzichtet werden kann.

Sollen Daten unkompliziert ausgetauscht werden, ist die Nutzung von Dateien unverzichtbar. Deutlich wird dies auch durch den regen Austausch von Placemarks in Communities wie der Google Earth Community.

Ändern sich die Daten jedoch häufig, so ist eine Bereitstellung als Datei nicht die erste Wahl, da der Aufwand zur Erstellung der Daten hoch sein kann.

3.2 Zugriff über nicht standardisierte Dienste

Werden Geodaten in Datenbanken vorgehalten, so werden die Vektordaten oftmals mittels eigener, an die einzelne Anwendung angepasster Dienste bereitgestellt. Diese Dienste sind als serverseitige Skripte in php, asp.net, Python, Ruby oder ähnlichem implementiert. Sie empfangen die Anfragen des Clients, erstellen die Anfragen an eine Datenbank und bereiten das Ergebnis der Abfrage in einem definierten Format auf. Dieses Ergebnis wird nun zurück an den Client gesendet und dort visualisiert.

Beispiel: Tourismusportal

Die Firma Alpstein Tourismus (http://www.alpstein.de) erstellt für Tourismusregionen interaktive Karten, die touristische Informationen enthalten. Für die Alpenregion Tegernsee Schliersee (http://www.alpenregion-tegernsee-schliersee.de) wurde beispielsweise eine solche interaktive Karte (http://www.outdooractive.com/live/Alpregio-Regionskarte/Tegernsee-Schliersee) erstellt. Dazu wird eine Grundkarte mit Informationen über Aktivitäten wie Wandern, Radwandern und Langlaufen angereichert, sowie Informationen zu Unterkunft und Verpflegung und zu vielen weiteren touristischen Themen.

Die Informationen kann der Nutzer aus Kategorien auswählen; im Hintergrund ruft die Anwendung die Vektordaten von einem eigenen Dienst ab. Für das Thema „Geocaching“ geschieht dies über folgenden Request:

```
http://www.alpserver.de/1.87/map/cluster.php?application=sfb&ACTIVEITEMS=&
LANG=de&SOURCEID=ic_60&SOURCE=tegsee&HEIGHT=594&WIDTH=1488&BBOX=1261006,60
42319,1374745,6087723&LEVEL=11&CRS=EPSG:900913&VERSION=1.2
```

Der Parameter, der den gewünschten Layer „Geocaching“ repräsentiert, wird in dieser Lösung SOURCEID genannt und besitzt hier den Wert ic_60.

Beispiel: Fahrradverleih nextbike

nextbike bietet auf seiner Homepage auch Informationen über die verfügbare Zahl der Räder an, die in einer Karte dargestellt werden (vgl. nextbike o.J.c:o.S.).

Die Informationen werden von einem php-Skript (http://nextbike.de/m/maps.php) dynamisch erzeugt und als XML-Datei an den Browser gesendet. Eine Einschränkung der Suchkriterien ist nicht ersichtlich.

Beispielanwendung: interaktiver Ortsplan Pfronten

Der interaktive Ortsplan von Pfronten bietet den Nutzern auch die Möglichkeit, einzelne Kategorien von Gastgebern aus- bzw. abzuwählen. Dabei werden nicht die in 4.1 genannten statischen KML-Dateien gefiltert, sondern für jede Kategorie ein einzelner Request an ein Perl-Skript gesendet. Als Filterkriterium wird dabei der Parameter type mit der Art der Unterkunft als Wert übergeben. Die Abfrage der Kategorie Ferienwohnung wird durch den Request

```
http://www.pfronten.de/cgi-
bin/siteengine.pl?GeoData::Extern::MarkerData&type=betrieb--fewo,feha,fedo
```

umgesetzt. Als Antwort wird, ähnlich wie bei nextbike, ein proprietäres XML-Dokument zurückgesendet.

Möchte der User genauere Informationen zu einer Ferienwohnung, so klickt er auf das Icon; es erfolgt ein weiterer Request an den Server, der die Informationen als HTML-Fragment zurückgibt. Diese werden dann in einem Balloon in der Karte visualisiert.

Eine Einbindung der Gastgeberinformationen in ein weiteres Informationssystem bedarf der genauen Analyse des Verhaltens der Webanwendung und der zurückgegebenen Inhalte.

Fazit

Gerade Datenbestände im Bereich des Tourismus können auch für andere Webmapping-Anwendungen interessant sein. So könnten die in einer Region liegenden Gemeinden die Informationen auch in ihre Ortspläne einbinden.

Wie an den obigen Beispielen deutlich wird, ist eine Einbindung in Fremdsysteme nur dann möglich, wenn die Bedeutung der Parameter, deren möglicher Werte und der Aufbau der Antworten bekannt sind. Der Autor der Webanwendung muss dann seine Anwendung so erstellen, dass die Requests zur Anforderung der Vektordaten mit den Spezifikationen dieser nicht standardisierten Dienste konform sind.

3.3 Zugriff über Standardsoftware

Der Trend, im Geomassenmarkt Vektorformate wie KML verstärkt einzusetzen, führt zwangsläufig zu der Entwicklung von Softwarepaketen, die eine einfache Bereitstellung dynamischer Daten ermöglichen. Auch aus diesem Bereich existiert eine Vielzahl an verschiedenen Lösungen. Aus dem Bereich der kommerziellen Software und proprietären Lösungen soll stellvertretend der ArcGIS Server von ESRI vorgestellt werden. Als Vertreter der Open Source Software werden die Anwendungen KmlMapServer und FeatureServer vorgestellt.

3.3.1 ArcGIS Server

Mit Erscheinen der Version 9.2 unterstützt ESRI in seinen Produktpaletten ArcGIS Desktop und ArcGIS Server (AGS) die Erstellung von KML-Dateien. Während mit der Desktop-Variante statische KML-Dateien erzeugt werden können, ist es möglich, im AGS dynamische KML-Dienste zu veröffentlichen.

Bei der Verwendung des AGS gibt es mehrere mögliche Anfragen, die KML und JSON als Ausgabeformate unterstützen. Je nach Anfrage werden dabei reine Vektordaten oder in Vektordaten mit Verweisen auf Rastergrafiken zurückgegeben (vgl. ESRI 2009e:o.S.).

3.3.2 KmlMapServer

KmlMapServer (http://www.itopen.it/soluzioni/kml-map-server) ist ein unter der GNU Version 3 lizensiertes Produkt der italienischen Firma ItOpen (http://www.itopen.it), die sich auf die Durchführung von WebGIS-Projekten mit OpenSource-Software spezialisiert hat (vgl. ItOpen 2009a:o.S.). Diese Software erweitert den UMN MapServer um eine KML-Ausgabe und wurde mit dem Ziel entworfen, dass, wann immer möglich, die Parameter dieselbe Bedeutung besitzen wie in OGC WFS-Diensten (vgl. ItOpen 2009b:o.S.). Da der KmlMapServer auf dem UMN MapServer basiert, bietet der KmlMapServer ebenfalls einen reinen Lesezugriff an.

Funktionalität

Der KMLMapServer besteht aus zwei Komponenten: dem layer server und dem icon server. Der layer server übernimmt die Bereitstellung der Daten als KML bzw. auch komprimiert als Kmz, der icon server liefert die zu den Punkten gehörenden Icons (vgl. ItOpen 2009b:o.S.).

Aufbau eines Requests

Eine Anfrage an den Webserver kann sich aus den folgenden Parametern zusammensetzen (vgl. ItOpen 2009b:o.S.):

- Request
  Gibt Ausgabeformat an – KML, KMZ oder icon
- Map
  Pfad zum mapfile
- Typename
  Auszugebende Layer
- Filter
  Filter, der den „OpenGIS Filter Encoding Implementation Specification" entsprechen muss
- Encoding
  Zeichensatzcodierung
- Bbox
  Boundingbox als räumlicher Filter

Lediglich der Parameter „map" wird zwingend benötigt; dieser UMN MapServer-typische Parameter gibt den Pfad zum sogenannten mapfile an. Darin werden die einzelnen Layer, deren Datenquellen, die Symbolisierung und allgemeine Diensteinstellungen konfiguriert.

Ein beispielhafter Request kann demnach so ausschauen:

```
http://www.myserver.com/kmlservice.php?map=/maps/mapfile.map&
typename=roads,rivers&filter=&encoding=&bbox=-180,-90,180,90
```

3.3.3 FeatureServer

Wie auch OpenLayers stammt der FeatureServer (http://www.featureserver.org) ebenfalls aus dem Hause MetaCarta (vgl. MetaCarta 2008:o.S.). Mit diesem in Python implementierten Server ist es möglich, Vektordaten als GeoJSON, GeoRSS und KML abzufragen und auch zu editieren (vgl. ebd:o.S.). Unter Einbindung des WPServer (http://code.google.com/p/webprocessingserver) können auch Geoprocessing-Funktionen genutzt werden (vgl. Schmidt 2008:o.S.).

Im Gegensatz zum KmlMapServer ist der FeatureServer ein REST-konformer Webservice (Representational State Transfer). REST ermöglicht es, mittels im http-Protokoll definierter Operationen (`GET, POST, PUT` und `DELETE`) Aktionen am Server auszulösen (vgl. Bayer 2002:3). Eine Einführung in REST Web Services bietet Bayer (2002).

Anfragen an RESTfull Webservices unterscheiden sich wesentlich von Requests nach dem CGI-Standard. An die URL angehängte Parameter werden weitestgehend vermieden (vgl. Bayer 2002:3ff). Um das Feature mit der ID 1 aus dem Layer „layer“ als KML zu erhalten, wird folgender Aufruf verwendet (vgl. MetaCarta 2008:o.S.):

```
GET http://example.com/featureserver.cgi/layer/1.kml
```

Analog dazu wird mit

```
DELETE http://example.com/featureserver.cgi/layer/1.kml
```

dieses Feature gelöscht (vgl. MetaCarta 2008:o.S.). Im Gegensatz zu SOAP-basierten Requests wird hier kein Parameter mit dem Wert „löschen“ an die URL des Dienstes angehängt.

Requests

Um Anfragen einschränken zu können, muss dennoch auf Parameter zurückgegriffen werden. Dazu stehen folgende Parameter zur Verfügung (vgl. MetaCarta 2008:o.S.):

- bbox
  Boundingbox als räumlicher Filter
- maxfeatures
  Maximale Anzahl der zurückgegebenen Features
- queryable
  Aufzählung der zu filternden Spalten

3.4 Analyse der Anforderungen

In den Kapiteln 4.2 und 4.3 wurde gezeigt, wie der Zugriff auf Geodaten über nicht standardisierte Dienste sowie über Standardsoftware erfolgen kann. Dabei wurden bereits einige Lösungen exemplarisch aufgeführt und analysiert. Um daraus allgemeingültige Aussagen treffen zu können, muss eine größere Zahl an Anwendungen untersucht und nach einheitlichen Kriterien analysiert werden.

3.4.1 Ziel

Ziel der Analyse ist es, an Hand von Anwendungen, die Vektordaten über nicht standardisierte Dienste nutzen, Anforderungen des Geomassenmarkts an einen Standard zur Bereitstellung von Vektordaten abzuleiten.

3.4.2 Aufstellung der Kriterien

Zu Beginn der Analyse werden Kriterien festgelegt, auf welche die Anwendungen hin untersucht werden. Für die Analyse sollen folgende Daten erhoben werden:

Name der Anwendung, URL, URL des Dienstes

Name der Anwendung, URL der Anwendung und untersuchte URL des Vektordaten bereitstellenden „Dienstes“

HTTP-Request-Methode

Methode des HTTP-Requests – GET oder POST

Architektur / Protokoll

REST, SOAP, CGI

Zugriff

Unterscheidung in Nur Lesezugriff (r) oder Lese- / Schreibzugriff (r/w)

Ausgabeformate

Aufzählung möglicher Ausgabeformate

Räumliche Filterung

Angabe, ob räumliche Filterung möglich ist

Mögliche Werte: ja oder nein

Referenzsystem

Angabe eines des verwendeten Referenzsystems. Ist kein Parameter zur Angabe des Referenzsystems im Request vorhanden, aus den Vektordaten jedoch das Referenzsystem ersichtlich, so wird dies vermerkt („kein Parameter“).

Attributive Filterung

Angabe, ob Filterung nach Attributen möglich ist

Mögliche Werte: ja oder nein

Sonstige Parameter

Angabe sonstiger Parameter

3.4.3 Durchführung der Analyse

Bei der Mehrzahl der untersuchten Anwendungen liegt keine öffentliche Dokumentation über den Datenzugriff vor. Jedoch ist es möglich, die Webanwendung in Hinblick auf entsprechende Requests zu untersuchen.

Für viele gängige Webbrowser existieren Add-Ons, die die Analyse der gesendeten Requests, deren Antworten und vieles mehr ermöglichen. Das bei der Analyse eingesetzte Tool Firebug (https://addons.mozilla.org/addon/1843) ist ein Add-On für den Browser Firefox (http://www.mozilla.com/en-US/firefox/personal.html). Wird bei aktiviertem Tool eine Anfrage an einen Webserver gesendet, so wird der Request registriert, werden die Parameter der Anfrage, die Antwort des Servers und die Antwortzeiten und -größen angezeigt und Information aus dem Header ermittelt.

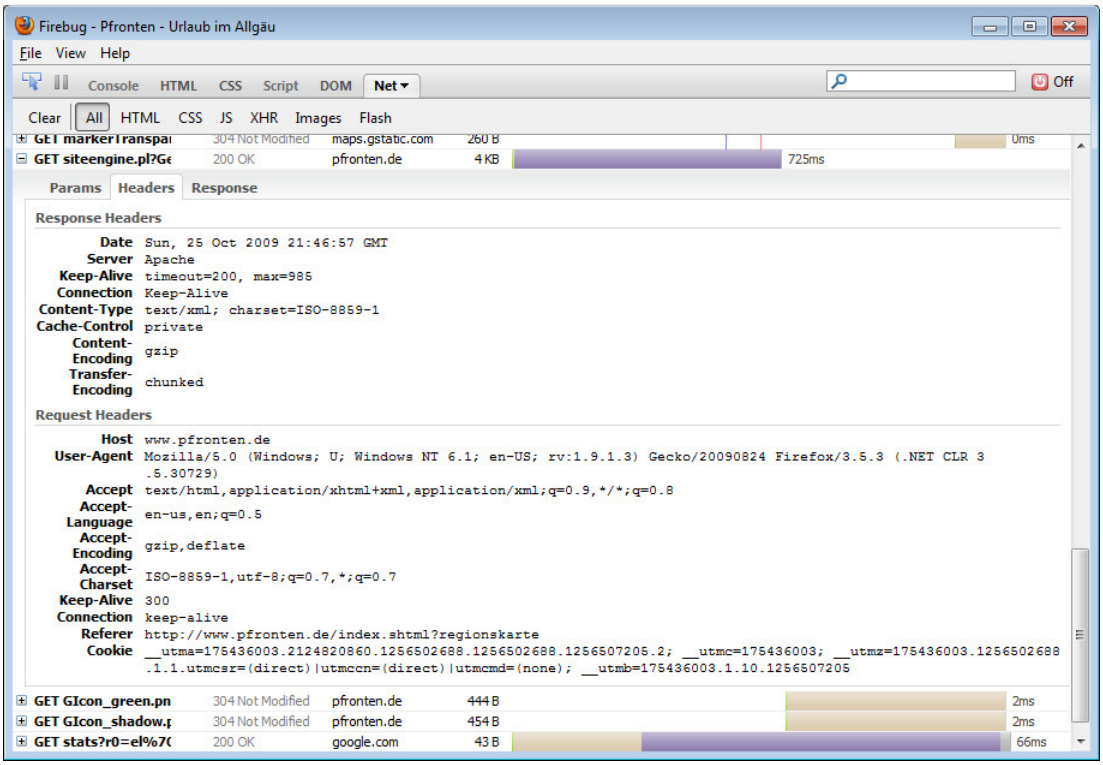


Abb. 4: Analyse einer Website mit Hilfe des Tools „Firebug“

Probleme bei der Untersuchung

Diese Form der Analyse bringt zwei Nachteile mit sich. Es können nur die vom Browser an den Webserver gesendeten Requests angezeigt werden. Dadurch können nur die Parameter, die beim analysierten Aufruf verwendet werden, in die Analyse eingehen. Andere Parameter, die eine Anwendung möglicherweise zusätzlich unterstützt, können so nicht erfasst werden.

Bei der Durchführung der Analyse ergab sich ein weiteres Problem: lediglich eine Anwendung (Alpstein) gibt in der Anfrage das Referenzsystem als Parameter mit, alle anderen Anwendungen nicht. Zwar lassen sich aus den Koordinaten der Ergebnisse bzw. der Boundingbox einer räumlichen Filterung die Referenzsysteme ableiten, dies kann jedoch zu Fehlern führen.

3.4.4 Ergebnisse der Analyse

Aus der großen Zahl der im Internet verfügbaren Webmapping-Anwendungen wurden zehn ausgesucht. Diese Auswahl wurde hinsichtlich der in Kapitel 3.4.2 aufgestellten Kriterien untersucht. Die detaillierten Ergebnisse dieser Anforderungsanalyse befinden sich im Anhang A.

3.4.5 Auswertung und Ableitung der Anforderungen

In diesem Kapitel wurden bisher verschiedene Anwendungen des Geomassenmarkts vorgestellt und analysiert. Aus den Ergebnissen der Analyse sollen nun Anforderungen des Geomassenmarkts an einen Standard zur Bereitstellung von Vektordaten abgeleitet werden.

Lesender Zugriff

Alle analysierten Webanwendungen bieten Lesezugriff auf ihre Daten an, keine bietet zusätzlichen Schreibzugriff auf die Daten. Dass ein Lesezugriff ausreicht, wird auch durch die Art der möglichen Datenquellen deutlich: diese bieten über die APIs ebenfalls meist nur Lesezugriff, Änderungen werden vom User in passwortgeschützten Bereichen der Webseiten vorgenommen.

Requests im CGI-Standard

Lediglich Nestoria unterstützt REST, alle anderen (9 von 10) Webanwendungen erwarten Anfragen im CGI-Standard.

Ausgabeformate

Die durchgeführte Analyse hat ergeben, dass fünf von zehn Anwendungen JSON bzw. GeoJSON als Ausgabeformat unterstützen. Proprietäres XML geben drei Dienste, KML, GeoRSS und Text jeweils ein Dienst zurück.

Obwohl KML und GeoRSS im Geomassenmarkt weit verbreitet sind, werden diese von den analysierten Diensten auffällig selten unterstützt. Vor diesem Hintergrund darf ein möglicher Standard kein Ausgabeformat vorschreiben. Um eine möglichst gute Interoperabilität zu ermöglichen, ist es dennoch sinnvoll, standardisierte Format wie GeoJSON, KML oder GeoRSS zu unterstützen.

Raumbezug

Vier der analysierten Anwendungen nutzen WGS 84 als Referenzsystem, weitere vier GK 3 bzw. GK4. Alpstein nutzt als einzige Anwendung eine Sphärische Mercatorprojektion. Bei einer Anwendung kann keine Aussage getroffen werden.

Bei der Analyse der verwendeten Raumbezugssysteme fallen zwei Aspekte auf: zum Einen werden in den wenigsten Anwendungen Raumbezugssysteme als Parameter, ähnlich dem Parameter „SRS=EPSG:4326“ eines WMS-Dienstes, mitgegeben. Zum Anderen werden auch Referenzsysteme wie Gauß-Krüger Zone 3 bzw. 4 eingesetzt.

Dies ist insofern bemerkenswert, als in Anwendungen des Geomassenmarktes Geodaten meist als geographische Koordinaten (WGS84 bzw. EPSG 4326) vorliegen oder in der von Google Maps verwendeten Sphärischen Mercatorprojektion (EPSG 900913) genutzt werden.

Aus der Streuung der verwendeten Referenzsystemen lässt sich ableiten, dass ein möglicher Standard WGS 84 als Standardreferenzsystem unterstützen muss. Zusätzlich sollte es aber auch möglich sein, von WGS 84 abweichende Referenzsysteme zu nutzen. Die Angabe eines EPSG-Codes muss bei Verwendung von WGS 84 nicht zwingend nötig sein.

Attributive Filterung

Sechs von zehn Anwendungen erlauben eine Filterung und damit eine Einschränkung der zurückgegebenen Ergebnisse. Ein Filterparameter besteht meist aus einem Schlüssel und dem Wert des Filters (Key-Value-Pair, KVP). Werden mehrere Filter verwendet, so werden diese mit AND verknüpft.

Ein massenmarkttauglicher Standard soll somit eine Filterung mittels KVP und die Verknüpfung mehrerer Filter mit AND unterstützen.

Räumliche Filterung

Vier von zehn Anwendungen erlauben eine räumliche Filterung. Diese Form der Filterung soll ein möglicher Standard ebenfalls unterstützen.

Temporale Filterung

Die Filterung nach Zeiträumen bietet nur eine Anwendung an. Da jedoch viele Datenquellen des Geomassenmarkts eine Suche nach Zeiträumen vorsehen, sollte ein Standard zur Bereitstellung von Vektordaten dies ebenfalls optional unterstützen.

Kapitel 4

Wfs Simple - OpenGIS-konforme Bereitstellung von Vektordaten

Im bisherigen Verlauf wurden drei wesentliche Punkte herausgearbeitet:

- Neuartige Anwendungen im Geomassenmarkts entstehen durch Kombination von Kartendiensten und primär nicht räumlicher Daten aus den verschiedensten Quellen
- Jede Datenquelle besitzt oft eigene Zugriffsmöglichkeiten
- Eine effektive Integration dieser Datenquellen erfordert einen Standard, der die Anforderungen aus Kapitel 3.4.5 erfüllt.

In diesem Kapitel wird ein OpenGIS-Standard, der Web Feature Service Simple, vorgestellt und Gemeinsamkeiten und Unterschiede zum Web Feature Service (WFS) herausgearbeitet.

4.1 Web Feature Service (WFS)

In der Einführung der Web Feature Service-Spezifikationen (vgl. Open Geospatial Consortium Inc. 2005) definiert das OGC den WFS als Dienst, der es Clients ermöglichen soll, räumliche Daten als GML zu empfangen und zu verändern (vgl. ebd:10).

Unterstützte Operationen

Operationen, die ein WFS unterstützen kann bzw. muss, sind (vgl. ebd:2f):

- GetCapabilities
- DescribeFeatureType
- GetFeature
- GetGmlObject
- Transaction
- LockFeature

Ausgehend von diesen Operationen können folgende WFS Typen gebildet werden (vgl. ebd:3):

Der Basic WFS umfasst mit den Operationen GetCapabilities, DescribeFeatureType und GetFeature die nötige Funktionalität für einen nur lesenden Zugriff (Read only).

Der XLink WFS umfasst zusätzlich zum Funktionsumfang des Basic WFS noch die GetGmlObject-Operation.

Der Transaction WFS erweitert den Basic WFS um die Transaction-Operation, um Daten zurückzuschreiben. Die GetGmlObject- und die LockFeature-Operationen sind optional möglich.

Ausgabeformate

Ein WFS darf Daten nur als Gml zurückgeben; in der Version 1.1.0 finden Gml 2.1.2 und 3.1.1 (vgl. ebd:34) Verwendung.

Filterung

Die Filterung nach nicht räumlichen Attributen basiert auf dem OpenGIS Filter Encoding Standard (FES) (vgl. ebd:23).

4.2 Web Feature Service Simple (WFS-S)

Der Webservice „Web Feature Service Simple" hat lediglich den Status eines „Discussion Paper" (vgl. Open Geospatial Consortium Inc. 2009b:o.S.) und ist somit kein offizieller Standard wie der Web Feature Service. Allerdings bietet das OGC im Gegensatz zu anderen Discussion Papers im ogcnetwork (http://www.ogcnetwork.net) eine Plattform zur Sammlung von Informationen und Tools.

Der WFS Simple-Standard wird vor allem durch zwei Punkte gekennzeichnet (vgl. Open Geospatial Consortium Inc. 2009c:o.S.)

- Der WFS Simple soll der Minimalstandard für räumlich-zeitliche Abfragen im Web durch die Parameter BBOX und TIME sein
- Es sollen räumliche Abfragen von Mainstream Webanwendungen wie Blogs ermöglicht werden.

Zudem wird mit dem WFS Simple-Standard der Ansatz verfolgt, einen einfacheren Standard als den WFS zu definieren, der sich mehr am Massenmarkt orientiert (vgl. Open Geospatial Consortium Inc. 2007a:28).

4.2.1 Service

Nach (ebd:29) wird als Servicename „WFSS“ festgelegt. An dieser Stelle sind die Spezifikationen widersprüchlich: unter Punkt „8.4.1.1 Service parameter“ wird jedoch von WFS-S (vgl.ebd.:29) gesprochen. Da im weiteren Verlauf der Spezifikationen allerdings ausschließlich WFSS (vgl. ebd.:44) verwendet wird, wird WFSS als der korrekte Wert betrachtet.

4.2.2 Version

Der Wert des Version-Parameters soll 0.6.0 lauten (vgl. Open Geospatial Consortium Inc. 2007:29).

4.2.3 Ausgabeformate

Ein WFS Simple darf die Daten in jedem beliebigen Format zurückgeben. Gml ist kein verpflichtendes Ausgabeformat (vgl. ebd:28).

4.2.4 Operationen

Die WFS Simple-Spezifikationen kennen drei Operationen (vgl. ebd):

- GetCapabilities
- DescribeFeatureType
- GetFeature

Mit Ausnahme der DescribeFeatureType-Operation sind alle Operationen verpflichtend. Damit stellt ein WFS Simple den gleichen Satz an Operationen wie ein Basic WFS zur Verfügung; es handelt sich somit um einen Read-only-Dienst.

GetCapabilities

Der WFS Simple unterstützt die GetCapabilities-Operation, wie sie in den OGC Web Services Common Specification (OGC 05-008c1) beschrieben wird (vgl. ebd:30).

Das vom WFS Simple zurückgegebene GetCapabilities-Dokument besteht nur aus den Abschnitten ServiceIdentification, ServiceProvider und OperationsMetadata (vgl. ebd:30).

DescribeFeatureType

Diese optionale Operation soll die von der GetFeature-Operation zurückgegebenen Daten näher beschreiben. Diese Metadaten sollen aus drei Komponenten bestehen, einer textlichen Darstellung, die die Daten näher beschreibt, einer Auflistung der abfragbaren Eigenschaften sowie eine formale Beschreibung der Daten als Xml-Schema o. ä. (vgl. ebd:33). Als Ausgabeformate schlagen die Spezifikationen Plain Text, Simple Gml oder BXFS vor.

BXFS (Basic XML Feature Schema) ist der Entwurf eines Xml-Schemas, um Antworten der DescribeFeatureType- und GetFeature-Operationen des WFS Simple zu definieren (vgl Open Geospatial Consortium Inc. 2009d:o.S.).

GetFeature

Die GetFeature-Operation stellt die zentrale Funktionalität eines WFS Simple, die Bereitstellung von Vektordaten, zur Verfügung. Außer dem Parameter „Request“ mit dem festen Wert „GetFeature“ sind alle anderen Parameter optional (vgl. Open Geospatial Consortium Inc. 2007:37f).

4.2.5 Filterung

Die GetFeature-Anfrage an einen WFS Simple kann durch Filter räumlich, temporal oder attributiv eingeschränkt werden. Alle vorkommenden Filter werden mit einer logischen AND-Verknüpfung verbunden (vgl. ebd:41)].

Räumliche Filterung

Eine räumliche Einschränkung erfolgt, wie bei anderen OpenGIS Web Services auch, mittels einer Boundingbox. Dazu wird dem Request der Parameter BBOX und die Werte lcc1,lcc2,ucc1,ucc2 angehängt. Das genaue Format ist in OGC 05-008c1 beschrieben.

Temporale Filterung

Die Spezifikationen des WFS Simple sehen eine temporale Filterung mittels des Parameters TIME vor. Die Zeitangaben orientieren sich an dem Standard nach ISO8061 (vgl.ebd:37). Durch Angabe einer Start- und Endzeit kann nach Zeitspannen gefiltert werden.

Attributive Filterung

Die Filterung der Attribute kann auf zwei Arten erfolgen. Der Parameter „Propertyname“ legt fest, welche Attributfelder bzw. Spalten einer Datenbank überhaupt ausgegeben werden sollen (vgl.ebd:41). Als Wert werden die gewünschten Spaltennamen als durch Kommata getrennte Liste übergeben.

Sollen hingegen nur nach bestimmten Eigenschaften gefilterte Features zurückgegeben werden, so kann dies mittels eines Parameters „`PropertyNameQuery=regex`“ (vgl.ebd:41) erfolgen. Als Schlüssel können nur Attribute verwendet werden, die in der Antwort auf die DescribeFeatureType-Operation als „queryable“ gekennzeichnet sind (vgl.ebd:33, auch ebd:41). regex stellt dabei das Filterkriterium in Form eines Regulären Ausdrucks dar, mit dem die Attributwerte gefiltert werden sollen.

Kapitel 5

Entwurf, Implementierung und Evaluation eines WFS Simple-Prototypen

5.1 Zielsetzung

Ziel des praktischen Teils dieser Master Thesis ist es, einen Prototypen eines Web Feature Service Simple zu erstellen, wie er in (vgl. ebd) spezifiziert ist.

Der Prototyp soll auf dem Microsoft .net-Framework und dem Microsoft Internet Information Server als Webserver basieren. Als Programmiersprache findet C# Verwendung.

Der Prototyp soll so entworfen werden, dass die Unterstützung weiterer Datenquellen und weiterer Ausgabeformate jederzeit hinzugefügt werden kann.

5.2 Vorgehen

Um oben genannte Ziele zu erreichen, sind vier Schritte notwendig:

Festlegung des Funktionsumfangs

In diesem Schritt werden die zu implementierenden Datenquellen, Ausgabeformate und Operationen festgelegt.

Entwurf

Auf Grundlage des Funktionsumfangs erfolgt der Entwurf des Prototypen. Da ein objektorientierter Ansatz verfolgt wird, geschieht dies mit der Methodik des Klassendesigns. Als Ergebnis stehen die Definitionen aller Klassen in Form eines Klassendiagramms fest.

Realisierung

Das festgelegte Klassendesign wird mit Leben gefüllt. Es werden die benötigten Funktionen implementiert und als Klassenbibliothek kompiliert.

Validierung

Die Validierung besteht aus drei Schritten:

- Bereitstellung von Testdaten
- Konfiguration des WFS Simple
- Durchführung der Validierung

5.3 Festlegung des Funktionsumfangs

Operationen

Die möglichen Operationen sind in (vgl. ebd.) festgelegt. Obligatorisch müssen GetCapabilities und GetFeature implementiert werden. Die optionale DescribeFeatureType-Operation soll ebenfalls implementiert werden.

Datenquellen

Als Datenquelle soll Microsoft SQL Server 2008 verwendet werden. Eine Erweiterung um weitere Datenquellen ist vorzusehen.

Ausgabeformate

Der WFS Simple soll die Vektordaten als KML zurückgeben. Die Erweiterung um weitere Ausgabeformate soll möglich sein.

BXFS als mögliches Ausgabeformat der DescribeFeatureType-Operation wird nicht implementiert. Stattdessen soll ein einfaches XML-Dokument zurückgegeben werden.

5.4 Klassendesign

5.4.1 Entwurf

Der Entwurf des Klassendesigns erfolgt mit Hilfe eines CASE-Tools. Dieser „Klassendesigner“ ist in Visual Studio 2005 integriert und ermöglicht so die Bearbeitung von Klassendiagrammen.

Vorteil dieser Einbindung ist, dass im Hintergrund ein Abgleich zwischen Klassendiagramm und Quellcode stattfindet. Dadurch wirken sich Änderungen am Quellcode auf das Klassendesign und umgekehrt aus. Eine Einführung in die Umsetzung mittels des Klassendesigners ist in Louis (2008:292ff) zu finden.

Als Ergebnis erhält man zum Einen das Klassendiagramm als Grafik und zum Anderen ein Grundgerüst der Klassen als Quellcode. Dieses wird im nächsten Schritt, der Realisierung, mit Leben gefüllt.

Das vollständige Klassendiagramm findet sich im Anhang B.

5.4.2 Diskussion des Klassendiagramms

Im Folgenden werden nun die Klassen des WFS Simple-Implementierung kurz erläutert. Aus Gründen der Übersichtlichkeit sind die privaten Eigenschaften ausgeblendet. Ein vollständiges Klassendiagramm mit allen privaten Eigenschaften findet sich im Anhang.

Namespace WfsSimple

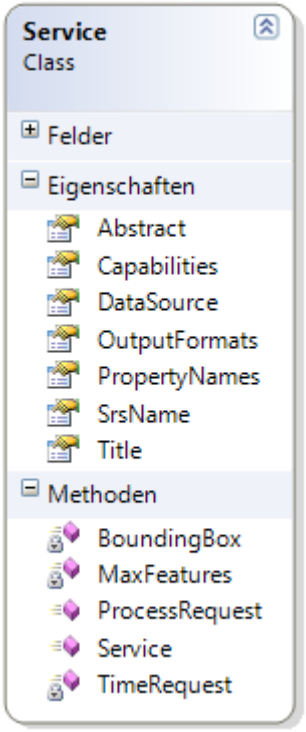


Abb. 5: Klassendiagramm Namespace WfsSimple

Der Namespace WfsSimple beinhaltet die Klasse „Service“. Die öffentliche Methode „ProcessRequest“ wertet die vom Browser empfangenen http-Anforderuneng aus, prüft diese auf die Gültigkeit der Parameter, fordert die Ergebnisse der Operationen GetCapabilities, DescribeFeatureType und GetFeature an und gibt diese an den Client zurück.

Über öffentliche Eigenschaften wird der Dienst konfiguriert. Die drei Eigenschaften „Capabilities“, „DataSource“ und „OutputFormats“ sind eigene Datentypen und werden in den Klassen der Namespaces WfsSimple.DataSource, WfsSimple.ExportFormats und WfsSimple.Capabilities definiert.

Namspace WfsSimple.DataSource

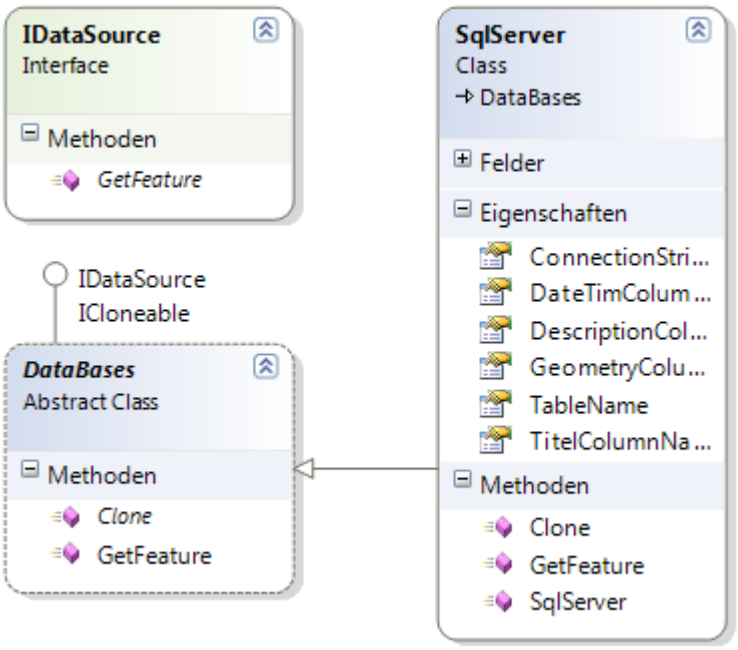


Abb. 6: Klassendiagramm Namespace WfsSimple.DataSource

Im Namespace WfsSimple.DataSources sind die Klassen für den Zugriff auf die unterschiedlichen Datenquellen enthalten. In der Schnittstelle „IDataSource“ ist die öffentliche Methode „GetFeature“ definiert, die an die abstrakte Klasse „DataBases“ vererbt werden. Die Klasse „SqlServer“ erbt wiederum die Methode „GetFeature“

Namespace WfsSimple.ExportFormats

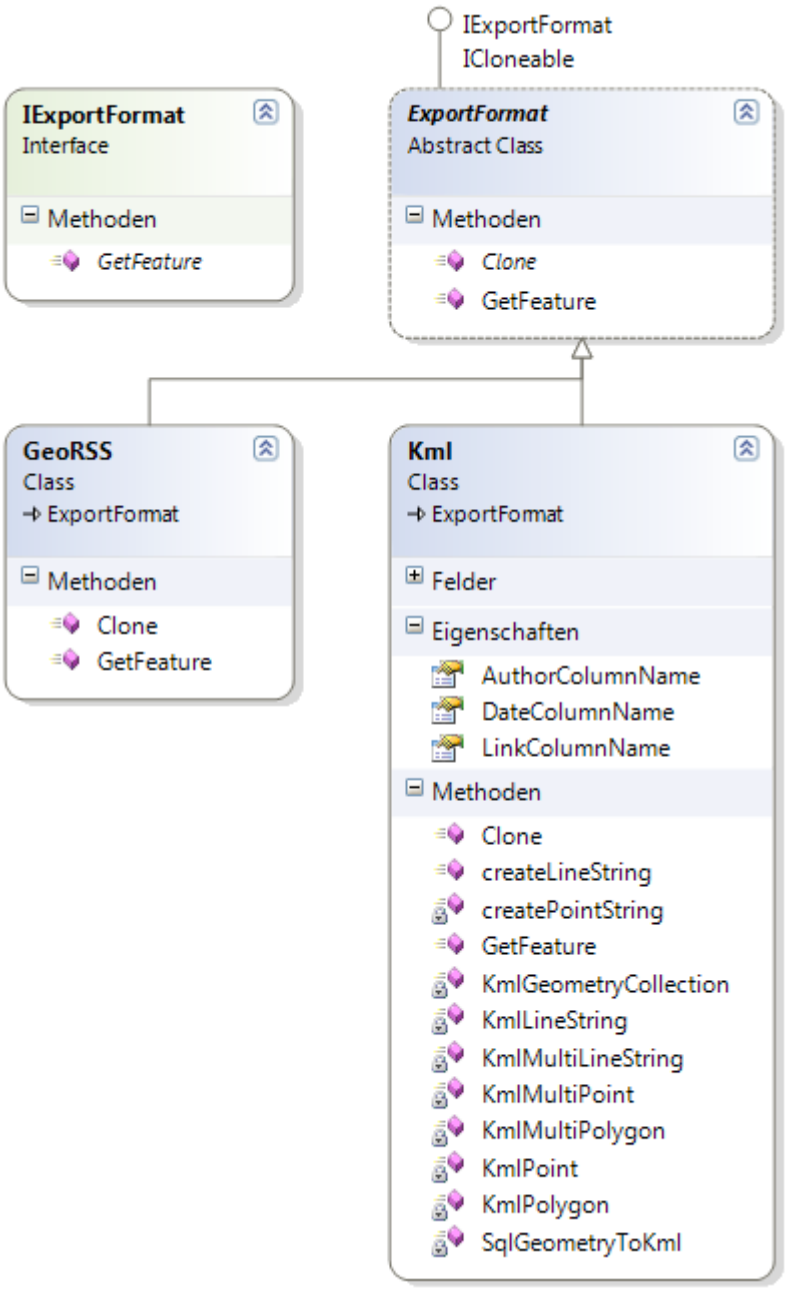


Abb. 7: Klassendiagramm Namespace WfsSimple.ExportFormats

Im diesem Namespace werden die vom WFS Simple unterstützten Ausgabeformate definiert. Hier stellt die Schnittstelle „IExportFormat“ die Methode „GetFeature“ zur Verfügung, die, über die abstrakte Klasse „ExportFormat“ vererbt, in der Klasse „Kml“ bzw. „GeoRSS“ implementiert wird.

Namespace WfsSimple.Capabilities

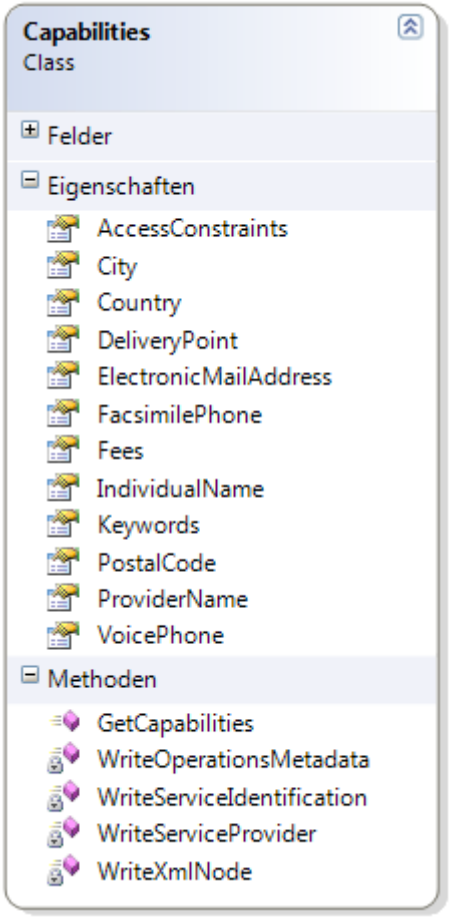


Abb. 8: Klassendiagramm Namespace WfsSimple.Capabilities

Der Namespace WfsSimple.Capabilities umfasst nur die Klasse „Capabilities“, welche die Antworten auf die GetCapabilities-Operation zurückgibt. Eine Realisation einer Schnittstelle sowie einer abstrakten Klasse ist nicht notwendig, da eine Möglichkeit zur Erweiterung nicht vorgesehen ist.

Namespace WfsSimple.DescribeFeatureType

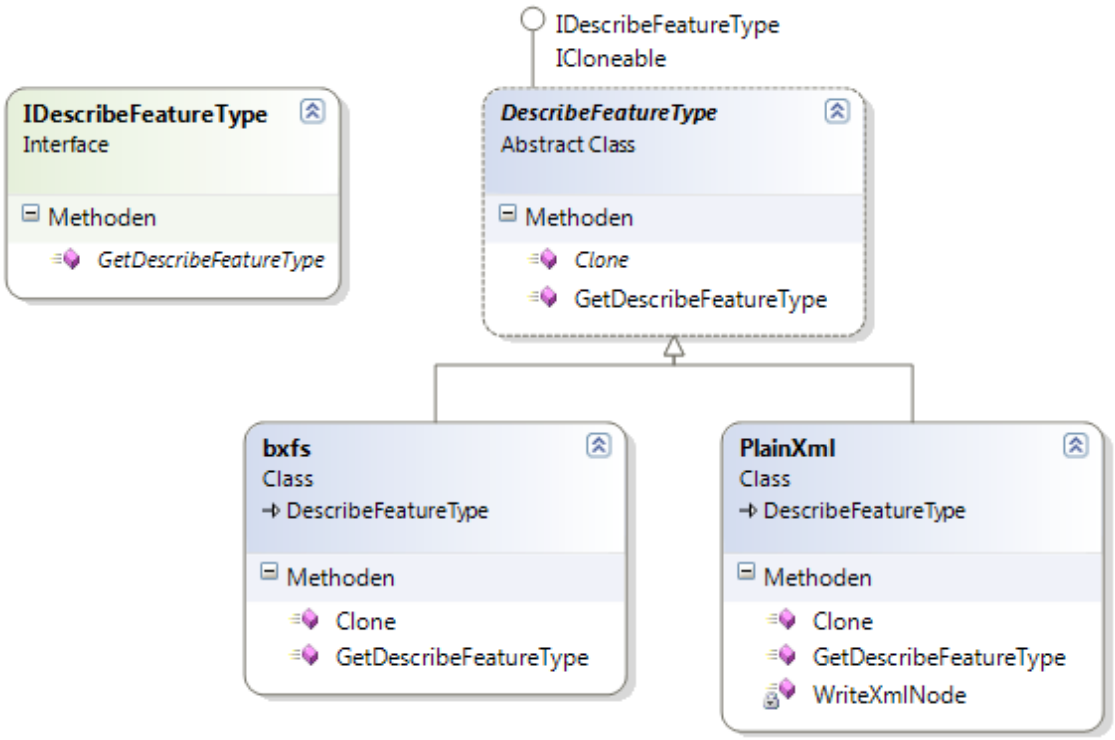


Abb. 9: Klassendiagramm Namespace WfsSimple.DescribeFeatureType

Im Namespace WfsSimple.DescribeFeature werden die Klassen zur Erstellung der Antworten auf die DescribeFeatureType-Operationen bereitgestellt. Auch hier stellt die Schnittstelle „IDescribeFeatureType“ die leere Methode „GetDescribeFeatureType“ zur Verfügung, die dann an die abstrakte Klasse „DescribeFeatureType“ vererbt wird. Die Klassen „PlainXml“ bzw. „bxfs“ implementieren schließlich diese Methode.

5.4.3 Erweiterbarkeit

Eine Anforderung an den WFS Simple-Prototypen ist die Erweiterbarkeit um weitere Datenquellen und Ausgabeformate. Dieser Anforderung wird das Klassendesign durch den Einsatz von Schnittstellen und abstrakten Klassen gerecht.

Das Vorgehen zur Erweiterung kann am Namespace WfsSimple.DataSources aufgezeigt werden. Dieser Namespace beinhaltet die Schnittstelle „IDataSource“, die abstrakte Klasse „DataBases“ und die Klasse „SqlServer“.

Soll nun eine weitere Datenbank als Datenquelle hinzugefügt werden, so muss diese die Methoden (GetFeature) und die Eigenschaften (ConnectionString, TableName etc.) der abstrakten Klasse „DataBases“ erben.

Soll der Namespace um einen anderen „Typ“ von Datenquellen wie APIs von Diensten erweitert werden, so muss zuerst eine abstrakte Klasse erstellt werden, die gemeinsame Eigenschaften wie einen ApiKey definiert. Die Implementierung der Funktionen zum Abruf der Dienste und die Formatumwandlung in ein DataSet erfolgt in den abgeleiteten Klassen.

5.5 Realisierung

Das im zweiten Schritt erstellte Klassendiagramm wird nun umgesetzt. Dabei finden einige Komponenten Verwendung, auf die nun besonders eingegangen werden soll.

5.5.1 SQL Server 2008

SQL Server 2008 ist die erste Version des Datenbankservers von Microsoft, die geographische Daten als eigenen Datentyp verwalten und analysieren kann. Neben den kostenpflichtigen Versionen steht auch die kostenfrei einsetzbare Version „SQL Server 2008 Express Edition“ zur Verfügung, die allerdings einigen Größenbeschränkungen unterliegt (vgl. Microsoft o.J.:o.S.).

Im SQL Server 2008 stehen zwei neue Datentypen zur Speicherung und Analyse räumlicher Daten zur Verfügung: geometry und geography.

Datentyp geography

Dem Datentyp SqlGeography liegt ein geographisches Koordinatensystem zu Grunde, als Standards wird WGS 84 verwendet (vgl. Aitchison 2009:35). Dieser Datentyp ist auf Objekte beschränkt, deren Ausdehnung maximal eine Hemisphäre bedeckt.

Datentyp geometry

Der Datentyp geometry ist geeignet für die Speicherung projizierter, ebener Daten und unterliegt keiner Beschränkung.

Da der Parameter BBOX des WFS Simple auch größere Gebiete als eine Hemisphäre abdecken kann, kann der Datentyp geography nicht verwendet werden. Deswegen wird im Prototyp durchgängig der Datentyp geometry gebraucht.

5.5.2 Reguläre Ausdrücke

Die Spezifikationen des WFS Simple sehen eine Filterung mittels Regulärer Ausdrücke (Regex bzw. RegExp) vor.

Reguläre Ausdrücke stellen eine mächtige Sprache zum Durchsuchen und Ändern von Texten dar (vgl. Liberty 2003:251. Im Gegensatz zur Suche mit Wildcards (* oder ?) können mittels Regulärer Ausdrücke auch Muster in Zeichenfolgen erkannt werden (vgl. ebd:251).

In Kapitel 4 dieser Arbeit wurde heraugearbeitet, dass eine Filterung meist nach dem Schema spalte=wert1,wert2,wert3 erfolgt. Eine Suche mit Wildcards oder nach bestimmten Mustern wurde nicht festgestellt. Dieses Suchschema lässt sich auch als Regulärer Ausdruck umsetzen. Dies geschieht mit Hilfe des Ausdrucks für Alternativen „|“. Das Äquivalent zu „wert1,wert2,wert3“ als Regulärer Ausdruck lautet somit „wert1|wert2|wert3“.

Reguläre Ausdrücke im .net-Framework

Auch das .net-Framework stellt Funktionen zur Nutzung von Regex zur Verfügung. Diese liegen im Namespace System.Text.RegularExpressions und umfassen Methoden zur Suche („Match“) und zum Ersetzen („Replace“).

Im WFS Simple werden lediglich die Funktionen zur Suche von Übereinstimmungen verwendet. Mit

```
Bool = Regex.IsMatch(input, pattern, RegexOptions.IgnoreCase);
```

wird ermittelt, ob die Zeichenfolge `input` den Regulären Ausdruck `pattern` enthält.

Reguläre Ausdrücke und Sql Server 2008

Obwohl die meisten Datenbanksysteme wie mySql, PostgreSql und Oracle Reguläre Ausdrücke verarbeiten können (vgl. Richardson o.J.:o.S.), fehlt diese Funktion im SQL

Server 2008 (vgl. ebd.:o.S.). Allerdings lässt sich dies durch die Erweiterung der Datenbank um benutzerdefinierte Funktionen mittels CLR (Custom Runtime Language) beheben.

Dazu ist es nötig, eine Klassenbibliothek zu erzeugen, welche die benutzerdefinierten Funktionen enthält und vom SQL Server ausgeführt wird (vgl. o. V. 2008:o.S.). Dadurch ist es möglich, die benutzerdefinierten Funktionen in einem T-SQL-Statement zu nutzen. Ein beispielhaftes Statement zur Suche nach Unterkünften der Kategorien Hotel oder Pension lautet:

```
SELECT * FROM Tabelle WHERE dbo.RegExMatch('Unterkunft', 'Hotel|Pension')
```

wobei der Ausdruck RegExMatch der benutzerdefinierten Methode der Klassenbibliothek entspricht.

5.6 Validierung

5.6.1 Ziel der Validierung

Durch eine Validierung soll sichergestellt werden, dass die vom WFS Simple zurückgegebenen Dokumente syntaktisch richtig und somit valide sind. Da Kml ein XML-Dialekt ist, erfolgt die Prüfung gegen das entsprechende XML-Schema.

5.6.2 Bereitstellung von Testdaten

Da ein vom WFS Simple- Prototypen erstelltes Kml-Dokument validiert werden soll, muss ein WFS Simple-Dienst umgesetzt werden. Dazu werden im ersten Schritt Testdatensätze bereitgestellt.

„Joe's Blue Lake Vicinity Map"

Zu den Simple Feature Specifications existiert ein „Conformance Test". Dieser besteht aus Vektordaten, die alle Geometrietypen der SFS umfassen (vgl. Open Geospatial Consortium 2001). Als Grafik ergibt dies die Karte von „Joe's Blue Lake Vicinity".

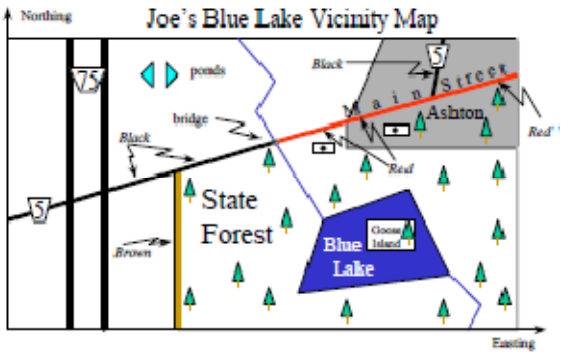


Abb 10: Ergebniskarte des Conformance Test
(vgl. Open Geospatial Consortium Inc. 2001:9)

Dazu wird ein SQL-Skript geschrieben, das eine Datenbanktabelle und Datensätze mit den in Open Geospatial Consortium Inc. (2001:11f) enthaltenen Vektordaten erstellt.

5.6.3 Konfiguration des WFS Simple

Im nächsten Schritt muss der WFS Simple-Dienst erstellt werden. Dies geschieht mittels eines sog. Generischen Handlers. „Ein ASP.NET-HTTP-Handler ist der Prozess (oder ‚Endpunkt'), der als Antwort auf eine Anforderung an eine ASP.NET-Webanwendung ausgeführt wird" (vgl. Microsoft 2009:o.S.). Beim Aufruf wird die Methode „ProcessRequest" aufgerufen, die die Anforderung verarbeitet und das Ergebnis an den Client zurückgibt.

Die Klassenbibliothek muss dem Generischen Handler hinzugefügt werden. Dazu muss die Klassenbibliothek in den Bin-Ordner der Webanwendung kopiert und der Namespace WfsSimple mit Hilfe der using-Direktive eingebunden werden.

In der ProcessRequest-Methode werden drei wesentliche Einstellungen getroffen:

- Allgemeine Eigenschaften des Dienstes:
  Allgemeine Eigenschaften des Dienstes müssen vorgegeben werden. Dies sind z. B. Titel, Abstract und angebotene Ausgabeformate
- Einstellungen für die GetCapabilities-Operation:
  Es müssen die für die Erstellung der Capabilities benötigten Informationen wie Ansprechpartner, AccessConstraints, Fees u. A. angegeben werden.
- Konfiguration der Datenquelle:
  Die Konfiguration ist abhängig von der Art der Datenquelle. Bei Nutzung des SQL Server 2008 sind dies u.a. ConnectionString, Tabellenname, Spaltennamen etc.

Ein beispielhafter Handler befindet sich im Anhang C.

5.6.4 Durchführung der Validierung

Die vom WFS Simple zurückgegebene KML-Datei soll vom Kmlvalidator (http://www.kmlvalidator.org) der Firma Galdos, der Entwicklerin von KML, validiert werden. Da kein vom Internet aus erreichbarer IIS zur Verfügung steht, wird die KML-Datei mit einem lokalen IIS erstellt, gespeichert und im Kmlvalidator hochgeladen.

Kmlvalidator validiert hinsichtlich drei Levels. Galdos erläutert die drei Levels in den FAQ (vgl. Kmlvalidator 2008:o.S.): Level 1 umfasst die Überprüfung der Muss-Anforderungen und Level 2 die Überprüfung von Sollanforderungen. Im Level 3 schließlich werden Sollanforderungen geprüft, die ein hohes Maß an Interoperabilität ermöglichen sollen.

Als Ergebnis gibt der Kmlvalidator an, welche Validierungsstufe erreicht bzw. verfehlt wurde. Die Validierung der vom WFS Simple zurückgegebenen Kml-Datei ist in allen drei Levels erfolgreich.

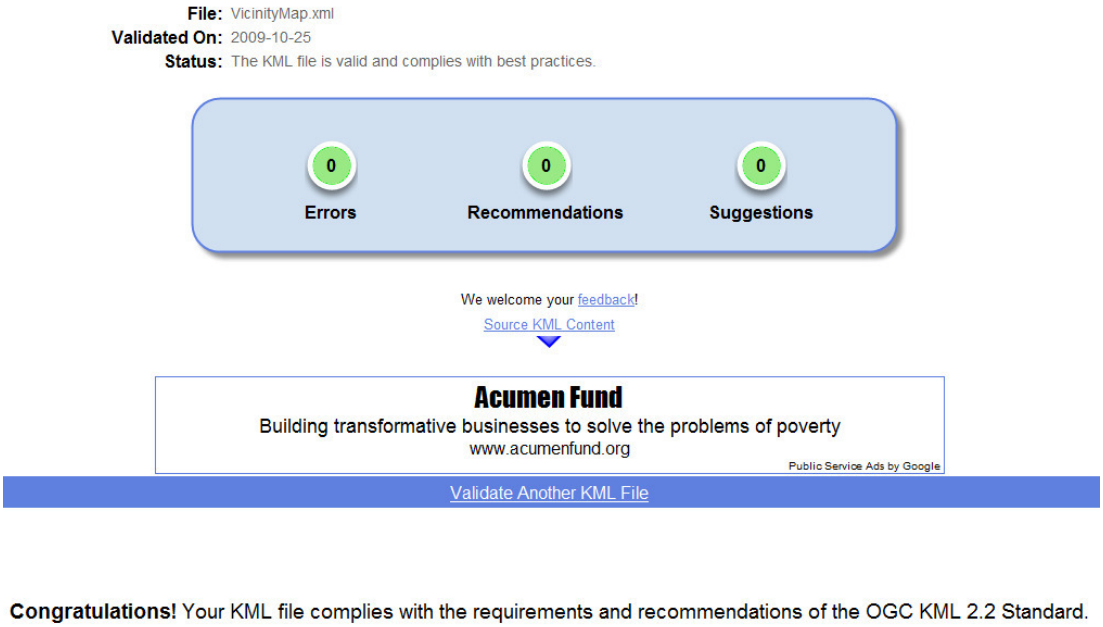


Abb. 11: Screenshot der erfolgreichen Validierung

5.7 Defizite des Prototypen

Auch wenn die Validierung der vom Prototypen erstellten KML-Dateien erfolgreich ist, weist die Umsetzung einige Defizite auf.

- Vollständige Umsetzungen der KML-Spezifiktionen
  Im Prototypen implementiert sind nur die grundlegenden Elemente wie die verschiedenen Geometrietypen und ExtendedData; alle anderen Elemente sind nicht implementiert. Es ist zu überlegen, ob und wie andere wichtige Elemente wie Styles, Icons, Viewpoints und Overlays sowie wichtige Eigenschaften wie Tesselating und Extruding umgesetzt werden können.
- Vollständige Umsetzung des Capabilities
  Im Prototypen ist nur ein Teil der möglichen Elemente aus Open Geospatial Consortium Inc. (2007a) umgesetzt. Um größtmögliche Interoperabilität und Konformität zu erhalten, müssen die restlichen Spezifikationen umgesetzt werden.
- Komponententests
  Nicht nur die KML-Dateien müssen validiert werden, auch die Einhaltung aller betroffenen OGC-Spezifikationen muss überprüft werden. Zusätzlich muss für die Verifikation der Klassenbibliothek ein Testprogramm erarbeitet werden.
- Nicht implementiert ist die Unterstützung anderer Koordinatensysteme als WGS 84 (EPSG 4326).
- Eine umfassende Fehlerbehandlung ist ebenfalls nicht implementiert.
- Auf Grund der Beschränkung des Datentyps geography auf eine Hemisphäre (siehe Kapitel 5.5.1) wird der Datentyp geometry verwendet.

Kapitel 6

Bewertung des WFS Simple-Standards

Im Mittelpunkt der Arbeit standen fünf Punkte: auf der Definition des Begriffs „Geomassenmarkt“ aufbauend wurden Kennzeichen des Geomassenmarkts untersucht. Anhand exemplarischer Anwendungen wurden im Anschluss die Anforderungen zur Bereitstellung von Vektordaten erarbeitet.

Im sich anschließenden Teil wurde ein Prototyp eines Web Feature Service Simple implementiert. Abschließend soll nun der WFS Simple-Standard bewertet werden.

6.1 Diskussion der Spezifikation

Bei genauer Betrachtung der Spezifikation bzw. beim Entwurf des Klassendesigns des Prototypen sind einige Punkte aufgefallen, die einer grundlegenden Diskussion bedürfen.

Ziel der optionalen DescribeFeatureType-Operation ist die Beschreibung des Formats der zurückgegebenen Daten. Soll der Dienst die Daten in einem standardisierten Format wie KML oder GeoRSS zurückgeben, so ist es sinnvoll, das entsprechende XML-Schema oder einen Link dorthin auszugeben. Andererseits müssen, um eine attributive Filterung zu ermöglichen, die abfragbaren Felder als solche gekennzeichnet werden (Open Geospatial Consortium 2007:33, 41), und zwar in der Antwort der DescribeFeatureType-Operation.

Dieser Widerspruch kann nur gelöst werden, indem die Angabe möglicher abfragbarer Felder in der Antwort auf die GetCapabilities-Operation enthalten ist. Dies ist prinzipiell möglich (vgl. Open Geospatial Consortium 2007b:) und wird auch bei WMS-Diensten angewendet. Wie aus (http://schemas.opengis.net/wms/1.3.0/capabilities_1_3_0.xsd) hervorgeht, wird als Attribut des Elements Layer festgelegt, ob ein Layer abfragbar ist. Auch beim WFS-Dienst ist dies analog möglich (vgl.).

Werden die abfragbaren Felder in der Antwort auf die GetCapabilities-Anfrage zurückgegeben, kann die DescribeFeatureType-Operation als wirklich optional angesehen werden.

6.2 Erfüllung der Anforderungen des Geomassenmarkts

In Kapitel 3.4 wurden durch eine Analyse bestehender Anwendungen die Anforderungen des Geomassenmarkts an einen Standard zur Bereitstellung von Vektordaten erarbeitet. Im Folgenden soll nun untersucht werden, inwieweit der WFS Simple diese Anforderungen erfüllt.

In der untenstehenden Tabelle werden dazu die Kriterien und die Ergebnisse der Anforderungsanalyse gegenübergestellt. In der letzen Spalte wird schließlich festgehalten, ob die Anforderungen aus Kapitel 3.4.5 vom WFS Simple erfüllt werden.

Kriterium	**Ergebnis der Anforderungsanalyse**	**WFS Simple** (vgl. OWS4)	**Erfüllung der Anforderung**
Zugriff	Nur lesend	Nur lesend	Ja
Requests	CGI-Standard	CGI-Standard	Ja
Ausgabeformate	JSON/GeoJSON KML GeoRSS andere	Beliebige Formate möglich	Ja
Referenzsystem	WGS 84 als Standard	EPSG 4326 (WGS 84) als Standard	Ja
	Angabe eines anderen SRS optional	Angabe eines anderen SRS möglich	Ja
Attributive Filterung	PropertyName=Value	PropertyName=REGEX	Nein
	Mehrere Filter mit „AND“ verknüpft	Mehrere Filter mit „AND“ verknüpft	Ja
Räumliche Filterung	Ja, optional	Ja, optional	Ja
Temporale Filterung	Ja, optional	Ja	Ja

Tab. 1: Kriterien und Ergebnisse der Anforderungsanalyse

Der Vergleich ergibt, dass die Spezifikationen des WFS Simple mit Ausnahme der attributiven Filterung mit den Anforderungen des Geomassenmarkts übereinstimmen. Während die ermittelten Anforderungen eine Filterung mit Key-Value-Pairs fordern, wird dies in den Spezifikationen des WFS Simple durch „Schlüssel=Regex“ gelöst.

6.3 Grenzen des WFS Simple-Standards

Die vorhin angesprochene Problematik der attributiven Filterung führt sogleich zu den Grenzen des WFS Simple-Standards.

Mittels Regulärer Ausdrücke können nur Zeichenfolgen, jedoch keine numerischen Werte gefiltert werden. Eine Filterung der letzten Erdbeben nach der Stärke beispielsweise ist ebenso wenig möglich wie die Auswahl von Standorten eines Fahrradverleihs mit einer Mindestzahl an verfügbaren Fahrrädern.

Diese Unzulänglichkeiten sind auch dem Autor des WFS Simple-Standards bekannt. Zusätzlich können Probleme auch bei Umlauten u. ä auftreten. Auf (vgl. Open Geospatial Consortium 2007:41f). Seiner Ansicht nach ist deswegen der Einsatz von Regex nicht ausreichend (vgl. ebd:42).

6.4 Abschließende Bewertung

Bei der Bewertung des WFS Simple zeichnete ein zweigeteiltes Bild. Während annähernd alle Anforderungen des Geomassenmarkts erfüllt werden, existiert in den Spezifikationen, die sich noch im Entwurfsstadium befinden, Nachbesserungsbedarf.

Das Verhältnis der GetCapabilities- und der DescribeFeatureType-Operationen zueinander muss geklärt werden, um eine bestmögliche Interoperabilität zu gewährleisten. Die Angabe der abfragbaren Felder sollte in den Capabilities des Dienstes erfolgen (siehe 6.1). Die optionale DescribeFeatureType-Operation soll das XML-Schema des Ausgabeformats zurückgeben.

Aus Sicht eines Webentwicklers, der einen WFS Simple-Dienst manuell als Datenquelle in seine Anwendung einfügt, sollten die Möglichkeiten, die die GetFeature-Operation zur Verfügung stellt, im Wesentlichen ausreichen.

6.5 Ausblick: Geodateninteroperabilität auch im Geomassenmarkt

In der vorliegenden Thesis wurde durch die Definition des Begriffs „Geomassenmarkt“ und die Darstellung seiner typischen Merkmale aufgezeigt, dass als Kern des Geomassenmarkts eine neuartige Nutzungsweise des Internets gesehen werden kann, die sich vor Allem durch eine intensive Nutzung geographischer Daten unterschiedlicher Herkunft und Formate auszeichnet. Diese Daten finden u.a. häufig in Mashups Verwendung.

Die bei der Durchführung der Anforderungsanalyse betrachteten Anwendungen zeigen aber auch, dass eine breite Nutzung durch andere Anwendungen bzw. Mashups durch unterschiedliche Diensteschnittstellen erschwert wird.

Die Betrachtung des WFS Simple-Standards hat ergeben, dass dieser die Anforderungen an einen Standard zum Datenaustausch im Geomassenmarkt im Wesentlichen erfüllt.

Der WFS Simple könnte als zentraler Bestandteil einer „Massenmarkt-Geodateninfrastruktur“ helfen, die Bereitstellung von Vektordaten zu standardisieren und zu vereinheitlichen. Die Folge des Einsatzes des WFS Simple ist somit eine höhere Interoperabilität zwischen Datenlieferanten und -konsumenten.

Literaturverzeichnis

Aitchison, A. (2009): Beginning Spatial with SQL Server 2008. New York: Apress.

Alby, T. (2008³): Web 2.0. Konzepte, Anwendungen, Technologien.München: Hanser.

Bayer, T. (2002): REST Web Services. Eine Einführung.<http://www.oio.de/public/xml/rest-webservices.pdf> (Zugriff: 03.09.2009) (Stand: November 2002)

Creative Commons (o.J.):Creative Commons Attribution 3.0 Unported. <http://creativecommons.org/licenses/by/3.0> (Zugriff: 17.08.2009)

ESRI (2009a): ArcGIS Online Help. Adding items. <http://www.arcgisonline.com/help/content/add/add.htm> (Zugriff: 15.09.2009)

ESRI (2009b): ArcGIS Online Help. Sharing your items with others. <http://www.arcgisonline.com/help/content/share/sharing.htm> (Zugriff: 15.09.2009, nur mit Anmeldung möglich)

ESRI (2009c): ArcGIS Explorer. <http://www.esri-germany.de/products/arcgis/explorer/index.html> (Zugriff: 10.10.2009)

ESRI (2009d): ArcGIS Explorer Key Features.<http://www.esri.com/software/arcgis/explorer/key-features.html> (Zugriff: 30.09.2009)

ESRI (2009e): KML output and operations.
<http://services.arcgisonline.com/ArcGIS/SDK/REST/index.html?servicesdirectory.html> (Zugriff: 03.10.2009)

Flickr (2004): A Round of New Features!
<http://blog.flickr.net/en/2004/06/29/news-2004-6-29-2> (Zugriff: 09.08.2009) (Stand: 29.06.2004)

Flickr (2009): Flickr Services - API Keys.
<http://www.flickr.com/services/api/misc.api_keys.html> (Zugriff: 09.08.2009)

Garret, J. (2005): Ajax: A New Approach to Web Applications.
<http://www.adaptivepath.com/ideas/essays/archives/000385.php> (Zugriff: 17.07.2009, nur mit Anmeldung möglich) (Stand: 18.02.2005)

GDAL (o.J.): OGR Vector Formats. <http://www.gdal.org/ogr/ogr_formats.html> (Zugriff: 10.09.2009)

Gehtland, J., Galbright, B. & Almaer, D. (2006): Ajax. Eine pragmatische Einführung in Web 2.0. München: Hanser.

Geodateninfrastruktur Deutschland (2009): GeoRSS - Nachrichtenverarbeitung mit Raumbezug <http://www.gdi-de.de/de_neu/thema/2009/c_thema_georss.html> (Zugriff: 03.08.2009) (Stand: 02.06.2009)

GeoRSS (2009): GeoRSS - Main Page. <http://www.georss.org> (Zugriff: 03.08.2009) (Stand: 22.04.2009)

Google (2005): Mapping your way.
<http://googleblog.blogspot.com/2005/02/mapping-your-way.html> (Zugriff: 16.06.2009)

Google (2009a): Navigating in Google Earth. <http://earth.google.com/support/bin/static.py?page=guide.cs&guide=22358&topic=22361&answer=148186#navcontrols> (Zugriff: 22.09.2009)

Google (2009b): Google Verzeichnis. <http://maps.google.de/gadgets/directory?synd=mpl> (Zugriff: 09.08.2009)

Google (2009c): Google Suggest. <http://www.google.de/webhp?complete=1> (Zugriff: 09.08.2009)

Google (2009d): Google Maps. <http://maps.google.de> (Zugriff: 09.08.2009)

Heise (2005): 3D-Navigator von Google. <http://www.heise.de/newsticker/meldung/3D-Navigator-von-Google-111961.html> (Zugriff: 22.09.2009) (Stand: 28.06.2005)

Höffken, S. (2009): Google Earth in der Stadtplanung. Die Anwendungsmöglichkeiten von Virtual Globes in der Stadtplanung am Beispiel von Google Earth. <http://www.isr.tu-berlin.de/downloads/publikationen/graue_reihe/Graue_Reihe-Heft_19-Hoeffken.pdf> (Zugriff: 07.07.2009)

Höhnke, A. (o.J.): Sichere Distribution digitaler Inhalte. Schutz digitaler Inhalte ohne Verschlüsselung. <http://www.geo-targeting.de/geotargetinganwendungen/targetdcontentdistribution/index.php> (Zugriff: 17.09.2009)

Howard, B., Daly, M., Doyle, A., Gillies, S., Schaub, T. & Schmidt, C. (2008): The GeoJSON Format Specification. <http://geojson.org/geojson-spec.html> (Zugriff: 18.06.2009) (Stand: 16.06.2008)

ItOpen (2009a): ItOpen - Open Web Solutions, WebGis Developement. <http://www.itopen.it> (Zugriff: 03.09.2009)

ItOpen (2009b): KML MapServer. <http://www.itopen.it/soluzioni/kml-map-server> (Zugriff: 03.09.2009)

JSON (o. J.): JSON. <http://json.org> (Zugriff: 18.06.2009)

Kmlvalidator (2008) Kml Validator Frequently Asked Questions. <http://www.kmlvalidator.com/content/faq.htm> (Zugriff: 22.09.2009)

Knorr, E. (2003): The Year of Web Services. In: CIO Special Issue Dezember 2003/Januar 2004, 90.

Koops, L. (2008): The most recent Version of the Internet. What exactly is Web 2.0? In: GEOInformatics Dezember 2008, 24-25

Kunze, R. (2006): SVGWeather - Entwicklung einer SVG Web Mapping Applikation zur Visualisierung von vierdimensionalen Daten am Beispiel von Wettervorhersagedaten.. <http://elib.ub.uni-osnabrueck.de/publications/diss/E-Diss603_thesis.pdf> (Zugriff: 21.06.2009)

Langfeld, D. (2006): Entwicklung einer SVG Web Mapping Applikation zur Visualisierung von Geoinformationen.. <http://www.inf.uos.de/prakt/pers/dipl/dlangfel.pdf> (Zugriff: 21.06.2009)

Liberty, J. (2003³): Programming C#. Sebastopol: O'Reilly.

Louis, D., &Strasser, S. (2008): Microsoft Visual C# 2008. Grundlagen, Techniken, Profi-know-how. Unterschleißheim: Microsoft Press.

Maguire (2008): GeoWeb 2.0 and its Implications for Geographic Information Science and Technology. <http://www10.giscafe.com/link/display_detail.php?link_id=23176> (Zugriff: 22.07.2009) (Stand: 23.03.2008

Majewski, B. (2006): Google Maps API Blog. Geocoding at last! <http://googlemapsapi.blogspot.com/2006/06/geocoding-at-last.html> (Zugriff: 09.08.2009) (Stand: 13.06.2006)

Mass Market Geo WG (2009): Mass Market Geo WG. <http://www.opengeospatial.org/projects/groups/massmarket> (Zugriff: 22.06.2009)

MetaCarta (2008): FeatureServer -- RESTful Geographic Feature Storage. <http://featureserver.org> (Zugriff: 03.09.2009)

Microsoft (2009): Einführung in HTTP Handler. <http://msdn.microsoft.com/de-de/library/ms227675(VS.80).aspx> (Zugriff: 22.09.2009)

Microsoft (o.J.): Compare Editions–Compact and Express. <http://www.microsoft.com/sqlserver/2008/en/us/compare-specialized.aspx> (Zugriff: 13.02.2009)

Mulka, Kyle (2005): WMS in Google Maps. <http://blog.kylemulka.com/2005/08/wms-in-google-maps> (Zugriff: 18.06.2009) (Stand: 22.08.2005)

nextbike (o.J.a): nextbike-Das Fahrradverleihsystem. <http://nextbike.de/de#sogehts> (Zugriff: 26.08.2009)

nextbike (o.J.b): Offizielle nextbike-Standorte. <http://nextbike.de/de#standortliste> (Zugriff: 26.08.2009)

nextbike (o.J.c): nextbike: Standorte. <http://nextbike.de/#standortkarte> (Zugriff: 26.08.2009)

o. V. (2008): Regular Expressions In MS SQL Server using CLR. <http://anastasiosyal.com/archive/2008/07/05/regular-expressions-in-ms-sql-server-using-clr.aspx> (Zugriff: 14.08.2009) (Stand: 05.07.2008)

Öfele, D. & Luderschmid, F. (2006): Studie: Earth Explorer – Systemevaluierung. <http://www.rtg.bv.tum.de/images/stories/downloads/aktuelles/systemevaluierung/studie_vergleichsmatrix.jpg> (Zugriff: 22.09.2009) (Stand: 03.08.2006

Open Geospatial Consortium Inc. (2001): OpenGIS Simple Features Test Protocol for CORBA. <http://portal.opengeospatial.org/files/?artifact_id=7589&version=1> (Zugriff: 22.09.2009)

Open Geospatial Consortium Inc. (2005a): Web Feature Service Implementation Specification. <http://portal.opengeospatial.org/files/index.php?artifact_id=8339> (Zugriff: 21.05.2009) (Stand: 03.05.2005)

Open Geospatial Consortium Inc. (2005b): OpenGIS® Implementation Specification for Geographic information - Simple feature access - Part 1: Common architecture. <http://portal.opengeospatial.org/files/?artifact_id=13227> (Zugriff: 18.06.2009) (Stand: 22.11.2005)

Open Geospatial Consortium Inc. (2007a): OWS-4 GeoDDS Mass Market (formerly GeoRSS) Interoperability Program Report. <http://portal.opengeospatial.org/files/?artifact_id=20431> (Zugriff: 25.01.2009) (Stand: 02.05.2007)

Open Geospatial Consortium Inc. (2007b): OGC Web Services Common Specification. <http://portal.opengeospatial.org/files/?artifact_id=20040> (Zugriff: 09.02.2007)

Open Geospatial Consortium Inc. (2008a): KML 2.2 SWG. <http://www.opengeospatial.org/projects/groups/kml2.2swg> (Zugriff: 21.09.2009) (Stand: 11.06.2008)

Open Geospatial Consortium Inc. (2008b): OGC® Approves KML as Open Standard. <http://www.opengeospatial.org/pressroom/pressreleases/857> (Zugriff: 19.07.2009) (Stand: 14.04.2008)

Open Geospatial Consortium Inc. (2008c): OGC® KML. <http://portal.opengeospatial.org/files/index.php?artifact_id=27810> (Zugriff: 19.07.2009) (Stand: 14.04.2008)

Open Geospatial Consortium Inc. (2009a): OGC® KML Standard Development Best Practices. <http://portal.opengeospatial.org/files/index.php?artifact_id=30203> (Zugriff: 07.07.2009) (Stand: 04.02.2009)

Open Geospatial Consortium Inc. (2009b): Discussion Papers. <http://www.opengeospatial.org/standards/dp> (Zugriff: 25.01.2009)

Open Geospatial Consortium Inc. (2009c): OGC Network. WFS Simple. <http://www.ogcnetwork.net/wfssimple> (Zugriff: 21.05.2009

Open Geospatial Consortium Inc. (2009d): WFS Simple - standard data format proposal (BXFS). <http://www.ogcnetwork.net/node/189> (Zugriff: 25.01.2009)

OpenGeoNames (o.J.a): About GeoNames. <http://www.geonames.org/about.html> (Zugriff: 18.06.2009)

OpenGeoNames (o.J.b): GeoNames WebServices overview. <http://www.geonames.org/export/ws-overview.html> (Zugriff: 18.06.2009)

O'Reilly, T. (2005): What Is Web 2.0. Design Patterns and Business Models for the Next Generation of Software. <http://oreilly.com/lpt/a/6228> (Zugriff: 21.06.2009)

Parsons, E. (2006): OGC and the "Mass Market" <http://www.edparsons.com/2006/12/ogc-and-the-mainstream-market> (Zugriff: 21.06.2009) (Stand: 14.12.2006)

Pfronten (o.J.a): Interaktiver Ortsplan Pfronten. <http://www.pfronten.de/index.shtml?regionskarte> (Zugriff: 26.08.2009)

Pfronten (o.J.b): o.T. <http://www.pfronten.de/se_data/geodata/Gastgeber.kml> (Zugriff: 02.09.2009)

Ramsey, P. (2008): Mashing up the Enterprise. <http://www.refractions.net/expertise/whitepapers/mashups/mashups> (Zugriff: 21.06.2009)

Richardson, T. (o.J.): Exploring Various SQL RegEx Syntax. <http://trentrichardson.com/2008/10/23/exploring-various-sql-regex-syntax> (Zugriff: 14.08.2009)

Rousse, L., Bergeron, S. & Harris, T. (2007): Participating in the Geospatial Web: Collaborative Mapping, Social Networks and Participatory GIS. In: The Geospatial Web. How Geobrowsers, Social Software and the Web 2.0 are Shaping the Network Society, London:Springer

Schmidt, C. (2008): Processing Data With FeatureServer. <http://featureserver.org/doc/Processes> (Zugriff: 03.09.2009)

Singh, R. (2008): WMTS discussion paper - require reviewers. <https://lists.opengeospatial.org/mailman/private/mass-market-geo/2008-September/000231.html> (Zugriff: 08.07.2009) (Stand: 22.09.2008)

Turner, A. & Forrest, B. (2008): Where 2.0: The State of the Geospatial Web. <http://cachefly.oreilly.com/radar/research/Where2.0_excerpt.pdf> (Zugriff: 22.07.2009) (Stand: September 2008)

Turner, A. (2004): Geolocation by IP Address. <http://www.linuxjournal.com/article/7856> (Zugriff: 17.09.2009) (Stand: 25.10.2004)

W3C (2004): Web Services Architecture. W3C Working Group Note 11 February 2004. <http://www.w3.org/TR/2004/NOTE-ws-arch-20040211/#whatis> (Zugriff: 22.07.2009) (Stand: 11.02.2004)

Wernecke, J. (2009): The KML Handbook. Geographic Visualization for the Web. Boston: Addison-Wesley.

Anhang A

Ergebnisse der Anforderungsanalyse

Name der Anwendung	**URL Diensturl (Anm.)**	**http-Methode**	**REST/ CGI**	**r/w**	**Ausgabeformat**
Vianovis Flash	http://stadtplan.landshut.de (1)	GET	CGI	r	JSON
Vianovis html	http://www.dachau.com/ dachaumaps (2)	GET	CGI	r	JSON
Sh-Portal	http://www.sh-tourismus.de (3)	GET	CGI	r	JSON
Pegelonline	http://www.pegelonline.wsv.de (4)	GET	CGI	r	Tabelle/Plain Text
GeländeDB	http://www.dhv.de/odb/geostart.php?lang=de (5)	GET	CGI	r	Proprietares Xml, Ausgabe auch als HTML möglich
Felsinfo	http://www.dav-felsinfo.de (6)	GET	CGI	r	JSON
Alpstein	http://www.outdooractive.com/live/Alpregio-Regionskarte/Tegernsee-Schliersee (7)	GET	CGI	r	JSON
Pfronten	http://www.pfronten.de/index.shtml?regionskarte (8)	GET	CGI	r	Proprietäres Xml
nextbike	*http://www.nextbike.de* (9)	GET	CGI	r	Proprietäres Xml
nestoria	http://www.nestoria.de (10)	GET	REST	r	KML, GeoRSS

Anm.: Aus Gründen der Übersichtlichkeit finden sich die URLs der analysierten Dienste auf Seite 92.

1 http://stadtplan.landshut.de./amfx/gateway.php?PHPSESSID=vtpmesieq9kvblueeq3u0ahdbog9u5cm

2 http://kwis.visitcity.de/visitcity/proxy.php?url=http%3A//kwis.visitcity.de/amfx/json.php/SymbolServNewPremium.getSymbols/%2527324%2527/%2527963%2527//

3 http://www.sh-tourismus.de/tash/controller/extent?&minX=3325819.1516&minY=5820183.5153&maxX=3990982.9820000003&maxY=6152765.430500001&mapScale=1500000&sessionId=0&topics=0&areas=0&targetGroups=0&isAggregation=true&portalClientId=2&layers=Tourismusinformation%2CBahnhof%2CHotelPension%2CFerienwohnungHaus%2CCampingplatz%2CJugendherberge%2CBauernhof%2CBettBike%2CWellnesshotelBeautyfarm%2CMuseumDenkmal%2CTheater%2CSchlossGarten%2CKlosterKirche%2CFreizeitpark%2CTierparkZoo%2CbaseMap%2CLeuchttuerme%2C

4 http://www.pegelonline.wsv.de/internal/karte/openlayers/pegelinfo

5 http://www.dhv.de/odb/services/geo/xml.php?qi=glp_flaechenpositionen00&xid=fe65a78b7b&filter[Name]=Wallberg&filter[Land]=DE&filter[Bdld]=32&filter[Art]=4&filter[Eign_gs]=-1

6 http://www.dav-felsinfo.de/viewer/jsonviewer/extent?&minX=3571882.1437&minY=5208994.7929&maxX=4167195.8343&maxY=5506651.6383&mapScale=1500000&sessionId=0&topics=undefined&areas=undefined&targetGroups=undefined&isAggregation=false

7 http://www.alpserver.de/1.87/map/cluster.php?application=sfb&ACTIVEITEMS=&LANG=de&SOURCEID=ic_60&SOURCE=tegsee&HEIGHT=594&WIDTH=1488&BBOX=1261006,6042319,1374745,6087723&LEVEL=11&CRS=EPSG:900913&VERSION=1.2

8 http://www.pfronten.de/cgi-bin/siteengine.pl?GeoData::Extern::MarkerData&type=betrieb--fewo,feha,fedo

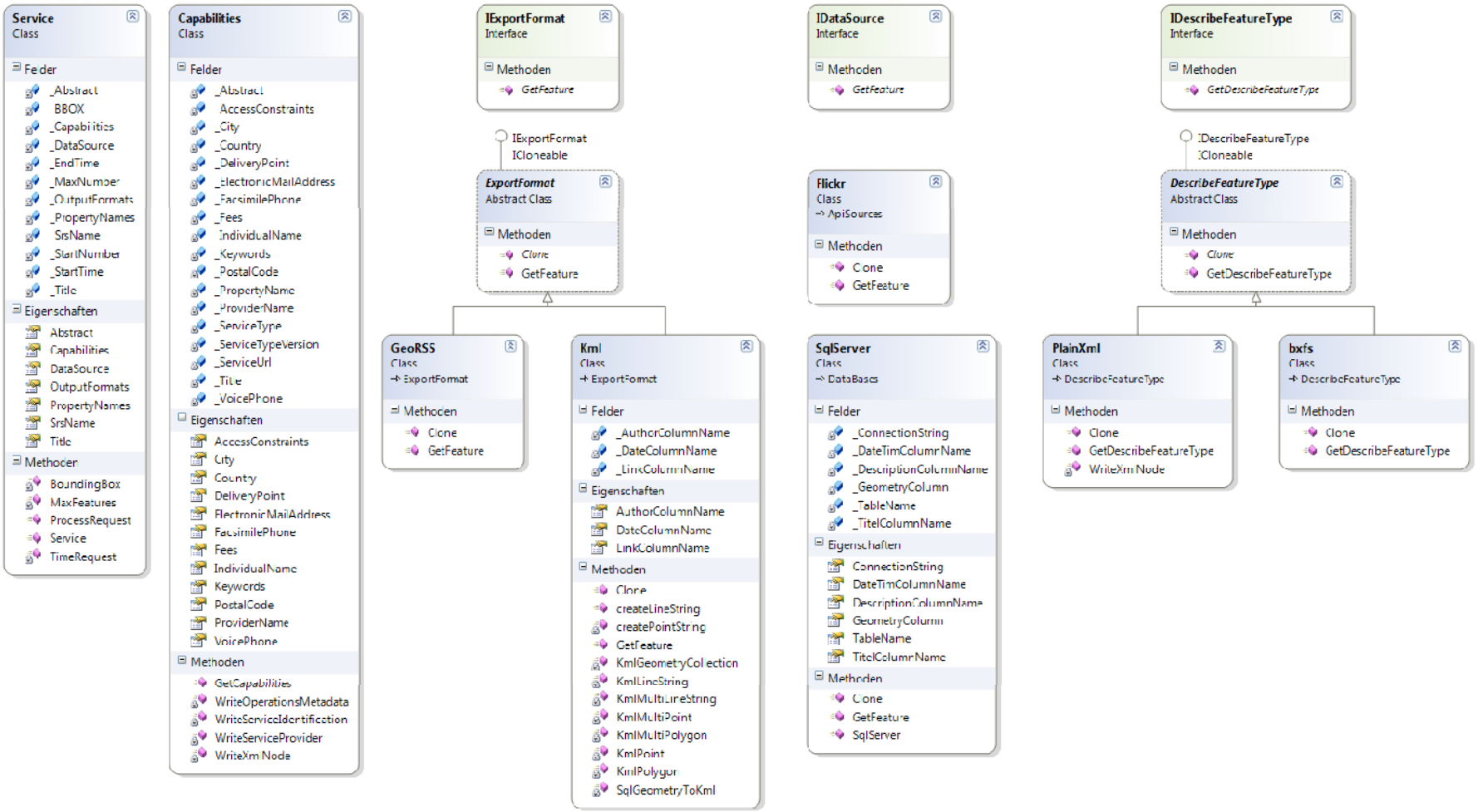
9 http://nextbike.de/m/maps.php

10 http://kml.nestoria.de/immobilien/kaufen/pirmasens
http://rss.nestoria.de/immobilien/kaufen/pirmasens

Name der Anwendung	Räuml. Filterung	SRS	Attribut. Filterung	Sonstige Parameter	Bemerkung
Vianovis Flash	Ja (BBOX als WKT)	GK4, kein Parameter			Nutzt amfphp
Vianovis html					Nutzt amfphp
Sh-Portal	Ja	GK3, kein Parameter	Ja, layers=…	Ja, z.B. mapScale	Basiert auf ArcIMS, identisch zu Felsinfo
Pegelonline	nein	GK3, kein Parameter	nein	nein	SOAP-Webservice verfügbar; zeitliche Filterung in Archiv möglich
GeländeDB	nein	WGS 84, kein Parameter	Ja		Bei HTML-Ausgabe: Paging möglich
Felsinfo	Ja	GK4, kein Parameter	Ja	Ja, z.B. mapScale	Basiert auf ArcIMS, identisch zu SH-Portal
Alpstein	Ja	Ja, EPSG 900913	Ja, SOURCEID=…	Ja, ähnlich WMS-Request	
Pfronten	Nein	WGS 84, kein Parameter	Ja, type=…		
nextbike	Nein	WGS 84, kein Parameter	Nein	Nein	
nestoria	Nein	WGS 84, kein Parameter	(Ja)	(Ja)	Filterung nach Ort und Miete/Kauf über REST-„Pfad“

Anhang B

Klassendiagramm


Service
Class
Felder
_Abstract
BBOX
_Capabilities
_DataSource
_EndTime
_MaxNumber
_OutputFormats
_PropertyNames
SrsName
_StartNumber
_StartTime
_Title
Eigenschaften
Abstract
Capabilities
DataSource
OutputFormats
PropertyNames
SrsName
Title
Methoden
BoundingBox
MaxFeatures
ProcessRequest
Service
TimeRequest
Capabilities
Class
Felder
_Abstract
AccessConstraints
_City
_Country
_DeliveryPoint
_ElectronicMailAddress
_FacsimilePhone
_Fees
_IndividualName
_Keywords
_PostalCode
_PropertyName
_ProviderName
_ServiceType
_ServiceTypeVersion
_ServiceUrl
_Title
_VoicePhone
Eigenschaften
AccessConstraints
City
Country
DeliveryPoint
ElectronicMailAddress
FacsimilePhone
Fees
IndividualName
Keywords
PostalCode
ProviderName
VoicePhone
Methoden
GetCapabilities
WriteOperationsMetadata
WriteServiceIdentification
WriteServiceProvider
WriteXmlNode
IExportFormat
Interface
Methoden
GetFeature
IExportFormat
ICloneable
ExportFormat
Abstract Class
Methoden
Clone
GetFeature
GeoRSS
Class
ExportFormat
Methoden
Clone
GetFeature
Kml
Class
ExportFormat
Felder
_AuthorColumnName
_DateColumnName
_LinkColumnName
Eigenschaften
AuthorColumnName
DateColumnName
LinkColumnName
Methoden
Clone
createLineString
createPointString
GetFeature
KmlGeometryCollection
KmlLineString
KmlMultiLineString
KmlMultiPoint
KmlMultiPolygon
KmlPoint
KmlPolygon
SqlGeometryToKml
IDataSource
Interface
Methoden
GetFeature
Flickr
Class
ApiSources
Methoden
Clone
GetFeature
SqlServer
Class
DataBases
Felder
_ConnectionString
_DateTimColumnName
_DescriptionColumnName
_GeometryColumn
_TableName
_TitelColumnName
Eigenschaften
ConnectionString
DateTimColumnName
DescriptionColumnName
GeometryColumn
TableName
TitelColumnName
Methoden
Clone
GetFeature
SqlServer
IDescribeFeatureType
Interface
Methoden
GetDescribeFeatureType
IDescribeFeatureType
ICloneable
DescribeFeatureType
Abstract Class
Methoden
Clone
GetDescribeFeatureType
PlainXml
Class
DescribeFeatureType
Methoden
Clone
GetDescribeFeatureType
WriteXmlNode
bxfs
Class
DescribeFeatureType
Methoden
Clone
GetDescribeFeatureType

Anhang C

Generischer Handler

```
<%@ WebHandler Language="C#" Class="wfssimple" %>

using System;
using System.Web;
using System.Xml;
using System.IO;
using System.Data.SqlClient;

public class wfssimple : IHttpHandler {

    public void ProcessRequest (HttpContext context)
        {
        WfsSimple.Service wfss = new WfsSimple.Service();
        wfss.Abstract = "Das ist die Beschreibung des Dienstes";
        wfss.Title = "Titel des Dienstes";

        // Capabilities festlegen
        WfsSimple.Capabilities.Capabilities cap = new
WfsSimple.Capabilities.Capabilities();

        cap.AccessConstraints = "";
        cap.City = "Regensburg";
        cap.Country = "Germany";
        cap.DeliveryPoint = "Straße Hausnummer 26";
        cap.ElectronicMailAddress = "mail@domain.de";
        cap.FacsimilePhone = "";
        cap.Fees = "";
        cap.IndividualName = "Vorname Nachnamr";
        cap.PostalCode = "01234";
        cap.ProviderName = "ProviderName";
        cap.VoicePhone = "+49(0)941-1234567";

        wfss.Capabilities = cap;
```

```
        // DataSource festlegen
        WfsSimple.DataSources.SqlServer dssql = new
WfsSimple.DataSources.SqlServer();
        dssql.ConnectionString =
@"server=localhost\sqlexpress;Trusted_Connection=yes;database=spatialdb";
        dssql.GeometryColumn = "geom";
        dssql.TableName = "roads";
        dssql.TitelColumnName = "title";
        dssql.DescriptionColumnName = "Description";

        // ... und zum Service hinzufügen
        wfss.DataSource = dssql;

        // Ausgabeformate festlegen
        wfss.OutputFormats.Add("Kml");
        wfss.OutputFormats.Add("geoRSS");

        wfss.ProcessRequest(context);
    }

    public bool IsReusable {
        get {
            return false;
        }
    }

}
```

Anhang D

Inhalt der CD-ROM

Der Ordner pdf enthält die vorliegende Masterthesis als PDF-Dokument. Die digitale Version kann farbige Grafiken enthalten, die in der Druckausgabe in Graustufen abgebildet sind.

Der Ordner WfsSimple enthält den Quellcode des Prototypen als Visual Studio 2005-Projektmappe.