

Hochschule Neubrandenburg
Studiengang Geoinformatik und Geodäsie

Entwicklung einer Web-basierten GIS-Umgebung für den Einsatz als Lernplattform

Masterarbeit
Zum Erlangen des akademischen Grades
Master of Engineering (M.Eng.)

Erstellt von: Daniel Vogel
URN: nbn:de:gbv:519-thesis 2013-0117-6

Erstprüfer: Prof. Dr.-Ing. Ernst Heil
Zweitprüfer: Prof. Dr.-Ing. Wehrenpfennig

Eidesstattliche Erklärung

Hiermit versichere ich, die vorliegende Masterarbeit ohne Hilfe Dritter und nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die aus den Quellen entnommen wurden, sind als solche kenntlich gemacht worden. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Neubrandenburg, den

Daniel Vogel

Kurzfassung:

Das Gebiet der Web-basierten Geoinformationssysteme ist seit der Verbreitung von Diensten wie Google Maps eines der bekanntesten Anwendungsgebiete der Geoinformatik und seit Jahren in schneller Weiterentwicklung begriffen. Ziel dieser Arbeit ist es, eine Lernplattform bestehend aus einer Reihe von aufeinander aufbauenden, dokumentierten Implementierungsbeispielen zu erstellen, die als Basis für die Lehre in diesem wichtigen Gebiet verwendet werden kann. Teil eins der schriftlichen Arbeit soll einen Überblick über bestehende Web-GIS Konzepte und Technologien geben, während Teil zwei die Lernplattform beschreibt und als Begleitmaterial für das Verständnis der enthaltenen Anwendungen dient.

Abstract:

The field of web-based GIS is one of the most widely known and used applications of geoinformatics ever since the introduction of Services like Google Maps. For many years there has been rapid development in this field. The goal of this thesis is to create a platform that serves as basis for learning and teaching in this important field. The platform is based on a number of documented implementation examples of web-gis components building on each other. The first part of the written document provides an overview of existing web-gis concepts and technologies while the second one serves as explanation of the platform and as a tutorial to facilitate understanding of the included applications.

Inhaltsverzeichnis

1 Überblick Web-GIS.....	1
1.1 Definition Web-GIS im Rahmen dieser Arbeit (Bezug zur GDI).....	1
1.2 Aufgabenbereiche der Software.....	2
1.2.1 (zentrale) Datenspeicherung und Abrufbarkeit.....	2
1.2.2 (standardisierte) Bereitstellung der Daten über das Web.....	2
1.2.3 Visualisierung und Interface zum Endnutzer im Client.....	3
1.2.4 Prozessierung von Geodaten.....	3
1.2.5 Metadatenverwaltung.....	4
1.2.6 Erstellung von Geodaten.....	4
1.3 verfügbare Softwarekomponenten und deren mögliche Aufgaben.....	5
1.3.1 Geodatenbanksysteme.....	6
1.3.2 OGC WMS / WMTS und TMS.....	7
1.3.3 OGC WFS / WFS-T /WFS-G und WCS/WCS-T.....	8
1.3.4 OGC WPS/WPCS.....	10
1.3.5 Desktop-GIS Systeme (Thick Clients).....	11
1.3.6 Browser-basierte Clients (Thin Clients).....	13
1.3.7 Web-GIS Frameworks	15
1.3.8 OGC CSW.....	16
1.3.9 Styled Layer Descriptor und Symbology Encoding	17
1.3.10 OGC Sensorstandards SOS,SPS, SensorML,O&M.....	18
1.3.11 OGC Security und DRM Standards WSS.....	19
1.4 verschiedene Web-GIS Szenarien.....	19
1.4.1 Desktop-GIS mit zentraler Datenhaltung.....	20
1.4.2 simple Einbindung einer Karte aus fremden Datenquellen.....	21
1.4.3 Individuelles Serverprogramm zur Verbindung mit eigenen Datenquellen.....	22
1.4.4 Web-GIS auf Basis von OGC konformen Webservices.....	23
1.4.5 Zusammenführung verschiedener verteilter Daten- und Dienstanbieter.....	25
2 Die Lernplattform.....	26
2.1 Konzept und Umgebung der LernPlattform.....	26
2.1 Beispieldatensätze.....	27
2.1.1 Vektordaten.....	27
2.1.2 Rasterdaten.....	28
2.1.3 Speicherort in der Lernplattform.....	29
2.2 Vorbereitung der Daten.....	29
2.2.1 Konvertierung der Shapefiles in eine Geodatenbank.....	29
2.2.2 Erstellung einer PostGIS Datenbank für die Vektordaten.....	32
2.2.3 Vorbereitung der Rasterdaten.....	34
2.2.4 Erzeugung von SLDs für die Gestaltung der Vektordaten.....	38
2.3 Bereitstellung der Daten über Webdienste.....	42
2.3.1 Bereitstellung mithilfe von Geoserver.....	42
2.3.2 Bereitstellung mithilfe von Mapserver.....	57
2.4 Web-basierter Zugriff auf die Daten.....	64
2.4.1 Einfache Karte mit WMS Daten.....	64
2.4.2 Karte mit mehreren Layern	65
2.4.3 Karte mit umfassenden Kontrollelementen.....	66
2.4.4 Vergleich und Einschätzung.....	66
2.4.5 Umsetzung in der Lernplattform.....	67
3 Fazit und Ausblick.....	67

1 Überblick Web-GIS

Der erste Teil dieser Arbeit dient dazu, einen allgemeinen Überblick über Aufgaben und Erscheinungsformen von Web-GIS zu geben und verschiedene aktuell verfügbare Softwarekomponenten zum Aufbau von Web-GIS zu beleuchten.

1.1 Definition Web-GIS im Rahmen dieser Arbeit (Bezug zur GDI)

Der Begriff Web-GIS ist soweit dem Autor bekannt nicht verbindlich und/oder allgemein akzeptiert definiert und kann üblicherweise sehr breit angewendet werden. Im Rahmen dieser Arbeit wird ein Web-GIS im Folgenden als ein Software-basiertes Geoinformationssystem gesehen, welches Internettechnologien verwendet und dynamische Inhalte verarbeiten kann.

Aus dem Begriff des Geoinformationssystems leiten sich neben den software-technischen noch verschiedene andere Aspekte wie die verarbeiteten Geodaten und die Modellierung der realen Geobjekte ab.

Diese Arbeit beschäftigt sich mit der Erstellung einer Lernplattform für den Einsatz in der Lehre des Gebietes der Geoinformatik und wird sich daher hauptsächlich auf den Aspekt der technischen Umsetzung von Web-GIS konzentrieren.

Ein Web-GIS wird üblicherweise als verteiltes GIS, dessen Komponenten sich also auf verschiedenen physischen Systemen befinden, betrieben. Dies ist jedoch kein zwingendes Merkmal, da sich unter Umständen, beispielsweise auf Entwicklungs- und Testsystemen, alle Komponenten auf derselben physischen oder virtuellen Maschine befinden können.

Das Thema des Web-GIS steht auch im Zusammenhang mit dem Begriff der Geodateninfrastruktur. Grundsätzlich kann hier gesagt werden, dass eine GDI üblicherweise ein Web-GIS beinhaltet bzw. verwendet, jedoch noch weitere technische und nicht-technische Anforderungen erfüllen muss. Die Schaffung von Geodateninfrastrukturen stellt hohe Anforderungen an die Interoperabilität der entsprechenden Web-GIS Komponenten des Systems, sodass eine Reihe von Standards und Normen durch gesetzliche und private Organisationen, wie ISO und OGC, geschaffen wurden, die die web-basierte gemeinsame Nutzung und Bearbeitung von Geodaten aus unterschiedlichen Quellen ermöglichen sollen. Diese Standards und darauf aufbauende Softwareprodukte können für die Erstellung verschiedener Web-GIS Projekte genutzt werden und werden im Folgenden wiederholt Erwähnung finden.

1.2 Aufgabenbereiche der Software

Dieser Abschnitt enthält eine Auflistung und Beschreibung von grundlegenden Aufgabenbereichen, die jedes Web-GIS erfüllen muss sowie optionale Aufgabenbereiche, die die Funktionalität des Web-GIS erweitern und beispielsweise für eine GDI notwendig sein können. Die folgende Kategorisierung ist dabei teilweise an [Steiniger&Hunter2012] angelehnt.

1.2.1 (zentrale) Datenspeicherung und Abrufbarkeit

Der Zugriff auf vorhandene Geodaten ist eine der Grundfunktionen eines Geoinformationssystems und muss vom Web-GIS umgesetzt werden. Dabei ist wie bei anderen Systemen eine konsistente, performante und gut rückzusichernde Lösung erstrebenswert. Im Fall von Geodaten besteht eine enge Verbindung zwischen dem geometrischen Datenteil, verschiedenen Metadaten wie z.B. das räumliche Bezugssystem oder die Genauigkeit, mit der die Daten erfasst wurden, und den Sachdaten bzw. Attributdaten der Geoobjekte. Diese Daten müssen daher wie bei anderen Geoinformationssystemen in fachlich geeigneter Weise abgelegt werden und sollten sich darüber hinaus möglichst gut über vorhandene Mittel in eine Web-Infrastruktur einbinden lassen.

Erlaubt das System den Endnutzern das Hinzufügen von Daten, stellt sich für die Datenspeicherung zusätzlich die Aufgabe, mehrere, ggf. sehr viele, konkurrierende Benutzereingaben in konsistenter Weise zu koordinieren.

1.2.2 (standardisierte) Bereitstellung der Daten über das Web

Die web-basierte Bereitstellung der gespeicherten Daten für den Client ist eine der definierenden Eigenschaften eines Web-GIS im Verhältnis zu anderen Geoinformationssystemen. Hierbei müssen die vorhandenen Daten aus den jeweiligen Speichern gelesen, in eine von Client nutzbare Struktur umgewandelt und an diesen übermittelt werden. Bei komplexeren Web-GIS und besonders im Falle von GDIs können die Ausgangsdaten auf verschiedenen physischen Systemen, die verschiedenen Anbietern bzw. Institutionen zugeordnet sind, in unterschiedlichen Formaten vorliegen. Dies macht eine standardkonforme Bereitstellung der Daten, die keine weiteren Anforderungen an die Koordination der Datenanbieter untereinander stellt und möglichst viele Datenformate und deren Inhalt repräsentieren kann, vorteilhaft.

Soll das System die Erstellung von Daten durch Endnutzer zulassen, ist ebenfalls ein Mechanismus zur Weiterleitung an und ggf. Umwandlung für das Datenspeicherungssystem notwendig.

1.2.3 Visualisierung und Interface zum Endnutzer im Client

Eine Schnittstelle für die Interaktion mit dem Nutzer zur Verfügung zu stellen ist ebenfalls eine der notwendigen Voraussetzungen für ein Geoinformationssystem. Der Client muss die Daten des übrigen Systems entgegennehmen und für den Nutzer visualisieren. Er muss dem Nutzer die Möglichkeit geben, die vorhandenen Daten zu organisieren und ggf. neue Anfragen des Nutzers entgegennehmen und an das übrige System weiterleiten, sodass diese über die Speicher- und Verteilungssysteme beantwortet werden können. Soll das Web-GIS dem Endnutzer die Möglichkeit bieten Daten zu erstellen, muss der Client dafür die notwendigen Werkzeuge zu Verfügung stellen und die erstellten Daten an die anderen Teile des Systems übergeben.

Die Besonderheit im Falle eines Web-GIS ist die Notwendigkeit, verschiedene separate Clients zu unterstützen, deren Fähigkeiten variieren können. Insbesondere wird zwischen thick Clients, Software mit eigenen, oftmals umfangreichen Fähigkeiten im Bereich der Geodatenverarbeitung, und thin Clients, üblicherweise normale Web Browser, unterschieden.

1.2.4 Prozessierung von Geodaten

Die Fähigkeit, Geodaten durch Operationen wie Prüfung topologischer Beziehungen, Erzeugung abgeleiteter Geometrien, Verschneidungen oder, im Fall von Rasterdaten, Map Algebra und Reklassifikation weiterzuverarbeiten, steht in vielen Desktop GIS Softwarelösungen in mehr oder weniger umfangreichem Ausmaß zur Verfügung. Web-GIS im Sinne der obigen Definition müssen solche Funktionen nicht zwingend beinhalten, können diese jedoch ebenfalls bieten.

Die Umsetzung von Prozessierungsaufgaben kann auf verschiedene Weise und in verschiedenen Teilen des Systems stattfinden. Die Entscheidung für eine bestimmte Lösung muss verschiedene Faktoren wie beispielsweise das Ausmaß der benötigten Funktionalität, den betroffenen Nutzerkreis und dessen Anforderungen, die Gesamtarchitektur des Systems und die Verteilung der ggf. erheblichen Anforderungen an Rechenleistung berücksichtigen. Ein System, dessen Nutzer Fachleute darstellen, die über leistungsstarke thick Clients verfügen, kann Prozessierungsaufgaben auf diese auslagern und somit umfangreiche Funktionalität erreichen ohne die Performance der Serverkomponente zu belasten. Ein Web-GIS, dessen Nutzer Laien sind, die typischerweise Web Browser, ggf. auf mobilen Endgeräten, nutzen, muss solche Aufgaben üblicherweise über Serverseitige Komponenten lösen und wird aus technischen und Gründen der Übersichtlichkeit für die Nutzer tendenziell nur ausgewählte Funktionen bieten. Eine GDI einer großen Institution kann wiederum die Nutzung einer eigenständigen standardkonformen Komponente für alle Prozessierungsaufgaben innerhalb des Systems vorschreiben.

Grundlegend müssen alle diese Komponenten die Anweisung, eine bestimmte Operation auszuführen und die benötigten Ausgangsdaten vom Nutzer bzw. den mit ihnen in Kontakt stehenden Systemkomponenten aufnehmen, die Operation durchführen und das Ergebnis zurückliefern können.

1.2.5 Metadatenverwaltung

Geodaten haben eine große Zahl an Eigenschaften, die in Form von Metadaten beschrieben werden und die für die Arbeit mit und anhand dieser Daten von entscheidender Bedeutung sein können. So ist es gerade bei Daten aus unterschiedlichen Quellen oft unvermeidlich, dass Daten beispielsweise in sehr unterschiedlicher Erfassungsgenauigkeit, Gebietsabdeckung und Aktualität vorliegen.

Die Verwaltung solcher Metadaten sowie die Aufbereitung und Darstellung dieser für den Endnutzer sind für viele Web-GIS, die nur eine oder wenige Datenquellen verwenden und/oder diese nur zur Unterstützung eines eng begrenzten Anwendungsbereiches wie z.B. der Fahrzeugnavigation einsetzen, meist nicht oder nur in sehr einfacher Weise notwendig. Hier kann die Eignung der gebotenen Daten im Vorfeld geprüft werden, was in vielen Anwendungsfällen auch den Erwartungen des Nutzers entspricht. Im Falle von GDIs und ähnlichen umfangreichen Anwendungen, in denen Daten vieler unterschiedlicher Quellen Nutzern mit ggf. sehr unterschiedlichen Anforderungen zugänglich gemacht werden sollen, stellt dieser Bereich hingegen eine grundlegende Anforderung an das Gesamtsystem dar, die durch eigene Softwarekomponenten umgesetzt werden muss.

Solche Komponenten müssen die Metadaten der Geodatenätze verwalten, darstellen und für den Nutzer abfragbar machen, sodass Datensätze anhand spezifischer Anforderungen gefunden werden können.

1.2.6 Erstellung von Geodaten

Die Erstellung von Geodaten durch das Web-GIS ist keine grundsätzlich notwendige Anforderung an ein solches, wobei Anwendungsfälle existieren, für die diese Aufgabe zentral ist. Systeme ohne Geodatenerstellung nutzen Daten, die von Quellen außerhalb desselben in das Speichersubsystem eingespeist wurden. Diese Unterscheidung beruht auf der entsprechenden Definition, welche Prozesse und Komponenten als Teil des Web-GIS angesehen werden und ist daher ggf. nicht eindeutig bzw. willkürlich.

Die Erstellung von Geodaten allgemein kann auf unterschiedliche Weisen erfolgen. Vereinfachend kann man zwischen der direkten Messung mithilfe von Messgeräten bzw. Sensoren und der

Generierung von Geodaten aus der Verarbeitung anderer Geodaten unterscheiden, wobei diese beiden Bereiche nicht immer klar trennbar sind.

Beispiele für vielen Nutzern leicht zugängliche Methoden der Geodatenerfassung sind die Nutzung von einfachen GPS bzw. GNSS Empfängern für Endnutzer sowie das Digitalisieren von Vektordaten, welches über entsprechende Komponenten auch auf Client Systemen verfügbar ist. In einem Anwendungsszenario mit sehr vielen Nutzern, die Daten erstellen können sollen wie z.B. das Open Street Map Projekt, muss das Web-GIS ein Interface für die Eingabe bzw. Erstellung der Daten zur Verfügung stellen, das auf möglichst weit verbreiteten Clients eingesetzt werden kann. Die von den Nutzern generierten Daten müssen durch die übrigen Komponenten des Systems weitergeleitet, gespeichert und verwaltet werden.

Beispiele für verhältnismäßig aufwendige Verfahren der messtechnischen Erfassung sind professionelle Vermessung durch Tachymetrie und High-End GNSS, Photogrammetrie, Laserscanning, Satellitenaufnahmen und spezialisierte Sensoren mit Positionsbestimmung. Solche Quellen stellen einen großen Teil der vorhandenen Geodaten zur Verfügung und werden aufgrund der hohen Kosten und Anforderungen üblicherweise von Institutionen und Unternehmen durch ausgebildete Spezialisten bereitgestellt. Ähnlich verhält es sich mit der Erzeugung von abgeleiteten Geodaten mithilfe komplexer Analyse und Prozessierungsmethoden. Diese erfordern neben spezieller Software vor allem umfangreiche fachliche Qualifikation und zeitliches Engagement der verantwortlichen Personen. Derartige Datenquellen können mithilfe geeigneter Komponenten direkt in das Web-GIS eingefügt werden, wobei tendenziell thick Clients und automatisierte Skripte zum Einsatz kommen. Es ist ebenfalls möglich dass die Kommunikation der Daten erstellenden Komponenten direkt mit dem Speichersystem erfolgt, wie beispielsweise durch die Anbindung eines Desktop-GIS an eine Datenbank.

1.3 verfügbare Softwarekomponenten und deren mögliche Aufgaben

Dieser Abschnitt stellt eine Reihe von Technologien und Softwarekomponenten vor, die aktuell für den Aufbau von Web-GIS zur Verfügung stehen. Dafür wird für jeden der Unterpunkte eine allgemeine Beschreibung gegeben, in der auch auf die möglichen Aufgabenbereiche und Einsatzgebiete der Komponente eingegangen wird, gefolgt von einer Auflistung und kurzen Vorstellung einiger konkreter Softwareprodukte in dem entsprechenden Bereich. Die Vorstellung von Produkten konzentriert sich dabei stark auf Open Source Systeme, die sich aufgrund ihrer Verfügbarkeit für alle Studierenden und der Möglichkeiten den Quellcode zu untersuchen für den Einsatz in der Lehre besonders eignen.

1.3.1 Geodatenbanksysteme

1.3.1.1 allgemein

Geodatenbanksysteme bzw. GDBS sind Datenbanksysteme, die die Fähigkeit haben, geometrische Objekte effizient zu speichern, abzufragen und räumliche Funktionen über diesen Daten auszuführen. Die Fähigkeiten der Systeme sind dabei individuell, wobei für Geodatenbanken eine Reihe von Normen und Standards existieren, die einen Teil der Funktionen moderner GDBS regeln. Besonders bedeutend sind hier die SFSQL und SQL/MM 3 [vgl. SFSQL11]. Viele häufig eingesetzte GDBS stellen räumliche Erweiterungen bestehender objekt-relationaler Datenbanksysteme dar.

Im Bezug auf Web-GIS Anwendungen werden Geodatenbanksysteme in erster Linie als Speichersysteme eingesetzt, wobei alle Vorteile der Nutzung von Datenbanken gegenüber anderen Formen der Speicherung zur Verfügung stehen.

Zusätzlich können die je nach konkretem Produkt sehr umfangreichen Verarbeitungsfunktionen auch für Prozessierungsaufgaben eingesetzt werden.

1.3.1.2 Produkte

- PostGIS - PostGIS ist eine räumliche Erweiterung der Open Source Datenbanksoftware PostgreSQL. Es unterstützt die Verarbeitung von traditionellen 2,5D sowie 3D Geometrien, Rasterdaten, Topologie, Linear Referencing und mithilfe der Erweiterung Pqrouting Netzwerk und Routing basierte Aufgaben [vgl. PG2].
- SpatiaLite - SpatiaLite ist eine räumliche Erweiterung der Open Source Datenbanksoftware SQLite. Es unterstützt die Verarbeitung von traditionellen 2,5D Daten, die Speicherung von Rasterdaten und Routing Aufgaben [vgl. SLite].
- Rasdaman - Rasdaman ist ein speziell für die Verarbeitung von sehr großen multidimensionalen Rasterdatensätzen entwickeltes Datenbankmanagementsystem. Es gibt derzeit die rasdaman community Variante, die als Open Source Produkt zur Verfügung steht und das proprietäre rasdaman enterprise, welches von der rasdaman GmbH vertrieben wird [vgl. Rasdaman].
- kommerzielle GDBMS - viele kommerzielle DBMS bieten, ggf. als separat zu lizenzierende Erweiterung, ebenfalls umfangreiche Funktionen zur Verwaltung und Prozessierung von Geodaten

1.3.2 OGC WMS / WMTS und TMS

1.3.2.1 allgemein

Die oben genannten Standards dienen der Erzeugung und Übermittlung von Karten. Karten stellen im Sinne der in den Standards verwendeten Definition visuelle Repräsentationen von Geodaten dar, die auf einem Computerbildschirm dargestellt werden können, dar. Es ist dabei hauptsächlich das Ziel, eine Graphik, üblicherweise in einem rasterbasierten Bildformat wie GIF, PNG oder JPEG oder einem vektorbasierten Format wie SVG oder webCGM, bereitzustellen, die mit minimalem Aufwand dargestellt und direkt betrachtet werden kann [vgl. WMSspec].

Der Web Map Service bzw. WMS ermöglicht es, Karten eines beliebigen Gebietes in einer beliebigen Projektion aus beliebig vielen Datenquellen zu erzeugen und zu veröffentlichen. Auf diese Weise können genau an die konkreten Anforderungen angepasste Karten erzeugt werden. Diese Flexibilität führt dazu, dass sich die generierten Karten sehr individuell unterscheiden und somit ein Zwischenspeichern bzw. Cachen dieser nicht praktikabel ist. Zusätzlich erfordert die individuelle Kartenerstellung relativ viel Rechenaufwand seitens des Servers, der den WMS bereitstellt.

Der Tile Map Service bzw. TMS basiert auf dem Konzept Karten aus vorgefertigten Kacheln zusammensetzen, wobei nur eine oder wenige Projektionen geboten werden, da für jede Projektion ein neuer Satz Kacheln bereitgehalten werden muss. Der TMS bietet aufgrund dieses Konzepts nur eine endliche Anzahl festgelegter Zoom- und damit Detailstufen. Diese Methode stellt jedoch eine Lösung für die möglichen Performance Probleme des WMS dar, da nur eine bestimmte Anzahl von vorgefertigten Kartenkacheln angeboten wird, müssen diese nur bei Aktualisierungen des Karteninhaltes erneut berechnet werden und können mit vertretbarem Aufwand zwischengespeichert werden. Diese Art der Kartenbereitstellung bietet sich vor allem bei Angeboten an, die über das Internet in sehr großem Umfang von sehr vielen Nutzern in Anspruch genommen werden, da hier die Belastung der Infrastruktur eine kritische Rolle spielt. Ein solches Projekt stellt beispielsweise OpenStreetMap dar.

Der TMS ist kein OGC Standard, sondern wurde von der Open Source Geospatial Foundation bzw. OSGeo entwickelt. Der Web Map Tile Service bzw. WMTS ist der offizielle OGC Standard, der einen Dienst zur Bereitstellung vorgefertigter Kartenkacheln regelt und soll den TMS ersetzen [vgl. WMTSpec].

Im Bezug auf Ihre Rolle in Web-GIS stellen diese Standards bzw. die Produkte, die sie implementieren, eine Möglichkeit zur standardisierten Bereitstellung von Daten dar, die weitgehende Verbreitung genießen.

1.3.2.2 Produkte

- Geoserver - Geoserver ist eine auf Java basierende Serveranwendung, die Geodaten aus verschiedenen Formaten aufnehmen und in Form OGC konformer Webstandards über das Web bereitstellt. Geoserver unterstützt Version 1.1.1 und 1.3.0 des Web Map Service, der Web Map Tile Service wird nicht direkt unterstützt, sondern über eine integrierte Variante von Geowebcache. Die Konfiguration kann vollständig über ein graphisches Interface durchgeführt werden [vgl. Gsmain].
- Mapserver - Mapserver ist eine auf C/C++ basierende Serveranwendung, die wie Geoserver Geodaten aus verschiedenen Quellen über das Web bereitstellt. Mapserver unterstützt ebenfalls verschiedene Versionen des WMS, der Web Map Tile Service wird über Mapcache, eine Anwendung aus der Mapserver Suite unterstützt. Die Konfiguration von Mapserver erfolgt über textbasierte Konfigurationsdateien, sogenannte Map-Files [vgl. umnMS].
- Deegree 3 - Das auf Java basierende Deegree Framework implementiert eine Reihe der Open Web Services bzw. OWS des OGC. Darin enthalten ist die Unterstützung von WMS in den Versionen 1.1.1 und 1.3.0 und des WMTS. Deegree verfügt über eine graphische Administrationskonsole, die in Kombination mit manuell zu editierenden XML-basierten Konfigurationsdateien der Konfiguration der Webservices dienen [vgl. deg3].
- Geowebcache - Geowebcache ist eine auf Java basierende Serveranwendung, die Kartenkacheln aus verschiedenen Quellen zwischenspeichern und über das Web weiterleiten kann. Auf diese Weise können Systeme, die WMS oder andere Kartendienste wie z.B. Google Maps nutzen, in ihrer Performance verbessert werden. Die Software ermöglicht den Aufbau eines TMS sowie WMTS [vgl. GWCACHE].
- Mapcache - Mapcache ist Teil der Mapserver Suite und wie Mapserver eine auf C/C++ basierende Serveranwendung. Mapcache erfüllt vergleichbare Aufgaben wie Geowebcache und unterstützt ebenfalls TMS und WMTS [vgl. umnMS].

1.3.3 OGC WFS / WFS-T / WFS-G und WCS/WCS-T

1.3.3.1 allgemein

Die oben genannten OGC Standards dienen dazu, Geodaten in einer standardisierten Weise über das Web bereitzustellen. Im Gegensatz zu den Standards der vorangegangenen Kategorie sollen hier die tatsächlichen Rohdaten übertragen werden. Diese können dann in Form von Karten visualisiert

oder in weiteren Prozessierungsschritten verarbeitet werden.

Der Web Feature Service bzw. WFS dient zur Übertragung von Vektordaten inklusive ihrer Attributdaten. Die Kodierung der Nutzdaten findet dabei durch GML statt.

Der Web Feature Service Transactional bzw. WFS-T ermöglicht die Manipulation der Geodaten in der Datenquelle des WFS-T. Der Dienst kann eingesetzt werden, wenn Nutzer die Möglichkeit haben sollen, über das Web-GIS Vektordaten zu erstellen, zu modifizieren oder zu löschen [vgl. WFSspec].

Der WFS-G soll einen sogenannten Gazetteer Service regeln, wobei es sich hier zum Zeitpunkt der Erstellung dieser Arbeit noch nicht um einen fertigen Standard handelt. Dieser Dienst dient der Bereitstellung von Informationen zu den geographischen Namen von Geoobjekten verknüpft mit deren Lage [vgl. WFS-Gspec].

Der Web Coverage Service bzw. WCS dient analog zum WFS der Übertragung von Geodaten, die bezogen auf den Raum kontinuierlich änderbare Phänomene darstellen, was üblicherweise den Anwendungsbereichen von Rasterdaten entspricht.

Die transaktionale Erweiterung des WCS, der WCS-T, ermöglicht analog zum WFS-T die Manipulation der mit dem WCS-T angebotenen Daten [vgl. WCSspec].

1.3.3.2 Produkte

- Geoserver - Geoserver bietet Unterstützung für WFS in den Versionen 1.0, 1.1 sowie 2.0, WFS-T und WCS in der Version 1.0 und 1.1.1 [vgl. GSdoc]
- Mapserver - Mapserver bietet Unterstützung für WFS in der Version 1.0 sowie 1.1, WCS in den Versionen 1.0, 1.1 sowie 2.0. WFS-T wird durch TinyOWS angeboten, ein ehemals eigenständiges Projekt, das aktuell Teil der Mapserver Suite ist. Bei der Nutzung von TinyOWS wird eine PostGIS Datenbank als Datenquelle benötigt. [vgl. MSdoc]
- Deegree 3 – Das Deegree 3 Framework bietet Unterstützung für WFS in den Versionen 1.0, 1.1 sowie 2.0. [vgl. deg3]
- Featureserver – Featureserver ist eine Serveranwendung auf Basis von Python, die ausschließlich dazu dient, Vektordaten über WFS und WFS-T bereitzustellen. [vgl. FeatServ]

1.3.4 OGC WPS/WPCS

1.3.4.1 allgemein

Die oben genannten Standards beschreiben Interfaces zur Einbindung von Prozessierungsdiensten für Geodaten in ein Web-GIS bzw. eine GDI.

Der Web Processing Service bzw. WPS definiert ein Interface für Dienste, die Vektordaten prozessieren. Der Standard regelt dabei nicht, welche Arten von Prozessierungsfunktionen angeboten werden [vgl. WPSspec].

Der Web Coverage Processing Service bzw. WCPS bezieht sich analog auf Dienste, die Rasterdaten prozessieren [vgl. WCPSspec].

1.3.4.2 Produkte

- Geoserver - Geoserver bietet über eine Erweiterung ebenfalls einen WPS Service an. Dieser unterstützt die aktuelle Version 1.0 des WPS und kann über Geoservers graphisches Interface konfiguriert werden. Zum Zeitpunkt des Schreibens können über diesen WPS die Funktionen der Java Topology Suite bzw. JTS, sowie Geoservereigene Prozesse genutzt werden [vgl. GSdoc].
- Deegree 3 - Der im Deegree 3 Framework integrierte WPS unterstützt die Version 1.0 des WPS und wird wie die anderen Deegree Webservices via XML Dateien konfiguriert. Die eigentlichen Prozessierungsfunktionen können modular als sogenannte Service Provider integriert werden und basieren immer auf Java Code [vgl. deg3].
- ZOO Project - Das ZOO Project ist eine Open Source Plattform für Geodatenprozessierung, die den WPS in der Version 1.0 unterstützt, wobei der ZOO Kern in der Sprache C erstellt wurde. Die räumlichen Prozessierungsalgorithmen, die über diesen Kern angeboten werden, können in den Sprachen C / C++, Fortran, Java, Python, PHP, Perl, JavaScript erstellt und gemeinsam als WPS zur Verfügung gestellt werden. Es existieren Beispielalgorithmen, die Funktionalität der Bibliotheken GDAL und OGR zur Verfügung stellen [vgl. ZOO].
- 52°North WPS - Der WPS der Forschungsgruppe 52°North ist eine Java basierte Serveranwendung für Geoprozessierung. Die Anwendung unterstützt vollständig den WPS Standard in der Version 1.0 und kann über eine graphische Weboberfläche konfiguriert werden. Derzeit besteht die Möglichkeit, Geoprozessierungsalgorithmen von GRASS GIS, SEXTANTE und R-Skripten mithilfe des Dienstes zu Nutzen [vgl. 52NWPS].

- PyWPS - PyWPS ist eine auf Python basierende Anwendung, die den WPS in der Version 1.0 unterstützt. Prozessierungsalgorithmen für PyWPS müssen in Python entwickelt werden. Derzeit besteht die Möglichkeit die Funktionalitäten aus GRASS GIS über den WPS zu nutzen [vgl. PyWPS].
- Rasdaman - Rasdaman ist die Referenzimplementierung des recht neuen WCPS Standards und unterstützt diesen in der Version 1.0. [vgl. Rasdaman].

1.3.5 Desktop-GIS Systeme (Thick Clients)

1.3.5.1 allgemein

Desktop-GIS Software Anwendungen sind darauf ausgerichtet, dem Nutzer allein ohne die Notwendigkeit für weitere Komponenten die Arbeit mit Geodaten zu ermöglichen. Sie bieten ein, üblicherweise graphisches, Interface für den Nutzer, können Geodaten visualisieren und editieren. Je nach Produkt bieten Desktop-GIS unterschiedlich ausgeprägte Funktionalität im Bezug auf räumliche Abfragen, Prozessierung von Geodaten und Erstellung und Design von Karten. Für die intensive, professionelle Arbeit im GIS-Bereich stellt diese Gruppe von Systemen die bewährteste Lösung dar und entsprechende Produkte im kommerziellen Bereich aber auch bei freier Software setzen derzeit die Maßstäbe für die Funktionalität, die von GIS Software erwartet werden kann. Im Zuge der Entwicklung von Web-GIS Technologien und Standards haben viele Desktop-GIS heute auch die Möglichkeit, diese zu Nutzen und in eine Web-GIS Struktur oder GDI eingebunden zu werden.

Im Kontext von Web-GIS werden Desktop-GIS oft als thick Clients eingesetzt. Als solche verfügen sie über umfangreiche Fähigkeiten und werden tendenziell von Nutzern eingesetzt, die über fortgeschrittene Kenntnisse und Anforderungen verfügen.

1.3.5.2 Produkte

- QGIS - QGIS oder QuantumGIS ist ein auf C++ und Python basierendes Desktop-GIS. Es bietet die Möglichkeit, eine große Zahl an Datenformaten inklusive GDBMS und Webservices zu verarbeiten. QGIS beinhaltet ein umfangreiches graphisches Nutzerinterface und enthält umfangreiche Funktionen zur Darstellung und Gestaltung von Geodaten. Das Programm bietet selbst nur wenige Funktionen zu Geoprocessing, lässt sich aber durch ein Plugin Interface ergänzen. Vorhandene Plugins stellen unter anderem Funktionen aus GRASS GIS, SAGA GIS, GDAL und R-Skripten zur Verfügung [vgl. QGIS].

- GRASS GIS - GRASS steht für "Geographic Resources Analysis Support System" und ist eines der bzw. das älteste Open Source GIS Projekt, das heute noch eingesetzt wird. Das Nutzerinterface ist an das Konzept der Unix-Shell angelehnt, mit der es auch einfach in Shellskripten integriert werden kann. Neben der Version für die Kommandozeile gibt es inzwischen auch ein graphisches Frontend, wobei das Konzept einzelner Module ähnlich der Unix Kommandos beibehalten wird. GRASS umfasst eine große Zahl an Modulen für Geoprocessing, die viele wissenschaftliche Fragestellungen abdecken [vgl. GRASS].
- SAGA GIS - SAGA steht für "System for Automated Geoscientific Analyses" und stellt ein auf C++ basierendes GIS mit Fokus auf wissenschaftlichen Analysen von Geodaten dar. Das Programm bietet eine kompakte graphische Benutzeroberfläche und eine große Zahl von Geoprocessingfunktionen, wobei der Schwerpunkt stärker im Bereich der Verarbeitung von Rasterdaten liegt. Der Großteil der Funktionalität von SAGA kann ebenfalls über ein kommandozeilenbasiertes Interface genutzt und somit als Bestandteil automatisierter Skripte eingesetzt werden. Nutzer können zusätzliche Module anhand einer C++ API entwickeln und einbinden. Im Bezug auf Kartenerstellung und Vektordatenerfassung bietet die Software nur grundlegende Funktionen. Der Großteil der Prozessierungsmodule von SAGA kann über die Java-basierte SEXTANTE Erweiterung auch von anderen GIS-Anwendungen genutzt werden [vgl. SAGA].
- GvSig - GvSig steht für "Generalitat Valencia Sistema de Información Geográfica". Es handelt sich dabei um ursprünglich von der spanischen Provinz Valencia finanziertes Open Source Projekt. Die Software hat einen ähnlichen Fokus wie QGIS, basiert anders als dieses auf Java und bietet eine andere Benutzeroberfläche. Die Prozessierungsfähigkeiten von GvSig können durch Erweiterungen, insbesondere SEXTANTE vergrößert werden. Erweiterungen für das GIS werden grundsätzlich in Java entwickelt [vgl. GvSig].
- uDig - uDig steht für "User-friendly Desktop Internet GIS" und stellt ein auf Java und insbesondere der Eclipse Plattform basierendes freies GIS dar. Das Programm bietet ähnlich anderen Desktop-GIS eine graphische Umgebung zur Visualisierung und Editierung von Geodaten. Der Fokus des Projekts liegt auf der Integration von Internetstandards, besonders WMS und WFS, sowie der Erweiterbarkeit durch das Pluginsystem der Eclipse Plattform [vgl. uDig].
- OpenJump - OpenJump ist ein auf Java basierendes freies GIS. Es bietet die Möglichkeit Geodaten verschiedener Formate zu öffnen und zu betrachten, wobei der Fokus auf

Vektordaten liegt. Die Stärken der Software liegen in den Bereichen der Geodateneditierung und -erzeugung sowie der Kartenerstellung und Topologieanalyse [vgl. OJump].

1.3.6 Browser-basierte Clients (Thin Clients)

1.3.6.1 allgemein

Webbrowser stellen ein nahezu auf allen Endgeräten vorhandenes Softwareprodukt, was sie zu besonders geeigneten Kandidaten als Clients für Web-Dienste, die ein möglichst großes Publikum erreichen sollen, macht. Die nativen Fähigkeiten von Browsern ermöglichen außerdem den einfachen Umgang mit nicht auf GIS bezogenen Teilen einer Internetpräsenz. Da Webbrowser ursprünglich in keiner Weise als GIS-Software entwickelt oder konzipiert wurden, sind relativ umfassende Erweiterungen nötig, um selbst grundlegende Funktionen zu ermöglichen.

Im Kontext von Web-GIS werden Webbrowser als thin-Clients eingesetzt, die nur grundlegende Funktionen, insbesondere die Visualisierung der Daten ein einfaches Nutzerinterface und ggf. einfache Editierfunktionen, bieten. Sie zielen auf Nutzer, die keine umfassenden Anforderungen im Bezug auf die Verarbeitung von Geodaten haben. Ein beispielhaftes Anwendungsgebiet ist die Einbindung von zweckbezogenen Karten in eine Website. Im Bezug auf die zukünftige Entwicklung ist denkbar, dass durch die Weiterentwicklung von client- und serverbasierten Komponenten Web-GIS und insbesondere GDI Funktionalitäten vergleichbar mit High-End Desktop-GIS anbieten können. Dabei ist nach Ansicht des Autors zu erwarten, dass Webbrowser, sofern nicht durch spezielle Clients ersetzt, weiterhin hauptsächlich auf die oben genannten Aufgabenfelder beschränkt bleiben und alle weiterführenden Aktionen durch Web-Dienste "in der Cloud" ausgeführt werden.

1.3.6.2 Produkte

- Proprietäre Web-Mapping APIs – Proprietäre APIs wie die von Google und Bing ermöglichen den Zugriff auf bestimmte Geodienste, wie z.B. auf Kacheln basierte Karten des entsprechenden Anbieters über einen Webbrowser. Die APIs erlauben neben dem Zugriff auf die Angebote des anbietenden Dienstleisters i.d.R. auch in begrenztem Umfang die Kombination mit eigenen Daten in festgelegten Formaten wie z.B. KML. Eine gemeinsame Nutzung von Inhalten des API Anbieters mit Quellen anderer Anbieter oder OWS ist i.d.R. nicht vorgesehen. Die bekanntesten Angebote wie Google Maps basieren auf der Seite des Clients vollständig auf Javascript, sodass der benötigte Code vom Browser direkt mit der entsprechenden Webseite geladen werden kann und somit auf der Seite des Nutzers keine Installation von zusätzlicher Software oder sonstige bewusste Schritte notwendig sind.

- OpenLayers - OpenLayers ist eine freie Javascript Bibliothek, die es ermöglicht, mithilfe eines Webbrowsers Geodaten einzulesen und zu visualisieren sowie ein Nutzerinterface für die Arbeit mit und Erstellung von Geodaten zu erstellen. Grundsätzlich ähnelt OpenLayers in vielen Aspekten einer der bekannten proprietären Web-Mapping APIs, das Ziel des Projektes ist es jedoch eine von einzelnen Anbietern unabhängige API zur Verfügung zu stellen. Das Projekt ermöglicht die Nutzung von proprietären APIs, ohne deren Syntax beherrschen zu müssen, die Nutzung verschiedener Datenformate wie KML, GeoRSS, GML oder WKT sowie von Webdiensten wie TMS, WMS, WFS und WPS. Die verschiedenen Ressourcen können gemeinsam in einer Karte genutzt bzw. für den Nutzer als Alternative angeboten werden. OpenLayers wird im Open Source Umfeld sehr häufig verwendet und ist integrierter Bestandteil vieler anderer Projekte wie z. B. Geoserver, das es für die Voransicht von veröffentlichten Layern zur Verfügung stellt, und des Großteils der unter 1.3.7 vorgestellten Web-GIS Frameworks [vgl. OL].
 - Geoext - Geoext ist eine Javascript Bibliothek, die OpenLayers mit dem ExtJS Javascript Framework verbindet. ExtJS dient dazu Webanwendungen zu entwickeln, die sich in der Bedienung und dem allgemeinen Nutzungserlebnis an Desktop-Anwendungen orientiert [vgl. GeoExt und Extinfo].
 - Mapquery - Mapquery ist ein Plugin für die JQuery Bibliothek, das die Funktionalität dieser mit OpenLayers verbindet. JQuery ist eine sehr weit verbreitete freie Javascript Bibliothek, die die Entwicklung von Javascript Inhalten erleichtern soll [vgl. MapQ und JQinfo].
 - Legato - Legato ist eine Javascript Bibliothek, die auf OpenLayers aufsetzt. Ziel des Projektes ist es vor allem, die Konfiguration der Webanwendung über XML Dateien zu ermöglichen und somit den Programmcode und Konfigurationsdaten zu trennen. Dadurch soll eine einfachere Erstellung und Wartung von OpenLayers basierten Anwendungen möglich werden [vgl. Legato].
- Leaflet - Leaflet ist wie OpenLayers eine Javascript Bibliothek, die einen Webbrowser um notwendige Funktionen eines GIS-Clients erweitern soll. Ziele des Projekts sind einfache Anwendbarkeit und gute Performance. Ein sehr breiter Funktionsumfang, wie OpenLayers ihn bietet, ist dagegen kein primäres Ziel. Zusätzliche Funktionen können durch ein Plugin System integriert werden [vgl. Leaflet].

- OpenScales - Openscales ist ein auf Flex und Actionscript basierendes Framework, das wie OpenLayers einen Web-GIS Client auf Basis eines Webbrowsers ermöglichen soll. Somit werden die zusätzlichen Funktionen nicht über die Javascript Komponenten des Browsers, sondern mithilfe des Flash Plugins ausgeführt. OpenScales ging aus dem Projekt FlexLayers hervor, das wiederum eine Portierung von OpenLayers nach Flex darstellt. Insofern sind Ähnlichkeiten in Funktionsumfang und Nutzung zwischen OpenScales und OpenLayers zu erwarten, wobei die beiden Projekte jedoch nicht direkt in Kontakt stehen [vgl. OScale].

1.3.7 Web-GIS Frameworks

1.3.7.1 allgemein

Die Projekte dieser Kategorie verbinden eigenen Code sowie verschiedene andere Softwarekomponenten, sowohl auf Geodaten zielende als auch allgemeine Bibliotheken und Frameworks, um dem Nutzer die Möglichkeit zu geben, komplexere Web-GIS Strukturen mit vergleichsweise niedrigerem Aufwand und in hoher Qualität zu erstellen. Im Rahmen dieses Ansatzes gibt es Projekte mit unterschiedlichen Zielen, die jeweils auf bestimmte Technologien und Komponenten ausgerichtet sind, sodass die Wahl eines Web-GIS Frameworks in besonders starkem Maße von den zu erreichenden Zielen und der damit verbundenen Systemumgebung abhängt.

1.3.7.2 Produkte

- Mapbender - Mapbender ist ein Framework zur Generierung von Web-GIS Portalen, die als Datenquellen Webservices verwenden. Es basiert auf PHP, Javascript, insbesondere der Mapquery Bibliothek, und einer PostgreSQL/PostGIS Datenbank, wobei frühere Versionen auch MySQL verwenden konnten. Mapbender ermöglicht die gesamte Konfiguration der Webservices und die Erstellung einer Portalanwendung über eine graphische Weboberfläche durchzuführen [vgl. Mbend].
- GeoMoose - Geomoose ist ein Framework zur Erstellung von Web-GIS Anwendungen. Es basiert auf HTML, Javascript, insbesondere OpenLayers und dem Dojo Toolkit, und optionalen PHP Skripten. Ziel des Geomoose Projektes ist es, ein für den Nutzer besonders unkompliziert einsetzbares System mit guter Performance zu bieten. Dabei soll es möglich sein, ein umfangreiches Web-GIS Portal ohne Programmierkenntnisse durch Editieren einer Konfigurationsdatei – des Mapbooks - zu erzeugen, Nutzern mit Programmierkenntnissen jedoch trotzdem gute Möglichkeiten für die Erweiterung der Anwendung zu bieten [vgl. GMoose].

- Geomajas - Geomajas ist ein Framework zu Erstellung von Web-GIS Anwendungen. Es basiert vollständig auf Java und dem Google Web Toolkit und integriert für den Umgang mit Geodaten die Bibliothek Geotools. Geomajas ist auf eine Client Server Architektur ausgerichtet, bei der möglichst viel der Funktionalität auf der Seite des Servers stattfindet. In diesem Zusammenhang verwendet das Projekt nicht einen der verbreiteten OpenLayers basierten Clients, die verhältnismäßig viel Funktionalität durch den Webbrowser abdecken [vgl. GMajas].
- MapFish - Mapfish ist ein Framework zu Erstellung von Web-GIS Anwendungen. Es basiert auf Python und dem Pylons Framework und integriert für den Umgang mit Geodaten unter anderem GeoExt für die Client Funktionen, die Shapely Bibliothek für Geoprozessierung und GeoAlchemy für die Anbindung verschiedener GDBMS [vgl. MFish].
- Cartaro - Cartaro ist eine Erweiterung des Content Management Systems Drupal, das PostgreSQL/PostGIS, Geoserver und OpenLayers integriert. Das Projekt ermöglicht, Geodaten als einen speziellen Datentyp zusammen mit anderen Inhalten im CMS Drupal zu verwalten und zu veröffentlichen [vgl. Cartaro].

1.3.8 OGC CSW

1.3.8.1 allgemein

Der Catalog Service for the Web bzw. CWS ist ein Standard für Metadatendienste. Diese dienen der Metadatenverwaltung und stellen Metadaten über Geodaten und Geodienste bereit. Solche Dienste können in Web-GIS, die Nutzern Zugang zu verschiedenen Datenquellen und Diensten bieten sollen, eingesetzt werden. CSW sind insbesondere beim Aufbau von GDI von Bedeutung [vgl. CSWspec].

1.3.8.2 Produkte

- Geonetwork - Geonetwork ist eine auf Java basierende Anwendung, die einen Metadatendienst realisiert. Die Software unterstützt den CSW Standard in der Version 2.0.2 sowie eine Reihe von Metadatenstandards. Geonetwork stellt ein Geoportal bereit, wofür andere Komponenten wie Geoserver und OpenLayers integriert werden [vgl. GeoNet].
- pycsw - pycsw ist eine auf Python basierende Anwendung, die einen Metadatendienst realisiert. Die Software unterstützt ebenfalls den CSW Standard in der Version 2.0.2, ist aber nicht auf die enge Integration mit bestimmten anderen Komponenten zu einer Portallösung ausgerichtet [vgl. pycsw].

1.3.9 Styled Layer Descriptor und Symbology Encoding

1.3.9.1 allgemein

Die beiden genannten Standards regeln die Beschreibung und Übermittlung von Darstellungsanweisungen für Vektor- und Rasterdaten. Die beiden Standards sind das Produkt der Teilung des ursprünglichen Styled Layer Descriptor Standards, weswegen viele Softwareprodukte im Bezug auf die Darstellung bzw. das Styling von Daten nur von Styled Layer Descriptor bzw. SLD sprechen [vgl. SLDspec].

Das Symbology Encoding bzw. SE definiert eine auf XML basierende Auszeichnungssprache zur Beschreibung von Darstellungsanweisungen für Geodaten [vgl. SEspec].

Der aktuelle Standard Styled Layer Descriptor regelt die Einbindung von individuellen Darstellungsanweisungen im SE in WMS Dienste.

Da der Großteil der Geodaten, mit Ausnahme bestimmter Rasterdaten wie Photographien, keine natürlich festgelegten für den Menschen wahrnehmbaren Eigenschaften haben, basiert jede Darstellung dieser Daten auf bewusst festgelegten Darstellungsfestlegungen. So können z.B. aus einem einzelnen Geodatensatz sehr unterschiedliche Kartendarstellungen generiert werden, deren Informationswert im Bezug auf den Betrachter und die für diesen interessanten Sachverhalte ebenfalls stark unterschiedlich sein kann. Die standardisierte Beschreibung dieser Festlegungen zur Darstellung ermöglicht im Kontext von Web-GIS die Wiedergabe der beabsichtigten Informationen auch im Zusammenspiel mit mehreren Komponenten unterschiedlicher Hersteller.

1.3.9.2 Produkte

- Desktop-GIS - Verschiedene Desktop-GIS bieten inzwischen die Möglichkeit, zusätzlich zu ihren nativen Formaten standardisierte Styling Anweisungen bzw. Legenden zu lesen und zu erzeugen. Beispiele hierfür sind GvSig, uDig, QGIS und OpenJump. Diese Methode der Erstellung ist besonders vorteilhaft, wenn manuell geeignete Darstellungsmöglichkeiten entwickelt und gestaltet werden sollen, da ein direktes und einfaches visuelles Feedback möglich ist.
- WebMapping Dienste - WMS sind für die Kartenerzeugung grundsätzlich auf die Anweisungen von SLD und SE angewiesen und unterstützen die Standards grundsätzlich sehr gut. Beispiele sind Geoserver, Deegree und Mapserver.
- Browserclients - OpenLayers als einer der am meisten verwendeten Browserclients für Geodaten unterstützt die Anwendung von SLD für die Darstellung von Rohdaten etwa von WFS.

1.3.10 OGC Sensorstandards SOS,SPS, SensorML,O&M

1.3.10.1 allgemein

Die genannten Standards sind Bestandteil des OGC Sensor Web Enablement Framework bzw. SWE. Dieses Projekt dient dazu, Standards für die Web-basierte Abfrage von Sensordaten zu schaffen.

Der Sensor Observation Service bzw. SOS dient dazu Daten über die Eigenschaften eines Sensors sowie die eigentlichen Messergebnisse abzurufen [vgl. SOSspec].

SensorML ist eine auf XML basierende Definitionssprache für die Beschreibung von Sensoren und ihren Eigenschaften und Parametern. Ein Sensor im Sinne der Standards können unterschiedliche Vorrichtungen sein. Beispiele dafür sind Satellitenkameras, aber auch ein einfache Webcam [vgl. SensorMLspec].

Observations and Measurements - XML bzw. O&M ist eine auf XML basierende Auszeichnungssprache, die der Beschreibung der eigentlichen Messergebnisse von Sensoren dient [vgl. O&Mspec].

Der Sensor Planning Service bzw. SPS dient der Steuerung von Sensoren und der Einstellung ihrer Parameter über einen Webservice [vgl. SPSspec].

1.3.10.2 Produkte

- 52°North SOS - Der SOS Server von 52° North unterstützt die Version 2.0 des Standards und ermöglicht die Bereitstellung von Sensormessungen und Eigenschaften in den vorgesehenen Standards O&M sowie SensorML. Darüber hinaus werden verschiedene SOS Client angeboten, unter anderem ein Web-Client und eine Erweiterung für uDig [vgl. 52NSensor].
- 52° North SPS - Der SPS Server von 52° North ist eine der ersten Implementierungen des SPS in freier Software und unterstützt den Standard in der Version 1.0 und 2.0 [vgl. 52NSensor].
- Mapserver - Mapserver enthält einen SOS Server, der jedoch noch auf einem Diskussionspapier des Standards beruht und daher nicht mit der aktuellen Version 2.0 kompatibel ist [vgl. MSdoc].

1.3.11 OGC Security und DRM Standards WSS

1.3.11.1 allgemein

Wie andere Arten von Daten, wie z.B. digitale Medien oder Software, stellen Geodaten einen wirtschaftlichen Wert dar und benötigen ggf. erheblichen Aufwand für ihre Erfassung. Gleichzeitig sind nicht alle Datensätze für eine öffentliche Verbreitung vorgesehen, weder kommerziell noch unentgeltlich, sondern sollen nur bestimmten Personen und Institutionen zugänglich sein. Um also eine auf Rechten und Lizenzen basierte Zugangskontrolle zu gewährleisten, können derzeit individuelle Lösungen entwickelt sowie teilweise auf bestehende Standards wie ISO 21000 oder ISO-REL zurückgegriffen werden. Ein Standard, der ein solches System regelt und auf die Besonderheiten von Geodaten (insbesondere von Web-basierten Geodaten, die durch Dienste wie WMS erst durch Anfrage erzeugt werden) eingeht, existiert bisher nicht. Die OGC hat bisher in diesem Kontext ein Referenzmodell für die Umsetzung solcher Dienste entwickelt, dieses stellt jedoch noch keinen Standard dar. [vgl. GeoDRMSpec]

1.3.11.2 Produkte

- 52° North WSS/WAS/WSC - Die hier genannten Dienste stellen eine Implementierung eines Zugangskontrolldienstes nach dem GeoDRM Referenzmodell dar. Der WSS bzw. Web Security Service stellt eine Java basierte Serveranwendung dar, die eine Zugangskontrolle zu WMS, WFS, WPS und SOS ermöglicht. Die Software kann über ein graphisches Web-Interface konfiguriert werden. Der WAS bzw. Web Authentication Service dient dazu, den Nutzer über verschiedene Methoden gegenüber dem WSS auszuweisen. Der WSC bzw. Web Security Client steht zwischen üblichen Clients für Geodaten wie Webbrowsern oder Desktop-GIS und dem WSS, sodass diese Clients mit dem Zugangskontrolldienst interagieren können, ohne selbst das WSS Protokoll beherrschen zu müssen [vgl. 52NDRM].

1.4 verschiedene Web-GIS Szenarien

Dieser Abschnitt stellt einige verschiedene Szenarien für Web-GIS Konfigurationen dar. Ziel ist es, dabei grundlegende Konzepte und charakteristische Anwendungsschwerpunkte darzustellen. Grundsätzlich existieren, schon aufgrund der weitgefassten Anwendungsgebiete, sehr viele unterschiedliche Möglichkeiten Web-GIS zu erstellen, die ggf. auch wesentlich komplexer sein können als die im Folgenden dargestellten Beispiele. Die hier dargestellten Konzepte können wiederum variiert oder als Bestandteil größerer Systeme umgesetzt werden.

1.4.1 Desktop-GIS mit zentraler Datenhaltung

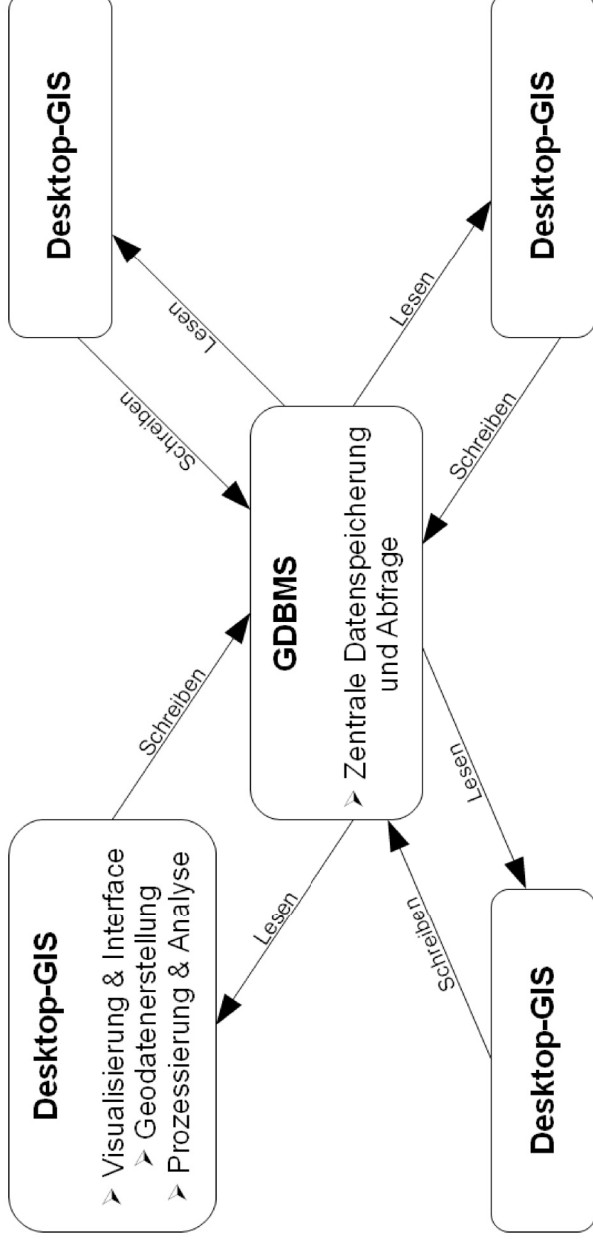


Abbildung 1: Schema für zentrale Datenbank mit Desktop-GIS Clients

Das Szenario beschreibt die vornehmliche Verarbeitung von Geodaten über verschiedene Desktop-GIS, die ihre Daten von einer gemeinsamen, zentralen Datenbank beziehen und ihre Ergebnisse in diese schreiben. Diese einfache Konfiguration eines Internet gestützten GIS eignet sich besonders zur Koordination der Arbeit in einer Institution oder einem Unternehmen, wobei nur eine überschaubare Anzahl an Mitarbeitern das System nutzt und eine klare Regelung der Zugriffsrechte und Verantwortungsbereiche existiert.

Im Zusammenhang mit einem Web-GIS, das einen größeren und ggf. anonymen Nutzerkreis erreichen soll, kann diese Konfiguration als ein Bestandteil zur Wartung und Generierung von Daten im Speichersystem durch den Betreiber des Systems eingesetzt werden.

1.4.2 simple Einbindung einer Karte aus fremden Datenquellen

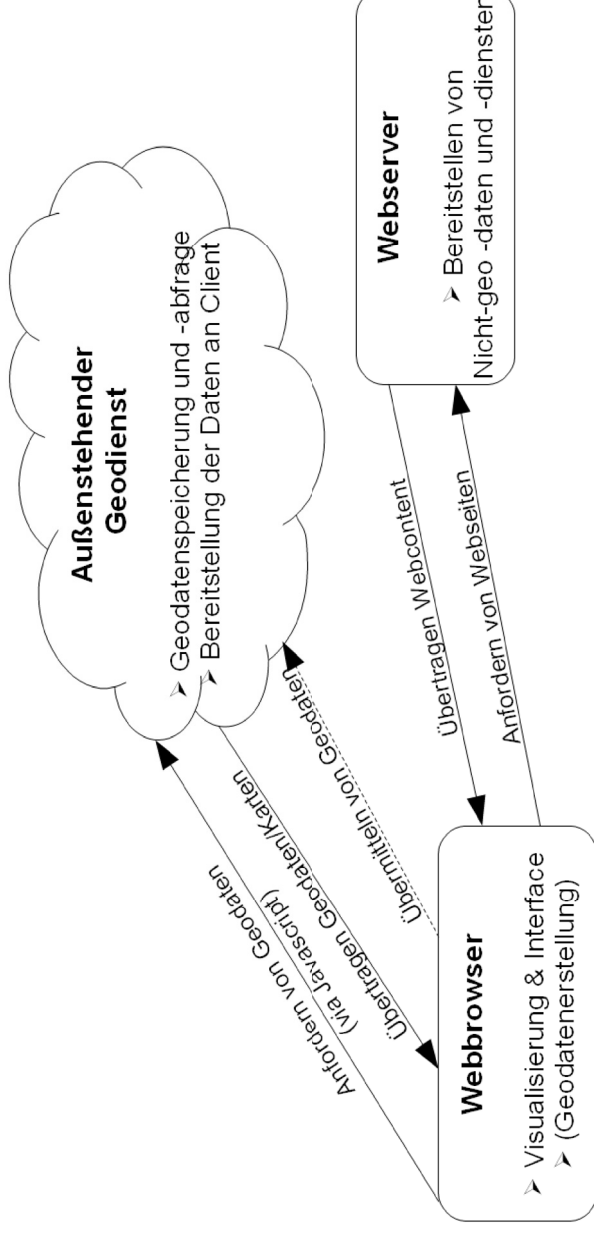


Abbildung 2: Schema für Verwendung fremder Kartendienste

Dieses Szenario beschreibt die Nutzung eines aus der Sicht des Anbieters einer Website fremden Geodienstes, wie sie beispielsweise bei der Einbindung einer Karte über die Google Maps API oder die Nutzung von Open Street Map über OpenLayers stattfindet. Der Browser des Nutzers erhält beim Laden der Webseite den Nicht-Geodaten bezogenen Inhalt sowie Javascript Code, der bei der Ausführung Kontakt zum Geodienstanbieter aufnimmt. Dieser erstellt dann mithilfe von Javascript und AJAX eine dynamische Karte, mit der der Nutzer ohne weiteren Kontakt zur ursprünglichen Website interagieren kann. Sofern der angebotene Geodienst auch das Erstellen oder editieren der Geodaten zulässt, wird auch diese Funktion über die Schnittstelle zwischen Browser und Geodienstanbieter abgewickelt.

Die Daten- und Prozessstruktur, die hinter dem Geodienstanbieter steht und letztlich ein eigenständiges, sehr umfangreiches Web-GIS darstellen kann, ist für den Nutzer der API grundsätzlich transparent, stellt also gewissermaßen eine Black Box dar. Dies wiederum bedeutet, dass somit die Einbindung relativ komplexer räumlicher Funktionen mit relativ geringem Aufwand für den Anbieter der Webseite möglich ist und erklärt die große Verbreitung dieser Konfiguration.

Eine Interaktion mit Teilen dieser Struktur über andere Wege, wie bei verteilten GIS auf Basis von OWS, ist grundsätzlich nicht vorgesehen. Dabei können natürlich auf OWS basierte Systeme so konfiguriert werden, dass der Zugang nur über bestimmte Interfaces möglich ist.

1.4.3 Individuelles Serverprogramm zur Verbindung mit eigenen Datenquellen

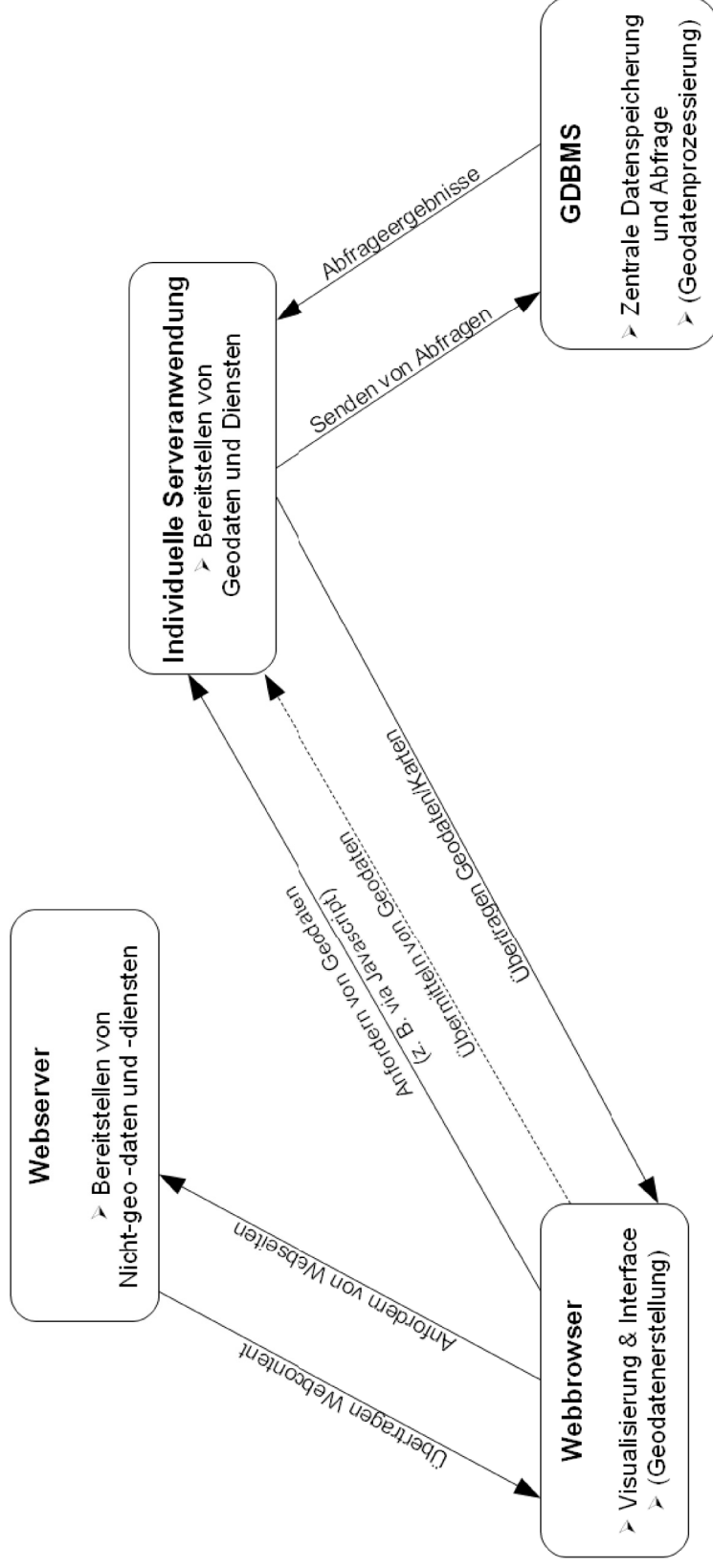


Abbildung 3: Schema für Web-GIS über individuelle Serverkomponente

Dieses Szenario stellt die Nutzung einer individuellen, Serveranwendung (z.B. in PHP, Python etc.) dar, die die angeforderten Geodaten zur Verfügung stellt. Ein Beispiel für diese Konfiguration ist das Senden von Vektordaten von der Serveranwendung z.B. als WKT oder GeoJSON und die Darstellung durch OpenLayers. Auf diese Weise ist auch das Erfassen von Daten und das Weiterleiten von Prozessierungsanfragen möglich. Die Geoprozessierung von Daten kann in diesem Fall außer durch die Anwendung direkt auch relativ einfach über das GDBMS erfolgen, da das Senden von SQL-Abfragen und das Entgegennehmen von Datensätzen aus der Datenbank Funktionen sind, die von allen gebräuchlichen Programmiersprachen über gut verfügbare Bibliotheken unterstützt werden.

Die gesamte Konfiguration ist der vorangegangenen weitgehend ähnlich mit dem Unterschied, dass die auf Geodaten bezogenen Aufgaben hier direkt vom Betreiber der Konfiguration gehandhabt werden. Eine Kombination mit fremden Diensten und insbesondere OGC konformen Diensten ist ebenfalls möglich und kann z.B. Kartenmaterial als Hintergrund zur Verfügung stellen.

1.4.4 Web-GIS auf Basis von OGC konformen Webservices

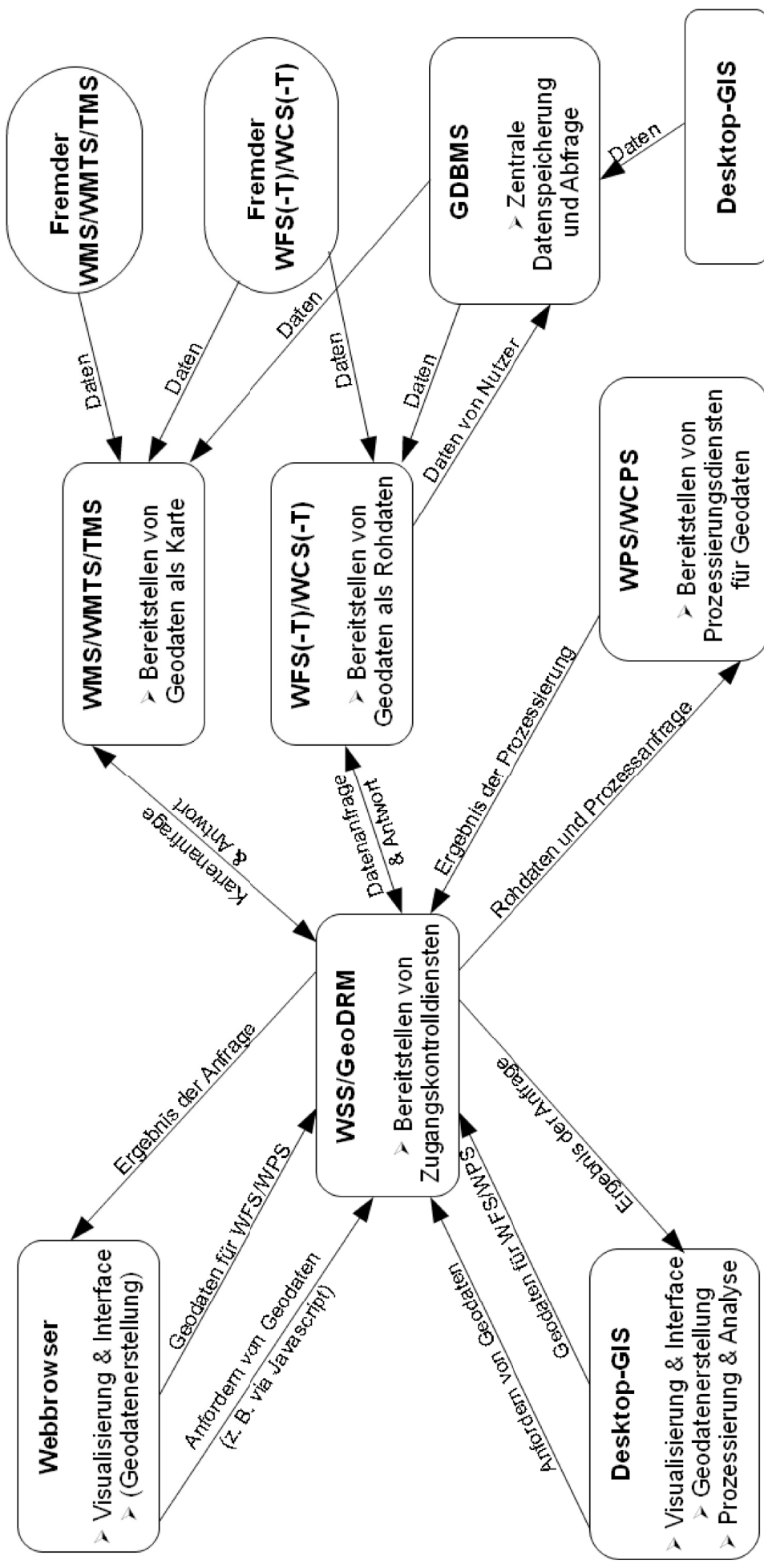


Abbildung 4: Schema für das Zusammenwirken von OGC Webservices

Dieses Szenario soll das Zusammenwirken verschiedener OGC Webservices darstellen. Als Endnutzer eines solchen Systems kommen Desktop-GIS und Webbrowser gleichermaßen infrage, obgleich Erstere tendenziell besser geeignet sind, große Mengen an Rohdaten von WFS oder WCS auszuwerten.

Dienste zur Bereitstellung von Daten in Form von Karten oder Rohdaten können direkt von Datenquellen wie GDBMS oder Dateien im Dateisystem des Servers (nicht dargestellt) gespeist werden oder ihre Daten ganz oder teilweise über andere Webservices erhalten. Kartengenerierende Dienste können dabei die Karten anderer gleichartiger Dienste weiterleiten oder aus Rohdaten Karten erstellen. Eine Gewinnung der Rohdaten aus fertigen Karten ist nicht vorgesehen und wäre auch technisch nur teilweise und unter großem Aufwand möglich. Transaktionale Dienste wie WFS -T und WCS-T ermöglichen die Entgegennahme und weitere Speicherung von vom Nutzer generierten Daten in einem Speichersystem. Im Fall von WFS-T bietet sich hierfür ein GDBMS besonders an.

Geoprozessierungsdienste wie der WPS benötigen als Teil einer Anfrage sowohl die zu verarbeitenden Rohdaten als auch die gewünschte Operation. Als Ergebnis werden die Ergebnisse zurückgesandt.

Der dargestellte WSS repräsentiert OGC Dienste zur Zugangskontrolle. Ist ein solcher Zugangskontrolldienst Teil des Systems, fungiert er als eine Art Vermittler zwischen den Nutzeranfragen und den Diensten, die die Anfragen erfüllen sollen. Auf diese Art kann kontrolliert werden welche Nutzer Zugriff auf welche Leistungen des Systems erhalten und ggf. können Entgelte für die Nutzung erhoben werden. Die Nutzung einer solchen Zugangskontrolle ist nicht zwingend vorgesehen und eine direkte Kommunikation der Nutzer mit den entsprechenden Zieldiensten, wie z.B. einem WMS, ist ohne weiteres möglich. Ein Verzicht auf einen GeoDRM Dienst ist z.B. sinnvoll, wenn die Daten eines Dienstes öffentlich zur freien Verfügung stehen sollen oder wenn eine andere Form der Zugangskontrolle angewendet wird.

Nicht in der Graphik dargestellt sind die "getCapabilities" Anfragen, die alle OGC Webservices unterstützen. Als Antwort auf eine solche Anfrage schickt ein OWS XML-kodierte Daten über die von ihm unterstützten Protokolle, deren Version und weiter Angaben über die von ihm gebotenen Leistungsmerkmale. Die Abfrage einer solchen Information durch den Client geht oftmals der eigentlichen Anforderung von Nutzdaten voraus. Auf OWS beruhende Komponenten bieten den Vorteil der Standardkonformität. Das bedeutet, das alle entsprechenden Komponenten garantiert bestimmte Leistungsmerkmale bieten und mit anderen Webservices über festgelegte Schnittstellen kommunizieren können. Die hat auch den Vorteil, dass einzelne Komponenten eines Web-GIS leichter durch andere ersetzt werden können und die Interaktion oder Zusammenführung mit anderen System ebenfalls erleichtert wird.

1.4.5 Zusammenführung verschiedener verteilter Daten- und Dienstanbieter

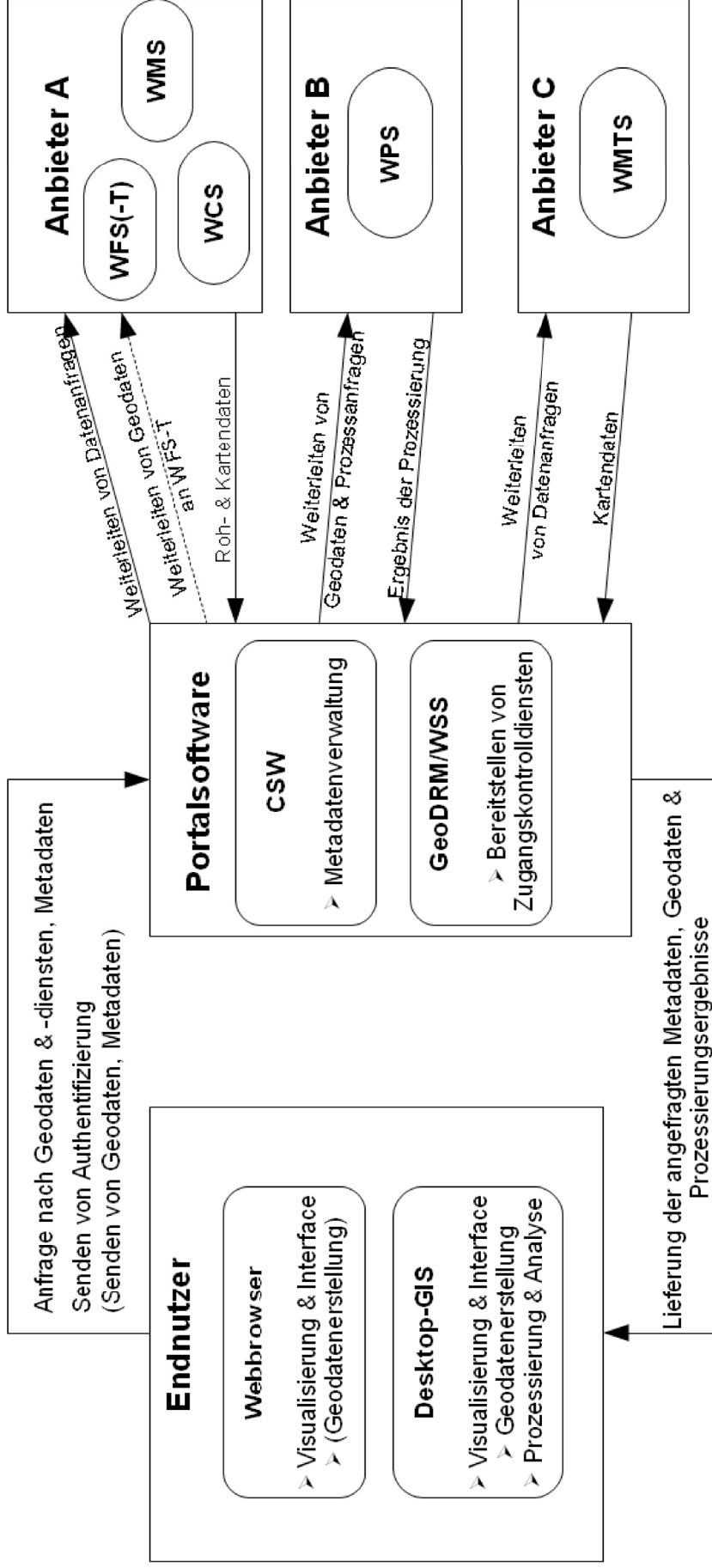


Abbildung 5: Schema für ein Web-GIS auf Basis von OGC Webservices

Dieses Szenario stellt das Zusammenführen mehrerer Anbieter in einem Web-GIS dar. Wie das vorangegangene Szenario liegt auch hier der Fokus auf der Verwendung von OGC Webservices. Die im gesamten Web-GIS angebotenen Leistungen können über Dienste zahlreicher, verschiedener Anbieter bereitgestellt werden. Diese Leistungen und Dienste werden in einem über ein zentrales Register verwaltet und dem Nutzer über ein Portal angeboten. Insbesondere bei der Verwaltung mehrerer Anbieter spielt der CSW eine wichtige Rolle, da es der Zugriff auf umfangreiche Metadaten für den Nutzer möglich macht, zu unterscheiden, welche Leistungen für ihn oder sie geeignet sind. Das Nutzerportal ist ebenfalls der geeignete Ort, um eine Zugangskontrolle für die Leistungen des Systems zu implementieren und für den Nutzer nachvollziehbar darzustellen.

2 Die Lernplattform

Der zweite Teil dieser Arbeit beschreibt das Konzept und die Inhalte der Lernplattform. Er gibt Erläuterungen zu den einzelnen Komponenten und deren Einsatz. In diesem Zusammenhang soll es dem Nutzer auch möglich sein, diesen Teil als eine Art begleitendes Tutorial zu den gegebenen Beispielen zu verwenden, um diese besser nachvollziehen oder für sich variieren zu können. Da die entwickelte Plattform hauptsächlich im Zusammenhang mit der Lehre im Bereich der Geoinformatik eingesetzt werden soll, wird auf Seiten des Nutzers ein sicheres Verständnis von Grundlagen der Informatik und Geoinformatik vorausgesetzt. Insofern wird auf Themen wie beispielsweise die Natur von Vektor- und Rasterdaten oder geodätischen Bezugssysteme nicht eingegangen, sondern vielmehr der Schwerpunkt auf die konkrete Umsetzung der behandelten Aufgaben mit den verwendeten Anwendungen und Daten gesetzt.

2.1 Konzept und Umgebung der LernPlattform

Das Konzept für die Lernplattform besteht darin, die verschiedenen Komponenten eines Web-GIS am Beispiel konkreter Daten und für verschiedene grundlegende Aufgaben umzusetzen und somit eine Sammlung von erläuterten Beispielen zur Verfügung zu stellen. Ein weiteres Ziel ist es dabei, bestimmte Aufgaben über mehrere unterschiedliche Softwareprodukte bzw. Technologien zu realisieren und somit neben der Möglichkeit diese unterschiedlichen Produkte kennenzulernen, einen Vergleich der Fähigkeiten und Nutzungseigenschaften zu ermöglichen.

Als Umgebung für die Lernplattform wurde das OSGeo Live Projekt in der zum Zeitpunkt der Erstellung aktuellen Version 6.5 und der Ausführung als virtuelle Maschine gewählt. Es handelt sich dabei um eine auf der Xubuntu (Ubuntu mit Xfce Window Manager) LINUX Distribution aufsetzende Systeminstallation, die verschiedene Open Source Produkte für Web-GIS Anwendung sowie weitere OSGeo Projekte und Beispieldaten beinhaltet. Diese Umgebung enthält den Großteil der verwendeten Programme und ist für jeden frei auf der Website des OSGeo Live Projektes herunterladbar. Die Software für das Abspielen der virtuellen Maschine ist über VMWare Player oder VirtualBox ebenfalls für jeden Nutzer der Plattform kostenlos möglich. Darüber hinaus befindet sich die OSGeo Live VM an der Hochschule Neubrandenburg bereits im Einsatz und ist somit als Grundlage für die Lernplattform bestmöglich verfügbar und mit minimalem Aufwand einsetzbar [vgl. OSGLive].

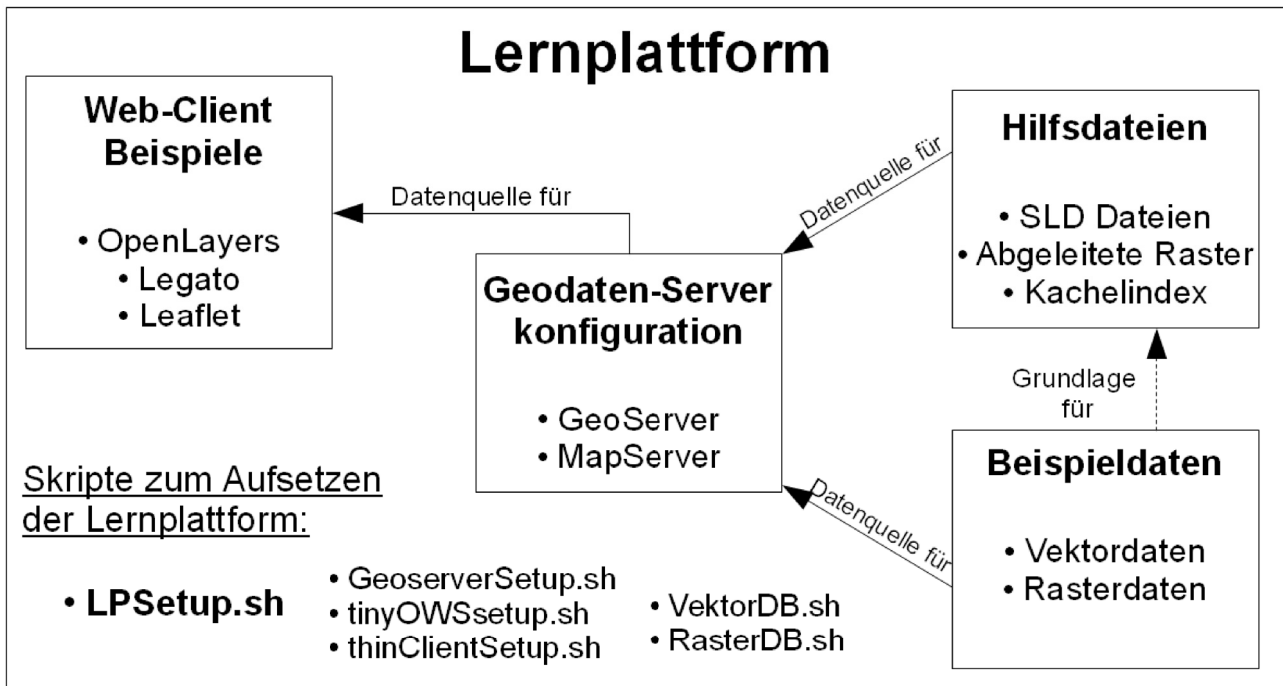


Abbildung 6: Schema der Lernplattform

Technisch besteht die Plattform aus den Beispieldatensätzen und einer Reihe von Konfigurations- und Programmdateien, die je nach Art der Dateien möglichst einfach, sowohl im Ganzen als auch für Teile der Plattform, durch Skripte in die Umgebung integriert bzw. für den Nutzer dargestellt werden können. Eine jeweils konkretere Beschreibung dazu befindet sich in den folgenden, einzelnen Abschnitten. Die Shell Skripte beruhen dabei auf den Bedingungen der OSGeo Live Umgebung, insbesondere im Hinblick auf Systemnutzer und deren Rechte sowie den Pfaden zu verschiedenen Konfigurationsdateien. Folglich ist die Plattform voraussichtlich nur auf diesem System ohne Anpassungen sofort voll einsetzbar. Der Großteil des Inhaltes ist jedoch vom Betriebssystem weitgehend unabhängig, sodass unter Anpassung der entsprechenden Skripte bzw. durch manuelle Platzierung entsprechender Konfigurationsdateien eine Nutzung auch unabhängig von der OSGeo VM möglich ist.

2.1 Beispieldatensätze

2.1.1 Vektordaten

Die Beispielvektordaten liegen in Form einer Reihe von ESRI Shapefiles (jeweils als Satz aus den entsprechenden *.shp, *.dbf, *.shx und *.prj Dateien) vor. Das räumliche Bezugssystem ist das EPSG 25833 und somit ETRS89 (WGS84 mit UTM-Abbildung Zone 33).

Nicht alle Shapefiles werden in den Beispielen verwendet und dienen nur als zusätzliche Demodaten für den Nutzer. Die in den Beispielen verwendeten Datensätze werden im Folgenden

kurz vorgestellt:

2.1.1.1 Punktdaten:

- hotels.shp – stellt die Standorte von Hotels in Neubrandenburg dar

2.1.1.2 Liniendaten:

- B_Str.shp – stellt die Bundesstraßen im Gebiet um Neubrandenburg dar
- L_Str.shp – stellt die Landesstraßen im Gebiet um Neubrandenburg dar
- verkehr.shp – stellt Verkehrswege innerhalb Neubrandenburgs dar
- radwanderwege.shp – stellt die Radwanderwege im Gebiet um Neubrandenburg dar

2.1.1.3 Polygone:

- flaechnutzung – stellt die verschiedenen amtlichen Flächennutzungen innerhalb des Neubrandenburger Stadtgebietes dar
- gewaesser - stellt verschiedene Gewässer in Neubrandenburg dar

2.1.2 Rasterdaten

Die Beispielrasterdaten bestehen aus den folgenden beiden Datensätzen:

- Eine Sammlung von sechs Orthophotos von Teilen Neubrandenburgs, die genau aneinander grenzende Gebiete darstellen und sich somit für die Bildung eines Mosaiks eignen. Die Daten liegen als Jpeg Dateien mit zugehörigen ESRI World Files (*.jfw Dateien) zur Angabe der Georeferenzierungsdaten und *.proj Dateien zur Beschreibung des Bezugssystems vor. Das Bezugssystem stellt wieder das EPSG 25833 dar. Die Dateien sind im Folgenden benannt:
 - 333825934.jpg; 333825936.jpg; 333845934.jpg; 333845936.jpg; 333865934.jpg; 333865936.jpg
- Eine LANDSAT Aufnahme des Gebietes von Neubrandenburg und Umgebung mit sechs Spektralkanälen. Die Datei liegt als GeoTIFF vor und ist folglich ohne weitere Dateien georeferenziert. Das Bezugssystem stellt wieder das EPSG 25833 dar. Die Datei ist im Folgenden benannt:
 - Neubrandenburg.tif

2.1.3 Speicherort in der Lernplattform

Die Beispieldatensätze befinden sich im Unterordner `./NB_sampledaten` der Lernplattform. Genauer sind die Vektordatensätze unter `./NB_sampledaten/Shapefiles/` und die Rasterdatensätze unter `./NB_sampledaten/Rasterfiles/WorldfileImages` (für die Orthophotos) sowie `./NB_sampledaten/Rasterfiles/GeoTIFF` (für die Landsat Aufnahme) zu finden.

2.2 Vorbereitung der Daten

2.2.1 Konvertierung der Shapefiles in eine Geodatenbank

ESRI Shapefiles stellen immer noch einen de facto Standard für die Speicherung von Geodaten dar und viele Datensätze stehen somit in diesem Datenformat zur Verfügung. Das Format selbst besitzt zahlreiche Einschränkungen im Vergleich zu modernen Dateiformaten und insbesondere gegenüber Geodatenbanken. Das hat zur Folge, dass die Konvertierung von Shapefiles in eine Geodatenbank eine für die Praxis wichtige Anforderung darstellt. Die verschiedenen Datenbanksysteme stellen dabei oftmals jeweils eigene Werkzeuge zur Verfügung. Diese Werkzeuge können in graphische Oberflächen eingebunden oder kommandozeilenbasiert sein, wobei verschiedene Systeme, u.a. Oracle Spatial, PostGIS und SpatiaLite, auch beide Varianten anbieten. Im Folgenden soll ein Überblick über verschiedene Möglichkeiten der Lösung des Problems gegeben werden. Die folgenden Abschnitte und die weiteren Komponenten der Lernplattform widmen sich danach konkret der Arbeit mit PostGIS bzw. bauen darauf auf.

2.2.1.1 Beispiel CLI Konverter von Oracle Spatial

Oracle Spatial bietet neben graphischen Tools zum Einlesen von Geodaten javabasierte Komponenten für das Konvertieren von Shapefiles, die über die Kommandozeile genutzt oder in Java Anwendungen integriert werden können. Die Vorteile einer kommandozeilenbasierten Anwendung liegen insbesondere in der Verwendbarkeit in Skripten, welche Konvertierung der Shapefiles automatisieren können. Die Komponenten lesen das Shapefile, konvertieren dessen Geometriedaten in das von Oracle Spatial genutzte SDO_Geometry Format und speichern den gesamten Inhalt des Dokuments in einen Oracle Datensatz. Die Nutzung erfolgt nach folgendem Schema:

```
java -cp [ORACLE_HOME]/jdbc/lib/ojdbc5.jar:
[ORACLE_HOME]/md/jlib/sdoutl.jar:[ORACLE_HOME]/md/jlib/sdoapi.jar
oracle.spatial.util.SampleShapefileToJGeomFeature -h db_host -p db_port
-s db_sid -u db_username -d db_password -t db_table -f shapefile_name [-i
table_id_column_name][-r srid][-g db_geometry_column][-x max_x,min_x][-y
max_y,min_y][-o tolerance]
```

Ohne auf alle einzelnen Parameter einzugehen, wird deutlich, dass der Funktion die Verbindungsparameter der Zieldatenbanktabelle, das zu konvertierende Shapefile sowie eine Reihe von geometrischen Eigenschaften (z.B. das Bezugssystem über srid), die aus dem Shapefile nicht ersichtlich sind bzw. nicht daraus gelesen werden, übergeben werden müssen.

Alle Angaben zum "Oracle Spatial Shapefile Converter" sind entnommen aus [O11g S.D1-D2]

2.2.1.2 Beispiel Nutzung der Spatialite-GUI

Spatialite bietet neben einer Reihe von Kommandozeilenwerkzeugen, die u.a. Shapefiles lesen und konvertieren können, die "spatialite-gui", die neben vielen anderen Funktionen das Importieren von Shapefiles erlaubt. Der grundlegende Ablauf lässt sich wie folgt darstellen:

- Erstellen einer neuen SQLite-Datenbankdatei

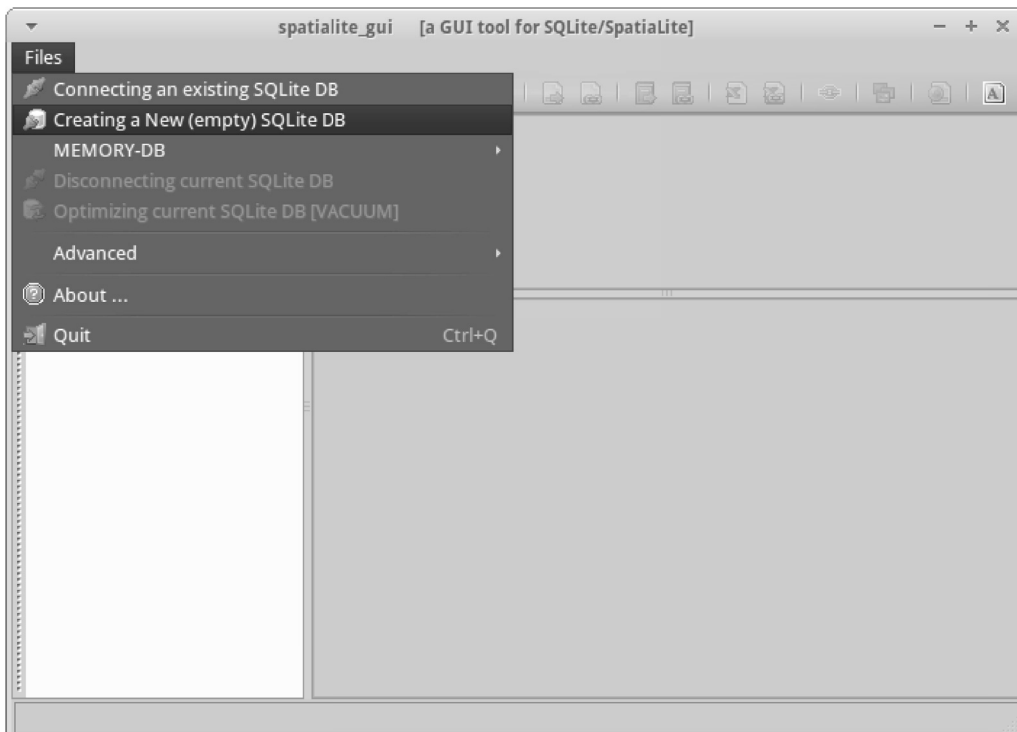


Abbildung 7: Oberfläche der spatialite-gui

- Aktivieren des Shapefile Imports

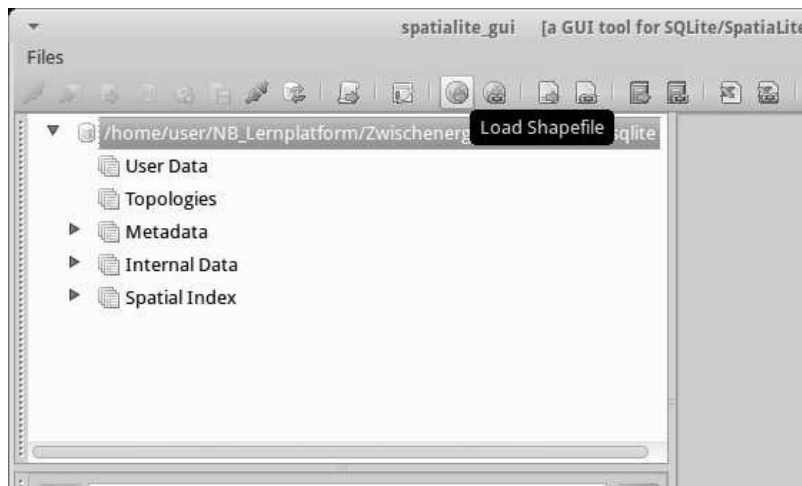


Abbildung 8: Schaltfläche für den Shapefile Import

- Konfiguration der Einstellungen des Imports. Insbesondere der SRID (Eindeutiger Bezeichner des räumlichen Bezugssystems) muss bei jedem Datensatz manuell festgelegt werden.
- Die nötigen Relationen inklusive räumlichen Indexen (optional wählbar) sind erstellt und die Daten des Shapefiles dort gespeichert.

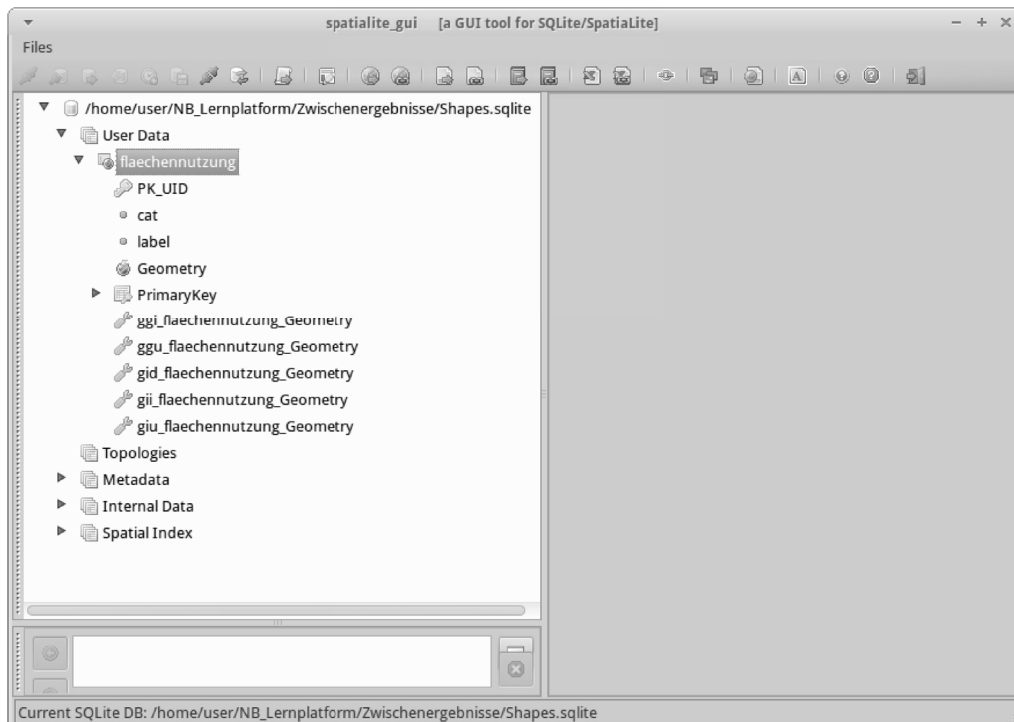


Abbildung 9: Ergebnis des Shapefile Imports

Die im Beispiel angefertigte SQLite-Datenbankdatei `Shapes.sqlite` steht in der LernPlattform unter `./Zwischenergebnisse/Spatialite/` zur Verfügung und enthält die unter 2.1 aufgeführten Vektordatensätze. Die allgemeinen Aussagen zu Spatialite beruhen auf [SLite], wobei insbesondere das Spatialite Cookbook für weitergehende Informationen zu dem System empfohlen wird.

2.2.1.3 nicht Datenbank-spezifische Methoden

Die vorangegangenen Methoden sind für ihre zugehörigen GDBMS effektiv und leicht zu benutzen. Sollen Daten jedoch in mehrere Datenbanksysteme geladen werden, bedeutet dies die Verwendung mehrerer Werkzeuge, was i.d.R. nicht gewünscht wird.

Eine Möglichkeit des Austausches von Datensätzen zwischen Datenbanken besteht in der Nutzung von SQL Skripten, die theoretisch aufgrund der Standardisierung einen Austausch der Daten ermöglichen sollen. Dies kann aber nur funktionieren, wenn keine anbieterspezifischen Funktionen oder Syntax verwendet werden. Für Geodaten sind hier die SFSQL und SQL/MM part 3 relevante Standards, die Funktionen und Syntax in bestimmten Ausmaß festlegen. Dennoch lassen die Standards insbesondere in Bezug auf die Syntax teilweise Freiheiten, sodass diese Möglichkeit nur bedingt zuverlässig ist.

Eine weitere Möglichkeit stellt die Nutzung spezieller Programme bzw. Bibliotheken für die Konvertierung von Geodaten dar. In diesem Zusammenhang bietet die OGR Bibliothek, welche Teil des GDAL Projekts ist, die Möglichkeit Shapefiles sowie andere Datenformate in PostGIS, Oracle Spatial, Spatialite sowie mehrere andere GDBMS zu importieren. Hierbei ist üblicherweise eine entsprechende, separat einzubindende Bibliothek notwendig, mit deren Hilfe OGR die Verbindung mit der Datenbank herstellt. Da OGR auch eingebunden in Kommandozeilenwerkzeuge verfügbar ist, lassen sich über Skripte Importaufträge in unterschiedliche GDBMS automatisieren. Der Export von Geodatensätzen aus Datenbanken in verschiedene Dateiformate ist über OGR ebenfalls möglich [vgl. GDAL/OGR].

2.2.2 Erstellung einer PostGIS Datenbank für die Vektordaten

Obwohl es grundsätzlich möglich ist, Vektordaten aus Shapefiles über Webdienste bereitzustellen, bietet die Speicherung in einer Datenbank verschiedene Vorteile, sodass alle verwendeten Datensätze in eine PostGIS Datenbank gespeichert werden sollen. Dafür werden folgende Schritte durchgeführt:

- Erzeugung einer leeren PostgreSQL Datenbank (über die Kommandozeile, andere Werkzeuge wie pgAdmin 3 können natürlich auch verwendet werden)
 - `psql -U user -c "CREATE DATABASE nbsample";`
- Laden der PostGIS Erweiterung, um räumliche Datentypen und Operationen zu unterstützen. An dieser Stelle wird die neuere Methode mithilfe von PostgreSQL Extensions verwendet, die ältere Methode mithilfe von verschiedenen *.sql Skripten ist natürlich weiterhin gleichwertig möglich.
 - `psql -d nbsample -c "CREATE EXTENSION postgis;"`
- Laden der PostGIS Erweiterung für Topologieunterstützung. Dieser Schritt ist für die folgenden Anwendungen optional. Die notwendigen PostGIS Funktionen für Vektor- und Rasterdaten stehen bereits durch die obige Erweiterung zur Verfügung.
 - `psql -d nbsample -c "CREATE EXTENSION postgis_topology;"`
- Laden der Shapefiles in die PostGIS Datenbank. Für diesen Zweck wird das zu PostGIS gehörige Hilfsprogramm `shp2pgsql` verwendet, das es ermöglicht, ESRI Shapefiles direkt in SQL Anweisungen zur Erstellung von PostGIS Tabellen zu konvertieren. Um diesen Prozess schneller und komfortabler zu gestalten, befindet sich im Verzeichnis der Shapefiles das Skript `SHAPESLADEN.sh`, das alle sich im selben Verzeichnis befindenden Shapefiles mithilfe von `shp2pgsql` konvertiert und in die vorbereitete Datenbank schreibt.
 - `cd ./NB_sampledaten/Shapefiles`
 - `sh SHAPESLADEN.sh`
- Erzeugen eines Datenbank `VIEWS` bzw. einer Sicht. `VIEWS` können das Ergebnis komplexer Abfragen sein und werden dynamisch aus ihren jeweiligen Quellen generiert. Dies kann praktisch vielfach eingesetzt werden. So können z.B. komplexe, fachbezogene Datenzusammenstellungen, ggf. mit durch Funktionen ermittelten Feldern, als `VIEWS` generiert werden, während die zugrundeliegenden Basisdaten der Datenbank in mehreren einfachen Tabellen gehalten werden. Die Möglichkeit `VIEWS` über Web-GIS zu veröffentlichen ist daher interessant und soll im Folgenden demonstriert werden. Dafür wird zunächst als Datenbeispiel ein `VIEW` angelegt, der aus der Tabelle `hotels` nur die Einträge auswählt, die nahe genug am Strand eines Gewässers aus der Relation `gewaesser` liegen.

- `psql -d nbsample -U user -c "CREATE VIEW strandhotel AS SELECT hotels.gid,hotels.label AS hotel,hotels.geom,ST_Distance(hotels.geom,gewaesser.geom)::Integer AS entfernung FROM gewaesser, hotels WHERE gewaesser.label ILIKE '%Tollensesee%' AND ST_DWithin(gewaesser.geom,hotels.geom,1500) ORDER BY entfernung asc"`
- Erzeugen von drei neuen leeren Tabellen, die für die Erfassung von Daten vorgesehen sind.
 - `psql -d nbsample -U user -c " CREATE TABLE public.editpoint(gid SERIAL,label character varying(80),geom geometry(Point,25833),CONSTRAINT editpoint_pkey PRIMARY KEY (gid))"`
 - `psql -d nbsample -U user -c " CREATE TABLE public.editline(gid SERIAL,label character varying(80),geom geometry(LineString,25833),CONSTRAINT editline_pkey PRIMARY KEY (gid))"`
 - `psql -d nbsample -U user -c " CREATE TABLE public.editpoly(gid SERIAL,label character varying(80),geom geometry(Polygon,25833),CONSTRAINT editpoly_pkey PRIMARY KEY (gid))"`

Die hier durchgeführten Schritte stehen im Skript `VektorDB.sh` im Hauptverzeichnis zur Verfügung und können einfach aktiviert werden.

2.2.3 Vorbereitung der Rasterdaten

2.2.3.1 Speicherung der Raster in der Datenbank

Die Speicherung von Rasterdaten in einer Datenbank ist noch nicht so verbreitet wie für Vektordaten, wird jedoch von PostGIS über den ehemals separaten Bestandteil PostGIS Raster vollständig unterstützt und soll hier anhand der Beispieldaten durchgeführt werden:

- Konvertieren und Laden der Landsat Aufnahme `neubrandenburg.tif`. Dazu wird das Hilfsprogramm `raster2pgsql` verwendet, das analog zu `shp2pgsql` dazu dient, Rasterdaten in SQL Anweisungen zu konvertieren. In diesem Beispiel wird die Datei einfach, ohne Kachelung, in der Datenbank gespeichert, da `neubrandenburg.tif` relativ

klein ist und in kurzer Zeit geladen werden kann. Der Aufruf verwendet die Parameter `-I` und `-C`, die respektive bewirken, dass für die neue Rastertabelle ein räumlicher Index und eine Registrierung im Metadaten `VIEW raster_columns` erzeugt wird.

```
◦ /usr/bin/raster2pgsql -s 25833 -I -C -M neubrandenburg.tif -F  
public.landsat | psql -U user -d nbsample
```

- Konvertieren und Laden der Orthophotos in eine Datenbanktabelle. Ziel der folgenden Schritte ist es, alle Orthophotos in eine Relation zu speichern, sodass beim Lesen aus der Datenbank bequem ein Mosaik erstellt werden kann. Deshalb werden fünf der sechs Bilder mit dem `-a` Parameter geladen, sodass diese nicht in eigene Tabellen gespeichert, sondern an das erste Bild angefügt werden. Da die sechs Dateien zusammen relativ groß sind und beträchtliche Ladezeit benötigen können, sollen die Bilder gekachelt und Overviews bzw. die Stufen von Bildpyramiden erstellt werden. Dies geschieht respektive mit den Parametern `-t` gefolgt von den Abmessungen der Kacheln und dem Parameter `-l` gefolgt von den gewünschten Verkleinerungsfaktoren des Overviews (4 entspricht $1/2^4$ mal sovielen Einträgen wie das Original). Um die mehrfache Speicherung von Daten einzuschränken, werden die Bilder unter Nutzung des `-R` Parameters nur in der Datenbank registriert und somit deren Metadaten verwaltet. Auf diese Weise findet keine eigentliche Konvertierung und Speicherung der Bilddaten statt, was in der Praxis interessant sein kann, wenn Rasterdaten zwar in der Datenbank verwaltet werden sollen aber z.B. aus Gründen der Performance nicht in die Datenbank gespeichert werden können. Aufgrund der Notwendigkeit, nacheinander mehrere Bilder in eine Relation zu speichern, können die Parameter `-I` und `-C` nicht verwendet werden, sodass die Registrierung der Metadaten und ggf. die Erstellung eines räumlichen Indexes separat erfolgen muss.

```
◦ /usr/bin/raster2pgsql -c -s 25833 -r -M -R -l 4 333825934.jpg -F  
-t 125x125 public.ortho | psql -d nbsample
```

```
◦ /usr/bin/raster2pgsql -a -s 25833 -r -M -R -l 4 333825936.jpg -F  
-t 125x125 public.ortho | psql -d nbsample
```

```
/usr/bin/raster2pgsql -a -s 25833 -r -M -R -l 4 333845934.jpg -F  
-t 125x125 public.ortho | psql -d nbsample
```

```
/usr/bin/raster2pgsql -a -s 25833 -r -M -R -l 4 333845936.jpg -F  
-t 125x125 public.ortho | psql -d nbsample
```

```
/usr/bin/raster2pgsql -a -s 25833 -r -M -R -l 4 333865934.jpg -F  
-t 125x125 public.ortho | psql -d nbsample
```

```
/usr/bin/raster2pgsql -a -s 25833 -r -M -R -l 4 333865936.jpg -F  
-t 125x125 public.ortho | psql -d nbsample
```

```
◦ psql -d nbsample -c "SELECT AddRasterConstraints('ortho'::name,  
'rast'::name);"
```

```
psql -d nbsample -c "SELECT  
AddRasterConstraints('o_4_ortho'::name, 'rast'::name);"
```

- Die so hinterlegten Rasterdaten können von geeigneten Anwendungen direkt genutzt oder vorher separat wieder als Datei extrahiert werden. GDAL stellt sowohl als Kommandozeilenwerkzeug, als auch als in andere Anwendungen integrierte Bibliothek einen der üblichsten Wege dar, auf Geodatenformate, insbesondere Rasterdaten, zuzugreifen. Deswegen soll im folgenden Schritt der Overview des Orthophotomosaiks mithilfe von GDAL als Gesamtbild ausgelesen werden.

```
◦ gdal_translate -of GTIFF PG:"host=localhost dbname='nbsample'  
user='user' password='user' schema='public' table='o_4_ortho'  
mode='2'" ortho_overview4.tif
```

Achtung: Der oben genannte Schritt führte zum Zeitpunkt der Erstellung zu einem Absturz des GDAL Prozesses. Der Grund dafür wird in einer Inkompatibilität der installierten GDAL Version 1.9 mit PostGIS 2.0 gesehen. Der Zugriff auf Rasterdaten funktionierte bereits mit Entwicklungsversionen von PostGIS 2.0, in denen bestimmte Metadaten noch nicht als Sichten, sondern als Tabellen realisiert waren. Es ist daher davon auszugehen, dass ein Update auf die GDAL Version 1.10 dieses Problem behebt. Ein solches Update steht jedoch über die Ubuntu Paketquellen zum Zeitpunkt des Schreibens noch nicht zur Verfügung.

- Die Schritte zu Speicherung der Rasterdaten stehen im Skript `RasterDB.sh` im Hauptverzeichnis zur Verfügung und können einfach aktiviert werden.

2.2.3.2 Erzeugen von Overviews

Das Erstellen von Overviews für Rasterdaten ist auch außerhalb der Datenbank basierten Speicherung relevant und soll hier noch einmal für das Orthophotomosaik durch GDAL durchgeführt werden:

- Erzeugen einer Bilddatei aus den Einzelbildern mithilfe von `gdal_merge`. Dabei findet wird das Bild in das GeoTiff Format gewandelt, da Dateitypen wie Jpeg und PNG nicht das Schrittweise Schreiben in die Datei unterstützen, dass `gdal_merge` benötigt.

- `gdal_merge.py -o orthomosaik.tif -of GTiff -co COMPRESS=JPEG /home/user/NB_Lernplattform/NB_sampledaten/Rasterfiles/WorldfileImages/*.jpg`
- Die verwendete Version von `gdal_merge` setzt den Parameter zur Komprimierung der Daten leider nicht um, sodass `orthomosaik.tif` sehr groß wird. Deshalb kann mit einem anderen GDAL Werkzeug komprimiert werden.
 - `gdal_translate -co COMPRESS=JPEG orthomosaik.tif orthomosaikjpg.tif`
- Overviews können mit `gdal_addo` erzeugt werden. Diese werden bei Tiff Dateien üblicherweise direkt in die Datei eingebettet und können von geeigneten Anwendungen genutzt werden. Es ist ebenfalls möglich, die Overviews durch den `-ro` Parameter als separate Dateien zu erstellen, was im Folgenden auch so umgesetzt wird.
 - `gdaladdo -ro -r nearest --config COMPRESS_OVERVIEW JPEG orthomosaikjpg.tif 4`

Die resultierende Datei ist wieder ein Overview der Verkleinerungsstufe 4 wie derjenige, der im Abschnitt zur Datenbank basierten Speicherung erzeugt wurde. Diese Datei hat, so wie auch das durch `gdal_merge` erzeugte Bild, keine Daten zur Georeferenzierung, da die vorhandenen Worldfiles nicht ausgewertet werden. Die notwendige Georeferenzierung lässt sich am leichtesten durch ein weiteres Worldfile erreichen. Ein Worldfile ist eine Textdatei mit sechs Zahlenwerten, wovon zwei die Koordinaten der linken oberen Bildecke, zwei weitere Angaben zur Rotation des Bildes und zwei die Ausdehnung der Bildpixel in Richtung der X- bzw. Y-Achse beschreiben. Das betreffende Worldfile konnte daher in einem Texteditor aus erzeugt werden, da die Lage der linken oberen Ecke des Overviews der, des linken oberen Teilbildes entspricht, die Rotation aller Teilbilder gleich und Null ist und das Ausmaß der Pixel sich entsprechen des Verkleinerungsfaktors auf das 16-fache (2^4) erhöht. Die notwendige `.prj` Datei beschreibt dasselbe Koordinatensystem wie vorher und kann daher einfach von einem Teilbild kopiert und umbenannt werden.

Der erzeugte Overview steht in der Lernplattform unter dem Namen `orthomosaikoverview4.tif` inklusive Worldfile und `.prj` Datei im Unterordner `./Zwischenergebnisse/Raster/` zur Verfügung.

2.2.3.3 Erzeugen eines Kachelindex

Unter einem Kachel oder Tile Index wird eine Vektordatendatei verstanden, die die Bezeichnungen

und räumlichen Ausdehnungen anderer Datensätze enthält. Er wird üblicherweise genutzt, um das gemeinsame Darstellen einer Reihe von Datensätzen, z.B. Rasterkacheln, effizient zu ermöglichen. Alternativen wie das Einbinden der Datensätze als unterschiedliche Layer einer Karte, wären bei sehr vielen solchen Datensätzen oft unpraktisch bzw. mit geringerer Performance verbunden.

Für die Nutzung in der Lernplattform wurde für die Orthophotos ein Kachelindex mithilfe von `gdaltindex` erstellt. Dieser kann genutzt werden dynamisch ein Mosaik dieser Dateien zu erstellen:

```
gdaltindex mosaik.shp
/home/user/NB_Lernplattform/NB_sampledaten/Rasterfiles/WorldfileImages/*.j
pg
```

Da der erzeugte Kachelindex `mosaik.shp` Pfade zu den Bilddateien beinhaltet, kann er nur dann direkt genutzt werden wenn der Speicherort der Dateien mit dem oben gezeigten übereinstimmt. Sofern die Lernplattform in einem anderen Verzeichnis liegt, muss ein neuer Index nach obigem Beispiel erzeugt werden.

Der erzeugte Kachelindex steht in der Lernplattform im Unterordner `./Zwischenergebnisse/Raster/` zur Verfügung.

2.2.4 Erzeugung von SLDs für die Gestaltung der Vektordaten

Die Dateien mit der Endung `*.sld` sind XML-kodierte Textdateien, die die Standards SLD und SE nutzen, um Darstellungsanweisungen für Geodaten zu speichern. Dieses Format wird von vielen Server- und Clientanwendungen im GIS Bereich unterstützt und soll auch in der Lernplattform genutzt werden.

Für die Erzeugung der Style Dateien gibt es verschiedene Möglichkeiten. Grundsätzlich kann ein Texteditor verwendet werden, um die Dateien zu erstellen. Hier stehen neben einfachen Texteditoren auch solche mit Syntaxhervorhebung für XML oder auch spezifisch für SLD sowie validierende XML-Editoren zur Verfügung. Dieser Ansatz erfordert jedoch gute Kenntnisse des Standards und seiner Syntaxelemente. Zusätzlich gibt es hier auch kein direktes visuelles Feedback, was die Gestaltung einer neuen Darstellungsvorschrift erschweren kann. Eine andere Möglichkeit stellt die Erzeugung aus Desktop-GIS heraus dar. Diese Anwendungen haben üblicherweise umfangreiche Funktionen zur visuellen Gestaltung von Geodaten, haben diese aber traditionell in eigenen, spezifischen Dateiformaten abgelegt. Aktuell bieten viele der im ersten Teil vorgestellten Systeme die Möglichkeit an, Gestaltungsvorschriften neben den eigenen Formaten auch in standardkonformen SLD Dateien zu speichern.

Für die Lernplattform wurden auf diese Weise einige einfache Gestaltungsdateien erstellt. Hierfür wurde das Desktop-GIS uDig verwendet. Beispielhaft wird im Folgenden der Prozess für den Datensatz `flaechennutzung.shp` dargestellt.

- Ausgangspunkt ist der geladene Datensatz in uDig.

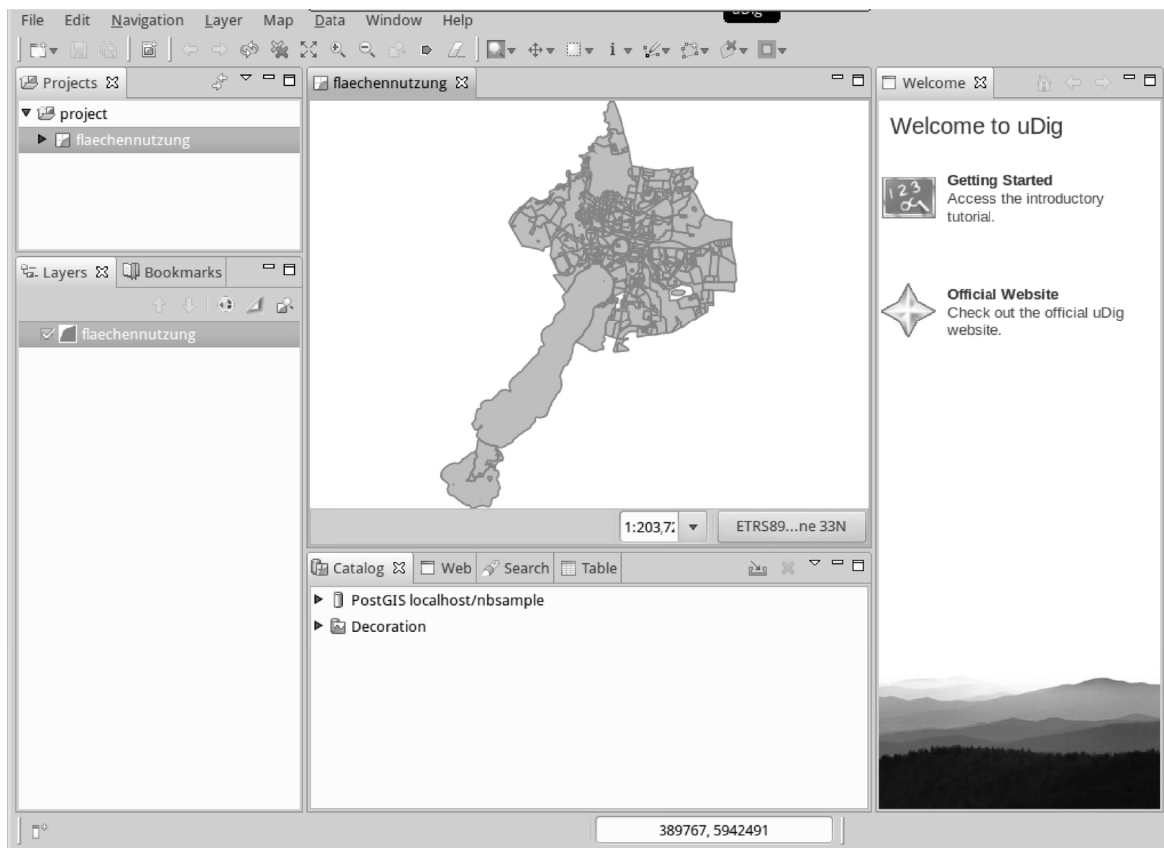


Abbildung 10: uDig GUI mit Beispieldatensatz

- Öffnen des Gestaltungsmenus unter Layer → Change Style

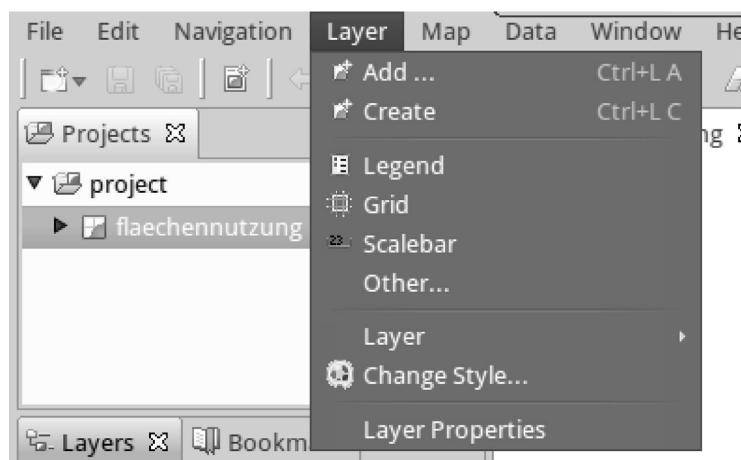


Abbildung 11: Change Style Option im uDig Layer Menü

- Das Gestaltungsmenü erlaubt verschiedene Gestaltungsansätze. Hier wurde unter der Option Theme eine Kategorisierung nach dem Attribut `label`, welches die Flächennutzung textuell beschreibt, durchgeführt und jeder Kategorie eine Farbe zugewiesen, die möglichst intuitiv mit der Nutzungsart assoziiert ist.

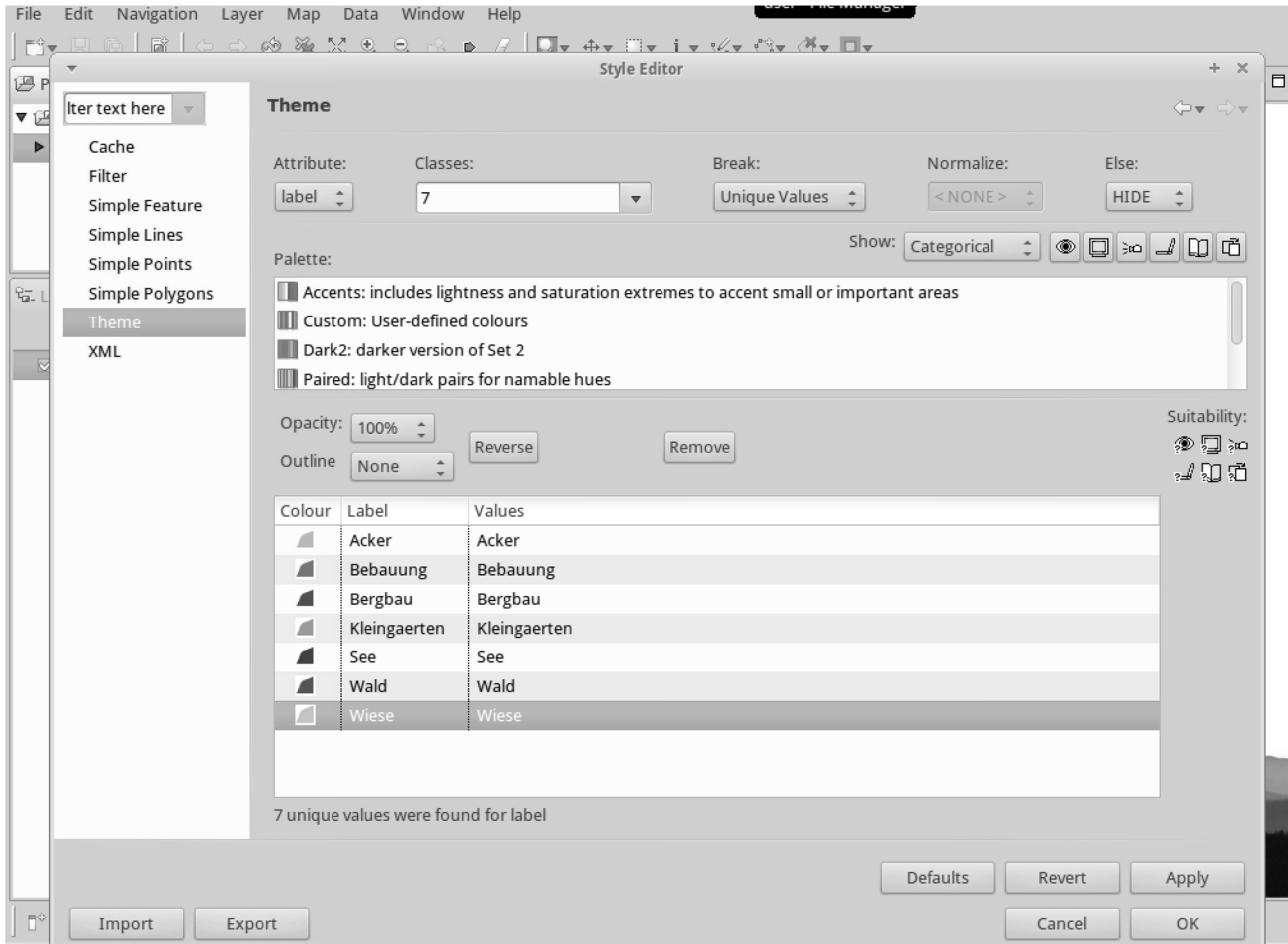


Abbildung 12: Beispielgestaltung der Flächennutzungsdaten in uDig

- Die so erstellten Gestaltungsvorschriften können unter dem Menüpunkt XML als SLD kodierte Anweisung angesehen und abgespeichert werden.

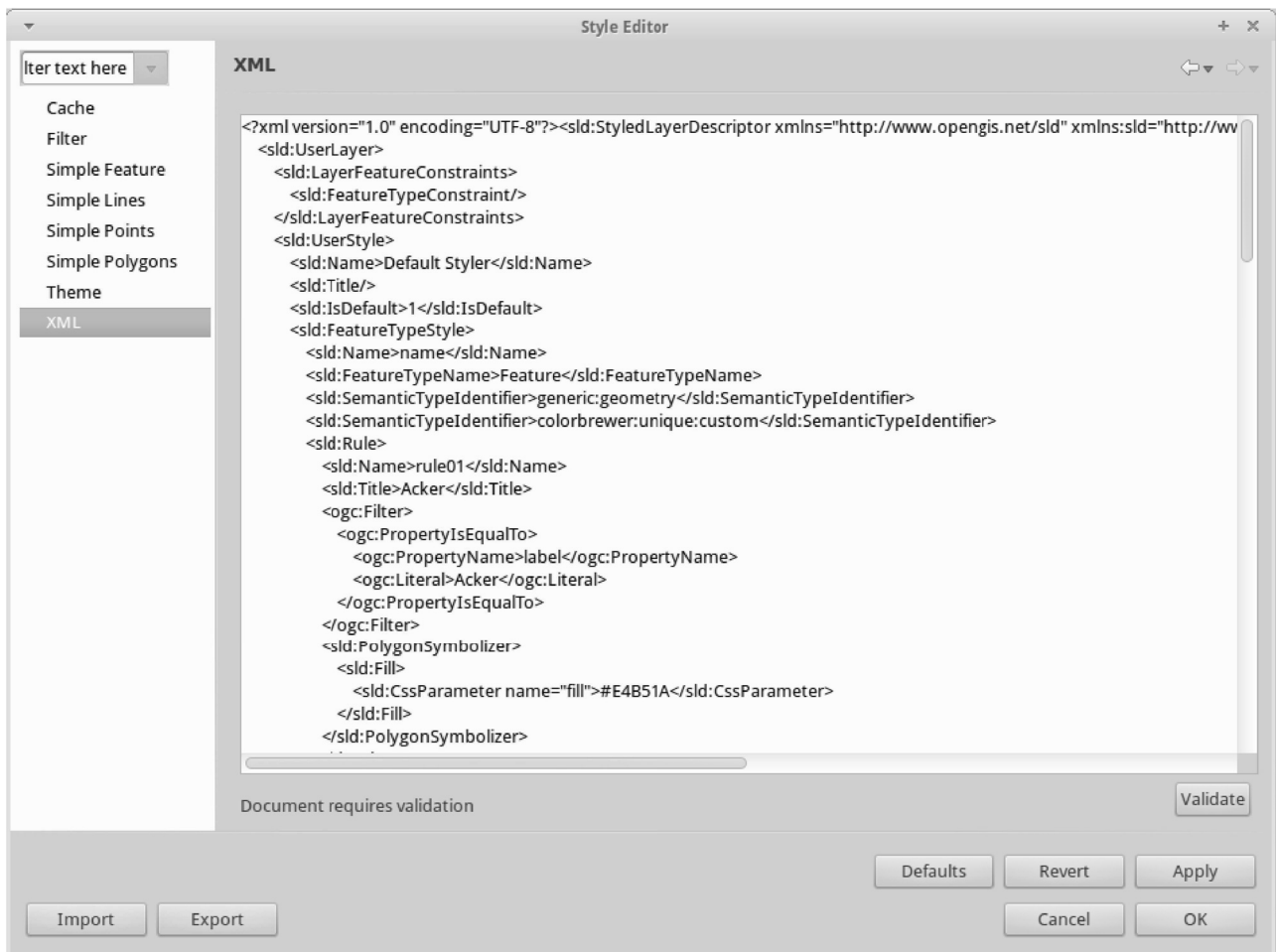


Abbildung 13: Die Beispielgestaltung in XML-Repräsentation

Achtung: Obwohl alle der im Abschnitt zu SLD des ersten Teils genannten Desktop-GIS das Erzeugen und importieren von SLD Dateien ermöglichen, ist diese Funktionalität zum Zeitpunkt der Erstellung dieser Arbeit scheinbar noch nicht ausgereift. Während verschiedener Versuche wurden teilweise nicht valide (Der Test erfolgte durch den in Geoserver integrierten Editor) oder nicht wie gewünscht funktionierende Dateien erstellt. Ein aussagekräftiger Vergleich und eine entsprechende Untersuchung der einzelnen Systeme würde den Rahmen dieser Arbeit sprengen und wäre höchstwahrscheinlich nur für sehr kurze Zeit relevant. Auf Basis einzelner Versuche ist jedoch der Eindruck entstanden, dass uDig über verhältnismäßig gute Fähigkeiten bei der Erstellung gültiger SLD Dokumente verfügt. Auf lange Sicht kann jedoch vermutet werden, dass die Unterstützung bei allen Desktop-GIS ein ausgereiftes Stadium erreichen wird.

Die hier erstellte `flaechennutzung.sld`, sowie alle weiteren in der Lernplattform enthaltenen Style Dateien stehen im Unterordner `./Zwischenergebnisse/SLD/` zur Verfügung.

2.3 Bereitstellung der Daten über Webdienste

2.3.1 Bereitstellung mithilfe von Geoserver

Geoserver wird über eine graphische Web-Oberfläche konfiguriert. Aufgrund der komplexen möglichen Anforderungssituationen an die Veröffentlichung von Geodaten stehen über die verschiedenen Teile der Oberfläche eine Vielzahl von Optionen zur Verfügung, die das Verhalten von Geoserver in vielen speziellen Situationen regeln. Im Folgenden werden die notwendigen Schritte zur Veröffentlichung der Beispieldaten über dieses Interface und die daran beteiligten Konzepte dargestellt. Für umfassendere Informationen zur Nutzung und Administration von Geoserver wird auf die Geoserver Dokumentation in [GSdoc] verwiesen.

The screenshot shows the GeoServer web interface. At the top left is the GeoServer logo. At the top right, it says "Logged in as admin." with a "Logout" button. The main content area is titled "Welcome" and contains the following information:

- 31 Layers (Add layers)
- 17 Stores (Add stores)
- 9 Workspaces (Create workspaces)

Service Capabilities:

Service	Version
WCS	1.0.0
	1.1.1
WFS	1.0.0
	1.1.0
	2.0.0
WMS	1.1.1
	1.3.0
TMS	1.0.0
WMS-C	1.1.1
WMTS	1.0.0

Security warnings:

- Please read the file `/usr/local/lib/geoserver-2.2.2/data_dir/security/masterpw.info` and remove it afterwards. This file is a **security risk**.
- Please remove the file `/usr/local/lib/geoserver-2.2.2/data_dir/security/users.properties.old` because it contains user passwords in plain text. This file is a **security risk**.
- The default user/group service should use digest password encoding.
- The administrator password for this server has not been changed from the default. It is **highly** recommended that you change it now. Change it

Strong cryptography available

This GeoServer instance is running version **2.2.2**. For more information please contact the administrator.

Abbildung 14: Startseite der Geoserver Web-Oberfläche

2.3.1.1 Anlegen eines Workspace

Die Workspaces in Geoserver dienen der Gruppierung von Daten und ähneln vom Konzept Namensräumen. Datenquellen und Layer können gleichlautende Bezeichnungen haben, solange sie unterschiedlichen Workspaces angehören. Dies bietet außerdem die Möglichkeit, verschiedene ansonsten globale Einstellungen für den Workspace und die enthaltenen Daten spezifisch festzulegen. Die auf der Geoserver Installation vorhandenen Workspaces können unter dem Menüpunkt Data → Workspaces angezeigt werden:

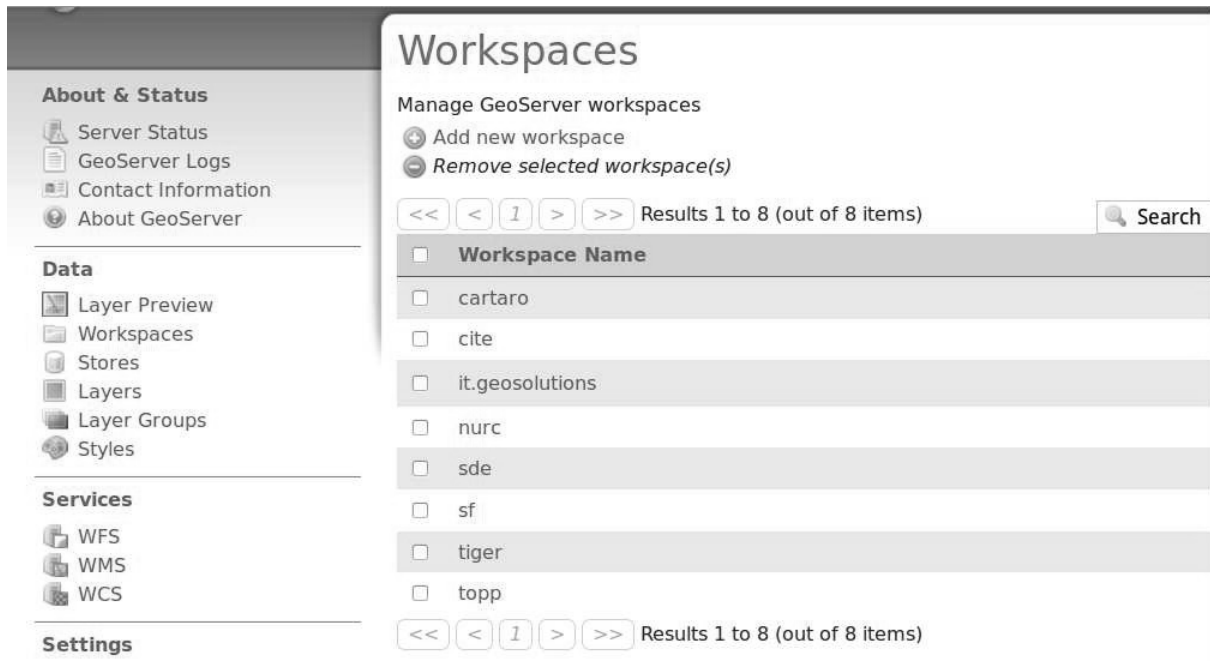


Abbildung 15: Auflistung der vorhandenen Workspaces

Im Rahmen der Lernplattform soll ein neuer Workspace namens nbsample angelegt werden. Dieser soll für den Großteil der weiteren Arbeiten mit Geoserver verwendet werden und wird somit als Standard (Default) festgelegt.

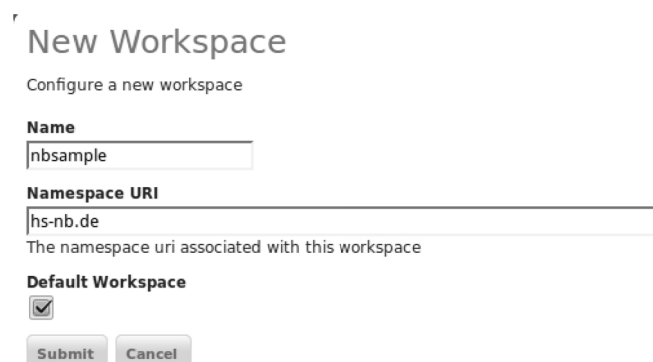


Abbildung 16: Oberfläche zum Anlegen eines neuen Workspace

Ist der Workspace erstellt, kann er durch einfaches Anklicken in der Liste der Workspaces editiert und weitere Eigenschaften festgelegt werden:

Edit Workspace

Edit existing workspace

Name
nbsample

Namespace URI
hs-nb.de
The namespace uri associated with this workspace

Default Workspace

Settings

Enabled

Contact
Daniel Vogel

Organization
Hochschule Neubrandenburg

Position
Student

Address Type
Work

Address
Musterstraße12345

City
Neubrandenburg

State
Mecklenburg-Vorpommern

ZIP code
17033

Country
Deutschland

Telephone

Fax

Email
gi08b35@hs-nb.de
 Verbose Messages
 Verbose Exception Reporting

Number of Decimals
8

Character Set
UTF-8

Proxy Base URL

Services

WMS
 WFS
 WCS

Save **Cancel**

Abbildung 17: Interface zum Editieren des Workspace

Die hier festgelegten Einstellungen ersetzen für den Workspace nbsample die globalen Einstellungen unter dem Menüpunkt Settings. Analog kann jeder der hier angewählten Dienste mit spezifischen Einstellungen konfiguriert werden, die die ansonsten gültigen globalen Einstellungen unter Services überschreiben. Hier wird der WFS so angepasst, das er keine Transaktionen zulässt und somit nicht als WFS-T fungiert. In der Praxis sollen zur Verfügung gestellte Daten (z.B. von Behörden) meist nicht von den Nutzern verändert werden können. Die WFS-T Funktionalität wird vielmehr für spezielle Angebote genutzt, die explizit auf von Nutzern erstellte Daten ausgerichtet sind.

Web Feature Service

Manage the publishing of feature data.

Workspace

nbsample

Service Metadata

Enable WFS

Strict CITE compliance

Maintainer

Online resource

Title

Abstract

This is the reference implementation of WFS 1.0.0 and WFS 1.1.0, supports all WFS operations including Transaction.

Fees

Access Constraints

Current Keywords

WFS
WMS
GEOSERVER

Remove selected

New Keyword

Vocabulary

Add Keyword

Features

Maximum number of features

Return bounding box with every feature

Features

Maximum number of features

Return bounding box with every feature

Service Level

Basic

Transactional

Complete

GML 2

SRS Style

Override GML Attributes

GML 3

SRS Style

Override GML Attributes

GML 3.2

SRS Style

Override GML Attributes

Conformance

Encode canonical WFS schema location

Encode response with

One "featureMembers" element

Multiple "featureMember" elements

SHAPE-ZIP output format

Use ESRI WKT format for SHAPE-ZIP generated .prj files

Submit

Cancel

Abbildung 18: WFS Konfigurationsoptionen (oberer Teil)

Abbildung 19: WFS Konfigurationsoptionen (unterer Teil)

2.3.1.2 Einstellen von SLDs

Geoserver benötigt als WMS wie bereits erwähnt Darstellungsvorschriften, um Geodaten in Karten zu konvertieren. Diese Darstellungsvorschriften bzw. Styles werden über SLD Dateien zur Verfügung gestellt. Unter Data → Styles können die auf der Geoserverinstallation vorhandenen Styles angesehen und neue hinzugefügt werden. Werden keine Styles vom Nutzer definiert, verfügt eine Standardinstallation von Geoserver über grundlegende Styles für verschiedene Datentypen, sodass eine Darstellbarkeit der Daten gewährleistet ist.

No validation errors.

New style

Type a new SLD definition, or use an existing one as a template, or upload a ready made style from your file system. The editor can provide syntax highlight and be brought to full screen. Click on the "validate" button to verify the style is a valid SLD document.

Name
Flaechennutzung_SLD

Workspace
nbsample

Copy from existing style
Choose One Copy ...

```
8 <sld:title/>
9 <sld:IsDefault>1</sld:IsDefault>
10 <sld:FeatureTypeStyle>
11 <sld:Name>name</sld:Name>
12 <sld:FeatureTypeName>Feature</sld:FeatureTypeName>
13 <sld:SemanticTypeIdentifier>generic:geometry</sld:SemanticTypeIdentifier>
14 <sld:SemanticTypeIdentifier>colorbrewer:unique:custom</sld:SemanticTypeIdentifier>
15 <sld:Rule>
16 <sld:Name>rule01</sld:Name>
17 <sld:Title>Acker</sld:Title>
18 <ogc:Filter>
19 <ogc:PropertyIsEqualTo>
20 <ogc:PropertyName>label</ogc:PropertyName>
21 <ogc:Literal>Acker</ogc:Literal>
22 </ogc:PropertyIsEqualTo>
23 </ogc:Filter>
24 <sld:PolygonSymbolizer>
25 <sld:Fill>
26 <sld:CssParameter name="fill">#E4B51A</sld:CssParameter>
27 </sld:Fill>
28 </sld:PolygonSymbolizer>
29 </sld:Rule>
30 <sld:Rule>
31 <sld:Name>rule02</sld:Name>
32 <sld:Title>Bebauung</sld:Title>
33 <ogc:Filter>
34 <ogc:PropertyIsEqualTo>
35 <ogc:PropertyName>label</ogc:PropertyName>
36 <ogc:Literal>Bebauung</ogc:Literal>
37
```

SLD file
Browse... Upload ...

Validate Submit Cancel

Abbildung 20: Hinzufügen neuer SLDs

Die Erzeugung eines neuen Styles ist über den entsprechenden Link in der Style Übersicht möglich. Dies öffnet den in Geoserver integrierten SLD Editor. Damit können Styles direkt als XML-Text eingegeben oder aus Dateien geladen werden. Eine Validierung der erzeugten Anweisungen gegen das XML-Schema des SLD Standards ist ebenfalls möglich. Styles können einem Workspace zugeordnet werden oder auch für alle Workspaces zu Verfügung gestellt werden (keine Angabe des Workspaces in der Eingabemaske).

2.3.1.3 Bereitstellen der Vektordaten

Die Organisation von Datenquellen verläuft in Geoserver über sogenannte Stores. Stores können je nach Art der Datenquelle einen oder mehrere Datensätze beinhalten. Die Veröffentlichung dieser Datensätze erfolgt wiederum über Layer. Die Übersicht über die vorhandenen Datenquellen steht unter Data→ Stores zur Verfügung.

Stores

Manage the stores providing data to GeoServer

⊕ Add new Store

⊖ Remove selected Stores

<< < | > >> Results 1 to 17 (out of 17 items) Search

<input type="checkbox"/>	Data Type	Workspace	Store Name	Type	Enabled?
<input type="checkbox"/>		nurc	arcGridSample	ArcGrid	✓
<input type="checkbox"/>		nurc	img_sample2	WorldImage	⚠
<input type="checkbox"/>		nurc	worldImageSample	WorldImage	✓
<input type="checkbox"/>		nurc	mosaic	ImageMosaic	✓
<input type="checkbox"/>		sf	sfdem	GeoTIFF	✓
<input type="checkbox"/>		nbsample	NB_Mosaik	ImageMosaic	✓
<input type="checkbox"/>		nbsample	NB_Landsat	GeoTIFF	✓
<input type="checkbox"/>		nbsample	ortho8world	WorldImage	✓
<input type="checkbox"/>		sf	sf	Shapefile	✓
<input type="checkbox"/>		nbsample	NBShapes	PostGIS	✓
<input type="checkbox"/>		tiger	nyc	Shapefile	✓
<input type="checkbox"/>		topp	taz_shapes	Shapefile	✓
<input type="checkbox"/>		topp	states_shapefile	Shapefile	✓
<input type="checkbox"/>		cartaro	cartaro	PostGIS	✓

<< < | > >> Results 1 to 17 (out of 17 items)







Abbildung 21: Auflistung der vorhandenen Stores

Die Übersicht der Stores zeigt für jeden Store, ob es sich um Vektor oder Rasterdaten handelt (Symbole unter Datentyp), den zugeordneten Workspace, seinen Namen und seinen Typ entsprechend der von Geoserver angebotenen Zugriffsarten auf Datenquellen. Das Anlegen neuer Stores ist ebenfalls möglich.






New data source

Choose the type of data source you wish to configure

Vector Data Sources

-  Directory of spatial files (shapefiles) - Takes a directory of shapefiles and exposes it as a data store
-  PostGIS - PostGIS Database
-  PostGIS (JNDI) - PostGIS Database (JNDI)
-  Properties - Allows access to Java Property files containing Feature information
-  Shapefile - ESRI(tm) Shapefiles (*.shp)
-  Web Feature Server - The WFSDataStore represents a connection to a Web Feature Server. This connection provides access to the Features published by the server, and the ability to perform transactions on the server (when supported / allowed).

Raster Data Sources

-  ArcGrid - Arc Grid Coverage Format
-  GeoTIFF - Tagged Image File Format with Geographic information
-  Gtopo30 - Gtopo30 Coverage Format
-  ImageMosaic - Image mosaicking plugin
-  WorldImage - A raster file accompanied by a spatial data file

Other Data Sources


-  WMS - Cascades a remote Web Map Service

Abbildung 22: Auswahl der möglichen Arten von Stores

In der Basisinstallation unterstützt Geoserver für Vektordaten nur Shapefiles, PostGIS, WFS und Java Property Dateien. Dies ist jedoch durch Plugins erweiterbar, sodass z.B. Oracle Spatial und Spatialite GDBMS verwendet werden können. Die Vektordaten der Lernplattform sollen als PostGIS Store bereitgestellt werden.

Abbildung 23: Konfigurationsoptionen eines PostGIS Stores (oberer Teil)

Abbildung 24: Konfigurationsoptionen eines PostGIS Stores (unterer Teil)

Die Konfiguration der PostGIS Datenquelle beinhaltet im Wesentlichen die Angabe der notwendigen Verbindungsparameter sowie verschiedene Optionen zur Datenübertragung zwischen Geoserver und dem DBMS.

Die so angelegte Datenquelle beinhaltet alle Datensätze der Tabelle nbsample, wobei VIEWS genauso verwaltet werden können wie Tabellen. Eventuell vorhandene Rasterdatensätze aus nbsample können jedoch auf diese Weise nicht veröffentlicht werden.

Der nächste Schritt besteht darin, einige Datensätze des neuen angelegten Stores über die Erstellung von Layers zu veröffentlichen. Die Übersicht der vorhandenen Layer ist über Data→Layer zugänglich.

Layers

Manage the layers being published by GeoServer

 Add a new resource

 Remove selected resources

<< < 1 2 > >> Results 1 to 25 (out of 29 items)

<input type="checkbox"/>	Type	Workspace	Store	Layer Name	Enabled?	Native SRS
<input type="checkbox"/>		nurc	arcGridSample	Arc_Sample	✓	EPSG:4326
<input type="checkbox"/>		nurc	img_sample2	Pk50095	⚠	EPSG:32633
<input type="checkbox"/>		nurc	worldImageSample	Img_Sample	✓	EPSG:4326
<input type="checkbox"/>		nurc	mosaic	mosaic	✓	EPSG:4326
<input type="checkbox"/>		sf	sf	roads	✓	EPSG:26713
<input type="checkbox"/>		sf	sf	streams	✓	EPSG:26713
<input type="checkbox"/>		sf	sf	archsites	✓	EPSG:26713
<input type="checkbox"/>		sf	sf	restricted	✓	EPSG:26713
<input type="checkbox"/>		sf	sf	bugsites	✓	EPSG:26713
<input type="checkbox"/>		sf	sfdem	sfdem	✓	EPSG:26713
<input type="checkbox"/>		nbsample	NB_Mosaik	NB_Mosaik	✓	EPSG:25833
<input type="checkbox"/>		nbsample	NB_Landsat	neubrandenburg	✓	EPSG:25833
<input type="checkbox"/>		nbsample	NBShapes	ortsumgehung	✓	EPSG:25833
<input type="checkbox"/>		nbsample	NBShapes	flaechennutzung	✓	EPSG:25833
<input type="checkbox"/>		nbsample	NBShapes	verkehr	✓	EPSG:25833
<input type="checkbox"/>		nbsample	NBShapes	baenke	✓	EPSG:25833
<input type="checkbox"/>		nbsample	NBShapes	strandhotel	✓	EPSG:25833
<input type="checkbox"/>		nbsample	ortho8world	orthomosaikoverview8	✓	EPSG:25833
<input type="checkbox"/>		tiger	nyc	poly_landmarks	✓	EPSG:4326
<input type="checkbox"/>		tiger	nyc	giant_polygon	✓	EPSG:4326
<input type="checkbox"/>		tiger	nyc	poi	✓	EPSG:4326
<input type="checkbox"/>		tiger	nyc	tiger_roads	✓	EPSG:4326
<input type="checkbox"/>		topp	taz_shapes	tasmania_roads	✓	EPSG:4326
<input type="checkbox"/>		topp	taz_shapes	tasmania_water_bodies	✓	EPSG:4326
<input type="checkbox"/>		topp	taz_shapes	tasmania_cities	✓	EPSG:4326

<< < 1 2 > >> Results 1 to 25 (out of 29 items)

Abbildung 25: Auflistung der vorhandenen Layer

Die Übersicht der Layer ähnelt der oben gezeigten Übersicht der Stores. Für Layer wird hier zusätzlich ihr zugehöriger Store sowie ihr ursprüngliches Bezugssystem angegeben. Die Erstellung eines neuen Layers ist von hier aus ebenfalls möglich und soll für den Datensatz flaechennutzung dargestellt werden.

New Layer

Add a new layer

Add layer from

You can create a new feature type by manually configuring the attribute names and types. **Create new feature type...**
On databases you can also create a new feature type by configuring a native SQL statement. **Configure new SQL view...**

Here is a list of resources contained in the store 'NBSshapes'. Click on the layer you wish to configure

<< < I > >> Results 0 to 0 (out of 0 items)

Published	Layer name	action
✓	baenke	Publish again
✓	flaechennutzung	Publish again
✓	ortsumgehung	Publish again
✓	strandhotel	Publish again
✓	verkehr	Publish again
	b_str	Publish
	bahn	Publish
	fluesse	Publish

Abbildung 26: Interface zum Anlegen neuer Layer

Die Erzeugung eines neuen Layers setzt voraus, dass ein Store gewählt wird. Der oben erstellte PostGIS Store bietet verschiedene Datensätze, sodass jeder dieser Datensätze über den "publish" Link als Layer veröffentlicht werden kann. Ist der Datensatz bereits als Layer veröffentlicht, führt der Link "Publish again" wie auch die Auswahl eines bestehenden Layers in der Layer Übersicht zur Editierungsmaske des entsprechenden Layers.

Edit Layer

Edit layer data and publishing

nbsample:flaechennutzung

Configure the resource and publishing information for the current layer

Data **Publishing** **Dimensions** **Tile Caching**

Basic Resource Info

Name
flaechennutzung

Title
flaechennutzung

Abstract

Keywords

Current Keywords
flaechennutzung
features Remove selected

New Keyword

Vocabulary

Add Keyword

Metadata links
No metadata links so far

Add link Note only FGDC and TC211 metadata links show up in WMS 1.1.1 capabilities

Coordinate Reference Systems

Native SRS
EPSG:25833 EPSG:ETRS89 / UTM zone 33N...

Declared SRS
EPSG:25833 Find... EPSG:ETRS89 / UTM zone 33N...

SRS handling
Force declared

Bounding Boxes

Native Bounding Box

Min X	Min Y	Max X	Max Y
377,897.9375	5,927,669	390,275.96875	5,942,545

Compute from data

Lat/Lon Bounding Box

Min X	Min Y	Max X	Max Y
13.154042504352	53.483812971655	13.346305221314	53.620197351111

Compute from native bounds

Feature Type Details

Property	Type	Nullable	Min/Max Occurrences
cat	Double	true	0/1
label	String	true	0/1
geom	MultiPolygon	true	0/1

Reload feature type △...

Save Cancel

Abbildung 27: Konfigurationsoptionen eines Vektorlayers (oberer Teil)

Edit Layer

Edit layer data and publishing

nbsample:flaechennutzung

Configure the resource and publishing information for the current layer

Data **Publishing** **Dimensions** **Tile Caching**

Edit Layer

Name
flaechennutzung

Enabled

Advertised

HTTP Settings

Response Cache Headers

Cache Time (seconds)

WMS Settings

Queryable

Default Style
polygon

Additional Styles

Available Styles

- poly_landmarks
- polygon
- pophatch
- population
- rain
- raster
- restricted
- simple_roads
- simple_streams
- tiger_roads

Selected Styles

- Flaechennutzung.SLD

Default Rendering Buffer

Default WMS Path

Authority URLs for this WMS Layer
No authority URLs so far

Add new authority URL

Layer Identifiers
No layer identifiers so far

Add new layer identifier

WMS Attribution

Attribution Text

Attribution Link

Logo URL

Logo Content Type

Logo Image Width
0

Logo Image Height
0

Auto-detect image size and type

KML Format Settings

Default Regionating Attribute
Choose One

Default Regionating Method
Choose One

Features Per Regionated Tile

WFS Settings

Per-Request Feature Limit
0

Maximum number of decimals
0

Save Cancel

Abbildung 28: Konfigurationsoptionen eines Vektorlayers (unterer Teil)

Das Editierfenster für Layer bietet verschiedene Optionen, die über vier Reiter erreichbar sind, wobei auf die für die Veröffentlichung notwendigen kurz eingegangen werden soll. Die Pflichtfelder "Name" und "Titel" stellen respektive den Bezeichner des Layers in WMS Anfragen (z.B. als GET-

Anfrage in der URL) und den für Menschen gut lesbar zu haltenden Bezeichner für Nutzer dar. Das optionale Abstract beinhaltet eine textuelle Beschreibung der Daten. Die beiden anzugebenden Bezugssysteme sind das native System, in dem der Layer gespeichert wird, sowie das deklarierte System, das Geoserver nach außen anzeigt. Für den Fall, dass diese beiden nicht identisch sind, regelt das "SRS Handling" die Vorgehensweise. Über die Bounding Boxen wird die räumliche Ausdehnung der zu veröffentlichenden Daten festgelegt. Diese Angaben sind notwendig, können aber wenn die Ausdehnung des gesamten Datensatzes berücksichtigt werden soll automatisch berechnet werden. Der Reiter "Publishing" erlaubt die Festlegung, ob der Layer "enabled" ist, also für Anfragen an Geoserver zur Verfügung steht. Die Angabe, ob ein Layer "advertised" ist, bestimmt, ob er in "getCapabilities" Anfragen aufgeführt wird und somit z.B. von GIS, die Geoserver üblicherweise über diese Anfrage kontaktieren, als Layer erkannt und aufgeführt wird. Der direkte Aufruf des Layers über seinen Titel ist jedoch auch möglich, wenn dieser nicht "advertised" ist. Weitere wichtige Angaben sind die Abrufbarkeit von Attributdaten des Layers über den WMS "getFeatureInfo" Befehl ("queryable" Kästchen) sowie die Zuordnung von Styles für die Darstellung der Daten in Karten. Dabei wird wenigstens ein Style benötigt, der der Standard ist und automatisch verwendet wird, wenn der Client keine Angaben bezüglich Styles macht. Zusätzlich können weitere Styles zugeordnet werden, die dem Client als Alternative angeboten werden können. Für weitergehende Informationen wird auf [GSdoc] verwiesen.

2.3.1.4 Bereitstellen der Rasterdaten

Die Veröffentlichung von Rasterdaten geschieht nach denselben Konzepten wie die von Vektordaten. Nur die Art der anzulegenden Stores und Teile der Konfiguration der Layer unterscheiden sich.

Raster Data Sources

- ▣ ArcGrid - Arc Grid Coverage Format
- ▣ GeoTIFF - Tagged Image File Format with Geographic information
- ▣ Gtopo30 - Gtopo30 Coverage Format
- ▣ ImageMosaic - Image mosaicking plugin
- ▣ WorldImage - A raster file accompanied by a spatial data file

Abbildung 29: Arten von Rasterdaten Stores

Für die Beispieldaten werden drei neue Stores angelegt, einer für das Landsat GeoTiff, ein WorldImage Store für den generierten Overview und ein Store für das Orthophotomosaik.

Abbildung 30: Anlegen eines GeoTIFF Stores

Abbildung 31: Anlegen eines WorldImage Stores

Abbildung 32: Anlegen eines Stores für Bildmosaik

Die Konfiguration der Stores verläuft ähnlich. Grundsätzlich gehört jeder Store zu einem Workspace und hat einen Namen, eine Beschreibung sowie einen Pfad zu den Rasterdaten. Bei WorldImage Stores muss in dem Ordner, in dem sich die Rasterdatei befindet, auch ein entsprechendes Worldfile und eine .prj Datei liegen. Der ImageMosaic Store verbindet nicht mit einer einzelnen Datei, sondern mit einem Ordner. Sofern die dort vorhandenen Dateien für die Bildung eines Mosaiks geeignet sind, wird automatisch ein solches erstellt und dabei auch ein Kachelindex generiert. Die Erstellung eines Layers aus diesen neuen Datenquellen erfolgt analog zum oben beschriebenen Prozess für Vektordaten.

2.3.1.5 Layergroups

Geoserver ermöglicht die Bildung von sogenannten Layergroups, Zusammenstellungen von Layern, die vom WMS Client wie einzelne Layer abgefragt werden können. Die Erstellung einer neuen Gruppierung ist entsprechend dem Bedienkonzept von Geoserver über das Übersichtsменю unter Data→ Layergroups möglich.

Layer group

Edit the contents of a layer groups

Name

Workspace

Bounds

Min X	Min Y	Max X	Max Y
377,512.54149411	5,927,365.765514	390,621.2577473	5,942,849.706160

Coordinate Reference System
 EPSG:ETRS89 / UTM zone 33N...

Layers

Position	Layer	Default Style	Style	Remove
↓	nbsample:flaechennutzung	<input type="checkbox"/>	Flaechennutzung_SLD	<input type="button" value="⊖"/>
↑ ↓	nbsample:orthomosaikoverview8	<input type="checkbox"/>	raster	<input type="button" value="⊖"/>
↑	nbsample:strandhotel	<input type="checkbox"/>	point	<input type="button" value="⊖"/>

<< < | > >> Results 1 to 3 (out of 3 items)

Tile cache configuration

Create a cached layer for this layer group

Enable tile caching for this layer

Metatiling factors
 tiles wide by tiles high

Gutter size in pixels

Tile Image Formats

image/png
 image/png8
 image/jpeg
 image/gif

Parameter Filters

Available gridsets

Gridset	Published zoom levels	Cached zoom levels	Grid subset bounds
EPSG:900913	<input type="text" value="Min"/> / <input type="text" value="Max"/>	<input type="text" value="Min"/> / <input type="text" value="Max"/>	Dynamic <input type="button" value="⊖"/>
EPSG:4326	<input type="text" value="Min"/> / <input type="text" value="Max"/>	<input type="text" value="Min"/> / <input type="text" value="Max"/>	Dynamic <input type="button" value="⊖"/>

Add grid subset:

Authority URLs for this WMS Layer
 No authority URLs so far

Layer Identifiers
 No layer identifiers so far

Abbildung 33: Interface zum Anlegen einer Layer group

Wie andere Layer benötigt die Layergroup einen Namen und Angaben über ihre räumliche Ausdehnung, wobei diese ebenfalls aus den Daten der enthaltenen Layer berechnet werden kann. Sie veröffentlicht nicht direkt Daten eines Stores, sondern die Zusammenstellung anderer einfacher Layer. Dabei können in der oben dargestellten Oberfläche Layer für die Gruppe hinzugefügt, Styles festgelegt und die Anordnung bzw. Überlagerung der Layer bestimmt werden. Bei der Anordnung der einzelnen Layer steht der in der Liste am weitesten unten aufgeführte am weitesten im Vordergrund. Layergroups sind nur für die Verwendung mit Kartendiensten bzw. WMS interessant.



Abbildung 34: Voransicht der erstellten Layer group

2.3.1.6 Voransicht

Geoserver bietet die Möglichkeit die vorhandenen Layer unter Data → Layer Preview in allen von Geoserver für diesen Layer unterstützten Datenformaten sowie über dynamisch erzeugte OpenLayers Seiten abzurufen und somit das Endergebnis der Konfiguration zu überprüfen.

Type	Name	Title	Common Formats
	topp:states	USA Population	OpenLayers KML GML
	cartaro:messungen	Messungen	OpenLayers KML GML
	cartaro:capitals	Capitals	OpenLayers KML GML
	nbsample:NB_Map1		OpenLayers KML
	spearfish		OpenLayers KML
	tiger-ny		OpenLayers KML
	tasmania		OpenLayers KML

Abbildung 35: Interface zur Abfrage von Voransichten

2.3.1.7 Realisierung in der Lernplattform

Die über die Web-Oberfläche getätigten Einstellungen schlagen sich in der Entstehung und Modifikation von Dateien in den Verzeichnissen der Geoserverinstallation nieder. Im Unterverzeichnis `./serverfiles/geoserver/data/` befinden sich die notwendigen Dateien, um Geoserver entsprechend der Lernplattform zu konfigurieren. Die Konfiguration wird durch das Skript `GeoserverSetup.sh` durchgeführt.

Hinweis:

Entsprechend der dargestellten Vorgehensweise wurde ein weiterer Workplace namens "nbediting" eingerichtet, der den transaktionalen WFS unterstützt und die drei dafür vorgesehenen PostGIS Datensätze veröffentlicht.

2.3.2 Bereitstellung mithilfe von Mapserver

Mapserver wird über textbasierte Konfigurationsdateien gesteuert, die mit einer eigenen Syntax kodiert sind. Die so erstellten Einstellungen können sehr komplexe Form annehmen und decken eine Vielzahl von Anforderungen ab. Im Folgenden werden die notwendigen Konfigurationen zur Veröffentlichung der Beispieldaten vorgestellt. Für umfassendere Informationen zur Nutzung und Administration von Mapserver wird auf [MSdoc] verwiesen. Im Folgenden sollen zugrunde liegende Konzepte für die Konfiguration von Mapserver erläutert werden. Dies beinhaltet nicht die Behandlung jedes in den Beispieldateien verwendeten Parameters. Diese sind entweder weitgehend selbsterklärend oder in den Beispieldateien durch Kommentare erklärt.

2.3.2.1 Das Mapfile

Das Mapfile ist eine Textdatei mit Konfigurationsanweisungen durch die festgelegt wird, welche Datensätze veröffentlicht werden und auf welche Weise dies geschehen soll. Bei der Verwendung von Mapserver über die CGI Schnittstelle, auf die sich hier konzentriert werden soll, wird das Mapfile als Parameter in der Mapserver aufrufenden URL übergeben. Dies geschieht nach folgendem Muster:

```
http://host/cgi-bin/mapserv?map=Pfad zum Mapfile/Mapfile.map
```

Diese URL stellt die Adresse des durch das Mapfile definierten GeodatenServers da. Die für die Lernplattform erstellten Mapfiles sind jeweils selbstständig und beinhalten alle Konfigurationen in ihren jeweiligen Hauptdokumenten. Grundsätzlich ist es jedoch möglich, über `INCLUDE` Parameter beliebige Elemente des Mapfiles aus separaten Dateien zu laden solange ein gültiges Mapfile mit der Endung ".map" diese Elemente zusammenfasst.

2.3.2.2 allgemeine Kartenparameter

Das Mapfile besteht aus hierarchisch geordneten Elementen, die im einfachsten Fall aus einem Bezeichner und einem Wert für diesen bestehen oder aber aus einer Reihe von Kindelementen. Hat ein Element Kindelemente, so wird das Ende der Definition dieser durch das Schlüsselwort `End` gekennzeichnet. In dieser Hinsicht ist das Mapfile ähnlich der Baumstruktur von XML aufgebaut.

Als Beispiel soll dieses `STYLE` Element dienen:

```
STYLE                #Elternelement
  COLOR 0 0 255      #Kindelement mit Bezeichner COLOR und
  OUTLINECOLOR 0 0 0 #Zahlenwerten als Inhalt
END                  #END beendet die Deklaration von STYLE
```

Dieses dient vorerst nur der Beschreibung der Syntaxstruktur. Auf Styling Anweisungen wird unter 2.3.2.6 inhaltlich eingegangen werden.

Jedes Mapfile hat genau ein `MAP` Element, das alle anderen Elemente beinhaltet. Seine direkten Kindelemente dienen hauptsächlich für die Festlegung globaler Konfigurationen, die alle Layer betreffen und der Deklaration von `LAYER` Elementen, die die eigentlichen zu veröffentlichenden Layer konfigurieren. Solche Elemente sind beispielsweise `SIZE`, das die Größe der zurückgelieferten Karte in Pixeln beschreibt (nicht relevant für Datendienste wie WFS), `IMAGECOLOR`, das die Hintergrundfarbe (für Gebiete, die nicht von Layern bedeckt sind) von Karten angibt oder `SHAPEPATH`, das den Standardpfad zu dateibasierten Daten definiert. Wird auf dieser Ebene ein `PROJECTION` Element definiert, gilt dieses für alle Layer solange diese keine eigenen `PROJECTION` Elemente auf ihrer Ebene (innerhalb des `LAYER` Elements) definieren.

2.3.2.3 Konfiguration Vektorlayer

`LAYER` Elemente konfigurieren die von Mapserver abrufbaren Datensätze und deren Erscheinung. Jedes `LAYER` Element beinhaltet wenigstens folgende Elemente:

- `NAME`, welches den Namen des Layers für Abfragen festlegt
- `STATUS`, welches festlegt ob der Layer abrufbar (Wert "ON"), nur im Mapfile definiert aber von außen nicht abrufbar (Wert "OFF") oder ohne spezielle Anfrage Teil jeder gelieferten Karte ist (Wert "DEFAULT")
- `TYPE`, welches die Art der Daten des Layers beschreibt. Der Wert von `TYPE` ist entweder `RASTER`, `POLYGON`, `LINE` oder `POINT`, um Raster- bzw. die verschiedenen Arten von Vektorlayern zu beschreiben.

- Einer Angabe zur Datenquelle des Layers. Hier wird oft ein `DATA` Element verwendet, das auf entsprechende Dateien im Dateisystem zeigt. Ist der Wert von `DATA` ein Dateipfad, wird dieser immer relativ zu dem im `SHAPEPATH` Element angegebenen Pfad interpretiert. Bei bestimmten Datenquellen sind zusätzliche Elemente notwendig. Bei Nutzung einer PostGIS Datenbank sind dies die Elemente `CONNECTIONTYPE` und `CONNECTION`. Neben Datenquellen, deren Zugriffsmethoden native Bestandteil von Mapserver sind, besteht ebenfalls die Möglichkeit Verbindungen über die OGR Bibliothek herzustellen.

Darüber hinaus können `LAYER` Objekte weitere Deklarationen enthalten. Besonders relevant sind hier das `PROJECTION` Element, das das Bezugssystem des Layers festlegt sowie die Styling Informationen und Metadaten, die weiter unten besprochen werden.

2.3.2.4 Konfiguration Rasterlayer

Rasterlayer folgen grundsätzlich den gleichen Konventionen wie Vektorlayer. Das `TYPE` Element von Raster Layern hat immer den Wert `RASTER`. Weitere Unterscheidungen sind nicht notwendig. Die Datenquellen von Rasterlayern können sein:

- Rasterdateien - Mapserver unterstützt direkt die Formate (Geo)Tiff, Png, Gif, Jpeg und Erdas LAN/.GIS, sofern die Mapserver Treiber aktiviert sind. Für diese als auch eine große Zahl anderer Formate kann Mapserver jedoch auch die GDAL Bibliothek verwenden, was laut [MSdoc] auch für potentiell von Mapserver direkt unterstützte Formate vorzuziehen ist.
- Kachelindexe - In diesem Fall ersetzt das `TILEINDEX` Element, welches den Pfad zu einer entsprechenden Indexdatei wie dem in 2.2.3.3 angefertigten `Mosaik.shp` zusammen mit einem `TILEITEM` Element, das das Attribut mit den Dateipfaden zu den Rasterdaten definiert, das `DATA` Element. Solche Layer stellen ein Mosaik aus den einzelnen Bildkacheln dar, können aber als ein einziger Rasterlayer angesprochen werden.
- Datenbanken - Zumindest im Fall von PostGIS Raster war laut [PG2] eine direkte Nutzung von in PostGIS Datenbanken gespeicherten Rastern möglich. Das in der Quelle gezeigte Beispiel bezieht sich auf eine frühere Version von PostGIS Raster (damals WKTRaster) und Version 1.8 der GDAL Bibliothek. Unter den Bedingungen der OsGeo Live Installation funktionierten entsprechende Layer Konfigurationen nicht. Als Grund dafür kann eine Inkompatibilität zwischen GDAL 1.9 und PostGIS 2.0 vermutet werden, was bedeutet, dass dieses Problem temporärer Natur sein wird.

Für Rasterlayer bietet Mapserver ein `PROCESSING` Element, das dazu dient, eine große Zahl verschiedener Modifikationen an dem dargestellten Raster vorzunehmen. In der Lernplattform wurde das Element genutzt, um die Bandzuordnung von `neubrandenburg.tif` zu verändern, sodass Vegetationsflächen in rot dargestellt werden.

2.3.2.5 Mapserver als WMS

Um als WMS zu fungieren benötigt ein ansonsten voll konfiguriertes Mapfile nur `METADATA` Elemente auf Ebene `MAP`- und `LAYER` Elemente, welche im wesentlichen die vom WMS Standard vorgegebenen Angaben beinhalten. Die in den Beispiel-Mapfiles enthaltenen Metadaten beschränken sich auf die für die Funktion notwendigen bzw. dringend empfohlenen Angaben. Je nach Anwendung ist es ggf. ratsamer, umfassende Metadatenangaben zu machen. Die auf `MAP`-Ebene notwendigen Angaben sind:

- `"wms_title"` - Das Element definiert den Namen des Servers für WMS Abfragen.
- `"wms_onlineresource"` - Das Element gibt die URL des WMS für `GetCapabilities` Anfragen an .
- `"wms_srs"` - das Element legt fest welche Projektionen der WMS für die Daten nach außen anbietet .
- `"wms_enable_request" "*" - Das Element legt fest, welche der WMS Anfragen Mapserver beantwortet. Das Sternsymbol dient als Platzhalter und erlaubt alle Anfragearten.`
- `"wms_server_version" "1.1.1" - Das Element legt die von Mapserver angebotene WMS Version fest.`

Auf der `LAYER`-Ebene wurden folgende Metadaten gesetzt:

- `"wms_title"` - Das Element definiert den Namen des Layers für WMS Abfragen und ist zwingend notwendig.
- `"wms_srs"` - das Element legt fest, welche Projektionen der WMS für die Daten nach außen für diesen Layer anbietet. Ist dieses Element auf der `MAP`-Ebene vorhanden, muss es hier nicht zwingend gesetzt werden. Ist kein globales `wms_srs` Element definiert, ist die Angabe hier notwendig und es können die Elemente auf der `LAYER`-Ebene genutzt werden, um für verschiedene Layer unterschiedliche Bezugssysteme anzubieten. Es wird empfohlen dieses Element grundsätzlich zu setzen, auch wenn es nicht zwingend nötig ist

2.3.2.6 Styling und SLD

Mapserver bietet ein eigenes System zur Gestaltung von Layern. Werden keine Gestaltungsanweisungen ins Mapfile eingefügt, können Layer ansonsten korrekt konfiguriert sein und ohne Fehlermeldung abgerufen werden, während die übermittelte Karte lediglich die festgelegte Hintergrundfarbe darstellt. Fehlende oder inkorrekte Gestaltungselemente sind ähnlich wie valide, aber inhaltlich inkorrekte SLDs bei Geoserver eine mögliche Fehlerquelle für die Arbeit mit Mapserver. Die an der Gestaltung von Layern hauptsächlich beteiligten Elemente sind:

- `STYLE` - Dieses Element enthält Kindelemente, die direkte Darstellungsanweisungen enthalten wie das `COLOR` oder `WIDTH` Element.
- `CLASS` - Alle `STYLE` Elemente sind immer von je einem `CLASS` Element umschlossen. In Verbindung mit `CLASSITEM` Elementen, die festlegen, auf welches Attribut sich die Regeln der folgenden `CLASS` beziehen, und `EXPRESSION` Kindelementen, die u.a. genutzt werden können, um den Wert eines Attributes zu bezeichnen, können Regeln zur differenzierten Gestaltung von Layern erstellt werden.
- `SYMBOL` - Diese Elemente definieren Kartensymbole und werden i.d.R. außerhalb des Mapfiles definiert und über das `MAP`-Level Element `SYMBOLSET` geladen, können aber auch direkt im Mapfile definiert werden. Die Einbindung der Symbole in die Layer erfolgt als Kindelement eines `STYLE` Elements.

Die Möglichkeit zur Nutzung von SLDs steht nur begrenzt, durch Übergeben eines Dateipfades oder der Style Definition als Parameter in der URL, mit der Mapserver angefragt wird, zur Verfügung. Dies hat die Nachteile, dass nur ein einzelnes SLD übergeben werden kann und dass diese Option nur nutzbar ist, wenn die entsprechende Anwendung das direkte Editieren der aufrufenden URL unterstützt. Dies ist beispielsweise bei Desktop-GIS nicht ohne weiteres möglich. Aus diesen Gründen empfiehlt sich für die Arbeit mit Mapserver weiterhin die Nutzung der nativen `STYLE` Elemente der Mapfiles.

2.3.2.7 Mapserver als WFS/WCS

Die WFS und WCS sind Datendienste und benötigen daher keine Styling Anweisungen, da die von ihnen übermittelten Rohdaten nicht primär für eine Kartendarstellung dienen bzw. wenn gewünscht durch den Client gestaltet werden. Abgesehen davon erfordert die Konfiguration eines Mapfiles als WFS bzw. WCS, ähnlich wie beim WMS, lediglich die Angabe geeigneter Metadaten Elemente. Die auf `MAP`-Ebene notwendigen Angaben sind:

- "wfs_title" - Das Element definiert den Namen des Servers für WFS Abfragen.
 - "wfs_onlineresource" - Das Element gibt die URL des WFS für GetCapabilities Anfragen an.
 - "wfs_srs" - Das Element legt fest, welche Projektionen der WFS für die Daten nach außen anbietet.
 - "wfs_enable_request" "*" - Das Element legt fest, welche der WFS Anfragen Mapserver beantwortet. Das Sternsymbol dient als Platzhalter und erlaubt alle Anfragearten.
- ➔ Die oben genannten Metadaten sind analog zu denen des WMS und können durch Elemente mit dem Präfix "ows" ersetzt werden und dadurch innerhalb desselben Mapfiles von WMS und WFS verwendet werden. Dies trifft auch auf alle folgenden Elemente zu, die für mehrere Webservices die gleiche Antwort zurückgeben. So kann die Anzahl der notwendigen Deklarationen in Multi-Webservice Mapfiles gesenkt werden.
- "wfs_server_version" - Das Element legt die von Mapserver angebotene WFS Version fest. (Hier ist beispielsweise kein ows* möglich, da die Versionsnummern der verschiedenen Webservices natürlich nicht gleich sind.)
 - "wcs_label" - Das Element beschreibt einen für den Nutzer bestimmten Namen des WCS Servers.
 - "wcs_onlineresource" - Das Element gibt analog zu WMS und WFS die URL des WCS für GetCapabilities Anfragen an.
- "wcs_enable_request"- Das Element legt fest, welche der WCS Anfragen Mapserver beantwortet. Das Sternsymbol dient als Platzhalter und erlaubt alle Anfragearten. Sollen spezifische Anfragearten erlaubt werden, ist eine separate Definition für WMS, WFS und WCS erforderlich, da die Anfragearten sich für die einzelnen Dienste unterscheiden. Der Sternplatzhalter kann jedoch mit dem "ows" Präfix verwendet werden.

Auf der LAYER-Ebene wurden folgende Metadaten gesetzt:

- "wfs_title"- Das Element definiert den Namen des Layers für WFS Abfragen und ist zwingend notwendig. Es verhält sich analog zum WMS Gegenstück. Das "ows" Präfix kann verwendet werden.

- "wfs_srs" - Das Element gibt das Bezugssystem an, das nach außen für den Layer angeboten wird. Version 1.0.0 des WFS unterstützt hier nur eine einzige Projektion.
- "gml_include_items" - Das Element legt fest, welche Attributdaten in die veröffentlichten Vektordaten eingebunden werden sollen.
- "gml_featureid" - Das Element legt fest welches Attribut als FeatureID, einer Art Primärschlüssel, dient. Es muss zwingend angegeben werden.
- "wfs_enable_request" "*" - Das Element legt auf LAYER-Ebene fest, welche der WFS Anfragen Mapserver beantwortet. Das Sternsymbol dient als Platzhalter und erlaubt alle Anfragearten.
- "wcs_label" - Das Element beschreibt einen für den Nutzer bestimmten Namen des WCS Layers.
- "wcs_rangeset_name" - Das Element stellt eine notwendige Angabe für "DescribeCoverage" Abfragen.
- "wcs_rangeset_label" - Das Element stellt eine notwendige Angabe für "DescribeCoverage" Abfragen.

2.3.2.8 WFS-T mit TinyOWS

Der von Mapserver angebotene WFS ist zumindest in der aktuellen Version grundsätzlich nicht transaktional. WFS-T wird durch TinyOWS, welches ein Teil der Mapserver Suite ist, zur Verfügung gestellt. TinyOWS ist eng an PostGIS gebunden und funktioniert derzeit nur mit diesem Speichersystem. Es ist möglich den Dienst sowohl über Mapfiles, als auch eigene XML-basierte Konfigurationsdateien aufzusetzen. Die TinyOWS Mapfiles besitzen verschiedene Eigenheiten und unterstützen nicht die volle Zahl an Mapfile Elementen. Dies ist darin begründet, dass TinyOWS ursprünglich ein eigenständiges Projekt war und erst vor einiger Zeit in die Mapserver Suite integriert wurde. Die notwendigen Parameter sind weitgehend selbsterklärend und definieren hauptsächlich die Pfade zu TinyOWS, Namen und Bezeichner des Servers und der Layer sowie die Datenbankverbindung. Für Einzelheiten wird auf die Beispieldatei in der Lernplattform verwiesen.

2.3.2.9 Realisierung in der Lernplattform

Im Unterverzeichnis `./serverfiles/mapserver/` stehen die drei Dateien `Lernplattform_WFS_WCS.map`, `Lernplattform_WMS.map` und `tinyows.xml` zur Verfügung. Die beiden Mapfiles können wie oben dargestellt aufgerufen werden, während tinyOWS

Konfigurationsdatei durch das Skript `tinyOWSsetup.sh` in den Suchpfad von TinyOWS gelinkt wird und dadurch nutzbar wird.

2.4 Web-basierter Zugriff auf die Daten

Der Zugriff auf Geodaten über das Web kann sowohl mit thick Clients wie Desktop-GIS als auch mit thin Clients, also über Browser erfolgen. Der Einsatz von Desktop-GIS wie etwa QGIS für diesen Zweck gestaltet sich mittlerweile sehr einfach und komfortabel und bedarf im Kontext dieser Arbeit keiner Erklärung. Die Ausparung dieses Bereiches soll jedoch keine verringerte Bedeutung der Nutzung solcher Clients andeuten. Vielmehr sind diese für die intensive Arbeit mit GIS und anspruchsvolle Aufgabenstellungen meist geeigneter und Browser-basierten Lösungen deutlich vorzuziehen.

In diesem Abschnitt sollen anhand verschiedener Aufgabenstellungen die drei Systeme OpenLayers, Legato und Leaflet im Bezug auf ihre Nutzung erläutert und miteinander verglichen werden. Dabei sollen hier nur die grundlegenden Aspekte dargestellt werden. Die vollständige Umsetzung ist wieder in den Beispielen der Lernplattform verfügbar. Die drei verwendeten Systeme wurden aufgrund ihrer unterschiedlichen Ansätze und Zielstellungen gewählt.

2.4.1 Einfache Karte mit WMS Daten

Die erste Aufgabenstellung besteht darin eine einfache Karte mit einem WMS Layer als Datenquelle zu erzeugen. Sie dient hauptsächlich dazu die grundlegenden Komponenten vorzustellen, die für die Kartenerzeugung mit den unterschiedlichen Systemen notwendig sind.

Allen drei Systemen ist gemein, dass sie folgende Strukturen und Aktionen benötigen:

- Wenigstens ein html Blockelement, das die Kartenstrukturen aufnimmt. Eine Kartenanwendung kann aus mehreren Komponenten (z.B. Schaltflächen, Legenden und in jedem Fall das Hauptkartenobjekt) bestehen, die jeweils im html verankert werden müssen.
- Das Laden der Bibliotheken selbst. Dabei beinhalten alle drei Bibliotheken neben Javascript auch CSS Dateien. Openlayers- und Leafletkarten benötigen jeweils nur eine dieser Bibliotheken, während Legato auf OpenLayers aufsetzt und somit beide Projekte einbinden muss.
- Zusätzliche CSS Dateien zur Gestaltung der Kartenfenster und Steuerelemente. Wenigstens ist die Definition der Größe des Kartenfensters notwendig, da die Bibliotheken aufgrund der Unterschiedlichkeit der von den Nutzern verwendeten Endgeräte diesbezüglich keine sinnvollen Standardwerte festlegen können.

- Die Erstellung der eigenen Karte. Bei OpenLayers und Leaflet bedeutet dies die Programmierung eines Javascript Skriptes unter Nutzung der Funktionen und Datenstrukturen der jeweiligen Bibliothek. Legato ermöglicht es, eigene Kartenanwendungen durch das Editieren von XML-Dateien zu konfigurieren. Dies soll die Erstellung von web-basierten Kartenanwendungen auch für Personen ohne Programmierkenntnisse ermöglichen.

Im Bezug zur Erstellung der Karte bestand die Aufgabe für Leaflet und OpenLayers im Wesentlichen darin, ein Kartenobjekt und einen WMS Layer über die entsprechenden Konstruktoren zu definieren und miteinander zu verknüpfen. Die Umsetzung mit Legato erforderte die Nutzung der Markups zur Definition eines WMS-Layers.

Während OpenLayers und damit zumindest theoretisch auch Legato es ermöglichen die Daten in ihrer nativen Projektion zu erstellen, solange keine Transformationen nötig sind, unterstützt Leaflet nur die Projektionen EPSG3857 (z.B. OSM, GoogleMaps), EPSG4326 (WGS84 lat/lon), EPSG3395 und ein kartesisches System. Folglich erscheint die Karte hier verzerrt.

2.4.2 Karte mit mehreren Layern

Diese Aufgabenstellung besteht darin, mehrere Layer in eine Karte einzubinden und die minimalen Werkzeuge für ihre Verwaltung zur Verfügung zu stellen. Dabei sollten möglichst lokale WMS Daten zusammen mit einer anderen Datenquelle verwendet werden. Als fremde Datenquelle wurde ein OpenStreetMap Layer gewählt. OpenLayers und Leaflet verhalten sich wieder im Grundsatz ähnlich bei der Definition mehrerer Layer und ihrer Einbindung in die Karte. Ein Unterschied besteht darin, dass relevante Eigenschaften wie der nach außen angezeigte Titel eines Layers oder seine Eigenschaft als Basis- oder Overlaylayer (Overlaylayer können beliebig ein- und ausgeblendet werden, es ist immer genau ein Basislayer einblendbar) bei OpenLayers Attribute des entsprechenden Layers darstellen und bei Leaflet erst durch das Kontrollelement zur Layerverwaltung zugewiesen werden. Legato bietet die Möglichkeit die Definition der Layer, des Karteninhaltes, in einer separaten Konfigurationsdatei von der restlichen Definition der Karte zu trennen. In dieser Datei können gemeinsame Parameter wie Bezugssystem und räumliche Ausdehnung der Karte sowie eine beliebige Anzahl von Layern deklariert werden. Die vom WMS-Server durchgeführte Reprojektion der WMS Layer in das Spherical Mercator System (EPSG3857 oder EPSG900913) konnte von Leaflet und OpenLayers erfolgreich veranlasst werden. Es ist davon auszugehen, dass dies prinzipiell auch mit Legato möglich sein sollte, da dem System alle OpenLayers Funktionen zur Verfügung stehen. Dennoch konnte eine erfolgreiche Reprojektion der

WMS Layer in das Bezugssystem von OSM leider nicht veranlasst werden, da die nötigen Konfigurationsanweisungen, sofern bereits vorhanden, nicht in der zur Verfügung stehenden Dokumentation auffindbar waren. Folglich konnte hier nur mit lokalen Daten gearbeitet werden.

2.4.3 Karte mit umfassenden Kontrollelementen

Diese Aufgabenstellung besteht darin die Möglichkeiten der thin Client Systeme die erzeugten Karten mit Steuerelementen zu versehen einzusetzen. Leaflet bietet weniger Steuerelemente als die anderen Systeme, wobei fast alle vorhandenen Elemente standardmäßiger Bestandteil aller Mapobjekte sind und üblicherweise nicht extra erstellt werden müssen. OpenLayers bietet alle Steuerelemente, die für Karten gebräuchlich sind und stellt somit gewissermaßen den Standard dar. Legato erlaubt den Zugriff auf alle Steuerelemente von Openlayers und erweitert diese um eigene Funktionen, wie die Möglichkeit die Größe des Kartenfensters zu verändern und eine erweiterte Legende, die es erlaubt die Reihenfolge der Layer zu verändern oder sie aus der Legende zu entfernen.

2.4.4 Vergleich und Einschätzung

Aufgrund der Arbeit mit den Systemen und den zur Verfügung stehenden Informationen aus den Quellen zieht der Autor folgendes Fazit:

- OpenLayers ist das älteste der getesteten Systeme und besitzt den größten Funktionsumfang, wobei dies auch Aufgabenbereiche betrifft, die hier nicht betrachtet werden konnten. Die über die Anwendungsbeispiele von [OL] und [OLdoc] zur Verfügung stehende Dokumentation ist vielfach hilfreich aber gerade im Hinblick auf die Lebensdauer des Projektes in vielen Gebieten noch unzureichend.
- Legato ist durch das Konzept der Konfiguration über XML und die mögliche Trennung von Daten und Code im Ansatz sehr interessant. Derzeit ist jedoch die Nutzung der vollen Funktionalität von OpenLayers über Legato noch nicht möglich. Viele vorhandene Funktionen sind noch experimentell und/oder undokumentiert. Die Performance war im Verhältnis zu den beiden anderen Systemen am langsamsten. In Anbetracht der Tatsache, dass es sich um ein noch relativ neues Projekt handelt, ist eine Verbesserung in den problematischen Bereichen in der Zukunft jedoch nicht unwahrscheinlich.
- Leaflet hat von allen Systemen den geringsten Funktionsumfang und die beste Performance. Die vorhandenen Funktionen sind darauf ausgerichtet üblichste Anforderungen abzudecken, sodass dies nur für anspruchsvolle Projekte einen Mangel darstellt. Die Dokumentation ist ausführlich, insbesondere für ein relativ junges Projekt.

2.4.5 Umsetzung in der Lernplattform

Die Beispiele befinden sich im Unterordner `./thinClient` der Lernplattform in nach System und Aufgabe geordneten Verzeichnissen. Nach der Ausführung von `thinClientSetup.sh` stehen die Beispiele unter `http://localhost/thinClient/` zur Verfügung. Die Beispiele wurden erstellt mit Informationen und sind angelehnt an Beispiele aus den Dokumentationen der Projekte unter [OL] [OLdoc] [Legato] [Leaflet] sowie den OpenGeo Workshop [OG-OLWS].

3 Fazit und Ausblick

Ziel der Arbeit war es zum einen, eine auf Anwendungsbeispielen begründete Lernplattform zu erstellen, und zum anderen einen allgemeinen Überblick über Web-GIS inklusive bestehender Technologien und Produkte zu geben.

Die entstandene Lernplattform demonstriert die Nutzung einiger verbreiteter Web-GIS Komponenten in den verschiedenen Bereichen von den Ausgangsdaten bis zur Aufbereitung für den Endnutzer im Client.

Auf Basis der Betrachtungen im ersten Teil wird ganz klar ersichtlich, dass eine große Vielzahl möglicher Einsatzgebiete und -anforderungen sowie verfügbarer Technologien für den Aufbau von Web-GIS existiert. Aufgrund der begrenzten zur Verfügung stehenden Zeit und der Breite des Themengebietes konnte folglich nur ein Teil der derzeit verfügbaren Technologien und Produkte beleuchtet werden. Gleichzeitig unterliegt gerade das Gebiet rund um Web-GIS einer schnellen Entwicklung, sodass in kurzer Zeit neue Konzepte, Technologien und Produkte verfügbar werden können. Viele bestehende Produkte werden ebenfalls ständig weiterentwickelt und werden dabei ggf. für neue Einsatzgebiete interessant.

Im Rahmen der Lehre wäre eine Fortführung und Erweiterung der Lernplattform eine Perspektive. Mögliche Erweiterungspotenziale liegen beispielsweise in der Umsetzung von Aufgaben mit Komponenten, die aus Zeitgründen nicht einbezogen wurden, der Einbindung neuer, nach dem Zeitpunkt des Schreibens entstandener Komponenten, der Umsetzung komplexerer Anwendungen mit spezifischen Zielsetzungen und deren Einbindung in die hier geschaffene Grundlage sowie der im Laufe der Zeit notwendig werdenden Anpassung der Lernplattform an neue Versionen der verwendeten Komponenten und des Basissystems.

Quellenverzeichnis

[Steiniger&Hunter2012] Steiniger, S., and Hunter : Free and open source GIS software for building a spatial data infrastructure. In E. Bocher and M. Neteler (eds): Geospatial Free and Open Source Software in the 21st Century: Proceedings of the first Open Source Geospatial Research Symposium

[SFSQL11] Open Geospatial Consortium Inc(22.11.2005): OpenGIS® Implementation Specification for Geographic information - Simple feature access - Part 1: Common architecture, (Version: 1.1.0)

[PG2] PostGIS 2.0.2SVN Manual SVN Revision (10640)
URL: <http://postgis.net/docs/manual-2.0/>
(05.04.20013)

[SLite] Projektseite von SpatiaLite und dessen Partnerprojekten
URL: <http://www.gaia-gis.it/gaia-sins/>
(02.04.2013)

[Rasdaman] Projektseite des Rasdamanprojektes
URL: <http://rasdaman.com/>
(03.04.2013)

[WMSspec] Open Geospatial Consortium Inc.(15.03.2006): OpenGIS® Web Map Server Implementation Specification (Version: 1.3.0)

[WMTSpec] Open Geospatial Consortium Inc.(06.04.2010): OpenGIS® Web Map Tile Service Implementation Standard (Version: 1.0.0)

[Gsmain] Projektseite des Geoserver Projekts
URL: <http://geoserver.org/display/GEOS/Welcome>
(05.04.2013)

[umnMS] Projektseite der Mapserver Suite
URL: <http://www.mapserver.org/>
(05.07.2013)

[deg3] Projektseite des Deegree3 Projektes
URL: <http://www.deegree.org/>
(05.04.2013)

[GWCACHE] Projektseite von GeoWebCache
URL: <http://geowebcache.org/>
(05.04.2013)

[WFSspec] Open Geospatial Consortium Inc.(02.11.2010): OpenGIS Web Feature Service 2.0 Interface Standard (Version: 2.0.0)

[WFS-Gspec] Open Geospatial Consortium(17.02.1012): Gazetteer Service -Application Profile of the Web Feature Service Best Practice (Version 1.0)

[WCSspec] Open Geospatial Consortium Inc.(27.10.2010): OGC® WCS 2.0 Interface Standard - Core (Version: 2.0.0)

[GSdoc] offizielle Geoserver Dokumentation
URL: <http://docs.geoserver.org/>
(07.07.2013)

[MSdoc] Offizielle Dokumentation der Mapserver Suite
URL: <http://www.mapserver.org/documentation.html>
(07.07.2013)

[FeatServ] Projektseite von Featureserver
URL: <http://featureserver.org/>
(05.04.2013)

[WPSspec] Open Geospatial Consortium Inc.(08.06.2007): OpenGIS® Web Processing Service (Version: 1.0.0)

[WCPSspec] Open Geospatial Consortium Inc.(25.03.2009): Web Coverage Service (WCS) — ProcessCoverages Extension (Version: 1.0.0)

[ZOO] Projektseite der ZOO WPS Plattform
URL: <http://zoo-project.org/>
(05.04.2013)

[52NWPS] Projektseite des 52°North WPS und verwandter Produkte
URL: <http://52north.org/communities/geoprocessing/>
(07.04.2013)

[PyWPS] Projektseite des PyWPS Dienstes
URL: <http://pywps.wald.intevation.org/>
(07.04.2013)

[QGIS] Projektseite von QuantumGIS
URL: <http://qgis.org/>
(10.04.2013)

[GRASS] Projektseite von GRASS GIS
URL: <http://grass.osgeo.org/>
(10.04.2013)

[SAGA] Projektseite von SAGA GIS
URL: <http://www.saga-gis.org/en/index.html>
(10.04.2013)

[GvSig] Projektseite von GvSig
URL: <http://www.gvsig.org/web/>
(10.04.2013)

[uDig] Projektseite von uDig
URL: <http://udig.refractor.net/>
(10.04.2013)

[OJump] Projektseite von OpenJump
URL: <http://openjump.org/>
(10.04.2013)

[OL] Projektseite von OpenLayers
URL: <http://www.openlayers.org/>
(05.08.2013)

[GeoExt] Projektseite von GeoEXT

URL: <http://www.geoext.org/>
(12.04.2013)

[Extinfo] Wkipdia Artikel über die ExtJS Bibliothek
URL: [https://en.wikipedia.org/wiki/Ext_\(JavaScript_library\)](https://en.wikipedia.org/wiki/Ext_(JavaScript_library))
(12.04.2013)

[MapQ] Projektseite von MapQuery
URL: <http://mapquery.org/>
(12.04.2013)

[JQinfo] Wkipdia Artikel über die JQuery Bibliothek
URL: <https://en.wikipedia.org/wiki/Jquery>
(12.04.2013)

[Legato] Projektseite von Legato
URL: <https://www.legato.net/display/LEGATO/Home>
(05.08.2013)

[Leaflet] Projektseite von Leaflet
URL: <http://leafletjs.com/>
(05.08.2013)

[OScale] Projektseite von OpenScales
URL: <http://openscales.org/documentation/>
(12.04.2013)

[Mband] Projektseite von Mapbender
URL: http://mapbender.org/Mapbender_Wiki
(15.04.2013)

[GMoose] Projektseite von Geomoose
URL: <http://geomoose.org/>
(15.04.2013)

[GMajas] Projektseite von Geomajas
URL: <http://www.geomajas.org/>
(15.04.2013)

[MFish] Projektseite von Mapfish
URL: <http://mapfish.org/>
(15.04.2013)

[Cartaro] Projektseite von Cartaro
URL: <http://cartaro.org/>
(15.04.2013)

[CSWspec] Open Geospatial Consortium Inc.(23.02.2007): OpenGIS® Catalogue Services Specification
(Version: 2.0.2)

[GeoNet] Projektseite von GeoNetwork
URL: <http://geonetwork-opensource.org/>
(12.04.2013)

[pycsw] Projektseite des pycsw Catalogue Service
URL: <http://pycsw.org/>

(12.04.2013)

[SLDspec] Open Geospatial Consortium Inc.(29.06.2007): Styled Layer Descriptor profile of the Web Map ServiceImplementation Specification (Version: 1.1.0)

[SEspec] Open Geospatial Consortium Inc.(21.07.2006): Symbology Encoding Implementation Specification (Version: 1.1.0)

[SOSspec] Open Geospatial Consortium(20.04.2012): OGC® Sensor Observation Service Interface Standard (Version: 2.0)

[SensorMLspec] Open Geospatial Consortium Inc.(17.07.2010): OpenGIS® Sensor Model Language (SensorML)Implementation Specification (Version: 1.0.0)

[O&Mspec] Open Geospatial Consortium(22.03.2011): Observations and Measurements - XML Implementation (Version: 2.0)

[SPSpec] Open Geospatial Consortium(28.03.2011): OGC® Sensor Planning Service Implementation Standard (Version: 2.0)

[52NSensor] Projektseite der 52°North Sensor-bezogenen Produkte
URL: <http://52north.org/communities/sensorweb/>
(07.04.2013)

[GeoDRMspec] Open Geospatial Consortium Inc.(28.02.2006): Geospatial Digital Rights Management Reference Model(GeoDRM RM) (Version: 1.0.0)

[52NDRM] Projektseite der 52°North DRM Produkte
URL: <http://52north.org/communities/security/>
(07.04.2013)

[OSGLive] Projektseite der OSGeo Live Plattform
URL: <http://live.osgeo.org/en/>
(05.06.2013)

[O11g] Chuck Murray(2012): Oracle® Spatial Developer's Guide 11g Release 2 (11.2) E11830 ()

[GDAL/OGR] Projektseite der GDAL/OGR Bibliotheken
URL: <http://www.gdal.org/>
(20.07.2013)

[OLdoc] Anwenderdokumentation OpenLayers
URL: <http://docs.openlayers.org/>
(05.08.2013)

[OG-OLWS] OpenGIS Workshop für OpenLayers
URL: <http://workshops.opengeo.org/openlayers-intro/>
(05.08.2013)

Abbildungsverzeichnis

Abbildung 1: Schema für zentrale Datenbank mit Desktop-GIS Clients.....	20
Abbildung 2: Schema für Verwendung fremder Kartendienste.....	21
Abbildung 3: Schema für Web-GIS über individuelle Serverkomponente.....	22
Abbildung 4: Schema für das Zusammenwirken von OGC Webservices.....	23
Abbildung 5: Schema für ein Web-GIS auf Basis von OGC Webservices.....	25
Abbildung 6: Schema der Lernplattform.....	27
Abbildung 7: Oberfläche der spatialite-gui.....	30
Abbildung 8: Schaltfläche für den Shapefile Import.....	31
Abbildung 9: Ergebnis des Shapefile Imports.....	31
Abbildung 10: uDig GUI mit Beispieldatensatz.....	39
Abbildung 11: Change Style Option im uDig Layer Menü.....	39
Abbildung 12: Beispielgestaltung der Flächennutzungsdaten in uDig.....	40
Abbildung 13: Die Beispielgestaltung in XML-Repräsentation.....	41
Abbildung 14: Startseite der Geoserver Web-Oberfläche.....	42
Abbildung 15: Auflistung der vorhandenen Workspaces.....	43
Abbildung 16: Oberfläche zum Anlegen eines neuen Workspace.....	43
Abbildung 17: Interface zum Editieren des Workspace.....	44
Abbildung 18: WFS Konfigurationsoptionen (oberer Teil).....	45
Abbildung 19: WFS Konfigurationsoptionen (unterer Teil).....	45
Abbildung 20: Hinzufügen neuer SLDs.....	46
Abbildung 21: Auflistung der vorhandenen Stores.....	47
Abbildung 22: Auswahl der möglichen Arten von Stores.....	48
Abbildung 23: Konfigurationsoptionen eines PostGIS Stores (oberer Teil).....	49
Abbildung 24: Konfigurationsoptionen eines PostGIS Stores (unterer Teil).....	49
Abbildung 25: Auflistung der vorhandenen Layer.....	50
Abbildung 26: Interface zum Anlegen neuer Layer.....	51
Abbildung 27: Konfigurationsoptionen eines Vektorlayers (oberer Teil).....	52
Abbildung 28: Konfigurationsoptionen eines Vektorlayers (unterer Teil).....	52
Abbildung 29: Arten von Rasterdaten Stores	53
Abbildung 30: Anlegen eines GeoTIFF Stores.....	54
Abbildung 31: Anlegen eines WorldImage Stores.....	54
Abbildung 32: Anlegen eines Stores für Bildmosaike.....	54
Abbildung 33: Interface zum Anlegen einer Layer group.....	55
Abbildung 34: Voransicht der erstellten Layer group.....	56
Abbildung 35: Interface zur Abfrage von Voransichten.....	56

Anhang

1 CD mit Inhalt:

- Kopie der Masterarbeit im PDF Format
- Die erstellte Lernplattform als Archivdatei