ETH Zürich
Institut für Kartografie
Studiengang Geomatik und Planung
Frühlingssemester 2009

Visualisierung von Naturgefahrendaten mit QGIS MapServer und OpenLayers

Bericht zur Bachelorarbeit

Leitung
Prof. Dr. Lorenz Hurni

Betreuung
Dr. Marco Hugentobler
Ionut Iosifescu-Enescu
Melanie Kunz

Autor
Andreas Eberle
Schönaustrasse 6
8400 Winterthur
06-906-200

Winterthur, 28. Mai 2009

I Vorwort

Die vorliegende Bachelorarbeit *Visualisierung von Naturgefahren mit QGIS MapServer und OpenLayers* wurde im Rahmen des Studiengangs *Geomatik und Planung* während des sechsten Semesters am Institut für Kartografie an der Eidgenössischen Technischen Hochschule Zürich erarbeitet. Das Departement für Bau, Umwelt und Geomatik definiert in seinem Merkblatt *Bachelorarbeit im Studiengang Geomatik und Planung* die Förderung der Fähigkeit der Studierenden zu selbständiger, praxisorientierter Problemlösung oder dem wissenschaftlichen Arbeiten als Zielsetzung.

Im Rahmen dieser Aufgabe habe ich die Zeit genutzt, um einen internetbasierten Kartendienst für die Visualisierung von Geodaten zu erstellen. Neben der Erfassung von geografischen Informationen mittels geodätischer Messtechnik oder den Methoden der Fernerkundung sowie der Verwaltung und Modellierung von Datenbeständen sehe ich in der Visualisierung und Präsentation von Datensammlungen eine äusserst spannende Herausforderung. Damit schaffe ich als Geomatiker einen Mehrwert auf der Basis bereits vorhandener geografischer Daten.

Heute liegen sehr viele Informationen mit Raumbezug digital vor, sind in Datenbanken integriert oder können allenfalls schon fertig symbolisiert von einem Internetservice bezogen werden. Wenn diese Ebenen, welche teilweise mit grossem Aufwand erhoben wurden, nun durch eine geschickte Darstellung aufbereitet und in Kombination zu einander gestellt werden, können neue Aussagen abgeleitet und damit ein erweiterter Nutzen generiert werden. Dies geht auf den systemischen Gedanken von Aristoteles zurück, dass das Ganze oder eben die Kombination mehr ist als die Summe der einzelnen Elemente. Dazu kommen die heutigen Möglichkeiten der Internettechnologie, welche die Präsentation einem breiten Nutzerkreis zugänglich machen und einen hohen Grad an Interaktion erlauben.

In die Arbeit habe ich die vermittelte Theorie der Vorlesungen der letzten fünf Semester einfliessen lassen und anhand dieses Projekts angewendet. Im Zentrum stehen die Fächer Informatik und GIS für die technische Realisierung und das Datenhandling sowie die kartografischen Lehrveranstaltungen für die Umsetzung der Darstellung und die Visualisierung der zur Verfügung stehenden Ebenen. Für das projektorientierte Vorgehen kommen die Werkzeuge des Systems Engineering zur Anwendung. Thematisch wird die Arbeit begleitet durch die Vorlesung Planung und Landentwicklung.

Nun liegt der Bericht zur Bachelorarbeit vor. Er beschreibt das projektorientierte Vorgehen, geht auf die technische Umsetzung ein und erläutert die kartografische Darstellung der vorliegenden Ebenen. Der Aufbau richtet sich nach den nötigen Arbeitsschritten und dokumentiert anhand von Codebeispielen alle Elemente, welche für eine funktionierende Webkarte beachtet werden müssen. Er beschreibt somit praxisnah einen möglichen Weg zur Realisierung dieser dynamischen Internetanwendung. Aufgrund der begrenzten Zeit verzichte ich auf umfassende Beschreibungen der Möglichkeiten der verwendeten Bibliotheken und Anwendungen und verweise auf die umfassenden Dokumentationen im Internet.

Herzlich danken möchte ich meinen Betreuenden Melanie Kunz, Dr. Marco Hugentobler und Ionut Iosifescu-Enescu vom Institut für Kartografie unter der Leitung von Prof. Dr. Lorenz Hurni der ETH Zürich für die Möglichkeit, diese Arbeit bei ihnen verfassen und damit verbunden ihre Entwicklung *QGIS MapServer* kennenlernen und anwenden zu können. Die technische und fachliche Unterstützung stand mir auch in hektischen Zeiten jederzeit zur Verfügung und hat zum Gelingen dieser Anwendung beigetragen.

Mit einem lieben Dank möchte ich an dieser Stelle auch meine beiden Lektoren Kaspar Stünzi und Pia Eberle erwähnen, welche mich auf grammatikalische Verbesserungsmöglichkeiten im vorliegenden Bericht hingewiesen haben.

Andreas Eberle

II Zusammenfassung

Die vorliegende Bachelorarbeit befasst sich mit der Visualisierung der Naturgefahrenhinweiskarte des Kantons Bern mit *QGIS MapServer* und *OpenLayers*. Das nötige, projektorientierte Vorgehen für die Erstellung dieser internetbasierten Kartenanwendung sowie die Bereitstellung und Konfiguration der erforderlichen technischen Komponenten, werden in diesem Bericht anhand des praxisnahen Beispiels schrittweise besprochen. Realisiert wird eine mögliche Lösung von dynamischer Webanwendung zur kartografischen Darstellung von geografischen Daten.

Serverseitig wird die objektrelationale Datenbank *PostGreSQL* mit der Erweiterung *PostGIS* zur Speicherung von geografischen Objekten aufgesetzt und die einzelnen Ebenen der Naturgefahrenhinweiskarte sowie der Basiskarte importiert. Anschliessend erfolgt die Einbindung dieser Datenquelle in *QGIS MapServer*. Diese Serveranwendung zur Bereitstellung von Kartenebenen wurde am Institut für Kartografie der ETH Zürich entwickelt. Sie basiert auf der OpenGIS-Spezifikation *Web Map Service WMS Version 1.3.0* und benutzt die Bibliotheken des Desktopprogramms *QuantumGIS* zum Verarbeiten sowie Rendern der Geodaten. Zur Konfiguration der einzelnen Layer wird die Darstellungsbeschreibungssprache *Styled Layer Descriptor SLD* des *Open Geospatial Consortium OGC* verwendet. *QGIS MapServer* ist in der Lage *Scalable Vector Graphics SVG* zu rendern, womit SVG-Grafiken zur Symbolisierung der Ebenen in *SLD* eingebunden werden können.

Die frei verfügbare *JavaScript*-Anwendung *OpenLayers* dient als Client unter anderem zur Visualisierung von *Web Map Services WMS*. Damit können unterschiedlichste Quellen von Geodaten, wie durch den *QGIS MapServer* realisiert, eingebunden und dargestellt werden. *OpenLayers* ist im Framework *MapFish Client*, welches weitere Funktionalitäten für eine Kartenapplikation bietet, für die Zusammenstellung des Kartenbildes integriert. Die JavaScript-Bibliothek *ExtJS*, welche ebenfalls in *MapFish* enthalten ist, dient zur Realisierung des *Graphical User Interface GUI* und kann für beliebige dynamische Webanwendungen verwendet werden.

Die besondere Herausforderung der Visualisierung der Naturgefahrenhinweiskarte besteht in der flächigen Darstellung der sechs verschiedenen Prozessarten, welche sich grössenteils überlagern. Die naturräumlichen Erscheinungen werden mit typischen, kartografisch sinnvollen Symbolisierungen und nicht wie in internetbasierten *Geografischen Informationssystemen GIS* oft verwendeten einfachen Farbgebungen und komplizierten Legenden dargestellt. Dazu werden die Ausprägungen der einzelnen Gefahren systematisch analysiert und daraus die grafischen Variablen abgeleitet. Diese Informationen fliessen in die Entwicklung von SVG-Grafiken ein, mit welchen die Flächen durch horizontales und vertikales Wiederholen texturiert werden. Dazu werden die Geometrieelemente der jeweiligen SVG-Dateien extrahiert und in die Layerbeschreibung in *SLD* integriert. Dies erfolgt entweder durch manuelles Kopieren der SVG-Elemente in die *QGIS MapServer*-Konfiguration in *SLD* statisch oder wie in dieser Arbeit angewendet durch JavaScript, welches die SVG-Elemente aus der entsprechenden Grafiksammlung liest und im Parameter *SLD_BODY* mit dem WMS-Request *HTTP GET* schickt, dynamisch.

III Inhaltsverzeichnis

I Vorwort I

II Zusammenfassung III

III Inhaltsverzeichnis IV

IV Abbildungsverzeichnis VII

1. Einleitung 1

1.1 Oberziel 1

1.1.1 Thematische Zielsetzung 1

1.1.2 Technische Zielsetzung 1

1.2 Inhaltsübersicht 2

2. Projektorientiertes Vorgehen 3

2.1 Auftrag 3

2.2 Erhebung / Analyse 3

2.3 Würdigung / Diagnose 4

2.4 Lösungsentwurf 4

2.5 Bewertung 4

3. Thematische Einordnung 5

3.1 Gefahrenhinweiskarte auf Stufe der Richtplanung 5

3.2 Naturräumliche Erscheinung der einzelnen Prozesse 6

3.2.1 Rutschungen 6

3.2.2 Steinschlag 6

3.2.3 Lawinen 7

3.2.4 Murgang 7

3.2.5 Übersarungen 7

3.2.6 Dolinen 7

3.2.7 Wichtige Elemente der Basiskarte 7

4. Technische Grundlagen 8

4.1 QGIS MapServer 8

4.1.1 Installation 8

4.1.2 Konfiguration 9

a. Metainformationen (wms_metadata.xml) 9

b. Layerkonfiguration (admin.sld) 9

4.1.3 Aufruf des QGIS MapServers 10

a. GetCapabilities 11

b. GetMap 11

4.2 Styled Layer Descriptor SLD 13

4.2.1 Einbinden von Scalable Vector Graphics SVG 15

4.3 OpenLayers 17

4.3.1 Objekt Map 17

4.3.2 Objekt Layer 18

a. Base Layer 18

b. Overlay Layers 18

4.3.3 Controls 19

5. Realisierung / Umsetzung 21
5.1 Datenhaltung in PostGIS 22
5.2 Konfiguration der Ebenen für QGIS MapServer 23
5.2.1 Basiskarte (Grundkarte) 23
a. Pixelkarte (Detailreiche Basiskarte) 23
b. VECTOR 25 (Vereinfachte Basiskarte) 24
5.2.2 Gefahrenhinweiskarte 25
5.3 Aufbau der Clientanwendung mit OpenLayers 26
5.3.1 HTML/JavaScript Applikation 27
5.3.2 Visualisierung der einzelnen Naturgefahren 30
a. Styled Layer Descriptor SLD 31
b. JavaScript-Automatismus 32
5.3.3 Scalable Vector Graphics SVG – Symbolsammlung 34
a. Rutschung 36
b. Steinschlag 37
c. Lawinen 38
d. Murgang 39
e. Übersarung 40
f. Dolinen 41
5.3.4 Integration der einzelnen Naturgefahren zu einem Kartenbild 42

6. Fazit und Ausblick 43

7. Quellenangaben 45
7.1 Literatur 45
7.2 Internet 45

Anhang A: Serverseitiger Code **47**
wms_metadata.xml 47
admin.sld 47

Anhang B: Clientseitiger Code **59**
Index.html 59
Map.js 62
Controls.js 63
Layers.js 64
Toolbar.js 66
Viewport.js 70
LayerTree.js 72

Anhang C: Scalable Vector Graphics SVG – Symbolsammlung **75**
Entwicklung und Realisierung möglicher Symbole zur Darstellung der einzelnen Naturgefahren 75

Anhang D: Darstellung der einzelnen Ebenen der Naturgefahren in der Kartenapplikation **76**
Rutschung 76
Steinschlag 77
Lawinen 78
Murgang 79
Übersarung 80
Dolinen 81

Anhang E: Ordner- und Filestruktur der CD-Beilage **82**
Dokumente 82
Client 82
Server/daten 82
Server/qgis_map_server 83

IV Abbildungsverzeichnis

Abbildung 1: QGIS Desktop Plug-In PublishToWeb für den Export einer Karte 10

Abbildung 2: Schema über den Aufbau einer dynamischen Internetkarte 21

Abbildung 3: QGIS Desktop Plug-In für den Import von ESRI Shapefiles in PostGIS 22

Abbildung 4: Elemente des Graphical User Interface GUI 26

Abbildung 5: Halbtransparente Überlagerung der Naturgefahren mit einfacher Flächenfüllung 30

Abbildung 6: Vektorgrafikeditor Inkscape für die Erzeugung von Scalable Vector Graphics SVG 34

Abbildung 7: Layerbaum mit Möglichkeit zur Änderung der Überlagerung sowie der Opazität einzelner Ebenen 42

1. Einleitung

Die Bachelorarbeit *Visualisierung von Naturgefahrendaten mit QGIS MapServer und OpenLayers* soll gemäss Definition der ETH Zürich die Fähigkeit zur selbständigen, praxisorientierten Problemlösung fördern, wissenschaftliches Arbeiten enthalten und einen Einblick in die Themen der Vertiefungsrichtungen des Masterstudiums geben.

Die sechs Ebenen der Naturgefahrenhinweiskarte des Kantons Bern liegen digital vor. Diverse Werkzeuge zur Präsentation im Internet stehen auf der Basis von OpenSource frei zur Verfügung. Nun sind mit diesen technischen Möglichkeiten die thematischen Aussagen darzustellen und entsprechende Interaktionen für den Nutzer zu ermöglichen, was durchaus einem potentiellen Auftrag in der zukünftigen Arbeitspraxis entsprechen könnte. Die Herausforderung bei der Darstellung der sechs vorhandenen Naturgefahren ist die relativ starke Überlagerung ihrer Prozessräume. Das heisst, dass die Lesbarkeit bei herkömmlicher Symbolisierung mittels einer der jeweiligen Ebene zugeordneter Farbe und einer einfachen Umrandung, wie sie in den meisten bestehenden Geografischen Informationssystemen GIS im Internet angewendet wird, eher schwierig ist. Eine kartografisch sinnvolle Lösung ist durch eine detailierte Analyse der jeweiligen naturräumlichen Erscheinung, welche in die passende Symbolisierung einfliesst, zu finden. Von dieser Ausgangslage leiten sich folgend die Ziele der vorliegenden Bachelorarbeit ab.

1.1 Oberziel

Hauptziel ist die Erstellung einer praxistauglichen Internetanwendung zur Visualisierung der Naturgefahrenhinweiskarte des Kantons Bern für potentielle Nutzer. Serverseitig basiert sie auf *QGIS MapServer* und clientseitig auf der Benutzerschnittstelle *OpenLayers*.

1.1.1 Thematische Zielsetzung

Die Darstellung der sechs vorliegenden Ebenen der Gefahrenhinweiskarte ist themagerecht für den *QGIS MapServer* aufzubereiten. Das heisst, für den zu visualisierenden Sachverhalt und die Massstabsvariabilität sind geeignete Symbolisierungen zu finden und die Darstellung der einzelnen Layer ist aufeinander abzustimmen. Geeignete Kombinationen von Ebenen sind zu untersuchen und mit einer Hintergrund-/Basiskarte zu hinterlegen.

1.1.2 Technische Zielsetzung

Die serverseitige Anwendung *QGIS MapServer* in Kombination mit der Datenhaltung in *PostGIS* soll erstellt, die Funktionalität kennengelernt und anschliessend sinnvoll für die Aufbereitung der geografischen Daten genutzt werden. Ebenfalls möchte ich mich in die clientseitige Anwendung *OpenLayers* einarbeiten und die Möglichkeiten der Benutzerinteraktion kennen lernen. Danach ist ein für die gestellte Aufgabe geeigneter Einsatz auszuarbeiten.

1.2 Inhaltsübersicht

Das angewendete und erprobte **projektorientierte Vorgehen**, welches auf dem Systems Engineering basiert und auch für zukünftige Problemstellungen als Lösungsstrategie angewendet werden kann, wird im ersten Teil des Berichts ausführlich erläutert. Damit wird die erlernte Theorie zur systematischen Erledigung einer Aufgabe anhand einer kartografischen Arbeit in die Praxis umgesetzt und anhand dieses Beispiels ganzheitlich durchgespielt.

Für die Realisierung der Visualisierung und einer passenden Präsentation der vorliegenden Daten sind die **thematische Einordnung** und die vertiefte Auseinandersetzung mit der Problemstellung der Naturgefahren zentral. Dazu wird der Bezug der Naturgefahrenhinweiskarte zur Raumplanung eingehend diskutiert und die naturräumlichen Erscheinungsformen der darzustellenden Informationsebenen detailliert aufgezeigt. Aus diesen Überlegungen wird im Kapitel 5 über die Realisierung und Umsetzung die Symbolisierung abgeleitet.

Neben der thematischen Komponente müssen die **technischen Grundlagen** und Möglichkeiten der vorgegebenen Werkzeuge vor der Realisierung von Lösungsvarianten ebenfalls bekannt sein. Die Funktionalität der serverseitigen Anwendung *QGIS MapServers* sowie der clientseitigen JavaScript-Bibliothek *OpenLayers* wird beschrieben.

Dann folgen die **Realisierung und Umsetzung**, also das Zusammenführen von Thema und Technik zu einer ganzheitlichen Lösung nach dem systemischen Gedanken. Im **Fazit und Ausblick** werden die optimierten Varianten zur Darstellung der einzelnen Ebenen der Naturgefahrenhinweiskarte diskutiert, die Zielerreichung analysiert und sinnvoll weiterzuverfolgende Aspekte aufgezeigt.

Der vorliegende Bericht über die Realisierung der Internetanwendung zur Visualisierung der Naturgefahrenhinweiskarte des Kantons Bern beinhaltet somit folgende Hauptteile:

- Projektorientiertes Vorgehen
- Thematische Einordnung
- Technische Grundlagen
- Realisierung und Umsetzung
- Fazit und Ausblick

2. Projektorientiertes Vorgehen

Die Visualisierung der Naturgefahrendaten mit *QGIS MapServer* und *OpenLayers* und damit die Erstellung der Internetapplikation erfolgt projektorientiert nach dem Planungszyklus, wie ihn Dr. Götz Schmidt in seinem Buch *Organisatorische Grundbegriffe* (2005) beschreibt und in weiten Teilen identisch ist mit dem Problemlösungszyklus des Betriebswirtschaftlichen Institutes der ETH Zürich. Durch das systemische Vorgehen ist sichergestellt, dass alle relevanten Aspekte berücksichtigt und bearbeitet werden. Die einzelnen Arbeiten gliedern sich in folgenden Phasen:

- Auftrag: Klärung und Abgrenzung der Problemstellung inkl. Terminplanung
- Erhebung: Beschaffung thematischer und technischer Informationen
- Analyse: Aufbereitung und Zusammenstellung der Möglichkeiten
- Würdigung: Mögliche technische Werkzeuge zur thematischen Umsetzung
- Lösungsentwurf: Variantenbildung
- Bewertung der optimierten Lösung

2.1 Auftrag

Der Auftrag ist durch das *Konzept zur Bachelorarbeit FS 2009* vom Februar 2009 konkretisiert. Die Ausschreibung des Instituts für Kartografie soll hier nochmals wiedergegeben werden:

Die Überschwemmungen der letzten Jahre zeigten die Aktualität von Naturgefahren sehr deutlich. Für den Schutz von Menschen und Gebäuden ist die Herstellung von Gefahren- und Gefahrenhinweiskarten sehr wichtig. Am IKA sind Daten des Kantons Bern von verschiedenen Naturgefahren vorhanden. Diese sollen überlagert und mit OGC-konformen internetbasierten Kartendiensten und entsprechenden Erweiterungen, welche am IKA entwickelt wurden, visualisiert werden. Die Daten können in räumlichen Datenbanken gehalten werden und mit der Software 'QGIS MapServer' als Bild an den Client gesendet werden. Für den Client kann eine OpenLayers basierte Entwicklung erweitert werden.

2.2 Erhebung / Analyse

Für die gestellte Zielsetzung, die Erstellung einer praxistauglichen Internetanwendung zur Visualisierung der Naturgefahrenhinweiskarte des Kantons Bern für potentielle Nutzer, müssen vorgängig Informationen beschafft werden. Zum Einen ist die Anwendung thematisch einzuordnen und die naturräumlichen Erscheinungen sind zu analysieren (Kapitel *3. Thematische Einordnung*). Als Zweites müssen die technischen Möglichkeiten der gewählten Tools möglichst umfassend bekannt sein, um eine gute Auswahl treffen zu können (Kapitel *4. Technische Grundlagen*).

2.3 Würdigung / Diagnose

Die Funktionalitäten und deren Eignung zur Darstellung von vorgegebenen Ebenen durch *QGIS MapServer* werden durch Tests ermittelt. Stärken und Schwächen der Anwendung sind zu erkennen und werden in den Kapiteln 3 und 4 aufgenommen. Diese führen zu Chancen und Risiken in der folgenden Erstellung der Applikation.

Ebenfalls wird abgeschätzt, wie viel und welche Interaktionen (Zoomstufe, usw.) dem Nutzer clientseitig durch *OpenLayers* zur Verfügung gestellt werden sollen, welche Werkzeuge von *OpenLayers* für die vorliegende Aufgabe sinnvoll sind und in welchen Kombinationen die einzelnen Ebenen eingebunden werden.

2.4 Lösungsentwurf

Mit den zuvor gewonnen Erkenntnissen wird die internetbasierte Karte zusammengestellt und in Kapitel *5. Realisierung und Umsetzung* beschrieben. Alle Ebenen sind themengerecht symbolisiert und die clientseitigen Werkzeuge in *OpenLayers* konfiguriert. Dabei werden verschiedene Varianten der Darstellungen und Kombinationen von Ebenen erstellt und miteinander verglichen, um so eine optimierte Lösung entwickeln zu können.

2.5 Bewertung

Die gesamte Arbeit wird in Kapitel *6. Fazit und Ausblick* gewürdigt, bewertet und auf die Erfüllung der Anforderungen und Ziele überprüft.

3. Thematische Einordnung

Um die gestellte Aufgabe optimal umsetzten und damit die vorhandenen Informationen den naturräumlichen Erscheinungen entsprechend darstellen zu können, ist eine thematische Vertiefung nötig. Als Erstes ist der Zweck der Naturgefahrenkarte im Rahmen der Raumplanung kurz zu erläutern. Anschliessend sind die Erscheinungsformen der einzelnen Prozesse zu beschreiben. Dabei beziehe ich mich auf den Erläuterungsbericht zur Gefahrenhinweiskarte des Kantons Bern und wiedergebe die wichtigsten Punkte in Bezug auf die kartografische Präsentation mittels Internetapplikation.

3.1 Gefahrenhinweiskarte auf Stufe der Richtplanung

Die Bundesgesetze und -verordnungen über den Wald und den Wasserbau verpflichten die Kantone, Gefahrenkarten zu erstellen und diese in der Richt- und Nutzungsplanung sowie bei den übrigen raumwirksamen Tätigkeiten zu berücksichtigen. Dass diese Bestimmung so in die Gesetzgebung aufgenommen worden ist, ist eine Lehre, die man insbesondere aus den schlimmen Hochwasserereignissen 1987 gezogen hat.[1]

Die **Gefahrenhinweiskarte** bietet eine Übersicht über mögliche Schadenereignisse und deren Potentiale in einem grösseren Raum sowie die daraus entstehenden Interessenskonflikte in der Nutzung des Raums. Damit kann sie als Art Vorstudie für die Gefahrenerkennung und Quantifizierung herangezogen werden und dient einem umfassenden Risikomanagement. Sie richtet sich an die Behörden und ist damit auf der Stufe der Richtplanung anzusiedeln.

Die **Gefahrenkarte**, welche die Gefahrenhinweiskarte konkretisiert, verortet verschiedene Gefahrenarten in einem Gebiet eindeutig, das heisst auf der Grundlage eines Parzellarkatasters. Detailliert werden die Intensität in Form von Gefahrenstufen und die Wahrscheinlichkeit eines Eintritts einer Gefährdung sowie die genaue räumliche Ausdehnung dargestellt. Die Ergebnisse fliessen in die Nutzungsplanung ein, beeinflussen Bauvorhaben und Bauzonen und sind somit allgemein verbindlich.

Die Gefahrenhinweiskarte des Kantons Bern wurde von 1994 bis 1998 erhoben und umfasst die Prozessräume für Dolinen, Lawinen, Murgänge, Rutschungen, Stein- und Blockschläge, welche vektoriell im ESRI Shape-Format vorliegen. Die computergestützte Erfassung basierte grösstenteils auf dem digitalen Höhenmodell (DHM25) und der Pixelkarte (PK25) sowie den Daten der Volkszählung des Bundesamtes für Statistik. Die Ebenen Rutschungen und Absenkungen wurden aufgrund geologischer Karten und Luftbildern klassisch aufbereitet. Gedruckt wurden zwei Karten mit den Kombinationen der Ebenen Erd- und Felsbewegungen (Rutschungen, Absenkungen, Stein- und Blockschlag) sowie Lawinen- und Murganggefahren. Weitergehende Informationen zum Erhebungsprozess sind dem Erläuterungsbericht zur Gefahrenhinweiskarte zu entnehmen.

[1] Bundesamt für Raumentwicklung ARE (2009): Gefahrenkarte.
http://www.are.admin.ch/themen/raumplanung/00244/00432/00433/index.html?lang=de (Zugriff 11. März 2009).

3.2 Naturräumliche Erscheinung der einzelnen Prozesse

Die Naturgefahren entstehen durch die oberflächliche Bewegung von Wasser-, Schnee-, Eis-, Erd- und Felsmassen. Für ihre Darstellung, welche sich an den natürlichen Erscheinungen der einzelnen Prozesse orientieren sollte, müssen diese vorgängig dokumentiert werden. Bewegungsmechanismen, beteiligte Materialien sowie die Kubatur spielen dabei eine Rolle. Weiter müssen die unterschiedlichen Pauschalgefälle und die damit verbundenen Reichweiten sowie der Untergrund und die Zusammensetzung des stürzenden oder fliessenden Materials berücksichtigt werden.

3.2.1 Rutschungen

Hangteile mit Fels- und Lockergestein an mässig geneigten bis steilen Böschungen kommen durch Scherbrüche an der Untergrenze der Massen in Bewegung. Die vielfältigen Erscheinungsformen können sowohl allmählich als auch plötzlich auftreten. Dabei ist Wasser in Form von Porenwasserdruck, Sickerströmung oder Quelldruck bei Tonmineralien ein entscheidender Faktor. Weiter sind die gesamte Kubatur, die Verschiebungsraten sowie die differentiellen Bewegungen an Grenz- und Gleitflächen zu berücksichtigen.

Gefährdet sind Objekte, welche sich auf der gleitenden Masse befinden oder in deren Bewegungsbahn liegen. Ebenfalls besteht die Gefahr, dass sich Rutschungen zu Hangmuren ausbilden oder Murgänge auslösen können.

Klassifiziert werden die Rutschungen nach der Tiefe der Gleitfläche in oberflächlich/flachgründig (0 – 2 m), mitteltief/mittelgründig (2 - 10 m) und tief/tiefgründig (> 10 m) sowie nach Aktivität, das heisst, durchschnittliche Rutschgeschwindigkeit in substabil/sehr langsam (0 – 2 cm/Jahr), wenig aktiv/langsam (2 – 10 cm/Jahr) und aktiv/schnelle Phasen (> 10 cm/Jahr)

3.2.2 Steinschlag

Der Steinschlag umfasst eher isolierte Sturzbewegungen in Form von Fallen, Springen oder Rollen von losen Einzelkomponenten bis zu Feldbrocken von einigen Kubikmetern an Hängen mit einem Pauschalgefälle von 30 – 35°. Dabei können Geschwindigkeiten von bis 30 m/s auftreten. Der Wald mit seiner Schutzfunktion trägt zur Reduktion der kinetischen Energie bei.

Die Abgrenzung des Steinschlags zu Fels- und Bergsturz erfolgt nach den Volumina und den Bewegungsmechanismen. So fallen beim Felssturz Gesteinspakete von 100 bis mehreren 100‘000 Kubikmetern mit einer Geschwindigkeit von 10 – 40 m/s. Der Bergsturz befördert Massen von mehreren Millionen Kubikmetern mit starken Wechselwirkungen zwischen den Komponenten und Geschwindigkeiten über 40 m/s. Dies kann zum Aufstauen von Bächen und Flüssen führen.

3.2.3 Lawinen

Lawinen entstehen an Hängen mit einem Pauschalgefälle von 20 – 25°. Als Fliesslawinen werden plötzliche und schnelle Talwärtsbewegungen von Schnee- und Eismassen in Form von Gleiten, Fliessen und Rollen bezeichnet. Unterschieden werden nasse Fliesslawinen mit Geschwindigkeiten zwischen 10 – 30 m/s und trockene Lawinen mit Geschwindigkeiten von 20 – 40 m/s.

Staublawinen bilden aufgewirbelte Schnee- oder Eiswolken mit Bewegungen von 30 – 70 m/s.

3.2.4 Murgang

Abhänge mit einem Gefälle zwischen 10 – 20° können in Kombination mit Hochwasser zu Murgängen führen. Dabei sind verschiedene Prozesse zu beobachten.

Wildbäche mit einem grossen Gefälle und stark wechselnden Abflussmengen führen zu intensiven Feststoffverlagerungen. Diese Materialtransporte stehen oft in Kombination mit anderen Hangprozessen, welche das Gerinne mit neuem Geschiebematerial beliefern. Bei starken Niederschlägen können schnell fliessende Gemische von Wasser und Feststoffen mit hohen Feststoffanteilen von 30 – 60% entstehen. Ablagerungen dieser Geschiebe durch die fliessenden Wassermassen werden als Übermurungen bezeichnet.

Hangmurgänge sind unabhängig von Gerinnen und entstehen an steilen Hängen mit geringdurchlässigen, erosionsanfälligen Quartärbildungen als oberflächlich schnelles Gemisch aus Lockergestein mit viel Wasser sowie einer grossen Prozessgeschwindigkeit von 1 - 10 m/s.

3.2.5 Übersarungen

Übersarungen sind Ablagerungen von Feststoffen ausserhalb des Gerinnes anlässlich von Überschwemmungen. Häufig treten sie im Bereich der Schwemmkegel von Wildbächen auf und stehen somit auch in Zusammenhang mit starken Niederschlägen und darauf folgenden Murgängen.

3.2.6 Dolinen

Gestein mit einem hohen Anteil an wasserlöslichen Stoffen wie Gips oder Malmkalk führen durch zirkulierendes Wasser zu einem verkarsteten Untergrund. Entstandene Hohlräume können dann einstürzen und stellen vor allem eine Gefahr für Sachwerte dar.

3.2.7 Wichtige Elemente der Basiskarte

Die Naturprozesse bedrohen Siedlungen und Verkehrsinfrastrukturen und damit den Lebensraum des Menschen. Diese Elemente müssen somit zwingend durch die Basiskarte visualisiert und anschliessend durch die Darstellung der Prozessräume überlagert werden. Ebenfalls sind Bodenbedeckung und Bewuchs relevant für die Ausbreitung der Naturgefahren. Waldflächen mit ihrer Schutzfunktion sind speziell auszuscheiden. Ebenfalls sind das Gewässernetz sowie die Form des Geländes entscheidend betreffend der Ausbreitung.

4. Technische Grundlagen

Internetbasierte Kartendienste stellen räumliche Informationen mittels Internettechnologie dar und erlauben, diese zu verteilen und zu bearbeiten. Der Grad der Interaktion und der Funktionalität ist je nach Anwendung und Nutzersicht recht unterschiedlich.

Grundlegend basiert die Anwendung auf zwei Komponenten. Datenquellen werden serverseitig aufbereitet und anschliessend an Clients, welche als Benutzerschnittstellen dienen, gesendet. Der Austausch zwischen Server und Client ist dabei weitgehend standardisiert und basiert auf den Spezifikationen des *Open Spatial Consortiums OGS*.

Eine dynamische Karte eignet sich für grössere Datenmengen, für Daten welche oft aktualisiert werden oder auf einer weiten räumlichen Ausdehnung beruhen. Neben den Standardfunktionalitäten wie *Pan* und *Zoom* stehen auch Möglichkeiten zur räumlichen Abfrage zur Verfügung. Die Karte wird bei jedem Aufruf anhand der Einstellungen des Benutzers aus den originalen Geodaten auf dem Server neu erstellt.[2]

4.1 QGIS MapServer

QGIS MapServer ist eine am Institut für Kartografie der ETH Zürich entwickelte OpenSource-Anwendung und basiert auf den OpenGIS-Spezifikationen *Web Map Service WMS Version 1.3*. Diese definieren einheitliche Zugriffe auf digitale Karten über das Internet. Sowohl Vektor- wie auch Rasterdaten verschiedener Formate und Quellen können zur Visualisierung integriert werden. Der Server bereitet diese gemäss einem entsprechenden Aufruf auf und sendet den generierten Kartenausschnitt über das Internet an den Client zurück.[3]

4.1.1 Installation

QGIS MapServer basiert auf *Quantum GIS (QGIS)* und nutzt dessen Bibliotheken zum Verarbeiten der GIS-Daten und zum Rendern der Karten. Der Programmcode ist wie die Desktopapplikation in C++ geschrieben und steht unter der *General Public License GPL* zur freien Nutzung bereit. Für die gängigen Betriebssysteme *Linux*, *MacOSX* und *Windows* werden kompilierte *FastCGI/CGI*-Module angeboten, welche zur Installation ins entsprechende Webserververzeichnis kopiert werden müssen.[4]

Beim Aufruf des *Common Gateway Interface CGI* wird durch den Webserver jeweils ein Verarbeitungsprozess zur Generierung der Karte gestartet. *FastCGI* erlaubt dem Server mehrere Request auch von unterschiedlichen Clients mittels eines einzigen Programmaufrufs zu beantworten.

[2] Sutter, J. (2008): Internet & GIS - Erstellen eines dynamischen WebGIS. Vorlesung GIS II, Frühjahrssemester 2008, ETH Zürich.
[3] Hugentobler, M., Iosifescu-Enescu, I. (2009): What is QGIS mapserver ? http://karlinapp.ethz.ch/qgis_wms/about/index.html (Zugriff 16. April 2009)
[4] Hugentobler, M., Iosifescu-Enescu, I. (2009): What is QGIS mapserver ? http://karlinapp.ethz.ch/qgis_wms/about/index.html (Zugriff 16. April 2009)

4.1.2 Konfiguration

a. Metainformationen (wms_metadata.xml)

Der angebotene Service des *QGIS MapServers* wird durch die Metadaten im Element *Service* der Datei *wms_metadata.xml* beschrieben. Diese beinhaltet den Titel und eine Beschreibung des Angebots sowie Angaben zur publizierenden Organisation und zur Kontaktperson. Diese Informationen kann ein Nutzer mit dem Aufruf *GetCapabilities* abrufen.

```
<!-- Service Metadata -->
<Service>
    <Name>VEC25_WMS</Name>
    <!-- Human-readable title for pick lists -->
    <Title>Vektor 25, WMS</Title>
    <!-- Narrative description providing additional information -->
    <Abstract>Bachelorarbeit FS 2009, Andreas Eberle</Abstract>
    <!-- Top-level web address of service or service provider <DCPType>. -->
    <OnlineResource type="simple" href="http://localhost" />
    <!-- Contact information -->
    <ContactInformation>
        <ContactPersonPrimary>
            <ContactPerson>Andreas Eberle</ContactPerson>
        </ContactPersonPrimary>
        <ContactVoiceTelephone></ContactVoiceTelephone>
        <ContactElectronicMailAddress>aeberle@student.ethz.ch</ContactElectronicMailAddress>
    </ContactInformation>
</Service>
```

b. Layerkonfiguration (admin.sld)

QGIS MapServer benutzt das XML Schema *Styled Layer Descriptor SLD* des *Open Geospatial Consortium OGC* zur Beschreibung der einzelnen, zu publizierenden Vektor- und Rasterdatensätze. Mit dieser Darstellungsbeschreibungssprache werden die einzelnen Kartenebenen zur Visualisierung eingebunden, die entsprechenden Symbolisierungen festgelegt und damit der Server konfiguriert.

Statisch werden die darzustellenden Layer direkt mittels *SLD* in der Datei *admin.sld* auf dem Server beschrieben. Dazu kann eine Karte mit *Quantum GIS* entworfen werden. Die einzelnen Vektor- oder Rasterebenen aus einer vorhandenen Datenbank oder einer Fileablage werden angegeben. Ebenfalls können bestehende WMS Layer hinzugefügt werden. Danach erfolgt die Symbolisierung der einzelnen Objekte. Ein Einzelsymbol kann für alle Elemente eines Layers gewählt werden oder es können verschiedene Symbole nach eindeutigen Werten einer Attributspalte bestimmt werden.

Gewählt werden Umrandungsfarbe und –stärke sowie die Füllfarbe für die darzustellende Fläche. Die Informationen zum Stil der Umrandung und der Füllung bei Flächenobjekten oder Symbolen von Punkten gehen beim Export des QGIS-Projekts in die SLD-Datei verloren. Die Flächen werden ausgefüllt und Punktobjekte mit einem einfachen Kreis dargestellt. Dazu können, wie im Abschnitt *Einbinden von Scalable Vector Graphics SVG* beschrieben, SVG-Grafiken direkt in *admin.sld* eingebunden werden.

Das QGIS-Projekt wird mit dem Plug-In *PublishToWeb* exportiert.

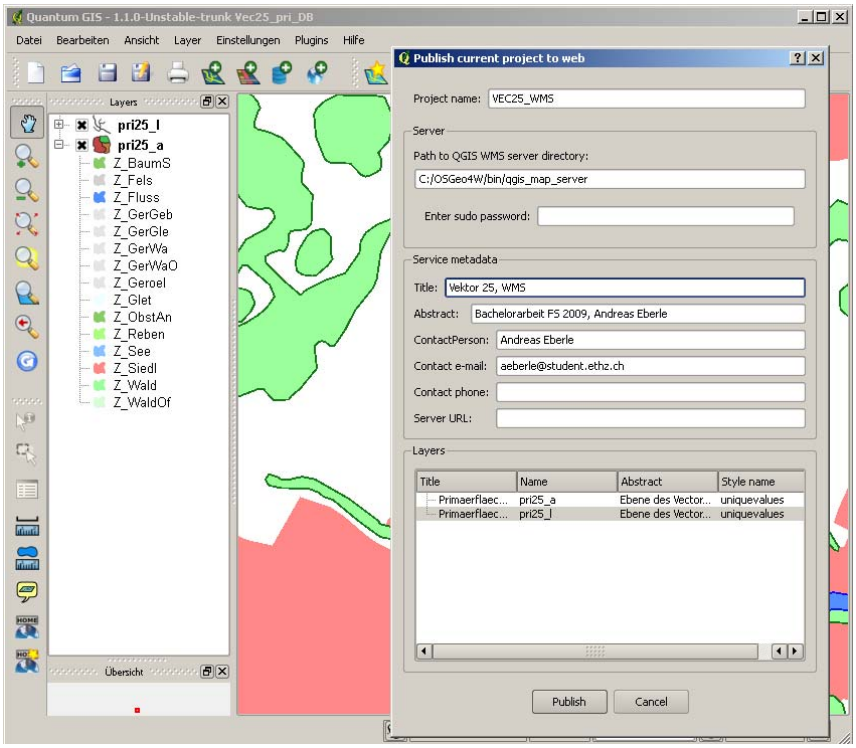


Abbildung 1: QGIS Desktop Plug-In PublishToWeb für den Export einer Karte

Die durch das Plug-In erzeugte und nachfolgend noch optimierte Datei *admin.sld* ist im *Anhang A: Serverseiteiger Code* abgedruckt. Wie schon erwähnt, können nun in dieser Datei manuelle Anpassungen angebracht werden. Deshalb wird im Abschnitt *Styled Layer Descriptor SLD* die entsprechende Struktur detailliert besprochen.

Für ein benutzerspezifisches Aussehen eines jeweiligen Layers kann *SLD* auch mit dem *HTTP GET* oder *HTTP POST Request* an den *QGIS MapServer* mit gesendet werden. Dieser entsprechende Aufruf wird im kommenden Abschnitt behandelt.

4.1.3 Aufruf des QGIS MapServers

Der *QGIS MapServer* wird gemäss der OpenGIS-Spezifikationen *Web Map Service WMS Version 1.3.0* mittels einem *HTTP GET* oder *HTTP POST Requests* angesprochen. Der *Uniform Resource Locator URL* können Parameter, bestehend aus Schlüsseln mit dazugehörigen Werten, gemäss der WMS-Spezifikation mitgeschickt werden. Die Parameter sind von der Basis-URL durch ein *Fragezeichen ?* getrennt und untereinander mit einem *Ampersand &* verbunden.

```
http://karlinapp.ethz.ch/fcgi-bin/qgis_map_server/webgisbern/qgis_map_serv.fcgi?
                                SERVICE=WMS&
                                Version=1.3.0&
                                REQUEST=GetCapabilities
```

Der *Web Map Service (WMS)* sowie die verwendete *Version (1.3.0)* müssen angegeben werden. *QGIS MapServer* unterstützt die Operationen *GetCapabilities* und *GetMap*, welche über den Parameter *REQUEST* aufgerufen werden.

a. GetCapabilities

Die Operation *GetCapabilities* dient dazu, die Metadaten, also Informationen über die Fähigkeiten und das Datenangebot eines Service, abzurufen.[5]

```
http://karlinapp.ethz.ch/fcgi-bin/qgis_map_server/webgisbern/qgis_map_serv.fcgi?
                        SERVICE=WMS&
                        Version=1.3.0&
                        REQUEST=GetCapabilities
```

Als Antwort, auch Response genannt, liefert der *Web Map Service WMS* ein XML-Dokument mit folgender Struktur:

- <Service> enthält die Metaangaben, wie im Abschnitt Konfiguration beschrieben, zum befragten Server
- <Capabilities> beschreibt die Fähigkeiten des Servers. Die Adresse für den Aufruf sowie die Ausgabemöglichkeiten werden angegeben.
- <Exception> spezifiziert Fehlermeldungen
- <Layer> listet die einzelnen Ebenen mit einer genauen Beschreibung auf, wie beispielsweise den für einen Aufruf benötigten Namen.

b. GetMap

Der Request *GetMap* liefert als Antwort einen durch die weiteren Parameter spezifizierten Kartenausschnitt. Dabei zentral sind die Pflichtangaben *LAYERS* gefolgt von der Auflistung der gewünschten Ebenen sowie *STYLES* mit der Liste der Darstellungsarten. Wie schon im vorhergehenden Abschnitt erwähnt, können einzelne Layer mit unterschiedlichen Styles abgefragt werden.

Weiter definiert *SRS* das geforderte geodätische Referenzsystem nach dem Code der *European Petroleum Survey Group EPSG* und *BBOX* beschreibt mittels minimalen und maximalen Koordinatenangaben den zu exportierenden Kartenausschnitt. Die Parameter *WIDTH* und *HEIGHT* stehen für die Grösse der Karte in Pixeln und *FORMAT* für das Dateiformat. Für GIF- und PNG-Bilder kann *TRANSPARENT* aktiviert werden.[6]

Folgendes Beispiel dient zum Aufruf der im Abschnitt Konfiguration beschriebenen Ebenen.

```
http://karlinapp.ethz.ch/fcgi-bin/qgis_map_server/webgisbern/qgis_map_serv.fcgi?
                        SERVICE=WMS&
                        REQUEST=GetMap&
                        Version=1.3.0&
                        LAYERS=pri25_a,str25_l&
                        STYLES=uniquevalues,singlesymbol&
                        SRS=...
```

[5] Schilcher, M., Donaubauer, A. (2008): Interoperabilität (OGC). Gastvorlesung im Rahmen der Lehrveranstaltung GIS II, Frühlingssemester 2008, ETH Zürich.

[6] Schilcher, M., Donaubauer, A. (2008): Interoperabilität (OGC). Gastvorlesung im Rahmen der Lehrveranstaltung GIS II, Frühlingssemester 2008, ETH Zürich.

Die Layer werden dabei in der Folge ihres Aufrufs übereinander gerendert. Im Beispiel überdeckt der Layer *str25_l* mit dem Style *singlesymbol* den Layer *pri25_a* mit dem Style *uniquevalues*. Dabei kann derselbe Layer auch mehrmals mit unterschiedlichen Styles aufgerufen werden, um beispielsweise eine Strasse mit einer breiten schwarzen und anschliessend einer schmalen hellen Linie als Parallelen darzustellen.

Anstatt der Verwendung der Parameter *LAYERS* und *STYLES* kann über den Parameter *SLD_BODY* ein SLD-Dokument als Stringfolge mit den Elementen *NamedLayer* und *NamedStyle* dem *HTTP GET Request* angehängt werden.

```
<StyledLayerDescriptor>
  <NamedLayer>
    <Name>pri25_a</Name>
    <NamedStyle>
      <Name>uniquevalues</Name>
    </NamedStyle>
  </NamedLayer>
  <NamedLayer>
    <Name>str25_l</Name>
    <NamedStyle>
      <Name>singlesymbol</Name>
    </NamedStyle>
  </NamedLayer>
</StyledLayerDescriptor>
```

An den Server wird dann folgender *HTTP GET Request* gesendet:

```
http://karlinapp.ethz.ch/fcgi-bin/qgis_map_server/webgisbern/qgis_map_serv.fcgi?
                          SERVICE=WMS&
                          Version=1.3.0&
                          REQUEST=GetMap&
                          SLD_BODY=<StyledLayerDescriptor><NamedLayer><Name>pri25_a</Name><NamedStyle><Name>
                          uniquevalues</Name></NamedStyle></NamedLayer><NamedLayer><Name>str25_l</Name><Name
                          dStyle><Name>singlesymbol</Name></NamedStyle></NamedLayer></StyledLayerDescriptor>
                          &
                          ...
```

Nun können nicht nur vordefinierte Styles durch *NamedStyle* verwendet, sondern auch benutzerspezifische Darstellungen, wie im folgenden Abschnitt *Styled Layer Descriptor SLD* beschrieben, innerhalb des Elements *UserStyle* mit dem Parameter *SLD_BODY* mit gesendet werden. Ebenfalls können mit dem Element *UserLayer* neue Ebenen kreiert werden.

Da die Länge eines *HTTP GET Requests* begrenzt ist und der Wert des Parameters *SLD_BODY* unter Umständen gross wird, kann alternativ die *GetMap* Methode in XML kodiert per *HTTP POST Request* an den *Web Map Service WMS* gesendet werden. In dieser Arbeit wird auf diese Anwendung verzichtet und auf einen „schlanken“ SLD_BODY-Wert geachtet.

4.2 Styled Layer Descriptor SLD

Die Struktur der Darstellungsbeschreibungssprache *Styled Layer Descriptor SLD* des *Open Geospatial Consortium OGC* zur Darstellung von Vektor- und Rasterdaten basiert auf einer Hierarchie von Elementen. Der äusserste, alles umschliessende Bereich ist *StyledLayerDescriptor*.

```
<StyledLayerDescriptor xmlns="http://www.opengis.net/sld">
```

Anschliessend folgt die Auflistung aller Layer, welche jeweils durch das Element *UserLayer* begrenzt werden.

```
<UserLayer xmlns="http://www.opengis.net/sld">
```

Dem einzelnen Layer wird unter *Name* eine eindeutige Bezeichnung zugewiesen. Diese ist nötig für den Aufruf mittels Parameter *LAYERS* eines *HTTP GET* oder *HTTP POST Request*. Dann folgen die freie Bezeichnung *Titel* sowie die Möglichkeit zur Angabe einer erweiterten Beschreibung:

```
<Name xmlns="http://www.opengis.net/sld">pri25_a</Name>
<Title xmlns="http://www.opengis.net/sld">Primaerflaechen</Title>
<Abstract xmlns="http://www.opengis.net/sld">Ebene des Vector 25</Abstract>
```

Folgende Auflistung zeigt Anbindungen unterschiedlicher Datenquellen und verschiedener Formate durch die Attributen *providerTyp*, *uri* und *epsg*.

- Vektordatensatz aus *PostGIS*

```
<HostedVDS xmlns="http://www.opengis.net/sld" providerType="postgres" uri="dbname='nathazbern'
                        host=karlinapp.ethz.ch port=5000 user='andreas' password='eberle2:!'
                        table=&quot;pri25_a&quot; (the_geom) sql=" epsg="21781" />
```

- Vektordatensatz *ESRI Shapefile*

```
<HostedVDS xmlns="http://www.opengis.net/sld" providerType="ogr"
                        uri="/home/andreas/Daten/vec25/pri25_a.shp" epsg="21781"/>
```

- Rasterdatensatz *GeoTIFF*

```
<HostedRDS xmlns="http://www.opengis.net/sld" uri="/home/andreas/Daten/dhm25/dhm.tif" epsg="21781"/>
```

- Virtual Raster

```
<HostedRDS xmlns="http://www.opengis.net/sld" uri="/home/andreas/Daten/dhm25/dhm.vrt" epsg="21781"/>
```

Die benutzerspezifische Symbolisierung wird unter *UserStyle* definiert.

```
<UserStyle xmlns="http://www.opengis.net/sld">
```

Da pro Layer mehrere Styles, beispielsweise für unterschiedliche Massstabsbereiche, definiert werden können, sind die einzelnen Blöcke durch das Element *Name* eindeutig gekennzeichnet. Diese Identifizierung muss bei gewünschter Verwendung mit dem *HTTP GET* oder *HTTP POST Request* mitgeschickt werden.

```
<Name xmlns="http://www.opengis.net/sld">uniquevalues</Name>
```

Anschliessend folgen die Symbolisierungen innerhalb von *FeatureTypeStyle*.

```
<FeatureTypeStyle xmlns="http://www.opengis.net/sld">
```

Dieses kann, wenn die Elemente eines Layers unterschiedlich nach einem Attributwert dargestellt werden soll, mehrere Regeln enthalten.

```
<Rule xmlns="http://www.opengis.net/sld">
```

Dazu wird das Element *Filter* eingeschoben, welches den Spaltenname im Element *PropertyName* sowie den entsprechenden Attributwert unter *Literal* zur Visualisierung enthält.

```
<Filter xmlns="http://www.opengis.net/ogc">
  <PropertyIsEqualTo xmlns="http://www.opengis.net/ogc">
    <PropertyName xmlns="http://www.opengis.net/ogc">objectval</PropertyName>
    <Literal xmlns="http://www.opengis.net/ogc">Z_BaumS</Literal>
  </PropertyIsEqualTo>
</Filter>
```

Folgend kann die eigentliche Symbolisierung angegeben werden. Die Liste zeigt einige Beispiele zu den unterschiedlichen Geometrietypen Punkt, Linie und Fläche mit einfachen Symbolen, das heisst die Bestimmung von Umrandungsfarbe *stroke* und Stärke *stroke-width* sowie der Flächenfarbe *fill* und einfachen Grafiken. Komplexe Grafiken und Strukturen zur Füllung von Flächen können als SVG-Grafiken eingefügt werden. Dies wird im nächsten Unterabschnitt *Einbinden von Scalable Vector Graphics SVG* behandelt.

- Polygon

```
<PolygonSymbolizer xmlns="http://www.opengis.net/sld">
  <Stroke xmlns="http://www.opengis.net/sld">
    <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke" >#000000</CssParameter>
    <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke-width" >0</CssParameter>
  </Stroke>
  <Fill xmlns="http://www.opengis.net/sld">
    <CssParameter xmlns="http://www.opengis.net/sld" :name="fill" >#89cb65</CssParameter>
  </Fill>
</PolygonSymbolizer>
```

- Linie

```
<LineSymbolizer xmlns="http://www.opengis.net/sld">
  <Stroke xmlns="http://www.opengis.net/sld">
    <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke" >#120fff</CssParameter>
    <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke-width" >1</CssParameter>
  </Stroke>
</LineSymbolizer>
```

- Punkt

```
<PointSymbolizer xmlns="http://www.opengis.net/sld">
  <Graphic xmlns="http://www.opengis.net/sld">
    <Mark xmlns="http://www.opengis.net/sld">
      <WellKnownName xmlns="http://www.opengis.net/sld">circle</WellKnownName>
      <Stroke xmlns="http://www.opengis.net/sld">
        <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke" >#000000</CssParameter>
        <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke-width" >0</CssParameter>
      </Stroke>
      <Fill xmlns="http://www.opengis.net/sld">
        <CssParameter xmlns="http://www.opengis.net/sld" :name="fill" >#32cf96</CssParameter>
      </Fill>
    </Mark>
```

```
        <Size xmlns="http://www.opengis.net/sld">2</Size>
      </Graphic>
    </PointSymbolizer>
```

Zum Schluss werden die noch offenen Bereiche geschlossen.

```
  </Rule>
  ...
 </FeatureTypeStyle>
</UserStyle>
</UserLayer>
</StyledLayerDescriptor>
```

4.2.1 Einbinden von Scalable Vector Graphics SVG

Die Standardsymbolisierung, welche durch die Elemente *CssParameter* in *SLD* beschrieben wird, kann durch SVG-Grafiken ergänzt oder ersetzt werden. Damit lassen sich beispielsweise einfarbige Flächenfüllungen mit Texturen belegen oder Punkten komplexe Symbole zuweisen, welche beim Export des QGIS-Projekts verloren gegangen sind.

Folgendes Beispiel zeigt die Symbolisierung der Fläche eines Polygons durch Repetition einer SVG-Grafik.

```
<PolygonSymbolizer xmlns="http://www.opengis.net/sld">
```

Man kann diese Art von Muster auch auf linienhafte Elemente beziehungsweise Flächenumrisse anwenden, die sich dann wie schlangenförmige Polygone verhalten. Allerdings ist zu beachten, dass auch diese wie die Flächen in Form eines Rasters gefüllt werden, ohne auf den Verlauf der Grundlinie zu achten.[7] Aus diesem Grund wird für die Umrandung eine einfache Linie mit gewünschter Farbe und Strichdicke bestimmt.

```
<Stroke xmlns="http://www.opengis.net/sld">
  <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke" >#000000</CssParameter>
  <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke-width" >0</CssParameter>
</Stroke>
<Fill xmlns="http://www.opengis.net/sld">
```

Innerhalb von *Fill* können nun SVG-Abschnitte durch das Element *pattern* eingebunden werden. *Pattern* bedeutet, dass die SVG-Grafik, wie im vorliegenden Beispiel mit einer Grösse von 40x40 Pixel auf 20x20 Pixel skaliert und die Fläche durch horizontales und vertikales Wiederholen gefüllt wird. Die Attribute *width* und *height* erlauben somit die Feinheit der Signaturen anzupassen und beispielsweise entsprechend den Massstabsverhältnissen auszugeben.

```
<pattern width="20" height="20">
  <g>
    <svg viewBox="0 0 40 40">
```

[7] Ueberschär, N., Winter, A. (2006): Visualisierung von Geodaten mit SVG im Internet. Heidelberg: Herbert Wichmann Verlag, 296 S.

Der Bereich *svg* enthält anschliessend die Geometrien, welche die Signatur ausmachen.

- Rechtecke mit entsprechender Höhe und Breite werden durch das Element *rect* wiedergegeben.

```
<rect style="fill:#d9db92;fill-opacity:1;" id="rect3236" width="40" height="40" x="0"
       y="0" />
```

- Mit dem *path*-Element sind beliebige Linienverläufe darzustellen. Das Attribut *style* bestimmt alle möglichen Aspekte zum Aussehen der jeweiligen Geometrie, wie beispielsweise die Füllung oder die Umrandung.

```
<path style="fill:none;stroke:#a55a1f;stroke-width:2px;" d="M 10,0 L 16,8 L 28,6 L 40,18"
       id="path3296" />
<path style="fill:none;stroke:#a55a1f;stroke-width:2px;" d="M 0,18 L 10,24 L 8,34 L 10,40"
       id="path3298" />
<path style="fill:none;stroke:#0d00ea;stroke-width:2px;" d="M 0,0 C 8,12 0,12 14,16 C
       28,20 20,24 30,32 C 40,40 40,40 40,40 L 40,40 L 40,40" id="path3302" />
```

- Ebenfalls werden Kreise und Ellipsen durch *path* definiert.

```
<path sodipodi:type="arc" style="fill:#4d4d4d;fill-opacity:1;stroke:#201811; stroke-
       width:1;" id="path3304" sodipodi:cx="10" sodipodi:cy="16.5" sodipodi:rx="3"
       sodipodi:ry="1.5" d="M 13,16.5 A 3,1.5 0 1 1 7,16.5 A 3,1.5 0 1 1 13,16.5 z"
       transform="scale(2,2)" />
    </svg>
   </g>
  </pattern>
 </Fill>
</PolygonSymbolizer>
```

4.3 OpenLayers

Nach der Einführung der nötigen serverseitigen Anwendung, das heisst deren Installation, Konfiguration und den Methoden, wie diese angesprochen werden kann, folgt nun der Überblick über *OpenLayers* als möglicher Ansatz für eine clientseitige Lösung einer dynamischen Webkarte. Eine umfassende Klassendokumentation ist im Internet verfügbar.

Als Projekt der *Open Source Geospatial Foundation OSGeo* ist *OpenLayers* eine freie JavaScript-Bibliothek zur Anzeige von dynamischen Karten im Webbrowser ohne serverseitige Abhängigkeiten. Dabei wird *OpenLayers* vor allem als Client für *Web Map Services* eingesetzt.[8]

OpenLayers arbeitet mit den beiden zentralen Komponenten *Map* und *Layer*. Das Objekt *Map* beinhaltet Informationen über die Projektion, den Perimeter, die Einheiten und weiteren allgemeinen Angaben. Layer-Objekte werden anschliessend dem Map-Objekt zugefügt und visualisieren die entsprechenden Datenquellen.[9]

4.3.1 Objekt Map

Neben der entsprechenden *id* können dem Map-Konstruktor weitere Eigenschaften optional zum Erzeugen eines neuen Objekts mitgegeben werden. Diese betreffen, wie in der Einleitung schon erwähnt, die Projektion, die Einheiten, die Kartenausdehnung sowie die möglichen Zoomstufen.

```
var options = {
    projection: new OpenLayers.Projection("EPSG:21781"),
    units: "m",
    maxExtent: new OpenLayers.Bounds(519000, 64000, 719000, 264000),
    controls: [],
    scales: [250000,125000,100000,75000,50000,25000]
}
var map = new OpenLayers.Map( "mymap", options );
```

Das Objekt *Map* wird nach dem Erzeugen in ein beliebiges Element eines HTML-Dokuments eingebunden. Dazu wird dem entsprechenden Element die eindeutige Identifikation mittels des Attributs *id* zugewiesen.

```
<html>
  <head>
    <title>Naturgefahrenhinweiskarte Kanton Bern</title>
  </head>
  <body>
    <div id="mymap"></div>
  </body>
</html>
```

Anschliessend können Layers verschiedenster Datenquellen erzeugt und dem Map-Objekt hinzugefügt, sowie ein Zentrum und eine Zoomstufe für einen Bildschirmausschnitt bestimmt oder auf den gesamten Umfang gezoomt werden.

[8] Enzyklopädie Wikipedia (2009): OpenLayers. http://de.wikipedia.org/wiki/OpenLayers (Zugriff am 18. April 2009)

[9] Openlayers.org (2009): What is OpenLayers? http://docs.openlayers.org/ (Zugriff am 18. April 2009)

4.3.2 Objekt Layer

Über das Objekt *Layer* werden verschiedene Datenquellen und –Formate in die Karte eingebunden. Definiert werden dabei der Zugriff sowie die Darstellung der entsprechenden Quelle.

a. Base Layer

OpenLayers bietet Programmierschnittstellen, sogenannte *Application Programming Interface API,* für den Zugriff auf verschiedene bekannte Rasterdatensätze wie GoogleMaps oder VirtualEarth an. Diese Datensätze liegen meist in einer bestimmten Projektion vor, welche clientseitig nicht geändert werden kann. Deshalb eignen sie sich gut als Hintergrund und werden als *Base Layer* definiert, wodurch sie neben dem Koordinatensystem auch die Zoomstufen der Karte bestimmen. In einer *Map* ist jeweils nur ein einziger *Base Layer* aktiv. Sie schliessen sich gegenseitig aus.

b. Overlay Layers

Der *Base Layer* wird durch einen oder mehrere *Overlay Layers,* sogenannte *Non Base Layers,* überlagert. Sie haben keinen Einfluss auf die Zoomstufen und übernehmen die Projektion der Karte. Jedoch kann der Zoombereich, in welchem sie aktiv sein können, über die Parameter *min/max scale/resolution* bestimmt werden.

Im folgenden Codebeispiel wird der vorgängig für den *QGIS MapServer* konfigurierte Vektordatensatz mittels WMS-Standard angesprochen. Die Variable *url* enthält den Pfad zur *QGIS MapServer*-Anwendung im entsprechenden fcgi-Verzeichnis des WebServers, gefolgt von einem Fragezeichen *?* zur Definition der Abfrageeigenschaften.

```
var url = "http://karlinapp.ethz.ch/fcgi-bin/qgis_map_server/webgisbern/qgis_map_serv.fcgi?";
```

Der *HTTP GET Request* wird spezifiziert durch entsprechende Parameter *params*. Die Transparenz kann in Zusammenhang mit den Grafikformaten *PNG* und *GIF* (veraltet) aktiviert werden und steht für *JPEG* nicht zur Verfügung. Dann folgt, wie im Abschnitt *Aufruf des QGIS MapServers* ausführlich beschrieben, die Liste der gewünschten Ebenen mit den dazugehörigen Styles.

```
var params = {
    transparent: "true",
    format: "image/png",
    layers: "pri25_a",
    styles: "uniquevalues"
};
```

Die Optionen *options* dienen zur Konfiguration der einzelnen Ebene in *OpenLayers*. Wie schon erwähnt, muss angegeben werden, ob die Ebene als *Base Layer* oder als *Overlay* eingebunden wird. Die Deckkraft sowie der Buffer sind zu bestimmen. Letzterer gibt an, wie viele Kacheln zusätzlich zum aktuellen Bildschirmausschnitt für ein rasches Verschieben der Karte zu laden sind.

```
var options = {
    isBaseLayer: false,
    opacity: 0.6,
    buffer: 0
};
```

Nach den Definitionen wird das Layer-Objekt erzeugt und dem Map-Objekt zugefügt.

```
new layer = OpenLayers.Layer.WMS.Untiled("pri25_a", url, params, options);
map.addLayer(layer);
```

Neben dem Einbinden von vorkonfigurierten Services gemäss obigem Beispiel besteht auch die Möglichkeit der Überlagerung von Vektordaten aus einer Datei mittels des *Vector Layer*. Verschiedene Standards wie GML können entweder direkt angesprochen oder müssen noch konfiguriert werden. Ebenso können Geometrieelemente für einen *Vector Layer* mit JavaScript erzeugt werden.

Weiter bestehen die Objekte *Marker* und *Text Layer*, mit welchen Punkte symbolisiert und beschriftet werden können.

4.3.3 Controls

Mittels *Controls* wird die Karte verändert, das heisst *OpenLayers* stellt dem Nutzer ein Interface zum Verändern des Kartenausschnitts bereit. *Pan* zum Verschieben sowie verschiedenen Funktionen zum Zoomen können angesprochen werden.

Folgende Controls sind meistens aktiviert: *Attribution* dient zur Beschreibung einzelner Layer mit Metainformationen direkt in der Karte. *Navigation* enthält die Controls *DragPan* und *ZoomBox* und kontrolliert somit die Mausaktionen des Benutzers zur Manipulation des Kartenausschnitts. Dieser kann beispielsweise verschoben oder eine Box zur Wahl eines Detailausschnitts kann aufgezogen werden. *PanZoom* erlaubt eine Manipulation des Kartenausschnitts über entsprechende Buttons.

Mit den folgenden Beispielen aus der vorliegenden Arbeit zur Visualisierung der Naturgefahrenhinweiskarte werden zum Einen die Navigation in einer Karte aktiviert, zum Anderen die Mausposition auf der Karte visualisiert.

```
var navControl = new OpenLayers.Control.Navigation({
    type: OpenLayers.Control.TYPE_TOGGLE,
    zoomWheelEnabled: true
});
map.addControl(navControl);

var mousePositionControl = new OpenLayers.Control.MousePosition({
    div: OpenLayers.Util.getElement('mousePos'),
    prefix: 'E: ',
    separator: '  N: ',
    numDigits: 0
```

```
});
map.addControl(mousePositionControl);
```

Ebenso wird eine Übersichtskarte in die Applikation als *Control* eingebunden. *OpenLayers* markiert in der *OverviewMap* den Kartenausschnitt mit einem Rechteck, welches verschoben werden kann und damit den Kartenausschnitt verändert.

```
var overviewMap = new OpenLayers.Control.OverviewMap({
            div: OpenLayers.Util.getElement('overview'),
            size: new OpenLayers.Size(200, 128),
            mapOptions: {
                projection: new OpenLayers.Projection("EPSG:21781"),
                units: "m",
                maxExtent: new OpenLayers.Bounds(519000, 64000, 719000, 264000),
                numZoomLevels: 4
            }
        });
        map.addControl(overviewMap);
```

In der Werkzeugsammlung *Panel* können mehrere *Controls* zur Steuerung der Karte zusammengefasst werden. Die Darstellung der einzelnen Buttons für den Aufruf der entsprechenden Funktionalität erfolgt über *Cascading Style Sheets CSS*.

5. Realisierung / Umsetzung

Nach dem projektorientierten Vorgehen werden nun die Erkenntnisse aus der Diskussion des darzustellenden Sachverhalts aus Kapitel *3. Thematische Einordnung* mit den erarbeiteten technischen Möglichkeiten aus Kapitel *4. Technische Grundlagen* umgesetzt. Die vorgängig einzeln analysierten und beschriebenen Aspekte werden zu einem Ganzen zusammengeführt und eine praxistaugliche Internetanwendung zur Visualisierung der Naturgefahrenhinweiskarte des Kantons Bern erstellt.

Das heisst, die Ebenen der Gefahrenhinweiskarte werden möglichst auf Grund ihrer natürlichen Erscheinungsformen für den *QGIS MapServer* aufbereitet. Die Massstabsvariabilität ist bei der Symbolisierung zu berücksichtigen und die Darstellung der einzelnen Layer aufeinander abzustimmen. Benutzerspezifische Veränderungen sind zu ermöglichen. Die Informationen werden mit geeigneten Basiskartenebenen hinterlegt.

Der *QGIS MapServer* in Kombination mit der Datenhaltung in *PostGIS* wird für die Ebenen der Naturgefahrenhinweis- sowie die Hintergrundkarte aufgesetzt und die clientseitige Applikation *OpenLayers* bereitgestellt. Folgendes Schema gibt einen Überblick über den Aufbau der Anwendung. Die einzelnen Elemente werden anschliessend detailliert besprochen.

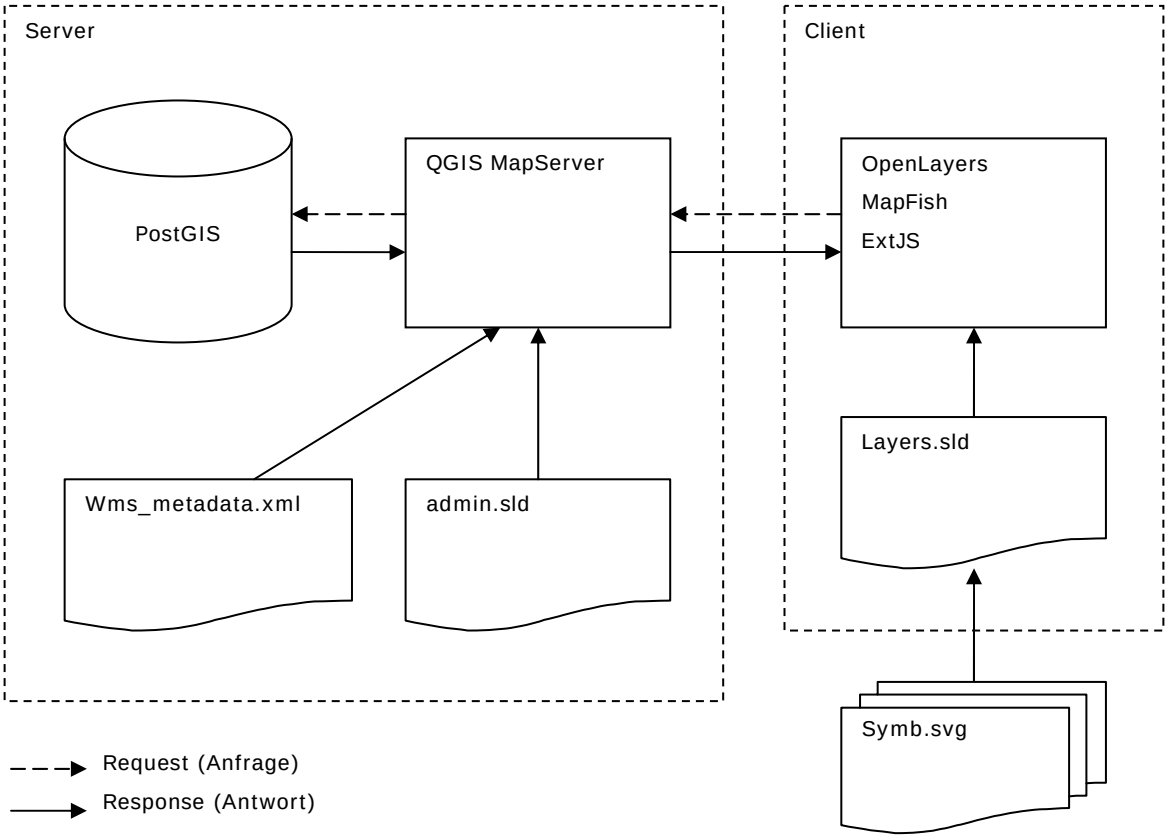


Abbildung 2: Schema über den Aufbau einer dynamischen Internetkarte

5.1 Datenhaltung in PostGIS

PostGIS erweitert die objektrelationale Datenbank *PostGreSQL* um geografische Objekte sowie Funktionen zu deren Manipulation und implementiert die *OpenGIS Simple Feature Access Spezifikation* des *Open Geospatial Consortium OGC*.[10] Auf Grund des beschränkten Umfangs des Berichts kann an dieser Stelle nicht näher auf die Konfiguration sowie die Funktionalität der Datenbank eingegangen werden. Ich verweise auf die Dokumentationen im Internet.

Die Inhalte aus den im ESRI Shapeformat vorliegenden Vektorinformationen der einzelnen Ebenen der Naturgefahrenhinweiskarte werden mit dem Programm *shp2pgsql* als *Structured Query Language SQL*-Statements in die jeweilige Dateien geschrieben.

```
USAGE: /usr/local/pgsql/bin/shp2pgsql.exe [<options>] <shapefile> [<schema>.]<table> > <SQL-File>
/usr/local/pgsql/bin/shp2pgsql ghkdol97.shp ghkdol97 > ghkdol97.sql
```

Diese SQL-Statements werden anschliessend über die Datenbankschnittstelle *psql* abgesetzt und damit die entsprechenden Tabellen erstellt und die Vektordaten importiert.

```
psql -h localhost -p 5000 -d nathazbern -U andreas -f ghkdol97.sql
```

Zur beschleunigten Abfrage und somit auch zur effizienten Darstellung der Geometrie wird das Feld *the_geom* indexiert.

```
create index ghkdol97_sindex on ghkdol97 using gist (the_geom);
```

Als Alternative bietet auch *QGIS Desktop* ein Plug-In mit übersichtlicher Benutzeroberfläche für den Import von *ESRI Shapes* in *PostGIS* an. Versuche zeigen jedoch, dass der Import grösserer Datensätze, wie beispielsweise den *VECTOR25* mit den Informationen der Landeskarte 1:25‘000, auf entfernten Servern wegen Zeit- und Volumenlimiten des Netzwerks zu Unterbrüchen führen kann.

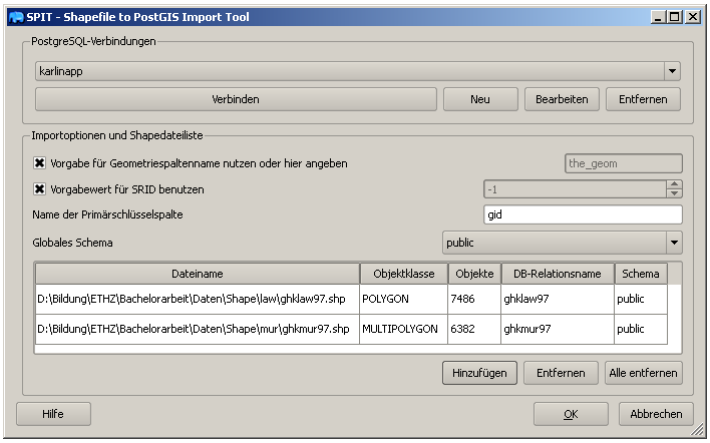


Abbildung 3: QGIS Desktop Plug-In für den Import von ESRI Shapefiles in PostGIS

[10] Enzyklopädie Wikipedia (2009): PostGIS. http://de.wikipedia.org/wiki/PostGIS (Zugriff am 25. April 2009)

5.2 Konfiguration der Ebenen für QGIS MapServer

Nach dem die geografischen Daten in der Datenbank bereit liegen, wird der Kartenserver wie in Kapitel *4. Technische Grundlagen* beschrieben installiert und die Metabeschreibung in der Datei *wms_metadata.xml* dem Projekt angepasst. Anschliessend erfolgt die Einbindung der darzustellenden Ebenen in *admin.sld*.

5.2.1 Basiskarte (Grundkarte)

Für die Darstellung der Prozessräume und Vorkommnisse, das heisst der georäumlichen Lage und Verbreitung der einzelnen Naturgefahren, wird eine topografische Karte als Basiskarte benötigt, welche durch Zeigen von Form, Gliederung und Bedeckung des Geländes eine Orts- und Lagebeschreibung gibt.

Die Basiskarte bietet in Bezug auf die thematischen Inhalte eine geometrische Bezugs- und Orientierungsmöglichkeit und lebt von der Beziehung des topografischen Elements zur thematischen Aussage. Dabei soll sie das eingefügte Thema möglichst sinnvoll ergänzen, aber möglichst wenig stören, und muss einen aktuellen Stand aufweisen. Massstab und Generalisierungsgrad müssen dem einzufügenden thematischen Inhalt entsprechen.[11]

a. Pixelkarte (Detailreiche Basiskarte)

Das Bundesamt für Landestopografie bietet die einzelnen Kacheln der Landeskarten digital im Rasterformat für das gesamte Gebiet der Schweiz an. Die Kartenfolge bietet Massstäbe von 1:25‘000 bis zu 1:1 Million mit entsprechenden Generalisierungsgraden und erfüllt damit einen Teil der Anforderungen an Hintergrundinformationen für unterschiedliche Bildschirmapplikationen sowie geografischer Informationssysteme GIS. Zudem unterliegen die Daten einer periodischen Nachführung.

Das Kartenwerk wird beispielsweise wie für die vorliegende Arbeit fertig konfiguriert über einen bestehenden *Web Map Service* bezogen. Dabei kann die Quelle in *QGIS MapServer* eingebunden und die Karte als neuer Layer zur Verfügung gestellt oder der Dienst direkt clientseitig geladen werden. Der Service liefert dem Aufruf entsprechend den richtigen Kartenausschnitt im passenden Massstab. Die Farben entsprechen der offiziellen Codierung des Bundesamtes, variabel bleibt dagegen die Opazität.

```
http://www.geofoto.ch/cgi/mapserv?
                              SERVICE=WMS&
                              VERSION=1.1.1&
                              REQUEST=GetMap&
                              MAP=/home/an/mapserverdata/relief_u_pk_kombi/relief_u_pk_kombi_jpeg.map&
                              LAYERS=pk_kombi&
                              FORMAT=image/jpeg&
                              SRS=EPSG:21781&
                              BBOX=613826.61546029,163342.16857745,618342.16857745,167857.72169461&
                              WIDTH=256&HEIGHT=256
```

[11] Hurni, L. (2008): Einführung in die thematische Kartografie. Vorlesung Thematische Kartografie, Herbstsemester 2008, ETH Zürich.

Mit dem breiten Informationsgehalt wie Kartennetz, Fixpunkte, Bauwerke, Gewässernetz, Höhenkurven und Reliefschummerung, Nomenklatur, Administrative Grenzen sowie Kulturgrenzen zählt die Landeskarte zu den detailreichen Basiskarten. Sie ist eher zur Beantwortung von Einzelfragen oder für die Detaillokalisierung geeignet.[12] Zu problematischen Konflikten führt die fix in die Rasterebene integrierte Beschriftung durch die Überlagerung der Naturgefahren mit der den Erscheinungsformen entsprechender Symbolisierung. Der Grundsatz, dass die Beschriftungen über den restlichen Kartenelementen liegen und wenn möglich freigestellt werden sollten, würde nicht eingehalten. So wird die Landeskarte in der vorliegenden Arbeit als Übersichtskarte verwendet und für den Hintergrund eine vereinfachte Basiskarte realisiert.

b. VECTOR 25 (Vereinfachte Basiskarte)

Das Bundesamt für Landestopografie bietet die Landeskarte 1:25‘000 ebenfalls im Datensatz *VECTOR25* in vektorieller Form mit getrennten Ebenen an. Für die vorliegende Arbeit kann somit eine gezielte Auswahl getroffen werden und die entsprechenden Shapes können ebenfalls in die im letzten Abschnitt besprochene Datenhaltung integriert werden.

Wie in Kapitel *3. Thematische Einordnung* bereits erwähnt, stehen die Naturgefahren in besonderer Beziehung zur menschlichen Infrastruktur. Somit sind *Siedlung* aus der Ebene Primärflächen und die Ebene *Strassen* von besonderem Interesse. Ebenfalls müssen die *Waldflächen* mit ihrer Schutzfunktion aus der Ebene Primärflächen dargestellt werden. Von weiterer Bedeutung sind das *Gewässernetz* sowie für die Topografie ein *Rasterrelief* einzubinden.

Die angesprochenen Ebenen werden in *QGIS Desktop* geladen und die gewünschten Flächen über die entsprechenden Filter symbolisiert. Anschliessend wird die Datei *admin.sld* mit der entsprechenden Konfiguration für *QGIS MapServer* mit dem Plug-In *PublishToWeb* in *SLD* wie in Kapitel *4. Technische Grundlagen* besprochen exportiert. Die Darstellung beruht auf der offiziellen Farbcodierung für den *VECTOR25* des Bundesamtes und wurde für die explizite Verwendung als Basiskarte angepasst. Strichstärken wurden verfeinert und hellere sowie gräulichere Farben gewählt.

[12] Hurni, L. (2008): Einführung in die thematische Kartografie. Vorlesung Thematische Kartografie, Herbstsemester 2008, ETH Zürich.

5.2.2 Gefahrenhinweiskarte

Neben dem serverseitigen Filtern und Symbolisieren von Datenquellen wie am Beispiel der Ebenen der vereinfachten Basiskarte gezeigt, kann die Symbolisierung auch mittels des Parameters *SLD_BODY* mit dem *HTTP GET* Aufruf, wie in Kapitel 4 erwähnt, mitgeschickt werden.

Dies ermöglicht auf flexible Art und Weise verschiedenste Symbole für die unterschiedlichen Naturgefahren zu testen und erlaubt dem Benutzer, die Symbolisierung clientseitig anzupassen. So erfolgt die detailierte Beschreibung der Darstellung der Erscheinungsformen der einzelnen Naturgefahren im folgenden Abschnitt über den *Aufbau der Clientanwendung mit OpenLayers*.

Serverseitig müssen die einzelnen Ebenen für den *QGIS MapServer* als *UserLayer* mit Namen definiert und die Quelle der Datenhaltung angegeben werden. Ebenfalls kann ein Style für eine Abfrage ohne *SLD* bereitgestellt werden. Im Beispiel wird eine einfache Flächenfüllung mit Umrandung für alle Elemente eines jeweiligen Layers der Naturgefahrenhinweiskarte verwendet und damit serverseitig auf das Element *Filter* verzichtet.

```
<UserLayer xmlns="http://www.opengis.net/sld">
  <Name xmlns="http://www.opengis.net/sld">law</Name>
  <Title xmlns="http://www.opengis.net/sld"></Title>
  <Abstract xmlns="http://www.opengis.net/sld"></Abstract>
  <HostedVDS xmlns="http://www.opengis.net/sld" providerType="postgres" uri="dbname='gisdb' host=localhost
                        port=5432 user='andebe' password='andebe' table=&quot;ghklaw97&quot; (the_geom)
                        sql="epsg="21781" />
  <UserStyle xmlns="http://www.opengis.net/sld">
    <Name xmlns="http://www.opengis.net/sld">default</Name>
    <FeatureTypeStyle xmlns="http://www.opengis.net/sld">
      <Rule xmlns="http://www.opengis.net/sld">
        <PolygonSymbolizer xmlns="http://www.opengis.net/sld">
          <Stroke xmlns="http://www.opengis.net/sld">
            <CssParameter xmlns="http://www.opengis.net/sld" name="stroke" >#0055ff</CssParameter>
            <CssParameter xmlns="http://www.opengis.net/sld" name="stroke-width" >1</CssParameter>
          </Stroke>
          <Fill xmlns="http://www.opengis.net/sld">
            <CssParameter xmlns="http://www.opengis.net/sld" name="fill" >#c2fcff</CssParameter>
          </Fill>
        </PolygonSymbolizer>
      </Rule>
    </FeatureTypeStyle>
  </UserStyle>
</UserLayer>
```

5.3 Aufbau der Clientanwendung mit OpenLayers

In der vorliegenden Arbeit wurde die clientseitige Browseranwendung mittels *MapFish Client* als JavaScript-Framework realisiert und basiert damit auf der *Web 2.0* Technologie. *MapFish* integriert *OpenLayers* für die Zusammenstellung der Kartengrafik wie in Kapitel *4. Technische Grundlagen* detailiert besprochen sowie *ExtJS* und *GeoExt* für die Realisierung des *Graphical User Interface GUI*.

Auf die umfassende JavaScript-Bibliothek *ExtJS* zur Bildung umfangreicher Internet-Applikationen kann im Rahmen dieser Arbeit nur in wenigen Punkten eingegangen werden. Umfangreiche Dokumentationen des *Application Programming Interface API* sind im Internet zu finden.

Die Kartenapplikation besteht aus einzelnen Komponenten, welche mittels *JavaScript* als Objekte erzeugt und in den kommenden Abschnitten detailierter betrachtet werden. Die folgende Abbildung bietet eine Übersicht über die einzelnen Elemente des entwickelten *Graphical User Interface GUI.*

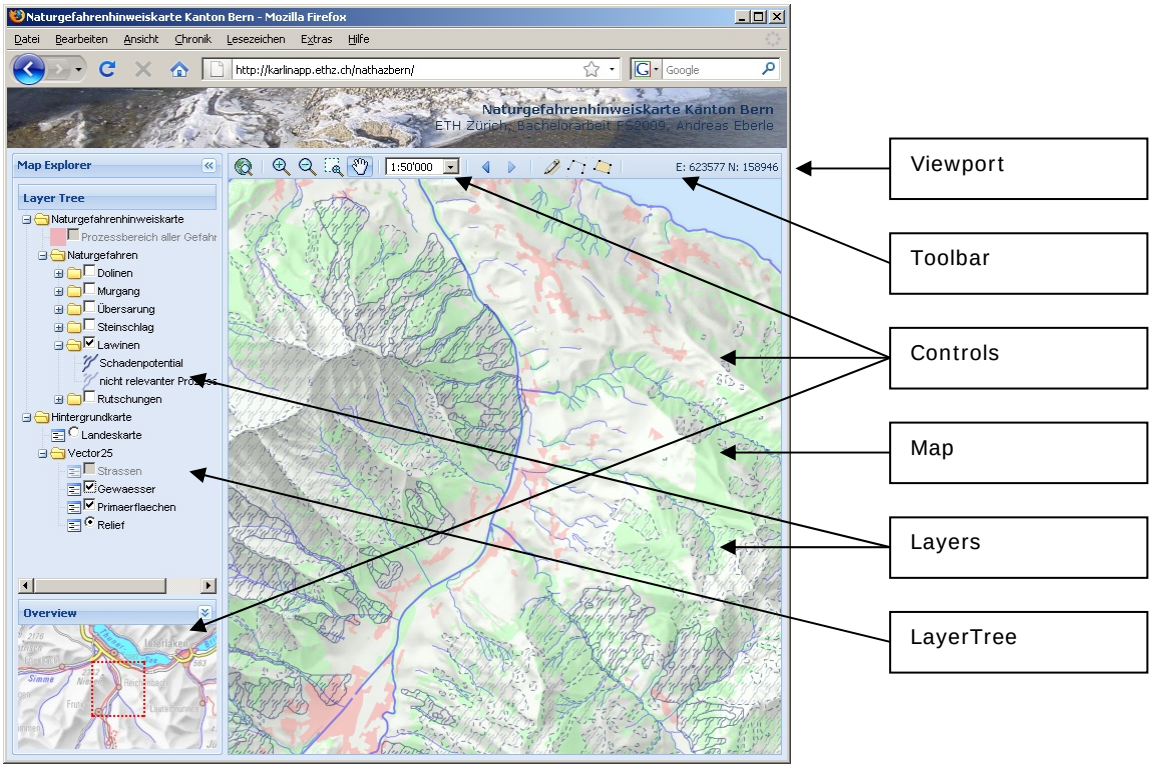


Abbildung 4: Elemente des Graphical User Interface GUI

5.3.1 HTML/JavaScript Applikation

Die ganze Anwendung ist in das HTML-Dokument *index.html* eingebunden, welche komplett im *Anhang B: Clientseitiger Code* wiedergegeben und hier im Aufbau erklärt wird. Sie enthält die Quellen zu den nötigen *Cascading Style Sheets CSS* und *JavaScript*-Bibliotheken.

```
<html>
  <head>
    <title>Naturgefahrenhinweiskarte Kanton Bern</title>

    <link rel="stylesheet" type="text/css" href="mfbase/ext/resources/css/ext-all.css" />
    <link rel="stylesheet" type="text/css" href="css/style.css" />

    <script type="text/javascript" src="mfbase/openlayers/lib/OpenLayers.js"></script>
    <script type="text/javascript" src="mfbase/ext/adapter/ext/ext-base.js"></script>
    <script type="text/javascript" src="mfbase/ext/ext-all-debug.js"></script>
    <script type="text/javascript" src="mfbase/mapfish/MapFish.js"></script>
```

Der nötige *JavaScript JS*-Code zur Erzeugung der jeweiligen Anwendungsobjekte wird der Übersichtlichkeit wegen in einzelne externe JS-Dateien geschrieben, welche wiederum im Header-Bereich des HTML-Dokuments eingebunden sind. Sie sind komplett im *Anhang B: Clientseitiger Code* zu finden.

```
    <script type="text/javascript" src="js/Controls.js"></script>
    <script type="text/javascript" src="js/Layers.js"></script>
    <script type="text/javascript" src="js/LayerTree.js"></script>
    <script type="text/javascript" src="js/Map.js"></script>
    <script type="text/javascript" src="js/Toolbar.js"></script>
    <script type="text/javascript" src="js/Viewport.js"></script>
```

Anschliessend folgen die Aufrufe der entsprechenden Klassen zur Erzeugung der angesprochenen Objekte.

```
    <script type="text/javascript">
        Ext.onReady(function() {
```

Als Erstes wird, wie in Kapitel *4. Technische Grundlagen* beschrieben, das OpenLayers-Objekt *Map* erzeugt und in das HTML-Element mit dem Attribut *id* und dem Wert *mymap* eingebunden. Der entsprechende Code ist in *Anhang B* unter *Map.js* zubinden. Da alle Ebenen der vorliegenden Arbeit im schweizerischen Projektionssystem *CH1903 / LV03* vorliegen, wird dieses über die entsprechende Nummer der *European Petroleum Survey Group EPSG* der Webkarte zugewiesen. Damit müssen die Daten zur Darstellung nicht aufwendig um projiziert werden.

Der gewünschte maximale Perimeter der Karte und die für den Nutzer abrufbaren Zoomstufen werden gesetzt. Die Daten der Naturgefahrenhinweiskarte wurden unter anderem auf Grundlage der Landeskarte 1:25‘000 erstellt. Diese Grenze sollte deshalb mit der grössten Zoomstufe nicht deutlich unterschritten werden.

```
Map.createMap('mymap', {
        projection: new OpenLayers.Projection("EPSG:21781"),
        units: "m",
        maxExtent: new OpenLayers.Bounds(519000, 64000, 719000, 264000),
        controls: [],
        scales: [250000,125000,100000,75000,50000,25000,12500]
    }
);
```

Nach dem Erzeugen des Map-Objekts können die *Controls* zur Manipulation der Karte zugefügt werden (*Anhang B*, *Controls.js*). Für die vorliegende Anwendung wird das Vergrössern und Verkleinern der Karte über zwei Buttons sowie die direkte Auswahl des gewünschten Massstabes aus einer Liste ermöglicht. Ebenfalls kann eine Box zum Bestimmen des Ausschnitts auf der Karte aufgezogen werden. Das Verschieben durch den Mauszeiger kann direkt in der Karte oder durch die Ausschnittsmarkierung in der Übersicht *OverviewMap* erfolgen.

```
Controls.addMapControls();
```

Nun werden die Layers für die Karte geladen (*Anhang B*, *Layers.js*). Wie im vorhergehenden Abschnitt angesprochen, wird die Landeskarte als detailreiche Basiskarte von einem bestehenden *Web Map Service* durch *OpenLayers* bezogen. Dazu wird der eindeutige Name des Layer-Objekts angegeben und der Aufruf parametrisiert.

```
Layers.createWmsLayer(
    "Swisstopo",
    "http://www.geofoto.ch/cgi/mapserv?",
    {map: '/home/an/mapserverdata/relief_u_pk_kombi/relief_u_pk_kombi_jpeg.map', layers:
              'pk_kombi', format: 'image/jpeg', SRS: 'EPSG:21781'},
    {isBaseLayer: true, attribution: "Pixelkarte (c) Swisstopo", buffer: 0, visibility: false}
);
```

Ebenso sollen als vereinfachte Basiskarte einzelne Ebenen des *VECTOR25* sowie das Relief, welche vorgängig in *QGIS MapServer* eingebunden und symbolisiert wurden, visualisiert werden.

```
var serv = "http://karlinapp.ethz.ch/fcgi-bin/qgis_map_server/webgisbern/qgis_map_serv.fcgi?";
Layers.createWmsLayer(
    "relief50",
    serv,
    {format: "image/jpeg", layers: "relief50", styles: "default"},
    {isBaseLayer: false, opacity: 0.5, buffer: 0, visibility: false}
);
Layers.createWmsLayer(
    "pri25_a",
    serv,
    {transparent: "true", format: "image/png", layers: "pri25_a", styles: "uniquevalues" },
    {isBaseLayer: true, opacity: 1, buffer: 0, visibility: false}
);
Layers.createWmsLayer(
    "str25_l",
    serv,
    {transparent: "true", format: "image/png", layers: "str25_l", styles: "uniquevalues" },
    {isBaseLayer: false, opacity: 0.4, buffer: 0, visibility: false}
);
Layers.createWmsLayer(
    "gwn25_l",
    serv,
    {transparent: "true", format: "image/png", layers: "gwn25_l", styles: "uniquevalues" },
    {isBaseLayer: false, opacity: 0.4, buffer: 0, visibility: false}
);
```

Die eigentlichen Ebenen der Naturgefahrenhinweiskarte, welche in *QGIS MapServer* nur rudimentär und ohne spezielle Symbolisierung eingebunden wurden, sollen nun mit dem Aufruf mitgeschickter *Styled Layer Descriptor SLD*-Definition dargestellt werden. Dies ermöglicht, viele verschiedene SVG-Symbole einfach zu testen. Die speziell dafür entwickelte Funktionalität wird deshalb im folgenden Abschnitt *Visualisierung der Naturgefahren* ausführlich erläutert.

```
Layers.createLayersFromSLD(
    "svg/sldLayers.xml",
    serv,
    {isBaseLayer: false, opacity: 0.8, buffer: 0, visibility: false}
);
```

In der Werkzeugleiste, dem MapFish-Objekt *Toolbar*, werden die Buttons mit den *Controls* zur Steuerung der Karte zusammengefasst (*Anhang B, Toolbar.js*).

```
Toolbar.createToolbar();
Toolbar.setToolbarContent();
```

Das ExtJS-Objekt *Viewport* positioniert alle vorgängig erzeugten Objekte im HTML-Dokument und legt das Erscheinungsbild wie Grösse und Funktionalität der einzelnen Bereiche fest (*Anhang B, Viewport.js*).

```
Viewport.createViewport();
```

Im MapFish-Objekt *LayerTree* werden die vorgängig erzeugten Layer eingebunden (*Anhang B, LayerTree.js*). Die Abfolge in der Karte sowie eine Gruppierung zusammengehörender Ebenen kann nun festgelegt und eine Manipulation dieser Überlagerung per *Drag and Drop* ermöglicht werden. Durch ein Kontextmenu per rechte Maustaste wird dem Anwender ebenfalls erlaubt, die Opazität eines einzelnen Layers zu verändern. Diese Anpassungsmöglichkeiten auf Benutzerseite sind für die vorliegende Anwendung mit einigen flächig überlagernden Erscheinungsformen besonders interessant.

Weiter besteht die Funktionalität im LayerTree-Model zur Angabe eines minimalen oder maximalen Massstabes, durch welche einzelne Einträge entsprechend deaktiviert oder wiederum aktiviert werden.

```
LayerTree.createLayerTree();
```

Zum Schluss der Generierung der JavaScript-Objekte wird die Karte auf eine gewünschte Position zentriert und eine Zoomstufe gewählt.

```
            map.setCenter(new OpenLayers.LonLat(619000, 164000),4);
        });    </script>
    </head>
```

Im Body werden HTML-Elemente als Platzhalter für die Objekte bereitgestellt.

```
    <body>
      <div id="north"><b>Naturgefahrenhinweiskarte Kanton Bern</b><br/>
      ETH Zürich, Bachelorarbeit FS2009, Andreas Eberle</div>
      <div id="tree"></div>
      <div id="legend"></div>
      <div id="overview"></div>
      <div id="mousePos"></div>
      <div id="mymap"></div>
    </body>
  </html>
```

5.3.2 Visualisierung der einzelnen Naturgefahren

Die besondere Herausforderung der Visualisierung der Naturgefahrenhinweiskarte besteht wie schon erwähnt darin, dass sechs verschiedene Prozessarten flächig dargestellt werden müssen. Nun können diesen Ebenen unterschiedlich einfache Flächenfüllungen zugewiesen und die Applikation mit einer detaillierten Legende, wie in vielen *Geografischen Informationssystemen GIS* im Internet üblich, ergänzt werden. Oder die bestehenden technischen Möglichkeiten werden nun genutzt und für jede Erscheinungsform wird jeweils eine realitätsnahe Symbolisierung entwickelt. Dies ist unter anderem auch ein wichtiges Ziel der vorliegenden Arbeit.

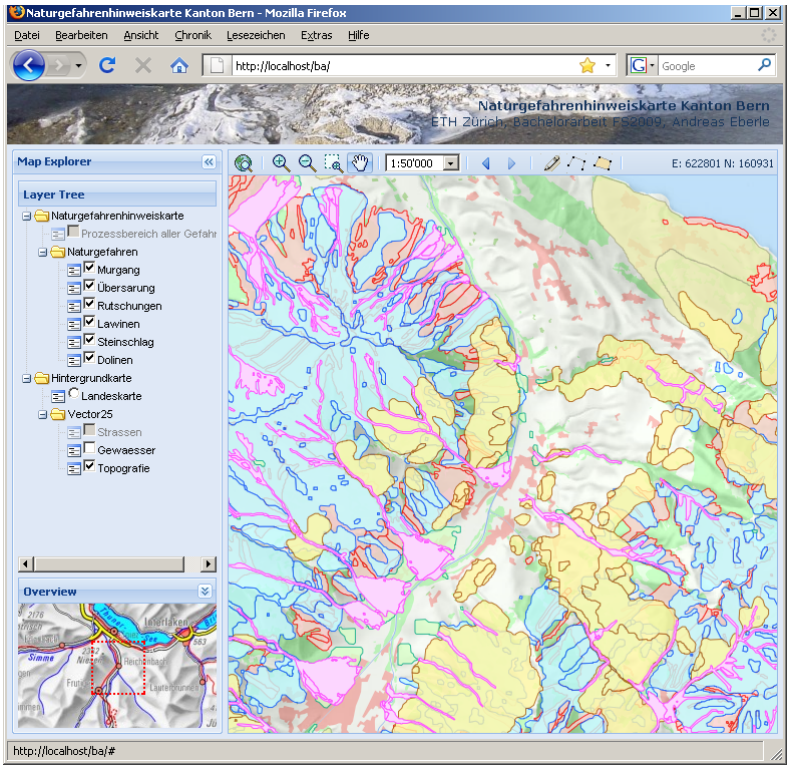


Abbildung 5: Halbtransparente Überlagerung der Naturgefahren mit einfacher Flächenfüllung

Neben einfachen Flächenfüllungen und Umrandungen können grafische SVG-Elemente in *Styled Layer Descriptor SLD* eingebunden werden. Dies kann direkt statisch in die *QGIS MapServer* Datei *admin.sld* geschrieben oder mit dem nun erläutertem Ansatz clientseitig in den *HTTP GET Request* integriert werden. Der grosse Vorteil von Letzterem besteht darin, dass die Datei *admin.sld* während der Testphase nicht ständig manuell abgeändert werden muss, sondern dass neue SVG-Grafiken erstellt und Elemente daraus automatisch in den *HTTP GET* Aufruf übernommen werden.

a. Styled Layer Descriptor SLD

Als Erstes werden die einzelnen Layer in Folge in der Datei *sldLayers.xml* in *SLD* beschrieben.

```
<StyledLayerDescriptor>
```

NamedLayer verweist auf einen auf dem Server bestehenden Layer mit dem entsprechenden Namen.

```
    <NamedLayer>
        <Name>law</Name>
```

Anschliessend folgt die benutzerdefinierte Symbolisierung in *UserStyle*.

```
        <UserStyle>
            <Name>law</Name>
            <FeatureTypeStyle>
```

Eine Regel mit entsprechendem Filter für die Darstellung nach einem bestimmten Attributwert wird definiert.

```
                <Rule>
                    <Filter>
                        <PropertyIsEqualTo>
                            <PropertyName>prozessber</PropertyName>
                            <Literal>1</Literal>
                        </PropertyIsEqualTo>
                    </Filter>
                    <PolygonSymbolizer>
```

Für die Umrandung wird eine einfache Signatur gewählt. Das Beispiel zeigt eine gestrichelte Linie von 4 Punkten mit jeweils 3 Punkten Abstand.

```
                        <Stroke>
                            <CssParameter name="stroke">#8173fe</CssParameter>
                            <CssParameter name="stroke-width">2</CssParameter>
                            <CssParameter name="stroke-dasharray">4 3</CssParameter>
                        </Stroke>
```

Die Fläche wird durch das SVG-Element *pattern* mit einer darin enthaltenen SVG-Grafik gefüllt. Das heisst, die entsprechende Grafik wird horizontal und vertikal repetiert und auf die Grösse des *patterns* wie im Beispiel 20 Pixel skaliert. Damit kann die Feinheit der Signatur angepasst werden. Dem SVG-Element wird das Attribut *name* zugewiesen, welches als Wert die Bezeichnung der entsprechenden SVG-Datei mit dem gewünschten Symbol enthält.

```
                        <Fill>
                            <pattern width="20" height="20">
                                <g>
                                    <svg name="law22.svg" viewBox="0 0 40 40"></svg>
                                </g>
                            </pattern>
                        </Fill>
                    </PolygonSymbolizer>
                </Rule>
                ...
            </FeatureTypeStyle>
        </UserStyle>
    </NamedLayer>
</StyledLayerDescriptor>
```

b. JavaScript-Automatismus

Die Layerfolge in *SLD* wird durch den JavaScript-Code *createLayersFromSLD* der Datei *Layers.js* (Anhang B) verarbeitet. Das heisst, in einer Schleife werden die einzelnen Ebenen als OpenLayers-Objekt *WMS* in die Karte eingebunden.

Der Funktion sind der Pfad der Datei *sldLayers.xml*, die Adresse des anzusprechenden WMS-Servers sowie Optionen für das OpenLayers-Objekt *WMS* anzugeben.

```
createLayersFromSLD: function(sldPath, url, options) {
```

Die SLD-Beschreibung wird per XML-Parser eingelesen und die einzelnen Layer nach dem Element *StyledLayerDescriptor* in eine Variable geschrieben.

```
var sldLayers = this.parseXML(sld);
var sldLayer = sldLayers.getElementsByTagName("StyledLayerDescriptor");
```

In der folgenden Schleife wird nun jeder einzelne Layer verarbeitet.

```
for (k=0;k<sldLayer.length;k++) {
```

Die SVG-Elemente innerhalb der Layerbeschreibung werden gesucht und alle diese innerhalb einer weiteren Schleife mit den entsprechenden Geometrieelementen der externen SVG-Grafiken komplettiert.

```
var svg = sldLayer[k].getElementsByTagName("svg");
for (l=0;l<svg.length;l++) {
```

Dazu wird der Name der externen SVG-Datei aus dem Attribut *name* des SVG-Elements gelesen und die komplette Pfadangabe dem XML-Parser zum Einlesen der entsprechenden Grafik übergeben.

```
var svgDocument =
        this.parseXML("svg/"+sldLayer[k].getElementsByTagName("svg")[l].getAttribute("name
        "));
```

In dieser externen Grafik werden die Geometrieelemente *rect* und *path* identifiziert und in die SLD-Beschreibung der jeweiligen Layer eingefügt.

```
var x=svgDocument.getElementsByTagName("rect");
var anz = x.length;
for (i=0;i<anz;i++) {
    sldLayer[k].getElementsByTagName("svg")[l].appendChild(x[0]);
}
var x=svgDocument.getElementsByTagName("path");
var anz = x.length;
for (i=0;i<anz;i++) {
    sldLayer[k].getElementsByTagName("svg")[l].appendChild(x[0]);
}
}
```

Der Name des neuen *OpenLayers*-Objekts wird aus dem Element *Name* gelesen und zum Schluss die mit den SVG-Elementen erweiterte SLD-Beschreibung den nötigen Parametern zum *HTTP GET Request* angehängt.

```
var name = sldLayer[k].getElementsByTagName("Name")[0].firstChild.data;
var params = {transparent: "true", format: "image/png", SLD_BODY:
        this.xmlToString(sldLayer[k]).replace(/  /g, "").replace(/\n/g,"")};
```

Mit allen nötigen Informationen wird nun der neue Layer als *OpenLayers*-Objekt erzeugt

```
        map.addLayer(new OpenLayers.Layer.WMS.Untiled(name, url, params, options));
    }
```

Die ganze Darstellung der einzelnen Ebenen der Naturgefahrenhinweiskarte kann nun durch Veränderung der SLD-Beschreibung zentral in einer Datei angepasst werden. Damit stehen alle Darstellungsmöglichkeiten von *SLD* zur Verfügung.

Für die Umrandung kann beispielsweise eine Farbe, eine Breite und wenn gewünscht eine Strichierung oder ein SVG-Symbol angegeben werden. Die Fläche wird durch ein SVG-Pattern mit einer SVG-Grafik gefüllt. Dabei wird die Geometrie der SVG-Grafik automatisch aus externen SVG-Dateien übernommen und nur der Dateiname ist manuell im Attribut *Name* des SVG-Elements anzupassen. Im kommenden Abschnitt folgt der Aufbau einer Sammlung von SVG-Grafiken zur Darstellung der einzelnen Naturgefahren.

Ebenfalls besteht so die Möglichkeit, die vorhandene Funktionalität des JavaScript-Codes zu ergänzen, um dem Benutzer weitere Interaktionen, wie die benutzerspezifische Auswahl der gewünschten Symbolisierung, zur Verfügung stellen zu können.

5.3.3 Scalable Vector Graphics SVG – Symbolsammlung

Für die einzelnen Ebenen der Naturgefahren werden nun SVG-Grafiken kreiert, welche in einer Fläche repetitiv wiedergegeben die naturräumlichen Erscheinungsformen möglichst gut abbilden und damit eine ausführliche Legende zum Verständnis des Kartenbildes erübrigen. Diese stehen als Sammlung zur Verfügung und können zur Darstellung mittels *Styled Layer Descriptor SLD* angegeben werden.

Für die Erstellung der einzelnen SVG-Grafiken wird der Vektorgrafikeditor Inkscape verwendet. Er bietet die Möglichkeit, die einzelnen grafischen Elemente auch direkt in SVG zu mutieren. Dies wird genutzt, um nicht zwingende Angaben des SVG-Codes für einen schlanken *HTTP GET* Aufruf zu eliminieren.

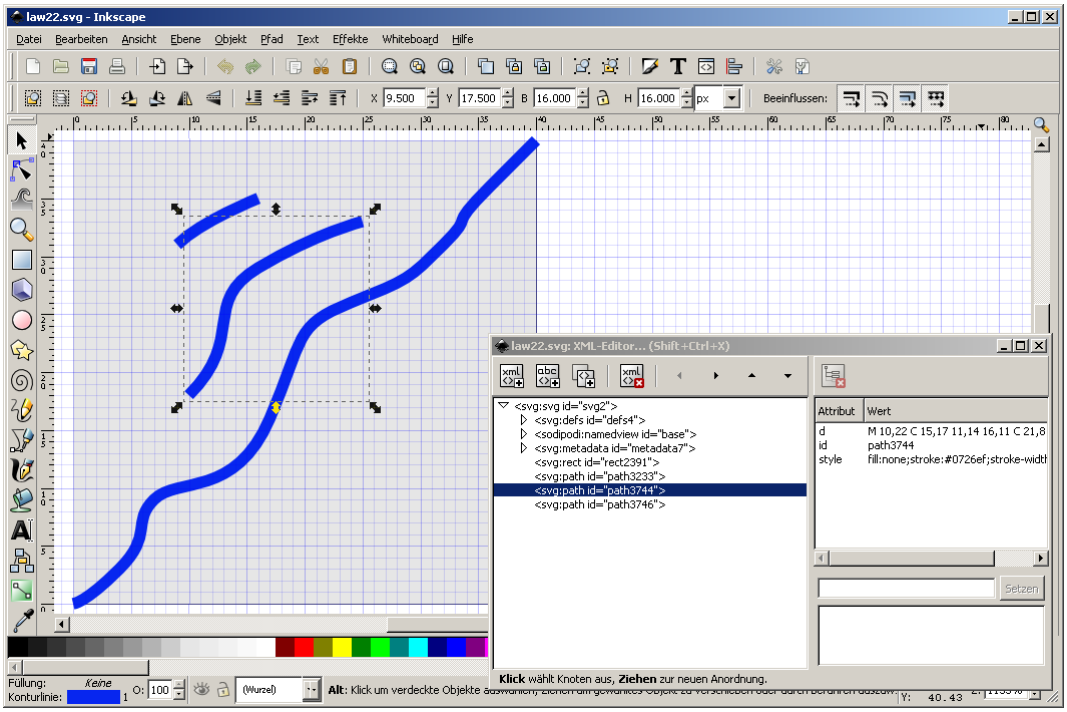


Abbildung 6: Vektorgrafikeditor Inkscape für die Erzeugung von Scalable Vector Graphics SVG

Für den Aufbau der SVG-Sammlung für die einzelnen Ebenen der Naturgefahrenhinweiskarte und damit die Festlegung der entsprechend typischen Werte der grafischen Variablen für die Füllung wie *Farben*, *Helligkeit* sowie die *Musterform*, *Korn-/Rasterweite* und die *Rasterwinkelung* der *Textur* wird auf Kapitel *3 Thematische Einordnung* zurückgegriffen. Systematisch werden die einzelnen Gefahren auf ihre Erscheinung analysiert, daraus die grafischen Variablen und die Darstellung abgeleitet.

Als Erstes stellt sich die Frage, ob die jeweilige Erscheinung eher flächenhaft ist oder durch einzelne Bruchelemente dominiert wird. Je nach Antwort wird ein entsprechendes Symbol flächenfüllender oder eben als offenes Raster realisiert und damit die Korn-/Rasterweite angepasst. Zudem kann daraus eine sinnvolle Überlagerung von mehreren Naturgefahren abgeleitet werden. Dazu muss das bewegte Material analysiert werden und die Ausprägung in die Textur einfliessen. Ebenfalls sind die Darstellungsfarben auf der natürlichen Erscheinung aufzubauen. Damit wird die Frage *Was gefährdet?* beantwortet.

Neben dem beteiligten Material sind die Bewegungsformen wie Fliessen oder Rutschen und entsprechende Geschwindigkeiten durch die Symbolisierung darzustellen. Auslösende Faktoren und berechenbares oder plötzliches Auftreten können allenfalls auch unterschieden werden. Einzelne Prozesse können andere Naturgefahren auslösen und sind in einer Prozesskette miteinander verbunden. Diese Verwandtschaft ist in der Symbolisierung ersichtlich. Damit wird die Frage *Wie bewegt sich die Gefährdung?* gelöst.

Die Naturgefahrenhinweiskarte bietet auf der Stufe der Richtplanung eine Übersicht über die potentiellen Gefahren und differenziert die einzelnen Prozessräume nur nach der Relevanz bezüglich eines möglichen Schadenpotentials. Diese Unterscheidung wird für alle Ebenen der Hinweiskarte durch die Flächenumrandung verdeutlicht. Der gesamte Prozessraum wird durch eine gestrichelte Linie mit einer relativ transparenten Füllung, der Bereich mit Relevanz für ein Schadenpotential mit einer durchgezogenen Linie und einer höheren Opazität dargestellt.

a. Rutschung

Die darzustellenden Ausprägungen der Naturgefahr *Rutschung* sind in folgender Übersicht zusammengetragen:

Material	Rutschungen beinhalten ganze Hangteile mit allenfalls eingelagerten Felsblöcken und Lockergestein sowie Wasser in Form von Poren-, Quelldruck oder Sickerströmung.
Bewegungsformen	Die Hangelemente können kriechend, gleitend, fliessend oder auch fallend zu Tal gehen. An den Grenz- und Gleitflächen entstehen dabei differentielle Bewegungen.
Auslösende Faktoren	Scherbrüche an den Massenuntergrenzen sowie Wassereinlagerungen führen zum Abrutschen der Hangteile.
Verwandtschaft	Rutschungen können durch weitere Wasserzufuhr zu Hangmuren werden oder Murgänge auslösen.
Klassifikation	- Tiefgründiger, relevanter Bereich bezüglich Schadenpotential (> 10 m) - Tiefgründiger, nicht relevanter Prozessbereich (> 10 m) - Mittel- und flachgründiger, relevanter Bereich bez. Schadenpotential (< 10 m) - Mittel- und flachgründiger, nicht relevanter Prozessbereich (< 10 m)

Aus den herausgearbeiteten Eigenschaften der Erscheinung *Rutschung* können folgende grafische Variable abgeleitet werden:

Farbe/Helligkeit	Aufgrund des bewegten Materials werden braune, grüne sowie eventuell blaue Farbtöne für die Symbolisierung verwendet.
Textur	Die Bewegung von ganzen Hangteilen wird durch eine flächige Textur mit eher grossen Rasterweiten dargestellt. Dabei könnten einzelne Einlagerungen in Form von grafischen Punktelementen wiedergegeben werden. Aufgrund der Übersichtlichkeit der Überlagerung durch weitere Naturgefahren wird darauf aber verzichtet. Die Scherbrüche sowie die kriechenden oder gleitenden Bewegungen, welche an der Massenuntergrenze als zum Hang horizontale Linien wahrgenommen werden, können durch ein Raster von parallel angeordneten Linien als Musterform wiedergegeben werden. Dabei wird die Rasterwinkelung im Gegensatz zur fliessenden Leserichtung (von oben links nach unten rechts) von Südwest nach Nordost gewählt und damit die Erscheinung der kompakteren Masse vermittelt.
Realisierung siehe	*Anhang C: Scalable Vector Graphics SVG – Symbolsammlung* *Anhang D: Darstellung der einzelnen Ebenen der Naturgefahren in der Kartenapplikation*

b. Steinschlag

Die darzustellenden Ausprägungen der Naturgefahr *Steinschlag* sind in folgender Übersicht zusammengetragen:

Material	Der Steinschlag beinhaltet Gestein als lose Einzelkomponenten bis hin zu einigen Kubikmetern in Form von Geröll oder Schutt.
Bewegungsformen	Dabei bewegen sich die einzelnen Masseelemente eher isoliert in der Art von Fallen, Springen oder Rollen hangabwärts.
Auslösende Faktoren	Lose Komponenten entstehen durch Frostsprengung, Erosion sowie Verwitterung. Regen- oder Schmelzwasser sowie Aktivitäten von Tier und Mensch können unter anderem zu den auslösenden Faktoren gezählt werden.
Verwandtschaft	Rutschungen unterscheiden sich von Steinschlag in der kompakteren und zusammenhängenden Masse mit einem Humus- und Wasseranteil. Fels- und Bergstürze sind definiert durch grössere Einzelkomponenten über einigen Kubikmetern.
Klassifikation	- relevanter Bereich bezüglich Schadenpotential - nicht relevanter Prozessbereich

Aus den herausgearbeiteten Eigenschaften der Erscheinung *Steinschlag* können folgende grafische Variable abgeleitet werden:

Farbe/Helligkeit	Aufgrund des steinigen Materials werden graue Farbtöne bis hin zu Schwarz für die Symbolisierung bevorzugt.
Textur	Die Bewegung von losen Einzelkomponenten wird durch eine offene Anordnung von einzelnen Punktelementen für kleine Gesteinsbrocken sowie Polygonflächen für Gesteinsblöcke dargestellt. Die Musterformen sind unterschiedlich und enthalten runde wie eckige Komponenten. Weiter soll die Rasterweite möglichst heterogen sein, so dass keine spezielle Rasterwinkelung erkennbar ist.
Realisierung siehe	*Anhang C: Scalable Vector Graphics SVG – Symbolsammlung* *Anhang D: Darstellung der einzelnen Ebenen der Naturgefahren in der Kartenapplikation*

c. Lawinen

Die darzustellenden Ausprägungen der Naturgefahr *Lawinen* sind in folgender Übersicht zusammengetragen:

Material	Schnee- und Eismassen bestimmen Lawinenniedergänge.
Bewegungsformen	Bewegungen von Fliesslawinen sind in ihrer Art gleitend, fliessend oder rollend. Staublawinen stürzen in Form von Schnee- oder Eiswolken zu Tal.
Auslösende Faktoren	Voraussetzung der Erscheinung sind grosse Schneemassen, in steilen Hängen liegend. Linienförmige Anrisse quer zum Hang führen schliesslich zum Niedergang.
Klassifikation	- relevanter Bereich bezüglich Schadenpotential - nicht relevanter Prozessbereich

Aus den herausgearbeiteten Eigenschaften der Erscheinung *Lawinen* können folgende grafische Variable abgeleitet werden:

Farbe/Helligkeit	Eis wird mit hellen Blautönen und Schneemassen mit einer halbtransparenten weissen Einfärbung wiedergegeben. Generell wird die Ebene auch zur Abgrenzung gegenüber den anderen Gefahren durch die Farbe Blau dominiert.
Textur	Die flächige Bewegung der Schnee- und Eismassen wird ebenfalls durch eine flächige Textur symbolisiert. Wie bei den Rutschungen werden die linienförmigen Anrisse, welche quer zum Hang stehen, durch parallel angeordnete Linien dargestellt. Jedoch werden nicht gerade, sondern wellenförmige Geometrieelemente zum Zeigen der teilweisen wolkenhaften Erscheinung als Musterform gewählt. Die Kornweite wird entsprechend der kompakten Masse eher eng gehalten und die Rasterwinkelung analog den Rutschungen von unten links nach oben rechts geführt.
Realisierung siehe	*Anhang C: Scalable Vector Graphics SVG – Symbolsammlung* *Anhang D: Darstellung der einzelnen Ebenen der Naturgefahren in der Kartenapplikation*

d. Murgang

Die darzustellenden Ausprägungen der Naturgefahr *Murgang* sind in folgender Übersicht zusammengetragen:

Material	Schlamm und Geschiebe als Gemisch von Wasser, Lockergestein sowie Feststoffen mit einem Anteil von 30 – 60 % prägen die Erscheinung.
Bewegungsformen	Die Masse wird als schnell fliessender Strom mit hoher Energie entlang des grössten Hanggefälles innerhalb von Geländeeinschnitten talwärts geführt. Dadurch werden die Feststoffe verlagert.
Auslösende Faktoren	Murgänge entstehen in Wildbächen mit grossem Gefälle und stark wechselnder Abflussmenge. Wenig verfestigtes Material von Geröll und Schutt wird wasserübersättigt und in Bewegung gesetzt. Hangmurgänge bilden sich an steilen Hängen mit gering durchlässigen, erosionsanfälligen Quartärbildung bei starken oder langanhaltenden Niederschlägen oder Schneeschmelze.
Verwandtschaft	Andere Hangprozesse wie Rutschungen oder Lawinen beliefern den oberen Gefahrenbereich mit Festmaterial. Im unteren Prozessraum führen Murgänge zu Ablagerungen in Form von Übermurungen.
Klassifikation	- relevanter Bereich bezüglich Schadenpotential - nicht relevanter Prozessbereich

Aus den herausgearbeiteten Eigenschaften der Erscheinung *Murgang* können folgende grafische Variable abgeleitet werden:

Farbe/Helligkeit	Aufgrund des bewegten Materials werden braune Farbtöne für die Feststoffanteile sowie blaue Farbtöne für die enthaltenen Wassermassen verwendet.
Textur	Die eher schmalen Prozessabschnitte innerhalb der Taleinschnitte sowie die auslaufenden Flächen in der Ebene sind mit einem flächigen Braunton dargestellt. Das enthaltene Wasser wird mit zueinander parallelen Wellenlinien als Musterform mit einer Rasterausrichtung von oben links nach unten rechts in Leserichtung als Zeichen für eine schnell fliessende Masse dargestellt. Der Abstand und damit die Rasterweite zwischen den Linien dürfen wegen der eher schmalen Prozessflächen nicht zu breit ausfallen.
Realisierung siehe	*Anhang C: Scalable Vector Graphics SVG – Symbolsammlung* *Anhang D: Darstellung der einzelnen Ebenen der Naturgefahren in der Kartenapplikation*

e. Übersarung

Die darzustellenden Ausprägungen der Naturgefahr *Übersarung* sind in folgender Übersicht zusammengetragen:

Material	Mit Übersarung werden Ablagerungen von Feststoffen ausserhalb eines Gerinnes bezeichnet.
Auslösende Faktoren	Überschwemmungen führen zu diesen Ablagerungen ausserhalb des Gerinnes im Bereich von Wildbachschwemmkegeln
Verwandtschaft	Bezeichnend oder verantwortlich für die Feststoffbereitstellung können beispielsweise Rutschungen sein. Anschliessend werden die Massen durch Murgänge innerhalb von Wildbächen zu Tal befördert.
Klassifikation	- relevanter Bereich bezüglich Schadenpotential - nicht relevanter Prozessbereich

Aus den herausgearbeiteten Eigenschaften der Erscheinung *Übersarung* können folgende grafische Variable abgeleitet werden:

Farbe/Helligkeit	Das abgelagerte Gemisch von Schlamm und Schutt weist eine relativ homogene braune Farbe auf. Nach Austrocknung wird diese an Helligkeit zunehmen. Da schon für Murgänge sowie Rutschungen unter anderem Brauntöne verwendet werden, wird für Übersarungen eine orange Färbung gewählt.
Textur	Wie schon erwähnt, kann das eintönige Material mit einer konstanten Flächenfarbe gefüllt werden. Allenfalls kann für kleine Einlagerungen eine feine Musterform von einzelnen Punktelementen oder parallelen Linien eingeführt werden. Die Rasterweite ist entsprechend der natürlichen Erscheinung relativ engmaschig. Die Rasterwinkelung kann frei gewählt werden.
Realisierung siehe	*Anhang C: Scalable Vector Graphics SVG – Symbolsammlung* *Anhang D: Darstellung der einzelnen Ebenen der Naturgefahren in der Kartenapplikation*

f. Dolinen

Die darzustellenden Ausprägungen der Naturgefahr *Dolinen* sind in folgender Übersicht zusammengetragen:

Material	Darzustellen ist bei der letzten Ebene der Hinweiskarte nicht eine voluminöse Materialansammlung, sondern im Gegenteil die Gefahr vor entstehenden Karsttrichtern, also einer schlot-, trichter- oder schüsselförmigen Senke von rundem oder elliptischem Grundriss.
Bewegungsformen	Der Boden über einem hohlen Karstraum kann in sich zusammenfallen.
Auslösende Faktoren	Zirkulierendes Wasser löst wasserlösliche Stoffe wie Gips oder Malmkalke. Wenn diese in grösseren Mengen im Untergrund vorkommen, entsteht ein verkarstetes Gebiet mit unterirdischen Hohlräumen, bei welchen die Decke zusammenbrechen kann.
Klassifikation	- schwache Disposition/Gefährdung - starke Disposition/Gefährdung

Aus den herausgearbeiteten Eigenschaften der Erscheinung *Dolinen* können folgende grafische Variable abgeleitet werden:

Farbe/Helligkeit	Die Darstellung dieser möglichen Trichter im Gelände basiert nicht auf der spezifischen Farbe eines bestimmten Materials. Vielmehr soll mit der Farbwahl der Eindruck einer Einsenkung unterstützt werden.
Textur	Die einzeln auftretenden Einsenkungen werden durch eine offene Anordnung von einzelnen Symbolen, welche für diese Erscheinung stehen, dargestellt. Als Musterform dienen konzentrische Kreise, die gegen das Zentrum dunkler eingefärbt sind. Die Kornweite kann relativ grob sein unter der Berücksichtigung, dass die Flächentextur nicht zerfällt.
Realisierung siehe	*Anhang C: Scalable Vector Graphics SVG – Symbolsammlung* *Anhang D: Darstellung der einzelnen Ebenen der Naturgefahren in der Kartenapplikation*

5.3.4 Integration der einzelnen Naturgefahren zu einem Kartenbild

Nach dem die Darstellung der Ebenen der Naturgefahrenhinweiskarte durch die Analyse ihrer Erscheinungsformen bestimmt ist, werden zum Schluss die einzelnen Layer in das Kartenbild der Applikation integriert und einzelne Farbgebungen zur klaren Unterscheidung zwischen den einzelnen Ebenen weiter differenziert. Sie überlagern die schon besprochenen Varianten der Basiskartendaten.

Wie zu Beginn des Berichts erwähnt, wurden die Naturgefahren auf zwei verschiedene Karten aufgeteilt gedruckt. Erd- und Felsbewegungen sowie Lawinen- und Murganggefahren wurden zusammengefasst. Eine solche statische Gruppierung möchte ich in dieser dynamischen Internetanwendung nicht verwenden, sondern auf die technischen Möglichkeiten zurückgreifen und dem Benutzer die Freiheit geben, einzelne Ebenen in der Karte frei ein- oder auszuschalten und die Reihenfolge der Überlagerung per *Drag and Drop* im Layerbaum zu verändern. Ebenfalls kann die Opazität pro Ebene benutzerspezifisch durch einen rechten Mausklick im Kontextmenu angepasst werden.

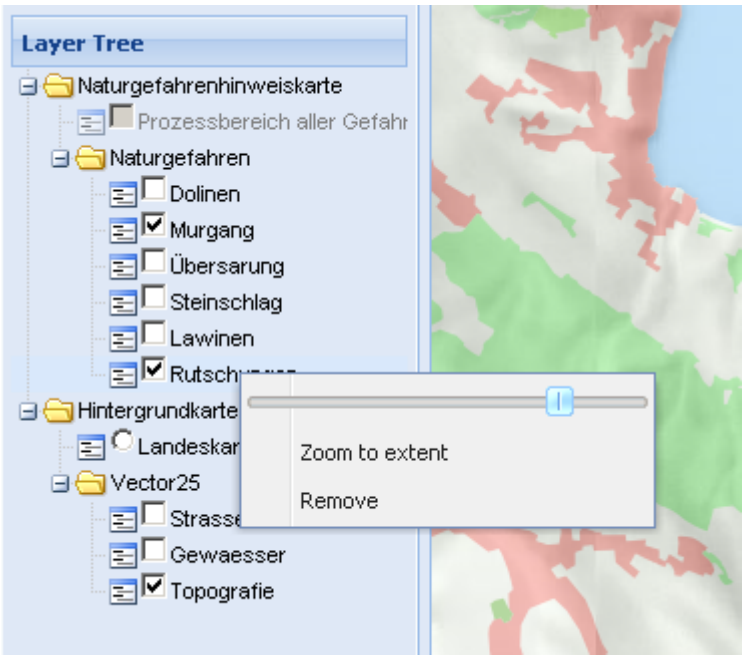


Abbildung 7: Layerbaum mit Möglichkeit zur Änderung der Überlagerung sowie der Opazität einzelner Ebenen

Damit bleibt die Anwendung flexibel und für unterschiedliche Fragestellungen können spezifische Kartenbilder ausgegeben werden. Alle Ebenen werden somit in einem Ordner *Naturgefahren* zusammengefasst und die anpassbare Reihenfolge der Überlagerung entsprechend der Rasterweite der jeweiligen Symbole aufgebaut. Stark überdeckende Texturen werden dabei von offener Symbolisierung überlagert.

Beim Erzeugen des *Map*-Objekts wurden sieben Massstabstufen zwischen 1:12‘500 und 1:250‘000 definiert. Da die Symbolisierung sowie die Unterscheidung der einzelnen Gefahren bei einer Massstabszahl grösser als 60‘000 schwierig wird, werden die Layer ab dieser Stufe deaktiviert und der *Prozessbereich aller Gefahren* als Übersicht aktiv. Diese Einteilung ist fix und kann durch den Benutzer nicht geändert werden.

6. Fazit und Ausblick

Da eine internetbasierte Kartenapplikation vorliegt, welche auf den geforderten technischen Anwendungen *QGIS MapServer* und *OpenLayers* beruht und die Ebenen der Naturgefahrenhinweiskarte zeigt, kann man sagen, dass das Hauptziel erreicht ist. Alle Komponenten server- wie clientseitig sind installiert, die nötigen Interaktionen dazwischen realisiert und der gesamte Prozess dokumentiert.

Damit ist auch das persönliche Ziel, den erforderlichen Prozess zur Erstellung einer Webkarte komplett anzuwenden und das nötige Know How zur Marktreife zu bringen, erfüllt. Die Abstützung auf das projektorientierte Vorgehen hat sich bewährt und ist auch in Zukunft in ähnlicher Form wieder anzuwenden. Dadurch ist sichergestellt, dass wichtige Aspekte zum richtigen Zeitpunkt angesprochen und berücksichtigt werden.

Im Vorwort habe ich unter anderem die Schaffung eines Mehrwerts durch erweiterte Nutzung bestehender Datenquellen mit optimierter Symbolisierung angesprochen. Aufgrund der vertieften Auseinandersetzung mit den heute zur Verfügung stehenden OpenSource-Komponenten für Webapplikationen wie beispielsweise *OpenLayers* und *ExtJS* integriert in *MapFish*, möchte ich an dieser Stelle auch auf das grosse Potential der vorhandenen technischen Möglichkeiten zur Realisierung einer Internetkarte hinweisen. Die Herausforderung besteht nun im geschickten Zusammenfügen der vorhandenen Elemente, so dass das Eine in Verbindung mit dem Andern an Bedeutung gewinnt.

Weiter möchte ich die Weiterentwicklung der Symbolisierung der heute üblichen internetbasierten GIS-Systeme zu einer der naturräumlichen Erscheinung entsprechenden Abbildung erwähnen. Somit kann durch das Neukombinieren von bestehenden Datensätzen, aber auch durch Weiterentwicklung der Symbolisierung ein erweiterter Nutzen und damit ein Mehrwert generiert werden. Die in Fülle vorliegenden geografischen Datensätze zu verschiedensten Themen können durch intelligente Symbolisierung einem breiten Publikum verständlich gemacht werden. Neben dem Experten, welcher eine Farbe sofort in Kombination mit dem entsprechenden Sachverhalt stellt, können auch Laien die Karten einfacher lesen und müssen sich nicht zwingend mit komplizierten Beschreibungen auseinander setzen.

QGIS MapServer mit der Fähigkeit *Scalable Vector Graphics SVG* zu Verarbeiten bietet dazu einen interessanten Ansatz. Mit der Erweiterung, dass die Symbolisierung durch den Client per *HTTP GET*-Parameter mitgesendet werden kann, entsteht nutzerseitig eine äusserst dynamische Anwendung. SVG-Symbolsammlungen können aufgebaut und irgendwo zentral zur Verfügung gestellt werden. Dazu ist die vorhandene SVG-Sammlung für beliebige Geodaten weiterzuentwickeln und auch mit Symbolen für Linien- und Punktobjekte zu ergänzen. Die Benutzer können aus dieser Sammlung entsprechende Symbole auswählen. Der Client verwendet dann diese für den Aufruf des *Web Map Service* und entsprechend zur Realisierung der Karte.

Fraglich bleibt natürlich, ob für die jeweiligen Erscheinungen der Naturgefahren durch die Optimierung der Darstellung die bestmöglichen Symbole in dieser Arbeit entwickelt wurden. Die Antwort darauf ist subjektiv und wird von den einzelnen Anwendern der Applikation sicherlich unterschiedlich beantwortet werden. Aber zumindest bestehen nun der Ansatz und die entsprechende technische Umsetzung, um nach Belieben andere Symbole relativ einfach verwenden zu können. Die Funktionalität der Applikation für den Benutzer könnte dahingehend weiterentwickelt werden, dass er neben der Opazität und der Layerreihenfolge auch direkt Einfluss auf die Auswahl der Symbolisierung nehmen könnte.

Die massstabsvariable Darstellung der einzelnen Ebenen könnte ebenfalls optimiert werden. Dazu sind weitere Regeln in der *SLD*-Beschreibung der einzelnen Layer einzuführen und die entsprechenden Massstabsbereiche zu definieren. Auf diese Anwendung wurde in der vorliegenden Arbeit verzichtet, da der *HTTP GET*-Request in seiner Länge begrenzt ist. Dafür müsste der Aufruf mit dem *Simple Object Access Protocol SOAP* realisiert werden.

Offen bleibt durch die Verwendung einzelner Ebenen des *VECTOR25* als Basiskarte die Beschriftung von Ortschaften und Siedlungsgebieten zur Orientierung. Diese müsste in weiterer Arbeit zugefügt werden. Alternativ ermöglicht die vorhandene Lösung, die Landeskarten, welche von einem bestehenden *Web Map Service* bezogen werden, temporär als Hintergrundinformation zu aktivieren.

Alle rechtlichen Aspekte betreffend der Veröffentlichung der verwendeten Daten wären für eine praxisrelevante Applikation ebenfalls zu klären.

7. Quellenangaben

7.1 Literatur

Arbeitsgemeinschaft GEOTEST AG – Geo 7 – Kellerhals + Haefeli AG – GIUB (1997): Gefahrenhinweiskarte des Kantons Bern 1:25‘000, Erläuterungsbericht. Bern: Amt für Wald des Kantons Bern, Abteilung Naturgefahren, Tiefbauamt des Kantons Bern, Energie- und Wasserwirtschaftsamt des Kantons Bern

Hurni, L. (2008): Einführung in die thematische Kartografie. Vorlesung Thematische Kartografie, Herbstsemester 2008, ETH Zürich.

Schalcher, H.R. (2006): Systems Engineering. Vorlesung Systems Engineering, Herbstsemester 2006, ETH Zürich.

Schilcher, M., Donaubauer, A. (2008): Interoperabilität (OGC). Gastvorlesung im Rahmen der Lehrveranstaltung GIS II, Frühlingssemester 2008, ETH Zürich.

Schmidt, G. (2005): Organisatorische Grundbegriffe. Giessen: Verlag Dr. Götz Schmidth, 186 S.

Sutter, J. (2008): Einführung in Internet-GIS - Überblick über die verschiedenen Möglichkeiten, um räumliche Daten im Internet darzustellen. Vorlesung GIS II, Frühjahrssemester 2008, ETH Zürich.

Sutter, J. (2008): Internet & GIS - Erstellen eines dynamischen WebGIS. Vorlesung GIS II, Frühjahrssemester 2008, ETH Zürich.

Ueberschär, N., Winter, A. (2006): Visualisierung von Geodaten mit SVG im Internet. Heidelberg: Herbert Wichmann Verlag, 296 S.

7.2 Internet

Bundesamt für Raumentwicklung ARE (2009): Gefahrenkarte.
http://www.are.admin.ch/themen/raumplanung/00244/00432/00433/index.html?lang=de (Zugriff 11. März 2009)

Enzyklopädie Wikipedia (2009): OpenLayers.
http://de.wikipedia.org/wiki/OpenLayers (Zugriff am 18. April 2009)

Enzyklopädie Wikipedia (2009): PostGIS.
http://de.wikipedia.org/wiki/PostGIS (Zugriff am 25. April 2009)

Hugentobler, M., Iosifescu-Enescu, I (2009): What is QGIS MapServer ?
http://karlinapp.ethz.ch/qgis_wms/about/index.html (Zugriffe 21. März und 16. April 2009)

Open Geospatial Consortium (2007): OpenGIS Styled Layer Descriptor Profile of the Web Map Service Implementation Specification.
http://www.opengeospatial.org/standards/sld/ (Zugriff im März 2009)

Open Geospatial Consortium (2006): OpenGIS Web Map Service (WMS) Implementation Specification.
http://www.opengeospatial.org/standards/wms/ (Zugriff im März 2009)

Openlayers.org (2009): OpenLayers Class Documentation.
http://dev.openlayers.org/releases/OpenLayers-2.7/doc/apidocs/files/OpenLayers-js.html (Zugriffe März bis Mai 2009)

Openlayers.org (2009): What is OpenLayers?
http://docs.openlayers.org/ (Zugriffe am 12. März und 18. April 2009)

Anhang A: Serverseitiger Code

wms_metadata.xml

```
<!-- Service Metadata -->
<Service>
    <Name>WMS</Name>
    <!-- Human-readable title for pick lists -->
    <Title>Naturgefahrenhinweiskarte WMS</Title>
    <!-- Narrative description providing additional information -->
    <Abstract>Bachelorarbeit FS 2009, Andreas Eberle</Abstract>
    <!-- Top-level web address of service or service provider <DCPType>. -->
    <OnlineResource type="simple" href="http://localhost" />
    <!-- Contact information -->
    <ContactInformation>
        <ContactPersonPrimary>
            <ContactPerson>Andreas Eberle</ContactPerson>
        </ContactPersonPrimary>
        <ContactVoiceTelephone></ContactVoiceTelephone>
        <ContactElectronicMailAddress>aeberle@student.ethz.ch</ContactElectronicMailAddress>
    </ContactInformation>
</Service>
```

admin.sld

```
<StyledLayerDescriptor xmlns="http://www.opengis.net/sld">
  <UserLayer xmlns="http://www.opengis.net/sld">
    <Name xmlns="http://www.opengis.net/sld">relief50</Name>
    <Title xmlns="http://www.opengis.net/sld"></Title>
    <Abstract xmlns="http://www.opengis.net/sld"></Abstract>
    <HostedRDS xmlns="http://www.opengis.net/sld" uri="D:\temp\Relief_Andreas\relief50.vrt" epsg="21781"/>
    <UserStyle xmlns="http://www.opengis.net/sld">
      <Name xmlns="http://www.opengis.net/sld">default</Name>
      <RasterSymbolizer>
        <ColorMap interpolation="linear">
          <ColorMapEntry color="#5b6553" quantity="0"/>
          <ColorMapEntry color="#f8fff3" quantity="255"/>
        </ColorMap>
      </RasterSymbolizer>
    </UserStyle>
  </UserLayer>
  <UserLayer xmlns="http://www.opengis.net/sld">
    <Name xmlns="http://www.opengis.net/sld">dol</Name>
    <Title xmlns="http://www.opengis.net/sld"></Title>
    <Abstract xmlns="http://www.opengis.net/sld"></Abstract>
    <HostedVDS xmlns="http://www.opengis.net/sld" providerType="postgres" uri="dbname='gisdb' host=localhost
                          port=5432 user='andebe' password='andebe' table=&quot;ghkdol97&quot; (the_geom)
                          sql="epsg="21781" />
    <UserStyle xmlns="http://www.opengis.net/sld">
      <Name xmlns="http://www.opengis.net/sld">default</Name>
      <FeatureTypeStyle xmlns="http://www.opengis.net/sld">
        <Rule xmlns="http://www.opengis.net/sld">
          <PolygonSymbolizer xmlns="http://www.opengis.net/sld">
            <Stroke xmlns="http://www.opengis.net/sld">
              <CssParameter xmlns="http://www.opengis.net/sld" name="stroke" >#aaaa7f</CssParameter>
              <CssParameter xmlns="http://www.opengis.net/sld" name="stroke-width" >1</CssParameter>
```

```
          </Stroke>
          <Fill xmlns="http://www.opengis.net/sld">
            <CssParameter xmlns="http://www.opengis.net/sld" name="fill" >#ededb2</CssParameter>
          </Fill>
        </PolygonSymbolizer>
      </Rule>
    </FeatureTypeStyle>
  </UserStyle>
</UserLayer>
<UserLayer xmlns="http://www.opengis.net/sld">
  <Name xmlns="http://www.opengis.net/sld">law</Name>
  <Title xmlns="http://www.opengis.net/sld"></Title>
  <Abstract xmlns="http://www.opengis.net/sld"></Abstract>
  <HostedVDS xmlns="http://www.opengis.net/sld" providerType="postgres" uri="dbname='gisdb' host=localhost
                        port=5432 user='andebe' password='andebe' table=&quot;ghklaw97&quot; (the_geom)
                        sql="epsg="21781" />
  <UserStyle xmlns="http://www.opengis.net/sld">
    <Name xmlns="http://www.opengis.net/sld">default</Name>
    <FeatureTypeStyle xmlns="http://www.opengis.net/sld">
      <Rule xmlns="http://www.opengis.net/sld">
        <PolygonSymbolizer xmlns="http://www.opengis.net/sld">
          <Stroke xmlns="http://www.opengis.net/sld">
            <CssParameter xmlns="http://www.opengis.net/sld" name="stroke" >#0055ff</CssParameter>
            <CssParameter xmlns="http://www.opengis.net/sld" name="stroke-width" >1</CssParameter>
          </Stroke>
          <Fill xmlns="http://www.opengis.net/sld">
            <CssParameter xmlns="http://www.opengis.net/sld" name="fill" >#c2fcff</CssParameter>
          </Fill>
        </PolygonSymbolizer>
      </Rule>
    </FeatureTypeStyle>
  </UserStyle>
</UserLayer>
<UserLayer xmlns="http://www.opengis.net/sld">
  <Name xmlns="http://www.opengis.net/sld">mur</Name>
  <Title xmlns="http://www.opengis.net/sld"></Title>
  <Abstract xmlns="http://www.opengis.net/sld"></Abstract>
  <HostedVDS xmlns="http://www.opengis.net/sld" providerType="postgres" uri="dbname='gisdb' host=localhost
                        port=5432 user='andebe' password='andebe' table=&quot;ghkmur97&quot; (the_geom)
                        sql="epsg="21781" />
  <UserStyle xmlns="http://www.opengis.net/sld">
    <Name xmlns="http://www.opengis.net/sld">default</Name>
    <FeatureTypeStyle xmlns="http://www.opengis.net/sld">
      <Rule xmlns="http://www.opengis.net/sld">
        <PolygonSymbolizer xmlns="http://www.opengis.net/sld">
          <Stroke xmlns="http://www.opengis.net/sld">
            <CssParameter xmlns="http://www.opengis.net/sld" name="stroke" >#ff55ff</CssParameter>
            <CssParameter xmlns="http://www.opengis.net/sld" name="stroke-width" >1</CssParameter>
          </Stroke>
          <Fill xmlns="http://www.opengis.net/sld">
            <CssParameter xmlns="http://www.opengis.net/sld" name="fill" >#fcd6ff</CssParameter>
          </Fill>
        </PolygonSymbolizer>
      </Rule>
    </FeatureTypeStyle>
  </UserStyle>
</UserLayer>
```

```
<UserLayer xmlns="http://www.opengis.net/sld">
  <Name xmlns="http://www.opengis.net/sld">ruk</Name>
  <Title xmlns="http://www.opengis.net/sld"></Title>
  <Abstract xmlns="http://www.opengis.net/sld"></Abstract>
  <HostedVDS xmlns="http://www.opengis.net/sld" providerType="postgres" uri="dbname='gisdb' host=localhost
                        port=5432 user='andebe' password='andebe' table=&quot;ghkruk97&quot; (the_geom)
                        sql="epsg="21781" />
  <UserStyle xmlns="http://www.opengis.net/sld">
    <Name xmlns="http://www.opengis.net/sld">default</Name>
    <FeatureTypeStyle xmlns="http://www.opengis.net/sld">
      <Rule xmlns="http://www.opengis.net/sld">
        <PolygonSymbolizer xmlns="http://www.opengis.net/sld">
          <Stroke xmlns="http://www.opengis.net/sld">
            <CssParameter xmlns="http://www.opengis.net/sld" name="stroke" >#aa5500</CssParameter>
            <CssParameter xmlns="http://www.opengis.net/sld" name="stroke-width" >1</CssParameter>
          </Stroke>
          <Fill xmlns="http://www.opengis.net/sld">
            <CssParameter xmlns="http://www.opengis.net/sld" name="fill" >#fde58a</CssParameter>
          </Fill>
        </PolygonSymbolizer>
      </Rule>
    </FeatureTypeStyle>
  </UserStyle>
</UserLayer>
<UserLayer xmlns="http://www.opengis.net/sld">
  <Name xmlns="http://www.opengis.net/sld">rum</Name>
  <Title xmlns="http://www.opengis.net/sld"></Title>
  <Abstract xmlns="http://www.opengis.net/sld"></Abstract>
  <HostedVDS xmlns="http://www.opengis.net/sld" providerType="postgres" uri="dbname='gisdb' host=localhost
                        port=5432 user='andebe' password='andebe' table=&quot;ghkrum97&quot; (the_geom)
                        sql="epsg="21781" />
  <UserStyle xmlns="http://www.opengis.net/sld">
    <Name xmlns="http://www.opengis.net/sld">default</Name>
    <FeatureTypeStyle xmlns="http://www.opengis.net/sld">
      <Rule xmlns="http://www.opengis.net/sld">
        <PolygonSymbolizer xmlns="http://www.opengis.net/sld">
          <Stroke xmlns="http://www.opengis.net/sld">
            <CssParameter xmlns="http://www.opengis.net/sld" name="stroke" >#aa5500</CssParameter>
            <CssParameter xmlns="http://www.opengis.net/sld" name="stroke-width" >1</CssParameter>
          </Stroke>
          <Fill xmlns="http://www.opengis.net/sld">
            <CssParameter xmlns="http://www.opengis.net/sld" name="fill" >#fef6a4</CssParameter>
          </Fill>
        </PolygonSymbolizer>
      </Rule>
    </FeatureTypeStyle>
  </UserStyle>
</UserLayer>
<UserLayer xmlns="http://www.opengis.net/sld">
  <Name xmlns="http://www.opengis.net/sld">ste</Name>
  <Title xmlns="http://www.opengis.net/sld"></Title>
  <Abstract xmlns="http://www.opengis.net/sld"></Abstract>
  <HostedVDS xmlns="http://www.opengis.net/sld" providerType="postgres" uri="dbname='gisdb' host=localhost
                        port=5432 user='andebe' password='andebe' table=&quot;ghkste97&quot; (the_geom)
                        sql="epsg="21781" />
  <UserStyle xmlns="http://www.opengis.net/sld">
    <Name xmlns="http://www.opengis.net/sld">default</Name>
```

```
    <FeatureTypeStyle xmlns="http://www.opengis.net/sld">
      <Rule xmlns="http://www.opengis.net/sld">
        <PolygonSymbolizer xmlns="http://www.opengis.net/sld">
          <Stroke xmlns="http://www.opengis.net/sld">
            <CssParameter xmlns="http://www.opengis.net/sld" name="stroke" >#ff0000</CssParameter>
            <CssParameter xmlns="http://www.opengis.net/sld" name="stroke-width" >1</CssParameter>
          </Stroke>
          <Fill xmlns="http://www.opengis.net/sld">
            <CssParameter xmlns="http://www.opengis.net/sld" name="fill" >#fccfcf</CssParameter>
          </Fill>
        </PolygonSymbolizer>
      </Rule>
    </FeatureTypeStyle>
  </UserStyle>
</UserLayer>
<UserLayer xmlns="http://www.opengis.net/sld">
  <Name xmlns="http://www.opengis.net/sld">syn</Name>
  <Title xmlns="http://www.opengis.net/sld"></Title>
  <Abstract xmlns="http://www.opengis.net/sld"></Abstract>
  <HostedVDS xmlns="http://www.opengis.net/sld" providerType="postgres" uri="dbname='gisdb' host=localhost
                        port=5432 user='andebe' password='andebe' table=&quot;ghksyn97&quot; (the_geom)
                        sql="epsg="21781" />
  <UserStyle xmlns="http://www.opengis.net/sld">
    <Name xmlns="http://www.opengis.net/sld">default</Name>
    <FeatureTypeStyle xmlns="http://www.opengis.net/sld">
      <Rule xmlns="http://www.opengis.net/sld">
        <PolygonSymbolizer xmlns="http://www.opengis.net/sld">
          <Stroke xmlns="http://www.opengis.net/sld">
            <CssParameter xmlns="http://www.opengis.net/sld" name="stroke" >#000000</CssParameter>
            <CssParameter xmlns="http://www.opengis.net/sld" name="stroke-width" >1</CssParameter>
          </Stroke>
          <Fill xmlns="http://www.opengis.net/sld">
            <CssParameter xmlns="http://www.opengis.net/sld" name="fill" >#ff7f7f</CssParameter>
          </Fill>
        </PolygonSymbolizer>
      </Rule>
    </FeatureTypeStyle>
  </UserStyle>
</UserLayer>
<UserLayer xmlns="http://www.opengis.net/sld">
  <Name xmlns="http://www.opengis.net/sld">ueb</Name>
  <Title xmlns="http://www.opengis.net/sld"></Title>
  <Abstract xmlns="http://www.opengis.net/sld"></Abstract>
  <HostedVDS xmlns="http://www.opengis.net/sld" providerType="postgres" uri="dbname='gisdb' host=localhost
                        port=5432 user='andebe' password='andebe' table=&quot;ghkueb97&quot; (the_geom)
                        sql="epsg="21781" />
  <UserStyle xmlns="http://www.opengis.net/sld">
    <Name xmlns="http://www.opengis.net/sld">default</Name>
    <FeatureTypeStyle xmlns="http://www.opengis.net/sld">
      <Rule xmlns="http://www.opengis.net/sld">
        <PolygonSymbolizer xmlns="http://www.opengis.net/sld">
          <Stroke xmlns="http://www.opengis.net/sld">
            <CssParameter xmlns="http://www.opengis.net/sld" name="stroke" >#00aa7f</CssParameter>
            <CssParameter xmlns="http://www.opengis.net/sld" name="stroke-width" >1</CssParameter>
          </Stroke>
          <Fill xmlns="http://www.opengis.net/sld">
            <CssParameter xmlns="http://www.opengis.net/sld" name="fill" >#daffc9</CssParameter>
```

```
          </Fill>
        </PolygonSymbolizer>
      </Rule>
    </FeatureTypeStyle>
  </UserStyle>
</UserLayer>
<UserLayer xmlns="http://www.opengis.net/sld">
  <Name xmlns="http://www.opengis.net/sld">pri25_a</Name>
  <Title xmlns="http://www.opengis.net/sld"></Title>
  <Abstract xmlns="http://www.opengis.net/sld"></Abstract>
  <HostedVDS xmlns="http://www.opengis.net/sld" providerType="postgres" uri="dbname='gisdb' host=localhost
                        port=5432 user='andebe' password='andebe' table=&quot;pri25_a&quot; (the_geom)
                        sql="epsg="21781" />
  <UserStyle xmlns="http://www.opengis.net/sld">
    <Name xmlns="http://www.opengis.net/sld">uniquevalues</Name>
    <FeatureTypeStyle xmlns="http://www.opengis.net/sld">
      <Rule xmlns="http://www.opengis.net/sld">
        <Filter xmlns="http://www.opengis.net/ogc">
          <PropertyIsEqualTo xmlns="http://www.opengis.net/ogc">
            <PropertyName xmlns="http://www.opengis.net/ogc">objectval</PropertyName>
            <Literal xmlns="http://www.opengis.net/ogc">Z_BaumS</Literal>
          </PropertyIsEqualTo>
        </Filter>
        <PolygonSymbolizer xmlns="http://www.opengis.net/sld">
          <Fill xmlns="http://www.opengis.net/sld">
            <CssParameter xmlns="http://www.opengis.net/sld" :name="fill" >#89cb65</CssParameter>
          </Fill>
        </PolygonSymbolizer>
      </Rule>
      <Rule xmlns="http://www.opengis.net/sld">
        <Filter xmlns="http://www.opengis.net/ogc">
          <PropertyIsEqualTo xmlns="http://www.opengis.net/ogc">
            <PropertyName xmlns="http://www.opengis.net/ogc">objectval</PropertyName>
            <Literal xmlns="http://www.opengis.net/ogc">Z_Fluss</Literal>
          </PropertyIsEqualTo>
        </Filter>
        <PolygonSymbolizer xmlns="http://www.opengis.net/sld">
          <Fill xmlns="http://www.opengis.net/sld">
            <CssParameter xmlns="http://www.opengis.net/sld" :name="fill" >#528fff</CssParameter>
          </Fill>
        </PolygonSymbolizer>
      </Rule>
      <Rule xmlns="http://www.opengis.net/sld">
        <Filter xmlns="http://www.opengis.net/ogc">
          <PropertyIsEqualTo xmlns="http://www.opengis.net/ogc">
            <PropertyName xmlns="http://www.opengis.net/ogc">objectval</PropertyName>
            <Literal xmlns="http://www.opengis.net/ogc">Z_GerWa</Literal>
          </PropertyIsEqualTo>
        </Filter>
        <PolygonSymbolizer xmlns="http://www.opengis.net/sld">
          <Fill xmlns="http://www.opengis.net/sld">
            <CssParameter xmlns="http://www.opengis.net/sld" :name="fill" >#e5e5e5</CssParameter>
          </Fill>
        </PolygonSymbolizer>
      </Rule>
      <Rule xmlns="http://www.opengis.net/sld">
        <Filter xmlns="http://www.opengis.net/ogc">
```

```
      <PropertyIsEqualTo xmlns="http://www.opengis.net/ogc">
        <PropertyName xmlns="http://www.opengis.net/ogc">objectval</PropertyName>
        <Literal xmlns="http://www.opengis.net/ogc">Z_GerWaO</Literal>
      </PropertyIsEqualTo>
    </Filter>
    <PolygonSymbolizer xmlns="http://www.opengis.net/sld">
      <Fill xmlns="http://www.opengis.net/sld">
        <CssParameter xmlns="http://www.opengis.net/sld" :name="fill" >#e5e5e5</CssParameter>
      </Fill>
    </PolygonSymbolizer>
  </Rule>
  <Rule xmlns="http://www.opengis.net/sld">
    <Filter xmlns="http://www.opengis.net/ogc">
      <PropertyIsEqualTo xmlns="http://www.opengis.net/ogc">
        <PropertyName xmlns="http://www.opengis.net/ogc">objectval</PropertyName>
        <Literal xmlns="http://www.opengis.net/ogc">Z_ObstAn</Literal>
      </PropertyIsEqualTo>
    </Filter>
    <PolygonSymbolizer xmlns="http://www.opengis.net/sld">
      <Fill xmlns="http://www.opengis.net/sld">
        <CssParameter xmlns="http://www.opengis.net/sld" :name="fill" >#89cb65</CssParameter>
      </Fill>
    </PolygonSymbolizer>
  </Rule>
  <Rule xmlns="http://www.opengis.net/sld">
    <Filter xmlns="http://www.opengis.net/ogc">
      <PropertyIsEqualTo xmlns="http://www.opengis.net/ogc">
        <PropertyName xmlns="http://www.opengis.net/ogc">objectval</PropertyName>
        <Literal xmlns="http://www.opengis.net/ogc">Z_Reben</Literal>
      </PropertyIsEqualTo>
    </Filter>
    <PolygonSymbolizer xmlns="http://www.opengis.net/sld">
      <Fill xmlns="http://www.opengis.net/sld">
        <CssParameter xmlns="http://www.opengis.net/sld" :name="fill" >#aaff7f</CssParameter>
      </Fill>
    </PolygonSymbolizer>
  </Rule>
  <Rule xmlns="http://www.opengis.net/sld">
    <Filter xmlns="http://www.opengis.net/ogc">
      <PropertyIsEqualTo xmlns="http://www.opengis.net/ogc">
        <PropertyName xmlns="http://www.opengis.net/ogc">objectval</PropertyName>
        <Literal xmlns="http://www.opengis.net/ogc">Z_See</Literal>
      </PropertyIsEqualTo>
    </Filter>
    <PolygonSymbolizer xmlns="http://www.opengis.net/sld">
      <Fill xmlns="http://www.opengis.net/sld">
        <CssParameter xmlns="http://www.opengis.net/sld" :name="fill" >#87c1ff</CssParameter>
      </Fill>
    </PolygonSymbolizer>
  </Rule>
  <Rule xmlns="http://www.opengis.net/sld">
    <Filter xmlns="http://www.opengis.net/ogc">
      <PropertyIsEqualTo xmlns="http://www.opengis.net/ogc">
        <PropertyName xmlns="http://www.opengis.net/ogc">objectval</PropertyName>
        <Literal xmlns="http://www.opengis.net/ogc">Z_Siedl</Literal>
      </PropertyIsEqualTo>
    </Filter>
```

```
        <PolygonSymbolizer xmlns="http://www.opengis.net/sld">
          <Fill xmlns="http://www.opengis.net/sld">
            <CssParameter xmlns="http://www.opengis.net/sld" :name="fill" >#ff8787</CssParameter>
          </Fill>
        </PolygonSymbolizer>
      </Rule>
      <Rule xmlns="http://www.opengis.net/sld">
        <Filter xmlns="http://www.opengis.net/ogc">
          <PropertyIsEqualTo xmlns="http://www.opengis.net/ogc">
            <PropertyName xmlns="http://www.opengis.net/ogc">objectval</PropertyName>
            <Literal xmlns="http://www.opengis.net/ogc">Z_Wald</Literal>
          </PropertyIsEqualTo>
        </Filter>
        <PolygonSymbolizer xmlns="http://www.opengis.net/sld">
          <Fill xmlns="http://www.opengis.net/sld">
            <CssParameter xmlns="http://www.opengis.net/sld" :name="fill" >#9bff9b</CssParameter>
          </Fill>
        </PolygonSymbolizer>
      </Rule>
      <Rule xmlns="http://www.opengis.net/sld">
        <Filter xmlns="http://www.opengis.net/ogc">
          <PropertyIsEqualTo xmlns="http://www.opengis.net/ogc">
            <PropertyName xmlns="http://www.opengis.net/ogc">objectval</PropertyName>
            <Literal xmlns="http://www.opengis.net/ogc">Z_WaldOf</Literal>
          </PropertyIsEqualTo>
        </Filter>
        <PolygonSymbolizer xmlns="http://www.opengis.net/sld">
          <Fill xmlns="http://www.opengis.net/sld">
            <CssParameter xmlns="http://www.opengis.net/sld" :name="fill" >#d7ffd7</CssParameter>
          </Fill>
        </PolygonSymbolizer>
      </Rule>
    </FeatureTypeStyle>
  </UserStyle>
</UserLayer>
<UserLayer xmlns="http://www.opengis.net/sld">
  <Name xmlns="http://www.opengis.net/sld">gwn25_l</Name>
  <Title xmlns="http://www.opengis.net/sld"></Title>
  <Abstract xmlns="http://www.opengis.net/sld"></Abstract>
  <HostedVDS xmlns="http://www.opengis.net/sld" providerType="postgres" uri="dbname='gisdb' host=localhost
                        port=5432 user='andebe' password='andebe' table=&quot;gwn25_l&quot; (the_geom)
                        sql="epsg="21781" />
  <UserStyle xmlns="http://www.opengis.net/sld">
    <Name xmlns="http://www.opengis.net/sld">uniquevalues</Name>
    <FeatureTypeStyle xmlns="http://www.opengis.net/sld">
      <Rule xmlns="http://www.opengis.net/sld">
        <Filter xmlns="http://www.opengis.net/ogc">
          <PropertyIsEqualTo xmlns="http://www.opengis.net/ogc">
            <PropertyName xmlns="http://www.opengis.net/ogc">objectval</PropertyName>
            <Literal xmlns="http://www.opengis.net/ogc">Bach</Literal>
          </PropertyIsEqualTo>
        </Filter>
        <LineSymbolizer xmlns="http://www.opengis.net/sld">
          <Stroke xmlns="http://www.opengis.net/sld">
            <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke" >#0000ff</CssParameter>
            <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke-width" >1</CssParameter>
          </Stroke>
```

```
  </LineSymbolizer>
</Rule>
<Rule xmlns="http://www.opengis.net/sld">
  <Filter xmlns="http://www.opengis.net/ogc">
    <PropertyIsEqualTo xmlns="http://www.opengis.net/ogc">
      <PropertyName xmlns="http://www.opengis.net/ogc">objectval</PropertyName>
      <Literal xmlns="http://www.opengis.net/ogc">Bachachs</Literal>
    </PropertyIsEqualTo>
  </Filter>
  <LineSymbolizer xmlns="http://www.opengis.net/sld">
    <Stroke xmlns="http://www.opengis.net/sld">
      <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke" >#0000ff</CssParameter>
      <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke-width" >1</CssParameter>
    </Stroke>
  </LineSymbolizer>
</Rule>
<Rule xmlns="http://www.opengis.net/sld">
  <Filter xmlns="http://www.opengis.net/ogc">
    <PropertyIsEqualTo xmlns="http://www.opengis.net/ogc">
      <PropertyName xmlns="http://www.opengis.net/ogc">objectval</PropertyName>
      <Literal xmlns="http://www.opengis.net/ogc">Bisse</Literal>
    </PropertyIsEqualTo>
  </Filter>
  <LineSymbolizer xmlns="http://www.opengis.net/sld">
    <Stroke xmlns="http://www.opengis.net/sld">
      <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke" >#00aaff</CssParameter>
      <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke-width" >1</CssParameter>
    </Stroke>
  </LineSymbolizer>
</Rule>
<Rule xmlns="http://www.opengis.net/sld">
  <Filter xmlns="http://www.opengis.net/ogc">
    <PropertyIsEqualTo xmlns="http://www.opengis.net/ogc">
      <PropertyName xmlns="http://www.opengis.net/ogc">objectval</PropertyName>
      <Literal xmlns="http://www.opengis.net/ogc">Fluss</Literal>
    </PropertyIsEqualTo>
  </Filter>
  <LineSymbolizer xmlns="http://www.opengis.net/sld">
    <Stroke xmlns="http://www.opengis.net/sld">
      <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke" >#0000ff</CssParameter>
      <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke-width" >2</CssParameter>
    </Stroke>
  </LineSymbolizer>
</Rule>
<Rule xmlns="http://www.opengis.net/sld">
  <Filter xmlns="http://www.opengis.net/ogc">
    <PropertyIsEqualTo xmlns="http://www.opengis.net/ogc">
      <PropertyName xmlns="http://www.opengis.net/ogc">objectval</PropertyName>
      <Literal xmlns="http://www.opengis.net/ogc">Kanal</Literal>
    </PropertyIsEqualTo>
  </Filter>
  <LineSymbolizer xmlns="http://www.opengis.net/sld">
    <Stroke xmlns="http://www.opengis.net/sld">
      <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke" >#0000ff</CssParameter>
      <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke-width" >3</CssParameter>
    </Stroke>
  </LineSymbolizer>
```

```
      </Rule>
      <Rule xmlns="http://www.opengis.net/sld">
        <Filter xmlns="http://www.opengis.net/ogc">
          <PropertyIsEqualTo xmlns="http://www.opengis.net/ogc">
            <PropertyName xmlns="http://www.opengis.net/ogc">objectval</PropertyName>
            <Literal xmlns="http://www.opengis.net/ogc">See</Literal>
          </PropertyIsEqualTo>
        </Filter>
        <LineSymbolizer xmlns="http://www.opengis.net/sld">
          <Stroke xmlns="http://www.opengis.net/sld">
            <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke" >#0000ff</CssParameter>
            <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke-width" >2</CssParameter>
          </Stroke>
        </LineSymbolizer>
      </Rule>
      <Rule xmlns="http://www.opengis.net/sld">
        <Filter xmlns="http://www.opengis.net/ogc">
          <PropertyIsEqualTo xmlns="http://www.opengis.net/ogc">
            <PropertyName xmlns="http://www.opengis.net/ogc">objectval</PropertyName>
            <Literal xmlns="http://www.opengis.net/ogc">Seeinsel</Literal>
          </PropertyIsEqualTo>
        </Filter>
        <LineSymbolizer xmlns="http://www.opengis.net/sld">
          <Stroke xmlns="http://www.opengis.net/sld">
            <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke" >#0000ff</CssParameter>
            <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke-width" >2</CssParameter>
          </Stroke>
        </LineSymbolizer>
      </Rule>
    </FeatureTypeStyle>
  </UserStyle>
</UserLayer>
<UserLayer xmlns="http://www.opengis.net/sld">
  <Name xmlns="http://www.opengis.net/sld">str25_l</Name>
  <Title xmlns="http://www.opengis.net/sld"></Title>
  <Abstract xmlns="http://www.opengis.net/sld"></Abstract>
  <HostedVDS xmlns="http://www.opengis.net/sld" providerType="postgres" uri="dbname='gisdb' host=localhost
                        port=5432 user='andebe' password='andebe' table=&quot;str25_l&quot; (the_geom)
                        sql="epsg="21781" />
  <UserStyle xmlns="http://www.opengis.net/sld">
    <Name xmlns="http://www.opengis.net/sld">uniquevalues</Name>
    <FeatureTypeStyle xmlns="http://www.opengis.net/sld">
      <Rule xmlns="http://www.opengis.net/sld">
        <Filter xmlns="http://www.opengis.net/ogc">
          <PropertyIsEqualTo xmlns="http://www.opengis.net/ogc">
            <PropertyName xmlns="http://www.opengis.net/ogc">objectval</PropertyName>
            <Literal xmlns="http://www.opengis.net/ogc">1_Klass</Literal>
          </PropertyIsEqualTo>
        </Filter>
        <LineSymbolizer xmlns="http://www.opengis.net/sld">
          <Stroke xmlns="http://www.opengis.net/sld">
            <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke" >#000000</CssParameter>
            <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke-width" >2</CssParameter>
          </Stroke>
        </LineSymbolizer>
      </Rule>
      <Rule xmlns="http://www.opengis.net/sld">
```

```
  <Filter xmlns="http://www.opengis.net/ogc">
    <PropertyIsEqualTo xmlns="http://www.opengis.net/ogc">
      <PropertyName xmlns="http://www.opengis.net/ogc">objectval</PropertyName>
      <Literal xmlns="http://www.opengis.net/ogc">2_Klass</Literal>
    </PropertyIsEqualTo>
  </Filter>
  <LineSymbolizer xmlns="http://www.opengis.net/sld">
    <Stroke xmlns="http://www.opengis.net/sld">
      <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke" >#000000</CssParameter>
      <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke-width" >1</CssParameter>
    </Stroke>
  </LineSymbolizer>
</Rule>
<Rule xmlns="http://www.opengis.net/sld">
  <Filter xmlns="http://www.opengis.net/ogc">
    <PropertyIsEqualTo xmlns="http://www.opengis.net/ogc">
      <PropertyName xmlns="http://www.opengis.net/ogc">objectval</PropertyName>
      <Literal xmlns="http://www.opengis.net/ogc">3_Klass</Literal>
    </PropertyIsEqualTo>
  </Filter>
  <LineSymbolizer xmlns="http://www.opengis.net/sld">
    <Stroke xmlns="http://www.opengis.net/sld">
      <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke" >#000000</CssParameter>
      <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke-width" >1</CssParameter>
    </Stroke>
  </LineSymbolizer>
</Rule>
<Rule xmlns="http://www.opengis.net/sld">
  <Filter xmlns="http://www.opengis.net/ogc">
    <PropertyIsEqualTo xmlns="http://www.opengis.net/ogc">
      <PropertyName xmlns="http://www.opengis.net/ogc">objectval</PropertyName>
      <Literal xmlns="http://www.opengis.net/ogc">A_Zufahrt</Literal>
    </PropertyIsEqualTo>
  </Filter>
  <LineSymbolizer xmlns="http://www.opengis.net/sld">
    <Stroke xmlns="http://www.opengis.net/sld">
      <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke" >#000000</CssParameter>
      <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke-width" >1</CssParameter>
    </Stroke>
  </LineSymbolizer>
</Rule>
<Rule xmlns="http://www.opengis.net/sld">
  <Filter xmlns="http://www.opengis.net/ogc">
    <PropertyIsEqualTo xmlns="http://www.opengis.net/ogc">
      <PropertyName xmlns="http://www.opengis.net/ogc">objectval</PropertyName>
      <Literal xmlns="http://www.opengis.net/ogc">Autob_Ri</Literal>
    </PropertyIsEqualTo>
  </Filter>
  <LineSymbolizer xmlns="http://www.opengis.net/sld">
    <Stroke xmlns="http://www.opengis.net/sld">
      <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke" >#000000</CssParameter>
      <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke-width" >2</CssParameter>
    </Stroke>
  </LineSymbolizer>
</Rule>
<Rule xmlns="http://www.opengis.net/sld">
  <Filter xmlns="http://www.opengis.net/ogc">
```

```
      <PropertyIsEqualTo xmlns="http://www.opengis.net/ogc">
        <PropertyName xmlns="http://www.opengis.net/ogc">objectval</PropertyName>
        <Literal xmlns="http://www.opengis.net/ogc">Autobahn</Literal>
      </PropertyIsEqualTo>
    </Filter>
    <LineSymbolizer xmlns="http://www.opengis.net/sld">
      <Stroke xmlns="http://www.opengis.net/sld">
        <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke" >#000000</CssParameter>
        <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke-width" >2</CssParameter>
      </Stroke>
    </LineSymbolizer>
  </Rule>
  <Rule xmlns="http://www.opengis.net/sld">
    <Filter xmlns="http://www.opengis.net/ogc">
      <PropertyIsEqualTo xmlns="http://www.opengis.net/ogc">
        <PropertyName xmlns="http://www.opengis.net/ogc">objectval</PropertyName>
        <Literal xmlns="http://www.opengis.net/ogc">Autostr</Literal>
      </PropertyIsEqualTo>
    </Filter>
    <LineSymbolizer xmlns="http://www.opengis.net/sld">
      <Stroke xmlns="http://www.opengis.net/sld">
        <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke" >#000000</CssParameter>
        <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke-width" >2</CssParameter>
      </Stroke>
    </LineSymbolizer>
  </Rule>
  <Rule xmlns="http://www.opengis.net/sld">
    <Filter xmlns="http://www.opengis.net/ogc">
      <PropertyIsEqualTo xmlns="http://www.opengis.net/ogc">
        <PropertyName xmlns="http://www.opengis.net/ogc">objectval</PropertyName>
        <Literal xmlns="http://www.opengis.net/ogc">BrueckLe</Literal>
      </PropertyIsEqualTo>
    </Filter>
    <LineSymbolizer xmlns="http://www.opengis.net/sld">
      <Stroke xmlns="http://www.opengis.net/sld">
        <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke" >#000000</CssParameter>
        <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke-width" >1</CssParameter>
      </Stroke>
    </LineSymbolizer>
  </Rule>
  <Rule xmlns="http://www.opengis.net/sld">
    <Filter xmlns="http://www.opengis.net/ogc">
      <PropertyIsEqualTo xmlns="http://www.opengis.net/ogc">
        <PropertyName xmlns="http://www.opengis.net/ogc">objectval</PropertyName>
        <Literal xmlns="http://www.opengis.net/ogc">Ein_Ausf</Literal>
      </PropertyIsEqualTo>
    </Filter>
    <LineSymbolizer xmlns="http://www.opengis.net/sld">
      <Stroke xmlns="http://www.opengis.net/sld">
        <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke" >#000000</CssParameter>
        <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke-width" >1</CssParameter>
      </Stroke>
    </LineSymbolizer>
  </Rule>
  <Rule xmlns="http://www.opengis.net/sld">
    <Filter xmlns="http://www.opengis.net/ogc">
      <PropertyIsEqualTo xmlns="http://www.opengis.net/ogc">
```

```
            <PropertyName xmlns="http://www.opengis.net/ogc">objectval</PropertyName>
            <Literal xmlns="http://www.opengis.net/ogc">Parkweg</Literal>
          </PropertyIsEqualTo>
        </Filter>
        <LineSymbolizer xmlns="http://www.opengis.net/sld">
          <Stroke xmlns="http://www.opengis.net/sld">
            <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke" >#000000</CssParameter>
            <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke-width" >0</CssParameter>
          </Stroke>
        </LineSymbolizer>
      </Rule>
      <Rule xmlns="http://www.opengis.net/sld">
        <Filter xmlns="http://www.opengis.net/ogc">
          <PropertyIsEqualTo xmlns="http://www.opengis.net/ogc">
            <PropertyName xmlns="http://www.opengis.net/ogc">objectval</PropertyName>
            <Literal xmlns="http://www.opengis.net/ogc">Q_Klass</Literal>
          </PropertyIsEqualTo>
        </Filter>
        <LineSymbolizer xmlns="http://www.opengis.net/sld">
          <Stroke xmlns="http://www.opengis.net/sld">
            <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke" >#000000</CssParameter>
            <CssParameter xmlns="http://www.opengis.net/sld" :name="stroke-width" >1</CssParameter>
          </Stroke>
        </LineSymbolizer>
      </Rule>
    </FeatureTypeStyle>
  </UserStyle>
</UserLayer>
</StyledLayerDescriptor>
```

Anhang B: Clientseitiger Code

Index.html

```
<html>
  <head>
    <title>Naturgefahrenhinweiskarte Kanton Bern</title>

    <link rel="stylesheet" type="text/css" href="mfbase/ext/resources/css/ext-all.css" />
    <link rel="stylesheet" type="text/css" href="css/style.css" />

    <script type="text/javascript" src="mfbase/openlayers/lib/OpenLayers.js"></script>
    <script type="text/javascript" src="mfbase/ext/adapter/ext/ext-base.js"></script>
    <script type="text/javascript" src="mfbase/ext/ext-all-debug.js"></script>
    <script type="text/javascript" src="mfbase/mapfish/MapFish.js"></script>

    <script type="text/javascript" src="js/Controls.js"></script>
    <script type="text/javascript" src="js/Layers.js"></script>
    <script type="text/javascript" src="js/LayerTree.js"></script>
    <script type="text/javascript" src="js/Map.js"></script>
    <script type="text/javascript" src="js/Toolbar.js"></script>
    <script type="text/javascript" src="js/Viewport.js"></script>

    <script type="text/javascript">
        Ext.onReady(function() {
            Map.createMap('mymap', {
                    projection: new OpenLayers.Projection("EPSG:21781"),
                    units: "m",
                    maxExtent: new OpenLayers.Bounds(519000, 64000, 719000, 264000),
                    controls: [],
                    scales: [250000,125000,100000,75000,50000,25000,12500]
                }
            );

            Controls.addMapControls();

            Layers.createWmsLayer(
                "Swisstopo",
                "http://www.geofoto.ch/cgi/mapserv?",
                {map: '/home/an/mapserverdata/relief_u_pk_kombi/relief_u_pk_kombi_jpeg.map', layers:
                        'pk_kombi', format: 'image/jpeg', SRS: 'EPSG:21781' },
                {isBaseLayer: true, attribution: "Pixelkarte (c) Swisstopo", buffer: 0, visibility: false}
            );

            var serv = "http://localhost/cgi-bin/qgis_map_server/qgis_map_serv.cgi?";
            var servkar = "http://karlinapp.ethz.ch/fcgi-bin/qgis_map_server/webgisbern/qgis_map_serv.fcgi?";

            Layers.createWmsLayer(
                "relief50",
                servkar,
                {format: "image/jpeg", layers: "relief50", styles: "default"},
                {isBaseLayer: false, opacity: 0.5, buffer: 0}
            );

            Layers.createWmsLayer(
                "pri25_a",
```

```
                serv,
                {transparent: "true", format: "image/png", layers: "pri25_a", styles: "uniquevalues"},
                {isBaseLayer: true, opacity: 1, buffer: 0, visibility: false}
            );

            Layers.createWmsLayer(
                "str25_l",
                serv,
                {transparent: "true", format: "image/png", layers: "str25_l", styles: "uniquevalues"},
                {isBaseLayer: false, opacity: 0.4, buffer: 0, visibility: false}
            );

            Layers.createWmsLayer(
                "gwn25_l",
                serv,
                {transparent: "true", format: "image/png", layers: "gwn25_l", styles: "uniquevalues"},
                {isBaseLayer: false, opacity: 0.4, buffer: 0, visibility: false}
            );

            Layers.createLayersFromSLD(
                "svg/sldLayers.xml",
                serv,
                {isBaseLayer: false, opacity: 0.8, buffer: 0, visibility: false}
            );


            this.vector = Layers.createVectorLayer();
            // vector layer is now accessible via toolbarExample.app.vector


            Toolbar.createToolbar();
            Viewport.createViewport();
            Toolbar.setToolbarContent();

            LayerTree.createLayerTree();

            map.setCenter(new OpenLayers.LonLat(619000, 164000),5);
        });
    </script>
</head>

<body>
    <div id="north"><b>Naturgefahrenhinweiskarte Kanton Bern</b><br/>
    ETH Zürich, Bachelorarbeit FS2009, Andreas Eberle</div>

    <div id="tree"></div>
    <div id="legend"></div>
    <div id="overview"></div>
    <div id="mousePos"></div>
    <div id="mymap"></div>

    <form id="scaleForm" >
    <!--<input id='scaleInput' type='text' name='scale' value='10000.0' size='10' maxlength='9'
                          onChange='Toolbar.zoomToScale(); return false;'/>-->
    <select id="scaleInput" name="scale" size="1" onChange='Toolbar.zoomToScale(); return false;'>
      <option value="12500">1:12'500</option>
      <option value="25000">1:25'000</option>
```

```
        <option value="50000">1:50'000</option>
        <option value="75000">1:75'000</option>
        <option value="100000">1:100'000</option>
        <option value="125000">1:125'000</option>
        <option value="250000">1:250'000</option>
      </select>
      </form>
    </body>

</html>
```

Map.js

```
/*
 * JavaScript-Code realisiert im Rahmen der Bachelorarbeit
 * Visualisierung von Naturgefahrendaten mit QGIS mapserver und OpenLayers
 *
 * ETH Zuerich, Institut fuer Kartografie, Fruehlingssemester 2009
 * Autor: Andreas Eberle, aeberle@student.ethz.ch
 * Mai 2009
 *
 * Erzeugen des Map-Objekts mit entsprechenden Parametern
 *
 */

var Map = {
    createMap: function(name,params) {
        map = new OpenLayers.Map(name,params);
        map.events.register('moveend', this, Toolbar.updateScale);
    }
}
```

Controls.js

```
/*
 * JavaScript-Code realisiert im Rahmen der Bachelorarbeit
 * Visualisierung von Naturgefahrendaten mit QGIS mapserver und OpenLayers
 *
 * ETH Zuerich, Institut fuer Kartografie, Fruehlingssemester 2009
 * Autor: Andreas Eberle, aeberle@student.ethz.ch
 * Mai 2009
 *
 * Controls zur Manipulation der Karte oder zum Anzeigen
 * userspezifischer Informationen / primaeres Interface für Map-Interaktionen.
 *
 */

var Controls = {
    addMapControls: function() {

        // Integration einer Uebersichtskarte
        var overviewMap = new OpenLayers.Control.OverviewMap({
            div: OpenLayers.Util.getElement('overview'),
            size: new OpenLayers.Size(200, 128),
            mapOptions: {
                projection: new OpenLayers.Projection("EPSG:21781"),
                units: "m",
                maxExtent: new OpenLayers.Bounds(519000, 64000, 719000, 264000),
                numZoomLevels: 4
            }
        });
        map.addControl(overviewMap);

        // Manipulation des Kartenausschnitts
        var navControl = new OpenLayers.Control.Navigation({
            type: OpenLayers.Control.TYPE_TOGGLE,
            zoomWheelEnabled: true
        });
        map.addControl(navControl);
        navControl.activate();

        // Abfrage der Mausposition auf dem Objekt Map
        var mousePositionControl = new OpenLayers.Control.MousePosition({
            div: OpenLayers.Util.getElement('mousePos'),
            prefix: 'E: ',
            separator: '  N: ',
            numDigits: 0
        });
        map.addControl(mousePositionControl);
    }
}
```

Layers.js

```
/*
 * JavaScript-Code realisiert im Rahmen der Bachelorarbeit
 * Visualisierung von Naturgefahrendaten mit QGIS mapserver und OpenLayers
 *
 * ETH Zuerich, Institut fuer Kartografie, Fruehlingssemester 2009
 * Autor: Andreas Eberle, aeberle@student.ethz.ch
 * Mai 2009
 *
 * Layers zum Einbinden von Datenquellen verschiedenster Herkunft /
 * Verarbeitung und Einbindung einer Liste von Layers in SLD-Beschreibung
 *
 */

var Layers = {

    // Verarbeiten eines XML-Dokuments
    parseXML: function(docPath) {
        try {
            //Internet Explorer
            xmlDocument=new ActiveXObject("Microsoft.XMLDOM");
        } catch(e) {
            try {
                //Firefox, Mozilla, Opera, etc.
                xmlDocument=document.implementation.createDocument("","",null);
            } catch(e) {
                alert(e.message);
                return;
            }
        }
        xmlDocument.async=false;
        xmlDocument.load(docPath);
        return xmlDocument;
    },

    // Konvertieren XML zu String
    xmlToString: function(xmlDoc) {
        try {
            var xmlString = (new XMLSerializer()).serializeToString(xmlDoc);
        } catch(e) {
            try {
                var xmlString = xmlDoc.xml;
            } catch(e) {
                alert("XMLSerializer not supported");
            }
        }
        return xmlString;
    },

    // Einbinden von WMS-Ebenen einer SLD-Auflistung
    createLayersFromSLD: function(sld, url, options) {
        var sldLayers = this.parseXML(sld);
        var sldLayer=sldLayers.getElementsByTagName("StyledLayerDescriptor");
        for (k=0;k<sldLayer.length;k++) {
            var svg = sldLayer[k].getElementsByTagName("svg");
            for (l=0;l<svg.length;l++) {
```

```
                var svgDocument =
                        this.parseXML("svg/"+sldLayer[k].getElementsByTagName("svg")[l].getAttribute("name
                        "));
                var x=svgDocument.getElementsByTagName("rect");
                var anz = x.length;
                for (i=0;i<anz;i++) {
                    sldLayer[k].getElementsByTagName("svg")[l].appendChild(x[0]);
                }
                var x=svgDocument.getElementsByTagName("path");
                var anz = x.length;
                for (i=0;i<anz;i++) {
                    sldLayer[k].getElementsByTagName("svg")[l].appendChild(x[0]);
                }
            }
            var name = sldLayer[k].getElementsByTagName("Name")[0].firstChild.data;
            var params = {transparent: "true", format: "image/png", SLD_BODY:
                        this.xmlToString(sldLayer[k]).replace(/  /g, "").replace(/\n/g,"")};
            map.addLayer(new OpenLayers.Layer.WMS.Untiled(name, url, params, options));
        }
    },

    // Einbinden eines Web Map Service
    createWmsLayer: function(name, url, params, options) {
        map.addLayer(new OpenLayers.Layer.WMS.Untiled(name, url, params, options));
    },

    // Einbinden von GoogleMaps
    createGoogleLayer: function(name,param) {
        var googleLayer = new OpenLayers.Layer.Google(name,param);
        googleLayer.projection = 'EPSG:3785';
        map.addLayer(googleLayer);
    },

    // Erzeugen eines VectorLayers
    createVectorLayer: function() {
        vectorLayer = new OpenLayers.Layer.Vector("vector", {
            displayInLayerSwitcher: false
        });
        map.addLayer(vectorLayer);

        vectorLayer.events.register('featureadded', vectorLayer, function(){
            var l = this.features.length;
            var msg = (l==1) ? 'There is one feature on map' : 'There are now ' + l + ' features on map';
            toolbarExample.app.setStatus(msg);
        });

        return vectorLayer;
    }
}
```

Toolbar.js

```
/*
 * JavaScript-Code realisiert im Rahmen der Bachelorarbeit
 * Visualisierung von Naturgefahrendaten mit QGIS mapserver und OpenLayers
 *
 * ETH Zuerich, Institut fuer Kartografie, Fruehlingssemester 2009
 * Autor: Andreas Eberle, aeberle@student.ethz.ch
 * Mai 2009
 *
 * Toolbar mit Buttons zur Steuerung der Karte
 *
 */

var Toolbar = {

    // Erzeugen der Werkzeugleiste
    createToolbar: function() {
        toolbar = new mapfish.widgets.toolbar.Toolbar({
            map: map,
            configurable: false
        });
        toolbar.autoHeight = false;
        toolbar.height = 25;
    },

    // Zufuegen eines Trennstrich
    addSeparator: function() {
        toolbar.add(new Ext.Toolbar.Spacer());
        toolbar.add(new Ext.Toolbar.Separator());
        toolbar.add(new Ext.Toolbar.Spacer());
    },

    // Zoomen entsprechend der Listenauswahl
    zoomToScale: function() {
        var element = document.getElementById('scaleInput');
        var scale = element.value - 100;
        map.zoomToScale(1.0/scale);
    },
    updateScale: function() {
        var element = document.getElementById('scaleInput');
        element.value = Math.round(map.getScale());
    },

    // Aufbau der Werkzeugliste
    setToolbarContent: function() {

        // Button Verleinern auf ganze Karte
        toolbar.addControl(
            new OpenLayers.Control.ZoomToMaxExtent({
                map: map,
                title: 'Verleinern auf ganze Karte'
            }), {
                iconCls: 'zoomfull',
                toggleGroup: 'map'
            }
        );
```

```
this.addSeparator();

// Button Vergroessern
toolbar.addControl(
    new OpenLayers.Control.ZoomIn({
        title: 'Vergr&ouml;ssern'
    }), {
        iconCls: 'zoomin',
        toggleGroup: 'map'
    }
);

// Button Verleinern
toolbar.addControl(
    new OpenLayers.Control.ZoomOut({
        title: 'Verkleinern'
    }), {
        iconCls: 'zoomout',
        toggleGroup: 'map'
    }
);

// Button Vergroessern auf Rechteck
toolbar.addControl(
    new OpenLayers.Control.ZoomBox({
        title: 'Vergr&ouml;ssern auf Rechteck'
    }), {
        iconCls: 'zoombox',
        toggleGroup: 'map'
    }
);

// Button Verschieben
toolbar.addControl(
    new OpenLayers.Control.DragPan({
        isDefault: true,
        title: 'Verschieben'
    }), {
        iconCls: 'pan',
        toggleGroup: 'map'
    }
);
this.addSeparator();

// Auswahlliste der Zoomstufen
toolbar.addElement('scaleForm');
this.addSeparator();

// Erfassen der jeweiligen Kartenausschnitte fuer die History
var nav = new OpenLayers.Control.NavigationHistory();
map.addControl(nav);
nav.activate();
// Zurueck zur letzten Ansicht
var buttonPrevious = new Ext.Toolbar.Button({
    iconCls: 'back',
    tooltip: 'Previous view',
    disabled: true,
```

```
    handler: nav.previous.trigger
});
toolbar.add(buttonPrevious);
// Vorwaerts zur naechsten Ansicht
var buttonNext = new Ext.Toolbar.Button({
    iconCls: 'next',
    tooltip: 'Next view',
    disabled: true,
    handler: nav.next.trigger
});
toolbar.add(buttonNext);

nav.previous.events.register(
    "activate",
    buttonPrevious,
    function() {
        this.setDisabled(false);
    }
);
nav.previous.events.register(
    "deactivate",
    buttonPrevious,
    function() {
        this.setDisabled(true);
    }
);
nav.next.events.register(
    "activate",
    buttonNext,
    function(){
        this.setDisabled(false);
    }
);
nav.next.events.register(
    "deactivate",
    buttonNext,
    function() {
        this.setDisabled(true);
    }
);
this.addSeparator();

// Punkt zeichnen
toolbar.addControl(
    new OpenLayers.Control.DrawFeature(vectorLayer, OpenLayers.Handler.Point, {
        title: 'Punkt zeichnen'
    }), {
        iconCls: 'drawpoint',
        toggleGroup: 'map'
    }
);

// Linie zeichnen
toolbar.addControl(
    new OpenLayers.Control.DrawFeature(vectorLayer, OpenLayers.Handler.Path, {
        title: 'Linie zeichnen'
    }), {
```

```
                iconCls: 'drawline',
                toggleGroup: 'map'
            }
        );

        // Flaeche zeichnen
        toolbar.addControl(
            new OpenLayers.Control.DrawFeature(vectorLayer, OpenLayers.Handler.Polygon, {
                title: 'Polygon zeichnen'
            }), {
                iconCls: 'drawpolygon',
                toggleGroup: 'map'
            }
        );
        this.addSeparator();
        toolbar.add(new Ext.Toolbar.Fill());

        // Mausposition erfassen
        toolbar.addElement('mousePos');
        toolbar.activate();
    }
}
```

Viewport.js

```
/*
 * JavaScript-Code realisiert im Rahmen der Bachelorarbeit
 * Visualisierung von Naturgefahrendaten mit QGIS mapserver und OpenLayers
 *
 * ETH Zuerich, Institut fuer Kartografie, Fruehlingssemester 2009
 * Autor: Andreas Eberle, aeberle@student.ethz.ch
 * Mai 2009
 *
 * Viewport zur Integration der einzelnen Bereiche der Applikation
 * Bestimmung der Eigenschaften der jeweiligen Fensterbereiche
 *
 */

var Viewport = {
    createViewport: function() {
        viewport = new Ext.Viewport({
            layout: 'border',
            items: [
                new Ext.BoxComponent({
                    region: 'north',
                    el: 'north',
                    height: 60,
                    border: false
                }), {
                    region: 'west',
                    title: 'Map Explorer',
                    width: 210,
                    border: true,
                    margins: '5 0 5 5',
                    frame: true,
                    split: true,
                    layout: 'border',
                    collapsible: true,
                    items: [{
                        title: 'Layer Tree',
                        region: 'center',
                        border: true,
                        el: 'tree'
                    },{
                        title: 'Overview',
                        region: 'south',
                        height: 150,
                        collapsible: true,
                        margins: '5 0 0 0',
                        contentEl: 'overview'
                    }]
                },{
                    region: 'center',
                    //title: 'map',
                    layout: 'fit',
                    frame: false,
                    border: true,
                    margins: '5 5 5 0',
                    items: [{
                        xtype: 'mapcomponent',
```

```
                        map: map,
                        tbar: toolbar,
                        border: false
                    }]
                }
            ]
        });
    }
}
```

LayerTree.js

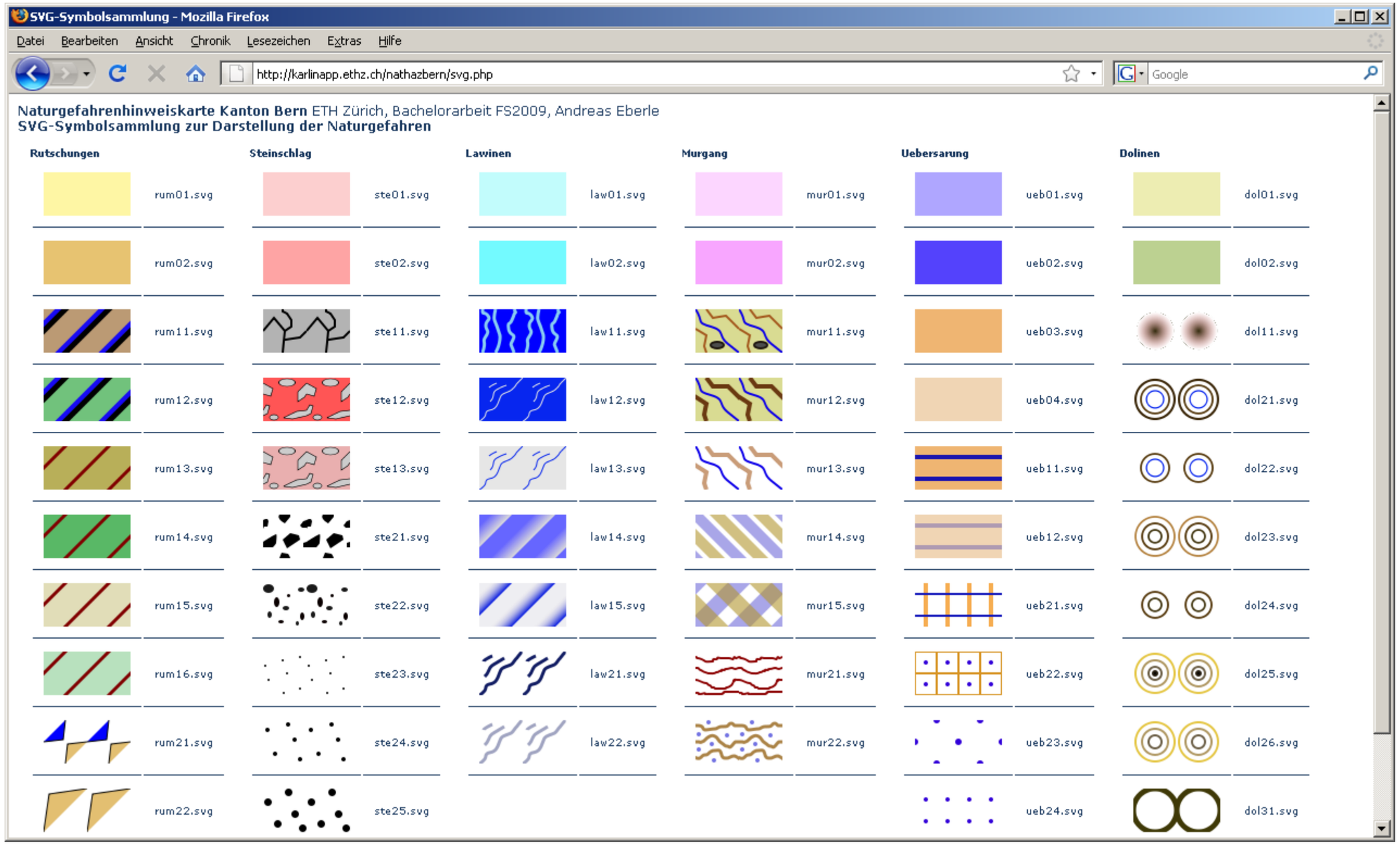
```
/*
 * JavaScript-Code realisiert im Rahmen der Bachelorarbeit
 * Visualisierung von Naturgefahrendaten mit QGIS mapserver und OpenLayers
 *
 * ETH Zuerich, Institut fuer Kartografie, Fruehlingssemester 2009
 * Autor: Andreas Eberle, aeberle@student.ethz.ch
 * Mai 2009
 *
 * Erzeugen des Layerbaums der Kartenapplikation mit Angaben
 * zu entsprechenden Anzeigeeigenschaften
 *
 */

var LayerTree = {
    createLayerTree: function() {

        // Modell mit layerspezifischen Anzeigeeigenschaften
        var model = [{
            text: "Naturgefahrenhinweiskarte",
            expanded:true,
            children:[{
                text: "Prozessbereich aller Gefahren",
                layerNames: ["syn"],
                minScale: 60000,
                checked: false,
                icon: "icons/syn1.png"
            },{
                text: "Naturgefahren",
                expanded:true,
                children:[{
                    text: "Dolinen",
                    layerNames: ["dol"],
                    maxScale: 60000,
                    checked: false,
                    children: [{
                        text: "starke Gefährdung",
                        icon: "icons/dol2.png"
                    },{
                        text: "schwache Gefährdung",
                        icon: "icons/dol1.png"
                    }]
                },{
                    text: "Murgang",
                    layerNames: ["mur"],
                    maxScale: 60000,
                    checked: false,
                    children: [{
                        text: "Schadenpotential",
                        icon: "icons/mur2.png"
                    },{
                        text: "nicht relevanter Prozessbereich",
                        icon: "icons/mur1.png"
                    }]
                },{
                    text: "Übersarung",
```

```
                layerNames: ["ueb"],
                maxScale: 60000,
                checked: false,
                children: [{
                    text: "Schadenpotential",
                    icon: "icons/ueb2.png"
                },{
                    text: "nicht relevanter Prozessbereich",
                    icon: "icons/ueb1.png"
                }]
            },{
                text: "Steinschlag",
                layerNames: ["ste"],
                maxScale: 60000,
                checked: false,
                children: [{
                    text: "Schadenpotential",
                    icon: "icons/ste2.png"
                },{
                    text: "nicht relevanter Prozessbereich",
                    icon: "icons/ste1.png"
                }]
            },{
                text: "Lawinen",
                layerNames: ["law"],
                maxScale: 60000,
                checked: false,
                children: [{
                    text: "Schadenpotential",
                    icon: "icons/law2.png"
                },{
                    text: "nicht relevanter Prozessbereich",
                    icon: "icons/law1.png"
                }]
            },{
                text: "Rutschungen",
                layerNames: ["rum"],
                maxScale: 60000,
                checked: false,
                children: [{
                    text: "Schadenpotential (tiefgründig)",
                    icon: "icons/rum11.png"
                },{
                    text: "nicht relevanter Prozessbereich (tiefgründig)",
                    icon: "icons/rum1.png"
                },{
                    text: "Schadenpotential (mittel- und flachgründig)",
                    icon: "icons/rum12.png"
                },{
                    text: "nicht relevanter Prozessbereich (mittel- und flachgründig)",
                    icon: "icons/rum2.png"
                }]
            }]
        }]
    },{
        text: "Hintergrundkarte",
        expanded: true,
```

```
                children: [{
                    text: "Landeskarte",
                    //icon: "http://www.openlayers.org/favicon.ico",
                    layerNames: ["Swisstopo"],
                    checked: false
                },{
                    text: "Vector25",
                    expanded:true,
                    children:[{
                        text: "Strassen",
                        layerNames: ["str25_l"],
                        maxScale: 40000,
                        checked: false
                    },{
                        text: "Gewaesser",
                        layerNames: ["gwn25_l"],
                        maxScale: 60000,
                        checked: false
                    },{
                        text: "Primaerflaechen",
                        layerNames: ["pri25_a"],
                        maxScale: 60000,
                        checked: true
                    },{
                        text: "Relief",
                        layerNames: ["relief50"],
                        checked: true
                    }]
                }]
            }];

            // Erzeugen des Layerbaums
            var tree = new mapfish.widgets.LayerTree({
                map: map,
                el: "tree",
                model: model,
                enableDD: true,
                ascending: false,
                plugins: [

                            mapfish.widgets.LayerTree.createContextualMenuPlugin(['opacitySlideDirect','zoomTo
                            Extent','remove']),
                    mapfish.widgets.LayerTree.createDisableLayersPlugin(72)
                ]
            });
            tree.render();
        }

    }
```

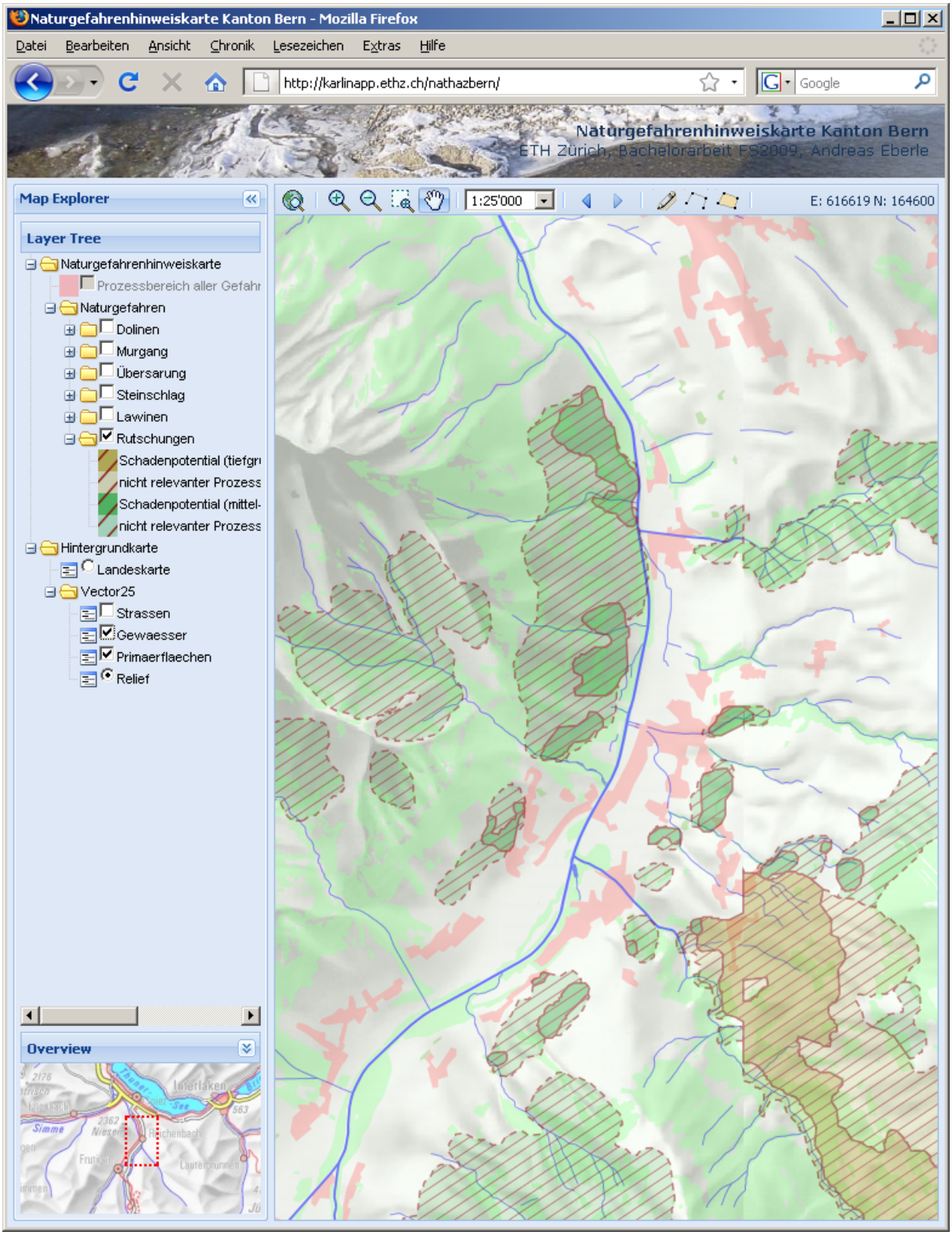
Anhang C: Scalable Vector Graphics SVG – Symbolsammlung

Entwicklung und Realisierung möglicher Symbole zur Darstellung der einzelnen Naturgefahren

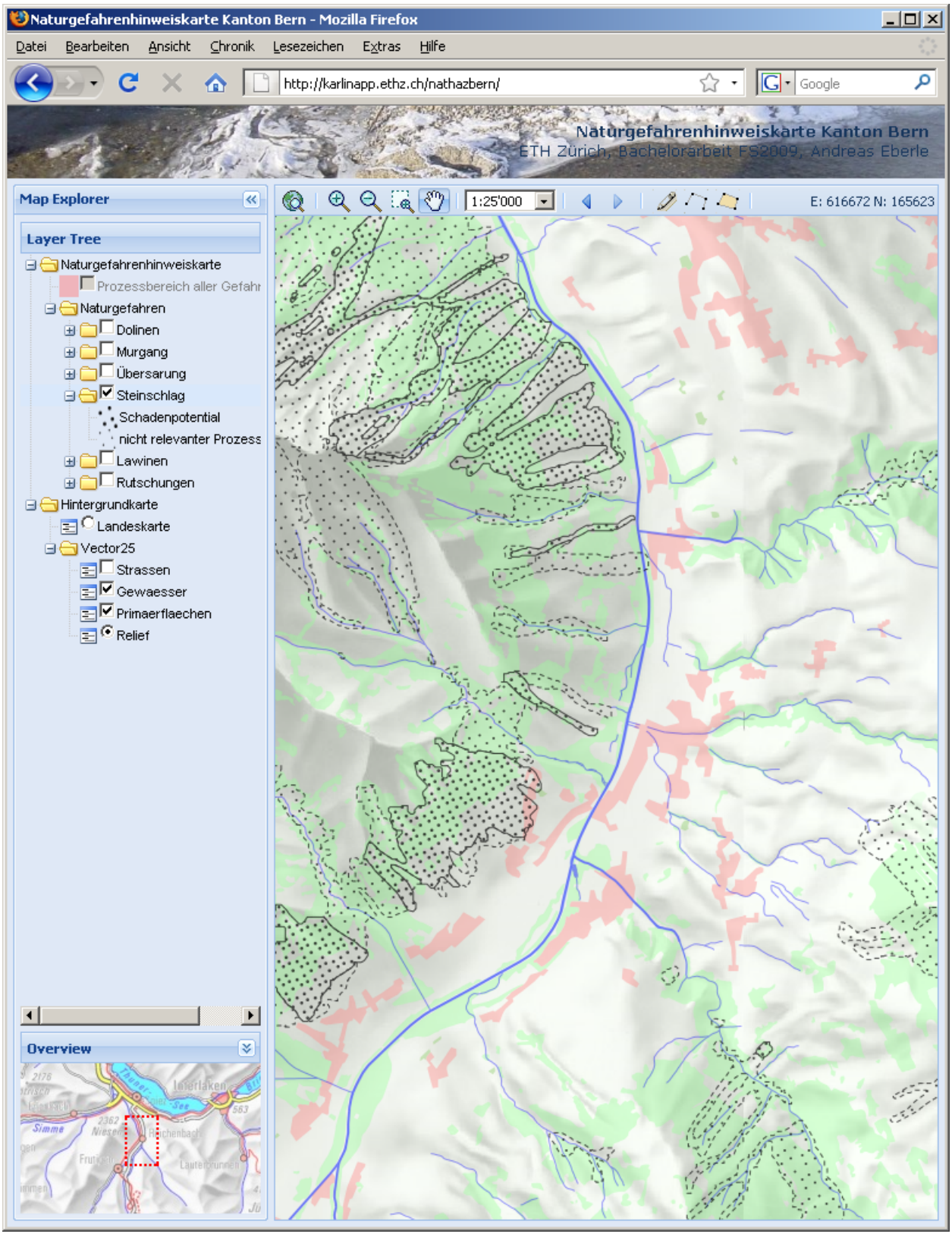


Anhang D: Darstellung der einzelnen Naturgefahren

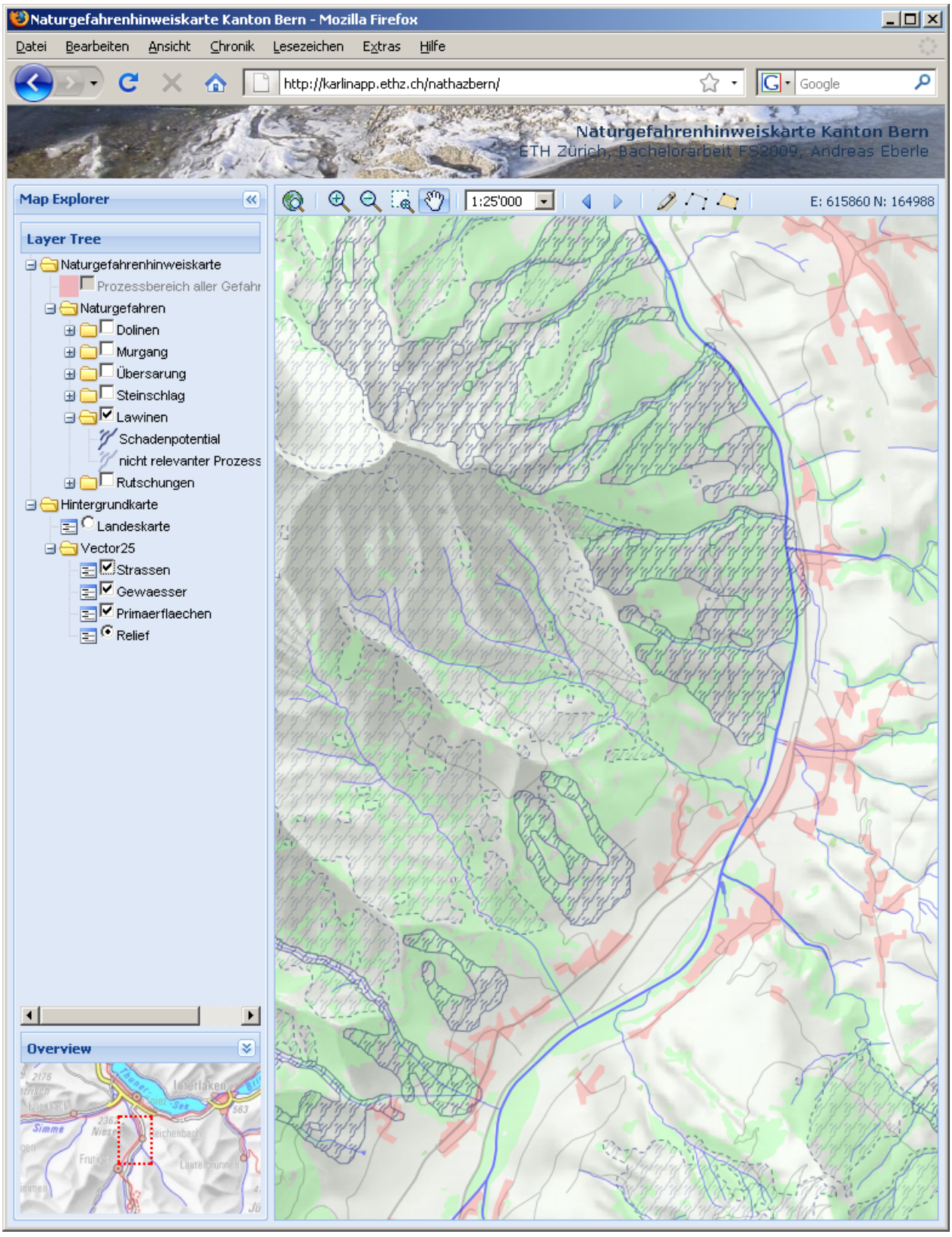
Rutschung



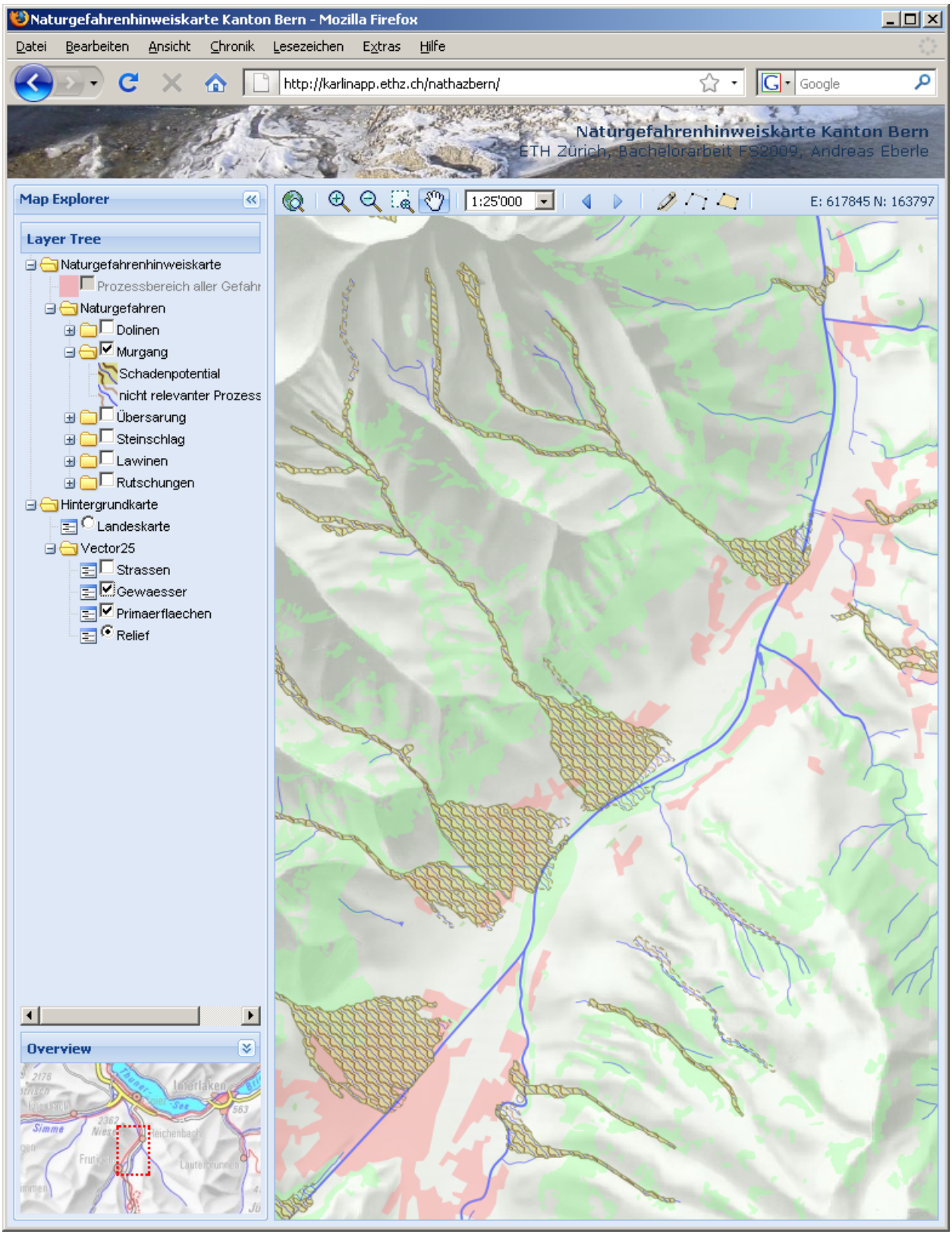
Steinschlag



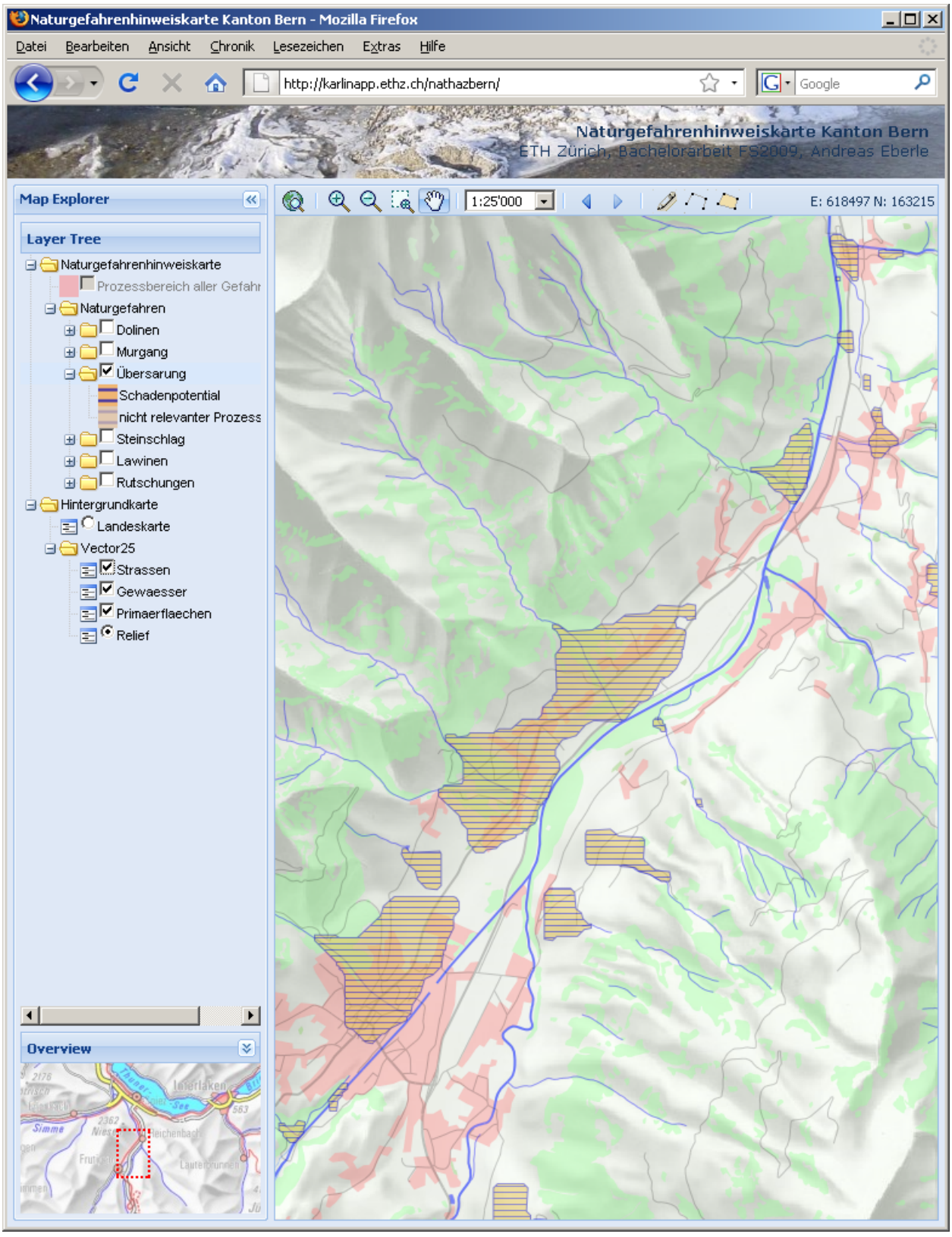
Lawinen


Naturgefahrenhinweiskarte Kanton Bern - Mozilla Firefox
Datei Bearbeiten Ansicht Chronik Lesezeichen Extras Hilfe
http://karlinapp.ethz.ch/nathazbern/
Google
Naturgefahrenhinweiskarte Kanton Bern
ETH Zürich, Bachelorarbeit FS2009, Andreas Eberle
Map Explorer
Layer Tree
Naturgefahrenhinweiskarte
Prozessbereich aller Gefahr
Naturgefahren
Dolinen
Murgang
Übersarung
Steinschlag
Lawinen
Schadenpotential
nicht relevanter Prozess
Rutschungen
Hintergrundkarte
Landeskarte
Vector25
Strassen
Gewaesser
Primaerflaechen
Relief
Overview
1:25'000
E: 615860 N: 164988

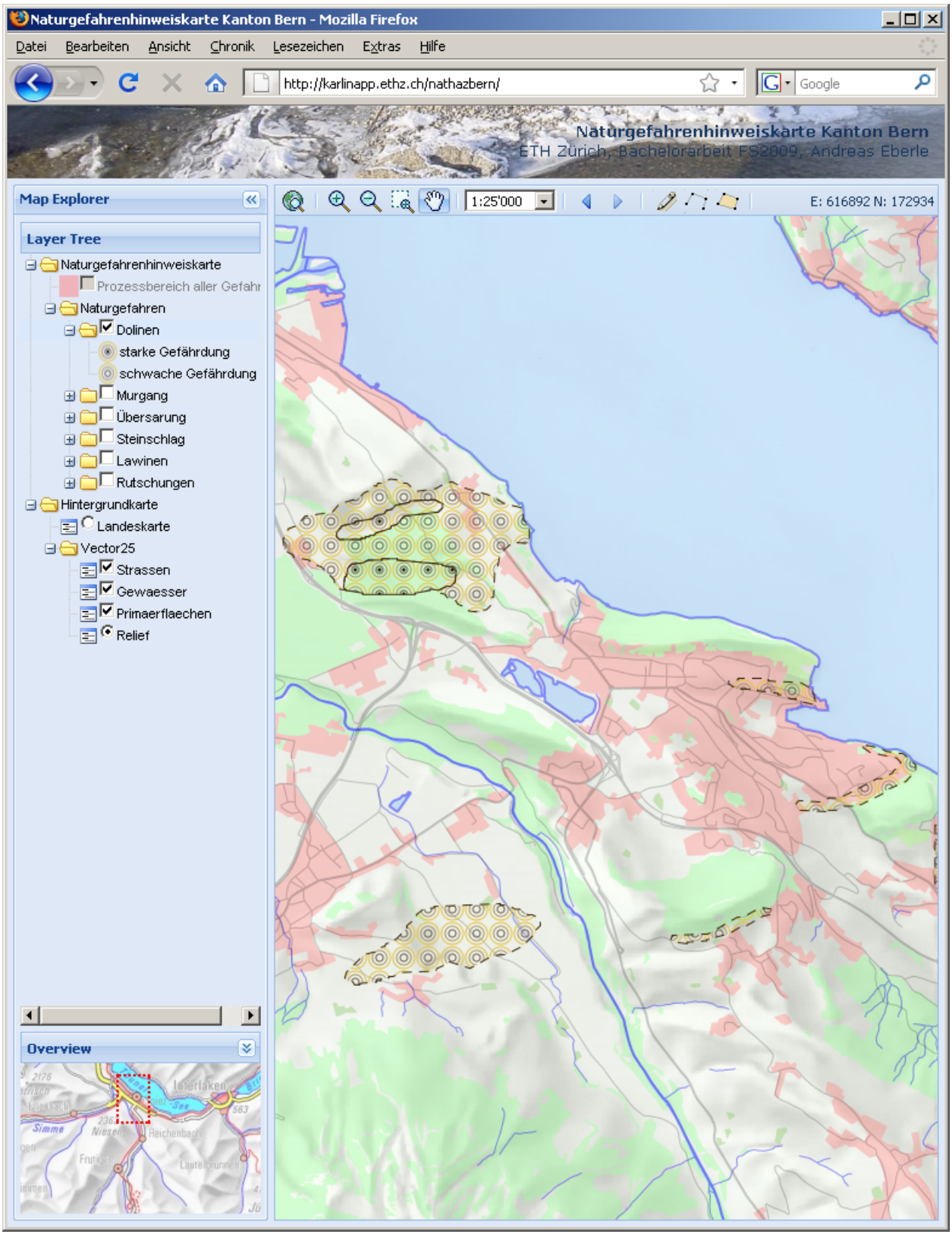
Murgang


Naturgefahrenhinweiskarte Kanton Bern - Mozilla Firefox
Datei Bearbeiten Ansicht Chronik Lesezeichen Extras Hilfe
http://karlinapp.ethz.ch/nathazbern/
Google
Naturgefahrenhinweiskarte Kanton Bern
ETH Zürich, Bachelorarbeit FS2009, Andreas Eberle
Map Explorer
Layer Tree
Naturgefahrenhinweiskarte
Prozessbereich aller Gefahr
Naturgefahren
Dolinen
Murgang
Schadenpotential
nicht relevanter Prozess
Übersarung
Steinschlag
Lawinen
Rutschungen
Hintergrundkarte
Landeskarte
Vector25
Strassen
Gewaesser
Primaerflaechen
Relief
Overview
1:25'000
E: 617845 N: 163797

Übersarung


Naturgefahrenhinweiskarte Kanton Bern - Mozilla Firefox
Datei Bearbeiten Ansicht Chronik Lesezeichen Extras Hilfe
http://karlinapp.ethz.ch/nathazbern/
Google
Naturgefahrenhinweiskarte Kanton Bern
ETH Zürich, Bachelorarbeit FS2009, Andreas Eberle
Map Explorer
Layer Tree
Naturgefahrenhinweiskarte
Prozessbereich aller Gefahr
Naturgefahren
Dolinen
Murgang
Übersarung
Schadenpotential
nicht relevanter Prozess
Steinschlag
Lawinen
Rutschungen
Hintergrundkarte
Landeskarte
Vector25
Strassen
Gewaesser
Primaerflaechen
Relief
Overview
1:25'000
E: 618497 N: 163215

Dolinen


Naturgefahrenhinweiskarte Kanton Bern - Mozilla Firefox
Datei Bearbeiten Ansicht Chronik Lesezeichen Extras Hilfe
http://karlinapp.ethz.ch/nathazbern/
Google
Naturgefahrenhinweiskarte Kanton Bern
ETH Zürich, Bachelorarbeit FS2009, Andreas Eberle
Map Explorer
Layer Tree
Naturgefahrenhinweiskarte
Prozessbereich aller Gefahr
Naturgefahren
Dolinen
starke Gefährdung
schwache Gefährdung
Murgang
Übersarung
Steinschlag
Lawinen
Rutschungen
Hintergrundkarte
Landeskarte
Vector25
Strassen
Gewaesser
Primaerflaechen
Relief
Overview
1:25'000
E: 616892 N: 172934

Anhang E: Ordner- und Filestruktur der CD-Beilage

Dokumente

Bericht:	/Dokumente/bacheloar_eberle_bericht.docx
Konzept:	/Dokumente/bachelor_eberle_konzept.docx
Poster:	/Dokumente/bachelor_eberle_poster.ai
Präsentation:	/Dokumente/bachelor_eberle_vortrag.pptx

Client

Applikation (HTML-Dokument):	/Client/index.html
SVG-Symbolsammlung (PHP):	/Client/svg.php
Controls (JavaScript):	/Client/js/Controls.js
Layers (JavaScript):	/Client/js/Layers.js
LayerTree (JavaScript):	/Client/js/LayerTree.js
Map (JavaScript):	/Client/js/Map.js
Toolbar (JavaScript):	/Client/js/Toolbar.js
Viewport (JavaScript):	/Client/js/Viewport.js
MapFish, OpenLayers, ExtJS	/Client/mfbase/*
Images, Styles:	/Client/styles/*
Layerbeschreibung (SLD):	/Client/svg/sldLayers.xml
SVG-Symbole:	/Client/svg/xxxYY.svg

Server/daten

Dolinen:	/Server/daten/ghk/ghkdol97.shp
Lawinen:	/Server/daten/ghk/ghklaw97.shp
Murgang:	/Server/daten/ghk/ghkmur97.shp
Rutschung:	/Server/daten/ghk/ghkrum97.shp
Steinschlag:	/Server/daten/ghk/ghkste97.shp
Übersarung:	/Server/daten/ghk/ghkueb97.shp
Übersicht:	/Server/daten/ghk/ghksyn97.shp
Relief:	/Server/daten/relief/reliXXX.tif
Gewässer:	/Server/daten/vec25/gwn25_l.shp
Primärflächen:	/Server/daten/vec25/pri25_a.shp
Strassen:	/Server/daten/vec25/str25_l.shp

Server/qgis_map_server

Layerkonfiguration:	/Server/qgis_map_server/admin.sld
Metainformationen:	/Server/qgis_map_server/wms_metadata.xml