MASARYKOVA UNIVERZITA
PŘÍRODOVĚDECKÁ FAKULTA
GEOGRAFICKÝ ÚSTAV

Diplomová práce

BRNO 2016 **LUDVÍK ADAMEC**

MASARYKOVA UNIVERZITA
PŘÍRODOVĚDECKÁ FAKULTA
GEOGRAFICKÝ ÚSTAV

Vektorové dlaždice ve webové kartografii

Diplomová práce

Ludvík Adamec

Vedoucí práce: Mgr. Bc. Jiří Kozel, Ph.D. **Brno 2016**

Bibliografický záznam

Autor:	Ludvík Adamec Přírodovědecká fakulta, Masarykova univerzita Geografický ústav
Název práce:	Vektorové dlaždice ve webové kartografii
Studijní program:	Geografie a kartografie
Studijní obor:	Geografická kartografie a geoinformatika
Vedoucí práce:	Mgr. Bc. Jiří Kozel, Ph.D.
Akademický rok:	2015/2016
Počet stran:	$77 + 4$
Klíčová slova:	vektorové dlaždice, webové mapy, přenos vektorových dat, GeoJSON, TopoJSON, Mapbox Vector Tile, prostorová indexovací služba

Bibliographic Entry

Author:	Ludvík Adamec Faculty of Science, Masaryk University Department of Geography
Title of Thesis:	Vector Tiles in Web Cartography
Degree Programme:	Geography and Cartography
Field of Study:	Geographical Cartography and Geoinformatics
Supervisor:	Mgr. Bc. Jiří Kozel, Ph.D.
Academic Year:	2015/2016
Number of Pages:	$77 + 4$
Keywords:	Vector Tiles; Web Maps; Vector data transmission; GeoJSON, TopoJSON, Mapbox Vector Tile; Spatial Indexing Service;

Abstrakt

Tato práce řeší optimalizaci přenosu prostorových vektorových dat pro potřeby webové kartografie a geoinformatiky. Rešeršní část se zabývá srovnáním možností přenosu vektorových dat ze serveru na klienta.

V praktické části byla navržena webová aplikace nad daty RUIAN. Přenos vektorových dat byl optimalizován pomocí metody vektorových dlaždic a prostorové indexovací služby. Pro obě metody byly dále navrženy a implementovány optimalizace.

V závěrečné části proběhlo výkonnostní měření, které porovnává vektorové dlaždice s prostorovou indexovací službou. Zároveň byl změřen i přínos navržených optimalizací.

Hlavním výsledkem práce je srovnání přístupů pro přenos vektorových dat a jejich rozšíření, které bylo prakticky ověřeno výkonnostním měřením.

Abstract

Thesis focus on optimization of a vector spatial data transmission for the needs in cartography and geoinformatics. Teoretical part deals with comparing of methods which transfer vector data from server to the client.

A web map application presenting data from RUIAN was designed in the framework of a practical part. The vector data transmission was optimised by a method vector tiling and a spatial indexing service. For both methods there was suggested optimization followed by implementation.

Finally measurement of performance was done. It resulted in technique comparison of the vector tiling and the spatial indexing service and their optimization.

Main outcome of the thesis is comparison of approches for the vector data transmission and their extension which was practically evaluated by perfomance measurement.

Masarykova univerzita

Přírodovědecká fakulta

ZADÁNÍ DIPLOMOVÉ PRÁCE

Student: **Ludvík Adamec**
Studijní program: **Geografie a kartografie**
Studijní obor: **Geografická kartografie a geoinformatika**

Ředitel Geografického ústavu PřF MU Vám ve smyslu Studijního a zkušebního řádu MU určuje diplomovou práci s tématem:

Vektorové dlaždice ve webové kartografii

Vector Tiles in Web Cartography

Zásady pro vypracování:

1. Popište techniku vektorových dlaždic, která se používá ve webové kartografii pro přenos geografických vektorových dat ze serveru na klienta.
2. Srovnejte tuto techniku s jinými podobnými technikami, například s rastrovými dlaždicemi nebo s přenosem kompletních (nerozsekaných) vektorových dat.
3. Srovnejte implementace vektorových dlaždic používaných v současné webové kartografii a navrhněte jejich možná vylepšení.
4. Navrhněte a vytvořte webovou mapovou aplikaci využívající vektorové dlaždice.

Rozsah grafických prací: podle potřeby

Rozsah průvodní zprávy: cca 60 až 80 stran

Seznam odborné literatury:

TARALDSVIK, Mats. The future of web-based maps: can vector tiles and HTML5 solve the need for high-performance delivery of maps on the web? Norwegian University of Science and Technology. Master thesis. 2012.128 s.
NORDAN, Robert Patrick Victor. An Investigation of Potential Methods for Topology Preservation in Interactive Vector Tile Map Applications. Norwegian University of Science and Technology. Master thesis. 2012. 97 s.
YING, Fangli. Management of spatial data for visualization on mobile devices. National University of Ireland Maynooth. Dissertation thesis. 2011. 206 s.

Jazyk závěrečné práce: čeština

Vedoucí diplomové práce: Mgr. Bc. Jiří Kozel, Ph.D.

Podpis vedoucího práce: .....

Datum zadání diplomové práce: listopad 2014
Datum odevzdání diplomové práce: do 5. května 2016

RNDr. Vladimír Herber, CSc.
pedagogický zástupce ředitele ústavu

Se zadáním diplomové práce souhlasím, jsem si vědom(a), že zadání práce je závazné.

Zadání práce převzal(a): ..... dne .....

Poděkování

Na tomto místě bych chtěl poděkovat Mgr. Bc. Jiřímu Kozlovi, Ph.D. za vstřícnost, cenné rady a výbornou spolupráci při zpracovávání diplomové práce. Dále bych chtěl poděkovat Soně Klusové, rodině a přátelům za jejich podporu.

Prohlášení

Prohlašuji, že jsem svoji diplomovou práci vypracoval samostatně s využitím informačních zdrojů, které jsou v práci citovány.

Brno 4. května 2016 .....

Ludvík Adamec

Obsah

1 ÚVOD

Webová kartografie se v posledních letech dynamicky rozvíjí, což souvisí s vzestupem všech odvětví navázaných na rozvoj internetu a informačních technologií. Rozvoj internetu sebou přinesl i velké množství prostorových dat, které je možné kartograficky vizualizovat.

Těmto změnám se přizpůsobuje i webová kartografie, která se od prezentace jednoduchých statických map přesunula přes interaktivní mapy, využívající zejména dat ve formě rastru, až k inovativním aplikacím a mapám, kde je datový zdroj často ve formě vektorových dat.

Vektorová data ve webových mapách přináší rozšířené možnosti pro interakci a vizualizaci, která se může dynamicky přizpůsobit každému uživateli podle jeho aktuálních potřeb. Personalizace map pomocí vektorových dat je důležitá, protože předat čtenáři mapy relevantní informaci ve světě zaplaveném daty je klíčovou úlohou kartografa. Vektorová data přinášejí široké možnosti do webové kartografie, ale zároveň přicházejí i s řadou výzev, které je nutné vyřešit. Jedním z úkolů je efektivní přenos dat, který zajistí rychlé zobrazení mapy. Toho lze dosáhnout například způsoby minimalizujícími objem přenášených dat ze serveru na klienta. Jednou z metod, která se zabývá efektivním přenosem dat, je metoda vektorových dlaždic, kterou se zabývá tato práce.

1.1 Cíle práce

Cílem práce bylo popsat techniku vektorových dlaždic, srovnat ji s jinými přístupy jako je WFS, prostorová indexovací služba nebo metoda rastrových dlaždic. Vektorové dlaždice a prostorová indexovací služba jsou technikami zabývající se přenosem geometrie pouze v geografické oblasti mapové dlaždice. Vektorové dlaždice geometrii objektů vždy rozdělí podle hranice dlaždice na části a prostorová indexovací služba posílá vždy kompletní geometrie, které mají průnik s dlaždicí.

Na základě rešerše metod byly navrženy optimalizace a úpravy pro vektorové dlaždice

a prostorovou indexovací službu, které by měli zlepšit použitelnost metod, zejména po stránce výkonnosti. Pro vektorové dlaždice byla oproti dnešním běžným postupům navržena rekonstrukce geometrií. Pro prostorovou indexovací službu byl navržen postup, který bude kombinovat vektorové dlaždice s prostorovou indexovací službou. Toho bylo dosaženo rozdělením geometrie pro prvky, které mají plochu výrazně větší, než je plocha mapové dlaždice.

Na technice vektorových dlaždic byla otestována i úspornost a efektivita uložení dat v topologickém formátu TopoJSON oproti formátu GeoJSON. Pro vektorové dlaždice byla navržena cache, která ukládá vygenerované dlaždice s cílem minimalizovat využití serveru při opakovaných dotazech.

Popsané techniky vektorových dlaždic a prostorové indexovací služby byly včetně navržených změn implementovány ve webové aplikaci, která jako datový zdroj používá data z RUIAN. Nad vytvořenou aplikací nakonec bylo provedeno automatizované měření, které porovnává výkonnost navržených změn s původním návrhem pro obě zmíněné metody.

1.2 Obsah kapitol

Kapitola 2 se zabývá popisem rastrových a vektorových formátů z pohledu webové kartografie a dále je zaměřena na metody přenosu rastrových dat ze serveru na klienta.

Kapitola 3 je zaměřena na techniky, jejichž cílem je přenášet vektorová data ze serveru na klienta. Zejména byl kladen důraz na metodu vektorových dlaždic a prostorové indexovací služby. Všechny techniky jsou popsány a zejména mezi sebou porovnány.

Kapitola 4 přináší možnosti optimalizace pro přenos vektorových dat, které je možné využít. Stěžejní je popis a srovnání datových formátů, které se používají ve webové kartografii.

Kapitola 5 představuje nástroje podporující metodu vektorových dlaždic. Byly přiblíženy serverové nástroje, webové služby i klientské řešení, které umožní vytvořit webovou mapu podporující přenos dat ve formě vektorových dlaždic. Kromě nástrojů budou zmíněny i aplikace, které v současnosti využívají techniku vektorových dlaždic.

Kapitola 6 se věnuje návrhu a implementaci webové aplikace, která poskytuje přenos dat z RUIAN ve formě vektorových dlaždic nebo prostorové indexovací služby. Kromě základní implementace bodou popsány i optimalizace, které jsou použity.

Kapitola 7 se zabývá výkonnostním měřením techniky vektorových dlaždic a prostorové indexovací služby na vytvořené aplikaci. Jsou zde prezentovány výsledky pro všechny metody i jejich optimalizace.

2 VEKTOROVÉ A RASTROVÉ FORMÁTY DAT VE WEBOVÝCH MAPÁCH

2.1 Rastrová data

Geografická data ve formě rastru, jsou vyjádřena sítí tvořenou jednotlivými elementy rastru, které jsou uspořádány obvykle do matice uspořádaných hodnot. Struktura rastru je většinou uspořádána do pravidelné mřížky, kdy základní elementy rastru jsou obvykle reprezentovány čtverci (pixely). Alternativou čtvercové sítě mohou být například obdélníky, trojúhelníky nebo šestiúhelníky. Rastrová data jsou uložena v kartézském souřadnicovém systému, který může mít vztažný bod. Vztažný bod reprezentuje připojení na souřadnicový systém. Tematická informace v rastru je uložena přiřazením hodnoty pixelu, což je obvykle celé nebo desetinné číslo. Prostorové rozlišení rastru je určeno délkou hrany pixelu. Rastrové formáty jsou vhodné například pro vyjádření spojitých veličin, jako je například teplota vzduchu nebo nadmořská výška. [1, s. 90]

2.2 Vektorová data

Vektorová data jsou složena ze dvou základních složek. První, která nese geometrickou informaci a druhá složka nese informaci popisnou. Prostorové souřadnice jsou uloženy v reálných souřadnicích a nedochází ke ztrátě prostorové informace. Geometrická složka může být reprezentována prostřednictvím topologicky-vektorového modelu nebo modelem netopologickým. Často využívaným netopologickým modelem je špagetový model. Tento model ukládá prostorovou informaci v uzlech a liniích, které jsou tvořeny sekvencí uzlů. Sekvence linií, která začíná i končí v identickém bodě, vytváří polygony. Nevýhodou špagetového modelu je, že nenese topologickou informaci, tedy dochází zde k redundanci dat a není přímo možné zjistit vztahy mezi geoprvky. [2]

Topologický vektorový model ukládá geometrii do uzlů a hran. Každá hrana je tvořena počátečním a koncovým bodem. Polygon je tvořen sekvencí hran. Topologický formát

umožňuje evidovat topologické vztahy mezi objekty ve formě základních topologických vlastností jako je konektivita (spojení objektů), orientace (směr orientovaných hran), přilehlost (sousednost ploch) a obsahování. V tomto formátu nedochází k redundanci dat, které mají například společnou hranici. Evidence topologické informace je důležitá například pro provádění určitých analytických operací. [1]

2.3 Srovnání vektorových a rastrových formátů v kontextu webové kartografie

Rastrový i vektorový formát má ve webové kartografii své nezastupitelné místo. Každý z formátů je vzhledem ke svým vlastnostem lépe využitelný k jinému účelu.

Rastrová data jsou výhodná pro reprezentaci spojitých jevů, jako jsou například měření fyzikálních veličin (teplota, vlhkost,..). Vektorová data mají využití pro diskrétní jevy, kde je nutné evidovat vymezení jednotlivých objektů (např. hranice parcel, adresní místa,..).

Rastrová data mají ve srovnání s vektorovými jednodušší datovou strukturu, proto je i jejich zpracování jednodušší. Pro některé jednodušší analytické operace je rastrový formát výhodnější ve srovnání s vektorovým formátem, kde složitost výpočtu analytických metod je vyšší. Nevýhodou rastrového formátu je, že umožňuje pouze omezené možnosti následné práce s daty. Ve srovnání s vektorem zde nepracujeme s konkrétními geoprvky, ale s obrazem jevu.

S vektorovými daty je jednodušší propojovat informace z několika zdrojů, které na sebe navzájem odkazují. Vektorový formát se také vyznačuje nižším objemem uložených dat ve srovnání s rastrem. Další výhodou vektorových formátu je možnost nést topologickou informaci, která je nutná pro provádění řady analytických operací. Tyto analytické operace mohou být vykonávány přímo na klientovi bez interakce se serverem.

Další rozšířené možnosti vektorového formátu přináší personalizace obsahu, která nám umožňuje zobrazit pouze část informace v datech, která je relevantní konkrétnímu uživateli. Navíc můžeme i za běhu měnit vizuální podobu mapy a přidávat nebo ubírat zobrazené informace. [3]

Vektorová data nám například umožňují provádět úpravy dat na straně klienta. Tato editace může být dobře využitelná například v oblasti dobrovolnického mapování. [4]

2.4 Přenos rastrových dat

V současnosti je velká část map prezentována ve formě interaktivních map na webu. Toto prostředí vyžaduje optimalizaci přenosu dat, protože přenášet data v celém jejich rozsahu není z důvodu datové náročnosti možné. Navíc interaktivní mapy umožňují uživateli měnit aktuální zobrazenou oblast nebo měřítko mapy, a je tedy nutné načítat data vždy s ohledem na aktuální oblast zájmu uživatele. Na základě těchto požadavků vznikly standardizované služby, které přenášejí data podle aktuálního zobrazení mapy. Mezi nejpoužívanější standardizované webové služby přenášející rastrová data patří Web Map Service (WMS)[5] a Web Map Tile Service (WMTS)[6].

WMS je specifikace konsorcia OGC standardizující rozhraní služby pro přenos map v podobě obrázku. Základní operací pro získání mapy obvykle v rastrovém formátu je dotaz GetMap. Dotaz vrací mapu podle zadaných parametrů. Mezi povinné parametry patří například BBOX, WIDTH a HEIGHT, podle kterých je určena oblast a měřítko požadované mapy. [5]

WMTS je standard, který si klade za cíl navrhnout škálovatelnou, vysoce výkonnou službu pro poskytování map na webu. WMTS služba je komplementární službou k WMS. WMS služba se zaměřuje na poskytování map podle zadaných parametrů, tedy je vhodná zejména pro dynamická data nebo vlastní kartografické znaky. Na druhé straně WMTS služba se zaměřuje na vysoce výkonné řešení pro statické mapy, které mají předem definovaný styl, obsah a systém dlaždic. Dlaždicování je technika, která mapové okno rozdělí na několik samostatných částí, pro které je mapa získávána samostatně. Výhodou je, že WMTS služba umožňuje ukládat již vytvořené dlaždice, z toho důvodu je opakované dotazování oproti službě WMS výrazně méně náročné na výkon na straně serveru a celková doba nutná pro získání dlaždice je významně zkrácena. [5] [6]

Obě služby umožňují přenášet data podle aktuální zobrazené oblasti v určitém měřítku, ale každá s jiným cílem. Služba WMS je určena pro dynamicky generované mapy a služba WMTS je specifikována jako vysoce výkonná služba pro předem vytvořené statické mapy.

3 PŘENOS VEKTOROVÝCH DAT VE WEBOVÉ KARTOGRAFII

Jeden z problémů při využití vektorových formátů dat na webu je nadměrný přenos dat. Často dochází k přenosu dat nejenom v aktuálním zobrazení mapy uživatele, s tím jsou spojeny i problémy s náročností zpracování a zdlouhavým přenosem dat ze serveru na klienta. Pokud budeme chtít například zobrazit mapu v okolí určitého města, ale naše data budou uložena ve vrstvách za kraje, tak bez použití určité prostorové selekce dat budeme přenášet nadměrné množství dat, které uživatel mapy nepotřebuje. Z toho důvodu je vhodné použít metody přenášející uživateli data pouze v určitém prostoru a v určité úrovní detailu prostorových dat [4].

Oproti popsaným metodám přenosu rastrových dat, které trpí problémem nemožnosti kombinovat vysoce výkonnou službu (služba WMTS) s dynamickým obsahem (služba WMS), je tento problém řešitelný při použití vektorových dat, kdy na straně serveru se předgenerují vektorová data, která se posílají na klienta, kde jsou dynamicky vizualizovány.

V následujících kapitolách budou představeny metody Web Feature Service (WFS), vektorových dlaždic, prostorové indexovací služby a UTFGrid, které umožňují optimalizovat přenos dat prostřednictvím extrakce dat na základě prostorové oblasti zájmu uživatele mapy.

3.1 WFS

Web Feature Service (WFS) je standard vytvořený organizací Open Geospatial Consortium (OGC) pro vytváření, úpravy a výměny vektorových dat na internetu pomocí protokolu HTTP[1] mezi mapovým serverem a klientem. WFS přenáší informace o objektech ve formátu Geography Markup Language (GML), založeném na XML [2]. [7]

[1] Hypertext Transfer Protocol

[2] XML - eXtensible Markup Language je obecný značkovací jazyk, který byl navržen pro snadné vytváření značkovacích jazyků pro různé účely.

WFS služba umožňuje komunikovat se serverem prostřednictvím několika požadavků, z nichž samotné získání prvku je realizováno dotazem GetFeature. Dotaz GetFeature umožňuje specifikovat řadu parametrů. Jedním z parametrů je parametr BBOX, který umožňuje specifikovat souřadnicemi hranice prostoru, z kterých mají být získány prvky.

Služba poskytuje možnosti pro dotazování podle zobrazené oblasti, ale bohužel není možné určit měřítko, pro které se mají geometrie získávat, proto jsou geometrie odeslány vždy v původní negeneralizované podobě.

Problémem při použití na webu může být kromě přílišné podrobnosti dat pro určité měřítko, také přenášení kompletních geometrií. WFS neumožňuje pro přenos dat provést rozdělení velkých geometrií na více částí, ale všechny geometrie i s velkým plošným rozsahem jsou odesílány vždy kompletní, což může způsobit problémy se zpracováním vzhledem k jejich objemu a také příliš dlouhé době přenosu dat ze serveru na klienta.

3.2 Vektorové mapové dlaždice

Dlaždicování je známá strategie pro rozdělování map a rozsáhlých geografických databází [8]. Při této strategii se využívá mapových dlaždic, které jsou v současnosti často využívány v rastrových formátech webových mapových služeb (WMTS). Vektorové dlaždice inspirované tímto postupem mají řadu společných i odlišných vlastností, které budou představeny.

Vektorové dlaždicování je metodou, kdy dochází k přenosu dat pouze v oblasti zájmu uživatele. Nedochází zde k přenosu dat, která uživatel nevyužije. Metodika vektorových mapových dlaždic v současnosti není popsána v žádném standardu organizace OGC, jedinou výjimkou je společnost Mapbox, která vydala vlastní otevřený standard pro jimi vyvinutý formát Mapbox Vector Tile [9]. Přesto jako bod odrazu může být brána specifikace organizace OSgeo s názvem Tile map service specification [10], která se zaměřuje na návrh komunikace mezi serverem a klientem pro metody přenosu dat ve formě mapových dlaždic.

Vektorová dlaždice je tedy malý prostorový výřez dat, obvykle čtverec, který obsahuje atributovou a geometrickou informaci. Geometrie plošných a liniových geoprvků je rozdělena na části podle BBOXu mapové dlaždice. Takto připravené dlaždice, které mají menší objem ve srovnání s kompletními daty je z hlediska času méně náročné přenášet na stranu klienta a uživatel nemusí tak dlouho čekat na zobrazení dat v mapě. Před zobraze-

ním mapy na straně klienta je nutné při využití vektorových dlaždic provést rekonstrukci dat do původní podoby [3]. Rekonstrukce dat probíhá spojením dlaždic do kompletních geometrií geoprvků. Tímto spojením se zachovává původní topologická informace, která byla v datech uložena.

Vektorové dlaždice mají řadu výhod oproti přenosu celé vrstvy dat. Hlavní výhodou je snížení množství dat přenášených ke klientovi [11], které je zejména důležité při omezené rychlosti internetu například na mobilních zařízeních. Další výhodou je nižší náročnost pro zpracování dat ve webovém prohlížeči. [12] Například pokud budeme chtít vizualizovat v prohlížeči data o desítkách MB, tak bez využití vektorových dlaždic nebo jiného optimalizovaného způsobu bude počáteční načtení mapy příliš dlouhé a následná práce s daty téměř nemožná.

Kromě prostorové extrakce dat je nutné uvažovat i o optimální přesnosti geometrie vzhledem k aktuálnímu požadovanému měřítku mapy. Je tedy nutné pro rozdílné měřítko (úroveň podrobnosti) poskytovat i adekvátní podrobnost geometrie a atributový obsah dlaždic. Snížení podrobnosti dat na požadovanou úroveň je realizováno prostřednictvím kartografické generalizace. [11][13]

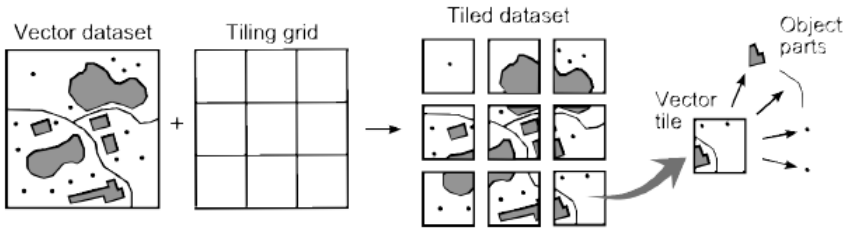
Vzhledem ke zmíněným požadavkům na zvýšení efektivity přenosu a zpracování dat je důležité na straně klienta ukládat již přenesené dlaždice, aby nedocházelo k nežádoucím opakovaným přenosům a redundanci dat.

Ve srovnání se službou WFS je hlavním rozdílem rozřezávání geometrie podle hranice dlaždice a zohlednění měřítka v generalizaci dat.

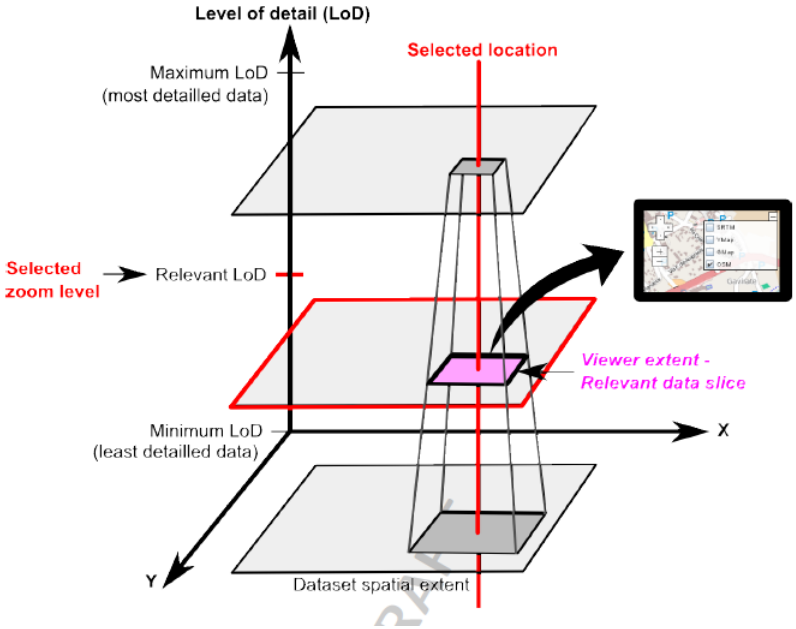
3.2.1 Generování a ukládání vektorových dlaždic

Při samotném generování dlaždic je nutné vektorová data rozdělit do mapových dlaždic. Toto rozdělení probíhá podle předem určeného gridu pro každou úroveň přiblížení. Po rozřezání celých geometrických objektů vzniknou vektorové dlaždice, které obsahují jednotlivé geometrické části (obr. č. 3.1). Typický grid používaný pro rozdělení je v oblasti webové kartografie postaven na kartografickém zobrazení Web Mercator (EPSG:3857)[14], které je odvozeno z Mercatorovy projekce. Toto zobrazení je využito například v případě Google Maps, OpenStreetMap a dalších aplikací. [15]

Generování dlaždic může probíhat na základě požadavku klienta, kdy dojde pouze k vygenerování určité oblasti v určitém stupni přiblížení (obr. č. 3.2). Druhou variantou je



Obrázek 3.1: Princip tvorby vektorových dlaždic. [4]



Obrázek 3.2: Vztah měřítka a úrovně podrobnosti dat. [4]

předgenerování mapových dlaždic. Při předgenerování dojde k uložení mapových dlaždic v celém rozsahu dat a všech požadovaných stupních podrobnosti, čímž předejdeme opakovanému generování dlaždic, které je výkonově náročné. Takto předgenerované mapové dlaždice mohou být uloženy do víceúrovňové databáze [13].

Před samotným rozdělením dat do mapových dlaždic je nutné upravit data na požadovanou úroveň podrobnosti (LoD). Tato úroveň podrobnosti je závislá na požadovaném měřítku a je možné ji ovlivnit zejména generalizací. [4]

3.2.2 Vytvoření mapy z vektorových dlaždic ve webovém prohlížeči

Pro vytvoření mapy v prohlížeči klienta je nutné provést tyto kroky. Nejdříve je nutné na požadavek klienta přenést vektorové dlaždice v oblasti zájmu a požadovaném LoD. Přene-

sené dlaždice jsou uloženy v cache uživatele. Z uložených dlaždic, je možné rekonstruovat původní data spojováním částí geometrií. Takto zrekonstruovaná data je možné zobrazit v mapě. Následně pokud se uživatel přesune na jinou oblast nebo změní úroveň přiblížení, tak se vždy kontroluje, jestli požadovaná data nejsou uložená v cache prohlížeče, pokud ne, tak dojde k odeslání požadavku dat na server.

Některé dřívější práce prezentovaly metodu vektorových dlaždic bez spojování geometrií na straně prohlížeče [16], čímž byla ztracena původní topologická informace a tato metodika mohla být využita pouze pro jednoduchou vizualizaci. Rekonstrukce původní geometrie umožní například provádění analytických operací a výpočtů. Zachování topologické informace se věnoval v diplomové práci zejména Taraldsvik [16] a dále Antoniou et al. [12].

Problematika efektivního zobrazení mapy pro uživatele je popsána v kapitole 4.

3.2.3 Rekonstrukce geometrické informace

Pro prezentaci ve webovém prohlížeči je nutné pro maximální využití vektorového formátu rekonstruovat původní geometrii. Rekonstrukcí geometrie umožníme na straně klienta provádět geometrické operace a data mohou být dále distribuována v původní kvalitě. Pro spojení jednotlivých částí geometrie je nutné při renderování vektorových dlaždic na straně serveru přidat jednotné identifikátory pro všechny části geometrie geoprvku. [17, s. 27] Před spojením se hledají nejdříve všechny části geometrie, které mají být spojeny. Nejjednodušším algoritmem pro vyhledání všech částí geometrii je nazván Global feature search. Tento jednoduchý algoritmus pro každou část geometrie v každé vektorové dlaždici prohledá všechny části geometrie a při shodě identifikátorů všechny nalezené části spojí. [17, s. 29]

Proces jednoduchého spojování má řadu kritických omezení a nevýhod. Největším z nich je, že v případě multi geometrií bude docházet k chybám, protože jakmile nebude jedna část geometrie mezi přenesenými dlaždicemi, tak nedojde k jejímu spojení. Vzhledem k tomu, že nikde není přenášena informace o počtu dlaždic nebo částí ve kterých se geometrie jednoho prvku nachází, tak není možné ani ověřit kompletnost rekonstruované geometrie. Další nevýhodou je časová náročnost hledání všech částí geometrie ve všech dlaždicích. Optimalizace algoritmů zaručující efektivnost a kompletnost geometrie je popsána v následující kapitole.

Samotný proces spojování je odlišný pro jednotlivé typy geometrie. Nejjednodušší

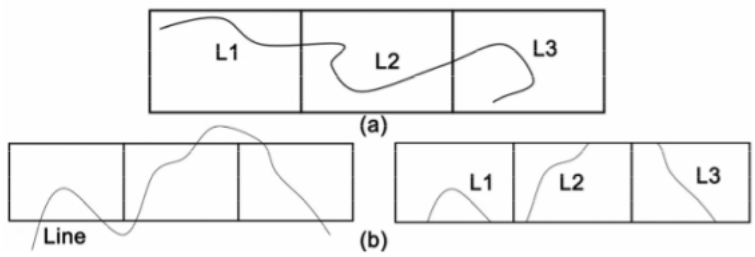


Fig. 3. The merge of a line that resides in different tiles can lead to: (a) polyline element or (b)multi-polyline element

Obrázek 3.3: Rekonstrukce liniové geometrie z polyline do multi-polyline geometrie. [12]

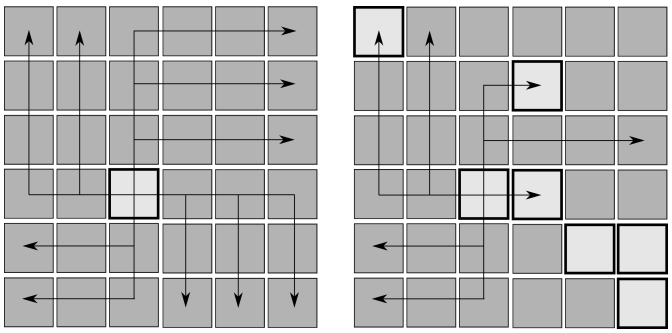
spojení se realizuje na bodových vrstvách, kde nedochází k žádné úpravě geometrie. Pro liniové geometrie se všechny části geometrie spojí do jedné geometrie, která je reprezentována geometrií typu polyline nebo multipolyline. Multipolyline vznikne i v případě, kdy původní geometrie byla geometrií typu polyline, ale nedošlo k načtení všech vektorových dlaždic, na kterých se geometrie nachází a dojde tedy k vytvoření několika izolovaných linií. (obr. č. 3.3). [12]

Pro polygonové vrstvy platí obdobný postup jako pro linie. Spojením geometrií vzniká vrstva s geometrií polygon nebo multipolygon.

3.2.4 Optimalizace rekonstrukce geometrie

Vzhledem k omezení algoritmu Global feature search byly v práci [17] navrženy další algoritmy pro vyhledání všech částí geometrie geoprvku. V této podkapitole budou přiblíženy dva, které splňují podmínku zaručení kompletnosti geometrie, tedy při jejich využití bude výsledná geometrie identická s geometrií původní.

První algoritmus s názvem Edge Pointers využívá ukazatelů, které jsou přidány ke každé geometrii. Tyto ukazatelé nesou odkaz na další části geometrie. Vzhledem k tomu, že někdy se může geometrie nacházet na velkém množství dlaždic, tak ukazatelé jsou pouze čtyři a ukazují na nejbližší další část geometrie v každém směru (sever, jih, východ, západ). Tímto postupem nedojde k velkému nárůstu velikosti dlaždic, který by nastal, pokud by se ukazatele uváděly pro všechny části geometrie. Výhodou tohoto algoritmu je, že zaručuje kompletnost geometrie a nedochází ke zvýšení jeho náročnosti s nárůstem komplexnosti a počtu dlaždic. Nevýhodou je, že je nutné implementovat na straně serveru přidávání ukazatelů. Další nevýhodou je nárůst velikosti dlaždic. Samotný postup vyhledávání ukazatelů



Obrázek 3.4: Shéma edge pointers. [17]

v okolí dlaždice je zobrazen na obr. č. 5.

Druhý algoritmus s názvem Central feature registry pracuje na principu uložení seznamu dlaždic, na kterých se geometrie nachází. Tento postup umožní jednoduše na základě identifikátoru geometrie získat kompletní seznam dlaždic, kde se nacházejí její další části. Algoritmus zajišťuje komplexnost geometrie. Nevýhodou je nutnost vygenerování dlaždic s geometriemi a indexů obsahující ke každé geometrii seznam dlaždic na kterých je rozdělena. Detaily a další algoritmy jsou uvedeny v práci Nordana [17, s. 29-40].

3.2.5 Pořadí vykreslení a rekonstrukce geometrie

Technika vektorových dlaždic se musí vzhledem k procesu rekonstrukce geometrie vypořádat s problematikou delšího čekání uživatele na mapu a zobrazení dat v mapě. Důvodem je, že před samotnou rekonstrukcí geometrie je nutné načítat vektorové dlaždice, které vzhledem k rozdílnosti komplexnosti geometrie mohou mít značnou velikost a tedy i načítání bude časově náročnější ve srovnání například s rastrovými dlaždicemi. Navíc po načtení nastává další časově náročná operace, kdy dochází k vyhledávání všech prvků geometrie a jejímu následnému spojení. Z toho důvodu je nutné pořadí procesu spojování a zobrazování dat pečlivě zvážit.

V zásadě existují 3 principy, kdy začít uživateli zobrazovat mapu a rekonstruovat geometrii. [17, s. 27]

a) Po načtení a spojení všech geometrií.

b) Progresivní způsob, kdy budou data zobrazována co nejdříve a následně budou překreslovány v průběhu rekonstrukce geometrie.

c) Okamžité zobrazení, které bude překresleno až na konci procesu, kdy bude geometrie kompletně rekonstruována.

První způsob je nejméně náročný na výkon, ale doba čekání uživatele na mapu bude velmi dlouhá. Druhý způsob zobrazí mapu dříve, ale stále bude na pozadí probíhat spojování geometrie a znovu vykreslování již rekonstruovaných geometrií. Poslední způsob ukazuje možnost, kdy jsou zobrazena nejdříve nerekonstruovaná data, uživatel tedy má ihned možnost vidět mapu a pracovat s ní v základním režimu. Následně dojde jednou k rekonstrukci geometrie, překreslení mapy a od této chvíle je uživateli umožněna úplná práce s daty.

3.3 Prostorová indexovací služba

V článku od Gafurriho [4] byla popsána metoda s originálním názvem Spatial Indexing Service, který byl pro účel této práce přeložen jako prostorová indexovací služba.

Prostorová indexovací služba je alternativní metodou k vektorovým dlaždicím. Rozdílem oproti vektorovým dlaždicím je v tom, že se přenáší kompletní geometrie geoprvků, které se nachází v aktuálně zobrazované oblasti. Prostorové indexování využívá jak na straně klienta, tak i na straně serveru identického prostorového indexu, typicky se jedná o quad-tree strukturu. [4]

Na serveru je uložený prostorový grid, který obsahuje reference na všechny geoprvky, které mají výskyt v určitém indexu. Na straně klienta je nutné mít dvě cache stejně jako u vektorových dlaždic. První seznam obsahuje již načtené indexy a druhá obsahuje samotné geoprvky. [4]

Oproti WFS službě je hlavní rozdíl v možnosti získat seznam prvků pro oblast dlaždice a na základě tohoto seznamu a seznamu již načtených prvků v paměti prohlížeče určit ty prvky, pro které se budou stahovat geometrie. Ve WFS službě je pro BBOX automaticky posílán kompletní seznam geometrií, to znamená, že při použití WFS služby budou některé geometrie stahovány několikrát, zatímco pro prostorovou indexovací službu pouze jedenkrát. Další přidanou možností prostorové indexovací služby je zohlednění požadované úrovně detailu, tedy geometrie se pro různá měřítka mapy stahují s různou úrovní generalizace. Nevýhodou oproti WFS je, že prostorová indexovací služba není standardizována a pokud je někde implementována, tak se jedná o proprietární řešení.

3.3.1 Vykreslení mapy prostřednictvím prostorové indexovací služby

Nejdříve je na straně klienta získán seznam požadovaných indexů na základě aktuálního zobrazení mapy. Pokud požadovaný index není již uložen v cache prohlížeče, tak je poslán požadavek na server. Server vrací odpověď klientovi se seznamem referencí na geoprvky obsažené v indexu, a následně jsou reference zkontrolovány v cache klienta. Pokud se zde nenacházejí, tak je poslán požadavek na server o zaslání kompletní geometrie geoprvku. Toto ověření je nutné, protože každý geoprvek je uložen jako reference ve všech indexech, kde došlo k průniku s jeho geometrií. Po zaslání geoprvku na klienta je okamžitě zobrazen, protože nemusí docházet k žádné rekonstrukci geometrie. [4]

3.4 Srovnání vektorových dlaždic a prostorové indexovací služby

Hlavním rozdílem je, že v případě vektorových dlaždic se přenáší pouze rozdělené geometrie dat v oblasti jedné dlaždice a následně je nutné takto rozdělené geometrie spojovat na straně prohlížeče. Naproti tomu prostorové indexování přenáší kompletní geometrie všech geoprvků, které mají průnik s požadovaným indexem. Na základě těchto rozdílů, lze usoudit, že využití vektorových dlaždic je výhodnější pro velké a jednoduché geometrie, zatímco prostorové indexy mají výhody pro složité komplexní geometrie s menší rozlohou. [4]

Nelze proto obecně určit lepší metodu, ale je nutné vybrat a uvažovat vždy na základě charakteru dat. Vektorové dlaždice budou výhodnější například pro vrstevnice nebo zaznamenané GPS trasy, a prostorové indexy budou lépe využitelné například pro parcely, malé plochy nebo bodové vrstvy.

3.5 UTFGrid

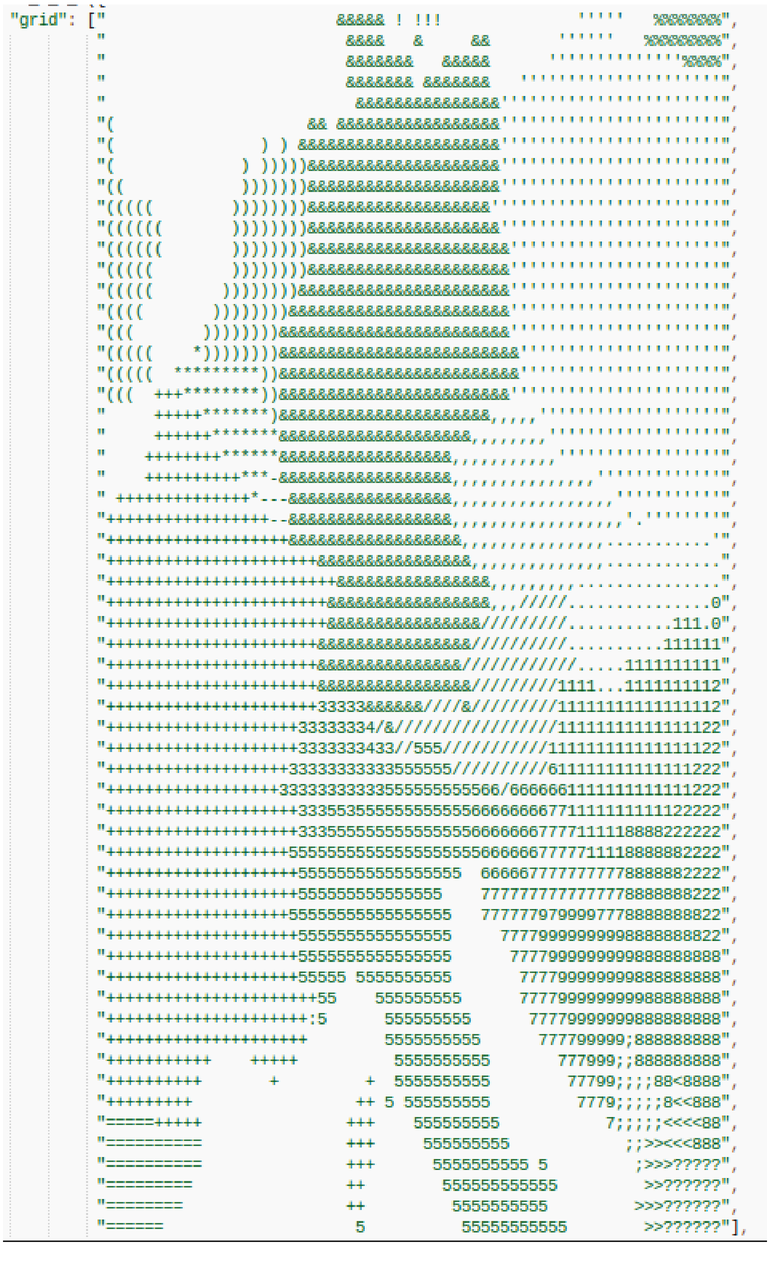
Kombinovaný přístup mezi rastrovými a vektorovými formáty přináší UTFGrid, který byl vytvořen proto, aby přinesl interaktivitu z vektorových do rastrových formátů. UTFGrid poskytuje data pomocí mapové dlaždice v rastrové formě, obdobně jako WMTS (ukázka na obr. 3.5), ale navíc kromě rastrového obrazu jsou přenášeny atributové informace a k nim připojená vektorová data ve zjednodušené formě. [18]

Tato vektorová data jsou přenášena ve formě gridu, kde každý jeden pixel nabývá jedné hodnoty, jejíž uložení je reprezentováno znakem z kódování UTF-8. Vektorový grid má

Obrázek 3.5: Ukázka rastrové dlaždice uložené ve formátu UTFGrid. [19]

délku strany 64 pixelů, což znamená, že jeden pixel vektorového gridu je ekvivalentní ke 4 pixelům gridu rastrového o straně 256 pixelů. Důvodem snížení rozlišení pro vektorový grid je, že interakce myší s jedním pixelem je nepřesná a proto je možné rozlišení snížit bez výrazného snížení přesnosti interakce ve výsledné mapě. Ukázka uložení vektorových dat z území Evropy ve fromě UTFGridu je na obr. 3.6. dlaždice. [20]

Výhodou tohoto způsobu je oproti rastrovým datům interaktivnost mapy, kdy například při přejetí kurzorem myši určitého polygonu se mohou zobrazovat jeho atributové informace. Ve srovnání s vektorovými metodami je zde výhoda v jednoduchosti zobrazení mapy, ale na druhou stranu s mapou není možné pracovat jako s reálnými vektorovými daty, například nelze jednoduše změnit vzhled určitých objektů, provádět manipulace s geometriemi nebo vytvářet analýzy nad vektorovými daty. Velikost přenášených vektorových dat je proměnlivá zejména v atributové složce, samotná vektorová data mají velikost vždy minimální.



Obrázek 3.6: Ukázka vektorové dlaždice uložené ve formátu UTFGrid. [19]

4 OPTIMALIZACE PŘENOSU VEKTOROVÝCH DAT

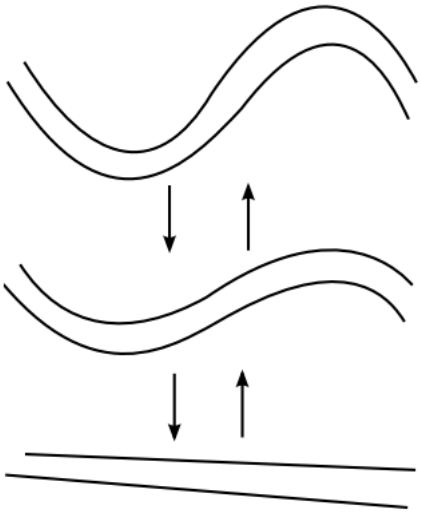
Webové stránky a kartografie na webu se musí vypořádat se specifickým chováním uživatelů. V případě webových stránek uživatelé očekávají co nejrychlejší získání odezvy na jejich požadavky.[21] Stejně tak lze předpokládat jejich požadavek na plynulou interakci v případě webových map a aplikací. Dosažením tohoto cíle může být realizováno například progresivním načítaní mapy [22], generalizací [13], optimalizací přenosu [16], přenosovým formátem dat [16] nebo způsobem jejich vykreslování do mapy [12].

4.1 Progresivní přenos vektorových dat

Progresivní přenos vektorových dat představuje efektivní způsob přenosu dat, který je založený na postupném přenosu dat ve formě úprav, které jsou postupně aplikovány do mapy. Nedochází tedy k nutnosti načíst data kompletně v jednom požadavku. [3][22]

Progresivní přenos dat je možné rozdělit na prostorem omezený přenos nebo měřítkem omezený přenos. Prostorové omezení je realizováno například v metodách vektorových dlaždic nebo prostorové indexovací služby, které již byly popsány. Měřítkem omezený přenos je založený na kontinuálním načítání geometrie prvků v mapě. Kontinuální načítání je realizováno opačným postupem, než probíhá generalizace pro jednotlivé úrovně podrobnosti geometrií (obr. č. 4.1). Nejdříve je tedy načtena základní, nejvíce generalizovaná geometrie s nejmenší datovou velikostí. Základní, méně podrobná geometrie je zobrazena v mapě a postupně s dalšími přicházejícími daty ze serveru je geometrie zdokonalována. [3]

Tímto postupem dojde k snížení velikosti přenášených dat, protože se omezí redundance při přenosu. Dalším významný přínosem je zobrazení alespoň částečné informace v mapě v nejkratším možném čase. Výhodou je, že uživatel může vykonávat interakci s mapou aniž by bylo nutné načíst kompletní data. Kompletní data jsou načítána pouze v oblasti, kde je uživatel vyžaduje a je ochotný na ně čekat delší dobu.



Obrázek 4.1: Vztah mezi generalizací a progresivním přenosem informací. [3]

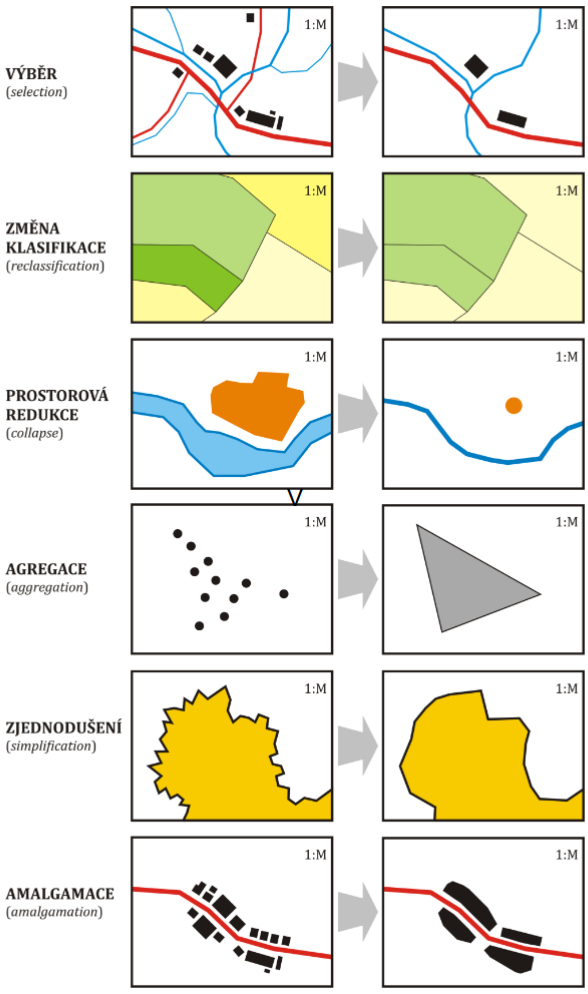
4.2 Generalizace

Pojem kartografická generalizace definuje slovník VÚGTK jako výběr a zevšeobecnění (zjednodušení) prvků a jevů zobrazovaných na mapě s přihlédnutím k měřítku nebo účelu mapy [23]. Jedná se tedy o metody kdy dochází k úpravě dat nebo jejich atributů. Generalizace se používá z řady důvodů. Pro vektorové dlaždice a webové mapy je důležité zejména za účelem snížení náročnosti na výkon na straně prohlížeče, snížení objemu přenášených dat ze serveru ke klientovi a zpřehlednění informace.

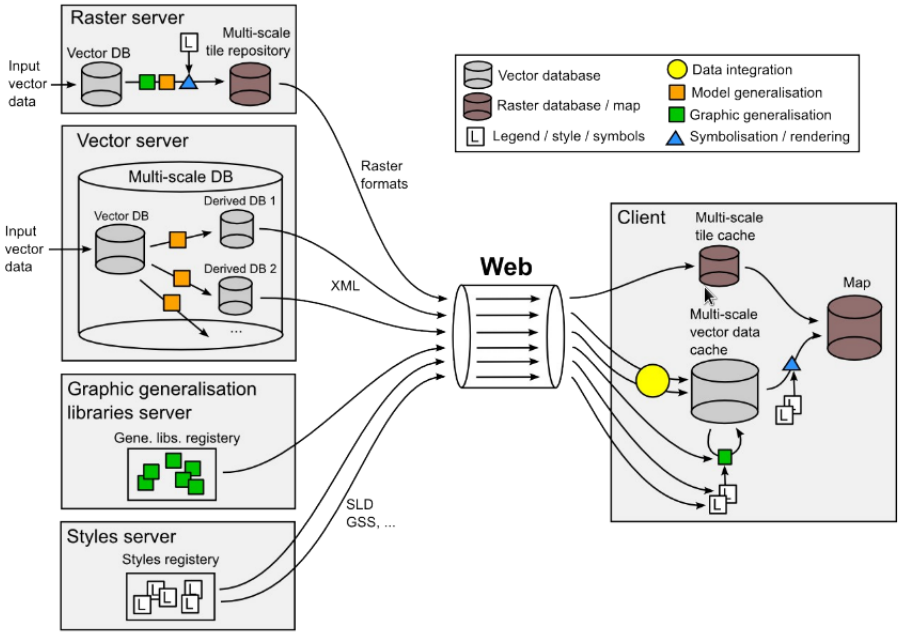
Generalizační operátory, které jsou zastoupeny samotnými algoritmy, řeší dílčí problémy kartografické generalizace. Mezi základní operátory (obr. č. 4.2) patří výběr, změna klasifikace, prostorová redukce, agregace, zjednodušení a amalgamace. [24] Operátory generalizace se do jisté míry u řady autorů odlišují. Samotné generalizační operátory a jejich algoritmické implementace jsou mimo rozsah této práce, proto nebudou dále rozváděny.

Cílem generalizace v tradiční kartografii bylo zajistit obsah mapy v požadovaném měřítku a na základě účelu mapy. Naproti tomu webová kartografie musí řešit problematiku interaktivity map, kdy generalizace musí být v souladu se změnami přiblížení mapy, změnami oblastmi zájmu nebo i tématu mapy. Z toho důvodu je v řadě vědeckých prací předmětem zájmu automatická generalizace, problematika vícenásobné reprezentace (multiscale database) nebo generalizace v reálném čase (on the fly) [25][26].

V práci Gaffuriho [13] byla navržena architektura podporující současné požadavky na



Obrázek 4.2: Operátory generalizace. [24]



Obrázek 4.3: Architektura systému pro webové mapy s generalizací. [13]

vektorové mapy. Gaffuri uvádí nutnost předgeneralizování zdrojové databáze do několika odvozených databází s rozdílnou úrovní podrobnosti.Takto předgenerovaná data snižují časovou náročnost přenosu dat ze serveru na klienta. Na straně klienta se nachází cache uchovávající vícenásobné reprezentace geoprvků. Klient musí dále podporovat odesílání požadavků na různě úrovně podrobností, zároveň by měl uložená data za běhu dále generalizovat a v posledním kroku jim nastavit styl a zobrazit je v mapě. Nevýhodou tohoto systému je značná náročnost na zpracování dat na serveru, kde jsou uchovány duplikáty geometrií v několika úrovních a každá změna v datech se musí vždy projevit na těchto všech úrovních.

Zmíněná architektura je dobře využitelná pro aplikace využívající vektorová data, které vyžadují vícenásobné reprezentace objektů a možnosti získávat data pouze za určitou oblast.

4.3 Formáty pro přenos vektorových dat na webu

Volba formátu pro přenos vektorových dat ve ve webovém prostředí hraje významnou roli, protože má vliv nejenom na obsah, ale také na velikost dat, která je kritická pro webové aplikace. Každé snížení objemu přenášených dat má významný přínos, protože dojde ke zkrácení doby přenosu dat ze serveru na klienta. Kromě snížení doby načítání

je vzhledem k popularitě mobilních zařízení nutné zohlednit i datové limity, které mají uživatelé mobilního internetu.

V následujících sekcích budou popsány otevřené formát pro přenos vektorových dat, konkrétně GML, GeoJSON, TopoJSON a otevřený binární formát pro vektorové dlaždice od společnosti Mapbox. Zmíněné formáty byly vybrány, protože jsou v současné době velmi populární pro přenos vektorových dat na webu.

Kromě volby formátu je nutné zohlednit i způsob uložení souřadnic, který může mít velký vliv na velikost. Například volba vhodného přesnosti dat, kdy budeme přenášet data s pouze nezbytným počtem desetinných míst může přinést významnou úsporu objemu dat.

4.3.1 GML

Geography Markup Language XML vznikl za účelem popisu geografických objektů a umožňuje uložení geometrie i atributů (vlastnosti) objektů. Skládá se ze dvou částí, z gramatiky a samotných dat. Gramatika popisuje, jaká data může dokument uchovávat. Pro její zápis se využívá standard XML Schema (XSD). Objekty reálného světa v GML nazýváme prvky (features). Aktuální verze 3.2.1 je zároveň ISO standardem (ISO 19136:2007). GML je také předepsaný technickými dokumenty INSPIRE a výchozím formátem služby WFS. GML se používá jako univerzální formát pro data, která mohou mít i komplikovanější stromovou strukturu. Díky tomu, že je postaven na XML, je jeho strojové zpracování možné například pomocí transformace XSLT. [27]

4.3.2 GeoJSON

Formát GeoJSON je nezávislým formátem pro přenos prostorových a atributových informací. GeoJSON vychází z formátu JSON, což je univerzální datový formát, který lze zpracovávat v libovolném programovacím jazyku. Formát JSON, z kterého vzniknul GeoJSON slouží pro komunikaci s webovými službami. JSON je populární alternativou k formátu XML. [28]

Předností formátu je jednoduchá struktura a nezávislost. Prostorová data se ukládají ve formě point, line, polygon, multipoint, multiline nebo multipolygon). V současnosti je formát podporovaný nejenom mezi webovými řešeními, ale je i v desktopových aplikacích. [28]

```
{
    "type": "FeatureCollection",
    "features": [
        {
            "type": "Feature",
            "geometry": {
                "type": "Point",
                "coordinates": [
                    49.0,
                    18.5
                ]
            },
            "properties": {
                "key": "value"
            }
        }
    ]
}
```

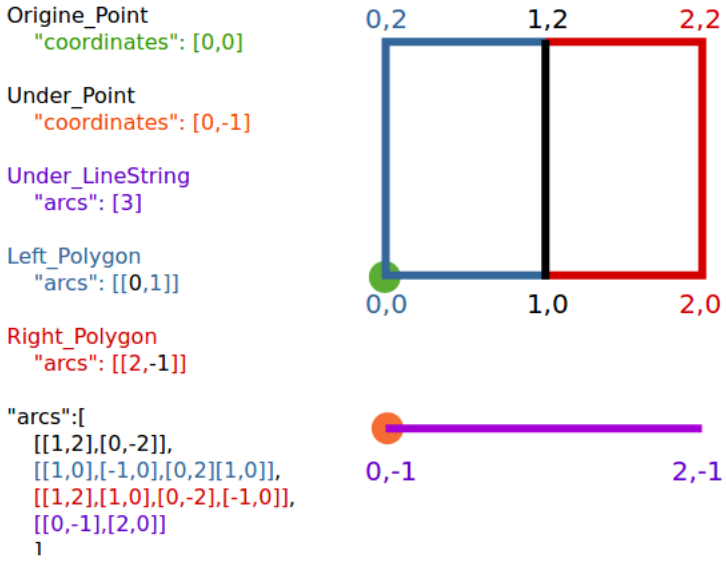
Ukázka kódu 4.1: Uložení dat ve formátu GeoJSON.

4.3.3 TopoJSON

TopoJSON je topologickým datovým formátem, který byl vytvořen jako rozšíření geografického formátu GeoJSON. Oproti formátu GeoJSON, který geometrie ukládá samostatně pro každý geoprvek jsou geometrie v TopoJSON formátu uloženy ve sdílené podobě 4.4, kdy se objekty odkazují na jednotlivé linové segmenty, z kterých se složí výsledná geometrie. Výhodou je, že nedochází k ukládání redundantní informace, tedy například pro sdílenou hranici mezi polygony bude tato hranice uložena pouze jednou namísto dvou samostatných geometrií. Přínosem tohoto postupu je nejenom přenos přidané informace v podobě topologie, ale také výrazné snížení objemu dat.

Další úspora velikosti je v tomto formátu možná pomocí optimalizace uložení souřadnic vertexů. TopoJSON umožňuje transformovat a relativizovat polohu souřadnic vůči počátku souřadnic, která se definuje pro samotný TopoJSON soubor. Tím se výrazně zkrátí hodnoty pro souřadnice s mnoha desetinnými místy, čímž dojde k ztrátové kompresi, která ale pro jisté účely může být výhodná. [29]

Snížení objemu dat je důležité zejména ve webovém prostředí, kde pro interaktivní mapy je kladen velký cíl na minimalizaci doby přenosu dat ze serveru na klienta. Snížení velikosti dat oproti formátu GeoJSON může být podle charakteru dat až o 80% [30]. [31]



Obrázek 4.4: Uložení dat ve formátu TopoJSON. [30]

4.3.4 Binární formáty

V současné době jsou vektorové dlaždice na internetu populární zejména díky společnosti Mapbox, která vyvinula otevřený binární formát pro vektorové dlaždice, který je nazván Mapbox Vector Tile v dokumentu vector-tile-spec [9]. Formát je postaven nad Google Protocol Buffers, což je binární formát vyvinutý a používaný Googlem pro efektivní výměnu dat, který si jako hlavní cíl klade být maximálně optimalizovaný co se týče velikosti uložení dat i rychlosti serializace a deserializace [32]. Kromě samotné specifikace pro uložení dat Mapbox vyvíjí otevřené nástroje a software, který umí ukládat, konvertovat, číst nebo prezentovat vektorové dlaždice ve formě webových map. Vektorové dlaždice v tomto formátu jsou v současnosti podporovány v nejpoužívanějších JavaScriptových knihovnách, které jsou určeny pro tvorbu webových map, konkrétně se jedná o OpenLayers, Leaflet, Mapbox GL JS, Mapzen Tangram, dokonce i společnost ESRI oznámila podporu formátu v produktu ArcGIS API for JavaScript [33].

Společnost Mapbox je zřejmě nejvíce propagující společností pro přenos dat ve formě vektorových dlaždic, ale způsob uložení i využití vektorových dlaždic je specifický.

Samotné uložení geometrie v dlaždicích je vždy v kartografickém zobrazení Web Mercator (EPSG:3857). Souřadnice jsou relativizovány vůči počátku dlaždice a uložení hodnoty souřadnice je vždy jako celé číslo, což znamená, že stejně jako pro formát TopoJSON dochází ke ztrátové kompresi.

Hlavním rozdílem oproti standardní metodě vektorových dlaždic je ovšem uložení geometrií s malým přesahem přes hranici dlaždice. Tento přesah je následně využit při vykreslování, protože tyto přesahy jsou vzhledem k překryvům mezi sousedními dlaždicemi skryty. Proto dochází k iluzi a výsledná mapa vypadá jako kdyby proběhla rekonstrukce geometrií a jejich spojování i když ve skutečnosti jsou geometrie stále rozděleny. Velkou výhodou tohoto přístupu je mnohem vyšší rychlost zobrazení dat v mapě, ale na druhou stranu takto zpracované geometrie neumožňují pracovat s geometriemi jako při klasické službě vektorových dlaždic, protože jeden geoprvek může mít geometrii rozdělenou mezi několik dlaždic. [9] Využití tohoto formátu je vzhledem k jeho specifičnosti vhodné zejména jako mapového podkladu, ale ve své základní podobě nelze použít například pro prostorové analýzy nebo editaci geometrií. [34]

Vzhledem k otevřenosti formátu by ovšem bylo možné využít dlaždic ve zmíněném formátu a zpracovat je stejným postupem jako pro klasické vektorové dlaždice, tedy provést rekonstrukci geometrií, což zmiňují i vývojáři nástrojů pro tento formát v internetové diskuzi [34]. Bohužel reálné nasazení tohoto postupu se nepodařilo v žádné z volně dostupných aplikací nalézt.

Vzhledem k narůstající podpoře formátu v řadě technologií se dá předpokládat, že využívanost tohoto formátu bude narůstat a v budoucnu se možná i bude využívat po implementaci rekonstrukce geometrií jako plnohodnotný vektorový formát.

4.4 Přednačítání vektorových dlaždic na straně klienta

V práci Antonioa et al. [12] byl navržen postup přednačítání mapových dlaždic ve chvílích neaktivity uživatele a jejich přípravou před samotným zobrazením v mapě. Na straně prohlížeče je nutné vytvořit strukturu mapového dokumentu, který je tvořen ve třech úrovních. První úroveň obsahuje pouze načtené mapové dlaždice v jejich původní podobě. Druhá úroveň představuje zrekonstruované geometrie vytvořené z dlaždic na první úrovni. A poslední třetí úroveň je samotná mapa v mapovém okně uživatele vytvořená z dat na druhé úrovni. Rozsah oblasti na úrovni první úrovně je větší než na druhé a ta má větší rozsah než úroveň třetí. Z toho důvodu se mapa v okolí zobrazené oblasti generuje předem a uživatel při pohybu s mapou již nemusí čekat na stahování dlaždic ze serveru, ani na rekonstrukci geometrie.

Na obr. č. 4.5 je znázorněna struktura mapového dokumentu. Zmíněná práce se vypořádala s efektivností posunu mapy, ale bohužel se již nezaměřuje na zoom mapy. Lze

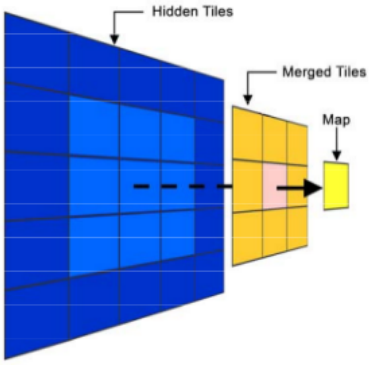


Fig. 2. The structure of the map document

Obrázek 4.5: Schéma přednačítávání dat. [12]

předpokládat, že by bylo možné načítat dopředu i dlaždice na menším i větším měřítku.

5 APLIKACE A PROJEKTY PODPORUJÍCÍ VEKTOROVÉ DLAŽDICE

V současnosti je dostupná velká řada nástrojů i společností, které se zabývají vektorovými dlaždicemi. Velká část nástrojů je dostupná jako open source. Projekty zabývající se dlaždicemi můžeme rozdělit na několik skupin

5.1 Servery pro generování vlastních dlaždic a poskytovatele vektorových dlaždic

OSM2VectorTiles

OSM2VectorTiles je projekt, který nabízí volně ke stažení vektorové dlaždice vytvořené z dat OpenStreetMap a dále poskytuje sadu open-source nástrojů pro použití a generování vektorových dlaždic. Projekt nabízí i technologie a nástroje, které zjednodušují sestavení vlastního serveru pro poskytování vektorových dlaždic. Samotné předpřipravené vektorové dlaždice je možné stahovat za celý svět nebo podle regionů. Dlaždice jsou pravidelně aktualizovány na základě aktuálních dat. [35]

Geoserver

Komerční verze balíku OpenGeoSuite, sloužící jako soubor open source nástrojů, nabízí možnost produkovat mapové dlaždice ve formátu GeoJSON, TopoJSON nebo MapBox Vector Tile. Kromě komerční verze Geoserveru existuje i open source rozšíření pro volně dostupnou verzi Geoserveru, které nabízí možnost vytvořit vektorové dlaždice, ale pouze ve formátu Mapbox Vector Tile. [36][37]

PGRestAPI

Open source mapový server, který jako jeden z možných výstupů, poskytuje vektorové dlaždice ve formě Mapbox Vector Tile. Samotné generování dlaždic je postaveno nad datovým úložištěm v databázi Postgresql s rozšířením Postgis. [38]

Mapbox

Jak již bylo zmíněno v kapitole 4.3.4, Mapbox nabízí řadu nástrojů od aplikace pro tvorbu mapových stylů pro dlaždice, přes generování vlastních dlaždic nad vlastními daty po poskytování mapových dlaždic prostřednictvím webové služby. Samotné poskytování mapových dlaždic je rozděleno na několik tarifů a základní varianta, která je omezená počtem stažených dlaždic je bezplatná. Mapbox nabízí další nástroje, které je možné využít při tvorbě mapové aplikace, využívající jako podklad jejich vektorové dlaždice. [39]

Mapzen - Vector Tile Service

Produkt Vector Tile Service společnosti Mapzen nabízí vlastní mapové dlaždice prostřednictvím online dostupné služby. Tato služba je pro testování a projekty menšího charakteru zdarma, pro komerční větší projekty je nutné dohodnout individuální podmínky. Velkou výhodou této služby je, že poskytuje dlaždice ve formátech GeoJSON, TopoJSON nebo Mapbox vector tile. [40]

TileStache

TileStache je server, který umí poskytovat data ve formě mapových dlaždic. Dlaždice mohou být v rastrovém nebo vektorovém formátu. Vektorové dlaždice mohou být vytvořeny ve formátu GeoJSON, TopoJSON nebo Mapbox Vector Tile. [41]

ESRI - ArcGIS API for JavaScript

Nabízí službu poskytující předpřipravené mapové dlaždice ve formátu Mapbox vector tile. Této služby je možné využít v produktech postavených na technologii ArcGIS API for JavaScript. Kromě samotné služby poskytující dlaždice s předpřipraveným kartografickým stylem je možné vytvořit vlastní kartografický styl. [42]

5.2 Klientská řešení podporující vektorové dlaždice

V této části budou popsány knihovny, které lze použít pro tvorbu interaktivních webových map a podporují nějakým způsobem využití vektorových dlaždic. Tabulka níže zobrazuje seznam dostupných knihoven rozdělených podle podpory zmiňovaných formátů pro vektorové dlaždice. Podpora byla zjišťována pro formáty TopoJSON, GeoJSON a Mapbox Vector Tile formát.

JavaScript knihovna	**GeoJSON**	**TopoJSON**	**Mapbox Vector Tile**	**Ukázka**
OpenLayers [43]	ANO	ANO	ANO	http://openlayers.org/en/v3.15.1/examples/mapbox-vector-tiles.html
ArcGIS API for JS [42]	Nezjištěno	Nezjištěno	ANO	http://arcg.is/1NbxHKj
D3 [44]	ANO	ANO	ANO	http://bl.ocks.org/mbostock/5593150
Tangram [45]	ANO	ANO	ANO	https://tangrams.github.io/carousel/?cinnabar
MapboxGL [46]	Nezjištěno	Nezjištěno	ANO	https://www.mapbox.com/mapbox-gl-js/api/
Leaflet [47]	ANO	Nezjištěno	ANO	https://github.com/SpatialServer/Leaflet.MapboxVectorTile

Tabulka 5.1: Srovnání podpory vektorových dlaždic v JS knihovnách.

Z tab. č. 5.1 je možné vidět, že JavaScriptové knihovny používané v současnosti pro tvorbu webových map se zaměřily na implementaci vektorových dlaždic. Je důležité poznamenat, že zřejmě ve všech zmíněných knihovnách jsou vektorové dlaždice implementovány pouze jako funkcionalita pro mapový podklad, tedy nedochází k rekonstrukci geometrií, ale pouze ke zobrazení vektorových dlaždic, které mají přesah geometrií přes hranici dlaždice. Tyto přesahy jsou při zobrazení skryty a geometrie ze sousedních dlaždic se v mapě vizualizuje jako zrekonstruovaná, i když je ve skutečnosti rozdělená na více částí.

Implementace vektorových dlaždic s podporou pro rekonstrukci geometrií je možné

zřejmě provést ve všech zmíněných knihovnách, ale tato funkcionalita by musela být naprogramována samotným uživatelem API bez podpory samotné knihovny.

5.3 Aplikace využívající vektorové dlaždice

Kromě zmíněných knihoven, které podporují při tvorbě webové aplikace vektorové dlaždice je důležité zmínit projekty, které využívají principu vektorových dlaždic pro své potřeby. Všechny aplikace, které budou zmíněny využívají vlastních proprietárních formátů a nástrojů.

Zřejmě nejvýznamnějším projektem využívající vektorové dlaždice je projekt Google Maps, který vektorové dlaždice využívá již od roku 2010 [48]. Dalším projektem je vědecký projekt OpenScienceMap [49]. OpenScienceMap nabízí mapového klienta využívajícího vektorových dlaždic ve formě aplikace pro Android. Kromě mobilní aplikace je na stránkách projektu zveřejněna webová aplikace a součástí projektu je i server, který generuje vektorové dlaždice z projektu OpenStreetMap. Zdrojový kód aplikace je otevřený a je možné ho využít pro další vývoj. Posledním projektem, který bude zmíněn je aplikace GIS Cloud, která vystupuje jako webový GIS [50].

Vzhledem k tomu, že nástroje i formáty byly vytvořeny pro samotné projekty, nelze jednoduše zjistit, jestli vektorové dlaždice rekonstruují nebo zobrazují geometrie z dlaždic rozdělené.

6 PRAKTICKÁ ČÁST

V rámci diplomové práce byla vytvořena aplikace pro prezentaci vektorových dat ve webovém prostředí. Zdrojem dat je registr územní identifikace, adres a nemovitostí (RUIAN), který obsahuje data o územních jednotkách na území celé ČR. RUIAN obsahuje data od úrovně hranic státu, přes hranice krajů až po katastrální parcely a stavební objekty. Vzhledem k tomu, že uživatel v mapě zobrazuje pouze určité území a není žádoucí ani možné načítat kompletní data RUIAN, bylo nutné využít některého z principů zmíněných v teoretické rešerši. Došlo k vytvoření webové aplikace, která využívá principu vektorových dlaždic (kap. 6.7) a prostorové indexovací služby (kap. 6.8). Oba přístupy byly následně optimalizovány (kap. 6.7.3) a proběhlo jejich výkonnostní měření (kap. 7).

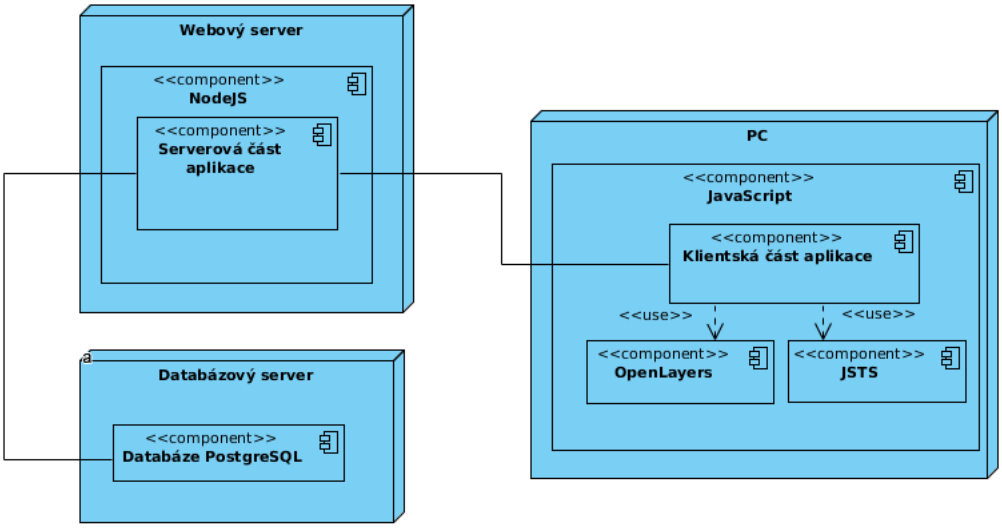
6.1 Architektura systému

Architektura systému pro vektorové dlaždice a prostorovou indexovací službu byla navržena podle Gaffuriho článku Toward web mapping with vector data [4]. Systém je zde rozdělen na dvě základní části, část serverovou a část klientskou, které mezi sebou komunikují přes HTTP[1]. V článku je serverová část rozdělena na několik částí, z nichž v této práci byla implementována pouze část týkající se vektorových dat.

Na straně serveru byla vytvořena databáze, do které byla nahrána vektorová data, a z nich byly odvozeny nové geometrie generalizací. Na straně serveru byla dále připravena funkcionalita, která zpracovává požadavky klienta a připravuje odpovědi s daty z databáze. Klientská část systému je mapa ve webovém prohlížeči, s kterou pracuje uživatel. Na straně klienta byla vytvořena funkcionalita zobrazující vektorová data. Klientská část dynamicky získává ze serveru vektorová data podle měřítka a aktuální zobrazené oblasti v mapě.

Základní architektura systému je znázorněna na obrázku 6.1. Architektura je rozdělena na základní zařízení, kterými jsou webový server, databázový server a počítač uživatele.

[1] HTTP - Hypertext Transfer Protocol je internetový protokol určený pro výměnu hypertextových dokumentů ve formátu HTML



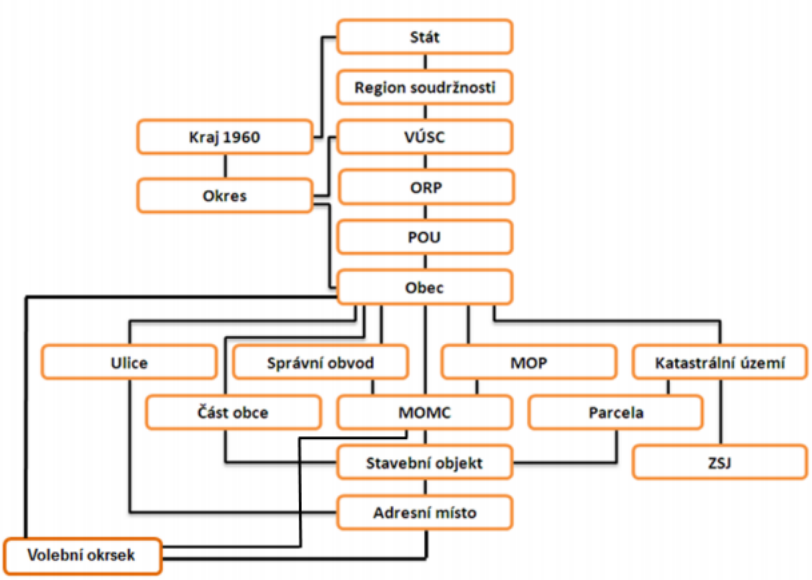
Obrázek 6.1: Architektura systému aplikace

Serverová část aplikace je vytvořena v běhovém prostředí NodeJS [51] a komunikuje s databází PostgreSQL v databázovém serveru. Klientská část je JavaScriptová aplikace, která jako základní komponenty využívá knihovnu pro zobrazení mapy OpenLayers [43] a knihovnu pro manipulace s vektorovými daty JSTS [52]. Aplikace byla vytvořena tak, aby podporovala vektorové dlaždice i prostorovou indexovací službu.

6.2 Zdrojová data

Jako datový zdroj pro aplikaci byla použita zmiňovaná data RUIAN. Data jsou rozdělená na dvě základní úrovně, od úrovně státu po obce a od obcí dále. Datový model je zobrazený na obr. 6.2, kde je uveden seznam všech vrstev, které jsou v RUIAN dostupné.

Přístup k datům RUIAN probíhá pomocí Veřejného dálkového přístupu (dále jen VDP). VDP je služba dostupná na webu Českého úřadu zeměměřického a katastrálního (ČUZK). Služba umožňuje stahovat data ve formě předpřipravených balíčků a to za území státu až obcí (data obsahují informace za administrativní celky vyšší úrovně) nebo detailnější úrovně (data za jednu obec). Předpřipravené balíčky se dále dělí podle obsahu, na kompletní data nebo jejich výběr. Dále je možné stahovat kompletní data (vydávaná jednou za měsíc) nebo data pouze s denními změnami. V diplomové práci byla stažena data kompletní s originálními negeneralizovanými hranicemi, aby mohla být využita i v nejvyšším stupni LoD. Samotné stažení dat probíhá dotazovací metodou GET protokolu HTTP na VDP.



Obrázek 6.2: Datový model RUIAN ([53])

Aplikace VDP umožňuje využití automatizace při stahování těchto dat, z toho důvodu byl vytvořen skript (ruianDownloader.py) automatizující stahování dat za požadované obce.

6.3 Uložení dat

Stažená data z RUIAN byla naimportována do databáze PostgreSQL, což je objektově-relační databázový systém, který je v současnosti vydáván pod open source licencí [2]. Pro práci a ukládání prostorových dat je možné systém rozšířit extenzí PostGIS [54]. PostGIS přidává možnosti ukládání a zpracování prostorových dat. Rozšíření je postaveno na specifikaci Simple Features od organizace OGC [55]. Instalací rozšíření přidáme do databáze základní geometrické typy (point, linestring, polygon, multipoint, multilinestring, multipolygon a geometrycollection), prostorový R-tree index a funkce pro analýzu a zpracování prostorových dat. [56, s. 17]

Pro import dat do databázového systému byl vytvořen skript v programovacím jazyce Python. Skript se obsluhuje z příkazového řádku a interně pro import dat využívá knihovny GDAL[3], která zahrnuje nástroj pro práci s vektorovými daty OGR. Knihovna OGR od verze 1.11 umožňuje práci s daty ve formátu VFR (Výměnný formát RÚIAN/ISÚI), který je použit pro uložení dat RUIAN. Formát VFR je XML formátem, který vznikl rozšířením

[2] http://opensource.org/licenses/postgresql
[3] http://www.gdal.org/

GML 3.2.1 (odpovídá ISO 19136:2007). [53]

Po importu zmíněným nástrojem dojde k vytvoření tabulek podle datového modelu na obr. 6.2.

6.4 Generalizace

Vzhledem k tomu, že základní myšlenka pro vektorové dlaždice i prostorovou indexovací službu je přenos generalizovaných dat na základě aktuálního LoD v mapě, tak bylo nutné vyřešit otázku generalizace dat. Byl zvolen přístup předgenerování samotných geometrií, před generováním za běhu (on the fly), které by výrazně snížilo výkonnost aplikace. Pro uložení generalizovaných geometrií byl pro každý LoD vytvořen nový sloupec, do kterého byly generalizované geometrie uloženy. Sloupce geometry1 až geometryN reprezentují různé úrovně detailu (LoD) a každý LoD je vhodný pro jiné měřítko v mapě (resolution).

ID	**geometry1**	**geometry2**	**...**	**geometryN**
feature1				
feature2				

Tabulka 6.1: Ukázka uložení generalizovaných geometrií.

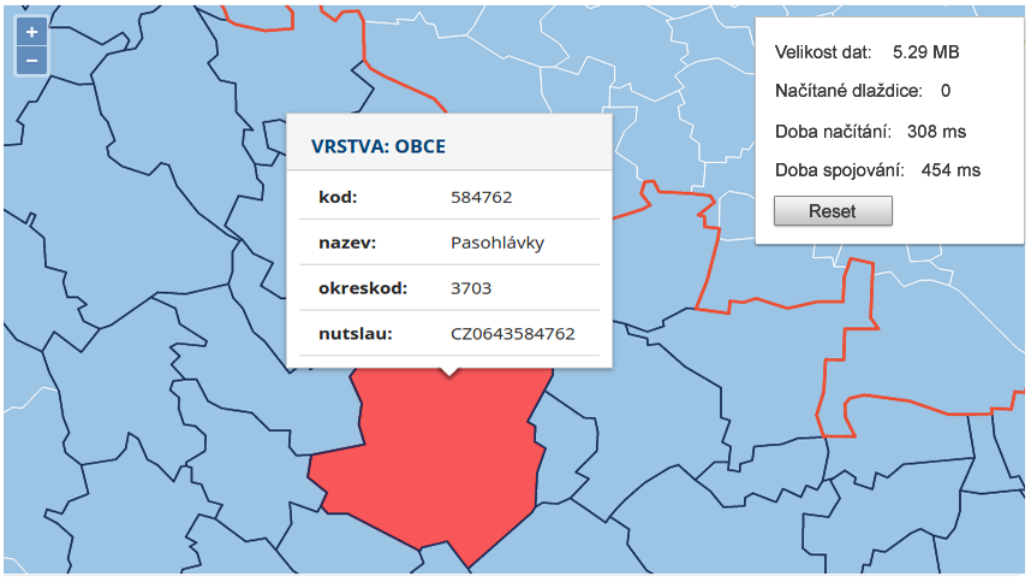
Vzhledem k počtu LoD a vrstev byl vytvořen skript (generalizePotsgis.py) pro automatizovanou generalizaci prostřednictvím nástroje Mapshaper [57]. Mapshaper je nástroj, který původně vznikl na univerzitě ve Wisconsin-Madisonu. V současné době je jako open-source dostupná druhá verze nástroje. Pro generalizaci dat byl využit algoritmus Visvalingam.

6.5 Klientská část aplikace

Klientská část aplikace je napsána v jazyce JavaScript a zobrazuje se prostřednictvím webového prohlížeče. Výstupem z aplikace je interaktivní webová mapa, se kterou pracuje uživatel aplikace. Ukázka aplikace je na obr. č. 6.3. Ukázková testovací aplikace je ve funkční podobě dostupná na `http://ruian.ludvikadamec.cz/`.

Pro vytvoření mapy ve webovém prostředí byla použita JavaScriptová knihovna OpenLayers [4]. OpenLayers je open-source JavaScriptová knihovna, navržená pro tvorbu in-

[4] http://openlayers.org/



Obrázek 6.3: Ukázka vytvořené mapové aplikace nad daty RUIAN.

teraktivních webových map. OpenLayers nabízí vývojářům API rozhraní – Application Programming Interface, které nabízí širokou funkcionalitu a podporu datových zdrojů. V současné době je knihovna již ve své třetí verzi.

6.5.1 API OpenLayers pro vektorová data

Pro potřeby aplikace zobrazující vektorová data pomocí vektorových dlaždic nebo prostorové indexovací služby je nutné se zaměřit na možnosti zpracování vektorových dat knihovnou. Knihovna OpenLayers je napsána technikou objektově orientovaného programování. Vzhledem k tomu, že hlavním cílem této práce je pracovat s vektorovými formáty, je nutné popsat základní třídy pro práci s mapou a vektorovými daty.

Jádro knihovny je třída ol.Map, která pracuje s ostatními třídami jako jsou vrstvy (ol.layer.Base), controls (ol.control.Control) nebo interactions (ol.interaction.Interaction).

Třída **ol.Map** pracuje s vrstvami rastrového nebo vektorového typu. Základními abstraktními třídami reprezentující vrstvy jsou třídy ol.layer.Base a ol.layer.Layer.

Vrstva ol.layer.Layer používá k uložení samotných dat třídu ol.source.Source, ktrerá je abstraktní třídou používající se pro tvorbu tříd reprezentující formát dat (např. ol.source.tile - data rozdělená do dlaždic, ol.source.Vector - vektorová data). Vektorové vrstvy jsou reprezentovány třídou **ol.layer.Vector**, která je základní třídou pracujícími s daty vektorového typu. Samotná data jsou v třídě uložena třídou ol.source.Vector.

Třída **ol.source.Vector** je základní třídou, která pracuje se samotnými daty vektorového typu. Vektorová data v knihovně OpenLayers jsou rozdělena do jednotlivých prvků, které knihovna reprezentuje objekty typu ol.feature. Objekty ol.feature obsahují vlastnosti (properties), které ukládají atributové informace nebo samotnou geometrii. Každý objekt ol.feature standardně obsahuje geometrii (reprezentována třídou ol.geom.Geometry), symbologii (reprezentována obvykle třídou ol.style.Style) a identifikátor (id). Geometrie uložená v objektech typu ol.feature se používá pro vykreslení dat v mapě. Vzhledem k tomu, že ol.feature zobrazuje v mapě geometrii z atributu geometry, tak bylo nutné pro výslednou aplikaci najít řešení, které by na základě aktuálního měřítka mapy měnilo obsah vlastnosti geometry a nastavila se vždy vhodná geometrie podle LoD. Jednotlivé LoD jsou reprezentovány generalizovanými geometriemi uloženými v databázi.

Základní třída ol.source.Vector byla vytvořena pro vrstvy u kterých se předpokládá, že geometrie geoprvků je reprezentována pouze jednou geometrií. Z toho důvodu bylo nutné pracovat s jinou třídou než ol.source.Vector.

Do verze 3.11 byla součástí knihovny OpenLayers třída s názvem **ol.source.TileVector**, která byla určena pro načítání dat ve formě vektorových dlaždic. Tato třída byla využita pro vytvoření prototypu aplikace, ale její využití způsobovalo při vývoji omezení kvůli některým funkcím, které v porovnání s třídou ol.source.Vector nebylo možné použít (např. konfigurace přednačítání dat i za hranici aktuálního zobrazení mapy). Z toho důvodu byla vytvořena na základě ol.source.Vector nová třída ol.source.MultiLevelVector.

Třída **ol.source.MultiLevelVector** byla navržena tak, že každá ol.Feature bude reprezentovat jednu entitu ve všech úrovních detailu. Zároveň každá ol.Feature bude mít více atributů s geometrií (geometry1 až geometryN), z nichž každá bude vhodná pro jiné rozlišení (resolution). Při změně rozlišení mapy se načte geometrie pro dané rozlišení a nastaví se jako atribut geometry. Třída ol.source.MultiLevelVector slouží tedy jako třída, která umožňuje dynamicky měnit geometrii a z toho důvodu byla využita pro vektorové dlaždice i prostorovou indexovací službu. Další třída určená pro vektorové dlaždice je třída ol.source.VectorTile [5].

Kromě způsobu uložení vektorových dat v OpenLayers bylo nutné vytvořit alternativní způsoby načítání dat pro vektorové dlaždice i prostorovou indexovací službu. OpenLayers

[5] Knihovna OpenLayers od verze 3.11 nabízí novou třídu ol.source.VectorTile. Tato třída nahrazuje zrušenou ol.source.TileVector. Nová třída ol.source.VectorTile je vhodná pro využití v systému vektorových dlaždic, kdy tyto dlaždice budou sloužit pouze jako mapový podklad. Výhodou tohoto systému je, že vykreslování geometrií v mapě je významně rychlejší, ale není možné s geoprvky (ol.feature) pracovat jako v případě prvků v ol.source.Vector, z toho důvodu nebyla tato třída použita.

ve třídě ol.source.Vector, a z ní odvozené třídě ol.source.MultiLevelVector, umožňuje definovat ve vlastnosti strategy tři základní funkce pro načítání dat. Základní strategie načítá kompletní data vrstvy při přidání do mapy (ol.loadingstrategy.all). Druhá strategie (ol.loadingstrategy.bbox), stejně jako třetí (ol.loadingstrategy.tile), načítá data vrstvy po částech (dlaždicích) podle aktuálního zobrazení mapy. Ve výsledné aplikaci byl zvolen třetí způsob načítání.

Kromě strategie načítání bylo nutné vytvořit i speciální loader, samostatně pro vektorové dlaždice i pro prostorovou indexovací službu. Pro vektorové dlaždice byl vytvořen vectorTileLoader a pro prostorovou indexovací službu spatialIndexLoader.

Loader v OpenLayers reprezentuje mechanismus, kterým jsou data stažena ze serveru, zpracována na straně klienta a přidána do vrstvy. Loader je vždy zavolán prostřednictvím funkce, která má několik parametrů, z nichž dva jsou využity pro získání dat. První parametr je extent (určuje geografický prostor, pro který se mají načíst data) a druhý je resolution (rozlišení mapy určuje LoD, které slouží k výběru správné míry generalizace).

6.5.2 Rekonstrukce geometrií

Vzhledem k tomu, že geometrie vektorových dlaždic se ze serveru posílají rozdělené na několik částí, tak byl vytvořen modul, který tyto části geometrií spojuje do původní podoby, aby nebyly v mapě reprezentovány v několika ol.feature objektech, ale vždy pouze v jednom. V rámci tohoto modulu (soubor mergeTools.js) bylo pro operaci spojení geometrií využito knihovny JSTS. [52]

JSTS je JavaScriptová knihovna poskytující základní geometrické operace (např. spojení geometrií nebo obalová zóna). JSTS vznikla odvozením od původní knihovny JTS, která stejnou funkcionalitu nabízela v jazyce Java.

Další části klientského řešení jsou popsány samostatně pro vektorové dlaždice a samostatně pro prostorovou indexovací službu.

6.6 Serverová část

Na straně serveru bylo pro aplikaci vektorových dlaždic a prostorové indexovací služby zvoleno běhové prostředí Node.js.

6.6.1 Node.js

Node.js je meziplatformní open source běhové prostředí, které využívá JavaScriptu na straně serveru. Node.js je postaven nad jádrem V8 JavaScript Engine od společnosti Google, které je součástí webového prohlížeče Google Chrome. Node.js je postaven na principu neblokující vstupně-výstupní operace (asynchronní I/O operace), které snižují režie procesoru a maximalizují výkon. [52]

Příkladem blokující operace je například čtení z databáze, kde dochází k čekání na výstup operace a dochází tedy k ”zastavení běhu” kódu. Node.js pracuje s neblokujícími vstupně-výstupními operacemi, kdy na vstupu je předáno zpětné volání, které se volá po dokončení operace, zatímco jsou vykonávány další operace a nedochází tedy k ”zastavení běhu”.

Vzhledem k tomu, že Node.js využívá jednovláknové zpracování, tak není příliš výhodný pro operace vysoce náročné na výkon procesoru. Je to z důvodu, že zpracování dalších požadavků by čekalo příliš dlouho ve srovnání s tradičními servery, využívajícími vícevláknové zpracování. Prostor, kde Node.js nachází využití, jsou aplikace vyžadující zpracování velkého množství méně náročných operací za krátký časový úsek. [58]

V kontextu systému vektorových dlaždic i prostorové indexovací služby se během krátkého časového úseku volá na server velký počet výkonově nenáročných dotazů. Z toho důvodu se Node.js ukazuje jako ideální kandidát, protože dotazy dokáže vyřídit rychleji a s menším výpočetním výkonem, než typický webový server pracující s blokujícím I/O operacemi.

6.6.2 Implementace serveru v Node.js

Pro snadný vývoj serverové části aplikace byl zvolen framework Express.js, což je Node.js webový aplikační serverový framework. Byl navržen jako jednoduchý framework usnadňující interakci s Node.js API. Základem je tedy jednoduchý webový server, který je možné rozšířit řadou funkcí ve formě pluginů. [59]

Základní funkcionalita webového serveru pro aplikaci je v souboru index.js. Je zde vytvořeno routování pro všechny požadavky, které jsou z klienta posílány. Pro každý požadavek byl vytvořen samostatný modul, který zpracovává vstupní parametry. Na základě nich se z databáze vytvoří odpověď a odešle ji zpět na klienta.

Popis funkcionality jednotlivých modulů je popsán samostatně pro požadavky vekto-

rových dlaždic v kap. 6.7.2 a pro prostorovou indexovací službu v kap. 6.8.3.

6.7 Implementace pro vektorové dlaždice

6.7.1 Popis implementace vektorových dlaždic na straně klienta

Soubor index.js obsahuje základní modul, který komunikuje dále s ostatními moduly aplikace. Jsou zde zadány konfigurační údaje, tedy informace nutné pro připojení do databáze a seznam vrstev RUIAN, které se mají v aplikaci načíst. Kromě konfigurace připojení je zde inicializována mapa a zároveň jsou zde do mapy přidány vrstvy, do kterých budou uloženy data z RUIAN. Vrstvy přidané do mapy jsou založeny na třídě ol.source.MultiLevelVector.

Kromě standardních vrstev reprezentující vrstvy RUIAN je zde navíc jedna vrstva, která je určena pouze pro načítání dat, protože v rámci jedné dlaždice jsou posílána data všech vrstev RUIAN. Tím dochází k významnému snížení počtu dotazů odeslaných na server, oproti postupu, kdy by se v jedné dlaždici načítala data pouze za jednu vrstvu. Načítání je realizováno prostřednictvím modulu VectorTileLoader (soubor vectorTileLoader.js), reprezentující loader pro zmíněnou fiktivní vrstvu, která se stará pouze o získávání dat pro všechny ostatní vektorové vrstvy. Tato vrstva si interně ukládá seznam dlaždic, které už byly načteny.

Modul VectorTileLoader se stará o samotné načítání dat a dále komunikuje s modulem MergeTools, který rekonstruuje původní geometrie.

Modul MergeTools vystupuje v systému jako dočasné úložiště nezrekonstruovaných geometrií před tím, než se spojí do původní podoby. Modul umožňuje zpracovat data pro vektorové dlaždice i prostorovou indexovací službu, zároveň umožňuje zpracovat data ve formátu GeoJSON a TopoJSON. Rekonstruovaná data jsou z modulu předána zpětným voláním do VectorTileLoader. Zde jsou zrekonstruované geometrie přidávány do jedné z vytvořených instancí třídy ol.source.MultiLevelVector, které zobrazují geoprvky v mapě.

Samotná komunikace se serverem probíhá AJAX voláním [6]. Volání na získání vektorové dlaždice je realizováno voláním metody GET na server. Ukázkové volání z JavaScriptu s využitím knihovny jQuery[7] je v ukázce kódu č. 6.1. Volání definuje URL adresu serveru (http://localhost:9001/se/renderTile), typ metody (GET), návratový typ (json) a objekt data.

[6] AJAX (Asynchronous JavaScript and XML) je označení pro technologie vývoje interaktivních webových aplikací, za pomoci asynchronního zpracování dotazů. https://developer.mozilla.org/en-US/docs/AJAX

[7] https://jquery.com/

Objekt data obsahuje tři vlastnosti, x, y a z, které reprezentují souřadnici dlaždice v tilegridu [61].

```
$.ajax({
  url: "http://localhost:9001/se/renderTile",
  type: "get",
  data: {"x": 1, "y": 1, "z": 1},
  datatype: 'json',
  success: function(data, status, xhr){},
  error:function(er){}
});
```

Ukázka kódu 6.1: Dotaz /renderTile pro získání vektorové dlaždice.

Na dotaz /renderTile je vrácen ze serveru objekt (ukázka kódu č. 6.2), který obsahuje XYZ souřadnice a dále obsahuje objekty, které mají jako název klíče použit název vrstvy. Objekty s klíčem v podobě názvu vrstvy obsahují samotná prostorová data ve formě GeoJSON.

```
{
  "xyz": {
    "x": 2237,
    "y": 1408,
    "z": 12
  },
  "json": {
    "obce": {
      "type": "FeatureCollection",
      "features": []
    },
    "okresy": {
      "type": "FeatureCollection",
      "features": []
    }
  }
}
```

Ukázka kódu 6.2: Odpověď ze serveru na dotaz /renderTile.

Příchozí data jednotlivých geoprvků jsou dočasně uložena v modulu MergeTools, kde po načtení všech dlaždic aktuálního zobrazení mapy dojde ke spojování s ostatními rozdělenými a již načtenými částmi geometrie. Zrekonstruované geometrie jsou uloženy do příslušných instancí ol.source.MultiLevelVector, která je zobrazí v mapě.

6.7.2 Implementace vektorových dlaždic na straně serveru

Na straně serveru je připraveno směrování na adrese **/renderTile** v souboru index.js. Zde dochází k předání vstupního i výstupního parametru funkce do modulu RenderTile (soubor render-tile.js), kde je požadavek obsloužen. Přicházející požadavek má vstupní parametry ve formátu souřadnic tilegridu, tedy souřadnice X (pořadí dlaždice od levého okraje), souřadnice Y (pořadí od horního okraje) a souřadnice Z (reprezentuje přiblížení mapy). Na základě těchto souřadnic je v modulu sestaven Structured Query Language (SQL) dotaz, který ořízne geometrie mající průnik s dlaždicí.

Samotný ořez je v Postgisu proveden funkcí **ST_Intersection**, která ořezává geometrii podle geometrie určené zeměpisnými souřadnicemi. Vzhledem k tomu, že ořez není možné provést na základě souřadnic XYZ, byl proveden přepočet těchto souřadnic do souřadnic zeměpisných, které určující hranici dlaždice. Tento přepočet byl vytvořen na základě nástroje Maptiler [8], který má za cíl rozdělit rastrový obraz do dlaždic. Nástroj pro toto rozdělení musí nejdříve určit seznam dlaždic v rámci geografického prostoru, z toho důvodu obsahuje i funkcionalitu pro převod mezi zeměpisnými a dlaždicovými souřadnicemi. Tento přepočet je implementován v jazyce Python, z toho důvodu byla v diplomové práci část tohoto přepočtu přepsána do jazyka JavaScript, aby bylo možné přepočet udělat v rámci Node.js.

Po přepočtu z dlaždicových do zeměpisných souřadnic je zjišťováno pro každou vrstvu, jestli v databázi obsahuje sloupec s geometrií podle souřadnice Z, která reprezentuje LoD. Pokud vrstva obsahuje geometrický sloupec, tak je z této vrstvy získána geometrie všech prvků, které mají průnik se souřadnicemi hranice dlaždice, přičemž jsou tyto geometrie podle hranice dlaždice ořezány. Z geometrií získaných v databázi je vytvořen výstupní objekt ve formátu GeoJSON.

Po získání geometrií všech vrstev, které obsahují požadovaný geometrický sloupec je na stranu klienta odeslána odpověď (ukázka kódu č. 6.2).

6.7.3 Optimalizace aplikace

V průběhu vývoje aplikace bylo zjištěno, že by bylo možné aplikaci vektorových dlaždic optimalizovat prostřednictvím cachování a změnou datového formátu, který je využíván pro přenos dat ze serveru na stranu klienta. Nově zvoleným formátem je TopoJSON, který

[8] http://www.maptiler.org/google-maps-coordinates-tile-bounds-projection/

oproti implementovanému formátu GeoJSON umožňuje výrazně snížit objem přenášených dat (viz kap. 4.3.3), což by podle předpokladů mělo zvýšit výkonnost aplikace.

Cachování bylo implementováno pomocí databáze Apache CouchDB[9], což je robustní, škálovatelná, distribuovaná, dokumentově-orientovaná databáze s HTTP rozhraním. Data v CouchDB jsou reprezentována ve formě dokumentu na bázi JSON objektu.

Samotná implementace cachování je navázaná na existující požadavek /renderTile, kde dochází nejdříve k dotazu do CouchDB cache, kde se ověří jestli dlaždice s požadovanými souřadnicemi XYZ je již uložena. Pokud je dlaždice nalezena, tak je odeslána z cache na stranu klienta. Pokud dlaždice v cache není nalezena, tak je postupem zmíněným v minulé kapitole vytvořena a odeslána klientovi. Zároveň po odeslání na stranu klienta je dlaždice uložena do cache v databázi CouchDB, aby při příštím požadavku na dlaždici nemusela být generována z databáze PostgreSQL. Výhodou tohoto postupu je, že dlaždice je z originální databáze generována pouze jednou, což výrazně sníží náročné operace vyhledávání geometrií mající průnik s dlaždicí a jejich ořezání. Přidanou hodnotou cachování je snížení požadavků na výkon na straně serveru.

6.8 Implementace pro prostorovou indexovací službu

V prostorové indexovací službě jsou vstupními parametry hranice dlaždice, ve formě zeměpisných souřadnic. Odpověď ze služby jsou kompletní nerozřezané geometrie vrstvy, které mají průnik s touto dlaždicí. Tento způsob je ve srovnání s vektorovými dlaždicemi výhodný pro určitý typ dat (mnoho malých polygonů), protože není nutné rekonstruovat geometrii, ale data se mohou okamžitě zobrazit v mapě. Nevýhody se projeví u objektů, které mají velmi velký rozměr, protože se budou přenášet nadbytečná data a některé extrémně velké objekty nemusí být vůbec možné přenést. Příkladem může být mapa zobrazující detailní hranici určitého pozemku v obci, kde budeme chtít zároveň zobrazovat i administrativní hranici kraje, která ale výrazně přesahuje aktuální zobrazenou oblast v mapě.

Vzhledem k zmiňovanému problému byla navržena změna, která umožní využít principu prostorové indexovací služby i pro objekty velkého rozměru.

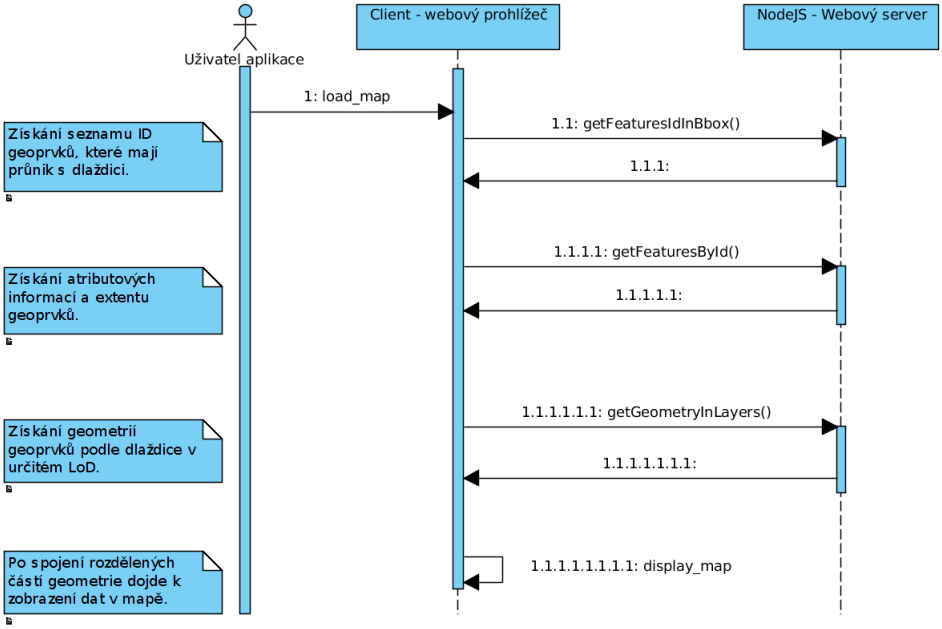
Oproti původní službě byly navrženy změny, které si kladou za cíl zvýšit výkonnost aplikace. Byl navržen postup, kdy se budou přenášet původní nerozdělené geometrie pro objekty, které nemají ve srovnání s plochu dlaždice výrazně větší plochu. V případě objektu

[9] http://couchdb.apache.org/

s velkou rozlohou oproti rozloze dlaždice dojde k rozdělení této geometrie obdobně jako při postupu vektorových dlaždic. Tato navržená změna umožní přenášet velkou část dat v původní formě a nebude již nutné tato data zpětně rekonstruovat. Výhodou této upravené prostorové indexovací služby je to, že není nutné provádět tolik výkonově náročných rekonstrukcí geometrií na straně klienta jako u vektorových dlaždic. Zároveň je možné přenášet polygony libovolné velikosti, což u původní prostorové indexovací služby nebylo možné.

6.8.1 Popis komunikace pro spatial indexing service

Implementace načítání dat prostřednictvím prostorové indexovací služby je ve srovnání s vektorovými dlaždicemi komplikovanější. Důvodem je způsob získávání dat. Samotná data jsou totiž získávána v několika GET požadavcích, které jsou na sebe navázány. Nejdříve proto bude zjednodušeně popsána komunikace aplikace se serverem (diagram na obr. č. 6.4) a následně bude popsána samotná implementace.



Obrázek 6.4: Sekvenční diagram prostorové indexovací služby.

První požadavek je **/getFeaturesIdInBbox**, který se posílá s parametrem na požadované vrstvy a hranicí dlaždice. Tento požadavek na straně serveru zjistí a vrátí všechny identifikátory prvků, které mají průnik s dlaždicí. Navíc v tomto požadavku jsou rozlišovány prvky, které bude nutné rozdělit podle hranice dlaždice nebo bude možné geometrie poslat kompletní. Na základě odpovědi, která se vrátí na klienta, je zkontrolována paměť,

která obsahuje identifikátory už načtených prvků. Tato paměť je vedena samostatně pro každou vrstvu a pro každý LoD. Navíc jsou identifikátory v každém LoD rozděleny na prvky, které se načítají kompletní a prvky, které se načítají rozdělené. Toto dělení je nutné evidovat, aby bylo možné následně určit ty prvky, které jsou tvořeny několika částmi. Je nutné je načítat v každé dlaždici, s kterou mají průnik oproti těm, které jsou načteny jednou za všechny dlaždice v LoD.

Navrácený seznam identifikátorů geoprvků je porovnán s pamětí evidující již načtené geoprvky a pro ty, které nejsou v žádné z pamětí se volá požadavek **/getFeaturesById**. Pro geoprvky posílané po částech, které už mají alespoň část načtenou z jiné dlaždice, se volá požadavek **/getGeometryInLayers**.

Požadavek **/getFeaturesById**, vrací atributová data a extent geoprvku v nejdetailnějším LoD. Extent se využívá později pro interní účely knihovny OpenLayers. Geoprvky jsou vraceny ve formátu GeoJSON. Je důležité poznamenat, že data neobsahují samotnou geometrii, ale jenom atributy, protože geometrie je získávána až v dalším požadavku. Odpověď ze serveru je zpracována, tak že všechny prvky jsou přidány do instance ol.source.MultiLevelVector podle identifikátoru vrstvy. Navíc je pro všechny geoprvky volán poslední požadavek **/getGeometryInLayers**, který posílá seznam identifikátorů a souřadnice hranice dlaždice. Server na tento požadavek vrací na GeoJSON s geometriemi podle LoD.

6.8.2 Klientská část pro spatial indexing service

V základním modulu aplikace (index.js) je ve srovnání s vektorovými dlaždicemi rozdíl pouze v použitém loaderu pro fiktivní vrstvu, která se stará o načítání dat. Namísto VectorTileLoader je zde použit **SpatialIndexLoader**. Kromě jiného loaderu není v modulu žádná jiná změna, přičemž konfigurace mapy a vrstev je pro vektorové dlaždice i prostorovou indexovací službu identická.

Modul SpatialIndexLoader je ve srovnání s VectorTileLoaderem velmi rozdílný, protože komunikace se serverem probíhá jinými požadavky, které jsou popsané v předešlé sekci.

Načtené rozdělené geometrie jsou dočasně uloženy v paměti modulu MergeTools, který tyto geometrie po načtení všech dat v aktuálně zobrazené oblasti mapy spojí do původní podoby. Spojené geometrie jsou zpětným voláním předány do SpatialIndexLoaderu. Nakonec jsou geometrie přidány do instancí ol.source.MultiLevelVector obdobně jako u vektorových

dlaždic.

Komunikace se serverem

Základní dotaz **/getFeaturesIdInBbox**, který zjišťuje seznam identifikátorů pro geoprvky mající průnik s dlaždicí je na ukázce kódu č. 6.3. Parametry dotazu jsou předávané v objektu data, jehož obsah je: db (jméno databáze), geom (jméno geometrického sloupce - závisí na LoD), level (aktuálním zoomu mapy), clipBig (hodnota true = v případě velkých geometrií dojde k rozřezání podle dlaždice, false = všechny geometrie se posílají kompletní), extent (souřadnice hranice dlaždice) a layers (seznam vrstev).

```
1  $.ajax({
2    url: 'http://localhost:9001/se/getFeaturesIdInBbox',
3    type: "get",
4    data: {
5      "db": 'vfr',
6      "geom": 'geometry_15',
7      "idColumn": 'id',
8      "level": 15,
9      "clipBig": true,
10     "requestType": "getFeaturesIdInBbox",
11     "extent": [16.721, 49.145, 16.732, 49.152],
12     "layers": ['obce', 'okresy', 'kraje', 'katastralniuzemi']
13   },
14   datatype: 'json',
15   success: function(data){....},
16   error:function(er){...}
17 });
```

Ukázka kódu 6.3: Dotaz /getFeaturesIdInBbox pro získání identifikátorů geoprvků.

Na dotaz se ze serveru vrací odpověď ve formě JSON (ukázka č. 6.4). Parametry pod klíčem layers obsahují pro každou vrstvu seznam identifikátorů na geoprvky, které mají průnik s hranicí dlaždice. Zároveň každý identifikátor nese informaci o tom, zda bude nutné pro požadovaný LoD geometrii rozdělit (hodnota 1) nebo se bude geometrie přenášet kompletní (hodnota 0).

```
{
  "layers": {
    "obce": {"57": 0, "305": 0, "529": 0 },
    "okresy": { "54": 1, "55": 1 },
    "kraje": { "7": 1, "8": 1 },
    "katastralniuzemi": { "66": 0, "363": 0}
  },
  "extent": ["16.721","49.145","16.732","49.152"],
  "level": "15"
}
```

Ukázka kódu 6.4: Odpověď na dotaz /getFeaturesIdInBbox.

Na základě získaného seznamu identifikátorů a porovnání s pamětí na straně klienta je volán požadavek **/getFeaturesByIdInLayers** nebo **/getGeometryInLayers**.

Dotaz **/getFeaturesByIdInLayers** získává atributové informace o geoprvku a jeho extent. Volání požadavku je na ukázce kódu č. 6.5. Posílají se parametry v podobě atributů: db (jméno databáze), geom (jméno geometrického sloupce - závisí na LoD), level (aktuálním zoomu mapy), extent (souřadnice hranic dlaždice) a idsInLayer (obsahuje seznam vrstev a pro každou vrstvu seznam identifikátorů geoprvků v dlaždici).

```
$.ajax({
  url: "http://localhost:9001/se/getFeaturesByIdInLayers",
  type: "get",
  data:  {
    "db": "vfr",
    "geom": "geomrow_12",
    "idColumn": 'ogc_fid',
    "level": 12,
    "requestType": "getFeaturesByIdInLayers",
    "extent": [15.908, 48.980, 15.996, 49.037],
    "idsInLayer": {
      "obce": "280",
      "katastralniuzemi": "337"
    }
  },
  datatype: 'json',
  success: function(data, status, xhr){...},
  error:function(er){...}
});
```

Ukázka kódu 6.5: Dotaz /getFeaturesByIdInLayers.

Server vrací odpověď jako objekt, obsahující seznam identifikátorů, atribut level a samotné geoprvky ve formátu GeoJSON (ukázka kódu č. 6.6), které obsahují atributové informace: id (identifikátor geoprvku), extent (extent geoprvku negeneralizované geometrie), layer (název vrstvy z které byl geoprvek získán). Geoprvky z odpovědi jsou přidány do

příslušné instance ol.source.MultiLevelVector (na základě údaje z atributu layer) a zároveň je pro všechny geoprvky odeslán požadavek na získání geometrie **/getGeometryInLayers**.

```
{
  "FeatureCollection": {
    "type": "FeatureCollection",
    "features": [
    {
      "type": "Feature",
      "properties": {
        "id": 280,
        "extent": [1773487.09534634, 6276742.51324453,
           1780613.42818589, 6284599.69162212],
        "layer": "obce"
      },
      "geometry": {"type": "Polygon", "coordinates": []}
    },
    {...}
    ]
  },
  "ids": {"obce": " '280'", "katastralniuzemi": " '337'" },
  "level": "12"
}
```

Ukázka kódu 6.6: Odpověď na dotaz /getFeaturesByIdinLayers.

Dotaz **/getGeometryInLayers** (ukázka kódu č. 6.7) získává ze serveru samotné geometrie geoprvků.

```
 $.ajax({
  url: "http://localhost:9001/se/getGeometryInLayers",
  type: "get",
  data:  {
    "db": "vfr",
    "geom": "geomrow_12",
    "idColumn": 'ogc_fid',
    "level": 12,
    "requestType": "getGeometryInLayers",
    "extent": [15.908, 48.980, 15.996, 49.037],
    "clipBig": true,
    "idsInLayer": {
      "obce": "280",
      "katastralniuzemi": "337"
    }
  },
  datatype: 'json',
  success: function(data, status, xhr){...},
  error:function(er){...}
});
```

Ukázka kódu 6.7: Dotaz /getGeometryInLayers pro získání geometrií geoprvků.

Odpověď (ukázka kódu č. 6.8) na požadavek je téměř identická jako pro požadavek **/getFeaturesByIdInLayers**. Rozdílný je obsah prvku (feature) v kolekci (FeatureCollection). Geoprvek je přenášen s geometrií, která je generalizována pro požadovaný LoD. Ve vlastnostech prvku (properties) je kromě identifikátoru prvku a názvu vrstvy přidán název geometrického sloupce (geomRow) a údaj rozlišující jestli je geometrie původní nebo rozdělená (vlastnost originalGeom).

Příchozí odpověď ze serveru je na klientovi zpracována. Geoprvky s kompletní geometrií jsou přidány do mapy a geoprvky s rozdělenou jsou dočasně uloženy v paměti modulu MergeTools. Modul po načtení všech dlaždic uložené geometrie spojí. Spojené geometrie jsou zpětným voláním předány do SpatialIndexLoaderu. SpatialIndexLoaderu přidá zrekonstruovanou geometrii jako atributovou vlastnost již existujícímu geoprvku (ol.feature) v příslušném ol.source.MultiLevelVector. Vlastnost je v ol.feature přidána pod klíčem, jehož název určuje pro jaký LoD bude geometrie použita (geometry1 až geomentryN).

```
{
  "FeatureCollection": {
    "type": "FeatureCollection",
    "features": [
      {
        "type": "Feature",
        "properties": {
          "id": 337,
          "layer": "katastralniuzemi",
          "geomRow": "geometry_12",
          "original_geom": true
        },
        "geometry": {
          "type": "Polygon",
          "coordinates": [
            [
              [17.019178, 49.096774],...
            ]
          ]
        }
      },...
    ]
  },
  "ids": {
   "obce": "280",
   "katastralniuzemi": "337"
 },
 "level": "12"
}
```

Ukázka kódu 6.8: Odpověď na dotaz /getGeometryInLayers.

6.8.3 Serverová část pro spatial indexing service

Na straně serveru je připraveno směrování v modulu server.js. Směrování pro GET požadavky je připraveno na adresách /getFeaturesIdInBbox, /getFeaturesByIdInLayers a /getGeometryInLayers. Požadavky, které jsou přijaty a zpracovávány ve stejnojmenných modulech. V každém modulu dochází k dotazování do Postgis databáze s daty RUAIN. Z těchto SQL dotazů jsou vytvořeny odpovědi, jejichž ukázky byly popsány v minulé kapitole.

7 MĚŘENÍ VÝKONNOSTI APLIKACE

V této kapitole budou srovnány základní charakteristiky, které je možné měřit na straně klienta. Konkrétně bude porovnána rychlost načtení dat, objem přenesených dat a rychlost rekonstrukce rozdělených částí geometrie.

7.1 Postup měření výkonnosti

Pro výkonnostní porovnání vektorových dlaždic i prostorové indexovací služby bylo vytvořeno **automatizované měření**, které simuluje uživatelskou interakci s mapou. Charakteristiky doby trvání načítání dat a doby trvání rekonstrukce geometrií jsou uvedeny v absolutních hodnotách (milisekundy), ale je spíše vhodné je vnímat jako relativní srovnání mezi oběma metodami.

Měření probíhalo na počítači, který měl spuštěný lokální webový i databázový server. Z toho důvodu došlo k minimalizaci doby přenosu dat mezi serverem a klientem, které by při přenosu dat přes internet při reálném nasazení aplikace mohlo mít významný vliv na rychlost aplikace. Důvodem pro testování na lokálním serveru je snaha eliminovat proměnlivou rychlost internetu, která by zkreslovala výsledky měření. Měření bylo opakováno pro každou veličinu minimálně desetkrát, měření s extrémními hodnotami byla z výsledků eliminována.

Získané proměnné jsou **velikost přenesených dat**, **doba kompletního načtení dat ze serveru na klienta** a **doba spojování rozdělených geometrií** v modulu mergeTools. Měřenými interakcemi mapy jsou inicializace mapy při zoomu 12, posun mapy doleva o velikost mapového pole, přiblížení mapy na zoom 14, oddálení na zoom 10 a nakonec oddálení na zoom 9.

Velikost mapového okna při měření byla 1800 (šířka) na 900 (výška) pixelů. Během jednoho měření bylo ve všech pěti interakcích načteno celkem 260 mapových dlaždic o rozměru 256 na 256 pixelů. Pro prostorovou indexovací službu se data za 260 dlaždic

načetla celkem v 685 dotazech na server.

V rámci jednotlivých interakcí byla načítána data, která při změně zoomu změnila charakter (např. z velkých polygonů na malé). Z toho důvodu bylo nutné definovat, jaký typ dat byl při jednotlivých interakcích typický (tab. č. 7.1). V rámci automatizovaného testování byla měření provedena na vrstvě obce, katastrální území, okresy a kraje z databáze RUIAN. Zmenšené náhledy mapy s načtenými daty je možné nalézt v příloze č.1.

Interakce	**Popis dat**
Iniciální načtení	Větší množství menších polygonů, dále velké polygony pouze pro okresy a kraje.
Posun doleva	Větší polygony pro okresy a kraje, několik menších polygonů vrstvy obce.
Přiblížení	Pouze velké polygony krajů a okresů.
Oddálení 3x	Velké množství malých polygonů pro obce a velmi malých pro katastrální území, střední polygony pro okresy a velké polygony pro kraje.
Oddálení 1x	Velmi vysoký počet velmi malých polygonů pro obce a katastrální území, menší až střední pro okresy a střední polygony pro vrstvu krajů.

Tabulka 7.1: Charakteristika dat pro měření interakcí mapy.

Pro **vektorové dlaždice** byly provedeny celkem čtyři varianty měření, měření podle formátu přenášených dat (GeoJSON nebo TopoJSON) a měření podle zdroje dat (originální databáze PostgreSQL nebo cache v databázi CouchDB).

Pro **prostorovou indexovací službu** byly testovány různé varianty pro maximální velikost geometrie, která nebude rozřezána podle hranice dlaždice. Parametr, který určuje pro geometrii, jestli bude přenášena kompletní nebo rozdělená, je určen jako maximální plošná velikost polygonu, který se bude přenášet ve své původní nerozřezané podobě.

Hodnotu parametru nebylo možné stanovit vzhledem ke všem úrovním přiblížení mapy stejně, ale velikost musela být určena pro každý LoD samostatně. Výpočet této kritické hodnoty je z toho důvodu vztažen k násobku plochy dlaždice (například hodnota 2x znamená, že polygon zůstane nerozdělen, pokud jeho plocha je menší než dvojnásobek plochy dlaždice). Parametr maximální velikosti polygonu byl testován s hodnotami 0.01x, 2x, 4x, 8x a 80 000x.

Hodnota 0.01 simuluje stav, kdy téměř všechny polygony by měly být rozřezány podle hranice dlaždice. Předpokladem je, že výsledky měření tohoto parametru by měly být velmi blízké měření pro vektorové dlaždice, čímž by měla být ověřena úspěšnost implementace

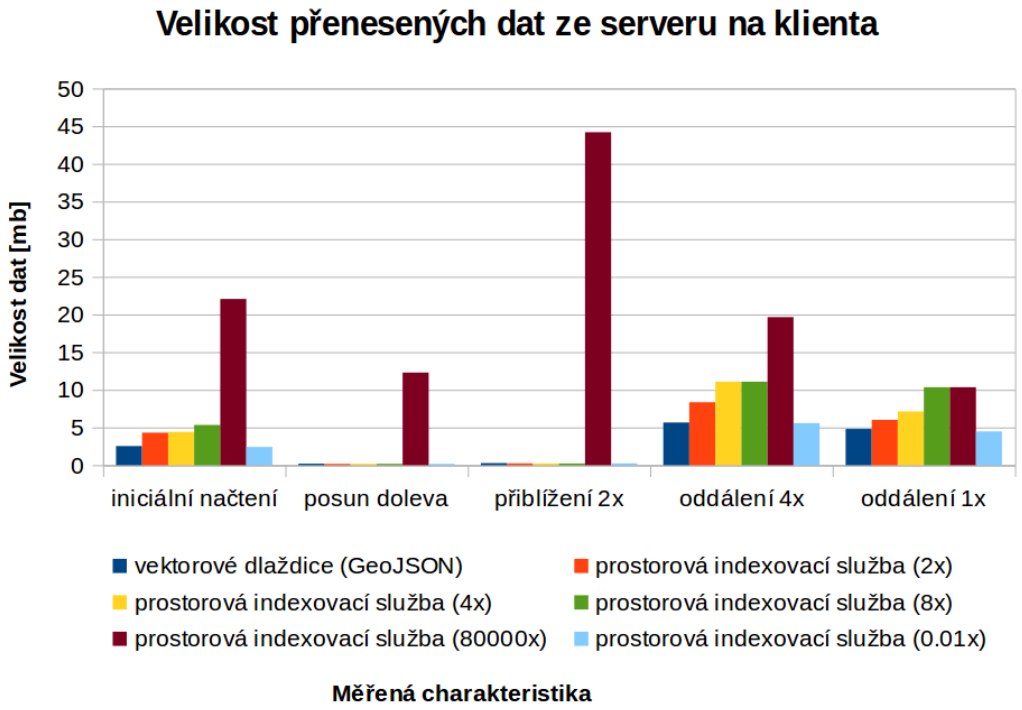
automatizovaného měření výkonnosti.

Hodnota 80 000 naopak simuluje původní prostorovou indexovací službu podle Gaffuriho [4]. Při původním návrhu se posílají vždy kompletní data, která nejsou nikdy rozdělena. Porovnání měření s hodnotou parametru 2x, 4x nebo 8x by mělo vyjádřit, jestli navržená úprava pro rozdělování velkých geometrií je přínosná.

7.2 Výsledky měření výkonnosti

Objem přenesených dat ze serveru na klienta

Na obr. č. 7.1 je graf zobrazující srovnání velikosti přenášených dat v megabajtech, údaje jsou rozdělené pro každou interakci i metodu. Pro všechna měření platí, že data jsou posílána ve formátu GeoJSON přímo z databáze PostgreSQL.



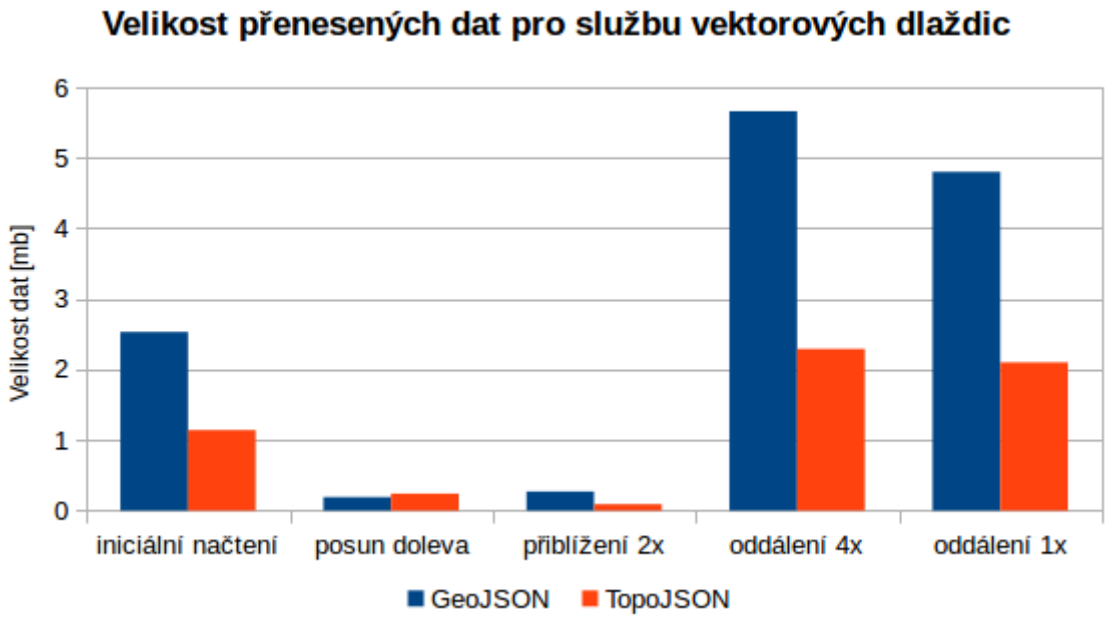
Obrázek 7.1: Velikost přenesených dat.

Předpokladem bylo, že původní návrh prostorové indexovací služby (parametr 80000x), bude mít výrazně vyšší naměřené hodnoty, než nová prostorová indexovací služba, která velké polygony rozděluje. V grafu je možné vidět, že velikost přenesených dat pro původní metodu je kromě poslední interakce (oddálení 1x) vždy minimálně dvakrát větší. Důvodem, proč při poslední interakci jsou hodnoty podobné ostatním je to, že mapa zobrazuje

téměř celé území, tedy při téměř všech metodách docházelo k přenosu kompletních dat. Nedocházelo tedy téměř k načítání dat, která by byla mimo zobrazenou oblast mapy.

Dalším předpokladem bylo, že pro vektorové dlaždice bude přenesená velikost dat menší, než pro jakoukoliv z měřených metod prostorové indexovací služby. Menší velikost by měla být způsobena nižším počtem dotazů na server, protože pro vektorové dlaždice se volá vždy pouze jeden požadavek, ale pro prostorovou indexovací službu se jedná až o tři dotazy. V grafu je možné znovu vidět, že velikost dat pro vektorové dlaždice dosahuje vždy minimálních hodnot. Výjimkou je poslední interakce oddálení mapy, kdy zřejmě dochází v určitých dlaždicích k chybám při dělení geometrie.

Na obr. č. 7.2 je srovnání velikosti přenesených dat pro formáty GeoJSON a TopoJSON při využití vektorových dlaždic. Z grafu je zřejmé, že využití formátu TopoJSON má ve srovnání s formátem GeoJSON pro testovaná data velký přínos, protože při některých naměřených charakteristikách byl objem přenesených dat méně než poloviční. Během jednoho měření byla celková velikost přenesených dat pro formát GeoJSON 13,45 MB a pro TopoJSON pouze 5,86 MB.

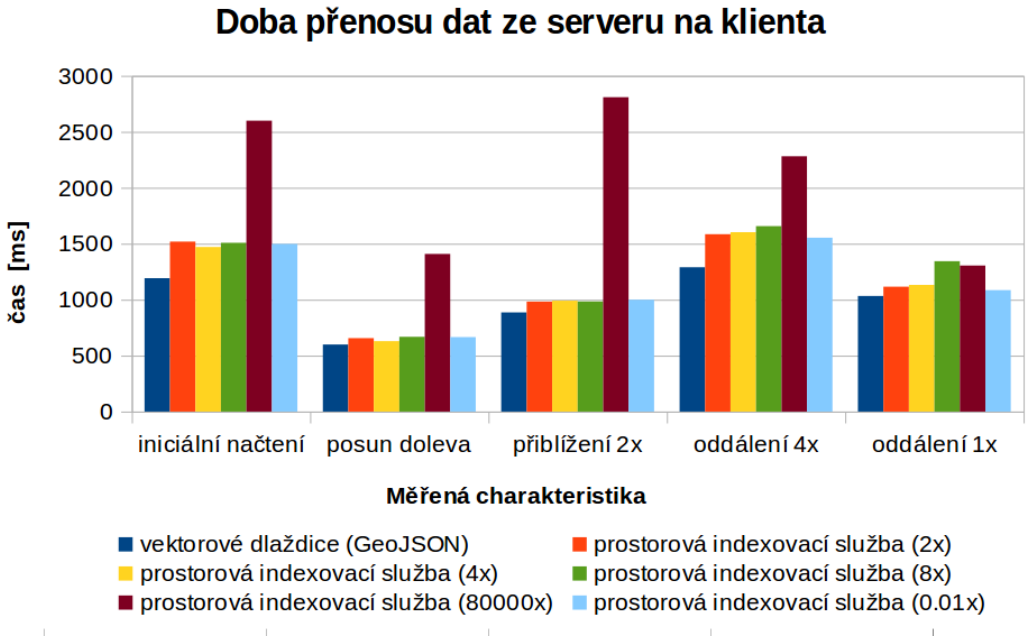


Obrázek 7.2: Velikost přenesených dat pro službu vektorových dlaždic.

Čas potřebný k přenosu dat ze serveru na klienta

Na obr. č. 7.3 je graf zobrazující srovnání doby přenosu dat ze serveru na klienta. Naměřené časy přenosu korespondují s naměřenými hodnotami velikosti přenesených dat. Nejrychleji byly přenášeny data pro vektorové dlaždice, potom pro upravenou prostorovou indexovací službu a nejdelší časy načítání se naměřily u původní indexovací služby.

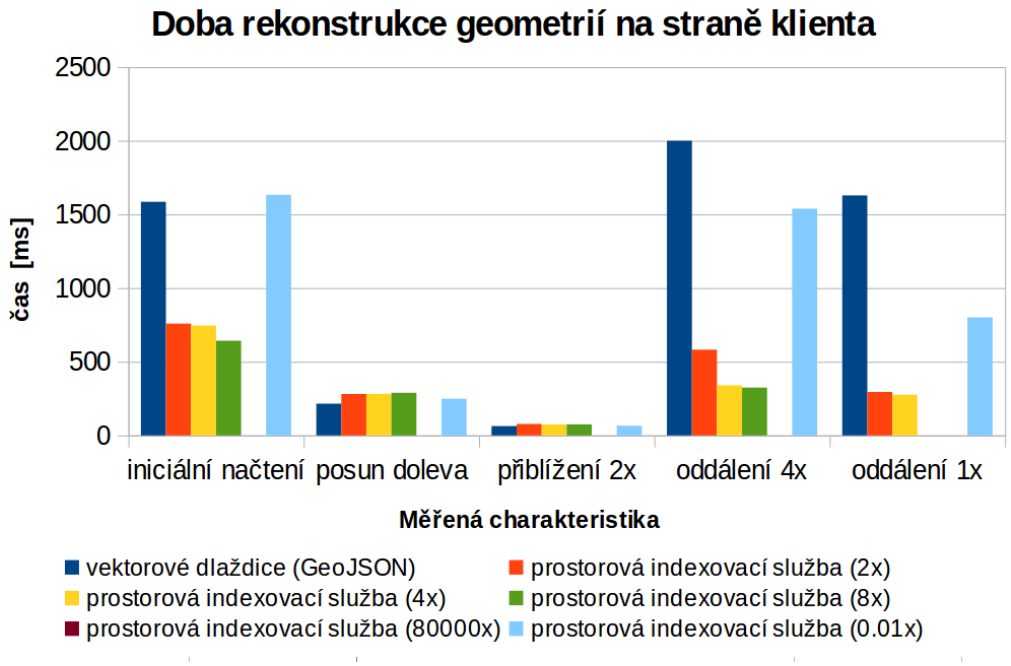
Při detailním srovnání časů s objemem přenesených dat, můžeme pozorovat několik rozdílů. Na grafu velikosti přenesených dat byla pro měření prostorové indexovací služby s parametry 2x, 4x a 8x vidět narůstající velikost dat, s tím jak velké mohou být maximální velikosti polygonů. Tento jev se ale neprojevuje vždy pro dobu načítání. Hodnota parametru 4x se pro iniciální načtení a posun doleva ukázala jako nejvýhodnější, pro přiblížení a oddálení jsou výsledky skoro stejné jako s parametrem 2x. Naopak parametr s hodnotou 8x se ukazuje jako nejméně výhodný z hlediska doby načítání. Je nutné poznamenat, že naměřené výsledky byly dosaženy na lokálním serveru, tedy přenos nebyl ovlivněn rychlostí internetu, proto velikost přenášených dat bude při reálném nasazení hrát ještě významnější roli.



Obrázek 7.3: Doba přenosu dat ze serveru na klienta.

Čas potřebný k rekonstrukci geometrií

Na obr. č. 7.4 je graf zobrazující srovnání času rekonstrukce rozdělených geometrií do původní podoby. V grafu je vidět, že maximální doba rekonstrukce geometrií se naměřila u vektorových dlaždic a prostorové indexovací služby s parametrem 0.01x, tedy kdy docházelo k dělení těch nejmenších polygonů. Naopak nulové hodnoty rekonstrukce dat jsou naměřeny pro původní návrh prostorové indexovací služby, protože nedocházelo k dělení geometrií. Pro další naměřené hodnoty prostorové indexovací služby je možné vidět trend, který koreluje s maximální povolenou velikostí polygonu. To znamená, že čím větší maximální velikost, tím kratší doba rekonstrukce dat, protože bylo méně polygonů, kterým se geometrie rozdělila.

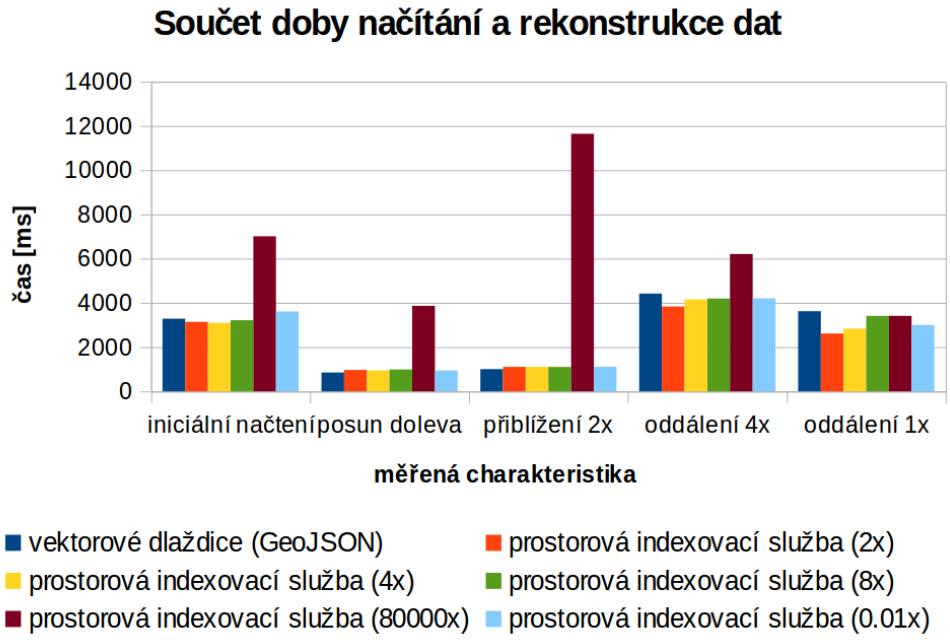


Obrázek 7.4: Doba rekonstrukce geometrií.

Celková doba pro zobrazení kompletní mapy

Na obr. č. 7.5 je graf zobrazující součet doby přenosu dat a doby rekonstrukce geometrie, tedy kompletní čas pro načtení mapy v prohlížeči. Naměřené časy mají rozdílné výsledky pro jednotlivé měřené interakce, protože každá interakce načítá data jiného charakteru. Pro data, která načítají pouze velké polygony (posun doleva a přiblížení 2x) byly nejrychlejší vektorové dlaždice, naopak větší množství menších polygonů (oddálení 4x a oddálení

1x) bylo výrazně dříve zobrazeno v mapě prostřednictvím prostorové indexovací služby s parametry 2x, 4x, 8x.



Obrázek 7.5: Součet doby načítání a rekonstrukce dat.

Měření tedy ukázalo, že pro menší polygony je vhodné využít prostorovou indexovací službu, protože nebude nutné čekat na rekonstrukci geometrie a mapa bude moci být uživateli dříve zobrazena. Naopak pro velké polygony je výhodnější použít vektorové dlaždice, kde navíc dojde ke snížení počtu požadavků odeslaných na server.

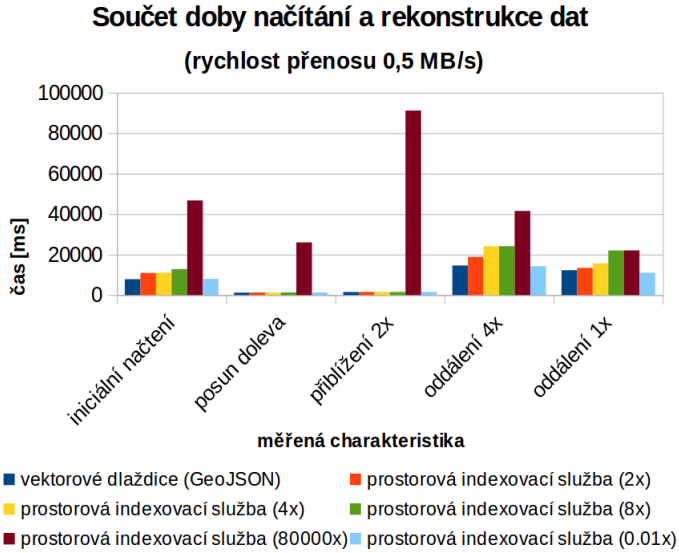
Doba pro zobrazení mapy při zohlednění rychlosti přenosu dat na internetu

Problémem měření na lokálním serveru je, že nebere do úvahy rychlost internetu, která vzhledem k velikosti dat bude mít výrazný vliv na naměřené hodnoty. Z tohoto důvodu byla nasimulována rychlost internetu přičtením času, který by byl potřebný k přenosu určité velikosti dat při určité rychlosti. Tento výpočet založený na naměřených datech velikosti přenesených dat (obr. č.7.1) byl pro každou měřenou charakteristiku i metodu připočten k hodnotám součtu doby načítání a rekonstrukce. Součtem vznikly výpočty pro rychlost přenosu dat 0,5 Mb/s (obr. č.7.6 a č.7.7), 1 Mb/s (obr. č.7.8) a 5 Mb/s (obr. č.7.9).

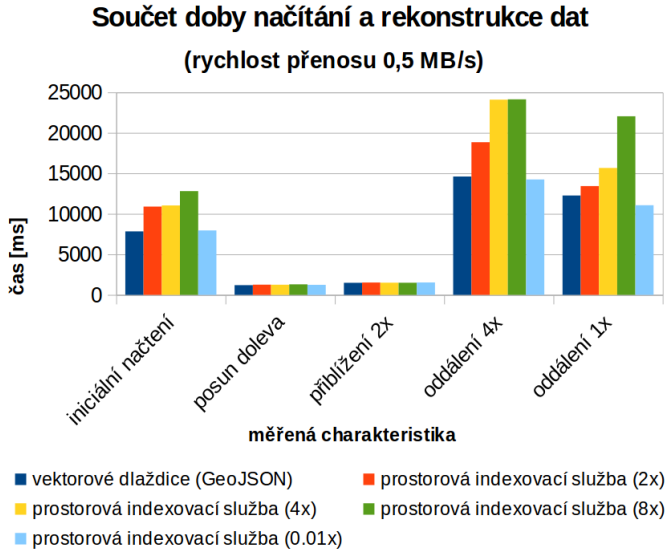
Z výsledků měření doby rekonstrukce geometrie a načítání dat ovlivněné rychlostí internetu, je vidět velký vliv objemu přenesených dat na celkovou dobu zobrazení mapy.

Zejména pro původní metodu prostorové indexovací služby, kde by se podle výpočtů mapa zobrazila za 20 až 90 vteřin.

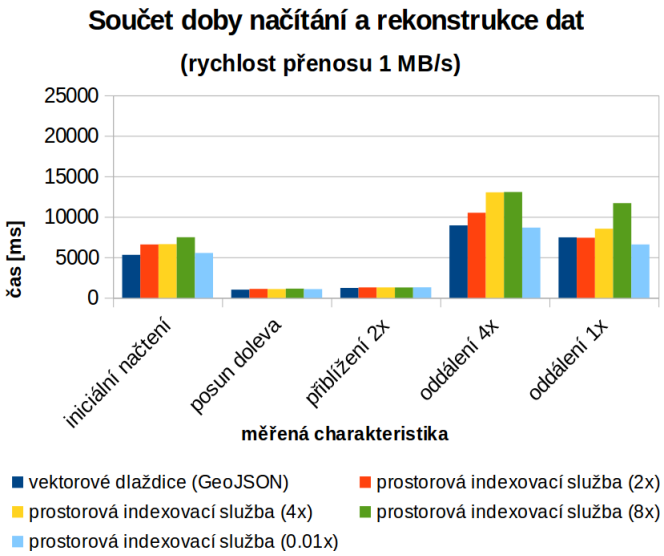
Mezi výsledky měření jednotlivých rychlostí je vidět, že každé zpomalení internetu má vliv na rychlost a proto metody, které přenáší větší množství dat, dosahují výrazně horších výsledků. Při pomalé rychlosti internetu (0,5 a 1 MB/s) bylo nejrychlejší zobrazení mapy dosaženo metodou vektorových dlaždic, která přenáší nejmenší objem dat i přes to, že doba rekonstrukce geometrií je pro ni nejdelší.



Obrázek 7.6: Součet doby přenosu a rekonstrukce geometrií při rychlosti 0,5 MB/s (včetně původní prostorové indexovací služby).



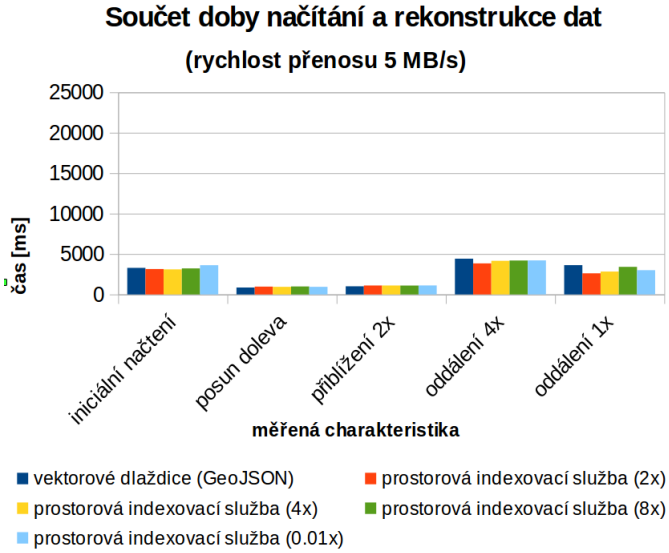
Obrázek 7.7: Součet doby přenosu a rekonstrukce geometrií při rychlosti 0,5 MB/s.



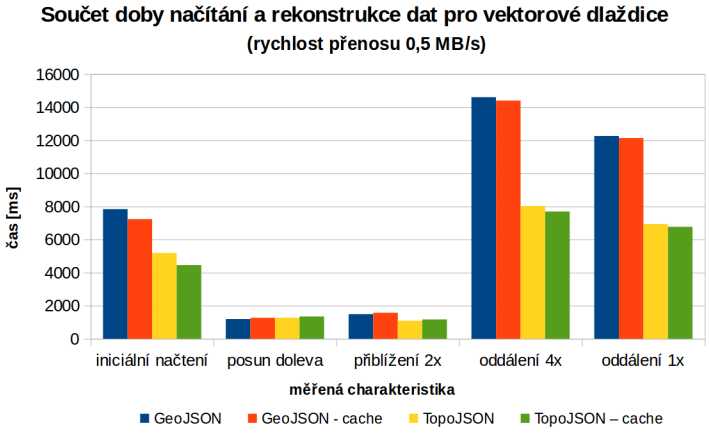
Obrázek 7.8: Součet doby přenosu a rekonstrukce geometrií při rychlosti 1 MB/s.

Na obr. č. 7.10 jsou výsledky měření optimalizace pro vektorové dlaždice prostřednictvím cache v CouchDB a úsporného formátu TopoJSON. Během měření bylo zjištěno, že vliv cache na zkrácení doby přenosu dat je při využití lokálního serveru minimální. Pokud by měření probíhalo na reálném serveru, který by byl vytížený větším počtem uživatelů aplikace, potom by naměřené hodnoty mohly být velmi rozdílné od hodnot naměřených na lokálním serveru. Dalším benefitem cachování je předpokládané snížení náročnosti na výkon serveru, které ale nebylo ověřeno v praxi.

Výsledky měření pro datové formáty GeoJSON a TopoJSON ukázaly, že se snížením velikosti dat se výrazně zkrátí čas potřebný pro přenos dat, z čehož vyplývá, že použití formátu TopoJSON zkrátí pro určité případy dobu kompletního načtení mapy téměř o polovinu. Zároveň bylo měřením ověřeno, že náročnost konverze z formátu TopoJSON do GeoJSON, vyžadovaném v modulu MergeTools, je ve srovnání s rekonstrukcí geometrií zanedbatelná a neovlivní tedy celkový čas, za který je výsledná mapa zobrazena.



Obrázek 7.9: Součet doby přenosu a rekonstrukce geometrií při rychlosti 5 MB/s.



Obrázek 7.10: Součet doby přenosu a rekonstrukce geometrií při rychlosti 0,5 MB/s pro službu vektorových dlaždic.

7.3 Závěr z měření výkonnosti

Z naměřených výsledků je možné vidět, že navržená změna prostorové indexovací služby, která bude rozdělovat geometrie má uplatnění, a umožní využít tuto metodu pro data polygonů rozdílných velikostí, což s původní službou nebylo možné.

Z výsledků je patrné, že rozhodující roli v aplikaci má rychlost internetu a velikost dat, protože s každým snížením rychlosti internetu se metody přenášející větší objem dat stávají méně použitelné. Pro pomalejší připojení k internetu se jako nejvýhodnější ukázala metoda vektorových dlaždic.

Při použití metody prostorové indexovací služby je vhodné určit maximální velikost polygonu tak, aby docházelo k přenosu kompletních geometrií pro prvky, jejichž plocha není výrazně větší než plocha dlaždice.

Datový formát TopoJSON umožní výrazně snížit objem přenášených dat ve srovnání s formátem GeoJSON, což výrazně zkrátí dobu nutnou pro přenos při nižších rychlostech internetu.

8 ZÁVĚR

Cílem diplomové práce bylo popsat principy a metody přenosu vektorových dat ze serveru na klienta prostřednictvím techniky vektorových dlaždic. Tato technika byla dále porovnána s ostatními metodami přenosu vektorových dat, s hlavním důrazem na metodu prostorové indexovací služby. Kromě samotných technik přenosu byly kladen důraz i na možnosti optimalizace přenosu dat v podobě generalizace nebo volby vhodného datového formátu, konkrétně srovnání formátu GeoJSON, TopoJSON a Mapbox Vector Tile.

Na základě zjištěných informací v teoretické části byla navržena a implementována webová aplikace zobrazující data z RUIAN. Výsledkem je aplikace využívající pro přenos dat techniku vektorových dlaždic i prostorové indexovací služby. Přínosem aplikace je implementace vektorových dlaždic s rekonstrukcí geometrií na straně klienta, což v současnosti není běžné. Ukázka aplikace je dostupná na http://ruian.ludvikadamec.cz/.

Po vytvoření základní verze aplikace došlo k navržení optimalizací pro obě techniky a jejich implementaci. Pro vektorové dlaždice byla vytvořena cache ukládající vygenerované dlaždice do CouchDB databáze, a navíc byl testován namísto formátu GeoJSON topologický formát TopoJSON. Pro prostorovou indexovací službu byl oproti standardní metodě přenosu kompletních geometrií naprogramován alternativní přístup. Ten na základě relativní velikosti plochy geometrie ve srovnání s plochou dlaždice rozhodne, kdy bude geometrie odeslána kompletní a kdy se rozdělí a bude se postupovat jako při technice vektorových dlaždic.

Nad aplikací bylo nakonec provedeno výkonnostní měření, které srovnávalo jednotlivé techniky a optimalizace z hlediska objemu přenesených dat, doby přenosu dat ze serveru na klienta a doby rekonstrukce dlaždic. Z výsledků vyplynulo, že navržené úpravy pro prostorovou indexovací službu jsou významným přínosem, protože velikost přenášených dat se významně zmenší, čímž dojde k velkému zrychlení aplikace. Navíc při této úpravě je možné nasadit tuto techniku pro data i s větší velikostí polygonů, což předtím nebylo možné. Pro vektorové dlaždice bylo na základě výsledků zjištěno, že využití formátu TopoJSON je

velkým přínosem, protože dojde k výraznému snížení objemu dat. Dalším zjištěním bylo, že nejvýznamnějším kritériem určujícím výkonnost aplikace je velikost dat, kterou je nutné minimalizovat zejména pokud se předpokládá omezení rychlosti přenosu dat přes internet. Na základě těchto měření bylo zjištěno, že pro pomalejší připojení je většinou výhodnější využít techniku vektorových dlaždic, protože objem přenesených dat ze serveru na klienta je ve srovnání s prostorovou indexovací službou menší.

Do budoucna by bylo přínosem se zaměřit na další optimalizaci aplikace, ať už po stránce uložení dat, kde by bylo vhodné otestovat binární formát Mapbox Vector Tile nebo po stránce optimalizace klientské části aplikace, zejména v procesu rekonstrukce geometrií. Zde by bylo možné se zaměřit na samotnou operaci spojení geometrií, která v této práci byla řešena knihovnou třetí strany. Další možnou optimalizací by mohlo být zobrazení nezrekonstruovaných geometrií, jako je běžné v současnosti v jiných aplikacích, což by mělo zlepšit plynulost interakce s mapou. Rekonstrukce geometrií by se prováděla až v případě, že uživatel potřebuje například pro interakci s geoprvkem geometrii kompletní. Zajímavou možností optimalizace by bylo využití technik vícevláknového zpracování, kdy by se náročná rekonstrukce geometrií přesunula do druhého vlákna a byla by oddělena od samotného uživatelského rozhraní. Dalším směrem pro vývoj by mohla standardizace metody vektorových dlaždic a prostorové indexovací služby při využití Web Processing Service [60], která by zpracovávala data získaná například službou WFS.

SEZNAM POUŽITÉ LITERATURY

[1] DEMERS, Michael N. Fundamentals of geographic information systems. 4th ed. Hoboken, NJ: Wiley, c2009, xiii, 443 p. ISBN 0470129069.

[2] VESELÝ, Martin. *TopoXML - výměnný formát topologie vektorových dat* [online]. Brno, 2007 [cit. 2015-09-03]. Disertační práce. Masarykova univerzita, Přírodovědecká fakulta. Vedoucí práce prof. RNDr. Milan Konečný, CSc. Dostupné z: http://theses.cz/id/wy2wmu/.

[3] CORCORAN, P, P MOONEY, M BERTOLOTTO a A WINSTANLEY. View-and scale-based progressive transmission of vector data. Computational Science and Its Applications- ICCSA 2011. Springer, 2011, 51-62.

[4] GAFFURI, Julien. Toward web mapping with vector data. In: Geographic Information Science. Springer Berlin Heidelberg, 2012. p. 87-101.

[5] Open Geospatial Consortium. Web Map Service Specification version 1.3.0. Open Geospatial Consortium [online]. 2006 [cit. 2015-09-03]. Dostupné z: http://portal.opengeospatial.org/files/?artifact_id=14416

[6] Open Geospatial Consortium. OpenGIS Web Map Tile Service Implementation Standard 1.0.0 [online]. [cit. 2015-09-03]. Dostupné z: http://portal.opengeospatial.org/files/?artifact_id=35326

[7] Open Geospatial Consortium. OpenGIS Web Feature Service 2.0 Interface Standard [online]. [cit. 2016-01-03]. Dostupné z: http://portal.opengeospatial.org/files/?artifact_id=39967

[8] GOODCHILD, Michael F. Tiling large geographical databases. Berlin, Heidelberg: Springer Berlin Heidelberg, 1990, s. 135-146. ISBN 9783540469247.

[9] Mapbox Vector Tile Specification. Mapbox [online]. [cit. 2016-01-03]. Dostupné z: http://mapbox.github.io/vector-tile-spec/

[10] Tile Map Service Specification. OSGeo [online]. [cit. 2016-01-03]. Dostupné z: http://wiki.osgeo.org/wiki/Tile_Map_Service_Specification

[11] PETERSON, Michael P. Maps and the internet. Maps and the internet / edited by Michael Peterson. 2005. ISBN 0080449441.

[12] ANTONIOU, V.;Haklay. Tiled vectors: A method for vector transmission over the Web. Web and Wireless Geographical Information Systems - 9th International Symposium, W2GIS 2009, Proceedings [online]. 2009, 5886 LNCS, 56 - 71 [cit. 2016-01-03]. ISBN 3642106005. ISSN 03029743.

[13] GAFFURI, Julien. Improving Web Mapping with Generalization. Cartographica [online]. 2011, 46(2), 83-91 [cit. 2016-01-03]. ISSN 03177173.

[14] Web Mercator. In: Wikipedia: the free encyclopedia [online]. [cit. 2016-05-03]. Dostupné z: https://en.wikipedia.org/wiki/Web_Mercator

[15] BATTERSBY, Sarah E.;Finn. Implications of Web Mercator and Its Use in Online Mapping. Cartographica [online]. 2014, 49(2), 85-101 [cit. 2016-01-13]. DOI: 10.3138/carto.49.2.2313. ISSN 03177173.

[16] TARALDSVIK, Mats, et al. The future of web-based maps: can vector tiles and HTML5 solve the need for high-performance delivery of maps on the web?. 2012.

[17] NORDAN, Robert Patrick Victor. An Investigation of Potential Methods for Topology Preservation in Interactive Vector Tile Map Applications. 2012.

[18] UTFGrid: Specification for UTFGrid [online]. [cit. 2016-05-03]. Dostupné z: https://github.com/mapbox/utfgrid-spec/blob/master/1.3/utfgrid.md

[19] Bl.ocks.org: JavaScript: UTFGrid demonstration [online]. [cit. 2016-01-13]. Dostupné z: http://bl.ocks.org/milkbread/6449317

[20] Mapbox. How Interactivity Works with UTFGrid [online]. [cit. 2016-01-13]. Dostupné z: https://www.mapbox.com/blog/how-interactivity-works-utfgrid/

[21] KRUG, Steve. Web design: nenuťte uživatele přemýšlet!. Brno: Computer Press, 2003. ISBN 80-722-6892-9.

[22] BERTOLOTTO, Michela;Egenhofer. Progressive Transmission of Vector Map Data over the World Wide Web. GeoInformatica [online]. 2001, 5(4), 345-373 [cit. 2016-01-13]. ISSN 13846175.

[23] Slovník VÚGTK: Kartografická generalizace. [online]. [cit. 2015-08-11]. Dostupné z: https://www.vugtk.cz/slovnik/5201_kartograficka-generalizace

[24] MILDORF, Tomáš. MODELOVÁ GENERALIZACE POZEMKOVÉHO DATOVÉHO MODELU [online]. Plzeň, 2012 [cit. 2016-02-03]. Disertační práce. Západočeská univerzita v Plzni, Fakulta aplikovaných věd. Dostupné z: http://theses.cz/id/n7hew9/

[25] CECCONI, A.;Galanda. Adaptive Zooming in Web Cartography. Computer Graphics Forum [online]. 2002, 21(4), 787-799 [cit. 2016-01-13]. ISSN 01677055.

[26] JONES, Christopher B.;Ware. Map generalization in the Web age. International Journal of Geographical Information Science [online]. 2005, 19(8/9), 859-870 [cit. 2016-05-03]. DOI: 10.1080/13658810500161104. ISSN 13658816.

[27] OpenGIS Geography Markup Language (GML) Encoding Standard 3.2.1. Open Geospatial Consortium [online]. [cit. 2015-11-14]. Dostupné z: http://portal.opengeospatial.org/files/?artifact_id=20509

[28] The GeoJSON Format Specification [online]. [cit. 2015-11-14]. Dostupné z: http://geojson.org/geojson-spec.html

[29] ORLIK, Antonin;Orlikova. Current trends in formats and coordinate transformations of geospatial data - based on MyGeoData Converter. Central European Journal of Geosciences [online]. 2014, 6(3), 354-362 [cit. 2016-05-03]. DOI: 10.2478/s13533-012-0178-8. ISSN 18961517.

[30] TopoJSON: A smaller GeoJSON with some neat tricks. 2013. Dostupné také z: http://www.somebits.com/~nelson/SotM-2013-TopoJSON.pdf

[31] TopoJSON: An extension to GeoJSON that encodes topology. GitHub [online]. [cit. 2016-05-03]. Dostupné z: https://github.com/mbostock/topojson/wiki

[32] Protocol Buffers - binární protokol nejen pro RPC. Dagblog [online]. 2008 [cit. 2016-05-03]. Dostupné z: http://www.dagblog.cz/2008_07_20_archive.html

[33] Mapbox Vector Tile Specification adopted by Esri. Mapbox [online]. 2015 [cit. 2016-05-03]. Dostupné z: https://www.mapbox.com/blog/vector-tile-adoption/

[34] Vector Tiles - What confused you about them? Reddit [online]. 2015 [cit. 2016-05-03]. Dostupné z: https://www.reddit.com/r/gis/comments/3wfvlm/vector_tiles_what_confused_you_about_them/

[35] OSM2VECTORTILES: Free Vector Tiles from OpenStreetMap [online]. [cit. 2016-05-01]. Dostupné z: http://osm2vectortiles.org/

[36] OpenGeo Suite User Manual: Using the vector tiles output [online]. [cit. 2016-05-01]. Dostupné z: http://suite.opengeo.org/docs/latest/dataadmin/vectortiles/index.html#dataadmin-vectortiles

[37] Geoserver – Rendering Binary Vector Tiles. Salzburg Research [online]. [cit. 2016-04-03]. Dostupné z: http://www.salzburgresearch.at/blog/geoserver-rendering-binary-vector-tiles/

[38] PGRestAPI (a.k.a. Chubbs Spatial Server). GitHub [online]. [cit. 2016-02-03]. Dostupné z: https://github.com/spatialdev/PGRestAPI

[39] Vector Tiles. Mapbox [online]. [cit. 2016-01-03]. Dostupné z: https://www.mapbox.com/vector-tiles/

[40] Vector Tile Service. Mapzen [online]. [cit. 2016-03-03]. Dostupné z: https://mapzen.com/projects/vector-tiles/

[41] TileStache-Vector. TileStache [online]. [cit. 2016-04-03]. Dostupné z: http://tilestache.org/doc/TileStache.Vector.html

[42] VectorTileLayer. ArcGIS API for JavaScript [online]. [cit. 2016-05-03]. Dostupné z: https://developers.arcgis.com/javascript/jsapi/vectortilelayer-amd.html

[43] OpenLayers3: A high-performance, feature-packed library for all your mapping needs. [online]. [cit. 2016-05-03]. Dostupné z: http://openlayers.org/

[44] Zoomable Tiles. D3 Plugins [online]. [cit. 2016-05-03]. Dostupné z: https://github.com/d3/d3-plugins/tree/master/geo/tile

[45] Tangram. Mapzen [online]. [cit. 2016-05-01]. Dostupné z: https://mapzen.com/projects/tangram/

[46] INTRODUCING MAPBOX GL. Mapbox [online]. [cit. 2016-04-03]. Dostupné z: https://www.mapbox.com/blog/mapbox-gl/

[47] Leaflet: Open-source JavaScript library for mobile-friendly interactive maps [online]. [cit. 2016-05-02]. Dostupné z: http://leafletjs.com/

[48] Under the hood of Google Maps 5.0 for Android. Google Official Blog [online]. [cit. 2016-05-03]. Dostupné z: https://googleblog.blogspot.cz/2010/12/under-hood-of-google-maps-50-for.html

[49] OpenScienceMap 0.5 [online]. [cit. 2016-02-09]. Dostupné z: http://www.opensciencemap.org/

[50] HTML5 Vector Map Engine. GIS Cloud Developers [online]. [cit. 2016-02-09]. Dostupné z: http://developers.giscloud.com/html5-vector-map-engine/

[51] Node.js [online]. [cit. 2016-02-09]. Dostupné z: https://nodejs.org/en/

[52] JavaScript Topology Suite - JSTS [online]. [cit. 2016-02-09]. Dostupné z: http://bjornharrtell.github.io/jsts/

[53] Struktura a popis výměnného formátu RÚIAN (VFR) v1.8.0. 2016. Dostupné také z: http://www.cuzk.cz/Uvod/Produkty-a-sluzby/RUIAN/2-Poskytovani-udaju-RUIAN-ISUI-VDP/Vymenny-format-RUIAN/Vymenny-format-RUIAN-(VFR)/Struktura-a-popis-VFR-1_8_0.aspx

[54] POSTGIS: Documentation manual [online]. [cit. 2016-02-09]. Dostupné z: http://postgis.refractions.net/documentation/manual-1.3SVN/ch04.html

[55] OPEN GEOSPATIAL CONSORTIUM. Simple Feature Access [online]. 28. 5. 2011 [cit. 2015-10-10]. Dostupné z: http://www.opengeospatial.org/standards/sfa

[56] OBE, Regina. a Leo. HSU. PostGIS in action. London: Pearson Education [distributor], c2011. ISBN 19-351-8226-9.

[57] Mapshaper: Tools for editing Shapefile, GeoJSON and TopoJSON files [online]. [cit. 2016-05-03]. Dostupné z: https://github.com/mbloch/mapshaper

[58] Why The Hell Would I Use Node.js? A Case-by-Case Tutorial. Toptal [online]. [cit. 2016-02-05]. Dostupné z: https://www.toptal.com/nodejs/why-the-hell-would-i-use-node-js

[59] BY ALEX R. YOUNG, MARC HARTER., by Alex R. Young, Marc Harter. Node.js in practice. 2014. ISBN 16-172-9093-9.

[60] OpenGIS Web Processing Service. In: Open Geospatial Consortium [online]. [cit. 2016-05-03]. Dostupné z: http://portal.opengeospatial.org/files/?artifact_id=24151

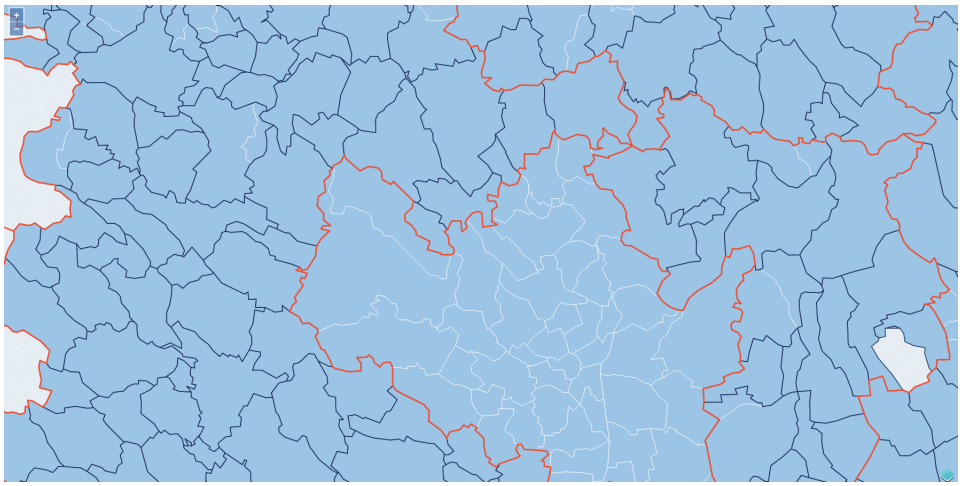
[61] Tiles à la Google Maps: Coordinates, Tile Bounds and Projection. MapTiler [online]. [cit. 2016-05-03]. Dostupné z: http://www.maptiler.org/google-maps-coordinates-tile-bounds-projection/

SEZNAM POUŽITÝCH ZKRATEK

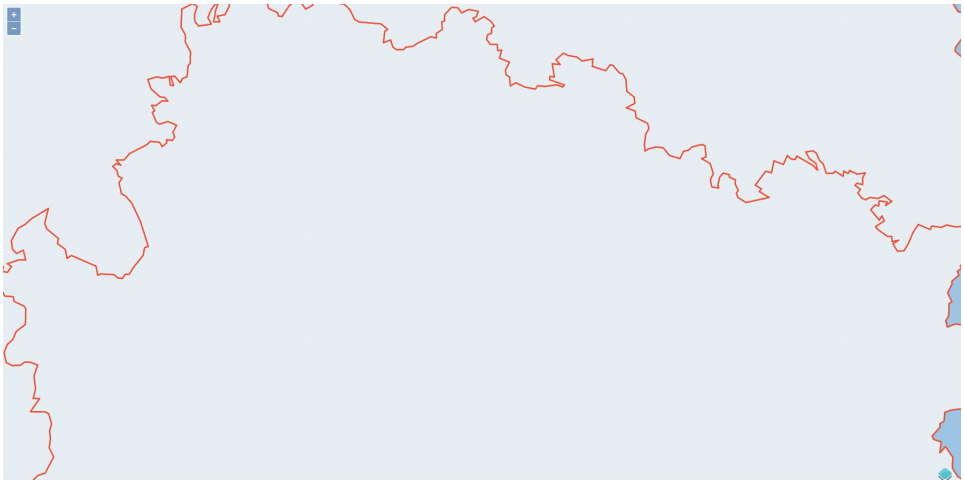
AJAX	Asynchronous JavaScript and XML
API	Application programming interface
ČUZK	Český úřad zeměměřický a katastrální
EPSG	European Petroleum Survey Group
GDAL	Geospatial Data Abstraction Library
GML	Geography Markup Language
GPS	Global Positioning System
ISO	International Organization for Standardization
JSON	JavaScript Object Notation
LoD	Level of detail
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
OGC	Open Geospatial Consortium
OSgeo	Open Source Geospatial Foundation
RUIAN	Registr územní identifikace, adres a nemovitostí
SQL	Structured Query Language
URL	Uniform Resource Locator
UTF-8	UCS Transformation Format
VDP	Veřejný dálkový přístup
VFR	Výměnný formát RÚIAN
VÚGTK	Výzkumný ústav geodetický, topografický a kartografický
WFS	Web Feature Service
WMTS	Web Map Tile Service
WMS	Web Map Service
XML	eXtensible Markup Language
XSD	XML Schema Definition
XSLT	Extensible Stylesheet Language Transformations

PŘÍLOHA 1

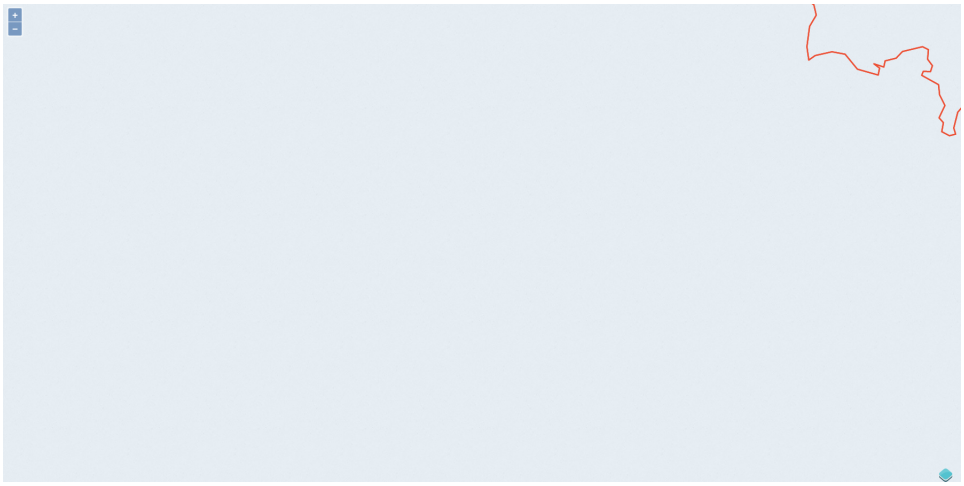
Náhledy mapy při automatizovaném testování výkonnosti



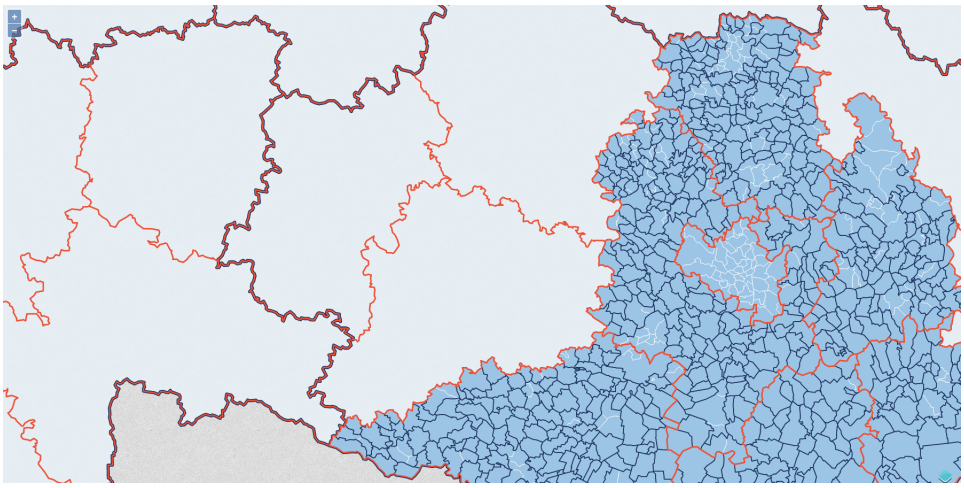
Obrázek 1: Náhled mapy při iniciální načtení.



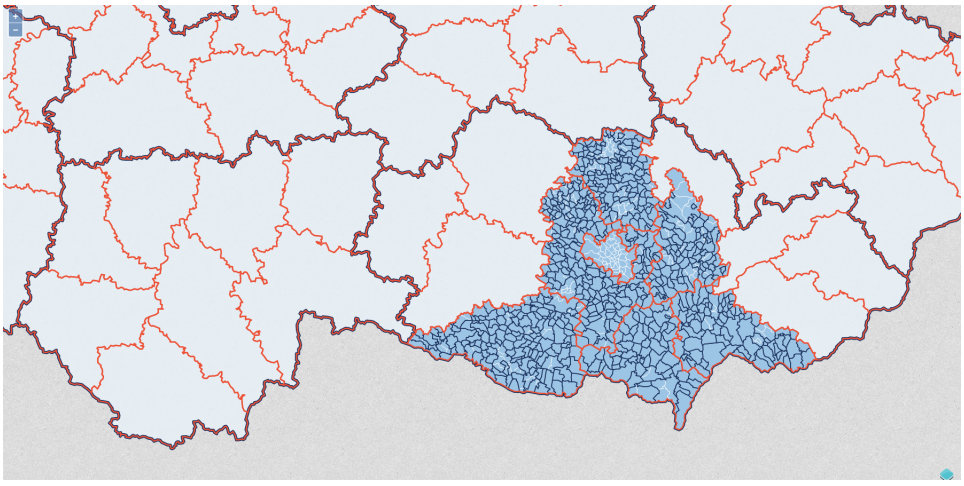
Obrázek 2: Náhled mapy při posunu doleva.



Obrázek 3: Náhled mapy při přiblížení.



Obrázek 4: Náhled mapy při oddálení.



Obrázek 5: Náhled mapy při oddálení č.2.

PŘÍLOHA 2

OBSAH PŘILOŽENÉHO CD

Přiložené CD obsahuje zdrojové kódy s nástroji pro import dat RUIAN do PostgreSQL databáze a dále obsahuje zdrojové kódy webové aplikace. Zároveň jsou zdrojové kódy dostupné v úložišti - https://github.com/LudvikAdamec/diplomova-prace.