ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

PROVOZNĚ EKONOMICKÁ FAKULTA

Datový model pro web GIS s integrací času

disertační práce

Autor: Ing. Jakub Konopásek

Školitel: doc. RNDr. Dana Klimešová, CSc., katedra Informačního inženýrství

Obsah

1 Úvod

V posledních několika letech je trendem vývoj aplikací pracujících s 3D prostorovými daty a s dynamickými prostorovými daty. U těchto dvou nadstaveb klasických dvou dimenzionálních geografických systémů se však naráží na problém relačních přístupů ukládání dat - jak uložit tří prostorová data do relační databáze / tabulky tak, aby s nimi šlo i nadále snadno a efektivně pracovat. Buď je třeba zásadně přepracovat dosavadní postupy, nebo vyvinout zcela nové přístupy (Fan et al. 2011). Je tedy třeba zkoumat a vyvíjet nové datové modely, algoritmy a nástroje, které budou umožňovat rychle a efektivně pracovat s rozmanitými daty jak v prostoru, tak v čase (Pultar et al. 2009, Fan et al. 2010). Na toto téma byla v posledních letech publikována řada vědeckých článků a toto téma je probíráno na řadě konferencí. Jako příklad lze uvézt sekce z jedné z největších GIS konferencí v tomto roce "17. International Multidisciplinary Scientific GeoConference SGEM 2017" sekce "Web-based geospatial data and GIS services" a "Multi-dimensional (2D, 3D, and 4D) spatial data modeling and data quality".

Kvůli potřebě mít data přístupná široké veřejnosti vzniká stále více webových aplikací, které umožňují časoprostorová data zobrazovat podle uživatelem zadaných kritérií či dokonce poskytují v různé míře podporu rozhodování uživatelem zadaných problémů (Komárková et al. 2010).

Komerční GIS řešení sice umožňují typické zobrazování a filtrování geografických dat na webu, jsou ale značně finančně náročné jak z hlediska pořízení, tak i provozu. Nehodí se na menší specializované aplikace a navíc často v malé míře podporují v čase se měnící data. U geografických dat s faktorem času je v tuto chvíli nutnost tvorby vlastní specializované webové aplikace prakticky vždy nutná, protože práce s temporálními daty je zatím velmi málo podporovaná běžně dostupnými nástroji (Fan et al. 2010, Khatri et al. 2006, Pultar et al. 2010).

Při tvorbě a provozu webových aplikací je dnes běžné používat tři základní roviny nástrojů. Z pohledu uživatele vidíme webové rozhraní, které je zpracováváno pro uživatele prohlížečem. Jedná se typicky o uživatelské rozhraní napsané v html/css a javascript, který poskytuje dynamické prvky, jako je právě vizualizace geografických dat. V dnešní době je již i volně dostupná řada aplikací/skriptů, které jsou schopny geografická data zobrazovat a uzpůsobovat jejich zobrazení potřebám uživatele.

Data jsou uživateli poskytována ze strany serveru, kde jsou zpracovávána specializovanou serverovou aplikací. Na serveru dnes můžeme nechat běžet řadu specializovaných aplikací a kompilovat skripty v řadě skriptovacích a programovacích jazyků. Ve většině případů se však používají skriptovací jazyky pro webové aplikace. Mezi nejběžnější skriptovací jazyky patří například skriptovací jazyk PHP či ASP. Na této úrovni pracujeme s daty a určujeme výstup – co se vlastně má uživateli zobrazit.

A třetí úrovní jsou systémy pro řízení báze dat. Ty nám umožňují data ukládat a pomocí dotazů z dat následně umožňují skriptovacímu jazyku získávat data, která jsou potřebná pro analýzu a zobrazení výstupu uživateli (Komárková et al. 2009).

Tato disertační práce se zabývá problémem jak uchovávat časoprostorová data pro potřeby webové aplikace. Součástí práce je analýza současného stavu, v níž jsou popsány různé možnosti uchování časoprostorových dat, architektura GIS a jsou zde analyzovány současné požadavky na časoprostorová data z hlediska webových GIS aplikací. Dále navazuje popis mnou navrženého řešení pro uchovávání časoprostorových dat. Navržený datový model používá jako základ metodu integrace času na úrovni řádku a tento přístup modifikuje, rozšiřuje a kombinuje i s přístupy z dalších datových modelů takovým způsobem, aby co nejvíce řešil specifika běžných webových GIS malé a střední velikosti. Tj. specifika webových GIS, které nepoužívají specializovaný GIS server. Navržený datový model lze použít jako základ pro návrh specifických databází, které pak umožní efektivně udržovat časoprostorová data a presentovat je uživateli prostřednictvím webových aplikací.

2 Řešené téma

2.1 Cíle disertační práce

Cílem této práce je navrhnout nový datový model, který by se hodil pro potřeby strukturování informací o geografickém objektu měnícím se v čase. Přičemž navržený model by mělo být možno použít jako základ pro specifické datové modely webových GIS aplikací pracujících s časoprostorovými daty.

Konkrétně se jedná o návrh datového modelu pro uchovávání časoprostorových dat a jejich metadat v relační databázi, tedy v systému řízení báze dat, který je běžně používaný ve webových službách. Navržený datový model, by měl být vhodný pro co nejvíce běžných webových GIS aplikací malého a středně velkého rozsahu. Proto by výsledný datový model měl vyhovovat potřebám současných webových GIS aplikací, jak z hlediska uživatelů a administrátorů těchto aplikací, tak z hlediska technických limitací webového prostředí a architektury webových GIS.

Model by měl umožňovat data připravit pro webové GIS aplikace. Měl by být optimalizován, s prioritou pro dotazy, které výzkum ukáže jako typické pro současné webové GIS aplikace.

Model by také měl umožňovat pokud možno co nejlepší manipulaci s daty - tj. jejich snadné ukládání, aktualizaci, údržbu, ale i kontrolu jejich validity a případnou možnost jejich stažení a oprav v případě zjištění chyby v nahraném setu dat. Toto všechno by však nemělo přijít výrazně na úkor předchozím požadavkům - model by měl být i přes jeho složitost použitelný pro velmi limitované prostředí webových služeb.

2.2 Metodika

Pro splnění cíle této práce – návrhu nového datového modelu se specializací pro webové GIS aplikace pracující s časoprostorovými daty jsem postupoval následovně:

- Nejprve jsem provedl analýzu současného stavu architektury geografických informačních systémů a analyzoval specifika architektury GIS ve webových službách

- Dále jsem provedl analýzu typů dat, které je možno ve webových geografických informačních systémech použít - jejich typy, strukturu a možnosti jejich využití v praxi, obzvláště pak ve webovém prostředí.

- Následně jsem se zaměřil na v čase se měnící data. Provedl jsem analýzu současného stavu - základních konceptů a typů času z hlediska databázových systémů. Analýzu v současnosti používaných datových modelů, jejich kladných stránek a omezení. Blíže jsem věnoval pozornost zakomponování času v relačním databázovém modelu a zakomponování fakturu času do geografických databází a výzkumu specializovaných datových modelů s tím spojených. Jedná se především o modely založené na bi-temporálních relačních databázích a modely založené na událostech.

- Po zjištění současného stavu v těchto třech oblastech jsem provedl výzkum a analýzu omezení a potřeb webových geografických informačních systémů a specializovaných webových GIS aplikací. Vycházel jsem při tom, jak z dlouholeté praxe, výzkumu existujících webových aplikací, jejich struktury, služeb, množství a struktury dat které poskytují. V průběhu výzkumu jsem vedl rozhovory s experty zabývajícími se časoprostorovými aplikacemi a administrátory několika webových geografických informačních systémů. Cílem tohoto výzkumu bylo zjistit například jaká zobrazení a dotazy na databázi typicky webové GIS aplikace používají a na co je třeba se zaměřit při optimalizaci navrhovaného datového modelu.

- Dále jsem přistoupil k vlastní tvorbě modelu, která vycházela z výše uvedeného výzkumu a analýz. Model jsem testoval v reálném prostředí, abych zjistil jeho limity a možnosti jeho využití. Na základě modelu v různých verzích jeho návrhu jsem provedl řadu case study a dokonce ho využil v praxi při návrhu reálných webových aplikací, na kterých jsem zpětně otestoval jeho využitelnost.

2.3 Motivace

Pracuji v oboru webových služeb již skoro dvacet let. Za tu dobu jsem se podílel na řadě projektů a vytvořil řadu unikátních jak GIS, tak normálních webových aplikací. V rámci svého studia jsem se podílel na výuce předmětů přímo spojených s návrhem informačních systémů, webovými službami a geografickými informačními systémy. Konkrétně jsem vedl semináře k předmětu Správa a zpracování geografických dat, Geografické informační systémy,

Databázové systémy, Informační inženýrství či předmět Projektování Informačních systémů. Tímto směrem se také ubírala moje bakalářská a diplomová práce.

Během doktorského studia jsem se prakticky výhradně zabýval výzkumem na téma webových aplikací obsahující geografická data. Měl jsem na toto téma grant pod universitní grantovou agenturou IGA. Výsledky výzkumu jsem publikoval na řadě vědeckých konferencí počínaje domácí doktorandskou konferencí Think Together pořádané Českou zemědělskou universitou. Na této konferenci jsme s kolegou získali ocenění za článek týkající se temporálních databází. Výzkum temporálních datových modelů a jejich porovnání a návaznost na webové služby dále vedl k účasti na konferenci NAUN v Paříži kde jsem měl dokonce dva příspěvky jeden jako hlavní autor a jeden jako spoluautor (Konopásek et al. 2013, Klimešová et al. 2013) a na jejich základě pak byly publikovány dva plnohodnotné výstupy (Konopásek et al. 2014, Klimešová et al. 2014). Na téma webových geografických informačních systému a časoprostorových databází jsem publikoval řadu článků do vědeckých časopisů (Konopásek a Klimešová 2016). Momentálně poslední článek na toto téma jsem (jako hlavní autor) publikoval na jaře v roce 2017 v mezinárodním časopise International Journal of Applied Engineering Research indexovaném v databázi Scopus (Konopásek a Klimešová 2017).

3 Analýza současného stavu

Jak již bylo zmíněno v úvodu, vývoj geografických informačních systémů se stále posouvá kupředu a v poslední době již nejde jen o klasická 2D data, ale o zaznamenávání a práci s v čase se měnícími objekty. Jedna z nejdůležitějších oblastí, a zároveň oblast, ve které dochází momentálně asi k největšímu použití GIS aplikací, je oblast webových aplikací. Webové technologie mají výhodu ve dvou důležitých věcech. Zaprvé umožňují přístup a šíření geografických dat na prakticky libovolnou platformu – tj. mají vysokou interoperabilitu. A zadruhé modularita webových služeb umožňuje jednoduše šířit a tvořit specializované GIS služby aplikace a vázat je na řadu dalších již existujících webových aplikací. Díky tomu je možno geografická data snadno šířit v žádoucí podobě velké skupině uživatelů, a to bez zvláštních požadavků na hardware a software těchto uživatelů (Komárková et al. 2010).

Integrace či vývoj webových aplikací pracujících s geografickými daty však samozřejmě přinášejí řadu úskalí. Řada vývojářů předních GIS aplikací již nabízí serverovou verzi svých GIS řešení, která umožňuje data vytvořená a zpracovaná v desktopových verzích těchto programů následně publikovat na webu. Bohužel tyto profesionální serverové systémy pro práci s geografickými daty jsou často velmi drahé. Často umožňují použít pouze předem připravené funkce pro práci s daty. Rozšířit je o funkce a zobrazení, které nejsou podporovány, je často velice náročné. Je zde také často problém s kompatibilitou dat mezi specializovanou GIS aplikací a dalšími aplikacemi, které potřebujeme použít (např. webovým systémem pro podporu rozhodování, či webovým systémem pro vkládání a úpravu vlastních dat) (Kopáčková et al. 2011, Bandyophadyay et al. 2012).

Pokud chceme uživateli zpřístupnit jím požadovaná prostorová data přes webový prohlížeč a tím minimalizovat hardwarové a softwarové nároky na uživatele, musíme většinou vždy zajistit alespoň následující. Musíme uložit veškerá data, která chceme uživateli poskytnout v podobě definované datovým modelem a mít k nim přístup skrze systém pro řízení báze dat, který s tímto datovým modelem umí pracovat. Musíme mít aplikaci, která na straně serveru dokáže data podle zadání zpracovat a připravit na výstup uživateli. A musíme připravit aplikaci, která bude zprostředkovávat komunikaci mezi uživatelem a se serverovou částí a bude zobrazovat uživateli požadované výstupy. Pokud chceme vytvořit systém na míru, aby vyhovoval specializované úloze integrace času, musíme použít speciální datový model a vytvořit aplikaci, která s ním bude schopná pracovat. V takovémto případě nestačí pouze mít datový model pro strukturování vlastních dat, ale je třeba se zabývat k nim náležícími

metadaty. Přímé a korektní zakomponování metadat do datového modelu nám umožní pracovat s různými sety dat jako s celkem, což je právě často případ časoprostorových GIS systémů. Ty totiž většinou pracují s několika sadami dat, získanými nad skupinou pozorovaných objektů, ale v různých časových úrovních (De Souza et al 2014, Ellul et al. 2014, ISO19115 2013).

3.1 Druhy dat v GIS a jejich struktura

Geografická data považujeme za data, která obsahují takzvané georeferencované informace. To jest informace nějakým způsobem umístěné v prostoru. Tato data mohou nabývat mnoha různých podob, ale mají společné to, že je aplikace GIS mohou zobrazit, či použít k analýze řady problémů. Běžně lze prostorová data kategorizovat na data vektorová a rastrová. Kromě prostorových dat pracujeme v GIS také s charakteristikami (atributovými daty) jednotlivých objektů. Propojení charakteristik objektů a umístění objektů je to, kvůli čemu GIS nabývá čím dál tím více na důležitosti. Navíc k prostorovým datům častokrát připojujeme metadata, což jsou data obsahující doplňující informace o vlastních geografických datech (kdo a kdy je vytvořil, co vlastně geografická data znamenají atd. (Charvát 2007).

Rastrová data

Rastrová data jsou typicky surová data, získaná z průzkumu v terénu. V rastrových datech jsou veškerá data reprezentována souvislým sledem pixelů. Respektive tedy řádky a sloupci po sobě jdoucích buněk, kde každá buňka má svoji hodnotu, která je pro uživatele vyobrazena specifickou barvou. Tím pádem jeden pixel nám vždy svojí hodnotou dává informaci o jedné charakteristice místa, které tento pixel představuje. Pro zobrazení více charakteristik daného místa pomocí rastru musíme použít více rastrových vrstev (typický příklad jsou multispektrální data). Nejběžnějším příkladem rastrových dat, se kterým se uživatel GIS na internetu běžně setkává, jsou ortofotomapy a další snímky pořízené ze satelitů či letadel (Charvát 2007).

Vektorová data

Vektorová data uchovávají geografické objekty jako geometrické tvary. Vektorová data běžně vznikají zpracováním dat rastrových.

Objekty ve vektorových datech mohou nabývat různých tvarů. Rozlišujeme vektorová data typu bod, linie a polygon.

Bod je vyjádřen dvojicí souřadnic x a y. V mapách typicky zeměpisnou délkou a šířkou ve stupních, či v metrech. V praxi je bod často používán pro objekty, které by byly příliš malé, kdyby měly být zobrazeny v měřítku mapy. Samozřejmě pro finální zobrazení pro uživatele je bod na mapě možno nahradit grafickou značkou.

Linie je nejběžněji používána pro zobrazení cest či řek. Jedná se o spojení nejméně dvou bodů. U objektů typu linie můžeme měřit vzdálenost.

Polygonem pak znázorňujeme objekty, které zabírají nějaký prostor. U polygonu můžeme vypočítat jeho obsah a obvod.

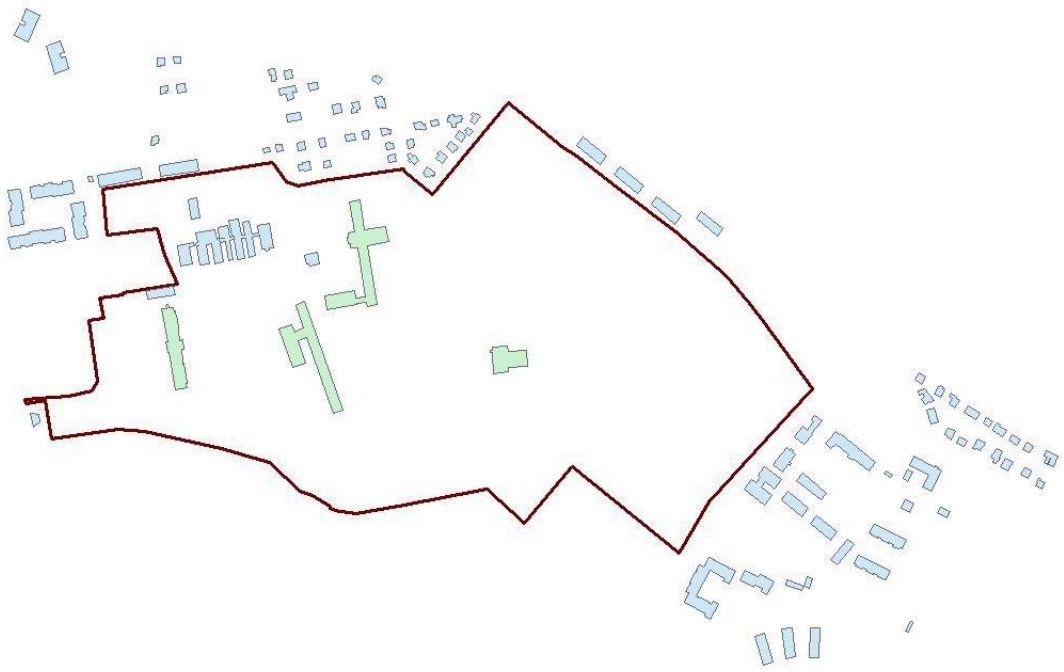
Objekty můžeme většinou zobrazit jako více než jedním z těchto tvarů. Záleží na tom, jaký tvar se nám hodí pro naše potřeby. Například město můžeme znázornit na mapě jako bod, nebo pokud potřebujeme zaznamenat jeho přesný tvar a hranice, tak ho zaznamenáme jako polygon. Příklad vyobrazení vektorové vrstvy je na obrázku číslo 1.

Vektorová data na rozdíl od rastrových pracují s objekty. To co by v rastrových datech mohlo vypadat, jako jeden objekt mohou být v datech vektorových dva objekty stejného typu se společnou hranicí. Vektorová data mohou obsahovat (mít k sobě připojena) kromě polohy objektu i další atributy objektu a další informace o objektu. Díky tomu, že máme data přesně rozdělená na dílčí objekty, můžeme s nimi snadněji provádět pokročilé analýzy, objekty filtrovat, zpřesňovat, zjišťovat vztahy mezi nimi atd.

Na rozdíl od rastrových dat, která přebírají formáty používané pro počítačovou grafiku je podoba vektorových dat z velké míry určována tím, v jakém programu byla data vytvořena. V zásadě každý významný GIS program má svůj formát pro ukládání vektorových dat a tento formát navíc častokrát prošel při svém vývoji řadou změn. Je tak často problém načíst starší vektorová data, i jen do nové verze programu, který je vytvořil.

Při konverzi vektorových dat z formátu do formátu je třeba také pamatovat na souřadnicový systém, ve kterém jsou data zobrazena. Některé formáty implicitně předpokládají použití daného souřadnicového systému, jiné umožňují si souřadnicový systém zvolit.

Vektorová data můžeme v praxi potkat v řadě podob. Zaprvé jako běžné soubory na disku, se kterými můžeme jednoduše pracovat, kopírovat je atd. S tuto formou uchovávání dat se běžně setkáme při práci se specializovanými GIS programy. Kromě toho můžeme vektorová data najít v databázi, do které je zprostředkován přístup přes GIS program, často ve formě webové služby. Mezi nejpoužívanější patří webové služby WMS a WFS. Tyto služby umožňují v různě omezené míře přistupovat na dálku k vektorovým datům uloženým na GIS serveru jak přes webové rozhraní, tak přes napojení skrze specializované GIS programy (Charvát 2007).



Obrázek č. 1 - Vektorová data - budovy fakult v areálu ČZU a okolní obytné domy

Atributová data

Hlavní věc, která odlišuje geografický informační systém od mapy je možnost provádět filtraci a analýzu geografických dat. K té je třeba, aby geografický informační systém měl kromě jejich polohy také přístup k řadě dalších charakteristik zkoumaných objektů. U vektorových dat můžeme u jednotlivých objektů uchovávat v geografickém informačním systému libovolný počet námi vyžadovaných charakteristik - atributů.

Většina geografických informačních systémů umožňuje pracovat s vektorovými vrstvami jako s klasickými tabulkami kde co řádek to objekt v dané vrstvě. Sloupce pak jsou atributy daných objektů. GIS systémy pak většinou generují jeden sloupec jako unikátní identifikátor a další sloupec (případně sloupce) do kterého nějakým způsobem vyznačují geografické souřadnice. Uživatel si následně může definovat libovolné množství dalších sloupců s charakteristikami objektů dané vrstvy. Typicky se jedná o stejné nebo podobné typy jako u klasických databází:

- znaková data (text, char)
- numerická data (int, bigint, double, float)
- datum a čas (date, time)

Zavedený způsob, že GIS pracuje s vrstvami jako s 2D tabulkami je právě důvod proč se dnes velmi těžko pracuje s časoprostorovými daty. Většina GIS zvláště těch webových neumí pracovat s časoprostorovými daty jinak než co vrstva to jeden snapshot v čase. Pokročilejší GIS pak zavádějí čas do atributové tabulky jako atribut (Charvát 2007).

3.1.1 Metadata

Jak již bylo zmíněno výše, metadata jsou množina informací popisující nějaká daná data (v našem případě většinou jak atributová tak geografická). Jedná se typicky o popis obsahu, geografický rozsah, časový rozsah, prostorové reference, jakost, reprezentaci (ISO19115 2013, Nařízení komise (ES) č. 1205/2008 2008, Charvát 2007). Jinými slovy popisují kdo, co, kdy, kde, proč a jak o daných geografických datech.

Metadata jsou v praxi často velmi opomíjena a podceňována a přitom jsou klíčová pro orientaci v různých geografických datech, jejich použití, zpracování a konverzi při práci s více datovými sadami. Zvláště u geografických dat je použití metadat obzvláště důležité, protože geografická data mohou mít mnoho podob a mohou být zpracována celou řadou způsobů a bez znalosti k nim příslušících metadat je často geografická data velmi těžké použít.

Metadata jsou také využívána ke katalogizaci vytvořených map. Existují internetové slovníky metadat, kde lze vyhledávat mapy podle metadat. Jedná se například o http://gisinventory.net/ či http://gcmd.nasa.gov/.

Metadaty v geografických systémech a tím jak je strukturovat a používat se zabývá celá řada směrnic a standardů. Jedná se především o Dublin Core (ISO 15836), ISO19115/ISO19119 a směrnici INSPIRE (ISO19115 2013, Nařízení komise (ES) č. 1205/2008 2008, Dublin core metadata nedatováno)

Dublin Core

Schéma Dublin Core určuje univerzální standard pro popis prakticky libovolných digitálních a i dalších objektů včetně webových stránek, obrázků, CD, knih atd. Nejdůležitější část standardu je Dublin Core Metadata Element Set, Version 1.1, která nám určuje množinu 15 položek metadat pro popis objektu (v případě GIS se nejběžněji jedná o sadu geografických dat) k němuž se metadata vztahují. Jedná se o následující:

- title - název,
- creator - autor,
- subject - klíčová slova či klíčový popis objektu
- description - podrobnější popis
- publisher - vydavatel
- contributor - přispěvatel
- date - datum
- type - typ objektu
- format - formát, velikost, doba trvání atd.
- identifier - unikátní identifikátor objektu
- source - zdroj odkud objekt pochází
- language - jazyk
- relation - vztah k jiným objektům
- coverage - pokrytí, hranice kde je objekt relevantní

- rights - práva týkající se objektu

Tyto položky nemají specifikované pořadí a mohou se pro jeden objekt opakovat. Můžeme je pak například zobrazit jako tagy HTML pro použité na webových stránkách.

Například:

<meta name="DC.Format" content="text">

<meta name="DC.Title" content="Muj text" >

<meta name="DC.Language" content="cs" >

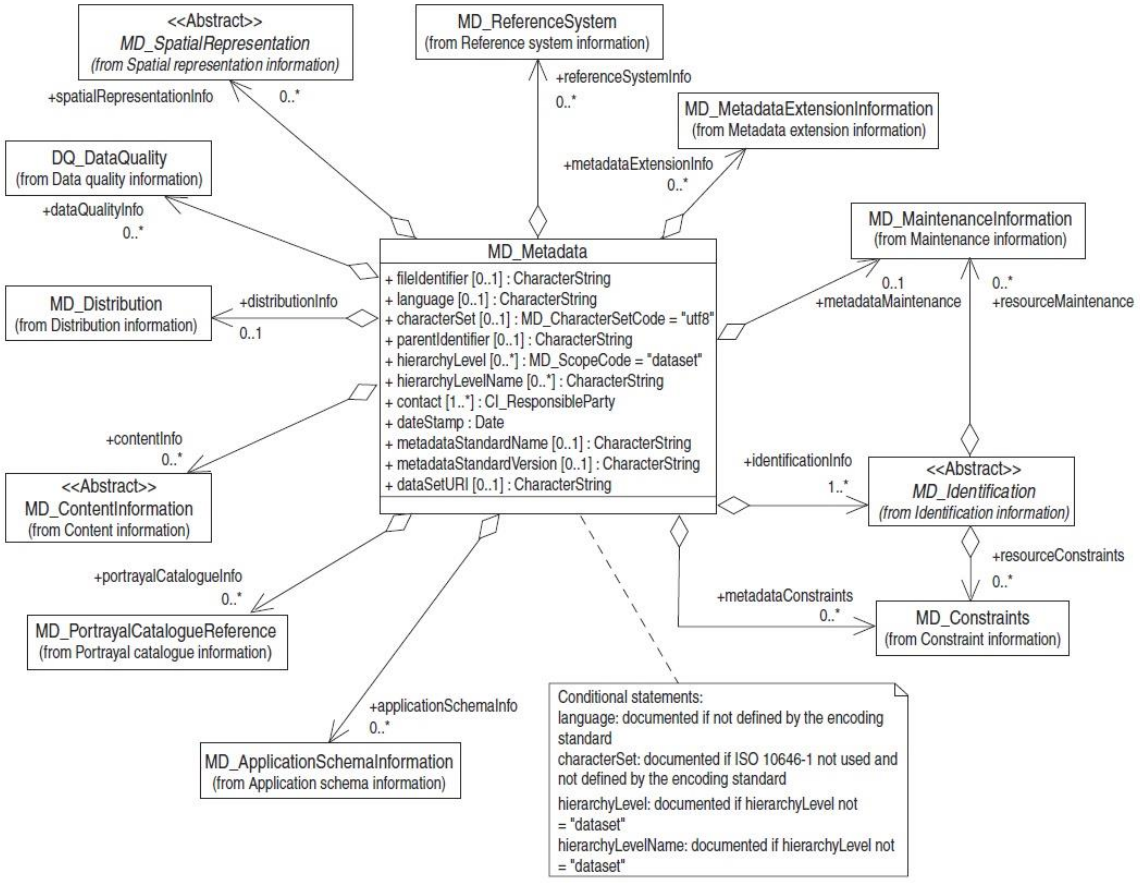
<meta name="DC.Publisher" content="jmeno" >

Dublin Core má kromě základního setu položek několik dalších úrovní, které umožňují možnost rozšíření a další specifikace (Dublin core metadata nedatováno).

ISO 19115

Novější norma ISO 19115, která je momentálně adaptována ČSN i v České republice a navazující ISO 19119 se momentálně používají v rozsáhlých geografických webových službách jako WMS. Strukturování metadat popsané v těchto normách ISO je mnohem podrobnější a složitější než Dublin Core, nicméně podle tvůrců novější normy ISO je ISO 19115 s Dublin Core kompatibilní. Zvláště pak ze směru ISO do Dublin Core. Převod opačným směrem může být obtížnější kvůli komplexnější struktuře standardu ISO 19115.

Schéma pro metadata popsané v ISO 19115 se dá tematicky rozčlenit do 12 balíčků (obrázek č. 2).



Obrázek č. 2 - UML model základních balíčků metadat normy ISO 19115 (převzané z ISO19115 (2013))

Podobně jako u Dublin Core je u ISO 19115 definován set základních položek, na rozdíl od Dublin Core jsou to však položky specificky vybrané pro popis geografických dat. V ISO 19115 je jich 22 a obsahují navíc mimo jiné položky jako znakovou sadu, prostorové rozlišení či referenční systém (Geospatial Metadata nedatováno).

Standardy ISO 19115 a 19119 rozšiřuje ISO 19139 o zásady implementace schématu metadat v jazyku XML. Metadata tak mohou být snadno zpracovávána a v jazyce XML exportována k dalšímu použití v jiných systémech.

INSPIRE

Třetí hlavní standard určující podobu metadat je stanoven v evropské směrnici INSPIRE. Základní rozdělení položek metadat v této směrnici není tak podrobně do hloubky stanovené jako u standardu ISO 19115. Ale určuje základní položky vytvořené přímo pro metadata prostorových dat a je podrobnější než Dublin Core. Je pravděpodobné, že bude pravděpodobně v budoucnu hodně používána v Evropě a ČR pro strukturování metadat.

Základní set položek metadat pro prostorová data podle směrnice INSPIRE je v tabulce č. 1 (Nařízení komise (ES) č. 1205/2008 2008).

Metadata souborů prostorových dat a sérií souborů prostorových dat

Reference	Prvky metadat	Násobnost	Podmínka
1.1	Název zdroje (Resource title)	1	
1.2	Abstrakt zdroje (Resource abstract)	1	
1.3	Typ zdroje (Resource type)	1	
1.4	Lokátor zdroje (Resource locator)	0..*	Povinný, pokud je k dispozici adresa URL pro získání dalších informací o zdroji a/nebo pro přístup k souvisejícím službám.
1.5	Jednotný identifikátor zdroje (Unique resource identifier)	1..*	
1.7	Jazyk zdroje (Resource language)	0..*	Povinný, pokud zdroj zahrnuje textové informace.
2.1	Tematická kategorie (Topic category)	1..*	
3	Klíčové slovo (Key word)	1..*	
4.1	Geografické ohraničení (Geographic bounding box)	1..*	
5	Časová reference (temporal reference)	1..*	
6.1	Původ (Lineage)	1	
6.2	Prostorové rozlišení (Spatial resolution)	0..*	Povinné u datových souborů a sérií datových souborů, jestliže lze stanovit odpovídající měřítko nebo hodnotu rozlišení.
7	Soulad (Conformity)	1..*	
8.1	Podmínky přístupu a použití (Conditions applying to access and use)	1..*	
8.2	Omezení veřejného přístupu (Limitations on public access)	1..*	
9	Odpovědná organizace (Responsible party)	1..*	
10.1	Kontaktní místo metadat (Metadata point of contact)	1..*	
10.2	Datum metadat (Metadata date)	1	
10.3	Jazyk metadat (Metadata language)	1	

Tabulka č. 1 - Základní rozdělení metadat podle směrnice INSPIRE (převzato z Nařízení komise (ES) č. 1205/2008 (2008) ze dne 3. prosince 2008, kterým se provádí směrnice Evropského parlamentu a Rady 2007/2/ES týkající se metadat)

Metadata a datové modely časoprostorových dat pro web

Existuje celá řada vědeckých studií, které se zabývají návrhem různých variací datových modelů, pro strukturování časoprostorových dat, aby umožňovaly snadné uschování a práci s více či méně specifickými daty a snadné řešení vybrané problematiky. Většina těchto vědeckých studií se opírá o case study, kde tyto datové modely byly použity pro tvorbu web GIS. V dalších částech této práce budou jejich hlavní kategorie popsány. Naprostá většina těchto datových modelů se však zabývá pouze návrhem datového modelu pro uchování časoprostorových dat, ale jak zaznamenat v těchto často značně složitých datových modelech metadata je pominuto. Přitom tím, že se jedná o specializovaný datový návrh, který je třeba většinou implementovat mimo specializované GIS servery a tím pádem zde chybí nástroje a prostředky těchto serverů, které častokrát umožňují metadata zavádět, uchovávat a pracovat s nimi. Navíc pokud se jedná o specializované časoprostorové modely tak při jejich implementaci vyvstává celá řada možných problémů, jak s metadaty naložit.

Jak již bylo popsáno metadata nám popisují nějaký set geografických dat. Pokud geografický informační systém pracuje pouze s jedním setem dat, tak v zásadě není potřebné při návrhu datového modelu a následně databáze zohledňovat metadata. Nicméně ve chvíli kdy pracujeme s časoprostorovými daty, tak většinou se jedná data získaná z různých zdrojů, různými lidmi a různým způsobem. Jedná se tedy vlastně o rozdílné sety geografických dat. Tyto sety dat a jejich charakteristiky může být nutné identifikovat z řady důvodů. Například se může stát, že daný set dat je chybný a musí se smazat, či je třeba zkontrolovat data zadaná určitou osobou. Většina datových modelů pro časoprostorová data toto řeší tím, že metadata, která by teoreticky mohly být třeba, zavádí jako atributy jednotlivých objektů. To však může způsobovat v některých případech redundanci dat. Na druhou stranu některé údaje, které jsou uvedené mezi základními ve schématech metadat musí v případě časoprostorových dat být uvedeny přímo u jednotlivých objektů a je tedy redundantní je znovu uvádět v případné tabulce s metadaty. Může se jednat například o časový údaj, kdy časovou referenci dat pro která metadata jsou, získáme dotazem na nejmenší a největší čas vlastních dat a je vlastně redundantní tento údaj mít znovu zapsaný napevno v metadatech a může to velmi snadno v tomto případě vést k chybám v datech (De Souza et al. 2014, Ellul et al 2014).

Mít možnost v jednom systému pracovat s více sety geografických dat tedy může být z časoprostorového hlediska klíčová záležitost a budu se propojením datového modelu pro vlastní časoprostorová data a jejich metadata ve své disertační práci zabývat.

3.1.2 GIS servery a webové služby

V dnešní době je celá řada geografických dat dostupná přes internet pro uživatele ať už skrze specializované desktopové aplikace či webové rozhraní. Většina těchto služeb je poskytována skrze GIS servery. Na GIS serveru běží v pozadí specializovaný databázový systém, který se za uživatele stará prakticky o všechno. Častokrát je to objektová databáze, avšak pro uživatele se data v naprosté většině případů stále prezentují jako relační databáze, resp. co jedna vrstva to jedna tabulka v databázi. Pro speciální analýzy a práci s časoprostorovými daty je tento přístup často nevyhovující. Proto se takovéto specializované aplikace staví na uživatelem vytvořených datových modelech a po zpracování požadovaných filtrů a vybrání dat k zobrazení uživateli se data konvertují do klasické podoby co vrstva to tabulka a to se pak teprve předává k zobrazovacímu modulu GISU.

Mezi nejrozšířenější standardy, které GIS servery poskytují, jsou WMS (Web Map Service) a WFS (Web Feature Service). WMS vrací uživateli jím vybrané a vyfiltrované údaje jako rastr a WFS vrací vektorová data. Oba standardy byly vyvinuty pod záštitou Open Geospatial Consortium. WFS vrací vektorová data pomocí jazyka GML (Geography Markup Language), který byl vyvinut pro export geografických dat také pod OGC (Michaelis et al. 2012, Charvát 2007).

3.1.3 Soubory s vektorovými daty

Při práci s vektorovými daty přicházíme do styku v zásadě se třemi druhy informací. Základní je informace o umístění a tvaru objektů v prostoru. Tu pak běžně rozšiřují další informace o objektech, respektive tedy informace o charakteristikách/atributech objektů. A nakonec jsou informace o tom, jakým způsobem se vektorová data zobrazují na mapě. Například pozice: 49°56'22.9"N 14°11'15.3"E; atribut jméno: Karlštejn; zobrazení na mapě: značka hradu a zobrazení jména.

Některé GIS programy všechna tato data shromažďují v jednom souboru a některé tato data rozdělují i do dvou či více souborů. ArcGIS společnosti ESRI, který patří mezi nejpoužívanější GIS programy používá formát shapefile (.SHP) pro vlastní prostorová data objektů, formát dbf pro atributy objektů, formát lyr pro informace o zobrazení objektů na mapě a řadu dalších souborů pro další informace. Jako další formáty použitelné pouze ve speciálních GIS programech stojí za zmínku určitě AutoCAD formát DQG a DXF.

Kromě těchto specializovaných formátů jsou zde formáty volně použitelné. Dnes nejpoužívanější jsou pravděpodobně formáty KML a GeoJSON.

3.2 Časoprostorová data

Konceptuální časoprostorový datový model

Data, používaná geografickým informačním systémem, se zpravidla dají rozdělit na

- prostorové informace (lokace a tvar geografických objektů),
- časové informace (změny v čase),
- atributy (vlastnosti, kvalita a charakteristiky geografických objektů) a
- topologické vztahy mezi jednotlivými objekty.

Tyto datové prvky se dají popsat pomocí teorie množin následovně. Množina atributů "A" - obsahuje vlastnosti geografických objektů. Množina "ST" je množina dvou nebo tří-dimenzionálních prostorových prvků určující místo a tvar geografických objektů s a vztahy mezi nimi a to vždy pro daný čas "t". Každý prvek množiny ST obsahuje k němu příslušnou skupinu atributů z množiny atributů A z času t (Shi et al. 2009).

$$ST = \{ si(t): f(si(t)) \in A, 0<i<\infty, -\infty<t<\infty \} \quad (3.1)$$

Integrace času jako atributu založená na relačním datovém modelu

Integrace času jako atributu je nejběžnější způsob jak zavézt čas do relační databáze a je to tím pádem způsob často využit pro tvorbu specializovaného geografického informačního systému s podporou času pro webové rozhraní.

Existují tři základní úrovně, kde se může faktor času objevit (Combi et al. 2010):

- na úrovni relace
- na úrovni řádku
- na úrovni atributu

Mít faktor času integrován na úrovni relace v zásadě znamená, že pro každý časový interval existuje samostatná tabulka, resp. vrstva. To vede ke značné redundanci dat a získat historii jednoho objektu znamená dotazovat se na každý časový interval zvláštním dotazem. Nicméně tento způsob uchovávání dat umožňuje velmi jednoduše získat data v konkrétním časovém okamžiku.

Faktor času na úrovni řádku znamená, že každý řádek v tabulce je označen časem platnosti daného řádku (nebo časem transakce či obojím). Ve chvíli kdy dojde ke změně objektu, který je řádkem reprezentován v databázi, tak je vytvořen nový řádek se stejným identifikátorem objektu, aktualizovanými hodnotami atributů objektu a novým časem platnosti (příklady v kapitole 4.3 Čas v relační databázi). Při tomto přístupu dochází k menší redundanci dat než když je faktor času integrován na úrovni relace, ale stále může docházet k poměrně značné redundanci, zvláště pokud má objekt hodně charakteristik a v čase se mění pouze některé, nebo se některé mění často a některé málo (viz. tabulka č. 2).

Id záznamu	Id pole	Čas platnosti	Vlastník	Vlastní pole od	Chráněná oblast	Naměřené hodnoty pH/KCl
1	1	leden 2015	Ivo	2005-01-01	Ano	4,6
2	1	únor 2015	Ivo	2005-01-01	Ano	5
3	1	březen 2015	Ivo	2005-01-01	Ano	4,7

Tabulka č. 2 - problém redundance s integrací faktoru času na úrovni řádku

V tabulce jsou uchovávány informace o zemědělských polích - jelikož uvažujeme, že se tabulka aktualizuje každý měsíc, je možné mít čas platnosti pouze jako jedno pole s identifikací měsíce a roku, po který je záznam platný. V čase se mohou měnit všechny

atributy pole kromě jeho ID, ale naměřené hodnoty pH se zadávají a mění každý měsíc, zatímco ostatní atributy se mění maximálně párkrát za existenci pole v systému. V tomto případě nám tím vzniká velká nežádoucí redundance, protože každý měsíc budou do tabulky znovu a znovu vnášeny řádky s opakujícími se hodnotami a změna bude vždy pouze v atributech Id záznamu, čas platnosti a naměřené hodnoty pH.

Poslední úroveň integrace času je na úrovni atributů. To je nejnižší možná úroveň a v zásadě to znamená opatřit každý atribut v řádku svým vlastním označením jeho času platnosti. K tomu se nejběžněji používají takzvané nested relation schemas (Combi et al. 2010). Jedná se o nástroj, který umožňuje do jednoho atributu ukládat množiny dat a pracovat s nimi. Běžné webové databáze tento nástroj nepodporují a tak tato úroveň integrace není tak rozšířena. V tabulce č. 3 můžeme vidět podobu databáze s faktorem času integrovaným na úrovni atributů. V buňkách tabulky můžeme mít více hodnot a u každé je uveden interval času platnosti a interval času transakce.

Patient	Ward
Rossi ⟨[1, +∞], [5, +∞] ⟩	Cardiology ⟨[1, 1] ∪ [4, +∞], [8, +∞] ⟩ Intensive Care Unit ⟨[2, 3], [5, +∞]⟩
Smith ⟨[5, +∞], [8, +∞]⟩	Internal Medicine ⟨[5, 9], [8, +∞]⟩ Neurology ⟨[10, +∞], [8, +∞]⟩
Hubbard ⟨[4, 12], [4, +∞]⟩	Intensive Care Unit ⟨[4, 9], [4, +∞]⟩ Cardiology ⟨[10, +∞], [10, 11]⟩ Pneumology ⟨[10, +∞], [12, 13]⟩ Pneumology ⟨[10, 12], [14, +∞]⟩

Tabulka č. 3 - Faktor času integrovaný na úrovni atributů (převzato z Temporal information systems in medicine (Combi et al. 2010))

V praxi se většinou vytvoří datový model s časem integrovaným na úrovni řádků, přičemž se pak posoudí jak často, případně jestli vůbec se budou jaké atributy v čase měnit a podle toho se v datovém modelu rozdělí atributy pod více tříd, aby byla datová redundance co nejmenší, ale aby zároveň netrpěl datový model přílišnou rozdrobeností a aby na něj nebylo obtížné tvořit dotazy. V případě tohoto modelu se s polohou nakládá jako s dalším běžným atributem.

3.2.1 GEO-ATOM a EDGIS

Jedna ze základních a již velmi dobře zdokumentovaných teoretických reprezentací časoprostorových dat je geo-atom. Geo-atom popisuje základní časoprostorová data dynamického geografického informačního systému jako asociaci mezi bodem v časoprostoru a vlastností. Jedná se tedy o n-tici prvků obsahující vektor x s časoprostorovými koordináty <x, y, z, t> (pro 2D GIS <x,y,t>), identifikátorem charakteristiky objektu (atributu) Z a specifické hodnoty tohoto atributu v čase a prostoru definovaném vektorem x - z(x), jak je zapsaná výrazem 3.2.

$$<x,Z,z(x)> \tag{3.2}$$

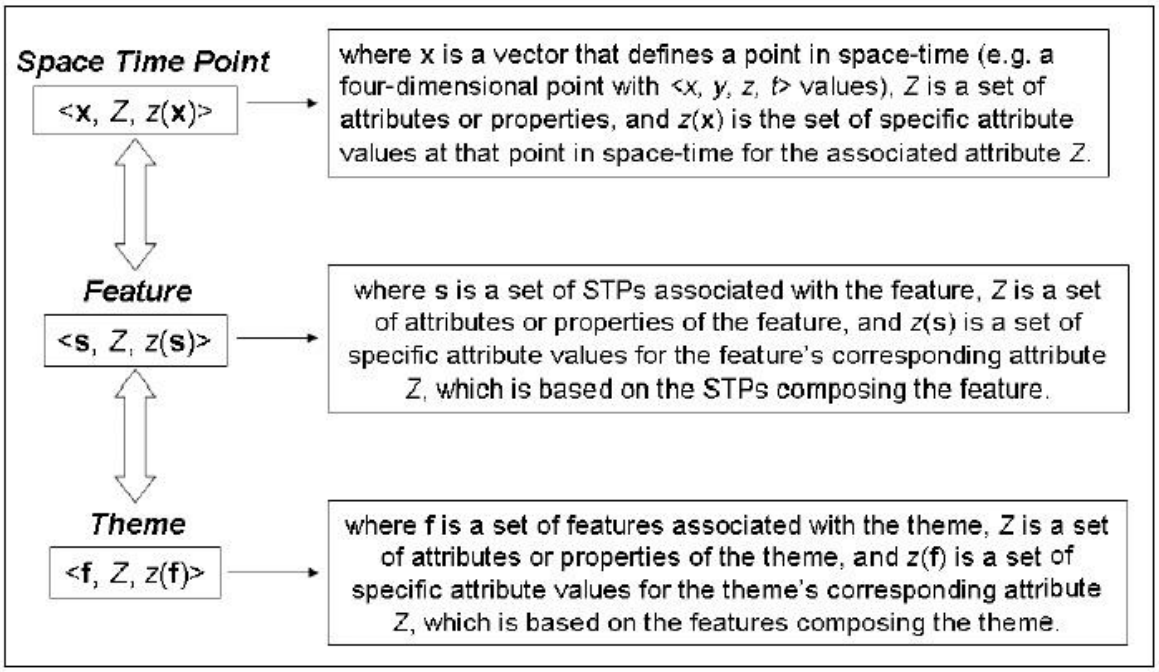
Takže příklad geo-atomu může být: na souřadnicích 49.67471, 15.82478 se hodnota atributu porost rovná listnatý les. Každý objekt v geografickém informačním systému může obsahovat libovolné množství geo-atomů, přičemž každý z nich obsahuje informace o stavu objektu v daném okamžiku (Goodchild et al. 2007).

Tato teoretická reprezentace byla použita pro tvorbu jednoho z moderních datových modelů pro uchovávání časoprostorových dat - EDGIS (Pultar et al. 2009, Pultar et al. 2010).

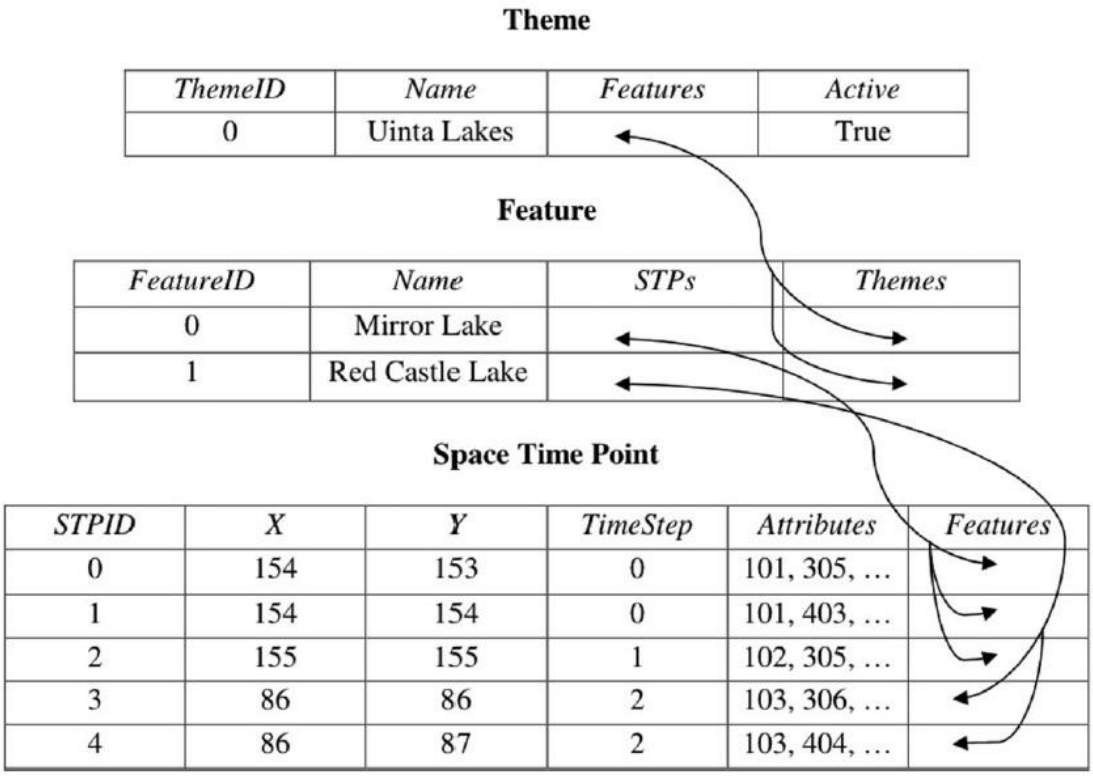
Tento přístup k integraci času a datový model z něj odvozený je momentálně ve fázi raného výzkumu. Pultar rozšiřuje geo-atom na space-time point, který definuje následovně.

$$<x,Z,z(x)> \tag{3.3}$$

Kde na rozdíl od geo-atomu Z je množina atributů a z(x) je množina hodnot těchto atributů. Objekty jsou v EDGIS definovány space-time body na nich závislými a mohou být sdružovány do motivů, které reprezentují geografické jevy. Pultar a kol. ve svém článku dále rozvádějí možnosti využití datového modelu EDGIS pro práci s dynamickými daty (Pultar et al. 2010). Na obr č. 3 je vidět závislost space-time bodů na objektech a motivech a na obrázku č. 4 je pak znázorněna datová struktura EDGIS.



Obrázek č. 3- vztah mezi základními prvky EDGIS (převzato z Wildfire Evacuation and Volunteered Geographic Information (Pultar et al. 2009))



Obrázek č. 4 - znázornění datové struktury EDGIS (převzato z EDGIS: a dynamic GIS based on spacetime points (Pultar et al. 2010))

3.2.2 Časoprostorový model zaměřený na události a čas

Časoprostorový model založený na událostech - Event-oriented Spatio-temporal Data Model a z něj odvozený model time-based integration method, který je momentálně ve fázi výzkumu, poskytuje na integraci času pohled odlišující se od zaběhlých modelů, kde se s časem pracuje jako s atributem pozorovaných objektů. V této metodě se považuje čas za plnohodnotný rozměr objektů v geografickém informačním systému. V návrhu podle tohoto datového modelu se používá oddělená tabulka událostí, kde se uchovávají vlastní změny objektů. Ta je napojená na tabulku vztahů, kde se pro snazší práci s daty uchovávají topologické vztahy mezi objekty (tj. informace o tom když se objekt se skládá z více částí). Tato tabulka je napojena na tabulku vlastních objektů, ve které jsou uchovány základní prostorové údaje objektu a jeho atributy (nebo napojení na tabulku atributů). Pro integrace prvku času se v praxi podle modelu jako je tento, kde je časový faktor brán jako další dimenze popisovaného objektu, používá zvláštní tabulka. Její řádky obsahují časy, kdy docházelo ke změnám. V běžném modelu, který s prvkem času nakládá jako s atributem, by byla časová složka uchovávána v tabulce spolu s dalšími atributy objektu. Díky uchovávání faktoru času

odděleně od vlastních atributů objektů, lze na rozdíl od běžného přístupu, poměrně jednoduchým dotazem snadno vyzískat časovou posloupnost změn libovolné skupiny objektů v libovolném časovém úseku. Y.T. Fan ve svých publikacích dokládá značný nárůst efektivity při dotazech při použití takovéhoto modelu oproti ostatním klasickým modelům (Fan et al. 2010, Fan et al. 2011).

3.3 Čas v relační databázi

V sadách dat můžeme narazit na použití časové hodnoty v řadě situací. Běžně rozlišujeme tři různé dimenze časových hodnot. Toto rozdělení platí obecně pro použití časové složky, jak pro prostorové databáze, tak pro databáze neprostorové (Tang et al. 2010, Ott a Swiaczny 2001, Combi et al. 2010).

Čas platnosti (Valid time)

Čas platnosti nebo doba platnosti je časový údaj, který stanovuje platnost uložených dat v našem modelovaném prostoru. Tj. říká, kdy se informace, které se čas platnosti týká, stala platná, případně kdy se stala zase neplatná. Respektive se objekt s danými charakteristikami objevil v našem modelovaném světě. Podle okolností a použitého datového modelu se může čas platnosti vztahovat na celý objekt, či pouze na jeho prostorové vlastnosti, či na jeho atributy nebo jejich část. V databázi kde pracujeme s faktorem času, obvykle můžeme najít čas platnosti ve formě intervalu (či případně i více intervalů), ale může to být také pouze jeden časový údaj. Nejčastěji v praxi je to datum, kdy daná informace vstoupila v platnost, a datum, kdy již platnosti pozbyla, tj. interval s dolní a horní časovou hranicí.

Object_id	Valid_from	Valid_to	Barva
001	2011-05-15 13:00:00		bílá
002	2011-05-15 13:00:00	2011-07-28 07:15:00	bílá

002	2011-07-28 07:15:00	2011-08-30 09:30:00	žlutá
002	2011-08-30 09:30:00		bílá
003	2011-03-01 11:15:00	2011-07-28 07:15:00	modrá
003	2011-09-11 07:15:00	2011-10-02 07:15:00	Bílá
...	...	...	

Tabulka č.4 - příklad tabulky se zavedeným časem platnosti

Z tabulky č. 4 vidíme, že objekt 001 existuje v databázi od 5. května 2011 a jeho platnost stále trvá (není vyplněna horní hranice času platnosti) a po celou dobu jeho trvání má objekt bílou barvu. Objekt s identifikátorem 002 existuje v databázi od stejné doby a jeho platnost také stále trvá, ale jeho barva se během jeho existence v systému dvakrát změnila. Objekt má tedy v tabulce tři záznamy pokaždé se stejným ID, ale s jinou, na sebe navazující dobou platnosti a jinou hodnotou atributu barva. Díky tomu že na sebe intervaly času platnosti navazují, máme nad objektem po celou dobu jeho existence (tedy od května do současné doby) v modelovaném světě přehled a víme přesně, kdy ke změně barvy došlo. Objekt číslo tři byl zaveden do modelovaného světa v březnu 2011 a v červenci byla jeho platnost ukončena a pro modelovaný svět od té doby tedy zanikl. V září se však znovu stal předmětem našeho pozorování a má tedy opět záznam v databázi. Z tabulky je vidět, že se změnila oproti předchozímu záznamu barva objektu 003 z modré na bílou, ale protože nemáme žádnou zprávu o jeho stavu mezi červencem a zářím, tak nevíme, kdy se tato změna barvy stala a nevíme ani, zda mezitím neměl ještě nějakou jinou barvu. V říjnu platnost objektu 003 opět vypršela a v momentální době se v modelovaném světě opět nevyskytuje a nevíme tedy, co se s tímto objektem děje.

Interval však může mít podobu například měsíce či roku, během něhož je informace platná, či nemusí být ohraničen z jedné strany vůbec. Nebo může být stanovena doba, kdy informace vstoupila v platnost a doba po kterou tato platnost trvá – jak je vidět v tabulce č. 5.

Object_id	Time_point	Time_unit (dny)
001	2011-05-15 13:00:00	2
002	2011-07-22 07:15:00	6
002	2011-09-12 17:45:00	6
003	2011-05-03 12:00:00	12
004	2011-07-02 12:00:00	5
...	...	...

Tabulka č. 5 - Příklad tabulky se zavedeným časem platnosti v podobě jednostranného intervalu s dobou platnosti

Tento způsob se většinou používá, pokud nás bude ve výsledném geografickém informačním systému zajímat především doba platnosti a u zkoumaných objektů víme dobu trvání platnosti relevantních dat předem. Záleží na tom, jak je databáze navržena. Jak bylo již zmíněno, v praxi převládá první typ s přesně danými hranicemi, protože nám umožňuje snazší dotazování na podobu modelovaného světa v různých časech.

Čas transakce (Transaction time)

Druhá varianta času, na kterou můžeme často v databázích, kde pracujeme s faktorem času narazit je čas transakce. Čas transakce nám neříká, kdy se udála změna modelovaného světa, respektive zaznamenaných objektů, ale kdy jsme my tu změnu zaznamenali do našeho systému, resp. databáze. Tj. kdy byl proveden příkaz INSERT či UPDATE, který přidal či přepsal data dokumentující změnu objektu v naší tabulce. Čas transakce většinou doplňuje čas platnosti. Závisí na potřebách a vlastnostech systému a uživatele jak a kdy čas transakce

zaznamenáváme. Ve většině datových modelů budeme většinou chtít alespoň vědět, kdy byl daný objekt přidán do databáze.

Object_id	Valid_from	Valid_to	Added	Removed	Barva
001	2011-05-15		2011-05-15		bílá
002	2011-07-22	2011-07-28	2011-07-22		bílá
002	2011-07-28	2011-08-28	2011-07-28		žlutá
002	2011-08-28		2011-07-28		bílá
...	...	...	...	...	

Tabulka č. 6 - Příklad tabulky se zavedeným časem platnosti a časem transakce

V tabulce č. 6 můžeme vidět, že objekt č. 001 má čas platnosti od května 2011 do současnosti a byl do databáze zaznamenán ve stejný den, jako jsme ho začali modelovat. Objekt č. 002 je pro nás relevantní od 22. července a opět jsme ho do databáze zanesli ve stejný den. 28. července jsme však zjistili, že objekt bude třeba na měsíc přemalovat na žlutou barvu. Jelikož víme, že objekt bude žlutý pouze měsíc a pak se vrátí jeho barva zpět na bílou - tj. známe čas platnosti předem, mohli jsme zadat horní hranici času platnosti a nový záznam se změnou barvy zpět na bílou předem také 28. července. Sloupec removed je prázdný protože všechna data jsou stále platná.

Uživatelem definovaný čas (User time)

Poslední typ času je uživatelem definovaný časový údaj. To je v zásadě jakýkoli čas, který není časem transakce ani časem platnosti. Jedná se tedy o jakoukoli informaci, která je relevantní danému objektu a vyjádřená časovou hodnotou. S tou výjimkou, že nesmí ovlivňovat validitu objektu nebo jeho atributů v čase a nesmí udávat čas úpravy informace pro daný databázový systém.

3.3.1 Základní druhy databází pro uchovávání časových dat

Podle toho zda databáze obsahuje čas platnosti a čas transakce můžeme rozdělit časoprostorové databáze na čtyři základní druhy (Tang et al. 2010, Ye et al. 2010).

- Snapshot database - nepodporuje práci ani s časem platnosti ani s časem transakce. Jedná se o konvenční databáze kde, pokud máme v čase se měnící data, tak pokud se určitý fakt modelovaného světa změní, tak se jeho předchozí verze přepíše. Máme přístup pouze vždy k datům momentálně platným. Tím pádem z takovéto databáze nelze zjistit historie změn. Databáze může obsahovat čas transakce či čas platnosti a tím pádem může součástí jeho atributů být informace o případných minulých či budoucích událostech, ale jedná se pouze o záznam momentální charakteristiky daného objektu. Databáze neumožňuje zaznamenávat historii objektů ani jejich změn.

ID pole	Majitel	Majitel_od	Pole
1	Ivo	1999-01-01	souřadnice (shape)
2	Tomáš	2001-01-01	souřadnice (shape)
3	Michal	2004-01-01	souřadnice (shape)
4	Jakub	2006-01-01	souřadnice (shape)
...	...	...	...

Tabulka č. 7 - příklad snapshot databáze s uživatelem definovaným časem

Tabulka č. 7 obsahuje zemědělská pole se jmény jejich vlastníků a data kdy byly majiteli zakoupeny. Sloupec Majitel_od není v tomto případě čas platnosti, ale pouze uživatelem definovaný čas. Z toho plyne, že pokud se změní majitel pole, změní se

záznam v databázi a do sloupce Majitel a Majitel_od se zadají nové hodnoty a staré hodnoty se tím ztratí a nemůžeme se k ní již nijak vrátit (viz tabulka č. 8).

ID pole	Majitel	Majitel_od	Pozemek
1	Ivo	1999-01-01	souřadnice (shape)
2	Michal	2002-01-01	souřadnice (shape)
...	...	...	...

Tabulka č. 8 - příklad snapshot databáze s uživatelem definovaným časem 2

- Historical database - podporuje práci s časem platnosti. Historické databáze na rozdíl od snapshotových již podporují tři dimenze - objekty, jejich atributy, a jejich vývoj v čase. Příklad je znázorněn v tabulce č. 9

ID	ID_pole	Majitel	Naměřené hodnoty pH/KCl	Platné_od	Platné_do
1	1	Ivo	5	1999-01-01	
2	2	Tomáš	5,2	2001-01-01	
3	3	Michal	4,9	2004-01-01	
4	4	Jakub	4,5	2006-01-01	

...		...	...	...	

Tabulka č. 9 - příklad databáze se zavedeným časem platnosti

Opět je použita tabulka zemědělských polí. Byly přidány dva sloupce - platné_do což je sloupec, do kterého se bude zaznamenávat horní hranice času platnosti. Dolní hranice času platnosti bude zaznamenána do sloupce platné_od. Rozdíl oproti databázi kde platné _od není čas platnosti, ale jen uživatelem definovaný čas je ten, že můžeme zaznamenat několik různých stavů a rozlišit je od sebe právě časem platnosti. V těchto ukázkách se čas integruje na úrovni řádků - způsobů integrace času je celá řada a budou popsány níže, ale zde ukázaný způsob záznamu času platnosti na úrovni řádků je nejběžnější.

U pole s ID 2 se stejně jako v předchozím případě změnil majitel, ale jak vidíme z tabulky č. 10, tak nyní máme v tabulce zaznamenány historický přehled, kdy má pole jakého majitele. V relačních databázích, které se používají pro webové služby a i běžně pro GIS musí být každý řádek unikátně identifikovatelný od jiných - tj. musí mít primární klíč. Pro tento příklad jsou zde zavedeny sloupce ID a ID_ pole, kde ID je identifikátor záznamu a ID_ pole je identifikátor daného pole který může mít v tabulce libovolný počet záznamů. Primární klíč v tabulce s časem platnosti lze vytvořit také jako složený klíč identifikátoru objektu a právě času platnosti (Tedy v tomto případě ID_ pole a Platné_od/ Platné_do).

ID	ID_pole	Majitel	Naměřené hodnoty pH/KCl	Platné_od	Platné_do
1	1	Ivo	5	1999-01-01	
2	2	Tomáš	5,2	2001-01-01	2005-01-01
20	2	Michal	5,2	2005-01-01	

3	3	Michal	4,9	2004-01-01	
4	4	Jakub	4,5	2006-01-01	
...		...	...	...	

Tabulka č. 10 - příklad databáze se zavedeným časem platnosti 2 - po změně majitele

V případě, že se zjistí, že byla změna majitele pole provedena chybně a bude třeba jí opravit, tak nám nezbývá než tuto chybu opravit a změnit hodnotu v daném řádku. Protože však v tabulce nezaznamenáváme čas transakce, tak tento zásah nemusí jít snadno zpětně dohledat (viz tabulka č. 11).

ID	ID_pole	Majitel	Naměřené hodnoty pH/KCl	Platné_od	Platné_do
1	1	Ivo	5	1999-01-01	
2	2	Tomáš	5,2	2001-01-01	2005-01-01
20	2	Ivo	5,2	2005-01-01	
3	3	Michal	4,9	2004-01-01	
...		...	...	...	

Tabulka č. 11 - příklad databáze se zavedeným časem platnosti - po opravě

- Roll-back database - podporuje práci s časem transakce. Stejně jako v případě historické databáze má roll-back databáze tři dimenze - objekty, jejich atributy, ale poslední dimenze neobsahuje údaje o změnách, kterými objekt prochází v čase, ale o změnách, kterými v čase prochází záznamy v databázi. To znamená, že uživatel může na databázi dotázat, jak vypadal její obsah v jakémkoli momentě. Pro práci s transakčním časem platí, že změny se vždy provádí na aktuálním stavu objektu v databázi - tj. nemělo by jít změnit něco zpětně. Oprava chyby z předchozího příkladu by tedy vypadala, viz tabulka č 12.

ID	ID_pole	Majitel	Naměřené hodnoty pH/KCl	Vloženo	Změněno
1	1	Ivo	5	1999-01-03	
2	2	Tomáš	5,2	2001-01-03	2005-01-03
20	2	Michal	5,2	2005-01-03	2005-01-05
21	2	Ivo	5,2	2005-01-05	
3	3	Michal	4,9	2004-01-03	
1	1	Ivo	5	1999-01-03	
...		...	...	...	

Tabulka č. 12 - příklad databáze se zavedeným časem transakce - po změně majitele a následné opravě

Z tabulky je vidět, že po opravě přibyl další řádek, aby byla oprava do databáze zaznamenána. Oproti tabulce předchozího příkladu můžeme vidět, že se změnila i všechna data a sloupce s časy se přejmenovala na vloženo / změněno. Což vyplývá z

toho, že do sloupců nyní zaznamenáváme čas transakce a ne čas platnosti. Změna majitele pole se pro účely této aplikace hlásí na začátku roku - tím pádem čas platnosti vždy byl 1. ledna. Ale čas transakce, kdy byla změna do databáze zaznamenána, je jiný.

Poznat zda se ve sloupci nachází čas transakce či čas platnosti podle pojmenování sloupce ve kterém se nacházejí, není často možné a někdy jsou navíc tyto časy stejné. Záleží na tom, jaké informace ve výsledku od navrhované databáze potřebujeme a jak se stanoví pravidla na zacházení s daty.

- Bitemporal database - podporuje práci jak s časem platnosti, tak s časem transakce. Výsledná relace obsahuje čtyři dimenze - objekty, jejich atributy a zaznamenává dva druhy času - čas platnosti a čas transakce. To nám umožňuje se na databázi dotazovat nejen, jak vypadala historie v daný okamžik, ale i jak vypadala historie z pohledu určité doby. Pokud máme databázi s časem platnosti, můžeme se ptát například, kdo vlastnil pole s ID 2 v lednu 2003? A odpověď bude, že Tomáš. Pokud je naše databáze navíc bitemporalní, tak se navíc můžeme ptát, kdy to bylo zaznamenáno - a odpověď bude - 3. 1. 2003. Příklad bitemporální databáze je v tabulce číslo 13.

ID	ID_pole	Majitel	Naměřené hodnoty pH/KCl	Platné_od	Platné_do	Vloženo	Změněno
1	1	Ivo	5	1999-01-01		1999-02-05	
20	1	Ivo	5	1999-01-01	2015-01-01	2014-11-03	
21	1	Ivo	4,2	2015-01-01		2014-11-03	2015-02-22
23	1	Ivo	4,5	2015-01-01		2015-02-22	

...		...	...	...			

Tabulka č. 13 - příklad databáze se zavedeným časem platnosti i časem transakce

Řádek s ID číslo 21 má ve sloupci Platné_do datum, kdy pro systém přestal platit a byl nahrazen hodnotami v řádku s id číslo 23. Z tabulky je zřejmé, že bitemporální tabulky mohou obsahovat značné množství redundantních dat, ale umožňuje nám to zjistit zpětně, co jsme předně věděli v určitý časový okamžik.

V praxi se nejběžněji setkáváme s tabulkami historickými s doplněným časem transakce v podobě intervalu. Zaznamenává se tedy pouze, kdy byla provedena poslední změna času platnosti. Resp. buď se čas transakce vždy přepisuje, aby byl k dispozici čas poslední úpravy, nebo se zaznamenává set dat všech transakcí, ale nezaznamenává, co přesně se upravilo. Není možné tedy zjistit, jak vypadala databáze v daný okamžik, ale víme, kdy byla změna provedena. Jako příklad je uvedena tabulka č. 14 - před opravou změny pH a tabulka č. 15 po ní.

ID	ID_pole	Majitel	Pole	pH/KCl	Platné_od	Platné_do	Od_vloženo	Do_vloženo
1	1	Ivo	souřadnice (shape)	5	1999-01-01	2015-01-01	1999-02-05	2014-11-03
20	1	Ivo	souřadnice (shape)	5,2	2015-01-01		2014-11-03	
2	2	Tomáš	souřadnice (shape)	5,3	2001-01-01		2001-02-03	

3	3	Michal	souřadnice (shape)	5,1	2004-01-01		2004-04-05	
...		...	...	...	...			

Tabulka č. 14 - příklad databáze se zavedeným časem platnosti jako plnohodnotné dimenze a časem transakce jako atributu 1 - před opravou

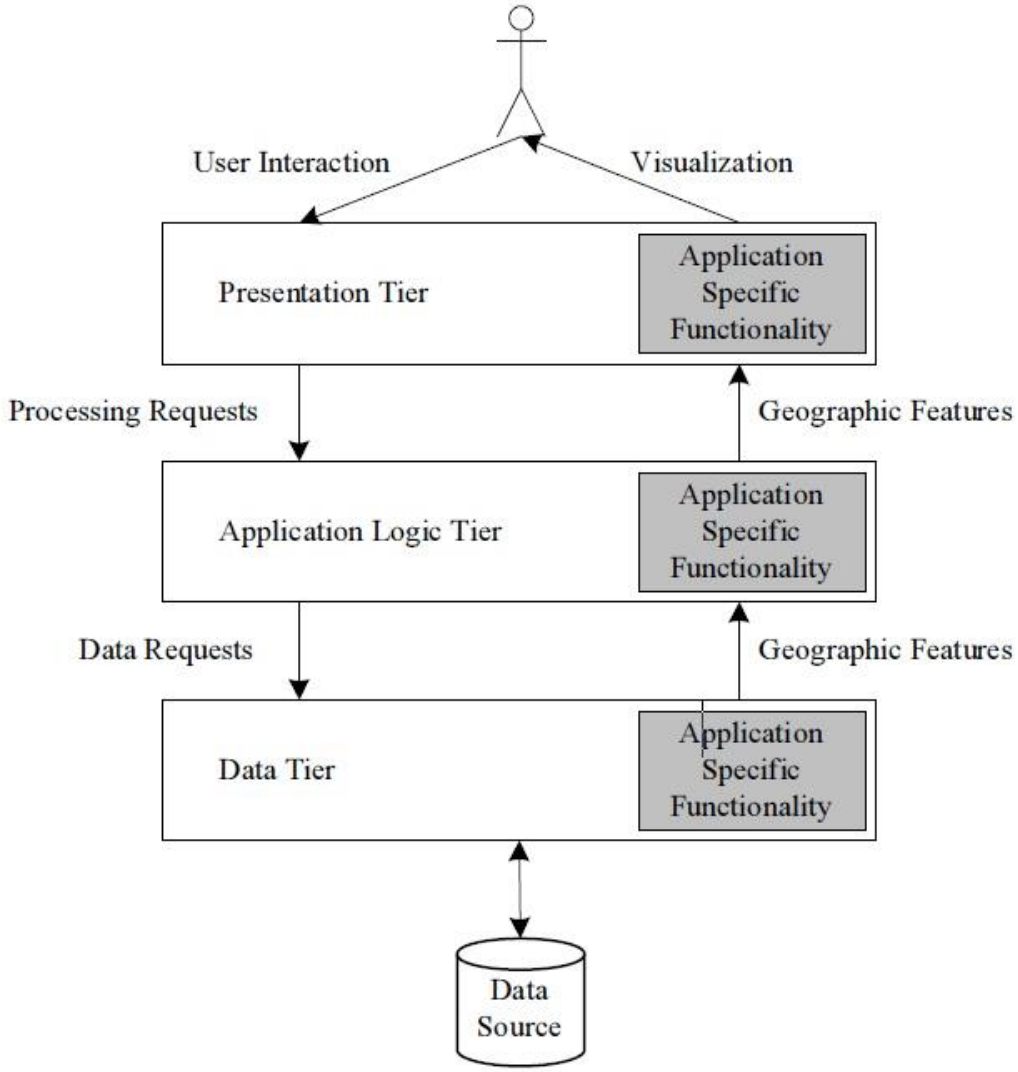
ID	ID_ pole	Majitel	Pole	pH/KCl	Platné_od	Platné_do	Od_vloženo	Do_vloženo
1	1	Ivo	souřadnice (shape)	5	1999-01-01	2015-01-01	1999-02-05	2014-11-03
20	1	Ivo	souřadnice (shape)	5,1	2015-01-01		2015-02-22	
2	2	Tomáš	souřadnice (shape)	5,3	2001-01-01		2001-02-03	
3	3	Michal	souřadnice (shape)	5,1	2004-01-01		2004-04-05	

Tabulka č. 15 - příklad databáze se zavedeným časem platnosti jako plnohodnotné dimenze a časem transakce jako atributu 1 - po opravě

3.4 Architektura geografických informačních systémů

Geografický informační systém se skládá z celé řady na sebe navazujících funkcionalit a na jeho architekturu se dá nahlížet z celé řady pohledů. Níže popsané rozdělení vychází ze

standardně používaného rozdělení specifikované ISO a Open Geospatial Consortium.



Obrázek č. 5 - Obecný systém architektury GIS (převzato z disertační práce *A Generic Architecture for Geographic Information Systems* ((Luaces 2004))

Z obrázku č. 5 vidíme, že architektura GIS se dá rozdělit do tří úrovní ((Ott a Swiaczny 2001, Luaces 2004, Tang et al. 2010).

- Data tier - úroveň správy dat - na této úrovni se provádí manipulace s daty pomocí dotazovacího jazyka. Jedná se tedy o systém řízení báze dat, který vrací sety geografických objektů k analýze a zobrazení.

- Application logic tier - úroveň zpracování dat - zde je implementována veškerá analýza dat a řešení problémů. Tato úroveň zajišťuje příjem požadavků od uživatele skrze presentační část GIS, zpracovává je a na jejich základě vytváří potřebné dotazy, které předává ke zpracování úrovni správy dat. Ta jí následně vrací potřebná geografická data a ty jsou v této vrstvě přetvářeny do podoby, která se může zobrazit uživateli.

- Presentation tier - úroveň zobrazení - zde je implementován user interface geografického systému. Na této úrovni tedy probíhá veškerá komunikace s uživatelem. Vstupy od uživatele jsou interpretovány a předány ke zpracování. A naopak po zpracování se výsledky interpretují uživateli v požadované formě.

Funkčnosti, které tyto úrovně geografickách informačních systémů poskytují, se následně dají dále podrobněji rozdělit - viz. obrázek č. 6.

- Úroveň správy dat:

    - data services - kolekce profilů specifické danému využití, zde mohou být implementovány specifické funkce pro zjednodušení manipulace s daty; zde mohou být implementovány služby jako WFS a WCS, které umožňují získávat data v různých podobách a formátech;

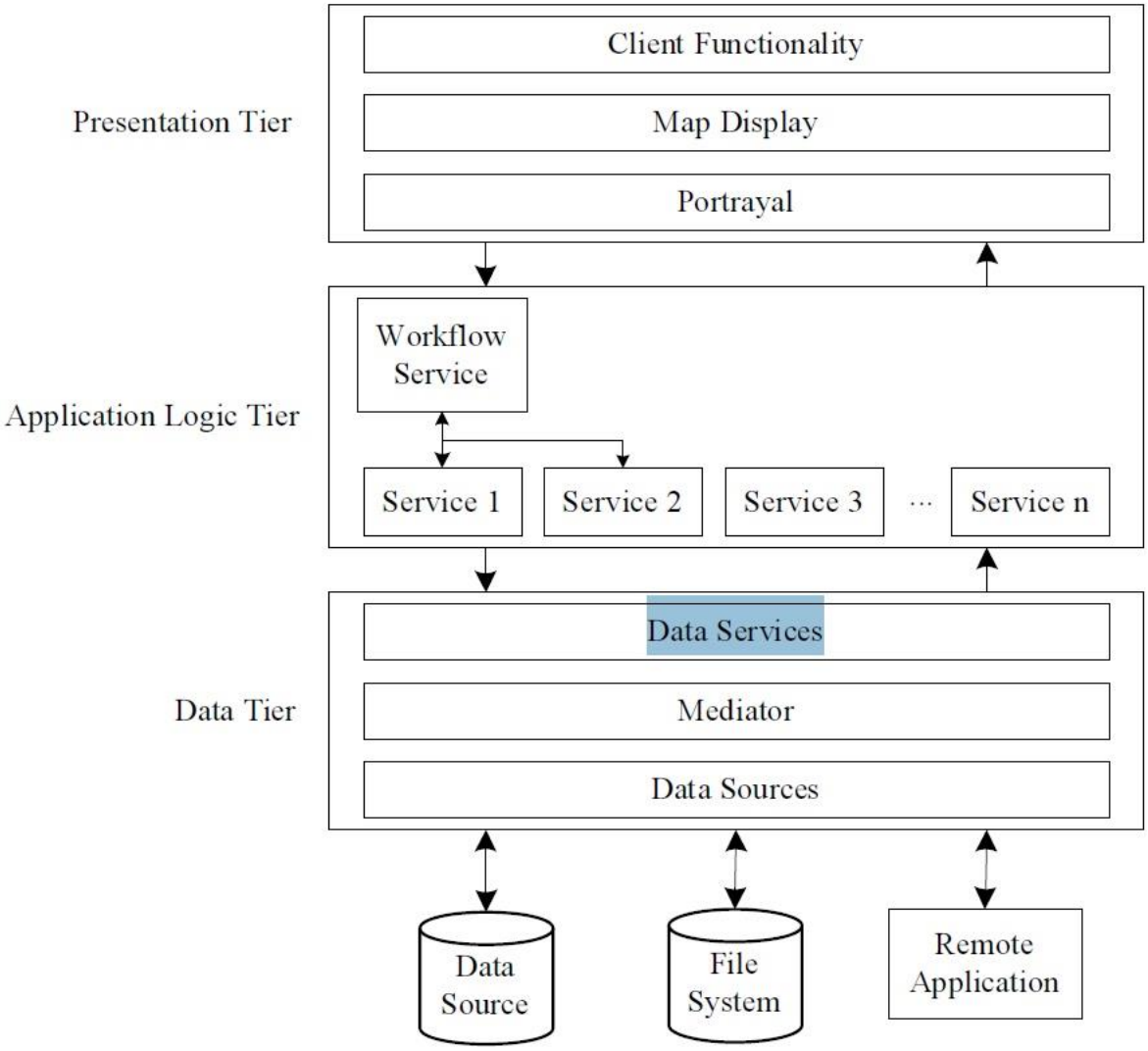
    - mediator - úroveň která zajišťuje přemostění mezi jednotlivými specifickými zdroji dat pomocí zavedení jednotného konceptuálního modelu pro geografické informace získané ze zdrojů dat a také zavedení dotazovacího jazyka;

    - data sources - nejnižší úroveň, která obsahuje moduly pro práci s různými specifickými zdroji dat.

- Úroveň zpracování dat - tato úroveň, jak již bylo řečeno, obsahuje jednotlivé analytické, transformační a další nástroje poskytované geografickým informačním systémem. Tato úroveň se nedělí do vrstev, ale do jednotlivých modulů, přičemž každý obsahuje specifickou funkčnost GIS. Jako příklad lze uvést například modul pro klasifikaci dat, navigaci, transformace dat mezi jednotlivými souřadnicovými systémy atd. Jednotlivé moduly na sebe samozřejmě mohou funkčně navazovat a vždy by měli

být napojené na workflow service modul, který by k nim měl umožňovat a řídit přístup podle potřeby uživatele či vývojáře.

- Úroveň zobrazení:
    - Portrayal layer - tato úroveň získává již zpracovaná geografická data a přetváří je na kartografické objekty, které se následně dají zobrazit uživateli. V této vrstvě se zpracovávají veškeré definice stylů jednotlivých objektů a stanovuje se zde, jak bude jaký objekt vypadat.
    - Map display layer - tato úroveň vykresluje dané kartografické objekty uživateli, umožňuje mu práci s kartografickými objekty a provádět na nich operace. Respektive poslat žádost o provedené operaci na úroveň zpracování dat.
    - Client functionality layer - tato úroveň zajišťuje funkcionalitu uživatelského rozhrani a zavádí dodatečné funkce, které jsou dostatečně jednoduché, že nepotřebují komunikaci s úrovní zpracování dat. Typicky se jedná o jednoduché operace měření či zoomování mapy ve chvíli, kdy již úroveň zobrazení má potřebná data v cashe.



Obrázek č. 6 - Softwarové vrstvy v architektuře GIS (převzato z disertační práce *A Generic Architecture for Geographic Information Systems* (Luaces 2004))

4 Datový model pro web GIS s integrací času

Výsledkem mého výzkumu je datový model pro časoprostorová data, který se hodí (tj. je na to navržen a optimalizován) pro uchovávání časoprostorových dat pro malé a středně velké webové GIS aplikace. Při vývoji tohoto datového modelu jsem čerpal z analýzy, jak architektury a současných webových geografických informačních systémů, tak analýzy současných datových modelů pro uchovávání časoprostorových dat. Dále jsem provedl výzkum potřeb současných webových GIS z hlediska uživatelů a provozovatelů těchto GIS aplikací. Byl brán v úvahu i současný trend vývoje a tedy i možný budoucí rozvoj.

4.1 Specifika web GIS a jejich vliv na datový model

V této části disertační práce jsou popsány modely, metody a předpoklady, ze kterých jsem vycházel při návrhu datového modelu pro časoprostorová data. Jedná se především o analýzu a porovnání architektury současných klasických velkých geografických systémů, které častokrát používají nejmodernější vědecké přístupy a technologie a malých či středních webových geografických systémů. Následuje souhrn průzkumu zaměření současných webových aplikací a dat, se kterými pracují a z toho vycházejících požadavků na jejich funkčnost. To je pak dále rozvedeno a jsou popsány běžné dotazy na databázi, pro které musí být datový model předně optimalizován.

4.1.1 Architektura Web GIS

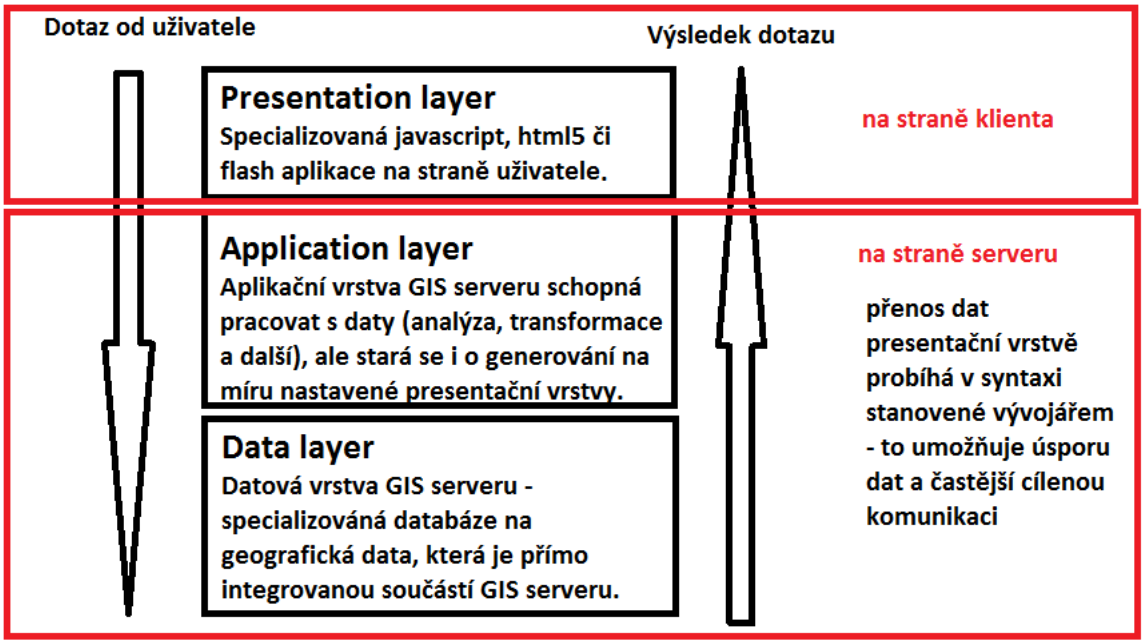
Jak už bylo zmíněno výše při popisu běžné architektury geografických informačních systémů, architektura geografických informačních systémů se dá běžně rozdělit na tři úrovně. Jedná se o úroveň správy dat, která se stará o přímou manipulaci s daty. Úroveň zobrazení, která se stará o zobrazení dat uživateli a veškerou komunikaci s ním. A úroveň zpracování dat, která je mezi těmito vrstvami, se stará se především o veškerou analýzu a další manipulaci s již načtenými daty. Patří do toho transformace těchto dat, tvorba nových dat různými funkcemi jako buffer, intersect či union a jejich dodání presentační vrstvě. U běžného GIS řešení se o všechno stará jeden software a rozdělení těchto úrovní je takto jasně definováno a řešeno a je to i běžně popisovaná architektura, jak v odborných knižních publikacích, tak starších článcích. Bere se to jako zaběhlé rozdělení a prakticky se o jiném rozdělení funkčností nemluví. Situace je však v praxi pro webové geografické systémy velmi jiná. Webové geografické informační systémy se obecně dají rozdělit do tří hlavních skupin.

Webový GIS s GIS serverem

První skupina je, že celý webový geografický systém je provozovaný jako mohutné celistvé řešení. Je to v zásadě port komerčního desktopového GIS systému do webového prostředí a to se také reflektuje v jeho pořizovací a udržovacích nákladech. Takovéto řešení je typické v případě, že je třeba pracovat s opravdu velkým objemem dat. A toto řešení dává uživateli k dispozici i pokročilé funkce GIS. Uživatel typicky může přes webové rozhraní nejen zobrazit geografické informace, ale i s nimi dále složitěji manipulovat, upravovat, ukládat je a provádět s nimi všechny možné úkony stejně jako v klasickém GIS software na desktopovém počítači. V takovémto případě se obvykle architektura webového GIS neliší od desktopové architektury.

Aplikační a datová vrstva jsou zde tvořeny specializovaným webovým GIS serverem. Veškerá data jsou většinou spravována přímo GIS serverem ve specializované geografické databázi či shapefile souborech a ne v externí databázi. Presentační vrstva je tvořena serverem na míru upraveným a vygenerovaným javascriptem, html5 či flash aplikací. Ta, po načtení k uživateli, častokrát po celou dobu zobrazení geografického systému u uživatele průběžně komunikuje v reálném čase s GIS serverem a nechává připravit a odeslat do uživatelova počítače aplikační vrstvou připravená data podle potřeby. To, že aplikace na straně klienta i serveru jsou na sebe úzce navázané, umožňuje častější a optimalizovanější komunikaci mezi nimi. Zobrazovací aplikace na straně klienta si dokáže říci přesně o to, co potřebuje, ve chvíli kdy to potřebuje. To umožňuje držet aplikaci na straně klienta jako čistě presentační vrstvu GIS systému a veškeré operace s daty předávat vlastnímu GIS serveru.

Schéma architektury webového GIS postaveného na specializovaném GIS serveru naleznete na obrázku číslo 7. Kromě množství GIS funkcí je tedy výhoda takovéto celistvé aplikace součinnost, kompatibilita a optimalizace jejich částí a s tím spojená možnost pracovat s velkými objemy dat. Jako příklad takovéhoto řešení lze uvézt především ArcGIS for Server (od roku 2017 nově ArcGIS Enterprise).



Obrázek č. 7 - Architektura webového GIS postaveného na specializovaném GIS serveru

Webový GIS bez GIS serveru

Velký rozdíl mezi GIS běžícími na specializovaných GIS serverech a těmi GIS běžícími na serverech pro webové prezentace je v úrovni zpracování dat a správy dat na straně serveru. U GIS běžících na specializovaném GIS serveru jsou obě tyto úrovně řešeny výrobcem GIS serveru a tvůrce webu k nim může přistupovat pouze velmi limitovaně. Navíc způsobem, který mu vývojář GIS serveru poskytuje. GIS server má tu výhodu, že je to software dělaný přímo na tvorbu webových geografických informačních systémů. Poskytuje vývojáři optimalizované prostředí na správu geografických dat a balíčky funkcí pro analýzu transformace atd. Rozsáhlost funkcí úrovně zpracování dat, které jsou součástí GIS severu, je úzce spjatá s jeho pořizovací cenou a výpočetními nároky, které server potřebuje.

Na druhou stranu mít geografická data na GIS serveru častokrát znamená, že tvůrce geografického informačního systému nemá přístup k tomu, v jakém datovém modelu jsou geografická data uložena, a je nucen používat ten datový model, ke kterému mu server dává přístup. To může způsobovat značné problémy, zvláště pro prostorová data s integrovaným faktorem času, neboť většina GIS aplikací (ať už desktopových nebo serverových) umožňuje uživateli pracovat s geografickými daty pouze v relacích, kde řádky odpovídají jednotlivých objektům a sloupce jejich atributům, přičemž vlastní geografická data jsou uživateli skryta pod jedním z atributů. Strukturováním dat do takovéto tabulky kde co řádek to objekt však s

přidáním faktoru času můžeme, jak již bylo popsáno v předchozích kapitolách, způsobit značnou redundanci dat, či způsobit obtížnou tvorbu potřebných dotazů nad těmito daty. Navíc značná výhoda GIS serverů, totiž že úroveň správy dat je optimalizována na geografická data, se častokrát nevztahuje na data s integrovaným faktorem času (Sarup a Shukla 2012, Emhmed et al. 2012).

Při tvorbě geografického informačního systému založeném na serveru pro webové prezentace značně záleží, jaké nástroje máme k dispozici. Úroveň zpracování dat typicky bude implementována ve skriptovacím jazyce na straně serveru - mezi nejpoužívanější patří PHP (v ČR je PHP k nalezení prakticky na libovolném webhostingu). Pro náročnější aplikace pak lze použít jazyky Ruby, Rails či Python, nicméně pro ty je často třeba mít více specializovaný server. Velká nevýhoda je, že funkce pro práci s geografickými daty nemusí být v daném jazyce implementována. Častokrát také kvůli tomu, že webový skriptovací server nemá výpočetní kapacitu nutnou pro provedení těchto funkcí, či jejich zpracování na webovém serveru by trvalo déle než je vhodné pro webové aplikace, kde se očekává okamžitá odpověď. Nejsme tolik vázáni prostředím přednastaveným nějakou třetí stranou jako i GIS serveru a řadu funkcí lze snadno vytvořit na míru, nicméně řadu funkcí často není možno získat s takovou funkčností a optimalizací, jak jsou k dispozici, když jsou součástí masivního GIS serveru.

Úroveň správy dat je z části spravovaná systémem pro řízení báze dat a z části skriptovacím jazykem, který zajišťuje z části i úrovně data services a mediator v úrovni správy dat. Mezi nejpoužívanější databázové systémy ve webových službách patří MySQL. Z dvaceti českých webhostingových firem dvacet poskytuje kombinaci PHP a MySQL jako primární nabídku skriptovacího jazyka a databáze pro webhosting. Pouze některé pak umožňují zvolit si další databáze jako volitelné. Jako další databázové systémy používané pro webové služby lze uvést například ORACLE a PostgreSQL. Díky tomu, že k datům nepřistupujeme skrze GIS server, ale máme přístup přímo k systému řízení báze dat, můžeme navrhnout datový model vyhovující požadavkům specializovaných aplikací - typicky právě aplikacím s integrovaným faktorem času. Je nutno zmínit, že databázové systémy běžně používané pro webhosting jsou typicky pouze relační databáze, což přináší řadu omezení při navrhování vhodného datového modelu. Základní charakteristiky jsou shrnuty v tabulce č. 16 (Bandyophadyay et al. 2012, Komárková et al. 2010, Singh et al. 2014).

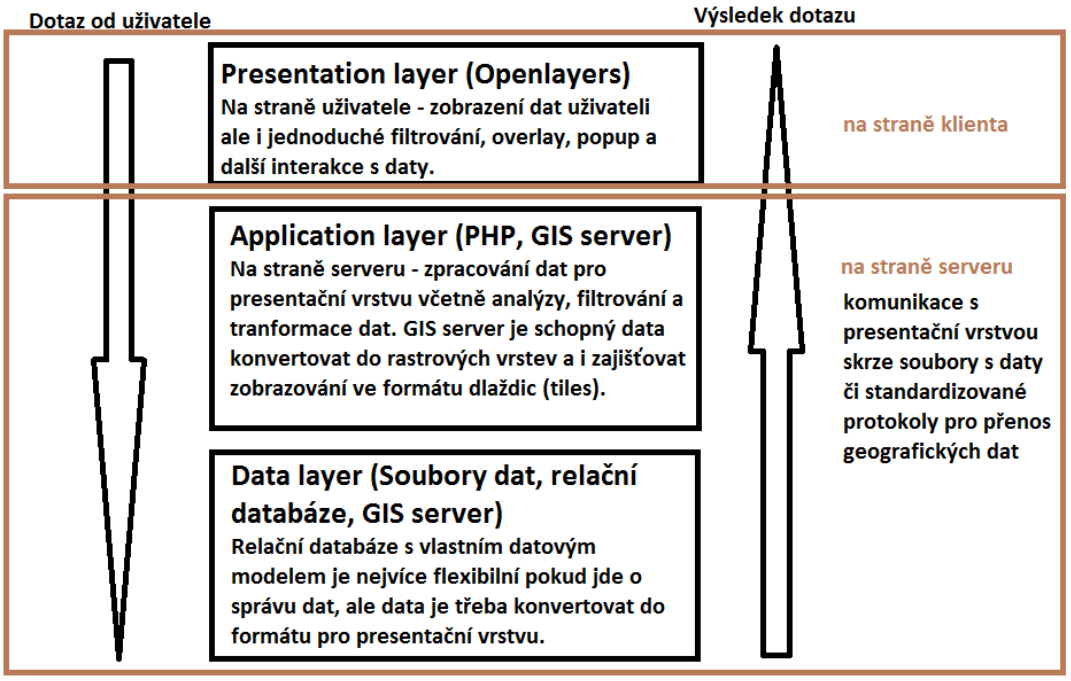
GIS server a jeho nástroje	• Třetí stranou vytvořený software častokrát se značně velkou uživatelskou komunitou a technickou podporou. • Úroveň správy dat, úroveň zpracování dat a zobrazovací úroveň jsou často vyvíjeny jednou firmou a jsou optimalizovány na práci mezi sebou. • Často velké pořizovací náklady. • Nutnost běhu speciálního software na serveru - tj. nutnost mít server - větší náklady na provoz. • Software GIS serveru je optimalizován na práci s geografickými daty. • Veškeré nastavení GIS serveru se dělá přes jeho rozhraní, což umožňuje snazší tvorbu výsledné uživatelské aplikace, ale nelze příliš vybočovat z kolejí nastavených GIS serverem (zvláště pokud jde o úroveň správy dat). • GIS server často obsahuje velké množství GIS analytických a transformačních funkcí. • Často malá podpora integrace času a nemožnost ji snadno vytvořit.
Webový server a jeho nástroje	• Na míru vyvíjený software se všemi výhodami a nevýhodami co z toho plynou. Respektive často výsledný geografický systém je sestavený z řady aplikací třetích stran a na míru vyvinuté aplikace, která je spojuje. Je nutné doprogramovat řadu věcí, které jsou v balíku služeb, který GIS server poskytuje již řešeny. • Úroveň správy dat, úroveň zpracování dat a zobrazovací

	úroveň jsou často samostatné moduly a je třeba se zabývat jejich kompatibilitou. • Malé pořizovací náklady. • Často větší limitace ze strany výpočetního výkonu serveru. • Výsledná GIS aplikace dělá přesně a pouze to co po ní vyžadujeme, takže i když často její součásti nejsou až natolik optimalizované jako komplexní řešení, tak díky tomu, že nemají až tak rozsáhlý Framework, jsou vlastně často mnohem úspornější. • Běží na běžně dostupných webhostingových serverech s minimálními udržovacími náklady. • Data v datovém modelu jaký zvolíme a možnost přímého přístupu k datům. Včetně vytvoření datového modelu, který podporuje integraci faktoru času. • Je možno integrovat snadno s dalšími aplikacemi jako jsou například systémy pro podporu rozhodování.

Tabulka č. 16 - Porovnání tvorby geografického informačního systému pomocí GIS serveru a na míru vytvořené aplikace na webovém serveru.

Kombinace GIS serveru a řešení na míru

Třetí typ možné architektury webového geografického informačního systému, se kterým se můžeme v praxi setkat, je postaven okolo GIS serveru. Nejedná se ale o celistvé řešení jako v prvním případě. Jedná se typicky o opensource GIS server, který má typicky mnohem méně funkcí než komplexní komerční GIS server. GIS server se stará pouze o funkci aplikační vrstvy, či případně aplikační a datové vrstvy. To jest ve webovém prostředí, které funguje na bázi klient-server modelu (client-server model), se stará opravdu pouze o stranu serveru (server side). Navíc může být GIS server napojený na separátní databázový systém, který se stará o vlastní data a tedy o datovou vrstvu architektury. Strana klienta (client side), je téměř vždy tvořena jiným software řešením. V praxi se v naprosté většině případů používá opensource javascriptová knihovna Openlayers. Zmiňuji zde opensource řešení, ale komerční GIS servery mohou také být použity pro tento typ architektury. Nicméně komerční řešení, jak bylo zmíněno, obsahuje při pořízení komplexní aplikační zázemí s na míru vytvořenou presentační a datovou vrstvou a tyto vrstvy jsou optimalizovány pro práci spolu. Tím pádem pokud pořizujeme komerční webový geografický informační systém, tak se typicky použije jako komplexní řešení. Schéma architektury webového GIS postaveného na kombinaci GIS serveru a klasického webového serveru a databáze naleznete na obrázku číslo 8.



Obrázek č. 8 - Architektura webového GIS postaveného za pomoci opensource nástrojů

Presentační vrstva - knihovna Openlayers

Knihovna Openlayers umožňuje celou řadu funkčností na straně klienta, jako je shlukování, podrobné nastavení zobrazení jednotlivých vrstev, či filtrování v již získaných datech (Bandyophadyay et al. 2012, Openlayers 3 nedatováno). Knihovna zařizuje veškeré zobrazování a práci s daty. Pouze se inicializuje a nastaví se, odkud má brát data. Následuje příklad nastavení základní inicializace a načtení vrstvy ve formátu GeoJSON se základním nastavením mapy.

```
var prizemi = new ol.layer.Vector({
     source: new ol.source.Vector({
          url: 'layers/prizemi.geojson',
          format: new ol.format.GeoJSON()
     }),
     style: mainstyle
});
map = new ol.Map ({
     controls: ol.control.defaults({}, []),
     layers: [prizemi],
     target: document.getElementById('map'),
     view: new ol.View({
          center: [1607019, 6462421],
          zoom: 20,
          minZoom: 19,
          maxZoom: 22,
          extent: [1606949,6462321,1607149,6462521]
     })
});
```

Komunikace s Presentační vrstvou - JSON / KML

Jak již bylo zmíněno výše v popisu architektury webového geografického informačního systému, web GIS systémy mohou komunikovat s presentační vrstvou v zásadě dvěma hlavními způsoby. Pokud presentační vrstva a aplikační vrstva jsou tvořeny aplikací od jedné firmy, tak jsou navrženy na to spolu pracovat a mohou typicky komunikovat svoji vlastní specifickou syntaxí. To umožňuje, aby presentační vrstva komunikovala průběžně s aplikační vrstvou a delegovala na ní více práce, tak jak je tomu u newebových GIS. Pokud ale je

presentační vrstva samostatná knihovna jako jsou Openlayers, tak ta je omezená na běžně používané formáty pro přenos geografických dat. Aplikace na straně uživatele typicky dostane od aplikační vrstvy geografického systému data, typicky v podobě souboru geografických dat a s těmi pak pracuje. Veškeré filtrování, editaci objektů, a další funkce se dělají na straně uživatele a akorát změny, které se mají uložit, se posílají zpátky serveru. Většinou opět ve stejném formátu, jako aplikace na straně uživatele data dostala. Nejběžněji používané volné formáty pro přenos vektorových dat jsou KML a JSON.

KML (Keyhole Markup Language) je notace XML, speciálně upravená pro ukládání prostorových dat. Původně to byl jazyk navrhnut firmou Keyhole pro zobrazení map na internetu. Později byla firma Keyhole koupena firmou Google, která následně KML adoptovala a používá pro své webové služby. Díky tomu je KML dnes jedním z nejpoužívanějších webových GIS formátů. KML se následně stalo mezinárodně uznávaným standardem pro uchovávání geografických dat pod záštitou Open Geospatial Consortium.

KML doplňuje jazyk GML. Zatímco GML se používá jako standard pro přenos geografických dat přes web. KML zároveň s možností přenášet čistá geografická data doplňuje o informaci o tom, jak se mají zobrazovat (Keyhole Markup Language nedatováno).

Základní podoba dat v KML souboru je následující:

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2">
<Placemark>
  <name>Karlštejn</name>
  <description>Hrad Karlštejn</description>
  <Point>
    <coordinates> 49.939579, 14.187491</coordinates>
  </Point>
</Placemark>
</kml>
```

Příklad ukazuje strukturu KML souboru s jedním objektem (se jménem a popiskem), který je určen jedním bodem na mapě. KML předpokládá použití světového souřadnicového systému WGS 84.

GeoJSON je otevřený formát založený na používaném JavaScript Object Notation, obdobně jako KML upraveném speciálně pro ukládání prostorových dat včetně jejich neprostorových atributů. GeoJSON není mezinárodně uznávaný standard, ale je tvořen komunitou internetových developerů (GeoJSON nedatováno). Syntaxe je kompaktnější než u KML což také přispívá k jeho popularitě.

Příklad syntaxe:

```
{
  "type" : "Feature",
  "id" : 12,
  "geometry" : {
     "type" : "Polygon",
     "coordinates" : [
       [
          [ 1607030.7178240754, 6462439.1382718645 ],
          [ 1607030.338681031, 6462439.0696924962 ],
          [ 1607030.2062408291, 6462440.133022856 ],
          [ 1607024.3434382677, 6462439.4027975462 ],
       ]
     ]
  },
  "properties" : {
     "ID" : 12,
     "Sekce" : "12130",
     "Mistnost" : "81",
     "sekcejmeno" : "Odbor vnitřní správy",
  }
}
```

4.1.2 Výzkum potřeb a specifik současných web GIS aplikací

Jak již jsem zmínil v kapitole motivace, již skoro dvacet let se živím tvorbou webových aplikací. Během této doby jsem se podílel na celé řadě projektů a od doby co jsem se při svém studiu začal zabývat geografickými informačními systémy, jsem to samozřejmě začal

promítat i do své práce. Za posledních deset let jsem vytvořil řadu webů, které používali v různé míře a různým způsobem geografická data včetně dat časoprostorových. Získal jsem při tom také celou řadu kontaktů, které jsem pak využil při výzkumu současného prostředí, využití a trendů webových GIS aplikací. V této kapitole tedy kromě použití svých zkušeností z praxe vycházím z nestrukturovaných rozhovorů, jak s uživateli a administrátory webů obsahující časoprostorová data, tak s dalšími lidmi, kteří se danou problematikou zabývají. Bohužel webových informačních systémů, které publikují v čase se měnící geografická data je velmi málo, zvláště v České republice. Navíc jsou to často informační systémy zabývající se velmi různorodými věcmi, takže možnost vézt rozhovory s lidmi zabývajícími se tímto tématem není mnoho. Proto jsem se na začátku mého výzkumu přiklonil k tomu vést nestrukturované rozhovory s velkou mírou volnosti.

Jako příklad mohu uvézt například několik rozhovorů, které jsem měl možnost vést se systémovými inženýry na Ministerstvu zemědělství ČR o jejich praxi a vizi do budoucna ohledně nakládání s geografickými daty a s tím, jak nakládají s historickými daty. Tyto rozhovory z hlediska výzkumu byly jedny z nejužitečnějších, protože jsem se z nich dozvěděl stav a technické zázemí řady GIS aplikací ve státní sféře, mi bohužel nebylo dovoleno nahrávat a mám z nich pouze zkrácený písemný záznam. Z těchto rozhovorů nakonec vzešla další spolupráce a momentálně na Ministerstvu zemědělství pracuji jako systémový inženýr zabývající projekty týkající se právě geografických informačních systémů s v čase se měnícími daty.

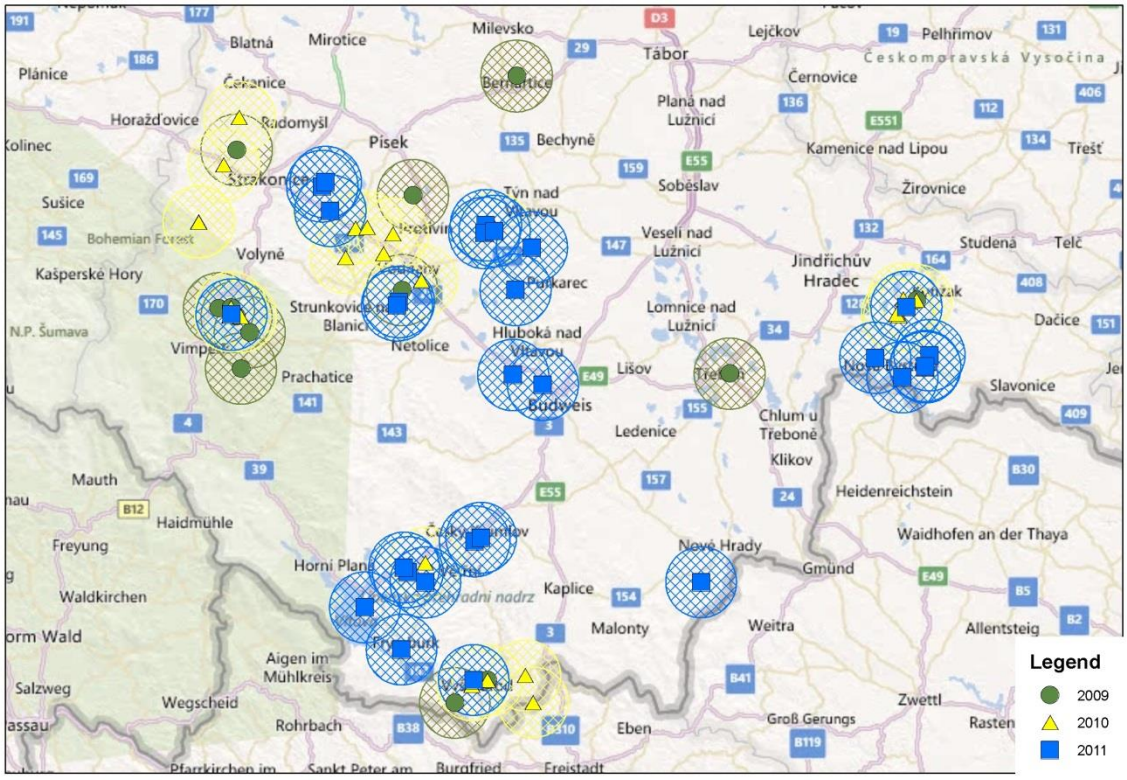
Celkově jsem zjišťoval, s jakým druhem dat pracují, jak tato data zpracovávají či by chtěli v budoucnu zpracovávat a jaké datové modely se používají. V následujícím textu nezmiňuji přesná čísla, ale pouze odhady. Například typu, že více než 60% aplikací používá integraci času na úrovni relace. Je to z důvodu, že z rozhovorů jsem častokrát získával informace z druhé ruky. Například ve zmíněných rozhovorech s lidmi zabývajícími se GIS na Ministerstvu zemědělství jsme mluvili o podobě celé řady systémů, se kterými tam přišli do styku, ale nemám přesné statistiky, protože častokrát ne všechny údaje byly dostupné. Webový GIS je sám o sobě zatím poměrně málo používán a webové GIS s dynamickými daty obzvláště. Měl jsem možnost osobně studovat do většího detailu strukturu a architekturu pouze 18 takovýchto GIS aplikací. Proto zakládám následující analýzu GIS aplikací ve webovém prostředí na expertním odhadu založeném z části na rozhovorech s experty a pouze z části na průzkumu existujících aplikací.

Zobrazení časoprostorových dat na webu

Moderní webové GIS aplikace a často ani aplikace desktopové, nepodporují mnoho způsobů jak zobrazit a pracovat s v čase se měnícími daty. Běžně používaná zobrazení se dají rozdělit do následujících kategorií:

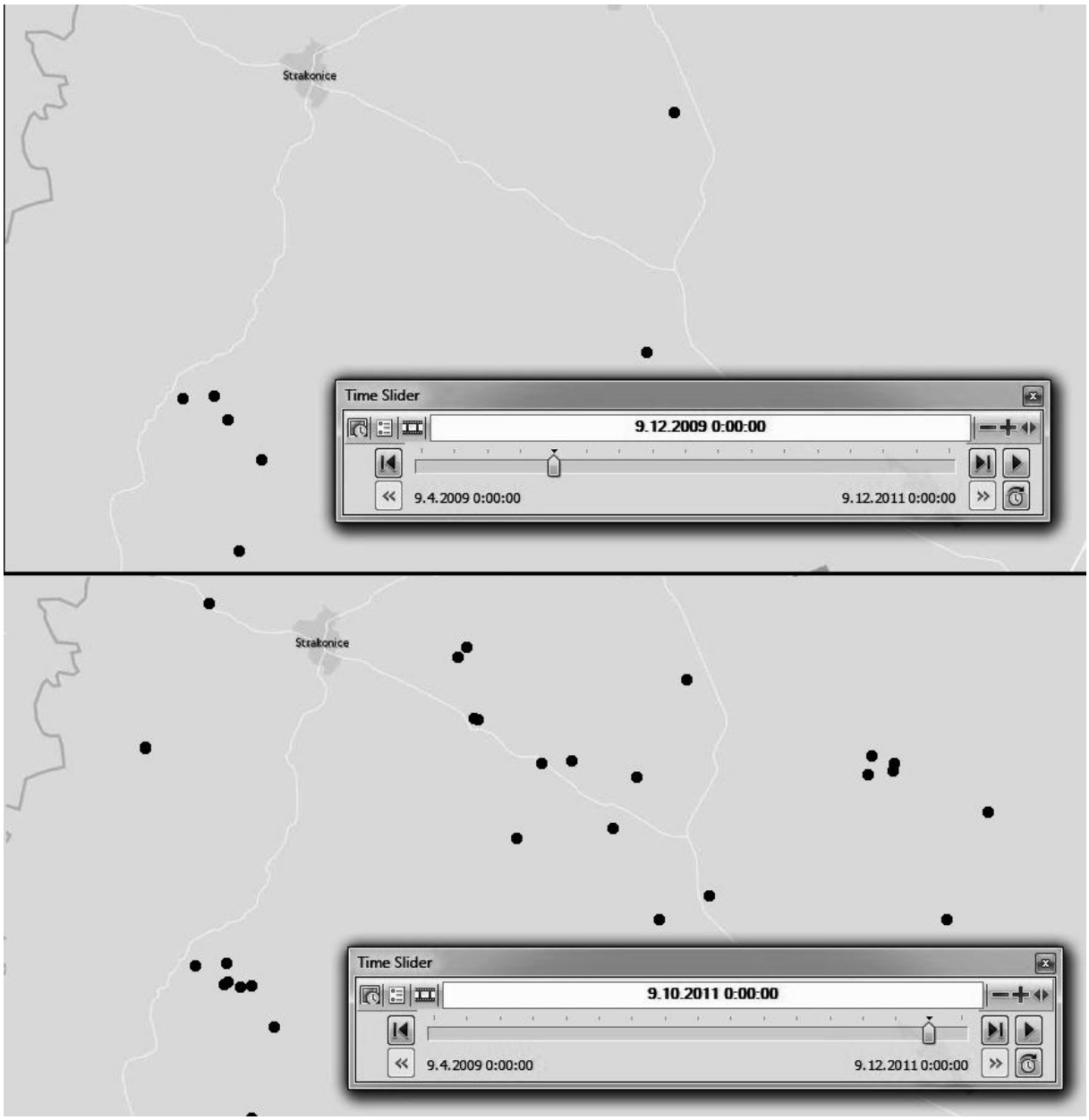
Zobrazení různých časových období ve zvláštních vrstvách - umí jakákoli aplikace GIS, protože vlastně není třeba jakákoli podpora faktoru času. Jedná se o úplně nejjednodušší způsob, kdy data strukturujeme do několika časových intervalů a zobrazujeme je v oddělených vrstvách. Data pro jednotlivé časové vrstvy jsou od sebe oddělena a dochází k jejich značné redundanci. Většinou je ideální data v takovéto formě neuchovávat, ale vždy jen jejich potřebnou část vybrat dotazem z databáze, kde mohou být uloženy v nějakém pokročilejším datovém modelu. Z vybraných dat se pak vytvoří na straně serveru na úrovni zpracování dat soubory s daty, které se předají presentační vrstvě na straně uživatele, která se stará o jejich zobrazení. Tento přístup je výhodný pokud se jedná o data, kde se objekty často mění, ale nemění se často jen některý z atributů, či jen některý z objektů (Tang et al. 2010, Atay 2010). Blíže je vysvětleno v kapitole, která se zabývají se databázemi pro časová data.

Mít data strukturována v jiném datovém modelu a strukturovat je do souborů pro presentační vrstvu vždy, když je uživatel chce zobrazit, sice je výpočetně mnohem náročnější, ale umožňuje kromě daleko snazší údržby dat také libovolně měnit časové rámce (vrstvy), do kterých jsou data strukturována. Není tedy problém například z rozdělení "co vrstva to čtvrtletí" přejít na "co vrstva to měsíc". Samozřejmě pokud se jedná o data v málo časových rámcích a neočekávají se časté aktualizace, je výhodnější uchovávat časová data ve stylu co časový rámec to zvláštní soubor, přímo ve stejném formátu jako je potřebný k jejich zobrazení. Příklad zobrazení toho typu je na obrázku č. 9. Jedná se o výstup z jedné z aplikací, které jsem vytvořil - tři vrstvy zobrazující výskyt včelího moru v jižních Čechách.



Obrázek č. 9 - zobrazení v čase se měnících dat s integrací času na úrovni vrstvy

Další možnost, jak jsou v čase se měnící data na webu často strukturována, je mít je v jedné vrstvě a pomocí posuvníku (slideru) měnit jejich zobrazení. Tento přístup umožňuje uživateli si zvolit přesný bod či interval v čase, ve kterém chtějí geografická data vidět. Data pro toto zobrazení nemohou být presentační vrstvě předána ve stejném formátu jako v předchozím případě, ale musí být strukturována do jedné tabulky s časem platnosti na úrovni jednotlivých řádků. To přináší mnohem menší redundanci dat (i když častokrát stále značnou), než když je čas integrován na úrovni vrstev. Navíc se nemusí data rozřazovat do neměnných časových úseků, ale můžeme zjistit či zobrazit od kdy přesně byla platná a kdy jejich platnost skončila.



Obrázek č. 10 - Zobrazení v čase se měnících dat v ArcGIS 10.1 pomocí slideru

Tento způsob zobrazení časoprostorových dat je již často podporován u klasických komerčních GIS programů. Obrázek č. 10 ukazuje funkci time slider v jednom nejrozšířenějších GIS komerčních programů - ArcGIS 10.

Použití slideru, přestože se zdá poměrně základní, ale není u většiny zobrazovacích GIS webových aplikací implementován. Většina z nich však umožňuje filtraci dat a má možnost tuto funkčnost doplnit jako rozšíření napojené přes jejich API. Skriptů, které takovéto použití slideru pro práci s časoprostorovými daty zajišťují je k dispozici velmi málo. Za zmínku stojí například knihovna Timemap, která lze napojit na Openlayers javascriptovou zobrazovací

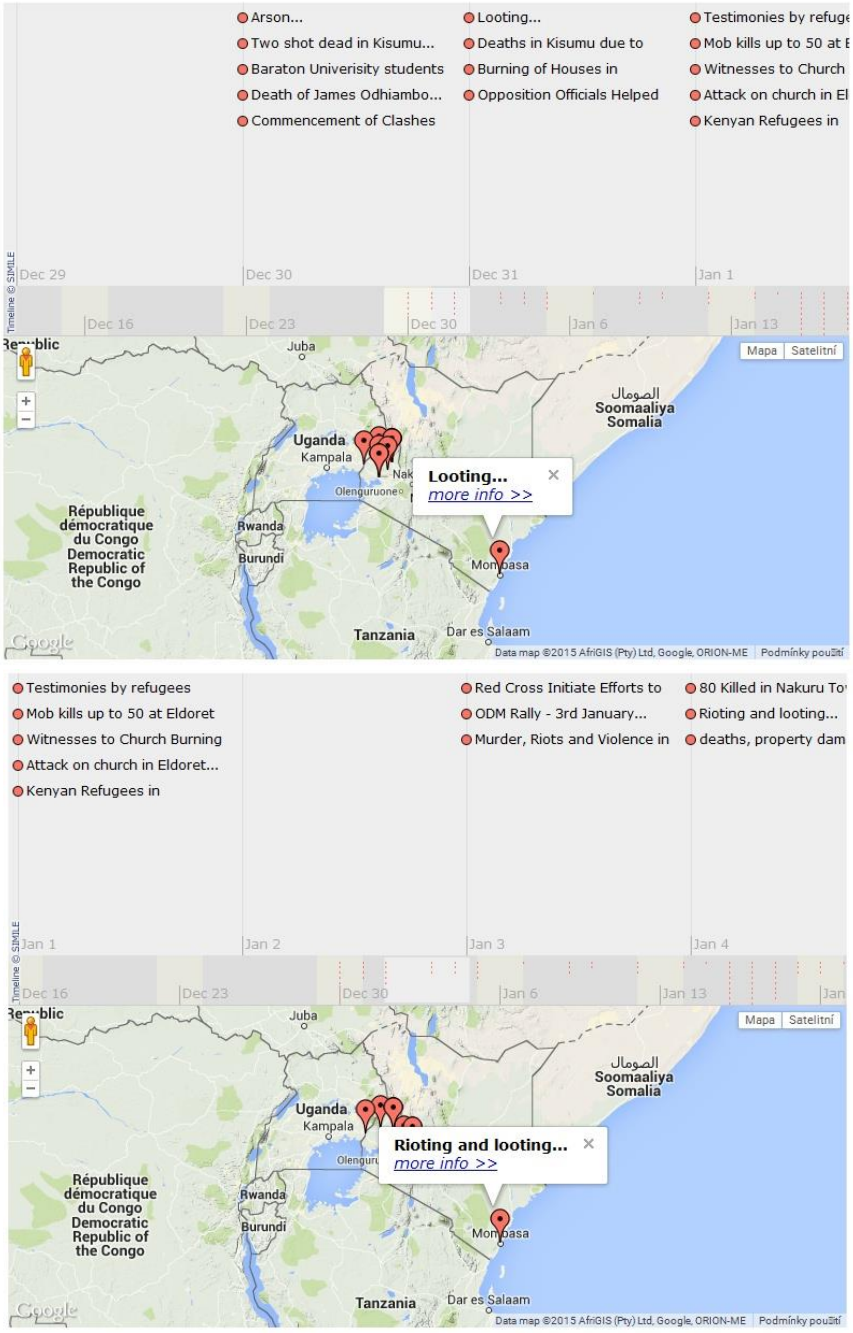
knihovnu a umožňuje filtrovat data podle vybraného času platnosti. Na obrázku č. 11 je možno vidět podkladovou mapu Google a na ní vyznačené body s výskytem nepokojů v Kenyi na přelomu roku 2007 a 2008 zobrazené pomocí javaskriptové knihovny Openlayers. Nad mapou zobrazovanou skriptem Openlayers se nachází část tvořená skriptem Timemap, kde je možno volit z jakého časového úseku se budou data na mapě zobrazovat, to je pak předáno jako parametry filtrování objektů ve vrstvě Openlayers, která tento filtr promítne do mapy. Vrstva obsahující nepokoje je skriptu Openlayers předána ve formátu KML. Pro ilustraci následuje podoba jednoho objektu:

```
<Placemark>
     <Point>
          <coordinates>
               34.553300,0.581000
          </coordinates>
     </Point>
     <name>
          Gun shots...
     </name>
     <TimeStamp>
          <when>
               2008-01-16
          </when>
     </TimeStamp>
     <description>
          <a
href="http://legacy.ushahidi.com/incident.asp?id=68"
target="_blank">more info >></a>
     </description>
</Placemark>
```

Můžeme z této datové struktury vyčíst, že čas platnosti zde není zadán pomocí intervalu, ale jedná se pouze o jedno datum a počítá se, že incidenty budou mít pro tento model trvání jednoho dne. Kromě času platnosti incidentu a jejich koordinátu (incidenty jsou zde zobrazeny jako body, resp. tedy zeměpisná šířka a délka ve stupních) máme k dispozici dva

atributy. Jedná se o jméno incidentu a jeho popisek, který obsahuje odkaz na jeho detail na externí webové stránce.

Skript Timemap nám v horní části jasně ukazuje kolik objektů, resp. nepokojů, je zobrazeno, a kdy se uskutečnili rozřazením jejich jmen do jednotlivých dnů (Timemap javascript library nedatováno).



Obrázek č. 11 - skripty Openlayers a Timemap zobrazující nepokoje v Kenyi (pořízeno z ukázky funkčnosti skriptu na domovské stránce Timemap (Timemap javascript library nedatováno))

Stejně jako v u integrace času na úrovni vrstvy, podoba v jaké skripty při této formě zobrazení vyžadují data strukturovat, není úplně ideální. Čas je zde sice integrován na úrovni řádku tabulky, ale pro funkčnost zobrazovacího skriptu je třeba, aby veškerá data, se kterými chceme pracovat, byla v jedné velké relaci, i když jejich uspořádání neodpovídá ani vzdáleně normální formě a může opět docházet ke značné redundanci dat.

Třetí způsob zobrazení časoprostorových dat na webu je v podobě časových řad, respektive zobrazení historického vývoje objektu. Na webu se častokrát takto nezobrazuje geografická složka objektu, ale pouze vývoj neprostorových charakteristik objektu. Pokud se zobrazuje prostorová složka, pak se většinou zobrazuje jako několik map, kde každá mapa zobrazuje objekt v jedné fázi jeho vývoje v historii. Případně některé aplikace zobrazují změnu, kde ukazuje vždy na mapě dvě vrstvy - objekt před změnou v jedné a objekt po změně v druhé vrstvě. Zobrazení vývoje objektu v čase je často velmi užitečné, ale velmi málo webových aplikací ho umožňuje. Dá se usuzovat, že to je primárně tím že datové modely pro uchovávání dat nepodporují nebo velmi špatně podporují dotazy, které by získali potřebná data. Momentálně nejvíce rozšířený způsob - uchovávání časoprostorových dat na úrovni relací, kde co vrstva to jeden časový rámec, se na zobrazení historie změn objektu naprosto nehodí. Pro získání změn objektu při takovémto datovém modelu a strukturování dat, by bylo třeba projít každou vrstvu a zjistit zda se daný objekt změnil či nikoli. Vzhledem k tomu, že co vrstva to soubor či tabulka v databázi, je zřejmé, že pro 60 časových vrstev by se muselo použít 60 dotazů a tedy zpracovat 60 vrácených výsledků.

Závěry důležité pro návrh datového modelu

Ve výsledku, analýza stávajících webových aplikací přinesla řadu důležitých poznatků pro tvorbu nového datového modelu speciálně navrženého pro uchovávání časoprostorových dat pro webové GIS aplikace. Jedním z největších přínosů je získání obvyklých zobrazení dat a tím pádem získání základních potřeb dotazů, které budou používány aplikací pro získávání dat z databáze. Na základě těchto potřeb je mimo jiné model v následující kapitole postaven a následně optimalizován. Jak ze zkušenosti, tak z výzkumu existujících webových GIS jsem zjistil, že pokud zobrazují nejenom historická, ale i aktuální data, tak zdaleka nejčastější dotaz je na zobrazení aktuálního stavu dat. Je to i dáno tím, že webová GIS aplikace při načtení typicky zobrazuje aktuální stav a teprve na základě požadavku uživatele změní pohled na historická data.

Druhý nejtypičtější dotaz je na podobu dat v určitém časovém okamžiku. Uživatel si zvolí časový okamžik, ať už pomocí posuvné lišty (slideru), nebo vybráním data ve formuláři a aplikace načte podobu mapy - objektů a jejich atributů, jak vypadaly ve vybraném časovém okamžiku.

Velmi používaný je i dotaz na jeden objekt a jeho historii. To jest časovou řadu změn objektu. Toto je velmi typické pro atributy objektu a málo pro zobrazení geografických změn. Ve většině zkoumaných aplikací se změny ve tvaru objektu pouze vypíší, podobně jako změny jiných atributů, ale nezobrazí se jejich podoba. Pouze minimum ze zkoumaných aplikací zobrazovalo změny za pomocí více map - mapa pro každou verzi tvaru objektu. Jednalo se pouze o aplikace, kde bylo z charakteru dat zřejmé, že změn nebude mnoho, protože načítání více mapových rámů na jednu stránku je náročné z hlediska výpočetních zdrojů. A pouze jedna ze z koumaných aplikací umožňovala si na změnu kliknout, přičemž to následně ukázalo dvě mapy - před změnou a po změně, což dobře obcházelo výše zmíněný problém. Tento typ dotazu je velmi používaný při zobrazení detailu objektu - typický případ použití je, že se zobrazí mapa všech objektů v určitém čase a uživatel může objekt zvolit kliknutím. Načež se otevře popup s detaily objektu - typicky obsahuje pouze kratší popis a pouze základní historii objektu. Případně aplikace dále umožňuje zobrazit samostatnou stránku objektu s jeho veškerými detaily.

Další, ale již méně používaný dotaz je dotaz na zobrazení objektů na mapě v určitém časovém intervalu. Typicky se tento dotaz používá, pokud víme, že změny v prostoru se vyskytují jen jednou za časový interval, ale změny atributů objektů vícekrát. Pak můžeme chtít uživateli nechat zvolit ne pouze časový bod kdy mapu zobrazit, ale interval, ve kterém provádíme statistické hodnocení atributů. Je třeba brát v úvahu, že pokud se objekt změnil po prostorové stránce v průběhu zvoleného intervalu, tak bude zobrazena pouze jedna verze jeho tvaru (typicky ta na začátku nebo na konci zvoleného intervalu). Příkladem využití může být geografický systém zemědělských polí, kdy víme, že změna hranic pole se provádí vždy pouze na začátku roku. Ale měření kvality půdy, znečištění a dalších faktorů se provádí nejméně jednou za měsíc. Potom uživateli můžeme chtít dovolit zobrazit nejen mapu, kde si uživatel zobrazí pole zabarvená podle hodnot jejich znečištění v určitý okamžik, ale i mapu, kde si uživatel vybere zobrazit pole zabarvená podle průměrné hodnoty znečištění v daném intervalu.

Velmi málo zkoumaných webových aplikací umožňuje dotazy na další objekty, které souvisejí se změnou hranic daného objektu v čase. To jest například, aby webová aplikace umožňovala zjistit, že změna tvaru pole A bylo ve skutečnosti zvětšení díky sloučení pole A s polem B. Současné webové aplikace zjistit charakter změny téměř nikdy neumožňují, protože to běžně používané datové modely neumožňují jednoduše zjistit. Z rozhovorů, které jsem vedl, jak administrátoři, tak uživatele řady webových GIS aplikací toto zobrazení považovali za užitečné a důležité.

Následující seznam shrnuje možné dotazy uživatele na webovou GIS aplikaci. Jsou řazeny podle očekávané priority:

1. Zobrazení aktuálního stavu objektů - tj. zobrazení vrstvy s objekty (mapy), jak momentálně vypadá ve skutečnosti.

2. Zobrazení aktuálního stavu objektů s filtrací podle atributů objektů či filtrací podle asociací s jinými objekty nebo atributy těchto objektů.

3. Zobrazení stavu objektů v určitém časovém okamžiku - tj. uživatel například zvolí datum a aplikace mu ukáže mapu se stavem objektů, jak tehdy vypadaly.

4. Zobrazení stavu objektů v určitém časovém okamžiku s filtrací podle atributů objektů či filtrací podle asociací s jinými objekty nebo atributy těchto objektů.

5. Zobrazení stavu objektů v určitém časovém intervalu - tj. uživatel zvolí časový interval a systém ukáže mapu, jak v daném časovém intervalu vypadala.

6. Zobrazení stavu objektů v určitém časovém intervalu.

7. Zobrazení historie vybraného objektu s filtrací podle atributů objektů.

8. Zobrazení detailu dané změny objektu - charakter změny a objekty které se na něm podílely.

9. Zobrazení sousedících objektů zvoleného objektu - tj. objektů, jejichž hranice se dotýkají nějaké části hranice zvoleného objektu.

10. Ostatní - například zobrazení limitované podle lokace.

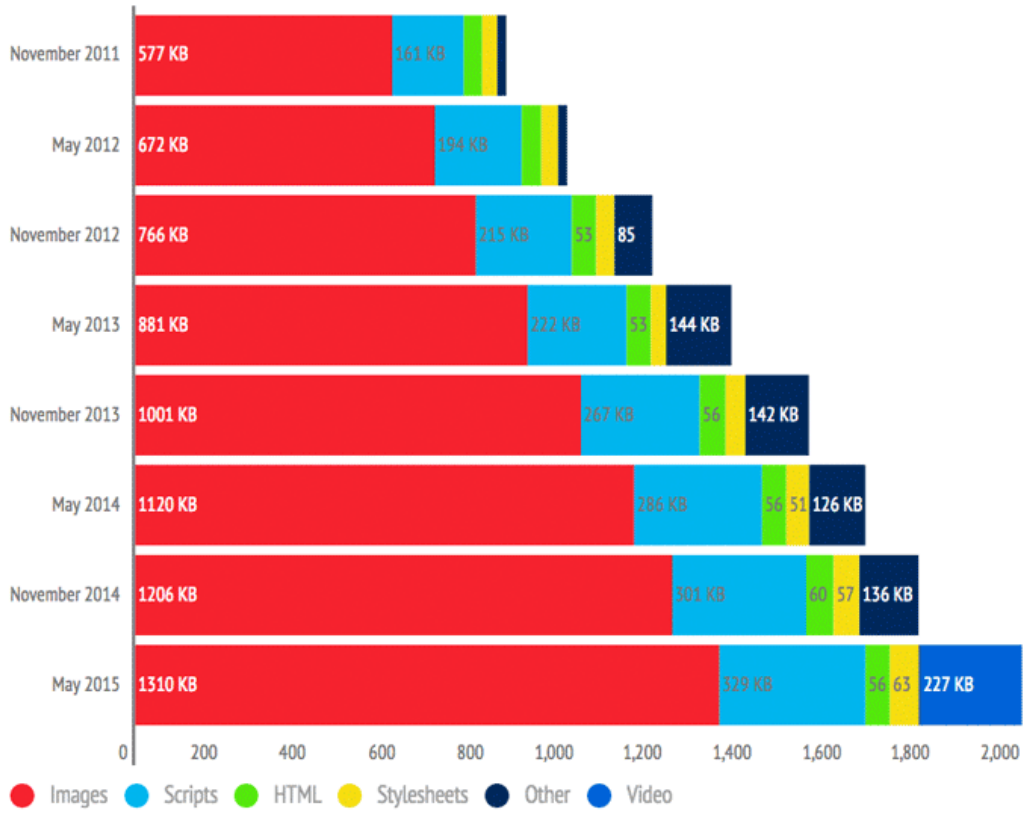
Kromě výzkumu možností, v jaké podobě webové GIS aplikace zobrazují uživateli data. Bylo třeba pro vývoj datového modelu stanovit limit způsobené specifiky webového prostředí. Největším limitem je bezesporu to, že webová stránka je typicky brána jako webová prezentace - počítá se zde s minimální prodlevou mezi žádostí uživatele o zobrazení webové stránky a jejím zobrazením. Navíc k webové stránce může přistupovat velké množství lidí najednou. Tím pádem webový server má často velmi omezené systémové prostředky a nelze provádět velké množství složitých operací. Navíc nelze jednoduše delegovat výpočty na stranu uživatele bez specializované aplikace, protože webové aplikace na straně uživatele jsou velmi náročné na výpočetní prostředky (typicky se jedná o aplikace v javascriptu). To je také důvod proč geografické systémy teprve v posledních letech dostávají využití ve webovém prostředí. Se zvyšujícími se výpočetními prostředky, jak na straně uživatele, tak na straně serveru se dá očekávat, že se webové geografické systémy budou objevovat stále více a více.

Ve stále se měnícím webovém prostředím lze velmi těžko stanovovat nějaké limity rychlosti načítání webových stránek pro uživatele, navíc geografické aplikace lze považovat za zvláštní případ, kde se očekává, že budou mít pomalejší odezvu než běžné webové stránky. Nicméně i tak lze stanovit alespoň přibližné očekávané hranice. Pro různé účely při návrhu a testování datového modelu jsem stanovil jako limit pro únosnou dobu načítání 2-3s. Řada statistik na internetu uvádí 2s jako průměrnou dobu načítání webové presentace komerční webové stránky v roce 2015 a 2016, přičemž se běžně mluví o této době načítání jako o vážném problému, který by se měl adresovat. Většina této velikosti jsou javascripty (typicky různé skripty sbírající statistiky) a obrázky a další grafika - viz obrázek č. 12 (Soasta web optimalizations and tests nedatováno).

Google v několika vyjádření uvádí jako doporučenou průměrnou dobu načítání půl sekundy, nicméně momentálně je velikost stránky vyhledávače google 1,9 MB a načítá se v průměru 1,5s (testováno na několika druzích připojení). Velmi podobně je to s velikostí i dobou načítání například český novinový portál iDnes. Z osobní zkušenosti a po rozhovorech s experty v oblasti GIS jsem si tedy stanovil za cíl, že doba načítání webové GIS aplikace by neměla přesáhnout 3 sekundy a 5 MB dat. Ideálně by však měla mít dobu načítání méně než 2s a celkovou velikost 3 MB dat.

Provedl jsem měření na řadě case study a projektů co jsem měl k dispozici a 100kb dat se rovná zhruba 100 objektům v úsporném režimu formátu GeoJSON, kde každý objekt je

polygon skládající se zhruba z 10 bodů a každý objekt má 12 atributů malé velikosti (číslo či textový řetězec do deseti znaků). Další zjištěné velikosti jsou popsané v tabulce č. 17.



Obrázek č. 12 - Statistiky průměrné velikosti webových stránek a jejich složení (Soasta web optimalizations and tests nedatováno)

Formát souboru	**Velikost**	**Počet prvků (polygon 10 bodů, 12 atributů s hodnotami o jednom slově)**
GeoJSON úsporný bez symboliky	100 kB	100 prvků
GeoJSON strukturovaný pro snadné čtení bez symboliky	250 kB	100 prvků

KML bez symboliky	240 kB	100 prvků
GeoJSON úsporný bez symboliky	3 MB	3000 prvků
GeoJSON úsporný bez symboliky	3 MB	Ve veškerých projektech i case study vystačily 3 MB na alespoň 500 prvků i s velmi rozsáhlými tvary atributy.

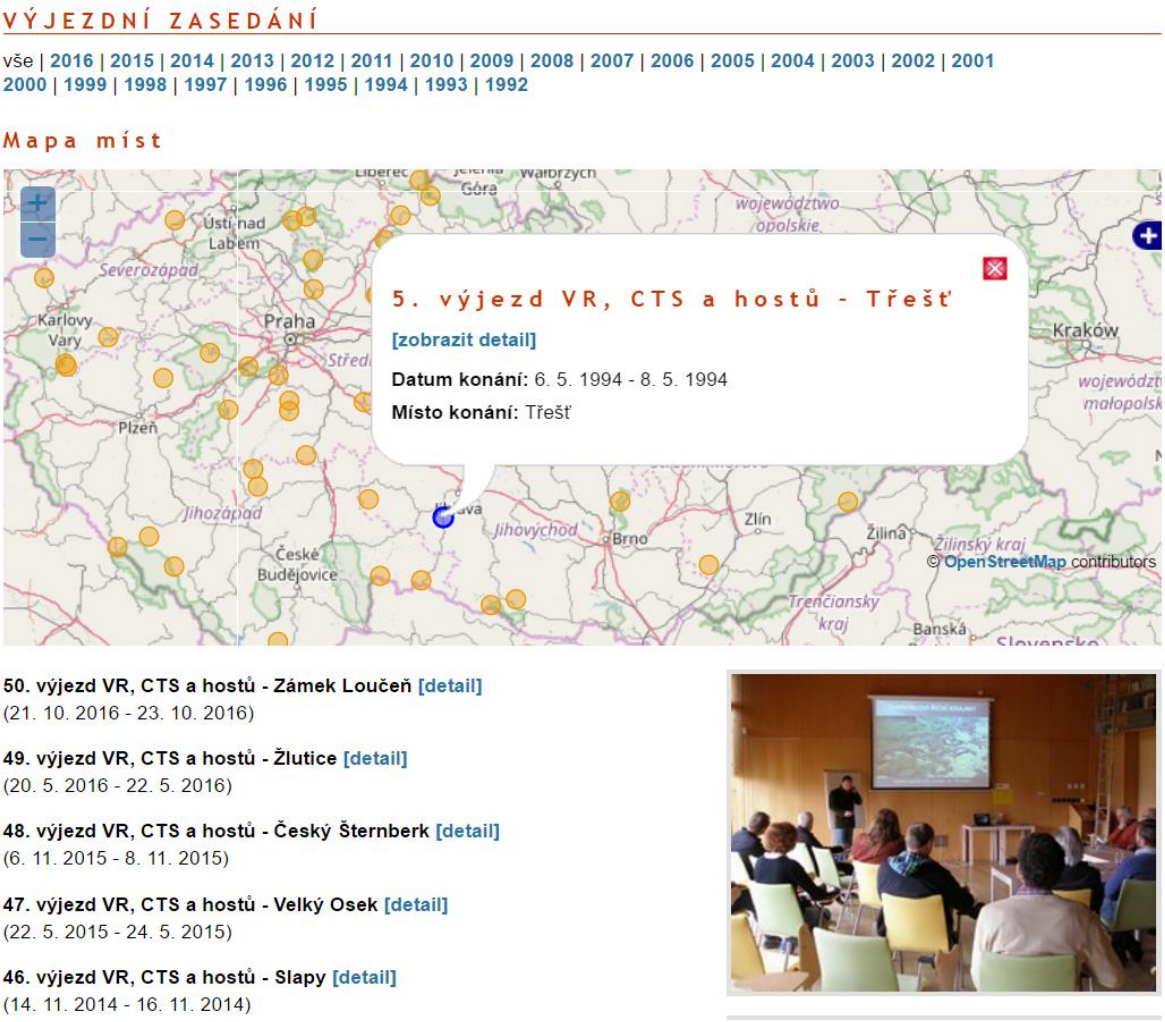
Tabulka č 17 - Velikosti souborů v závislosti na počtu geografických objektů.

Z tabulky lze vidět, že JSON je o trochu úspornější na přenesená data než formát KML. Proto jsem ve většině svých projektů a i při testování zde popsaného modelu jsem primárně používal formát GeoJSON. Formát KML má v praxi výhodu, že se do tohoto formátu častokrát lépe konvertuje nastavení symboliky objektů mezi různými programy (tj. styl jakým se objekty vektorové vrstvy zobrazují na mapě) a navíc je vysoce kompatibilní s jakoukoli aplikací od svého tvůrce (Google) včetně google maps.

Nicméně pro použití v prostředí webových služeb je často úspornější nastavit styl zobrazení znovu přímo v nastavení presentační vrstvy GIS (např. v Openlayers). Nastavení stylů integrované přímo v souborech dat navíc v závislosti na drobných rozdílech ve standardech formátu nefunguje, nebo způsobuje různorodé problémy.

Výše uvedená je analýza velikostí webů a potažmo souborů geografických dat, která se posílají k zobrazení uživateli. Z analýzy je zřejmé, že pokud je potřeba, aby aplikace zobrazovala najednou velké množství dat, tak to nelze bez specializovaného GIS serveru. GIS server, kromě toho, že dává přístup ke GIS analytickým a dalším funkcím umožňuje data publikovat jako služby (services) - například služby WMS a WFS a umožňuje rozdělit mapu do dílců (tiles). Tyto služby umožňují uživateli posílat pouze ta data, na která se uživatel zrovna dívá - tj. jsou vidět v mapovém okně. Pokud uživatel posune mapu na nové místo, tak se mu data automaticky načtou. Nicméně k tomu je třeba mít GIS server, který není součástí běžného webhostingu a nelze provádět filtrování a další funkce na straně uživatele.

Z osobní zkušenosti z rozhovorů co jsem vedl s uživateli i administrátory a i z výzkumu odborných článků o případových studiích (Olyazadeh et al 2016, Karnatak et al 2016, Dhamaniya et al 2016) vyplývá, že v naprosté většině případů se uživateli nezobrazuje zdaleka tolik objektů najednou. Běžné aplikace, se kterými jsem pracoval, obsahují do dvou tisíc objektů, takže se obešli bez GIS serveru, či ho potřebovali pouze kvůli specializovaným GIS funkcím. Naopak většina aplikací, se kterými jsem přišel do styku, geografická data pouze různým způsobem zobrazovala a neprováděla analýzu, a v tom případě GIS server není třeba. Naopak možnost přistupovat a manipulovat s daty přímo v MySQL či jiné databázi bez potřeby manipulovat s GIS serverem byla administrátory vítaná.



Obrázek č. 13 - Ukázka interaktivní mapy s overlay zobrazujícím detail zvoleného objektu - webová stránka výjezdních zasedání Centra pro teoretická studia (vlastní tvorba).

Další část výzkumu stávajících webových aplikací menší a střední velikosti, která je naprosto nutná pro tvorbu datového modelu je analýza dat z hlediska struktury jejich obsahu. Z provedeného výzkumu bylo zjištěno, že více než 80% zkoumaných webových aplikací zobrazuje uživateli v jeden čas primárně jednu vrstvu. Tato vrstva je většinou interaktivní a dovoluje uživateli zvolit objekt této vrstvy a zobrazit o něm informace, případně přejít na stránku objektu, kde jsou komplexní informace o objektu včetně jeho historie (viz obr. č 13). Další vrstvy jsou typicky neinteraktivní a pouze doplňují graficky zobrazení.

Pro velkou část webových aplikací (14 z 18 mnou přímo zkoumaných a i podpořeno rozhovory s experty) nehraje hlavní roli změna tvaru a změna polohy objektu, jako spíše změna atributů objektů. Tyto atributy pak ovlivňují styl zobrazení dané vrstvy. Neznamená to, že by zaznamenávání změny prostorové složky objektu nebylo důležité, ale typicky se s ní kvůli nedostatku výpočetních prostředků neprovádí žádná analýza, ale web GIS slouží jako presentace vrstev, které již jsou výsledkem analýzy provedené jinde. Případně geografická složka slouží pouze jako prostředek, jak lépe zobrazit atributová data uživateli. Toto také plyne z toho, jak je obtížné získávat složitější geografická data. To se provádí většinou pro specializovaný výzkum, který se zpracovává na klasických desktopových GIS a většinou se nepresentuje na webu, případně pokud ano, tak se presentují jen výsledky a ne historická data.

Toto tvrzení je podpořené i mým výzkumem existujících webových GIS aplikací (Konopásek a Klimešová 2016, Konopásek et al. 2014) a rozhovory s experty zabývajícími se tvorbou webových aplikací. Zdaleka nejtypičtější pro zkoumané aplikace byl provádět sběr dat a následně tato data vázat na předem připravené geografické objekty. Níže uvádím dva, tématem velmi odlišné, ale strukturou vlastně podobné, příklady z praxe.

- Webová GIS aplikace zemědělských polí a stavu půdy. Hlavní objekt sledování je zde zemědělské pole. Mapa zobrazuje barvu pole podle atributu, který obsahuje jejich naměřené znečištění a v detailu pole zobrazuje konkrétní jednotlivé naměřené hodnoty. Pole se samozřejmě mohou v čase měnit a mohou se i do systému zadávat nová, či naopak zanikat, ale hlavní změna v čase je u pravidelně měřených hodnot znečištění, které se do systému zadávají.

- Webová GIS aplikace kanceláří a jejich vybavení. Hlavní objekt sledování je zde kancelář a v ní umístěný majetek. GIS umožňuje zobrazit kanceláře podle různých

charakteristik, např. podle toho jestli obsahují projektor a PC či nikoli. Po zvolení kanceláře je možno vidět majetek, který je k ní připsán. Zde se kanceláře prakticky nikdy nebudou měnit tvarem, ale místnosti, které nejsou kanceláře, nebudou vedeny a status místnosti se může v čase změnit. Hlavní změnu v čase budou doznávat charakteristiky místnosti typu počet projektorů, počítačů, stolů atd. Typicky se bude systém aktualizovat při každé pravidelné inventuře.

Z provedeného výzkumu současných webových aplikací také vyšlo, že velká část aplikací provádí aktualizaci časových dat v dávkách, často pravidelně. Z aplikací, které jsem měl možnost zkoumat, 10 z 18 provádělo aktualizace v dávkách. Z rozhovorů pak ještě větší procento aplikací, zvláště aplikace z oblasti zemědělství. V příkladech výše šlo o pravidelné měření znečištění a pravidelnou inventuru majetku.

4.2 Návrh datového modelu

V této kapitole budou popsány výsledky výzkumu relevantní při návrhu datového modelu pro web GIS se sezónními (v čase se měnícími) daty. Následně je popsán model samotný a jeho výhody a nevýhody. Model byl navržen specificky pro potřeby webového prostředí, jak jsou stanovená v předchozí a i v této kapitole.

Jak již bylo nastíněno, momentální stav webových geografických informačních systémů s v čase se měnícími daty je, že data se uchovávají:

- s integrací času na úrovni relaci - to jest pro každý nový časový rámec je vytvořena nová vrstva, které odpovídá nový shapefile či tabulka v databázi,

  nebo s integrací času na úrovni řádku - to jest, jedná se o bitemporální tabulku, kde všechny časové rámce jsou v jedné tabulce či shapefile a pro každou změnu objektu v čase se vytváří nový řádek označený validním časem. Je to přístup založený na konceptuálním časovému modelu, jak je popsán v analýze současného stavu a časoprostorovému objektově orientovaném přístupu. Časoprostorový objekt je zde vyjádřen, jako set (tuple) složený z id objektu, geometrické složky objektu S v čase t, atributům objektu A v čase t a časové složky t – viz výraz 4.1. (Tang et al. 2010, Fan et al. 2010, Fan et al. 2011)

$$<oid, S_i(t), A_i(t), t> \quad (4.1)$$

Tyto dva datové modely se používaly ve všech stávajících webových GIS aplikací, které nepoužívají GIS server, a které jsem měl možnost zkoumat. Jedná se o 12 aplikací s různými typy zaměření a samozřejmě v čase se měnícími daty. Popularity těchto zobrazení, jak bylo blíže popsáno v dřívějších částech této práce, vychází to z toho, že pouze takto strukturovaná data umí presentační vrstva webového GIS přijmout. Navíc teprve v nedávné době se zvedla výpočetní síla webových serverů natolik, že mohou pracovat se složitějšími datovými modely a data následně konvertovat pro použití presentační vrstvy na straně uživatele. Tyto dva používané datové modely a způsoby uchovávání dat však přinášejí značné nedostatky.

- Oba způsoby mají velkou redundanci dat a tím pádem data zabírají daleko více místa a obtížněji se v nich data udržují.

- U těchto datové modelů nelze snadno použít dotazy na některé typy zobrazení dat, které lze považovat za důležité pro GIS aplikace ve webovém prostředí. Integrace času na úrovni relace, která je zdaleka nejpoužívanější, neumožňuje prakticky vůbec dotazování na změny objektu. Integrace času na úrovni řádku už dotazování na historii objektu sice umožňuje, ale častokrát jsou dotazy velmi složité a zpomalené nutností systému pro řízení báze dat procházet při zpracování dotazu redundantní data. Běžně používané modely také neumožňují složitější prostorové dotazování na změny tvaru objektů a lokaci objektů.

- Další nedostatek vychází z povahy administrace webových služeb. Z rozhovorů a výzkumu, který jsem prováděl, vyšlo, že pro web GIS většinou nejdůležitější složka dat, která se mění, není geografická složka, ale atributové charakteristiky objektům, podle kterých pak chceme uživateli podobu objektů presentovat. To má za následek, že atributová data objektů webové aplikace často chce měnit, zpracovávat a zadávat osoba, která s geografickými systémy neumí příliš dobře pracovat. Protože jsou v těchto datových modelech geografická data sloučena s atributovými, tak to je vlastně často technicky velmi obtížné.

V návaznosti na řešení těchto nedostatků jsem provedl analýzu datových modelů pro časoprostorová data, které se nepoužívají ve webovém prostředí, ale tyto nedostatky řeší. Hlavní přístupy jsou v základu shrnuté již výše v kapitole Analýza současného stavu. Vybrané modely a jejich řešení jsem se snažil integrovat do datového modelu s integrací času na úrovni řádku (tj. času jako atributu objektu), samozřejmě s ohledem, aby byl model i nadále

schopen fungovat v omezeném webovém prostředí. Model integrující čas na úrovni řádku jsem vybral jako základ pro vývoj a optimalizaci specializovaného datového modelu pro web GIS, protože se pro webový i klasický GIS přece jen občas používá a je alespoň částečně kompatibilní s presentační vrstvou webových GIS i relačními databázemi. Navíc má oproti mnohem více používané integraci času na úrovni relace, kde co vrstva to časový rámec, mnohem méně nedostatků.

Metoda integrace času na úrovni řádku byla již rámcově popsána v kapitole analýza současného stavu. Je to jedna z metod, které umožňují integrovat čas v relačním databázovém modelu, což je pro webové prostředí nezbytné. Tato metoda přidává čas k objektu jako normální atribut - charakteristiku objektu, přičemž při každé změně objektu se vytváří nový řádek, kde je zachycena nejnovější verze objektu a starý záznam se nemaže, ale přidá se konec intervalu validního času - tj. do kdy starý záznam platil. Řádky v tabulce podle tohoto modelu se dají popsat jako:

$$<\text{oid}, S_i(t), A_i(t), t> \qquad (4.2)$$

Kde oid je unikátní identifikátor objektu. V čase se nemění - tj. umožňuje podle něj najít aktuální a všechny historické verze objektu v systému. $S_i(t)$ jsou geografické souřadnice, respektive údaje o tvaru a poloze objektu, v čase t. $A_i(t)$ je set atributů patřící k objektu v čase t. A t je časová složka určující validní čas objektu (Tang et al. 2010).

Jeden z hlavních problémů tohoto datového modelu je redundance dat. I když je redundance daleko menší než u nejpoužívanějšího datového modelu s integrací času na úrovni relace, může být stále velmi významná. Zvláště pokud se typicky nemění všechny atributy najednou. Tento nedostatek řeší třetí z metod integrace času jako atributu - integrace času přímo na úrovni atributu. Minimalizuje se zde sice redundance, ale na úkor složitosti práce s daty. Navíc tato metoda není vhodná pro relační databáze ani pro webové prostředí. Její vhodné využití je nutné pracovat s databází, která umožňuje plnou podporu takzvaných nested relation schemas a práce s ní je výpočetně náročná. Můžeme však jít napůl cesty mezi těmito metodami a rozdělit atributy objektu do skupin podle četnosti jejich změn v čase a použití. Z analýzy charakteristiky dat používaných ve webových GIS aplikací s v čase se měnícími daty v předchozí kapitole jsem ve finále navrhl rozdělení objektu do 4 skupin.

Jak již bylo popsáno v analýze výše, historická data publikovaná ve webových GIS aplikacích jsou typicky zadávána do systému v dávkách a navíc častokrát pravidelně. Z toho při analýze

charakteristik dat vzešlo, že atributy objektu jde v naprosté většině případů logicky rozdělit do dvou skupin (Ott a Swiaczny 2001, Tang et al. 2010):

- Atributy, respektive data, která se mění relativně hodně, často i v pravidelných intervalech - pro potřeby mého návrhu jsem je pojmenoval jako data sezónní.

- Atributy, respektive data, která se nemění pravidelně. Často jsou v porovnání s ostatními atributy statická, ale mohou se občas změnit. Častokrát i mimo relativně pravidelné intervaly zadávání datových setů sezónních dat. Pro naprostou většinu webových GIS, jak již bylo v předchozí kapitole analyzováno, sem patří i určení tvaru a polohy objektu (shape).

Ve více jak 70% mnou zkoumaných aplikací se navíc nacházela potřeba uchovávat data, která se nemění, nebo stav jejich změny nepotřebujeme zkoumat. Navíc při rozhovorech s administrátory ostatních aplikací, bylo několikrát potvrzeno, že sice tam tyto atributy nemají, ale doplňují se jiným způsobem mimo databázi a chtělo by je tam vlastně mít. Častokrát se jednalo o dodatečné unikátní označení objektu (kromě klíče používaného databází) a textová poznámka. Pro většinu GIS aplikací vyplynulo ideální rozdělení - na část objektu, která se nemění v čase, část který se mění málo a nepravidelně a část která se mění často a pravidelně. V této poslední části se nachází data, na která je kladen důraz při prezentaci objektu uživateli. Při testování tohoto rozdělení atributů objektu u již existujících aplikací a při konzultování tohoto rozdělení s administrátory web GIS aplikací a dalšími experty v oboru se ukázalo, že se hodí na většinu případů a značně redukuje redundanci dat. Navíc toto rozdělení umožňuje na libovolnou skupinu atributů asociovat další třídu, která poskytuje dodatečné informace a je pak závislá na validním čase skupiny atributů ke které patří.

Poslední rozdělení objektu na 4 finální skupiny atributů má dvojí odůvodnění. Po testovém nasazení tohoto datového modelu na data z již existujících aplikací, bylo zjištěno, že ve druhé skupině - to jest v datech, která se mění s malou frekvencí a samostatně dochází k poměrně velké redundanci dat, a to obzvláště u atributu s tvarem objektu (shape). Ve zkoumaných aplikacích se tvar a umístění objektu často neměnil s takovou frekvencí, jako některé jiné atributy umístěné do této skupiny atributů. Navíc tato skupina vždy už z její definice obsahuje atributy, které se většinou mění samostatně, což způsobovalo častou redundanci atributu s tvarem objektu. Navíc je to velikostně obsáhlá informace, kde jedno její opakování je velikostně jako redundance 20 jiných atributů. Viz tabulka č. 18, kde redundance tvaru

objektu je 2x1000 znaků. Přitom se každý z atributů změnil jednou. Ostatní atributy díky svému charakteru nepřesáhli redundanci v rámci stovek znaků, ale 1000 znaků pro jeden polygon je poměrně běžné. Závisí samozřejmě na počtu znaků a zvoleném kartografickém zobrazení. Z toho vyplívá, že je vhodné rozdělit ještě tuto skupinu objektů na tvar objektu a ostatní atributy objektu.

ID	ID_objekt	Tvar objektu	Název	Vlastník	Čas validity od	Čas validity do
1	1	Shape 1 (1000 znaků)	Název 1	Jméno 1	2014-01-01	2014-05-05
2	1	Shape 1 (1000 znaků)	Název 1	Jméno 2	2014-05-05	2015-01-01
3	1	Shape 1 (1000 znaků)	Název 2	Jméno 2	2015-01-01	2015-03-01
4	1	Shape 2 (900 znaků)	Název 2	Jméno 2	2015-03-01	
...		...	...	...		

Tabulka č. 18 - příklad tabulky s daty s malou frekvencí změny a častou změnou pouze u jednoho atributu

Druhé odůvodnění proč toto udělat vychází z analýzy výzkumu současných webových aplikací již popsaných výše. A to, že menší a střední web GIS aplikace typicky slouží jako prezentace výsledků výzkumu či sběru dat provedeného předem v jiném programu. Zvláště geografická část - přidávání a změna nových tvarů objektů se musí provádět ve specializované GIS aplikaci jako je například ArcMap od společnosti ESRI. Typicky administrátor zběhlý v

práci s GIS si drží vrstvu objektů v takovémto programu offline a po změně je importuje do webového rozhraní. Mít geografická data v separátní tabulce tuto práci značně ulehčuje.

Pro potřeby modelu jsem nazval v návrhu datového modelu třídy, které vznikly z tohoto rozdělení objektu následovně:

- Objekt - statické atributy objektu. Tj. charakteristiky objektu, které se nemění, nebo nás nezajímá jejich vývoj v čase - jako příklad lze uvézt další unikátní identifikační označení objektu, textovou poznámku k objektu a další atributy specifické pro danou aplikaci.

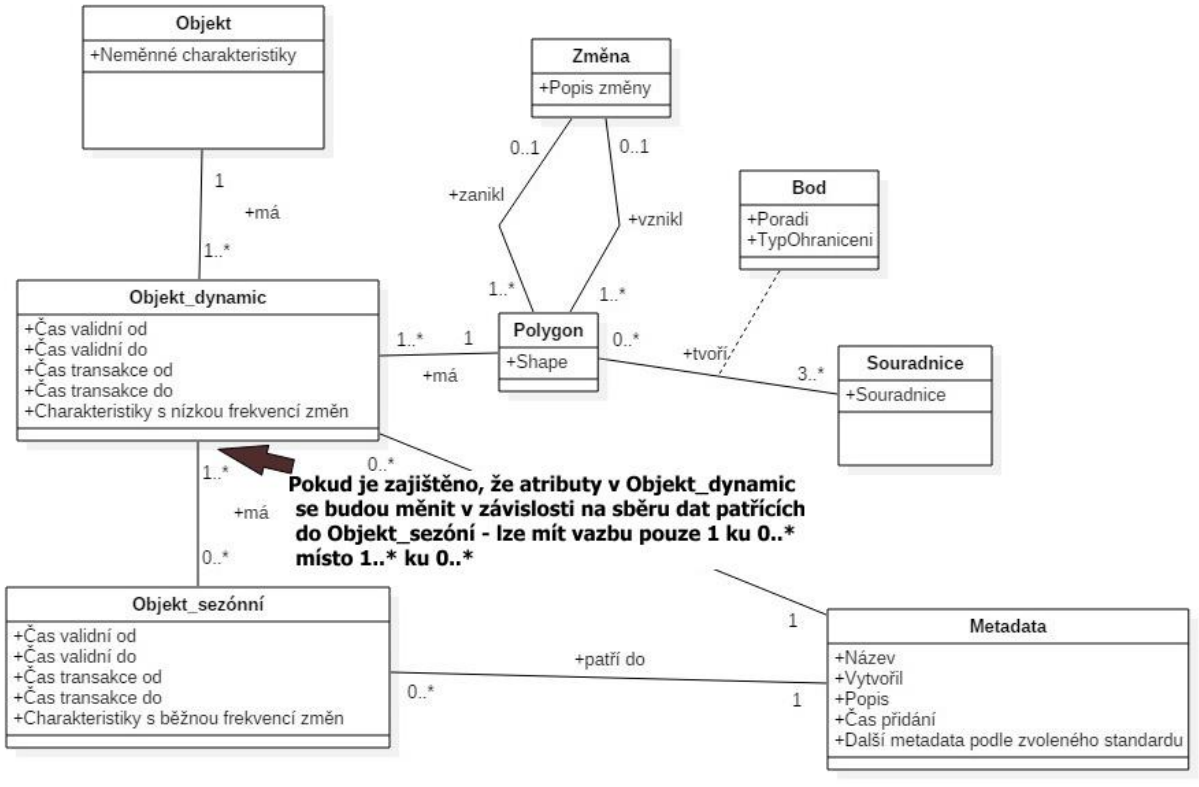
- Objekt_dynamic - atributy objektu s malou frekvencí změny. Tj. charakteristiky objektu, které se mění málo často a nepravidelně. Jako příklad lze uvézt typicky jméno objektu, vlastník objektu (může být atribut nebo asociace na další objekt) a další atributy specifické pro danou aplikaci. Vzhledem k tomu, že se jedná o dynamická data, nachází se zde typicky validní čas pro určení platnosti dat a čas transakce pro lepší administraci dat v případě chyby.

- Objekt_sezónní - atributy objektu s běžnou frekvencí změny. Tj. skupina charakteristik, které získáváme běžným sběrem dat a nepatří ani do jedné ze skupin popsaných výše. Vzhledem k tomu, že se jedná o dynamická data, nachází se zde typicky validní čas pro určení platnosti dat a čas transakce pro lepší administraci dat v případě chyby.

- Polygon - tvar a umístění objektu (shape). Tato třída obsahuje atribut s tvarem objektu ve formátu pro snadné použití při generování dat pro presentační vrstvu. Tedy typicky ve formátu tvaru používaném GeoJSON nebo KML. Pokud se očekává, že aplikace bude používat specializované prostorové funkce databáze, je vhodné mít pro snazší a rychlejší dotazování zde ještě prostorová data jednou, ve tvaru rozpoznávaném systémem pro řízení báze dat. Přestože se současné běžné systémy pro řízení báze dat nemohou rovnat GIS serveru, většina podporuje základní dotazy pro práci s prostorovými daty, jako je hrubý výběr podle lokace objektů (vybere data ve zvoleném rámci, který je tvořen souřadnicemi zadanými uživatelem).

Celý model je možno vidět na obrázku číslo 14 a rozdělení popsané výše je zvýrazněné na obrázku č. 15. Pro každou instanci třídy Objekt existuje v tabulce Objekt_dynamic 1..n verzí

objektu v čase, podle počtu změn jeho atributů. Pokud je zajištěno, že změny ne často se měnících atributů v Objekt_dynamic se zaznamenávají na základě datových souborů, které přinášejí ostatní běžně se měnící data. Tak může asociace mezi třídou Objekt_dynamic a Objekt_sezónní být 1 až 0..n. Může to být tato násobnost, protože nehrozí, že by se data změnila uprostřed sezóny - tj. že by sezóna měla být přiřazena ke dvěma instancím Objekt_dynamic. Násobnost na straně Objekt_sezónní je zde 0..N, protože se předpokládá, že mohou být objekty založeny dříve, než se do systému začnou sbírat a zadávat sezónní data. U web GIS je toto zvláště nutné, protože se typicky musí připravit tvary objektů, které často nejsou součástí normálního sběru dat. Pokud ale existuje možnost, že se budou charakteristiky objektu ve třídě Objekt_dynamic měnit i mimo sběr sezónních dat, tak je nutné použít mezi těmito třídami kardinalitu na 1..* ku 0..*.



Obrázek č. 14 - Návrh datového modelu pro web GIS s integrací času.

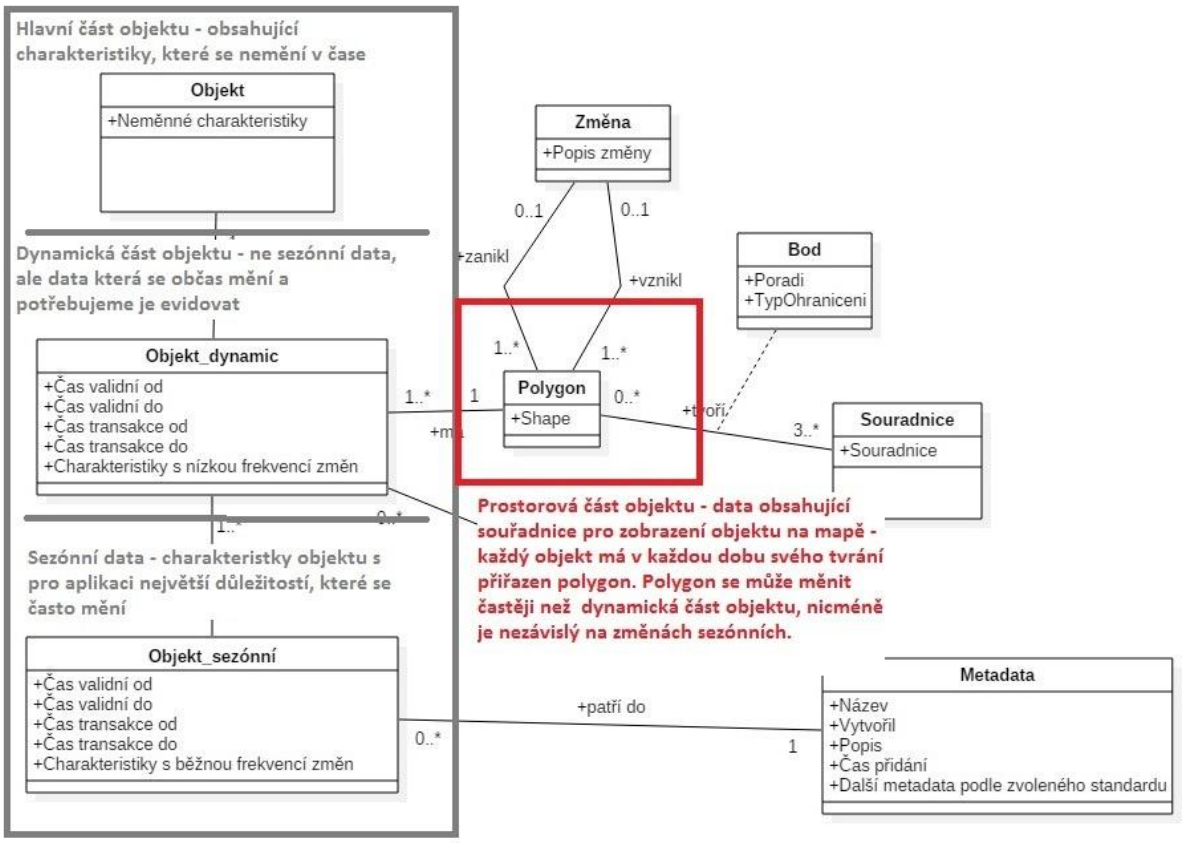
Třída Polygon obsahuje tvar objektu. Jak již bylo analyzováno výše, počítá se, že tvar objektu patří mezi atributy objektu, které se nemění často. Je proto asociována s třídou

Objekt_dynamic a to s násobností, že tvar může patřit k více instancím třídy Objekt_dynamic, ale jedna instance Objekt_dynamic může mít pouze jeden tvar. Redukuje se tím, jak bylo zamýšleno redundance atributu obsahující tvar objektu. Ten jak bylo již výše zmíněno, je často největší na počet znaků ze všech atributů objektu.

Tím, že třída Object_dynamic má vždy napojení na jeden Polygon, pokud chceme změnit tvar, musíme vytvořit novou verzi objektu ve třídě Objekt_dynamic. Toto sice přináší redundanci ostatních atributů ve třídě Objekt_dynamic oproti tomu, kdybychom asociovali třídu s tvarem objektu přímo na třídu Objekt. Ale přináší to také značnou výhodu při tvorbě dotazů, protože třída Objekt_dynamic obsahuje časová data, takže napojené instance třídy Polygon jsou rovnou časově zařazena podle validního času ve třídě Objekt_dynamic. Tím pádem při běžném dotazu na celkový stav objektu v určitém časovém bodě, není třeba porovnávat časové rozmezí zvlášť pro třídu Polygon a zvlášť Objekt_dynamic.

Tato podoba datového modelu s rozdělením objektu do čtyř tříd, umožňuje udržovat velmi malou redundanci dat a pohodlnou práci s aktualizací a managementem dat. Zároveň testování modelu v praxi ukázalo, že je model i přes nutnost použití několika spojení (join) při dotazech stále velmi použitelný i pro velké množství dat a splňuje nutné limity stanovené ve výzkumu potřeb webového prostředí a webových GIS aplikací. Navíc je tento model navržen speciálně pro malé a středně velké webové GIS aplikace a je použitelný s minimálními úpravami pro většinu z nich.

Model však stále pouze s těmito čtyřmi třídami neumožňuje snadné provádění všech dotazů, které byly v předchozím výzkumu a rozhovorech s administrátory a uživateli webových GIS jako potřebné. Konkrétně neumožňuje jednoduše získávat informace o změnách tvaru objektů a dotazy na sousedící objekty. Jak již bylo několikrát zmíněno, vlastní změna geometrie objektu často není to, na co je webový GIS primárně zaměřen. Většina současných GIS aplikací s dynamickými daty se zaměřuje na to zobrazovat v čase se měnící data objektů, které jsou umístěné v prostoru. S důrazem na tu první část a zobrazení v prostoru je použito často primárně jako nástroj pro lepší zobrazení dat uživateli. Data se mění často, ale tvar objektu málo. Ale o to více je nutné vědět více o charakteru změny tvaru, když nastane.



Obrázek č. 15 - Návrh datového modelu pro web GIS s integrací času - rozdělení atributů do skupin.

Jako příklad, na kterém je dobře vidět důležitost mít možnost zjistit a zobrazit charakter změny, lze uvézt již několikrát zmíněná web GIS aplikace se zemědělskými poli a průzkumem složení a kvality půdy. Aplikace představuje uživateli zemědělská pole a po zobrazení detailu pole, uživatel vidí vývoj jeho historie. Pokud nejsme schopni zjistit charakter změn tvaru, vidíme pouze, že za dobu co je pole v systému se mu změnil 3x tvar a zdvojnásobila se hodnota znečištění. Pokud jsme schopni zjistit charakter změn tvaru, vidíme navíc, že jedna ze změn tvaru, nebyla pouze drobná úprava hranic pole, ale sloučení s vedlejším polem. Následně si uživatel může zobrazit historii sloučeného pole a zjistit, jaké mělo před sloučením hodnoty znečištění a jestli sloučení mělo výrazný vliv na zvýšení znečištění pole, nebo zda stouplo znečištění nezávisle na změně tvaru. Presentace této možnosti často současné webové GIS aplikace nepodporují. Jak již však bylo zmíněno výše v kapitole "Analýza potřeb a specifik současných web GIS aplikací", většina uživatelů i administrátorů, se kterými jsem vedl nestrukturované rozhovory, projevili o možnost podpory takovýchto dotazů zájem.

Datové modely, které toto podporují, patří do kategorií modelů zabývajících se katastrálními daty a jsou to modely typicky navržené pro použití v mohutných aplikacích a pro velké množství dat. Existují datové modely, jak pro model integrace času na úrovni atributu (time as attribute), tak pro model, kde je atribut brán, jako plnohodnotná dimenze objektu jako jsou ESTDM a time-based integration method. Nejvhodnější možnost rozšíření mnou navrženého datového modelu se jevilo použít část z katastrálního datového modelu založeného na bázi integraci času na úrovni atributu, jelikož můj model je na něm také založen.

Tento databázový model ve své nejjednodušší verzi přidává k tabulce s objekty další dvě tabulky. Tabulku změn (event information table) a souhrnou tabulku změn (changing relationship table). Ve výsledku strukturuje model informace o objektu do těchto tří skupin dat:

Tabulka změn obsahuje informace o tom, jaký je charakter změny. Datový model, který adaptuji do svého návrhu, definuje záznam v této tabulce jako tuple složený z unikátního identifikátoru změny - EID, čas změny ET, typ změny EA a popis změny EI. Typicky se rozlišují tyto změny typů:

  - Rozdělení (division či segmentation)
  - Sloučení (combination, mergence)
  - Změna hranic (boundary adjustment)
  - Komplexní změna (complex change)
  - Změna atributu (attribute change)

- Tabulka objektů obsahuje atributy objektu včetně tvaru objektu. Přesně je záznam v této tabulce definovaný jako tuple složený z unikátního identifikátoru objektu ObjID, tvaru objektu ObjS, atributů objektu ObjA, identifikátoru změny, která tento tuple vytvořila OSE a případně identifikátor změny, která tento tuple označila jako historický.

- Souhrnná tabulka změn, ve které jsou explicitně zaznamenány změny mezi objekty. Umožňuje snadněji tvořit určité dotazy týkající se časových a polohových vztahů mezi objekty změny.

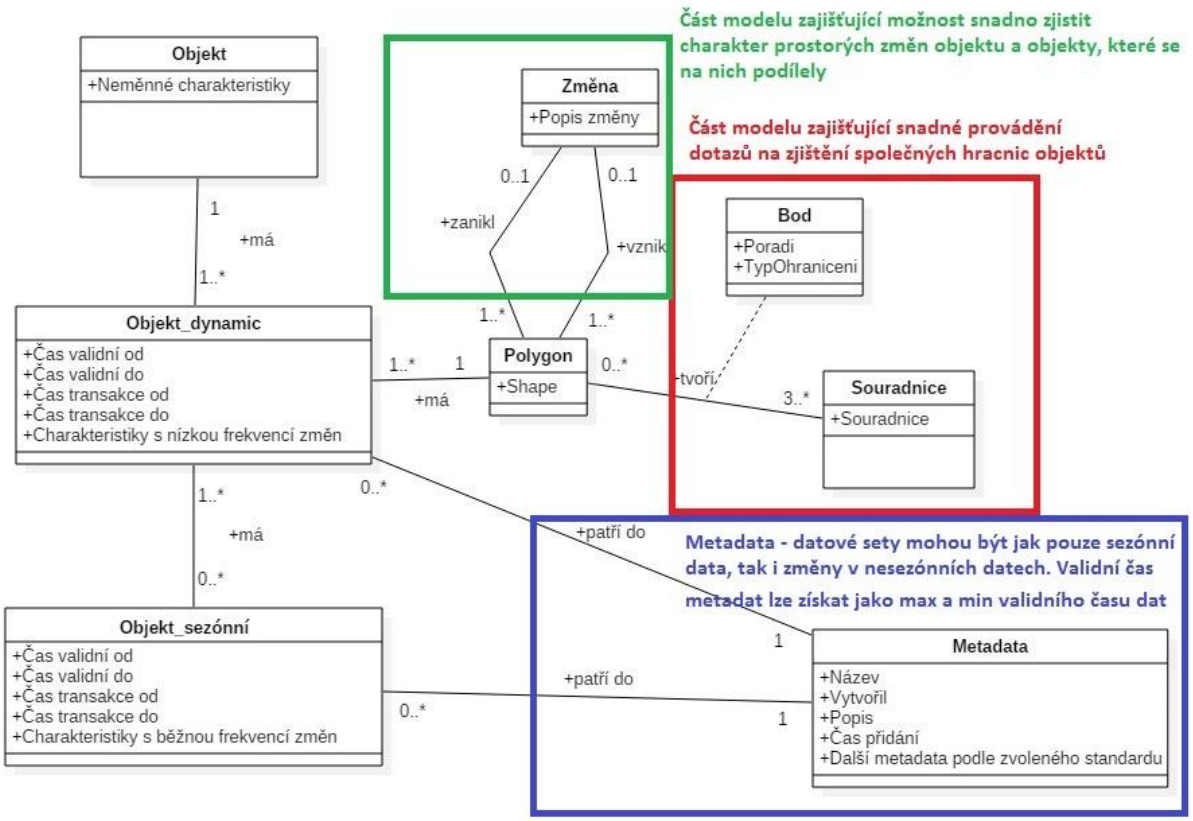
Obrázek č. 16 obsahuje souhrnný záznam tohoto modelu jak popsán Y. Fanem.

	Integration method regarding time as an attribute
Time	
Event	$\text{Event}(i) = f(\text{EID}, \text{ET}_i, \text{EA}_i, \text{EI}_i)$
Status quo of spatial entity	$\text{Object}(i) = f(\text{ObjID}, \text{Obj}S_i, \text{Obj}A_i, \text{OSE}_i)$
Historical spatial entity	$\text{Object}(i) = f(\text{ObjID}, \text{Obj}S_i, \text{Obj}A_i, \text{OSE}_i, \text{OEE}_i)$
Alternation relationship	$\text{Relation}(i) = f(\text{FID}, \text{CID}, \text{EID})$

Obrázek č. 16 - datový model s uchováváním charakteru změn tvaru objektů. Převzato z Fan et al. (2011)

Pro použití tohoto přístupu v specializovaném datovém modelu pro web GIS s integrací času šlo nakonec mnoho věcí zjednodušit. Souhrnnou tabulku změn není třeba do mnou navrhovaného datového modelu zahrnout, protože v prostředí webového GIS se neočekává, že by se prováděl typ dotazů na změny a polohu vícero objektů najednou, který se používá pro analýzy, ale typicky se bude zobrazovat uživateli pouze presentace detailu jednoho objektu. Zároveň také pro mnou navržený model není třeba v třídě Změna uchovávat časovou složku, ale ta je zajištěna napojením na třídu Polygon a následně Objekt_dynamic. Je to dané tím, že v mnou navrženém modelu je část prostorová oddělená od ostatních atributů a v tabulce změny se uchovávají záznamy pouze o změnách v prostoru a ne i v atributech, jako je tomu v modelu, ze kterého odvozuji.

Jako poslední věc, kterou jsem do mnou navrhovaného modelu zakomponoval, je možnost zjišťovat polohově sousední objekty k uživatelem zvolenému objektu. Je to rozšíření modelu, které jsem řešil u jedné z case study. U běžného webového GIS se s možností zobrazit sousední objekty málokdy setkáváme, ale pro určité specializované aplikace to může být užitečné. Pokud aplikace nepoužívá GIS server, tak typicky nemůžeme dotaz podle lokace jednoduše ve webovém prostředí použít. Pokud se však jedná o zjištění sousedních objektů, tj. objektů, které se zvoleného objektu přesně dotýkají, tak je to za určitých předpokladů možné. Hlavní předpoklad je pro veškeré objekty v systému, pokud spolu sousedí, mít na svých hranicích společné body, aby je bylo možno porovnat.



Obrázek č. 17 - Návrh datového modelu pro web GIS s integrací času - doplnění modelu o další části.

Pokud je tento předpoklad platný, tak se můžeme dotazem efektivně zeptat na objekty, které sdílí body se zkoumaným objektem. Pro použití v datovém modelu byl tvar objektu ze třídy Polygon rozdělen na jednotlivé souřadnice, které byly zaznamenány v připojené třídě. O souřadnicích pro každý polygon navíc musíme vědět, v jakém pořadí jsou při tvoření polygonu spojeny a jaký je to typ ohraničení. Typ ohraničení značí, zda je to vnější nebo vnitřní okruh polygonu (inner / outer ring). Tj. zajišťuje možnost mít polygony s dírou uprostřed nich.

Obrázek č. 17 má vyznačeny zbylé části modelu, doplněné o tabulku pro metadata, která je napojená jak na Objekt dynamic, tak Objekt sezónní.

4.3 Testování modelu

Pro testování navrženého datového modelu, bylo vytvořeno několik case study, kde byl model upraven a doplněn pro skutečné modelové případy (Konopásek a Klimešová 2017, Konopásek a Klimešová 2016). Tyto case study budou blíže zmíněny níže.

Model byl také testován přesně v podobě, jak je popsán v kapitole výše, aby se co nejpřesněji ověřili jeho limity a schopnost použití v praxi. Model byl naplněn jednoduchými testovacími daty a byly testovány běžné dotazy, identifikované z výzkumu popsaného v kapitole "Výzkum potřeb a specifik současných web GIS aplikací". Cílem testování bylo zjistit, zda model dokáže vracet presentační části web GIS aplikace data v rozumném časovém rozpětí (do 2 sekund). Protože však je toto velmi závislé na rychlosti internetového připojení uživatele, testování bylo omezeno na rychlost provedení dotazu databázovým systémem a následně rychlost jak rychle jednoduchým cyklem dokáže skriptovací jazyk z vrácených dat sestavit KML či GeoJSON soubor.

Testování bylo provedeno na serverech dvou českých webhostingů. Jako systém pro řízení báze dat (database management system) byla použita opensource databáze MySQL, která patří u našich hostingů k nejpoužívanějším. Jako skriptovací jazyk pro tvorbu KML a GeoJSON souborů a měření doby zpracování dotazů jsem použil PHP. Nepodařilo se mi najít přesné statistiky, ale při průzkumu českých hostingových společností jsem nenašel žádnou, která by kombinaci PHP a MySQL nenabízela. Samotný testovací skript jsem se snažil držet co v nejjednodušší podobě - při použití v praxi, jak je běžné v kombinaci s CMS systémem, lze očekávat většinou tak 100ms navíc pro zpracování ostatních funkcí webu. Měření času bylo provedeno funkcí microtime, který vrací současný Unixový čas (Unix timestamp) v mikrosekundách. Výsledný čas byl pak pro použití v grafech převeden na sekundy a zaokrouhlen na tři desetinná místa.

```
$time1 = microtime(true);
/* Provádění dotazu na databázi */
$time2 = microtime(true);
/* Sestavování KML či GeoJSON souboru */
$time3 = microtime(true);
echo '<table><tr>
<td>Čas zpracování dotazu: '.str_replace('.', ',',
round(($time2 - $time1), 3)).'</td>
```

```
<td>Čas vytvoření souboru: '.str_replace('.', ',',
round(($time3 - $time1), 3)).'</td>
</tr></table>';
```

Celý použitý testovací skript lze najít v příloze. Tabulky v databázi byly pro účely testování přejmenovány, neobsahují čas transakcí a tabulku pro metadata, protože pro účely testování nebyly podstatné. Byla doplněna asociační tabulka. Pro přesnou strukturu databáze viz obrázek č. 18. Pro zjednodušení nebyl generován validní čas jako datum, ale pouze jako číslo označující časovou jednotku.

Testovací data byla vygenerována pomocí skriptů PHP. Data byla náhodně vygenerována, přičemž jsem se snažil napodobit strukturu dat, kde systém běží zhruba 5 let což je dvojnásobek průměrné životnosti webové presentace. Generátor jsem se snažil nastavit, aby vyšlo, že se sezónní data sbírají každé dva měsíce, ale ne každý objekt existuje od začátku a ne každý objekt existuje až do současnosti. Přičemž nesezónní data se mění zhruba jednou za rok, ale ne pravidelně. Část skriptů na ukázku je ve zjednodušené podobě k dispozici v přílohách. Struktura dat pro testování je následující:

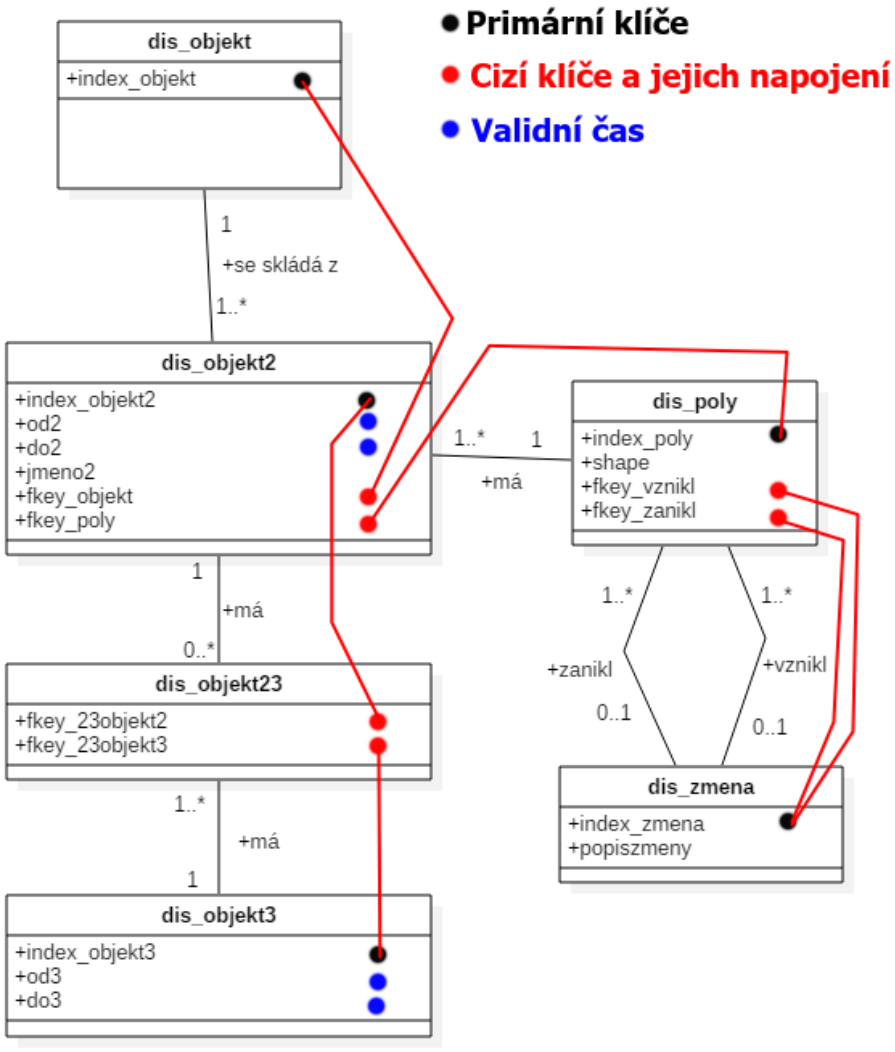
- Model byl testován na počet objektů v rozmezí 2000 až 5000 v krocích po 500 objektech. Jak již bylo popsáno výše, 5000 objektů odpovídá zhruba limitu počtu objektů, které lze poslat uživateli z hlediska velikosti přenesených dat. Navíc při více objektech aplikace většinou již dosahují velikosti a stylu zaměření práce s nimi, kde je třeba použít GIS server.

- Pro každý objekt bylo vygenerováno náhodně přidání do systému v čase t=rand(0,10).

- Pro každý objekt byl následně přidán záznam o 0 - 9 změnách, přičemž každá změna proběhla po uplynutí 1 - 5 časových jednotek. Pro každý objekt, tedy existovalo v teoreticky průměrně 5 změn. Data byla pro testování generována 3x a vyšlo v průměru 5,48 změn na objekt. V průměru měl objekt trvání 23,9 časových jednotek.

- Pro každou změnu byly vygenerovány sezónní data s frekvencí jedna sezóna na jednu časovou jednotku. Pokud nebyl u změny stanoven horní limit času validity, tak se vygenerovala data na 5 sezón. V průměru každý objekt byl v systému 23,9 sezón.

- Pro každý objekt byl vygenerován polygon a při každé změně, kterou objekt prošel, byla 50% šance, že se změní i tvar objektu. V praxi vyšlo, že každý objekt měl v průměru jednu změnu tvaru.



Obrázek č. 18 – Návrh databáze pro testování navrženého modelu

4.3.1 Výsledky testování

Testovány byly dotazy, běžně používané pro presentaci dat z datového modelu. Cílem dotazování bylo zjistit, jak se model chová při určitých objemech dat. Respektive jestli je schopen odezvy, která je přijatelná pro webové rozhraní a to při určitém naplnění daty. Metoda a prostředí testování jsou popsané výše. Všechna testování byla provedena desetkrát na běžném webhostingu společnosti Onebit, desetkrát na dvou různých webhostingových

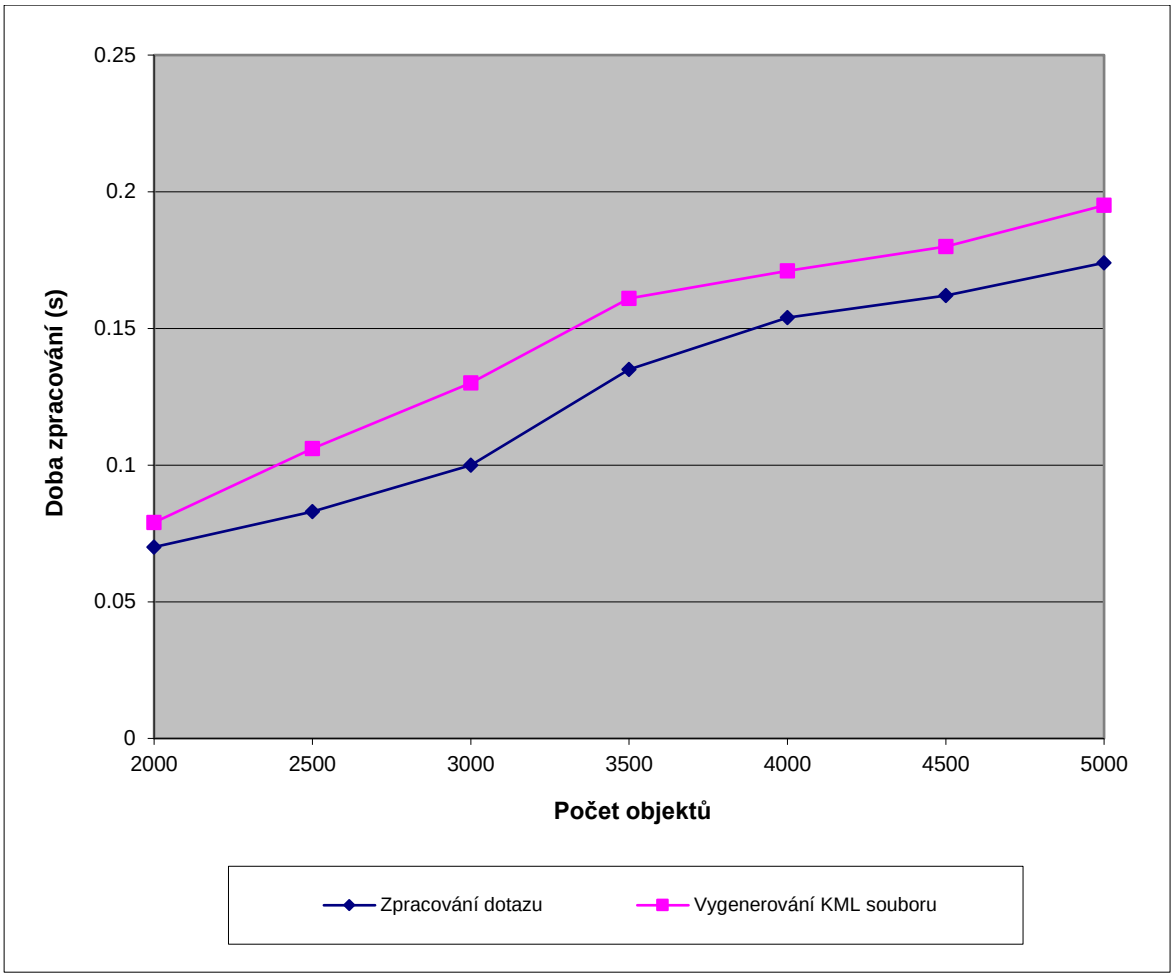
serverech společnosti Wedos. Výsledky měření jsou tedy vždy průměr z 30 testování. I přes to jsou tam v některých grafech patrné občasné drobné výkyvy způsobené pravděpodobně přetížením serveru, který dělí svoje výpočetní zdroje mezi řadu webhostingů a jejich uživatelů. U jednotlivých grafů jsou i ukázky dotazů na model v jazyku SQL. Počty řádků pro test vygenerovaných dat najdete v tabulce č. 19.

POČET OBJEKTŮ (ZÁZNAMY V DIS_OBJEKT)	POČET ZÁZNAMŮ V DIS_OBJEKT2	POČET ZÁZNAMŮ V DIS_OBJEKT3	POČET ZÁZNAMŮ V DIS_POLY
2000	11462	48492	4416
2500	14464	60350	5484
3000	18012	71526	6307
3500	21018	84055	7413
4000	24024	96188	8395
4500	27030	107913	9482
5000	30042	119547	10525

Tabulka č. 19 – počty dat vygenerovaných dat pro testování modelu

Na obrázku číslo 19 je graf rychlosti zpracování dotazu na vytvoření vrstvy pro zobrazení mapy v určitém – pravděpodobně nejpoužívanější dotaz webové aplikace. Dotaz má následující podobu:

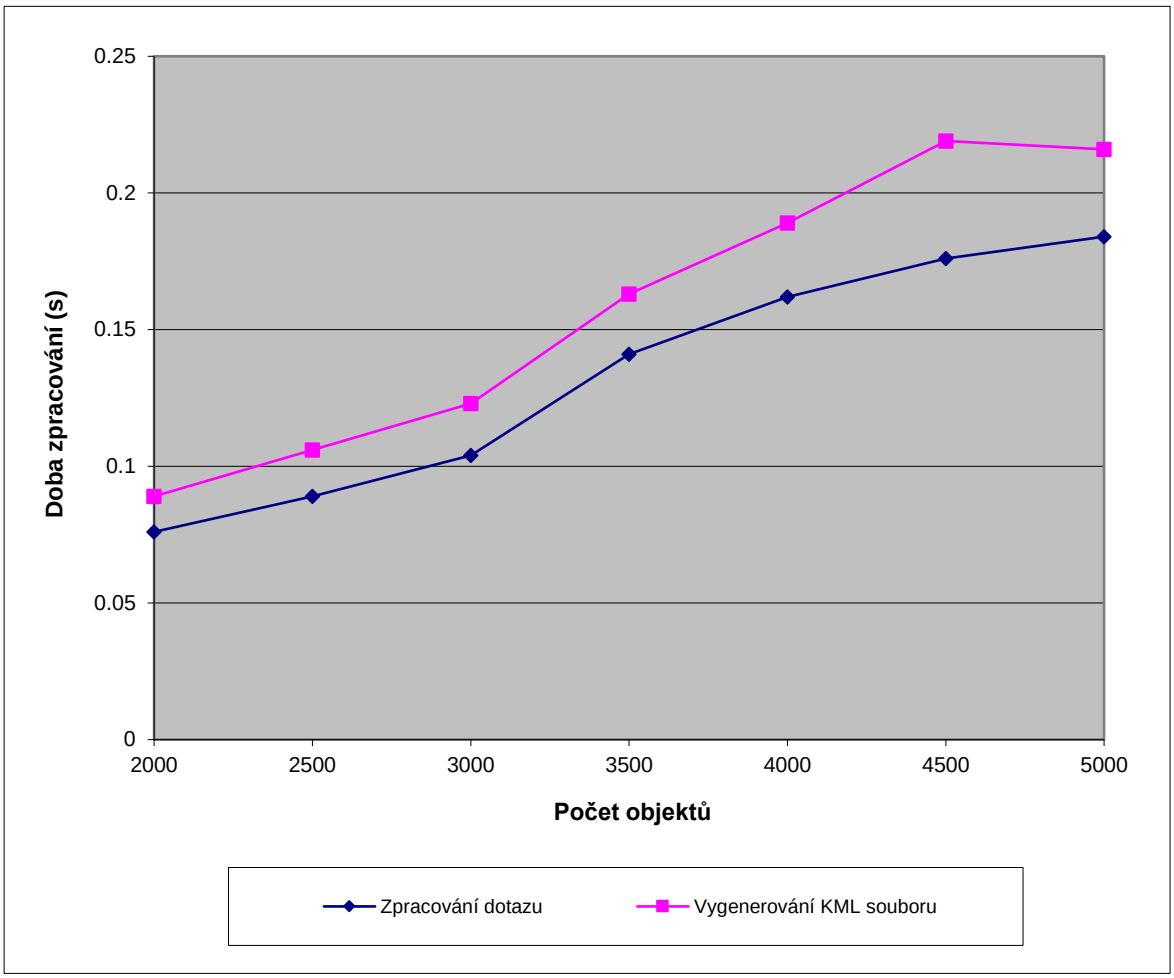
```
SELECT SQL_NO_CACHE *,NOW() FROM `dis_objekt`,
dis_objekt2, dis_objekt23, dis_objekt3, dis_poly LEFT
JOIN dis_zmena as vznikl on
vznikl.index_zmena=fkey_vznikl LEFT JOIN dis_zmena as
zanikl on zanikl.index_zmena=fkey_zanikl WHERE
index_objekt=fkey_objekt and
index_objekt2=fkey_23objekt2 and do2="0" and
fkey_23objekt3=index_objekt3 and fkey_poly=index_poly
```



Obrázek č. 19 – graf rychlosti dotazu na zobrazení aktuálních dat objektu.

Následující dotaz je také velmi často používaný. Vrátí stav existujících objektů v čase 10. Z tohoto dotazu je vidět, že pro model s 5000 objekty a 120 tisíci záznamy se sezónními daty objektů je možno vygenerovat stav objektů v libovolném okamžiku během 0,2 sekundy. To je naprosto dostačující doba pro použití ve webové aplikaci. I když připočteme čas na zpracování zbytku aplikace a doba jejího přenosu k uživateli doba zobrazení stránky bude ve většině případů menší než jedna sekunda. Pro uživatele s rychlým připojením to bude pravděpodobně kolem půl sekundy. Graf s rychlostí pro různá množství dat je vidět na obrázku číslo 20. Dotaz má následující podobu:

```
SELECT SQL_NO_CACHE *,NOW() FROM `dis_objekt`,
dis_objekt2, dis_objekt23, dis_objekt3, dis_poly LEFT
JOIN dis_zmena as vznikl on
vznikl.index_zmena=fkey_vznikl LEFT JOIN dis_zmena as
zanikl on zanikl.index_zmena=fkey_zanikl WHERE
index_objekt=fkey_objekt and
index_objekt2=fkey_23objekt2 and od2 <= 10 and (do2 > 10
or do2 = "") and od3 <= 10 and (do3 > 10 or do3 = "")
and fkey_23objekt3=index_objekt3 and
fkey_poly=index_poly ORDER BY `index_objekt` ASC
```

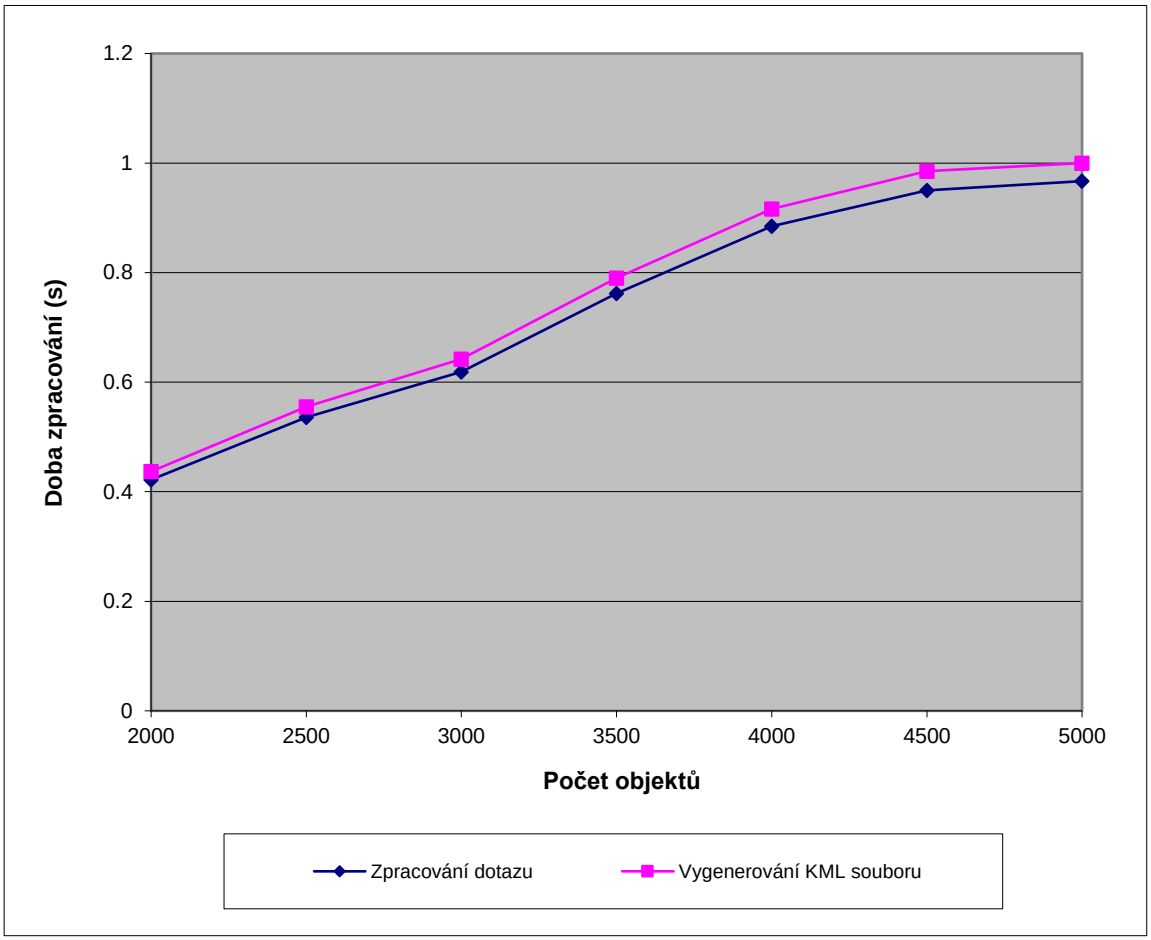


Obrázek č. 20 – graf rychlosti dotazu na zobrazení dat objektu v určitém čase.

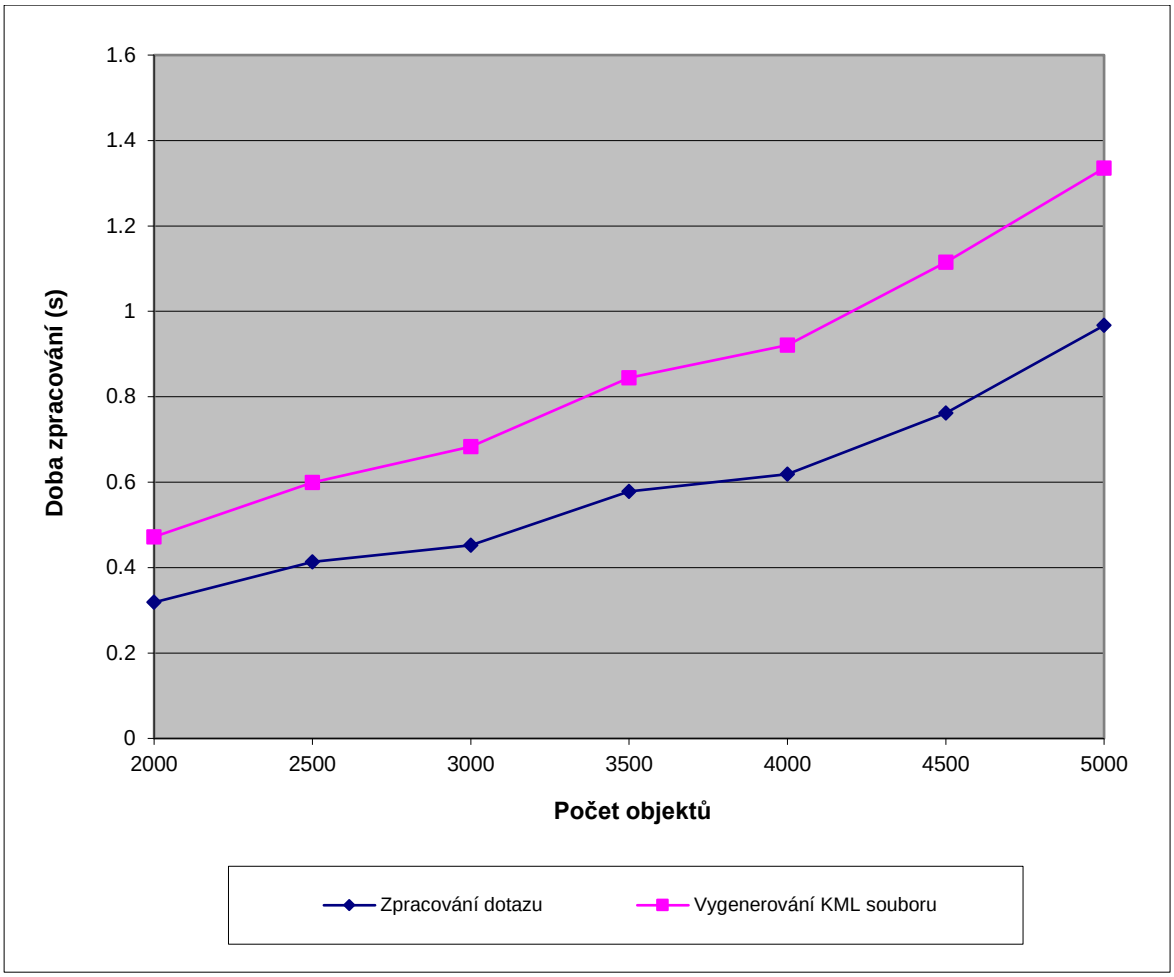
Další testovaný dotaz vybere všechny sezónní data od doby 5 do doby 15 a zjistí běžné statistiky za tuto dobu - celkovou sumu, průměr, minimum, maximum a počet. Z grafu na obrázku číslo 21 je vidět, že zpracování tohoto dotazu pro databázi s 5000 objekty je již velmi

dlouhé a zabere 1,35 sekundy. Při databázi s 5000 objekty vybraný dotaz provádí statistickou analýzu na 53 celkem tisících sezónních záznamů, které odpovídají zadanému rozmezí. Pro porovnání doba stejného dotazu, akorát bez klauzule „group by“, a tím pádem bez počítání statistik je na obrázku č. 22. Podoba dotazu je následující:

```
SELECT SQL_NO_CACHE *, sum(od3) as suma, avg(od3) as
avg, min(od3) as min, count(*) as count, NOW() FROM
`dis_objekt`, dis_objekt2, dis_objekt23, dis_objekt3,
dis_poly LEFT JOIN dis_zmena as vznikl on
vznikl.index_zmena=fkey_vznikl LEFT JOIN dis_zmena as
zanikl on zanikl.index_zmena=fkey_zanikl WHERE
index_objekt=fkey_objekt and
index_objekt2=fkey_23objekt2 and
fkey_23objekt3=index_objekt3 and fkey_poly=index_poly
and (do2>5 or do2=0) and od2 <= 15 and (do3>5 or do3=0)
and od3 <= 15 GROUP BY index_objekt ORDER BY
`index_objekt` ASC
```

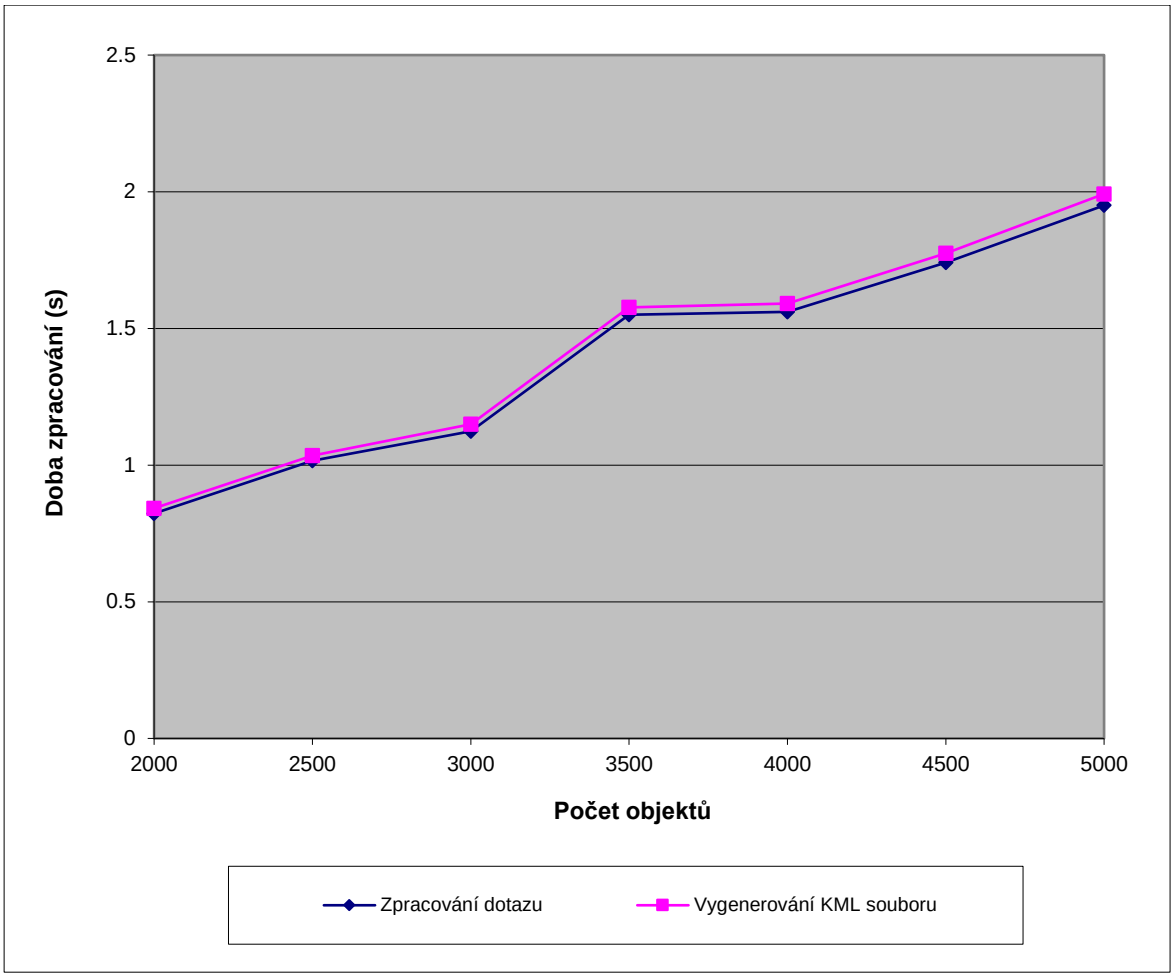


Obrázek č. 21 – graf rychlosti dotazu na zobrazení statistik sezónních dat v určitém intervalu.

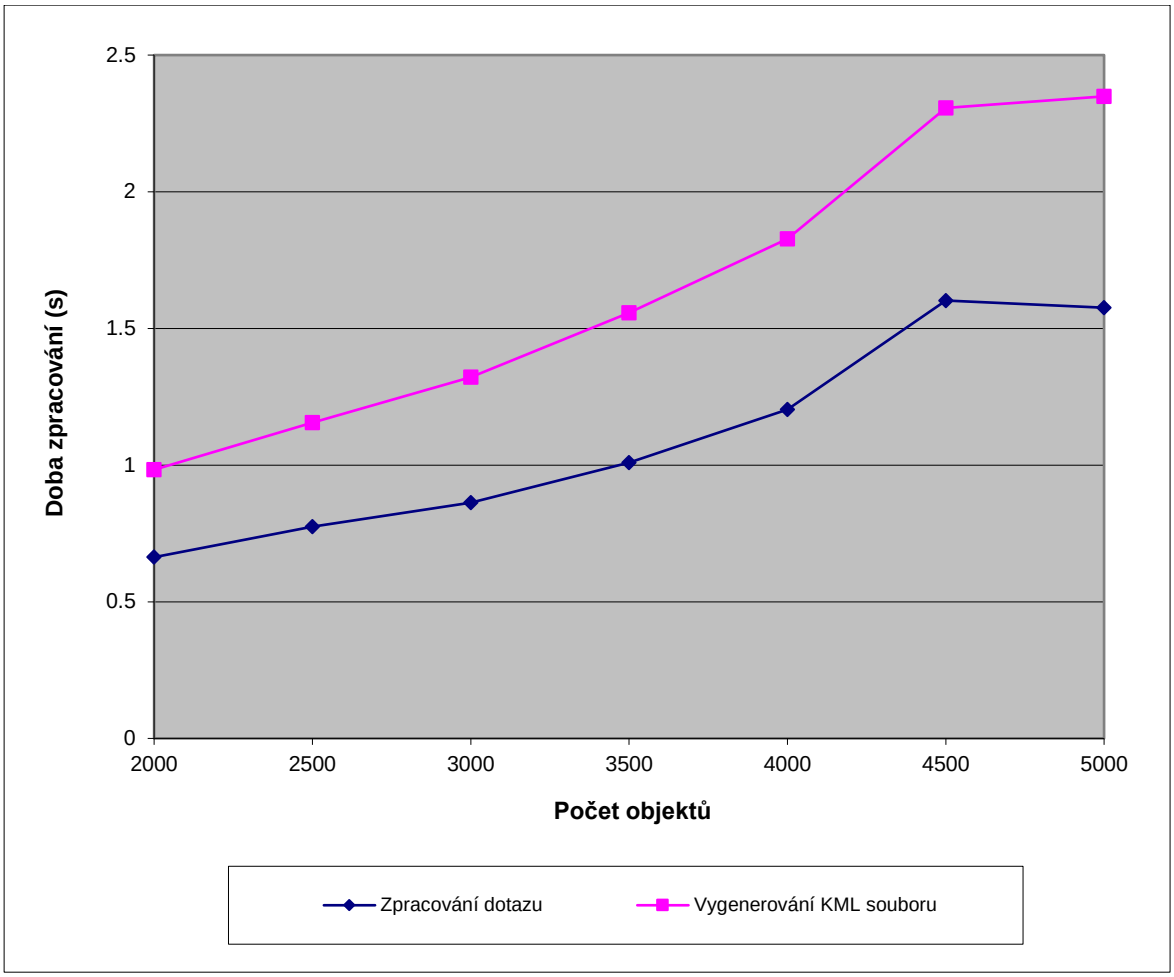


Obrázek č. 22 – graf rychlosti dotazu na vybrání sezónních dat v určitém intervalu.

Další popisovaný dotaz je podobný předchozímu, akorát obsahuje větší časový interval. Vybere všechny sezónní data od doby 5 do doby 40 a zjistí běžné statistiky ta tuto dobu - celkovou sumu, průměr, minimum, maximum a počet. Z grafu na obrázku číslo 23 je vidět, že zpracování tohoto dotazu pro databázi s 5000 objekty je již velmi dlouhé a zabere téměř 2 sekundy. Když se připočítá doba přenosu dat k uživateli a doba přenosu zbytku webové aplikace, se již blíží únosné hranici rychlosti odezvy. Tento dotaz však provádí statistiku s prakticky všemi sezónními daty (110 tisíc záznamů) a nepředpokládá se, že časový interval bude běžně zadáván v takovémto rozsahu. Pro porovnání doba odezvy stejného dotazu, akorát bez klauzule „group by“ a tím pádem bez počítání statistik je na obrázku č. 24.



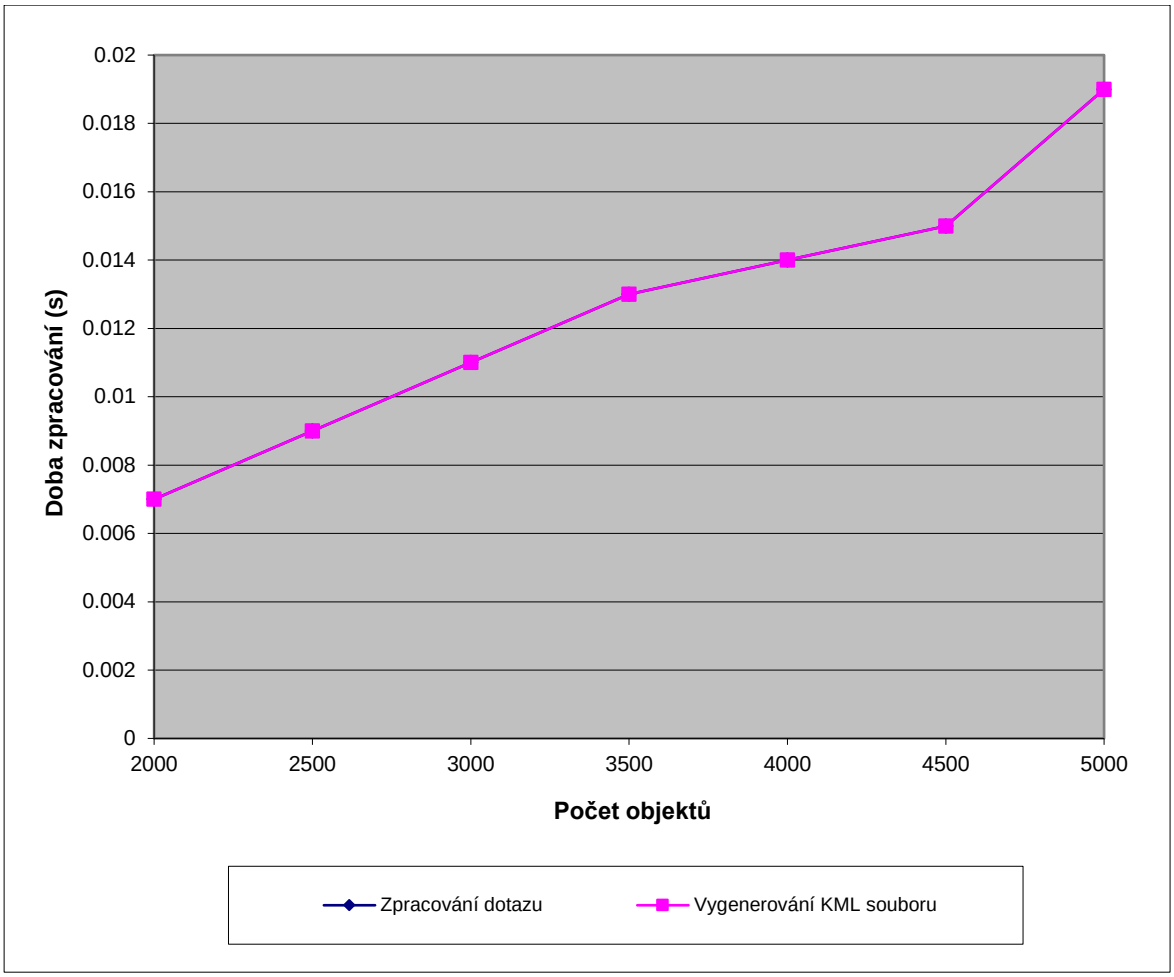
Obrázek č. 23 – graf rychlosti dotazu na vybrání sezónních dat v určitém intervalu – přes sto tisíc záznamů.



Obrázek č. 24 – graf rychlosti dotazu na vybrání sezónních dat v určitém intervalu – přes sto tisíc záznamů.

Poslední důležitý dotaz ukazuje možnost zjištění historického vývoje zvoleného objektu. Informace o objektu číslo jedna byly v systému shromažďovány po 11 sezón - je to zrovna objekt, který jich má vygenerováno méně. Jak je z grafu na obrázku číslo 25 vidět, tento dotaz může být proveden velice rychle. Dotaz je následující:

```
SELECT SQL_NO_CACHE *, NOW() FROM `dis_objekt`,
dis_objekt2, dis_objekt23, dis_objekt3, dis_poly LEFT
JOIN dis_zmena as vznikl on
vznikl.index_zmena=fkey_vznikl LEFT JOIN dis_zmena as
zanikl on zanikl.index_zmena=fkey_zanikl WHERE
index_objekt=fkey_objekt and
index_objekt2=fkey_23objekt2 and
fkey_23objekt3=index_objekt3 and fkey_poly=index_poly
and index_objekt=1 ORDER BY `index_objekt` ASC
```



Obrázek č. 25 – graf rychlosti dotazu na vybrání historického vývoje objektu.

Z uvedených grafů rychlosti je zřejmé, že navržený datový model i při největším testovaném objemu dat 5000 objektů a více jak 100 tisíc sezónních dat nepřekročil při žádném testovaném dotazu limit dvou sekund. Dá se tedy říci, že i přes svoji složitost model může být bez problému používán pro menší a střední webové aplikace s daty pohybujícími se v rámci deseti tisíců.

4.4 Porovnání modelu s existujícími alternativami

V této kapitole je shrnuto porovnání navrženého datového modelu s ostatními modely, které se používají v prostředí webových GIS. Jak již bylo popsáno v části analýzy webového prostředí, webové GIS aplikace používají typicky dva druhy datových modelů:

- Model s integrací času na úrovni relaci - kde pro každý nový časový rámec je vytvořena nová vrstva, které odpovídá nový shapefile či tabulka v databázi. Tento

přístup se používá v největší míře. Dovoluje snadnou integraci do webových aplikací, protože data se zobrazují, tak jak je uchováváme - co časový rámec to vrstva.

- Model s integrací času na úrovni řádku – kde jsou data uchovávána v typicky bitemporální tabulce. Všechny časové rámce jsou v jedné tabulce či shapefile a pro každou změnu objektu v čase se vytváří nový řádek s časem platnosti (validním časem). Podpora výchozího zobrazení tohoto modelu webovými aplikacemi je zatím ne příliš dobrá, Dá se však očekávat, že v budoucnosti poroste, protože na rozdíl od předchozího modelu umožňuje uživateli si zvolit zobrazit data v jakémkoli časovém bodu či intervalu. Z toho také vyplývá, že z tohoto modelu můžeme snadno ihned vygenerovat model předchozí a použít ho také se současnými GIS aplikacemi, ale mít výhodu po stránce snadnějšího managementu a menší redundanci dat.

Následuje tabulka číslo 20 s porovnáním mnou vytvořeného datového modelu s těmito datovými modely. Pokud je uvedeno více možností složitosti, tak je první uvedena možnost, která se očekává v běžné situaci. Například pokud je uvedena složitost malá až střední, tak většinou je malá, v některých případech střední.

	Datový Model S Integrací Času Na Úrovni Relace	Datový Model S Integrací Času Na Úrovni Řádku	Datový Model Pro Web Gis Se Sezónními Daty
Složitost datového modelu v databázi	Střední - pro každou vrstvu tabulka či soubor.	Malá - jedna tabulka.	Velká - pět a více tabulek.
Složitost dotazu na vytvoření vrstvy se současným stavem objektů	Žádná - vrstva již existuje.	Malá - dotaz na jednu tabulku.	Střední až velká - podle potřebných dat dotaz přes dvě až pět tabulek.

Složitost dotazu na historii daného objektu	Není možné provézt	Malá - dotaz na jednu tabulku	Střední až velká - podle potřebných dat dotaz přes dvě až pět tabulek.
Složitost dotazu na charakter změny tvaru objektu	Není možné provézt	Není možné provézt	Střední - podle potřebných dat dotaz přes tři až čtyři tabulky.
Složitost dotazu na sousední objekty zvoleného objektu	Není možné provézt	Není možné provézt	Střední - podle potřebných dat dotaz přes čtyři až šest tabulek.
Typická redundance dat	Velmi velká - pro každou vrstvu se všechna data o všech objektech mění spolu.	Velká až střední - záleží na tom, jak se data mění. Atributy a poloha objektu se vždy mění společně.	Střední až velmi malá - záleží na tom, jak se data mění. Atributy jsou rozdělené do tří skupin podle četnosti a charakteru jejich změn a poloha objektu je uvedena zvlášť.
Složitost managementu dat a případných rollback změn	Malá až velmi velká – malá pro změny v jednom čase a velmi velká pro změny naskrz	Střední – pracuje se v jedné tabulce a je třeba pracovat s validním a transakčním časem.	Střední až malá – informace o objektu jsou rozřazeny v několika tabulkách, ale díky způsobu jejich rozřazení se

	časovými rámci.		často pracuje pouze v rámci jedné. Geografická data jsou ve zvláštní tabulce, kterou lze snadno celou synchronizovat s offline programem ve kterém byly vytvořeny.
Složitost přidávání nových změn části atributových dat v čase	Střední - přidá se nová vrstva / tabulka. Ale podle charakteru dat je možné, že je třeba přiřadit nová atributová data k prostorovým datům a k datům, které se nezměnily.	Malá – Přidává se řádky do tabulky pro každou změnu. Ale podle charakteru dat je možné, že je třeba přiřadit nová atributová data k prostorovým datům a k atributovým datům, které se nezměnily	Velmi malá až malá – díky rozdělení atributů do skupin a oddělení geografických dat je možno přidat jen sezónní data či pouze evidovat změny v méně se měnících datech, aniž by se museli slučovat data z ostatních atributových skupin.
Složitost přidávání nových změn geografických dat v čase	Velmi malá – data jsou ve stejném formátu jako v klasickém GIS, je třeba jen připojit atributová data, což může být provedeno v GIS na stolním	Malá – Přidávají se řádky do tabulky pro každou změnu. Je třeba přiřadit nová prostorová data k atributovým datům, což může být provedeno v GIS na	Velmi malá až střední – díky rozdělení atributů do skupin a oddělení geografických dat je možno přidat jen geografická data. Následně je ale nutné

	počítači a stačí importovat do web GIS. Je třeba ale přidávat geografická data spolu s atributovými což může být velké množství dat.	stolním počítači. Je třeba ale přidávat geografická data spolu s atributovými což může být velké množství dat.	je připojit k datům atributovým v jiné tabulce, respektive vytvořit řádky v tabulce s málo se měnícími atributovými daty, kde se změní cizí klíč – napojení na tabulku geografických dat. Lze snadno provézt automatickým skriptem. Přidat informace o charakteru změn tvaru objektu je lehké, ale je třeba ho připravit předem. To samé s částí modelu týkající se snadného získávání sousedních objektů

Tabulka č. 20 – Porovnání s běžně používanými datovými modely

4.5 Publikační výstupy, case study a ověření modelu v praxi

Výsledky svého výzkumu jsem publikoval na řadě konferencí, jak v České republice, tak v zahraničí. Veškeré mé dosavadní publikační výstupy se týkají geografických informačních systémů a všechny, kde jsem hlavním autorem, se přímo týkají tématu této práce. Mé dosavadní publikace najde v tabulce č. 21.

Typ	Bibliografická citace
Jsc	KLIMEŠOVÁ, Dana, KONOPÁSEK, Jakub a OCELÍKOVÁ, Eva. Pyramidal Objects and Comparing Objects Using Similarity Measures. *International Journal of Mathematical Models and Methods in Applied Sciences.* 2014, (8): 138-145. ISSN 1998-0140. Dostupné také z: http://www.naun.org/main/NAUN/ijmmas/2014/a242001-304.pdf
Jsc	KLIMEŠOVÁ, Dana a KONOPÁSEK, Jakub. Image processing in cooperation with GIS. *International Journal of Mathematical Models and Methods in Applied Sciences.* 2013, 7(4): 388-395. ISSN 1998-0140.
D	KONOPÁSEK, Jakub, GOJDA Ondřej a KLIMEŠOVÁ, Dana. Using Temporal Database as a Source for Web GIS. *Recent Advances in Information Science.* 2013: 272-276. ISSN 1790-5109. Dostupné také z: http://www.wseas.us/e-library/conferences/2013/Paris/ECCS/ECCS-40.pdf
D	KLIMEŠOVÁ, Dana, KONOPÁSEK, Jakub a OCELÍKOVÁ, Eva. Measures of Objects Similarity. *Recent Advances in Information Science.* 2013: 180-184. ISSN 1790-5109. Dostupné také z: http://www.wseas.us/e-library/conferences/2013/Paris/ECCS/ECCS-26.pdf

Jrec	KONOPÁSEK, Jakub, GOJDA, Ondřej a KLIMEŠOVÁ, Dana. Spatiotemporal Data Model for Web GIS. *International Journal of Computers.* 2014, (8): 76-81. ISSN 1998-4308. Dostupné také z: http://www.naun.org/main/NAUN/computers/2014/a102007-145.pdf
Jsc	KONOPÁSEK, Jakub a KLIMEŠOVÁ, Dana. Data Model and Case Study of Seasonal Data in Web GIS. *International Journal of Applied Engineering and Research*. 2017, 12(8): 1691-1696. ISSN 0973-4562. Dostupné také z: https://www.ripublication.com/ijaer17/ijaerv12n8_30.pdf
D	KONOPÁSEK, Jakub a KLIMEŠOVÁ, Dana. Data model for working with seasonal data in webGIS. *AGRARIAN PERSPECTIVES XXV.*. 2016: 155-161. ISSN 2464-4781 (Online). Dostupné také z: http://ap.pef.czu.cz/static/proceedings/2016

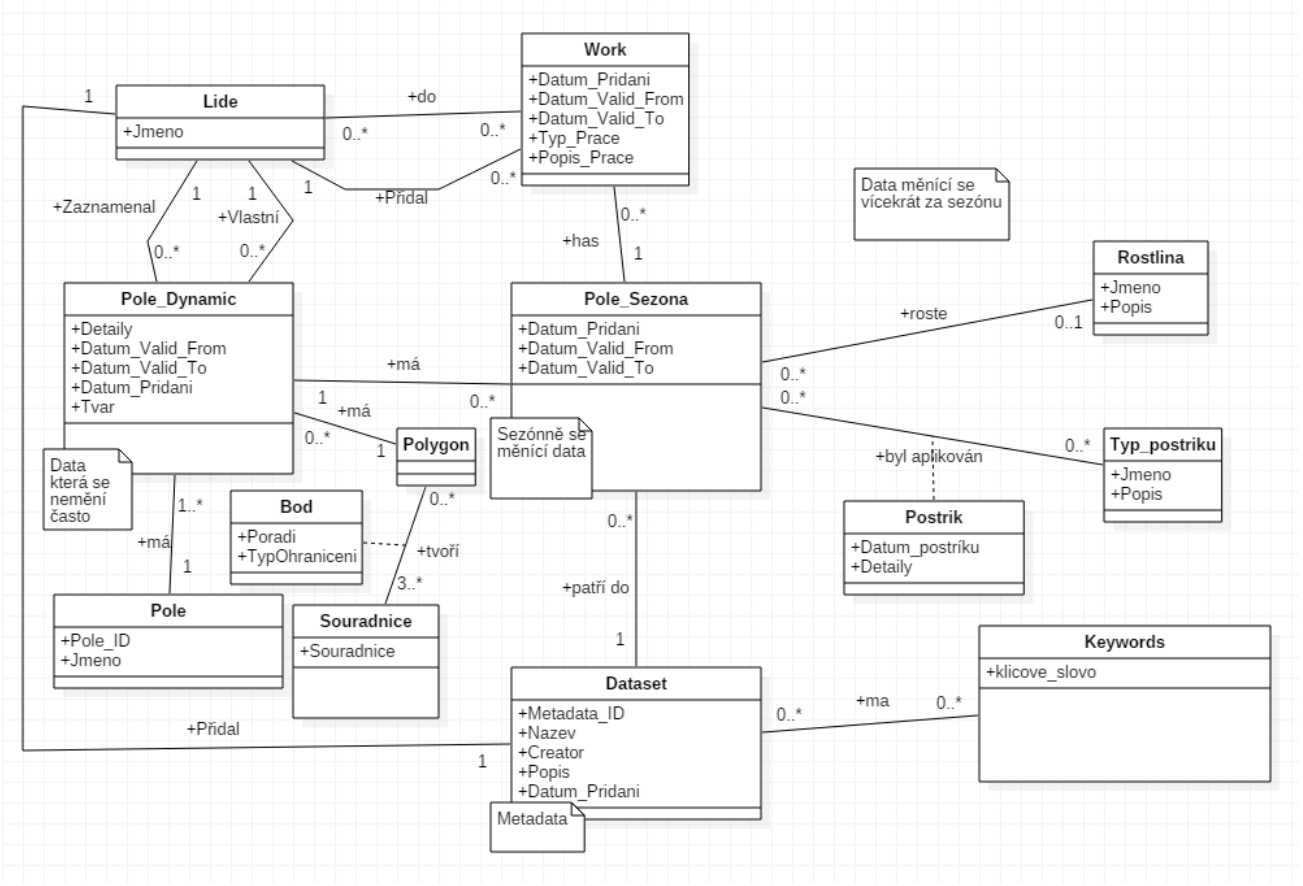
Tabulka č. 21 - seznam publikačních výstupů

Navržený datový model byl během svého vývoje testován na několika případových studiích, které pomohly zjistit jeho slabiny a tím podpořily jeho další rozvoj. V počátečních fázích vývoje se jednalo o case study vytvořené během dvouletého grantu IGA v letech 2012-14. Jednalo se o aplikaci obsahující plán budov České zemědělské university, respektive místností a věcí do nich přiřazených inventurou. Plán byl dále doplněn připojením zaměstnanců university k jednotlivým místnostem. Data se samozřejmě měnila v čase – označení místností a lidé v nich se měnily jednou za semestr a majetek v místnostech jednou za rok při inventuře. Výsledky výzkumu a této case study byly předneseny na konferenci NAUN WSEAS v Paříži (Konopásek et al 2013, Konopásek et al 2014). V rámci grantu byl vytvořen prototyp aplikace, ale byl nakonec naplněn pouze testovacími daty a aplikace se nikdy nedočkala plné realizace. Nicméně v tomto roce byly poznatky z této case study využity pro tvorbu reálné aplikace pro budovu Ministerstva Zemědělství, samozřejmě s již s poslední verzí navrženého datového modelu. Aplikace má být pro uživatele uvedena do provozu na začátku léta. Bohužel z bezpečnostních důvodů je aplikace k dispozici pouze interně pro zaměstnance a není možné uveřejnit její detaily. Na obrázku číslo 26 je alespoň pro ilustraci základní pohled na aplikaci

na aplikaci se zobrazeným detailem místnosti. Tato aplikace sice běžně neukazuje uživatelům historická data, ale je postavená na datovém modelu navrženém v této práci a data uchovává pro potřeby administrátorů pro další analýzu.

Obrázek č. 26 – Mapová aplikace plánu budovy ministerstva zemědělství s detailními informacemi o jednotlivých místnostech

Další case study obsahovala návrh aplikace zobrazující statistiky zemědělských polí, včetně stavu jejich půd a plodin co se na nich testují. Tato case study prošla několika předělávkami a sloužila jako hlavní case study pro testování uplatnění modelu v praxi, kde je navrhovaný datový model rozšířen o další třídy patřící specializovaně k dané aplikaci. Model a výsledky case study byly v první verzi presentovány na konferenci Agrarian Perspectives v roce 2016 (Konopásek a Klimešová 2016). Výsledky výzkumu a finální verze case study s modelem rozšířeným o další prvky jako práškování a evidenci prací na poli byl presentován v mezinárodním časopise International Journal of Applied Engineering and Research na jaře 2017 (Konopásek a Klimešová 2017). Jedna z výsledných podob modelu pro case study je vidět na obrázku číslo 27.



Obrázek č. 27 – Datový model pro case study zemědělských polí (Konopásek a Klimešová 2017)

5 Závěr

Cílem práce bylo navrhnout nový datový model, který by se hodil pro potřeby strukturování dat pro webovou GIS aplikaci pracující s časoprostorovými daty. Datový model pro web GIS se sezónními daty, jehož návrh je popsaný v této práci, plně vyhovuje předpokladům zjištěným z analýzy potřeb webového rozhraní, specifikům architektury webových GIS a je vhodný pro použití ve velkém množství webových GIS aplikací. Navržený datový model sice má větší složitost, než běžně používané datové modely, ale kompenzuje to mnohem snazší manipulací s daty. Navíc z testování modelu vyplynulo, že i když má větší složitost než běžně používané modely, je schopen v praxi vracet data s vyhovující odezvou i po jeho rozšíření pro použití v reálných aplikacích. Tím se dá cíl práce považovat za splněný.

Mnou navržený datový model vychází z analýzy současných datových modelů. Jako jeho základ byl použit datový model s integrací času na úrovni řádku. Model jsem upravil pro specifika zjištěná z analýzy současných webových aplikací. Následně jsem model rozšířil o struktury typické pro datový model katastrálních dat, aby umožňoval pokročilé dotazy na historii objektů. Jako poslední úprava modelu, bylo přidání doplňku, který umožňuje provádět ve zjednodušené podobě dotazy na sousedící objekty. Tím umožňuje řešit s rozumnou složitostí naprostou většinu dotazů, které webové GIS systémy potřebují pro presentaci dat uživateli. Zároveň svým strukturováním dat umožňuje jejich snadnou údržbu.

Součástí práce jsou výsledky testování použitelnosti modelu. Model byl realizován v praxi, naplněn testovacími daty a testoval jsem, zda je schopný provádět dotazy, které se od něho očekávají při různých úrovní naplnění daty a v časovém limitu přijatelném pro webové prostředí.

Výsledky výzkumu byly předneseny na řadě konferencí s dobrou odezvou, jak v České republice, tak v zahraničí. Datový model byl otestován v několika case study, které pak byly také popsány ve vědeckých publikacích. Datový model byl také úspěšně použit v praxi pro několik reálných projektů. Mnou navržený datový model je použitelný pro velké množství různorodých webových GIS aplikací. V budoucnu je možno ho nadále rozšiřovat, aby zahrnoval podporu dalších specifických dotazů, pro potřeby více specializovaných aplikací.

6 Seznam obrázků

7 Seznam tabulek

8 Seznam literatury a použitých zdrojů

ATAY, Canan Eren. A Comparison of Attribute and Tuple Time Stamped Bitemporal Relational Data Models. Proceedings of the International Conference on Applied Computer Science. Las Vegas, 2010.

BANDYOPHADYAY, Mainak, SINGH, Maharana Pratap a SINGH, Varun. Integrated visualization of distributed spatial databases An open source Web-GIS approach. In: 2012 1st International Conference on Recent Advances in Information Technology (RAIT) [online]. 2012 [cit. 2016-11-15]. DOI: 10.1109/rait.2012.6194600.

BANDYOPHADYAY, Mainak, SINGH, Maharana Pratap a SINGH, Varun. Integrated visualization of distributed spatial databases An open source Web-GIS approach. In: 2012 1st International Conference on Recent Advances in Information Technology (RAIT) [online]. 2012 [cit. 2015-07-21]. DOI: 10.1109/rait.2012.6194600.

COMBI, Carlo, KARAVNOU-PAPILIOU, Elpida a SHAHAR, Yuval. Temporal Databases. Temporal information systems in medicine. New York: Springer. 2010: 45-85. ISBN 1441965432.

COMBI, Carlo, KARAVNOU-PAPILIOU, Elpida a SHAHAR, Yuval. Temporal information systems in medicine. New York: Springer, c2010, xxviii, 376 p. ISBN 1441965432-.

DE SOUZA, Wagner Dias, LISBOA-FILHO, Jugurta, DE SOUSA CÂMARA, Jean Henrique, NUNES VIDAL FILHO, Jarbas a DE PAIVA OLIVEIRA, Alcione, 2014. ClickOnMap: A Framework to Develop Volunteered Geographic Information Systems with Dynamic Metadata. Computational Science and Its Applications – ICCSA 2014 [online]: 532-546 DOI: 10.1007/978-3-319-09129-7_39

DHAMANIYA, Ashish, Mathew SONU, M. KRISHNANUNNI, Pynam PRAVEEN a Austin JAIJIN. Development of Web Based Road Accident Data Management System in GIS Environment: a Case Study. Journal of the Indian Society of Remote Sensing [online]. 2016, 44(5), 789-796 [cit. 2017-06-07]. DOI: 10.1007/s12524-016-0547-8. ISSN 0255-660x. Dostupné z: http://link.springer.com/10.1007/s12524-016-0547-8

Dublin Core Metadata Element Set, Version 1.1. Dublin Core [online]. [cit. 2016-07-21]. Dostupné z: http://dublincore.org/documents/dces/

ELLUL, Claire, TAMASH, Nart, Feng Xian, STUIVER John a RICKLES, Patrick. Using Free and Open Source GIS to Automatically Create Standards-Based Metadata in Academia. OSGeo Journal. 2014, 13: 51-60.

EMHMED, Abubakar Agil a CHELLAPAN, Kalaivani. Modeling a Homogenate GIS Architecture Based On Service Oriented Architecture for the Tourism Mapping Needs. International Journal of Scientific and Engineering Research. 2012, 3(1). ISSN 2229-5518.

FAN, Yating., YANG, Jianyu, ZHU Dehai. a WE, K.L.. A time-based integration method of spatio-temporal data at spatial database level. Mathematical and Computer Modelling [online]. 2010, 51(11-12): 1286-1292 [cit. 2016-11-15]. DOI: 10.1016/j.mcm.2009.10.032.

FAN, Yating, Jianyu YANG, Dehai ZHU a Chao ZHANG. Research on the efficiency of querying historical data with the spatio-time data integration method. Mathematical and Computer Modelling [online]. 2011, 54(3-4): 912-918 [cit. 2016-11-15]. DOI: 10.1016/j.mcm.2010.11.015.

GeoJSON. GeoJSON [online]. [cit. 2015-07-21]. Dostupné z: http://geojson.org/

Geospatial Metadata. Federal Geographic Data Comittee [online]. [cit. 2015-07-21]. Dostupné z: http://www.fgdc.gov/metadata

GOODCHILD, Michael F., YUAN, May a COVA, Thomas J.. Towards a general theory of geographic representation in GIS. International Journal of Geographical Information Science [online]. 2007, 21(3): 239-260 [cit. 2017-03-01]. DOI: 10.1080/13658810600965271.

CHARVÁT, Karel. Geografická data v informační společnosti. Zdiby: Výzkumný ústav geodetický, topografický a kartografický, Odvětvové informační středisko, 2007, 269, [10] s. ISBN 9788085881288.

ISO19115. INTERNATIONAL STANDARD ISO 19115: Geographic information — Metadata. 2013.

KARNATAK, Harish Chandra, SARAN, Sameer, BHATIA, Karamjit a ROY, P. S.. Multicriteria Spatial Decision Analysis in Web GIS Environment. GeoInformatica [online].

2007-9-24, 11(4), 407-429 [cit. 2017-06-07]. DOI: 10.1007/s10707-006-0014-8. ISSN 1384-6175. Dostupné z: http://link.springer.com/10.1007/s10707-006-0014-8

Keyhole Markup Language. KML [online]. [cit. 2016-09-11]. Dostupné z: https://developers.google.com/kml/

KHATRI, Vijay, RAM, Sudha a SNODGRASS, Richard T.. On augmenting database design-support environments to capture the geo-spatio-temporal data semantics. Information Systems [online]. 2006, 31(2): 98-133 [cit. 2016-11-15]. DOI: 10.1016/j.is.2004.10.001.

KLIMEŠOVÁ, Dana a KONOPÁSEK, Jakub. Image processing in cooperation with GIS. *International Journal of Mathematical Models and Methods in Applied Sciences.* 2013, 7(4): 388-395. ISSN 1998-0140.

KLIMEŠOVÁ, Dana, KONOPÁSEK, Jakub a OCELÍKOVÁ, Eva. Pyramidal Objects and Comparing Objects Using Similarity Measures. *International Journal of Mathematical Models and Methods in Applied Sciences.* 2014, (8): 138-145. ISSN 1998-0140. Dostupné také z: http://www.naun.org/main/NAUN/ijmmas/2014/a242001-304.pdf

KOMÁRKOVÁ, Jitka, JAKOUBEK, Kamil a HUB, Miloslav. Usability evaluation of web-based GIS. In: Proceedings of the 11th International Conference on Information Integration and Web-based Applications & Services - iiWAS '09 [online]. 2009 [cit. 2016-11-20]. DOI: 10.1145/1806338.1806443.

KOMÁRKOVÁ, Jitka, SEDLÁK, Pavel, NOVÁK, Martin, MUSILOVÁ, Alena a SLAVÍKOVÁ, Veronika. Problems in Usability of Web-Based GIS. Proceedings of the International Conference on Applied Computer Science (ACS). Atény: WSEAS Press, 2010: 419-425. ISBN 978-960-474-225-7.

KOMÁRKOVÁ, Jitka, SEDLÁK, Pavel, NOVÁK, Martin, SLAVÍKOVÁ, Veroníka a MUSILOVÁ, Alena. Problems in Usability of Web-Based GIS. Proceedings of the International Conference on Applied Computer Science (ACS). Athens, WSEAS Press, 2010.

KONOPÁSEK, Jakub a KLIMEŠOVÁ, Dana. Data Model and Case Study of Seasonal Data in Web GIS. International Journal of Applied Engineering and Research. 2017, 12(8): 1691-1696. ISSN 0973-4562. Dostupné také z: https://www.ripublication.com/ijaer17/ijaerv12n8_30.pdf

KONOPÁSEK, Jakub a KLIMEŠOVÁ, Dana. Data model for working with seasonal data in webGIS. AGRARIAN PERSPECTIVES XXV.. 2016: 155-161. ISSN 2464-4781 (Online). Dostupné také z: http://ap.pef.czu.cz/static/proceedings/2016

KONOPÁSEK, Jakub, GOJDA Ondřej a KLIMEŠOVÁ, Dana. Using Temporal Database as a Source for Web GIS. *Recent Advances in Information Science.* 2013: 272-276. ISSN 1790-5109. Dostupné také z: http://www.wseas.us/e-library/conferences/2013/Paris/ECCS/ECCS-40.pdf

KONOPÁSEK, Jakub, GOJDA, Ondřej a KLIMEŠOVÁ, Dana. Spatiotemporal Data Model for Web GIS. International Journal of Computers. 2014, (8): 76-81. ISSN 1998-4308. Dostupné také z: http://www.naun.org/main/NAUN/computers/2014/a102007-145.pdf

KOPÁČKOVÁ, Hana, JONÁŠOVÁ, Hana, MIKEŠOVÁ, Iva a HEJLOVÁ, Jana. Gathering of Requirements on Web GIS Development - the Example of Bikeway Mapping Application. Recent Researches in Circuits, Systems, Communications and Computers: Proceedings of the 2nd European Conference of Computer Science (ECCS'11). Stevens Point, WSEAS Press, 2011: 290-295.

LUACES, Miguel a RODRÍGUEZ, Ángel. A Generic Architecture for Geographic Information Systems. Coruña Spain, 2004. Dostupné také z: http://lbd.udc.es/Repository/Thesis/432677332J.pdf. Doctoral thesis. Universidade da Coruña.

MICHAELIS, Christopher D. a AMES, Daniel P.. Considerations for Implementing OGC WMS and WFS Specifications in a Desktop GIS. Journal of Geographic Information System [online]. 2012, 04(02): 161-167 [cit. 2017-04-21]. DOI: 10.4236/jgis.2012.42021.

NAŘÍZENÍ KOMISE (ES) č. 1205/2008 ze dne 3. prosince 2008, kterým se provádí směrnice Evropského parlamentu a Rady 2007/2/ES týkající se metadat. Ú řední věstník Evropské unie. 2008. Dostupné také z: http://inspire.gov.cz/sites/default/files/documents/MetadataCZ.pdf

OLYAZADEH, Roya, AYE, Zar Chi, JABOYEDOFF Michel a DERRON, Marc-Henri. Prototype of an open-source web-GIS platform for rapid disaster impact assessment. Spatial Information Research [online]. 2016, 24(3), 203-210 [cit. 2017-06-07]. DOI: 10.1007/s41324-016-0017-y. ISSN 2366-3286. Dostupné z: http://link.springer.com/10.1007/s41324-016-0017-y

Openlayers 3. Openlayers [online]. [cit. 2017-07-21]. Dostupné z: http://openlayers.org/

OTT, Thomas a SWIACZNY Frank. Time-integrative geographic information systems: management and analysis of spatio-temporal data. Berlin: Springer-Verlag, 2001, xiii, 234 s. ISBN 3540410163.

PULTAR, Edward, RAUBAL, Martin, COVA, Thomas J a GOODCHILD, Michael F. Dynamic GIS Case Studies: Wildfire Evacuation and Volunteered Geographic Information. Transactions in GIS. 2009, 13: 85-104. DOI: 10.1111/j.1467-9671.2009.01157.x. ISSN 13611682. Dostupné také z: http://doi.wiley.com/10.1111/j.1467-9671.2009.01157.x

PULTAR, Edward, COVA, Thomas, YUAN, May a GOODCHILD, Michael. EDGIS: a dynamic GIS based on space time points. International Journal of Geographical Information Science. 2010, 24(3): 329-346. DOI: 10.1080/13658810802644567. ISSN 1365-8816.

REVESZ, Peter. Introduction to databases: From Biological to Spatio-Temporal. London: Springer. 2010: 67-79. ISBN 1849960941.

SARUP, Jyoti a SHUKLA, Vimal. Web-Based solution for Mapping Application using Open-Source Software Server. International Journal of Informatics and Communication Technology (IJ-ICT). 2012, vol. 1, no. 2. DOI:10.11591/ij-ict.v1i2.1490.

SHI, Wenzhong, KWAN, Kawai, SHEA, Geoffrey a CAO, Jiannong. A dynamic data model for mobile GIS. Computers & Geosciences [online]. 2009, 35(11): 2210-2221 [cit. 2015-07-21]. DOI: 10.1016/j.cageo.2009.03.002.

SINGH, Sunil Pratap a SINGH, Preetvanti. Mapping Spatial Data on the Web Using Free and Open-Source Tools: A Prototype Implementation. Journal of Geographic Information System [online]. 2014, 06(01): 30-39 [cit. 2015-07-21]. DOI: 10.4236/jgis.2014.61004.

Soasta web optimalizations and tests. [online]. [cit. 2017-02-11]. Dostupné z: https://www.soasta.com/blog/page-bloat-average-web-page-2-mb/

TANG, Yong, YE, Xiaoping a TANG, Na. Temporal information processing technology and its applications. New York: Springer, 2010, xviii: 349. ISBN 3642149588.

Timemap javascript library. Timemap [online]. [cit. 2016-11-51]. Dostupné z: http://code.google.com/p/timemap/

YE, Xiaoping, PENG, Zewu a GUO Huan. Spatio-Temporal Data Model and Spatio-Temporal Databases. Temporal Information Processing Technology and Its Application. Berlin, Springer Berlin Heidelberg. 2010: s. 91-112.

9 Příloha – generování testovacích dat

Generování statické části objektů

```
$sqlx  =  $GLOBALS["C_MySQL"]->query("SELECT * FROM
dis_objekt ORDER BY index_objekt desc LIMIT 1");
$sqlxres  =  $GLOBALS["C_MySQL"]->next_row($sqlx);
$id = $sqlxres["index_objekt"]+1;



if (!$_GET["i"])$_GET["i"]=10;
$_GET["i"]= $_GET["i"]+$id;



for ($i = $id; $i <= $_GET["i"]; $i++) {
     $qry='INSERT INTO `dis_objekt` (`index_objekt`)
VALUES ("'.$i.'");';
     if ($_GET["auto"]){ $q =  $GLOBALS["C_MySQL"]-
>query($qry); }
     $out.='<div>'.$qry.'</div>';
     $id++;
}
echo $out;
```

Generování dynamické části objektů

```
$dbsql  =  $GLOBALS["C_MySQL"]->query("SELECT * FROM
dip_objekt WHERE index_objekt not in (SELECT fkey_objekt
FROM dip_objekt2)");
while ($db  =  $GLOBALS["C_MySQL"]->next_row($dbsql)){
                    $qrycount=0;
                    $qryx="";
                    $qry2="";
                    $qry3="";
                    if (!$_GET["i"])$_GET["i"]=10;
                    $t=rand(0,10);
                    if ($_GET["i"])
$x=rand(1,$_GET["i"]);
```

```
                    for ($i = 1; $i <= $x; $i++) {
                    $t2=$t+rand(1,5);
                    $qry='INSERT INTO `dis_objekt2`
(`fkey_objekt`, `od2`, `do2`, `jmeno2`) VALUES
("'.$db["index_objekt"].'", "'.$t.'",
"'.(($i<$x)?$t2:'').'", "xz'.rand(0,50000).'");';
                    $qry2 .= (($qry2=="")?'':',
').'("'.$db["index_objekt"].'", "'.$t.'",
"'.(($i<$x)?$t2:'').'", "xz'.rand(0,50000).'")';
                    $t=$t2;
                    if ($qrycount>50){
                         $qry2 = 'INSERT INTO
`dis_objekt2` (`fkey_objekt`, `od2`, `do2`, `jmeno2`)
VALUES '.$qry2.';';
                         if ($_GET["auto"]){ $q =
$GLOBALS["C_MySQL"]->query($qry2); }
                         $out.='<div>'.$qry2.'</div>';

                         $qrycount=0;
                         $qryx="";
                         $qry2="";
                    }
                    }

                    $qry2 = 'INSERT INTO `dis_objekt2`
(`fkey_objekt`, `od2`, `do2`, `jmeno2`) VALUES
'.$qry2.';';
                    if ($_GET["auto"]){ $q =
$GLOBALS["C_MySQL"]->query($qry2); }
                    $out.='<div>'.$qry2.'</div>';
                    echo $out;
                    }
}
```

Generování části objektů obsahujících sezóní data

```
$dbsql  =  $GLOBALS["C_MySQL"]->query("SELECT * FROM
dip_objekt2 WHERE index_objekt2 NOT IN (SELECT
fkey_23objekt2 FROM dip_objekt23)");
```

```
while ($db["db1"]  =  $GLOBALS["C_MySQL"]-
>next_row($dbsql)){}
$qrycount=0;
$qryx="";
$qry2="";
$qry3="";
$sqlx  =  $GLOBALS["C_MySQL"]->query("SELECT * FROM
dis_objekt3 ORDER BY index_objekt3 desc LIMIT 1");
$sqlxres  =  $GLOBALS["C_MySQL"]->next_row($sqlx);
$id = $sqlxres["index_objekt3"]+1;


foreach ($db["db1"] as $key => $value) {
     if (!$value["do2"]) $value["do2"]=$value["od2"]+5;
     for ($i = $value["od2"]; $i <= $value["do2"]; $i++)
{
          $qry='INSERT INTO `dis_objekt3`
(`index_objekt3`,`od3`, `do3`, `fkey_objekt2`) VALUES
("'.$id.'", "'.$i.'", "'.($i+1).'",
"'.$value["index_objekt2"].'");';
          $qry2 .= (($qry2=="")?'':', ').'("'.$id.'",
"'.$i.'", "'.($i+1).'", "'.$value["index_objekt2"].'")';
          $qry='INSERT INTO `dis_objekt23`
(`fkey_23objekt2`,`fkey_23objekt3`) VALUES
("'.$value["index_objekt2"].'", "'.$id.'");
          ';
          $qry3 .= (($qry3=="")?'':',
').'("'.$value["index_objekt2"].'", "'.$id.'")';
          $qryx .= $qry;
          $id++;
          $qrycount++;
          if ($qrycount>80){
               $qry2 = 'INSERT INTO `dis_objekt3`
(`index_objekt3`,`od3`, `do3`, `fkey_objekt2`) VALUES
'.$qry2.';';
               if ($_GET["auto"]){ $q =
$GLOBALS["C_MySQL"]->query($qry2); }
               $out.='<div>'.$qry2.'</div>';
               $qry3 = 'INSERT INTO `dis_objekt23`
(`fkey_23objekt2`,`fkey_23objekt3`) VALUES '.$qry3.';';
```

```
                if ($_GET["auto"]){ $q =
$GLOBALS["C_MySQL"]->query($qry3); }
                $out.='<div>'.$qry3.'</div>';
                $qrycount=0;
                $qryx="";
                $qry2="";
                $qry3="";
           }
     }
}
if ($qry2){
     $qry2 = 'INSERT INTO `dis_objekt3`
(`index_objekt3`,`od3`, `do3`, `fkey_objekt2`) VALUES
'.$qry2.';';
     if ($_GET["auto"]){ $q =  $GLOBALS["C_MySQL"]-
>query($qry2); }
     $out.='<div>'.$qry2.'</div>';
     $qry3 = 'INSERT INTO `dis_objekt23`
(`fkey_23objekt2`,`fkey_23objekt3`) VALUES '.$qry3.';';
     if ($_GET["auto"]){ $q =  $GLOBALS["C_MySQL"]-
>query($qry3); }
     $out.='<div>'.$qry3.'</div>';
}
echo $out;
```

Generování části objektů obsahujících geografická data

```
$dbsql  =  $GLOBALS["C_MySQL"]->query("SELECT * FROM
dip_objekt2 WHERE fkey_poly = 0");
while ($db["db2"]  =  $GLOBALS["C_MySQL"]-
>next_row($dbsql)){}
$sqlx  =  $GLOBALS["C_MySQL"]->query("SELECT * FROM
dis_poly ORDER BY index_poly desc LIMIT 1");
$sqlxres  =  $GLOBALS["C_MySQL"]->next_row($sqlx);
$id = $sqlxres["index_poly"]+1;



$sqlx  =  $GLOBALS["C_MySQL"]->query("SELECT * FROM
dis_zmena ORDER BY index_zmena desc LIMIT 1");
```

```
$sqlxres  =  $GLOBALS["C_MySQL"]->next_row($sqlx);
$idz = $sqlxres["index_zmena"]+1;

foreach ($db["db2"] as $key => $value) {
     $z=0;
     $x=$value["fkey_objekt"];
     if ($y and $x==$y){
          $z=rand(0,1);
          if ($z==1)$z=rand(0,1);
     }else{
          $z=1;
     }
     if ($z==1){
          $zmena=null;
          if ($y and $x==$y){
               $qry='INSERT INTO `dis_zmena`
(`index_zmena`,`popiszmeny`) VALUES ("'.$idz.'", "zmena
tvaru");';
               if ($_GET["auto"]){ $q =
$GLOBALS["C_MySQL"]->query($qry); }
               $out.='<div>'.$qry.'</div>';
               $zmena=$idz;
               $qry='UPDATE `dis_poly` SET fkey_zanikl=
"'.$idz.'" WHERE index_poly="'.($id-1).'";';
               if ($_GET["auto"]){ $q =
$GLOBALS["C_MySQL"]->query($qry); }
               $out.='<div>'.$qry.'</div>';

               $idz++;
          }
          $qry='INSERT INTO `dis_poly`
(`index_poly`,`shape`, `fkey_vznikl`, `fkey_zanikl`)
VALUES ("'.$id.'", "'.rand(0,500000).'", "'.$zmena.'",
"");';
          if ($_GET["auto"]){ $q =  $GLOBALS["C_MySQL"]-
>query($qry); }
          $out.='<div>'.$qry.'</div>';
     }
```

```
     $qry= 'UPDATE `dis_objekt2` SET fkey_poly =
"'.($z==1?$id:($id-1)).'" WHERE
index_objekt2="'.$value["index_objekt2"].'";';
     if ($_GET["auto"]){ $q =  $GLOBALS["C_MySQL"]-
>query($qry); }
     $out.='<div>'.$qry.'</div>';
     if ($z==1){
          $id++;
     }
     $y=$value["fkey_objekt"];
}
echo $out;
```