Univerzita Pardubice

Fakulta ekonomicko-správní

Hodnocení webových rozhraní GIS aplikací pomocí heat map

Vojtěch Zákoutský

Bakalářská práce

2011

Univerzita Pardubice
Fakulta ekonomicko-správní
Akademický rok: **2010/2011**

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Vojtěch ZÁKOUTSKÝ**
Osobní číslo:	**E090899**
Studijní program:	**B6209 Systémové inženýrství a informatika**
Studijní obor:	**Informatika ve veřejné správě**
Název tématu:	**Hodnocení webových rozhraní GIS aplikací pomocí heat map**
Zadávající katedra:	**Ústav systémového inženýrství a informatiky**

Zásady pro vypracování:

Problematika heat map
Implementace aplikace, její výhody a nevýhody
Hodnocení výsledků aplikace

Rozsah grafických prací:

Rozsah pracovní zprávy:

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam odborné literatury:

J. Castagnetto a kol.: PHP - Programujeme profesionálně. Computer Press. Brno 2001. ISBN 80-7226-310-2

P. Staníček: CSS - kaskádové styly. Computer Press. Brno 2003. ISBN 80-7226-872-4

R. Škultéty: JavaScript - programujeme internetové aplikace. Computer Press. Praha 2001. ISBN 80-7226-457-5

L. Lacko: Web a databáze - programujeme internetové aplikace. Computer Press. Praha 2001. ISBN 80-7226-555-5

Vedoucí bakalářské práce: **Ing. Martin Novák**
Ústav systémového inženýrství a informatiky

Datum zadání bakalářské práce: **4. října 2010**

Termín odevzdání bakalářské práce: **6. května 2011**

L.S.

doc. Ing. Renáta Myšková, Ph.D.
děkanka

doc. Ing. Jiří Křupka, Ph.D.
vedoucí ústavu

V Pardubicích dne 4. října 2010

Prohlášení

Tuto práci jsem vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorský zákon, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Souhlasím s prezenčním zpřístupněním své práce v Univerzitní knihovně.

V Pardubicích dne 5. 5. 2011

Vojtěch Zákoutský

Poděkování

Tímto bych chtěl poděkovat vedoucímu mé bakalářské práce Ing. Martinu Novákovi za pomoc při tvorbě bakalářské práce a za čas, který mi věnoval. Dále pak všem, kteří se podíleli na testování aplikace a tvorbě dat pro generování výstupů.

ANOTACE

Tato bakalářská práce se zabývá využitím heatmap při hodnocení aplikačního rozhraní webových GIS aplikací. Dále popisuje, co to jsou heatmapy, jejich vznik a historii. Cílem je implementace heatmapového řešení, které by poskytovalo potřebný výstup pro hodnocení rozhraní zvolené GIS aplikace. Vytvořený skript je pak zhodnocen jak z hlediska využitelnosti a vytvořených výstupů, tak v porovnání s konkurenčními řešeními.

KLÍČOVÁ SLOVA

Heatmapa, GIS aplikace, PHP, JavaScript

TITLE

Rating web interface of GIS applications via Heat Map

ANOTATION

This bachelor thesis deals with the evaluation GIS application interface using the heatmap. It also describes what the heatmap is, their origin and history. The aim is to implement heatmap solution that would provide the required output for evaluation the interface of chosen GIS applications. Script is then evaluated both in terms of efficiency and output generated, and compared with competing solutions.

KEYWORDS

Heatmap, GIS aplication, PHP, JavaScript

Obsah

Úvod

Cílem této práce je obeznámení s pojmem heatmapa, historií jejího vzniku, výhodnosti této reprezentace a efektivity, kterou přináší jejich využití. Heatmapy jsou jednoduše stvořitelné a snadno pochopitelné, navíc v kombinaci s GIS aplikací dávají nové možnosti při hodnocení jejího využívání uživateli.

Webové GIS aplikace jsou dnes hojně využívanou službou na internetu. Mezi známé zástupce patří mapy od společnosti Google, nebo jejich český ekvivalent na portálu mapy.cz od společnosti Seznam.cz. Heatmapy jsou tak způsobem pro efektivní a snadno pochopitelné grafické zobrazení aktivity uživatelů, jejich bodů zájmů a využívání jednotlivých funkcí. Díky vizuální reprezentaci, kterou známe z běžného života, a zobrazení přes danou aplikaci, tak poskytují prostředek pro hodnocení aplikace i lidem, kteří nejsou běžně obeznámeni s tvorbou a návrhem aplikačního rozhraní, například manažerům.

Obsahem práce je i seznámení s výběrem technologií, jazyků a standardů, které byly pro práci využity. Popsány jsou v druhé kapitole včetně jejich výhod a důvodu, proč byly využity.

Druhým a nejdůležitějším bodem práce, popsaným v třetí kapitole, je samotná implementace skriptu na sledování uživatelů a generování heatmap. Skript bude postaven na míru vybrané webové GIS aplikaci a stane se jeho nedílnou součásti. Jeho návrh bude ale vycházet z předpokladu možného použití i v jiných aplikací a bude co nejvíce abstraktní a konfigurovatelný. Výsledek tohoto bodu pak bude zhodnocen a porovnán s dostupnými heatmapovými aplikacemi.

Posledním bodem práce je pak zhodnocení výstupů aplikace a jejich využití v praxi.

1. Heatmapy

Kapitola pojednává o tom, co jsou to heatmapy, jejich vzniku, historii a využití.

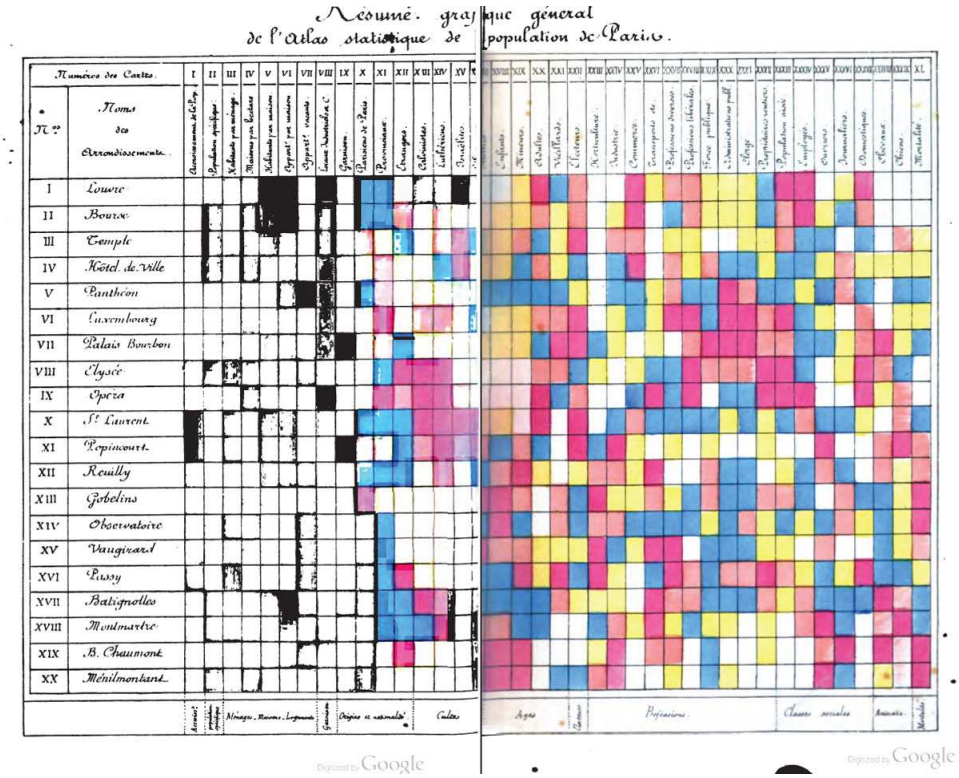
Heatmapy jsou dvojdimenzionální reprezentace dat, kde jednotlivé hodnoty jsou reprezentovány jako barvy. Jejich hlavní výhody spočívají ve dvou bodech. Zaprvé, díky přirozené škále barev podobné teplotní stupnici minimalizují objem znalostí, nutných k jejich pochopení. Například víme, že žlutá je teplejší než zelená, oranžová teplejší než žlutá a červená znamená horko. Poté není těžké přijít na to, že množství tepla je přímo úměrné hladině zobrazované veličiny. Zadruhé, heatmapy jsou zobrazovány přímo nad podnětem, kterého se týkají. Jelikož data nemohou být již blíže, je potřeba jen malé duševní úsilí k jejich přečtení. [2]

Heatmapy mají velký význam pro noviny, hlášení či prezentace, protože sumarizují obrovské množství dat, které by jinak bylo velice obtížné zanést do grafů, pokud jsou v numerické podobě. Heatmapa tak dokáže odhalit podstatu dat včetně vzorů trendů, které se mohou v datech nacházet. [2]

V oblasti uživatelské zkušenosti mohou heatmapy reprezentovat rozličné typy dat od užívání (například kliknutí, stisknutí klávesnice), přesnost, či vizuální podobnost. [2]

1.1 Historie

Heatmapy jako takové získaly věhlas především díky shlukové analýze, jejich původ je ale mnohem starší. Nejstarší známé využití heatmap se datuje do roku 1873 a jeho autorem je Toussaint Loua. Ta byla výsledkem sumarizace 40 různých map Paříže, zobrazujících charakteristiky (národnost, profese, věk, sociální třída) 20 okresů využívající barevnou škálu od bíle, přes žlutou až po červenou. [11]



Obrázek 1: Škálogram od Toussaint Loua [1]

S prvním způsobem jak přeorganizovat datovou tabulku přišel antropolog Petrie (1899). V tabulce reprezentující antropologické nálezy přeskupoval řádky a sloupce tak, aby největší hodnoty byly co nejblíže hlavní diagonále. [11]

S dalším, opět geografickým využitím heatmap přišel v roce 1914 W. C. Brinton. Jednalo se o ohodnocení jednotlivých států USA vzhledem k jejich možnostem vzdělávání (počet studentů, počet vysokých škol, náklady na žáka a další). [11]

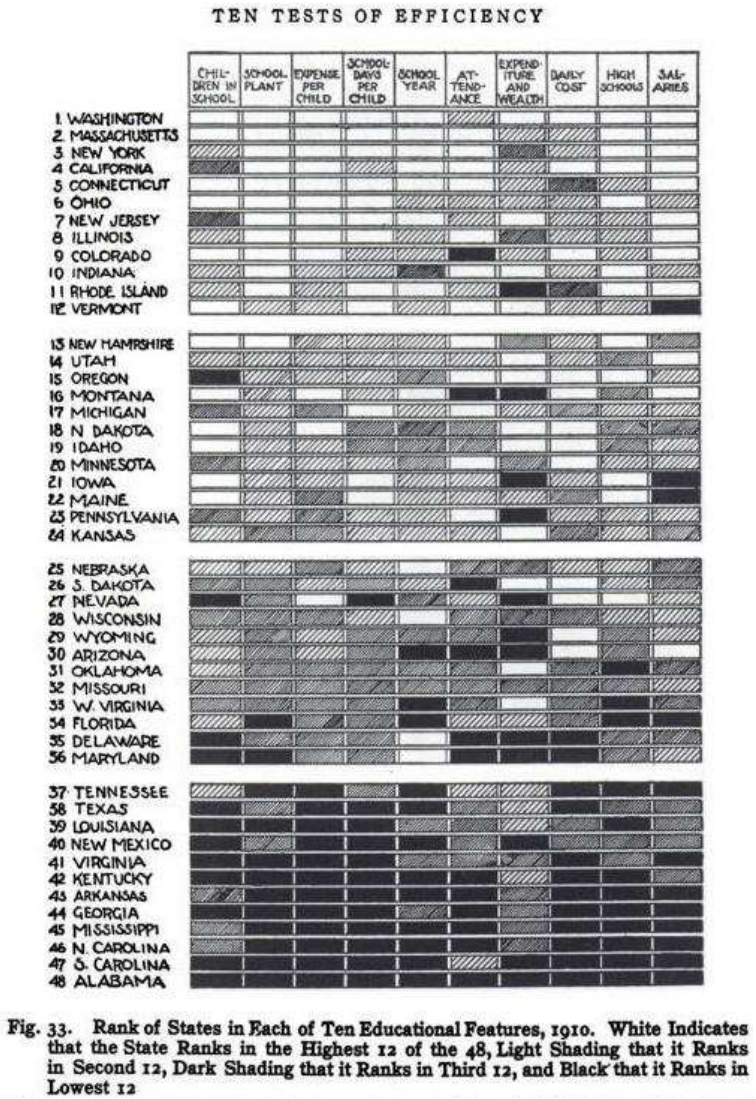


Fig. 33. Rank of States in Each of Ten Educational Features, 1910. White Indicates that the State Ranks in the Highest 12 of the 48, Light Shading that it Ranks in Second 12, Dark Shading that it Ranks in Third 12, and Black that it Ranks in Lowest 12

Obrázek 2: Ohodnocení školství podle Brintona [1]

1.1.1 Guttmanův škálogram

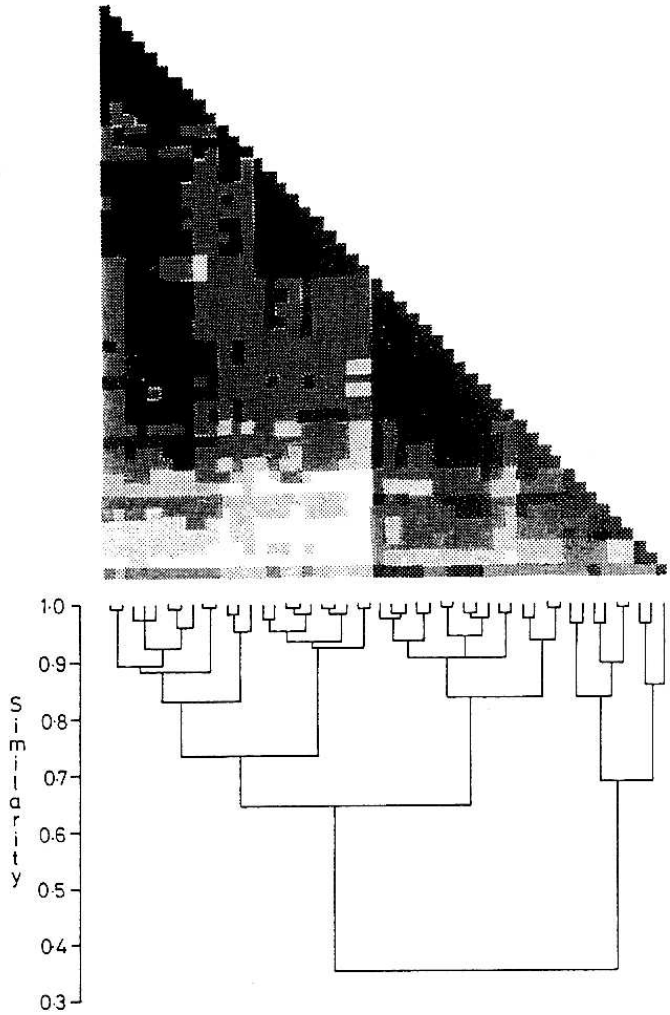
Padesát let po Petrie představil Louis Guttman permutaci matice pro odhalení různých jednorozměrných struktur. Guttmanuv škálogram byla přímá metoda aplikace deterministického modelu na binární matici. V Guttmanově metodě se matice ručně permutovala na jednorozměrnou stupnici s tím, že pod kvazi diagonálou mělo být co nejvíce jedniček a nad ní co nejvíce nul. Taková matice byla prohlášená za škálovatelnou. [11]

Škálogram našel široké uplatnění zejména ve společenských vědách. V roce 1979 přišel Wilkinson s prvním počítačovým programem pro jejich tvorbu. Statistici

Goodman (1975) a Andrich (1978) přišli se stochastickým zobecněním, které umožnilo širší využití škálogramů. [11]

1.1.2 Shluková analýza

Krátce poté, co začal být populární Guttmanův škálogram, našla v heatmapách uplatnění shluková analýza. Průkopníkem této myšlenky byl Peter Sneath (1957) jehož myšlenky později v programu SHADE (1973) použil Robert Ling. [11]



Obrázek 3: Permutovaná matice sloužící k určení podobnosti/nepodobnosti [11]

1.2 GIS aplikace

Pojem geografický informační systém (GIS) se většinou používá pro označení geograficky, resp. prostorově orientované počítačové technologie, integrovaných systémů pro různé aplikace, jakož i nové disciplíny, které se velmi rychle vyvíjí a rozšiřují. Název této technologie pochází z anglického Geographic/Geographical Information System. Jednotlivé pojmy z této oblasti jsou často zaměňovány, používány v různých zemích či oborech v různých významech a někdy naopak splývají – jak v češtině, tak v angličtině. [5]

GIS jako konkrétní aplikace je chápán jako komplexní funkční informační systém „geografického typu“, který je součástí řízení jisté organizační jednotky. Pod GIS lze přiřadit v širším slova smyslu všechny systémy používající a zpracovávající údaje polohově vázané k povrchu Země. V užším slova smyslu se k nim mohou přiřadit jen takové systémy, které jsou schopné vykonávat typické operace (digitalizace mapových podkladů, kartografická projekce, specifické analýzy, kombinování údajů, tvorba mapových výstupů apod.) [5]

V našem případě budeme hovořit o takzvané Web-GIS aplikaci, která je určená pro prohlížení a práci s geografickými daty pomocí internetového prohlížeče. Díky tomu lze do aplikace vložit jednoduchý sledovací kód a sledovat práci uživatelů, převážně kam na stránce klikají. Tyto data budou spolu s dalšími informacemi o GIS aplikaci odesílány na server k uložení a pozdějšímu zpracování. To nám poté umožní zjistit body zájmu uživatelů, které ovládací prvky používají, které nepoužívají vůbec, a celkovou ergonomii prostředí. Dále pak například zajímavé body na mapě.

2. Použité technologie

Tato kapitola pojednává o zvolených technologiích, programovacích jazycích a důvod jejich použití. Popisuje jak jednotlivé jazyky, tak jejich klíčové vlastnosti, funkce a knihovny, které byly použity při implementaci heatmapy. Rozebrány budou i výhody této volby a jiná možná řešení.

2.1 JavaScript

JavaScript je objektový jazyk na straně klienta. Název byl zvolen kvůli popularitě jazyka Java, ale nemá s ním nic společného. Původně měl za velkého konkurenta jazyk Visual Basic Skript od společnosti Microsoft. Ten byl ale podporován jen v prohlížečích Internet Explorer od stejné společnosti a nakonec přestal být používán nadobro.

JavaScript zažil velký rozmach hlavně s příchodem a širokým využíváním technologie AJAX (Asynchronous JavaScript and XML). Ta díky možnosti komunikace prohlížeče se serverem bez nutnosti nového načtení stránky, umožňuje vytvářet uživatelsky přívětivější aplikace.

JavaScript je používán na straně klienta a to na zachytávání jednotlivých událostí a jejich následné odesílání na server. Přehled událostí zobrazuje následující tabulka.

Tabulka 1: Použité události jazyka JavaScript

Název události	**Funkce**
onLoad(*akce*)	Akce se provede po načtení stránky.
onUnload(*akce*)	Akce se provede při přechodu prohlížeče na jinou stránku
onClick(*akce*)	Akce se provede při kliknutí myší na daný objekt.

JavaScript jako takový pracuje s webovou stránkou za pomocí několika hlavních objektů. Použit byl objekt Window a jeho vlastnost Location, pro přístup k aktuální adrese webové stránky a pak objekt Document pro přístup k vlastnostem stránky s mapou.

2.2 CSS

Tabulky kaskádových stylů (Cascading Style Sheets) jsou nadstavbou značkovacích jazyků HTML, XHTML či XML. Slouží k popisu prezentace dokumentu, aniž by jakkoliv ovlivňovaly jejich obsah a strukturu. [9]

Možnosti CSS byly využiti v prezentační části aplikace, kdy byly použity vlastnosti přetékání a určení pozice. Následující přehled ukazuje hlavní použité vlastnosti CSS.

- **Display** - Určuje, jaký model formátování se pro prvek použije a jaké bude generovat elementy.
- **Position** – určuje, jaké poziční schéma se použije pro formátování prvku. Pozice ovlivňuje také následující vlastnosti *top*, *right*, *bottom*, *left*, *z-index* a vlastnosti obtékání *float* a *clear*.
- Souřadnice prvku, tedy vlastnosti **right, left**, **top** a **bottom** – určují vzdálenost horní (top), pravé (right), spodní (bottom) resp. levé (left) hrany okraje (margin) prvku od horní, pravé, spodní, resp. levé hrany jeho omezujícího bloku. Při relativním pozicování (relative) určují posun prvku z jeho pozice v normálním toku.
- **Z-index** – určuje vrstvu, v níž se zobrazuje prvek, pokud je absolutně pozicovaný. V ostatních schématech nemá žádný význam.
- **Overflow** – určuje, zda je obsah blokového prvku oříznut, pokud přesahuje jeho rám. [9]

2.3 PHP

PHP (Hypertext Preprocessor) je programovací jazyk na straně serveru navržený pro tvorbu webových aplikací. Jeho nespornou výhodou je, že je to open-source projekt a je tedy zdarma. Dále je to rozšířenost, snadná instalace, uživatelská základna a množství literatury věnující se právě tomuto jazyku. Jazyk sám se syntaxí velice podobá jazyku C/C++. Další konkurenční výhodou je fakt, že veškeré softwarové zázemí se dá pořídit zdarma, a tak existuje mnoho levných hostingů či dokonce hostingů zdarma. Často kritizovanou nevýhodou je horší implementace objektového programování oproti jiným jazykům a také absence frameworků v základu.

Jazyk PHP byl použit na straně serveru a zajišťuje komunikaci s klientem a dále pak ukládání a následnou reprezentaci dat. Grafické knihovny tohoto jazyka bohužel dovolují jen základní práci s obrázky, ale i tak jsou dostatečně použitelné pro účely generování heatmap.

2.4 MySQL

MySQL je relační databáze standardu SQL. Díky dlouholetému používání kombinace PHP a právě MySQL jsou v PHP implementovány všechny potřebné funkce pro komunikaci s touto databází a existuje i mnoho literatury na toto téma. MySQL sice nedosahuje vždy tak skvělých výkonů jako komerční databázové řešení, zdatně jim sekunduje a pro malé a střední projekty je ideální. Problémem je pomalý vývoj a po převzetí společnosti SUN, která MySQL vyvíjela, společností ORACLE, i nejistá budoucnost.

2.5 Model Klient-Server

Výpočetní model klient/server je dnes nejčastěji využíván aplikacemi, které spadají do široké a velmi vágně definované škatulky „Informačních systémů". Představit si pod tím můžeme například aplikace pro podporu nejrůznějších agend všelijakých firem, institucí a dalších orgánů - od účetnictví, skladového hospodářství, až třeba po systémy na podporu rozhodování. Všechny takovéto aplikace přitom mají některé společné rysy. Například se v nich vždy vyskytují nějaká základní data, která musí být vhodným způsobem uskladněna, a současně k nim musí být zajištěn takový přístup, jaký vyhovuje povaze a charakteru samotné aplikace. Další nezbytností je pak nějaká forma komunikace s uživatelem, spočívající jednak ve sběru dotazů a jiných příkazů a povelů od uživatele směrem k aplikaci, a na druhé straně pak i nezbytné zobrazování (či honosněji prezentaci) získaných výsledků. No a pak tu musí být ještě třetí část, která zajišťuje všechno to co je pro danou aplikaci specifické a konkrétní, neboli realizuje vlastní logiku celé aplikace. Například půjde-li o účetnictví, jsou právě zde implementována všechna pravidla, způsobující správné promítnutí jedné účetní operace do všech účtů, podúčtů, knih, listů apod., kterých se to týká. [8]

Zmíněné tři druhy činností, které jsme si právě vymezili, jsou v různých odborných pramenech označovány různě. Naznačme si zde alespoň jednu terminologii, zavedenou prestižní konzultační firmou The Gartner Group:

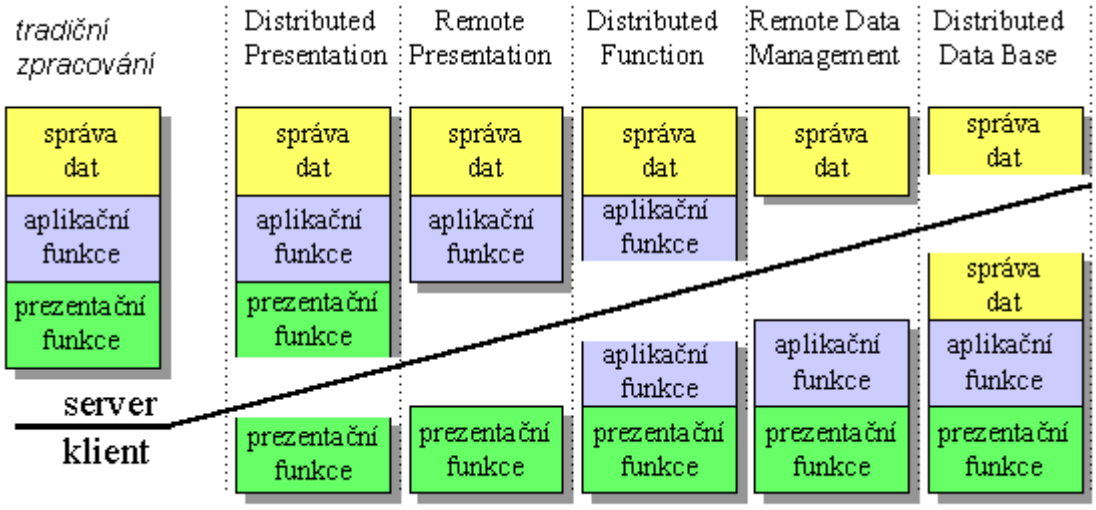
- **Presentation** (uživatelské rozhraní a komunikace s uživatelem, neboli prezentační činnosti),
- **Application Function** (vlastní logika aplikace, tj. aplikační činnosti),
- **Data Management** (správa dat). [8]

Možností, jak rozdělit tyto činnosti mezi klienta a server je celkem 5:

- **Distributed Presentation**: zde jsou prezentační činnosti rozděleny mezi klienta a server (který navíc sám vykonává i všechny zbývající činnosti).

Na straně serveru se nejčastěji jedná o generování výstupů ve znakovém režimu do textového okna, které ale ve skutečnosti není nikde zobrazováno. Jeho obsah je místo toho přenášen klientovi, a teprve ten jej zobrazuje uživateli (obvykle již v grafickém režimu emulujícím znakový režim textového okna). Analogicky pak pro obrácený směr, týkající se vstupů od uživatele. Hlavní výhodou tohoto řešení, které se asi nejvíce blíží modelu host/terminál, je skutečnost, že serverová část aplikace může používat standardní systémové prostředky pro zobrazování na znakových výstupních zařízeních.

- **Remote Presentation**: v této variantě jsou veškeré prezentační funkce ponechány na klientovi, zatímco veškeré aplikační funkce a funkce spojené se správou dat zajišťuje server.
- **Distributed Function**: v této variantě jsou všechny prezentační funkce na klientovi, veškerá správa dat na serveru, a o aplikační funkce se obě složky dělí. Některé činnosti, související s vlastní „logikou" aplikace, tedy zajišťuje sám klient, zatímco zbývající obstarává server.
- **Remote Data Management**: tato varianta předpokládá, že veškeré prezentační i aplikační činnosti zajišťuje klient, zatímco server se věnuje pouze správě dat.
- **Distributed Data Base**: v rámci této varianty zajišťuje většinu činností klient - veškeré prezentační činnosti, veškeré aplikační činnosti, zatímco o udržování a správu dat se dělí se serverem. Jak již název této varianty napovídá, je tato varianta určena zejména pro podporu distribuovaných databází. [8]



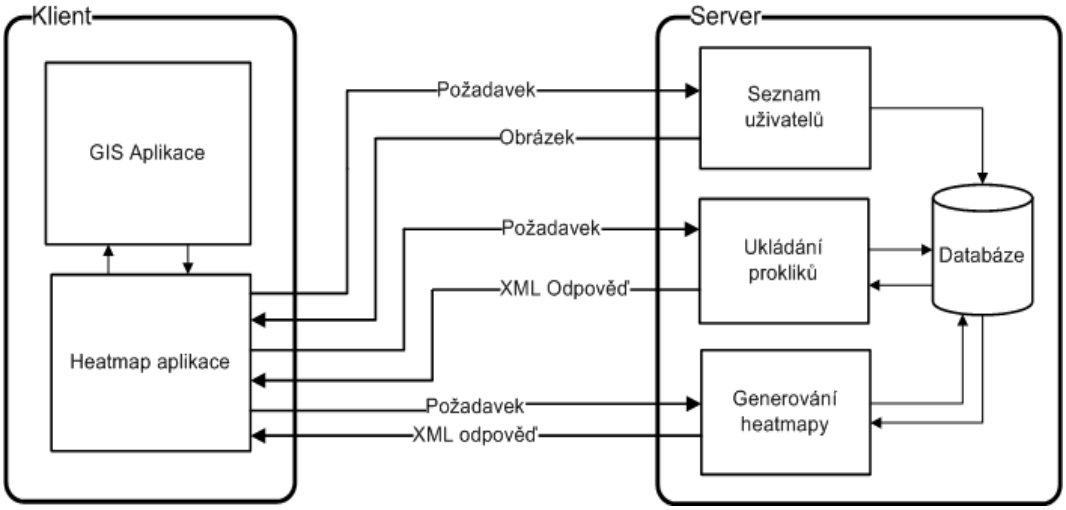
Obrázek 4: 5 variant modelu klient/server [8]

Pro aplikaci byl vybrán model **Distributed Function** kdy část aplikace, která se stará o uložení a zpracování dat, pracuje na serveru a část sledující uživatele, která výsledně zobrazuje heatmapy, je na straně klienta. Datová část reprezentovaná databází MySQL, je na serveru.

3. Návrh a implementace

Tato kapitola pojednává o implementaci celé aplikace heatmap. Aplikace jako taková bude rozdělena na skript na straně klienta, napsaný v jazyce JavaScript, a dále na program na straně serveru, napsaný v jazyce PHP, starající se o ukládání dat do databáze a dále o generování samotné heatmapy.

Dále budou popsány silné a slabé stránky celého řešení. Generované heatmapy budou zhodnoceny jak po stránce použitelnosti, tak po stránce grafické. Výkon aplikace a srovnání s dostupnou konkurencí pak tvoří konec kapitoly.



Obrázek 5: Komunikace mezi serverem a klientem [vlastní]

3.1 Použitá aplikace

Pro implementaci heatmapy byla vybrána GIS aplikace Přemysla Lédla, který ji napsal a obhájil jako svou bakalářkou práci v roce 2010 na Univerzitě Pardubice pod názvem: **AJAX a jeho využití v GIS aplikacích.** Tato webová GIS aplikace, ač velice jednoduchá, umožňuje základní i některé pokročilé operace s mapami. Bakalářská práce posloužila jako dokumentace a usnadnila tak implementaci heatmap. V aplikaci tak byly kromě pozice kliknutí uživatele zaznamenávány všechny parametry aplikace, jako jsou přiblížení, vybraná oblast, jednotlivé vrstvy

a zdali jsou zobrazeny. Ty jsou pak spolu se souřadnicemi odeslány na server a zpracovány při prezentaci.

3.2 Návrh databáze

Návrh databáze se odvíjí od zvolené GIS aplikace a dalších prvků, které bylo třeba ukládat pro další zpracování. Při návrhu byly v konceptuální části vybrány tyto entity, které budou zpracovány databází.

- Datum a čas kliknutí,
- IP adresa počítače,
- ID uživatele,
- souřadnice kliknuti myši (X a Y),
- rozměry mapového pole a vybraná oblast mapy,
- zvětšení mapy,
- jednotlivé vrstvy, a zdali jsou zapnuté a vypnuté.

Jako primární klíč zde byl vybrán čas kliknutí, IP adresa a ID uživatele. Vzhledem k tomu, že všechny ostatní atributy se váží právě k tomuto klíči, byla nakonec celá struktura navržena jako jedna databázová tabulka.

V technologické části byla celá tabulka vytvořena v prostředí databázové platformy MySQL verze 5. Vzhledem k potřebám častého ukládání dat do databáze je jako typ úložiště vhodnější použít MyISAM[1] místo často používané InnoDB[2], která může v určitých případech způsobovat značné prodlení[3].

[1] Starší typ databázového úložiště založené na stromové struktuře uložené v souborech. [6]
[2] Novější typ databáze umožňující transakční zpracování dotazů [7]
[3] Rozdíly v rychlosti databázových řešení byly otestovány Jakubem Vránou [10].

Pro uložení dat byla zvolena jen jedna tabulka a to z důvodů přístupu k datům v reálném čase, jednoduchým dotazům, rychlé odezvě a nižší režii na dotazy při ukládání dat. Z těchto důvodů tabulka nesplňuje ani první normální formu.

3.3 Aplikace na straně klienta

Aplikace na straně klienta je psaná v jazyce JavaScript a je přímo implementována do sledované aplikace. Sleduje hlavní proměnné této aplikace a využívá i některé z jejich funkcí. Náplní práce této aplikace je jednak zjišťovat kliknutí uživatelů a odesílat je na server a dále se stará o rozhraní a vykreslení heatmapy jako takové.

3.3.1 Sledování uživatelů

Pro sledování uživatelů jsou využity události jazyka JavaScript, konkrétně událost onClick, která reaguje na kliknutí myší uživatelem. Je určena pro celý objekt Document, který reprezentuje celou webovou stránku aplikace. Pokud uživatel klikne kdekoliv na stránce, je automaticky vyvolána tato událost a je zpracována funkce, která se postará o odeslání souřadnic kliknutí na server.

Pokud je aplikace spuštěna poprvé, je uživateli přidělen identifikátor. Ten je vytvářen z funkce getTime objektu Date jazyka JavaScript, která vrací počet milisekund od 1. 1. 1970 a dále pak z náhodného čísla v rozmezí 0 až 1000, které je přidáno za časovou značku. Výsledkem je číslo o 14 až 16 číslicích. Tento systém sice nezaručuje přidělení unikátního identifikátoru ve všech případech, pro plánovanou zátěž aplikace v kombinaci s IP adresou počítače je ale dostatečný.

Identifikátor je následně uložen do cookie[4] s platností 2 hodiny a bude následně používán při každém volání aplikace a odeslán spolu s dalšími daty na server. Při

[4] Cookies jsou malé textové informace, které se dají uložit na počítači klienta. [3]

spuštění aplikace ze stejného počítače ale jiného prohlížeče je pak vytvořen nový identifikátor a uložen do nového cookie a chová se tak jako nový uživatel.

Souřadnice kliknutí nám poskytuje jazyk JavaScript jakožto součást DOM. Hraje zde ale roli rozdílnost jednotlivých internetových prohlížečů a i jednotlivých verzí. Hodnoty jsou tak zjišťovány individuálně pro různé verze prohlížečů. Krom pozice kliknutí je zjišťována i pozice a rozměr mapy v aplikaci, pozice některých ovládacích prvků na mapě, rozměr obrazovky, přesněji pak rozměr vykreslované plochy webové stránky. Důležitý je pak parametr *onMap*, který definuje, zdali kliknutí bylo či nebylo na mapě. To se určí jednak z pozice mapy, tedy zda bylo kliknutí uvnitř, či vně plochy zabrané mapou a také podle pozice speciálních ovládacích prvků. Těmi jsou ovládač zvětšení mapy a ovládací kříž sloužící pro pohyb po mapě. Ty jsou umístěny přes mapu, a kdyby byly tato kliknutí započítány, ovlivnilo by to generovanou heatmapu.

Bylo vyzkoušeno několik možností odesílání dat na server. Mezi prvními zde byl AJAX, který se pro tuto možnost na první pohled hodí nejvíce, a díky implementaci funkcí v GIS aplikaci by byla realizace této možnosti nejrychlejší. Bohužel AJAX je v různých prohlížečích různě rychlý, jednotlivé dotazy se provádějí postupně, takže k odeslání dat může dojít s velkým zpožděním a pokud mezi odesláním dojde k načtení nové stránky nebo k zavření internetového prohlížeče, žádná data se neodešlou. U AJAXu jsou také nepříjemná omezení, kdy v některých prohlížečích nelze odeslat požadavek na adresu, která není pod aktuální doménou.

Další možností bylo využití skrytých objektů typu iFrame[5]. Toto řešení není tak elegantní jako řešení s využitím technologie AJAX, obchází ale některá jeho omezení a je podstatně rychlejší. Ani tento způsob se neukázal jako příliš spolehlivý a ne všechna data dorazila na server.

[5] iFrame je element jazyka HTML. Jedná se o plovoucí rám, který může obsahovat jinou internetovou stránku.

Jako nejefektivnější se ukázal způsob, kdy je pomocí jazyka JavaScript vygenerován nový HTML element, konkrétně obrázek. Tomuto obrázku lze přiřadit adresu umístění, která obsahuje všechny parametry. Skript na serveru si pak jen načte data uložená v URL, zpracuje je a vrací prázdný obrázek. Daný obrázek není ani vložen do dokumentu a je po vytvoření automaticky zničen. U této metody nebylo zjištěno žádné zpoždění, je velice spolehlivá, a tak je hojně využívána i u komerčních heatmapových aplikací.

3.3.2 Rozhraní aplikace na straně klienta

Rozhraní aplikace na straně klienta je velice jednoduché. Většina potřebných parametrů pochází přímo z GIS aplikace. Pro ovládání tak slouží červený obdélník v pravém horním rohu GIS aplikace, obsahující tlačítko pro zobrazení/skrytí mapy, volbu typu zobrazení a dále časový filtr a volbu uživatele.

Obrázek 6: Rozhraní aplikace na straně klienta [vlastní]

Tlačítko pro zobrazení mapy interaktivně mění svůj popisek a funkci podle stavu aplikace. Výběr módu aplikace je pak zaškrtávací políčko, kdy nezaškrtnuté značí zobrazení heatmapy jen přes mapové pole a zaškrtnuté pak celostránkový režim.

Časové filtry se pak dělí na dvě trojice políček, kdy první představuje den, měsíc a rok začátku zobrazovaného intervalu a druhá trojice pak opět den, měsíc a rok konce intervalu.

Posledním prvkem je výběr uživatele, pro kterého má být heatmapa vygenerována. Možnosti jsou buď všichni uživatelé, nebo jeden konkrétní. Seznam uživatelů je získáván přímo ze serveru a to z uživatelů, kteří jsou obsaženi v databázi.

3.3.3 Rozhraní pro heatmapu

Druhou funkcí aplikace na straně klienta je vytváření rozhraní pro vygenerovanou heatmapu. Rozhraní není na první pohled nijak bohaté a umožňuje prakticky jen mapu zobrazit a skrýt. To je důsledkem toho, že aplikace sleduje aktuální parametry GIS aplikace a podle toho zobrazuje patřičnou mapu.

Aplikace funguje na principu technologie AJAX. Při vyvolání požadavku na zobrazení heatmap je na server odeslán požadavek s aktuálními parametry aplikace, včetně rozměrů a pozice mapy a následně je mu vrácen XML soubor, který obsahuje parametry jako je *status, src* či *rid*. Tyto parametry jsou podrobně rozebrány v kapitole 3.4.

Pokud XML soubor obsahuje parametr *src*, vloží se do dokumentu nový obrázek, jemuž je přidělena právě tato adresa. Pokud je ve výsledku obsažen parametr *rid*, odešle se na server nový požadavek s tímto parametrem a očekává se nový soubor XML s výsledky. Princip postupného generování mapy je podrobně popsán v kapitole 3.5.

Kromě vygenerované mapy je zbytek stránky překryt čtyřmi průhlednými bloky, čímž se zamezí používání GIS aplikace. Dále se zakáže odesílání kliknutí na server.

3.4 Aplikace na straně serveru

Na straně serveru pracují dvě hlavní aplikace. První se jmenuje *script.php* a stará se o zpracování a uložení kliknutí uživatelů. Druhá má název *generateheatmapextxml.php* a stará se o vygenerování heatmap. Tyto aplikace budou popsány v následujících kapitolách. Nedílnou součástí aplikace je i konfigurační soubor *conf.php*. Ten obsahuje nastavení většiny parametrů včetně přístupových údajů do databáze.

Následující přehled udává důležité proměnné a konstanty, které je nutné před spuštěním aplikace nastavit.

- **Server** – adresa serveru s databází,
- **user, pass** – uživatelské jméno a heslo nutné pro přístup k databázi,
- **database** – jméno databáze,
- **MS_URL** – adresa, kde se nachází naše GIS aplikace (doporučené je uvést celou URL včetně http://),
- **pointsize** – rozměr bodu v pixelech (doporučená hodnota je 20 pixelů),
- **density** – hustota bodů, která slouží k vytvoření mřížky o dané velikosti,
- **cparam** – upravuje špičatost generovaných bodů (v kapitole 3.6.4 označen jako *beta),*
- **mapcache** – udává v sekundách, po jakou dobu se nebude generovat nový, obrázek, přijde-li požadavek na stejnou oblast mapy,
- **maxgeneratingtime** – maximální délka běhu hlavního skriptu (funkce parametru je detailně popsána v kapitole 3.6.7).

3.5 Zpracování kliknutí

O zpracování kliknutí se stará aplikace *script.php*. Všechny potřebné parametry jako souřadnice kliknutí, velikost a pozice mapy, atd. dostane v URL. Ty pak dále zpracovává. Důležité předávané parametry jsou:

- **x, y** – souřadnice x a y kliknutí myší,
- **extent** – rozloha a souřadnice mapy, obsahuje 4 hodnoty – souřadnice levého horního a pravého dolního rohu obdélníku. Je určen v mapových jednotkách,
- **mpx, mpy, mpw, mph** – souřadnice x a y mapy na stránce a její šířka a výška,
- **ip, uid** – IP adresa uživatele, unikátní identifikátor přidělení uživateli
- *další parametry související s mapou a uživatelem.*

Každý uživatel obdrží při prvním přístupu ke GIS aplikaci unikátní identifikátor *uid*. Je kombinací přístupového času a náhodně vygenerované hodnoty. Tato hodnota je dále odesílána s ostatními parametry na server a může být použita k rozlišení kliknutí podle uživatelů.

Nejdůležitější je v tomto případě parametr *extent* neboli rozsah. Tento parametr zpřístupňuje přímo GIS aplikace a udává souřadnice a velikost zobrazovaného území. Jelikož jsou souřadnice kliknutí udané v pixelech a parametr *extent* v jednotkách v závislosti na typu mapy, je nutné souřadnice kliknutí pro další zpracování přepočíst. Parametr *extent* je navíc vztažen k určitému souřadnicovému systému, načež souřadnice kliknutí jsou vztaženy pouze k poloze na obrazovce.

Pro přepočet slouží výše popsané parametry. Nejdříve se vypočítá šířka a délka mapy a podělí se šířkou a délkou v pixelech. Tím je vypočten poměr počtu mapových jednotek na jeden pixel. Ve správném případě by měl být vertikální i horizontální poměr stejný. Souřadnice kliknutí v mapových jednotkách je pak vypočten jako násobek poměru a souřadnic kliknutí v pixelech. K této hodnotě se pak připočte, respektive odečte příslušná souřadnice s parametru *extent*.

Zpracované parametry jsou pak uloženy do MySQL databáze a jsou připraveny pro využití při generování mapy.

3.6 Generování heatmapy

Pro generování samotné mapy je využita standardní grafická knihovna jazyka PHP nazvaná GD. Její funkce nejsou rozsáhlé, ale obsahuje vše potřebné včetně generování průhledných obrázků typu PNG, který slouží jako výstup.

O generování heatmap se stará aplikace *generateheatmapextxml.php* jež poskytuje široké možnosti nastavení jak přes konfigurační soubor *conf.php*, tak přes jednotlivé parametry v URL. Mezi nejdůležitější parametry v URL patří:

- **extent -** rozloha a souřadnice mapy, obsahuje 4 hodnoty – souřadnice levého horního a pravého dolního rohu obdélníku (je určen v mapových jednotkách),
- **map_width, map_height –** šířka a výška generované mapy,
- **fp** – generování mapy v celostránkovém režimu,
- **rid –** identifikátor zdroje, využívá se při opětovném generování stejné mapy.

3.6.1 Barevná škála

Výsledná heatmapa bude reprezentovat rozložení počtu kliknutí na stránce. Tento počet bude reprezentován barvou z barevné škály. Škála obsahuje celkem 100 hodnot, představující přechod od modré, jakožto minima, přes zelenou a žlutou, až k maximální červené. Pro vytvoření tohoto přechodu je využit algoritmus, který postupně mění červený, zelený a modrý kanál barvy.

Obrázek 7: Barevná škala přechodu [vlastní]

3.6.2 Reprezentace mapy

Pro reprezentaci mapy nelze využít klasické dvourozměrné pole jazyka PHP. Pole větší než 1000 x 800 bodů v paměti zabere více než 64MB, což bývá standardní limit, který si může skript alokovat. Proto musí být pro reprezentaci využit přímo obrázek vytvořený pomocí knihovny GD. S ním nelze pracovat přímo jako s běžným polem, ale s jednotlivými body lze pracovat pomocí funkcí knihovny GD. Navíc je toto řešení méně náročné na paměť.

Nejprve je tedy vytvořen obrázek o dané velikosti, standardně 900 x 600 pixelů, a v něm je zapnut kanál alfa, který umožňuje vytváření průhledných obrázků. Následně je celý vyplněn průhlednou barvou. V obrázku je dále zakázáno splývání průhledných barev takzvaný alpha blending.

3.6.3 Vyhledávání bodů

Všechny body jsou uloženy v databázi a jsou předzpracovány, jak bylo popsáno v kapitole 3.5. Zde opět slouží parametr *extent* jako identifikátor požadované oblasti mapy. Díky předzpracování souřadnic kliknutí nyní můžeme zjednodušit dotaz a vybrat jen body, kde kliknutí náleží do potřebné oblasti.

Všechny souřadnice jsou nyní v mapových jednotkách, ale generují se v pixelech. Souřadnice se tak přepočtou obdobným postupem jaký je popsán v kapitole 3.5. Následně jsou uloženy do vícerozměrného pole, které slouží pro další zpracování.

Pole s body je vícerozměrné. První dva indexy určují souřadnice x a y, v tomto pořadí a jako hodnota slouží jednorozměrné pole se dvěma údaji. První je počet kliknutí na tento bod, druhý je velikost generovaného bodu. Pokud více bodů odpovídá stejným souřadnicím, zvýší se hodnota prvního parametru o 1. Při nastavení parametru *density* v konfiguračním souboru, jsou body v daném okolí slučovány do jednoho bodu. Je tak vytvořena mřížka o daném rozestupu pixelů.

Pokud jsou generovány body v celoobrazovkovém módu, jsou nejprve načteny body, které jsou mimo mapu a jsou uloženy do stejného pole jako body na mapě. Body na mapě jsou pak načteny běžným způsobem s tím, že jejich pozice na výsledné mapě je posunuta vůči začátku o pozici mapy. Pozice mapy je odesílána v URL a posun bodů tak odpovídá aktuálnímu umístění mapy na obrazovce.

3.6.4 Vykreslování bodů do mapy

Pro vykreslení bodů je využit prázdný obrázek a pole s body. Barva jednotlivých pixelů v obrázku je reprezentována celočíselnou hodnotou, do jednotlivých bodů

tak lze dočasně uložit i jinou celočíselnou hodnotu potřebnou pro generování mapy, ta pak bude převedena na barevnou škálu.

Pole s body je postupně procházeno a z hodnot jsou pak do obrázku vykreslovány body. Hodnoty jsou souřadnice *x* a *y, výška bodu* odpovídající počtu kliknutí na daný bod a pak *velikost bodu*. Na souřadnici **x** a **y** je vložena *výška bodu*. Do okolí odpovídajícímu *velikosti bodu* se vloží postupně se snižující hodnoty, které představují efekt rozmazání. Efekt rozmazání je generován vzorcem, který vzešel vlastním testováním různých variant. Původní vzorec vycházel z funkcí s radiální bází. Generování probíhá podle následujícího vzorce.

Rovnice 1: Funkce vytvářejícíefekt rozmazání

$$hodnota = vyska\_bodu - alfa * \left(\frac{vyska\_bodu}{pocet\_barev} \right) * \sqrt{(x - cx)^2 \pm (y - cy)^2}$$

Hodnota je výsledná hodnota daného bodu. *Pocet_barev* zde reprezentuje počet barev ve škále barevného přechodu. Hodnota *cx* a *cy* určují střed generovaného bodu a celá odmocnina reprezentuje euklidovskou vzdálenost daného bodu od středu.

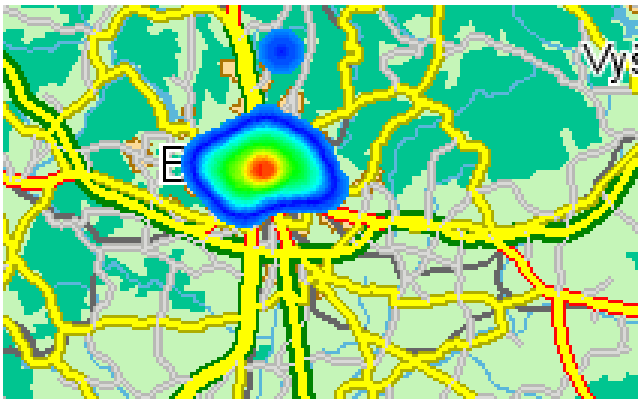
Rovnice 2:Výpočet parametru alfa

$$alfa = \frac{vyska\_bodu}{beta * \left(\frac{vyska\_bodu}{pocet\_barev} \right) * \sqrt{(cx)^2 \pm (cy)^2}}$$

Parametr *alfa* určuje špičatost. Jedná se o výšku bodu podělenou hodnotou v nejvzdálenějším bodu od středu a tedy s nejnižší hodnotou, dále vynásobený parametrem *beta*. Ten ještě více upravuje špičatost. Po testování byla jako ideální zvolena hodnota 0,6. Hodnoty jsou dále zaokrouhleny na celá čísla. Záporné hodnoty jsou změněny na nulu. Přepočet znázorňuje následující obrázek.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
2	-3	-3	-2	-1	-1	0	0	0	1	0	0	0	-1	-1	-2	-3	-3	-4	-5
3	-3	-2	-1	0	0	1	1	2	2	2	1	1	0	0	-1	-2	-3	-3	-4
4	-2	-1	0	1	1	2	3	3	3	3	3	2	1	1	0	-1	-2	-3	-4
5	-1	0	1	2	2	3	4	4	4	4	4	3	2	2	1	0	-1	-2	-3
6	-1	0	1	2	3	4	5	5	5	5	5	4	3	2	1	0	-1	-2	-3
7	0	1	2	3	4	5	6	6	6	6	6	5	4	3	2	1	0	-1	-2
8	0	1	3	4	5	6	7	7	8	7	7	6	5	4	3	1	0	-1	-2
9	0	2	3	4	5	6	7	8	9	8	7	6	5	4	3	2	0	-1	-2
10	1	2	3	4	5	6	8	9	10	9	8	6	5	4	3	2	1	-1	-2
11	0	2	3	4	5	6	7	8	9	8	7	6	5	4	3	2	0	-1	-2
12	0	1	3	4	5	6	7	7	8	7	7	6	5	4	3	1	0	-1	-2
13	0	1	2	3	4	5	6	6	6	6	6	5	4	3	2	1	0	-1	-2
14	-1	0	1	2	3	4	5	5	5	5	5	4	3	2	1	0	-1	-2	-3
15	-1	0	1	2	2	3	4	4	4	4	4	3	2	2	1	0	-1	-2	-3
16	-2	-1	0	1	1	2	3	3	3	3	3	2	1	1	0	-1	-2	-3	-4
17	-3	-2	-1	0	0	1	1	2	2	2	1	1	0	0	-1	-2	-3	-3	-4
18	-3	-3	-2	-1	-1	0	0	0	1	0	0	0	-1	-1	-2	-3	-3	-4	-5
19	-4	-3	-3	-2	-2	-1	-1	-1	-1	-1	-1	-1	-2	-2	-3	-3	-4	-5	-6
20	-5	-4	-4	-3	-3	-2	-2	-2	-2	-2	-2	-2	-3	-3	-4	-4	-5	-6	-7

Obrázek 8: Výpočet efektu rozmazáni pro velikost bodu 20 [vlastní]



Obrázek 9: Výřez mapy, ukázka reálného efektu rozmazání [vlastní]

Pokud jsou dva či více bodů blízko sebe a vzájemně se překrývají, dochází mezi nimi k interakci, jež způsobí navýšení hodnoty ve společné oblasti. Nejlépe vypadající výsledek vykazuje prosté sečtení hodnot obou bodů.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
2	-3	-3	-2	-1	-1	0	0	0	1	0	0	0	-1	-1	0	0	0	1	1	0	0	0	-1	-1	-2	-3	-3	-4	-5
3	-3	-2	-1	0	0	1	1	2	2	2	1	1	0	0	1	1	2	2	2	2	1	1	0	0	-1	-2	-3	-3	-4
4	-2	-1	0	1	1	2	3	3	3	3	3	2	2	2	2	3	3	3	3	3	3	2	1	1	0	-1	-2	-3	-4
5	-1	0	1	2	2	3	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	3	2	2	1	0	-1	-2	-3
6	-1	0	1	2	3	4	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	4	3	2	1	0	-1	-2	-3
7	0	1	2	3	4	5	6	6	6	6	7	7	7	7	7	7	6	6	6	6	6	5	4	3	2	1	0	-1	-2
8	0	1	3	4	5	6	7	7	8	7	8	9	9	9	9	8	7	8	8	7	7	6	5	4	3	1	0	-1	-2
9	0	2	3	4	5	6	7	8	9	8	9	9	9	9	9	9	8	9	9	8	7	6	5	4	3	2	0	-1	-2
10	1	2	3	4	5	6	8	9	10	10	10	9	9	9	9	10	10	10	10	9	8	6	5	4	3	2	1	-1	-2
11	0	2	3	4	5	6	7	8	9	8	9	9	9	9	9	9	8	9	9	8	7	6	5	4	3	2	0	-1	-2
12	0	1	3	4	5	6	7	7	8	7	8	9	9	9	9	8	7	8	8	7	7	6	5	4	3	1	0	-1	-2
13	0	1	2	3	4	5	6	6	6	6	7	7	7	7	7	7	6	6	6	6	6	5	4	3	2	1	0	-1	-2
14	-1	0	1	2	3	4	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	4	3	2	1	0	-1	-2	-3
15	-1	0	1	2	2	3	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	3	2	2	1	0	-1	-2	-3
16	-2	-1	0	1	1	2	3	3	3	3	3	2	2	2	2	3	3	3	3	3	3	2	1	1	0	-1	-2	-3	-4
17	-3	-2	-1	0	0	1	1	2	2	2	1	1	0	0	1	1	2	2	2	2	1	1	0	0	-1	-2	-3	-3	-4
18	-3	-3	-2	-1	-1	0	0	0	1	0	0	0	-1	-1	0	0	0	1	1	0	0	0	-1	-1	-2	-3	-3	-4	-5
19	-4	-3	-3	-2	-2	-1	-1	-1	-1	0	0	0	-2	-2	-1	-1	-1	-1	-1	-1	-1	-1	-2	-2	-3	-3	-4	-5	-6
20	-5	-4	-4	-3	-3	-2	-2	-2	-2	0	0	0	-3	-3	-2	-2	-2	-2	-2	-2	-2	-2	-3	-3	-4	-4	-5	-6	-7

Obrázek 10: Výpočet efektu rozmazání pro dva blízké body [vlastní]

3.6.5 Normalizace

Barevná škála má omezený rozsah, v našem případě 100 hodnot, které odpovídají přechodu od modré k červené. Počet kliknutí v bodech ale může nabývat značných velikostí a je nutné hodnoty přepočítat. K tomuto slouží proces normalizace.

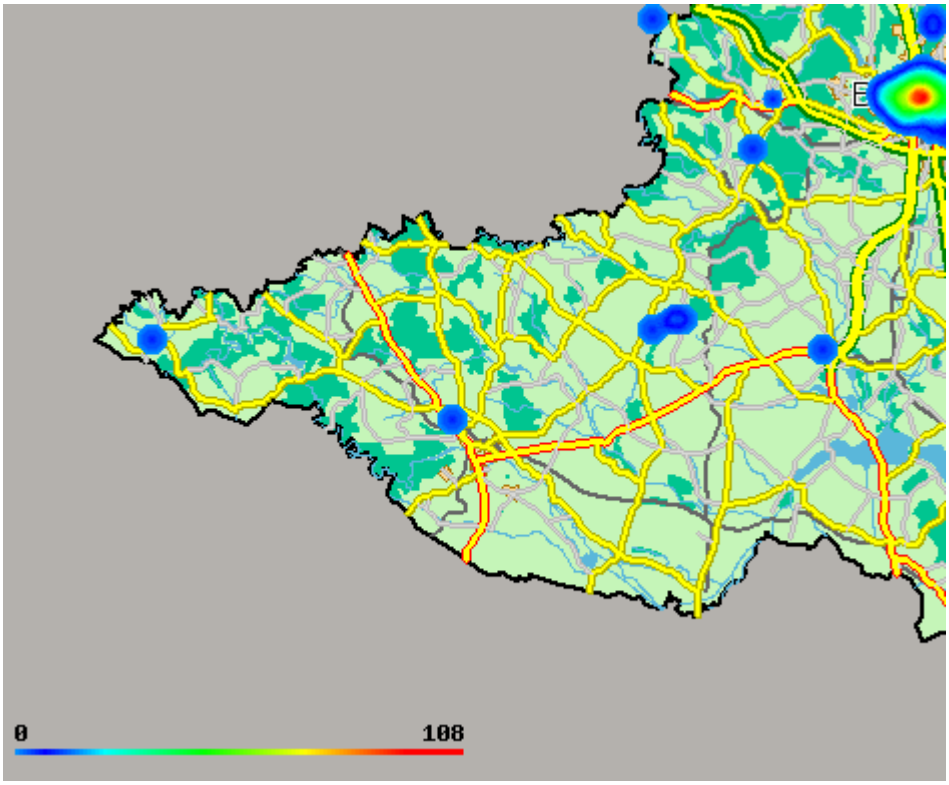
Jelikož nejsou data uložena v klasické tabulce a přístup k jednotlivým bodům je časově i algoritmicky náročný, nedochází zde k přepočítání obrázku na normalizované hodnoty. Místo toho je vypočítána pouze norma z podílu velikosti barevné škály a maximální hodnoty velikosti bodu. Ta je pak využita při vykreslování finálního obrázku, což je popsáno v následující kapitole.

3.6.6 Vykreslování mapy

Mapa je vykreslena z obrázku upraveného dle popisu v předchozích kapitolách. Nyní obsahuje dva druhy bodů: původní průhledné body a body odpovídající námi vygenerovaným hodnotám. Skript projde všechny body a upraví jen ty, které neodpovídají průhledné barvě. Hodnota je nejprve vynásobena *normou*

a zaokrouhlena tak, aby odpovídala jednomu z indexů barevné škály. Z tabulky barev je pak vybrán odpovídající odstín a uložen zpátky do obrázku.

Na závěr je do levého dolního rohu vloženo měřítko. To obsahuje jednak barevnou škálu od nejnižší po nejvyšší, která byla v heatmapě použita, a dále jsou nad ní vloženy informace o minimální a maximální hodnotě.



Obrázek 11: Výřez mapy s měřítkem [vlastní]

3.6.7 Opakované volání aplikace

Vykreslování heatmap je časově náročný proces, který může narážet na limity nastavení jazyka PHP. Tím může být především proměnná *time_limit*, která určuje maximální dobu běhu skriptu v sekundách a jejíž hodnota nebývá na mnoha serverech uživatelsky měnitelná. Skript tedy počítá s tím, že mapa nemusí být vygenerována v daném čase.

Konfigurační soubor obsahuje konstantu *maxgeneratingtime*, udávající dobu běhu hlavního programu v sekundách. Její hodnotu je doporučeno nastavit na *time_limit* snížený o časovou rezervu 5 - 10 sekund.

Program je rozdělen na jednotlivé úseky, které postupně provádí úkoly popsané v předchozích kapitolách. Postup generování je udáván proměnnou *status*.

Pokud skript během generování přesáhne konstantu *maxgeneratingtime*, uloží potřebné hodnoty do speciálního souboru a vytvoří unikátní identifikátor *rid*. Ten odešle spolu s proměnnou *status* zpátky aplikaci na straně klienta ve formátu XML, viz kapitola 3.5. Ta obdrží informaci o nedokončeném úkolu a zavolá aplikaci znovu právě s parametrem *rid*. Pokud je skript ukončen při generování bodů do obrázku nebo později, je dočasně uložen i doposud vytvořený obrázek.

Hodnota parametru *rid*, název vygenerovaného obrázku a název souboru s dočasnými daty jsou stejné. Jsou tvořeny časovou značkou, parametry mapové aplikace a výběrem uživatele. Z řetězce je posléze vytvořen hash, sloužící jako hodnota parametru a název souborů. Pokud je v aplikaci povolena cache, lze pak podle aktuálního nastavení aplikace vygenerovat shodný hash a načíst potřebné soubory.
Parametry ukládané do dočasného souboru jsou:

- **imagesource** – název souboru s obrázkem,
- **status** – status, na kterém aplikace skončila,
- **parametr1, parametr2** – pomocné parametry odpovídající indexům v cyklech, kde program skončil,
- **extent** – rozsah mapy,
- **map_width, map_height** – výška a šířka mapy,
- **posun_x, posun_y** – posun bodů na mapě, který je využit v zobrazení na celou stránku,
- **mapmin, mapmax** – minimum a maximum bodů na mapě,
- **qi** – udává, na kterém místě se zastavilo načítání bodů z databáze.

Aplikace volána s parametrem *rid* si nejprve načte informace uložené v souboru a přenastaví se do bodu, ve kterém skončila. Pokud byla například aplikace

ukončena při konečném vykreslování mapy, jsou přeskočeny předchozí kroky, je načten doposud vygenerovaný obrázek a dále se pokračuje z daného místa.

3.6.8 Časový filtr

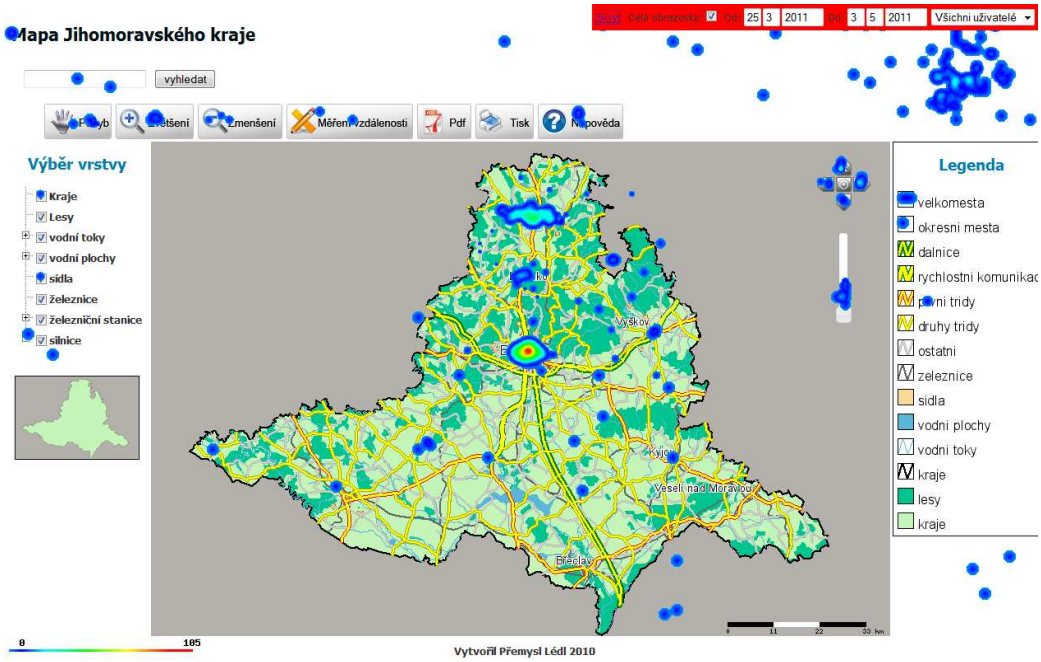
Skript umožňuje generování heatmapy v zadaném časovém intervalu. Interval je určen parametry *starttime* a *endtime* jakožto začátek a konec sledovaného období. Jejich hodnota je určena pomocí časové značky systému UNIX. Parametry jsou nepovinné, a pokud nejsou vyplněny, vyberou se všechny body. Pokud je vyplněn jeden z nich, jsou vybrány body, které byly do databáze uloženy po, respektive před danou časovou značkou, případně v zadaném intervalu.

Aby byla práce s heatmapou jednoduchá, byl do aplikace na straně klienta přidán formulář s volbou zobrazovaného období. Zvolené datum je automaticky převedeno na odpovídající časovou značku a odesláno spolu s dalšími hodnotami na server.

3.6.9 Výstupy z aplikace

Výstup z aplikace je dvojího druhu: mapa pouze pro vybrané mapové políčko a dále pak mapa pro celou aplikaci, obsahující kliknutí i pro rozhraní aplikace.

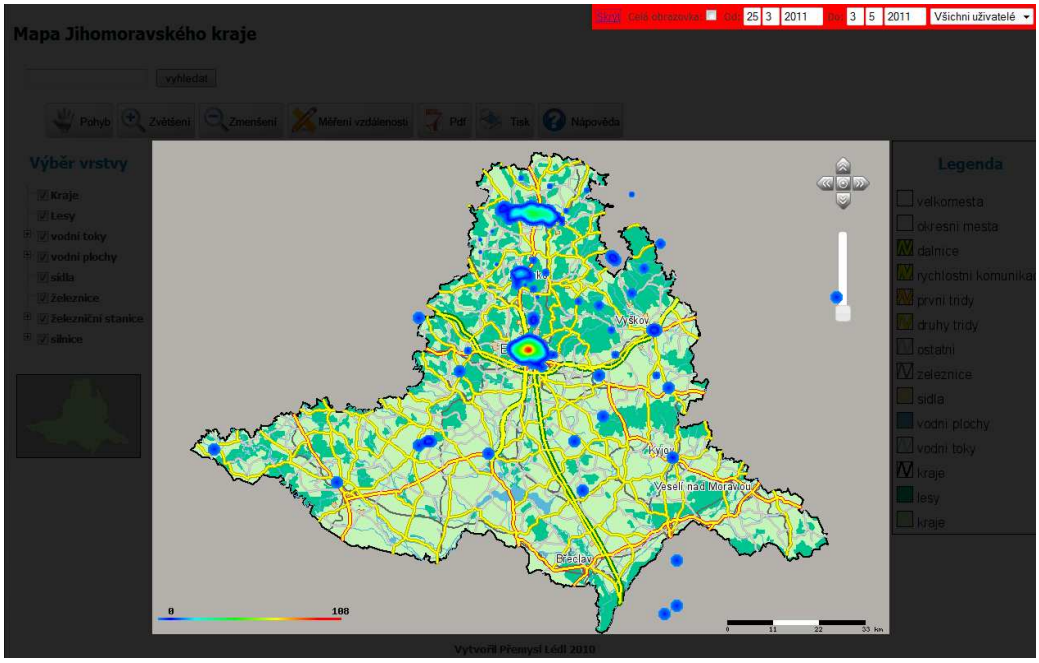
Následující obrázky představují výstupy aplikace při zobrazení na celou obrazovku a při zobrazení jen nad mapou. Nastavení GIS aplikace je stejné. V zobrazení na celou obrazovku je mapa zobrazena přes celou aplikaci a jsou vidět body odpovídající kliknutím. Při zobrazení nad mapou jsou kolem vygenerované heatmapy černé průhledné bloky.



Obrázek 12: Výstup z aplikace při zobrazení na celou obrazovku [vlastní]

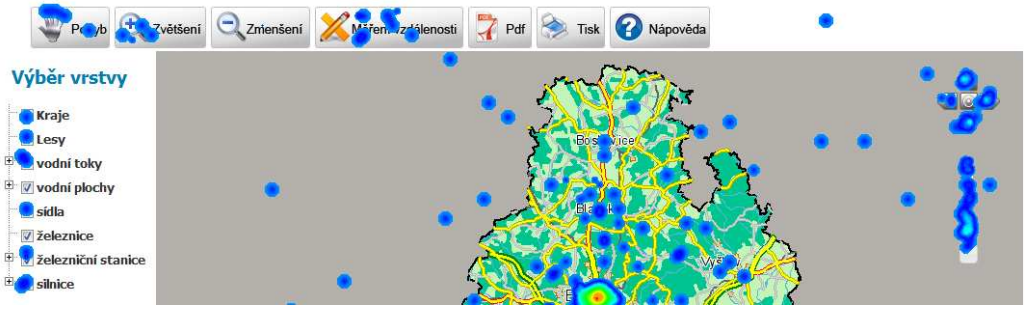
Výstup na obrázku 12 nám dává přehled o celkovém využití dané aplikace. Z předchozích obrázků je jasně patrné, že uživatele zajímalo město Brno. Naopak jiná města je téměř nezajímala. Dále můžeme zhodnotit přehlednost GIS aplikace a využívání jednotlivých funkcí. Na obrázku 13 je pak stejný výstup zobrazený v módu jen nad mapou.

Na obrázcích 12 a 13 jsou zobrazeny prokliky od všech uživatelů. Volbou ve filtru můžeme zobrazovat kliknutí od jednotlivých uživatelů a sledovat rozdílné zájmy a využívání funkcí. To nám dává lepší přehled a přidává další možnosti hodnocení aplikace.



Obrázek 13: Výstup z aplikace při zobrazení nad mapou [vlastní]

Na následujícím obrázku je výstup aplikace po použití několika studenty 2. ročníku. Je zřetelné využití posuvníku pro přiblížení mapy a časté použití navigačních šipek. Dále také tlačítko pro měření vzdálenosti bylo několikrát využito. Využití možnosti zobrazení jen některých vrstev je ojedinělé. Naopak tlačítka pro výstup ve formátu pdf, tisk nebo nápověda nebyly použity nikdy.



Obrázek 14: Výstup z aplikace - využívání jednotlivých funkcí [vlastní]

3.7 Výhody a nevýhody aplikace

Mezi hlavní výhody patří jednoduchost a to jak ovládání aplikace na straně klienta, tak instalace na server a dále pak konfigurovatelnost. Většina parametrů lze

skriptu odeslat v URL a tím lze upravovat její chod bez nutnosti úpravy nastavení na serveru. Stačí tak upravit jen prezentační část v jazyce JavaScript.

Druhou nespornou výhodou je orientace řešení přímo na GIS aplikaci. Žádné dostupné ať již komerční či open source řešení nepočítá s obdobným nasazením a neumí oddělit data pro mapu a pro aplikaci.

Aplikace také počítá s nasazením na méně výkonných či více omezených serverech a umožňuje generování mapy ve více krocích. Lze ji tak použít i tam, kde by jiná řešení nedokázala správně fungovat.
Velkou nevýhodou je rychlost aplikace. Celostránková heatmapa s 1500 kliky a velikostí 1200 x 800 je vytvořena v průměru za 15 sekund. Nutno ale podotknout, že v tomto případě byl kvůli serveru nastaven parametr *maxgeneratingtime* na 5 sekund, a skript tak byl spouštěn vícekrát. Při vyšším časovém limitu by mohl být obrázek vytvořen během jednoho běhu aplikace a čas by mohl být kratší.

Další nevýhodou je použitá prezentační vrstva. Aplikace postavené na technologii Flash umožňují změnu výstupu v reálném čase. U map generovaných pomocí grafické knihovny PHP je nutné vždy spustit skript a vygenerovat nový výstup. Technologie Flash také značně snižuje zátěž serveru. Ten jen odesílá strukturovaná data a o prezentaci se stará až klient. Pro změnu výstupu mnohdy není potřeba ani dotaz na server a jsou pouze znovu zpracována již obdržená data.

3.8 Porovnání s dostupnými službami

Zde jsou představeny známé heatmapové aplikace a služby, které nabízejí obdobné služby jako navržená aplikace. Pro porovnání byla vybrána jedna česká aplikace, jedna open source aplikace a jedna známá zahraniční aplikace. Jelikož žádná z aplikací není optimalizována pro práci s heatmapami, jsou porovnávány hlavně možnosti, výkon a použitelnost.

3.8.1 mYx

Mezi známé heatmapové služby patří mYx. O veškerý provoz se stará sám provozovatel, na uživateli je jen implementace. Výhodou je zpracování celé aplikace v češtině a to včetně přehledně udělaného FAQ[6]. Nevýhodou je, že aplikace je až na 30 denní zkušební lhůtu placená a to v rozmezí 260-800Kč měsíčně podle velikosti sledovaného webu.

Samotná aplikace je pak velice přehledně udělaná. Pro prezentační část byla zvolena technologie Flash[7], která je pro tyto účely vhodnější než generování mapy pomocí PHP. Mapa se tak generuje přímo v prohlížeči, umožňuje různé filtrace, změnu sledovaného období, změnu barev a další možnosti v reálném čase. Navíc je celá zabezpečena pomocí uživatelského jména a hesla proti nechtěnému prohlížení cizími osobami.

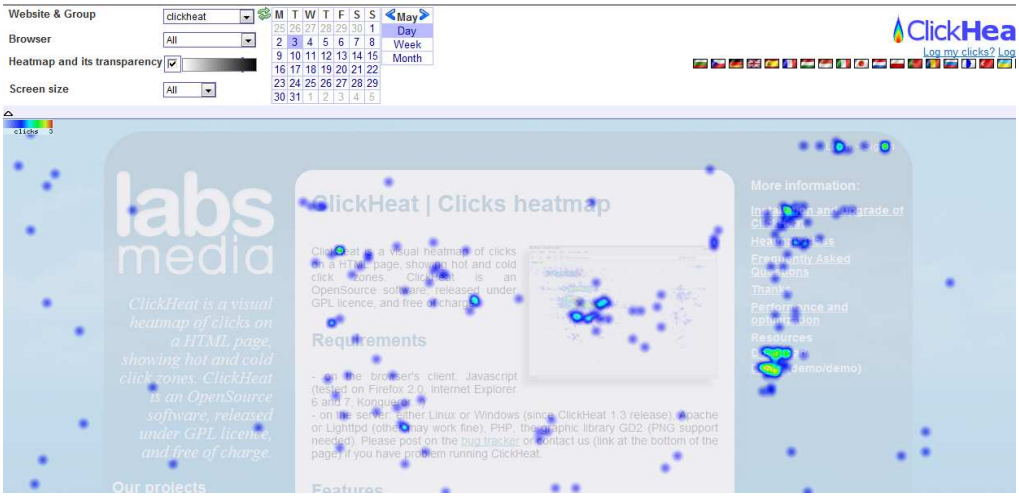
K této aplikaci je možné si přiobjednat i další analytické služby od profesionálních společností z oblasti internetového marketingu.

Obrázek 15: Náhled rozhraní myx.cz [vlastní]

[6] Z anglického Frequently Asked Questions neboli často kladené otázky
[7] Flash je multimediální platforma, dříve určena především pro tvorbu křivkových animací, posléze interaktivních animací, později pak webových aplikací a her, v současné době pak obecně internetových aplikací [4]

3.8.2 ClickHeat

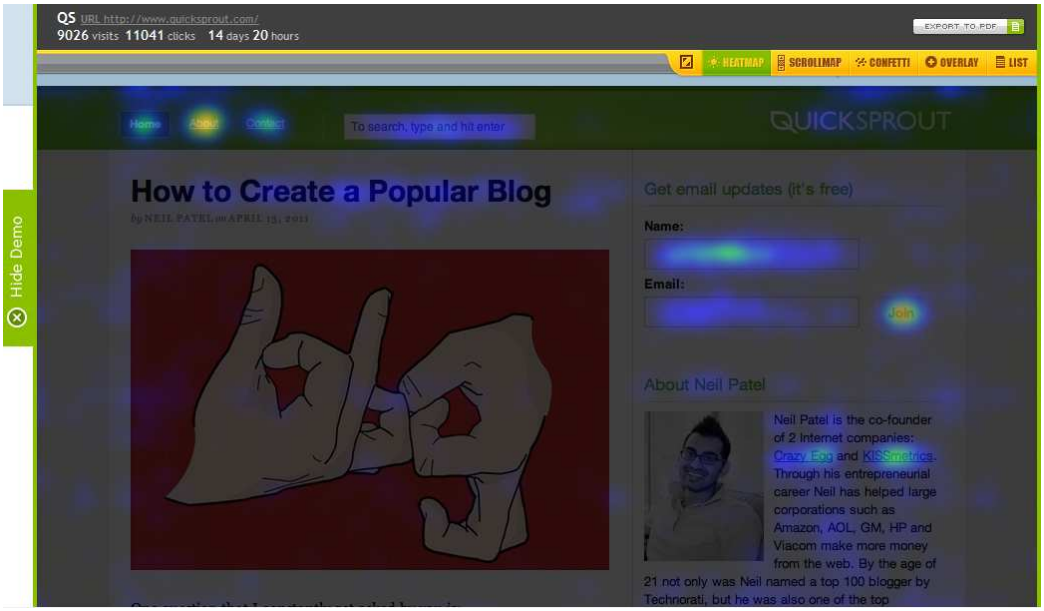
Click heat je open source aplikace pro generování heatmap na webu. Naprogramována je v PHP a využívá i JavaScript a CSS. Výhodou je, jak již bylo jednou zmíněno, nulová cena a upravitelnost. Dále také, že rozhraní aplikace je provedeno v mnoha jazycích včetně češtiny. Nevýhodou je dokumentace pouze v anglickém jazyce a pak celkové provedení aplikace, které se v možnostech ani provedení nemůže komerčním aplikacím rovnat. Její možnosti jsou taktéž omezené, povoluje pouze časový filtr a změnu průhlednosti heatmap přes překrývanou webovou stránku. Aplikace podporuje zabezpečení přístupu k heatmapě pomocí hesla. Mezi další nevýhody patří nutnost instalace a zprovoznění aplikace samotným uživatelem, čemuž nenapomáhá absence jakéhokoliv manuálu.



Obrázek 16: Náhled rozhraní ClickHeat [vlastní]

3.8.3 CrazyEgg

Aplikace podobná mYx. Pro prezentační vrstvu je opět použita technologie Flash a to jí dává lepší vizuální možnosti. Navíc kromě standardní heatmap nabízí přehledné sledování prokliků jednotlivých hypertextových odkazů na stránce. Naopak ale nenabízí ani základní časové filtrování, případně změnu vzhledu heatmap. Použitá barevná paleta není dostatečně výrazná. Oproti mYx je provedení o dost jednodušší a zaostává i po vizuální stránce.



Obrázek 17: Náhled rozhraní CrazyEgg [vlastní]

Závěr

Prvním cílem této práce bylo seznámení s heatmapami, jejich historií a využitím. Je zde popsán jejich vznik a prvotní využití, které vyústilo v další vývoj a využití v dalších odvětvích lidské činnosti. Heatmapy dnes využíváme v běžném životě, aniž bychom to postřehli.

Dále byly popsány technologie potřebné pro implementaci našeho řešení nad danou GIS aplikaci. Jedná se především o jazyky PHP a JavaScript, které slouží na straně serveru a straně uživatele.

Hlavním bodem byla samotná implementace řešení. Začlenění skriptu do GIS aplikace je velice jednoduché a pro vyšší efektivitu využívá i některé její funkce. Naproti tomu generování heatmap je jak algoritmicky, tak časově náročné a na méně výkonném serveru může docházet ke zpoždění, či dokonce k předčasnému ukončení bez požadovaných výsledků. Proto musel být algoritmus několikrát přepsán, aby se snížily jeho paměťové nároky, zrychlil se a bylo ho možné opětovně zavolat tak, aby dokončil již rozdělanou práci.

Znalosti při implementaci byly následně využity při zhodnocení celé aplikace a následně při porovnání s dostupnými heatmapovými řešeními. Tato aplikace těží především z toho, že byla přímo napsána pro GIS aplikaci a ze své konfigurovatelnosti a možnosti úprav. Naopak zaostává především v prezentační části kvůli využití jiné technologie.

Posledním bodem jsou pak samotné výstupy. Již při implementaci byl na vzhled a přehlednost výstupu kladen maximální důraz. Díky snadné čitelnosti a jednoduchosti, je pak možné zhodnotit využití dané GIS aplikace. Data v databázi navíc obsahují další cenné údaje o chování uživatelů a jsou tak vhodná k dalšímu zkoumání a zpracování, nejen pomocí heatmap.

Seznam použitých zkratek

AJAX	Asynchronous JavaScript and XML
CSS	Cascading Style Sheets
DOM	Domument Object Model
GIS	Geographic information system
HTML	Hypertext Markup Language
PDF	Portable Document Format
PHP	Personal home page / Hypertext Preprocessor
PNG	Portable Network Graphics
URL	Uniform Resource Locator
XHTML	eXtensible HyperText Markup Language
XML	XML Linking Language

Seznam obrázků

Seznam tabulek

Seznam rovnic

Seznam příloh

Seznam použité literatury

[1] FRIENDLY, Michale; DENIS, Daniel J. Datavis.ca [online]. c2009, [cit. 2011-04-12]. Milestones in the History of Thematic Cartography, Statistical Graphics, and Data Visualization. Dostupné z WWW: < http://www.datavis.ca/milestones/index.php >.

[2] JACKO, Julie A. *Human-computer Interaction : New trends*. Berlín: Springer, 2009. 913 s. ISBN 978-3-642-02573-0.

[3] Jak psát web [online]. 2011, [cit. 2011-04-03]. Cookies. Dostupné z WWW: <http://www.jakpsatweb.cz/enc/cookies.html>.

[4] *Jak psát web* [online]. 2011, [cit. 2011-03-22]. Úvod do platformy Adobe Flash. Dostupné z WWW: <http://flash.jakpsatweb.cz/adobe-flash/>.

[5] KOMÁRKOVÁ, Jitka; KOPÁČKOVÁ, Hana. Geografické informační systémy. Pardubic: Univerzita Pardubice, 2008. 55 s. ISBN 978-80-7395-120-7.

[6] *MySQL* [online]. c2011, [cit. 2011-03-19]. The MyISAM Storage Engine. Dostupné z WWW: <http://dev.mysql.com/doc/refman/5.5/en/myisam-storage-engine.html>

[7] MySQL [online]. c2011, [cit. 2011-03-19]. The InnoDB Storage Engine. Dostupné z WWW: <http://dev.mysql.com/doc/refman/5.0/en/innodb-storage-engine.html>.

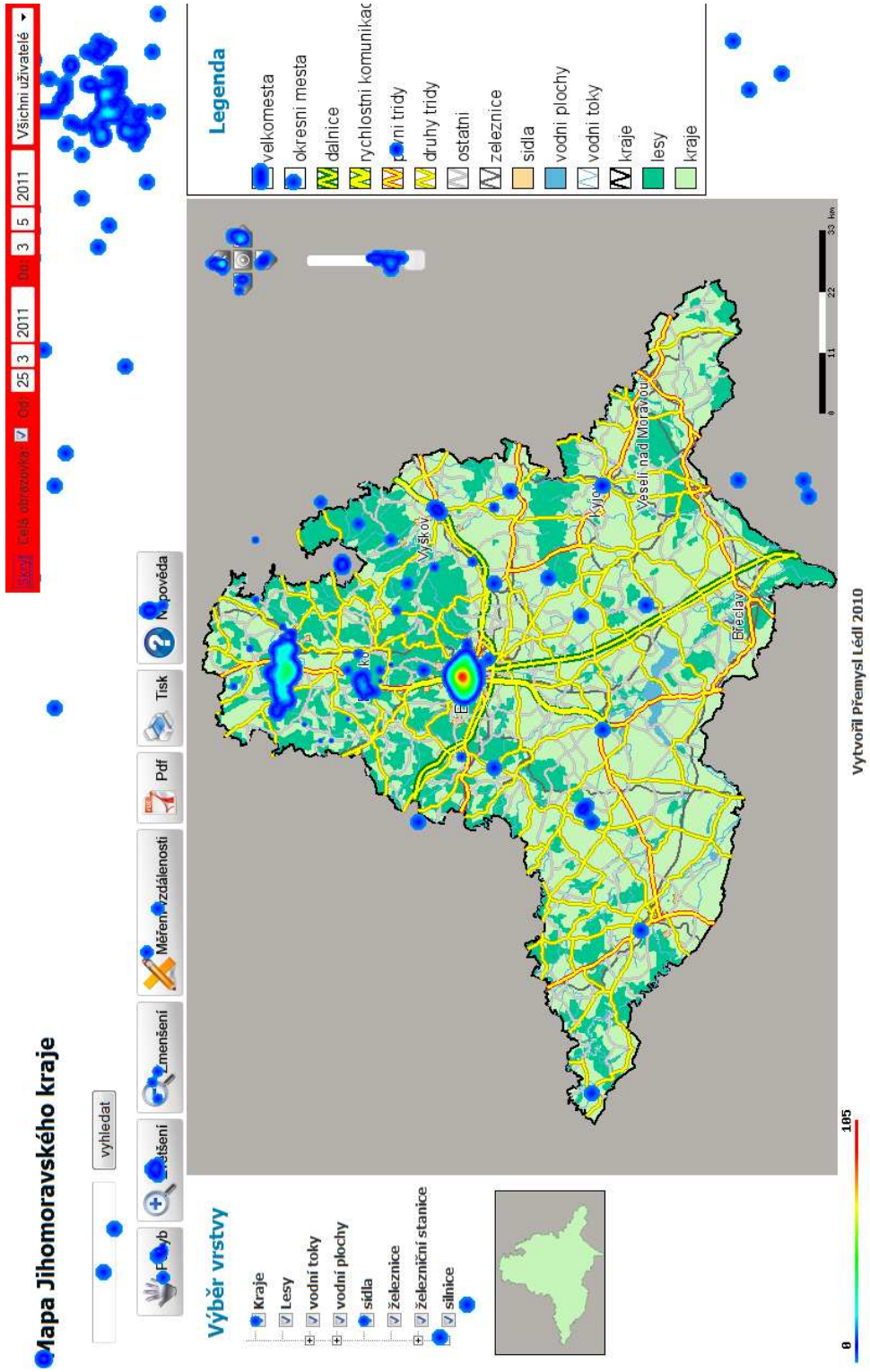
[8] PETERKA, Jiří. Klient/server na různé způsoby. e*Archiv.cz: archiv článků a přednášek Jiřího Peterky* [online]. 12. 3. 1996, [cit. 2011-02-10]. Dostupný z WWW: <http://www.earchiv.cz/a96/a611k150.php3>.

[9] STANÍČEK, Petr. CSS kaskádové styly. Praha: Computer Press, 2003. 178 s. ISBN 80-7226-872-4.

[10] VRÁNA, Jakub. PHP triky [online]. 2008-5-16, [cit. 2011-04-17]. Rychlost vkládání do InnoDB tabulek. Dostupné z WWW: <http://php.vrana.cz/rychlost-vkladani-do-innodb-tabulek.php>.

[11] WILKINSON, Leland; FRIENDLY, Michael. The History of the Cluster Heat Map [online]. 18. 1. 2008, [cit. 2011-02-14]. Dostupný z WWW: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.165.4766&rep=rep1&type=pdf>.

Příloha 1 – Náhled vygenerované heatmapy



Příloha 2 – Zdrojový kód aplikace pro generování heatmapy

```
<?php header("Content-Type: text/xml; charset=utf-8"); ?>
<image>
<?php
  $scriptstart = time();
  require_once("conf.php");
  mysql_connect($server,$user,$pass) or die(mysql_error());
  mysql_select_db($database) or die(mysql_error());
  mysql_query("SET CHARACTER SET utf8");     // pripojeni bude v UTF-8

  $point = array();
  $colors = array();

  $unfinished = 0; // zdali byly vygenerovany vsechny body z databaze

  $imagename = "";
  $mappar =
array('kraje','lesy','toky','vodpl','toky_nazev','vodpl_nazev','sidla','zeleznice','silnice','dalnice','rychl
o','prvni','druhy','ostatni','zel_stan','zel_stan_nazev');
  $mapdata = "";
  $mapquery = "";
  $mapname = "";
  foreach($mappar AS $k => $v){
    if(isset($_GET[$v])) {
       $val = 1;
    } else $val = 0;

    $mapquery .= " AND `".$v."` = ".$val."";
    $mapname  .= $val;
  }

  $starttime       = getArrayData($_GET,"starttime",0);
  $endtime         = getArrayData($_GET,"endtime",getEndTime());
  $map_width       = getArrayData($_GET,"mpw",map_width);
  $map_height      = getArrayData($_GET,"mph",map_height);
  $fullpage        = getArrayData($_GET,"fp",0);  // celostrankova mapa
  $rid             = getArrayData($_GET,"rid",0); // resource ID
  $uid             = getArrayData($_GET,"uid",0); // resource ID

  $userquery = "";
  if($uid){
    $userquery .= " AND `uid` = '".$uid."'";
    $mapname   .= $uid;
  } else {}


// pozice a rozmery mapy v GIS aplikaci
  $gismap_x = getArrayData($_GET,"mpx",gismap_x);
  $gismap_y = getArrayData($_GET,"mpy",gismap_y);
  $gismap_w = getArrayData($_GET,"mpw",map_width);
  $gismap_h = getArrayData($_GET,"mph",map_height);

  $screen_w = getArrayData($_GET,"scrw",screen_width);
```

```
  $screen_h = getArrayData($_GET,"scrh",screen_height);

  $posun_x = 0;  // posun mapy v ose x
  $posun_y = 0;  // posun mapy v ose y

  $img = "";
  $limitstart = 1;
  $points = array( );  // pole s body
  $hm = array( );
  $imagename ="";
  $parametr1 = 1;
  $parametr2 = 1;
  $extent = getArrayData($_GET,"extent",defaulextent); // extent
  $mapmin = 100;
  $mapmax = -100;
  $qi = 0;
  $savestatus = 0;
  $norma = 1;

  // dodelani obrazku z cache
    if($rid){
    $f = prgcachedir.$rid.".php";
    if(file_exists($f)){
      include($f);
    }

    if(!empty($imagesource)){
       $img = imagecreatefrompng(imagecachedir.$imagesource);
    }

    if(!empty($img)){
      $white = imagecolorallocatealpha($img, 255, 255, 255,127);
      $black = imagecolorallocatealpha($img, 0, 0, 0,127);
    }

  // novy obrazek
  }else {
    $status = status_start;


if($fullpage){
       $map_width       = $screen_w;
       $map_height      = $screen_h;
       $posun_x = $gismap_x;  // posun mapy v ose x
       $posun_y = $gismap_y;  // posun mapy v ose y
       $status = status_fppointsunfinished;
    }

    //$hm = createHeatmapArray($map_width,$map_height);
  }

  if(empty($imagename)) $imagename =
sha1($starttime.$endtime.$map_width.$map_height.$extent.$mapname).".png";

  //       MAP CACHE
  if((mapcache > 0) && file_exists(imagecachedir.$imagename) && ($status >= status_start)){
       $status = status_done;
  }
```

```
if($status > status_done) {
  if(empty($img)){
    $img = imagecreatetruecolor($map_width,$map_height); // vytvoreni obrazku mapy

    $white = imagecolorallocatealpha($img, 255, 255, 255,127); // vytvoreni pruhledne barvy
    $black = imagecolorallocatealpha($img, 0, 0, 0,127); // vytvoreni pruhledne barvy
    imagefill($img,0,0,$white); // vyplneni obrazku pruhlednou barvou
  }

  imagealphablending($img, false); // povoleni pruhlednosti
  imagesavealpha($img,true);    // povoleni ulozeni alfa kanalu


  // vypocet souradnic extent a pomer v pixelech
  $extents = explode(" ",$extent);
  $ex_minx = floor($extents[0]);
  $ex_miny = floor($extents[2]);
  $ex_maxx = ceil($extents[5]);
  $ex_maxy = ceil($extents[8]);

  $pomerx = $gismap_w / ($ex_maxx - $ex_minx);
  $pomery = $gismap_h / ($ex_maxy - $ex_miny);

  //   Barevna skala
  for($i=0;$i<=100;$i++){
    $alpha = 0;
    if($i<=25){
      $colors[100-$i] = imagecolorallocatealpha($img, 255, ($i*10), 0,$alpha);
    } else if($i<=50){
      $k = $i-25;
      $colors[100-$i] = imagecolorallocatealpha($img, 255-($k*10), 255,0,$alpha);
    } else if($i<=75){
      $k = $i-50;
      $colors[100-$i] = imagecolorallocatealpha($img, 0, 255, ($k*10),$alpha);
    } else if($i<=90){
      $k = $i-75;
      $colors[100-$i] = imagecolorallocatealpha($img, 0, 255-($k*17), 255,$alpha);
    } else if($i<=100){
      $k = $i-90;
      $colors[100-$i] = imagecolorallocatealpha($img, 0, ($k*20), 255,$alpha);
    }
  }
  for($i=0;$i<=2;$i++) $colors[$i] = $white;

}

//   Body z databaze CELOSTRANKOVE
  if($status >= status_fppointsunfinished){
  $status = status_pointsunfinished;

  $q  = "SELECT * FROM `mapdata` WHERE `type` = 0 AND `mapex` = 0 AND `mapey` = 0
AND date > ".$starttime." AND date < ".$endtime." ".$userquery;
  $result = mysql_query($q);
// echo("<qf>'".$q."'</qf>");
  $qi = 1;
  if($result){
```

```
        mysql_field_seek($result,$parametr1-1);
        while($row = mysql_fetch_array($result)){

          if((time() - $scriptstart) > maxgeneratingtime) {
            saveCache(status_pointsunfinished,"",$parametr1+$qi);
            $status = status_break;
            break;
          }

          $x = abs(round($row["x"]));
          $y = abs(round($row["y"]));

          // prepocet na hustotu
          $x = $x - ($x % (density));
          $y = $y - ($y % (density));

          $state = $row["state"];
          if($state <0 ) $state = 0;
          if(($x > 0) && ($y > 0) && ($x < $map_width) && ($y < $map_height)) {
            if(!isset($points[$x][$y])) $points[$x][$y] = array(10,pointsize-(($state-1)*4));
            else $points[$x][$y] = array($points[$x][$y][0] + 1,pointsize-(($state-1)*4));
          }

          $ph = $points[$x][$y];

          $qi++;
        }
      }
      resetParametrs();
    }


    //   Body z databaze
    if($status >= status_pointsunfinished){
      //$status = status_mapunfinished;
      $status = status_mapunfinished;

      //sql dotaz na body
      $q  = "SELECT * FROM `mapdata` WHERE `type` = 0 AND `mapex` >= ".$ex_minx." AND
`mapex` <= ".$ex_maxx." AND `mapey` >= ".$ex_miny." AND `mapey` <= ".$ex_maxy." AND
date > ".$starttime." AND date < ".$endtime." ".$userquery;
      $result = mysql_query($q);
      $qi = 1;
      if($result){
        mysql_field_seek($result,$parametr1-1);
        while($row = mysql_fetch_array($result)){
         //  echo("<sstatus".$qi.$qi.">".$status."</sstatus".$qi.$qi.">");
          if((time() - $scriptstart) > maxgeneratingtime) {
            saveCache(status_pointsunfinished,"",$parametr1+$qi);
            $status = status_break;
            break;
          }

          $x = $posun_x + abs(round(($row["mapex"] - $ex_minx) * $pomerx));
          $y = $posun_y + abs(round(($row["mapey"] - $ex_maxy) * $pomery));

          // prepocet na hustotu
          $x = $x - ($x % (density));
```

```
        $y = $y - ($y % (density));

        $state = $row["state"];
        if($state <0 ) $state = 0;
        if(($x > 0) && ($y > 0) && ($x < $map_width) && ($y < $map_height)) {
          if(!isset($points[$x][$y])) $points[$x][$y] = array(10,pointsize-(($state-1)*4));
          else $points[$x][$y] = array($points[$x][$y][0] + 1,pointsize-(($state-1)*4));
        }
        $qi++;
      }
    }
    resetParametrs();
}

//   Vytvoreni mapy
if($status >= status_mapunfinished){
    $status = status_imageunfinished;
    // zapsani bodu do pole

    if(isset($points[$parametr1][$parametr2])){
     $p =  $points[$parametr1][$parametr2];
    }
    else $p = reset($points);

    while($p){
      $pk = key($points);
      $r = reset($p);

      while($r){
        $rk = key($p);

        if((time() - $scriptstart) > maxgeneratingtime) {

          saveCache(status_mapunfinished,"",$pk,$rk);
          $status = status_break;
          break;
        }
        createPointInImage($img,$r[0],$pk,$rk,$map_width,$map_height,$r[2]);

        $r = next($p);
      }
      $p = next($points);
    }
    resetParametrs();
}


if($mapmax > colormax) $norma = (colormax / $mapmax);
else $norma = colormax / 100;
echo("<mapmax>".$mapmax."</mapmax>");

//   Vykresleni bodu do obrazku
if($status >= status_imageunfinished){
    $status = status_done;
    createColorHeatmap($img,$map_width,$map_height);
    imageSetRange($img,$map_width,$map_height);
    imagepng($img,imagecachedir.$imagename); // vypsani obrazku na vystup
}
```

```
if($status){
  echo("<status>".$status."</status>");
}else{
  echo("<status>".$savestatus."</status>");
}

// v poradku jsme vygenerovali
if($status == status_done){
  echo("<src>".MS_URL.imagecachedir.$imagename."</src>");

  // smazani cache
  if(!empty($rid) && (is_file(prgcachedir.$rid.".php"))) unlink(prgcachedir.$rid.".php");

// nekde nastal probelem
} else {
  echo("<rid>".$rid."</rid>");
}

// nastavy dane parametry na default
function resetParametrs(){
  global $parametr1, $parametr2;

  $parametr1 = 1;
  $parametr2 = 1;

}

// vytvori prazdnou heatmapu
// @x : sirka heatmapy [int]
// @y : vyska heatmapy  [int]
function createHeatmapArray($x,$y=0){

  if($y <= 0) $y = $x; // pokud mam ctvercovou mapu

  $hm = array();

  for($i=1; $i<=$x;$i++){
    for($j=1; $j<=$y;$j++){
      $hm[$i][$j] = 0;
    }
  }
  return $hm;
}


// vytvori ze zadane heatmapy (pole) obrazek
 // @img : obrazek [image]
// @x : sirka heatmapy [int]
// @y : vyska heatmapy  [int]
function createColorHeatmap(&$img,$x,$y=0){

  global $status, $colors, $white, $scriptstart, $parametr1, $parametr2,$norma,  $white;
  if($y <= 0) $y = $x;

  for($i=($parametr1); $i < $x;$i++){
    for($j=($parametr2); $j < $y;$j++){
```

```
        if((time() - $scriptstart) > maxgeneratingtime) {

          global $imagename;
          saveCache(status_imageunfinished,$imagename,$i,$j);
          $status = status_break;
          return $hm;
        }

        $h = imagecolorat($img,$i,$j);
        if($h != $white){
          $c = isset($colors[round($h*$norma)])?$colors[round($h*$norma)]:$white;
          imagesetpixel($img,$i,$j,$c);
        }
      }
      $parametr2 = 1;
    }
  }

// vytvori v zadane heatmape (image) na zadanych souradnicich (x,y) bod a prislusne rozmazani
// @img : obrazek [image]
// @strong : vyska/sila bodu [int]
// @x : souradnice bodu [int]
// @y : souradnice bodu  [int]
// @w : sirka heatmapy [int]
// @h : vyska heatmapy  [int]
// @size : sirka rozmazani [int]
function createPointInImage(&$img,$strong,$x,$y,$w,$h,$size=pointsize){
  global $colors,$white,$norma,$mapmax,$mapmin;

  $center = floor($size / 2);
  if($strong > 0){
    $nmax = cparam*($strong/colormax)*sqrt(pow($center,2)+pow($center,2));
    $par = ($strong/$nmax); // parametr dorovnava vytvorenou alphu na zadanou silu (strong)

      for ($i = 1; $i < $size; $i++){
        if(($i+$x-$center)<=0) continue;
        if(($i+$x-$center)>$w) continue;
          for ($j = 1; $j < $size; $j++){
              if(($j+$y-$center)<=0) continue;
              if(($j+$y-$center)>$h) continue;

        $norma = $par*($strong/colormax)*sqrt(pow($i-$center,2)+pow($j-$center,2));
        $alpha = round($strong - $norma);
        $shmv = imagecolorat($img,$i+$x-$center,$j+$y-$center);
        if($shmv == $white) $shmv = 0;
        $hmmin = max(min($shmv,$alpha),0);
        $hmmax = min(max($shmv,$alpha),colormax);
        $alpha = abs(ceil($hmmax) + ceil($hmmin));
        if($alpha > $mapmax) $mapmax = $alpha;
        if($alpha < $mapmin) $mapmin = $alpha;

        imagesetpixel($img,$i+$x-$center,$j+$y-$center,$alpha);
          }
        }
     } else imagesetpixel($img,$x,$y,0);
}
```

```
// vrati data z pole na danem indexu
// pokud neni pole, nebo index neexistuje, vraci default
// @arr    - pole v nemz se hleda index []
// @index  - index ktery se ma v poli najit [string/int]
// @defaul - defaultni navratova hodnota (pokud neni pole/ index neexistuje / ..)
function getArrayData(&$arr,$index,$default = 0){
  if(is_array($arr)){
    if(isset($arr[$index])){
      return $arr[$index];
    }
    else return $default;
  } else return $default;
}

// vraci casovou znacku podle nastaveni cache obrazku
function getEndTime(){
  $t = time();
  if(mapcache > 0) {
    $tt = $t % mapcache;
    return (($t-$tt)+mapcache);
  } else return $t;
}


// vytvolri textovou PHP reprezentaci pole
// @arr     - pole ktere se prevede na retezec [array]
// @indexes - ulozi se i indexy [int]
function buildArray(&$arr, $indexes = 1){
  $s = "array(";

  if(!is_array($arr)) return $arr;

  if($indexes>0){
     foreach($arr as $k => $v){
       if(is_array($v)) {
         $s .= $k."=>".buildArray($v).",";
       } else {
         $s .= $k."=>".$v.",";
       }
     }
  } else {
    foreach($arr as $k => $v){
      if(is_array($v)) {
        $s .= buildArray($v,0).",";
      } else {
        $s .= $v.",";
      }
    }
  }

  $s.= ")";

  return $s;

}
```

```
 // vepise do dane mapy barevnou skalu
  // @img : obrazek [image]
  // @w : sirka heatmapy [int]
  // @h : vyska heatmapy  [int]
  function imageSetRange($img,$w,$h){
    global $mapmax,$mapmin,$colors;

    if($mapmax <= 0 ) return 0;

    if($mapmax >= 100) $mpmax = 100; else $mpmax = $mapmax;
    $interval = 200 / $mpmax;

    $black = imagecolorallocate($img, 0, 0, 0); // vytvoreni cerne

    imagestring( $img , 3 , 20, $h-35, $mapmin , $black );
    imagestring( $img , 3 , 220, $h-35, $mapmax , $black );

    for($i=1;$i<=$mpmax;$i++){
      imagefilledrectangle($img, 30+($i*$interval) , $h-20 , (($i+1)*$interval)-1 , $h-18 ,
$colors[$i] );
    }

  }

  // nacte data z db a ulozi je do promennych
  // @status - status generovani mapy [integer]
  // @imagesource - cesta k souboru s obrazkme [string]
  // @p1   - parametr 1 [integer]
  // @p2   - parametr 2 [integer]
  function saveCache($status,$imagesource,$p1=0,$p2=0){

    global $points, $hm, $rid,$starttime,$endtime,$extent,$mapmin,$mapmax,$qi;
    global $savestatus,$imagename,$posun_x,$posun_y,$map_width,$map_height;
    global $scriptstart;


    if(empty($rid)) $rid = basename($imagename,".png");

    $hmfn = prgcachedir.basename($imagename,".png").".txt";

    if($f=fopen(prgcachedir.$rid.".php",'w')){
      fwrite($f, "<?php\n");

      fwrite($f, "\$imagesource = '".$imagesource."';\n");
      fwrite($f, "\$imagename = '".$imagename."';\n");
      fwrite($f, "\$status = '".$status."';\n");
      fwrite($f, "\$parametr1 = '".$p1."';\n");
      fwrite($f, "\$parametr2 = '".$p2."';\n");
      fwrite($f, "\$extent = '".$extent."';\n");

      fwrite($f, "\$map_width = '".$map_width."';\n");
      fwrite($f, "\$map_height = '".$map_height."';\n");

      fwrite($f, "\$posun_x = '".$posun_x."';\n");
      fwrite($f, "\$posun_y = '".$posun_y."';\n");

      fwrite($f, "\$mapmin = '".$mapmin."';\n");
      fwrite($f, "\$mapmax = '".$mapmax."';\n");
```

```
        fwrite($f, "\$qi = '".$qi."';\n");

        fwrite($f, "\$points = ".buildArray($points).";\n");

       // fwrite($f, "foreach(\$hm as \$k => \$v) \$hm[\$k] = explode(\";\",\$v);\n");

        fwrite($f, "?>\n");
        fclose($f);

      } else {
        $rid = 0;
      }
    }
?>
</image>
```