JIHOČESKÁ UNIVERZITA V ČESKÝCH BUDĚJOVICÍCH
PEDAGOGICKÁ FAKULTA

KATEDRA FYZIKY

Bakalářská práce

Autor: Jiří Novák

Vedoucí práce: Ing. Michal Šerý

2008

Open source a Mapový server

Open Source and Map Server

Prohlášení:

Prohlašuji, že jsem tuto práci vypracoval samostatně a že jsem všechny použité zdroje uvedl v Seznamu použité literatury na konci této práce. Zároveň povoluji Katedře fyziky PF JU v Č. Budějovicích libovolné využití této práce.

V Českých Budějovicích dne 25. dubna 2008 Novák Jiří

Obsah

Úvod

Open source a mapové servery mají v dnešním světě a každodenním používání informačních technologií a výpočetní techniky a Internetu své místo a zaslouží si být presentovány nejen odborníkům, ale i širší veřejnosti. A jakým lepším způsobem, než pomocí sítě sítí, Internetu.

V této práci se zabývám nejen Open source a mapovými servery, tedy jejich obecnějším vysvětlením a představením, ale hlavně možnostmi presentovat nástin vzniku mapových serverů.

Celá práce je rozdělena do několika kapitol. V první z nich se, kvůli uvedení do problematiky, zabývám vysvětlením Open source softwaru. Tato kapitola krom základních informací obsahuje také další popis týkající se již zmiňovaného Open source.

V další kapitole se již zabývám vysvětlením a přiblížením problematiky Geografických Informačních Systémů, jako co je to GIS, jaké jsou součásti GISu atd. Práce pokračuje nástinem problematiky a vývoje mapových serverů, od úvodu přes principy fungování, instalaci, adresářovou strukturu až po data a první spouštění mapového serveru a nástin jeho projekce.

1. Open source software

Open source nebo také open-source software (dále pak OSS) je počítačový software s otevřeným zdrojovým kódem. Otevřenost zde znamená jak technickou dostupnost kódu, tak legální dostupnost - licenci software, která umožňuje, při dodržení jistých podmínek, uživatelům zdrojový kód využívat, například prohlížet a upravovat.

V užším smyslu se OSS míní software s licencí vyhovující definici prosazované Open Source Initiative. Pro odlišení se někdy open source software vyhovující požadavkům OSI označuje Open Source (s velkými písmeny).

V nepřesném ale poměrně běžném vyjadřování se označení *open source* používá i pro mnoho vlastností, které s otevřeností zdrojového kódu nesouvisí, ale vyskytují se u mnoha open source programů. Například může jít o bezplatnou dostupnost software, vývoj zajišťovaný úplně nebo z podstatné části dobrovolnickou komunitou nebo „nekomerčnost“.

Souvisejícím tématem je svobodný software (free software) - tento pojem prosazuje Free Software Foundation pro podmnožinu open source software dostupnou pod svobodnou licencí, která musí oproti Open Source licenci splňovat ještě další podmínky, například musí umožňovat uživatelům šířit díla odvozená z původního programu.

1. 1. Srovnání některých vlastností open source

1. 1. 1. Bezpečnost

Z hlediska bezpečnostních děr v software je otevřenost kódu dvojsečná zbraň. Chyby v programech může hledat mnohem širší skupina lidí (nebo i automatických pomůcek) a je proto naděje, že se snáze opraví. Na druhou stranu zranitelnosti mohou snáze najít i útočníci. V současném paradigmatu informační bezpečnosti full disclosure se ovšem považuje za obecně výhodnější, když jsou informace dostupné všem, i za tu cenu, že jsou dostupné útočníkům. Alespoň u populárních programů s velkou základnou uživatelů a vývojářů lze předpokládat, že „uživatelská“ strana má výrazně větší prostředky (především více času kvalifikovaných lidí) než cracker.

Nespornou výhodou otevřeného zdrojového kódu je ohromné ztížení možnosti propašování zadních vrátek a trojských koní.

1. 1. 2. Známý open source software

- operační systémy GNU/Linux, FreeBSD, FreeDOS, ReactOS, Sun Solaris
- databázové servery MySQL, Firebird, PostgreSQL
- webserver Apache
- skriptovací jazyk PHP Hypertext Preprocessor
- monitorovací software nmap, nagios
- kancelářský software OpenOffice.org
- Internetový prohlížeč Mozilla Firefox
- bitmapový editor GIMP
- 3D grafický editor Blender
- video přehrávač VLC media player
- DOSový audio přehrávač MPXPLAY
- překladače jazyka „C“ (včetně „C++“ a některých dalších) GCC a OpenWATCOM
- překladač FreeBASIC
- podcatcher (program pro stahování multimediálních souborů, podcastů) Juice
- archivační a kompresní program 7-Zip
- hra Frozen Bubble
- komunikační software Miranda IM
- editor myšlenkových map FreeMind
- IDE pro vývoj aplikací Eclipse
- prostředí pro vývoj ERP, CRM, SCM aplikací JFire

1. 1. 3. A co že je to OpenSource?

OpenSource zní jako zaklínadlo napříč všemi IT sektami. Vyslovují ho ti, kteří za ním vidí spásu a mluví o něm i ti, kteří v něm vidí záhubu.

Možná, že existují i lidé, kteří pořád ještě tak docela nevědí, co to ten OpenSource je. Anglické slovo Source v tomto významu znamená text programu, napsaný v nějakém jazyce, kterému rozumí člověk, nikoliv však počítač. Pokud je

Source, lze program upravit, ti inteligentnější z něho dokážou i zjistit, jak program funguje. Aby však bylo možno tento program spustit, musíte ho tzv. zkompilovat. Lidsky řečeno to znamená, že z původního „source“ vznikne naprosto nečitelná změť různých znaků a čísel, kterým už nerozumí člověk, zato však mu rozumí počítač.

To je ta forma, kterou dostanete jako uživatel na CD. A protože každý číslíčko v této formě programu závisí na přesné poloze všech ostatních, nelze program nijak upravit, protože i když význam těch číslíček znáte, nemůžete je libovolně měnit.

Jediným rozumným způsobem, jak program změnit, je tedy udělat změnu v jeho Source (tedy v tom původním lidsky čitelném textu). Z toho vyplývá, že kdo vlastní Source, může si s takovouto aplikací dělat co se mu zlíbí. Např. tam všude dopsat JiritU, přeházet pár tlačítek a vydávat aplikaci za svůj výtvor (ano, skutečně takový případy jsou). Proto si každá firma svůj Source brání, protože u softwarových firem je to v podstatě jediná hodnota, kterou mají. Ve chvíli, kdy by jej někdo odcizil a zveřejnil, původní firmě už nikdo nedá téměř ani korunu, každý si totiž může aplikaci změnit tak, aby v ní nikdo tu původní nepoznal, sám si jí může zkompilovat a používat (pravda je taková, že pro osobní použití je jednodušší ukrást již hotovou aplikaci).

Z pohledu firmy tvořící SW je tedy „tajný“ Source způsobem, jak z něho mít peníze. Potřebujete novou funkci ? Napište výrobci. Výrobce jí doplní do Source, zkompiluje a prodá vám novou verzi. Když budete mít Source vy a budete umět programovat, doplníte si funkci sám a výrobce utře. Z pohledu většiny zákazníků je jim jedno, jestli mají nebo nemají Source. Problém nastane ve chvíli, kdy aplikace obsahuje chybu, která je pro zákazníka kritická. Mnohdy musí čekat několik měsíců, než výrobce vadu odstraní, ačkoliv kdyby sám měl Source k dispozici, oprava chyby by zabrala třeba jen několik minut.Ještě horší je, že firma, která si koupí např. obchodní systém je na svém dodavateli programu závislá, protože když původní SW dodavatel zkrachuje, nebude nikdo, kdo by obchodní systém dál vyvíjel, nebo alespoň udržoval ve funkčnosti. Koupím si SW, který poběží např. na Windows95. Firma, která mi jej dodala, zkrachuje. Mně se nic nestalo, na aplikaci to nemá žádný vliv, té je to jedno, běží. Jenže za 3 roky přijde Windows98. Moje koupená aplikace na Windows98 neběží. Pokud si koupím nový počítač, nebudu na něm už mít Windows95 (ty se už neprodávají), ale Windows98.

To znamená, že na něm svou zaplacenou aplikaci nemůžu nainstalovat, ani používat. Technologický pokrok mojí firmy se tedy s pádem dodavatele SW zastavil. Nemůžu dělat nic. Můžu si koupit za 2-3 milióny nový SW od jiné firmy, za ½ miliónu jej dát místo původního, přeškolit zaměstnance, převést data. Rázem jsem se ocitl v červených číslech. A to celé se může opět opakovat. Osud mé firmy je opět v rukou cizí firmy, jejíž chod nemohu nijak ovlivnit.

1. 1. 4. A nebo si můžu vybrat OpenSource

Anglické Open zde znamená „otevřený, přístupný“. Znamená to, že zmiňovaný Source, je volně k dispozici. Každému. Navíc každý programátor může přidávat své vlastní nápady a aplikaci inovovat. Může si Source stáhnout, upravit pro potřebu své firmy. Když se vyskytne chyba, můžu si jí ihned sám opravit a dát opravu k dispozici zpět ostatním. Problém je, že původní tvůrce aplikace nedostane nejspíš ani korunu.

Z tohoto vznikla zakořeněná fáma, že OpenSource = zadarmo. Teoreticky tomu sice tak není, ve skutečnosti si málokterá firma „lajzne“ dát k dispozici source a chtít po lidech peníze. Ano, možné to je, ale není to moc efektivní. Jenomže i vývoj OpenSource vyžaduje peníze.

S přibývající složitostí projektu už to přestává být možné. Firma si většinou nebude na blind živit programátora, kterej bude studovat pět miliónů řádek programu, aby byl připraven je v případě nutnosti doplnit, nebo opravit. Takže bude pro firmu jednodušší, když v případě potřeby kontaktuje dotyčné, kteří mají aplikaci na starosti (bývá to skupinka těch nejznalejších, kteří na programu pracují). Jejich "guruovi" sdělí, že potřebují, aby jim ten software v kanceláři taky reguloval teplotu.

Guru odpoví - je to možné, chceme za to 20 tisíc. Protože pro firmu je výhodnější zaplatit 20 tisíc (jeden programátorův měsíční plat) a mít úpravy rychle a dle potřeby, než živit programátora.

Ale co když i tito „správci aplikace“ ztratí motivaci na programu pracovat ? V čem je výhoda, když jsem stejně závislý, akorát mám k dispozici Source, kterému stejně nerozumím?

Jednak v tom, že za určitý finanční obnos se vždycky najde někdo, kdo za peníze Source prozkoumá do té míry, aby jej dokázal inovovat (a vyjde to levněji, než nákup jiného softwaru a s tím spojená dato-zaměstnanecká transformace). Další věc je ta, že tak, jako zde na pooh.cz přibývají a ubývají „grafomani“,stejným způsobem přicházejí a odcházejí lidé, kteří se o původní projekt starají. Za 5 let už z původních tvůrců nemusí nikdo žít, ale projekt žije dál, protože je pořád někdo, kdo ho používá a kdo se o něj jednak ve vlastním a jednak v komerčním zájmu stará. Pravděpodobnost úmrtí projektu, který je starší 2 let je téměř nulová.

Poslední věcí, kterou je OpenSource výhodný, už není pro uživatele patrná na první pohled - je to způsob tvorby. Projekt nevzniká tak, že se několik lidí uzamkne v místnosti bez oken a tam tvoří. Do projektu totiž vidí „každý kdo má zájem“ a jakákoliv neschopnost je velmi rychle odstraněna. V životě totiž všichni, kdož něco tvoří, musí nést za své jednání následky (i přestože to dělá zdarma). Být členem správy projektu je vetšinou výsadou, nikoliv právem, a lidem v kolektivech kterří chtějí dosáhnout výsledků se moc nedaří. Takže u tvůrců OpenSource projektu je větší jistota, že se jedná o odborníky, než u klasického „tajného“ Source, kde člověk nikdy neví, že na něm poslední půlrok pracoval nějaký bratranec paní účetní, která ho k tomuto prosadila a který mimochodem o programování neví zhola nic.

OpenSource je způsob, jak co nejvíce minimalizovat rizika závislosti uživatele na používaném softwaru, umožňuje uživatelům do vývoje více "kecat" i aktivně se zapojit. Na druhou stranu vyžaduje odlišný obchodní model, kde projekt většinou financuje úzká skupina firem, která buď požaduje po tvůrcích úpravy na míru, nebo do projektu investuje tak, že platí za školení zaměstnanců, popř. si může investor předplatit asistenční služby atd.

Mezi známé OpenSource projekty patří i např. Firebird, ve kterém náš malý státeček tvoří roli nikoliv nevýznamnou, nebo taktéž česká iniciativa sdružení projektů pro programování v Delphi nazvaná Delphree, a další. Nepsaným "skladištěm" OpenSource je pak Sourceforge kde je několik tisíc aktivních projektů.

Server je v informatice obecné označení pro počítač (tedy hardware) nebo software, který poskytuje nějakou službu dalším počítačům nebo programům. Ve světě Unixu se softwarový server nazývá „démon“ (anglicky „daemon“), v Microsoft Windows se nazývá „služba“ (anglicky „service“). Služba může být nabízena v rámci jednoho počítače (obsluha připojené tiskárny, správa automatických aktualizací, apod.) nebo i počítačové sítě (sdílené disky, síťová tiskárna, WWW server, autentizační server, a další). Poskytování služby je pak realizováno pomocí aplikačního síťového protokolu, například HTTP pro webový server, LPT pro tiskový server, SMB pro sdílení disků a tiskáren ve Windows.

Počítač nebo proces, který službu využívá, se nazývá klient, architektura, která používá tento princip, se nazývá klient-server.

Obr. 1. 1. Server rack

Obr. 1. 2. Stejný server rack vyfocený zezadu

Servery jsou dnes tam, kde jsou potřeba služby pro ostatní PC v síti. Většina serverů je vyhrazena pro tento účel, ale mohou být využívány i k jiným účelům. Například v malé kanceláři, výkonný desktop PC může sloužit pro obojí najednou. Jako stanice pro uživatele a zároveň jako server pro skupinu dalších počítačů. Termín „Server“ bylo převzato ze slova „Serve (sloužit)“, proto slouží celé síti k vykonávání tisku několika uživatelů, nebo jako souborový server pro aplikace nebo data, samozřejmě může pro uživatele zprostředkovávat i mnoho dalších služeb.

1 . 1. 5. Hardware serveru

Server může být postaven z běžného hardwaru, který se používá u stolních počítačů, ale mnohem častěji se používá specializovaný hardware optimalizovaný pro potřeby serveru. Rozdíl je zejména ve spolehlivosti. Servery většinou běží 24 hodin denně 365 dní v roce zapnuté, tudíž musí mít větší životnost. Větší životnost a spolehlivost souvisí také s tím, že při výpadku serveru může vzniknout společnosti, která ho používá, velká finanční ztráta. To má za následek mnohem vyšší pořizovací náklady.

Dnešní výrobci serverů (například DELL, INTEL, …) používají běžný hardware, který si sami otestují a zaručují spolehlivost. Ke svým serverům dále poskytují servis, kdy podle uzavřené smlouvy musí do určité doby (př. do 24. hodin) odstranit hardwarovou závadu. Problém ale bývá v tom, že mohou používat některé speciální komponenty a pokud výrobci přestanou za této situace poskytovat technickou podporu (zejména pokud jsou už staré), tak může být velký problém s jejich opravou nebo rozšířením.

Na vzhled serverů se nehledí, většinou jsou zavřeny v osamocené místnosti, nainstalovány do racku a jsou navštěvovány pouze za účelem údržby nebo oprav. Servery jsou nejčastěji umístěny v místnosti bez oken, s omezeným přístupem a s klimatizací, která drží stálou teplotu v místnosti okolo 20°C.

Procesory (CPU) jsou jednou z nejdůležitějších hardwarových částí serverů. Musí řešit spoustu dotazů od uživatelů a je zapotřebí, aby měli co nejmenší chybovost. V serverech je často používáno více procesorů, dnes místo více procesorů se čím dál častěji používají procesory s více jádry. Mezi nejznámější serverové procesory patří Xeon od firmy Intel a Opteron od firmy AMD. Servery většinou obsahují obyčejnou integrovanou grafickou kartu (často se do nejnovějších serverových základních desek montují grafické karty, které už existují mnoho let). Často nemají zvukovou kartu, rozhraní pro joystick, USB atd. Taková zařízení jsou pro chod serveru zcela zbytečná.

Serverem se však dá nazývat i "domácí server", který je založen (postaven) na stolní (domácí) platformě (PC). Tyto servery se používají převážně jako ftp a databázové servery. Jediné čím se od standardních PC odlišují např.: větší počet

výkonných LAN síťových karet, větším množstvím HDD a výkonnějším CPU, zároveň musí být kvalitně chlazeny.

1. 1. 6. Operační systém serveru

Hlavní rozdíl mezi desktop PC a serverem je v softwaru. Na většině serverů běží operační systém, který je navrhován speciálně jako serverový systém.

Microsoft Windows převládá na poli operačních systému určených pro desktop, ale co se týče serverových systémů jsou daleko populárnější systémy jako FreeBSD, Solaris a GNU/Linux – tyto systémy jsou odvozené nebo jsou součástí Unixových operačních systémů. Unix byl původně operační systém pro minipočítače, ale jak minipočítače byly nahrazovány servery, tak logicky i Unix byl přepsán pro využití na serverech.

1. 1. 7. Servery v Internetu

Skoro veškerá struktura v internetu je schovaná za klientským serverem. Mnoho miliónů serverů po celém světě je propojeno mezi sebou a tím tvoří celý Internet.

Mezi mnoho služeb poskytující Internetové servery patří: Web, Domain Name System (DNS), elektronická pošta, přenos souborů, audio a video, on-line hry a mnoho dalších. Doslova každá akce prováděná uživatelem internetu očekává jednu nebo více protiakcí od jednoho nebo více serverů.

1. 1. 8. Druhy serverů

- webový server – především v síti Internet poskytuje WWW stránky
- souborový server – slouží např. v podnikové síti jako centrální úložiště dat (dokumentů)
- databázový server – slouží jako úložiště strukturovaných dat (databází), umožňuje provádět vyhledávání
- tiskový server – zpřístupňuje ostatním uživatelům sítě služby tiskárny
- faxový server
- proxy server – zprostředkovává ostatním uživatelům sítě přístup do sítě jiné (např. Internet)

- aplikační server – počítač specializovaný na provoz nějaké aplikace
- herní server – nabízí hraní her s více hráči (multiplayer)

a další .

Nutno zdůraznit, že jeden server (jeden stroj) může kombinovat více funkcí.

2. Geografické Informační Systémy

Mnoho informací v našem životě (a to ať jsme jedinec, malá firma nebo nadnárodní koncern) je spojeno s dvěma problémy:

- většina informací je spojena s geografickým určením místa tyto informace je při narůstajícím množství čím dál tím těžší vyhodnocovat, zpracovávat a třídit. Některé studie ukazují, že téměř 70% informací má společný prvek - geografické určení místa. Lidé se již dlouhou dobu snaží tyto informace pomocí různých nástrojů zvládnout. Po desetiletích výzkumu a všemožných praktických pokusech můžeme říci, že teprve v poslední době, s obrovským nárůstem výkonu počítačů, dochází k hromadnému rozšiřování prostředků pro manipulaci s geografickými informacemi. Jedná se o tzv. systémy GIS (Geografické Informační Systémy), které pomáhají člověku zpracovávat velká množství dat v závislosti na geografickém určení místa. Systémy GIS se tak stávají hlavním a asi zatím jediným zpracovatelem dat, určených v závislosti na zeměpisné poloze.

2. 1. Co jsou to systémy GIS

Systémy GIS (Geographical Information Systems) jsou jednou z významných aplikací počítačové grafiky. Bohužel se zatím neprávem netěší takové přízni jako např. systémy CAD, i když s určitostí můžeme říci, že jejich význam (a obzvláště pro budoucnost) je plně srovnatelný. Je to bohužel způsobeno nízkou obecnou informovaností veřejnosti. Důvodů je samozřejmě více - vysoká cena, ne vždy ideální rozhraní mezi počítačem a člověkem (tzv. interface), vždy je potřeba vykonat veliké množství práce (sběr dat, digitalizace, následné upravení dat atd.) než jsou vidět první výsledky práce. To vše způsobuje relativně nízké využití tohoto mocného nástroje v našich krajích.

Jak bychom tedy co nejjednodušeji popsali systémy GIS ? Pokud bychom použili některé z profesionálních definic, pak můžeme říci: GIS je informační systém, navržený pro práci s daty, která jsou reprezentována prostorovými nebo geografickými

souřadnicemi. Je to automatizovaný systém pro sběr dat, jejich uchovávání, třídění, úpravu, analýzu a následné zobrazení.

GIS poskytuje možnost znázorňovat realitu pomocí uskupení různých mapových vyjádření (např. topografické, geologické, vegetační, hydrometeorologické, katastrální a jiné mapy, letecké či družicové snímky atd.), a v to libovolné kombinaci. Se všemi těmito informacemi lze nadále pracovat při tvorbě analýz, prognóz a modelů různých situací. Tato grafická (mapová) vyjádření jsou pomocí GISu úzce provázána s informacemi obsaženými v databázích, a to činí GIS účinným nástrojem. Díky názornosti, kvantifikovatelnosti přehlednému grafickému vyjádření poskytuje GIS významnou podporu managementu. Možnosti GISu (názorné grafické informace svázané popisnými informacemi) jsou významné obzvláště ve státní správě , kde mohou usnadnit a zefektivnit rozhodování. Navíc umožní zrychlení přístupu k různým mapám a oborovým databázím a zlepšení jejich provázanosti. Na různé otázky, jako např.: kde co postavit či vybudovat, kdy co a jak udržovat, jak se projeví určitý zrealizovaný záměr za měsíc, za rok či za 10 let atd. může GIS dát odpověď anebo ji pomoci nalézt. GIS jako nástroj lze používat na úlohy různých měřítek (tzn. na různě velká území) tedy i ve státní správě lze použít GIS na různých úrovních: na centrální, na úrovni okresů i na úrovni komunální. Uplatnění GIS ve státní správě je typické v následujících oblastech:

- nástroj pro tvorbu analýz a prognóz z hlediska regionálního rozvoje, jakož i pro porovnávání prognóz s realitou (přímo v mapových vyjádřeních)
- nástroj pro velkoplošnou kontrolu prostorových a stavových změn např. lesních porostů, zemědělských ploch, urbanizovaných prostor, zásahů v krajině (různé stavby apod.), využití leteckých snímků
- nástroj pro podporu při řízení povolování staveb
- podpora při vydávání souhlasu nebo nesouhlasného stanoviska k návrhům územně plánovací dokumentace pro velké územní celky
- podpora k vyjádření při schvalování návrhů tras celostátních a tranzitních liniových staveb a jejich staveb a jejich součástí
- nástroj pro optimalizaci (ochranářskou i ekonomickou) různých zásahů v krajině
- kontrola hospodaření v lese, bezlesí, kontrola ochrany lesa
- podpora managementu velkoplošných chráněných území (národních parků apod.)

Ve výčtu uplatnění by bylo možno pokračovat ještě dlouho, protože GIS je asi nejvíce mohutnou platformou s mocnými analytickými nástroji pro prostorová data a databázové systémy a vždy bude pouze záviset na člověku, který bude s GISem pracovat a na jeho v fantazii při využití tohoto nástroje.

Protože se nejsnáze demonstrují možnosti systémů GIS na příkladech, je zde nyní uvedeno několik nejvýznamnějších použití systémů GIS v praxi, Tyto však bohužel pocházejí většinou ze zahraničí, protože , jak už bylo zmíněno dříve, využití systémů GIS není v našich krajích zatím příliš vysoké:

- síť supermarketů hledá lokalitu pro svůj nový obchodní dům v závislosti na demografických, socio-ekonomických ukazatelích populace pravděpodobných budoucích zákazníků v navrhované oblasti (příjmové skupiny, věk obyvatelstva, dostupnost, vzdálenost od ostatních center, blízkost jiných atrakcí, dostupnost zásobování - variací na tento problém by mohlo být nalezení lokality pro benzínovou pumpu, skládku odpadů, zemědělské stavby, nového školního zařízení atd.)
- městské zastupitelstvo hledá minimální cestu pro čištění a údržbu ulic, pouličního osvětlení, odvoz odpadů apod.
- v případě nepředvídané havárie, odhad šíření rychlosti znečištění podzemních vod, šíření ropné skvrny po havárii tankeru
- v oblasti ochrany životního prostředí může být GIS nasazen od jednoduchých aplikací jako pouhá databáze např. stávajícího stavu lesních ploch až po složité aplikace jako modelování eroze půdy při velkých deštích
- ve zdravotnictví může být použito nalezení nejrychlejší cesty k pacientovi v závislosti na stavu silnic, jejich průjezdnost; GIS však může být použit např. v epidemiologii k nalezení centra šíření epidemie
- vhodné umístění odpadkových košů ve vzdálenosti např. 5 min chůze od centra rychlého občerstvení - různá distribuční nebo sběrná centra (ať již komerční či nekomerční) hledající optimální umístění svých center, vzhledem k rozmístění svých potenciálních odběratelů - ta samá centra, ale tentokrát hledající optimální cestu (přičemž optimální většinou znamená minimální, ať už z hlediska časového nebo vzdálenostního) ke svým zákazníkům

Jedno z nových uplatnění se nalézá v navigačních systémech, které jsou zabudované přímo v dopravním prostředku - zde právě fungují systémy GIS jako podpůrný prostředek . Je možno říci, že pokud splňujete několik podmínek (cestujete v Severní Americe, máte notebook napojený přes modem a mobilní telefon na Internet, což již není v současné době taková exotická představa) můžete již sami velice jednoduše a lacino na adrese http://www.mapquest.com využívat navigačního systému, můžete si nalézt jakoukoliv soukromou nebo komerční adresu a nechat si ji zobrazit na mapě a zvětšit až na úroveň hledané ulice s vyznačením kde se nachází dané popisné číslo. Můžete si rovněž nechat vypsat itinerář jízdy mezi jakýmikoli dvěma místy v Severní Americe (i Kanada a Mexiko). Podobně si můžete v Austrálii v Sydney najít adresu na mapě města ještě před tím než tam pojedete a aniž by bylo nutno si kupovat mapu (adresa http://www.whitepages.com.au). Rovněž velká budoucnost se vkládá do spojení systémů GIS s další bouřlivě se rozvíjející oblasti systémů GPS (Global Positioning Systems). Tyto systémy jsou dnes schopné určit Vaši polohu kdekoliv na Zemi s přesností několika metrů a přitom jejich velikost nepřesahuje příliš velikost kapesních kalkulátorů. Proto jsou velice vhodné pro navigační systémy ve spojení se systémy GIS.

Se zmínkou o celosvětové počítačové síti Internet, ke které je připojena i naše škola, bych se rád zmínil o zajímavém projektu v USA. Tento projekt si dává za cíl usnadnit výměnu dat mezi jednotlivými státy USA a vzájemnou informovanost o tom, kde je co vytvořeno, kde existují jaká demografická a geografická data zpracované pro systémy GIS. Po připojení na danou adresu se před Vámi objeví mapa USA s vyznačenými jednotlivými státy. Po kliknutí na jednotlivý stát unie se vypíší veškeré datové soubory použitelné pro systémy GIS související s tímto státem. Toto místo neustále expanduje s tím jak přibývají nové a nové informace a jeho největší význam tkví v tom, že nedochází zbytečně k současné tvorbě stejných datových souborů (což je u systému GIS ta nejnáročnější část).

2. 2. Problematika GIS

Je však potřeba upozornit na jeden fakt - podle tohoto popisu by se mohlo zdát, že se jedná o všemocný prostředek, který za nás vyřeší veškeré problémy. Systémy GIS jsou zajisté mocným nástrojem, ale nesmíme při jejich nasazení zapomenout, že samy

o sobě jsou pouze sadou výkonných nástrojů, které jsou využívány člověkem a že to budou právě lidé, kteří budou řešit tyto problémy. Pokud použijeme analogie se systémy CAD, ke kterým mají systémy GIS asi nejblíže (a na jejichž platformě je i několik GISových programů založeno jako jejich nadstavba), můžeme říci, že se jedná o prostředek, který pomáhá člověku při rozhodování.

GIS je systém složený z několika navzájem provázaných prvků, které se všechny podílí na úspěšném vyřešení problémů (viz. obr.1)

Při definování problému bývá zpravidla nejzávažnějším problémem definování všech faktorů, které mohou naše rozhodování ovlivňovat. Jak již bylo řečeno systémy GIS nejsou samospasitelným řešením a mohou se vyskytovat situace, kdy je nasazení GISů nevhodné. Proto by si měl každý zodpovědný pracovník před použitím systému GIS položit několik zásadních otázek pro řešení daného problému:

- jaké existují dostupné mechanismy získávání, skladování a zpětného získávání geografických dat
- v jakém formátu a množství se vyskytují
- jak přesné jsou tyto informace
- co získáme použitím systémů GIS
- jaké máme alternativy k použití systémů GIS a kolik nás budou tyto alternativy stát
- kdo bude koncovým uživatelem námi vytvořeného modelu a kolik těchto uživatelů bude
- jaký výkon můžeme očekávat v otázce rychlosti přístupu a vyhodnocování dat
- jaké je předpokládané množství zpracovávaných dat
- jaké množství již existujících archivů bude nutno převést do digitální podoby, jakou dobu to bude trvat a jaké budou náklady na toto převedení
- je tento model určen rovněž i pro budoucí smysluplné rozšiřování a modifikace nebo je vytvořen pouze pro vyřešení jediného specifického problému

Teprve pokud z odpovědí vyplyne, že použití systému GIS by bylo přínosem, můžeme se začít zabývat vlastním tvorbou projektu. Jedním z dalších rozhodnutí musí být, který systém GIS budeme používat. Výrobců je mnoho a každý se nás snaží přesvědčit, že ten jeho výrobek je ten právě nejlepší na trhu. Nesmíme proto hned

podlehnout reklamním tlakům a nejprve se zaměřit na vlastní problém. Nikoliv tedy na nabízené GISy, ale nejdříve pořádně analyzovat daný problém, určit co od něj očekáváme, jaké výsledky by měl poskytovat, jaká data budeme zpracovávat, jakou budou mít strukturu a jaké budeme požadovat výstupy. Rovněž nezanedbatelným kritériem by mělo být zhodnocení kvalifikace obsluhy, která bude se systémem GIS pracovat - tzn. měli bychom se dívat i jaké uživatelské rozhraní ten , který systém poskytuje. V neposlední řadě se budeme určitě zajímat o to, jaké hardwarové nároky má daný systém a jeho cenu. Protože jak již bylo řečeno, výrobců je mnoho a systémy se prodávají za různé ceny, které mohou mít v některých případech povahu sharwarového poplatku u jednoduchých zobrazovacích systémů GIS až po mnohadesetitisícové hodnoty u takových mohutných a výkonných systémů jako je program ARC/INFO firmy ESRI nebo GeoWorks od firmy Intergraph.

2. 3. Součásti GISu

Plnohodnotný geografický informační systém se - stejně jako obecný informační systém - skládá ze 4 součástí:

- Hardware - nejčastěji osobní počítač s barevným monitorem, skener pro možnost vstupu obrazových dat, tiskárna či plotter pro možnost mapového výstupu.
- Software - specializovaná sada programů pro analýzu a vizualizaci geodat.
- Data - nejdůležitější a často finančně nejnáročnější součást GISu.
- Pracovníci (uživatelé) - lidé se znalostmi geografie schopní obsluhovat informační technologie.

2. 4. Geodata, geoobjekty

Data, se kterými GIS pracuje se nazývají geodata. Geodata se skládají z jednotlivých geoobjektů. Geoobjekt je část modelované reality, kterou je možno na dané úrovni generalizace v GISu modelovat jako jeden objekt. Geoobjekt obsahuje dva druhy informací:

- prostorové informace (tvar, poloha, topologie)
- neprostorové informace (atributy, specifické pro každý typ objektu)

Generalizací v prostředí GISu se rozumí problém toho, jak podrobně realitu modelovat. Např. město lze v GISu reprezentovat jedním objektem, nebo množinou objektů (budov, parcel, ulic, ploch apod.).

2. 5. Dimenze geoobjektů

Základní dělení geoobjektů je dělení podle počtu dimenzí. Reálné objekty na zemském povrchu jsou vždy trojrozměrné. Do prostředí GIS se však transformují podle potřebné úrovně generlizace.

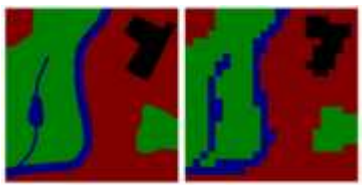
- 0D geoobjekty - Bezrozměrné objekty, body, definované pouze svou polohou. Příkladem může být například autobusová zastávka v GISu modelujícím dopravu nebo GSM vysílač v GISu mobilního operátora modelující pokrytí signálem.

- 1D geoobjekty - Objekty jednorozměrné, úseky čar, s konečnou délkou a nulovou plochou. Pomocí 1D geoobjektů se nejčastěji modelují silnice, řeky, apod.

- 2D geoobjekty - Objekty dvojrozměrné, polygony, s konečným obvodem a konečnou plochou.

- 3D geoobjekty - Objekty trojrozměrné, polyhedrony. V GISech se používají výjimečně, ve specifických případech. Třetí rozměr je v GISech nejčastěji modelován pomocí tzv. Digitálního modelu terénu (DMT, DEM).

2. 6. Mapové vrstvy



Obr. 2. 1. Ilustrace rozdílu mezi vektorovou a rastrovou mapovou vrstvou

Geoobjekty popisující stejné téma se sdružují a ukládají do mapových vrstev, někdy také nazývaných tematické mapové vrstvy. Takovým tématem může být např. vodstvo, silnice, typy půd, nadmořská výška, apod.

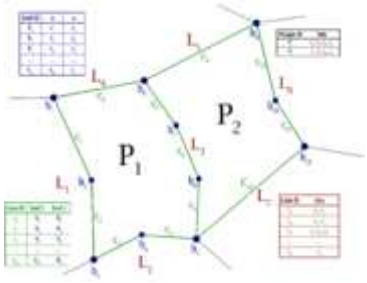
Smyslem dělení geodat do mapových vrstev je usnadnit analýzu dat. Ta je nejčastějším důvodem pro nasazení GISu pro modelování reality.

Každá mapová vrstva je uložena v jednom datovém souboru, který lze samostatně přenášet a používat ve více mapových projektech. Mapové vrstvě se někdy také říká monotematická mapa, případně zkráceně mapa (např. mapa řek, mapa silnic, apod.). Mapové vrstvy se dělí podle modelovaných dat a druhu použití na dva typy - vektorové a rastrové.

2. 6. 1. Vektorové mapové vrstvy

Ve vektorových mapových vrstvách jsou data uložena pomocí bodů a čar. Bod je základním elementem s definovanou polohou (souřadnicí) a nemá z geometrického hlediska žádný rozměr. Čára je úsečka nebo křivka spojující dva body. V běžných GISech se z důvodů zjednodušení křivka reprezentuje pomocí seřazené sekvence bodů spojených přímou čarou. Modelování geodat pomocí vektorů úzce souvisí s teorií grafů.

2. 6. 2. Vektorové modely



Obr. 2. 2. Ukázka uložení vektorových dat v hierarchickém vektorovém modelu

Vektorová data jsou v GISech organizována a ukládána podle různých vektorových modelů.

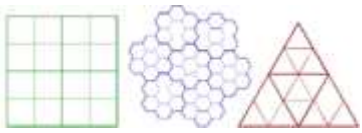
- Špagetový model - Ve špagetovém modelu jsou všechny typy objektů, bez ohledu na počet dimenzí, uloženy v jednom heterogenním seznamu.Tento seznam má pouze 2 položky:
    - *typ objektu* - bod, čára, polygon
    - *parametry objektu* - jedna či více souřadnic

  Ve špagetovém modelu není obsažena žádná informace o topologii (sousednost, orientace, konektivita, obsahování) a proto je tento model pro analýzu geodat obtížně použitelný.Navíc zde dochází k redundanci dat.

- Hierarchický model - Hierarchický model ukládá data hierarchicky s ohledem na počet dimenzí. Vychází s faktu, že polygon se skládá z několika linií, linie z několika úseček, úsečky jsou pak spojením dvou bodů. Tyto elementy jsou pak v GISu uloženy samostatně, nejčastěji v geodatabázi.

- Topologický model - Topologický model je kompromisem mezi špagetovým a hierarchickým modelem. Ukládají se pouze body a čáry, přičemž k čáře lze připojit informaci o její orientovanosti, podle níž lze pak určit sousedný polygon vlevo a vpravo.

2. 6 3. Rastrové mapové vrstvy



Obr. 2. 3. Různé typy rastrů - čtvercový, šestiúhelníkový, trojúhelníkový

Rastrových mapových vrstev se používá k modelování veličin, které jsou spojitě definovány na celém modelovém prostoru. Příkladem může být mapová vrstva nadmořské výšky, mapa typu půd, vegetace, atmosférického tlaku, teploty, apod.

Prostor je v rastrových mapových vrstvách rozdělen na množství malých plošek, jejichž rozměr je dostatečně malý na to, aby bylo možno na jejich povrchu hodnotu dané veličiny považovat za konstantní.

Dělení prostoru může být buď pravidelné, nebo nepravidelné, buňky rastru mohou být různého tvaru (čtverec, trojúhelník, šestiúhelník). V naprosté většině případů se ale v GISech používá pravidelné dělení prostoru čtvercovou mřížkou.

Každé buňce rastru přísluší hodnota sledované veličiny v daném místě. Tato hodnota (typicky číselná) může být dvojího typu, podle nějž se také rastrové mapové vrstvy mohou dělit na:

- Rastrové vrstvy výčtového typu - Každá buňka rastru obsahuje jistý kód, typicky celočíselný, z rozsahu 1...N. Tento kód reprezentuje kategorii sledovaného jevu. Nutnou součástí rastrové vrstvy tohoto typu je proto překladová tabulka, která kódy interpretuje. Rastry výčtového typu se používají tam, kde má zkoumaný jev konečný počet hodnot (např. typ vegetace), nebo tam, kde lze spojitou veličinu rozdělit do konečného počtu kategorií (např. nízká, střední a vysoká hustota zalidnění).

- Rastrové vrstvy hodnotového typu - Každá buňka rastru nese informaci o diskretizované hodnotě spojité veličiny, která může teoreticky nabývat nekonečného počtu hodnot. V praxi je samozřejmě omezena rozsahem a přesností použitého datového typu (integer, float). Takto reprezentované veličině se v prostředí GISu někdy říká prostorový proces. Příkladem prostorového procesu může být nadmořská výška, atmosférický tlak, teplota, apod.

3. Mapový server

O mapových serverech bylo před časem docela slyšet. Bez nich si určitě řada z nás dovede představit svůj další internetový život jen těžko. Existují velké mocné aplikace na velkých serverech, ale i malé servříky pro zveřejňování například dat z lokálních průzkumů. O tom, jak si takovou vlastní malou aplikaci vytvořit, bude pojednávat tento seriál.

Mapové servery hýbou světem. Snad každý z vás již zabrousil na některý ze serverů zobrazujících mapy, ať již na některém z českých porálů (Atlas.cz, Centrum.cz nebo například Seznam.cz), a nebo na specializované weby, jako jsou mapové servery Ústavu pro hospodářskou úpravu lesů, komerční data firem Geodis nebo společnosti T-Mapy a v neposlední řadě v poslední době se objeviší servery Google maps, kde lze najít některé mapové listy ve slušném rozlišení i na území ČR, a nebo Virtual Earth společnosti Microsoft. V tomto seriálu bych vás rád seznámil se základy tvorby mapových serverů krok za krokem. Průvodce by měl umožnit potenciálním uživatelům snazší nástup do této problematiky.

3.1. Úvod k mapovým serverům

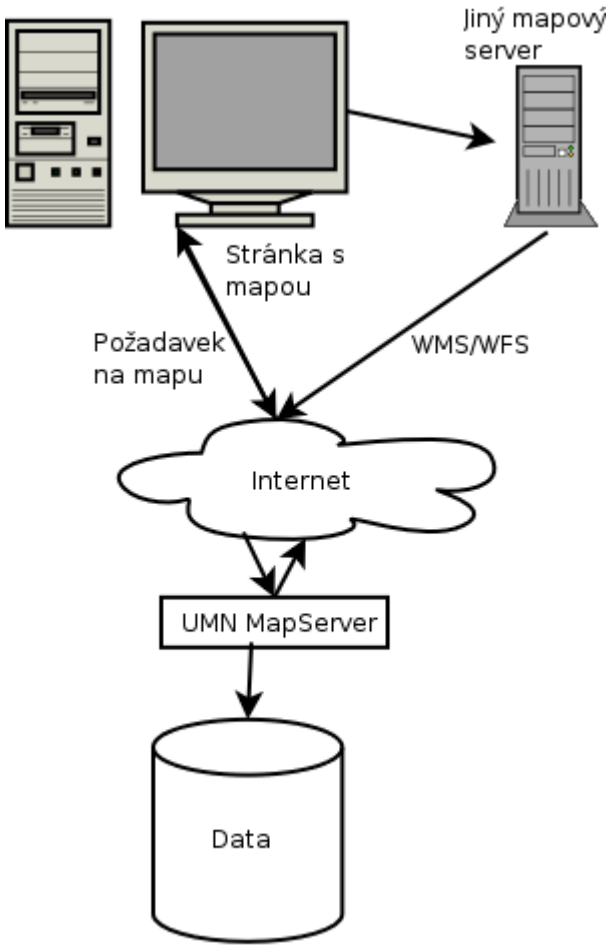
Mapové servery jsou programy pracující na architektuře klient-server, zpracovávající data s geografickým vztahem. Mohli bychom také říci, že jsou to v podstatě geografické informační systémy, které jsou ovšem ovládány pouze pomocí parametrů -- textově -- a neinteraktivně. Spolupracují s některým z webových serverů, který jim předá potřebné parametry z webového formuláře. Ty jsou zpracovány a zpět je vrácen buď soubor s mapou, a nebo výsledek dotazu.

Mapových serverů je celá řada -- některé jsou komerční (na příklad od firem ESRI, TopoL nebo T-Mapy), některé jsou uvolněny pod některou z licencí umožňující jejich svobodné užívání. Mezi ty druhé patří asi nejoblíbenější mapový server -- MapServer univerzity v Minnesotě. Tento program sice není uvolněn pod oblíbenou GNU GPL, jeho licence je ale dostatečně ,,svobodná`` na to, aby umožňovala kolektivní vývoj a další věci z toho plynoucí. Původně vznikl jako školní projekt, jak už to ale u podařených projektů bývá, začal žít svým životem. Budu-li v budoucnosti hovořit o mapovém serveru, budu tím mít na mysli právě mapový server z Minnesoty.

3. 1. 1. Princip fungování

Existují v zásadě tři možnosti, jak používat program mapserver:

- buď jako CGI-program vracející za základě vstupních parametrů obrázek mapy (map)
- nebo jako CGI-program vracející na základě vstupních parametrů a šablony hotovou internetovou stránku s mapou
- nebo můžeme použít celou řadu rozhraní k různým programovacím jazykům -- MapScript, jako jsou Python, PHP či oblíbený Perl.
-



Obr. 3. 1. Pokus o zobrazení schématu fungování mapserveru

Na obrázku jsem se pokusil znázornit schéma, jak mapserver pracuje. Z pracovní stanice přijde dotaz na server s požadovanými mapovými vrstvami a souřadnicemi

zájmového území. MapServer vyrobí z daného území obrázek mapy se všemi vrstvami, legendu, referenční mapu a tak dále. Vše je definováno v konfiguračním souboru. Zpět ke klientovi je v závislosti na požadavku vrácen obrázek mapy, celá HTML stránka nebo výsledek dotazu do databáze. Jednotlivé servery spolu mohou komunikovat pomocí služeb WMS, WFS a dalších a poskytovat si tak na vzájem mapy.

Rád bych se postupně zmínil o prvních dvou možnostech. Pokročilejší uživatel by již měl být schopen napsat si požadovanou aplikaci na základě dokumentace. Aby člověk mohl skutečně naplno využít veškeré možnosti mapserveru, je potřeba umět následující jazyky nebo se alespoň trochu vyznat v jejich základech:

- HTML slouží k vytváření výsledné aplikace. Není na škodu umět CSS, pomocí kterých lze efektivně dosáhnout požadovaného vzhledu. Z funkčního hlediska jsou nejvíce používány formulářová pole, pomocí nichž se předávají parametry mapserveru.
- JavaScript je programovací jazyk, který z duše nemám rád. S jeho pomocí lze ovšem výslednou aplikaci rozpohybovat, nastavit ty správné hodnoty do těch správných formulářových polí, zapnout ty správné funkce a nakonec i ,,zmáčknout`` tlačítko tvořené obrázkem, aby bylo ,,vidět``, jaký nástroj je právě aktivní. Naštěstí existuje již celá řada příkladů, které stačí pouze zkopírovat z výborné dokumentace a případně je jen maličko přiohnout tak, aby dělaly, co mají.
- pokročilejší uživatelé a programátoři jistě uvítají možnost ,,jít o úroveň níže`` - vzít věci pěkně do vlastních rukou a pomocí svého oblíbeného programovacího (skriptovacího) jazyka napsat celou mapovou aplikaci podle sebe - s mapserverem pouze v pozadí. Perl, Python a PHP jsou ty základní. Velká výhoda je, že funkce jsou pro všechny jazyky stejné, takže dokumentace je jednotná.
- komu skriptovací jazyky nestačí a chtěl by se podívat aplikaci přímo na střeva, využije samozřejmě znalost Céčka...
- v Javě jsou psány některé rozšiřující applety, o kterých se zmíníme také. Ačkoliv sám jsem zastáncem co nejjednodušších stránek, JBox rozšiřuje vlastnosti webové mapy natolik, že mi za to stojí.
- budete-li mít uložena některá vektorová data v databázi PostgreSQL

3. 1. 2. Instalace

Existuje několik možností, jak Mapserver nainstalovat: klasickou cestou ze zdrojových kódů a nebo z připravených balíčků. Ať již tak, nebo tak, připravte se na poměrně hodně závislostí, z nichž jako nejzákladnější bych rád zmínil:

- Knihovna GDAL/OGR slouží k načítání vektorových a rastrových dat v různých formátech. Formátů pro rastry a vektory je skutečně požehnaně. Většina svobodných GISů (na příklad GRASS nebo QGIS) využívají právě tyto knihovny pro svou práci -- v systému jsou tedy tak jako tak přítomny.
- Knihovna Proj se stará o převody mezi různými souřadnými systémy. Kartografická projekce je, velmi zjednodušeně řečeno, způsob, jak 3D povrch zemského tělesa (Geoidu) převést na 2D plochu -- mapu. Jsou různé způsoby vedoucí k různým zkreslením. Každá kartografická projekce má navíc ještě trochu jiný souřadnicový systém s různými počátky, z nichž jeden z nejobskurnějších je v České republice dosud hojně používaný S-JTSK, který má nejen „prohozené osy`` x a y, ale jejich kladný směr je směrem doleva-dolů. Kromě tohoto systému se v České republice používá systém WGS-84, S-42 a nebo prostý Latitude-Longitude udávající pozice ve sférických souřadnicích. O všechny převody se ale stará právě knihovna Proj. Stačí pouze vědět, v jakém systému je mapa uložena, a všechny vrstvy budou zobrazeny na jednom místě. Není problém mít mapy uloženy v jakémsi obecném systému X,Y a všem těmto probémům se vyhnout za předpokladu, že znáte rozlišení rastrů a můžete je zohlednit.

Instalace ze zdrojových kódů probíhá klasickou trojicí configure; make; make install. V každém případě by měla vzniknout jedna binárka, která by měla být umístěna v adresáři s CGI skripty webového serveru, tedy například /usr/lib/cgi-bin/. Instalace se ověří tak, že webový prohlížeč nasměrujete na adresu http://localhost/cgi-bin/mapserv, a měli byste v okně prohlížeče dostat hlášku podobnou této:

No query information to decode. QUERY_STRING is set, but empty.

Pokud se tento nápis objevil, pak je vše v pořádku a může se pokračovat dále. Pokud ne, je nutné zkusit se podívat například do logu webového serveru (/var/log/apache/error.log), problém asi bude někde v cestách či v právech.

3. 1. 3. Doporučená adresářová struktura

Data jsou dnes asi to nejcennější, co je. Účelem mapového serveru není primárně data distribuovat v jejich původní formě, ale zprostředkovávat pouze náhled na ně. Proto musí být uložena někde bezpečně v systému tak, aby na ně sice ,,viděl`` uživatel, který na ně vidět má (uživatel, pod kterým běží webový server, na příklad www-data), ale aby se jen tak ,,nepovalovala`` na webu, aby si je kdokoliv mohl stáhnout. Proto vytvořme adresář s daty, např. ve /var/mapservdata.

Tento adresář se rozdělí na dva, v jednom budou vlastní data -- mapy ve vektorovém a rastrovém formátu, ve druhém různé HTML šablony, například pro výsledky různých dotazů.

Někde v kořenovém adresáři webových stránek (/var/www/) kvůli pořádku je dobré vytvořit ještě adresář pro mapserveří stránky, soubor s CSS styly, applety a podobně. Další adresář, který je zapotřebí, je místo, do kterého si bude Mapserver ukládat všechny dočasné obrázky a ze kterého si je bude brát webový klient. Například /var/www/mapserver/temp/.

Výsledná adresářová struktura by tedy měla vypadat asi následovně:

```
/var/mapservdata/       # adresář na data
/var/mapservdata/data/  # podadresář s vlastními daty
/var/mapservdata/tmpls/ # podadresář se šablonami a dalšími proprietamy
/var/www/mapserver/     # adresář s CSS, applety a podobně
/var/www/mapserver/temp/ # dočasné a výsledné obrázky pro klienta
/usr/lib/cgi-bin/mapserv # vlastní CGI mapserver
```

Nikdo nikomu samozřejmě nebrání vytvořit si strukturu vlastní. Toto je pouze návrh, na kterém by mělo být na první pohled zřejmé, co je vidět pouze ,,ze systému`` a co také ,,z webu``.

3. 1. 4. Data

Jak přijít k datům a jaký by měla mít formát? Data jsou rastrová a vektorová. Rastrová data mohou být všechna, která jsou podporována knihovou GDAL, jedná se tedy zejména o standardní TIFFy, ale i o JPEGy, PNG, GIF a další již speciální formáty. Vektorová data mohou být ve všech formátech podporovaných knihovnou OGR. Jak je na tom systém, se zjistí prostě příkazy gdal-config - formats a ogrinfo - formats.

Pokud máte vlastní data, zvykovým standardem bývá formát TIFF (GeoTIFF) pro rastry a ESRI Shapefile pro vektory, uložte je nyní do vašeho adresáře na data pro mapový server, v našem případě tedy do /var/mapservdata/data/. Nemáte-li zatím žádná data, stáhněte si připravený dataset a rozbalte jej do vašeho adresáře. Jedná s o data z tutoriálu ze stránek Mapserveru (resp. jsou vykopírovaná přímo z balíku Itasca-demo (zip):

```
cd /var/mapservdata/data/
tar -xvzf /cesta/k/souboru/s/daty/mapserv_data.tgz
ls -l
```

Všimne si, že vedle rastrových souborů s koncovkou *.tif (které jsou pouze dva) existují ještě textové soubory s koncovkou *.tfw, ve kterém jsou uloženy informace o souřadném systému, souřadnice levého horního rohu, rozlišení ve směru x a y a případná rotace. Soubory ve formátu ESRI Shapefile jsou rozděleny do třech: *.shp, *.shx, *.dbf. Musí být vždy pohromadě. Soubor *.dbf se může prohlížet a upravovat případně pomocí nějakého tabulkového kalkulátoru, ovšem pozor, některé programy načítají pouze prvních (oocalc) 32 000 řádků, což se může na první pohled jevit jako hodně vysoké číslo, na druhý ovšem velmi málo, neboť zbytek souboru bývá prostě odříznut.

Data jsou připravena, připravenou adresářovou strukturu a nainstalovaný mapserver.

4. Mapfile

Mapfile je konfigurační soubor, ve kterém je definována jednak zobrazovaná oblast a jednak to, jak se mají jednotlivé vrstvy vykreslovat. Vyladěný mapfile – to je cíl každého mapserveristy.

Mapfile nemusí být "viděn" z webu, proto se může editovat v adresáři s daty:

```
cd /var/mapservdata/
$EDITOR jezera.map
```

Struktura mapfilu je poměrně jednoduchá: Mapfile se skládá z několika objektů navzájem do sebe vnořených a ukončený vždy slovem END. Každý parametr se zapisuje na samostatnou řádku, na velikosti písma nezáleží (END a end jsou interpretovány stejně).

Jsou tři základní objekty, které mají různé vlastnosti:

```
MAP
 |-LAYER
     |-CLASS
        |-STYLE
        |-LABEL
     |-CLASS
     |-...
 |-LAYER
 |-...
```

MAP je něco jako projekt - je definován územím, kartografickou projekcí, referenční mapou a podobně. V tomto objektu se nastavují globální vlastnosti výsledných map.

LAYER už jsou jednotlivé mapové vrstvy, které se dále mohou dělit na CLASS třídy s vlastním STYLE - stylem zobrazení. Příkladem budiž LAYER Vodstvo s CLASSes na příklad Potoky a řeky a Nádrže.

Zprvu je nutné připravit si mapfile o jedné vrstvě - hranice území, jednotlivé řádky mohou být komentovány:

```
##############################################################
# začátek všech vrstev
##############################################################

  LAYER # začátek vrstvy
    NAME        staty_plochy # jak se jmenuje
    DATA        states_ugl # v jakém souboru (souborech) se nachází
    TYPE        POLYGON # typ dat
    STATUS      OFF
    CLASS # začátek třídy
        NAME 'Plochy států'
        STYLE # jak bude třída vypadat
            COLOR 225 225 185
        END # konec stylu
    END # konec tříd
  END # konec vrstvy

  LAYER # začátek vrstvy
    NAME        staty_hranice # jak se jmenuje
    DATA        states_ugl # v jakém souboru (souborech) se nachází
    TYPE        LINE # typ dat
    STATUS      OFF
    CLASS # začátek třídy
        NAME 'Hranice států'
        STYLE # jak bude třída vypadat
            COLOR 128 128 128
        END # konec stylu
    END # konec tříd
  END # konec vrstvy
##############################################################
# konec všech vrstev
##############################################################
```

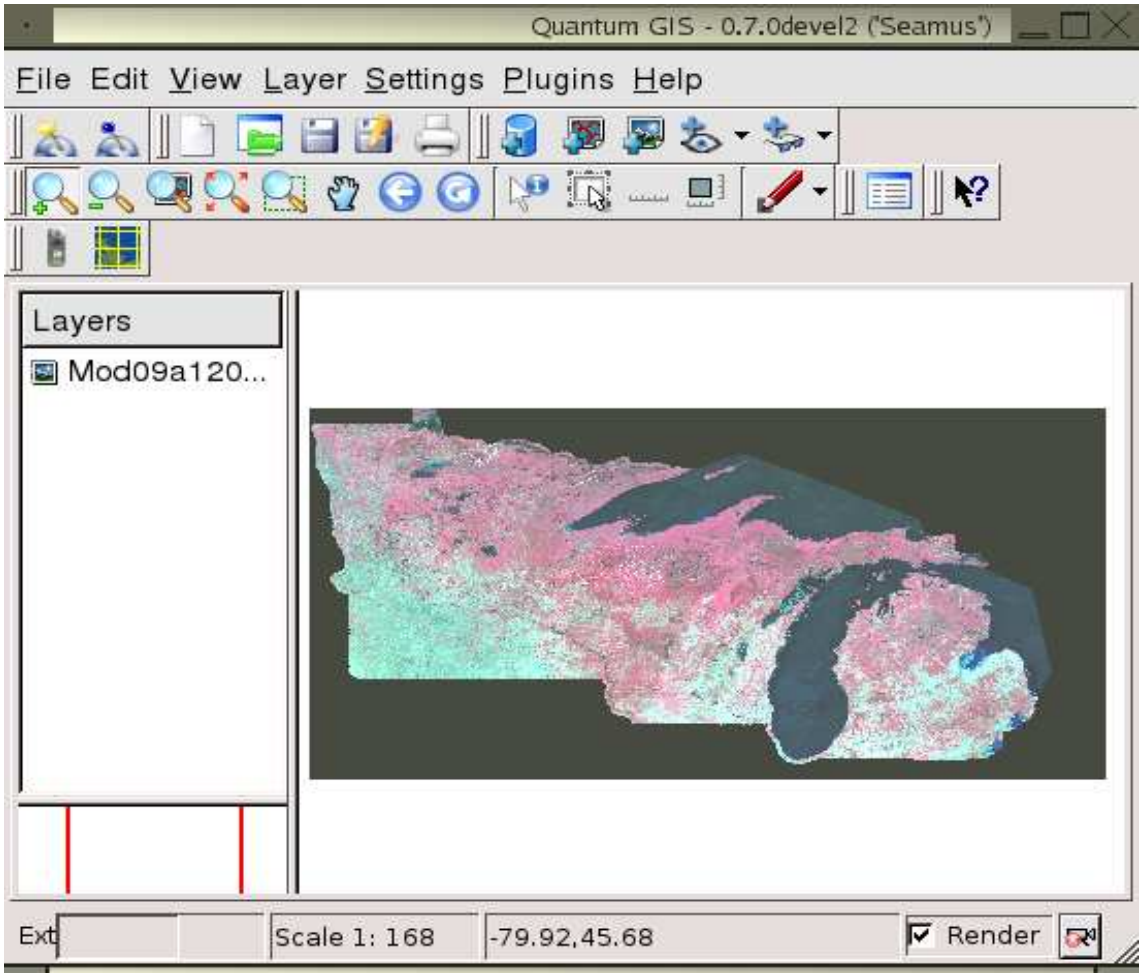
Jednotlivé položky trochu blíže:

NAME

Každý objekt může být pojmenován. Většinou je to nezbytné, jindy ne tolik. Pomocí jména se na jednotlivé objekty můžeme později odkazovat.

EXTENT

To jsou hraniční souřadnice projektu. Jsou zadávány v pořadí západ jih východ sever. Vždy je nutno vědět, na jakém území se pracuje. V případě nejistoty je nezbytné sáhnout po některém ze specializovaných programů - GISů a nebo použijte na příklad utilitu ogrinfo: ogrinfo -al -so states_ugl.shp.



SHAPEPATH

je cesta k adresáři s daty z pohledu systému. Adresář a data musí mít nastaven parametr pro čtení pro uživatele, pod kterým běží váš webový server. Data můžete mít i jinde, v tom případě ale musí být v parametru DATA uložena celá cesta.

IMAGECOLOR

Barva pozadí mapy. Všechny barvy jsou nastavovány pomocí RBG s rozsahem 0-255.

IMAGETYPE

Typ obrázku, který se má mapserveru vrátit. Možnosti jsou GIF, JPEG, PNG a WBPM

DATA

Soubor s daty, bez koncovky. Tento soubor (tedy konkrétně ctybdpy2.shp) je uložen v adresáři SHAPEPATH.

TYPE

Říká, o jaký typ dat se jedná. Rozeznáváme POLYGON (plochy), LINE (linie), POINT (body) v případě vektorových dat a dále máme k dispozici typy RASTER a ANNOTATION.

STATUS

Nastavuje výchozí příznak zobrazování vrstvy. Možné parametry jsou ON, OFF a nebo DEFAULT. OFF znamená, že nebude-li vrstva explicitně vyvolána parametrem layer v URL (viz níže), nebude vykreslena.

CLASS

Uvozuje objekt třídy. Tříd může být jedna až mnoho v rámci každé vrstvy. Třídy jsou vytvářeny na základě nějakého atributu vrstvy -- v případě vektorových dat na základě hodnot ze sloupečku z databáze, v případě rastrových dat na základě hodnot rastru. A nebo může třída pokrývat "všechny hodnoty" jako v tomto případě.

STYLE

Aby se vůbec něco zobrazilo, musí být řečeno, jak se to má udát.

OUTLINECOLOR

Barva hraničních linií polygonu

Mapfile je doslova "ořezaný na kost". Postupně budeme přidávat další parametry a atributy a vytvářet mapu k obrazu našemu. Zatím pracujeme pouze s jednou vrstvou a v ní je pouze jedna třída.

4. 1. První spuštění

Poté, co je napsán a uložen mapfile, je možné se pustit do vlastní HTML stránky. Ta se uloží do souboru /var/www/mapserver/mapserv.html:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN" "http://www.w3.org/TR/REC-html40/loose.dtd">
<html lang="cs">
<head>
<meta http-equiv="Content-language" content="cs">
<meta http-equiv="Content-type" CONTENT="text/html; charset=iso-8859-2">

<meta name="resource-type" content="document">
<title>Moje první Mapserveří aplikace</title>
</head>
<body>
<h1>Budiš mapa</h1>
<div>
    <img border="1" src="/cgi-bin/mapserv?map=/var/mapservdata/jezera.map&layer=staty&mode=map" alt="Mapa" >

</div>
Ahoj, světe!
</body>
</html>
```

A nyní se již může nasměrovat webový prohlížeč na kýženou adresu http://localhost/mapserver/mapserv.html.

Když se obrázek nezdaří, je nutné zkusit nasměrovat browser přímo na adresu mapserveru. Objeví-li se hláška *msLoadMap(): Unable to access file. (/var/mapservdata/jezera.map)* nebo nějaká podobná, je zřejmé, že je problém s právy a nebo například s názvem adresáře.

Místo adresy s obrázkem je adresa programu mapserv s klasickým způsobem předání parametrů přes webové rozhraní. V tomto případě jsou parametry

map

Udávající plnou cestu k našemu mapfilu

layer

Vrstvy, které se mají vykreslit

mode

Mód, v jakém má mapserver fungovat. Mód map vrací obrázek mapy, query vrací výsledek dotazu na zadané souřadnice, o tomto módu ale až později, browse vrací vygenerovanou HTML stránku na základě šablony (viz dále).

Pokud šlo vše tak, jak má, zobrazí se vám první HTML stránka s on-line vytvořenou mapou.

5. Mapové projekce

Aby bylo možno geoid jménem Země nějak zobrazit v 2D rovině, byly postupem času navrženy tzv. kartografické projekce – způsoby, snažící se o tento převod s co možná nejmenším zkreslením. Každá projekce má jiný počátek, může mít i jiný směr hlavních os, velmi často je dělaná na míru konkrétnímu tzv. referenčnímu elipsoidu – matematickému modelu zemského tělesa. MapServer umí díky knihovně Proj převádět mapy v různých kartografických projekcích do jedné – vše v reálném čase. Díky této vlastnosti tak můžete mít zdrojové mapy v různých projekcích (pro Českou republiku na příklad S-JTKS, WGS84) a ještě je zobrazovat v projekci zcela jiné (na příklad Lat-Long).

Ale než se tak stane, musí se v mapfilu definovat, v jaké projekci jsou vstupní mapy a v jaké projekci má být mapa výsledná. Jsou v zásadě dvě možnosti: buď projekci definujeme „ručně”, jménem, elipsoidem a dalšími parametry, nebo vezmeme již projekci předdefinovanou číslem (kódem), což je náš případ. Jednotlivé projekce se svými kódy jsou uloženy v souboru epsg (v mém Debianu: /usr/share/proj/epsg). Tyto kódy jsou standardizovány European Petroleum Survey Group (EPSG).

Víme, že naše vstupní mapy jsou v projekci WGS 84, která má kód 4326 a rádi bychom, aby se mapy zobrazovaly v projekci US National Atlas Equal Area, s kódem 2163. Náš mapfile tedy pozměníme do následující podoby:

```
# počátek mapfilu
MAP
  NAME Jezera   # každý objekt by měl být pojmenován
  SIZE 400 300  # výchozí velikost mapy v pixelech
  # EXTENT -97.238976 41.619778 -82.122902 49.385620 # hraniční souřadnice
  EXTENT 208398.01 -372335.44 1285308.08 632638.93 # hraniční souřadnice v projekci US Nat. Atl.
  UNITS KILOMETERS  # mapové jednotky
  SHAPEPATH "/var/mapservdata/data/" # Cesta k datům
  IMAGECOLOR 255 255 255 # Barva pozadí
  IMAGETYPE PNG # typ výsledného obrázku
  PROJECTION
         "init=epsg:2163"
  END
############################################################
# začátek všech vrstev
############################################################

  LAYER # začátek vrstvy
    NAME        staty_plochy # jak se jmenuje
    DATA        states_ugl # v jakém souboru (souborech) se nachází
    TYPE        POLYGON # typ dat
    STATUS      OFF
    PROJECTION
          "init=epsg:4326"
    END
    CLASS # začátek třídy
        NAME 'Plochy států'
        STYLE # jak bude třída vypadat
            COLOR 225 225 185
        END # konec stylu
    END # konec tříd
  END # konec vrstvy

  LAYER # začátek vrstvy
    NAME        staty_hranice # jak se jmenuje
    DATA        states_ugl # v jakém souboru (souborech) se nachází
    TYPE        LINE # typ dat
    STATUS      OFF
    PROJECTION
          "init=epsg:4326"
    END
    CLASS # začátek třídy
        NAME 'Hranice států'
        STYLE # jak bude třída vypadat
            COLOR 128 128 128
        END # konec stylu
    END # konec tříd
  END # konec vrstvy
############################################################
# konec všech vrstev
############################################################
END # konec mapfilu
```

Co se změnilo? Předně byl zaveden nový objekt PROJECTION v objektu MAP a dále byl pro každou vrstvu (LAYER) také zaveden zvláštní objekt PROJECTION, definující projekci, ve které se pracuje. Dále byly upraveny hraniční souřadnice výsledné mapy (řádek EXTENT), respektující nový souřadný systém. Obnovíte-li stránku, uvidíte, že došlo i ke změně zobrazení (zkreslení) výsledné mapy – mapa je více „plochá”, ne tak „spláclá” – použitá projekce je tedy pro dané území vhodnější (což se ale nerozhoduje na základě „hezčí” mapy, ale na základě parametrů projekce).

Další věc, která se změnila, je řádek EXTENSION, kde se změnily hodnoty souřadnic (nová mapová projekce má jinam posunut počátek a další parametry). Souřadnice vypočítáme na příklad pomocí programu cs2cs (Coordinate system to coordinate system) z balíku knihovny Proj.

Máme tedy několik vrstev definovaných v mapfilu a přetransformovaných z jedné projekce do druhé.

6. Třídy podle atributu

Třídy je možno vytvářet na základě jejich atributů. V souboru obsahujícím státy states_ugl.shp, resp. v připojené databázové tabulce states_ugl.shp můžeme vidět sloupečky atributů:

```
$ ogrinfo -al -so states_ugl.shp
[...]
AREA: Real (12.3)
PERIMETER: Real (12.3)
STATESP020: Integer (11.0)
STATE: String (20.0)
STATE_FIPS: String (2.0)
CLASS: String (5.0)
```

Rozhodnutí je klasifikovat mapu na základě sloupečku class. Ten obsahuje buď slovo land nebo water. Specifikace sloupečku se děje atributem CLASSITEM, hledané výrazy jsou specifikovány atributem EXPRESSION, přičemž lze využívat i regurální výrazy:

```
LAYER # začátek vrstvy ploch států
  NAME         staty_plochy
  DATA         states_ugl
  TYPE         POLYGON
  STATUS       OFF
  CLASSITEM    'class'
  PROJECTION
       "init=epsg:4326"
  END

  CLASS # třída pevniny
      NAME 'Plochy států'
      EXPRESSION 'land'
      STYLE # jak bude třída vypadat
          COLOR 225 225 185
      END # konec stylu
  END # konec pevniny

  CLASS # třída jezer
      NAME 'Plochy jezer'
      EXPRESSION 'water'
      STYLE
          COLOR      198 198 255
        END
  END # konec jezer
END # konec vrstvy
```

Po reloadu stránky by se měla jezera objevit. Všechny textové řetězce se vyplatí uzavřít do uvozovek. Na pořadí vrstev záleží. Vrstvy se vykreslují tak, jak jsou v mapfilu psány – odshora dolů. Kdybychom vrstvy otočili,

```
LAYER
    NAME        staty_hranice
    ...
END
LAYER
    NAME        stat_plochy
    ...
END
```

nebudou hranice států vůbec vykresleny, ačkoliv budou zapnuty a vše bude korektně nastaveno.

6.1. Textové štítky v mapě

Důležitou funkcí jsou textové popisky v mapách. Musí být vyvedeny v určitém měřítku, nesmí se překrývat. Sloupeček v tabulce, ze kterého se mají štítky brát, se nastavuje atributem LABELITEM. Vlastní vzhled popisku je popsán v objektu LABEL, ukončeném END. Písmo může být buď bitmapové a nebo TrueType fonty. Používáme-li TrueType, musíme nastavit na začátku mapfilu cestu k seznamu aliasů k těmto fontům, tzv. fontlistu. Každý řádek v souboru má dvě klíčová slova: Alias, pomocí kterého se budeme na font odkazovat, a cesta/k/fontu.ttf. Cesta je buď relativní (vzhledem k fontlistu), nebo absolutní. Máte-li nějaký adresář s TrueType fonty, máte práci o to jednodušší. Pokud ne nebo chcete spíše využít nějaké jiné, můžete je vybrat na příklad z archivu Itasca-demo. Vytvořme tedy soubor font.list v adresáři:

```
cd /var/mapservdata
$EDITOR font.list

mono-bold-oblique       /usr/share/fonts/truetype/freefont/FreeMonoBoldOblique.ttf
mono-bold               /usr/share/fonts/truetype/freefont/FreeMonoBold.ttf
mono-oblique            /usr/share/fonts/truetype/freefont/FreeMonoOblique.ttf
mono                    /usr/share/fonts/truetype/freefont/FreeMono.ttf
sans-bold-oblique       /usr/share/fonts/truetype/freefont/FreeSansBoldOblique.ttf
sans-bold               /usr/share/fonts/truetype/freefont/FreeSansBold.ttf
sans-oblique            /usr/share/fonts/truetype/freefont/FreeSansOblique.ttf
sans                    /usr/share/fonts/truetype/freefont/FreeSans.ttf
serif-bold-italic       /usr/share/fonts/truetype/freefont/FreeSerifBoldItalic.ttf
serif-bold              /usr/share/fonts/truetype/freefont/FreeSerifBold.ttf
serif-italic            /usr/share/fonts/truetype/freefont/FreeSerifItalic.ttf
serif                   /usr/share/fonts/truetype/freefont/FreeSerif.ttf
```

Přidat řádek s atributem FONTSET do objektu MAP:

```
MAP
  NAME ITASCA   # každý objekt by měl být pojmenován
  SIZE 400 300  # výchozí velikost mapy v pixelech
  EXTENT 208398.01 -372335.44 1285308.08 632638.93 # hraniční souřadnice v projekci US Nat. Atl.
  UNITS METERS
  SHAPEPATH "/var/mapservdata/data/" # Cesta k datům
  IMAGECOLOR 255 255 255 # Barva pozadí
  IMAGETYPE PNG
  FONTSET '/var/mapservdata/font.list'
  [...]
```

A nyní doplnit do objektu CLASS objekt LABEL. Ukažme si ucelenější blok:

```
CLASS # třída pevniny
    [...]
    CLASSITEM   'class' # podle čeho dělit třídy
    LABELITEM   'state' # z jakého sloupečku brát popisky
    LABEL
        COLOR 132 31 31
        SHADOWCOLOR 218 218 218
        SHADOWSIZE 2 2
        TYPE TRUETYPE
        FONT arial-bold
        SIZE 12
        ANTIALIAS TRUE
        MINDISTANCE 300
        BUFFER 4
    END # end of label
```

Atributy jsou vesměs samovysvětlující. Ty méně jasné:

SIZE

Velikost písma se zadává buď v pixelech (v případě TrueType fontů), nebo klíčovými slovy (tiny,small,medium,large,giant).

MINDISTANCE

Udává minimální vzdálenost mezi opakovanými štítky (v pixelech).

BUFFER

V pixelech zadaný obvod okolo každého štítku tak, aby se tyto navzájem nekryly.

Štítky jsou vždy vykreslovány na novém místě tak, aby byly pokud možno uprostřed mapového objektu a aby se zároveň s okolními štítky nepřekrývaly.

6. 2. Přidávání rastrových vrstev

Doposud jednalo pouze s vektorovými soubory, uloženými ve formátu ESRI Shape. Rastrová vrstva je jako každá jiná:

```
LAYER # začátek družicového snímku
  NAME          modis
  DATA          'mod09a12003161_ugl_ll_idxa.tif'
  STATUS        OFF
  TYPE          RASTER
  OFFSITE       71 74 65
  PROJECTION
        "init=epsg:4326"
  END
END # konec družicového snímku.
```

Mapfile je zpracováván odshora dolů a tedy vrstvy definované později překryjí vrstvy definované dříve, proto je rastrová vrstva před oběma vrstvami vektorovými.

6. 3. Jak se MapServer ovládá

MapServer je CGI program, kam se nejlépe zadají parametry pomocí formulářových polí. Ta mohou být buď „zjevná” (tedy například typu text a dalších), nebo skrytá (typ hidden). Hodnoty v těchto polích jdou nastavit buď přímo – klikem myši nebo vstupem z klávesnice, nebo nepřímo, pomocí JavaScriptových funkcí. JavaScript celou webovou stránku na straně klienta pěkně rozhýbe: tlačítka se vyzdvihují a zamačkávají, po najetí myši se pěkně mění a proto se bez trochy JavaScriptu nelze obejít.

Ať tak či tak, po stisknutí tlačítka „Zobraz mapu” musí být všechna pole formuláře nastavena na potřebné hodnoty.

7. Mapové vrstvy

Pro ovládání aplikace je nutné nyní přidávat další formulářové prvky. Prvně přidáním ovládání pro to, jaké vrstvy se vlastně mají zobrazovat a jaké ne. Jako všechno ostatní i vrstvy se ovládají pomocí políček formuláře:

```
[...]
<td>
    <!-- vrstvy -->
    <input name="layer" value="modis" type="checkbox" [modis_check]>Družicový snímek<br>
    <input name="layer" value="staty_plochy" type="checkbox" [staty_plochy_check]>Plochy států<br>
    <input name="layer" value="staty_hranice" type="checkbox" [staty_hranice_check]>Hranice států<br>

    <!-- /vrstvy -->
</td>
<td>
    <!-- obrázek s mapou -->
    <input type="image" name="img" src="[img]" width="400" height="300" border="0">
    <!-- /obrázek s mapou -->
</td>

[...]
```

Změnit status vrstvy modis na OFF a reload stránky.

7. 1. Legenda

Máme dvě možnosti, jak vytvořit legendu: buď jako obrázek nebo pomocí HTML.

Obrázek

Obrázek je snazší možnost, ale výsledek není moc hezký. Vyrobí se prostě tak, že se do šablony vloží následující obrázek.

```
<img name="legend" src="[legend]">
```

a do mapfilu objekt LEGEND (třeba za WEB):

```
# Legenda
LEGEND
    KEYSIZE 12 12
    LABEL
      TYPE BITMAP
      SIZE MEDIUM
      COLOR 0 0 89
    END
    STATUS ON
END
```

HTML legenda

HTML legenda je zřejmě lepší, protože se dá formátovat například pomocí CSS spolu se zbytkem stránky. Výroba je komplikovanější, protože je nutné pro ni vyrobit šablonu. Do mapfilu, do objektu LEGEND, se musí vložit ještě jeden řádek:

```
# Legenda
LEGEND
    KEYSIZE 12 12
    STATUS ON
    LABEL
      TYPE BITMAP
      SIZE MEDIUM
      COLOR 0 0 89
    END
    TEMPLATE "/var/mapservdata/tmpls/legenda.html"
END
```

A dále se musí tento soubor vytvořit.

```
[leg_class_html]
<tr>
  <td>
      <img src="[leg_icon width=20 height=10]" width=20 height=10>
  </td>
  <td>

      [leg_class_name]
  </td>
</tr>
[/leg_class_html]
```

Do HTML šablony vložíme další klíčové slovo:

```
<!-- /vrstvy -->
<!-- legenda -->
<hr>

<table border=0 name="legend">
    [legend]
</table>
<!-- /legenda -->
```

Nyní stačí už jen reloadovat stránku a legenda bude generována automaticky.

7. 2. Referenční mapa

Je to malá mapička s vyobrazenou aktuální oblastí, je nezbytnou součástí každé aplikace. Uživateli poskytuje přehled a může sloužit jako rychlá navigace po území. Pro její přidání se musí nejdříve přidat patřičné řádky do mapfilu a pak patřičné řádky do HTML šablony. Před vlastní editací mapfilu a šablony je nutné vytvořit obrázek referenční mapky a vědět její hraniční souřadnice. Obrázek uložit do adresáře /var/mapservdata/tmpls/ s názvem refmap.png. Hraniční souřadnice této mapy převezmeme z celkového území.

7. 3. Měřítko

Měřítko mapy je další nepostradatelná věc, která by neměla chybět. V podstatě jsou znovu dvě možnosti, jak jej vyjádřit: pomocí číselného údaje (například 1:10 000), nebo pomocí grafického měřítka (jakéhosi pravítka) na mapě (nebo vedle ní).

První (textový) příklad je prostě vložením tagu [scale] do šablony stránky, například:

```
<!-- /vrstvy -->
<!-- legenda -->
<hr>

<table border=0 name="legend">
    [legend]
</table>
<!-- /legenda -->
```

Grafické pravítko se pak definuje v mapfilu:

```
<!-- obrázek s mapou -->
1:[scale]<br>
<input type="image" name="img" src="[img]" width="400" height="300" border="0">
<!-- /obrázek s mapou -->
```

Objekt SCALEBAR může být definován několika atributy. Parametr STATUS, který je nastaven na hodnotu EMBED – což znamená, že pravítko bude integrováno do výsledného obrázku s mapou.

7. 4. Vylepšení funkcí pomocí Java appletu

Při tvorbě nějaké webové stránky, je důležité, aby byla tato stránka a její funkce přístupná pro co největší počet potencionálních čtenářů – bez ohledu na platformu nebo browser, ve kterém se na stránku dívají. Java applety rozhodně nepatří mezi technologie, které by byly z tohoto pohledu bezproblémové.

Applet jBox ovšem patří mezi programy, které svou přidanou hodnotou bohatě převyšují své případné nedostatky. jBox umožňuje „hezčí" funkci zoom (zvětšování), unzoom a pan (posun), stejně jako některé další funkce, které by pomocí DHTML šly naprogramovat jen těžko.

7. 5. Měření vzdáleností a ploch

Díky appletu se nechají i měřit plochy a vzdálenosti pomocí myši. Opět se musí doplnit některé javascriptové funkce, aby byl výsledek hezčí a výsledky se budou zobrazovat on-line.

Funkce vytvoří proměnnou defaultStatus, do které uloží textový řetězec a nakonec nastaví hodnotu divu divConsole na tento textový řetězec. Tento div musíme ještě vytvořit:

```
<!-- /obrázek s mapou -->
<div name="divConsole">
 
<br>
 
</div>
```

Po vytvoření lomené linie o 3 bodech a více se začne počítat i plocha. K ověření přesnosti můžete přeměřit měřítko mapy.

7. 6. Učesání některých číselných hodnot pomocí Java Scriptu

Některé hodnoty (měřítko, celková plocha) jsou zobrazovány s nesmyslnou přesností, což nepůsobí moc „pěkně". Tyto hodnoty je dobré zaokrouhlit opět pomocí Java Scriptu pomocí funkce Math.round(x).

Stejným způsobem upravit i textové měřítko mapy, které se zatím řešilo pouze tagem 1:[scale]. Výsledné měřítko se sice nebude zobrazovat úplně přesně, ale bude rozhodně přehlednější:

```
[...]
<!-- obrázek s mapou -->
1:<script language="JavaScript">document.write(Math.round([scale]/100)*100);</script><br>
[...]
```

Závěr

Snahou této práce bylo ukázat nástin toho, o čem může být, napsat takovou jednoduchou aplikaci na zobrazování map a také jsem se pokusil upozornit na zajímavé funkce a pomocné programy, jejichž implementací se funkcionalita aplikace zvýší. Nebylo zde zmíněno vyhledávání v mapě a databázi a možnosti spojení s programovacími jazyky jako je např. Perl nebo Python. Tato spojení jsou velmi dobře popsána v různé dokumentaci či k nalezení na internetu.

Za povšimnutí určitě také stálo vysvětlení významu použitých zkratek či slov, s jejichž významem nemusí být zcela všichni plně seznámeni. Samozřejmě, že tuto práci bylo možné zaměřit zcela na jiný cíl, kterých je v této problematice vícero.

Touto prací jsem chtěl popsat a vysvětlit významy jako např. Open source, GIS, mapový server a přiblížit problematiku programování mapového serveru. Na tomto místě bych také velice rád poděkoval vedoucímu této práce (pan Ing. Michal Šerý) za cenné rady a připomínky k této práci.

Seznam použité literatury

[1] Tuček., GIS Geografické informační systémy, Principy a praxe, Computer Press, 1998, 424p.

[2] Anonymus: Pracujeme s geografickým informačním systémem ArcView GIS, Computer press 199, 364p.

[3] Burrough P.A. & McDonnel R.A. (1998): Principles of Geographical Information Systéme. Oxford, 333 p.

[4] Pauknerová. Er Kučera T. (1997): Informační zdroje pro využití nástrojů GIS v ochraně přírody a krajiny – Ed. AOPK ČR, 92 p.

[5]Aplikace moderních metod operační analýzy v prostředí GIS = Application of modern operational analysis methods in the environment GIS / Dušan Janoštík . Brno : Vysoké učení technické v Brně, c2004. 28 s. Sg.

[6]Open Source - vývoj webových aplikací : Linux, Apache, MySQL, Perl a PHP / James Lee, Brent Ware . Praha : Mobil Media, 2003. 448 s. Sg.

[7] http://www.opensource.org/docs/definition.php

[8] http://www.oss.cz/aktuality/news_item.2008-03-17.9843366175

[9] http://www.pooh.cz/aktovka-x/a.asp?a=2006553

[10] http://cs.wikipedia.org/wiki/Hlavn%C3%AD_strana

[11] http://www.esri.com/software/arcgis/arcgisserver/index.html

[12] http://opensource.gene.cz/osy.html

[13] http://www.gis.com/showcase/index.html

[14] http://www.pardubickykraj.cz/index.asp?thema=3464&category=

Anotace

Cílem této bakalářské práce bylo popsat problematiku spojenou s Open source softwarem a mapových serverů. Práce se zabývá vysvětlením základních pojmů, které se týkají již zmíněné problematiky a nástinem a postupem vývoje mapového serveru pro téměř kompletní použití.

Annotation

Purposes those bachelor work was describe problems connection with Open source software and map server. Work describes explication basic notions that the teak already mentioned problems and outline and progress development charted server for almost full using.