Vyšší odborná škola a obchodní akademie, s. r. o.
Pražská 3, České Budějovice

Univerzální systém pro webovou prezentaci firmy s využitím technologie PHP

ABSOLVENTSKÁ PRÁCE

Vedoucí absolventské práce: PaedDr. Petr Pexa

Autor: Ladislav Vacek

České Budějovice, 2005

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně, a že jsem veškerou použitou literaturu uvedl v Seznamu použité literatury.

Ladislav Vacek

Obsah

1 Úvod

Ve své absolventské práci se zabývám zviditelněním webu na internetu. V první části se zabývám optimalizací webu pro vyhledavače a ve druhé přepisováním URL adres pomocí Mod_Rewrite. Práce je určena webmasterům a lidem, kteří znají a zajímají se o tuto problematiku, která úzce souvisí s tvorbou webových stránek.

Internetové vyhledávače se v poslední době stávají stále častějším způsobem hledání informací. Téměř každý, kdo pracuje s internetem, nějaký vyhledávač použil a každý webmaster, případně majitel webových stránek nejednou uvažoval o tom, jak si zajistit lepší pozici ve vyhledavačích. Dosažení prvních míst lze dosáhnout vytvořením takového webu, který bude vstřícný k robotům vyhledávačů. Aby robot vyhledávače správně zaindexoval webové stránky, musí splňovat některé pravidla. Nesmějí se používat technologie, které nedokážou roboti přečíst, musíme zajistit správný výběr klíčových slov, zajištění validity stránek, vytvoření zpětných odkazů. Ještě do nedávna byla optimalizace pro vyhledávače v České Republice velkou neznámou. Nyní nastal čas, kdy je nutné web optimalizovat, aby se dostal na přední místa vyhledávačů.

Jedna z důležitých pravidel SEO je mít hezkou URL adresu stánek. URL by se měla tvářit jako statická stránka a obsahovat klíčová slova. Statické stránky robot snadněji zaindexuje. Proto, aby se web tvářil jako statický a přitom byl dynamický nám slouží mod_rewrite. Je to modul Apache, který přepisuje URL adresy. Popisuji jak mod_rewrite funguje a jeho direktivy. Jelikož je mod_rewrite poměrně složitý, předpokládají se u čtenáře alespoň základní znalosti webového serveru.

V praxi jsem tyto znalosti aplikoval na internetovém serveru pyramida-plzen.cz.

2 SEO - Search Engine Optimization

Tím jak zviditelnit webovou stránku se zabývá Search Engine Marketing - SEM, který se skládá ze dvou částí, placené a neplacené. V placené části se platí za zviditelnění přímo vyhledávači. Neplacená část se nazývá Search Engine Optimization - SEO (optimalizace pro vyhledávače). V SEO části hlavně záleží na tom, jak jsou stránky naprogramované, zda mají kvalitní obsah a vhodná klíčová slova. Dalším důležitým faktorem jsou zpětné odkazy, které nám zvyšují Rank stránek, díky němuž je web ve výsledcích vyhledávání na předních místech.

2.1 Předpoklady pro optimalizaci

Roboti fulltextových vyhledávačů vidí stránky přibližně stejně jako textový prohlížeč Lynx, takže v případě použití některých technologií jako je například Javascript, Flash, Frame, nedokáží zaindexovat celý web. Ne všichni roboti mají stejné problémy, některé dokáží indexovat dynamické URL, ale ve Flashi nedokáží rozpoznat odkazy, jiné zase naopak. Každý robot pracuje na jiné bázi. Pro kontrolu můžeme zadat příkaz:

```
site:example.com example
```

a porovnáváme počet zaindexovaných stránek s celkovým počtem stránek daného webu.

2.1.1 Javascript

Javascript je programovací jazyk, který se používá na webových stránkách, za účelem efektivity webu na straně uživatele. Javascript není příliš oblíben, jelikož v každém prohlížeči reaguje trochu jinak. Roboti Javascript velmi často

neindexují. Pokud je navigace webu provedena pomocí Javascriptu, roboti nedokáží odkazy najít a zaindexují pouze úvodní stránku.

Jsou ale situace, kdy se Javascript dá efektivně použít, například pro kontrolu polí odesílaného formuláře. Pokud ho použijeme, je vhodné ho schovat do souboru.

<script type=”text/javascript” src=”x.js”></script>

Dělá se to za účelem přehlednosti a větší čitelnost pro roboty. Pokud by bylo Javascriptu na stránce opravdu hodně, nemusel by robot stránku vůbec zaindexovat.

2.1.2 Flash

Flash je aplikace vyvinutá firmou Macromedia. Jedná se o objekt umístěný na webové stránce. Je vhodný pro jednoduché animace, reklamu. Hlavním cílem je zaujmout uživatele.

Flash je pro optimalizaci nepoužitelný, jelikož ho roboti většinou neumí číst, ale například robot Google ho dokáže už částečně přečíst. Vše je teprve ve fázi vývoje. Flash nelze použít pro navigaci, jelikož robot vyhledávače tak zaindexuje pouze první stranu a ostatní stránky již nevidí. Když už je tedy na stránce Flash, je nutné ho doplnit textovými informacemi, popřípadě udělat neflashovou verzi.

2.1.3 Splash page

Splash page je vstupní stránka, která obsahuje buďto veliký obrázek nebo flashovou animaci. Po jejím ukončení často přichází přesměrování.

Někteří roboti vyhledavačů takové stránky jednoduše označí za spam a odmítnou je zaindexovat, protože nemají dostatek kvalitního obsahu. Pokud web obsahuje Splash page , měla by být u Flashe možnost přeskočit intro a ve spodní části stránky odkazy na hlavní část webu.

2.1.4 Rámce

Rámce se používají k rozvrhnutí stránky. Vedle tabulek a CSS je to základní možnost jak stránku rozvrhnout

Hlavní problém rámců je s vyhledávacími roboty. Pro orboty jsou rámce velký problém, odmítají je indexovat a zaindexují pouze část noframes. Pokud robot zaindexuje obsah jednotlivých stránek v rámcích, je problém při zobrazení uživateli. Nabídnou mu totiž odkaz na konkrétní stránku a ne na rámce.

Pak se většinou zobrazí stránka bez navigačního menu, běžný uživatel neví jak pokračovat, a ve většině případů opustí web. Pokud chce někdo odkázat na konkrétní stránku webu, tak to v podstatě nejde, nebo jde, ale pak nelze vidět příslušné menu.

Pokud se rozhodneme rámce použít, je nutné optimalizovat alespoň obsah tagu noframes. Ten se pak ukazuje ve vyhledávačích.

2.1.5 Klikací obrázkové mapy

Klikací obrázková mapa je obrázek rozčleněný v HTML na menší části. Každá taková část může odkazovat na jinou stránku. Používá se proto, aby se obrázek nemusel v grafickém programu pracně rozdělovat.

Ne všechny vyhladavače si s tím dokáží poradit a proto je vhodné, aby na stránce, kde se odkazuje v obrázkové mapě, byl ještě jeden textový odkaz.

2.1.6 Problém s dynamickými URL

Dynamická URL obsahuje otazník, za kterým si script předává parametry. Například:

```
www.example.com/clanek.php?clanek=1765
```

Vyhledavače většinou neindexují stránky s mnoha parametry. Například Google indexuje maximálně tři proměnné, ale některé menší vyhledavače to nedokáží vůbec. Pro uživatele, ale i pro vyhledavače je jednodušší zapamatovat si statickou URL stránky. Příkladem takového URL může být:

```
www.example.com/nadpis-clanku.html
```

Pro vylepšení URL můžeme použít mod_rewrite. Tato problematika je popsána v druhé části práce.

2.1.7 Časté změny v URL

URL adresy by měly být co nejvíce stabilní. Je nutné je navrhnout tak, aby se nemusely dlouhou dobu (pokud možno nikdy) předělávat. Jestliže dnes někdo odkáže na článek na určitém webu, měla by tato URL adresa fungovat i za tři roky. To je hlavně z důvodu uživatelů ne kvůli SEO. Pokud se již z nějakého důvodu na webu předělávají URL, například kvůli převodu dynamických stránek pomocí mod_rewrite na statické, je potřeba původní URL přesměrovat na novou. V žádném případě nesmí původní URL přestat fungovat.

Než se změny projeví ve vyhledávačích, může to trvat i půl roku, a nefunkční odkazy z jiných stránek se nemusí spravit nikdy. Menším problémem je, ponechat funkční obě verze, ale to pak vede k duplicitnímu

obsahu. Správné přesměrování je potřeba provést pomocí http hlavičky s kódem 301 – Moved Permanently. Po přesměrování pomocí kódu 301 by měly roboti zaindexovat novou adresu. V PHP se to dělá následujícím způsobem:

```
<?
header("HTTP/1.1 301 Moved Permanently");
header("Location: http://www.example.com/");
header("Connection: close");
?>
```

Při přesměrování se nesmí vynechávat první řádek, tedy kód 301 Moved Permanently, jinak by server nahlásil místo toho 302 Found (moved temporarily), ač to na první pohled vypadá stejně, pro robot to znamená, že původní stránku našel.

Pokud není možné řešení pomocí http hlaviček na straně serveru, lze použít v krajním případě meta tagu:

```
<meta http-equiv="refresh" content="10;url=nove-url">
(mělo by být delší než 1 sekunda)
```

Samozřejmě, že takové přesměrování nemusí vždy fungovat, takže je nutné na takovou stránku umístit i odkaz na novou URL adresu.

2.1.8 Duplicitní URL

Pro vyhledávače jsou URL adresy:

```
http://www.example.com/
http://www.example.com/index.html
http://example.com
http://example.com/index.html
```

čtyři různé stránky. Z toho důvodu je nutné důsledně dodržovat při odkazování na stránku jednu verzi, jinak dochází zbytečně k rozdělování PageRanku na duplicitní stránky. Stejně tak www.example.com/neco je jiná stránka než www.example.com/neco/. Vyhledávače vždy zobrazí pouze jednu verzi stránky, která je duplicitní. Měla by to být ta, na kterou se nejčastěji odkazuje. Ostatní jsou ve výsledcích vynechány.

2.1.9 Validita HTML kódu

Současné webové prohlížeče dokáží zobrazit HTML kód i s hodně chybami. Když chybí ukončení značky, snaží se ji vhodně doplnit. Robot vyhledávače by to měl zvládnout také, ale může se stát, že při neukončené značce může zaměnit text za HTML značku. Aby se zabránilo takovéto chybě, používají se pro kontrolu validátory, které nám případné chyby odhalí.

2.1.10 Velikost stránky

Velikost stránky není důležitá jen pro uživatele, kteří mají pomalé připojení na internet, ale i pro roboty vyhledávačů. Pokud velikost přesáhne určitou hodnotu, mohou stránku přestat stahovat a zaindexovat pouze stáhnutou část. Velikost lze snížit používáním CSS stylů, umístěním Javascriptu do externího souboru, nahrazením obrázků texty, atd.

2.2 Klíčová slova

Základem optimalizace je výběr klíčových slov a jejich rozmístění na stránce. Pokud na stránce není určité slovo, tak ji pod tímto slovem nemůže vyhledávač ani najít.

Naprostý základ optimalizace je, že každá stránka na optimalizovaném webu musí být unikátní. Vyhledávače hodnotí každou stránku zvlášť. Proto je nutné soustředit se na všechny stánky a nejenom na úvodní stranu.

2.2.1 Analýza klíčových slov

Analýza klíčových slov má dvě fáze. V první fázi se snažíme získat co nejvíce klíčových slov, na které bysme mohli web optimalizovat. Cílem je, abysme nezapomněli žádné slovo, které by mohlo být důležité. Není důležité snažit se optimalizovat pro nejvíce lukrativní slova, ale vyhledat více synonym v daném oboru. Protože optimalizovat na "velké fráze" typu "Ubytování Praha" nebo "Auto Škoda" je velice těžké a pro nový web by se to zřejmě ani nevyplatilo. Při výběru klíčových slov, nesmíme zapomínat na skloňování a na množné čísla, ne všechny vyhledávače je umí odvodit. Základní pravidlo klíčových slov je, že musíme hledat relevantní slova k danému webu.

Tato analýza se skládá ze dvou částí. Vertikální a laterální analýzy. Vertikální analýza dodá různé tvary daného slova (např. ze slova "bazén" udělá "bazénové", "bazénová technika", "bazénu" "bazénem" atd.). Laterální analýza naopak vyhledá taková slova, která vyhledávají lidé, kteří hledají něco týkající se slova "bazén" (např. se vám objeví "filtrace", "koupání", ale i "čištění" nebo "zastřešení").

V druhé fázi naopak z těch často stovek vybraných slovních spojení hledáte právě ta, na která se vám vyplatí optimalizovat.

2.2.2 Stop slova

Stop slova jsou běžná slova, které nenesou žádnou nebo téměř žádnou informaci. Jedná se většinou o spojky, předložky atd. V češtině se jedná hlavně o a, i, nebo, když, v, na, pod. Vyhledávače dost často těmto slovům přikládají nižší váhu nebo je úplně ignorují. Dělají to kvůli zrychlení hledání. Je tedy téměř zbytečné dávat stop slova do Title, description nebo keyworks.

2.2.3 Nevhodná slova

Jedná se o slova, která snižují hodnotu příslušné stránky na určitý dotaz. Pro uživatele, který nechce obdchod, ale stránku s recenzí produktu je nevhodným slovem například koupit, cena, prodej. Nevhodná slova na určité téma vadí pouze v titulku nebo v URL.

Zvláštní kategorií jsou nevhodná slova spojená se sexem. Obecně se předpokládá, že pokud uživatel něco hledá nechce stránky se sexem, takže na ně jsou často uvaleny speciální filtry, které stránku úplně vyřazují nebo minimálně snižují hodnocení. Samozřejmě filtr se neuplatní v případě, že uživatel hledá přímo tato nevhodná slova.

2.2.4 Hustota klíčových slov v textu stránky

Vyhledavače neurčují důležitost klíčového slova na stránce podle jeho četnosti, ale podle jeho hustoty výskytu klíčového slova. Hustota na stránce tedy vyjadřuje podíl jeho četnosti (počtu výskytů) ku celkovému počtu slov celého textu. Pro klíčové slovo se za optimální většinou považuje hustota mezi 2 až 7%. Pokud hustota převýší určitou hodnotu (pro každý vyhledavač jinou) může to mít záporný efekt. Hustota klíčových slov bývá často přeceňována, zdaleka však nepatří mezi nejdůležitější kritéria. Optimální počet slov na

stránce můžeme zjistit tak, že začneme na malé hustotě (2%) a postupně zvyšujeme hustotu. Po přidání jednoho slova je nutné počkat až se změna projeví v testovaném vyhledavači. Pokud se stránka posouvá nahoru, lze hustotu dále zvyšovat. Bohužel existuje celá řada dalších vlivů (změny na ostatních stránkách), které není možné úplně odstínit.

2.2.5 Umístění klíčových slov

Velmi důležité je, kde je slovo na stránce umístěno. Podle důležitosti je to pravděpodobně:

- title,
- description
- keywords
- ostatní meta značky
- nadpisy (H1, H2...)
- začátek stránky
- tvar URL dané stránky
- ALTy u obrázků
- text samotné stránky
- používání tagů   a

Důležitost se může u jednotlivých vyhledávačů lišit a to i v průběhu času.

Title

```
<title>Titulek stránky</title>
```

Jednoznačně nejdůležitější tag na stránce je Title. Je v podstatě jediný, který má význam ve všech vyhledavačích. Každá stránka na konkrétním webu by měla mít jiný Title, který nejlépe popisuje obsah stránky a obsahuje

vyhledávané slovo. Ač by se zdálo, že je to nejjednodušší pravidlo, velké množství webmásterů to nebere na vědomí.

Jako Title je tedy nutné například používat název firmy a klíčové slovo. Na místo na příklad Pyramida Plzeň je dobré použít Pyramida Plzeň – bazény, sauny nebo něco podobného. Doporučená délka Title je do 70ti znaků, což je přibližný počet znaků, které se zobrazují ve výsledcích vyhlédávání.

Meta desctription

```
<meta name="description" content="popis stránky">
```

Meta tag description některé vyhledávače zobrazují u popisku stránky ve výsledcích vyhledávání, takže se vyplatí zde napsat něco smysluplného. Desctription používá většina vyhledávačů. Stejně jako u titulku je důležité,aby u každé stránky bylo description, které ji nejlépe vystihuje, tedy pro každou stránku odlišné. Doporučená délka je do 250ti znaků.

Keywords

```
<meta name="keywords" content="klíčová slova">
```

Meta tag keywords již tak jednoznačný není, většina vyhledávačů ho nepoužívá. To ovšem neznamená, že je na škodu ho vyplnit klíčovými slovy. Opět každá stránka potřebuje vlastní keywords, stejně Title a descriptiva.

Pokud se klíčové slovo vyskytuje pouze v meta tagu description nebo keywords a není již ve vlastním textu, pak většinou vyhledávač danou stránku na toto slovo nenajde. Je tedy zbytečné vkládat do těchto meta tagů jiná slova než jsou v textu. To platí i o překlepech nebo o psaní slov bez háčků a čárek, pokud to není jinde než v meta tazích je to ztráta času.

Nadpisy H1 - H6

```
<h1>Nadpis stránky</h1>
```

Pokud je něco v nadpisu, mělo by to mít logicky větší váhu. Platí, čím je delší tag H1, tím má klíčové slovo v něm menší význam. Váha je samozřejmě největší u H1, není nutné hledat využití pro H4 a níže, jejich je logicky nižší. H1 se smí na stránce opakovat pouze jednou, ostatní nadpisy vícekrát.

Klíčová slova v URL

Většina vyhledavačů přikládá URL hodně velký význam. Je tedy vždy užitečné mít klíčové slovo v URL. Větší význam mívá doménové jméno než zbytek URL. Ve zbytku URL je užitečné mít klíčová slova, které se oddělují znaky – (mínus) a / (lomítko). Podtržítko (_) slouží jako spojovací znak. Vyhledavač čte prodejnu_bazenu jako prodejnubazenu

Popisky u obrázků

```
<img src="obrazek.gif" alt="obsah obrázku" title="shrnutí obsahu">
```

U každého obrázku by měl být vyplněný atribut alt, který se používá k zastoupení obsahu obrázku. Existuje celá řada uživatelů, kteří mají na modemu obrázky vypnuté. Robot vyhledávače neumí přečíst. Volitelný je atribut title, který se ukazuje když se na chvíli zastaví myší nad obrázkem. Je trošku méně významný než alt a měl by obsahovat obecné shrnutí obsahu obrázku.

U obrázků o velikosti 1x, které slouží pouze ke grafickým účelům je nutné nechat atribut alt prázný. Použití klíčového slova u takového obrázku by mohl vyhledavač shledat jako spam.

Text samotné stránky

Obsah je na webu to nejdůležitější. Většinou platí, že čím kvalitnější obsah tím méně optimalizace je potřeba. Protože na stránku s kvalitním obsahem častěji lidé odkazují. Obsah se píše vždy pro uživatele a ne pro vyhledavače, měl by tak být co nejvíce přirozený.

Čím více kvalitního a aktuálního obsahu na webu je, tím více uživatelů z vyhledavačů může získat. Web, který má deset stránek s kvalitním obsahem může získat například třicet kliků z vyhledavačů denně. Web, který má tisíc stránek s kvalitním obsahem může získat třista a více kliků za den.

Tučný text a kurzíva

```
<strong>tučný text</strong><em>kurzíva</em>
```

Doporučuje se mít na stránce alespoň jednou klíčové slovo tučně a jednou kurzívou. Velký význam to ale pravděpodobně nemá, sledují to jen některé vyhledavače. Hustota tučného textu a kurzívy na stránce by měla být taková, aby byla co největší přehlednost textu.

2.3 Zpětné odkazy

Pro budování zpětných odkazů můžeme použít následujících pět metod.

2.3.1 Registrace v katalozích

Jeli zcela nový web, své první zpětné odkazy získáme z internetových katalogů. České stránky by se měli vždy zaregistrovat do katalogů Seznam, Centrum, Atlas a ODP.

Neměli by jsme zapomenout na specializované oborové katalogy, ze kterých může chodit návštěvníků také hodně, a navíc dobře zacílených.

2.3.2 Kvalitní obsah

Nejcennější zpětné odkazy jsou ty, které vznikají sami od sebe. Někomu se zalíbí obsah některé stránky a spontánně na ni odkáže na svém webu. Atraktivní obsah přitom nemusejí tvořit jen zveřejněné informace. Zpětné odkazy přitahují i různé praktické webové aplikace odpovídající vašemu oboru, třeba počasí ve světě a převody měn pro cestování, či kalkulátor tepelných ztrát pro topenáře.

2.3.3 Výměna odkazů

Zpětné odkazy se budují i vzájemnou dohodou mezi majiteli webů. Najdeme cizí stránky, které mají tematicky příbuzný obsah, ale přímo nekonkurují. Pak na ně odkážeme a zároveň poprosíme jejich webmastera o odkaz na vlastní web.

2.3.4 Publikování na cizích serverech

Velmi cenné odkazy se můžeme získat z oborových publikačních serverů či z online verzí klasických odborných periodik. Stačí, když občas napíšeme zajímavý článek a nabídneme vhodnému magazínu k vydání.

2.3.5 Účast v diskusních fórech a konferencích

Na internetu je mnoho diskusních fór a konferencí, na kterých miliony lidí diskutují. Nějaké jistě existují i v daném oboru. Příspěvky musí být opravdu hodnotné, jinak by to mohlo mít opačný efekt. Je nutné nezapomenout se podepsat, včetně připojení jednoho souvisejícího odkazu.

2.4 Hodnocení stránek - Rank

Rank je číselná relativní hodnota důležitosti webových stránek. Každý vyhledavač používá k hodnocení stránek vlastní technologií Rank, která je ovlivněná více než stem faktorů. Hodnocení Rank, používá většina vyhledávačů, Google má PageRank, Seznam S-Rank, Jyxo JyxoRank. Každá jednotlivá stránka má svůj Rank. Jeho přesný výpočet není znám a pro větší objektivitu se často mění. Všeobecně záleží na počtu stránek, které na daný web odkazují, na velikosti jejich Rank a počtu odkazů, které obsahují. Pokud hodně kvalitních webů odkazuje na určitou stránku, bude i ona stránka pravděpodobně něčím zajímavá. Rank stránky není veřejný, nelze ho tedy přesně zjistit. Orientačně ho lze zjistit pomocí Toolbar daného vyhledavače, pokud ho má. Všeobecně tedy platí, že čím větší je hodnota Rank, tím výše se web zobrazuje ve vyhledavačích na dané klíčové slovo.

2.5 Výsledky SEO

Toto byly pouze základní pravidla, které by se měli dodržovat na každém nově vznikajícím webu. Nelze nikdy stoprocentně zaručit, že když web podle těchto pravidel vytvoříme, že budeme na prvních místech ve vyhledávačích. Záleží ještě na dalších okolních faktorech jako je konkurence, placené přednostní výpisy, atd.. Pokud se ale bude pravidel držet máme velkou šanci se na přední místa dostat.

3 mod_rewrite

Pomocí mod_rewrite můžeme vylepšit odkazy na webové stránce tak, aby byly srozumitelnější pro návštěvníky a vstřícnější k vyhledavačům. Mod_rewrite je modul Apache, který slouží k překladu URL adres.

3.1 Instalace mod_rewrite

Mod_rewrite je standardní modul obsažený v distribuci Apache, takže by měl být k dispozici bez instalace. Aby mohl být modul používán, musí se před kompilací Apache zadat volby -enable-shared-most, -enable-shared-all nebo -enable-rewrite. Dále musí soubor httpd.conf obsahovat:

```
LoadModule rewrite_module libexec/mod_rewrite.so
```

3.2 Jak mod_rewrite funguje

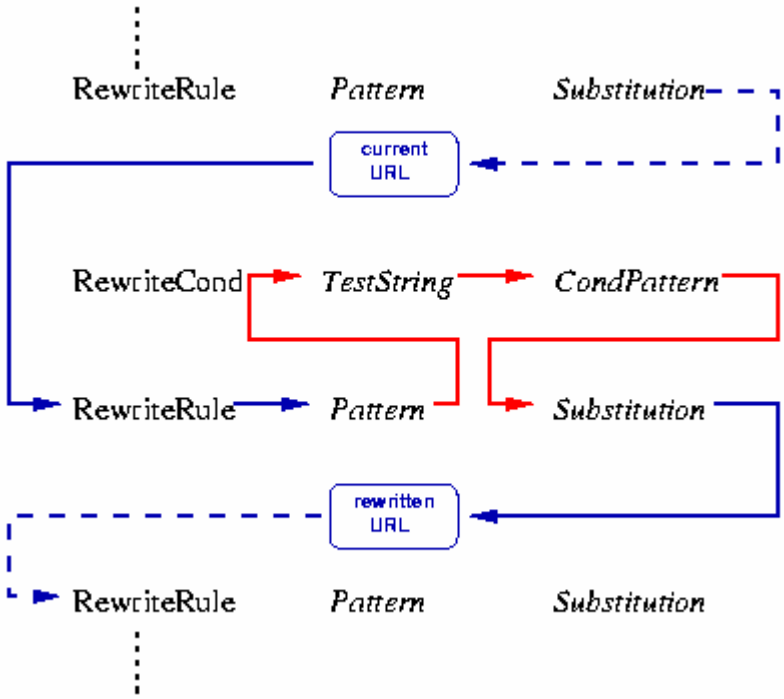
Mod_rewrite se ovládá pomocí souboru .htaccess, který je umístěn v příslušném adresáři. V souboru jsou napsány direktivy a pravidla pro přepisování.

```
RewriteEngine on
RewriteRule ^shop/item/(.*)\.htm$ index.php?item=$1
[L,QSA]
```

Při každém požadavku na nějakou stránku na serveru zkontroluje její adresu podle seznamu svých pravidel a pokud požadovaná adresa v seznamu je, aplikuje na ni konkrétní přepisovací pravidlo.

3.2.1 Zpracování sady pravidel

Pořadí pravidel v sadě pravidel je velmi důležité, protože engine přepisující URL je zpracovává podle speciálního pořadí. Engine předpisující URL provádí cyklus, který čte pravidlo po pravidle v sadě pravidel, dokud nenajde konkrétní pravidlo, které souhlasí s odpovídajícími podmínkami RewriteCond.



Jak můžeme vidět, nejdříve je URL spojeno se vzorem pro každé pravidlo. Pokud vzor nesouhlasí, mod_rewrite okamžitě ukončí zpracování pravidla a pokračuje s dalším pravidlem. Pokud vzor souhlasí, mod_rewrite porovnává odpovídající podmínky. Pokud žádné neexistují, tak jen nahradí URL s novou hodnotou, která se vytvoří z řetězce Substitution a pokračuje dalším pravidlem. Ale pokud podmínky existují, začne vnitřní cyklus, zpracovávající podmínky v pořadí v jakém jsou uvedeny. Pro podmínky je postup odlišný. Neporovnává se vzor s URL. Místo toho nejdříve vytvoří řetězec TestString z rozšířených

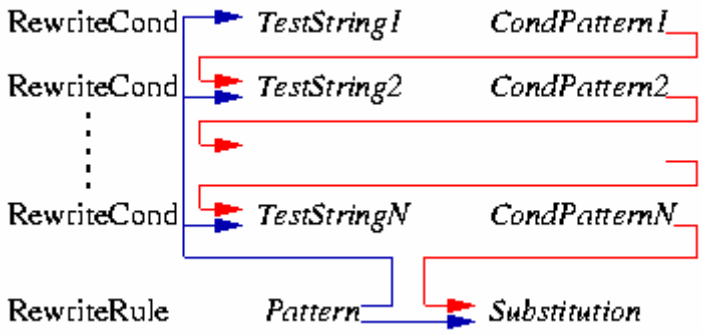
proměnných, zpětných referencích, atd. a pak ho zkusí porovnat se vzorem CondPattern. Pokud tento vzor nesouhlasí, potom všechny podmínky a odpovídající pravidla nebudou souhlasit. Pokud tento vzor souhlasí, další podmínka je zpracována dokud již žádná další podmínka není k dispozici. Pokud všechny podmínky souhlasí, zpracování pokračuje se substitucí URL s Substitution.

3.2.2 Psaní speciálních znaků

Od Apache 1.3.20, speciální znaky v řetězcích TestString a Substitution mohou být uvozeny zpětným lomítkem ('\') (to znamená, že se chovají jako ostatní znaky, bez jejich speciálního významu). Jinými slovy, můžeme např. zahrnout znak dolar do Substitution řetězce použitím '\$' potom s tím mod_rewrite nebude zacházet jako se zpětnou referencí.

3.2.3 Schopnost zpětné reference

Kdykoliv použijeme závorku v Pattern nebo v nějakém CondPattern, budou vytvořeny zpětné reference, které mohou být použity s řetězci $N a %N. Ty mohou být použity k vytvoření řetězců Substitution a TestString. Obrázek ukazuje, které zpětné reference jsou poslány k rozšíření.



3.3 Hlavní direktivy pro přepisování

Aby modul fungoval, jsou potřeba pouze dvě z direktiv mod_rewrite. První z nich, RewriteEngine, zapíná jádro pro přepisování. Druhá, RewriteRule, je jediná direktiva, která musí být uvedena ve všech sadách pravidel pro přepisování.

3.3.1 RewriteEngine - zapínání nebo vypínání přepisování

Direktiva RewriteEngine je velmi jednoduchá, ale je pro mod_rewrite důležitá. Zapíná nebo vypíná přepisování URL:

```
RewriteEngine on | off
```

Přepisování je implicitně vypnuto a modul nedělá vůbec nic. Pro použití mod_rewrite musí být aspoň jedna direktiva RewriteEngine on. Nastavení se provádí v httpd.conf nebo v souboru .htaccess.

Konfigurace pro přesměrování mod_rewrite nejsou implicitně děděny. To znamená, že virtuální hostitelé nedědí konfiguraci přesměrování jako většinu

jiných obecných konfiguračních direktiv od primárního serveru, takže obvykle je potřeba RewriteEngine v každém virtuálním hostiteli.

3.3.2 RewriteRule - definování sady pravidel

Direktiva RewriteRule je nejdůležitější součástí mod_rewrite. Bez direktivy RewriteRule nemá žádná jiná z direktiv mod_rewrite nějaký efekt. Každá direktiva RewriteRule vyjadřuje jedno pravidlo pro porovnávání ve formě:

```
RewriteRule Pattern Substitution
```

Každé pravidlo se skládá ze vzorku (Pattern), s nimiž je původní URL porovnáváno, a akcí, které se mají provést, pokud je porovnání úspěšné. Vzorkem je regulární výraz, který je porovnáván s aktuálním URL, na něž je pravidlo aplikováno. Toto aktuální URL nemusí být URL původního požadavku, ale mohlo být změněno předchozími direktivami RewriteRule.

Mod_rewrite rozlišuje porovnávání s regulárními výrazy následujícími důležitými způsoby:

- Regulární výraz může být negován tím, že před ním bude uveden znak logické funkce NOT (!).
- V regulárních výrazech RewriteRule mohou být použity zpětné reference. Jsou to zpětné reference ve formě $N, kde N může být celé číslo od 0 do 9 a je nahrazeno obsahem N-té sady závorek v části pravidla Pattern. V následující direktivě RewriteRule část URL, která odpovídá výrazu uvnitř první sady závorek, nahrazuje $1 v substituci a část odpovídající druhé sadě nahrazuje $2:

  ```
  RewriteRule ^/(.*)/(.*) /index.php?a=$1&b=$2
  ```

- V regulárních výrazech RewriteCond mohou být použity zpětné reference. Jsou to zpětné reference ve formě %N, kde N může být celé číslo od 0 do 9 a je nahrazeno obsahem N-tých závorek v poslední porovnávané direktivě RewriteCond.
- Proměnné prostředí kde speciální formát %{proměnná} je nahrazen hodnotou proměnné z prostředí Apache. To je užitečné zejména pokud předchozí direktivy RewriteRule byly použity pro nastavení libovolných prostředí. Proměnné prostředí hraje důležitou roli v podmíněném přepisování RewriteCond, ale možná ho někdy použijeme i pro přepisování URL pomocí direktivy RewriteRule.
- Volání mapovacích funkcí. Jsou to volání pro mapování, dříve definovaná pomocí RewriteMap ve formě ${mapname:key|default}.

Všechna pravidla přepisování se aplikují na substituci. URL je zcela nahrazeno substitucí a proces přepisování pokračuje, dokud zde nebudou další pravidla nebo dokud nebude RewriteRule v sadě explicitně ukončeno příznakem [L]. Pokud se substituce skládá z jediného znaku – (minus), nebude provedena žádná substituce a URL je předáno nezměněno dalšímu RewriteRule. To je užitečné, když chceme vytvořit více porovnání (pomocí příznaku [C] pro řetězení direktiv RewriteRule) předtím, než přepíše URL. Jiným využitím je použití RewriteRule, které nepřepisuje URL, ale místo toho má příznak, indikující určitou odpověď, která se má odeslat uživateli jako například příznak [F], který vrací odpověď se stavem http 403 (Forbidden).

Pro substituci přidáním [flags] jako třetího parametru můžeme nastavit speciální příznaky direktivy RewriteRule. Flags je čárkami oddělený seznam následujících příznaků:

- R | redirekt [=kód] - považuje URL za přesměrování. Do odpovědi je umístěn implicitní stavový kód HTTP 302 (Found). Můžeme definovat jiný stavový kód odpovědi v rozmezí 300-400 přidáním

=kód k příznaku [R], kde kódem může být číslo nebo jeden ze tří symbolických názvů: temp (implicitně), permanent nebo seeother.

- F | forbidden - klientovi je okamžitě odeslána odpověď HTTP se stavem 403 (Forbidden).
- G | gone - klientovi je okamžitě odeslána odpověď HTTP se stavem 410 (Gone).
- P | proxy - vynutí, aby bylo přepsané URL zpracováno interně modulem proxy mod_proxy, které musí být nainstalován, aby fungoval. Tento příznak také způsobí, že zpracování mod_rewrite skončí s příznakem [L].
- L | last - indikuje, že by zpracování mod_rewrite mělo skončit touto sadou pravidel. Příznak bude efektivní, pouze pokud vzorek odpovídá URL. Pokud porovnání nesouhlasí, zpracování přepisování bude pokračovat další direktivou RewriteRule v aktuální sadě pravidel.
- N | next - spouští zpracování přepisování od první direktivy RewriteRule v aktuální sadě pravidel. Proces přepisování začíná s přepsaným URL z poslední direktivy RewriteRule.
- C | chin - řetězí aktuální pravidlo s následujícím. Pokud nějaké pravidlo v sadě řetězených pravidel neodpovídá, všechna následující pravidla v řetězci jsou ignorována.
- T | type=MIME - vynutí typ MIME pro cílový soubor přepsaného URL. Používá se často pro simulaci direktivy ScriptAlias, která vynucuje, aby všechny soubory uvnitř cílového adresáře měly typ MIME application/x-httpd-cgi.
- NS | nosubreq - způsobí, že aktuální pravidlo bude přeskočeno, pokud je aktuální požadavek interním podpožadavkem.

- NC | nocase - vynucuje, aby porovnání vzorku s URL nerozlišovala malá a velká písmena.
- QSA | qsappend - pokud je přepisováné URL řetězec dotazu a přepisované URL je také řetězcem dotazu, tento příznak vynucuje, aby byla přepsána část řetězce dotazu ke stávajícímu URL místo jeho nahrazení. Používá se tehdy, pokud chceme přidat více dat do řetězce dotazu skrze pravidlo přepisování.
- S | skip=num - vynucuje, aby jádro pro přepisování přeskočilo následující num pravidel v případě, že vzorek odpovídá URL v aktuálním pravidlu.
- E | env=var:value - nastavuje proměnou prostředí, pokud vzorek odpovídá URL v aktuálním pravidlu. Hodnota může obsahovat zpětné reference RewriteRule i RewriteCond ($N i %N). Často se používá pro nastavení podmínek řízení přístupu. Tento příznak můžeme použít více než jednou v jedné direktivě RewriteRule pro nastavení více než jedné proměnné prostředí.
- PT | passthrough - vynutí, aby jádro pro přepisování nastavilo pole URI interní struktury request_rec na hodnotu pole filename. Jelikož mod_rewrite překládá URL na názvy souborů, přepsané URL je známé interně servru jako název souboru.

3.3.3 RewriteCond - podmíněné přepisování

Před direktivami RewriteRule mohou být jedna nebo více direktiv RewriteCond, z nichž každá funguje podmínkou pro jedno pravidlo, která bude vyhodnocena jako pravda nebo nepravda pro určení, zda pravidlo bude aplikováno. Před aplikací jakéhokoliv RewriteRule je aplikováno jen tehdy,

pokud jsou všechny předchozí direktivy RewriteCond vyhodnoceny jako pravda. Pokud nebude více podmínek RewriteCond spojeno pomocí příznaku [OR] a je vyhodnoceno jako jediné pravidlo, bude RewriteRule ignorováno pokud nějaká z předchozích podmínek nebude splněna. RewriteCond má následující syntaxi:

```
RewriteCond   MatchString    Pattern
```

Parametr MatchString

MatchString je čistě řetězový řetězec, jenž může obsahovat následující konstrukce, které jsou nahrazeny před porovnáním řetězce se vzorkem:

- Zpětné reference RewriteRule jsou zpětné reference ve formě $N, kde N může být celé číslo od 0 do 9 a je vyplněno obsahem N-té sady závorek v části Pattern RewriteRule, které jej následuje, a všechny jiné direktivy RewriteCond s ním seskupené.
- Zpětné reference RewriteCond jsou zpětné reference ve formě %N, kde N může být celé číslo od 0 do 9 a je vyplněno obsahem N-té sady závorek v naposledy vyhodnocované direktivě RewriteCond.
- Proměnné serveru jsou proměnné ve formě % {název_proměnné}, kde název_proměnné je jedním z následujících:
    - Hlavičky HTTP: HTTP_USER_AGENT, HTTP_REFERER, HTTP_COOKIE, HTTP_FORWARDER, HTTP_HOST, HTTP_PROXY_CONNECTION, HTTP_ACCEPT.
    - Připojení a požadavek: REMOTE_ADDR, REMOTE_HOST, REMOTE_USER, REMOTE_IDENT, REQUEST_METHOD, SCRIPT_FILENAME, PATH_INFO, QUERY_STRING, AUTH_TYPE.

- Interní hodnoty serveru: DOCUMENT_ROOT, SERVER_ADMIN, SERVER_NAME, SERVER_ADDR, SERVER_PORT, SERVER_PROTOCOL, SERVER_SOFTWARE.
- Hodnoty systému: TIME_YEAR, TIME_MON, TIME_DAY, TIME_HOUR, TIME_MIN, TIME_SEC, TIME_WDAY, TIME.
- Speciální proměnné: API_VERSION, SCRIPT_URI, SCRIPT_URL, REQUEST_URI, REQUEST_FILENAME, IS_SUBREO.

- Proměnné prostředí - speciální formát $ { ENV:variable} je vyplněn hodnotou proměnné z prostředí Apache. To je užitečné hlavně tehdy, kdy byly předchozí direktivy RewriteRule použity pro nastavení libovolných proměnných prostředí.
- Hlavičky HTTP jsou speciální format % {HTTP:header}, který je vyplněn hodnotou hlavičky z vyhodnocovaného požadavku HTTP.
- Dopředné vyhledávání – speciální formát % {LA-U:variable} je vyplněn hodnotou proměnné, která bude obvykle nastavena později v cyklu požadavku v závislosti na URL. Například pokud jste chtěli podmínit své přepisování hodnotou REMOTE_USER v kontextu podle serveru (tj. pomocí direktiv v httpd.conf), musíme použít dopřené vyhledávání, protože mod_rewrite provádí svou práci dříve než fáze autorizace nastaví proměnnou . Pokud přepisování nastává v kontextu podle adresáře (ze souboru .htaccess), objeví se ve fázi fixup cyklu a po fázích autorizace. Jiná forma dopředného vyhledávání používá formát % {LA-F:variable}, který je vyplněn hodnotou proměnné variable v závislosti na názvu souboru a ne URL. Tato forma je zapotřebí jen zřídka.

Parametr Pattern

Pattern v direktivě RewriteCond je regurální výraz. Který je porovnán s direktivou MatchString. Pokud souhlasí, celá direktiva RewriteCond bude vyhodnocena jako pravda. Pokud nesouhlasí, bude direktiva vyhodnocena jako nepravda. Tento regurální výraz byl nahrazen modulem mod_rewrite následujícím způsobem:

- Před Pattern můžete přidat znak ! pro negaci porovnávání vzorku.
- Místo porovnávání s regulárním výrazem můžete použít jedno z následujících porovnání MatchString s Pattern. Každé porovnání je vyhodnoceno jako pravda nebo nepravda.
    - <Pattern pracuje s Pattern jako s čistě textovým řetězcem a porovnává jej lexikálně (pomocí alfanumerických a jiných znaků) s MatchString. Vyhodnotí se jako pravda, pokud je MatchString lexikálně menší než Pattern.
    - >Pattern pracuje jako Pattern jako s čistě textovým řetězcem a porovnává jej lexikálně s MatchString. Vyhodnotí se jako pravda, pokud je MatchString lexikálně větší než Pattern.
    - =Pattern pracuje s Pattern jako s čistě textovým řetězcem a porovnává jej lexikálně s MatchString. Vyhodnotí se jako pravda, pokud je MatchString lexikálně stejný jako Pattern. Pokud je Pattern jen „“, bude se MatchString porovnávat s neexistujícím řetězcem.
    - -d pracuje s MatchString jako s cestou a zjišťuje, zda existuje a zda je adresář.

- -f pracuje s MatchString jako s cestou a zjišťuje, zda existuje a zda je běžný soubor.
- -s pracuje s MatchString jako s cestou a zjišťuje, zda existuje soubor větší než nula.
- -l pracuje s MatchString jako s cestou a zjišťuje, zda existuje a zda je symbolický odkaz.
- -F pracuje s MatchString jako s cestou a zjišťuje, zda existuje a zda je přístupná skrze všechny pro server aktuálně nastavená řízení přístupu pro tuto cestu. Požívá pro zjištění interní podpožadavek, takže při častém použití této volby dochází ke snižování výkonu.
- -U pracuje s MatchString jako s URL a zjišťuje, zda je přístupné skrze všechna aktuálně nastavená řízení přístupu pro tuto cestu. Používá pro zjištění interní požadavek, takže při častém používání této volby dochází ke snižování výkonu.

- Kromě toho můžete nastavit speciální příznaky pro Pattern přidáním [flags] jako třetího parametru direktivy RewriteCond. Příznak může být buď jeden nebo mohou být oba za sebou (oddělené čárkou):
  - ornext | OR - kombinuje aktuální podmínku a další podmínku s logickým OR místo implicitního AND. Pokud je některý z údajů pravdivý, budou dohromady vyhodnoceny jako pravda.
  - nocase | NC - nehledí při porovnávání MatchString s Pattern na velká a malá písmena.

3.4 Nastavení základního URL: direktiva RewriteBase

Direktiva RewriteBase explicitně nastavuje základní URL pro přepisování podle adresářů. Direktiva je aplikována pouze pro přepisování podle adresářů, třeba v kontejneru <Directory>, ale obvykle v souboru .htaccess. Je nezbytná v případech, kdy URL požadavku neodpovídá místnímu systému souborů, obyčejně proto, že byla pro přesměrování požadavků na místa mimo DocumentRoot použita direktiva Alias.

Aby přepisování podle adresářů fungovalo, je výsledné URL z těchto posledních přepisů v cyklu znovu posláno serveru (interně) a začíná nový cyklus požadavku. URL je řečeno, aby bylo znovu předáno jádru serveru. Pravidlo přepisování, je-li aplikováno v kontextu adresáře, odstraňuje RewriteBase (což je implicitně prefix místního adresáře), aplikuje pravidlo na zbytek URL a pak před něj přidá RewriteBase. V případě, že se příchozí URL nemapuje na místní systém souborů, budete chtít změnit RewriteBase tak, aby odpovídalo prefixu příchozích URL a ne prefixu místního adresáře.

3.5 Direktiva RewriteOptions

Nastavení mod_rewrite implicitně nejsou děděna. Toto chování musí být explicitně zapnuto v každém kontextu, kde je tato dědičnost požadována. Pro tento účel můžete použít direktivu RewriteOptions. Ačkoliv byla navržena jako direktiva s obecným účelem pro nastavování speciálních možností modulu mod_rewrite, existuje aktuálně pouze jedna volba, kterou je možno nastavovat, a to inherit. Takže existuje pouze jediná forma použití direktivy:

```
RewriteOptions   inherit
```

To způsobí, že aktuální konfigurace mod_rewrite bude děděna z nadřazených nastavení. Existují dva zcela odlišné způsoby použití v závislosti na kontextu:

- Virtuální hostitel – pokud je direktiva RewriteOptions inherit nastavena v kontextu virtuálního hostitele, budou nastavení mod_rewrite děděna z primárního serveru. To zahrnuje podmínky přepisování, pravidla a mapy.
- Adresář – v kontextu adresáře (například je-li direktiva RewriteOptions inherit obsažena v souboru .htaccess), jsou nastavení mod_rewrite děděna z nadzařeného adresáře souboru .htaccess, pokud existují.

3.6 Protokolování přepisování

Při použití `mod_rewrite` je vždy dobré zapnout protokolování tohoto modulu, zejména pokud ladíte svá pravidla pro přepisování. Protokolování poskytované modulem je dokonalé a velmi užitečné. Může být také poučné, pokud věnujete čas sledování vývoje přepisování skrze sady pravidel. Pro použití protokolování musíme zadat soubor, do něhož bude protokol ukládán, a musíme nastavit úroveň informací, které budou protokolovány. Modul mod_rewrite pro to poskytuje samostatné direktivu.

3.6.1 Zadání umístění souboru protokolu: RewriteLog

Direktiva RewriteLog udává název souboru, do něhož server protokoluje akce přepisování, které provádí. Zde je příklad:

RewriteLog "/mod_rewrite.log"

Pokud nebude soubor začínat lomítkem (/), předpokládá se, že jde o cestu relativní k ServerRoot. Direktiva by se měla objevit v konfiguraci pro každý server pouze jednou.

Pro vypnutí protokolování přepisování nenastavuje název souboru na /dev/null. I když to vypadá, že neprobíhá zapisování do žádného protokolu, nejde o skutečné vypnutí protokolování. Pouze jednoduše přesměrujeme výstup protokolování do ovladače souboru systému null. Činnosti protokolování jsou stále prováděny a spotřebovávají prostředky procesoru. Pro vypnutí protokolování se buď odstraní direktiva RewriteLog nebo se použijte RewriteLogLevel 0.

3.6.2 Nastavení úrovně protokolování: RewriteLogLevel

Direktiva RewriteLogLevel nastavuje úroveň výřečnosti souboru protokolu přepisování. Implicitní úrovní je 0, což znamená žádné protokolování, zatímco 9 nebo více znamená protokolování prakticky veškeré činnosti.

```
RewriteLogLevel    2
```

RewriteLogLevel 2 produkuje protokol, který demonstruje různé etapy procesu přepisování a může být užitečný při zjišťování efektu všech direktiv RewriteCond a RewriteRule. Hodnoty větší než 2 výrazně zpomalí server Apache a měly by být používány pouze v průběhu ladění.

3.7 Direktiva RewriteMap

Direktiva RewriteMap pojmenovává mapu přepisování pro pozdější použití v RewriteRule a udává zdroj mapy přepisování, který může být použit pro

vyhledávání informací na základě hledání klíče a pozdější vložení do substitučního řetězce RewriteRule. Má následující syntaxi:

```
RewriteMap MapName MapType:MapSource
```

Mapa přepisování je hlavně vyhledávací tabulkou. Skládá se z nějakého počtu dvojic proměnná / hodnota.

Pokud je jedna z následujících konstrukcí vložena do substitučního řetězce RewriteRule, bude nahrazena výsledkem vyhledávání RewriteMap, pokud bude klíč nalezen v mapě. Jinak bude konstrukce nahrazena výchozí hodnotou DefaultValue (pokud je zadána), nebo prázdným řetězcem.

```
${ MapName:LookupKey }
${ MapName:LookupKey | DefaultValue }
```

Direktiva RewriteMap se může objevit více než jednou. Pro každou mapovací funkci se použije jedna direktiva RewriteMap pro deklarování jejího souboru s mapou přepisování. I když nemůžeme mapu deklarovat v kontextu adresářů, můžeme tuto mapu v kontextu adresářů použít.

Mohou být použity následující kombinace MapType a MapSource:

- Standardní čistý text (MapType txt)
- Náhodný čistý text (MapType rnd)
- Soubor Hash (MapType dbm)
- Externí funkce (MapType prg)
- Interní funkce (MapType int)

3.7.1 Standardní čistý text

Standartní mapa přepisování je tou, v níž je MapSource čistě textový soubor obsahující dvojice klíč/hodnota, na každém řádku jednu. Prázdné řádky jsou ignorovány. Komentáře začínají znakem #.

Například můžeme vytvořit následující textový soubor map.txt, který slouží jako mapa přepisování názvů kategorii na její ID:

```
##
## map.txt – rewriting map
##
bazeny                  100
bazeny/tampa            101
bazeny/waikiki          102
sauny                   200
prislusenstvi           300
folie                   400
```

Následující direktiva RewriteMap povolí tento soubor jako čistě textovou mapu přepisování, na kterou je možno odkazovat v dalších direktivách RewriteRule názvem category:

```
RewriteMap category txt:/map.txt
```

Pokud název souboru zadaný v direktivě RewriteMap neexistuje, Apache zapíše chybovou zprávu do svého chybového protokolu a nespustí se.

RewriteRule pro přístup k této mapě by mohla vypadat takto:

```
RewriteRule ^(.*)$ open.php?categoryid=${category:$1}
```

Toto pravidlo používá substituční řetězec (`${category:$1}`) následujícím způsobem:

```
${  MapName:LookupKey  }
```

Vzorek v tomto pravidle odpovídá URL, jenž bude vždy názvem kategorie. Obsah závorek uzavírající tento název nahradí token $1 v substituci tak, že se název stane klíčem ve vyhledávání v dříve definované mapě přepisování s názvem category. Hodnota je použita při vytváření přepsaného URL doslovně.

3.7.2 Náhodný čistý text

Tento typ mapy je variací čistě textové mapy přepisování. Vyhledává své hodnoty úplně stejně, ale provádí určité další zpracování po vyhledání. Pokud získaná hodnota obsahuje jeden nebo více znaků |, předpokládá se, že obsahuje více než jednu alternativní hodnotu. Hodnota dosazená do pole Substitution direktivy RewriteRule, které odkazuje na tuto mapu, je náhodně vybrána ze seznamu.

Může to působit jako frivolní funkce, ale byla navržena pro vyvažování zátěže v situaci s reverzní proxy, kde jsou hledanými hodnotami názvy serverů.

```
##
##  map.txt -- rewriting map
##
static  www1|www2|www3|www4
dynamic www5|www6
```

RewriteRule pro přístup k této mapě by mohla vypadat takto:

```
RewriteMap servers rnd:/map.txt
```

3.7.3 Soubor Hash

Soubor hash s mapou přepisování se používá přesně stejně jako vyhledávání v čistě textové mapě, s výjimkou toho, že je textový soubor nejprve zkompilován do binárního formátu DBM Linuxu. DBM je implementováno v Linuxu jako knihovna rutin, které spravují datové soubory, obsahující dvojice klíč/hodnota a poskytující optimalizované vyhledávání dat založené na řetězcových klíčích. To značně urychluje proces vyhledávání, ale ve většině situací bude vyhledávání v čistém textu skoro stejně rychlé jako použití DBM a bude jednodušší pro spravování. Soubor hash DBM by měl být použit pouze tehdy, pokud vaše mapa přepisování narůstá na několik set řádků a vy se snažíte dosáhnout nejvyššího možného výkonu.

3.7.4 Interní funkce

Existují čtyři interní funkce Apache, které mohou být vyvolány pomocí mapy přepisování:

- toupper - převádí všechny znaky klíče na velká písmena.
- tolower - převádí všechny znaky klíče na malá písmena.
- escape - mění speciální znaky v klíči a překládá je na hexadecimální kódy. To správně formátuje klíč pro jeho použití jako URL.
- unescape - převádí speciální znaky v klíči (URL s hexadecimálními kódy) zpět na speciální znaky.

První dvě funkce nejsou nijak zvlášť užitečné, ale poslední dvě mohou být velmi užitečné, pokud budeme někdy muset pracovat s URL, obsahujícími speciální znaky, které musí být převedeny pro zpracování na hexadecimální kódy. Následující RewriteRule přebírá URL, jenž mohlo být zpracováno předchozími direktivami RewriteRule a převádí jej. Příznak [R] pak přesměrovává požadavek:

```
RewriteRule    ^ ( .* )  ${escape:$1}   [R]
```

Změna speciálních znaků v URL je často zapotřebí, pokud by URL mohlo obsahovat znaky, které mají speciální význam. URL obsahující znak ? je například potřeba změnit, aby tento znak nebyl interpretován serverem jako parametry předané skriptu. V tomto příkladu si všimněte, že pro použití těchto interních funkcí není potřeba žádná direktiva RewriteMap, jelikož již existují jako vestavěné mapy přepisování.

3.7.5 Externí program pro přepisování

Zdrojem mapy může být také uživatelem napsaný program. Tento program může být napsán v jakémkoliv jazyce, který umí přijmout vstup na svém standardním vstupu (stdin), upravit jej a vrátit hodnotu na svůj standardní výstup (stdout). Jelikož program nebude provádět nic více, než jednoduché mapování vstupního klíče na odpovídající výstupní hodnotu, výborně se pro něj hodí interpretované skripty shellu (a zejména Perl).

Tento program je spuštěn jednou, při spuštění Apache. Je navržen pro zadání nepřetržité smyčky, ve které čeká na vstup na svém standardním vstupu. Když přijme hodnotu klíče od mod_rewrite, provede vyhledání a vrátí s ním spojenou hodnotu (vždy následovanou znakem nového řádku) pro vložení do

substitučního řetězce RewriteRule. Pokud program nenajde žádnou odpovídající hodnotu pro daný klíč, měl by vrátit buď prázdný řetězec nebo řetězec null. Tento program by měl být udržován co nejjednodušší a nejpřizpůsobivější. Pokud se váš program zastaví, zastaví se server Apache v tomto bodě, kde zpracovává RewriteRule, která odeslala data do externího programu.

4 Závěr

Optimalizace pro vyhledávače hraje důležitou roli pro získání nových návštěvníků z fulltextových vyhledávačů.

V první části absolventské práce jsem zmínil pouze hlavní pravidla optimalizace. V České Republice ještě není optimalizace tolik rozšířená jako ve světě, proto postačí pouze základní optimalizace. Pokud je web v hodně konkurenčním prostředí, bude nutností udělat některé další kroky.

Mod_rewrite je jednou z možností vylepšení URL adresy. Je to poměrně mocný nástroj pro přepis URL adres. Setkáváme se zde s jedním problémem a to je jeho špatná podpora v komerčním webhostingu. Než začneme vytvářet nový web, je nutné si zjisti podporu mod_rewrite na serveru.

V praxi jsem optimalizaci provedl na webu http://www.pyramida-plzen.cz. Výsledky optimalizace nemusejí být vidět hned. Musíme však počkat než robot celý web zaindexuje a to může trvat i několik měsíců. Použitím mod_rewrite jsem pomohl nejenom robotům vyhledávačů, ale i uživatelům v lepší orientaci na webu.

5 Přílohy

5.1 Regulární výrazy

Výraz	Vysvětlení
^	začátek řetězce
$	konec řetězce
.	jeden libovolný znak
*****	opakování libovolně mockrát
+	opakování alespoň jednou
?	opakování nejvýše jednou
{min,max}	opakování od min do max
{n}	opakování právě n-krát
0(podvýraz)	vytvoření podvýrazu
[abc]	výčet znaků (znaky a,b,c)
[a-zA-Z]	výčet znaků (všechna písmena)
[^0-9]	negace výčtu (vše kromě číslic)

5.2 Stavová hlášení HTTP protokolu

1xx informační

100 – Continue – pokračování – přijata část požadavku, klient může pokračovat v zasílání požadavku.

101 – Switching Protocols – přepínání protokolu – server přepíná protokol.

2xx úspěch

200 – OK – v pořádku – vše v pořádku, server zasílá odpověď.

201 – Created – vytvořeno – výsledkem zpracování dotazu bylo vytvoření nového objektu, který lze identifikovat pomocí URI. URI vytvořeného dokumentu je posíláno v těle odpovědi.

202 – Accepted – akceptováno – dotaz přijat, probíhá zpracování.

203 – Non-authoritative Information – nesměrodatná informace

204 – No Content – netřeba měnit dokument – požadavek byl úspěsný, jeho výsledkem nejsou žádná data pro klienta.

205 – Reset Content – obnovený obsah dokumentu – požadavek splněn, klient smí obnovit původní obsah dokumentu (primárně určeno jako informace pro klienta, že smí vynulovat obsah formuláře).

206 – Partial Content – neúplný obsah dokumentu – server splnil část GET požadavku pro zdroj.

3xx přesměrování požadavku

300 – Multiple Choices – více voleb – požadovaný dokument je dostupný na několika místech, klient musí vybrat jeden z dokumentů a znovu vyslat dotaz.

301 – Moved permanently – objekt přesunut – objekt byl trvale přestěhován na nov URI (oznámeno v hlavičce Location). Klient se musí zeptat na novém umístění.

302 – Moved temporarily – objekt dočasně přesunut – objekt dočasně přesunut.

303 – See Other – lze nalézt pod jinými URI – požadavek může být nalezen pod jiným URI.

304 – Not modified – nezměněno – beze změny. Byl-li požadavek GET, přístup povolen a dokument nezměněn, smí server odpovědět tímto kódem. Odpověď 304 nesmí obsahovat tělo zprávy.

305 – Use Proxy – použij proxy – požadavek musí být znovu poslán prostřednictvím proxy uvedené v URL.

306 – zatím nepoužito, rezervováno

307 – Temporary Redirect – dočasně přesunuto – požadovaná stránka byla dočasně přesunuta na nové URI.

4xx chyba klienta (chybný požadavek)

400 – Bad request – chybný požadavek – server nerozumí požadavku. Příčinou může být chybně formulovaný dotaz nebo chyba v URI adrese.

401 – Unauthorized – neautorizovaný přístup – neoprávněný přístup k webové stránce (klient nesplnil identifikační požadavky). Některé stránky jsou povolené pouze pro přístup z určitých domén.

402 – Payement Required – zatím nepoužito, rezervováno pro budoucí použití

403 – Forbidden – obecná chyba – Access forbidden – server by rád odpověděl, ale nemá to povoleno.

404 – Not found – objekt nenalezen – objekt s požadovaným URL neexistuje. Tento chybový kód je nejčastější. Příčinou bývá buď překlep v zápisu URL nebo neexistence (zánik) objektu.

405 – Method Not Allowed – nepovolená metoda – metoda specifikovaná v požadavku není povolena.

406 – Not Acceptable – neakceptovatelné – server může generovat pouze odpověď, která není klientem akceptována.

407 – Proxy Authentication Required – je požadovaná proxy autentifikace – před obsloužením požadavku musí být tento požadavek autentifikován proxy serverem.

408 – Request Timeout – vypršení doby požadavku – potřeba požadavku je delší, než kolik si server připravil na čekání.

409 – Conflict – konflikt – požadavek nemůže být splněn z důvodu konfliktu.

410 – Gone – ukončeno – požadovaná stránka již není nadále přístupná.

411 – Lenght Required – je požadována délka – server neakceptoval požadavek, protože hlavička "Content-Lenght" není definována.

412 – Precondition Failed – přednastavená podmínka je chybná – podmínka ,která je dána v požadavku byla serverem vyhodnocená jako chybná.

413 – Request-url Entity Too Large – požadované množství je příliš velké – server neakceptoval požadavek, protože požadované množství je příliš velké.

414 – Request-url Too Long – URI požadavku je příliš dlouhé – požadavek nebyl akceptován serverem. Chyba se objeví, je-li "POST" požadavek překonvertován na požadavek "GET" s dlouhou dotazovací informací.

415 – Unsupported Media Type – nepodporovaný typ média – server neakceptoval požadavek, protože typ média není podpotován.

416 – Requested Range Not Satisfiable – požadovaný rozsah je nesplnitelný – je-li v požadavku hlavička Range vyplněna rozmezím hodnot, které nevyhovují rozsahu hodnot aktuálně vybraného zdroje, může server vrátit tuto chybu.

417 – Expectation Failed – předpoklad skončil chybou – předpoklad zadaný v hodnotě hlavičky požadavku Except nemůže server dosáhnout.

5xx chyba serveru

500 – Internal server error – vnitřní chyba serveru – při zpracování dotazu došlo v programu serveru k blíže neurčené chybě.

501 – Not implemented – neimplementováno – hlášení serveru, pokud je po něm vyžadována metoda, kterou neovládá.

502 – Bad gateway – špatná brána – může hlásit proxy – tato chyba je zaslána zprostředkujícím serverem, pokud na váš dotaz obdržel od původního serveru špatnou odpověď.

503 – Service unavailable – služba nedostupná – může být způsobeno přetížením serveru – server momentálně nedokáže dotaz obsloužit.

504 – Gateway Timeout – doba průchodu vypršela

505 – HTTP Version Not Supported – nepodporovaná verze HTTP – server nepodporuje verzi HTTP protokolu.

6 Literatura a ostatní použité zdroje

[1] Smička, Radim: Optimalizace pro vyhledávače - SEO, Jaroslava Smičková, Dubany 2004, počet str. 126, ISBN 80-239-2961-5

[2] Aulds, Charles: Linux - administrace serveru Apache, 1. vydání, Grada Publishing 2003, počet str. 535, ISBN 80-247-0640-7

[3] Schlossnagle, George: Pokročilé programování v PHP5, 1. vyd., Zoner Press 2004, počet str. 640, ISBN 80-86815-14-5

[4] Špinar, David: Tvoříme přístupné webové stránky, 1. vydání, Zoner Press 2004, počet str. 360, ISBN 80-86815-11-0

6.1 Internetové zdroje

http://httpd.apache.org/docs-2.0/mod/mod_rewrite.html

http://www.sitepoint.com/article/guide-url-rewriting

http://httpd.apache.org/docs-2.0/misc/rewriteguide.html

http://www.sovavsiti.cz/2003/mod_rewrite.html

http://seo.nawebu.cz/

http://www.abowe.brbla.net/2/stavova-hlaseni-http-protokolu.php

http://www.root.cz

http://www.interval.cz