VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

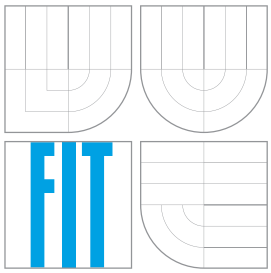
NÁSTROJ PRO VYHODNOCOVÁNÍ OPTIMALIZACE WEBOVÝCH STRÁNEK

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE KAREL HÁK
AUTHOR

BRNO 2010

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

NÁSTROJ PRO VYHODNOCOVÁNÍ OPTIMALIZACE WEBOVÝCH STRÁNEK

A TOOL FOR EVALUATION OF WEB PAGE OPTIMIZATION

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

KAREL HÁK

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. RADEK BURGET, Ph.D.

BRNO 2010

Abstrakt

Práce se zabývá principy webových vyhledávačů, způsoby, jakými lze webové stránky optimalizovat pro vyhledávače a návrhem nástroje, který bude na základě zvolených parametrů (*klíčová slova, pozice ve vyhledávačích, ohodnocení stránky - SRank a PageRank*) vyhodnocovat míru optimalizace konkrétní webové stránky. Dále je v práci rozebrána implementace navrženého nástroje, použité techonologe (*PHP, XML-RPC, CSS, JavaScript*) a knihovny (*Nette framework*).

Abstract

This bachelor's thesis deals with the web search engines, the ways of web page optimization for search engines and the design of a tool, which evaluates the degree of web page optimization based on selected parameters (*keywords, position in search engines, website ranking - SRank and PageRank*). The thesis also describes the implementation of the designed tool, used technologies (*PHP, CSS, JavaScript, Git*) and libraries (*Nette framework*).

Klíčová slova

optimalizace pro vyhledávače, seo, vyhledávače, php

Keywords

search engine optimization, seo, search engines, php

Citace

Karel Hák: Nástroj pro vyhodnocování optimalizace webových stránek, bakalářská práce, Brno, FIT VUT v Brně, 2010

Nástroj pro vyhodnocování optimalizace webových stránek

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Radka Burgeta, PhD. a uvedl všechnu použitou literaturu a prameny.

.....
Karel Hák
18. května 2010

Poděkování

Děkuji vedoucímu bakalářské práce Ing. Radku Burgetovi, PhD. za odborné vedení a cenné připomínky.

Obsah

Kapitola 1

Úvod

V době, kdy počet webových stránek roste neuvěřitelným tempem, je stále složitější přilákat návštěvníky právě na ty naše. Jedním ze způsobů, jak toho lze docílit, jsou tzv. vyhledávače a katalogy webových stránek. S jejich pomocí se uživatelé internetu dostávají na stránky, jejichž adresu neznají nebo ji zapomněli. Jejich popisem se zaobírá kapitola 2. Dozvíme se zde, co to katalogy a vyhledávače jsou a jaké mechanizmy využívají pro nalezení stránek. Dále si popíšeme aktuální situaci na českém trhu.

Pokud chceme potenciálu vyhledávačů využít naplno, měli bychom se začít zajímat o optimalizaci pro vyhledávače. Její hlavním cílem je vylepšit pozici optimalizované stránky ve výsledcích vyhledávání. Kapitola 3 se zabývá faktory ovlivňujícími pozici ve výsledcích vyhledávání, nekalými praktikami, hodnocením webových stránek vyhledávači a upozorňuje na nejčastější nedostatky stránek. Poslední část kapitoly pojednává o přínosu optimalizace pro běžné uživatele.

Cílem této práce je návrh a implementace softwarového nástroje, který by usnadnil optimalizaci webových stránek. Tento nástroj dokáže nejen ohodnotit relevanci stránky k daným klíčovým slovům, ale také poradit, co je potřeba udělat pro dosažení lepších výsledků. Kapitola 4 obsahuje specifikaci požadavků, návrh a implementaci nástroje. Následující kapitola číslo 5 se zabývá ukázkovou aplikací. V kapitole 6 nalezneme informace o použitých technologiích a knihovnách.

Závěrečná kapitola se zabývá zhodnocením výsledků, předpokládanou budoucností vývoje a plánovaným využitím nástroje.

Kapitola 2

Vyhledávače a katalogy webových stránek

2.1 Katalogy

Katalog webových stránek můžeme velmi dobře přirovnat k telefonnímu seznamu. Stejně jako v telefonním seznamu můžeme prohlížet telefonní čísla rozdělená podle měst či oborů, můžeme v katalogu prohlížet odkazy na webové stránky rozdělené do různých kategorií. Většina katalogů navíc umožňuje uživatelům ve své databázi vyhledávat.

Nové stránky se nepřidávají automaticky. Děje se tak až po vyplnění přidávacího formuláře. U kvalitních katalogů je navíc vyžadováno schválení stránky editorem. Ten zkontroluje, zda stránka neodporuje pravidlům katalogu, zda je umístěna ve správných kategoriích a v případě, že nenalezne závažné nedostatky, potvrdí její zařazení do katalogu. U některých katalogů bývá vyžadován poplatek za přidání, respektive ponechání odkazu. Odměnou za tento poplatek bývají velmi často doplňující údaje, které lze k záznamu doplnit. Můžeme také narazit na katalogy podmiňující přidání tím, že na stránkách bude umístěn zpětný odkaz na katalog.

Více informací lze dohledat v literatuře [30], [15], [25] a [23].

2.1.1 Příklady katalogů

Na tomto místě jsou uvedeny pouze nejznámnější, resp. nejvýznamnější katalogy webových stránek. Pokud potřebujeme obsáhlý seznam katalogů, lze využít webových stránek www.seznamkatalogu.cz (*jedná se v podstatě o katalog katalogů*), které umožňují katalogy vyhledávat a filtrovat podle různých parametrů. Tento seznam můžeme velmi dobře použít pro vybudování zpětných odkazů (více informací o zpětných odkazech se nalézá v kapitole 3.1.2).

České

- *Seznam* — univerzální katalog webových stránek s odděleným katalogem firem (*firmy.cz*)
- *najisto.cz* — katalog firem s ověřenými kontakty vzniklý sloučením databází Centrum.cz a Atlas.cz

Zahraniční

- *Yahoo Directory* — první rozsáhlý katalog webových stránek
- *ODP (Open Directory Project)* — Největší lidmi budovaný internetový katalog. Díky přísným pravidlům pro přidávání nových stránek se jedná o velmi kvalitní databázi. Pro vlastní potřeby ji využívá i internetový vyhledávač Google. Existuje i česká sekce, která se však dle mých zkušeností potýká s problémem velmi nízké aktivity editorů.

2.2 Vyhledávače

Hlavním účelem vyhledávačů, jak již samotný název napovídá, je vyhledávání webových stránek s požadovaným obsahem. Při určování, která stránka se ve výsledcích zobrazí a na jaké pozici, se bere v potaz mnoho faktorů (některé z nich jsou popsány v kapitole 3.1). Největší roli by však měl hrát samotný obsah webové stránky, odtud plyne často používaný přívlastek *fulltextové* vyhledávače/vyhledávání [30]. To přináší velkou výhodu oproti katalogům, ve kterých jsou popisky velmi často nepřesné nebo se vůbec neshodují s vlastním obsahem stránek.

2.2.1 Popis funkčnosti

Funkčnost vyhledávačů lze rozdělit do těchto fází [26][24][16]:

- *Stahování*
- *Indexace*
- *Vyhledávání*

Stahování

O shromažďování dat se stará speciální program, pro který se v našich končinách nejvíce zažil název robot, používá se však i název crawler, bot či spider. Tento robot automaticky prochází známé webové stránky, ukládá jejich obsah pro následné zpracování a vyhledává na nich odkazy na další stránky, se kterými udělá totéž. Zároveň si udržuje seznam již navštívených stránek, aby nedocházelo k nesmyslnému stahování stále stejných stránek. Weby z tohoto seznamu jsou jednou za čas navštíveny znovu, aby se stáhl aktuální obsah.

Indexace

Stažená data je nutné zpracovat a uložit do databáze. Tato databáze obsahuje všechna nalezená slova a odkazy na stránky, které dané slovo obsahují. Je zřejmé, že tato databáze musí nabývat ohromných rozměrů. Sekvenční prohledávání takovéto databáze by zabralo neúnosné množství času. Proto se zavádí tzv. indexace dat. Tento proces vytvoří ze seznamu nalezených slov strukturu zvanou invertovaný seznam. V této struktuře je ke každému klíčovému slovu uložen seznam stránek, ve kterých se nachází. Toho se následně využívá při vyhledávání. Více informací lze nalézt v literatuře [13], [12], [22] a [29].

Vyhledávání

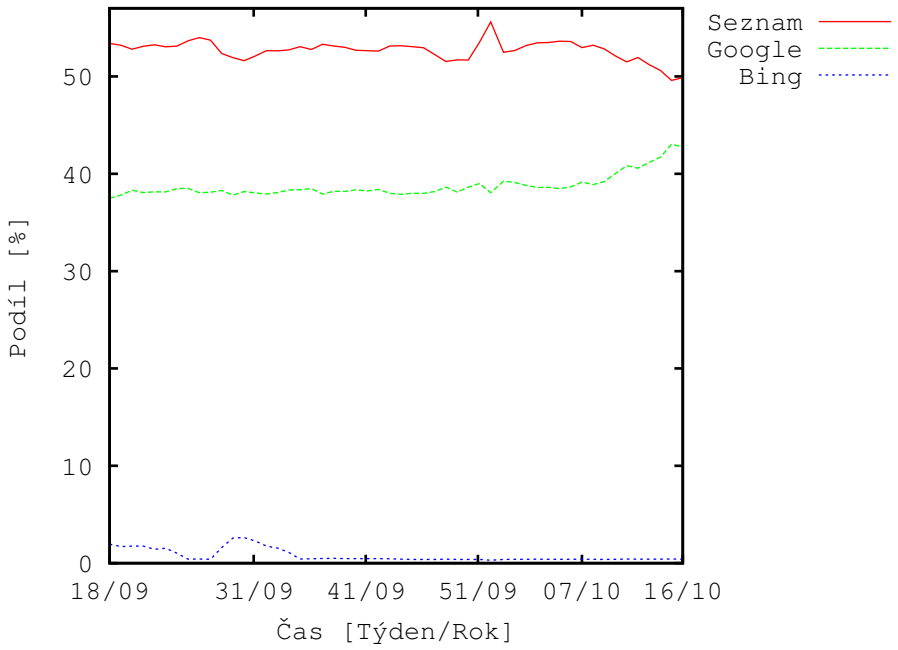
Díky indexaci můžeme při vyhledávání velmi snadno získat stránky, které obsahují všechna slova hledaná uživatelem. Odkazy na nalezené stránky se seřadí podle vyhledávačem stanovených kritérií (viz kapitola 3.1) a zobrazí uživateli. Pro zvýšení uživatelského komfortu bývá odkaz velmi často doplněn o titulek stránky, úryvek textu a další údaje. Je nutno zdůraznit, že samotné vyhledávání nikdy nemůže pracovat s plně aktuálními informacemi. Existují však mechanizmy, které zajišťují, aby roboti stránky s často se měnícím obsahem (např. zpravodajské servery) navštěvovaly častěji než stránky, které se mění jen výjimečně.

2.2.2 Nejpoužívanější vyhledávače v České republice

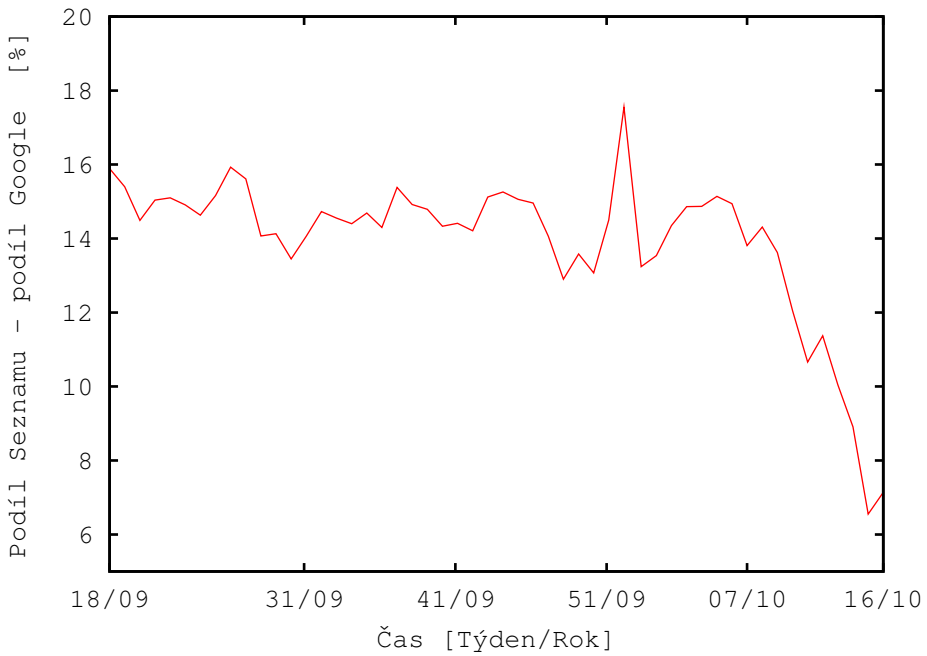
Pro zjištění, které vyhledávače se nejvíce používají v České republice, nám nejlépe poslouží globální statistiky některé ze společností nabízející monitorování přístupů na stránku. V této bakalářské práci bylo využito volně dostupných statistik od společnosti TOPlist, jejíž služby ke dni 30. 4. 2010 využívá celkem 781 982 webových stránek, což považuji za dostatečně reprezentativní vzorek českého webu. Statistiky byly naměřeny v týdnech *18/2009-16/2010*.

Ze statistik byly nejprve odstraněny podíly katalogů, vyhledávačů zboží a vyhledávače s podílem na trhu menším než 1%. Dále byly sloučeny podíly vyhledávače Google s ostatními službami, které ho pro vyhledávání využívají. Z těchto hodnot byl následně vytvořen graf (viz obrázek 2.1). Je zde jasně vidět, jak výsadní postavení mají vyhledávače Seznam a Google u nás. Spolu s nimi se v grafu umístil pouze vyhledávač Bing od Microsoftu, jehož podíl se během sledovaného roku dostal rovněž pod 1% a výhledově by se v tomto grafu neobjevil.

Zajímavé je také, jak se vyvíjel podíl ve vyhledávání Seznamu vůči Googlu (viz obrázek 2.2). Náskok podílu na trhu u Seznamu klesl z počátečních 15,89% až na hodnotu 7,14%. Google tedy ve sledovaném roce dokázal snížit náskok Seznamu více než o polovinu. Procentuální nárůst využívání Googlu se odehrál převážně v posledních dvou měsících, kdy se na internetu zvýšil výskyt reklam právě na Google. Je tedy otázkou, jestli si Google nové uživatele dokáže udržet či dokonce pokračovat v současném trendu získávání uživatelů. V kladném případě by mohl být již příští rok nejpoužívanějším vyhledávačem u nás.



Obrázek 2.1: Vývoj podílu vyhledávačů v období 18/2009 - 16/2010



Obrázek 2.2: Náskok Seznamu oproti Google 18/2009 - 16/2010

Kapitola 3

Optimalizace pro vyhledávače

Od doby, kdy vznikl první vyhledávač řadící jiným než abecedním způsobem, se lidé začali zabývat tím, jakými metodami lze ovlivnit pozici stránky ve výsledcích vyhledávání daných klíčových slov. Aplikace těchto metod je nazývána optimalizací pro vyhledávače nebo také *SEO* (od anglického výrazu *Search Engine Optimization*). Na dnešním ve většině oblastí přesyceném trhu nabývá optimalizace stále větší důležitosti a je zdrojem příjmu nejedné firmy.

3.1 Co ovlivňuje pozici webové stránky

Faktorů, které ovlivňují pozici webové stránky je velmi mnoho. Tyto faktory se rozdělují do dvou skupin [30][8]:

- *On-page faktory*
- *Off-page faktory*

Předtím, než se na jednotlivé faktory podíváme podrobněji, je důležité zmínit, že konkrétní váhy jednotlivých faktorů jsou vyhledávači důkladně utajovány.

3.1.1 On-page faktory

On-page faktory jsou takové faktory, které můžeme přímo ovlivnit zásahem do zdrojového kódu stránky nebo jejím umístěním. Pojďme se nyní na faktory patřící do této skupiny podívat podrobněji:

Kvalita kódu

Chceme-li, aby si vyhledávače z našich stránek odnesly maximum informací, je potřeba dodržovat několik základních pravidel:

- validita kódu — Validita kódu je jeden z velmi často přeceňovaných faktorů. To však neznamená, že bychom standardy při psaní stránek neměli dodržovat. Jejich dodržování nám zajistí, že vyhledávač bude schopen "pochopit" význam textu, který je dán sémantikou značek.
- zvýrazňování klíčových slov — Zvýrazněním klíčového slova je myšleno jeho umístění v tagu se speciálním významem (např. nadpis, tučné písmo, kurzíva, odkaz, meta informace apod.).

- používání tagů pro účely, pro které byly určeny (např. *tag h1* pro nadpis hlavní úrovně)
- udržování nadpisů ve strukturované formě (*h1* → *h2* → *h3* nikoliv *h2* → *h1* → *h3*)
- nadpis hlavní úrovně (*tag h1*) by měl být pouze jeden
- dodržování doporučené délky tagů

  - popisek stránky (*meta tag description*) — přibližně 250 znaků
  - klíčová slova stránky (*meta tag keywords*) — maximálně 5-8 slov
  - titulek stránky (*tag title*) — okolo 70 znaků

Obsah stránek

Obsah stránek je pro vyhledávače to nejdůležitější, proto bychom mu měli věnovat nejvíce pozornosti. Během vytváření obsahu bychom měli dbát především na:

- unikátnost obsahu — Je vhodné, aby každá stránka na webu měla unikátní text, titulek ale i meta informace. V opačném případě se můžeme od vyhledávače dočkat určité penalizace.
- volbu klíčových slov — Dříve než začneme plnit obsah stránek, je dobré si ujasnit, pro která klíčová slova budeme webové stránky optimalizovat. Při výběru klíčových slov můžeme využít statistik některého z vyhledávačů. V nich můžeme zjistit, jak často uživatelé dané klíčové slovo vyhledávají a jak velká pro toto slovo existuje konkurence. Někdy může být výhodné zvolit slova, jejichž vyhledávání není tak časté, ale je zde malá nebo dokonce neexistující konkurence.
- hustotu klíčových slov — Hustota klíčových slov nám udává počet výskytů klíčových slov vůči ostatnímu textu. Za optimální hustotu bývá označována hodnota mezi 3-7% (tedy 3-7 slov ze 100). Pokud je hustota daného slova příliš vysoká, mohou vyhledávače stránku penalizovat.
- co nejvíce textu — Stránky by měly obsahovat oproti html tagům maximum textu.

Umístění stránek

- struktura URL — URL adresa by měla být co nejkratší a obsahovat slova relevantní k obsahu. Dále není příliš vhodné, aby obsahovala parametry.
- rychlost webu — Pomalý web dokáže uživatele webu rychle odradit. Všimli si toho i v Googlu a proto v nedávné době začali rychlost stránek používat jako jeden z faktorů ovlivňující pozici stránky. Více informací lze dohledat v literatuře [6].

Správná indexace webu

Aby uživatelé mohli dohledat informace na celém našem webu, je nutné zajistit správné zaindexování celého webu vyhledávačem. Toho lze docílit pomocí:

- odkazů — Vyhledávač, stejně jako uživatel, by měl být schopen pomocí odkazů projít všechny stránky webu. Tyto odkazy by rovněž měly mít nějaký smysluplný text, vypovídající o obsahu cílové stránky. U stránek se složitější strukturou může být velmi užitečné klasické menu doplnit o tzv. drobečkovou navigaci [11].
- umístěním odkazu na mapu webu (tzv. *sitemap*) do souboru robots.txt — Jedná se o speciální XML soubor se seznamem webových stránek, který výrazně urychlí indexaci celého webu. Více informací lze nalézt v [4].

Může však nastat i situace, kdy nechceme, aby indexace některých částí webu proběhla. Tuto situaci lze vyřešit:

- souborem robots.txt — slouží mj. k zakázání/povolení indexace určité sekce webu. Toto nastavení lze provést globálně pro všechny roboty nebo pouze pro zvolené. V kódu 3.1 je uveden příklad takového souboru. Více informací lze dohledat v literatuře [5].
- meta tagem — Umožňuje zakázat indexaci konkrétní stránky (viz kód 3.2)
- odkazem s atributem rel nastaveným na *nofollow* — Roboti takovéto odkazy nenásledují (viz kód 3.3)

```
User-Agent: *
Disallow: /hidden/

User-Agent: GoogleBot
Disallow: /not4google/

sitemap: http://www.example.com/sitemap.xml
```

Kód 3.1: Ukázka souboru robots.txt

```
<html>
<head>
<meta name="robots" content="noindex,nofollow">
...
```

Kód 3.2: Ukázka použití meta tagu pro zamezení indexace celé stránky robotem

```
<a href="http://www.example.com" rel="nofollow">
        Robots don't follow this link
</a>
```

Kód 3.3: Ukázka odkazu, který robot nebude následovat

3.1.2 Off-page faktory

Jedná se o faktory, které nemůžeme přímo ovlivnit vlastní webovou stránkou. Mezi hlavní off-page faktory patří:

- počet zpětných odkazů — Čím více stránek odkazuje na naše stránky, tím lepšího výsledku je možné dosáhnout.
- kvalita zpětných odkazů — Kvalita odkazu je dána ohodnocením (viz 3.3) odkazující stránky .
- text zpětných odkazů — Je vhodné používat smysluplné a výstižné popisky (*U Lípy* vs. *Restaurace U Lípy*)
- relevantnost zpětných odkazů — Odkaz z podobně zaměřeného webu má větší váhu než z webu s úplně odlišnou tématikou.

3.2 Zakázané metody v SEO

Již jsme si popsali, co vše ovlivňuje pozici webové stránky ve vyhledávači. Nyní se podíváme na metody, kterých bychom se při optimalizaci měli vyvarovat, pokud nechceme, aby vyhledávač stránky postihl penalizací nebo dokonce úplným odstraněním z výsledků. Některé z uvedených metod jsou vyhledávače schopny detekovat automaticky, u jiných se musí spoléhat na nahlášení těchto praktik uživatelem vyhledávače. Informace čerpány v literatuře [30] a [9].

- skryté/nečitelně malé texty a odkazy
- doorway pages — Stránky s vysokou mírou optimalizace na daná klíčová slova, jejichž jediným cílem je odkazovat na domovskou stránku a zajistit tak její lepší pozici ve vyhledávači.
- cloaking — Vyhledávači se podstrčí stránka s jiným obsahem, než který je dostupný běžnému uživateli.
- duplicitní obsah — Obsah stránky je ve stejné nebo velmi podobné formě dostupný pod více adresami.
- komentářový spam — Vkládání nevyžadaných, nerelevantních příspěvků s reklamou na diskuzních fórech.
- odkazové farmy — Velké množství, většinou nepříliš kvalitních, navzájem propojených stránek. Díky vysokému počtu odkazů se dosahuje zvýšení hodnocení stránky vyhledávačem. Všichni účastníci známých farem bývají silně penalizováni.
- podvodné odkazy — Odkazy, které jsou většinou pomocí javascriptu přesměrovány na stránky s odlišným obsahem než bylo očekáváno.

3.3 Hodnocení webových stránek

Vyhledávače mají propracovaný systém hodnocení, pomocí kterého se vypočítává tzv. důležitost nebo také popularita webové stránky. Přesný způsob hodnocení webových stránek bývá vyhledávači většinou tajen. Při výpočtu hrají roli především off-page faktory. Výsledná hodnota je pak využívána jako jeden z faktorů při určování pozice ve vyhledávání. Hodnocení používají i roboti, kteří stránky s vyšším hodnocením navštěvují častěji.

- PageRank — Hodnocení stránek využívané Googlem. Byl pojmenován po jednom ze zakladatelů Googlu (*Larry Page*). Zjistit hodnotu PageRanku lze pouze orientačně. Tato hodnota se nazývá *Google Toolbar PageRank* (pokud v následujícím textu budeme mluvit o PageRanku, budeme mít namysli právě tuto hodnotu) a nabývá hodnot 0-10. Google tuto hodnotu umožňuje zjistit pomocí aplikace Google Toolbar, která se integruje do internetového prohlížeče. Více informací o PageRanku lze nalézt v literatuře [17] a [14].

- SRank — Hodnocení, které využívá Seznam. Ani Seznam neumožňuje zjistit přesnou hodnotu SRank. Přibližnou hodnotu lze zjistit pomocí Seznam Lištičky, což je obdoba Google Toolbaru od Googlu. I zde hodnoty nabývají hodnot 0-10. Další informace nalezneme v literatuře [7].

3.4 Časté nedostatky webových stránek

Tvůrci webových stránek většinou opakují jedny a ty samé chyby. Mezi nejčastější z nich patří:

- příliš mnoho klíčových slov — Velmi často se objevuje snaha stránky optimalizovat pro mnoho klíčových slov. To samozřejmě nepřináší tak dobré výsledky. Tuto situaci lze nejlépe vyřešit rozdělením jedné stránky do několika stránek zaměřených na užší výběr klíčových slov.

- neexistence mapy webu — Ani dobře prolinkované webové stránky nám nedokážou nahradit soubor s mapou webu. Pro vyhledávač je mnohem snažší zkontrolovat jediný dokument oproti změnám než procházet složitou strukturou celého webu a hledat změny od poslední návštěvy.

- nevhodně zvolené texty odkazů — Velmi často se můžeme setkat s texty odkazů s nulovou vypovídající hodnotou (např. *"zde"*).

- globálně nastavené META informace — Globálně nastavené META atributy bývají velmi častým jevem u dynamicky tvořených stránek. Autoři redakčních systémů často zapomínají do systému implementovat možnost jejich nastavení nebo jsou administrátoři při jejich vyplňování nedbalí.

- duplicita obsahu — Chyba postihující téměř všechny webové stránky. Přestože si tvůrce dává záležet na vytvoření originálního obsahu, na webu se může vyskytnout duplicitní obsah. Chyba bývá totiž velmi často v návrhu systému, který neprovádí tzv. kanonizaci URL, což je proces přesměrování duplicitních URL adres na jednu hlavní. Problém nastává většinou u úvodních stránek (*"www.example.com"* není to samé co *"www.example.com/index.html"*) a při přechodu na novou strukturu URL adres (původní adresy bývají velmi často ponechány, ale nejsou přesměrovány na nové, které jsou také dostupné). Více informací lze dohledat v [18] a [19]

3.5 Přínos pro běžné uživatele internetu

Pokud je optimalizace stránek provedena důkladně, dochází ke zkvalitnění zdrojového kódu, vlastních textů, prolinkování webu a v neposlední řadě také k posunu takového webu na přední pozice ve výsledcích vyhledávání. Z toho plyne několik výhod pro uživatele:

- relevantnější výsledky vyhledávání
- optimalizovaný web bývá zpravidla přístupnější pro osoby s různým zdravotním postižením
- snadnější orientace ve struktuře webové stránky

Kapitola 4

Nástroj pro testování optimalizace webových stránek

4.1 Specifikace požadavků

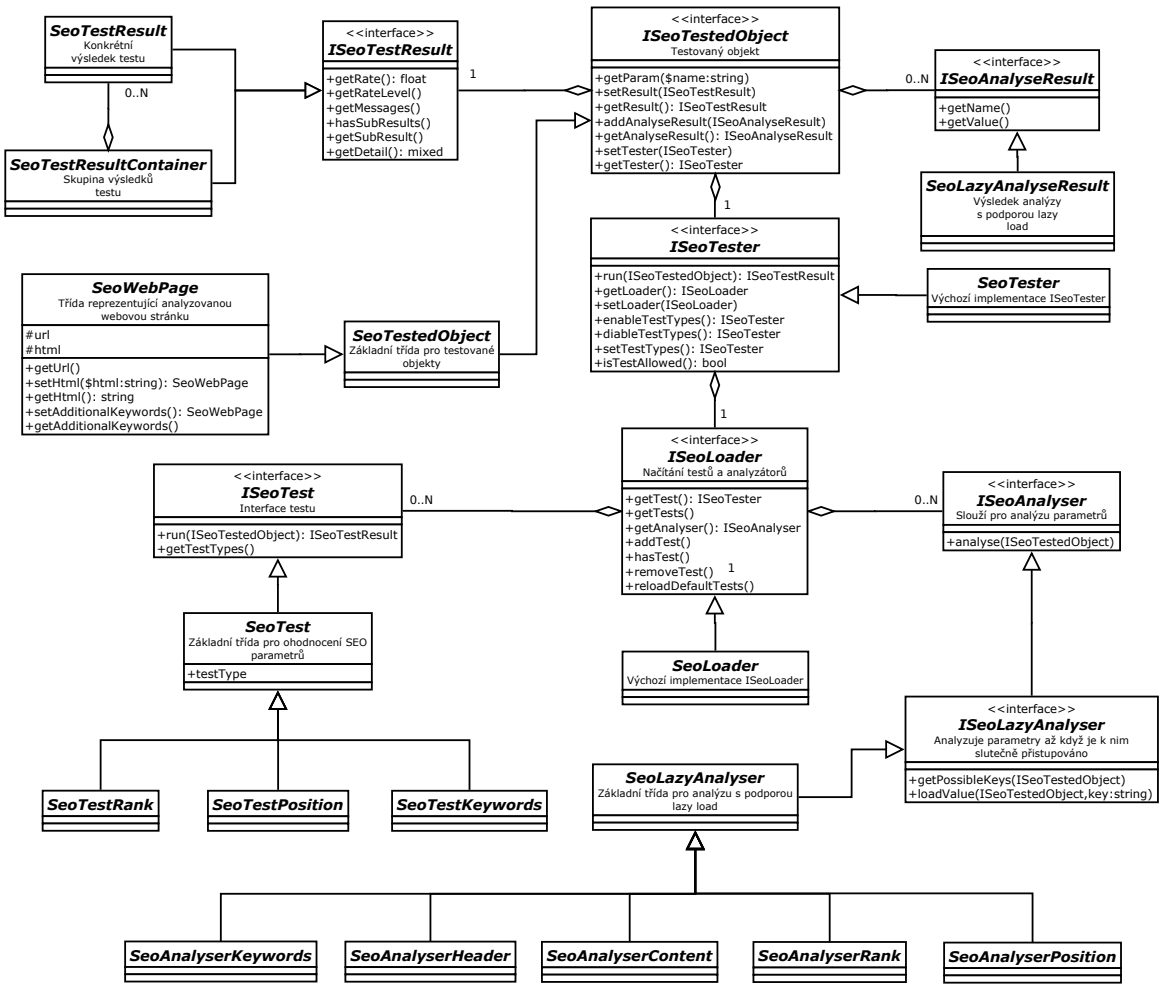
Nástroj by měl umožňovat hodnocení relevance stránky k daným klíčovým slovům, zjišťovat pozici ve výsledcích vyhledávání na Seznamu a Googlu a zjišťovat hodnotu SRank a PageRank. Měl by být snadno rozšiřitelný a měl by umožňovat snadné nahrazení jednotlivých částí. Důležitá je také snadná konfigurovatelnost a znovupoužitelnost nástroje. Posledním požadavkem na nástroj je vytváření tipů, jak dosáhnout lepších výsledků a jejich snadná lokalizace do zvoleného jazyka.

4.2 Návrh

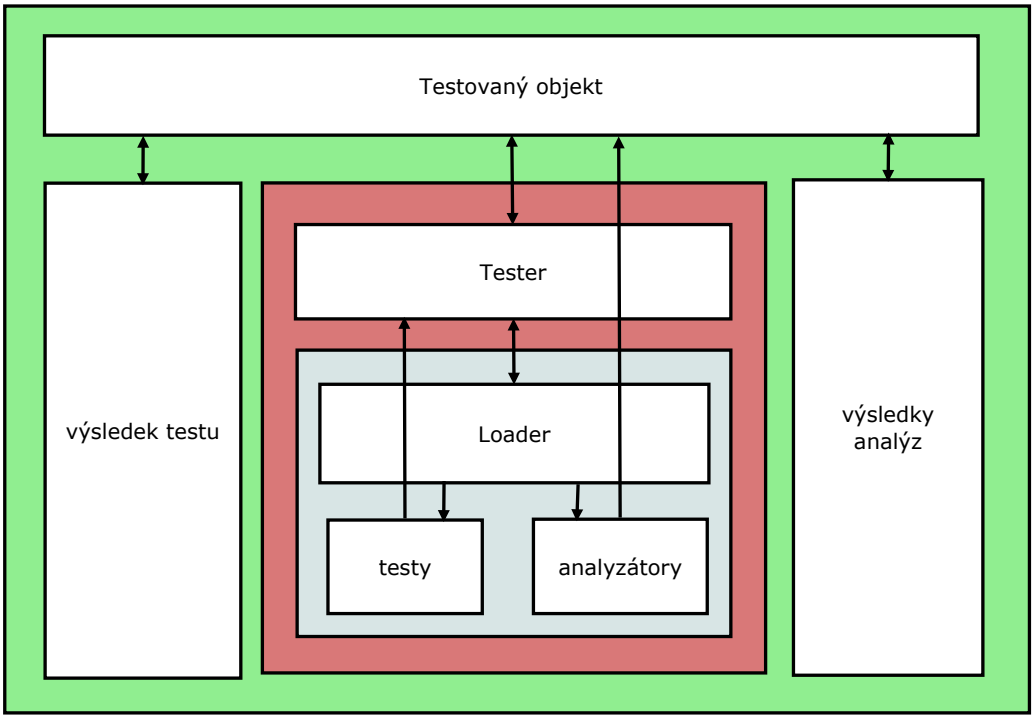
Nástroj jsem se rozhodl pro větší přehlednost rozdělit do několika částí, které budou definovány pomocí rozhraní. To umožní vytvořit nástroj, jehož jednotlivé části na sobě budou mít minimální závislost. Díky tomu bude možné v budoucnu některé části nahradit, což povede k dobré znovupoužitelnosti kódu. Na obrázku 4.1 si lze prohlédnout výsledek návrhu v podobě diagramu tříd. V dalším obrázku 4.2 se nachází blokové schéma znázorňující jak spolu jednotlivé části nástroje komunikují.

4.2.1 Analyzátory

Analyzátor (*ISeoAnalyser*) slouží ke zjišťování údajů využívaných v testech, případně v jiných analyzátorech. Tyto údaje jsou zjišťovány na základě parametrů získaných z *ISeoTestedObject*. Důvodem oddělení analyzátorů od vlastních testů je, že některé testy mohou vyžadovat stejné údaje a bylo by zbytečné je v každém testu zjišťovat znovu. Jelikož analyzátory často provádí časově náročné operace, byl navržen potomek *ISeoLazyAnalyser*, který k původní funkčnosti přidává provádění veškerých výpočtů až v době, kdy jsou zapotřebí. Toto chování bývá označováno jako *lazy loading* a je velice výhodné, pokud ve spuštěných testech nepotřebujeme znát veškeré údaje, které je analyzátor schopen zjistit. Tato vlastnost musí být podporována konkrétní implementací *ISeoAnalyseResult*.



Obrázek 4.1: Digram tříd



Obrázek 4.2: Blokové schéma znázorňující komunikaci mezi jednotlivými částmi nástroje

4.2.2 Testy

Testy (*ISeoTest*) využívají údajů zjištěných v analyzátorech pro ohodnocení, jak dobře si testovaný objekt stojí v dané oblasti. Jelikož vyhledávače dávají různým parametrům rozdílnou váhu, musí i testy vracet výsledky s určitou váhou. Pro výsledky bylo navrženo rozhraní *ISeoTestResult*, které mimo uložení výsledku a váhy testu umožňuje uchovávat různé informační zprávy, které se dají využít k zobrazení tipů pro uživatele nástroje. Dále zde jsou metody pro zjištění, zda tento výsledek obsahuje nějaké podvýsledky a jejich případné získání, aby bylo možné zobrazit více podrobností jak se k výsledku došlo.

4.2.3 Tester

Jádro celého nástroje tvoří tester, který je reprezentován rozhraním *ISeoTester*. Jeho hlavním účelem je spouštět nad objektem typu *ISeoTestedObject* testy, z jejichž výsledků je tvořeno celkové skóre. Seznam testů se získává za pomoci objektu typu *ISeoLoader*. *ISeoTester* dále umožňuje povolit či zakázat určité typu testů a zjistit, zda je daný test povolený či nikoliv.

4.2.4 Testovaný objekt

Nyní se konečně dostáváme k tomu, okolo čeho se celý nástroj neustále točí, a tím je testovaný objekt (*ISeoTestedObject*). Aby bylo možné tento objekt analyzovat, musí zde být určité prostředky pro získání známých parametrů (např. *html* pro získání html kódu). Dále zde jsou metody pro získávání a nastavování výsledků jednotlivých analýz a celého testu.

4.2.5 Loader

Loader (*ISeoLoader*) slouží k odstranění závislosti na názvech tříd analyzátorů a testů ze všech objektů, které je využívají. Na vyžádání vytvoří instanci požadovaného analyzátoru či testu a zároveň umožňuje spravovat seznam testů, který následně využívá *ISeoTester*.

4.3 Implementace

Implementace byla provedena na základě návrhu popsaného v kapitole 4.2.

4.3.1 Analyzátory

Jak lze vidět v diagramu tříd na obrázku 4.1, všechny navržené analyzátory mají společného předka, kterým je *SeoLazyAnalyser*. Za zmínku stojí pouze metoda *analyse*, která by měla analyzovat testovaný objekt a uložit do něj výsledek této analýzy. Jak jsme si ale již uvedli v návrhu, mnohdy by to znamenalo zbytečné provádění časově náročných operací. Analyzátor tedy vytvoří výsledek typu *SeoLazyAnalyseResult* s podporou lazy loadingu, do kterého uloží reference na sebe a na testovaný objekt. Když poté chceme přistoupit k některé hodnotě výsledku, provede se pomocí referencí její výpočet/zjištění. Nyní se pojďme podívat na to, jak byly implementovány jednotlivé analyzátory:

Analyzátory hlaviček a obsahu

Analyzátor hlaviček a analyzátor obsahu jsou si funkčně velmi podobné, proto se na jejich implementaci podíváme společně. Oba slouží ke zjišťování určitých hodnot ze zdrojového kódu stránky.

V analyzátoru hlaviček byla implementována podpora pro zjišťování titulku stránky, klíčových slov a popisku stránky z meta informací. Analyzátor obsahu je schopen zjistit hodnoty nadpisů první až třetí úrovně, texty odkazů a alternativní popisky obrázků.

Pro získání těchto hodnot bylo využito jazyka XPath [20].

Analyzátor hodnocení webových stránek

Jendá se o analyzátor sloužící k získání hodnot SRank a PageRank testované stránky. Implementace tohoto analyzátoru se ukázala jako velký problém, jelikož vyhledávače způsoby, jak tyto hodnoty zjistit, utajují a informací vedoucí k potřebnému cíli bylo tedy velmi málo. Byly nalezeny pouze již hotové skripty [10], u nichž nebylo možné ověřit, zda pro zjištění požadovaných hodnot využívají nejnovější metody a zda tedy budou fungovat i v budoucnu. Problém se podařilo vyřešit až s nápadem prozkoumat funkčnost lišt, které vyhledávače poskytují jako rozšíření internetových prohlížečů a zobrazují na nich právě požadované hodnoty. Po stažení instalačních balíčků pro *Firefox* bylo zjištěno, že jde o archivy obsahující mj. javascriptové soubory, které požadovanou funkčnost obstarávají. Algoritmy získané z těchto scriptů ukázaly, že předchozí obavy o aktuálnost nalezených řešení byly oprávněné. Nyní si ve stručnosti popíšeme:

- zjištění hodnoty SRank — Jak bylo zjištěno z kódu *Seznam lištičky*, pro zjištění hodnoty SRank se používá vzdálené volání funkce *getRank* pomocí XML-RPC požadavku zasílaného na adresu http://srank.seznam.cz. Tato funkce přijímá tři parametry:
  - číslo procesu prohlížeče — v implementaci nahrazeno náhodným číslem

  - url adresa stránky, pro kterou se má SRank zjistit
  - struktura obsahující číslo panelu v prohlížeči a kontrolní součet — Tato část byla v nalezeném scriptu nahrazena číslem 0, server pak nevrací hodnoty 0-10, ale hodnoty 0-255 (po úpravě hodnoty 0-100). Nelze však zaručit, že to bude fungovat i v budoucnu, proto jsem se rozhodl použít postup shodný se *Seznam lištičkou*.

- zjištění hodnoty PageRank — Nalezení algoritmu v *Google Toolbaru* bylo o poznání složitější, než v *Seznam lištičce*. Byla zde poznat snaha co největšího utajení této informace (odstraněné formátování, proměnné pojmenované jedním písmenkem). Zjišťování hodnoty *PageRanku* se provádí pouhým stažením obsahu z dané URL adresy, která obsahuje mj. tyto parametry:
  - url adresa stránky, pro kterou se má PageRank zjistit
  - *ch* — Jedná se o speciálně zakódovanou URL adresu. Google funkci provádějící toto zakódování nazval *awesomeHash*. Při jejím přepisování do jazyka PHP narazíme na problém neexistence operátoru >>>, tedy bitový posun vpravo s vkládáním nul zleva bez ohledu na znaménko [31]. Tento operátor byl nahrazen funkcí kombinující několik známých bitových operátorů. Za zmínku stojí také řetězec, který je pro zakódování použit. Píše se v něm, že zjišťování *PageRanku* je proti podmínkám služby Google. To bychom měli brát na vědomí, pokud budeme využívat výsledný nástroj. Algoritmus pro výpočet této hodnoty Google čas od času mění a lze předpokládat, že starší metody (např. ta z nalezeného scriptu) postupně přestanou fungovat.

Analyzátor klíčových slov

Analyzátor sloužící k vytvoření seznamů klíčových slov z určité části html kódu stránky. Pro detekci klíčových slov byl vymyšlen algoritmus umožňující detekci frází až do libovolně nastavené maximální délky:

- normalizace řetězce — odstranění html značek, převod html entit na znaky, odstranění přebytečných mezer apod.
- provede rozdělení řetězce do vět na základě oddělovacích znaků (např. *,!,;*), aby se nevytvářely fráze ze slov, které spolu nesouvisí
- pro každou větu:

  1. vytvoříme prázdnou frontu slov pro pomocné účely
  2. vytvoříme seznam obsažených slov a pro každé slovo:

     (a) navýšíme čítač celkového počtu slov (aby bylo možné dopočítat hustotu výskytu slova)
     (b) zkontrolujeme, zda se jedná o tzv. stop slovo, pokud ne, navýšíme čítač výskytu daného slova
     (c) pokud je maximální délka fráze větší než 1 pak:

i. uložíme slovo do pomocné fronty
ii. pokud je délka pomocné fronty větší než maximální délka fráze, je z fronty vymazán první prvek
iii. na základě slov uložených ve frontě navýšíme čítače všech podfrází, které nezačínají ani nekončí stopslovem

Analyzátor pozic ve vyhledávačích

Analyzátor pozic je schopen naleznout stránku ve výsledcích vyhledávání na Seznamu a Google na libovolné pozici až do nastaveného maxima. Nejprve je stažena stránka s výsledkem vyhledávání, ve které se poté vyhledávají odkazy na stránky opět pomocí XPath. Pro každou nalezenou adresu se zkoumá, zda se nejedná o stránku testovanou. Celý proces se opakuje, dokud není nalezen hledaný odkaz, dosaženo hledaného maxima nebo konce vyhledávání.

4.3.2 Testy

Před implementací jednotlivých testů byl, podobně jako u analyzátorů, vytvořen společný předek *SeoTester*, který provádí činnosti shodné ve všech testech. Dále bylo potřeba implementovat rozhraní *ISeoTestResult*, abychom měli do čeho ukládat výsledky testů. Pro tyto účely byly vytvořena třída *SeoTestResult* a *SeoTestResultContainer*. První zmiňovaná třída slouží k vytváření koncových výsledků, druhá slouží pro seskupování výsledků do skupin. *SeoTestResultContainer* vrací při požadavku na výsledek testu průměrnou hodnotu obsažených testů a průměrnou váhu testů. Následující sekce se budou věnovat konkrétním testům.

Test klíčových slov

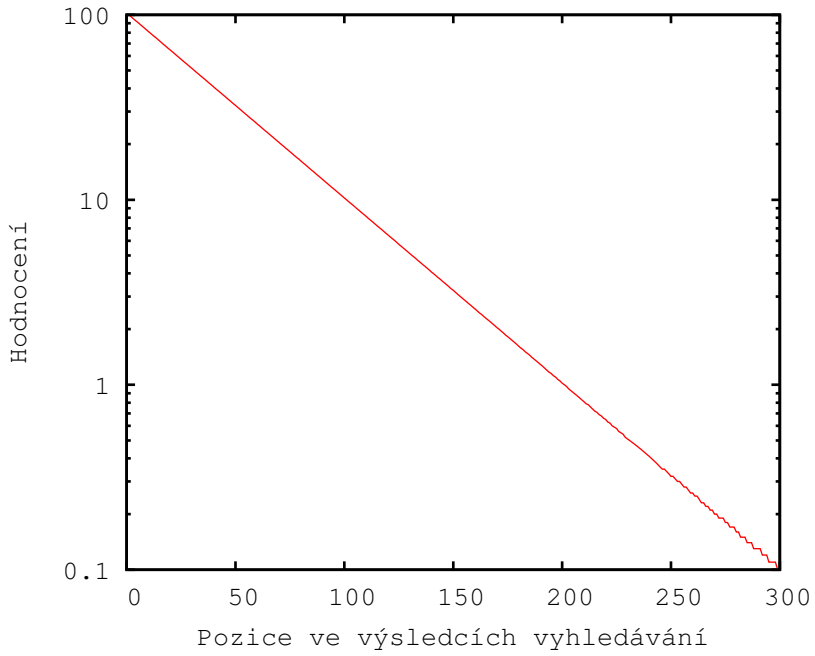
Test klíčových slov provádí hodnocení relevance stránky ke zvoleným klíčovým slovům na základě hustoty daného slova ve stránce. Dále hodnotí, zda je slovo umístěno v různých částech dokumentu (titulek, meta informace, nadpis apod.). V případě, že si nepřejeme výskyt v některé části testovat, můžeme upravit nastavení testu pomocí statických atributů. Výsledky jednotlivých částí jsou do sebe postupně zanořeny, takže lze detailně zobrazit, kvůli čemu jsme konkrétně dostali málo nebo naopak hodně bodů. K těmto výsledkům jsou navíc přidávány informační zprávy, které lze zobrazit uživateli.

Test pozic ve vyhledávačích

Tento test slouží k ohodnocení pozic ve výsledcích vyhledávání daných klíčových slov. Pro jednotlivá klíčová slova dojde k vytvoření podvýsledku, jehož hodnocení je dáno funkcí

$$0.1^{\frac{P}{100}-0.01} \cdot 100 \tag{4.1}$$

kde P je aktuální pozice stránky ve výsledcích vyhledávání. Průběh této funkce si můžeme prohlednout na obrázku 4.3. Takto vypočtené výsledky jsou poté opět seskupeny pomocí *SeoTestResultContainer*, čímž získáme výsledné hodnocení. Test umožňuje nastavit, pro které vyhledávače se má ohodnocení provádět.



Obrázek 4.3: Graf hodnocení testu pozic v závislosti na pozici ve výsledcích vyhledávání

Test hodnocení webových stránek

Test hodnocení vyhledávačů je nejjednoduším implementovaným testem. Vlastní ohodnocení se provádí pomocí velmi jednoduchého vzorce:

$$\frac{R}{R_{max}} \cdot 100 \tag{4.2}$$

kde R je dosažené hodnocení stránky a R_{max} je maximální hodnocení. Ani v tomto testu nechybí nastavení toho, která hodnocení se mají při vypočítávání konečného výsledku brát v úvahu.

4.3.3 Tester

Tester byl implementován v podobě třídy *SeoTester*. V této třídě je hlavní funkčnost (spouštění testů) obstarána v metodě *run*. Jsou zde za pomoci nastaveného loaderu získány všechny testy, které se mají nad testovaným objektem provést. Tyto testy jsou následně procházeny v cyklu a v případě, že jsou povoleného typu, pokusíme se získat jejich výsledek. To se nám nemusí povést vždy, protože některý z vyžadovaných parametrů testovaného objektu nemusí být dostupný (např. nemůžeme otestovat pozici ve výsledcích vyhledávání, když neznáme url adresu). Jednotlivé výsledky jsou opět seskupeny pomocí *SeoTestResultContainer*, který následně uložíme do testovaného objektu jako konečný výsledek.

4.3.4 Testovaný objekt

Pro testované objekty byl vytvořen předek *SeoTestedObject*, který obstarává základní činnosti jako je nastavení testeru, získání výsledku testů apod. Dále je zde mechanizmus, který

získávání parametrů mapuje na volání metod s patřičným názvem (pokud např. požadujeme parametr *html*, provede se volání metody *getHtml*).

Webová stránka

Webová stránka je implementovaná třídou *SeoWebPage*, reprezentující skutečnou webovou stránku pomocí základních dvou parametrů – URL adresa a html kód. Dále je zde možnost nastavení klíčových slov. Objekt typu *SeoWebPage* nelze vytvořit přímo, ale pouze pomocí metod *fromUrl* a *fromString*. Metoda *fromUrl* nejprve vytvoří instanci webové stránky a poté provede automatické stažení jejího html kódu.

4.3.5 Loader

Loader byl implementován pomocí velmi jednoduché třídy *SeoLoader*, která si udržuje seznam již vytvořených testů a analyzátorů. Vytvoření analyzátoru či testu je docíleno pomocí mapování jejich názvu na název třídy.

Kapitola 5

Ukázková aplikace

5.1 Specifikace požadavků

Ukázková aplikace by měla sloužit pro demonstraci funkčnosti a základního nastavení testovacího nástroje. Uživateli by měla umožnit zadání URL adresy testované stránky, zvolit klíčová slova a prováděné testy. Na základě těchto údajů bude aplikace spuštět testovací nástroj a zobrazovat detailní informace o výsledku spuštěných testů. Aplikace by dále měla být jednoduše rozšiřitelná o další nastavení a podporovat vícejazyčnost. Aplikace samotná by také měla jít příkladem a měla by se vyvarovat nedostatků webových stránek uvedených v kapitole 3.4.

5.2 Grafický návrh

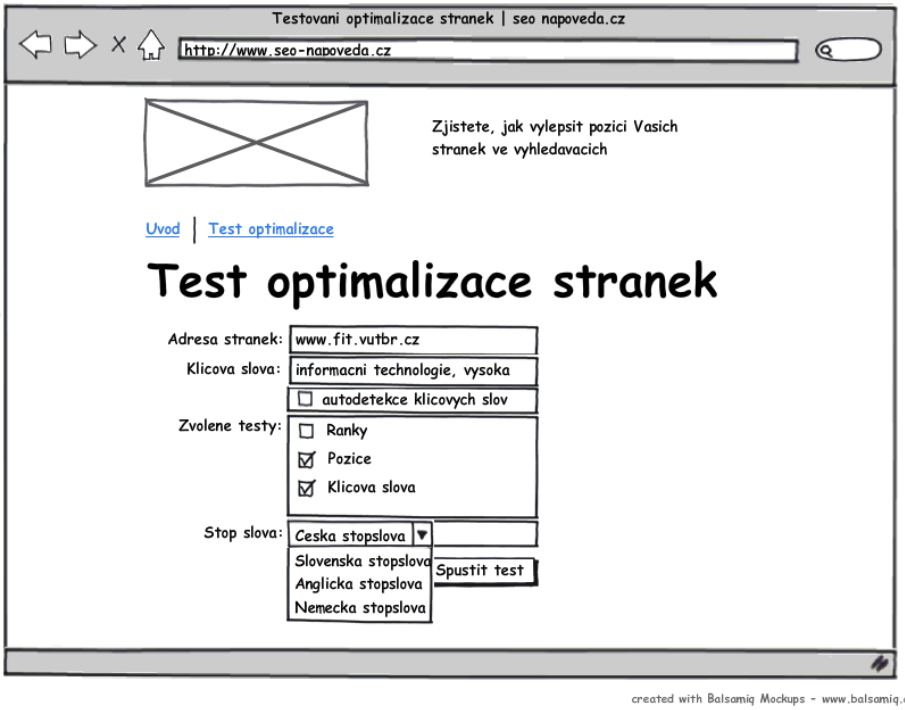
Během tvorby grafického návrhu bylo dbáno na přehledné a jednoduché ovládání. Pro vytvoření základního náčrtku, jak by měla výsledná aplikace vypadat, bylo využito nástroje Balsamiq [1]. Na obrázku 5.1 si lze prohlédnout návrh uživatelského rozhraní. Další obrázek 5.2 zachycuje představu o tom, jak by se měly výsledky provedených testů zobrazit uživateli.

5.3 Implementace

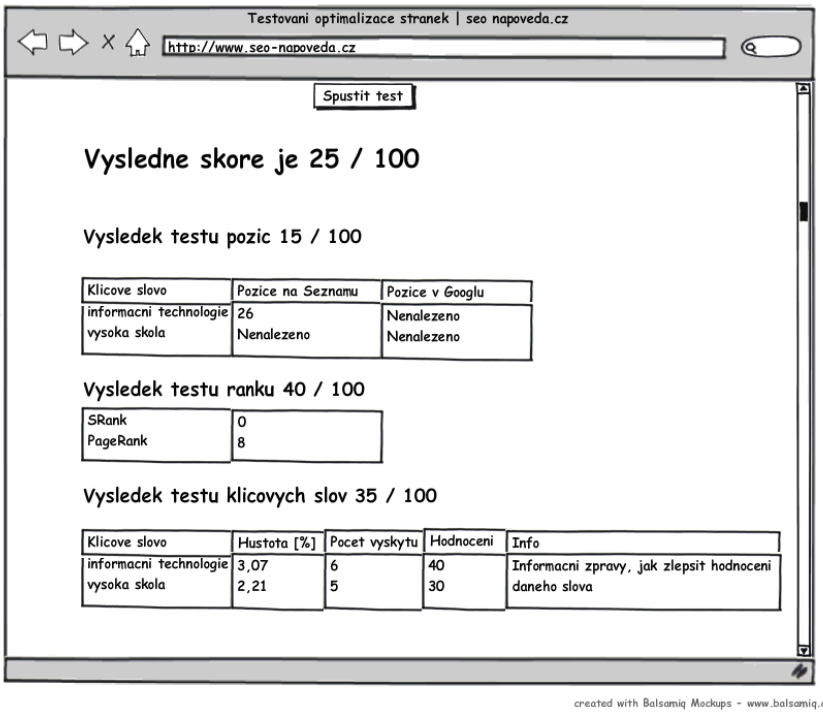
Pro implementaci ukázkové aplikace bylo využito MVC frameworku Nette, který výrazným způsobem usnadňuje vývoj webových prezentací. Jeho podrobnější popis se nachází v kapitole 6.5.

5.3.1 Vstupní formulář pro spuštění testu

Pro vstup od uživatele bylo využito nástroje pro tvorbu formulářů, který je součástí frameworku. S jeho pomocí byl vytvořen formulář s potřebnými vstupními poli a definováno několik jednoduchých pravidel. Tato pravidla slouží k validaci formuláře na straně serveru, ale zároveň i v klientské části, ve které je automaticky vygenerován obslužný kód nejen pro validaci, ale i skrývání některých částí formuláře v závislosti na vyplněných hodnotách. Pokud odeslaný formulář projde validací, je vyvolána nastavená obslužná metoda, která zajistí zpracování dat. V opačném případě dojde k zobrazení chybové hlášky uživateli. V obslužné metodě je vytvořen testovaný objekt *SeoWebPage*, nastaveny zvolené testy a poté je testovaný objekt předán do šablony.



Obrázek 5.1: Náčrtek uživatelského rozhraní ukázkové aplikace



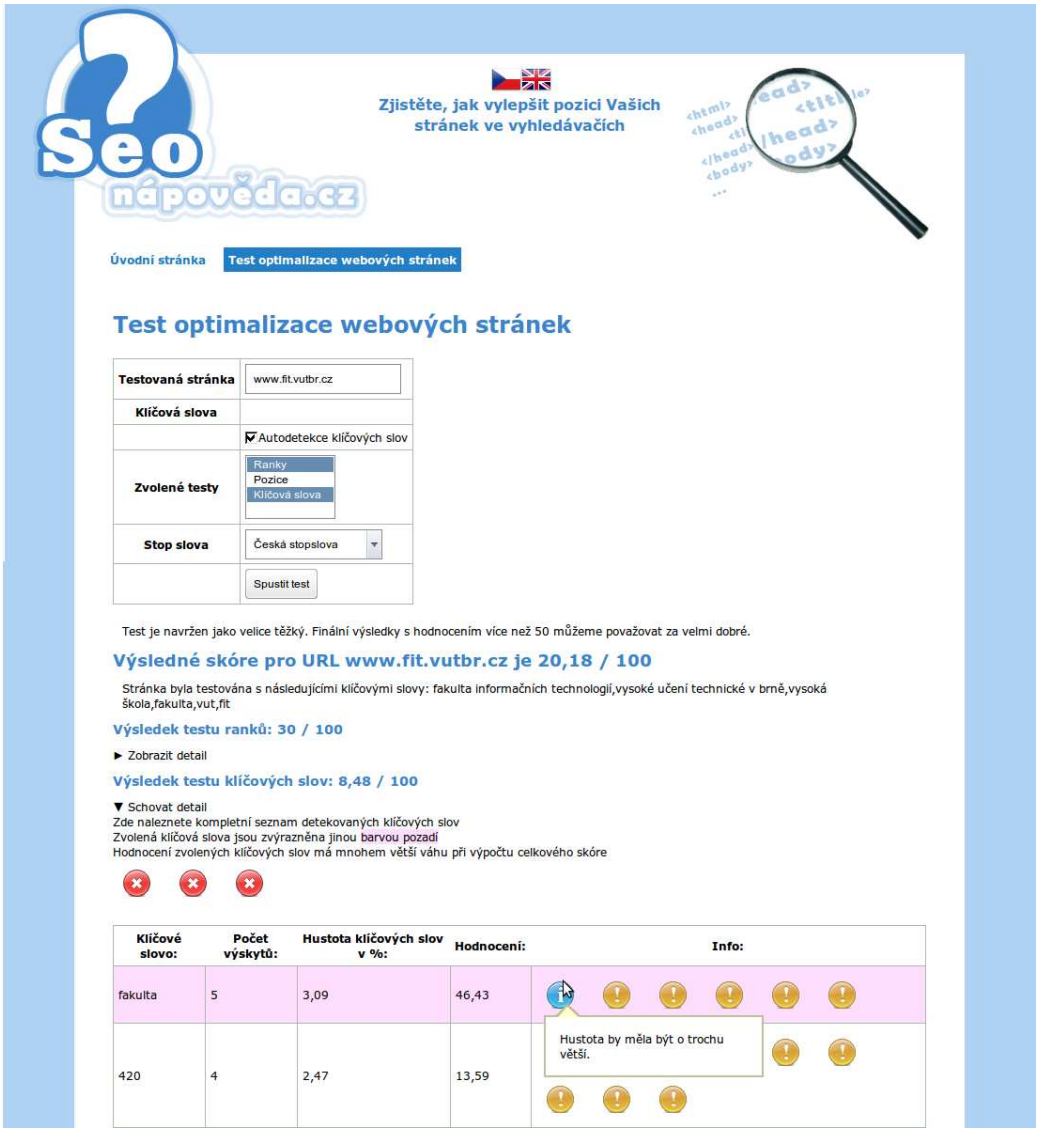
Obrázek 5.2: Náčrtek zobrazení výsledků testů

5.3.2 Zobrazení výsledků

Zobrazení výsledků probíhá v šabloně, výsledky jsou získávány přímo z objektu *SeoWebPage*, který je nastaven při odeslání formuláře. Nejprve je uživateli zobrazeno celkové skóre a poté podrobné výsledky jednotlivých testů. V případě spuštění testu klíčových slov dojde navíc k zobrazení informačních tipů, jak dosáhnout lepších výsledků. Pro zvýšení uživatelského konfortu a větší přehlednost byla pomocí javascriptu přidána možnost skrývání a zobrazování detailů. Výsledek si můžete prohlédnout na obrázku 5.3

5.3.3 Lokalizace

Jelikož ukázková aplikace i nástroj pro testování webových stránek byly vyvinuty v anglickém jazyce, bylo potřeba provést lokalizaci celé aplikace. K tomuto účelu bylo využito nátroje GNU gettext, pro který je v jazyce PHP nativní podpora. Nejprve byl ze zdrojových souborů získán seznam textů, které byly za pomoci aplikace *Poedit* přeloženy a následně uloženy do souboru, který je gettextem využíván.



Obrázek 5.3: Zobrazení výsledků v ukázkové aplikaci

Kapitola 6

Popis použitých technologií a knihoven

6.1 PHP

Jazyk PHP [21] je skriptovací jazyk pracující na straně serveru. Jedná se o jazyk primárně určený pro vytváření dynamických webových stránek, nic nám však nebrání vytvořit PHP script jako konzolovou aplikaci. Existuje zde dokonce možnost vytvořit klasickou desktopovou aplikaci včetně grafického uživatelského rozhraní pomocí projektu PHP-GTK [3]. Podstatnou výhodou PHP je jeho velmi vysoké rozšíření na webových serverech. Jedná se o dynamicky slabě typovaný procedurální jazyk s podporou objektového modelu. PHP je *case-sensitive*, ale existují zde výjimky pro názvy funkcí, tříd a metod. O vykonání PHP se stará interpret, který se spouští při nalezení značek obalujících PHP kód. Takto spuštěný kód je nahrazen jeho výstupem.

Od PHP verze 5, která byla pro implementaci použita, byla podstatně vylepšena podpora objektového programování. Stále je zde však co zlepšovat, jelikož standartní objekty v PHP dávají až příliš volnosti, což může vést k větší četnosti těžko odhalitelných chyb (např. přiřazení do neexistujícího atributu nezpůsobí chybu). Aktuální stabilní verze PHP 5.3 navíc přidala podporu pro jmenné prostory a spoustu dalších funkcí.

6.2 JavaScript

JavaScript je objektově orientovaný scriptovací jazyk. Přestože tomu název napovídá, nemá s jazykem Java společného nic víc, než podobnou syntaxi. Největšího rozšíření se dočkal mezi tvůrci www stránek díky rozsáhlé podpoře ze strany internetových prohlížečů, které jej na straně klienta interpretují. Pokud vyvijíme internetové stránky, měli bychom brát vždy na vědomí, že ne každý uživatel internetu má JavaScript zapnutý. Měli bychom ho tedy používat spíše na zvýšení uživatelského komfortu a nespoléhat se na jeho vykonání.

6.3 CSS

CSS (Cascading Style Sheet), neboli kaskádové styly slouží k definici vzhledu webových stránek, případně XML dokumentů. S jeho pomocí můžeme např. změnit barvu nadpisů, odkazů, zvětšit odsazení odstavců, barvu či obrázek pozadí apod. Pokud CSS umístíme do externího souboru, můžeme velmi snadno upravit vzhled celého webu z jednoho místa.

Lze také využít inline zápisu CSS kódu přímo do HTML atributu *style*, toho velmi často používají *WYSIWYG* [28] (*What You Seen Is What You Get*) editory webových stránek. V kódu 6.1 si lze prohlédnout ukázku obsahu CSS souboru.

```
body {
    color: #606060;
}

#header #top-navigation li.current {
    background: url(/images/current.gif) no-repeat left top;
    height: 23px;
}
```

Kód 6.1: Ukázka obsahu CSS souboru

6.4 Systém správy verzí - GIT

Správa verzí nám umožňuje evidovat průběh vývoje software a kdykoliv se vracet v historii, pokud zjistíme, že se provedené změny příliš nepovedly. Vysoký přínos přináší také do týmové spolupráce — systém automaticky hlídá kolize mezi jednotlivými členy týmu, umožňuje zjistit, kdo které změny provedl apod.

GIT [27] je distribuovaný systém správy verzí navržený Linusem Torvaldsem, tedy autorem operačního systému Linux. Oproti rozšířenému *SVN* přináší několik zásadních výhod:

- snadné vytváření a slučování větví
- uložiště lze jednoduše sdílet
- každý vývojář má lokálně uloženou kompletní historii změn, což lze považovat za velmi dobrý systém zálohování
- verzování funguje i lokálně, zveřejnit změny mohu až když jsem zcela spokojen s výsledkem
- lze vytvořit systém schvalování změn například zkušenějším programátorem
- veškerá meta data jsou uložena pouze v jednom adresáři

6.5 Nette framework

Pro jazyk PHP existuje velké množství frameworků, které výrazným způsobem zefektivňují a zpřehledňují napsaný kód. Nette framework [2] (dále jen nette) je jedním z nich. Přestože za vývojem nette stojí z velké části pouze jedna osoba, David Grudl, jedná se o velice povedený a v mnoha ohledech inovativní framework. Mezi hlavní přednosti patří vysoká bezpečnost, integrovaná podpora pro *AJAX* aplikace, kvalita ladících nástrojů a vysoký výkon.

Aplikace vyvinuté v nette mají svůj životní cyklus. Nejprve je z URL adresy vytvořen objekt žádosti. Takto vytvořená žádost je předána dál ke zpracování, vytvoří se objekt tzv. *Presenteru*. Pro každou část webu je výhodné mít jeden *Presenter*, který obstarává s ní spojené akce a následné zobrazení pomocí šablon.

Nette se nemusí využívat pouze jako celek, ale lze využít pouze některých částí. Velmi dobře můžeme využít např. třídu *Object* jako předka našich vlastních tříd. Tato třída je oproti třídě *stdClass* (která je v PHP používána standardně) více striktní, přidává podporu událostí, atributů pouze pro čtení a několik dalších vylepšení. Velkým usnadněním může být také třída *RobotLoader*, která ze zadaných adresářů vytvoří seznam všech tříd a obstarává automatické vložení potřebných souborů, když jsou zapotřebí.

Kapitola 7

Závěr

V této bakalářské práci byl navržen a následně úspěšně implementován nástroj provádějící hodnocení relevance stránky k daným klíčovým slovům, zjišťování pozice dané stránky pro daná klíčová slova ve výsledcích vyhledávání na Seznamu a Googlu a zjišťování hodnoty SRank a PageRank dané stránky.

Za přínos práce považuji především navržený a implementovaný systém testování, který by se dal jen s minimálními úpravami použít pro obecně jakékoli testování, dále algoritmus pro detekci klíčových slov a frází libovolné délky. Za přínosnou považuji také implementaci algoritmu pro získání PageRanku na základě zdrojových souborů Google Toolbaru.

Výsledný nástroj plánuji i nadále rozvíjet. Především bych chtěl rozšířit sadu testů, které by bylo možné spouštět. Nabízí se jich hned několik — kontrola kvality zdrojového kódu stránky, počet zpětných odkazů, existence mapy webu, rychlost odezvy stránek apod. Dále bych se chtěl více zaměřit na usnadnění prezentace samotných výsledků, aby došlo k minimalizaci logiky potřebné pro samotné zobrazení.

Prvním uplatněním této práce je zpřístupnění ukázkové aplikace veřejnosti, která díky tomu může nástroj využívat pro zkvalitnění vlastních webových stránek. Za tímto účelem byla registrována doména http://www.seo-napoveda.cz, na které se bude ukázková aplikace v budoucnu postupně rozrůstat.

Za nejvýznamnější možnost uplatnění však považuji propojení nástroje s redakčním systémem, který by před samotným uložením stránky provedl zhodnocení klíčových slov a poradil, co vše by se ještě mohlo v obsahu zlepšit. Pokud by tento redakční systém ukládal historii všech změn a současně si vedl statistiky pozic ve výsledcích vyhledávání, měl by správce webu dokonalý přehled o tom, jaké změny jsou pro vyhledávače nejvíce účinné a podobné úpravy následně aplikovat na dalších stránkách.

Dalším uplatněním nástroje by mohlo být díky jeho univerzálnímu návrhu např. testování rychlosti či paměťové náročnosti různých operací, to by však pochopitelně vyžadovalo vytvoření nových analyzátorů, testů a testovaných objektů.

Literatura

[1] Balsamiq [online]. http://balsamiq.com/.

[2] Nette - Hlavní přednosti [online]. http://nettephp.com/cs/hlavni-prednosti.

[3] PHP-GTK project [online]. http://gtk.php.net/.

[4] Sitemaps XML format [online]. http://www.sitemaps.org/protocol.php, 2008-02-27 [cit. 2010-05-06].

[5] About /robots.txt [online]. http://www.robotstxt.org/robotstxt.html, 2009-07-16 [cit. 2010-05-06].

[6] Using site speed in web search ranking. http://googlewebmastercentral.blogspot.com/2010/04/using-site-speed-in-web-search-ranking.html, 2010-04-29 [cit. 2010-05-06].

[7] Algoritmus [online]. http://napoveda.seznam.cz/cz/hledani-fulltext-algoritmus-vyhledavani-razeni-vysledku-faq-dotazy.html, 2010 [cit. 2010-05-06].

[8] Optimalizace webu [online]. http://help.seznam.cz/cz/fulltext-hledani-v-internetu/optimalizace-webu/, 2010 [cit. 2010-05-06].

[9] Zakázané optimalizační techniky [online]. http://napoveda.seznam.cz/cz/fulltext-hledani-v-internetu/ceho-se-pri-optimalizaci-vyvarovat/zakazane-optimalizacni-techniky/, 2010 [cit. 2010-05-06].

[10] PHP script na zjištění PageRanku a SRanku [online]. http://xrank.cz/download, 2010 [cit. 2010-05-07].

[11] Colter, A.: Drobečková navigace na webu [online]. http://interval.cz/clanky/drobeckova-navigace-na-webu/, 2006 [cit. 2010-05-08].

[12] Houdek, A.: Nové principy využívané v internetovských vyhledávacích strojích [online]. http://www.ikaros.cz/node/312, 1999 [cit. 2010-05-05], iSSN 1212-5075.

[13] Houdek, A.: Principy fungování vyhledávacích strojů [online]. http://www.ikaros.cz/node/1026, 1999 [cit. 2010-05-05], iSSN 1212-5075.

[14] Janovský, D.: Záhadný Google Toolbar PageRank [online].
http://www.lupa.cz/clanky/zahadny-google-toolbar-pagerank/, 2005
[cit. 2010-05-07], iSSN 1213-0702.

[15] Janovský, D.: Katalogy - Jak psát web [online].
http://www.jakpsatweb.cz/katalogy.html, 2010-04-28 [cit. 2010-04-28].

[16] Janovský, D.: Vyhledávače - Jak psát web [online].
http://www.jakpsatweb.cz/vyhledavace.html, 2010-05-01 [cit. 2010-05-01].

[17] Janovský, D.: Google PageRank [online].
http://www.jakpsatweb.cz/seo/pagerank.html, 2010-05-02 [cit. 2010-05-07].

[18] Janovský, D.: Duplicity a podobnosti [online].
http://www.jakpsatweb.cz/seo/duplicity-podobnosti.html, 2010-05-02
[cit. 2010-05-08].

[19] Janovský, D.: Přirozené duplicity a kanonizace [online].
http://www.jakpsatweb.cz/seo/kanonizace.html, 2010-05-02 [cit. 2010-05-08].

[20] Kolektiv autorů: XML Path Language. http://www.w3.org/TR/xpath/, 1999
[cit. 2010-05-12].

[21] Kolektiv autorů: *Programujeme PHP profesionálně*. Computer Press, 2002,
iSBN 80-7226-310-2.

[22] Kolektiv autorů: Index (search engine).
http://en.wikipedia.org/wiki/Index_(search_engine), 2010-04-20
[cit. 2010-05-02].

[23] Kolektiv autorů: Internetový katalog.
http://cs.wikipedia.org/wiki/Internetový_katalog, 2010-05-02
[cit. 2010-05-02].

[24] Kolektiv autorů: Internetový vyhledávač.
http://cs.wikipedia.org/wiki/Internetový_vyhledávač, 2010-05-02
[cit. 2010-05-02].

[25] Kolektiv autorů: Web directory. http://en.wikipedia.org/wiki/Web_directory,
2010-05-02 [cit. 2010-05-02].

[26] Kolektiv autorů: Web search engine.
http://en.wikipedia.org/wiki/Web_search_engine, 2010-05-02 [cit. 2010-05-02].

[27] Kolektiv autorů: Git (software). http://en.wikipedia.org/wiki/Git_(software),
2010-05-07 [cit. 2010-05-12].

[28] Kolektiv autorů: WYSIWYG. http://en.wikipedia.org/wiki/WYSIWYG,
2010-05-07 [cit. 2010-05-12].

[29] Pánek, K.: Šrotujeme text [online]. http://www.ikaros.cz/node/312, 2002
[cit. 2010-05-05], iSSN 1213-0702.

[30] Smička, I. R.: *Optimalizace pro vyhledávače - SEO*. Jaroslava Smičková, Dubany, 2004, iSBN 80-239-2961-5.

[31] Yehuda Shiran, Ph.D.: The Bitwise Right Shift Operator. `http://www.webreference.com/js/tips/991019.html`, 1999 [cit. 2010-05-14].

Dodatek A

Obsah DVD

- adresář *demoApp* — Zdrojové kódy ukázkové aplikace. Pro instalaci na hosting stačí zkopírovat obsah tohoto adresáře do kořenového adresáře webu a nastavit oprávnění k zápisu do složek *app/temp*, *app/log* a *public/temp*.
- adresář *toolApi* — programová dokumentace nástroje
- adresář *tex* - zdrojové kódy této písemné práce
- *xhakka00.pdf* - tato písemná práce v elektronické podobě

Dodatek B

Požadavky pro chod ukázkové aplikace

Pro správnou funkčnost testovacího nástroje je zapotřebí PHP ve verzi 5.2, které bylo při implementaci využito. Dále jsou vyžadována rozšíření *cURL*, *XML-RPC* a *gettext* pro jazyk PHP, která bývají standardně nainstalovány na většině hostingů. Server dále musí mít podporu pro soubor *.htaccess* a umožňovat přepisování URL adres pomocí *mod_rewrite*.