Univerzita Pardubice

Fakulta elektrotechniky a informatiky

Fulltextové vyhledávání ve webových stránkách firem

Miroslav Novosvětský

Diplomová práce

2010

Univerzita Pardubice
Fakulta elektrotechniky a informatiky
Akademický rok: **2009/2010**

ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Bc. Miroslav NOVOSVĚTSKÝ**
Studijní program:	**N2646 Informační technologie**
Studijní obor:	**Informační technologie**
Název tématu:	**Fulltextové vyhledávání ve webových stránkách firem**
Zadávající katedra:	**Katedra softwarových technologií**

Zásady pro vypracování:

- Implementovat webový fulltextový vyhledávač (ála Google). Pro procházení webových stránek je nutné nejprve použít crawler webových stránek a následně implementovat fulltextové vyhledávání nad získaným seznamem. Pro fulltextové vyhledávání je možné použít Java knihovnu Lucene. - Ověření funkčnosti a relevance vyhledávání bude na konkrétním seznamu webových stránek firem (v současné době je k dispozici testovací vzorek cca. 70 000 záznamů). - Výstupem bude webová aplikace, která po zadání hledaného slovního spojení vypíše hledané webové stránky firem seřazené podle relevance. Součástí výstupu každého hledaného záznamu bude také textový náhled na webovou stránku. Bude možné omezit vyhledávání firem podle kraje (funkce lokálního vyhledávání). - V interní databázi firmy Seico s.r.o. je dalších 200 tisíc firem z celé České republiky. Není v lidských silách standardními nástroji projít tento seznam firem a najít jejich webové stránky (pokud existují). Je tudíž nutné navrhnout a implementovat způsob pro efektivní získání dalších webových stránek firem. Předpokladem je, že výstupem analýzy bude nutnost vytvoření speciálně navržené aplikace pro sekretářku, která poloautomatickým způsobem projde zbývající firmy a pokusí se s pomocí aplikace nalézt webové adresy zbývajících firem.

Rozsah grafických prací:

Rozsah pracovní zprávy:

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

Seznam odborné literatury:

[1] GOSPODNETIC, Otis, HATCHER, Erik. Lucene in Action. [s.l.] : Manning, 2004. 456 s. ISBN 1932394281. [2] LOTON, Tony. Web Content Mining with Java. [s.l.] : Wiley, 2002. 320 s. ISBN 047084311X. [3] ECKEL, Bruce. Thinking in Java. 4th edition. [s.l.] : Prentice Hall, 2006. 1150 s. ISBN 0131872486. [4] SIERRA, Kathy. Head First Servlets and JSP. 2nd edition. [s.l.] : O'Reilly Media, Inc., 2008. 911 s. ISBN 0596516681. [5] Online zdroje na Internetu

Vedoucí diplomové práce: **Ing. Jiří Pinkas**
Katedra informatiky v dopravě

Datum zadání diplomové práce: **30. října 2009**

Termín odevzdání diplomové práce: **21. května 2010**

L.S.

prof. Ing. Simeon Karamazov, Dr.
děkan

doc. Ing. Antonín Kavička, Ph
vedoucí katedry

V Pardubicích dne 10. listopadu 2009

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

Souhlasím s prezenčním zpřístupněním své práce v Univerzitní knihovně.

V Pardubicích dne 18. 5. 2010

Miroslav Novosvětský

ANOTACE

Tato diplomová práce se zabývá možnostmi a využitím vlastního fulltextového vyhledávače pro vyhledávání ve webových stránkách firem. Teoretická část by měla čtenáři přiblížit co to fulltextové vyhledávání je, jak funguje a popsat možnosti a cesty implementace vlastního řešení. Praktická část se věnuje vlastní implementaci vyhledávače ve webových stránkách firem dostupného na adrese www.analyza-trhu.cz.

KLÍČOVÁ SLOVA

fulltextové vyhledávání, Java, Lucene, Regain

TITLE

Fulltext search in the company web pages

ANNOTATION

This Diploma thesis examines the topic of the possibilities of using a full-text search engine for searching the web sites of companies. The theoretic part introduces the reader to the full-text search engine, explains the process and describes the possibilities and techniques of implementing own solutions. The practical part is based on its own implementation of search engine companies in the web sites available on the www.analyza-trhu.cz.

KEYWORDS

fulltext search, Java, Lucene, Regain

Obsah

Seznam použitých zkratek

CERN – Evropská organizace pro jaderný výzkum.

FTP – (File Transfer Protocol) Protokol pro přenos souborů.

PERL – Je interpretovaný programovací jazyk.

MIT – Massachusettský technologický institut.

URL – Uniform Resource Locator („jednotný lokátor zdrojů“).

HTML – (HyperText Markup Language) značkovací pro tvorbu webových stránek.

PDF – (Portable Document Format) přenosný formát dokumentu.

1 Úvod

Tato diplomová práce reaguje na požadavek firmy Seico, s. r. o., na vytvoření fulltextového vyhledávače ve webových stránkách firem. Zaujala mě jak možnost dozvědět se více o fulltextovém vyhledávání, se kterým se každodenně setkávám, tak možnost vytvořit si vlastní vyhledávač a uplatnit tak znalosti získané, jak studiem na Univerzitě Pardubice, tak praxí ve firmě Seico, s. r. o.

Práce by měla poskytnout ucelený přehled o stávající situaci na poli fulltextových vyhledávačů a vyhledávacích technologií. Dále by měla čtenáři pomoci nahlédnout do problematiky vytvoření vlastního vyhledávání, popsat a analyzovat možnosti jeho vytvoření.

Praktická část bude rozdělena do dvou oddílů, které budou popisovat dva produkty naprogramované pro potřeby této práce. První oddíl se zabývá prvním produktem, kterým bude fulltextový vyhledávač ve webových stránkách českých firem. Tento vyhledávač bude schopen po zadání vyhledávaného výrazu vrátit seznam výsledků ve kterých se zadaný výraz vyskytuje.

Ve druhém oddílu bude popsán návrh a implementace aplikace, která bude sloužit pro poloautomatické doplnění stávající databáze webových stránek firem o další adresy firemních internetových stránek, které jsou nutné k vytvoření indexu pro vyhledávač.

2 Historie vyhledávání

Tato kapitola nám přiblíží českou a světovou historii vyhledávání. Dozvíme se zde nejen kdo a kdy na světě přišel s prvním podnětem na nutnost třídění a práci s informacemi, i to, jak se vyhledávání vyvíjelo v České republice.

2.1 Světová historie vyhledávání

Tato kapitola se převážně inspiruje z anglického článku „History of Search Engines: From 1945 to Google 2007“ [1].

Jako první přišel s nápadem uchovávání, sdílení a uspořádání vědeckých informací Vannevar Bush, který v roce 1945 v článku As We May Think publikovaném v časopisu The Athlantic Monthly navrhl projekt s názvem Memex, který jako první používá výraz hypertext.

Když se posuneme dále od prvního publikovaného nápadu na sdílení dat a informací k vlastnímu fungování celé internetové sítě, tak první, kdo se pokusil utřídit data o jednotlivých webserverech, byl Tim Berners-Lee. Byl to člověk, který vytvořil a staral se o seznam dostupných internetových stránek. Návštěvníci zde také mohli najít informace o hypertextu a technických detailech, které se vztahovaly k vytvoření jejich vlastních internetových stránek.

Bohužel tento způsob nemohl obstát a dlouho fungovat ve světě rychle se měnícího internetu, proto na scénu přišel první nástroj pro vyhledávání na internetu, kterým byl Archie (název byl vytvořen ze slova archive bez v). Vytvořil ho Alan Emtage, student na McGill University in Montreal. Princip byl založen na tom, že program stahoval výpis adresářů a všech souborů umístěných na veřejném anonymním FTP serveru a ukládal si tyto informace do databáze. Pak bylo

možné vyhledávat v názvech těchto souborů. Nicméně program ještě neukládal obsah stránek.

Se vzestupem Gopheru, což byla aplikační vrstva TCP/IP protokolu, podobně jako Word Wide Web (známý jako dnešní web), vznikly další dva nové programy Veronica a Junghead. Podobně jako Archie vyhledávaly názvy souborů a titulky uložené v Gopher index systému.

V létě roku 1993 už byly udržovány četné specializované katalogy pro web, ale ještě neexistoval vyhledávač. Oscar Nierstrasz na Univerzitě v Ženevě napsal PERL scripty, které zrcadlily tyto katalogy do jednotného formátu a vytvořily tak základ pro W3Catalog, což byl první jednoduchý vyhledávací stroj vydaný 2. září 1993.

V červnu roku 1993 Marrhew Gray na MIT vytvořil něco, co by se dalo nazvat jako první webový robot s názvem World Wide Web Wanderer, který využíval indexu s názvem „Wandex“. Wanderer byl využíván ke změření velikosti webu. Druhý vyhledávací webový stroj byl Aliweb, který se objevil v listopadu roku 1993. Aliweb nevyužíval robota pro prohledávání stránek, ale závisel na informacích, které mu předali administrátoři webových stránek pomocí indexového souboru v určeném formátu.

Dalším přelomovým vyhledávačem byl JumpStation vydaný v roce 1993. Byl to první vyhledávač, který dokázal zkombinovat principy vyhledávání jak je známe dnes. Jedná se o procházení, indexování a vyhledávání. Tyto principy budou později v této práci podrobněji popsány. Vzhledem k omezeným možnostem bylo vyhledávání omezeno pouze na hlavičky a titulky webových stránek.

Jedním z prvních fulltextových vyhledávačů byl WebCrawler, který se objevil v roce 1994. Na rozdíl od předchůdců byl schopen vyhledat jakékoli slovo

kdekoli v naindexované webové stránce. Tím se stal předlohou pro velké množství dalších vyhledávačů a jako první byl znám široké veřejnosti.

Brzy na to se objevilo mnoho dalších vyhledávačů, které soupeřili o popularitu. Můžeme zde zmínit například Magellan, Excite, Infoseek, Inktomi, Northern Light, AltaVista a Yahoo.

V druhé polovině 90. let se roztrhl pytel se společnostmi nabízejícími vyhledávání. Byla to doba, kdy se investoři prali o to, kdo bude moci zafinancovat tyto projekty.

V roce 1998 studenti Sergey Brin a Larry Page vytvořili prototyp vyhledávače Google. Brzy se ukázalo, že kvalita výsledků natolik převyšuje všechny dostupné vyhledávače, že doslova převálcoval ostatní hráče na trhu. Důvod byl jednoduchý, tito dva mladíci si kladli za cíl, že budou dělat jednu věc a to velice dobře. Tou věcí bylo vyhledávání. Byli také první, kdo řadil výsledky vyhledávání podle vlastního algoritmu PageRank.

Obrázek 1: Logo Google.com
Zdroj: http://www.google.com

Dalším, kdo nemohl zůstat pozadu v oblasti vyhledávání, byl Microsoft. Objevil se na trhu vyhledávačů a vyhledávacích technologií v roce 1998, kdy spustil vyhledávací službu MSN Search, která využívala výsledky vyhledávání od Inktomi. V roce 1999 začala stránka zobrazovat smíchané výsledky z Looksmart a Inktomi. V roce 2004 začal Microsoft s přechodem na vlastní vyhledávací technologii a vlastní web crawler s názvem Live Search, který vygradoval v roce 2009, kdy byl vydán nový vyhledávač Bing.

2.2 Česká historie vyhledávání

Jestliže se budeme bavit o historii českých vyhledávacích serverů, musíme zmínit čtyři hlavní hráče na našem trhu. Těmito hráči jsou Seznam, Atlas, Centrum a Jyxo.

2.2.1 Seznam

Seznam založil Ivo Lukačovič v roce 1996 ještě co by student vysoké školy. Seznam se stal vůbec prvním katalogovým vyhledávačem na českém internetu. To znamená, že se všechny odkazy na stránky zadávaly manuálně a později v nich bylo možné podle kategorií vyhledávat. Toto řešení bylo na tehdejší dobu naprosto dostačující, protože na českém internetu bylo odhadem 500 až 1000 stránek. Již v roce 1998 navštěvovalo seznam 100 tisíc lidí denně, což je na tehdejší dobu neuvěřitelné číslo. V roce 2002 Seznam navázal spolupráci s firmou Google a začal jako první využívat jejich vyhledávání. Uživatelé tuto změnu vítali, nicméně Seznam později přešel na spolupráci s vyhledávačem Jyxo, který jediný v té době umožnil skloňování a časování. Koncem roku 2005 se Seznam vydává vlastní cestou fulltextového vyhledávání. O tom, že vyhledávání Seznamu je velice kvalitní produkt vypovídá i fakt, že si jeho vyhledávání seznamu zanedlouho koupil i server Atlas.

Obrázek 2: Logo Seznam.cz
Zdroj: http://www.seznam.cz/

2.2.2 Atlas

Portál atlas.cz vznikl v roce 1997 a jako první nabídl českému uživateli full-textové vyhledávání. V roce 1999, kdy vznikla společnost Atlas.cz a.s do hry vstoupila společnost Microsoft a zahájila s Atlasem marketingovou spolupráci. Jednalo se v první řadě o to, že ho využila jako domovskou stránku pro prohlížeč Internet Explorer, který byl nedílnou součástí operačního systému Windows. Pokud jste spustili tento prohlížeč, jako první se vám zobrazil právě český portál Atlas.cz. To samozřejmě znamenalo ohromnou konkurenční výhodu.

Spolupráce trvala do roku 2001, kdy vstoupil do dění druhý investor – firma Epic Holding. Poté začala expanze na Slovensko a Ukrajinu, kde byly založeny sesterské portály atlas.sk a atlasua.net.

V únoru 2008 byla oznámena integrace dvojky a trojky českého vyhledávacího portálu Atlas.cz a Centrum.cz, o měsíc později vznikla nová společnost Centrum Holdings, která pod svá křídla mimo jiných projektů bere i Centrum.cz a Atlas.cz.

Obrázek 3: Logo Atlas.cz
Zdroj: http://www.atlas.centrum.cz

2.2.3 Centrum

Dalším českým projektem, který se zabývá vyhledáváním a o kterém bych se rád zmínil, je Centrum.cz.

Za zmínku stojí, že za tímto projektem stáli dva ambiciozní studenti magisterského studia Dopravní fakulty Univerzity Pardubice Ondřej Tomek a Oldřich Bajer.

V roce 1997 založili firmu NetCentrum s.r.o., ve které Ondřej Tomek zúročil předchozí zkušenosti nabyté studijní stáží ve Spojených státech. V roce 1999 přišel zlomový okamžik, kdy byl po několikaměsíčním ladění spuštěn portál Centrum.cz. Nemohli si vybrat vhodnější dobu než konec devadesátých let, kdy se investoři doslova prali o možnost zainvestování některého z rodících se internetových projektů. Nakonec po pětiměsíčním vyjednávání koupil menšinový podíl ve firmě Intel a jeden z fondů patřící bance ING.

Centrum se vyvíjelo – v roce 1997 došlo k prodeji firmy NetCentrum, v roce 2008 byla firma přejmenována na Centrum Holdings a tato firma zároveň koupila také server Atlas.cz.

Obrázek 4: Logo Centrum.cz
Zdroj: http://www.centrum.cz/

2.2.4 Jyxo

Jyxo je vyhledávač o kterém je nutné se v této kapitole zmínit. Nejedná se o jeden z prvních vyhledávačů, které se na českém internetu objevili, ale jedná se určitě na svoji dobu o naprosto jedinečný úhel pohledu na vyhledávání na poli českých internetových stránek.

Firma Jyxo s.r.o. zastřešující vyhledávač Jyxo byla Michalem Illichem založena v roce 2002, kdy už předchozí zmiňované vyhledávače dlouhou dobu fungovaly. Otázkou tedy zůstává, jak je možné, že si našla na českém trhu místo a přežila až do dnešních dob? Odpověd je jednoduchá, stejně jako Google se Jyxo ubíralo směrem velmi kvalitního vyhledávání bez dalších portálových služeb. Hlavní výhodou bylo, že od začátku se jednalo o vyhledávač fulltextový, který jako první uměl česká slova a výrazy skloňovat a časovat. Díky tomu jeho služeb využívaly další české vyhledávače, které neměly svoje vlastní fulltextové vyhledávání. Aktuálním vlastníkem této firmy potažmo vyhledávače je společnost CET 21, s. r. o., která mimo jiné provozuje televizi Nova a portál tn.cz.

Obrázek 5: Jyxo
Zdroj: http://www.jyxo.cz

3 Fulltextové vyhledávání

Fulltextové vyhledávání je způsob vyhledávání informací v databázi nebo textových souborech. Důležitý poznatek je, že nás nezajímá pouze nadpis dokumentu, ale fulltextové vyhledávání musí zajistit vyhledávání v celém textu internetové stránky nebo dokumentu.

Fulltextové vyhledávání funguje funguje tak, že uživatel zadá hledané slovo nebo frázi, která je porovnána s předem indexovanými dokumenty. Uživateli je vrácen dokument nebo dokumenty, které výsledku vyhovují a které jsou seřazeny podle určitých pravidel.

V případě webových vyhledávačů, kterým se budeme dále věnovat, se technicky jedná o tři základní stavební kameny, kterými je procházení stránek, jejich indexace a vlastní vyhledávání.

3.1 Procházení a indexace

Procházení a indexaci zmiňuji dohromady, protože to jsou témata, která samostatně při webovém vyhledávání existovat nemohou.

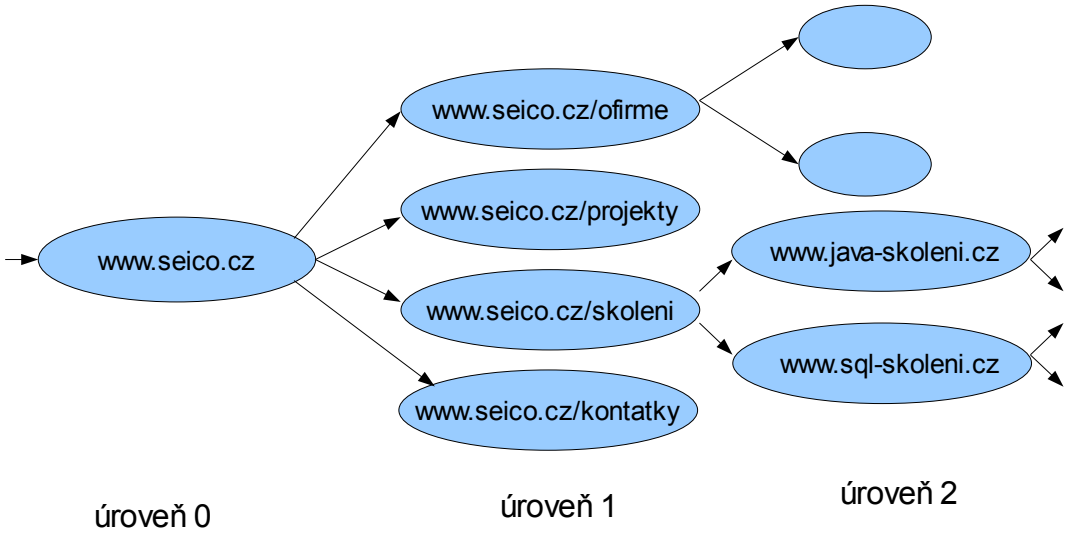
Pro procházení se používají crawlery. Jsou to programy, které nám umožňují projít a stáhnout webové stránky z určitého serveru nebo přímo z celé internetové sítě tak o což se snaží například Google.

Základní princip fungování crawleru je v tom, že se mu předá buď jedna vstupní webová stránka nebo seznam webových stránek, ze kterých se má začít procházet, a ukončující podmínka, například úroveň, po kterou se má procházet.

Program pak vstoupí do první webové stránky, získá odkazy, které jsou na ní obsažené a vytvoří si seznam dalších procházených stránek. Tento postup se na každé další stránce rekurzivně opakuje, dokud nedosáhneme nastavené úrovně nebo jiné ukončující podmínky.

Na obrázku 6 je vidět, jak crawler vstoupil na internetovou prezentaci firmy Seico s.r.o. (www.seico.cz). Tam našel odkazy na stránky o firmě, projekty, školení a kontakty, přičemž tyto stránky také navštíví. Na obrázku je také naznačeno, jak se dostane díky odkazům ve stránce školení na servery www.java-skoleni.cz a www.sql-skoleni.cz.

Další věcí, která je na obrázku zřetelná, je vysvětlení pojmu úroveň. V případě, že budeme brát stránku www.seico.cz jako vstupní, tedy na nulté úrovni, tak zde vidíme, že druhá úroveň jsou podstránky webu www.seico.cz a třetí odkaz na www.java-skoleni.cz a www.sql-skoleni.cz.

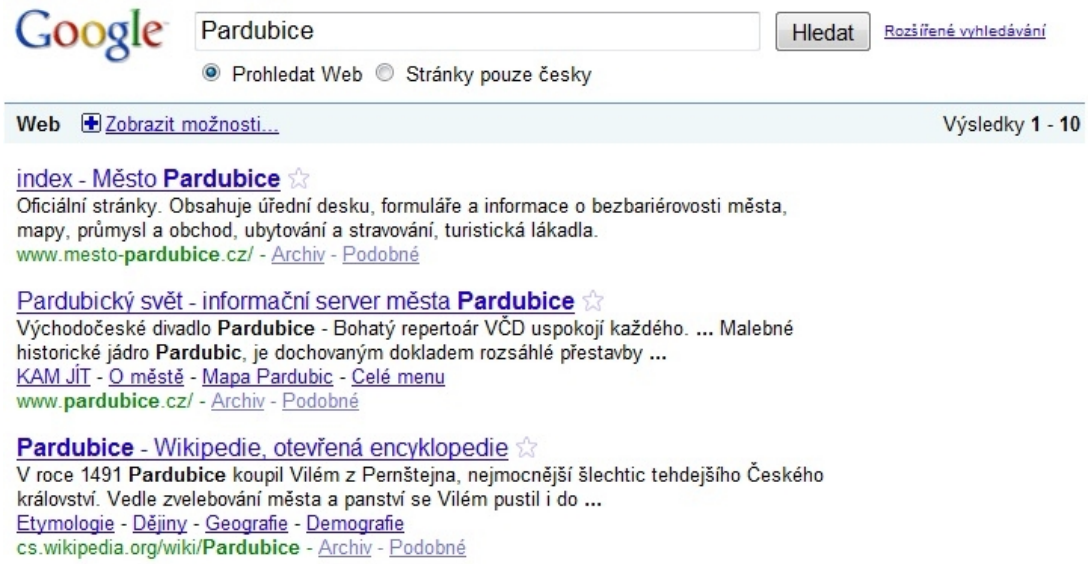


Obrázek 6: Procházení stránek

Při každém navštívení stránky je spuštěna indexace, která nám zajistí uložení obsahu dokumentu do indexačního souboru nebo souborů, ve kterých je možné později vyhledávat. Důvod pro tento postup je naprosto jednoduchý – není technicky možné vyhledávat v reálném čase nad aktuálními daty.

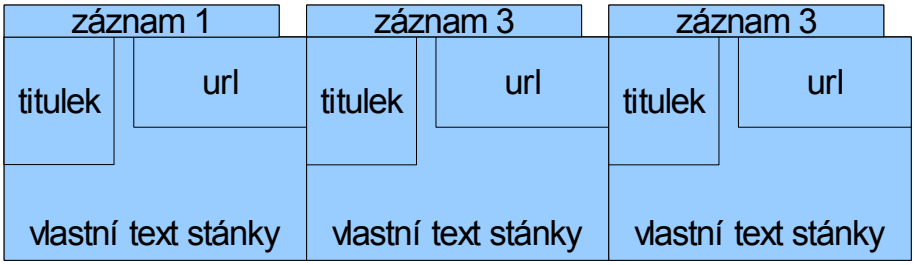
3.2 Vyhledávání

Nejprve se podíváme na vyhledávání z uživatelského pohledu. Jedná se způsob, kdy uživatel spustí některý z vyhledávačů a zadá klíčové slovo. Během několika okamžiků vidí výsledky seřazené podle relevance od nejpřesnějšího (nejrelevantnějšího) výsledku. Následující obrázek znázorňuje první tři výsledky, které nám Google vrátí jako výsledek, když zadáme do vyhledávání „`Pardubice`“. Následně má uživatel možnost kliknout na některý z vyhledávaných odkazů a přejít na konkrétní stránku.



Obrázek 7: Google vyhledávání
Zdroj: http://www.google.com

Co se ale děje na pozadí vyhledávání? Hledaný výraz je předán programu, který podle určitého algoritmu prohledá soubor s indexem ze kterého je na základě dotazu schopen vrátit pole výsledků. Obrázek číslo 8 nám ukazuje jak může takové pole vypadat. Je to zjednodušený teoretický pohled, který nám zobrazuje, že s každým záznamem je naindexováno nejen vlastní tělo stránky, ale také její titulek a URL adresa. Z těchto informací se následně sestaví výsledek vyhledávání. Samozřejmě předpokladem pro tuto funkcionalitu je to, že informace musejí být do indexu také uloženy s poli URL, titulek a vlastním textem stránky.



Obrázek 8: Vrácené záznamy vyhledávání

3.2.1 Vyhledávací operátory

V kapitole vyhledávání určitě nesmíme zapomenout na vyhledávací operátory. Tato podkapitola určitě není kompletní seznam všeho co můžeme v dotazu na vyhledávač použít, ale spíš by nám měla dát představu o tom co to vyhledávací operátory jsou k čemu, proč a jak se používají. Vyhledávací operátory jsou nepsaným pravidlem, které dnes téměř všechny vyhledávače zohledňují, ale nejsou žádným způsobem standardizované.

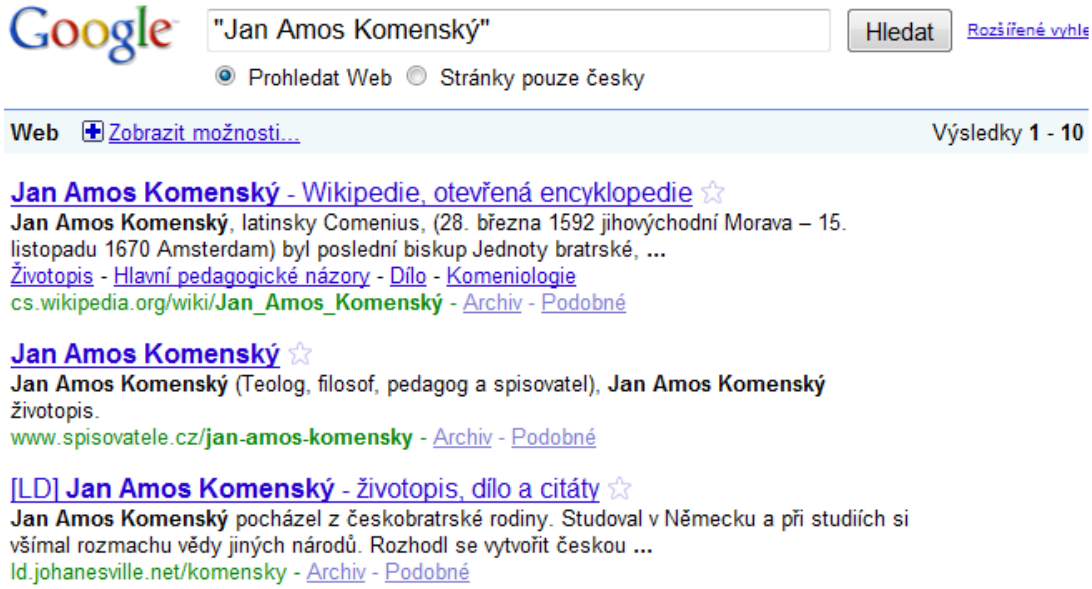
Vyhledávání bez operátoru

V případě vyhledávání například textu „`Univerzita Pardubice`“ se může ve výsledku objevit jak stránka, která obsahuje celý výraz, tak stránka, která obsahuje jen jedno z vyhledávaných hesel. Je to tím, že se zde automaticky využije operátor sjednocení OR o kterém si podrobněji povíme. Je ale nutné si uvědomit, že se každý vyhledávač může chovat při nezadaném operátoru odlišně. Vyskytují se výjimečné případy, kdy mezi termíny vyhledávač umístí operátor průniku AND, o kterém bude také ještě řeč.

Operátor „uvozovky“

Toto je operátor, který nám umožní hledat pouze přesný zadaný výraz. Jako příklad uvedu vyhledávání výrazu „`Jan Amos Komenský` “ (který je třeba zadat

v uvozovkách), tedy "`Jan Amos Komenský`" V tomto případě budou vráceny pouze výsledky, ve kterých se tento výraz vyskytuje v přesném znění. Například stránky ve kterých se vyskytuje J. A. Komenský ve výsledku vráceny nebudou.



Obrázek 9: Google vyhledávání pomocí uvozovek
Zdroj: http://www.google.com/

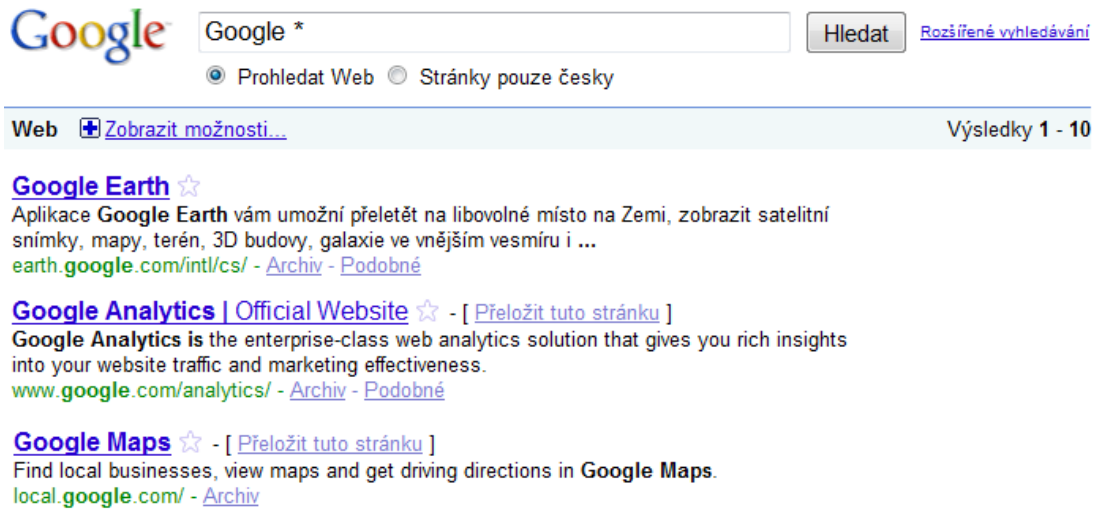
Operátor „mínus“

Tuto volbu využijeme, když chceme z výsledku vyhledávání odstranit stránky, které obsahují nějaký výraz. Například budeme chtít vyhledat všechny univerzity kromě Pardubické. V tomto případě zadáme vyhledávači „`Univerzita -Pardubice`“.

Operátor „hvězdička“

Ať už se bavíme o hvězdičce při obyčejném vyhledávání na pevných discích nebo ji využijeme pro vyhledávání na internetu, obvykle vystupuje jako zástupný znak nebo zástupné znaky. Tím chci říct, že si pod ní můžeme představit cokoli. V případě, kdy budeme chtít vyhledat všechny produkty Google stačí nám zadat do

vyhledávání „`Google *`“ a vyhledávač nám vypíše výsledky pro Google Earth, Google Analytics nebo Google Mapy viz. obrázek číslo 10.

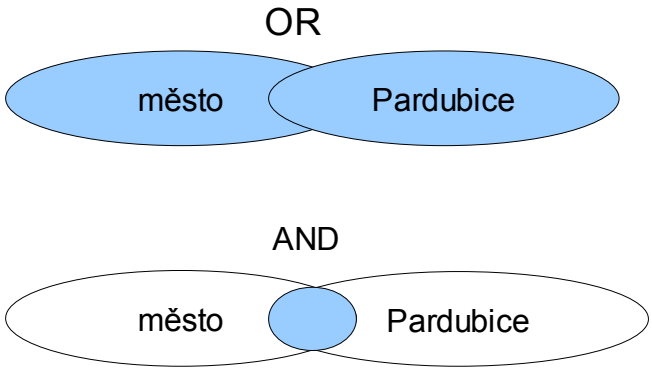


Obrázek 10: Google vyhledávání pomocí hvězdičky
Zdroj: http://www.google.com/

Operátory OR a AND

Tyto dva operátory jsou v podstatě množinové funkce. Operátor OR nám zastupuje sjednocení množin. To v praxi znamená, že když zadáme do vyhledávače text „město OR Pardubice“, zajistí nám to, že budou vyhledány výsledky s textem město Pardubice, dostihy Pardubice nebo město Hradec Králové.

Na druhou stranu AND je průnik množin a díky tomu bude vyhledávač při zadání „město AND Pardubice“ vracet pouze výsledky, kde se vyskytují tato dvě slova. Ale pozor, text nemusí být ve schodném pořadí a vedle sebe. Oba případy jsou vysvětleny na dalším obrázku.



Obrázek 11: Operátory OR a AND

3.3 Možnosti vyhledávání ve stránkách firem

Když se podíváme na stávající stav možností zjištění informací o firmách na českém trhu, tak zjistíme, že nemáme jinou možnost než zjišťovat informace o firmách pouze ve velkém množství informací, které nám poskytují různé katalogy. Zamyslíme-li se nad těmito stávajícími možnostmi vyhledávání ve stránkách firem, tak jsou velice omezené.

Nejznámějším službou pro hledání informací o firmách v České republice je v současné době katalog Firmy.cz. Na tomto serveru sice vyhledávání firem najdeme, ale není to fulltextové vyhledávání, které by nám umožnilo prohledávat přímo webové prezentace firem, ale prohledává pouze informace uložené v tomto katalogu.

Jednou z hlavních nevýhod katalogů je, že do nich jednou někdo informace uloží, ale už se nestará o to, zda jsou stále správné. Mnohdy se stává, že adresa firmy už dávno neplatí, URL adresa firemních stránek není dostupná nebo je firma dávno v konkurzním řízení a nefunguje, ale v katalogu ji stále najdeme.

Proto jsme se rozhodli vytvořit aplikaci, která by průběžně indexovala internetové stránky firem a uživateli nabídla přehledné vyhledávání v aktuálních informacích.

4 Způsoby třídění výsledků

Jednou z nejdiskutovanějších otázek v oblasti fulltextového vyhledávání je, jak třídit výsledky vyhledávání. V následující kapitole budou popsány základní metody řešení tohoto problému.

4.1 Meta-vyhledávače

Meta-vyhledávače nabízejí uživateli alternativní možnost vyhledávání. Jedná se o princip, kdy poskytovatel služby nemá žádný svůj vlastní zdroj dat (index), ale parazituje na zdrojích dat ostatních vyhledávacích služeb. Nabízí však klasické políčko pro zadání dotazu a vrátí množinu výsledků na jaké jsou uživatelé zvyklí.

To v praxi znamená, že odešle dotaz na vyhledávače Google, Yahoo a další, vrácené výsledky si přetřídí a vrátí uživateli seznam stránek, které dotazu vyhovují.

Na první pohled se může zdát, že je to nejsnadnější cesta k vlastnímu vyhledávání, protože se nemusíme starat o index, který je nutné průběžně aktualizovat, máme více zdrojů informací a můžeme si relevanci upravit podle vlastního uvážení.

Bohužel díky odeslání dotazu na více služeb najednou, následnému zpracování a vypsání výsledků se vyhledávání podstatným způsobem zpomalí.

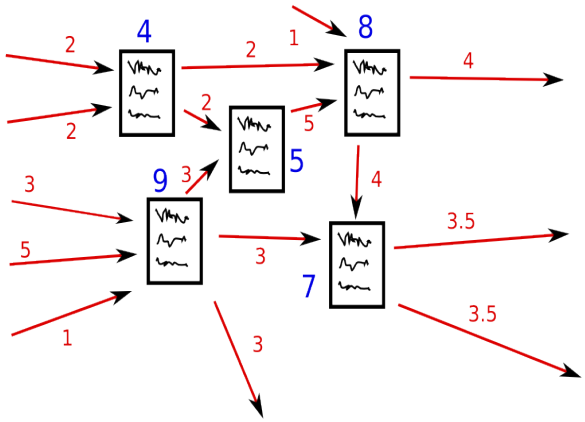
Příkladem takového vyhledávání jsou například servery Dogpile, Excite, InfoDump, Kartoo, MetaCrawler, Meraureka, Search.com a WebCrawler. [7]

4.2 Hodnocení relevance dokumentů

Vyhledané informace jsou ve výsledku vyhledávání vypsány a seřazeny podle určitých kritérii. Je mnoho faktorů, které jsou důležité pro ovlivnění pozice konkrétního výsledku. Nejprve se podíváme na způsoby, kterými se řeší pasivní upřednostnění jednotlivých stránek pomocí takzvaných Ranků. Ty jsou použity jako pomocná proměnná při řazení výsledků, to znamená upřednostnění bez ohledu na hledaný výraz a informace na stránce uložené.

4.2.1 PageRank

Je veličinou, kterou vymyslela společnost Google. Pracuje na naprosto jednoduchém a známém principu udělování a přijímání hlasů. Co ale jednotlivé hlasy ve světě internetu jsou? Jsou jimi právě hypertextové odkazy, kterými je provázán celý internet. PageRank interpretuje odkaz ze stránky A na stránku B jako hlas stránky A pro stránku B. Technologie PageRank poté vyhodnotí důležitost stránky podle získaných hlasů. Podrobněji nám to vysvětlí následující obrázek.



Obrázek 12: PageRank
Zdroj: http://cs.wikipedia.org/wiki/PageRank

4.2.2 Rovnice pro výpočet PageRanku [9]

Google zveřejnil základní rovnici pro výpočet PageRanku. Je to rovnice, která se v uvedené podobě již nepoužívá, přesná podoba aktuálního algoritmu není zatím známa a je obchodním tajemstvím společnosti Google.

$$PR(A) = (1-d) + d\,(PR(T1)/C(T1) + ... + PR(Tn)/C(Tn))$$

Vysvětlení veličin:

- **PR(A)** je PageRank stránky A
- **PR(T1) – PR(Tn)** je PageRank stránek, které odkazují na stránku A
- **d je tlumící faktor,** jehož hodnota se pohybuje mezi 0 a 1
- **C(Tn)** je počet odkazů na stránkách, které odkazují na stránku A

4.2.3 Srank

Srank je ukazatel, který využívá k hodnocení vyhledaných stránek český vyhledávač Seznam. Základní princip je podobný jako u PageRanku – zohledňuje odkazy, které na stránku míří, ale i ty, na které stránka odkazuje. Základní algoritmus Sranku není znám a je podobně jako aktuální podoba PageRanku obchodím tajemstvím společnosti Seznam.

4.2.4 Alexa rank

Jedná se o ukazatel společnosti Alexa, která se k vyhodnocování stránek postavila opět jiným způsobem. Společnost poskytla uživatelům plugin do vyhledávače a začala sbírat informace o návštěvnosti webových stránek. Hodnoty tohoto ranku se pohybují od 1 do 10 000 000 – podle počtu návštěvníků vašich stránek. Čím je tato hodnota nižší, tím je návštěvnost vyšší.

Bohužel tento princip naráží na geografické rozvržení uživatelů s nainstalovanou lištou v prohlížeči, které zařídí shromáždění údajů. Většina uživatelů lišty totiž pochází z USA a tím jsou lokalizované stránky znevýhodněné.

4.2.5 Analýza podle hypertextu

Všechny fulltextové vyhledávače samozřejmě analyzují i obsah stránek, aby bylo možné vůbec něco vyhledat. Nejde ale většinou o prosté procházení textu, protože by bylo možné vnutit robotovi text v podobě například meta značek. U je důležité, kde se ve stránce text nachází a jakou má pozici uvnitř textu. Kupříkladu hledaný výraz v nadpisu bude mít určitě větší váhu než výraz v obyčejném odstavci. Přístup Googlu je založen také na tom, že analyzuje obsah sousedních webových stránek a zajišťuje zobrazení výsledků, které nejlépe odpovídají dotazu uživatele.

5 Možnosti vlastního vyhledávání

V předchozích kapitolách jsme si popsali co je vyhledávač a jakým způsobem funguje. Nyní se podíváme na to, jaké cesty vedou k vytvoření vlastního vyhledávání.

V dalších částech této práce bude popisován vyhledávač, který se úzce specializuje na stránky českých firem a pro který jsme si museli vybrat jednu z cest vlastního fulltextového vyhledávání. Níže jsou uvedeny možnosti tvorby vlastního vyhledávání.

5.1 Google Soap Search API

Google Soap Search API nám umožní si na vlastní stránky vložit vyhledávání od společnosti Google. Nesmírná výhoda tohoto postupu je, že se nemusíme starat o vlastní index, který tudíž nemusíme aktualizovat.

Na druhou stranu dostáváme ve výsledcích vyhledávání pouze takové informace, které Google dopředu naindexuje a nemáme možnost si výsledky nebo algoritmy relevance upravit. Další velkou nevýhodou je, že je možné toto vyhledávání zdarma použít pouze pro 1000 dotazů denně a nesmí se podle licence používat komerčně.

5.2 Parazitování na některém vyhledávači

Další možností, kterou zde uvádím pouze jako čistě teoretickou možnost, na které se určitě nedá postavit kvalitní služba, je parazitování na některém vyhledávači.

Celý princip je založen na tom, že návštěvník přijde na naše stránky a my bychom mu nabídli přehledné vyhledávání. Po zadání dotazu bychom odeslali požadavek na některý vyhledávací stroj například v tomto tvaru „`http://www.google.cz/search?q=Univerzita+Pardubice`“, ten řekne vyhledávači, že má vypsat výsledky pro výraz Univerzita Pardubice. Ten vrátí HTML stránku s výsledku, kterou není problém rozparsovat a výsledky si vrátit do vlastního vyhledávání. Následně je výsledek vyhledávání prezentován návštěvníkovi, který ani nemusí tušit, že právě vyhledával přes Google vyhledávání.

5.3 Vytvoření vlastního vyhledávače na zelené louce

Tato varianta by se mohla zdát jako nejlepší řešení. Rozebereme si podrobněji, co všechno je potřeba udělat, než by nám náš vlastní naprosto jednoduchý vyhledávací program začal pracovat.

Nejprve by bylo potřeba vymyslet aplikaci, která nám načte a zanalyzuje jednotlivé webové stránky (crawler). Pak způsob, jakým budou data ukládána do indexu a jak soubor nebo soubory s indexovanými daty budou vypadat.

Další věcí, kterou by bylo potřeba vymyslet je, jak se bude v souborech vyhledávat a jak vracet uživateli přehledné výsledky v co nejkratším čase.

Popsání uvedeného principu je jednoduché, ale skrývá se za tím práce na několik tisíc hodin. V tomto případě bych se držel hesla proč vymýšlet již vymyšlené. Tím chci říct, že se na internetu dá najít mnoho open source produktů, které můžeme využít a ušetřit si tím mnoho hodin práce. Tomuto tématu se bude věnovat další podkapitola.

5.4 Využít dostupných open source produktů

Toto je cesta, která je pro náš vyhledávač nad internetovými stránkami firem nejpřijatelnější a vysvětlím proč.

Je mnoho způsobů, jak si můžeme vlastní vyhledávač představit co chceme vyhledávat a jak vlastně chceme výsledky řadit. Ani jedna možnost z předchozích uvedených způsobů tvorby vyhledávače, krom vytvoření všeho od nuly nám nedává volné ruce.

Když si uvědomíme, že není dobrým nápadem začít na zelené louce a všechno si od začátku přizpůsobovat a ladit sami, pak nám zbude poslední možnost. A to využít již hotová řešení a poskládat z nich funkční vyhledávač, který bude vyhledávat přesně to co po něm budeme chtít.

Tuto možnost jsem zvolil pro vytvoření našeho vyhledávače, přičemž jsem použil produkt s názvem Regain, který je celý postaven nad Lucene což je open source knihovna pro tvorbu indexu a vyhledávání v něm. Podrobněji se k těmto dvěma produktům vrátíme později.

5.5 Zhodnocení jednotlivých přístupů

Nyní jsme si popsali několik přístupů k tvorbě vlastního vyhledávače. Následující tabulka nám zohledňuje některá kritéria, která jsou důležitá pro vytvoření našeho vyhledávače českých webových stránek firem.

Zajímalo mě, jaká je cena řešení, jestli mám přístup ke zdrojovým kódům, jestli máme kontrolu nad tím co se do indexu ukládá a co chceme v indexu hledat, dále časová náročnost řešení a licence.

Z níže uvedené tabulky vyplývá, že bychom mohli uvažovat o dvou přístupech, buď vlastní řešení na zelené louce, nebo využít open source technologií. Nakonec díky časové náročnosti vytvoření vlastního řešení na zelené louce byla zvolena druhá možnost a to využití open source technologíi.

	Cena	**Přístup ke zdrojovým kódům**	**Kontrola nad indexem**	**Časová náročnost implementace**	**Licence**
Google Soap Search API	podle užití	ne	ne	nízká	Google
Parazitování na některém vyhledávači	zdarma	ne	ne	nízká	sporná
Vytvoření vlastního vyhledávače na zelené louce	zdarma	ano	ano	velmi vysoká	vlastní
Využít dostupných open source produktů	zdarma	ano	ano	vysoká	open source

Tabulka 1: Zhodnocení jednotlivých technologii.

6 Použité technologie

Tato kapitola přiblíží, jaké technologie jsme si pro vytvoření vyhledávače vybrali a proč. Bude zde vysvětleno, jaký programovací jazyk byl zvolen, proč jsme si vybrali nadstavbu Regain pro knihovnu Lucene.

6.1 Programovací jazyk

První nad čím bylo nutné se zamyslet byl výběr programovacího jazyka. Naše požadavky byly takové, že jsme chtěli objektově orientovaný jazyk, ve kterém bude možné vytvořit webovou aplikaci nejlépe nezávislou na platformě.

Od začátku jsme přemýšleli o třech adeptech, kterými byly PHP, C# a JAVA. V následující tabulce jsem udělal porovnání těchto jazyků.

	Pouze objektový jazyk	Knihovny pro fulltextové vyhledávání	Multiplatformnost
PHP	Ne	Ano	Ano
C#	Ano	Ano	Ne
JAVA	Ano	Ano	Ano

Tabulka 2: Porovnání programovacích jazyků

Z tabulky je zřejmé, že ze zápolení nakonec vyšla vítězně JAVA. Důvodů bylo hned několik, vyhovovala všem dopředu daným požadavkům, je to objektově orientovaný jazyk, nebyl problém pro ni sehnat příslušné knihovny na usnadnění a urychlení práce a posledním důvodem je to, že je multiplatformní.

Obrázek 13: Logo Java
Zdroj: http://blog.taragana.com/index.php/category/linux/kde-desktop/

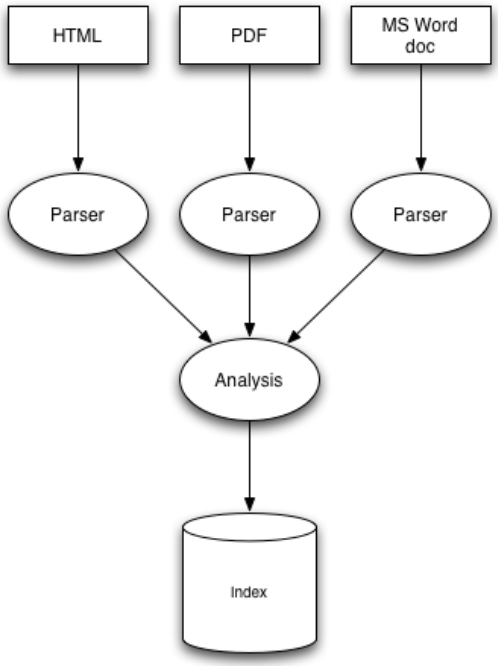
6.2 Apache Lucene

V následujícím textu budu převážně vycházet z anglické knihy „Lucene in Action“. [11]

Když bylo rozhodnuto o programovacím jazyku, přišlo na řadu zvolit variantu vyhledávání. Jak bylo popsáno v předchozí kapitole, vybrali jsme cestu přes open source knihovny. Nejlepší knihovna pro fulltextové vyhledávání je určitě Lucene. Jeho výhoda je v tom, že je to knihovna napsaná v Javě a umí velice dobře a rychle vyhledávat. Lucene za nás udělá několik základních věcí, které si popíšeme v dalších podkapitolách.

6.2.1 Vytvoření indexu

Při vytvoření a vkládání dat do souboru indexu se dějí tři základní věci. Nejprve se vstupní text převede na prostý text, poté se nechá projít analyzátorem a následně je přidán do indexu. Tento postup nám vysvětluje následující obrázek číslo 14.



Obrázek 14: Vytvoření indexu
Zdroj: [11]

Na obrázku je zřetelné, že nás vcelku nezajímá co vstupuje do analyzátoru jako vstupní data. Funguje to tak, že parsery všechna vstupní data, ať HTML stránku, PDF dokument nebo MS Word dokument převedou na prostý text, se kterým analyzátor pracuje vždy stejně.

Analyzátor je třída, která se postará o to, aby se do indexu uložilo vše potřebné. Například při ukládání této diplomové práce do indexu určitě nebudeme chtít ukládat předložky, spojky nebo slova, která nejsou pro podstatu textu důležitá. Těmto slovům se říká v dokumentaci Lucence „stop words“. Analyzátor nám zajistí, že se do indexu dostane pouze to, co je pro nás zajímavé, jinými slovy „stop

words“ se neukládají. Důvodem, proč programátoři knihovny zvolili možnost vyměnit třídu analyzátoru, je to, že každý jazyk má jiná slova, která nechceme indexovat a stačí vyměnit třídu analyzátoru a vše správně funguje. Další věcí je, že i když pro Lucene není implicitně vytvořena třída „CzechAnalyzer“, nebylo velkým problémem ji napsat. Následující text ukazuje, jak fungují jednotlivé typy anglických analyzátorů.

```
Analyzing "The quick brown fox jumped over the lazy dogs"
SimpleAnalyzer:
[the] [quick] [brown] [fox] [jumped] [over] [the] [lazy] [dogs]
StopAnalyzer:
[quick] [brown] [fox] [jumped] [over] [lazy] [dogs]
Analyzing "XY&Z Corporation - xyz@example.com"
WhitespaceAnalyzer:
[XY&Z] [Corporation] [-] [xyz@example.com]
```

6.2.2 Přidávání do indexu

Už víme, jak se získají a analyzují data, která do indexu chceme přidávat. Nyní se podíváme, jak přidávání do indexu funguje a jaké máme volby. Nejpřehlednější asi bude popsat si metodu addDocument(Directory dir), která se o uložení nové stránky do indexu postará.

```
public void addDocument (Directory dir,String url, String content,
String title)throws IOException {
      IndexWriter writer =
      new IndexWriter(dir,new CzechAnalyzer,true);
      Document doc = new Document();
      doc.add(Field.Keyword("URL", url));
      doc.add(Field.UnIndexed("country", title));
      doc.add(Field.Text ("content", content));
      writer.addDocument(doc);
      writer.optimize();
      writer.close();
}
```

Z tohoto zdrojového kódu je zřetelné, jak Lucene přidává další dokumenty do indexu. Metodě je předána instance třídy `org.apache.lucene.Directory`, která zapouzdřuje informace o fyzickém uložení indexu. Dále v parametrech naj-

deme řetězce content, url a title, ve kterých je pro příslušnou stránku uložena URL adresa, obsah stránky a titulek.

V metodě si vytvoříme instanci třídy `org.apache.lucene.IndexWriter`, kterému předáme kromě informace o umístění indexu také instanci `CzechAnalyzer`. Tím je vytvořen nástroj pro ukládání do indexu.

Dalším krokem je vytvoření instance třídy `org.apache.lucene.document.Document`, který do kterého si nejprve uložíme jednotlivá ukládaná pole a poté ho celý vložíme do indexu. Přidávání polí do dokumentu se dělá metodou `add()`. Ta nám zajistí, aby každá zaindexovaná stránka zapouzdřovala více polí podobně jako je to naznačeno na obrázku 8. Jako parametr metody se vkládá `Field.ZpusobNaindexovani(String klic, String hodnota)`, kde první textový parametr (String) je klíč, přes který se později k prvku dostane a druhý je jeho hodnota. `ZpůsobNaindexování` jako název uvedené statické metody je zástupný výraz, který jsem zde použil, ale v praxi je možné použít tyto čtyři volby:

• `Field.Keyword(String, String)` – Tuto volbu využijeme, když chceme uložit do indexu něco v čem budeme chtít vyhledávat, ale nebudeme chtít, aby nám text před uložením prošel analyzátorem. Používá se například k ukládání www adres, telefonních čísel nebo dat.

• `Field.UnIndexed(String, String)` – Toto je volba vhodná pro data, která budeme chtít zobrazovat s výsledkem vyhledávání, ale nebudeme v nich chtít nikdy hledat. Toto pole je do indexu uloženo tak, jak ho předáme. Nedoporučuje se vkládat do tohoto pole velké objemy dat, protože pak index může dosáhnout obrovské velikosti.

• `Field.UnStored(String, String)` – Toto je v podstatě přesný opak předchozího UnIndexed. Data jsou ukládána pro vyhledávání, ale nedostaneme se

už nikdy k předchozímu textu. Toto se využívá například pro ukládání textů webových stránek.

• `Field.Text(String, String)` – Volbu text použijeme pro případ, že chceme v textu vyhledávat a mít možnost se vrátit k předchozímu textu například pro výpis detailu vyhledávání. Data jsou přes analyzátor ukládána do indexu.

6.2.3 Práce se soubory indexu a vyhledávání

Nyní víme jak vytvořit soubor s naindexovanými informacemi a v této podkapitole se podíváme, jak v těchto informacích rychle a účinně vyhledávat.

Pro lepší pochopení principu na kterém celé vyhledávání v Lucene funguje si rozebereme jednu metodu, která nám zajistí vrácení výsledků na základě zadaného dotazu.

```
public void testTerm() throws Exception {
     IndexSearcher searcher = new IndexSearcher(directory);
     Term t = new Term("content", "java");
     Query query = new TermQuery(t);
     Hits hits = searcher.search(query);
     searcher.close();
}
```

Na začátku vidíme, že je vytvořena instance třídy `IndexSearcher`, které se předá informace o uložení souborů s indexem a která se nám postará o přístup k indexu pro vyhledávání.

Další důležitá třída je `Term`, která při vytváření instance bere jako první parametr pole, ve kterém se má vyhledávat. Pro připomenutí, jedná se o pole které se do indexu ukládalo pomocí „`doc.add(Field.Text ("content", "nějaký text třeba java"))`“ . Díky konstruktoru `new Term("content", "java")` vyhledáváme v poli content heslo java.

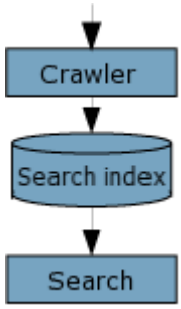
Instance třídy `Query` se předá metodě search objektu IndexSearcher a vrátí nám seznam výsledků.

Lucene seznam výsledků řeší obalovým objektem `Hits`, ze kterého jsme schopni získat seznam výsledků vyhledávání, který nám bude stačit pro výpis do stránky.

Nakonec musíme `IndexSearcher` uzavřít, aby bylo možné opět přistupovat k souboru s indexem.

6.3 Regain

Tato kapitola se bude zabývat aplikací open source aplikací Regain, kterou jsem v mé diplomové práci využil. Tato aplikace poskytuje konkrétní implementaci knihovny Lucence přímo určenou k webovému vyhledávání. Tato aplikace už v základu nabízí crawler pro procházení a ukládání webových stránek a uživatelské rozhraní pro zobrazení výsledků, jak je znázorněno na následujícím obrázku.



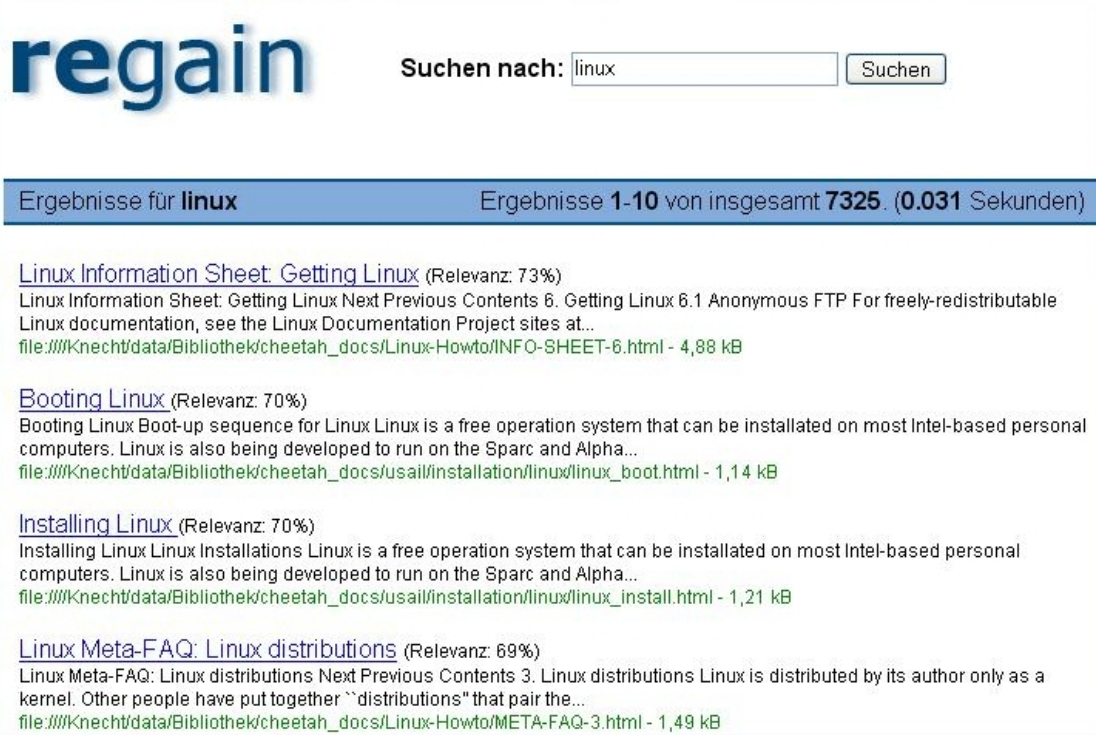
Obrázek 15: Regain Diagram

Zdroj: http://regain.sourceforge.net/details.php

Regain nám poskytuje několik základních vlastností, díky kterým jsme se rozhodli ho využít jako základ našeho vyhledávače. Jednou z nejdůležitějších je systém konfiguračních XML souborů, díky kterým je možné si celé chování vyhledávače, jak na straně indexování, tak na straně vyhledávání upravit přesně podle

konkrétního využití. Jako příklady nastavení mohu uvést například zdroje pro indexování, typy povolených souborů k indexování (HTML, PDF, doc, atd.) nebo způsob zobrazení vyhledaných výsledků.

Dalšími příjemnými vlastnostmi jsou propracovaný způsob procházení a ukládání webových stránek v závislosti na nastaní v konfiguračním souboru nebo uživatelské rozhraní pro vyhledávání, které splňuje standardy dnešních vyhledávačů.



Obrázek 16: Regain vyhledávání
Zdroj: http://regain.sourceforge.net/details.php

Na předchozím obrázku je zřejmé, že vyhledávání Regain funguje podobně jako Google vyhledávání. Hlavním zásadním rozdílem je to, že relevanci vyhledávaných výsledků nehodnotí pomocí stejného algoritmu jako Google, ale záleží na relativní četnosti vyhledávaného výrazu. To znamená, že když se slovo v dokumentu se 100 slovy objeví 5 krát, je relevance vyhodnocena jako vyšší než když v

dokumentu s 1000 slovy najde výraz 10 krát. Další rozdíl můžeme najít ve způsobu použití. Regain se používá k naindexování omezeného množství internetových stránek (většinou jednoho webu) oproti Googlu, který má podstatně širší záběr.

Závěrem této kapitoly nutno říct, že požadavky na naši aplikaci byly tak specifické, že nebylo možné pouze upravit konfigurační soubory a začít vyhledávač používat. Bylo nutné přepsat a upravit velkou část aplikace, do konfiguračních souborů přidat vlastní atributy a aplikaci dodat funkcionalitu, která byla požadována. Některé z realizovaných úprav budou popsány v další kapitole.

7 Návrh a implementace aplikace Analýza trhu

Tato kapitola se bude zabývat postupem při vytváření aplikace pro vyhledávání na webových stránkách firem v České republice.

7.1 Zadání

Na začátku celého projektu byl dotaz: „Jaké firmy v České republice pracují s nějakou technologií nebo se zabývají konkrétní činností?“. Firma Seico s. r. o. se zabývá školením Java technologií a má chce v rámci cílené reklamy oslovit firmy, které by mohly mít zájem o takové školení.

Prvotní předpoklad byl takový, že firma, která na svých stránkách bude mít zastoupený výraz Java s největší pravděpodobností bude s Javou pracovat a zaměstnávat Java programátory. Pak už by nebyl problém tyto firmy oslovit a nabídnout jim školení.

Toto je pouze jeden z mnoha modelových příkladů, ke kterým by se mohl tento způsob vyhledávání hodit. Proto jsme se s firmou Seico s. r. o. rozhodli, že se bude investovat čas a prostředky do vývoje této aplikace.

Požadavky na aplikaci:

- Možnost naindexovat zadaný seznam webových stránek v co nejkratším čase,
- při procházení stránek nenásledovat odkazy „ven“ (na cizí webové stránky), respektovat robots.txt,
- počítat s rozdílným kódováním indexovaných stránek,

- přístup k databázi pro ukládání údajů o úspěšném naindexování nebo nenaindexování firemního webu,
- možnost vytvoření částí indexu na více počítačích najednou a následné spojení,
- velká rychlost odezvy vyhledávání i při vysokém množství naindexovaných stránek,
- co možná nejvyšší rychlost naindexování všech dostupných stránek českých firem,
- s výsledky vypisovat i údaje o umístění společnosti, počtu zaměstnanců, typu společnosti a například počtu let na trhu.
- možnost filtrovat výsledky vyhledávání podle dalších kritérií, kraje, typu společnosti atd. ,
- od každého serveru (každé firemní stránky) zobrazovat pouze jeden výsledek ve výsledcích vyhledávání,
- vymyslet způsob, jak doplnit stávající databázi firem, kterou má firma Seico s.r.o. k dispozici o další adresy firemních webů.

7.2 Sběr informací

Dříve než jsem pozornost zacílil na vlastní tvorbu vyhledávače bylo potřeba se zamyslet nad tím, kde a jak se vezmou informace o firmách a jejich webových stránkách.

Již rok před nápadem na vytvoření této aplikace se začalo se sběrem dat o firmách a to z několika zdrojů. Prvním zdrojem, který se procházel a stahovala se z něj data byl Administrativní registr ekonomických subjektů (ARES), dalším byly veřejné databáze firem a poslední, díky kterému jsme získali velké množství webových adres bylo průběžné stahování nabídek z úřadu práce do naší databáze.

Z těchto zdrojů jsme nakonec měli informace o většině firem a fyzických osob, které působí v České republice. Záznamů v databázi bylo kolem 4 miliónů, z toho zhruba 250 tisíc aktivních společností s ručením omezeným a akciových společností, které nás pro naše vyhledávání zajímají nejvíc.

Při následné analýze dat jsme zjistili, že státní správa nemá zdaleka informace korespondující s realitou. V první chvíli jsme měli pouze 77 000 spárovaných informací to znamená www adresu spárovanou s identifikačním číslem (IČ) firmy a dalšími informacemi, jako například adresou, krajem a například počtem zaměstnanců. Bohužel se později zjistilo, že víc jak třetina těchto www adres neexistuje nebo neodpovídá realitě.

Proto přede mnou stál úkol získat zbylých zhruba 152 000 webových adres, které nemáme vůbec a navíc opravit adresy, které máme, ale nejsou dostupné. Pro tento účel byla vytvořena speciální GUI aplikace, která se starala o získání těchto adres. Této aplikaci se budu věnovat v závěru práce.

7.3 Kódování

Jeden z nejpalčivějších problémů dnešních webových stránek je kódování. Bohužel pro kódování nebyl nikdy vytvořen žádný standard a proto jsem se musel vypořádat s nutností zjistit kódování stránky, překódovat ji do UTF-8 a teprve poté ji uložit do indexu. Regain nám tento mechanizmus nabízí, ale bohužel část se zjištěním kódování nefunguje příliš dobře, proto bylo potřeba tuto část přeprogramovat.

Pro zjištění kódování webových stránek jsem využil knihovnu Mozilla Charset Detector, která je pod open source licencí a je vytvořená v Javě. Díky tomu nebyl problém knihovnu připojit ke stávající aplikaci a začít ji využívat.

Při testování aplikace jsem došel k závěru, že to byl krok správným směrem, protože před využitím této knihovny byla do indexu uložena s chybným kódováním pouze každá zhruba pátá stránka. Nyní je občas jen problém s překódováním některých znaků, ale stránky se ukládají správně.

7.4 Indexování stránek, které používají rámy

Dalším problémem bylo ukládání stránek, které byly vytvořeny pomocí rámů. V dnešní době to je méně používaná zastaralá technika tvorby webu, ale bohužel je stále velké procento webových stránek, které ji využívají. Crawler v základu dělal to, že přišel na úvodní stránku, kde nenašel žádné další odkazy, protože zde byla pouze definice rámů a stránku opustil.

Pro jasnější pochopení přikládám následující část kódu, která ukazuje modelový případ úvodní stránky s rámy. Je zde stránka rozdělena na 30% pro menu a 70% pro vlastní obsah. Vidíme zde, že stránka neobsahuje žádné odkazy, jak je

známe, ale pouze odkazy na stránky menu.html a hlavni.html. Tento příklad znázorňuje následující část HTML kódu.

```
<HTML>
<HEAD>
    <TITLE>Úvodní stránka</TITLE>
</HEAD>
<BODY>
    <FRAMESET COLS="30%, 70%">
        <FRAME SRC="menu.html">
        <FRAME SRC="hlavni.html">
    </FRAMESET>
</BODY>
</HTML>
```

Bylo tedy nutné v aplikaci rozpoznat, že se jedná o webovou stránku s rámy zjistit odkazy na HTML dokumenty, ze kterých se stránka skládá a projít je. Pro rozparsování stránky s rámy jsem použil třídu `ParserCallBack`, která umí odchytit jednotlivé entity HTML dokumentu. Díky tomu už nebyl problém dostat se k navazjícím dokumentům, které jsem potřebovat pro další projití stránky.

7.5 Konfigurační soubor pro indexaci

Regain jako takový už v základu požívá konfigurační xml soubor pro indexování. Tento soubor je vlastně jediným zdrojem informací pro konfiguraci, nastavení procházení stránek a jejich ukládání do indexu. My se budeme věnovat nastavením, která jsou pro naši aplikaci důležitá. Některá z nich byla k dispozici už se základem Regainu, ale mnohé bylo nutné doplnit a doprogramovat jejich funkcionalitu do tříd aplikace. Díky tomu jsme nakonec docílili toho, že je možné většinu změn indexace udělat úpravou jednoho XML souboru bez nutnosti zasahovat do zdrojových kódů.

7.5.1 Nastavení indexovaných webů

Jednou z nejdůležitějších informací, kterou můžeme najít v konfiguračním souboru je tag `startlist`. Tento tag pod sebe agreguje seznam webových stránek, které chceme indexovat s dodatečnými informacemi. Na následujícím řádku je příklad tagu `start`, který slouží k uchování informací o jednom firemním webu.

```
<startlist>
<start parse='true' index='true' name='SEICO s.r.o.'
ico='25268171' region='6' companySize='1' companyAge='12'
companyType='1' companyForm='112' > http://www.seico.cz
</start>
</startlist>
```

První dva atributy tagu `start`, které jsou dostupné už v základu Regainu nastavují, jestli se má stránka parsovat a jestli se má ukládat do indexu. Další jsem si dodělal, protože jsem potřeboval ke každé stránce také do indexu ukládat další informace, kterými jsou název firmy, identifikační číslo firmy, místo působení atd. Většinou to bylo řešeno navázáním na identifikační číslo určitého záznamu, například uložení čísla 6 do atributu region znamená, že firma Seico, s. r. o., působí v Královéhradeckem kraji. Tím jsme minimalizovali množství dat ukládaných do indexového souboru.

Uvnitř tagu `start` vidíme příslušnou webovou adresu firmy která slouží jako vstupní bod pro indexování webové stránky.

7.5.2 Nastavení průchodu jedné stránky

Při testování aplikace se ukázalo, že pro rychlejší indexování všech webů je potřeba omezit počet procházených stránek jednoho serveru. Hlavním důvodem pro toto omezení byly různé rozsáhlé eshopy, jejichž indexace by trvala nepřija-

telnou dobu. Tuto informaci jsem proto přidal do indexu pod tag s názvem `maxloopofserver` a nastavil ho na 300 stránek.

Dalším problémem, který jsme museli vyřešit, byla hloubka procházení stránek. Regain tuto problematiku vůbec neřeší. Bylo tedy nutné upravit zdrojové kódy a zjišťovat v jaké úrovni se pro daný server nacházíme. Tuto informaci jsme do konfiguračního xml souboru vložili pomocí tagu `levelsOfCrawling`. Pro naši aplikaci jsme použili dvojúrovňové procházení, což znamená, že jsme naindexovali pouze všechny stránky, na které vedou odkazy z úvodní stránky webu. Hloubku procházení stránek podrobněji řeší kapitola Procházení a indexace.

7.5.3 Whitelist a blacklist

Konfigurace nám také umožňuje vytvoření seznamu stránek, které chceme procházet a indexovat (whitelist) a seznamu stránek, které nechceme procházet a indexovat (blacklist). Samozřejmě, že nebudeme do těchto seznamů zapisovat přesnou url, kterou chceme vynechat nebo naopak uložit. Možné jsou proto dva postupy. Buď zadáme do tagu `prefix` pouze začátek adresy, kterou (ne)chceme procházet nebo tyto seznamy také akceptují regulární výraz, který se následně na adresu aplikuje. Následující kód je ukázka whitelistu, který přijme všechny začátky webových adres a zároveň adresy s vyjmenovanými koncovkami určené regulárním výrazem. Ostatní adresy vynechává.

```
<whitelist>
     <prefix>http://</prefix>
     <prefix>http://www</prefix>
     <prefix>https://www</prefix>
     <regex>([^"]*\.(html|htm|php|asp))</regex>
</whitelist>
```

Naopak blacklist funguje tak, že vše co se vyhodnotí jako shodné se automaticky neprochází i v případě, že před tím tato adresa prošla v pořádku whitelistem. Rozdíl v definici blacklistu a whitelistu je pouze v tagu uvozujícím seznam kriterií. Blacklist začíná tagem `blacklist`, přičemž whitelist tagem `whitelist`.

7.5.4 Informace o připojení do databáze

Další položky, které bylo nutné přidat do konfiguračního souboru, byly informace nutné pro připojení do databáze, která slouží pro ukládání informací o úspěšném, neúspěšném nebo přesměrovaném požadavku na stažení dané stránky firmy.

Bylo tedy nutné přidat do souboru informace o uživatelském jménu a heslu, URL umístění databázového serveru a název databáze. Toto jsem nakonec vyřešil tak, že jsme všechny tyto položky vložily do konfiguračního souboru pod tag `database`.

7.5.5 Další položky nastavení

V konfiguračním souboru ještě najdeme další velké množství nastavení a konfigurací. Můžeme například nastavit jak často se vytváří bod návratu pro indexování, což je bod, od kterého můžeme v indexování dat pokračovat, když se nám například vypne server nebo spadne připojení. Můžeme si také nastavit do jakého adresáře budeme ukládat výsledné soubory s indexovanými daty. V dokumentaci Regainu jsou všechny základní tagy pro konfiguraci popsány.

7.6 Rychlost indexování

Vůbec první návrh celého vyhledávače byl postaven pouze na knihovně Lucene. Bohužel při prvním pokusu indexovat data jsem přišel na to, že doba indexování zmiňovaných zhruba 70 tisíc firemních webů by byla zhruba jeden měsíc, ale v produkčním nasazení při indexování řádově stovek tisíc firemních webu by doba byla nepřijatelná. Proto bylo nutné tento problém vyřešit. Nejprve jsme zkoušeli použít vláknovou podporu Lucene, ale nakonec jsme přišli na to, že knihovna Regain má podstatně lepší podporu vláken, pomocí které se podstatně zkrátí doba indexování.

Při testování a odlaďování knihovny Regain jsme se nakonec díky vláknům, možnosti nastavení hloubky indexování a počtu indexovaných stránek dostali na dobu indexování čtyři dny, která už byla podstatně přijatelnější.

Pro jeho ještě větší zrychlení jsem navrhl systém, kdy bude možné rozdělit konfigurační soubor na tři různé soubory, přičemž v každém budou informace nutné k naindexování části firemních stránek. Tyto soubory se spustí na několika nezávislých počítačích a následně po dokončení indexace se indexy spojí do jednoho. Tímto postupem by bylo možné naindexovat webové stránky na pěti počítačích za méně než jeden den. Toto je jedna z věcí se kterou se do budoucna počítá, ale zatím je vytvořena pouze v testovací verzi a v aplikaci běžící na internetu jsme ji zatím nepoužívali.

7.7 Informace o indexovaných datech

Dalším požadavkem na aplikaci, který jsme museli vyřešit, byla vlastnost ukládání odezvy z jednotlivých webových prezentací firem. Bylo tedy nutné při dotazu na tyto webové stránky zjistit stavové kódy http protokolu (Tabulka 3), což jsou kódy, které jsou odeslány klientovi na základě požadavku a uložit tyto zjištěné informace v přehledné podobě do databáze.

Kategorie	Rozsah stavových kódů	Popis
Informační	100 – 199	Zprávy definované konkrétní aplikací.
Úspěch	200 – 299	Požadavek byl úspěšně zpracován.
Přesměrování	300 – 399	Klient musí pro konečné zpracování požadavků vykonat určitou činnost. O této činnosti se uživatel nemusí dovědět.
Chyba klienta	400 – 499	Problémy na straně klienta.
Chyba serveru	500 – 599	Problémy na straně serveru.

Tabulka 3: Tabulka stavových kódů [13]

Získání návratového http kódu serveru jsem vyřešil pomocí knihovny `HttpClient`, přes kterou jsem se dotázal na úvodní stránku každé firmy a nechal si vrátit návratový kód a v případě přesměrování také stránku, na kterou byl crawler přesměrován.

Zjištěné informace byly následně uloženy do databáze, ve které jsme evidovali návratový kód, identifikační číslo firmy, webovou adresu a adresu, kam jsme byli v případě přesměrování přesměrováni. S těmito informacemi se dá ná-

sledně pracovat třeba tak, že můžeme po několikerém neúspěšném indexování daného webu téměř s jistotou říci, že daná webová adresa není funkční.

7.8 Úpravy aplikace na straně vyhledávání

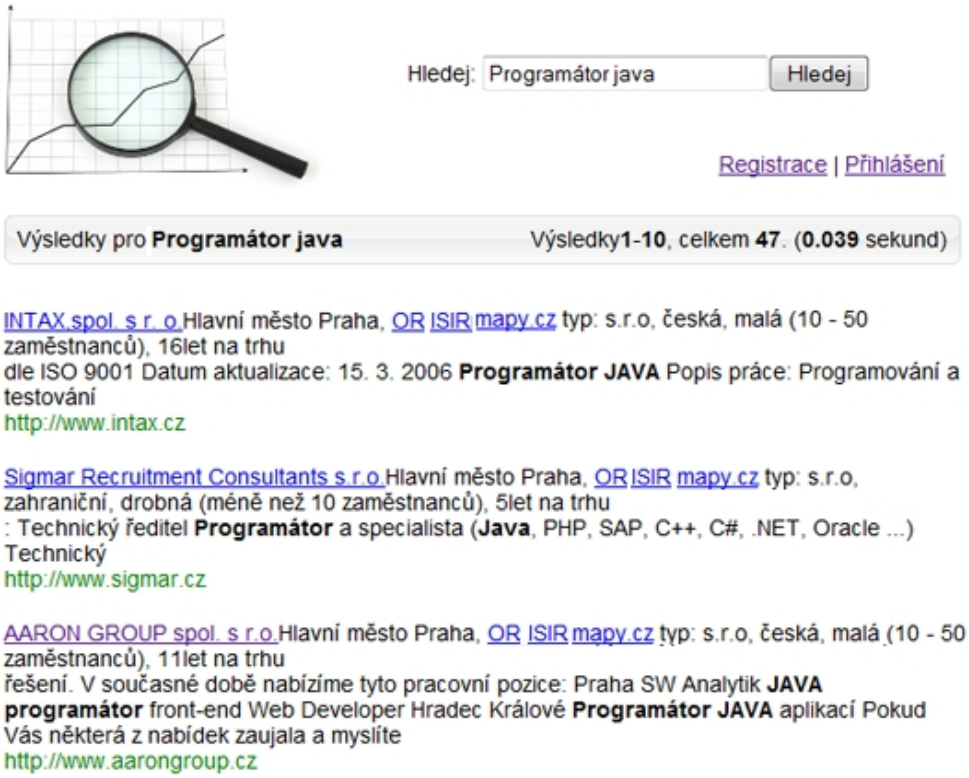
Regain jako takový má vytvořeno určité základní prostředí pro výpis výsledků vyhledávání. Mým úkolem bylo upravit výpis výsledků vyhledávání pro naše potřeby.

Regain na straně zobrazení výsledků šel cestou knihovny vlastních tagů díky kterým vypisuje informace do stránky. Jedná se o způsob, kdy si nadefinujeme tagy JSP stránek, na které si navážeme zdrojový kód. Pak stačí ve stránce tento tag zavolat a na pozadí se provede část zdrojového kódu, která přísluší k tomuto tagu.

Některé tagy jsem si přepsal, aby vyhovovaly mým potřebám a jiné jsem dopsal, protože s nimi v základu knihovna vůbec nepočítala.

7.9 Výpis výsledků

Jedním z dalších požadavků na naši aplikaci bylo vypsat do výsledků ke každé společnosti pouze jeden záznam. Bylo tedy nutné upravit výstup aplikace tak, aby bylo možné výsledky z každého serveru vyfiltrovat, vybrat ten nejrelevantnější a zbytek vůbec nevypisovat. To se nakonec podařilo a výsledek můžete vidět na následujícím obrázku, kde jsou vidět tři výsledky vždy od jiné společnosti srovnané podle relevance.



Obrázek 17: Výpis výsledků
Zdroj: http://www.analyza-trhu.cz

7.9.1 Výpis detailu

Každý detail, který je ve stránce vypsán má několik základních částí. Jako hlavní dominuje celý název firmy, kde bylo slovo nebo výraz nalezen. Potom zde můžeme vidět dodatečné informace o firmě, které jsme si přiindexovali, celou adresu úvodní stránky, krátký text, kde byl výraz nalezen a odkazy do Obchodního rejstříku, Insolvenčního rejstříku a na Mapy.cz. Po kliknutí na tyto odkazy se nám zobrazí příslušná firma na těchto serverech. Na následujícím obrázku vidíme výpis detailu v případě vyhledání výrazu `Seico s.r.o.`

Obrázek 18: Výpis detailu
Zdroj: http://www.analyza-trhu.cz

7.10 Vyhledávání v aplikaci Analýza trhu

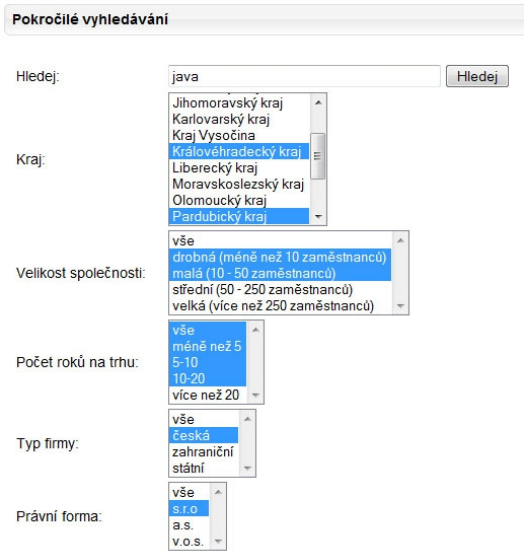
Neměli bychom zapomenout část aplikace, která se věnuje vyhledávání. V aplikaci analýza trhu jsou k dispozici dva typy vyhledávání. Prvním je vyhledávání klasické a druhým pokročilé. Pokročilé vyhledávání je rozšířením jednoduchého v tom, že u pokročilého můžeme výsledky vyfiltrovat podle různých kriterii. Na následujícím obrázku je vidět jak vypadá klasické vyhledávání. Je zde pouze pole pro zadání výsledku a tlačítko hledej.

Obrázek 19: Jednoduché vyhledávání
Zdroj: http://www.analyza-trhu.cz

Na obrázku 20 je pokročilé vyhledávání. Zadáváme v něm hledaný výraz úplně stejně jako v předchozím případě jen s tím rozdílem, že si můžeme blíže vybrat jaké typy výsledků vypisovat. Díky formulářovým prvkům umožňujícím vybrání více možností najednou můžeme určit přesné parametry firem, které mají zůstat ve výsledcích pro vyhledávání. V příkladě z obrázku 20 chceme vypsat pouze firmy, na jejichž stránkách se vyskytuje výraz `java` a které vyhovují těmto kritériím:

- jsou z Královéhradeckého a Pardubického kraje,
- mají do 50 zaměstnanců,
- jsou maximálně 20 let na trhu,

- jsou to české společnosti s ručením omezeným.



Obrázek 20: Pokročilé vyhledávání
Zdroj: http://www.analyza-trhu.cz

Za zmínku také ještě stojí, že pro tyto formulářové prvky jsem využil možnosti napsat si vlastní tag, který mi tuto funkcionalitu zajistí. Následující kód nám ukazuje jak je možné tento tag v praxi ve stránce využít. Vidíme zde tři atributy, atribut `multiselect` říká, jestli má uživatel mít možnost vybírat více položek najednou, atribut `field` je název pole, přes které se k prvku ve zdrojovém kódu dostaneme a atribut `option` je seznam prvků, ze kterých chceme vybírat.

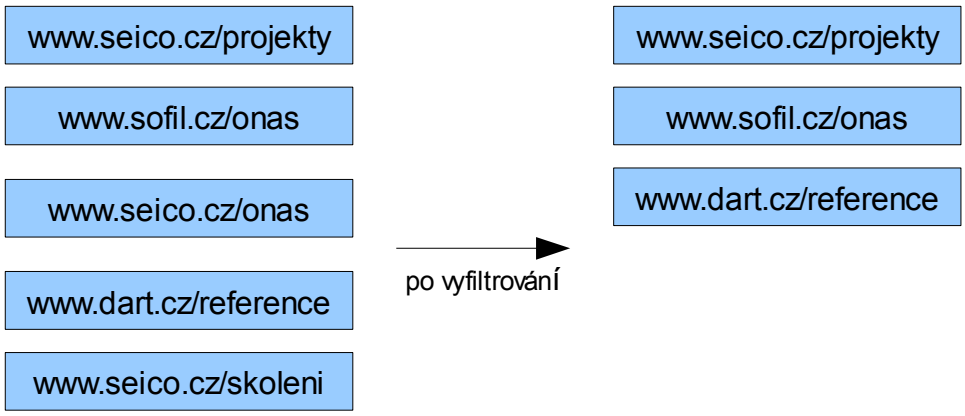
```
<search:input_select multiselect="true" field="companyType"
option="vše,česká,zahraniční,státní"/>
```

Oba typy vyhledávání se samozřejmě řídí stejnými pravidly pro vyhledávání. Můžeme zde používat operátory, které jsem uváděl v kapitole 4.2.1 Vyhledávací operátory a mnohé další, které je možné nalézt v dokumentaci Lucene.

7.11 Řešení relevance

Náš vyhledávač řeší relevanci (řazení výsledků) na základě relativní četnosti vyhledávaného slova ve stránce. Relativní četnost výskytů znamená, že stránka o 1 000 výrazech, kde bude vyhledávané slovo obsaženo 50krát bude mít nižší relevanci než stránka o 100 výrazech, kde bude hledaný výraz obsažen 7krát. Toto je princip, který je převzatý už z knihovny Lucene.

Protože jsou výsledky filtrovány, bylo nutné vymyslet, jak přizpůsobit řazení výsledků. Nakonec jsem to udělal tak, že se vezme v potaz pouze nejrelevantnější výsledek daného serveru a ostatní výsledky se ve výsledcích nezobrazí. Tento princip nám ukáže následující obrázek. Zde je zřetelné, jak si poradíme s výpisem stránek ze třech serverů. Vždy necháme vypsán pouze první výpis detailu podle pravidel popsaných v minulé kapitole a zbytek odstraníme.



Obrázek 21: Filtrování výsledků

7.12 Aplikace pro zjištění webových stránek firem

Nyní bude popsána aplikace, která byla navržena a implementována z důvodu nutnosti nalezení webových stránek firem, které nám v databázi chyběly.

Protože na celé internetové síti neexistuje jediný katalog, kde by se daly najít všechny internetové stránky firem (katalogy sice existují, ale neposkytují validní informace), muselo se jít jinou cestou. Jako nejlepší se ukázalo využít výsledků vyhledávání od společnosti Google, protože poskytuje nejlepší vyhledávací stroj. Princip je takový, že se vyhledávači zadá název firmy včetně „a .s." v případě akciové společnosti nebo „s. r. o." v případě společnosti s ručením omezeným a předpoklad je takový, že by se v prvních deseti výsledcích měly objevit stránky firmy v případě, že existují.

Tato metoda se ukázala jako nejlepší přístup, bohužel bylo nutné vymyslet, jak proces co možná nejvíce urychlit. Původní doba na jednu stránku, která by se přes vyhledávač ručně zadala, vyhledala a určila byla stanovena na 30 sekund, které by byly nutné pro vyhledání stránky a zkopírování odkazu do naší databáze. Když vezmeme v potaz, že bylo nutné projít 152 tisíc stránek, které jsme neměli k dispozici vůbec a asi 30 tisíc stránek, které jsme měli, ale které nebyly dostupné, dostáváme se k číslu 182 tisíc webových stránek firem, při kterém by nám vyhledávání chybějících webů trvalo 3 792 dní (182 000 · 30 sekund / 60 / 24). Proto bylo nutné vytvořit program, který celý proces co možná nejvíc zautomatizuje.

7.12.1 Požadavky na aplikaci

Jedná se o jednoúčelovou aplikaci, od které budeme chtít, aby uživateli nabídla co možná nejjednodušší a nejrychlejší způsob vyhledávání internetových stránek firem.

Aplikace by měla uživateli nabídnout tyto základní funkcionality:

- jednoduché a přehledné zobrazení výsledků vyhledávání,
- načíst vstupní soubor se seznamem IČ a názvů firem bez www adres a doplňovat do něj zjištěné webové adresy,

- jedním kliknutím se buď dostat na webovou stránku zobrazeného detailu nebo uložit webovou stránku k příslušné firmě do souboru,
- tlačítko na vyhledání společnosti na webu Firmy.cz k hledané společnosti,
- možnost přeskočení výsledků,
- přehledné statistiky,
- editace webové adresy před uložením do výstupního souboru.

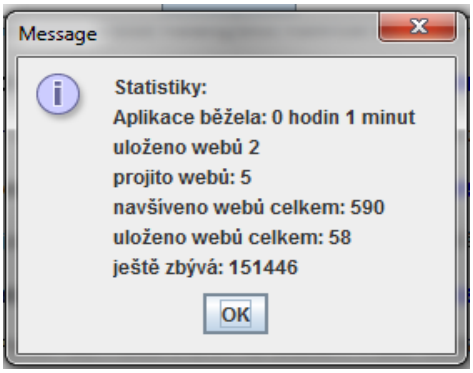
7.12.2 Výsledná aplikace

Výsledná GUI aplikace je uvedena na následujícím obrázku. V hlavičce je přehledně zobrazen název firmy, tlačítko pro vyhledání společnosti na webu Firmy.cz, IČ firmy a tlačítka „další“ a „zpět“. Je zde jednoduché intuitivní ovládání. V případě, že odkaz vyhovuje, stačí stisknout tlačítko OK pro uložení odkazu k firmě, v případě, že je potřeba adresu před uložením upravit, stiskneme šedivé „uprav“. Dále je možné kliknout na odkaz, díky kterému se otevře prohlížeč s příslušnou adresou.

V případě, který je na následujícím obrázku by stačilo kliknout na tlačítko OK a web společnosti Seico, s. r. o., který je na webové stránce `www.seico.cz`, by se uložil do souboru k příslušnému identifikačnímu číslu a názvu firmy.

Obrázek 22: Aplikace pro zjištění webových stránek firem.

Další důležitou věcí, která byla do aplikace implementována jsou přehledné statistiky, které uživateli při vypnutí nebo najetí na Edit → Statistiky říkají, jak dlouho s aplikací pracoval, kolik webových stránek našel a nebo u kolika firem se mu webovou stránku přiřadit nepodařilo. Okno se statistikami nám ukazuje další obrázek.



Obrázek 23: Statistiky

7.12.3 Řešené problémy

Prvním s problémů byla nutnost nezobrazovat výsledky z katalogů firem jako jsou Firmy.cz nebo HBI, protože to nejsou odkazy na hlavní stránku firmy. To se vyřešilo vytvořením blacklistu, do kterého bylo možné vkládat regulární výrazy, které se pak porovnávaly s webovými adresami ve vyhledaných výsledcích a když se shodovaly, tak nebyl výsledek zobrazen. Při naplnění blacklistu se ukázalo, že velké množství výsledků vyhledávání je možné přeskakovat, protože u daného dotazu všechny výsledky odpovídají některé položce v blacklistu.

To bohužel přineslo druhý problém, Google přišel na to, že dotazy na něj provádí nějaký automatický systém a zakázal nám přístup k výsledkům vyhledávání. Řešení byla minimálně dvě, buď jsem mohl vkládat dostatečný časový interval mezi jednotlivé dotazy na www.google.com, což by podstatně prodloužilo čas pro získání adres a prodloužit odezvu od uživatele, nebo jsem mohl předem stáhnout výsledky jako offline stránky na harddisk a následně nechat aplikaci pracovat s těmito stránkami uloženými na našem počítači.

Vydal jsem se druhou cestou a pronajal jsem si deset statických adres, mezi jednotlivé dotazy jsem vložil náhodný interval od 1 do 3 vteřin a začal stahovat výsledky vyhledávání na harddisk. Stahování trvalo 2 dny, ale po této době jsem měl na serveru asi 152 tisíc offline stránek (v první fázi se stahovaly pouze weby, které jsme neměli vůbec), se kterými jsem už mohl z aplikace jednoduše pracovat a nebát se zamezení přístupu ze strany Google.

7.13 Možnosti dalších rozšíření vyhledávače Analýza trhu

I přes důkladnou analýzu a propracování všech úprav aplikace je jistě velké množství věcí o které by bylo možné stávající vyhledávač vylepšit.

Asi jako první věc, kterou by bylo dobré v dohledné době upravit je výpis výsledků ve vyhledávači podle relevance. Relevance je v současné době, jak bylo již uvedeno počítána z relativního výskytu vyhledávaného výrazu na každé stránce firmy. Tím se bohužel stává, že se občas řadí nezajímavé a nedůležité webové stránky před podstatně významnější. Řešením by bylo například indexovat do větší hloubky nebo využít určitého průměrování relevance všech indexovaných stránek dané firmy.

Další věcí ke zlepšení je automatizace indexování. Cílem by mělo být, aby se firmy indexovaly jednou za čtrnáct dní samy. Aplikace už je pro tento postup částečně nastavená, ale ještě není možné automaticky spustit indexování a zaměnit nový index za starý bez nutnosti výpadku služby.

8 Závěr

Cílem teoretické části diplomové práce bylo přiblížit čtenáři, jaké je pozadí fulltextového vyhledávání a jeho možnosti pro využití na vyhledávání ve firemních internetových stránkách. Výsledkem této analýzy bylo, že se čtenář dozví, jak funguje indexace, procházení stránek na internetu a jakým způsobem se dá později ve vytvořeném indexačním souboru vyhledávat. Dále nám teoretická část ukazuje, jak jednotlivé vyhledávače řeší řazení výsledků ve výpisu vyhledaných informací, a poskytuje ucelený přehled možností pro vytvoření vlastního vyhledávače s konkrétním návrhem realizace.

V rámci praktické části byl navržen a vytvořen vyhledávač, který má za cíl změnit stávající situaci vyhledávání firemních stránek na českém internetu a poskytnout jeho uživatelům možnost fulltextově vyhledávat v předem indexovaných internetových stránkách českých firem. Výstupem této části je produkt s název Analýza trhu, který je dostupný na internetové adrese www.analyza-trhu.cz a možnost ho do určité míry bezplatně používat. Směr dalšího rozvoje bych viděl v přidání dalších funkcionalit a zdokonalení stávajících. Některé z věcí, nad kterými bude nutné se do budoucna zamyslet, jsou například princip řazení výsledků vyhledávaní nebo zautomatizování indexace a možnost nahrazení starého indexu novým bez nutnosti manuálního zásahu správce.

9 Bibliografie

[1] *Search Engine History.com* [online]. 2007 [cit. 2010-03-10]. Dostupné z WWW: <http://www.searchenginehistory.com/#before>.

[2] *Tim Berners-Lee - Wikipedia, the free encyclopedia* [online]. 2010-9-3 [cit. 2010-03-10]. Dostupné z WWW: <http://en.wikipedia.org/wiki/Tim_Berners-Lee>.

[3] *Historie Atlasu: Světlé začátky – eMag.cz* [online]. 2008-6-20 [cit. 2010-03-10]. Dostupné z WWW: <http://www.emag.cz/historie-atlasu-svetle-zacatky/>.

[4] *Atlas.cz – Wikipedie, otevřená encyklopedie* [online]. 2010-1-21 [cit. 2010-03-10]. Dostupné z WWW: <http://cs.wikipedia.org/wiki/Atlas.cz>.

[5] *Rozprodej českého Internetu pokračuje prodejem Centra – LUPA* [online]. 2007-12-5 [cit. 2010-03-10]. Dostupné z WWW: <http://www.lupa.cz/clanky/rozprodej-internetu-pokracuje-prodejem-centra/>.

[6] *Centrum.cz: odmítli kufr peněz a vydělali – iDNES.cz* [online]. 2004-10-14 [cit. 2010-03-10]. Dostupné z WWW: <http://ekonomika.idnes.cz/centrum-cz-odmitli-kufr-penez-a-vydelali-ff0-/eko_profily.asp?c=2004M240H04A>.

[7] *Hledání na webu* [online]. 21.3.2002 [cit. 2010-02-10]. Dostupné z WWW: <http://www.zine.cz/mirror/AZOld/overdrive/hledani.htm>.

[8] *Informace o korporaci - Přehled technologie* [online]. 2010 [cit. 2010-02-10]. Dostupné z WWW: <http://www.google.com/corporate/tech.html>.

[9] *Google PageRank* [online]. 12.3.2005 [cit. 2010-03-10]. Dostupné z WWW: <http://www.artic-studio.net/clanky/google-pagerank/>.

[10] *Ranky – PageRank, SRank, Jyxo Rank, Alexa* [online]. 2010 [cit. 2010-03-10]. Dostupné z WWW: <http://ranky.cz/>.

ne HATCHER, ERIK; GOSPODNETIC, OTIS. *Lucene in Action*. Greenwich : MANNING, 2004. 456 s. ISBN 1-932394-28-1.

[12] *Regain* [online]. 2009-08-03 [cit. 2010-04-16]. Regain manual [regain manual]. Dostupné z WWW: <http://regain.murfman.de/wiki/doku.php>.

[13] *Interval.cz* [online]. 29. 05. 2002 [cit. 2010-04-16]. Stavové kódy a hlášení v o povědi protokolu HTTP | Interval.cz. Dostupné z WWW: <http://interval.cz/clanky/stavove-kody-a-hlaseni-v-odpovedi-protokolu-http>.

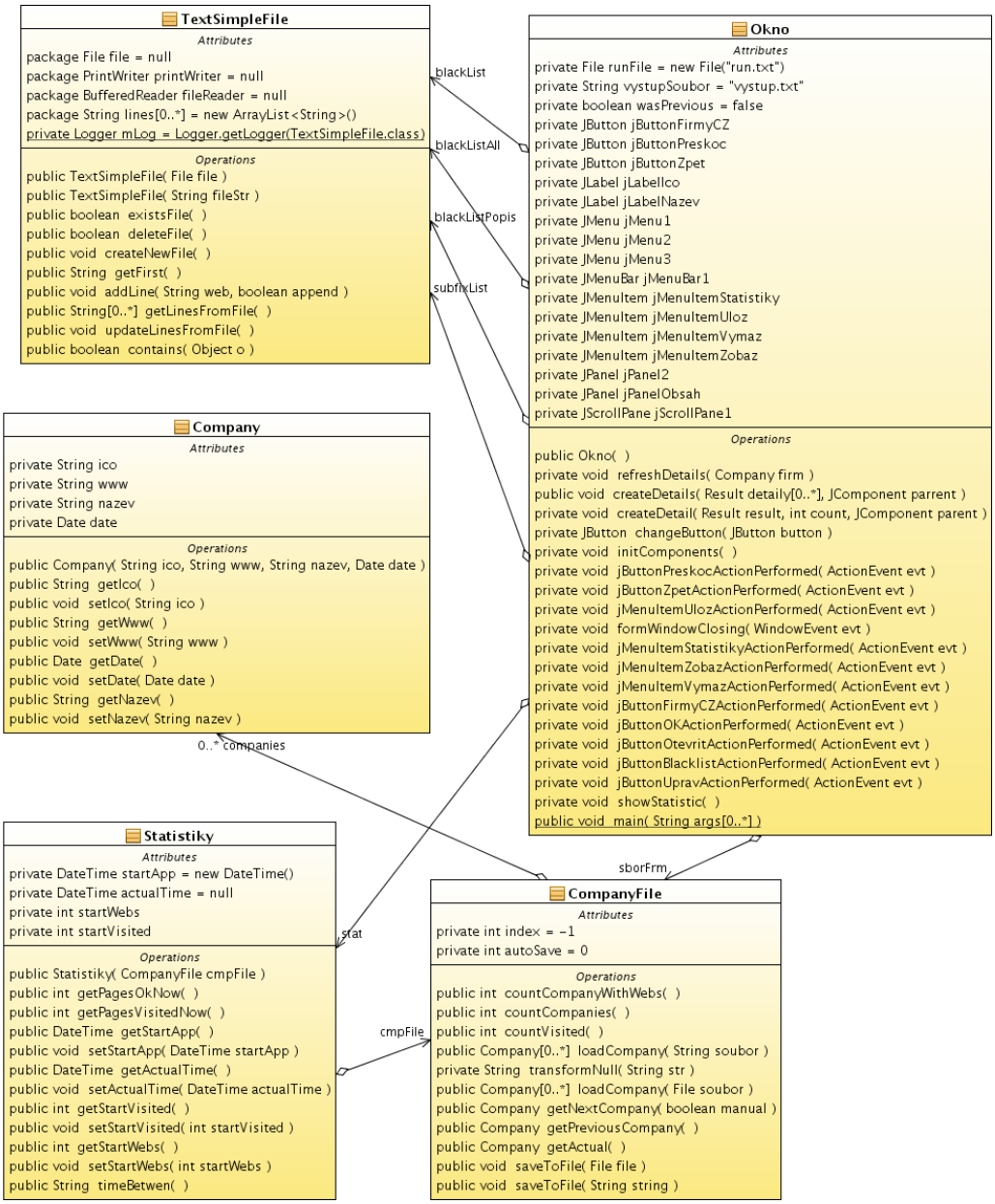
[14] GOSPODNETIC, Otis, HATCHER, Erik. Lucene in Action. Manning, 2004. 456 s. ISBN 1932394281.

[15] LOTON, Tony. Web Content Mining with Java. Wiley, 2002. 320 s. ISBN 047084311X.

[16] ECKEL, Bruce. Thinking in Java. 4th edition. Prentice Hall, 2006. 1150 s. ISBN 0131872486.

[17] SIERRA, Kathy. Head First Servlets and JSP. 2nd edition. O'Reilly Media, Inc., 2008. 911 s. ISBN 0596516681.

Příloha A



Obrázek A1: Diagram tříd aplikace pro získávání webových stránek

Obsah přiloženého CD

Na přiloženém CD najdete aplikace:

- RegainAppCrawler – aplikace pro ukládání stránek do indexu,
- RegainSearch – aplikace pro vyhledávání nad vytvořeným indexem (nasazená na adrese www.analyza-trhu.cz),
- GoogleSearchApp – jednoúčelová aplikace pro získání webových stránek firem.