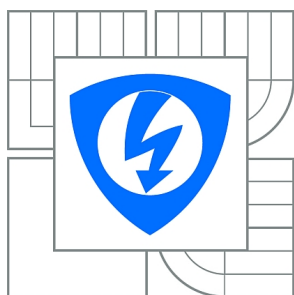


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH
TECHNOLOGIÍ
ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF TELECOMMUNICATIONS

EXTRAKCE TEXTOVÝCH DAT Z INTERNETOVÝCH STRÁNEK

SEMESTRÁLNÍ PRÁCE
SEMESTRAL THESIS

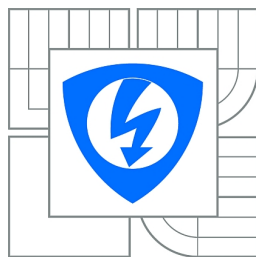
AUTOR PRÁCE
AUTHOR

Bc. ZDENĚK MAZAL

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. LUCIE FOJTOVÁ

BRNO 2010



**VYSOKÉ UČENÍ
TECHNICKÉ V BRNĚ**

**Fakulta elektrotechniky
a komunikačních technologií**

Ústav telekomunikací

Semestrální práce

magisterský navazující studijní obor
Telekomunikační a informační technika

Student: Bc. Zdeněk Mazal

ID: 77854

Ročník: 2

Akademický rok: 2010/2011

NÁZEV TÉMATU:

Extrakce textových dat z internetových stránek

POKYNY PRO VYPRACOVÁNÍ:

Prostudujte a stručně popište problematiku extrakce textových dat z internetových stránek. Na tomto základě navrhnete koncept algoritmu pro získávání textových dat a implementujte ho v některém z programovacích jazyků (Java, C++). Ověřte funkčnost programu na reálných datech a vytvořte databázi extrahovaných textů, která bude poté využita pro další zpracování.

DOPORUČENÁ LITERATURA:

[1] Gupta, S., Kaiser, G.: Extracting content from accessible web pages. ACM International Conference Proceeding Series; Vol. 88 [online], Dostupné z [www](http://portal.acm.org/citation.cfm?doid=1061811.1061816):

<http://portal.acm.org/citation.cfm?doid=1061811.1061816>

[2] Gottron, T., Martin, L.: Estimating web site readability using content extraction. WWW '09:

Proceedings of the 18th international conference on World wide web, [online], Dostupné z [www](http://www2009.org/proceedings/pdf/p1169.pdf):

Termín zadání: 21.9.2010

Termín odevzdání: 15.12.2010

Vedoucí práce: Ing. Lucie Fojtová

prof. Ing. Kamil Vrba, CSc.

Předseda oborové rady

UPOZORNĚNÍ:

Autor semestrální práce nesmí při vytváření semestrální práce porušit autorská práva třetích osob, zejména nesmí zasahovat nedovoleným způsobem do cizích autorských práv osobnostních a musí si být plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č.40/2009 Sb.

ABSTRAKT

Tato práce se zabývá získáváním textových dat z webových stránek, přehledem jednotlivých wrapperů pro stahování textu a způsobů jejich extrakce dat. Obsahuje i přehled nejpoužívanějších programů pro extrakci dat z internetu. Součástí je program, vytvořený v programovacím jazyku Java, který umožňuje získávat textová data z konkrétních webových stránek a ukládat je do xml souboru.

KLÍČOVÁ SLOVA

Extrakce informací, wrappery, problémy při extrakci dat, Java program, získání textových dat z webu.

ABSTRACT

This work focus at data and especially text mining from Web pages, an overview of programs for downloading the text and ways of their extraction. It also contains an overview of the most frequently used programs for extracting data from internet. The output of this thesis is a Java program that can download text from a selection of servers and save them into xml file.

KEYWORDS

Information extraction, wrapper, problems with extraction data, the Java program, mining text from Web pages

MAZAL, Zdeněk *Extrakce textových dat z internetových stránek*: diplomová práce. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2011. 43 s. Vedoucí práce byl Ing. Lucie Fojtová

PROHLÁŠENÍ

Prohlašuji, že svou diplomovou práci na téma „Extrakce textových dat z internetových stránek“ jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.

Brno

.....

(podpis autora)

OBSAH

Úvod	9
1 Teoretický úvod	10
1.1 Potřeba extrakce dat z Internetu a dolování dat	10
1.2 Třídění dokumentů	11
1.3 Získávání dat ze strukturovaných dokumentů	14
1.3.1 Manuální extrakce	14
1.3.2 Automatická extrakce	14
1.3.3 Wrappery	14
1.3.4 Dokumentový objektový model (DOM)	16
1.4 Rozbor již existujících programů	19
1.4.1 WebData Extractor 1.6	19
1.4.2 Web Data Extractor 8.0	19
1.4.3 Happy Harvest 2.5.6	19
1.4.4 Internetové dolování dat ze stránek	20
1.4.5 Web-Harvest 2.0	20
1.4.6 HTMLasText	21
1.5 Základní struktura html stránek	22
1.6 Kódování stránek	23
1.7 Problémy při extrakcích dat	25
2 Praktická část	27
2.1 Zadání projektu	27
2.2 Výstup programu	27
2.3 Návrh programu	27
2.4 Výběr programovacího jazyka	29
2.5 Realizace programu a jeho běh	29
2.5.1 Vytvoření databáze odkazů	30
2.5.2 Vysektování požadovaného textu	31
2.5.3 Uložení textu do xml souboru	32
2.6 Grafické uživatelské prostředí	32
2.7 Jednotlivé třídy programu	33

2.8 Test programu	35
3 Závěr	37
Literatura	38
Seznam symbolů, veličin a zkratk	41
Seznam příloh	42
A CD se zdrojovými kódy	43

SEZNAM OBRÁZKŮ

1.1	Celkové zpracování dolování v textu [1]	13
1.2	Diagram vytváření termu [1]	13
1.3	Struktura dokumentového objektového modelu	17
1.4	GUI Web-Harvest 2.0	20
1.5	GUI HTMLasText	22
2.1	Účel vytváření databáze textů	27
2.2	Schéma běhu programu	30
2.3	Diagram stahování odkazů	31
2.4	Vzhled programu a ovládací prvky	32
2.5	Vzhled programu pro uživatelské nastavení	34

SEZNAM TABULEK

2.1	Čas potřebný pro stažení a vyfiltrování 200 stránek	35
-----	---------------------------------------------------------------	----

ÚVOD

S rozvojem Internetu je stále více textových informací vyhledáváno za pomoci tohoto média, ať už to jsou novinové články, odborné studie nebo postřehy lidí. Stále větší procento lidí se odklání od tištěných novinových deníků a využívá jako zdroj každodenních informací internetové zpravodajské deníky. Roste proto i potřeba tyto informace třídit a zpracovávat. Největším problémem, s kterým se musejí nástroje pro extrakci textových informací vypořádat, aby bylo dosaženo požadovaných výsledků, je struktura internetových stránek. Neboť stránky se zprávami obsahují nejenom text samotné zprávy, ale také velké množství reklamních banerů, upozornění a dalších prvků, které nejsou pro danou zprávu důležité.

Diplomová práce se zabývá potřebou extrakcí dat z Internetu a dolování dat. Dále jsou představeny některé z problémů, které mohou vzniknout při dolování textů a jejich řešení. Prezentuje také již vytvořená řešení extrakce textových dat z internetu a jejich ukládání. Součástí práce je i návrh a realizace programu v programovacím jazyku Java, který dokáže extrahovat informace z určených webových stránek. Aby bylo dosaženo kvalitních výstupních dat, byla zvolena ruční konfigurace vstupních parametrů a to pro pět českých zpravodajských serverů. Pro jednoduchou obsluhu programu bylo vytvořeno grafické uživatelské prostředí pomocí knihovny SWING. Získaná data, jako je datum a text, pak budou ukládána do dané struktury xml souboru, z které budou přístupné pro další programy.

1 TEORETICKÝ ÚVOD

1.1 Potřeba extrakce dat z Internetu a dolování dat

Se vzrůstajícím počtem dat na Internetu dochází ke ztrátě přehlednosti informací v nich obsažených. Metodami získávání potřebných informací z množství rozdílných dat se zabývá dolování dat. Jeho úkolem je shromažďování dat a jejich následná analýza. K analýze dat se využívají statistické metody, neuronové sítě nebo samoučící algoritmy. Mezi základní kroky dolování dat dle [13] patří:

- Pochopení cílů projektu – zaměření a rozebrání hlavních cílů, k nimž má dolování dat směřovat, tedy i plán jakých cílů má být dosaženo.
- Pochopení dat projektu – prvotní sběr dat a jejich následná analýza kvality, prvotní vzhled dat, hledání skrytých informací.
- Příprava dat – předzpracování dat a uložení do správného formátu a struktury, které zpracují navazující programy.
- Hodnocení dat – v této fázi dochází k vyhodnocení výsledků, tedy zda-li došlo k požadovaným výsledkům, které jsme potřebovali.

Dolování dat je obecnější formou získávání dat, práce se bude zabývat především dolováním textových dat. Podle hrubého odhadu [5] je 80% světových elektronických dat v textové podobě. Převážnou většinu dokumentů pro dolování v textech, které jsou obsahově zajímavé, tvoří elektronické dokumenty, prezentace, emailová komunikace, diskuzní fóra, každodenní vědecké a novinové zprávy. Proto je velice důležité surová data zpracovat a následně vyhodnocovat. Techniky dolování v textech byly úspěšně využity například pro vyhodnocování uživatelských stížností při emailové komunikaci, indexaci dokumentů a vyhledávání v databázích.

Proces dolování textových dat je možno rozdělit na dvě hlavní fáze. První fází je předzpracování vstupních dat, kdy z výchozího dokumentu je vyselektován požadovaný text, tedy jsou vynechány všechny ostatní prvky dokumentu, jako jsou obrázky, značkovací prvky a další netextové informace. Následně se ze získaných dat odebere formátování textu (velikost písma, písmo, styl), další úpravy záleží na účelu, pro který je obsah získáván. Většinou jsou odstraněna i slova, která neovlivní analýzu textových informací – spojky, částice atd. Mezi další nástroje, které se používají

při této fázi zpracování textu, patří především spell-checker (kontrola pravopisu) a lematizátor. Lemmatizátor slouží k určení tzv. lemmy pro každé slovo v analyzovaném textu, tedy jeho základní tvar ("zkouškou" = "zkouška"), často jsou i přiřazené mluvnické kategorie (podstatné jméno, 6. pád apod.). Ve složitějších případech musí lematizátor odvozovat základní tvar pomocí kontextu z vět. [13]

Poslední fází dolování textových dat je ohodnocení textu a zobrazení výsledků. Dochází k vyhodnocení jednotlivých termů vzniklých v předešlé fázi a jejich následná analýza a vyhodnocení.

1.2 Třídění dokumentů

Textová data jsou většinou získávána ne z jednoho dokumentu, ale z celé množiny dokumentů. Je proto potřeba před samotnou extrakcí dokumenty roztrždit. Mezi nejpoužívanější techniky třídění dokumentů dle [7] patří:

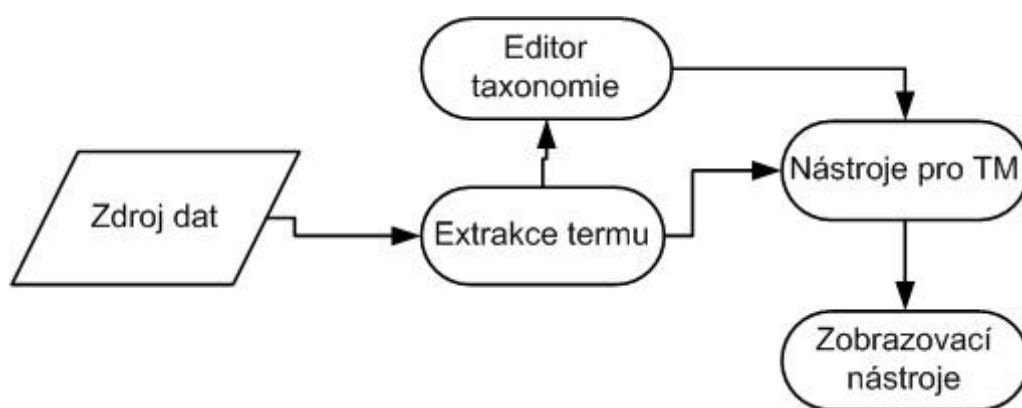
- **Kategorizace** – dokumenty se přiřazují do předem daných tříd podle jejich obsahu. Systém pro toto třídění má nejprve velkou množinu správně zařazených vzorových dokumentů - trénovacích dat. Jeho učící algoritmus si podle nich vybuduje přiřazovací hodnoty (rozhodovací strom), pozitivní hodnocení pro dokument v kategorii a negativní hodnocení pro dokumenty v ostatních kategoriích. Podle těchto rozhodovacích stromů systém zařazuje nové dokumenty do příslušných tříd. Automatické třídění textu do kategorií má velmi široké praktické využití včetně indexování dokumentů pro vyhledávání, organizování rozsáhlých webových zdrojů a automatické extrakce metadat. Metadata jsou strukturovaná data, která mohou sloužit k snadnému vyhledávání. Příkladem metadat může být katalogizační lístek v knihovně.
- **Shlukování informací** – dokumenty podle tohoto třídění jsou rozděleny na menší bloky (shluky), každý takto vzniklý blok je popsán několika společnými vlastnostmi a znaky. Je proto důležité, aby data vkládaná do těchto jednotlivých shluků obsahovala tyto znaky stejné nebo hodně podobné. Na závěr je pak každý shluk popsán a charakterizován.
- **Extrakce informací** – tato metoda se používá u strukturovaných dokumentů, které obsahují značkovací jazyk, tedy především dokumenty xml a html. Metoda vyžaduje trénovací data, podle kterých pozná, které části dokumentu

se budou extrahovat podle značkovacích prvků. Tento způsob extrakce má nevýhodu, že pokud dojde ke změně jednotlivých značkovacích prvků nebo struktuře dokumentu, bude interpretace extrakce chybná. Extrakce informací je rozebrána dále podrobněji v kapitole 1.3.

- **Sumarizace**– rekonstruuje a zkracuje (komprese) původní text do podoby lehce čitelné pro člověka, tedy žádné značky ani algoritmy. Existuje několik druhů sumarizace. První ze způsobů je jednoduché vytvoření výsledného extraktu přímo zkopírováním vět a slov obsažených v původním textu. Naproti tomu druhým způsobem vznikne extrakt nepřímo textem, který není obsažen ve vstupu nebo ho vyjadřuje jiným způsobem, využívá se statistických a heuristických metod nebo kombinací obou. Dalším způsobem je abstrakt, který hodnotí dokument a kvalitu tohoto dokumentu na základě určitých kritérií. Tyto sumarizace umožňují čtenáři v rychlosti zjistit obsah textu.
- **Selekce informací**– na základě trénovacího dokumentu jsou vybrány a uloženy sady klíčových slov do řídicího slovníku, tento slovník je pak použit u třídění nových dokumentů. Každá sada klíčových slov ze slovníku je spojena s určitým dokumentem a řídicí algoritmus si vytvoří klasifikátory, podle nichž se rozhoduje, ke kterým slovům ze slovníku nový dokument přiřadí. Tomuto způsobu se říká přiřazení ke klíčovému slovu. Další metodou v této kategorii je metoda zvaná získávání klíčových slov, kde na rozdíl od první metody neexistuje řídicí slovník, ale klíčová slova nebo celá souvětí jsou brána přímo z dokumentu tak, aby ho co nejvíce charakterizovala.
- **Identifikace jazyka** – poměrně jednoduchá metoda, která třídí dle jednotlivých izolovaných slov nebo na základě po sobě jdoucích písmen, porovnáváním se slovy v databázi.[22]
- **Třídění dle autora** – většina dokumentů obsahuje autora textu nebo autor není potřeba, ale u ostatních je možné určit autora textu poměrně jednoduchou metodou. Pokud máme dostatečně velkou trénovací množinu dat od daného autora, tak na základě výskytu a používání jednotlivých slov lze odvodit autora.
- **Zobrazení**– tato metoda se používá jako doplňková k ostatním metodám. Například při shlukování ve 2-D prostoru ukazuje, jaké je rozložení dokumentů a jakým způsobem byly shluky vytvořeny. Je možno proto lépe posoudit, zda

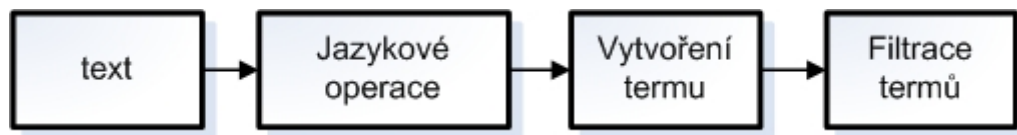
bylo provedeno správně. Stejně tak při kategorizaci metodou rozhodovacích stromů bývá používána vizualizace. [13]

Obrázek 1.1 představuje hlavní strukturu dolování textu. Prvním krokem je získání databáze surového textu z dokumentů. K tomuto textu jsou přiřazeny základní objekty, následuje proces generování atributů pro nástroje dolování v textu, které určí tzv. KDD (knowledge discovery) – vyhledávání znalostí. V konečné fázi dojde k zobrazení výsledků.



Obr. 1.1: Celkové zpracování dolování v textu [1]

Na obrázku 1.2 jsou zobrazeny jednotlivé kroky pro vytvoření termu. Do bloku jazykové operace patří vytváření tokenů, označování slov a lemmatizace. Při vytváření tokenů jsou například odstraněna všechna interpunkční znaménka (?,!,=..). Slova, která mají stejný význam nahrazena jedním slovem nebo data převedena do jednoho formátu. U označování slov dochází k vyhodnocení slov zda-li se jedná například o podstatné jméno v množném, jednotném čísle, přídavné jméno, číslo nebo vlastní jméno apod. dle potřeby. Pro vytvoření termů se používají pouze slova, která jsou označena a vyberou se taková, která splňují kritéria pro výběr. [1]



Obr. 1.2: Diagram vytváření termu [1]

1.3 Získávání dat ze strukturovaných dokumentů

Jedním z rozsáhlých zdrojů, ze kterých je možno získávat textová data, jsou strukturované dokumenty. Nejpoužívanější formáty pro strukturované dokumenty jsou html a xml. Hlavním úkolem systému na dolování textu ze strukturovaných dat je odstranění všech značek, značkovacích prvků, obrázků a dalších prvků dokumentu, které by znemožnily zpracování získaného textu dalším nástrojům. Výstupem je tedy prostý text nebo jednodušeji strukturovaný dokument, který je čitelný nástrojům, které provádí další analýzu a zařazení textu. Podle [22] je možné rozlišit tři hlavní techniky dolování textu ze strukturovaných dokumentů – wrappery, spojení shlukovaných dokumentů a hodnocení webových stránek.

1.3.1 Manuální extrakce

Na základě pozorování a rozboru zdrojového kódu požadované internetové stránky (neměnné struktury) se napíše program, který extrahuje cílová data. Hlavním problémem tohoto řešení je, že v případě změny internetové stránky přestává program fungovat a musí dojít k jeho aktualizaci. Na druhé straně nespornou výhodou tohoto řešení je úspěšnost a přesnost dolovaných dat.

1.3.2 Automatická extrakce

Systém automaticky hledá vzory pro extrakci dat. Protože využívá automatického prohledávání, je tento způsob vhodný pro hledání v rozsáhlých a nekonkrétních stránkách. Při tomto druhu extrakce musí mít systém nejprve data, na kterých si natrénuje, kde jsou potřebná data umístěna a jaké mají shodné prvky, podle nich si vytvoří pravidla, která používá na nové dokumenty.

1.3.3 Wrappery

Wrapper [9] je nástroj pro extrakci informací konkrétního informačního zdroje. Každý wrapper se skládá z několika extrakčních pravidel. Z překladu wrapper značí obálku, což dobře vystihuje způsob práce wrapperu, na začátku procesu extrakce jsou určena extrakční pravidla (obálky jednotlivých částí dokumentu), při průchodu

dokumentu jsou vybírány části, které vyhovují a jsou obklopeny řetězcí z množiny extrakčních pravidel. Pro každý dokument je potřeba vytvořit novou množinu extrakčních pravidel. Extrakční pravidla jsou tedy úzce spojena se strukturou dokumentu. Existuje několik způsobů, jak ze strukturovaných dat pomocí wrapperu získat potřebná data. V následující kapitole si tyto způsoby popíšeme. Wrappery lze rozdělit na dvě hlavní skupiny: řetězcové wrappery, které vidí stránku jako řetězec textu a wrappery využívající dokumentových objektových stromů, kde wrapper hledá důležitou část vytvořeného stromu.

Řetězové wrappery

Řetězové wrappery pracují se strukturovanými dokumenty jako s proudem za sebou jdoucích znaků. Jednotlivé části, které je potřeba extrahovat, jsou označeny konkrétními značkami – html značky. Jak je patrné, tak dokumenty pro tyto řetězové wrappery musí mít předem známou strukturu. Existuje několik druhů řetězových wrapperů [9], nejpoužívanější jsou LR, HLRT, OCLR, HOCLRT, NLR a NHLRT.

- **Zleva-doprava** - označuje metodu Left-Right, textové informace pro extrakci jsou ohraničeny levými a pravými značkami. Metoda prochází dokument do nalezení požadované značky a zde začne se získáváním textu, dokud nenarazí na značku označující konec.
- **HLRT(head-left-right-tail)** - navazuje na metodu zleva-doprava a snaží se snížit její nepřesné získávání dat. Kromě parametrů left a right se přihlíží ještě na další parametry head a tail, tedy konec hlavičky a konec dokumentu. Systém s touto metodou bude extrahovat dokument od výskytu konce hlavičky po označení konce dokumentu s využitím metody left-right.
- **OCLR** – opening-closing-left-right. U HLRT jsou data pro extrakci oddělena parametry head a tail, podobně pracuje i metoda OCLR, s tím rozdílem, že dokument je rozčleněn na několik částí, z nichž každá je označena parametry opening a closing, ostatní text se do extrakce nezahrnuje. Mezi těmito parametry se opět využívá metoda left-right. Tím se zamezí získání dat z nedůležitých částí dokumentu.[11]
- **HOCLRT**– head-open-close-left-right-tail. Jedná se o kombinaci metod OCLR a HLRT, kdy mezi značkami head a tail jsou získávány pouze úseky open a close.

- **N-LR a N-HLRT** – na rozdíl od předchozích metod, které se využívají u dat s pravidelnou strukturou, tato metoda se používá při extrakci hierarchických stromových struktur jako například víceúrovňové seznamy.

Stromové wrappery

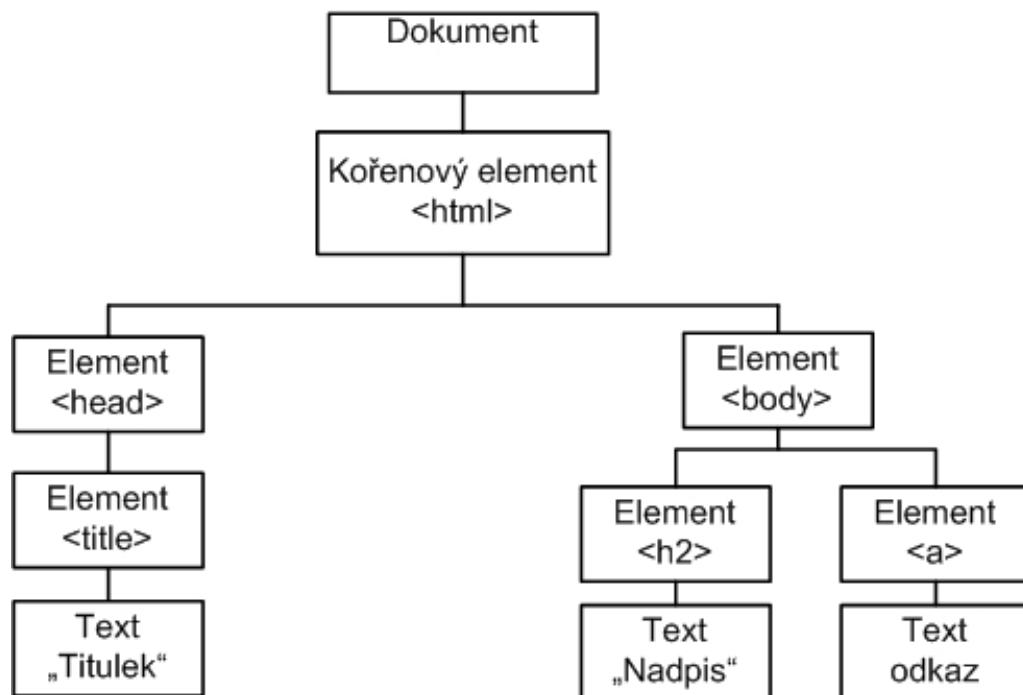
Stromové wrappery nepracují přímo s html nebo xml kódem jak tomu bylo u řetězových wrapperů, ale vstupními daty jsou pro ně dokumentové objektové stromy (Document Object Model – dokumentový objektový model), v kterých vyhledávají relevantní stromové větve. Na následujících řádcích bude dokumentový objektový model rozebrán podrobněji.

1.3.4 Dokumentový objektový model (DOM)

Dokumentový objektový model je aplikační programové rozhraní, které umožňuje programům a skriptům pro dynamický přístup aktualizovat strukturu, styl a obsah dokumentů (elementy, atributy, text..). Dokument může být zpracováván a jeho výsledky mohou být uloženy zpět do stránky [20]. DOM strom reprezentuje HTML nebo XML dokument jako data se stromovou strukturou. Díky této struktuře je možné rozlišit podřazené, nadřazené nebo rovnocenné elementy. Protože se celý syntakticky analyzovaný dokument načítá do paměti, používá se tam, kde se k jednotlivým elementům přistupuje náhodně nebo pravidelně. V případě, že je nutná jednorázová úprava dokumentu, používá se sekvenční metoda SAX (spimple API for xml), která je méně paměťově náročná a rychlejší. Obrázek 1.3 vyobrazuje ukázkovou strukturu DOM stromu.

Dokumentový objektový model je rozdělen do dvou hlavních částí: DOM jádro a DOM HTML a XML. Dokumentové objektové jádro představuje nástroje pro XML dokumenty a slouží také jako základ pro DOM HTML. DOM HTML a XML používané spolu s jádrem poskytují pohodlný náhled do dokumentu. Skládají se z objektů a metod, které umožňují přímý přístup do konkrétních dokumentů. Podle [17] dokumentový objektový model přistupuje k HTML dokumentu přes následující objekty:

- Element – zástupce jednotlivých elementů (H1, head...).
- Attr – zástupce atributů (style, class..).
- Comment – zástupce komentářů .



Obr. 1.3: Struktura dokumentového objektového modelu

- Text – zastupuje text.
- Ostatní méně využívané objekty (jmenné prostory).

Základním a nejdůležitějším elementem dokumentu je kořenový element, který je zastoupen objektem document, proto se ke každému prvku, který mu je podřazen přistupuje pomocí tohoto objektu. Jakoukoliv html stránku lze rozkreslit na stromovou strukturu, kde nejvyšší pozici zaujímá kořenový element html, který se dále větví na head a body, jenž jsou mezi sebou sourozenci a dětmi kořenového elementu. Oba jsou objekty typu element. Element h1 je dítětem elementu body a potomkem kořenového elementu. K elementu h1 je připojen objekt typu (attr) - atribut style. Element p je sourozencem elementu h1 a má dvě děti - jeden objekt typu element - strong a jeden typu comment - komentář. Zde se poprvé objevují textové objekty. Prvním je "DOM", který je dítětem elementu h1 a druhým je "Document Object Model" dítě elementu strong [17].

Pro vyhledání relevantních údajů z dokumentových objektových modelů existuje celá řada způsobů. Mezi hlavní patří konečné stromové automaty a výběry řetězců s využitím dotazovacího jazyka XPath a Xquery.

Xpath

Jedná se o počítačový jazyk, který se používá pro identifikaci jednotlivých uzlů v xml dokumentu a jeho následnou extrakci. Je standardizován a řízen organizací W3C. Pomocí tohoto jazyka lze vybírat a upravovat jednotlivé elementy v dokumentu. V dokumentovém objektovém stromu definuje správnou cestu, tedy přechody mezi jednotlivými uzly. Tyto přechody jsou definovány třemi složkami: osou (směrem prohledávání), testy uzlů ve svislém směru a posloupnost predikátů [19].

1.4 Rozbor již existujících programů

Tato kapitola je zaměřená na již realizovaná a používaná řešení, zabývající se získáváním a ukládáním potřebných údajů z internetových stránek, která nachází uplatnění v mnoha specifických oborech. U většiny nalezených programů je poměrně složité nastavení, platí tedy čím lepší a kvalitnější výstup, tím složitější a propracovanější nastavování. Jednotlivé programy mají své výhody a nevýhody v různých situacích, protože každý je naprogramován pro různá prostředí. Nejvíce bude rozebráno řešení Web-Harvest 2.0, jenž svými možnostmi a kvalitou převyšuje ostatní programy.

1.4.1 WebData Extractor 1.6

Program slouží pro extrakci veškerých elementů ze stránek, umožňující stáhnout jednotlivé obrázky, skripty, hudbu, screeny, emaily, linky atd. Program umí procházet i rekurzivně stránky do hloubky, na které směřují odkazy na zadané stránce. Lze tedy weby prohlížet v offline režimu. Z internetových stránek extrahuje veškerý text pouhým odstraňováním html tagů, ale neumí text filtrovat a nezobrazovat tlačítka a reklamy. [14]

1.4.2 Web Data Extractor 8.0

Jedná se o placený program, který dokáže získávat url adresy, meta tagy, textové informace, emaily, telefony a faxy. Pro každou extrakci je možné detailní nastavení toho, co se má a nemá stahovat. Taktéž umožňuje rekurzivní procházení stránek, jak v rámci webu, tak i na ostatní weby, na které vedou url adresy.

1.4.3 Happy Harvest 2.5.6

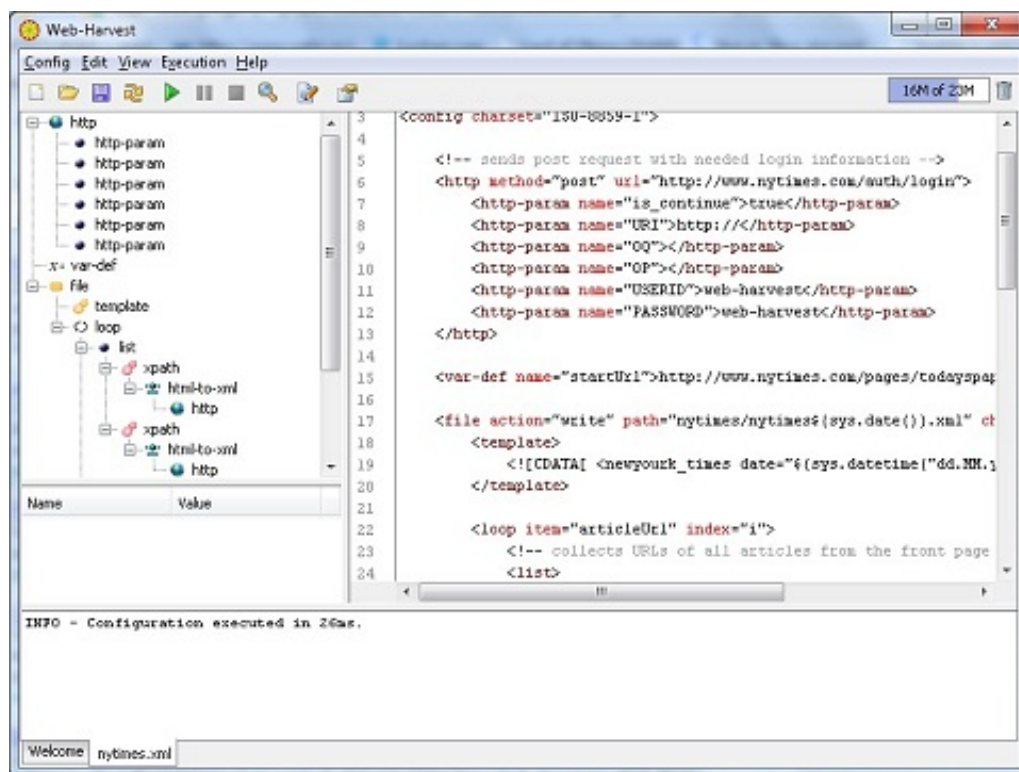
Přehledný a jednoduchý program s ukázkovými nastaveními a získáváním dat. Extrakce z webových stránek probíhá podle předem daných pravidel, které si uživatel sám zvolí. Jednotlivá pravidla jde nastavovat samostatně pro extrakci obrázků, textů, tabulkových dat, linkových adres a komentářů. [3]

1.4.4 Internetové dolování dat ze stránek

Kromě spustitelných programů pro práci s html soubory a internetovými stránkami existují i online prostředí, která nabízejí podobnou službu. Příkladem může být konzole od společnosti yahoo na stránce <http://developer.yahoo.com/yql/> . Využívá jazyka YQL (Yahoo! Query Language), což je expresivní SQL jazyk, který umožňuje filtrovat, strukturovat a třídit data z webových stránek. Avšak tato konzola je omezena na určitý počet dotazů.

1.4.5 Web-Harvest 2.0

Patří mezi nejpropracovanější programy na extrahování užitečných informací a práci s xml, html dokumentů a webových stránek. Program Web-Harvest [21] je napsán v jazyce Java a jedná se o opensource licenci, konkrétně BSD licence. Pro získání dat z dokumentů a stránek využívá program XSLT, Xpath, Xquery a regulárních výrazů. Navíc lze doplnit do Web-Harvest vlastní knihovnu pro rozšíření schopnosti extrakce. Web-Harvest podporuje podmíněné větvení, práci se soubory, html a xml zpracování a zpracování výjimek. Každý proces získání dat je definován



Obr. 1.4: GUI Web-Harvest 2.0

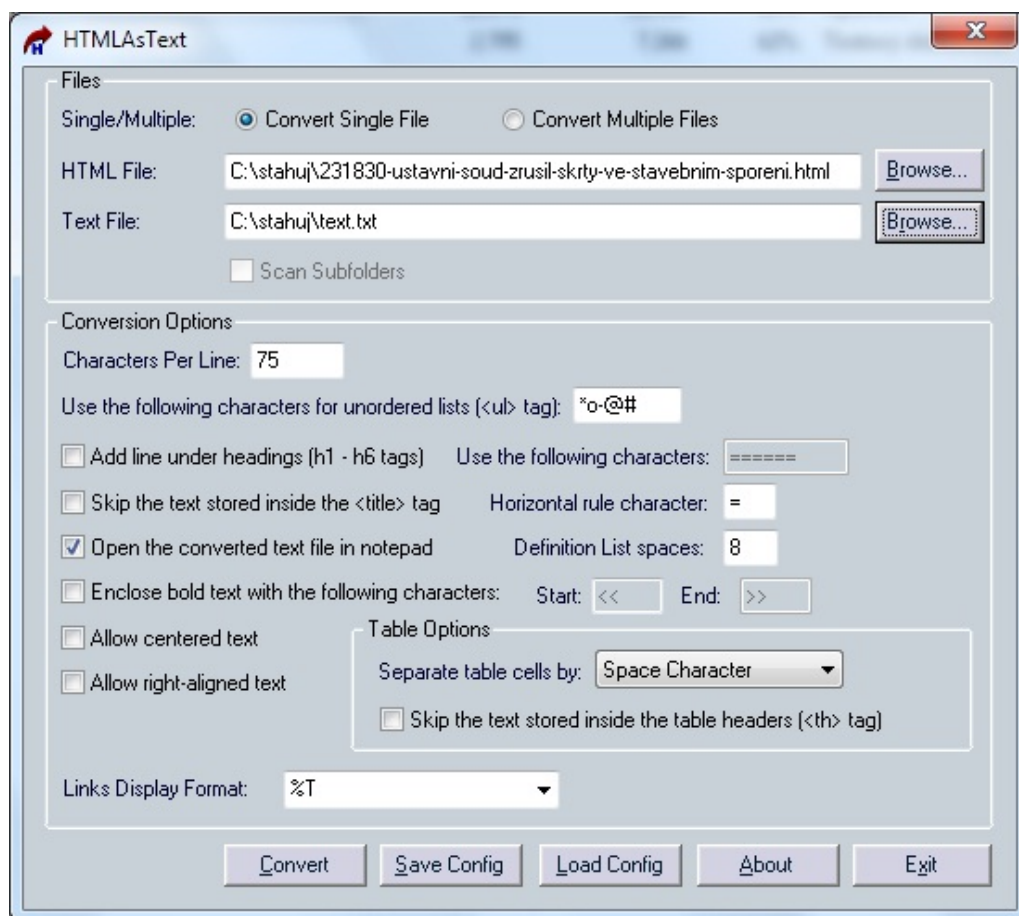
v jednom nebo více konfiguračních souborů, pomocí xml formátu. Každý proces je popsán konkrétním prvkem xml nebo strukturou prvků xml. Web-Harvest používá GUI – grafické uživatelské rozhraní, které usnadňuje práci s programem a zvyšuje přehlednost, jak je patrné z obrázku 2. Mimo GUI je možné program spouštět a konfigurovat pomocí příkazové řádky. Následující ukázkový kód zobrazuje první řádky nastavení pro extrakci textových dat ze stránek idnes.cz. Jedná se pouze o část kódu pro představu, jakým způsobem se zadávají jednotlivá získávací pravidla:

```
<?xml version="1.0" encoding="UTF-8"?>
<config charset="windows-1250">
<var-def name="startUrl">http://zpravy.idnes.cz/padaly-z-nebe.html</var-def>
<file action="write" path="idnes/idnes${sys.date()}.xml" charset="UTF-8">
<template>
<![CDATA[ <idnes_cas date="${sys.datetime("dd.MM.yyyy")}"> ]]>
</template>
<loop item="a" index="i">
```

Jak je patrné z ukázkového kódu nastavování jednotlivých pravidel, je poměrně složité i pro zkušené uživatele, kteří se musí naučit syntaxi programu Web-Harvest. Na druhou stranu poskytuje program výhodu nastavení co nejmenších detailů, přesně podle potřeb uživatele. Manuál obsahuje i ukázkové příklady pro nastavení základních úloh. Například extrakce všech stránek z internetových novinových článků podle určitých omezení, jednoduché nastavení wrapperu webových stránek, který stahuje celé stránky a ukládá je do html souboru. Nebo ukázka, jak zobrazit všechny výrobky značky Canon z webových stránek yahoo.com a stáhnout zadané obrázky.

1.4.6 HTMLasText

Tento program umožňuje vyextrahování textu z html souboru, tedy nepřipojuje se k serveru, kde stáhne stránku, ale využívá již staženou stránku v html formátu. Vyextrahovaný text pak ukládá do textového souboru. Umožňuje jednotlivé uložení a nahrání uživatelského konfiguračního souboru. Nastavení však obsahuje méně položek, než by bylo potřeba pro získání čistého textu, výsledný text například obsahuje části MENU a podobně.



Obr. 1.5: GUI HTMLasText

1.5 Základní struktura html stránek

Každá stránka by měla mít určitou strukturu, která zaručí její správné zobrazení ve všech prohlížečích. Standarty a schvalováním jednotlivých prvků se zabývá World Wide Web Consortium, zkráceně W3C, které existuje už od roku 1994. Jeho hlavním cílem je rozvíjet world wide web do jeho plného potenciálu vývojem protokolů a směrnic, které zajistí dlouhodobý růst webu. [18] Html dokument tvoří několik značek (tagů), jenž dodávají konečný vzhled dokumentu nebo mají informativní charakter. Převážnou většinu html značek tvoří párové značky, které vymezují konec a začátek působení značky. Ale existují i takové, které jsou nepárové, například !doctype, určující typ dokumentu. Hlavní kostru html dokumentu je možné rozdělit na hlavičku dokumentu a tělo dokumentu, základ tedy vypadá následovně:

```
<HTML>
```

```
<HEAD>
```

hlavička_dokumentu

</HEAD>

<BODY>

tělo_dokumentu

</BODY>

</HTML>

Každý html dokument je uvozen <HTML> a </HTML>, které jsou povinné, avšak většina prohlížečů si tento znak umí doplnit. Do hlavičky jsou hlavně uloženy informace pro klientské prohlížeče a v prohlížeči se nezobrazují (titulek, použité kódování, klíčová slova stránky a další vlastnosti stránky). Titulek je možné použít pro popis celého extrahovaného textu. [8]

Pro uživatele je daleko důležitější část ohraničená značkami <BODY> a </BODY>, kde jsou zapsána data, která se zobrazují uživateli (obrázky, texty, animace atd.). Zde se nachází jednotlivé prvky, které jsou určeny pro extrakci. Hodně významné při extrakci textů jsou tzv. obálky bez významu, tedy neutrální značky, které nemají vliv na formátování. Data uvnitř těchto obálek mají stejné vlastnosti nebo patří do stejné třídy. Ve většině případů jsou těmito obálkami uvozeny texty, které se extrahují, například pro server idnes.cz jsou tyto obálky označeny FULLTEXTSTART pro začátek textu a FULLTEXTSTOP pro konec textu článku. I když už tedy získáme oblast dokumentu, kde se text pro extrakci nachází, není to konečná fáze, neboť oblast obsahuje další značky a prvky, které by text znehodnocovaly. Jedná se především o zalomení řádků, odkazy na stránky v textu, formátování textu, obrázky. Tyto značky a data mezi nimi je třeba odstranit pro požadovaný výsledek. Pokud je potřeba zachovat určité formátování textu, lze některé značky ponechat a přeformátovat na tvar, který zpracují další nástroje, pro které je text určen.

1.6 Kódování stránek

Základním problémem při stažení a uložení stránek je kódování. Pro anglickou (americkou) gramatiku dostačuje pro korektní zobrazení 128 znaků, to je ale nedostačující počet pro další používané jazyky včetně češtiny. Proto tyto znaky musely být rozšířeny, tak aby došlo ke korektnímu zobrazení všech používaných znaků jednotlivých jazyků. Znaková sada byla rozšířena čísly od 128 do 255, která se přizpů-

sobují jednotlivým jazykům a rozlišností jejich znaků. Základ 128 znaků ASCII je společný pro všechny jazyky a rozšířením o další znaky říkáme kódování. Pokud se zvolí chybné kódování a stránka bude obsahovat znaky, které nejsou zahrnuty ve zvoleném kódování, pak dojde k znehodnocení celého staženého textu. Kódování češtiny bylo nadefinováno hned v několika standardech. [8] Kromě již téměř zaniklých kódování (kódování bratří Kamenických, Mac, PC latin...) se používají pro češtinu následující kódování:

- **Windows-1250**– standardizovaný již od roku 1996, často používaný na systémech windows, většina jeho programů (Notepad, FrontPage...) je používá jako kódování standardní. [6] Toto kódování používají např. tyto servery: www.idnes.cz, www.ihned.cz, www.neviditelnypes.lidovky.cz, www.hn.ihned.cz.
- **UTF-8**– Veškerá specifikace je v RFC 2279. Jedná se o nejčastější zápis Unicode. Do této znakové sady je možné zapsat rozdílné znaky všech jazyků. Je tedy možné psát v různých jazycích najednou. K dispozici je totiž 31 bitů, tedy 2 na 31 možných znaků. Základní vlastností tohoto kódování je možnost využití v programech, které umožňují pouze osmi bitové kódování. [2] Příklady: www.novinky.cz, www.denik.cz, www.super.cz, www.blesk.cz, www.ahaonline.cz, www.ceskenoviny.cz, www.zive.cz, www.financninoviny.cz.
- **ISO 8859-2**– Publikované v roce 1987, známé také jako Latin-2. Většina nových programů pracuje s tímto kódováním bez chyb a to i v případě produktů od Microsoft. Ostatní jazyky, které mohou používat toto kódování, se přiklání k této znakové sadě (Slovensko, Polsko, Slovinsko). Tuto znakovou sadu, také hojně využívají unixové servery a správa informačního systému České republiky. [15] Příklady: www.aktualne.centrum.cz.

Jak je patrné z přehledu kódování nejčtenějších českých zpravodajských serverů, nejpoužívanější znakovou sadou je UTF-8, kterou je možné zapsat všechny používané znaky jednotlivých jazyků. Naopak nejméně rozšířenou znakovou sadou je ISO 8859-2, která se moc často nepoužívá.

Je několik možností jak nastavit u prohlížeče správný výběr znakové sady. Jednak uživatelské, u kterého musí nastavit uživatel v prohlížeči danou znakovou sadu nebo je možné umístit konkrétní meta tag do hlavičky html kódu, například:

```
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-2">  
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
```


Tento meta tag dává informaci, jakou znakovou sadu má prohlížeč použít pro danou html stránku. V některých případech je i možné překódování stránek přímo na straně serveru. Klient zasílá na server s přehledem znakových sad, které akceptuje a server pak překóduje stránku do požadovaného kódování. Častým problémem je neuvedení použitého kódování v hlavičce nebo rozdílného kódování části stránek. V takových případech programy obtížně nebo chybně překládají dané znaky. Například `aktualne.centrum.cz` má rozdílnou hodnotu HTTP charsetu a META charset, kdy jednou je uvedeno kódování UTF-8 a v další části ISO-8859-2.

1.7 Problémy při extrakcích dat

Existuje několik rozsáhlejších problémů, s kterými se programy na získávání dat z internetových stránek musí vypořádat a bezchybně data získat. Internetové stránky jsou psány způsobem, který je dobře čitelný a zpracovatelný počítači. Pokud tedy je zapotřebí získat data, která budou dobře čitelná a zpracovatelná, je důležité změnit strukturu těchto dat a odstranit řídicí značky, sloužící pro počítače. Mezi hlavní problémy při zpracování a získávání dat z webových dokumentů patří:

- **Různá struktura**– jednotlivé webové stránky by sice měly dodržovat určitá pravidla pro strukturu konsorcia W3C, ale těchto pravidel je velké množství s řadou prvků. Programy pro dolování dat se s tímto problémem musí vypořádat, buď ručním vytvářením parametrů nebo automatickým určováním pravidel, na základě určité podobnosti jednotlivých struktur a podobnosti vytvářených stránek.
- **Chybně napsané stránky**– častou chybou v kódu html stránek jsou nezačaté párové značky, které se skládají ze začínajícího a zakončovacího znaku. S párovými značkami se objevuje další nevhodný prvek a to překřížení znaků. U všech stránek by mělo být uvedeno v jakém formátu je objekt kódován a typ dokumentu. Mezi další chyby patří hodnoty atributů bez uvozovek. S těmito nedostatky si většina prohlížečů poradí, takže chyby nemají velký vliv na zobrazení, proto nenutí vývojáře stránek tyto chyby kontrolovat a opravovat. Problém může nastat u programů, které hledají části data právě v určitých částech mezi párovými značkami. [12]
- **Změna stránek**– úprava již existujících stránek způsobí nefunkčnost získá-

vání dat u nástrojů, které jsou naprogramovány podle předem daných pravidel podle původních stránek. Pro opětovnou funkčnost se musí zajistit přeprogramování jednotlivých pravidel podle nového rozložení dat, tak aby byla pravidla opět aktuální.

- **Blokace robotů**– některé webové servery jsou blokovány na používání robotů pro stahování potřebných dat. Omezení se může týkat počtu přístupů za jednotku času nebo objemem odeslaných dat na jednu IP adresu nebo soubor IP adres. Základním řešením tohoto problému je nastavit mezi stahováním jednotlivých stránek určitý interval, po který robot čeká, než začne stahovat další stránku. Dalším řešením je připojení robota k proxy serveru, který funguje jako anonymizer, pokud tedy dojde k blokaci původní IP adresy, připojením na proxy server, dojde ke změně IP adresy a robot je v síti identifikován jako jiný počítač.

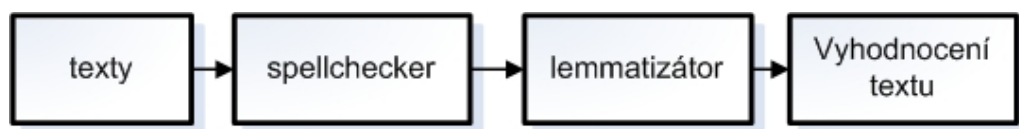
2 PRAKTICKÁ ČÁST

2.1 Zadání projektu

Hlavním cílem diplomové práce mělo být sestavení programu v jazyku java nebo c++, který ze zadaných internetových stránek dokáže získat čistá textová data, která budou tvořit souvislý text, tedy bez rušivého textu jakou jsou reklamy, menu, odkazy... Tato data uloží v požadované struktuře do xml souboru a přejde na další stránku, se kterou provede předcházející kroky.

2.2 Výstup programu

Výstupem má být soubory ve formátu xml, které budou obsahovat vydolovaný text, datum a titulek. Xml soubory se budou dále zpracovávat pomocí spell-checkeru, stemmeru a lematizátoru, které jsou součástí práce dalších diplomových a bakalářských prací. Lematizátor vytváří pro každé slovo základní tvar a určí také mluvnické kategorie. Hlavním účelem programu je tedy získávání dat a jeho ukládání pro další programy, které k nim budou přistupovat jak je patrné z obrázku 2.1.



Obr. 2.1: Účel vytváření databáze textů

2.3 Návrh programu

Při studování html kódu jednotlivých zpravodajských serverů si můžeme všimnout jednotlivých společných prvků. Mezi hlavní patří to, že veškerý důležitý text je vkládán do jednotlivých odstavců - tagů `<p>` a `</p>`. To nám zjednoduší vyhledávání požadovaného textu. Program tedy vyhledá tyto tagy a veškeré informace, ohraničené těmito znaky si uloží. Dalším společným znakem z redakčních systémů je i vkládání textu mezi určité značky. Například pro novinky.cz to je "FULLTEXTSTART" a "FULLTEXTSTOP" nebo pro ceskenoviny.cz "zacatek nasazeni

bbTextu” a ”konec nasazeni bbTextu”. Více v teoretické části 1.5. Těmito znaky získáme další upřesnění pro vyhledání správného textu. Prvními pokusy s programem byla snaha přizpůsobit vyhledávání těchto značek automaticky, tak aby bylo možné univerzální využití. Avšak při testování se nepodařilo nalézt vhodný algoritmus, aby tyto obálky našel bez chyby. Při zvýšení univerzálnosti se snížila kvalita vydolovaného textu, naopak při kladení důrazu na kvalitu textu byla univerzálnost nízká. Proto byla zvolena možnost sestavit pravidla pro čtyři servery ručně, tak aby bylo možné vydolovat text bez chyb. Navíc je možné stáhnout více stránek při jednom spuštění a program si sám vyhledává další odkazy a ukládá již stažené stránky, aby nedošlo k replikaci stránek. Grafické rozhraní také obsahuje uživatelské rozhraní, kde jsou pro představu základní vstupní parametry, se kterými program pracuje. Hlavním cílem programu je vytvoření databáze xml souborů s texty, které jsou potřebné pro další programy, jako například spell-checker.

Program lze rozdělit do čtyř hlavních kroků, které musí bezchybně vykonat, aby bylo dosaženo požadovaných výsledků:

- **Stažení stránky** – při tomto kroku dojde k navázání spojení s webovým serverem a následným uložením html kódu stránky do souboru na místní úložiště. Hlavní problém by mohl nastat v kódování stránky, neboť každá stránka používá různé znakové sady. Jestliže by došlo k chybně zadané znakové sadě, výsledná data budou znehodnocena. HTML kód stránky bude uložen do textového souboru, s nímž se bude nadále pracovat.
- **Získání textu** – ze získaného souboru s html kódem se vyfiltrují nepotřebné údaje jako jsou reklamy, části menu, odkazy, značky a řídicí informace. Výsledkem je pak čistý text, tak jak je zobrazen v internetovém prohlížeči. Z textu se také vybere titulek a datum, které budou sloužit k identifikaci a zpřesnění textu.
- **Uložení textu** – dalším krokem programu musí být uložení vydolovaných textů, datumu a popisku do formátu a struktury, která bude přístupná dalším programům, jenž budou text dále zpracovávat. Výstupním formátem programu byl zvolen xml soubor, který bude obsahovat titulek, čistý text a datum. Každý text (stránka) bude uložen do jednoho xml souboru.
- **Získání dalších odkazů** – pokud by bylo potřeba zpracovat pouze jednu stránku, je tento krok zbytečný. Jestliže je ale potřeba stáhnout více stránek při

jednom běhu programu, musí si program průběžně ukládat jednotlivé odkazy, nacházející se na daných stránkách. Podle kritérií pak postupuje na stránky, jejichž odkazy má uložené v databázi.

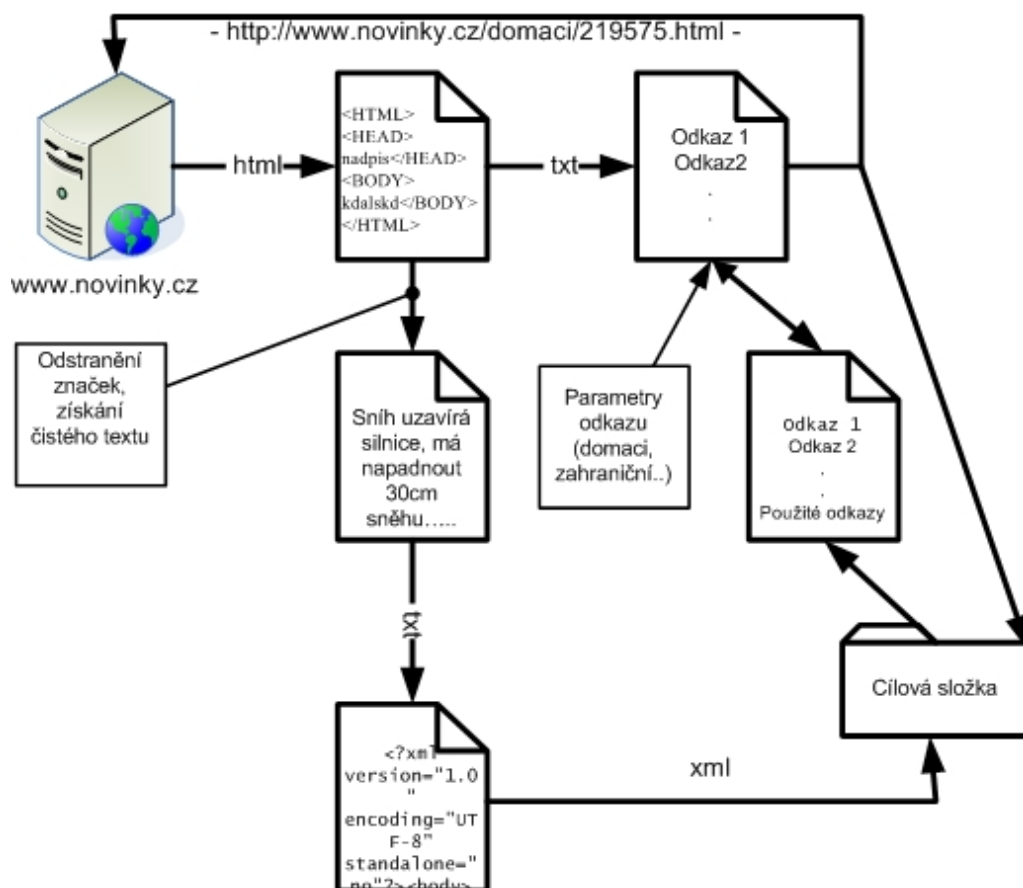
V grafickém prostředí by měl být výběr složky, do které budou ukládány xml soubory a počet stránek, které je potřeba stáhnout.

2.4 Výběr programovacího jazyka

Z dostupných programovacích jazyků bylo vybráno prostředí Java, které díky svým možnostem využívání standartně dodávaných knihoven, převyšuje ostatní programovací jazyky. Pomocí standartních i přiložených knihoven lze jednoduše zpracovat různorodé požadavky (navázání spojení, přenos dat, práce se soubory-vytváření, mazání, načtení a zápis). Další výhodou programovacího prostředí Java je překlad do bytecodu, který je přenositelný mezi platformami. Až při spuštění se přeloží do strojového kódu daného stroje, na rozdíl od jiných programovacích jazyků, které se ihned překládají do strojového kódu. Javu je tedy možné používat na libovolném systému. Z grafického prostředí byla zvolena knihovna uživatelských prvků SWING, která poskytuje všechny potřebné komponenty pro jednoduché grafické rozhraní ovládání programu. Pro grafické prostředí budou potřeba následující položky - tlačítka, textové pole pro specifikaci zadání a textové pole pro zobrazení výsledků.

2.5 Realizace programu a jeho běh

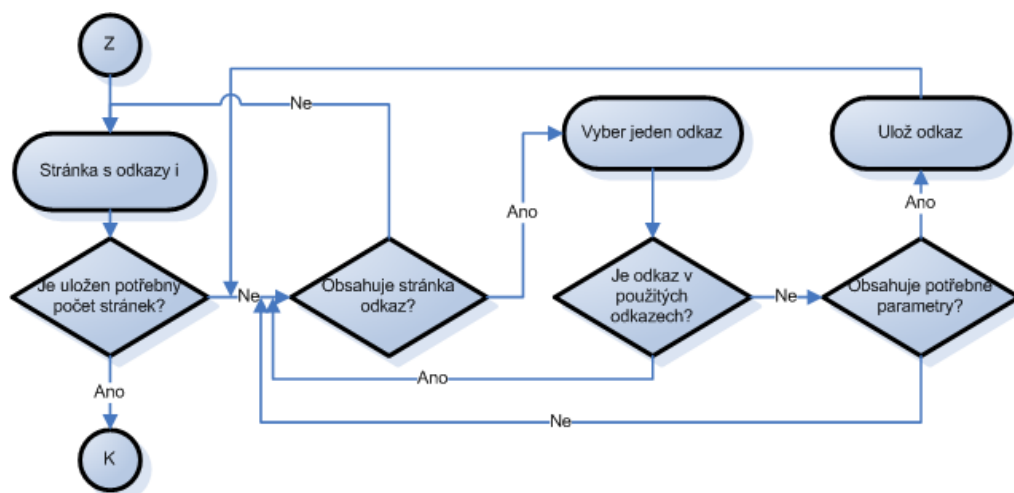
Program byl realizován v programovacím jazyku Java a k jednoduché obslužnosti přispívá grafické uživatelské prostředí. Na obrázku 2.2 je schéma běhu programu, práce se soubory a jeho jednotlivé kroky. Prvním krokem je navázání spojení s webovým serverem, na kterém jsou uloženy požadované internetové stránky pro stažení. Po navázání spojení je stažena konkrétní stránka a uložena do textového souboru, s kterým se dále pracuje. Je důležité, aby soubory byly správně kódovány, neboť při špatně nastaveném kódování dojde ke ztrátě některých znaků například diakritiky.



Obr. 2.2: Schéma běhu programu

2.5.1 Vytvoření databáze odkazů

Prvním krokem, který vykoná program, je vytvoření databáze odkazů, které je možno použít pro další otvírání a stahování jednotlivých stránek. Jestliže jsou parametry vstupu v pořádku, vytvoří se textový soubor, který obsahuje odkazy pro jednotlivé stránky. Vstupními parametry jsou v tomto případě datum, ze kterého se mají stáhnout články a rubriky, ze kterých se mají stáhnout články. Pokud tedy máme celou html stránku převedenou do textové podoby, je možno z ní začít čerpat užitečné informace. Jednotlivé odkazy se kontrolují s odkazy, které jsou v souboru `pouziteOdkazy.txt`, pakliže již tento odkaz byl použit, nezapíše se a je přeskočen. Tento odkaz také musí splňovat určité kritéria, tak aby nedošlo k nežádoucímu přesměrování na jiný než požadovaný web. Všechny tyto odkazy jsou ukládány do textového souboru. Na obrázku 2.3 je zobrazen diagram stahování odkazů. Po skončení běhu programu se tento soubor se všemi odkazy smaže.



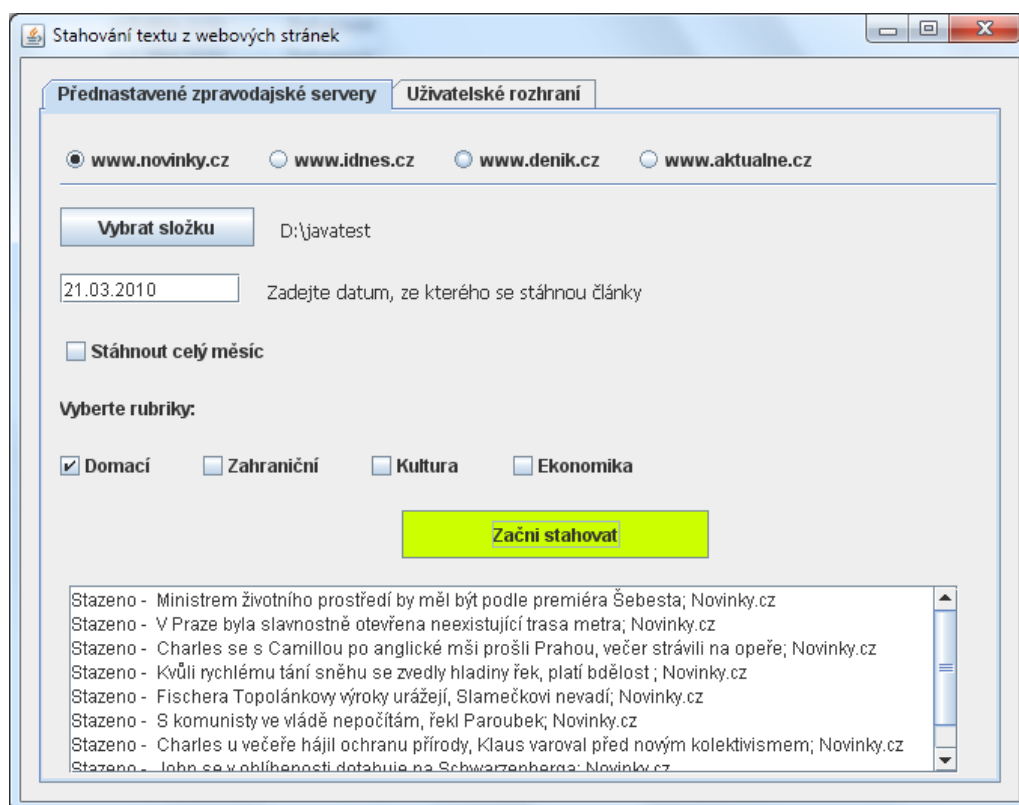
Obr. 2.3: Digram stahování odkazů

2.5.2 Vyselektování požadovaného textu

Dalším krokem programu je třídění textu, kde dochází k vyfiltrování požadovaných textových dat. Nejdříve je vybrán jeden odkaz ze souboru všech odkazů, dojde ke stažení odkazované stránky a html kód je uložen do textového souboru. Z tohoto textového souboru je prvně uložen nadpis stránky, jenž se nachází přímo mezi značkami `<title>` a `</title>` v hlavičce dokumentu, získání této informace tedy není žádný problém. Ani získání datumu nepředstavuje větší problém, protože program je koncipován na konkrétní weby, tedy datum se nachází vždy v konkrétních meta značkách. V případě, že datum je dnešní případně včerejší (zapsáno včera, předečím apod.), je automaticky přepsáno do správného formátu. Problém nastává u získání samotného textu v těle dokumentu. Tento text je většinou různě formátovaný, může obsahovat další netextové informace, jako jsou obrázky, videa, banery a další prvky, které nesou různé značení. Program získává text pouze z těla html dokumentu a odstraňuje zbytek kódu, který je pro další zpracování nepodstatný. Na odstranění jednotlivých prvků značkovacího jazyka, tak aby zůstal pouze text, využívá program databázi nejpoužívanějších značek, které zpracuje podle zadaných pravidel. Tato pravidla pro dolování textu jsou vytvářeny ručně. Pro každý web jsou tato pravidla podobná. Kód obsahuje všechny extrahovací pravidla, která se na daném webu nacházejí.

2.5.3 Uložení textu do xml souboru

Textová data je potřeba uložit do podoby, ve které budou přístupná pro další programy. Soubory xml jsou ukládány do zadané složky a pro každou stránku je vytvořen nový soubor. Soubor obsahuje title, text a datum.



Obr. 2.4: Vzhled programu a ovládací prvky

2.6 Grafické uživatelské prostředí

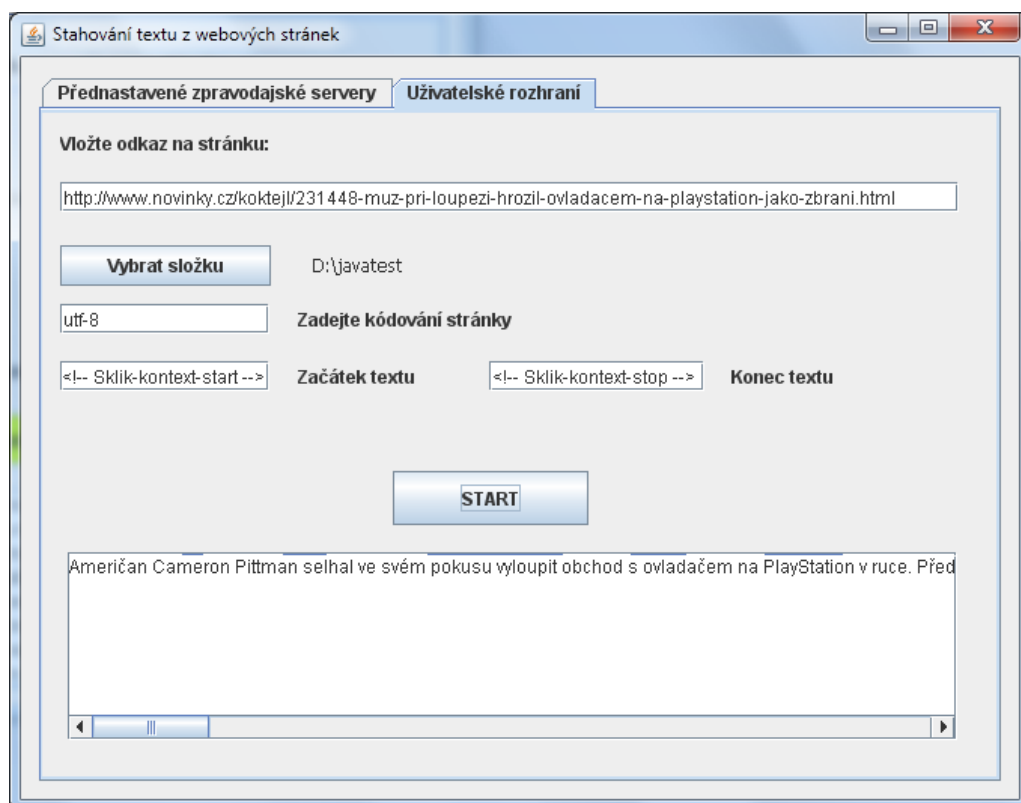
Pro jednoduché nastavení a zobrazení výsledků slouží grafické ovládací prvky. Grafické uživatelské rozhraní bylo vytvořeno pomocí Java knihovny Swing. Na obrázku 2.4 je zobrazen vzhled programu pro přednastavené zpravodajské servery a jednotlivé ovládací prvky pro nastavení dolování textu. Program obsahuje dvě záložky - přednastavené zpravodajské servery a uživatelské rozhraní. V první záložce je možné stahovat texty z přednastavených serverů a v druhé záložce je pak jednoduché univerzální stahování textů, které je však samozřejmě méně přesné a bez možností vytvářet databáze s odkazy. Jak je vidět z obrázku 2.4 jsou zde přednastaveny čtyři

servery - www.novinky.cz, www.idnes.cz, www.denik.cz a www.aktualne.cz. Dalším prvkem je tlačítko na výběr složky (Vybrat složku), do které se budou ukládat všechny xml soubory se staženými texty a také soubor s odkazy na stránky, jenž už byly zpracovány. Informační textové pole napravo od tlačítka zobrazuje, jaká složka byla vybrána. Dalším parametrem je zadání datumu, tento parametr určuje, z jakého data se stáhnou články z webu novinky.cz. V případě dalších serverů není možné vybrat přímo datum článků, je možné zadat pouze počet článků, které se mají stáhnout. Je to z důvodu přístupu na archivované články, kde na www.novinky.cz směřuje přímo odkaz s datem. Pokud by bylo nutné dané texty třídit dle data i na ostatních serverech, generované xml soubory obsahují datum vydání článku. V případě, že je zaškrtnuto políčko "Stáhnout celý měsíc" u novinky.cz, program stáhne celý měsíc, který je zadán v datu. Poslední volbou je výběr jednotlivých rubrik (domací, zahraniční, kultura a ekonomika). Tlačítko "Začni stahovat" zahájí celý proces získávání textových dat. V dolní části textového pole jsou pak při ukončení programu vypsány zpracované stránky. Na obrázku 2.5 je zobrazeno jednoduché univerzální nastavení stahování textu. Je zde výběr složky, kam se stáhne xml soubor s textem, textové pole pro vložení kódování stránky a značky, které označují začátek a konec textu.

2.7 Jednotlivé třídy programu

Program obsahuje pro jednoduchou orientaci patnáct java tříd. Zde je jejich přehled a stručný popis jejich funkce.

- **Hlavní.java** – základní třída, kde se třídí vstupní informace a odesílají se do dalších procesů na zpracování. Pro jednotlivé rubriky jsou přepsány potřebné údaje, tak aby došlo k načtení správného odkazu. Mezi každým stažením stránky je časová prodleva, která zajistí, aby nedošlo k odpojení z důvodu častého přístupu na server.
- **NacteniSouboru.java** – pomocí této funkce se obsah souboru načte do proměnné, program tedy pracuje s textovými soubory. Načítá se například jak soubor se staženým html kódem, tak databáze odkazů a použitých odkazů.
- **ZapisDoSouboru.java** – tato třída umožňuje zapisovat proměnné do souboru. Opět pracuje pouze s textovými soubory a zapisuje databázi odkazů a použité



Obr. 2.5: Vzhled programu pro uživatelské nastavení

odkazy.

- `Zkontrolovani_Souboru.java` – v této třídě dojde ke kontrole, zda-li existuje soubor, s kterým chceme dále pracovat. V případě, že soubor neexistuje, automaticky se vytvoří.
- `Stazeni_Stranky.java` – vstupními parametry jsou adresář, kódování a odkaz. Tato funkce umožní stažení stránky ze zadaného odkazu se správným kódováním do zadaného adresáře. Html kód je uložen do textového souboru.
- `Vytvoreni_odkazu_*server.java` – pro každý server jsou sestaveny parametry pro stažení požadovaného počtu nebo podle data stránek, které jsou uloženy a následně jeden po druhém staženy a dále zpracovávány.
- `Orezani_*server.java` – zde dochází ke konečnému očištění textu od všech nežádoucích prvků, výsledkem je tedy text bez reklam, částí menu nebo odkazů. Pro každý server jsou parametry vytvořeny ručně.
- `Uzivatel.java` – tato třída obsluhuje uživatelské rozhraní aplikace.
- `Vytvoreni_xml.java` – Tato funkce uloží všechny stažené texty, datумы a

nadpisy do souboru xml, dle zadané struktury a adresáře.

2.8 Test programu

V tabulce 2.1 jsou zobrazeny naměřené časy potřebné pro stažení 200 stránek pro každý zpravodajský server. Časy jsou uvedeny i s čekacím časem, který se spustí po každém stáhnutí stránky tak, aby nedocházelo k zatěžování serveru a případného zamezení přístupu na server. Mezi jednotlivými měřeními je rozptýl průměrně 20 sekund, což mohlo být způsobeno různou velikostí jednotlivých stránek, čekacími dobami na odpověď serveru a vytížením procesoru. V tabulce jsou uvedeny servery, které nejsou nakonfigurovány v grafickém prostředí. Byly vytvořeny za účelem přesnějšího zhodnocení délky běhu programu. Vyselektovaný text z těchto serverů je přiložen v přídatném souboru.

server	Potřebný čas na stáhnutí 200 stránek [s]			
číslo měření	1	2	3	4
www.idnes.cz	150	140	180	160
www.denik.cz	210	190	190	180
www.aktualne.cz	170	160	165	180
www.sport.cz	130	140	140	160
www.ihned.cz	170	150	160	180
www.ceskenoviny.cz	190	180	210	180

Tab. 2.1: Čas potřebný pro stažení a vyfiltrování 200 stránek

Každá rubrika byla otestována na maximální počet stránek pro každý server zvlášť. Během tohoto stahování nedošlo k žádnému zacyklení ani chybě běhu. Náhodně byly také zkontrolovány textové výstupy u několika stránek. Zkontrolované textové výstupy neobsahují žádné znaky, které by byly nežádoucí nebo je neobsahoval text. Není ale vyloučeno, že se objeví prvky, které obsahují pouze jednotlivé stránky, které zůstanou v textu jako nedůležité. Program neobsahuje kontrolu chyb proti zacyklení. Při vytváření pravidel pro jednotlivé servery docházelo k chybě nejčastěji v třídě `Vytvoreni_odkazu.*server.java`, kdy byly zadány chybné parametry hledaného odkazu a odkazu, který obsahuje tyto odkazy. V této třídě probíhá

hledání jednotlivých odkazů do doby, dokud je nalezený počet menší než požadovaný. V případě, že je zadán špatně jeden z odkazů, počet odkazů nikdy nedosáhne požadovaného počtu. Ochranou by mohlo být například n-té opakování běhu a následné zastavení běhu programu. Jestliže datum nebo vyselektovaný text neobsahuje žádná data, vytvoří se pouze prázdný xml soubor.

3 ZÁVĚR

První část semestrálního projektu se zabývá teorií o získávání textových dat z internetových stránek. Je zde stručně nastíněn princip získávání dat, který je obecnější formou dolování textových informací. Z teorie získávání textových dat jsou vypsány jednotlivé možnosti třídění dokumentů dle různých kritérií a získávání dat ze strukturovaných dokumentů. Další část se zabývá wrapappery, které slouží k získávání dat z konkrétních informačních zdrojů. Závěr teoretické části obsahuje přehled nejzajímavějších a nejpřesnějších programů na získávání dat z webových stránek.

V praktické části byl realizován program s grafickým rozhraním knihovny SWING v programovacím prostředí Java. Bylo zvoleno ruční nastavení jednotlivých parametrů pro čištění textu staženého z jednotlivých stránek, z důvodu vysoké kvality takto vyselektovaného textu a maximální omezení chyb, tak jak potvrdily výsledné testy programu. Základ vytěžení textu je však stejný pro každý server - z html kódu je vyselektován text mezi obálkami (různé pro každý server - vkládány ručně), následně dojde k výběru textu mezi jednotlivými znaky, uvozujícími odstavce. Tímto je získán text, který však obsahuje nesouvislé znakové části, jako jsou reklamy, menu apod. Na odstranění těchto dat jsou do programu vložena ručně vytvořená pravidla. Program umožňuje stahování článků ze čtyř serverů (novinky.cz, idnes.cz, denik.cz a aktualne.cz). Je možné si zvolit, ze kterých rubrik se dané články mají stáhnout a jejich počet. Jednotlivé odkazy jsou kontrolovány s databází již uložených odkazů, je tedy možné kontinuálně stahovat neduplicitní texty. Výstupem jsou pak xml soubory, které obsahují nadpis, text a datum vydání článku.

LITERATURA

- [1] ATKINSON, J.A. *Institute for Communicating and Collaborative Systems* [online]. 2003 [cit. 2010-12-15]. Text Mining: Principles and applications. Dostupné z URL: <<http://labs.rightnow.com/colloquium/papers.php>>.
- [2] BŘÍZA, Petr. *interval.cz* [online]. 2003 [cit. 2011-05-11]. Znakové sady v praxi - UTF-8 Dostupné z URL: <<http://interval.cz/clanky/znakove-sady-v-praxi-utf-8/>>.
- [3] *BST download* [online]. 2004 [cit. 2010-12-17]. Dostupné z URL: <<http://bstdownload.com/reviews/happy-harvester-2/>>.
- [4] Gottron, T., Martin, L.: *Estimating web site readability using content extraction*. [online]. 2009 [cit. 2010-12-15] Dostupné z URL: <www2009.org/proceedings/pdf/p1169.pdf>.
- [5] GRIMES, Seth.. *ClaraBridge* [online]. 2009 [cit. 2010-12-15]. BridgePoint Article. Dostupné z URL: <<http://www.clarabridge.com/default.aspx?tabid=137&ModuleID=635&ArticleID=551>>.
- [6] JANOVSKEÝ, Dušan. *Jak psát webe* [online]. 2008 [cit. 2011-05-11]. Čeština / cestina. Dostupné z URL: <<http://www.jakpsatweb.cz/cestina.html>>.
- [7] KOPÁČKOVÁ, Hana. *Dspace.upce.cz* 2008 [cit. 2010-12-15]. MANAŽERSKÉ ROZHODOVÁNÍ ZA VYUŽITÍ METOD PRO ZPRACOVÁNÍ DOKUMENTŮ. Dostupné z URL: <http://dspace.upce.cz/bitstream/10195/35140/1/KopackovaH_Manazerske20rozhodovani_VS_2006.pdf>.
- [8] KUČERA, Miroslav. *Základní struktura dokumentu* [online]. 1999 [cit. 2010-12-17]. Miroslav Kučera. Dostupné z URL: <<http://webdesignledger.com/tips/the-most-common-html-and-css-mistakes-to-avoid>>.
- [9] KUSHMERICK, Nickolas, WELD, Daniel S., DOORENBOS, Robert B. Doorenbos. *In Intl. Joint Conference on Artificial Intelligence* [online]. 1997 [cit.

- 2010-12-15]. Wrapper Induction for Information Extraction. Dostupné z URL: <<http://citeseer.ist.psu.edu/kushmerick97wrapper.html>>.
- [10] LIU, Bing. *Web Data Mining*. Amazon.com : Hardcover , 2006. 236 s.
- [11] NEKVASIL, Marek. *VYUŽITÍ ONTOLOGIÍ PŘI INDUKCI WRAPPERU*. Praha, 2006. 58 s. Diplomová práce. Vysoká škola ekonomická v Praze.
- [12] NOACK, Shannon. *Webdesignledger* [online]. 2011 [cit. 2010-12-17]. The Most Common HTML and CSS Mistakes to Avoid. Dostupné z URL: <<http://webdesignledger.com/tips/the-most-common-html-and-css-mistakes-to-avoid>>.
- [13] SEDLÁČEK, Petr. *Www.fi.muni.cz* [online]. 2005 [cit. 2010-12-15]. Text mining a jeho možnosti. Dostupné z URL: <<http://www.fi.muni.cz/usr/jkucera/pv109/2003p/xsedlac5.htm>>.
- [14] *SmElis* [online]. 2006 [cit. 2010-12-17]. Dostupné z URL: <<http://www.smelis.com/>>.
- [15] SNAJDR, Petr. *Čeština* [online]. 2006 [cit. 2011-05-11]. Proč právě ISO-8859-2?. Dostupné z URL: <<http://www.cestina.cz/whyISO.html>>.
- [16] Srivastava, Ahok N. *Text Mining*. NW, 2009. 279 s. Classification, Clustering and Application.
- [17] *Tvorba-Webu.cz* [online]. 2008 [cit. 2010-12-15]. DOM: Document Object Model. Dostupné z URL: <<http://www.tvorba-webu.cz/dom/>>.
- [18] *W3C* [online]. 1994 [cit. 2010-12-17]. Dostupné z URL: <<http://www.w3.org/>>.
- [19] *W3C* [online]. 1999 [cit. 2010-12-15]. XML Path Language (XPath). Dostupné z URL: <<http://www.w3.org/TR/xpath/>>.
- [20] *W3SCHOOLS* [online]. 2006 [cit. 2010-12-15]. HTML DOM Introduction. Dostupné z URL: <http://www.w3schools.com/html/dom/dom_intro.asp>.
- [21] *Web-Harvest* [online]. 2006 [cit. 2010-12-15]. Dostupné z URL: <<http://web-harvest.sourceforge.net/>>.

- [22] WITTEN, Ian H. *Computer Science* [online]. 2006 [cit. 2010-12-15]. Text mining. Dostupné z URL: <people.ischool.berkeley.edu/hearst/text-mining.html>.

SEZNAM SYMBOLŮ, VELIČIN A ZKRATEK

KDD vyhledávání znalostí – knowledge discovery

TM dolování v textu – text mining

LR left-right – zleva-doprava

HLRT head-left-right-tail – hlavička-zleva-doprava-tělo

OCRL opening-closing-left-right – začátek-závěr-zleva-doprava

HOCLRT head-open-close-left-right-tail – hlavička-záčátek-závěr-zleva-doprava-tělo

DOM document object model – dokumentový objektový model

SAX simple API for xml – jednoduché programové rozhraní pro xml

XML extensible markup language – rozšířený značkovací jazyk

HTML hypertext Markup language – značkovací jazyk pro hypertext

BSD Berkeley Software Distribution

GUI graphical user interface – grafické uživatelské rozhraní

SEZNAM PŘÍLOH

A	CD se zdrojovými kódy	43
---	-----------------------	----

A CD SE ZDROJOVÝMI KÓDY

- spustitelný program *.java
- projekt se zdrojovými kódy
- databáze textů
- Diplomová práce